Signal Compression Methods for Low-Power Implants

by

Russell Dodd

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

INTEGRATED CIRCUITS AND SYSTEMS

Department of Electrical and Computer Engineering
University of Alberta

Abstract

The pairing of implantable micro-electrode arrays with micro-electronic devices allows the observation of neural activity within nervous systems in a more comprehensive and potentially more effective way compared to single electrode recording. State-of-the-art tethered recording systems use implanted micro-electrode arrays and can record raw data rates of at least 24 Mbps over 100 channels. To make the systems fully implantable, the communication link needs to be wireless with the system power dissipation low enough for battery and/or near-field wireless operation. This thesis details a study of signal compression methods intended for implanted wireless low-power neural signal recording implementations. ASIC simulations of compression methods are presented for proof of concept and comparisons, while ASIC implementations provide power consumption measurements for select methods. It is shown by using a windowed noise-based dual-threshold neural spike detector that an energy savings of 90% can be expected compared to systems with no compression. Furthermore, an additional 80% compression per spike can be achieved using multilevel wavelet decomposition with no effect on the classification accuracy and relatively small increase in power consumption. Finally, a feature extraction method is presented and is simulated to demonstrate a neural spike compression ratio of 87% and classification accuracy of 96%.

Preface

This thesis is an original work by Russell Dodd. Some of the research conducted for this thesis forms part of a research collaboration, led by Dr. Vivian Mushahwar at the University of Alberta with the AHFMR ProjectSMART Team. The team website can be found at smartneuralprostheses.med.ualberta.ca.

Chapters 4, 5, and 6 review and expand upon original work that has been published in the following papers.

R. Dodd, B.F Cockburn, V. Gaudet, "Adaptive dual-threshold neural signal compression suitable for implantable recording," pp. 8401 - 8405, ICASSP 2014, Florence, Italy, May 2014.

R. Dodd, B.F Cockburn, V. Gaudet, "Neural spike compression using feature extraction and a fuzzy c-means codebook," 5 pages, ISMVL 2014, Bremen, Germany, May 2014.

R. Dodd, B. Crowley, V. Gaudet, V. Mushahwar, B. Cockburn, "Microelectronics for in-vivo neural recording," IFESS 2012, September 2012.

R. Dodd, B. Cockburn, V. Gaudet, "Two compression techniques for neural recording," HCDC Banff Summer School 2011, August 2011.

Dedicated to my parents.

Acknowledgements

I would like to thank my supervisors Dr. Vincent Gaudet and Dr. Bruce Cockburn for the opportunity to work on this project and for their leadership. I learned a lot. I would also like to thank Dr. Vivian Mushahwar and Dr. Richard Stein for their insight into the world of neuroscience. It has been a mind-expanding experience.

In addition, a thank you to the members of ProjectSMART, whose thoughtfulness and dedication are inspiring to me: Andrea Jeffery, Rahul Samant, Rui Zhou, Dirk Everaert, Kenton Hamaluik. You all made the team meetings so memorable and fun. It was a roller coaster ride of research excitement!

I would like to thank my lab-mates Brendan Crowley, Jinghang Liang, David Sloan, Andrew Hakman, John Koob and Caitlin Davis for their intuition and helpfulness. There have been too many wonderful discussions to count and I hope for many more.

Finally, I would like to thank my wife for her love and support.

Contents

List of Tables

List of Figures

List of Acronyms

ADC analog-to-digital converter

ASIC application-specific integrated circuit

BMI brain-machine interface

CMOS complementary metal-oxide semiconductor

CPL complementary pass-gate logic

DCT discrete cosine transform

DRG dorsal root ganglion

DSP digital signal processor

DFT discrete Fourier transform

DWT discrete wavelet transform

DPCM differential pulse-coded modulation

ECG electrocardiography

FCM fuzzy c-means

FIFO first-in first-out

FIR finite impulse response

FPGA field-programmable gate array

FSK frequency shift-keying

LFSR linear-feedback shift register

LNA low-noise amplifier

MEA micro-electrode array

MSB most significant bit

MSE mean squared error

NEO non-linear energy operator

OOK on-off keying

PCA principal component analysis

RF radio frequency

SER spike error rate

SIP system-in-package

SOC system-on-chip

SMART sensory motor adaptive rehabilitation technology

SNR signal-to-noise ratio

SPI serial peripheral interface

SRAM static random-access memory

UWB ultra-wide-band

VLSI very large scale integration

VHDL very-high-speed-integrated-circuit hardware description language

Chapter 1

Introduction

The human nervous system is complex and is not yet well understood. The continuing development of neural interfaces for rehabilitation and analysis of the nervous system is needed to further the state-of-the-art for neuro-prostheses and brain-machine interfaces (BMIs) [9]. The development of implantable micro-electrode arrays (MEAs) has created an opportunity to study nervous systems *in vivo*, with large groups of neurons accessed simultaneously by implanted microelectronic recorders and stimulators. A generic neural recording interface is shown in Figure 1.1.

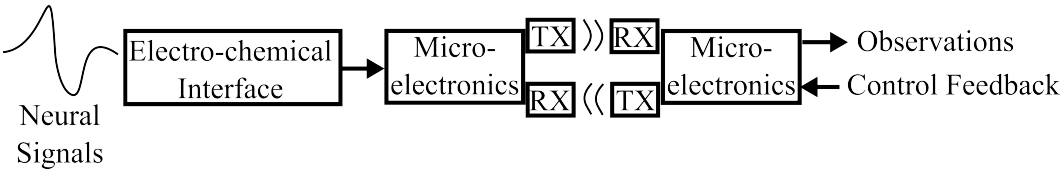


Figure 1.1: A generic implantable wireless neural recording system.

The microelectronics of the recording interface rely on the computational and energy efficiencies available from low-power transistor technologies. The advancement of silicon processing technologies, which still follows Moore's law, provides more transistors per unit area and less power per bit with each technology generation [10]. Design strategies that reduce energy consumption are what make implantable microelectronics practical. The aim of this work is to exploit the computational capabilities of application-specific integrated circuits (ASICs) and system-on-chips (SOCs) to enable implantable neural recording for nervous system rehabilitation and study.

The development of *in vivo* wireless electronics has been limited by small power supplies (10's of mW) and low heat dissipation restrictions [11,12]. These limitations can be avoided by moving any energy-expensive processing to an *ex-vivo* base station processor, where power and heat are no longer severe limiting factors. However, the wireless transmission of data from the implant to the base station is a costly energy operation compared to baseband signal processing. If the data rate is too large for the power constraints of the implant then additional processing must be performed to reduce the data rate, with minimal loss in signal content, and therefore reduce the power consumption of the device. A key caveat is that the additional signal processing must save more energy, by reducing the data rate, than the energy cost of that operation. This relation is expressed in Equation 1.1 and is the motivation for including signal compression into implanted wireless electronics that have large raw data rates.

$$\frac{E_{tx}}{\text{bit}} \geq \frac{E_{tx}}{\text{bit}} CR + \frac{E_c}{\text{bit}} \tag{1.1}$$

A baseband symbol encodes a discrete quantity of signal information, such as a portion of a digitized measurement of a signal amplitude at a particular sampling time. In practice, a symbol represents a group of one or more binary digits. CR denotes the compression ratio which is the ratio of the number of symbols out and the number of symbols in. So $CR = \frac{1}{4}$ would mean that for every 4 symbols input to the compression module only 1 new symbol is produced. E_{tx} is the energy required to transmit any symbol wirelessly and E_c is the energy required by the compression operation to attain the CR. The energy of a system with no compression is E_{tx}, while the energy of a system with compression is reduced by the compression ratio CR, but simultaneously increased by E_c. The focus for the compression system is to keep the power as small as possible and to understand the tradeoffs of algorithms and architectures with respect to CR vs. E_c as applied to the recording of neural signals from multi-electrode arrays. This means reducing the CR at the expense of increasing E_c, while keeping enough signal fidelity for base station processing.

Investigating implementations of real-time low-power lossy compression for neural signals is the subject of this thesis. The resulting simulations and implementations are part of a larger design goal to develop a wireless implantable neural recording system for the sensory motor adaptive rehabilitation technology (SMART) Project at the University of Alberta. The SMART project team's goal is to restore motor function to people with irreparably damaged nervous systems [13, 14]. The SMART Team's current implantable recording technology enables real-time tethered neural signal analysis in animals, but is unable to provide portable untethered subject experimentation. The development of the wireless neural implant is being coordinated with multiple groups from multiple disciplines to enable untethered weight-bearing subject analysis towards a mobile method of neural signal rehabilitation/intervention. One group that directly interacts with the development of the microelectronics include members from neuroscience which operate the experiments to obtain insight into walking mechanisms and neural pathways in order to develop rehabilitation interventions/prosthetics. In addition, the microelectronics need a biocompatible interface that members from material engineering are working on for the neural electrodes and for encapsulating the microelectronics. The neuroscientists provide the constraints and desired functionality in which the microelectronics must operate. The constraints include a small physical volume with low heat dissipation while recording neural signals within a noisy electro-chemical environment.

The neural recording system is split into a front-end analog/digital neural signal processor and wireless transmitter. The choice of wireless communication scheme led to implications on the front-end processor. The wireless transmitter must send data through several centimetres of tissue, a layer of skin, and several meters of free space. This means that the energy required for transmitting a data bit is greater than the energy required for signal processing operations. Therefore compression should reduce the overall energy consumption of the neural recording system. The recording system's signal processing is split into an analog front-end and digital processor for neural signal compression. The overall purpose of the system evolved over time during the thesis; from spike detection and continuous recording to later include spike sorting-like operations. The overall goal was to provide the functionality to observe the neural signal at different levels of compression so that the user can be confident in the spike detection and processing being provided in the implanted microelectronics. This idea can be thought of as a *neural oscilloscope*.

The rest of this thesis is organized as follows. Chapter 2 reviews the characteristics of neurons and their firing mechanisms. The state-of-the-art in neural interfaces for recording and stimulating neural activity is reviewed. The concept of spike sorting is explained and a practical example is presented. The chapter is concluded with observations on the constraints and work needed on neural interfaces. Chapter 3 presents and evaluates examples of compression on recordings of neural signals. Important metrics of compression performance are discussed with respect to the algorithm. The chapter concludes with a discussion of the

simulated performance of the compression algorithms and the choice of further work in the proceeding chapters. Chapter 4 presents the detailed simulation and implementation of the dual-threshold spike detection circuit. The circuit performance is compared to the state-of-the-art and measured results from a fabricated chip are presented. Chapter 5 presents the detailed simulation and implementation of the discrete wavelet transform, which can operate in tandem or independently of the dual-threshold spike detector. Simulated and measured results from a fabricated chip are presented and compared to the corresponding state-of-the-art performance figures. Chapter 6 presents the simulation and implementation of a feature extraction based compressor that must operate with a spike detector. Post-layout simulated results are stated and compared to the state-of-the-art. Chapter 7 concludes the thesis with a review of the conclusions of each previous chapter.

Chapter 2

Review of Neural Signal Recording

The aim of this chapter is to review the properties of the neuron, its communication mechanisms and artifical electro-chemical interfaces.

2.1 Neurons

This section reviews relevant terminology and the key properties of neurons. A neuron is a cell that transmits information by electrical and chemical signalling. There are an estimated 10^{12} neurons in the human nervous system [1]. A simplified drawing of a neuron is shown in Figure 2.1. The neuron is composed of three parts: the cell body, the dendrites and the axon. In Figure 2.1, the axon is mostly coated in sections by a myelin sheath; the small uncoated sections of axon are called Nodes of Ranvier. The length of a neuron can vary from a couple of micrometers to 1-2 meters. A neuron communicates to another neuron through a small gap region called a synapse. There are an estimated 10^{15} synapses in the human brain [1]. Most neurons are polarized; they receive synaptic input at the dendrite and make synapses at the axonal end. The number of connections made to a single neuron can be very large. A dendrite of a mammalian motor neuron can receive inputs from about 10^4 synapses [1]. A large number of neurons make feedback synapses into their pre-synaptic cells and can even form serial feedback loops comprised of several neurons. The complex network of connections and the small size of neurons is what makes their study difficult. Most early studies on neurons were performed on the unusually large giant squid axon [15], which enabled the study of the the physiological processes involved in creating action potentials. The recent development of micro-electrodes has enabled the study of large populations of micrometer-sized neurons.

2.2 Action Potentials and the Hodgkin-Huxley Model

A neural signal pulse travelling along an axon is known as an action potential or spike and has measurable electrical and chemical properties. The signals are primarily carried by transmembrane ion currents composed of the disassociated ions sodium Na^+, potassium K^+, calcium Ca^{2+}, and chloride Cl^-. The movement of these ions is described by physical laws of diffusion, drift and mobility, which is why the neuron is often compared to an electrical circuit. Neurons and wires both transmit information by using potentials (differences in accumulated charges) and currents (movement of charge). Therefore standard circuit theories, like Kirchhoff's Law, can be applied to calculate the potentials and currents within

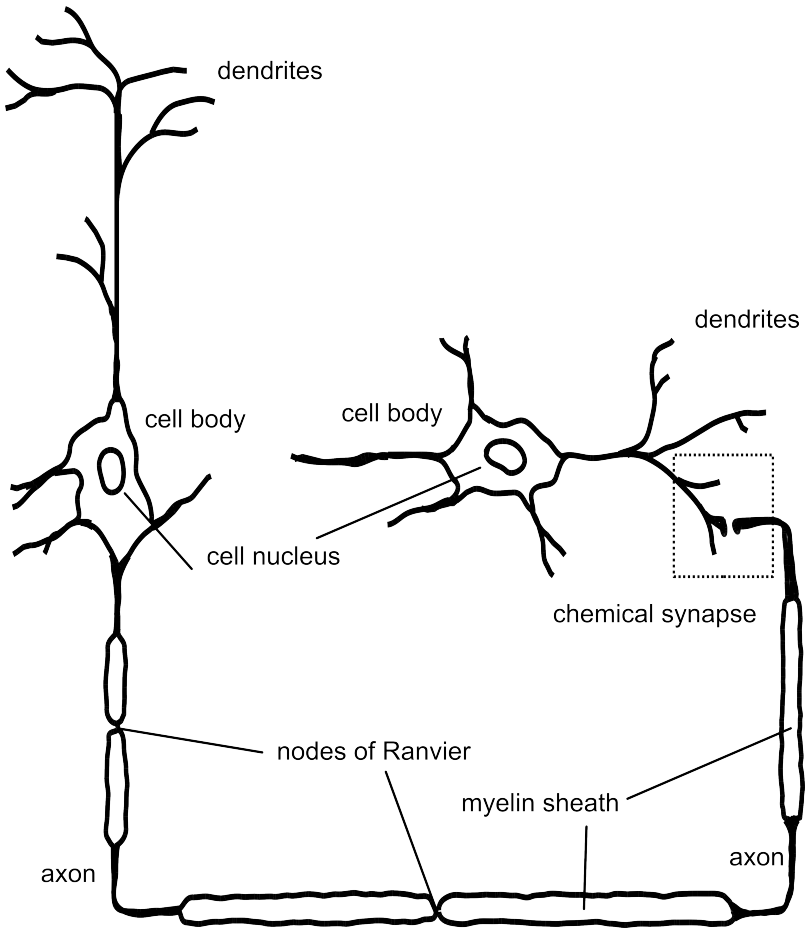


Figure 2.1: A neuron is composed of a cell body with connectors called dendrites and axons. They can physically or chemically connect to other neurons through synapses. Most of an axon is covered by an insulating myelin sheath that speeds up signal propagation. The Nodes of Ranvier are short unmyelinated regions on the axon in between each myelinated section [1].

the neuron. An extracellular recording of an action potential has the characteristic triphasic shape shown in Figure 2.2. The first phase experiences a relative increase in positive potential as the spike approaches and in the second phase the neuron membrane depolarizes as the spike travels past the recording point. The third phase captures the membrane settling from the spike passing. The shape arises from ion movements across the neuron membrane and is the reason why action potentials are usually called spikes.

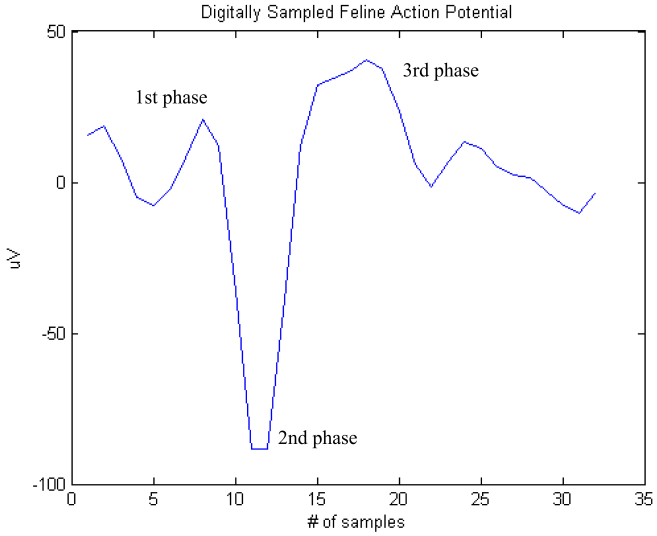


Figure 2.2: Matlab reproduction of feline DRG recording sampled at 30 kSps, after bandpass filter stage (250 Hz - 7.5 kHz) in the software suite Cerebus (courtesy of Drs. V. Mushahwar and R. Stein, University of Alberta).

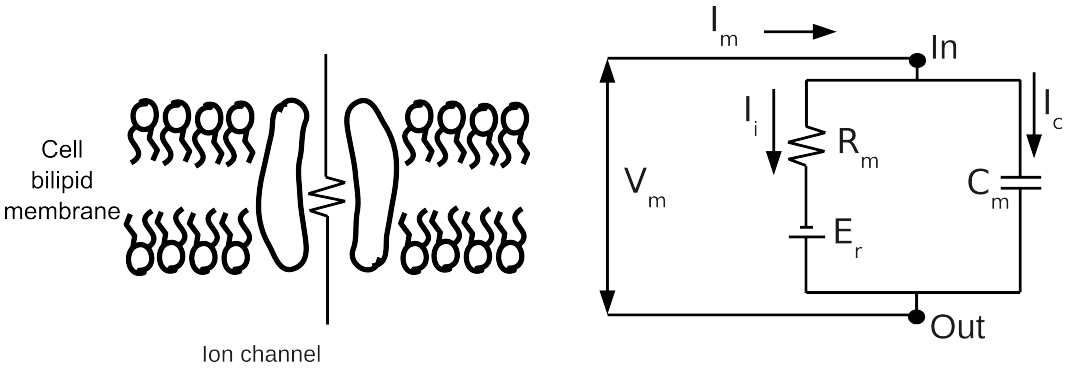


Figure 2.3: A simplified model of the neural membrane and its representation as a circuit schematic [1].
I_m - total current flowing across a region of the membrane
V_m - membrane potential
I_i - ionic current
I_c - capacitive current
C_m - specific membrane capacitance
R_m - specific membrane resistance
E_r - resting potential of the cell

Figure 2.3 shows two schematics for a neuron membrane from [1]. Ions in a resting system are non-uniformly distributed and act like a DC potential difference, E_r, that drives

cross-membrane currents. The power required to sustain these currents comes from cellular metabolism. The cross membrane current is described by Equation 2.1 with $G_m = \frac{1}{R_m}$ and where the Thevenin equivalent circuit values are $V_{th} = E_R \left(1 - \frac{R_m s C_m}{R_m s C_m, -1}\right)$ and $R_{th} = \frac{R_m}{1+sC_m R_m}$. This implies that at higher frequencies the output impedance of the membrane is reduced. However, a high frequency event only occurs during an action potential, which requires a change in the conductances of the model. When an action potential arrives at a segment of the membrane, the model must be revised to account for the changes in ion gradients and channel conductances. The excitation and conduction of axons can be described by the Hodgkin-Huxley model for parallel conductances [1]. This model is similar to Cable Theory and is shown in Figure 2.4. The conductances due to the major ions are shown on the left in the non-propagating model and then the resting potentials and conductances are lumped into r_m for the propagating model. In addition, r_i models internal resistance to the action potential propagation.

$$I_m = C_m \frac{dV_m}{dt} + G_m \left(V_m - E_r\right) \tag{2.1}$$

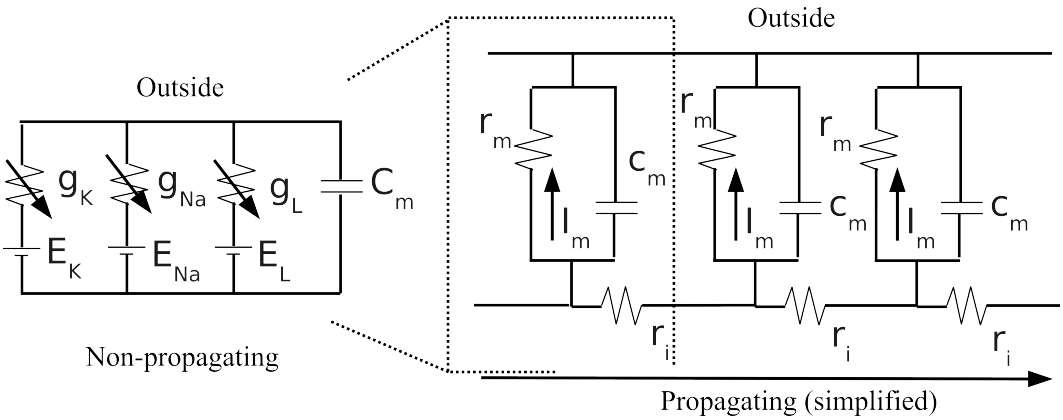


Figure 2.4: Hodgkin-Huxley parallel conductance model for the squid axon alongside a simplified view of a neural axon using Cable Theory [2–5].
g_L - constant for modelling the effects of leakage
g_K - potassium conductance
g_{Na} - sodium conductance

The Hodgkin-Huxley model is the result of four principles that dictate the movement of ions [1–5].

Principle 1) Fick's Law: Ions flow from a high concentration zone to a low concentration zone.

$$\boldsymbol{J}_{diff} = -D \frac{\partial [C]}{\partial \boldsymbol{c}}$$

Here $\boldsymbol{J}_{diff}$ is the diffusion flux (molecules/s-cm^2), D is the diffusion coefficient (cm^2/s), $[C]$ is the concentration of ions (molecules/cm^3), and $\boldsymbol{c}$ is the position.

Principle 2) Ohm's Law for drift: Ions drift through the medium in response to forces caused by electric fields acting on the charge of the ions. Current flow is limited by the resistance experienced by the ions interacting with the conducting medium.

$$\boldsymbol{J}_{drift} = \partial_{el} \boldsymbol{E}$$

$\boldsymbol{J}_{drift}$ is the drift flux (molecules/s-cm^2), ∂_{el} is electrical conductivity (molecules/V-s-cm), and $\boldsymbol{E}$ is the electric field (V/cm) .

Principle 3) Einstein's relation between diffusion and mobility: Diffusion and drift processes in the same medium are additive.

$$D = \frac{kT}{q}\mu$$

where D is the diffusion coefficient, k is Boltzmann's constant (1.38 x 10^{-23} J/ K), T is absolute temperature (K), q is the charge of the molecule (C), and μ is mobility (cm^2/V-s).

Principle 4) Space-charge neutrality: In a given volume the amount of positive charge is equal to the amount of negative charge. In transient situations such as across the neuron membrane, space-charge neutrality might not hold.

The Hodgkin-Huxley model was developed for the giant squid axon and it remains the basis for more advanced models such as the stochastic Hodgkin-Huxley model, which uses a stochastic approach instead of a deterministic approach to modelling the activation of an ion channel [16]. In these models, statistical and probabilistic techniques are used to predict action potentials based on the currents and conditions of the ion channels in the membrane. Parameters can be adjusted based on neural noise and currents in the system [17] to give a usable neural model that is accurate enough to give reasonable signal shape predictions. A recent model from [7] is able to replicate 20 behaviours of the Izhikevich dynamical neuron model [6], while only consisting of addition and multiplexing logic units. The twenty behaviours are: tonic spiking, phasic spiking, tonic bursting, phasic bursting, mixed mode, spike frequency adaptation, Class 1 excitable, Class 2 excitable, spike latency, subthreshold oscillations, resonator, integrator, rebound spike, rebound burst, threshold variability, bistability, depolarizing after-potential, accommodation, inhibition-induced spiking, and inhibition-induced bursting. These behaviours are shown in Figure 2.5 from [6].

The propagation of an action potential along an axon generally occurs as shown in Figure 2.6. At instant 1 a synaptic potential triggers an action potential at point X. This causes current to flow as indicated. The observation point P, outside the membrane, sees an increase in positive ion density. At instant 2 the action potential depolarizes the membrane as it travels along the neuron. Point P sees a large increase in negative ion density as the action potential passes by. At instant 3, the membrane near point P is again positive after the spike has passed.

An action potential can travel along an axon at a speed of 1 m/s to over 100 m/s and the velocity generally increases with the axon diameter. The myelin sheath inhibits the movement of ions and causes faster propagation in an effect called saltatory conduction [1]. Therefore it is typical to try and record from the cell bodies rather than the axons. The shortest amount of time between action potentials along the same axon, must be greater than the refractory period, which is caused by the *hyperpolarization* of the membrane. This period can range from 1 ms to 5 ms.

Neurons can be classified based on their discharge patterns, as shown in Figure 2.5, and by their function. *Afferent neurons* convey information from tissues and organs into the central nervous system (sensory), *efferent neurons* transmit signals from the central nervous system to the effector cells (motor), and *interneurons* connect neurons within regions of the central nervous system.

2.3 Neural Interfaces

Neural action potentials change the concentrations and charges of the space or fluid around neurons, therefore it is possible to record spikes without being in contact with the neuron by measuring the potential of the surrounding extracellular fluid. This also means spikes can be stimulated with μA currents and recorded in terms of an electrical potential in micro-volts, μV. This electrochemical interaction allows electronics to be interfaced with neuron(s).

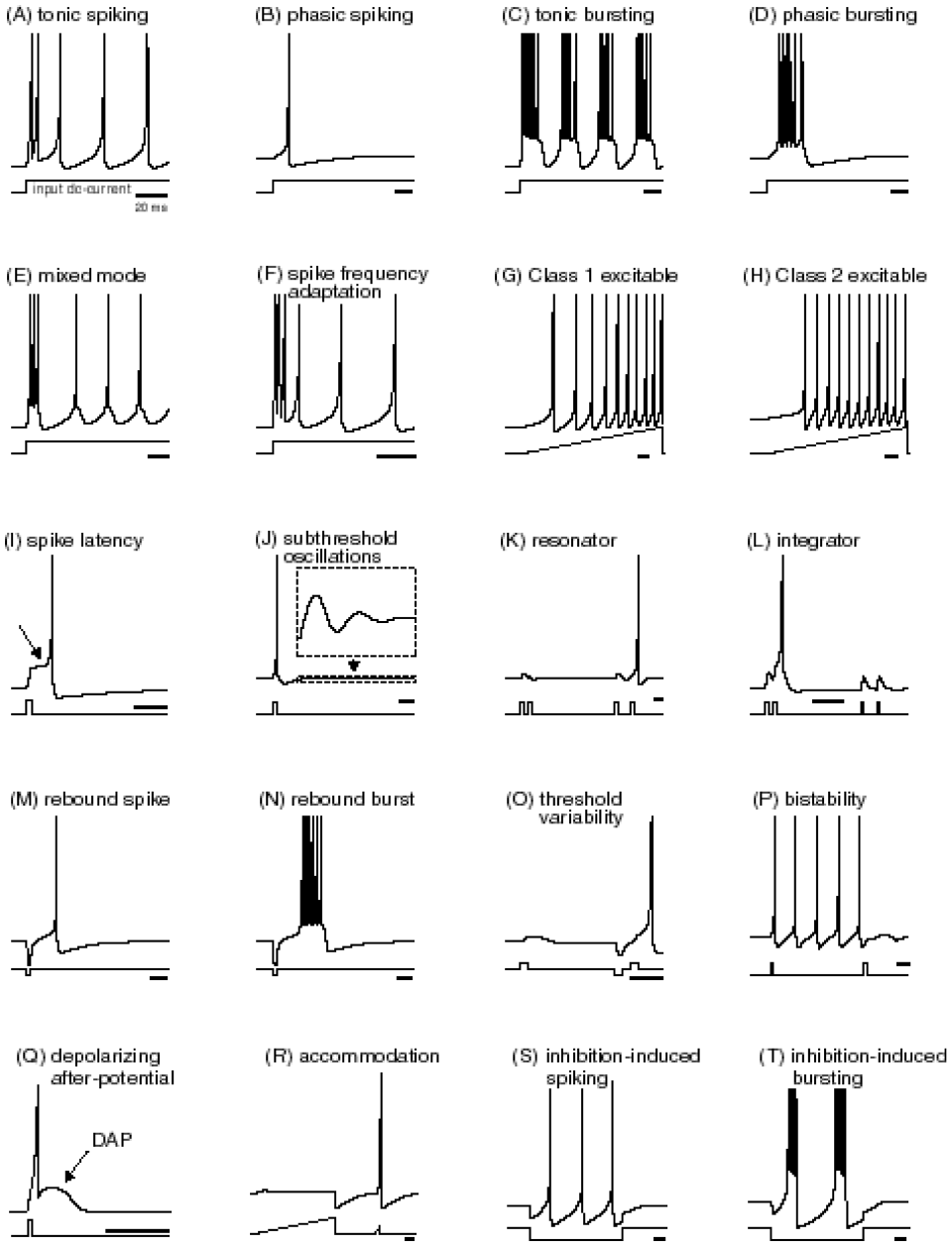


Figure 2.5: The 20 dynamical neuron behaviours described in [6, 7]. An electronic version of the figure and reproduction permissions are freely available at www.izhikevich.com.

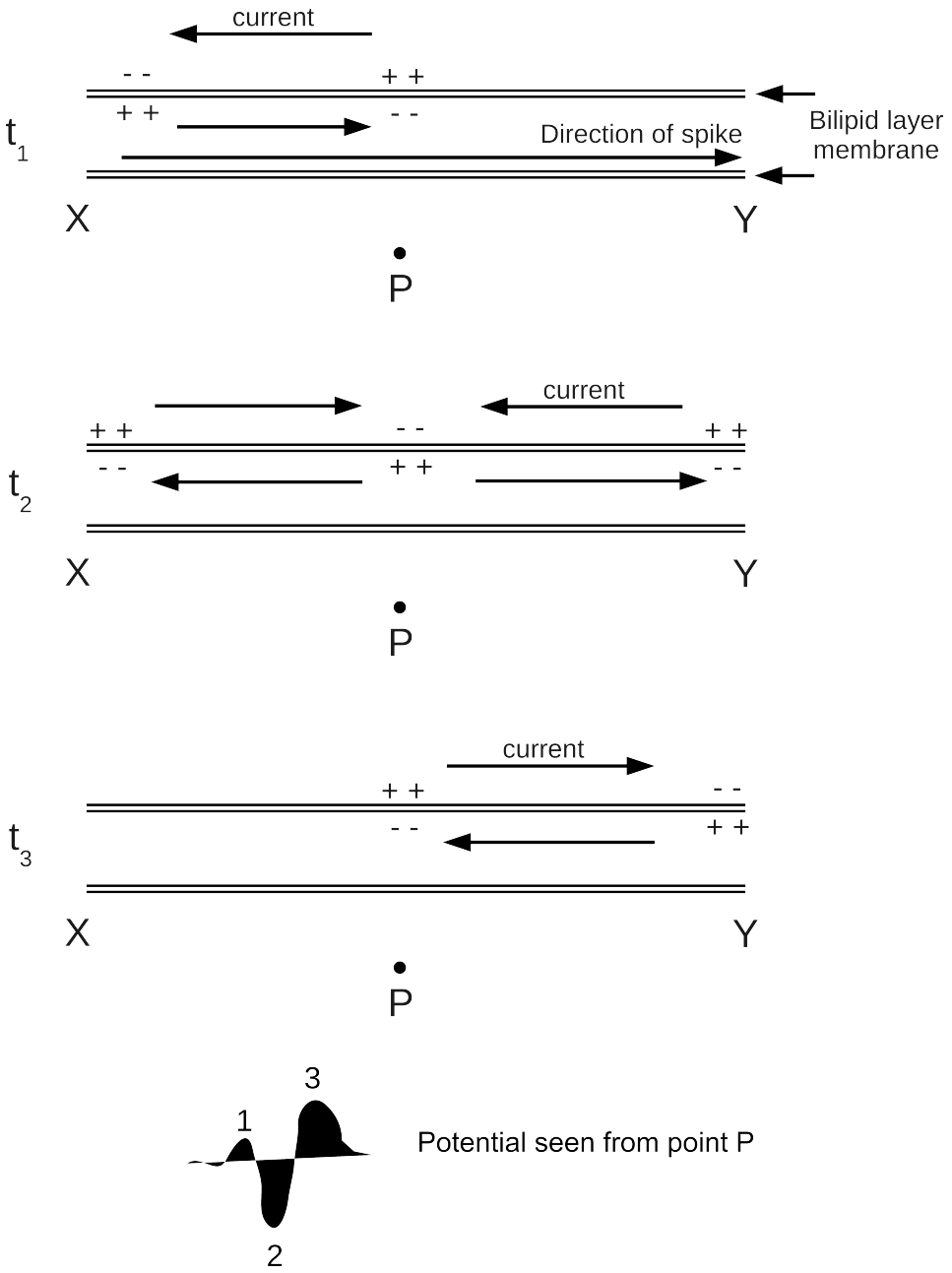


Figure 2.6: The propagation of an action potential along an axon as observed from point P, as described in [1].

A well-known example of successfully interfacing electronics to neurons is the implantable hearing aid or cochlear implant [18].

Extracellular fluid channel models typically assume a constant conductivity of the extracellular space [1]. Any neural interface consists of the following components: a signal source, electrode(s), and electronics. The choice of neurons controls the choice of electrodes. If a single neuron or small group of neurons is studied, a single probe or a tetrode (4-electrodes bundled together) can be used. The length, tip-shape and diameter of the electrode can be calibrated for the dimensions of the neuron. The choice of electrode changes the requirements of the electronics. The capacitance and resistance of the electrode should be accounted for, but the length and location of the electrode determines whether the electronics are placed *in-vivo* or alternatively *ex-vivo*. An example of a MEA is shown in Figure 1 of [19]. This array is known as the Utah Electrode Array and has a pitch of 400 μm [19].

The electrical model for the cell-electrode interface of a MEA from [20] and [21] is shown in Figure 2.7. It is assumed that the electrolytic solution (extracellular space) is grounded with a reference electrode. The model assumes a passive membrane. The cell is at the top and the electrode array is at the bottom, with only one electrode described. The subscript m denotes membrane characteristics. R_s is called the seal resistance and represents the distance between the electrode and the cell. It has ranges of 1-10 MΩ [21]. C_{dl} is the double-layer capacitance of the electrode surface, while R_{ct} is the Faradic charge transfer resistance. The frequency dependent Z_w is known as the Warburg impedance and models the constant phase angle created from diffusion across the surface of the electrode, it is typically negligible due to the small surface size of the electrode [21]. C_p is the coupling capacitance, which assumes that the ground is shared, either through the substrate or metal. R_r and C_{in} model the routing resistance and input capacitance, respectively.

Electronics inside the body face severe constraints on power, size, and encapsulation compared to electronics outside the body. The power must be self-contained (battery, energy harvesting, wireless induction) and large enough to run the device for its intended lifetime. The device must be small enough to fit in the biological system without damaging the surrounding tissue (through heat dissipation) and survive the attack of the immune system (inflammation). Different implant sites have separate requirements. For example, an implant in the brain is different from an implant in the dorsal root ganglion, in terms of neural density, function and inflammation responses.

2.4 Application to Spinal Cord Injury

Spinal cord injury is an incident in which the neural pathways in the spinal column may be partially or completely severed between their destination (brain-to-site disconnect). This limits sensory perception and motor control. In order to diagnose and rehabilitate people with spinal cord injuries, intimate knowledge of the nervous system's communication is needed. By analysing the neural system's signals researchers can determine which neural pathways are affected by motor commands and how the signals map to intended behaviour. In addition, they can replicate these signals through stimulation to recreate functions lost in injury [22]. In order to map neural signals to behaviour more efficiently, groups of neurons need to be recorded. This need can be met by large micro-electrode arrays. Multiple electrode arrays are now a standard tool [23] and these are implanted and wired to computers to allow analysis and mapping of neural spikes to muscle movements. This allows the creation of movement by replicating recorded spikes through stimulation [14]. However, multiple recording sites are desirable so that different groups of neurons can be analysed for complex behaviour. This is difficult due to the wires attached to array implants from the computer. Also, and perhaps more importantly, the solution to producing muscle stimulation control requires a system that can allow for the movements of a mobile and flexible subject. There-

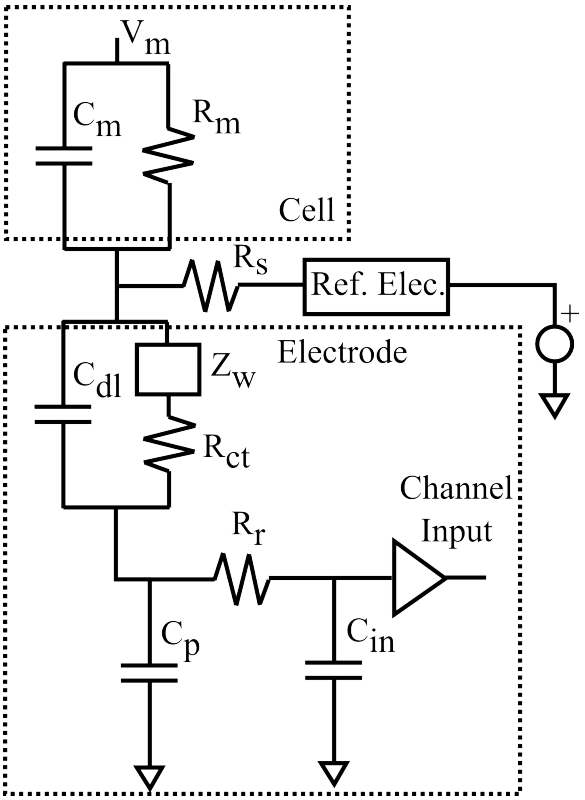


Figure 2.7: Electrical model of the cell-electrode interface for a MEA.
V_m - Cell membrane-seal voltage
C_m - Cell membrane-seal capacitance
R_m - Cell membrane-seal resistance
R_s - Seal resistance (distance between)
C_{dl} - Electrode surface double-layer capacitance
Z_w - Electrode surface diffusion Warburg impedance
R_{ct} - Electrode surface Faradic charge-transfer resistance
R_r - Electrode routing path resistance
C_p - Electrode coupling capacitance
C_{in} - Electrode channel input capacitance

fore the application requires a portable solution: a wireless implantable micro-electrode neural recording/stimulation device.

The design must be small enough to implant into the human body, specifically around the spine and/or dorsal root ganglion (DRG). It must not damage the tissue around the implant through heat dissipation, chemical poisoning or mechanical deformation. It needs to permit a full range of movement as well as mechanical deformation of the spine. It must be able to interface with the electrode array and be able to send a large number of neural signals to the base station for further processing. The neural signals can potentially have raw data rates of 240 kb/s per channel. Thus for arrays having up to 100 electrodes, the total data rates can be as high as 24 Mb/s, which taxes the capabilities of most implemented mobile wireless standards never mind an implanted device. Therefore the device must incorporate some signal reduction processing so that it transmits feasible data rates. The size constraint means that the system is most likely custom-made, whether with off-the-shelf components, custom components or an ASIC SOC. The system must be encapsulated with a material that makes it biocompatible. In addition, the design must also be able to compensate for any direct current (DC) offset from tissue. Most importantly it must be able to record micro-volt action potentials which are embedded in the noisy saline tissue environment.

A vision for the entire recording scheme is shown in Figure 2.8. It includes multiple implant sites, and a wireless power scheme. The arrays are attached by bundles of insulated wires, which are integrated into a platform for bonding to the ASIC chip. The mixed signal processor packetizes data for a wireless transmission via the stacked radio frequency (RF) chip. Both an uplink and downlink should be available, but with reduced uplink bandwidth for programmability. The base station should enable the user to choose the recording mode of operation and the stimulation profiles. Ideally, all power would be transferred wirelessly, but that creates very strict limitations on the power consumption, so the addition of a re-chargeable battery connected to the encapsulation is very likely.

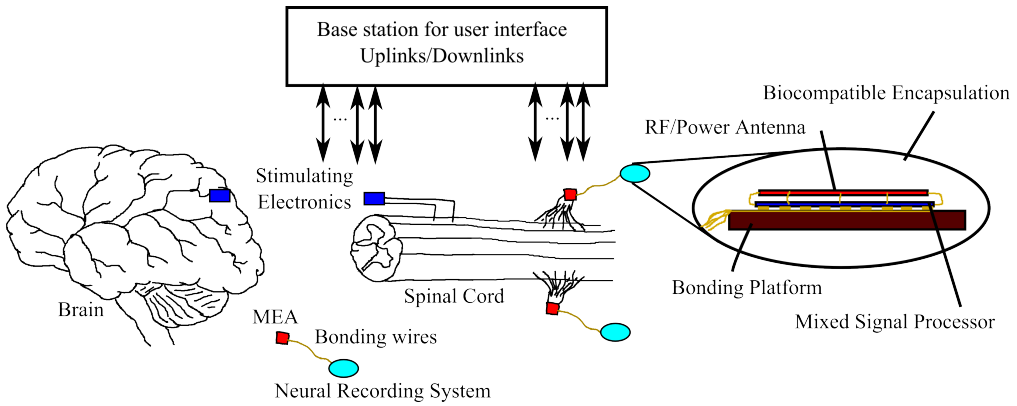


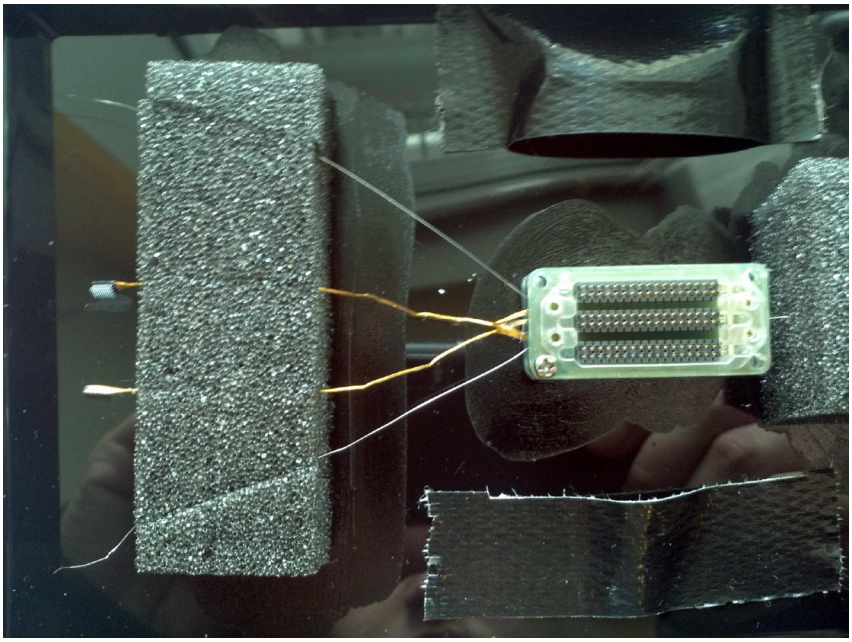
Figure 2.8: A vision for implanted recording and stimulation.

2.5 State-of-the-Art Systems

The implantable design specification makes the complementary metal-oxide semiconductor (CMOS) SOC or system-in-package (SIP) implementation a very promising solution. A micro-chip allows a complex micro-electronic system in a small space. It can handle low-power applications and has no moving parts, so the design can be protected (encapsulated) from tissue. Furthermore, since CMOS packages and chips are made at the same small scale as the micro-electrodes, interfacing should be easier.

For comparison, an implantable but tethered system is described here. Two pictures of

components from an existing tethered system are shown in Figure 2.9. Figure 2.9a shows two recording micro-electrode arrays, with 400 μm electrode pitch, bonded to a printed circuit board with a ribbon cable connector attached. The connector and system is described in [24]. The ICS-96 connector is connected with Samtec 0.05 pitch FTSH 36-pin connectors and Samtec ribbon cables. Figure 2.9b shows an analog front-end from Blackrock Microsystems, which is part of their Cerebus system. It amplifies, filters and digitizes data and then sends it optically to a large processor bank. The choice of optical data transmission is required to handle the magnitude of the data transmission bandwidth.



(a) Micro-electrode arrays attached to a ribbon cable.

(b) Analog front-end with optical data transmission.

Figure 2.9: Tethered system components.

Table 2.1: Circuit implementations of reported neural recording devices.

Source(s)	System Size	Technology	Power	Data
[33]	8.8x7.2 mm^2	0.35 μm 4M2P CMOS	6 mW, 9.5 mW/cm^2	90 Mb/s, 128 channels, 9-bit, 40 kS/s
[34]	Fits on a rat	[35,36] for front-end	645 mW	64 channels, 20 kHz
[27,28, 37,38]	0.22 mm^2	0.18 μm CMOS	75 μW, 34.1 mW/cm^2	32 channels
[29,30, 39,40]	0.2 mm^2	250 nm TSMC and 70 nm	0.36 mW/100 channels, 180 mW/cm^2	10 b/sample, 5 b/coefficient, 6 b/alphabet
[41]	1.875 x3.75 mm^2	90 nm LP	13 pJ/cycle @ 0.4 V, 18.5 mW/cm^2	100 MHz
[42]	0.0013 mm^2	65 nm LP CMOS	5 μW, 385 mW/cm^2	20 kS/s, 8 b/sample, 2 channels
[43]	4.7 x 5.9 mm^2	0.5 μm 3M2P CMOS	13.5 mW, 49 mW/cm^2	330 kb/s

The Cerebus recording system is completely external and is broken into two sets of specifications: 1) 128-channel front-end amplifier and 2) 128-channel neural signal processor. The front-end amplifier has an amplification gain of 5,000, an input range of -8.095 mV to 8.095 mV and the input referred noise of 3 μV_{RMS}. It uses a 1^{st} and 3^{rd} order Butterworth filter to achieve a high pass cut-off frequency of 250 Hz and low pass cutoff frequency of 7.5 kHz, respectively. The neural signal processor samples at 30 kSps with 16 bit/sample and is capable of 8^{th} order high/low pass digital filtering on all channels as well as online spike extraction and classification using time/amplitude windows and/or template matching.

The existing state-of-the-art *in vivo* design is typically limited to no more than 100 recording channels/device. The channels are processed with an analog front-end low-noise amplifier (LNA) and filters followed by an analog-to-digital converter (ADC). If the system is sending the data wirelessly then a compression module is necessary [25]. Before wireless transmission the digitized signals are either compressed by a discrete wavelet transform (DWT) [25–28] or transmitted with little or no processing beyond threshold detection. Compression is also performed with classification using a vocabulary for patterns [29, 30]. Another approach is lower fidelity switch-based delta read-outs [31, 32] to avoid the ADC and compression.

Table 2.1 is a technical summary of various designs from the literature. The designs are reviewed in the following subsections. The contributions proposed in this thesis are that compression is dominated by spike detection and that utilizing shape compression and/or spike sorting may allow a low-power implementation for a large (>100) number of channels. The method of spike detection and compression, however, may vary.

2.5.1 On the Fly Feature Extractor, Chae et al.

This subsection is short review of the performance and operating conditions of the work presented in [33]. This paper presents a 128-channel neural recording ASIC. The circuit contains 128-channels which are split into 16-channel recording blocks with spike detection and feature extraction. The ADC creates 9-bit sampled data at 40 ksamples/s/channel. Spike detection is performed with a non-linear energy operator (NEO), which is described in detail in Chapter 3, followed by a derivative-based frequency-shaping filter for feature extraction. The NEO threshold is calibrated off-chip in a training phase and in order to improve their spike classification the peaks of the spike waveforms are stored. Finally, the data is transmitted wirelessly using an impulse radio based ultra-wide-band (UWB) transmitter at 90 Mb/s using on-off keying (OOK) modulation. The chip dissipates 6 mW and has an area of 8.8x7.2 mm^2. The latency is 41 cycles (1.025 ms) and the digital signal processor (DSP) power is 0.1 mW, with a 1.65 V power supply.

2.5.2 Neuroplat Wireless Neural Recorder, Szuts et al.

This subsection reviews the cortical neural recorder presented in [34]. The system combines the Neuroplat ASIC with various off-the-shelf components. It records from 64 channels at a 20 kHz sampling rate. This is one of the few devices that is capable of a longer transmission distance of 60 m at 2.380 GHz. The signals being processed also have a smaller pass band of 90-2300 Hz compared to other devices, which go up to 7500 Hz. The device is implanted, but not enclosed inside the rodent brain. It operates with a 1.6 V power supply and dissipates 165 mW in the Neuroplat ASIC and 480 mW in the other components. This power dissipation allows the device to operate for 6 hours.

2.5.3 Optimized DWT for Neuroprosthetics, Oweiss et al.

This subsection reviews the work developed in [27, 28, 37, 38]. The DSP utilizes a lifting-based DWT architecture since it has lower memory requirements and outperforms a B-spline architecture. The wavelet basis, from the Matlab library, that was reported to give the best neural spike matching is the symlet4 wavelet. The system records from 32-channels and uses a 4-level DWT with 76 μW of power in 0.22mm^2 of area. The architecture was optimized at the arithmetic level with a pass-gate logic 1-bit full adder in a ripple carry adder configuration and an optimized Booth multiplier. In addition, memory units were made of static random-access memory (SRAM) instead of registers. The compression came from the energy compaction of the wavelet transform and the application of a threshold. An option to use spike detection after the DWT was presented as well as a spatial whitening filter to aggregate near channels together and avoid common noise sources.

2.5.4 DWT with Vocabulary Encoding, Narasimhan et al.

This subsection reviews the work presented in [29,30,39,40]. A DWT is applied to the neural signal similarly to the work of Oweiss et al., but the system continues to use the wavelet coefficients to define a space for clustering the spikes. Sets of spikes in a cluster are matched to an alphabet, which makes up the vocabulary. The vocabulary is built through offline training and performs a spike sorting analysis. The architecture focuses on implementation of clock and module gating control. The system is able to process 20 channels with 6 b/alphabet entry, which can give a very high degree of compression. The results presented so far are simulated with 100 channels estimated to require 360 μW.

2.5.5 Biomedical Signal Processor Running ECG, Ashouei et al.

This subsection reviews the work of [41]. The digital system can run a complex electrocardiography (ECG) algorithm with feature extraction and motion artifact cancellation. It consumes 13 pJ/cycle, while running a continuous wavelet transform-based algorithm at 0.4 V. The chip was developed by IMEC and NXP. It uses 15 power domains and can support frequency ranges from 1 to 100 MHz. Interestingly, the authors argue that complex systems must operate with power supply voltage scaling above subthreshold levels because the achievable clock frequencies, on-chip memory, and computation precision when using subthreshold voltage scaling are limited to simple algorithms like the ECG algorithm.

2.5.6 Neural Signal Acquisition IC, Muller et al.

This subsection reviews the work in [42]. The system avoids an analog data path by replacing the typical analog filters with a dual mixed-servo loop, which digitizes both action and local field potentials. This allows the system to operate with a low voltage supply of 0.5 V and consume 5 μW of power while operating two channels at 20 kHz, with 8 bits/sample. The system transmits raw data and does not perform any spike detection or compression.

2.5.7 100-Electrode Neural Recording System, Harrison et al.

This subsection reviews the work presented in [43–45]. The system is a complete neural recording solution from implant to wireless transmission. The ADC samples at 15 ksamples/s with 10 bits/sample and transmits detected neural spikes. The neural spikes are detected using a user programmed threshold or alternatively (but not implemented) the threshold can be calibrated by estimating the noise as a band-limited Gaussian source. The threshold is then set at an integer multiple of the standard deviation of the noise. The standard deviation of Gaussian noise can be found by finding the threshold at which the probability of Gaussian noise exceeding the threshold is 0.159. The typical integer multiplication is N=3. This threshold is chosen to ignore the majority of false detections that should occur in a Gaussian shaped noisy channel. The chip operates at a supply of 3.55 V and 13.5 mW. The spike detection data is transmitted using frequency shift-keying (FSK) at 433 MHz. The system experiences performance degradation due to the analog and digital systems interfering with each other. The authors also suggest that sending spike features and clustering externally may be a good choice for future systems.

2.5.8 Comparison of the State-of-the-Art

In the systems reviewed a wide range of silicon technologies with different minimum feature sizes are utilized. Sizes range from 65 nm to 500 nm. The various systems have different analog front-ends as well as different wireless specifications. In general the larger silicon technologies are used for the analog circuitry in order to reduce leakage power and improve noise performance. The smaller technologies are used for digital processing. The most power efficient and high throughput system is described in [33]. The power efficiency comes from the use of an ultra-wide band antenna using a near field communication link. The energy cost of sending a data bit across the communication link is comparable to the energy cost of signal processing computations in silicon. This means that additional signal processing is not needed and would be detrimental to the overall power dissipation of the system. Therefore only the digitized raw signal is transmitted. The disadvantage of the system is that the transmission coil must be close to the skin in order for the link to operate, which may not be a feasible option for implantation. The remaining systems use wireless transmission through free space that have larger energy costs for sending data. Therefore the remaining systems reduce the power dissipation of their system by compressing the data

through spike detection and spike compression. The neural spike detection methods are known algorithms with novel implementations like the NEO and absolute threshold [43–45]. The spike compression methods are predominantly wavelet-based like in [27–30,37–40]. The wavelet encoding methods utilize a vocabulary approach or a threshold approach for reducing the amount of data in the transmission. In terms of compression, the vocabulary-based approach of [40] provides the most compression with respect to energy spent.

2.6 Spike Sorting

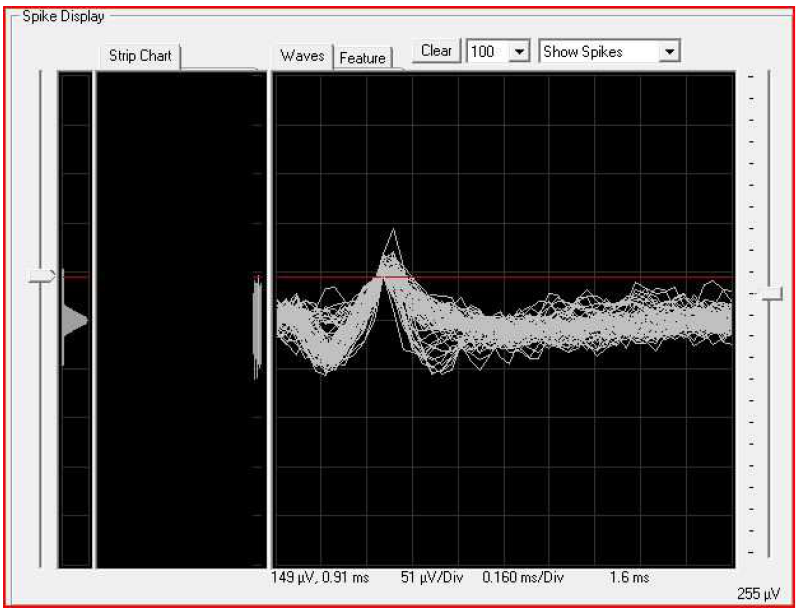
The process of mapping recorded action potentials back to the originating neurons is called *spike sorting*. All of the reported neural recording systems are trying to enable or perform spike sorting. A sequence of spikes from a single neuron is used, typically offline, to develop stimulation protocols and to learn how the nervous system encodes information. During experiments, spike activity and their respective firing rates are associated to a behaviour or stimulus occurring during that time. To test the recorded neural code, experiments are repeated with neural stimulation to recreate the behaviour.

Spike sorting is an active research area and there is no agreement on what algorithms are best [23]. Spike sorting is the combination of two separate processes: spike detection and spike identification. Spike detection determines between when a neural signal is an action potential or when it is noise. For the most part, spike detection in the implementation literature is dominated by two techniques: threshold-crossing and energy operators. Spike identification labels similar spikes, such that different neural sources can be attributed to be the cause of each respective spike. Spike identification is most commonly done with principal component analysis (PCA).

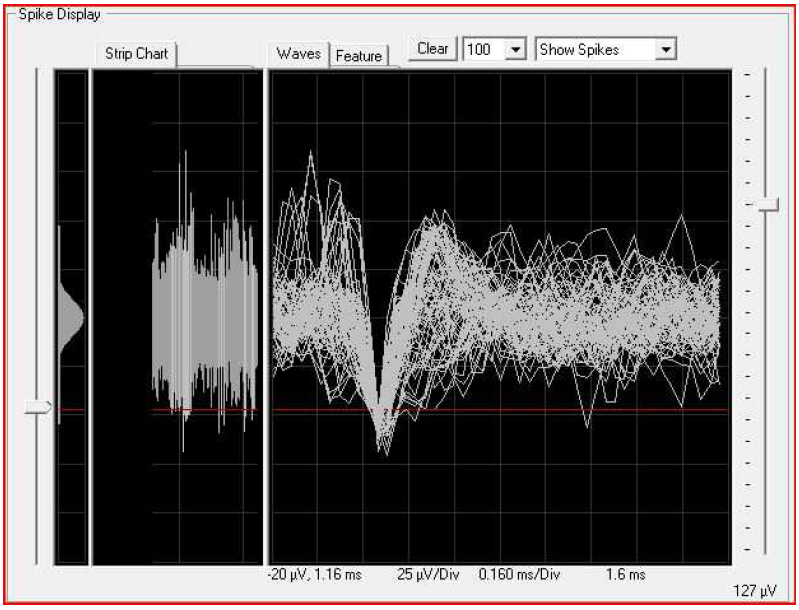
In the Blackrock Microsystems software, spike detection by threshold crossing can be either supervised or adaptive [24]. Most thresholds are defined by a user and can be either a positive/negative threshold crossing or both. In Figure 2.10, a positive and negative threshold are shown from a single channel neural recording in Cerebus software [46]. In each plot a 1.6-ms time capture around the detection event (threshold crossing) is made, with most of the 1.6 ms after the detection event. Each time the neural signal crosses the threshold, the new 1.6-ms spike is superimposed on top of previously detected spikes. The thresholds can also be adaptively based on noise seen during inactive periods of recording by scaling the threshold above the estimated noise level.

Spike detection on the basis of a signal energy operator typically increases the contribution of fast changes and reduces the contribution of slow changes. The algorithm then relies on a threshold on the energy operator output instead of the spike voltage signal. In [47] threshold crossing is compared to a NEO and matched filtering. The authors recommend, according to their cost functions, that an absolute threshold technique is better than the NEO, although, they only consider digital implementations of the NEO. The matched filter based approaches were considered computationally expensive compared to threshold detection and NEO detection.

If we consider that a single neuron may fire at 500 Hz, a very high estimate, and has a spike duration of 1 ms, then within a typical 1-s sample, 50% of the neural signal is noise. Neurons must usually wait a short time after a spike to send another spike. This time is called the *refractory period* and is a result of the hyperpolarization of the cell membrane. In feline experiments the refractory period is about the same duration as a spike, 1 ms [48]. A lower neural activity estimate, based on experimental recordings of the DRG of a feline, has firing rates of 20-100 Hz, which yields a signal reduction. Given the range of the firing rates of single neurons, spike detection has the potential to be the largest single contributor to neural signal compression. This assumes that the detector is efficient at detecting spikes and avoids false detections. The system must also consider that the end-user may want spike



(a) Neural signal detections from a positive threshold.



(b) Neural signal detections from a negative threshold with increased zoom.

Figure 2.10: Superposition of spikes occurring from different thresholds on a single channel neural recording.

shape information and/or a statistical measure of firing rate. The *spike error rate (SER)* is a measure of false detections and would also serve as a metric of the wasted power for the transmission side of the system. Identifying/extracting spikes from neural recordings with a low signal-to-noise ratio (SNR) is an active research area [49, 50].

Once a spike has been detected, the information that it carries needs to be transmitted. There are three main views of neural coding. They are exemplified by the rate-coding hypothesis, the temporal-coding hypothesis and population coding [9]. The rate-coding hypothesis states that the information produced by a neuron is carried in the firing rate of the action potentials, whereas the temporal-coding hypothesis states that the information is carried in the precise timing of the action potentials. Population coding relies on relative timing of spikes within a group of neurons. As this is still an open question for neural systems, both shape and timing information of a detected spike is required for any post-processing. In addition, when populations of neurons are analysed together, we can investigate the coordinated-coding hypothesis which states that messages are conveyed, in part, by relationships among signals from multiple neurons. No matter the perspective, spike timing and spike shape information are likely both required given the current state of knowledge.

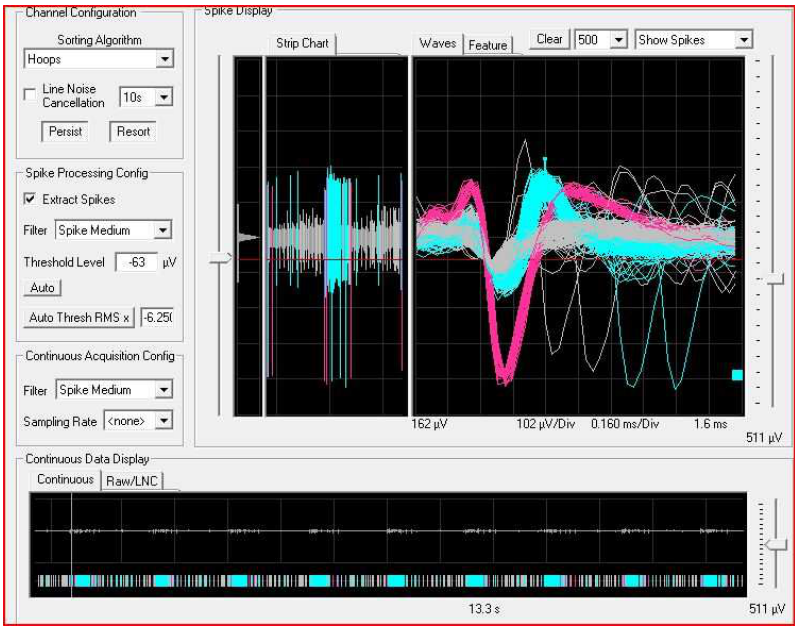
In spike sorting a unit is defined to be a group of spikes that are considered to have originated from the same neural source. Spike identification labels spikes to a specific unit or neuron. A unit can be made of a single spike or 100's of spikes. A set of units is the direct result of spike sorting as it identifies the action potentials to their respective (perceived) neural sources. Not all detections are mapped to a unit and these detections are then considered as unknown or noise. Figure 2.11a is an example of two units. Two distinct groups of spikes are seen and are classified separately: pink and blue.

Figure 2.11a shows the result of supervised spike identification on a single channel from a recording of a feline DRG during controlled leg movement. Every time the neural signal crosses the threshold, a 1.6-ms snapshot around the threshold crossing is commenced. This generates a plot of all the possible spike candidates to be classified into units. In this channel, two units are identified from all of the events that cross a threshold of 63 μV. The units are generated by user-defined lines called hoops. The user observes the superposition of N (500 in Figure 2.11a) events and draws a line on the screen. If any portion of a snapshot passes through that line it becomes a member of that unit. Multiple lines can be selected for a single unit. All remaining events are left unclassified. Figure 2.11b shows an event plot of all 96 channels showing events that were mapped to units. It shows that most channels share the same periods of increased activity, while some channels that were not analysed still show the behaviour, which makes an interesting case for simple threshold detection on good SNR channels. The combination of Figure 2.11a and Figure 2.11b can qualitatively show the quality of the recording and spike sorting. The channels should show periods of increased firing rates. Figure 2.11a shows a good channel.

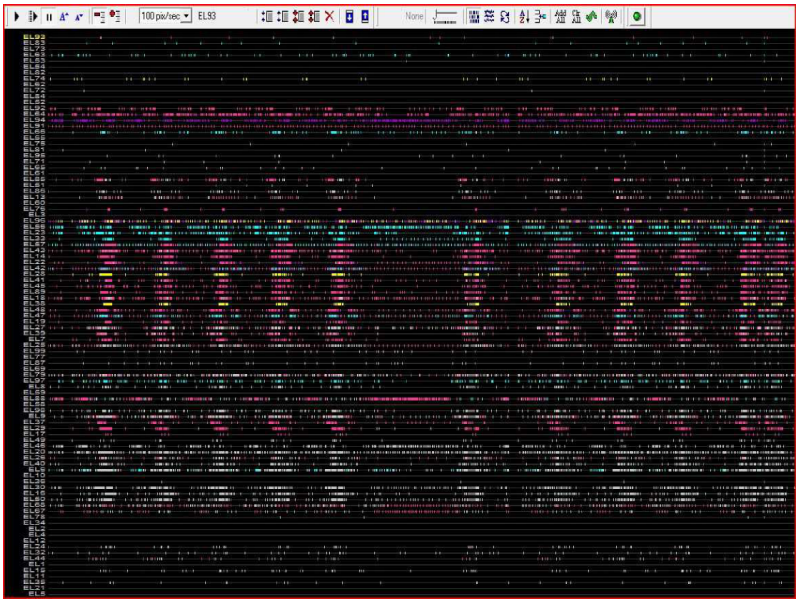
Spike sorting is typically done in software. Software suites from the companies Neuroshare, Plexon and BlackRock Microsystems are available that can take recordings (using their own proprietary format) from their implanted devices and perform offline spike sorting using different analysis techniques like PCA and template matching. PCA is the most frequently used spike sorting method despite not being proven to be the best [51]. Different clustering techniques used in the literature for spike sorting are listed and briefly described below. They can reveal interesting properties of action potentials that may be used in implanted implementations.

2.6.1 Principal Component Analysis

PCA is a transform that reduces the dimensionality of a data set with interrelated variables set into a smaller number of artificial variables that are called *principal components* [51–54].



(a) Spike superposition with two units identified.



(b) 60 s recording of 96 channels with periods of increased firing rates visible.

Figure 2.11: Cerebus spike sorting software results.

The components are the result of an orthogonal linear transform that creates the greatest variance by any projection of the data. The first few components capture most of the variation present in all of the original variables. The first component accounts for the greatest amount of variance and with each consecutive component the variance decreases. Therefore we typically plot the first and second components to observe any similarities between data points and observe any resulting clusters. PCA is used in this thesis as part of a supervised clustering technique, which is described in the next chapter. PCA can also be used for pattern recognition and factor analysis. An interesting property of PCA is that it automatically projects to the subspace where the global solution of K-means clustering lies [55].

2.6.2 K-means Clustering

K-means clustering takes n observations of the data x_i and clusters them into k spherical sets while minimizing the within cluster sum of squares, as shown in Equation 2.2 [51,56,57].

$$arg \quad min \sum_{j=1}^{k} \sum_{i=1}^{n} ||x_i - \mu_j||^2 \tag{2.2}$$

where μ is the mean of points in set i. The Euclidean distance to cluster centres and their variance are the metrics for cluster comparison. The k-means clustering algorithm is made of the following steps:

- Place K points into the space. These points are the initial centroids.
- Assign each object to the group that has the closest centroid.
- When all are assigned, recalculate the centroids.
- Repeat the second and third step until the centroids do not move.

K-means clustering is also used in dictionary learning with supervised or unsupervised approaches available and can be used to develop the templates used in template matching [58]. In the thesis work k-means clustering could be used to provide an unsupervised method for clustering the principal components.

2.6.3 Template Matching

Using known spike shapes or models to classify new action potentials is known as template matching. Additionally, we may find parts of a signal that are like the model and match the spike piecewise. The template shapes can be based on cluster centres found in either PCA clustering or k-means clustering. Then if a new spike lies within a known cluster then it is matched to the template or centroid. However, if we can simplify the template to a number of features we can match spike features like: amplitude, peak-to-peak amplitude, spike width, positive and negative slope, position of positive and negative peaks by using fuzzy clustering or k-means clustering [51].

2.6.4 Bayesian Clustering

A probabilistic approach to clustering is to model each cluster with a multivariate Gaussian where the likelihood of the data given a particular set s_k is given by

$$p(x|s_k, \mu_k, \sum_k) \tag{2.3}$$

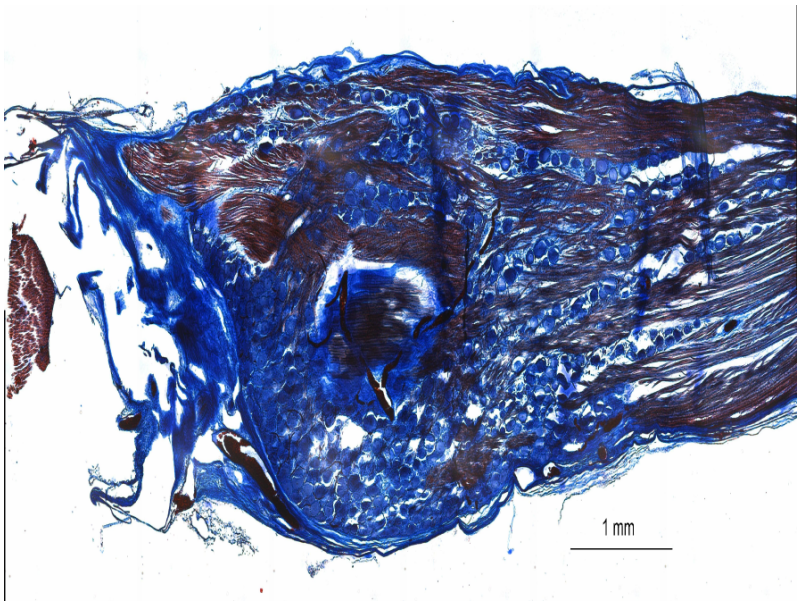
where μ_k and $\sum_k$ are the mean and covariance matrix for the set s_k, respectively [51]. If the Gaussian cluster model is accurate most of the data falls within the three-sigma error boundary, and the user is able to quantify the certainty of the classification, which is an advantage of Bayesian clustering.

2.7 Discussion on Neural Signal Recording

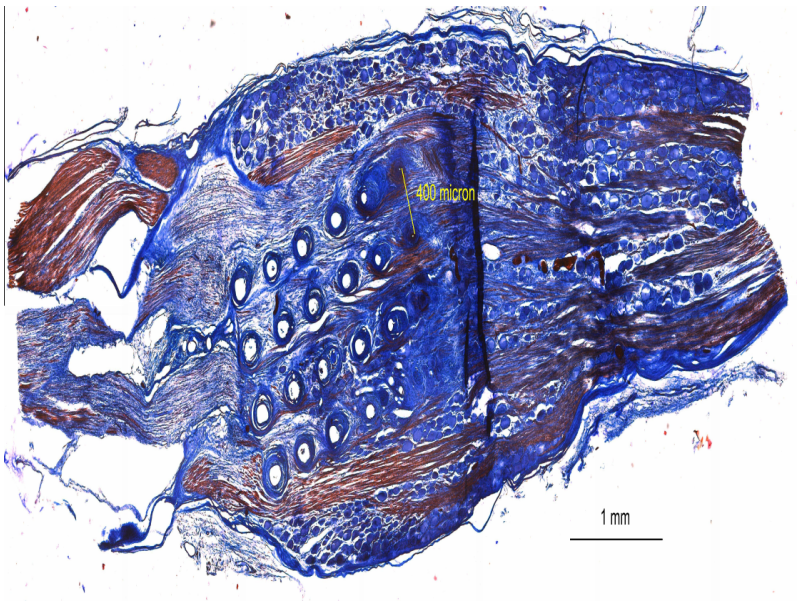
Neural systems are complicated communication structures that have detailed models that are useful for practical experimentation and simulation. These models rely on assumptions on how information is carried. Most of the neural information appears to be transmitted by spikes, whether it is the timing, shape, and rate of the spikes is still not defined for all neurons (even though most evidence points at the timing, we still require the shape to tell neurons apart during spike sorting). This gives us a specification for the recording device: it must have enough flexibility so that it may capture both the timing and shape information of spikes. Therefore, at the very least, any signal processing in the implant could throw away inactive recordings to help attain feasible data rates. It is also evident that spike detection is the best method for reducing data bandwidths. The difficulty of spike detection is finding spikes that are very close to the noise level. This can be counteracted in two ways: use a better recording electrode interface, or apply more sophisticated processing algorithms to filter out noise and spikes.

The performance of neural recording devices is highly dependent on the electrodes that they use. One of the key factors is the proximity of the 'active' portion of the electrode to the neuron, which is why we use micro-electrode arrays so that more electrodes are near (more) neurons. This is necessary to give better spike sorting performance. Additionally, we have seen only one approach in the literature that takes into account the spatial location of the electrode with respect to the array [59]. This has the potential to further increase the compression since a neuron may be recorded by several electrodes, although this is unlikely in our experiments. Figure 2.12 shows histological samples from the site of an implanted array. Figure 2.12a is the tissue under the uninsulated tips of the electrode, while Figure 2.12b is an unhealthy tissue cross section from where the electrode array is implanted. An electrode array with pitch of 400 μm, in this case, is unlikely to see the same neuron on more than one electrode, however it is likely that multiple neurons may be captured by a single electrode. Figure 2.12 also shows a major disadvantage of implanted MEAs in that the tissue is damaged and signals are likely to degrade with time as inflammation interferes with the electrodes ability to sense ionic changes in the extracellular fluid around neuron cell bodies [60]. Unpublished results and measurements from [60] indicate a chronic experiment can be expected to have a decrease in capturing spike activations. Therefore, it is important to include a scheme of graceful degradation in the neural implants, as electrode failures, and inactive electrodes are highly likely.

Identifying the tradeoffs between efficient spike detection, efficient spike compression and SER is necessary to build an efficient processing architecture. The relationship of a processing algorithm (detection or compression) with SER, power consumption, data rates and physical (logical) area are important metrics to compare options at the state-of-the-art. It was observed in the tethered feline DRG recordings that if a channel has a signal with good SNR it is typically a 1:1 mapping of neuron to electrode. However, channels with low SNR and no clearly visible activity are completely ignored by the spike sorting algorithm, yet still transmit events based on aggressive threshold detections to give a high SER. Extracting spikes from the noise of low SNR channels is an open problem and work with matched filters and non-linear energy operators has shown it is possible [47, 61]. Nonetheless, compression of neural signals is an active research area for circuit design and processing algorithms. In the signal classification literature, with respect to neural signals,



(a) Histological sample of healthy tissue from below the MEA. Courtesy of Dirk Everaert at the University of Alberta.



(b) Histological sample of unhealthy tissue from the site of MEA implantation. Courtesy of Dirk Everaert at the University of Alberta.

Figure 2.12: Histological samples from a feline DRG.

both PCA and template matching show good performance, but Bayesian methods have an additional advantage of giving the user a confidence on the cluster choice. It is not clear at this time what algorithm and what implementation would best suit a real-time implanted spike classification system, but it does appear that researchers need a real-time unsupervised online spike classification system for dense electrode arrays. These systems are difficult to design since they need to minimize the SER, while compensating for effects like spike overlap, shape irregularities, electrode drift, stimulation artifacts, and connective tissue growth. This thesis focuses on implementations that can provide different modes of operation to create a neural oscilloscope. We want to provide neuroscientists with a scalable tool that allows a flexible look at the neural signals being acquired by the implanted electrode array, such that a signal can be viewed at different information levels, from raw data, spike shape and timing, and finally just activity rates.

Chapter 3

Evaluation of Alternative Compression Methods

This chapter reviews compression methods and presents simulations of single-channel neural signals using various techniques that could be applied to the *real-time* compression of neural signals. The first section introduces lossy compression and the second section focuses on spike detection methods since they are already known to have large bandwidth reduction potential. The third section reviews compression methods for bandwidth reduction on a continuous recording, which includes transformation, compressed sampling, and feature extraction. The final section concludes the chapter with observations on the discussed alternative methods. This chapter contains an in-depth background to various signal processing algorithms and can be skipped with no loss of essential information for the succeeding chapters. The conclusion of this chapter summarizes the main findings.

3.1 Lossy and Lossless Compression

Compression, also known as source coding, is the process of removing statistical redundancy from a signal in order to express it more efficiently in terms of information per bit. The process finds a code to best match the source such that the Shannon entropy of the coded signal is maximized. The Shannon Source Coding Theorem [62] establishes the mathematical limit to possible data compression.

There are two types of compression: lossless and lossy [63]. Lossless compression is a process that does not reduce the total amount of information being delivered, whereas lossy compression reduces the total amount of information. In most systems, compression is typically used to conserve transmission bandwidth with little concern over power consumption, but in an implantable wireless sensor power is a primary concern. The aim of an implanted compression unit is to reduce the transmission bandwidth without increasing the net power consumption (compression plus transmission) and hopefully even reduce the net power consumption compared to an uncompressed system. Equation 3.1 gives the relation between the two systems. The system with compression is on the right and the one without compression is on the left. The symbol CR means the compression ratio and is defined as the number of bits out to the number of bits in for the compression algorithm. E_{tx} is the energy required to capture, encode and transmit a bit, and E_c is the power required to compress a sample by CR.

$$\frac{E_{tx}}{bit} \geq CR\frac{E_{tx}}{bit} + \frac{E_c}{bit} \tag{3.1}$$

Lossless compression followed by decompression allows the original data to be recon-

structed exactly. A non-compressed signal $x(t)$ can be represented using the Nyquist-Shannon sampling theorem [62, 64], which states: *If a function x(t) contains no frequencies higher than B Hz, then the signal is completely represented by taking its samples at a series of points spaced 1/(2B) seconds apart.*

Most digital sampling is based on this theorem. Consider expressing a signal $x(t)$ in terms of its Fourier spectrum $X(w)$

$$x(t) = \frac{1}{2\pi} \int_{-2\pi B}^{2\pi B} X(w) e^{iwt} dw \tag{3.2}$$

Let $t = \frac{n}{2B}$. If we assume that the signal power $X(w)$ of angular frequency w is zero outside the pass band edge frequency B then

$$x\left(\frac{n}{2B}\right) = \frac{1}{2\pi} \int_{-2\pi B}^{2\pi B} X(w) e^{iw\frac{n}{2B}} dw. \tag{3.3}$$

The left side is the sampled time-domain signal $x(t)$ at sampling points $1/2B$ apart and the right side is the Fourier series expansion of the function $X(w)$ taking the interval $-B$ to B as the period. This means the samples of $x\left(\frac{n}{2B}\right)$ determine the Fourier coefficients in the series expansion of $X(w)$ [62] and, therefore, the samples determine the function completely. Some examples of lossless compression that can build upon the sampled representation of a signal include: Huffman coding [65], LZ77 [66], run length encoding and substitution coding in general.

Although not lossless, in some practical scenarios when the data is subject to human interpretation, the result of lossy compression can appear effectively lossless. By modelling the human sensory system, aspects of sampled information can be lost since the limits of human sensory perception do not require these samples; thus the compression appears to a human to be lossless [67]. An example of this is sound representation using the MP3 lossy compression standard [68]. In addition, important information may be in the signal, but may be difficult for humans to perceive without signal enhancement, which may be especially true for neural signals, given the large parallel scale of neural structures. In this thesis lossless compression is largely ignored since we need a larger compression gain which can be provided through lossy compression.

Lossy compression means that the data is represented in such a way that exact recovery of the original data is not possible, but a sufficiently accurate approximation is [69]. This data loss causes distortion in the recovered signal compared to the original signal. The relationship between distortion and the information rate is described by Rate-distortion theory. Rate-distortion theory was introduced by Shannon in 1948 [62] and was further refined in 1959 [70].

One measure of distortion is the expected value of the square of the difference between input and output, that is the mean squared error (MSE) of the difference. This is shown in Equation 3.4 with the estimated parameter θ and estimator $\hat{\theta}$.

$$MSE(\hat{\theta}) = E[(\hat{\theta} - \theta)^2] \tag{3.4}$$

The MSE can be viewed as a measure of how much the signal's energy is reduced by removing the predictable information. Therefore a better compressor (predictor) has a lower MSE. In general, lossy compression techniques can be characterized by the following:

- time to compress, operations/compressed symbol
- time to reconstruct, operations/reconstructed symbol
- compression ratio, bits out/bits in
- quality of compression

The first two characteristics measure complexity and could help determine if an algorithm is too computationally expensive for low-power real-time processing. The compression ratio defines the amount of compression performed. Lastly, is the compression quality, which is measured either by the human eye and qualitative judgements or mathematically, for example, by the MSE or false positive rates.

The remainder of this chapter focuses on reviewing lossy compression algorithms that may be applicable to neural signal compression including algorithms that could potentially classify neurons for spike sorting. Each method is, or has been applied with various degrees of success to neural data. The focus of these examples is to examine the potential for compression and the quality of the reconstructions. Unless otherwise stated, we assume that the operation is performed on discrete Nyquist sampled data $x[n]$.

3.2 Neural Spike Detection

Neural spike detection can be the single greatest contributor to the reduction of signal bandwidth. This is especially true if some electrodes are not acquiring any spiking activity. The effectiveness of a given neural spike detection method is hard to confirm since we can not know if the neural data is spiking below the noise level, only if it is spiking above it. Therefore, artificially constructed signals are sometimes used to systematically evaluate spike detection methods. The methods discussed in the following subsections use both real recordings and artificial recordings and measurements of false detections and missed detections are reported.

3.2.1 Experimentally Acquired Neural Spike Activity

The experimentally acquired neural spiking activity is from experiments similar to those described in [71]. The raw signals from an MEA implanted in a feline dorsal root ganglion are recorded and sorted offline. The offline sorting software suite is Offline Sorter from Plexon [72]. A screen capture of some of the software output is in Figure 3.1.

The software suite provides manual, semi-automatic, and automatic spike sorting methods. The manual methods are shown in Figure 3.1. In the left window any waveform that crosses the threshold is superimposed in the window. A user can draw a line segment anywhere a spike is believed to cross. Any spike that crosses the line is considered to be from the same neural source or unit. The second method asks the user to circle clusters from the PCA projection of the first two components. Any spike detections that are placed in the circles are considered from the same unit. The semi-automatic and automatic spike sorting tools use the same cluster selection methods, but do not require user intervention. The recordings used to evaluate the neural spike detection methods in this thesis use the manual clustering and waveform selection methods. Note that the some of the detections remain unsorted even though they crossed the threshold. The ability to correctly automate the spike sorting procedure is strongly desired by end-users of the neural implants. The clustering and automation of spike sorting is not part of the scope of this thesis work, although it is briefly touched upon in the feature extraction compression method.

3.2.2 Construction of Artificial Neural Activity

Neural recordings do not allow an ideal controlled comparison of spike detection methods, so researchers have constructed artificial signals to mimic and model the behaviour of neural recordings. In artificial recordings the location of spikes is known and the SNR of the spikes is controllable. An example of artificially constructed neural signals is from [73] and through their Matlab software package NeuroCube [74]. An example of the generated output from NeuroCube is shown in Figure 3.2. NeuroCube models a cube of neurons. In the example

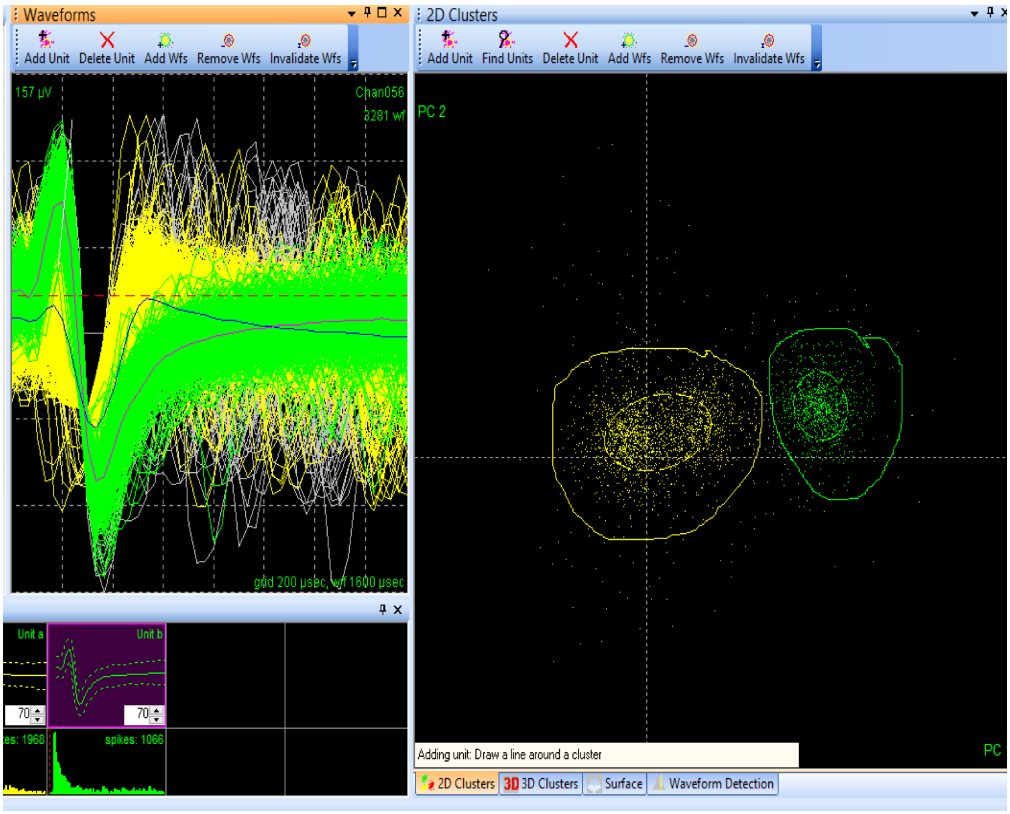


Figure 3.1: Output from the Offline Sorter software suite (demo version).

300,000 neurons are placed within 1 mm^3 with only 7% of them active. The triangles marked red are considered close to the black circular electrode. All other neurons are far away and contribute to the background noise.

After inspecting the program and the outputs available, it appears that some of the spike templates being used are biased to one domain. They do not always demonstrate strong bi-phasic or even tri-phasic shapes as seen in the recordings from the feline DRG. This is attributed to the filtration range of the spike template generation as described in [73]. The simulations also do not appear to be as noisy as the DRG recordings, which is probably due to the lack of electronic noise of the front-end electronics. Two examples of an artificial recording are shown in Figures 3.3 and 3.4 with spikes shown in red. The wide variance in spike shape means that relying on a strong bi-phasic or tri-phasic signal may not yield a wide range of acceptance for different spike types.

3.2.3 Absolute Threshold

An absolute threshold is typically used for spike detection [8, 51, 75] and is considered apart of the state-of-the-art spike detectors. Applying an absolute threshold to an input signal means that only samples above a specified amplitude β_p are accepted, as shown in Equation 3.5.

$$\hat{x}[n] = \begin{cases} x[n] & \text{if } |x[n]| \geq \beta_p \\ 0 & \text{if } |x[n]| < \beta_p \end{cases} \tag{3.5}$$

However, in the case of capturing neural action potentials, once the threshold is broken all succeeding samples for a safe interval (1 ms) are accepted and considered part of the

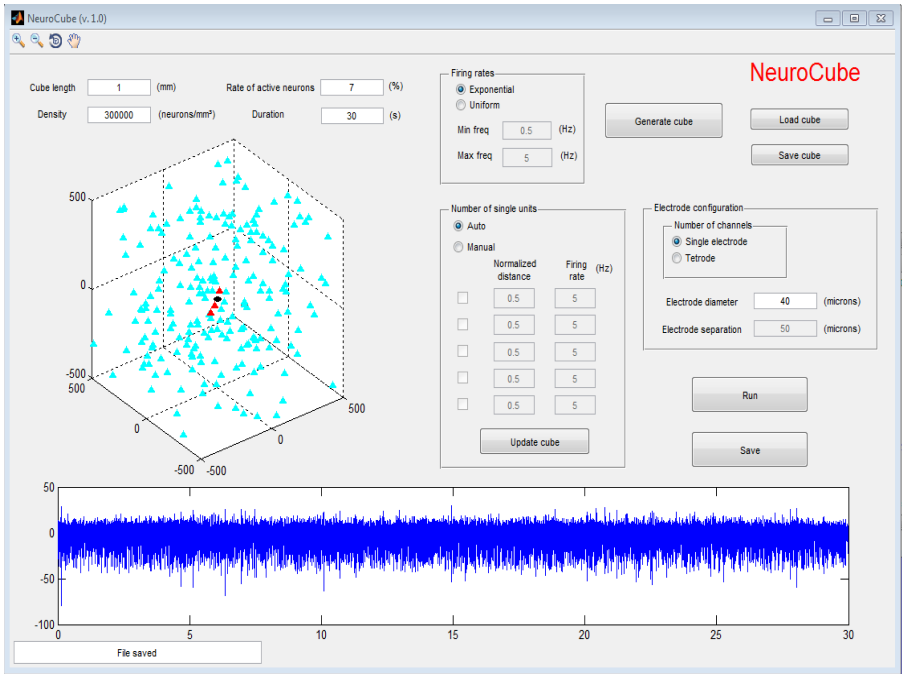


Figure 3.2: Output from the NeuroCube Matlab software package.

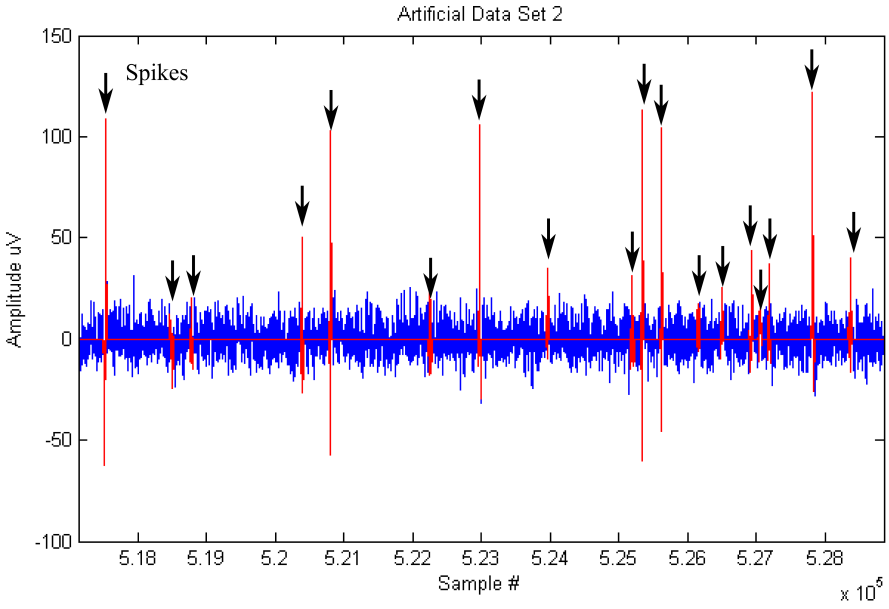


Figure 3.3: Artificial neural signal with known spikes labelled red.

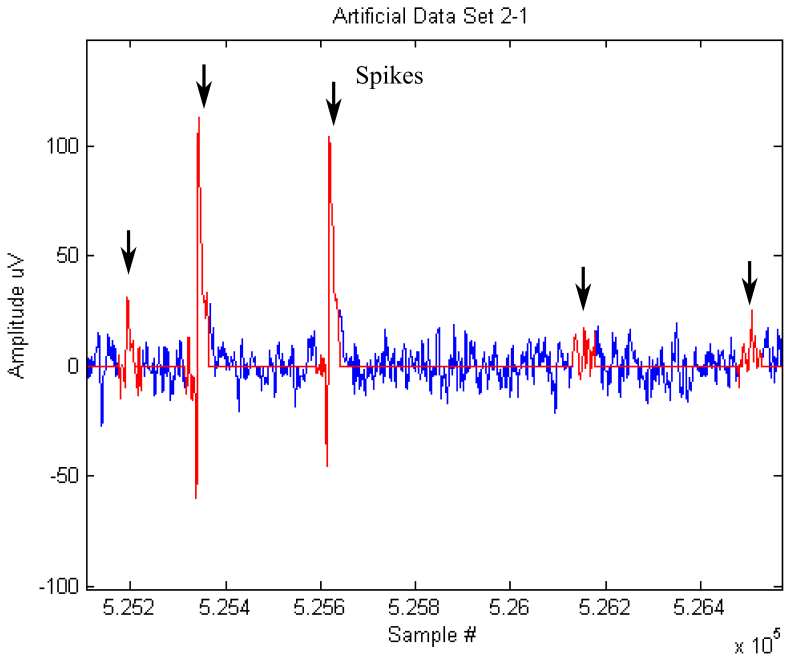


Figure 3.4: Artificial neural signal with known spikes labelled with red.

candidate spike. The detector assumes that the spike lasts m samples. A spike window of $m = 1$ shows only samples that are above the threshold. Equation 3.6 shows how the detected signal $\hat{x}[n]$ only tracks the input signal for m samples, where n_t is the value of n when the threshold is first triggered. An example of absolute value threshold detection is shown in Figure 3.5 with $m = 48$. A threshold value of 60 μV was picked to demonstrate the disadvantage of choosing a static threshold that is too high. The threshold has missed action potentials marked by '*' symbols. The absolute value threshold trigger also misses the beginning of the action potential, which means a small delay buffer is needed.

$$\hat{x}[n] = \begin{cases} x[n] & \text{if } x[n] \geq \beta_p \text{ for } n_t \leq n \leq n_t + m - 1 \\ 0 & \text{if } x[n] < \beta_p \end{cases} \tag{3.6}$$

The advantage of applying an absolute value operation is that both peaks and troughs can be detected by a single threshold instead of two. However, choosing the value of that single threshold is not obvious, especially since neural signals can contain action potentials of varying amplitudes in the positive and negative domain, as shown in Figure 3.5. In OfflineSorter, the threshold can be manually adjusted for the entire recording or it can be based on a measure of the noise.

3.2.4 Threshold Calibration

Threshold-based detection methods rely on the placement of thresholds. Automated thresholds exploit measurement of noise and variance in the signal to automatically select thresholds. A noise-based threshold is described in [75] and shown in Equation 3.6. Figure 3.5 is duplicated in Figure 3.6 for the noise-based threshold, plus the addition of a small delay buffer.

$$\beta_p = 4\sigma_N, \quad \sigma_N = median\left(\frac{|x[n]|}{0.6745}\right) \tag{3.7}$$

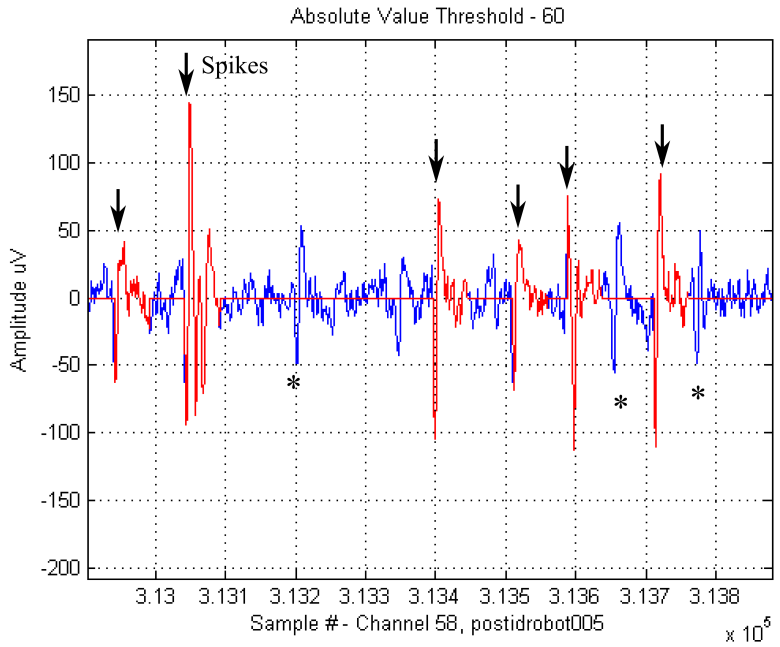


Figure 3.5: Example of amplitude-based threshold detection on a feline DRG recording.

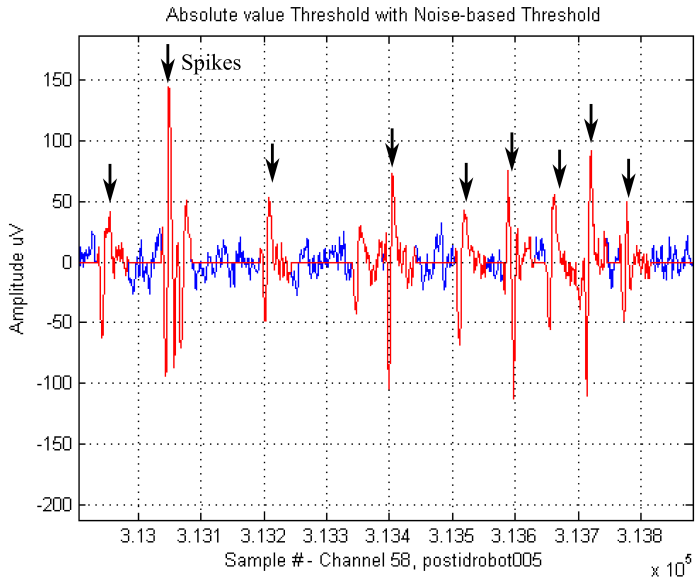


Figure 3.6: Example of noise-based threshold detection on a feline DRG recording.

The number of detections on the example channel is 4617 compared to 3117 (1224 unsorted, 1524 unit A, 19 unit B, 350 unit C) in OfflineSorter. There were 0 missed detections, but 1500 false detections. The error rates on five artificial channels from [73] are shown in Table 3.1. The true positives are actual spikes found and the false detections are detections of non-existent spikes. The total available column is the number of spikes actually in the recording. The error rate is $\frac{\text{false detections}}{\text{total detections}}$ and the miss rate is $\frac{\text{total available}-\text{true positives}}{\text{total available}}$.

Table 3.1: Noise-based threshold detection on artificial recordings

Data Set	True Positives	False Detections	Total Available	Error Rate	Miss Rate
Sim1	1482	129	2415	8.0%	38.6%
Sim2	2226	100	3214	4.3%	30.7%
Sim3	2329	115	3283	4.7%	29.1%
Sim4	2250	112	3193	4.7%	29.5%
Sim5	1304	123	2328	8.6%	44.0%

Noise Envelope

Instead of calculating an average or variance, the envelope of the noise can be estimated. A window-based noise envelope threshold is described in [76] and the algorithm steps are shown below.

Calculate the maximum absolute value in the window, $Mx_i = max(|x[n]|)$.
If $Mx_i > KEn_{i-1}$, assume there is a spike and ignore the window otherwise
if $Mx_i > En_{i-1}$, $En_i = \alpha * Mx_i + (1-\alpha) * En_{i-1}$ or
if $Mx_i < En_{i-1}$, $En_i = \beta * Mx_i + (1-\beta) * En_{i-1}$

The authors of [76] set $\alpha = 0.02$ and $\beta = 0.5$ for 50 ms windows. The value of K is set to 4 for these simulation results. The number of detections on the example channel is 4151 with 0 missed detections and 1034 false detections. The error rates on the artificial channels are shown in Table 3.2.

Table 3.2: Noise envelope-based threshold detection on artificial recordings

Data Set	True Positives	False Detections	Total Available	Error Rate	Miss Rate
Sim1	1875	1559	2415	64.5%	22.4%
Sim2	2731	1409	3214	34.0%	15.0%
Sim3	2479	506	3283	17.0%	24.5%
Sim4	2444	503	3193	17.1%	23.5%
Sim5	1721	1245	2328	42.0%	26.1%

3.2.5 Nonlinear Energy Operator

The NEO was first characterized in [77] as the Teager energy operator and creates an estimate that also factors in the frequency of the signal as shown in Equation 3.8. The NEO is a pre-detection process that accentuates the spikes. Examples of the operator output are shown in Figure 3.7.

$$Y_{NEO}[n] = x^2[n] - x[n+\delta]x[n-\delta], \qquad 1 \leq \delta \leq 4 \tag{3.8}$$

The frequency information makes the estimator increase the size of spikes, but the operator output also requires a scaled threshold [78], which has an experimentally defined scaling

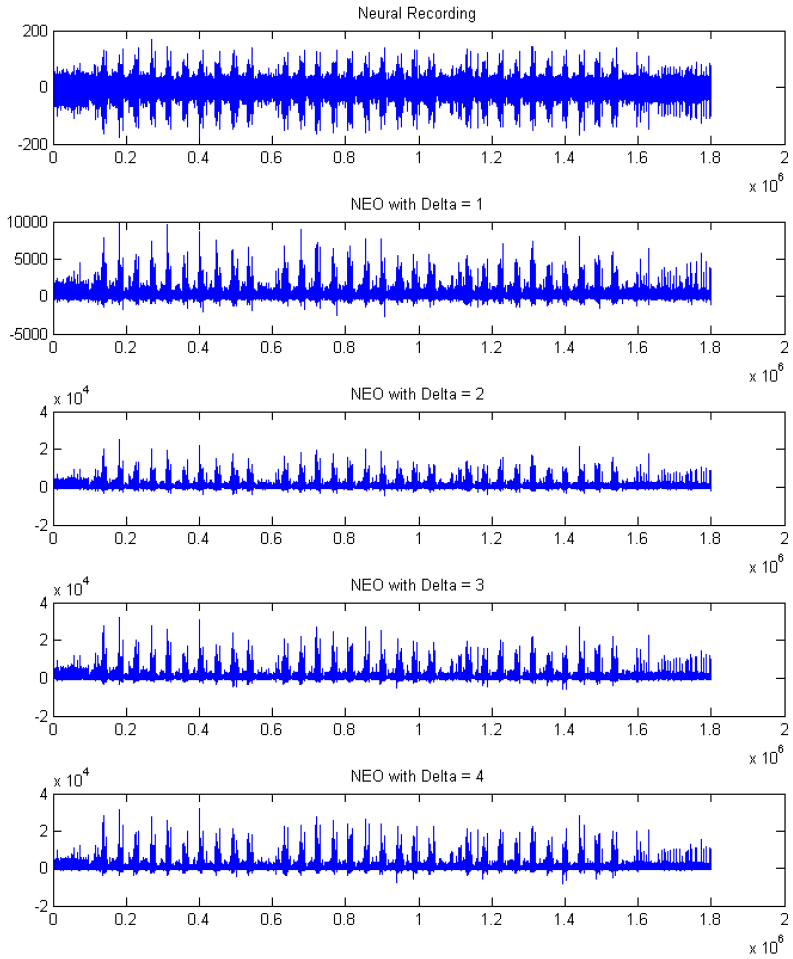


Figure 3.7: Example of a NEO on a feline DRG recording.

factor that depends on the signal. The results for the biological example, with $\delta = 1$, is 4195 spike detections and the results for the artificial channels are shown in Table 3.3.

Table 3.3: NEO threshold detection on artificial recordings, $\delta = 1$

Data Set	True Positives	False Detections	Total Available	Error Rate	Miss Rate
Sim1	1396	917	2415	39.7%	42.2%
Sim2	1965	275	3214	12.3%	38.9%
Sim3	2198	722	3283	24.7%	33.1%
Sim4	2120	699	3193	24.8%	33.6%
Sim5	1296	1154	2328	47.1%	44.3%

3.2.6 Two Thresholds in a Window

A second threshold was introduced by researchers to take advantage of the bi-phasic nature of the neural spike. In [79] both a positive threshold and negative threshold must be crossed in a time window. The authors also filter the trough, but that is not considered here. A time window of 0.9 ms is used in the example. The thresholds are calibrated using the noise envelope method, but for the positive and negative samples separately. In Figure 3.8 two thresholds are applied and spike detections are shown. The thresholds can be seen moving up and down.

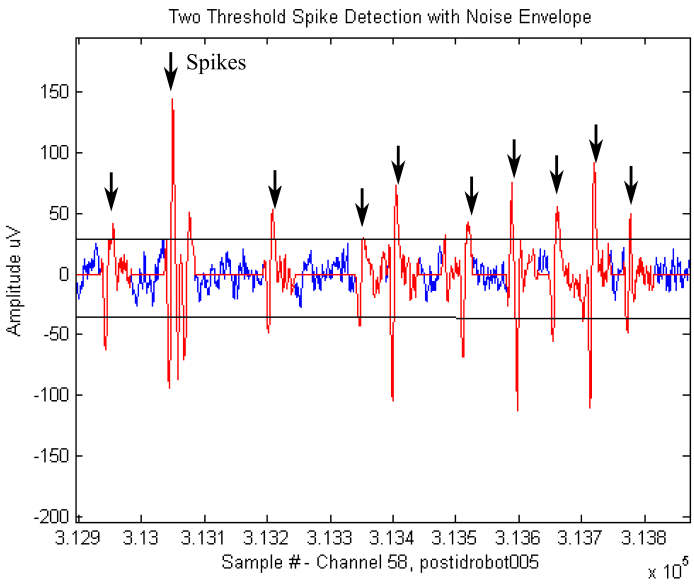


Figure 3.8: Two thresholds applied to a neural recording from the DRG with noise envelope windows.

There are 3206 detected spikes compared to the 3117 from Offline Sorter. The results for the artificial recordings are shown in Table 3.4.

3.3 Waveform Shape Compression

This section focuses on the compression of the neural spike shape after being found through detection. The noise between spikes is usually not considered for transmission. The quality

Table 3.4: Two-threshold detection on artificial recordings

Data Set	True Positives	False Detections	Total Available	Error Rate	Miss Rate
Sim1	539	115	2415	17.6%	77.8%
Sim2	1210	65	3214	5.1%	62.4%
Sim3	1136	67	3283	5.6%	65.4%
Sim4	1120	49	3193	4.2%	64.9%
Sim5	414	113	2328	21.4%	82.2%

of the compression is determined by the compression ratio, the MSE of the reconstructions, and the PCA classification. The reference example is a DRG recording containing 4617 absolute threshold spike detections. Each sample is 8 bits and the PCA classification is shown in Figure 3.9 and 3.10 with 6 identified classes. The scores are the representation of the spike detection in the principal component space. The space for the two first components is shown in Figure 3.11. A score is calculated by taking the spike detection and multiplying each sample by its respective variable and taking the sum.

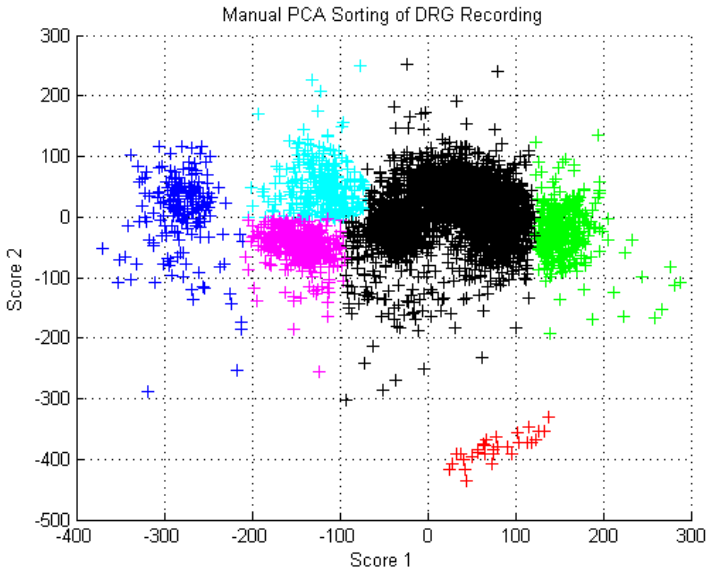


Figure 3.9: Manually-defined PCA clusters for the DRG example recording.

The artificial recordings have a known number of classes, as shown in Table 3.5. However, after spike detection using the absolute threshold, the classifications are different as shown in the detection rows of Table 3.5. False detections can sometimes create a new class, and there are missed detections as shown in the previous sections. The classification numbers shown here are used to compare the classification accuracy of the methods in this section.

3.3.1 Predictive Coding

Predictive coding can be used to guess the next sample amplitude based on previous samples. A good predictor is usually based on a model of the signal as shown in speech applications [80]. If the predictor is good, then the system can have the choice of sending the difference between the prediction and the actual shape or just that the detection occurred.

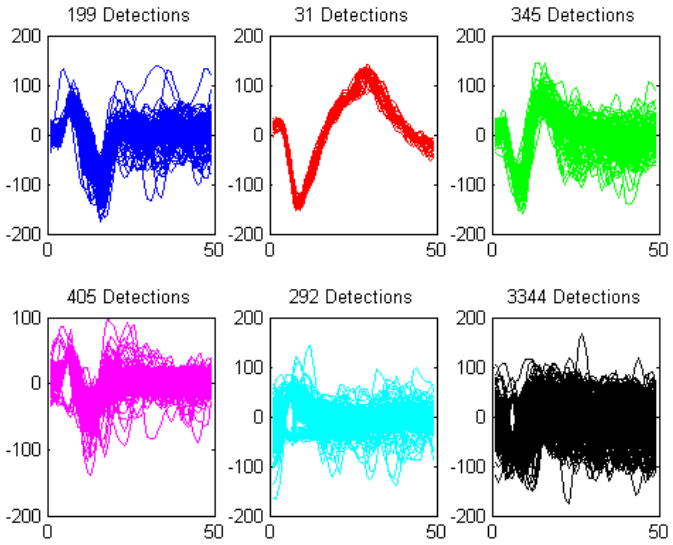


Figure 3.10: Spike detections belonging to 6 identified spike classes.

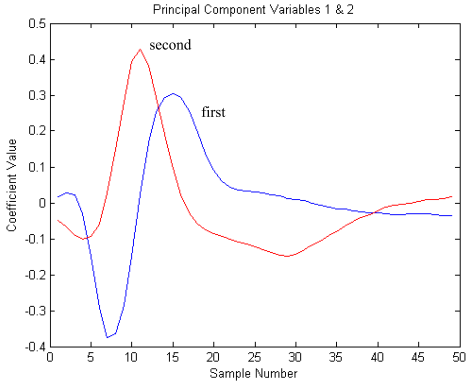


Figure 3.11: The variables belonging to the first two principal components. Blue = 1. Red = 2.

Table 3.5: Spike detections belonging to simulated classes

Data Set	Class 0	Class 1	Class 2	Class 3	Total
Sim1	2179	126	110	–	2415
Sim1 Detections	1333	126	110	42	1611
Sim2	2156	529	529	–	3214
Sim2 Detections	1271	528	527	–	2326
Sim3	2195	565	523	–	3283
Sim3 Detections	2284	159	–	–	2443
Sim4	2143	509	541	–	3193
Sim4 Detections	1902	305	154	–	2361
Sim5	2210	62	56	–	2328
Sim5 Detections	1260	37	56	73	1426

Differential Pulse Coded Modulation

An example of predictive coding is differential pulse-coded modulation (DPCM) [81], which is the application of prediction to the quantization of an error signal. A block diagram for DPCM is shown in Figure 3.12.

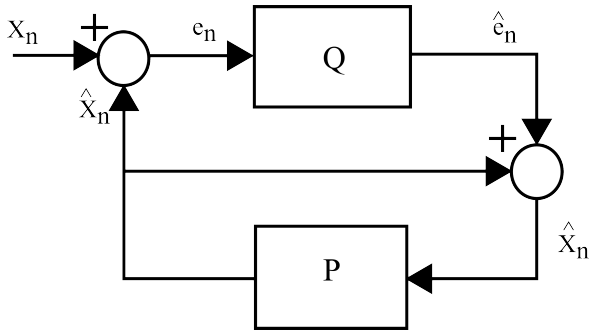


Figure 3.12: Differential Pulse Coded Modulation block diagram

The DPCM coder uses linear prediction (an all-pole predictor) of order N in Block P described by Equation 3.9.

$$\tilde{x}_n = -\sum_{i=1}^{N} a_i \hat{x}_{n-i} \tag{3.9}$$

The estimated amplitude of the signal at index n is $\tilde{x}_n$ and previously reproduced values are $\hat{x}_{n-i}$. To get the coefficients a_i one can use either an empirical set of data or base it off an already defined process, like a Gaussian process. The Levinson-Durbin algorithm can be used to build coefficients with memory size N.

If linear prediction is applied to predict spike shape, then the coefficients can be built empirically using spikes already seen. If we make the quantizer perfect we can test the performance of the predictor by setting $\hat{e}_n = e_n$. The prediction and error results for different predictors of varying memory size N are shown in Figure 3.13. The coefficients of the previous spike are used in the prediction of the next spike.

With a perfect quantizer the compression gain occurs from only transmitting the error message shown in Figure 3.14 in red. Only for $N = 2$ and $N = 3$ is the dynamic range of the error signal smaller than the original (blue) signal. The reduction is greater than 50%, which could mean that 1 bit/sample could be saved. If we assume an ADC with 8-bit samples then the result is a compression ratio of 0.8750. Calculation of the coefficients is a complicated operation, so if we assume that these values are programmable then the linear prediction coder operates as a finite impulse response (FIR) filter with a multiplication and addition for every order (stage) N. The classification and MSE performance would be identical to the examples if the full error signal is transmitted. However, if the error signal is quantized a greater amount of compression can be gained at some loss of reconstruction fidelity. If the error is binned for a maximum of 100 to (-100) for N bits the resulting average MSE for different N is shown in Figure 3.15. The reconstruction of the values requires the first value of each detection and the error signals as quantized. The resulting compression from the quantization is also shown in Figure 3.15. At $N = 4$ the compression is approximately 50 % and the MSE is 122, which gives 91.7 % accurate classification results as shown in Figure 3.16. The classification percentage is calculated by taking the summation of the differences for classes that are below the expected value $(199-179+31-30+345-12+292-262)/4616 = 384/4616$ or taking the summation of the differences for classes that are above the expected value $(3720 - 3344 + 413 - 405)/4616 = 384/4616$. The classification accuracy for the 5 artificial channels is shown in Table 3.6.

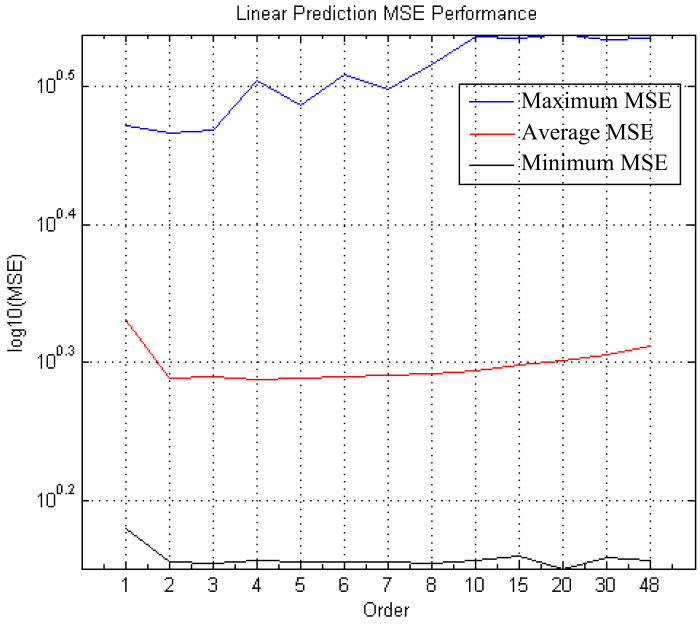


Figure 3.13: Quality of the spike prediction given a linear predictor of order N.

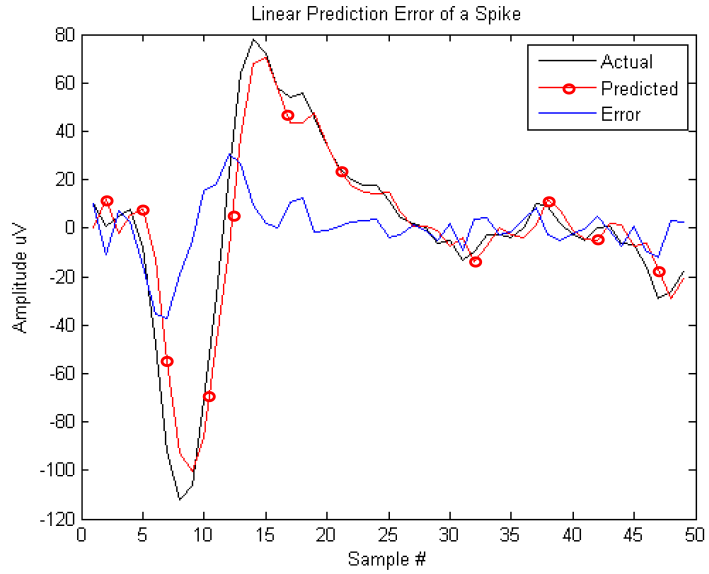


Figure 3.14: Error from the linear predictor of order 2 on an example spike.

Table 3.6: Artificial classification of linearly predicted spikes with quantization $N = 4$.

Data Set	Classification Accuracy	Comments
Sim1	85.4 %	looks like 1 cluster
Sim2	98.2 %	3 clusters close to each other
Sim3	93.5 %	looks like 1 cluster
Sim4	87.1 %	two groups visible
Sim5	88.4 %	looks like 1 cluster

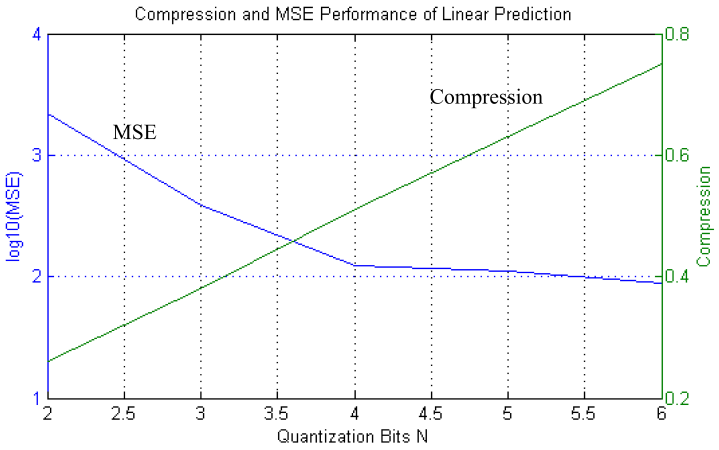


Figure 3.15: Performance of quantized linear predictor on example DRG recording.

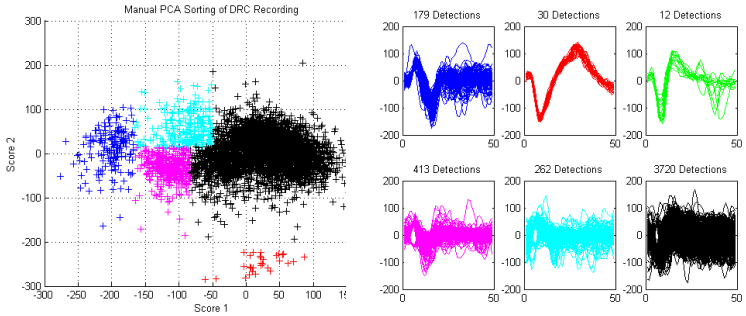


Figure 3.16: Classification results from DRG recording with quantization of 4.

3.3.2 Dictionary Coding

A dictionary (or codebook) approach is a technique for substitution coding. Substitution coding can be described as replacing one symbol or vector (pattern) with another (usually) smaller symbol or vector. In a typical substitution coding approach, the most commonly occurring vectors are replaced with shorter vectors and less commonly occurring vectors are replaced with longer vectors. The disadvantages of this approach are that the codebook must be shared (transmitted even) to the decompressor from the compressor. The codebook must be stored and the entries must have a specified format to expand for new entries and be addressed (hash tables). If there are many patterns to be covered, a small low power implementation for the compressor is unlikely. It is also possible to construct a partial codebook that is used to identify sections of a waveform shape. This is known as a syntactic codebook [82]. Codebooks can also be used to compress the error signal left after a first stage compressed waveform is removed.

Codebook approaches fare well in situations where patterns are repeated [83], which is promising when considering the low variability of spike shapes on a good SNR electrode. At least two types of codebook entries are possible: a single spike or a sequence of spikes. A sequence of spikes would not allow the end user to know the shape of the spike (except that the spikes would be considered to be very similar), so in keeping with the flexibility of the neural signal requirements described in Chapter 2, a codebook that records single spike events is appropriate.

Various formulations for selecting a codebook entry exist and each determinant can heavily impact the size of the codebook. For example some end users may consider $N = 2$ from Figure 3.14 a good enough predictor so that only a label, indicating that the predictor encountered a spike, is transmitted. Each channel could use a bank of predictors and therefore the best predictor could be chosen and then its label transmitted. Conversely, features could be extracted from the spikes and the codebook labels can be generated in the receiver.

To assess the performance of codebook entry formulation, a series of simulation results are given. The entries are characterized based on features like peak-to-peak amplitude and the location of spikes within a 1 ms frame. Each codebook is initialized with a guess entry, G_i, that candidate spike events are compared against, so that either a new entry, G_{i+1}, is created or the event is considered to be a duplicate. In Figure 3.17 is one group of plots showing the 35 entries of one particular codebook resulting from a single 60 s recording on one channel for a total 4616 spikes. The determination of entries in this codebook is defined by the following criteria:

- The peak-to-peak height and width must be within 30 μV and 10 sample locations, respectively.
- The first peaks must have the same polarity (peaks or troughs).

Over half of the spikes were captured by three entries in this dictionary, 41.8%, 11.9%, and 10.1% which means the dictionary is most likely not a good fit for the data. Other codebook formulations are summarized in Table 3.7. The discriminant features are: the peak amplitude feature requires the largest positive and negative (magnitude) values in a detected pulse to be within 25 μV of each other; the maximum slope feature calculates the steepest slope and requires it to be within 1; the P2P height and width features require the distance between the largest positive and negative values to be similar; the two maximum peaks feature forces the largest magnitude peaks to be within 25 μV and 10 sample locations; and the polarity feature separates the spikes into groups of peak first or trough first; and finally the last entry combines polarity and P2P height and width.

In order to determine if a codebook in Table 3.7 is suitable, without considering MSE, consider the implementation costs. A codebook with a small size is desirable since the

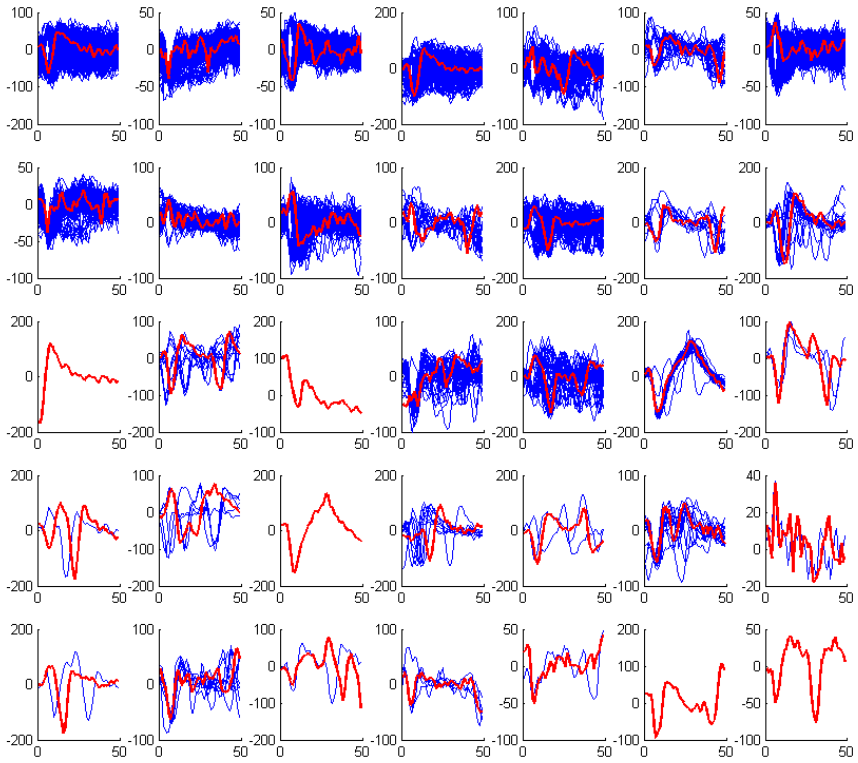


Figure 3.17: Codebook entries for a single electrode. Blue plots are the acquired signals for that entry and the red plot is the representation of that entry.

Table 3.7: Various codebooks applied to a single channel recording.

Features	Codebook Size	Top 3 %	MSE
Peak amplitude	20	93.2	737
Maximum slope	33	32.7	704
P2P height and width	19	84.7	735
Two max. peaks	40	79.2	737
Polarity	2	100	645
Polarity, P2P	35	63.8	531

implementation should be smaller and the size of entry labels is smaller. Also, most of these codebooks were initialized with a good guess for an entry. The last codebook uses 35 entries. This means that the labels would need to be 6 bits long. Therefore transmitting a dictionary containing 35 entries at 8 bits/sample with 30 samples/entry requires $35 * 8 * 30 = 8400$ bits and 4616 spikes create an additional 4616*6 = 27696 bits for the labels. The original message would contain 8 bits/samples * 30 samples/event * 4616 events = 1107840 bits for a compression ratio of 0.025. The disadvantage would be the large amount of processing required to find the appropriate entry. Each entry requires two comparisons so, in the worst case, a candidate spike could undergo 35*2 comparisons. In addition, the average MSE of all the codebook entries is 531 which is not close enough for accurate spike sorting. The PCA clusters only 35 entries, which makes clustering difficult. Therefore the scatter plot places circles for the principal components which has a diameter proportional to the number of entries as shown in Figure 3.18.

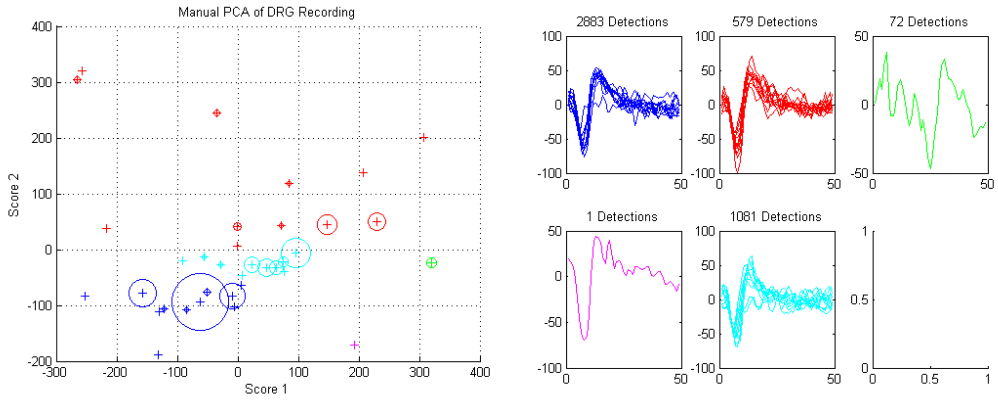


Figure 3.18: PCA plot with entry popularity proportional to diameter.

The classification accuracy for the example DRG recording is around 50%. The classification performance of the dictionary on the artificial channels is far more promising. The classification results, codebook size, and MSE are shown in Table 3.8.

Table 3.8: Dictionary coding performance on artificial data sets.

Dataset	Codebook Size	MSE	Classification Accuracy %
Sim1	14	158.8	93.4
Sim2	15	128.8	99.9
Sim3	11	175.9	99.3
Sim4	12	317.8	69.9
Sim5	10	170.4	94.3

3.3.3 Transformation Coding

Transformation coding [67,83] is a popular technique for lossy compression. The compression performance is determined by the ability of the new basis functions to accurately represent the data with the fewest possible coefficients.

Discrete Fourier Transform

The fast Fourier transform or discrete Fourier transform (DFT) is one the most widely used transforms for signal processing in digital systems. It has fast and compact hardware

implementations that are exploited in compression standards like MPEG and JPEG [67]. With the recent formulation of compressed sensing and sparse sampling, the conventional DFT transform has a runtime speed up in a new transform known as the sparse Fourier transform [84]. In this section the DFT is considered for neural spike compression. The one-dimensional DFT, as defined in [67], is shown in Equation 3.10.

$$F(k) = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} f(n) e^{-j2\pi/N} \qquad 0 \leq k \leq N-1 \tag{3.10}$$

The DFT results from each of the 4616 spike detections from a DRG recording is shown in Figure 3.19. The largest coefficients are at the beginning and end, so the initial compression attempts are to remove the small coefficients and keep the large ones.

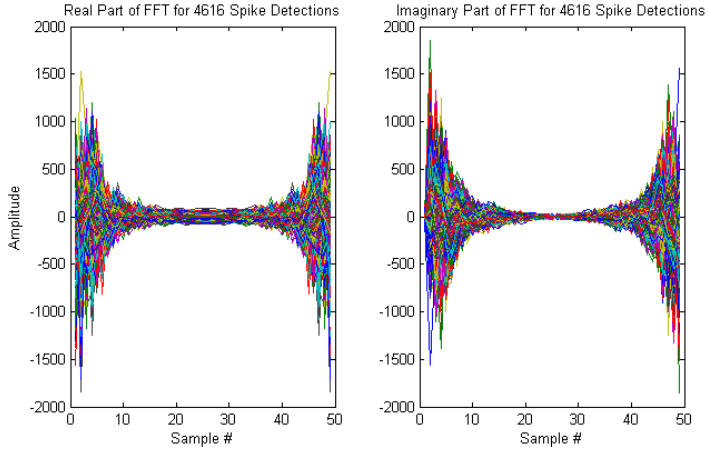


Figure 3.19: Superimposed DFT coefficients from spike detections.

The compression is sought by keeping only the K largest DFT coefficients sorted by absolute magnitude. In Figure 3.20 the value of K is increased and the resulting declining MSE is recorded for all detections. The MSE for $K > 8$ coefficients provides reconstructions with an average MSE below 100.

The MSE performance depends on K. Each coefficient is complex-valued and therefore must be treated as two values. If we assume that the retained frequencies are known beforehand, which is reasonable since spikes use the same coefficient locations, as shown in Figure 3.21, then only encoding the amplitudes in some fixed order is needed. If 8-bit coefficients are used, then ignoring the effects of quantization, the compression ratio can range from 23.5% at $K = 9$ to 52.2% at $K = 20$. The classification accuracy for the example DRG recording is 99.1% and the clustering results are shown in Figure 3.22.

The classification accuracy for $K = 10$ on the artificial channels is presented in Table 3.9.

Discrete Cosine Transform

The discrete cosine transform (DCT) is a standard transform that is used in MP3 and is known for its energy compaction [67]. Unlike the DFT, the DCT only creates real values. In this section we consider using the DCT to compress neural spikes. The one-dimensional DCT, as defined in [67], is shown in Equation 3.11.

$$F(k) = \alpha(k) \sum_{n=0}^{N-1} f(n) \left[cos \frac{(2n+1)\pi k}{2N} \right] \qquad 0 \leq k \leq N-1 \tag{3.11}$$

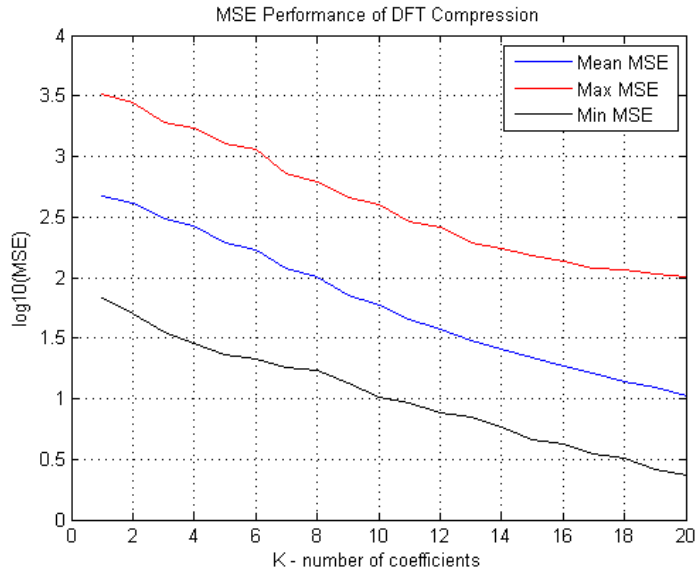


Figure 3.20: MSE compression results with the K largest coefficients.

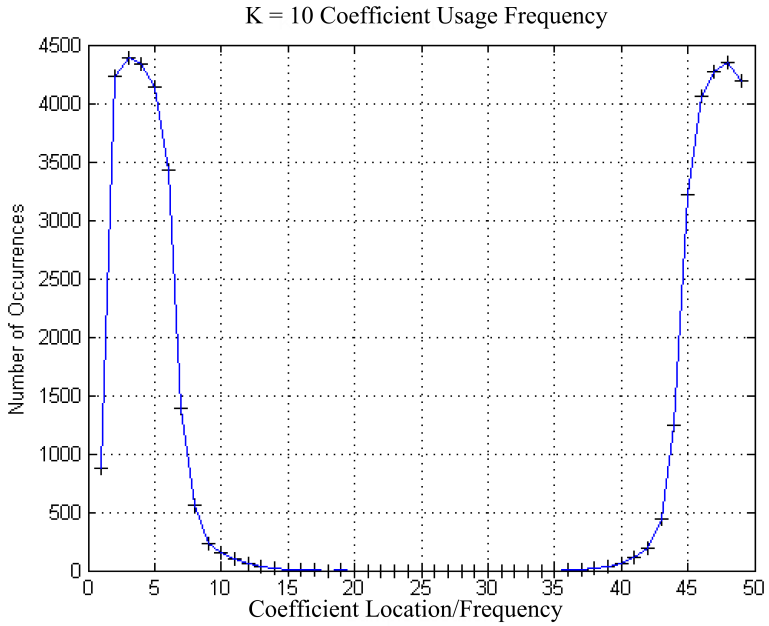


Figure 3.21: Coefficient usage frequencies for all spikes with $K = 10$.

Table 3.9: Classification accuracy for the artificial data sets using the DFT with $K = 10$ retained coefficients.

Dataset	MSE	Classification %
Sim1	27.6	99.1
Sim2	48.2	100.0
Sim3	27.7	99.8
Sim4	26.0	99.9
Sim5	18.8	99.7

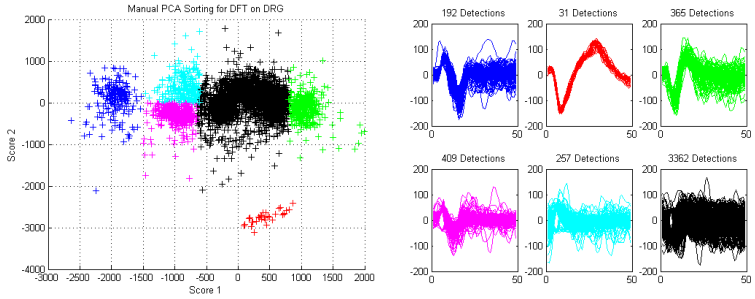


Figure 3.22: Clustering results on the DRG with $K = 10$.

where $\alpha(0) = \sqrt{1/N}$ and $\alpha(k) = \sqrt{2/N}, 1 \leq k \leq N-1$. The DCT coefficients of all the spike detections from a DRG recording are plotted in Figure 3.23.

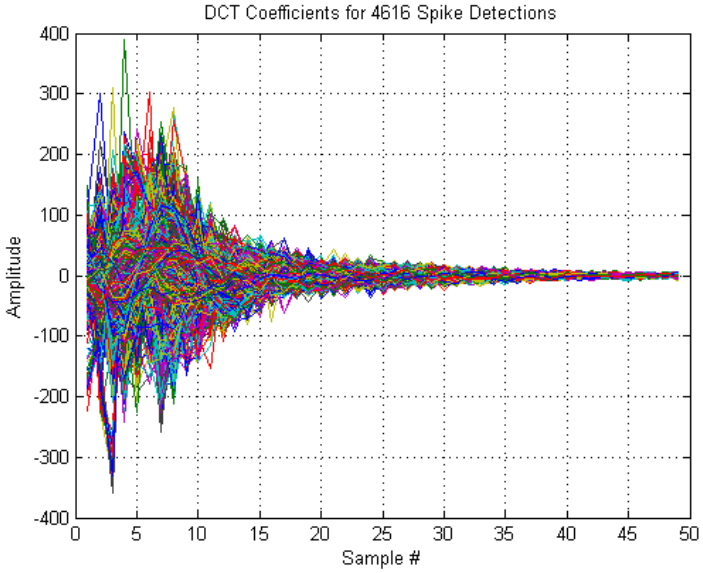


Figure 3.23: Superimposed DCT coefficients from spike detections.

The compression is performed by keeping the K largest DCT coefficients sorted by absolute magnitude for each spike. The K chosen coefficients could change with each spike, depending on the coefficients produced. In Figure 3.24 the value of K is increased and the declining MSE is recorded for all detections. The MSE for $K > 6$ coefficients provides reconstructions with an average MSE below 100.

The compression capability of the algorithm depends on K. In the DCT the locations of the coefficients can be assumed to be known beforehand. Figure 3.25 shows that the top 10 most frequently used coefficients are within the first 10 occurring coefficients, so only encoding the amplitudes is needed. The locations can be assumed with a penalty to the MSE. If 8-bit coefficients are used, then ignoring the effects of quantization, the compression ratio can range from 14.3% at $K = 7$ to 40.8% at $K = 20$. The classification accuracy for the example DRG recording is 99.7% and the clustering results are shown in Figure 3.26.

The classification accuracy for $K = 10$ on the artificial channels is presented in Table 3.10.

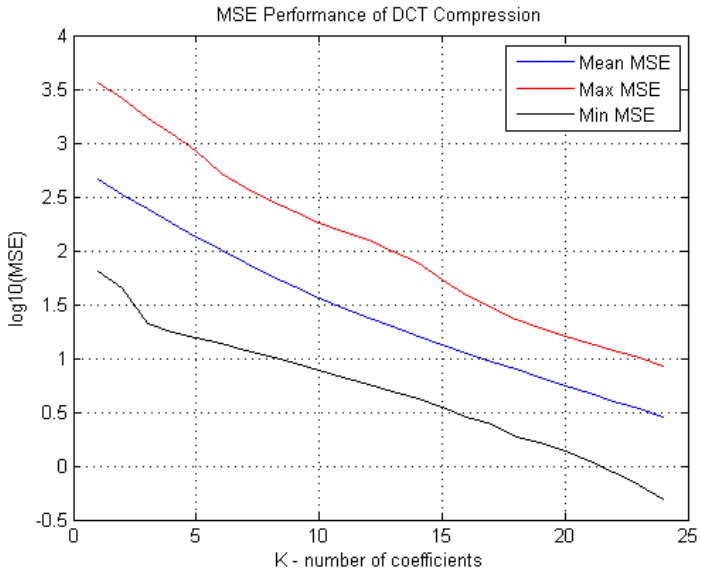


Figure 3.24: MSE compression results with the K largest coefficients.

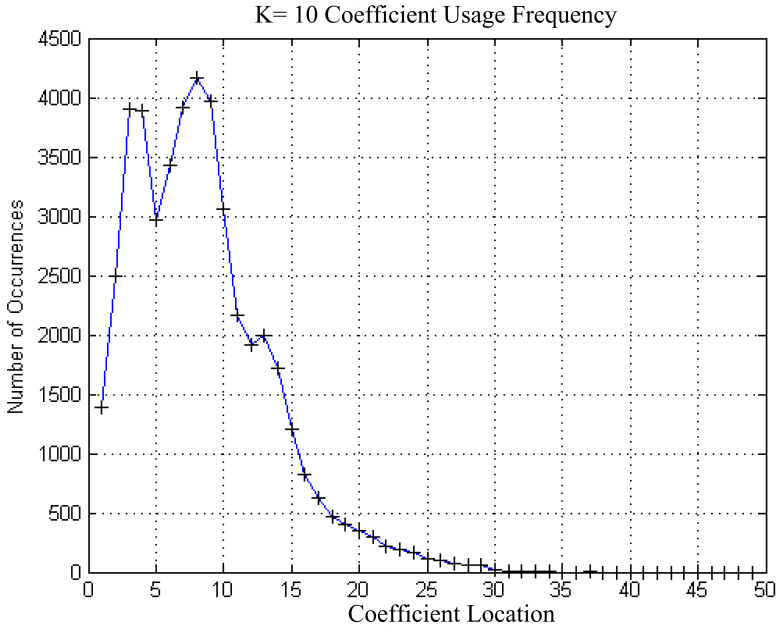


Figure 3.25: Coefficient usage frequency for all spikes with $K = 10$.

Table 3.10: DCT performance on artificial data sets with $K = 10$.

Dataset	MSE	Classification %
Sim1	17.4	99.9
Sim2	32.9	100.0
Sim3	16.5	100.0
Sim4	15.9	100.0
Sim5	12.0	99.2

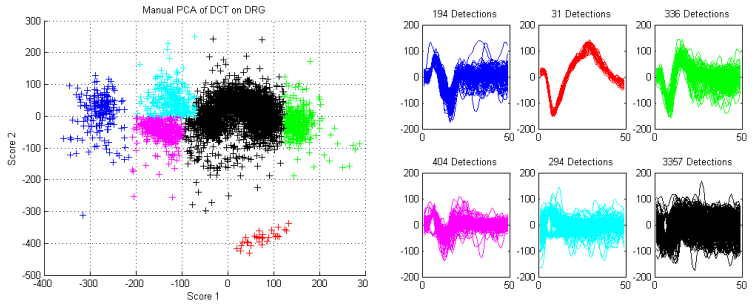


Figure 3.26: Clustering results on the DRG with $K = 10$.

Discrete Wavelet Transform

In this section we consider the one-dimensional DWT. There is already a great deal of published work on it for state-of-the-art neural recording designs as shown in Table 2.1. The DWT can be considered a combination of a transform and subband decomposition. The transform itself consists of two FIR filters: an approximation (also called low pass) filter and a detail (or high pass) filter. If we consider a discrete N point sequence $x[n]$ and use the approximation filter $h[]$ and the detail filter $g[]$ then the DWT coefficients can be found through Equations 3.12 and 3.13. The resulting coefficients of each filter can then be passed again through another filter, in such a way that we build up a multi-level transform based on a cascade of multiple stages of transform filters.

$$x_{detail}[n] = \sum_{k=-\infty}^{\infty} x[k]\, g[2n-k] \tag{3.12}$$

$$x_{approx}[n] = \sum_{k=-\infty}^{\infty} x[k]\, h[2n-k] \tag{3.13}$$

A block diagram of multi-level DWT decomposition using the tree algorithm appears in Figure 3.27. The down arrows represent down sampling by 2, and the blocks $g[-n]$ and $h[-n]$ represent the detail and approximation filters, respectively.

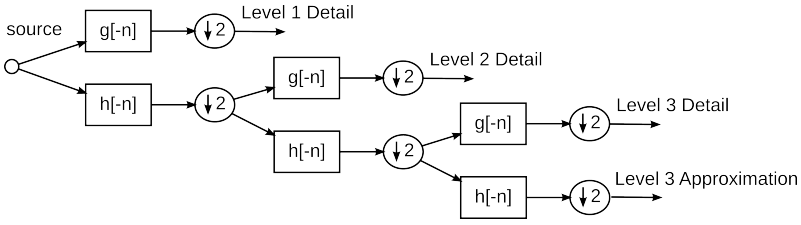


Figure 3.27: Multi-level DWT encoder using the tree algorithm

The synthesis of the reconstructed waveform from DWT coefficients is performed according to Equation 3.14. The block diagram for the reconstruction hardware datapath is shown in Figure 3.28. The up arrows represent up sampling and the index on the filters is no longer negative.

$$\hat{x}_{[}n] = \sum_{k=-\infty}^{\infty} x[k]h[2n-k] + \sum_{k=-\infty}^{\infty} x[k]g[2n-k] \tag{3.14}$$

There are two wavelet properties that determine the performance of the DWT: the number of levels and the wavelet basis. Fortunately, a good wavelet basis for neural spikes

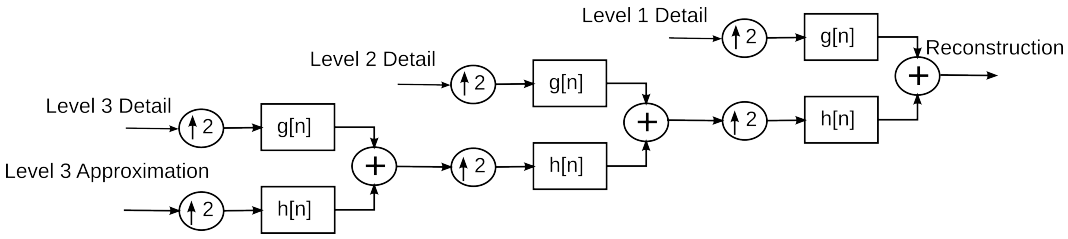


Figure 3.28: Multi-level DWT reconstruction for 3 levels.

has already been identified: the symlet-4 wavelet [28]. In order to create a fair comparison to other transforms, only 1-level of wavelet decomposition is explored here. The DWT coefficients of all the spike detections from a DRG recording are shown in Figure 3.29.

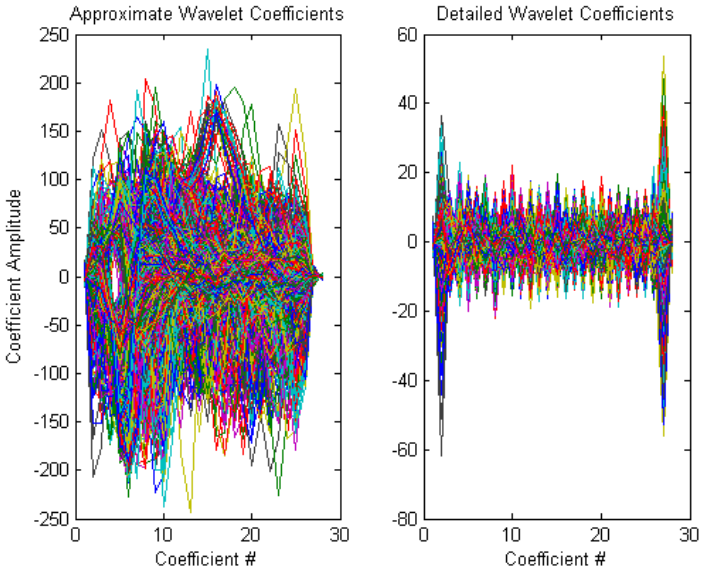


Figure 3.29: Superimposed DCT coefficients from spike detections.

The compression is performed by keeping the K largest DWT coefficients sorted by absolute magnitude. In Figure 3.30 the value of K is increased and the MSE is recorded for all detections. The MSE for $K > 7$ coefficients provides reconstructions with an average MSE below 100.

The compression performance depends on K. In the DWT the locations of the coefficients can be assumed to be known beforehand. Figure 3.31 shows that only the approximate coefficients are likely to be needed. So only encoding the K largest approximation amplitudes is needed, their locations can be fixed with an acceptable penalty to the MSE. If 8-bit coefficients are used, then ignoring the effects of quantization, the compression can range from 16.3% at $K = 8$ to 40.8% at $K = 20$. Since the first level approximate coefficients are the most important for reconstruction accuracy, it is reasonable to believe multiple-levels may provide different degrees of accuracy with each DWT level. The classification accuracy for the example DRG recording is 99.3% and the clustering results are shown in Figure 3.32.

The classification accuracy for $K = 10$ on the artificial channels is presented in Table 3.11.

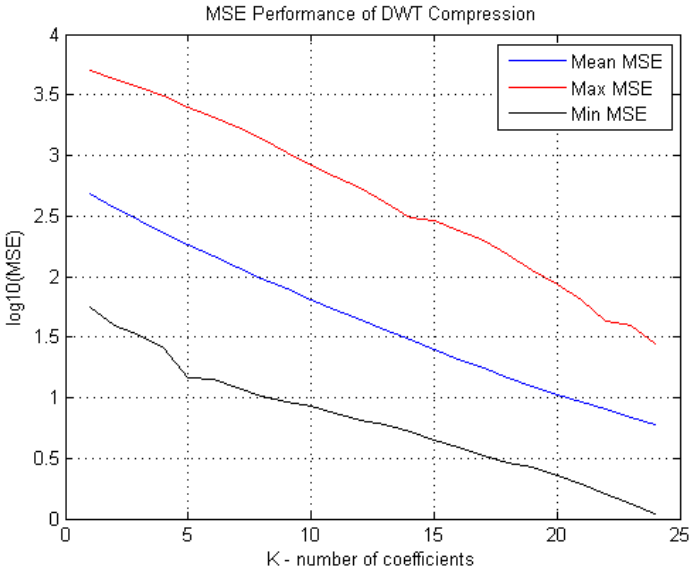


Figure 3.30: MSE compression results with the K largest coefficients.

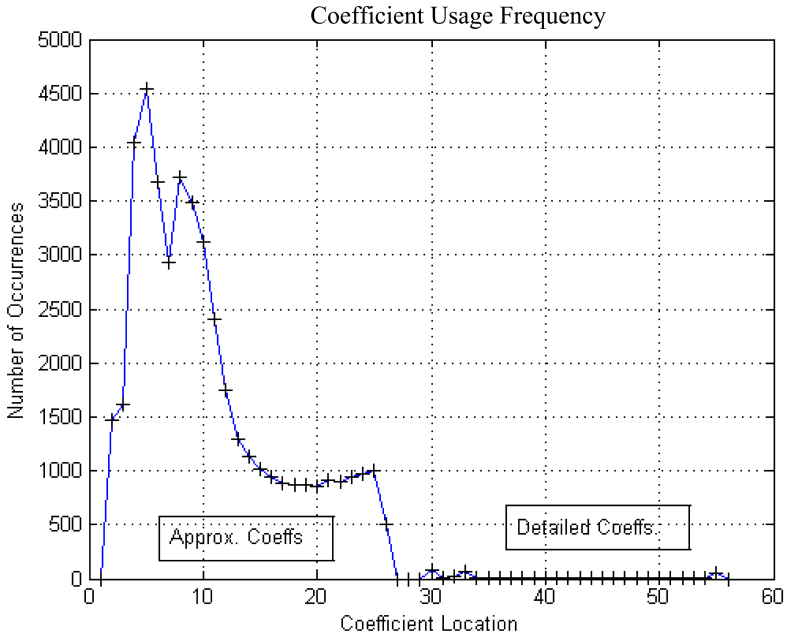


Figure 3.31: Coefficient locations for all spikes with $K = 10$.

Table 3.11: DWT accuracy on artificial data sets with $K = 10$.

Dataset	MSE	Classification %
Sim1	26.0	99.7
Sim2	67.3	100.0
Sim3	19.8	99.9
Sim4	26.9	100.0
Sim5	22.6	99.6

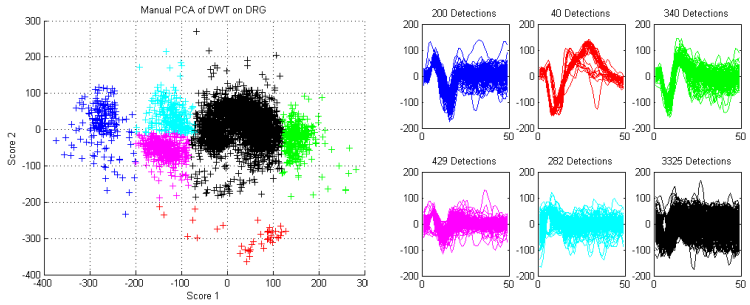


Figure 3.32: Clustering results on the DRG with $K = 10$.

3.3.4 Compressed Sampling

Compressed sampling is a recent formulation for sampling highly parallel physical interfaces or very high bandwidth/throughput interfaces. It relies on the fact that data distributed over space and time can be sampled by non-local projections instead of delta-function samples (standard ADC) [85–88]. In the neural recording environment, the bandwidth of a group of signals can be quite large for low-power wireless transmission. The compressed sensing approach aims to leverage the potential for a sparse signal space in order to reduce the transmission bandwidth. This section reviews the work done in [8] that covers signal-dependent recovery with compressed sensing and plots some simulations demonstrating model-based compressed sensing from [89] .

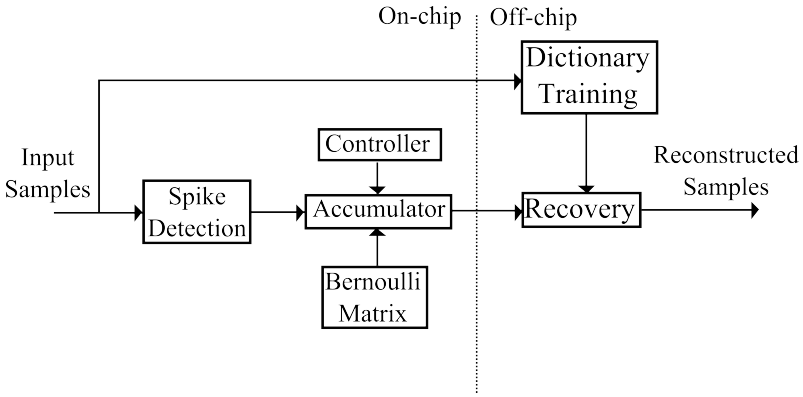


Figure 3.33: Block diagram of compressed sampling architecture from [8].

A block diagram of a compressed sampling module is shown in Figure 3.33. The front-end is very simple. It consists of a spike detection module, an accumulator and method of producing a Bernoulli Matrix (or Binary Random Matrix). A linear-feedback shift register (LFSR) or memory bank can be used to generate this sensing matrix. The controller determines how often and when the accumulated K samples are taken. The samples are taken at random locations in the spike detection and the number of samples taken affects the reconstruction performances as well as the compression. Generally, more samples means a more accurate reconstruction. However, too many samples ruins the reconstruction accuracy. The majority of the computations in compressed sampling occur in the mapping of compressed samples to a reconstruction dictionary. In [8] the dictionary is signal dependent, so this requires offline spike detection and spike classification in order to build the dictionary. In the model-based approach the reconstruction occurs from a dictionary built upon a family of wavelets. The reconstruction of spikes is based on techniques like Basis Pursuit [90]. Several papers [8, 87, 90, 91] describe similar setups for compressed sampling.

In [8] the dictionaries are split into two parts. One part is for recovery of the detected spikes and another is for the recovery of the noisy signal in between spikes. The signal dependent dictionary improves the compressed sampling performance compared to the model-based approach presented here. A spike compression rate of 8-16 is attainable with near perfect classification accuracy. The power dissipation is 0.27 μW at 20 kHz. This operational point is used for any comparisons made later. A demonstration of the model-based compressed sensing method discussed in [89] is replicated here in simulation with neural recordings.

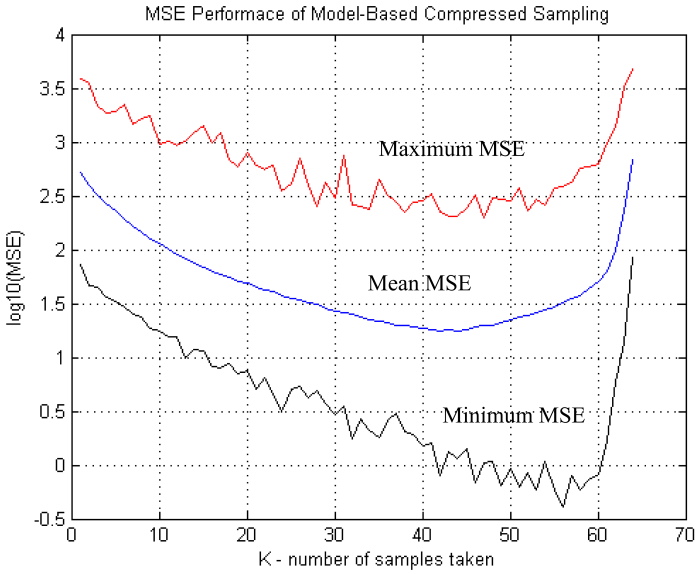


Figure 3.34: MSE compression results with K random samples.

The MSE performance of the compressed sampling approach on the neural recording example is plotted in Figure 3.34. The classification accuracy with $K = 10$ is almost perfect. The manual PCA and resulting classes are plotted in Figure 3.35. The results are different from the previous examples because the frame sizes must be a power of 2, so a spike detection window of 64 samples was used. This resulted in 331 fewer detections compared to using a spike detection window of 48 because some spikes occurred before the 64 samples was finished and were therefore missed. Accounting for this discrepancy, the classification is excellent. The artificial simulations have a similar performance, which is shown in Table 3.12.

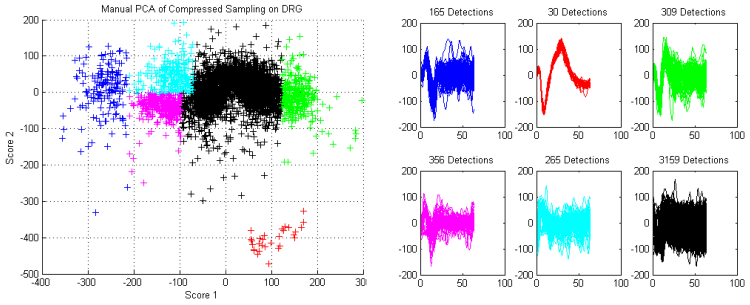


Figure 3.35: Clustering results on the DRG with $K = 10$.

Table 3.12: Compressed sensing accuracy on artificial data sets with $K = 10$.

Dataset	MSE	Classification %
Sim1	27.4	98.1
Sim2	39.3	100
Sim3	29.4	99.9
Sim4	30.9	99.8
Sim5	26.5	99.6

3.4 Discussion on Compression Methods

The current trend in integrated circuit technology, as forecast by Moore's Law [10], gives more computational capability in less area with less energy/bit than previous generations of circuits. This occurs because with each new generation, transistors are getting smaller and their switching dissipates less power. As described in ITRS 2012 [92], designers can expect improvements that will allow them to pack more processing power into smaller places. It has been assumed in this work that the energy cost (pJ/bit) of compressing data is less than the energy cost of transmitting data and therefore it is advantageous to compress the data in order to save transmission power. This may not be the case given all of the possible compression methods and their respective computational costs. Therefore characterizing the energy cost of compression circuits is vital to their potential application in implanted systems.

In this chapter spike detection, discrete wavelet transform and compressed sampling have shown the best potential compression performance. The disadvantage of threshold-based spike detection is that its success relies on the calibration of a threshold that lies beyond the channel noise, yet still captures most (preferably all) spikes. This means that the design requires user-based programmability for each threshold and/or a method of self-calibration. The compressed sampling approach described in Section 3.3.4 also requires spike detection, but has the option of capturing the noisy intervals. In addition, a frequent occurrence in the actual implanted experiments is that channels do no work at all; either through electrode damage and/or not being in close proximity to a neuron. Therefore a channel may not have any spikes above the noise and would therefore not be worth recording, so the system should be modular so that inactive channels do not consume precious power.

The following subsections contain short reviews of the sections presented in this chapter and recommend options for further work.

Spike Detection

Spike detection appears to be greatest contributor to implantable neural signal acquisition. It is also used in compressed sampling to select the proper reconstruction dictionary. Improvements to spike detection could drastically increase system performance, since threshold detection often captures noise events that are thrown away later during spike sorting and misses spikes that have low SNR. Spike shapes and noise levels change over time and the automation of adjusting the spike detection to compensate allows the scalability of the implants to reach 100's over multiple sites as user supervision becomes impracticably time consuming. Therefore the implementation of a method of spike detection that reduces false detections, misses fewer spikes and can adaptively change to the neural signal recording environment is explored in Chapter 4.

Linear Predictive Coding

The calculation of coefficients for the predictive filter is an expensive operation consisting of many operations. However, given that spikes tend to have the same general shapes, the coefficients could be calculated and programmed during a training phase or a set of linear predictors could be used and only the best one selected. However, the resulting compression of 50% may not be worth the energy cost of applying the second order linear prediction filter and other methods yield higher compression gains, which is more suitable for a large scalability. Therefore linear predictive coding is not considered for further development.

Dictionary Coding

The use of substitution coding means developing discriminants for entries to determine suitable substitutions. Calculating a discriminant could involve multiple comparisons and arithmetic operations. The formation of discriminants in the codebooks presented is based on values that are easily calculated and then matching those values to the nearest entry (template matching). If the codebook is efficient at representing the spikes seen on the channel then it performs a large amount of compression, however the size of the codebook could grow very large. The codebooks could be improved if recordings are analysed by statistical software suites like the statistical package for the social sciences (SPSS), in which the contributions of discriminants could be compared against each other. A major concern for dictionary coding is the architecture of the implementation and the energy cost of comparing multiple entries to a single candidate event. The impressive compression gains of the dictionary approach along with the suitable classification rates means that it is worth further investigation, which is the focus of Chapter 6.

Transformation Coding

This chapter covered three types of transformation coding for the neural signal: Fourier, cosine and wavelet. All of them created reconstructions with low MSE and high classification rates. Therefore the compression gains and cost need to be considered. The DCT and DWT have the best compression gains, but the wavelet has the opportunity to improve the compression by adding another level of decomposition. The second part of the compression gains is the assumption that the locations of the coefficients can be averaged and therefore known *apriori*. This is investigated in Chapter 5. The application of the DWT for transformation coding also has well known circuits and proven performance shown in the literature. The desirable feature of DWT is that it can trade off operational cost with compression gain with more operations/level yielding a greater compression ratio. Therefore Chapter 5 presents work using the DWT and presents the subsequent implementation of a multilevel wavelet transform.

Compressed Sampling

Compressed sampling is a relatively new technique that can give compression ratios of 80-88% [49]. It provides excellent compression, while also allowing accurate classification at a low-cost transmission side implementation. The work presented in [49] and [8] is far along in development and at this time the only other approach to be taken is to alter the formulation of the dictionaries and or optimizing the circuits used. Compressed sampling is a potentially powerful method for neural signal recording, but is left as future work.

Overview

The overall purpose of the neural recording device is to provide an observer to the targeted neurons. Each implant may have a different application, from prosthetic device interfaces to

the study of DRG motor and sensor signals. Each application may require a different fidelity or representation of the neural signal. Therefore the vision for the progression of the thesis work is towards a neural oscilloscope. Such an instrument would provide the user with the ability to control the accuracy of reconstructions, from full raw recordings to compressed spike detections and spike rates. Many artificial neural control loops operate on spike rates, but in order to have confidence in those spike rates, a progression from fully reconstructed neural signals to detected neural spike reconstructions to spike rates may be needed. This way clustering algorithms like PCA and k-means sorting may be used to confirm the identity of spiking classes.

Chapter 4

Neural Spike Detection: Simulation, Implementation, and Results

In this chapter the simulation and implementation of a new neural spike detection algorithm is presented. The aim of the algorithm is to reduce false detections and lower the miss rate compared to the detection methods covered in the previous chapter, while having an energy cost per bit less than uncompressed wireless transmission. The first section presents the formulation and simulation results of the spike detection method. The second section covers the implementation of a 16-channel neural spike detection circuit and measurements from a fabricated chip. The chapter is concluded with a discussion and comparison on the results.

4.1 Spike Detection

In the previous chapter four methods of threshold-based spike detection were reviewed: average noise, noise envelope, NEO, and two thresholds in a window (dual-threshold). The average noise and dual-threshold methods achieved the lowest false detection rates of approximately 4%, while the noise envelope method (with single threshold) achieved the lowest miss rate of 15%. Therefore the first generation of the spike detection algorithm combines elements of dual-threshold detection and noise envelope detection since noise-based thresholds did not perform as well as the noise envelope method.

The application of a noise envelope for both positive and negative amplitudes allows the calibration of two separate thresholds. It also may allow the user to selectively find only positive peaks or negative troughs. The method of noise envelope calculation from [76] is repeated here:

Calculate the maximum absolute value in the window, $Mx_i = max(|x[n]|)$.
If $Mx_i > KEn_{i-1}$, assume there is a spike and ignore the window otherwise
if $Mx_i > En_{i-1}$, $En_i = \alpha * Mx_i + (1-\alpha) * En_{i-1}$ or
if $Mx_i < En_{i-1}$, $En_i = \beta * Mx_i + (1-\beta) * En_{i-1}$

The authors of [76] set $\alpha = 0.02$ and $\beta = 0.5$ for 50-ms windows. The value of K is set to 4. In simulations the signal voltage amplitudes are stored in μV with floating point precision, but in the implemented system integer-based representations are used so a multiplication by a factor of 0.02 (0.98) and 0.5 can be potentially expensive unless we perform it by shifting the integer left to get a division by a power of 2. This way we can attain

a factor of 0.03125 and 0.5 by a left shift of 5 and 1 places, respectively, but cannot get the 0.98 without additional operations. Alternatively, we can use an adder to add/subtract values onto the estimates. Assume that the previous positive noise envelope $En_{i-1} = 20$ μV and a maximum value of $Mx_i = 70$ μV is found in the current window, which satisfies $< K * En_{i-1}$. The new noise envelope estimate is $En_i = 1.4 + 19.6 = 21$, which could have also been accomplished by just adding the value of '1' to the previous window. If the maximum value was $Mx_i = 18$ μV then $En_i = 9 + 10 = 19$, which could have been accomplished by subtracting a value of '1' from the previous window. This method of changing thresholds is also shown in the work of [93]. The new algorithm for noise envelope calibration contains increments or decrements instead of multiplications, as shown below.

Calculate the maximum absolute value in the window, $Mx_i = max(|x[n]|)$.
If $Mx_i > KEn_{i-1}$, assume there is a spike and ignore the window otherwise
if $Mx_i > En_{i-1}$, $En_i = En_{i-1} + INC$ or
if $Mx_i < En_{i-1}$, $En_i = En_{i-1} - INC$

where INC is the value of the decrement or increment. We implement a threshold, used to determine spike detection, that can be placed right at the noise envelope or above it. This provides a modifier to control the aggressiveness of the spike detection, so we can reduce the number of false detections on channels with known large spikes. The noise envelope frame size from [76] is set to 50 ms and reducing it creates a tighter envelope as shown in Figure 4.1. A tighter envelope results in an increase in the number of total detections. However, the dual-threshold method still has a high rate of missed detections in the simulated data sets as shown in the previous chapter. To counteract this class of error, a monophasic spike, the option to choose only one threshold or the ability to independently adjust the negative threshold is needed. This may increases the number of false detections, but no greater than the other methods. The monophasic spikes can be manually adjusted by moving the negative threshold above (positively) the negative envelope. This way the trough of the spike passes through the negative threshold. Therefore the system has the ability to selectively pick strong biphasic neural signals from the recording and the ability to aggressively find small spikes or zoom out and find only large spikes.

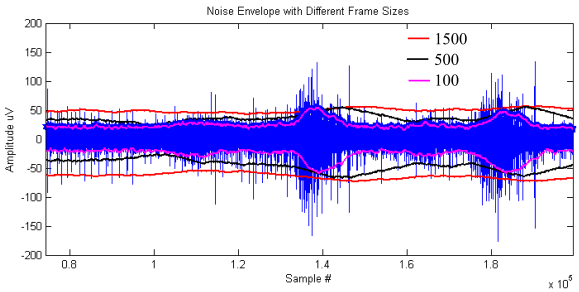


Figure 4.1: Positive and negative envelopes with different frame size.

Another feature of the algorithm is that it is adaptive to the noise in the channel [76]. However, it is susceptible to dramatic and fast changes as shown in Figure 4.2. Therefore two changes are introduced. A quick check is added to each frame calculation to ensure that a burst of spikes just above the envelope does not cause the failure of detecting smaller spikes right after and a smaller high limit check is used as demonstrated with the black plot in Figure 4.2. The $K = 4$ is replaced with an additive limit, set to 25 μV.

The resulting improved spike detection algorithm is shown below. It works well unsupervised as it adjusts to the noise envelope of the channel, but it can be customized to each channel with supervised modification to the frame size, high limit, and increment values.

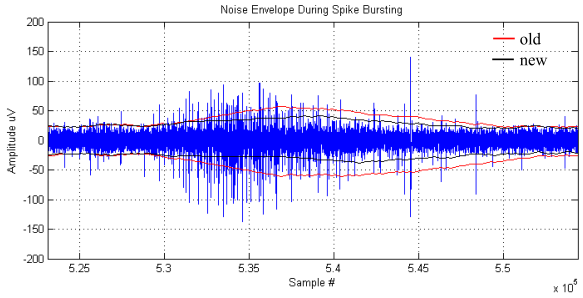


Figure 4.2: Noise envelope not compensating for fast neural signal changes.

If needed, the thresholds can be manually adjusted away from the noise to decrease the number of false detections. The simulation results are shown in the next section and are compared to the state-of-the-art.

Calculate the maximum absolute value in the window, $Mx_i = max(|x[n]|)$.

If $Mx_i > En_{i-1} + \text{HighLimit}$

$En_i = En_{i-1}$

else if $Mx_i > En_{i-1}$

$En_i = En_{i-1} + INC$

else if $Mx_i < En_{i-1} - HighLimit$

$En_i = Mx_i$

else

$En_i = En_{i-1} - INC$

4.1.1 Simulation Results

The simulation results of a spike detection algorithm depend heavily on the noise and neural spikes contained in the test recording. When the SNR is low the false detection rates and miss rates are expected to increase compared to recordings with high SNR. Figure 4.3 shows the detection performance found for Sim1 from the five artificial data sets in [73]. The plots are organized into three adaptations on the same data set by manually adjusting the available thresholds to attain an expected error rate. Red bars are relatively high error rates of approximately 74%, black bars are medium error rates of about 48%, and finally the blue bars are low error rates of 1-3%. Sim1 has 2415 available spikes for each possible detection.

If we look at the high error rate results (red bars) and the middle error rate (black bars), the difference in the algorithms occurs in the miss rate (bottom right). The proposed algorithm does 2% percent better than the others. In the low error rate (blue bars) region the miss rate is a fraction of a percent worse than for the absolute threshold method, but the number of false detections is lower. The detection rates for the remaining simulation sets are shown in Table 4.1. The 95% confidence intervals are $\pm$ 2.12 and $\pm$ 1.71 for the error rate and miss rate, respectively. More simulations, with known spiking behaviour, are needed to establish statistical significance. Only five artificial recordings are available and the possible error in the findings is calculated to be 2%. The values of 10 and 20 in Table 4.1 are offsets to the noise envelope based thresholds, which are used to compensate for the monophasic spikes seen in the artificial recordings [73]. On the simulated data set, which incorporates a larger variance of spike types, the proposed algorithm matches the other state-of-the art methods. However, the proposed algorithm has better performance on the

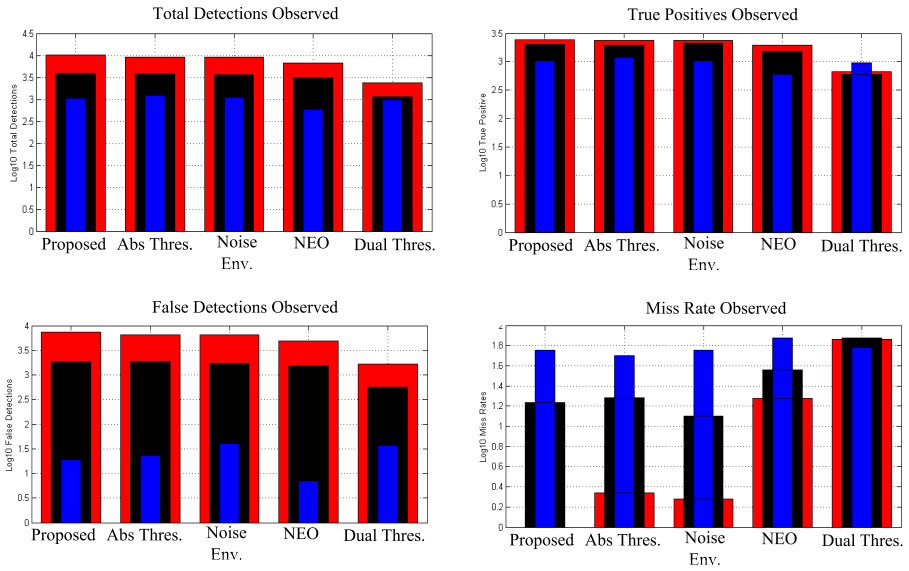


Figure 4.3: Detection performance at different error rates: low (blue) - 1-3%, middle (black) - 48%, high (red) - 74%. Top left: total detections. Top right: true positives. Bottom left: false detections. Bottom right: miss rates.

feline neural recordings from the DRG. The example recording has 86 false detections and detects 71% of the actual offline spikes. However, only 60% of the original 3117 spikes were used in the spike sorting method, leaving the remainder unused, which makes the effective miss rate 0 and the false detection penalty small. The identification of useful spikes was set by users running the neural recording experiments and their identification of useful spike detections for their neural recording protocol. It is this selection process that the dual-threshold noise envelope tries to mimic and therefore attain the useful spike detections in the DRG recordings.

Table 4.1: Noise envelope dual-threshold detection on artificial recordings

Data Set	True Positives	False Detections	Total Available	Error Rate	Miss Rate
Sim1 (10,20)	1457	140	2415	8.8%	39.7%
Sim2 (10,20)	2310	104	3214	4.3%	28.1%
Sim3 (10,20)	2305	120	3283	4.9%	29.8%
Sim4 (10,20)	2266	101	3193	4.3%	29.0%
Sim5 (10,20)	1287	128	2328	9.0%	44.7%

The main feature of the proposed algorithm is the ability to create a trigger window to capture any neural spikes larger than the thresholds by adjusting them above and below the noise envelope, as shown in Table 4.1. The positive threshold was placed 10 μV above the noise envelope and the negative threshold was placed 20 μV above the negative noise envelope to compensate for the monophasic spikes. Ideally, in future work, this kind of compensation could be automatic.

Overall, the small number of false detections from the proposed algorithm should reduce the total power dissipation of the implementation. The feline DRG recording shows the greatest improvement for the proposed algorithm, which highlights the advantage of the system. The system can detect spikes required for spike sorting in the feline DRG with fewer unwanted detections being transmitted. The artificial signals from [73] demonstrated

that the algorithm matches or exceeds the performance of competing algorithms provided that the noise envelope is adjusted to compensate for the monophasic spikes. The algorithm can provide a flexible tool that can capture neural spikes as desired. The next section covers the implementation of an adaptable dual-threshold neural spike detector.

4.2 Spike Detection Circuit Implementation

This section details the implementation of an ASIC test chip, NC1, for the proposed noise envelope dual-threshold spike detector. The design is specified in very-high-speed-integrated-circuit hardware description language (VHDL), simulated in ModelSim, and then synthesized in Synopsys Design Compiler and Cadence First Encounter using the 130-nm IBM technology library.

The filtered analog data from a recording MEA is assumed to be multiplexed before going through a single ADC. The ADC is most likely similar to the ultra-low-power successive approximation ADC from [94]. We assume that the ADC provides parallel[1] sign-magnitude 8-bit samples at a sampling rate of 30 kSps per recording channel. The logic style is constrained to conventional CMOS, although considerations for low-power digital libraries, like complementary pass-gate logic (CPL) style circuits and operation in the sub-threshold regime, could be added in the future [95].

The proposed spike detection circuit operates on 16 channels simultaneously in a multiplexed method as shown in the block diagram of Figure 4.4. A serial peripheral interface (SPI) controller is included to allow users to specify the channel parameters and a separate test channel is included to provide performance measurements from a single channel only. The SPI itself may be used for initial calibration of the implanted detector or ongoing user intervention from a wireless link.

In Figure 4.4 there are two clock domains: CLK and SLOW_CLK. The CLK is operating at 16 times the frequency of the SLOW_CLK to compensate for the 16 multiplexed channels. Therefore the input stream must contain 16 samples, one from each channel in a loop. The spike detection channels are controlled by registers which are programmable through the SPI, but also have a default mode of operation. The default is provided so that the detector can operate with no user intervention at all. The processed data from the channels is combined into a single fast output stream for analysis in a field-programmable gate array (FPGA). In order for the spike detector to be used in a wireless implant, spike timings and channel identification numbers need to be concatenated onto spike detections. This part of the architecture is not implemented in NC1. Each channel contains a user-programmable spike detection module. A block diagram of the spike detector design is shown in Figure 4.5.

The spike detector has two modules. One that calculates the noise envelope and another that performs the different modes of spike detection. There are four modes available: 'Always On', 'Positive Only', 'Negative Only', and 'Dual'. The 'Always On' mode operates like a data pass-through. In this mode the noise envelope is not needed and can be clock-gated off by setting the Control Register (0). The 'Positive Only' mode detects neural spikes with only a positive threshold, which can either be set externally or determined from the noise envelope. The 'Negative Only' mode detects neural spikes with only a negative threshold and finally the 'Dual' mode detects neural spikes using the dual-threshold spike detection method. The thresholds can be set from the external registers or the noise envelope. The signal 'Signal Ready' provides a single bit spike detection indication. The spike detection module is clock-gated with the Control Register (1) in case the channel should be turned off. The modules are described in greater detail in the next two subsections.

[1] The choice of parallel transmission has implications on the digital system and should be explored further.

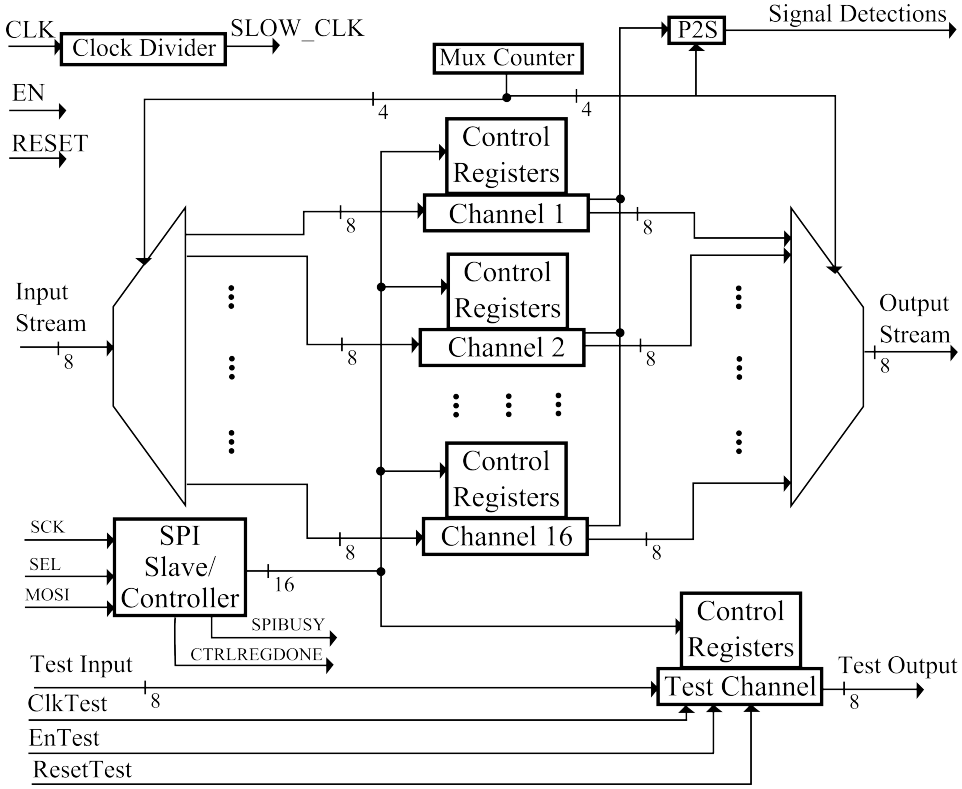


Figure 4.4: NC1 system overview.

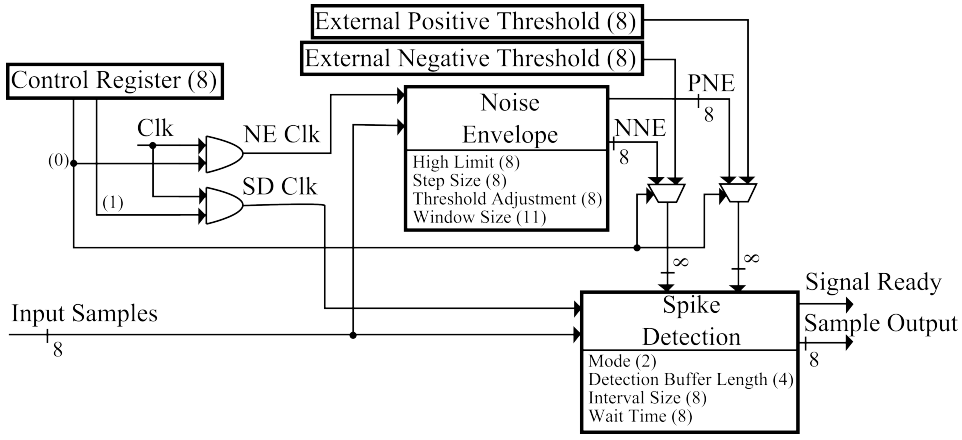


Figure 4.5: Neural spike detection system overview.

4.2.1 Noise Envelope Module

The noise envelope module is shown in Figure 4.6. If enabled the system makes a comparison every clock cycle in order to find the maximum or minimum value in the window. The input samples are separated into either a positive-side or negative-side. This is done to allow an easy method of implementing the different detection modes. At the end of every window, determined by the window controller and window size, the noise envelope is calculated and updated. In addition the window step size, the limits of spike detection and the placement of the thresholds, relative to the envelope, can be programmed from SPI registers. These features are included to give the neural recording a functionality similar to an oscilloscope trigger.

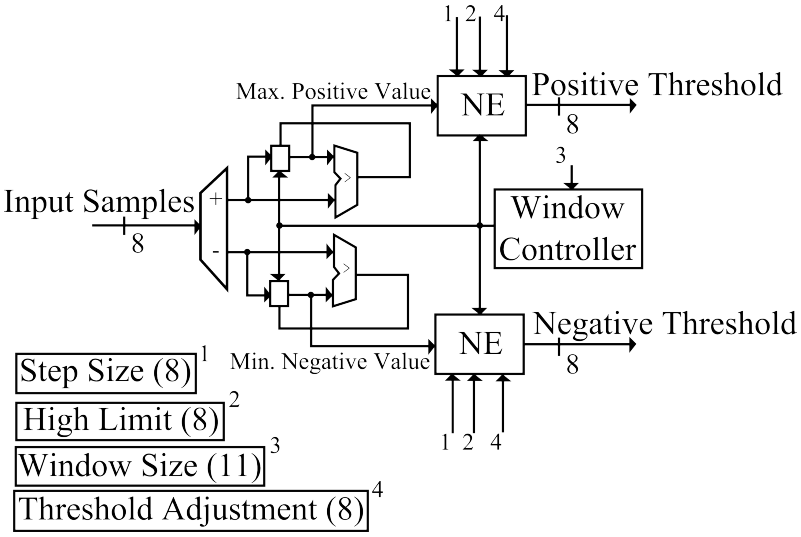


Figure 4.6: Components for calculation of the neural noise envelope.

A power saving feature of the system design is that by increasing the size of the window, the noise envelope calculation occurs less often than when the system uses smaller windows. This change has to be initiated externally for this version. All components are synchronous, but the controller has an asynchronous reset, so that it may be initialized before clock arrival in case the channel is being turned on. The window controller, on start-up and soft reset, initializes the envelope to the maximum value within the first 60 samples. It is possible that a spike could skew the envelope, but the envelope drops with each successive window that passes. Therefore, during the initialization of the system, it is better to use smaller windows which will quickly find and track the envelope.

4.2.2 Spike Detection Module

The spike detection module is shown in Figure 4.7. The programmable registers can change the behaviour of the spike detection, but the default mode of operation is 'Dual' (to use both thresholds), with a 48-sample interval, waiting up to 20 samples between threshold crossings, and buffering 6 samples before the first threshold-crossing.

All samples are continuously written into the register memory by the write address counter. This is done in order to capture the short duration (up to 16 samples) of the signal before the spike detection. The state machine enables and initializes the read address counter for sample output. The four programmable SPI registers allow the state machine to change the mode of operation, the wait time between spikes (dual-threshold mode), the number of samples to capture during a spike detection, and the number of samples to capture before

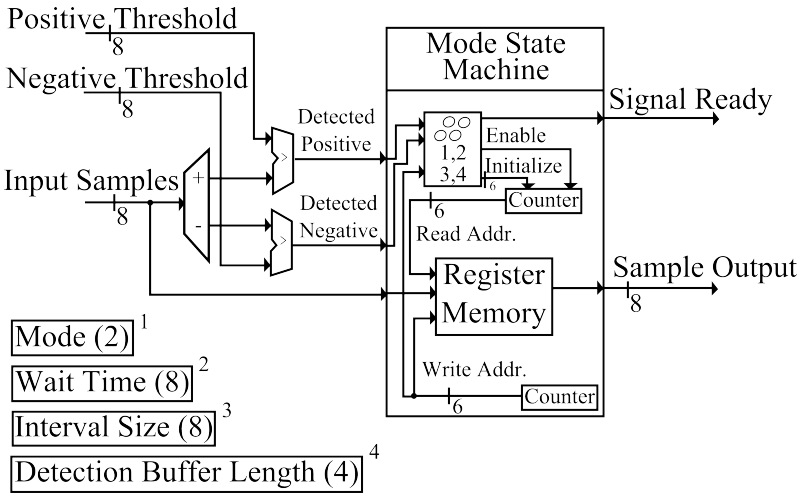


Figure 4.7: Components for detecting neural spikes using thresholds.

the first threshold crossing. The state machine tracks the interval size and wait times with two counters. If the mode is set to 'Always On', the sample data is written and read every cycle. A memory bypass would be more efficient in the next design iteration. The register memory is not typical, but has gated registers. The data is available for all the registers, but only the register of interest is clocked and its contents written. The register reading operation occurs before the write operation. A shift register architecture was avoided for the memory register since a shift register's switching activity increases as the buffer grows, while this architecture is at most only writing one register per cycle.

4.3 NC1 Simulations and Measurements

The logical equivalence between the synthesized designs and Matlab simulations is verified using netlists from Synopsys Design Compiler and Cadence First Encounter for post-layout confirmation. The measurements are saved as ASCII files and supplied back into Matlab for comparison. The design flow stages are shown in Figure 4.8. At each stage of simulation if the system has a bug or is not able to meet the required clock frequency then the VHDL modules are modified and the flow started anew. Each synthesis stage adds more complexity and lengthens the amount of time required to simulate the system. The post-layout Verilog files are simulated with parasitic timing analysis and activity-aware power calculations. The VHDL test benches use an ideal clock with a 500-kHz frequency, which supplies an input data rate of 4 Mbps. The input data is extracted from portions of recordings from the feline DRG experiments.

The synthesis of a VHDL design specification requires the selection of a process technology. The technologies available are 130 nm, 90 nm, and 65 nm. The 130-nm technology node also has optional low-power transistors. The threshold detection module is used to see the different effects the technology has on area and power as shown in Table 4.2. Only results from Synopsys Design Compiler are considered at this stage. A 30,000 ns clock cycle is used in the typical-typical simulation corner of the technology at 25 C. The low-power 130-nm technology has the best power performance since the predicted leakage power is lower than the other technologies.

The synthesis results for the noise envelope module in the 130-nm low-power technology node are summarized in Table 4.3.

The synthesized layout of the first test chip NC1, in the 130-nm low-power technology,

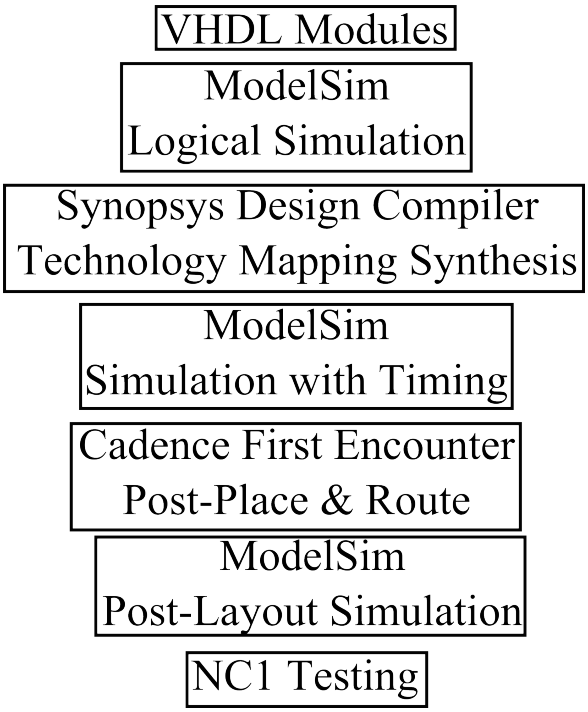


Figure 4.8: Stages of the chip design.

Table 4.2: Power and area estimates for the threshold detection module

Metric	130 nm LP	90 nm	65 nm
Voltage	1.2 V	1 V	1 V
Area (μm^2)	29,890	15,011	8472
Timing (slack ns)	29,993	29,998	29,999
Dynamic Power (nW)	131.3	112.2	58.9
Leakage Power (nW)	34.91	21,270	40,000

Table 4.3: Power and area estimates for the noise envelope module

Metric	Noise Envelope
Area (μm^2)	7202
Timing (slack ns)	29,991
Dynamic Power (nW)	62.12
Leakage Power (nW)	9.0751

created in Cadence First Encounter is in Figure 4.9. Three different areas are highlighted. The red highlighted area shows where the modules for the 16 channels are located. The green highlighted area shows where the test channel circuitry is placed and the blue highlighted area shows where the SPI modules and control registers are. The synthesized power and area results for NC1 are in Table 4.4. NC1 is 2x2mm and the equivalent number of gates is 189,290 with each gate being 4.32 μm^2.

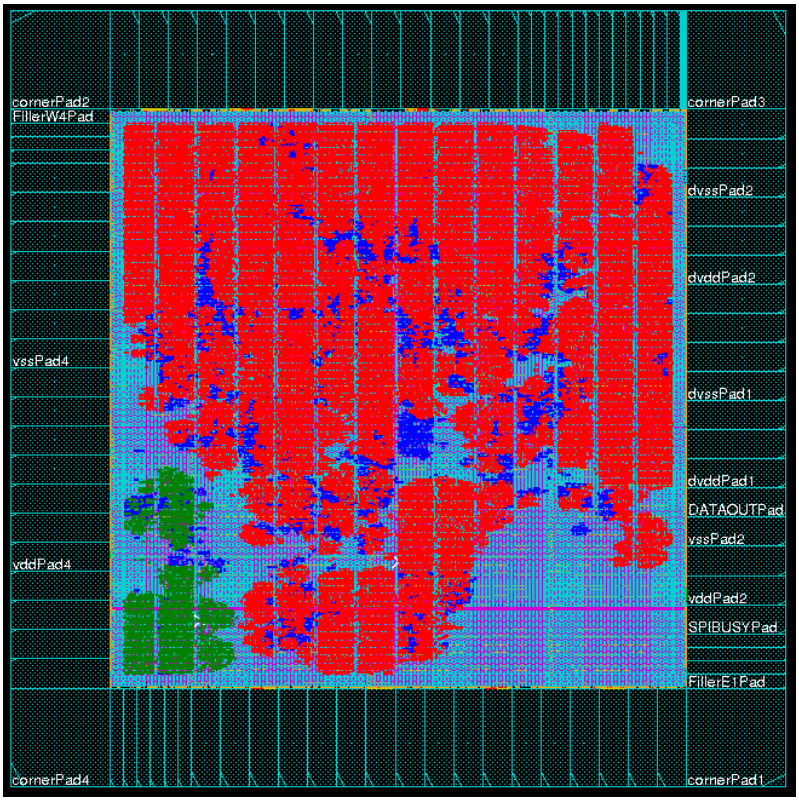
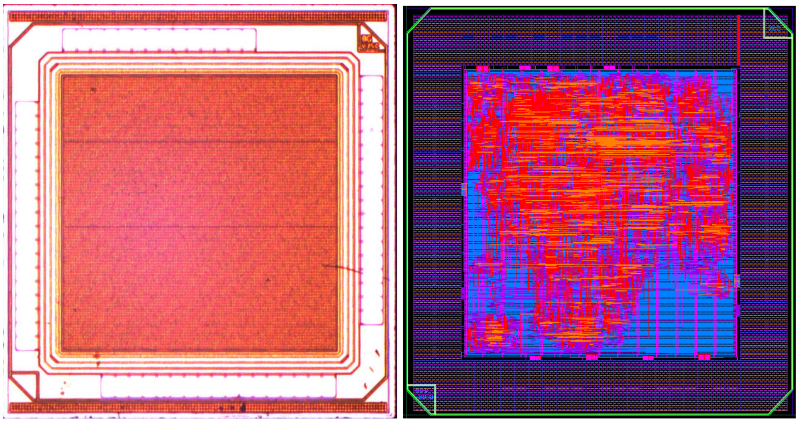


Figure 4.9: Placement of NC1 modules. Red = spike detection 84.2%. Green = test channel 5.3%. Blue = control circuitry 10.5%.

Table 4.4: Power and area estimates for NC1

Metric	Synopsys Design Compiler	Cadence First Encounter
Voltage	1.2 V	1.08 V
Area (μm^2)	875,538	817,733
Timing (slack ns)	1993	1849
Dynamic Power (μW)	41.2936	85
Leakage Power (μW)	1.0159	2732

The fabricated layout of NC1 is shown in Figure 4.10 (b) and Figure 4.10 (a) is a picture of one of the chips. Logical equivalence was verified through all stages of the NC1 design. A design flaw was found during testing that was not found during simulations. The flaw does not permit any of the registers to be written and they are therefore always at default values. The cause of the error is optimization based, as explained in the next subsection. Otherwise, the NC1 chip works exactly as expected. An example of the input and output with NC1 is shown in Figure 4.11. The output is delayed compared to the input and latches at the last known value of the window.



(a) (b)

Figure 4.10: NC1 chip and NC1 layout from Cadence Virtuoso

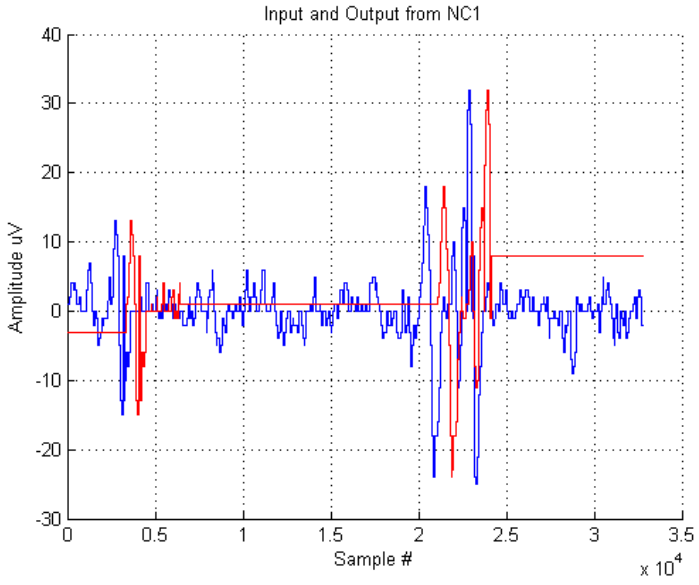


Figure 4.11: Test capture of recorded neural signal input and output from NC1.

The testing of NC1 included an FPGA for data input and output. The Xilinx ChipScope tool was used for data retrieval and Matlab was used to compare with the simulated results. The post-layout simulations are fixed to specific voltages like 1.08 V, 1.2 V, 1.32 V, which is controlled by the process corners available from the technology library. Therefore, only those points can be compared with simulated estimates. The observed power dissipation of the chip core at different voltage levels is shown in Figure 4.12, as well as simulated results for comparison. Each power measurement is calculated by taking the average of 100 current readings and multiplying it by the measured core vdd.

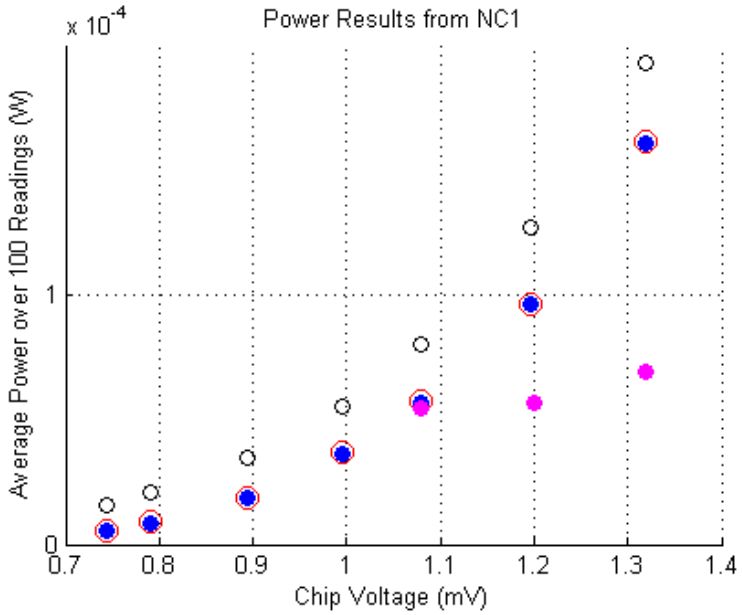


Figure 4.12: Real-time averaged power results captured from NC1. 16 Channels (black circles). Test channel (blue dots). Static power (red circles). Post-layout simulations (magenta filled circles).

Three readings for every core voltage are shown. The first is with power supplied, but with no circuit activity (red). All inputs are grounded and all outputs are stable. This corresponds to a measure of static power. The second readings are for total power on the test channel (blue dots) and the third readings are for the 16 channels (black circles). The test channel results are slightly larger than the static results, which is not clear from the graph in Figure 4.12. The simulated measurement for 1.08 V is close to the actual static power measured, but the others are far away. The power measurements at a core vdd of 744 mV are included to show the trend, but the output contains errors, so the lowest useful tested voltage is considered to be 0.8 V. This gives a power dissipation (test channel) of 8.9 μW at an input data rate of 240,000 bits/s. The resulting $\frac{E_c}{bit} = 37$ pJ per input bit. However, this includes the large static power (no clock or data) of the entire chip and if we consider the 16 channels at 21 μW at an input data rate of 3,840,000 bits/s then the resulting $\frac{E_c}{bit} = 5.5$ pJ per input bit. Therefore as long as the power of transmission is greater than 6 pJ per bit then the spike detection algorithm is saving system power. If we consider a 3.0 V power amplifier from an ultra low-power wireless system on-chip [96] at the lowest setting 6.8 mA (-18 dBm RF output power) at the maximum data rate of 2 Mbps then each bit costs 1.02 nJ. Therefore, on a per bit basis, the spike detection costs the system 0.6 % of the energy spent to send a bit wirelessly.

There is an important discrepancy between the simulated power estimates and the observed power of NC1. The post-layout simulations break power estimates into leakage, in-

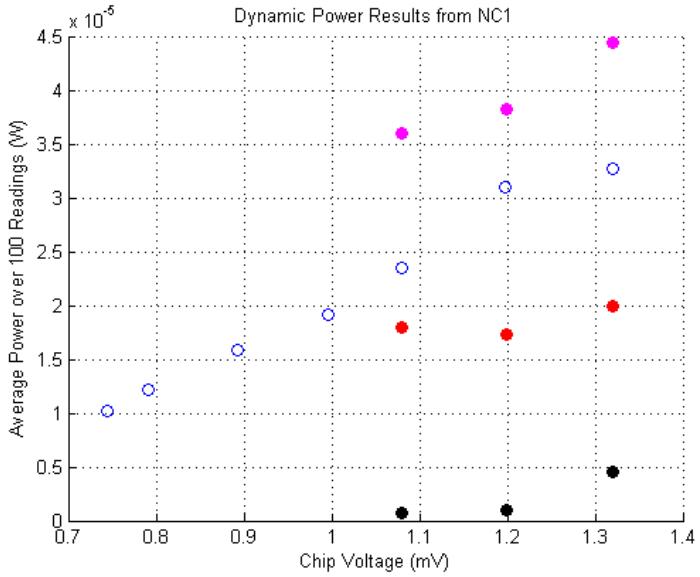


Figure 4.13: Averaged dynamic power results captured from NC1. 16 Channels (blue circles). Post-layout simulation internal power (magenta filled circles). Post-layout simulation switching power (red filled circles). Post-layout simulation leakage power (black filled circles).

ternal, and switching power. The leakage power of the core does not exceed 1 μW whereas the measured static power is very large in comparison. If we compare the separate power estimates to the measured dynamic activity we get the plot in Figure 4.13. Once again, only the core voltages of 1.08 V, 1.2 V, and 1.32 V are comparable. This could mean that either the power estimation is not very useful or that NC1 has flaws that dramatically increases the quiescent current draw. The potential flaw is discussed in the next subsection.

4.3.1 SPI Design Error

The SPI controller in the Encounter synthesized netlist has a design bug, which was unfortunately not caught until after both designs (NC1 and NC2) were fabricated. The bug was not caught because the Verilog code from the Synopsys output was simulated instead of the Encounter output, while simulating the SPI functionality. The bug prevents any register from being programmed after the reset is toggled. A block digram of the circuit architecture pertaining to a single bit of the received data register is shown in Figure 4.14. The simulated signals are shown in Figure 4.15. The low-power optimization step within the Encounter script, paired with the asynchronous resets of modules within the channels, is the suspected cause of the design bug as the script does not correctly protect the SPI state machine from optimizations that would render other logic useless.

On a reset high n71 is set high, which in turn forces n103 to be stuck at 0. This creates a permanent stuck-at fault since n47 is unable to change n103. The exact reason for this circuit configuration is unknown. The module OAI22X implements two 2-input OR gates into a 2-input NAND gate and the DFFRX module is a flip-flop with reset. The state register signal, shown in Figure 4.15, has a high impedance value that prevents it from entering the final state of the SPI state machine (11) where the transmitted data is copied from the temporary received data register to the output of the slave controller. This state transition into the final state usually occurs when the bit counter (counts input bits from transmitted data) is greater than the width of the received data register, but the bit counter

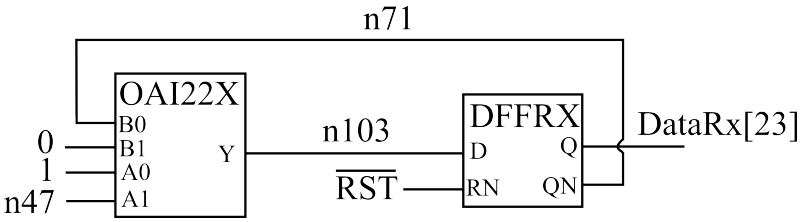


Figure 4.14: Block diagram of SPI circuit that prevents the received data register (only one bit shown) from being written after Reset goes high.

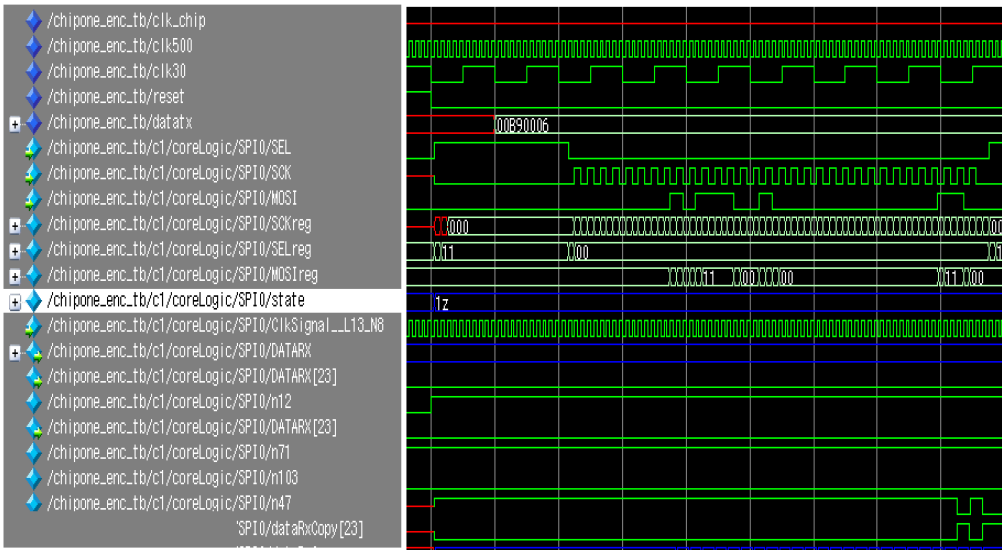


Figure 4.15: Post-layout simulation of SPI components for a single bit of the received data register.

is no longer present in the post-layout code. The exact same VHDL code was used in a FPGA and the bug was not seen. Therefore, in the low-power optimization commands of the script-generated layout a bug was introduced. Removing the optimization removes the bug.

4.4 Discussion and Contributions

A neural spike method was presented that reduces the number of false detections in experimental DRG recordings. The compressive gains of spike detection depend on the number of neural spikes present. By capturing only spike detections a system can provide a compression of over 90%. The spike detector implemented in NC1 has a energy cost of 6 pJ per input bit with 30 kS/s on 16 channels. The resulting power dissipation for the spike detector is approximately 21 μW or 2.6 mW/cm^2. If we constrain a fictional 16-channel system consisting of only the neural spike detector and wireless radio to 1 mW and assume an overall compression ratio of only 90%, we could process 9.4 times more input bits than an uncompressed system. The spike detection system, even with a large static power dissipation, saves the overall system energy compared to a system with no compression at all.

The noise envelope dual-threshold spike detection has a power consumption per bit that is similar to the energy consumption of other systems. However the power per area is lower than any of the reported full systems. Future work to reduce the power dissipation at a circuit level could improve the results. The implementation of NC1 with a standard input/output library may have limited the voltage scaling of the core. The core may work at voltages below 0.8 V, but due to the noise margins of the level-shifted output drivers the signal value would be wrong. A direction for future work in the implementation is to remove the relatively large static power dissipation, which does not match the simulated estimates. A simulation of the flawed post-layout netlist shows a large number of memory elements undefined or unknown due to the SPI bug. We believe that these lines are actively switching or toggling causing a large false static power dissipation that is effectively, no longer static, but most likely demonstrating an activity factor of 0.2, at least for the 1.08 V simulation, which matches closely to that estimate at that activity factor. These undefined signals may also be causing crowbar current caused by nodes floating at mid-voltage levels, because they are undefined.

There is a large potential for future work on the system due to its low frequency of operation and modularity. Modules can be power-gated. Power gating modules would reduce the static power dissipation. Only the threshold comparators need to be continuously running. All other modules are only needed on known intervals or threshold crossings. There are few memory elements that must be maintained, so the overall system power can be very low. This design change would make the spike detection power dissipation almost completely based on neural activity.

The second interesting feature needed for future systems is the ability to automatically determine bad channels and thus turn off the recording circuits and/or put them to good use on another channel. This leads to a 'processor cache'. A set of spike detectors are available for an even larger number of neural channels. The channels are tested and the spike detection circuits are applied to only electrodes that have useful neural signals. This could really improve the scalability of the system.

4.4.1 Comparison to the State-of-the-Art

The implementation of the dual-threshold spike detection gives us a comparison value for energy per input bit of 6 pJ/bit at a signal-dependent compression ratio that is typically close to 98%. It is difficult to compare only the spike detection algorithm since other implementations include other aspects of their system design in different technology nodes at

different voltage levels. However, the reduction in false detections is important in reducing system power dissipation and in [97] the noise-envelope dual-threshold spike detection method, presented in this chapter, is shown to reduce the number of false detections by at least 50% and increasing the number of true positives compared to the absolute threshold method.

In [8] a compressed sensing based neural recording implementation is presented. The authors use a signal-dependent model-based recovery dictionary to improve upon model-based compressed sensing recovery. The on-chip circuitry for compressed sampling is simple and low-power. It is capable of encoding both neural spikes and the noise in-between. The authors use a programmable (external) absolute threshold for neural spike detection to choose between two different recovery dictionaries. One dictionary is designed to reconstruct spike detections, which requires a training phase. The other dictionary is intended to reconstruct the noisy samples. The absolute threshold is shown in [97] to have 50% more false detections than the method presented in this chapter. In addition, the thresholds are externally programmed and are therefore not adaptable like the noise-envelope based thresholds. However, the power dissipation of the compressed sampling calculation is very small at 0.27 μW at 0.6 V per channel (continuous) whereas the dual-threshold spike detector requires 1.3 μW per channel. The area requirements for the two methods are similar with the compressed sampling requiring 200x300 μm versus the 200x200 μm required by a single channel dual-threshold spike detector. Overall, the signal dependent compressed sampling approach has a strong advantage in power dissipation, but has poorer spike detection performance due to the selected spike detection method, which may, depending on the number of false detections, cost more energy than the small increase seen in the dual-threshold spike detector from this chapter.

In [98] a 24-channel neural recording system is presented. The authors use a spike detection that can operate in either single-threshold or dual-threshold modes. The thresholds are set externally. The single threshold method suffers the same false detection rates as the absolute threshold method, but the dual-threshold, if programmed correctly, will match the performance of the noise-envelope based threshold presented in this thesis. The thresholds are implemented in an analog circuit, but the decision to actually transmit a spike is made in a microcontroller. The microcontroller is external to the analog ASIC and has large power consumption compared to this work. The lowest measured current from the microcontroller is 13 mA. The lowest voltage available to the microcontroller is 1.8 V so therefore the power budget for their activity-aware signal capturing is an order of magnitude larger than the spike detector presented in this thesis. The power savings come from the dedicated circuitry for spike detection versus a generic microcontroller architecture.

In [99] a NEO-based spike detector method is presented with simulated results. The authors introduce an analog circuit that calculates the NEO output as well as a threshold that adapts to the signal conditions. The simulation results indicate a low rate of false detections around 1%, however the detection of actual spikes suffers with 30% missed detections on a 'easy' neural recording. The estimated power dissipation is 1.5 μW at 1.8 V with a logic area of 0.03 mm^2. The spike detector presented in this chapter has similar true detection rates, with a slightly larger false detection rate, but dissipates less power at 1.3 μW in a larger area (0.04 mm^2). The choice of calculating the NEO with an analog implementation is what makes the NEO-based threshold low-power since using the large values generated by the NEO operator digitally would increase the power dissipation considerably. The utilization of analog circuitry to represent a signal with a large range of values is key to the success of the NEO-based spike detector. Applying an analog-based approach to a threshold comparison could potentially provide a lower power implementation, however the need to capture samples before the initial threshold-crossing would still be necessary. In addition, the implementation of the noise envelope is best suited for a state-machine in the digital domain. Therefore the dual-threshold window-based noise envelope spike detector,

as presented in this chapter, is best suited for digital implementations.

Chapter 5

Wavelet-Based Spike Compression: Simulations, Implementations and Results

This chapter presents the simulation and implementation of the multilevel DWT compression module for neural spike compression. The compression module reduces the number of bits being sent per spike detection and also provides an alternative method of encoding a continuous recording. The first section presents the formulation and simulation results of the discrete wavelet compression method. The second section covers the implementation and testing results of a 16-channel spike detector with multilevel wavelet compression. The chapter concludes with a discussion on the results and future work including comparisons to the state-of-the-art.

5.1 Multilevel DWT Compression

In Chapter 4 the power cost of neural spike detection was shown to be considerably less than the cost of wireless transmission. Therefore adding more processing to reduce the transmission bandwidth may yield further power reductions in the system. The DWT was identified in Chapter 3 as a strong candidate for neural signal compression. By using the symlet-4 basis [28] a single-level of DWT can capture neural spike shapes with a MSE less than 100 using only 8 coefficients, which yields a signal that is 84% smaller and has only a small loss in accuracy in subsequent spike classification. A study of the multilevel DWT offers the ability to add or remove additional processing depending on the compression needs. The block diagram of the tree structure for DWT decomposition from Chapter 3 is replicated in Figure 5.1 [67]. The g block is the detailed filter and the h block is the approximate filter. The down arrow with a two represents decimation by a factor of 2. In a stationary wavelet transform this decimation step does not occur [100]. As the symlet-4 basis is well-suited for neural signals, the approximate coefficients contain most of the signal energy. Therefore adding levels, which reduces the number of approximate coefficients, may help reduce the overall number of coefficients needed for accurate representation of the neural signal.

An alternative to the tree structure for multilevel wavelet decomposition is the lifting architecture [101]. In the work by [37] the lifting architecture is identified as a more efficient structure for producing both approximate and detailed coefficients. However, a tree structure that utilizes FIR filters is used here, and is not folded [102]. This is done to enable an easy way of not performing the detailed filter decompositions since it may be the case that

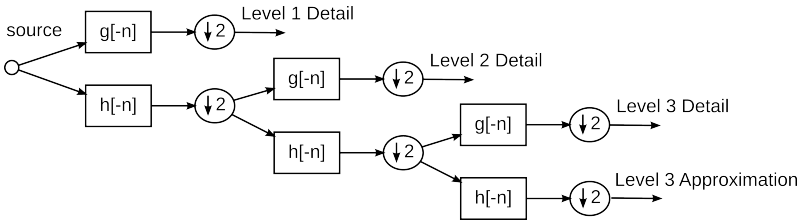


Figure 5.1: Multi-level DWT encoder using the tree algorithm

detail filter coefficients are not desirable in the compressed representation. In future work the lifting architecture may be explored [101].

5.1.1 Simulations with Coefficient Selection

A multilevel discrete wavelet transform demonstrates good MSE performance with few coefficients and can allow the user to select the number of levels to change the compression gains. All of the wavelet coefficients for each spike detection in a neural recording can be found in a Matlab simulation with zero-padded discrete wavelet decomposition. These coefficients are sorted for each spike and the K largest coefficients are selected to represent the neural spike. The trade-off between the number of coefficients used with respect to the average MSE performance is summarized in Figure 5.2 and Table 5.1. By selecting a certain number of coefficients a non-linear threshold is implemented that is unknown until the coefficients have been calculated. In Figure 5.2 and Table 5.1 the second-level has the best performance. The first level has the worst; this is generally true for most of the feline DRG recordings during spike detection. The data in Table 5.1 is from a simulation over 192 channels with a 95% confidence interval for the plus/minus.

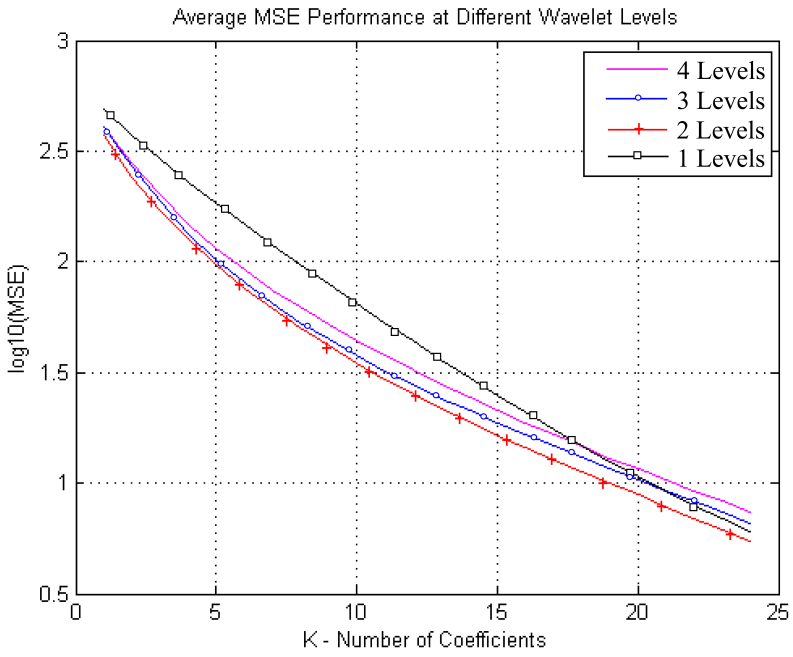


Figure 5.2: MSE performance at different levels with the number of coefficients selected.

Ideally the implementation would sort all the coefficients and select the best ones, but this creates two types of overhead costs to the system. The first overhead cost is the

Table 5.1: MSE performance at different wavelet levels with the number of coefficients selected over 192 channels.

K	5	10	15	20	25
Level 1	519.35 ± 46.1	131.99 ± 8.3	45.41 ± 2.1	15.40 ± 2.1	6.13 ± 0.1
Level 2	246.31 ± 21.5	62.18 ± 4.0	21.59 ± 1.1	9.75 ± 0.4	4.89 ± 0.2
Level 3	250.53 ± 22.5	55.67 ± 4.3	21.54 ± 1.2	10.50 ± 0.5	5.49± 0.2
Level 4	257.13 ± 25.8	63.69 ± 4.4	26.52 ± 1.5	13.22 ± 0.6	7.09 ± 0.3

operation of sorting the incoming samples, but the most important overhead is that each coefficient must be encoded with its location within the frame. This additional location information doubles the size of the compressed signal. Therefore the coefficient locations should be determined beforehand so that only the coefficient amplitudes need be sent. This means the best coefficient locations on average need to be determined and a penalty in MSE should be expected. For the same channel shown in Figure 5.2 the following 24 coefficient locations were found in a 48-sample sized window.

Level 1: [5,4,8,6,9,10,7,11,12,3,2,13,14,24,25,15,22,23,21,20,19,16,18,17].
All coefficients are approximate.
Level 2: [4,6,5,7,3,8,**22**,9,13,14,12,11,10,**20,21,23**,2,**24,25,31,26,27,30,28**].
Level 2 approximate coefficients and **level 2 detailed coefficients**.
Level 3: [5,4,8,6,9,10,7,11,12,3,2,**13,14,24**,<u>25</u>,**15,22,23,21,20,19,16,18,17**].
Level 3 approximate coefficients and **level 3 detailed coefficients**
and <u>level 2 detailed coefficients</u>.

The approximate locations are shown in regular weight font, the bold font locations come from the last level and the underlined locations are from the detailed filter in level 2 with the 3-level decomposition. The detailed coefficients from level 1 are never used, so that FIR filter could potentially be omitted. The level 2 and level 3 coefficients are mostly from the last level approximation and detail levels. The MSE performance from taking the average coefficient locations is shown and compared to the ideal performance in Figure 5.3 for a single channel. The average MSE performance from 192 channels is shown in 5.2. The results are generated from simulating the channels with the same pre-selected coefficients, which should give a worst case scenario. The pre-selected coefficients could be tailored to each channel to improve performance.

Table 5.2: MSE performance at different wavelet levels with the number of pre-selected coefficients over 192 channels.

K	5	10	15	20	25
L 1	1542.82 ± 182.1	558.15 ± 66.9	232.23 ± 24.7	112.61 ± 13.2	16.34 ± 1.5
L 2	765.71 ± 89.5	219.54 ± 27.5	100.81 ± 19.3	31.52 ± 19.3	21.28 ± 1.8
L 3	855.22 ± 100.7	755.75 ± 94.2	754.02 ± 94.2	725.19 ± 92.1	176.12 ± 22.7

The penalty when using the pre-selected coefficients for the level 2 decomposition is equivalent to using 5 more coefficients. A 1.8 ms frame has 48 samples and 5 coefficients is 10% of the entire frame. In addition, the third level reconstruction, when using pre-selected coefficients, is worse than the first and second level reconstruction. The third level is therefore not needed during spike detection. The selection of coefficients in the compression system is concerned with maximizing the fidelity of the reconstructed neural spikes, but how does it perform on the entire signal that contains mostly noise? The entire

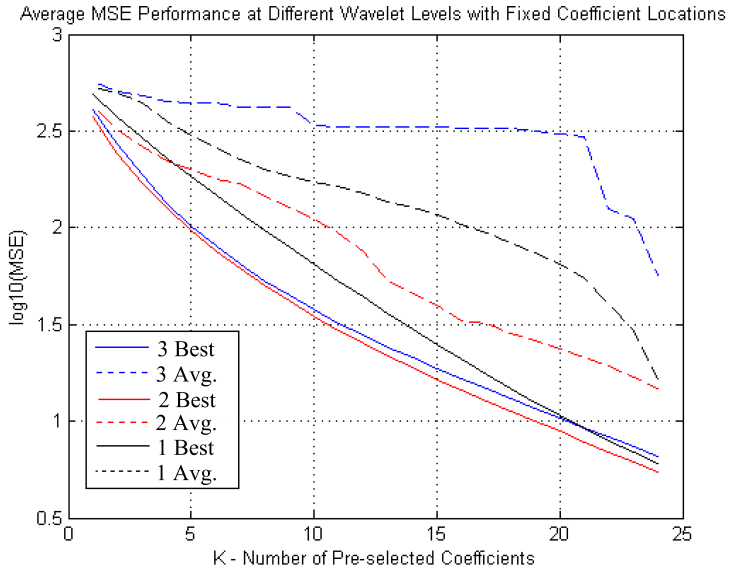


Figure 5.3: MSE performance at different levels with fixed pre-selected coefficients.

neural recording is compressed in consecutive blocks of 48 sample windows and the mean MSE for the three levels of wavelet decomposition is presented in Figure 5.4.

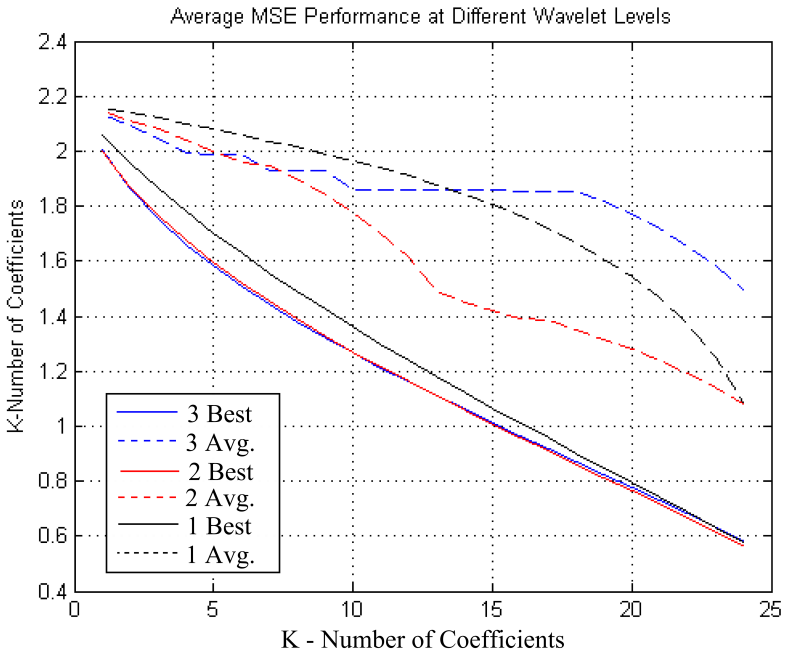


Figure 5.4: MSE performance on a continuous neural recording from a feline DRG.

The third level has a small MSE performance advantage when encoding with 5 coefficients or less, which provides a 90% reduction in data transmission. If the frame size is increased to 100 samples and the coefficient locations are transmitted the third level outputs provide superior reconstruction with respect to average MSE. The fourth level of wavelet reconstructions did not provide any reconstruction advantages. The average MSE over 96

channels is 165 and 175 for level 3 and level 2 reconstructions, respectively. The standard deviation is 256. Therefore, all three levels could be implemented to give an option for improved MSE reconstruction performance during continuous recordings. On the 5 sets of artificial data the average MSE is 51 and 46 with a standard deviation of 9 for the second and third level reconstructions, respectively.

5.2 Implementation of Multilevel DWT

This section presents the implementation of a 16-channel test chip called NC2. NC2 has 16 spike detection modules and 8 multilevel DWT modules that use six 8-tap FIR filters. The system architecture for NC2 is shown in Figure 5.5. NC2 uses the same spike detection modules and control scheme as NC1, but instead of a multiplexed output all channels are converted to serial outputs (P2S). There was only enough room to implement eight DWT modules for the system, so only the first 8 channels have the ability to further compress the output. This scheme could be replaced in the future with a single DWT module shared across all 16 channels. Additional control blocks are used to provide the coefficients and coefficient locations for the DWT modules.

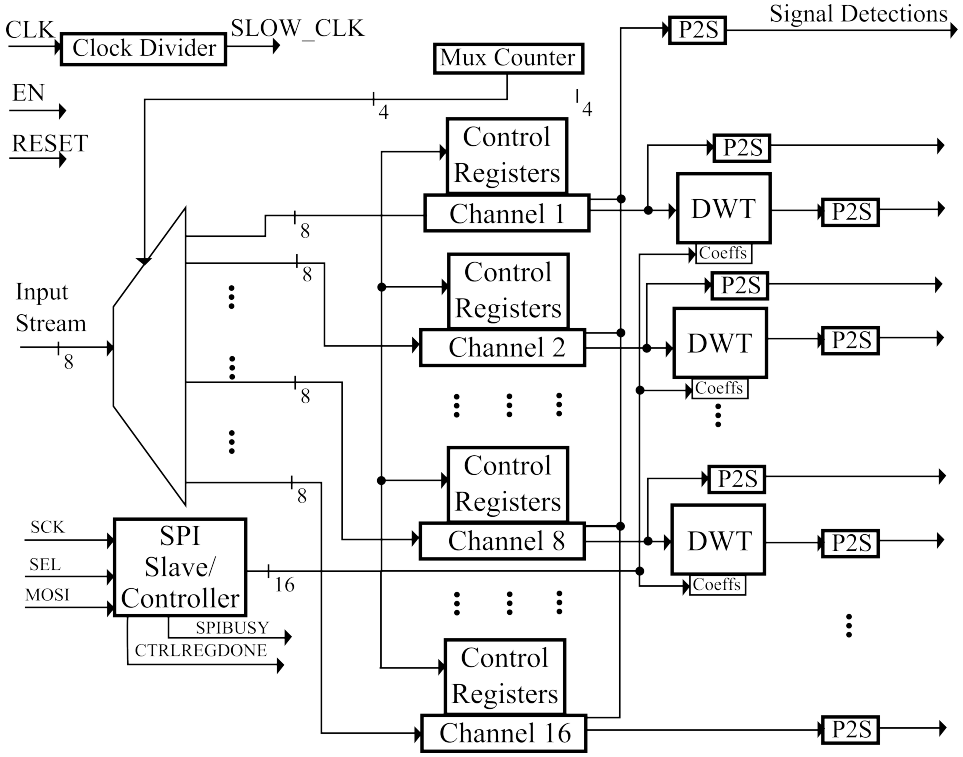


Figure 5.5: NC2 system overview.

A block diagram of a single FIR filter for the multilevel DWT is shown in Figure 5.6. It consists of clocked memory elements, combinational adders and combinational multipliers. Fast operational speed is not a major concern in the wavelet decomposition so the data path is controlled using a gate with memory which aligns the data so that the correct samples are added and multiplied together. This means that a decimator is no longer needed on the output. In addition, the multipliers' output is double the bit-width of the input data stream, so the data must be sliced at the output to get back to 8 bits. The 7 most significant bits (MSBs) and the sign bit are selected for the final output. The sign-

magnitude hexadecimal representation of the approximation and detail filter coefficients are listed below in Table 5.3 along with their fixed point values and ideal values from Matlab.

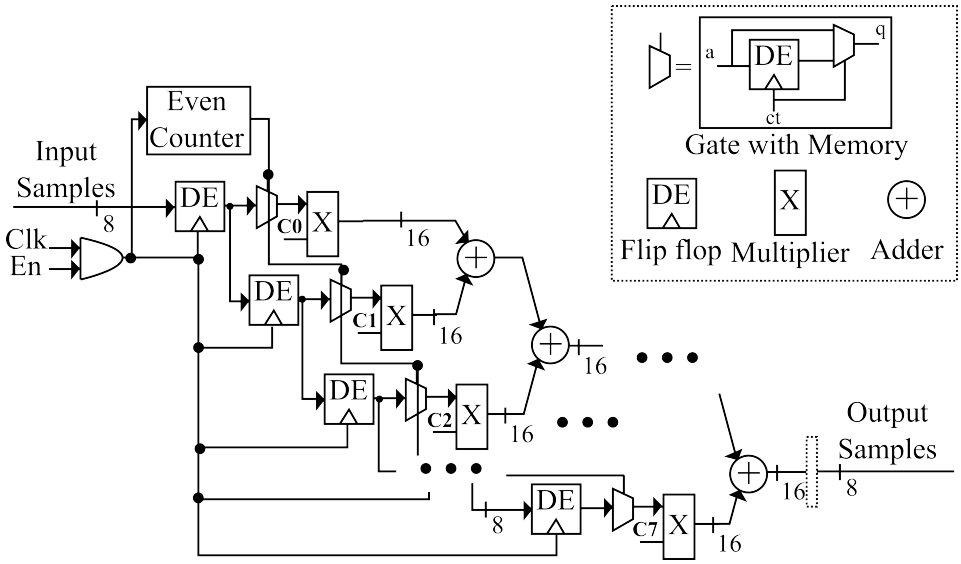


Figure 5.6: Block diagram of the 8-tap FIR filter-based design for implementing a single decomposition level of the symlet-4 DWT.

Table 5.3: Coefficient values for Symlet-4 wavelet decomposition

Coefficient	Hex	Fixed Point	Ideal
A0	89	-0.0703	-0.0757
A1	83	-0.0234	-0.0296
A2	3F	0.4921	0.4976
A3	66	0.7968	0.8037
A4	26	0.2968	0.2978
A5	8C	-0.0937	-0.0992
A6	81	-0.0078	-0.0126
A7	04	0.0312	0.0322
D0	84	-0.0312	-0.0322
D1	81	-0.0078	-0.0126
D2	0C	0.093	0.0992
D3	26	0.2968	0.2978
D4	E6	-0.7968	-0.8037
D5	3F	0.4921	0.4976
D6	03	0.0234	0.0296
D7	89	-0.0703	-0.0757

The fixed point representation for the coefficients has almost no effect on the MSE, as shown on a single channel in Figure 5.7. The architecture presented here can be improved upon by either using an integer-lifting approach [101] or B-spline factorization scheme [103]. From [37] the authors indicate that the integer lifting architecture is best when optimizing for area and power instead of throughput. In addition, alternative compositions of the FIR filter could be used to reduce the very large scale integration (VLSI) implementation area as seen in the work of [104].

A 3-level DWT is shown in Figure 5.8. All detail and approximation filters are included in case the coefficients are needed and to give an upper limit on the power dissipation of the

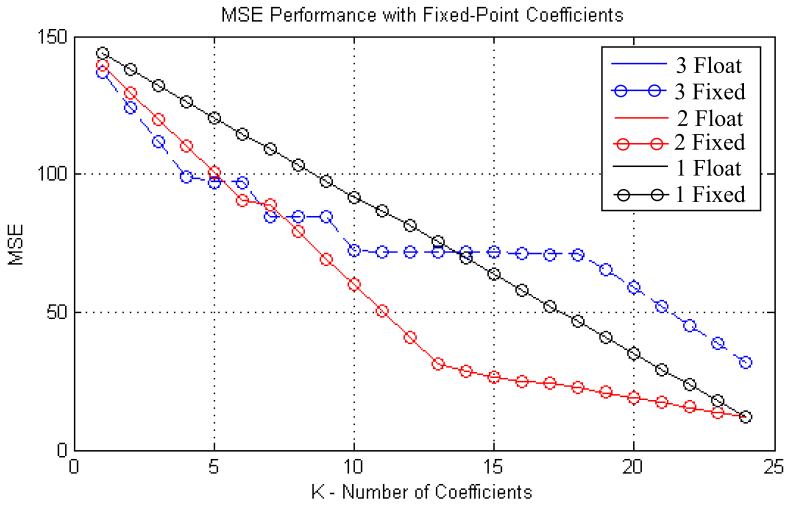


Figure 5.7: MSE performance with fixed-point coefficients and average locations.

multilevel DWT. Two sets of coefficients are needed for the detail and approximation filters. All detail and approximation filters use the same coefficients, respectively. The outputs of each filter are stored in a first-in first-out (FIFO) memory or passed on to the next level. Each sample is written to an address specified by the appropriate counter. The counters are latched on divided down clocks since each level has a reduced by two sample rate. The FIFO output is selected by the state machine and the read address is shared among all the memories. This may create an extra power dissipation as only one FIFO output is selected at a time for output. The level 1 coefficients are read first and the level 3 coefficients are read last. The data read speed is assumed to be faster than the write speed and in this system the speeds are defined by the system input clock, which is 16 times faster than the channel clock.

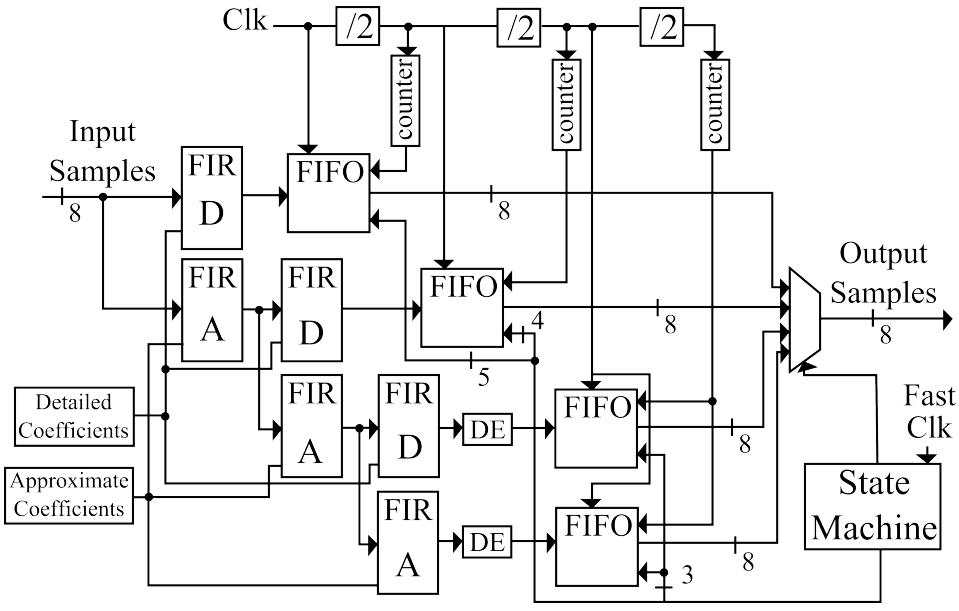


Figure 5.8: Block diagram of a 3-level DWT.

5.3 NC2 Simulations and Test Results

The logical equivalence between the synthesized designs and Matlab simulations is done using netlists from Synopsys Design Compiler and Cadence First Encounter for post-layout confirmation. The design is specifed in VHDL and fabricated using the IBM 130-nm design kit on a 3x3 mm chip. The design flow stages are the same as described for NC1 in Chapter 4. The VHDL test benches use an ideal clock frequency of 500 kHz, which supplies an input data rate of 4 Mbps from portions of recordings from the feline DRG experiments. The Synopsys Design Compiler synthesized results for the FIR filter and the 3-level DWT module are shown in Table 5.4. A clock with a period of 2000 ns is used for the DWT module fast clock and a 32,000 ns clock period is used for the main clock. The FIR filter has a similar power estimate to that of the threshold detection module, but is larger in area. The 3-level DWT module should dominate the area and power dissipation of the new system.

Table 5.4: Power and area estimates for the DWT modules

Metric	FIR Filter	3 Level DWT
Area (μm^2)	35,739	250,974
Timing (slack ns)	31998	1995/31998
Dynamic Power (μW)	0.102	2.3
Leakage Power (nW)	46.9	335

The synthesized power and area results for NC2 are in Table 5.5. The dynamic power prediction of Cadence First Encounter is larger than the Synopsys prediction, but the leakage power is smaller.

Table 5.5: Power and area estimates for NC2

Metric	Synopsys Design Compiler	Cadence First Encounter
Voltage	1.2 V	1.08 V
Area (μm^2)	2,948,343	2,866,667
Timing (slack ns)	1953	1997
Dynamic Power (μW)	42.4	451.8
Leakage Power (μW)	3.86	3.2

As expected, the DWT modules utilize most of the chip area as shown in Figure 5.9. The addition of 8 additional DWT modules is most likely prohibitive due to the area required by the logic, so therefore future iterations should have fewer DWT modules operating at a faster frequency. The fabricated chip and layout from Cadence are shown in Figure 5.10. The amount of top metal routing and congestion is greater over the DWT modules. Logical equivalence was found through all stages of the NC2 design and yet the SPI bug was still present after optimization. This means that the default mode is the only operational mode, which causes some problems. The wavelet decomposition is operating in its continuous mode, which means that it expects consecutive windows of 46 samples and is producing all of the 64 coefficients. However it is receiving only spike detections with windows of 48 samples. The samples between spikes are always the same (latched outputs) so the dynamic power should be slightly higher than the expected value for spike detection mode. The window mismatch causes a discrepancy, which negatively effects the expected reconstruction. Snippets of experimental input into the wavelet module and the resulting output of NC2 are shown in Figure 5.11; with the plots (manually) overlaid on top of each other. Given all of the wavelet coefficients the reconstructions should be near perfect, but the left plot in Figure 5.11 has an MSE of 6 and the right plot has an MSE of 35.

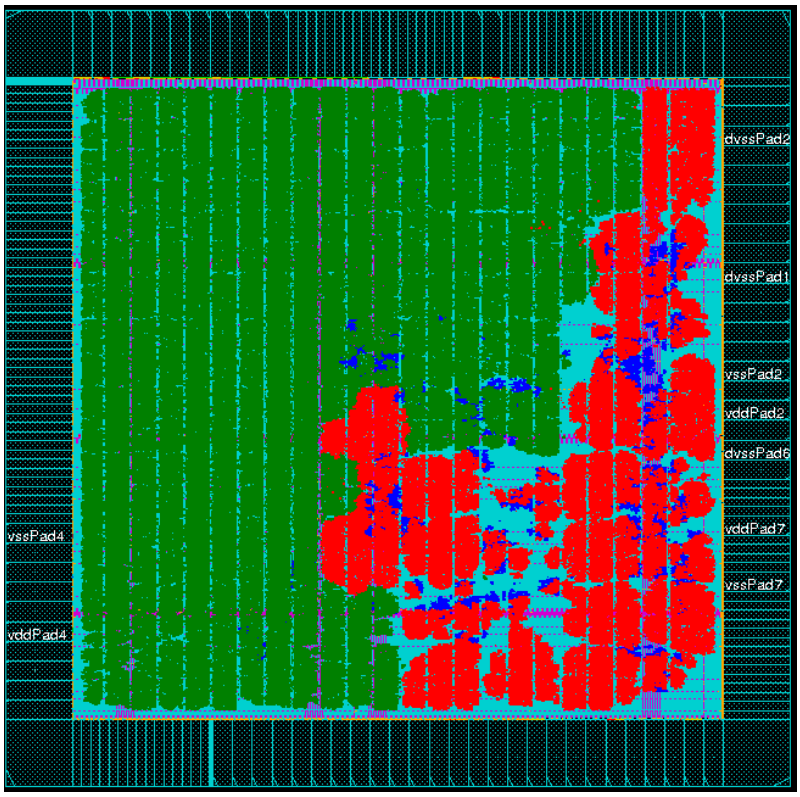


Figure 5.9: Placement of NC2 modules. Red = spike detection. Green = DWT. Blue = control circuitry.

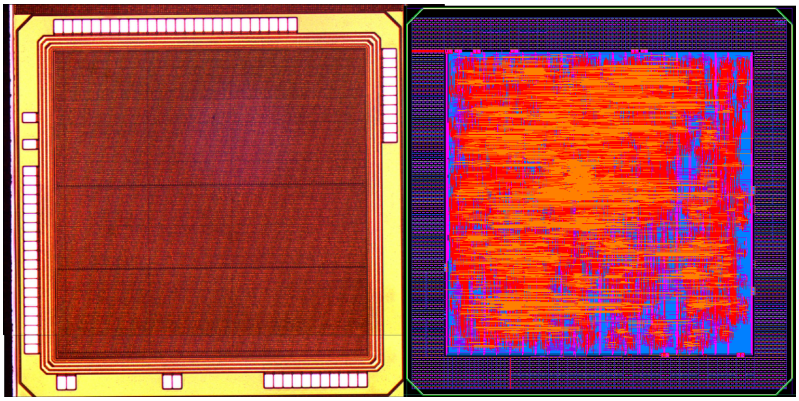


Figure 5.10: NC2 chip and NC2 layout from Cadence Virtuoso

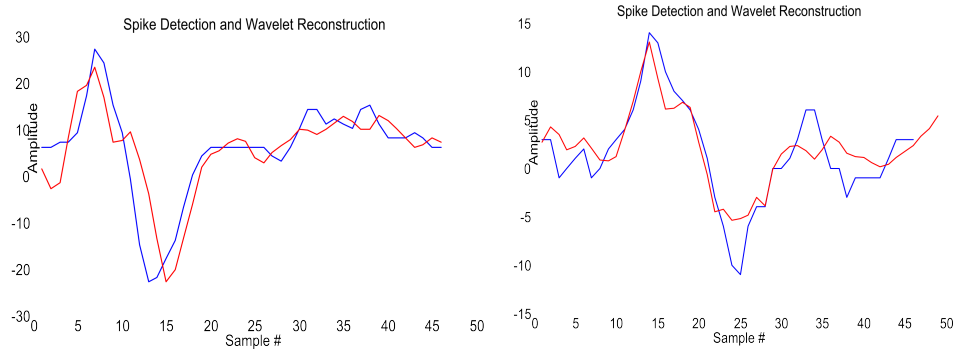


Figure 5.11: Plots of captured signal from the testing of NC2. Left plot has a MSE of 6. Right plot has a MSE of 35. Red is reconstructed. Blue is original.

The test setup for NC2 is the same as NC1, except the outputs are serial and not multiplexed and there is no test channel. The measured power results are shown in Figure 5.12. Each power measurement is calculated by taking the average of 100 current readings and multiplying it by the measured core vdd. Three different measurements are taken at each voltage level. The small blue circles are with power supplied, but all inputs and outputs are stable. This should correspond to static power. The red circles are measured when data is being sent only to a single channel and all other channels are zero. The black circles are measured when the data is sent to all 16 channels. The post-layout simulated results are for all 16 channels and are higher than the measured results. The simulation results are fixed to specific voltages of 1.08 V, 1.2 V, and 1.32 V. It appears the power estimate poorly modelled the activity factor in the DWT modules since the spike detection module power estimations are closer in Chapter 4.

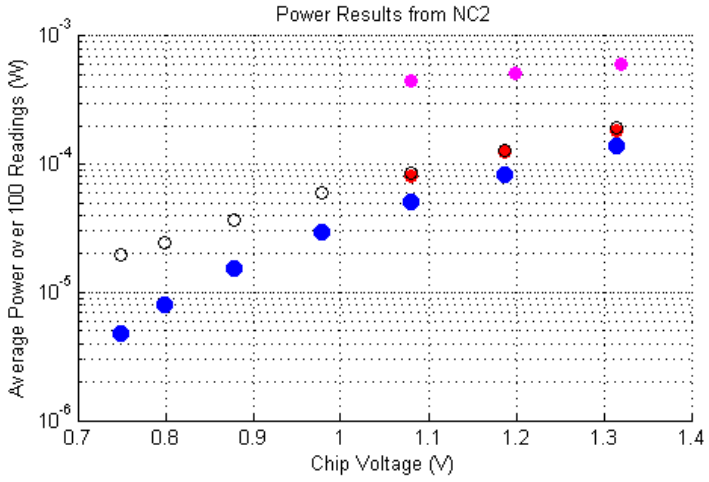


Figure 5.12: Real-time averaged power results captured from NC2. 16 Channels (black circles). 1 Channel (red circles). Static power (blue circles). Post-layout simulations (magenta filled circles).

In continuous mode the measured values should actually be higher than expected since the wavelet module would be constantly outputting values, but the spike detection outputs are constant most of the time, so there is actually relatively little signal activity. The measurement at 749 mV contains errors, but is included to complete the declining trend. Therefore the lowest useful tested voltage is 0.8 V. This gives a tested power dissipation of

24.1 μW at an input data rate of 3,840,000 bits/s. The resulting energy per bit is $\frac{E_c}{bit} = 6.3$ pJ per input bit. This estimate should be an upper-limit to a properly functioning module operating on spikes only, not the continuous mode. If we compare this to the spike detection alone, then the addition of eight 3-level DWT modules resulted in an increase of only 1 pJ per input bit and therefore the system is still saving energy per bit compared to wireless transmission. However, this very small increase is cause for concern since a large amount of logic has been introduced and the small increase in energy consumption is counter-intuitive.

The small increase might be explainable through the same flaw that prevents the SPI from being programmed. The addition of the DWT modules did not increase the static power. In fact the measured 'static power' in NC2 is actually lower than NC1 by 6.5 μW at 1.08 V. This could mean that the static power dissipation observed in NC1 is affected by the introduction of the DWT modules. However, only 8 channels would share this effect. The mechanism of this effect is also not clear. It is far more likely that the removal of the test channel and its control circuitry is responsible for the reduction. NC1 at 1.08 V has a static power measurement of 56.85 μW. If we evenly divide that over the 17 channels the static power contribution of each channel is 3.3 μW. The static power measurement of NC2 at 1.08 V is 50.35 μW, which is close to the 52.8 μW predicted by using only 16 channels in NC2 (3.3 μW/channel). If a fluctuation of 0.16 μW/channel of the false static power is acceptable between different chips on different boards at different times then the source of the large false static power is most likely in the floating nets of the spike detection units as presented in Chapter 4.

If we compare the dynamic power dissipation between NC1 and NC2 we hope to see a small increase. The activity of the wavelet modules should be modulated by the spiking activity. The wavelet modules should not have lots of bit flips while processing constant data. At 1.08 V NC1's total current draw with 16 active channels is 74.3357 μA with a dynamic current increase of 21.6963 μA. The dynamic current increase in NC2 for 16 active channels is 33.0836 μA. If we assume the spike detection modules are responsible for the same current in NC1 and NC2 then the difference is 12.1 μA over 8 wavelet modules. If we average the current per channel and wavelet module, the spike detection module uses 1.35 μA while the wavelet module uses 1.51 μA. This is an increase of only 15%, despite being 575% larger in area. This effect is what we hoped to prove more concisely with NC2; large processing modules after spike detection have a small effect on the overall power dissipation of the system because the activity factors are being modulated by the infrequent spiking activity. If the wavelet module were to be properly inactive the current increase would be even smaller.

5.4 Discussion and Contributions

The goal of the implementation of a multilevel DWT module for neural signal compression was to demonstrate feasibility and to permit experimental power measurements. The power dissipation of the spike detection and the wavelet modules is considerably smaller than the expected power dissipation of a wireless transmitter with an energy per bit of 1 nJ/bit. However, the area required for implementing the 3-level wavelet modules is larger than the spike detection modules. If area is a constraint in future designs then sharing one wavelet module for all the channels is advisable, but likely with the expectation of increasing the power dissipation. An important finding we were expecting from the test chip is that any processing that happens after spike detection does not have a relatively large effect on the power dissipation. Unfortunately, due to a bug in the programmability of the test chip, this could not be exactly confirmed. However, we demonstrated that the 3-level DWT module only required 15% more current than spike detection modules despite being 575% larger in logic area. The post-layout results of the spike detection and wavelet modules were used

in a publication [97] that describes the spike detection algorithm as well as the nonlinear wavelet threshold. The post-layout results serve as an upper limit to the potential of the wavelet module operating in spike detection mode. The NC2 test results indicate that the large static power is sourced in the spike detection modules and this must be addressed in future work. These modules demonstrate relatively large static power consumption, which hopefully can be reduced in future iterations. By removing the low-power optimization step on the SPI slave the design flaw should no longer be present.

The fabricated NC2 test chip is capable of producing full wavelet decompositions, which does not provide any compression capability. However, if the simulations are any indication of intended performance, the compression per spike could be 10 samples for every 48 samples received, without degrading the accuracy of the spike classifications. The power cost of adding this extra level of compression is measured and estimated to be an extra 1 pJ/ input bit for a total of 7 pJ/bit. Given that wireless transmission is estimated at 1 nJ/bit, then the potential energy savings are huge. For every 1000 bits compressed the transmitter could send only 7 bits wirelessly using the same power. The measurement of a continuous operating wavelet for raw signal encoding is needed for future work.

5.4.1 Comparison to the State-of-the-Art

In [105] a neural spike recording channel implementation is presented. The authors use a piece-wise linear approximation of the spike shape as features for spike compression and classification. The classification accuracy is evaluated with an artificial signal and the accuracy is around 90% at a SNR of 6 dB. This is equivalent or greater than the SNR of the neural recordings from the DRG since 6 dB means the spike has an amplitude twice as large as the noise. The digital portion of their design does not include spike detection, but only feature extraction. It requires 200 nW for a single channel and provides compression of up to 56 bits per detected spike, while the proposed feature compression from this chapter would require 80 bits per spike detection ($K = 10$) and is measured to require 1.51 μA at 0.8 V for 1.2 μW (dynamic power). Therefore the presented DWT compression method outperforms with respect to classification accuracy and reconstruction shape, but requires more power per channel to provide a smaller amount of compression per spike. The wavelet approach described in this chapter also allows the compression of continuous data, but this could not be measured in terms of power.

In [8] a compressed sensing based neural recording implementation is presented. The authors use a signal-dependent model-based recovery dictionary to improve upon model-based compressed sensing recovery. The on-chip circuitry for compressed sampling is simple and low-power. It is capable of encoding both neural spikes and the noise in-between. The authors use a programmable (external) absolute threshold for neural spike detection to choose between two different recovery dictionaries. One dictionary is designed to reconstruct spike detections which requires a training phase. The other dictionary is intended to reconstruct the noisy samples. The number of samples used to represent a spike can be changed, but high classification accuracy occurs for 10 samples and up in the high SNR region. This compression ratio is equivalent to the compression provided by the DWT presented in this chapter. The main advantage of the DWT approach is that it needs less samples in signals with a high noise level compared to compressed sensing. However, the power dissipation of the compressed sampling calculation is very small at 0.27 μW at 0.6 V per channel (continuous) whereas the DWT is around 1.2 μW per channel (dynamic) in spike detection mode. The area requirement for the compressed sampling is also very small at 200x300 μm versus the 500x500 μm required by a 3-level DWT module. Overall, the signal dependent compressed sampling approach has some strong advantages in power dissipation and area, but has poorer compression performance in low SNR signals compared to the wavelet approach presented in this chapter.

Chapter 6

Neural Spike Feature Extraction: Simulations, Implementations and Results

In this chapter the simulation of a neural spike feature extraction method is presented. The goal is to find a small number of features that can represent the different types of spikes seen in the neural recordings. The features can then be used to correctly classify (spike sort) neural spikes without reconstructing them. The extracted features are also used to generate a codebook. We explore using PCA, k-means, and fuzzy c-means (FCM) clustering on the features and measure the classification accuracy. The first section briefly reviews spike classification and presents the selection of features used to represent neural signals and their clustering performance with respect to classification accuracy. The second section presents the post-layout simulation results of the spike detector and feature extractor. The last section discusses the results, future work, and compares the work to state-of-the-art examples.

6.1 Neural Spike Classification

One of the purposes of neural signal recording is to identify and classify the signals that the nervous system utilizes for sensory and motor functions. The previous chapters have assumed that a close approximation of the neural spike shape is needed to properly identify or cluster spikes into their respective neural sources. This chapter takes the operation of clustering and moves a portion of the calculation into the implant. Only the features required for the spike clustering/classification are extracted and transmitted. The problem is to identify a small set of easy-to-compute features that can provide the necessary dimensions for discriminating different neural sources. Three types of clustering are used to evaluate the features and their ability to properly classify spikes. The most popular approach is PCA, but k-means clustering and FCM clustering are explored in hopes of developing them into automated methods later. Other supervised and unsupervised clustering techniques are left for future work.

6.1.1 Neural Spike Feature Extraction

The enumeration of potentially useful neural spike features can create a very long list, especially if transformations like the DWT [27], PCA [51], and discrete derivative [75] are used. Therefore the list of features is confined to aspects that can be computed inexpensively

from the time-domain representation of the neural spike. The features under evaluation for classification are listed below and shown in Figure 6.1.

1. Largest positive peak amplitude
2. Largest negative peak amplitude
3. Largest positive peak sample location
4. Largest negative peak sample location
5. Polarity (positive peak or negative trough first)*
6. Peak-to-peak amplitude
7. Area of first positive peak*
8. Area of first negative peak*
9. First positive peak start sample location*
10. First positive peak end sample location*
11. First negative peak start sample location*
12. First negative peak end sample location*
13. Positive peak width
14. Negative peak width

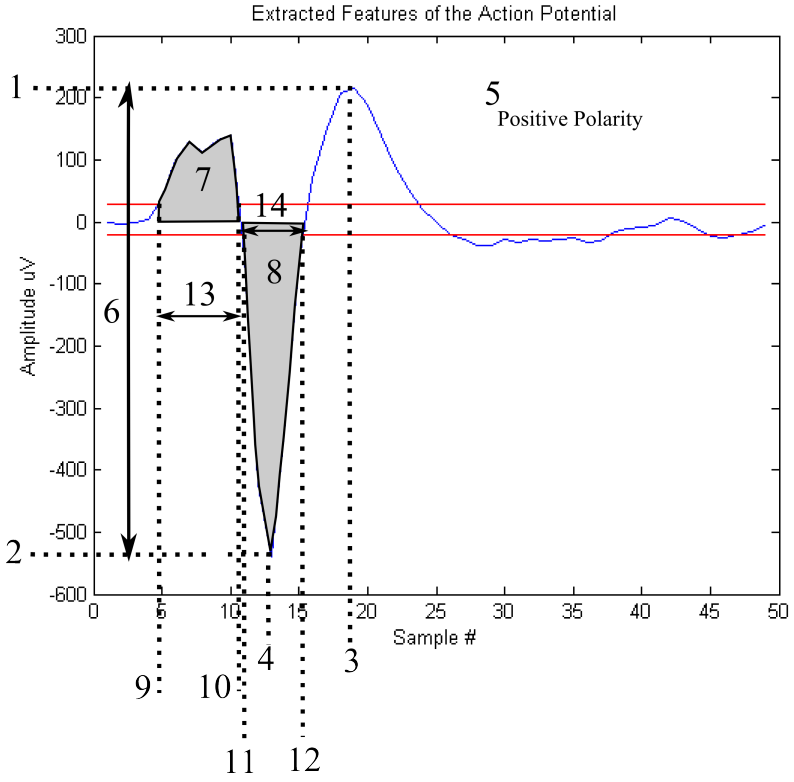


Figure 6.1: The possible features for extraction.

The most appropriate features for clustering may depend on the clustering method used. In order to determine which features are needed, a round of simulations is performed and the ability of the feature to classify the spikes is assessed. The classification of spikes defined only be their extracted features is compared to the classification of the same spikes given the entire spike shape. In addition, any false detections from the feline DRG recordings are considered as well. The dual-threshold noise envelope spike detector from Chapter 4 is used to detect spikes. The number of unique spike shapes per channel is known to be variable, but small, and therefore a supervised analysis of the results is taken. This means there may be user bias in the classification results. Figure 6.2 shows plots of the 14 features for every spike detection in a sample channel. Each plot has the feature characteristic for each spike class on the y-axis and the x-axis is the spike number for its class. The colours mean the spike belongs to a specific spike class previously determined using PCA and complete spike shape information. Therefore plots that show data points of the same colour in the same area and separate from other data points are good candidates for feature extraction. For example, the polarity feature is an excellent feature to be included. It creates two distinct groups, whereas the largest positive peak has lots of classes with the same feature value.

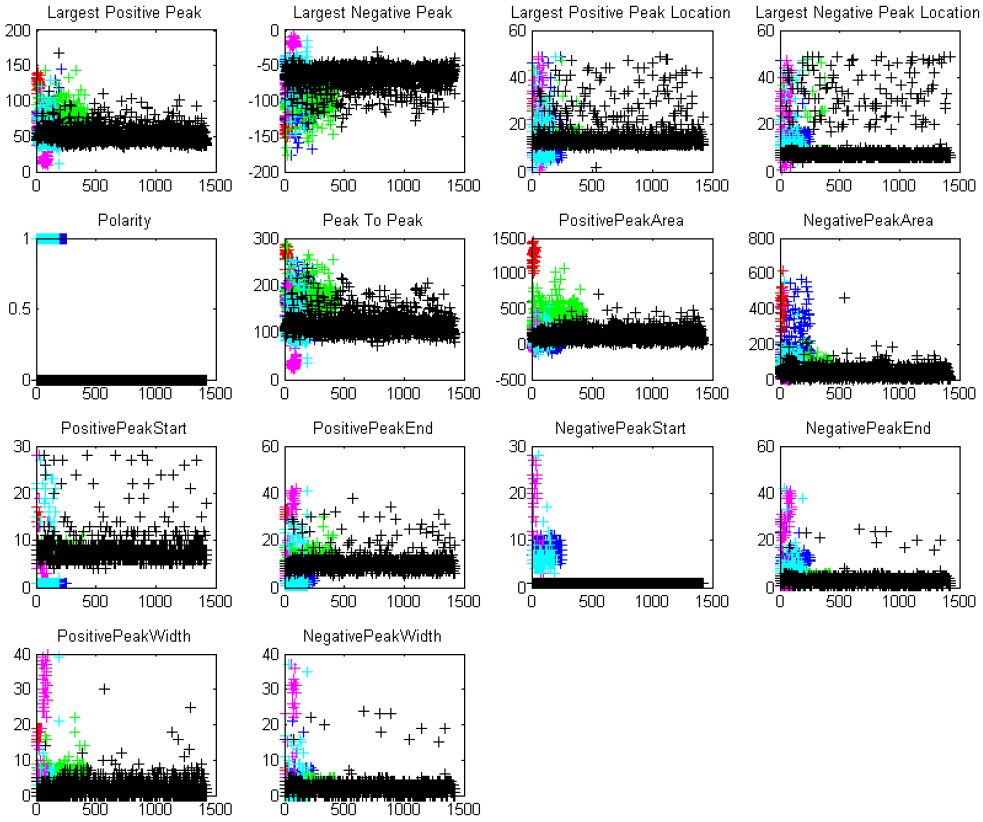


Figure 6.2: The possible features for extraction and their individual clusters on a single channel.

PCA Clustering with Extracted Features

The PCA clusters for spike detections of the sample channel are shown in Figure 6.3a. This plot is generated by taking every sample from every spike detection in a recording and applying a PCA. The first two dimensions or components of the PCA output are then plotted as 'Score 1' and 'Score 2'. The colours are added after the examination of the spike

shapes and individual neural sources identified. The process of defining neural sources is iterative and supervised. For example, all green data points are believed to come from a single neural source that is different from all other neural sources on this recording because it has the appearance of a cluster. This is confirmed by plotting a superposition of all the spikes in that cluster and making sure that the 'general' spike shape is the same. The colours from Figure 6.3a match the colours used in Figure 6.2. If we try to recreate Figure 6.3a using only the 14 extracted features we produce Figure 6.3b. The PCA is unable to create easily classified clusters by just using the feature-based information as-is. For example, the spikes from the 'black' neural source are now overlapping spikes from other neural sources. Therefore, if we attempted classification using only clustering on the PCA plot, we would be unable to regain the true neural sources as depicted in Figure 6.3a.

K-means Clustering with Extracted Features

The supervised k-means clustering used here requires the number of clusters [58] as a supervised input. We assume 6 clusters. When the full spike shape information for every spike detection is provided to the k-means clustering algorithm the resulting cluster centres are plotted in Figure 6.4. These represent what a spike that belongs to that cluster should look like. The plots are recognizable as neural spikes, or shapes similar to neural spikes. The superposition of all the spike detections according to their calculated class is shown on the left in Figure 6.5. The k-means result is similar to the superposition plot provided by the PCA, which is plotted on the right in Figure 6.5, but the order of the classes has been rearranged.

The feature-based clustering for k-means classification appears to be more promising than PCA. The left plots of Figure 6.5 are recreated in Figure 6.6, but this time the k-means algorithm could only assign classes based on the extracted features. The new plots appear to be fairly similar to the plots using the full spike information. The spike counts of each class are indicated in the title of each plot. The centres of the features are plotted in Figure 6.7. The y-axis of each plot is the feature value and the x-axis of each plot is the class number. A feature is a good discriminator if it is has large peaks.

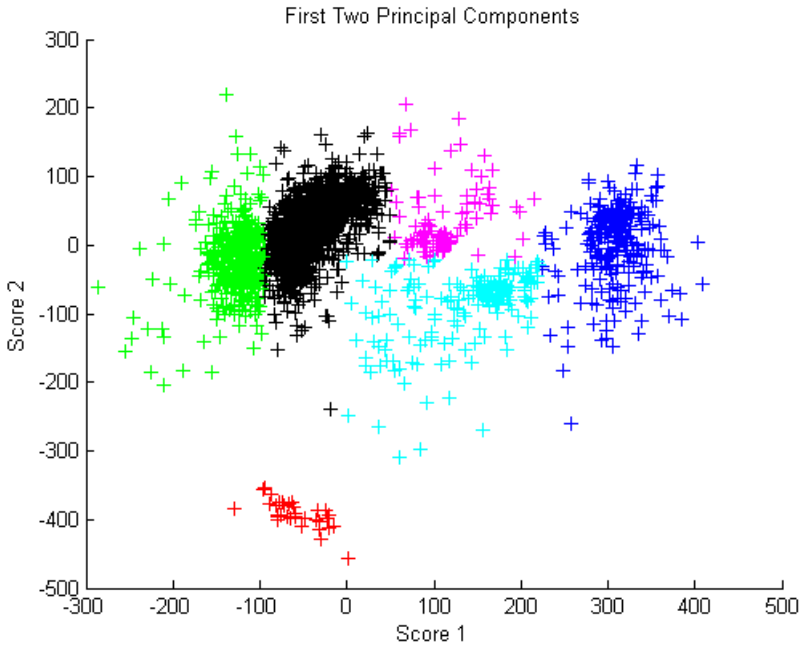
Comparing the original k-means clusters using full spike shape information to the feature-based clusters, the difference in classifications is 24%. Therefore, one in four spikes is assigned to the wrong class compared to the original k-means clusters. If the results are compared to the PCA classifications the difference is worse at 35% wrongly classified.

Fuzzy C-Means Clustering with Extracted Features

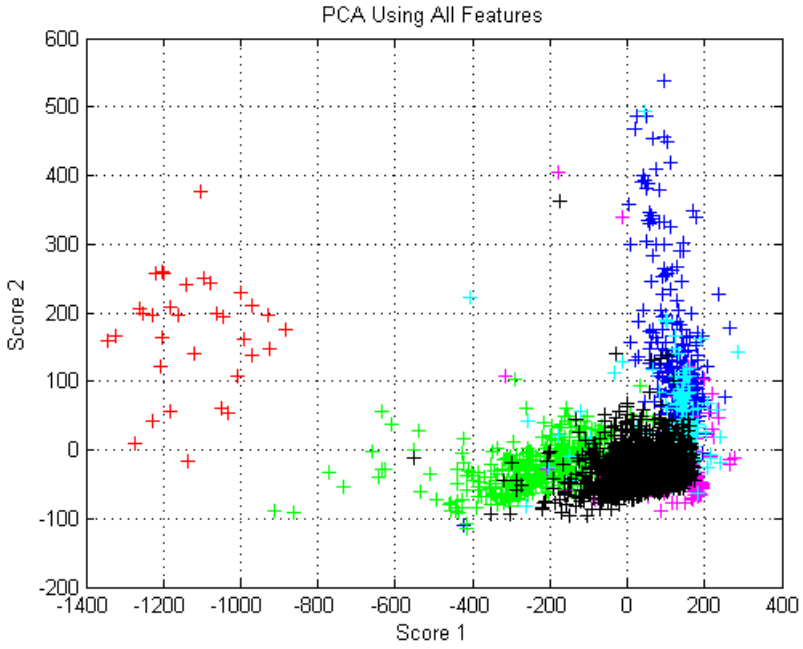
The supervised FCM clustering approach is selected due to its known fast implementations and incremental methods for time-series [106]. The method used for classification requires only the number of templates to generate, which is kept at 6. The output of the FCM clustering method is a probability that a spike belongs to a template class and the input is the same as k-means and PCA. The probability is based on calculated centres. The centres for each of the fourteen features are shown in Figure 6.8.

If all the samples from every spike detection are analysed using FCM clustering a different clustering from PCA and k-means is created as shown in Figure 6.9. The red class spikes, with a very wide positive peak, are absorbed into the first class now shown in the top-left plot in magenta. In addition, the green and blue plots have very similar positive and negative peaks.

The clusters/classes created from performing FCM clustering on the extracted features are plotted in Figure 6.10. These plots exhibit a classification error rate of 47% compared to full spike shape clustering with PCA. This means that the k-means algorithm appears to be the best algorithm for classifying neural spikes based on extracted features.



(a) Manual clustering using first two components of PCA.



(b) The first two principal components using all of the possible features

Figure 6.3: PCA classification of spike detections plotted in the PCA feature space.

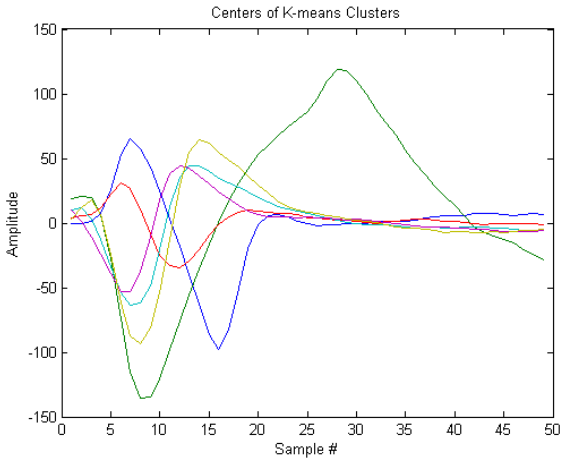


Figure 6.4: The calculated centres of the 6 clusters formed in the k-means method.

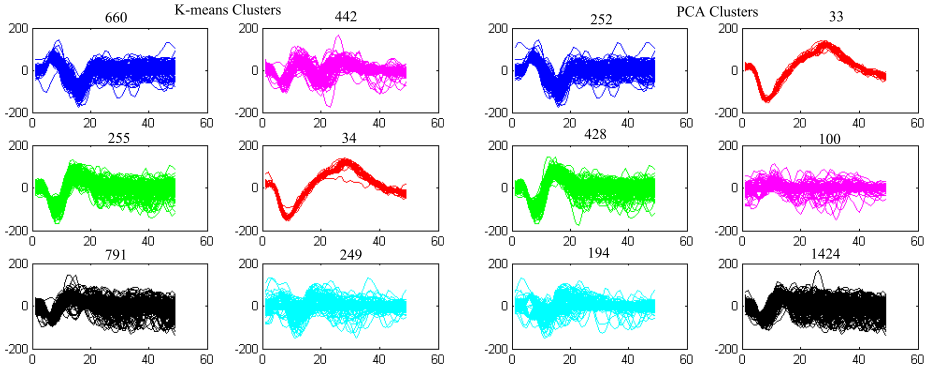


Figure 6.5: All the spike detections plotted in their respective classes. Left is k-means. Right is PCA. Colours match between the two plots. The title of each plot is the number of spike detections in the class.

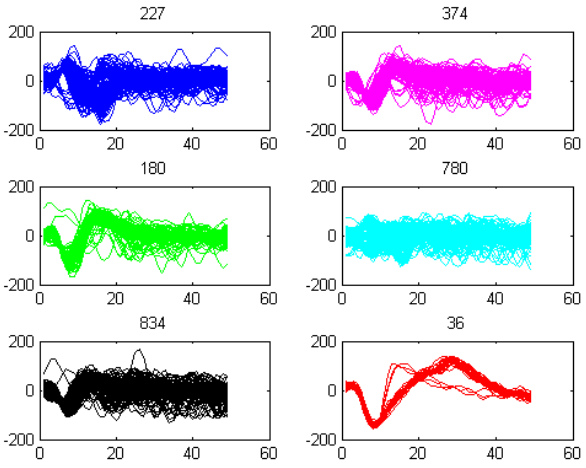


Figure 6.6: All the spike detections plotted in their respective classes using only the features extracted. The title of each plot is the number of spike detections in the class. Classes assigned based on the k-means algorithm.

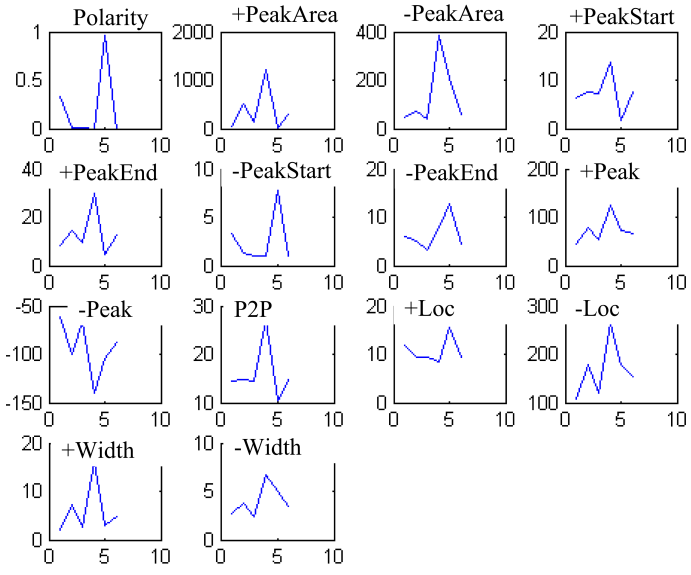


Figure 6.7: The calculated centres of the 6 clusters formed in the k-means method using only the features provided.

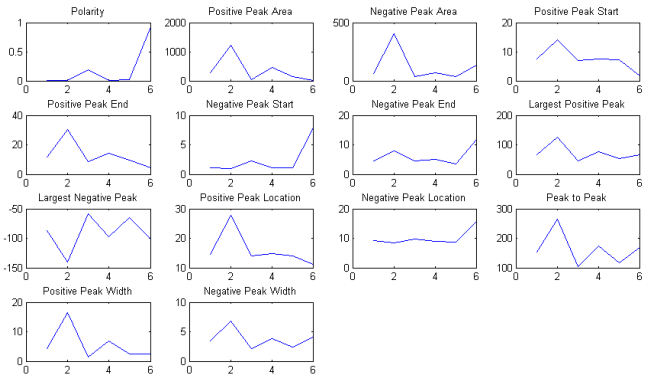


Figure 6.8: The FCM centres generated for 6 classes given the fourteen features for 2431 spike detections.

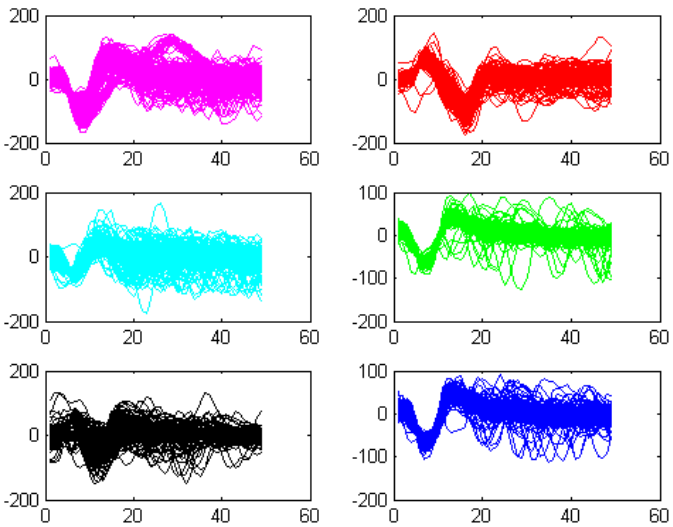


Figure 6.9: The superposition of spikes given 6 templates using fuzzy c-means clustering. Class assignment based on clustering with full spike information.

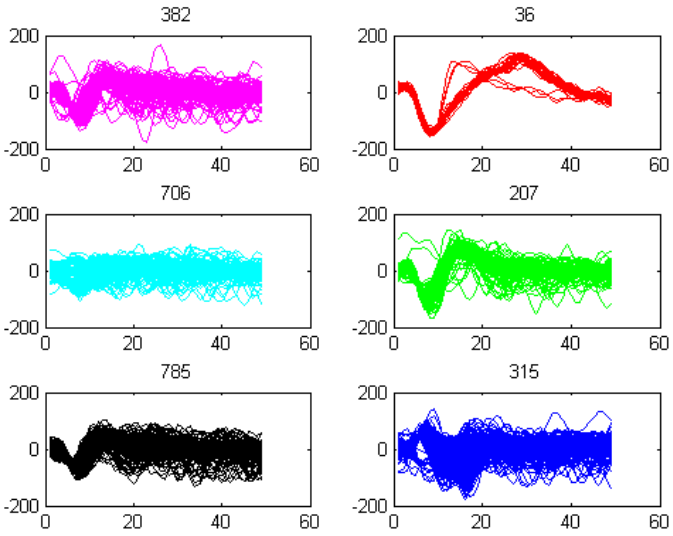


Figure 6.10: All the spike detections plotted in their respective classes using only the extracted features. The title of each plot is the number of spike detections in the class. Classes assigned based on FCM clustering.

6.1.2 Improving Feature Extraction Classification

In the previous section the PCA, k-means and FCM algorithms were used to classify the extracted features into their classes. PCA was not quantified because the classification was going to fail and k-means and FCM have a classification error of 25% and 47%, respectively. This is an unacceptable amount of error. In this section the classification method is given an additional step to improve the classification accuracy.

Using all the features with equal importance/weight does not work. A weighted approach, meaning that some features are more important than others, could give the separation needed in the feature space to give reliable classification with respect to the manual PCA of full spike shapes. The simplest extracted feature is the *polarity*. From observations of Figures 6.4, 6.7 and 6.8, the feature *polarity* also plays an important role in separating classes. Two spikes from different neural sources could potentially be identical in every other way except in their polarity. Therefore the first step to building a better classifier is to use the polarity as a hard classifier and split the classification of extracted features into two groups with different polarity. In addition, the number of classes should be large enough to contain a large number of neural sources. Therefore the total number of possible classes is increased to 16 as two groups of 8. The new clusters from k-means and FCM are shown in Figure 6.11. The PCA classification is not shown since it still performs poorly. By selecting and adding some of the new 16 classes together a classification of the original 6 classes can be found. The new classification error is 7% and 4% for k-means and FCM, respectively.

Another aspect of the classification of neural spikes using extracted features is the option to provide approximate representations of the classes. This option can also be used to find classes that are very similar and should be joined together into a single class. This could also be considered a codebook. A template for a class is generated based on the first 10% of the spike detections. This means that both the full spike detections are transmitted and the extracted features calculated. The templates for the example channel with FCM clustering are plotted in Figure 6.12. The blue plots are a superposition of the spikes assigned to the class even though only extracted features were used to calculate classes.

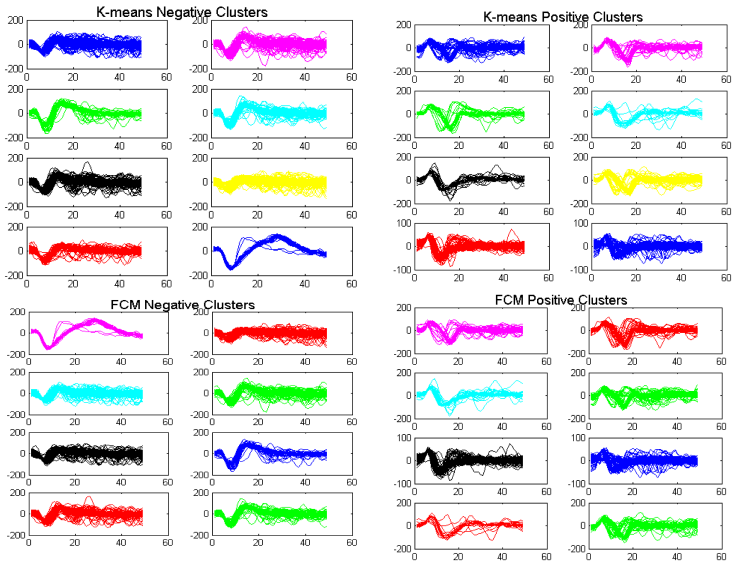


Figure 6.11: All the spike detections plotted in their respective classes using only the features extracted. The title of each plot is the number of spike detections in the class.

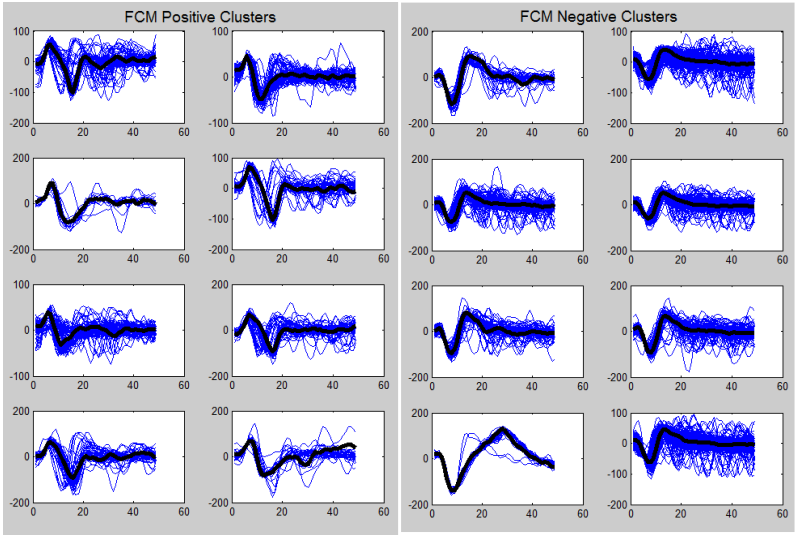


Figure 6.12: Classes based on FCM clustering with templates plotted in black.

6.2 Implementation of Feature Extraction

The feature extraction architecture does not depend on the clustering method of the extracted features since both k-means and FCM clustering would use the same features. The performance of the classification is not effected if we use only the seven extracted features listed in Table 6.1.

Table 6.1: Seven required features for clustering.

Polarity (positive peak or negative trough first)
Area of first positive peak
Area of first negative peak
First positive peak start location
First positive peak end location
First negative peak start location
First negative peak end location

Many of the other features can be calculated or approximated by using only these features if needed. The dual-threshold spike detector must be utilized as it allows the identification of both peaks and troughs, which is needed for the extracted features. The state machine used is similar to the dual-threshold spike detector except that extracted values are stored. A block diagram of the system is shown in Figure 6.13. The state machine is slightly more complicated than the original spike detector because several counters are required to track the peak area. The area feature also requires the use of an accumulator, which can sum the areas of both peaks and troughs. All features are calculated as the spike samples are written to memory. This means that on the first threshold crossing, depending on the polarity, some of the features are calculated.

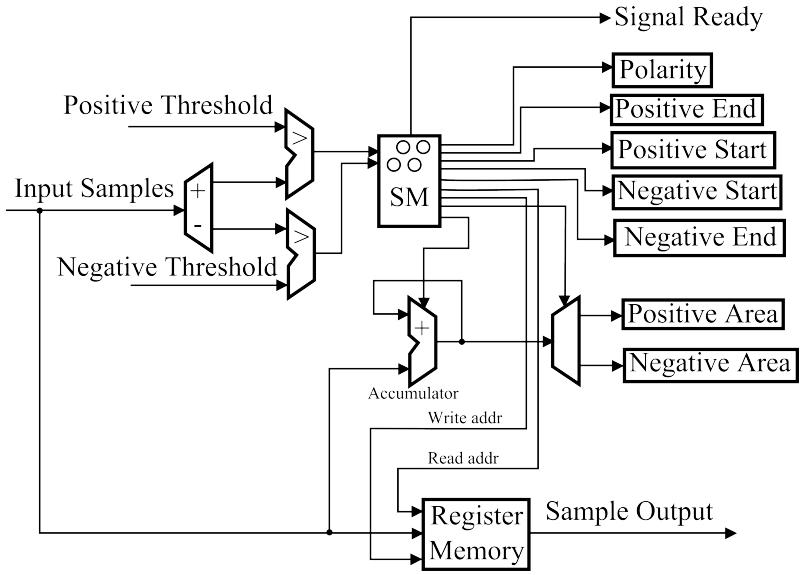


Figure 6.13: Block diagram of the components required to make dual-threshold spike detector with feature extraction.

6.3 Simulated Results

The logical equivalence between the synthesized designs and Matlab simulations is done using netlists from Synopsys Design Compiler and Cadence First Encounter for post-layout confirmation. The design is specified in VHDL and the post-layout netlist is synthesized using the IBM 130-nm design kit. The design flow stages are the same as NC1, but only replicated here for a single feature extraction module. The VHDL testbenches use an ideal clock frequency with period of 30,000 ns. The Synopsys Design Compiler synthesized results for the feature extraction state machine are shown in Table 6.2.

Table 6.2: Power and area estimate for the feature extraction module versus the spike detector without feature extraction

Metric	No Extraction	With Extraction
Voltage	1.2 V	1.2 V
Area (μm^2)	29,890	35,740
Timing (slack ns)	29,993	29,991
Dynamic Power (nW)	131.3	184.5
Leakage Power (nW)	34.91	42.11

The feature extraction module is estimated to dissipate 36% more power than regular spike detection. The synthesized logic takes up 19% more area with 5000 μm^2. This extra expense, however, is able to yield a 87% reduction in transmission per spike size. If the measured power dissipation of 21 μW, from NC1 at 0.8 V, is increased by 36% the expected power dissipation is 28.5 μW with an energy per bit of 7.4 pJ/bit. Therefore, at a cost of 2 pJ per input bit, the module saves the system 87% of its transmission cost per spike, while only suffering a 4% penalty in classification accuracy.

6.4 Discussion and Contributions

The goal of the feature extraction circuit is to provide an alternative method of encoding neural spikes that allows accurate spike sorting and also saves an implanted wireless system considerable energy compared to sending the entire spike. The power dissipation of the spike detection and the feature extraction is considerably smaller than the expected power dissipation of a wireless transmitter with an energy per bit of 1 nJ/bit compared to 7.4 pJ/bit. The 19% increase in area is small compared to wavelet encoding, but the system is not able to accurately reconstruct the shape of the spike, but only give templates (averages). The simulation results of the feature extraction circuit led to a publication [107] where dual-threshold spike detection was improved for the extracted feature case and classification accuracies within 1% are demonstrated on a neural recording from the feline DRG. The accuracy was improved because some false detections were skewing the results and a special rule was introduced to avoid them.

If NC1 is considered, the power cost of adding this extra level of compression is measured and projected to be an extra 2 pJ per input bit for a total of 7.4 pJ/bit, which is more than the energy increase from the wavelet modules in NC2. This happens because the wavelet decomposition only occurs on successful spike detections, whereas the first possible peak detected activates several features to be extracted. The decision to calculate some features on threshold-crossings negatively impacted the power dissipation compared to the wavelet module that waits for a full spike detection. The number of threshold-crossings is far greater than the number of spikes detected. This can be changed in a future version, so that the features are only extracted if a spike is detected instead of calculating the feature for each peak/trough threshold-crossing.

6.4.1 Comparison to the State-of-the-Art

In [108] a feature extraction and classification method requiring few spike features is presented. The authors calculate the widths of the peak and trough or the two distances between the three zero-crossings of a spike detection. They utilize a dual-threshold spike detection method with supervised thresholds. The major difference of this work, with respect to feature extraction and classification is the use of Mahalanobis clustering which provides better clustering performance than k-means and PCA. Therefore the authors have provided a smaller set of features that can attain excellent spike classification equal to the classification accuracy presented here. Interestingly, the authors explicitly state that "the overall system performance is compromised mainly by poor spike detection rather that inadequate feature extraction or classification" which enforces the value of having a spike detector, like the one presented in this thesis, that avoids false detections. No circuit implementation or estimates are provided for hardware comparison.

In [105] a neural spike recording channel implementation is presented. The authors use a piece-wise linear approximation of the spike shape as features for spike classification. The classification accuracy is evaluated with an artificial signal and the accuracy is around 90% at a SNR of 6 dB. This is equivalent or greater than the SNR of the neural recordings from the DRG since 6 dB means the spike has an amplitude twice as large as the noise. The digital portion of their design does not include spike detection, but only feature extraction. It requires 200 nW for a single channel and provides compression of up to 56 bits per detected spike, while the proposed feature extractor from this chapter would require 49 bits per spike detection and is expected to dissipate an estimated 1.8 μW with spike detection included. Therefore the presented feature extraction method outperforms with respect to classification accuracy, but may require more power per channel to provide approximately the same amount of compression.

Chapter 7

Conclusion

This thesis explores some signal processing techniques and implementations for the compression of neural signals. It identifies neural spike detection, discrete wavelet transforms, neural spike feature extraction and compressed sampling as possibilities for implementation. Three methods of compression are investigated in-depth, with two silicon implementations presented: windowed dual-threshold spike detection, three-level discrete wavelet transform with coefficient thresholds, and feature extraction with codebook approximations. It was found that implementing a method of dual-threshold spike detection reduces the expected power dissipation of the implanted system and provides, typically, over 90% signal compression. By applying an adaptive noise-based threshold the neural spike detector can avoid 50% of false detections compared to other methods. The energy cost of this spike detection was measured in a test chip NC1 to 6 pJ per input bit. This is higher than state-of-the-art implementations [8], [33] that operate on 1-2 pJ per input bit, but lower than the estimated 1 nJ/bit cost of transmitting the bits wirelessly.

A second test chip, NC2, was designed and fabricated to measure the cost of additional compression through multilevel wavelet decomposition. It was found that by making additional signal processing dependent on the neural spike firing rates a large number of processing steps on each spike is possible, with a relatively small cost in energy consumption. The 3-level wavelet decomposition resulted in a 1 pJ per input bit increase in energy dissipation. It was further calculated that the 15% increase in dynamic switching power was relatively low considering the 575% increase in logic area. Therefore, the focus of any wireless implantable spike detector should be on the power reduction of the analog front-end and spike detector. In addition to the two test chips, a simulation of a feature extraction implementation based on NC1 is estimated to consume 7.4 pJ per input bit. The feature extraction method can provide 87% compression per spike, while still allowing near perfect spike classification on experimental neural recordings from the DRG. The increase in logic area was only 19% compared to the spike detector alone, but the increase in energy consumption was greater than the 3-level wavelet decomposition. This, in part, reaffirms the idea that the front-end of the spike detector is most important when aiming for a low-power implementation.

The next section presents the main contributions of the thesis work and reiterates the comparisons to other published work from each chapter.

7.1 Technical Contributions and Comparison to the State-of-the-Art

1. I developed a spike detection algorithm that is shown to reduce false detections in the recordings of feline DRGs. The noise envelope dual-threshold spike detector is

shown in [97] to reduce false detections by 50% for artificial recordings with different noise levels and can reduce false detections by 77% for experimental neural recordings from the feline DRG. The algorithm was implemented in a FPGA and in a test chip named NC1. The power dissipation of NC1 was measured to be higher than the state-of-the-art with respect to input bits, but methods of reducing the power in the future are identified. The simulations and power measurements resulted in two publications, [109] and [97]. This work has reduced the number of false detections during spike detection compared to other published solutions as discussed below.

(a) In [8] the authors use a programmable absolute threshold for neural spike detection. The absolute threshold is shown in [97] to have 50% more false detections than the method presented in this thesis. However, the power dissipation of their method is 0.27 μW at 0.6 V per channel whereas the dual-threshold spike detector requires 1.3 μW per channel. The area requirements for the two methods are similar. Overall, the absolute threshold approach has an advantage in power dissipation, but has poorer spike detection performance, which may, depending on the number of false detections, cost more energy than the small increase seen in the dual-threshold spike detector from this thesis.

(b) In [98] the authors use a spike detection method that can operate in either single-threshold or dual-threshold modes, but the thresholds are set externally. The single threshold method suffers the same false detection rates as the absolute threshold method, but the dual-threshold, if programmed correctly, will match the performance of the noise-envelope based threshold presented in this thesis. The thresholds are implemented in an analog circuit, but the decision to actually transmit a spike is made in a microcontroller. The microcontroller is external to the analog ASIC and has large power consumption compared to this work. The lowest measured current from the microcontroller is 13 mA. The power budget for their activity-aware signal capturing is an order of magnitude larger than the spike detector presented in this thesis. The power savings come from the dedicated circuitry for spike detection versus a generic microcontroller architecture.

(c) In [99] the authors introduce an analog circuit that calculates the NEO output as well as a threshold that adapts to the signal conditions. The simulation results indicate a low rate of false detections of around 1%, however the detection of actual spikes suffers with 30% missed detections on a 'easy' neural recording. The estimated power dissipation is 1.5 μW at 1.8 V with a logic area of 0.03 mm^2. The spike detector presented in this thesis has similar true detection rates, with a slightly larger false detection rate, but dissipates less power at 1.3 μW in a larger area (0.04 mm^2).

2. I developed a neural spike compression method based on known wavelet encoding techniques. The method had not been applied to neural spike detection and is shown to offer compression up to 88% per spike at an extra energy cost of 1 pJ/bit compared to spike detection alone. The extra compression is shown not to decrease the classification accuracy of the neural spikes, which is similar to the performance shown in [8]. The three level wavelet decomposition was not fully tested due to a design error. However, the test chip NC2 was functional enough to provide an energy measurement for the one functional mode. The simulations and power measurements of this work also contributed to the publication of [97]. The amount of compression provided by the wavelet decomposition is similar to the state-of-the-art, but is known to have better compression for signals with low SNR [8].

(a) In [105] the authors use a piece-wise linear approximation of the spike shape as features for spike compression and classification. The classification accuracy is

evaluated with an artificial signal and the accuracy is 90% at a SNR of 6 dB. The feature extraction requires 200 nW for a single channel and provides compression of up to 56 bits per detected spike, while the proposed feature compression from this thesis would require 80 bits per spike detection ($K = 10$) and is measured to require 1.51 μA at 0.8 V for 1.2 μW (dynamic power). Therefore the presented DWT compression method outperforms with respect to classification accuracy, but requires more power per channel to provide a smaller amount of compression per spike. The wavelet approach described in this chapter also allows the compression of continuous data, but this could not be measured in terms of power.

(b) In [8] the authors use a signal-dependent model-based recovery dictionary to improve upon model-based compressed sensing recovery. The on-chip circuitry for compressed sampling is capable of encoding both neural spikes and the noise in-between. The number of samples used to represent a spike can be changed, but high classification accuracy occurs for 10 samples and up in the high SNR region. This compression ratio is equivalent to the compression provided by the DWT presented in this thesis. The main advantage of the DWT approach is that it needs less samples in signals with a high noise level compared to compressed sensing. However, the power dissipation of the compressed sampling is 0.27 μW at 0.6 V per channel (continuous) whereas the DWT is 1.2 μW per channel (dynamic) in spike detection mode. Overall, the signal dependent compressed sampling approach has some strong advantages in power dissipation and area, but has poorer compression performance in low SNR signals compared to the wavelet approach presented in this thesis.

3. I developed a feature extraction method for spike classification, which was shown to be within 1% for recordings of feline DRG [107]. This work was only simulated up to post-layout. The classification accuracy is over 95% while the state-of-the-art has errors of 10% [105]. Estimates on power were extrapolated based on percentage increases and the design is estimated to dissipate 1.4 pJ more per input bit compared to spike detection alone. The point of computation in the spike detection algorithm is argued to be the main reason behind the relatively large increase in energy dissipation. The simulated results and improved spike detection are presented in a publication [107].

   (a) In [108] the authors calculate the widths of the peak and trough or the two distances between the three zero-crossings of a spike detection. They utilize a dual-threshold spike detection method with supervised thresholds. The major difference of this work, with respect to feature extraction and classification is the use of Mahalanobis clustering which provides better clustering performance than k-means and PCA. Therefore the authors have provided a smaller set of features that can attain excellent spike classification equal to the classification accuracy presented here. Interestingly, the authors explicitly state that "the overall system performance is compromised mainly by poor spike detection rather that inadequate feature extraction or classification" which enforces the value of having a spike detector, like the one presented in this thesis, that avoids false detections. No circuit implementation or estimates are provided for hardware comparison.

   (b) In [105] the authors use a piece-wise linear approximation of the spike shape as features for spike classification. The classification accuracy is evaluated with an artificial signal and the accuracy is 90% at a SNR of 6 dB. It requires 200 nW for a single channel and provides compression of up to 56 bits per detected spike, while the proposed feature extractor from this chapter would require 49 bits per spike detection and is expected to dissipate an estimated 1.8 μW with spike detection

included. Therefore the presented feature extraction method outperforms with respect to classification accuracy, but may require more power per channel to provide approximately the same amount of compression.

7.2 Future Work

There are multiple directions for future work. The following list describes some possibilities starting with those that have the most potential working towards those with less promise.

- Pursue the development of a monolithic implementation that contains spike detection, wavelet decomposition and feature extraction with the analog front-end in one chip and eventually the wireless module too. One key feature for the signal processor would be to give operational modes so that the user can acquire the desired data at different levels of detail ranging from raw waveforms to spike detection rates. The spike detection rates could be based on a spike sorting-like operation that is based on feature extraction methods from Chapter 6.
- Design the circuits with low-power techniques such as sub-threshold circuit operation, power gating, and asynchronous operation. This direction would be best split into two projects with asynchronous approaches separate from low-power approaches because asynchronous design does not typically work well with standard tools and it could be very troublesome to properly simulate a large design.
- Develop a compression system to work at 100's of MHz such that the circuit only needs to operate in very quick bursts and then turn off while acquiring more data.
- Develop a system architecture that is flexible enough to provide resources to only the working electrodes. This could mean less overall circuitry and therefore a smaller power dissipation. This approach could take advantage of the fact that most electrodes on the electrode array do not intercept any neural signals above the noise.

Bibliography

[1] D. Johnston and S. Wu, *Foundations of Cellular Neurophysiology.* MIT Press, 1995.

[2] A. L. Hodgkin and A. F. Huxley, "Currents carried by sodium and potassium ions through the membrane of the giant axon of Loligo," *J. Physiol. (Lond)*, vol. 116, pp. 449–472, 1952a.

[3] ——, "The components of membrane conductance in the giant axon of Loligo," *J. Physiol. (Lond)*, vol. 116, pp. 473–496, 1952b.

[4] ——, "The dual effect of membrane potential on sodium conductance in the giant axon of Loligo," *J. Physiol. (Lond)*, vol. 116, pp. 497–506, 1952c.

[5] ——, "A quantitative description of membrane conductance and its application to conduction and excitation in nerve," *J. Physiol. (Lond)*, vol. 117, pp. 500–544, 1952d.

[6] E. Izhikevich, "Which model to use for cortical spiking neuron?" *IEEE Transactions on Neural Networks*, vol. 14, no. 5, pp. 1063–1070, 2004.

[7] A. S. Cassidy *et al.*, "Cognitive computing building block: a versatile and efficient digital neuron model for neurosynaptic cores," *International Joint Conference on Neural Networks*, p. 10 pgs., 2013.

[8] J. Zhang *et al.*, "An efficient and compact compressed sensing microsystem for implantable neural recordings," *IEEE Transaction on Biomedical Circuits and Systems*, vol. PP, no. 99, p. 13, 2013.

[9] R. B. Stein, E. R. Gossen, and K. E. Jones, "Neuronal variability: noise or part of the signal?" *Nature Reviews Neuroscience*, vol. 6, pp. 389–397, May 2005.

[10] G. E. Moore, "Cramming more components onto integrated circuits," *Electronics*, vol. 38, no. 8, April 1965.

[11] R. Harrison, "A low-power low-noise CMOS amplifier for neural recording applications," *IEEE Journal of Solid-State Circuits*, vol. 38, no. 6, 2003.

[12] T. Seese, H. Harasaki, G. Saidel, and C. Davies, "Characterization of tissue morphology, angiogenesis, and temperature in the adaptive response of muscle tissue to chronic heating," *Laboratory Investigation*, vol. 78, no. 12, pp. 1553–1562, 1998.

[13] [Online]. Available: www.smartneuralprostheses.com

[14] R. B. Stein and V. K. Mushahwar, "Reanimating limb movements after injury or disease," *Trends in Neuroscience*, vol. 28, pp. 518–524, 2005.

[15] A. L. Hodgkin and B. Katz, "The effect of sodium ions on the electrical activity of the giant axon of the squid," *J. Physiol. (Lond)*, vol. 108, pp. 37–77, 1949.

[16] E. Schneidman, B. Freedman, and I. Segev, "Ion channel stochasticity may be critical in determining the reliability and precision of spike timing," *Neural Computation*, pp. 1679–1703, 1998.

[17] R. Stein, "Some models of neuronal variability," *Biophysical Journal*, vol. 7, pp. 37–68, 1967.

[18] E. Zahedi and M. A. M. Ali, "Implants for human rehabilitation," *ICSE 2002, Penang Malaysia*, pp. 39–42, 2002.

[19] H. Wark *et al.*, "A new high-density (25 electrodes/mm2) penetrating microelectrode array for recording and stimulating sub-millimeter neuroanatomical structures," *Journal of Neural Engineering*, vol. 10, p. 10, August 2013.

[20] N. Joye, A. Schmid, and Y. Leblebici, "An electrical model of the cell-electrode interface for high-density microelectrode arrays," *IEEE Engineering in Medicine and Biology Society Conference*, pp. 559–562, August 2008.

[21] J. Guo, J. Yuan, and M. Chan, "Modeling of the cell-electrode interface noise for microelectrode arrays," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 6, no. 6, pp. 605–613, December 2012.

[22] A. Branner, "Long-term stimulation and recording with a penetrating microelectrode array in cat sciatic nerve," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 1, pp. 146–157, January 2004.

[23] E. N. Brown, R. E. Kass, and P. P. Mitra, "Multiple neural spike train data analysis: state-of-the-art and future challenges," *Nature Neuroscience*, vol. 7, no. 5, pp. 456–462, May 2004.

[24] *Cerebus Neural Signal Processor User's Manual*, Blackrock Microsystems.

[25] P. T. Thorbergsson, "Signal modeling and data reduction for wireless brain-machine interfaces," Ph.D. dissertation, Faculty of Engineering, LTH, Lund University, 2012.

[26] C. Dumortier, B. Gosselin, and M. Sawan, "Wavelet transforms dedicated to compress recorded ENGs from multichannel implants: comparitive architectural study," *IEEE ISCAS*, pp. 2129–2132, 2006.

[27] K. Oweiss, A. Mason, Y. Suhail, A. Kamboh, and K. Thomson, "A scalable wavelet transform VLSI architecture for real-time signal processing high-density intra-cortical implants," *IEEE Transaction on Circuits and System I: Regular Papers*, vol. 54, no. 6, pp. 1266–1278, 2007.

[28] A. M. Kamboh, M. Raetz, K. Oweiss, and A. Mason, "Area-power efficient VLSI implementations of multichannel DWT for data compression in implantable neuroprosthetics," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 1, no. 2, June 2007.

[29] S. Narasimhan, Y. Zhou, H. Chiel, and S. Bhunia, "Low-power VLSI architecture for neural data compression using vocabulary-based approach," *BIOCAS 2007*, pp. 134–137, 2007.

[30] S. Narasimhan, H. J. Chiel, and S. Bhunia, "A preferential design approach for energy-efficient and robust implantable neural signal processing hardware," *IEEE EMBS*, pp. 6383–6386, 2009.

[31] J. G. Harris, J. C. Principe, J. C. Sanchez, D. Chen, and C. She, "Pulse-based signal compression for implanted neural recording systems," *IEEE ISCAS 2008*, pp. 344–347, 2008.

[32] J. N. Y. Aziz *et al.*, "256-channel neural recording and delta compression microsystem with 3D electrodes," *IEEE Journal of Solid-State Circuits*, vol. 44, no. 3, March 2009.

[33] M. S. Chae, Z. Yang, M. R. Yuce, L. Hoang, and W. Liu, "A 128-channel 6mW wireless neural recording IC with spike feature extraction and UWB transmitter," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 17, no. 4, pp. 312–321, 2009.

[34] T. Szuts *et al.*, "A wireless multi-channel neural amplifier for freely moving animals," *Nature Neuroscience*, vol. 14, no. 2, pp. 263–270, February 2011.

[35] W. Dabrowski, P. Grybos, and A. Litke, "A low noise multichannel integrated circuit for recording neuronal signals using microelectrode arrays," *Biosensors and bioelectronics*, vol. 19, pp. 749–761, 2004.

[36] W. Dabrowski, "Development of front-end ASICs for imaging neuronal activity in live tissue," *Nuclear Instruments and Methods in Physics Research*, vol. 541, pp. 405–411, 2005.

[37] A. Kamboh and A. Mason, "Comparison of lifting and b-spline DWT implementations for implantable neuroprosthetics," *5th IEEE Conference on Sensors*, pp. 811–814, 2006.

[38] K. Oweiss, "A systems approach for data compression and latency reduction in cortically controlled brain machine interfaces," *IEEE Transactions on Biomedical Engineering*, vol. 52, no. 7, pp. 1364–1377, July 2006.

[39] S. Narasimhan, M. Cullins, H. Chiel, and S. Bhunia, "Wavelet-based neural pattern analyzer for behaviorally significant burst pattern recognition," *IEEE Engineering in Medicine and Biology Society Conference*, pp. 38–41, 2008.

[40] S. Narasimhan, H. J. Chiel, and S. Bhunia, "Ultra-low-power and robust digital-signal-processing hardware for implantable neural interface microsystems," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 5, no. 2, pp. 169–178, April 2011.

[41] M. Ashouei *et al.*, "A voltage-scalable biomedical signal processor running ECG using 13pj/cycle at 1MHz and 0.4V," *IEEE International Solid-State Circuits Conference Digest of Technical Papers 2011*, pp. 332–334, February 2011.

[42] R. Muller, S. Gambini, and J. Rabaey, "A 0.013mm2, 5uW, DC-coupled neural signal acquisition IC with 0.5V supply," *IEEE Journal of Solid-State Circuits*, vol. 47, no. 1, pp. 232–243, January 2012.

[43] R. Harrison *et al.*, "A low-power integrated circuit for a wireless 100-electrode neural recording system," *IEEE Journal of Solid-State Circuits*, vol. 42, no. 1, pp. 123–133, January 2007.

[44] R. Harrison, "The design of integrated circuits to observe brain activity," *Proceedings of the IEEE*, vol. 96, no. 7, pp. 1203–1216, July 2008.

[45] P. Watkins, G. Santhanam, K. Shenoy, and R. Harrison, "Validation system of adaptive threshold spike detector for neural recording," *IEEE Engineering in Medicine and Biology Society Conference*, pp. 4079–4082, September 2004.

[46] B. Microsystems. (2014, January) Data acqucerebus - cerebus. [Online]. Available: http://www.blackrockmicro.com/

[47] I. Obeid and P. Wolf, "Evaluation of spike-detection algorithms for a brain-machine interface application," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 6, pp. 905–911, June 2004.

[48] V. Mushahwar and D. Stein. (2010-2012) Experimental electronic recordings of feline neural recordings. Electronic data provided and may not be publically available.

[49] Y. Suo, J. Zhang, R. Etienne-Cummings, T. Tran *et al.*, "Energy-efficient two-stage compressed sensing method for implantable neural recordings," *IEEE Biomedical Circuits and Systems Conference (BioCAS)*, 2013.

[50] M. L. Homer, A. V. Nurmikko, J. P. Donoghue, and L. R. Hochberg, "Sensors and decoding for intracortical brain computer interfaces," *Annual Review of Biomedical Engineering*, vol. 15, pp. 383–405, 2013.

[51] M. Lewicki, "A review of methods for spike sorting: the detection and classification of neural action potentials," *Network: Computing Neural Systems*, vol. 9, no. 4, pp. 53–78, 1998.

[52] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philos*, vol. 2, no. 11, pp. 559–572, 1901.

[53] H. Hotelling, "Analysis of a complex of statistical variables into principal components," *Journal of Educational Psychology*, vol. 24, pp. 417–441, 1933.

[54] I. Jolliffe, *Principal Component Analysis, Second Edtion.* Springer, 2002.

[55] C. Ding and X. He, "K-means clustering via principal component analysis," *Proceedings of the 21st International Conference on Machine Learning, Banff, Canada*, 2004.

[56] J. MacQueen, "Some methods for classification and analysis of multivariate observations," *Proceedings of 5th Berkely Symposium on Mathematical Statistics and Probability*, vol. 1, pp. 281–297, 1967.

[57] M. Telgarsky and A. Vattani, "Hartigan's method: k-means clustering without Voronoi," *Proceedings of the 13th International Conference on Artificial intelligence and Statistics (AISTATS)*, vol. 9, pp. 820–827, 2010.

[58] A. Coates and A. Y. Ng, "Learning feature representations with K-means," *Neural Networks: Tricks of the Trade, Springer*, p. 20, 2012.

[59] J. Dragas, D. Jackel, F. Franke, and A. Hierlemann, "An unsupervised method for on-chip neural spike detection in multi-electrode recording systems," *IEEE Engineering in Medicine and Biology Society (EMBS)*, pp. 2535–2536, July 2013.

[60] V. S. Polikov, P. Tresco, and W. Reichert, "Response of brain tissue to chronic implanted neural electrodes," *Journal of Neuroscience Methods*, vol. 148, pp. 1–18, 2005.

[61] K. Kim and S. Kim, "Neural spike sorting under nearly 0-dB signal-to-noise tario using nonlinear energy operator and artificial neural-network classifier," *IEEE Transactions on Biomedical Engineering*, vol. 47, no. 10, pp. 1406–1411, October 2000.

[62] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, July, October 1948.

[63] P. Wayner, *Compression Algorithms*. Morgan Kaufman, 2000.

[64] H. Nyquist, "Certain topics in telegraph transmission theory," *Transactions of the American Institute of Electrical Engineers*, vol. 47, no. 2, pp. 617–644, 1928.

[65] D. A. Huffman, "A method for the construction of minimum-redundancy codes," *Proceedings of the IRE*, pp. 1098–1101, September 1952.

[66] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Transactions on Information Theory*, vol. IT-23, no. 3, pp. 337–343, May 1977.

[67] M. Mandal, *Multimedia Signals and Systems*. Kluwer Academic Publishers, 2003.

[68] E. P. Ruzanski, "Effects of MP3 encoding on the sounds of music," *IEEE Potentials*, pp. 43–45, March/April 2006.

[69] T. Berger and J. D. Gibson, "Lossy source coding," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2693–2723, October 1998.

[70] C. E. Shannon, "Coding theorems for a discrete source with a fidelity criterion," *IRE Conv. Rec*, vol. 7, pp. 142–163, 1959.

[71] J. Rigosa, D. Weber, A. Prochaska, and R. S. S. Micera, "Neuro-fuzzy decoding of sensor information from ensembles of simultaneously recorded dorsal root ganglion neuron for functional electrical stimulation applications," *Journal of Neural Engineering*, vol. 8, no. 4, p. 046019, August 2011.

[72] [Online]. Available: http://www.plexon.com/products/offline-sorter

[73] J. Martinez, C. Pedreira, M. Ison, and R. Quiroga, "Realistic simulation of extracellular recordings," *Journal of Neuroscience Methods*, vol. 184, pp. 285–293, 2009.

[74] [Online]. Available: http://www2.le.ac.uk/centres/csn/software/neurocube.

[75] S. Gibson, J. Judy, and D. Markovic, "Comparison of spike-sorting algorithms for future hardware implementation," *IEEE Engineering in Medicine and Biology Society (EMBS)*, pp. 5015–5020, August 2008.

[76] L. Traver, C. Tarin, P. Marti, and N. Cardona, "Adaptive-threshold neural spike detection by noise-envelope tracking," *Electronics Letters*, vol. 43, no. 24, November 2007.

[77] J. F. Kaiser, "On a simple algorithm to calculate the 'energy' of a signal," *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 1, pp. 381–384, April 1990.

[78] S. Mukhopadhyay and G. Ray, "A new interpretation of nonlinear energy operator and its efficacy in spike detection," *IEEE Transactions on Biomedical Engineering*, vol. 45, pp. 180–187, February 1998.

[79] T. Borghi *et al.*, "A power-efficient analog integrated circuit for amplication and detection of neural signals," *IEEE Engineering in Medicine and Biology Society (EMBS)*, pp. 4911–4915, August 2008.

[80] J. Makhoul, "Linear prediction: a tutorial review," *Proceedings of the IE*, vol. 63, no. 4, pp. 561–580, April 1975.

[81] A. Gersho and R. Gray, *Vector quantization and signal compression*. Kluwer Academic Publishers, 1991.

[82] R. Walters, J. C. Principe, and S.-H. Park, "Spike detection using a syntactic pattern recognition approach," *Proceedings of the Annual International Conference on Engineering in Medicine and Biology*, vol. 6, pp. 1810–1811, November 1989.

[83] N. Jayant and P. Noll, *Digital Coding of Waveforms*. Prentice-Hall , Inc., 1984.

[84] H. Hassanieh, P. Indyk, D. Katabi, and E. Price, "Nearly optimal sparse Fourier transform," *44th ACM Symposium on Theory of Computing (STOC)*, p. 12, January 2012.

[85] D. L. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, September 2004.

[86] E. J. Candes, "Compressive sampling," *Proceedings of the Interational Congress of Mathematicians*, vol. Madrid, Spain, 2006.

[87] S. Kirolas, "Analog-to-information conversion via random demodulation," *IEEE Design, Applications, Integration and Software*, pp. 71–74, 2006.

[88] D. Healy and D. J. Brady, "Compression at the physical interface," *IEEE Signal Processing Magazine*, pp. 67–71, March 2008.

[89] R. G. Baraniuk, V. Cevher, M. F. Duarte, C. Hegde, and M. B. Wakin. (2014) Model-based compressive sensing. [Online]. Available: http://dsp.rice.edu/research/compressive-sensing/model-based

[90] R. G. Baraniuk, "Compressive sensing," *IEEE Signal Processing Magazine*, pp. 118–124, July 2007.

[91] J. A. Tropp, "Random filters for compressive sampling and reconstruction," *ICASSP 2006*, pp. 872–875, 2006.

[92] I. W. Group. International technology roadmap for semiconductors 2012. Last accessed on December 20, 2013. [Online]. Available: http://www.itrs.net/Links/2012ITRS/2012Chapters/2012Overview.pdf

[93] C. Peng, P. Sabharwal, and R. Bashiullah, "An adaptive neural spike detector with threshold-lock loop," *IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 2133–2136, May 2009.

[94] M. Scott, B. Boser, and K. Pister, "An ultra-low power ADC for distributed sensor networks," *Proceedings of the 28th European Solid-State Circuits Conference, ESSCIRC 2002*, pp. 255–258, September 2002.

[95] R. B. A.P. Chandrakasan, S. Sheng, "Low-power CMOS digital design," *IEEE Journal of Solid-State Circuits*, vol. 27, no. 4, pp. 473–484, April 1992.

[96] *nRF24LE1 Ultra-low Power Wireless System On-Chip Solution*, Nordic Semiconductor, August 2010.

[97] R. Dodd, B. F. Cockburn, and V. Gaudet, "Adaptive dual-threshold neural signal compression suitable for implantable recording," *Internation Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. (accepted)., p. 5, 2014.

[98] S. Mitra *et al.*, "24-channel dual-band wireless neural recorder with activity-dependent power consumption," *IEEE Internation Solid-State Circuits Conference (ISSCC)*, pp. 292–294, February 2013.

[99] E. Koutsos, S. E. Paraskevopoulou, and T. G. Constandinou, "A 1.5 uW NEO-based spike detector with adaptive-threshold for calibration-free multichannel neural interfaces," *IEEE Internation Symposium on Circuits and Systemsn (ISCAS)*, pp. 1922–1925, May 2013.

[100] Y. Yang, A. Kamboh, and A. Mason, "Adaptive threshold spike detection using stationary wavelet transform for neural recording implants," *IEEE Biomedical Circuits and Systems Conference (BioCAS)*, pp. 9–12, November 2010.

[101] H. Liao, M. K. Mandal, and B. F. Cockburn, "Efficient architectures for 1-D and 2-D lifting-based wavelet transforms," *IEEE Transactions on Signal Processing*, vol. 52, no. 5, pp. 1315–1326, May 2004.

[102] T. Denk and K. Parhi, "Architectures for lattice structure based orthonormal discrete wavelet transforms," *International Conference on Application Specific Array Processors*, pp. 259–270, 1994.

[103] C.-T. Huang, P.-C. Tseng, and L.-G. Chen, "VLSI architecture for discrete wavelet transform based on b-spline factorization," *IEEE Workshop on Signal Processing Systems (SIPS 2003)*, pp. 346–350, 2003.

[104] D. A. Parker and K. K. Parhi, "Low-area/power parallel FIR digital filter implementations," *Journal of VLSI Signal Processing*, vol. 17, no. 1, pp. 75–92, 1997.

[105] A. Rodriguez-Perez, J. Ruiz-Amaya, M. Delgado-Restituto, and A. Rodriguez-Vazquez, "A low-power programmable neural spike detection channel with embedded calibration and data compression," *IEEE Transaction on Biomedical Circuits and Systems*, vol. 6, no. 2, pp. 87–100, April 2012.

[106] S. Aghabozorgi, M. R. Saybani, and T. Y. Wah, "Incremental clustering of time-series by fuzzy clustering," *Journal of Information Science and Engineering*, vol. 28, pp. 671–688, 2012.

[107] R. Dodd, B. F. Cockburn, and V. Gaudet, "Neural spike compression using feature extraction and a fuzzy c-means codebook," *IEEE Symposium on multiple-valued logic (ISMVL '14)*, p. 5, 2014.

[108] A. M. Kamboh and A. J. Mason, "Computational efficient neural feature extraction for spike sorting in implantable high-density recording systems," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 21, no. 1, pp. 1–9, January 2013.

[109] R. Dodd *et al.*, "Microelectronics for in-vivo neural recording," *International Function Electrical Stimulation Society (IFESS) Conference 2012*, 2012.