ULTRA-LOW POWER ANALOG CIRCUITS FOR SPIKE FEATURE EXTRACTION
AND DETECTION FROM EXTRACELLULAR NEURAL RECORDINGS

By
CHRISTY LEIGH ROGERS

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2007
I dedicate this dissertation to my loving parents Greg and Regina and my loving fiancé, Xiao for all of their loving support as I worked on my research. My mom’s homemade frozen dinners and my dad’s dinner outing when he was near Gainesville for business kept me from losing too much weight while finishing my dissertation. And Xiao knows all of his specific encouragements. My entire family and church friends have been praying hard for me. Only with God’s strength and the support of family, friends, and my advisor, Dr. Harris, could I finish.
ACKNOWLEDGMENTS

The guidance from my undergraduate associate Dean Dr. Merckel and Professor Dr. Khoie helped me pursue a Ph.D. in the first place. Encouragement from my fiancé Xiao, my advisor Dr. Harris, and my parents, and friends Kwansun and Du have helped me stay the course even when I doubted wanting to finish my PhD. For that I am forever grateful.

I thank my advisor Dr. John G. Harris for the opportunity to work under him and learn so much about research and academia in general and specifically bioinspired circuitry. I am especially grateful for his encouragement to take the internship at Texas Instruments where I meet my fiancé and his guidance in my research as well as life such as encouragement to finish my degree before getting married. Even though I didn’t always listen so well, I at least got part of the message. I appreciate Dr. Harris’ understanding and, while at times pushing too hard, not giving up on me even when I was distracted from my research and my productivity was very lacking. Dr. Harris’ experience and knowledge from many years as a professor facilitated my studies during my PhD.

My committee Dr. Harris, Dr. Principe, Dr. Bashiruolah, Dr. Sanchez, and Dr. Ding as well as Dr. Fox, helped me though out my research by providing valuable feedback. Their insights, comments, and questions to ponder were very valuable to shaping this dissertation content. Their combined expertise across areas allowed each area of my research to be carefully examined.

I want to thank Dr. Sanchez and his Neuroprosthetics Research Group for all of their hard work in obtaining the neural recordings from training the rats and performing the implantation surgery, to setting up the entire experimental recording apparatus. Dr. Sanchez also marked the ground truths used to test our spike detection and feature extraction methods. Jack DiGiovanna, Dr. Sanchez’s Ph.D. student, assisted with the Tucker-Davis Technologies (TDT) coding for the chip test setup.

Many labmates have participated in discussion about my research and provided helping insight and/or questions but most particular Dr. Du Chen who graduated last
year provided much collaboration and insightfulness. Jie “Jessie” Xu did the layout for the leaky integrate-and-fire (LIF) chip and assisted in testing it. Also, as Dr. Harris moved me from 444 to 489 to distribute the English speakers I was infused with Chinese in 489 which helped me on the trip to China to meet my future in-laws. Specifically Xiaoxiang and her Chinese lesson recordings and Du have been instrumental in teaching me Chinese.

Last but not least I’d like to acknowledge my funding sources. My research was supported under a National Science Foundation (NSF) Graduate Research Fellowship, a University of Florida Presidential Fellowship, and a Defense Advanced Research Projects Agency (DARPA) sponsor grant #N66001-02-C-8022.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>4</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>9</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>10</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>13</td>
</tr>
<tr>
<td>CHAPTER</td>
<td></td>
</tr>
<tr>
<td>1 INTRODUCTION</td>
<td>14</td>
</tr>
<tr>
<td>1.1 Neural Signal Properties</td>
<td>15</td>
</tr>
<tr>
<td>1.2 Extracellular Neural Recording System Overview and Constraints</td>
<td>19</td>
</tr>
<tr>
<td>1.2.1 Electrodes</td>
<td>21</td>
</tr>
<tr>
<td>1.2.2 Amplifier</td>
<td>22</td>
</tr>
<tr>
<td>1.2.3 Wireless Data Transmission</td>
<td>24</td>
</tr>
<tr>
<td>1.2.4 Spike Sorting</td>
<td>24</td>
</tr>
<tr>
<td>1.3 Neural Data Reduction</td>
<td>27</td>
</tr>
<tr>
<td>1.3.1 Data Reduction for Spike Sorting</td>
<td>28</td>
</tr>
<tr>
<td>1.3.2 Data Reduction with Spike Detection</td>
<td>29</td>
</tr>
<tr>
<td>1.4 University of Florida’s Neural Recording Bandwidth Reduction Strategies</td>
<td>32</td>
</tr>
<tr>
<td>1.4.1 Biphasic Signal Coding with Reconstruction</td>
<td>32</td>
</tr>
<tr>
<td>1.4.2 Pulse-Based Feature Extraction</td>
<td>36</td>
</tr>
<tr>
<td>1.4.3 Spike Detection</td>
<td>38</td>
</tr>
<tr>
<td>1.5 Dissertation Structure</td>
<td>39</td>
</tr>
<tr>
<td>2 PULSE-BASED FEATURE EXTRACTION AND SPIKE SORTING: IMPLEMENTATION</td>
<td>40</td>
</tr>
<tr>
<td>2.1 Pulse-Based Feature Extraction</td>
<td>40</td>
</tr>
<tr>
<td>2.2 Data Reduction with Pulse-Based Feature Extractor</td>
<td>41</td>
</tr>
<tr>
<td>2.3 Spike Sorting with Pulse Trains</td>
<td>42</td>
</tr>
<tr>
<td>2.4 Matlab Simulations Results</td>
<td>47</td>
</tr>
<tr>
<td>2.4.1 Data</td>
<td>47</td>
</tr>
<tr>
<td>2.4.1.1 Neurosimulator data</td>
<td>47</td>
</tr>
<tr>
<td>2.4.1.2 Rat data</td>
<td>49</td>
</tr>
<tr>
<td>2.4.2 Spike2</td>
<td>50</td>
</tr>
<tr>
<td>2.4.3 Spike Sorting Results: Neurosimulator data</td>
<td>53</td>
</tr>
<tr>
<td>2.4.4 Rat data spike sorting results</td>
<td>53</td>
</tr>
<tr>
<td>2.4.5 Future Work</td>
<td>65</td>
</tr>
<tr>
<td>3 PULSE-BASED FEATURE EXTRACTOR AND SPIKE SORTING: IMPLEMENTATION</td>
<td>66</td>
</tr>
<tr>
<td>2 SOFTWARE FRONT-END AND BACK-END</td>
<td></td>
</tr>
</tbody>
</table>


## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-1</td>
<td>Feature extractors misclassifications compared to Spike2.</td>
<td>62</td>
</tr>
<tr>
<td>3-1</td>
<td>Spike sorting performance percent error.</td>
<td>70</td>
</tr>
<tr>
<td>3-2</td>
<td>Bandwidth reduction sorting error comparison</td>
<td>71</td>
</tr>
<tr>
<td>4-1</td>
<td>Spike sorting performance (percent error) from leaky integrate-and-fire (LIF) feature extraction chip</td>
<td>84</td>
</tr>
<tr>
<td>4-2</td>
<td>Bandwidth (pulses/s) from LIF feature extraction chip</td>
<td>84</td>
</tr>
<tr>
<td>4-3</td>
<td>Bandwidth reduction, power consumption, and sorting error comparison</td>
<td>85</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>1-1</td>
<td>Typical extracellular spike waveform with high signal to noise ratio (SNR).</td>
<td>16</td>
</tr>
<tr>
<td>1-2</td>
<td>Sketch of a neuron with the parts labelled.</td>
<td>16</td>
</tr>
<tr>
<td>1-3</td>
<td>Waveforms recorded from a linear silicon hexatode from a pyramidal cell with hypothesized position of the hexatode along the somatodendritic axis.</td>
<td>17</td>
</tr>
<tr>
<td>1-4</td>
<td>Block diagram of wireless front-end neural recording system.</td>
<td>19</td>
</tr>
<tr>
<td>1-5</td>
<td>Block diagram for four degrees of data reduction.</td>
<td>21</td>
</tr>
<tr>
<td>1-6</td>
<td>UF overall neural data reduction approaches.</td>
<td>33</td>
</tr>
<tr>
<td>1-7</td>
<td>An example of an input signal and its biphasic representation.</td>
<td>35</td>
</tr>
<tr>
<td>1-8</td>
<td>Block diagram of biphasic encoding with integrate-and-fire (IF) neuron.</td>
<td>36</td>
</tr>
<tr>
<td>1-9</td>
<td>Block diagram of biphasic encoding with leaky integrate-and-fire (LIF) neuron.</td>
<td>37</td>
</tr>
<tr>
<td>1-10</td>
<td>Feature extraction and subsequent sorting implementation schemes.</td>
<td>38</td>
</tr>
<tr>
<td>2-1</td>
<td>Block diagram of biphasic encoding.</td>
<td>42</td>
</tr>
<tr>
<td>2-2</td>
<td>Neural simulator spike signatures from six different neurons at two different time periods.</td>
<td>44</td>
</tr>
<tr>
<td>2-3</td>
<td>Neural simulator spike signatures convolved with a Gaussian to determine the distance between it and the templates.</td>
<td>45</td>
</tr>
<tr>
<td>2-4</td>
<td>Spike sorting error as a function of the pulse train distance Gaussian ( \sigma ).</td>
<td>46</td>
</tr>
<tr>
<td>2-5</td>
<td>Neural simulator signal with all six neurons and the biphasic pulse train output from the leaky integrate-and-fire (LIF) circuit for a bandwidth of 455 pulses/s.</td>
<td>48</td>
</tr>
<tr>
<td>2-6</td>
<td>Neural waveform recorded from rat003. Column two is zoomed in from column one.</td>
<td>50</td>
</tr>
<tr>
<td>2-7</td>
<td>Spike2’s templates for neurosimulator data.</td>
<td>51</td>
</tr>
<tr>
<td>2-8</td>
<td>Spike2’s principal component analysis (PCA) for neurosimulator data.</td>
<td>52</td>
</tr>
<tr>
<td>2-9</td>
<td>Spike2 example template for rat data.</td>
<td>55</td>
</tr>
<tr>
<td>2-10</td>
<td>Spike2’s PCA analysis for rat data.</td>
<td>56</td>
</tr>
<tr>
<td>2-11</td>
<td>Spike2’s histogram analysis for rat data.</td>
<td>57</td>
</tr>
<tr>
<td>2-12</td>
<td>Spike2’s templates.</td>
<td>58</td>
</tr>
</tbody>
</table>
2-13 Feature extractor templates. .................................................. 59
2-14 Pile plots of Spike2’s sorted spikes. ........................................ 60
2-15 Pile plot of feature extractor’s correctly sorted spikes referenced to Spike2’s results. 61
2-16 Pile plot of feature extractor’s correctly sorted spikes with those misclassified as that neuron overlaid with a dashed black line. ........................ 63
2-17 Pile plot of feature extractor’s correctly classified spikes overlaid with the spikes from that class but misclassified as another class in that classes color ......... 64
3-1 Bandwidth (pulses/s) changes for threshold and leakage values parameters and integration capacitor at 10 pF. ................................................................. 67
3-2 Spike sorting error changes for threshold and leakage values parameters ....... 68
3-3 Spike sorting error as a function of bandwidth for three SNRs. ................. 69
4-1 Leaky integrate-and-fire (LIF) circuit. ........................................ 74
4-2 Voltage to current convertor circuit for LIF. ................................ 75
4-3 Operational transconductance amplifier OTA for voltage to current convertor circuit for LIF. ................................................................. 76
4-4 Comparator circuit for LIF. ....................................................... 77
4-5 Reset and refractory period circuit for LIF. .................................. 78
4-6 Leaky circuit for LIF implemented as a $G_m$ current source. ............ 79
4-7 Layout for the LIF circuit. ......................................................... 80
4-8 Initial chip test setup: Compare UF’s feature extractor with Dr. Sanchez’s Spike2 results using neural simulator (487 NEB). ............................... 81
4-9 Intermediate chip test setup: Set UF’s feature extractor chip parameters using prerecorded data from rat that will be used in in vivo experiments (487 NEB). 82
4-10 Final test setup: Compare UF’s analog amp and feature extractor with TDT’s amp and Dr. Sanchez’s Spike2 results (rat lab). First setup with neural simulator then when working use rat. ................................................................. 83
5-1 Single-scale detector block diagram with snapshots of the waveform after each block. ................................................................. 87
5-2 Data used for simulations. ....................................................... 88
5-3 Plot of receiver operating characteristic (ROC) curves, 0dB SNR. ........... 89
5-4 Single-scale spike detector circuit diagram. ................................ 92
Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

ULTRA-LOW POWER ANALOG CIRCUITS FOR SPIKE FEATURE EXTRACTION
AND DETECTION FROM EXTRACELLULAR NEURAL RECORDINGS

By

Christy Leigh Rogers

May 2007

Chair: John G. Harris
Major: Electrical and Computer Engineering

The purpose of this dissertation is to investigate an ultra-low power implant for spike
detection and spike feature extraction in neural recording applications to dramatically
reduce the required communication bandwidth through the skin. Implanted systems
impose four major constraints: low power consumption, small size, robustness, and limited
bandwidth. The developed solution is two fold. For applications which do not require
spike sorting a lower power and lower bandwidth solution exists. A novel multi-scale
continuous wavelet approach is used to decompose the signal into several frequency bands
to allow for individual thresholds at each band to more accurately detect the presence of
a spike. For applications which require spike sorting, a spike feature extraction algorithm
was developed to extract information about the spikes so bandwidth is not wasted
transmitting information not relevant to spike sorting. The feature extractor’s bandwidth
reduction was designed with a system level view to optimize the back-end spike sorting
while using minimal bandwidth. Analog very large scale integration (VLSI) circuitry was
chosen to implement both the spike detection algorithm and feature extraction algorithm
to allow for an ultra-low power and compact solution for the integration of many channels
in an implanted device. Preliminary theoretical analysis and chip measurement results
show suitability for in vivo neural recording applications for both algorithms.
CHAPTER 1
INTRODUCTION

The neuron is the basic information processing unit in the brain. Neurons use electrical pulses, called action potentials or spikes, to transmit information. Their extracellular electrical pulses can be recorded using microelectrodes that are implanted into the brain. These recordings are called neural recordings since they record the voltage potential caused by neurons. To best study neural information processing, many neurons must be simultaneously recorded in awake behaving subjects. State-of-the-art recording systems require microelectrode arrays with hundreds of electrodes implanted into the brain.

While much is still unknown about the brain, researchers have now learned enough to integrate neural prostheses with the brain [1]. One example of a neuroprosthetic is a Brain-Machine Interface (BMI) [2]. Motor-based BMIs extract information from neural recordings collected in the motor, premotor and parietal cortices with the goal of creating predictive models for the subject’s intent of motor movement to directly control a robotic device. Eventually, these devices could allow paraplegics to control a robotic arm to feed themselves or turn the pages of a book. Neural prosthetics require long-term neural recordings which necessitate wirelessly transmitting the data from the electrodes through the skin. If a wire from the electrode passed through the skin to send the data, infection is risked and the subject tethered. Also, having many wires coming out of the head and being tethered restricts movement of the subject.

Current instrumentation technology and surgical procedures allow for the simultaneous recording of hundreds of electrodes. The bottleneck is how to transfer the large bandwidth raw data streams wirelessly. Transmitting the raw voltage signals from hundreds of channels is not possible with the current wireless bandwidth limits. Furthermore, even if these high data rates could be met, the power dissipation of the electronic circuitry would severely drain the implanted power supply and exceed the power dissipation limits for preventing tissue damage.
The presented research addresses this bottleneck with two approaches to bandwidth reduction based on if spike sorting is required or not. A novel analog spike detection circuit to only transmit spikes times dramatically reduces the required transmission bandwidth for applications which do not require spike sorting. The detector is lower in power and more compact than existing spike detection methods without compromising performance. For applications which require spike sorting a feature extraction method was developed which still dramatically reduces bandwidth compared to current instrumentation but it does require more bandwidth than only transmitting spike times with the spike detector. The feature extractor is also lower in power and more compact than existing methods for data reduction which allow for spike sorting without compromising performance.

1.1 Neural Signal Properties

One of the most widely recorded neural signals is the extracellular biopotential generated electrochemically from individual neurons. After a neuron receives sufficient stimuli from other neurons, its cell membrane depolarizes which causes extracellular ionic currents to flow. This in turn causes voltage sensitive channels to open allowing ions (Na+ and K+) to pass through the neuron’s membrane. The result is a change in the extracellular single-unit potential in the shape of a spike (also commonly referred to as an action potential) with a peak-to-peak amplitude of 50 µV–500 µV when measured extracellularly [3]. The amplitude varies inversely with the distance between the electrode and the neuron. A typical extracellular spike waveform from a high signal to noise ratio (SNR) neural recording from a rat is shown in Figure 1-1. The frequency content of spikes is mainly between 100 Hz and 6 KHz [4] with widths varying between 0.4 ms and 3 ms [5, 6] depending on how far the electrode is from the neuron and what part of the neuron is closest to the electrode. A labelled drawing of the neuron is shown in Figure 1-2. As the spike propagates from the soma (cell body) along the axon, the spike amplitude becomes attenuated and the width increases [7] as shown in Figure 1-3.
Figure 1-1. Typical extracellular spike waveform with high signal to noise ration (SNR).

Figure 1-2. Sketch of a neuron with the parts labelled.
Figure 1-3. Waveforms recorded from a linear silicon hexatrobe from a pyramidal cell with hypothesized position of the hexatrobe along the somatodendritic axis to the right. Note variation in wave shape along the somatodendritic axis adapted from Harris et. al. [7].
One electrode may record from as many as four to six neurons but there will be many more distant neurons whose signals become part of the noise on the signal. Typical recorded noise levels, which include distant neurons and the electrode’s thermal noise, are around 20 $\mu$Vrms $^8$. It is difficult to define SNR for a neural waveform because the signal (the spike) has varying amplitudes and widths. Spikes with a larger amplitude and width will have a higher SNR than smaller amplitude and narrower spikes from the same channel with the same level of noise. This alone would suggest SNR should be defined for each neuron within a recorded waveform. However, the signal is also transient and non-stationary so the amplitude and width of spikes from a single neuron also change over time. Therefore, SNR is often defined using the average of all the spikes in the waveform with typical SNR ranges between 0 dB to 12 dB $^9$.

Single neurons deplete their chemical reservoirs after producing a spike, inactivating the Na channels which can not reopen until the membrane potential returns to a negative value near the threshold. This sets a minimum time required to replenish their reservoirs before the neuron can spike again. This period is called the refractory period and is typically around 1 ms $^{10}$. Because extracellular neural recordings may contain the response of more than one neuron, spikes from different neurons could fire very close together, within the refractory period of a single neuron. Spikes from different neurons can even overlap and create a superimposed waveform.

Many electrodes are often used to record neurons from multiple sites within the brain. A collective group of implanted electrodes are referred to as an electrode array and is encased in cranioplast that coats a screw anchored into the skull. This physically keeps the electrodes from moving in reference to the skull. The brain however is itself floating in cerebrospinal fluid (CSF) and can shift by many microns over time. As the brain moves, the distance between the electrode and the neurons it is recording from also change so the spike shape (amplitude and width) can change over time as the brain shifts. This results in SNR fluctuations in the signal. To make matter worse, unavoidable electrochemical
effects at the electrode-tissue interface introduce DC offsets ranging from 1–2 V across the recording sites [11].

1.2 Extracellular Neural Recording System Overview and Constraints

Neural recording systems usually consist of an analog front-end near the recording site, a means to get the data off the subject, and an external processing element. The analog front-end consists of electrodes and amplifiers. Electrodes are intrusively placed in the brain to measure the neural potentials. This signal is in the µV range so it must be amplified close to the recording site so noise does not corrupt the small signal. The amplification also allows for improved processing later in the system. Data reduction is performed and then, a wired or a wireless telemetry unit transmits the data away from the subject for more intense processing on the back-end where power and size constraints are less stringent. A block diagram of a wireless front end and back-end system is shown in Figure 1-4.
To continuously record from awake subjects the electrodes must be implanted and the signal wirelessly sent out of the body because of the risk of infection if wires were to pass through the skin. Even in current animal neural recordings that use wires from the electrodes out of the skin there are two problems in addition to infection. The animal must be tethered, which restricts its natural movements, and the wires must be harnessed in a fashion so they do not become entangled. Often commutators are used to prevent tangling and torque applied to the prosthetic from movement of the wires but this limits on the number of wires.

In addition to the electrodes several pieces of neural instrumentation electronics, such as an amplifier, data reducer, and circuitry to transmit the data, should also be implanted to avoid any wires passing through the skin. Low power circuitry is necessary due to the difficulty of charging or changing implanted batteries as well as to prevent tissue damage. Power can be broadcast into the device, but studies have shown that if the brain tissue increases in temperature any more than about 1°C the brain tissue will be damaged [12]. Power dissipation over 80 $mW/cm^2$ has been reported to cause general tissue damage [13]. Circuit area must also be small due to the limited area of implantation between the subjects’ skull and skin especially on small animals.

BMI systems place strong constraints on the wireless transmission because hundreds of channels are currently recorded with the desire to reach thousands in the future. Transmitting raw voltages from 100 channels at a 25 KHz sampling rate and 8-bits of resolution will generate data rates around 20Mbps. Furthermore, even if these high data rates could be met, the power dissipation of the electronic circuitry would severely drain the implanted power supply and exceed the power dissipation limits for preventing tissue damage. Therefore, data reduction is required. Because spikes are sparse within neural recordings, only transmitting information about the spikes provides further significant reduction in the required transmission bandwidth compared to sampling and quantizing the entire signal. After the spikes are detected, there are three major
options for bandwidth reduction that depend on the application. This results in a total of four data reduction techniques. In the order of their data reduction, with all but the first requiring spike detection, the approaches are: Approach 1: sample and quantize the raw data using conventional techniques, Approach 2: only transmit a clip of the waveform around the spike, Approach 3: send features needed for spike sorting, Approach 4: only transmit the spike times. The four data reduction schemes are depicted in Figure 1-5.

1.2.1 Electrodes

Electrodes are the first stage of the system so their performance directly impacts the rest of the system. The better the neural recording SNR is from the electrodes, the more accurate the rest of the system performs. Therefore, it is crucial that the electrode properties allow for the most effective recordings. The electrode must be biologically compatible so it can continue recording over time as opposed to being encapsulated with Glia protein, blocking its recording ability [1]. Electrode material, size, tip sharpness, and pliability should be carefully selected to minimize noise and tissue damage while maintaining the ability for precise implantation. The details on specific tradeoffs in electrode design is not presented here but one can refer to the literature [8, 14, 15].
The total input referred noise of the neural recording system must be significantly smaller than the smallest amplitude neural signal of interest. The peak-to-peak amplitude of neural signals can be as low as $50 \mu V$, so it is important that the total input noise be below about $20 \mu V$ [9]. The electrode contribution to the total noise of the system is mainly comprised of neural background noise and thermal noise. Neural background noise is the summation of all the distant neurons’ electrical potentials. The thermal noise occurs at the metal-electrolyte interface and is related to electrode impedance and the recording bandwidth, which has a $1/f$ frequency dependence [15].

There are two major categories of electrodes: passive and active. Passive electrodes do not contain any interfacing electronic circuitry on the electrode substrate [16] and are usually made of metal [8, 17, 18] or glass [16]. Active electrodes include electronic circuitry on the same substrate as the recording electrode [16]. The on-chip circuitry can minimize the number of leads on the chip as well as minimize the leakage and noise associated with sending a very small signal ($\mu V$) over wires. In 1975, Wise at the University of Michigan was the first to produce an active electrode [19] and advances are still being made [16, 20].

### Amplifier

Because the neural signal has such a low amplitude, it must be amplified prior to further processing. The amplifier must have very low noise, filter the signal, and be fully integrated and low-power to allow for implantation. Frequencies outside the neural spike range, between 100 Hz and 6 KHz [4], should be removed from the signal to reduce the noise. If a clever scheme is not used to filter out the low frequency noise, a large off-chip capacitor will be required making the circuitry too large to implant.

Ji et al. have reported one type of Complementary metal-oxidesemiconductor (CMOS) amplifier which shares the same substrate with silicon electrodes [20]. The amplifier provides a mid-band gain of 51 dB without amplifying the DC frequency components. However, one major issue with this design is the gain variability from probe to probe or
even from channel to channel on the same probe. Gains also drift during probe use and with ambient light levels, which will eventually saturate the amplifier.

Najafi and Wise were one of the earliest teams to reduce the random DC component at the electrode-electrolyte interface with a reverse-biased diode to clamp the input with the high resistance of the junction depletion region [4]. Ji et al. and Akin et al. employed an internal bandlimiting method by using diode-capacitor filters to form the low cut-off frequency [20, 21]. This scheme suffers from several issues: optical drift that reduces reliability, limited dynamic range, and high variability of the lower cut-off frequency [22]. Dagtekin et al. reported a multi-channel chopper-modulated neural recording amplifier that uses the chopper modulation technique and an unbiased location in the system as a reference to minimize the effects of the DC drift of the neural signals [23]. No chip performance metrics were ever published, only simulated data.

Chandran et al. employ a sub-threshold NMOS transistor as a high value shunt resistor to attenuate the DC offset to stabilize the DC baseline [24]. This resistor creates the lower cut-off frequency of the amplifier. Up to 400 mV of DC input can be handled without sacrificing the AC performance. However, the amplifier is unable to reject negative DC input values. Moheseni et al. modified Chandran’s design by employing a subthreshold PMOS transistor to reject both negative DC and positive DC input [22]. A laser trimmed resistor is used to accurately set the lower cut-off frequency. This design can not tolerate a DC shift higher than 400 mV. The drawback is the extra process step required for laser trimming.

Chen and Harris from the University of Florida have used clever circuitry based on Harrison’s design [11] to develop a low-noise ultra-low power fully integrated neural amplifier (bioamplifier) to meet the requirements of an implantable device. The UF bioamplifier provides a gain of almost 40 dB, input referred noise of only 9.56 $\mu V_{rms}$, a CMRR of about 59 dB, and a power supply rejection ration (PSRR) of about 45 dB [25]. It has a low cutoff frequency of 0.3 Hz and a high cutoff frequency of 5.4 KHz. These
specifications allow the amplifier to perform without too much added noise which would corrupt the neural signal.

1.2.3 Wireless Data Transmission

Many neural prosthetic groups are working on wireless telemetry systems to transmit data out of the brain and send power to the implanted circuitry. The difficult part is that the circuitry on the subject’s end must be implanted necessitating small area and small power consumption. There are two major schemes that groups have used to transmit the data wirelessly: AM (amplitude modulation) and FM (frequency modulation). Akin et al. \[21\] report a system where the digitized signal is encoded into an 8-bit pulse-position modulation (PPM), pulse-code modulation (PCM) and transmitted with AM. The system dissipates 2.8 mW of power and its die area is 0.7 mm$^2$ per channel. Huang et al. \[26\] use a 9-bit PPM to encode the data and transmit it using FM. The core area is 3.6 mm $\times$ 4.350 mm for one channel. Both of these systems utilize inductively coupled radio frequency (RF) telemetry for both the power and the data transfer. The analog-to-digital convertors (ADCs) and modulators expand the die area and power consumption considerably so they must be carefully designed to allow small area and low power consumption necessary for implantation.

1.2.4 Spike Sorting

For in vivo extracellular neural recordings, multiple neurons are recorded from the same channel and the spike shape can be used to distinguish amongst the individual neuron signals through a process called spike sorting. Popular spike sorting methods are too computationally intensive for implanted devices. Thus, the spike sorter is typically pushed to the back-end where power and area constraints are less stringent.

Neuroscientists rely on a variety of spike sorting methods utilizing different features of the spikes with no community wide agreement as to which spike sorter is the best. Most researchers simply use the default spike sorting software that comes with the neural instrumentation hardware. Tradeoffs exist between performance and the ability to run
the spike sorter realtime. Also, spike sorting performance is often directly related to
the amount of time the neuroscientist spends to setup the spike sorting parameters.
Spike sorting is a classical problem in the neuroscience community, with many proposed
algorithms in the literature. A overview of the major spike sorting methods is provided
below. For a more in depth review of spike sorting algorithms consult Lewicki [27].

There are several major categories of spike sorters: template matching, clustering
approaches, independent component analysis (ICA), neural networks, and simple threshold
based methods. An overview of each group of spike sorters with proper references to the
literature follow. Also, some groups use combinations of the above methods to refine
the classes through the use of different features in an effort to improve spike sorting
performance.

Template matching does not require any feature extraction as an average waveshape
for the spikes from each neuron is used. A neuroscientist determine the number of distinct
neurons as well as the templates. Tools such as principal component analysis (PCA) and
histograms can be used to see examine the templates. Plotting the first 2-3 principle
components shows how much separation exists between classes and gives the user a visual
aid to see how many distinct classes exist. Histograms show if the firing rate of each
neuron violates fundamental limits meaning that class contain spikes from more than one
neuron and the templates need to be reformed. Often template matching is performed by
using a matched filter to find which template most closely matches each spike to classify
it. A threshold is used so a waveshape is classified as noise if it does not closely match any
of the templates.

When feature extraction is used it is often followed by cluster based spike sorting
to determine the neuron for each spike. By plotting each feature on its own axis a
multi-dimensional graph is formed where any clustering algorithm can be used to classify
the data. Popular features are the spike amplitude and width, principal components,
wavelet coefficients [28], and slope of rise and fall time. The most popular feature
extraction method is PCA \[29\] because it can be used to extract a compact set of orthogonal features. Some popular options for clustering are the simple k-means or nearest neighbor where each cluster location is marked as the mean of the data within the cluster. A spike is then classified to the cluster with the closest mean Euclidean distance. More elaborate methods, such as Bayesian clustering, use statistical information about the neurons and their spike shapes and are best suited when the clusters have significant overlap or differ from a spherical distribution. Many more clustering algorithms exist and are surveyed in books such as the classic one by Duda, Hart and Stork \[30\].

ICA is a special case of blind source separation. It separates a multivariate signal into additive subcomponents. It assumes the number of electrodes equals the number of sources which only approaches the truth for a large number of electrodes. It also assumes sources are mixed linearly. ICA is used in tetrode recording as usually the number of signals is closer to the number of neurons compared to single site electrode recordings \[31\].

Some groups have applied neural networks to solve the feature extraction problem but another method must be used to sort the data to provide the ground truths to train the neural network \[32\]. Thus, the neural network based spike sorter can only be as good as the spike sorter used to provide ground truths to train the neural network.

Threshold-based spike sorters are the simplest spike sorters and often precede more complicated spike sorters as a spike detector. The thresholds can be simple voltage thresholds (positive and negative) along with some rules such as the waveform must pass through two thresholds within a certain time (a hoop). This then forms a voltage-time threshold. One might also impose that the spike must first go positive and then negative. To differentiate between spikes, different threshold rules are applied to differentiate spikes by their amplitude, width, and/or rise or fall time \[27\]. Some on-line spike sorters, such as Tucker-Davis Technolgies’ (TDT’s) SortSpike2 \[33\], use this threshold-based spike sorter as a spike detection step to separate spike waveforms from the raw data.
Currently real time spike sorters ignore the possibility of overlapping spikes due to the computational complexity of overlap resolution methods. Template matching is the only method to address overlap and it uses a subtraction method to separate out the overlapped spikes \[27\].

All of the mentioned spike sorters require human-tuned parameters which affect the accuracy of spike sorting. Generally less than three features are used to spike sort so neuroscientists can view the feature clusters and properly set the parameters. A few attempts at fully automatic spike sorting have been made but neuroscientists have not embraced them as they increase sorting error. The human tuned parameters introduce a variability in spike sorting across different neuroscientists.

A study on the variability of manual spike sorting using human-configured on-line sorting algorithms by Wood, Black, Varghese-Irwin, Fellows, and Donoghue \[34\] showed a wide variability in the number of neurons and spike detected in real data. The number of spikes varied four fold and the number of neurons was only correct 25% of the time. To obtain specific error values, synthetic data was used so the ground truths were known. Average error rates of 23% false positives and 30% false negatives were obtained with the synthetic data. This variance and error in current spike sorting methods makes it difficult to compare spike sorters with real data as accurate ground truths are not known. The best approach currently available is for many expert spike sorters to use off-line spike sorters to mark a data set and an average taken.

### 1.3 Neural Data Reduction

Transmitting the raw voltage signals from hundreds of channels is not possible with the current wireless bandwidth limits and thus data reduction is necessary. As neuroscientists do not want to limit the number of recorded electrodes, the data reduction must limit the data sent in each electrode to decrease the overall required bandwidth. The reduction in information sent must preserve the pertinent information.
Most neurons encode information in spikes, with the exception of retinal neurons. It is debated as to whether the information is encoded in the rate of spike firings or with individual spike times, but either way only recording spike times for a single neuron is the ultimate data reducer and either encoding mechanism can still be evaluated. For single spiking neurons, signal information is encoded in spike times and not the amplitude or shape of the spike. However for in vivo extracellular neural recordings, multiple neurons are recorded from the same channel so spike sorting may be necessary thus requiring relevant features to be preserved.

BMI systems currently use a human tuned and computationally intensive spike sorting process, which recovers several individual neural signals from each electrode at the cost of additional power consumption and increased system size which necessitates performing spike sorting outside the body. This results in a larger communication bandwidth because windows of data around possible spikes or features from those windows must be transmitted compared to only sending out spike times. Recent results suggest that the spike sorting step may possibly be eliminated without severe degradation of BMI performance \[2, 35\] thus lending credibility to solely transmitting spike time for data reduction for some applications such a low precision BMIs (approach 4 in Figure 1-5). Details are given in Section 1.3.1. However this claim has not been exhaustively tested and neuroscientists will continue to require spike sorting for studying the brain. For cases where spike sorting is necessary there are data reduction schemes that retain more information than solely transmitting spike times but at the expense of higher data bandwidths (approaches 1, 2, and 3 in Figure 1-5 and Section 1.3.1).

1.3.1 Data Reduction for Spike Sorting

If spike sorting is required, either features of the spikes (approach 2 in Figure 1-5) or windows around the spikes (approach 3 in Figure 1-5) can be transmitted while still obtaining significant data reduction compared to transmitting the sampled and quantized raw waveforms (approach 1 in Figure 1-5). Of course there is a tradeoff between data
reduction and the amount of information retained. Neural signal data reduction is a general issue with a variety of solutions being pursued. Popular data reduction methods which allow for spike sorting include spike detection followed by different options to reduce the data.

One option is to wirelessly transmit a sampled and digitized clip of the raw waveform surrounding the spike for spike sorting outside the subject where power and size constraints are less stringent [9]. This allows for data reduction while retaining the use of traditional spike sorting methods on the back-end. The main drawback is currently complex digital VLSI circuitry is used to store the waveform until spike detection is performed and perform spike detection. The power requirement of the VLSI circuitry is currently prohibitive for implantation.

Another option is to extract and send the features themselves for spike sorting [36], but how to get the features out wirelessly at low-power is problematic as they need to be quantized and sent in a group with all the features from one spike along with the spike time. Currently no group has yet to solve this issue.

1.3.2 Data Reduction with Spike Detection

The ultimate data compression scheme for neural recordings is spike detection, where only spike times or binned spike counts are transmitted ([37] approach 4 in Figure 1-5). This greatly reduces the bandwidth required to transmit the neural signals because spike occurrences are sparse within neural data. Spike detection is also the first step in most of the data reduction methods which preserve enough information for spike sorting as shown in Figure 1-5.

Spike detection must be as accurate as possible because missed detection errors propagate through the system as missed information (missed neural spike). False detections also propagate through the system as incorrect additional information (false neural spikes) unless windows of data around the spikes are transmitted. Then, a spike sorting process can allow some of the windows to be classified as noise to reduce the false
detection error but additional bandwidth to transmit the false alarms is still required. Therefore, systems that transmit windows of data around the spike often use lower thresholds to increase detection performance, but the lower the threshold the higher the power dissipation and required bandwidth which can cause a bottleneck in wireless data transmission so care must be taken when selecting the thresholds. One solution is to monitor the bandwidth and adjust the thresholds to maximize its utilization while remaining within the power limits.

Spike detection is a long standing issue in neuroscience. Popular spike detection methods include amplitude thresholding, absolute value, energy based, wavelets, matched filters, and template matching. Currently, there is no consensus in the community as to the best approach to spike detection, particularly for robust, unsupervised, and computationally simple methods. Each of the proposed detection techniques have shortcomings for implanted applications.

Amplitude thresholding is the simplest and lowest in power spike detector since it is a subset of the other methods for this binary detection problem. It is the easiest to use with only one parameter to set, the threshold level, and it is the most common spike detection tool used though it is often paired with additional processing to achieve acceptable detection performance. For instance, it can be paired with requirements that the signal pass through two thresholds (sometimes called a hoop) within a certain amount of time to increase its performance. This can insure the slope is sufficient as it passes the thresholds or that the threshold crossing is spike like in that the signal rises to cross the threshold and then falls to cross the threshold again within a certain period of time. The amplitude threshold detector’s performance quickly begins to fail as SNR drops though and it is not robust to DC drift [27].

The absolute value of the signal can be used to allow for an equal detection rate of spikes with larger positive or negative amplitudes. The tradeoff is the detector must be blinded after a detection so that spikes with more than one phase are not detected
twice. The blinding period limits the detector’s resolution between two spikes because it prohibits two spikes from being detected unless they are farther apart than the blinding period. Obeid and Wolf have reported that though the absolute value spike detection method does not perform the best among spike detectors it is very cost effective for their setup in terms of computation cycles, performance, and required bandwidth to transmit the data [9].

Several groups have tried to use non-linear energy operators (NEO) to detect spikes [38–40] but they are too sensitive to noise, so they only perform well for extremely high SNRs. Recently a new class of multiresolution TEOs (Teager Energy Operator, a type of NEO) was presented with noticeable improvement over previous NEO’s for low SNR data [41]. The multiresolution approach allows the detector to impose additional constraints on the energy function to consider it a spike, which keeps the detector from being as sensitive to noise. Compared to the previous energy detectors this method requires more computation since, for multiresolution, it requires several TEO’s to be evaluated in parallel and then combined to make a final decision.

Recently wavelets have become popular because they allow for localization in both the frequency and time domain which is important for transient and non-stationary signals such as neural data. Several groups have developed off-line spike detectors based on wavelets [42, 43]. With the increase in PC computational power, algorithms have been designed for wavelet decomposition to run in almost real-time [44]. Applying near real-time wavelet algorithms to spike detection has resulted in better performance than existing single scale methods but current wavelet circuits consume too much power for implantation.

The most complex and possibly one of the more accurate spike detection methods is matched filters or template matching. If the signal was embedded in white noise, the matched filter would maximize the SNR and be the most accurate spike detector, but distant neurons, correlated with the signal, creating non-white noise [45, 46]. MATCHED
filters require stable spike template(s) obtained from a human expert selecting the spikes in the waveform and taking an average of them to use as a template. For spike sorting a separate template is required for each neuron and with spike detection a single average template of all spikes can be used. The template is convolved with the signal and the result is thresholded to determine the presence of a spike since signals similar in shape to the template will produce a large response while the noise should produce a much smaller response.

The problem with matched filters is that the filtering circuitry requires too much power to be implanted and the spike templates tend to be unstable. An option is to use a simple spike detector (such as amplitude threshold), with the threshold set low as to not miss many spikes, as a preprocessor to parse out windows of data around possible spikes to transmit. Then, outside the body, where more power is available, the windowed data could be compared to the template and if the window differed too much the waveform could be disregarded as noise. While a lower threshold increases the final detection performance, the drawbacks are an increase in transmission bandwidth from sending out more noise waveform clips to reduce missed spikes.

1.4 University of Florida’s Neural Recording Bandwidth Reduction Strategies

The University of Florida has three approaches for neural bandwidth reduction for different application needs but all three can be implemented in low-power analog VLSI circuitry for implantation. The three different approaches are shown in order of decreasing bandwidth in Figure 1-6. Neural signals have a rather sparse number of neural spikes with the rest of the signal noise. As the noise is not important, it would be best to use the bandwidth on the spike portions of the signal and all three methods take advantage of this.

1.4.1 Biphasic Signal Coding with Reconstruction

The first approach the University of Florida (UF) has chosen to reduce the data bandwidth is to encode the data to reduce bandwidth so that in theory it can be perfectly
Figure 1-6. UF overall neural data reduction approaches.
reconstructed on the back-end and traditional spike sorting techniques can be applied. This approach was developed by Dr. Chen in her Ph.D. studies at the University of Florida, thus only a brief overview will be provided here but the reader can refer to her dissertation [47] and conference paper [48] for additional information.

This encoding is done using a biphasic-pulse representation because pulses are digital which are more robust to noise than analog in wireless transmission yet it is lower power (100 µW) than digital because it does not require an ADCs. The bandwidth can be reduced by more than four times over a traditional ADC sampled system at 25 KHz with 12-bits of resolution.

The biphasic signal encoding uses pulses to represent when the integral of the waveform (its area) surpasses a positive or negative threshold. An example of an input signal and it’s biphasic representation is shown in Figure 1-7. This automatically increases the bandwidth during spikes as the area is larger and reduces it during noise as the area is smaller. By setting the area threshold appropriately you can theoretically obtain perfect reconstruction [48]. In practice, only the spike portions need to be reconstructed close to perfect so the area threshold could be set only with consideration to perfectly reconstruct the spike portions of the signal.

A block diagram for the biphasic encoding system is shown in Figure 1-8. If the output of the integrator, \( y(t) \) reaches the positive threshold of the comparator, \( \theta \), the output of the comparator raises and resets the integrator after a short delay, \( \tau \), in the feedback loop. Similarly, if the output of the integrator \( y(t) \), reaches the negative threshold, \(-\theta\), the output of the comparator drops and also resets the integrator. The delay, not shown in the simplified block diagram, sets a maximum bandwidth and with more strict constraints it still allows for theoretical perfect reconstruction [47]. The timing of two consecutive pulses must satisfy the following equation:

\[
\frac{1}{C} \int_{t_i+\tau}^{t_{i+1}} x(\Delta)d\Delta = \theta_i
\]  

\( (1-1) \)
Figure 1-7. An example of an input signal and it’s biphasic representation. A) Shows the input signal. B) Shows the biphasic representation.
where $\theta_i \in \{-\theta, \theta\}$.

1.4.2 Pulse-Based Feature Extraction

In this work, UF has started to pursue the second approach, a spike feature extraction method that reduces the required bandwidth even further. Most people optimize the front-end for data reduction but we prefer to optimize the complete system by considering the bandwidth reduction effect on spike sporting.

This spike feature extraction method also uses biphasic pulses to encode the data, but it only preserves features about the spike and little information about the noise by not following the strict constraints for reconstruction. Also, a leaky term, shown as a resistor in Figure 1-9, is added to allow a greater reduction in bandwidth by subtracting out the noise as well as providing synchronization (first pulses do not depend on the previous samples (noise)) for the pulse-train output at the time of the spike.

It allows spike sorting to be directly performed on data that is wirelessly transmitted reducing the complexity on the back end. The spike sorting uses a traditional method of template matching but is untraditional because the waveform is pulse trains. The feature
Figure 1-9. Block diagram of biphasic encoding with leaky integrate-and-fire (LIF) neuron.

extractor can reduce the bandwidth more than one-order of magnitude lower than the
UF biphasic encoding and more than two-orders of magnitude lower than traditional
ADC sampled data at 25 KHz with 12-bits of resolution while maintaining a similar spike
sorting error.

This data reduction approach is divided into four schemes based on implementation
as illustrated in Figure 1-10. All of the schemes use a pulse-train based spike sorter in
software on a PC that was designed specifically for the feature extraction algorithm.
The first implementation scheme uses existing front end data reduction techniques
such as using and ADC and replaces the back end spike sorting software with its
feature extraction algorithm and spike sorter algorithm implemented on a PC platform.
Implementation scheme two takes advantage of the feature extractors bandwidth reduction
by placing the algorithm on a digital signal processor (DSP) in in the front-end and
using the same sorter as in approach one. Implementation scheme three is a hybrid
approach and it implements the front-end feature extractor in analog to use it’s low power
advantage with the same back-end spike sorter. Implementation scheme four is purely
analog both in the front-end with the feature extractor and the back-end with the spike
sorter.
Implementation scheme 1–3 will be presented in detail in Chapters 2, 3 and 4. Implementation scheme 4 will not be presented in detail as power savings with an analog back-end is not currently necessary because the power limits are much greater than in the front-end.

1.4.3 Spike Detection

The third bandwidth reduction approach UF has taken (Figure 1-6) is the most dramatic in bandwidth reduction since only the spike time is transmitted, but it does not allow for spike sorting [49, 50]. As previously mentioned this may be appropriate for applications such as in low-precision BMIs [2, 35]. The dramatic reduction in bandwidth allows more electrodes to be recorded which is helpful in many applications. The UF approach for spike detection originated with a bandpass filter and evolved to a multi-scale spike detection circuit based on wavelets. Both approaches are ultra-low power and robust while the multi-scale spike detector allows for better performance than other simple spike
detectors that could be implanted. This approach will be presented in detail in Chapters 5 and 6.

1.5 Dissertation Structure

The development of a low power implantable circuit for spike feature extraction and another circuit for detection of spikes which both dramatically reduce the required communication bandwidth out of the skin are presented in this dissertation. An introduction to extracellular neural recordings and current systems has been presented. The remainder of this dissertation follows UF’s data reductions approaches. Beginning with three implementations schemes for the feature extractor and pulse train spike sorter in Chapters 2, 3 and 4. Chapter 2 introduces the novel pulse-based feature extractor followed by Chapter 3 where the bandwidth reduction is examined and Chapter 4 provides the details of the feature extractor circuit and chip results. Data reduction with Feature extraction is followed by data reduction with spike detection where chapter 5 introduces the novel single-scale spike detector and Chapter 6 extends the single scale spike detector to multiple scales increasing the performance with minimal power increase. Chapter 7 concludes the dissertation with an overview of previous chapters and a summary of contributions.
CHAPTER 2
PULSE-BASED FEATURE EXTRACTION AND SPIKE SORTING:
IMPLEMENTATION 1 BACK-END SOFTWARE

2.1 Pulse-Based Feature Extraction

Some of the pulse-based feature extractor work presented in this chapter has been previously published [51]. The pulse-based feature extractor can be implemented entirely in software, entirely in analog circuitry, or a hybrid with analog circuitry for the front-end feature extractor and software for the back-end spike sorter. Pulse based spike feature extraction has been used in this work for low-power and low-bandwidth data transmission. The circuit implementation entails modifying the current spike detector’s current threshold to an area threshold using a leaky integrate-and-fire neuron. Instead of transmitting the raw waveform, the pulse-based feature extraction method encodes information about the spike in a biphasic pulse train. This greatly reduces the bandwidth required to transmit the spike trains especially because spike occurrences are sparse within the neural data, while the pulse communication offers lower power transmission options such as ultra-wide band (UWB). The encoding scheme uses pulses based on area per time thresholds to represent the spike while the noise is mostly discarded. Only the spikes and their time within the spike train contain information so not transmitting information about the noise saves power without any drop in system performance.

The noise is discarded more severely than when reconstruction is needed by using a leaky term with biphasic encoding. This allows the pulse trains for spikes with different preceding noise to still synchronize which aids in spike sorting. The system diagram is shown in Figure 2-1 with the leaky term to subtract out noise in the signal. The leaky value sets the cutoff frequency for the low-pass filter formed with the integrator. This leakiness along with the proper threshold settings allows for very few pulses to represent the noise and the majority of pulses to contain information about the spikes. The leaky component changes the equation for the constraint of two consecutive pulses to
where $\theta_i \in \{-\theta, \theta\}$ and C is related to the integration capacitor and the R is related to the leak value.

Once the pulse trains have been transmitted, a classifier can then perform spike sorting outside the body where power issues are not so critical. The encoded pulses for each spike serve as a spike signature, where a pulse-based spike sorting algorithm is used to classify the spikes. The classifier would be trained once in the initial setup and then could be periodically retrained if necessary by sending short segments of the raw waveforms from one electrode at a time. One of the more difficult cases for this type of spike sorter would be two spikes from different neurons but with the same area. However, in this case, the taller and narrower spike would have more spikes in a shorter time period so the two would have different spike signatures and could still be distinguished.

### 2.2 Data Reduction with Pulse-Based Feature Extractor

The feature extractor employs a leaky integrate-and-fire neuron to produce its biphasic pulse representation. A block diagram for the biphasic encoding system is shown in Figure 2-1. If the output of the integrator, $y(t)$, reaches the positive threshold of the comparator, $\theta$, the output of the comparator raises and resets the integrator after a short delay, $\tau$, in the feedback loop. Similarly, if the output of the integrator $y(t)$, reaches the negative threshold, $-\theta$, the output of the comparator drops and also resets the integrator. The leak term is a fixed value to filter out noise. The leak value sets the cutoff frequency for the low-pass filter formed with the integrator. This leak value along with the proper threshold settings allows for very few pulses to represent the noise and the majority of pulses to contain information about the spikes. The timing of two consecutive pulses must satisfy the following equation:

$$\frac{1}{C} \int_{t_i + \tau}^{t_{i+1}} x(\Delta)e^{\frac{\Delta - t_{i+1}}{RC}} d\Delta = \theta_i$$

(2-2)

41
where $\theta_i \in \{-\theta, \theta\}$ and $C$ is related to the integration capacitor and the $R$ is related to the leak value.

2.3 Spike Sorting with Pulse Trains

For our feature extractor the signals are biphasic pulse trains thus traditional spike sorting algorithms can not be directly applied. The encoded pulses for each spike serve as a spike signature, where a pulse-based spike sorting algorithm is used to classify the spike. While distortion metrics for spike trains have been studied in areas such as neuroscience and genetics, many methods are computationally complex and far from real time such as the edit distance [52]. Another idea is to low-pass filter the pulse trains with a function, such as an exponential, so more traditional signal processing can be applied [53]. Instead of trying to reconstruct the signal, the spike sorter used for the pulse-based feature extractor similarly convolves the pulse train with a Gaussian function, where the $\sigma$ determines if the detector is more of a coincidence detector ($\sigma$ much smaller than the interpulse interval) or a pulse count detector (large $\sigma$). A Gaussian function was chosen as it is more concentrated around the peak allowing the $\sigma$ to better control the detector type. Once the pulse train is convolved with the Gaussian, it is then compared to each user defined neuron template. The template with the lowest MSE is a match unless it exceeds the maximum allowed MSE and then it is considered noise.
Examples of the spike signatures for the six neural simulator neurons spikes (the neural simulator data is explained in detail in Section 2.4.1.1) are shown in Figure 2-2. As previously mentioned, comparing two pulse trains is computationally expensive so the signature is convolved with a Gaussian to allow traditional template matching signal processing techniques to be applied. An example of the signatures once they are convolved with a Gaussian for the neural simulator data are shown in Figure 2-3.

To show the importance of selecting the Gaussian $\sigma$, Figure 2-4 shows the error versus $\sigma$ value on the neural simulator data set. If the best detector was somewhere between a coincidence detector and spike count detection, the curve would have a U-shape with a sweet spot for $\sigma$ to where the distance between the pulse trains is somewhere between a coincidence detector and a pulse count detector. The six neurons in the neural simulator are all very distinct and the noise is low so it do not require any coincidence detecting to classify the spikes. Thus, the curve is more like half a U with the right end flattening out because as $\sigma$ continues to increase (less and less coincidence detector) there is little change in error.

There is a problem with using a single value of $\sigma$. Spikes with different amplitudes have different interspike intervals and in fact within a single spike the interspike interval changes since it is part of the signal encoding. The problem is that the value of $\sigma$ which corresponds to a coincidence detector or a pulse count detector depends on the interspike interval of the pulse train. Thus, for a single spike one $\sigma$ values means part of pulse train distance will be computed using one type of detector while other parts of the spike will be more toward the other type of detector. This is a problem because the detector type changes based on inter-pulse interval which is useful as a feature. A useful variance of $\sigma$ in time might be for the detector to become less of a coincidence detector towards the end of the spike because the leak value only synchronizes the beginning of the spike and by the end of the spike the accumulated noise will cause the later pulse times to deviate move. Three different ways to set $\sigma$ will be discussed.
Figure 2-2. Neural simulator spike signatures from six different neurons at two different time periods. The first two rows represent one time period and the next two rows represent another time periods, each with the raw signal on top and the signature for each spike below.
Figure 2-3. Neural simulator spike signatures convolved with a Gaussian to determine the distance between it and the templates. A) Pulse spike signature. B) Pulse spike signature convolved with a Gaussian.
Figure 2-4. Spike sorting error as a function of the pulse train distance Gaussian $\sigma$.

One solution is an adaptive leaky integrate-and-fire (LIF) threshold to create a more uniform pulse rate and is inspired from the biological neuron’s adaptive threshold mechanism to keep the firing rate from saturating and information being lost [54]. Another solution is the addition of a refractory period which does not allow the LIF to fire another pulse until after a certain period of time which sets a maximum firing rate. In this case though the signal is ignored during the refractory period so information is lost and presumably the adaptive threshold method preserves more information and is thus more desirable. The third option is to change the $\sigma$ value according to the spike template interspike interval to produce a more constant detector across the spike.

The focus of this research is feature extraction not spike sorting, but in order to analyze the performance of the feature extractor spike sorting must be performed. Thus, the spike sorting procedure was kept simple to purely show the feature extraction has potential. Results in this paper were obtained by simply using the first spike from each neuron as a template. This is a worse case template formation because often an average over several spikes is taken to eliminate noise. A second or third spike template could be
used to estimate the pulse jitter statistics that could be used to set the $\sigma$ of the Gaussian. As the feature extractor reduces the noise level, using a single spike as a template proved adequate in this data set.

### 2.4 Matlab Simulations Results

#### 2.4.1 Data

Two data sets were used to evaluate the feature extractor. One data set came from a neural simulator so the ground truths are known and the other was recorded in vivo from a rat and an expert marked the sorting ground truths using Spike2.

#### 2.4.1.1 Neurosimulator data

The pulse-based feature extractor algorithm was tested with neural recordings from Bionic’s 128-channel hardware neural signal simulator. The use of a neural signal simulator allows the ground truths, the time of each spike and which neuron it came from, to be known. The neural simulator outputs a repeated 11 s pattern of spikes from three different action potentials with amplitudes of $100 \mu V - 150 \mu V$ and a width of 1 ms. The interspike interval is 1 s for 10 s and then reduces to 10 ms for 1 s of burst firing. To increase the number of neurons on one channel the reference was chosen as another channel instead of ground. The referenced channel was carefully chosen to be a 5 ms delayed version of the first channel. In this manner, the simulated neural signal contains spikes from six different neurons with no superimposed spikes which are not addressed in this work since they are problematic for all spike sorting algorithms.

The UF bioamplifier [25], with a gain of 100 and a low cutoff frequency of 0.3 Hz and a high cutoff frequency of 5.4 KHz, was used to amplify the neural simulator output. The amplified signal was then digitized at $\sim 24.4$ KHz and 34.6 s were captured with a digital logic analyzer. The average spike firing rate for the data set is 19 Hz. The signal’s SNR is about 30dB. A portion of the signal during bursting with all six neural spikes is shown in Figure 2-5(A).
Figure 2-5. Neural simulator signal with all six neurons and the biphasic pulse train output from the LIF circuit for a bandwidth of 455 pulses/s. A) Neural simulator signal with all six neural spikes, one through six from left to right. The second row is the biphasic pulse train output from the LIF circuit with the bandwidth at 455 pulses/s. B) Zoomed in spike three. C) Zoomed in spike four.
2.4.1.2 Rat data

The feature extraction algorithm was then tested on neural recordings from male Sprague-Dauley rats chronically implanted with 50 μm polyimide insulated tungsten microwire electrode arrays in layer V of the forelimb region of the primary motor cortex. The data was sampled at 25 KHz and bandpass filtered between 0.5 and 12 KHz using hardware from Tucker-Davis Technologies. Action potential widths ranged from 0.4 ms–1.2 ms with amplitudes as high as 137 μV. High SNR recordings were chosen to increase the confidence of the ground truth spikes times.

Ground truths were labelled by a human expert using Spike2 [55]. More detailed analysis of spike sorting with Spike2 follows in Section 2.4.2 while a brief explanation of the spike detection used in Spike2 is explained here. The method of marking spikes was to first parse out data segments with a possible spike. This was done using a conservative (low) threshold and extracting segments around the threshold crossing from the waveform. Then, these segments were examined and only those which actually contained a spike were kept and labelled in the spike time file. This means there were few false alarms but spikes with large negative peaks could have been omitted if their second phase did not cross the positive threshold. We determined that in one of the data sets over the first five seconds of data 18 spikes fit this criteria. Thus, when characterizing the feature extractor’s sorting performance, the error components, false alarms and missed detections, will be analyzed separately. This will allow false alarms from missed detections in Spike2 (our ground truth) not to count as an error.

To increase the ground truth accuracy and remove any bias of an individual expert several experts should mark the ground truths with their averaged results becoming the ground truth. Marking spike times is a tedious task though, so it is difficult to enlist human experts to mark a lot of data. For the parsing out of possible spikes a second negative threshold could be used in addition to the positive one to decrease the
number of missed detections in the ground truths but this could also decrease the sorting performance.

Figure 2-6 shows the original neural data waveform. SNR was calculated for the signal in terms of power using an averaged spike shape.

![Neural waveform](image)

Figure 2-6. Neural waveform recorded from rat003. Column two is zoomed in from column one.

2.4.2 Spike2

*Spike2* [55], a popular commercial program first written in 1988, which can spike sort offline, is used as a comparison to the feature extractor’s spike sorting performance. *Spike2* first performs crude spike detection by capturing windows around events that cross a user defined threshold(s). Then, spike sorting is performed with a combination of template matching and a PCA based cluster cutting. templates from the neural simulator data are shown in Figure 2-7 and PCA analysis shows the well separated six classes in Figure 2-8. This process requires the user to select many parameters during the template setup such as the number of templates and allowable variation within the template. *Spike2* provides the user with a interactive visual display to assist in setting the spike sorting parameters. The parameters were set by an expert in the field with the same procedures used in typical experiments.
Figure 2-7. Spike2’s templates for neurosimulator data.
Figure 2-8. Spike2’s principal component analysis (PCA) for neurosimulator data.
### 2.4.3 Spike Sorting Results: Neurosimulator data

*Matlab* was used to simulate the pulse-based feature extractor and its spike sorter. The LIF was set with a threshold and leakage value such that its spike sorting error was minimal resulting in an error of 0.5% compared to Spike2’s 6.1%. Over a wider range of threshold and leakage values the feature extractor obtains \(<3\)% error. The percent error was calculated using Equation 3-1. Figure 2-5(B) shows examples of the biphasic output for spikes from two different neurons. The regions between spikes did not have any pulses.

\[
\text{% error} = \frac{\Sigma \text{missed spikes} + \Sigma \text{false positives}}{\text{total number spikes} + \Sigma \text{false positives}}
\]  

There are several reasons for the feature extractor’s outstanding performance. First, the data set has a high SNR with distinct spike shapes thus it is a relatively easy spike sorting data set, but this is also true for Spike2. Second, the feature extractor has fewer parameters to set than Spike2 so with limited data is could be better optimized. This is why this is not the only data set used to analyze the feature extractor’s performance. Third, the feature extractor utilizes the leaky parameter to reduce noise which leads to increased sorting error and its features are robust to noise.

### 2.4.4 Rat data spike sorting results

As others working in this area have stated, it is difficult to test a spike sorting algorithm on a real data set because the ground truths are not known. Thus, the new algorithm is penalized for its errors as well the ground truth errors if it get those correct. Therefore, some groups have solely focused on developing realistic simulated data where the ground truths are known with complete accuracy. As the feature extractor was designed to eventually be implanted as part of a neural recording system, it is necessary to analyze its performance with real data compared to current spike sorting systems.

The “ground truths” are used as a first pass and then the false alarms and misclassifications are examined to see if they were correct and the ground truths were wrong or if they are indeed error. Statistical analysis can be used to see if the “errors” are indeed errors (for
example PCA can be performed to see how well the ground truth classes are clustered etc.) but for the most part it is a combination of statistical analysis as well as an experts gut instinct from years of experience. The terms error, misclassification, and false alarm will be used rather loosely as the “ground truths” are not absolute in themselves and contain error.

The rat data spike sorting “ground truths” obtained from an expert in the area, using Spike2. Some missed detections occurred because only a single positive threshold was used so there are missed spikes which can be picked out just by looking at the data, but the positive portion of those spikes was attenuated due to noise. The templates were formed and refined using PCA. The results templates are shown in Figure 2-9 and the PCA clusters are shown in Figure 2-10. Each color corresponds to a different class with black being spikes that were classified as noise. It is difficult to tell how well isolated the PCA clusters are in 2-D but Spike2 allows the user to rotate the figure to analyze it better. The red and blue classes and the cyan and magenta are the closest classes so there is likely some misclassifications between those pairs due to noise in the signal. As the SNR is poorer than the neural simulator data and the spike shapes between classes are more similar, there is less distinction between the classes making it a more difficult sorting problem.

Histograms of times between spikes can also be plotted to see if more than one neuron was classified into the same class. Because absolute refractory periods are known to be about 1 ms, if there are any spikes that fire within the refractory period of the neuron then those must be misclassified. The converse is not true as two neurons could be clustered together but never fire with in the refractory period of each other. Figure 2-11 shows the histogram for the sorted rat data.

The difference between Spike2’s classification and the feature extractor is 33% error. Because the ground truths have error themselves this difference is not necessarily error and must be examined in a different manner. A first comparison is to see if the templates
Figure 2-9. Spike2 example template for rat data.
Figure 2-10. Spike2’s PCA analysis for rat data.
look similar but as Spike2 uses the time domain and the feature extractor encodes the signal in a pulse train and then convolves it with a Gaussian so they can not be directly compared. The templates for Spike2 are shown in Figure 2-12 and the templates for the feature extractor are shown in Figure 2-13.

Next, the correctly classified spikes will be examined. Pile plots of Spike2’s sorted spikes are shown in Figure 2-14. The a pile plot of the spikes sorted correctly referenced to Spike2’s results from the feature extractor are shown in Figure 2-15. Notice neurons 2, 3, and 6 appear cleaner in the feature extractor while neuron 1 has more outlying spikes. This could mean the classification for the feature extractor is tighter on neuron 2, 3, and 6 but looser for neuron 5.

There are three types of errors to examine: misclassification, false alarms, and missed detections. Table 2-1 shows the difference in classification between Spike2 and the feature extractor (misclassifications). The table shows the pulse-based feature extractor and sorter misclassifies certain neurons more than others. For instance neuron 2 and 4 never receive
Figure 2-12. Spike2’s templates.
Figure 2-13. Feature extractor templates.
Figure 2-14. Pile plots of Spike2’s sorted spikes.
Figure 2-15. Pile plot of feature extractor’s correctly sorted spikes referenced to Spike2’s results.
misclassifications so their templates and rules must be distinct enough from the other neurons. However neuron 3 had many of its spikes misclassified at neuron 1 and neuron 5 had many of its spikes misclassified as neuron 3 so these classes need better separability.

Table 2-1: Feature extractors misclassifications compared to Spike2.

<table>
<thead>
<tr>
<th>actual neuron</th>
<th>classified as</th>
<th>neuron</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>tot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>-</td>
<td>0</td>
<td>4</td>
<td>2</td>
<td>48</td>
<td>65</td>
<td>119</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td>197</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>5</td>
<td>6</td>
<td>208</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>-</td>
<td>2</td>
<td>62</td>
<td>69</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td>42</td>
<td>0</td>
<td>241</td>
<td>2</td>
<td>-</td>
<td>11</td>
<td>296</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td>8</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>28</td>
<td>-</td>
<td>43</td>
</tr>
<tr>
<td>tot</td>
<td></td>
<td></td>
<td>250</td>
<td>5</td>
<td>246</td>
<td>8</td>
<td>83</td>
<td>145</td>
<td></td>
</tr>
</tbody>
</table>

To try and discern which misclassification may have been legitimate (meaning were actually an error with Spike2 or just distorted by noise enough the spike resembled two templates) pile plots of the true neuron spikes with all of the spikes misclassified as from that neuron are examined. Figure 2-16 shows pile plots of the feature extractor’s correctly classified spikes overlaid with the spikes misclassified as from that neuron in the color neuron they actually belong to. Most of the spikes misclassified as 2 and 5 reside close to those detected as 2 so they may be errors from Spike2. However, most of the spikes misclassified as 1 and 6 differ enough from those detected to show the rules for those classes are too loose.

Figure 2-17 shows pile plots of the feature extractor’s correctly classified spikes overlaid with the spikes from that class but misclassified as another class in that classes color. These pile plots shows how the feature extractor’s errors relate to Spike2’s classification. For example for neuron 3 the spikes the feature extractor misclassified different enough they may have come from another neuron. Those spikes from neuron 2 misclassified by the feature extractor were at the edges of the pile plot so likely if the feature extractor rules for class 2 were loosened or the rules for class 4 were tightened those misclassification would disappear.
Figure 2-16. Pile plot of feature extractor’s correctly sorted spikes with those misclassified as that neuron overlaid with a dashed black line.
Figure 2-17. Pile plot of feature extractor’s correctly classified spikes overlaid with the spikes from that class but misclassified as another class in that classes color.
This analysis shows the feature extractor and sorter’s parameters could be tweaked to improve performance but as the desired result is to outperform Spike2 (as was shown possible with the neural simulator data) only experts can objectively state which sorting is acceptable.

2.4.5 Future Work

Several areas of the feature extraction algorithm need to be further studied in a different work to improve sorting performance while decreasing bandwidth. They are listed below:

- **σ selection**: The inter-pulse interval of each signature varies based on the waveform. If the sigma remains constant, the type of detector (coincidence or pulse count) changes as the inter-pulse interval changes. Implications of not adjusting the Gaussian’s $\sigma$ as the pulse rate changes need to be examined while methods to adjust $\sigma$ explored. Also, it could be advantage to purposefully adapt $\sigma$ making it more of a coincidence detector at the beginning of the spike when the pulses are more synchronized from the leaky component.

- **Methods for automatically setting leakiness and threshold for optimal performance**: It is believed the optimal leakiness value is related to the noise floor and the optimal threshold value to the spike amplitudes. This needs to be studied an automatic method to set these values developed such as based on SNR of the signal the leaky and threshold values are set.

- **More extensive performance measures**: The pulse-based feature extractor and spike sorter need be tested on more extensive data sets with different SNRs and spike shapes to ensure its performance across data. Also, a better analysis of performance on data sets without ground truths is needed to evaluate in vivo performance.
CHAPTER 3
PULSE-BASED FEATURE EXTRACTOR AND SPIKE SORTING: IMPLEMENTATION 2 SOFTWARE FRONT-END AND BACK-END

Moving the feature extractor to the front-end allows for a dramatic decrease in bandwidth though it is not without effect on spike sorting performance. That relationship is examined in this chapter while the principles of operation for the feature extractor and sorter are not repeated from Chapter 2.

3.1 Bandwidth Parameters

The front-end of the feature extractor is responsible for sending out pulses which represent the best features for spike sorting with minimal bandwidth. It is a tradeoff between spike sorting performance and bandwidth. In the pulse-based feature extractor three parameters determine the bandwidth, the leak value, threshold values, and the integration time constant. The integration time constant is normalized to a value of 10 pF and the effect of the leak value and threshold value on bandwidth and spike sorting performance is explored.

Two extremes are to set the leakage so high that none of the signal passes through or to set the thresholds so high that the signal never surpasses it to send any pulses. In either of these two situations no information is preserved. Another extreme is to set the leakage to zero and the thresholds very low and this preserves all of the information, more than enough to allow for perfect reconstruction on the back-end so bandwidth is wasted.

Increases in the leakage and threshold values both decrease the bandwidth as shown in Figure 3-1. While more than one leakage and threshold value will give the same bandwidth they do not necessarily provide the same sorting error, described in Section 3.2.1 and is desirable to minimize, as they do not preserve the same information as shown in the plot of Figure 3-2. Figure 3-3 shows sorting error versus minimum bandwidth for the leakage and threshold combinations at three different SNRs. This plots show the inverse relationship between bandwidth and sorting error. It also shows, that as SNR decreases the sorting error increases as expected. The lower the SNR the less pointwise
Figure 3-1. Bandwidth (pulses/s) changes for threshold and leakage values parameters and integration capacitor at 10 pF.

separation there is between pulse trains of different neurons meaning it is easier to misclassify one as the other increasing the errors. However, the performance of all spike sorters drops as SNR decreases.

3.2 Matlab Simulations Results

The neural simulator data was used so the ground truths are known. For details about the neural simulator data set please refer to Section 2.4.1.1.

3.2.1 Spike Sorting Results: Neural Simulator Data

Matlab was used to simulate the pulse-based feature extractor and its spike sorter as in Section 2.4 but not also in regards to bandwidth. The LIF was set with a threshold and leakage value such that its spike sorting error was similar to Spike2’s which resulted in a bandwidth of 455 pulses/s. Figure 2-5(B) shows examples of the biphasic output for spikes from two different neurons. The regions between spikes did not have any pulses. The biphasic output was then spike sorted with the results shown in Table 3-1 along
Figure 3-2. Spike sorting error changes for threshold and leakage values parameters and integration capacitor at 10 pF. The color represents log(bandwidth) (pulses/s)

with the results from $Spike2$. The results are divided into each neuron class and the percent correctly classified (true positives, tp) and the false positives (fp) which are spikes incorrectly classified as from that neuron. The best case is 100% tp and 0% fp. The percent error is calculated with equation 3-1.

$$\%\text{ error} = \frac{\Sigma \text{ missed spikes} + \Sigma \text{ false positives}}{\text{total number spikes} + \Sigma \text{ false positives}}$$  \hspace{1cm} (3-1)

As Table 3-1 shows, neuron 2 (the smallest spike) is one of the hardest to classify for both sorters. Neuron 2 is more poorly classified with the feature extractor at the lower bandwidth, 455 pulses/s because it did not have enough pulses to represent it and some information was lost. The addition of an adaptive threshold [56] would help to even out the number of pulses for different amplitude spikes to keep more similar amounts of information for all spikes without having to increase the number of spikes for all neurons as in the result with 680 pulses/s. The adaptive threshold will create a
Figure 3-3. Spike sorting error as a function of bandwidth for three SNRs.
more uniform sorting performance across neurons without having to increase bandwidth as significantly. The adaptive threshold is inspired from the biological neuron’s adaptive threshold mechanism to keep the firing rate from saturating and information being lost [54]. Another solution is the addition of a refractory period which does not allow the LIF to fire another pulse until after a certain period of time. In this case though the signal is ignored during the refractory period so information is lost and presumably the adaptive threshold method preserves more information and is thus more desirable.

Table 3-1: Spike sorting performance percent error.

<table>
<thead>
<tr>
<th>neuron</th>
<th>Feature extractor 455 pulses/s</th>
<th>Feature extractor 680 pulses/s</th>
<th>Spike2 300 Kbps</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>% tp</td>
<td>% fp</td>
<td>% error</td>
</tr>
<tr>
<td>1</td>
<td>100</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>69.7</td>
<td>2.6</td>
<td>31.5</td>
</tr>
<tr>
<td>3</td>
<td>100</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>97.3</td>
<td>0</td>
<td>2.8</td>
</tr>
<tr>
<td>5</td>
<td>96.4</td>
<td>1.9</td>
<td>5.4</td>
</tr>
<tr>
<td>6</td>
<td>100</td>
<td>0.9</td>
<td>0.9</td>
</tr>
<tr>
<td>avg</td>
<td>93.9</td>
<td>0.8</td>
<td>6.8</td>
</tr>
</tbody>
</table>

Overall at 455 pulses/s the feature extractor had 6.8% error compared with Spike2 which had 6.1% error. While maintaining a similar classification error to traditional sorting with Spike2, the feature extractor requires much less bandwidth with only 455 pulses/s compared to 300 Kbps for a traditional 25 KHz sampled signal at only 12-bits. 1 pulse/s is equivalent to 1bps. UF’s biphasic output for reconstruction on the back-end would require 71.9K pulses/s. The pulse-based feature extractor can reduce its bandwidth even further if more sorting error can be tolerated or increase its bandwidth to lessen sorting errors. The two are inversely related.

At 680 pulses/s the feature extractor actually has less error than Spike2 for this data set showing it is competitive. More data simulations need to be performed across different SNRs and data sets to see if the trend continues, but one possible explanation is that the feature extractor preserves the important information in distinguishing between spikes.
while eliminating extraneous information. Extra information can make it more difficult for a neuroscientist to optimally set the spike sorting parameters in Spike2 making it harder to distinguish between spikes from different neurons.

A summary table comparing different data reduction techniques and their affect on sorting error are shown in Table 3-2.

Table 3-2: Bandwidth reduction sorting error comparison

<table>
<thead>
<tr>
<th></th>
<th>ADC</th>
<th>Biphasic with reconstruction</th>
<th>Biphasic feature extraction</th>
<th>Biphasic feature extraction</th>
<th>Spike detection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Front-end bandwidth</td>
<td>300 Kbps</td>
<td>72 K pulses/s</td>
<td>455 pulses/s</td>
<td>680 pulses/s</td>
<td>20bps+ channel bw</td>
</tr>
<tr>
<td>back-end spike sorting error</td>
<td>6.1%</td>
<td>≥ 6.1%</td>
<td>6.8%</td>
<td>3.8%</td>
<td>N/A</td>
</tr>
</tbody>
</table>

The addition of an adaptive threshold [56] would help to even out the number of pulses for different amplitude spikes to keep more similar amounts of information for all spikes without having to increase the number of spikes for all neurons as in the result with 680 pulses/s. The adaptive threshold will create a more uniform sorting performance across neurons without having to increase bandwidth as significantly. The adaptive threshold is inspired from the biological neuron’s adaptive threshold mechanism to keep the firing rate from saturating and information being lost [54]. Another solution is the addition of a refractory period which does not allow the LIF to fire another pulse until after a certain period of time. In this case though the signal is ignored during the refractory period so information is lost and presumably the adaptive threshold method preserves more information and is thus more desirable.

3.2.2 Future Work

In addition to the future work items in Chapter 2, several areas of the feature extraction algorithm need to be further studied in to improve sorting performance while decreasing bandwidth. They are listed below:
• **Adaptive threshold versus refractory period:** To equalize information across spikes regardless of amplitude either an adaptive threshold or a refractory period can be implemented. The adaptive threshold adapts the LIF comparator threshold proportionally to the output pulse rate in a similar manner at real neurons due to increase dynamic range and keep from saturating and information being lost. The refractory period keeps the LIF from firing for a certain time after each pulse. It is simpler to implement but information is lost. Analysis must be done to examine the tradeoffs between the techniques in terms of bandwidth, circuit area, power consumption and their affect of spike sorting performance.

• **How to optimally set leakiness and threshold values to optimize both bandwidth and performance:** Different applications will have different constraints for bandwidth and performance. It would be good to have a simple way to take a desire value be it bandwidth or spike sorting performance of limits for both the bandwidth and spike sorting performance and automatically optimally set the leakiness and threshold values.
CHAPTER 4
PULSE-BASED FEATURE EXTRACTION AND SPIKE SORTING: IMPLEMENTATION 3 HYBRID

While a purely software implementation shows promise in comparison to Spike2, a major advantage of the pulse-based feature extractor is that it can be efficiently implemented using compact low-power analog hardware. This is done with approach three, a hybrid solution, an an analog feature extractor in the front-end and a software spike sorter at the back-end.

4.1 Circuit Design

An overview of the feature extraction algorithm was provided in Chapters 2 and 3. Now the details of the circuit will be presented. The chip was built using the AMI 0.5 µm CMOS technology.

4.1.1 Circuitry

The feature extraction algorithm was developed amenable to low-power analog circuitry so it can be implanted as part of a neural recording system if desired. The integration time constant is realized as a capacitor and is set to 10 pF at fabrication time. The leakage value and threshold values can be adjusted after fabrication for the desired bandwidth and spike sorting error.

The feature extractor circuit, shown in Figure 4-1, takes a current input and encodes the neural signal’s shape in a biphasic pulse train using a leaky integrate and fire (LIF) neuron, a simple extension (such as adding a $G_m$ current source or a resistor in parallel to the integrator capacitor for the leakiness) of the biphasic IF neuron [48]. The LIF neuron integrates the signal and then produces a positive pulse when the integrated signal rises above one threshold and a negative pulse when it falls below a second threshold. The leakiness of the LIF sets an area per time threshold to filter out noise while preserving the spikes. This allows the noise in the signal to only trigger an occasional stray pulse, and thus keeps the bandwidth and power consumption even lower.
Each block of the LIF will be explained though only the leaky component will be discussed in detail as the other IF parts follow previous work as explained in detail in Dr. Chen’s publications [47, 48] and Dr. Li’s dissertation [57].

4.1.1.1 Voltage to current converter circuit

The input to the LIF needs to be a current, however the output of the bio-amplifier [25] which precedes the LIF provides a voltage. Thus, a voltage to current circuit must be used which is shown in Figure 4-2. $C_1$ rejects DC which is crucial to limit the feature extractor’s bandwidth. A common differential PMOS input and cascoded output operational transconductance amplifier (OTA) is used in the voltage to current block and it’s schematic is Figure 4-2. A more detailed analysis of this circuit can be found in Dr. Li’s dissertation [57].

4.1.1.2 Comparator circuit

Figure 4-4 shows the comparator circuit which consists of three stages: the input preamplifier, a decision block, and an output buffer. The input signal is sensed by the
input pair M1/M2 and differential currents are copied to the next stage, the decision block. In the decision block positive feedback is used to isolate the input pair to help decrease the kickback noise. The cross-connected pair M7/M8 increase the gain of the comparator. The diode-connected pair M10/M11 provides hysteresis to reject the noise on the input signal. M13 provides a DC shift to guarantee the swing of the decision circuit output is in the common-mode range of the output buffer. M14-M18 form an output buffer to convert the final output of the comparator into a logic-level signal. An inverter was added to the output of the buffer as an additional gain stage. A more detailed analysis of the comparator can be found in Dr. Chen’s work [25, 47].

4.1.1.3 Reset and refractory period circuit

The reset component resets $V_{mem}$ after either a positive or negative pulse occurs using a simple OR gate. The circuit also contains the option for a refractory period (time delay when the reset stays on) but it is minimized when the LIF is used for feature extraction. The refractory period component is realized by an asymmetric current-starved inverter depicted in Figure 4-5 and Dr. Chen’s dissertation [47] includes a more thorough analysis. The circuit is composed of the input CMOS pair M1/M2 with an additional
Figure 4-3. Operational transconductance amplifier OTA for voltage to current convertor circuit for LIF.

series-connected PMOS transistor M3 in the pull-up. A PMOS transistor M4 is employed for testing purpose. The control voltage $V_{bias-refractory}$ adjusts the output rise time and thus the refractory period. For feature extraction the $V_{bias-refractory}$ is set for minimal refractory period.

4.1.1.4 Leaky circuit

The addition of a leaky component is what distinguished the LIF from the IF. The leaky component, a $G_m$ current source, was realized using an OTA configured as a unity
Figure 4-4. Comparator circuit for LIF.

gain-follower. The OTA’s schematic is depicted in Figure 4-6. The OTA’s bias voltage adjusts the amount of leakage. For a small input signal range such as neural signals have the current is linear. A positive input value will sink current while a negative input value sources current.

4.1.1.5 Chip specifics

Cadence SpectreS simulations show the LIF circuit consumes about 30 $\mu W$ of power. The LIF circuit was fabricated using AMI 0.5 $\mu m$ CMOS technology. The chip is a 1.5 mm $\times$ 1.5 mm 40-pin DIP with 504 $\mu m \times 356 \mu m$ of circuit area as shown in Figure 4-7. The pinout for the leaky integrate-and-fire feature extractor chip used can be found in APPENDIX B. The chip has differential input and uses $+/-$ 2.5V power supplies.

4.1.2 Test Setup

Prior to testing the chip with any neural signals, bench top testing was done using a function signal generator to ensure the chip was functioning properly. Three test setups were used to progress towards in vivo testing of the chip Figures 4-8, 4-9, and 4-10. The
Figure 4-5. Reset and refractory period circuit for LIF.

test setups were developed to limit the amount of time required to test the chip in the rat lab as those resources are limited and shared by many people.

The initial chip test setup shown in Figure 4-8 allows the basic parallel recording setup to be tested in NEB 487 without requiring a rat. The two things recorded in parallel are the output of the UF bio-amplifier [25] and the output of the current TDT neural recording system used in the Neuroprosthetics Research Group (NPG) lab. TDT is used as the master clock and triggers the logic analyzer to record the LIF’s bi-phasic output at the desired time. The pulse times are later used for the pulse-based sorting algorithm to compare the feature extractor and pulse-based sorter sorting performance to Spike2’s performance based on the bio-amplifier’s output. Using the neural simulator as the input signal to the two systems allows the testing to be done outside of the rat lab and provides a known ground truth.

The intermediate test setup shown in Figure 4-9 replaces the neural simulator with a file output using TDT. The file out should be +/− 10 V to minimize added noise from the
Figure 4-6. Leaky circuit for LIF implemented as a $G_m$ current source.

RX5 digital-to-analog convertor (DAC) and the PA5 that is used to attenuate the signal to the desired neural signal level. This allows any neural data file to be used as input for the chip. This is especially useful if simulated data with known ground truths is to be used.

The final test setup uses in vivo recordings and is shown in Figure 4-10. It is the same as the initial setup except the input signal is now from a live rat and the complete TDT system used in the NPG lab is recorded in parallel as well. This allows a comparison
between two complete systems as well as any offline spike sorters as the amplified raw neural data is recorded.

4.2 Chip Results

The feature extractor chip was tested using the test setups mentioned previously in Section 4.1.2 and are described in detail below. From benchtop testing the LIF chip was found to have more noise than the previous IF chip though both suffer from feedback from the digital pulse output to the analog input even with the addition of off chip decoupling capacitors between the digital power and ground input pins and separate decoupling capacitors between the analog power and ground input pins. Another student laid out the LIF chip but did not follow the IF chip layout with just adding the leaky component...
so apparently the layout to minimize the digital to analog feedback was not as careful as with the IF chip. This noticeably degrades the performance of the LIF when used for reconstruction but is not as large a factor when the LIF chip is used for feature extraction as the features are purposely robust to noise as well as the sorting algorithm.

4.2.1 Neural Simulator

The neural simulator provides absolute ground truths which make the performance analysis of the feature extractor straightforward and a good intermediate step between bench top testing and in vivo testing where there are no absolute ground truths. In order to use the same exact input at different leakage and threshold setting for the LIF the intermediate test setup from Figure 4-9 was used. However, to mimic the complete system, neural simulator data recorded from the UF bio-amplifier \cite{25} was used. The data was rescaled to +/- 10 V to reduce noise as mentioned in Section 4.1.2 and then attenuated with the PA5 back to it’s original amplitude level. The process of digital-to-analog conversion in the RX5 and attenuation in the PA5 introduce additional noise on the order
Figure 4-9. Intermediate chip test setup: Set UF's feature extractor chip parameters using prerecorded data from rat that will be used in in vivo experiments (487 NEB).

of $\mu$V than if the direct output of the amplifier went to the LIF chip however the ability to replay the exact same data for each leakage and threshold parameter combination outweighed the additional noise present.

A leakage and threshold value were chosen to obtain the desired bandwidth from the simulation results presented in Section 3.2.1. Then, a value above and below that was chosen to test the LIF chip as simulation results do not always map exactly to chip results because of non-ideal factors. This resulted in a total of nine data set recordings.

The sorting error is shown in Table 4-1 and the bandwidth is shown in Table 4-2. The spike sorting error is similar to the simulation results in Section 3.2.1 for the lower bandwidth values but as the bandwidth increases the noise from the digital output feedback to the analog input is increased. Thus, the largest bandwidth data has the most effect. Also, at this point detection (being able to separate individual spikes) becomes an issue because of the current implementation of the software. This combination of problems results in an inability to spike sort at all with 99% error. The leakage level of 747.4 nA is not large enough to lose the noise and thus produced poor sorting results. Even when the leakage value is increased to 847.3 nA the error is still larger than expected for the bandwidth required. This is likely due to the large feedback noise from the digital
output to the analog input which adds many pulses to the signal. This was concluded after observing individual spike signature varying more than expected for even the largest of leakage current. The feedback noise is short but large in amplitude so it will perturb the timing of the pulses and often add extra pulses which could account for the increased variation in spike signatures compared to simulation results. Another noise source is the quantization of pulse times by the logic state analyzer (LSA) being 5 ns but this was accounted for in the simulations results.

The best performance was obtained with a leakage values of 946.1 nA and threshold value of 130 mV which produced a bandwidth of 1.31 K pulses/s and an error of 4.7%. This shows the LIF chip is capable of obtaining good sorting performance but requires more bandwidth than expected. More careful layout to separate the sensitive analog signals form the digital signals on the chip should allow the bandwidth to decrease for the same performance.

A summary table comparing the different bandwidth reduction techniques’ bandwidth, power consumption, and sorting performance are shown in Table 4-3. The feature extractor offers the lowest bandwidth and lowest power option while still being competitive in spike sorting so it appears very promising.
Table 4-1: Spike sorting performance (percent error) from leaky integrate-and-fire (LIF) feature extraction chip.

<table>
<thead>
<tr>
<th>threshold (mV)</th>
<th>130</th>
<th>150</th>
<th>170</th>
</tr>
</thead>
<tbody>
<tr>
<td>leakage nA</td>
<td>747.4</td>
<td>99.3</td>
<td>25.0</td>
</tr>
<tr>
<td>847.3</td>
<td>27.3</td>
<td>18.7</td>
<td>15.2</td>
</tr>
<tr>
<td>946.1</td>
<td>4.7</td>
<td>13.2</td>
<td>21.3</td>
</tr>
</tbody>
</table>

Table 4-2: Bandwidth (pulses/s) from LIF feature extraction chip.

<table>
<thead>
<tr>
<th>threshold (mV)</th>
<th>130</th>
<th>150</th>
<th>170</th>
</tr>
</thead>
<tbody>
<tr>
<td>leakage nA</td>
<td>747.4</td>
<td>1.65k</td>
<td>1.19k</td>
</tr>
<tr>
<td>847.3</td>
<td>1.42k</td>
<td>1.05k</td>
<td>885</td>
</tr>
<tr>
<td>946.1</td>
<td>1.31k</td>
<td>948</td>
<td>804</td>
</tr>
</tbody>
</table>

4.2.2 In Vivo with Rat

Ultimately the chip will be used in vivo with rats, but the performance analysis of the chip using rat data is difficult because ground truths are not known. Presently, simulated data sets from recorded data are being used to test the hardware using the intermediate test setup from Figure 4-9. This allows a wide variety of realistic data to be tested but because the ground truths are known a better analysis of the results can be done.

4.2.3 Future Work

In addition to the future work items in Chapters 2 and 3, the circuit design and chip design add several items that need further work.

- **How to implement the Leak**: Either a simple variable resistor can be used to set the leakage current or an ideally constant current source. The resistor would itself be a form of adaption by taking more current for larger signals. The effect of the two different leakage implementation on spike sorting performance as well as circuit area and power need to be investigated.

- **How to implement adaptive threshold in circuitry**: Circuits have already been developed [56] to implement the adaptive threshold as mentioned in Chapter 3 as an improvement to the feature extractor so the previous work can be built upon.

- **Feature extraction chip layout**: The layout needs special attention to reduce the feed through noise from the digital output to the analog input. An excellent well know book on analog layout is written by Alan Hastings [58].
Table 4-3: Bandwidth reduction, power consumption, and sorting error comparison

<table>
<thead>
<tr>
<th></th>
<th>ADC</th>
<th>Biphasic with reconstruction</th>
<th>Biphasic feature extraction</th>
<th>Biphasic feature extraction</th>
<th>Spike detection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Front-end bandwidth</td>
<td>300 Kbps</td>
<td>72 K pulses/s</td>
<td>455 pulses/s</td>
<td>680 pulses/s</td>
<td>20bps+ channel bw</td>
</tr>
<tr>
<td>back-end spike sorting error</td>
<td>6.1%</td>
<td>≥6.1%</td>
<td>6.8%</td>
<td>3.8%</td>
<td>N/A</td>
</tr>
<tr>
<td>power</td>
<td>3 mW</td>
<td>100 µW</td>
<td>30 µW</td>
<td>30 µW</td>
<td>3 µW</td>
</tr>
</tbody>
</table>

- **In vivo rat testing:** Once the feature extraction chip is redesigned to improve performance it can be tested in vivo with the rats using the test plan outline in Section 4.1.2. The test plan has been fully demonstrated up until the in vivo testing and the TDT in vivo testing code has been written and tested.
CHAPTER 5
SINGLE-SCALE SPIKE DETECTOR

UF’s third approach to neural data bandwidth reduction is the most dramatic and requires an implantable spike detector. For implanted circuitry an analog implementation is advantageous over a digital implementation because it has a much lower power consumption and it can be more compact in size. Thus, the spike detection algorithm chosen was limited to one that was amenable to an analog circuit implementation. Taking inspiration from Smith’s auditory onset detection scheme [59] a single-scale spike detection algorithm based on filtering was developed. Some of the single-scale spike detector work presented in this chapter has been previously published [49].

5.1 Algorithm

The basic operating principal of the single-scale spike detector is to use the thresholded difference of two low-pass filters to enhance the spike and stabilize the baseline. One filter has a higher cutoff frequency to remove high frequency noise and the other has a lower cutoff frequency to create a local average. When the difference between the signal and the local average rises above a threshold, a spike is detected. This method is robust to changes in the noise level as well as DC offsets, both of which are common for long term neural recordings. The basic algorithm blocks are shown in Figure 5-1 along with examples of the signal at every stage. Low pass filters are known to have a simple, low-power, and small area implementation in analog using the subthreshold region of operation [60, 61]. More details on the circuitry are provided later in Section 5.3.

5.2 Matlab Simulations

To analyze the algorithm’s performance prior to circuit design Matlab simulations with data from in vivo neural recordings were used. First the data will be described and then the detector’s performance results will be shown.

5.2.1 Data

High SNR neural recordings, sampled at 20 KHz, were used to increase the confidence of the ground truth spikes times determined from the data set. Then, white Gaussian
noise was added to give the detection problem a more realistic SNR level. A slowing
varying 1 Hz, 10 mV amplitude sinusoid was also added to the signal to simulate the
slowly varying DC offset. Figure 5-2 A) shows the original neural data waveform and B)
shows the 0 dB SNR waveform with an offset. SNR was calculated for the signal in terms
of power using an averaged spike shape.

5.2.2 Receiver Operating Characteristics (ROC) Curves

Receiver operating characteristic (ROC) curves are typically used to quantify the
performance of detection algorithms across the full range of thresholds [62]. ROC curves
plot the probability of a correct detection (also known as a hit) versus the probability
of a false detection (also known as a false alarm). There is always a trade-off between
the optimal detection of all the spikes and the erroneous detection of noise as a spike.
This spike detection problem also requires spike time estimation, meaning that the
performance curve could lie below the chance line for the detection problem. A detection
was considered correct if it occurred within 300 $\mu$s of the actual spike time. The ROC
Figure 5-2. Data used for simulations. A) Original waveform. B) 0 dB signal to noise ratio (SNR) waveform with offset. Column two is zoomed in from column one.
Figure 5-3. Plot of receiver operating characteristic (ROC) curves, 0dB SNR.

curves in Figure 5-3 shows that as a larger percentage of spikes are detected more noise will be falsely detected as a spike (also known as a false alarm). The optimal curve would start with no detections and no false alarms and go straight to 100% correct detection with no false alarms. The ratio of correct detections to incorrect detections can be set to the desired operating point on the ROC curve by choosing the corresponding threshold level.

To determine the desired circuit cut-off frequencies for the spike detector, ROC curves were constructed from nested cut-off frequency iterations around typical spike frequencies 100 Hz – 6 KHz [4]. The circuit’s cut-off frequencies were chosen with the minimum number of false alarms at 90% correct detection to be 1.4 KHz and 5.3 KHz.

Once the cutoff frequencies were selected, the algorithm was tested using the in vivo recordings described in Section 5.2.1. The single-scale detection method was compared to the threshold method without any filtering at 0dB SNR with the results shown in Figure 5-3. For comparison purposes the two methods were examined at their 90% correct detection operating point. The single-scale method outperformed the amplitude threshold without filtering method by over 30dB in terms of false alarm rate. Because spikes are
sparse in neural data the probability of a false alarm needs to be a fraction of a percent not to swamp the number of correct detections.

The data used has an average spiking rate of 76 Hz so during one second of data at 90% correct detections there should be 68 correct detections out of 76. At 0 dB the single-scale method had a $2 \times 10^{-4}$ probability of a false alarm, 4 false detections per second. The incorrect detection probability for the single-scale detector was 6%. For the amplitude threshold method there were 230 false detections per second, so its incorrect detection percentage is much greater at 76%.

The single-scale detection method consistently outperformed the amplitude threshold method over varying SNR values. Once the SNR became too low, $-2 \, dB$, neither method performed well. Here the single-scale method degraded to 25% incorrect detections and the amplitude threshold method was extremely poor at 82% incorrect detections.

Second-order filters were simulated for the single-scale spike detection circuit but their performance over first-order filters was negligible. Since second-order filters require additional chip area and power without noticeable performance improvement, they were not investigated further.

Analysis of the Matlab simulation results showed that at 90% correct detection almost all of the false alarms came from noise riding on the second peak of the action potential. Spike-like noise over other parts of the signal was only detected 2% of the time a false alarm occurred. The major portion of false alarms could be reduced by blinding the detector for a short period after it detects a spike. The trade-off to this would be the detector losing resolution between spikes. The amount of time the detector is blinded equals the minimum time required between spikes for detection.

Since the amplitude threshold is a subset of the single-scale method’s circuitry, it will consume less power. However, its lack of robustness to low SNR and slowly varying DC offsets hinders its performance for BMI devices. Matlab simulations with real neural recordings showed that with 90% correct detections at 5dB SNR the single-scale method
outperformed the amplitude thresholding method with 1% incorrect detections versus the 71% incorrect detections respectively. This performance continued for lower SNR such as 0 db with the single-scale detector having only a 6% incorrect detection rate while that of the amplitude threshold method was 76%. Though the results of computationally intensive methods such as matched filtering and template matching were not examined, they are sure to provide better results given enough information is known about the signal. However, even if enough information about the signal was known, the power consumption of matched filtering and template matching and their required supervision to adjust parameters as spike shapes and noise change over time prohibits implantation.

5.3 Circuit Design

The three major factors in this circuit design relate to the need for implantation and are low-power, small area, and robustness for the desired computation. Figure 5-4 shows the overall circuit block diagram for the single-scale spike detector. Each of the blocks will be briefly discussed. An operational transconductance amplifier (OTA) is configured as a follower integrator for the first-order low-pass filters. The OTAs are run in the subthreshold region to reduce power [60, 61] and to allow the capacitors to be small enough to fit on chip. The bias voltages are set off chip to enable adjustment of the cutoff frequencies after fabrication.

The desired cut-off frequencies for the two filters were found to be 1.4 KHz and 5.3 KHz from Matlab simulations described in Section 5.2.2. With the OTAs’ bias voltages set for a transconductance, $g_m$, of 150 nA/V and desired cutoff frequencies of 1.4 KHz and 5.3 KHz for the low-pass filters in Figure 5-4 the corresponding capacitance values are $C_1 = 22.5 \text{ } pF$ and $C_2 = 4.9 \text{ } pF$ from Equation. 5–1.

$$C = \frac{g_m}{2\pi f_c}$$  (5–1)
Neural spikes can vary in width from 0.4 ms to 3 ms \([5, 6]\) depending on the species and brain area so the cut-off frequencies are set to remove all of the noise outside the spike frequency ranges for the particular application.

After the signal has been filtered, the difference of the two filtered signals is taken using an OTA. The output is then thresholded with current, which is set with \(V_{\text{thresh}}\). For complete unsupervised operation an automatic method for setting the threshold would be needed. This thresholded signal is sent through two inverters to ensure a binary output decision.

*Cadence SpectreS* simulations show the circuit consumes an average of 1 \(\mu W\) of power. The single-scale spike detector chip was fabricated using AMI 0.5 \(\mu m\) CMOS technology. The chip is a 1.5 \(mm \times 1.5\) \(mm\) 40-pin DIP with 253 \(\mu m \times 223\) \(\mu m\) of circuit area. The layout is shown in Figure 5-5.

### 5.4 Chip Results

Due to the difficulty of testing the chip with real neural data, the chip was first tested with two basic signal generators to crudely approximate neural data. A square wave was used to mimic the spike and a high frequency sine wave was used to simulate noise on the
Figure 5-5. Layout for onset spike detector chip.
signal. The input signal was based on three characteristics of real neurons: spike width, time between spikes, and amplitude.

Because the circuit detects the onset of the spike, the effective spike width is the width of the first rise in the action potential. With infinite SNR, this would mean approximately half the action potential time, but as SNR degrades it reduces. The chips functionality was tested with pulse widths of 100–400 µs.

The second signal characteristic is the time between spikes. Individual neurons have a refractory period, which sets a minimum time between spikes. But, electrodes often record from more four to six neurons so the refractory period is not a determining factor in the minimum time between spikes. It is then optimal to detect a spike as close the previous one as possible since superimposed spikes can not be discriminated between as is the case with most spike detectors including this one. The filter delays and the time to charge the load capacitance are the two factors which determine the minimum detectable time between spikes for the circuit.

Amplitude is the third characteristic of the input signal. Extracellular neural signals have peak-to-peak amplitudes of 50 µV–500 µV [3]. This small signal must first be amplified to give a larger voltage swing for the analog spike detection circuit to be more accurate. Today, low noise, low power neural amplifiers, such as the UF bioamplifier [25], can achieve a gain of up to 100, so the input signal amplitude ranges between 5 and 50 mV.

The result of a 35 mV square wave with a 125 µs pulse width at 25% duty cycle combined with a 15 mV high frequency sine wave (to mimic neural noise) is shown in Figure 5-6 as the bottom waveform. It shows that 10 µs after the input spikes the output goes high for a short period.

The chip was tested over a wide range of input signal characteristics loosely patterned after neural data. The threshold voltage allows the chip to be adjusted to change the false alarm penalty, and correspondingly its probability of correct detection, in accordance with
Figure 5-6. Singe-scale spike detector chip results with signal generator pulse waveform as input. Ch2, top waveform, is the input signal from a signal generator and Ch3, bottom waveform, is the output signal.
its ROC curve. The chip detected the crude neural signal representation down to about 25 mV with 15 mV of noise. The minimum detectable amplitude is not a fixed value because it depends on the SNR and the duration of the spike. Jitter measurements were not performed because the output jitter is negligible with the standard 100 ms bins used to decrease the sparseness of neural data before further processing.

These initial chip testing results are in no way exhaustive and the crude signal approximations used for these tests is not an adequate performance measure. Bionic’s 128-Channel Neural Signal Simulator was used as a more realistic test input for the spike detector chip. The neural simulator outputs a repeated 11 second pattern of spikes from three different action potentials with amplitudes of 100 µV–150 µV and a width of 1 ms. The interspike interval is 1 s for 10 s and then the interspike interval reduces to 10 ms for one second of burst firing. None of the spikes were superimposed in the output. The UF bioamplifier [25], with a gain of 100 a low cutoff frequency of 0.3 Hz and a high cutoff frequency of 5.4 KHz, was used to amplify the neural simulator’s output, as would be used in a BMI system.

The spike detector was able to detect 99% of the spikes without any false alarm. This is approximately one missed detection per second during peak neural firings. By decreasing the threshold slightly the detector reached a 100% detection rate but a few spikes were detected twice creating false alarms. Blinding the detector for a short period after every detection would eliminate this problem, but would also keep the detector from detecting two spikes closer than the blinding period. Another option would be to only use the positive half of the signal with a half-wave rectifier. The disadvantage of this method is if the electrode recording is referenced to another electrode the peak-to-peak amplitude of a spike that first goes negative and then sharply rises is decreased by about half. Two 50 ms examples of the spike detector’s output from the amplified neural simulator signal are shown in Figs. 5-7 and 5-8.
This is a preliminary performance characterization. To properly characterize the spike detector chip with a longer waveform, more realistic SNR, and with different width spikes. The chip was tested in vivo with a rat at the UF McKnight Brain Institute, but do to the recording equipment used at the time only short time segments were recorded. Using the Tucker-Davis Technologies (TDT) [63] parallel recording test setup as described in Section 4.5.2, the chip could now be tested with more extensive statistical tests of the system performance and direct comparison top an existing system. This was not done as at the time the recording system was in place the feature extractor algorithm was being worked on and deems more important to focus our resources on.

Figure 5-7. Single-scale spike detector chip results. A) Amplified neurosimulator waveform input to the chip. B) Chip’s output.
Figure 5-8. Single-scale spike detector chip results. A) Amplified neurosimulator waveform input to the chip. B) Chip’s output.
CHAPTER 6
MULTI-SCALE SPIKE DETECTOR

The multi-scale spike detector extends the single-scale spike detection method, presented in the previous chapter, to multiple scales to allow the detection of spikes with a wider range of widths without sacrificing performance. The key idea is to implement wavelet decomposition and improve spike detection by independently controlling thresholds for each scale. Each thresholded scale can then be combined to provide a single output indicating a spike occurrence. Another option is to use each thresholded scale, which corresponds to a small range of spike widths, as a feature for spike sorting. Width is one of the most important spike sorting features with the other being amplitude, which could easily be added to the detector using a simple peak detector as attempted in Horiuchi’s paper [36]. Some of the multi-scale spike detector work presented in this chapter has been previously published [50].

6.1 Optimal Threshold

From basic detection theory the optimal threshold for detection is known to vary with the SNR of the signal. For neural signals the SNR not only varies from channel to channel and over time as the noise varies but it varies within each channel dependent on the width of the spike to detect. Figure 6-1 shows how the SNR for a signal with the same noise is affected when the spike width changes. Since typical neural spikes vary in width from 0.4 ms to 3 ms [5, 6], the multiresolution approach is essential to the spike detector’s performance because it enables a separate threshold to be set for each frequency band, which allows for a equal ratio of type I errors (false alarms) to type II errors (missed detections) across a wide variety of spike widths.

6.2 Algorithm

Multiresolution spike detection requires localization in both the time and frequency domain which can be achieved with wavelets. Traditional wavelet circuits consume too much power for implantation. Wavelets can be formed using band bandpass filters where the ratio between the center frequency and the bandwidth for each filter remains constant.
(also know as constant Q) where the filters span the frequency space. This wavelet method was chosen because an ultra-low power analog implementation already exits, the multi-scale gamma filter. It is illustrated in Figure 6-2 and the circuit details will be explained in Section 6.4.

After the signal is decomposed into frequency bands each band is thresholded to determine the presence of a spike at that scale. The outputs of all the scales are ORed together after appropriate compensation for their varying delays. If this spike detector was to be used in an application requiring spike sorting, the individual scale output could be transmitted to send the spike width feature and a simple peak detector circuit [36] could be implemented to send the amplitude feature of the spike. This would allow for spike sorting outside the subjects body where the constraints on circuit size and power are not as stringent.

6.3 Matlab Simulations

6.3.1 Scale Combination Method

The main idea to combine the output at each scale is to OR the binary outputs of each individual scale after appropriate compensation for their varying delays. The delay at

Figure 6-1. SNR versus spike width.
each scale was determined from the step response delay for its bandpass filter as shown in Figure 6-3. The combination starts with the lowest frequency scale being shifted in time to account for the additional delay from the second lowest frequency scale. Then, the two scales are ORed. It is important to ensure that a spike will not be detected twice due to the noise amplitude varying between frequency scales. In order to eliminate the spurious detections, a small minimum distance criterion is enforced in the algorithm at each scale. If this minimum distance is not met, the latter spike is removed. Then, the new combined scale is combined with the next lowest frequency scale. This continues until there is only one output scale left.

Figure 6-4 illustrates the output from each scale for several spike widths and their final combined output. Figure 6-4 A) shows the input which contains spike widths within the range of the typical 0.3 ms–3 ms values. The beginning of the waveform starts with the widest spike, 3 ms, and goes to the narrowest, 0.3 ms. The widths are: 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1, 1.5, 2, 2.5 and 3 ms wide. The next four plots, Figure 6-4 B), C), D) and E) show the bandpass filter outputs corresponding to 6K–3.4K Hz, 3.4K–1.9K Hz, 1.9K–1.1K Hz, and 1.1K–600 Hz respectively and the detected spikes for each scale.
with circles. It can be seen that the narrow spikes are better detected with the high frequency bands and the wider spikes are better detected with the low frequency bands as is expected. Note that while for this very high SNR input the middle frequency band could detect all of the spikes, for more typical SNR signals it would not be possible to detect all of the spikes on a single scale without having many false alarms. Figure 6-4 F) shows the final chip output, the combination of each scales output.

6.3.2 Threshold Scaling from One Scale to Others

Setting thresholds by hand is cumbersome especially for a large number of channels. For the multi-scale algorithm each scale has a different threshold independently set so this may prohibit finding the optimal threshold levels for the algorithm to work its best. To solve this problem and to allow a 2-D ROC curve to be constructed the thresholds are related to one another so that setting one threshold automatically sets the rest at the same detection ratio.
Figure 6-4. Filtered scales and detected spikes on each scale and combined output. A) The concatenated input signal with spikes widths starting at 3 ms and going as small as 0.3 ms. B) 6K–3.4K Hz bandpass filtered signal with the detected spikes on this scale shown with circles. C) 3.4K–1.9K Hz bandpass filtered signal with the detected spikes on this scale shown with circles. D) 1.9K–1.1K Hz bandpass filtered signal with the detected spikes on this scale shown with circles. E) 1.1K–600 Hz bandpass filtered signal with the detected spikes on this scale shown with circles. F) Combined output from each scale.
From basic detection theory in ideal cases where the probability distribution functions (pdfs) of the signals are known, Gaussian, and share the same variance, the Bayes’ detector Equation. 6–1 provides the optimal threshold for the desired costs of false alarms \( (C_{10}) \), missed detection \( (C_{01}) \), and correct detections \( (C_{11} \text{ for the signal and noise and } C_{00} \text{ for noise}) \).

\[
\lambda = \sigma_0^2 \frac{\ln\left(\frac{P_0}{P_1}\right) \left| C_{10}C_{01} \right|}{\mu_1 - \mu_0} + \frac{\mu_1 + \mu_0}{2} \tag{6–1}
\]

If all of the costs are weighted equally, the cost term becomes one and doesn’t affect the equation. In this equation \( \sigma^2 \) represents the variance, \( \mu \) the mean, and \( P_i \) is the probability of noise \( (P_0) \) or the probability of spike plus noise \( (P_1) \). A visual representation of Bayes’ detector equation is shown in Figure 6-5. The distribution on the left is the noise and the distribution on the right is the signal (spike plus noise). \( y_t \) is the threshold or \( \lambda \) from Equation. 6–1. The dark grey colored region is the false alarms (type I errors) and the light grey shaded region is the missed detections (type II errors). If the costs of all errors and correct detections were weighted equally then for this example the threshold would be set at 0.5.

Using Bayes’ equation the individual thresholds of each scale are related to one another so that setting one threshold automatically sets the rest. We are working to make the threshold relationships from one scale to another more robust since some parameters, such as the probabilities and variance, are preset based on past performance but in reality they vary over time and this is included in the future works section. One option for setting the threshold on the one scale is to allow raw segments of each channel to be periodically transmitted out so that a human can adjust the parameters for the circuit periodically and send the new parameters back to the circuit. This could mean that each channel takes turns having its raw data transmitted while the rest of the channels only send their detected spikes so the overall transmission bandwidth limitations are still met.
6.3.3 Data

The algorithm was tested on neural recordings from male Sprague-Dawley rats chronically implanted with 50 µm polyimide insulated tungsten microwire electrode arrays in layer V of the forelimb region of the primary motor cortex. The data was sampled at 25 KHz and bandpass filtered between 0.5 and 12 KHz using hardware from Tucker-Davis Technologies. Action potential widths ranged from 0.4 ms–1.2 ms with amplitudes as high as 137 µV. High SNR recordings were chosen to increase the confidence of the ground truth spikes times.

Ground truths were labelled by a human expert. The method of marking spikes was to first parse out data segments with a possible spike. This was done using a conservative (low) threshold and extracting segments around the threshold crossing from the waveform. Then, these segments were examined and only those which actually contained a spike were kept and labelled in the spike time file. This means there were few false alarms but spikes with large negative peaks could have been omitted if their second phase did not cross the
Figure 6-6. Figure 6-4 with the threshold, shown with a dashed horizontal line, set for plot B) and automatically set for the remaining bands.
positive threshold. We determined that in one of the data sets over the first five seconds of data 18 spikes fit this criteria. When characterizing a detector's performance, the false alarms should be analyzed to see if any of them look like spikes and are perhaps ones the human expert missed or if the detector is truly detecting noise as a spike.

To increase the ground truth accuracy and remove any bias of an individual expert several experts should mark the ground truths with their averaged results becoming the ground truth. Marking spike times is a tedious task though, so it is difficult to enlist human experts to mark a lot of data. For the parsing out of possible spikes a second negative threshold could be used in addition to the positive one to decrease the number of missed detections in the ground truths.

To make this high SNR recording more similar to the typical recordings, white Gaussian noise was added. A slowly varying 0.1 Hz, 10 µV amplitude sinusoid was also added to the signal to simulate DC offsets. Figure 6-7 A) shows the original neural data waveform and B) shows the 0 dB SNR waveform with an offset. SNR was calculated for the signal in terms of power using an averaged spike shape.

6.3.4 Receiver Operating Characteristics (ROC) Curves

Receiver operating characteristic (ROC) curves are typically used to quantify the performance of detection algorithms [62]. There is always a trade-off between the optimal detection of all the spikes and the erroneous detection of noise as a spike. This detection problem also requires spike time estimation. A detection was considered correct if it occurred within 500 µs of the actual spike time. The ratio of correct detections to incorrect detections can be set to the desired operating point on the ROC by choosing the corresponding threshold level.

The multi-scale detection method was compared to the single-scale method and the simple amplitude threshold at 0 dB SNR over 120 s of neural data with the results shown in Figure 6-8. The single scale method was shown to outperform the amplitude threshold in Chapter 2, so this discussion will only compare the multi-scale detector
Figure 6-7. Data set used for simulations. A) Original waveform B) 0 dB SNR waveform with offset. Column two is zoomed in from column one.

to the single-scale detector. For comparison purposes the methods were examined at their 90% correct detection operating point. The multi-scale method only had 15 false alarms per second, while the single-scale detector had 112. Thus, the multi-scale detector outperformed the single-scale detector by over 15dB in terms of false alarms for a 90% correct detection rate. Because spikes are sparse in neural data the probability of a false alarm needs to be a fraction of a percent not to swamp the number of correct detections.

The data used has an average spiking rate of 63 Hz so during one second of data at 90% correct detections there should be about 57 correct detections out of 63. The multi-scale algorithm has a similar performance boost over various SNRs down to -5 dB but below this SNR level none of the methods tested performed well enough to be used in a BMI system. For larger SNR values the multi-scale algorithm continued to outperform the single-scale method. An analysis of the false alarms detected by the multi-scale method at 90% correct detections showed that almost one-third of the false alarms were
from spikes whose negative side was larger than the threshold but whose positive side was smaller than the threshold. This means that ground truths, such as our own, done by examining events that surpass a single threshold produce a bias and may be misleading if not examined in the performance analysis.

6.4 Circuit Design

The continuous time wavelet decomposition is achieved with a multi-scale gamma filter [64] (cascade of low-pass filters) as shown in Figure 6-9 by taking the difference of adjacent taps, $X_k - X_{k-1}$. To achieve a wide-range of cut-off frequencies, a resistive line is connected along the bias controls of each low-pass filter. With the OTAs operated in the subthreshold region, this linear voltage drop across the resistive line provides an exponential change in the bias currents, which in turn proportionally varies the cutoff frequencies. This allows each filter to be constant Q, meaning the ratio between the center frequency of the filter and the bandwidth remains constant, which provides localization in
both the time and frequency domains. The formal proof that the difference of neighboring taps of the multi-scale gamma filter is a wavelet is included in APPENDIX A.

The transfer function of the k\textsuperscript{th} stage of the multi-scale gamma filter is given by

$$H_k(s) = \frac{1}{a^k \tau s + 1}$$ \hspace{1cm} (6–2)

where $a$ is a preset attenuation factor given by Equation 6–3.

$$a = \left( \frac{\tau_{low}}{\tau_{high}} \right)^{\text{numtaps} - 1}$$ \hspace{1cm} (6–3)

Currently, a five-tap gamma filter, Figure 6-9, with 10 pF capacitors is used to provide four frequency scales. The difference of each set of neighboring taps, $X_k - X_{k-1}$, form bandpass filters and are thresholded to determine the presence of a spike at each scale. If the outputs of all the scales should be combined they can be ORed together after appropriate compensation for their varying delays; however, circuitry for this has not been designed. If this spike detector was to be used in an application requiring spike sorting, each scale’s output would be transmitted to send the spike width feature and a simple peak detector circuit [36] could be implemented to send the amplitude feature of the spike.

_Cadence SpectreS_ simulations show the circuit consumes about 3 $\mu W$ of power. The multi-scale spike detector circuit was fabricated using AMI 0.5 $\mu m$ CMOS technology. The chip is a 1.5 $mm \times 1.5 mm$ 40-pin DIP with 461 $\mu m \times 221 \mu m$ of circuit area with the layout shown in Figure 6-10.
Figure 6-10. Multi-scale spike detector chip layout.
6.5 Chip Results

Preliminary testing was done first with the signal generator and then with the neural simulator as was done for the single-scale chip. In this case these preliminary tests served to show the chips were functioning, but they could not test the multi-scale nature of the chip because the neural simulator only produces spike with the same width. In vivo testing or use of an arbitrary waveform generator and in vivo data file will be required to obtain a signal with multiple spike widths. Preliminary in vivo results cannot be obtained using an oscilloscope as the oscilloscopes we have only record from four channels and we need to record the input to the chip as well as its four outputs. The setup to record each scale of the chip’s output from in vivo testing with rats at the UF McKnight Brain Institute allows the chip to be properly characterized. The setup includes parallel recordings of both the TDT system and the multi-scale spike detector’s results to allow direct comparisons between the two systems. This testing was foregone as a novel pulse-based feature extractor (discussed in Chapters 2-4) was being developed and the decision was made to focus all of the efforts on the feature extractor.
CHAPTER 7
CONCLUSIONS

7.1 Overall Conclusions

UF’s pulse-based feature extractor dramatically reduces the required wireless bandwidth while remaining low-power, 100 $\mu$W, without sacrificing the ability to spike sort. Simulation results with known-ground truths show the pulse-based feature extractor preserves enough information to allow for similar spike sorting results as Spike2 with over two-orders of magnitude less bandwidth, 455 pulses/s compared to 300 Kbps. This is over one-order of magnitude less bandwidth than the biphasic encoding for reconstruction. Further bandwidth reductions can be obtained if the increased sorting error can be tolerated. The pulse-based feature extractor shows promising results that it can solve the data reduction problem while still allowing for spike sorting. Further data bandwidth reduction with an adaptive threshold or refractory period are yet to be studied.

The feature extractor circuit was fabricated using AMI 0.5 $\mu$m CMOS technology. The chip is a 1.5 mm $\times$ 1.5 mm 40-pin DIP with 504 $\mu$m $\times$ 356 $\mu$m of circuit area. It consumes 30 $\mu$W of power and the chips performed close to the simulation results even with the noise from the feedback from the digital output to the analog input of the circuit, showing the features are robust to noise. The chip shows promising results towards suitability for in vivo neural recordings.

A fully-integrated ultra low-power multi-scale neural spike detector has also been demonstrated. It implements a continuous wavelet transform using ultra low-power circuitry to allow for implantation. It only consumes 3 $\mu$W of power and when it is used in conjunction with the UF bioamplifier it will consume even less power since the amplifier lowpass filters the signal at 5.4 KHz. This allows for one less cascaded filter in the multi-scale gamma filter. The multi-scale spike detector circuit was fabricated using AMI 0.5 $\mu$m CMOS technology. The chip is a 1.5 mm $\times$ 1.5 mm 40-pin DIP with 461 $\mu$m $\times$ 221 $\mu$m of circuit area. The multi-scale spike detector’s performance was characterized against simpler detection methods through ROC curves as shown in Figure
6-8. The figure shows that the multi-scale spike detector outperforms the amplitude thresholding method, the simplest spike detector.

The power required for amplitude thresholding is less than for the multi-scale detector since it is a subset of the multi-scale detector. The power required for the multi-scale spike detector, 3 $\mu W$, is still negligible though compared to the power required for the rest of the neural recording system since today’s neural amplifiers alone consume around 80 $\mu W$ of power. Therefore, the power saved by using an amplitude threshold for spike detection is negated by its larger number of false alarms to obtain the same detection rate. Similarly, the multi-scale spike detector outperforms the single-scale method and though it uses 2 $\mu W$ more power, at the system level the extra power required is negligible.

Theoretical analysis, simulations, and chip measurement results show that the multi-scale spike detector is a good compromise between power, transmission bandwidth, area, and performance for an implantable device.

7.2 Contribution Summary

A novel approach was taken in designing a front-end neural data bandwidth reducer by optimizing the complete system apposed to just optimizing the front-end. Because of this, a drastic data reduction, over two-orders of magnitudes less compared to existing ADC system, was achieved using a pulse-based feature extractor while still preserving competitive spike sorting performance compared to commercial products. The feature extractor does not require explicit spike detection as most feature extractors do. The novel pulse-based feature extractor algorithm and low-power analog circuit were designed, analyzed, simulated, and tested. The analog feature extractor circuit is low power $< 30 \mu W$.

A novel pulse train based sorting algorithm was also developed, analyzed, and tested to enable a system performance metric for the feature extractor with spike sorting error.
For applications that do not require spike sorting a multi-scale spike detector circuit was designed, analyzed, simulated, and tested. The power is ultra-low at 3 \( \mu \text{W} \) and the detection performance is improved over the simpler single scale spike detector.
APPENDIX A

PROOF: THE DIFFERENCE OF THE MULTI-SCALE GAMMA FILTER’S ADJACENT TAPS IMPLEMENT A CONTINUOUS WAVELET DECOMPOSITION

The multi-scale gamma filter circuit was previously stated in Chapter 3 to be a continuous wavelet but without a formal proof. The formal proof follows: Wavelets localize well in both the time and frequency domain. One way to define a mother wavelet is with a bandpass filter [65] which can be implemented as the difference of two lowpass filters as in Equation A–1 with Equation A–2 being the transfer function of the k\textsuperscript{th} stage of a cascade of lowpass filters since the difference of adjacent filters will be a bandpass filter.

\[ \psi_k(\omega) = \Phi_{k+1}(\omega) - \Phi_k(\omega) \quad (A-1) \]

\[ \Phi_k(\omega) = \prod_{k=i}^{N} \frac{1}{a^k \tau_S + 1} \quad (A-2) \]

In our case this cascade of lowpass filters is the multi-scale gamma filter with the cutoff frequencies of each filter varying on a log scale when the operational transconductance amplifier’s (OTA) are run in the subthreshold region and their bias voltages varied linearly through the resistive line (see Figure A-1). This log variation of cut-off frequencies allows each filter to be constant Q, meaning the ratio between the center frequency of the filter and the spectrum width remains constant. Therefore, the desired localization in both the time and frequency domains is obtained. The equation for the continuous wavelet

![Multi-scale Gamma filter circuit](image_url)

Figure A-1. Multi-scale Gamma filter circuit.
The continuous wavelet transform (CWT) at scale $k$ is Equation A–3

$$ F\{W_k f\} = F\{S_{a^{k+1}} f\} - F\{S_{a^k} f\} \quad (A-3) $$

and the corresponding cascaded filter structure is shown in Figure A-2 [66].

To prove that the multi-scale gamma filter circuit is a wavelet the mother wavelet must meet the admissions condition, Equation A–4, which guarantees localization in both the time and frequency domain as well as the existence of an inversion formula for the continuous wavelet transform [65, 67]. If the number of frequency bands is sufficiently high, the gamma kernel constitutes a complete set in $L_2$ space [68] and is continuous so Equation A–4 simplifies to $\Psi(0) = 0$. For our case $\Psi_k(\omega)|_{\omega=0} = 0$, so the admissions condition holds and guarantees that the multi-scale gamma filter is a wavelet.

$$ \int_{-\infty}^{\infty} \frac{|\hat{\Psi}(\omega)|^2}{\omega} d\omega < \infty \quad (A-4) $$

While wavelet bases are not required to be orthogonal they often are to simplify the reconstruction of the signal. Though the gamma bases are linearly independent they are not orthogonal. The gamma filter is easy to implement in analog with low power consumption and small die area which are both very important to our project since this circuit will be implanted under the skin. Since signal reconstruction is not needed in spike...
detection the advantages of using the non-orthogonal wavelet outweigh the disadvantages for our application and the gamma kernel was used. If orthogonal wavelets were required the multi-scale Laguerre filter [66] could be used.

This completes the proof that the difference of multi-scale Gamma filter’s adjacent taps implement a continuous wavelet decomposition.
APPENDIX B
LEAKY INTEGRATE-AND-FIRE (LIF) T69K-AS CHIP PINOUT

1. Vss of Gm analog power (-2.5V)
2. Vcc of Gm analog power(-2.5V)
3. Same as 1
4. N/A
5. Same as 2
6. Ground (0V)
7. N/A
8. Reference input (Theoretically 0V)
9. N/A
10. Bias current of Gm (8uA)
11. Leaky current
12. N/A
13. Bias current of output buffer OTA (100nA)
14. N/A
15. Output of Gm, also the input of comparators
16. N/A
17. Input
18. N/A
19. Vdd of Pad (2.5V)
20. N/A
21. Vss of comparators digital power (-2.5V)
22. Vdd of comparators digital power (2.5V)
23. N/A
24. Bias current of comparator first stage (450nA)
25. Negative threshold voltage
26. Positive threshold voltage  
27. N/A  
28. Bias current of comparator second stage (1μA)  
29. N/A  
30. Vss of Pad (-2.5V)  
31. XOR of pulses  
32. Complimentary XOR output  
33. Refractory current  
34. Negative direction pulse  
35. N/A  
36. Reset voltage (0V)  
37. N/A  
38. Positive direction pulse  
39. Vdd digital power (2.5V)  
40. Vss digital power (-2.5V)
REFERENCES


BIOGRAPHICAL SKETCH

Christy L. Rogers was born in Orange Park, FL, on January 7, 1980 to the loving parents Regina and Greg Rogers. She has one younger brother, Shawn Rogers. Christy will marry her dream guy, Xiao She, in June 2007. Christy received the B.S. degree in electrical engineering Summa Cum Laude and with University Honors from the University of North Florida (UNF), Jacksonville, FL, in 2002.

Since 2002, Christy has been a research assistant in the Computational NeuroEngineering Laboratory (CNEL) at the University of Florida working under Dr. John Harris on the Brain Machine Interface Project. Christy is a 2003 recipient of a National Science Foundation Graduate Research Fellowship and also received the UF Presidential Fellowship. Her research interests are biologically inspired analog signal processing and mixed-signal integrated circuit design. Specifically, her interests lie in developing an ultra-low power implant for spike feature extraction in neural recording applications.

Christy received the M.S. degree in electrical engineering from the University of Florida (UF), Gainesville, FL in 2004 and her Ph.D. degree in electrical engineering in May 2007. She will work at Texas Instruments in Dallas, TX with their medical device group.