J. Neural Eng. **4** (2007) 309–321 doi:10.1088/1741-2560/4/3/016

A single-chip signal processing and telemetry engine for an implantable 96-channel neural data acquisition system

Michael Rizk, Iyad Obeid[1], Stephen H Callender and Patrick D Wolf

Department of Biomedical Engineering, Duke University, Durham, NC 27708, USA

E-mail: mr38@duke.edu

Received 15 March 2007
Accepted for publication 25 June 2007
Published 20 July 2007
Online at stacks.iop.org/JNE/4/309

Abstract

A fully implantable neural data acquisition system is a key component of a clinically viable cortical brain–machine interface. We present the design and implementation of a single-chip device that serves the processing needs of such a system. Our device processes 96 channels of multi-unit neural data and performs all digital processing necessary for bidirectional wireless communication. The implementation utilizes a single programmable logic device that is responsible for performing data reduction on the 96 channels of neural data, providing a bidirectional telemetry interface to a transceiver and performing command interpretation and system supervision. The device takes as input neural data sampled at 31.25 kHz and outputs a line-encoded serial bitstream containing the information to be transmitted by the transceiver. Data can be output in one of the following four modes: (1) streaming uncompressed data from a single channel, (2) extracted spike waveforms from any subset of the 96 channels, (3) 1 ms bincounts for each channel or (4) streaming data along with extracted spikes from a single channel. The device can output up to 2000 extracted spikes per second with latencies suitable for a brain–machine interface application. This device provides all of the digital processing components required by a fully implantable system.

1. Introduction

Today’s neural data acquisition systems have been effective in simultaneously recording extracellular potentials from hundreds of neurons [1–3]. Such systems for performing ensemble recordings are crucial components of brain–machine interfaces (BMIs). BMIs have been shown to enable monkeys to control a cursor and perform reaching and grasping tasks based on their measured neural activity [4–6]. These devices provide a possible means of restoring motor function and communication to individuals suffering from nerve damage due to spinal cord injury, stroke or neurodegenerative diseases. Recent work has shown the feasibility of human use of a BMI to allow a tetraplegic individual to control a computer cursor, a prosthetic hand and a robotic limb [7].

Neural data acquisition systems have obstacles that must be overcome for clinical applications of these systems to truly become feasible. Typically, systems that obtain high-resolution recordings of neural signals involve a percutaneous (through the skin) connection between electrodes and wires that travel to various pieces of electronic equipment [6, 7]. This method of acquiring neural signals is accompanied by limitations and possible complications that can be undesirable within a research setting and are definitely problematic within the context of a clinical human BMI. The chronic break in the skin increases the likelihood of infection while the tethering effect of the wires provides an unnatural restraint on motion. Additional restraints are often employed to prevent the animal from damaging or detaching cables or other equipment. While this setup may be suitable for a research environment, it is not appropriate for long-term human use [8]. Thus, especially for clinical applications like a BMI, current neural data acquisition systems are not appropriate. These limitations point to the

[1] Present address: Department of Electrical and Computer Engineering, Temple University, Philadelphia, PA 19122, USA.

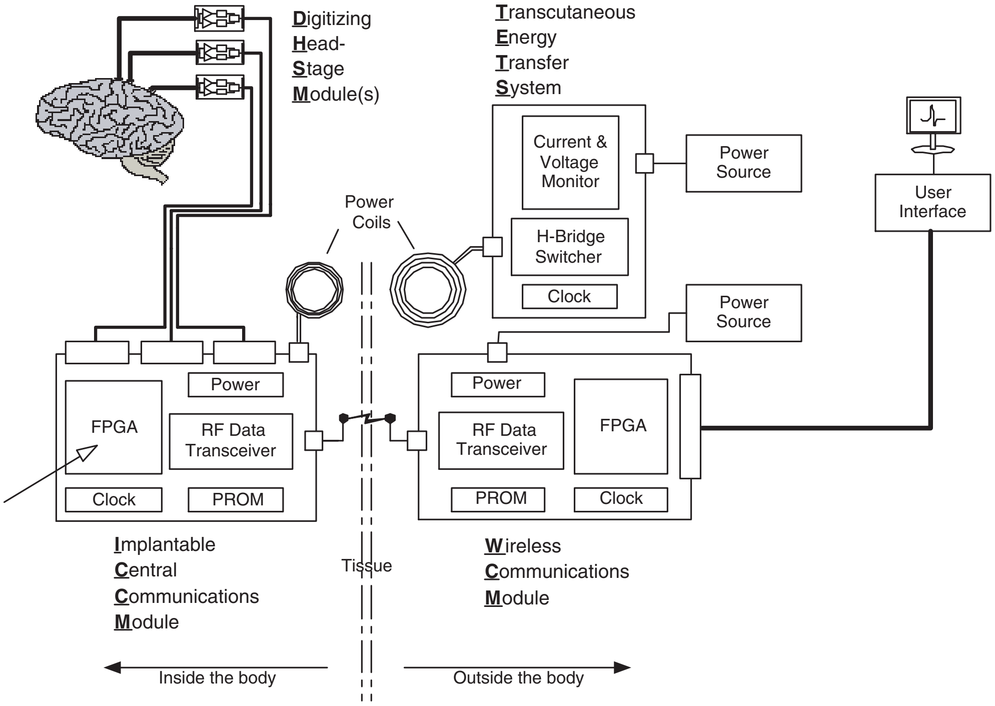

Figure 1. A block diagram of the neural data acquisition system. The implantable central communications module (ICCM) located in the bottom left of the figure corresponds to the board presented in figure 5. The FPGA located on this module (indicated by the open arrow) is the focus of this paper. The ICCM receives neural data from the digitizing headstage modules (DHSMs) and receives commands from and sends data to the wireless communications module (WCM).

need for a fully implantable neural data acquisition system. In fact, the development of a fully implantable neural recording system is one of the major challenges that must be overcome in order for a clinical BMI to be successful [8].

A high channel-count neural data acquisition system collects a large amount of data, tens of megabits per second. One of the major challenges faced by a fully implantable system is the passing of large amounts of information acquired inside of the body to a device outside of the body in real time. A fully implantable system cannot communicate with the outside world using wires that pass through the skin. A means for wireless communication is required. The communication system should have reasonably high data rates (on the order of hundreds of kilobits per second) and the ability to transmit over significant distances (on the order of a meter) through a combination of air and biological tissue. In addition, a form of data reduction within the implantable system is critical to reduce the burden on the telemetry system. If data reduction is not included in the system, a telemetry system with a much higher data rate must be used.

We present a single-chip solution for performing data reduction on 96 channels of neural data and interfacing with a one megabit per second (Mbps) radio frequency (RF) transceiver. An earlier version of this chip was presented in [9]. The implementation of this solution has been completed and the device has been used in live animal recordings.

2. Design

2.1. Overview

The single-chip neural signal processing and telemetry engine has been implemented in a field programmable gate array (FPGA). This device is one component of a fully implantable neural data acquisition system. While this device is the focus of the work presented here, of necessity other components of the system must be introduced and briefly described. A block diagram of the overall system is presented in figure 1.

Our device interfaces with three digitizing headstage modules (DHSMs) and a commercial 1 Mbps RF transceiver (TR1100, RF Monolithics, Inc., Dallas, TX). Each DHSM amplifies, filters and digitizes 32 channels of neural data. The passband of the DHSM amplifiers extends from 500 Hz to 8 kHz. The neural data are sampled at 31.25 kHz with 12 bits of resolution. Six lines carry the digitized neural data from the three DHSMs to our device, with each line containing serialized time-multiplexed data for 16 channels. The programmable logic for processing the neural data and interfacing with the transceiver has been implemented in a Xilinx (San Jose, CA) Virtex-II XC2V1000 FPGA. This implementation has been performed using a combination of the VHDL and Verilog hardware description languages and consists of more than 50 distinct modules. The general organization of this processing is shown in figure 2. Data reduction is applied to each set of 16 channels independently

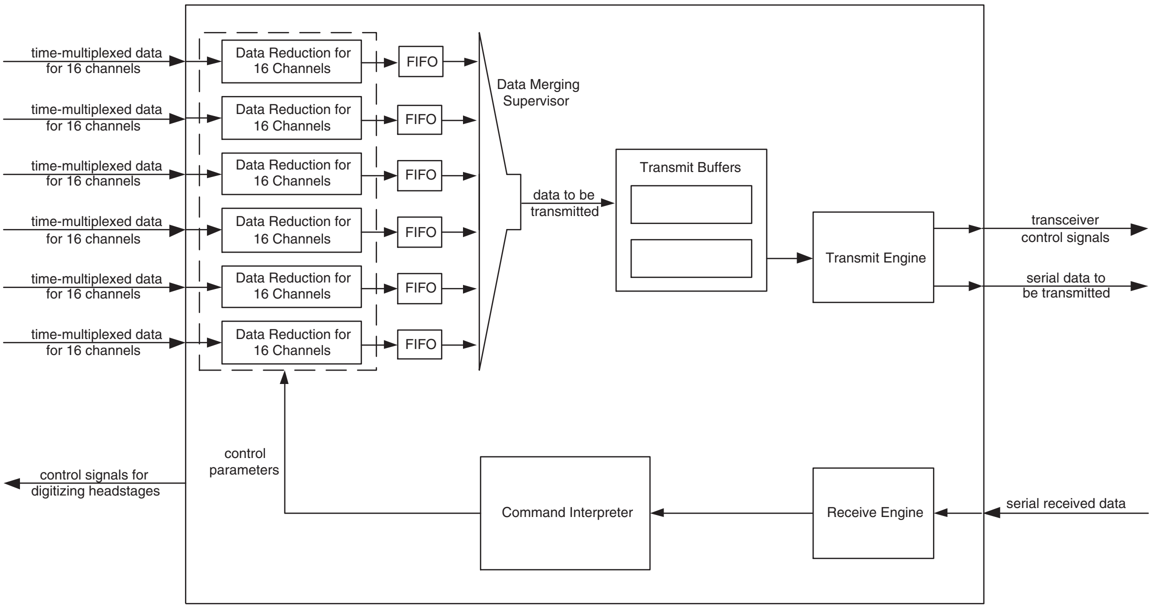
time-multiplexed data for 16 channels
Data Reduction for 16 Channels
FIFO
Data Merging Supervisor
data to be transmitted
Transmit Buffers
Transmit Engine
transceiver control signals
serial data to be transmitted
control parameters
control signals for digitizing headstages
Command Interpreter
Receive Engine
serial received data

Figure 2. A block diagram of the processing that is performed by our device. The two primary functions of our device are to perform data reduction on 96 channels of neural data and to provide a telemetry interface to a transceiver. Data reduction is performed in parallel on six sets of 16 channels of data that are provided by the digitizing headstages. Data from the six data reduction modules are temporarily buffered before being merged into a single datastream and stored in a transmit buffer. Two transmit buffers are present. During any given 50 ms period, one is being filled while the other is being emptied. The telemetry interface consists of a transmit engine, which processes the stream of output data that is to be transmitted, and a receive engine, which processes the stream of data sent to our device by an external command module. The command interpreter adjusts the parameters for data reduction based on the received commands and configuration information.

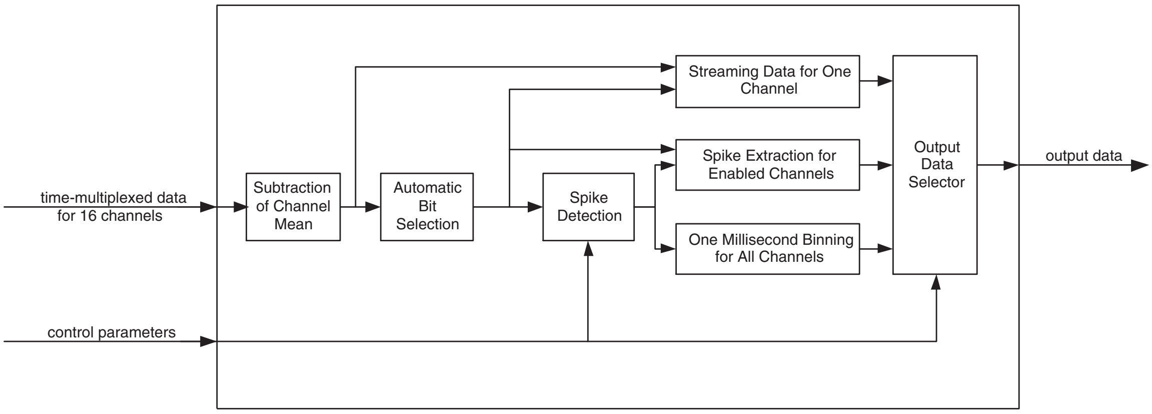
time-multiplexed data for 16 channels
Subtraction of Channel Mean
Automatic Bit Selection
Spike Detection
Streaming Data for One Channel
Spike Extraction for Enabled Channels
One Millisecond Binning for All Channels
Output Data Selector
output data
control parameters

Figure 3. Data reduction for one set of 16 channels. Data reduction includes automatic bit selection as well as the use of four different output modes. The output options are (1) streaming data from a single channel, (2) extracted waveforms for all enabled channels, (3) 1 ms bincounts for all channels and (4) streaming data and extracted spikes from a single channel.

and simultaneously. The output from the six data reduction modules is then merged together to form a single stream of data that can be sent to the transceiver.

Our device and the transceiver are contained in the implantable central communications module (ICCM). The ICCM receives commands from and sends data to the wireless communications module (WCM), which is located outside of the body and in turn communicates with a computer running a custom-designed graphical user interface. (Refer to figure 1.)

2.2. Data reduction

Data reduction is a key function of our device. Without data reduction, and assuming no overhead for telemetry, fewer than three full channels of data could be transmitted using a 1 Mbps transceiver

$$\frac{(10^6\ \text{bits s}^{-1})/(12\ \text{bits/sample})}{(31\,250\ \text{samples/s/channel})} = 2.67\ \text{channels}.$$

In order to effectively make use of the 96 channels of input data, data reduction is necessary. Figure 3 shows a diagram of the data reduction scheme for 16 channels.

2.2.1. Automatic bit selection. One effective method of reducing the amount of data is to simply reduce the bit resolution of the output data. The DHSMs use 12-bit analog-to-digital converters (ADCs) and pass all twelve bits to our device. However, the use of all twelve bits is generally

Table 1. Input-referred voltage range and resolution for the sets of 8 bits that can be selected through automatic bit selection. These input-referred values reflect a gain of 5000 before the ADCs and a full-scale range of 2.4 V for the ADCs. Bit 0 is the least significant bit of the ADC output. Bit 11, the most significant bit of the ADC output, plays the role of a sign bit. The first column indicates the range in which the maximum offset from the mean observed during the monitoring period must fall for a given set of bits to be selected.

Maximum observed input-referred offset from the mean (μV)	Bits selected	Input-referred voltage range (μV)	Input-referred resolution (μV)
<11.7	0 through 6 and 11	−15 to 15	0.12
11.7 to 23.4	1 through 7 and 11	−30 to 30	0.23
23.4 to 46.9	2 through 8 and 11	−60 to 60	0.47
46.9 to 93.8	3 through 9 and 11	−120 to 120	0.94
>93.8	4 through 10 and 11	−240 to 240	1.88

unnecessary. For low amplitude signals, the highest bits contain no information. For high amplitude signals, the resolution provided by the lowest bits is not critical. These bits can be ignored with minimal loss of useful information. Our device performs automatic bit selection to select the 'best' 8 bits for each channel. The 'best' 8 bits are the bits that provide the highest resolution while still providing a sufficient range on a given channel. This determination is made based on approximately 2 s worth of data that are collected during an initial monitoring period. This monitoring period is initiated by a Reset command received from the WCM. (The commands are described in section 2.4.) The bits are selected to allow for a roughly 25% increase in deviation from the channel mean over that seen during the monitoring period. Any further processing within the device is then performed on these 8 bit samples. In addition, the mean value for each channel is computed during the initial monitoring period. From then on, the mean computed for a given channel is subtracted from incoming samples on that channel. Consequently, the data to be processed for each channel are 'zero mean.' Selection of bits for a given sample is made after the mean has been subtracted. The most significant bit of the 12 bit sample essentially performs the function of a sign bit, so it is always preserved regardless of which other bits are selected. Automatic bit selection significantly reduces the amount of data, allows for a relatively large input range and retains the resolution of low-amplitude signals. Figure 4 illustrates automatic bit selection.

The DHSMs have a voltage gain of 5000, and the ADCs on the DHSMs have a full-scale range of 2.4 V. Table 1 shows the input-referred voltage range and resolution corresponding to each set of 8 bits that can be chosen during automatic bit selection. Additionally, the table shows the set of bits that will be selected as a function of the maximum deviation from the channel mean observed during the monitoring period.

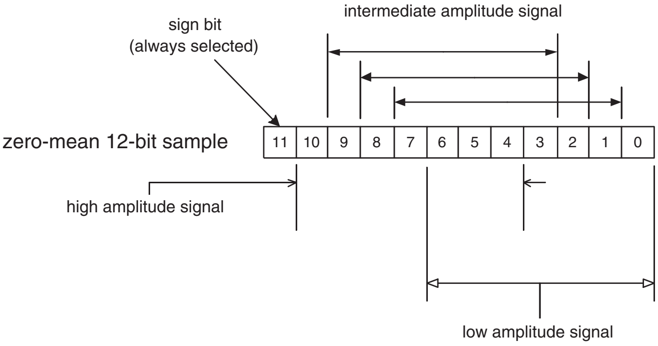

Figure 4. Automatic bit selection. For each 12 bit sample from the digitizing headstages, our device subtracts the channel mean and then selects the best 8 bits. When signals have low amplitudes, the large range provided by the most significant bits is unnecessary, so these bits are not used. When signals have large amplitudes, the resolution provided by the least significant bits is unnecessary. For signals with intermediate amplitudes, dropping a combination of most significant bits and least significant bits is optimal. The ranges in the figure show which set of bits would be used for signals of high, intermediate and low amplitudes. The most significant bit (bit 11) is essentially a sign bit, so it is always used regardless of which set of 7 additional bits is selected.

2.2.2. Data output modes. The data reduction scheme takes advantage of the fact that, in general, information regarding the spiking activity of neurons is what is truly of interest. However, since at times the raw neural signal is of interest, the scheme also allows for the full datastream from a single channel to be output. The device has the option of outputting data in one of the following four modes: raw streaming data for a single channel, extracted spike waveforms for any subset of the 96 channels, 1 ms bincounts for all 96 channels and streaming data from a single channel along with the extracted spikes for that channel. In the streaming data mode, the option exists of outputting either all 12 bits or only the automatically selected 'best' 8 bits for each sample. In the extracted spikes mode, spikes are detected on all of the enabled channels using a threshold applied to the absolute value of samples. This is equivalent to applying both a positive and a negative threshold. This method of spike detection has been shown to be appropriate for systems with limited computational resources [10]. For each channel i, the threshold is computed as $\alpha\frac{1}{N}\sum_{j=1}^{N}|x_{ij}|$, where α is an operator-defined parameter, x_{ij} are 'zero-mean' samples collected from channel i and $N = 512$. For each channel, the mean of the absolute value of samples is calculated using approximately 16 ms worth of data collected following the initial monitoring period described in section 2.2.1. This provides an easily computed measure of the background noise level on the channel. The parameter α allows the operator to select the sensitivity of the spike detector. Typical values of α are 6, 7 and 8. On a given channel, a spike is detected any time the threshold is crossed as long as a 1.28 ms refractory period has passed since the last detection.

For each detected spike, a 48 byte packet is constructed containing the number of the channel on which the spike occurred, a 2 byte timestamp with 1 ms resolution and a 45 byte spike waveform. Thus shape, timing and channel information are included in the output for each detected spike. The timestamp is synchronized to a timer contained in the external WCM. As a result, the timing of spikes extracted within the body can be known relative to events occurring outside of the body. Minor adjustments to the timer in our device are made every 50 ms to prevent drift between the internal and external timers. In addition, every 4 s, the internal timer is set to exactly match the external timer. The spike waveform includes 10 samples before the threshold crossing and 35 samples after it.

In cases in which the spike shape is not of interest, the data can be reduced even further by detecting spikes and outputting bincounts for each 1 ms period. (In general a 'bincount' is simply a count of the number of spikes that occurs during a given period of time. Since the temporal extent of a neural spike is on the order of 1 ms, each 1 ms bincount has a value of either 0 or 1.) The fourth output mode, streaming data with extracted spikes for a single channel, presents both raw and reduced data for one channel. For each of these modes, our device outputs a packet of data every 50 ms.

2.2.3. Merging of output data. As stated earlier, six lines of serial data, each containing data from 16 channels, serve as input to our device. These six lines are processed in parallel resulting in up to six separate streams of reduced neural data. Only one stream of data can be sent to the transceiver. Thus, as the data reduction on each of the individual lines is performed, the output from the six data reduction modules is merged together to form a single stream of data to be sent to the transceiver.

This merging procedure is trivial in all data output modes except for the extracted spikes mode. In the extracted spikes mode, the merging of data is complicated by the fact that the amount of data (based on the number of spikes) and the location of the data (based on the set of 16 channels on which a given spike occurs) cannot be known beforehand. To deal with this, an extra level of buffering is used to temporarily hold spikes in each individual 16-channel data reduction module until those spikes can be merged into the single output datastream. Each data reduction module has its own 2 kilobyte first-in-first-out (FIFO) buffer. Each FIFO can hold 42 extracted spikes. As spikes are extracted and placed into these FIFOs, a data merging supervisor keeps track of the time when spikes are placed into the individual FIFOs and then reads them out in the proper order. As spikes are read out, the merged output datastream is stored in a transmit buffer. The system has two transmit buffers, each capable of holding 4800 bytes, which corresponds to exactly 100 extracted spikes. During a given 50 ms period, one transmit buffer is being filled while the other is being read out so that its data can be transmitted by the transceiver. At the end of the 50 ms period, the buffers are switched so that the one that is now empty can be filled with new data and the one that just finished being filled can have its data transmitted.

Each transmit buffer has a finite amount of storage space. Before reading a spike out from a FIFO, the data merging supervisor checks to make sure that the transmit buffer has space to hold a new spike. If the transmit buffer is full, spikes remain in the FIFOs until the swap of transmit buffers frees up space to hold more spikes. As a result, it is possible that a FIFO may fill up and be unable to hold additional spikes. If spikes are extracted within a data reduction module that has a full FIFO, these spikes are dropped from the output datastream. With each set of extracted spikes that is output, our device provides an indication for each of the six data reduction modules of whether or not any spikes had to be dropped from that module during the corresponding 50 ms period.

2.3. Telemetry engine

A telemetry link provides a means of communication between an implanted portion of a neural data acquisition system and the external world. Our device contains transmit and receive engines that interface with a 916.5 MHz, 1 Mbps transceiver. The telemetry link allows an external module (such as the WCM) to send commands and configuration information to the implantable portion of the neural data acquisition system and allows the implantable module to send data back to the external module.

The transceiver uses a three-line interface. One line determines the transceiver's transmit/receive state. Another line serves as the input to the transceiver when in the transmit mode. The third line serves as the output from the transceiver when in the receive mode.

The transmit engine formats the single stream of output data so that it can be transmitted successfully over the telemetry link. A packet consisting of the output data and header information is assembled so that the receiver can both interpret and perform error-checking on the received data. Each byte is encoded into a 10 bit symbol (8b/10b encoding [11]) to ensure that the bitstream that is sent to the transceiver is dc balanced, to allow for the use of special symbols and to ensure that the minimum transition requirements for bit recovery are met. Finally, the encoded bitstream is sent serially to the transceiver.

The receive engine takes the bitstream that is output from the transceiver and reconstructs and decodes it. No timing or framing information is received apart from that which can be extracted from the received serial datastream. When receiving data, the receive engine first recovers the bitstream present in the received data line using a data locked loop [12]. Once the individual bits and framing information have been recovered, 10 bit symbols are decoded and interpreted based on the packet structure. Finally, the received commands and data are passed out of the receive engine for further use within the system.

The presence of both a transmit engine and a receive engine provides our device with the capability for bidirectional communication. The transmit and receive engines are complementary. In other words, the receive engine is designed to properly interpret data that has been processed by the transmit engine. Consequently, both the implantable module and the external module use these engines to interface with their respective transceivers.

2.4. Command interpretation

Our device is capable of responding to the following five commands from the WCM: (1) Loopback, (2) Write Configuration Registers, (3) Read Configuration Registers, (4) Request Acquired Data and (5) Reset. In typical operation, the ICCM expects to receive a command from the WCM every 50 ms.

The Loopback command is sent by the WCM in a packet along with 256 data bytes. Upon receiving this packet, the ICCM responds by echoing back the received data. This command is used to ensure that the communication pathway is operating properly. The Write Configuration Registers command is used to configure settings within our device. The packet in which this command is sent contains information for setting a channel enable list, the data acquisition mode (one of the four modes described in section 2.2.2), and the operator-defined parameter used in calculating thresholds. In response to this command, the ICCM updates its internal settings and sends a copy of the received data back to the WCM. The Read Configuration Registers command allows the external module to determine the current settings of our device. In response to this command, our device transmits its current configuration settings. The device also sends an indication of which 8 bits were selected for each channel during the automatic bit selection. This information is necessary for mapping the data that are sent back to the external module into actual voltages. The Request Acquired Data command is the command that allows the WCM to obtain the neural data that has been acquired over the past 50 ms. The format of the data output by our device in response to this command is based on the data acquisition mode setting. The final command, Reset, initiates the calculation of channel means, automatic bit selection and the calculation of thresholds.

3. Results

The single-chip signal processing and telemetry engine has been implemented and tested. A circuit board for our device and other components of the implantable module for a neural data acquisition system, including the transceiver, has been designed and is shown in figure 5. The fully populated board measures 40× 50× 15 mm^3 in size, weighs approximately 17 g and serves as the ICCM.

3.1. FPGA utilization and power consumption

The VHDL and Verilog hardware description of the device consists of approximately 6000 lines of code. The design utilizes almost all of the resources of the FPGA. Specifically, 49% of the FPGA's 10 240 flip flops, 93% of the FPGA's 10 240 4-input look-up-tables and 100% of the FPGA's 40 18-kilobit blocks of RAM are used by the design.

The FPGA consumes approximately 100 mW of power. Our device uses 53 mA from a 1.5 V supply and 7.6 mA from a 3.3 V supply. Each 16-channel data reduction module is responsible for approximately 10 mW of power consumption. The data merging, telemetry processing, command interpretation and system supervision have a combined consumption of about 20 mW. Additionally, the FPGA has a quiescent power consumption of roughly 20 mW, bringing the total power consumed to 100 mW.

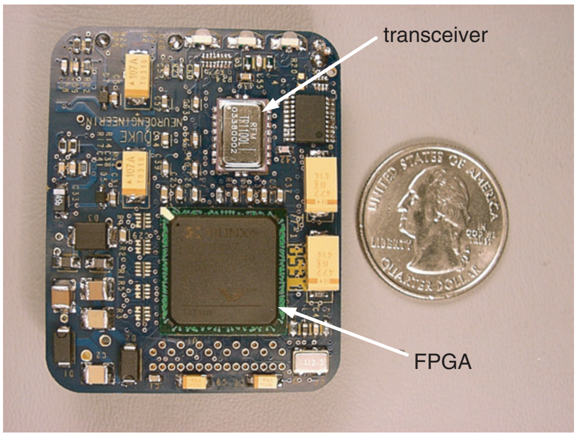

Figure 5. A circuit board designed for components of a fully implantable module of a neural data acquisition system. This board serves as the implantable central communications module (ICCM). The arrows indicate our device (the FPGA) and the transceiver with which it interfaces.

(This figure is in colour only in the electronic version)

Table 2. Bits selected for waveforms with spikes of various peak amplitudes.

Spike amplitude (μV)	Bits selected
0	2 through 8 and 11
20	2 through 8 and 11
30	3 through 9 and 11
70	3 through 9 and 11
80	4 through 10 and 11

3.2. Automatic bit selection

In order to test the automatic bit selection feature of our device, we used a National Instruments (Austin, TX) 6723 analog output card to generate waveforms with spikes of varying amplitudes. One period of a 2 kHz sine wave with a 180° phase shift was used as the spike waveform. The spikes occurred at a rate of 20 times per second. No noise was incorporated into these waveforms. The waveforms were attenuated by a factor of 10 000 using a resistor network in order to bring the voltages into the range of recorded extracellular potentials (generally less than 500 μV peak to peak). The resistor network had an output impedance of 220 kΩ. Each waveform was used as an input to a single channel of a DHSM. Our device performed automatic bit selection on each waveform. Table 2 shows which bits were selected for waveforms with different spike amplitudes. Spike amplitudes were increased in steps of 10 μV. The table shows the results for spike amplitudes at the boundaries for the various sets of bits. The two lowest sets of bits, bits 0 through 6 and bits 1 through 7, were never selected due to the noise added to the waveforms by the recording circuitry. Discrepancies between the results shown in table 2 and the first two columns of table 1 are also due to the noise added by the recording circuitry.

3.3. Spike extraction

In order to investigate the circumstances under which our device will drop extracted spikes from the output datastream, we examined the effect of the firing rate and the distribution of enabled channels on when a data reduction module's FIFO would overflow. In each trial, either 16 or 32 channels were enabled. The following three channel distributions were used: (1) 16 channels enabled—all 16 channels belonging to the same data reduction module, (2) 16 channels enabled—8 channels belonging to one data reduction module and the other 8 channels belonging to a different data reduction module and (3) 32 channels enabled—16 channels belonging to one data reduction module and the other 16 channels belonging to a different data reduction module. For each trial, one of these channel distributions was selected and a per-channel firing rate was selected. Based on this firing rate, waveforms were generated with equal spacing between the spikes and provided as input to our system as described in section 3.2. For a given trial, identical waveforms were used as input to all of the enabled channels.

When 16 channels all belonging to the same data reduction module were enabled, spikes were first dropped at a firing rate of 126 spikes per second per channel. When 16 channels evenly split between two data reduction modules were enabled, spikes were also first dropped at a firing rate of 126 spikes per second per channel. In both of these cases, no FIFO overflow was observed at a firing rate of 125 spikes per second per channel. Also, in both cases at a firing rate of 126 spikes per second per channel, roughly 20% of the packets indicated that spikes were dropped. At a firing rate of 130 spikes per second per channel, generally every other packet indicated dropped spikes.

In the case of 32 enabled channels, spikes were first dropped at a firing rate of 63 spikes per second per channel. Approximately, 15% of the packets indicated that spikes were dropped. In this case, no FIFO overflow was observed at a firing rate of 62 spikes per second per channel. At a firing rate of 65 spikes per second per channel, generally every fourth packet indicated dropped spikes.

3.4. Latency

We explored the latencies present in our system between when a spike occurs and when the corresponding extracted spike is received and is ready for further use within the WCM. We examined how these latencies behave as the overall firing rate of the input channels changes. A waveform with a firing rate of approximately one spike per second was used as input on a 'target channel.' The latency was computed as the time between when the analog output card generated a spike for this channel and when a spike from this channel was received and ready for further processing within the WCM. Four different trials were run, each with different overall firing rates. In each trial, the latencies for 500 spikes were measured. Spikes that were generated but were never received by the WCM were considered to be dropped spikes.

In each trial, 25 channels in addition to the target channel were enabled. Twelve of these channels belonged to the same data reduction module as the target channel and the other 13 channels belonged to a different data reduction module. In the first trial, the target channel was the only channel with spikes. The mean latency was 26.7 ms. A histogram of the latencies for this case is shown in figure 6(a). In the second trial, each of the 25 enabled non-target channels had a firing rate of 40 spikes per second, for an overall firing rate of approximately 1000 spikes per second, or 50 spikes per 50 ms. The mean latency was 51.0 ms. The histogram for this case is shown in figure 6(b). In the third trial, each of the 25 enabled non-target channels had a firing rate of 79 spikes

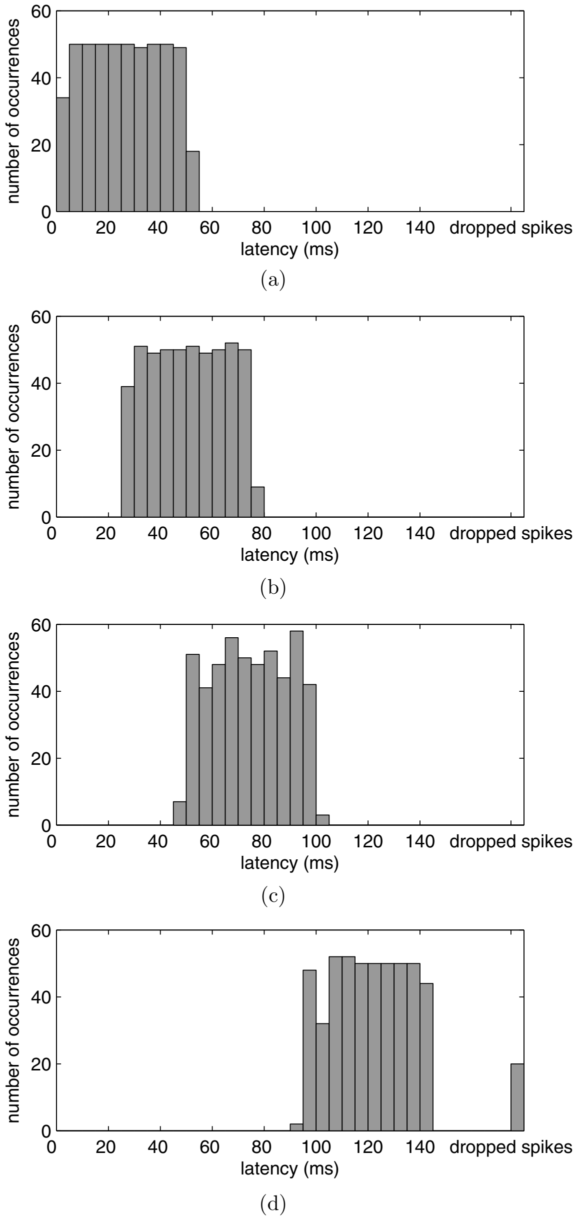

Figure 6. Histograms of latencies between when the spikes occurred and when they were received and ready for further processing by the external module. The overall firing rates on the inputs were (a) 1 spike per second, (b) approximately 1000 spikes per second, (c) approximately 2000 spikes per second and (d) approximately 2125 spikes per second. Each bin covers a 5 ms range. A bin for the spikes that were not transmitted due to buffer overflow is included as well.

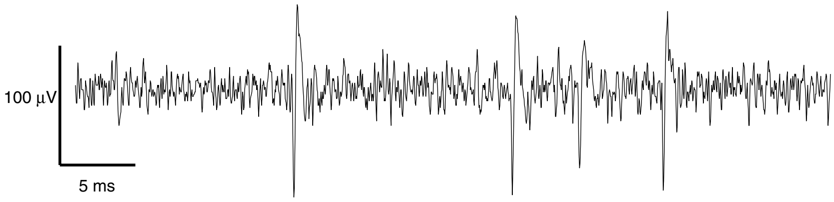

Figure 7. Output of the streaming data mode. The figure shows 50 ms of data recorded from an owl monkey.

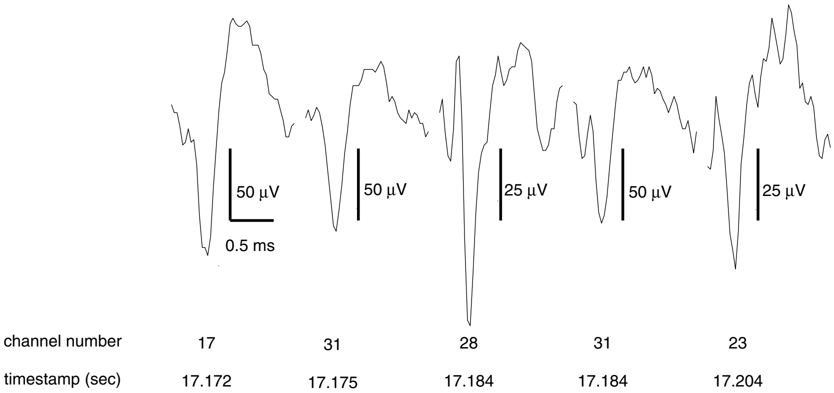

Figure 8. Output of the extracted spikes data acquisition mode. The figure shows spikes extracted during a 50 ms period of recording from a rat. For each detected spike, a 1 byte channel number, a 2 byte timestamp and a 45-byte waveform are output by our device. In the figure, the 3 byte header has been removed for each spike, and the corresponding channel number and timestamp have been indicated below each extracted waveform. During this particular 50 ms period, two spikes were extracted from one channel (channel number 31) and one spike was extracted from each of three other channels. A vertical scale is provided for each individual extracted waveform because, due to different bits being automatically selected for the different channels, the channels do not all have the same vertical scale. The horizontal scale is the same for all the extracted spikes.

per second, for an overall firing rate of just under 2000 spikes per second, or 100 spikes per 50 ms. The mean latency was 74.9 ms, and the histogram is shown in figure 6(c). No spikes were dropped in these first three cases. In the final trial, each of the 25 enabled non-target channels had a firing rate of 85 spikes per second, for an overall firing rate of approximately 2125 spikes per second. The mean latency was 120.3 ms, and 20 out of the 500 spikes were dropped. The histogram for this case is shown in figure 6(d).

3.5. Telemetry

The Loopback command was used to test the digital processing involved with telemetry and to ensure the integrity and reliability of the telemetry link. As stated earlier, the Loopback command is sent along with known data by the WCM every 50 ms. The ICCM receives this data and sends a copy of what it has received back to the WCM. In this test, the WCM compared the data it received from the ICCM with the data that was originally sent. The WCM then provided an indication of whether or not the received packet contained the correct data. In addition, the WCM kept track of the number of dropped packets (packets that were sent for which no response was received). The system was run in the Loopback mode for 1 h. Out of a total of 72 000 packets (one every 50 ms), 12 packets (0.017%) contained errors and 24 packets (0.033%) were dropped. During this test, a distance of 2 m separated the ICCM from the WCM.

3.6. Live animal recordings

Using the ICCM in conjunction with up to three DHSMs and a module for sending commands to and receiving data from our device (i.e. a WCM), we successfully recorded neural data from an owl monkey and from Sprague Dawley rats. In these cases, only the microelectrode arrays were implanted. This allowed us to test the functionality of the device with real data without having to implant an entire module.

Each of the various output modes was tested during the live animal recordings. Figure 7 shows 50 ms of 8 bit streaming data recorded from an owl monkey. Figure 8 shows the output from our device while in the extracted spikes mode. The figure shows spikes extracted during a 50 ms period and includes information concerning the spike shape, time and channel. This spike extraction was performed while recording from a rat with 32 enabled channels. During this particular 50 ms period, spikes were detected on four channels. Finally, figure 9 shows the output of the streaming data with extracted spikes mode. This includes 50 ms of raw neural data followed by the spike extracted during that time period.

4. Discussion

4.1. Spike sorting

Our device has been designed to be one component of a BMI. Within a BMI application, spike sorting is a commonly used signal processing step. The extracted spikes mode of our

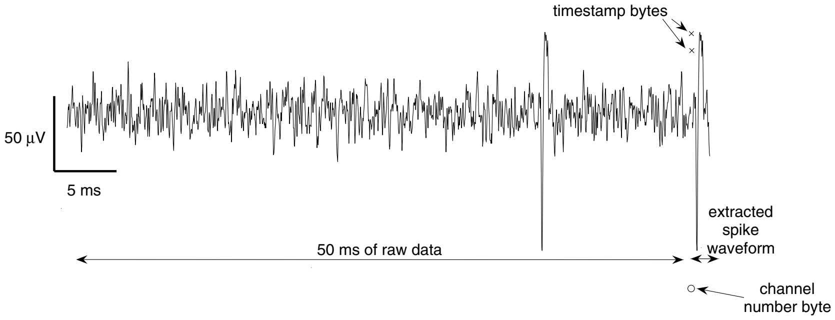

Figure 9. Output of the streaming data with extracted spikes data acquisition mode. The figure shows 50 ms of data recorded from an owl monkey followed by the spike extracted during that period of time. As in the extracted spikes mode, each extracted waveform is accompanied by a 1 byte channel number and a 2 byte timestamp.

device was developed with this in mind. This mode provides the information necessary for spike sorting to be performed outside of the body where it can be supervised and modified easily. At the same time, the data that are not necessary for spike sorting, the baseline voltage trace between spikes, are discarded within our device resulting in significant data reduction. High channel-count implantable devices that do not provide the extracted spike shape [13–15] preclude standard spike sorting as a further processing step. It should be noted, however, that some level of spike discrimination might still be possible in such systems. For example, while the system presented in [14] does not output extracted spike waveforms, it does provide spike timing and amplitude information, which may be used in discriminating spikes that are recorded on the same electrode. Nevertheless, to the best of our knowledge, the device presented here is the first implantable, high channel-count neural signal processor with the capability of outputting extracted spike waveforms.

4.2. Data reduction

The amount of data reduction resulting from spike extraction ultimately depends on the rate at which spikes are being detected. The total number of spikes that are present in the recorded waveforms depends on two factors—the number of neurons from which the electrodes are recording and the firing rates of these neurons. The system described here has 96 channels. It is very unlikely that active units will be present on all 96 channels. However, having 60% to 80% of the channels yield active units is not unreasonable [1, 16]. Having a single electrode record from multiple neurons is also quite common. As a result, a value of 1 to 2 is realistic for the ratio of total units to electrodes [1, 16]. Thus, a generous estimate of the total number of neurons from which the electrodes of a 96-channel system might record is 200. Typical neural firing rates fall in the range of roughly 1 to 10 spikes per second for resting neurons and roughly 10 to 100 spikes for active neurons [13]. At any given time, only a fraction of the neurons will be active. Assuming an average firing rate of 20 spikes per second per neuron, the extracted spikes mode would need to output 192 kilobytes per second (kBps)

$$(200\text{ neurons})\left(\frac{20\text{ spikes s}^{-1}}{\text{neuron}}\right)\left(\frac{48\text{ bytes}}{\text{spike}}\right) = 192\text{ kBps}.$$

In order to output all samples for all channels at 8 bits each, an output data rate of 3000 kBps would be needed

$$\left(\frac{31.25\text{ kBps}}{\text{channel}}\right)(96\text{ channels}) = 3000\text{ kBps}.$$

Consequently, spike extraction can be expected to reduce the amount of data by roughly a factor of 15. A spike extraction scheme similar to that described here has been used in live neural recordings and was found to reduce data by 97% [17].

Data reduction through spike extraction is only beneficial if spikes are detected properly. If true spikes are not detected, valuable information will be lost. If detections are made where no true spikes are present, the efficiency of the data reduction will decrease. As stated in section 2.2.2, our system performs spike detection by applying a threshold to the absolute value of the data. This is a computationally simple method of performing spike detection. While more complex spike detection algorithms exist and may provide better performance, this thresholding method is suitable for a system with limited computational resources [10]. (A detailed evaluation of spike detection methods is beyond the scope of this paper. See [10, 18–25] for more on spike detection.) We are currently investigating methods for improving how thresholds are selected in order to ensure acceptable spike detection performance. This investigation includes looking at the effect of the window size over which an estimation of the noise level is made, comparing various operators used for estimating noise levels and comparing the performance of adapting and non-adapting thresholds.

4.3. Spike extraction limitations

In an ideal neural data acquisition system, the performance of spike extraction would be equivalent to that of spike detection. For each detection, a given number of samples before and after the detection point would be stored as the extracted spike waveform. As long as spikes were detected successfully, there would be no reason for them to not be extracted properly. However, in systems with limited resources, such as that described here, spike detection is only one factor in determining how well extracted spikes can be acquired by the system. In a fully implantable neural data acquisition system, memory and data rate limitations must be considered as well. A spike that has been extracted must be stored in memory until it can be transferred out of the body via the telemetry link. In our system, the telemetry link has a finite data rate that allows for a maximum of 100 extracted spikes to be transmitted in a given 50 ms period. Consequently, if too many spikes are extracted within a given period of time, our device will be unable to store and transmit some of them. These spikes must simply be dropped from the output datastream.

The results presented in section 3.3 indicate that the major factor in determining when spikes will be dropped is the total spike rate across all channels. For all three distributions of enabled channels, the first FIFO overflows were observed when the total spike rate was slightly more than 2000 spikes per second. A rate of 2000 spikes per second corresponds to a rate of 100 spikes per 50 ms period. Considering that our device can output a maximum of 100 spikes per 50 ms period, it is not surprising that spikes are dropped once this rate is exceeded. If more than 100 spikes are detected in a 50 ms period, the transmit buffer will be full and spikes will begin to accumulate in the individual data reduction module FIFOs. If spike rates remain above the 100 spikes per 50 ms limit for an extended period of time, the FIFOs will eventually fill up and be unable to hold more spikes. In the two cases in which 16 channels were enabled and the firing rate per channel was 125 spikes per second, no overflow of the FIFOs occurred. These cases correspond to having exactly 100 spikes in each 50 ms period. As expected, no spikes are dropped as long as the spike limit is not surpassed.

The test setup used in section 3.3 allowed us to essentially look at FIFO overflow as a function of the firing rate. It must be noted, however, that realistic neural data are very unlikely to have perfectly constant firing rates. The results that we have presented represent 'steady-state' behaviors. In other words, the results show how our system behaves after the firing rates of the inputs have been constant for a significant period of time (on the order of 1 s). In cases in which the overall firing rate is greater than 100 spikes per 50 ms, the steady-state behavior occurs only after the FIFOs have had enough time to gradually fill up. In fact, the FIFOs can serve as temporary buffers so that the system can accommodate brief periods of overall spike rates in excess of the 100 spikes per 50 ms limit without dropping any spikes.

The results presented earlier indicate that the steady-state behaviors of the two channel distributions with 16 enabled channels are very similar. However, these two cases differ in how long they take to reach this steady state. In the case in which the 16 channels were split between two different data reduction modules, approximately twice as many packets were sent before an occurrence of FIFO overflow as in the case in which the 16 channels all belonged to the same data reduction module. This difference in 'transient' behavior is a result of the fact that during the pre-steady-state period, the individual FIFOs in the former case are in essence each filling up at half the rate of the single FIFO in the latter case.

The steady-state behavior of our device is mostly determined by the total spike rate across all channels while the transient behavior is additionally influenced by the distribution of enabled channels. As a result, the circumstances under which our device will drop spikes are largely determined by the overall distribution of spikes in time with some influence by the distribution of spikes in space.

As mentioned in section 3.2, no noise was incorporated into any of the waveforms used for these experiments. This was done intentionally so that only 'true' spikes would be detected and extracted. By eliminating false detections, we were able to carefully control how many spikes were extracted over a given period of time. With real neural data, however, it must be assumed that some false detections will occur. Noise that is incorrectly extracted as a spike will take up space in the transmit buffer and could contribute to overflows of the FIFOs.

4.4. Latency

Even when spikes are not dropped, high spike rates may increase the delay between when the spikes are detected and when they are transmitted [26]. The ICCM transmits extracted spikes once every 50 ms. If the telemetry system had infinite bandwidth, then the expected average latency would be 25 milliseconds and we would expect the latencies to be uniformly distributed between 0 and 50 milliseconds. Spikes that occur near the beginning of a 50 ms period would be expected to have a latency close to 50 ms while spikes that occur near the end of a 50 ms period would be expected to have a latency close to 0 ms. However, since the implantable module can only transmit a finite amount of data during each 50 ms period, it is possible that spikes which are detected during one 50 ms period will not be able to be transmitted at the end of that 50 ms period, thus increasing the average latency. Also, the amount of time required to transmit a complete packet increases as the size of the packet increases. As a result, packets that contain large numbers of spikes will take a non-negligible amount of time to be transmitted. Spikes in such packets will also have a higher than ideal average latency. For a clinical BMI, total latencies should be less than 200 ms [27].

Our results show that latencies do vary with the total firing rate. However, even in the worst case that we tested, the latencies remained safely below 200 ms. In the first trial, very few spikes were present. The distribution of latencies as presented in figure 6(a) is close to what would be expected in the ideal case described above. The mean latency, 26.7 ms, and the distribution of latencies indicate that no more than roughly 2 ms of latency are added by processing in the system. Almost all latency present in this case is due to the fact that extracted spikes are transmitted by our device at the end of 50 ms periods.

In the second trial, the overall firing rate was approximately 1000 spikes per second, or 50 spikes per 50 ms. This is one half of the 100 spikes per 50 ms period limit. Figure 6(b) again shows a relatively uniform distribution of latencies over a 50 ms range. However, the mean and distribution are shifted up by approximately 25 ms from the first case. This is a result of the fact that the WCM waits until it has received an entire packet before it is ready to perform further processing on any of the extracted spikes. The 25 ms shift is due directly to the fact that a packet containing 50 spikes takes approximately 25 ms to be transmitted.

In the third trial, the overall firing rate was just under 2000 spikes per second, or 100 spikes per 50 ms. Even though no spikes were dropped during this trial, each packet was either full or nearly full. Once again, figure 6(c) shows a more or less uniform distribution of latencies over a 50 ms range. The mean and distribution are shifted up by approximately 50 ms from the first case and by approximately 25 ms from the second case. Again, this shift is due to the fact that a packet containing 100 spikes takes approximately 50 ms to be transmitted.

In the final trial, the overall firing rate was approximately 2125 spikes per second. This rate exceeded the 100 spikes per 50 ms limit and resulted in dropped spikes. Spikes that were not dropped had latencies that were relatively uniformly distributed between about 95 ms and 145 ms. The large increase in latencies as compared to the previous case can be explained by the fact that in most cases during this trial, spikes that were detected during one 50 ms period would be unable to be transmitted at the end of that period. Rather, they would be forced to sit in our device until the end of the next 50 ms period.

Figure 6(d) has two significant features—the presence of dropped spikes and the noticeable dip in the 100 ms to 105 ms range. These two features are most likely related. Since the overall firing rate in this case is not tremendously greater than the 100 spikes per 50 ms limit, the buffers would only be completely filled near the end of a given 50 ms period. Consequently, spikes would only be dropped if they occurred near the end of a 50 ms period. If these spikes had not been dropped, they would be expected to have a latency of approximately 100 ms to 50 ms due to sitting in our device for an extra period as a result of a full transmit buffer followed by another 50 ms required for transmitting a full packet. Thus, the dip in the histogram is a result of the fact that spikes that would have fallen within a specific latency range were dropped instead of being transmitted.

We do not foresee problems with the latencies measured here. However, if necessary, it is possible to decrease the latency. The total latency within our system is essentially made up of the following two components: (1) the time from when a spike is detected until it is transmitted and (2) the time from when a spike is received by the external module until it is ready for further use within that module. The latency contributed by the first component is basically the time between when a spike is detected and when a given 50 ms period ends. Since the timing of spikes is independent of the times at which 50 ms periods start and end, we would expect this first component to contribute on average 25 ms of latency. Reducing the interval between when the WCM sends commands (and consequently between when our device responds to commands) would in turn reduce the first component of latency. For example, if spikes were sent out of the ICCM every 20 ms instead of every 50 ms, on average the time between when a spike would be detected and when it would be transmitted would be roughly 10 ms. The trade-off for increasing the frequency at which packets are sent out of the ICCM is that a larger percentage of the available bandwidth will need to be used for transmitting packet overhead. This is a result of the fact that each packet has a fixed amount of overhead and more packets will be transmitted when commands are received at a higher frequency.

The latency contributed by the second component is a direct result of the way the WCM processes data. Currently, our WCM does not use any of the data that it has received until it has received a complete packet. This is done so that error checking (through the use of a parity byte) can be performed before any of the data are processed further. Since a full packet can take up to almost 50 ms to transmit, the possibility exists that data will sit in the WCM for a significant amount of time before being used. An alternative, which would essentially eliminate this component of the latency, is to allow the WCM to use data as soon as they are received. The consequence in this case is that simple error-checking schemes which depend on knowledge of the contents of an entire packet will have extremely limited usefulness.

4.5. Flexibility

While we anticipate that our device will primarily be used in the extracted spikes mode, we have implemented it in such a way that flexibility in how data reduction is performed is available. Many different research and clinical applications of a neural data acquisition system exist. These applications do not all have the same requirements regarding the collected neural data. A fully implantable neural data acquisition system that is flexible enough to perform different forms of data reduction can be useful in these various applications while still limiting the amount of unnecessary information that is sent out and thus reducing the burden on the telemetry system. Even within a single experiment, flexibility in how data are processed and output is desirable. For example, while a given experiment might ultimately use the extracted spikes mode of our device, the streaming data mode could be a valuable tool for the investigator in making an initial assessment of the quality of the signals on individual channels. Also, cases exist in which the costs incurred by spike sorting are not necessarily justified by the potential information gained from this processing step [28]. In such cases, the flexibility to simply output spike detections, rather than extracted waveforms, is desirable.

The fact that the device has been implemented in programmable logic also provides flexibility. Our design can serve as a starting point for other systems with signal processing and telemetry needs. The modular nature of the design lends itself to modifications of individual portions of the system while leaving the overall system architecture intact.

4.6. Telemetry

Development of our device has been centered on the communication needs of a fully implantable neural data acquisition system. The transceiver interface clearly plays a crucial role in the telemetry link. Also, the data reduction scheme was implemented in large part as a response to telemetry limitations. The use of a custom packet structure and telemetry protocol has allowed us to implement the telemetry engine along with the data reduction algorithms in a single chip. This has also allowed us to limit overhead so that as much neural data as possible can be efficiently output by our device. The command-and-response structure of our telemetry protocol allows for flexibility in terms of what data can be obtained from the implantable module. It also allows the external module to determine when packets are being dropped and the communication pathway is not working properly. The results of the Loopback test presented in section 3.5 confirm that our device is capable of successful, reliable bidirectional communication with a suitable external module.

It should be noted that our system employs a relatively simple telemetry protocol. While the protocol allows for a baseline level of error-checking and allows the external module to determine when packets are being dropped, it does not provide a means for the retransmission of dropped commands or data. Since commands are sent every 50 ms, the WCM can simply repeat a given command (for example, Write Configuration Registers or Read Configuration Registers) until it receives a response from the ICCM. Additionally, the ICCM is set up to send out data at the end of each 50 ms period if the last command that it received was a request for data and it has not received any commands within the last 50 ms. Given these facts, the dropping of commands should not pose a significant problem. The more important question is whether or not providing a means for the retransmission of dropped data is justifiable. The cost of providing a means of retransmission would be quite high. Implementing a more complex communication protocol would consume valuable computational resources. More importantly, a significant increase in the amount of space allocated for storing data would be required since data would need to continue to be stored in the implantable module even after the original transmission. Otherwise, a future retransmission of the data would not be possible. The discussion of latency presented earlier suggests that data that are delayed significantly more than the delays that are already present in the system will be ‘too old’ to be useful in a real-time system. Providing the capability to retransmit dropped data would likely add significantly to the latency of the data. Thus, we feel that for a system with limited resources that is intended to be used in real time, the benefits of providing a more complex telemetry protocol do not justify the costs. Clearly, there are limitations to the telemetry system presented here. In the context of the full neural data acquisition system, we feel that these limitations will be best addressed through optimization of antennas, antenna configurations and device placement rather than through increasing the complexity of the telemetry protocol.

4.7. Power consumption

The power consumption of our device is high compared to that of devices such as those described in [13–15]. This is in large part due to the fact that our device was implemented in an FPGA while the other devices were implemented in application-specific integrated circuits (ASICs). The exact power savings gained by implementing a design in an ASIC rather than an FPGA depends on the specifics of the design, the FPGA and other factors. Nevertheless, for low-power systems, a reduction in power consumption by a factor of 10 or more is not an unreasonable result of switching from an FPGA implementation to an ASIC implementation [29]. Thus, an ASIC implementation of our design would be expected to consume 10 mW or less. Additionally, our implementation work has focused on functionality and constraints of resources within the FPGA. As a result, the device has not been optimized to reduce power consumption. We expect that optimization will provide further power savings. At the current stage of development of our system, the ability to program and modify the processing within our device is essential. As a result, we have chosen to retain this flexibility through the use of an FPGA even at the cost of higher power consumption.

Due to the power requirements of our system, using an implanted battery is not feasible. Rather, our system is designed to obtain its power transcutaneously through the use of inductive coils. We have successfully powered our system using a transcutaneous power link. Providing power to our device has not proven to be an obstacle.

One of the main benefits of performing data reduction in an implantable module is the power savings obtained by having to transmit only a small fraction of the neural data. As stated in section 4.2, transmitting all 8 bit samples from 96 channels would require a data rate of 3000 kBps, or 24 Mbps. Using 12 bit samples instead of 8 bit samples would push the data rate up to 36 Mbps. All else being equal, the power transmitted by a transceiver must increase linearly with the data rate in order to transmit the data the same distance with the same bit error rate [30]. Consequently, data reduction drastically decreases the system’s required transmit power. It should be noted, however, that since power is consumed within a transceiver apart from that which is ultimately transmitted, the overall power consumption of the transceiver does not necessarily increase linearly with the data rate [30].

One option for transmitting all of the data from a 96-channel system with no data reduction would be to use an IEEE 802.11g-compliant transceiver. Such a device can achieve a maximum data rate of 54 Mbps [31] and has a transmitted energy per bit comparable to our system. These devices typically consume several hundred milliwatts of power when transmitting data [31–34]. On the other hand, the 1 Mbps transceiver used in our system consumes on the order of 35 mW [35]. It should also be noted that a system with no data reduction would not necessarily eliminate the need for all processing of data. Some sort of merging of the many channels of data and much of the telemetry processing present in our system would likely still be necessary. One other alternative for transmitting large amounts of data is to use an infrared link. Using an infrared emitter/detector pair, up to 40 Mbps can be transmitted with less than 100 mW of power consumption [36]. Two drawbacks of transcutaneous infrared telemetry are the high sensitivity to alignment between the emitter and detector and the very limited range over which signals can be successfully transmitted [36].

5. Conclusion

A single-chip 96-channel neural signal processing and telemetry engine has been demonstrated. This device serves as the core processing unit in a fully implantable portion of a neural data acquisition system. It performs the processing necessary for data reduction and for a transceiver interface. This device is a critical component in the communication pathway that allows command and configuration information to be passed into and multi-unit neural data to be passed out of an implanted module.

Acknowledgments

The authors gratefully acknowledge the contributions of Chad Bossetti, Deborah Won, Thomas Jochum, Ellen Dixon-Tulloch and Joseph O'Doherty. This work was sponsored by DARPA contract no. N66001-02-C-8022.

References

[1] Nicolelis M A, Dimitrov D, Carmena J M, Crist R, Lehew G, Kralik J D and Wise S P 2003 Chronic, multisite, multielectrode recordings in macaque monkeys *Proc. Natl Acad. Sci.* **100** 11041–6

[2] Suner S, Fellows M R, Vargas-Irwin C, Nakata G K and Donoghue J P 2005 Reliability of signals from a chronically implanted, silicon-based electrode array in non-human primate primary motor cortex *IEEE Trans. Neural Syst. Rehabil. Eng.* **13** 524–41

[3] Kim S J, Manyam S C, Warren D J and Normann R A 2006 Electrophysiological mapping of cat primary auditory cortex with multielectrode arrays *Ann. Biomed. Eng.* **34** 300–9

[4] Serruya M D, Hatsopoulos N G, Paninski L, Fellows M R and Donoghue J P 2002 Instant neural control of a movement signal *Nature* **416** 141–2

[5] Taylor D M, Tillery S I and Schwartz A B 2002 Direct cortical control of 3D neuroprosthetic devices *Science* **296** 1829–32

[6] Carmena J M, Lebedev M A, Crist R E, O'Doherty J E, Santucci D M, Dimitrov D F, Patil P G, Henriquez C S and Nicolelis M A 2003 Learning to control a brain-machine interface for reaching and grasping by primates *PLoS Biol.* **1** 193–208

[7] Hochberg L R, Serruya M D, Friehs G M, Mukand J A, Saleh M, Caplan A H, Branner A, Chen D, Penn R D and Donoghue J P 2006 Neuronal ensemble control of prosthetic devices by a human with tetraplegia *Nature* **442** 164–71

[8] Lebedev M A and Nicolelis M A 2006 Brain-machine interfaces: past, present and future *Trends Neurosci.* **29** 536–46

[9] Rizk M, Obeid I, Callender S and Wolf P 2005 A single-chip 96-channel neural signal processor and telemetry engine for a brain-machine interface *BMES 2005: Proc. 2005 Biomed. Eng. Soc. Ann. Fall Meeting* p 631

[10] Obeid I and Wolf P D 2004 Evaluation of spike-detection algorithms for a brain-machine interface application *IEEE Trans. Biomed. Eng.* **51** 905–11

[11] Widmer A X and Franaszek P A 1983 A DC-balanced, partitioned-block, 8B/10B transmission code *IBM J. Res. Dev.* **27** 440–51

[12] Sawyer N 2005 Data recovery. Xilinx Application Note 224

[13] Harrison R R, Watkins P T, Kier R J, Lovejoy R O, Black D J, Greger B and Solzbacher F 2007 A low-power integrated circuit for a wireless 100-electrode neural recording system *IEEE J. Solid-State Circuits* **42** 123–33

[14] Olsson III R H and Wise K D 2005 A three-dimensional neural recording microsystem with implantable data compression circuitry *IEEE J. Solid-State Circuits* **40** 2796–804

[15] Sodagar A M, Wise K D and Najafi K 2007 A fully integrated mixed-signal neural processor for implantable multichannel cortical recording *IEEE Trans Biomed. Eng.* **54** 1075–88

[16] Nicolelis M A L, Ghazanfar A A, Faggin B M, Votaw S and Oliveira L M O 1997 Reconstructing the engram: simultaneous, multisite, many single neuron recordings *Neuron* **18** 529–37

[17] Obeid I 2004 A wireless multichannel neural recording platform for real-time brain machine interfaces *PhD Thesis* Duke University

[18] Bankman I N, Johnson K O and Schneider W 1993 Optimal detection, classification, and superposition resolution in neural waveform recordings *IEEE Trans. Biomed. Eng.* **40** 836–41

[19] Chandra R and Optican L M 1997 Detection, classification, and superposition resolution of action potentials in multiunit single-channel recordings by an on-line real-time neural network *IEEE Trans. Biomed. Eng.* **44** 403–12

[20] Kim K H and Kim S J 2003 A wavelet-based method for action potential detection from extracellular neural signal recording with low signal-to-noise ratio *IEEE Trans. Biomed. Eng.* **50** 999–1011

[21] Nenadic Z and Burdick J W 2005 Spike detection using the continuous wavelet transform *IEEE Trans. Biomed. Eng.* **52** 74–87

[22] Choi J H, Jung H K and Kim T 2006 A new action potential detector using the MTEO and its effects on spike sorting systems at low signal-to-noise ratios *IEEE Trans. Biomed. Eng.* **53** 738–46

[23] Rutishauser U, Schuman E M and Mamelak A N 2006 Online detection and sorting of extracellularly recorded action potentials in human medial temporal lobe recordings, *in vivo J. Neurosci. Methods* **154** 204–24

[24] Borghi T, Gusmeroli R, Spinelli A S and Baranauskas G 2007 A simple method for efficient spike detection in multiunit recordings *J. Neurosci. Methods* **163** 176–80

[25] Thakur P H, Lu H, Hsiao S S and Johnson K O 2007 Automated optimal detection and classification of neural action potentials in extra-cellular recordings *J. Neurosci. Methods* **162** 364–76

[26] Bossetti C A, Carmena J M, Nicolelis M A L and Wolf P D 2004 Transmission latencies in a telemetry-linked brain-machine interface *IEEE Trans. Biomed. Eng.* **51** 919–24

[27] Donoghue J P 2002 Connecting cortex to machines: recent advances in brain interfaces *Nature Neurosci.* **5** (Suppl.) 1085–8

[28] Won D S and Wolf P D 2004 A simulation study of information transmission by multi-unit microelectrode recordings *Network: Comput. Neural Syst.* **15** 29–44

[29] Amara A, Amiel F and Ea T 2006 FPGA versus ASIC for low power applications *Microelectron. J.* **37** 669–77

[30] Wang A Y and Sodini C G 2006 On the energy efficiency of wireless transceivers *ICC 06: IEEE Int. Conf. on Commun. 2006* vol 8

[31] Ferro E and Potorti F 2005 Bluetooth and Wi-Fi wireless protocols: a survey and a comparison *IEEE Wireless Commun.* **12** 12–26

[32] Chen T M *et al* 2007 A low-power fullband 802.11 a/b/g WLAN transceiver with on-chip PA *IEEE J. Solid-State Circuits* **42** 983–91

[33] Maxim Integrated Products 2007 (June) MAX2825 key specifications www.maxim-ic.com/quick_view2.cfm/qv_pk/4191

[34] NXP Semiconductors 2007 (June) BGW211 specifications http://www.nxp.com/acrobat_download/literature/9397/75015598.pdf

[35] RF Monolithics, Inc. 1999 TR1100 datasheet

[36] Guillory K S, Misener A K and Pungor A 2004 Hybrid RF/IR transcutaneous telemetry for power and high-bandwidth data *Proc. 26th Ann. Int. Conf. of the IEEE EMBS* vol 2 pp 4338–40