

Low-Power Digital Signal Processing for an Implantable Neural Recording System

Paul T. Watkins

University of Utah Department of Electrical
and Computer Engineering
50 S Central Campus Dr Rm 3280 MEB
Salt Lake City, UT 84112-9206

watkins@eng.utah.edu

ABSTRACT

In this paper, we describe the design of digital signal processing components for a fully-implantable, wireless neural recording system. The digital signal processing is intended to extract neural signals from a noisy background and thereby enable more efficient use of transmission bandwidth. Low power consumption is critical to the system. To this end, the algorithms for the digital signal processing are simplified as much as possible. The use of static CMOS logic vs. pass transistor logic is considered. Low supply voltage operation is also investigated. The signal processing modules are designed for a commercially available 0.5 μm process.

Keywords

Low power, biomedical implants, digital signal processing.

1. INTRODUCTION

We are developing a fully-implantable wireless neural recording system [1]. The proposed system includes a mixed-signal integrated circuit flip-chip bonded onto a 10 by 10 microelectrode array. The integrated circuit includes neural signal amplifiers for each electrode, an analog-to-digital converter, and RF circuitry for wireless telemetry. Digitized waveforms from all 100 electrodes would generate prohibitively high data rates for wireless transmission. In the current system, only one channel is digitized. All other channels transmit one bit per millisecond to indicate whether an action potential, or spike, has been detected. The RF circuitry, which consumes 50% of the entire system power, is constantly running, transmitting data even when there is only background noise in the signal. In addition, the transmit bandwidth is limited to roughly 330 kbps.

On-chip digital signal processing will help reduce the system power dissipation and enable more efficient use of the available bandwidth. The digital signal processing will enable digitized action potentials from multiple electrodes to be detected, time stamped, queued up and transmitted. Transmitting neural waveforms or even the maximum and minimum values of the spike will enable spike sorting to be performed in the outside world, and this can improve the performance of algorithms which decode the neural data. An additional power saving aspect of the digital signal processing would be to power down the RF circuitry while the transmit queue is filling up; this could result in drastic power savings.

A block diagram for the entire DSP system is shown in Figure 1. An incoming data stream would first have its mean removed,

eliminating offsets from the amplifier and ADC. A threshold would then be calculated based on signal statistics; this is necessary because noise levels vary from electrode to electrode and can vary with time. Spike detection would be performed using either the calculated threshold or a user-programmable threshold, for maximum flexibility. If a spike is detected, a 1-ms window of data containing the spike would be extracted. If there is an overflow condition in the FIFO, then only the maximum and minimum samples would be extracted from this data window for transmission. The data would then be time-stamped. The extracted data, time stamp and electrode ID would be queued up for transmission. The transmit queue would be formatted into an appropriate packet, then undergo channel encoding and transmission.

We will focus on developing the functional blocks for threshold generation and spike detection. Initially, we will assume only a single channel is being digitized. We will then investigate increasing the number of channels that can be supported by multiplexing the digital signal processor or by including multiple instances of single channel processors.

2. RESEARCH ASPECT

Because the digital signal processing subsystems are being developed for a biomedical implant, the power consumption of the circuits should be kept to a minimum to prevent tissue heating and cell necrosis. The digital signal processing blocks will thus be optimized for low power. The amount of area consumed by these blocks is also limited. We will approach the goals of low power and small area on an algorithmic level and in the implementation level.

In wired and wireless experiments, threshold generation is typically done by computing the rms value of a sliding time-window of the digitized neural data, multiplying this value by some factor K , and using this as a threshold [2, 3]. Windows of 250 ms or longer are typical [2]. The time window shifts after each sample, and the threshold is updated. This can be expensive in terms of the computation of the square and square root and the buffering of the data in the time window: all the samples must be stored so that with each update, the oldest sample can be subtracted and the newest sample added. We are proposing a simpler computation to generate the threshold. Using the mean of the absolute value rather than the rms value generates a roughly equivalent metric without expensive computations. If the mean is taken over a number of samples that is a power of 2, division can be accomplished efficiently by shifting.

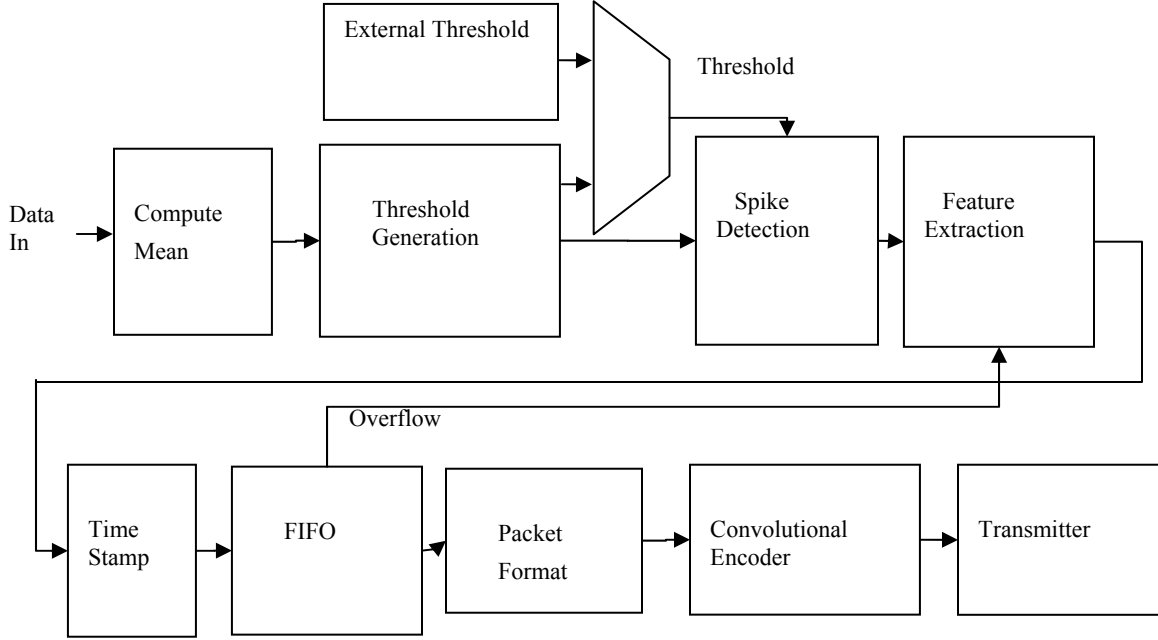


Figure 1: Block diagram of the digital signal processing system

Also, in order to reduce the buffering of the data used in the threshold calculation, we are proposing to update the threshold once per window rather than once per sample. During the course of the window, the absolute values of the samples are accumulated. At the end of the window, the mean is taken by shifting, and the multiplication by K is performed. This value of the threshold is used for the entire next time window; the accumulator then gets reset and the process starts over. This eliminates all buffering of the samples.

For spike detection, many algorithms have been proposed, including matched filtering and/or computation of a nonlinear energy operator. Researchers at Duke University have performed a simulation study showing that the absolute value operator is just as effective as these much more sophisticated algorithms [4]. We have thus chosen to use the absolute value operator on the data because this is computationally very simple.

To implement these algorithms, we will explore low power digital design options. We will explore the use of a low supply voltage, since the digital signal processing blocks do not need to operate at high frequencies. We will also consider the use of pass gate logic, since it has been claimed that this can offer 50% power savings compared to static logic [5].

3. BLOCK LEVEL DESIGN

3.1 Threshold Generation

Thresholds will be calculated according to equation (1):

$$Threshold = K \left(\frac{1}{N} \sum_{i=1}^N |x_i - \mu| \right) \quad (1)$$

First the mean will be subtracted from the data; the mean will be computed in a separate block. This will eliminate the offsets of the amplifier and ADC. Then the absolute value of the difference will be taken. This value will be added to an accumulation register. After the N samples have been accumulated, the average will be taken by dividing by N . Since we are constraining N to be a power of two, this can be accomplished by right-shifting the sum by the $\log_2(N)$ bits. This value will be multiplied by K and used as the threshold for the data for the subsequent window's worth of samples. We are using $K = 8$, which eliminates a multiplier and works reasonably well as a threshold. Details are presented in the Golden Model section below. We have chosen $N = 16384 = 2^{14}$; at 15 kSamples/sec, this gives us a time window of 1 second, which works reasonably well in simulations.

During the first window after the block has been reset, not enough data has accumulated for a threshold to be set according to data statistics. We arbitrarily set the threshold for this first window to a reasonably high value. Otherwise, there is the potential for a large number of false positives to happen and possibly cause overflow to occur.

3.2 Spike Detection

Spike detection is performed by comparing the absolute value of the data with the threshold. The absolute value of the data was

computed in the threshold generation block, and so it will be shifted into the spike detection block along with the data itself. If no spike is detected, the data will be buffered. A four sample buffer will be continuously updated so that once a spike is detected, the pre-threshold portion of the spike will also be transmitted. If a spike is detected, a counter is started and one millisecond worth of samples, including the pre-threshold samples in the buffer, will be passed to the next block. A spike detected/data output valid signal will also be set.

4. DESIGN FLOW

The design flow for these modules is as follows. First, the Golden Model is developed in Matlab. The Golden Model uses stimulus files which are binary vectors stored in a text file. The Golden Model outputs are also written to text files. This enables easy reuse of stimulus vectors and comparison of outputs with the behavioral model, which is implemented in Verilog. The modules are synthesized using Design Compiler and the UofU_Digital_v1_1 library, which is intended for AMI's 0.5 μm CMOS process. Automatic Place and Route is accomplished using Silicon Ensemble. Finally, post-layout mixed-mode simulations are accomplished using Cadence's Virtuoso Analog Environment, SpectreS and Verilog.

5. GOLDEN MODEL

The golden models for the threshold generation block and spike detection block have been developed in Matlab. Additional Matlab scripts have been developed to generate test vectors for the golden model and the behavioral model. Test vectors can be generated that consist of either synthetic waveforms, random data, or pieces of actual neural data. The neural data was recorded from the motor cortex of a rhesus monkey at the Neural Prosthetic Systems Laboratory at Stanford University, using a commercially available Cerebrus Neural Data Acquisition System. Using actual neural data was important for testing the automatic threshold generation, since we are proposing a simpler algorithm and its performance depends on statistics of neural signals.

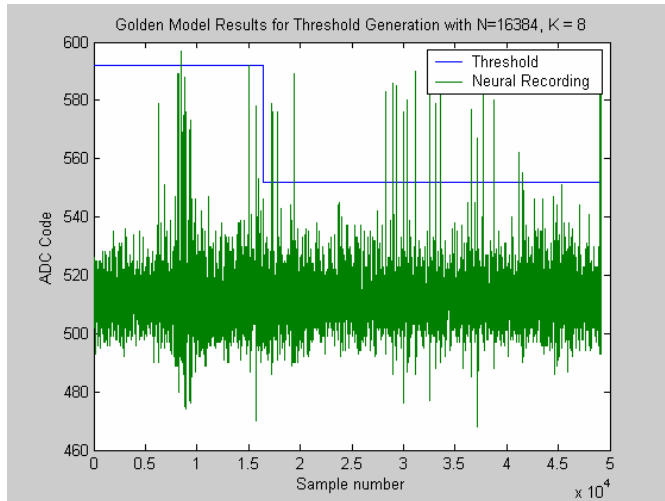


Figure 2: Golden model for threshold generation block with $N=16384$, $K=8$. Note that threshold for first window is reset to 80 above the mean.

Results of the Matlab golden model for the threshold generation block are presented in Figure 2. This simulation used $N = 2^{14}$ and $K = 8$. It can be seen that the automatically generated threshold does a good job of rejecting noise and detecting the larger action potentials. However, a number of smaller action potentials are not detected.

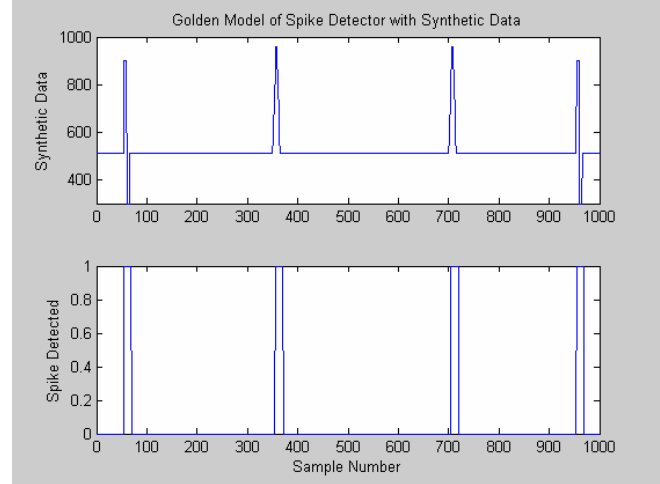


Figure 3: Golden model for the spike detection block using synthetic data.

In Figure 3, we have applied a data stream consisting of four synthetic spike waveforms to the golden model of the spike detection block. These data, shown in the upper plot, will be used for the golden model testing of both the spike detection and feature extraction blocks. In the lower plot, we see the "Spike Detected" signal which stays high for 16 samples once a spike is detected. This signals the feature extraction block to store these samples, including the four samples before the threshold was crossed.

6. BEHAVIORAL MODEL AND VERIFICATION

Behavioral Verilog code has been written and tested for the threshold generation and spike detection blocks. The code was tested using Verilog XL in a GUI-based environment. Matlab was used to generate the test vectors in the appropriate format. The test bench reads the test vectors and applied them as stimulus to the module. The test bench also writes the outputs to a text file for ease of validation against the golden model.

Test vectors consist of neural data streams consisting of 10 bit samples each. The data streams consist of easily recognizable synthetic waveforms, random data, or actual pre-recorded neural data. For ease of debugging and for simulation time considerations, the behavioral models are operating on windows much shorter than the ultimate autothreshold window length of 2^{14} samples.

Initially, synthetic neural data were used to validate the Golden Model. These same data were used with the behavioral code as well; finally, randomly generated test data were used. The simulations using the random test data uncovered a number of discrepancies between the behavioral and golden models, which were then corrected. In particular, the synthetic data never had a spike occurring immediately after another spike. The behavioral

code initially would not detect this spike because there was a 1 clock cycle reset phase. This was corrected so that there is no delay.

7. SYNTHESIS RESULTS AND VERIFICATION

Synthesis has been accomplished for the AutoThreshold and Spike Detection modules using Design Compiler. Power and area estimates created during synthesis are presented in Table I below. The synthesized structural verilog code has been simulated and compared against the Golden Model, with both synthetic data and random data. The synthesized circuit matched the behavioral model and golden model. The AutoThreshold module consists of 6428 transistors. The Spike Detection module consists of 3775 transistors.

Table I: Area and Power Estimates

	Area (μm^2)	Power (μW)
Initial Estimate	250000	200
AutoThreshold Synthesis	610017	99.1
Spike Detection Synthesis	362525	26.8
Total Synthesis	972542	125.9

8. DESIGN FOR TESTABILITY METHODOLOGY AND RESULTS

To enable testing of the system, full-scan design [6] is being used. All of the flip flops are scan-flops, consisting of a D flip-flop with a multiplexer on its data line. This enables the user to shift in/out desired states and verify the outputs. Full-scan design also facilitates delay fault testing. Full-scan design is possible because the associated performance penalty is irrelevant, since we will be clocking the system at 2 MHz maximum. There is some increase in area and power associated with the use of scan flops instead of normal Dflip-flops. This is shown in Table II below.

The scan chain has been incorporated into the Verilog behavioral description of the modules and then synthesized. The scan chain required the addition of ScanIn, ScanShift, and ScanOut signals to the modules. At each clock edge, if ScanShift is high, all of the registers act as shift registers; the order of the registers in the scan chain is specified in the Verilog code. If ScanShift is low, all the registers act as they did before the scan chain was included in the module. Simulations were performed to verify that the inclusion of the scan chains did not adversely affect the behavior of the modules.

Table II: Area and Power Estimates

	Area (μm^2)	Power (μW)
Initial Estimate	250000	200
AutoThreshold Synthesis	610017	99.1
AutoThreshold + Scan	850270	127.2
Spike Detection Synthesis	362525	26.8
Spike Detection + Scan	535065	47.9
Total, with Scan	1385335	175.1

9. PLACE AND ROUTE

Automatic Place and Route of the AutoThreshold and Spike Detection modules was accomplished using Silicon Ensemble. The results are shown in figures 4 and 5 below. The AutoThreshold module is 602.4 μm by 597 μm . The maximum achievable density was 80%. The Spike Detection module measures 448.8 μm by 447 μm . The maximum achievable density was 90%.

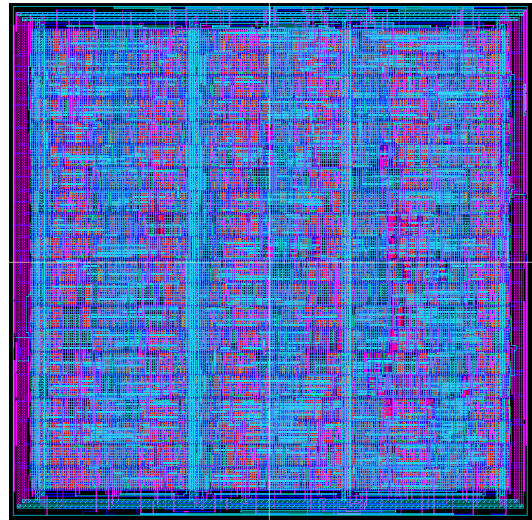


Figure 4: AutoThreshold Module

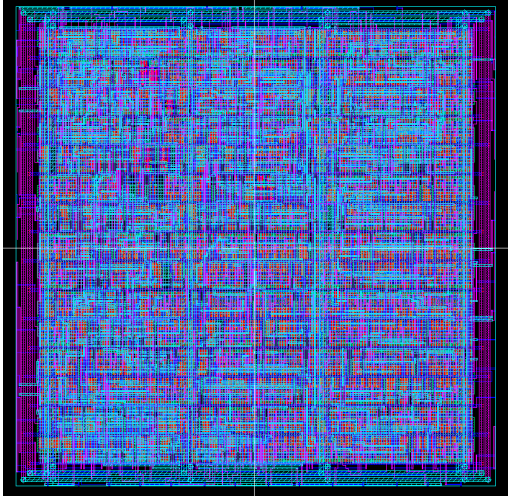


Figure 5: Spike Detection Module

Table III: Converging Area Estimates

	Initial	Synthesis	P&R
AutoThreshold		850270	359632.8
Spike Detection		535065	200613.6
Total	250000	1385335	560246.4

10. LOW POWER SUPPLY SIMULATIONS

We proposed to investigate two approaches to low power digital design: lowering the power supply voltage and using pass transistor logic. To explore lowering the power supply voltage, mixed signal simulations were performed in the Cadence tool suite. The synthesized version of the Spike Detection module was imported into Cadence. We placed two inverters on each input and output of the Spike Detection module. The module and the inverters nearest it were simulated in the analog domain using SpectreS. The inverters one step removed from the module were simulated using Verilog XL. In this manner, the test fixture previously developed for Verilog simulations could be used to apply the correct stimuli.

Because of the long time required to run analog simulations, we only simulated the design with one set of ten random inputs. The random inputs almost certainly do not include the worst-case input vector, so it is important to be conservative in lowering the power supply so that the worst-case scenario will still function.

We then performed a binary search sweep with the power supply voltage, starting at 3.3 V. The Spike Detection module did not work at 0.8 V, it worked marginally at 1 V, but it works fine at 1.2 V. We believe the 1.2 V supply gives it enough margin to operate at the 2 MHz clock rate. Power estimates for the blocks operating at 1.2 V are included in Table III below. These were calculated from the synthesis estimates using equation (2).

$$P = f C V_{dd}^2 \quad (2)$$

Table IV: Power Estimates

Power Supply	3.3 V	1.2 V
AutoThreshold + Scan	127.2 μ W	16.8 μ W
Spike Detection + Scan	47.9 μ W	6.3 μ W
Total	175.1 μ W	23.1 μ W

We also proposed investigating pass transistor logic for low power design. We chose to use static CMOS gates rather than pass transistor logic for a number of reasons. First, pass transistor design is much more labor intensive than the standard CMOS design flow, since the pass transistor circuits would need to be designed and laid out by hand. Second, we have found that the expected power savings from pass transistor logic are disputed. Apparently, much of the publicity was due to a number of preliminary studies that only compared full adder circuit implementations. When a broader variety of circuits are compared, static CMOS was found to be superior in most situations [7].

After place and route had been accomplished, we repeated the mixed mode simulations using the circuit extracted from layout, including parasitic capacitance. The simulations were also run at all of the process corners. This was done to verify the circuit still performed at 2 MHz at the low supply voltage, and to estimate the actual power consumption. The spike detection module operated correctly at all process corners. The power consumed was measured for the typical corner; it is presented in Table V with the (scaled) synthesis estimates for comparison. It can be seen that the simulated power consumption is much higher than was predicted by the synthesis tools.

Table V: Power Estimates for the Spike Detection Module

	Synthesis	Simulation
Vdd = 3.3 V	47.9 μ W	710 μ W
Vdd = 1.2 V	6.3 μ W	49.4 μ W

The nominal threshold voltages for the 0.5 μ m process are 0.7 V for nMOS transistors and -0.9 V for pMOS transistors, so we have not entered the subthreshold region of operation at 1.2V. The synthesis tool power estimates consist of dynamic power only; no leakage terms are included. However, in the 0.5 μ m process, leakage should be negligible, so this should not account for the discrepancy. The simulated power also includes the power consumed by the driving and loading inverters. This should not make a large difference. Another hypothesis to explain the discrepancy was that the clock driver was too slow, so the gates were switching very slowly and thereby suffering from high crowbar current. This proved not to be the case, because simulations run with more powerful clock drivers did not show reduced power consumption in the module.

11. INTERFACING AND PHYSICAL DESIGN ISSUES

In order to test the modules individually and together, we will require a number of pins. The global pins include clock, reset, and the DFT scan chain signals ScanIn, ScanShift, and ScanOut. Top level inputs and outputs to these modules include DataIn<9:0>, Mean<9:0>, DataOut<9:0>, DataValidOut, and CountOut<3:0>. For debugging and testing, we will use the scan chain to view values of internal registers, such as the threshold and the accumulator registers in the AutoThreshold module.

12. EXTENSION TO A MULTICHANNEL SYSTEM

The digital signal processing modules we have developed are for a single channel system. Our power estimates have been made by clocking the single channel system 100X faster than necessary, to estimate the power consumption of a 100 channel system. Because of the area consumed by the single channel system, the best way to expand this system for multiple channels would be to multiplex the hardware and clock it faster, instead of having dedicated hardware for each channel. This would require a 100:1 multiplexer on the input of the system, as well as additional registers and multiplexers inside the modules.

13. CONCLUSION

We have developed two modules of a low-power digital signal processing system for an implantable, wireless neural recording system. To reduce the power consumption of these modules, the algorithms were simplified as much as possible. Static CMOS was chosen over pass transistor logic because this simplified the design flow greatly, and the power savings of pass transistor logic are disputed. Power consumption was further reduced by lowering the power supply voltage to 1.2 V. This is possible

because the maximum clock frequency required by the modules is 2 MHz. Validation of the modules was accomplished by means of mixed-mode simulations of the schematics and of the extracted circuits.

14. REFERENCES

- [1] R. Harrison, P. Watkins, R. Kier, R. Lovejoy, D. Black, R. Normann, and F. Solzbacher, "A Low-Power Integrated Circuit for a Wireless 100-Electrode Neural Recording System." *Digest of Technical Papers, IEEE International Solid State Circuits Conference (ISSCC)*, pp. 554-555, February, 2006.
- [2] K. S. Guillory, R. A. Normann, "A 100-channel system for real time detection and storage of extracellular spike waveforms," *J. Neuroscience Methods*, vol. 91, pp. 21-29, 1999.
- [3] R. Olsson and K. Wise, "A three-dimensional neural recording microsystem with implantable data compression circuitry," *ISSCC Dig. Tech. Papers*, pp. 558-559, 2005.
- [4] I. Obeid, P. D. Wolf, "Evaluation of spike-detection algorithms for a brain-machine interface application," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 6, pp. 905-911, 2004.
- [5] A. Chandrakasan, W. Bowhill, F. Fox, *Design of High-Performance Microprocessor Circuits*, New York, IEEE Press, 2001, p. 136.
- [6] A. Chandrakasan, W. Bowhill, F. Fox, *Design of High-Performance Microprocessor Circuits*, New York, IEEE Press, 2001, p. 530.
- [7] R. Zimmerman, W. Fichtner, "Low-Power Logic Styles: CMOS Versus Pass-Transistor Logic," *IEEE Journal of Solid-State Circuits*, vol. 32, no. 7, pp 1079-1090, July 1997.