Low-Power, Intelligent Sensor Hardware Interface for Medical Data Preprocessing

Fei Hu, Shruti Lakdawala, Qi Hao, *Member, IEEE*, and Meikang Qiu, *Senior Member, IEEE*

Abstract—This work proposes an interface design of a low-power programmable system on chip for intelligent wireless sensor nodes to reduce the overall power consumption of the heart disease monitoring system, by lending them the capability of processing complex functions and performing rapid computations on a large amount of data at the node. This facilitates the node to intelligently monitor a medical signal for impending events instead of transmitting the signal to the base station constantly. Lowering the transmission data rate decreases the transmission power consumption in a node, thereby lengthening the node life and in turn increasing the reliability of the network. This work also implements a thresholding technique, which controls the data transmission rate depending on the value of the monitored signal, and a cardiac monitoring system that performs computations at the node for the detection of either a skipped heart beat or a reduced heart rate variability, in which event the signal is transmitted to the base station for monitoring/recording or alerting the crew. The performance analysis of the system shows that there are reductions in the system power consumption and data transmission rate, which in turn reduces the network traffic and averts congestion.

Index Terms—Biomedicine, energy-efficiency, hardware interface, sensor hardware design, wireless sensor networks.

I. INTRODUCTION

WIRELESS sensor networks (WSNs) consist of a number of sensor nodes that are deployed over a region for wirelessly monitoring real world data. Ongoing developments in wireless communication, integrated circuit design and microelectromechanical systems have resulted in smaller, low-power, low-cost sensors and electronic devices, which make WSNs a reality. A single tiny, untethered node relies on its limited local resources and has minimal capabilities. However when a number of these tiny, untethered nodes are sprinkled in a region, they are able to extend their capabilities through advanced networking protocols to hop data autonomously from one node to another before reaching the distant destination. Collectively, these nodes have a substantial processing capability. WSNs are increasingly versatile with the miniaturization of sensor nodes and can be used for a wide range of applications [1]. WSNs have

F. Hu and Q. Hao are with the Department of Electrical and Computer Engineering, The University of Alabama, Tuscaloosa, AL 35487 USA (e-mail: fei@eng.ua.edu; qh@eng.ua.edu).

S. Lakdawala is with the Department of Computer Engineering, The University of Rochester, Rochester, NY 14627 USA.

M. Qiu is with the Department of Electrical and Computer Engineering, The University of New Orleans, Orleans, LA 70148 USA (e-mail: mqiu@uno.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.
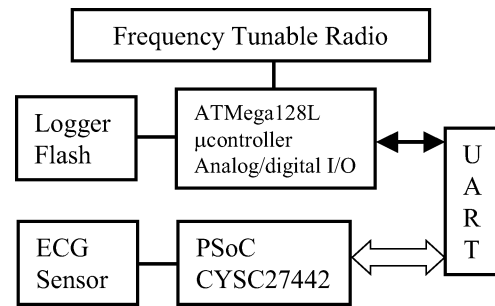
Fig. 1. PSoC mixed-signal interfaced sensor platform.

enormous potential benefits in the area of physiological monitoring and biomedical applications. It achieves patient mobility that facilitates the physiological study of patients or subjects during their normal activities. Remote physiological monitoring for vital signs, such as blood pressure, electrocardiogram, pulse, and body temperature, enable patients to receive a better medical care and helps maintain their health. It is useful for diagnostic and research purposes whereby, a collection of data from subjects is required over a long period of time. A good example is the study of heart rate variability (HRV) [2] of subjects over a period of 24 h for the study of the risk of heart attacks. We have designed cardiac sensor networks [3]–[5] and used a simulator to generate all types of irregular heart beat signal patterns.

For applications requiring detailed monitoring over a long period of time using wireless sensor nodes, the finite power budget of individual nodes is a primary design constraint. Key to increasing power efficiency in individual nodes is to reduce the amount of transmitted data, as the radio transmission consumes a major percentage of total power. Since the computation cost is several orders lower than the communication cost at individual nodes [6], local processing of the monitored data points and transmitting fewer data bits over the network especially over a long distance would save a substantial amount of power. Usually, the computational capabilities within individual nodes are limited and the sensor node requires external custom amplification or filtering circuitry.

This paper proposes a novel approach of interfacing a highly versatile programmable system on chip (PSoC) mixed-signal array with a wireless sensor node MICA2 from Crossbow Inc., as shown in Fig. 1. The purpose is to extend the computational capabilities of the node, while keeping its power consumption in check. A PSoC mixed-signal array is a low-power PSoC that allows programming of analog and digital (mixed-signal) components that are typically used in embedded system. It also has an inbuilt microcontroller, which integrates and controls all of the programmed components. With the extended

computational capabilities of this mixed-signal array, complex filtering and triggering functions as well as application specific data compression or suppression algorithms can be implemented at individual nodes. This can further reduce the data to be transmitted over the network, resulting in reduced transmission time and significant reduction in network traffic.

In this work, the reduction of transmission data is achieved on a thresholding principle: the raw sensor readings are compared to a threshold value. If the reading is above the threshold, it is transmitted to the base station immediately; however, if it is below the threshold, it is transmitted after being averaged out with other sensor readings below the threshold. The sensor readings below the threshold are thereby suppressed, reducing transmission data and saving power consumption in communications.

Moreover, an application of PSoC interfaced sensor node in physiological monitoring is developed and analyzed, which further proves that the low-power PSoC's computational capabilities helps reduce transmission data and network traffic to a substantial level. It measures the HRV using a statistical method and monitors the cardiac signal for skipped beats as well. HRV refers to the alterations in the beat-to-beat heart rate [7]. Reduced HRV is used as a marker of unhealthy conditions. The PSoC continuously processes the cardiac signal and computes the HRV and at the same time keeps looking for a skipped heart beat. Only when a heart beat is skipped or when the HRV drops, the sampled cardiac signal is sent back to the base station, reducing the transmission workload over the network.

The rest of this paper is organized as follows. Section II reviews the related work in wireless sensor-based cardiac monitoring. Section III presents experimental hardware and software platforms. Section IV investigates the intelligent wireless sensor interface design between PSoC and MICA2. Section V describes the application of a PSoC interfaced MICA2 for a cardiac monitoring system. Section VI shows experimental results and analyzes the improved system performance. Section VII concludes the paper and outlines future works.

II. RELATED WORK

Wearable, unobtrusive devices that monitor the patients for vital signs are currently being studied extensively. WSN has an enormous potential in this area. Harvard University is currently working on project "Codeblue" [8], which is developing hardware as well as software platforms of medical sensor networks using motes supplied by Crossbow Inc., San Jose, CA. They have also developed medical sensors such as EKG [9], pulse oximeter sensor, and motion activity sensor based on the popular MicaZ and Telos mote designs. It performs real-time data collection on the patients' physiological status, which is monitored and stored at the base station.

They are also studying the issues related to wireless medical-sensor networking, such as node mobility, patterns of packet loss, variations in data rate, etc. During their study, it was observed that the packet reception ratio, i.e. the number of received packets divided by the number of transmitted packets [8], decreased with the increase in the data rate, with an exception of a single hop where the reception ratio remains pretty good

up to a data rate of 50 packets per second. A similar trend is observed with multiple senders with a single receiver. This is because of limitation of the available bandwidth for each node. Also, increased data rate result in packet queues on each node and might eventually force packets to be dropped. In their study, it is observed that for a high reception ratio, a low data rate is encouraged, of about five packets per second or less for ten senders. This could be acceptable in case of blood pressure or pulse oximetry sensors, but not for an ECG monitoring system, which requires sending constant packets of the cardiac signal samples.

In this work, the whole system design is optimized for cardiac data compression and low-power wireless transmission. We demonstrate that by interfacing a PSoC to MICA2, a wireless ECG monitoring system can be developed that sends out data only when the cardiac signal might seem abnormal, or send back the estimates of HRV. This will reduce the network traffic after lowering the data transmission rate of an ECG monitoring device.

III. HARDWARE AND SOFTWARE PLATFORMS

A. Wireless Sensor Hardware and Software Platform

Our design is based on a tiny wireless sensor platform, called Mica2, created by Crossbow Inc. [10]. It uses an Atmel mega 128L microprocessor as its central processor, a Chipcon CC1000 tranceiver, and an external flash memory of 512 KB. The pins of the microcontroller are brought out to a 51 pin expansion connector, of which some of them are analog I/O's that are inputs to an analog to digital converter (ADC). The sensor board can hence directly connect the analog equivalent of the sensing element to the ADC of the microcontroller. It is powered on by 2 AA batteries.

TinyOS [11] is an operating system designed specifically for WSN. The 4 KB of RAM space, which is shared among the stack, global variables, and all the static variables, puts a very tight memory constraint on the system. Therefore, it is designed to minimize the code size and facilitate faster implementation using a modular architecture. TinyOS as well as its customized application on MICA2 are written in NesC programming language. It supports a programming model that allows component-based programming and event-driven execution, thereby supporting a flexible concurrency model.

B. PSoC Hardware Platform

The PSoC family supplied by Cypress consists of an array of configurable analog and digital blocks with a central microprocessor. It is very useful in building customized designs on one chip. PSoC architecture has a central processor, Data SRAM, flash program memory, configurable analog and digital blocks, programmable I/O ports, clock generator, and some other system resources. The M8C microprocessor is a powerful processor with an 8-bit Harvard architecture and can operate on speeds as high as 24 MHz. The general purpose I/O pins (GPIO) allow great flexibility for external interfacing, because each pin's drive mode can be selected from eight options and it

also has an option to generate an interrupt in case of predefined events at a pin. The PSoC family has different device groups based on the digital/analog pin count, the program memory, data memory and the number of configurable digital/analog blocks. These device groups are further divided into different parts depending on the package type and pin count. In this work, we use PSoC chip CY8C27443, 28 pin PDIP package, which has eight configurable digital blocks and 12 configurable analog blocks. It has 256 bytes of RAM and 16 KB of program memory. The range of supply voltage for PSoC varies from 3.0 V to 5.25 V.

C. PSoC Software Platform

The PSoC design software needs to be different from those used for programming traditional fixed function microprocessor because of the configurable analog/digital hardware blocks. The PSoC designer integrated development environment (IDE) developed by Cypress Microsystems, San Jose, CA, is an innovative software development environment for PSoC. It is a GUI-based design suite that simplifies the designing of the configurable blocks, provides the libraries for developing customized application code, and also provides advanced debugging tools using an emulator to support the programming of the PSoC chip.

The PSoC application code is developed in three main stages: (1) Device editor (DE) allows the user to choose the functional component (e.g. ADC, filter, timer, counter, etc.) for each of the analog/digital blocks. (2) Application editor (AE) allows the user to write the code either in C language or assembly language. The analog/digital blocks must be initialized in the code using the generated component libraries. It also gives access to the interrupt service routine. (3) Debugger: Debugging is the last step in the development process of PSoC. PSoC designer IDE provides an in-circuit-emulator, which allows debugging at the full operating speed of PSoC. It allows the user to define complex breakpoint events and large trace buffers that allow the user to monitor registers, memory locations, etc.

IV. INTELLIGENT WIRELESS SENSOR "INTERFACE" DESIGN

This section will discuss our implementation of the interface between MICA2 and PSoC in detail. It focuses on the hardware setup and programming of the devices. Both MICA2 and PSoC support universal asynchronous receive/transmit (UART) serial communication. MICA2 supports a baud rate of 57 600 bits per second. The UART pins in the MICA2 mote is brought to the 51 pin connector. However, since this is a development phase, direct soldering on the machine soldered components of MICA2 sensor node is avoided. Instead a breakout board is used, which brings out the signals at the 51 pin connector. The breakout board (the serial gateway MIB510CA) is made available by Crossbow Technology Inc. [10]. The serial gateway provides a RS232 interface, converting the UART signals to RS232 using MAX-3223I chip. It has a 9-pin RS232 connector. This board is generally used for interfacing the mote with the base station or for programming the motes, using its onboard processor.

A. Programming PSoC for UART Serial Communication

For the PSoC to be interfaced with MICA2, its digital blocks need to be programmed to support UART communication at a baud rate of 57 600 bps. This section describes in detail the programming of digital and analog blocks of PSoC for sampling the sensor signal and then transmitting it serially to MICA2.

1) Amplifier: The programmable gain amplifier (PGA) uses only one peripheral (analog) block. It is used to provide high input impedance to the transducer. In our design, the PGA is programmed for unity gain, the reference is selected as a GND, and the analog bus is disabled. For a unity gain, the output MUX is connected to the top of the resistor string and feedback to the inverting input. The transfer function of the PGA is therefore as follows:

$$V_O = (V_{IN} - V_{GND}) \left(1 + \frac{R_b}{R_a}\right) + V_{GND}. \qquad (1)$$

Equation (1) is compatible to any standard amplifier, where the input voltage is V_{IN}, output is V_O, V_{GND} is the ground reference, R_b and R_a are the amplifier resistors. The DE of the PSoC automatically programs the appropriate resistor values (R_b and R_a) depending on the user-defined gain value.

2) Analog to Digital Converter: The PSoC has a number of different ADC's to select from depending on the required sampling rate, resolution, SNR, and the number of peripheral analog/digital blocks available. It offers incremental type (algorithmically equivalent to a dual slope type), successive approximation as well as delta–sigma ADC's.

In our design, a variable resolution incremental ADC has been selected, because of its flexibility to finely tune the sample rate and change the resolution as per the application requirement during the development stage. It uses integrator, comparator, reference signals, counter, and a pulse width modulator (PWM). In our design the ADC has been programmed for a resolution of 11 bits and the data clock 2.4 MHz. It uses one analog peripheral block and three digital peripheral blocks for 8-bit counter and 16 bit PWM.

3) UART Blocks: UART blocks use two digital peripheral blocks, one for receiving and the other for transmitting. It is required to be programmed for a bit rate of 57 600 bps for interfacing with MICA2. The input clock to the UART blocks needs to be eight times the required baud rate. Hence, the clock is 460.8 KHz. The transmit interrupt request is enabled; therefore, once a transmit register is empty the microcontroller is interrupted to send more data.

4) DE Placing and Routing: The UART blocks, PGA, and the ADC are placed in the DE. They are programmed to route the signals correctly. The output of the PGA is routed to the input of the ADC, and the UART receives and transmits signals that are routed to the port pins. The global resources and the user-parameters for each of the programmed components are defined in the DE.

5) UART to RS232 Conversion at PSoC End: The UART voltage signals from the PSoC chip vary between 0 V and 5 V as per the TTL/CMOS voltage levels. This needs to be converted to RS232 levels that vary between 12 V and −12 V. RS 232 is

a more reliable serial communication protocol; because it has a greater noise margin.

B. MICA2 Programming

The remote mote must be programmed such that it is able to receive the data serially from the PSoC and transmit it wirelessly to the base mote. On the other hand, the base mote must be programmed such that it is able to receive data sent by the remote mote and transmit the data to the computer. TinyOS has a built-in application code (called TOSBase) that can bridge the serial and wireless channel. Whenever it receives a packet serially through UART, it transmits it wirelessly to the other mote, or, when it receives a radio packet wirelessly, it transmits it serially, in most applications to a base station.

C. Base Station Setup

At the base station to which the base mote is connected serially, the computer must continuously listen to the serial port for the received packets. It must be able to extract the sampled data from the packets. TinyOS developers have provided with a Java-based program, called SerialForwarder [35] that listens to the specified serial port at a user-defined baud rate. Another Java-based program, called Oscope [35], is provided to the users. This program is run in conjunction with the SerialForwarder. It retrieves the packets from the SerialForwarder and is able to correctly decipher the TOS message structure. It then displays the sampled data in a graphical display. In our base station setup, we program the SerialForwarder to listen the serial port at a baud rate of 57 600, since that is the serial baud rate of MICA2.

V. ENGINEERING APPLICATIONS OF INTELLIGENT SENSOR INTERFACE

This section discusses the application of a PSoC interfaced MICA2. It includes two parts: a thresholding principle and a cardiac monitoring system.

A. Thresholding Implementation

Thresholding is used to reduce the transmission data by suppressing the data below a predetermined threshold value. In our prototype design, ten sensory signal samples are saved in an array. These samples are then compared to the threshold value. If all the samples are below the threshold, their average is computed and added to the data payload. Once the entire payload is filled, the packet is ready to be sent. However, if any sample in the array is above the threshold, the entire array of all ten samples are sent immediately to the base station. It does not require programming of extra peripheral blocks. Such a system, where the transmission rate is controlled by the value of the element being monitored, is useful in the implementation of an auto-controlled indoor environment where a quick action is required to be taken to bring the sensed element back below the threshold.
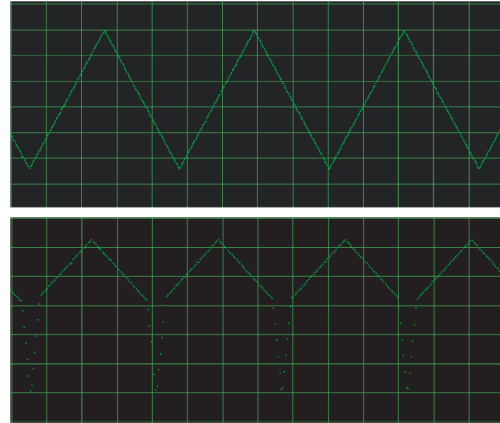

Fig. 2. Illustration of data samples. Upper: before thresholding; Lower: after thresholding.

The sample rate of the implemented system is given by the following:

$$SampleRate = \frac{DataClock}{2^{Bits+2} + CalcTime} \quad (2)$$

where $DataClock = 2.4$ MHz, $Bits = 11$ bits, and $CalcTime = 114$. Hence, the sample rate is about 289 Hz. While above the threshold, at least 28 packets are required to be transmitted per second since each packet contains ten samples; however, below the threshold only three packets are required to be sent per second, because for every ten samples, one average value is computed. Therefore, a packet containing ten average values carries the information of 100 samples below the threshold. Such a system reduces a great deal in the data volume when the sample values are below the threshold.

Fig. 2 shows the data samples displayed in the Oscope GUI. The upper part shows the data samples before implementation, and the lower part shows the results after implementation of the thresholding technique. It can be seen that the number of packets with data values below the threshold is reduced by a considerable amount.

B. Cardiac Monitoring System

1) Concept of Monitoring the Cardiac Signal Within the PSoC: In case of a wireless cardiac monitoring system, the sampled cardiac signal would need to be transmitted to the base station constantly at a minimum rate of ten packets in 1 s (i.e. a sampling rate of about 100 Hz). If there is more than one patient monitored at the same time, then it would result in network congestion and the average reception ratio would begin to drop, and the cardiac signal would appear distorted at the base station, which makes it difficult to monitor the signal precisely. Hence, a cardiac monitoring system was conceptualized using the PSoC for monitoring either a skipped heart beat or for a drop in the HRV. Only after the occurrence of either event is the sampled cardiac signal transmitted to the base station. Such a system takes advantage of the analog front end of the PSoC and its low-power computational capabilities.
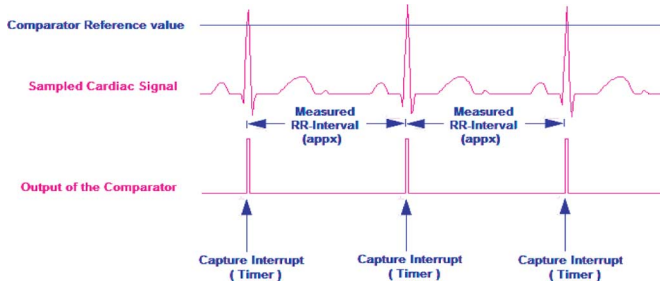
Fig. 3.   Illustration of measuring the RR-interval using PSoC.

The HRV of a cardiac signal is measured by computing the standard deviation (SD) of the RR-interval that varies periodically and therefore its SD must not be too low. If the SD reduces, then it is used as a marker for unhealthy conditions or a possible indication of an impending heart attack. The SD is defined as

$$\text{SD} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2} \qquad (3)$$

where n is the number of samples, x_i is the value of ith sample (or RR-interval), $\bar{x}$ is the average of all the samples (i.e. average of all the RR-intervals).

2) Measuring the RR-Interval: In our design, the R-wave is detected using an analog comparator. The reference value of the comparator is set such that it detects the positive edge of the R-wave, as shown in Fig. 3. The output of the comparator is fed to the capture input of a 16-bit timer. The timer is set to a capture interrupt, which copies the value of the count register to the compare register of the timer when it encounters a positive edge at the capture input. It also interrupts the microcontroller to read it. The timer continues to run; hence, when the next capture interrupt occurs, the user only needs to subtract the present captured value from the previous one to measure the elapsed time between the two interrupts, which is the RR-interval. In the system implementation, peripheral blocks such as an analog comparator and a 16-bit timer need to be programmed. The procedure is described as follows.

1) Measuring a Skipped Beat: To detect a skipped heart beat, a counter is used, which resets itself each time an R-wave occurs. If an R-wave does not occur for an interval of 1.92 s, a variable is flagged to indicate a skipped heart beat, and the sampled cardiac signal is immediately transmitted to the base station for recording/monitoring purpose. However, to measure an interval of 1.92 s, a different approach is utilized. Only one digital block is available after programming the other components; therefore, an 8-bit counter is programmed. The counter is set to interrupt approximately every 3.88 ms, using a clock of 32.967 KHz (derived from the system clock). A timeout variable is utilized to measure the interval of 1.92 s. It is decremented every 3.88 ms in the counter-ISR from an initial value of 500. When the timeout variable reaches zero, about 1.92 s would have elapsed. The timeout variable is set back to 500 within the GPIO ISR. The GPIO interrupt occurs during the positive edge of the comparator output (Fig. 3). The GPIO interrupt is generated by routing the comparator output to

one of the PSoC pins and enabling an (positive) edge triggered interrupt at the pin. This way the timeout variable will reach zero only when an R-wave fails to occur for a period of 1.92 s. Programming of the counter and the GPIO interrupt will be discussed next.

2) Placing and Routing in the DE: To implement the described cardiac monitor, an analog comparator, a 16-bit timer, and an 8-bit counter are placed. The output of the PGA (dark-green) is routed to the input of the comparator (purple), which detects the positive edge of the R-wave. The output of the analog comparator is routed to the capture input of the 16-bit timer (light blue) and the GPIO Port 0 [12]. The GPIO interrupt is enabled for a positive edge on Port 0. An 8-bit counter (light-green) is placed in the last available digital peripheral block. Both the timer and the counter are clocked by a 32.967 KHz clock, which is derived from the system clock using inbuilt clock dividers. The period of each clock is approximately 30 μs. Therefore, to set the counter to interrupt every 3.88 ms, a count of 127 (decimal) is loaded in the count register, and the terminal count interrupt is enabled. The count register is decremented at every positive edge of the clock, and the microcontroller is interrupted to jump to the Counter ISR when it reaches its terminal count of 0. The timer is initialized with a maximum count of 0xFFFF (65 535, decimal) in the count register. It is decremented with each clock. Since the timer interrupt is set to a capture interrupt, care should be taken that the capture input is asserted once every 65 535 clocks (i.e. within 1.96 s); otherwise, there will be an overflow when subtracting with the previously captured value of the timer-count-register. This problem is overcome using the counter, which gives indications in case that an R-wave does not occur for 1.92 s.

3) Application Editor: In the AE, each of the peripheral components must to be initialized and the ISR for the counter, timer, and GPIO interrupts need to be defined. For the calculation of SD, the measured values of RR-interval are stored in an array within the timer ISR. After ten values of RR-interval are saved in the array, their SD is calculated using (2). In our design, since the cardiac signal is being generated by a function generator, it has a constant RR-interval instead of having periodic variations. Therefore, an event detection was made possible.

4) Measuring SD Over a Period of 5 min: In order to analyze the HRV, a short-term interval of 5 min or long interval of 24 h is recommended for the calculation of SD of RR-intervals. For measuring the SD over an interval of 5 min, it would require enough memory to store at least 300 values (60 beats per minute is the heart rate of a trained athlete) of RR-interval so as to subtract individual RR-intervals from the calculated average. The PSoC family used in our design has only 256 bytes of RAM. Therefore, it might require a different family of PSoC, which has more RAM space provided. Otherwise, a different formula of SD should be used, which can allow us to compute the SD on the fly. The shortcut formula of SD is given here, based on the SD formula in (2).

$$\text{SD} = \sqrt{\frac{n\sum_{i=1}^{n}(x_i)^2 - \left(\sum_{i=1}^{n}x_i\right)^2}{n(n-1)}} \qquad (4)$$

Estimated PSoC Power Calculations for CY8C27xxx Family Device

Version 1.0						Enter parameters in yellow
project name	Thresholding					See estimated result in green
part number	CY8C27443		Cypress MicroSystems, Inc.			Indicates something is wrong
pin count	28					

| CPU Clock | 3.000 | MHz | | OpAmp Bias | 0 | |
| Vdd | 3.30 | V | | | | |

CPU Current	2.049	mA

Analog Block	Power Level (0-3)		
ACB00	2	0.600	mA
ACB01	2	0.600	mA
ACB02	0	0.000	mA
ACB03	0	0.000	mA
ASC10	0	0.000	mA
ASD11	0	0.000	mA
ASC12	0	0.000	mA
ASD13	0	0.000	mA
ASD20	0	0.000	mA
ASC21	0	0.000	mA
ASD22	0	0.000	mA
ASC23	0	0.000	mA
Analog Block Current		1.200	mA

Digital Block	Block Clk (MHz)			GPIO Count	GPIO Clk (MHz)		
DBB00	2.400	0.029	mA	0	0.000	0.000	mA
DBB01	0.000	0.000	mA	0	0.000	0.000	mA
DCB02	0.461	0.009	mA	1	0.461	0.021	mA
DCB03	0.461	0.009	mA	0	0.000	0.000	mA
DBB10	2.400	0.029	mA	0	0.000	0.000	mA
DBB11	2.400	0.029	mA	0	0.000	0.000	mA
DCB12	0.000	0.000	mA	0	0.000	0.000	mA
DCB13	0.000	0.000	mA	0	0.000	0.000	mA
Digital Block Current	0.107	mA		GPIO Current	0.021	mA	

A_Buffer Power	0

Row	Row Clk (MHz)			Analog Power				Analog Output Buffer			
Row_0_Output_0	0.000	0.000	mA	SC Off/Ref Low	0	0.000	mA	AnalogOutBuf_0	0	0.000	mA
Row_0_Output_1	0.000	0.000	mA	SC Off/Ref Med	0	0.000	mA	AnalogOutBuf_1	0	0.000	mA
Row_0_Output_2	0.461	0.014	mA	SC Off/Ref High	0	0.000	mA	AnalogOutBuf_2	0	0.000	mA
Row_0_Output_3	0.000	0.000	mA	SC On/Ref Low	1	0.385	mA	AnalogOutBuf_3	0	0.000	mA
Row_1_Output_0	0.000	0.000	mA	SC On/Ref Med	0	0.000	mA	Analog Output Buffer Current		0.000	mA
Row_1_Output_1	0.000	0.000	mA	SC On/Ref High	0	0.000	mA				
Row_1_Output_2	0.000	0.000	mA	Reference Circuit Current	0.385	mA		Estimated Total PSoC Current	3.774	mA	
Row_1_Output_3	0.000	0.000	mA								
Row Current	0.014	mA						Estimated Total PSoC Power	12.456	mW	

Fig. 4. Power consumption in PSoC for implementation of the thresholding algorithm.

where n is the number of samples, x_i is the value of ith sample (or RR-interval).

Using (4), we can calculate the SD on the fly for an interval of 5 min or more, because it only requires the current value of the RR-interval and does not require the average value of samples.

VI. RESULTS AND PERFORMANCE ANALYSIS

A. Thresholding Performance Results

Cypress MicroSystems has made a power calculator available to the PSoC users, which helps developers to estimate the power consumption in the chip. It depends on the number of peripheral blocks used, frequency of clock used for each of them, the number of pins and rows (on which the signals are routed) to be driven, etc. Based on our power calculator results, as shown in Fig. 4, we can estimate the current and power consumption for our thresholding and cardiac monitoring system. In the implementation of the thresholding concept, with a sampling rate of about 289 Hz, a resolution of 11 bits, a CPU clock of 3 MHz, a power supply of 3.3 V, a low reference voltage on the SC block (ADC), and driving only the ROW 0 output for serial transmission @ 0.461 MHz (8 times the baud rate), the estimated current consumption is about 3.774 mA.

Table I shows the comparison results between the implemented system with thresholding and the original system without thresholding. For a sample rate of about 289 Hz, about 28 packets are transmitted in 1 s. It assumes that the radio transmits the packets for about 80% of the time and is idle for 20% of the time. Model 1 is the implementation without the PSoC and Model 2 is the implementation with PSoC. Since only an average of ten samples is sent when the samples are below the threshold, we can see a reduction of 10% of packets while the signal is below the threshold. In Table I, we setup three cases to see how effective the designed thresholding algorithm in the PSoC interface is in terms of reducing sensor power consumption. Case I: Heart beating signals that are generated all below

TABLE I
EVALUATION OF POWER CONSUMPTION W/O PSoC INTERFACE

CASE I: 100% of the time below the threshold				
		Model 1		**Model 2**
Radio				
Current xmit	12mA	80%	9.6	8%	0.96
Current receive	8mA	0%		0%	
Current sleep	2μA	20%	0.0004	92%	0.0018
PSoC				
Active current	3.774mA	0%	0	100%	3.774
Current consumption:		9.6004		4.7358
Consumption reduced by:	50.67%			

CASE II: 50% of the time below the threshold				
		Model 1		**Model 2**
Radio				
Current xmit	12mA	80%	9.6	44%	5.28
Current receive	8mA	0%		0%	
Current sleep	2μA	20%	0.0004	56%	0.00112
PSoC				
Active current	3.774mA	0%	0	100%	3.774
Current consumption:		9.6004		9.05512
Consumption reduced by:	5.68%			

CASE III: 0% of the time below the threshold				
		Model 1		**Model 2**
Radio				
Current xmit	12mA	80%	9.6	80%	9.6
Current receive	8mA	0%		0%	
Current sleep	2μA	20%	0.0004	20%	0.0004
PSoC				
Active current	3.774mA	0%	0	100%	3.774
Current consumption:		9.6004		13.3744

Model 1: without PSoC; **Model 2**: with PSoC.

the detection threshold. Case II: For half of the experiment time, the generated sensing signals are below the threshold. Case III: No signals are below the threshold.

B. Cardiac Monitoring System

In order to test the SD calculated for the RR-interval of ten samples, the debugger is used along with the fast running emulator. The results are checked in the watch windows as shown in Table II. Note that in Table II, the ten samples of RR-interval are all around 0x8181 (33 153 in decimal). Since the clock to the timer has a period of about 30 μs, the measured RR-interval is about 33 153 × 30 μs = 0.9945 s, which is very close to the applied 1 Hz cardiac signal from the function generator. To test the system for a skipped beat, the frequency of the signal was reduced beyond 0.5 Hz. It was noticed that the remote mote immediately sent out the sampled cardiac signal to the base station, because it had not detected an R-wave for an interval of about 2 s. Initially, the timer interrupt did not read the RR-intervals correctly. This is because of bouncing at the output of the comparator (at the capture input of the timer). The bouncing of the signal was causing the capture interrupt to occur at every positive edge and record incorrect values of RR-interval. Hence

TABLE II
WATCH WINDOW IN THE DEBUGGER RUNNING THE CARDIAC
MONITORING SYSTEM

Watch/Global Name	
iRR_Interval10	0x8180, 0x8180, 0x8181, 0x8181, 0x8180 0x8182, 0x8181, 0x8183, 0x8181, 0x8182
iSample	0x0000, 0x0000, 0x0000, 0x0000, 0x0000 0x0000, 0x0000, 0x0000, 0x0000, 0x0000
k	0x0a
mean	33153.097656
std_dev	0.943401
total	0x00050f0b
variance	0.890005
variance_tot	8.900055
wait	0x2000

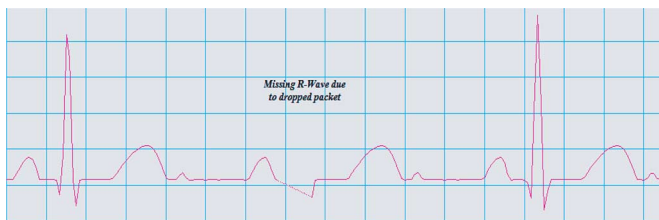Fig. 5. Power consumption in PSoC of the cardiac monitoring system.

Fig. 6. Distorted signal at the base station due to dropped packet.

for debouncing, a wait variable (in Table II) is used for applying a short delay after the first positive edge at the output of the comparator, and the impending interrupts are cleared.

The power consumption of the PSoC in a cardiac monitoring system is calculated using the power calculator, as shown in Fig. 5. For a system with a sampling rate of about 74 Hz, a resolution of 13 bits, a CPU clock of 12 MHz, a power supply of 3.3 V, a high reference voltage for the analog blocks, and driving only the ROW 0 output for serial transmission @ 0.461 MHz (8 times the baud rate), the estimated current consumption is about 10.939 mA. If the sampling frequency is increased beyond 74 Hz, which is about 7–8 packets in 1 s, the packets

start dropping, due to which the display or recording of cardiac signal at the base station gets affected. This is shown in Fig. 6.

VII. CONCLUSION AND FUTURE WORKS

This work presents the implementation of an interface between MICA2 motes and a low-power PSoC. It demonstrates its capability of performing complex event detection. Two applications, i.e. a thresholding algorithm and a cardiac monitoring system, were developed and analyzed for their performance. It was observed that PSoC's current consumption was less than that of CC1000's transmission current consumption in both the applications. Therefore, a PSoC interfaced MICA2 would help curtail the overall power consumption in an individual node, if the PSoC successfully facilitates a reduction in its transmission data. The reduction in transmission data in individual nodes also gives rise to fewer packets being transmitted up and down the network path, which in turn reduces network traffic, chances of congestion, and keeps the reception ratio high. This reduces the overall power consumption of the sensor network, which in turn increases the lifetime of individual sensor nodes and makes the network more reliable.

To commercialize this work, the PSoC interfaced MICA2 design should be further investigated to make it a more reliable physiological monitoring device. The threshold-based wireless transmission triggering mechanism is a simple processing scheme. We will add wavelet-based data mining algorithms in the PSoC interface to further reduce data transmissions. More sensors for measuring body temperature, blood pressure, and a patients pulse could be interfaced together along with the impending ECG sensor for the developed application. The PSoC sleep power is about 21.5 μW, while the Atmel microcontroller used in MICA2s active power is 33 mW and sleep power is 75 μW. A sleep/wake-up algorithm could be implemented in medical applications with low-sampling or low-transmission rates. The PSoC is a reconfigurable device, and it is flexible to change the threshold, sampling frequency, and the range of sensor readings, according to the sensing accuracy requirements.

REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey," *Comput. Netw. (Elsevier)*, vol. 38, pp. 1–20, 2006.
[2] E. Kristal-Boneh, M. Raifel, P. Froom, and J. Ribak, "Heart rate variability in health and disease," *Scan. J. Work Environ. Health*, vol. 21, pp. 85–95, 1995.
[3] F. Hu, M. Jiang, and Y. Xiao, "Low-cost wireless sensor networks for remote cardiac patients monitoring applications," *Wireless Commun. Mobile Comput.*, vol. 8, no. 4, pp. 513–529, 2008.
[4] S. Kumar, K. Kambhatla, F. Hu, M. Lifson, and Y. Xiao, "Ubiquitous computing for remote cardiac patient monitoring: A survey," *Int. J. Telemed. Appl.*, vol. 2008, pp. 1–19, Jun. 2008.
[5] F. Hu, M. Jiang, M. Wagner, and D. Dong, "Privacy-preserving telecardiology sensor networks: Towards a low-cost, portable wireless hardware/software co-design," *IEEE Trans. Inf. Technol. Biomed.*, vol. 11, no. 6, pp. 619–627, Nov. 2007.
[6] G. J. Pottie and W. J. Kaiser, "Embedding the internet: Wireless integrated network sensors," *Commun. ACM*, vol. 43, no. 5, pp. 51–58, May 2000.
[7] I. Kawachi and the Allostatic Load Working Group. Heart rate variability. [Online]. Available: http://www.macses.ucsf.edu/Research/Allostatic/notebook/heart.rate.html

[8] V. Shnayder, B. Chen, K. Lorincz, T. R. F. FulfordJones, and M. Welsh, "Sensor networks for medical care," Div. Eng. Appl. Sci., Harvard Univ. Tech. Rep. TR-08-05, 2005.

[9] T. R. F. Fulford-Jones, G.-Y. Wei, and M. Welsh, "A portable, low-power, wireless two-lead EKG system," in *Proc. 26th IEEE EMBS*, San Francisco, CA, Sep. 2004, pp. 51–58.

[10] (2005). The crossbow website. [Online]. Available: http://www.xbow.com/

[11] T. C. Forum. An open-source os for the networked sensor regime. [Online]. Available: http://www.tinyos.net

[12] R. Min, M. Bhardwaj, S.-H. Cho, E. Shih, A. Sinha, A. Wang, and A. Chandrakasan, "Low-power wireless sensor networks," Correspondence to Marek Malik, PhD, MD, Chairman, Writing Committee of the Task Force, Dept. Cardiol. Sci., St George's Hospital Medical School, Cranmer Terrace, London SW17 0RE, UK.
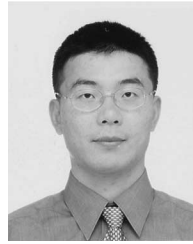
Qi Hao (M'06) received the B.E. and M.E. degrees from Shanghai Jiao Tong University, China, in 1994 and 1997, respectively, and the Ph.D. degree from Duke University, Durham, NC, in 2006, all in electrical and computer engineering.

Currently, he is an Assistant Professor at the Department of Electrical and Computer Engineering, The University of Alabama, Tuscaloosa, AL. His Postdoctoral training at the Center for Visualization and Virtual Environment, The University of Kentucky, was focused on 3-D computer vision for human tracking and identification. His current research interests include compressive wireless sensors, intelligent wireless sensor networks, and biomedical image processing.

Fei Hu (M'01) received the first Ph.D. degree in signal processing at Shanghai Tongji University, China, in 1999, and the second Ph.D. degree in electrical and computer engineering at Clarkson University, Potsdam, NY, in 2002.

Currently, he is an Associate Professor at the Department of Electrical and Computer Engineering, The University of Alabama, Tuscaloosa. His recent research interests include wireless networks, wireless security, and their applications in biomedicine. His research has been supported by NSF, Cisco, Sprint, and other sources. He has authored or coauthored more than 100 journal/conference papers and book (chapters). He is also the editor for over five international journals.

Shruti Lakdawala (M'04) is currently a graduate student in the Department of Computer Engineering, The Rochester Institute of Technology, NY.

His recent research interests include wireless networks, wireless security, and their applications in biomedicine.

Meikong Qiu (SM'08) received the B.E. and M.E. degrees from Shanghai Jiao Tong University, China, and the M.S. and Ph.D. degrees in computer science from The University of Texas at Dallas, Richardson, TX, in 2003 and 2007, respectively.

He had worked at Chinese Helicopter R&D Institute and IBM. From August 2007, he has been an Assistant Professor of electrical and computer engineering at The University of New Orleans, New Orleans, LA, and also serves as the Program Chair of IEEE EmbeddedCom'09 and EMCom'09. His research interests include embedded systems, computer security, and wireless sensor networks. He has authored or coauthored more than 60 papers.

He is an IEEE Senior member. He has been on various chairs and TPC members for many international conferences, such as IEEE SEC08 IEEE CSE08, IEEE ESO08, IEEE GlobeCom'08, and IEEE RTCSA09. He received the Air Force Summer Faculty Award 2009.