Energy Comparison and Optimization of Wireless Body-Area Network Technologies

Le Yan †, Lin Zhong ‡ and Niraj K. Jha †
†Department of Electrical Engineering, Princeton University, Princeton, NJ 08544
‡Department of Electrical and Computer Engineering, Rice University, Houston, TX 77005

Abstract

Wireless body-area networks (WBANs) have revolutionized the way mobile and wearable computers communicate with their users and I/O devices. We investigate an energy-efficient wireless device driver for low-duty peripherals, sensors and other I/O devices employed in a WBAN to communicate with a more powerful central device. We present an extensive comparative study of two popular WBAN technologies, 802.15.1 (Bluetooth) and 802.15.4 (ZigBee), in terms of design cost, performance, and energy efficiency. We discuss the impact of tunable parameters of the wireless device driver on connection latency and energy consumption for both Bluetooth and ZigBee. We address dynamic resource management in higher-level protocols by investigating the trade-off between connection latency and energy consumption. We propose an energy-efficient power-down policy that utilizes the interval between consecutive connection requests for energy reduction; we study an adaptive connection latency management technique that adjusts various tunable parameters dynamically to achieve minimum connection latency without changing the energy consumption level. Our measurements and experimental results show that these techniques are very effective in reducing energy consumption while meeting connection latency requirements.

Categories and Subject Descriptors: C.4 [Performance of Systems]: Design Studies.
General Terms: Performance, Measurement.
Keywords: Energy-efficient design, wireless body-area network, power consumption.

I. Introduction

A body-area network (BAN) is a computer network used for communications among computing and I/O devices within the physical reach of a human user or personal operating space. In recent years, there has been a significant increase in applications based on wireless BAN (WBAN) technologies, *e.g.*, IEEE Standards 802.15.1/Bluetooth [1] and 802.15.4/ZigBee [2], especially in wearable computing [3], [4], health monitoring [5]–[8], location awareness and identification [9], and smart objects [10]. A *computer-to-computer model* has been adopted for wireless peripheral devices in many WBANs, in which WBAN members have their own operating system (OS) to control wireless communication. For example, the Intel personal server [4] uses Bluetooth to communicate with existing computing infrastructure; the IBM Linux Watch [3] also supports Bluetooth. Both devices run Linux.

Acknowledgments: This work was supported by NSF under contract no. CCF-0428446.

The computer-to-computer model is suitable for scenarios in which multitasking is required, *e.g.*, when sensing and communication are performed in parallel. However, for smaller-scale applications, such as sensors and other information-capturing devices in a WBAN, a simpler *computer-to-device model* may be more suitable, especially in terms of energy efficiency and design cost, without the presence of an OS. In the computer-to-device model, a mobile computer serves as the center or host of a WBAN; other WBAN members serve the host as peripherals or slaves; the host OS treats its wireless peripherals in a fashion similar to wired peripheral devices. Wireless peripheral devices are already commercially available in the market. For example, wireless desktops including a mouse and keyboard [11] allow users to enjoy better maneuverability. However, such devices normally have proprietary and ad hoc designs. The computer-to-device model has not been well investigated. Its potential and limitations have not been fully explored.

In this work, we propose to investigate the computer-to-device model for a wireless device driver for low duty-cycle peripherals, sensors and other I/O devices in a WBAN. We believe this model is of great interest to the BAN community because wirelessly interconnecting body-worn computers, sensors and other I/O devices has posed a significant energy efficiency challenge. We first study design issues and application scenarios of the computer-to-device model, and then present case studies for two popular WBAN technologies, Bluetooth and ZigBee, to investigate the effect of tunable parameters in the wireless device driver on connection latency and energy consumption. Based on the case studies, we address dynamic resource management, including power management and adaptive connection latency management, in higher-level protocols by investigating the trade-off between energy consumption and connection latency. We make the following contributions in this work.

- To the best of our knowledge, our energy efficiency model is the first for wireless device drivers in the context of body-area wireless communication. The model is suitable for low duty-cycle peripherals and sensors on which multitasking is not necessary.
- We provide firsthand and extensive measurement data for the connection latency and energy consumption tradeoffs for Bluetooth and ZigBee, two popular WBAN technologies. We believe they will be invaluable in WBAN system design.
- We provide an extensive comparative study of Bluetooth and ZigBee in terms of performance and energy efficiency. We investigate the impact of tunable parameters in the wireless device driver on connection latency and energy consumption for both WBAN technologies.

The paper is organized as follows. In Section II, we define a system model for a wireless device driver in WBANs and discuss its application scenarios and design issues in

terms of connection latency and energy efficiency. We use a wireless wrist-watch to illustrate the application of the model and present our experimental setup. In Sections III and IV, we discuss how tunable parameters in the wireless device driver affect connection latency and energy consumption in the context of Bluetooth and ZigBee, respectively. In Section V, we present higher-level protocols for dynamic resource management. These protocols consider the trade-off between energy consumption and connection latency. We offer a comparative study between Bluetooth and ZigBee in Section VI. We present discussions in Section VII and conclude in Section VIII.

II. A Wireless Device Driver for WBANs

In this section, we first define the computer-to-device model of wireless device drivers for low-duty cycle peripherals and sensors in WBANs. We then present its design issues for communication protocols and address various application scenarios.

A. Computing model for a wireless device driver

The control operations of a wireless peripheral device are performed by code specific to the device. This code is called the *wireless device driver*. Fig. 1 shows the architecture of the proposed computing model for wireless device drivers in WBANs. A host can control multiple wireless peripheral devices. It typically has much more hardware resources available to it than the peripherals. In WBANs, the host is typically a mobile system, such as a handheld computer, a mobile phone, or a personal server [4]. The wireless device driver is part of its OS. The driver relies on wireless communication protocols for a reliable connection with the peripheral. It functions as an interface between the peripheral and host applications that need the peripheral. For example, the wireless device driver sends control commands to the peripheral upon an application request. The software on the wireless peripheral collects data, *e.g.*, from sensor readings, and sends them through wireless communication to the host. The wireless device driver then retrieves the data. Host applications can thus access the data through the OS. The model is also applicable to the case when two hosts collect the readings from the same sensor. The sensor can serve the hosts as a wireless peripheral device in a time-divided fashion. The model does not support multitasking.

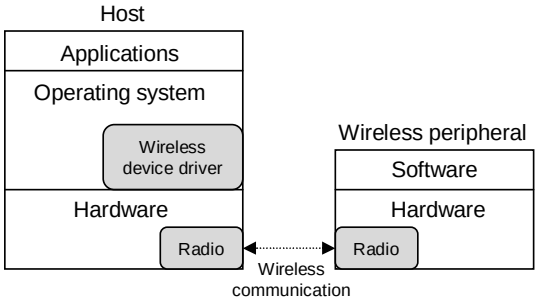


Fig. 1. Computing model for a wireless device driver in WBANs

The proposed model is different from other sophisticated computer-to-computer models. The computer-to-device model is interrupt-driven without control of the OS. It has lower hardware requirements and design complexity. On the other hand, in the computer-to-computer model, the OS on a wireless peripheral device controls its exchange of data with the host or other devices. The devices function as a computer. This model is suitable for scenarios in which multitasking is required. For example, in a wireless sensor network (WSN) for forest fire detection [12], a large number of sensor nodes are randomly deployed in a fire-prone forest to detect fires. The sensor nodes relay the exact origin of the fire to the end users. Meanwhile, they monitor the possibility of fire at their own locations. In this case, sensing and communication have to be done at the same time. However, for WBANs, low duty-cycle peripherals and sensors are deployed within the range of an individual and multitasking may not be necessary. Embedding an OS in wireless peripheral devices increases design cost.

B. Design issues for a wireless device driver

The wireless connection between the host and peripheral devices is enabled through communication protocols at different levels. Lower-level protocols, such as Bluetooth and ZigBee stacks, are typically responsible for a secure and reliable data exchange channel. The wireless device driver, however, needs to implement a higher-level protocol that interprets the data, when the peripheral conveys some information in the form of a data stream to the host. We use a simple byte-based communication protocol for the wireless device driver. The protocol is based on commands executed between the wireless device driver and its peripheral devices. It specifies the format of the communication command, as shown in Fig. 2. The communication command is demarcated by a header and a tail. Its type is specified by *command type*. Type I command is the *information* command, which updates the internal memory of the wireless device for display. It contains up to 176 bytes of *command data*, which specify not only the text to be displayed but also how it should be displayed. We will address other types of communication commands later.

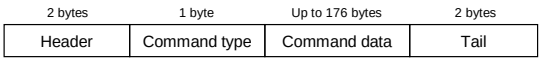


Fig. 2. Communication protocol implemented by a wireless device driver

For energy efficiency reasons, the wireless peripheral device is not always connectable. It switches its state between connectable and idle continuously, as shown in Fig. 3. This is called an *active session* in this work. The radio becomes connectable for T_{ps} seconds every T_c seconds. When the radio on the device is in the idle state, it will not respond to any connection request from the host, which leads to a longer connection latency, potentially as long as $T_c - T_{ps}$. We define *connectable ratio* as $\gamma = T_{ps}/T_c$. If $\gamma = 1$, that is $T_{ps} = T_c$, then the radio is always connectable, and thus the connection latency is minimized. However, the average power consumption of the wireless peripheral device in the active session, P_{active}, is given by Equation (1).

$$\begin{aligned} P_{active} &= \gamma * P_{connectable} + (1-\gamma) * P_{idle} \\ &= \gamma * (P_{connectable} - P_{idle}) + P_{idle} \end{aligned} \quad (1)$$

where $P_{connectable}$ and P_{idle} represent the power consumption when the radio is in the connectable and idle states, respectively. While increasing γ reduces the connection latency, it increases P_{active}. Thus, T_{ps} and T_c impact the connection latency and energy consumption significantly. For obtaining energy-efficient wireless communication, we can consider the trade-off between connection latency and power consumption by tuning parameters T_{ps} and T_c.

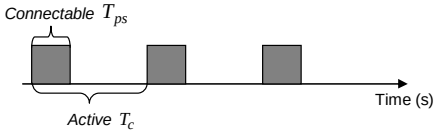


Fig. 3. Timing of an active radio session

C. Application scenarios

Wireless device drivers can not only be employed in computer peripherals, such as wireless keyboard, mouse, and headset for human-computer interaction, and universal remote control [13] for home entertainment system control, but also to control the light, lock, and curtain equipped with a wireless radio. However, we are especially interested in its applications to WBANs for wearable/mobile pervasive computing, including health monitoring [6], [14]. In a health monitoring system, patient information, *e.g.*, temperature and blood glucose level, can be measured by body-worn sensors. A handheld or mobile phone [8], acting as the host, collects health information through the wireless device driver. Applications running on the host can access the information from the wireless device driver. They can forward the information to medical professionals through Internet connectivity on the host.

Another wearable pervasive computing scenario is using a wireless wrist-watch as the secondary user interface between a handheld and its body sensor network [8]. While the IBM Linux Watch and Microsoft SPOT Watch [15] can be viewed as complete computer systems, the CacheWatch introduced in [16] runs as a dumb interface device without an OS. In this work, we use the CacheWatch concept to illustrate our computer-to-device model. Fig. 4 shows the hardware platform for a host and wrist-watch using a wireless transceiver for implementing the wireless device driver. We chose Sharp Zaurus SL-5600 [17], running Embedix Linux, as the host. Wireless transceiver A is attached to the Zaurus using an RS232 adapter. The Zaurus controls wireless transceiver A via an RS232 interface with 9600bps baud rate. The wrist-watch can display text messages, which may have different latency tolerances. Without an OS, it is a wireless peripheral device instead of a standalone computer. The watch is powered by a 3.6V supply with three AAA batteries. It is controlled by a microcontroller, PIC16F88. The software on PIC16F88 was developed using PicBasic Pro [18]. PIC16F88 drives the LCD directly for displaying information and controls wireless transceiver B through a UART interface with a 9600bps connection. It reads data from the UART and interprets them based on the wireless communication protocol discussed in Section II.B. MAX604, a voltage regulator controlled by PIC16F88, provides the power supply for wireless transceiver B. The wireless device driver is written in C++. It is responsible for the configuration and control of wireless transceiver B on the watch. It also collects data from this transceiver and relays them to the corresponding application on the host.

D. Experimental setup

We evaluate the wireless device driver based on several factors: connection latency L, energy consumption E_A of wireless transceiver A attached to the Zaurus, and energy consumption E_B of wireless transceiver B attached to the wrist-watch.

Connection latency is the interval between wireless transceiver A initiating a connection request and receiving an acknowledgment from wireless transceiver B. The wrist-watch typically seeks connection with the Zaurus when the user requests it. The time it takes to establish a connection is an important part of user experience. We will discuss it in detail in the context of Bluetooth and ZigBee in Sections III and IV, respectively. A C++ program is used to generate 15 connection requests randomly and measure the corresponding connection latencies.

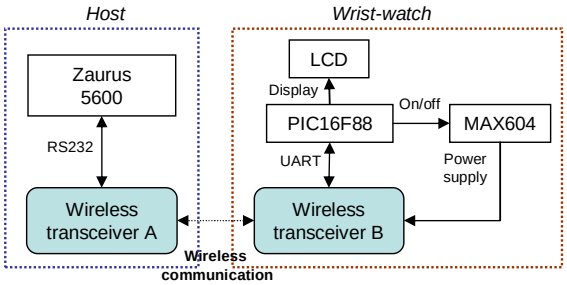


Fig. 4. An application example of the wireless device driver model

We measure power with an Agilent 34401A digital multimeter connected to a Windows-based PC via a GPIB cable. We obtain the power consumption by measuring current through a $R = 0.1\Omega$ sense resistor connected in series with the power supply to the wireless transceiver. We use a C++ program on the PC to sample the voltage drop V_R across the resistor at 220Hz. The program calculates the current I through the resistor based on $I = V_R/R$. It then calculates the power consumption P using $P = VI$, where V is 3.3V.

III. Case Study I: Bluetooth

In this section, we use a Bluetooth module as the wireless transceiver. We first discuss various features of the Bluetooth module. We then discuss the impact of different tunable parameters of the wireless device driver on the connection latency and energy consumption.

A. Promi-ESD class II Bluetooth module

The Promi-ESD class II Bluetooth module from Initium [19] is used in this work, which can be configured and controlled through a UART interface. The module conforms to Bluetooth Specification v1.1 [21]. Two Promi-ESD modules are used as wireless transceivers A and B, as shown in Fig. 4. Wireless communication using Bluetooth is connection-oriented. A Bluetooth device allows other devices to connect to it by entering the page scan mode. As shown in Fig. 5, page scan is conducted in short bursts, T_{pss} seconds every T_{cs} seconds. This session is called the *page scan session*, which corresponds to the *connectable session* mentioned in Section II. The connectable session is conducted for T_{ps} seconds every T_c seconds. To establish a connection, the Zaurus first sends a connection request to the attached Promi-ESD A using the AT connection command (ATD) via the RS232 interface. Promi-ESD A then enters the page mode, in which it transmits an ID packet directed at the intended Promi-ESD B attached to the wrist-watch. After it gets an acknowledgment from Promi-ESD B, it responds with a frequency hop synchronization (FHS) packet. On reception of the FHS packet, Promi-ESD B enters the connection state. Once the connection is established, Promi-ESD A sends a "CONNECT" message to the Zaurus. The delay between the Zaurus sending a connection request and receiving a "CONNECT" message is the connection latency L for the Bluetooth-based system used in this work.

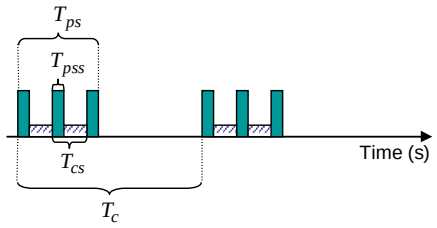


Fig. 5. Timing of Bluetooth page scan session

The power consumption of the connectable session can be represented as:

$$P_{connectable} = \tau * (P_{page\_scan} - P_{standy}) + P_{standby} \quad (2)$$

where P_{page_scan} and $P_{standby}$ are the power consumption of the Promi-ESD module when it enters page scan and standby modes, respectively. $P_{page\_scan} \geq P_{standby}$ and $\tau = P_{page\_scan}/P_{standby}$. Under $T_{pss} = 80$ms and $T_{cs} = 640$ms, $P_{connectable}$ and P_{idle}, as illustrated in Equation (1), are 43mW and 23mW, respectively.

B. Tunable parameters

For the wrist-watch using a Promi-ESD module, the tunable parameters are T_{pss}, T_{cs}, T_{ps}, and T_c. T_{pss} and T_{cs} can be changed by adjusting the S-registers on the Promi-ESD module, S41 and S42, respectively. The default values are $T_{pss} = 80$ms and $T_{cs} = 640$ms.

We first discuss the impact of Bluetooth-specific tunable parameters, T_{pss} and T_{cs}, on connection latency. Let Promi-ESD B be in the connectable session continuously, that is $T_{ps} = T_c$. Connection latency L determines how long it takes the Zaurus to establish a connection with the watch. Fig. 6 shows L under different values of T_{pss} and T_{cs}. It can be seen that for a given value of T_{pss}, *e.g.*, $T_{pss} = 40$ms, L decreases as T_{cs} reduces.

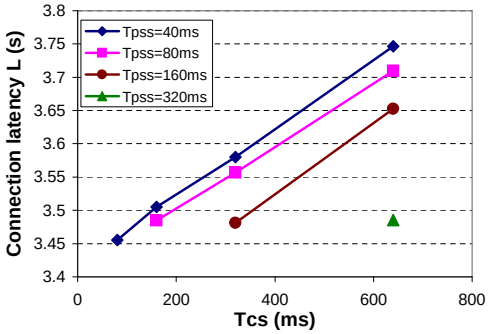


Fig. 6. Connection latency under different values of T_{pss} and T_{cs} on the Bluetooth-based system

Next, we discuss how tunable parameters, T_{ps} and T_c, affect connection latency and energy consumption. T_{pss} and T_{cs} are set to their default values, 80ms and 640ms, respectively. Fig. 7 shows L under different values of T_{ps} and T_c. It can be seen that for a given value of T_{ps}, *e.g.*, $T_{ps} = 2$s, L decreases as T_c decreases. E_A corresponding to $T_c = 3$s is smaller by 34.2% with respect to E_A under $T_c = 7$s, as shown in Fig. 8. E_B corresponding to $T_c = 3$s is smaller by 23.3% with respect to E_B under $T_c = 7$s. E_B decreases as T_c decreases, even though the power consumption of Promi-ESD B increases. This is due to the large reduction in L. For a given value of T_c, *e.g.*, $T_c = 4s$, L decreases as T_{ps} increases. E_A under $T_{ps} = 3$s is smaller by 35.4% with respect to E_A under $T_{ps} = 1$s. E_B under $T_{ps} = 3$s is smaller by 34.0% with respect to E_B under $T_{ps} = 1$s.

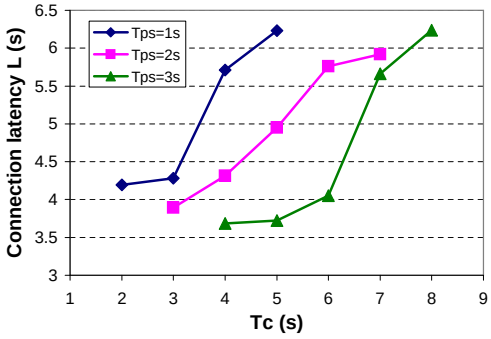
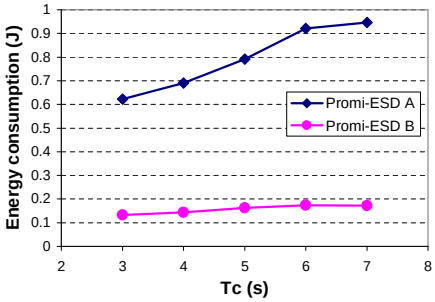
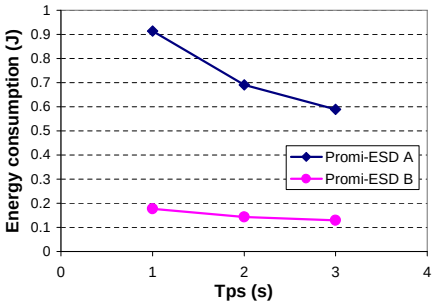


Fig. 7. Connection latency under different values of T_{ps} and T_c on the Bluetooth-based system



(a) $T_{ps} = 2$s



(b) $T_c = 4$s

Fig. 8. Energy consumption of Promi-ESD A and B under different values of T_{ps} and T_c

IV. Case Study II: ZigBee

In this section, we use a ZigBee module as the wireless transceiver. We first present the features of the ZigBee module. We then discuss the impact of different tunable parameters on connection latency and energy consumption.

A. ZigBee module

The Crossbow MICAz [22] is used as the ZigBee module. MICAz is the latest generation of Motes from Crossbow Technology. It uses the Chipcon CC2420 RF transceiver. It conforms to 802.15.4 [23] and runs in beacon mode under the control of TinyOS 1.1.7 [24]. The data rate is 250kbps when operating at 2.4GHz. The MICAz can be controlled through a UART interface on its 51-pin expansion connector. Two MICAz modules are used as wireless transceivers A and B, as shown in Fig. 4. As opposed to Bluetooth, MICAz has no connection establishment mechanism. It sends data

to the recipient by using the address of a specific MICAz or a broadcast address specified in the packet header. In this work, the Zaurus first instructs the attached MICAz A to send a packet to confirm that the radio on MICAz B is turned on. If the packet is received by MICAz B, the radio stays in the on state for the connection and an acknowledgment packet is sent back. On reception of the acknowledgment, MICAz A sends a message "CONNECT" back to the Zaurus. The delay between the Zaurus sending a confirmation request and receiving a "CONNECT" message is the connection latency L for the ZigBee-based system used in this work.

For energy efficiency, the radio on MICAz A is turned on for T_{ps} seconds every T_c seconds. When the radio is turned off, MICAz A is not connectable. The wakeup and shutdown latency of the radio is negligible (less than 1ms). $P_{connectable}$ and P_{idle}, as illustrated in Equation (1), are 84mW and 15mW, respectively.

B. Tunable parameters

For the wrist-watch that uses the MICAz module, the tunable parameters are T_{ps} and T_c. We discuss how both affect connection latency and energy consumption. Fig. 9 shows connection latency L under different values of T_{ps} and T_c. It can be seen that for a given value of T_{ps}, *e.g.*, $T_{ps} = 2$s, L decreases as T_c decreases. Energy consumption E_A of MICAz A corresponding to $T_c = 3$s is smaller by 85.7% with respect to E_A under $T_c = 7$s, as shown in Fig. 10. Energy consumption E_B of MICAz B corresponding to $T_c = 3$s is smaller by 75.1% with respect to E_B under $T_c = 7$s. For a given value of T_c, *e.g.*, $T_c = 4s$, L decreases as T_{ps} increases. E_A for $T_{ps} = 3$s is smaller by 63.4% with respect to E_A under $T_{ps} = 1$s. E_B for $T_{ps} = 3$s drops by 43.2% with respect to E_B under $T_{ps} = 1$s. The reason is that although the power consumption of MICAz B increases, the higher reduction in L results in a reduction in E_B. Therefore, L, E_A, and E_B decrease as γ increases.

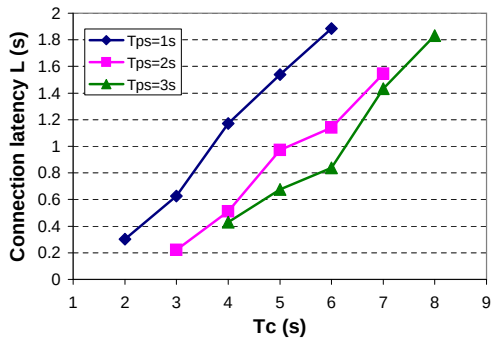


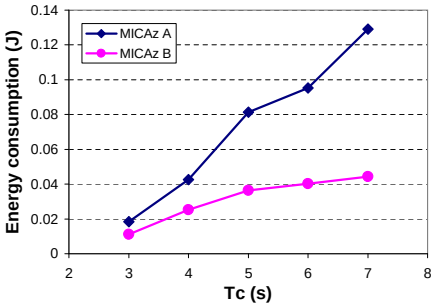
Fig. 9. Connection latency under different values of T_{ps} and T_c on the ZigBee-based system

V. Dynamic Resource Management

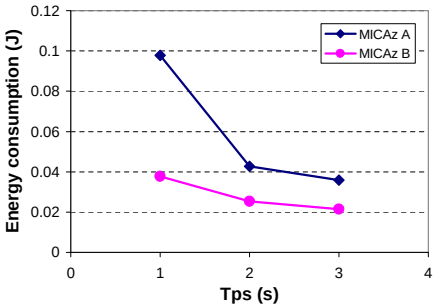
Based on the case studies of Bluetooth and ZigBee in Sections III and IV, respectively, we next propose dynamic resource management techniques that can be employed in both systems through higher-level protocols.

A. Energy-efficient power-down policy

After the host disconnects from the wireless peripheral device, if the device remains in an active session until the next connection request arrives, there is unnecessary energy consumption. Thus, the wireless peripheral device can be



(a) $T_{ps} = 2s$



(b) $T_c = 4s$

Fig. 10. Energy consumption of MICAz A and B under different values of T_{ps} and T_c

switched to the power-down mode for energy reduction. Assume the interval between two consecutive connection requests is Δt. Suppose the transition time overhead for being powered down and woken up are δ_d and δ_u, respectively. Similarly, the transition energy overhead for being powered down and woken up are ϵ_d and ϵ_u, respectively. Then, if the following conditions are satisfied, the wireless peripheral device can be powered down.

$$\Delta t > \delta_d + \delta_u \tag{3}$$

$$P_{active} * \Delta t > P_{down} * (\Delta t - \delta_d - \delta_u) + \epsilon_d + \epsilon_u \tag{4}$$

where P_{down} is the power consumption when the wireless device is powered down. In this work, P_{down} of the Promi-ESD and MICAz modules are 323μW and 466μW, respectively. The reduction in energy consumption E_B of wireless transceiver B, as shown in Fig. 4, can be expressed as:

$$\Delta E_B = P_{active} * \Delta t - P_{down} * (\Delta t - \delta_d - \delta_u) - \epsilon_d - \epsilon_u \tag{5}$$

A type II command, called the *management* command, is implemented to enable the host to power down the wireless peripheral device at run-time. The command data have information on the next connection schedule from the host to the wireless peripheral device. Given the connection schedule, the Zaurus can power down Promi-ESD/MICAz B using the management command. The Microchip PIC16F88 on the wrist-watch is programmed to wake up Promi-ESD/MICAz B before the next connection request arrives. If the next connection schedule is unknown, some prediction mechanism can be employed to predict the value of Δt. We employ *AVG(w)* prediction, which computes an exponentially moving average of past connection request arrival times as follows.

$$t_i = \frac{wt_{i-1} + m_{i-1}}{w + 1} \tag{6}$$

where w is a decay factor. t_i and m_i denote the predicted and measured values of the connection initiation time for the i^{th} connection, respectively. However, any inaccuracy in the prediction may affect the connection latency and energy consumption of modules on the host and wrist-watch. Let us consider one such connection request. Suppose the host initiates the connection request at time t. The predicted connection schedule is for time t_p. Suppose $t_{min} = \delta_d + \delta_u$. Let us consider four scenarios, as follows.

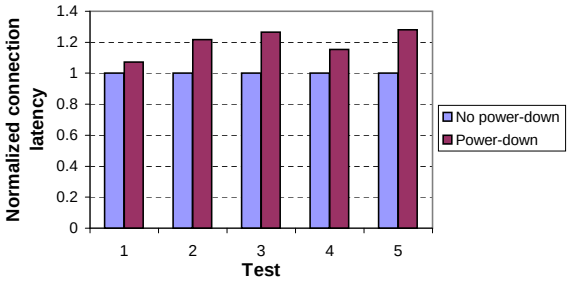
- $t_p = t$
  If the prediction is accurate, there is no impact on connection latency L and energy consumption E_B of wireless transceiver B.
- $t_{min} < t < t_p$
  If t_p is larger than t, the device is still in the power-down mode when the host initiates the connection request. Thus, with respect to the case when the prediction is accurate, L increases by $t_p - t$. Energy consumption E_A of wireless transceiver A increases by $(t_p - t) * P_{page\_scan}$. To reduce E_A, the host can start the paging process at time t_p, instead of t.
- $t_{min} < t_p < t$
  The device wakes up at time t_p, before the host sends the connection request. L and E_A may remain unchanged. However, E_B increases by $(t - t_p) * P_{down}$.
- $t_p \leq t_{min} < t$
  The device will not be powered down. L and E_A may remain unchanged. E_B increases by $t * P_{connectable} - (t - t_{min}) * P_{down} - \epsilon_d - \epsilon_u$.

Figs. 11 and 12 show the effectiveness of the power-down policies on the systems using Bluetooth and ZigBee, respectively. Each of Tests 1-5 initiates 15 connection requests from the host randomly. Two policies are investigated for the wireless device driver: no power-down policy and power-down policy with unknown connection request schedule. For the Bluetooth-based system, E_B reduces by 25.0% under power-down policy with respect to no power-down policy while the connection latency increases by 19.8%. Similarly, for the ZigBee-based system, E_B is reduced by 30.7% under power-down policy with respect to no power-down policy, while the connection latency increases by 26.7%.

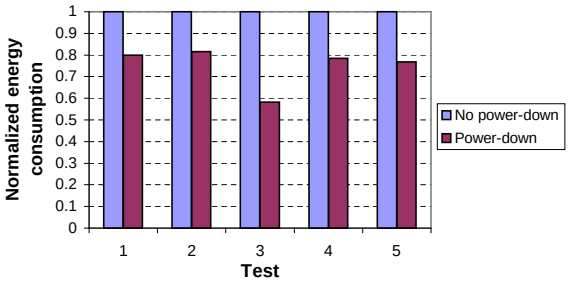
B. Adaptive connection latency management

As discussed in Sections III and IV, tunable parameters affect both the connection latency and energy consumption. Two types of commands are implemented to adjust tunable parameters dynamically. A type III command, called the *configuration* command, is implemented to adjust the timing parameters, T_{ps} and T_c, at run-time. The command data specify the values for T_{ps} and T_c. A type IV command, called the *Bluetooth-specific configuration* command, is implemented to specify the values of T_{pss} and T_{cs} for the Bluetooth-based system. The values must be slot-based (one slot equals 625μs) according to the specification of the Promi-ESD module. The time overhead of the above commands depends on current values of tunable parameters. For example, in the ZigBee-based system, it takes 0.3s on an average to switch from $T_{ps} = 1$s and $T_c = 2$s to $T_{ps} = 2$s and $T_c = 3$s. In the Bluetooth-based system, under $T_{ps} = 1$s and $T_c = 2$s, it takes 3.5s on an average to switch from $T_{pss} = 40$ms and $T_c = 80$ms to $T_{pss} = 80$ms and $T_c = 160$ms.

Given the same values of γ and τ (Bluetooth-specific), Promi-ESD and MICAz have the same level of power consumption, according to Equations (1) and (2). Table I shows the connection latency under different systems and parameters. For the same system with the same values of γ or τ, different tunable parameters result in different values of connection latency. For example, when $\gamma = 1/3$ on Promi-ESD B in the Bluetooth-based system, the connection latency under $T_{ps} = 1$s and $T_c = 3$s reduces by 5.8% compared to $T_{ps} = 3$s and $T_c = 9$s. When $\gamma = 1/2$ on MICAz B in the ZigBee-based system, the connection latency under $T_{ps} = 1$s and $T_c = 2$s reduces by 64.1% compared to $T_{ps} = 3$s and $T_c = 6$s. Therefore, given the level of power consumption of the wireless peripheral device, the tunable parameters can be adjusted dynamically to achieve a minimum connection latency.



(a) Connection latency



(b) Energy consumption E_B of Promi-ESD B

Fig. 11. Effectiveness of the power-down policy on the Bluetooth-based system

TABLE I

CONNECTION LATENCY UNDER THE SAME POWER CONSUMPTION LEVEL

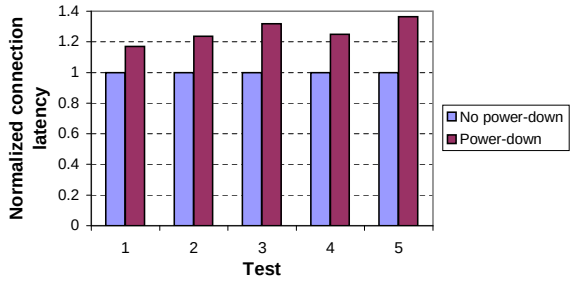
System	Parameters	L
Promi-ESD $\tau = 1/4$	$T_{pss} = 40$ms, $T_{cs} = 160$ms	3.51s
	$T_{pss} = 80$ms, $T_{cs} = 320$ms	3.56s
	$T_{pss} = 160$ms, $T_{cs} = 640$ms	3.65s
Promi-ESD $\gamma = 1/3$	$T_{ps} = 1$s, $T_c = 3$s	4.19s
	$T_{ps} = 2$s, $T_c = 6$s	4.31s
	$T_{ps} = 3$s, $T_c = 9$s	4.45s
MICAz $\gamma = 1/2$	$T_{ps} = 1$s, $T_c = 2$s	0.30s
	$T_{ps} = 2$s, $T_c = 4$s	0.51s
	$T_{ps} = 3$s, $T_c = 6$s	0.84s

VI. A COMPARATIVE STUDY

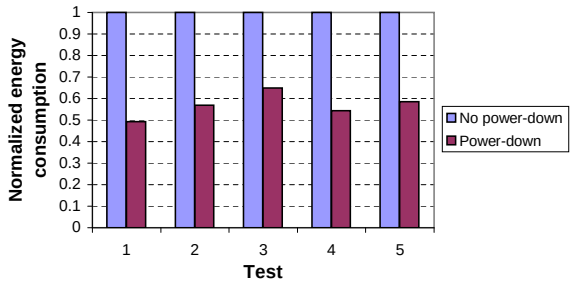
In this section, we present a comparative study of Bluetooth and ZigBee in terms of connection latency and energy consumption.

A. Same tunable parameters

We first compare Bluetooth and ZigBee in terms of connection latency and energy consumption for the same tunable



(a) Connection latency



(b) Energy consumption E_B of MICAz B

Fig. 12. Effectiveness of the power-down policy on the ZigBee-based system

parameters, T_{ps} and T_c, of the wireless device driver.

Connection latency is important for user experience in WBANs. Fig. 13 shows a comparison of the normalized connection latency for the Bluetooth-based and ZigBee-based systems with the same values of T_{ps} and T_c. T_{pss} and T_{cs} of the Promi-ESD module remain the default values ($T_{pss} = 80$ms and $T_{cs} = 640$ms). It can be observed that the MICAz module establishes a connection between the Zaurus and wrist-watch much faster than the Promi-ESD module. For example, when $T_{ps} = 2$s and $T_c = 4$s, the Promi-ESD module takes 4.3s to establish a connection, while the MICAz module takes 0.5s. For the different values of T_{ps} and T_c shown in Fig. 13, the connection latency using the MICAz module is on an average 87.0% smaller with respect to the Promi-ESD module. Thus, the MICAz module has a faster connection establishment mechanism compared to the Promi-ESD module with the same values of T_{ps} and T_c.

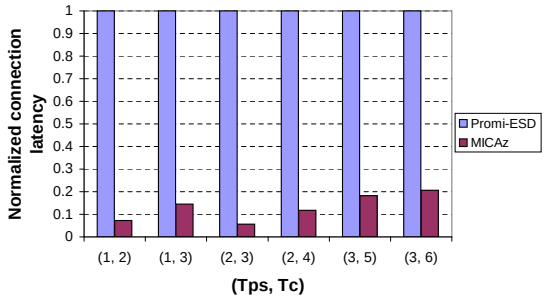


Fig. 13. Comparison of connection latency with same T_{ps} and T_c

Power consumption is also a critical issue in WBAN design. Fig. 14 shows the normalized power consumption of the Bluetooth and ZigBee modules when they are in an active session with the same values of T_{ps} and T_c. It can be seen that the Promi-ESD module consumes less power than the MICAz module. For example, the power consumption of the Promi-ESD module in an active session is 33.2mW when $T_{ps} = 2$s and $T_c = 4$s. However, the power consumption of the MICAz module is 49.5mW, which is 49.1% larger compared to the Promi-ESD module. For the different values of T_{ps} and T_c shown in Fig. 14, the power consumption of the MICAz module in an active session is 43.7% larger on an average with respect to the Promi-ESD module. Thus, with the same values of T_{ps} and T_c, the Promi-ESD module is more power-efficient than the MICAz module.

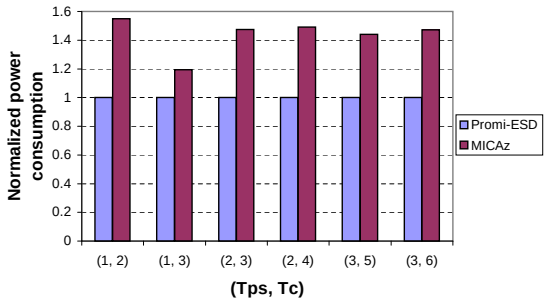


Fig. 14. Comparison of power consumption in an active session with same T_{ps} and T_c

B. Different tunable parameters

Next, we discuss Bluetooth and ZigBee in terms of connection latency and energy consumption with different values of tunable parameters, T_{ps} and T_c.

Fig. 15 shows the connection latency under the same power consumption level in an active session on Bluetooth and ZigBee systems. The same power consumption level corresponds to different values of T_{ps} and T_c, as shown in Table II. Obviously, under the same power consumption level, the MICAz module yields smaller connection latency than the Promi-ESD module. For example, the power consumption in an active session of the Promi-ESD module with $T_{ps} = 1$s and $T_c = 2$s and the MICAz module with $T_{ps} = 1$s and $T_c = 4$s are the same, 32.3mW. However, the connection latency using the MICAz module is 1.2s, compared to 4.2s using the Promi-ESD module. This yields a reduction of 71.4% in the connection latency. For the cases shown in Fig. 15, the connection latency using the MICAz module is on an average 72.0% smaller with respect to the Promi-ESD module under the same level of power consumption. Therefore, the MICAz module provides a more energy-efficient transmission mechanism for small data packets than the Promi-ESD module in WBANs.

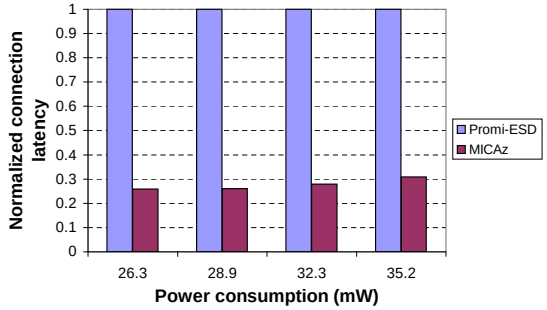


Fig. 15. Comparison of connection latency under the same power consumption level in an active session

VII. Discussions

The proposed computer-to-device model is suitable for scenarios when low duty-cycle peripherals and sensors are

TABLE II

(T_{ps}, T_c) CORRESPONDING TO THE SAME POWER CONSUMPTION LEVEL

Power consumption	Promi-ESD	MICAz
26.3mW	(2s, 8s)	(1s, 5s)
28.9mW	(1s, 3s)	(3s, 8s)
32.3mW	(1s, 2s)	(1s, 4s)
35.2mW	(3s, 4s)	(2s, 6s)

deployed within the range of an individual, and multitasking may not be necessary. It is interrupt-driven without the control of the OS, and thus has lower hardware requirements and design complexity. On the other hand, the computer-to-computer model has an OS to control its peripherals and sensors, which increases design cost and energy consumption. It is suitable for scenarios in which multitasking is required.

We presented firsthand measurement data for two commercial Bluetooth and ZigBee modules when used in wireless device drivers. Bluetooth has a higher bandwidth and better availability than ZigBee on mobile devices. It has been widely used in commercial products, such as mobile phones. The integration of Bluetooth technology into mobile products is more advanced than ZigBee. ZigBee is designed to provide a lower power consumption than Bluetooth but for WSN applications. With tunable parameters set to the same values in an active session, we found that the Bluetooth module, Promi-ESD, consumes less power during an active session, while incurring a higher connection latency compared to the ZigBee module, MICAz. Under different values of tunable parameters, both modules can achieve the same level of power consumption in an active session. However, the ZigBee/MICAz module takes less time to establish a connection, which can provide better user experience than the Bluetooth/Promi-ESD module.

Although our measurements and observations were made using two specific implementations of Bluetooth and ZigBee, we believe they represent the state-of-the-art for both WBAN technologies. It is important to note that MICAz is a complete sensor module and is more than a ZigBee-Serial adapter. However, the difference in power consumption between Promi-ESD and MICAz is primarily due to the RF receiver for ZigBee and the processor in MICAz that runs the 802.15.4/ZigBee protocol stacks. Therefore, we believe the power comparisons between Promi-ESD and MICAz are representative of Bluetooth and ZigBee.

The recent announcement of the Wibree Radio technology [25] by Nokia and its partners will introduce a new possibility in low-power body-area communication. Since Wibree is particularly targeted at low duty-cycle short-range communication, we expect most of the proposed higher-level energy optimization technologies can be readily applied to Wibree-based body-area devices.

VIII. CONCLUSIONS

WBANs play an important role in the deployment of wearable/mobile pervasive computing systems. In this work, we presented a computing model for a wireless device driver for low duty-cycle peripherals, sensors, and other I/O devices in a WBAN. The proposed model is useful for many applications, such as wearable computing, home entertainment, and health monitoring. We discussed its design issues in terms of higher-level communication protocols based on standard WBAN technologies: Bluetooth and ZigBee. Several communication commands, such as information, management, configuration, and Bluetooth-specific configuration commands, were implemented to adjust multiple tunable parameters of the wireless device driver dynamically, which impact both connection latency and energy consumption, as shown in the two case studies using Bluetooth and ZigBee. Given the power consumption level, the adaptive connection latency management technique can achieve a minimum connection latency. The energy-efficient power-down policy we introduced can reduce the energy consumption further.

REFERENCES

[1] IEEE 802.15.1, *Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Wireless Personal Area Networks (WPANs)*, IEEE, Mar. 2002.

[2] Zigbee™Alliance, http://www.zigbee.org.

[3] C. Narayanaswami, N. Kamijoh, M. Raghunath, T. Inoue, T. Cipolla, J. Sanford, E. Schlig, S. Venkiteswaran, D. Guniguntala, V. Kulkarni, and K. Yamazaki, "IBM's Linux watch: The challenge of miniaturization," *IEEE Computer*, vol. 35, no. 1, pp. 33–41, Jan. 2002.

[4] R. Want, T. Pering, G. Danneels, M. Kumar, M. Sundar, and J. Light, "The personal server: Changing the way we think about ubiquitous computing," in *Proc. Int. Conf. Ubiquitous Computing*, pp. 194–209, Sept. 2002.

[5] E. Jovanov, D. Raskovic, J. Price, J. Chapman, A. Moore, and A. Krishnamurthy, "Patient monitoring using personal area networks of wireless intelligent sensors," in *Proc. Symp. Annual Rocky Mountain Bioengineering*, pp. 373–378, Apr. 2001.

[6] Personal health monitoring system, http://www.phmon.de/englisch/.

[7] N. F. Timmons and W. G. Scanlon, "Analysis of the performance of IEEE 802.15.4 for medical sensor body area networking," in *Proc. Conf. Sensor & Ad Hoc Communications & Networks*, pp. 16–24, Oct. 2004.

[8] L. Zhong, M. Sinclair, and R. Bittner, "A phone-centered body sensor network platform: Cost, energy efficiency and user interfaces," in *Proc. IEEE Body Sensor Network Wkshp.*, Apr. 2006.

[9] Hewlett Packard iPAQ Bluetooth GPS receiver, http://www.hp.com.

[10] F. Siegemund and T. Krauer, "Integrating handhelds into environments of cooperating smart everyday objects," in *Proc. European Symp. Ambient Intelligence*, pp. 160–171, Nov. 2004.

[11] Microsoft wireless optical desktop pro, http://www.microsoft.com.

[12] I. F. Akyildiz, Y. S. W. Su, and E. Cayirci, "Wireless sensor networks: A survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002.

[13] J. Nichols and B. A. Myers, "Studying the use of handhelds to control smart appliances," in *Proc. Int. Conf. Distributed Computing Systems Wkshp.*, p. 274, May 2003.

[14] K. Ouchi, T. Suzuki, and M. Doi, "Lifeminder: A wearable healthcare suppport system using user's context," in *Proc. Int. Conf. Distributed Computing Systems Wkshp.*, p. 791, July 2002.

[15] Microsoft SPOT, http://www.microsoft.com.

[16] L. Zhong and N. K. Jha, "Energy efficiency of handheld computer interfaces: Limits, characterization, and practice," in *Proc. Int. Conf. Mobile Systems, Applications, & Services*, pp. 247–260, June 2005.

[17] Sharp Zaurus SL-5600, http://www.sharpusa.com.

[18] PicBasic Pro compiler, http://www.melabs.com/products/pbp.htm.

[19] Initium Promi-ESD Class II, http://www.initium.co.kr.

[20] AT command sets, http://www.modem.com/general/extendat.html.

[21] Bluetooth®, http://www.bluetooth.org.

[22] Crossbow MICAz ZigBee Series, http://www.xbow.com.

[23] IEEE 802.15.4, *Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)*, IEEE, Oct. 2003.

[24] TinyOS, http://www.tinyos.net.

[25] Wibree Radio Technology, http://www.wibree.com/.