Idle Time Power Management for Personal Wireless Devices

Nevine AbouGhazaleh, Robert N. Mayo, Parthasarathy Ranganathan
Mobile and Media Systems Laboratory
HP Laboratories Palo Alto
HPL-2003-102
September 17th , 2003*

E-mail: nevine@cs.pitt.edu, (Bob.Mayo, Partha.Ranganathan) @hp.com

wireless,
power,
energy,
802.11

Energy Consumption is becoming a critical concern that directly affect the mobility and lifetime of mobile and embedded systems. Different embedded and mobile applications place different demands on the wireless communication system and thus the energy consumption of the wireless cards. Past work has studied managing the energy consumed in wireless cards; we sought to evaluate some of these techniques by implementing them on our system and measuring the actual energy consumption by the wireless card.

We tested two schemes. The first increases the card's sleeping duration allowing power to be reduced by putting the card in low power states for longer times. The second technique is more aggressive; the power manager completely shuts-down the card during idle periods. When applying these two techniques for an email application, we get 25% energy savings for extending the sleep time up to 60 sec, while the savings reaches 89% when shutting down the card for the same duration.

* Internal Accession Date Only
Approved for External Publication
© Copyright Hewlett-Packard Company 2003

Idle Time Power Management for Personal Wireless Devices

Nevine AbouGhazaleh, Robert N. Mayo, Parthasarathy Ranganathan

nevine@cs.pitt.edu, (Bob.Mayo,Partha.Ranganathan)@hp.com

Technical Report

Abstract

Energy Consumption is becoming a critical concern that directly affect the mobility and lifetime of mobile and embedded systems. Different embedded and mobile applications place different demands on the wireless communication system and thus the energy consumption of the wireless cards. Past work has studied managing the energy consumed in wireless cards; we sought to evaluate some of these techniques by implementing them on our system and measuring the actual energy consumption by the wireless card.

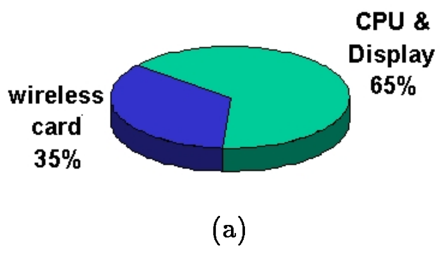
We tested two schemes. The first increases the card's sleeping duration allowing power to be reduced by putting the card in low power states for longer times. The second technique is more aggressive; the power manager completely shuts-down the card during idle periods. When applying these two techniques for an email application, we get 25% energy savings for extending the sleep time up to 60 sec, while the savings reaches 89% when shutting down the card for the same duration.

1 Introduction

Mobile systems are gaining in popularity, and in these systems energy use is critical. While the systems provide a great deal of utility, this is partially offset by the need to carry the device and to recharge it periodically. Thus, major metrics for evaluating a mobile system include the size of the device, its weight, and the length of time it runs between battery charges. All these factors can be improved by any power management technique that reduces the energy needed by the system.

Several power management techniques have been proposed for different components in mobile systems, such as the processing and storage units. With the advancement in wireless technology, wireless communication is becoming increasingly essential for mobile devices. On the other hand, they contribute to a relatively large portion of the energy consumption in mobile devices. In measurements of the iPaq running web browsing application [7], the wireless card consumed around 35% of the total system's energy consumption as shown in Figure 1. Other studies [5] have shown up to half of the system power is consumed by the wireless card. Thus reducing the energy consumption of this unit can yield a large reduction in the overall energy consumption of the system.

Different mobile applications place different demands on the wireless communication system and thus the energy consumption of the wireless cards. This difference can be exploited to save energy in the system. As an example demand, media applications have time constraints to be met (such as the display rate for MPEG frames transferred using a wireless link), thus a relatively large activity takes place in the wireless system to produce the required timely response. Applying aggressive energy saving schemes during the execution of these applications may lead to undesired degradation of service. On the other hand, for less resource demanding applications where the wireless card is idle for relatively large periods (such as email and web browsing applications); power management techniques can be applied without much sacrifice in the user response time. Figure 1-b shows the activity (as a percentage of time) of a wireless card connected to an iPaq running a web browsing application. The card is actively communicating with other nodes about 11% of the time, while the majority of the card time is spent idle with no user data transfer. Thus, based on the executed application requirements, a proper power management technique can save energy by exploiting these idle periods.



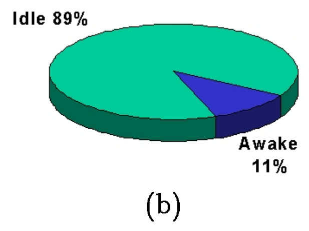


Figure 1: Time and power breakdown for the iPaq.

Past work has studied managing the energy consumed in wireless cards; we sought to evaluate some of these techniques by implementing them on our system and measuring the actual energy consumption by the wireless card. We also evaluated these techniques when applied to an email system as a sample popular application running on mobile systems. We are primarily concerned with techniques that reduce the power of the wireless card during its idle time so that the user response-time is least affected (if at all). We tested two schemes, the first increases the card's sleeping duration allowing power to be reduced by putting the card in low power states for longer times. The second technique is more aggressive; the

Table 1: The different power states of the wireless network card.

State	Description
Active	Card busy sending or receiving data
Standby/waiting	No data transmission or reception but the radio is powered up
Sleep	Radio is powered off, other circuits on
Off	All circuits are powered off
Idle	periodic alternation between Standby and Sleep state, no user data is exchanged

power manager completely shuts-down the card during idle periods.

In the next section, we present some background about the wireless protocols and cards. Section 3 describes the experimental setup uses to evaluate the techniques, followed by the evaluation of two techniques for reducing the wireless power during idle periods (Section 4). We also show the potential problems resulting from applying such optimizations and propose adequate solutions. In Section 5, we present experimental results demonstrating the potential energy savings from optimizations while running typical mobile applications like email. Section 6 briefly describes some of the related work in this domain, followed by our conclusions in Section 7.

2 Background

2.1 Wireless network interface cards

A wireless network interface card, *NIC*, has four main states of operation and each of those states has different power requirement. The states – in a descending order of their power consumption – are: active, standby (or waiting), sleep, and off states. (Table 1). The most power demanding state is the active state, where the card is busy sending and receiving data. The majority of the power consumed is for powering the radio subsystem. The card is said to be waiting (or in standby state) when there is no data transmission or reception activity taking place, however the card is watching the medium and the radio's receiver is turned on. If the radio is turned off then the card enters a sleep state, however no components are powered-on such as clocks and the microcontroller [13]. The NIC enters the off state when power to the entire card is removed. Transition from one state to the other consumes time and power overheads. The lower the power state of the card, the more energy is needed for transition from this state to a higher power level. The card is said to be *idle* when there is no user data exchange taking place, however the card periodically alternates between the sleep and standby states.

2.2 Power management in IEEE 802.11b

The IEEE-802.11b standard defines protocols for two sets of networks; the infrastructure and the ad-hoc network topologies. An infrastructure wireless network has a set of wired stations called access points, *APs*. Each access point is responsible for buffering messages to the mobile stations found within its range. APs also act as a gateway to those stations. On the other hand, in an ad-hoc network each station is responsible for maintaining its own traffic without the buffering support of a base station as in case of the infrastructure network [6]. For the remaining of this paper, the terms mobile station and station are used interchangeably.

It is easier to apply power management schemes in an infrastructure network due to the presence of access points that are always powered up. As shown in Figure 2, each AP periodically sends a traffic indication map *TIM*, frame (also known as *beacon*) to all stations within its range at equal time intervals. These intervals are called *listen intervals*. The TIM frame indicates which of the stations has stored messages in the AP buffers. Each station wakes up (by turning on its radio and entering the standby state) to listen to the TIM. If there is any data stored for this station, it starts reception from the AP. Otherwise it returns back to its sleep state. The station has the flexibility to sleep for a multiple of beacon intervals on the expense of increasing the data delays.

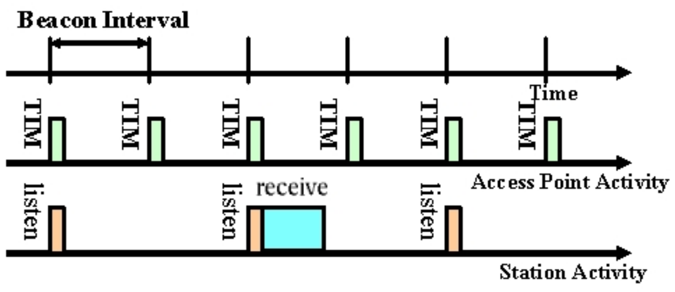


Figure 2: Power management protocol in the 802.11b infrastructure network.

Many of the wireless cards that are compatible with the 802.11b standard operate in two modes. The first mode is a continuous mode, where the radio subsystem is always powered up and the card is in a continuous wait (standby) and ready to receive data from the AP. The other mode is a power saving mode (often called *PSM*), where the card is allowed to sleep during the listen interval periods. In the next section we present our experimental setup for evaluating power management techniques.

3 Experimental setup

To evaluate the efficiency of our power management schemes, we use a data acquisition system, DAQ, to measure the total power drained by the wireless card during different activities.

Our experimental network consists of a mobile station and another machine (not necessarily mobile). The mobile station is a Compaq Aramada laptop running Linux 2.4.17 that communicates with a Compaq desktop running Windows NT 2000. All communications take place through an access point. A Cisco Aironet 340 wireless card is attached to the laptop through a PCMCIA port, while the desktop is connected to the network through an Ethernet card.

To measure the power of a card, we use a PCMCIA extender to expose the supply voltage pins of the card. A resistance of 1 ohm is connected in series with the card. We use a National Instruments 6034E DAQ card to measure the voltage across this resistor. The DAQ samples the voltage drop on the resistance with a rate of 10,000 samples (readings) per sec. From Ohm's law ($V = IR$), we compute the current drawn by the wireless card, I. By knowing the supply voltage of the wireless card, V_{dd} (typically 5V or 3.5V), the power, P, is computed according to the relation $P = V_{dd}I$. Figure 3 illustrates the DAQ setup, where GND is the ground voltage, and V_{DAQ} is the voltage drop measured by the DAQ across the resistance R.

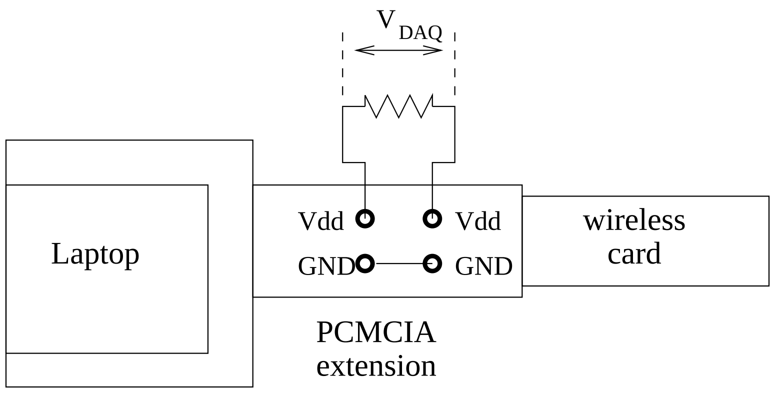


Figure 3: The DAQ setup for measuring the power consumption in wireless cards.

Idle power management involves adapting some parameters (e.g. listen interval) and modes in the software device driver that controls the wireless card. Wireless card configurations can be changed on the fly (without restarting the driver) using the wireless extension API and tools [16]. **iwconfig** is one of the tools that manipulate wireless card common parameters such as the transmission rate and power. The command:

```
iwconfig <interface name> power period <listen interval>
```

is used to change the listen interval value. On the other hand, iwconfig can change the card mode by only switching the default power management mode on and off. To add the ability of switching to different modes, we extend the wireless extension API to set the new modes in the device driver. We also disable the reception of broadcast messages in the mobile station to isolate the effect of power management due to the station messaging.

4 Idle Time Power Management Techniques

This paper targets the idle-time (the inactivity time when there is no user data transmission or reception) power consumption since it highly contributes to the total energy consumed by the NIC as discussed in Section 2. During idle time, the network card listens to the media by periodically turning on and off the radio at every beacon (typically every 100 msec). Although the sleep state power consumption is relatively low compared to the standby state, the idle power dominates the total energy consumption. Next we discuss two optimizations that can be applied to the device driver to minimize the idle power. Further power savings can be achieved by extending the sleep time, and reducing the sleep power of the network card.

4.1 Extending the sleep time

One of the network card parameters directly affecting the power and performance of the network is the listen interval. Increasing the listen interval would potentially decrease the average power consumption due to the reduction in the frequency of activating the radio to listen to the beacon message, at the expense of increasing the buffering time of incoming messages in the AP. Figure 4 shows two power consumption profiles for the network card: one with a 100 msec listen interval, and another with a 5 sec interval. In our experiments we varied the listen interval from 100msec to 65sec (the maximum value permitted in the Aironet card). Figure 5 shows the average power consumption of the card normalized to the case of the listen interval set to its default value; 100 msec. A listen interval of 50 sec achieves 25% power savings over the default power management in the 802.11b protocol. We noticed that with the increase in the listen interval, the power saving rate decreases. This is due to that fact that at larger intervals there are fewer radio wake-ups per unit time that can be spared to save power.

The support for varying the listen interval and its ranges are card-specific parameters. From our measurements, we found that a Cisco Aironet 340 series card [14] supports up to 65.5 sec sleeping time. While an Orinoco card [15] (newer version of Lucent Wavelan cards) supports only up to 2 sec. Other older manufactured cards do not have the flexibility of setting the listen interval. Next we discuss the potential effects of varying the listen interval - as part of the data link layer - on higher level protocol layers.

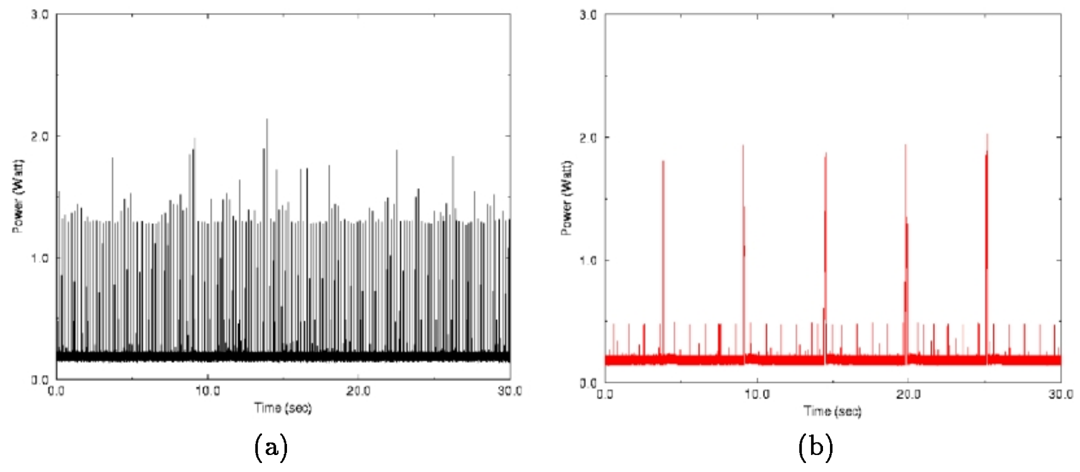


Figure 4: The consumed power during idle times with (a) 100 msec listen interval and (b) 5 sec listen interval.

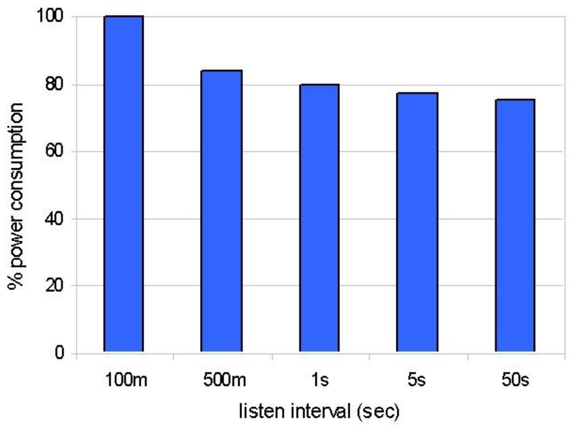


Figure 5: Reduction in the average idle power with varying the listen interval.

4.1.1 Effect of listen interval on higher level layers

Increasing the listen interval in a mobile station induces more delay in the network, thus increasing the roundtrip time. Higher level protocols like TCP base their decisions mainly on the monitored network delays. For example, the congestion control decides to drop a packet if acknowledgments were not delivered within certain time interval. Due to our optimization, two primary problems arise from increasing the listen interval period: (1) increasing the TCP start-up time and (2) authentication and handshaking conflicts. Next we briefly describe these problems and their proposed solutions.

Long TCP start-up time

Establishing a TCP connection is an incremental process; i.e. the sender starts by sending a packet and then waits for its acknowledgment before TCP increases its window size by one. TCP keeps on incrementing the window size until the connection reaches its steady state. The startup time is critically important since TCP is characterized by its short-lived connections. If the startup time is relatively long that it might dominate the overall connection time then it will result in a performance and response time degradation. Assuming an always-on connection (the NIC is in a standby state), this start-up time is a function of the network round trip time, *(RTT)*.

Applying an extended listen interval is equivalent to disabling the connection during the sleeping times of the NIC. This extra sleeping time introduces more delays in establishing the connection as the station will not receive messages or send acknowledgments except at the end of a listen interval. In other words, RTT is bounded by the listen interval of the mobile station.

To solve this problem we suggest a hybrid scheme that combines the power saving and the continuous modes. The NIC is placed into the power saving mode; however, if the station has data in its buffer ready to be sent, the NIC is switched to the continuous mode. It stays in the continuous mode until the card finishes sending all its data. The card then switches back to the power saving mode. By applying this scheme we overcome including the sleeping periods in the RTT and thus reducing the startup time.

Authentication and handshaking conflicts

In establishing a connection between the mobile station and a remote server, an authentication and handshaking protocol takes place. For example in an FTP connection, a client requests for establishing a connection to the server. The server asks for an authentication then the client replies with the required information. In between this sequence of the requests and replies, the mobile station goes to sleep according to the power saving mode. Same as in the TCP start-up, if the sleeping time increases the connection establishment time increases. Furthermore, if the listen interval exceeds the timeout of receiving the authentication, the client may assume the server is not available and the connection is refused.

As a remedy for this problem, we speculate that if some event happens, there is a high probability that it will happen again soon. Mapping this to the station's network traffic; at each packet transmission, there is a probability of a follow-up network traffic. We assume that this probability decreases as time advances. For example, each request is expected to be followed by a reply or acknowledgment. On the other hand, as time progresses with no network activity, we expect no activity in the near future as well and act accordingly. Based on this assumption we gradually increase the sleep time as long as there is no traffic to/from the station. After each packet transmission, the listen interval is set at its default value; 100 msec, then it is doubled– if no activity occurs– till it reaches its maximum value (64 sec in case of the Aironet 340 card). Through prediction of arriving packets, we dynamically increase the listen interval and thus eliminate possible conflicts in the handshaking and authentication process. This mode is called a Fast Power Savings Mode, or *FPSM* for short. Figure 6 shows the gradual increase in the listen interval (e.g., period from 9.5 sec to 23 sec) and its reset to the default value at any data transmission and reception (e.g.,at time 26.5 sec).

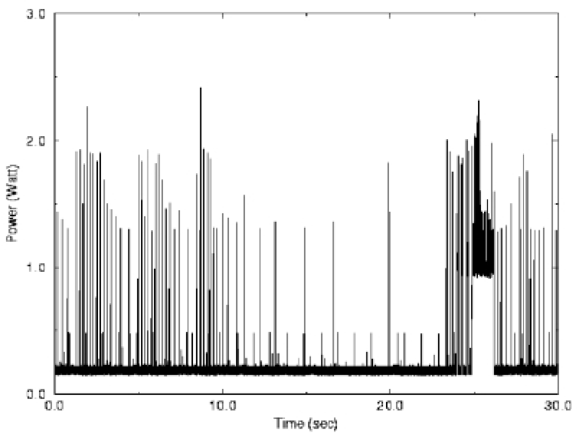


Figure 6: The gradual increase in the listen interval after each transmission or reception of data.

FPSM compares favorably with PSM, due to its ability to gradually adapt to the changing network loads. As a result of the graceful increase in listen interval values, the average delay that a message experiences decreases compared to the delay experienced when using PSM. On the other hand, the average power consumed is expected to be slightly higher than PSM due to the reduction in the total sleep time of the card operating on FPSM (average length of listen intervals is smaller in FPSM than PSM). Applying FPSM or PSM is adequate for application with frequent network usage, however if the length of the idle time between incoming/outgoing date can be predicted to be relatively large, then we can select a more aggressive technique for power management such as the one discussed in the next section.

4.2 Reducing the sleep power

Since increasing the sleeping time of the network card reduces the energy consumption by only 25%, we investigated further improvement in the remaining 75% of the energy consumed. The relatively high power consumed in the sleep state motivates the need for a more aggressive optimization to reduce the total energy consumed, such as switching the card to a lower power state during inactive periods. Since the software control through the device driver does not completely turn the card off (it only turns the radio subsystem off, while the microcontroller is left on), we suspend the PCMCIA card to put the NIC to an off state.

Figure 7 shows the power consumed in the processes of turning the card off then on again after 8.1 sec. Indeed there is no power drained during the off time, however switching the card back on consumes a considerable amount of power and time overhead for the transition. During this transition, the card negotiates with the access point and exchanges setup information. This exchange of information along with the warm-up of electronics [13] in the card, are the main consumer of the time and power overheads. The Aironet card needs about 1.9 sec and 1.85 J to fully reach the standby state.

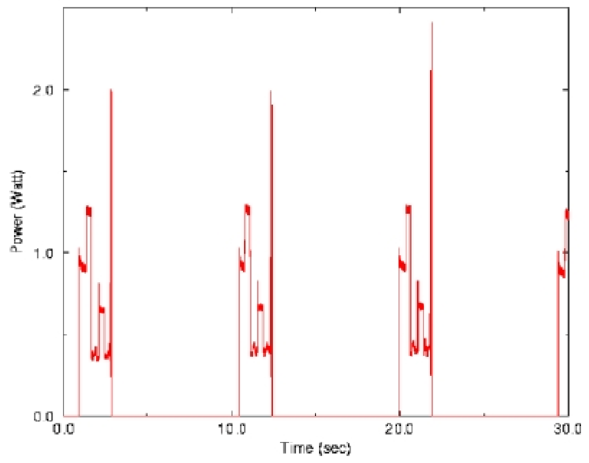


Figure 7: The power consumption with a shut-down during sleeping times.

The transition to (from) the off state– complete shut down (turning on) of the network interface– is done through suspending (resuming) the PCMCIA card in the laptop using "suspend" ("resume") system calls defined in the Linux kernel.

Scheduling the state transitions should be guided by a break-even point between the current and target states. This point defines when it is beneficial– from the energy prospective– to switch to a lower power state (taking into account the overhead cost of such transition) rather then continue operation in the current state. Next we discuss how to find this break-even point.

4.3 Power states break-even point

Selecting one of the two presented optimizations to apply is highly dependent on the inactivity time and the overhead of transition. For small inactivity periods, the first optimization (setting the listen interval) shows it effectiveness over completely shutting the card off. This is due to the high overheads associated with activating the card, which overshadows the power saving from residing in the off state for short periods. On the other hand, when the inactivity periods increase, the card set in the off state uses less power than when staying in the sleep state.

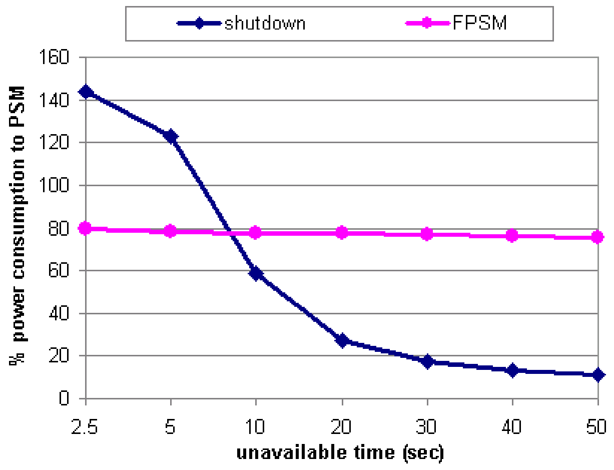


Figure 8: the average idle-power consumption while varying the sleep time.

Figure 8 compares power the FPSM and the shut-down techniques based on the average power consumed versus the unavailable time spent in each scheme. In both techniques, the card is periodically switched to the active state to poll the AP then switched back to its power saving state. The unavailable time is defined by the listen interval or the shutdown periods. FPSM increases its listen interval starting from 100 msec and doubling after every other beacon till it reaches its maximum value. We compare this to the shutdown scheme, including the overhead time to restart the card in the unavailable time of the card. For example, if the card is shutdown for 5 sec and resumes to its active state in 1.9 sec, then the unavailable time is considered to be 6.9 sec.

Computing a break-even point gives an indication on when to use each of the presented optimizations. A transition into a lower state is beneficial if the energy consumed in a lower state including the transition overhead is less than the energy consumed when operating in the current state. That is, it is beneficial to switch from the sleep state to the off state if the card consumes less power in the off state (including the transition cost) than being in a sleep state i.e., $E_{off} + E_{trans} <= E_{sleep}$, where $E_{off}, and E_{sleep}$ are

Table 2: The characteristics of the message trace collected for a week.

Type	average size	median size	avg. # of messages/day
Incoming	10.2k	3	24.6
Outgoing	233.7	2.5	7

the energy consumed in an off and sleep states respectively, and E_{trans} is the energy cost for switching a card in active state to and from an off state. We can deduce that $t_{off}\ P_{off} + E_{trans} <= P_{sleep}\ t_{sleep}$, where $P_{off}, and P_{sleep}$ are the average power consumed in an off and sleep states respectively, and t_{off} and t_{sleep} are the times spent in off and sleep states respectively. For the Aironet card, $E_{trans} = 1.85J$ (transition energy from active to off is of a negligible cost and from off to active consumes 1.85J), $P_{sleep} = 200mwatt$, and $P_{off} = 0$. Substituting in the formula and solving for t_{sleep}, we get a break-even point around 9.25 sec, which is the point of crossing curves in Figure 8. Thus if a running application can tolerate network communication with unavailable times that exceeds 9.25 sec, then it is beneficial to use the shutdown policy to save energy, however, if the application is delay sensitive to its network traffic, then less aggressive optimizations should be followed such as PSM or FPSM. Next we evaluate these two techniques and measure their energy consumption when running an email application while varying its response time requirements.

5 Email Application

Applying the aforementioned optimizations is highly based on the type of application running on the mobile station and its response-time sensitivity. We experimented with an email application as an example of a popular application running in mobile devices. The application's non-demanding response time requirements enable us to see the extent of power savings that can be achieved using our techniques. Applying these techniques for other applications with different network requirements will lead to different power savings.

We collected an incoming/outgoing message trace from a typical user's mailbox for the duration of a week. The message sizes range from few Kbytes to hundreds of Kbytes depending on its contents. The message size includes the size of message header, message body and any attachment if found. Table 2 shows the characteristics of the collected trace. To run the trace on the mobile station, we use a script that calls *sendmail* and *fetchmail* applications to send and receive messages respectively. In addition, *fetchmail* is called in the script once after each inactive period to check for incoming messages. To emulate the mailer idle times, we delay the processor between receiving and sending events for a period equal to the corresponding trace idle times.

To handle the huge amount of data collected from the DAQ for a week-long trace, we reduce the DAQ's sampling frequency to 1000 samples per second, and divide each day's trace into a set of smaller duration traces. We collect and process the data after collecting each of the smaller traces. We also fast forward the trace idle periods in units of one minute, i.e., skip all whole-minute periods (one minute increments) in the trace that do not include any message exchange – but may contain beacons. To compensate for the skipped idle minutes, we average (1) the power measured from the modified trace and (2) the average power consumed in a single idle minute multiplied by the number of fast forwarded idle minutes in the trace. The average power consumed in an idle minute is computed by measuring the card power for several minutes of idle time. The mean is computed after eliminating the highest and lowest power values measured in a minute interval. We do the same procedure for each experiment configuration.

Using this setup, we compare three power management schemes; PSM, FPSM, and shutdown, as shown in Figure 9. FPSM and shutdown are measured for multiple time constants. The power consumption for each scheme is normalized to the PSM at 100 msec. As we can see, the shutdown method is superior to PSM and FPSM in terms of power savings. This comes at the cost of making the wireless system unavailable for the period it is shut off. However, since this particular application can tolerate this behavior, the energy savings are superior.

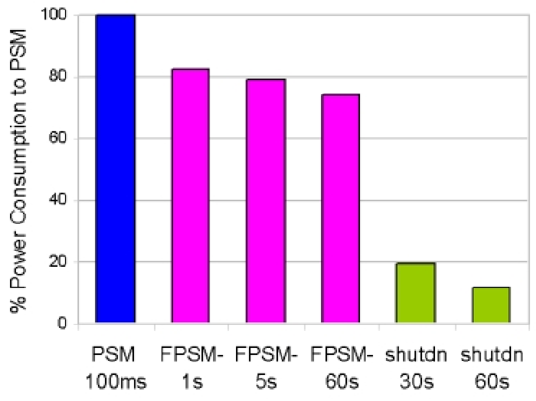


Figure 9: Comparison between different power management schemes with varying the unavailable time.

6 Related work

Previous work has addressed some of the problems of minimizing the power consumed during the idle time in infrastructure and ad-hoc networks. A bounded slow-down protocol [3] has been proposed for infrastructure networks, dynamically adjusting the idle times (equivalent to the listen interval) for increasing the energy savings. Idle time adjustments are based on a given slowdown factor of the round-trip time. The authors didn't consider turning the card off during extended idle times. Simunic et al. [8] presented a approach based on semi-Markov decision processes for minimizing the energy consumption. Their optimization policy selects the transitions between the different power states of the network card, using events in the OS. They didn't explore varying the listen interval. Jung et al. [2] proposed a dynamic variation of the idle time in ad-hoc networks based on monitored network parameters. They presented rules for increasing and decreasing the idle times. Havinga et al. [1] identified some problems of wireless multimedia networking and presented several solutions that focus on energy efficiency. They also proposed the employment of a quality of service mechanism for achieving energy efficient systems. A quantitative comparison (through experimental results) among different wireless cards was presented by Synack [17], evaluating the performance and the energy capabilities of each of tested cards. A technique much different than the above studies is to add a second low-power radio channel [9] for notifications of incoming data, thereby allowing the shut down of the rest of the system to reduce the idle power.

7 Summary and Future work

This work presents an evaluation of power management schemes that aim to reduce the power consumption of wireless cards during their idle times. We compared two optimizations and explored some of the potential problems arising from applying such schemes. The first optimization increases the duration of a card being in the sleep state by reducing the card's frequency of polling the access point. While this optimization reduces the power consumed in periodically turning on the radio to receive control data from an access point, the energy savings is constrained by the sleep power especially for long idle sleep periods. The second optimization can be applied to further reduce the idle-time power by turning off the card during inactivity periods. This is beneficial when expecting long inactivity periods such that the power savings can overcome the energy overhead of switching the card on and off. As the transitioning time from a low power state to an active state increases with the decrease in the power level (for example transition from sleep to active consumes less time than from off to active), the selection of a card state is dependent on the application time-response requirement. Applying these two optimizations for an email application, we get 25% energy savings for extending the sleep time up to 60 sec, while the savings reached 89% when shutting off the card for the same duration. However, for relatively short inactivity periods (up to 9 sec for the Aironet card), extending the sleep time would consume less power than shutting off the card.

Future work includes designing an application programming interface (API) that allows applications to specify their network requirements. Based upon this data, an appropriate selection of card state transitions may be selected that optimizes power while still meeting the requirements. For example, telephony applications would require low latency during connected calls and 1-2 second latency when waiting for a call. Given that other applications running at the same time may have different requirements, we need to design a way to integrate the requirements of all the running network applications to achieve an acceptable response time for the entire system. For example polling every minute works well for email, but not for phone calls or emergency pages. There is also the need to develop more power conscious wireless cards that consume less energy especially in the non active states.

This work is part of the larger e-Scale project. e-Scale seeks to provide adaptability, so the power consumed by the hardware matches the requirements of the application [10]. In practice, current hardware often includes features only required by the most demanding applications, and power is consumed by these features even when they aren't needed. Various forms of adaptability can do a better job, matching the power consumption to the features actually needed by the application. For processors, we've developed multi-core processor designs that select the most power-efficient core for the workload presented by the application [11]. This results in a 40% reduction in energy at the cost of a 3% speed reduction due to adaptability overhead. Since display subsystems also consume large amounts of power, we've investigated OLED displays in the energy scale-down context [12]. Our approach is to adapt, in color and intensity, sections of the displayed image to use less power while still meeting the information presentation needs of the particular application. This results in close to a 40% reduction in energy use, at the cost of a slightly changed user interface. In general, adapting the hardware to match the application's requirements is a promising energy scale-down technique.

References

[1] Havinga, P.J.M., Smit, G.J.M., Minimizing energy consumption for handheld computers in Moby Dick. Proceedings IEEE International Conference on Personal Wireless Communication ICPWC'97, ISBN 0-7803-4298-4, pp. 306-311, December 1997.

[2] Eun-Sun Jung and Nitin H. Vaidya An Energy Efficient MAC Protocol for Wireless LANs. IEEE INFOCOM, 2002.

[3] Ronny Krashinsky, Hari Balakrishnan Minimizing Energy for Wireless Web Access with Bounded Slowdown. Proc. ACM MobiCom'02, pp.119–130, 2002.

[4] R. Kravets and P. Krishnan Application-Driven Power Management for Mobile Communication. Wireless Networks 6, pp. 263–277, 2000.

[5] M. Stemm and R. Katz, Measuring and Reducing Energy Consumption of Network Interfaces in Hand-Held Devices, Proceedings of the 3rd International Workshop on Mobile Multimedia Communications (MoMuC-3), September, 1996.

[6] Bob O'Hara, Al Petrick, "The IEEE 802.11 Handbook: A Designer's Companion". IEEE, 1999. ISBN 0-7381-1855-9.

[7] Kathy Richardson, Understanding iPaq 802.11b Power Measurements, Internal Memo, Compaq Computer, 2002.

[8] Tajana Simunic, Luca Benini, Peter Glynn, Giovanni De Micheli, Dynamic power management for portable systems. Proc. ACM MobiCom, 2000.

[9] Eugene Shih, Raramvir Bahl, Michael Sinclair, Wake on Wireless: An Event Driven Energy Saving Strategy for Battery Operated Devices. Proc. ACM MobiCom, 2002.

[10] Robert Mayo and Parthasarathy Ranganathan, Energy consumption in mobile devices: Why Future Systems Need Requirements-Aware Energy Scale-Down, Technical Report HPL-2003-167, Hewlett Packard Laboratories, 2003.

[11] Rakesh Kumar, Keith Farkas, Norm P. Jouppi, Parthasarathy Ranganathan, and Dean M. Tullsen, A Multi-Core Approach to Addressing the Energy-Complexity Problem in Microprocessors, Workshop on Complexity-Effective Design, 2003.

[12] Subu Iyer, Annie Luo, Robert N. Mayo and Parthasarathy Ranganathan, Energy-Adaptive Display System Design for Future Mobile Environments, Proceedings of the First International Conference on Mobile Systems, Applications, and Services, May 2003.

[13] Prism II power management modes. http://www.intersil.com/data/an/AN9907.pdf, August, 2002.

[14] Cisco Aironet 340 user manual.
http://www.cisco.com/warp/public/cc/pd/witc/ao340ap/index.shtml, August, 2002.

[15] The Orinoco wireless card. http://www.orinocowireless.com, August, 2002.

[16] The wireless extension Tools.
http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Tools.html, August, 2002.

[17] Power Consumption of Wireless Networking PC Cards.
www.synack.net/wireless/consumption.html, August, 2002.