An Embedded Platform with Duty-Cycled Radio and Processing Subsystems for Wireless Sensor Networks

Zhong-Yi Jin[1], Curt Schurgers[2], and Rajesh Gupta[3]

[1] UCSD Dept. of Computer Science & Eng `zhjin@cs.ucsd.edu`
[2] UCSD Dept. of Electrical & Computer Eng `curts@ece.ucsd.edu`
[3] UCSD Dept. of Computer Science & Eng `rgupta@cs.ucsd.edu`

Abstract. Wireless sensor nodes are increasingly being tasked with computation and communication intensive functions while still subject to constraints related to energy availability. On these embedded platforms, once all low power design techniques have been explored, duty-cycling the various subsystems remains the primary option to meet the energy and power constraints. This requires the ability to provide spurts of high MIPS and high bandwidth connections. However, due to the large overheads associated with duty-cycling the computation and communication subsystems, existing high performance sensor platforms are not efficient in supporting such an option. In this paper, we present the design and optimizations taken in a wireless gateway node (WGN) that bridges data from wireless sensor networks to Wi-Fi networks in an on-demand basis. We discuss our strategies to reduce duty-cycling related costs by partitioning the system and by reducing the amount of time required to activate or deactivate the high-powered components. We compare the design choices and performance parameters with those made in the Intel *Stargate* platform to show the effectiveness of duty-cycling on our platform. We have built a working prototype, and the experimental results with two different power management schemes show significant reductions in latency and average power consumption compared to the *Stargate*.

1 Introduction

A wireless sensor network (WSN) consists of a collection of wireless sensor nodes which are small embedded devices with on-board sensors and wireless radios. Without wires, sensor nodes either rely on limited energy supply from batteries or harvested energy from intermittent sources like solar or wind. To ensure long lifetimes demanded by the application and deployment scenarios, the sensor nodes have to be very energy efficient. Popular sensor nodes such as the Berkeley Mote [1] address this issue using low power hardware as well as aggressive power management techniques. However, their design choices also make these nodes useful only to applications requiring limited processing power, short communication range and low network bandwidth.

A common WSN architectural solution to these constraints is to deploy within the sensor network a small number of high performance nodes equipped with high powered components like fast processors or high bandwidth radios. As high performance sensor nodes are usually placed in the same environment as regular sensor nodes, they also rely on limited energy sources. Therefore, energy-efficiency is critical for the high performance nodes to last as long as the rest of the sensor nodes.

There are two observations that we can explore to improve the energy-efficiency of high performance nodes. Firstly, as a general fact, using components with high peak power consumption doesn't necessarily imply high energy consumption. Studies have shown that when a sufficient amount of data need to be processed or transmitted, high performance processors or radios that consume more power than their sensor node counterparts may complete the same amount of work faster and therefore end up using less energy [2, 3]. Secondly, in the specific case of sensor networks, those high powered components are not required to be active all the time as sensor networks usually do not generate large amounts of data until certain triggering events are detected. In other words, the node or its components needs to be active only a fraction of the time to achieve application goals. Therefore, duty-cycling based power management techniques such as selectively enabling or disabling components are important in reducing energy consumption for high performance nodes.

A platform needs to meet two requirements to support efficient duty-cycling. One is that it needs to consume very little (or no) power when there are no ongoing activities. While general purpose high performance nodes such as the *Stargate* [4] provide a good balance of performance and power consumption, they are not designed to support efficient power management via duty-cycling. For example, the lowest power consumption of a *Stargate* is 16.2mW in its inactive suspended state [5]. In contrast, a typical sensor node such as the Telos Mote uses only about 10μW in a similar state [6]. This high standby power consumption significantly limits the effectiveness of duty cycling, making the *Stargate* less energy efficient for very low duty cycle sensor network applications such as environmental monitoring. The other requirement is that a platform needs to be able to activate or deactivate various subsystems with very little overheads according to runtime demands. Existing high performance nodes built around 32-bit embedded processors and embedded versions of traditional desktop Operating Systems (OS), both old [4] and new [7, 3], generally take a long time to transit in and out of the suspend or power off states, bringing significant energy and performance overheads to duty-cycling [5, 7, 3].

In this paper, we describe the design, implementation and evaluation of a wireless gateway node (WGN) that enables efficient power management through duty-cycling. Gateway nodes are often intrinsic parts of sensor networks and are required to bridge data between sensor networks and servers/devices in other networks. Without gateway nodes, data collected by a sensor network are only local to that sensor network and therefore no remote monitoring or interactions can be achieved. The low and bursty traffic load of sensor networks makes the WGN an ideal application of our low power design.

Specifically, we focus on using Wi-Fi (802.11b) radios to interface sensor nodes with devices in other networks because Wi-Fi radios offer higher bandwidth and consume less energy per bit than most existing sensor node radios [8], making them a useful air interface for many sensor network applications. Although our focus is on the gateway nodes, the basic design and techniques can also be applied to any other high performance nodes or used in the context of wakeup radios [9] or any other multiple radio hierarchies [10] to reduce switching overheads with respect to energy and latencies.

2 Related Work

To energy efficient design, the importance of separating real-time monitoring functions that have to be optimized for low power from functions invoked with light duty-cycles is first unveiled in the development of the WINS nodes [11]. The WINS node enables continuous sensing over an extend period of time by partitioning the system into a low powered event-detection subsystem and a high powered processing subsystem which can be enabled or disabled on demand.

Our work is directly comparable to the emerging Micro-servers that are being explored to solve a large body of sensor network research problems [12, 7]. Triage [7] extends the lifetimes of their Micro-servers by trading latency for power reduction. Its tiered architecture consists of a slightly modified *Stargate* computer and a MicaZ mote, which is used to power on the *Stargate* only when sufficient amount of data are being batched for processing. Due to the large latency in powering on/off the *Stargate*, their platform is not usable for our gateway application in terms of delay and power consumption. The LEAP platform [3] faces similar issues as stated in their future work section. The PASTA platform [13] also uses an Intel PXA255 processor. Since their demonstrated mode of operation is to keep the processor module in sleep state (7.3mW) during periods of inactivity to save power, no experiments and latency numbers are reported for activating this module from power-off state.

3 Design Approach

Fig. 1 shows a high-level block diagram of our design. To minimize standby energy consumption, we exploit the low power operation of a sensor node processor and use it for subsystem scheduling and power management. A second, more powerful application processor is used to provide on demand processing capabilities to subcomponents such as the Wi-Fi radio, which are physically connected to the application processor. Power to each individual subcomponent is either controlled directly by the sensor node processor or indirectly by the application processor through the sensor node processor. We use a serial interface for inter-processor communication because it is supported in various forms by most of the existing sensor node processors.

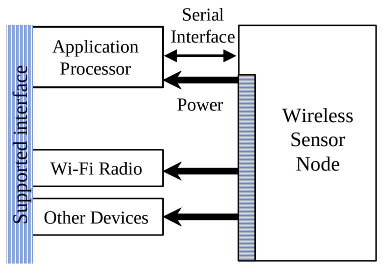

Fig. 1. WGN Block Diagram

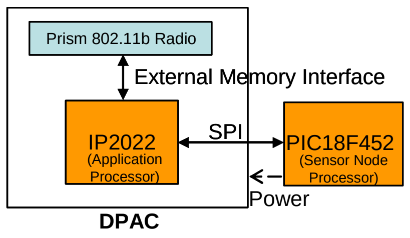
Prism 802.11b Radio
External Memory Interface
IP2022 (Application Processor)
SPI
PIC18F452 (Sensor Node Processor)
Power
DPAC

Fig. 2. WGN Architecture

Unlike the PASTA platform [13] where multiple microcontrollers are used to regulate power to the modules, we have the sensor node processor acting as the “master” device for power management. Our approach simplifies the design while also taking advantage of the fact that the sensor node processor is almost always on for sensing or networking purposes. This enhances the efficiency of running the power management itself. Besides supporting the low-power sleep mode as described in the PASTA paper, we also want to be able put the application processor in and out of power-off mode without introducing significant energy and latency overheads.

In contrast to the low-power sleep mode where a certain amount of power is still consumed by the application processor running in power saving modes, a processor uses no power at all while in power-off mode. However, it generally takes less time to resume a program from low-power mode when the program is either suspended or running at a slower speed than to reload the entire program by powering the application processor back on from power-off mode. While it is clear that having minimal standby power consumption would extend the operation time, some real time applications also have time constraints, as they need to complete certain tasks within a fixed amount of time. Instead of making the power and latency trade off at design time, the sensor node processor needs to be able to put the application processor into either low-power mode or power-off mode at runtime based on application requirements.

To reduce the energy and latency overheads in activating the application processor from power-off mode, we found it is critical to minimize both hardware and software startup time. On the hardware side, commercial microcontrollers for embedded devices are usually designed to support fast startup. On the software side, it is important to minimize the amount of the code that needs to be loaded and executed every time during boot up. This include both OS and application specific code.

For our specific WGN application, we use the power saving mode of the 802.11b protocol [14] to lower the latencies in switching the Wi-Fi radio in and out of low-power mode. We develop techniques to take advantage of certain features of the power saving mode of the 802.11b protocol [14] to further reduce communication and synchronization overheads in terms of energy and latency.

4 Platform Implementation

The hardware architecture of our WGN is illustrated in Fig. 2. As explained in the introduction, our main contributions are on the level of the architectural design approach (see also section 3). To illustrate these ideas and perform experiments, we have to make specific design choices for our test bed platform. Our approach, however, is not restricted to these specific choices alone.

We use the Ubicom IP2022 processor [15] as our application processor. With a maximum clock speed of 120MHz (approximately 120MIPS), this 8-bit processor is less powerful than the 400MHz (480MIPS, Dhrystone 2.1) PXA255 on the *Stargate*. However, it is significantly faster than the microcontrollers on most existing sensor nodes and provides sufficient processing capability to our gateway application. With its integrated flash memory and RAM, this processor doesn’t need complex external hardware support and thus doesn’t incur extra energy and latency overheads. We also find the IP2022 a convenient choice as it is used in the DPAC module [16] with a Wi-Fi radio.

Because of a clear separation of the master (sensor node processor) and the rest of the system on our platform, we can select any existing sensor nodes such as the Berkeley Mote [1] for our power management purposes. Our design only requires that the sensor node processor and the application processor can communicate and wake each other up as necessary. This is easier to implement than those that require external logics or chip-set supports as in PC platforms. Commonly used serial interfaces such as the UART, I2C or SPI are sufficient to meet these requirements. We select the SPI as it supports duplex data transfers at a rate of more than 11Mbps, sufficient to sustain the throughput of the Wi-Fi radios.

For evaluation purposes, we use for power management a home grown sensor node equipped with a PIC18F452 microcontroller that consumes about 8mW (3.3V) in full power mode (10MIPS). The hardware SPI on the PIC is still available. The power to the DPAC module is managed by the PIC through a MOSFET switch. Alternatively, we can use an I2C power switch to control additional components. We use Ubicom IPOS, a lightweight OS for the IP2022. The IPOS provides the very basic OS functions and gets compiled with the target application. Our entire gateway application is about 60 Kbytes in uncompressed form.

5 Power Management Schemes

We experiment with two different power management schemes for our gateway application to evaluate the platform as well as to understand the power versus latency tradeoffs. We refer to these two schemes broadly as *power-gating* and *power-saving* modes. In the former, the emphasis is on subsystem shutdown for both communication and processing, while the latter seeks to exploit various slowdown modes. In the following subsections, we describe the design choices behind the two schemes.

5.1 Power-Gating Scheme

Our *power-gating* scheme saves power by putting the system into *power-gating* mode according to online demands. While in the *power-gating* mode, the Wi-Fi radio and the application processor are powered off and no packets can be sent or received. A successful use of the *power-gating* scheme requires participating gateway nodes and devices in other networks to coordinate their active periods while minimizing the total amount of energy required for such synchronizations [17]. We measure the overheads of such protocols when used with our WGN. The overheads are quantified either as time in seconds or energy in Joules calculated by integrating power over time.

5.2 Power-Saving Scheme

Our *power-saving* scheme reduces power consumption by putting the radio in the 802.11b power saving mode [14] while keeping the application processor in various low power modes. Since most Wi-Fi radios natively support the 802.11b power saving mode, it is significantly faster to put the radio in and out of the power saving mode than to suspend and resume the entire radio in each duty cycle. Although the speedup is hardware dependent, it is reported in one case that it is almost 86 times faster to resume from power saving mode than from suspended mode [10].

One challenge in supporting such a scheme is to synchronize power saving states across the processors, the radio and the access point. In the 802.11b power saving mode, a radio wakes up periodically to check for any incoming packets buffered at the AP (Access Point) and the sleep duration is determined by the listen interval. A listen interval is established during the association process and is fixed until new association attempts are made. Since the re-association process involves multiple packet exchanges between the station and the AP, it is expensive to change the listen interval frequently. However, a fixed listen interval is not ideal for most sensor network applications as events occur randomly. Long listen intervals introduce large communication delays while short listen intervals waste energy if there are no events or data to send or receive. Thus, the choice of the listen interval is a matter of design tradeoff between energy savings and latency incurred. Instead of listening at a fixed interval, we use an event-driven approach to transition in and out of the power saving mode as explained in its implementation below.

Our strategy is based on the 802.11b standard [14] that after a station exits power saving mode, the AP should send all buffered packets to that station as if they just arrived. Therefore, if the listen interval is set to an arbitrary large value (up to 2^{16} beacon intervals), one can eliminate its effects and dynamically control the packet receiving time by forcing the station out of the 802.11b power saving mode. Although a successful packet exchange is required between the station and the AP to enable or disable the 802.11b power saving mode, this can by done by simply changing the power management bit in the frame control field of any outgoing packets. A potential benefit of this strategy is that with some buffering, sending and receiving can be performed within the same active periods and therefore reduces the total amount of time the radio and the application processor need to be awake. We experiment with this approach on our WGN by sending and receiving 1024 bytes of data in an UDP packet every 6 seconds for a period of 30 minutes and observe no packet loss. We also verify this technique on a Linux laptop with a Netgear WG511 Wireless PC card (V1.0) and latest hostAP driver (V0.5.1) and observe similar results. Note that to avoid overrunning the buffer of the AP, a small packet should be sent to instruct the receiving device running our scheme to wakeup more frequently before transmitting a large amount of data.

5.3 Measurements

Power consumption is measured as the product of voltage and current. Our WGN is directly powered by a DC power supply of 3.3V. To measure current, a resistor of 1Ohm is connected in series with the DC power supply. The voltage drop across the resistor is sampled using a National Instrument DAQPad-6020E (12Bit, 100KS/S) and stored in Lab-View spreadsheet format. For simplicity, sensor data are randomly generated by the PIC processor rather than from real sensors or other sensor nodes.

Table 1. Latencies and Power consumption

	Stargate		Our WGN		
	Suspend Wi-Fi scheme	Suspend system scheme	Always on scheme	Power gating scheme	Power saving scheme
Enable Latency	0.485s	3.329s	-	0.28s	0.03s
Enable Power	0.751w	0.155w	-	0.545w	0.693w
Active Power	2.009w	2.009w	1.419w	1.419w	1.419w
Disable Latency	0.313s	0.757s	-	0s	0.003s
Disable Power	1.62w	1.11w	-	1.419w	1.32w
Sleep Power	0.751w	0.054w	-	5.13mw	0.495w

6 Experimental Results and Analysis

Average system power consumption is calculated[4] based on the amount of energy consumed in one working-period, which is defined as the period from the beginning of one active period to the beginning of the next active period. A power managed working-period can be further divided into four sub-periods: a period to enable the system, a period of doing the real work, a period to disable the system and a sleep period. A duty-cycle is calculated as the percentage of the time that a system does real work over an entire working-period. It does not include time spent in enabling or disabling the system. Note that we can maintain a fixed duty cycle by proportionally changing the working time and the duration of the working-period.

Table 1 lists the durations of these periods as well as the corresponding power consumptions in these periods for both the *Stargate* and our system running different power management schemes. We choose to compare our platform with the *Stargate* because it is one of a few gateway nodes commonly used in sensor networks. Other gateway nodes, such as those based on the Soekris board [18], share similar architecture as the *Stargate*.

For our WGN, the enable-power of the *power-gating* scheme is less than that of the *power-saving* scheme because not all components are powered up at the same time. The high sleep-power of our *power-saving* scheme is caused by the limitations of the Intersil chip as reported in [19]. The *Stargate* has very high sleep-power because its PXA255 processor is in charge of power management and can not be powered off completely.

In the remainder of this section, we compare the performance of our power management schemes using the WGN with the performance of the following two commonly used schemes on the *Stargate*:

1. Suspend-Wi-Fi Scheme: Suspend the Wi-Fi radio only.
2. Suspend-System Scheme: Suspend both the Wi-Fi radio and the *Stargate* computer.

The *Stargate* data are based on measurements from [5]. We combine the latencies that are reported separately for the Wi-Fi radio and the PXA255 and compute the average power consumption. Similar to our approach, the authors in [5] measure data

[4] $Average\ System\ Power = \frac{Total\ energy\ consumed\ in\ one\ working-period}{Duration\ of\ the\ working-period}$

without sensors attached. Accordingly, we use the “Processor Core Idle” data for the Suspend-Wi-Fi scheme and the “Proc./Radio Core Sleep” data for the Suspend-System scheme. The active power consumption in the active period is based on 50% TX and 50% RX. For our own schemes, the power consumed by the entire WGN is reported. The load on the 1Ohm resistor is included to simulate a real sensor. Our always-on scheme simply keeps the system in maximum power all the time and is used to serve as a baseline. The sleep-power of our *power-gating* scheme is the same as the sleep-power of the attached sensor node.

Fig. 3 shows the average system power consumption of the five schemes running under various working-periods and at a fixed 1% duty cycle, which is very common for sensor networks. With a fixed duty cycle, the time spent doing real work increases in proportion to the duration of the working-period, and therefore the average amount of real work per unit time remains constant. Large duty-cycle latencies mean less sleep time. The WGN running our *power-gating* scheme performs about 6 times better than the *Stargate* running the suspend-system scheme for large working-periods where the active power dominates. For short working-periods where the transition (enable/disable) power becomes dominant, we perform up to 7 times better. This is partially due to the small transition latencies that result from applying the 802.11b based power saving techniques described in section 5.2.

Fig. 4 shows the lifetimes of our WGN and the *Stargate* running the five schemes under various working-periods and at a fixed 1% duty cycle. They are computed based on a power supply of 2200 mAh at 3 volts from a pair of AA batteries. The WGN could last longer with smaller duty cycles because of the extremely low sleep-power.

Although it is possible with some hardware modifications to eliminate the sleep power of the *Stargate* processor by powering it off and to reduce the active-power of the *Stargate* processor by dynamic voltage scaling (DVS) [7, 3], our WGN would still perform better because of lower duty-cycle latencies. This is a direct result of the new architecture we propose.

While it seems that the enable/disable latencies are not important in terms of average system power consumption for applications that are dominated either by active or sleep power, they are critical in determining the responsiveness of a system. A large latency in activating a subsystem from low-power or power-off mode would be prohibitive for many sensor network applications to employ duty-cycling, not to mention the energy overhead associated with the delay. Smaller latencies also provide additional space for applications to trade latencies for energy savings. Fig. 5 shows the system response time under different power management schemes. When running power-gating scheme, our WGN is about 12 times better than the *Stargate* running the suspend-system scheme. The WGN running the power-saving scheme is about 16 times better than the *Stargate* running the suspend-Wi-Fi scheme.

7 Conclusions and Future Work

In this paper, we present the design and optimizations of a low power wireless gateway node. By introducing a dual-processor hardware architecture and a choice of appropriate duty-cycling of the processing and radio subsystems, we successfully reduce the

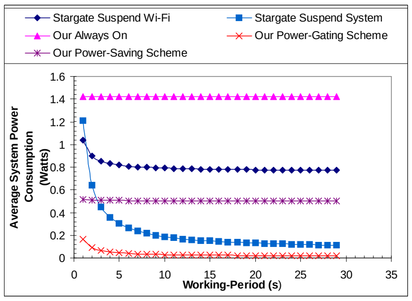

Fig. 3. Average system power consumption under various working-periods and at a fixed 1% duty cycle

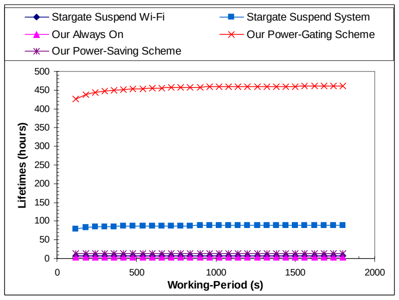

Fig. 4. Lifetimes under various working-periods and at a fixed 1% duty cycle (2200 mAh at 3 volts)

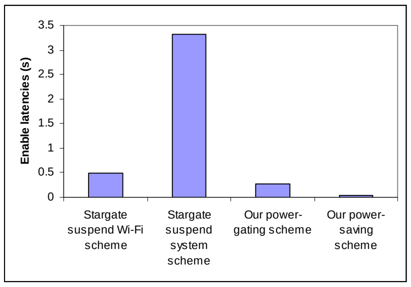

Fig. 5. System response time

standby power consumption while also providing support for spurts of high MIPS and high bandwidth connections. We are also able to improve the performance of duty-cycling with respect to energy and latencies by reducing software and networking protocol related overheads and through careful system integration. The result is a platform that supports efficient power management through duty-cycling. We believe our architecture can be useful for building other types of high performance nodes or the emerging Micro-servers.

In our ongoing work, we are exploring ways to optimize the performance of our WGN in sensor networks running various low power MAC protocols or wakeup protocols. We are also planning to replace our PIC based sensor nodes with Telos motes [6] in these experiments for compatibility with existing sensor network applications and for evaluation purposes.

References

1. Polastre, J., Hill, J., Culler, D.: Versatile low power media access for wireless sensor networks. In: SenSys ’04: Proceedings of the 2nd international conference on Embedded networked sensor systems, New York, NY, USA, ACM Press (2004) 95–107

2. Jejurikar, R., Gupta, R.: Dynamic voltage scaling for systemwide energy minimization in real-time embedded systems. In: ISLPED '04: Proceedings of the 2004 international symposium on Low power electronics and design, New York, NY, USA, ACM Press (2004) 78–81
3. McIntire, D., Ho, K., Yip, B., Singh, A., Wu, W., Kaiser, W.J.: The low power energy aware processing (leap)embedded networked sensor system. In: the Fifth International Conference on Information Processing in Sensor Networks, Nashville, Tennessee, USA, ACM Press (2006) 449–457 1127846 449-457.
4. Stargate http://www.xbow.com.
5. Margi, C.B., Petkov, V., Obraczka, K., Manduchi, R.: Characterizing energy consumption in a visual sensor network testbed. In: 2nd International IEEE/Create-Net Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities. (2006)
6. Polastre, J., Szewczyk, R., Culler, D.: Telos: Enabling ultra-low power wireless research. In: the Fourth International Conference on Information Processing in Sensor Networks. (2005)
7. Banerjee, N., Sorber, J., Corner, M.D., Rollins, S., Ganesan, D.: Triage: A power-aware software architecture for tiered microservers. Technical Report 05-22, University of Massachusetts-Amherst (April 2005)
8. Raghunathan, V., Pering, T., Want, R., Nguyen, A., Jensen, P.: Experience with a low power wireless mobile computing platform. In: the 2004 international symposium on Low power electronics and design, Newport Beach, California, USA, ACM Press (2004) 363–368 1013322 363-368.
9. Gu, L., Stankovic, J.: Radio-triggered wake-up capability for sensor networks. In: Real-Time and Embedded Technology and Applications Symposium, 2004. Proceedings. RTAS 2004. 10th IEEE. (25-28 May 2004) 27–36
10. Agarwal, Y., Schurgers, C., Gupta, R.: Dynamic power management using on demand paging for networked embedded systems. In: Asia South Pacific Design Automation Conference (ASP-DAC'05), China (2005) 755–759
11. Pottie, G.J., Kaiser, W.J.: Wireless integrated network sensors. Commun. ACM **43**(5) (2000) 51–58 332838.
12. Rahimi, M., Baer, R., Iroezi, O.I., Garcia, J.C., Warrior, J., Estrin, D., Srivastava, M.: Cyclops: in situ image sensing and interpretation in wireless sensor networks. In: SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems, New York, NY, USA, ACM Press (2005) 192–204
13. Schott, B., Bajura, M., Czarnaski, J., Flidr, J., Tho, T., Wang, L.: A modular power-aware microsensor with $>$1000x dynamic power range. In: IPSN '05: Proceedings of the 4th international symposium on Information processing in sensor networks, Piscataway, NJ, USA, IEEE Press (2005) 66
14. 802.11b Spec 1999 edition http://grouper.ieee.org/groups/802/11.
15. Ubicom IP2022 http://www.ubicom.com.
16. DPAC www.dpactech.com.
17. Ye, W., Heidemann, J., Estrin, D.: An energy-efficient mac protocol for wireless sensor networks. In: Infocom '02,, New York, NY (2002) 1567–1576
18. Hartung, C., Han, R., Seielstad, C., Holbrook, S.: Firewxnet: a multi-tiered portable wireless system for monitoring weather conditions in wildland fire environments. In: MobiSys 2006: Proceedings of the 4th international conference on Mobile systems, applications and services, New York, NY, USA, ACM Press (2006) 28–41
19. Pering, T., Raghunathan, V., Want, R.: Exploiting radio hierarchies for power-efficient wireless device discovery and connection setup. In: VLSID '05: Proceedings of the 18th International Conference on VLSI Design held jointly with 4th International Conference on Embedded Systems Design (VLSID'05), Washington, DC, USA, IEEE Computer Society (2005) 774–779