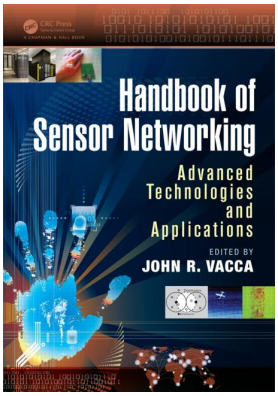

Handbook of Sensor Networking
Advanced Technologies and Applications

John R. Vacca

Sensor Network Platform and Operating Systems

Publication details
https://test.routledgehandbooks.com/doi/10.1201/b18001-4
Xinheng (Henry) Wang, Shancang Li
Published online on: 13 Jan 2015

2

Sensor Network Platform and Operating Systems

Xinheng (Henry) Wang
University of the West of Scotland

Shancang Li
University of the West of Scotland and University of Bristol

2.1 Introduction

Platform is defined as "a standard for the hardware of a computer system, which determines what kinds of software it can run" by Oxford Dictionary. As its definition implies, a platform defines the hardware and software of a system, also a standardized solution to the problems. The platform of wireless sensor network (WSN) is, therefore, a system of standardized hardware and software that enables sensing and wireless networking. This makes sensing and networking the two most important components in WSNs.

A good platform will bring many benefits to the system designer, manufacturer, and service operator. It will reduce the time of products to market. This is particularly attractive in modern competitive international market. Based on the basic platform design, further components could be added or removed from the platform to meet a specific demand. Thus, its basic hardware components and software could be reused by other applications. In addition, based on a basic platform and through further development, the bugs in the software and hardware faults could be corrected over and over to improve the performance of the platform. Further developed applications based on the basic platform will be more reliable, and a high-quality service could be guaranteed.

WSN platform comprises two subplatforms: one is the subplatform of individual network node; the other is the subplatform of networked nodes. In this chapter, we will first look at the design of individual node, and then we will look at the design and management of the network formed by these nodes.

2.2 Architecture of a WSN Node

For a WSN node to work properly and fulfill its basic functions, it has to have four basic components, namely, sensing, communication, processing, and power, as shown in Figure 2.1.

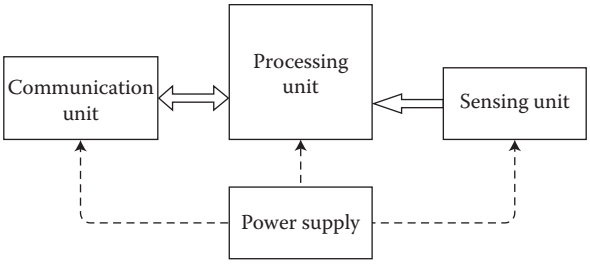

FIGURE 2.1 Basic architecture of a WSN node.

Processing unit plays an extremely important role in the platform. Based on the power and size of the platform, the processing unit could be classified into four levels, from embedded PCs to PDA-class platforms, such as Stargate [1] and CerfCube [2], and from low-power motes to even lower-power system-on-a-chip (SoC) platform.

Embedded PCs and PDA-class platforms roughly form the same class in terms of its hardware architecture. Both are employed with 32-bit CPUs and megabyte memories and secondary storage such as flash memory or disk. However, embedded PCs are no longer used nowadays because of the evolvement of technology and its size limitation that makes them difficult to be deployed. PDA-class platforms are mainly used as gateways to collect, store, and process the data from the sensor network and communicate with external networks. Here, Stargate and CerfCube are used as an example to illustrate the architecture design of this kind of platforms.

Stargate was designed by Intel and then licensed to Crossbow Technology for commercialization. The functioning block of a Stargate is shown in Figure 2.2.

Stargate employs an Intel PXA255 processor, which uses the Intel XScale™ microprocessor. XScale is a 32-bit RISC microarchitecture, which is noted for its efficiency by using less silicon transistors and thus consuming less power, making it smaller and less expensive to manufacture. It is also efficient in processing multimedia data through the support of 16-bit data types and enhanced 16-bit multiply and accumulates operations that accelerate the multimedia CODEC algorithms.

On Stargate board, working with PXA255 processor is the Intel SA-1111 companion chip, which enables direct access to one SDRAM system memory and one 32 MB flash memory and provides buffering for one PCMCIA slot and one compact flash slot for further storage. A MICA2/GPID connector and an optional I2C are also provided to connect the Crossbow's MICA2 sensor nodes and other electronic devices to Stargate smoothly. A watchdog timer, an LED, and a gas gauge can also be found embedded on the board. Stargate's daughter card provides extra interfaces for more functional components to be embedded on the board, including a 10Base-T Ethernet port, a USB port, a JTAG port, and an external ac power supply adapter port.

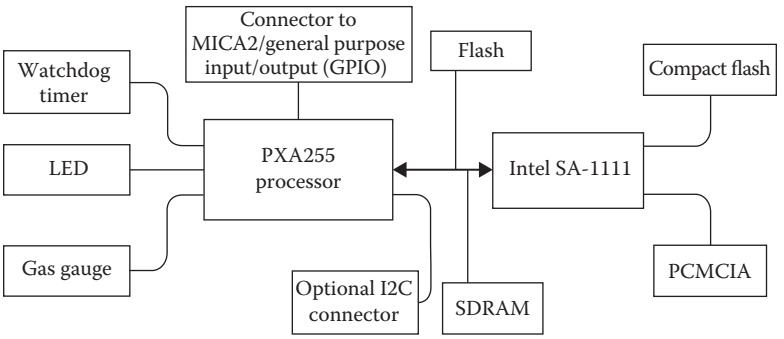

FIGURE 2.2 Stargate block diagram.

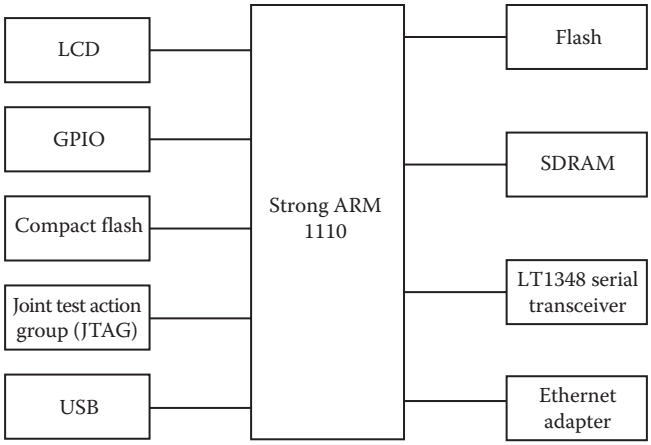

FIGURE 2.3 Block diagram of CerfBoard.

CerfCube represents another main series of platforms by implementing ARM processors, where XScale and ARM processor are two mainstream processors for handheld devices. CerfCube is the product of Intrinsyc Software International, Inc. Inside CerfCube is its board called CerfBoard, whose block diagram is shown in Figure 2.3.

CerfBoard has a few series of boards with different processors. Take CerfBoard V3.0 as an example, it integrates an Intel Strong ARM 1110 Microprocessor, 16 MB flash, 32 MB SDRAM, an Ethernet port, three RS232 serial ports, LCD interface, CompactFlash socket, USB interface, JTAG interface, and 16 digital I/O lines that all lines have programmable interrupt capacity and the first four have LED indicators.

Scaling down from PDA-class platforms integrated with processors like XScale and ARM is another popular class of sensor platforms. Motes are the most famous examples in this class. Motes are a legendary sensor platform, which was originally developed by UC Berkeley and licensed to industry for commercialization.

Take TelosB developed from Crossbow as an example to illustrate its architecture and functionalities. The block diagram is shown in Figure 2.4.

Telos (ver. B) platform employs an 8 MHz TI MSP430 microcontroller with 10 kB RAM, a 1 MB external flash for data logging, an IEEE 802.15.4/ZigBee compliant RF transceiver with integrated on-board antenna, a USB interface for data collection and programming, and an optional on-board sensors of light, temperature, and humidity.

Comparing with Stargate and CerfCube, there are a few significant differences in processing unit between motes and embedded PC. The main differences are listed in Table 2.1.

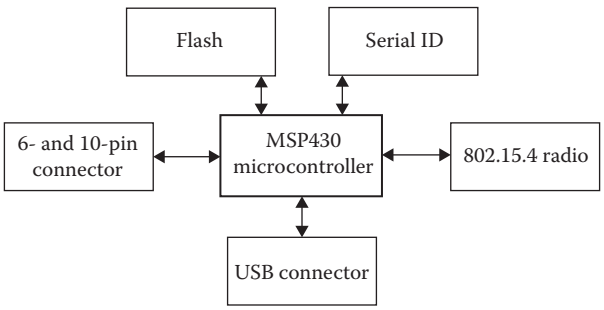

FIGURE 2.4 Block diagram of TelosB.

TABLE 2.1 Differences in Processing Unit between Motes and Embedded PC

		Motes	Embedded PC
Processor	Type	Microcontroller	Microprocessor
	CPU clock	Low, for example, 8–25 MHz for TI MSP430	High, for example, 100–400 MHz for PXA255
	Power consumption	Low, several milliwatts	High, several hundreds milliwatts
RAM		Integrated in microcontroller, a few kilobytes	External, dozens of megabytes
Flash		Integrated in microcontroller, size in kilobytes	External, size in megabytes

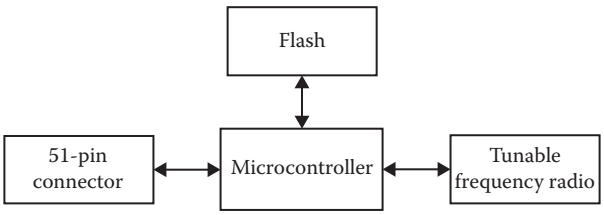

FIGURE 2.5 Block diagram of a MICA2 node.

The microcontroller is a small device on a single integrated circuit containing a processor core, memory, and programmable I/O peripherals. Flash memory and a small amount of RAM are also integrated on chip. Because it operates at low clock rate frequency and low bit words, it consumes far less power than microcontroller used in embedded PC, which enables it suitable for developing battery-powered sensor node.

The latest successor of TelosB is MICA2 mote. The block diagram of a MICA2 node is shown in Figure 2.5.

MICA2 features several improvements over its predecessors, including tunable multichannel transceiver, TinyOS distributed operating system, and support for wireless remote programming. MICA2 platform employs an Atmel ATmega128L microcontroller that runs TOS from its internal flash memory. Using TOS, the processor can be configured to run sensor application/processing, and the network/radio communications stack simultaneously. Its 51-pin expansion connector supports analog inputs, digital I/O, I2C, SPI, and UART interfaces. These interfaces make it easy to connect to a wide variety of external peripherals [3]. These features would enable it to be more suitable for commercial applications. However, it is still based on microcontroller architecture. Many varieties of this class of platforms are available on the market for research and commercial applications. Readers may find dozens of platforms easily on Internet or published papers. The most popular examples include BTnode [4], Nymph [5], iSense [6], and Smart Dust [7].

The latest development of WSN platform is the SoC platform. SoC is a further step from microcontroller-based platform to integrate all components into a single chip, including communication unit and sensing unit. A key advantage of SoC is that the size of device will be reduced, which is particularly attractive to design wearable sensor networks or body area networks. Here, an example of monitoring electrocardiography (ECG) by a SoC solution is illustrated in [8] to demonstrate its architecture design.

The platform architecture is shown in Figure 2.6. As shown in Figure 2.6, this SoC platform has five functional blocks, including (1) a system startup module (SSM) for remote battery control and initial frequency allocation of communication unit that a 5% duty-cycled body-channel communication

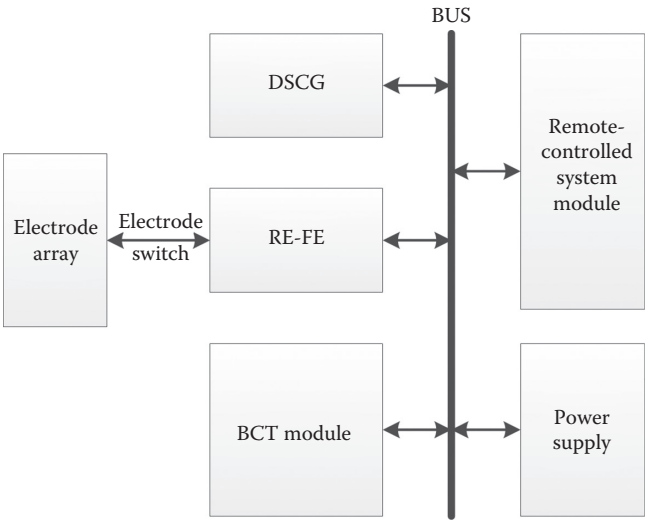

FIGURE 2.6 Platform architecture of a SoC design for monitoring ECG.

is used; (2) four reconfigurable electrode sensor front ends (RE-FE), with each connected to four voltage sensing electrodes to achieve reconfigurable sensing and digitization; (3) a differential sinusoidal current generator (DSCG) for a 5% duty-cycled high-quality balanced current injection; (4) a digital module containing a finite state machine controller with special-purpose registers, a 20 kB SRAM for data storage, a 10th-order FIR filter, an 8:1 compression block, and a packet encoder/decoder; and (5) a duty-cycled body-channel transceiver (BCT) for low-energy external data communication.

There are two significant differences between this SoC-based design and previously described embedded PC and microcontroller-based motes: one is its controller design, and the other is the communication design, which will be discussed later. The controller used in this is a state machine based one.

A state machine is generally defined as a mathematical model of computation used to design both computer programs and sequential logic circuits. It is conceived as an abstract machine that can be in one of a finite or infinite number of states. A finite state machine is a state machine with a finite number of states.

Generally, a state machine is not good for numerical calculation but good for any system that occupies various distinct states. In addition, a state machine is good for any applications with state changes, such as user interface design, pattern recognition tasks, and communication protocols. Also, state machines could be used to construct simple controllers for simple tasks, for example, the controller in ECG monitoring that we have just discussed.

A key advantage of a state machine is its ease to modify. This characteristic makes the state machines the backbone of field-programmable gate array (FPGA), which is a reconfigurable design. This forms another category of sensor platform design, which is a reconfigurable sensor platform.

SoC in designing sensor network platform is still in its infantry stage. A lot of efforts are required to evaluate the performance, flexibility, cost effectiveness, and also environmental resilience of this kind of design.

2.3 Reconfigurable WSN Platforms

Different from microprocessor-based WSN platform, reconfigurable, this is also referred to as reprogrammable, but reconfigurable is used throughout this chapter to reduce the confusion. WSN platform offers the flexibility in implementing application-driven functionalities and ability in reducing the energy consumption, which is critical to WSN applications.

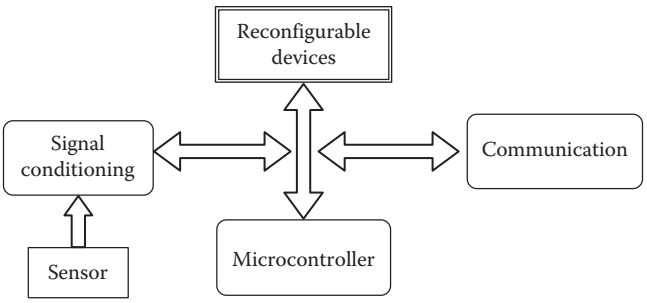

FIGURE 2.7 Block diagram with microcontroller and reconfigurable device.

Recent development has seen two types of hardware implementation in terms of reconfigurability: one is the combination of microcontroller with reconfigurable device, which I called half reconfigurability, and the other is pure reconfigurable device without microcontroller. Examples of the first type of implementation could be found in [9,10]. Its cooperation with other modules like microcontroller, sensing, and communication is illustrated in Figure 2.7.

As shown in Figure 2.7, a reconfigurable device is controlled by a microcontroller. Its function acts like a gate for microcontroller. It generates control signals for the signal conditioning unit. Whenever it detects data acquired from the sensor, it might trigger the microcontroller to accept it.

An example of a pure reconfigurable device–based WSN platform is demonstrated by Nokia wrist-attached wearable devices [11]. Its system architecture is shown in Figure 2.8. In this example, the reconfigurable device is FPGA, which has the full control of the platform.

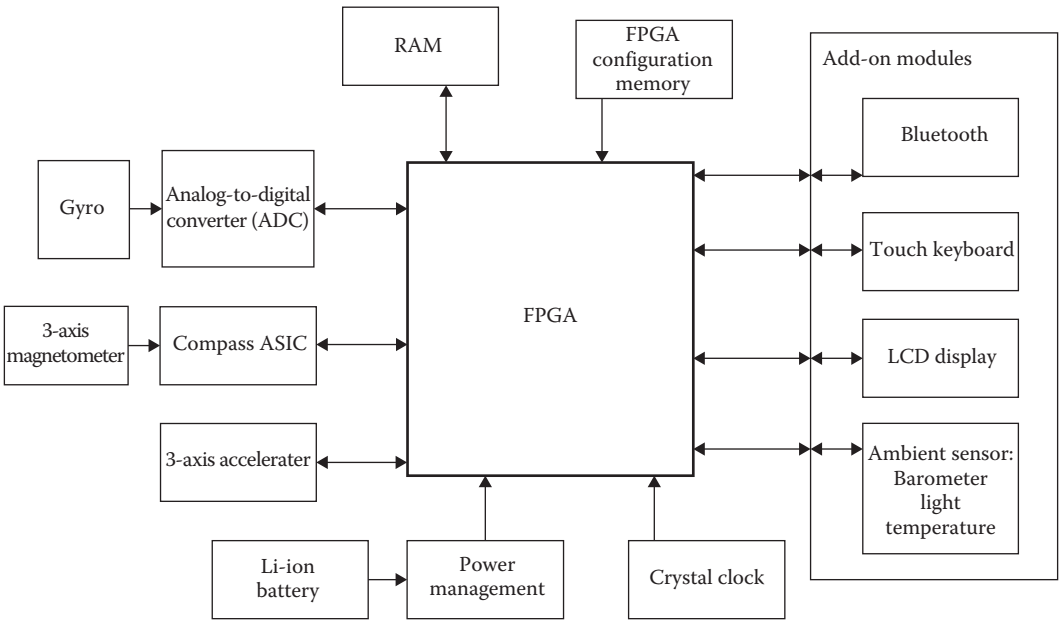

FIGURE 2.8 FPGA-based WSN platform.

2.4 Sensing

While processing unit is the core of a WSN platform that controls the data acquisition, transmission, and possible data processing, sensing unit is the one that interacts with the environment to convert the physical reading into a signal that the system can read and understand. With the development of electronic and manufacturing technologies, sensors are becoming smaller and more accurate. However, electronics and manufacturing are out of the scope of this book; further details about sensor technologies can be found from the handbook *Sensor Technology Handbook* [12].

2.5 Communications

The communication unit plays another important role in a WSN. It decides how the sensor nodes are working together and how the sensor nodes transmit the acquired physical data to a data receiving unit. This is the unit that forms wireless sensor nodes into a network.

The communication unit forms the network into different scales and formats, such as one-to-one simple network, one-to-many or many-to-one network, or many-to-many networks scaling from small to large with few to hundreds or thousands of nodes.

Strictly to WSNs, communication is done via wireless communication protocols over RF channel. A few popular wireless communication protocols have been developed and widely used in WSNs because of their characteristics in forming reliable, large-scale WSNs or commercial maturity in some applications. Some other protocols are also used in some critical environments.

In terms of WSN, ZigBee is possibly the number one protocol because of its low power consumption and capability in forming wireless mesh and ad hoc networks, which enrich the applications of WSNs. The WSN formed by ZigBee could reach a very large scale, theoretically supporting 65,535 network nodes [13]. However, in practice, it is impossible to reach such a high number because of various constraints. The largest network seen so far is more than 800 nodes, implemented in University of California, Berkeley, and the Intel Berkeley Research Lab [14]. Military networks may be larger, but no report is released.

The basis of ZigBee is the IEEE 802.15.4 standard, which specifies the physical layer and media access control for low-power wireless networks. Other wireless networking protocols, such as ISA100.11a, WirelessHART, and MiWi, are also based on this standard.

Bluetooth is possibly the second popular networking protocols in forming a WSN. The networking architecture of Bluetooth is quite different from ZigBee. Bluetooth uses master–slave model to form a network, where a master node can communicate with a maximum of seven slave nodes. This forms a *piconet*. However, the slave node can also act as a master node and communicate with another seven slave nodes, which forms a scatternet. However, because of the limitation of direct communication between slave nodes and formation of the network, the size of the network formed by Bluetooth normally is not big.

The popularity of applying Bluetooth in WSNs is because of commercial availability and support of many electronic devices. Bluetooth is widely used in health-care monitoring where a large sensor network is not necessary.

The advantage of Bluetooth is its higher transmission rate comparing with ZigBee, with the price of high power consumption. However, with the introduction of the latest version of Bluetooth low energy (BLE), Bluetooth still stands a strong candidate in WSNs in some applications like health-care monitoring.

Ultra wideband (UWB) is another technology quite often used in WSNs. One main characteristic of UWB is its high data transmission speed. This makes it a best choice in WSNs for video transmissions, that is, video surveillance [15]. Another characteristic of UWB is its ability in accurate localization, which makes it one of the best tools in location and tracking in WSNs, that is, outdoor sports [16]. The basis of UWB is IEEE 802.15.4a, which specifies the physical layer of UWB and also chirp spread spectrum (CSS).

Because of the advantages of WSNs and industrial needs, but limitations of technologies of ZigBee, Bluetooth, UWB, etc., in industrial environment, two specific industrial standards were developed, namely, WirelessHART [17] and ISA100.11a [18]. WirelessHART is the enabling wireless capability on existing HART protocol. The three key capabilities claimed by the developer are reliability, security and privacy, and effective power management, which make it a good option in process plant. At the same time, ISA100.11a aims to provide secure and reliable wireless communication for noncritical monitoring and control applications. They have used the similar physical layer technologies that are specified by the IEEE 802.15.4 standard but different architecture and technologies in data link layer and network layer, which enable them to support different topologies of the networking, however reliable and secure in process and manufacturing industries.

Both protocols compete in the same market to become the de facto standard. The competition is going on. The performance of these two protocols is being improved. However, who is going to win stands a mystery.

A detailed comparison between these two protocols is out of the scope of this chapter. However, readers can refer to the reference for further details [19].

IEEE 802.15.4 is the standard that specifies the physical layer and media access control layer for low-rate wireless personal area networks. It is the basis for the wireless protocols described earlier, such as ZigBee, Bluetooth/BLE, UWB, WirelessHART, and ISA100.11a. In order to enable these protocols to operate for IPv6 network addressing, a new protocol IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN) was developed [20]. 6LoWPAN has defined the encapsulation and header compression mechanisms. Examples of applications of 6LoWPAN could be seen in hospital monitoring [21], home automation design [22], and agricultural environment monitoring [23].

The standard protocols discussed earlier are widely used in WSNs to form the single-hop or multihop wireless networks for monitoring and control. In order to enable them to work efficiently in WSNs, some other concerns such as mobility management, energy-efficient routing, and source scheduling should be considered as well. However, this is out of the scope of this chapter.

Standard protocols have obvious advantages in applications, that is, ease of implementation. However, the coexistence of some of the protocols is a big problem because of sharing of the same frequency band, that is, 2.4 GHz frequency band, which causes serious interference problems. This is more critical in some sensitive applications like health care. Therefore, nonstandard communication technologies are quite often used in this kind of applications. For example, in the aforementioned SoC implementation of monitoring cardiac disease, a 5% duty-cycled body-channel communication link is implemented for data transmission, which is more energy efficient.

2.6 Power Unit

Power unit is the simplest unit in a WSN platform but plays an important role in the determination of the lifetime of the WSNs. Because most of the sensor nodes are battery powered, how to make the battery work longer to extend the lifetime of the network node is becoming a core issue that the WSN designer, developer, and applicant have to consider. In order to reduce the power consumption in a WSN platform, quite a few measures have been taken, including scheduling wake-up and sleep pattern, adoption of low-power microcontroller and SoC design, application of low-power communication protocols, energy-efficient routing protocols, new design of cross-layer protocols, and task scheduling schemes. This involves the consideration of every aspect of a WSN platform.

Except those techniques mentioned earlier, one popular technique to make the battery/platform work longer is the application of energy harvesting technology by utilizing external energy sources. Energy harvesting, also known as power harvesting or energy scavenging, is a process by which energy is derived from external sources, for example, solar power, thermal energy, wind energy, salinity gradients, and kinetic energy, and stored for the wireless sensor node to work for longer time [24]. All these forms of energy harvesting have been successfully used in WSN platforms.

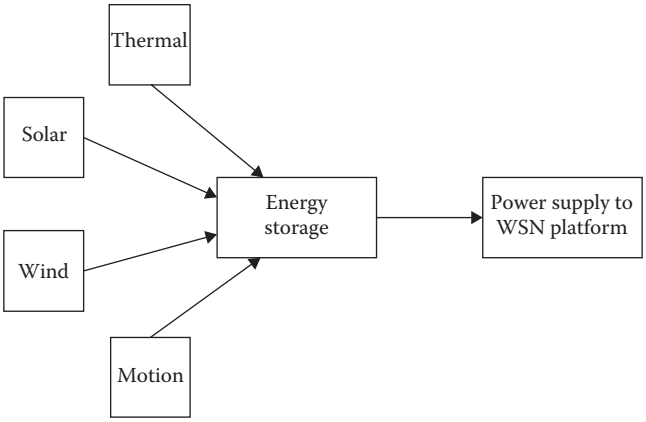

FIGURE 2.9 Schematic approach of energy harvesting.

The schematic approach of energy harvesting is simple, shown in Figure 2.9. Relevant devices convert solar, wind, thermal, or motion into energy. Then, the energy will be stored in a WSN platform by the form of recharging the battery or supercapacitor. Rechargeable battery or supercapacitor can work with or without battery on the WSN platform to supply power. Depending on the system requirement or environment, one or more forms of energy harvesting technologies could be used together such as solar and wind.

Solar power harvesting is possibly the most popular energy harvesting technology used for WSNs. Its solution is simple; normally, a solar panel is attached to the WSN platform. Examples of applying solar power harvesting include AmbiMax [25], Heliomote [26], Prometheus [27], and Everlast [28]. Although solar harvesting is a mature technology, one important issue still needs to be considered. This is efficiency. Efficiency is important in energy harvesting because it affects the cost, design, and lifetime of the system. In order to increase the efficiency, one key technique is the application of maximum power point tracking (MPPT) to obtain the maximum available power. MPPT is widely used in solar energy harvesting but also applicable in wind generation and all other energy harvesting.

While solar and wind are popular energy resources for WSNs in environment monitoring, they are hardly used in body area sensor networks for health-care monitoring because of the inability in deploying such kind of extra devices around the body. Energy harvesting for health-care monitoring depends on the energy generated from the human body itself, such as thermal energy and motion energy. For example, authors in [29] demonstrated a thermal energy harvesting technology to provide power for monitoring ECG, electromyography (EMG), or electroencephalography (EEG) signals, illustrated in Figure 2.10. They have used a commercial off-the-shelf (COTS) thermoelectric generator, which usually provides a very low output voltage (<100 mV) that is not high enough to power the sensor system to work. In order to solve this problem, the authors designed a boost converter of converting voltages as low as 30 mV. The energy harvested from the thermal is able to make the signal acquisition, signal processing, and transmission work without a battery. A batteryless sensor node is developed, which is ideal for health monitoring.

Another popular source of energy generated from the human body is the motion, for example, movement of limbs. The transformation from the motion to electrical power could be done by any kind of electromechanical transduction, such as electrostatic, electromagnetic, and piezoelectric. While design of electrostatic and electromagnetic transduction is more complex, piezoelectric becomes the main form for transferring body motion to electrical energy.

The phenomenon of piezoelectric is that the electric charge could be accumulated on some materials under mechanical stress, as illustrated in Figure 2.11. Therefore, to make piezoelectric happen,

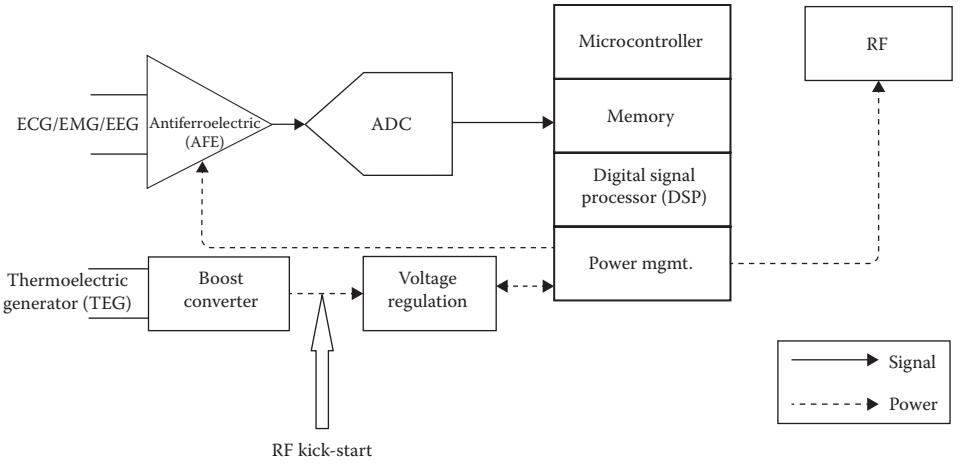

FIGURE 2.10 Thermal harvesting for body area sensor networks.

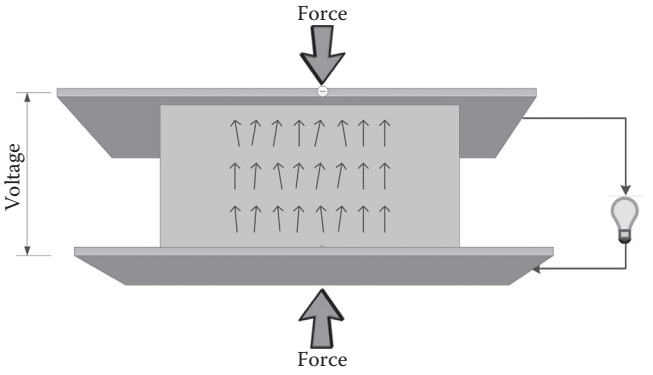

FIGURE 2.11 Illustration of piezoelectric.

the most important is to find the right material. One popular material used in piezoelectric harvesting is polyvinylidene fluoride (PVDF).

The history of piezoelectric energy harvesting is not long. Over the last decades, a few application technologies have been developed, by using the movement of the human body like walking and movement inside the human body like respiration and blood flowing. The first one to use it to harvest the energy from human walking to powering the electronic devices is researchers from the Media Lab at Massachusetts Institute of Technology (MIT) in the late 1990s [30]. At the same time, the pioneers of harvesting the energy from the motion of ribs when respiring [31] are researchers from Universität des Saarlandes and fluctuation of blood pressure [32] are from University of Pittsburgh, where these technologies are used to power implantable devices. Details of these technologies could be found from the references.

2.7 Operating Systems

Section 2.2 introduced the architecture and key components of a wireless sensor platform. These are the hardware part. How to make this hardware to work effectively is the job of operating system. This section will introduce the operating system for a wireless platform.

When talking about WSN operating systems, TinyOS is the one that cannot be missed. TinyOS, written in nesC language, is the legendary operating system, originally developed for Crossbow's mote sensor nodes.

Regarding TinyOS, there are numerous papers, books, and wikis online to introduce it. A research paper titled "TinyOS: an open operating system for wireless sensor networks" [33], written by researchers who developed this program, has rich sources about the techniques and features about TinyOS. Here, we briefly introduce the key features of TinyOS.

Because of the key characteristics of WSNs of low power, low cost, small size, distributed, concurrency of multiple operations on one board, and application driven, TinyOS was designed to meet the requirements of these characteristics. In order to do that, two approaches are adopted throughout the course of design and development, namely, event centric and platform for innovation, where flexibility is focused to enable the platform for various applications.

TinyOS is a component-based embedded system. A component is a computational entity that has one or more interfaces. Applications are developed by connecting the interfaces of the components.

Components abstract the hardware by a three-layer hierarchy, called hardware abstraction architecture (HAA), as shown in Figure 2.12.

HAA employs three sublayers, hardware presentation layer (HPL), hardware adaptation layer (HAL), and hardware interface layer (HIL). HPL is right on top of the hardware to represent the full functional capacity of the hardware. It is hardware dependent but hides the most hardware-dependent codes in order to develop a higher-level abstraction. The middle layer of HAA is HAL. HAL is built on top of HPL to build useful abstractions. In contrast to HPL, it has two key characteristics: one is that state is allowed to be maintained in this layer for performing arbitration and resource control, and the other is that hardware-specific applications can be built, as illustrated in Figure 2.12. On the other hand, the top layer HIL is doing the opposite job. It will convert the hardware-specific abstractions provided by HAL to hardware independent for cross-platform applications.

From the aforementioned description, it can be seen that two types of applications could be developed in TinyOS: one is hardware dependent, based on HAL layer, and the other is hardware independent, based on HIL. These two types of applications could be run in parallel. Now let us have a look at how TinyOS interacts with the four key modules in a WSN platform.

Currently, TinyOS supports three types of microcontrollers, namely, Atmel ATmega128L, Texas Instruments MSP430, and Intel XScale PXA271, and a few platforms developed by using one of the three types of microcontrollers, for example, the TelosB, MICA2, MICA2DOT, MICAz, TinyNode, and BTnode3.

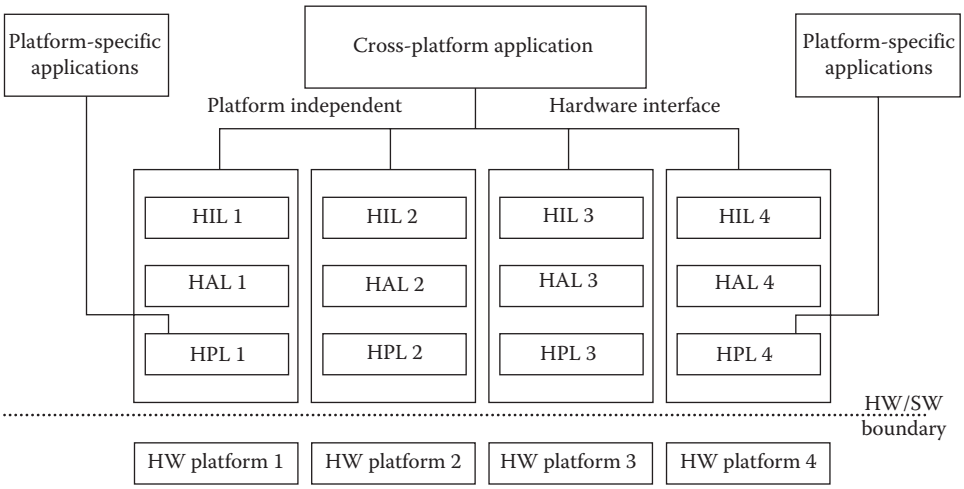

FIGURE 2.12 Hardware abstract architecture.

For sensors, data acquisition components are built on HIL interfaces. If a special component is required, for example, data acquisition for high frequency or accurate sampling, this component has to be built on HAL.

For communications, TinyOS adopts an unusual cross-layer optimization approach to develop a low-power stack. The current radio chips that it supports are CC1000, CC2420, CC2530, AT86RF212, and AT86RF230. In addition, various routing protocols have been developed for networking the sensor nodes into a WSN. In terms of power management, which is critical to the applications of WSNs, TinyOS divides the management into two types depending on the type of devices, namely, microcontroller power management and device power management.

Aside from TinyOS, some other tiny RTOSs have also gained the attention from a large community, such as UCOS [34], eCos [35], and μCLinux [36]. All these software packages are open source to support the research community. Some commercial ones are also available on the market, including MicroC/OS-II (Micrium), Nucleus (Mentor Graphics), and ThreadX (Express Logic). Different software packages have different features. Readers will find the details from the references.

2.8 Summary

This chapter describes the WSN platform and software. It started from describing the architecture of sensor nodes, scaling from embedded PC, microcontroller-based sensor node, to SoC. Then the details of each module are discussed, such as processing, communications, sensing, and power supply. Since the power consumption is a key element to the operation of WSNs, measures of energy harvesting are discussed to prolong the life of WSNs. Last, the operating system of WSN was discussed by taking TinyOS as an example.

References

1. Stargate platform. http://platformx.sourceforge.net/. Accessed on April 16, 2014.
2. CerfCube platform. http://ecos.sourceware.org/ecos/boards/cerfcube.html. Accessed on April 16, 2014.
3. MICA2 Wireless Measurement Systems. Document Part Number: 6020-0042-04. http://www.eol.ucar.edu/isf/facilities/isa/internal/CrossBow/DataSheets/mica2.pdf. Accessed on April 16, 2014.
4. Beutel, J., Kasten, O., and Ringwald, M., BTnodes—A distributed platform for sensor nodes, *First International Conference on Embedded Networked Sensor Systems, SenSys 2003*, Los Angeles, CA, November 5–7, 2003.
5. Chen, J., Fearing, R.S., Armstrong, B., and Burdick, J., NYMPH: A multiprocessor for manipulation applications, *Proceedings of the IEEE International Conference on Robotics and Automation*, San Francisco, CA, April 1986, Vol. 3, pp. 1731–1736.
6. Buschmann, C. and Pfisterer, D., iSense: A modular hardware and software platform for wireless sensor networks, 6. Fachgespräch Sensornetzwerke, 15, 2007, pp. 15–18.
7. Kahn, J.M., Katz, A.R.H., and Pister, A.K.S.J., Next century challenges: Mobile networking for "Smart Dust", *Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, Seattle, WA, 1999, pp. 271–278.
8. Yan, L., Bae, J., Lee, S., Roh, T., Song, K., and Yoo, H.-J., A 3.9 mW 25-electrode reconfigured sensor for wearable cardiac monitoring system, *IEEE Journal of Solid-State Circuits*, 46(1), 353–364, January 2011.
9. Portilla, J., Riesgo, T., and De Castro, A., A reconfigurable FPGA-based architecture for modular nodes in wireless sensor networks, *Third Southern Conference on Programmable Logic, 2007* (*SPL'07*), Mar del Plata, Argentina, February 28–26, 2007, pp. 203–206.
10. Jovanov, E., Milenkovic, A., Basham, S., Clark, D., and Kelley, D., Reconfigurable intelligent sensors for health monitoring: A case study of pulse oximeter sensor, *Proceedings of IEEE Engineering in Medicine & Biology Society*, 7, 4759–4762, 2004.

11. Ahola, T., Korpinen, P., Rakkola, J., Ramo, T., Salminen, J., and Savolainen, J., Wearable FPGA based wireless sensor platform, *Proceedings of IEEE Engineering in Medicine & Biology Society*, Lyon, France, pp. 2288–2291, August 22–26, 2007.
12. Wilson, J., *Sensor Technology Handbook*, Elsevier, Amsterdam, the Netherlands, 2004.
13. Pan, M.-S., Tsai, C.-H., and Tseng, Y.-C., The orphan problem in ZigBee wireless networks, *IEEE Transactions on Mobile Computing*, 8(11), 1573–1584, November 2009.
14. Largest Tiny Network Yet. http://webs.cs.berkeley.edu/800demo/, 2001. Accessed on April 16, 2014.
15. Huang, X., Dutkiewicz, E., Gandia, R., and Lowe, D., Ultra-wideband technology for video surveillance sensor networks, *2006 IEEE International Conference on Industrial Informatics*, Singapore, August 16–18, 2006, pp. 1012–1017.
16. Oppermann, I., Stoica, L., Rabbachin, A., Shelby, Z., and Haapola, J., UWB wireless sensor networks: UWEN—A practical example, *IEEE Communications Magazine*, 42(12), S27–S32, December 2004.
17. WirelessHART protocol. http://www.hartcomm.org. Accessed on April 16, 2014.
18. ISA100.11a protocol. http://www.isa100wci.org/. Accessed on April 16, 17, 2014.
19. Petersen, S. and Carlsen, S., WirelessHART versus ISA100.11a: The format war hits the factory floor, *IEEE Industrial Electronics Magazine*, 5(4), 23–34, December 2011.
20. http://6lowpan.net/. Accessed on April 16, 2014.
21. Jara, A.J., Zamora, M.A., and Skarmeta, A.F.G., HWSN6: Hospital wireless sensor networks based on 6LoWPAN technology: Mobility and fault tolerance management, *International Conference on Computational Science and Engineering, 2009* (*CSE'09*), Vancouver, British Columbia, Canada, August 29–31, 2009, Vol. 2, pp. 879–884.
22. Tudose, D.S., Voinescu, A., Petrareanu, M., Bucur, A., Loghin, D., Bostan, A., and Tapus, N., Home automation design using 6LoWPAN wireless sensor networks, *2011 International Conference on Distributed Computing in Sensor Systems and Workshops* (*DCOSS*), Barcelona, Spain, June 27–29, 2011, pp. 1–6.
23. Wang, X. and Yin, X., Agricultural environment real-time monitor and control system based on 6LoWPAN sensor networks, *Transactions of the Chinese Society of Agricultural Engineering*, 26(10), 224–228(5), October 2010.
24. Jain, A., and Bhullar, S., Emerging Dimensions in the Energy Harvesting, *IOSR Journal of Electrical and Electronics Engineering*, 3(1), 70–80, November and December, 2012.
25. Park, C. and Chou, P.H., AmbiMax: Autonomous energy harvesting platform for multi-supply wireless sensor nodes, *Third Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, 2006* (*SECON'06*), Reston, VA, September 25–28, 2006, Vol. 1, pp. 168–177.
26. Raghunathan, V., Kansal, A., Hsu, J., Friedman, J., and Srivastava, M.B., Design considerations for solar energy harvesting wireless embedded systems, *Fourth International Symposium on Information Processing in Sensor Networks, 2005* (*IPSN'05*), Los Angeles, CA, April 15, 2005, pp. 457–462.
27. Jiang, X., Polastre, J., and Culler, D., Perpetual environmentally powered sensor networks, *Proceedings of the Fourth International Symposium on Information Processing in Sensor Networks*, Los Angeles, CA, 2005.
28. Simjee, F. and Chou, P.H., Everlast: Long-life, supercapacitor-operated wireless sensor node, low power electronics and design, *Proceedings of the 2006 International Symposium on Low Power Electronics and Design* (*ISLPED'06*), Tegernsee, Germany, October 4–6, 2006, pp. 197–202.
29. Zhang, F., Zhang, Y., Silver, J., Shakhsheer, Y., Nagaraju, M., Klinefelter, A., Pandey, J. et al., A batteryless 19 μW MICS/ISM-band energy harvesting body area sensor node SoC, *2012 IEEE International Solid-State Circuits Conference Digest of Technical Papers* (*ISSCC*), San Francisco, CA, February 19–23, 2012, pp. 298–300.
30. Kymissis, J., Kendall, C., Paradiso, J., and Gershenfeld, N., Parasitic power harvesting in shoes, *Proceedings of the Second IEEE International Conference on Wearable Computing*, Pittsburgh, PA, USA, 1998, pp. 132–139.
31. Hausler, E. and Stein, E., Implantable physiological power supply with PVDF film, *Ferroelectrics*, 60(1), 277–282, October 1984.

32. Ramsay, M.J. and Clark, W.W., Piezoelectric energy harvesting for bio-MEMS applications, *Proceedings of SPIE*, 4332, 429–438, 2001.
33. Levis, P., Madden, S., Polastre, J. Szewczyk, R., Whitehouse, K., Woo, A., Gay, D., Hill, J., Welsh, M., Brewer, E., and Culler, D., TinyOS: An Operating System for Sensor Networks, *Ambient Intelligence*, Springer Berlin Heidelberg, 2005, pp. 115–148.
34. UCOS operating system. http://micrium.com/rtos/ucosii/overview/. Accessed on April 16, 2014.
35. eCos operating system. http://ecos.sourceware.org/. Accessed on April 16, 2014.
36. μCLinux operating system. http://www.uclinux.org/. Accessed on April 16, 2014.