Zigbee Protocol

Related terms:

Security, Application Layer, Gateway, Bluetooth, Protocol Layer

View all Topics

Learn more about Zigbee Protocol

ZigBee and IEEE 802.15.4 Protocol Layers

Shahin Farahani, in ZigBee Wireless Networks and Transceivers, 2008

3.1 Zigbee and IEEE 802.15.4 Networking Layers

ZigBee wireless networking protocol layers are shown in Figure 3.1. The ZigBee protocol layers are based on the International Standards Organization (ISO) Open System Interconnect (OSI) basic reference model [1]. There are seven layers in the ISO/OSI model, but ZigBee implements only the layers that are essential for low-power, low-data-rate wireless networking. The lower two layers (PHY and MAC) are defined by the IEEE 802.15.4 standard [2]. The NWK and APL layers are defined by the ZigBee standard [3]. The security features are defined in both standards. A network that implements all of the layers in Figure 3.1 is considered a ZigBee wireless network.

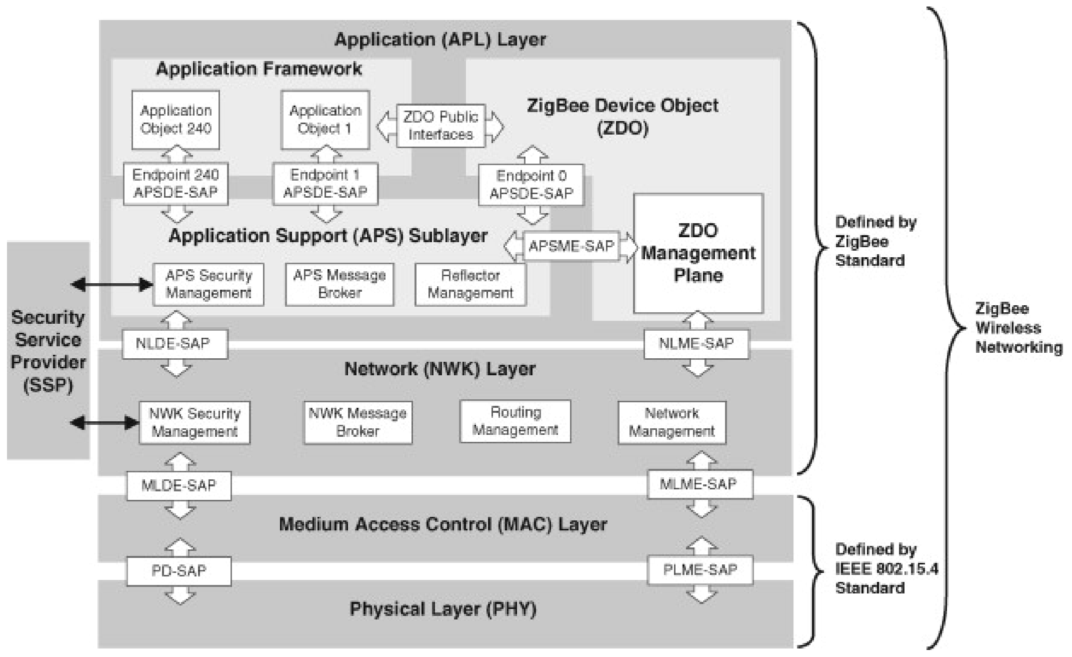


Figure 3.1. ZigBee Networking Protocol Layers

Each layer communicates with the adjacent layers through service access points (SAPs). A SAP is a conceptual location at which one protocol layer can request the services of another protocol layer. For example, in Figure 3.1, the PHY Data Service Access Point (PD-SAP) is where the MAC layer requests any data service from the PHY layer.

> Read full chapter

Lower-power wireless mesh networks for machine-to-machine communications using the IEEE802.15.4 standard

T. Watteyne, in Machine-to-machine (M2M) Communications, 2015

4.4.2 ZigBee

The ZigBee Alliance was arguably the first to finalize a complete protocol stack for low-power mesh networks. The ZigBee protocol stack is rooted entirely in the legacy IEEE802.15.4-2003 and IEEE802.15.4-2006 MAC protocol. It differentiates between full-function devices (FFD) and reduced-function devices (RFD), where the former are typically used as routing nodes, the latter as nonrouting leaf nodes. A ZigBee network consists of a number of FFDs collecting data from RFDs located around it.

A ZigBee network is not time-synchronized. Routing nodes therefore need to leave their radio on all the time to be ready to receive data packets from an RFD at any time. In a practical setting, this often means having the router nodes be mains-powered. As a result, most ZigBee networks have been used in a “star” topology, where RFDs send data to single collector FFDs a single hop away.

A ZigBee network operates on a single frequency channel. While it is possible to reconfigure the complete network to communicate at a different frequency, the network does not channel hop and is therefore prone to the effects of external interference and multipath fading highlighted above.

> Read full chapter

ZigBee/IEEE 802.15.4 Networking Examples

Shahin Farahani, in ZigBee Wireless Networks and Transceivers, 2008

Publisher Summary

ZigBee networking has a diverse range of application scenarios in which ZigBee devices can increase efficiency or reduce cost. Full ZigBee protocol implementation has the advantage of reliable mesh networking capability. However, if the application is simple, it might be possible to implement only IEEE 802.15.4 layers. Home automation is one of the major application areas for ZigBee wireless networking. Home automation uses include security systems, meter reading systems, irrigation systems, light control systems, and multizone heating, ventilation, and air-conditioning (HVAC) systems. In consumer electronics, ZigBee can be used in wireless remote controls, game controllers, a wireless mouse for a personal computer, and many other applications. This chapter reviews the application of ZigBee in wireless remotes. At the industrial level, ZigBee mesh networking can help in areas such as energy management, light control, process control, and asset management. This application includes asset management and personnel tracking and livestock tracking. Furthermore, ZigBee also finds its application in the sector of healthcare. A ZigBee gateway provides the interface between a ZigBee network and other networks, such as an Internet Protocol (IP) network, thus helping hospitals improve patient care and relieve hospital overcrowding by enabling them to monitor patients at home. Some other applications of ZigBee devices include hotel room guest access and fire extinguishers.

> Read full chapter

ZigBee Basics

Shahin Farahani, in ZigBee Wireless Networks and Transceivers, 2008

1.4 The Relationship Between ZigBee and IEEE 802.15.4 Standards

One of the common ways to establish a communication network (wired or wireless) is to use the concept of *networking layers*. Each layer is responsible for certain functions in the network. The layers normally pass data and commands only to the layers directly above and below them.

ZigBee wireless networking protocol layers are shown in Figure 1.3. ZigBee protocol layers are based on the Open System Interconnect (OSI) basic reference model [9]. Dividing a network protocol into layers has a number of advantages. For example, if the protocol changes over time, it is easier to replace or modify the layer that is affected by the change rather than replacing the entire protocol. Also, in developing an application, the lower layers of the protocol are independent of the application and can be obtained from a third party, so all that needs to be done is to make changes in the application layer of the protocol. The software implementation of a protocol is known as protocol stack software.

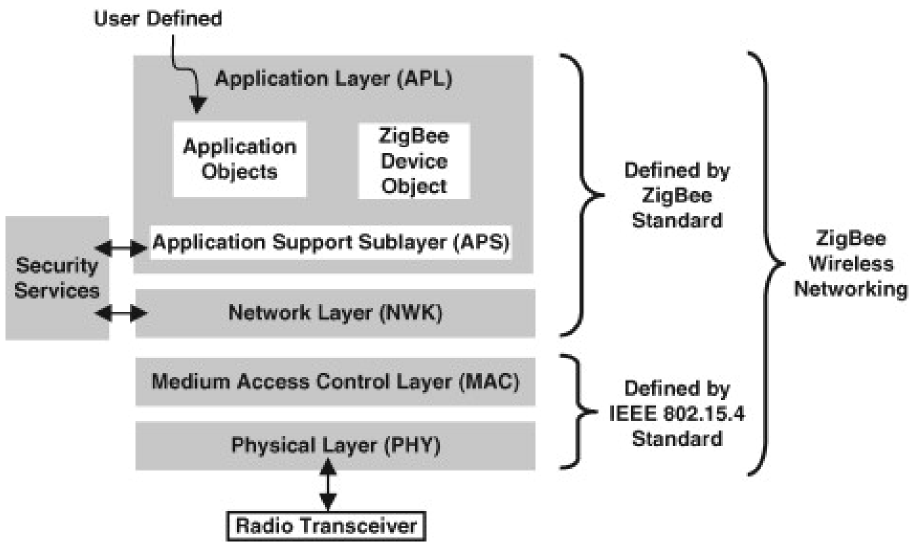

User Defined
Application Layer (APL)
Application Objects
ZigBee Device Object
Application Support Sublayer (APS)
Security Services
Network Layer (NWK)
Medium Access Control Layer (MAC)
Physical Layer (PHY)
Radio Transceiver
Defined by ZigBee Standard
Defined by IEEE 802.15.4 Standard
ZigBee Wireless Networking

Figure 1.3. ZigBee Wireless Networking Protocol Layers

As shown in Figure 1.3, the bottom two networking layers are defined by the IEEE 802.15.4 standard [5]. This standard is developed by the IEEE 802 standards committee and was initially released in 2003. IEEE 802.15.4 defines the specifications for PHY and MAC layers of wireless networking, but it does not specify any requirements for higher networking layers.

The ZigBee standard defines only the networking, application, and security layers of the protocol and adopts IEEE 802.15.4 PHY and MAC layers as part of the ZigBee networking protocol. Therefore, any ZigBee-compliant device conforms to IEEE 802.15.4 as well.

IEEE 802.15.4 was developed independently of the ZigBee standard, and it is possible to build short-range wireless networking based solely on IEEE 802.15.4 and not implement ZigBee-specific layers. In this case, the users develop their own networking/application layer protocol on top of IEEE 802.15.4 PHY and MAC (see Figure 1.4). These custom networking/application layers are normally simpler than the ZigBee protocol layers and are targeted for specific applications.

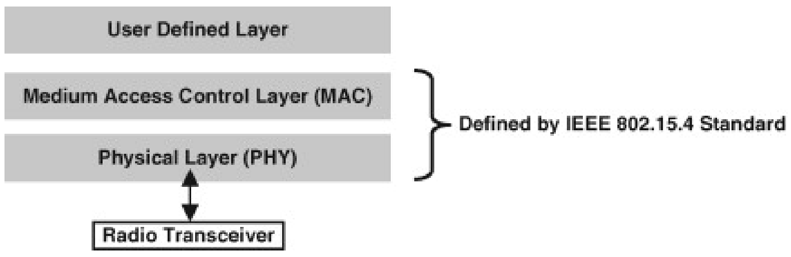

User Defined Layer
Medium Access Control Layer (MAC)
Physical Layer (PHY)
Defined by IEEE 802.15.4 Standard
Radio Transceiver

Figure 1.4. A Networking Protocol can be Based on IEEE 802.15.4 and not Conform to the ZigBee Standard

One advantage of custom proprietary networking/application layers is the smaller size memory footprint required to implement the entire protocol, which can result in a reduction in cost. However, implementing the full ZigBee protocol ensures interoperability with other vendors' wireless solutions and additional reliability due to the mesh networking capability supported in ZigBee. The decision of whether or not to implement the entire ZigBee protocol or just IEEE 802.15.4 PHY and MAC layers depends on the application and the long-term plan for the product.

Physical-level characteristics of the network are determined by the PHY layer specification; therefore, parameters such as frequencies of operation, data rate, receiver sensitivity requirements, and device types are specified in the IEEE 802.15.4 standard.

This book covers the IEEE 802.15.4 standard layers and the ZigBee-specific layers with the same level of detail. The examples given throughout this book are generally referred to as ZigBee wireless networking examples; however, most of the discussions are still applicable even if only IEEE 802.15.4 PHY and MAC layers are implemented.

> Read full chapter

Hello ZigBee

In Zigbee Wireless Networking, 2008

Publisher Summary

This chapter provides information on how to apply ZigBee appropriately to achieve wireless control that simply works. The wireless control market has a number of unique needs for which ZigBee is ideally suited, because ZigBee is highly reliable, cost-effective, able to achieve very low power, highly secure, and an open global standard. Low data rate of Zigbee adds a constraint in achieving the low power and low cost criteria. ZigBee is all about wireless monitoring and control and is a standard networking protocol aimed at the wireless control market. The ZigBee protocol fits on 8-bit microcontrollers, with 16- and 32-bit solutions available. It is great at wireless control; where anywhere from two to thousands of nodes are all connected together, in a multi-hop mesh network. It enhances reliability through mesh networking, acknowledgments and use of the robust IEEE 802.15.4 standard. Multiple silicon and stack vendors, ZigBee modules, and many available resources all contribute to low development costs for ZigBee devices. It uses AES 128-bit security for encryption and authentication. ZigBee Alliance membership is required in order to ship ZigBee technology in products and it provides early access to specifications. ZigBee covers many markets, including home, commercial and industrial automation, medical, and location-based services. ZigBee networks can be put together in a very ad hoc (random) fashion, and they simply work. ZigBee can communicate to individual nodes or groups and these devices remember their settings across resets and power outages.

> Read full chapter

Developing Embedded Platforms for Ambient Assisted Living

Cristian-Győző Haba Ph.D. Professor, ... Valeriu David Ph.D. Professor, in Ambient Assisted Living and Enhanced Living Environments, 2017

9.5.3 Fixed and Mobile Nodes

In Figure 9.7 we present the architecture of a System for Developing Medical Applications (SDMA) that was used to develop and implement the fixed and mobile nodes of a health monitoring system (Breniuc et al., in press).

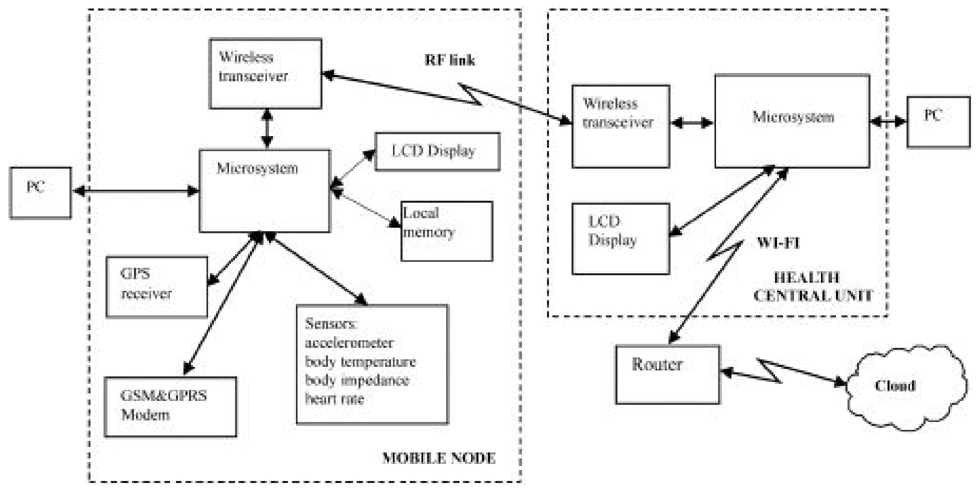


Figure 9.7. The SDMA block diagram including a Mobile Node.

A SDMA includes at least two modules: Health Central Unit (HCU) and a Fixed Node (FN) or a Mobile Node (MN). The Health Central Unit and the Fixed Nodes are placed in the patient surveillance and monitoring area while the Mobile Nodes are usually worn or carried by the patient. The modules communicate using RF links which determines also the longest distance between the modules. Health Central Unit controls the operation of the FN and MN via the RF link and also uses this link for data transfer. In our prototype the RF communication was implemented using the ZigBee protocol. This protocol allows connection of HCU or FN and MN to other existing complex systems which support ZigBee protocol. An example is the ability to establish a connection between the FNs or MNs with the Home Automation System described in paragraph 9.5.1.

Data are processed by the Health Central Unit and can also be sent for storage and display to a PC. Data considered important is displayed on a local display. A greater versatility of the Health Central Unit is obtained using a microsystem that can connect using a wired or wireless connection to the Internet and thus access cloud services. Collected health data can be sent to be stored in patient record where it can be also accessed by authorized persons (doctors, caregivers, family members etc.).

The FN or MN has a flexible and reconfigurable structure so it can cover a great number of use cases. The number and type of sensors available on the FN or MN are established based on the number and type of physiological signals measured for the monitored patient. For example, in the case of free fall detection, three axis accelerometers are used. Temperature is measured with appropriate temperature sensors, respiration is measured using thoracic impedance while for heart rate are used ECG modules. Normally, the operation of the FNs and MNs is controlled by the HCU but in certain applications the FN or the MN can work also as standalone system where all control and processing is performed locally in the node. For the standalone operation the FN or MN should include the following components:

- local display for displaying important data collected from patient;
- local memory for temporary storage of data to be transferred to a PC or over the Internet for further processing;
- GPS receiver for patient localization;
- wireless communication link for data exchange but also for sending warning messages in critical situations. By critical situation we understand the case when the patient's health or life is in danger (ex. freefall, increased heart rate, abnormal body temperature etc.).

For maintenance purposes, debugging or software updating a serial connection to a PC is desirable.

Component selection for building the SDMA must follow the same rules used for selecting the modules for the HA system. In order to reduce costs, maintain compatibility and reuse code, for the SDMA prototype we have used modules manufactured by Digilent Inc. similar to those used in prototyping the HA. Some modules unavailable from Digilent or other manufacturers were designed and prototyped by authors with a special care to maintain compatibility with other components of the system (Pmod or Arduino connectors, voltage levels etc.).

As wireless transceiver we have used PmodRF1 module but we also made tests with the PmodRF2 module (based on Microchip MRF24J40 transceiver, IEEE 802.15.4 certified, ISM band 2.405–2.48 GHz operation, 250 kbit/s or 625 kbit/s transfer rate, about 300 m maximum distance communicating range, simple SPI communication interface). For local display, local storage, localization using GPS system and GSM&GPRS messaging we have used the same modules as for the HA implementation.

When choosing the microsystem to implement the FN or MN we must consider which are the patient's physiological measurements that must be collected and what are the already available modules that can do these measurements. In this regard, the modules to be used can be compatible with Pmods modules or can be chipKIT modules compatible with Arduino chipKIT shields.

For our prototypes we have used both type of modules and when not available, we have built measurement modules compatible with either of the types. For example, for the Heart Rate monitoring we have used chipKIT Max32 microsystem compatible with Arduino shield ‘Mega’ form factor. The chipKIT Max32 board is based on a PIC32MX795F512L microcontroller, the same as for the chipKIT Pro Mx7. The advantage of using this form factor was the easy connection of various ECG Arduino shields available on the market.

The Health Central Unit was built using the chipKIT WF32 microsystem with the following features: Microchip PIC32MX695F512L microcontroller (80 MHz operating frequency, 512K Flash, 128K SRAM, 43 I/O pins, 12 analog inputs etc.), micro SD card connector, USB 2.0 OTG controller with A and micro-AB, Microchip MRF24WG0MA Wi-Fi module etc. These features offer an integrated local storage of data on a Micro SD card and connection of the system to a Wi-Fi router so no additional modules are needed to implement these functionalities. This type of microsystem can be used also for building more complex FN or MN that need more memory and computing resources.

Similar to the HA system, the supply of Health Central Unit, FNs and MNs can be done via the microsystem USB interface or from batteries.

The application running on the Health Central Unit was written and tested using the MPIDE environment with all the advantages deriving from the use of an Arduino programming environment.

In Breniuc et al. (in press) is presented a Mobile Node prototype configured for detection of free fall in elderly people. FreeFall detection was implemented using a PmodACL module based on an Analog Devices ADXL345 3-axis accelerometer. The resolution of the accelerometer (specialized for Free Fall detection) is programmable in the range , and can be controlled through either an SPI or an I2C interface. The Mobile Node is built using the chipKIT MX3 microsystem and PmodRF1 transceiver. In Figure 9.8 is presented the prototype for the Mobile Node that detects freefall.

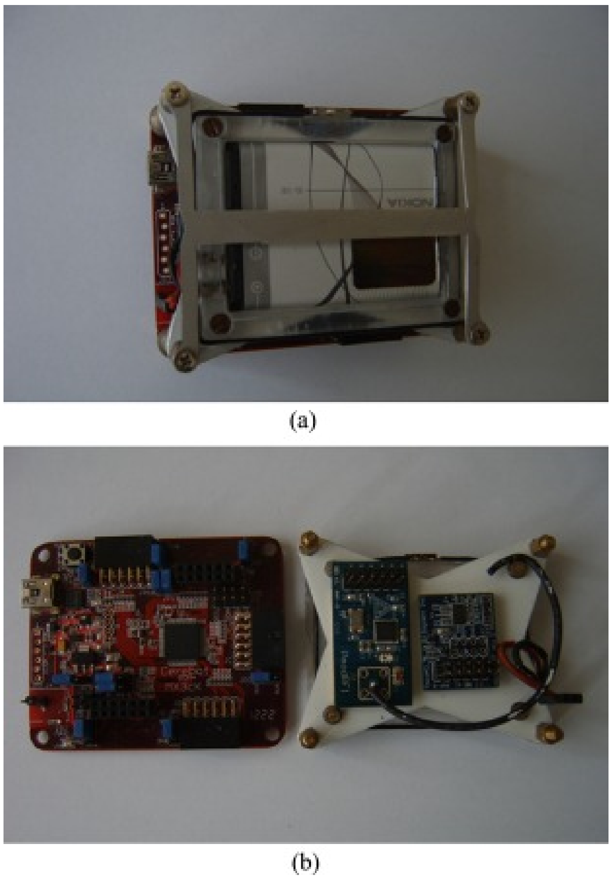


Figure 9.8. Mobile node prototype for freefall detection a) packed and b) unpacked.

In Breniuc et al. (2014) is presented a system for impedance measurements based on an Analog Devices AD5933 circuit (Figure 9.9). The system uses a chipKIT MX4 microsystem, a PmodOLED2 display, a PmodSD SD card interface and a PmodENC encoder for function selection. The system may be used for respiration measurements.

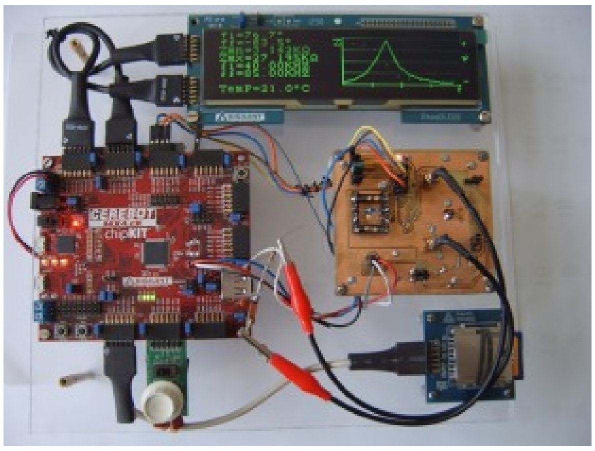


Figure 9.9. Prototype of the impedance meter used for respiration measurements.

A similar system was prototyped for measurements of heart rate as well as for body temperature and humidity. The Mobile Node included an Olimex ECG/EMG module for heart rate measurement and a Sensirion SHT21 sensor for temperature and humidity measurement. The module is built around the chipKIT WF32 microsystem which can connect directly to a wireless router. A PmodOLED display is used to show data locally, RF communication is done using the PmodRF1 transceiver and function selection is performed using a PmodENC encoder. The prototype of the mobile node designed for heart rate measurement is presented in Figure 9.10.

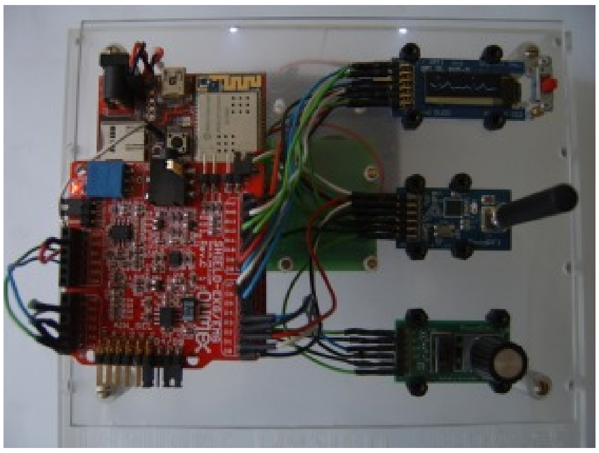


Figure 9.10. Mobile node prototype for heart rate measurement.

> Read full chapter

Deciding on ZigBee

In Zigbee Wireless Networking, 2008

2.1.2 Other Wireless Technologies

802.15.4 is a great MAC and PHY specification, and ZigBee, which relies on the 802.15.4 MAC and PHY, is a great networking protocol. ZigBee is well-targeted at the wireless sensor and control network space. But 802.15.4 is not the only radio which targets this space, and ZigBee is not the only networking protocol which runs on the 802.15 MAC and PHY:

- *Non-ZigBee 802.15.4 radios* are not supported. One thing that is not obvious to the casual observer is the fact that ZigBee runs only on 2.4 GHz (ISM band) 802.15.4 radios. The 802.15.4 specification allows for 900 MHz, and 868 MHz radios as well, but ZigBee doesn't run on those radios, due mainly to data rate. ZigBee requires a higher data rate than the 40 kbps offered by the 900 MHz standard and 20 kbps offered by the 868 MHz standard. The 2.4 GHz 802.15.4 specification operates at 250 kbps, a rate sufficient for the ZigBee protocol. The newer 2006 802.15.4 specification offers higher data rates in the sub-1 GHz RF spectrum, but those have not yet been adopted by ZigBee. The other reason ZigBee chose the 2.4 GHz spectrum is that products can interoperate worldwide in this RF space.
- *Wireless USB* is emerging as a great technology for small, battery-operated devices. For example, the Cypress CYWUSB6934 is used in a number of point-to-point devices such as a wireless mouse. This technology is very inexpensive, great on batteries, but is not meant to function on a scale as large as ZigBee can, and doesn't have the security. Wireless USB is based on Ultra Wideband (UWB) which supports 480 Mbps data rate over a distance of two meters (about 6.6 feet). If the speed is lowered to 110 Mbps, UWB will go a longer distance (up to 10 meters or about 30 feet). This technology generally focuses on the PC peripherals, but could be adapted to the sensor and control space.
- *WiFi™* is also starting to see more use in sensor and control networks. WiFi is a ubiquitous, well understood technology that previously targeted the PC and laptop space. It's a bit more expensive than ZigBee, typically requiring a larger CPU to run the full protocol. WiFi can mesh like ZigBee and if you Google “WiFi mesh” you'll see a number of vendors that serve this space, but there are no interoperable standards for meshing WiFi. WiFi is not as good with batteries, using up batteries in hours, rather than the months to years of ZigBee, but not all control networks require batteries.
- Bluetooth™ is very secure, and is used in headsets and cell phones everywhere. Bluetooth is relatively inexpensive, but due to its frequency-hopping technology, does not have the battery life of a ZigBee device. Think of Bluetooth

- battery life as days, not the months to years like ZigBee. Also, Bluetooth doesn't scale well and is typically limited to networks of seven devices or less, whereas ZigBee networks can contain thousands of devices.
- *Wibree* falls under the Bluetooth Special Interest Group, and is aimed at watches and body sensors. It's designed to be very, very low power, but again is limited in the number of nodes in a network. At the time of this writing, Wibree was not in production, but still in the specification stage.
- *Z-Wave,* by Zensys (see http://www.zen-sys.com), is useful if you are planning home automation products. Z-Wave is aimed specifically at the home automation space, and while not a standard (Zensys is the only manufacturer of Z-Wave), it is used by some of the large home automation manufacturers, including Danfoss, Leviton, and Universal Electronics. The new ZW0301 contains a 32 K flash 8051 CPU, and a radio that operates in the 868 MHz space for Europe and the 908 MHz space for the U.S., at 9 kbps (compared to ZigBee's 250 kbps).Zensys is naturally worried about ZigBee, because these two technologies compete directly, and so Zensys has formed a group called the Z-Wave Alliance, and regularly releases white papers promoting Z-Wave over ZigBee.
- *Proprietary radios* are still the norm today. I remember this used to be the case in operating systems, too. Everyone had their own system, but while there are still quite a number of proprietary operating systems, most OEMs realize it is cheaper in most cases (all things considered) to go with a commercial operating system. I believe this will happen to radios over time in this space.There are so many good radios in various RF frequencies. Texas Instruments makes one called the Dolphin. Nordic Semiconductor makes a variety of low power radios, such as the nRF2401A. The biggest issue with the proprietary radios is the lack of a protocol. So much of RF communication is in the software. If you have to write your own, it can cost significant amounts of development effort, which equals time and money.Which brings me to software protocols.

ZigBee operates in the worldwide 2.4 GHz ISM band.

Proprietary radios are the primary competitor of ZigBee today.

> Read full chapter

Sensor Materials, Technologies and Applications

M. Niedermayer, in Comprehensive Materials Processing, 2014

13.06.4.4 Smart Sensor Systems for Condition Monitoring

After discussion of the three different approaches of cost reduction, the concrete design options of cost-optimized sensor networks for the condition monitoring in paper mills are now to be analyzed. The selected scenario belongs to the industrial applications of the machine diagnoses, which in the future will gain considerable importance for the remote maintenance of machines. This allows technicians to predict the aging of certain wearing parts in advance and to determine the appropriate moment for replacement of the related machine components. As a result, the cost of maintenance at regular intervals can be reduced significantly without accepting additional machine downtimes. Depending on the planned fabrication quantities, the various design options were assessed to identify a tailored solution that fully exploits the specific cost reduction potential.

The self-sufficient sensor nodes for condition monitoring of machines should measure accelerations on motor shafts and equipment enclosures to trigger an inspection in case of abnormal vibrations. Users of such structural health monitoring have agreed on the following conditions:

- Network size: 50–200
- Acceleration range: 0–10-fold acceleration
- Temperature range: –40 to 125 °C
- Coverage of radio communication: 5–30 m
- Measurement intervals: every 5 s
- Communication intervals: every 60 s
- Operating time: >1 year

Ideally, the wireless sensor nodes, which are designed to be mounted on rotating axles and other moving parts, should not affect the mechanical dynamics. Since the corresponding miniaturization of the smart sensors is limited for an envisioned operation interval of several years, the upper bound of volume and weight had been set to 1 cubic inch and 10 g. In general, the system architecture should include two microchips for data acquisition and wireless communication as well as an MEMS accelerometer. The sensor for temperature measurements is usually integrated on the chip for the sensor interface. For a broad scope of applications, the radio chip should be able to operate with a radio frequency of 2.4 GHz and allow for an energy-efficient wireless communication with data rates in the kilobaud range.

The chip for the data acquisition and flow control should be equipped with a microcontroller core that includes internal data and program memory. This requirement came from the expected fabrication quantities, as development of special hardware components for the measurement algorithms and wireless protocols has been proved too costly. This allows for remote adaptation of parameters with moderate effort.

While powerful 32 bit microcontrollers are used for accurate machine diagnoses, more energy-efficient 8 or 16 bit microcontrollers were sufficient since only large abnormal vibrations should be detected here. The memory requirements depended on the extent of data preprocessing and the features of the radio protocol. For a minimalistic functionality, 2 kB of data storage and 10 kB for storing the program instructions were adequate, whenever other assistance nodes can support the network control. The use of the ZigBee protocol standard in conjunction with a spectral analysis, however, would require 8 and 128 kB of data and program memory, respectively. Analogous to the cost analysis of universal smart microsystems (see also 13.06.4.2), the choice fell on the Mica-Z™ architecture (Figure 29).

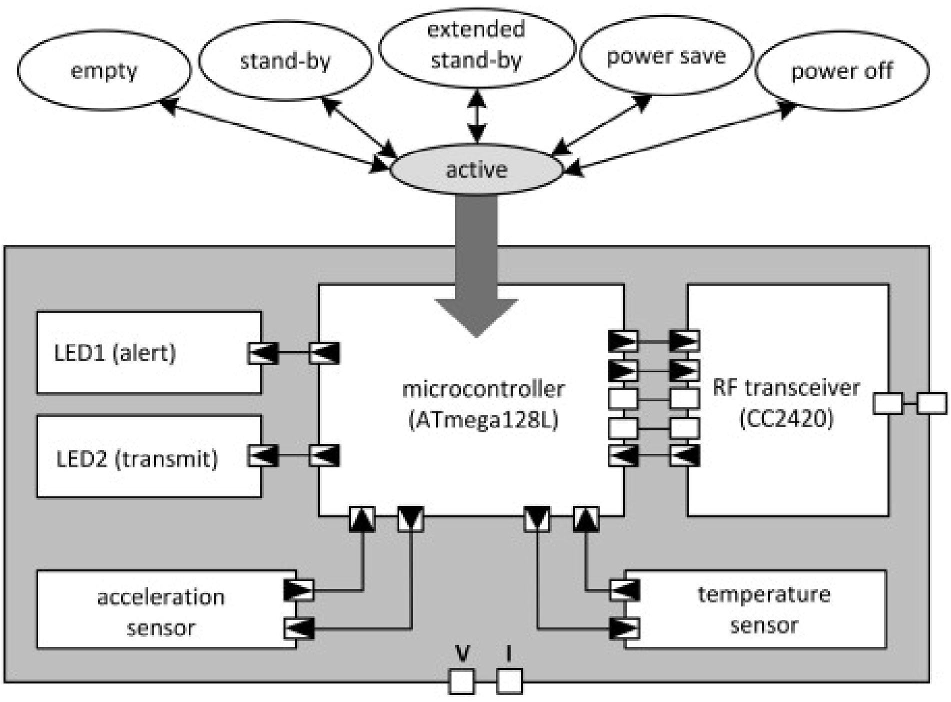


Figure 29. State chart for analyses of power consumption.

The characterization included functional units for the microcontroller core, memory, and system clock units and the essential periphery components such as timer, analog-digital converter, and SPI interface. The resultant operation time and battery size have been determined in specific modifications of application scenario (Figure 30). The allowed-for assessment on basis of these characteristics, which measures of synchronization and error correction increase or decrease the size of the energy buffer. A lower average power consumption was not always the decisive factor. Indeed, a balanced load profile resulted in a smaller system volume. An additional crystal oscillator for a precise real-time clock led to a reduced synchronization overhead so that the additional space required for the encapsulated, hermetically sealed quartz crystal was justified by smaller energy storage.

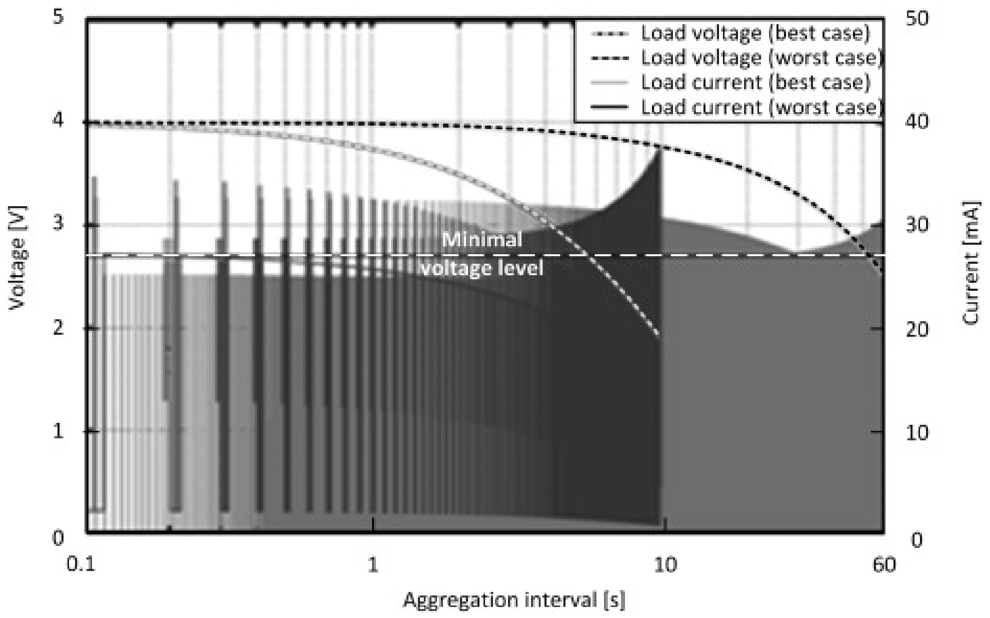
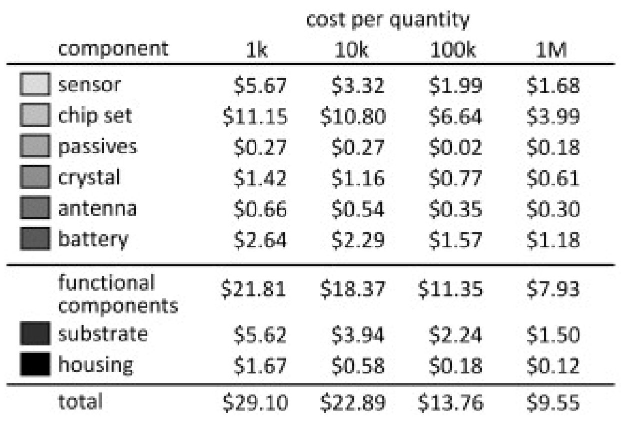


Figure 30. Exemplary load profile for prediction of the operation time.

A cost analysis of the smart sensor system revealed the following cost aggregation list according to Figure 31. The dominant cost share of about 40% was caused by the chipset consisting of the microcontroller and the radio transceiver. To reduce the chip costs, a microcontroller with integrated radio transceiver could be chosen as a single-chip solution. This measure was rejected because the existing microchips at the time of analysis would have led to significant compromises regarding desired functionality.



	cost per quantity			
component	1k	10k	100k	1M
sensor	$5.67	$3.32	$1.99	$1.68
chip set	$11.15	$10.80	$6.64	$3.99
passives	$0.27	$0.27	$0.02	$0.18
crystal	$1.42	$1.16	$0.77	$0.61
antenna	$0.66	$0.54	$0.35	$0.30
battery	$2.64	$2.29	$1.57	$1.18
functional components	$21.81	$18.37	$11.35	$7.93
substrate	$5.62	$3.94	$2.24	$1.50
housing	$1.67	$0.58	$0.18	$0.12
total	$29.10	$22.89	$13.76	$9.55

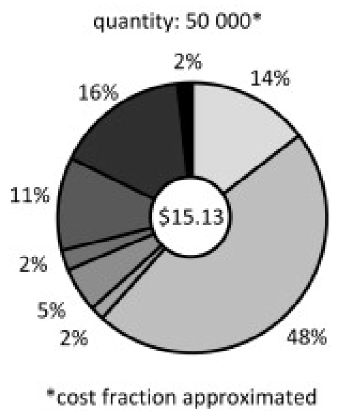


Figure 31. Overview of component costs.

The cost for the MEMS accelerometer of just under 20% can be roughly halved if an analog sensor is applied instead of the digital version. No further costs are required as functional units for analog-digital conversion on the chosen microcontroller can be used. This potential cost saving, however, was not exploited as this measure significantly lowers energy efficiency and processing speed.

The low cost share of discrete passive components allowed for the conclusion that an integration of these components into substrate layers was not justified, because the cost increase of the substrate was much higher than the potential savings. Nevertheless, the antenna cost can be decreased by implementing the antenna on the module substrate. Alternatively, a thin copper wire can also be used. Yet, a commercial antenna was chosen here because it ensured a sufficient antenna efficiency to limit the power consumption during the radio communication.

A lithium primary cell was used for the power supply. Cheaper alkaline batteries with comparable energy capacity lead to significantly larger and heavier wireless sensor nodes, which were not preferred by the user. The option of smaller and cheaper energy storages in combination with vibration transducers or thermoelectric generators was not practical because the cost of additional components can lead to a much more expensive power supply.

While the costs for all functional components were based on commercially available standard components, only the module substrates and the housing were manufactured here as application-specific components. The share of direct costs for these two categories would decrease from 25 to 17% if greater production volumes are planned. This relationship is exacerbated if the corresponding expenditures for the elaboration of production documents are assigned immediately. The production quantity was set to 100 000 units for the actual planning. Depending on marketing success, it can be expected that the real module quantity will range between 10 000 and 500 000.

With the help of the individual cost models from Section 13.06.4.1, cost estimates have been calculated for the various technological realization variants. Since the decision on a specific manufacturing technology was coupled strongly with the expected fabrication quantity, the design decisions were based on subjective expectations. A separate estimation of direct and indirect costs regarding the different assembly variant was very useful for that reason. Depending on the planned production quantity, the optimum of such a radio sensor system for machine diagnostics ranges between \$8 and \$72 if one always chooses the most cost-effective fabrication technology. Thus, a shift of the cost optimum is immediately apparent whenever the number of modules to be produced changes.

A conventional approach would have led to a preferred use of standard technologies by mounting a microcontroller chip, a radio transceiver chip, and an acceleration sensor in individual component packages on a PWB. The proposed model-based design allows for cost savings of 18% if the most cost-efficient implementation variant is identified. A design study is shown in Figure 32(a). The compact module design is based on an SBU substrate of 0.2 square inch. While the radio transceiver chip is provided in an MLF package, the microcontroller should be integrated with the acceleration sensor together into one multicomponent package.

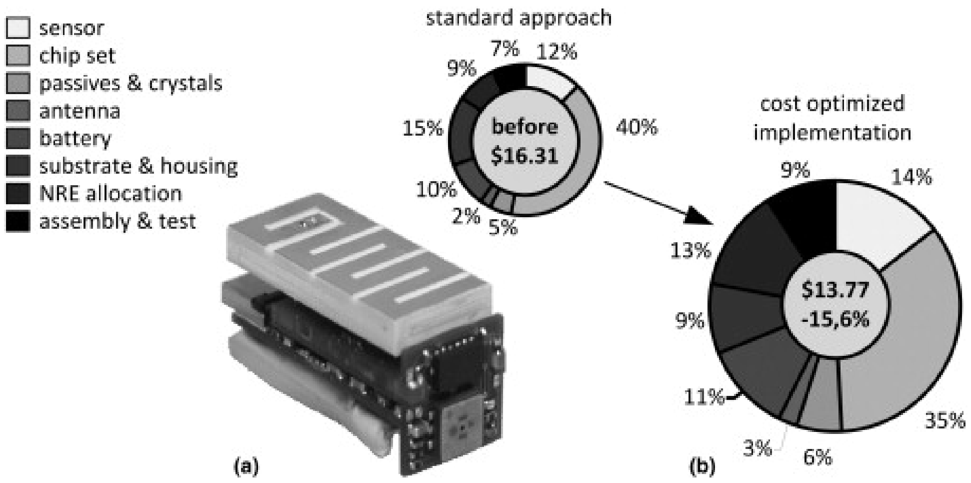


Figure 32. Design study of the sensor node for machine diagnoses (a) and resultant cost changes (b).

The cost benefits arise from a much larger yield per production panel. The joint packaging of the acceleration sensor and microcontroller provides further cost savings regarding the module assembly in combination with an improved test interface. Thus, calibration costs can decline. The achieved cost reduction from $16.31 to $13.77 (Figure 32(b)) can be exploited as an additional margin to utilize the advantages of technology leadership. Alternatively, it should be considered to follow a strategy of price leadership by reducing the unit prices to increase the fabrication quantity whenever a high price elasticity of the market can be expected.

The cost modeling supported primarily the selection of technology variations. Nevertheless, the refined cost models also allowed an improved assessment of further measures to reduce costs. The availability of the various cost elements helped to identify cost drivers and to determine strategies for further cost reductions by means of expert interviews. For the scenario considered here, a list of optimization proposals was obtained in accordance with Table 9. After discussion of each measure, the cost models were refined to predict the resulting cost savings.

Table 9. Review of other potential cost-saving measures

Exemplary development tasks for cost reduction	**Savings per device**	
#1 Optimization of energy efficiency and battery	$0.98	7.1%
#2 Implementation of a network calibration	$0.47	3.4%
#3 Preparation of machine code for storage in a ROM unit	$0.32	2.3%
#4 Optimization of the radio unit including antenna	$0.22	1.6%
#5 Reduction of drift effects by sensor packaging	$0.17	1.2%

It should be noted that an implementation of all proposals would not result in the sum of the cost benefits. Some of the proposals influence the cost effects of other measures. Thus, the net effect of combined calibration measures on software, circuit, and production level is difficult to predict even by experts, as much interdisciplinary knowledge is necessary to overlook these dependencies. Therefore it is advisable not to pursue the full diversity of the individual proposals, but merely to implement a series of particularly effective cost-reduction measures.

During the assessment of various application examples, the introduced cost modeling has proven to be a powerful tool for cost assessment and optimization. Compared to the conventional approach without a cost modeling, the particularly cost-effective implementation variants of smart microsystems can be identified much more quickly by the proposed model-based design methodology.

> Read full chapter

ZigBee Applications

In Zigbee Wireless Networking, 2008

4.3.1 Channels

For those of you who haven't heard it before, the 2.4 GHz band is a worldwide unlicensed portion of the RF spectrum for use by many radio products. This band is also known as the ISM (Industrial, Scientific, and Medical) band. WiFi™ exists in this band. So does Bluetooth™. So do cordless phones and microwave ovens. And so does ZigBee.

ZigBee uses the same channel set as specified in 802.15.4. In the 2.4 GHz band, these channels are numbered 11 through 26. Channel numbers 0 through 10 are defined by the sub-1 GHz 802.15.4 radios, but ZigBee (at least to date), doesn't run on the sub-1 GHz radios.

Note that if you use the Freescale SMAC networking stack (which is not ZigBee, but does use the same Freescale 802.15.4 radios), the channels are numbered 0 through 15. You can translate these numbers into MAC/ZigBee channels 11 through 26.

Channels are really just a portion of the RF spectrum. In the case of 802.15.4, each of these channels is separated by 5 MHz in the 2.4 GHz band, as shown in Figure 4.11. 802.15.4 uses Direct Sequence Spread Spectrum (DSSS) to spread the packets into symbols and reassemble them on the other end, verifying that the data was decoded correctly through use of a 16-bit CRC. In Chapter 7, “The ZigBee Networking Layer,” I describe the 802.15.4 PHY in a little more detail, but other books describe this PHY standard better than I do. This book is about the ZigBee protocol. If you'd like to learn more about 802.15.4, try IEEE 802.15.4 Low-Rate Wireless Personal Area Networks: Enabling Wireless Sensor Networks, by José A. Gutierrez.

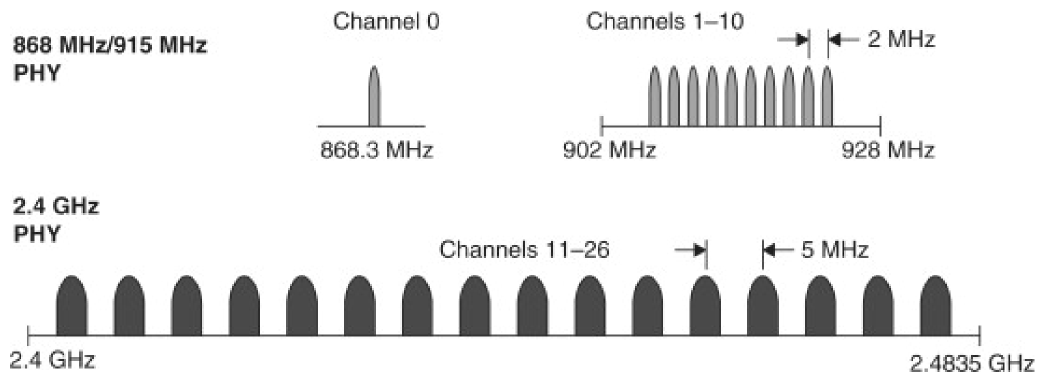


Figure 4.11. IEEE 802.15.4 Channels

The 802.15.4 radio forms the foundation for ZigBee. Two interesting points about this radio is that it is half-duplex (it can listen or talk, but not both at the same time) and accesses one channel at a time. So, a device listening on channel 15 won't hear anything on channels 11 through 14 or 16 through 26.

ZigBee as a protocol does not typically change channels. Bluetooth, like some other wireless technologies, is a channel-hopping protocol and some believe that channel-hopping is required for reliability. Not so. Due to the robust nature of O-QPSK and DSSS, 802.15.4 radios are very robust, even in a noisy RF environment. In fact, tests performed at the ZigBee Alliance with WiFi turned up to maximum on all channels, did not cause ZigBee to lose even one packet, over a series of thousands of data requests. Occasionally there were some retries, but not a single packet was dropped.

So how does a ZigBee device decide what channel to use? It is defined by the Application Profile. In Home Automation, for example, any ZigBee device wishing to join a network must scan all channels, and be able to join any network on any channel. If the profile is a private profile, it may choose to limit the device to one or any set of the 16 available channels.

But scanning channels does take time. When forming a network, ZigBee performs two scans:

- An energy detect scan
- An active scan

The energy detect scan is used to determine which channels are the quietest. The active scan sends out a Beacon Request, and is used to determine what other ZigBee or 802.15.4 PAN IDs are currently in use on that channel within hearing range of the radio. By default, ZigBee chooses the channel with the fewest networks, and which is the quietest (in that order).

Only the active scan (detecting other networks) is used when joining a network. If a network is already formed, and the Beacon Response can be heard by the joining node, it is assumed that the channel is quiet enough for communication.

The scan duration is defined by 802.15.4 specification, and is an integer between 0 and 14. See Table 53—MLME-SCAN.request parameters in the 802.15.4 specification for details. Since the value is used in a formula involving superframe duration, I've converted those times to milliseconds for your convenience (see Table 4.2).

Table 4.2. MAC Scan Durations

Value	**Scan Duration (ms)**	**Value**	**Scan Duration (ms)**
0	31	8	3,948
1	46	9	7,880
2	77	10	15,744
3	138	11	31,473
4	261	12	62,930
5	507	13	125,844
6	998	14	251,674
7	1,981		

In the Freescale platform, both channel selection and scan duration are fully under application control. The defaults for these are properties that can be set in BeeKit. The channels are defined by mDefaultValueOfChannel_c in ApplicationConf.h, and the scan duration by gScanDuration_c in BeeStackConfiguration.h. The defaults are channel 25 and scan duration 3. Each platform will have its own way of defining the channel set and scan duration.

In the example, *Example 4-2 ZigBee Channels*, I demonstrate scanning all the channels and then scanning a single channel for comparison. Notice how much longer it takes to scan all of the channels versus only one. (It's actually 16 to 1 in this case, since the example scans all 16 channels, then only 1 channel).

This example will use two ZigBee nodes: one to form the network (the ZigBee Coordinator), and one to join the network (a ZigBee End-Device). They happen to be a Home Automation light and switch, but they could be any application for the purposes of channel selection.

To follow the example with actual hardware, open the following two projects inside CodeWarrior:

- Chapter04\Example4-2 ZigBee Channels\NcbZcHaOnOffLight.mcp
- Chapter04\Example4-2 ZigBee Channels\SrbZedHaOnOffLight.mcp

If this process is still new to you, read Chapter 3, "The ZigBee Development Environment," which describes how to download images to the Freescale boards, step-by-step.

Load the two programs into the NCB and SRB boards, respectively. If this process is not familiar to you, go back to Chapter 3 and read about developing for the Freescale platform.

Then reset both boards. Press SW1 on both boards to form/join the network. Notice how long the running lights on the LEDs chase each other. This is the amount of time it takes to scan all 16 channels. The actual channel chosen is random (the channel is the third field, line 2 on the NCB board's LCD display), depending on how noisy each channel is. But, usually, channel 11 (the first one tried) is quiet enough to form the network on that channel.

The NCB display may look something like this:

Ha OnOffLight
Cfg 3bab 11 000

Next, reset both boards. This time, press SW4 on the NCB board to select a channel. Select channel 15. The channel number will change on the LCD display. Note also the hex pattern on the LEDs. Press SW4 on the other (SRB) board to select the same channel. Since the SRB doesn't have a display, make sure the channel pattern on the LEDs match what is on the NCB. Only then, press SW1 on both boards to form/join the network. Notice how much more quickly the network forms?

In the code, channel selection is made through a bit mask set of channels, decided initially at compile time. This channel mask can also be adjusted at run-time like it was using SW4 in the example. The bit mask uses a set of 32 bits to represent channels. Recall that the channels are numbered 11 through 26 by 802.15.4, so bits 11 through 26 are used to represent the channel mask. The comment shows how to set the bits properly, but it's much easier in BeeKit where it provides a check-box for each channel:

/*The default channel list defines which channels to scan when forming orjoin-
ing a network. Default=0x02000000=channel 25. The channel list is abitmap,
where each bit describes a channel (for example bit 12 corresponds tochannel
12). Any combination of channels can be included. ZigBee supportschan-
nels 11-26.3 2 2 2 1 1 0 0 01 8 4 0 6 2 8 4 00000 0000 0000 0000 0000
1000 0000 0000=0x00000800=channel 110000 0100 0000 0000 0000 0000
0000 0000=0x04000000=channel 260000 0010 0000 0000 0000 0000 0000
0000=0x02000000=channel 25 (default)0000 0111 1111 1111 1111 1000 0000
0000=0x07fff800=all channels 11-260000 0000 1000 0000 0001 0000 0000
0000=0x00801000=channels 23 and 12

```
*/
#ifndef mDefaultValueOfChannel_c
#define mDefaultValueOfChannel_c 0x06000800
#endif
```

If your application wishes to set this at run-time, and you are not using the common user ASL interface used in all of the Freescale applications, simply set the global variable gSelectedChannel to the desired set of channels prior to calling ZDO_Start().

The scan duration is found in BeeStackConfiguration.h and is just a number, 0 through 14. See Table 4.2, "MAC Scan Durations," for how this number translates into actual time. Scan duration is set at compile-time in the Freescale solution. ZigBee does not mandate a particular scan time for nodes:

```
/*Scan Duration*/
#ifndef gScanDuration_c
#define gScanDuration_c 3
#endif
```

Often, when defining a private profile application, I'll select channels 15, 20, 25, and 26, as these channels aren't the same as on common WiFi installations. I do this not for ZigBee's sake, but for WiFi's. If the application gets too chatty, it could cause WiFi to miss a packet or two, and degrade the performance of WiFi.

As mentioned before, ZigBee 2007 includes the ability to switch channels. Like forming or joining a network, this is designed to be a rare event for most ZigBee applications. I'll explain that feature in a bit more detail in Appendix A, "ZigBee 2007 and ZigBee Pro."

Channels are a portion of the RF spectrum.

Of the 16 channels, ZigBee selects the channel with the fewest networks by default.

> Read full chapter

Digital Twin Shop-Floor

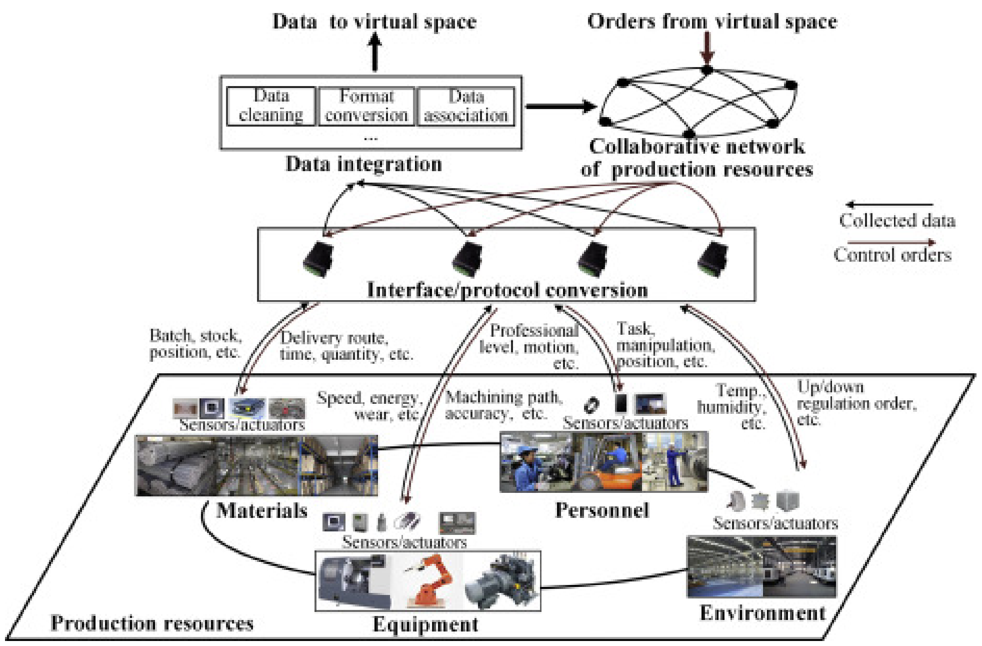
Fei Tao, ... A.Y.C. Nee, in Digital Twin Driven Smart Manufacturing, 2019

4.4 Implementation of Digital Twin Shop-Floor

Based on the concept of the DTS, the implementation method for each component of the DTS is discussed, including the implementation for the PS, VS, SSS, and SDTD, respectively. The implementation method introduced in this section has been investigated in the authors' previous work [6].

4.4.1 Physical Shop-Floor

As shown in Fig. 4.5 [6], except for the traditional production abilities, the PS in DTS is also able to realize interconnection and interaction. It means that from the vertical perspective, the production resources can convey data to the virtual space and receive control orders from it. While horizontally, they can sense other resources and regulate the behaviors of themselves or others.



Data to virtual space
Orders from virtual space
Data cleaning
Format conversion
Data association
...
Data integration
Collaborative network of production resources
Collected data
Control orders
Interface/protocol conversion
Batch, stock, position, etc.
Delivery route, time, quantity, etc.
Professional level, motion, etc.
Task, manipulation, position, etc.
Speed, energy, wear, etc.
Machining path, accuracy, etc.
Temp., humidity, etc.
Up/down regulation order, etc.
Sensors/actuators
Sensors/actuators
Materials
Personnel
Sensors/actuators
Sensors/actuators
Environment
Production resources
Equipment

Figure 4.5. Interconnection and interaction in PS.

Data of the production resources should be retrieved in real-time. Considering the heterogeneity, different sensors are applied to data acquisition. For the personnel, portable devices can be used to collect physical power, motion, and work progress. Wired or wireless sensors can be deployed on equipment to collect the states such as machine speed, tool wear, and axis vibration. For equipment with a high degree of automation, these data can be read from embedded modules of its own. RFID can be used for lifecycle tracking of materials, and environmental sensors are applied to detecting real-time changes.

However, these data are usually transmitted with different interfaces (e.g., RS232, CAN, and ZigBee) and communication protocols (e.g., Profibus, TCP/IP, and Modbus), which makes it difficult to implement unified data access to the virtual space. Hence, it is necessary to deploy customized access modules, through which the data from different sources can be transformed into a uniformed interface and protocol. In addition, since the accessed data always have different formats, types, and information models, data integration including cleaning, format conversion, association, etc. should be carried out.

Based on the integrated data, a collaborative network that describes the production resources and their relations is built. In the network, the resources are represented by nodes that have the abilities to sense, compute, and interact, while their relations are expressed by edges. According to the orders from the virtual space, a node asks others to cooperate with it and also responds to requests propagated by others, which gives the PS stronger adaptability, flexibility, and robustness.

Orders from the collaborative network are transmitted to the customized access modules for interface and protocol conversion to adapt to different communication modes of actuators in the PS. Finally, the orders can be executed to control and coordinate the production.

4.4.2 Virtual Shop-Floor

As a digital mirror, to simulate the physical counterpart with high fidelity, the VS is built according to four aspects, that is, geometry, physics, behavior, and rules. As shown in Fig. 4.6 [6], the modeling process of a computer numeric control (CNC) machine is taken as an example. First, three-dimension geometry models are built to describe the shapes, sizes, positions, and assembly relations of the machine components. The common tools for these models include SolidWorks, 3D MAX, AutoCAD, CATIA, etc. Secondly, physical properties (e.g., cutting force, torque, and wear) are given to the geometry models to form the physics models, which can analyze the physical phenomena, such as deformation, cracking, and corrosion. The finite element method (FEM) can be used for simulation at this level. Then behavior models are built to describe the machine responsive mechanisms under driving factors such as NC programs and disturbing factors such as manual interferences. Finite state machines, neural networks, complex networks, etc. can be applied to describing the responsive processes. Finally, rules of associations, constraints, and deductions are modeled to describe the domain knowledge and make the above three kinds of models capable of evaluating, reasoning, and predicting. To build the rule models, data mining algorithms, such as Apriori, support vector machine, and *K*-means, can be used.

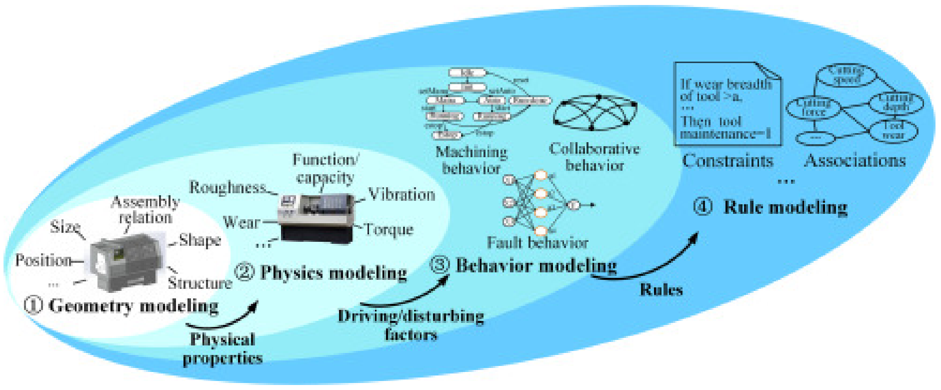


Figure 4.6. Four steps of modeling for CNC machine in VS.

After modeling, the four models are fused in both function and structure to form a complete virtual CNC machine. For other entities in the PS, the modeling processes are similar.

To ensure the accuracy of the models, verification, validation, and accreditation (VV&A) is adopted to test the transformations from models to program codes, compare the inputs and outputs of the models with entities, and estimate the models' sensitivity. In addition, with virtual reality (VR) and augmented reality (AR), the VS presents vivid three-dimension images and overlaps the virtual models on physical entities, thus providing an immersive environment for users.

As shown in Fig. 4.7 [6], model operation and model evolution are parallel processes. During operation, the models run synchronously with the physical counterparts. Calibration strategies are generated through comparing the models with entities to support model evolution, which means a higher fidelity to the PS. At the same time, the evolved models can support more accurate estimation, verification, optimization, and prediction for the operation process.

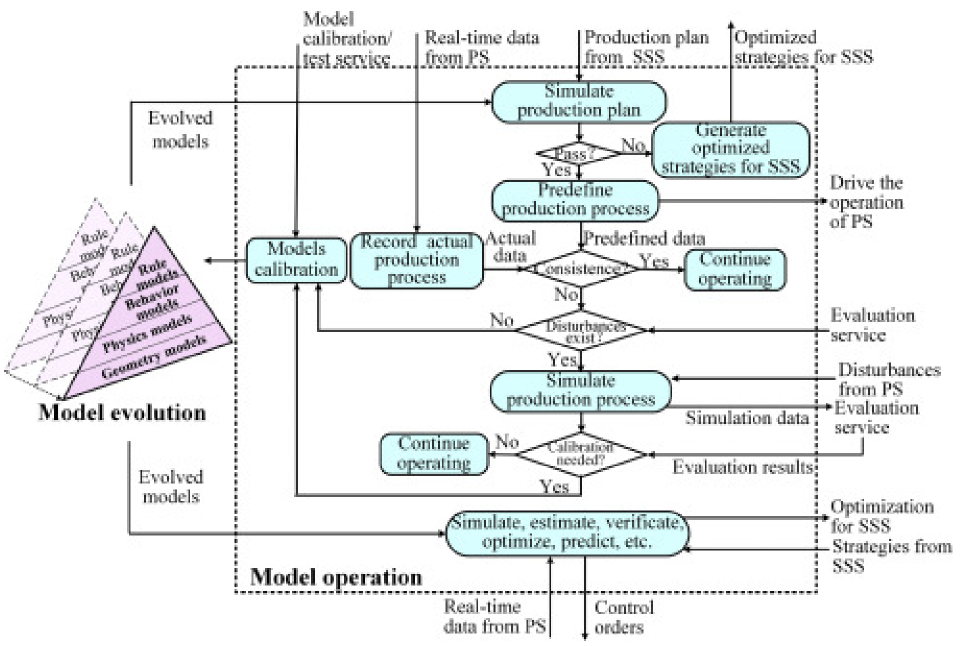


Figure 4.7. Operation and evolution of models in VS.

First, the production plan from the SSS is transmitted to the VS for verification. Based on the verified plan, the predefined virtual production process will form and drive the actual production in the PS. As the entities start to work, real-time data generated by them are recorded and compared with data from the predefined process to evaluate the consistency. If the two kinds of data are aligned, accuracy of the models can be confirmed; otherwise evaluation service will be scheduled to estimate the reason for inconsistency. In this process, if disturbances do not exist in the PS, it can be considered that the inconsistency is caused by defects of the models, and that calibrations are needed. If disturbances indeed exist in the PS, they will be captured and provided for the models. Simulation needs to be carried out again with consideration of the disturbances that were unknown to the VS previously. If the simulation results still cannot reach an agreement with the actual states, calibrations are needed; otherwise the models are considered accurate. With the models evolving to approach the entities continuously, the simulations for the PS and the SSS will be more accurate.

4.4.3 Shop-Floor Service System

The SSS provides various services to support the management and control of the PS as well as the operation and evolution of the VS. As shown in Fig. 4.8 [6], resources such as data, models, and algorithms are encapsulated into the subservices and then selected to form composite services for meeting demands from the PS or the VS. The operation mechanism is described as follows.

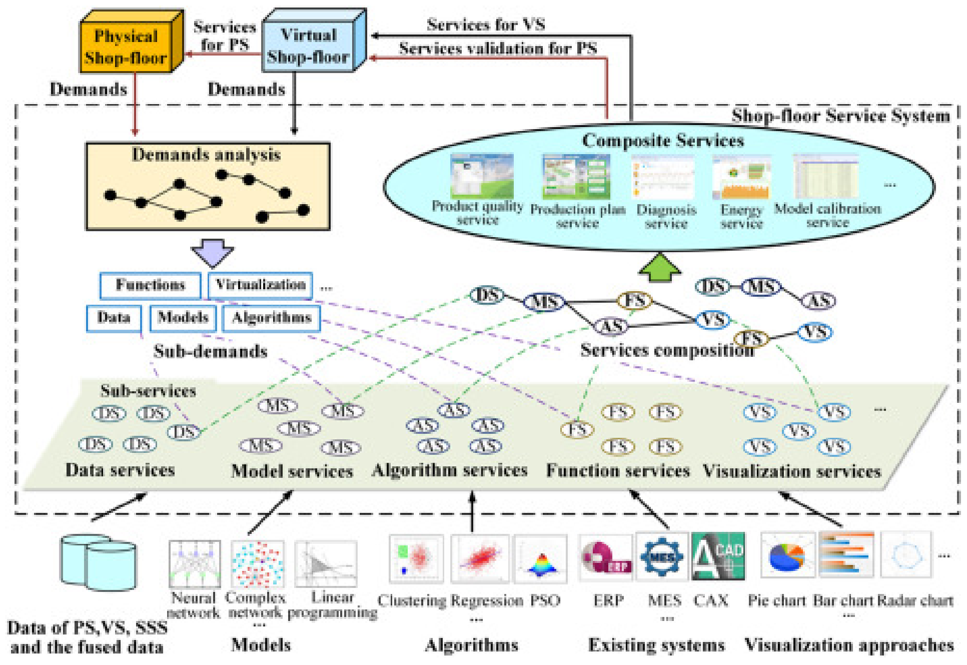


Figure 4.8. Operation mechanism of SSS.

On one side, subservices in the SSS are transformed from resources including data, models, algorithms, existing systems, visualization approaches, etc. To realize the transformations from the resources to the subservices, processes such as service description, virtualization, and encapsulation are needed. On the other side, demands from the PS mainly include production planning and scheduling, quality testing, congress control, etc., which require solving the existing problems quickly and preventing possible faults during production. Demands from the VS mainly require model calibration and test, data mining, etc., to support the model operation and evolution. These demands can be decomposed into subdemands that usually focus on problems like what data should be used, which model is most suitable, and which algorithm provides the best solution. According to the subdemands, the SSS selects suitable subservices from the candidates, and then combines them under certain rules to form the composite services for the PS or the VS under the constraints on time, cost, quality, etc. The services are monitored at runtime and will be recomposed if working improperly. For the VS, the composite services are transmitted to the models directly. For the PS, they are conveyed to the VS first for verification, and then to the PS for execution.

4.4.4 Shop-Floor Digital Twin Data

SDTD mainly consists of PS data, VS data, SSS data, and fused data of the three parts, as well as knowledge for modeling, data processing, and standards. The PS data mainly include production resource data, production process data, environment data, etc. The PS data are directly generated by entities in the physical space without further processing and considered as physical data. The VS data refer to model parameters and data of simulation, evaluation, optimization, and prediction, while the SSS data mainly involve data of various services. The latter two types of data are generated from the virtual space and defined as virtual data. The fused data are the integration and fusion of the physical and virtual data through data comparison, association, combination, and clustering. For example, for a tool, wear data from the physical space can be combined with both the simulated data about stress, deformation, and strength, and service data about maintenance records from the virtual space to form the fused data. It presents a tool correlating both the physical and virtual information, providing a more consistent, accurate, and comprehensive representation compared with the data from a single aspect.

The construction of SDTD is described as shown in Fig. 4.9 [6]. First, data from PS, VS, and SSS with various formats, types, structures, and encapsulations are transformed into a unified form. Second, the data are cleaned to remove the "dirty" data (e.g., errors, duplicate data, invalid data) and replenish the missing data through clustering, regression, and filtering. The related data can be associated and combined based on certain rules. Third, to achieve consistent and comprehensive interpretation, the physical data and the virtual data can be converged to form the fused data through data fusion algorithms, such as Kalman filter, neural network, and Bayesian inference.

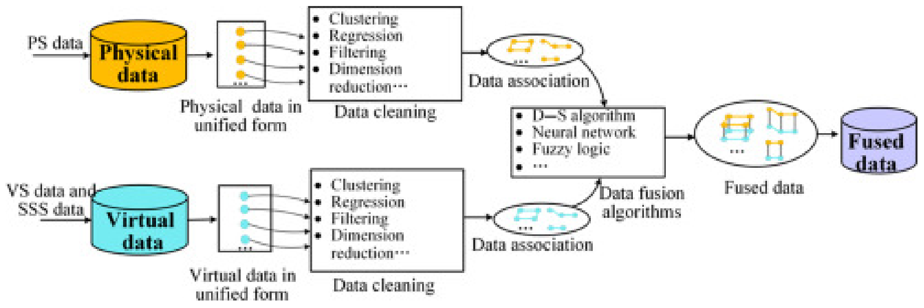


Figure 4.9. Data fusion of SDTD.

SDTD is optimized iteratively through the interaction between different kinds of data. On one side, the historical data is updated and expanded with real-time data joining, while the real-time data can be tested and corrected according to the knowledge accumulated in the historical record. On the other side, the physical data can be evaluated and simulated using the virtual data, while the latter can be compared with the former to confirm accuracy.

> Read full chapter