Marsland Press
Journal of American Science 2009:5(3) 1-7

The Design of Home Care Assistant System by the ZigBee Technology

Chien-Yuan Liu

Computer Science and Information Engineering Department, Chengshiu University,
Niausong Township, Kaohsiung County, Taiwan, 833, ROC
Email: cyliu@csu.edu.tw

Abstract

Along with a tendency towards the combination of wireless network and digital life, there has never been a better time to develop the application of wireless sensor network based on ZigBee. This paper would demonstrate the design of ZigBee application for a home care assistant system. ZigBee technology is based on the IEEE 802.15.4 standard and is extended with a network layer and an application layer to constitute a wireless network enabling architecture. During the formation of ZigBee application system, a system designer can build from a lower protocol stack, then a complex operating system, to a higher application program. In contrast with the brick-and-mud mode, it is more efficient to utilize an application framework for developing a particular application system. The work of this paper is to design a feasible home care assistant system with the application framework published by the Texas Instrument. With wireless features and automation capability built in ZigBee, the designated home care assistant system is not too hard to code during the design phase, is convenient to deploy the sensing devices, provides with operation security and low requirement for maintenance. It also relieves the burden of home care nurse and enhances the quality of caring. [Journal of American Science 2009:5(3) 1-7] (ISSN: 1545-1003)

Keywork: Home Care, IEEE 802.15.4, Wireless Sensor Network, ZigBee, TI Z-Stack

Abbreviations: HCAS: home care assistant system; WSN: wireless sensor network; API: Application Programming Interface; AF: Application Framework; PAN: Personal Area Network; PHY: physical layer; MAC: media access control layer; ISM: industrial, sciential, and medical; CCA: clear channel assessment; PPDU: PHY protocol data unit; SHR: synchronous header; PHR: PHY header; PSDU: PHY service data unit; SFD: start of frame delimiter; LLC: logic link control; GTS: guaranteed time slot; MHR: MAC header; MSDU: MAC service data unit; MFR: MAC footer; FCS: frame check sequence; CSMA/CA: carrier sense multiple access / collision avoidance; NWK: network layer; APS: application support sub-layer; APL: application layer; ZDO: ZigBee device object; DE: data entity; ME: management entity; SAP: service assess point; APO: application object; EP: endpoint; ZDP: ZigBee device profile; FFD: full function device; RFD: reduced function device; CAP: contention access period; OSAL: operating system abstraction layer; HAL: hardware abstraction layer; ADC: analog to digital convertor.

1 Introduction

Recently, because of the rapid development of the technology of micro sensor and micro electro-mechanical systems, the advancement of wireless communications, and the integration with computers, it is the exact chance for industry and academic to invest in WSN.

WSN can apply to the application domain of home care, home control, security, and position tracking, etc [1], [2], [3], [4]. The features of WSN [5] are low cost, low power consumption, low volume, easy deployment, programmable, dynamic construct, and low maintenance after installation in great scale [6].

ZigBee technology [7], [8] is one of the best solutions for WSN. ZigBee is based on IEEE 802.15.4 and is extended with network and application upper layers for simplifying the design effort of a WSN application. With proper integration of sensors, actuators, and ZigBee, it would be efficient to develop and build a wireless real-time monitor and control system.

The work of this paper attempts to utilize ZigBee integrated with bio-medical sensors to develop a HCAS. Next section introduced the IEEE 802.15.4 standard which is used as the lower layers of ZigBee. ZigBee specification was presented in section 3, including the upper layers of APL, APS, and NWK, ZDP and application profile. TI Z-Stack, containing OSAL API, ZDO API, ZDP API, AF API, APS API, NWK API, and

HAL API, was described in section 4. Section 5 explained the functions and application design of HCAS. Conclusions and future enhancement were given in section 6.

2 IEEE 802.15.4 standard

The IEEE 802.15.4 standard is specified for a PAN [9], [9]. It contains PHY and MAC layers.

2.1 PHY

The PHY of IEEE 802.15.4 defines three ISM bands located at 868 MHz, 902 MHz, and 2405 MHz, respectively. Where, the ISM band at 868 MHz provides one channel with 20 kbps bandwidth. The ISM band at 902 MHz contains 10 channels each with 40 kbps bandwidth. The ISM band at 2405 MHz consists of 16 channels each with 250 kbps bandwidth. The ISM bands at 868 MHz and 902 MHz are locally suitable in Europe and northern America. Whereas, the ISM band at 2405 MHz is globally applicable. Therefore, it is the common selection of PHY at 2405 MHz. Figure 1 is the ISM bands and channels of IEEE 802.15.4 PHY.

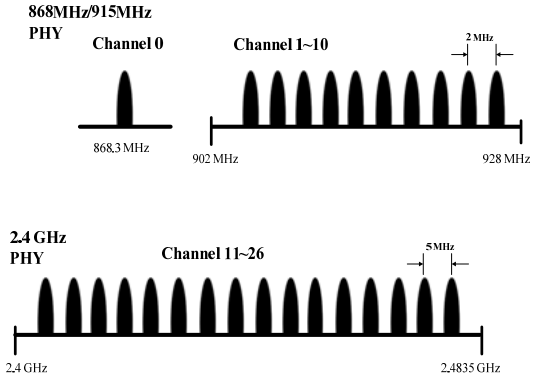

Figure 1 Bands and channels of IEEE 802.15.4 PHY

The functions of PHY specify radio transceiver opening and closing, channel selection, CCA evaluation, LQI inspection, channel energy detection, and frame signals transmission and reception, etc. The frame of PHY is named PPDU. There are 4 types of PPDU, including beacon frame, data frame, acknowledgement frame, and MAC command frame.

In Figure 2, the PPDU is composed of SHR, PHR, and PSDU. SHR is consisted of preamble bits and SFD. The length of preamble is 32 bits and the contents of it are all zero. The length of SFD is 8 bits and the contents of it are fixed as 11100101. The length of PHR is 8 bits. The first bit of PHR is 0 and the remainder 7 bits denote the length of the PSDU. PSDU contains the MAC data frame. The length of PSDU is limited within 127 bytes.

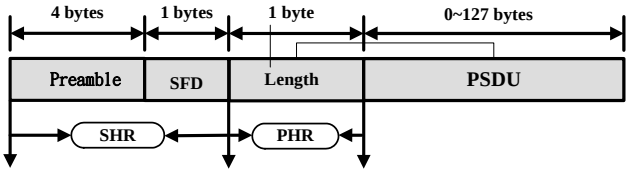

Figure 2 PPDU Frame Format

2.2 MAC

The MAC Layer is divided into two sub-layers: LLC and MAC sub-layers. LLC is commonly adopted in all series of the IEEE 802 standards. Whereas MAC is vary for different PHY. The MAC of IEEE 802.15.4 provides data service and management service to the upper layer. It is in charge of beacon management, channel access control, GTS management, frame inspection, frame transmission and response, and device association and de-association. Since the functions of MAC are not too complex to choose a high speed CPU, thus a low-power consumption and low-cost micro-controller can be adopted for the realization of the MAC protocol.

Figure 3 depicts MAC frame format. It is composed of MHR, MSDU, and MFR. MHR is consisted of 2 bytes control field, 1 byte sequence number, and at most 20 bytes address information. Control field denotes the frame type, the format of address information field, and the acknowledge mode for the frame transmission. The data sequence number is used as the frame identifier during frame communications. The replied frame should contain the same sequence number to indicate its acknowledgement to the previously transmitted frame. Thus, a reliable transmission could be achieved. The address field could contain a 64 bits IEEE address or a 16 bits network address. MFR is the FCS for inspecting whether the reception is successful or not. The MAC of IEEE 802.15.4 utilizes the CSMA/CA mechanism to control the access to wireless channel to solve the contention problem.

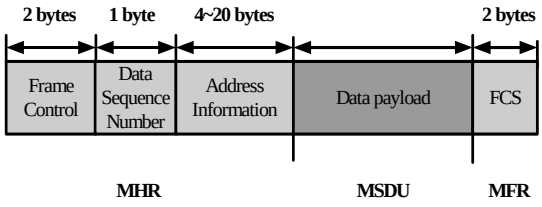

Figure 3 MAC Frame Format

3 ZigBee Specification

The draft of ZigBee specification was initiated by Honeywell, Invensys, Mitsubishi Electric, Motorola, and Philips in 2003 [11]. The latest specification was released on December 2006. ZigBee adopts IEEE 802.15.4 as its MAC and PHY. In addition, NWK, APS, APL, ZDO, and security service are added over the MAC and PHY. Therefore, ZigBee has already become a complete network protocol for wireless application. The protocol stack of ZigBee is illustrated in Figure 4.

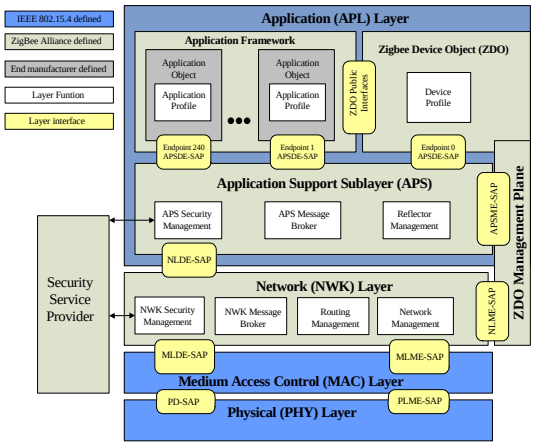

Figure 4 ZigBee Stack

Each layer in ZigBee stack is consisted of two entities: DE and ME. DE provides data services, i.e. data packet transmission and reception. ME provides management services. Normally, the lower layer is the service provider and provides its services through the interface of SAP for the upper layer. Each SAP would implement a certain functionality to support its particular service.

The kernel of ZigBee Stack is NWK. NWK takes care of joining into or leaving network, finding route, and data packet transmission or reception for a ZigBee device.

The major functions of APS in APL include the ZigBee network formation and binding, data transmission management, and network security. The function of APO in APL is to implement the processing logic for the ZigBee application. In the other words, APO is the basic unit of application program. APO can communication with a remote APO through ZigBee network to realize specific application, e.g. to turn on a remote switch for lighting. Each APO should allocate only one EP. An EP is identified by an EP ID ranged from 1~240. EP ID 0 denotes ZDO. EP ID 241~255 is reserved.

ZDP describes the behavior and properties of a ZigBee device and is mandatorily defined by the ZigBee Alliance. ZigBee application profile describes the functions and parameters for a particular application and is mandatorily defined by the ZigBee Alliance, too. Each ZigBee application profile has a unique Profile ID. Manufacturers should obey the ZDP and ZigBee application profile to design their ZigBee products. This is to assert the interoperability between different vendors. User can define a private application profile to be used in users' private application domain if the interoperation is not required. In ZigBee specification, each APO implements only one application profile. A unique EP ID should be allocated to an APO.

There are two types of ZigBee devices: FFD and RFD. FFD can act as a coordinator or router, can communicate with all types of devices, and can support any type of network topologies. RFD can act only end device and can communicate only with a coordinator. There is only one coordinator in a ZigBee network, but it allows multiple routers and end devices.

ZigBee network specifies two network topologies: central controlled star network and peer-to-peer mesh network. Start topology was shown in Figure 5(A). In star network, there are multiple devices surround a central coordinator. The central coordinator responses for the network formation and maintenance, time slot planning and allocation, channel access control, GTS management, etc. The surrounding devices follow the policy, obtained from broadcasting beacon transferred by the coordinator, to access the channel. Figure 5(B) depicts a peer-t-peer mesh network. In the network, devices are not necessary to communicate with the coordinator. But RFD is only able to communicate via FFD.

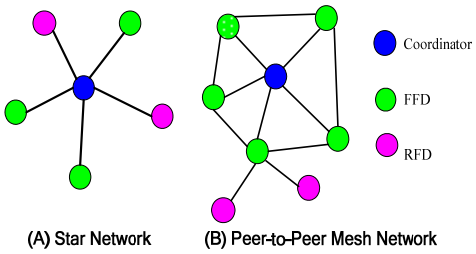

Figure 5 ZigBee Network Topology

The operation manners of ZigBee network are classified into beacon-enabled network and non beacon-enabled network. In beacon-enable network, the beacon is transmitted from a coordinator. The purpose of the beacon is to inform all devices the time synchronization information, PAN ID for devices to join, and super-frame structure. Super-frame with GTS and without GTS was shown in Figure 6 and 7, respectively. Within GTS, all time slots could only be used by certain preserved devices. The duration between beacon and GTS is named CAP and is accessed with CSMA/CA arbitration mechanism. In non beacon-enabled network, all devices merely access the channel with CSMA/CA arbitration mechanism in asynchronous way.

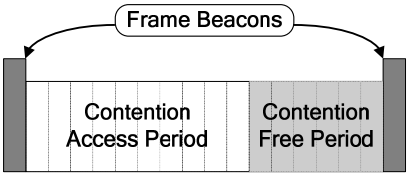

Figure 6 Super-Frame with GTS

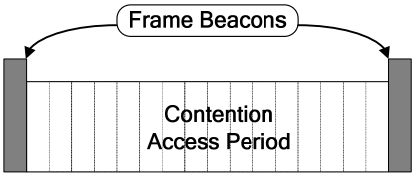

Figure 7 Super-Frame without GTS

4 TI Z-Stack

First of all to develop a ZigBee application system is to implement the ZigBee protocol stack. This is a great challenge to a developing team. Currently, some companies have developed their own ZigBee Stack as a package, e.g. Texas Instrument Z-Stack [12]. The project developing time could be obviously shortened by using TI Z-Stack to develop a ZigBee application system [13].

There are some categories of API in the TI Z-Stack, including OSAL API, ZDO API, ZDP API, AF API, APS API, NWK API, and HAL API. An application program can call these API to accomplish its communication and control requirements.

4.1 OSAL API

Through calling OSAL API [14], the program coding of APO can be independent with operating system, kernel modules, and system main control loop or hardware interrupts. This is because OSAL provides the following functions in hardware independent manner.

- Task registration, initialization, and startup.
- Messages delivery between tasks.
- Synchronous mechanism between tasks.
- Interrupt handling mechanism.
- Timers control.
- Dynamic memory management.

4.2 ZDO API

ZDO provides the management functions to a ZigBee device. An APO/EP can manage a coordinator, router, or end device through calling ZDO API [12]. The management functions contain network formation, device discovery, joining network, binding to other APO/EP, and security management.

ZDP defines the functionalities a ZigBee device should implement. ZDP specifies a device descriptor for the device and defines separate cluster by a cluster ID and a pair of EP ID and a command message. Through the command message, ZDP provides the following functions to ZDO and APO/EP.

- Start device network.
- Device discovery.
- Binding end devices, device association and disassociation.
- Network management.

The binding table is only stored at a coordinator. Therefore, only a coordinator can receive the binding request command and normally acts as a reflector to forward the messages between the binding end devices.

4.3 AF API

AF is the interface between AP and APS. APO/EP calls AF API [12] to perform wireless communication through APS and NWK. AF is also a receiving multiplexer for multiple APO/EP. AF provides the following functions to application.

- APO/EP management.
- Data packet transmission and reception

4.4 APS API

APS API [12] provides the following management functions to the upper application program.

- Management for binding table.
- Management for group table.
- Quick address search.

Besides the above mentioned functions, APS also provides data packet services. However, an application program could only use these data services via the interface of AF.

The binding table of APS is created in the static RAM. The capacity of binding table (maximum entry numbers and maximum cluster numbers per entry) can be specified by device configuration properties. Cluster denotes the relationship of a particular application for all related APO on various devices in ZigBee network. Each cluster has 8 bits cluster ID. When a message transferred with a particular cluster ID but not a destination network address, it denotes to send this message to all APO on related devices. The cluster message is sending through a coordinator (reflector). Reflector forwards the message in uni-cast format to each APO on related devices by looking at the entries of its binding table.

4.5 NWK API

NWK API [12] provides the following functions to the upper layers.

- Network management.
- Address management.
- Network status and utility functions.

In addition to the management function, NWK also provides data services. However, an application layer could not directly access data service in NWK. Instead, an application layer should call the function of AF to send data message.

4.6 HAL API

Application program can access hardware resources through HAL API [15]. These resources include timer, GPIO, UART, and ADC register. Basically, HAL API is classified into three types.

- Initialization function calls: these functions are called to initiate certain services or to set some parameters related to hardware platform. Normally, these functions are called during initialization phase after power on.
- Service access function Calls: they are also named as service functions. These functions can access directly to the content of hardware register (such as ADC register) [16], [17] or can control the action of a hardware component (such as LED on/off).
- Callback function calls: they are also named as event handler. The content of these functions should be implemented by the application programmer. These functions will be called and the status and message will be passed back to upper application task when a related hardware event occurred (such as interrupt, counting event, time-out event, or message packet arrival, etc.). Note that CPU-intensive computation or critical section should be avoided in the implementation of these functions to maintain the handling efficiency and response speed.

Generally, HAL driver provides timer, GPIO, LED, key, USRT, and ADC services to upper application. However, not all platforms provide all services to an application. Besides, devices can configure different hardware service features during the startup phase.

5 HCAS Design

The configuration of the HCAS is consisted of a personal computer, a ZigBee coordinator, and multiple ZigBee end devices.

Figure 8 shown the man-machine interface performed in the main control PC. In Figure 8, PC is connected to a coordinator through USB cable. Via the coordinator, PC transmits a sensor’s data

request to end devices, saves the received data into database, and displays the sensor's data by graphic user interface. Then, it sends control instructions to end devices according to pre-defined rules.

The sensing data measured by HCAS include body temperature, heart beat rate, and the temperature of environment. These sensors were installed on separate end devices. Firstly, PC sends a data request command to end devices to collect sensor's data. Then, the collected data are written into database. If the body temperature and the heart beat rate are over a preset threshold, HCAS would generate an alarm to a caring nurse and would send a control instruction to the corresponding end devices to turn on the ventilation equipment. After suitable treatment by the caring nurse and the cared person becomes normal, HCAS will sends a control instruction to turn off the ventilation equipment. To maintain a comfortable environment, the indoor temperature is monitored by HCAS. HCAS operates and reacts to the indoor temperature in similar manner of body temperature except to inform the caring nurse by an alarm.

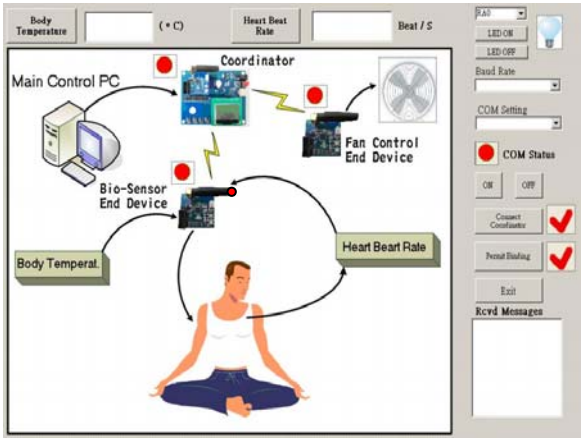

Figure 8 HCAS Man-machine Interface

The TI Z-Stack was adopted in the design of HCAS. To use TI Z-Stack, one must follow its application design framework. When an application program executed in operating system, it is an application task of OSAL. The design procedure of HCAS is explained as follows.

OSAL_HCAS_App.c module is the startup module of HCAS. In OSAL_HCAS_App.c, the application should call osalTaskAdd() function to add and register system tasks and application task into operating system. The system tasks include Hal_Init() to start and initiate HAL driver task, macTaskInit() to start and initiate IEEE 802.15.4 protocol stack, nwk_init() to start and initiate ZigBee stack, APS_Init() to start and initiate APS service and mamagement task, ZDApp_Init() to start and initiate ZDO task. Finally, the application should call osalTaskAdd() to add and register HCAS_App_Init() to start and initiate application object with application profile and application specific functions.

Basically, the former system tasks are declared within TI Z-Stack. Application programmer is not required to modify theses functions. As for the last step to add HCAS_App_Init(), it is related to the specific application and is needed to be implemented by the application designer. HCAS_App.c is the main application module of HCAS. Within this module, some functions need to be instantiated, including the HCAS_App_Init() to initiate the device, and HCAS_App_ProcessEvent() to handle system event for the application.

HCAS_App_Init() setups and startups the device and network with pre-defined system and device configuration properties. Next, it registers APO to the operating system. Then it registers application task events to the operating system. HCAS_App_ProcessEvent() is the main event processing function of the application. Within this function, many previously registered events passed back from system will be inspected and distributed to the corresponding handling functions, such as the receiving event from other devices would call the callback function HCAS_App_MsgCB() to handle the required response, or the interrupt event from hardware keys or switch would call the callback function HCAS_App_HandleKeys() to handle the required action, or the time-out event from timer would call the function HCAS_App_HandleTmr() to handle the necessary procedure and then setups the timer by calling osal_start_timerEx() for the next periodic event.

In event handling functions, if the device needs to send a data message to other device, it should call the AF_DataRequest() function in AF API for that sending request. If the device wants to read the sensor's data on other device, it should call the HalAdcRead() function in HAL API. If the device needs to send data via RS232 to PC or other device, it should call the HalUARTWrite() function in HAL API. If the device wants to turn its LED on, it should call the HalLedSet() function in HAL API. Other action requirements can refer to [12], [13], [14], [15], [18].

6 Conclusions

First challenge during the phase of the design and implementation for the HCAS is to study all related specifications, and the definitions and calling methods of all related API. This paper described our experience about the application framework of TI Z-Stack which was adopted for the development of HCAS.

The purpose of the pilot project of HCAS is to survey a feasible development platform based on ZigBee and to implement the pilot system to build up the required experiences upon the platform. Thus, the function of pilot HCAS is designed with ordinary scenario and uncomplicated system configuration.

Next stage, HCAS will be enhanced with practical scenario and advanced control facilities. For instance, the ventilation equipment could be replaced with air conditioner driven by continuous-speed inverter. In this case, not only the temperature can be controlled, but also the humility. The monitored bio-medical signals could be added by oxi-meter, breath-meter, blood-pressure meter, glucose meter, position tracking, and tilt-plus-gyro meter to detect fall down status. In addition, HCAS MMI could be redesigned from PC GUI to web-based GUI. Thus, medical experts can diagnose the cared person remotely, or the relatives can understand the healthy situation of the cared person.

References

[1] S. Helal,W. M. H. E. Zabadani, J. King, Y. Kaddoura, and E. Jansen, “The gator tech smart house: a programmable pervasive space,” IEEE Computer, vol.38, no.3, 2005, pp. 50–60.

[2] H. Park, M. B. Srivastava, and J. Burke. Design and implementation of a wireless sensor network for intelligent light control. In Proc. of Int’l Symposium on Information Processing in Sensor Networks (IPSN), 2007.

[3] P. E. Ross, “Managing Care through the Air”, IEEE Spectrum, December 2004, p. 26-31.

[4] C. Sunny, R. Peter,S. Bill, B. Sara, “Technology for care networks of elders”, IEEE Pervasive Computing, v3, n2, April/June, 2004, p. 22-29.

[5] Akyildiz, I., Su, W., Sankarasubramaniam, Y., and Cayirci, E., “A Survey on Sensor Networks”, IEEE Communications Magazine, August 2002.

[6] Yueh-Feng Lee, “Design and Implementation of a ZigBee-based Over-the-air Programming Platform,” Proceedings of CCL Technical Journal, PP. 29-35, March 2007.

[7] [H.-M. Tsai, O. K. Tonguz, C. Saraydar, T. Talty, M. Ames, and A. Macdonald, "ZigBee-based Intra-car Wireless Sensor Networks: A Case Study," IEEE Wireless Communications Magazine, December 2007.

[8] Wei-kou Li, Chih-Hung Chou, Zhi-Feng Lin, “Design and Implementation of a Zigbee-based Communication Substrate for Wireless Sensor Networks,” Proceedings of NCS, Oct 2006.

[9] IEEE Standard 802.15.4a™-2007.

[10] IEEE Standard 802.15.4™-2006.

[11] ZigBee Specification, December 1, 2006.

[12] Z-Stack API _F8W-2006-0021.pdf

[13] Z-Stack Sample Applications, F8W-2006-0023.pdf

[14] OSAL API_F8W-2003-0002.pdf

[15] HAL Driver API _F8W-2005-1504_.pdf

[16] cc2430.pdf (http://focus.ti.com/docs/prod/folders/print/cc2430.html)

[17] cc2431.pdf (http://focus.ti.com/docs/prod/folders/print/cc2431.html)

[18] Simple API for Z-Stack _F8W-2007-0021.pdf