International Journal of Future Generation Communication and Networking
Vol. 9, No. 6 (2016), pp. 55-64
http://dx.doi.org/10.14257/ijfgcn.2016.9.6.06

Study of IoT Terminal Interface Platform Based on Embedded Technology and Zigbee Protocol

Cui Jiantao[1*] and Zhao Xiaojun[2]

[1,2]*College of Software Engineering, Zhengzhou University of Light Industry, Zhengzhou, Henan, China 450002*
cuijiantao132@163.com

Abstract

With the development of IoT (interest of things) and network routing technology, IoT terminal data collection, integration and transmission technology improved significantly. Currently, research and use of IoT gradually widespread, but more of the terminal platforms were exploited for particular field. The versatility and compatibility of platform restrict its use and promotion. Based on the point, this paper aims to develop an IoT embedded system with flexible interface. Exploit the flexible interface with multi-sensor terminal access and ZigBee protocol to achieve MANET, sensor flexible access, low level of energy consumption, network isolation. The system has a high level of scalability and important value and practical significance to IoT application development.

Keywords*: Interest of things, embedded system, ZigBee, Flexible interface*

1. Introduction

The terminal platform of the IoT (the interest of things) is the basic module of it. It is the key of networking equipment. We should use the terminal platform to collect and handle various external sensing data, and transmit the data to the Internet through the various means of communication. Therefore, the level of the IoT development is largely depending on the level of embedded the terminal platform facing the IoT [1]. Currently, the development and promotion of the IoT encountered a number of bottlenecks, including: interface compatibility issues, energy and security issues. These factors restrict the development of the IoT technology.

The IoT is a new-emerging technique. After the age of Internet and wireless sensor network, people are gradually entering the era of The IoT. The IoT refers to uniquely identifiable objects (things) and their virtual representations in an Internet-like structure. The embedded system has been used to solve the IoT.Formal methods have been used in the analysis and verification of similar wireless systems, for example, Ballarini and Miller used the PRISM model checker to verify the Medium Access Control protocol [2]. The same method has been used to analyze wireless local area networks standard for a sub protocol of the IEEE 802.11 [3]. In another approach, the ETMCC model checking has been used for analysis a different variety of the central access protocol with the IEEE 802.11 [4].

Routing algorithm is the second point we focus on. Buonadonna *et al*. [5] states adjustments are one of the most difficult tasks in sensor deployments. The challenges are difficult to calibrate a massive number of sensors and inconveniences at physically accessing the sensors as they may be deployed in hostile environments. For the reason traditional calibration methods are not useful to sensor networks. To solve this problem, G. Tolle [6], V. Bychkovskiy [7] and M. Takruri [8] have proposed parametric calibration methods. C. Taylor [9] and K. Whitehouse [10] has investigated adjustment on mobile sensor networks.

ISSN: 2233-7857 IJFGCN
Copyright ⓒ 2016 SERSC

Other work in the paper is ZigBee protocol. The study of ZigBee includes the work in [11] which use opnet to simulate ZigBee. Fruth used PRISM model checker for probabilistic verification of the IEEE 802.15.4 networking standard [12]. Yang *et al*. used OMNeT++ to separate event simulation for MSSNs verification. ZigBee protocol was imitated in the data link layer [13]. Duflot *et al*. using the DTMC model to present an analysis of the discovery phase of bluetooth wireless communication protocol [14]. Chunqing and Jiancheng studied the data security protection of ZigBee technology [15]. The work in used Higher Order Logic theorem certificating for analysis probabilistic properties in wireless systems [16].

In the approach, we analyze embedded routing algorithm, and verify the functional behaviors of the ZigBee protocol stack considering the transaction level functionality of the protocol. We use multi-sensor embedded terminal with flexible interface and the ZigBee protocol to build an IoT access platform which achieved ad-hoc networks, flexible access of sensor, low power consumption and network isolation.

2. Issues Formulation

At present, interest of things is penetrating into all areas of daily life. In order to handle and unify the different information, we need a smart interface platform with good compatibility. There are two kinds of smart interface platform of the IoT: the traditional PC platform and embedded platform. With the development of chip technology, embedded system has many outstanding advantages than the traditional PC platform. Therefore, in this paper, the design of the platform will be based around an embedded system. Figure I is the basic architecture of an embedded system.

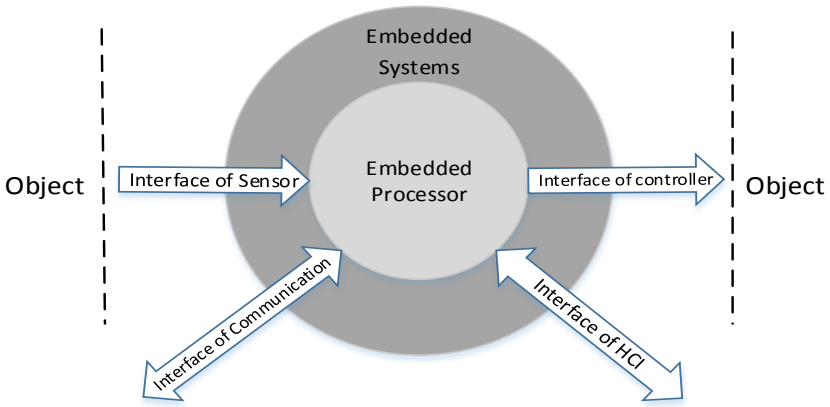

Figure 1. Architecture of Embedded System

According to the common classification embedded processor and embedded operating system, there are six basic designs to choose:

1. 80C51 microcontroller-based solutions;
2. ARM-based and no operating system solutions;
3. ARM-based and uCos solutions;
4. ARM-based and uCLinux solutions;
5. ARM-based and Linux solutions;
6. FPGA and DSP-based solution.

Costs from the price, the development cycle, computing capability, six aspects of scalable, multi-process support, application support for program analysis, comparing the results shown in Table 1:

 Copyright ⓒ 2016 SERSC

Table 1. Comparison of System Design Schema

Schema	Cost	Operation Ability	Scalability	Multi-process support	Application support
1	Low	Low	Poor	No	No
2	Medium	Forced	Poor	No	No
3	Medium	Forced	Medium	Yes	Yes
4	Medium	Forced	Forced	Yes	Yes
5	Medium	Forced	Strong	Yes	Yes
6	Forced	Strong	Poor	Yes	No

To avoid compatibility problems, we choose Solution 5 as the foundation of platform.

3. IoT and Embedded System Architecture

IoT generally divided into three layers, the perception layer, the network layer and the application layer of three parts [17]. Perception layer responsible to interact various physical objects and collect physical parameters and event information. Network layer has broader connectivity features, which could transport information to the application layer accessible, secure and reliable.

The ultimate goal of IoT is to achieve the wisdom. Figure 2 is a classic FIG from knowledge to wisdom [18]. IoT must collect, integrate and analyze information according to this progressive relationship to achieve the "wisdom". As an IoT system, the platform must have such a function too.

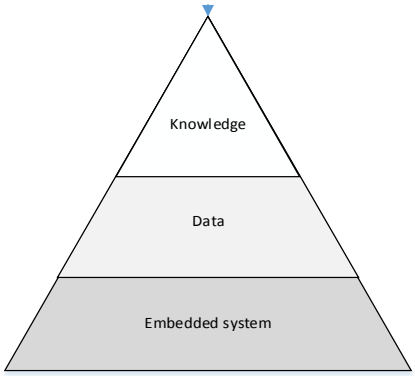

Figure 2. FIG from Knowledge to Wisdom

The overall scheme is shown in Figure 3. Due to the need of accessing different types of sensor, the platform should to support multiple sensor mechanism and development framework. We choose application layer to responsible to configure driver selection mechanism. Application layer sends configuration information to terminal, and then terminal load the corresponding driver after receiving the information. This mechanism could ensure the compatibility and scalability of the platform system.

Copyright ⓒ 2016 SERSC

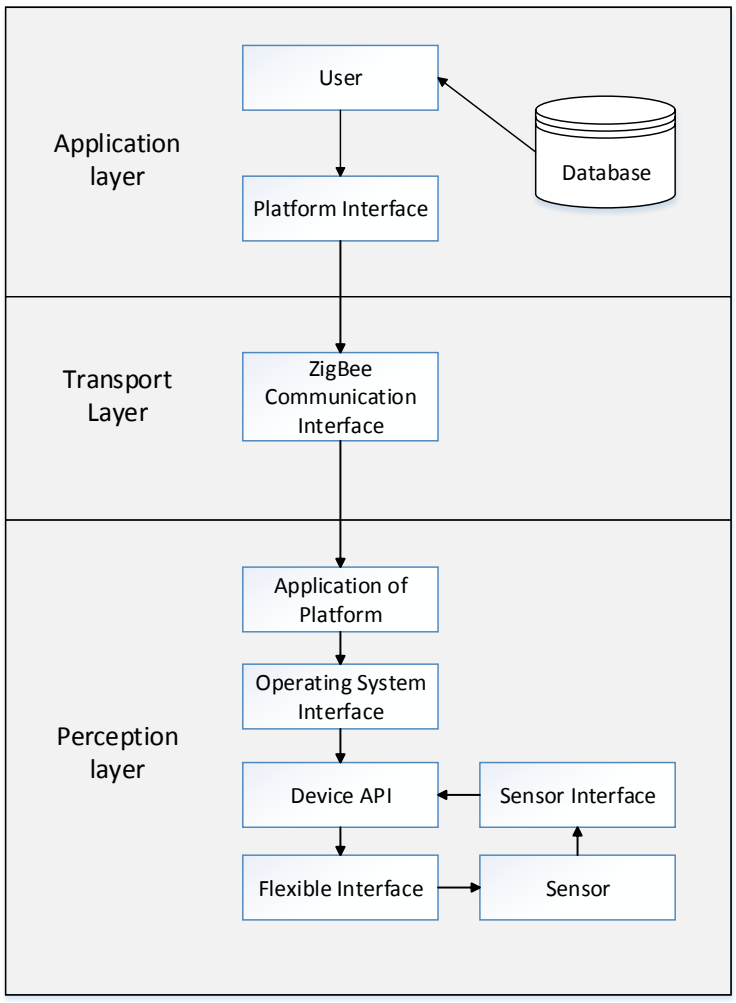

Figure 3. The Overall Scheme

According to design requirement described above, this paper makes a very high demand for hardware design. Sensor is an important part of the platform. If applications access to the hardware directly, the coupling of the system will be too much. Changing in demand or sensor will lead to the change of the whole system. And post-maintenance cost would be too high. Such platform was hardly to support expansion. So we need a kind of sensor which is able to access many types of application, and designed with a common sensor interface. In other words, we need a more advanced way to manage and access the hardware.

Rely on the hierarchical ideology of embedded operating system, we separate the various parts of the code, and call each other code through the interface. This approach can ensure the stability of the entire system, and isolate each part effectively. When the function changed, the system could operate effectively. The diagram of embedded operating system and drivers shown in Figure 4:

 Copyright ⓒ 2016 SERSC

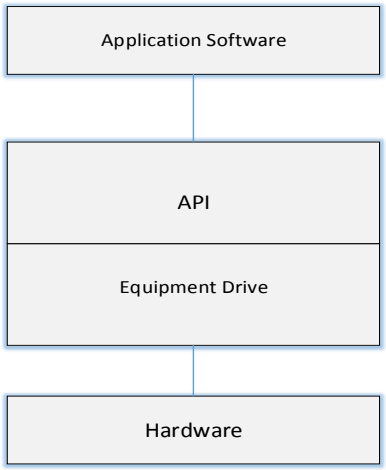

Figure 4. The Diagram of Embedded Operating System and Drivers

The applications connect to the hardware layer through the middle layer but not directly. So when hardware changed, we only need to change the hardware device driver to ensure the normal operation of the hardware. In order to achieve the flexible interface, we should define a virtual device in the program first. The algorithm is as follows:

```
typedef struct _virtualD
{
        char* devicename;
        int (*open)();
        int (*read) (char* data, int size);
        int (*write) (char* data, int size);
} virtualD;
```

The function defines a character pointer and three function pointers that can be expanded on demand.

Embedded hardware design is the base of the system, but without embedded software, the system cannot run normally. The involved embedded software could divide in two types. One is embedded software running on the embedded IoT platform. This kind of embedded software includes coordinator and terminal node. Coordinator is in charge of building and maintaining network, responding to the information sent by user's client. Terminal node is in charge of collecting sensor data, corresponding to coordinators. The other kind of embedded software is working on user's client, using to interact with users.

In summary, the process of the IoT platform is shown in Figure 5.

Copyright ⓒ 2016 SERSC

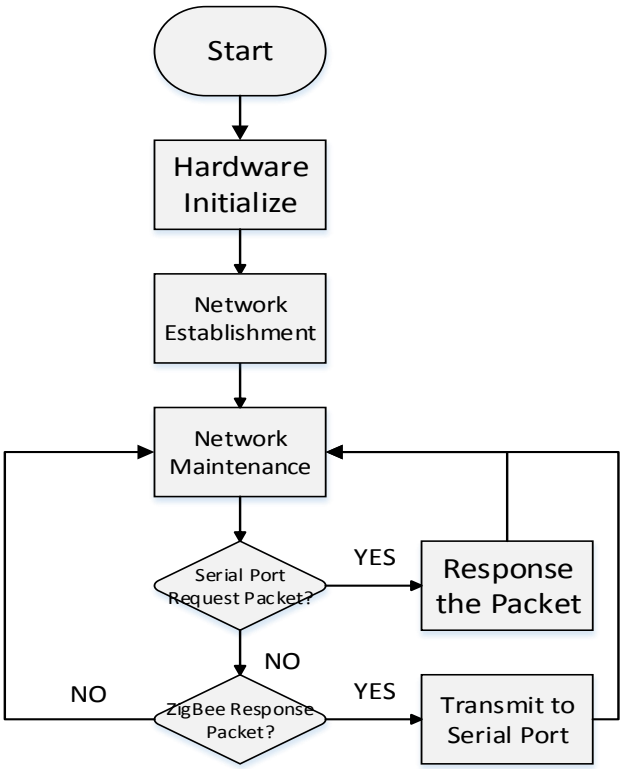

Figure 5. Process of the IoT Platform

3.1. The Stack of ZigBee

Embedded terminals should have the ability to deal with transections and change configuration flexibility for the different needs. The stack of ZigBee, Z-Stack, is based on osal (Operating System Abstraction Layer). The core of system resource management mechanism is multitasking. This section will introduce the principle and workflow of this operating system.

The purpose of Z-Stack's main function is initialize the system and executive function. There is a loop in the main function aims to handle various events which nasmed osal_start_syste(). Function prototype is as follows:

```
void osal_start_system( void )
{
    for(;;); // Forever Loop
    {
        osal_run_system();
    }
  }
```

osal_run_system()is the role of polling and handling events.The function is as follow:

```
void osal_run_system(void)
{
    unit8 idx = 0;
    osalTimeUpdateO;
    Hal_ProcessPoll();
    do {
        if (tasksEvents[idx]) //Task is highest priority that is ready
        {
            break;
        }
```

 Copyright ⓒ 2016 SERSC

```
    } while (++idx < tasksCnt);
if (idx < tasksCnt)
{
        unitl6 events;
        events = tasksEvents[idx];
        tasksEvents[idx] = 0; // Clear the Events for this task
        events = (tasksAiT[idx])( idx,  events);
        activeTaskID = TASK_NO_TASK;
        HAL_ENTER_CRITICAL_SECTION(intState);
        tasksEvents[idx] = events; //Add back unprocessed events to the current task
        HAL_EXIT_CRITICAL_SECTION(intState);
    }
}
```

3.2. ZigBee Routing Algorithm Analysis

Improved routing algorithm is to make full use of network bandwidth, improve data transmission rate and transmission quality. ZigBee networks generally use tree routing algorithm. The key mechanism of tree topology is network short address assignment and route configuration.

After finish creating a network, the coordinator will assign itself a short address 0x0000, set depth to 0. If node i add to the network by node k, node k would become the parent node of node i. According to the short address A_k and the network depth $Depth_k$ of node k, the short address of node i is A_i , and network depth is $Depth_i=Depth_k+1$.

If a new node is a terminal node, it won't have route function. According to the depth d, parent node k would assign the follow short address to the child node i:

$$A_n = A_k + C_{skip}(d) \times R_m + n \qquad 1 \le n \le C_m - R_m \tag{1}$$

The k in the formula is a sequence factor. It means it is the k-th child node which has routing function of the parent node. If the new node is a routing node, network address will be assigned according to the following formula:

$$A_n = A_k + 1 + C_{skip}(d) \times (n-1) \qquad 1 \le n \le C_m - R_m \tag{2}$$

Address remains allocated in order, C_{skip} Obtain by the following formula:

$$C_{skip} = \begin{cases} \dfrac{1 + C_m - R_m - C_m - R_m^{L_m - d - 1}}{1 - R_m} & R_m = 1 \\ 1 + C_m \times (L_m - d - 1) \end{cases} \tag{3}$$

Assuming that the target node's address is A, short address of the router network node is a, depth is d. The basis that analyzing the target node is or not its child node is:

$$a \le A \le a + C_{skip}(d-1) \tag{4}$$

If the target node is its child node, the next-hop address is:

$$J = \begin{cases} A, \\ a + 1 + \left[\dfrac{A - a - 1}{C_{skip}(d)} \right] \times C_{skip}(d) \end{cases} \tag{5}$$

If not, the next-hop is the parent node of the router.

The tree routing algorithm is a static routing algorithm which is suitable for static node or moving less node. The algorithm responds rapidly to the data packets transport.

Copyright ⓒ 2016 SERSC

4. Experience

This section will test and verify the functions and the parameters of the system.

4.1. Network Testing of Embedded Terminal Node

Packet Sniffer is a ZigBee protocol analyzer software which could analyze and decode the packet on protocol layers. The experiment would capture and analysis ZigBee packets by Packet Sniffer. Firstly, connect software and emulator, then open the coordinator and terminal nodes of the platform. Finally, the packet-capture software would get data. As showed in Table 2, line 1 to line 7 is a process of building network. We could find that once the network layer management function gets the PANID, it will immediately set source network address 0x0000, and then become a coordinator.

Table 2. Process of Building Network

Package Num	Time	Length	Dest.PAN	Dest.Address	Sour.Address
1	0	10	0xFFFF	0xFFFF	NULL
2	15527868	29	0x1111	0xFFFF	0x0000
3	23493459	10	0xFFFF	0xFFFF	NULL
4	23498776	29	0x1111	0x0000	NULL
5	25497043	10	0xFFFF	1xFFFF	NULL
6	25499763	29	0x1111	0x0000	NULL
7	26023518	10	0xFFFF	0xFFFF	NULL

After opening, the terminal node would join the corresponding network by identifying PANID. As showed in Table 3, the source address in the first line is 0x1716151413120735, the MAC address of the router. The node has not joined the network yet, so it didn't have a network address. And the destination address is the network address of the coordinator who's PANID is the same to the node's. That is 0x0000. The first line means the node is sending join request to the coordinator. When the coordinator receives the request packet, it will send a packet which short_addr_Assoc.status is success. This packet has a short address inside allocated by the stack. As we can see from the data in line 7, the source address of the terminal node is no longer a MAC address, but a short address 0x20CA. This means the node joins the network successfully. Then we finish the process of setting up network and taking node into the network.

Table 3. Process of Accessing in the Network

Package Num	Time	Length	Dest.PAN	Dest.Address	Sour.PAN	Sour.Address
1	26239602	21	0x1111	0x0000	0x1111	0x1716151413120735
2	26790426	5	NULL	NULL	NULL	NULL
3	26792259	19	0x1111	0x0000	NULL	0x1716151413120735
4	27304854	5	NULL	NULL	NULL	NULL
5	27305657	28	0x1111	0x1716151413120735	NULL	0x171615141312073B
6	27308284	5	NULL	NULL	NULL	NULL
7	27309936	40	0x1111	0x0000	NULL	0x20CA
8	27316972	5	NULL	NULL	NULL	NULL
9	27318157	40	0x1111	0xFFFF	NULL	0x0000

 Copyright ⓒ 2016 SERSC

4.2. Flexible Sensor Function Test

According to the description in section 3, to build the flexible interface, first, we should realize the call interface of open function to set parameters to ensure the sensor to work properly. Then realize the write function which could send the appropriate commands to set the working status of the sensor, the type of collected data and other functions. Lastly, realize read function to send data into the array pointed by parameter data through the read data function of the sensor. Finally, call device_register() function to register the device into the kernel.

4.3. Power Test and Verification

Through test by Power Monitor, platform power consumption is shown in Figure 6. The consumption could keep between 100mA and 400mA, which mean the algorithm could control power consumption effectively.

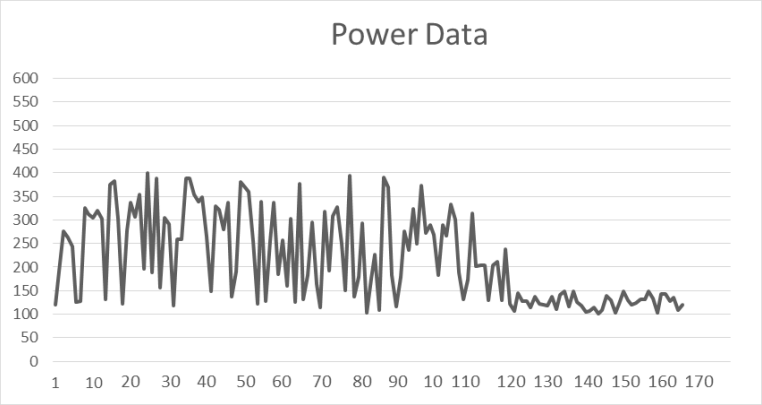

Figure 6. Power Consumption

5. Conclusion

In the paper, we designed a generic flexible accessible embedded interface platform for IoT application, and researched embedded system, ZigBee protocol, sensor technology. Based on this research, we considered the commonality of IoT terminal, studied and explored generic design from develop approach, node design. We realized versatile connect to the platform using API interface. The using of ZigBee protocol improved the quality of data transmission. We overcome the shortcoming of embedded platform, like un-compatible and unstable, achieved a good integration of IoT and embedded technology.

References

[1] "2.4GHz IEEE 802.15.4 and ZigBee application CC253X on-chip system solution user's Guide", Texas Instrument, **(2009)**.
[2] P. Ballarini and A. Miller, "Model Checking Medium Access Control for Sensor Networks", 2nd International Symposium on Leveraging Applications of Formal Methods, Verification and Validation, IEEE Computer Society Press, **(2006)**, pp. 255-262.
[3] M. Kwiatkowska, G. Norman and J. Sproston, "Probabilistic model checking of the IEEE 802.11 wireless local area network protocol", Process Algebra and Probabilistic Methods, Performance Modeling and Verification, Springer-Verlag, V. 2399 of LNCS, **(2002)**, pp. 169-187.
[4] M. Massink, D. Latella and J. P. Katoen, "Model Checking Dependability Attributes of Wireless Group Communication", Dependable Systems and Networks, IEEE Computer Society Press, **(2004)**, pp. 711-720.
[5] P. Buonadonna, D. Gay, J.Hellerstein, W.Hong, S. Madden, "Task: sensor network in a box", EWSN, Istanbul, Turkey, pp. 133-144.

Copyright ⓒ 2016 SERSC

[6] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay and W. Hong, "Amacroscope in the redwoods", SenSys, San Diego, California, USA, pp. 51-63.
[7] V. Bychkovskiy, S. Megerian, D. Estrin and M. Potkonjak, "A collaborative approach to in-place sensor calibration", IPSN, Palo Alto, CA, USA, pp. 301-316.
[8] M. Takruri, S. Challa and R. Chakravorty, "Recursive bayesian approaches for auto calibration in drift aware wireless sensor networks", Journal of Networks, vol. 5, **(2010)**, pp. 823-832.
[9] C. Taylor, A. Rahimi, J. Bachrach, H. Shrobe and A. Grue, "Simultaneous localization, calibration, and tracking in an ad hoc sensor network", IPSN, Nashville, Tennessee, USA, pp. 27-33.
[10] K. Whitehouse and D. Culler, "Calibration as parameter estimation in sensor networks", WSNA, Atlanta, Georgia, USA, pp. 59-67.
[11] U. Pesovec, J.Mohoroko and Ž.Cucej, "Upgraded OPEN-ZB 802.15.4 Simulation Model", ELMAR International Symposium, IEEE Computer Society Press, **(2010)**, pp. 281-284.
[12] M. Fruth, "Probabilistic Model Checking of Contention Resolution in the IEEE 802.15.4 Low-Rate Wireless Personal Area Network Protocol", Symposium on Leveraging Applications of Formal Methods, Verification and Validation, IEEE Computer Society Press, **(2006)**, pp. 290-297.
[13] X. Yang, K. Liu, Y. Cui and J. Zhang, Modeling and Simulation for Maritime Surveillance Sensor Networks", IEEE International Symposium on Communications and Information Technologies, IEEE Computer Society Press, **(2010)**. pp. 215-219.
[14] M. Duflot, M. Kwiatkowska, G. Norman and D. Parker, "A Formal Analysis of Bluetooth Device Discovery", International Journal on Software Tools for Technology Transfer, vol. 8, no. 6, **(2006)**, pp. 621-632.
[15] L. Chunqing and Z. Jiancheng," Research of ZigBee's Data Security and Protection. In IEEE International Forum on Computer Science-Technology and Applications", IEEE Computer Society Press, **(2009)**, pp. 298-302.
[16] O. Hasan and S. Tahar, "Probabilistic Analysis of Wireless Systems Using Theorem Proving", Electronic Notes in Theoretical Computer Science, vol. 242, no. 2, **(2009)**, pp. 43-58.
[17] CC2430 Chip Datasheet. XianTan Skye electronic technology Co., Ltd.
[18] H. Xia and W. Ma, "Design of WSN Based on CC2430", Electronic Technology Application, vol. 5, **(2007)**, pp.45-48.

 Copyright ⓒ 2016 SERSC