On the Design and Implementation of a Tailored DHCP Protocol for Wireless Sensor Networks based on FreeOS

Fei Gao[1,2,a], Miao Guo[1], Xia Wang[1],Yu Wang[2]

[1]Key Lab of Wireless Sensor Networks,Yunnan Minzu Unversity,Kunming,Chian,650500

[2]The Networking Research Lab ,UNC Charlotte ,Charlotte, United States, NC 28223

[a] wdxy_gf@163.com

Keywords:WSN, DHCP, Tailoring, Client/Server; FreeOS; Serial Port Debugging.

Abstract: Based on the principle of Dynamic Host Configuration Protocol (DHCP), this paper proposes a new implementation of a tailored DHCP protocol used for IP address management in wireless sensor networks (WSN). Several key design issues, such as frame structure of tailored DHCP packets, address assignment scheme, client/server processing program and timing processing program are introduced. An experimental system is designed and implemented via embedded FreeOS micro OS and tailored DHCP program over two types of sensor nodes. Experimental results show that the proposed protocol can successfully support interconnection, packets exchange and IP address assigning and retrieving among sensor nodes within IP-based networks.

I. Introduction

As being a special kind of wireless networking, the wireless sensor networks (WSN) have some distinguished features such the small smart nodes based, high dense deployment, self-organization, multiple hop, changeful topology and so on when to compare with traditional computer networks. The nodes of WSN are very frequently to join in or leave from network due to its limited power supply, sleep schemes or environment influences. So, the management of node and topology control is the challenge issue. Basically, the communication protocol of WSN is IEEE802.15.4 or Zigbee, they have low transmission rate and energy consume. The Bluetooth or infrared technology also is possible option. Normally, the WSN node’s address is identified and managed depending on its series number or MAC. In some applications, for example, when a host in Internet wants to control a node’s state, this method could be not efficient and flexible enough.

The IP (Internet Protocol) is the network-routing protocol aimed to provide a connectionless packet transmission between hosts, being them on different networks and even with different network technologies. So, it has been adopted by LAN and Internet for many years. Now, IP is becoming the incontestable headstone for most modern interconnection networks system and applications. Also, some IP-based solutions has been proposed for wireless sensor networks construction in recent years [1,2,3]. As being a core part of IP protocol suite, the DHCP has been used for the assignment and revocation of IP address when the devices access to or disconnect from Internet or other networks[4]. However, the DHCP program could not be embedded into the WSN nodes directly due to the following reasons: (1) To compare with LAN, WSN node has limit`ed processing abilities, autonomous power supply, RF bandwidth, storage space and so on. So, it would be better if there less information involved in processing procedure. For example, the CC2420 has only 128 bytes used for the RF T/R frame. Meanwhile, the MAC frame defined by IEEE802.15.4 is less than 255 bytes. (2) By DHCP principle, there at least four packets (DISCOVER、OFFER、REQUEST、ACK) are required to achieve an IP address’s generating and assigning process. For the WSN, less transmission process means less resource consumption and better delay performance. (3) By DHCP, when a terminal logout from a LAN, the server will know and retrieve the IP address after its lease expire. But in WSN, in order to retrieve IP source in time, the server should sense node’s joining or leaving and renew its available address list as quickly as possible, rather than wait for relatively longer lease expire time.

The remainder of the paper will introduce the details of the working principle, the tailoring and redefining of the frame structure of PDHCP, experiment condition and so on.

II. Design of Tailored DHCP

A.Message Frame Structure of DHCP

According to the WSN's attributes and actual needs of application system, we defined a tailored DHCP structure through tailoring the DHCP's that is shown as Figure 1. The signification of each field is described in Table 1.

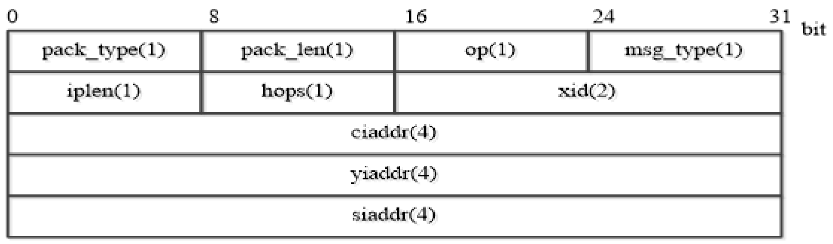


Figure 1: the frame of tailored DHCP

Table 1: Description of tailored DHCP Frame Field

Packet Type	Description
pack_type	Packet type, 1= PDHCP packet.
pack_len	Packet length.
op	Operation code, 1=BOOTREQUEST from client to sever; 2 =BOOTREPLY from server to client
msg_type	Message type, 1=PDHCP_REQUEST; 2=PDHCP_ACK; 3=PDHCP_ONLINE; 4=DHCP_ONLINE_ACK
iplen	IP address length, set constant 4x.
hops	Number of hops.
xid	Transmission series id number.
ciaddr	IP address of client.
yiaddr	IP address of pre-assigned client by server.
siaddr	IP address of server.

B.Tailored DHCP Working Procedure Design

Similar to DHCP, Tailored DHCP also uses the REQuest/ACKnowledgement scheme to build the connection between client and server node. But different from DHCP processing, the times of handshake and required packets of PDHCP are reduced in order to save the access time of node. The Figure 2(a) is a diagrammatical draw of the C/S interconnecting procedure. Firstly, a client broadcasts its REQ to enquire whether there is a server existing within its coverage. If yes, the server will reply an ACK to the client with a pre-distributed IP address. Finally, the client will reply a SELECT packet to the serve to conform that the IP is accepted. As you see, there only three packets involved in this procedure.

However, in actual WSN system, there could be more than one server deployed within a client's coverage. So a client could receive more than one ACK after its REQ has been sent. For simplification, in our proposal, the client node will choose the server whose ACK is the first coming as its current server node. Other server's ACKs will be discarded, as shown in Figure 2(b).

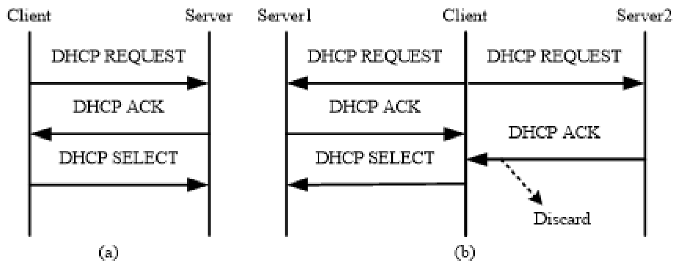


Figure 2: Tailored DHCP acquire IP and choose server

Another crisis issue is how to withdraw the IP address when a client is sleep, dead or out of the server's coverage. Our idea is to let every server broadcast ONLINE to enquire all nodes which are covered by the current server and to judge the state of the nodes by receiving an ONLIEN ACK. If a client replied an ONLINE ACK packet to sever, it means to say 'I am online'. If one node is out of online, the server will not receive any reply from it. After three enquiring times, in this situation, the IP address will be taken back and prepared for the next requirement by the server.

Depending on the above, there three main blocks are included in our design that are called as client programs, server programs and timing programs. We will discuss them in following chapter.

III. Implement of Tailored DHCP

A.Client Main Program Flow

There three possible states are defined for a client node in our proposal, i.e. IDLE, REQUEST_SENT, IP_IN_USE. Corresponding, there three variables will be stored in each client's RAM of node for recording different states. Figure 3 give the client work main program flow chart. The breach program is not drawn here for the paper's simplification.

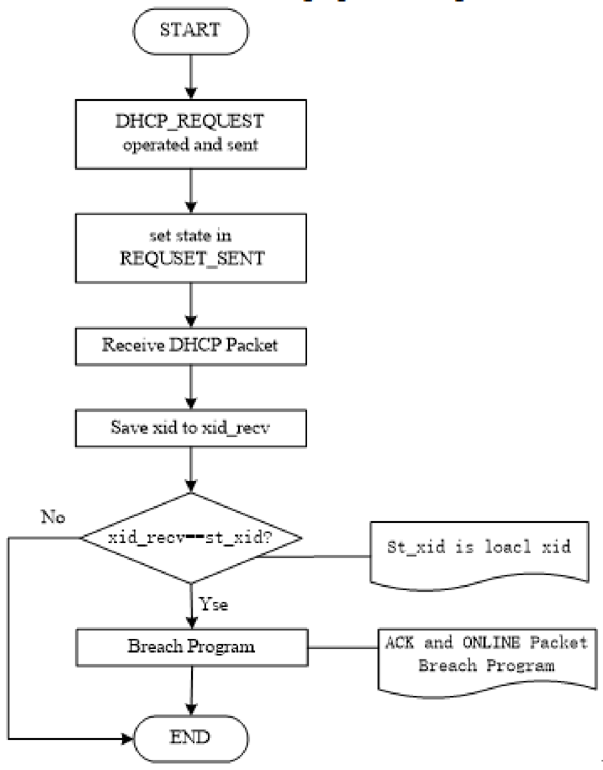


Figure 3: Main program flow on client-side

B.Server-side Program

For server-side, the assigning and retrieving on IP address is its core issue. The data structure is designed to manage every node IP, like as:

```
  typedef struct _ip_m_block {
    u8 dhcp_tick : 8;
    u8 dhcp_state : 8;
    u16 mac_addr : 16;
} ip_m_block;
```

Here, 'dhcp_tick', the first component in ip_m_block, is the node's IP unit timer. The second component 'dhcp_state' will be set to one of four states (IDLE, ACK_SENT, IP_IN_USE, ONLINE_SENT). The last component 'mac_addr' represents the node's MAC . It can be set as the long address (8 bytes) or short address (2 byte, our option).

The packets received by server include three types, i.e. DHCP_REQUEST, DHCP_SELECT, and DHCP_ONLINE_ACK. The main program of server-side is given in Figure 4. From it we can see there three sub-flowcharts are designed to process the packets of DHCP_REQUEST, DHCP_SELECT, and DHCP_ONLINE_ACK respectively. Due to the paper length limit, they are not presented here.

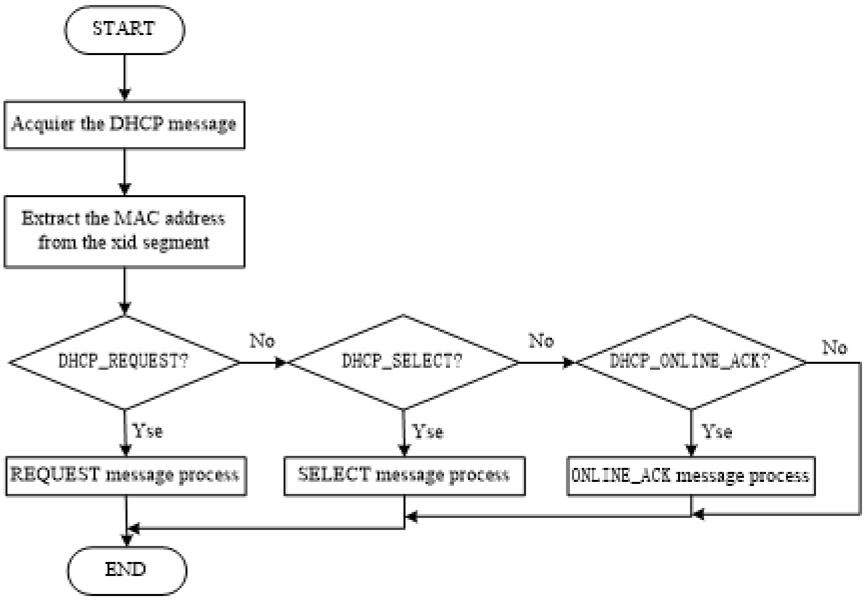


Figure 4: Main program flow on server-side

C.Timing

Because of the unreliable and unstable wireless communication environment, the packet loss often occurs during the message exchanging in actual WSN system. To develop an efficient timing scheme is very important for both client-side and server-side to guaranty the accurate and reliable packet re-transmitting and exchange between server and client after some packets lost. Our timing scheme's diagrammatic flow for client and server is shown in figure 5 and figure 6 respectively. Relatively, the server-side timing scheme is more complex than client-side due to it takes charge more process events. It includes one main and three breach programs. For the reason of the limited of pages, the beach programs are not shown in here.

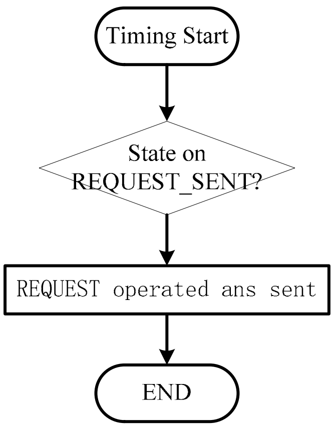
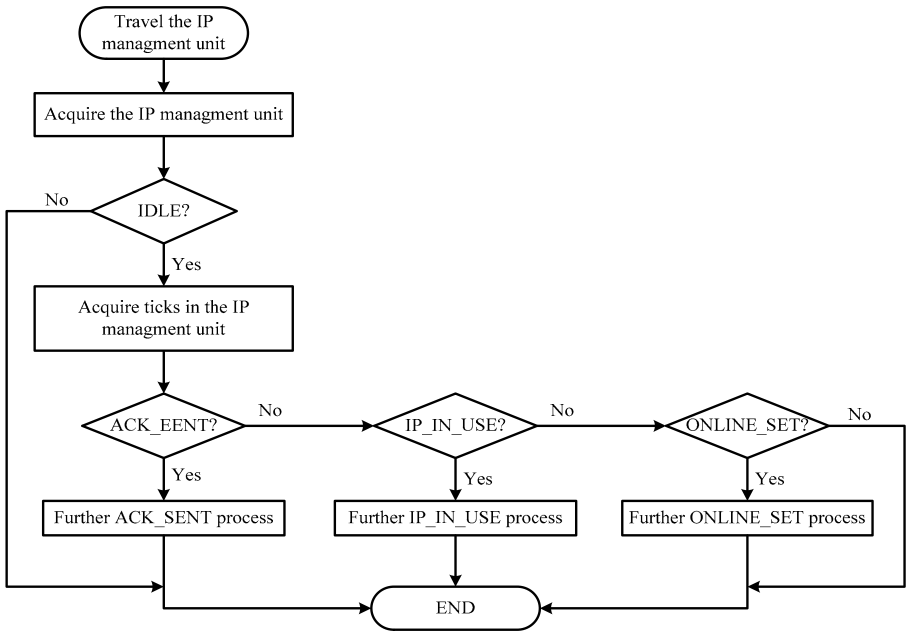


Figure 5:Timing Program of Client-side


Travel the IP managment unit
Acquire the IP managment unit
No
IDLE?
Yes
Acquire ticks in the IP managment unit
ACK_EENT?
No
IP_IN_USE?
No
ONLINE_SET?
No
Yes
Yes
Yes
Further ACK_SENT process
Further IP_IN_USE process
Further ONLINE_SET process
END
Figure 6: Main timing program flow on Server-side

IV. Experiment and Conclusion

A. Equipment and Environment

In order to examine the performance of the tailored DHCP, a real WSN experiment system had been established. Here, we employed two types of WSN set as being our test bed, i.e. HBE-Zigebx WSN and GAINZ WSN products. They are manufactured by Hanback Co. of Korea [5] and Shenlian Tech Co. of China [6] respectively. The two products are equipped ATmega128 micro CPU [7] and CC2420 RF transceiver[8]. The software platforms of program developing we used are Windows 7, Series Assistant and IAR for AVR5. In consideration of simplicity and compatibility, an open source operating system - FreeRTOS is adopted and embedded into the nodes to support PDHCP program running[9]. Especially, the frequency of crystal oscillator must be adjusted to fit different kinds of node through the frequency configuration in macro file of configCPU_CLOCK_HZ. The configuration method is to store the messages such as node type, operation frequency, node IP, DHCP_XID and et al in EEPROM. When power on, the node can read own frequency out via the function mote_osc()。

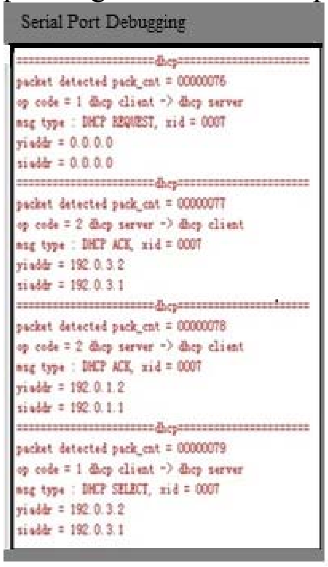
B. Scene I: Single Server and Single Client

Let one GAINZ node be a client and one HBE-ZigbeX be a server, which connected with PC's serial port. The programs are embedded into nodes via programming board, including .hex and .eep. files. By supporting of the software of Serial Port Assistant, the instruction code '1' was sent to server to inform it that only DHCP work process should be presented in this experiment.

Figure 7(a) shows the information exchanging procedure between client and server. It is easy to find there are five message packets, which were explained in Table 1, and five exchanging states being completed in Figure 7(b). From the five states, we can conclude the PDHCP works following the principle we introduced in chapter 3 and 4.

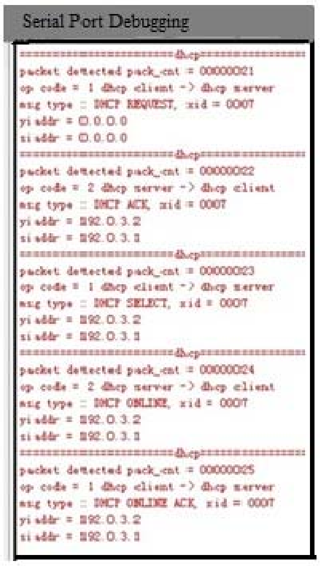
C.Scene II: Multiple Servers and Single Client

In this scene, we want to examine how a client to choose a server when there are more than one server in its RF coverage. There two server nodes and one client node were deployed in this experiment. The result of experiment is shown in figure 9(b). From it, we can see the client (ixd=0007) received the ACKs with the IP address of 192.0.1.2 and 192.0.3.2 from the two servers which IP address are 192.0.3.1 and 192.0.1.1 respectively after it broadcasted REQUEST in air. Depending on the PDHCP principle we mentioned in Figure 4, the client node accepted.



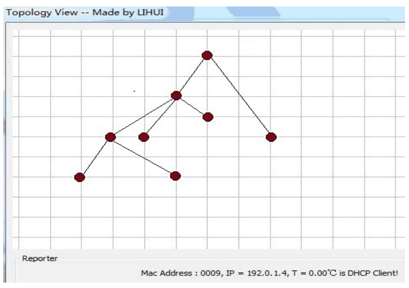
```
Serial Port Debugging
=========================dhcp=========================
packet detected pack_cnt = 00000076
op code = 1 dhcp client -> dhcp server
msg type : DHCP REQUEST, xid = 0007
yiaddr = 0.0.0.0
siaddr = 0.0.0.0
=========================dhcp=========================
packet detected pack_cnt = 00000077
op code = 2 dhcp server -> dhcp client
msg type : DHCP ACK, xid = 0007
yiaddr = 192.0.3.2
siaddr = 192.0.3.1
=========================dhcp=========================
packet detected pack_cnt = 00000078
op code = 2 dhcp server -> dhcp client
msg type : DHCP ACK, xid = 0007
yiaddr = 192.0.1.2
siaddr = 192.0.1.1
=========================dhcp=========================
packet detected pack_cnt = 00000079
op code = 1 dhcp client -> dhcp server
msg type : DHCP SELECT, xid = 0007
yiaddr = 192.0.3.2
siaddr = 192.0.3.1
```

Figure 7(a): Single Server and Single Client



```
Serial Port Debugging
=========================dhcp=========================
packet detected pack_cnt = 00000021
op code = 1 dhcp client -> dhcp server
msg type : DHCP REQUEST, xid = 0007
yi addr = 0.0.0.0
si addr = 0.0.0.0
=========================dhcp=========================
packet detected pack_cnt = 00000022
op code = 2 dhcp server -> dhcp client
msg type : DHCP ACK, xid = 0007
yi addr = 192.0.3.2
si addr = 192.0.3.1
=========================dhcp=========================
packet detected pack_cnt = 00000023
op code = 1 dhcp client -> dhcp server
msg type : DHCP SELECT, xid = 0007
yi addr = 192.0.3.2
si addr = 192.0.3.1
=========================dhcp=========================
packet detected pack_cnt = 00000024
op code = 2 dhcp server -> dhcp client
msg type : DHCP ONLINE, xid = 0007
yi addr = 192.0.3.2
si addr = 192.0.3.1
=========================dhcp=========================
packet detected pack_cnt = 00000025
op code = 1 dhcp client -> dhcp server
msg type : DHCP ONLINE ACK, xid = 0007
yi addr = 192.0.3.2
si addr = 192.0.3.1
```

Figure 7(b): Multiple Servers and Single Client



Mac Address : 0009, IP = 192.0.1.4, T = 0.00℃ is DHCP Client!

Figure 8: Topology structure

192.0.3.2 to be its own IP address because the ACK from the server of 192.0.3.1 was prior to be received by the client than the server of 192.0.1.1. The result is shown in figure 8.

D.Conclusions

From above, we can conclude the results here:

a) The micro OS FreeOS had been embedded into the sink node and work nodes successfully. The macros and parameters we defined are working well as we expect.

b) The programs we developed based on the principle of PDHCP had been embedded into the nodes successfully, including both server nodes and client nodes.

c) From the monitoring data results on the transmission between servers and clients, we can conclude that the PDHCP has performed the connecting between nodes and the IP address management task successfully. So the PDHCP we proposed is correct and efficient.

Acknowledgements

This work has been supported by the Special Project of Science Research of Yunnan Education Ministry, Grant 201109, General Project of Scientific Research Fund of Yunnan Provincial Education Department, Grant 2014Y254, and the National Science Fund of China, Grant 61461056.

REFERENCES

[1] Joel J. P. C. Rodrigues and Paulo A. C. S. Neves: *A survey on IP-based wireless sensor network solutions*, Int. J. Commun. Syst. Vol. 23(2010), p. 963–981

[2] Sungmin Hong, Daeyoung Kim, Minkeun Ha, et al, SNAIL: *An IP-Based Wireless Sensor Network Approach To the Internet of Things*, IEEE Wireless Communications • December (2010), p. 34-41

[3] S.S. Bhunia, S.K. Das, S. Roy and N. Mukherjee: *Mobility management in IP based Wireless Sensor Network using TinyOS*, 2012 Sixth International Conference on Sensing Technology (ICST), p. 759-764

[4] R. Droms, RFC 1541:Dynamic Host Configuration Protocol[S/E], information on http://tools.ietf.org/pdf/rfc1541.pdf

[5] Information on http://www.hanback.cn/en/pro.asp?anclassid=6&anclass=USN,Hanback Electronics

[6] Information on http://www.wsn.org.cn/_d275112918.htm，Shenlian Tech

[7] 8-bit Atmel Microcontroller with 128Kbytes In-System Programmable Flash, information on http://www.atmel.com/Images/doc2467.pdf

[8] 2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver, information on http://www.ti.com/lit/ds/symlink/cc2420.pdf

[9] Zhibo Pang, Kan Yu, J. Akerberg, M. Gidlund: *An RTOS-based architecture for industrial wireless sensor network stacks with multi-processor support*[C], 2013 IEEE International Conference on Industrial Technology (ICIT), 25-28 February, Cape Town, South Africa(2013)