How to implement DHCP in W7100A

Version 1.1

`

Table of Contents

1   Introduction

DHCP server uses DHCP to assign details like IP address or settings to the client that uses the DHCP. This application note will explain how to implement DHCP client by using W7100.

2 DHCP

DHCP uses UDP (User Datagram Protocol) through Transport Layer, and communicates with the DHCP server using UDP broadcast. Fig.1 describes the communication between the DHCP server and client.
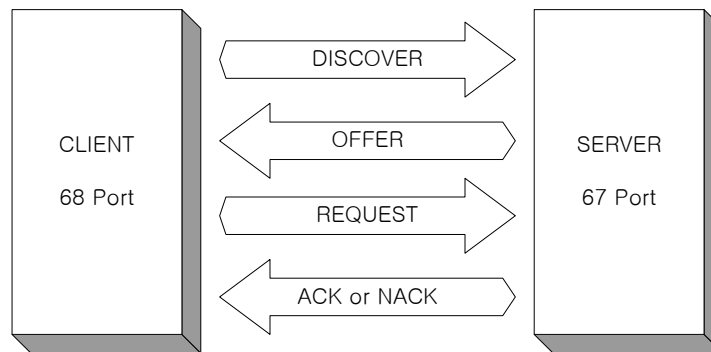

Fig. 1 DHCP Message

DHCP Client broadcasts the 'DISCOVERY message' on the connected network. If a DHCP Server exists on the computer network, the DHCP Server will receive the 'DISCOVERY message' and send the 'OFFER message' to the DHCP client. 'OFFER message' consists of an IP that can be used by DHCP client, Gateway(G/W), network information that is the same as the DNS server IP, and information like lease time. DHCP Client can acknowledge the DHCP Server by receiving the 'OFFER message,' and send the 'REQEUST message' to use the information suggested by the server. Then after the 'REQUEST message' is received, the DHCP Server decides whether the Lease duration (the period of time where the DHCP client IP address can be used) and Network configuration can be used or not. If the above information can be used, the DHCP Server will send an ACK message to the DHCP Client; or a NACK message if the information can't be used.

3 DHCP Client

3.1 Demonstration

Fig.2 shows the result of DHCP Client Test by connecting iMCU7100EVB and network gear. The DISCOVER/OFFER/REQUEST/ACK messages can be checked. The DHCP Client network configuration assigned by the DHCP Server can be checked also.
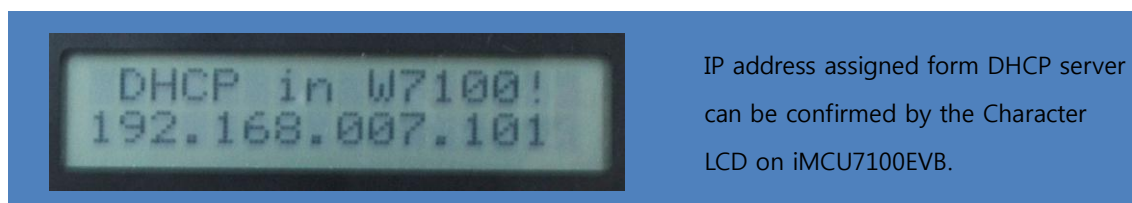

Fig. 2 DHCP Client TEST

IP address assigned form DHCP server can be confirmed by the Character LCD on iMCU7100EVB.

4 Software

4.1   Socket

4.1.1 OPEN

 Since the internet connection is not required between the DHCP Server and DHCP Client, DHCP needs to be opened by UDP. Use Sn_MR_UDP as shown below to OPEN the socket (line382).

```
dhcp_app.h
/* UDP port numbers for DHCP */
18 :      #define DHCP_SERVER_PORT  67        /* from server to client */
19 :      #define DHCP_CLIENT_PORT  68        /* from client to server */
156 :     #define SOCK_DHCP   0 // (0~7)


dhcp_app.c
382 :     socket(SOCK_DHCP, Sn_MR_UDP, DHCP_CLIENT_PORT, 0x0)
```

4.1.2 SEND

The user must set the other user's IP address and port before sending the DHCP data. The DHCP_SERVER_PORT is fixed to 67. When DISCOVER or REQUEST is operated, The IP address uses the BROADCAST address (255.255.255.255) to send data. After receiving the DHCP Server IP from the DHCP message, use the parsed Server IP[] to send data (line204).

```
dhcp_app.c
/* DST IP : BroadCasting*/
    202 :         S_port = DHCP_SERVER_PORT;
    203 :         if(REREQ != 0)
    204 :         for (i=0; i<4; i++) addr[i] = ServerIP[i];
    205 :         else
    206 :         for (i=0; i<4; i++) addr[i] = 0xFF;


// send MSG to Broadcasting
    213 :         sendto(s, (uint8 *)(&MSG.op), RIP_MSG_SIZE, addr, S_port);
```

4.1.3 RECEIVE

This section is identical with the RECEIVE section of UDP. Please refer to 'How to implement UDP for W7100.'

```
dhcp_app.c
```

---

228 :	len = recvfrom(s, (uint8*)&MSG.op, length, ServerAddrIn, &ServerPort);

4.1.4 DHCP Message Format

The structure below is composed of codes that work with the DHCP message. (Please refer to the RFC1541 document for more details on DHCP Message Format and each Field). The fileds below the ciaddr field are used to send network information; and the options field is used to send message type and information like client identifier.

```
dhcp_app.h

117 :    #define OPT_SIZE          298      //DHCP total size : 576
118 :    #define RIP_MSG_SIZE      (236 + OPT_SIZE)
119 :    #define POS_OPTION        321


121 :    typedef struct _RIP_MSG{
122 :      uint8   op; //Message op code / message type. 1 = BOOTREQUEST, 2 = BOOTREPLY
123 :      uint8   htype; //Hardware address type (e.g., '1' = 10Mb Ethernet)
124 :      uint8   hlen; //Hardware address length (e.g. '6' for 10Mb Ethernet)
125 :      uint8   hops; //Client sets to zero, optionally used by relay agents
126 :      uint32   xid; //Transaction ID.
127 :      uint16   secs; //Seconds passed since client began the request process
128 :      uint16   flags;
129 :      uint8   ciaddr[4]; //Client IP address
130 :      uint8   yiaddr[4]; //Your(Client) IP address
131 :      uint8   siaddr[4]; //Server IP address
132 :      uint8   giaddr[4]; //Relay agent IP address
133 :      uint8   chaddr[16]; // Client hardware address
134 :      uint8   sname[64];          //Optional server host name
135 :      uint8   file[128]; //Boot file name
136 :      uint8   OPT[OPT_SIZE]; //Optional parameters
137 :    }RIP_MSG;
```

4.2 DHCP Functions

The functions for implementing the DHCP client are listed under dhcp_app.h below.

```
dhcp_app.h

145 :    void send_DHCP_DISCOVER(SOCKET s); /* SEND DISCOVER MESSAGE to BROADCAST*/
146 :    char parseDHCPMSG(SOCKET s,uint16 length);      /* for parsing DHCP MESSAGE */
147 :    char DHCP_SetIP();                              /* for setting IP ADDRESS */
```

```
148 :    void check_dhcp(void);                              /* for DCHCP processing */

149 :    void send_DHCP_REQUEST(SOCKET s, uint8 REREQ);  /* SEND REQUEST MESSAGE */

150 :    void set_network(void);                    /* setting network configuration   */
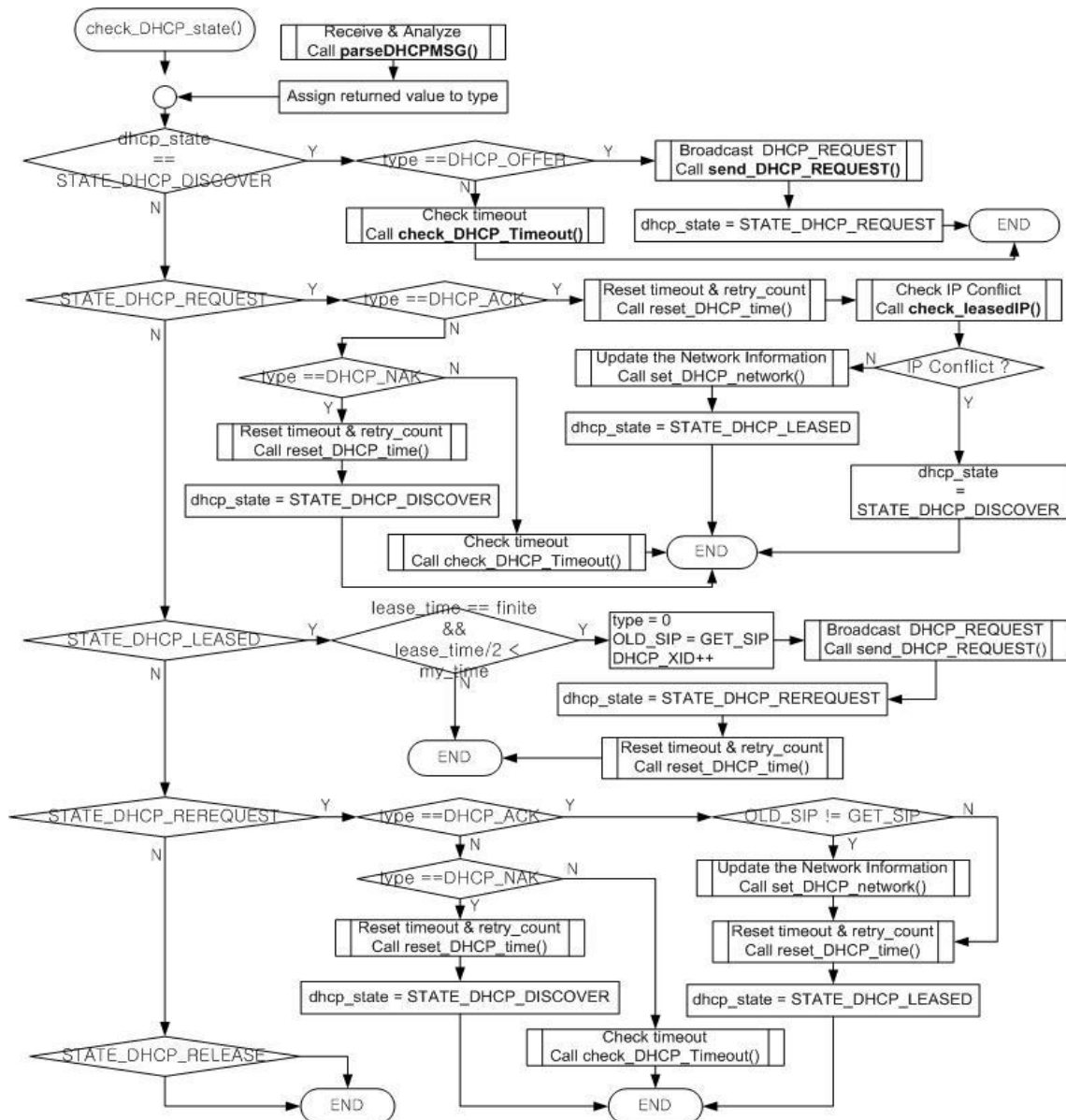```



Fig. 3 check_dhcp function

Fig.3 shows the flowchart of check_dhcp(). DHCP_OFFER, DHCP_ACK, and DHCP_REQUEST are operated depending on the change (transition) of dhcp_state. Fig.4 shows the flowchart of parseDHCPMSG(). parseDHCPMSG() is used to phishing (parsing) DHCP messages. Fig.5 shows the flowchart of check_LeaseIP() and send_DHCP_REQUEST().
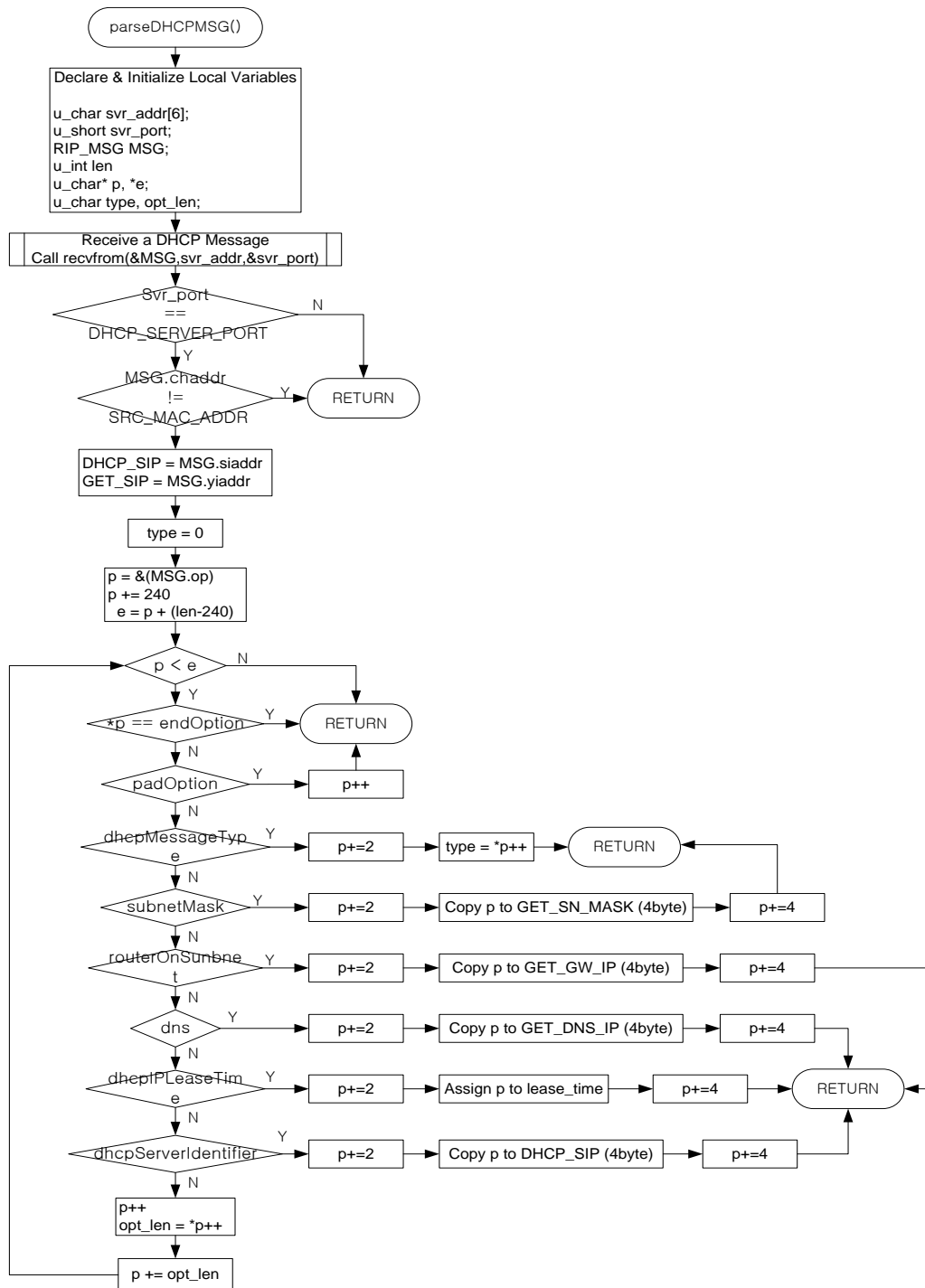
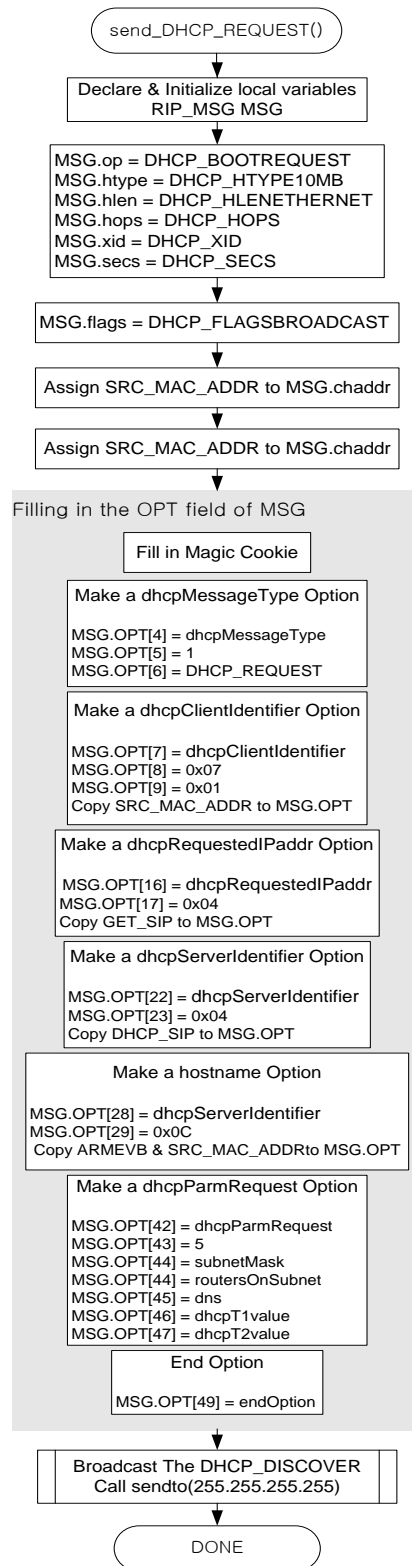Fig. 4 parseDHCPMSG() function flowchart

```
┌─────────────────────────┐
│   send_DHCP_REQUEST()   │
└─────────────────────────┘
              │
┌─────────────────────────────────┐
│  Declare & Initialize local     │
│  variables RIP_MSG MSG          │
└─────────────────────────────────┘
              │
┌─────────────────────────────────┐
│ MSG.op = DHCP_BOOTREQUEST       │
│ MSG.htype = DHCP_HTYPE10MB      │
│ MSG.hlen = DHCP_HLENETHERNET    │
│ MSG.hops = DHCP_HOPS            │
│ MSG.xid = DHCP_XID              │
│ MSG.secs = DHCP_SECS            │
└─────────────────────────────────┘
              │
┌─────────────────────────────────┐
│ MSG.flags = DHCP_FLAGSBROADCAST │
└─────────────────────────────────┘
              │
┌─────────────────────────────────┐
│ Assign SRC_MAC_ADDR to MSG.chaddr│
└─────────────────────────────────┘
              │
┌─────────────────────────────────┐
│ Assign SRC_MAC_ADDR to MSG.chaddr│
└─────────────────────────────────┘
              │
```

Filling in the OPT field of MSG

Fill in Magic Cookie

Make a dhcpMessageType Option
MSG.OPT[4] = dhcpMessageType
MSG.OPT[5] = 1
MSG.OPT[6] = DHCP_REQUEST

Make a dhcpClientIdentifier Option
MSG.OPT[7] = dhcpClientIdentifier
MSG.OPT[8] = 0x07
MSG.OPT[9] = 0x01
Copy SRC_MAC_ADDR to MSG.OPT

Make a dhcpRequestedIPaddr Option
MSG.OPT[16] = dhcpRequestedIPaddr
MSG.OPT[17] = 0x04
Copy GET_SIP to MSG.OPT

Make a dhcpServerIdentifier Option
MSG.OPT[22] = dhcpServerIdentifier
MSG.OPT[23] = 0x04
Copy DHCP_SIP to MSG.OPT

Make a hostname Option
MSG.OPT[28] = dhcpServerIdentifier
MSG.OPT[29] = 0x0C
Copy ARMEVB & SRC_MAC_ADDRto MSG.OPT

Make a dhcpParmRequest Option
MSG.OPT[42] = dhcpParmRequest
MSG.OPT[43] = 5
MSG.OPT[44] = subnetMask
MSG.OPT[44] = routersOnSubnet
MSG.OPT[45] = dns
MSG.OPT[46] = dhcpT1value
MSG.OPT[47] = dhcpT2value

End Option
MSG.OPT[49] = endOption

```
┌─────────────────────────────────┐
│  Broadcast The DHCP_DISCOVER    │
│  Call sendto(255.255.255.255)   │
└─────────────────────────────────┘
              │
      ┌───────────────┐
      │     DONE      │
      └───────────────┘
```

Fig. 5 send_DHCP_REQUEST() funciton flowchart

Document History Information

Version	Date	Descriptions
Ver. 0.9ßeta	Sep, 2009	Release with W7100 launching
Ver. 0.91	Feb, 2011	Modify the source code and fig. 2
Ver. 1.0	Mar, 2011	Modified for W7100A QFN 64pin package

Copyright Notice