How to implement IPRAW for W7100

Version 1.2

Table of Contents

Ver. 1.1   2

1   Introduction

IPRAW is the type of data communication that uses TCP and IP layer, which is the lower protocol layer of UDP. Fig.1 shows the encapsulation process of data as it goes down the protocol stack. W7100 is embedded a Hardwired TCP/IP that is structured from Link layer to Transport later, excluding the Application Layer in OSI 4 layers. The W7100 supports IPRAW mode for data processing in IP layer protocols like ICMP (0x01) and IGMP (0x02) according to the protocol number. But if user needs, the host can directly process the IPRAW by opening the SOCKET n to IPRAW. This application note described ICMP, one of the IP Layer's upper protocol, and how it is used as a simple Ping application.

<Fig.1> Encapsulation of data as it goes down the protocol stack

2   IPRAW SOCKET

The W7100 supports up to eight independent SOCKETs simultaneously and can use all SOCKETs in IPRAW mode. Before creating of SOCKETn (the n-1 th SOCKET) in IPRAW mode, it must be configured which protocol of the IP Layer (protocol number) is going to be used. The protocol configuration of protocol is set by using SOCKET n protocol register (Sn_PROTO).

Protocol	Number	Semantic	W7100 Support
-	0	Reserved	O
ICMP	1	Internet Control Message Protocol	O
IGMP	2	Internet Group Management Protocol	O
TCP	6	Transmission Control Protocol	X
EGP	8	Exterior Gateway Protocol	O
UDP	17	User Datagram Protocol	X
Others	-	Another Protocols	O

Table 2 Key Protocol in IP layer

Tab.2 shows the key protocol in IP layer. Since TCP (0x06) and UDP (0x11) are already embedded in W7100, these protocol numbers are not supported when using IPRAW mode. The SOCKET communication of IPRAW mode is allowed only between assigned protocol numbers. The ICMP SOCKET cannot receive not assigned protocol data except assigned protocol data such as IGMP. After initialization of W7100, the Ping Reply is processed automatically. However, be aware that the Hardwired Ping Reply Logic is disabled if ICMP is opened as SOCKET n in IPRAW mode,

The structure of IPRAW data is as below. The IPRAW data is consisted of a 6bytes PACKET-INFO and a DATA packet. The PACKET-INFO contains information of transmitter (IP address) and the length of DATA-packet. The data reception of IPRAW is the same as UDP data reception, except processing the port number of transmitter in UDP PACKET-INFO. (For more details, refer to "9.2.2.1 Unicast & Broadcast" of the W7100 datasheet.) If the size of the transmitted DATA is larger than RX memory free size of SOCKET n, user can't receive neither the transmitted DATA nor the fragmented DATA.

<Fig.2> The received IPRAW data format

The lifecycle of SOCKET in IPRAW mode is composed OPEN, SEND, RECEIVE, and CLOSE. The lifecycle of SOCKET is explained in the next sections.

2.1 OPEN

First, Specify SOCKET number to 's' and set the protocol number to Sn_PROTO. Open the SOCKETn with IPRAW mode by calling socket(). And then wait until the Sn_SR is changed to SOCK_IPRAW. Sn_SR is checked by calling getSn_SR(). When Sn_SR is changed to SOCK_IPRAW(0x32), The SOCKETn OPEN is completed.

```
/* sets Protocol Number */
s = 0; // set SOCKET 0 (From 0 to 7)
IINCHIP_WRITE(Sn_PROTO(s), IPPROTO_ICMP);
/* OPEN SOCKETn */
socket(s, Sn_MR_IPRAW, port, mode);
while(getSn_SR(s) != SOCK_IPRAW);
```

Example 2.1 OPEN Socket

2.2 SEND

The data_buf is send to the destination address (add) by using the sento(). The SOCKETs opened in the IPRAW mode and the specify port is used.

```
/* Send Ping-Request to the specified peer. */
// max_size_tx_buf must be smaller than the maximum size of the TX buffer
* data_buf[max_size_tx_buf] = (uint8 *)0x7000; // set position of data buffer
 sendto(s,(uint8 *)&data_buf,sizeof(data_buf),addr,port)
```

Example 2.2 SEND DATA

2.3 RECEIVE

User can easily get the received data using the recvfrom() function in our W7100A driver. The SOCKETs opened in the IPRAW mode and the specify port is used.

```
/* Check received data */
//rlen indicates the received data size in the RX buffer.
//rlen must be smaller than the maximum size of the RX buffer
if ( (rlen = getSn_RX_RSR(s) ) > 0)
/* Received data : len is a length included the PACKET-INFO and the DATA packet.*/
len = (recvfrom(s, (uint8 *)data_buf,rlen,addr,&port);
```
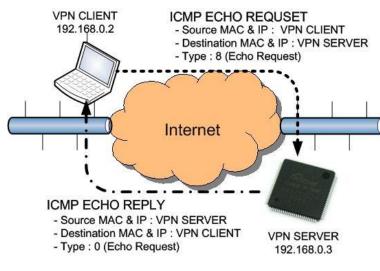
Example 2.3 RECEIVE DATA

2.4 CLOSE

No anymore need of IPRAW SOCKETs, extinct the SOCKETs by calling close();

3   ICMP (Internet Control Message Protocol)

<Fig.3> ICMP ECHO REQUSET/REPLY

Ver. 1.1   5

ICMP Echos are used mostly for troubleshooting. When a problem exists in the process of two hosts communicating to one another, a few simple ICMP Echo requests show whether the two hosts have their TCP/IP stacks configured correctly or not.

Fig.3 shows the very well known 'ping' command. In the case of ICMP Echo Request (ping) Packet, the Type field takes a value of 8. In the case of ICMP Echo Reply (ping reply) Packet, the Type field takes a value of 1. Tab.3.1 and Tab.3.2 shows, respectively, the message format and the message type of ICMP

1Byte	1Byte
Type	Code
Check Sum	
Type dependant	
Data	

Type	Semantic
0	Echo Reply
3	Destination Unreachable
4	Source Quench
5	Redirect
8	Echo Request
11	Time Exceeded
12	Parameter Problem
13	Timestamp
14	Timestamp Reply
15	Information Request
16	Information Reply

Table 3.1 ICMP Message Format   Table 3.2 ICMP Message Type

As shown in Fig.3, when the "Ping" command is executed, the source (VPN client) sends Ping Echo Request to the Destination (VPN server). Then, the Destination responds to the Ping Echo Request from the Source. The Ping Echo Reply has the same properties (like ID, Sequence Number, and data) as the Ping Echo Request. Therefore, the source can confirm the connection of a specific destination by comparing the Ping Echo Reply's properties with the Ping Echo Reply's properties.

3.1   Ping Implements

Ping Message Format is shown in Tab.3.1.1 The Type field of the Ping Message takes only the value of 8 or 0. The Code Field of the Ping Message takes the only one vale 0. The Ping Message is consisted with 4bytes of type filed, 2bytes of check sum, 2bytes of ID, and 2bytes of sequence number. The Ping data is filled up with the data field of variable length.

1Byte	1Byte
8 (0)	0
Check Sum	
ID	
Sequence Number	
Ping Data	

Table 3.1.1 Ping Message Format

To design the ping message easily, the pingmsg structure is defined as below in Example 3.1.

```c
#define BUF_LEN 32
#define PING_REQUEST 8
#define PING_REPLY 0
#define CODE_ZERO 0


typedef struct pingmsg
{
  uint8   Type;        // 0 - Ping Reply, 8 - Ping Request
  uint8   Code;        // Always 0
  int16   CheckSum;    // Check sum
  int16   ID;          // Identification
  int16   SeqNum;      // Sequence Number
  int8  Data[BUF_LEN]; // Ping Data   : 1452 = IP RAW MTU -sizeof(Type+Code+CheckSum+ID+SeqNum)
} PINGMSGR;
```

Example 3.1 Ping Message structure

Ping Application can be designed by using UDP related Application Programming Interfaces (API) from Socket API of the W7100 driver program. Tab.3.1.2 shows the Socket API functions.

API Function Name	Semantic
Socket	Open socket with IPRAW Mode
Sendto	Send Ping Request to Peer
Recvfrom	Receive Ping Reply from Peer
Close	Close Socket

Table 3.1.2 Socket API functions

The designed Ping Application sets the number of transmission of the Ping Reply, Ping Request, Destination IP Address, and Port Number as the parameter. Then, the user can ask a specific peer to send the specific number of Ping Requests and receive the Ping Reply.

uint8 ping(SOCKET s, uint16 count, uint8 *addr, uint16 port)

Function Name	Ping	
Arguments	s	– socket number
	count	- counter for Ping Request (Default = 3)
	addr	- Peer IP Address
	port	– send/receive port number

Table 3.1.3 ping function

uint8 ping_request(SOCKET s, uint8 *addr, uint16 port)

Function Name	ping_request	
Arguments	s	– socket number
	addr	- Peer IP Address
	port	– send/receive port number

Table 3.1.4 ping function

uint8 ping_reply (SOCKET s, uint8 *addr, uint16 port, uint16 len)

Function Name	ping_reply	
Arguments	s	– socket number
	addr	- Peer IP Address
	port	– send/receive port number
	len	- packet length

Table 3.1.5 ping function

uint16 checksum(uint8 * data_buf, uint16 len)
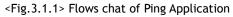
Function Name	Checksum	
Arguments	data_buf	- ping message
	len	- ping message length

Table 3.1.6 ping function

Fig.3.1.1 shows the flow chart of a simple Ping Application. The Ping Application process is divided into the calculation of checksum, the Ping Request process, and the Ping Reply which are designed functions at section 3.1

<Fig.3.1.1> Flows chat of Ping Application
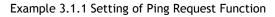
Ver. 1.1   9

● Calling Ping Function

The Ping Application Function requires the destination IP address and a specific number of ping requests and the Ping Request Function is called after the initialization and network configuration of W7100. Example 3.1.1 shows the process of setting Ping Application Function.

```c
/* main.c */
/* setting of Destination IP address & Port */
  pDestaddr[4]= {192,168,1,3};
  pPort = 3000;
/* set ping request count */
  pCount = 3;
/* Calling ping_request function */
/* pring_request( SOCKETn, COUNT, DESTINATION_IP, PORT) */
  ping (0, pCount, &pDestaddr, pPort);
```

Example 3.1.1 Setting of Ping Request Function

● Ping Request

The process of the ping request executes for making header and data packets, setting the protocol, and sending data packets to the target. The Ping Request Processing is shown in Example 3.1.2 The Checksum() is executed after making header and data. The ping request is then sent to the Host PC by using the SOKET which is create in IPRAW and is defined ICMP.

```c
/* ping_request.c */
/* make header of the ping-request   */
  PingRequest.Type = PING_REQUEST;     // Ping-Request
  PingRequest.Code = CODE_ZERO;        // Always '0'
  PingRequest.ID = htons(RandomID++); // set ping-request's ID to random integer value
  PingRequest.SeqNum = htons(RandomSeqNum++);
                         // set ping-request's sequence number to ramdom integer value
/* Do checksum of Ping Request */
  PingRequest.CheckSum = 0;
  PingRequest.CheckSum = htons(checksum((uint8*)&PingRequest,sizeof(PingRequest)));
        :
/* set ICMP Protocol */
  IINCHIP_WRITE(Sn_PROTO(s), IPPROTO_ICMP);
/* open the SOCKET with IPRAW mode */
  socket(s,Sn_MR_IPRAW,port,0) ;
/* sendto ping_request to destination */
  sendto(s,(uint8 *)&PingRequest,sizeof(PingRequest),addr,port);
```
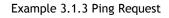
Example 3.1.2 Ping Request

- Ping Reply

Example 3.1.3 shows the ping reply processing. The reply request processing receives the data the MACRAW mode by using of RECEVFROM function. Check if the type of the received data is set to Ping_Reply (0). If that is the case, the ping message will be displayed.

```c
/* ping.c */
/* receive data from a destination */
  rlen = recvfrom(s, (uint8 *)&PingReply,rlen,addr,&port);
/* check the Type */
  if(PingReply.Type == PING_REPLY) {
      /* check Checksum of Ping Reply */
      tmp_checksum = PingReply.CheckSum;
      PingReply.CheckSum = 0;
      PingReply.CheckSum = checksum((uint8*)&PingReply,sizeof(PingReply));
          :
      /* Output the Destination IP and the size of the Ping Reply Message*/
          :
  }else{
      printf (" Unknown msg. \n");
  }
```

Example 3.1.3 Ping Request

3.2 Ping Request Demonstration

After downloading the binary file of the Ping application, confirm the package of iMCUW7100API for the demonstration in the order shown below. (For more information, refer to iMCUW7100 ISP User's guide or iMCUW7100 Debugger User's guide)

- Confirm the testing environment. Refer to iMCU7100EVB Manual
  Connect test PC to iMCUW7100EVB by directly using UTP cable.
  Connect test PC to iMCUW7100EVB by directly using Serial able.
  Connect 5V power adaptor to test PC
- Confirm the network information of Test PC as the following
  Source IP Address : 192.168.1.3 (It's up to test PC)
  Gatway IP Address : 192.168.1.1
  Subnet Mask : 255.255.255.0

- After executing serial terminal program (ex: HyperTerminal), set up the properties as followed,

Properties	Setting Value
Bits Per second (Baud Rate)	115200 bps (Max 230400bps)
Data Bits	8 Bits
Stop Bits	1 Bits
Parity	No
Flow Control	None

Table13 Setting of Terminal program

- Turn on the power switch of iMCUW7100API.
  - Check lighting on power LED (D13) of iMCUW7100API when powering on.

Fig.3.2 shows the execution results of a Ping Application. The results show the network information of W7100 (local host) and the ping reply which is responded from peer host.

```
=============================================
        W7100    Net Config Information
=============================================
MAC ADDRESS IP : 00.08.dc.00.00.00
SUBNET MASK   : 255.255.255.0
G/W IP ADDRESS : 192.168.1.1
LOCAL IP ADDRESS : 192.168.1.2


-------PING_TEST_START--------
Destination IP   : 192.168.1.3
No.0
Send Ping Request   to Destination (192.168.1.3 ) ID:1234   SeqNum:4321 CheckSum:726a
Reply from 192.192.192.192   ID:1234 SeqNum:4321   :data size 3072 bytes
No.1
Send Ping Request   to Destination (192.168.1.3 ) ID:1235   SeqNum:4322 CheckSum:7268
Reply from 192.192.192.192   ID:1235 SeqNum:4322   :data size 3072 bytes
No.2
Send Ping Request   to Destination (192.168.1.3 ) ID:1236   SeqNum:4323 CheckSum:7266
Reply from 192.192.192.192   ID:1236 SeqNum:4323   :data size 3072 bytes
-------PING_TEST_END--------
```

<Fig.3.2> Execution result of Ping Request

Document History Information

Version	Date	Descriptions
Ver. 0.9	2009	Release with W7100 launching
Ver. 1.0	Mar, 2011	Modify for W7100A QFN 64pin package
Ver. 1.1	Jun, 2012	Fixed some awkward expressions of English documents

Copyright Notice