IPRAW Application Note for W5100S

Version 1.0.0

For more information, visit our website at http://www.wiznet.io

Table of Contents

1 IPRAW Introduction 3
2 IPRAW SOCKET 4
2.1 IPRAW Life Cycle 5
2.1.1 OPEN 5
2.1.2 SEND 5
2.1.3 RECEIVE 5
2.1.4 CLOSE 6
3 IPRAW Application Example 7
3.1 ICMP (Internet Control Message Protocol) Echo 7
3.2 Ping Implementation 8
4 Document History Information 14

1 IPRAW Introduction

IPRAW mode enables processing of protocols higher than IP layer among TCP/IP layer. **Figure 1** shows the data encapsulation process where application data is transferred to each lower layer. W5100S IPRAW mode supports protocols such as ICMP (0x01) according to the number specified in the Protocol field in the IP header. In W5100S, some functions of ICMP are already implemented as hardwired, but user can open the n-th socket of W5100S in IPRAW mode according to need and implement other functions and protocols of ICMP directly by software.

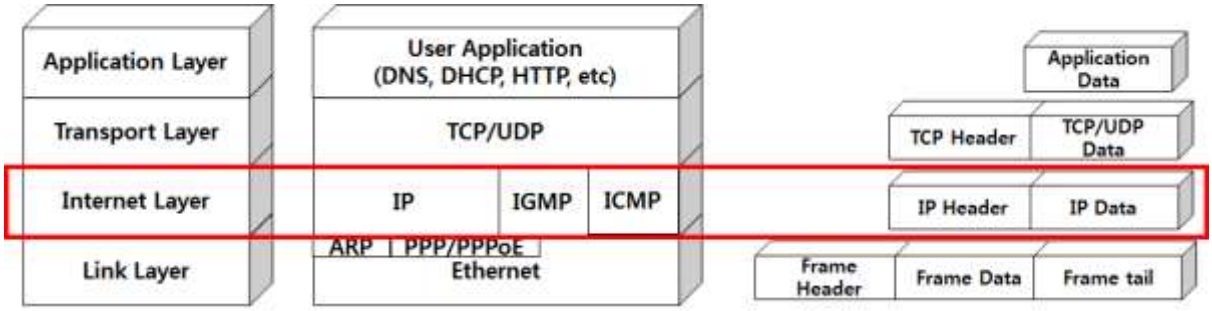

Figure 1 Encapsulation of data as it goes down the protocol stack

note

If IPRAW is used, only one socket per protocol should be used, and if several sockets are used, socket number is given priority in descending order. Therefore, you can not use multiple sockets with the same protocol.

You can not use UDP, TCP, IGMP, and IPv6 related protocols in IPRAW mode because that protocols can only be used as hardware.

2 IPRAW SOCKET

W5100S supports 4 SOCKETs, and all SOCKETs support IPRAW mode. When using SOCKETn (n-th SOCKET) in IPRAW mode, you must set the protocol number field of the IP header. For protocol number, it must be set to SOCKET n protocol register (Sn_PROTO) before socket open command.

Table 1 Key Protocol in IP layer

Protocol	Number	Semantic	W5100S Support
HOPOPT	0	Reserved	X
ICMP	1	Internet Control Message Protocol	O
IGMP	2	Internet Group Management Protocol	X
IPv4	4	IPv4 encapsulation	O
TCP	6	Transmission Control Protocol	X
UDP	17	User Datagram Protocol	X
IPv6	41	IPv6 encapsulation	X
Others	-	Another Protocols(not related to IPv6)	O

Table 1 shows the IP layer protocols. Sockets opened in IPRAW mode do not support TCP (0x06) or UDP (0x11). Also, IPRAW mode socket set to ICMP protocol can not receive data of other than ICMP protocol. when After initialization of W5100S, W5100S is automatically processes the ping reply to the ping request. however, that if IPRAW SOCKET n is Opened with the ICMP protocol, the Hardwired Ping Reply Logic is disabled.

The structure of IPRAW Data is shown in Figure 2. IPRAW data consists of 6 bytes of Packet Information and Data packet. Packet information contains the information of sender(IP address) and length of Data packet. Data reception in IPRAW mode is the same as UDP data reception except port number processing of the sender in Packet Information of UDP.

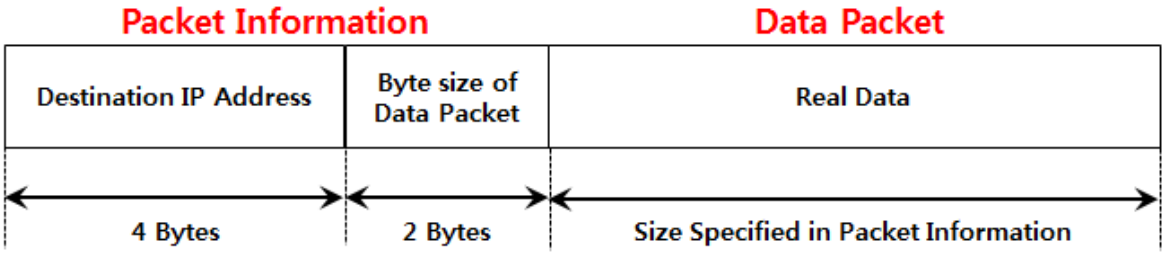

Figure 2 received IPRAW data format

2.1 IPRAW Life Cycle

IPRAW SOCKET Lifecycle consists OPEN, SEND, RECEIVE, CLOSE. Let's take a look at the lifecycle and implementation method of IPRAW SOCKET through ping application example.

2.1.1 OPEN

Select the s socket number as s, set the Protocol number to ICMP in Sn_PROTO, and open SOCKET n set in IPRAW mode using socket function. If Sn_SR is checked changed to SOCK_IPRAW (0x32), OPEN of SOCKET n is completed.

```
/* Create Socket */
IINCHIP_WRITE(Sn_PROTO(s), IPPROTO_ICMP);      // set ICMP Protocol
if(socket(s,Sn_MR_IPRAW,port,0)!=s){ // open the SOCKET with IPRAW mode, if fail then Error
printf( "\r\n socket %d fail r\n",     (s)) ;
}
/* Check socket register */
while(getSn_SR(s)!=SOCK_IPRAW);
```

Example 1 Socket Open

2.1.2 SEND

Sendto function is used to send the information stored in the Ping Request to the destination address. Use Socket configured in IPRAW mode.

```
/* sendto ping_request to destination */
// Send Ping-Request to the specified peer.
if(sendto(s,(uint8_t *)&PingRequest,sizeof(PingRequest),addr,port)==0){
printf( "\r\n Fail to send ping-reply packet   r\n") ;
}
```

Example 2 Send Data

2.1.3 RECEIVE

Data received from the destination address is stored in the data_buf using the recvfrom function. Use Socket configured in IPRAW mode.

```
if ( (rlen = getSn_RX_RSR(s) ) > 0){
        /* receive data from a destination */
len = recvfrom(s, (uint8_t *)data_buf,rlen,addr,&port);
}
```

Example 3 Receive Data

2.1.4 CLOSE

You can use the close function if you do not need the IPRAW socket anymore.

```
close(s);
```

Example 4 Close Socket

3 IPRAW Application Example

Let's implement the ICMP protocol echo request and echo reply as the IPRAW Application Example.

3.1 ICMP (Internet Control Message Protocol) Echo

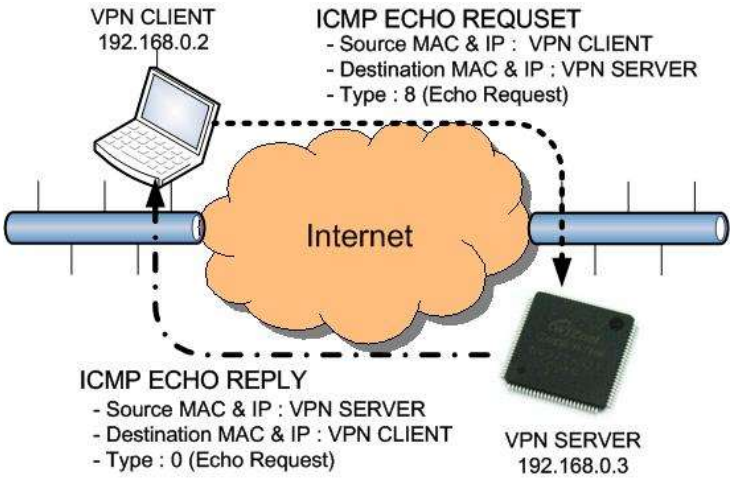

Figure 3 ICMP ECHO REQUSET/REPLY

ICMP Echo Message is called a PING packet and It is mainly used for troubleshooting. If there are two hosts with communication problems, it is possible to know whether the configuration of TCP/IP stack of two hosts is correct by PING packet. Figure 3 shows the request / reply process of PING packet. In case of PING Request Packet, Type filed has a value of 8, and in case of PING Reply packet, it has a value of 0. Table 2 and Table 3 show the message format and message type, respectively.

Table 2 ICMP Message Format

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 ICMP Message Type

1 Byte	1 Byte
Type	Code
Check Sum	
Type dependent	
Data	

When the PING command is executed, the source (VPN client) sends a ping request packet for the destination (VPN server) as shown in Figure 3. destination receiving the request packet sends ping reply to the source. The PING Reply consists of the same ID, same Sequence Number, and same Data as the PING Request. Therefore, the source can confirm the connection with the specific destination by comparing the PING Reply received from the Destination with the ID, Sequence Number, and Data of the PING Request.

3.2 Ping Implementation

ICMP Type field of the Ping Message has '0' (Ping Reply) or '8' (Ping Request), and the code field has only '0'. The Check Sum, ID, and Sequence Number fields are each 2 bytes. Ping data has a variable length. Table 4 shows the Ping message format.

Table 4 Ping Message Format

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

We used a structure to easily implement Ping Message, which we defined in Example 5.

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

typedef struct pingmsg
{
   uint8_t   Type;          // 0 - Ping Reply, 8 - Ping Request
   uint8_t   Code;          // Always 0
   int16_t   CheckSum;      // Check sum
   int16_t   ID;            // Identification
   int16_t   SeqNum;        // Sequence Number
   int8_t    Data[BUF_LEN];// Ping Data   : 1452 = IP RAW MTU - sizeof(Type+Code+CheckSum+ID+SeqNum)
} PINGMSGR;
```

Example 5 Ping Message Structure

Ping application can be implemented using ioLibrary's Socket API mentioned in Table 5.

Table 5 Socket API Functions

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

The designed Ping Application sets the Destination IP Address as the parameter. Also use request_flag parameter to set whether peer sends Ping Request or W5100S sends Ping Request. If request_flag is 1, W5100S send Ping Request to peer and receive Ping Reply. And check CheckSum and SeqNum(Sequence Number) to ensure that it is correct Ping Reply. If request_flag is 0, peer send Ping Request to Ethernet Chip and receive Ping Reply. Thiss Ping Application is considered the case of immediately receiving Ping Reply after sending Ping Request from W5100S. So if W5100S does not receive Ping Reply and sends Ping Request consecutively, it may not perform properly. In order to solve such a problem, user should directly modify the Ping Application.

uint8 ping_auto(SOCKET s, uint8 *addr, uint8_t request_flag)

Table 6 ping_auto function

Function Name	ping_auto
Arguments	s - socket number addr - Peer IP Address request_flag - ping mode

uint8 ping_request(SOCKET s, uint8 *addr)

Table 7 ping_request function

Function Name	ping_request
Arguments	s - socket number addr - Peer IP Address

uint8 ping_reply (SOCKET s, uint8_t *addr, uint16_t len, uint8_t request_flag)

Table 8 ping_reply function

Function Name	ping_reply
Arguments	s - socket number addr - Peer IP Address len - packet length request_flag - ping mode

uint16 checksum(uint8 * data_buf, uint16 len)

Table 9 checksum function

Function Name	Checksum
Arguments	data_buf - ping message len - ping message length

Figure 4 shows the flow of a simple ping application. Ping Application pocess is divided into calculation of CheckSum, Ping Request process, and Ping Reply.

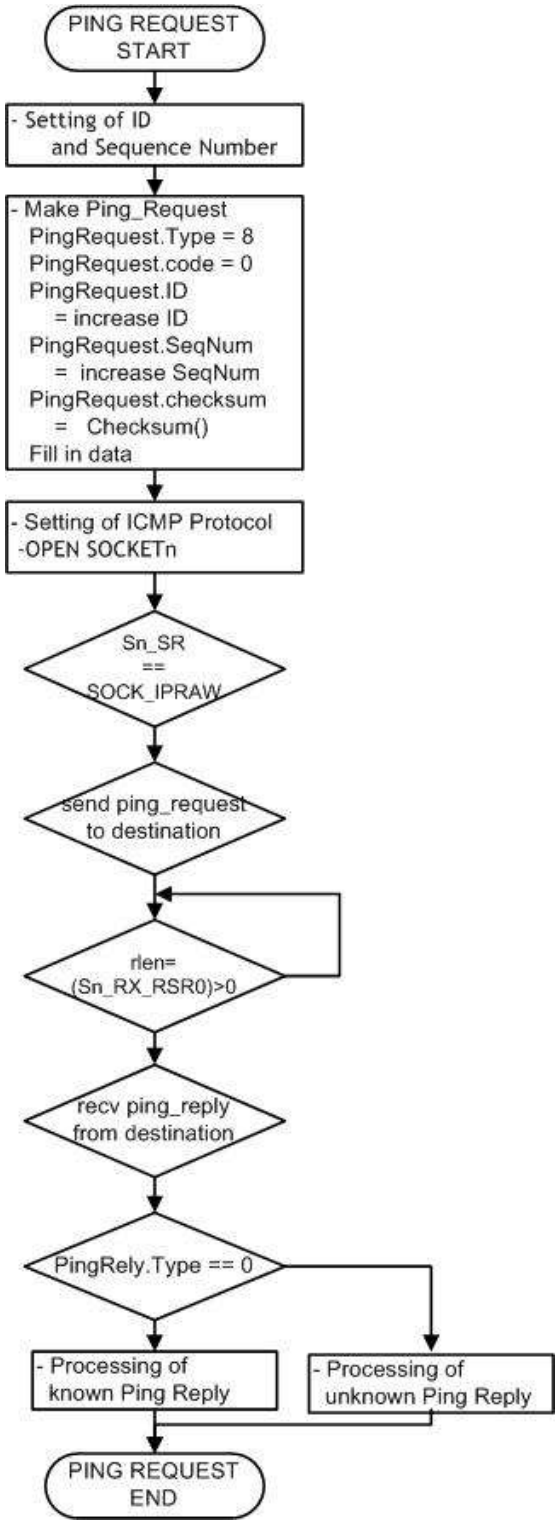

Figure 4 Flow chart of Ping Application

- Calling Ping Function

Ping Application Function requires the destination IP address and the Ping Request Function is called after the initialization and network configuration of W5100S. Example 6 shows the process of setting Ping Application Function.

```
/* main.c */
/* setting of Destination IP address */
pDestaddr[4]= {192,168,0,2};
/* Control Ethernet chip(W5100S) mode of request or reply*/
//request_flag = 0;       //Send request ping from outside to Ethernet Chip(W5100S)
request_flag = 1;       //Send request ping from Ethernet Chip(W5100S) to outside
/* Calling ping_request function */
ping_auto(0,pDestaddr, request_flag);
```

Example 6 Setting of Ping Request Function

- Ping Request

Process of the Ping Request executes for making header and data packets, setting protocol, and sending data packets to the target. Ping Request processing is shown in Example 7. Checksum is executed after making header and data. Ping Request is then sent to Host PC by using SOCKET which is create in IPRAW and is defined ICMP.

Set Protocol to ICMP and open SOCKET to IPRAW mode, and send Ping Request to IPRAW mode using sendto function.

```
/* 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
 random 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);
socket(s,Sn_MR_IPRAW,3000,0) ; /* open the SOCKET with IPRAW mode */
/* send ping_request to destination */
sendto(s,(uint8 *)&PingRequest,sizeof(PingRequest),addr,3000);
```

Example 7 Ping Request

- Ping Reply

Example 8 shows the Ping Reply processing. if the type of the received data is set to '0' Ping

Reply information message will be displayed.

```
/* ping.c */
/* receive data from a destination */
len = recvfrom(s, (uint8 *)data_buf, rlen, addr,(int16_t*)destport);
/* check the Type */
if(data_buf[0]== PING_REPLY)
{
   printf("PING_REPLY\r\n");
   PingReply.Type                    = data_buf[0];
   PingReply.Code                    = data_buf[1];
   PingReply.CheckSum       = (data_buf[2]<<8) + data_buf[3];
   PingReply.ID             = (data_buf[5]<<8) + data_buf[4];
   PingReply.SeqNum         = (data_buf[7]<<8) + data_buf[6];

         :
/* check Checksum of Ping Reply */
   tmp_checksum = ~checksum(&data_buf,len);
     :
     :

if(PingRequest.SeqNum == PingReply.SeqNum)
{
   if(tmp_checksum != 0xffff)
   printf("tmp_checksum = %x\r\n",tmp_checksum);
   else
   {
   /*   Compare Checksum of Ping Reply and Ping Request */
        if(comp_request_checksum == comp_reply_checksum)
        {
         /*   Output the Destination IP and the size of the Ping Reply Message   */
         printf("Reply from %d.%d.%d.%d   ID:%x SeqNum:%x   :data size %d bytes
           CheckSum %x\r\n",(addr[0]),   (addr[1]),   (addr[2]),   (addr[3]),   htons(PingReply.ID),
           htons(PingReply.SeqNum),   (rlen+6), PingReply.CheckSum );
         printf("\r\n");

         /*   SET ping_reply_receiver to '1' and go out the while_loop (waiting for ping reply)   */
         ping_reply_received =1;
         }
        :
}else{
       printf(" Unknown msg. \n");
}
```

Example 8 Ping Reply

4 Document History Information

Version	Date	Descriptions
Ver. 1.0.0	A 2018	Release

Technical Support: forum.wiznet.io or support@wiznet.io
Sales & Distribution: sales@wiznet.io

For more information, visit our website at www.wiznet.io