IPRAW Application Note for W5100S

Version 1.1.0

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
3.2.1 Calling PING Function 13
3.2.2 PING Request 13
3.2.3 PING Reply 14
4 Document History Information 15

1 IPRAW Introduction

W5100S provides IPRAW Mode which allows the user to directly process the Protocol above the IP Layer. SOCKET, which is OPEN with IPRAW Mode, stores IP Layer higher information on the TCP/IP Layer in SOCKET TX / RX buffer so that the HOST can process it directly. **Figure 1** shows the Data encapsulation process.

HOST can set the Protocol Field value of IP Header in IPRAW Mode, and can implement various IP Layer Protocol.

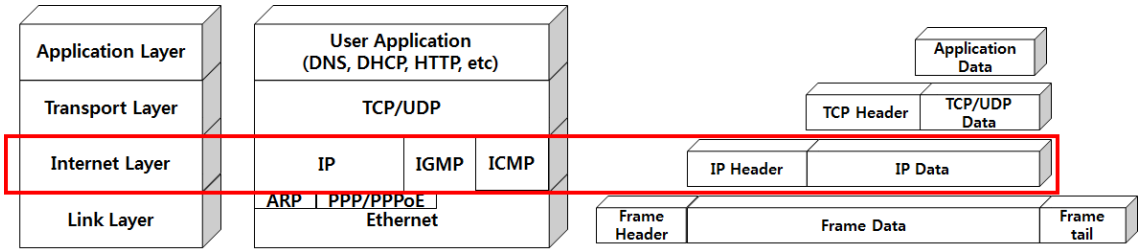

Figure 1 Encapsulation of Data as it goes down the Protocol stack

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 cannot use multiple SOCKETs with the same Protocol.

You cannot 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 can be set to IPRAW Mode. Before SOCKET OPEN in IPRAW MODE, IP Layer Protocol must be set by Sn_PROTO(SOCKET n Protocol Register).

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 Protocol supported by IPRAW Mode SOCKET. IPRAW Mode SOCKET does not support TCP (0x06), UDP (0x17), IPv6 (0x41) communication and does not support communication other than the Protocol set in Sn_PROTOR.

When set to Sn_PROTOR = ICMP (0x01), W5100S no longer supports Auto PING Reply for PING (Packet Internet Grouper) Request. Instead, the PING Request Packet is stored in IPRAW Mode SOCKET n RX Buffer Block.

The Data structure received in IPRAW Mode is shown in Figure 2. The received Data consists of 6 bytes of PACKET-INFO and Data, and PACKET-INFO contains the peer's information (IP address) and length of Data Packet.

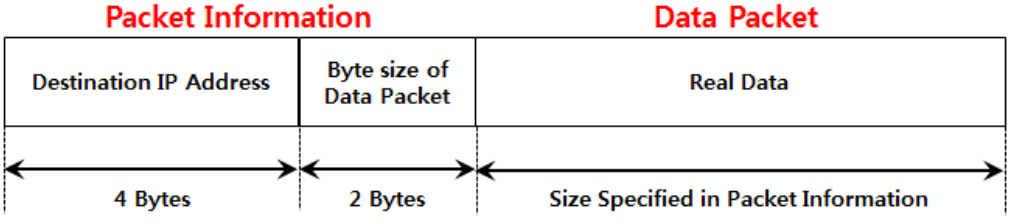

Figure 2 received IPRAW Data Format

2.1 IPRAW Life Cycle

IPRAW SOCKET Life Cycle consists OPEN, SEND, RECEIVE and CLOSE. The following shows the Life Cycle and implementation method of the IPRAW Mode SOCKET through the PING Application example.

2.1.1 OPEN

First, set SOCKET number in ‘s’ and ICMP Protocol number in Sn_PROTO. Then use socket() to open SOCKET n in IPRAW Mode. If Sn_SR is changed to SOCK_IPRAW(0x32), SOCKET n OPEN 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

Use sendto() Function to sned the information stored in PING Request to 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

Use recvfrom() Function to store the Data received from Destination Address in data_buf. 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

SOCKET is closed. Use when IPRAW Mode SOCKET is no longer needed.

```
close(s);
```

Example 4 CLOSE SOCKET

3 IPRAW Application Example

IPRAW Application example can be used to implement ICMP Echo Request and Reply, of ICMP (Internet Control Message Protocol).

3.1 ICMP (Internet Control Message Protocol) Echo

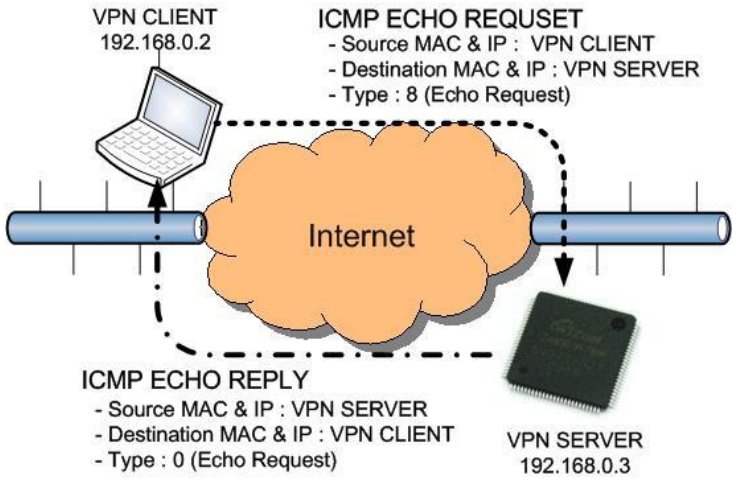

Figure 3 ICMP ECHO Request/Reply

ICMP is a Protocol that shares IP network status and error information, and used in PING. Among them, Echo Request and Echo Reply are called ICMP Echo Message, and used in PING Request Packet and PING Reply Packet. It is used to check whether the IP Datagram between devices is reached.

Figure 3 shows the Request/Reply process of the ICMP Echo Message. For Echo Request, the type field value is 8. And for Echo Reply, the value is 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

Table 3 ICMP Message Type

1 Byte	1 Byte
Type	Code
Check Sum	
Type dependent	
Data	

If 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 PING Request Packet sends PING Reply Packet to the source. The PING Reply Packet consists of the same ID, Sequence Number, and Data as the PING Request Packet. Therefore, the source can confirm the connection with the specific destination by comparing the PING Reply Packet received from the Destination with the ID, Sequence Number, and Data of the PING Request Packet.

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	

The struct was used to easily implement PING Message, and it was 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 PING Application provided as an example sets the SOCKET and Destination Address to use in IPRAW Mode as parameters. The designed PING Application consists of two Mode that W5100S sends PING Request Packet and W5100S receives PING Request Packet, use request_flag parameter to set the Mode.

If request_flag is '1', W5100S is set to Mode that SEND PING Request Packet. W5100S sends a specific number of PING Request Packet to Peer, and PING Reply Packet corresponding to each PING Request Packet. After, W5100S confirms using CheckSum and SeqNum(Sequence Number) that PING Reply Packet received from Peer is correct.

If request_flag is '0', W5100S is set to Mode that RECEIVE PING Request Packet. W5100S sends PING Reply Packet corresponding to each PING Request Packet received from Peer to Peer. Also, Confirms using CheckSum and SeqNum that PING Request Packet received from Peer is correct.

The Functions used mainly in the examples provided are as follows.

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

It is the main Function that performs all the operations of IPRAW PING Application, and it consists of PING_request(), PING_reply(), and etc.

uint8 PING_request(SOCKET s, uint8 *addr)

Table 7 PING_request Function

Function Name	PING_request	
Arguments	s	- SOCKET number
	addr	- Peer IP Address

This Function is used when request_flag is '1', and it is used to send PING Request Packet from W5100S to Peer.

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

It is a Function used when receiving PING Request Packet or PING Reply Packet from Peer. This Function differs in behavior according to request_flag.

uint16 checksum(uint8 * data_buf, uint16 len)

Table 9 checksum Function

Function Name	Checksum

Arguments	data_buf - PING Message len - PING Message length

This Function is called in PING_request() or PING_reply(), and it calculates CheckSum.

Figure 4 shows the simple flow chart of the PING Request process.

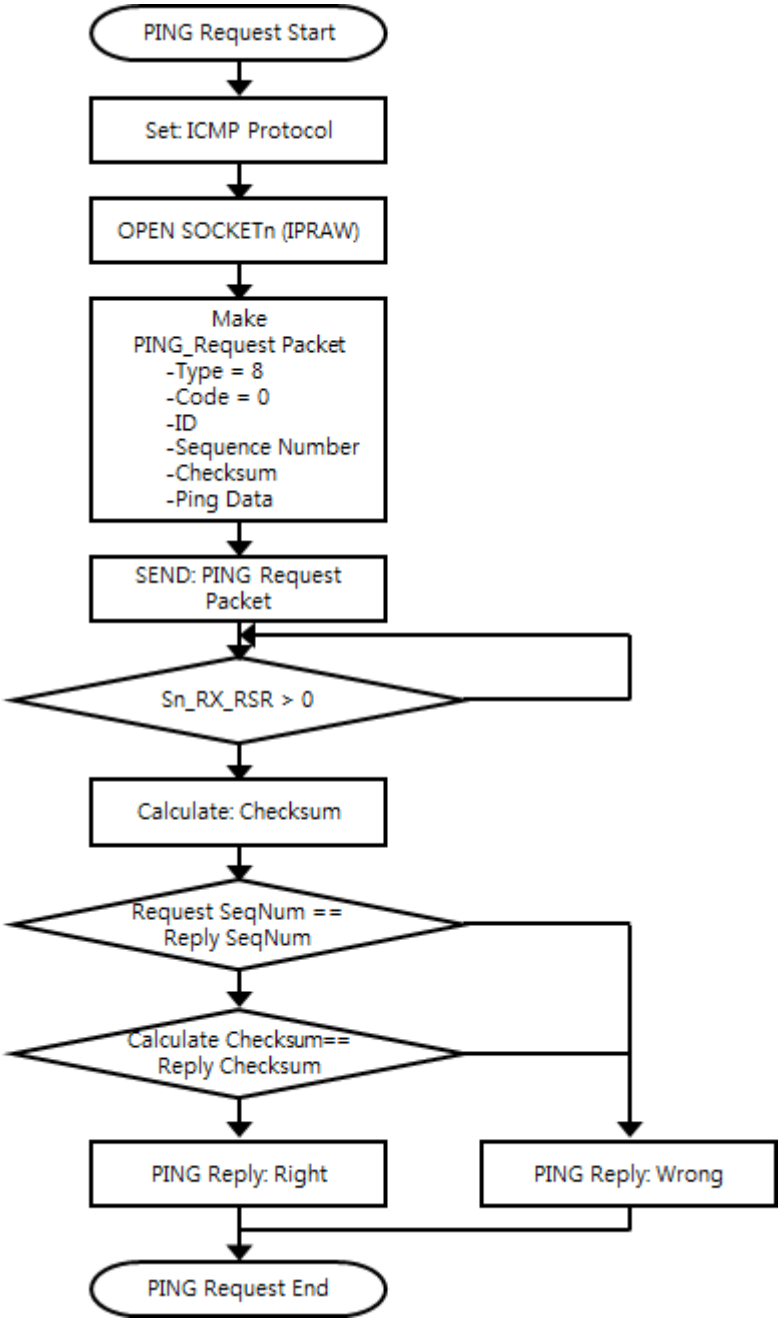

Figure 4 Flow chart of PING Application

3.2.1 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_auto Function */
PING_auto(0,pDestaddr, request_flag);
```

Example 6 Setting of PING Request Function

3.2.2 PING Request

To transmit PING Request, set the Protocol to ICMP Protocol, open IPRAW Mode SOCKET and store Header and Data of PING Request in buffer. Example 7 shows the process of PING Request.

Set Protocol to ICMP and OPEN SOCKET to IPRAW Mode. After making Header and Data of PING Request, calculate CheckSum, and SEND PING Request to Peer using sendto() Function.

```
/* PING.c */
/* set ICMP Protocol */
IINCHIP_WRITE(Sn_PROTO(s), IPPROTO_ICMP);
socket(s,Sn_MR_IPRAW,3000,0) ; /* OPEN the SOCKET with IPRAW Mode */

=============================================================================

/* 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)));
        :
/* SEND PING Request to destination */
sendto(s,(uint8 *)&PINGRequest,sizeof(PINGRequest),addr,3000);
```

Example 7 PING Request

3.2.3 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 */
else if(data_buf[0]== PING_REPLY)
{
   printf("PING_REPLY\r\n");
   /* for comp checksum */
   PINGReply.Type                    = data_buf[0];
   PINGReply.Code                    = data_buf[1];
   PINGReply.CheckSum       = (data_buf[3]<<8) + data_buf[2];
   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 = PINGReply.CheckSum;

/* Calculate CheckSum of PING Reply */
   PINGReply.CheckSum = 0;
   PINGReply.CheckSum = htons(checksum((uint8_t*)&PINGReply, sizeof(PINGReply)));

if (tmp_checksum != PINGReply.CheckSum)
{
    printf(" \r\n Request CheckSum is incorrect %x should be %x \r\n", (tmp_checksum),
            htons(PINGReply.CheckSum)) ;
}
else
{
 printf(" \r\n Request CheckSum is correct \r\n") ;
}
/*   End of comp checksum */

/*   For SEND PING Reply Packet */
            :
PINGReply.Type   =    0;

for(i=0; i<len-8 ; i++)
{
   PINGReply.Data[i] = data_buf[8+i];
}

/*   Calculate Checksum of PING Reply Packet */
PINGReply.CheckSum = 0;
PINGReply.CheckSum = htons(checksum((uint8_t*)&PINGReply, sizeof(PINGReply)));

send_rep = sendto(s, (uing8_t *)&PINGReply, sizeof(PINGReply), addr, 3000);
```

Example 8 PING Reply

4 Document History Information

Version	Date	Descriptions
Ver. 1.0.0	Apr, 2018	Release
Ver. 1.1.0	MAY, 2018	Modified Contents of 3.2 PING Implementation, Figure 1, Figure 4, PING Request Code and PING Reply Code. Edit typo.