

How to connect ADSL

Document History

Ver 1.0 (OCT 26, 2005)	First release for W3150A
Ver 2.0 (AUG 15, 2006)	Second release for W3150A+ Added figure and more information

© 2006 WIZnet Co., Inc. All Rights Reserved.

For more information, visit our website at <http://www.wiznet.co.kr>

This application note shows how to use W3150A+ under the PPPoE(ADSL) environment.

Below chart shows registers related to using W3150A+ in PPPoE(ADSL) conditions.

S0_CR (Socket 0 Command Register) [R/W] [0x0401] [0x00]

About PPPoE command

Value	Symbol	Description
0x23	PCON	Starts ADSL connection (starts PPPoE Discovery)
0x24	PDISCON	Ends ADSL connection
0x25	PCR	Sends REQ message in each Phase (For the detail of each phase, refer to below.)
0x26	PCN	Sends NAK message in each Phase
0x27	PCJ	Sends REJECT message in each Phase

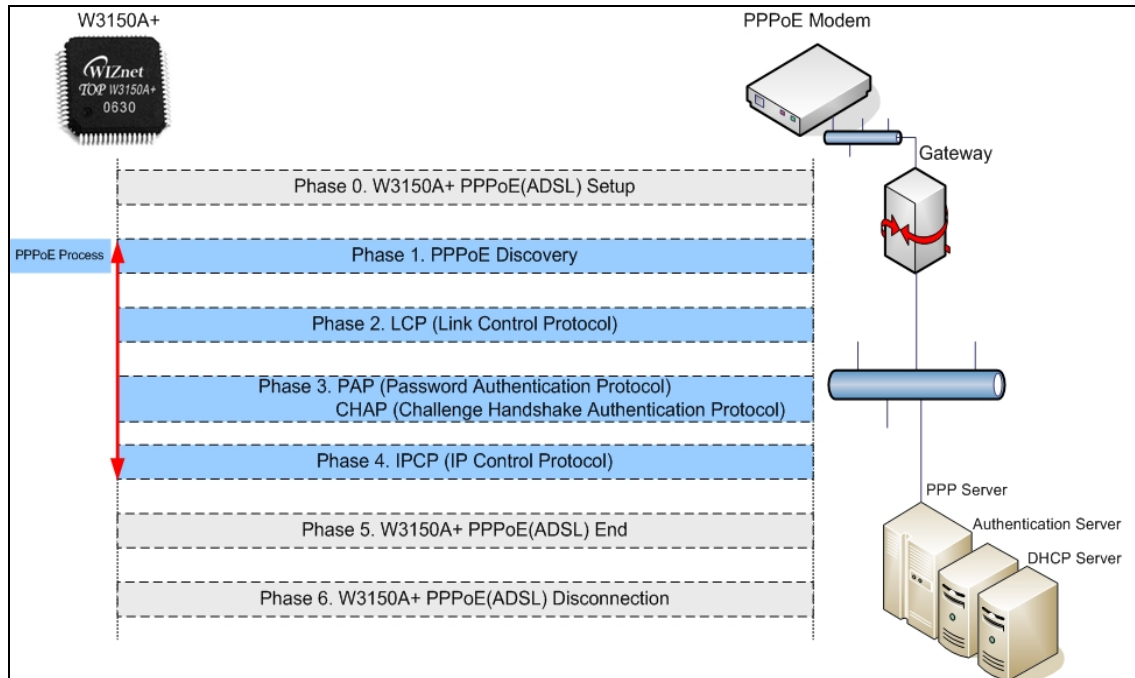
S0_IR (Socket n Interrupt Register) [R] [0x0402] [0x00]

About PPPoE Interrupt

7	6	5	4	3	2	1	0
PRECV	PFAIL	PNEXT	Reserved	TIMEOUT	RECV	DISCON	CON

Bit	Symbol	Description
7	PRECV	Indicates non-supporting option data is received
6	PFAIL	Indicates PAP Authentication Fail
5	PNEXT	Go to the next phase (For the detail of each phase, refer to the below.)
4	Reserved	Refer to the W3150A+ Datasheet.
3	TIMEOUT	
2	RECV	
1	DISCON	
0	CON	

Below figure shows the PPPoE(ADSL) connection process of W3150A+.



1. Phase 0

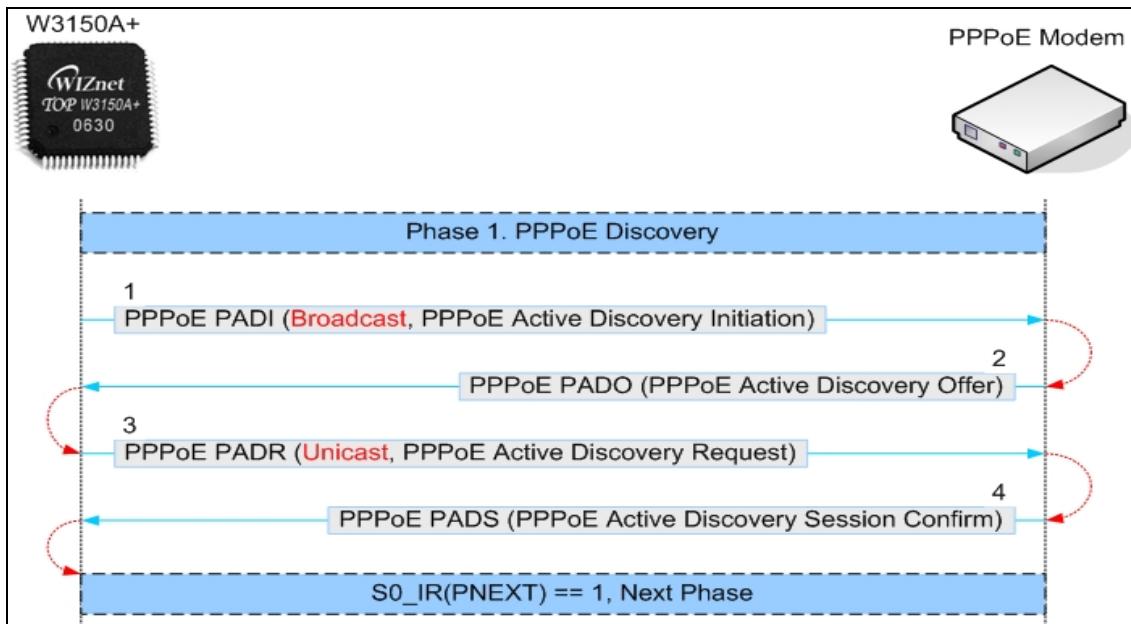
- PPPoE(ADSL) Setup

```
{
/* W3150A+ PPPoE(ADSL) initialization */
PHASE0 :
/* Set PPPoE bit in MR(Common Mode Register) */
MR = 0x08;
/* Set the value of PTIMER and PMAGIC */
PTIMER = 200; // set about 5 second
PMAGIC = 0x01;
/* Set PPPoE mode on socket 0 mode register */
SO_MR = 0x05;
/* Set OPEN command */
SO_CR = OPEN;
}
```

2. Phase 1

- PPPoE(ADSL) Discovery Process

Below figure shows the PPPoE(ADSL) Discovery process.



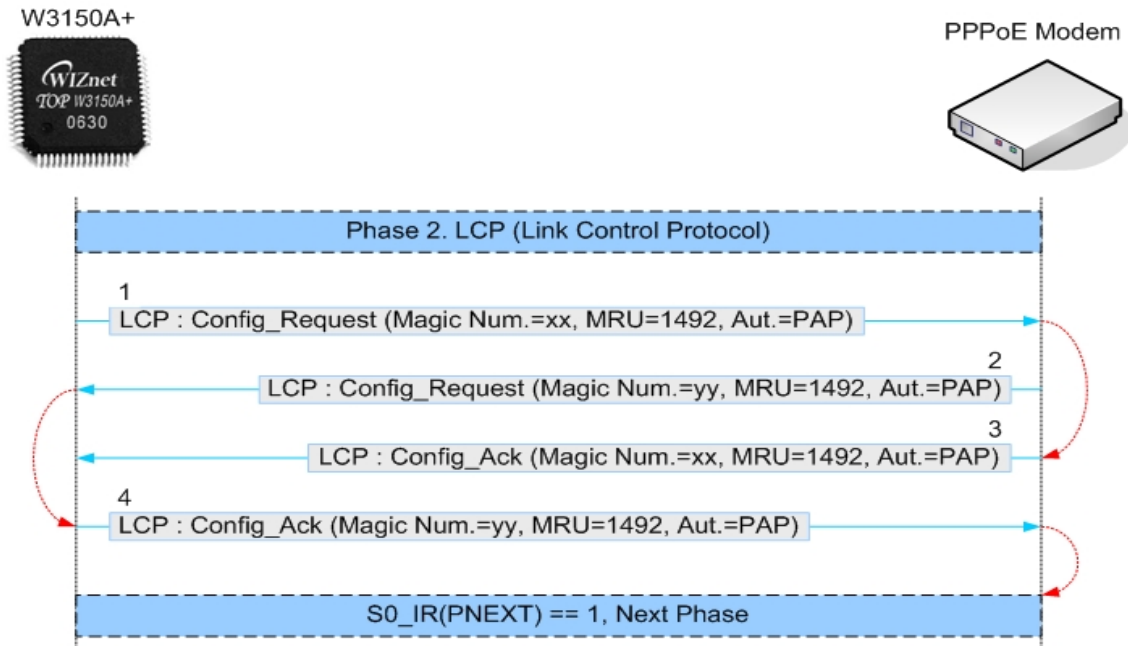
Through the step of PPPoE discovery, the IP address of PPPoE server(ADSL Server) and PPPoE session ID can be acquired.

```
{
/* Set Socket 0 Command Register as PCON for PPPoE(ADSL) connection.
   PPPoE(ADSL) discovery process starts */
S0_CR = PCON;
while
{
    wait some time
    /* check whether PNEXT bit of socket 0 Interrupt Register is set
       If PNEXT bit is set, PPPoE(ADSL) discovery process ends */
    if (S0_IR(PNEXT) == '1')
    {
        goto Next Phase (Phase2);
    }
    if (overtime) goto PHASE0;
}
}
```

3. Phase 2

- PPPoE(ADSL) LCP(Link Control Process) Process

Below figure shows LCP process.



By using LCP(Link Control Protocol), the authentication protocol type and MRU is negotiated. W3150A+ supports options of [Maximum Receive Unit\(0x01\)](#), [Authentication Protocol \(0x03\)](#), and [Magic-number\(0x05\)](#).

Below chart shows type values supported by W3150A+.

W3150A+ Support Type

Type	Name	Data
0x01	Maximum_Receive_Unit(MRU)	2 Bytes
0x03	Authentication_Protocol	PAP(0xC023), CHAP(0xC223)
0x05	Magic_Number	4 Bytes

```

{
    /* prepare option field of LCP
       Type, Length and option values comprise LCP configuration option field.
       Type(0x05, Magic number), Length(0x06, 6bytes), Magic number(4bytes) */
    option_array = {0x05, 0x06, PMAGIC, PMAGIC, PMAGIC, PMAGIC};
    copy option_array to socket 0 TX memory;
    /* for copying, refer to TCP sending process in 5.Functional description of datasheet.*/
    /* send LCP Config_REQ message */
    S0_CR = PCR;
    while
    {
    
```

wait some time

/ check PRECV bit of S0_IR is set */*

if (S0_IR(PRECV) == '1')

{

Get the *option_array* from RX memory of socket 0;

/ for getting the data, refer to TCP receiving process in 5.Functional description of datasheet */*

Parsing *option_array* and save reject option to *reject_option_array*

Support LCP option (Magic_Number Option)

Protocol ID (0xC021)	Code (0xFF)	ID (0xFF)	Length (0xFFFF)	Type (0x05)	Length (0x06)	Option data (4 bytes)
-------------------------	----------------	--------------	--------------------	----------------	------------------	--------------------------

Not Support LCP option

Protocol ID (0xC021)	Code (0xFF)	ID (0xFF)	Length (0xFFFF)	Type (0xFF)	Length (0xFF)	Option data
-------------------------	----------------	--------------	--------------------	----------------	------------------	-------------

Configure Reject LCP Option

Protocol ID (0xC021)	Code (0x04)	ID (0xFF)	Length (0xFFFF)	Type (0xFF)	Length (0xFF)	Option data
-------------------------	----------------	--------------	--------------------	----------------	------------------	-------------

Figure 1. Configure Reject LCP Option

{

Skip 6 bytes in *option_array*; // ppp header 6 bytes

/ each option field consist of [kind(1) | len(1) | value(n)] */*

Parsing all option fields as below

{

while (exist option field)

{

/ check support option type */*

if (option(kind) != {0x01, 0x03, 0x05})

save the option fields to *reject_option_array*;

}

}

}

Copy *reject_option_array* to socket 0 TX memory;

/ sends LCP Config_REJ message */*

S0_CR = PCJ;

```

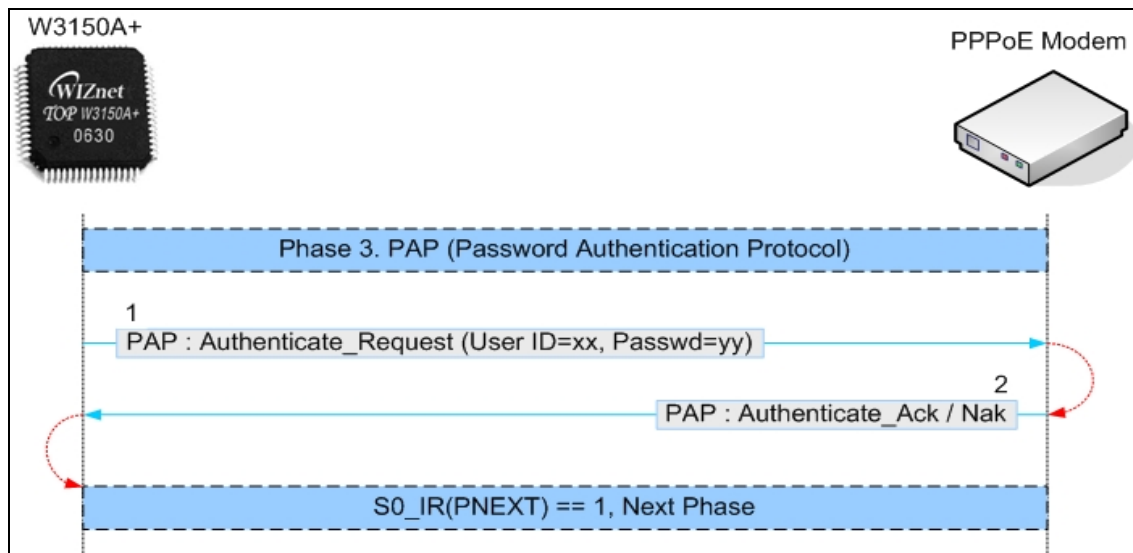
    }
    /* check PNEXT bit of S0_IR is set */
    if (S0_IR(PNEXT) == '1') goto PHASE3;
    if (overtime) goto PHASE0;
}
}

```

4. Phase 3

4.1 PAP(Password Authentication Protocol) Process

Below figure shows the PAP process.



Authentication process is performed with ID and password by using authentication protocol acquired by Phase2. In this case, PAP(Password Authentication Protocol) is used as authentication protocol.

```

{
    /* prepare option field of PAP Auth_REQ */
    /* [ IDlen(1) | ID(IDlen) | PWDlen(1) | PWD(PWDlen) ] */
    Save { IDlen(1),ID(IDlen),PWDlen(1),PWD(PWDlen) } to option_array
    copy option_array to TX memory of socket 0;
    /* send PAP Auth_REQ */
    S0_CR = PCR;
    while
    {
        wait some time
    }
}

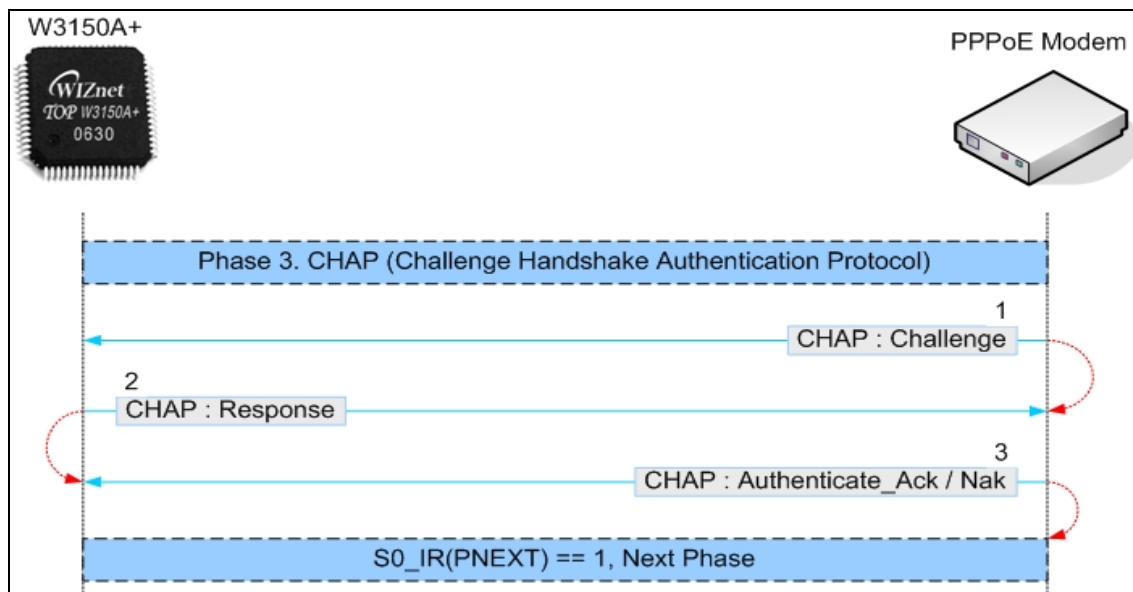
```

```

/* check PFAIL bit of S0_IR is set */
if (S0_IR(PFAIL) == '1')
{
    Re-check ID, Password
    Go to PHASE0;
}
/* check PNEXT bit of S0_IR is set */
if (S0_IR(PNEXT) == '1') goto IPCP;
if (overtime) goto PHASE0;
}
}
    
```

4.2 CHAP(Challenge Handshake Authentication Protocol) Process

Below figure shows PAP process.



Authentication process is performed with ID and password by using authentication protocol acquired by Phase2. In this case, CHAP(Challenge Handshake Authentication Protocol) is used as authentication protocol.

```

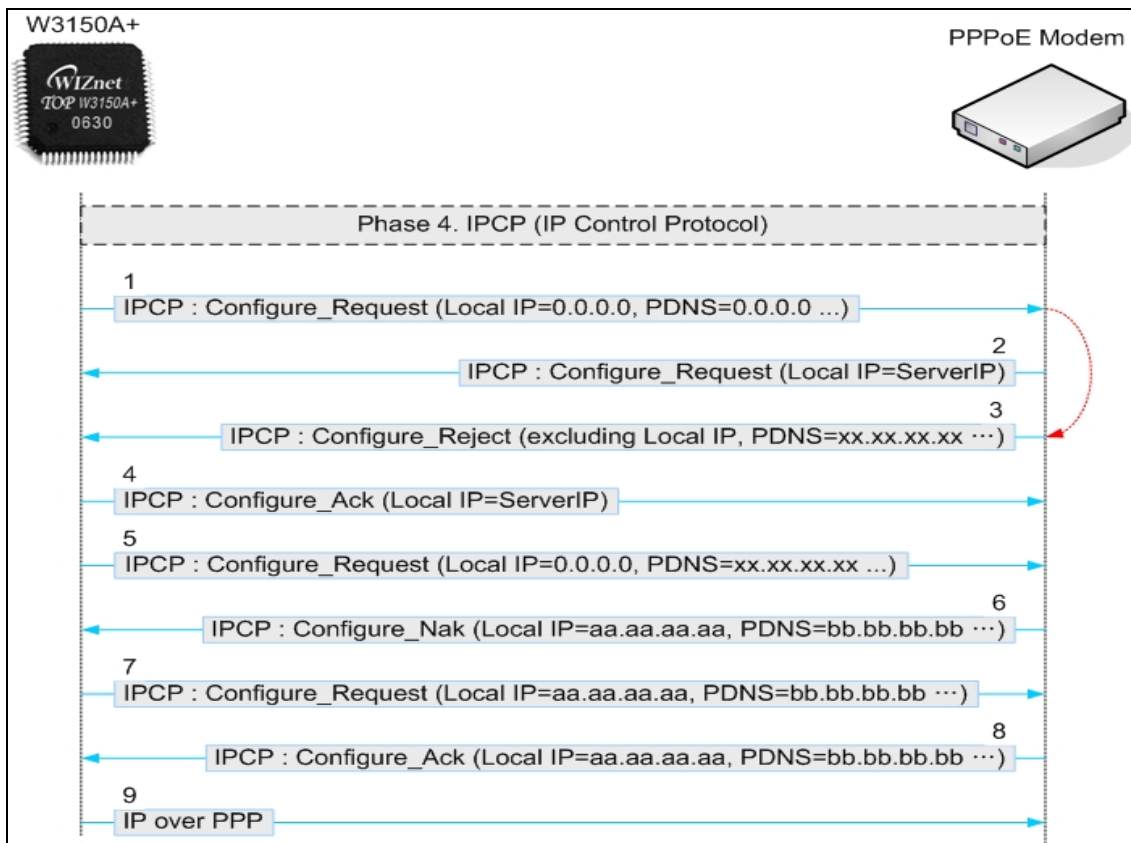
{
    /* for copying, refer to TCP receiving process in 5.Functional description of datasheet.*/
    /* receive PPP Challenge packet from PPP server */
    S0_CR = CRECV;
}
    
```



```
/* prepare CHAP Response packet to PPP server */
/* [ CHAP_ID(1) | Length(2) | HV(n) with MD5(Message Digest 5) ] */
/* for copying, refer to TCP sendign process in 5.Functional description of datasheet.*/
/* send PAP Auth_REQ */
S0_CR = PCR;
while
{
    wait some time
    /* check PFAIL bit of S0_IR is set */
    if (S0_IR(PFAIL) == '1')
    {
        Re-check ID, Password
        goto PHASE0;
    }
    /* check PNEXT bit of S0_IR is set */
    if (S0_IR(PNEXT) == '1') goto IPCP;
    if (overtime) goto PHASE0;
}
}
```

Phase 4> IPCP

Below figure shows the IPCP process.



In this phase, IP address is acquired by using IPCP. (If necessary, DNS and Gateway IP can be acquired, but only IP address is enough in ADSL)

```
{
    /* prepare option field of IPCP */
    option_array = {0x03, 0x06, 0x00, 0x00, 0x00, 0x00};
    copy option_array to socket 0 TX memory;
    /* send IPCP Config_REQ message */
    SO_CR = PCR;

    while
    {
        wait some time
        /* check PRECV bit of SO_IR is set */
        /* It is because IP address assigned to NAK message is sent from a server. */
        if (SO_IR(PRECV) == '1')
        {
            Get the received data of socket 0 RX memory and save to ip_option_array;
        }
    }
}
```

```

/* Parsing ip_option_array as below */
Skip 6 bytes in ip_option_array; // ppp header 6 bytes
{
    /* Parsing all option fields as below */
    /* each option field consist of [ kind(1) | len(1) | value(n) ] */
    while (exist option field)
    {
        /* check ip option field */
        if (option(kind) == 0x03)
        {
            save the option fields to option_array;
            goto IPCP_END;
        }
    }
}
}
if (overtime) goto PHASE0;
}

IPCP_END:
Copy option_array to socket 0 TX memory;
/* resend IPCP Config_REQ message */
SO_CR = PCR;
while
{
    wait some time
    /* check PNEXT bit of S0_IR is set */
    if (S0_IR(PNEXT) == '1') goto PHASE5;
    if (overtime) goto PHASE0;
}
}

```

Phase 5 > End

All the process for ADSL connection is finished. Close the 0th socket and use it.

```

{
    /* set CLOSE command */
}

```

```
SO_CR = CLOSE;  
}
```

Phase 6 > ADSL Disconnection

```
{  
    /* Set PPPoE bit in MR(Mode Register). */  
    MR = 0x08;  
    /* Set PPPoE mode on socket 0 mode register */  
    SO_MR = 0x05;  
    /* set the ADSL server information */  
    SO_DHAR = PPPoE_Server ;  
    SO_DPORT = PPPoE_Session_ID;  
    /* Set OPEN command */  
    SO_CR = OPEN;  
    /* Set PDISCON command for starting to disconnect to ADSL server */  
    SO_CR = PDISCON;  
    /* set CLOSE command */  
    SO_CR = CLOSE;  
}
```