W5100S Application note

**How to use the interrupt**

(AN02001)



<http://www.wiznet.co.kr>

**Introduction**

W5100S는 1개의 Interrupt Pin을 제공하며 User는 Interrupt pin을 통해 Event가 발생되는 시점을 정확하게 파악하여 효과적인 프로그래밍을 할 수 있다. W5100S의 Ethernet packet 통신 처리(IP 충돌감지, WOL magic packet 수신, 각 통신 SOCKET별 데이터 송수신 등)를 위한 Event가 발생할 경우 INTn pin이 Low로 assert된다.



<W5100S INTn Pin >

INTn pin의 기본 설정값은Enable이며 MR2(Mode register 2)의 IEN bit를 통하여 Enable/Disable 설정이 가능하다.

**Contents**

1. Interrupt configuration ∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙
   1. 관련 Register ∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙
   2. INTn의 동작 ∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙
   3. Common interrupt ∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙
   4. Socket interrupt
   5. Socket-less command interrupt
2. Host측에서의 Interrupt 처리 ∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙
   1. Host의 Interrupt 설정 ∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙
   2. Packet receive ineterrupt configuration example ∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙
   3. Socket-less interrupt configuration example ∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙
3. 기타 ∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙
   1. Interrupt 사용시 주의점 ∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙
   2. RTOS에서 Interrupt 사용시 주의점 ∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙∙
4. Interrupt configuration
   1. Registers related to interrupt.

|  |  |  |  |
| --- | --- | --- | --- |
| Symbol | Address | Description | Remarks |
| INTLEVEL | 0x0013-14 | Interrupt Low Level Timer |  |
| IR | 0x0015 | Interrrupt register |  |
| IMR | 0x0016 | Interrupt mask register |  |
| IR2 | 0x0020 | Interrupt register 2 |  |
| IMR2 | 0x0021 | Interrupt maks register 2 |  |
| MR2 | 0x0030 | Mode register 2 |  |
| SLIR | 0x005E | Socket-less command interrupt register |  |
| SLIMR | 0x005F | Socket-less command interrupt mask register |  |
| Sn\_IR | 0x0403+0x100\*n | Socket n interrupt register |  |
| Sn\_IMR | 0x042C+0x100\*n | Socket n interrupt mask register |  |
|  |  |  |  |

<Table 1. Interrupt 관련된 Registers>

Table 1은 인터럽트와 관련있는 register들을 나타낸 것이다. 각 register의 자세한 설명은 W5100S Data를 참조하라. Interrupt register들은 크게 세 가지로 나누어 진다. 첫 번째, Interrupt register는 발생된 Event를 확인할 수 있는 register이다. 두 번째, Interrupt mask register는 Interrupt register에 1:1 대응되는 masking register로, 해당 bit가 1로 assert되어 있어야 해당 event가 발생했을 때 INTn이 Low로 assert된다. 마지막으로 INTn pin자체의 동작을 결정하는 register가 있다. Mode register 2(MR2)는 INTn pin의 Enable/Disable을 결정하는 IEN bit를 지니고 있다. Mode register 2의 IEN이 Enable(‘1’) 상태여야만 Event가 발생하면 INTn 핀이 Low로 assert된다.

* 1. INTn의 동작

INTn 핀은 Event 발생 여부를 Host에게 전압 레벨의 변화로 알려준다. INTn 핀의 전압 레벨은 기본적으로 High 상태를 유지하고 Event 발생 시 Low 상태로 변한다. 이 때, INTLEVEL의 값이 0이 아니면 INTLEVEL register에 설정된 시간 이후 Low 상태로 변한다. Host에서의 Event 처리가 끝나면 해당 Interrupt register bit에 ‘1’을 설정하여 interrupt가 clear할 수 있으며 INTn 핀의 상태는 High로 변한다.

그림그림그림그림그림 상준아 그림 그려줘~ 그림 그려줘~

<그림 넣어요~>

* 1. Common interrupt

IR은 IP Conflict, Port unreachable, PADT/LCPT receive 그리고 각 Socket의 event를 감지하는 interrupt를 제공한다. IR과 1:1 대응되는 IMR의 bit를 ‘1’로 설정하여 해당 interrupt를 활성화한다. IMR가 0이면 event 발생으로 인해 IR의 각 비트가 1로 변해도 INTn은 Low로 assert되지 않는다.

IR2는 MAGIC packet을 수신 event를 감지하는 interrupt를 제공한다. 1:1 대응되는 IMR2의 bit를 ‘1’로 설정하여 interrupt를 활성화한다.

만약 IP CONFLICT interrupt를 확인하고자 한다면 다음과 같은 과정을 따라야 한다.

Register configuration

start

MR2 |= 1<<6 // MR2의 6번(IEN) bit 활성화 – INTn pin 활성화

IMR |= 1<<7 // CONFLICT event 활성화

end

IP Conflict error occurred

(In host’s interrupt handler)

start

if(IR && 1<<7) //CONFLICT interrupt 확인

Do something!!

IR |= 1<<7; //Clear CONFLICT interrupt

end

* 1. Socket interrupt

W5100S는 Socket의 상태 변화 event를 감지하는 interrupt를 제공한다. Sn\_IR과 1:1 대응되는 Sn\_IMR의 bit를 ‘1’로 설정하여 해당 interrupt를 활성화시킬 수 있다. Sn\_IMR에 설정된 Socket event가 발생하면 Sn\_IR의 해당 bit가 ‘1’로 assert되고 IR의 Sn\_INT bit도 ‘1’로 assert된다. 이 때, MR2의 IEN bit나 IMR의 해당 interrupt mask bit가 비활성화 상태이면 INTn이 Low로 assert되지 않는다. Socket event로 인해 INTn이 Low로 assert되면 Host는 IR을 읽어 몇번 Socket에서 event가 발생했는지 확인한 후 Sn\_IR을 통해 어떤 Interrupt가 발생했는지 확인해야만 한다.

만약 0번 소켓의 RECV interrupt를 확인하고자 한다면 다음과 같이 과정을 따라야 한다.

Register configuration

start

MR2 |= 1<<6 // MR2의 6번(IEN) bit 활성화 – INTn pin 활성화

IMR |= 1<<0 // IMR의 0번 S0\_INT bit 활성화 – Socket 0 interrupt 활성화

S0\_IMR |= 1<<2 // RECV mask bit 활성화

end

RECV event occured

(In host’s interrupt handler)

start

if(IR && 1<<0) // S0\_INT 확인

if(S0\_IR && 1<<2) // RECV interrupt 확인

Do something!

IR |= 1<<0; // Clear Socket 0 interrupt

S0\_IR |= 1<<2;//Clear Socket 0 RECV interrupt

end

Socket의 상태 변화에 따른 Event는 SENDOK, TIMEOUT, RECV,DISCN,CON이 있으며 자세한 사항은 W5100S 데이터 시트를 참조하라.

* 1. Socket-less command interrupt

W5100S는 Socket open 없이 PING이나 ARP를 송수신할 수 있는 기능이 추가되었고 그에 따른 interrupt 기능이 추가되었다. SLIR은 PING이나 ARP에 대한 응답 event를 감지하는 interrupt를 제공한다. SLIR과 1:1 대응되는 SLIMR의 bit를 ‘1’로 설정하여 해당 interrupt를 활성화시킬 수 있다. SLIMR에 설정된 Socket-less command의 응답 event가 발생하면 SLIR의 해당 bit가 ‘1’로 assert되고 INTn이 Low로 assert된다.

EX) Socket-less command를 이용해 PING packet을 전송하고 PING 응답에 대한 interrupt를 확인하고자 한다면 다음과 같은 과정을 따라야 한다.

Register configuration

start

MR2 |= 1<<6 //MR2의 6번(IEN) bit 활성화 – INTn pin 활성화

SLIMR |= 1<<0 //SLIMR의 0번(SLCMD\_PING) bit 활성화 – PING 응답 interrupt 활성화

end

Interrupt occurred

start

if(SLIR && 1<<0) //PING 응답 interrupt 확인

Do something

SLIR |= 1<<0; // Clear Ping 응답 interrupt

end

Socket-less command 응답 event는 TIMEOUT, PING, ARP가 있으며 자세한 사항은 W5100S 데이터 시트를 참조하라.

1. Host에서의 Interrupt 처리
   1. Host측에서의 interrupt 설정

Host는 W5100S과 연결되어 제어하는 MCU(Micro controller unit)를 의미한다. MCU는 일반적으로 pin의 상태 변화(전압 Level 변화, Edge 등)를 감지하여 그에 맞는 동작을 할 수 있는 Hardware component를 내장하고 있다. External interrupt unit 혹은 Event contoller driver등 다양한 이름으로 사용되고 있으나 동작은 거의 비슷하므로 본 문서에서는 External interrupt라고 명칭하도록 하겠다. External interrupt를 구성하기 위해서는 MCU의 관련 Hardware 초기화와 Interrupt handler가 필요하다. Interrupt handler는 W5100S에서 Event가 발생했을 때에 실행되는 Software를 말한다.

* 1. Packet receive ineterrupt example

해당 Example은 STM32F1XX 시리즈 MCU를 기준이며 WIZnet Ethernet IC 공식 library인 ioLibrary를 사용하였다.

W5100S의 INTn핀이 STM32F1XX의 GPIOB의 1번 핀에 연결되어 있고 Socket 0의 RECV interrupt만 활성화 되어 있을 경우.

Initialize external interrupt hardware unit

void InitializeExternalInterrupt(void)

{

GPIO\_InitTypeDef GPIO\_InitStructure;

EXTI\_InitTypeDef EXTI\_InitStructure;

/\*GPIO initialize\*/

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_IPU;

GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_1;

GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz;

GPIO\_Init(GPIOB, & GPIO\_InitStructure);

/\*External interrupt initialize\*/

GPIO\_EXTILineConfig(GPIOB\_PortSourceGPIOB, GPIO\_PinSource1);

EXTI\_InitStructure.EXTI\_Line = EXTI\_Line8;

EXTI\_InitStructure.EXTI\_Mode = EXTI\_Mode\_Trigger\_Falling;

EXTI\_InitStructure.EXTI\_LineCmd = ENABLE;

EXTI\_Init(&EXTI\_InitStructure);

}

W5100S interrupt configuration

MR2 |= 1<<6 // MR2의 6번(IEN) bit 활성화 – INTn pin 활성화

IMR |= 1<<0 // IMR의 0번 S0\_INT bit 활성화 – Socket 0 interrupt 활성화

Interrupt handler

void EXTI1\_IRQHandler(void)

{

//setMR2(getMR2() & ~(1<<6));//Global interrupt disable 이거 고려고려

if(EXTI\_GetITStatus(EXTI\_Line1) == SET) //Check the interrupt.

{

//(1)Set the global interrupt flag

interruptflag = 1;

setSn\_IR(0xff); //Clear socket interrupt 요것도 고려

}

//Clear External interrupt flag

EXTI\_ClearFlag(EXTI\_Line1);

}

(1) It is not recommended to execute too many commands in the interrupt handler. This can cause serious problem to your system. Just sets the flag in the interrupt handler and executes the functions outside the interrupt handler.

* 1. Socket-less command interrupt example

해당 Example은 STM32F1XX 시리즈 MCU 기준이며 WIZnet Ethernet IC의 공식 library인 ioLibrary를 사용하여 작성하였다.

W5100S의 INTn 핀이 STM32F1XX의 GPIOB의 1번 핀에 연결괴어 있고 Socket-less command의 PING의 응답만 활성화되어 있는 경우.

External interrupt를 활성화하는 부분은 1.2 Packet receive interrupt example과 동일하다.

void EXTI1\_IRQHandler(void)

{

//setMR2(getMR2() & ~(1<<6));//Global interrupt disable

if(EXTI\_GetITStatus(EXTI\_Line1) == SET) //Check the interrupt.

{

//(1)Set the global interrupt flag

interruptflag = 1;

setSLIR(0xff); //Clear socket interrupt 요거 고려

}

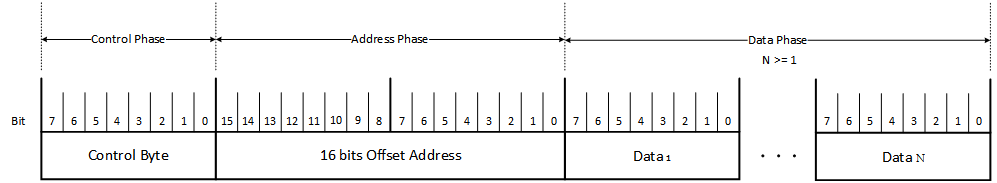
//Clear External interrupt flag

EXTI\_ClearFlag(EXTI\_Line1);

}

1. 기타
   1. Interrupt 사용시 주의점

W5100S의 접근(Read or Write)은 Frame으로 구성되어 있다.(Datasheet External interface 참조) 한 Frame은 각각 Address phase, Control phase, Data phase로 이루어져 있는데 통신 도중 이 Frame의 순서가 지켜져야만 한다. 예를 들어 SPI Frame은 아래 그림과 같은 형식을 유지해야만 한다.



<SPI Frame>

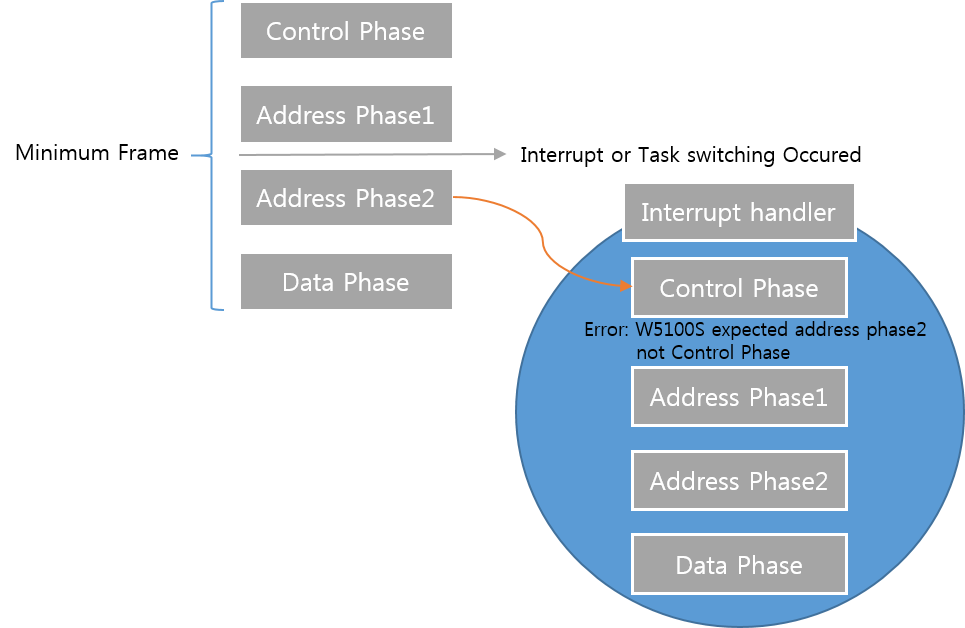
하지만 Interrupt를 사용하는 상황에서 해당 Frame의 순서에 문제가 생길 수 있다. 예를 들어 RECV interrupt를 활성화시킨 상태에서 W5100S의 Data를 읽는 상황을 생각해보자. Data를 읽기 위해 Control byte를 보내고 그 다음 상위 Address byte를 보낸 직 후 Interrupt가 발행하게 되면 MCU는 하던 일을 멈추고 Interrupt handler로 점프해 코드를 수행하게 된다. 이 때 당신이 W5100S의 register를 읽고 쓰거나 W5100S와 같은 addres, data bus, 혹은 SPI bus를 공유하는 IC에 접근한다면 W5100S의 Frame은 더 이상 유효하지 않게 된다. 이 문제를 해결하기 위해 Frame이 손상되지 않는 최소 단위를 지정해 한 Frame이 진행되고 있을 때에는 같은 자원을 사용하는 Interrupt handler가 수행되지 않도록 해야 한다.

반드시 순차적으로 수행되어져야 할 Frame을 일반적으로 critical section 이라고 부르며 ioLibrary에 포함된 reg\_wizchip\_cris\_cbfunc() 함수를 통해 critical section이 수행될 때의 동작을 설정할 수 있다.

처음의 예와 같이 Recv interrupt를 활성화 한 상태에서 Send를 수행한다고 가정하자. 이 때 Send 동작의 최소 Frame이 시작되기 전에 Interrupt는 Disable되어 있어야만 한다. 그리고 Send의 최소 Frame이 끝나면 Interrupt를 Enable시켜 Frame이 종료된 후 Interrupt\_handler가 동작할 수 있도록 프로그래밍 해야한다.

* 1. RTOS에서 Interrupt 사용 시 주의점.

RTOS도 Task의 전환 때문에 3.1. Interrupt 사용시 주의점에서 설명한 문제점과 동일한 현상이 발생할 수 있다. 마찬가지로 최소 Frame이 동작하는 중간에 Task switching이 일어나 다른 Task를 수행할 때 같은 자원(Bus)을 사용하게 되면 Frame이 손상되어 정상적인 데이터 통신을 할 수 없게 된다. 이를 막기 위해 reg\_wizchip\_cris\_cbfunc()를 이요해 scheduler를 lock/unlock하는 함수를 등록하거나 RTOS에서 제공하는 Mutex(Mutual exclusive) 등의 기능을 활용하는 것이 바람직하다.



<Behavior when an interrupt or task switching occurs>