

# 「SpresenseでUDP送信をすると15秒ほどで送信側の動作が停止してしまう」について

# 更新履歴

版数	更新内容	更新日
0.2	Refsを追加した。	2023/01/12
0.1	初版を0.1版とする。	2022/09/09

# 目次

- ◆ 問題点
- ◆ 原因調査
- ◆ エージングテストした結果
  - ◆ 高温
  - ◆ サーマルシャット発生したこともある
  - ◆ タイムアウトが発生して、復帰できない
- ◆ 対策方法：HWリセット
- ◆ 復帰できない場合、HWリセットして、サンプルの紹介

※W5500-Etherボード（SPRESENSE用）概要を参照する場合、下記SSUPのGitHubをご参照ください。

<https://github.com/SonySemiconductorSolutions/ssup-spresense/tree/main/Arduino/W5500>

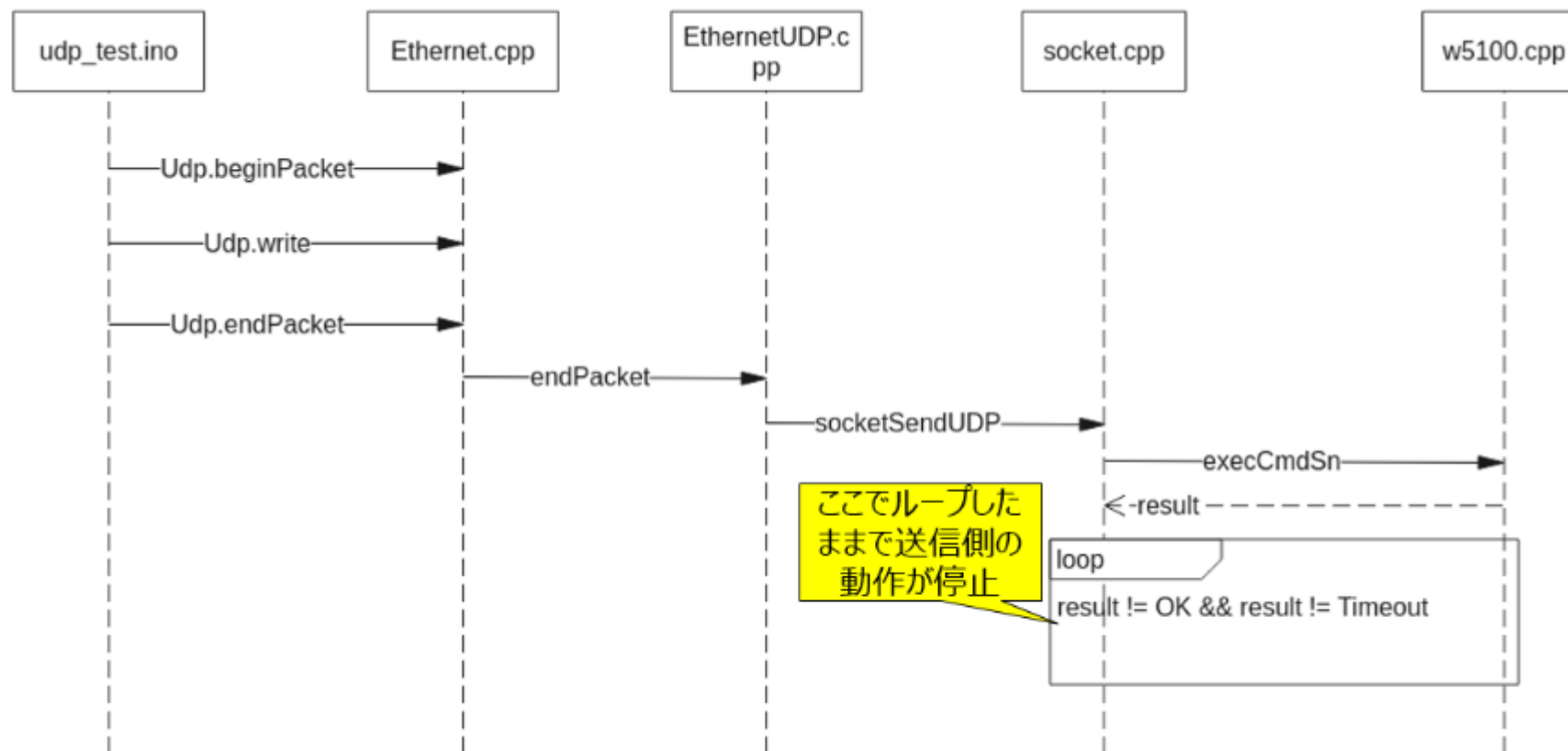
# SpresenseでUDP送信をすると15秒ほどで送信側の動作が停止してしまう

## 問題点

現在、Spresenseの有線ネットワーク拡張ボードであるクレイン電子のW5500-Etherを用いてSpresenseで撮影した画像をUDPにて送信するシステムを開発しているのですが、プログラム起動直後は送信できているのですが、15秒ほどで送信側の動作が停止してしまい、長時間動作させるのが困難な状況です。

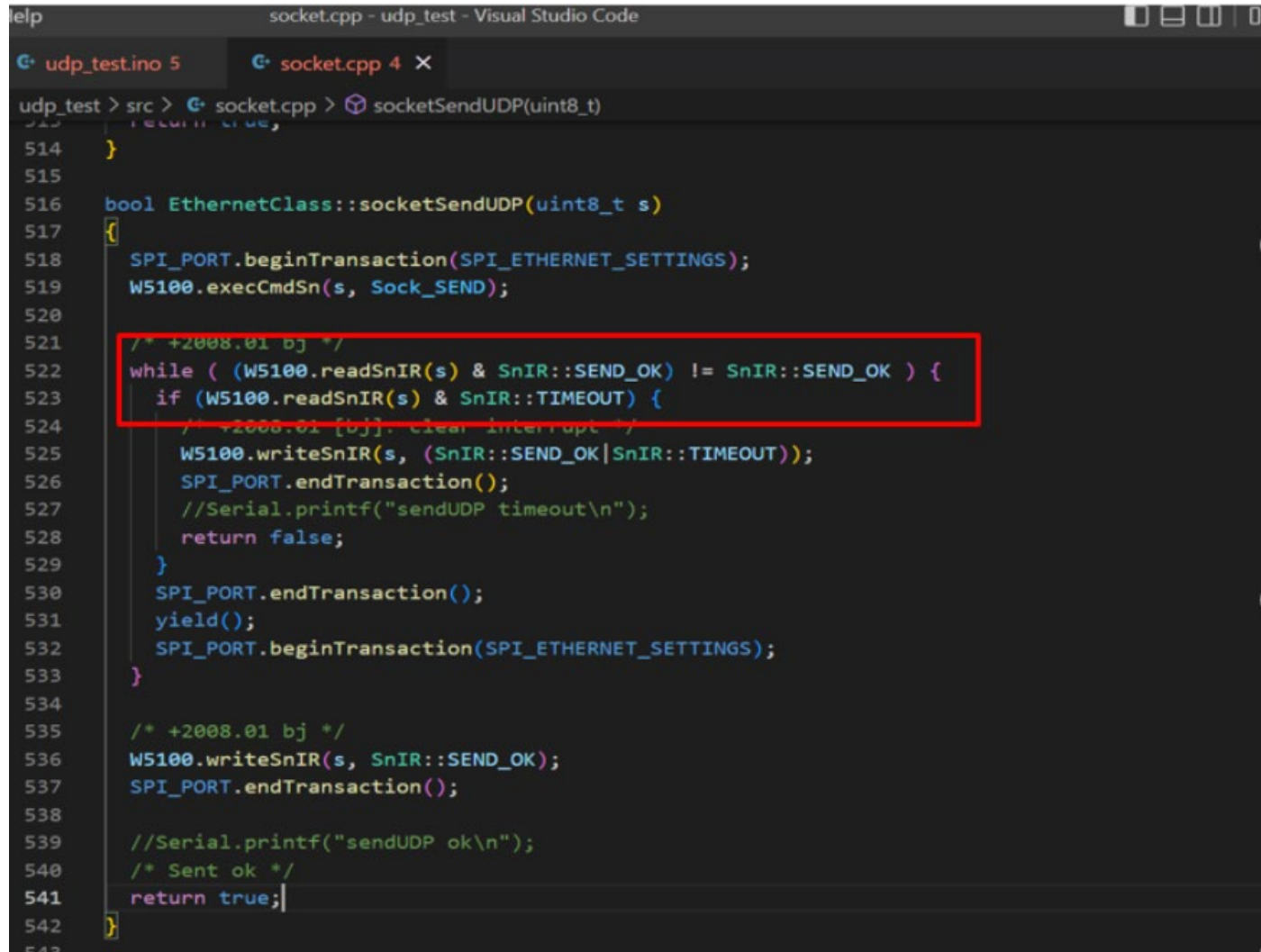
またSpresense側の機能をUDPの送信のみに限定した場合にも同様の問題が生じるため、Cameraなどの問題ではなくEthernetに関連する問題だと考えています。

## 原因



# SpresenseでUDP送信をすると15秒ほどで送信側の動作が停止してしまう

原因



```
socket.cpp - udp_test - Visual Studio Code
udp_test > src > socket.cpp > socketSendUDP(uint8_t)
514 }
515
516 bool EthernetClass::socketSendUDP(uint8_t s)
517 {
518     SPI_PORT.beginTransaction(SPI_ETHERNET_SETTINGS);
519     W5100.execCmdSn(s, Sock_SEND);
520
521     /* +2008.01 bj */
522     while ( (W5100.readSnIR(s) & SnIR::SEND_OK) != SnIR::SEND_OK ) {
523         if (W5100.readSnIR(s) & SnIR::TIMEOUT) {
524             /* +2008.01 [bj]. clear interrupt */
525             W5100.writeSnIR(s, (SnIR::SEND_OK|SnIR::TIMEOUT));
526             SPI_PORT.endTransaction();
527             //Serial.printf("sendUDP timeout\n");
528             return false;
529         }
530         SPI_PORT.endTransaction();
531         yield();
532         SPI_PORT.beginTransaction(SPI_ETHERNET_SETTINGS);
533     }
534
535     /* +2008.01 bj */
536     W5100.writeSnIR(s, SnIR::SEND_OK);
537     SPI_PORT.endTransaction();
538
539     //Serial.printf("sendUDP ok\n");
540     /* Sent ok */
541     return true;
542 }
```

# SpresenseでUDP送信をすると15秒ほどで送信側の動作が停止してしまう

## クレイン電子さんと問い合わせした結果

本件の問題について改修版のソースコードを送付いたします。

srcファイルを添付のものに置き換えてもう一度お試しください。

添付ファイル同梱の.inoファイルはこちらで実験用に編集したもので  
無視して頂いて構いません。下記に、不具合についてまとめます。

- **【内容】**  
UDP連続送信時、10秒程度のランダムなタイミングで送信が停止されフリーズしてしまう。
- **【不具合の原因】**  
Ethernetライブラリのtimeout処理に不備があった。
- **【詳細】**  
socket.cpp内の EthernetClass::socketSendUDP() において、UDP転送に問題が生じ、timeoutしているにも関わらずtimeoutフラグが (W5500チップ仕様の問題の可能性あり？ 追及せず) 立たず無限ループに陥っている事が分かった。
- **【対策】**  
対象のループに、別途Arduino関数のmillis()によるカウンタを設け、W5500\_TIMEOUT 秒(デフォルト100msec、#defineで定義)後にtimeoutを判定する処理を追加。  
参考としたissue、およびPullRequest  
[https://urldefense.com/v3/\\_\\_https://github.com/arduino-libraries/Ethernet/pull/175/files\\_\\_!!JmoZiZGBv3RvKRSx!76rWaUyyIB6F-pJmjfPkDxURH4o2xDigjr8ZQOwIAcbccdRM1LjSYWsieN6JfB3liWsxjExR8HHaKe5vS3rBrtn04L8\\$ \[github\[.\]com\]](https://urldefense.com/v3/__https://github.com/arduino-libraries/Ethernet/pull/175/files__!!JmoZiZGBv3RvKRSx!76rWaUyyIB6F-pJmjfPkDxURH4o2xDigjr8ZQOwIAcbccdRM1LjSYWsieN6JfB3liWsxjExR8HHaKe5vS3rBrtn04L8$ [github[.]com])  
本家にもまだマージされていないため、潜在的なライブラリ不具合と推察します。
- 対策後、対象のエラーでスタックすることなく送信及び処理が続行された。

# SpresenseでUDP送信をすると15秒ほどで送信側の動作が停止してしまう

## SSUPでエーシングテストした結果

### 【現象1】

W5500ハードウェアチップが高温であること。(手で触ると、焼けどする程度)  
長時間実行すると、EtherNet有線のランプが消えて、pingはできないことがある。

### 【原因調査】

高温でEthernet有線のランプが消えたことはサーマルシャットが発生した  
かもしれないため、W5500データシートで確認しました。

それで Power Down Modeがサポートされていることを確認できました。  
ただ、具体的にどういつきにPower Down Modelに遷移するか、記載されていません。

42 / 66 | 88% + | [ ] [ ]

PHYCFGR (W5500 PHY Configuration Register) [R/W] [0x002E] [0b10111XXX]

PHYCFGR configures PHY operation mode and resets PHY. In addition, PHYCFGR indicates the status of PHY such as duplex, Speed, Link.

Bit	Symbol	Description																																				
7	RST	<b>Reset [R/W]</b> When this bit is '0', internal PHY is reset. After PHY reset, it should be set as '1'.																																				
6	OPMD	<b>Configure PHY Operation Mode</b> 1: Configure with OPMD[2:0] in PHYCFGR 0: Configure with the H/W PINs(PMODE[2:0]) This bit configures PHY operation mode with OPMD[2:0] bits or PMODE[2:0] PINs. When W5500 is reset by POR or RSTn PIN, PHY operation mode is configured with PMODE[2:0] PINs by default. After POR or RSTn reset, user can re-configure PHY operation mode with OPMD[2:0]. If user wants to re-configure with PMDC[2:0], it should reset PHY by setting the RST bit to '0' after the user configures this bit as '1' and OPMD[2:0].																																				
5-3	OPMDC	<b>Operation Mode Configuration Bit[R/W]</b> These bits select the operation mode of PHY such as following table. <table><tr><th>5</th><th>4</th><th>3</th><th>Description</th></tr><tr><td>0</td><td>0</td><td>0</td><td>10BT Half-duplex, Auto-negotiation disabled</td></tr><tr><td>0</td><td>0</td><td>1</td><td>10BT Full-duplex, Auto-negotiation disabled</td></tr><tr><td>0</td><td>1</td><td>0</td><td>100BT Half-duplex, Auto-negotiation disabled</td></tr><tr><td>0</td><td>1</td><td>1</td><td>100BT Full-duplex, Auto-negotiation disabled</td></tr><tr><td>1</td><td>0</td><td>0</td><td>100BT Half-duplex, Auto-negotiation enabled</td></tr><tr><td>1</td><td>0</td><td>1</td><td>Not used</td></tr><tr><td>1</td><td>1</td><td>0</td><td>Power Down mode</td></tr><tr><td>1</td><td>1</td><td>1</td><td>All capable, Auto-negotiation enabled</td></tr></table>	5	4	3	Description	0	0	0	10BT Half-duplex, Auto-negotiation disabled	0	0	1	10BT Full-duplex, Auto-negotiation disabled	0	1	0	100BT Half-duplex, Auto-negotiation disabled	0	1	1	100BT Full-duplex, Auto-negotiation disabled	1	0	0	100BT Half-duplex, Auto-negotiation enabled	1	0	1	Not used	1	1	0	Power Down mode	1	1	1	All capable, Auto-negotiation enabled
5	4	3	Description																																			
0	0	0	10BT Half-duplex, Auto-negotiation disabled																																			
0	0	1	10BT Full-duplex, Auto-negotiation disabled																																			
0	1	0	100BT Half-duplex, Auto-negotiation disabled																																			
0	1	1	100BT Full-duplex, Auto-negotiation disabled																																			
1	0	0	100BT Half-duplex, Auto-negotiation enabled																																			
1	0	1	Not used																																			
1	1	0	Power Down mode																																			
1	1	1	All capable, Auto-negotiation enabled																																			

srcUtility\W5100.h  
\_\_GP\_REGISTER8(PHYCFGR\_W5500, 0x002E);  
// PHY Configuration register, default: 10111xxx

```
srcUtilityWw5100.h  
__GP_REGISTER8 (PHYCFGR_W5500, 0x002E);  
// PHY Configuration register, default: 10111xxx
```

[Refs]

[https://cdn.sparkfun.com/datasheets/Dev/Arduino/Shields/W5500\\_datasheet\\_v1.0.2\\_1.pdf](https://cdn.sparkfun.com/datasheets/Dev/Arduino/Shields/W5500_datasheet_v1.0.2_1.pdf)

# SpresenseでUDP送信をすると15秒ほどで送信側の動作が停止してしまう

## SSUPでエージングテストした結果

### 【現象2】

小型扇風機でW5500ハードウェアチップにクールダウンしながらエージングテストを実施し

タイムアウトの現象が発生して、復帰できない場合がある。

※もう一つのW5500アドオンボードで14時間以上実行させ、スタックやタイムアウトなどの現象は発生せず、問題なく実行できています。  
エアコン温度を低めに設定したことが奏功したか？（環境温度を21-22度に維持させ、発熱はあったものの、動作への支障は出なかった）

### 【提案】

タイムアウトが発生して復帰できない場合はHWリセットを実施して、プログラムを再実行する。



# SpresenseでUDP送信をすると15秒ほどで送信側の動作が停止してしまう

## HWリセット方法調査

- W5500の回路図から参照すると、W5500のRSTNはSpresenseのEMMC\_DATA3(PIN\_21)と接続してることです。  
W5500リセット方法は下記です。

```
//W5500_Eth RESET# = LOW  
digitalWrite(PIN_D21, LOW);  
//RESET should be held low at least 500 us for W5500 reset.  
delay(1);  
//W5500_Eth RESET# = HIGH  
digitalWrite(PIN_D21, HIGH);
```

## [詳細プログラム]

<https://github.com/SonySemiconductorSolutions/ssup-spresense/tree/main/Arduino/W5500/udpTimeoutReset>

## [Refs]

[https://crane-elec.co.jp/wp/wp-content/uploads/2021/05/w5500\\_r2\\_sch.pdf](https://crane-elec.co.jp/wp/wp-content/uploads/2021/05/w5500_r2_sch.pdf)

