

Renesas S5D9 用サンプル(e2studio WIRE_DHCP_TCP_RSA)の説明

(e2studio Version:2022-10 / SSP Version 2.4.0)

1. Sample の免責について

- **Sample** に関する **Tel/Fax** でのご質問に関してはお受けできません。ただし、メールでのご質問に関してはお答えするよう努力はしますが、都合によりお答えできない場合もありますので予めご了承ください。
- **Sample** ソフトの不具合が発見された場合の対応義務はありません。また、この関連ソフトの使用方法に関する質問の回答義務もありませんので承知の上ご利用下さい。
- **Sample** ソフトは、無保証で提供されているものであり、その適用可能性も含めて、いかなる保証も行いません。また、本ソフトウェアの利用により直接的または間接的に生じたいかなる損害に関しても、その責任を負わないものとします。

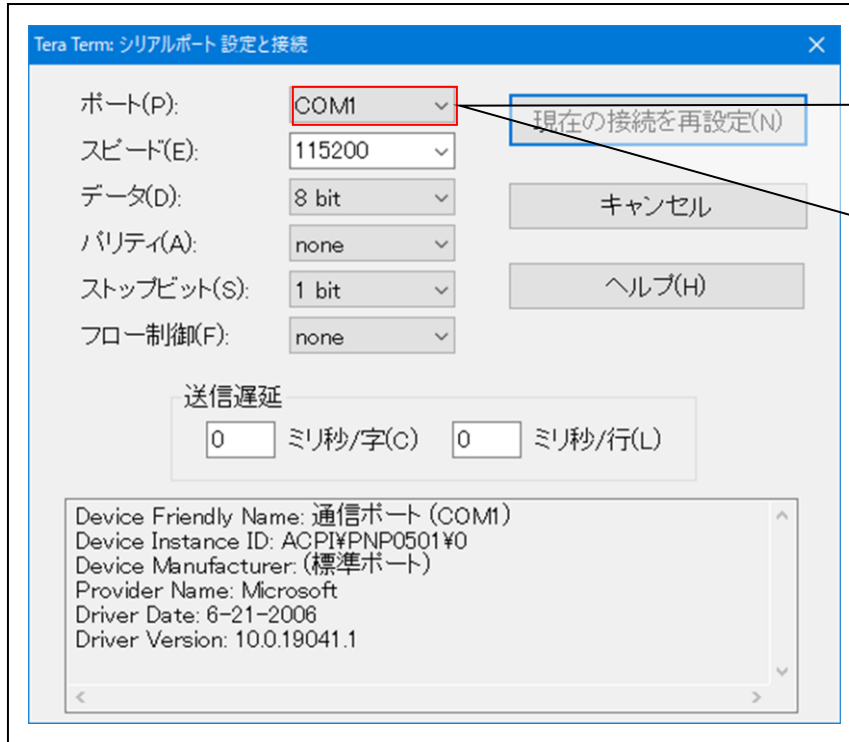
2. サンプルのプロジェクト名

ワークスペース名	概要	プロジェクト名
S5D9_e2std_WIRE_NetX_1	有線 LAN 接続した DHCP と TCP 通信のサンプル セキュリティエンジン(SCE7) Crypto ドライバーの RSA を使用したサンプル (暗号・復号)	WIRE_DHCP_TCP_RSA_ETH Azure RTOS モードで動作 NetX DHCP Client (g_dhcp_client0) TCP 通信(Client) (nx_tcp_socket_.....) 暗号・復号(RSA) (g_sce_rsa_r.p_api->encrypt) (g_sce_rsa_r.p_api->decrypt)

統合開発環境
Renesas e2studio(Version 2022-10)
SSP(Version2.4.0)

3. Tera Term Pro のインストール

- ① 「teraterm-4.106.exe」 を検索してダウンロードする。
- ② PC にインストールし実行する
- ③ シリアルポートの設定



COM 番号は、
PC 側でシリアル通信可能
な番号を指定する。

115200BPS

8bit

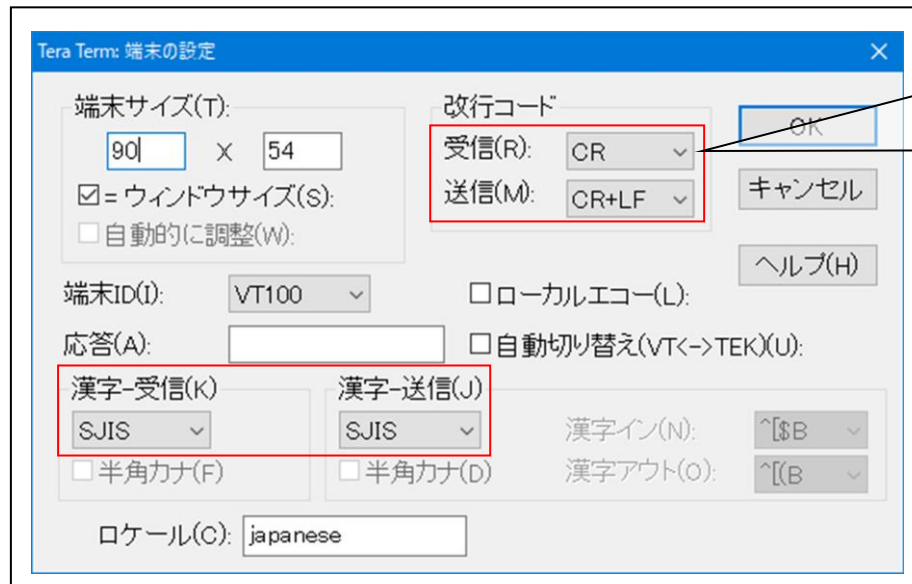
none

1bit

none

の仕様にする。

④ 端末の設定

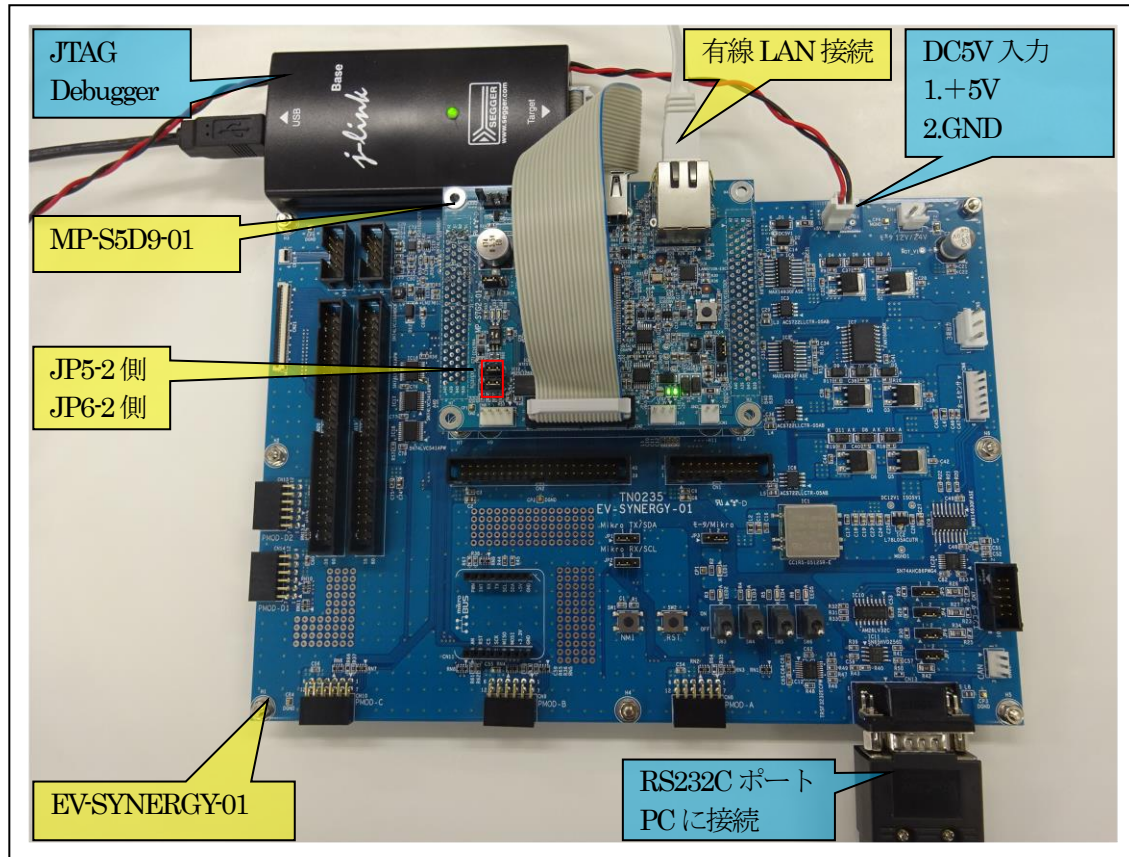


USB シリアルコンバー
タ使用時に CR コ
ードがカットされる
設定の場合は、**受
信 : LF** にして下さ
い。

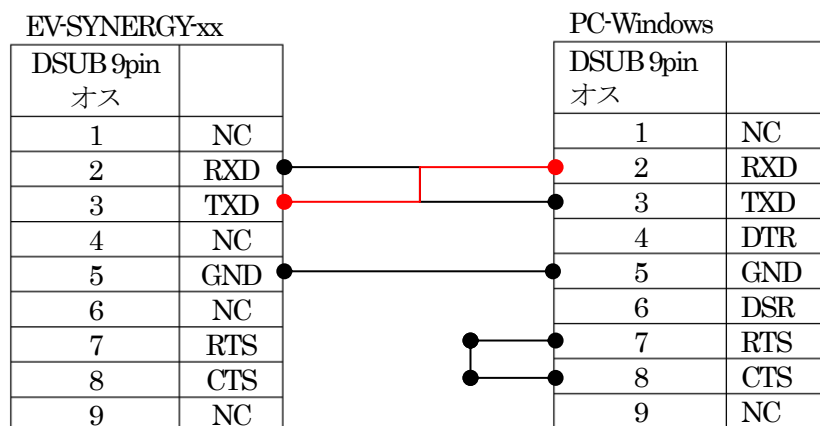
赤枠の設定にする。

4. 動作構成

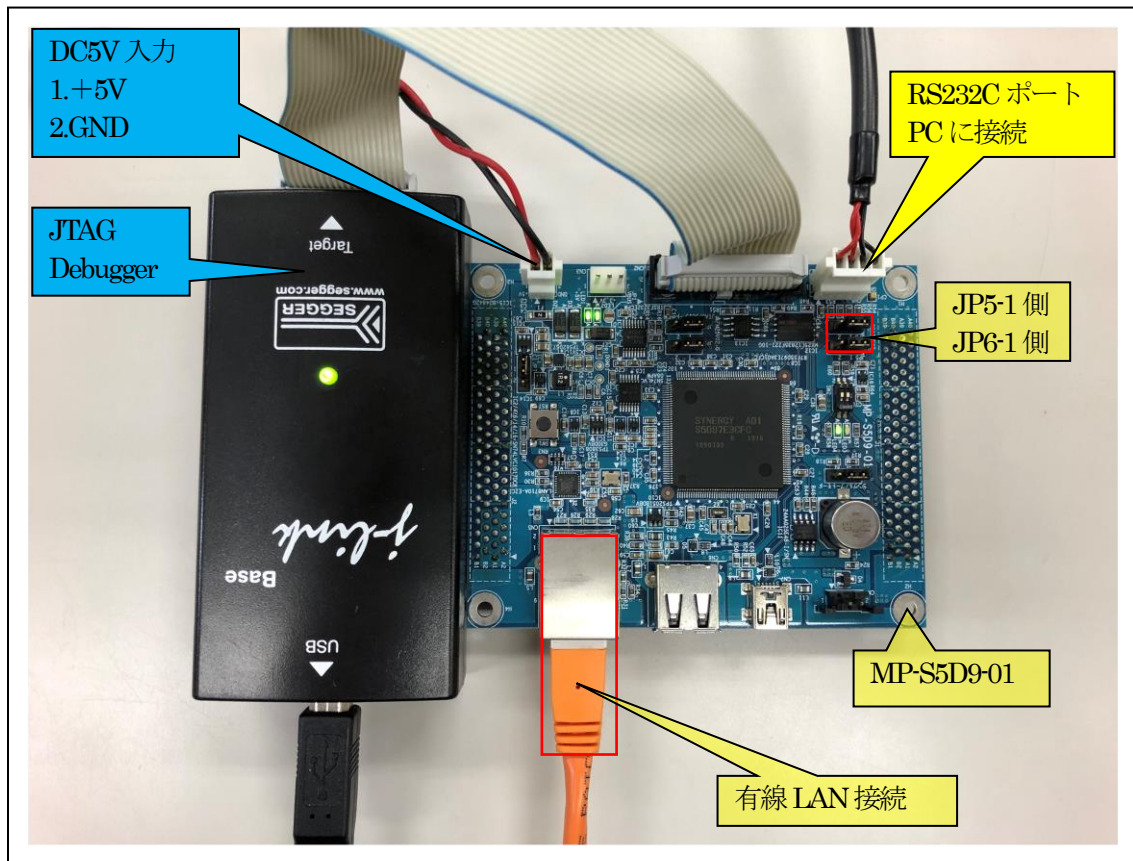
【EV-SYNERGY01】を使用の場合



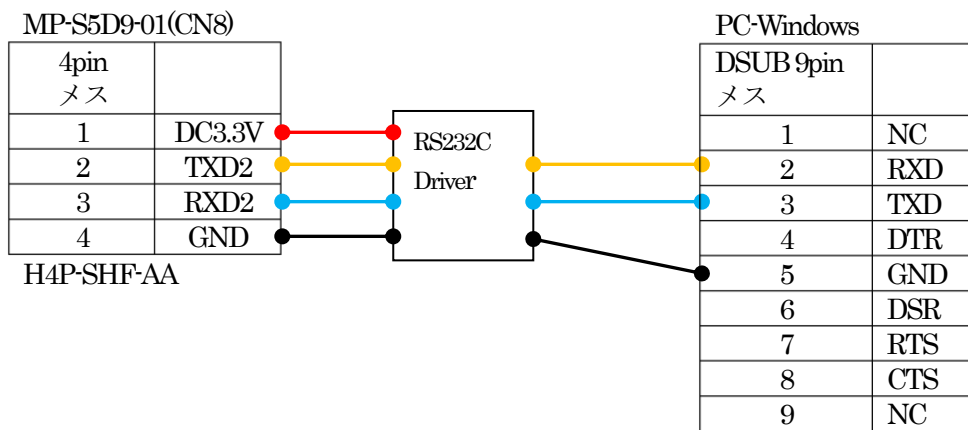
- ①PC機と接続するRS232Cケーブルは、市販「クロスケーブル」でも可能です。
- ②USB-シリアル変換ケーブルを使用される場合は、「StarTech.com社 ICUSB232FIN」推奨
- ③自作する場合は、下記の配線になります。



【MP-S5D9-01】のみ使用の場合



- ①PC機と接続するRS232Cケーブルは、製作が必要です。
- ②「RS232C-Driver」は、下記URLの「RS232CAB4」を推奨します。
http://tool-kobo.ddo.jp/Files/Product/RS232_422/RS232CAB.htm



5. 「S5D9_e2std_WIRE_NetX_1」サンプルの説明

5-1. 「WIRE_DHCP_TCP_RSA_ETH」フォルダ構成とファイル名

S5D9_e2std_WIRE_NetX_1\WIRE_DHCP_TCP_RSA_ETH		
Debug	WIRE_DHCP_TCP_RSA_ETH.elf	ELF ファイル、JTAG で使用
	WIRE_DHCP_TCP_RSA_ETH.map	MAP ファイル、アドレス情報管理
	WIRE_DHCP_TCP_RSA_ETH.srec	モトローラーHEX ファイル
	その他	自動生成ファイル
Script	S5D9.ld	ロケーション定義ファイル
Src	Command.c	コマンド入力処理のサンプルファイル
	Command.h	Command.c 用ヘッダーファイル
	tcp_cmd_thread_entry.c	TCP Cmd Thread サンプルファイル
	wire_dhcp_thred_entry.c	DHCP client Thread サンプルファイル
sce7_rsa	sce7_rsa.c	RSA の暗号・復号化サンプルファイル
	sce7_rsa.h	sce7_rsa.c 用ヘッダーファイル
MP-S5D9-01 (リンク指定)	e2p.c	E2PROM 処理モジュール
	e2p.h	e2p.c 用ヘッダーファイル
	led.c	LED 処理モジュール
	led.h	led.c 用ヘッダーファイル
	sci2.c	シリアル通信処理モジュール
	sci2.h	sci2.c 用ヘッダーファイル
	stchar.c	文字系処理モジュール
	stcahr.h	stchar.c 用ヘッダーファイル
synergy_gen	Generate を行うと作成されるファイル	
Synergy	Generate を行うと作成されるファイル	
synergy_cfg		
Configuration.xml	プロジェクト Generation ファイル	
PIN-MP-S5D9-01.pincfg	PIN configuration 用ファイル	
WIRE_DHCP_TCP_RSA_ETH.jlink	Jlink デバッガー用ファイル	
その他	自動生成ファイル	

5-2. サンプルの動作説明

<WIRE DHCP Thread>

1) DHCP による IP アドレスの取得を待つ

Term 画面

< 1 > 「<tcp_thread waiting to get IP address>」

< 2 > 「<interface status check>」

MP 基板に実装した EEPROM より MAC アドレス読みドライバにセットする。

< 3 > 「<Start WIRE NetX DHCP>」

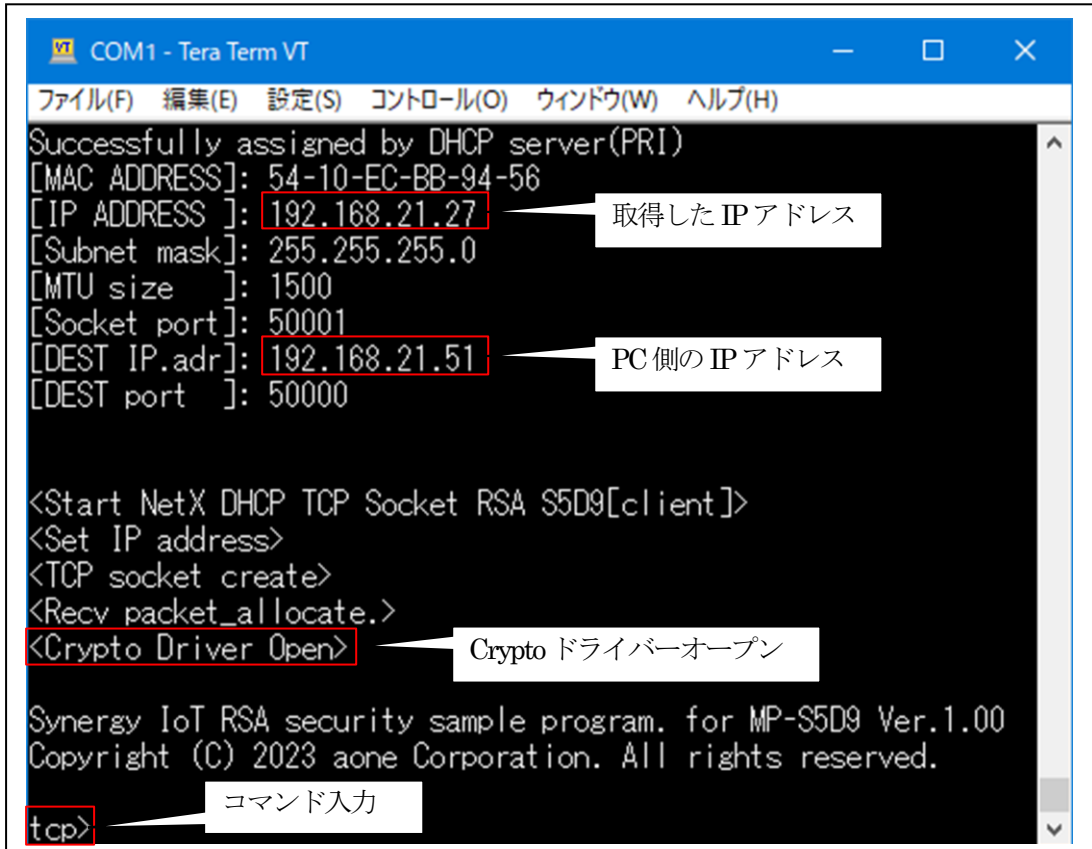
< 4 > 「<Wait DHCP State Change>」

< 5 > 「<Start WIRE NetX DHCP>」

< 6 > 「<Wait DHCP BOUND>」 の順次処理して表示する。

・IP アドレス取得成功により、MP 基板上の LED3 を led blink thread で 100msec 毎に点滅

2) IP ドレス取得情報を Term 画面に表示する。



```

COM1 - Tera Term VT
ファイル(F) 編集(E) 設定(S) コントロール(O) ウィンドウ(W) ヘルプ(H)
Successfully assigned by DHCP server(PRI)
[MAC ADDRESS]: 54-10-EC-BB-94-56
[IP ADDRESS ]: 192.168.21.27
[Subnet mask]: 255.255.255.0
[MTU size ]: 1500
[Socket port]: 50001
[DEST IP.adr]: 192.168.21.51
[DEST port ]: 50000

<Start NetX DHCP TCP Socket RSA S5D9[client]>
<Set IP address>
<TCP socket create>
<Recv packet_allocate.>
<Crypto Driver Open>

Synergy IoT RSA security sample program. for MP-S5D9 Ver.1.00
Copyright (C) 2023 aone Corporation. All rights reserved.

tcp>
  
```

取得した IP アドレス

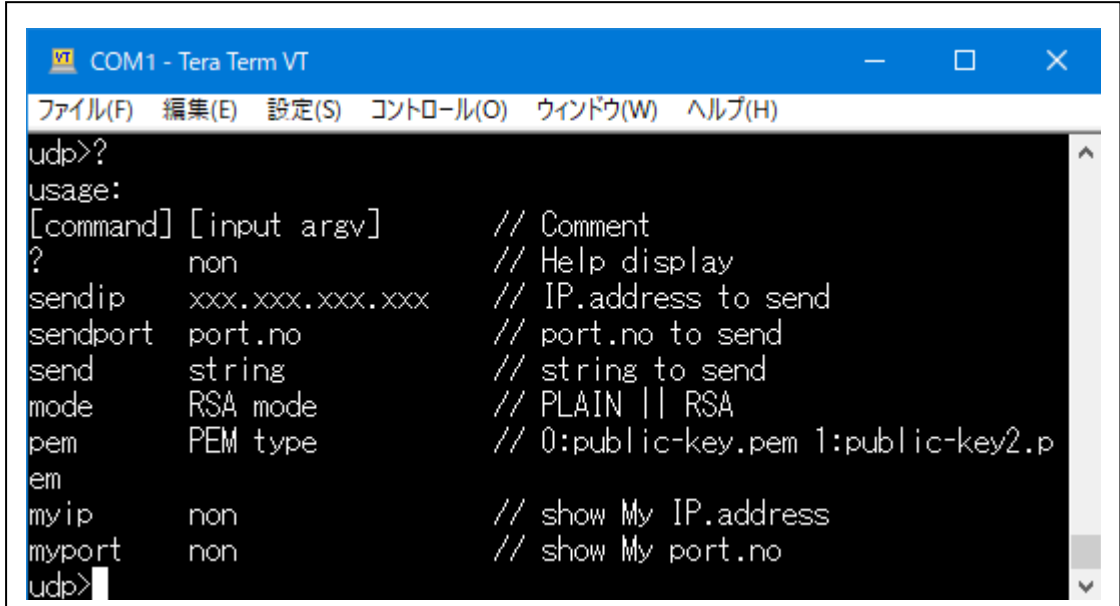
PC 側の IP アドレス

Crypto ドライバーオープン

コマンド入力

<TCP Cmd Thread>

- 3) オープニングメッセージを表示してコマンド入力を待つ。
 - ・TCPThread 起動後、MP 基板上の LED4 を 100msec 毎に点滅
- 4) コマンドの説明
 - (1) ? (ヘルプコマンド) コマンド一覧の表示



```

COM1 - Tera Term VT
ファイル(F) 編集(E) 設定(S) コントロール(O) ウィンドウ(W) ヘルプ(H)
udp>?
usage:
[command] [input argv] // Comment
? non // Help display
sendip xxx.xxx.xxx.xxx // IP.address to send
sendport port.no // port.no to send
send string // string to send
mode RSA mode // PLAIN || RSA
pem PEM type // 0:public-key.pem 1:public-key2.pem
myip non // show My IP.address
myport non // show My port.no
udp>
  
```

- (2) `sendip` (送信先 IP アドレスの登録と参照)

<機能>

- ・送信先 (PC 機側) の IP アドレスを登録する。

<例>

```

xxx> sendip 192.168.21.9↵
xxx> sendip↵
[send IP]: 192.168.21.51
  
```

- (3) `sendport` (送信先ポート番号の登録と参照)

<機能>

- ・送信先 (PC 機側) のポート番号を登録する。

<例>

```

xxx> sendport 50000↵
xxx> sendport↵
[send port]: 50000
  
```

- (4) `send` (文字列の送信)

<機能>

- ・送信先 (PC 機側) に文字列を送信する。

<例>

```

xxx> send 12345678123456781234567812345678↵
  
```

(5) mode (暗号モードの設定と参照)

<機能>

- ・送信先 (PC機側) に文字列を送信するための暗号モードを指定する。

①<PLAIN> 平文で送信

②<RSA>文字列を「RSA モード」で暗号化して送信する。

<例>

```
xxx> mode PLAIN↵
```

```
xxx> mode RSA↵
```

```
xxx> mode↵
```

```
[Now mode]: RSA
```

(6) pem (PEMタイプの指定)

<機能>

- ・暗号・復号化するための公開鍵・秘密鍵の PEM タイプを指定する。

①<0> (公開鍵)public-key.pem : (秘密鍵) private-key.pem を使用する。

②<1> (公開鍵)public-key2.pem : (秘密鍵) private-key2.pem を使用する。

<例>

```
xxx> pem 0↵
```

```
xxx> pem 1↵
```

(7) myip (DHCP で取得した自 IP アドレスの参照)

<機能>

- ・DHCP で取得した自 IP アドレスを表示する。

<例>

```
xxx> myip↵
```

```
[My IP]: 192.168.21.9
```

(8) myport (基板側で登録したポート番号の参照)

<機能>

- ・基板側で登録したポート番号を表示する。

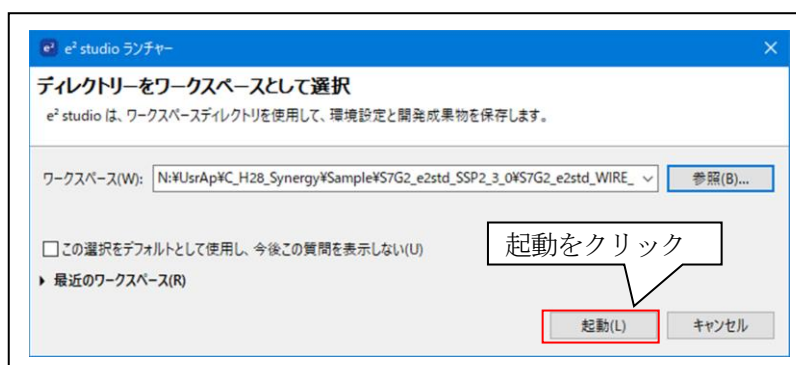
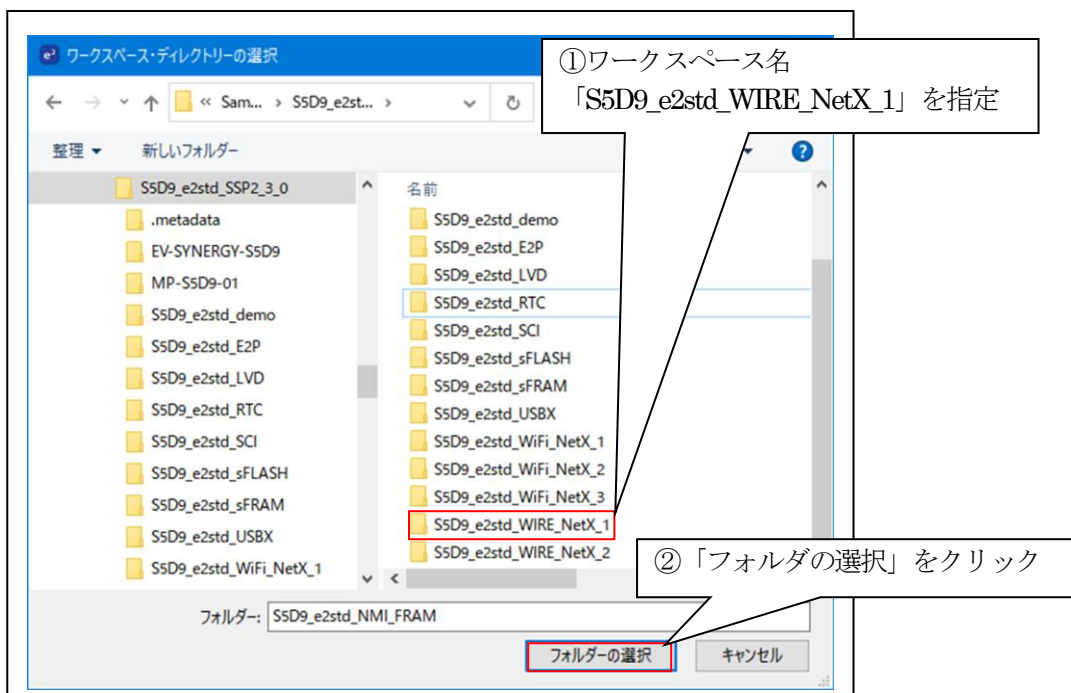
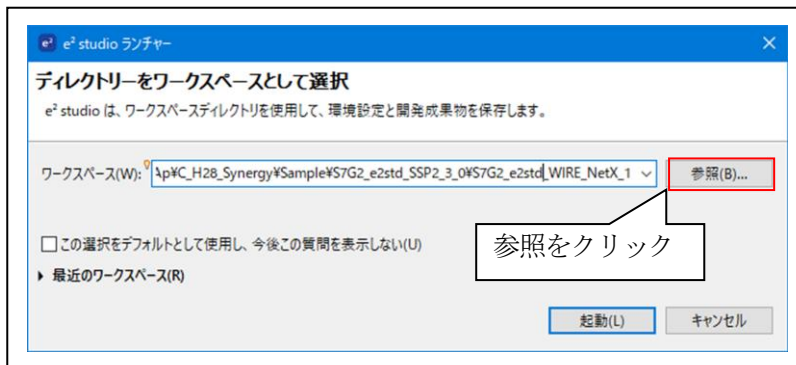
<例>

```
xxx> myport↵
```

```
[My port]: 50001
```


6. 「S5D9_e2std_WIRE_NetX_1」のワークスペースを指定する。

6-1. ワークスペース名の指定



6-2. プロジェクトのインポート

☆詳細操作は「[e2studio_synergy_Import.pdf](#)」の2項を参照して下さい。

7. デバッグ操作

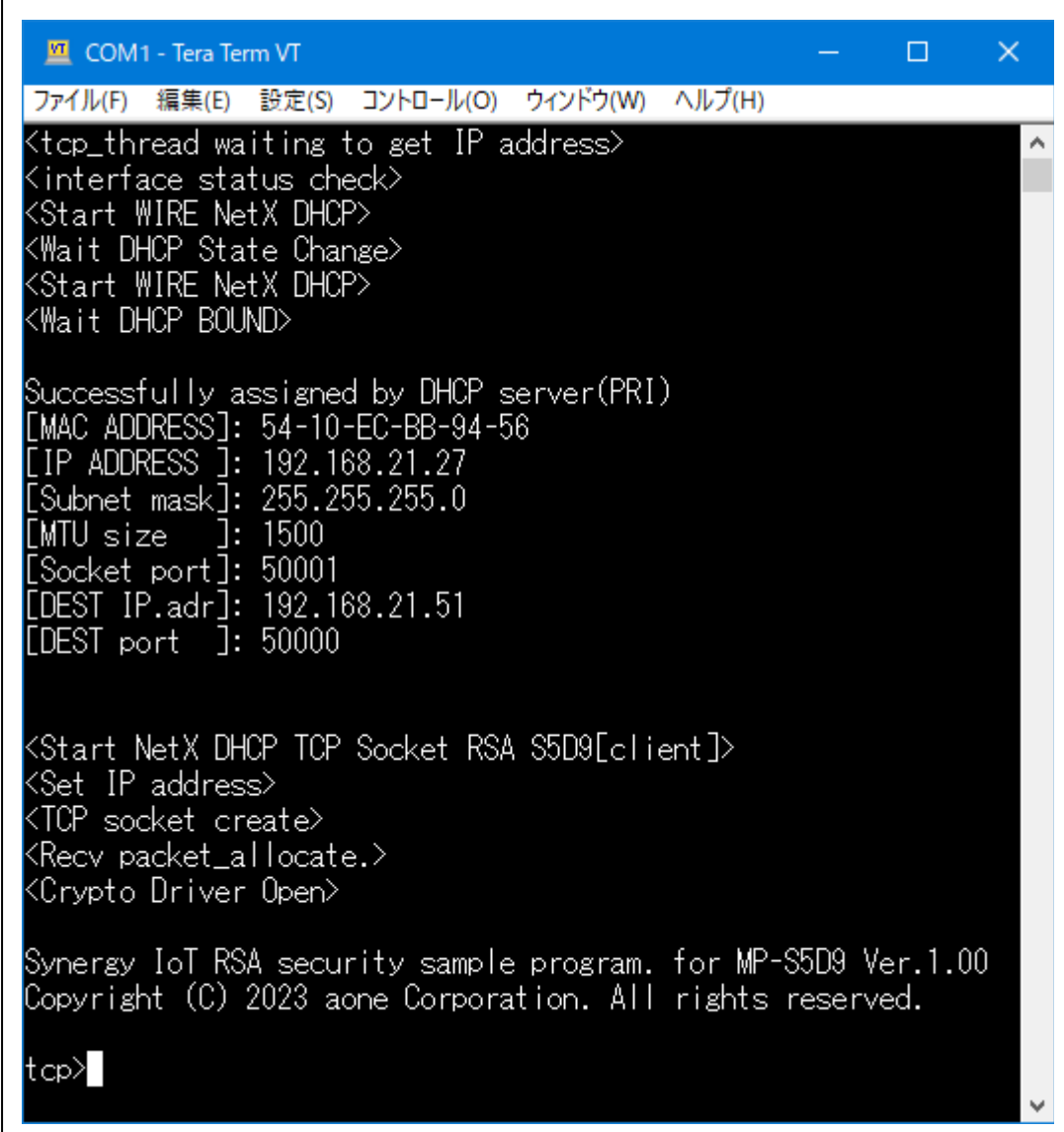
7-1. デバッグ構成の設定

☆詳細操作は「[e2studio_synergy_Import.pdf](#)」の3-1項を参照して下さい。

7-2. デバッグの開始

☆詳細操作は「[e2studio_synergy_Import.pdf](#)」の3-2項を参照して下さい。

<WIRE_DHCP_TCP_RSA 実行画面>



```

COM1 - Tera Term VT
ファイル(F) 編集(E) 設定(S) コントロール(O) ウィンドウ(W) ヘルプ(H)
<tcp_thread waiting to get IP address>
<interface status check>
<Start WIRE NetX DHCP>
<Wait DHCP State Change>
<Start WIRE NetX DHCP>
<Wait DHCP BOUND>

Successfully assigned by DHCP server(PRI)
[MAC ADDRESS]: 54-10-EC-BB-94-56
[IP ADDRESS ]: 192.168.21.27
[Subnet mask]: 255.255.255.0
[MTU size   ]: 1500
[Socket port]: 50001
[DEST IP.adr]: 192.168.21.51
[DEST port  ]: 50000

<Start NetX DHCP TCP Socket RSA S5D9[client]>
<Set IP address>
<TCP socket create>
<Recv packet_allocate.>
<Crypto Driver Open>

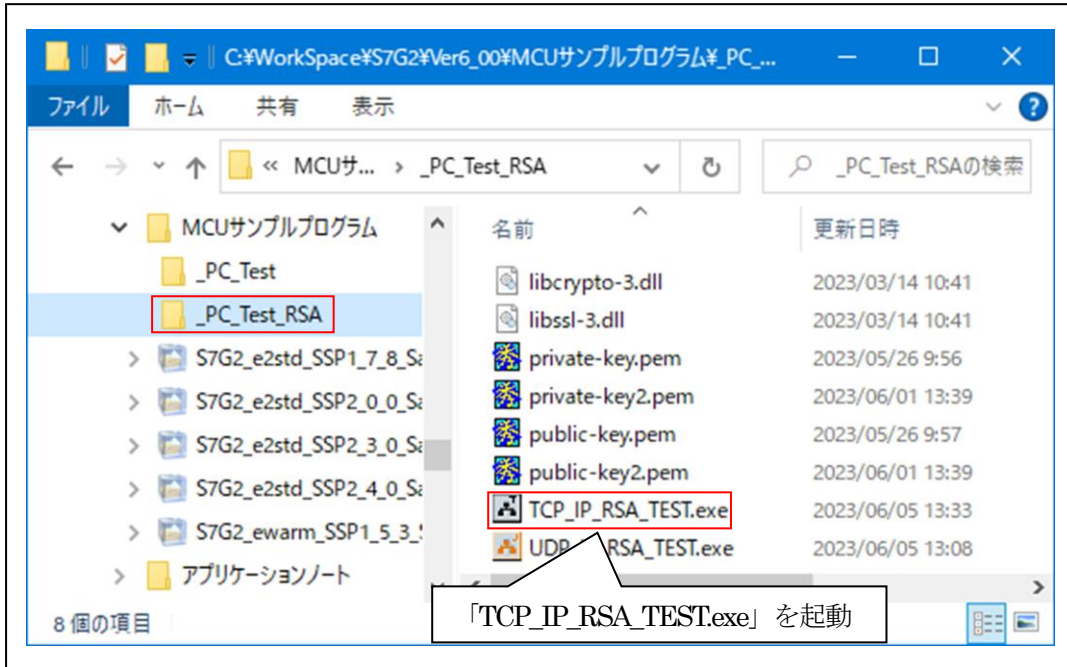
Synergy IoT RSA security sample program. for MP-S5D9 Ver.1.00
Copyright (C) 2023 aone Corporation. All rights reserved.

tcp>
  
```

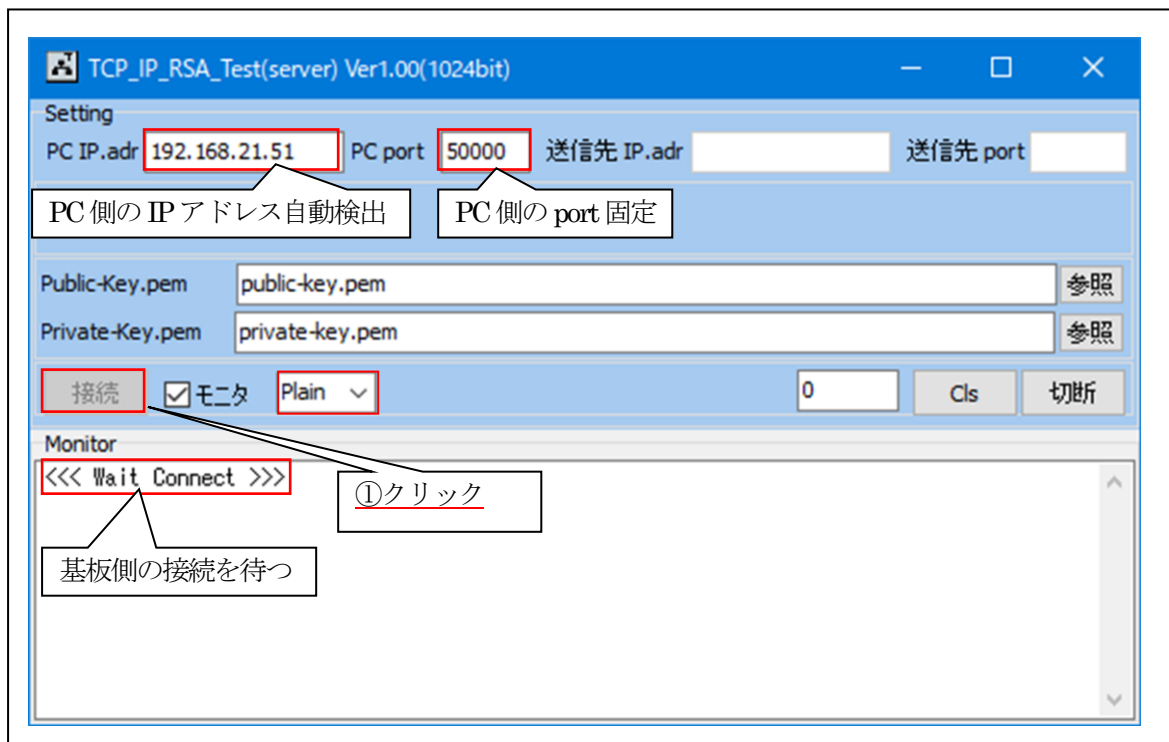
7-3. Windows PC側のテスト用プログラムで動作確認 (PLAIN (平文) モード)

1) 「TCP_IP_RSA_TEST」を起動する。

プログラム場所【ご購入 CD¥MCU サンプルプログラム¥_PC_Test_RSA】



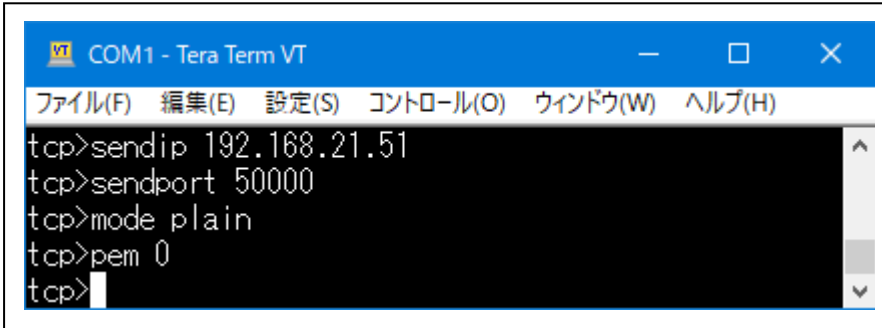
2) 「TCP_IP_RSA_TEST」の各項目を設定して「基板」側からの「接続」を待つ。



3) 「基板」側の各項目の確認と設定。

- ①PC 側の IP アドレスを設定する。
ex) sendip 192.168.21.51<↵
- ②PC 側のポート番号を設定する。
ex) sendport 50000<↵
- ③ 「plain」 (平文) モードを指定する。
ex) mode plain<↵
- ④ 「pem」 (PEM) タイプを指定する。
ex) pem 0<↵

⑤実行画面



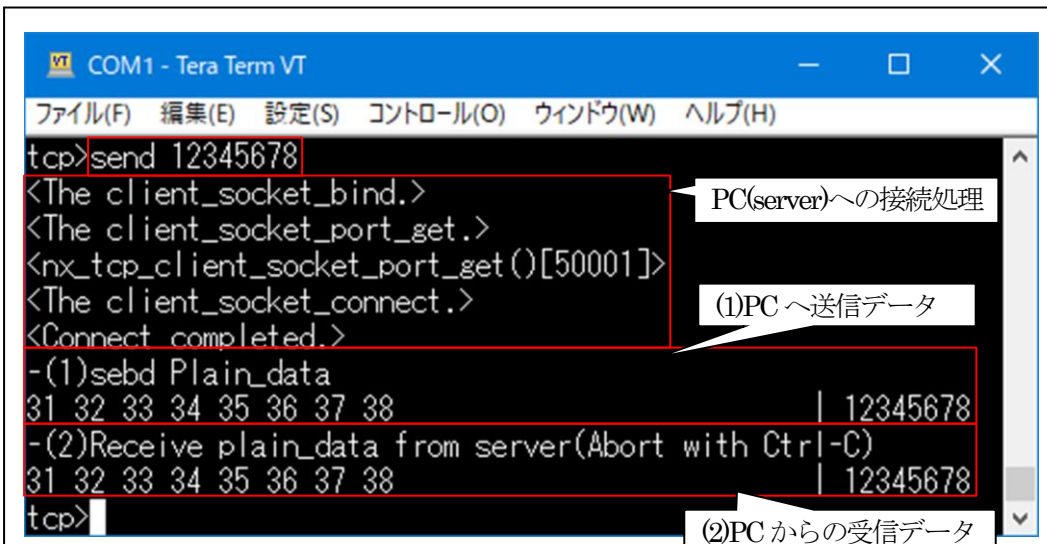
```

COM1 - Tera Term VT
ファイル(F) 編集(E) 設定(S) コントロール(O) ウィンドウ(W) ヘルプ(H)
tcp>sendip 192.168.21.51
tcp>sendport 50000
tcp>mode plain
tcp>pem 0
tcp>
  
```

4) 「基板」側から「PC」(server)側へ平文テキストを送信する。

- ①テキスト文を送信する。(未接続の場合は、接続を実行)

ex) send 12345678



```

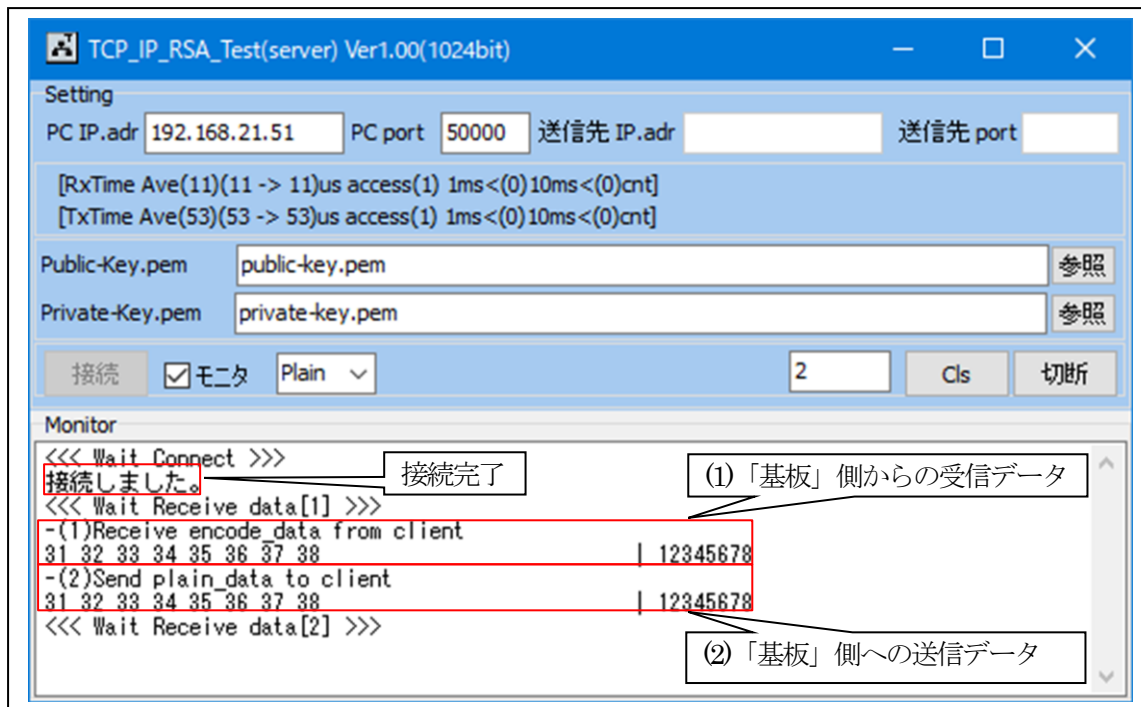
COM1 - Tera Term VT
ファイル(F) 編集(E) 設定(S) コントロール(O) ウィンドウ(W) ヘルプ(H)
tcp>send 12345678
<The client_socket_bind.>
<The client_socket_port_get.>
<nxtcp_client_socket_port_get()[50001]>
<The client_socket_connect.>
<Connect completed.>
-(1)sebd Plain_data
31 32 33 34 35 36 37 38 | 12345678
-(2)Receive plain_data from server(Abort with Ctrl-C)
31 32 33 34 35 36 37 38 | 12345678
tcp>
  
```

PC(server)への接続処理

(1)PC へ送信データ

(2)PC からの受信データ

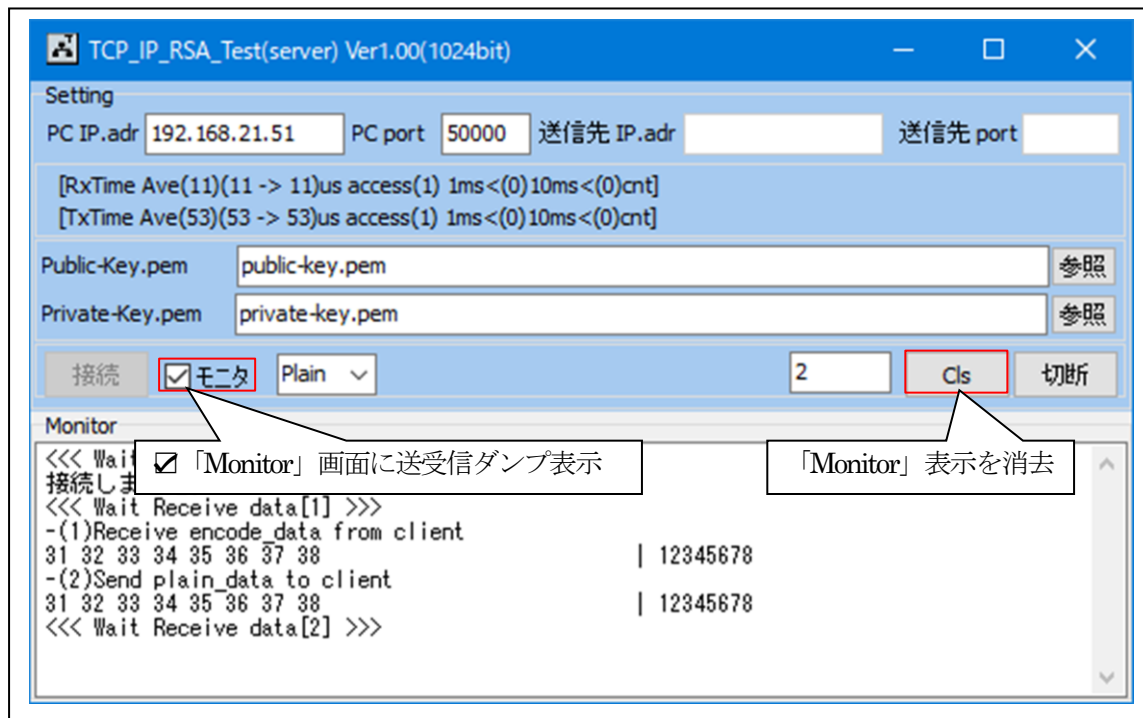
5) 「TCP_IP_RSA_TEST」側の送受信を確認する。



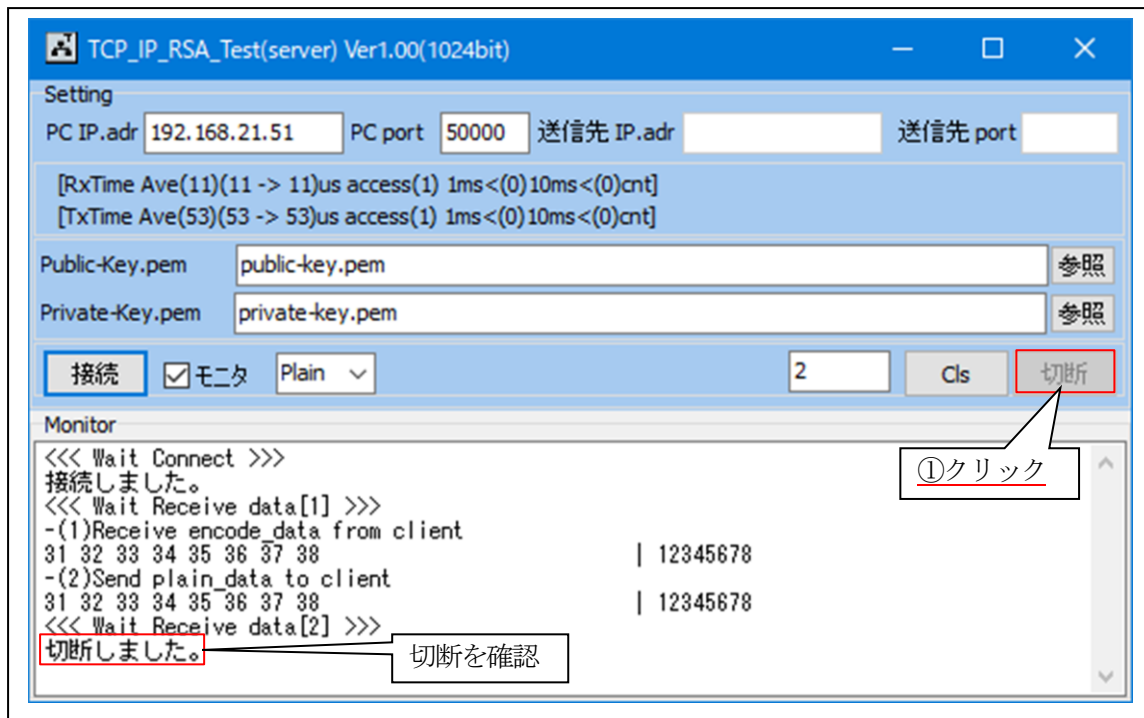
【Error 表示】

- 受信時の接続失敗 「"error!! client_socket_connect() [error code]"」
- 受信異常 「"error!! recvfrom() [error code]"」
- 送信異常 「"error!! sendto() [error code]"」

6) 「TCP_IP_RSA_TEST」その他の操作



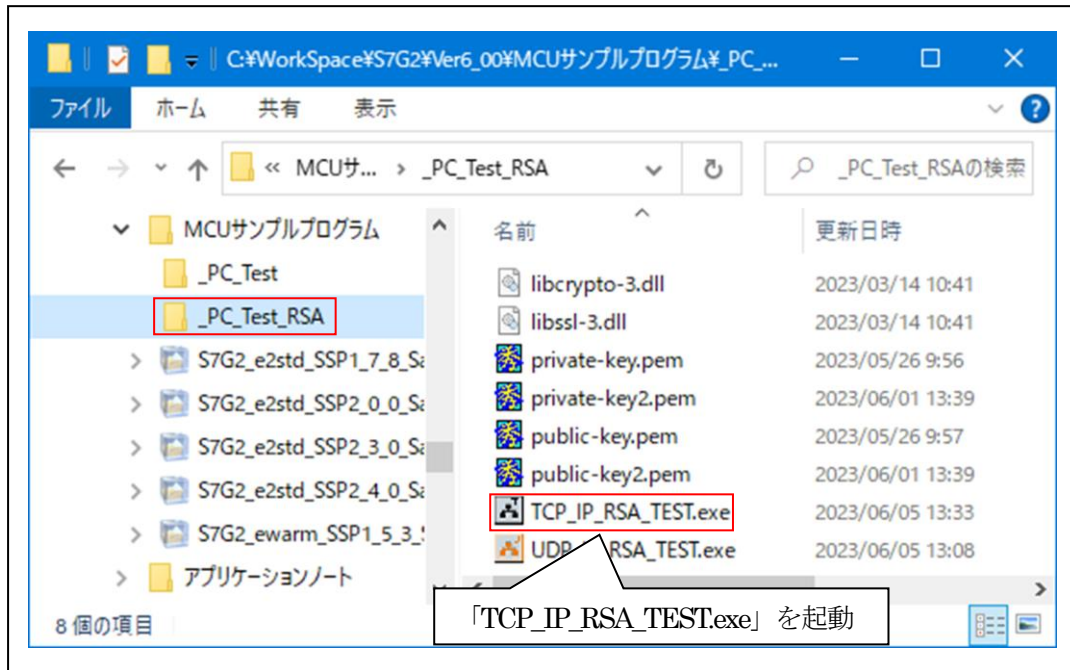
7) TCP_IP-Portを「切断」する。



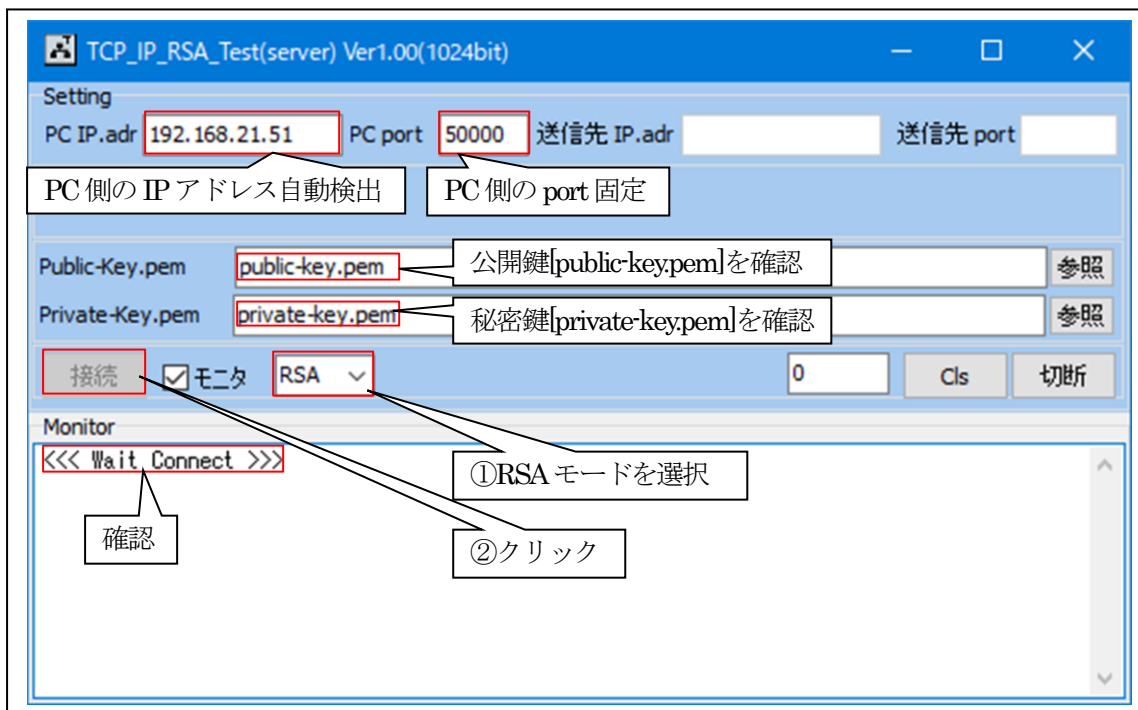
7-4. Windows PC 側のテスト用プログラムで動作確認 (RSA モード/public-keypem タイプ)

1) 「TCP_IP_RSA_TEST」を起動する。

プログラム場所【ご購入 CD¥MCU サンプルプログラム¥_PC_Test_RSA】



2) 「TCP_IP_RSA_TEST」の各項目を設定して「基板」側からの「接続」を待つ。



3) 「基板」側の各項目の確認と設定。

①PC 側の IP アドレスを設定する。

ex) sendip 192.168.21.51<↵

②PC 側のポート番号を設定する。

ex) sendport 50000<↵

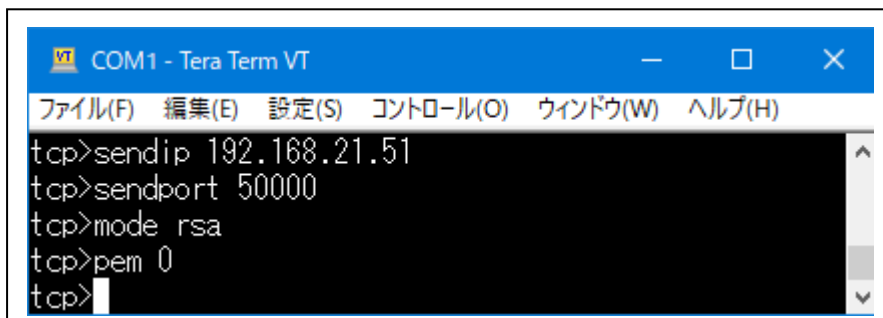
③ 「rsa」 (RSA) モードを指定する。

ex) mode rsa<↵

④ 「pem」 (publik-key.pem) タイプを指定する。

ex) pem 0<↵

⑤実行画面



The screenshot shows a terminal window titled "COM1 - Tera Term VT". The menu bar includes "ファイル(F)", "編集(E)", "設定(S)", "コントロール(O)", "ウインドウ(W)", and "ヘルプ(H)". The terminal content shows the following commands and their execution:

```

tcp>sendip 192.168.21.51
tcp>sendport 50000
tcp>mode rsa
tcp>pem 0
tcp>
  
```


4) 「基板」側から「PC」(server)側へRSA暗号テキストを送信する。

①テキスト文を送信する。(未接続の場合は、接続を実行)

ex) send 12345678

```

COM1 - Tera Term VT
ファイル(F) 編集(E) 設定(S) コントロール(O) ウィンドウ(W) ヘルプ(H)
tcp>send 12345678
<The client_socket_bind.>
<The client_socket_port_get.>
<nx_tcp_client_socket_port_get()[50001]>
<The client_socket_connect.>
<Connect completed.>
-(1)Plain data
31 32 33 34 35 36 37 38 20 20 20 20 20 20 20 20 20 20 20 20 | 12345678.....
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 | .....
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 | .....
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 | .....
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 | .....
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 | .....
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 | .....
-(2)Send encode_data to server
41 B4 71 92 34 65 8F EF 9B 3B FC 0F 33 72 DC 50 59 7D DC 7A 60 71 F1 1B 9E 65 D1 81 71 85 8B A0 3C 8B 8B 9F 01 F7 5B 13 7A 0E 36 F3 10 33 CE 7F 49 46 D9 04 AD B2 57 FF DF 71 1A 9B 2F 34 F2 86 15 0B 6F 78 F6 2E 15 EC 30 9B BD B1 F3 93 AF 99 A9 C7 AD AA F5 83 DC 79 D3 CE 01 BF BA 29 33 33 9C 82 12 6E C4 FD 92 ED B9 46 81 D2 0B B0 E5 B0 C4 21 65 56 7C 26 17
.z.6..3ホ.1Fw.ユ1w
.°q../4...ox...
.0.アア..ツ.ウヌエ..7
yモホ.ソコ)33...nt..
.ガF.メ.-.ト!eV|&
-(3)Receive encode_data from server
41 B4 71 92 34 65 8F EF 9B 3B FC 0F 33 72 DC 50 59 7D DC 7A 60 71 F1 1B 9E 65 D1 81 71 85 8B A0 3C 8B 8B 9F 01 F7 5B 52 C4 8A 13 7A 0E 36 F3 10 33 CE 7F 49 46 D9 04 AD B2 57 FF DF 71 1A 9B 2F 34 F2 86 15 0B 6F 78 F6 2E 15 EC 30 9B BD B1 F3 93 AF 99 A9 C7 AD AA F5 83 DC 79 D3 CE 01 BF BA 29 33 33 9C 82 12 6E C4 FD 92 ED B9 46 81 D2 0B B0 E5 B0 C4 21 65 56 7C 26 17
4.q...<.....[Rト
.z.6..3ホ.1Fw.ユ1w
.°q../4...ox...
.0.アア..ツ.ウヌエ..7
yモホ.ソコ)33...nt..
.ガF.メ.-.ト!eV|&
-(4)Create decode_data
31 32 33 34 35 36 37 38 20 20 20 20 20 20 20 20 20 20 20 20 | .....
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 | .....
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 | .....
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 | .....
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 | .....
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 | .....
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 | .....
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 | .....
tcp>
    
```

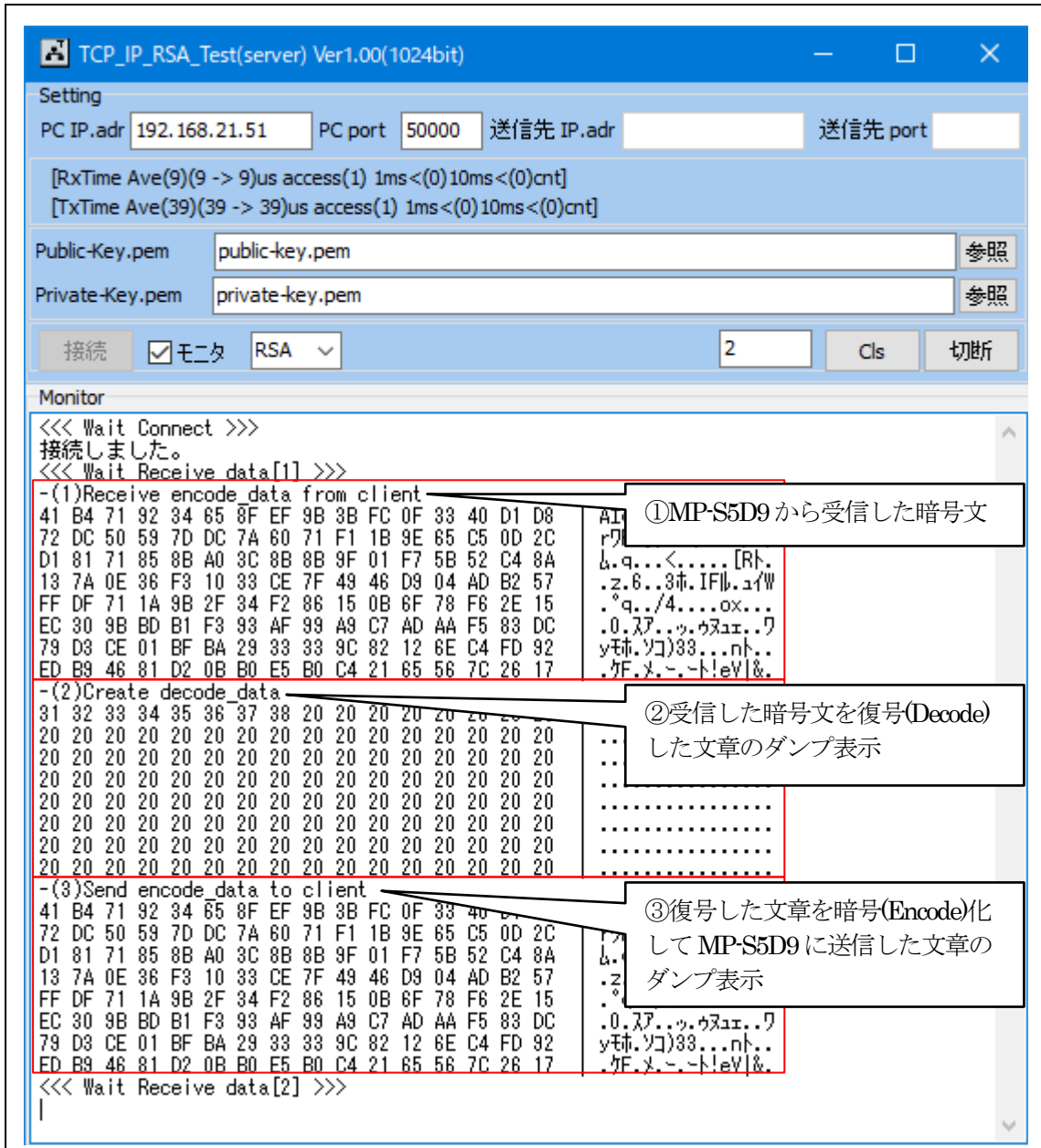
①コマンド入力した平文のダンプ表示

②平文を暗号化 (Encrypt) した文章を PC (サーバー) に送信した文章のダンプ表示

③PC (サーバー) から暗号文を受信したダンプ表示

④受信した暗号文を復号 (Decrypt) した文章のダンプ表示
①と④が同等であることを確認し、PC (サーバー) 側の②とも同等であることを確認する。

5) 「TCP_IP_RSA_TEST」側の送受信を確認する。



The screenshot shows the 'TCP_IP_RSA_Test(server) Ver1.00(1024bit)' application. The 'Setting' section includes fields for PC IP.adr (192.168.21.51), PC port (50000), and fields for destination IP and port. It also shows timing statistics for reception and transmission, and fields for Public-Key.pem and Private-Key.pem. The 'Monitor' section displays the following log output:

```

  <<< Wait Connect >>>
  接続しました。
  <<< Wait Receive data[1] >>>
  -(1)Receive encode_data from client
  41 B4 71 92 34 65 8F EF 9B 3B FC 0F 33 40 D1 D8
  72 DC 50 59 7D DC 7A 60 71 F1 1B 9E 65 C5 0D 2C
  D1 81 71 85 8B A0 3C 8B 8B 9F 01 F7 5B 52 C4 8A
  13 7A 0E 36 F3 10 33 CE 7F 49 48 D9 04 AD B2 57
  FF DF 71 1A 9B 2F 34 F2 86 15 0B 6F 78 F8 2E 15
  EC 30 9B BD B1 F3 93 AF 99 A9 C7 AD AA F5 83 DC
  79 D3 CE 01 BF BA 29 33 33 9C 82 12 6E C4 FD 92
  ED B9 46 81 D2 0B B0 E5 B0 C4 21 65 56 7C 26 17
  ①MP-S5D9 から受信した暗号文

  -(2)Create decode_data
  31 32 33 34 35 36 37 38 20 20 20 20 20 20 20 20
  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
  ②受信した暗号文を復号(Decode)
  した文章のダンプ表示

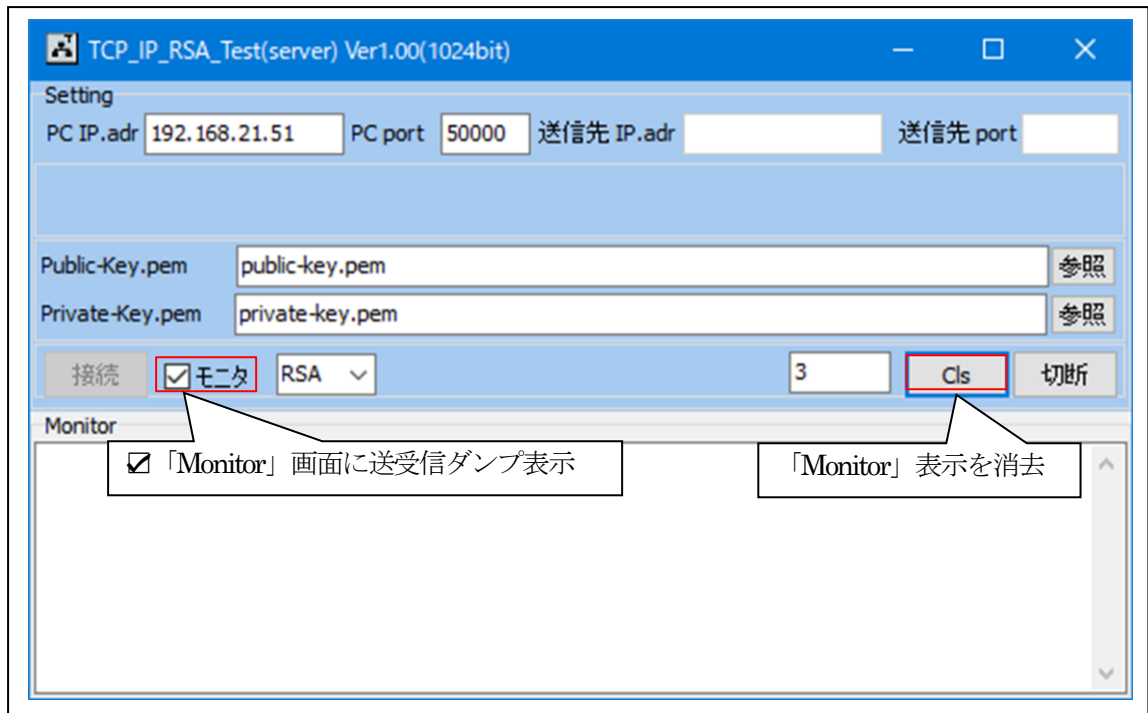
  -(3)Send encode_data to client
  41 B4 71 92 34 65 8F EF 9B 3B FC 0F 33 40 D1 D8
  72 DC 50 59 7D DC 7A 60 71 F1 1B 9E 65 C5 0D 2C
  D1 81 71 85 8B A0 3C 8B 8B 9F 01 F7 5B 52 C4 8A
  13 7A 0E 36 F3 10 33 CE 7F 49 48 D9 04 AD B2 57
  FF DF 71 1A 9B 2F 34 F2 86 15 0B 6F 78 F8 2E 15
  EC 30 9B BD B1 F3 93 AF 99 A9 C7 AD AA F5 83 DC
  79 D3 CE 01 BF BA 29 33 33 9C 82 12 6E C4 FD 92
  ED B9 46 81 D2 0B B0 E5 B0 C4 21 65 56 7C 26 17
  ③復号した文章を暗号(Encode)化
  して MP-S5D9 に送信した文章の
  ダンプ表示

  <<< Wait Receive data[2] >>>
  |
  
```

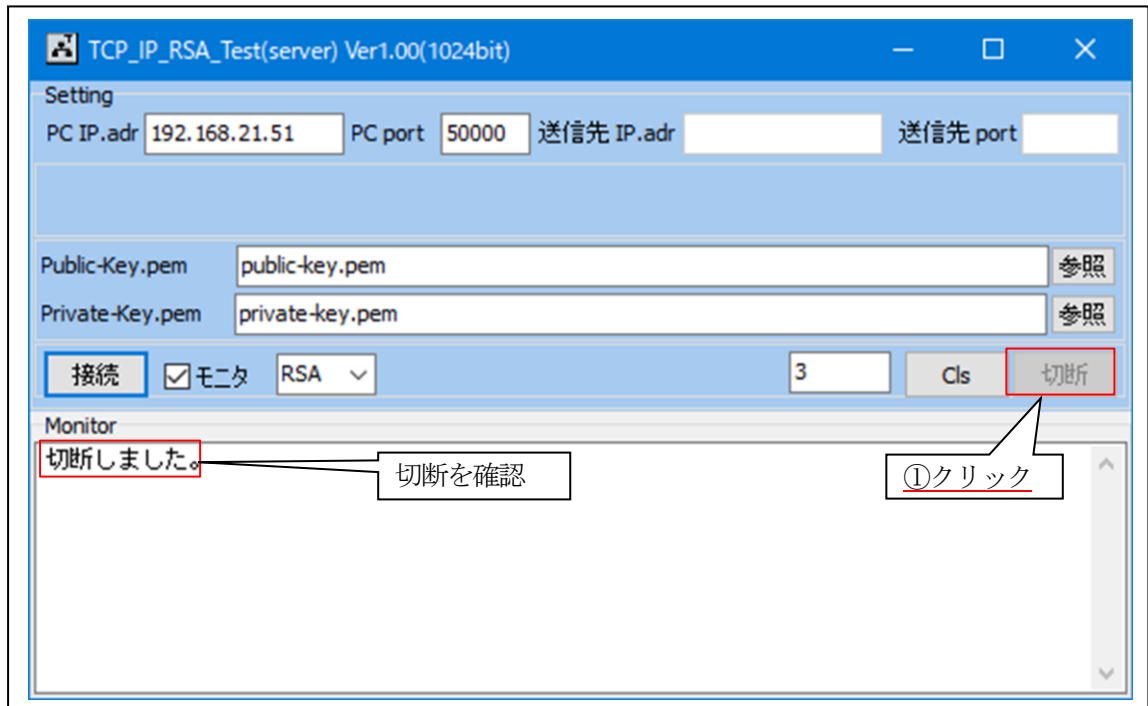
【Error 表示】

- 受信時の接続失敗 「"error!! client_socket_connect()[error code]"」
- 受信異常 「"error!! recvfrom()[error code]"」
- 送信異常 「"error!! sendto()[error code]"」

6) 「TCP_IP_RSA_TEST」 その他の操作



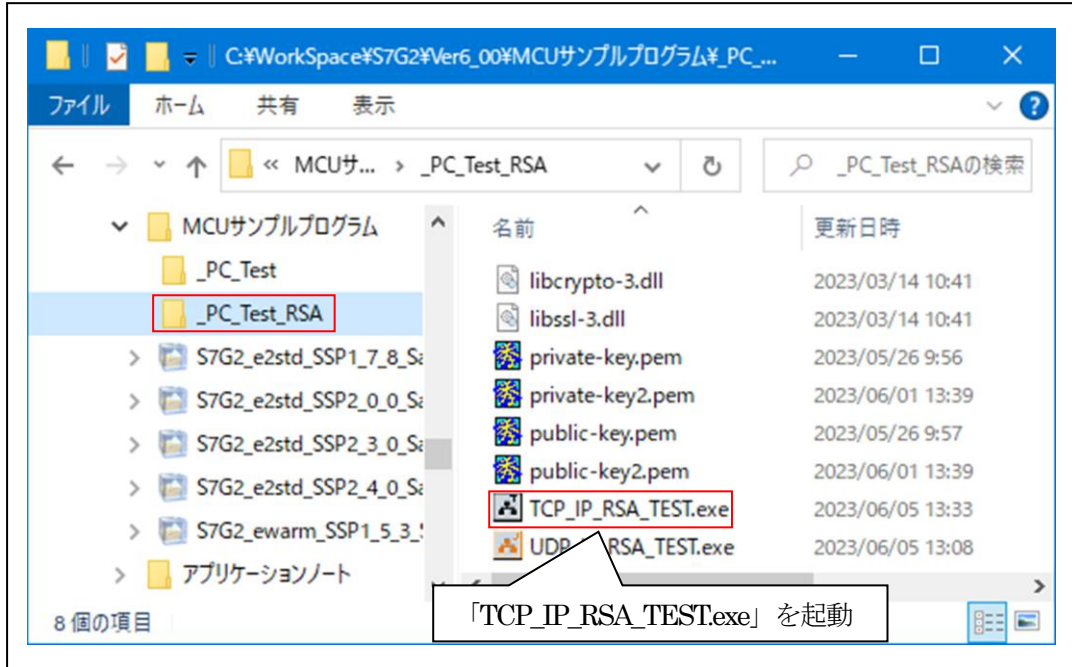
7) TCP_IP-Portを「切断」する。



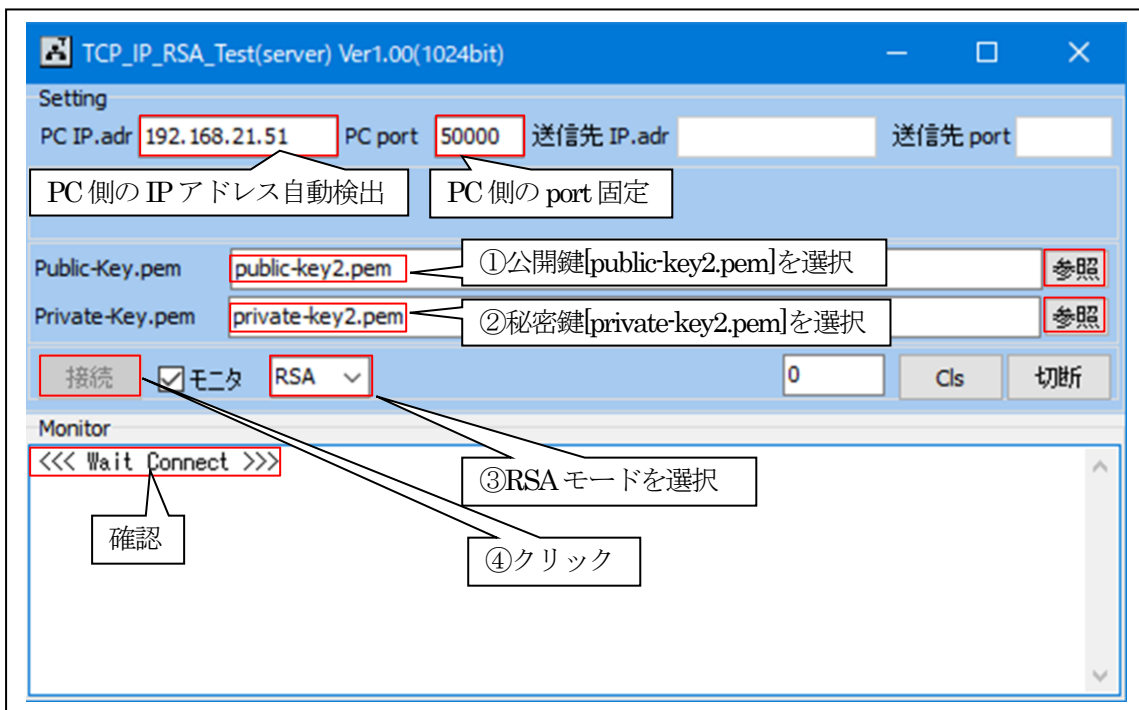
7-5. Windows PC 側のテスト用プログラムで動作確認 (RSA モード/public-key2.pem タイプ)

1) 「TCP_IP_RSA_TEST」を起動する。

プログラム場所【ご購入 CD¥MCU サンプルプログラム¥_PC_Test】



2) 「TCP_IP_RSA_TEST」の各項目を設定して「基板」側からの「接続」を待つ。



3) 「基板」側の各項目の確認と設定。

①PC 側の IP アドレスを設定する。

ex) sendip 192.168.21.51<↵

②PC 側のポート番号を設定する。

ex) sendport 50000<↵

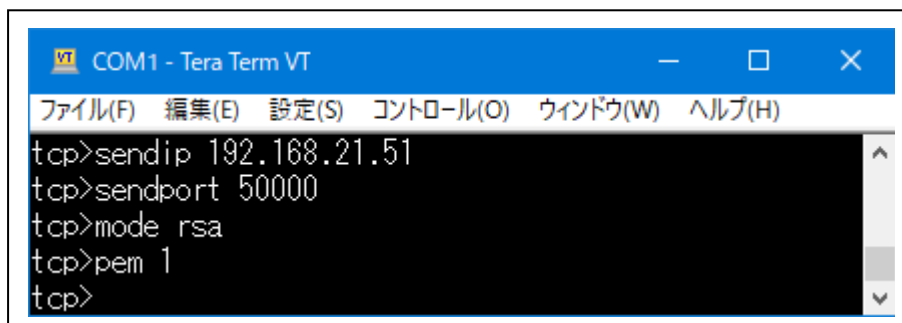
③ 「rsa」 (RSA) モードを指定する。

ex) mode rsa<↵

④ 「pem」 (public-key2.pem) タイプを指定する。

ex) pem 1<↵

⑤実行画面



The screenshot shows a terminal window titled "COM1 - Tera Term VT". The menu bar includes "ファイル(F)", "編集(E)", "設定(S)", "コントロール(O)", "ウィンドウ(W)", and "ヘルプ(H)". The terminal content shows the following commands and prompts:

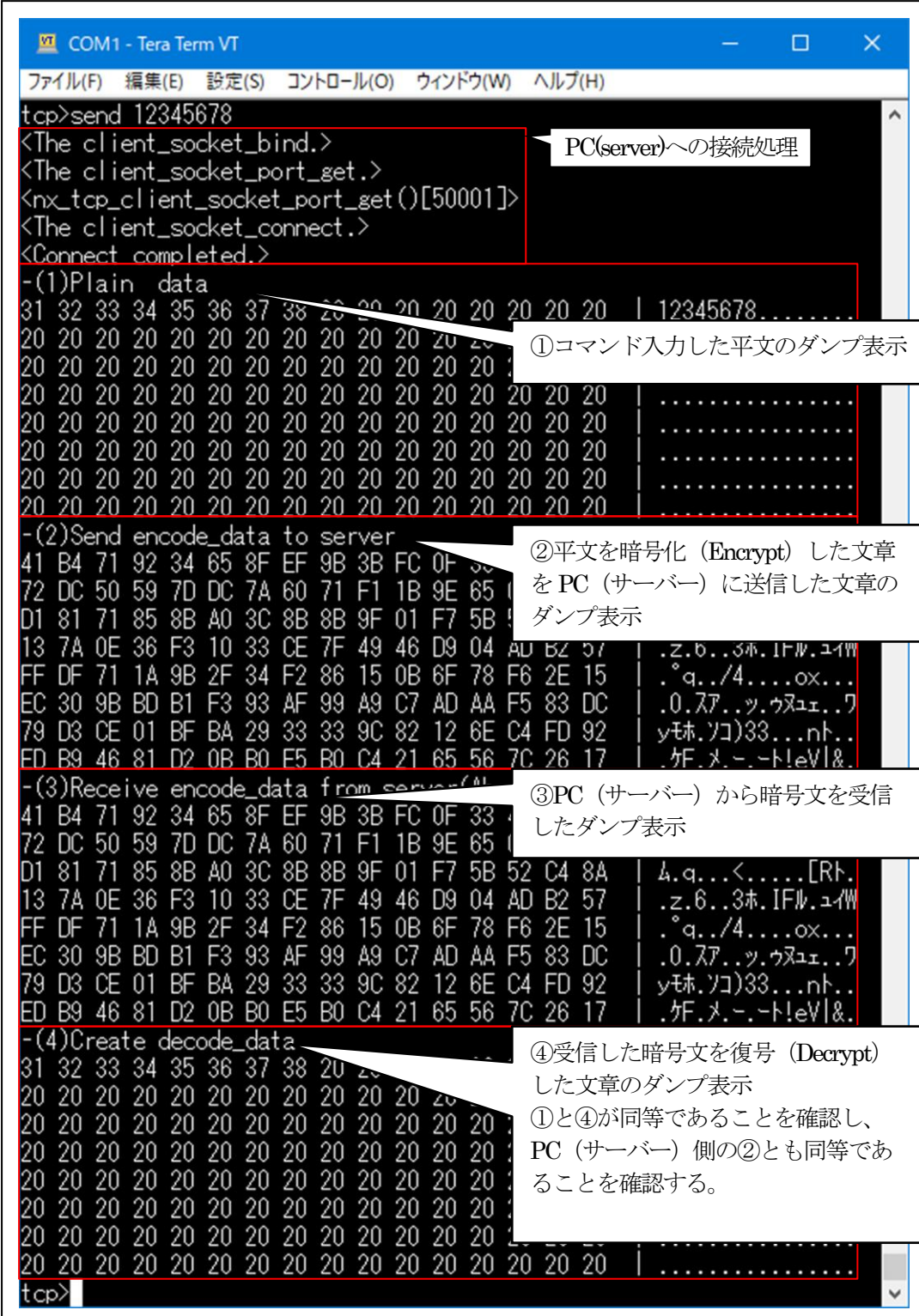
```

tcp>sendip 192.168.21.51
tcp>sendport 50000
tcp>mode rsa
tcp>pem 1
tcp>
  
```

4) 「基板」側から「PC」(server)側へRSA暗号テキストを送信する。

①テキスト文を送信する。(未接続の場合は、接続を実行)

ex) send 12345678



The screenshot shows a terminal window titled "COM1 - Tera Term VT" with the following output and annotations:

```

tcp>send 12345678
<The client_socket_bind.>
<The client_socket_port_get.>
<nx_tcp_client_socket_port_get()[50001]>
<The client_socket_connect.>
<Connect completed.>
-(1)Plain data
31 32 33 34 35 36 37 38 20 20 20 20 20 20 20 20 20 20 20 20 | 12345678.....
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
-(2)Send encode_data to server
41 B4 71 92 34 65 8F EF 9B 3B FC 0F 33 4D 5E 7D
72 DC 50 59 7D DC 7A 60 71 F1 1B 9E 65 40 57 7C
D1 81 71 85 8B A0 3C 8B 8B 9F 01 F7 5B 52 C4 8A
13 7A 0E 36 F3 10 33 CE 7F 49 46 D9 04 AD B2 57
FF DF 71 1A 9B 2F 34 F2 86 15 0B 6F 78 F6 2E 15
EC 30 9B BD B1 F3 93 AF 99 A9 C7 AD AA F5 83 DC
79 D3 CE 01 BF BA 29 33 33 9C 82 12 6E C4 FD 92
ED B9 46 81 D2 0B B0 E5 B0 C4 21 65 56 7C 26 17
.z.6..3ホ.1Fル.ユ1W
.°q../4...ox...
.0.7ア..ツ.ウヌユ..7
yモホ.ソコ)33...nt..
.ガF.メ.-.ート!eV|&
-(3)Receive encode_data from server(1)
41 B4 71 92 34 65 8F EF 9B 3B FC 0F 33 4D 5E 7D
72 DC 50 59 7D DC 7A 60 71 F1 1B 9E 65 40 57 7C
D1 81 71 85 8B A0 3C 8B 8B 9F 01 F7 5B 52 C4 8A
13 7A 0E 36 F3 10 33 CE 7F 49 46 D9 04 AD B2 57
FF DF 71 1A 9B 2F 34 F2 86 15 0B 6F 78 F6 2E 15
EC 30 9B BD B1 F3 93 AF 99 A9 C7 AD AA F5 83 DC
79 D3 CE 01 BF BA 29 33 33 9C 82 12 6E C4 FD 92
ED B9 46 81 D2 0B B0 E5 B0 C4 21 65 56 7C 26 17
h.q...<.....[Rト.
.z.6..3ホ.1Fル.ユ1W
.°q../4...ox...
.0.7ア..ツ.ウヌユ..7
yモホ.ソコ)33...nt..
.ガF.メ.-.ート!eV|&
-(4)Create decode_data
31 32 33 34 35 36 37 38 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
tcp>
  
```

Annotations in the image:

- PC(server)への接続処理**: Points to the connection setup commands.
- ①コマンド入力した平文のダンプ表示**: Points to the hex dump of the plain text "12345678".
- ②平文を暗号化 (Encrypt) した文章を PC (サーバー) に送信した文章のダンプ表示**: Points to the hex dump of the encrypted data.
- ③PC (サーバー) から暗号文を受信したダンプ表示**: Points to the hex dump of the received encrypted data.
- ④受信した暗号文を復号 (Decrypt) した文章のダンプ表示**: Points to the hex dump of the decrypted data.

Additional text in the annotation box:

①と④が同等であることを確認し、PC (サーバー) 側の②とも同等であることを確認する。

5) 「TCP_IP_RSA_TEST」側の送受信を確認する。

The screenshot shows the TCP_IP_RSA_Test application window. The Monitor section displays the following log output:

```

<<< Wait Connect >>>
接続しました。
<<< Wait Receive data[1] >>>
-(1)Receive encode data from client
43 48 75 9B 80 24 F2 B3 F8 E9 BF E2 0E EA BF 82
D1 E1 AB 8E BF CF 5C 82 AA 2B B1 AD A6 D3 B1 FB
4A 06 7D A0 6B 40 B2 6E CC AE BE BA A7 BC 14 52
42 14 0D 08 A6 BA 65 3D EF BD 06 B3 E1 E8 0E 6D
B6 97 09 07 48 F4 48 6A 82 A5 1C D8 8F 99 EB 33
26 CD 6E E4 C9 B8 68 26 DE 07 49 8E 5B D2 05 8C
E3 7F F2 E0 5E DE 5D 84 38 E3 62 13 78 0C F1 74
E2 4A 59 1E 66 0D 71 97 E2 FD D4 AC 63 77 1F E5
<<< Wait Receive data[2] >>>

```

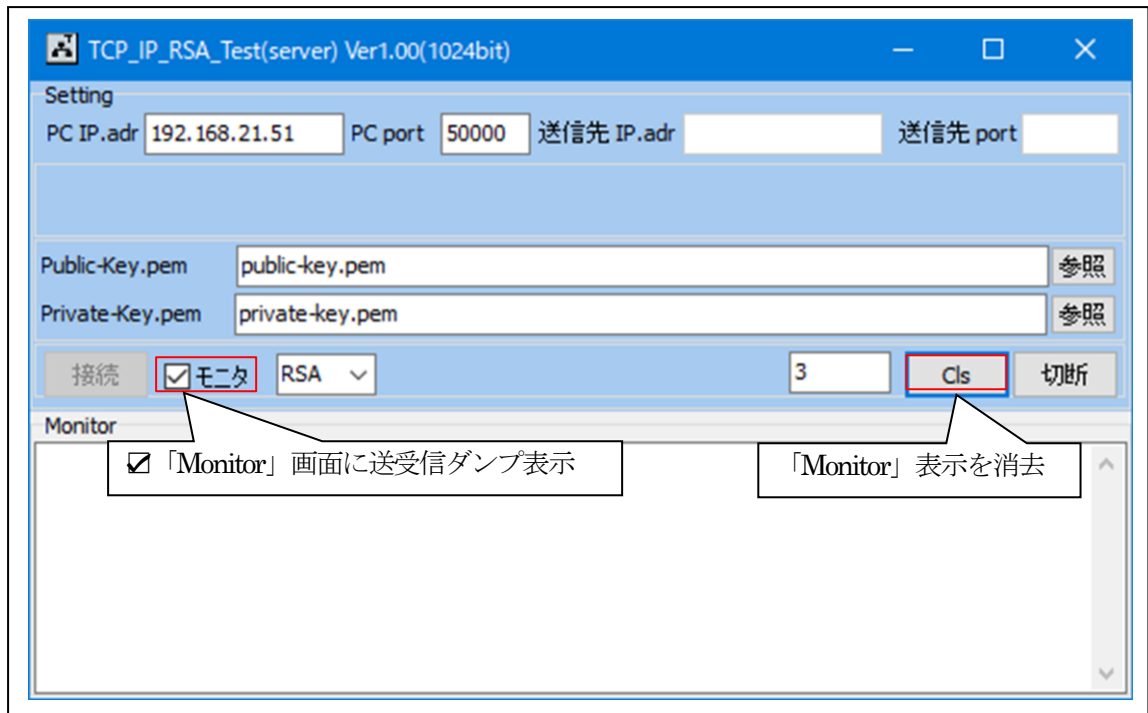
Three callout boxes provide explanations for the data flow:

- ①MP-S5D9 から受信した暗号文** (Received ciphertext from MP-S5D9): Points to the first block of hex data.
- ②受信した暗号文を復号(Decode)した文章のダンプ表示** (Dump display of the article decoded from the received ciphertext): Points to the second block of hex data.
- ③復号した文章を暗号(Encode)化してMP-S5D9 に送信した文章のダンプ表示** (Dump display of the article encoded into ciphertext and sent to MP-S5D9): Points to the third block of hex data.

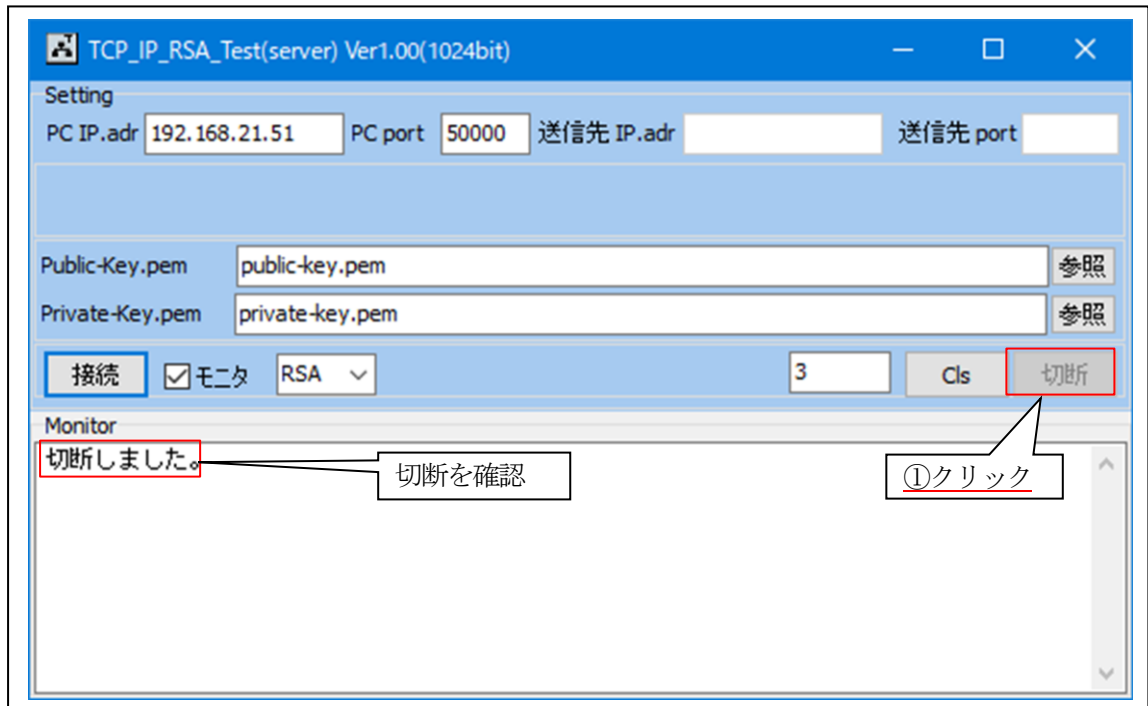
【Error 表示】

- ・ 受信時の接続失敗 「"error!! client_socket_connect()[error code]"
- ・ 受信異常 「"error!! recvfrom()[error code]"
- ・ 送信異常 「"error!! sendto()[error code]"

6) 「TCP_IP_RSA_TEST」 その他の操作



7) TCP_IP-Portを「切断」する。

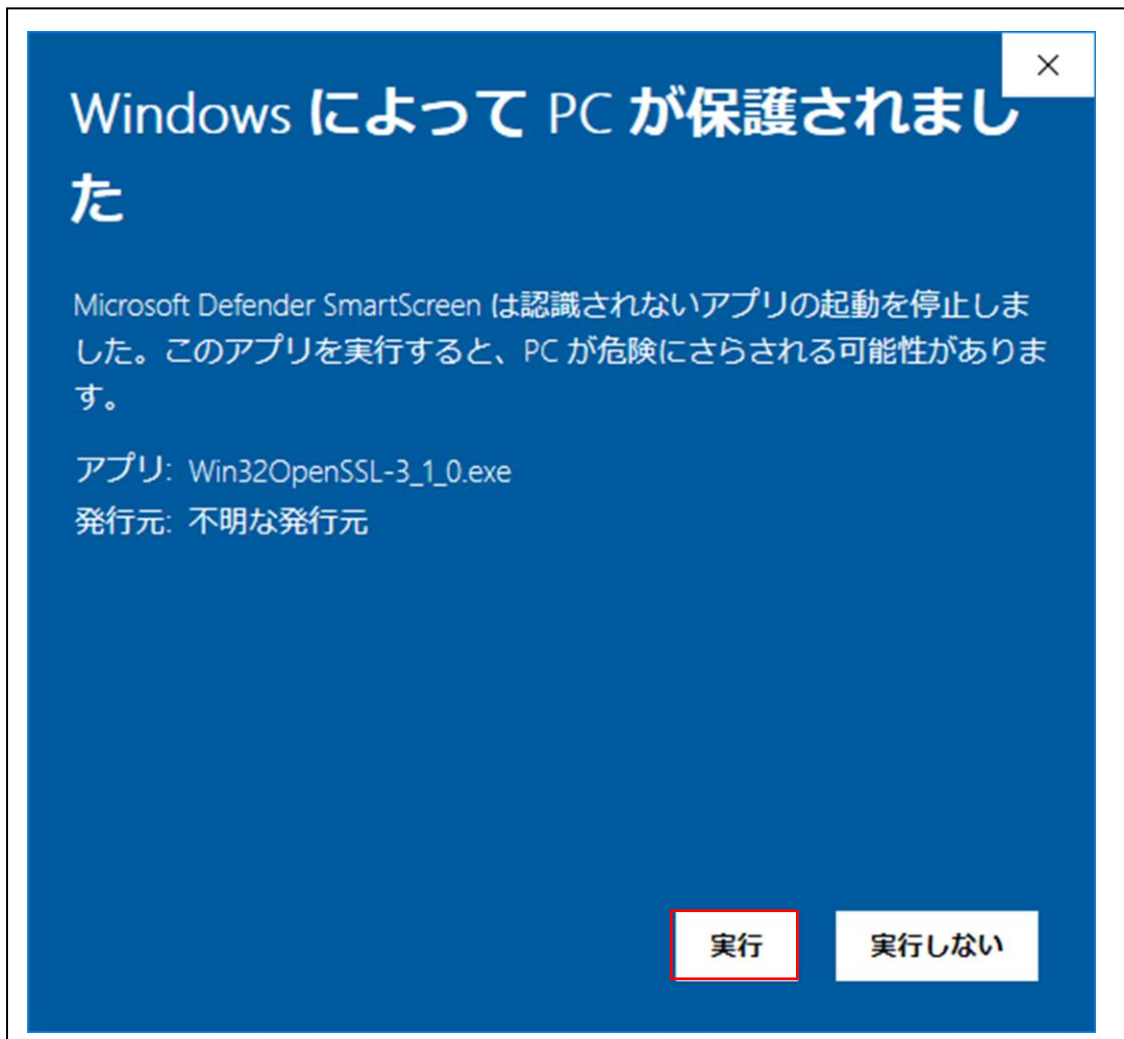
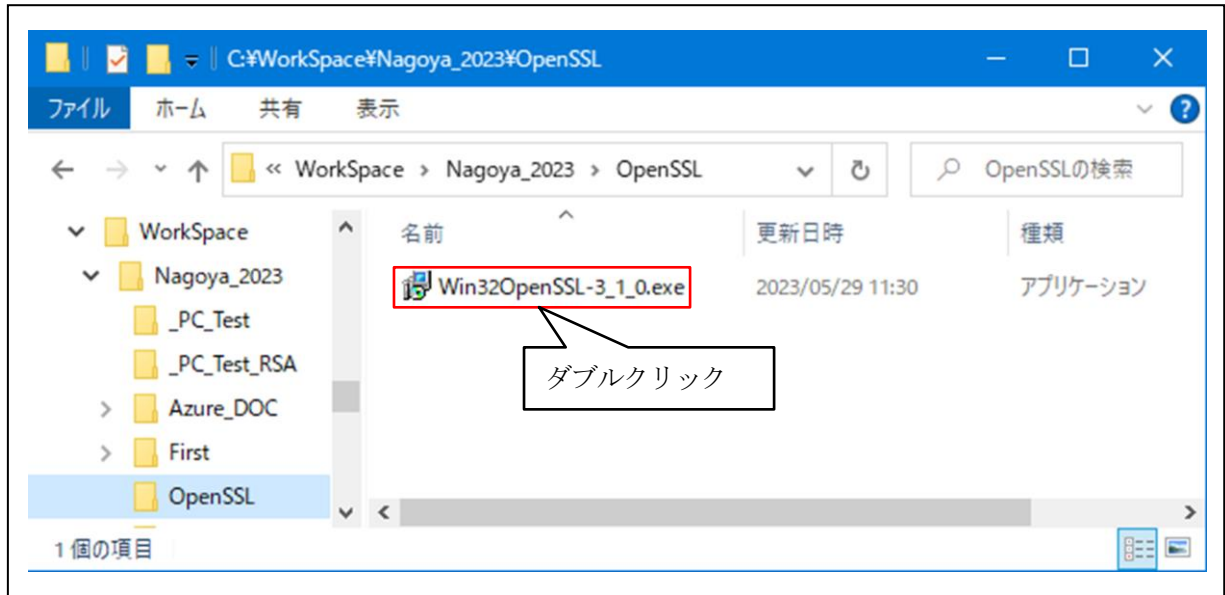


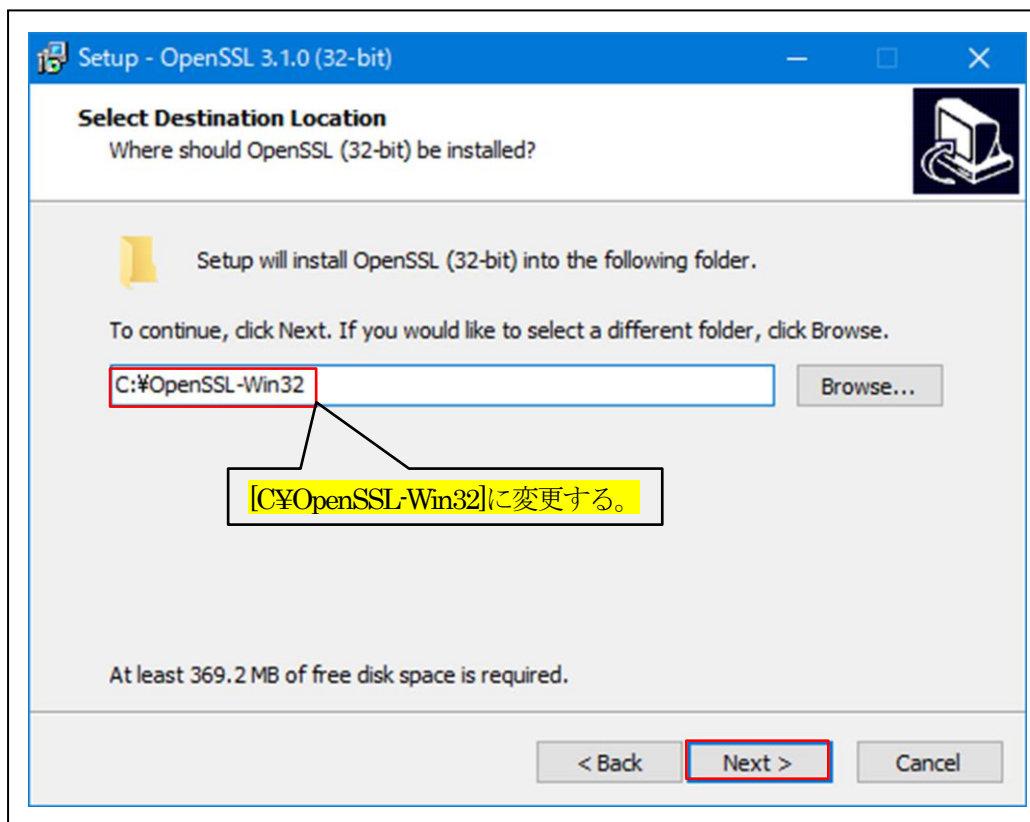
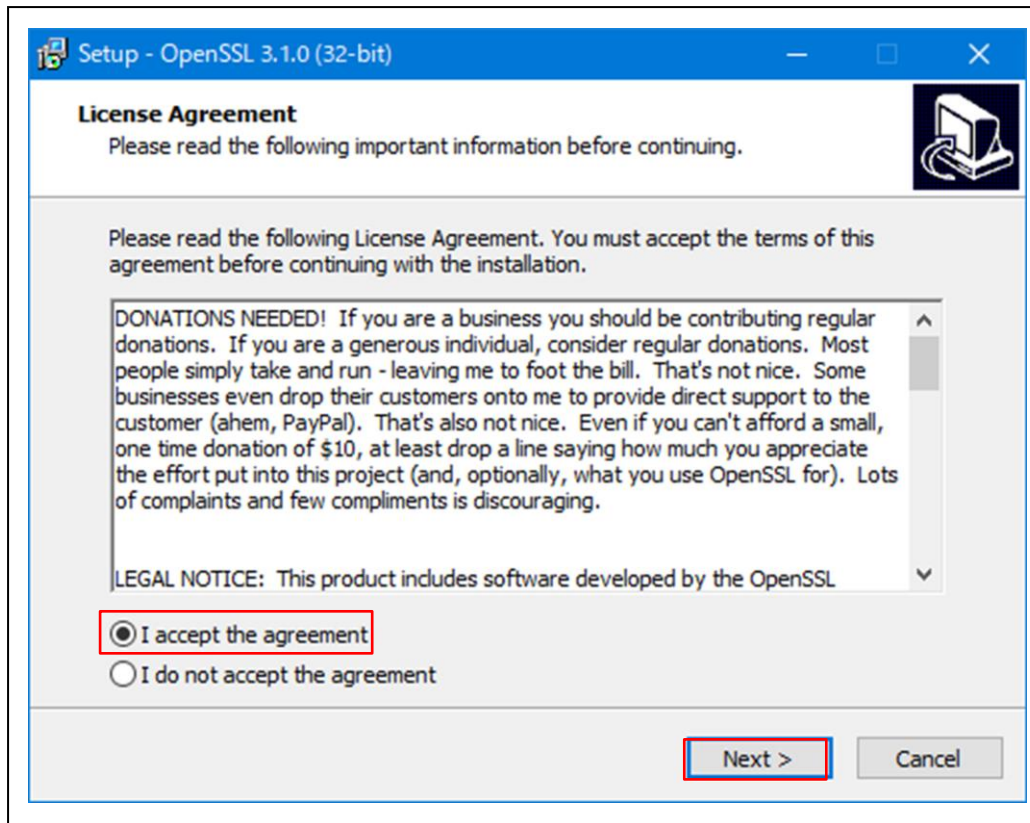
7-6. デバッグの終了

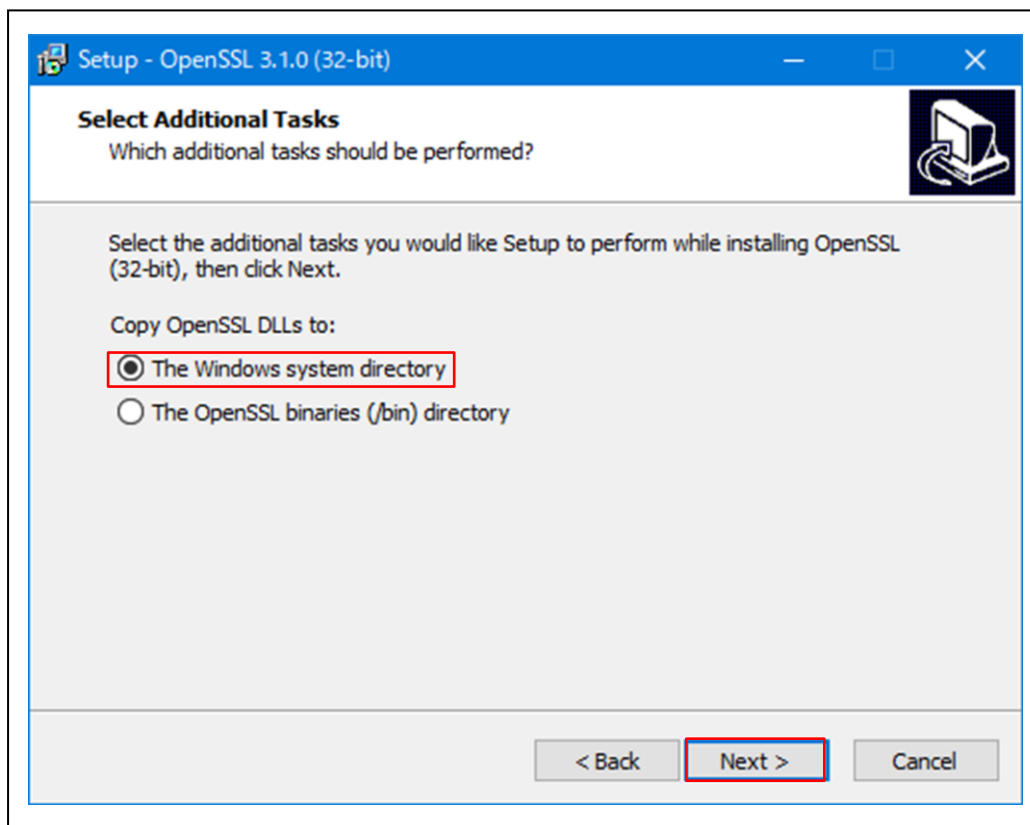
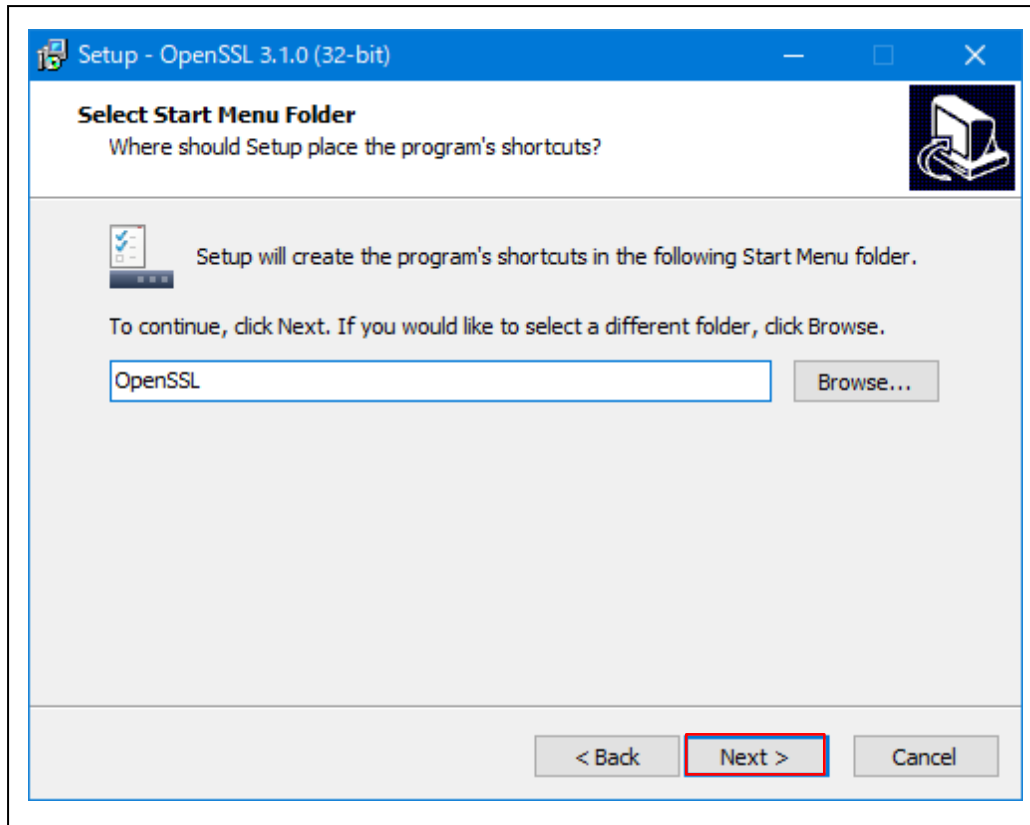
☆詳細操作は「[e2studio_synergy_Import.pdf](#)」の3-3項を参照して下さい。

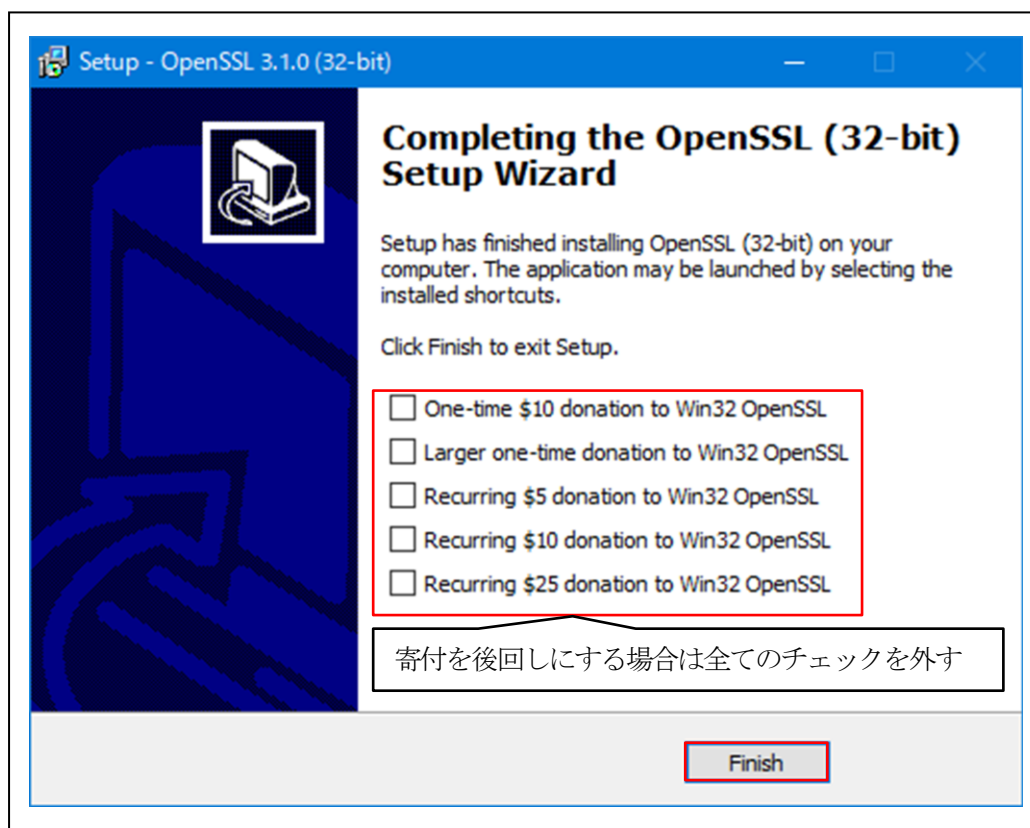
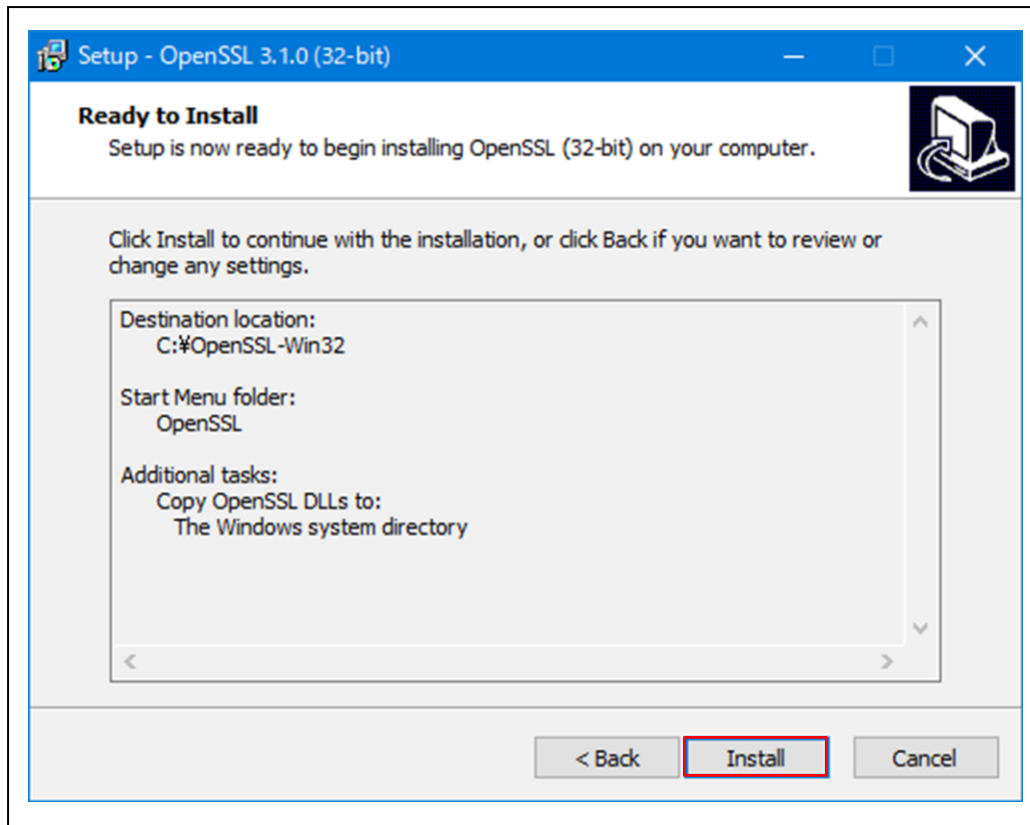
8. 公開鍵と秘密鍵を新しく作成する。

- 1) OpenSSL をインストールする。
 ☆既にインストール済みの場合は2) へ進む



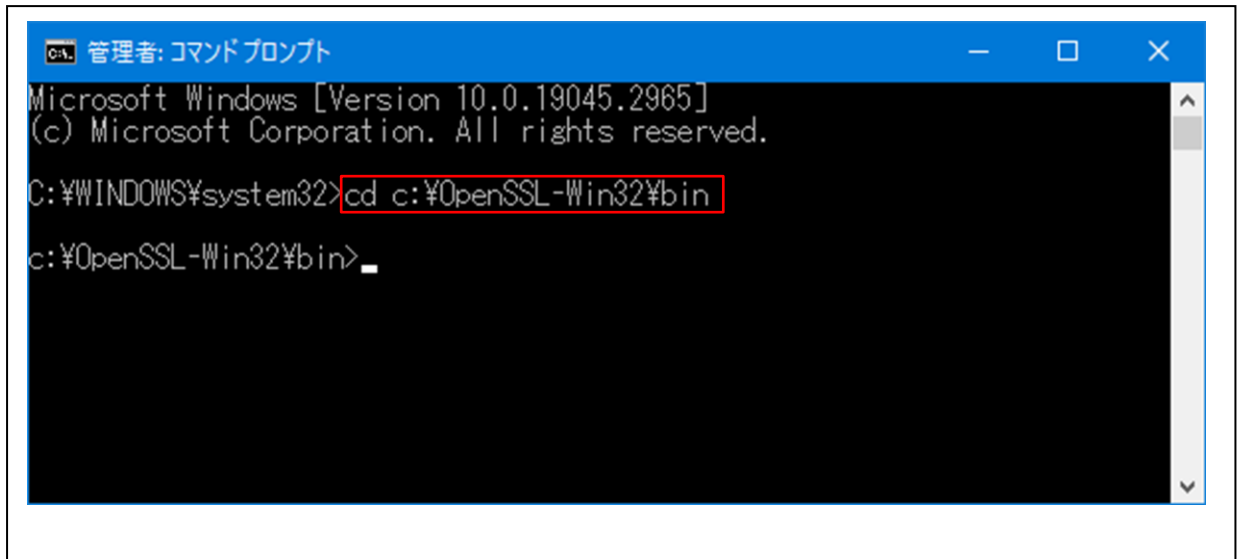






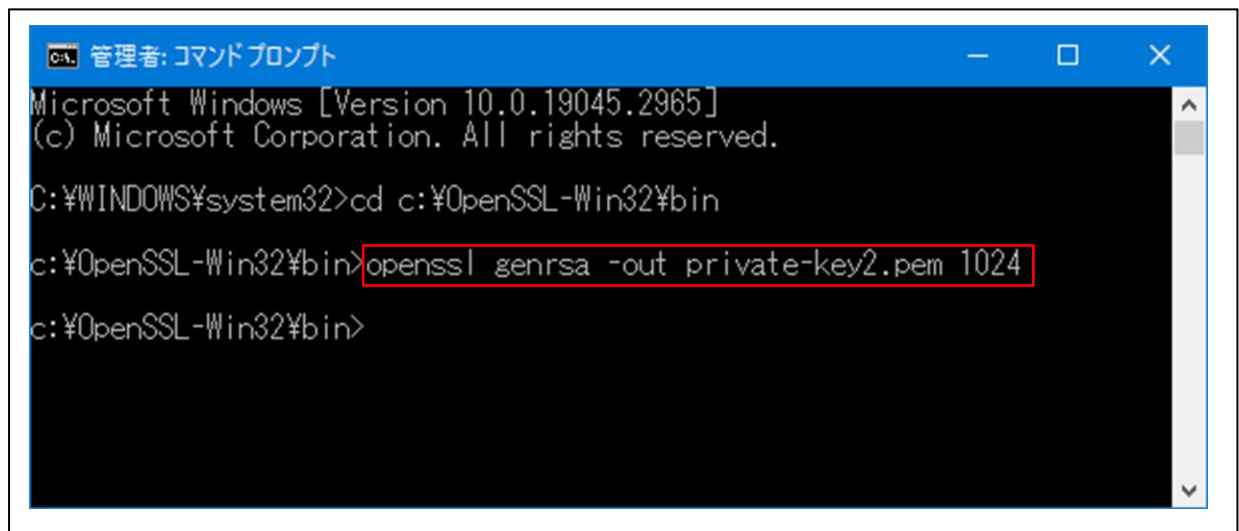
<OpenSSLのインストール終了>

- 2) OpenSSL を起動して秘密鍵と公開鍵を生成する。
 2-1) コマンドプロンプトを起動して CD コマンドにて「OpenSSL-Win32\bin」に移動する。

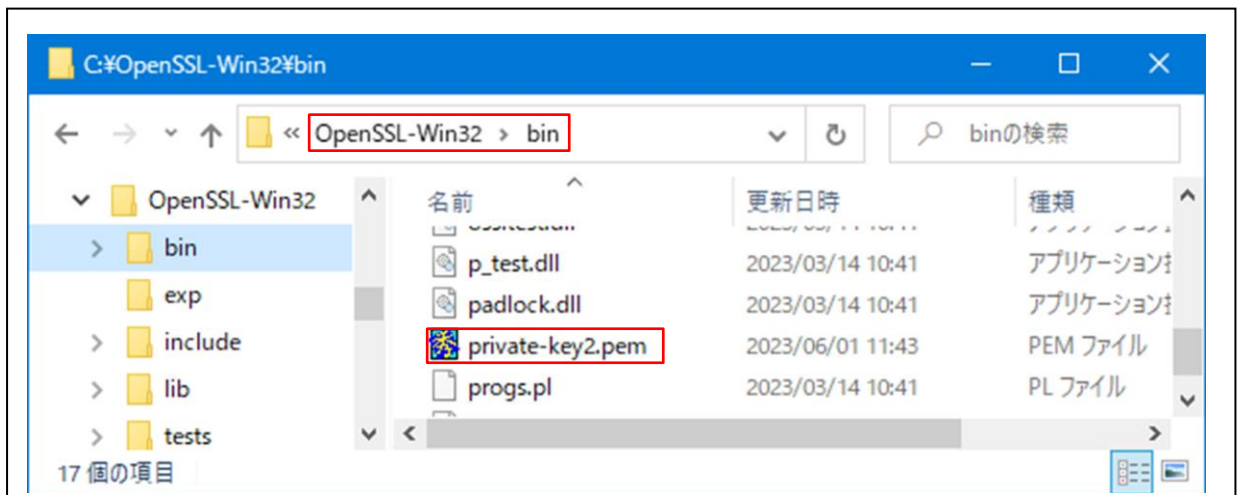


cd c:\OpenSSL-Win32\bin

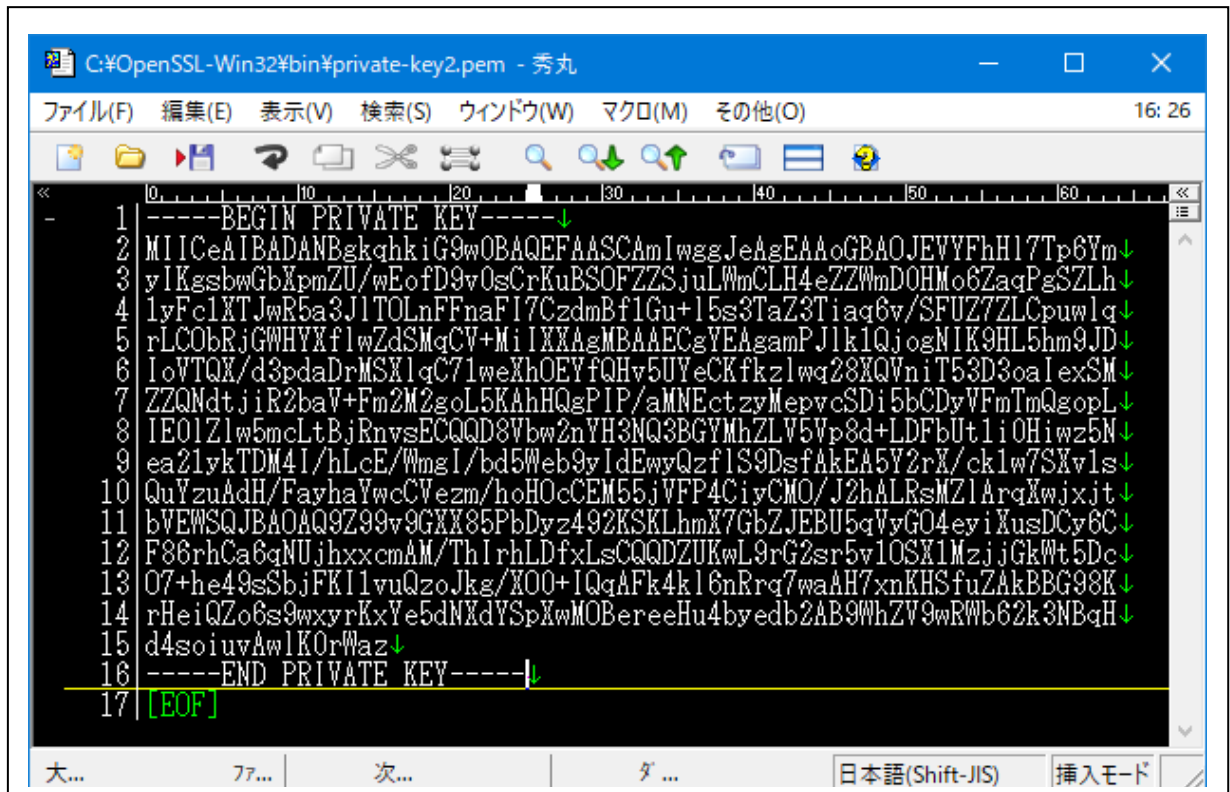
- 2-2) RSA 秘密鍵の生成(private-key2.pem)



openssl genrsa -out private-key2.pem 1024



2-3) 生成した秘密鍵「private-key2.pem」をメモ帳等のエディタで内容確認



The screenshot shows a Notepad window titled "C:\OpenSSL-Win32\bin\private-key2.pem - 秀丸". The window contains the following text:

```

1 -----BEGIN PRIVATE KEY-----
2 MIICeAIBADANBgkqhkiG9w0BAQEFAASCAmIwggJeaAgEAAoGBA0JEVYFhH17Tp6Ym
3 yIKgsbwGbXpmZU/wEofD9v0sCrKuBSOFZSjuLWmCLH4eZZWmDOHMo6ZaqPgSZLh
4 1yFc1XTJwR5a3JlTOLnFFnaF17CzdmBf1Gu+15s3TaZ3Tiaq6v/SFUZ7ZLCpuwlq
5 rLCObRjGWHYXflwZdSMqCV+Mi1XXAgMBAEECgYEAgamPJ1k1QjogNIK9HL5hm9JD
6 loVTQX/d3pdaDrMSX1qC71weXhOEYfQHv5UYeCKfkzlwq28XQVniT53D3oaIexSM
7 ZZQNdtjiR2baV+Fm2M2goL5KAhHQgPIP/aMNEctzyMepvcSDi5bCDyVFmTmQgopL
8 IE01Zlw5mcLtBjRnvsECQQD8Vbw2nYH3NQ3BGYmHZLV5Vp8d+LDFbUtli0Hiwz5N
9 ea2lykTDM4I/hLcE/WmgI/bd5Web9yIdEwyQzflS9DsfAkEA5Y2rX/cklw7SXvls
10 QuYzuAdH/FayhaYwcVezm/hOHOcCEM55jVFP4CiyCMO/J2hALRsMZlArqXwjxjt
11 bWEWSQJBAAQ9Z99v9GXX85PbDyz492KSKLhmX7GbZJEBU5qVyG04eyiXusDCy6C
12 F86rhCa6qNUjhxxcmAM/ThIrhLDfxLsCQQDZUKwL9rG2sr5v1OSX1MzjjGkwt5Dc
13 O7+he49sSbjFKI1vuQzoJkg/X00+IQqAFk4kl6nRrq7waAH7xnKHSfuZakBBG98K
14 rHeiQZ06s9wxyrKxYe5dNXdYSpXwMOBereehHu4byedb2AB9WhZV9wRWb62k3NBqH
15 d4soiuvAwlKOrWaz
16 -----END PRIVATE KEY-----
17 [EOF]
  
```

The status bar at the bottom of the window shows "大...", "77...", "次...", "ダ...", "日本語(Shift-JIS)", and "挿入モード".

2-4) RSA 公開鍵の生成(public-key2.pem)

```

管理: コマンド プロンプト
Microsoft Windows [Version 10.0.19045.2965]
(c) Microsoft Corporation. All rights reserved.

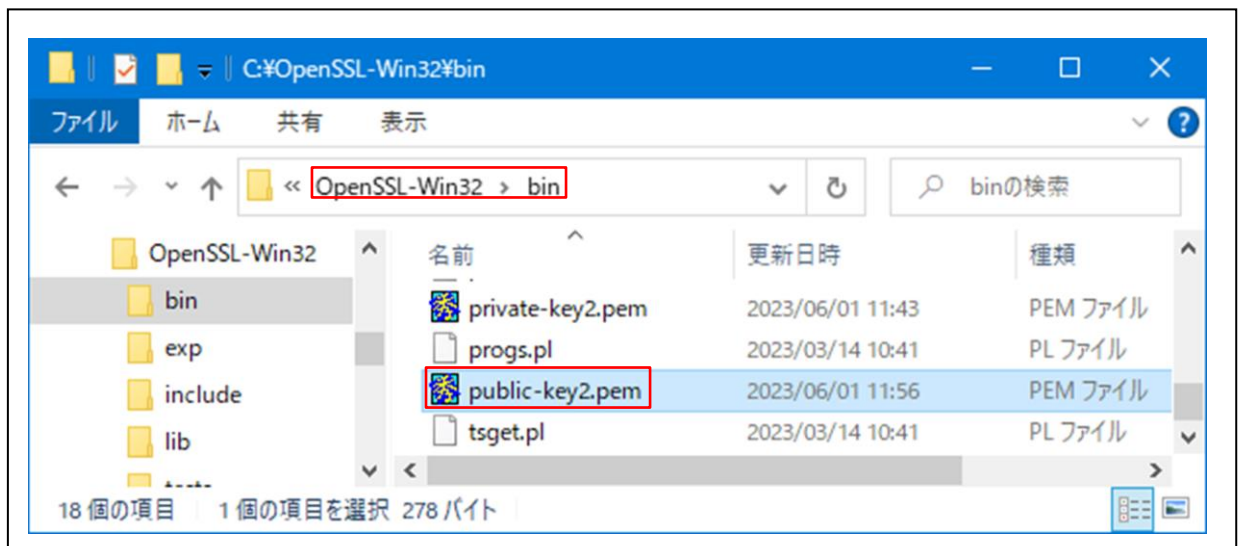
C:\WINDOWS\system32>cd c:\OpenSSL-Win32\bin

c:\OpenSSL-Win32\bin>openssl genrsa -out private-key2.pem 1024

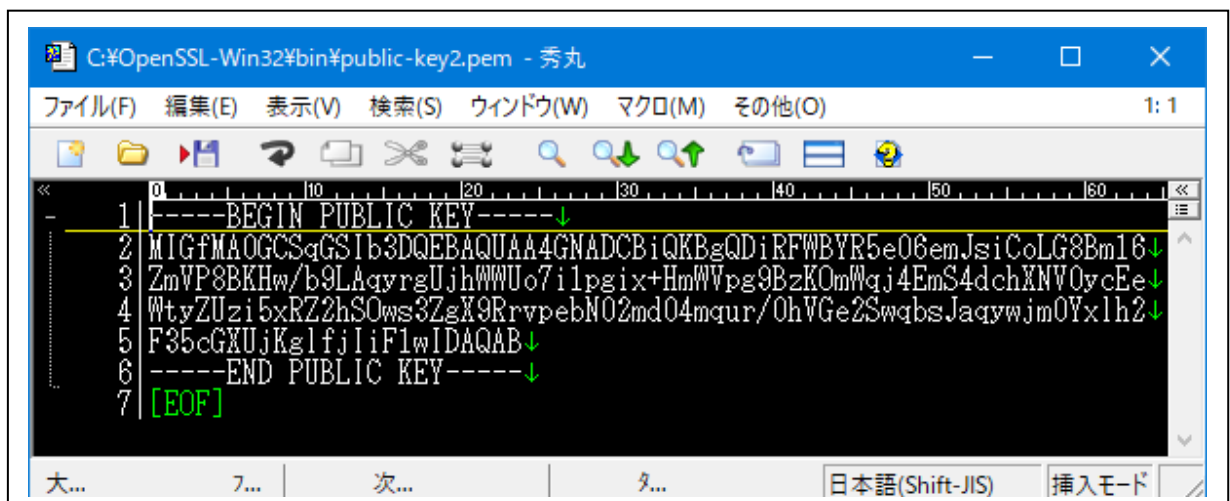
c:\OpenSSL-Win32\bin>openssl rsa -in private-key2.pem -pubout -out public-key2.pem
writing RSA key

c:\OpenSSL-Win32\bin>
    
```

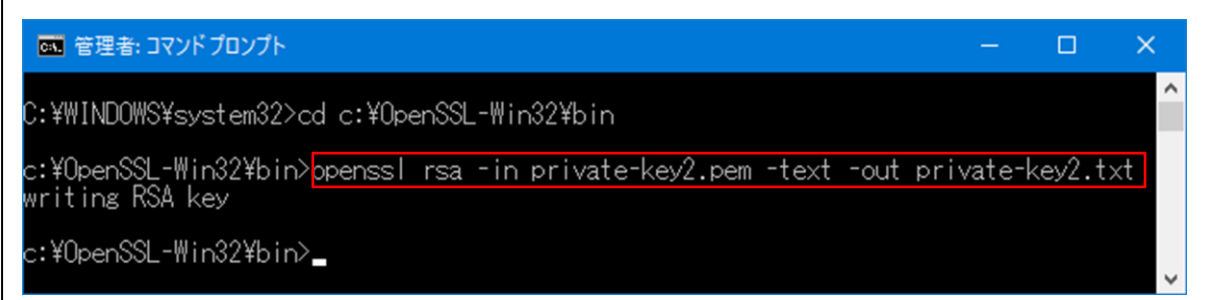
openssl rsa -in private-key2.pem -pubout -out public-key2.pem



2-5) 生成した秘密鍵「public-key2.pem」をメモ帳等のエディタで内容確認



2-6) RSA 秘密鍵(private-key2.pem)を DER 形式のテキストファイルを生成

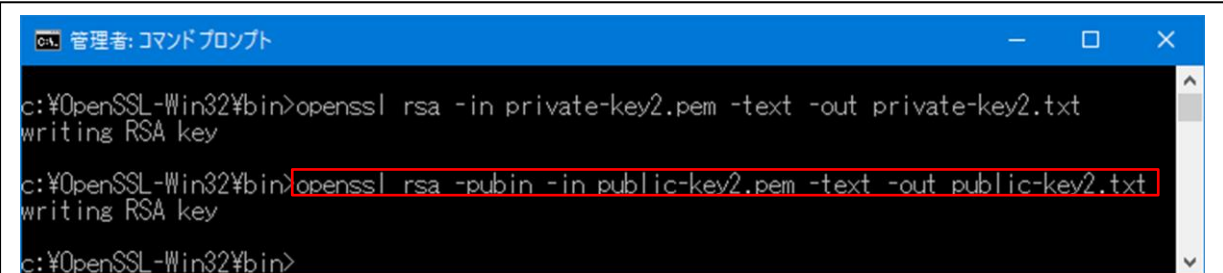


```

C:\WINDOWS\system32>cd c:\OpenSSL-Win32\bin
c:\OpenSSL-Win32\bin>openssl rsa -in private-key2.pem -text -out private-key2.txt
writing RSA key
c:\OpenSSL-Win32\bin>
  
```

openssl rsa -in private-key2.pem -text -out private-key2.txt

2-7) RSA 公開鍵(public-key2.pem)を DER 形式のテキストファイルを生成

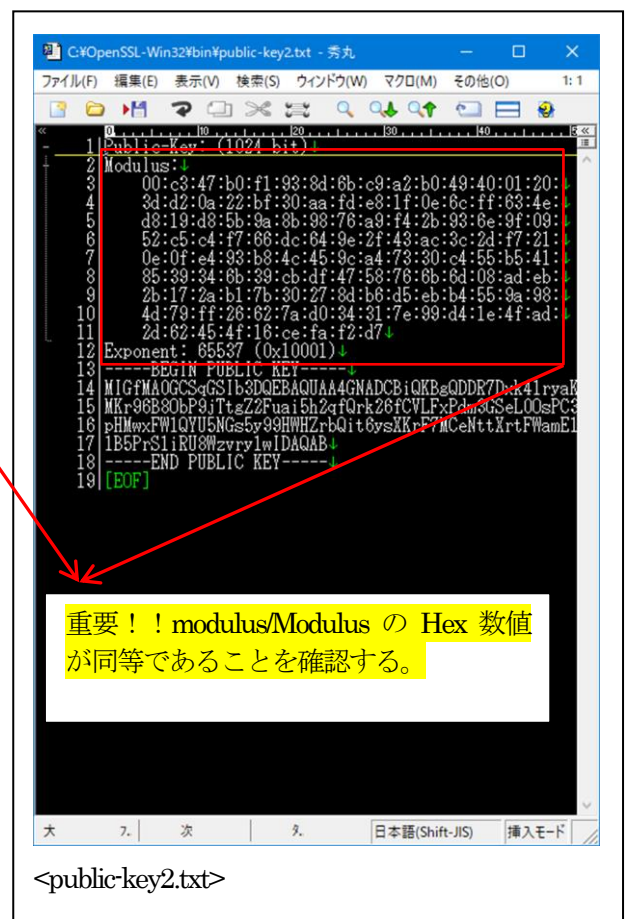
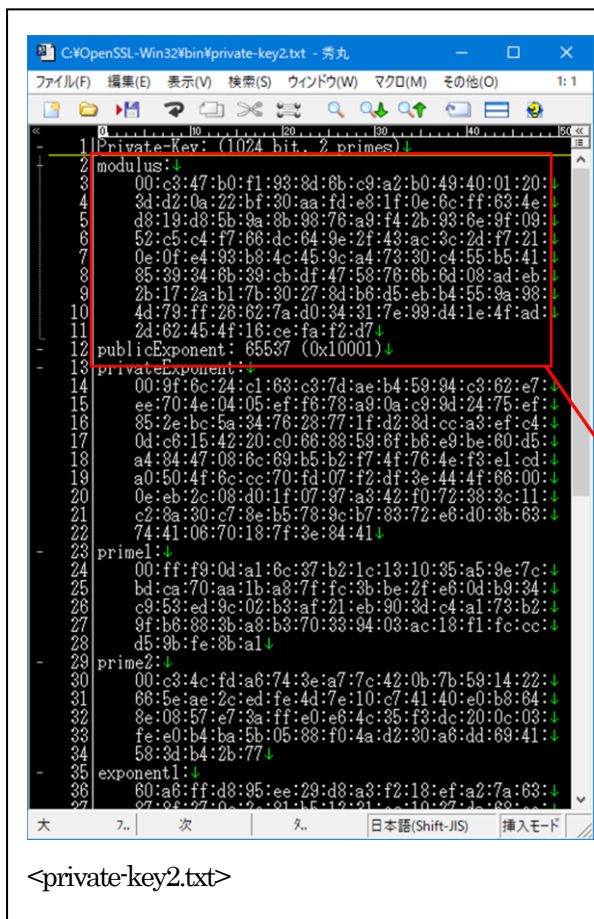
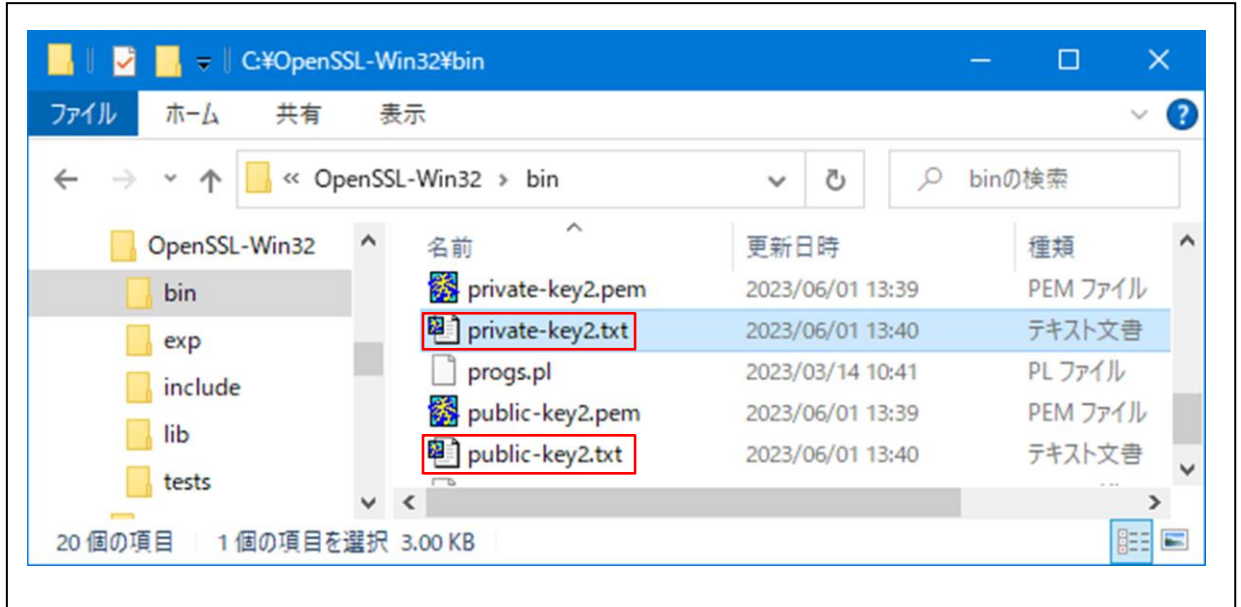


```

c:\OpenSSL-Win32\bin>openssl rsa -in private-key2.pem -text -out private-key2.txt
writing RSA key
c:\OpenSSL-Win32\bin>openssl rsa -pubin -in public-key2.pem -text -out public-key2.txt
writing RSA key
c:\OpenSSL-Win32\bin>
  
```

openssl rsa -pubin -in public-key2.pem -text -out public-key2.txt

2-8) 生成した DER 形式のテキストファイルをメモ帳等のエディタで内容確認



2-9 生成した DER 形式の秘密鍵(private-key2.txt)の説明

```

1 Private-Key: (1024 bit, 2 primes)
2 modulus:
3 00:c3:47:b0:f1:93:8d:6b:c9:a2:b0:49:40:01:20:
4 3d:d2:0a:22:bf:30:aa:fd:e8:1f:0e:6c:ff:63:4e:
5 d8:19:d8:5b:9a:8b:98:76:a9:f4:2b:93:6e:9f:09:
6 52:c5:c4:f7:66:dc:64:9e:2f:43:ac:3c:2d:f7:21:
7 0e:0f:e4:93:b8:4c:45:9c:a4:73:30:c4:55:b5:41:
8 85:39:34:6b:39:cb:df:47:58:76:6b:6d:08:ad:eb:
9 2b:17:2a:b1:7b:30:27:8d:b6:d5:eb:b4:55:9a:98:
10 4d:79:ff:26:62:7a:d0:34:31:7e:99:d4:1e:4f:ad:
11 2d:62:45:4f:16:ce:fa:f2:d7
12 publicExponent: 65537 (0x10001)
13 privateExponent:
14 00:9f:6c:24:c1:63:c3:7d:ae:b4:59:94:c3:62:e7:
15 ee:70:4e:04:05:ef:f6:78:a9:0a:c9:9d:24:75:ef:
16 85:2e:bc:5a:34:76:28:77:1f:d2:8d:cc:a3:ef:c4:
17 0d:c6:15:42:20:c0:66:88:59:6f:b6:e9:be:60:d5:
18 a4:84:47:08:6c:69:b5:b2:f7:4f:76:4e:f3:e1:cd:
19 a0:50:4f:6c:cc:70:fd:07:f2:df:3e:44:4f:66:00:
20 0e:eb:2c:08:d0:1f:07:97:a3:42:f0:72:38:3c:11:
21 c2:8a:30:c7:8e:b5:78:9c:b7:83:72:e6:d0:3b:63:
22 74:41:06:70:18:7f:3e:84:41
23 prime1:
24 00:ff:f9:0d:a1:6c:37:b2:1c:13:10:35:a5:9e:7c:
25 bd:ca:70:aa:1b:a8:7f:fc:3b:be:2f:e6:0d:b9:34:
26 c9:53:ed:9c:02:b3:af:21:eb:90:3d:c4:a1:73:b2:
27 9f:b6:88:3b:a8:b3:70:33:94:03:ac:18:f1:fc:cc:
28 d5:9b:fe:8b:a1
29 prime2:
30 00:c3:4c:fd:a6:74:3e:a7:7c:42:0b:7b:59:14:22:
31 66:5e:ae:2c:ed:fe:4d:7e:10:c7:41:40:e0:b8:64:
32 8e:08:57:e7:3a:ff:e0:e6:4c:35:f3:dc:20:0c:03:
33 fe:e0:b4:ba:5b:05:88:f0:4a:d2:30:a6:dd:69:41:
34 58:3d:b4:2b:77
35 exponent1:
36 60:a6:ff:d8:95:ee:29:d8:a3:f2:18:ef:a2:7a:63:
37 87:8f:27:0c:2e:81:b5:12:31:ec:10:27:da:68:ee:
38 24:3c:b2:0d:eb:1f:13:e5:c8:9f:2a:21:f4:77:dc:
39 0a:a6:42:30:9f:20:9c:b9:24:f1:d4:b1:7a:cd:35:
40 53:23:b8:41
41 exponent2:
42 00:9e:86:43:ed:74:b5:d6:a5:19:c6:d0:1c:82:27:
43 5a:08:c8:40:57:fd:ef:50:c0:b5:bb:d5:b9:92:72:
44 ac:87:41:32:c3:c5:26:fc:48:28:13:05:3e:4e:02:
45 80:44:4c:ee:67:2b:9a:f7:d1:03:c3:d1:ee:2e:9d:
46 87:ce:ae:0c:09
  
```

[modulus]
公開鍵として使用する。

[modulus]と
[privateExponent]
を組み合わせると秘密鍵
に使用する。

[prime1] 以下の数値
は、秘密鍵を生成する
ために使用した素数等
の数値です。無視して
も構いません。

3) 公開鍵よ秘密鍵を S5D9 側に登録するため DER 形式のデータを C 言語用に編集する。

<説明>

先頭の[00]は無視する

```

1 Private-Key: (1024 bit, 2 primes)
2 modulus:
3 00:c3:47:b0:f1:93:8d:6b:c9:a2:b0:49:40:01:20:
4 3d:d2:0a:22:bf:30:aa:fd:e8:1f:0e:6c:ff:63:4e:
5 d8:19:d8:5b:9a:8b:98:76:a9:f4:2b:93:6e:9f:09:
6 52:c5:c4:f7:66:dc:64:9e:2f:43:ac:3c:2d:f7:21:
7 0e:0f:e4:93:b8:4c:45:9c:a4:73:30:c4:55:b5:41:
8 85:39:34:6b:39:cb:df:47:58:76:6b:6d:08:ad:eb:
9 2b:17:2a:b1:7b:30:27:8d:b6:d5:eb:b4:55:9a:98:
10 4d:79:ff:26:62:7a:d0:34:31:7e:99:d4:1e:4f:ad:
11 2d:62:45:4f:16:ce:fa:f2:d7
12 publicExponent: 65537 (0x10001)
13 privateExponent:
14 00:3f:6c:24:c1:63:c3:7d:ae:b4:59:94:c3:62:e7:
15 ee:70:4e:04:05:ef:f6:78:a9:0a:c9:9d:24:75:ef:
16 85:2e:bc:5a:34:76:28:77:1f:d2:8d:cc:a3:ef:c4:
17 0d:c6:15:42:20:c0:66:88:59:6f:b6:e9:be:60:d5:
18 a4:84:47:08:6c:69:b5:b2:f7:4f:76:4e:f3:e1:cd:
19 a0:50:4f:6c:cc:70:fd:07:f2:df:3e:44:4f:66:00:
20 0e:eb:2c:08:d0:1f:07:97:a3:42:f0:72:38:3c:11:
21 c2:8a:30:c7:8e:b5:78:9c:b7:83:72:e6:d0:3b:63:
22 74:41:06:70:18:7f:3e:84:41
23 [EOF]
  
```

先頭の[00]は無視する

<modulus の編集>

e2studio でビルドが通るように C ベースの Hex 記述に編集する。

```

1 Private-Key: (1024 bit, 2 primes)
2 modulus:
3 // 0 1 2 3 4 5 6 7
4 0xc3,0x47,0xb0,0xf1,0x93,0x8d,0x6b,0xc9,
5 0xa2,0xb0,0x49,0x40,0x01,0x20,0x3d,0xd2,
6 0x0a,0x22,0xbf,0x30,0xaa,0xfd,0xe8,0x1f,
7 0x0e,0x6c,0xff,0x63,0x4e,0xd8,0x19,0xd8,
8 0x5b,0x9a,0x8b,0x98,0x76,0xa9,0xf4,0x2b,
9 0x93,0x6e,0x9f,0x09,0x52,0xc5,0xc4,0xf7,
10 0x66,0xdc,0x64,0x9e,0x2f,0x43,0xac,0x3c,
11 0x2d,0xf7,0x21,0x0e,0x0f,0xe4,0x93,0xb8,
12 0x4c,0x45,0x9c,0xa4,0x73,0x30,0xc4,0x55,
13 0xb5,0x41,0x85,0x39,0x34,0x6b,0x39,0xcb,
14 0xdf,0x47,0x58,0x76,0x6b,0x6d,0x08,0xad,
15 0xeb,0x2b,0x17,0x2a,0xb1,0x7b,0x30,0x27,
16 0x8d,0xb6,0xd5,0xeb,0xb4,0x55,0x9a,0x98,
17 0x4d,0x79,0xff,0x26,0x62,0x7a,0xd0,0x34,
18 0x31,0x7e,0x99,0xd4,0x1e,0x4f,0xad,0x2d,
19 0x62,0x45,0x4f,0x16,0xce,0xfa,0xf2,0xd7
20 publicExponent: 65537 (0x10001)
  
```

<privateExponent の編集>

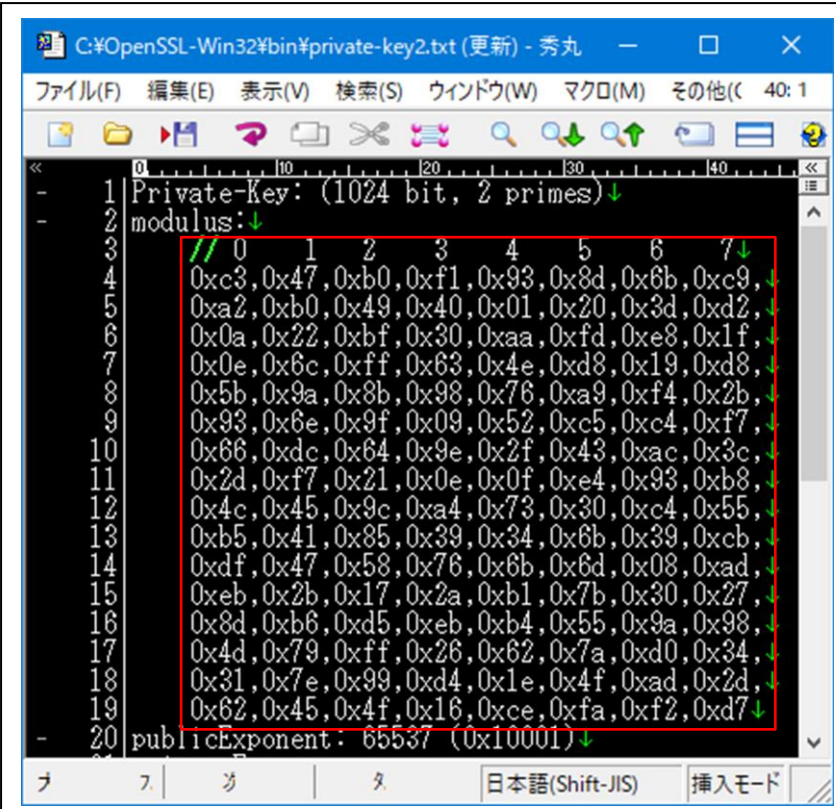
```

C:\OpenSSL-Win32\bin\private-key2.txt (更新) - 秀丸
ファイル(F) 編集(E) 表示(V) 検索(S) ウィンドウ(W) マクロ(M) その他(C) 40: 1
<< 0 10 20 30 40 >>
21 privateExponent:↓
22 // 0 1 2 3 4 5 6 7↓
23 0x9f,0x6c,0x24,0xc1,0x63,0xc3,0x7d,0xae,
24 0xb4,0x59,0x94,0xc3,0x62,0xe7,0xee,0x70,
25 0x4e,0x04,0x05,0xef,0xf6,0x78,0xa9,0x0a,
26 0xc9,0x9d,0x24,0x75,0xef,0x85,0x2e,0xbc,
27 0x5a,0x34,0x76,0x28,0x77,0x1f,0xd2,0x8d,
28 0xcc,0xa3,0xef,0xc4,0x0d,0xc6,0x15,0x42,
29 0x20,0xc0,0x66,0x88,0x59,0x6f,0xb6,0xe9,
30 0xbe,0x60,0xd5,0xa4,0x84,0x47,0x08,0x6c,
31 0x69,0xb5,0xb2,0xf7,0x4f,0x76,0x4e,0xf3,
32 0xe1,0xcd,0xa0,0x50,0x4f,0x6c,0xcc,0x70,
33 0xfd,0x07,0xf2,0xdf,0x3e,0x44,0x4f,0x66,
34 0x00,0x0e,0xeb,0x2c,0x08,0xd0,0x1f,0x07,
35 0x97,0xa3,0x42,0xf0,0x72,0x38,0x3c,0x11,
36 0xc2,0x8a,0x30,0xc7,0x8e,0xb5,0x78,0x9c,
37 0xb7,0x83,0x72,0xe6,0xd0,0x3b,0x63,0x74,
38 0x41,0x06,0x70,0x18,0x7f,0x3e,0x84,0x41↓
39 ↓
40 [EOF]

```

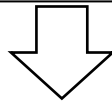
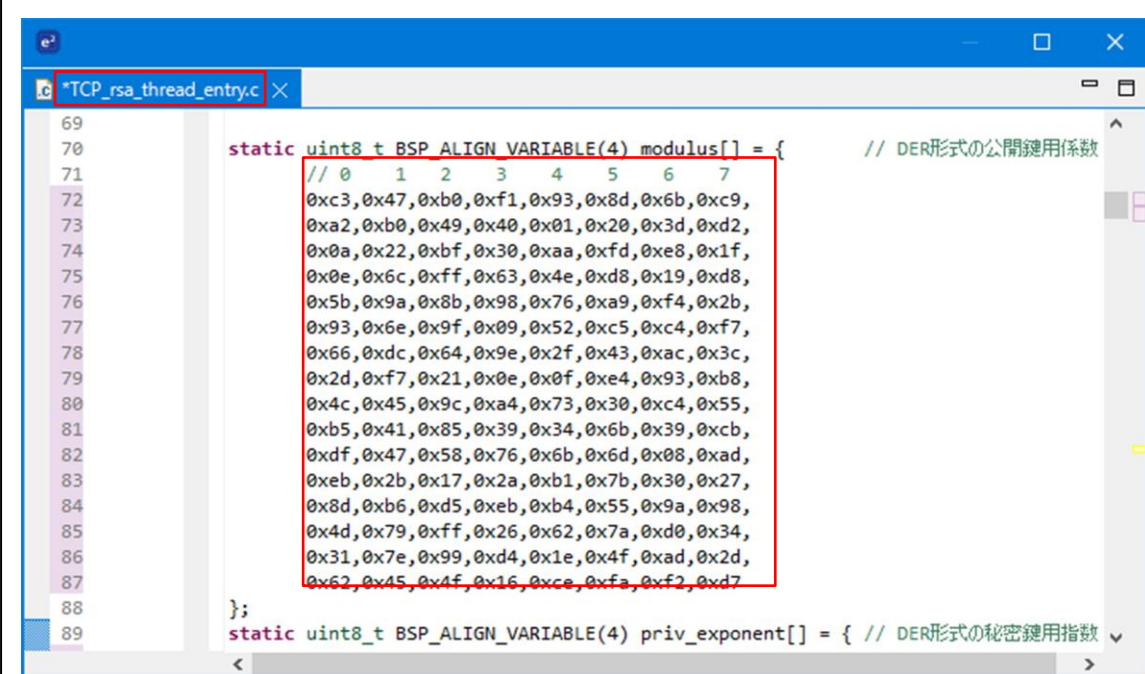
e2studio でビルドが通
るように C ベースの
Hex 記述に編集する。

4) 前項の3) で編集した DER 形式の「modulus」を置き換える



```

C:\OpenSSL-Win32\bin>private-key2.txt (更新) - 秀丸
ファイル(F) 編集(E) 表示(V) 検索(S) ウィンドウ(W) マクロ(M) その他(C) 40:1
1 Private-Key: (1024 bit, 2 primes)
2 modulus:
3 // 0 1 2 3 4 5 6 7
4 0xc3,0x47,0xb0,0xf1,0x93,0x8d,0x6b,0xc9,
5 0xa2,0xb0,0x49,0x40,0x01,0x20,0x3d,0xd2,
6 0x0a,0x22,0xbf,0x30,0xaa,0xfd,0xe8,0x1f,
7 0x0e,0x6c,0xff,0x63,0x4e,0xd8,0x19,0xd8,
8 0x5b,0x9a,0x8b,0x98,0x76,0xa9,0xf4,0x2b,
9 0x93,0x6e,0x9f,0x09,0x52,0xc5,0xc4,0xf7,
10 0x66,0xdc,0x64,0x9e,0x2f,0x43,0xac,0x3c,
11 0x2d,0xf7,0x21,0x0e,0x0f,0xe4,0x93,0xb8,
12 0x4c,0x45,0x9c,0xa4,0x73,0x30,0xc4,0x55,
13 0xb5,0x41,0x85,0x39,0x34,0x6b,0x39,0xcb,
14 0xdf,0x47,0x58,0x76,0x6b,0x6d,0x08,0xad,
15 0xeb,0x2b,0x17,0x2a,0xb1,0x7b,0x30,0x27,
16 0x8d,0xb6,0xd5,0xeb,0xb4,0x55,0x9a,0x98,
17 0x4d,0x79,0xff,0x26,0x62,0x7a,0xd0,0x34,
18 0x31,0x7e,0x99,0xd4,0x1e,0x4f,0xad,0x2d,
19 0x62,0x45,0x4f,0x16,0xce,0xfa,0xf2,0xd7
20 publicExponent: 65537 (0x10001)
  
```

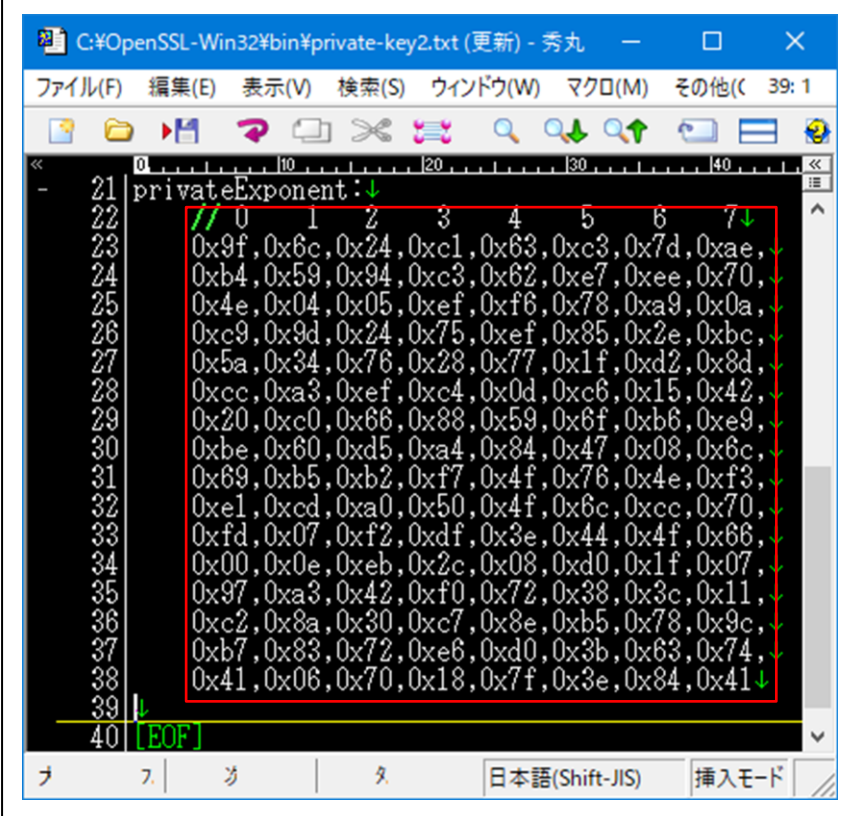



```

*TCP_rsa_thread_entry.c
69
70 static uint8_t BSP_ALIGN_VARIABLE(4) modulus[] = { // DER形式の公開鍵用係数
71 // 0 1 2 3 4 5 6 7
72 0xc3,0x47,0xb0,0xf1,0x93,0x8d,0x6b,0xc9,
73 0xa2,0xb0,0x49,0x40,0x01,0x20,0x3d,0xd2,
74 0x0a,0x22,0xbf,0x30,0xaa,0xfd,0xe8,0x1f,
75 0x0e,0x6c,0xff,0x63,0x4e,0xd8,0x19,0xd8,
76 0x5b,0x9a,0x8b,0x98,0x76,0xa9,0xf4,0x2b,
77 0x93,0x6e,0x9f,0x09,0x52,0xc5,0xc4,0xf7,
78 0x66,0xdc,0x64,0x9e,0x2f,0x43,0xac,0x3c,
79 0x2d,0xf7,0x21,0x0e,0x0f,0xe4,0x93,0xb8,
80 0x4c,0x45,0x9c,0xa4,0x73,0x30,0xc4,0x55,
81 0xb5,0x41,0x85,0x39,0x34,0x6b,0x39,0xcb,
82 0xdf,0x47,0x58,0x76,0x6b,0x6d,0x08,0xad,
83 0xeb,0x2b,0x17,0x2a,0xb1,0x7b,0x30,0x27,
84 0x8d,0xb6,0xd5,0xeb,0xb4,0x55,0x9a,0x98,
85 0x4d,0x79,0xff,0x26,0x62,0x7a,0xd0,0x34,
86 0x31,0x7e,0x99,0xd4,0x1e,0x4f,0xad,0x2d,
87 0x62,0x45,0x4f,0x16,0xce,0xfa,0xf2,0xd7
88 };
89 static uint8_t BSP_ALIGN_VARIABLE(4) priv_exponent[] = { // DER形式の秘密鍵用指数
  
```

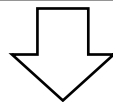
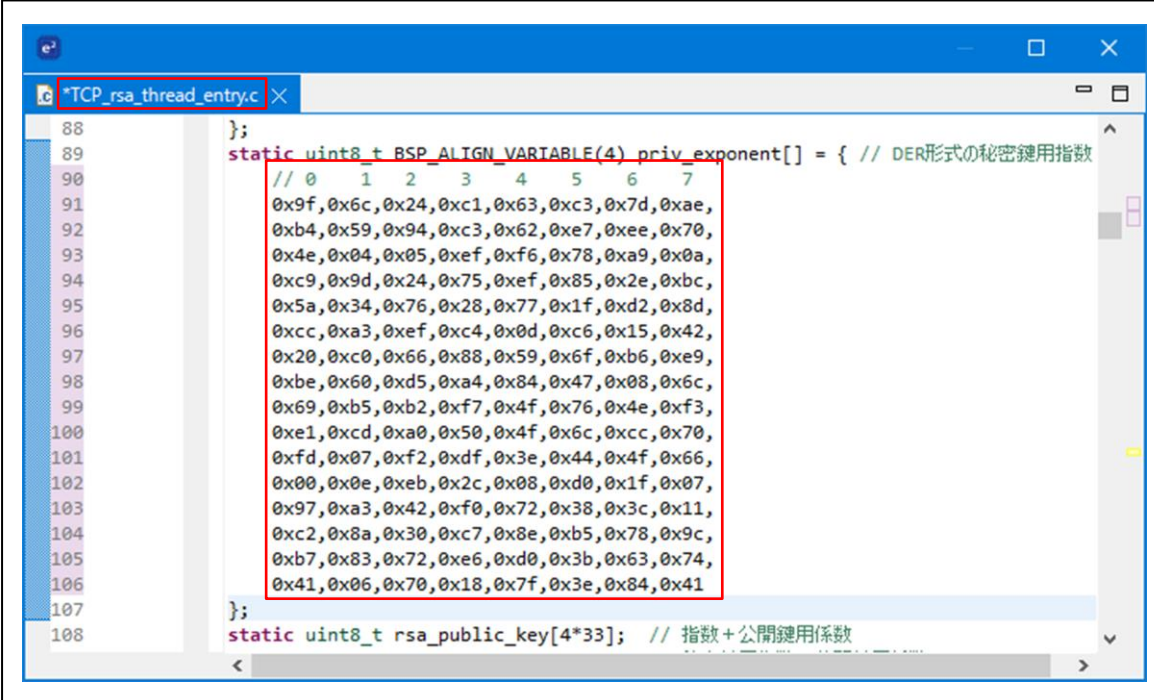
変数「`static uint8_t BSP_ALIGN_VARIABLE(4) modulus[]`」

5) 前項の3) で編集した DER 形式の「privateExponent」を置き換える



```

21 privateExponent:
22 // 0 1 2 3 4 5 6 7
23 0x9f,0x6c,0x24,0xc1,0x63,0xc3,0x7d,0xae,
24 0xb4,0x59,0x94,0xc3,0x62,0xe7,0xee,0x70,
25 0x4e,0x04,0x05,0xef,0xf6,0x78,0xa9,0x0a,
26 0xc9,0x9d,0x24,0x75,0xef,0x85,0x2e,0xbc,
27 0x5a,0x34,0x76,0x28,0x77,0x1f,0xd2,0x8d,
28 0xcc,0xa3,0xef,0xc4,0x0d,0xc6,0x15,0x42,
29 0x20,0xc0,0x66,0x88,0x59,0x6f,0xb6,0xe9,
30 0xbe,0x60,0xd5,0xa4,0x84,0x47,0x08,0x6c,
31 0x69,0xb5,0xb2,0xf7,0x4f,0x76,0x4e,0xf3,
32 0xe1,0xcd,0xa0,0x50,0x4f,0x6c,0xcc,0x70,
33 0xfd,0x07,0xf2,0xdf,0x3e,0x44,0x4f,0x66,
34 0x00,0x0e,0xeb,0x2c,0x08,0xd0,0x1f,0x07,
35 0x97,0xa3,0x42,0xf0,0x72,0x38,0x3c,0x11,
36 0xc2,0x8a,0x30,0xc7,0x8e,0xb5,0x78,0x9c,
37 0xb7,0x83,0x72,0xe6,0xd0,0x3b,0x63,0x74,
38 0x41,0x06,0x70,0x18,0x7f,0x3e,0x84,0x41
39
40 [EOF]
  
```

```

88 };
89 static uint8_t BSP_ALIGN_VARIABLE(4) priv_exponent[] = { // DER形式の秘密鍵用指数
90 // 0 1 2 3 4 5 6 7
91 0x9f,0x6c,0x24,0xc1,0x63,0xc3,0x7d,0xae,
92 0xb4,0x59,0x94,0xc3,0x62,0xe7,0xee,0x70,
93 0x4e,0x04,0x05,0xef,0xf6,0x78,0xa9,0x0a,
94 0xc9,0x9d,0x24,0x75,0xef,0x85,0x2e,0xbc,
95 0x5a,0x34,0x76,0x28,0x77,0x1f,0xd2,0x8d,
96 0xcc,0xa3,0xef,0xc4,0x0d,0xc6,0x15,0x42,
97 0x20,0xc0,0x66,0x88,0x59,0x6f,0xb6,0xe9,
98 0xbe,0x60,0xd5,0xa4,0x84,0x47,0x08,0x6c,
99 0x69,0xb5,0xb2,0xf7,0x4f,0x76,0x4e,0xf3,
100 0xe1,0xcd,0xa0,0x50,0x4f,0x6c,0xcc,0x70,
101 0xfd,0x07,0xf2,0xdf,0x3e,0x44,0x4f,0x66,
102 0x00,0x0e,0xeb,0x2c,0x08,0xd0,0x1f,0x07,
103 0x97,0xa3,0x42,0xf0,0x72,0x38,0x3c,0x11,
104 0xc2,0x8a,0x30,0xc7,0x8e,0xb5,0x78,0x9c,
105 0xb7,0x83,0x72,0xe6,0xd0,0x3b,0x63,0x74,
106 0x41,0x06,0x70,0x18,0x7f,0x3e,0x84,0x41
107 };
108 static uint8_t rsa_public_key[4*33]; // 指数 + 公開鍵用係数
  
```

変数「`static uint8_t BSP_ALIGN_VARIABLE(4) priv_exponent[]`」

6) クリーン・ビルドする。

9. 注意事項

- ・本文書の著作権は、エーワン（株）が保有します。
- ・本文書を無断での転載は一切禁止します。
- ・本文書に記載されている内容についての質問やサポートはお受けすることが出来ません。
- ・本文章に関して、ルネサス エレクトロニクス社への問い合わせは御遠慮願います。
- ・本文書の内容に従い、使用した結果、損害が発生しても、弊社では一切の責任を負わないものとします。
- ・本文書の内容に関して、万全を期して作成しましたが、ご不審な点、誤りなどの点がありましたら弊社までご連絡くだされば幸いです。
- ・本文書の内容は、予告なしに変更されることがあります。

10. 商標

- ・e2studio は、ルネサス エレクトロニクス株式会社の登録商標、または商品名称です。
- ・Renesas Synergy[™]および S3A7/S5D9/S7G2 は、ルネサス エレクトロニクス株式会社の登録商標、または商品名です。
- ・その他の会社名、製品名は、各社の登録商標または商標です。

10. 参考文献

- ・「S3A7 ユーザーズマニュアル ハードウェア編」 ルネサス エレクトロニクス株式会社
- ・「S5D9 ユーザーズマニュアル ハードウェア編」 ルネサス エレクトロニクス株式会社
- ・「S7G2 ユーザーズマニュアル ハードウェア編」 ルネサス エレクトロニクス株式会社
- ・ルネサス エレクトロニクス株式会社提供のサンプル集
- ・「e2studio ユーザーズマニュアル 入門ガイド」 ルネサス エレクトロニクス株式会社
- ・「SSP vx.x.x User's Manual」 ルネサス エレクトロニクス株式会社
- ・「X-Ware Component Documents for Renesas Synergy[™]」 ルネサス エレクトロニクス株式会社
- ・その他

〒486-0852

愛知県春日井市下市場町 6-9-20

エーワン株式会社

<https://www.robin-w.com>

