Intrepid Control Systems, Inc.

アプリケーションノート

Vehicle-Spy スタンドアロン ロギング　キャプチャー ファンクションブロック 実践編

（株）日本イントリピッド・コントロール・システムズ

目次

1. 概要

本アプリケーションノートは、当社製品 neoVI RED/FIRE 上で Vehicle Spy3 (以下 VSpy3 と表記)の Capture Function Block を使用したスタンドアロンでのロギング方法について neoVI FIRE を使用して解説します。

本アプリケーションノートは当社ウェブサイトの“製品マニュアルや仕様書“の“アプリケーションノート：データロギング編“にあります”アプリケーションノート_Vehicle-Spy_データロギング編.pdf“を読まれて内容を理解された方を対象にしています。

Capture Function Block（キャプチャ・ファンクションブロック方法（*1）。

	PC ロギング	スタンドアロンロギング
生値保存	可能	同左
シグナル値保存	可能	同左
保存データの VSpy３上での再生	可能	同左
保存形式	csv / vsb / asc / caniff / mat / mdf	同左
特徴	・柔軟なトリガー設定 ・スクリプト・ファンクションブロックやグラフィカルパネルと連動した高度な設定	同左

表 1.1

注記）
(*1)：Vehicle Spy には今回取り上げるロギング方法以外に幾つかのロギング方法が存在します。

2. 試験環境

2.1. 本試験で使用したソフトウェア

PC OS: Windows10（Windows 8/8.1/7/XP いずれでも使用可能です）
Vehicle Spy3 Version: 3.7.1.83

2.2. 本試験でのシステムブロック

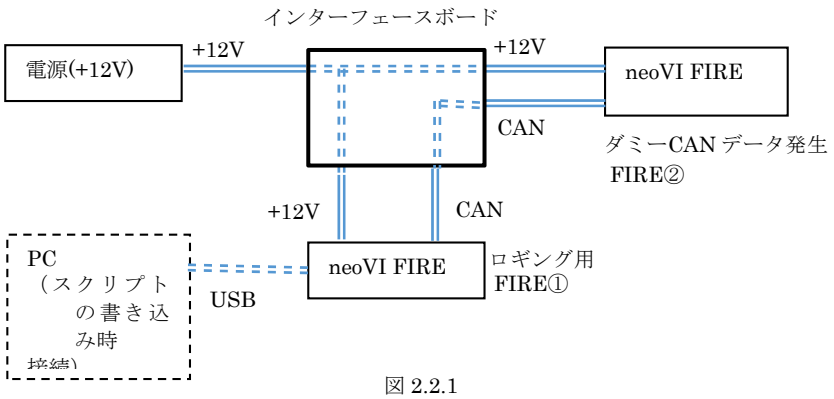


図 2.2.1

2.3.　試験写真(ロギング時)

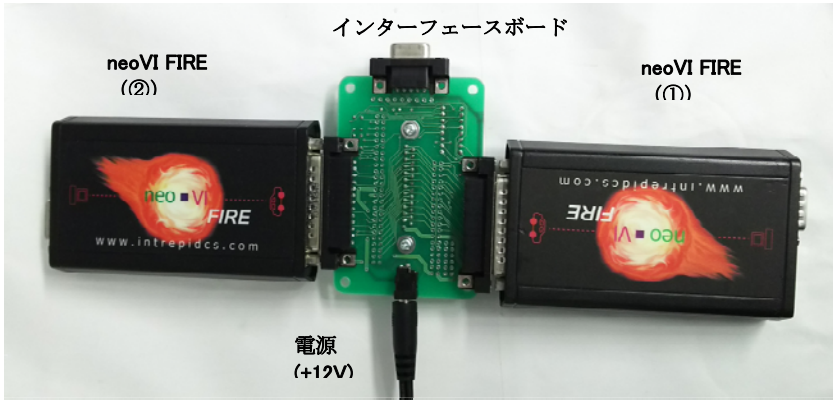


写真 2.3.1

3.　準備

本試験では OBD 等に接続する代わりにダミーデータを使用してロギングを行います。　ダミーデータは neoVI FIRE②にダミーデータ発生用のスクリプトを書き込みそのスクリプトを実行します。
実際には OBD 等のロギング対象物からのデータをロギングします。　対象物に接続できる環境をお持ちの場合はその対象物からのデータを使用しますので本項を飛ばして次の項に進んでも問題ありません。

3.1.　ダミーデータ発生用スクリプト

スクリプト名：　FIRE_DumyData_B01.vs3
スクリプト内容：
1)　ダミーデータ発生用 ID:　ID は 0x01, 0x0A, 0x32, 0x64 の 4 つ使用。
2)　発生周期：それぞれ以下の周期でデータを発生します。

ID	周期
0x01：	約 20ms
0x0A：	約 40ms
0x32：	約 80ms
0x64：	約 120ms

3)　シグナル：それぞれの ID に対して以下のバイトを使用してダミーシグナルを作成しています。それぞれのシグナル値は送信毎に値をインクリメントして送信します。(初期値 0)。

ID	定義バイト	シグナル名
0x01：	Byte 7,8	Counter_1_1
0x0A：	Byte 5,6	Counter_2_1
0x32：	Byte 3,4	Counter_3_1
0x64：	Byte 1,2	Counter_4_1

実行結果：

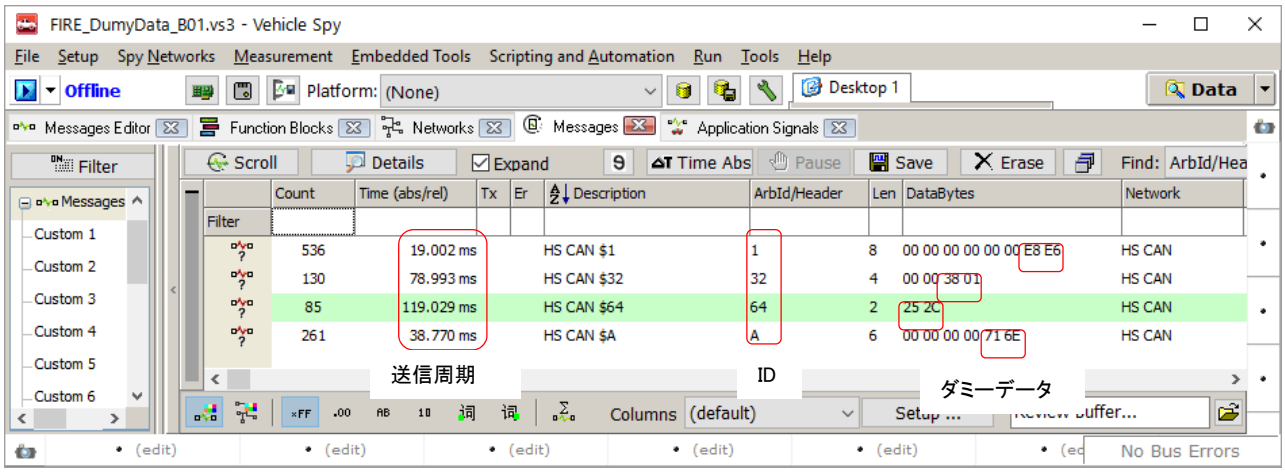


図 3.1.1

4. キャプチャ・ファンクションブロックによるロギング例

4.1. 電源ON後からCANバス上の全データを500メッセージ分1回だけロギング

4.1.1. スクリプト作成

1. 2.2 項に示す接続状態で行って下さい。実際にスタンドアロンロギングを開始するまでは FIRE②（OBD 等のロギング対象物）を接続する必要はございません。

2. Capture Function Block 設定画面へ移行。　以下の図中に示すように、“Function Blocks” -> “Capture” と進んで下さい。

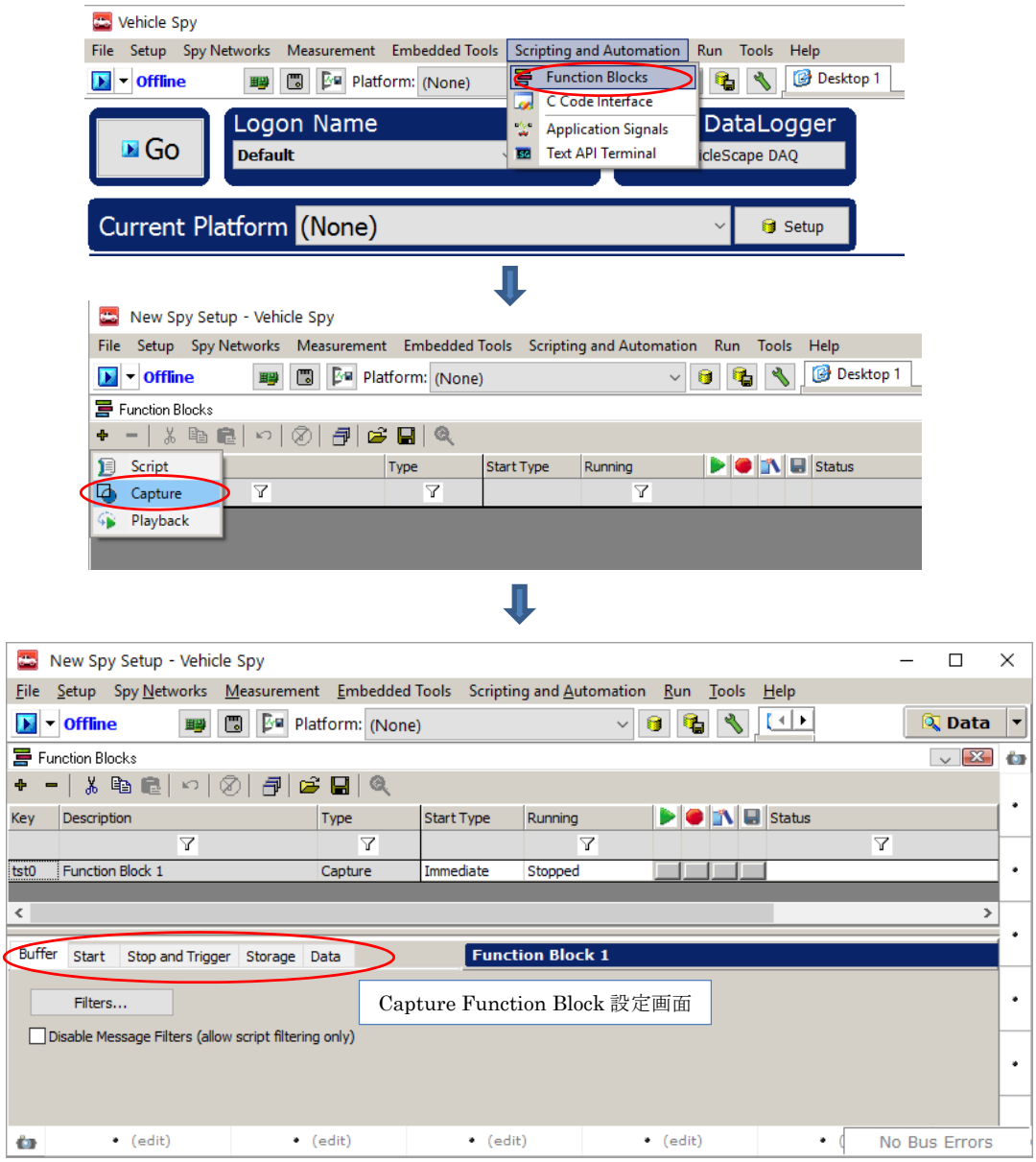


図 4.1.1.1

3.　設定内容。　どのようにロギングを行うかの設定を行います。

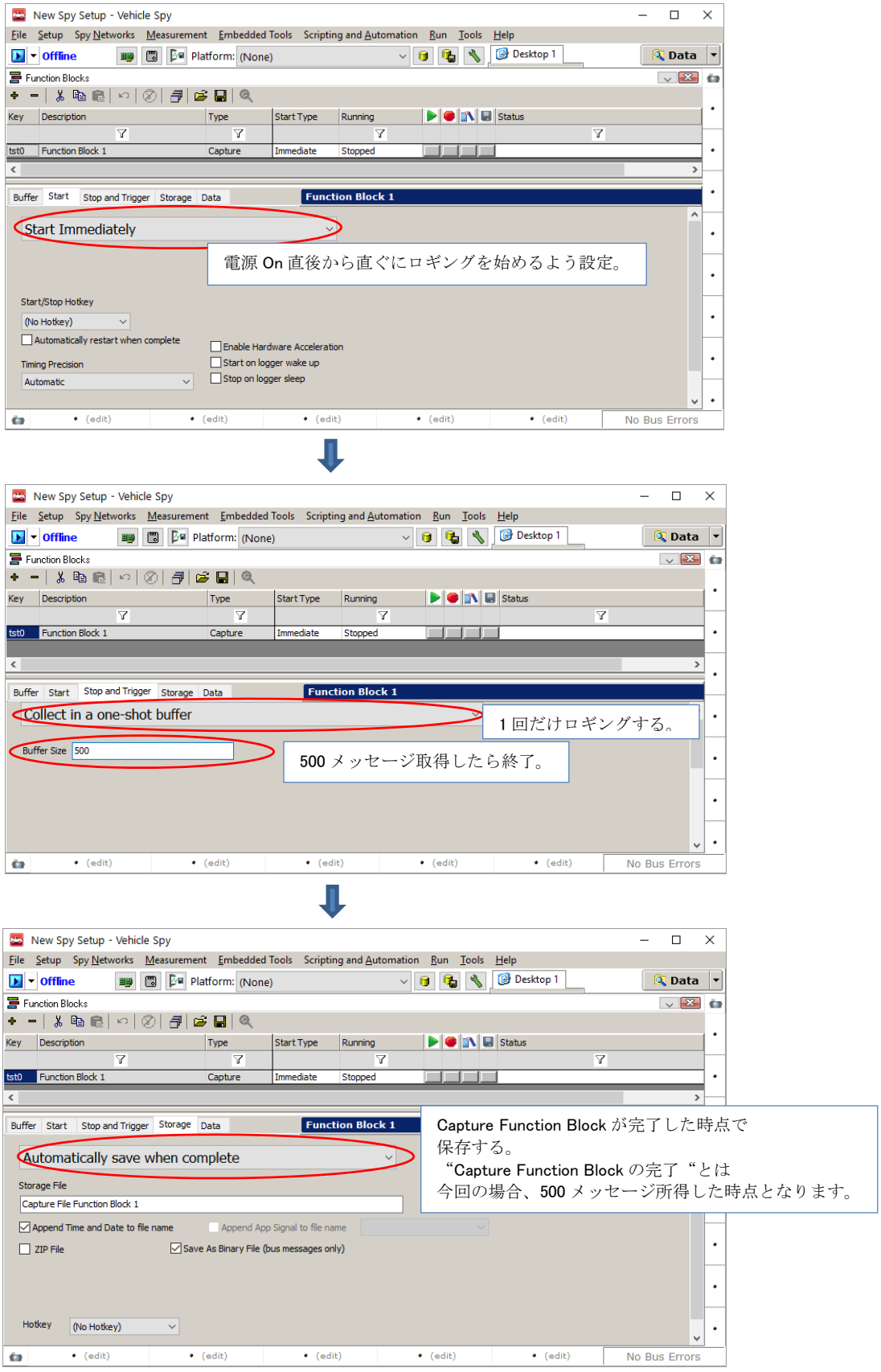


図 4.1.1.2

4.　作成スクリプトを保存。　作成したスクリプト（設定した内容）はファイル名を付けて保存します。（VS3 ファイルとして保存）。
メニューバーの“File”->”Save as”でファイル名指定。

4.1.2. スクリプトの書き込み

1. CoreMini で SD-Card へスクリプトの書き込みを行います。

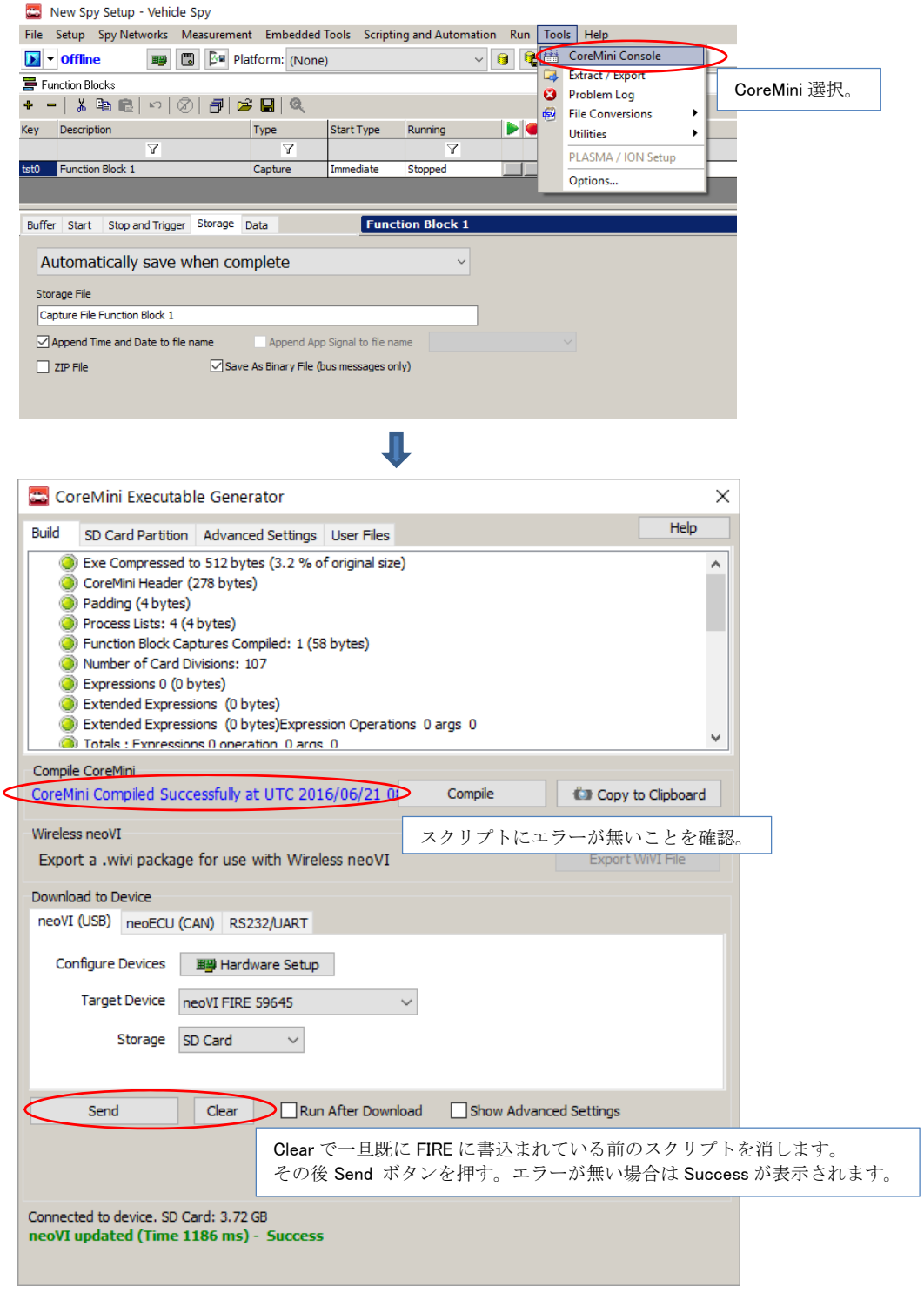


図 4.1.2.1

4.1.3.　ロギングスタート及び確認

1.　スタンドアロンロギング開始

上記 4.1.2 項の書き込み後、PC からのロギング用 FIRE①への USB ケーブルを外し FIRE①の電源を一旦落とします。ダミーデータ発生用 FIRE②（FIRE②が無い場合は OBD 等）とロギング用の FIRE①を接続します。　その後再度両 FIRE への電源を On します。OBD 等へ接続している場合は OBD 側の電源（車の電源）とロギング用 FIRE①の電源を On します。

2.　スタンドアロンロギング中

ロギング中は FIRE①の赤 LED が点滅しています。　問題がなければ約 1 分程度で 500 メッセージはロギングできます。

3.　スタンドアロンロギング終了

FIRE①の電源を Off にして、FIRE①から SD-Card を抜き取って PC へ挿入します。

写真 4.1.3.1

4.1.4.　SD－Cardからのデータのエクストラクト方法

1.　ロギングデータのエクストラクト

PC に SD-Card を挿入したら、以下の手順でロギングデータを SD-Card よりエクストラクトします。“Extract/Export”を選択して“Extract to VSPY binary[.vsb]”をクリックします。

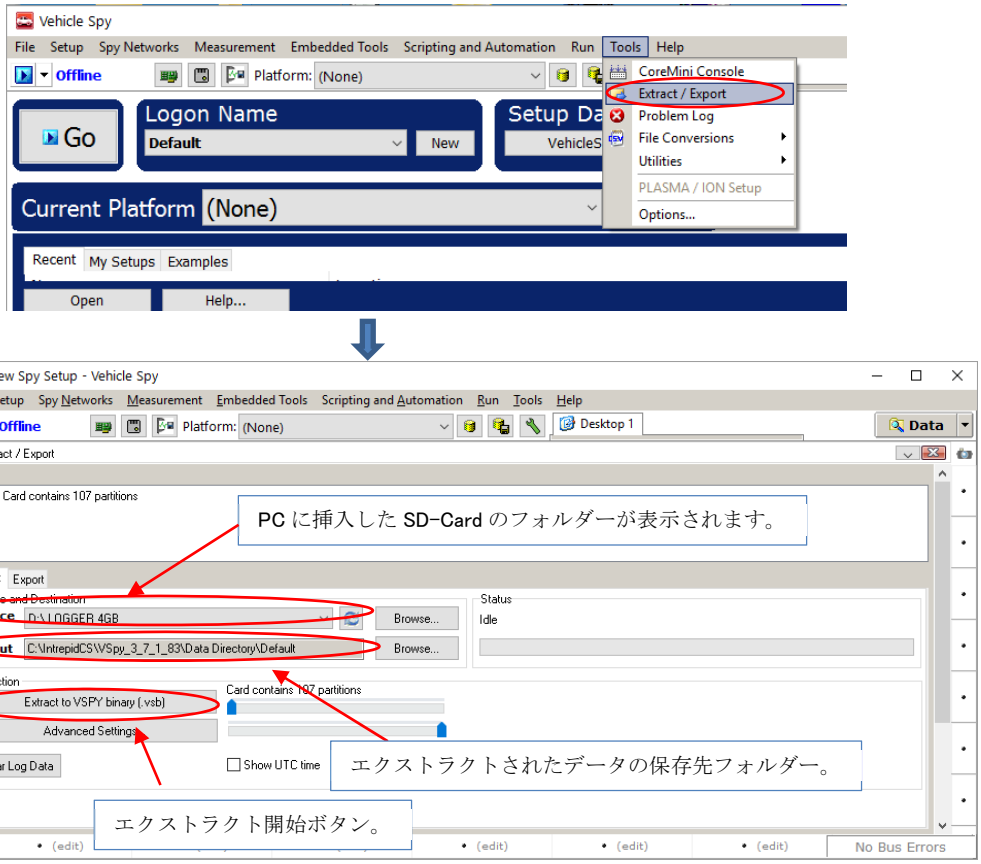
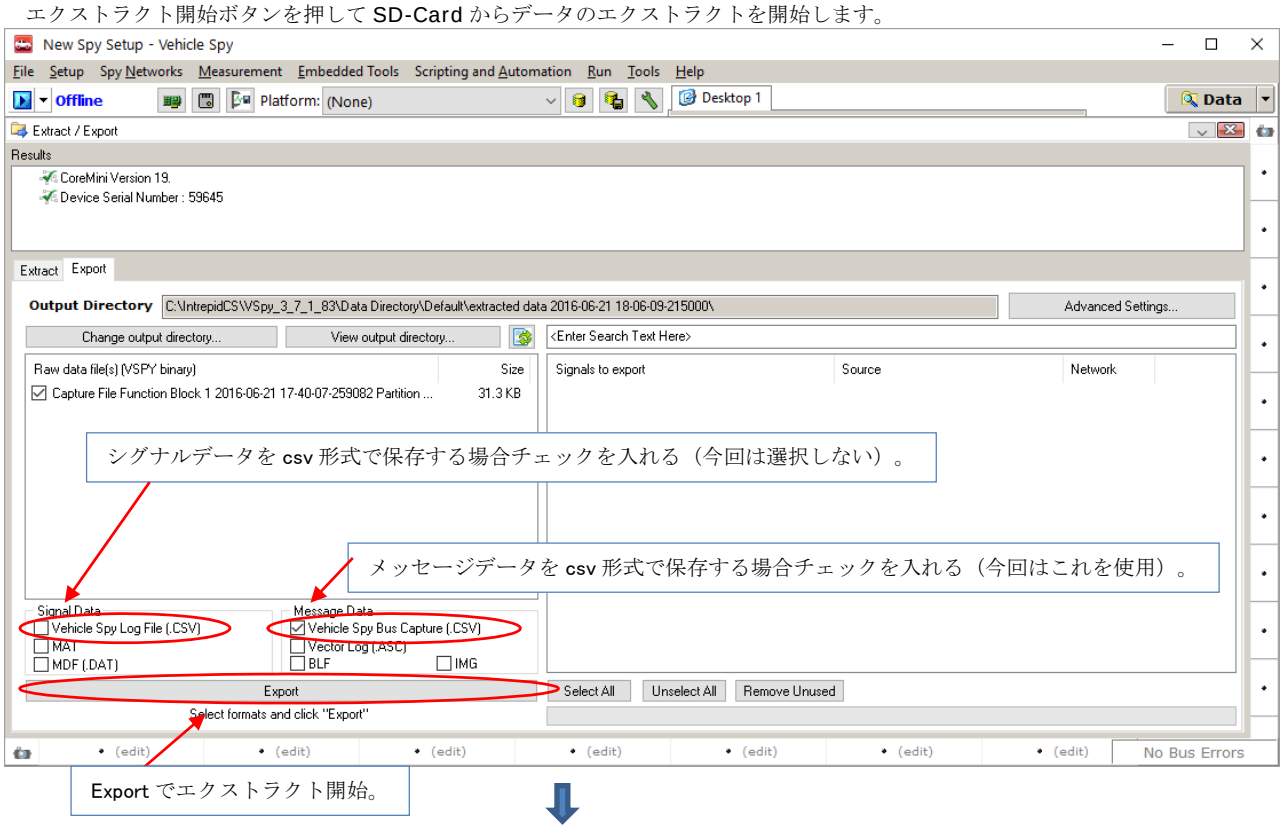


図 4.1.4.1

2.　データエクストラクト開始

エクストラクト開始ボタンを押して SD-Card からデータのエクストラクトを開始します。



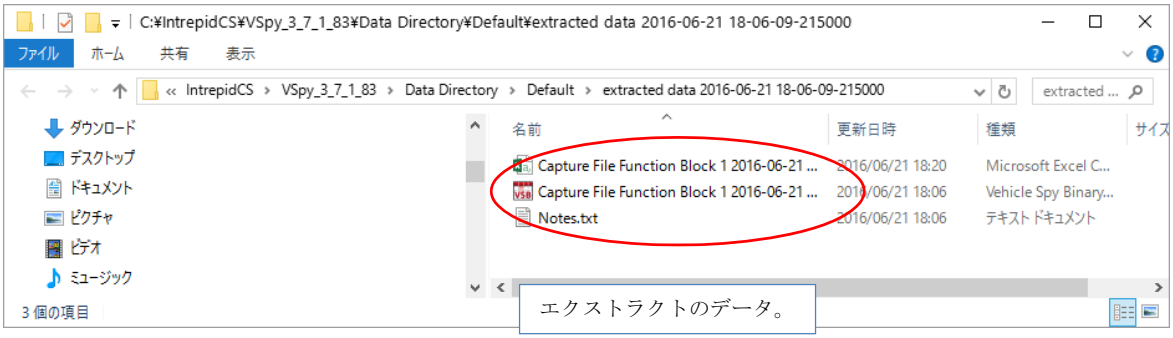


図 4.1.4.2

4.1.5. ロギングデータの確認

1. SD-Card よりエクストラクトされ PC に保存されたファイルの内容確認。

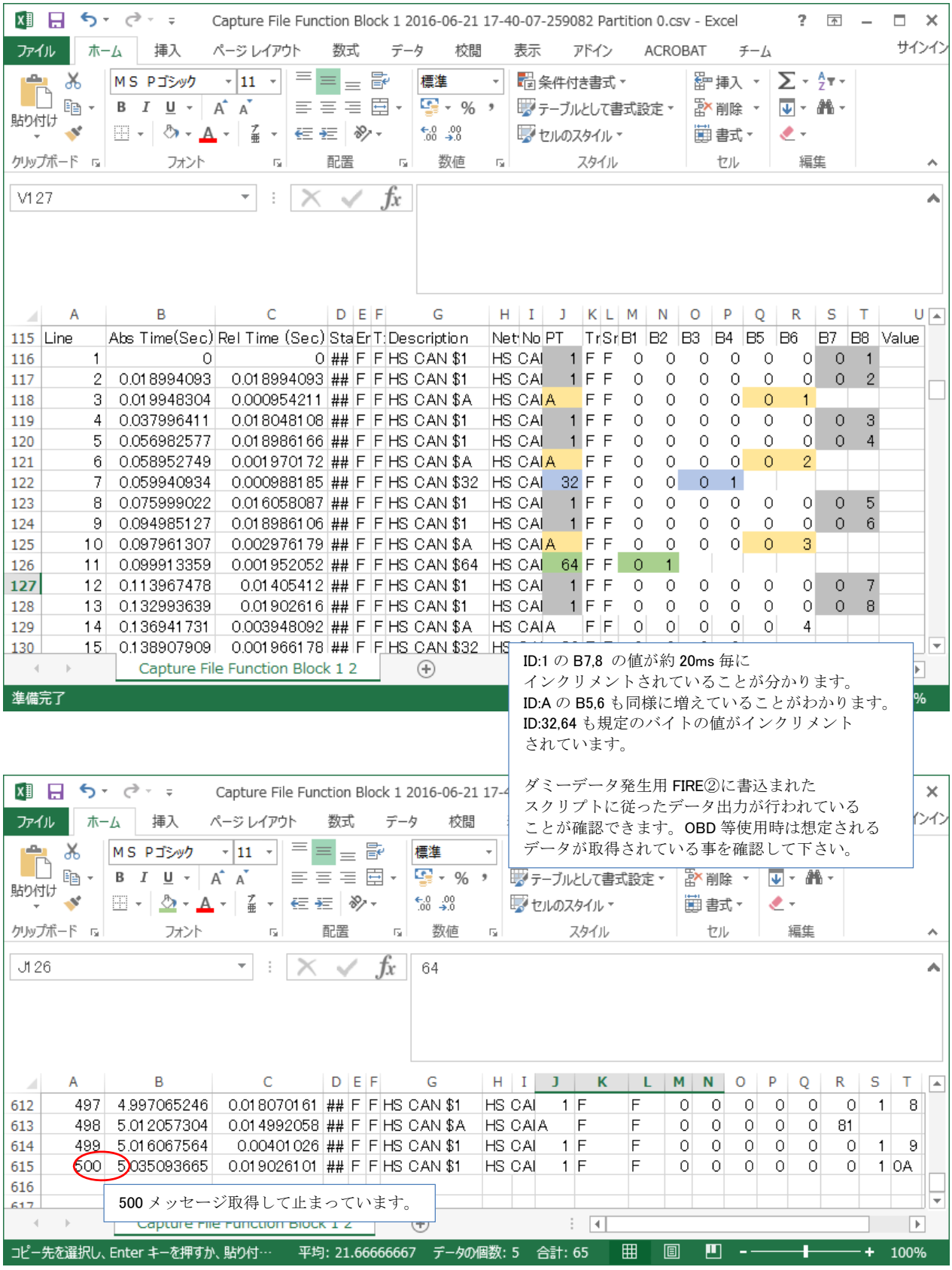


図 4.1.5.1

4.2.　メッセージの特定バイト値が規定の範囲内にあるときだけ1回ロギング

4.2.1.　スクリプト作成

1)　本ロギングでは、3 項のダミーデータを使用したロギングを行います。
2)　CAN　ID:0x64 の Byte2 の値が約 100 以上 150 以下の期間のみロギングを行います。

1.　データ受信メッセージの作成。　FIRE_DumyData_B01.vs3 で作成したメッセージを流用します。　受信メッセージをマニュアルで一つ一つ作成しても構いません。

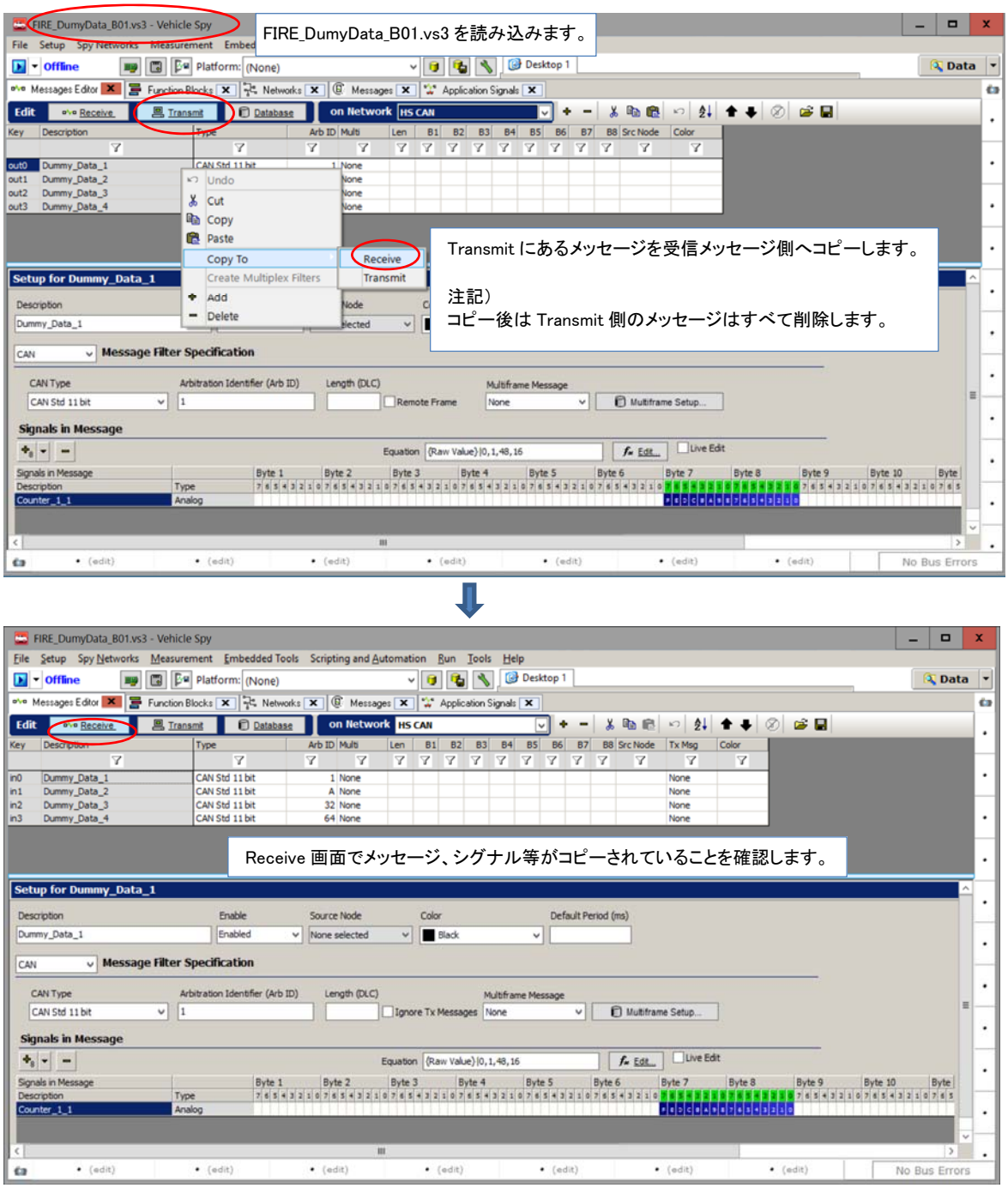
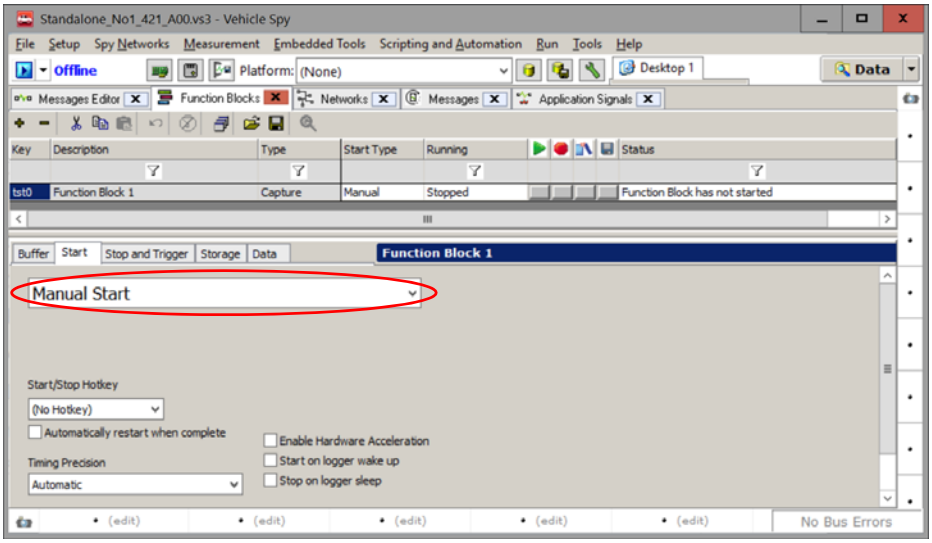
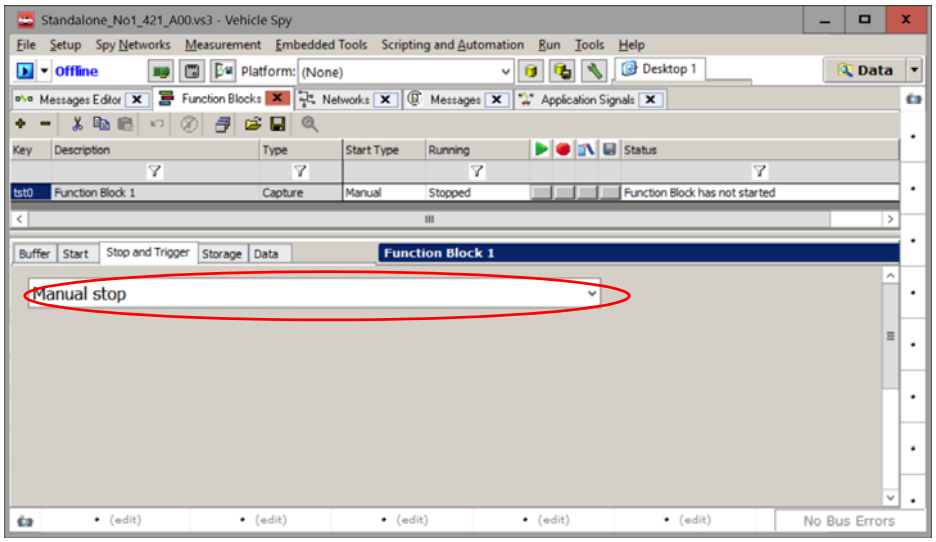


図 4.2.1.1

2.　受信メッセージ作成後 FIRE_DumyData_B01.vs3 と混同しない為に別名保存します。“File” -> “Save as”　今回は“Standalone_No1_421_A00.vs3“として保存。保存後 VSpy3 を終了します。

3. Capture Function Block の作成。VSpy を立ち上げて、4.2.1.2 項で保存した“Standalone_No1_421_A00.vs3“を読み込みます。以下のようにロギング条件（スタート、ストップ等）をすべてマニュアルに設定します。





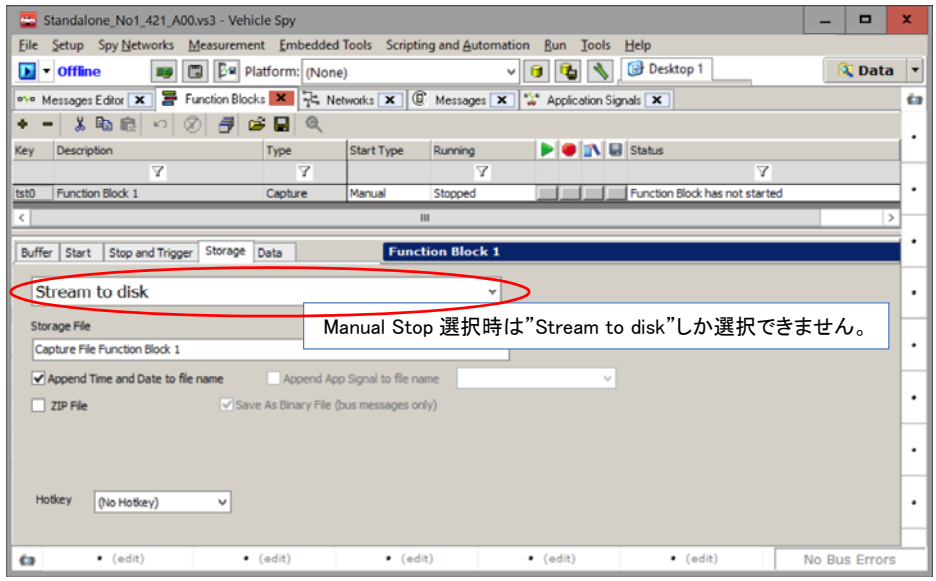


図 4.2.1.3

4. Script Function Block の作成。本スクリプトで 4.2.1.3 項作成の Capture Function Block の制御を行います。以下の Function Block 2 が実際のスクリプトとなります。

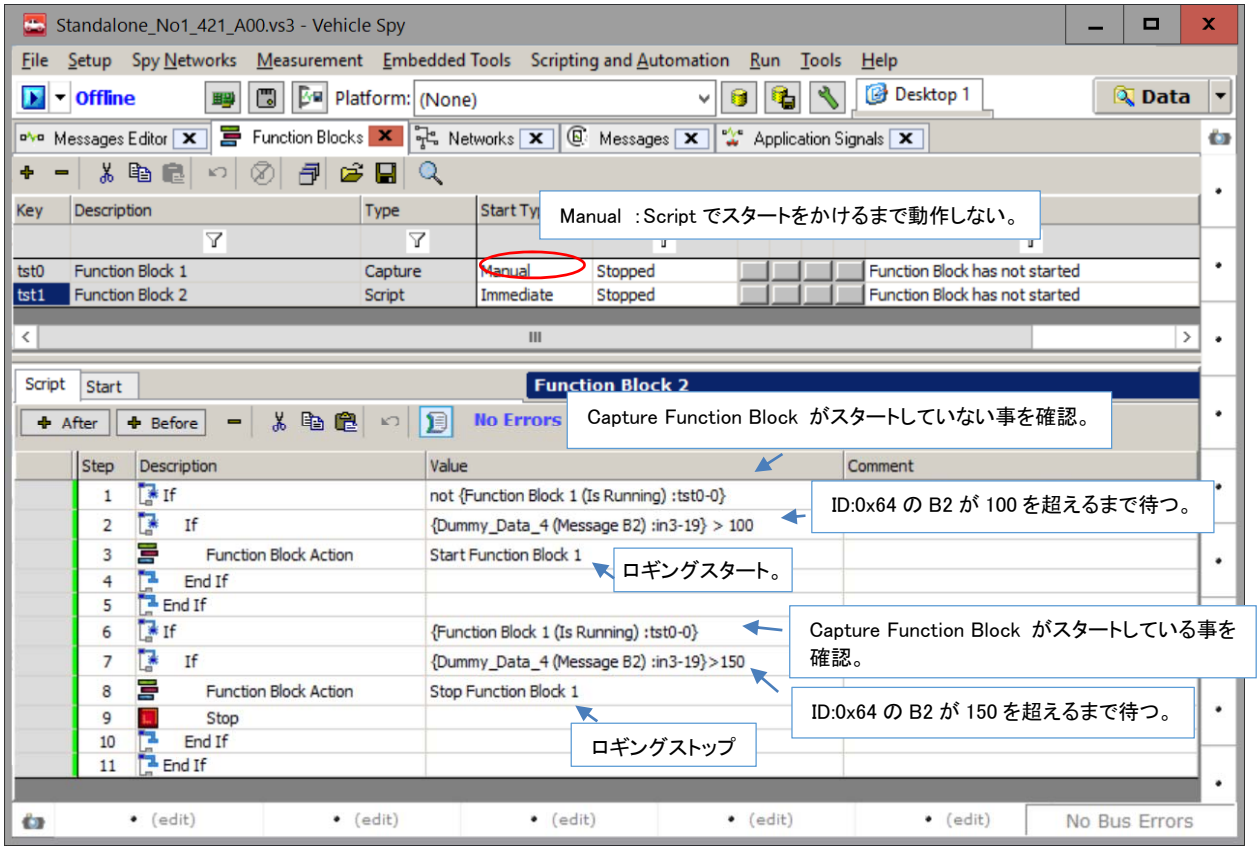


図 4.2.1.4

4.2.2. PCロギングでスクリプトの動作確認

1. スクリプトを SD-card へ書き込む前に PC モードでスクリプトの動作確認ができます。　スクリプトの記述が完了した後、以下のように Run with Transmit でロギングが開始されます。

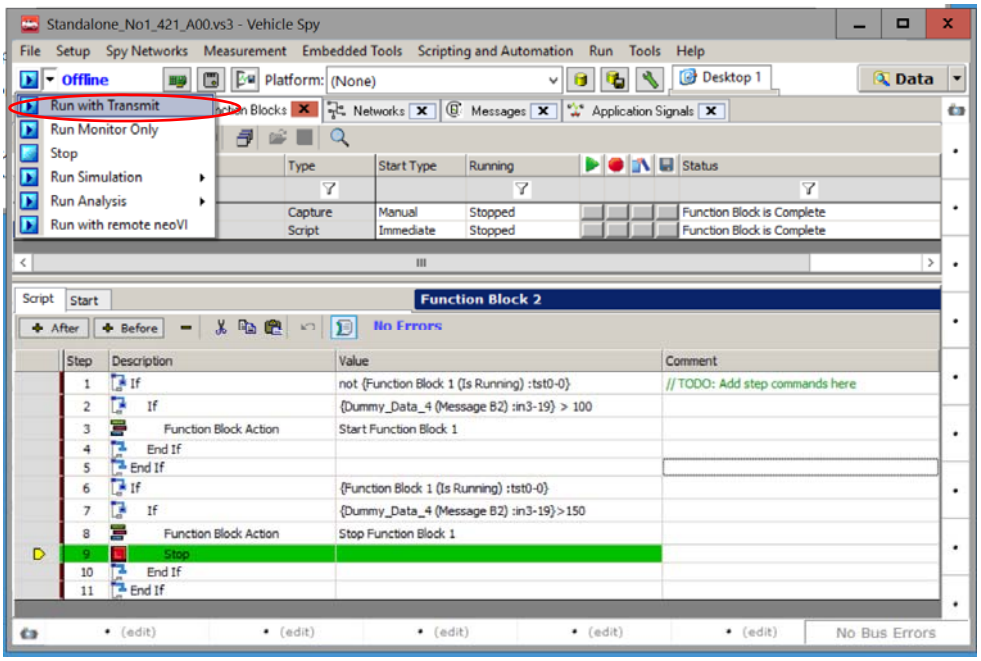


図 4.2.2.1

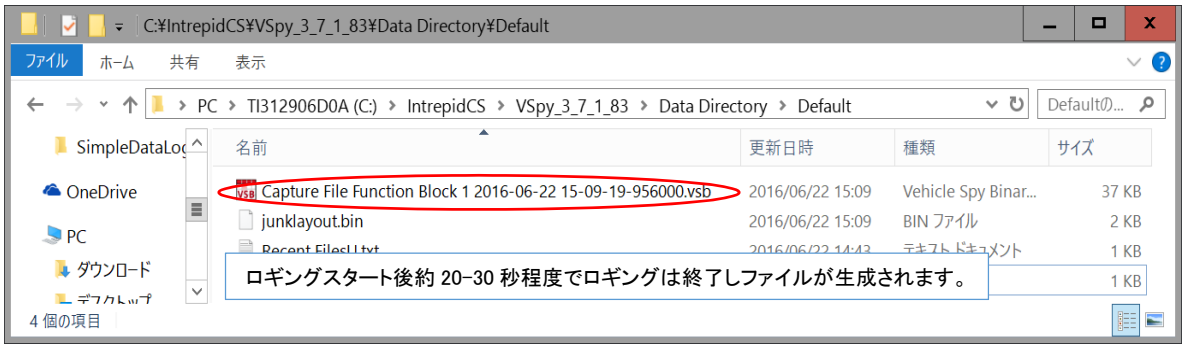


図 4.2.2.2

4.2.3.　スクリプトの書き込み

4.1.2 項参照。

4.2.4.　ロギングスタート及び確認

4.1.3 項参照。

4.2.5.　SD-Cardからのデータのエクストラクト方法

4.1.4 項参照。

4.2.6.　ロギングデータの確認

4.1.5 項参照。

4.3.　シグナル値の値が特定値になったらロギング

4.3.1.　準備

1. シグナル値をロギングするにはデータベースの設定が必要です。　設定方法は当社ウェブ上にあります“アプリケーションノート Vehicle Spy3 データベース　基礎編”を参照下さい。
2. 今回は 3 項のダミーデータを受信する為に、既に作成した Receive メッセージから VSDB ファイルを作成します。

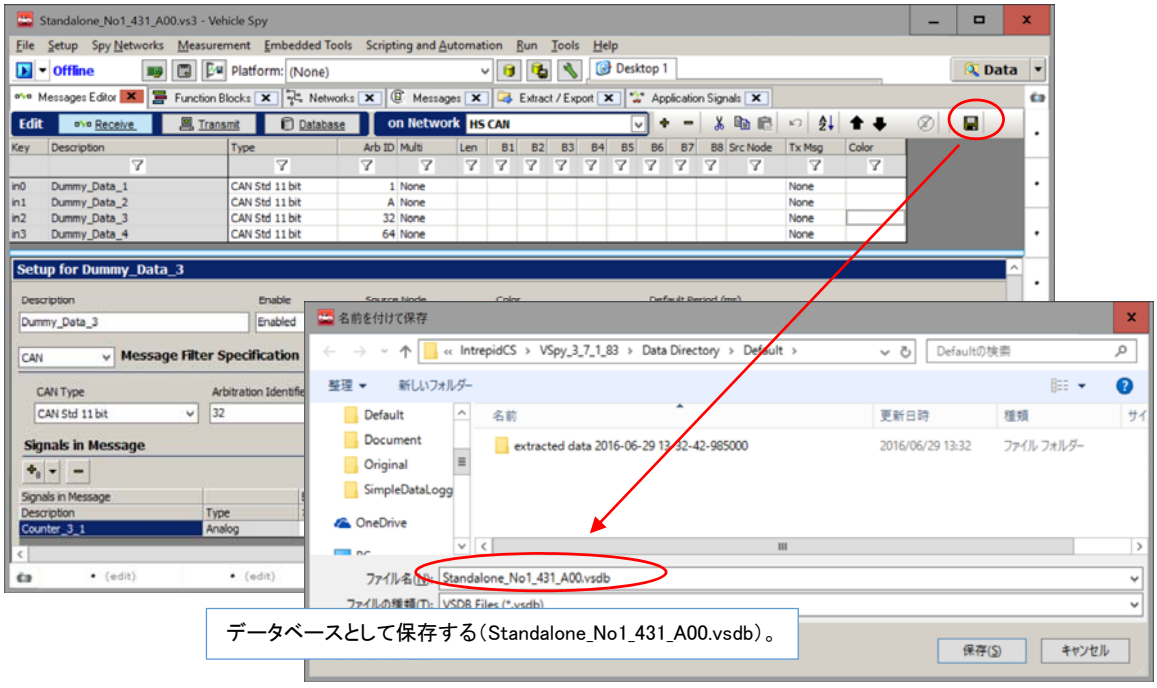


図 4.3.1.1

4.3.2.　スクリプト作成

1. 本ロギングでは、3 項で作成したダミーデータをロギングします。
2. 以下のロギング条件を使用します。
   “Start”条件：　Manual Start
   “Stop and Trigger”条件：　Collect in a one-shot bugger
   “Storage”条件：　Automatically save when complete
3. Function Block の内容。
   1) CAN　ID:0x64 のシグナル値（Counter_4_1）が 150 になったらロギングをスタート。 バッファサイズが 100 になったら保存。バッファサイズが一杯になった時点（100 メッセージ）で Automatically save when complete の条件が成立して保存となります。
   2) CAN　ID:0x64 のシグナル値（Counter_4_1）が 300 になったらロギングをスタート。 バッファサイズが 100 になったら保存。

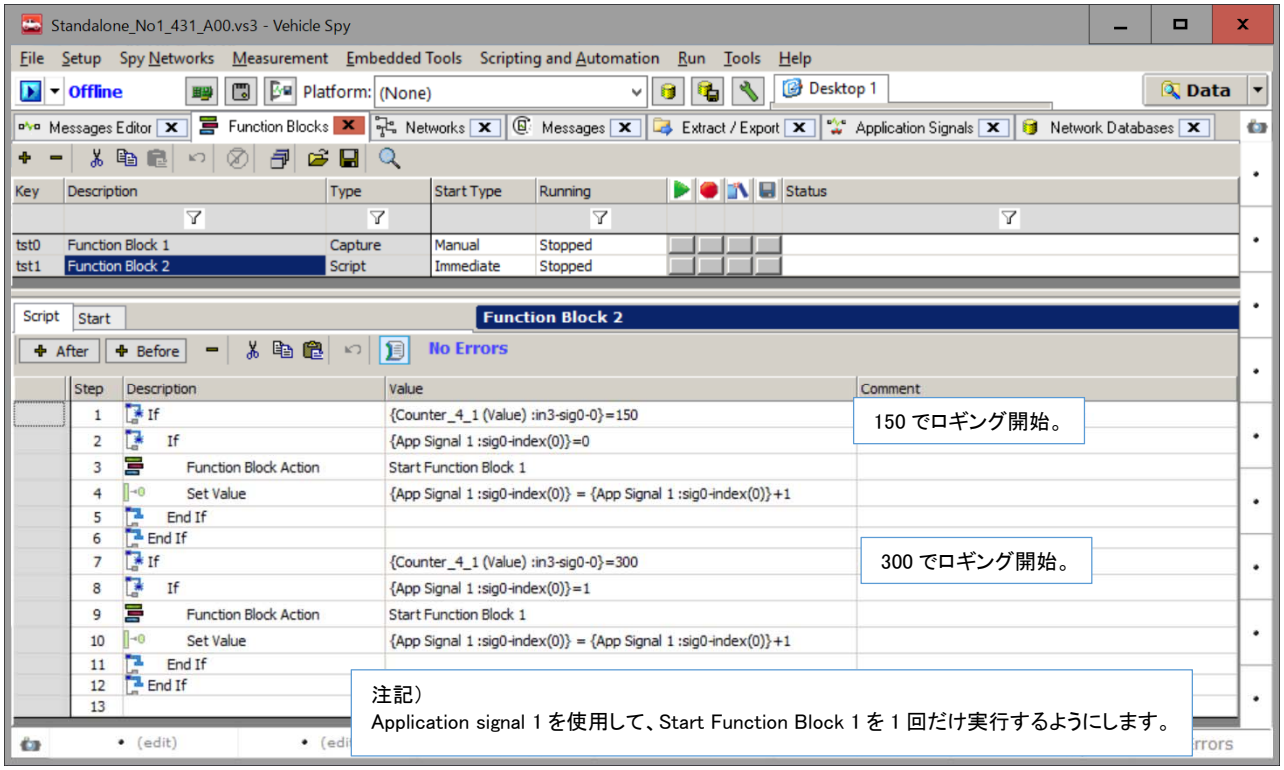


図 4.3.2.1

3. 本スクリプトをファイル名“Standalone_No1_431_A00.vs3”で保存します。

4.3.3.　スクリプトの書き込み

4.1.2 項参照。

4.3.4.　ロギングスタート及び確認

4.1.3 項参照。

4.3.5.　SD-Cardからのデータのエクストラクト方法

1. ロギングデータのエクストラクト
   4.1.4.1 項参照

2. データエクストラクト開始
   今回はシグナルデータも一緒にエクストラクトしますので、以下のように Vechicl Spy Log File(.CSV)にチェックを入れます

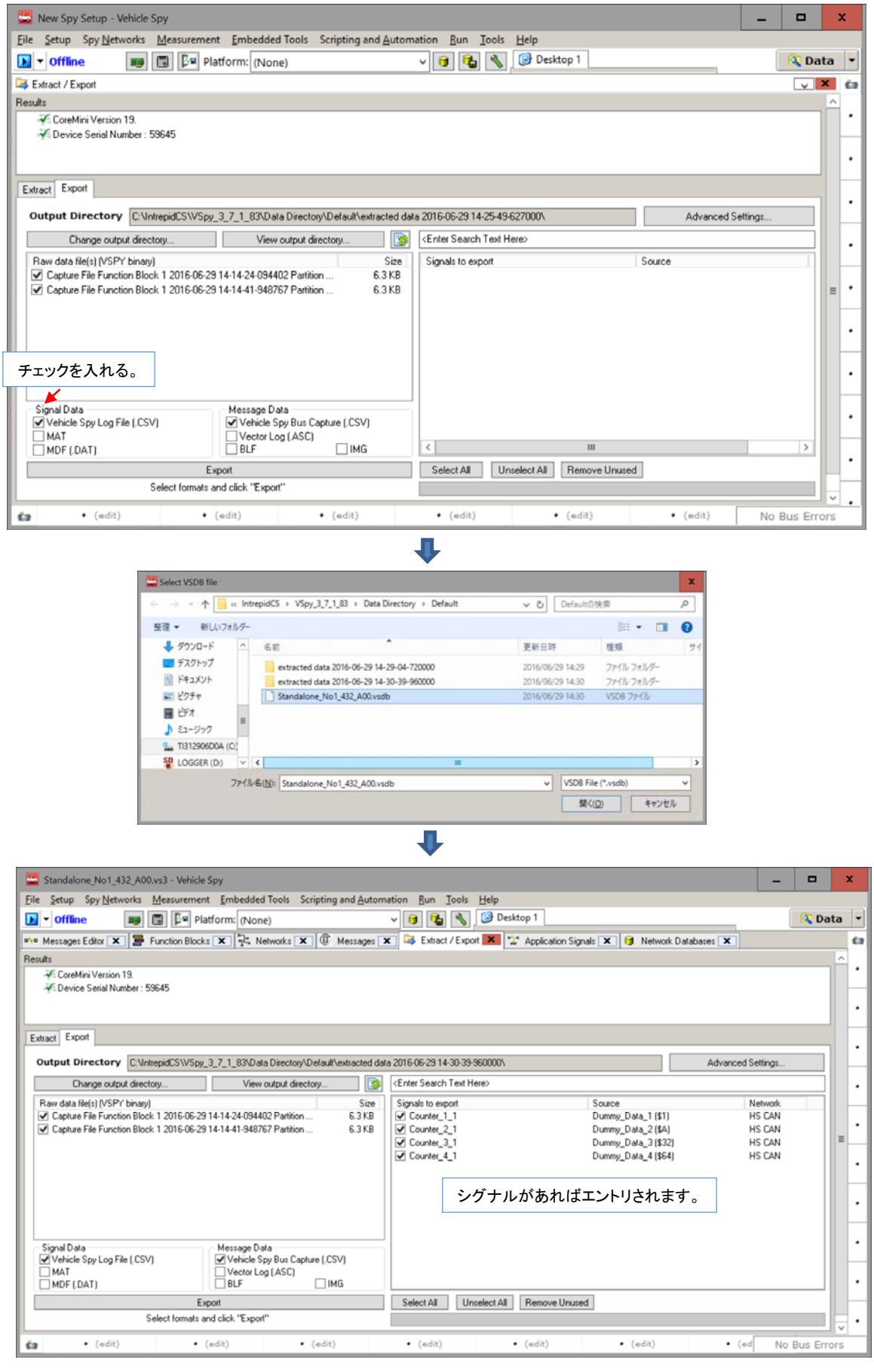


図 4.3.5.1

4.3.6. ロギングデータの確認

1. 4.1.5 項と同様にデータの確認を行います。

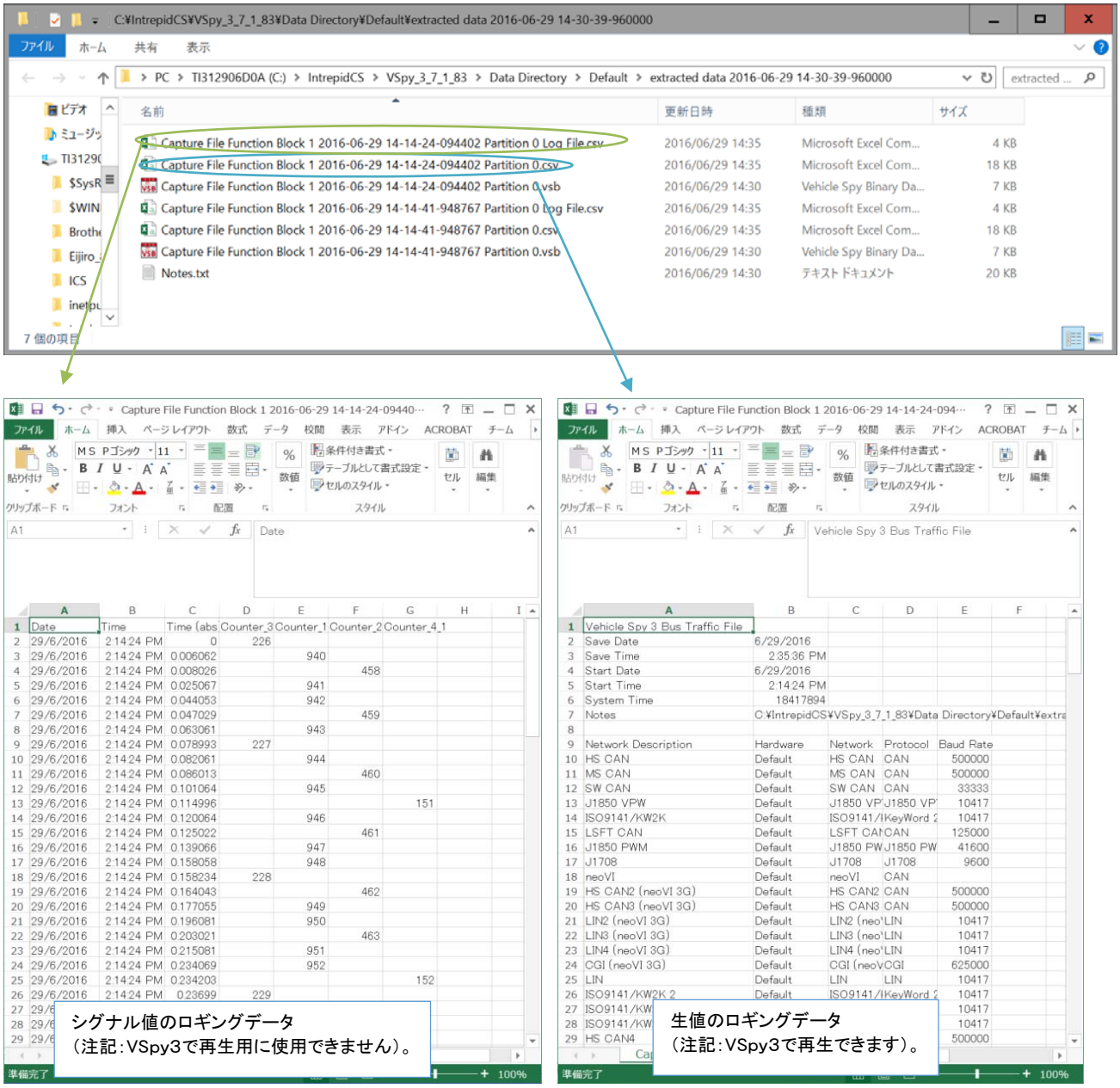


図 4.3.6.1

2. データは 4.3.2.3.(1)項と 4.3.2.3.(2)で指定される内容が保存されています。　尚、シグナル値のロギングデータは VSpy3 で再生することができませんので注意が必要です。VSpy3 での再生は生値を含んだ CSV ファイル、及び VSB ファイル等を使用します。

4.4. キャプチャ・ファンクションブロック作成時の注意事項

4.4.1. CoreMiniコンパイル時のワーニング

1. 以下の条件を選択した場合にはワーニングが出力されます。　この場合、スタンドアロンロギングが正常に行われませんので注意が必要です。

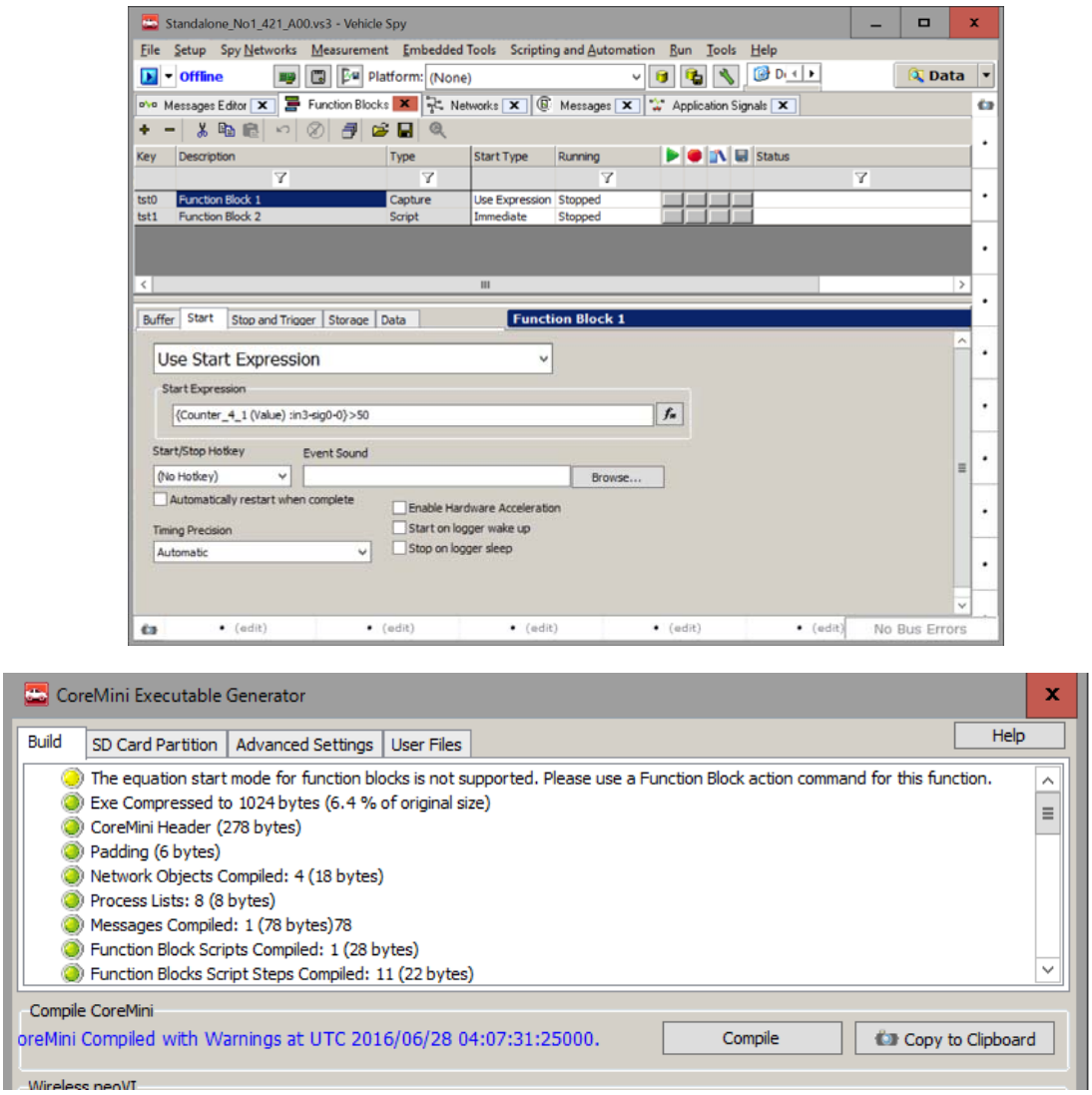


図 4.4.1.1

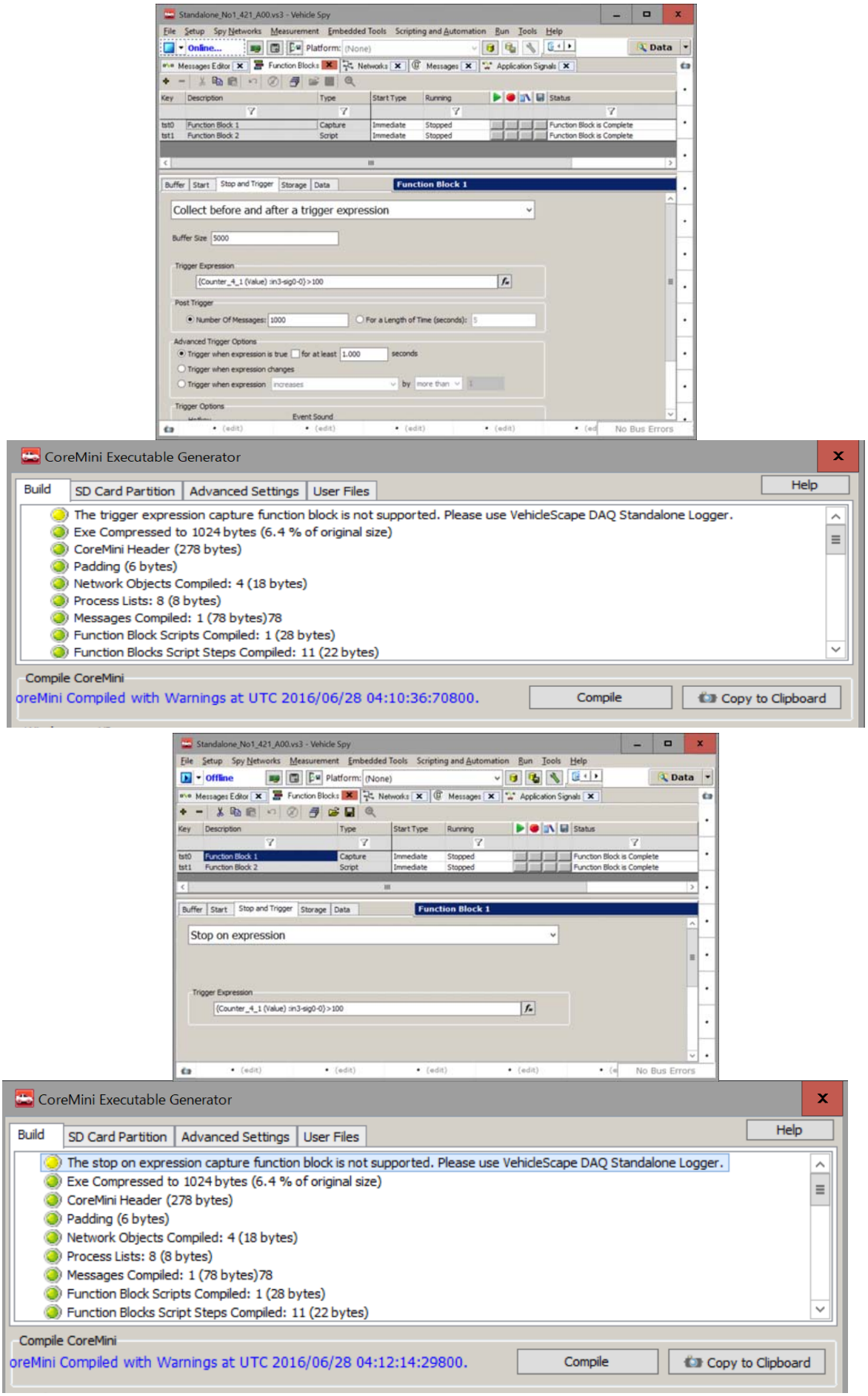
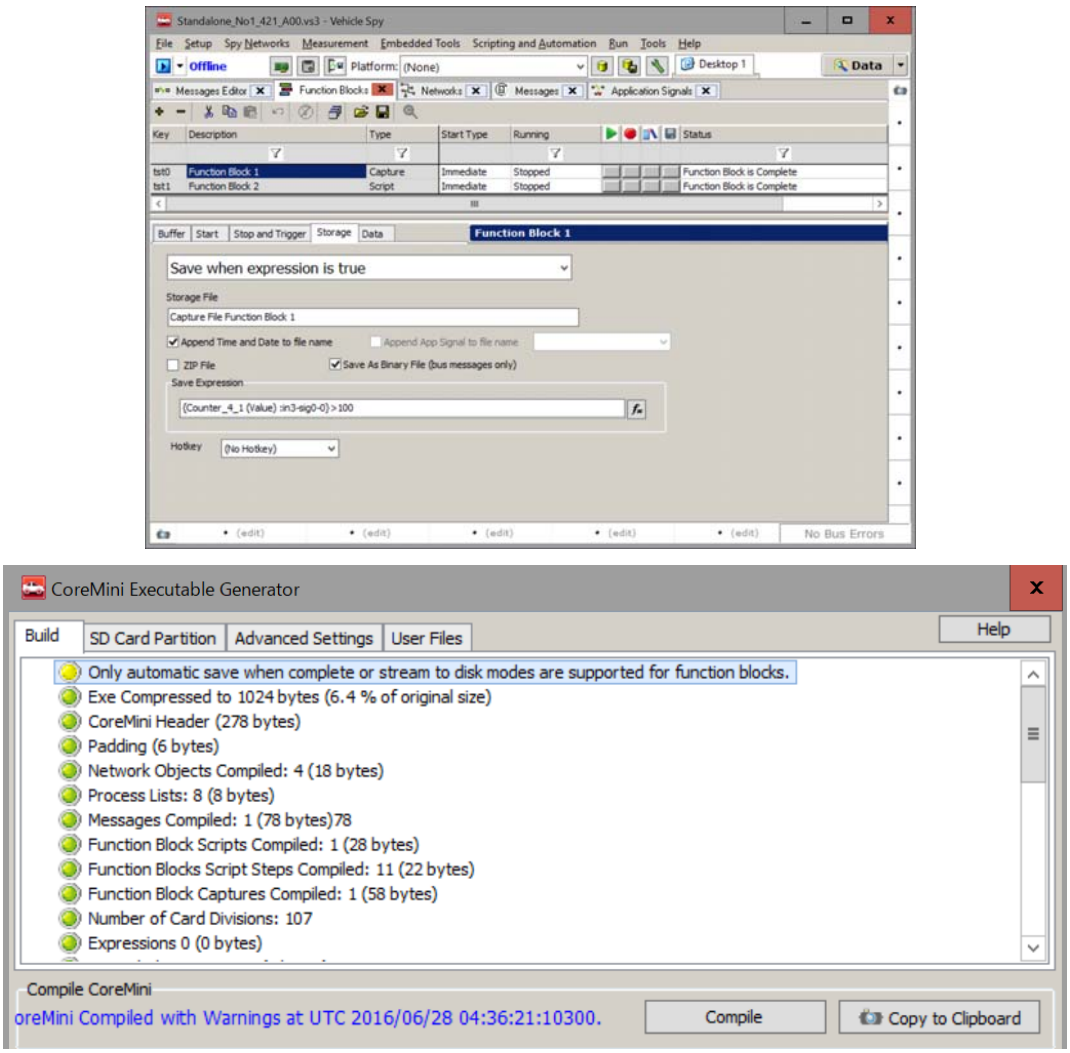

Standalone_No1_421_A00.vs3 - Vehicle Spy
Collect before and after a trigger expression
Buffer Size 5000
Trigger Expression
{Counter_4_1 (Value) :in3-sig0-0}>100
Post Trigger
Number Of Messages: 1000
Advanced Trigger Options
Trigger when expression is true
Trigger when expression changes
Trigger Options
CoreMini Executable Generator
Build
SD Card Partition
Advanced Settings
User Files
Help
The trigger expression capture function block is not supported. Please use VehicleScape DAQ Standalone Logger.
Exe Compressed to 1024 bytes (6.4 % of original size)
CoreMini Header (278 bytes)
Padding (6 bytes)
Network Objects Compiled: 4 (18 bytes)
Process Lists: 8 (8 bytes)
Messages Compiled: 1 (78 bytes)78
Function Block Scripts Compiled: 1 (28 bytes)
Function Blocks Script Steps Compiled: 11 (22 bytes)
Compile CoreMini
oreMini Compiled with Warnings at UTC 2016/06/28 04:10:36:70800.
Compile
Copy to Clipboard
Stop on expression
The stop on expression capture function block is not supported. Please use VehicleScape DAQ Standalone Logger.
oreMini Compiled with Warnings at UTC 2016/06/28 04:12:14:29800.

図 4.4.1.2


Standalone_No1_421_A00.vs3 - Vehicle Spy
File Setup Spy Networks Measurement Embedded Tools Scripting and Automation Run Tools Help
Offline
Platform: (None)
Desktop 1
Data
Messages Editor
Function Blocks
Networks
Messages
Application Signals
Key Description Type Start Type Running Status
tst0 Function Block 1 Capture Immediate Stopped Function Block is Complete
tst1 Function Block 2 Script Immediate Stopped Function Block is Complete
Buffer Start Stop and Trigger Storage Data
Function Block 1
Save when expression is true
Storage File
Capture File Function Block 1
Append Time and Date to file name
Append App Signal to file name
ZIP File
Save As Binary File (bus messages only)
Save Expression
(Counter_4_1 (Value) :in3-sig0-0) > 100
Hotkey
(No Hotkey)
No Bus Errors

CoreMini Executable Generator
Build
SD Card Partition
Advanced Settings
User Files
Help
Only automatic save when complete or stream to disk modes are supported for function blocks.
Exe Compressed to 1024 bytes (6.4 % of original size)
CoreMini Header (278 bytes)
Padding (6 bytes)
Network Objects Compiled: 4 (18 bytes)
Process Lists: 8 (8 bytes)
Messages Compiled: 1 (78 bytes)78
Function Block Scripts Compiled: 1 (28 bytes)
Function Blocks Script Steps Compiled: 11 (22 bytes)
Function Block Captures Compiled: 1 (58 bytes)
Number of Card Divisions: 107
Expressions 0 (0 bytes)
Compile CoreMini
oreMini Compiled with Warnings at UTC 2016/06/28 04:36:21:10300.
Compile
Copy to Clipboard

図 4.4.1.3

4.5.　PCモードとスタンドアロンモードの違い

1. キャプチャ・ファンクションブロックの設定内容によっては PC モード（*1）とスタンドアロンモードに違いが発生する場合があります。
   注記）
   (*1)　FIRE/RED に PC を接続（USB ケーブル接続）して PC 上からスクリプトを動作させるモードです。

4.5.1.　“Trigger“ Function Block Action

1. 以下の例は、特定のシグナル値（Counter_4_1）がある値（100 以上）を超えたときにキャプチャ・ファンクションブロック（Function Block1）にトリガーを発生させてロギングする設定です。　本設定はスタンドアロンモードでは有効になりません（PC モードではロギングできます）。

“Start”条件：　Start Immediately
“Stop and Trigger”条件：　Collection and after a manual trigger
“Storage”条件：　Automatically save when complete

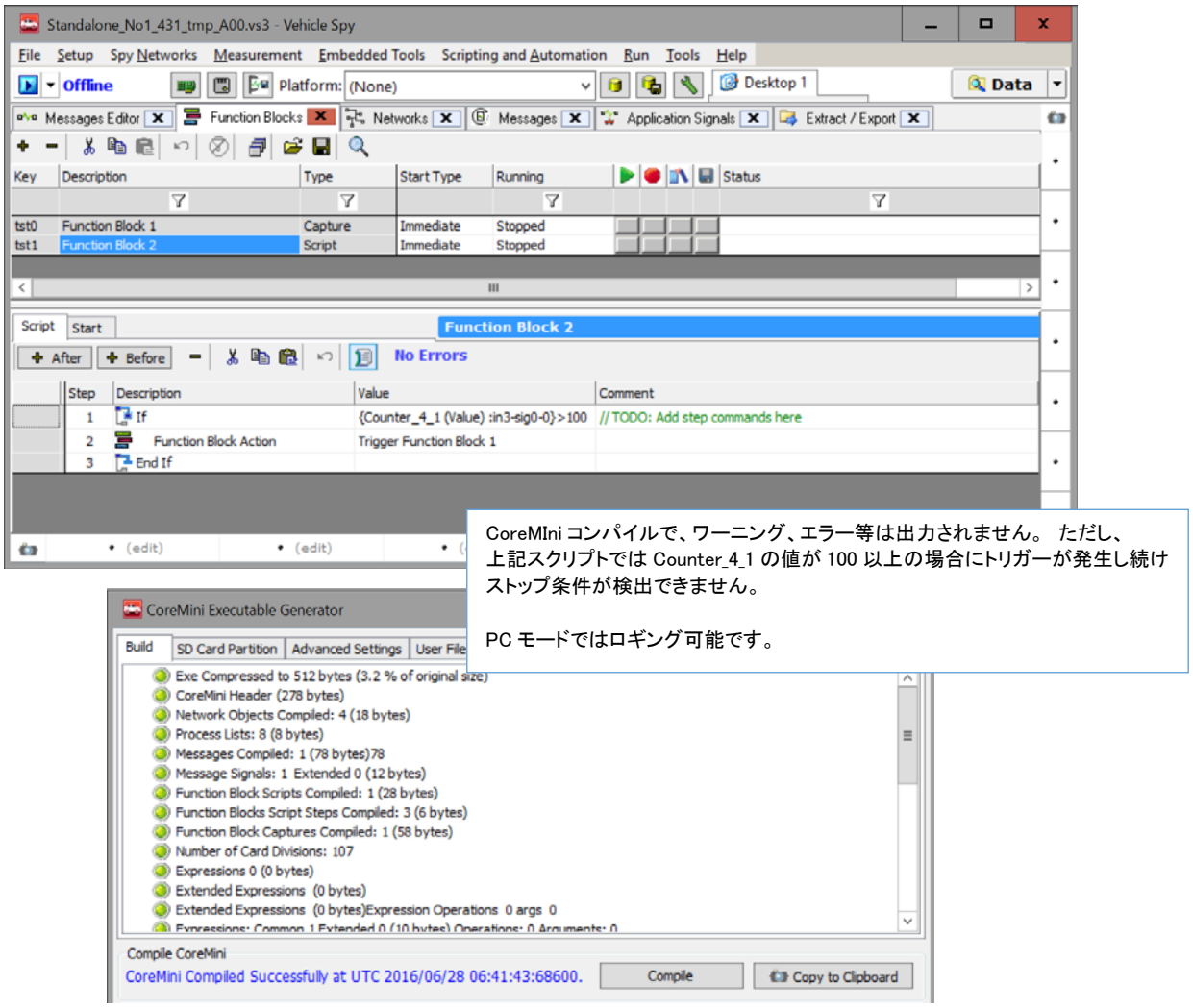


図 4.5.1.1

4.5.2. “Save” Function Block Action

1. 以下の例では、シグナル値（Counter_4_1）が 50 になったらロギングをスタートし、100 及び 125 で“Save“を実行します。本条件での設定はスタンドアロンモードで有効になりません（PC モードではロギングできます）。

“Start”条件：　Manual Start
“Stop and Trigger”条件：　Collect in a one-shot bugger
“Storage”条件：　Automatically save when complete

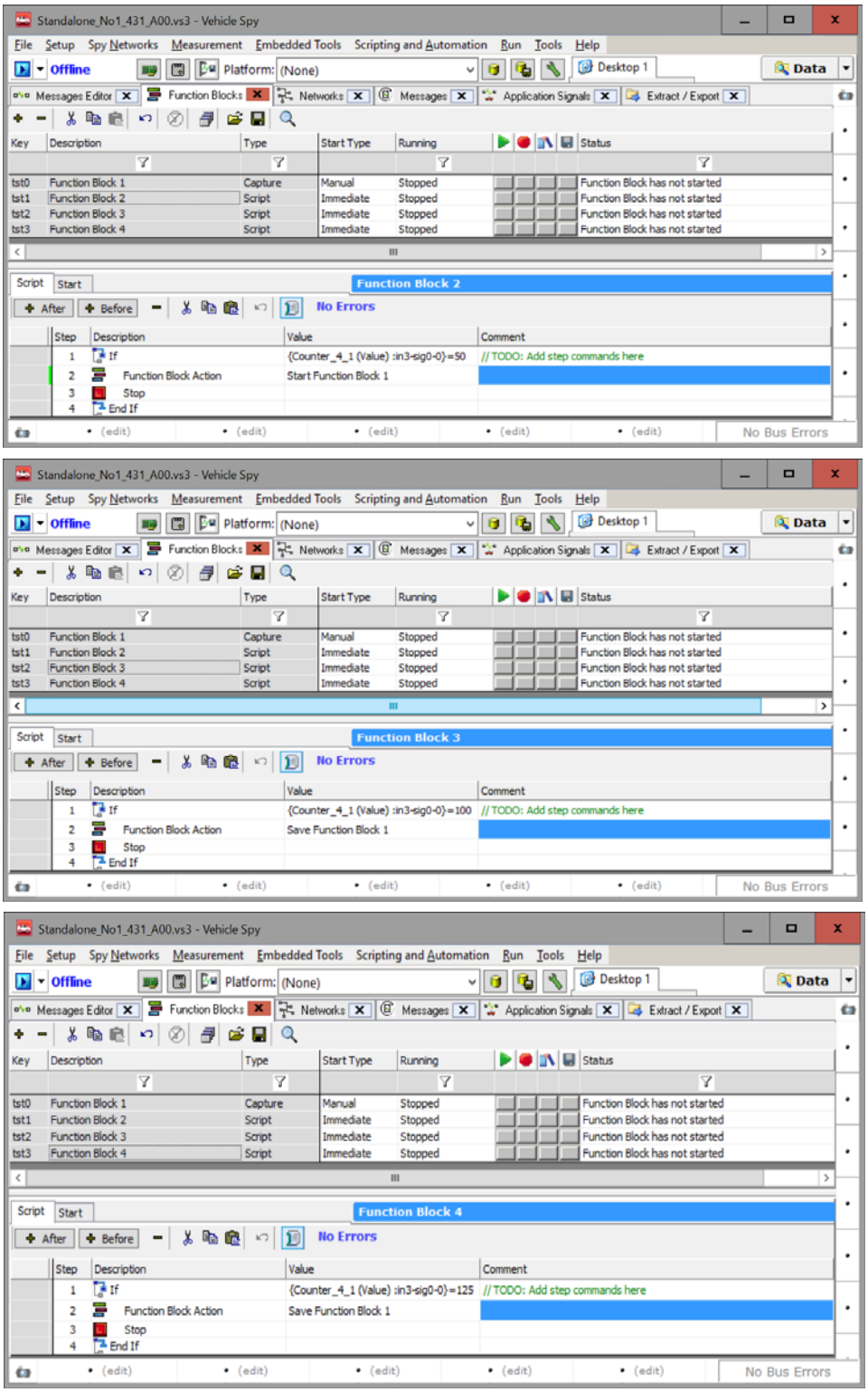


図 4.5.21

4.5.3. スタンドアロンモードでのキャプチャ・ファンクションブロックの設定方法

スタンドアロンモードでは 4.1 項、4.2 項、4.3 項で示すように、以下の 3 つの条件でキャプチャ・ファンクションブロックを設定し必要に応じてスクリプト・ファンクションブロックでロギング方法をコントロールすることをお奨め致します。

4.5.3.1. 条件1

"Start"条件：　Start Immediately
"Stop and Trigger"条件：　Collect in a one-shot bugger
"Storage"条件：　Automatically save when complete

4.5.3.2. 条件2

"Start"条件：　Manual Start
"Stop and Trigger"条件：　Manual Stop
"Storage"条件：　Stream to disk

4.5.3.3. 条件3

"Start"条件：　Manual Start
"Stop and Trigger"条件：　Collect in a one-shot bugger
"Storage"条件：　Automatically save when complete

5. スタンドアロンロギング中のLEDの点滅状態

スタンドアロンロギング中 FIRE/RED の LED ランプの状態は以下のようになります。

1. キャプチャ・ファンクションブロックを使用したスタンドアロンロギング中は、赤 LED が早い点滅、緑 LED が OFF 状態となります。
   詳細は以下を参照下さい。
   1) http://www.intrepidcs.com/support/ICSDocumentation/neoVIHardware/neoVIhelpdoc.html

6. サンプルプログラム

本資料で使用したサンプルファイルまたはデータベース等は当社ウェブ上に以下のファイル名で掲載してあります。
ファイル名：Capture_DAQ_Standalne_Log_Example_A00.zip

7. まとめ

本アプリケーションノートは、当社の Vehicle Spy 3 ソフトウェアのキャプチャ・ファンクションブロックに関して解説していますが、機能のすべてを網羅した解説とはなっておりません。

ご不明な点等ございましたら、icsjapan@intrepidcs.com までメールで質問して頂ければと思います。

8. 変更履歴

日付	バージョン	変更内容	作成者
2015/07/06	1.0	初版作成	春川

9. 用語一覧

PC ロギング	当社ハードウェア製品を PC に接続した状態で、車載ネットワーク等からのデータを取得し、そのデータを PC 上に保存すること。
スタンドアロンロギング	PC から独立した状態で、当社ハードウェア製品内蔵の SD カードに対して、車載ネットワーク等からのデータを保存すること。
スタンドアロンモード	neoVI ハードウェアは PC から独立した状態　（ワイヤ接続していない状態）。
バッファー	測定中のデータの PC メモリ上の保存領域。Messages 画面上の Save ボタン押下により、このバッファー上のデータが PC 上へ保存されます
スタートボタン	Vehicle Spy 画面左上の青い矢印ボタン。
生値、生データ	CAN などの車載ネットワーク上の、デコードされていない生のメッセージ。単にメッセージとも言います。
シグナル	生値を、RPM や車速など人間が読み取れる内容に変換したデータ。
データディレクトリボタン	Vehicle Spy 画面右上のフォルダー印のボタンを押下すると、Vehicle Spy のデータ出力先フォルダーが開きます。
CoreMini（コアミニ）	neoVI のスタンドアロン動作に必要な設定ファイル。用例：CoreMini をコンパイルする。CoreMini を neoVI へダウンロードする。
vsb（Vehicle Spy Binary）	Vehicle Spy 固有の、CAN メッセージ等の生値保存形式。