

LABCAR-PINCONTROL V2.1

ユーザーズガイド



著作権について

本書のデータを ETAS GmbH からの通知なしに変更しないでください。ETAS GmbH は、本書に関してこれ以外の一切の責任を負いかねます。本書に記載されているソフトウェアは、お客様が一般ライセンス契約あるいは単一ライセンスをお持ちの場合に限り使用できます。ご利用および複写はその契約で明記されている場合に限り、認められます。

本書のいかなる部分も、ETAS GmbH からの書面による許可を得ずに、複写、転載、伝送、検索システムに格納、あるいは他言語に翻訳することは禁じられています。

© **Copyright 2013** ETAS GmbH, Stuttgart, Germany

本書で使用する製品名および名称は、各社の（登録）商標あるいはブランドです。

R2.1.0 JP - 6.2013

目次

1	はじめに	5
1.1	本書について	5
1.2	本書の使用方法	5
1.2.1	表現について	5
1.2.2	表記上の規則	6
2	LABCAR-PINCONTROL V2.1 の操作方法	8
2.1	LABCAR-PINCONTROL V2.1 の設定	8
2.1.1	PC のネットワークインターフェースカードの設定	8
2.1.2	イーサネットの設定	9
2.1.3	ES4440.1 のシステム設定	10
2.2	ワイヤハーネスファイル	12
2.2.1	ワイヤハーネスファイルの作成	13
2.3	LABCAR-PINCONTROL V2.1 の操作	17
2.4	メインメニュー	29
3	API	31
3.1	概要	31
3.1.1	COM コントローラの機能	31
3.1.2	CAN-API の使用	32
3.1.3	COM-API および CAN-API のデータ内容	32
3.2	CAN メッセージの設定とシーケンス	32
3.2.1	スタンドアロン構成におけるシングルエラーのシミュレーション	33
3.2.2	スタンドアロン構成におけるマルチエラーのシミュレーション	34
3.2.3	マスタ/スレーブ構成におけるシングルエラーのシミュレーション	34
3.2.4	マスタ/スレーブ構成におけるマルチエラーのシミュレーション	37
3.3	初期設定	38
3.3.1	COM コントローラの初期設定	38
3.3.2	CAN を使用する場合の初期設定	38

3.4	コマンドの詳細説明	38
3.4.1	CAN コマンドの構造	39
3.4.2	全コマンドに共通する情報	40
3.4.3	エラーコード	40
3.4.4	IDN コマンド	42
3.4.5	Open_Load	43
3.4.6	Open_Load_realtime	44
3.4.7	ShortCut_xUBATTy_20A	45
3.4.8	ShortCut_xUBATTy_20A_realtime	46
3.4.9	Pin2PinFirstChWithoutLoad	47
3.4.10	Pin2PinSecondChannelWithoutLoad	48
3.4.11	Pin2PinFirstChRealtimeWithLoad	49
3.4.12	Pin2PinSecondChRealtimeWithLoad	50
3.4.13	RInline_realtime	51
3.4.14	Pullup_Pulldown_xUBATTy_20A_realtime	53
3.4.15	Open_Load_400V	54
3.4.16	ShortCut_xUBATTy_400V	55
3.4.17	ShortCut_xUBATTy_400V_Ex	56
3.4.18	Pin_2_Pin_400V	57
3.4.19	Pin_2_Pin_400V_Ex	58
3.4.20	Reset_all_errors	59
3.4.21	Activate_relay	60
3.4.22	Activate_realtime_switch	62
3.4.23	TestFuses	64
3.4.24	CurrentMeasurement	65
4	お問い合わせ先	66
	索引	67

1 はじめに

LABCAR-PINCONTROL V2.1 は、ES4440.1/2 コンパクト故障シミュレーションモジュールに付属するソフトウェアです。ES4440.1/2 コンパクト故障シミュレーションモジュールを操作してリアルタイムエラーシミュレーションを行い、ECU の診断機能のテストを行うことができます。

LABCAR-PINCONTROL V2.1 には、ES4440.1/2 の設定や操作を行うためのユーザーインターフェースと、自動テストのための COM コントローラが含まれています。

LABCAR-PINCONTROL V2.1 には以下の機能があります。

- マニュアル操作によるエラーシミュレーションの実行
 - “Failure Set” (本書では「故障セット」と記します) の作成と管理
「故障セット」とは、ある特定の機能に関連する ECU シグナルのグループ (例: ラムダセンサ関連の全シグナル) を定義したものです。
 - エラーシミュレーションを行うシグナルとエラー種別の選択
 - 接点不良エラーのパラメータ (周波数、デューティサイクル) の設定
 - マウスクリックによるエラーシミュレーションの実行
- ES4440.1/2 コンパクト故障シミュレーションモジュールのコンフィギュレーション設定
 - モジュールの IP アドレスと CAN アドレスの設定
 - 3 種類のシステムステータス (スタンドアロン、マスタ、スレーブ) の選択
 - ES4440.1/2 のセルフテスト (自己診断テスト) とヒューズテストの実行
- テストの自動実行のための COM-API

1.1 本書について

本書では、以下の事柄について説明しています。

- 第 1 章 「はじめに」 (5 ページ)
本章です。
- 第 2 章 「LABCAR-PINCONTROL V2.1 の操作方法」 (8 ページ)
LABCAR-PINCONTROL V2.1 の設定方法と操作方法について説明します。
- 第 3 章 「API」 (31 ページ)
ES4440.1/2 コンパクト故障シミュレーションモジュールの自動テストを行うための情報です。自動テストには、LABCAR-PINCONTROL V2.1 の COM コントローラ、または CAN を使用します。

1.2 本書の使用方法

1.2.1 表現について

ユーザーが実行するすべてのアクションは、いわゆる “Use-Case” 形式で記述されています。つまり以下に示すように、操作を行う目標がタイトルとして最初に簡潔に定義され、その下に、その目標を実現するために必要な操作手順が列挙されています。

目標の定義：

前置き ...

- 手順 1
手順 1 についての説明 ...
- 手順 2
手順 2 についての説明 ...
- 手順 3
手順 3 についての説明 ...

まとめ ...

具体例：

新しいファイルを作成する：

新しいファイルを作成する際は、他のファイルをすべて閉じておきます。

- **File** → **New** コマンドを選択します。
“Create file” ダイアログボックスが開きます。
- 新しいファイルの名前を、“File name” フィールドに入力します。
ファイル名は 8 文字以内でなければなりません。
- **OK** をクリックします。

新しいファイルが作成され、ユーザーが指定した名前で保存されます。このファイルを使用して以降の操作を行います。

1.2.2 表記上の規則

本書は以下の規則に従って表記されています。

表記例	説明
File → Exit コマンドを選択して、...	メニューコマンドは、 青の太字 で表記します。
OK をクリックして、...	ユーザーインターフェース上のボタン名は、 青の太字 で表記します。
<Ctrl> を押して、...	キーボードの各キーは、 <> で囲んで表記します。
“Open File” ダイアログボックスが開きます。	プログラムウィンドウ、ダイアログボックス、入力フィールド等のタイトルは、“ ” で囲んで表記します。
setup.exe ファイルを選択します。	リストボックス、プログラムコード、ファイル名、パス名等のテキスト文字列は、Courier フォントで表記します。

その他、重要な語や新出の語は**太字**または**斜体**、または「」で囲んで示されていて、特に重要な注意事項は、以下のように表記されています。**注記**

ユーザー向けの重要な注意事項

また PDF 文書において、索引、および他の部分を参照する箇所（例：「xxxx を参照してください」と記述された箇所の「xxxx」の部分）については、その参照先へのリンクが設けられているので、必要な参照箇所を素早く見つけることができます。

2 LABCAR-PINCONTROL V2.1 の操作方法

本章では、LABCAR-PINCONTROL V2.1 の設定と操作について説明します。

- LABCAR-PINCONTROL V2.1 の設定 (8 ページ)
LABCAR-PINCONTROL V2.1 を操作するための準備作業について説明します。
- ワイヤハーネスファイル (12 ページ)
プロジェクトの中核となる「ワイヤハーネスファイル」の作成方法について説明します。
- LABCAR-PINCONTROL V2.1 の操作 (17 ページ)
ユーザーインターフェースを操作してエラーをシミュレートする方法を説明します。
- メインメニュー (29 ページ)
LABCAR-PINCONTROL V2.1 のメニューコマンドについて説明します。

2.1 LABCAR-PINCONTROL V2.1 の設定

PINCONTROL での作業を始める前に、イーサネット接続と ES4440.1/2 システムについての設定を行う必要があります。

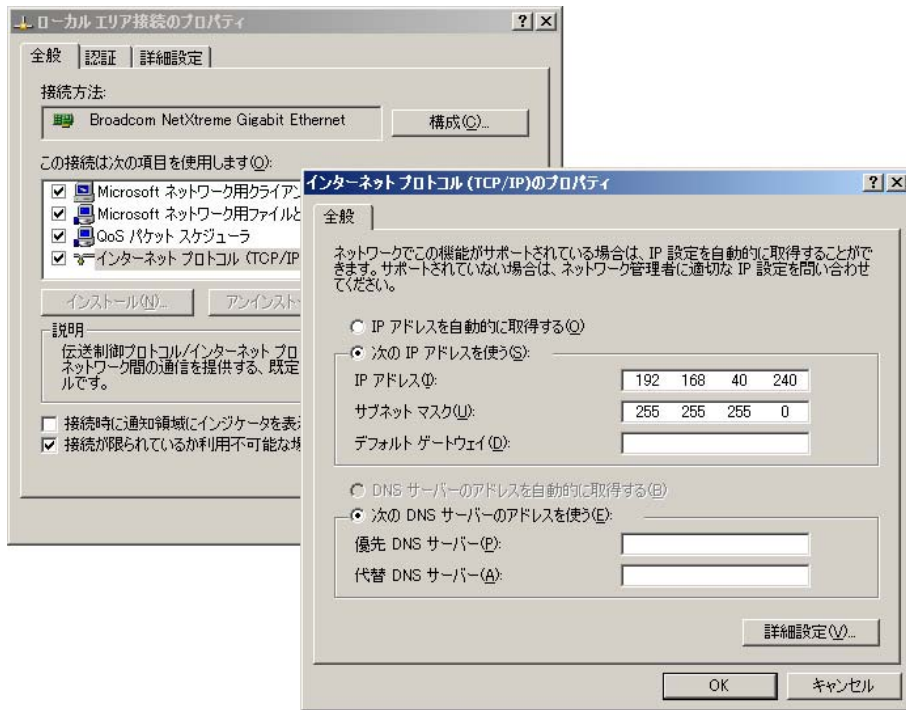
2.1.1 PC のネットワークインターフェースカードの設定

ES4440.1/2 コンパクト故障シミュレーションモジュールの操作に使用するネットワークインターフェースカードについて、以下の設定を行ってください。

TCP/IP を設定する：

- Windows のスタートメニューから **設定** → **コントロールパネル** を選択します。
- “コントロールパネル” ウィンドウの **ネットワーク接続** をダブルクリックします。
- 使用する接続/デバイスを選択します。
- そのエントリを右クリックして **プロパティ** を選択します。
“ローカルエリア接続プロパティ” ダイアログボックスが開きます。
- “インターネットプロトコル (TCP/IP)” というコンポーネントを選択します。

- **プロパティ** をクリックします。
"インターネットプロトコル (TCP/IP) のプロパティ" ダイアログボックスが開きます。



- 以下の値を、上図のように設定します。
IP アドレス : 192.168.40.240
サブネットマスク : 255.255.255.0
IP アドレスは、任意の有効なアドレスを設定することができます。
- **OK** をクリックしてすべてのウィンドウを閉じます。

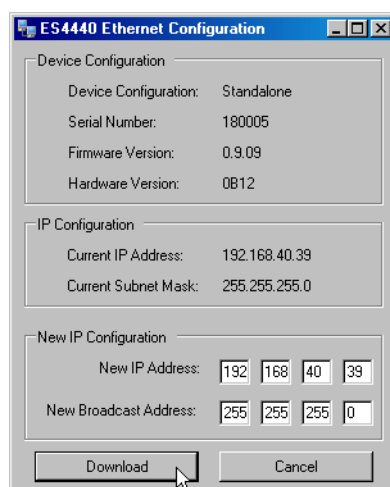
2.1.2 イーサネットの設定

本項では、ES4440.1/2 コンパクト故障シミュレーションモジュールにイーサネットアドレスを割り当てる方法について説明します。

イーサネットについて設定する：

- LABCAR-PINCONTROL V2.1 のメインメニューから **Tools → ES4440 Ethernet Configuration** を選択します。
- 設定時には設定を行うデバイスしかオンにできないので、その旨を通知するメッセージが表示されます。
- **OK** をクリックします。

- “ES4440 Ethernet Configuration” ダイアログボックスが開きます。



このダイアログボックスには、イーサネット接続に関する設定情報（使用する IP アドレスとサブネットマスクなど）が表示されます。

- アドレスを変更する場合は “New IP Configuration” フィールドに新しい値を入力します。
- **Download** をクリックすると、設定内容がハードウェアにダウンロードされます。

または

- **Cancel** をクリックすると、変更操作が取り消されてダイアログボックスが閉じます。

2.1.3 ES4440.1 のシステム設定

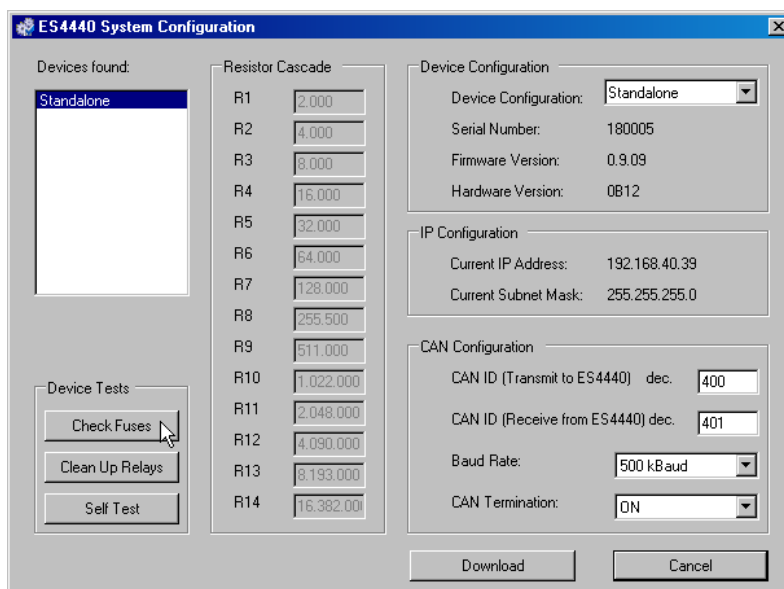
システム設定では、各ハードウェアのステータス（スタンドアロン、マスタ、スレーブ）を指定し、さらに、CAN を使用して制御を行う場合は CAN インターフェースについて設定します。

注記

本項で説明する方法でシステムを設定するには、設定対象のハードウェアがイーサネット接続されている必要があります。

- LABCAR-PINCONTROL V2.1 のメインメニューから、**Tools** → **ES4440 System Configuration** を選択します。

“ES4440 System Configuration” ダイアログボックスが開きます。



ここでは以下の設定を行うことができます。

デバイスを設定する：

- “Devices found” フィールドから、設定の対象とする ES4440.1/2 を選択します。
- “Device Configuration” フィールドで、モジュールのステータス (Standalone、Master、Slave1、...Slave14) を選択します。
- 上記以外の項目を変更する必要がない場合は、**Download** をクリックします。

ハードウェアの設定変更が実行され、ダイアログボックスが閉じます。

- 他に変更する項目がある場合は、それらを変更し、最後に上述のように設定変更を実行します。

IP 設定を確認する：

“IP Configuration” フィールドに、現在選択されているハードウェアの IP アドレスとサブネットマスクが表示されます。

CAN インターフェースを設定する：

- ES4440.1/2 の制御を CAN インターフェース経由で行う場合は、“CAN Configuration” の各フィールドで CAN インターフェースの設定を行います。
 - CAN ID (Transmit to ES4440) dec.
ES4440.1 への送信メッセージにセットするデバイス ID を指定します。

- CAN ID (Receive from ES4440) dec.
ES4440.1 からの送信メッセージにセットするデバイス ID を指定します。
- Baud Rate:
転送速度を指定します。“500 kBaud” から “1 Mbaud” までの値を選択できます。
- CAN Termination:
デバイスに CAN ターミネータが取り付けられているかどうかを指定します。

抵抗カスケードについての情報を確認する：

内部抵抗カスケードの実際の抵抗値は “Resistor Cascade” フィールドに表示されます。

リレー接点のクリーニングとセルフテストを実行する：

- “Device Tests” フィールドで、モジュールテスト用のアクションを実行できます。各アクションについての詳細は、「リレー接点をクリーニングする：」（26 ページ）、「ヒューズをテストする：」（27 ページ）、「セルフテストを実行する：」（27 ページ）を参照してください。

2.2 ワイヤハーネスファイル

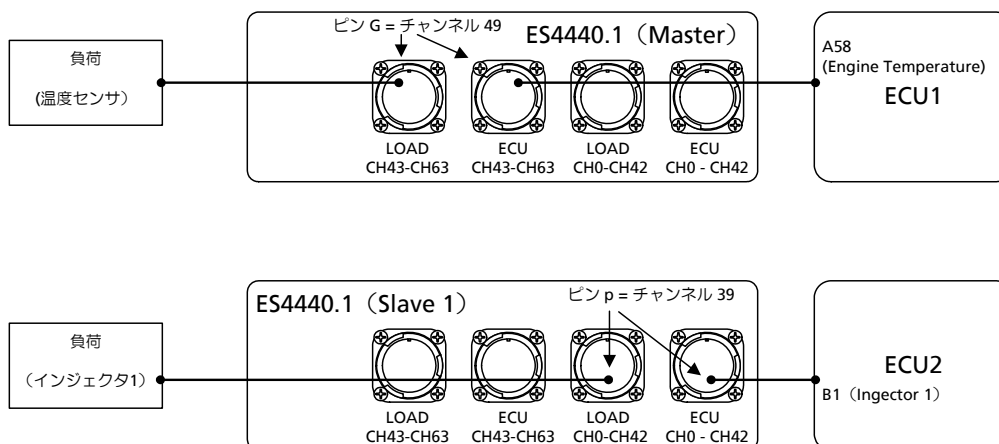
ワイヤハーネスファイルは、プロジェクトの中核となる重要なファイルです。このファイルには、ES4440.1/2 コンパクト故障シミュレーションモジュールの各チャンネルに接続される ECU シグナルについての情報が設定されます。

つまりこのファイルは、各 ECU チャンネルについて以下のデータが定義されたものです。

- ECU Name (ECU の名前)
- Pin Number (ピン番号)
- ES4440 Name (ES4440 の名前)
- Channel Number (チャンネル番号)

上記以外に Pin Name (ピンの名前) も定義されますが、これはピン番号に対する注釈なので、一意である必要はありません。

以下の図に示される配線例を用いて、ワイヤハーネスファイルの作成方法を説明します。



この例では、2 台の ECU が 2 台の ES4440.1/2 に接続されています。

- ECU Name : ECU1/ECU2
- Pin Number : A58/B1
- Pin Name : Engine Temperature / Injector 1
- ES4440 Name : Master / Slave 1
- ES4440 Connector : ECU/LOAD CH43-CH63 / ECU/LOAD CH0-CH42
- ES4440 Pin : G (チャンネル 49) / p (チャンネル 39)

2.2.1 ワイヤハーネスファイルの作成

本項では、製品に付属しているサンプルファイルを使用して、各テーブル（Excel シート）内に保存されたワイヤハーネスデータをもとに LABCAR-PINCONTROL V2.1 用プロジェクトのワイヤハーネスファイルを作成する方法を紹介します。

まず始めに、C:\Documents and Settings\All Users\Application Data\ETAS\LABCAR-PINCONTROL\2.1\Excel\LABCAR-PINCONTROLV2.0_Example.xls というサンプルファイルを開いてください。

この Excel ファイルは以下の 3 つのテーブル（Excel シート）で構成されています。

- **WireHarnessData**
ECU のポートと ES4440 のポートとのマッピングデータを定義します。
- **ES4440WireHarnessSignals**
“WireHarnessData” テーブルに定義されたデータを評価するための詳細な情報を定義します。
- **Execute_Example**
このテーブルからマクロを実行して、ES4440 に単純な “Open Load” エラーを発生させることができます（サンプルプロジェクトでのみ実行可能です）。

“WireHarnessData” テーブル

このテーブルに、ECU のポートと ES4440 のポートの割り当てを定義します。

	A	B	C	D	E	F	G	H	I
1	ECU Name	Pin Number	Pin Name	ES4440 Name	ES4440 Connector	ES4440 Pin			
2	ECU1	A1	Signal A1	Standalone	ECU30V_1	A			
3	ECU1	A2	Signal A2	Standalone	ECU30V_1	B			
4	ECU1	A3	Signal A3	Standalone	ECU30V_1	C			
5									
6									
7									
8									
9									
10									

各列に以下の情報を設定します。

- ECU Name**
 ECU の名前
 (一意の名前を使用してください。)
- Pin Number**
 ECU のピン番号
 (一意の名前を使用してください。数字以外の文字も使用できます。)
- Pin Name**
 接続されるシグナルの名前などの、ECU のピンについての詳細な説明
 (一意である必要はありません。)
- ES4440 Name**
 ECU のピンに接続される ES4440 の名前
 (“ES4440WireHarnessSignals” テーブルの “ES4440 Device Names” に定義されているデバイス名から選択します。)
- ES4440 Connector**
 ECU シグナルを接続する ES4440 のコネクタ
 (“ES4440WireHarnessSignals” テーブルの “Connector/Pin/Channel/Electric Type” に定義されているコネクタ名から選択します。)
- ES4440 Pin**
 ECU シグナルを接続する ES4440 のピンを定義します。
 (“ES4440WireHarnessSignals” テーブルの “Connector/Pin/Channel/Electric Type” に定義されているピン名から選択します。)

上記の項目を定義すると、このテーブルの各行は以下のマッピングを表すことになります。

<ECU Name> + <Pin Number> (+ <Pin Name>) =
 <ES4440 Name> + <ES4440 Connector> + <ES4440 Pin>

この式に示されるように、“Pin Name” は “Pin Number” のコメントとして使用されるだけなので、必ずしも一意である必要はありません。

ECU のすべてのピンを ES4440.1/2 に接続したら、ワイヤハーネスファイルのシグナル定義はすべて完了です。

最後に、定義したデータを LABCAR-PINCONTROL V2.1 用の XML ファイルに変換するための設定を行います。

データの調整

データを XML ファイルに変換するための設定を行うためには、まず “ES4440WireHarnessSignals” テーブルを選択します。このテーブルには、マクロを使用して定義済みワイヤハーネスデータから XML ファイルを生成する際に必要なデータが含まれています。

LABCAR-PINCONTROL V2.0 XML CREATION SETTINGS					
General Settings					
TableName	WireHarnessData		XML		
StartRow	2		full Path c:\example.xml		
EndRow	4				
Wire Harness Settings			Connector	Pin	Channel
ECU Name	A		ECU30V 1	A	0 HC
Pin Number	B		ECU30V 1	B	1 HC
Pin Name	C		ECU30V 1	C	2 HC
ES4440 Name	D		ECU30V 1	D	3 HC
ES4440 Connector	E		ECU30V 1	E	4 HC
ES4440 Pin	F		ECU30V 1	F	5 HC
			ECU30V 1	G	6 HC
			ECU30V 1	H	7 HC
			ECU30V 1	J	8 HC
			ECU30V 1	K	9 HC
			ECU30V 1	L	10 HC
			ECU30V 1	M	11 HC
			ECU30V 1	N	12 HC
			ES4440 Device Names		
			Standalone		
			Master		
			Slave<n>		

このテーブルで以下の情報を設定します。

General Settings:

- **TableName**

“ECU Name”、“Pin Number”、“Pin Name”、“ES4440 Name”、“ES4440 Connector”、“ES4440 Pin” のデータが含まれているテーブルの名前

- **Start Row**

“WireHarnessData” テーブルの評価開始行

- **End Row**

“WireHarnessData” テーブルの評価終了行

Wire Harness Settings:

“WireHarnessData” テーブル内で以下の各情報が定義される列を指定します。

- **ECU Name**

ECU の名前

- **Pin Number**

ECU のピン番号

- **ECU Pin Name**

ECU のピン名

- **ES4440 Name**

ES4440.1/2 の名前

- “ES4440 Device Names” グループ（下記参照）に表示されている名前のいずれかが使用されます。

- **ES4440 Connector**

ES4440.1/2 のコネクタ

- “Connector/Pin/Channel/Electric Type” グループ（下記参照）に定義されているコネクタのいずれかが使用されます。

- **ES4440 Pin**

ES4440.1/2 のコネクタ内のピン

- “ES4440 Pin” はいずれかの “ES4440 Connector” に属します。
“Connector/Pin/Channel/Electric Type” グループに定義されているピンのいずれかが使用されます。

XML:

- **Full Path**

マクロで生成するワイヤハーネスファイルの名前とパス

ES4440 Device Names:

使用するすべての ES4440.1/2 の名前

Connector/Pin/Channel/Electric Type:

ES4440.1/2 のすべてのポートについての定義

注記

背景色が灰色のフィールドの内容は変更しないでください。変更すると、生成されるファイルに不具合が生じたり、ファイルが生成されなくなる可能性があります。

ワイヤハーネスファイルの作成

すべての設定が終了したら、以下のようにしてワイヤハーネスファイルを生成します。

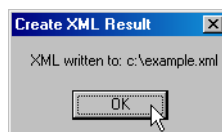
ワイヤハーネスファイルを作成する：

- “Execute_Example” テーブルを選択します。
- **Create WireHarnessFile for ES4440** ボタンをクリックします。

または

- ツール → マクロ → マクロ ... を選択します。
- “CreateXML” というマクロを選択し、**実行** をクリックします。

マクロが正常に実行されると、以下のダイアログボックスが開きます。



2.3 LABCAR-PINCONTROL V2.1 の操作

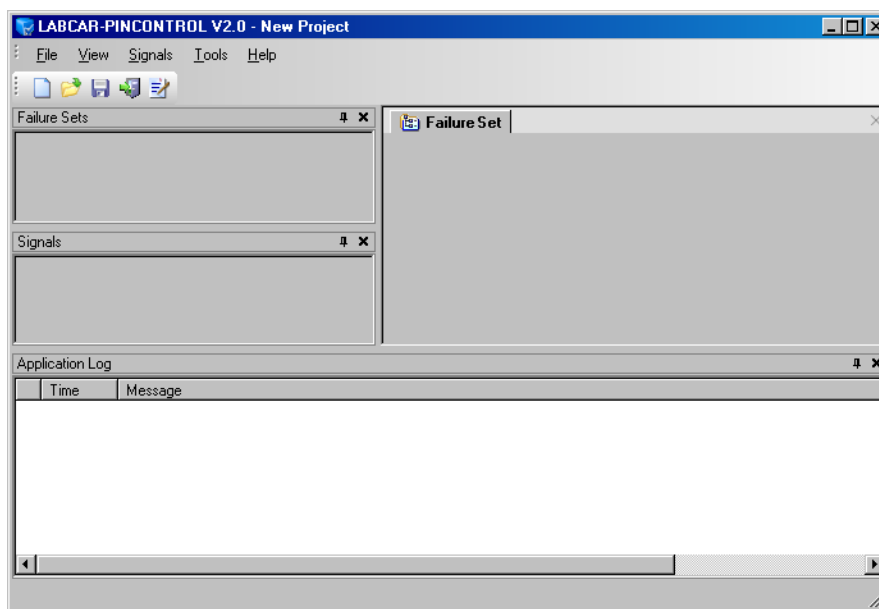
本項では、LABCAR-PINCONTROL V2.1 のユーザーインターフェースから ES4440.1/2 コンパクト故障シミュレーションモジュールを操作する方法を説明します。

- 「LABCAR-PINCONTROL V2.1 を起動する：」（17 ページ）
- 「ワイヤハーネスファイルからシグナルをインポートする：」（18 ページ）
- 「新しい故障セットを作成する：」（19 ページ）
- 「シグナルを追加する：」（20 ページ）
- 「シグナルを削除する：」（21 ページ）
- 「マルチエラーをシミュレートする：」（22 ページ）
- 「接点不良をシミュレートする：」（23 ページ）
- 「接触抵抗をシミュレートする：」（24 ページ）
- 「電流を測定する：」（25 ページ）
- 「リレー接点をクリーニングする：」（26 ページ）
- 「ヒューズをテストする：」（27 ページ）
- 「セルフテストを実行する：」（27 ページ）
- 「表示オプションを設定する：」（27 ページ）

LABCAR-PINCONTROL V2.1 を起動する：

- Windows のスタートメニューから、プログラム → ETAS → LABCAR-PINCONTROL V2.1 → PinControl.exe を選択します。

LABCAR-PINCONTROL V2.1 のウィンドウが開きます。



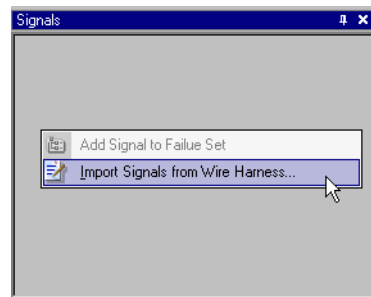
LABCAR-PINCONTROL V2.1 のユーザーインターフェースは以下のセクションに分かれています。

- Signals
プロジェクト用のワイヤハーネスファイルに定義されたシグナルの一覧が表示されます（18 ページの「ワイヤハーネスファイルからシグナルをインポートする：」を参照してください）。
- Failure Sets
シグナルが割り当てられた「故障セット」（“Failure Set”）の一覧が表示されます（19 ページの「新しい故障セットを作成する：」を参照してください）。
- Failure Set
このセクションには、さまざまなタイプのエラーが分類されたタブが含まれます。故障セットを選択し、シグナルをこのセクションに割り当てることにより、故障セットのシグナル構成が定義されます。いずれかのタブ上で、シミュレートしたいエラーを設定し、最後にボタンをクリックしてエラーシミュレーションを実行します（22 ページの「マルチエラーをシミュレートする：」を参照してください）。
- Application Log
処理ステータスやシステムメッセージが表示されます。

まず始めに、ワイヤハーネスファイルに保存されているシグナルをインポートします。

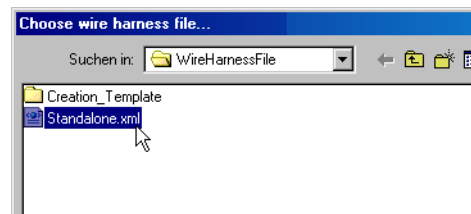
ワイヤハーネスファイルからシグナルをインポートする：

- “Signals” ウィンドウを右クリックします。
- ショートカットメニューから、**Import Signals from Wire Harness...** を選択します。

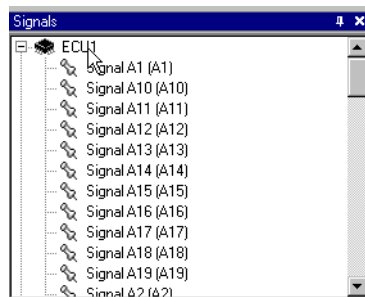


ファイル選択ウィンドウが開きます。

- ワイヤハーネス仕様が定義されている XML ファイルを選択します。ここでは、下図に示されるサンプルファイル Standalone.xml を使用します。



シグナルがインポートされ、「Signals」ウィンドウに表示されます。



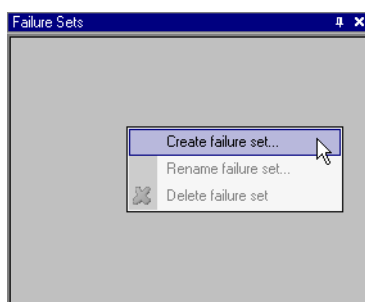
- **Save as** コマンドでプロジェクトを保存します。

これで、ECU と ES4440.1/2 コンパクト故障シミュレーションモジュールの接続に関する情報がすべてプロジェクトにインポートされました。

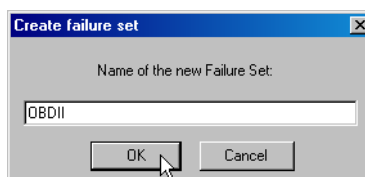
次に、1つ（または複数）の故障セット（“Failure Set”）を作成し、実際に使用するシグナルを定義します。

新しい故障セットを作成する：

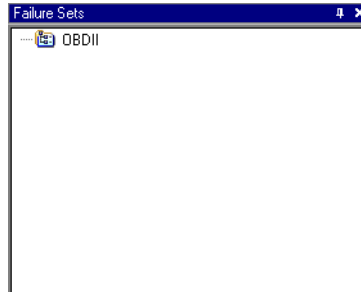
- “Failure Sets” ウィンドウを右クリックします。
- ショートカットメニューから **Create failure set...** を選択します。



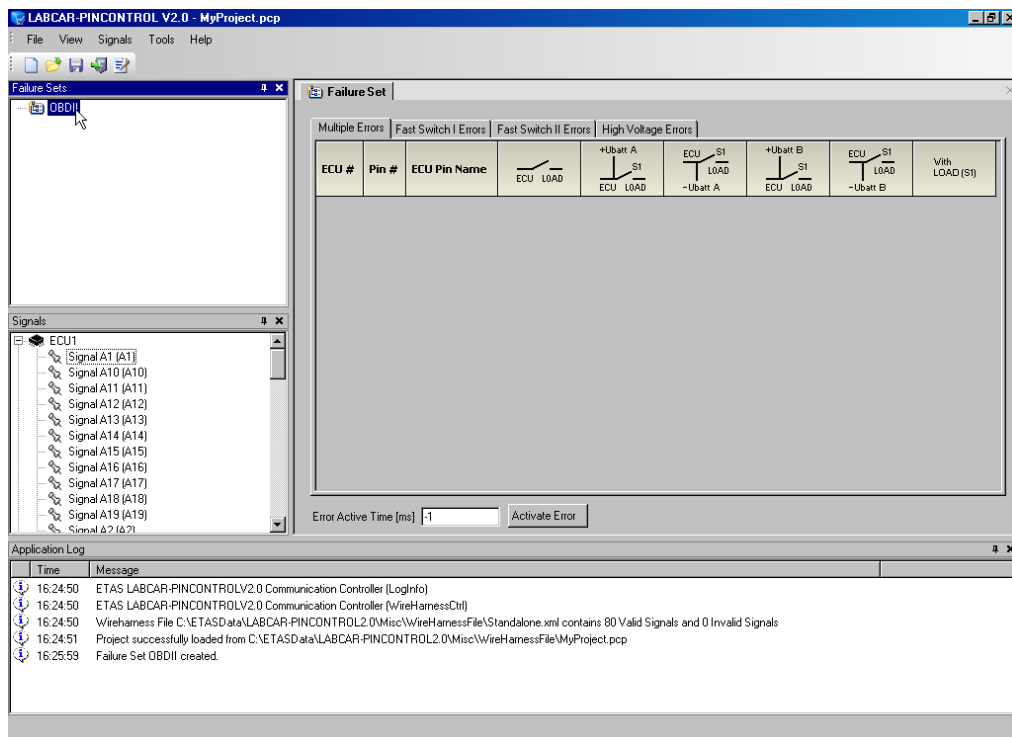
- 作成する故障セットの名前を、下図の例のように入力します。



- **OK** をクリックします。
上図のように入力した場合、“OBDII” という故障セットが作成されます。



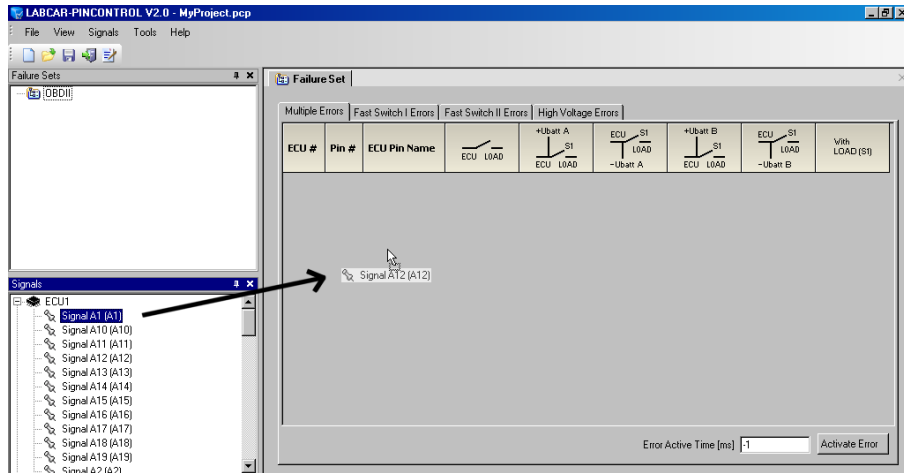
- 新しく作成された故障セットをクリックします。
シミュレートできるエラーが、ユーザーインターフェースの右側の各タブに分類されて表示されます。



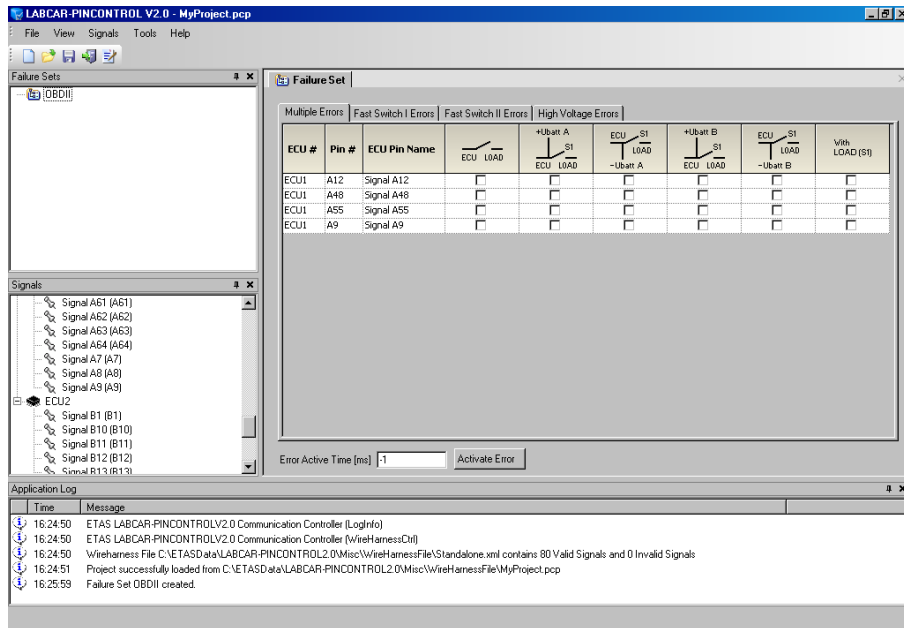
次に、シグナルを割り当てて故障セット“OBDII”を完成させます。

シグナルを追加する：

- “Signals” ウィンドウからシグナルを選択します。
- シグナルをタブ上にドラッグ（マウスでクリックし、ボタンを押したまま移動）します。この際、どのタブが選択されていてもかまいません。



- 同じ方法で、必要なシグナルをすべて故障セットに割り当てます。









- プロジェクトを保存します。
これで、故障セットが定義されたプロジェクトが完成しました。
同じプロジェクト内に複数の故障セットが必要な場合、上記の操作を繰り返して故障セットを追加します。

故障セットに割り当てたシグナルを削除するには、以下のように操作します。

シグナルを削除する：

- 不要なシグナルを右クリックします。

- **Remove Signal** を選択します。








Multiple Errors			Fast Switch I Errors		Fast Switch II Errors		High Voltage Errors	
ECU #	Pin #	ECU Pin Name						
ECU1	A12	Signal A12	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ECU1	A48	Signal A48	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ECU1	A55	Signal A55	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ECU1	A9	Signal A9	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ECU1	A64	Signal A64	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

そのシグナルが故障セットから削除されます。

つまり故障セット内の複数のシグナルについて、異なる種類のエラー¹を同時に発生させるには、以下のように操作します。このようにして発生させるエラーは「マルチエラー」と呼ばれます。

マルチエラーをシミュレートする：







- ここでは例として、シグナル “Signal A12”、“Signal A48”、“Signal A64” について、断線をシミュレートする “Open Load” 列のチェックボックスをオンにします。
- シグナル “Signal A55” には +UBatt_A への短絡 “Short against +UBatt_A” を選択します。また、エラー発生時に負荷が接続されたままになるように、“with LOAD (S1)” をオンにします。
- さらに、シグナル “Signal A9” についても +UBatt_A への短絡を選択します。

Failure Set									
Multiple Errors			Fast Switch I Errors		Fast Switch II Errors		High Voltage Errors		
ECU #	Pin #	ECU Pin Name							
ECU1	A12	Signal A12	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ECU1	A48	Signal A48	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ECU1	A55	Signal A55	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
ECU1	A9	Signal A9	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ECU1	A64	Signal A64	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Error Active Time [ms] 40

¹ 各種エラーについての詳細は、ES4440.1/2 コンパクト故障シミュレーションモジュールのユーザーズガイドを参照してください。

- このタブの最下部の “Error Active Time” フィールドに、エラーの持続時間（エラーをアクティブ状態にしておく時間）を入力します。
“-1” を入力すると、**Activate Error** が再度クリックされるまでエラー状態が持続されます。
- **Activate Error** をクリックします。
エラーシミュレーションが実行され、エラーがアクティブ状態になります。エラーがアクティブである間は、タブ上のシグナルの背景色が緑色になります。

Failure Set									
Multiple Errors			Fast Switch I Errors	Fast Switch II Errors	High Voltage Errors				
ECU #	Pin #	ECU Pin Name	 ECU LOAD	 +Ubatt A	 ECU S1 LOAD	 ECU S1 LOAD	 +Ubatt B	 ECU S1 LOAD	With LOAD (S1)
ECU1	A12	Signal A12	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ECU1	A48	Signal A48	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ECU1	A55	Signal A55	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ECU1	A9	Signal A9	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ECU1	A54	Signal A54	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

“Fast Switch I Errors” タブと “Fast Switch II Errors” タブでは、接点不良によるシグナルのチャタリングをシミュレートすることもできます。この場合、指定された持続時間にわたってエラーがアクティブのままになるのではなく、所定の周波数とデューティサイクルでアクティブ/非アクティブが切り替わります。

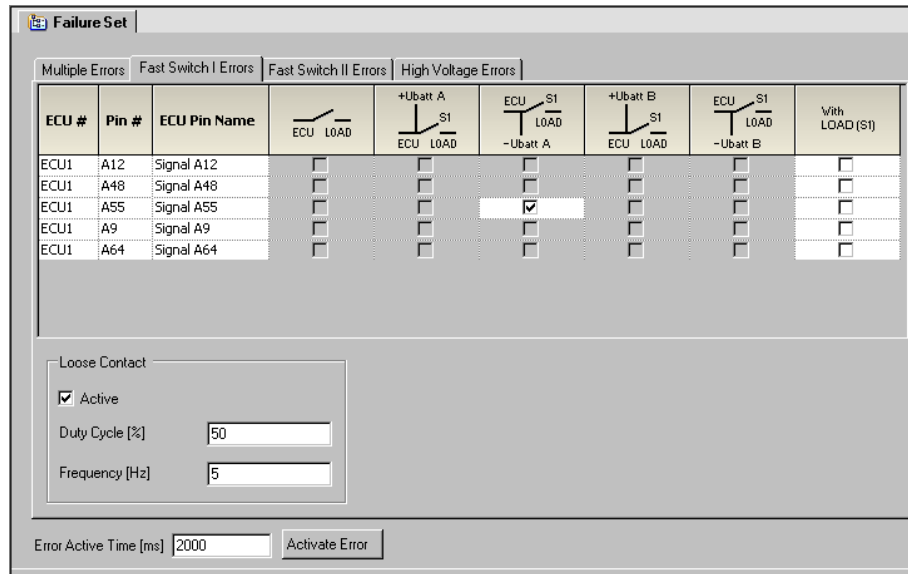
接点不良をシミュレートする：

- ここでは例として “Fast Switch I Errors” タブに切り替えます。

注記

タブを切り替えると、切り替える前のタブ上でのエラー設定は解除されます。

- エラーを選択します。
- タブの下部にある “Loose Contact” フィールドの “Activate” オプションをオンにします。
- “Duty Cycle” フィールドにデューティサイクル [%] を入力します。
- “Frequency” フィールドに周波数 [Hz] を入力します。



- “Error Active Time” フィールドに接点不良シミュレーションの持続時間 [ms] を入力します。
- **Activate Error** をクリックしてエラーシミュレーションを実行します。

接触抵抗のエラー（ライン抵抗、ピン間抵抗、バッテリー電圧へのリーク電流）は、“Fast Switch II Errors” タブを使用します。

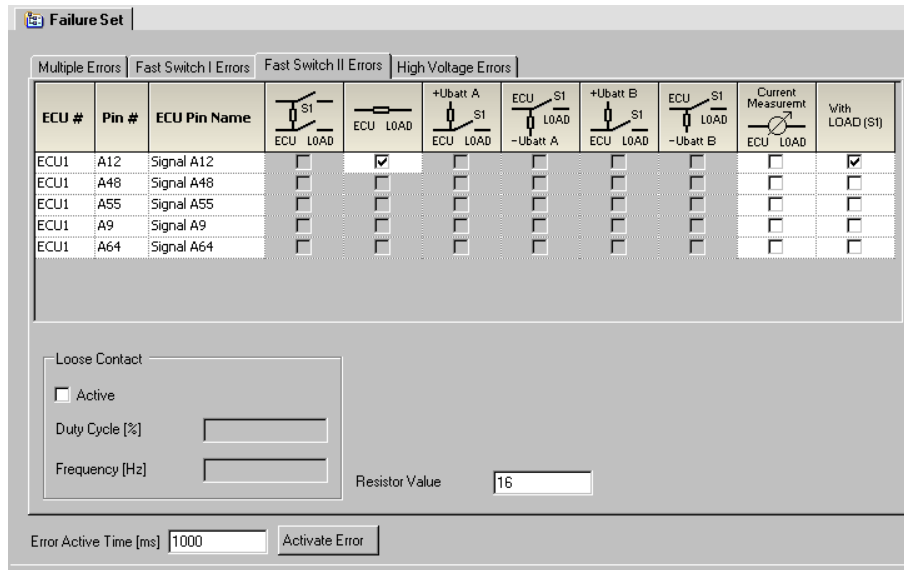
接触抵抗をシミュレートする：

- “Fast Switch II Errors” タブでエラーを選択します（図の例ではライン抵抗）。

注記

ピン間エラーをシミュレートする場合は、2つのシグナルを割り当てる必要があります。

- “Resistor Value” フィールドに抵抗値 [Ohm] を入力します。

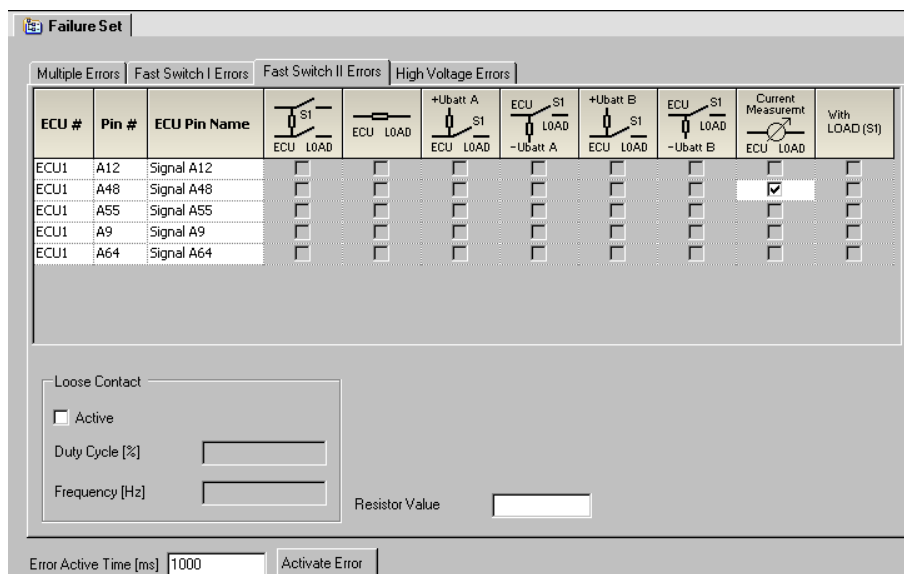


- “Error Active Time” フィールドにエラーシミュレーションの持続時間 [ms] を入力します。
- **Activate Error** をクリックしてエラーシミュレーションを実行します。

シグナル線を通る電流を測定する必要がある場合、“Fast Switch II Errors” タブで、シグナルをフロントパネルの “Current” コネクタにルーティングすることができます。

電流を測定する：

- シグナルを選択します。
- “Error Active Time” フィールドに測定時間 [ms] を入力します。



- **Activate Error** をクリックしてエラーシミュレーションを実行します。

- 選択されたシグナルが ES4440.1/2 コンパクト故障シミュレーションモジュールのフロントパネルの "Current" ポートに接続されるので、このコネクタからチャンネルの電流を測定することができます。

酸化したリレー接点のクリーニングを行うには、以下のように操作します。

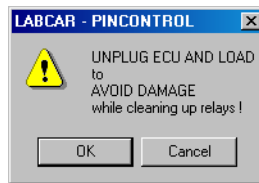
リレー接点をクリーニングする：

- **Tools → ES4440 System Configuration** を選択します。

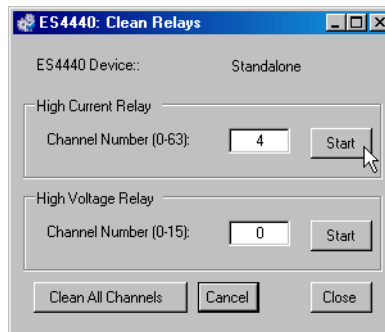
"ES4440 System Configuration" ダイアログボックスが開きます。

- "Devices found" フィールドから、クリーニングを行うモジュールを選択します。
- "Device Tests" フィールドの **Clean Up Relays** をクリックします。

「クリーニングを行う前には必ずすべての "LOAD" コネクタと "ECU" コネクタからケーブルを取り外してください」という旨のワーニングメッセージが表示されるので、その指示に従ってください。



- 次のダイアログボックスで、個々のチャンネルのリレー、またはすべてのリレーをクリーニングすることができます。



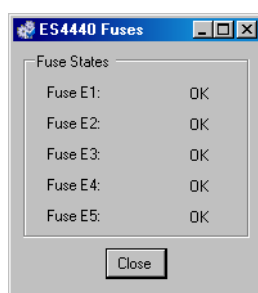
注記

1回のクリーニングにおいて10回のスイッチングが行われるので、すべてのリレーをクリーニングするには数分かかる可能性があります。

エラーレール上の 5 個のヒューズの状態をテストするには、以下のように操作してください。

ヒューズをテストする：

- **Tools** → **ES4440 System Configuration** を選択します。
“ES4440 System Configuration” ダイアログボックスが開きます。
- “Devices found” フィールドから、ヒューズテストを行うモジュールを選択します。
- “Device Tests” フィールドの **Check Fuses** をクリックします。
ヒューズテストが実行され、結果が表示されます。



注記

デバイス内のヒューズの位置と仕様については、ES4440.1/2 コンパクト故障シミュレーションモジュールのマニュアルを参照してください。

ES4440.1/2 に内蔵されている PLD、CAN コントローラ、EEPROM の状態を、セルフテストによってチェックすることができます。セルフテストを実行するには、以下のように操作してください。

セルフテストを実行する：

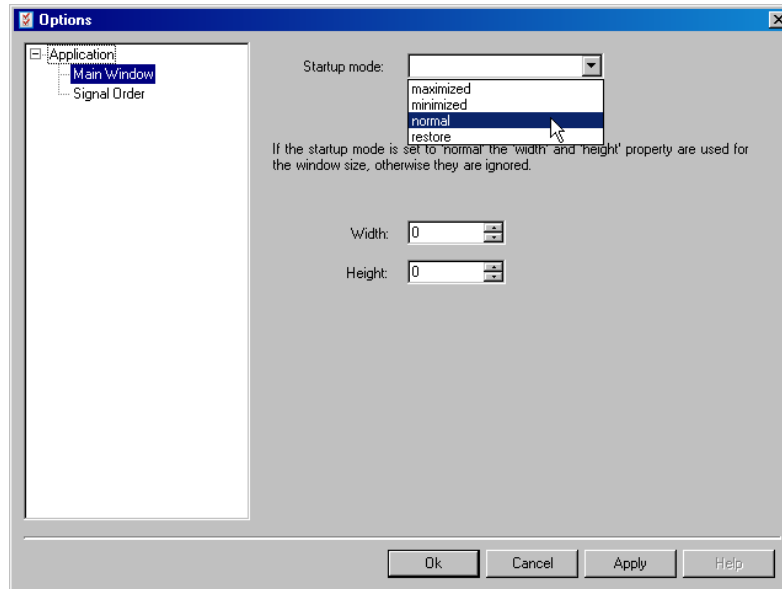
- **Tools** → **ES4440 System Configuration** を選択します。
“ES4440 System Configuration” ダイアログボックスが開きます。
- “Devices found” フィールドから、以下のアクションの対象にするモジュールを選択します。
- “Device Tests” フィールドの **Self Test** をクリックします。
セルフテストが実行され、結果がログウィンドウに表示されます。

LABCAR-PINCONTROL V2.1 起動時のユーザーインターフェースの表示スタイルや “Signals” ウィンドウ内のシグナル表示順序は、以下のようにして変更することができます。

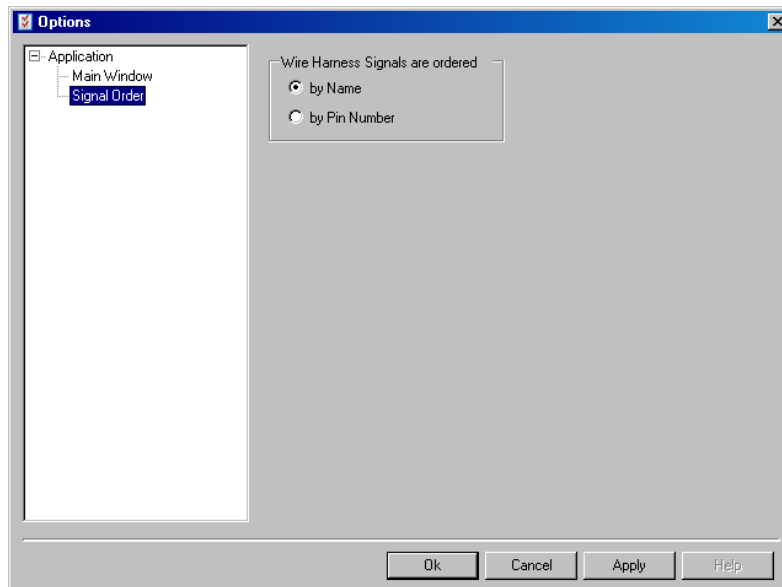
表示オプションを設定する：

- **Tools** □ **Options...** を選択します。

- “Options” ウィンドウの左側のリストボックスから “Main Window” オプションを選択します。
- “Startup Mode” フィールドから、メインウィンドウの表示スタイルを選択します。



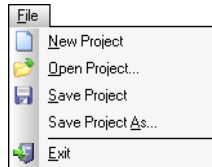
- “normal” を選択した場合は、ウィンドウの幅と高さをそれぞれ “Width” と “Height” に設定します。
- “Signal Order” オプションでは、“Signals” ウィンドウに表示されるシグナルの表示順（シグナル名、またはピン番号の順）を指定します。



2.4 メインメニュー

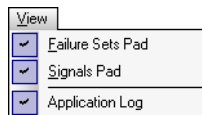
本項では、PINCONTROL メインメニューについて説明します。

“File” メニュー



- **File** → **New Project**
新しいプロジェクトを作成します。
- **File** → **Open Project...**
保存されているプロジェクトを開きます。
- **File** → **Save Project**
現在ロードされているプロジェクトを保存します。ES4440.1/1.2 の設定データ（システム設定とイーサネット設定）や故障セットのデータ（ECU 名、ピン名、ピン番号、）が保存されます。
- **File** → **Save Project As...**
現在ロードされているプロジェクトを新しい名前でも保存します。
- **File** → **Exit**
LABCAR-PINCONTROL V2.1 を終了します。

“View” メニュー



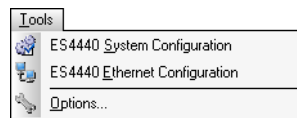
- **View** → **Failure Sets Pad**
プロジェクトの故障セットが表示されるセクションの表示／非表示を切り替えます。
- **View** → **Signals Pad**
プロジェクトにインポートされたワイヤハーネスファイルのシグナルが表示されるセクションの表示／非表示を切り替えます。
- **View** → **Application Log**
ログウィンドウの表示／非表示を切り替えます。

“Signals” メニュー



- **Signals** → **Import Signals from Wire Harness...**
シグナルデータをワイヤハーネスファイルからインポートします。

"Tools" メニュー



- **Tools** → **ES4440 System Configuration**
ES4440.1/2 のシステム設定を行うためのダイアログボックスを開きます。
- **Tools** → **ES4440 Ethernet Configuration**
ES4440.1/2 のイーサネットインターフェースの設定を行うためのダイアログボックスを開きます。
- **Tools** → **Options...**
表示オプションを定義するためのダイアログボックスを開きます。

"Help" メニュー

- **Help** → **About**
バージョン情報と ETAS の連絡先アドレスを表示するウィンドウを開きます。

3 API

本章では、LABCAR-PINCONTROL V2.1 の COM コントローラまたは CAN 経由で ES4440.1/2 コンパクト故障シミュレーションモジュールを自動的に操作する方法について説明します。

本章は以下の項に分かれています。

- 「概要」(31 ページ)
- 「CAN メッセージの設定とシーケンス」(32 ページ)
- 「初期設定」(38 ページ)
- 「コマンドの詳細説明」(38 ページ)

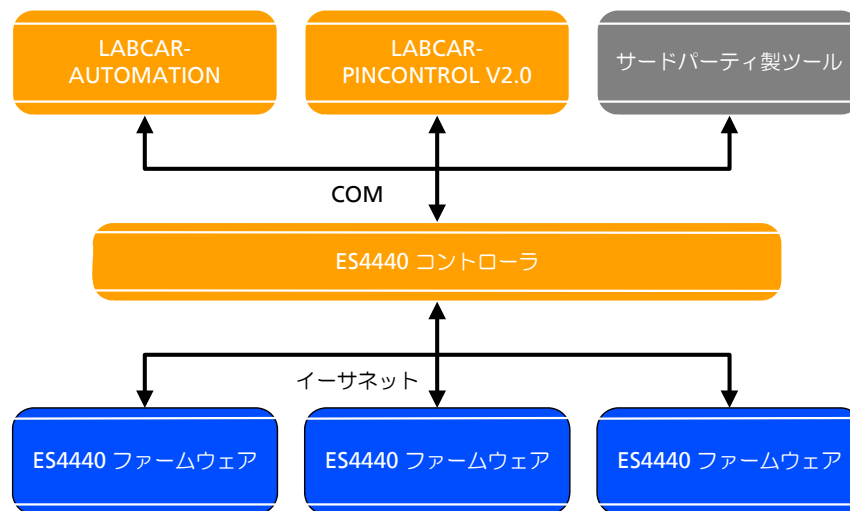
3.1 概要

ES4440.1/2 コンパクト故障シミュレーションモジュールの自動操作は、イーサネットまたは CAN 経由で行えます。イーサネットプロトコルは付属の COM コントローラでサポートされています。CAN プロトコルは、リアルタイムシミュレーションターゲットを CAN ボード経由で 1 台または複数の ES4440.1/2 に接続する場合に使用されます。

本章には、COM コントローラのメソッド呼び出しと CAN メッセージのデータ構造が定義されています。

3.1.1 COM コントローラの機能

下図は、システム全体の中での COM コントローラの働きを示しています。



COM コントローラの入力値は、ECU の名前とピンの名前です。COM コントローラは、ワイヤハーネスファイルを用いて、これらの名前に ES4440.1/2 とピン番号を割り当てます。

さらに、COM コントローラは COM メソッドをイーサネットプロトコルに転送します。また、複数の ES4440 をマスタ/スレーブ構成で使用する場合、COM コントローラはコマンドを各モジュールに振り分ける役割も果たします。

3.1.2 CAN-API の使用

CAN メッセージは、各 ES4440 モジュールに直接送信されます。ES4440 モジュールが相互に通信することはできないので、ユーザーは以下のことを確実に行う必要があります。

- ECU の名前およびピンの、ES4440 モジュールへのマッピング
- マスタ/スレーブ操作の場合は、ES4440 モジュールへのコマンド配信

3.1.3 COM-API および CAN-API のデータ内容

COM コントローラへの入力データとして ECU の名前とピン名が必要ですが、CAN-API の場合は ES4440 モジュールのピン番号が必要です。この点を除けば、COM-API と CAN-API のデータ内容はまったく同じです。

3.2 CAN メッセージの設定とシーケンス

本項では、各 ES4440 モジュールへの CAN メッセージ配信について説明します。ES4440 でさまざまなタイプのエラーをミュレートする際のコマンドシーケンスを紹介し、さらに複数の ES4440 モジュールに CAN メッセージを送信する手順についても紹介します。

エラーシミュレーションの実行には、一般に、以下のルールが適用されます。

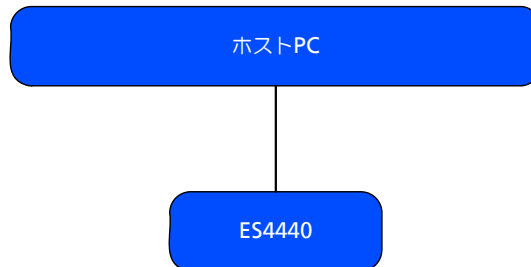
- リレー（メカニカルリレー）を用いてシミュレートされるエラーの場合、エラーのアクティブ化は常にマスタモジュールにより行われます。このため、全 ES4440 モジュールにおいてエラーのオン/オフ切り替えが確実に同期して行われます。
- MOSFET（半導体リレー）を用いるエラーの場合は、そのエラーをシミュレートするモジュール自身によってエラーがアクティブ化されます。
- エラーを発生させるコマンドを ES4440 モジュールに送信した後は、エラーのリセットコマンドを発行してエラー状態を解除する必要があります。
- ピン間エラーは例外的なエラーなので、他のエラーとは別に扱われます。

注記

マスタ/スレーブ構成について紹介しているシナリオの内容は、一例です。ここで "Slave 1" 上で実行されるものとして示されているコマンドは、実際にはマスタや他のスレーブでも同じように実行できます。

3.2.1 スタンドアロン構成におけるシングルエラーのシミュレーション

まず始めに、1 台の ES4440.1/2 コンパクト故障シミュレーションモジュール上でシングルエラー（単一のエラー）を発生させるためコマンドシーケンスの例を紹介します。



リレーを用いたエラーを発生させるシーケンス

1. エラー設定

高電流エラー：

以下のいずれかのコマンドを発行します。

- Open_Load
- ShortCut_xUBATTy_20A
- Pin2PinFirstChWithoutLoad /
Pin2PinSecondChWithoutLoad

高電圧エラー：

以下のいずれかのコマンドを発行します。

- Open_Load_400V
- ShortCut_xUBATTy_400V
- Pin_2_Pin_400V

2. エラーのアクティブ化

- Activate_relay

3. エラーのリセット

- Reset_all_errors

MOSFET を用いたエラーを発生させるシーケンス

1. エラー設定

以下のいずれかのコマンドを発行します。

- Open_Load_realtime
- ShortCut_xUBATTy_20A_realtime
- Pin2PinFirstChRelatimeWithLoad
- Pin2PinSecondChRealtimeWithLoad
- RInline_realtime
- Pullup_Pulldown_xUBATTy_20A_realtime

2. エラーのアクティブ化

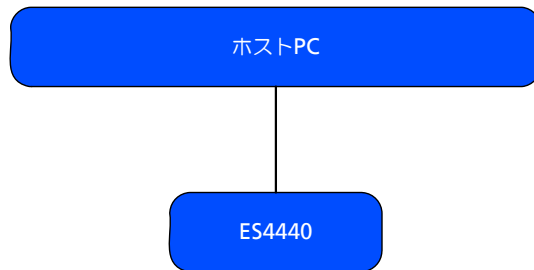
- Activate_realtime_switch

3. エラーのリセット

- Reset_all_errors

3.2.2 スタンドアロン構成におけるマルチエラーのシミュレーション

1 台の ES4440.1/2 コンパクト故障シミュレーションモジュール上でマルチエラー（複数のエラー）を発生させるためのコマンドシーケンスの例をあげます。



リレーを用いたエラーを発生させるシーケンス

1. エラー設定

以下のコマンド（高電流エラーの設定）を、任意に組み合わせて最大 10 個まで ES4440.1/2 に送信できます。

- Open_Load
- ShortCut_xUBATTy_20A

2. エラーのアクティブ化

- Activate_relay

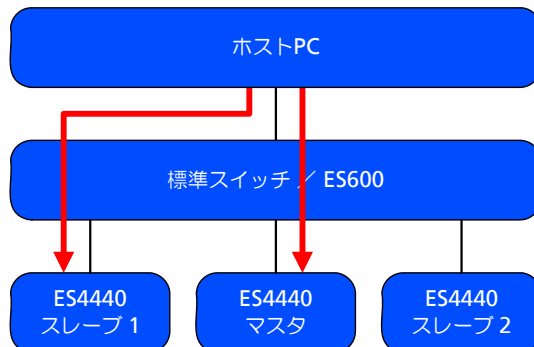
3. エラーのリセット

- Reset_all_errors

3.2.3 マスタ/スレーブ構成におけるシングルエラーのシミュレーション

マスタ/スレーブシステム上でシングルエラー（単一のエラー）をシミュレートするためのコマンドシーケンスの例を紹介します。

リレーを用いたエラーを発生させるシーケンス



1. エラー設定（スレーブ 1）

以下のコマンド（高電流エラーの設定）を、任意に組み合わせて最大 10 個まで ES4440.1/2 に送信できます。

- Open_Load
- ShortCut_xUBATTy_20A

2. エラーのアクティブ化（マスタ）

- Activate_relay

3. エラーのリセット (スレーブ1) *

- Reset_all_errors

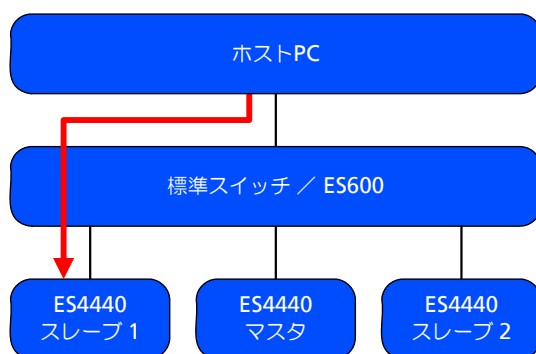
4. エラーのリセット (マスタ) **

- Reset_all_errors

* スレーブに “Reset_all_errors” コマンドを送信すると、スレーブ上ではコマンドのセーブのみが行われます。

** マスタへの “Reset_all_errors” コマンドにより、マスタ上のエラーと、リセットコマンドがセーブされているすべてのスレーブ上のエラーが、同期してリセットされます。

MOSFET を用いたエラーを発生させるシーケンス



1. エラー設定

以下のいずれかのコマンドを発行します。

- Open_Load_realtime
- ShortCut_xUBATTy_20A_realtime
- RInline_realtime
- Pullup_Pulldown_xUBATTy_20A_realtime

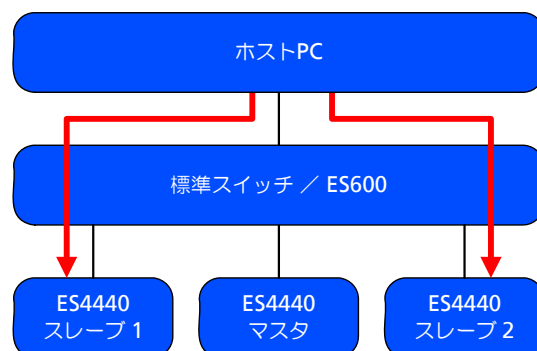
2. エラーのアクティブ化

- Activate_realtime_switch

3. エラーのリセット

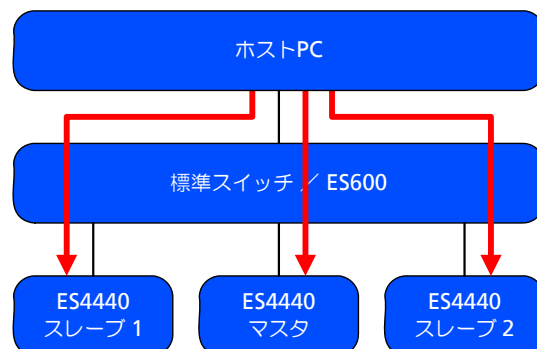
- Reset_all_errors

特殊なケース：ピン間エラー（負荷あり）



1. 第1のピンのエラー設定（スレーブ1）
 - － Pin2PinFirstChRelatimeWithLoad
2. 第2のピンのエラー設定（スレーブ2）
 - － Pin2PinSecondChRealttimeWithLoad
3. エラーのアクティブ化（スレーブ1）
 - － Activate_realttime_switch
4. エラーのリセット（スレーブ1）
 - － Reset_all_errors
5. エラーのリセット（スレーブ2）
 - － Reset_all_errors

特殊なケース：ピン間エラー（負荷なし）



1. 第1のピンのエラー設定（スレーブ1）
 - － Pin2PinFirstChWithoutLoad
2. 第2のピンのエラー設定（スレーブ2）
 - － Pin2PinSecondChWithoutLoad
3. エラーのアクティブ化（マスタ）
 - － Activate_relay
4. エラーのリセット（スレーブ1）
 - － Reset_all_errors
5. エラーのリセット（スレーブ2）
 - － Reset_all_errors

6. エラーのリセット (マスタ)

– Reset_all_errors

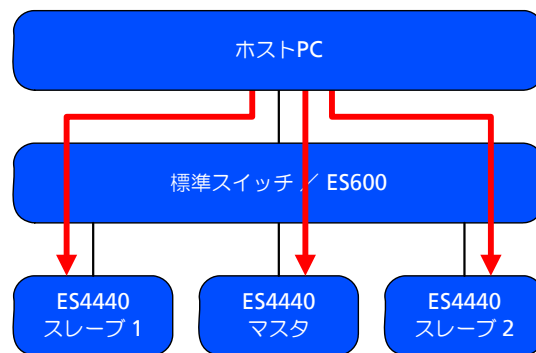
注記

ピン間エラーのオン/オフ切り替えはリレーにより行われます。第 1 のピンと第 2 のピンの間のエラー経路にはヒューズはありません。

3.2.4 マスタ/スレーブ構成におけるマルチエラーのシミュレーション

マスタ/スレーブシステム上でマルチエラー（複数のエラー）をシミュレートするためのコマンドシーケンスの例を紹介します。

リレーを用いたエラーを発生させるシーケンス



1. エラー設定 (スレーブ 1)

以下のコマンド（高電流エラーの設定）を、任意に組み合わせて最大 10 個まで ES4440.1/2 に送信できます。

– Open_Load
– ShortCut_xUBATTy_20A

2. エラー設定 (スレーブ 2)

スレーブ 1 と同様に高電流エラーを設定します。

– Open_Load
– ShortCut_xUBATTy_20A

3. エラーのアクティブ化 (マスタ)

– Activate_relay

4. エラーのリセット (スレーブ 1)

– Reset_all_errors

5. エラーのリセット (スレーブ 2)

– Reset_all_errors

6. エラーのリセット (マスタ)

– Reset_all_errors

3.3 初期設定

3.3.1 COM コントローラの初期設定

COM コントローラの初期設定は以下のように行います。

```
Dim returnValue As Integer
Dim WireharnessPath As String
'access CommCtrlAccess
Set ctrlA = CreateObject
    ("ETAS.PTS.PINCONTROLV2.CommCtrl.CommCtrlAccess")
'using CommCtrlAccess class you can try to access the
singleton instance
Set ctrl = ctrlA.CommCtrlInstanceinstance
Set ctrl = ctrlA.CommCtrlInstance
returnValue =
ctrl.InitErrorSimulationUsingFile(WireharnessPath)
```

“Exit” は以下のように行います。

```
'free CommCtrl Singleton instance
Call ctrlA.FreeCommCtrlInstance
```

注記

COM コントローラは、一度に 1 つのアプリケーションでしか使用できません。

3.3.2 CAN を使用する場合の初期設定

CAN については初期設定は必要ありません。

3.4 コマンドの詳細説明

本項では、以下の全コマンドについて、機能と構文を説明します。

- IDN コマンド (42 ページ)
- Open_Load (43 ページ)
- Open_Load_realtime (44 ページ)
- ShortCut_xUBATTy_20A (45 ページ)
- ShortCut_xUBATTy_20A_realtime (46 ページ)
- Pin2PinFirstChWithoutLoad (47 ページ)
- Pin2PinSecondChannelWithoutLoad (48 ページ)
- Pin2PinFirstChRealtimeWithLoad (49 ページ)
- Pin2PinSecondChRealtimeWithLoad (50 ページ)
- RInline_realtime (51 ページ)
- Pullup_Pulldown_xUBATTy_20A_realtime (53 ページ)
- Open_Load_400V (54 ページ)
- ShortCut_xUBATTy_400V (55 ページ)
- ShortCut_xUBATTy_400V_Ex (56 ページ)
- Pin_2_Pin_400V (57 ページ)

- Pin_2_Pin_400V_Ex (58 ページ)
- Reset_all_errors (59 ページ)
- Activate_relay (60 ページ)
- Activate_realtime_switch (62 ページ)
- TestFuses (64 ページ)
- CurrentMeasurement (65 ページ)

3.4.1 CAN コマンドの構造

本項の各コマンドの説明においては、CAN 送信メッセージと CAN 受信メッセージの構造が以下の形式で示されています。

CAN 送信メッセージ

COM コマンド (引数)		
第1バイト	コマンド ID	値
第2バイト	パラメータ 0	値
第3バイト	パラメータ 1	値
第4バイト	パラメータ 2	値
第5バイト	パラメータ 3	値
第6バイト	パラメータ 4	値
第7バイト	パラメータ 5	値
第8バイト	パラメータ 6	値

表 3-1 送信メッセージの構造

CAN 受信メッセージ

応答		
第1バイト	コマンド ID	値
第2バイト	パラメータ 0	値
第3バイト	パラメータ 1	値
第4バイト	パラメータ 2	値
第5バイト	パラメータ 3	値
第6バイト	パラメータ 4	値
第7バイト	パラメータ 5	値
第8バイト	パラメータ 6	値

表 3-2 受信メッセージの構造

注記

ES4440.1/2 でバッファオーバーフローが発生するのを防ぐため、CAN メッセージはシングルショットモードでのみ送信するようにしてください。

3.4.2 全コマンドに共通する情報

以下に全コマンドに共通する情報について説明します。

チャンネル番号

高電流チャンネルには 0 ~ 63 の番号が付き、高電圧チャンネルには 0 ~ 15 の番号が付きます。

CAN 送信メッセージのパラメータ1 の定義

パラメータ 1 (第 3 バイト) は以下のビット構成です。コマンドに応じて必要なビットを設定します。

ビット 0	load	1 : 負荷ありのエラーシミュレーション 0 : 負荷なしのエラーシミュレーション
ビット 1	xUBatty	0 : +UBatt_A
ビット 2		1 : -UBatt_A
ビット 3		2 : +UBatt_B
		3 : -UBatt_B
		4 : +UBatt_C
		5 : -UBatt_C
ビット 4	current	0 : 電流測定オフ 1 : 電流測定オン
ビット 5	set	1 : エラーセット (エラーをアクティブにする) 0 : エラーリセット
ビット 6	duration_flag	0 : エラー状態を無期限に (リセットされるまで) 持続します。 1 : エラーの持続時間は "duration_time" で指定されず。
ビット 7	不使用	

3.4.3 エラーコード

下記のエラーコードは受信メッセージの第 8 バイトで転送されます。また API コマンド `GetLastErrorSimulationAnswer()` の場合も第 8 バイトでエラーコードが転送されます。

結果	意味
0x0	コマンド OK
0x21	スレーブアドレス用パラメータが不正 (> 16)
0x22	未定義コマンド
0x23	フラッシュ書き込み時に不正なデータ型を検出
0x24	LED テスト用パラメータが不正
0x25	IP アドレスの値が不正 (正しい値 : < 256)
0x26	CAN ボーレート用パラメータが不正
0x27	CAN ターミネータ用パラメータが不正
0x28	CAN 識別子タイプ用パラメータが不正
0x29	カスケードチャンネル用パラメータの値が大きすぎる (正しい値 : < 15)
0x2a	レジスタカスケード用パラメータが不正
0x2b	(未使用)

結果	意味
0x2c	フラッシュリードアクセスのアドレスが不正 (正しいアドレス: < 513)
0x2d	フラッシュリードアクセスのデータ長が不正 (正しいデータ長: < 17)
0x2e	フラッシュライトアクセスのアドレスが不正 (正しいアドレス: < 513)
0x2f	フラッシュライトアクセスのデータ長が不正 (正しいデータ長: < 17)
0x30	PLD エラー
0x31	EEPROM チェックサムエラー
0x32	CAN コントローラにアクセスできない
0x41	シミュレーションコマンドから妥当性エラーが返った
0x42	リファレンスリレーが検出できない
0x43	無期限のエラー発生が要求されているのに関わらず "duration_time" の値が "0xFFFF" でない
0x44	シミュレーションコマンドが認識できない
0x45	PLD エラー: コマンドが実行できない
0x46	"duration time" の値が有効範囲を超えている (正しい値: 1 ~ 5000, 0xFFFF)
0x47	別のエラーシミュレーションがまだアクティブになっているので、Reset_all_errors で終了させる必要がある
0x48	リレーの最大数に達している
0x49	マルチエラーフラグのエラー
0x4a	指定のチャンネル番号が有効範囲を超えている
0x4b	周波数または周期が有効範囲を超えている
0x4c	システム温度 > 60 °C
0x4d	レジスタカスケードの温度 > 60 °C
0x4e	MOSFET の温度 > 60 °C
0x4f	システム温度センサの故障
0x50	レジスタカスケード温度センサの故障
0x51	MOSFET 温度センサの故障
0x52	レール電圧異常 (短絡の可能性あり)
0x53	抵抗値が無効

3.4.4 IDN コマンド

ES4440.1/2 を識別するためのコマンドです。

COM コントローラへのコマンド送信

```
int ctrl.CommandIDN(string deviceKey);
```

“deviceKey” の値 : Standalone, Master, Slave1, ..., Slave14

COM コントローラからの応答の取得

```
byte[] Answer =  
    ctrl.GetLastSystemConfigurationAnswer();
```

CAN 送信メッセージ

IDN コマンド

第 1 バイト	コマンド ID	0x0
第 2 バイト	パラメータ 0	不使用
第 3 バイト	パラメータ 1	不使用
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	不使用

CAN 受信メッセージ

応答

第 1 バイト	コマンド ID	0x0
第 2 バイト	パラメータ 0	Device Config - バイト 1 *
第 3 バイト	パラメータ 1	Device Config - バイト 0 *
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	コマンドの実行結果

* Device Config” の値 :

- Standalone : 255
- Master : 0
- Slave 1 ~ Slave 14 : 1 ~ 14

3.4.5 Open_Load

ECU と負荷の間のラインを遮断するエラーを設定します。このエラーは、リレーによりオン/オフ切り替えが行われます。最大 10 個のエラーを同時にオン/オフすることができ、コマンド応答内の "channels left" の値によって、あといくつのチャンネルが使用可能であるかがわかります。

COM コントローラへのコマンド 送信

```
int ctrl.OpenLoad(string ecu, string ecuPin,
                  int durationType, int set);
```

COM コントローラからの応答の取得

```
byte [] Answer = ctrl.GetLastErrorSimulationAnswer();
```

CAN 送信メッセージ

Open_Load (channel_nr, duration_flag, set)

第 1 バイト	コマンド ID	0x1
第 2 バイト	パラメータ 0	ES4440 のチャンネル番号
第 3 バイト	パラメータ 1	set, duration_flag (40 ページ参照)
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	不使用

CAN 受信メッセージ

応答

第 1 バイト	コマンド ID	0x1
第 2 バイト	パラメータ 0	ES4440 のチャンネル番号
第 3 バイト	パラメータ 1	channels left
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	コマンドの実行結果

3.4.6 Open_Load_realtime

ECU と負荷の間のラインを遮断するエラーを設定します。このエラーは MOSFET によりオン/オフ切り替えが行われ、シングルエラーとしてのみシミュレートできません。

COM コントローラへのコマンド送信

```
int ctrl.OpenLoad_RealTime(string ecu,
                           string ecuPin, int durationType);
```

COM コントローラからの応答の取得

```
byte[] Answer = ctrl.GetLastErrorSimulationAnswer();
```

CAN 送信メッセージ

Open_Load_realtime (channel_nr, duration_flag)

第 1 バイト	コマンド ID	0x2
第 2 バイト	パラメータ 0	ES4440 のチャンネル番号
第 3 バイト	パラメータ 1	duration_flag (40 ページ参照)
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	不使用

CAN 受信メッセージ

応答

第 1 バイト	コマンド ID	0x2
第 2 バイト	パラメータ 0	ES4440 のチャンネル番号
第 3 バイト	パラメータ 1	不使用
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	コマンドの実行結果

3.4.7 ShortCut_xUBATTy_20A

高電流チャンネルについて、バッテリー電圧へのラインの短絡について設定します。このエラーは、リレーによりオン/オフ切り替えが行われ、複数のチャンネルで同時にシミュレートすることができます。

COM コントローラへのコマンド送信

```
int ctrl.Shortcut_xUBATTy_20A(string ecu,
                               string ecuPin, int load, int xUBATTy,
                               int durationType, int set);
```

COM コントローラからの応答の取得

```
byte[] Answer = ctrl.GetLastErrorSimulationAnswer();
```

CAN 送信メッセージ

ShortCut_xUBATTy_20A (channel_nr, load, xUBATTy, duration_flag, set)

第1バイト	コマンド ID	0x3
第2バイト	パラメータ 0	ES4440 のチャンネル番号
第3バイト	パラメータ 1	load, xUBatty, set, duration_flag (40 ページ参照)
第4バイト	パラメータ 2	不使用
第5バイト	パラメータ 3	不使用
第6バイト	パラメータ 4	不使用
第7バイト	パラメータ 5	不使用
第8バイト	パラメータ 6	不使用

CAN 受信メッセージ

応答

第1バイト	コマンド ID	0x3
第2バイト	パラメータ 0	ES4440 のチャンネル番号
第3バイト	パラメータ 1	channels left
第4バイト	パラメータ 2	不使用
第5バイト	パラメータ 3	不使用
第6バイト	パラメータ 4	不使用
第7バイト	パラメータ 5	不使用
第8バイト	パラメータ 6	コマンドの実行結果

3.4.8 ShortCut_xUBATTy_20A_realtime

高電流チャンネルについて、バッテリー電圧へのラインの短絡について設定します。このエラーは MOSFET によりオン/オフ切り替えが行われ、シングルエラーとしてのみシミュレートできます。

COM コントローラへのコマンド送信

```
int ctrl.Shortcut_xUBATTy_20A_RealTime
    (string ecu, string ecuPin, int load,
     int xUBATTy, int durationType);
```

COM コントローラからの応答の取得

```
byte[] Answer = ctrl.GetLastErrorSimulationAnswer();
```

CAN 送信メッセージ

ShortCut_xUBATTy_20A_realtime (channel_nr, load, xUBATTy, duration_flag)

第 1 バイト	コマンド ID	0x4
第 2 バイト	パラメータ 0	ES4440 のチャンネル番号
第 3 バイト	パラメータ 1	load, xUBatty, duration_flag (40 ページ参照)
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	不使用

CAN 受信メッセージ

応答

第 1 バイト	コマンド ID	0x4
第 2 バイト	パラメータ 0	ES4440 のチャンネル番号
第 3 バイト	パラメータ 1	不使用
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	コマンドの実行結果

3.4.9 Pin2PinFirstChWithoutLoad

2本のライン間の短絡について、第1のラインを定義します。第2のラインは、“Pin2PinSecondChannelWithoutLoad” コマンドで定義します（48ページを参照してください）。

注記

このエラーはリレーによってオン/オフ切り替えが行われ、負荷や抵抗なしでシミュレートされます。2本のピン間にはヒューズはありません。

COM コントローラへのコマンド送信

```
int ctrl.Pin2PinChannel1_WithoutLoad(string ecu,
                                     string ecuPin, int durationType);
```

COM コントローラからの応答の取得

```
byte[] Answer = ctrl.GetLastErrorSimulationAnswer();
```

CAN 送信メッセージ

Pin2PinFirstChWithoutLoad (channel_nr1, duration_flag)

第1バイト	コマンド ID	0x5
第2バイト	パラメータ 0	ES4440 のチャンネル番号 1
第3バイト	パラメータ 1	duration_flag (40 ページ参照)
第4バイト	パラメータ 2	不使用
第5バイト	パラメータ 3	不使用
第6バイト	パラメータ 4	不使用
第7バイト	パラメータ 5	不使用
第8バイト	パラメータ 6	不使用

CAN 受信メッセージ

応答

第1バイト	コマンド ID	0x5
第2バイト	パラメータ 0	ES4440 のチャンネル番号 1
第3バイト	パラメータ 1	不使用
第4バイト	パラメータ 2	不使用
第5バイト	パラメータ 3	不使用
第6バイト	パラメータ 4	不使用
第7バイト	パラメータ 5	不使用
第8バイト	パラメータ 6	コマンドの実行結果

3.4.10 Pin2PinSecondChannelWithoutLoad

ライン間短絡の第 2 のラインを定義します。

注記

このエラーはリレーによってオン/オフ切り替えが行われ、負荷や抵抗なしでシミュレートされます。2 本のピンの間にはヒューズはありません。

COM コントローラへのコマンド 送信

```
int ctrl.Pin2PinChannel2_WithoutLoad(string ecu,
                                     string ecuPin, int durationType);
```

COM コントローラからの応答の取得

```
byte[] Answer = ctrl.GetLastErrorSimulationAnswer();
```

CAN 送信メッセージ

Pin2PinSecondChannelWithoutLoad (channel_nr1, duration_flag)

第 1 バイト	コマンド ID	0x6
第 2 バイト	パラメータ 0	ES4440 のチャンネル番号 2
第 3 バイト	パラメータ 1	duration_flag (40 ページ参照)
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	不使用

CAN 受信メッセージ

応答

第 1 バイト	コマンド ID	0x6
第 2 バイト	パラメータ 0	ES4440 のチャンネル番号 2
第 3 バイト	パラメータ 1	不使用
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	コマンドの実行結果

3.4.11 Pin2PinFirstChRealtimeWithLoad

2本のライン間で発生する、負荷と有限抵抗を伴う短絡について、第1のラインを定義します。

このエラーは、MOSFETによってオン/オフ切り替えが行われます。

COM コントローラへのコマンド送信

```
int ctrl.Pin2PinChannel1_RealTime_WithLoad
    (string ecu, string ecuPin, int resistor,
     int current, int durationType);
```

COM コントローラからの応答の取得

```
byte[] Answer = ctrl.GetLastErrorSimulationAnswer();
```

CAN 送信メッセージ

Pin2PinFirstChRealtimeWithLoad (channel_nr1, duration_flag)

第1バイト	コマンド ID	0x7
第2バイト	パラメータ 0	ES4440 のチャンネル番号 1
第3バイト	パラメータ 1	current, duration_flag (40 ページ参照)
第4バイト	パラメータ 2	不使用
第5バイト	パラメータ 3	抵抗値 - バイト 0
第6バイト	パラメータ 4	抵抗値 - バイト 1
第7バイト	パラメータ 5	抵抗値 - バイト 2
第8バイト	パラメータ 6	抵抗値 - バイト 3

CAN 受信メッセージ

応答

第1バイト	コマンド ID	0x7
第2バイト	パラメータ 0	ES4440 のチャンネル番号 1
第3バイト	パラメータ 1	不使用
第4バイト	パラメータ 2	不使用
第5バイト	パラメータ 3	不使用
第6バイト	パラメータ 4	不使用
第7バイト	パラメータ 5	不使用
第8バイト	パラメータ 6	コマンドの実行結果

3.4.12 Pin2PinSecondChRealtimeWithLoad

ライン間の負荷と有限抵抗を伴う短絡について、第 2 のラインを定義します。
このエラーは、MOSFET によってオン/オフ切り替えが行われます。

COM コントローラへのコマンド送信

```
int ctrl.Pin2PinChannel2_RealTime_WithLoad
    (string ecu, string ecuPin, int durationType);
```

COM コントローラからの応答の取得

```
byte[] Answer = ctrl.GetLastErrorSimulationAnswer();
```

CAN 送信メッセージ

Pin2PinSecondChRealtimeWithLoad (channel_nr2, duration_flag)

第 1 バイト	コマンド ID	0x8
第 2 バイト	パラメータ 0	ES4440 のチャンネル番号 2
第 3 バイト	パラメータ 1	duration_flag (40 ページ参照)
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	不使用

CAN 受信メッセージ

応答

第 1 バイト	コマンド ID	0x8
第 2 バイト	パラメータ 0	ES4440 のチャンネル番号 2
第 3 バイト	パラメータ 1	不使用
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	不使用

3.4.13 RInline_realtime

MOSFET を用いたライン抵抗の切り替えについて設定します。

COM コントローラへのコマンド送信

```
int ctrl.RInline_RealTime(string ecu1,
                          string ecuPin1, int resistor,
                          int current, int durationType);
```

COM コントローラからの応答の取得

```
byte [] Answer = ctrl.GetLastErrorSimulationAnswer();
```

CAN 送信メッセージ

RInline_realtime (channel_nr1, resistor, duration_flag, current)

第 1 バイト	コマンド ID	0x9
第 2 バイト	パラメータ 0	ES4440 のチャンネル番号
第 3 バイト	パラメータ 1	current, duration_flag (40 ページ参照)
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	抵抗値 - バイト 0 * (LSB)
第 6 バイト	パラメータ 4	抵抗値 - バイト 1 *
第 7 バイト	パラメータ 5	抵抗値 - バイト 2 *
第 8 バイト	パラメータ 6	抵抗値 - バイト 3 * (MSB)

* バイトオーダーは Motorola フォーマットに対応しています。

```
Byte n      (LSB)
Byte n+1
Byte n+2
Byte n+3   (MSB)
```

したがって、抵抗値が 0x1234 である場合は以下のように設定します。

```
Byte n      0x34 (LSB)
Byte n+1   0x12
Byte n+2
Byte n+3   (MSB)
```

CAN 受信メッセージ

応答

第 1 バイト	コマンド ID	0x9
第 2 バイト	パラメータ 0	ES4440 のチャンネル番号
第 3 バイト	パラメータ 1	不使用
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用

応答		
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	コマンドの実行結果

3.4.14 Pullup_Pulldown_xUBATTy_20A_realtime

チャンネルを抵抗（プルアップ/プルダウン）経由でバッテリー電圧に接続します。

COM コントローラへのコマンド送信

```
int ctrl.PullUp_PullDown_20A_RealTime
    (string ecu, string ecuPin, int load, int xUBATTy,
     int resistor, int current, int durationType);
```

COM コントローラからの応答の取得

```
byte[] Answer = ctrl.GetLastErrorSimulationAnswer();
```

CAN 送信メッセージ

Pullup_Pulldown_xUBATTy_20A_realtime (channel_nr, load, xUBATTy, resistor, duration_flag, current)

第1バイト	コマンド ID	0xB
第2バイト	パラメータ 0	ES4440 のチャンネル番号
第3バイト	パラメータ 1	xUBatty, current, duration_flag, load (40 ページ参照)
第4バイト	パラメータ 2	不使用
第5バイト	パラメータ 3	抵抗値 - バイト 0 * (LSB)
第6バイト	パラメータ 4	抵抗値 - バイト 1 *
第7バイト	パラメータ 5	抵抗値 - バイト 2 *
第8バイト	パラメータ 6	抵抗値 - バイト 3 * (MSB)

* バイトオーダーについては、51 ページの「RInline_realtime」の項を参照してください。

CAN 受信メッセージ

応答

第1バイト	コマンド ID	0xB
第2バイト	パラメータ 0	ES4440 のチャンネル番号
第3バイト	パラメータ 1	不使用
第4バイト	パラメータ 2	不使用
第5バイト	パラメータ 3	不使用
第6バイト	パラメータ 4	不使用
第7バイト	パラメータ 5	不使用
第8バイト	パラメータ 6	コマンドの実行結果

3.4.15 Open_Load_400V

ECU と負荷の間の高電圧ラインの遮断について設定します。

このエラーは、リレーによってオン/オフ切り替えが行われ、シングルエラーとしてのみシミュレートできます。

COM コントローラへのコマンド送信

```
int ctrl.OpenLoad_400V(string ecu, string ecuPin,
                      int durationType, int set);
```

COM コントローラからの応答の取得

```
byte[] Answer = ctrl.GetLastErrorSimulationAnswer();
```

CAN 送信メッセージ

Open_Load_400V (channel_nr, set, duration_flag)

第 1 バイト	コマンド ID	0xD
第 2 バイト	パラメータ 0	ES4440 のチャンネル番号
第 3 バイト	パラメータ 1	duration_flag, set (40 ページ参照)
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	不使用

CAN 受信メッセージ

応答

第 1 バイト	コマンド ID	0xD
第 2 バイト	パラメータ 0	ES4440 のチャンネル番号
第 3 バイト	パラメータ 1	不使用
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	コマンドの実行結果

3.4.16 ShortCut_xUBATTy_400V

高電圧チャンネルのバッテリー電圧への短絡を設定します。このエラーは、リレーによってオン/オフ切り替えが行われ、負荷なしのシングルエラーとしてのみシミュレートできます。

COM コントローラへのコマンド送信

```
int ctrl.ShortCut_xUBATTy_400V
    (string ecu, string ecuPin, int xUBATTy,
     int durationType, int set);
```

COM コントローラからの応答の取得

```
byte[] Answer = ctrl.GetLastErrorSimulationAnswer();
```

CAN 送信メッセージ

ShortCut_xUBATTy_400V (channel_nr, xUBATTy, set, duration_flag)

第1バイト	コマンド ID	0xE
第2バイト	パラメータ 0	ES4440 のチャンネル番号
第3バイト	パラメータ 1	xUBATTy, duration_flag, set (40 ページ参照)
第4バイト	パラメータ 2	不使用
第5バイト	パラメータ 3	不使用
第6バイト	パラメータ 4	不使用
第7バイト	パラメータ 5	不使用
第8バイト	パラメータ 6	不使用

CAN 受信メッセージ

応答

第1バイト	コマンド ID	0xE
第2バイト	パラメータ 0	ES4440 のチャンネル番号
第3バイト	パラメータ 1	不使用
第4バイト	パラメータ 2	不使用
第5バイト	パラメータ 3	不使用
第6バイト	パラメータ 4	不使用
第7バイト	パラメータ 5	不使用
第8バイト	パラメータ 6	コマンドの実行結果

3.4.17 ShortCut_xUBATTy_400V_Ex

高電圧チャンネルのバッテリー電圧への短絡を設定します。このエラーは、リレーによってオン/オフ切り替えが行われ、シングルエラーとしてのみシミュレートできます。負荷あり/負荷なしを選択できます。

COM コントローラへのコマンド送信

```
int ctrl.ShortCut_xUBATTy_400V_Ex
    (string ecu, string ecuPin, int xUBATTy,
     int load, int durationType, int set);
```

COM コントローラからの応答の取得

```
byte[] Answer = ctrl.GetLastErrorSimulationAnswer();
```

CAN 送信メッセージ

ShortCut_xUBATTy_400V_Ex (channel_nr, load, xUBATTy, set, duration_flag)

第 1 バイト	コマンド ID	0xE
第 2 バイト	パラメータ 0	ES4440 のチャンネル番号
第 3 バイト	パラメータ 1	load, xUBATTy, duration_flag, set (40 ページ参照)
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	不使用

CAN 受信メッセージ

応答

第 1 バイト	コマンド ID	0xE
第 2 バイト	パラメータ 0	ES4440 のチャンネル番号
第 3 バイト	パラメータ 1	不使用
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	コマンドの実行結果

3.4.18 Pin_2_Pin_400V

2つの高電圧チャンネル間の短絡（負荷なし）を設定します。

このエラーは、リレーによってオン/オフ切り替えが行われ、負荷なしのシングルエラーとしてのみシミュレートできます。

COM コントローラへのコマンド送信

```
int ctrl.Pin2Pin_400V
    (string ecu1, string ecuPin1, string ecu2,
     string ecuPin2, int durationType);
```

COM コントローラからの応答の取得

```
byte[] Answer = ctrl.GetLastErrorSimulationAnswer();
```

CAN 送信メッセージ**Pin_2_Pin_400V (channel_nr1, channel_nr2, duration_flag)**

第1バイト	コマンド ID	0xF
第2バイト	パラメータ 0	ES4440 のチャンネル番号 1
第3バイト	パラメータ 1	duration_flag (40 ページ参照)
第4バイト	パラメータ 2	ES4440 のチャンネル番号 2
第5バイト	パラメータ 3	不使用
第6バイト	パラメータ 4	不使用
第7バイト	パラメータ 5	不使用
第8バイト	パラメータ 6	不使用

CAN 受信メッセージ**応答**

第1バイト	コマンド ID	0xF
第2バイト	パラメータ 0	ES4440 のチャンネル番号 1
第3バイト	パラメータ 1	不使用
第4バイト	パラメータ 2	ES4440 のチャンネル番号 2
第5バイト	パラメータ 3	不使用
第6バイト	パラメータ 4	不使用
第7バイト	パラメータ 5	不使用
第8バイト	パラメータ 6	コマンドの実行結果

3.4.19 Pin_2_Pin_400V_Ex

2つの高電圧チャンネル間の短絡を設定します。

このエラーは、リレーによってオン/オフ切り替えが行われ、シングルエラーとしてのみシミュレートできます。負荷あり/負荷なしを選択できます。

COM コントローラへのコマンド送信

```
int ctrl.Pin2Pin_400V_Ex
    (string ecu1, string ecuPin1, string ecu2,
     string ecuPin2, int load, int durationType);
```

COM コントローラからの応答の取得

```
byte[] Answer = ctrl.GetLastErrorSimulationAnswer();
```

CAN 送信メッセージ**Pin_2_Pin_400V (channel_nr1, channel_nr2, duration_flag)**

第1バイト	コマンドID	0xF
第2バイト	パラメータ0	ES4440のチャンネル番号1
第3バイト	パラメータ1	load, duration_flag (40ページ参照)
第4バイト	パラメータ2	ES4440のチャンネル番号2
第5バイト	パラメータ3	不使用
第6バイト	パラメータ4	不使用
第7バイト	パラメータ5	不使用
第8バイト	パラメータ6	不使用

CAN 受信メッセージ**応答**

第1バイト	コマンドID	0xF
第2バイト	パラメータ0	ES4440のチャンネル番号1
第3バイト	パラメータ1	不使用
第4バイト	パラメータ2	ES4440のチャンネル番号2
第5バイト	パラメータ3	不使用
第6バイト	パラメータ4	不使用
第7バイト	パラメータ5	不使用
第8バイト	パラメータ6	コマンドの実行結果

3.4.20 Reset_all_errors

ES4440.1/2 コンパクト故障シミュレーションモジュール上のすべてのエラーをリセットします。

COM コントローラへのコマンド送信

```
int ctrl.ResetAllErrors();
```

COM コントローラからの応答の取得

```
byte[] Answer = ctrl.GetLastErrorSimulationAnswer();
```

CAN 送信メッセージ**Reset_all_errors ()**

第 1 バイト	コマンド ID	0x10
第 2 バイト	パラメータ 0	不使用
第 3 バイト	パラメータ 1	不使用
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	不使用

CAN 受信メッセージ**応答**

第 1 バイト	コマンド ID	0x10
第 2 バイト	パラメータ 0	不使用
第 3 バイト	パラメータ 1	不使用
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	コマンドの実行結果

3.4.21 Activate_relay

所定の時間だけリレーをクローズさせ、設定済みエラーをアクティブにします。

すでに発行されているエラー設定用コマンドで“duration_flag”（第3バイトのビット6）がセットされている場合（“duration_flag”=1）は、“duration_time”に20ms～5sの範囲内の値を設定します。“duration_flag”=0の場合は、“duration_time”の値を-1または65535（0xFFFF）に設定する必要があります。

注記

マルチエラーを設定する場合は、必ず、すべてのエラーの“duration_flag”パラメータの値を同じにしてください。

リファレンスリレーで測定されたスイッチング遅延時間が、コマンド応答として返信されます。

COM コントローラへのコマンド送信

```
int ctrl.ActivateRelay(int durationTime);
```

COM コントローラからの応答の取得

```
byte[] Answer = ctrl.GetLastErrorSimulationAnswer();
```

CAN 送信メッセージ

Activate_relay (duration_time)

第1バイト	コマンド ID	0x12
第2バイト	パラメータ 0	不使用
第3バイト	パラメータ 1	duration_time [ms] - バイト 0
第4バイト	パラメータ 2	duration_time [ms] - バイト 1
第5バイト	パラメータ 3	不使用
第6バイト	パラメータ 4	不使用
第7バイト	パラメータ 5	不使用
第8バイト	パラメータ 6	不使用

“duration_time”パラメータには、20ms～5000msの範囲内の値（ただし20msの倍数）を設定します。

値を0xFFFFにすると、エラーが無期限でアクティブになります。

“channel_type”パラメータには以下のいずれかの値を設定します。

- 0：高電流チャンネル
- 1：高電圧チャンネル

CAN 受信メッセージ

応答

第1バイト	コマンド ID	0x12
第2バイト	パラメータ 0	delay_time0 20AのNOリレー*のクローズ時間（100 μs 単位）
第3バイト	パラメータ 1	delay_time1 20AのNOリレー*のクローズ時間（100 μs 単位）

応答		
第4バイト	パラメータ 2	delay_time0 20A の NC リレー ** のオープン時間 (100 μ s 単位)
第5バイト	パラメータ 3	delay_time1 20A の NC リレー ** のオープン時間 (100 μ s 単位)
第6バイト	パラメータ 4	delay_time0 400V の NC リレー ** のクローズ時間 (100 μ s 単位)
第7バイト	パラメータ 5	delay_time1 400V の NC リレー ** のクローズ時間 (100 μ s 単位)
第8バイト	パラメータ 6	コマンドの実行結果

* NO = Normally Open (ノーマリーオープン)
** NC = Normally Closed (ノーマリークローズ)

3.4.22 Activate_realtime_switch

MOSFET によるエラースイッチをクローズさせ、設定済みエラーをアクティブにします。

すでに発行されているエラー設定用コマンドで “duration_flag” (第3バイトのビット6) がセットされている場合 (“duration_flag” = 1) は、“duration_time” に 1ms ~ 5s の範囲内の値を設定します。“duration_flag” = 0 の場合は、“duration_time” の値を -1 または 65535 (0xFFFF) に設定する必要があります。

COM コントローラへのコマンド送信

```
int ctrl.ActivateRealTimeSwitch(int mode,
    int durationTime, int dutyCycle, int frequency);
```

COM コントローラからの応答の取得

```
byte[] Answer = ctrl.GetLastErrorSimulationAnswer();
```

CAN 送信メッセージ

Activate_realtime_switch (mode, duration_time, dutycycle, frequency)

第1バイト	コマンド ID	0x13
第2バイト	パラメータ 0	mode
第3バイト	パラメータ 1	duration_time - バイト 0
第4バイト	パラメータ 2	duration_time - バイト 1
第5バイト	パラメータ 3	不使用
第6バイト	パラメータ 4	接点不良発生のデューティサイクル
第7バイト	パラメータ 5	接点不良発生の周波数 - バイト 0
第8バイト	パラメータ 6	接点不良発生の周波数 - バイト 1

“duration_time” パラメータには、1ms ~ 5000ms の範囲内で 1ms 単位の値を指定します。値を 0xFFFF にすると、エラーが無期限にアクティブ状態となります。

接点不良シミュレーションを行わない場合は、パラメータ 4、5、6 の値を 0xFF にしてください。

“mode” パラメータには以下のいずれかの値を設定します。

- 0: 静的エラー。エラー持続時間は “duration_time” により定義されます。
- 1: 接点不良シミュレーション。

デューティサイクルの値には、周波数に応じて以下の制限があります。
周波数 3Hz ~ 100Hz の場合: 1 ~ 99%、2Hz の場合: 50%

CAN 受信メッセージ

応答

第1バイト	コマンド ID	0x13
第2バイト	パラメータ 0	mode
第3バイト	パラメータ 1	duration_time - バイト 0
第4バイト	パラメータ 2	duration_time - バイト 1
第5バイト	パラメータ 3	duration_time - バイト 2

応答		
第 6 バイト	パラメータ 4	duration_time - バイト 3
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	コマンドの実行結果

3.4.23 TestFuses

エラーレール用のヒューズのステータスを取得します。

COM コントローラへのコマンド送信

```
int ctrl.TestFuses(string deviceKey,
                  out int[] fuses);
```

ヒューズのステータスは、fuses という配列に格納されます。fuses[n] にヒューズ n+1 (n = 0 ~ 4) のステータスが格納され、ステータスが 1 なら「ヒューズは正常」、0 なら「ヒューズは不良」です。

"deviceKey" には、Standalone, Master, Slave1, ..., Slave14 のいずれかの値を設定します。

COM コントローラからの応答の取得

```
byte[] Answer = ctrl.GetLastErrorSimulationAnswer();
```

CAN 送信メッセージ**IDN コマンド**

第 1 バイト	コマンド ID	0x14
第 2 バイト	パラメータ 0	不使用
第 3 バイト	パラメータ 1	不使用
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	不使用

CAN 受信メッセージ**応答**

第 1 バイト	コマンド ID	0x14
第 2 バイト	パラメータ 0	Fuse_status
第 3 バイト	パラメータ 1	不使用
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	コマンドの実行結果

各ヒューズのステータスは、Fuse_status の下記の各ビットに格納されます。

MSB				LSB			
ビット 8	ビット 7	ビット 6	ビット 5	ビット 4	ビット 3	ビット 2	ビット 1
-	-	-	E5	E1	E3	E4	E2

(各ビットの値 = 0 : 不良、1 : 正常)

3.4.24 CurrentMeasurement

ES4440.1/2 コンパクト故障シミュレーションモジュールのフロントパネルにある“Current”コネクタにチャンネルを接続し、電流を測定できるようにします。

COM コントローラへのコマンド送信

```
int ctrl.CurrentMeasurement
    (string ecu, string ecuPin, byte set);
```

COM コントローラからの応答の取得

```
byte[] Answer = ctrl.GetLastErrorSimulationAnswer();
```

CAN 送信メッセージ

CurrentMeasurement (channel_nr)

第 1 バイト	コマンド ID	0x15
第 2 バイト	パラメータ 0	channel_nr
第 3 バイト	パラメータ 1	不使用
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	不使用

CAN 受信メッセージ

応答

第 1 バイト	コマンド ID	0x15
第 2 バイト	パラメータ 0	channel_nr
第 3 バイト	パラメータ 1	不使用
第 4 バイト	パラメータ 2	不使用
第 5 バイト	パラメータ 3	不使用
第 6 バイト	パラメータ 4	不使用
第 7 バイト	パラメータ 5	不使用
第 8 バイト	パラメータ 6	コマンドの実行結果

4 お問い合わせ先

製品に関するご質問等は、各地域の ETAS 支社までお問い合わせください。

ETAS 本社

ETAS GmbH

Borsigstrasse 14	Phone:	+49 711 89661-0
70469 Stuttgart	Fax:	+49 711 89661-106
Germany	WWW:	www.etas.com/

日本支社

イータス株式会社

〒 220-6217	Phone:	(045) 222-0900
神奈川県横浜市西区	Fax:	(045) 222-0956
みなとみらい 2-3-5	E-mail:	sales.jp@etas.com (営業窓口)
クイーンズタワー C 17F	WWW:	www.etas.com/

その他の支社

上記以外の各国支社につきましては、ETAS ホームページをご覧ください。

各国支社	WWW:	www.etas.com/ja/contact.php
技術サポート	WWW:	www.etas.com/ja/hotlines.php

索引

A

API 31

C

CAN インターフェース
設定 11

CAN コマンド

Activate_realtime_switch 62

Activate_relay 60

CurrentMeasurement 65

Open_Load 43

Open_Load_400V 54

Open_Load_realtime 44

Pin_2_Pin_400V 57

Pin_2_Pin_400V_Ex 58

Pin2PinFirstChRealtimeWithLoad 49

Pin2PinFirstChWithoutLoad 47

Pin2PinSecondChannelWithoutLoad
48

Pin2PinSecondChRealtimeWithLoad
50

Pullup_Pulldown_xUBATTy_20A_realti
me 53

Reset_all_errors 59

Rinline_realtime 51

ShortCut_xUBATTy_20A 45

ShortCut_xUBATTy_20A_realtime 46

ShortCut_xUBATTy_400V 55

ShortCut_xUBATTy_400V_Ex 56

TestFuses 64

一般的なコマンド構成 39

channels left 43

E

ES4440.1

システム設定 10

F

Failure Set

「故障セット」を参照

L

LABCAR-PINCONTROL

起動 17

T

TCP/IP

設定 8

い

イーサネット

設定 9

く

クリーニング

リレー接点 26

こ

故障セット

作成 19

シグナルの削除 21

シグナルの追加 20

し

シグナル

インポート 18
シミュレート
 接触抵抗 24
 接点不良 23
 マルチエラー 22

せ

接触抵抗
 シミュレート 24
設定
 CAN インターフェース 11
 ES4440.1 10
 TCI/IP 8
 イーサネット 9
 表示オプション 27
 マスタ/スレーブ 11
接点不良
 シミュレート 23
セルフテスト 27

て

電流
 測定 25

と

問い合わせ先 66

ひ

ヒューズテスト 27
表記
 規則 6
 操作手順 5
表示オプション
 設定 27

ま

マスタ/スレーブ
 設定 11
マルチエラー
 シミュレート 22

め

メインメニュー 29

り

リレー接点
 クリーニング 26

わ

ワイヤハーネスファイル 12
 作成 13