

CX2500 Series CODESYS Edition User Manual

(For Programming)

東京計器株式会社 TOKYO KEIKI INC.

油圧制御システムカンパニー

CCOT-24-016 Rev.1



1

Contents

1. 安全上(の注意	5
2. はじめ	て	7
3. 関連マン	ニュアル	8
4. 使用可能	能言語について	9
5. メイン国	画面について	
6. プログ	ラミング	
6.1. メニ	モリ容量	
6.2. ディ	バイスの I/O 更新設定	
6.3. プロ	ログラム作成フロー	14
6.4. PO	DU	
6.4.1.	POU の追加方法	
6.5. タン	スク	
6.5.1.	タスクの追加	
6.5.2.	タスクの設定・POU の割り当て	
6.6. ウン	オッチドッグタイマ	23
6.7. DU	JT	
6.7.1.	DUT の作成方法	
6.8. 変勢	数	
6.8.1.	一般的な型	
6.8.2.	変数宣言	
6.8.3.	サブレンジ	
6.8.4.	配列	
6.8.5.	列举型	
6.8.6.	型の別名定義	
6.8.7.	構造体	
6.8.8.	構造体(拡張)	
6.8.9.	共用体	
6.8.10.	定数	
6.8.11.	グローバル変数・ローカル変数	
6.8.12.	保持変数・持続変数	
6.9. ライ	イブラリ	
6.9.1.	ライブラリマネージャー	
6.9.2.	共通ライブラリ	
6.9.3.	当社独自ライブラリ	
6.9.4.	ライブラリアイテムの使用方法	51
6.9.5.	ユーザー定義のライブラリ	

Κεικί

CCOT-24-016	Rev.1
-------------	-------

6.10. ビルド	63
6.11. ログイン・ログアウト	
6.11.1. ログイン手順	
6.11.2. ログアウト手順	
7. CX2500 の機能について	
7.1. CX2500 機能一覧	
7.2. 機能ドライバについて	
7.2.1. 機能ドライバ画面	
7.2.2. Internal Parameters タブ	
7.2.3. Internal I/O Mapping タブ	
7.3. デジタル入力・イグニッション入力	
7.3.1. Internal Parameters タブ	
7.3.2. Internal I/O Mapping タブ	
7.4. 周波数入力	
7.4.1. Internal I/O Mapping タブ	
7.5. 2相カウンタ入力	
7.5.1. Internal I/O Mapping タブ	
7.5.2. カウンタ値セットの流れ	
7.6. アナログ入力	
7.6.1. Internal Parameters タブ	
7.6.2. Internal I/O Mapping タブ	
7.7. 内部電源電圧監視入力	
7.7.1. Internal I/O Mapping タブ	
7.8. 基板温度監視入力	
7.8.1. Internal I/O Mapping タブ	
7.9. デジタル出力	
7.9.1. Internal Parameters タブ	
7.9.2. Internal I/O Mapping タブ	
7.10. PWM 出力	
7.10.1. Internal Parameters タブ	
7.10.2. Internal I/O Mapping タブ	
7.10.3. DOモード	
7.10.4. 初期状態	
7.10.5. エラー時の処理と解除方法	
7.11. RS232C	
7.11.1. 列举型	
7.11.2. 構造体	
7.11.3. 関数	
7.11.4. ポートオープン・クローズ	

Κεικί

CCOT-24-016 Rev.1	NEIKI
7.11.5. ポートクローズ	
7.11.6. 受信	
7.11.7. 送信	
7.12. CAN	
7.12.1. 列举型	
7.12.2. 構造体	
7.12.3. 関数	
7.12.4. 初期設定	
7.12.5. 受信	
7.12.6. 送信	
7.13. タイマカウンタ	
7.13.1. 型の別名定義	
7.13.2. 関数	
7.14. RTC	
7.14.1. 列举型	
7.14.2. 構造体	
7.14.3. 関数	
7.14.4. 初期状態とバックアップ時間	
8. フィールドバスについて	
8.1. 概要	
8.2. 共通設定	
8.2.1. フィールドバスの紐づけ	
8.3. J1939	
8.3.1. CANbus	
8.3.2. J1939_Manager	
8.3.3. CAN デバイス	
8.4. CANopen	
8.4.1. CANbus	
8.4.2. CANopen_Manager	
8.4.3. CAN デバイス	
9. デバッグ機能について	
9.1. 基本画面	
9.2. アプリケーションの動作開始・停止	
9.2.1. 動作開始	
9.2.2. 動作停止	
9.3. シングルサイクル	
9.4. 変数モニタ	
9.4.1. 変数値表記方式の変更	
9.4.2. ウォッチ	

Κεικί

CCOT-24-016 Rev.1	NEIKI
9.5. 値の書き込み・強制	
9.5.1. 値の書き込み	
9.5.2. 値の強制	
9.5.3. 値の強制解除	
9.6. リセット	
9.6.1. リセット手順	
9.7. ブレークポイント	
9.7.1. ブレークポイントの設定(通常)	
9.7.2. ブレークポイントの設定(条件付き)	
9.7.3. ブレークポイントの無効化設定	
9.7.4. ブレークポイントの設定削除	
9.7.5. ブレークポイント中の操作	
9.7.6. ブレークポイントステータス	
9.8. デバイスログ	
9.9. トレース	
9.9.1. トレースの作成	
9.9.2. トレースの設定	219
9.9.3. トレースの開始・終了	
9.9.4. トレースの保存	
9.9.5. トレース履歴の読み込み	
9.10. タスクステータス監視	
9.11. シミュレーション(オフラインデバッグ)	
9.12. Visualization	
9.12.1. Visualization 作成	
9.12.2. 変数とアイテムの紐づけ例	
9.12.3. Visualization 実行例	
10. Tips	
10.1. プログラム中に IDE と通信できなくなった時は・・・	
10.1.1. セーフモード起動	
11. Revision history	



1. <u>安全上の注意</u>

本マニュアルで使用している安全に関する表示の意味は次の通りです。本書に記載した注意事項は、安全上重 大な内容を記載していますので、必ず厳守下さい。

なお、ハードウェアに関する安全上の注意については CX2500 機能仕様書(文書番号: CCOT-23-017)を熟読下 さい。

警告	この表示を無視して誤った取扱いをすると、人が死亡、又は重傷を負う可能性が想定される内 容が記載されています。
注意	この表示を無視して誤った取扱いをすると、人が傷害を負ったり、物的損害の発生が想定される
	内谷やての他自意すべき内谷が記載されています。

\oslash	禁止	この表示は <mark>実施してはいけない</mark> 内容が記載されています。
\$	必須	この表示は本製品を使用する上で <u>必ず実施する必要がある</u> 内容が記載されています。



デバッグ中におこなう動作開始/停止や値の強制、並びに出力の強制等は、安全に動作するこ 警告 とを操作前の確認を必ずおこなったうえで実施して下さい。確認不十分の場合、操作により事故 につながる可能性が有ります。 必須

	计辛	弊社が配布するデバイス定義・機能ドライバファイル及びランタイムファイルを編集し使わないで
	任恳	下さい。CX2500 並びに CODESYS の機能が損なわれたり、誤動作につながる可能性がありま
\oslash	禁止	す。

	警告	本製品に関する物品の分解や改造等はおこなわないで下さい。破損や誤動作、事故につなが る可能性があります。
\oslash	禁止	

	警告	本製品の使用に際し、万が一本製品に故障・不具合が発生した場合でもフィールセーフなどの 保全機能を機器外部などにユーザー側で必ず持たせて下さい。
0	必須	

	注意	本製品の仕様内で必ず使用して下さい。製品仕様外での使用は事故につながる可能性がありますので絶対におこなわないでください。
\bigcirc	禁止	

	注意	通電中に端子に触れたり、配線変更等をしないでください。製品の破損や感電などの事故につ ながる可能性が有ります。
\oslash	禁止	

		CX2500 や CODESYS 等のソフトウェアアップデートにより、製品ソフトの表示・表記が実際のも
	注意	のと本書記載のものが一部異なる場合があります。その際は、実際の表示・表記内容に従って
		下さい。



7

2. <u>はじめに</u>

本書は CX2500 シリーズ(以降、本製品と呼称)及び CODESYS®の使用に必要な情報やプログラミングなどに ついて説明している文書です。

本書および CX2500 については、安全上の注意ならびに下記項目について同意した場合のみご使用下さい。

【厳守項目】

- このマニュアルを熟読して下さい。
 - ▶ 本書は本製品を正しく使うための文書のため、必ず最後まで熟読の上で製品を使用して下さい。
- このマニュアルを大切に保管して下さい。
 - ▶ 本製品を取り扱う場合、このマニュアルは重要ですので、いつでも参照できるように身近に且つ大切に 保管ください。
- このマニュアルを取扱者の手元に届けて下さい。
 - ▶ 代理店など、本製品の仲介になる方は、必ずこのマニュアル(URL 情報含む)を実際に取り扱う方の手元 に届けて下さい。
- このマニュアルを紛失した場合は直ぐに補充して下さい。
 - ▶ 万が一、このマニュアルを紛失した場合は、弊社営業所または購入先までご連絡下さい。
- このマニュアルを断りなく転載することはできません。

【注意項目】

- このマニュアルは予告なく変更する場合があります。
- このマニュアルに記載されている CODESYS の画面表示等はバージョンアップによって一部異なる場合が あります。その際は、実際の画面表示で確認下さい。
- CODESYS は 3S-Smart Software Solutions GmbH の登録商標です。
- NXP、NXP ロゴは NXP B.V の商標です。
- 上記以外で、このマニュアルに記載されている会社名、製品名はそれぞれ弊社もしくは第三者の商標や登録 商標です。
- 本書および CX2500 仕様の内容を逸脱して製品を使用したことによって生じた不具合故障やその他損害に ついて、弊社は一切責任を負いません。
- 弊社は、本ソフトウェアについて、ソフトの使用やバグ、誤動作や不具合、その他本ソフトウェアにより生じた損害についても一切責任を負いません。
- 弊社は本書に記載されている情報に誤りがないことを保証するものではありません。本書に記載された情報の誤りにより、ユーザーまたは第三者に損害が生じた場合においても弊社は一切責任を負いません。



3. <u>関連マニュアル</u>

本製品のマニュアルの一覧は下記の通りです。各種目的に応じてご使用下さい。

Table 1 CX2500 シリーズ マニュアル一覧

名称	摘要
CX2500Codesys_UserManual_ForSetup	CODESYS のインストールとそのセットアップ、CX2500の接続方
	法などについて記載しています。
CX2500Codesys_UserManual_ForProgramming	本製品を扱う上での、CODESYS でのプログラミングやデバッグ
(本書)	の基礎的な内容、CX2500に搭載されている各種機能について
	記載しています。



4. 使用可能言語について

CODESYS では、国際電気標準会議(IEC)が制定した IEC61131-3 に規定されている PLC のプログラミング 言語で開発することができます。使用できる言語は全部で 5 種類(Table 2)あり、ユーザーが作成するアプリケ ーションの用途に合わせてそれぞれ選択し使用することができます。

プログラミング言語	記述形式	特長
ラダー図言語	グラフィック	従来から日本国内で最も用いられている言語であり、シーケンス処理の
(LD 言語)		記述に適している。
命令リスト言語	テキスト	アセンブラに似たローレベル言語であり、高速に動作する小さいモジュー
(IL 言語)		ルを記述するのに適している。
構造化テキスト言語	テキスト	パソコン用プログラム言語に似たテキスト言語で、数式演算の記述又は
(ST 言語)		IFTHEN 文並びに FORDO 文のような分岐制御の記述に適している。
ファンクションブロック言語	グラフィック	FB 同士のピンをつなぎ合わせて電子回路図のように記述できる言語で
(FBD 言語)		あり、連続したデータ処理の記述に向いている。最近では、ラダー図言語
		と混在記述できるプログラミングツールもある。
シーケンシャルファンクシ	グラフィック	状態遷移に基づくフローチャートであり、順序制御処理の記述に向いてい
ョンチャート言語		る。
(SFC 言語)		

Table 2 IEC61131-3 のプログラミング言語*1

※1 出典: JEMA「PLC アプリケーションの開発効率化検討」

Table 3 プログラミング言語の得意・不得意*1

主な使用状況	LD 言語	IL 言語	ST 言語	FBD 言語	SFC 言語	
単純なリレーシーケンス処理	Ø	×	Δ	Δ	×	
数式演算処理	Δ	×	Ø	\bigcirc	×	
状態遷移に基づく順序制御	Δ	×	0	×	Ø	
(ステップシーケンス処理)						
連続的なアナログ信号処理	Δ	×	\bigcirc	Ø	×	
複雑な情報処理	Δ	×	Ø	Δ	×	
プログラムメモリ制約の厳しい場合	\bigcirc	Ø	Δ	Δ	×	
最も高速に性能を求められる場合	0	Ø	\bigcirc	\bigcirc	×	
運転方案と対応がとりやすい表現	×	×	\bigcirc	\bigcirc	Ø	
動作を視覚的に確認したい場合	0	×	×	Ø	\bigcirc	
注記 記号の意味は、次による。						
◎:最も適している、○:適している、Δ:困難な場合もある、×:適さない						



5. <u>メイン画面について</u>

CODESYS-IDE のメインウィンドウについて解説します。下図はプロジェクト(使用言語:ST)を開いたときの基本的な画面です。なお、各部の位置や表示可否については、ユーザーが自由にカスタマイズできます。

1																			
SampleProject_Cx2500.project - CODESYS																		-	a x
File Edit View Project Build O	nline De	bug Tools	Window	Help	2015	r													T
	929) (C.); 9	S (S M	14 14 14 1		Ap	plication [Device	e PLC Logic]	- 09 09 - 1	■ % [[2 22 5	9.19.16	第 〒 ▽								
Devices - # X	P 🗐 P	LC_PRG X																	
SampleProject_Cx2500	2	♦ ×					_				PROGRAM	PLC_PRG			_				3
PLC Logic	1	Scope	Name ucVal0	Address	Data type	Initialization 0	Comment	Attributes											L
E C Application	2	🕈 VAR	ucVal1		USINT	10													
PLC_PRG (PRG) Task Configuration																			
🖻 🍪 MainTask																			
- @ PLC_PRG 1 Cx2500Ain (Cx2500Ain)																			
Cx2500Cntin (Cx2500Cntin)	¢1										A 7								>
Cx2500Do (Cx2500Do)	2																		
Cx2500Pwm (Cx2500Pwm)	4																		
1			E																
4			D																
	-																		
	Messages	Fotal 0 error	(s), 0 warnin;	g(s), 0 messag	e(s)	O error(s)	0 warning(s)	0 message	r(s) 🗙 💥										- 4
	Descript	ion													Project	_	Object	Position	
																6			
																<u> </u>			
_																			
POUs Se Devices	El Messa	iges - Total 0 e	rror(s), 0 war	ning(s), 0 mes	sage(s) 🔣 W	'atch 1 🚮 Break	points 📮 Mer	nory											
4 •												Last build: 🔿 0 📀 0 Pr	ecompile 🦯 🔗	Project	ser (nohody)		INS	Lo.4 Col.1 Ch.1	

Figure 1 メイン画面

#	名称	摘要
1	タイトルバー	このエリアには、画面全体のサイズ変更や CODESYS の終了ができる機能があります。
2	メニューバー	このエリアには CODESYS の各種コマンドがあり、目的別にメニュー表示されています。
3	ツールバー	このエリアには、各種コマンドがアイコンで表示されています。アイコンを押すとそれぞれ
		に割り当てられたコマンドがはたらきます。
4	デバイスウィンドウ	このエリアには、現在編集しているプロジェクトのデバイスやプログラム等がツリー状に
		一覧表示されています。
5	エディタウィンドウ	このエリアでは、変数宣言・プログラムの編集等、ユーザーアプリケーションの編集作業
		をおこなうことができます。
6	メッセージ・デバッグ	このエリアでは、ビルド結果の詳細やブレークポイントやウォッチ等のデバッグ情報が表
	情報ウィンドウ	示されます。
7	ステータスバー	このエリアには、ビルド結果やアプリの運転状態等の情報が表示されます。





6.1. メモリ容量

CX2500(CODESYS版)について、ユーザーがアプリケーション開発に使用できるメモリは下記の通りです。

種別	名称	摘要	メモリ容量
ROM	Memory area 0	アプリケーションのソース・実行コード保存領域	1MB
RAM	Memory area 1	アプリケーションの通常変数データ保存領域	84KB
RAM ^{%2}	Memory area 2	アプリケーションの保持変数※2データ保存領域	4KB
	Memory area 3	アプリケーションの持続変数**2データ保存領域	8KB

Table 5 CX2500(CODESYS 版) ユーザー用メモリ容量

※2 保持変数・持続変数については、6.8.12 項を参照。



6.2. デバイスの I/O 更新設定

デバイスの I/O(入出力)データ更新に関する設定をおこないます。I/O とは、本製品において主に Table 6 が 該当します。入力データはタスクの始めに、出力データはタスクの終わりにそれぞれ更新がおこなわれます。 そのデータ更新に関する設定をおこないます。設定画面は、デバイスウィンドウから「Device(CX2500)」を

ダブルクリックし、表示される「Device」→「PLC Settings」(Figure 2)になります。各設定項目については Table 7 を参照下さい。

Table 6 CX2500 I/O						
機能名	参照先					
<u>デジタル入力</u>	7.3 節					
イグニッション入力	7.3 節					
<u>周波数入力</u>	7.4 節					
2 相カウンタ入力	7.5 節					
<u>アナログ入力</u>	7.6 節					
<u>内部電源電圧監視入力</u>	7.7 節					
基板温度監視入力	7.8 節					
デジタル出力	7.9 節					
PWM 出力	7.10 節					

evices 🗸 🗸 🗸	PLC_PRG Device X		
SampleProject_Cx2500 Bevice (CX2500) Pict Cloic Pict Cloic	Communication Settings Applications Backup and Restore Files Log PLC Settings PLC Shell Users and Groups Access Rights Symbol Rights Licensed Software Metrics	Application for I/O handling Application PLC Settings Update I/O while in stop Behavior for outputs in stop Set all outputs to default Always update variables Disabled (update only if used in a task) Bus Cycle Options Bus cycle task Additional Settings	**
>	Task Deployment		
Devices POUs	<		
essages - Total 1 error(s), 0 warning(s), 0 messag	e(s)		-
ild	• O error(s) 😗 O warning	g(s) 0 message(s) 🗙 🕅	
escription		Project	Object Position
The application is up to date			

Figure 2 Device PLC Settings



項目	デフォルト	摘要				
Application for I/O	[Application]	I/O データ更新を担当	当するアプリケーション名。			
Handling		選択肢	摘要			
		Application	ユーザーアプリケーションで I/O 更新処理を			
			おこなう。			
		None	ユーザーアプリケーションで I/O 更新処理を			
			おこなわない。			
PLC Settings	無効(チェック無し)	有効時、PLC(CX250				
Update I/O while in		│ │続ける。 無効時は PL	_C 動作時のデータで停止される。			
stop						
Behavior for outputs in	ΓSet all outputs to	PLC が動作停止して	いる時の出力状態を設定する。			
stop	Default」	選択肢	摘要			
		Keep current	動作停止時の値を保持する。			
		values				
		Set all outputs to	I/O が各機能ドライバで設定したデフォルト			
		default	値にする。			
		Execute program	停止時の出力制御を指定したプログラムで			
			実行する。			
Always update	「Disabled」	 有効時、全ての I/O	データをプログラムの使用有無に関わらず更新			
variables		する。無効時は各 1/0	O の機能ドライバでデータを更新するか設定す			
		る必要がある。				
Bus Cycle Options	「 <unspecified>」</unspecified>	バスサイクルタスクを	·制御するタスクを選択する。「 <unspecified>」を</unspecified>			
		選択した場合は、最初	も短い周期のタスクになる。			
Additional Settings	全て無効	項目	有効化(チェック有)時の効果			
		Enable diagnosis	有効にすると、ライブラリ			
		for devices	「CAADeviceDiagnosis ^{※3} 」を用いたデバイス			
			診断が可能となる。			
		Show I/O	ビルド時、I/O 構成に関する警告をエラーと			
		warnings as errors	して表示する。			
		Enable symbolic	各 I/O に変数を自動的に作成し割り当て			
		access for IOs	る。			

※3 ライブラリ「CAADeviceDiagnosis」とそのデバイス診断機能については、CODESYS オンラインヘルプを参照下さい。



6.3. <u>プログラム作成フロー</u>

プログラムの作成の流れの例を下記に示します。







6.4. <u>POU</u>

POU(Program Organization Unit)は、プログラムの単位を指します。ユーザーはこの POU を作成し(1 つは プロジェクト作成時に自動生成される)、そこにプログラム(処理)を記述していきます。従来のラダーと異なり、 IEC61131-3 では POU をアプリケーションの機能・制御毎に複数作成することで、特定の処理がどの POU(機 能・制御)に割り当てられているのかが分かりやすくなっています。ただし、1 つの POU に対して記述できる言 語は1種類に限定されることに留意下さい。

POUには以下の通り、3種類のタイプが有ります。

Table 8	POU	タイプ一覧

タイプ	摘要	呼び出し方の例
Program	・タスクに割り当てることで呼び出すことができる。	6.5.2 項
	・他の POU からも呼び出すことは可能。	
	・POU 内の変数値は保持される。	
Function	・呼び出す際は、呼び出し元の POU で Function block のインスタンス(変数宣言部で	6.9.5.2 項
block	宣言する)を作成する。	
	・それぞれのインスタンス内の変数値は保持される。	
	・出力を複数持つことができる。	
Function	・呼び出す際は、Function blockのように呼び出し元の POU で宣言する必要は無い。	6.9.4 項
	コード部から Function 名で直接呼び出せる。	
	・Function内の変数値は全て保持されない。(呼び出す毎に初期値になる)	
	・出力は1つしか持てない。	



CCOT-24-016 Rev.1 6.4.1. POUの追加方法

ここでは、プロジェクトへの POU の追加例を示します。

① デバイスウィンドウにて、「Application」を右クリックして下さい。

vices 🗸 🗸 🗙	PLC PRG M Device >	2					
SampleProject_Cr/2500 SampleProject_Cr/2500 Device (Cx2500) Application Application Application Application P.C.pRG (PRG) P.C.pRG (PRG) P.C.pRG (PRG) Cx2500cnt (Cx2500ch) Cx2500cnt (Cx2500ch) Cx2500cnt (Cx2500ch) Cx2500cn (Cx2500ch) Cx2500ch Cx2500ch	Communication Settings Applications Backup and Restore on Files Log PLC Settings PLC Shell Users and Groups Access Rights Symbol Rights Licensed Software Metrics	Application for I/O handling PLC Settings U Uddat I/O while in stop Behavior for outputs in stop Always update variables Bus cycle Options Bus cycle task Additional Settings Generate force variables fo Show I/O warnings as error	Application	1			
Devices POUs	Task Deployment						
ssages - Total 1 error(s), 0 warning(s), 0 message ild	(s)	ing(s) \rm 0 message(s) 🗙 💥					•
escription The application is up to date Build complete 0 errors, 0 warnings : Ready for	r download				Project	Object	Position

Figure 4 メイン画面 Application の選択

② 表示されたコンテキストメニューから「Add Object」→「POU…」を選択して下さい。

File Edit View Project Buik	d Onlin	e Debug Tools Windo	ow I	Help Help · · · · · · · · · · · · · · · · · · ·	rice PLCLogic] • ଓ ଓ → 田 代 [日 9日 4日 1日 谷 中 第 市 予
United Sample/Poject_CA2500 ■ Sample/Poject_CA2500 ■ Poice (CA2500) ■ Processor ■ Cassor ■	Cut Copy Paste Delete Proper Add Ol Add Fo Edit Ob Edit Ob Edit Ob Edit Ob	Communication Settings Applications ies ject ider ject the get with consection from device symptomorphics Licensed Software Metrics Task Deployment		Application for I/O handling PLC Settings DIdate I/O while in stop Behavior for outputs in stop Always update variables Avis Group Carn table CNC program CNC settings Communication Manager DUT External File Global Variable List Global Variable List Global Variable List Interface Persistent Variables Port 4	Application
POUs		<	(B)	POU	• • • • • • • • • • • • • • • • • • •
Messages - Total 1 error(s), 0 warning(s), 0 Build) message(s)	#1 #1	POU for Implicit Checks Redundancy Configuration	• 4
Description The application is up to date Build complete 0 errors, 0 warnings :	Ready for	download	• () 8 8 8 8 9 9 9 9	Symbol Configuration Text List Trace Trend Recording Manager Unit Conversion Visualization	Project Object Position
			- -	Visualization Manager	

Figure 5 Applcation コンテキストメニュー POU の選択



③ 「Add POU」ウィンドウが表示されるので、以下3点を記入・選択し「Add」ボタンを押して下さい。

設定項目							
Name	POU 名を入	、力する。					
Туре	以下 3 つのタイプから 1 つを選択する。						
	タイプ	設定値					
	Program	特に無し。					
	Function	•Extend	:有効化する(チェックを入れる)と、既に作成されている				
	block		Function blockを拡張することができる。				
		 Implements 	:有効化する(チェックを入れる)と、既に定義されている				
			Function block インターフェースの処理内容を記述できる。				
		• Final	:有効化して(チェックを入れる)作成したものを、別の Function				
			block で拡張できなくする。				
		•Abstract	:有効化して(チェックを入れる)POU を作成すると、抽象				
			Function block(入出力のブロックのみ)で作成される。これは				
			上記 Extend 用の Function block として使用されることが一般				
			的。				
		•Access	:以下2つから選択する。				
		specifier	(1)INTERNAL:作成した POU 内でのみ呼び出すことができる。				
			(2)PUBLIC:どの POU からも呼び出すことができる。				
		•Method	:インターフェースの実装を行う場合の実装言語を選択する。				
		implementation					
		language					
	Function	Return type(返り	直)のデータ型を入力する。				
Implementation	以下 5 言語	から1つを選択す	る。				
language	•Continuou	s Function Chart(C	FC)				
	 Continuou 	s Function Chart(C	FC) – page oriented				
	•Function E	Block Diagram(FBD)					
	•Ladder Lo	gic Diagram(LD)					
	Structure	d Text(ST)					
	•Sequentia	I Function Chart(SF	FC)				

Table 9 Add POU 設定項目



Add POU	×
Create a new POU (Program Organization Unit)	
Name	
POU_Ctrl	1
Type	1
Program	
O Function block	
Extends	
Implements	
Final Abstract	
Access specifier	
~	
Method implementation language	
Continuous Function Chart (CFC)	
○ Function	
Return type	
Implementation language	
Structured Text (ST)	*
Add Cancel	

Figure 6 Add POU ウィンドウ

④ 「Application」ツリーに作成した POU が表示されます。

SampleProject_Cx2500.project* - CODESYS		_	o ×
File Edit View Project Build Online D	bug Tools Window Help		₹2
🗎 🚅 🔲 📾 🗠 a 👗 🗈 🛍 🗶 🖓 🍇	🍇 📗 🎕 🎕 🍓 顓・ 🕆 🏙 Application [Device: PLC Logic] ・ 🧐 🧐 🕞 😑 😤 [三 雪 雪 雪 雪 巻 中 第 雪 型		
Devices X	PLC PRG M Device POU CtrlA X		
= ☐ SampleProject Cx2500	1 PROGRAM FOU_CtrlA		1
□ Device (CX2500)	2 VAR		
E-B PLC Logic	a END_VAR		
Application	1		
DUT_SAMPLE			
Library Manager			100
DOL CTA (PRG)			
E M Task Configuration			
🖹 🍪 MainTask			
PLC_PRG			
Cx2500Ain (Cx2500Ain)			
Cx2500Cntn (Cx2500Cntn)			
Cx2500Dn (Cx2500Dn)			
Cx2500Frgin (Cx2500Frgin)			
ff Cx2500Pwm (Cx2500Pwm)			
			100
POUs <			100 [[
lessages - Total 1 error(s), 0 warning(s), 0 message(s)			• # ?
Build	 O 0 error(s) O warning(s) O message(s) X X 		
Description	Pro	ject Object	Position
The application is up to date			
Build complete 0 errors, 0 warnings : Ready for downlo	ad in the second s		
	Last build 🔿 0 🔿 0. Pressentile 🖌 🥂 Pressent (schools) nut	1 Cold Chil	
	Last bolin V v recompile V Project user: (hobody) INS Lh	I COLI CHI	

Figure 7 POU 作成後



6.5. <u>タスク</u>

プログラム(POU)を動作させるには、タスクへの割り当てが必要です。タスクは、POU がどのような実行周期・方式で処理するかを管理します。タスクに割り当てられていない POU は実行されません。

タスクには下記のようにタイプが分かれています。なお、最大タスク数は3つです。各タスクは、別の実行中のタスクがある場合、それが終わるまで実行されることはありません。タスク周期<タスク実行時間(9.10節参照) にしてしまうと処理に大きな遅れが出る場合があることに注意して設計して下さい。

Table 10 タスクのタイプ

タイプ	摘要
Cyclic	実行周期が設定でき、その設定した周期でタスクが実行される。
Event	実行イベントとして変数(BOOL 型 ^{*4})が設定でき、その変数が 0(FALSE)→1(TRUE)になった時に
	タスクが1度実行される。

※4 BOOL型については、6.8.1 項を参照下さい。

6.5.1. <u>タスクの追加</u>

タスクの追加方法について下記に示します。

① デバイスウィンドウの「Task Configuration」にカーソルを合わせ、右クリックして下さい。

SampleProject_Cx2500.project* - CODESYS File Edit View Project Build Onlin	ne Debug Tools Window Help	□ × ₹2
🛅 📽 🖬 I 🚳 I 🗠 🗠 👗 🐚 🛍 🗙 I 🖗	🛿 🍪 🍓 🏠 月 🖄 🦄 🖷 🎽 🛗 🏥 Application [Device: PLC Logic] 🔹 🧐 🦻 🕞 🔳 % 💭 💷 🖆 🏷 中 第 中 第 中 ジ	
Devices 4 X Image: SampleProject_Cr2200 Image: SampleProject_Cr2200 Image: SampleProject_Cr2000 Image: SampleProject_Cr2000 Image: SampleProject_Cr2000 Ima	PLC_PRG Cx2500An Device MainTask X Configuration Priority (031): Type G Cyclic V Interval (e.g. t#200ms) 100 Watchdog Bradle Time (e.g. t#200ms) Sensitivity	ns v
	Add Call X Remove Call Change Call Adve Up ♦ Move Down Open POU POU Comment POL PRG	
Z Devices POUs Messages - Total 1 error(s), 0 warning(s), 0 message	۵	- -
Precompile	- 🧿 0 error(s) 😗 0 warning(s) 🜒 0 message(s) 🗙 💥	
Description	Project Object	Position
	Last build 🔿 0 🛧 0 Precompile 🥒 🧏 Protect user: (nobadiv)	() (2)

Figure 8 メイン画面 Task Configrarion の選択



CCOT-24-016 Rev.1 ② 表示されたコンテキストメニューから「Add Object」→「Task...」を選択して下さい。

SampleProject Cx2500.project* - CODE	SYS – D
File Edit View Project Build	Online Debug Tools Window Help
1 🖉 🖬 🛃 🗠 a 👗 🖻 🚯 1	× 桷 協 桷 猛 貝 別 別 酒 圖 10+ 11 圖 Application (Device: PLC Logic) - 📽 岡 ト 😑 😤 [三 일 🖕 罰 合 11 名 ト 罰 (デ) シ
Devices 👻	🖡 🗙 👔 PLC_PRG 👔 Cx2500Ain 👔 Device 🎁 Library Manager 🔗 MainTask 🗙
SampleProject_Cx2500	Configuration
E Device (CX2500)	
PLC Logic	Priority (0.31): 1
= O Application	Type
DUT_SAMPLE	(Cyclic y Interval (e.g. t#200ms) 100 ms
Diprary Manager	
	Wathdog
Task Configuration	
🗐 🎲 MainTask	Cut
PLC_PR	Copy ms) ms
Cx2500Ain (Cx2500Ain) 💼	Paste
Cx2500Cntin (Cx2500Cn 🗙	Delete
Cx2500Din (Cx2500Din)	Properties
Cx2500Do (Cx2500Do)	en contractione de la contraction de la contract
Cx2500Frqin (Cx2500Frq	Add Object) 🚱 Task nge Call 🐨 Move Up 🐨 Move Down 📄 Open POU
CX2500PWm (CX2500PWI	Add Folder Comment
6	Edit Object
	Edit Object With
-	
😪 Devices 👔 POUs	
Messages - Total 1 error(s), 0 warning(s), 0 m	essage(s)
Precompile	
Description	Project Object Porition
Description	rigett objett romon
	Last build: 🔿 0 😗 0 Precompile 🗸 👘 Project user: (nobody) 🔇

Figure 9 Task Configration コンテキストメニュー Task の選択

③ 「Add Task」ウィンドウが表示されるので、タスク名を入力し「Add」ボタンを押して下さい。

Add Task		\times
An IEC task		
Name		
Task1		
	Add	Cancel

Figure 10 Add Task ウィンドウ



④ タスクが新しく追加され、タスクの設定画面が表示されると完了です。

SampleProject_Cx2500.project* - CODESYS File Edit View Project Build Onlin	e Debug Tools Window Help 结晶 44 (月 領 領 領 酒 酒 〇 〇 〇 〇 〇 〇 〇 〇 〇 〇 〇 〇 〇 〇 〇	- 0 × ₹2
Devices • # # SampleProject_Cx2500 • Device (Cx2500) • Device (Cx2500) • Device (Cx2500) • Dut_Clopic • Dut_Clopic • Dut_SAMPLE • Out_SAMPLE • Out_SAMPLE	PLC_PRG Cx2500An Configuration Priority (0.31): Type Cyclic Interval (e.g. t#200ms) D00 Watchdog Dnable Time (e.g. t#200ms) Sensitivity Add Call X Remove Call Change Call * Move Up * Move Down * Open POU POU Comment	
Messages - Total 1 error(s), 0 warning(s), 0 message(~ ₽ >
Description	Project Obje	ct Position

Last build: 🗿 0 😗 0 Precompile 🗸 🦙 Project user: (nobody) 🖉 🐼



6.5.2. <u>タスクの設定・POU の割り当て</u>

タスクの設定画面では、以下のような設定項目があります。各種アプリケーションの仕様に合わせ設定して下 さい。

iority (031):		
ype Ocyclic v	Interval (e.g. t#200ms) 100	ms
Vatchdog		
Enable		
Time (e.g. t#200ms)		m
Constituity 1		
1 Isoto Isot		
Add Call 🗙 Remove Call 🗔	Change Call 🗇 Move Up 🔿 Move Down 🍑 Open POU	
	Comment	
OU		

Figure 11 タスク設定画面

Table 11 タスク 設定項目

#	項目	摘要		
1	Priority ^{%5}	複数のタスクがある場合の当該タスクの実行優先度を設定する。優先度は0~31まで設定で		
		き、値が小さいほど優先度は高くなる。		
2	Туре	タスクのタイプ(Table 10)を設定する。		
		タイプ	設定値	
		Cyclic	Interval(周期時間)を設定する。単位は ms。	
		Event	Eventを設定する。Event に設定できる変数はユーザーが定義した BOOL 型	
			変数に限られる。	
3	Watchdog	タスクのウォッチドッグタイマを設定できる。詳細は 6.6 節を参照。		
4	POU Settings	タスクに割り	当てる POU を設定できる。1 つのタスクに POU を複数割り当てた場合、この実	
		行リストの最	上段にある POU から順番に実行される。	

※5 本製品はマルチタスク方式ではありません。優先度が高いタスクでも、実行中の別のタスクが終わるまで実行されないことに 留意して下さい。

コマンド名	摘要
Add Call	タスクに POU を追加する。
Remove Call	タスクの実行リストで選択した POU を削除する。
Change Call	タスクの実行リストで選択した POU を別の POU に変更する。
Move Up	タスクの実行リストで選択した POU を 1 段リストアップ(実行順を 1 つ上げる)する。
Move Down	タスクの実行リストで選択した POU を 1 段リストダウン(実行順を 1 つ下げる)する。
Open POU	タスクの実行リストで選択した POU のエディタ画面を開く。

Table 12 POU 割り当て コマンド一覧



6.6. <u>ウォッチドッグタイマ</u>

ウォッチドッグタイマはプログラム(アプリケーション)が暴走や異常停止していないかを確認するための機能 を指します。この機能を有効にして監視時間を設定した場合、プログラムの実行時間が監視時間を超過した時 にウォッチドッグエラーとしてプログラムの実行を停止します。

ウォッチドッグタイマの設定はタスク毎におこなう必要が有ります。ウォッチドッグタイマは下図の通り、 タスクの設定画面で設定できます。設定項目としては、機能の有/無効の他に Time・Sensitivity が有ります。 ウォッチドッグタイマエラーの条件とその例を Table 13 に示します。

PLC_PRG Cx2500Ain	Device 👘 Library Manage	er 🕸 MainTask 🗙	
Priority (031): 1 Type Cyclic V	nterval (e.g. t#200ms) 100		ms V
Watchdog Enable Time (e.g. t#200ms) Sensitivity 1			ms ∨
🖶 Add Call 🗙 Remove Call 📝	Change Call 🕆 Move Up 🐥 Move Dow	rn → Open POU	
	Comment		

Figure 12 ウォッチドッグタイマ 設定エリア

Table 13 ウォッチドッグタイマ エラー条件とその例

エラー条件	下記いずれかを満たすこと。
	(1)プログラムの実行時間が Time ^{※6} × Sensitivity ^{※6} の時間を超えた。
	(2)プログラムの実行時間が Sensitivity ^{※6} の回数連続で Time ^{※6} の時間を超えた。
例	Time:20[ms]、Sensitivity:5を設定した場合のエラー条件は下記いずれかを満たす時。
	(1)プログラムの実行時間が 20[ms]×5=100[ms]を超えた。
	(2)プログラムの実行時間が 5 回連続で 20[ms]を超えた。

※6 この Time · Sensitivity は、タスク設定画面で Time · Sensitivity に設定した値を指す。



6.7. <u>DUT</u>

DUT(Data Unit Type)は、ユーザーが定義できるデータ型のことです。DUT として定義できる型は以下の通りです。

データ型	参照先
<u>列挙型</u>	6.8.5 項
型の別名定義	6.8.6 項
構造体	6.8.7 項
<u> 共用体</u>	6.8.9 項

Table 14 DUT 定義可能なデータ型

6.7.1. <u>DUT の作成方法</u>

DUT の作成方法を下記に示します。

① デバイスウィンドウの「Application」にカーソルを合わせ右クリックして下さい。

Sumplex Polet, CA2500 Configuration Pic Direct (CA2500) Pic Interval (e.g. t#200ms) Direct (Rig) Pic Interval (e.g. t#200ms) Pic Direct (Rig) Watchdog Pic Direct (Rig) Pic Interval (e.g. t#200ms) Pic Direct (Rig) Pic Interval (e.g. t#200ms)	ms V
CX20001 (CX20001) Produce A Renove Call Catalge Call I Move of Move Down Open FOO	
□ CL2500Frain (Cx2500Frain) POU Comment □ ① Cx2500Frain (Cx2500Frain) ● PLC_PRG □ ① Cx2500Frain (Cx2500Frain) ● PLC_PRG □ ① Cx2500Frain (Cx2500Frain) ● PLC_PRG	
≤ >> See Devices ∫① POUs	
Messages - Total 2 error(p), 0 warning(p), 0 message(p)	→ ậ
Precompile • 0 1 error(s) • 0 warning(s) • 0 message(s) × 🔆	
	osition

Figure 13 メイン画面 Application の選択



② 表示されるコンテキストメニューから「Add Object」→「DUT...」を選択して下さい。



Figure 14 Application コンテキストメニュー DUT の選択



③ 「Add DUT」ウィンドウが表示されるので、DUT の名前と定義したい型を選択し、「Add」ボタンを押して下さい。

Add DUT	×
Create a new data unit	type
Name	
DUT_TEST0	
Туре	
Structure	
Extends	
○ Enumeration	
Textlistsupport	
◯ Alias	
Base type	>
OUnion	
	Add Cancel

Figure 15 Add DUT ウィンドウ

Table 15 DUT タイプと設定項目

タイプ	設定項目	摘要
Structure	Extends	有効化すると、既に定義されている構造体をさらに拡張した構造体を定義で
(構造体)		きる。対象となる構造体の型を入力するか「…」ボタンを押して型を選択する。
Enumeration	Textlist support	有効化すると、列挙型のメンバが1つ追加された状態で定義が作成される。
(列挙型)		
Alias	BaseType	別名定義する型を入力するか「>」ボタンを押して型を選択する。
(型の別名定義)		
Union	—	型名以外設定項目無し。
(共用体)		



④ DUT が新しく追加され、そのエディタウィンドウが表示されれば完了です。エディタウィンドウにて所望の メンバを記述してデータ型を完成させて下さい。

SampleProject_Cx2500.project* - CODESYS		- ¤ ×
File Edit View Project Build Onlin	ne Debug Tools Window Help	₹2
🛅 🚅 📕 🚳 🗠 🗠 👗 🛅 🛍 🗶 🖗	🕯 🎲 🍓 🌿 📕 🐄 🎢 🌾 🔚 🎬 - 📑 🛗 Application [Device: PLC Logic] 🔹 🥞	(영 → = ≪ [피 데 데 데 양 ㅎ 罰 = ∿
Devices 👻 🕂 🗙	Tait PLC PRG T RA Cx2500Ain T RA Device T A Library Manager T 와 M	lainTask 🕼 Task1 🖓 DUT TESTO 🗙
SampleProject_C-2200 SampleProject_C	1 TIPE DUI_TESTO : 2 STRUCT 2 END_TIPE 5	
< >>		100 1
Messages - Total 2 error(s), 0 warning(s), 0 message	(5)	• a :
Precompile	O 1 error(s) O warning(s) O message(s) X X	
Description		Project Object Position
O Task1: The task event is invalid.		Sample Task1 [
l	Last build: 😋 0 😗 0 Precompile 😋 🧃	👸 Project user: (nobody) INS Ln 1 Col 1 Ch 1 💡 🙆

Figure 16 メイン画面 DUT 作成後



6.8. <u>変数</u>

6.8.1. <u>一般的な型</u>

変数はデータをメモリに格納するためのもので、ユーザーはそのデータを使って制御(処理)をおこなったりします。変数には「型」があり、この「型」宣言をして変数に格納できる値の範囲を定義する必要が有ります。

項目	型名	値の範囲	メモリサイズ[bit]		
ブール	BOOL	0,1	1		
整数(符号有り)	SINT	-128~127	8		
	INT	-32768~32767	16		
	DINT	$-2^{31} \sim 2^{31} - 1$	32		
	LINT	-2 ⁶³ ~2 ⁶³ -1	64		
整数(符号なし)	USINT	0~255	8		
	UINT	0~65535	16		
	UDINT	0∼2 ³² −1	32		
	ULINT	0∼2 ⁶⁴ −1	64		
	BYTE	0~255	8		
	WORD	0~65535	16		
	DWORD	0∼2 ³² −1	32		
	LWORD	0∼2 ⁶⁴ −1	64		
実数	REAL	-3.402E+38 ~ -1.175E-38	32		
		1.175E−38 ~ 3.402E+38			
	LREAL	-1.797E+308~-2.225E-308	64		
		2.225E-308~1.797E+308			
文字列	STRING	(文字数+1)×8	(文字数+1)×8		
	WSTRING	(文字数+1)×16	(文字数+1)×16		

Table 16 一般的な変数の型

PLC_PRG X PROGRAM PLC_PRG 1 2 VAR З 3 bFlg : BOOL := FALSE; //Declaration of "BOOL" 4 ucVal0 : USINT := 0; //Declaration of "UINT" 5 //Declaration of "REAL" rVal : REAL := 1.0; 6 sStr : STRING := 'Tokyokeiki'; //Declaration of "STRING" 7 END VAR

Figure 17 例:一般的な型の宣言



6.8.2. <u>変数宣言</u>

変数の宣言は POU のエディタ部(下図赤枠内)でおこなう必要があります。 変数の宣言手法としては、主に以下の3つがあります。詳細は後述します。

【宣言手法】

- 「Add Variable」で宣言
- 「Textual View」で宣言
- 「Tabular View」で宣言





Textual・Tabular View の切り替えは青枠のボタンからできます。



Figure 19 POU エディタ Tabular View



CCOT-24-016 Rev.1 6.8.2.1. 「Add Variable」で宣言

ここでは、「Add Variable」で変数を宣言する手法を記します。

変数宣言部で右クリックして下さい。表示されるコンテキストメニューから「Refactoring」→「Add Variable...」を選択します。

PLC_PRG Cx2500Ain PLC_PRG	Library Manager	MainTask POU_CtrlA X	•
🈼 🔹 🐥 🗙		PROGRAM POU_CtrlA	b
Scope Name Address Data typ	e Initialization Comment	Attributes	
	X Cut		
4	Сору Сору		
	🔁 Paste	~	_
1	× Delete		
	Select All		
	Advanced	•	
	Refactoring	Add Variable	
	Edit Declaration Header		
	🔖 Insert		
	Move Down		
	1 Move Up		
		100	R
<			>

Figure 20 変数宣言部コンテキストメニュー Add Variable



 ② 「Add variable '??' to '(POU の名前)''」ウィンドウが表示されるので、各設定項目に所望の値を設定し「OK」 ボタンを押します。

Add variable '??' to 'POU_CtrlA'.			×
Scope VAR ~	Name ValA	Type ULINT	~ >
Object POU_CtrlA [Application] V	Initialization	Address	
Flags CONSTANT RETAIN PERSISTENT	Comment		^ ~
		OK	Cancel

Figure 21 Add variable '??' to '(POU の名前)' ウィンドウ

Table 17 Add Variable 設定項目

設定項目	摘要
Scope	「VAR」を選択する。(ローカル変数の場合) ^{※7}
Name	変数名を入力する。
Туре	変数の型を入力する。
Object	変数が属する POU 名を選択する。(ローカル変数の場合) ^{※7}
Initialization	変数の初期値を入力する。
Address	値は記入しない。
CONSTANT	チェックを入れて有効化した場合、この変数は定数として扱われる。定数値は Initialization 欄に
	入力する。
RETAIN	チェックを入れて有効化した場合、この変数は保持変数※8として扱われる。
PERSISTENT	チェックを入れて有効化した場合、この変数は持続変数※8として扱われる。
Comment	変数に関するコメント(メモ)を自由に記入できる。

※7 グローバル変数の場合、宣言の仕方が異なる。6.8.11 項を参照。

※8 保持変数・持続変数については 6.8.12 項を参照。



③ 変数宣言部に変数が追加されます。

		-
	PIC_FKG CX25004II DEVICE LUGary Manager Maintask POU_CLIA X	
8		
T.	3 ValA: ULINT := 0;	
L.	4 END_VAR	
	5	
		100
	-	
		100
<		>

Figure 22 Add Variable 宣言結果

6.8.2.2. 「Textual View」で宣言

ここでは、「Textual View」で変数を宣言する手法を記します。

- ① 変数宣言部が Tabular View になっている場合は Textual View にして下さい。
- ② 下図のように、宣言したい変数を直接記入します。これで宣言は完了です。

📄 PLC_PRG 🔐 Cx2500Ain 🔐 Device 👔 Library Manager 😒 MainTask 📄 POU_CtrlA 🗙	•
1 PROGRAM POU Ctrlà	The later
Vala: OLINI := 0:	
4 END_VAR	
5	
	100
1	
	100
	100
<	>

Figure 23 Textual View での宣言



CCOT-24-016 Rev.1 6.8.2.3. 「Tabular View」で宣言

ここでは、「Tabular View」で変数を宣言する手法を記します。

- ① 変数宣言部が Textual View になっている場合は Tabular View にして下さい。
- ② 「Insert」ボタンを押して下さい。

\$ J	X						PROGRAM POU_CtrlA	
^	Scope	Name	Address	Data type	Initialization	Comment	Attributes	
sert								
Т								
								د
1							A Y	
								100

Figure 24 Tabular View Insert の選択

③ 下図のように、仮の変数が表示されるので、各設定項目に所望の値を入力して下さい。

PLC	_PRG 🔐 Cx	2500Ain	Device	Library Man	ager 🚷	MainTask POU_CtrlA X	-
(4)	* X					PROGRAM POU_CtrlA	1
^	Scope Nam	e Address	Data type	Initialization	Comment	Attributes	
	VAR newV	ar	BOOL				
<							>
٢							100

Figure 25 Tabular View 仮変数追加

Keiki

CCOT-24-016 Rev.1

④ 入力が終わったら変数の宣言完了です。



Figure 26 Tabular View 宣言結果

6.8.3. <u>サブレンジ</u>

サブレンジ型で宣言すると、宣言した変数値の範囲を元の型の範囲内よりも絞って指定することができます。

 usBox : UINT (200..1000);
 // Declaration of "UINT (range : 200-1000)"

 Figure 27 宣言例(Textual View): サブレンジ型の宣言(UINT 型で 200~1000 のみ格納可能)

6.8.4. <u>配列</u>

同じ型のデータを複数格納したい場合、配列を使うと複数のデータを1つの変数名称で管理・使用すること が可能です。

```
8 usAry : ARRAY [0..10] OF UINT; //Declaration of "ARRAY(UINT)"
```

Figure 28 宣言例(Textual View): 配列の宣言(10 個の INT データ持ちの1次元配列)

```
1

2 //Array

3 FOR Idx := 0 TO 10 DO

4 usAry[Idx] := Idx * 2;

5 END_FOR

6
```

Figure 29 呼出例:配列(配列の各要素に値を代入する)



6.8.5. <u>列挙型</u>

列挙型は、複数の定数を1つの型に纏めて管理することができます。例えば、列挙型で変数を宣言した場合、 その型に無い値は「誤った値」としてその変数に割り当てることはできないため、エラーの判定が容易になりま す。また、型に含めた定数に名前を付けることによって、その値の意味が分かりやすくなりコードの可読性が良 くなります。

なお、この定義は DUT を作成(6.7.1 項)する必要があります。



//Enum

enEnum0 := EN_SAMPLE_ENUM0.ENUM_SP2; //Set "ENUM_SP2 (=2)" to "ucVal0"

Figure 31 呼出例:列挙型(列挙型変数に ENUM_SP2 を代入する)


6.8.6. <u>型の別名定義</u>

ユーザーは Table 16 などの既にある型を別の名前で定義することができます。分かりやすい名前をつけるこ とによって、その型で宣言した変数がアプリケーション上でどんな用途で使われるのかが分かりやすくなります。

なお、この定義は DUT を作成(6.7.1 項)する必要があります。 呼び出し方については、元々の型と同じ方法になります。



Figure 32 宣言例(Textual View):型の別名定義

6.8.7. 構造体

構造体は型の異なる複数のデータを纏めて格納できます。構造体は関連性のあるデータを纏めることでそれら を一括管理できます。

なお、この定義は DUT を作成(6.7.1 項)する必要があります。

```
1 TYPE ST_SAMPLE_STRUCT0 :
2 STRUCT
3 ucMember0 : USINT;
4 ulMember0 : ULINT;
5 ucMemArray0 : ARRAY[0..3] OF USINT;
6 END_STRUCT
7 END_TYPE
8
```

(a)構造体の定義

stSmp0 : ST_SAMPLE_STRUCT0;

//Declaration of "STRUCT"

(b)aの型を用いての変数宣言

Figure 33 宣言例(Textual View):構造体の定義

9 //Struct 10 stSmp0.ucMember0 := 0; 11 stSmp0.ulMember0 := 10; 12 stSmp0.ucMemArray0[0] := 11; 13 stSmp0.ucMemArray0[1] := 22; 14

Figure 34 呼出例:構造体(構造体変数の各メンバに数値を代入する)



6.8.8. <u>構造体(拡張)</u>

既に定義されている構造体をさらに拡張し、別の構造体名で定義することができます。定義は DUT を作成 (6.7.1 項)する必要があります。構造体(拡張)は、Table 15 に従って設定し定義して下さい。



Figure 35 宣言例(Textual View): 拡張構造体の定義(Figure 33 で定義した構造体を拡張)

```
16
     //Struct(Ext)
17
     stSmpl.ucMemberl := 0;
     stSmpl.ulMemberl := 10;
18
19
     stSmpl.ucMemArray1[0] := 11;
     stSmpl.ucMemArrayl[1] := 22;
20
21
22
     stSmpl.ucMember0 := 0;
23
     stSmpl.ulMember0 := 10;
     stSmpl.ucMemArray0[0] := 11;
24
25
     stSmpl.ucMemArray0[1] := 22;
26
```

Figure 36 呼出例:拡張構造体(構造体変数の各メンバに数値を代入する)



6.8.9. <u>共用体</u>

共用体は、同じメモリ領域を複数のデータの型で共用できるものを指します。共用体内で宣言した変数をそれ ぞれメンバと言いますが、それらは同じメモリ領域をシェアしているため、メモリの節約ができます。また、メ ンバの型はそれぞれ別の型で定義できるため、用途に応じた型で使うことが可能です。

下図は、8バイトのメモリ領域を「8個のUSINT型データを持つ配列」と「1個のULINT型データ」が共用できる共用体の定義例です。

なお、この定義は DUT を作成(6.7.1 項)し宣言する必要があります。



(a)共用体の定義

unSmp : UN_SAMPLE0;

//Declaration of "UNION"

(b)a の型を用いての変数宣言

Figure 37 宣言例(Textual View): 共用体の定義

Figure 38 呼出例:共用体(共用体変数とUSINT 型変数に数値7を代入する)



6.8.10. <u>定数</u>

変数への値の代入などで定数を記述する場合、その種類によって下表の通り記述する必要が有ります。

項目	型(一般的なもの)	記述方法					
ブール	BOOL	TRUE(1)、FA	LSE(0)				
整数	SINT	進数表記					
	INT	進数	記述方法	記述例			
	DINT	2 進数	2#(数値)	2#11001000			
	LINT	8 進数	8#(数値)	8#310			
	USINT	10 進数	(数値)	200			
	UINT	16 進数	16#(数値)				
	UDINT		•		-		
	ULINT						
	BYTE						
	WORD						
	DWORD						
	LWORD						
実数	REAL	実数表記					
	LREAL	例:1.2、3.45	E+6				
文字列	STRING	'(文字列)'					
		例:'Tokyoke	eiki'				
	WSTRING	"(文字列)"					
		例:"Tokyoke	eiki"				

Table 18 定数 記述方法

6.8.11. <u>グローバル変数・ローカル変数</u>

変数には、宣言する場所によってグローバル変数とローカル変数に分けられます。グローバル変数の宣言手順は、ローカル変数(6.8.2 項)と異なります。

Table 19 グローバル変数とローカル変数

名称	摘要	変数宣言場所
グローバル変数	全ての POU で使用できる変数。変数の値も共有される。	Global Variable List
ローカル変数	宣言した POU 内でのみ使用できる変数。	各 POU



6.8.11.1. グローバル変数の宣言

ここでは、グローバル変数の宣言手順を示します。

グローバル変数リストがデバイスツリーにない場合、グローバル変数リストを作成します。デバイスウィンドウの「Application」にカーソルを合わせ右クリックして下さい。既にリストを作成している場合は手順⑤から始めて下さい。

vevices -	7 X PLC_PRG Device X	Library Manager 🚯 MainTask 📄 POU_CtrlA				•
Application C 2530 Porter (C2300) C 2500 C 25000 C 250000 C 25000 C 25000 C 250000 C	Communication Settings Sam Applications Backup and Restore Piles Log PLC Settings PLC Settings PLC Settings PLC Settings Access Rights Symbol Rights Lucesed Software Metrics	Netnork Gateway • Device •	MyDevice (active) Device Name: MyDevice Device Admen: Dysie Admen: Dysie Admen: Dysie Admen: Target Type: Hype Vender: Typy Vender: Typy Vender: Typy Vender:	Y		^
1	> Task Deployment		Target Version: 3.5.18.30			~
Devices POUs	¢		010120100			>
lessages - Total 1 error(s), 0 warning(s), 0) message(s)				-	ņ >
recompile	O error(s) O warning(s)	0 message(s) X X	During	Ohinst	Desition	_
Jest Ipauri		Last build 💿 0 🙊 0 Preco	noje 🗸 🔞 Project user:	(nobody)	Position ⊘	6

② 表示されたコンテキストメニューから「Add Object」→「Global Variable List...」を選択します。



Figure 40 Application コンテキストメニュー Global Variable List の選択



③ 「Add Global Variable List」ウィンドウが表示されるので、リスト名を入力し「Add」ボタンを押して下さい。

Add Gl	lobal Variable List	×
۵	Create a new global variable list	
Name GVL		
	Add Cancel	

Figure 41 Add Global Variable List ウィンドウ

 ④ デバイスツリーにグローバル変数リストが追加され、さらにリストのエディタ画面も表示されます。これで、 リストの作成は完了です。



Figure 42 メイン画面 Global Variable List 追加後



⑤ 次に、グローバル変数を宣言します。グローバル変数リストのエディタ画面で所望の変数を宣言します。下 図は、初期値0のUDINT型グローバル変数を宣言した時の宣言例です。これで変数の宣言は完了です。

\square) P	PLC_PRG 🕅 Device 🍿 Library Manager 🐼 MainTask 👔 POU_CtrlA 🏾 🍘 GVL 🗙	•
	1	(attribute 'qualified_only')	1
8	2	VAR GUBAAL G 1974a10 - IDDRT -= 0	
	4	END VAR	
		-	
		100	R

Figure 43 Global Variable List エディタ 変数の宣言

6.8.11.2. グローバル変数の呼出

グローバル変数を POU で呼び出す時の記述方法について記します。ここでは、6.8.11.1 項で宣言したグロー バル変数を例に呼び出しをおこないます。

グローバル変数を呼び出す際は(グローバル変数リスト名).(変数名)と記述することで、記述した POU で呼び 出し(使用)することができます。

PLC_PRG Pevice	Library Manager 🛛 🍪 MainTask	POU_CtrlA 🗙 🎑 GVL	_
1 PROGRAM POU_CtrlA			
E 2 VAR			
3 ValA: ULINT := 0;			
4 END_VAR			
5			
			100
			100 [SN
1	1		
2 GVL.g udval0 := 100;			
			100
			100 EN

Figure 44 POU エディタ グローバル変数の宣言



6.8.12. <u>保持変数 持続変数</u>

変数の中には、CX2500の電源を落とした後でも値を保存し保持できる変数があります。保持変数と持続変数 です。なお、電源を落とす際に変数の値を保存するには、必ず Figure 45 の手順で行う必要が有ります。バッテ リの電源をいきなり落とす(停電等)と変数値が保存できないことに留意して下さい。

Table 20 保持変数と持続変数

名称	摘要
保持(RETAIN)変数	下記以外では変数の値を保持する。
	・アプリケーションの書き換え
	・Reset Cold(コールドリセット) ^{※9}
	•Reset Origin(PLC 初期化) ^{※9}
持続(PERSISTENT)変数	Reset Origin ^{※9} 以外では変数の値を保持する。

※9 これらのリセットについては 9.6 節を参照。



色付き部がユーザーによって机上若しくは実際の現場で行う必要があるもの



6.8.12.1. <u>保持変数の宣言手順</u>

保持変数の宣言手順を記します。ここではローカル変数で宣言していますが、グローバル変数でも宣言可能で す。

POU の変数宣言部にて所望の変数を宣言します。Add Variable(6.8.2.1 項参照)で宣言する場合、「Add variable '??' to '(POU の名前)"」ウィンドウで変数の情報を入力する際に下図のように「RETAIN」を有効化 (チェックを入れる)必要が有ります。

Add variable '??' to 'POU_CtrlA'.		×
Scope VAR ~	Name RetainVal0	Type USINT V >
Obiect POU_CtrlA [Application] ~	Initialization 0	Address
Flags CONSTANT RETAIN	Comment RETAIN VALUE	
		OK Cancel

Figure 46 Add Variable 保持変数の宣言

② 変数の情報を入力した後、「OK」ボタンを押すと変数の宣言は完了です。保持変数の場合、変数のスコープ は「VAR」ではなく「VAR RETAIN」になります。

б	VAR	RETAIN						
7		RetainVal0	:	USINT	:=	0;	//RETAIN	VARIABLE
8	END	VAR						
9								

Figure 47 POU エディタ 保持変数宣言後



6.8.12.2. <u>持続変数の宣言手順</u>

持続変数は必ずグローバル変数で宣言する必要があります。手順は以下の通りです。

 「Persistent Variables」がデバイスツリーにない場合、「Persistent Variables」を作成します。デバイスウ ィンドウの「Application」にカーソルを合わせ右クリックして下さい。既にリストを作成している場合は手 順⑤から始めて下さい。

SampleProject_Cx2500.project* - CODESYS		– 0 ×
File Edit View Project Build Onli	ne Debug Tools Window Help	₹2
🖹 🚅 🔚 🎒 🗠 🗠 👗 🗈 🕄 🌶	🕯 🤹 🍓 🌿	
Devices - 4 ×	👔 PLC_PRG 👔 Device 👔 Library Manager 🚳 MainTask 👔 POU_CtrlA 🗙 🍏 GVL	•
SampleProject_C-2500 SampleProject_C-	<pre>END_VAR END_VAR VAR RETAIN VALUE // RETAIN VALUE RetainVal0: USINT := 0; END_VAR // RetainVal0 := 100; RetainVal0 := 100; </pre>	▲ 100 @ ×
Cx2500Frqin (Cx2500Frqin) Cx2500Pivm Cx2500Pivm Cx2500Pivm PDUIs PDUIs		100

Figure 48 メイン画面 Application の選択

② 表示されるコンテキストメニューから「Add Object」→「Persistent Variables...」を選択して下さい



Figure 49 Application コンテキストメニュー Persistent Variables の選択



③ 「Add Persistent Variables」ウィンドウが表示されるので、名称を入力し「Add」ボタンを押して下さい。

Add Pe	ersistent Variables		×
Ţ	Create a new global var	iable list	
Name			
Persist	tentVars		
		Add	Cancel

Figure 50 Add Persistent Variables ウィンドウ

④ デバイスツリーに「Persistent Variables」が追加され、さらに「Persistent Variables」のエディタ画面も表示されます。これで、「Persistent Variables」の作成は完了です。



Figure 51 メイン画面 Persistent Variables 追加後



- ⑤ 持続変数を宣言します。ここでは「Add Variable(6.8.2.1 項)」で宣言する場合の手順を記します。「Persistent Variables」のエディタ画面にて右クリックをして下さい。
- ⑥ 表示されるコンテキストメニューから「Refactoring」→「Add Variables」を選択して下さい。

/ 📄 PL	C_PRG	ľ	🎁 Library Manager 🛛 🚷	MainT	ask	POU_CtrlA	🔕 GVL	🤇 🙆 PersistentVa	s x	•
1	{attribute 'qual	ifie	d_only'}							1
2	VAR_GLOBAL PERSI	STEN	T RETAIN							
4	END VAR	Ж	Cut							
	-		Сору							
		6	Paste							
		\times	Delete							
			Select All		1					
			Advanced	•	1					
			Input Assistant							
			Refactoring		anti.					
		_	Relactoring		1. U	Add variable				
			Add All Instance Paths							
			Reorder List and Clear Gaps							
			Save Current Values to Recipe		1					
			Restore Values from Recipe							
		_								
										100

Figure 52 Persistent Variables エディタ コンテキストメニュー

 ⑦ 「Add variable '??' to '(Persistent Variables 名称)'」ウィンドウにて変数の情報を入力します。この時、 「RETAIN」と「PERSISTENT」の2つを有効化(チェックを入れる)する必要があります。

Add variable '??' to 'PersistentVars'.		×
Scope VAR_GLOBAL ~	Name PersistVal0	Type USINT V >
Obiect PersistentVars [Application] V	Initialization 0	Address
Flags CONSTANT RETAIN PERSISTENT	Comment PERSISTENT VALUE	^ ~
		OK Cancel

Figure 53 Add Variable 持続変数の宣言



⑧ 情報を入力した後、「OK」ボタンを押すと変数の宣言は完了です。持続変数の場合、変数のスコープは「VAR」ではなく、「VAR GLOBAL PERSISTENT RETAIN」になります。これで、変数の宣言は完了です。

	PLC_PRG 🔐 Device 👘 Library Manager	🍪 MainTask	POU_CtrlA	GVL 🖉	🗿 PersistentVars 🗙	-
	<pre>1 {attribute 'qualified_only'} 2 } 3 } 4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</pre>					
н	2 VAR GLUDAL PERSISTENT RETAIN	_				
	4 // PERSISTENT VALUE					
	5 PersistVal0: USINT := 0;					
	END_VAR					
						100

Figure 54 Persistent Variables List 持続変数宣言後

6.8.12.3. <u>保持変数・持続変数の呼出</u>

ここでは、保持変数・持続変数を POU で呼び出す際のエディタでの記述方法について記します。各変数は下 図の通り記述することによって POU で使用可能になります。

4 END_VAR	^ 👿
5	
S VAD DETAIN	
8 // RETAIN VALUE	
9 RetainVal0: USINT := 0;	
10 END_VAR	
11	
	100
1	
GVL.g_udVal0 := 100;	
4 RetainVal0 := 100: //Call RETAIN value	
5 PersistentVars.PersistVal0 := 200; //Call PERSISTENT value	
	100

Figure 55 POU エディタ 保持・持続変数の呼出



6.9. <u>ライブラリ</u>

ライブラリは、POU や DUT などのアイテムを纏めた 1 つのファイルのことを指します。これは、これまで作 成したような IEC アプリケーション用のプロジェクトとは別のファイルで保存します。これにより、ユーザー はいくつものアプリケーションプロジェクトで利用・流用することができ、工数の削減につなげることができま す。

6.9.1. <u>ライブラリマネージャー</u>

アプリケーションプロジェクトには Library Manager が紐づけられています。Library Manager では、その プロジェクトで使用できるライブラリの一覧及び情報の確認とライブラリの追加・削除などが行えます。

ライブラリマネージャーを開くには、デバイスウィンドウの「Library Manager」をダブルクリックして下さい。Library Manager 画面にてライブラリファイルを左クリックすると画面下部にそのライブラリに含まれるア イテムの情報が表示されます。

SampleProject_Cx2500.project* - CODESYS File Edit View Project Libraries	Build Online Debug Tools Window Help Ma 산 Ma 산 I 및 및 및 및 개 (圖) (월 - 급) (편 Applica	tion [Device: PLC	Logic] 🔹 😋 🥨	→ = % (C=	≣ ¢⊒ '	¢ ¢	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	- 0 > (2
evices - 4 >	C I PLC PRG R Device St MainTask	Library Manage	r 🗙 🎑 Persisten	tVars	POU CtrlA	🧟 GVL		
SampleProject_Cx2500	Add Library X Delete Library Properties in Details	Placeholders	👔 Library Reposi	tory 🕕 Icon Le	gend 🗎	Summary		0
Device (CX2500)	Libraries used in application 'Device.Application'							
	Name			Namespace	Effective	e Version		
	B-1 3SLicense = 3SLicense, 3.5, 18.0 (3S - Smart Software Solution	ins GmbH)		35 LICENSE	3.5.18.0			
DUT_SAMPLE	BreakpointLogging = BreakpointLogging Functions, 3,5,17,0	(3S - Smart Softwa	re Solutions GmbH)	BPLog	3.5.17.0	0		
	CAA Device Diagnosis = CAA Device Diagnosis, 3.5, 18.0 (CA)	A Technical Worko	oup)	DED	3.5.18.0	0		
	CANbus, * (3S - Smart Software Solutions GmbH)			CANbus	3.5.17.0	0		
	IoStandard = IoStandard, 3.5.17.0 (System)			IoStandard	3.5.17.0	0		
Tack Configuration	Standard = Standard, 3.5.18.0 (System)			Standard	3.5.18.0	0		
- Se MainTack	SysCom = SysCom, 3.5.17.0 (System)			SvsCom	3.5.17.0	0		
	R. SveTime * (Svetam)			SueTima	3 5 17 0	0		
	Contents of selected library 'Standard, 3.5.18.0 (System)'	Details about se	lected library elemen	t 'CTD'				
S Tack1	Chadard 2.5 18 0 (Sustan)	····	ute 🔊 Creatical		-			
	Bistahla Exaction Placks	inputs/out	Graphical	7 Documentatio	n			
		FUNCTION_E	LOCK CTD					
Cx2500Catia (Cx2500Catia)			Name	Tune	Inho	Addross	Initial	Comment
Cx2500Din (Cx2500Din)		At norm	Traine .	type	minem	Address	initiai	comment
Cx2500Da (Cx2500Da)			0	BOOL				Rising edge: Decre
Cx2500Erain (Cx2500Erain)			LOAD	BOOL				IRUE : Set C
Cx2500Pwm (Cx2500Pwm)	E Microlanov II		PV	WORD				Start value for decr
CA2500F Will (CA2500F Will)	Christen Europhians	Ø OUTPUT	Q	BOOL				TRUE IF CV
		OUTPUT	CV	WORD				Current counter value
	Tringer							
Devices POUs	< >							
ssages - Total 0 error(s), 0 warning(s), 0 messa	ge(s)							→ ‡
rary Manager	- O error(s) (0 warning(s) (0 message(s) >	< 💥						
scription							Proj	ect Object Position
			Last build: 🔿 🗈		ile . 🖌	68	Project user: (ashadu) 🗿 🤇

Figure 56 メイン画面 Library Manager

ライブラリの登録・削除は、それぞれ「Add library」・「Delete Library」ボタンを押すことでできます。なお、 CODESYS-IDE にインストールされていないライブラリは登録ができません。インストール方法は 6.9.5.2 項を 参照して下さい。



6.9.2. <u>共通ライブラリ</u>

共通ライブラリは、CODESYS にデフォルトでインストールされており、プロジェクトに含まれているライブ ラリ群を指します。ユーザーが主に使用するライブラリは CANbus・Standard・SysCom・SysTime です。な お、ライブラリの内、RTC のみ当社独自のユーザー定義ライブラリ(Cx2500RtcLibrary)になります。

名称	摘要
3SLicense	システムライブラリ
BreakpointLogging	ブレークポイント用ライブラリ
CAA Device Diagnosis	デバイス診断用ライブラリ
CANbus	CAN 用ライブラリ(7.12 節参照)
IoStandard	システムライブラリ
Standard	IEC 標準コマンド用ライブラリ
SysCom	RS232C 用ライブラリ(7.11 節参照)
SysTime	タイマカウンタ・RTC 用ライブラリ(7.13、7.14 節参照)

Table 21 共通ライブラリー覧

6.9.3. <u>当社独自ライブラリ</u>

CX2500 の機能の内、RTC をユーザーアプリケーションで利用する際は当社独自ライブラリ(CX2500 Rtc Library)をプロジェクトに紐づける必要があります。利用する際は、6.9.5.2 項を参考にして CX2500 Rtc Library のインストールおよび紐づけを行ってください。



CCOT-24-016 Rev.1 6.9.4. ライブラリアイテムの使用方法

ここでは、Library Manager に紐づけられているライブラリファイル内のアイテムを POU に呼び出して処理 に使う方法を記します。例として、SysTime ライブラリの中にある SysTimeCore ライブラリを使用し、CX2500 が起動してからの経過時間(タイマカウンタ)を取得する Function を呼び出します。

	🗙 🙆 PersistentVars	POU_CtrlA	👩 GVL			•
🛃 Add Library 🔀 Delete Library 🛛 😁 Properties 👘 Details 🛛 🛤 Placeholders	👔 Library Repository 🕕 Ic	on Legend 🗎	Summary			00
Libraries used in application 'Device.Application'						
Name	Namesp	ace Effectiv	e Version			^
👾 📒 CANbus, * (3S - Smart Software Solutions GmbH)	CANbus	3.5.17.0	0			
IoStandard = IoStandard, 3.5.17.0 (System)	IoStanda	d 3.5.17.0	0			
Standard = Standard, 3.5. 18.0 (System)	Standard	3.5.18.0	0			
🖼 📃 SysCom = SysCom, 3.5.17.0 (System)	SysCom	3.5.17.0	0			
SysTime, * (System)	SysTime	3.5.17.0	0			
🕮 - 📒 SysTimeCore = SysTimeCore, 3.5.17.0 (System)	SysTimeC	ore 3.5.17.0	0			
🕮 🔚 SysTimeRtc = SysTimeRtc, 3.5.17.0 (System)	SysTimeR	tc 3.5.17.0	0			
FUNCTION Sy	/sTimeGetMs: UDINT					
FUNCTION SY	vsTimeGetMs: UDINT Name Type	Inhe	Address	Initial	Comment	
G STSTUPE FUNCTION Sy B SysTimeGetMs SysTimeGetMs SysTimeGetUs	Name Type SysTimeGetMs UDINT	Inhe	Address	Initial	Comment	
SysTimeGetMs SysTimeGetMs SysTimeGetUs SysTimeGetUs	rsTimeGetMs: UDINT Name Type SysTimeGetMs UDINT	Inhe	Address	Initial	Comment	
SysTimeGetMs SysTimeGetUs SysTimeGetUs	rsTimeGetMs: UDINT Name Type SysTimeGetMs UDINT	Inhe	Address	Initial	Comment	
Tig Styline FUNCTION Sy B SysTimeGetMs Function Sy B SysTimeGetUs ™	rsTimeGetMs: UDINT Name Type SysTimeGetMs UDINT	Inhe	Address	Initial	Comment	
'is stratement FUNCTION Sy Is SysTimeGetMs Is SysTimeGetMs Is SysTimeGetUs *	rsTimeGetMs: UDINT Name Type SysTimeGetMs UDINT	Inhe	Address	Initial	Comment	
SysTimeGetMs SysTimeGetMs SysTimeGetMs SysTimeGetMs SysTimeGetMs	rsTimeGetMs: UDINT	Inhe	Address	Initial	Comment	
SysTimeGetMs SysTimeGetMs SysTimeGetMs SysTimeGetUs	rsTimeGetMs: UDINT	Inhe	Address	Initial	Comment	
SysTimeGetMs FUNCTION Sy SysTimeGetMs SysTimeGetMs SysTimeGetUs SysTimeGetUs	rsTimeGetMs: UDINT	Inhe	Address	Initial	Comment	

Figure 57 Library Manager SysTimeCore ライブラリ情報

POU のエディタ画面で下記のように(ライブラリ名).(関数名)で記述して下さい。DUT などの他のアイテムでも、同様に(ライブラリ名).(アイテム名)で記述する必要があります。

/	PLC	C_PRG 🖷 Device 🍪 MainTask 🎁 Library Manager 🍯 PersistentVars 📄 POU_CtrlA 🗙 🎑 GVL	•
	8	RetainVal0: USINT := 0;	^ <u>b:</u>
	9	END VAR	
	10		
Β	11	VAR	
	12	TimeCnt : UDINT := 0;	
	13	END VAR	
	14		
	15		
	16		
	17		_
	18	100	R ~
	1	A 7	
	2	TimeCnt := SysTimeCore.SysTimeGetMs ():	
	-		
	4		
	5	GVL.g udVal0 := 100;	
	6		
	7		
	8	RetainVal0 := 100; //Call RETAIN value	
	9	PersistentVars.PersistVal0 := 200; //Call PERSISTENT value	
			100 බ
			100

Figure 58 POU エディタ ライブラリ関数の呼出



6.9.5. ユーザー定義のライブラリ

ユーザー定義のライブラリとは、ユーザーが自ら作成したライブラリのことを指します。ライブラリファイル 内のアイテムの呼び出す方法は後述します。なお、ユーザー定義ライブラリのアイテム使用方法は 6.9.4 項と同 様です。

6.9.5.1. ユーザー定義ライブラリの作成

ここでは、ライブラリの作成方法を記します。

① メニューバーの「File」→「New Project..」を選択して下さい。



Figure 59 メイン画面 New Project の選択

KEIKI

CCOT-24-016 Rev.1

② 「New Project」ウィンドウにて下記の設定をおこなった後、「OK」ボタンを押して下さい。

🖹 New Pro	ject			
Categories		Templates		
	oraries Diects	`	`	-
	ojecta	CODESYS container library	CODESYS interface library	CODESYS library
		•	۰	
		Empty library	External CODESYS library	
		<		>
A new COD	ESYS library with coding rule	compliant structures		
Name	TK_SAMPLE_LIBRARY			
Location	C:¥Users¥dengi¥Desktop			~
			OK	Cancel



項目	設定値
Category	「Libraries」を選択する。
Template	「CODESYS library」を選択する。
Name	ライブラリ名(ユーザー任意)を記入する。
Location	プロジェクトの保存先(ユーザー任意)を「…」ボタンを押して選択する。

Κεικί

CCOT-24-016 Rev.1

③ ライブラリが作成され、ライブラリウィンドウが表示されます。編集の際は下図赤枠の「POUs」タブをクリ ックすると編集リストが表示されます。

TK_SAMPLE_LIBRARY.library - CODESYS		- a ×
File Edit View Project Build Online Debug Tools Window Help		₹2
○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○	▶ ■ 🖑 (1 = 0 = 1 = 2 = 3 0 1 = 1 = 3)	
Vols • a × Image: Comparison • a × Image: Library Manager • a • a × Image: Decision of the second se		
Messanes - Total 1 error(s), 0 warning(s), 0 messane(s)		X
Library Manager • O 1 error(s) • 0 warning(s) • 0 message(s) × 🕅		
Description		Project Object Position
Could not open library '#CBML'. (Reason: The library 'Common Behaviour Model, 3.5.12.0 (3S - Smart Software Solutions GmbH)' has not bee	n installed to the system.)	TK_SAM Library
	Last build: 🔿 0 😗 0 🛛 Precompile 🧹	Project user: (nobody) 🛛 🖗 🐼

Figure 61 ライブラリ 編集画面



 ④ ライブラリウィンドウから、「Project Information」をダブルクリックします。すると、「Project Information」 ウィンドウが表示されるので、下記の通りライブラリ情報を入力して下さい。これは、Library Manager で 当該ライブラリを選択した時に表示されるライブラリ情報です。
 1 カ第一「OK」 ボタンた押して下さい。

入力後、「OK」ボタンを押して下さい。

Project I	nformation							\times
File	Summary	Prope	erties	Statistics	Licensing			
Соп	npany		TOKY	O KEIKI I	NC.			
Title	2		TK_S	AMPLE_L	IB			
Ver	sion		3.5.1	7.0			Re	leased
Libr	ary Categor	ies						
Defa	ault namesp	ace	TK_SA	AMPLE_LIB				
Plac	eholder		TkSan	npleLib				
Auth	or		TOKY	O TAROU				
Des	cription		Samp	le library fi	or programn	ning mar	nuals	< >
The	fields in bo	ld lett	ers are	used to id	dentify a lib	irary.		
Auton	natically gen	erate '	Library	/ Informati	on' POUs			
Auton	Automatically generate 'Project Information' POUs							
					OK	(С	ancel

Figure 62 Project Information ウィンドウ

		-		
÷	TkSampleLib = TK_SAMPLE_LIB, 3.5.17.0 (TOKYO KEIKI INC.)	TK_SAMPLE_LIB	3.5.17.0	0

Figure 63 Figure 62 で設定した際の Library Manager 上での表示

項目	設定値
Company	ユーザーの社名等を入力する。
Title	ライブラリ名を入力する。
Version	ライブラリのバージョンを入力する。
Library Categories	設定不要。
Default namespace	ライブラリ名等を入力する。
	(アイテムを呼び出すときに必要になる。)
Placeholder	ライブラリ名を入力する。
Author	ユーザーの名前等を入力する。
Description	ライブラリに関する説明を入力する。
Automatically generate 'Library Information' POUs	無効化(チェックを外す)
Automatically generate 'Project Information' POUs	無効化(チェックを外す)

Table 22 Project Information ウィンドウ 設定項目(値は全て任意値)



⑤ ここからは、アイテム(POU や DUT 等)をライブラリ上で作成して下さい。ライブラリに入れたいアイテム を作成して下さい。作成方法はアプリケーションプロジェクトでおこなうときと同様です。ここでは例とし て構造体 ST_SMP_LIB と POUFB_SMP_LIB(POU、Function block)を作成しました。

POUs 👻 🕂 🗙	👔 Library Manager 🎑 Constants 🖓 ST_SMP_LIB 🗙	- /	PO	UFB_SMP_LIB X					•
TK_SAMPLE_LIBRARY	1 TYPE ST_SMP_LIB :	->	\$	• X	FUNCTION	BLOCK PUB	IC POUFB_SM	IP_LIB	15
Constants Coolerants Cooler	<pre>1 TIPE SI_SME_LLB : 2 STRUCT 3 usTestMem : USINT; 4 ulTestMem : ULINT; 5 END_STRUCT 6 END_TYPE 7</pre>	<	1 2 3	X Scope VAR_INPUT VAR_INPUT VAR_OUTPUT VAR_OUTPUT	FUNCTION Name ucInput0 ucInput1 bOutput	Address	IC POUFB_SN Data type USINT USINT BOOL	Initialization	C
ST_SMP_LIB (STRUCT) ST_SMP_LIB (STRUCT) Source Structure Toples Structure Project Information Project Information Project Settings	100		34567	IF (uclnput) + bOutput := ELSE bOutput := END_IF	ucInput1) TRUE; FALSE;	>= 100 THE	3		100

Figure 64 ライブラリ編集例 構造体と Function block の追加

⑥ アイテムを一通り作成し終えたらビルドをおこなって下さい。メニューバーの「Build」→「Check all Pool Objects」を選択して下さい。

TK_SAMPLE_LIBRARY.library* - CODESYS File Edit View Project Libraries E	uild Online Debug Tools Window Help		_	□ × ▼2
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	Check all Pool Objects F11 □* □# ■<	4≣ +≣ 8 ¢ ≅ 7 7⁄2		
POUs 🖵 🕂	Generate Runtime System Files			-
TK_SAMPLE_LIBRARY	Clean Details 🕞 Placeholders 👘 Library Repository (🕦 Icon Legend 🚔 Summary		0 0
🕸 🗀 Libname	Clean all			
Library Manager				
- Project Information - Project Settings	Name Mamesee Eff Mamespace Eff Mamespace Eff Mamespace Eff Mamespace Eff Mamespace Eff Mamespace Eff Mamespace Eff Mamespace Eff Seff Status CAA Toppes – CAA Types Extern, 3.5.17.0 (CAA Technical Workgroup) CAA 3.5 CBML – Common Behaviour Model, 3.5.17.0 (DS - Smart Software Solutions GmbH) CBML 3.5 CBML – Common Behaviour Model, 3.5.17.0 (DS - Smart Software Solutions GmbH) CBML 3.5 CBML – Common Behaviour Model, 3.5.17.0 (DS - Smart Software Solutions GmbH) CBML 3.5 Mamespace Eff Software Solutions GmbH (DS - Smart Software Solutions GmbH (DS - Sm	Tective Version 5.19.0 0 5.17.0 0 5.17.0 0		
Cevices POUS				
Messages - Total 1 error(s), 0 warning(s), 0 message	(5)			→ ∓ ×
Build	O error(s) O warning(s) O message(s) X X			
Description Build started: Application: Typfry code Comple complete 0 errors, 0 warnings			Project Object	Position
	Last build: 🔿 0 👩	0 Precompile 🗸 Pro	iject user: (nobody)	Ø 🕸

Figure 65 ライブラリのビルド

⑦ ビルド後、エラーが無いことを確認します。エラーが出た場合はその情報に従って修正し、再度ビルドして下さい。



⑧ CODESYS-IDE のライブラリリポジトリにライブラリを登録します。ライブラリリポジトリは CODESYS-IDE で使用するライブラリを登録・保存しています。 リポジトリに登録され、プロジェクトへの紐づけが可能になります。

TK SAMPLE LIBRARY.library* - CODESYS						_	o ×
File Edit View Project Libraries E	Build Online Debug Tools Window Help						₹2
🎦 🖨 🔚 🗿 🕼 🖂 🕷 🛍 🗙	() 🏘 🍇 🍇 [] [1] 1] 1] 1] 1] 18 [18] 18] 18] 18] 18] 18] 18] 18] 18] 18]	• ■ № 〔=	G∃ d∃ →∃ \$ d	罰 = 型			
POUs Save Project and Install into Libra	ry Repository ry Manager X		0 · · · ·	A			•
TK_SAMPLE_LIBRARY	Add Library X Delete Library Properties Details Add Library Placeholders	Library Reposit	tory 🕕 Icon Legend	😑 Summary			00
Library Manager	Libraries used in [Pool]						
Project Information	Name	Namespace	Effective Version				
Project Settings	CAA FB Factory = CAA FB Factory, 3.5.19.0 (CAA Technical Workgroup)	FBF	3.5.19.0	•			
	CAA Types = CAA Types Extern, 3.5.17.0 (CAA Technical Workgroup)	CAA	3.5.17.0				
	CBML = Common Behaviour Model, 3.5.17.0 (3S - Smart Software Solutions GmbH)	CBML	3.5.17.0				
	•						
< > >							
POUs							
Messages - rotal Lerror(s), 0 warning(s), 0 message							• + •
Build	• O error(s) • O warning(s) • O message(s)					_	
Description					Project	Object	Position
Build started: Application:							
Typity code							
comple complete 0 errors, 0 warnings							
		Last build: 🧲	🕽 U 🕐 U Precompi	ie 🧹	Project user: (noi	body)	🕡 🐼

Figure 66 ライブラリリポジトリへの登録



CCOT-24-016 Rev.1 6.9.5.2. ユーザー定義ライブラリの紐づけ

ここでは、ユーザー定義ライブラリのプロジェクトへの紐づけを記します。6.9.5.1 項と異なり、ライブラリリ ポジトリへの登録が済んでいない場合(違う PC の CODESYS でライブラリを作成していた場合等々)、 CODESYS-IDE のライブラリリポジトリにライブラリを登録する必要が有ります。既にライブラリリポジトリ への登録が完了している場合は手順⑤から始めて下さい。

① メニューバーから「Tools」 \rightarrow 「Library Repository…」を選択して下さい。

SampleProject_Cx2500.project* - CODES	15						- 0 -
File Edit View Project Libraries	Build Online Debug	Tools Window Help					₹
) 📽 🔛 I 🚳 I 🗠 🖓 🐘 🛍 🗙	🗛 🍪 🍓 🚰 🏢 🐄	CODESYS Installer	C Logic] 🝷 😘 🕬	→ = % 0	1910; *18 •	> 熙 글 캇	
		👔 Library Repository					
evices 🗸 🗸 🖡	🗙 📝 Device 🧕	Device Repository	er 🗙 📄 POU_Ctrl	A TI PLC	PRG 🍪 MainTas	*	
SampleProject_Cx2500	 Add Library × Dele 	Visualization Style Repository	rs 🛛 🎁 Library Repos	itory 🕕 Icon Le	egend 🖹 Summary		o
Device (CX2500)	Libraries used in applicati	Lisanse Peneritany	-				
B II PLC Logic	Name	CDC LIA Information Model Presentation		Namespace	Effective Version		
= O Application	CAA Device Diago	OPC OA Information Model Repository	(mar)	DED	2.5.19.0	1	
DUT_SAMPLE	B-I CANbus, * (35 - S	License Manager	proop)	CANbus	3.5.17.0		
GVL	IoStandard = IoSt	Device License Reader		IoStandard	3.5.17.0		
Di C PPG (PPG)	Standard = Stand	Customize		Standard	3.5.18.0	•	
	B- SysCom = SysCon	Options		SysCom	3.5.17.0	•	
Task Configuration	🖻 - 📒 SysTime, * (Syste	Import and Export Options		SysTime	3.5.17.0	•	
🖷 🕸 MainTask							
- DLC_PRG		Edge Gateway	,				
- DOU_CHA		Miscellaneous	•				
Task1		Scripting	•				
T PersistentVars							
- 👔 Cx2500Ain (Cx2500Ain)							
🚮 Cx2500Cntin (Cx2500Cntin)							
Cx2500Din (Cx2500Din)							
Cx2500Do (Cx2500Do)							
Cx2500Frgin (Cx2500Frgin)							
Cx2500Pwm (Cx2500Pwm)							
Paulas Paula							
pevices 10 Pous							
lessages - Total 0 error(s), 0 warning(s), 0 me	sage(s)						- ₽
	 O error(s) 	😗 0 warning(s) 🚯 0 message(s) 🗙 💥					
Description					Project	Object	Position

Figure 67 メイン画面 Library Repository の選択

② 「Library Repository」ウィンドウが表示されるので、「Install」ボタンを押します。

前 Library R	epository	×
Location	System V (C:¥ProgramData¥CODESYS¥Managed Libraries)	Edit Locations
Installed L Company	Varies (All companies) V Miscellaneous) pplication	Install Uninstall Export
	nces ntern ystem 'est Versions (not recommended) Ise Cases	Find
≺ ✓ Group	by category	Details Trust Certificate Dependencies
Library P	rofiles	Close

Figure 68 Library Repository ウィンドウ



③ 「Select Library」ウィンドウが表示されるので、登録したいライブラリファイルの置き場所を探し、ファイルを選択した上で「開く」ボタンを押して下さい。この例では、登録したいライブラリファイルをデスクトップに置いています。

E * 新しいフォルダー E * 1 まで宣言	→ × ↑ 🗖 > PC > Desktop >			✓ ひ ○ ○ De	sktopの検索
 表で宣言 へ 名前 2 町日時 種類 サイズ OneDrive PC 3D オブジェクト TK_SAMPLE_LIBRARY.library 2024/02/28 11:42 CODESYS library 296 KB TKSampleLibrary0.library 2024/02/28 10:47 CODESYS library 154 KB グウンロード デキュメント ビクチャ ビデオ S1-ジック Windows (C) Windows (C) 	星 ▼ 新しいフォルダー				8== •
 ③ 3D オブジェクト ● TK_SAMPLE_LIBRARY.library 2024/02/28 11:42 CODESYS library 296 KB ● TkSampleLibrary0.library 2024/02/28 10:47 CODESYS library 154 KB 使 グラカ ビ グラカ ビ グラカ ビ グラカ ビ グラカ ビ ジカ シ ミュージック Windows (C) 	_ 表で宣言 ^ 名前 ^ ● OneDrive ■ PC	更新日時	種類	サイズ	
 Desktop ● TkSampleLibraryQ.library 2024/02/28 10:47 CODESYS library 154 K8 ● ダウンロード ● ダウンロード ● ゲキュント ■ ビラオ ■ ビラオ ● ジラブク ■ ビデオ ● Windows (C:) ● Windows (C:) ● マークロー・ 	■ 3D オブジェクト ● TK_SAMPLE_LIBRARY.library	2024/02/28 11:42	CODESYS library	296 KB	
	 Desktop ▼ガンロード ドキュメント ビクチャ ビブオ トニージック 	2024/02/28 10:47 種類: CODESYS library サイズ: 295 KB 更新日時: 2024/02/28 11:42	CODESYS library	154 KB	

Figure 69 Select Library ウィンドウ ライブラリの選択



④ 「Library Repository」上にライブラリが登録されます。これでライブラリリポジトリへのライブラリの登録は完了です。

iii Library Repository	×	💼 Libra	rary Repository	×
Location System (C:\#ProgramData\CODESYS\Managed Lbraries)	✓ Edit Locations	Locati	tion System v (C:WrogramData¥CODESYSW4anaged Libraries)	Edit Locations
Instaled Libraries Company (Al companies) Company (Al companies) Company Application Company Intern Company Control (Control (Contr	Instal Uninstall Export Find Detais Trust Certificate Dependencies	Insta Com	aled Lbrares npamy (All comparies)	Instal Uninstall Export Find Details Trust Certificate Dependencies
Library Profiles	Close	Libr	vary Profiles	Close

Figure 70 Library Repository ウィンドウ 追加されたライブラリの確認

⑤ ここからは、アプリケーションプロジェクトへのライブラリの紐づけをおこないます。アプリケーションプロジェクトにて、デバイスウィンドウの Library Manager をダブルクリックし Library Manager 画面を開いて下さい。

SampleProject_Cx2500.project* - CODESYS	- 0	\times
File Edit View Project Libraries Build Online Debug Tools Window Help		2
11 🚔 🔲 🚔 🗠 🖂 🕹 📾 🕮 🗙 🖓 🌿 🏘 🌿 👫 🐫 💷 🦄 🏙 🕼 👘 🖒 🖽 Application (De	ice: PLC Logic] • 😻 🚳 🖒 🔳 📽 🗐 🗐 🖕 💷 🕄 🗠 🛒 🐨 🐨	
	A A A B BIA AI INFLAIA	
		-
Devices • 4 × 12 No Device to Persistentivars to Gu. 211 Library		
SampleProject_CX2500 Add Library > Delete Library > Properties (a) Details (a) Place Add Library > Delete Library > Properties (a) Details (a) Place	molders 📳 Library Repository 🕡 Icon Legend 🖻 Summary	
Libraries used in application 'Device.Application'		
Replication Name	Namespace Effective Version	^
DUT_SAMPLE JSLicense = 3SLicense, 3.5.18.0 (3S - Smart Software Solutions GmbH	_35_LICENSE 3.5.18.0	
	rt Software Solutions GmbH) BPLog 3.5.17.0 🔮	
- 👔 Library Manager CAA Device Diagnosis = CAA Device Diagnosis, 3.5.18.0 (CAA Technic	il Workgroup) DED 3.5.18.0 🗣	
PLC_PRG (PRG)	CANbus 3.5.17.0	
POU_CtriA (PRG)	IoStandard 3.5.17.0	
🖙 🧱 Task Configuration 🦳 🔚 Standard = Standard, 3.5.18.0 (System)	Standard 3.5.18.0 0	~
🗟 🥸 MainTask		
- IC_PRG		
@] POU_CHA		
Task1		
T PersistentVars		
Cx2500Pwm (Cx2500Pwm)		
👷 Devices 👔 POUs		
Maecanae - Total O error(e) O warning(e) O maecana(e)	_	. .
	•	^
• O denor(s) • O warning(s) • O message(s)		
Description	Project Object Position	
	Last build: 🗿 0 🕐 0 Precompile 🧹 🥳 Project user: (nobody) 🕖	2

Figure 71 メイン画面 Library Manager の選択



CCOT-24-016 Rev.1 ⑥ Library Manager 画面にて、「Add Library」ボタンを押して下さい。

POU_Ctr		PRG 🍪 MainTask	•
🛨 Add Library 🔀 Delete Library 🖙 Properties 🟐 Details 🕼 Placeholders 🎁 Library Repo	sitory 🕕 Icon Le	gend 📋 Summary	0 0
Libraries used in application 'Device.Application'			
Name Add Library	Namespace	Effective Version	^
☞ 3SLicense = 3SLicense, 3.5.18.0 (3S - Smart Software Solutions GmbH)	_3S_LICENSE	3.5.18.0	
BreakpointLogging = Breakpoint Logging Functions, 3.5.17.0 (3S - Smart Software Solutions GmbH)	BPLog	3.5.17.0	
CAA Device Diagnosis = CAA Device Diagnosis, 3.5.18.0 (CAA Technical Workgroup)	DED	3.5.18.0	
🕀 - 📒 CANbus, * (3S - Smart Software Solutions GmbH)	CANbus	3.5.17.0 0	
🕮 📃 IoStandard = IoStandard, 3.5.17.0 (System)	IoStandard	3.5.17.0	
Estandard = Standard, 3.5.18.0 (System)	Standard	3.5.18.0	~

Figure 72 Library Manager Add Library の選択

⑦ 「Add Library」ウィンドウが表示されるので、下記のように紐づけたいライブラリ名を入力してライブラリ を選択します。選択した状態で「OK」ボタンを押して下さい。

Add Library		×
TK_SAMPLE		
Match	Library	
····[TK_SAMPLE_LIB		
	OK	Cancel

Figure 73 Add Library ウィンドウ ライブラリの検索



⑧ Library Manager 上にライブラリが表示されれば紐づけは完了です。紐づけ後、ライブラリのアイテムを使用できます。

📊 Device 🏾 🎑 PersistentVars 🖉 GVL 🎢 Library M	ager 🗙 📄 POU_CtriA 📄 PLC	PRG 🔮 MainTask	
🗄 Add Library 🗙 Delete Library 🖙 Properties 🗃 Details 🗐 Place	ders 🎁 Library Repository 🕕 Icon L	Legend 🗎 Summary	0 0
Libraries used in application 'Device.Application'			
Name	Namespace	Effective Version	1
Image: Bernard Canalysis and Canalysis a	CANbus	3.5.17.0 🟮	
🗟 - 📙 IoStandard = IoStandard, 3.5.17.0 (System)	IoStandard	3.5.17.0	
— E Standard = Standard, 3.5.18.0 (System)	Standard	3.5.18.0	
SysCom = SysCom, 3.5.17.0 (System)	SysCom	3.5.17.0	
P_I_ SucTime * (Suctors)	SugTime	25170	
Image: TkSampleLib = TK_SAMPLE_LIB, 3.5.17.0 (TOKYO KEIKI INC.)	TK_SAMPLE_L	.IB 3.5.17.0 🚯	
Control Research Structs Control Research Control R	POUFB_SMP_LIB USINT BOOL bOutput		

Figure 74 Library Manager ライブラリ追加後

	ľ	S	PersistentVars	🧭 GVL 👔	Library Mana	ager	POU_CtrlA		PLC_PRG	🍪 MainTask	📝 👔 P(U_FB >	•	Tool	Box	-	ņ	×
	۷.	-	PROGRAM POU_FB							b	Ξ (Gener	al		^			
ľ		^	Scope	Name	Address	Data tvi	pe		Initialization	Comment	Attributes				*	Network		
I		1	A VAP	ucVal0		LISTNE			10						- 😰	Box		
I		2 VAR ucVali						90							Box with EN/ENO			
I	3 A VAR bResult				BOOL			FALSE						-VAR	Assignment			
I					TK SAM	PLE LIB.POUFB SMP I	IB							->	Jump			
I			• • • • • •												RET	Return		
I.														-	Input			
ŀ													t	Branch				
		1		POUF	B_SMP_LIB	_0									12	Execute		
				TK_SAMPLE_I	LIB.POUFB	SMP_LI	в								Boolea	an Operators		
			ucVa	10 — ucInput0		bOutp	ut bResult								-81	AND (2 Inputs)		
			ucVa	11 — ucInput1											8	AND (3 Inputs)		
															-21	OR (2 Inputs)		
ľ															21	OR (3 Inputs)		
															1	XOR (2 Inputs)		
L														81	Math (Operators		
L															1	ADD (2 Inputs)		
																ADD (3 Inputs)		
															- 🖃	SUB		
L															×	MUL		
										[► + Q	100 %				DIV		
	c												>		1	EQ		¥

Figure 75 POU エディタ ライブラリ関数(Function block)の呼出例



6.10. ビルド

IEC アプリケーションを作成して CX2500 に書き込む前に、必ずビルド(コード生成)をおこなう必要が有ります。

ビルドは、CODESYS メイン画面の「Build」タブの「Generate Code」からできます。ビルドしてエラーが出た場合はエラー内容に沿って修正して下さい。エラー無しの場合、CX2500 にアプリケーションを書き込むことができます。

「Build」タブ・コンテキストメニューの各要素については下表を参照して下さい。



Figure 76 ビルド(コード生成)

Table 23	ビルドメニュー
----------	---------

摘要
IEC アプリケーションのビルド(コード生成)をおこなう。
プロジェクト内のアプリケーションの内、アクティブになっているアプリケーションの前回ビ
ルド情報をクリアする。
プロジェクト内の全てのアプリケーションのビルド情報をクリアする。

※10 Clean や Clean all は、主にデバイス定義ファイルを更新した時(Setup 編参照)等に使用する。



CCOT-24-016 Rev.1 6.11. ログイン・ログアウト

IEC アプリケーションを作成しビルドまで完了したら、CX2500 にアプリケーションを書き込むことができま す。アプリケーションを書き込んでデバッグをおこなっている状態のことをログインと言います。一方、そのロ グイン状態を終了(CODESYS-IDE と CX2500 との接続を終了する)することをログアウトと言います。

6.11.1. <u>ログイン手順</u>

ここではログインの手順を示します。

- ① 別紙 Setup 編の「デバイスの接続」を参照し、CX2500 と CODESYS-IDE を接続した状態にして下さい。
- ② メニューバーの「Online」タブから「Login」を選択して下さい。

i 🚅 📕 🚑 🗠 🗠 👗 🛍	0	Login	Alt+F8	🖽 Application [Device: PLC Logic] 🔹 🧐 📦 🖬	V [II 9] 6] 7] 8 [4] 50 [7] 8		
Herices 4 3 SampleProject_Cr/2500 Device (Cr/2500) Device (Cr/2	~	Logout Create Boot Application Download Multiple Download Reset Varm Reset Cold Reset Cold Reset Origin Simulation Security Assign Server Applications on PLC Stetlings	Ctrl+F8	Gateway • Device • Gateway • Device • Gateway Gateway-1 IP-Address: Iocalhost Port: 1217	MyDevice (active) Device Name: MyDevice 301.1001		
≥ Devices ID POUs	>	Users and Groups Access Rights Symbol Rights Licensed Software Metrics Task Deployment Status Information			Target ID: 17E5 0001 Target Type: 4096 Target Vendor: TOKYO KEIKI INC. Target Version: 3.5.18.30		
fessages - Total 0 error(s), 0 warning(s), 0	messa	ige(s)	0 warning(s)	0 message(s) 🗙 💥			▼ ₽
Description					Project Object	Position	

Figure 77 Login の選択



③ 下記のようなアプリケーション書き込みの確認ウィンドウが表示されますので、「Yes」ボタンを押して下さい。なお、既に書き込もうとしているプロジェクトのアプリケーションと同じアプリケーションが CX2500に書き込まれている場合、このウィンドウは表示されずに手順⑤へ遷移します。

Devices 👻 🕈 🗙	Device X			•					
SampleProject_Cx2500	Communication Settings Scan Network Gateway + Device +								
PLC Logic	Applications								
Library Manager	Backup and Restore								
Task Configuration	Files								
⊨ S MainTask	Log	Gateway							
	PLC Settings	CODESYS	Yupevice (active) Vevice Name:						
	PLC Shell	Warning: An application 'Application' is currently in RUN mode on the PLC. As there is no matching compile information, this existing application needs to be contracted.	lyDevice Device Address:						
	Users and Groups	Click 'Yes' to download the latest code or 'No' to abort.	301.1001						
	Access Rights	Yes No Details	.7E5 0001						
	Symbol Rights		- larget Type: 						
	Licensed Software Metrics		TargetVendor: TOKYO KEIKI INC.						
	Task Deployment		Target Version: 3.5.18.30						
	Status								
A Devices POUs	Information			~					
Messages - Total 0 error(s), 0 warning(s), 0 m	nessage(s)		•	д х					
	- O error(s) 🕐 0 warning(s) 🚺 0 message(s) 🗙 🔆							

Figure 78 書き込み確認ウィンドウ

④ アプリケーション書込みが始まりますので、書き込みが終了するまで待機します。

Devices 🗸 4 X	Device X		
 ■ SampleProject_0.2000 ■ Device (connected) (CI2000) ■ PIC togic ■ PIC togic ■ Device (CPRC) ■ Device (CPRC)<th>Communication Settings Applications Backup and Restore Backup and Restore Log PLC Settings PLC Settings PLC Shell Users and Groups Access Rights Symbol Rights Licensed Software Metrics Task Deployment Satus</th><th>Scan Network Gateway Device Gateway Gateway</th><th></th>	Communication Settings Applications Backup and Restore Backup and Restore Log PLC Settings PLC Settings PLC Shell Users and Groups Access Rights Symbol Rights Licensed Software Metrics Task Deployment Satus	Scan Network Gateway Device Gateway	
2 Devices POUs	Information		
fessages - Total 0 error(s), 0 warning(s), 7 me	essage(s)	warning(s) 🛛 7 message(s) 🗙 💥	- ų ;

Figure 79 書き込み中画面



⑤ 以下のようなデバッグ画面に遷移するとアプリケーション書込み(ログイン)は完了です。ログイン中のデバ ッグについては9章を参照下さい。



Figure 80 書き込み終了後の画面

赤枠内のデバッグ関連のアイコンボタンが有効化される

6.11.2. <u>ログアウト手順</u>

ログアウトは、メニューバーの「Online」タブから「Logout」を選択して下さい。若しくは、Figure 80 のようなログイン中の画面で、ツールバーのログアウトアイコン「[♥]」を押すことでもログアウトができます。なお、 CX2500 内のアプリケーション運転状態はログアウト時点での状態を維持することに留意して下さい。



7. <u>CX2500 の機能について</u>

7.1. CX2500 機能一覧

CX2500の機能一覧は下表の通りです。各機能の使い方は関連の節を参照して下さい。

項目	機能名	チャネル数	参照先	備考
入力	<u>デジタル入力</u>	23	7.3 節	-
	イグニッション入力	1	7.3 節	-
	<u>周波数入力</u>	8	7.4 節	-
	2 相カウンタ入力	8	7.5 節	-
	<u>アナログ入力</u>	30	7.6 節	-
	<u>内部電源電圧監視入力</u>	-	7.7 節	-
	基板温度監視入力	3	7.8 節	-
出力	<u>デジタル出力</u>	16	7.9 節	-
	<u>PWM 出力</u>	10	7.10 節	-
通信	<u>RS232C</u>	2	7.11 節	
	CAN	5	7.12 節	Ch.3、4 のみ通信速度 250kbps 以下。
その他	タイマカウンタ	_	7.13 節	-
	RTC	1	7.14 節	-

Table 24 CX2500(CODESYS 版) 機能一覧



CCOT-24-016 Rev.1 7.2. 機能ドライバについて

機能ドライバとは、CX2500に搭載されている以下の入出力機能の制御をおこなうためのモジュールです。 下記機能を使用する際は、「CX2500Codesys_UserManual_ForSetup」の 5.2 節にあるようにプロジェクトの デバイスに紐づける必要があります。各機能ドライバの共通する設定については 7.2.1 項以降に纏めています。 各ドライバの設定画面については、下図赤枠のように表示されているドライバをそれぞれダブルクリックす ると表示されます。

区分	機能名	対応機能ドライバ名		
入力	<u>デジタル入力</u>	Cx2500Din		
	イグニッション入力			
	周波数入力	Cx2500Frqin		
	2 相カウンタ入力	Cx2500Cntin		
	<u>アナログ入力</u>	Cx2500Ain		
	内部電源電圧監視入力			
	基板温度監視入力			
出力	<u>デジタル出力</u>	Cx2500Do		
	PWM 出力	Cx2500Pwm		

.ces 🗸 4 🗙	Library Manager	PLC_PRG Device X	MainTask 🔐 POU_FB 📄 P	OU_CtrlA 🛛 😓 Task1	Ta:▼	ToolBox	▼ ₽
SampleProject_Cx2500	Communication Settings Applications	Scan Network Gateway ~	Device ~		Î		
GUL GUL Turary Manager PLC_PRG (PRG) POU_FR (PRG) GU POU_FR (PRG) GU FAL Configuration GU PALTARK GU PLC_PRG	Files		Gateway	•••			
	PLC Settings		<simulation mode=""></simulation>	${\scriptstyle \smallsetminus}$ ${\scriptstyle < simulation mode >}$			
	PLC Shell						
- ④ POU_CtrlA = 🕸 Task1	Users and Groups						
POU_FB	Access Rights						
Cx2500Ain (Cx2500Ain) Cx2500Cntin (Cx2500Cntin)	Symbol Rights						
Cx2500Din (Cx2500Din)	Licensed Software Metrics						
Cx2500Frqin (Cx2500Frqin)	Task Deployment Status						
evices POUs	Toformation C				>		
	e(s)						~ 9
	E(S)						•

Figure 81 機能ドライバの紐づけ



CCOT-24-016 Rev.1 7.2.1. <u>機能ドライバ画面</u>

各機能ドライバは、以下の4つのタブ(機能によってはInternal Parameters タブが無い)で構成されています。 これらのタブの内、Internal Parameters タブ(7.2.2 項)と Internal I/O Mapping(7.2.3 項)タブはユーザーが設 定・制御に使うことになります。

ernal Parameters	Find		Filter Sho	w all			 Add FB for 	IO Char	nnel → Go to Instance
anal I/O Mapping	Variable		Mapping	Channel	Address	Туре	Default Value	Unit	Description
	🖃 🧤			DinInputState	%IB89				Input status of Ch.0-22 state of it cha
tus	🖨 - 🏘)		Ch0_7	%IB89	BYTE			
		🍫 Di0	***	Ch0	%IX89.0	BOOL			TRUE:ON, FALSE:OFF
ormation		Application.PLC_PRG.Di1	~	Ch1	%IX89.1	BOOL			TRUE:ON, FALSE:OFF
		🍫		Ch2	%IX89.2	BOOL			TRUE:ON, FALSE:OFF
		🦄		Ch3	%IX89.3	BOOL			TRUE:ON, FALSE:OFF
		🦄		Ch4	%IX89.4	BOOL			TRUE:ON, FALSE:OFF
		🦄		Ch5	%IX89.5	BOOL			TRUE:ON, FALSE:OFF
		🍫		Ch6	%IX89.6	BOOL			TRUE:ON, FALSE:OFF
		🍬		Ch7	%IX89.7	BOOL			TRUE:ON, FALSE:OFF
	😟 🗄 ᡟ)		Ch8_15	%IB90	BYTE			
	😟 ᡟ)		Ch16_22	%IB91	1 BYTE 2.0 BOOL			
	- L 🍫			IgnInputState	%IX92.0				Input status of ignition. TRUE:ON, FA
	<	<							
			Reset Map		ping Al	ways updatev	ariables Enabled	l 1 (use b	ous cycle task if not used in any task)
	* = Create new variable * = Map to existing variable								

Figure 82 機能ドライバ画面

Table 26 機能ドライバ タブ一覧

タブ名称	摘要
Internal Parameters	機能の初期設定をおこなう必要がある場合に表示されるタブ。
	(例)デジタル入力の入力形式選択、PWM 出力のディザ周波数 etc.
Internal I/O Mapping	機能の制御、機能の制御中の状態モニタに使う。
Status	機能ドライバの稼働状態を確認することができる。稼働状態の場合は「Running」と表示
	される。
Information	機能ドライバの情報(ドライバのバージョン等)を確認できる。



7.2.2. Internal Parameters タブ

Internal Parameters タブでは、機能の初期設定をおこなう必要のある要素がある場合のみ表示されます。 ユーザーはこのタブにある要素の設定を必ずおこなって下さい。設定は、ビルド前に各要素の Value 列に所望 の値を入力するだけです。設定値が TRUE 若しくは FALSE を設定する場合は、要素の Value 列をクリックす ることで TRUE/FALSE を切り替え・選択することができますユーザーが Value 列で値を設定しない場合、各要 素の設定値は「Default Value」列の値になることに注意して下さい。

各機能の Internal Parameter タブの解説ついては、後述する各節を参照して下さい。

ternal Parameters	Parameter	Туре	Value	Default Value	Unit	Description	
PWM frequency		UINT(501000)	200	200	Hz	Frequency for PWM control(= PWM dither). Settable range:50-1000[H	
ernal I/O Mapping	🖶 🖗 PID parameters					PID parameters for each channel.	
Status	🖹 🖗 Ch0						
	🦇 🌾 Кр	UINT	200	200		PID-P value	
nformation	🏟 Ki	UINT	50	50		PID-I value	
	🧼 🖗 Kd	UINT	500	PID-D value			
	🖻 🖗 Ch1						
	🖻 🖗 Ch3						
	🗎 🖗 Ch4						
	🗎 🖗 Ch5						
	🗎 🖗 Ch6						
	🗎 🖗 Ch7						
	🕀 🖗 Ch8						
	😟 🖗 Ch9						
	😑 🛛 🧼 Enable DO mode					Bitfield mean DO mode enable/disable.	
	EnableDoModeCh0_7	BYTE	0				
	EnableDoModeCh8_9	BYTE	0				
	🖗 Ch8	BOOL	FALSE	FALSE		TRUE:DO mode, FALSE:PWM mode	
	ch0	BOOL	EALSE.	EALSE		TRUE-DO mode EALSE-PWM mode	

Figure 83 機能ドライバ Internal Parameters タブ選択時の画面

Table 27	Internal	Parameters	タブ	各列の概要
----------	----------	------------	----	-------

項目	ユーザー編集可否	摘要
Parameter	×	各要素の名称が表示されている。機能にチャネル毎に設定する要素がある
		場合はチャネル番号も表示されている。
Туре	×	各要素に設定する値のデータ型が表示されている。
Value	0	ユーザー設定値。Value にユーザーが設定した値が、アプリケーション起動
		時に各要素に設定される。
Default Value	×	各要素のデフォルト値が表示されている。ユーザーが Value を設定しない場
		合はこの値が要素に設定される。
Unit	×	各要素の単位が表示されている。単位が無い場合は空欄。
Description	0	各要素の説明が表示されている。



7.2.3. Internal I/O Mapping タブ

Internal I/O Mapping タブは、機能の制御・状態モニタをおこなうために使われます。本タブで設定する必要 がある項目については 7.2.3.1 項以降に記しています。

internal Parameters	Find		Filter Show all		- ↓ A	dd FB for IO Char	nnel	Go to Instance
internal I/O Mapping	Variable	Mapping	Channel	Address	Туре	Default Value	Unit	Description
ancentar yo Happing			PwmOutputCurFb	%IW65				Feedback current of Ch0-9.
Status	* >		Ch0	%IW65	UINT		mA	
	*		Ch1	%IW66	UINT		mA	
nformation	* >		Ch2	%IW67	UINT		mA	
	*		Ch3	%IW68	UINT		mA	
	* >		Ch4	%IW69	UINT		mA	
			Ch5	%IW70	UINT		mA	
	🍫		Ch6	%IW71	UINT		mA	
	* >		Ch7	%IW72	UINT		mA	
	🍫		Ch8	%IW73	UINT		mA	
			Ch9	%IW74	UINT		mA	
	۰. 🌪		ErrorState	%IB150				Error state of Ch0-9.
	1		PwmOutputCurCommand	%QW11				PWM output command of C
	🕀 🍢		DoOutputCommand	%QB42				DO output command of Ch0
	1 🛱 🍢		SetEmgergencyStop	%QB44				Set emergency stop of Ch0
	😟 🍢		ClearEmgergencyStop	%QB46				Clear emergency stop of Ch
	<							
	Reset Mapping Always update variables Use parent de						ce settin	g
	* = Create new variable	~ i ∳ = Ma	p to existing variable					
	Bus Cycle Options Bus cycle task Lies assess have such assisted as the cycle tasks							

Figure 84 機能ドライバ Internal I/O Mapping タブ選択時の画面

項目	ユーザー編集可否	摘要
Variable	0	ユーザー設定値。ユーザーアプリケーションで使用する変数を設定する。設定
		方法は 7.2.3.1 項参照。
Mapping	0	Variable に設定した変数がどこで宣言されて割り当てられたものか確認でき
		る。
		🌾 :本タブで入力し宣言された変数
		🎲 :POU 側で宣言された変数
Channel	×	各要素の名称が表示されている。機能にチャネル毎に設定する要素がある場
		合はその枝にチャネル番号が表示されている。
Address	×	要素に割り当てた変数のアドレスが表示されている。
Туре	×	各要素のデータ型が表示されている。
Default Value	0	ユーザー設定値。値を設定した時、アプリケーション起動時の初期値となる。
Unit	×	各要素の単位が表示されている。単位が無い場合は空欄。
Description	×	各要素の説明が表示されている。

Table 28 Internal I/O Mapping タブ 各列の概要


7.2.3.1. 変数の割り当て

各機能を制御するには、このタブで各要素に変数を割り当てる必要があります。割り当て方は2通りあります。

【変数割り当て方法】

- Internal I/O Mapping 上で変数を入力する。
 - ▶ タブ上の Variable 列に直接入力すると自動的に宣言され要素に割り当てられる。
- POU上で宣言している変数を入力する。
 - ▶ タブ上の Variable 列でクリックすると「…」ボタンが表示されるので、ボタンを押して POU 上で宣言された変数を選択する。その後下図のようになり要素にその変数が割り当てられる。

iternal Parameters	Find	ter Show all			Add FB for IO Ch	annel 🗂 Go to I	Instance	
nternal I/O Manning	Variable	Mapping	Channel	Address	Туре	Default Value	Unit	Descript '
itema yo napping			PwmOutputCurFb	%IW65				Feedbad
tatus			ErrorState	%IB150				Error sta
	÷		PwmOutputCurCommand	%QW11				PWM out
formation	🍫 PwmCom0	*	Ch0	%QW11	UINT(02000)	0	mA	
			Ch1	%QW12	UINT(02000)		mA	
			Ch2	%QW13	UINT(02000)		mA	
			Ch3	%QW14	UINT(02000)		mA	
			Ch4	%QW15	UINT(02000)		mA	
			Ch5	%QW16	UINT(02000)		mA	
			Ch6	%QW17	UINT(02000)		mA	
	· · · · · · · · · · · · · · · · · ·		Ch7	%QW18	UINT(02000)		mA	
			Ch8	%QW19	UINT(02000)		mA	
	↓ ↓ ↓ ~ ~ ~		Ch9	%QW20	UINT(02000)		mA	
	😟 * ø		DoOutputCommand	%QB42				DO outpu
	*		SetEmgergencyStop	%QB44				Set emer
		-	SetEmgStopCh0_7	%QB44	BYTE			
	Application.PLC_PRG.PwmStop0	~ >	Ch0	%QX11.0	BOOL			TRUE:Se
	ara 🍫		Ch1	%QX44.1	BOOL			TRUE:Se v
	<							>
	TPLIE:Set emergency stop. EALSE:Nothing in partic	ular	Reset Manning Alway	e undatevarial		vice eatting		
	indeliber einergeney stopy i Aebentou ing in parae		And y	o apoace variat	Use parent de	vice seturing		
	🍫 = Create new variable 🌍 = Map t	o existing vari	able					
	Rua Cuda Catiana							
	Bus Cycle Options							

Figure 85 変数割り当て例



7.2.3.2. デフォルト値の設定

変数を割り当てした後、アプリケーション起動時の初期値となるデフォルト値を必ず設定して下さい。設定し ない場合、アプリケーション起動後不定値となってしまい、想定しない制御・動作になる場合があります。

デフォルト値の設定は、下図のように変数を割り付けた要素の Default Value 列に値を入力することで設定で きます。設定値が BOOL 型(TRUE 若しくは FALSE)の場合は、要素の Default Value 列でクリックすることに よって値を設定できます。

<u>なお、変数が以下に該当する場合、この機能ドライバ画面上で Default Value を入力・設定できません。ユー</u> <u>ザは各プログラム(POU)上で変数を初期化する必要があります。</u>

【機能ドライバ画面上で Default Value が設定できない変数】

- POU で宣言した変数
- PWM 出力の電流出力指令 FB(PwmOutputCurFb)など、ユーザが CX2500 に設定をおこなう機能に該当しないもの(各機能ドライバの部分 7.3~7.10 節を参照)

ernal Parameters	Find		Filter Show all		- + A	dd FB for IO Channe	el → [(Go to Instance
ernal I/O Manning	Variable	Mapping	Channel	Address	Туре	Default Value U	Jnit [Description
cinar yo mapping	÷-**		PwmOutputCurCommand	%QW11			P	WM output command of
itus	- VwmCom0	***	Ch0	%QW11	UINT(02000)	0 m	ıΑ	
			Ch1	%QW12	UINT(02000)	m	hΑ	
ormation	**		Ch2	%QW13	UINT(02000)	m	hΑ	
			Ch3	%QW14	UINT(02000)	m	hA	
			Ch4	%QW15	UINT(02000)	m	hΑ	
			Ch5	%QW16	UINT(02000)	m	ıΑ	
	*		Ch6	%QW17	UINT(02000)	m	A	
	*		Ch7	%QW18	UINT(02000)	m	hΑ	
	- **		Ch8	%QW19	UINT(02000)	m	hA	
	· · · · · · · · · · · · · · · · · ·		Ch9	%QW20	UINT(02000)	m	hA	
	🖷 - * @		DoOutputCommand	%QB42			D	O output command of C
	÷**		SetEmgergencyStop	%QB44			S	et emergency stop of Cl
			SetEmgStopCh0_7	%QB44	BYTE	0 D	de este a	(DO (D))//A)
	PwmDo0	***	Ch0	%	ergency stop or cho-	9. Regardless of mod	de select	ion(DO/PWW), stop o
			Ch1	%QX44.1	BOOL		Т	RUE:Set emergency sto
			Ch2	%QX44.2	BOOL		Ţ	RUE:Set emergency stop
			Ch3	%QX44.3	BOOL		Т	RUE:Set emergency stop
	<							>
			Reset Map	oping Alw	vays update variables	Use parent device s	setting	
	 Bus Cycle Options Bus cycle task Use particular 	→ = Ma rent bus cycle se	ap to existing variable tting V Recreate	equired tasks				

Figure 86 デフォルト値設定例



7.2.3.3. 機能ドライバ制御可否・サイクルの設定

機能ドライバを機能させるには、「変数を常に更新(Always update variables)」と「バスサイクルタスク(Bus cycle task)」の設定が必要になります。それぞれ選択タブから設定を選択して下さい。デフォルトはどちらも親 デバイス(Device(CX2500))の設定に従うことになっています。

下図の例では、それぞれの設定を「親デバイスに従う」(ユーザーアプリケーションで処理に使用している変数のみ更新)にしています。Di2 はユーザーアプリケーションの POU で宣言していますが、処理に使用していません。その為制御値(ここではデジタル入力 Ch.2 の入力状態)が取得されません。

nal I/O Mapping	Variab	le	Mapping	Channel	Address	Туре	Default Value	Current Value	Prepared Va
				DinInputState	%IB89			Only subelements upd	
s	□	- 🍫		Ch0_7	%IB89	BYTE		Only subelements upd	
		🍫 Di0	***	Ch0	%IX89.0	BOOL		TRUE	
nation		🗝 🍫 Application.PLC_PRG.Di1	~⊘	Ch1	%IX89.1	BOOL		FALSE	
		Application.PLC_PRG.Di2	~	Ch2	%IX89.2	BOOL			
		*		Ch3	%IX89.3	BOOL			
		···· 🍫		Ch4	%IX89.4	BOOL			
		🍫		Ch5	%IX89.5	BOOL			
		···· 🍫		Ch6	%IX89.6	BOOL			
		I 🍫		Ch7	%IX89.7	BOOL		Not updated	
		🍫		Ch8_15	%IB90	BYTE			
		¥ø		Ch16_22	%IB91	BYTE			
				IgnInputState	%IX92.0	BOOL		Not updated	
	**************************************	*9 *9		Ch8_15 Ch16_22 IgnInputState	%IB90 %IB91 %IX92.0	BYTE BYTE BOOL		Not updated Not updated Not updated	
	<	N, FALSE:OFF		Reset Mappi	ing Alway	/s updatevar	iables Use paren	t device setting	
	* = 0	Create new variable 🏻 🍫 =	Map to existi	ng variable					

Figure 87 機能ドライバ「変数を常に更新」と「バスサイクルタスク」の設定箇所

Table 29	変数を常に更新(Always update variables)	選択肢
----------	----------------------------------	-----

選択肢	摘要
Use parent device setting	選択すると親デバイス設定(6.2 節の Always update variables で設定
	したもの)に従い、機能の各要素の制御値設定・取得する範囲を設定
	する。
Enabled 1	選択すると、ユーザーアプリケーションのタスクで使用していない変
(use bus cycle task if not used in any task) $^{\!$	数、変数を割り当てていない要素も含め全ての要素の制御値設定・
	取得を毎サイクルおこなう。

		この設定にすると、ユーザーアプリケーションの処理に使っていない要素にも数値が設定されてしまう。変数を割
		り当ててないものは起動時不定値になる。そのため、例えば PWM 出力値の場合、処理に使用していないチャネ
	注意	ルの出力電流指令値に 2000mA 等の大きな値が設定されてしまう可能性があり、負荷が接続されている場合に
※ 11		ユーザーが想定していない動作となり極めて危険である。よって、特別な事情が無い限り親デバイスの設定を
		Disable にした上で「Use parent device setting」を選択すること。



Table 30 パスサイクルタスク(Bus cycle task)^{※12} 選択肢

選択肢	摘要
Use parent bus cycle setting	選択すると親デバイスのバスサイクル(6.2 節の Bus Cycle Options で
	設定したもの)と同じ周期で、機能の制御値設定・取得がおこなわれ
	る。
(ユーザーが定義したタスク名)	選択したタスクのバスサイクルで機能の制御値設定・取得がおこな
	われる。

※12 バスサイクルタスクを短い周期のタスク(例えば、数 ms 等)に極力設定しないでください。そうした場合、IO ドライバの 入出力値は CX2500 内部で高頻度に更新処理をおこなうため、他の処理に遅れが生じる可能性があります。これは、IO ド ライバに限らず通常のタスク時間設定の際も留意して下さい。

7.2.3.4. アプリケーションで使用しない機能ドライバの設定

機能ドライバの内、ユーザーで使用しない機能がある場合は機能ドライバの制御を無効化すると、アプリケー ション内部で処理する時間を短縮することが可能です。ユーザーは、アプリケーションで使用しない機能ドライ バについて、最低限下記設定をおこなう必要があります。

【設定値】

- Table 7 の「Always update variables」で「Disabled」を選択する。
- Table 29 で「Use parent device setting」を選択する。

CCOT-24-016 Rev.1 7.3. デジタル入力・イグニッション入力



デジタル入力とイグニッション入力は機能ドライバ Cx2500Din を用いることで使用することができます。

7.3.1. Internal Parameters タブ

デジタル入力で初期設定する要素は入力形式の選択(Input selection)のみです。下表の解説を参考に設定して下さい。なお、イグニッション入力に関する設定要素はありません。

nternal I/O Mapping		BYTE	253		input selection for each channel.	
Status	Ch0	BOOL	200			
information			TDUE	TDUE	TRUE-BAT INDUIT EAUSE-CND INDUIT	
iformation		BOOL	EALCE	TRUE	TRUE BAT INDUC EALSE CND INDUC	
formation	4 Ch2	BOOL	TDUE	TRUE	TRUE-BAT INDUCT FALSE-GND INDUCT	
		BOOL	TRUE	TRUE	TRUE-BAT INPUT, FALSE-GND INPUT	
	¢ ch3	BOOL	TRUE	TRUE	TRUE-BAT IND IT FAI SE-GND IND IT	
		BOOL	TRUE	TRUE	TRUE BAT INPUT, FALSE GND INPUT	
		BOOL	TRUE	TRUE	TRUE:BAT INPUT, FAI SE:GND INPUT	
	¢ ch7	BOOL	TRUE	TRUE	TRUE:BAT INPLIT. FAI SE:GND INPLIT	
	□ @ Ch8 15	BYTE	253	1102		
	Ch8	BOOL	TRUE	TRUE	TRUE:BAT INPLIT, FALSE:GND INPLIT	
	- Ø Ch9	BOOL	FALSE	TRUE	TRUE:BAT INPUT, FALSE:GND INPUT	
	Ch10	BOOL	TRUE	TRUE	TRUE:BAT INPUT, FALSE:GND INPUT	
	- 🖗 Ch11	BOOL	TRUE	TRUE	TRUE:BAT INPUT, FALSE:GND INPUT	
	• Ch12	BOOL	TRUE	TRUE	TRUE:BAT INPUT, FALSE:GND INPUT	
	Ch13	BOOL	TRUE	TRUE	TRUE:BAT INPUT, FALSE:GND INPUT	
	Ch14	BOOL	TRUE	TRUE	TRUE:BAT INPUT, FALSE:GND INPUT	
	Ch15	BOOL	TRUE	TRUE	TRUE:BAT INPUT, FALSE:GND INPUT	
	⊕ Ch16 22	BYTE	127			

Figure 88 Cx2500Din Internal Parameters タブ画面

Ch.1 と Ch.9 をローサイド入力で設定した時の例(それ以外はハイサイド入力)

要素名(項目)	データ型	設定範囲	摘要	
Input selection	BOOL	TRUE/FALSE	・デジタル入力の)入力形式(ハイサイド/ローサイド)の選択。
			・チャネル毎に認	定できる。
			設定値	チャネル設定
			TRUE	ハイサイド入力
			FALSE	ローサイド入力

Table 31 デジタル入力 Internal Parameter タブ 要素一覧



CCOT-24-016 Rev.1 7.3.2. Internal I/O Mapping タブ

Internal I/O Mapping タブではデジタル入力とイグニッション入力の入力状態を取得できます。なお、本機 能ドライバでは変数の Default Value をドライバ画面上で直接設定することはできません。各 POII 上で初期値

能ドライバでは変数の Default Value をドライバ画面上で直接設定することはできません。各 POU 上で初期値 を設定して下さい。

iable * * - * - * - * Di0 - * Application.PLC_PRG.Di1 - * - * Application.PLC_PRG.Di2 - * - * -	Mapping	Channel DinInputState Ch0_7 Ch0 Ch1 Ch2 Ch3 Ch3 Ch4 Ch5	Address %IB89 %IB89 %IX89.0 %IX89.1 %IX89.2 %IX89.3 %IX89.4	Type BYTE BOOL BOOL BOOL BOOL BOOL	Default Value	Unit	Description Input status of Ch.0-22 state of it d TRUE:ON, FALSE:OFF TRUE:ON, FALSE:OFF TRUE:ON, FALSE:OFF
Yp Image: Non-State	*** *** ***	DinInputState Ch0_7 Ch0 Ch1 Ch2 Ch3 Ch4 Ch5	%IB89 %IB89 %IX89.0 %IX89.1 %IX89.2 %IX89.3 %IX89.4	BYTE BOOL BOOL BOOL BOOL BOOL			Input status of Ch.0-22 state of it d TRUE:ON, FALSE:OFF TRUE:ON, FALSE:OFF TRUE:ON, FALSE:OFF TRUE:ON, FALSE:OFF
□ 10 □ 10 □ 10 □ 10 □ 10 □ 10 □ 10 □ 10 □ 10 □ 10 □ 10 □ 10 □ 10 □ 10 □ 10 □ 10 □ 10	*** ** **	Ch0_7 Ch0 Ch1 Ch2 Ch3 Ch3 Ch4 Ch5	%IB89 %IX89.0 %IX89.1 %IX89.2 %IX89.3 %IX89.4	BYTE BOOL BOOL BOOL BOOL BOOL			TRUE:ON, FALSE:OFF TRUE:ON, FALSE:OFF TRUE:ON, FALSE:OFF TRUE:ON, FALSE:OFF
- 10 Di0 - 10 Application.PLC_PRG.Di1 - 10 Application.PLC_PRG.Di2 - 10 - 10	*** *** **	Ch0 Ch1 Ch2 Ch3 Ch4 Ch5	%IX89.0 %IX89.1 %IX89.2 %IX89.3 %IX89.4	BOOL BOOL BOOL BOOL BOOL			TRUE:ON, FALSE:OFF TRUE:ON, FALSE:OFF TRUE:ON, FALSE:OFF TRUE:ON, FALSE:OFF
Application.PLC_PRG.Dl1 Application.PLC_PRG.Dl2	70	Ch1 Ch2 Ch3 Ch4 Ch5	%IX89.1 %IX89.2 %IX89.3 %IX89.4	BOOL BOOL BOOL BOOL			TRUE:ON, FALSE:OFF TRUE:ON, FALSE:OFF
Application.PLC_PRG.Dl2	*	Ch2 Ch3 Ch4 Ch5	%IX89.2 %IX89.3 %IX89.4	BOOL BOOL BOOL			TRUE:ON, FALSE:OFF
- 19 - 19 - 19 - 19 - 19 - 19		Ch3 Ch4 Ch5	%IX89.3 %IX89.4	BOOL			TRUE:ON, EALSE:OFF
		Ch4 Ch5	%IX89.4	BOOL			inversity i nearly i
Image: Second		Ch5					TRUE:ON, FALSE:OFF
¥ø ¥ø			%IX89.5	BOOL			TRUE:ON, FALSE:OFF
		Ch6	%IX89.6	BOOL			TRUE:ON, FALSE:OFF
		Ch7	%IX89.7	BOOL			TRUE:ON, FALSE:OFF
🚊 🍫		Ch8_15	%IB90	BYTE			
🚊 - 🦄		Ch16_22	%IB91	BYTE			
**		IgnInputState	%IX92.0	BOOL			Input status of ignition. TRUE:ON,
		7					
		Reset Map	ping Alwa	ays update v	variables Use pare	ent devid	ce setting
= Create new variable 🏾 🍎 =	Map to existi	ng variable					
	 Create new variable Cycle Options Cycle task Use parent bus cycle 		Ch16_22 IgnInputState Greate new variable Cycle Options cycle task Use parent hus cycle setting Recreate r		** ** Ch16_22 %LB91 BYTE ** IgnInputState %LX92.0 BOOL ** IgnInputState %LX92.0 BOOL ** Reset Mapping Always updates ** Create new variable ** ** Cycle Options ** Recreate required tasks		Ch16_22 %IB91 BYTE IgnInputState %IX92.0 BOOL Reset Mapping Always update variables Use parent devi Create new variable

Figure 89 Cx2500Din Internal I/O Mapping タブ画面

Table 32 デジタル入力・イグニッション入力 Internal I/O Mapping タブ 要素一覧

要素名(Channel 列)	データ型	取得/設定値 範囲	摘要	
DinInputState	BOOL	TRUE/FALSE	・デジタル入力の ・チャネル毎に取)入力状態(ON/OFF)。 2得できる。
			取得值	チャネル状態
			TRUE	ON
			FALSE	OFF
IgnInputState	BOOL	TRUE/FALSE	・イグニッションフ	しカの入力状態(ON/OFF)。
			取得值	チャネル状態
			TRUE	ON
			FALSE	OFF



7.4. <u>周波数入力</u>

周波数入力は機能ドライバ Cx2500Frqin を用いることによって使用することができます。なお、周波数入力 は初期設定する要素はありません。そのため、Internal Parameter タブもありません。

7.4.1. Internal I/O Mapping タブ

Internal I/O Mapping タブでは周波数入力各チャネルの入力周波数と ON/OFF 入力状態を取得できます。なお、本機能ドライバでは、変数の Default Value をドライバ画面上で直接設定することはできません。各 POU 上で初期値を設定して下さい。

us	Variable	Mapping	Channel FrqinFrq	Address %ID24	Туре	Default Value	Unit	Description
Irmation	₽ ¥ø		FrqinFrq	%ID24				
rmation	¥ø		a 1 a					Input frequency value of Ch.0-7. Unit :
	×		Chu	%ID24	UDINT		0.01Hz	
	· · · · · · · · · · · · · · · · · · ·		Ch1	%ID25	UDINT		0.01Hz	
	🍫		Ch2	%ID26	UDINT		0.01Hz	
	🍫		Ch3	%ID27	UDINT		0.01Hz	
	🍫		Ch4	%ID28	UDINT		0.01Hz	
	🍫		Ch5	%ID29	UDINT		0.01Hz	
	🍫		Ch6	%ID30	UDINT		0.01Hz	
	L 🍫		Ch7	%ID31	UDINT		0.01Hz	
	🖮 ᡟ		FrqinDinState	%IB128				Din state of Frgin Ch.0-7.
	🖮 🍫		Ch0_7	%IB128	BYTE			
	🍫		Ch0	%IX128.0	BOOL			TRUE:ON, FALSE:OFF
	🍫		Ch1	%IX128.1	BOOL			TRUE:ON, FALSE:OFF
	🍫		Ch2	%IX128.2	BOOL			TRUE:ON, FALSE:OFF
	🍫		Ch3	%IX128.3	BOOL			TRUE:ON, FALSE:OFF
	妆		Ch4	%IX128.4	BOOL			TRUE:ON, FALSE:OFF
	🍫		Ch5	%IX128.5	BOOL			TRUE:ON, FALSE:OFF
	妆		Ch6	%IX128.6	BOOL			TRUE:ON, FALSE:OFF
	L ᡟ		Ch7	%IX128.7	BOOL			TRUE:ON, FALSE:OFF

Figure 90 Cx2500Frqin Internal I/O Mapping タブ画面

要素名(Channel 列)	データ型	取得/設定値	摘要				
		範囲					
FrqinFrq	UDINT	0.00~4000.00	・周波数入力の入力された周波数[0.01Hz]。				
			・チャネル毎に取	マ得できる。			
			・ハードウェア仕	様外の周波数が入力された場合、取得値は			
			不定になる。				
FrqinDinState	BOOL	TRUE/FALSE	・周波数入力の	入力状態(ON/OFF)。			
			・チャネル毎に取	マ得できる。			
			取得値	チャネル状態			
			TRUE	ON			
			FALSE	OFF			

Table 33 周波数入力 Internal I/O Mapping タブ 要素一覧



7.5. 2相カウンタ入力

2 相カウンタ入力は機能ドライバ Cx2500Cntin を用いることによって使用することができます。なお、2 相 カウンタ入力は起動時にユーザーが設定する要素はありません。そのため、Internal Parameter タブもありま せん。

7.5.1. Internal I/O Mapping タブ

Internal I/O Mapping タブでは、2 相カウンタ入力のカウント値設定/取得や ON/OFF 入力状態を取得できま す。なお、下記以外の機能については、変数の Default Value をドライバ画面上で直接設定することはできま せん。各 POU 上で初期値を設定して下さい。

【ドライバ画面上で Default Value を設定できる機能】

- EnableSetCount(2相カウンタへのカウント値設定許可)
- CntinCnt_Set(2相カウンタ入力の各チャネルへ設定するカウンタ値)

ernal I/O Mapping	FING		Filter Show all			• T AC		do to instance
tus	Variable	Mapping	Channel	Address	Туре	Default Value	Unit	Description
	📮 🎽		CntinCnt_Get	%ID18				Counter input value of each channel. (For "ge
ormation	🍫		Ch0_1	%ID18	DINT			
			Ch2_3	%ID19	DINT			
	🍫		Ch4_5	%ID20	DINT			
	🍆		Ch6_7	%ID21	DINT			
	🛱 🏘		DinState	%IB88				Din state of Ch.0-7(feature:GND INPUT)
	😟 🍫		Ch0_7	%IB88	BYTE			
	🖨 ^K ø		EnableSetCount	%QB0				Enable feature to set counter value.
	🖹 🍢		Ch0_7	%QB0	BYTE			
	···· *ø		EnableSetCh0_1	%QX0.0	BOOL			TRUE:Set count to CNTIN Ch0_1, FALSE:Do
	····· ***		EnableSetCh2_3	%QX0.1	BOOL			TRUE:Set count to CNTIN Ch2_3, FALSE:Do
	···· *ø		EnableSetCh4_5	%QX0.2	BOOL			TRUE:Set count to CNTIN Ch4_5, FALSE:Do
			EnableSetCh6_7	%QX0.3	BOOL			TRUE:Set count to CNTIN Ch6_7, FALSE:Do
	🖮 * ø		CntinCnt_Set	%QD1				Counter input value of each channel. (For "s
	***		Ch0_1	%QD1	DINT			
	···· **		Ch2_3	%QD2	DINT			
	* ø		Ch4_5	%QD3	DINT			
	*		Ch6_7	%QD4	DINT			
	1							
					_			
			Re	eset Mapping	Always	updatevariables	Use pa	rent device setting
	Mr. Craste and in	hin 2 M		-				
	= Create new varia	Die 🖗 = M	ap to existing variabl	e				

Figure 91 Cx2500Cntin Internal I/O Mapping タブ画面



Table 34	2相カウンタ入力	Internal I/O	Mapping タブ	要素一覧
----------	----------	--------------	------------	------

要素名(Channel 列)	データ型	取得/設定値 範囲	摘要	
CntinCnt_Get	DINT	$-2^{31} \sim 2^{31} - 1$	・2 相カウンタ入	カのカウント値(取得値)。
			・チャネル毎に取	2得できる。
			・カウント値(取得	骨値)は左記の上/下限値に達した場合、それ
			以上の値へカウ	ントアップ/ダウンしない。
DinState	BOOL	TRUE/FALSE	・2 相カウンタ入	カの入力状態(ON/OFF)。
			・チャネル毎に取	2得できる。
			取得值	チャネル状態
			TRUE	ON
			FALSE	OFF
EnableSetCount	BOOL	TRUE/FALSE	・2 相カウンタ入	カへのカウント値設定許可。
			・値が TRUE でお	ある間、CntinCnt_Set に設定したカウント値を
			CX2500 に設定す	する。
			設定値	処理内容
			TRUE	対応するチャネルへ CntinCnt_Set のカウ
				ンタ値を CX2500 に設定する。
			FALSE	対応するチャネルへのカウンタ値設定はお
				こなわない。
CntinCnt_Set	DINT	$-2^{31} \sim 2^{31} - 1$	・2 相カウンタ入	カの各チャネルへ設定するカウント値。
			・チャネル毎に認	と定できる。
			▪要素 EnableSet	tCount の対応するチャネルが TRUE であると
			きにカウンタ値か	「設定される。
			・左記設定値範疇	囲外の値は設定できない。



CCOT-24-016 Rev.1 7.5.2. カウンタ値セットの流れ

2相カウンタ入力の各チャネルヘカウント値をセットするには、プログラム上で下記のような流れでおこなう 必要が有ります。ただし、EnableSetCount が TRUE の間毎サイクルカウンタ値が設定されてしまうので、毎サ イクル設定する必要が無い場合は設定後にプログラム上で EnableSetCount を必ず FALSE にセットして下さ い。

(1)要素 CntinCnt_Set へ所望のカウント値をセットする。

カウント値を設定したいチャネルの CntinCnt_Set にカウント値をセットする。



(2) EnableSetCount を TRUE にセットする。

所望のチャネルの EnableSetCount を TRUE にセットする。



(3)CX2500 にカウント値がセットされる。

カウント値が実際に設定されたかは要素 CntinCnt_Get で確認する。



(4)EnableSetCountをFALSEにセットする。

Figure 92 2相カウンタ入力 カウント値の設定フロー



アナログ入力は機能ドライバ Cx2500Ain を用いることによって使用することができます。

7.6.1. Internal Parameters タブ

アナログ入力で初期設定する要素は入力形式の選択(Input selection)のみです。下表の解説を参考に設定して下さい。

nternal Parameters	Parameter	Туре	Value	Default Value	Unit	Description	
nternal I/O Mapping	E @ Ch0 7	BYTE	0			input selection for each channel.	
	€ @ Ch8_15	BYTE	0				
tatus	Ch16 23	BYTE	2				
formation	Ø Ch16	BOOL	FALSE	FALSE		TRUE:RESISTANCE INPUT(1K-OHM), FALSE:VOLTAGE INPUT	
Ironnacion	- 🔷 Ch17	BOOL	TRUE	FALSE		TRUE:RESISTANCE INPUT (1K-OHM), FALSE:VOLTAGE INPUT	
	- 🖗 Ch18	BOOL	FALSE	FALSE		TRUE:RESISTANCE INPUT (1K-OHM), FALSE:VOLTAGE INPUT	
	- 🔷 Ch19	BOOL	FALSE	FALSE		TRUE:RESISTANCE INPUT(1K-OHM), FALSE:VOLTAGE INPUT	
	- 🖗 Ch20	BOOL	FALSE	FALSE		TRUE:RESISTANCE INPUT(1K-OHM), FALSE:VOLTAGE INPUT	
	- 🔷 Ch21	BOOL	FALSE	FALSE		TRUE:RESISTANCE INPUT(1K-OHM), FALSE:VOLTAGE INPUT	
	🖗 Ch22	BOOL	FALSE	FALSE		TRUE:RESISTANCE INPUT(1K-OHM), FALSE:VOLTAGE INPUT	
	Ch23	BOOL	FALSE	FALSE		TRUE:RESISTANCE INPUT(1K-OHM), FALSE:VOLTAGE INPUT	
	🖻 - 🖗 Ch24_29	BYTE	16				
	Ch24	BOOL	FALSE	FALSE		TRUE:RESISTANCE INPUT(100-OHM), FALSE:VOLTAGE INPUT	
	🔷 🖗 Ch25	BOOL	FALSE	FALSE		TRUE:RESISTANCE INPUT(100-OHM), FALSE:VOLTAGE INPUT	
	🔷 🖗 Ch26	BOOL	FALSE	FALSE		TRUE:RESISTANCE INPUT(100-OHM), FALSE:VOLTAGE INPUT	
	🔷 🖗 Ch27	BOOL	FALSE	FALSE		TRUE:RESISTANCE INPUT(100-OHM), FALSE:VOLTAGE INPUT	
	- 🖗 Ch28	BOOL	TRUE	FALSE		TRUE:4-20mA INPUT, FALSE:VOLTAGE INPUT	
	🔶 🖗 Ch29	BOOL	FALSE	FALSE		TRUE:4-20mA INPUT, FALSE:VOLTAGE INPUT	

Figure 93 Cx2500Ain Internal Parameters タブ画面

Ch.17 を抵抗(1kΩ)入力、Ch.28 を 4-20mA 入力で設定した時の例(それ以外は 0-5V 電圧入力)

要素名(項目)	データ型	設定範囲	摘要	
Input selection	BOOL	TRUE/FALSE	・アナログ入	、カの入力形式の選択。
			・チャネル毎	に設定できる。
			設定値	チャネル設定
			TRUE	Ch.0~15 :ON/OFF 入力(電圧入力 0~32V)
				Ch.16~23 :抵抗入力(内部 1kΩプルアップ)
				Ch.24~27 :抵抗入力(内部 100 Ωプルアップ)
				Ch.28~29 :4-20mA 入力
			FALSE	電圧入力(0~5V)

Table 35 アナログ入力 Internal Parameter タブ 要素一覧



7.6.2. Internal I/O Mapping タブ

Internal I/O Mapping タブでは、アナログ入力の AD 値や ON/OFF 入力状態(Ch.0~15)の取得が可能です。 なお、本機能ドライバでは、変数の Default Value をドライバ画面上で直接設定することはできません。各 POU 上で初期値を設定して下さい。

ternal I/O Mapping		Filter Show all			Add FB for IO Cha	annel 👘 Go to Instance
centar yo happing	Variable	Mapping Channel	Address	Type	Default Value U	Init Description
		AdValue	%IW0			Ad value of Ch.0-29
atus	🖶 ᡟ	DinState	%IB60			Din state of Ch.0-15, Only w
	🗎 🛱 🏘	Ch0_7	%IB60	BYTE		
ormation	🍫	Ch0	%IX60.0	BOOL		TRUE:ON, FALSE:OFF
	*	Ch1	%IX60.1	BOOL		TRUE:ON, FALSE:OFF
	🦘	Ch2	%IX60.2	BOOL		TRUE:ON, FALSE:OFF
	. 👘 🦄	Ch3	%IX60.3	BOOL		TRUE:ON, FALSE:OFF
	🐐	Ch4	%IX60.4	BOOL		TRUE:ON, FALSE:OFF
	* >	Ch5	%IX60.5	BOOL		TRUE:ON, FALSE:OFF
	🍫	Ch6	%IX60.6	BOOL		TRUE:ON, FALSE:OFF
	L 🦘	Ch7	%IX60.7	BOOL		TRUE:ON, FALSE:OFF
	😟 ᡟ	Ch8_15	%IB61	BYTE		
	±	InternalVoltageWarning	%IB62			Bitfield mean Warning of inte
	*	AdValue_Battery	%IW32	UINT	di	git Ad value of battery voltage
	😟 🏘	BoardTemp	%IW33			Ad value that mean board te
	±-*9 ★-*9 *9 ₩-*9	Ch8_15 InternalVoltageWarning AdValue_Battery BoardTemp	%IB61 %IB62 %IW32 %IW33	UINT	di	Bitfield mean Warning git Ad value of battery v Ad value that mean b

Figure 94 Cx2500Ain Internal I/O Mapping タブ画面(赤枠:アナログ入力部)

Table 36 アナログ入力 Internal I/O Mapping タブ 要素一覧

要素名(Channel 列)	データ型	取得/設定値	摘要				
		範囲					
AdValue	UINT	0~4095	・アナログ入力の AD 値[digit]。				
			・チャネル毎に取	マ得できる。			
DinState	BOOL	TRUE/FALSE	・アナログ入力の) Ch.0~15 の入力状態(ON/OFF)。			
			・ただし、Input so	election でチャネルの設定を TRUE(ON/OFF			
			入力)にしていな	いと取得できないことに注意。			
			取得値	チャネル状態			
			TRUE	ON			
			FALSE	OFF			



7.7. 内部電源電圧監視入力

内部電源電圧監視入力は製品内部電源の異常監視するための機能です。機能ドライバ Cx2500Ain を用いることによって使用することができます。

7.7.1. Internal I/O Mapping タブ

機能ドライバ Cx2500Ain の Internal I/O Mapping タブでは、内部電源電圧監視の異常モニタ値やバッテリ 電圧 AD 値の取得が可能です。なお、本機能ドライバでは、変数の Default Value をドライバ画面上で直接設 定することはできません。各 POU 上で初期値を設定して下さい。

Internal Parameters	Find		Filter Show all		•	Add FB for IO	Channel	→ Go to Instance
Internal I/O Manning	Variable	Mapping	Channel	Address	Туре	Default Value	Unit	Description
Internal 1/0 Happing	😐 🍫		AdValue	%IW0				Ad value of Ch.0-29
Status	۰ 🐌 🖈		DinState	%IB60				Din state of Ch.0-15, Only wh
			InternalVoltageWarning	%IB62				Bitfield mean Warning of interr
nformation	🖻 鞭		WrnBit	%IB62	BYTE			TRUE:Internal voltage is abno
	¥ø		Rsv0	%IX62.0	BOOL			
	🐪		Wrn_Int7_5v	%IX62.1	BOOL			Internal 7.5V warning bit
	¥ø		Wrn_Int5v	%IX62.2	BOOL			Internal 5V warning bit
	* >		Wrn_IntA3_3v	%IX62.3	BOOL			Internal A3.3V warning bit
	*		Wrn_Int2_5vr	%IX62.4	BOOL			Internal 2.5VREF warning bit
	···· 🔖		Wrn_Int5vs0	%IX62.5	BOOL			Internal 5VS0 warning bit
	¥ø		Wrn_Int5vs1	%IX62.6	BOOL			Internal 5VS1 warning bit
	i 🍫		Wrn_Int5vs2	%IX62.7	BOOL			Internal 5VS2 warning bit
	*		AdValue_Battery	%IW32	UINT		digit	Ad value of battery voltage
	±*₽		BoardTemp	%IW33				Ad value that mean board te
	<					_		
			Reset Map	ping Alway	vs update vai	riables Enabled 1 (use hus cv	de task if not used in any task)
	🍫 = Create new variabl	e ~ 🎓 = Ma	ap to existing variable		,			
	-Bus Cycle Options							

Figure 95 Cx2500Ain Internal I/O Mapping タブ画面(赤枠:内部電源電圧監視入力部)



Table 37 内部電源電圧監視入力 Internal I/O Mapping タブ 要素一覧

要素名(Channel 列)	データ型	取得/設定値 範囲	摘要		
InternalVoltageWarning	BOOL	TRUE/FALSE	·各内部電源電	王の異常モニタ(TRUE	/FALSE)。
			・各要素の意味		
			要素名	摘要	
			Rsv0	予約ビット	
			Wrn_Int7_5v	内部電源 7.5V 異常-	モニタビット
			Wrn_Int5v	内部電源 5V 異常モ	ニタビット
			Wrn_IntA3_3V	内部電源 3.3V 異常-	モニタビット
			Wrn_Int2_5vr	内部電源 2.5V 異常-	モニタビット
			Wrn_Int5vs0	センサ用電源 Ch.0 昇	異常モニタビット
			Wrn_Int5vs1	センサ用電源 Ch.1 昇	異常モニタビット
			Wrn_Int5vs2	センサ用電源 Ch.2 昇	異常モニタビット
			・取得値の意味		
			取得值	チャネル状態	
			TRUE	内部電源電圧が異常	
			FALSE	正常	
AdValue_Battery	UINT	0~4095	・バッテリ電圧の	AD 值[digit]。	
			・AD 値から電圧	値への換算式の一例	は下記の通り。
			【AD 値•電圧値 <i>V</i>	換算式】	
			$AD = \frac{1}{V_{MCU}} \times AL$	$D_{MAX} \times R$	
			AD : AD	生値[digit]	
			AD _{MAX} :12b	itAD 最大值。4095[dig	;it]。
			V _{MCU} :MC	Uの電源電圧。3.3[V]。	5
			∨ :内部	鄂電源電圧[V]	
			R :分臣	王比(下表)	
			 ・監視対象ごとの)分圧比 R の値(詳細)	は機能仕様書を参
			照。)		
			項目		Rの値(TYP)
			BAT		0.048
			7.5V		0.248
			5V、3.3VA、2.5	VREF、5VS[0-2]	0.313



7.8. 基板温度監視入力

基板温度監視入力は製品内部の基板上に搭載したサーミスタ(RT)による温度モニタ機能です。機能ドライバ Cx2500Ain を用いることによって使用することができます。

7.8.1. Internal I/O Mapping タブ

機能ドライバ Cx2500Ain の Internal I/O Mapping タブでは、基板温度 AD 値の取得が可能です。なお、本 機能ドライバでは、変数の Default Value をドライバ画面上で直接設定することはできません。各 POU 上で初 期値を設定して下さい。

iternal Parameters	Find		Filter Show all			Add FB for IO	Channel	Go to Instance
nternal I/O Manning	Variable	Mapping	Channel	Address	Туре	Default Value	Unit	Description
ternar yo happing	💻 🍫		AdValue	%IW0				Ad value of Ch.0-29
tatus	😟 - 🍫		DinState	%IB60				Din state of Ch.0-15, Only
	😐 – 🍫		InternalVoltageWarning	%IB62				Bitfield mean Warning of int
ormation	*		AdValue_Battery	%IW32	UINT		digit	Ad value of battery voltage
	No		BoardTemp	%IW33				Ad value that mean board t
	¥ø		Ch0	%IW33	UINT		digit	
	¥ø		Ch1	%IW34	UINT		digit	
			Ch2	%IW35	UINT		digit	
	¢							
	<		Reset Map	ping Alwa	ys update va	riables Enabled 1 (use bus cy	de task if not used in any task
	<	le ~ = Mi	Reset Map	ping Alway	ys update va	riables Enabled 1 (use bus cy	de task if not used in any task)
	<	ie ~ = Mi	Reset Map ap to existing variable	ping Alwa	ys update va	riables Enabled 1 (use bus cy	de task if not used in any task

Figure 96 Cx2500Ain Internal I/O Mapping タブ画面(赤枠:基板温度監視入力)



Table 38 基	板温度監視入力	Internal I/O	Mapping タブ	要素一覺
------------	---------	--------------	------------	------

要素名(Channel 列)	データ型	取得/設定値	摘要			
		範囲				
BoardTemp	UINT	0~4095	・製品基板上に打	搭載したサーミスタから得られる周囲温度の AD		
			值[digit]。			
			・チャネル毎に取得できる。			
			・取得値は AD 値のため、温度については下式に従い換算す			
			チャネル番号	製品筐体内サーミスタ位置		
			Ch.0	MCU 周辺温度		
			Ch.1	パワー素子温度		
			Ch.2	コネクタ付近温度		

【AD 値→温度換算式】

(1)サーミスタの抵抗値:R[kΩ]

$$R = R_P \times (\frac{AD_{MAX} \times V_{MCU}}{V_{MCU} \times AD} - 1)^{-1}$$

(2)温度:T[℃]

$$T = (B^{-1} \times \ln\left(\frac{R}{R_0}\right) + T_0^{-1})^{-1} - 273$$

$\mathbf{R}_{\mathbf{P}}$: 入力プルアップ抵抗。4.7[kΩ]。
AD _{MAX}	: 12bitAD 最大值。4095[digit]。
AD	: AD 値[digit]。
V _{MCU}	: MCU の電源電圧。 3.3[V]。
В	: サーミスタ B 定数。3500[K]。
\mathbf{R}_0	: 常温 25[℃]時のサーミスタ抵抗値。10[kΩ]。
T_0	: 常温 25[℃]+273=298[K]。



7.9. <u>デジタル出力</u>

デジタル出力は機能ドライバ Cx2500Do を用いることで使用することができます。

7.9.1. Internal Parameters タブ

デジタル出力で初期設定する要素は出力形式の選択(Output selection)のみです。下表の解説を参考に設定して下さい。

ternal I/O Mapping	internal Parameters	Parameter	Туре	Value	Default Value	Unit	Description	
atus formation f	Internal I/O Mapping		BYTE	253			ouput selection for each diamine (excluding entor 15).	
atus - Image: Chi image: Chimage: Chi image: Chimage: Chi image: Chi image			BOOL	TRUE	TRUE		TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT	
formation - I Ch2 BOOL TRUE TRUE TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT - I Ch3 BOOL TRUE TRUE TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT - I Ch4 BOOL TRUE TRUE TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT - I Ch4 BOOL TRUE TRUE TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT - I Ch5 BOOL TRUE TRUE TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT - I Ch6 BOOL TRUE TRUE TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT - I Ch6 BOOL TRUE TRUE TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT - I Ch6 BOOL TRUE TRUE TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT - I Ch7 BOOL TRUE TRUE TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT - I Rsv BYTE 0xFF 0xFF Don't care(because Ch8_15 are fixed at HIGH SIDE INPUT.)	tatus	🔷 🖗 Ch1	BOOL	FALSE	TRUE		TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT	
Ch3 BOOL TRUE TRUE TRUE TRUEHIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Ch4 BOOL TRUE TRUE TRUEHIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Ch5 BOOL TRUE TRUE TRUEHIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Ch6 BOOL TRUE TRUE TRUEHIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Ch7 BOOL TRUE TRUE TRUEHIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Ch7 BOOL TRUE TRUE TRUE TRUEHIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Ch7 BOOL TRUE TRUE TRUE TRUEHIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Ch7 BOOL TRUE TRUE TRUE TRUEHIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Ch7 BOOL TRUE TRUE TRUE TRUEHIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Ch7 BOOL TRUE TRUE TRUE TRUEHIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Ch7 BOOL TRUE TRUE TRUE TRUEHIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Ch7 BOOL TRUE TRUE TRUE TRUEHIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Ch7 BOOL TRUE TRUE TRUE TRUEHIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Ch7 BOOL TRUE TRUE TRUE TRUEHIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Ch7 BOOL TRUE TRUE TRUE TRUEHIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Ch7 BOOL TRUE TRUE TRUE TRUEHIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Ch7 BOOL TRUE TRUE TRUE TRUEHIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Ch7 BOOL TRUE TRUE TRUE TRUEHIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Ch7 BOOL TRUE TRUE TRUE TRUEHIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT, FALSE:	formation	🖗 Ch2	BOOL	TRUE	TRUE		TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT	
 		🔷 🖗 Ch3	BOOL	TRUE	TRUE		TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT	
 		🖤 🖗 Ch4	BOOL	TRUE	TRUE		TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT	
Image: Ch6 BOOL TRUE TRUE TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Image: Ch7 BOOL TRUE TRUE TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Image: Ch7 BOOL TRUE TRUE TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Image: Ch7 BOOL TRUE TRUE TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Image: Ch7 BOOL TRUE TRUE TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT Image: Ch7 BOOL OxFF OxFF Don't care(because Ch8_15 are fixed at HIGH SIDE INPUT.)		- Ø Ch5	BOOL	TRUE	TRUE		TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT	
Ch7 BOOL TRUE TRUE TRUE TRUE: UTPUT, FALSE:LOW SIDE OUTPUT Rsv BYTE 0xFF 0xFF Don't care(because Ch8_15 are fixed at HIGH SIDE INPUT.)		🗇 Ch6	BOOL	TRUE	TRUE		TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT	
Rsv BYTE 0xFF 0xFF Don't care(because Ch8_15 are fixed at HIGH SIDE INPUT.)		🗇 Ch7	BOOL	TRUE	TRUE		TRUE:HIGH SIDE OUTPUT, FALSE:LOW SIDE OUTPUT	
		🖳 🛷 Rsv	BYTE	0xFF	0xFF		Don't care(because Ch8_15 are fixed at HIGH SIDE INPUT.)	

Figure 97 Cx2500Do Internal Parameters タブ画面

Ch.1をローサイド出力で設定した時の例(それ以外はハイサイド出力)

要素名(項目)	データ型	設定範囲	摘要			
Output selection	BOOL	TRUE/FALSE	・デジタル出力の)出カ形式(ハイサイド/ローサイド)の選択。		
			・チャネル毎に設定できる。(ただし、Ch.8~15 についてはパ			
			サイド出力固定の	のため設定項目無し)		
			設定値	チャネル設定		
			TRUE	ハイサイド出力		
			FALSE	ローサイド出力		

Table 39	デジタル出力	Internal Parameters タブ	要素一覧
----------	--------	------------------------	------



7.9.2. Internal I/O Mapping タブ

Internal I/O Mapping タブではデジタル出力の出力設定、及び出力状態の取得ができます。なお、下記以外の機能については、変数の Default Value をドライバ画面上で直接設定することはできません。各 POU 上で 初期値を設定して下さい。

【ドライバ画面上で Default Value を設定できる機能】

● DoOutputCommand(デジタル出力の出力指示設定)

	Madabla	Manalaa	Channel	Address	Turne	DefaultMalue	11-24	Description
al I/O Mapping	variable	Mapping	Channel	Address %OB20	iype	Default value	Unit	Description
	÷		Ch0_7	%QB20	BYTE			
	- ** Do0	**	Ch0	%QX20.0	BOOL	FALSE		TRUE:ON, FALSE:OFF
ation	- **		Ch1	%QX20.1	BOOL			TRUE:ON, FALSE:OFF
	- **		Ch2	%QX20.2	BOOL			TRUE:ON, FALSE:OFF
	- **		Ch3	%QX20.3	BOOL			TRUE:ON, FALSE:OFF
			Ch4	%QX20.4	BOOL			TRUE:ON, FALSE:OFF
	- **		Ch5	%QX20.5	BOOL			TRUE:ON, FALSE:OFF
	- **		Ch6	%QX20.6	BOOL			TRUE:ON, FALSE:OFF
	- **		Ch7	%QX20.7	BOOL			TRUE:ON, FALSE:OFF
			Ch8_15	%QB21	BYTE			
	😑 - 🍫		DoFbState	%IB93				Output Feedback of Ch0-15. TRUE:ON, F
	😑 🍫		Ch0_7	%IB93	BYTE			
	- *>		Ch0	%IX93.0	BOOL			TRUE:ON, FALSE:OFF
	- *>		Ch1	%IX93.1	BOOL			TRUE:ON, FALSE:OFF
	- *>		Ch2	%IX93.2	BOOL			TRUE:ON, FALSE:OFF
	- *>		Ch3	%IX93.3	BOOL			TRUE:ON, FALSE:OFF
	- *>		Ch4	%IX93.4	BOOL			TRUE:ON, FALSE:OFF
	<							>
			Res	et Mapping	Always upda	atevariables Er	nabled 1	(use bus cycle task if not used in any task)
	🍫 = Create new variable	🍫 = Ma	ap to existing variable					

Figure 98 Cx2500Do Internal I/O Mapping タブ画面

Table 40	デジタル出力	Internal I/O	Mapping タブ	要素一覧
----------	--------	--------------	------------	------

要素名(Channel 列)	データ型	取得/設定値 範囲	摘要			
DoOutputCommand	BOOL	TRUE/FALSE	・デジタル出力の出力指示設定(ON/OFF)。			
			・チャネル毎に設定できる。			
			設定値	チャネル設定		
			TRUE	ON		
			FALSE	OFF		
DoFbState ^{%13}	BOOL	TRUE/FALSE	・デジタル出力の出力状態(ON/OFF)。			
			・チャネル毎に取得できる。			
			取得值	チャネル状態		
			TRUE	ON		
			FALSE	OFF		

※13 DoFbState は各チャネルの出力端子電圧(HIGH/LOW)で出力状態(ON/OFF)を判定している。そのため、端子に負荷を接続していないと電圧を正確に読み取れず、正しい出力状態を判定できないことに注意。



7.10. <u>PWM 出力</u>

PWM 出力は機能ドライバ Cx2500Pwm を用いることで使用することができます。

7.10.1. Internal Parameters タブ

PWM 出力で初期設定する要素は下表の通りです。下表の解説を参考に設定して下さい。

ernal Parameters	Parameter	Туре	Value	Default Value	Unit	Description
	PWM frequency	UINT(501000)	250	200	Hz	Frequency for PWM control(= PWM dither). Settable range: 50-1000
rnal I/O Mapping	PID parameters					PID parameters for each channel.
	⊟-					
15	— 🌵 Кр	UINT	250	200		PID-P value
mation	🖗 Ki	UINT	55	50		PID-I value
	🧼 🖗 Kd	UINT	505	500		PID-D value
	⊞– ∲ Ch1					
	I⊞ I € Ch3					
	I≣- ∲ Ch4					
	III					
	III → IV Ch6					
	⊞ ∲ Ch7					
	I≣- ∲ Ch8					
	I ← Ø Ch9					
	😑 🔮 Enable DO mode					Bitfield mean DO mode enable/disable.
	EnableDoModeCh0_7	BYTE	0			
	EnableDoModeCh8_9	BYTE	1			
	🖗 Ch8	BOOL	TRUE	FALSE		TRUE:DO mode, FALSE:PWM mode
	d cho	BOOL	EALSE.	EALSE		TRUE DO mode, FAI SE PWM mode

Figure 99 Cx2500Pwm Internal Parameterss タブ画面

上図はディザ周波数と Ch.0 の PID 値、Ch.8 の出力モードをデフォルト値から変更した時の例

要素名(項目)	データ型	設定範囲	摘要		
PWM frequency	UINT(501000)	50~1000	・PWM 出力全チ	ャネルのディザ周波数(PWM 制御周波	
			数)[Hz]。		
			・全チャネル共通のため、チャネル毎に個別に設定するこ		
			はできない。		
PID parameters	各種 UINT	各種 0~65535	・出力チャネルの PID パラメータの設定。		
			・チャネル毎に設定できる。		
			要素名	摘要	
			Кр	PID 制御用の P(比例)値	
			Ki	PID 制御用の I(積分)値	
			Kd	PID 制御用の D(微分)値	
Enable_DO_mode	BOOL	TRUE/FALSE	・出力チャネルの出力制御モード(7.10.3 項参照)の設定		
			・チャネル毎に語	安定できる。	
			設定値	チャネル設定	
			TRUE	DO 出力モード	
			FALSE	PWM 出力モード	

Table 41 PWM 出力 Internal Parameters タブ 要素一覧



7.10.2. Internal I/O Mapping タブ

Internal I/O Mapping タブでは PWM 出力の出力設定・状態取得等が可能です。なお、下記以外の機能については、変数の Default Value をドライバ画面上で直接設定することはできません。各 POU 上で初期値を設定して下さい。

【ドライバ画面上で Default Value を設定できる機能】

- PwmOutputCurCommand(各チャネルの出力電流指示設定)
- DoOutputCommand(各チャネルの DO 出力指示設定)
- SetEmergencyStop(出力チャネルへの緊急停止時設定)
- ClearEmergencyStop(緊急停止状態の出力チャネルへの緊急停止解除設定)

ternal Parameters	Find	Filter Show all			Add FB for IO Cha	annel [→] Go to	Instance	2
ternal I/O Manning	Variable	Mapping	Channel	Address	Туре	Default Value	Unit	Descripti
			PwmOutputCurFb	%IW65				Feedback
atus	😟 - 🍫		ErrorState	%IB150				Error state
	🚊 🍫		PwmOutputCurCommand	%QW11				PWM outp
formation	🗇 PwmCom0	×.	Ch0	%QW11	UINT(02000)	0	mA	
			Ch1	%QW12	UINT(02000)		mA	
	* *		Ch2	%QW13	UINT(02000)		mA	
	*		Ch3	%QW14	UINT(02000)		mA	
	*		Ch4	%QW15	UINT(02000)		mA	
	*		Ch5	%QW16	UINT(02000)		mA	
	*		Ch6	%QW17	UINT(02000)		mA	
	*		Ch7	%QW18	UINT(02000)		mA	
			Ch8	%QW19	UINT(02000)		mA	
	· · · · · · · · · · · · · · · · ·		Ch9	%QW20	UINT(02000)		mA	
	🖬 ^r ø		DoOutputCommand	%QB42				DO outp
	🗐 *		SetEmgergencyStop	%QB44				Set emer
	😟 ^r .		ClearEmgergencyStop	%QB46				Clear em
	<							
			Reset Mapping Alway	s update varia	bles Use parent de	vice setting		
	🌾 = Create new variable	🍫 = Map to existing var	able					
	Bus Cycle Options							
	Due sude tests		Decreate required tasks					

Figure 100 Cx2500Pwm Internal I/O Mapping タブ画面



Table 42 PWM 出力 Internal I/O Mapping タブ 要素一覧(1/2)

要素名(Channel 列)	データ型	取得/設定値	摘要		
		範囲			
PwmOutputCurFb	UINT	0~65535	・PWM 出力の出	¦カ電流フィードバック[mA]。	
			・チャネル毎に取得できる。		
ErrState	BOOL	TRUE/FALSE	・PWM 出力のチ	ャネルステータス。	
			・チャネル毎に耳	双得できる。	
			・エラーについて	〔は 7.10.5 項を参照。	
			・各要素の意味		
			要素名 摘要		
			EmergencySto	p 緊急停止ステータス	
			ShortCircuit	短絡エラーステータス	
			WireBreak	断線エラーステータス	
			・エラー状態		
			取得值	ステータス	
			TRUE	エラー	
			FALSE 正常		
PwmOutputCurCommand	UINT(02000)	0~2000	・各チャネルの出力電流指示設定[mA]。		
			・チャネル毎に設定できる。		
			・DO モードのチャネルはこの要素をセットしても		
			CX2500には設定されない。		
			・下記組み合わせのチャネルは同時に出力できず、		
			れぞれ一方のみ出力できる。		
			・Ch.0とCh.1		
			・Ch.2 と Ch.3		
			・Ch.4 と Ch.5		
			・Ch.6 と Ch.7		
			・Ch8 と Ch.9		
DoOutputCommand	BOOL	TRUE/FALSE	・各チャネルの [DO 出力指示設定(TRUE/FALSE)。	
			・チャネル毎に語	と定できる。	
			・PWM モードの	チャネルはこの要素をセットしても	
			CX2500 には設	定されない。	
			 PwmOutputCu 	rCommandと同様に隣り合うチャネルは	
			同時出力できな	いことに注意。	
			設定値	チャネル設定	
			TRUE	DO モード: ON	
			FALSE	DO モード: OFF	



Table 43 PWM 出力	Internal I/O Mapping タブ	要素一覧(2/2)
-----------------	-------------------------	-----------

要素名(Channel 列)	データ型	取得/設定値 範囲	摘要	
SetEmergencyStop	BOOL	TRUE/FALSE	 ・出力チャネル・ 	~の緊急停止指示設定
			(TRUE/FALSE)	10
			•TRUE の間は.	、毎バスサイクル指定チャネルを緊急停
			止し続けること	に注意。
			設定値	チャネル設定
			TRUE	出力強制停止・緊急停止状態へ移行
			FALSE	処理無し
ClearEmergencyStop	BOOL	TRUE/FALSE	·緊急停止状態	の出カチャネルへの緊急停止解除指示
			設定(TRUE/FA	LSE)。
			・TRUE の間は.	、毎バスサイクル指定チャネルの緊急停
			止を解除し続け	ることに注意。
			設定値	チャネル設定
			TRUE	エラー解除・出力可能状態
			FALSE	処理無し



CCOT-24-016 Rev.1 7.10.3. DO モード

PWM 出力チャネルの出力制御モードは 2 つあります。これらは機能ドライバ Cx2500Pwm の Internal Parameter タブ・Enable_DO_mode 要素で設定します。各チャネルは、この Enable_DO_mode 要素で設定したモードでしか起動後出力制御できません。そのため、チャネルに接続する負荷の用途に合わせて出力モードを 選択し使用して下さい。

Table 44 PWM 出力 出力モード一覧

モード名	摘要		
PWM モード	・チャネルを PWM 出力として使用できる。		
	・PwmOutputCurCommand要素の設定が有効になる。		
	・DoOutputCommand 要素の設定が無効になる。		
DO モード	・チャネルをデジタル出力(ハイサイド出力)として使用できる。		
	・DoOutputCommand 要素の設定が有効になる。		
	・PwmOutputCurCommand要素の設定が無効になる。		

7.10.4. 初期状態

PWM 出力のチャネルは安全性を考慮し、起動時全て緊急停止状態にしています(ErrState 要素で確認可能)。 そのため、各チャネルは緊急停止状態を解除(ClearEmergencyStop 要素を TRUE 設定)しないと出力できないこ とに注意して下さい。

7.10.5. エラー時の処理と解除方法

PWM 出力には短絡エラーと断線エラーという2つのエラーがあります。それぞれエラーを検知した時、検知 したチャネルは出力を停止し、緊急停止状態(ErrState 要素の EmergencyStop が TRUE)に移行します。緊急停 止状態に移行した後に再度出力をおこないたい場合、まずは緊急停止状態の解除(ClearEmergencyStop 要素を TRUE にセット)をおこなう必要があります。緊急停止状態を解除すると ErrState 要素の EmergencyStop が FALSE になります。解除後、出力設定された値で出力を自動で再開されることに注意して下さい。



7.11. RS232C

ユーザーは RS232C(シリアル通信)を使用することが可能です。機能を使用するには、ライブラリ SysCom が 登録されている必要があります。使用できる関数は下記の通りです。

また、SysCom ライブラリについては CODESYS オンラインヘルプにも記載があります。そちらも合わせて参照して下さい。

Table 45	RS232C(S)	/sCom ラ	イブラリ)	閗数—暫
	1.02020(0)		1 2 2 11	内外 元

機能区分	関数名	摘要
ポート制御	SysComOpen2	シリアル通信ポートをオープンする。
	SysComClose	シリアル通信ポートをクローズする。
受信	SysComRead	内部バッファから受信したメッセージを取得する。
送信	SysComWrite	メッセージを送信する。
非対応	SysComGetSettings	非対応
	SysComGetSettings2	
	SysComOpen	
	SysComOpen3	
	SysComPurge	
	SysComSetSettings	
	SysComSetSettings2	
	SysComSetTimeout	

7.11.1. <u>列挙型</u>

RS232Cで使用する SysCom ライブラリで定義されている列挙型は下記の通りです。ただし、本製品非対応の ものは除きます。

型名	SYS_COM_BAUDRATE					
摘要	RS232C のボーレートを表す。					
列挙子	名前	値	説明			
	SYS_BR_4800	4800	4800bps			
	SYS_BR_9600	9600	9600bps			
	SYS_BR_19200	19200	19200bps			
	SYS_BR_38400	38400	38400bps			
	SYS_BR_57600	57600	57600bps			
	SYS_BR_115200	115200	115200bps			



型名	SYS_COM_PARITY				
摘要	RS232C のパリティチェックの種類を表す。				
列挙子	名前	値	説明		
	SYS_NOPATIRY	0	パリティチェック無効		
	SYS_ODDPARITY	1	奇数パリティチェック		
	SYS_EVENPARITY	2	偶数パリティチェック		

型名	SYS_COM_PORTS		
摘要	RS232C のポート番号を表す。		
列挙子	名前	値	説明
	SYS_COMPORT_NONE	0	ポート無し(<mark>非対応</mark>)
	SYS_COMPORT1	1	ポート 1(RS232C の Ch.0 にあたる)
	SYS_COMPORT2	2	ポート 2(RS232C の Ch.1 にあたる)
	SYS_COMPORT3	3	ポート 3(<mark>非対応</mark>)
	SYS_COMPORT4	4	ポート 4(<mark>非対応</mark>)

型名	SYS_COM_STOPBITS				
摘要	RS232C のストップビット長を表す。				
列挙子	名前	値	説明		
	SYS_ONESTOPBIT	1	1 ビット		
	SYS_ONE5STOPBITS	2	(非対応)		
	SYS_TWOSTOPBITS	3	2 ビット		

型名	SYS_COM_TIMEOUT		
摘要	RS232C の送受信タイムアウト待ち時間を表す。		
列挙子	名前	値	説明
	SYS_NOWAIT	0	タイムアウト待ち時間無し
	SYS_INFINITE	0xFFFFFFFF	タイムアウト待ち時間無限(処理待ちし続ける)
備考	CX2500内部処理の仕様上、 <u>非対応</u> 。この列挙型を使用した関数の呼び出しの際には任意値を設定する		
	こと。		



7.11.2. <u>構造体</u>

RS232C で使用する SysCom ライブラリで定義されている構造体は下記の通りです。ただし、本製品非対応の ものは除きます。

型名	SysComSettings				
摘要	RS232C ポートの通信パラメー	ータ設定用構造体			
メンバ	型				
	SYS_COM_PORTS	sPort	RS232C ポート番号		
	SYS_COM_STOPBITS	byStopBits	ストップビット		
	SYS_COM_PARITY	byParity	パリティ		
	SYS_COM_BAUDRATE	ulBaudrate	ボーレート		
	SYS_COM_TIMEOUT	ulTimeour	タイムアウト待ち時間(<mark>非対応)</mark>		
	UDINT	ulBufferSize	受信バッファサイズ <mark>(非対応</mark>)		

型名	SysComSettingsEx						
摘要	RS232C ポートの通信拡張機能パラメータ設定用構造体						
メンバ	型	名前	説明				
	BYTE	byByteSize	(非対応)				
	BOOL	bBinary					
	BOOL	bOuttxCtsFlow					
	BOOL	bOuttxDsrFlow					
	BOOL	bDtrControl					
	BOOL	bDsrSensitivity					
	BOOL	bRtsControl					
	BOOL	bTXContinueOnXoff					
	BOOL	bOutX					
	BOOL	bInX					
	ВҮТЕ	byXOnChar					
	ВҮТЕ	byXoffChar					
	WORD	wXonLim					
	WORD	wXoffLim					
備考	SysComOpen2 関数の引数で	はあるが、CX2500 内部処理の)仕様上、本構造体のメンバは使用されない。よ				
	って、関数の引数として使用する際は任意値を設定すること。						



7.11.3. <u>関数</u>

RS232Cで使用できる関数について、それぞれ下記に示します。ただし、本製品非対応のものは除きます。

関数名	SysComOpen2					
摘要	RS232C ポートをオ	ープンする。				
引数	型		名前	説明		
(INPUT)	POINTER_TO SysComSettings		pSettings	通信パラメータ		
	POINTER_TO		pSettingsEx	通信拡張機能用パラメータ(非対応)		
	SysComSettingsEx					
	POINTER_TO RTS_	IEC_RESULT	pResult	ERR_OK(0x0)	:オープン成功	
				ERR_PARAMETER(0x2)	:オープン失敗(引数異	
					常)	
				ERR_NOTINITIALIZED(0x3)	:オープン済み	
引数	無し					
(IN_OUT)						
返り値	型	名前		説明		
(OUTPUT)	RTS_IEC_HANDLE	SysComOpe	n2	オープンしたポートのポート	番号	
備考	・本関数呼び出し後受信が開始される。					
	・構造体 pSettings のメンバ sPort には必ず SYS_COMPORT1 又は SYS_COMPORT2 を入れること。					
	・構造体 pSettingsE	Ex は C2500 ゆ	可部処理に不使	「用の為、関数を呼び出す際は	は任意値をそれぞれ設定す	
	ること。					

関数名	SysComClose					
摘要	RS232C ポートをク	ローズする。				
引数	型		名前		説明	
(INPUT)	RTS_IEC_HANDLE		hCom		ポート番号 <mark>(</mark>	1 又は 2 を設定する
					こと)	
引数	無し					
(IN_OUT)						
返り値	型	名前		説明		
(OUTPUT)	RTS_IEC_RESULT	SysComClose)	ERR_OK(0x0)	: クロ	ーズ成功
				ERR_PARAMETER()x2) :クロ	ーズ失敗(引数異常)
備考	・送受信を中止、これまでに受信したバッファをクリアし、ポートをクローズする。					
	<u>・引数 hCom には必</u>	<u>がず1又は2を</u>	·設定すること。			



関数名	SysComRead						
摘要	内部バッファから受信したメッセージを取得する。						
引数	型		名前	説明	月		
(INPUT)	RTS_IEC_HANDLE		hCom	ポー	-ト番号(1 又は2を設)	定すること)	
	POINTER_TO_BYTE		pbyBuffer	受信	言データ取得用データノ	、 ッファのポインタ	
	UDINT		ulSize	取襘	取得したい受信データサイズ		
	SYS_COM_TYMEOU	JT	ulTimeout	受信タイムアウト待ち時間(非対応)			
	POINTER_TO RTS_IEC_RESULT		pResult	ER	R_OK(0x0)	:取得成功	
				ER	R_FAULED(0x1)	:取得失敗(受信バッファ	
						オーバーフロー、通信エラ	
						—)	
				ER	R_PARAMETER(0x2)	:取得失敗(引数異常)	
引数	無し						
(IN_OUT)							
返り値	型	名前			説明		
(OUTPUT)	UDINT	SysComRead			取得できたデータの+	ナイズ	
備考	<u>・引数 hCom には必</u>	シず1又は2を	設定すること。	_			
	・CX2500 内の受信	バッファサイズ	は 512。ユー+	ゲーカ	「本関数を呼び出す前」	こそのサイズより多いデータ	
	を受信していた場合、CX2500 は受信バッファオーバーフローエラーとなる。						
	・本関数呼び出し後	、引数 pResult	がERR_FAILE	D (27	なった場合はエラーが	発生しており、内部で送受信	
	ができない状態にな	ぶっている。 再居	度送受信をおこ	なう	場合はポートクローズ-	→オープンをおこなう必要が	
	ある。						



関数名	SysComWrite						
摘要	メッセージの送信部	定・送信をおこ	なう。				
引数	型		名前	説明	説明		
(INPUT)	RTS_IEC_HANDLE		hCom	ポート	ポート番号(1 又は 2 を設定すること)		
	POINTER_TO_BYTE		pbyBuffer	送信·			
	UDINT		ulSize	送信	送信したいデータサイズ		
	SYS_COM_TYMEOU	JT	ulTimeout	it 送信タイムアウト待ち時間(<mark>非対応)</mark>		(非対応)	
	POINTER_TO RTS_	IEC_RESULT	pResult	ERR_	OK(0x0)	:送信設定成功	
				ERR_	FAILED(0x1)	:送信設定失敗(受信バッフ	
						ァオーバーフロー、通信エラ	
						—)	
				ERR_	PARAMETER(0x2)	:送信設定失敗(引数異常)	
				ERR_	PENDING(0xA)	:送信設定成功(備考参照)	
引数	無し						
(IN_OUT)							
返り値	型	名前			説明		
(OUTPUT)	UDINT	SysComWrite			送信設定できたデー	ータのサイズ	
備考	<u>・引数 hCom には必</u>	ふず1又は2を	設定すること	<u>o</u>			
	・CX2500 内の送信	言バッファサイン	ズは 512。そ	のサイ	イズより多いデータを	送信設定しようとした場合、	
	pResult は ERR_P	ENDING となる	₀。 バッファサ・	イズを起	置えている分のデータ	は、先に送信設定されている	
	データが送信完了し	した後、CX2500)が自動で送	信する	0		
	・本関数で多量のラ	ータを速い周期	期で呼び出し	続ける	場合、CX2500 の送	信処理が完了していない場合	
	がある。その場合も ERR_PENDING となる。この場合、 <mark>送信周期を長くしたり、1 度に送るデータ量を少な</mark>						
	くする必要がある。						
	・本関数呼び出し後	、引数 pResult	: が ERR_FAIL	.ED (こフ	なった場合はエラーカ	「発生しており、内部で送受信	
	ができない状態にな	こっている。再度	度送受信をお	こない	場合はポートクロース	、→オープンをおこなう必要が	
	ある。						



CCOT-24-016 Rev.1 7.11.4. ポートオープン・クローズ

RS232C でメッセージの送受信を行うには、SysComOpen2 関数を呼び出し、ポートオープンする必要があり ます。ポートオープン後、メッセージ受信待ち状態になり送受信が可能になります。



Figure 101 RS232C ポートオープン・クローズ呼出例(ST 言語)

7.11.5. <u>ポートクローズ</u>

通信エラーなどで受信・送信ができなくなった場合、SysComClose 関数を呼び出しポートクローズして下さい。ポートクローズ後、エラークリアされます。また、それまで内部で保持していた送受信バッファのデータは 全て0クリアされることに留意して下さい。ポートクローズの呼び出し例は Figure 101 を参照して下さい。



7.11.6. <u>受信</u>

CX2500 は内部で RS232C の受信バッファ(サイズ: 512 バイト)を持っています。そのため、CX2500 はポートオープン後、メッセージを対向機から受信するとその内部バッファにメッセージを保存します。ただし、513 バイト以上のデータは内部で保有できません。その際は受信バッファオーバーフローエラーとなり SysComRead 関数を呼び出しても受信できなくなります。よって、ユーザーはバッファオーバーフローになる前に適度に SysComRead 関数を呼び出してバッファからメッセージを取り出して下さい。

受信バッファオーバーフローエラー含め、受信できなくなった場合、ポートクローズ→オープンをおこなって 再度受信を試みるようにして下さい。

) CTR	L_IO_MAIN 🗙 📄 TE	ST Cx2500Air	n î M	Device PRG	Cx2500D	o 👔 Library	Manager 🔣 Tas	k Configuration		-
٧	😼 🔹 🔸 🗙 PROGRAM CTRL_TO_MAIN					Þ					
	^	Scope	Name	Address	Data type	Initialization	Comment	Attributes			~ 💷
	15	🖗 VAR	lComCh		UDINT	2					
	16	🖗 VAR	RsltReadCom		UDINT	0					
	17	🖗 VAR	SucRecvBufNum		UDINT	0					
	18	🖗 VAR	ulBufSize		UDINT	20					
	19	🖗 VAR	RsltSendCom		UDINT	0					
	20	🖗 VAR	SucSendBufNum		UDINT	0					
	21	🖗 VAR	ComBuf		ARRAY[020] OF BYTE						
											~
<										>	
	47										^
	48										
	49 (* RS232C *)										
	50	//Recv	Cueller Cueller Deed	(hCarry)	1 Carron alternation	DD (Compute)			0		
	52	Suckecvbuinum := :	Syscom.Syscomkead	(ncom :=	IComen, poybuiler := .	ADR(COMBUI),	ulbize := ulbu	rsize, ullimeout :=	0, presuit := A	DR(RSITReadCom));	
	US 17 (DeltPeadCom = 0) AND (SucRecvBufNum > 0) THEN										
	54 Sucsendbuffum: = SysCom, SysComWrite (hCom := 1 ComCh. pbyBuffer := ADR(ComBuf), ulSize := SucRecvBufNum, ulTimeout := 0, pResult := ADR(RsltSendCom));										
	55 END IF										
	56 -										
	57										

Figure 102 RS232C 送受信呼出例(ST 言語)

7.11.7. <u>送信</u>

メッセージを送信(SysComWrite 関数の呼び出し)は受信バッファオーバーフローや通信エラーが発生してい ない場合におこなうことができます。エラー等で送信できなかった場合、ポートクローズ→オープンをおこなっ てから再送信を試みるようにして下さい。送信関数の呼び出し例は Figure 102 を参照して下さい。



7.12. <u>CAN</u>

CAN 通信を利用するには、下記いずれかを利用する必要があります。ユーザーアプリケーションで使用する 通信プロトコルに合わせ選択し使用して下さい。本節では、CANBus ライブラリを使用する場合について解説し ています。フィールドバスを使用する際は、本節ではなく8章を参照して下さい。

また、CANBus ライブラリ、各種フィールドバスについては、CODESYS オンラインヘルプにも記載があり ます。そちらも合わせて参照して下さい。

Table 46 CAN 利用手段 手段 摘要 フィールド ・ユーザーアプリケーションの通信プロトコルが J1939 や CANopen でおこなう場合に使用する。 バス ・ユーザーは初期設定や送受信で関数を呼び出す必要が無い。各種設定は IO ドライバのように専用画 (8 章参照) 面で簡単に設定できる。 ・各プロトコル用のファイル(下記)が用意できれば、それを読み込むだけで送受信メッセージが簡単に定 義・設定できる。 プロトコル名 ファイル名 J1939 DBC(CAN データベース)ファイル^{※14} CANopen EDS ファイル^{※15} CANBus ・CANBus ライブラリの関数をユーザーが手動で呼び出して CAN 通信制御をおこなう。 ライブラリ ・機能を使用するには、ライブラリ CANBus(CAA CanL2)が必要。使用できる関数は Table 48 参照。 ・ユーザーアプリケーションの通信プロトコルが J1939 や CANopen でない場合に使用する。 ・制御の自由度が高い分、ユーザーアプリケーションで初期設定を含め緻密な制御が求められる。

- ※14 DBC ファイルは CAN メッセージフレームの各データに識別名称などを結び付けることによって、生の CAN データ値を ユーザーが読みやすくすることができるものです。DBC ファイルは Vector 社製 CANalyzer(CANdb++)などの生成できる ツールを入手して作成する必要があります。
- ※15 EDS(Electronic Data Sheet)ファイルは CANopen などのフィールドネットワークで接続したいデバイスの接続情報(メッ セージ構成・IO 情報他)を含んだファイルのことです。こちらも CX2500 と通信させるデバイスの EDS ファイルをデバイ ス生産会社から入手するか、CANalyzer など EDS ファイルを作成できるツールを用いて独自に作成する必要があります。

チャネル番号	ボードレート
0~2	125kbps、250kbps、500kbps、1Mbps
3、4	125kbps、250kbps

Table 47 CAN 設定可能ボードレート



Table 48 CAN(CAA CanL2 ライブラリ) 関数一覧

機能区分	関数名	摘要
ドライバ関連	DriverOpenH	ヒープメモリを割り当てたドライバを作成し、CAN ネットワークをオープ
		ンする。
	DriverClose	ドライバ(ネットワーク)を削除する。
レシーバー	CreateIdAreaReceiver	エリア(CAN-ID 範囲指定)レシーバー(ハンドラ)を作成する。
関連	RegisterIdArea	エリアレシーバーに受信できる CAN-ID を追加する。
	UnregisterIdArea	エリアレシーバーから受信できる CAN-ID の条件を削除する。
	CreateMaskReceiver	マスクレシーバー(ハンドラ)を作成する。
	CreateSingleIdReceiver	特定 CAN-ID のみ受信するレシーバー(ハンドラ)を作成する。
	DeleteReceiver	レシーバーを削除する。
メッセージハン	CreateMessage	メッセージハンドラを作成する。
ドラ関連	CloneMessage	指定のメッセージを複製する。
	FreeMessage	指定のメッセージハンドラを削除する。
メッセージ取得	Read	受信したメッセージを取得する。
メッセージ送信	Write	メッセージを送信する。
ステータス取	GetDiagnosis	バスの診断(チャネルステータス)情報を取得する。
得関連	GetBaudrate	ボードレートを取得する。
	GetBusState	バスステータスを取得する。
	GetLostCounter	ロストメッセージの総数を取得する。
	GetReceiveCounter	受信したメッセージの総数を取得する。
	GetReceiveErrorCounter	受信エラーカウンタ値を取得する。
	GetReceivePoolSize	受信待ちのバッファ数を取得する。
	GetReceiveQueueLength	受信済みのメッセージ数を取得する。
	GetTransmitCounter	送信済みメッセージの総数を取得する。
	GetTransmitErrorCounter	送信エラーカウンタ値を取得する。
	GetTransmitPoolSize	送信設定待ちバッファ数を取得する。
	GetTransmitQueueLength	送信完了待ちメッセージ数を取得する。
	GetMessageDataPointer	指定メッセージのデータポインタを取得する。
	GetMessageId	指定メッセージの CAN-ID を取得する。
	GetMessageLength	指定メッセージのデータ長を取得する。
	GetMsgCount	受信バッファに残っているメッセージ数を取得する。
	GetNetId	指定したメッセージのネットワーク ID(チャネル番号)を取得する。
	LostMessage	指定 ID のロストメッセージ数を取得する。
	IsSendingActive	指定チャネルの送信バッファが Empty(送信可能状態)であるか判定す
		る。
	Is29BitIdMessage	指定のメッセージが拡張 ID のメッセージかどうか判定する。



	IsRTRMessage	指定メッセージがリモートフレームかどうか判定する。
	IsTransmitMessages	前回送信設定したメッセージの送信が完了したか判定する。
非対応	DriverOpenP	非対応
	DriverGetSize	
	GetTimeStamp	
	GetBusload	
	GetBusAlarm	
	ResetBusAlarm	
	DisableSyncService	
	EnableSyncService	
	GetCiAState	
	SetCiAState	



7.12.1. <u>列挙型</u>

CAN で使用する CAA CANL2I ライブラリで定義されている列挙型は下記の通りです。

型名	BUSSTATE				
摘要	CAN バスのステータスを	·表す。			
列挙子	名前	値	説明		
	UNKNOWN	0	未定義(ドライバ非オープン)		
	ERR_FREE	1	エラー無し		
	ACTIVE	2	エラーアクティブ		
	WARNING	3	エラーワーニング		
	PASSIVE	4	エラーパッシブ		
	BUSOFF	5	バスオフ		

型名	ERROR					
摘要	CAA CANL2I ライブラリ関数の返り値を表す。					
列挙子	名前	値	説明			
	NO_ERROR	0	エラー無し			
	NO_29BIT_ID	10202	29BIT-ID(拡張 ID)サポート非対応			
	WRONG_BAUDRATE	10203	ボードレート非対応			
	NO_MEMORY	10204	内部メモリの空き無し			
	INVALID_NETID	10205	ネットワーク ID(CAN チャネル番号)が不正値			
	INVALID_PRIORITY	10206	優先度が不正値			
	INVALID_DRIVER_HANDLE	10207	ドライバハンドラが不正値			
	INVALID_MESSAGE_HANDLE	10208	メッセージハンドラが不正値			
	INVALID_ID_HANDLE	10209	ID ハンドラが不正値			
	NO_DRIVER	10210	CANドライバが利用不可			
	SENDING_ERROR	10211	送信エラー			
	NO_SYNC_SERVICE	10212	SYNC(同期)サービス非対応			
	NO_SYNC_PRODUCER	10213	SYNC プロデューサー非対応			
	NO_SYNC_CONSUMER	10214	SYNC コンシューマー非対応			
	NO_SYNC_EVENT	10215	SYNC イベント非対応			
	NO_SYNC_WINDOW	10216	SYNC ウィンドウ非対応			
	NO_SYNC_FOREWARNTIME	10217	SYNC 早期警告非対応			
	WRONG_PARAM	10224	パラメータが不正値			
	INVALID_HANDLE	10249	ハンドラが不正値			



7.12.2. <u>構造体</u>

CAN で使用する CAA CANL2I ライブラリで定義されている構造体は下記の通りです。

型名	DIAGNOSIS_INFO						
摘要	CAN ネットワーク 診断(チャネルステータス)情報を表す構造体						
メンバ	型	名前	説明				
	UINT	uiBaudrate	ボードレート[kbps]				
	USINT	usiBusload	バス負荷率[%] <mark>(非対応)</mark>				
	BOOL	xBusAlarm	バスアラーム(<mark>非対応)</mark>				
	BUSSTATE	eBusState	バスステータス				
	CAA.COUNT	ctTxCounter	送信バッファセット完了総数				
	CAA.COUNT	ctTxErrorCounter	送信エラーカウンタ値				
	CAA.COUNT	ctRxCounter	受信済みメッセージ総数				
	CAA.COUNT	ctRxErrorCounter	受信エラーカウンタ値				
	CAA.COUNT	ctLostCounter	メッセージロスト総数				
	CAA.COUNT	ctFreeRxMessages	受信待ちバッファ数				
	CAA.COUNT	ctMessagesRxQueue	受信済みメッセージ数				
	CAA.COUNT	ctFreeTxMessages	送信設定待ちバッファ数				
	CAA.COUNT	ctMessagesTxQueue	送信完了待ちメッセージ数				


7.12.3. <u>関数</u>

CAN で使用できる関数について、それぞれ下記に示します。ただし、本製品非対応のものは除きます。

関数名	DriverOpenH					
摘要	ヒープメモリを	割り当てたドライバを	を作成し、CAN ネットワーク	?をオープンする。		
引数	型	名前	説明			
(INPUT)	USINT	usiNetId	ネットワーク ID(チャネル	番号:0~4)		
	UINT	uiBaudrate	ボードレート[kbps]			
	BOOL	xSupport29Bits	拡張 ID をサポートするた	Ŋ		
			TRUE	: サポートする		
			FALSE	:サポートしない		
	CAA.COUNT	ctMessages	ドライバに割り当てる送付	言バッファ数(最大 10。それ以上は 10 で制		
			限される)			
	POINTER TO	peError	NO_ERROR(0)	:オープン成功(エラー無し)		
			WRONG_BAUDRATE(102	203) :引数のボードレートが不正値		
			NO_MEMORY(10204)	:内部メモリ空き無し		
			INVALID_NETID(10205)	:引数のネットワークIDが不正値		
			NO_DRIVER(10210)	:ネットワークオープン失敗		
			INVALID_HANDLE(10249)) :ネットワークオープン失敗		
			MBM_NOMORE_MEMOR	Y(30103) : 内部メモリ空き無し		
引数	無し					
(IN_OUT)						
返り値	型	名前		説明		
(OUTPUT)	CAA.HANDLE	DriverOpenH		作成されたドライバハンドラ		

関数名	DriverClose								
摘要	作成したドライバ(ネットワーク)を削除する。								
引数	型	型 名前 説明							
(INPUT)	CAA.HANDLE		hDriver	ドライバハンドラ					
引数	無し								
(IN_OUT)									
返り値	型	名前		説明					
(OUTPUT)	ERROR	Driver	rClose	NO_ERROR(0)	:エラー無し				
				INVALID_HANDLE(10249)	:引数のハンドラが不正値				



関数名	CreateIdAreaReceiver								
摘要	エリア(CAN-ID 範囲指定)レシーバー(ハンドラ)を作成する。								
引数	型	名前	説明						
(INPUT)	CAA.HANDLE	hDriver	ドライバハンドラ						
	BOOL	xAlwaysNewest	非対応(FALSE を設定すること)						
	CB.EVENT	eEvent	非対応(0を設定すること)						
	BOOL	xEnableSyncWindow	非対応(FALSE を設定すること)						
	POINTER TO	peError	NO_ERROR(0)	:エラー無し					
	ERROR		NO_MEMORY(10204)	:内部メモリ空き無し又は					
				引数 xAlwaysNewest が					
				不正值					
			INVALID_DRIVER_HANDLE(10207)	:引数のハンドラが不正値					
			NO_DRIVER(10210)	:引数のドライバが不正値					
			WRONG_PARAM(10224)	:引数 eEvent が不正値					
			MBM_NOMORE_MEMORY(30103)	:内部メモリ空き無し					
引数	無し								
(IN_OUT)									
返り値	型	名前	説明						
(OUTPUT)	CAA.HANDLE	CreateIdAreaReceiver	作成されたレシーバーハンドラ						
備考	エリア指定のレ	シーバーは、拡張 ID が	非対応であることに注意。						



関数名	RegisterId	RegisterIdArea						
摘要	エリア指定	ミのレシー	バーに受	受信できる	CAN-ID をi	追加する。		
引数	型		名前		説明			
(INPUT)	CAA.HAN	DLE	DLE hReceiverId		レシーバーハンドラ			
	CL2I.COBID		CL2I.COBID cobIdSt		レシーバー	レシーバーに登録したい CAN-ID の最小値		
	CL2I.COB	ID	cobIdE	nd	レシーバー	に登録した	-い CAN-ID の最大値	
	BOOL		xRTRV	alue	リモートフレ	ノームサポ	— F	
					TRUE	:リモート	フレームを受信する	
					FALSE	:通常の	フレームを受信する	
	BOOL		xRTRM	ask	リモートフレ	ノームマス	ל	
					TRUE	:受信時	リモートフレームかどうか検証する	
					FALSE	:受信時	リモートフレームの検証をしない	
	BOOL		X29Bitl	dValue	FALSE を討	没定する		
	BOOL		X29BitIdMask		TRUE を設	定する		
	BOOL		xTransmitValue		TRUE	: CX2500)が送信したメッセージ	
					FALSE	: CX2500)が送信したメッセージではない	
	BOOL		xTrans	mit Ma sk	TRUE	:このレシ	レーバーで受信したメッセージが、CX2500	
						が送信	したメッセージであるか判定する	
					FALSE	:このレジ	レーバーで受信したメッセージが、CX2500	
						が送信	したメッセージであるか判定しない	
引数	無し							
(IN_OUT)								
返り値	型	名前		説明				
(OUTPUT)	ERROR	Register	IdArea	NO_ERR	OR(0)		:エラー無し	
				NO_29BI	T_ID(10202)		:引数x29BitIdValue•x29BitIdMaskが上記	
				設定値以外				
				INVALID_ID_HANDLE(10209)		(10209)	:引数のハンドラが不正値	
				WRONG	_PARAM(102	224)	:引数が不正値	
							(cobIdStart > cobIdEnd or ハンドラ不	
							正)	



関数名	UnregisterIdArea						
摘要	エリア指定のレ	シーバーから受信 ⁻	できる CAN-ID の条件を削除する。				
引数	型	名前	説明				
(INPUT)	CAA.HANDLE	hReceiverId	レシーバーハンドラ				
	CL2I.COBID	cobIdStart	レシーバーから削除したい CAN-ID の最小値				
	CL2I.COBID	cobIdEnd	レシーバーから削除したい CAN-ID の最大値				
	BOOL	xRTRValue	リモートフレームサポート				
			TRUE :リモートフレームを受信する				
			FALSE :通常のフレームを受信する				
	BOOL	xRTRMask	リモートフレームマスク				
			TRUE :受信時リモートフレームかどうか検証する				
			FALSE :受信時リモートフレームの検証をしない				
	BOOL	X29BitIdValue	FALSE を設定する				
	BOOL	X29BitIdMask	TRUE を設定する				
	BOOL	xTransmitValue	TRUE : CX2500 が送信したメッセージ				
			FALSE : CX2500 が送信したメッセージではない				
	BOOL	xTransmitMask	TRUE :このレシーバーで受信したメッセージが、CX2500				
			が送信したメッセージであるか判定する				
			FALSE :このレシーバーで受信したメッセージが、CX2500				
			が送信したメッセージであるか判定しない				
引数	無し						
(IN_OUT)							
返り値	型	名前	説明				
(OUTPUT)	ERROR	UnregisterIdArea	NO_ERROR(0) :エラー無し				
			INVALID_ID_HANDLE(10209) :引数のハンドラが不正値				
			WRONG_PARAM(10224) :引数が不正値				
			(cobIdStart > cobIdEnd)				



関数名	CreateMaskRe	CreateMaskReceiver						
摘要	マスクレシーバ	マスクレシーバー(ハンドラ)を作成する。						
引数	型	名前	1	説明				
(INPUT)	CAA.HANDLE	hDriv	/er	ドライバハン	バラ			
	CL2I.COBID	coble	dValue	CAN-ID				
	CL2I.COBID	coble	dMask	CAN-ID マク	マク			
	BOOL	xRTF	RValue	リモートフレ	ームサ ,	ポート		
				TRUE	:リモー	-トフレームを受信	する	
				FALSE	:通常(のフレームを受信 ⁻	する	
	BOOL	×RTF	RMask	リモートフレ	ームマス	マク		
				TRUE	:受信旺	寺リモートフレーム	かどうか検証する	
				FALSE	:受信旺	寺リモートフレーム	の検証をしない	
	BOOL	X29E	BitIdValue	拡張 ID サオ	ポ ー ト			
				TRUE	:拡張)	ID を受信する		
				FALSE	:標準]	ID を受信する		
	BOOL	X29BitIdMask		拡張 ID マス	くク			
				TRUE : 受信時拡張 ID かどうか検証する				
				FALSE	:受信田	受信時拡張 ID の検証をしない		
	BOOL	xTra	nsmitValue	TRUE	: CX25	0 が送信したメッセージ		
				FALSE	: CX25	00 が送信したメッ	セージではない	
	BOOL	xTra	nmitMask	TRUE	:このレ	シーバーで受信	ったメッセージが、CX2500	
					が送付	言したメッセージで	あるか判定する	
				FALSE	:このレ	シーバーで受信	ったメッセージが、CX2500	
					が送付	言したメッセージで	あるか判定しない	
	BOOL	xAlw	aysNewest	受信メッセー	ージの取	得順序		
				TRUE	:最新0	の受信メッセージが	から取得する。	
				FALSE	:最もさ	い受信メッセーシ	がら取得する。	
	CB.EVENT	eEve	nt	非対応(0 を	設定する	ること)		
	BOOL	xEna	bleSyncWindow	非対応(FAL	.SE を設	定すること)		
	POINTER TO	peEr	ror	NO_ERROR	(0)		:エラー無し	
	ERROR			NO_MEMOR	Y(10204	4)	:内部メモリ空き無し	
				INVALID_DF	RIVER_H	ANDLE(10207)	:引数のハンドラが不正値	
				NO_DRIVER	(10210)		:引数のハンドラが不正値	
				WRONG_PA	RAM(10	224)	:引数 eEvent が不正値	
				MBM_NOMC	DRE_MEI	MORY(30103)	:内部メモリ空き無し	
返り値	型		名前			説明		
(OUTPUT)	CAA.HANDLE		CreateMaskRec	eiver		作成されたレシ-	ーバーハンドラ	



関数名	CreateSingleIdReceiver						
摘要	特定 CAN-ID のみ受信するレシーバー(ハンドラ)を作成する。						
引数	型	名前	説明	説明			
(INPUT)	CAA.HANDLE	hDriver	ドライバハンドラ	,			
	CL2I.COBID	cobId	CAN-ID				
	BOOL	×RTR	リモートフレーム	、サポート			
			TRUE :リ	モートフレームを受信	言する		
			FALSE :通	自常のフレームを受信	する		
	BOOL	X29BitId	拡張 ID サポー	•			
			TRUE : 拢	、張 ID を受信する			
			FALSE : 標	[準 ID を受信する			
	BOOL	xTransmit	TRUE : C	X2500 が送信したメ	ッセージをこのレシーバー		
			7	受信する			
			FALSE : C	X2500 が送信したメ	ッセージはこのレシーバー		
			7	受信しない			
	BOOL	xAlwaysNewest	受信メッセージの	の取得順序			
			TRUE :最新の受信メッセージから取得する。				
			FALSE :	FALSE :最も古い受信メッセージから取得する。			
	CB.EVENT	eEvent	非対応(0 を設定	ミすること)			
	BOOL	xEnableSyncWindow	非対応(FALSE	を設定すること)			
	POINTER TO	peError	NO_ERROR(0)		:エラー無し		
	ERROR		NO_MEMORY(1	0204)	:内部メモリ空き無し		
			INVALID_DRIVE	R_HANDLE(10207)	:引数のハンドラが不正値		
			NO_DRIVER(102	210)	:引数のハンドラが不正値		
			WRONG_PARAM	M(10224)	:引数 eEvent が不正値		
			MBM_NOMORE	_MEMORY(30103)	:内部メモリ空き無し		
引数	無し						
(IN_OUT)							
 返り値	型	名前		説明			
(OUTPUT)	CAA.HANDLE	CreateSingleIdRe	ceiver	作成されたレシー	バーハンドラ		



関数名	DeleteReceiver							
摘要	レシーバーを削除する。							
引数	型		名	前		説明		
(INPUT)	CAA.HANDLE			eceiverId		レシーバーハンドラ		
引数	無し							
(IN_OUT)								
返り値	型	名前		説明				
(OUTPUT)	ERROR	DeleteReceive	er	NO_ERROR(0)	: I	ラー無し		
				INVALID_ID_HANDLE(10209)	:引	数のハンドラが不正値		
				NO_DRIVER(10210)	:引	数のハンドラが不正値		

関数名	CreateMessage						
摘要	メッセージハンドラる	を作成	 する。				
引数	型		名前	説明			
(INPUT)	CAA.HANDLE		hDriver	ドライバハンドラ			
	CL2I.COBID		cobID	CAN-ID			
	USINT		usiLength	メッセージのデータ長			
	BOOL		xRTR	リモートフレームの	のサポート		
				TRUE		:リモートフレーム	
				FALSE		:リモートフレームではない	
	BOOL X29Bi		X29BitID	拡張 ID のサポート			
				TRUE		:拡張 ID	
				FALSE NO_ERROR(0) INVALID_DRIVER_HANDLE(10207)		:標準 ID	
	POINTER TO ERR	OR	peError			:エラー無し	
						:引数のドライバハンドラが	
						不正値	
引数	無し						
(IN_OUT)							
返り値	型	名前	J		説明		
(OUTPUT)	CAA.HANDLE	Crea	ateMessage		作成されたメッセ	ージハンドラ	



関数名	CloneMessage							
摘要	指定のメッセージを	複製する。						
引数	型	名前	説明					
(INPUT)	CAA.HANDLE hMessage		メッセージハンドラ					
	POINTER TO ERROR peError		NO_ERROR(0)	:エラー無し			
			INVALID_MESSAGE_HANDLE(10208)		:引数のハンドラが			
					不正値			
引数	無し							
(IN_OUT)								
返り値	型	名前		説明				
(OUTPUT)	CAA.HANDLE	CloneMessage		複製されたメッセージハンドラ				

関数名	FreeMessage							
摘要	指定のメッセージハンドラを削除する。							
引数	型		名前	説明				
(INPUT)	CAA.HANDLE hMessage メッセージハンドラ							
引数	無し							
(IN_OUT)								
返り値	型	名前	説明					
(OUTPUT)	ERROR	FreeMessage	NO_ERROR(0)	:エラー無し				
			INVALID_MESSAGE_HANDLE(10208)	:引数のハンドラが不正値				



関数名	Read							
摘要	受信したメッセージを取得する。							
引数	型		名前	説明				
(INPUT)	CAA.HANDLE		hReceiverId	レシーバーハンドラ				
	POINTER TO CAA	COUNT	pctMsgLeft	Read 関数呼出し後、受信バッファに残っている受信済				
				のメッセージ数				
	POINTER TO ERROR		peError	NO_ERROR(0)	:エラー無し			
				INVALID_NETID(10205)	:引数のハンドラが不正値			
				INVALID_ID_HANDLE(10209)	:引数のハンドラが不正値			
				NO_DRIVER(10210)	:引数のハンドラが不正値			
引数	無し							
(IN_OUT)								
返り値	犚	名前		説明				
(OUTPUT)	CAA.HANDLE	Read		取得した受信メッセージのハン	バラ			
備考	Read 後、受信デー	タを保存し	たら、FreeMess	age 関数を呼び出し、リソースを	解放すること。			



関数名	Write								
摘要	メッセージを送信する。								
引数	型	名前	説明						
(INPUT)	CAA.HANDLE	hDriver	ドライバハンドラ						
	CAA.HANDLE	hMessage	メッセージハンドラ						
	USINT	usPriority	送信優先度(1~8、数字が小さいほど)	憂 先 度高)					
	BOOL	xEnableSyncWindow	非対応(FALSE を設定すること)						
引数	無し								
(IN_OUT)									
返り値	型	名前	説明						
(OUTPUT)	ERROR	Write	NO_ERROR	:エラー無し					
			INVALID_NETID(10205)	:引数のドライバハンドラ					
				が不正値					
			INVALID_PRIORITY(10206)	:引数の優先度が不正値					
			INVALID_DRIVER_HANDLE(10207)	:引数のドライバハンドラ					
				が不正値					
			INVALID_MESSAGE_HANDLE(10208)	:引数のメッセージハンド					
				ラが不正値					
			NO_DRIVER(10210)	:引数のドライバハンドラ					
				が不正値					



関数名	GetDiagnosis							
摘要	バスの診	断(チャネルステー	·タス)情	報を取得する。				
引数	型			名前	説明			
(INPUT)	CAA.HAN	DLE		hDriver	ドライバハンドラ			
	POINTER	TO DIAGNOSIS_I	NFO	pDiagnosisInfo	診断情報			
引数	無し							
(IN_OUT)								
返り値	型	名前	説明					
(OUTPUT)	ERROR	GetDiagnosis	NO_ER	ROR(0) : 17	- ラー無し			
備考	引数 hDriver が不正値(例:ドライバハンドラではない値)の場合でも、返り値 GetDiagnosis には							
	NO_ERRC	PR が入る。また、p	Diagnos	sisInfo にも不定値がセットされてしま	ミうことに注意。これは CX2500 で			
	はなくCO	DESYS 自体の動	作仕様で	である。				



関数名	GetBaudrate						
摘要	ボードレートを取得	する。					
引数	型	名	前	説明			
(INPUT)	CAA.HANDLE	hD	Driver	ドライバハンドラ			
	POINTER TO ERROR peError		Error	NO_ERROR(0)		:エラー無し	
				INVALID_DRIVER	_HANDLE(10207)	:引数のハンドラが不正値	
				NO_DRIVER(1021	0)	:引数のハンドラが不正値	
引数	無し						
(IN_OUT)							
返り値	型	名前			説明		
(OUTPUT)	UINT	GetBau	udrate		ボードレート[kbps]	

関数名	GetBusState						
摘要	バスステータスを取	収得す	る。				
引数	型		名前	説明			
(INPUT)	CAA.HANDLE		hDriver	ドライバハンドラ			
	POINTER TO ERROR		peError	NO_ERROR(0)		:エラー無し	
				INVALID_DRIVER	_HANDLE(10207)	:引数のハンドラが不正値	
				NO_DRIVER(1021	0)	:引数のハンドラが不正値	
引数	無し						
(IN_OUT)							
返り値	型	名育	ţ,		説明		
(OUTPUT)	BUSSTATE	Get	BusState		バスステータス		



関数名	GetLostCounter						
摘要	ロストメッセージの総数を取得する。						
引数	型	名前	説明	説明			
(INPUT)	CAA.HANDLE	hDriver	hDriver ドライバハンドラ				
	POINTER TO ERRO	OR peError	NO_ERROR(0)		:エラー無し		
			INVALID_DRIVER	INVALID_DRIVER_HANDLE(10207) :引数のハンドラが不正値			
			NO_DRIVER(1021	0)	:引数のハンドラが不正値		
引数	無し						
(IN_OUT)							
返り値	型	名前		説明			
(OUTPUT)	CAA.COUNT	GetLostCounte	r	ロストメッセージ糸	送数		

関数名	GetReceiveCounter							
摘要	受信したメッセージ	の総	数を取得する	3 。				
引数	犚		名前	説明	説明			
(INPUT)	CAA.HANDLE		hDriver	ドライバハンドラ				
	POINTER TO ERROR		peError	NO_ERROR(0)	:エラー無し			
			INVALID_DRIVER	_HANDLE(10207)	:引数のハンドラが不正値			
				NO_DRIVER(1021	0)	:引数のハンドラが不正値		
引数	無し							
(IN_OUT)								
返り値	型	名育	ĴŢ		説明			
(OUTPUT)	CAA.COUNT	Get	ReceiveCour	nter	受信メッセージ総	数		



関数名	GetReceiveErrorCounter						
摘要	受信エラーカウンタ値を取得する。						
引数	型		名前	説明			
(INPUT)	CAA.HANDLE		hDriver	ドライバハンドラ			
	POINTER TO ERROR		peError	NO_ERROR(0)	NO_ERROR(0) :エラー無し		
				INVALID_DRIV	ER_HANDLE(10207)	:引数のハンドラが不正値	
				NO_DRIVER(10210) :引数のハンドラが不正		:引数のハンドラが不正値	
引数	無し						
(IN_OUT)							
返り値	型	名前	ī		説明		
(OUTPUT)	CAA.COUNT	GetF	ReceiveError	Counter	受信エラーカウンタ	直	

関数名	GetReceivePoolSize							
摘要	受信待ちのバッファ	·数を []]	取得する。					
引数	型		名前	説明				
(INPUT)	CAA.HANDLE		hDriver	ドライバハンド	5			
	POINTER TO ERROR		peError	NO_ERROR(0)		:エラー無し		
				NO_DRIVER(10	0210)	:引数のハンドラが不正値		
				INVALID_HAND	DLE(10249)	:引数のハンドラが不正値		
引数	無し							
(IN_OUT)								
返り値	型	名前	Ţ		説明			
(OUTPUT)	CAA.COUNT	Getl	ReceivePoolS	Size	受信待ちバッファ数			



関数名	GetReceiveQueueLength								
摘要	受信済みのメッセージ数を取得する。								
引数	型		名前	説明					
(INPUT)	CAA.HANDLE		hDriver	ドライバハンドラ					
	POINTER TO ERROR		peError	NO_ERROR(0)	:エラー無し				
				INVALID_NETID(10205) :引数のハンドラが不正値					
				NO_DRIVER(10210)	:引数のハンドラが不正値				
				INVALID_HANDLE(10249)	:引数のハンドラが不正値				
引数	無し								
(IN_OUT)									
返り値	型	名前		説明					
(OUTPUT)	CAA.COUNT	GetR	eceiveQueueLength	受信済みメッセージ数					

関数名	GetTransmitCounter						
摘要	送信済みメッセージ	の総数	数を取得す	3.			
引数	型	:	名前	説明			
(INPUT)	CAA.HANDLE	ł	hDriver	ドライバハンドラ			
	POINTER TO ERROR		peError	NO_ERROR(0)		:エラー無し	
				INVALID_DRIVER_	HANDLE(10207)	:引数のハンドラが不正値	
				NO_DRIVER(1021	0)	:引数のハンドラが不正値	
引数	無し						
(IN_OUT)							
返り値	型	名前			説明		
(OUTPUT)	CAA.COUNT	GetTr	ransmitCou	nter	送信済みメッセー	ジ総数	



関数名	GetTransmitErrorCounter						
摘要	送信エラーカウンタ	植を取	収得する。				
引数	型		名前	説明			
(INPUT)	CAA.HANDLE		hDriver	ドライバハンドラ			
	POINTER TO ERROR		peError	NO_ERROR(0)		:エラー無し	
				INVALID_DRIVER	_HANDLE(10207)	:引数のハンドラが不正値	
				NO_DRIVER(1021	0)	:引数のハンドラが不正値	
引数	無し						
(IN_OUT)							
返り値	型	名前			説明		
(OUTPUT)	CAA.COUNT	GetT	ransmitError	rCounter	送信エラーカウンタ	值	

関数名	GetTransmitPoolSize						
摘要	送信設定待ちバッフ	ファ数	を取得する。				
引数	型		名前	説明			
(INPUT)	CAA.HANDLE		hDriver	ドライバハンド	5		
	POINTER TO ERROR peError		peError	NO_ERROR(0)		:エラー無し	
				NO_DRIVER(10	0210)	:引数のハンドラが不正値	
				INVALID_HAND	DLE(10249)	:引数のハンドラが不正値	
引数	無し						
(IN_OUT)							
返り値	型	名前	Ţ		説明		
(OUTPUT)	CAA.COUNT	Get	TransmitPool	Size	送信設定待ちバッ	ファ数	



関数名	GetTransmitQueu	GetTransmitQueueLength				
摘要	送信完了待ちメッ	送信完了待ちメッセージ数を取得する。				
引数	犚		名前	説明		
(INPUT)	CAA.HANDLE		hDriver	ドライバハンドラ		
	POINTER TO ERF	ROR	peError	NO_ERROR(0)	:エラー無し	
				INVALID_NETID(10205)	:引数のハンドラが不正値	
				NO_DRIVER(10210)	:引数のハンドラが不正値	
				INVALID_HANDLE(10249)	:引数のハンドラが不正値	
引数	無し					
(IN_OUT)						
返り値	型	名前		説明		
(OUTPUT)	CAA.COUNT	A.COUNT GetTransmitQueue		送信完了待ちバッファ数		
		Leng	th			

関数名	GetMessageDataPointe	GetMessageDataPointer			
摘要	指定メッセージのデータ	タポインタを取得	身する。		
引数	型	名前	説明		
(INPUT)	CAA.HANDLE	hMessage	メッセージハンドラ		
	POINTER TO ERROR	peError	NO_ERROR(0)		:エラー無し
			INVALID_MESSAGE_H	IANDLE(10208)	:引数のハンドラが
					不正値
引数	無し				
(IN_OUT)					
返り値	型	名前		説明	
(OUTPUT)	POINTER TO CL2I.DATA GetMess		ageDataPointer データポインタアドレス		<i>х</i>



関数名	GetMessageId	GetMessageId				
摘要	指定メッセージの	CAN-I	Dを取得する。			
引数	型		名前	説明		
(INPUT)	CAA.HANDLE		hMessage	メッ・	セージハンドラ	
	POINTER TO ERF	ROR	peError	NO_	ERROR(0)	:エラー無し
				INV	ALID_MESSAGE_HANDLE(10208)	:引数のハンドラ
						が不正値
引数	無し					
(IN_OUT)						
返り値	型	名前			説明	
(OUTPUT)	CL2I.COBID GetMessageId			CAN-ID		

関数名	GetMessageLength	GetMessageLength				
摘要	指定メッセージのテ	ータ長を	取得する。			
引数	型		名前	説明		
(INPUT)	CAA.HANDLE		hMessage	メッセー	ジハンドラ	
	POINTER TO ERROR		peError	NO_ERROR(0) :		:エラー無し
				INVALID	_MESSAGE_HANDLE(10208)	:引数のハンドラ
						が不正値
引数	無し					
(IN_OUT)						
返り値	型	名前			説明	
(OUTPUT)	USINT	GetMe	ssageLength		データ長	



関数名	GetMsgCount	GetMsgCount				
摘要	受信バッファに残っ	ているメッセージ数	を取得する。			
引数	型	名前	説明			
(INPUT)	CAA.HANDLE	hReceiverId	レシーバーハ	シドラ		
	POINTER TO ERRO	DR peError	NO_ERROR(0))	:エラー無し	
			INVALID_NET	TID(10205)	:引数のハンドラが不正値	
			INVALID_ID_H	ANDLE(10209)	:引数のハンドラが不正値	
			NO_DRIVER(10210)		:引数のハンドラが不正値	
引数	無し					
(IN_OUT)						
返り値	型	名前		説明		
(OUTPUT)	CAA.COUNT GetMsgCount			受信メッセージ残数		

関数名	GetNetId	GetNetId				
摘要	指定したメッセージ	ジのネッ	トワーク ID(チャネル	番号)を取得する。		
引数	型		名前	説明		
(INPUT)	CAA.HANDLE		hMessage	メッセージハンドラ		
	POINTER TO ERROR		peError	NO_ERROR(0)	:エラー無し	
				INVALID_MESSAGE_HANDLE(10208)	:引数のハンドラ	
					が不正値	
引数	無し					
(IN_OUT)						
返り値	型	名前		説明		
(OUTPUT)	USINT	GetNe	tId	ネットワーク ID(チャネル番号:0~4)		



関数名	LostMessages						
摘要	指定 ID のロストメッ	指定 ID のロストメッセージ数を取得する。					
引数	型		名前	説明			
(INPUT)	CAA.HANDLE		hReceiverId	レシーバーハンドラ			
	POINTER TO ERRO	DR	peError	NO_ERROR(0)	:エラー無し		
				INVALID_ID_HANDLE(10209)	:引数のハンドラが不正値		
引数	無し						
(IN_OUT)							
返り値	型	名前		説明			
(OUTPUT)	UINT	LostN	lessages	ロストメッセージ数			

関数名	IsSendingActive					
摘要	指定チャネルの	送信バッ	ファが Empty(送信	言可能状態)であるか判定する。		
引数	型		名前	説明		
(INPUT)	CAA.HANDLE		hDriver	ドライバハンドラ		
	POINTER TO E	RROR	peError	NO_ERROR(0)	:エラー無し	
				INVALID_DRIVER_HANDLE(10207)	:引数のハンドラが不	
					正値	
引数	無し					
(IN_OUT)						
返り値	뢵	名前		説明		
(OUTPUT)	BOOL	IsSendir	gActive	TRUE	:送信不可状態(エラー	
					など)	
				FALSE	:送信可能状態	



関数名	Is29BitIdMessage					
摘要	指定のメッセージが拡張	指定のメッセージが拡張 ID のメッセージかどうか判定する。				
引数	型	名前	説明			
(INPUT)	CAA.HANDLE	hMessage	判定したいメッセージハンドラ			
	POINTER TO ERROR	peError	NO_ERROR(0)	:エラー無し		
			INVALID_MESSAGE_HANDLE	:引数のハンドラが		
			(10208)	不正値		
引数	無し					
(IN_OUT)						
返り値	型	名前	説明			
(OUTPUT)	BOOL	Is29BitIdMessage	TRUE	:拡張 ID		
			FALSE	:拡張 ID ではない		

関数名	IsRTRMessage	IsRTRMessage				
摘要	指定メッセージが	リモート	フレームかど	うか判定する。		
引数	ᆋ		名前	説明		
(INPUT)	CAA.HANDLE		hMessage	判定したいメッセージハンドラ		
	POINTER TO ER	ROR	peError	NO_ERROR(0)		:エラー無し
				INVALID_MESSAGE_HANDLE(102	08)	:引数のハンドラが
						不正値
引数	無し					
(IN_OUT)						
返り値	型	名前		説明		
(OUTPUT)	BOOL	IsRTR	Message	TRUE	:リモート	マレーム
				FALSE	:リモート	-フレームではない



関数名	IsTransmitMessa	IsTransmitMessage				
摘要	レシーバーで受	信したメ	ッセージが CX	2500 から送	信されたメッセージであるかを	判定する。
引数	型		名前	説明		
(INPUT)	CAA.HANDLE		hMessage	判定したし	ヽメッセージハンドラ	
	POINTER TO EF	RROR	peError	NO_ERRC	R(0)	:エラー無し
				INVALID_	MESSAGE_HANDLE(10208)	:引数のハンドラが
						不正値
引数	無し					
(IN_OUT)						
返り値	型	名前		説明		
(OUTPUT)	BOOL	IsTran	smitMessage	TRUE	:CX2500 から送信されたメッ+	セージである
				FALSE	:CX2500 から送信されたメッ+	セージではない
備考	CX2500 が送信I	したメット	ュージを受信す	るには、各	レシーバー作成の際に ID 等と	共に xTransmit 系引数
	(xTransmit•xTra	nsmitVa	lue•xTransmitl	Mask)を TR	UE にして作成する必要がある。	



7.12.4. <u>初期設定</u>

CAN 通信をおこなう際、ユーザーは用途に応じて下記のハンドラを作成する必要があります。なお、ドライ バハンドラについては他のどのハンドラよりも必ず先に作成しなければなりません。

各ハンドラの作成方法については7.12.4.1項以降に後述します。

名称摘要ドライバハンドラ・作成すると、CX2500 の CAN チャネルの制御が可能になる。
・これを作成しないと、レシーバーハンドラやメッセージハンドラの作成ができないことに注意。
・使用するチャネルに対してそれぞれ 1 つ作成。レシーバーハンドラ・CX2500 が CAN メッセージを受信するためのレシーバーを指す。
・作成後 CX2500 内部で CAN メッセージの受信が自動でおこなわれる。
・1 つのドライバハンドラ(1 チャネル)に対して複数作成可能。メッセージハンドラ・CX2500 で CAN メッセージの送受信するために必要なメッセージ構造体。
・1 つのドライバハンドラ(1 チャネル)に対して複数作成可能。

Table 49 CAN 各種ハンドラ



CCOT-24-016 Rev.1 7.12.4.1. <u>ドライバハンドラの作成</u>

DriverOpenP

ユーザーアプリケーションで CAN を使用するには、CX2500の使用したい CAN チャネル番号に対して1つ ドライバを作成する必要があります。ドライバ(ハンドラ)作成によって、CAN メッセージの送受信制御やバスス テータス情報管理が可能になります。ドライバハンドラは他のハンドラを作成する前に必ず作成して下さい。

	Table 50 ドライバハンドラ 作成用関数一覧
関数名	摘要
DriverOpenH	CX2500 のヒープメモリを割り当て、CANドライバを作成する。

CX2500のRAMメモリを割り当て、CANドライバを作成する。

	1	(* Tribicliantian *)
	2	
	4	IF DCANINIT = FALSE THEN
	3	
	4	(* Open Driver *)
	5	hCanDriver := CL2.DriverOpenH(usiNetId := 1, uiBaudrate := 250, xSupport29Bits := TRUE, ctMessages := ctTxMsg, peError := ADR(eCanErr));
	6	
	7	(* Make Receiver *)
Β	8	IF eCanErr = ERROR.NO_ERROR THEN
Β	9	hReceiver := CL2.CreateMaskReceiver(
	10	hDriver:= hCanDriver,
	11	cobldValue := 0,
	12	cobldMask := 0,
	13	<pre>xRTRValue := 0,</pre>
	14	<pre>xRTRMask := 0,</pre>
	15	x29BitIdValue := FALSE,
	16	x29BitIdMask := TRUE,
	17	xTransmitValue := FALSE,
	18	xTransmitMask := TRUE,
	19	xAlwaysNewest := FALSE,
	20	<pre>eEvent := CB.EVENT.NO_EVENT,</pre>
	21	xEnableSyncWindow := FALSE,
	22	peError := ADR(eCanErr)
	23);
	24	
	25	//Initialization Completed
	26	bCanInit := TRUE;
	27	END_IF
	28	

Figure 103 ドライバハンドラおよびレシーバーハンドラ 作成・呼び出し例(ST 言語)

この例でのレシーバーハンドラは全ての標準フォーマット CAN-ID を受信する設定



7.12.4.2. レシーバーハンドラの作成

CAN メッセージを受信するには、ユーザーアプリケーションにてレシーバーを作成する必要が有ります。レ シーバーには下記の種類があり、いずれかを作成した後に CX2500 内部でメッセージ受信が自動的におこなわれ ます。アプリケーションの用途に合わせレシーバー(ハンドラ)を選択・作成して下さい(呼び出し例は Figure 103)。 なお、レシーバーは各チャネル複数作成可能です。ただし、各チャネル最大 16 個までです(フィールドバスも同 様。もし定義してもメッセージを受信できない場合は送受信メッセージの種類数を調整して下さい)。レシーバー 作成に関する注意は Table 51 を参照して下さい。

名称	作成用関数名	摘要
ID エリア指定レシーバー	CreateIdAreaReceiver	・ユーザーが指定した範囲内の CAN-ID のみ受信をおこなう
		レシーバー。
		・CAN-ID の範囲指定はレシーバー作成後、RegisterIdArea
		関数を呼び出すことでおこなうことができる。
		・受信できる CAN-ID の範囲は RegisterIdArea 関数の引数
		cobIdStart~cobIdEndとなる。
		・ID 範囲内の CAN-ID1 つにつき作成可能レシーバー数をラ
		ンタイム内部で1つ消費してしまうことに注意。レシーバーハ
		ンドラ自体は、範囲指定した CAN-ID が複数の場合でも、1 つ
		しか作成されない。この1つのハンドラに範囲指定した CAN-
		ID を満たすメッセージが受信・格納される。
		・拡張 ID フォーマットは非対応。
マスクレシーバー	CreateMaskReceiver	・複数の CAN-ID を受信可能なレシーバー。
		・エリア指定レシーバーと異なり、範囲ではなくビットマスクで
		受信できる CAN-ID をフィルタする。例は次ページ参照。
		・x29BitIdMaskを FALSE に設定してこのレシーバーを作成し
		た場合、x29BitIdMaskをTRUEで設定した場合に比べ、作成
		可能レシーバー数を1つ多く消費することに注意。ただし、レ
		シーバーハンドラ自体は、いずれの場合でも1つしか作成さ
		れない。この1つのハンドラに Mask 条件をクリアした CAN-
		ID を満たすメッセージが受信・格納される。
シングル ID レシーバー	CreateSingleIdReceiver	・特定の 1 つの CAN-ID のみ受信をおこなうレシーバー。

Table 51 CAN レシーバーの種類



(例)マスクレシーバー

CAN-IDの Value 及び Mask(CreateMaskReceiver 関数の引数)を下記の通り設定した場合、受信 ID1~3の 内、レシーバーが受信・受信バッファに保存できるのは検証ビットの Value と値が同じ受信 ID3 のみ。

項目	Bit											ID 値	
	11	10	9	8	7	6	5	4	3	2	1	0	
cobIdValue	0	1	1	1	0	1	1	1	1	0	0	1	0x779
cobIdMask	0	0	0	1	0	0	1	0	0	0	1	1	0x123
受信検証 Bit	×	×	×	0	×	×	0	×	×	×	0	0	-
受信 ID1	0	1	1	0	1	1	0	1	1	0	1	0	0x6DA
受信 ID2	0	1	1	1	0	1	1	1	1	0	1	1	0x77B
受信 ID3	0	1	0	1	0	1	1	0	0	1	0	1	0x565

赤色:検証NG、青色:検証OK

7.12.4.3. メッセージハンドラの作成

CX2500から CAN メッセージを送信する場合、メッセージ情報を格納するメッセージハンドラを事前に作成 する必要が有ります。メッセージハンドラは CreateMessage 関数を呼び出すことで作成できます。作成したメ ッセージハンドラは Write 関数を呼び出してメッセージを送信するか、FreeMessage 関数を呼び出すことで解 放されることに留意して下さい。

_	🖬 Device 📄 CAN_Ctrl 🗙 🖬 J1939_Manager 👘 CANbus 🖷 J1939_CX2500 👘 J1939_ECU 👘 Library Manager								
٩	PROGRAM CAN_Ctrl								
	^	Scope	Name	Address	Data type	Initialization	Comment	Attributes	
	9	🔷 VAR	hTxMsg		CAA.HANDLE	0			
	10	< VAR	pTxData		POINTER TO CL2I.DATA	0			
	11	< VAR	i		USINT	0			
<									
	44	(* S	Gend Data *)						
	45	//Cre	ate TX-MSG St	tructure					
Ξ	46	hTxMs	g := CL2.Crea	ateMessage	(
	47	h	Driver:= hCan	Driver,					
	48	c	obID := UINT#	:16#110,					
	49	u	siLength := 8						
	50	х	RTR := FALSE,						
	51	х	29BitID := FA	LSE,					
	52	р	eError := ADF	(eCanErr));				

Figure 104 メッセージハンドラ 作成・呼出例(ST 言語)



7.12.5. <u>受信</u>

CAN メッセージの受信は、レシーバーハンドラ作成後、CX2500 内部で自動的におこなわれます。受信したメ ッセージをユーザーアプリケーションで取り出すには、Read 関数を呼び出す必要が有ります。Read 関数を呼び 出すと、受信メッセージハンドラが返り値として得られ、メッセージ内の情報をアプリケーションで使用できま す。使用後は FreeMessage 関数を呼び出してメッセージハンドラ(リソース)を解放するようにして下さい。

なお、CX2500 内部で受信したメッセージをユーザーアプリケーションで取得しない状態が続くと、受信バッファが満杯になりそれ以上メッセージを受信・保存することができなくなる場合があります。定期的に受信メッセージの取得をおこなって下さい。

					PROGRAM CAI	N_Ctrl		
Scope	Name	Address	Data type	Initialization	Comment	Attributes		
VAR	eCanErr		ERROR	ERROR.NO_ERROR				
VAR	hCanDriver		CAA.HANDLE	0				
🔷 VAR	ctTxMsg		CAA.COUNT	10				
VAR	hReceiver		CAA.HANDLE	0				
VAR	ctRxLeft		CAA.COUNT	5				
VAR	hRxMsg		CAA.HANDLE	0				
VAR	pRxData		POINTER TO CL2I.DATA	0				
VAR	hTxMsq		CAA.HANDLE	0				
hRxMs IF hF	sg := CL2.Rea RxMsg <> CAA. //If CX2500 R	d(hReceive gc_hINVALI ecv CAN me	rId:=hReceiver, pctM D THEN essage, get it's data	sgLeft:=ADR(ctRxI	eft), peErr	<pre>:or:=ADR(eCanErr));</pre>		
hRxMs IF hI	sg := CL2.Rea RxMsg <> CAA. //If CX2500 R	d(hReceive gc_hINVALI ecv CAN me	erId:=hReceiver, pctM D THEN essage, get it's data	sgLeft:=ADR(ctRxI	eft), peErr	<pre>:er:=ADR(eCanErr));</pre>		
hRaMs IF hi	sg := CL2.Rea RxMsg <> CAA. //If CX2500 R DRxData := CL	d(hReceive gc_hINVALI eov CAN me 2.GetMessa	rId:=hReceiver, pctM D THEN ssage, get it's data sgeDataPointer(hMessa	sgLeft:=ADR(ctRxI ge := hRxMsg, pe	eft), peErr	cor:=ADR(eCanErr)); R(eCanErr));		
hRxMs IF hI END_J CL2_I	sg := CL2.Rea RxMsg <> CAA. //If CX2500 R pRxData := CL IF FreeMessage(h)	d(hReceive gc_hINVALI ecv CAN me 2.GetMessa Message :=	rId:=hReceiver, pctM D THEN ssage, get it's data geDataPointer(hMessa	sgLeft:=ADR(ctRxI ge := hRxMsg, peB	eft), peErr	<pre>cor:=ADR(eCanErr)); R(eCanErr));</pre>		
hRxMs IF hI END_J CL2.I hRxMs	sg := CL2.Rea RxMsg <> CAA. //If CX2500 R DRxData := CL FreeMessage (hi sg := CAA.ge)	d(hReceive gc_hINVALI ecv CAN me 2.GetMessa Message := hINVALID;	<pre>rId:=hReceiver, pctM D THEN wssage, get it's data ugeDataPointer(hMessa + hRxMsg);</pre>	sgLeft:=ADR(ctRxI ge := hRxMsg, peI	eft), peErr	<pre>cor:=ADR(eCanErr)); R(eCanErr));</pre>		
hRxMs IF hI END_J CL2.I hRxMs pRxDa	sg := CL2.Rea RxMsg <> CAA //If CX2500 R pRxData := CL FreeMessage(hl sg := CAA.gc_ ata := CAA.gc	d(hReceive gc_hINVALI ecv CAN me 2.GetMessa Message := hINVALID; _pNULL;	<pre>xId:=hReceiver, pctM D THEN ssage, get it's data geDataPointer(hMessa = hRxMsg);</pre>	sgleft:=ADR(ctRxI ge := hRxMsg, peI	Seft), peErr	cor:=ADR(eCanErr)); R(eCanErr));		
hRodds IF hF END_J CL2_I hRodds pRxDa	sg := CL2.Rea ArMsg <> CAA. //If CX2500 R RxData := CL FreeMessage(h) sg := CAA.gc_] ata := CAA.gc	d(hReceive gc_hINVALI ecv CAN me 2.GetMessa Message := hINVALID; _pNULL;	<pre>vrId:=hReceiver, pctM D THEN sssage, get it's data geDataPointer(hMessa = hRxMsg);</pre>	sgLeft:=ADR(ctRxI ge := hRxMsg, peB	Seft), peErr	<pre>cor:=ADR(eCanErr)); {(eCanErr));</pre>		
hRodMs IF hI END_1 CL2.I hRodMs pRxDa	sg := CL2.Rea AuMsg <> CAA. //If CX2500 R RxData := CL FreeMessage (h) sg := CAA.gc_) ata := CAA.gc	d(hReceive gc_hINVALI eov CAN me 2.GetMessa Message := hINVALID; _pNULL;	vId:=hReceiver, pctM D THEN sssage, get it's data geDataPointer(hMessa = hRxMsg);	sgLeft:=ADR(ctRxI ge := hRxMsg, peI	eft), peErr	<pre>cor:=ADR(eCanErr)); {(eCanErr));</pre>		
hRadMs IF hS END_1 CL2.I hRadMs pRxDa	sg := CL2.Rea RaMsg <> CAA. //If CX2500 A RxData := CL FreeMessage(hl sg := CAA.gc_] ata := CAA.gc Send Data *	<pre>d (hReceive gc_hINVALI eov CAN me 2.GetMessa Message := hINVALID; _pNULL;)</pre>	vId:=hReceiver, pctM D THEN sssage, get it's data geDataPointer(hMessa = hRxMsg);	sgLeft:=ADR(ctRxI ge := hRxMsg, peI	eft), peErr	<pre>cor:=ADR(eCanErr)); (eCanErr));</pre>		
hRadMs IF hS END_1 CL2.I hRadMs pRxDa	sg := CL2.Rea RxMsg <> CAA. //If CX2500 R pRxData := CL freeMessage(hl sg := CAA.gc_] ata := CAA.gc_ Send Data * seate TX-MSG S seate CX-MSG S	<pre>d (hReceive gc_hINVALI eov CAN me 2.GetMessa Message := hINVALID; _pNULL;) tructure</pre>	<pre>vId:=hReceiver, pctM D THEN sssage, get it's data geDataPointer(hMessa = hRxMsg);</pre>	sgLeft:=ADR(ctRxI ge := hRxMsg, peB	eft), peErr	<pre>cor:=ADR(eCanErr)); (eCanErr));</pre>		
hRodds IF hS END_1 CL2.1 hRodds pRxDa (* 2 //Cre hToda	sg := CL2.Read RxHSg <> CAA., //If CX2500 R RxData := CL If TreeMessage(h sg := CAA.gc_) ata := CAA.gc Send Data * eate TX-MSG S sg := CL2.Cre	<pre>d (hReceive gc_hINVALI eov CAN me 2.GetMessage Message := hINVALID; _pNULL;) tructure ateMessage Dp:uor</pre>	<pre>xId:=hReceiver, pctM D THEN ssage, get it's data geDataPointer(hMessa = hRxMsg);</pre>	sgLeft:=ADR(ctRxI ge := hRxMsg, peH	eft), peErr	<pre>cor:=ADR(eCanErr)); R(eCanErr));</pre>		
hRadds IF hi END 1 CL2.I hRadds pRxDa (* 3 //Crv hTadds 1	sg := CL2.Read RxMSg <> CAA., //If CX2500 = CL RxData := CL FreeMessage(h ata := CAA.gc_) ata := CAA.gc_) ata := CAA.gc Send Data * Seate IX-MSG S sg := CL2.Cre Driver:= hCau	<pre>d (hReceive gc_hINVALI eov CAN me 2.GetMessag Message := hINVALID; _pNULL;) tructure ateMessage nDriver, is=110</pre>	<pre>xId:=hReceiver, pctM D THEN ssage, get it's data geDataPointer(hMessa = hRxMsg);</pre>	sgLeft:=ADR(ctRxI ge := hRxMsg, peH	eft), peErr	<pre>cor:=ADR(eCanErr)); R(eCanErr));</pre>		
hRodfis IF his r END_1 CL2.1: hRodfis pRxDa (* 3 //Cr4 hTxHs h c	sg := CL2.Read RxMSg <> CAA. //If CX2500 PCAA. RxData := CL TreeMessage(h) sg := CAA.gc_] ata := CAA.gc_] ata := CAA.gc Send Data * Seate TX-MSG S sg := CL2.Cre Driver:= hCau cobID := UINT ssiLength :=)	<pre>d (hReceive gc_hINVALI eov CAN me 2. GetMessa Message := hINVALID; </pre>	<pre>vrId:=hReceiver, pctM D THEN vssage, get it's data geDataPointer(hMessa + hRxMsg); + hRxMsg);</pre>	sgLeft:=ADR(ctRxI ge := hRxMsg, pel	eft), peErr	<pre>cor:=ADR(eCanErr)); t(eCanErr));</pre>		

Figure 105 CAN 受信処理例(ST 言語)



7.12.6. <u>送信</u>

CAN メッセージの送信は Write 関数を呼び出すことでおこなうことができます。なお、送信にはメッセージ ハンドラが必要になりますので、CreateMessage 関数を事前に呼び出しメッセージ情報を設定の上送信処理を おこなって下さい。

なお、CreateMessage 関数で作成したメッセージハンドラは Write 関数が正常に処理された場合(返り値が NO_ERROR の場合)、内部で自動的に FreeMessage 関数が呼び出されて解放されます。この場合、ユーザは FreeMessage を呼び出す必要はありません。Write 関数が失敗した場合(返り値が NO_ERROR 以外の場合)、ユーザ自身で必ず FreeMessage 関数を呼び出してメモリを解放して下さい。下図 64~67 行目がその処理例にな ります。

また、メッセージハンドラとして使用した変数は使用後、必ず下図(70、71 行目)のようにリセット処理をして 下さい。リセットせずに続けて使用した場合、予期せぬ挙動になる場合があります。

<u>_</u> []	J 193	39_CX2500	👔 Library	/ Manager	CANbus		1939_ECU	Dev	/ice	Cx2500Ain		J1939_Manager	CAN_Cti	ri x
۰ (4 4	X						PRO	GRAM CAN_	Ctrl				
	9	Scope	Name hTxMsg	Address	Data type CAA.HANDLE		Initializatio 0	n	Comment	Attributes				
	11	VAR	pTxData		POINTER TO CL	2I.DATA	0							
/		VAR			USINI		U							
` <u></u>								N	-					
Ξ	45 46 47 48 49 50 51	//Cr hTxM	<pre>eate TX-MSG St sg := CL2.Creat hDriver:= hCar cobID := UINT usiLength := St xRTR := FALSE, x29BitID := FALSE</pre>	tructure ateMessage nDriver, \$16\$110, 3, ALSE,										
	52		peError := ADB	R(eCanErr));									
Э	54	IF h	TxMsg <> CAA.g	gc_hINVALI	D THEN	r (bMassa	ao io hTuM			(0ConFre))				
Ξ	57 58 59 60		IF pTxData <> FOR i := (pTxDat END_FOR	CAA.gc_pN) TO 7 DO ta^[i] := ;	ULL THEN pRxData^[i];		.	- <u>,</u> , <u>,</u>						
Э	51 52 53 54 55 55 56 57 58 59		END_IF //Send DATA eCanErr := CL2 IF eCanErr <> CL2.FreeMe END_IF //Clear_bandl/	2.Write(hD CL2.ERROR assage(hMe	river:= hCan .NO_ERROR TH ssage := hTxl	Driver, EN Msg);	hMessage :	= hTxMsg	, usiPrior	ity := 1,	xEnable	SyncWindow	:= FALSE);	
	70		hTxMsg := CAA.	gc_hINVAL	ID;									
	71	END	pTxData := CAM	A.gc_pNULL	4									Γ

Figure 106 CAN 送信処理例(ST 言語)



7.13. <u>タイマカウンタ</u>

本機能は、ランタイム起動(CX2500 に電源投入)してからの経過時間を取得できます。この経過時間は、IDE でのユーザー操作によるアプリケーション停止中(ブレークポイント等)も時間が加算され続けることに留意して 下さい。本機能の使用には、ライブラリ SysTime(SysTimeCore)が登録されている必要が有ります。使用できる 関数は下記の通りです。

また、SysTime ライブラリについては CODESYS オンラインヘルプにも記載があります。そちらも合わせて 参照して下さい。

Table 52 タイマカウンタ(SysTimeCore ライブラリ) 関数一覧

機能区分	関数名	摘要
経過時間取得	SysTimeGetMs	ランタイムが起動してからの経過時間[ms]を取得する。
	SysTimeGetUs	ランタイムが起動してからの経過時間[µs]を取得する。
非対応	SysTimeGetNs	非対応。

7.13.1. <u>型の別名定義</u>

タイマカウンタで使用する SysTimeCore ライブラリで定義されている型は下記の通りです。

Table 53	SysTimeCore ライブラリ	型の別名定義一切	覧
----------	-------------------	----------	---

名称	ベース型名	摘要
SYSTIME	ULINT	経過時間の取得などに使用される。

7.13.2. <u>関数</u>

タイマカウンタで使用できる関数について、それぞれ下記に示します。ただし、本製品非対応のものは除きま す。

関数名	SysTimeGetMs						
摘要	ランタイムが起露	ランタイムが起動してからの経過時間[ms]を取得する。					
引数	無し	無し					
(INPUT)							
引数	無し	無し					
(IN_OUT)							
返り値	型	名前	説明				
(OUTPUT)	UDINT SysTimeGetMs 経過時間[ms]						



関数名	SysTimeGetUs							
摘要	ランタイムが起動してからの経過時間[µs]を取得する。							
引数	無し	無し						
(INPUT)								
引数	型		名前			説明		
(IN_OUT)	SYSTIME		pUsTin	ne	経過時間[µs]			
返り値	型名前		説明					
(OUTPUT)	RTS_IEC_RESULT SysTimeGet		Js ERR_OK(0x0) : 取得		:取得	成功		



7.14. <u>RTC</u>

ユーザーは CX2500 内蔵の RTC(リアルタイムクロック)の計時及び時刻を設定・取得することが可能です。機能を使用するには、当社独自ライブラリ CX2500RtcLibrary(Cx2500 Rtc Library)が登録されている必要が有ります。使用できる関数は下記の通りです。

Table 54 RTC(CX2500RtcLibrary ライブラリ) 関数一覧

機能区分	関数名	摘要
カレンダ設定	SetDate_Rtc	RTC ヘカレンダ(時刻)を設定する。
カレンダ取得	GetDate_Rtc	RTC からカレンダ(現在時刻)を取得する。

7.14.1. <u>列挙型</u>

RTC で使用する CX2500RtcLibrary ライブラリで定義されている列挙型は下記の通りです。

型名	EN_RTC_DAY_OF_WEEK_IEC		
摘要	曜日を表す。		
列挙子	名前	值	説明
	DW_SUN	0	日曜日
	DW_MON	1	月曜日
	DW_TUE	2	火曜日
	DW_WED	3	水曜日
	DW_THU	4	木曜日
	DW_FRI	5	金曜日
	DW_SAT	6	土曜日

Κεικι

CCOT-24-016 Rev.1

7.14.2. <u>構造体</u>

RTC で使用する CX2500RtcLibrary ライブラリで定義されている構造体は下記の通りです。

型名	ST_RTC_TIME_IEC			
摘要	時刻をあらわす構造体。			
メンバ	型	名前	説明	
	USINT(059)	ucSec	秒(0~59)	
	USINT(059)	ucMin	分(0~59)	
	USINT(023)	ucHour	時間(0~23)	
	EN_RTC_DAY_OF_WEEK_IEC	ucWeek	曜日(0(Sunday)~7(Saturday))	
	USINT(131)	ucDay	日(1~31)	
	USINT(112)	ucMonth	月(1~12)	
	USINT(099)	ucYear	年(例:2024の場合は下二桁の 24を設定)	

7.14.3. <u>関数</u>

RTC で使用できる関数について、それぞれ下記に示します。

関数名	SetDate_Rtc			
摘要				
引数	型	名前	説明	
(INPUT)	ST_RTC_TIME_IEC	stRtcTime	設定したい時刻	
引数	無し			
(IN_OUT)				
返り値	型	名前	説明	
(OUTPUT)	RTS_IEC_RESULT	SetDate_Rtc	ERR_OK(0x0)	:設定成功
			ERR_FAILED(0x1)	:設定失敗(引数異常・RTC エラー)



関数名	GetDate_Rtc			
摘要	RTC から現在時刻を取得する。			
引数	無し			
(INPUT)				
引数	型	名前	説明	
(IN_OUT)	ST_RTC_TIME_IEC	stRtcTime	取得した時刻	
返り値	型	名前	説明	
(OUTPUT)	RTS_IEC_RESULT	GetDate_Rtc	ERR_OK(0x0)	:取得成功
			ERR_FAILED(0x1)	:取得失敗(RTC エラー)
備考	時刻が設定されていない、若しくは長期放電により時刻が消失した場合は取得できる値が不定値となる			
	ことに注意。			



7.14.4. 初期状態とバックアップ時間

RTCには初期設定(時刻設定)がされていません。そのため、ユーザーはアプリケーション内で任意のタイミン グにカレンダを設定する必要が有ります。

また、時刻を RTC に設定した後、CX2500 に長い間電源を入れないと、RTC へ設定した時刻が失われます。 電源を落としてから時刻が失われるまでのバックアップ時間の目安については、CX2500 の機能仕様書を参照し て下さい。

なお、初期状態・時刻が失われた状態で時刻を取得すると取得値は不定値になりますので留意して下さい。



8. フィールドバスについて

8.1. 概要

CX2500 は CAN フィールドバス(J1939・CANopen)を使用することができます。フィールドバスは、通信機能を有する機器間で情報をやり取りするために接続するバスシステムを指します。

CODESYS では、CAN ライブラリ(7.12節)を使っても CAN 通信をおこなうことができます。ただ、フィール ドバス機能を使うと、下記のような利点があります。

CANopen については、サンプルプロジェクトが CODESYS Store にて無料配布されていますので、ダウンロ ードして参考コードとして確認することができます。また、マニュアルに記載無き情報については CODESYS オ ンラインヘルプなどを参照して下さい。

【主な利点】

- ユーザーアプリケーションの通信プロトコルが J1939 や CANopen でおこなう場合に便利。
- ユーザーは初期設定や送受信で CAN ライブラリのように関数を呼び出す必要が無い。各種設定は IO ドラ イバのように専用画面で簡単に設定できる。
- 各プロトコル用のファイル(下記)が用意できれば、それを読み込むだけで同一バス内のデバイス向け送受信
 メッセージが簡単に定義・設定できる。
 - ▶ J1939 : DBC ファイル^{※14}
 - ▶ CANopen : EDS ファイル^{※15}



8.2. <u>共通設定</u>

ここでは、プロトコル問わず、フィールドバスを使う際の共通設定について記します。フィールドバスを使用 する際、機能ドライバのようにプロジェクトのデバイスにフィールドバスアイテムを紐づける必要があります。 紐づける必要があるのは以下の通りです。手順については 8.2.1 項を参照して下さい。

名称	デフォルト名称		摘要	
	J1939	CANopen		
バスマスタ	CANbus	CANbus	フィールドバスを使用するチャネルや CAN の基	
			本的な通信設定・ステータスモニタができる。	
プロトコルマネージャー	J1939_Manager	CANopen_Manager ^{※16}	各プロトコルの基本設定やデバイス用の	
			DBC/EDS ファイルなどをインポート/エクスポート	
			することができる。	
CAN デバイス	J1939_ECU	(各デバイスに依る)	バス内のノードとなるデバイス情報。デバイスが	
			おこなう送受信メッセージの設定およびモニタを	
			おこなうことができる。	

Table 55 フィールドバス デバイスに紐づけるアイテム

※16 CAN デバイスがローカルデバイス(スレーブ)の場合は不要。その場合、CAN デバイスはバスマスタから直接紐づける。


8.2.1. <u>フィールドバスの紐づけ</u>

ここではフィールドバスセットの紐づけ手順を示します。例として J1939 でおこなっていますが、CANopen でも同様の手順をおこなうことで設定できます。

(1) まず、バスマスタの紐づけをおこないます。デバイスウィンドウの「Device」にカーソルを合わせ右クリッ クして下さい。

SampleProject_Cx2500.project - O File Edit View Project Bi	ODESYS uild Online Debug Tools W	índow Help	× c –
🛅 📽 🖶 🚳 🗠 🗠 👗 🐚 I	🗈 🗙 (西)结 🐴 🤮 (川) 🏦	🕅 🎕 🔚 🛅 • 🔓 🕅 Application [Device: PLC Logic] 🔹 😂 👀 🔸	■ ※ (目 51 位 11 谷 中 開 甘 ひ
Devices - 4 ×	Device X Cx2500Pwr	POU_CHA	
Completing Crasses Completing Crasse	Communication Settings	Scan Network Gateway - Device -	^
B O Applicatio	e (CX2500) tions		
DUT_SAMPLE	Backup and Restore		
Library Manager	Files		•••••••••••••••••••••••••••••••••••••••
POU_CtriA (PRG	Log	Gateway Sateway	MyDevice (active)
B I Task Configurat	PLC Settings	IP-Address: localbest	Device Name:
= ₩ MainTask -@] PLC_PR	PLC Shell	Port	Device Address:
E Task1	Users and Groups	1217	Target ID:
롄 POU_FE 역 Trace_MainTask	Access Rights		17E5 0001
PersistentVars Visualization Mar	Symbol Rights		4096
Visualization	Licensed Software Metrics		TOKYO KEIKI INC.
Cx2500Cntin (Cx2500Cn	Task Deployment		Target Version: 3.5.18.30
Cx2500Do (Cx2500Do)	Status		
Cx2500Pvm (Cx2500Pv	Information		
	Messages - Total 0 error(s), 0 warning	s), 0 message(s)	- + ×
		 O error(s) 0 warning(s) 0 message(s) 	×
K >	Description		Project Object Position

Figure 107 デバイスウィンドウ Device の選択

(2) 表示されるコンテキストメニューから「Add Device...」を選択して下さい。



Figure 108 Device Add Device の選択



(3) 「Add Device」ウィンドウが表示されるので、デバイス一覧の中から「CANbus」を選択して「Add Device」 ボタンを押して下さい。

Arises Arises Arises Arises Arises Arises Arises Arises Arises Arises Arises Arises Arises Arises Arises Arises Arises Arises Arises Arises A	Add Device						
Action Action Action Action	Iame CANbus						
Append device best device best device device best device	Action						
store for a full text search Vendor Name Vendor Vendor Vendor Vendor Vendor Vendor <th>Append device Insert devi</th> <th>ce O Plug device O</th> <th>Update device</th> <th></th> <th></th> <th></th> <th></th>	Append device Insert devi	ce O Plug device O	Update device				
Name Version Description Image: Mediamous: Image: Mediamous: Society Control Society Coctex Society Control Socie	String for a full text search	Vendor	<all vendors=""></all>				
	Name	Vendor		Version	Description		
the CANbas the CANbas S-Smart Software Solutions Grieft 4.1.1.0 Needed for all fieldbases which communicate over the CANBus, e.g. CAtopen or 31939. Sector CANBas Software Solutions Grieft 4.1.1.0 CAHBus on a netX device The CANBus, e.g. CAtopen or 31939. Sector CANBas Software Solutions Grieft 4.1.1.0 CAHBus on a netX device Device the CANBus, e.g. CAtopen or 31939. Sector CANBas Software Solutions Grieft Sector CANBas	B Miscelaneous						
	Fieldbuses						
gi lett CAPbas S - Smart Software Solutions GmbH 4.1.1.0 CAPbas on a netX device * @ Ethomet Adapter * Ethomet Adapter * @ Ethomet Adapter * # homes * @ ThomeSolutions * * * @ ThomeSolutions Solutions GmbH * * Categories: Coluda * * * Weaker: S - Smart Software Solutions GmbH * * Categories: Coluda * * * * Weaker: S - Smart Software Solutions GmbH * * * * Order Number: Othomet Solutions GmbH * * * * * Order Number: Othomet Solutions Communicate over the CAPBus, e.g. CARepen or 11939. * * * Uppend Selected device as last child of * * * * * (You can select another target node in the nangetor whil	CANbus	3S - Smart Softwar	re Solutions GmbH	4.1.1.0	Needed for all fieldbusses which communicate over the CANBus, e.g. CANopen or J1939.		
	NetX CANbus	3S - Smart Softwar	re Solutions GmbH	4.1.1.0	CANbus on a netX device		
Extended Adapter	Bud EtherCAT						
	EtherNet/IP						
Here Wedus Profile	Home&Building Automat	tion					
Profiles	😸 💶 Modbus						
Event and the set of the set	Profibus						
Induced Version 1: Constraints of the constraint of the constraints of the constrain	Profinet IO	all varians (for avaarts		outdated us	winan		00
Immer CANbus Vendor, S5: Smit Software Solutions (mbH Categories: CANbus Version: 4.1.0 Order Number: Description: Needed for all fieldbusses which communicate over the CANbus, e.g. CANopen or 31939. Uppend selected device as last child of Nerice (ros can select another target node in the navigator while this window is open.) (for can select another target node in the navigator while this window is open.)	er er en en er	an reisions (rei experts i		obtobico rei	200		
Visual: 20 - Jank 3 Joint 2 Joi	Name: CANbus	en Enhalinge Carble					
Wyssian: 4.1.10 Similar 4.1.10 Order Number: Description: Needed for all fieldbusses which communicate over the CANkup, e.g.: CANkpen or 31939. uppend selected device as last child of levice Image: Similar 4.1.10 (rou can select another target node in the navigator while this window is open.) Image: Similar 4.1.10	Categories: CANbus	re solutions amon					
Order Number: Description: Needed for all fieldbusses which communicate over the CAVBus, e.g. CAVlopen or 31939. uppend selected device as last child of Perice @ (fou can select another target node in the navigator while this window is open.)	Version: 4.1.1.0						2
Uppend selected device as last child of Period Period (Fou can select another target node in the navigator while this window is open.)	Order Number:	Faldburger which community	picate over the CAN	Dr a a CA	Monan or 11020		-
Append selected device as last child of bevice (You can select another target node in the navigator while this window is open.)	Description: Needed for al	i neidousses which commu	nicate over the CAN	ious, e.g. CA	Nopen or 31333.		
uppend selected device as last child of Perice (fou can select another target node in the navigator while this window is open.)							
Uppend selected device as last child of Device Ø (You can select another target node in the navigator while this window is open.)							
Uppend selected device as last child of Verice 0 (You can select another target node in the navigator while this window is open.)							
Append selected device as last child of Jevice (You can select another target node in the navigator while this window is open.)							
Device (Fou can select another target node in the navigator while this window is open.) Add figures Flores	Append selected device as last	child of					
(You can select another target node in the navigator while this window is open.)	Device						
Add Pawing Close	(You can select another target)	et node in the navigator v	while this window is	s open.)			
						Add	Close

Figure 109 Add Device ウィンドウ CANbus の選択

(4) 「Add Device」ウィンドウを閉じると、デバイスウィンドウ上に CANbus が追加されています。

SampleProject_Cx2500.project* - CC File Edit View Project Buil Edit View Project Buil Edit View Project Buil Edit View Project Buil	DDESYS Id Online Debug Tools V 을 X 44 등 44 등 1 및 위	indow Hep 清 滑 L色 L 1 - C 一冊 Application [Device: PLC Logic] • OG	- 0 ×
■ SampleProject_0.2500 ■ Device (CL300) ■ Dut_SAMPLE ■ Dut_SAMPLE ■ OU_CL40 Reg ■ POU_FR (Reg) ■ OU_CL40 Reg ■ POU_FR (Reg) ■ OU_CL40 Reg ■ POU_FR (Reg) ■ POU_FR (Reg)	Communication Settings Applications Backup and Restore Files Log PLC Settings PLC Shell Users and Groups Access Rights Symbol Rights Licensed Software Metrics Task Deployment Status	Scan Network Gateway - Device -	
CANbus (CANbus)			~
CAN	messages - total u error(s), 0 warning bus (CANbus) Description		Position

Figure 110 デバイスウィンドウ CANbus 追加



(5) 次に、プロトコルマネージャー(J1939_Manager)を追加します。追加した「CANbus」にカーソルを合わせ 右クリックして下さい。



Figure 111 デバイスウィンドウ CANbus の選択

(6) 表示されるコンテキストメニューから「Add Device...」を選択して下さい。

Supplex Device (actor) Communication Settings Scan Network Cateriary • Device • Image: Supplex Device (actor) Applications Applications Applications Image: Supplex Device (actor) Backup and Restore Files Image: Supplex Device (actor) Image: Device Figure (actor) Backup and Restore Files MyDevice (actor) Image: Device (actor) Image: Device Figure (actor) Device (actor) Device (actor) Image: Device (actor) Image: Device (actor) Image: Device (actor) Device (actor) Device (actor) Image: Device (actor)	Devices	→ ₽ X	M Device X M Cx250	Pwm				•
PC Log: Applications Backup and Restore Backup and Restore Files Backup and Restore POU_CISA Repso Files DOU_FOR (PRG) Device Address: DOU_FOR (PRG) PC Settings POU_FOR (PRG) PC Settings PL Settings P-Address: Port Device Address: Port Device Address: Port Port Port Port Port Device Address: Port	SampleProject_Cx Device (CX250	2500 💌	Communication Settings	Scan Network Gateway - Device -				^
Image: Construction	≕ 🗐 PLC Logic ≅- 🚺 Appli ⊮- 🛄 D	cation JT_SAMPLE	Applications Backup and Restore		•			
IP POUC (CAAL PRG) 0g IP POUC (CAAL PRG) 0g IP Address: Device (Active) IP Address: Device Name: IP Address: Device Address: IP Address: Target Type: IP Address: Target Vendor: IP Address: Target Vendor: IP Address: Target Version: IP Address: JP Address: IP Address: JP Address: IP Address: Target Version: IP Address: JP Address: IP Address: JP Address: IP Address: JP Address: <	🎒 G 🚹 U 📄 PL	/L brary Manager .C_PRG (PRG)	Files		Gateway	••		
Paddress: Device Name: Port Device Address: Port Device Address: Port 03913002 Part 1217 P		DU_Ctria (PRG DU_FB (PRG)	Log	Gateway-	~	MyDevice (active)	~	
Port Device Address: Oppy 1217 Past 0501_5002 Past 125002 Past<	⊟-∭3 та ⊜-₫	ask Configurati MainTask	PLC Settings	IP-Address localhost		Device Name: MyDevice		
Image: Discover and Delete Paraget ID: Image: Discover and Delete Target ID: Image: Discover and Delete Target Npe: Image: Discover and Delete Target Net:		X Cut	PLC Shell	Port: 1217		Device Address: 0301.5002		
Image: Constraint of the sector of the se	B-\$	Copy				Target ID: 17E5 0001		
Refactoring Refactoring Weil Properties Cossonar Add Object		× Delete				Target Type: 4095		
CX25004 Image: Very Decision CX25004 Add Object CX25004 Target Version: CX25004 Add Object CX25004 Image: Version:	- 🛃 Vz	Refacto	oring •	-		Target Vendor:		
	Cx2500Air	Add Ol	bject			Target Version:		
Classer inset Device	- Cx25000	Add De	evice			3.5.18.30		
		Insert L	Device					
G CASOOW Disable Device	Cx2500Pw	Disable	e Device					~
Update Device Vide Device		Update	e Device	ing(s), 0 message(s)				- 4 ×
Edd Object Wrth		Edit Ot	bject With	- O error(s) 😗 0	varning(s) 9 0 message(s)	*		
FORT LUMADOUDO	Modules Services	Import	mappings from CSV			. Toject	- open	
c Earl O mapping From CSU.	•	Export	mappings to CSV		Last build: 😋 0 😗 0	Precompile 🧹 😭	Project user: (nobody)	() (à

Figure 112 CANbus Add Device の選択



(7) 「Add Device」ウィンドウが表示されるので、デバイス一覧から「J1939_Manager」を選択して「Add Device」 ボタンを押して下さい。

🕤 Add Device										
Name J1939_Manager										
Action Append device	OPlug device O	Jpdate device								
String for a full text search	Vendor	<all vendors=""></all>								, ,
Name Generation States	Vendor 35 - Smart Software	Versic	n Description	1						
Group by category Display all V Mame: J1939 Manager Vendors 3 - Smart Software Sc Categories: J1939 Manager Version: 3.5.17.0 Order Humber: Description: J1939 Manager	versions (for experts o	nly) 🗌 Display outdated	d versions			 	 		₩ ₩0	
Append selected device as last chil CANbus (You can select another target no	d of ode in the navigator w	while this window is open.)		 	 	 	 			
								Add E	Device	Close

Figure 113 Add Device ウィンドウ J1939_Manager の選択

(8) 「Add Device」ウィンドウを閉じると、デバイスウィンドウの「CANbus」下に「J1939_Manager」が追加 されています。

~ ∓ X	Device X M Cx2500Pw	POU_CHA	
mpleProject_Cx2500	Communication Settings	Scan Network Gateway • Device •	
B-O Application	Applications		
DUT_SAMPLE GVL	Backup and Restore		
Library Manager	Files		
POU_CtrlA (PRG)	Log	Sateway MyDevice (active)	
Solution	PLC Settings	IP-Address: Device Name:	
⊨ 🥵 MainTask 🕮 PLC_PRG	PLC Shell	Port: Device Address:	
POU_CtriA	Users and Groups	1217 0301.5002	
POU_FB	Access Rights	18/get ID: 17/E5 0001	
Trace_main ask	Symbol Rights	Target Type: 4096	
Visualization Manager	Licensed Software Metrics	Target Vendor: TOKYO KERI INC.	
Cx2500Ain (Cx2500Ain)	Task Deployment	Target Version: 3.5.18.30	
Cx2500Do (Cx2500Do)	Status		
- 🚮 Cx2500Frqin (Cx2500Frqin) - 🚮 Cx2500Pvm (Cx2500Pvm)	Information		
- File and the second s	<		

Figure 114 デバイスウィンドウ J1939_Manager の追加



(9) 次に、フィールドバスチャネルのネットワークに接続するデバイスを登録します。このデバイスは CX2500 の通信相手又は CX2500 にあたります。追加した「J1939_Manager」にカーソルを合わせ右クリックして下 さい。

🖬 🍜 🗠 🗠 👗 🖻 🛍 🗙 👫	43 4월 44 비 위 위 위 위 (백) (백)	🔤 - 🗂 🛗 Application [Device:	PLC Logic] 🝷 😋 👀 📦 🔳 👋	: [고 91 4] 8 [후 栗 루 장	
	X Device X Cx2500P	wm	.C_PRG		
SampleProject_Cx2500	Communication Settings	Scan Network Gateway - D	evice +		
E Q Application	Applications				
CONT_SAMPLE GVL	Backup and Restore				
Library Manager	Files			•	
POU_CtrlA (PRG)	Log		Gateway	✓ MyDevice (active) ✓]
E-S Task Configuration	PLC Settings		IP-Address: localhost	Device Name: MyDevice	
PLC_PRG	PLC Shell		Port:	Device Address:	
œ POU_CtriA ⊟-⊗ Task1	Users and Groups		121/	Target ID:	
- 셴 POU_FB - 데 Trace_MainTask	Access Rights			17E5 0001	
PersistentVars Visualization Manager	Symbol Rights			4096	
Visualization	Licensed Software Metrics			Target Vendor: TOKYO KEIKI INC.	
Cx2500Cntin (Cx2500Cntin)	Task Deployment			Target Version: 3.5.18.30	
Cx2500Der (Cx2500Der)	Status				
Cx2500Frqin (Cx2500Frqin)	Information				_
[]] [J1939_Manager (J1939_Manager)	Messages - Total 0 error(s), 0 warnin	ng(s), 0 message(s)			•
		- O em	or(s) 🕐 0 warning(s) 🔇 0 messag	e(s) 🗙 💥	
	Description			Project Object Po	sition

Figure 115 デバイスウィンドウ J1939_Manager の選択

(10) 表示されるコンテキストメニューから「Add Device...」を選択して下さい。

SampleProject_Cv2500_project* - CODESYS File Edit View Project Build Onlin E Eit View Project Build Onlin E Eit View Project Build Onlin	ne Debug Tools Window 14 1일 🍓 1일 제 🧐 개 개	Help	05 03 → ± 42 (3 93 43 43 8	- a × ▼
Devices • • • • • • • • • • • • • • • • • • •	tions and Restore	POUCHA PC_RG	MyDevice (active) MyDevice (active) Device Name: hyDevice	
Edit Object Edit Object With	warning(s), 0 mes	sage(s)	sage(s) 🗙 💥	v ậ X
Edit IO mapping Modules Devices Import mappings Export mappings	from CSV to CSV	Last b	Project uild: ⊙ 0 ⊕ 0 Precompile ✓ 😭	Project user: (nobody)

Figure 116 J1939_Manager Add Device の選択



(11) 「Add Device」ウィンドウが表示されるので、デバイス一覧から「J1939_ECU」を選択して「Add Device」 ボタンを押して下さい。なお、CANopen の場合、CODESYS にデフォルトでインストール(登録)されている デバイスとは別に、CANopen デバイス(EDS ファイル)を新しく CODESYS-IDE に追加する場合は、別紙 「CX2500Codesys_UserManual_ForSetup」4.4節の要領で EDS ファイルのインストール(登録)をおこなっ てください。

Inter Adom Adom Adom Adom Origing for & fall lent search Version Adom Version Adom Version Adom Version Adom Version Adom Version	Add Device										>
Action Action	Name J1939_ECU										
Append selected device as last child of Jissy ECU Cor select another target node in the navigator while this window is open.) Cor control of the contr	Action										
String for a full text search Vendor Name Vendor Vendor Venion Description Image: String for a full text search Vendor Vendor <t< th=""><th>Append device</th><th>device OPlug device</th><th>) Update device</th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th></t<>	Append device	device OPlug device) Update device								
Name Verdor Version Description Image: State State State State State State State State Image: State State State State State State State State Image: State State State State State State State State Image: State State State State State State State State Image: State State State State State State State State Image: State State State State State State State State Image: State State State State State State State State Image: State State State State State State State State Image: State State State State State State State State Image: State State State State State State Image: State State State State State State State Image: State	String for a full text search	Vendo	<all vendors=""></all>								~
	Name	Vendor		Version	Description						
	E- I Fieldbuses										
Image: 1999_ECU 35 - Smart Software Solutions Goldet 3.5. 16.0 31999 ECU Image: 1999_ECU Display all versions (for experts only) Display outdated versions Image: 1999_ECU Neme: 1999_ECU Image: 1999_ECU Vereidor: 3.5. Solid Order Humber; 7?? Image: 1999_ECU Append selected device as last child of 1999_Hours Image: 1999_ECU Image: Image	E- SM 31939										
C Group by category Display all versions (for experts only) Display outdated versions C Group by category: Display all versions (for experts only) Display outdated versions C Memory: 5: Source Solutions GorbH C Version: 1:539 ECU Append selected device as last child of 11393_Hanager (You can select another target node in the navigator while this window is open.)	J1939_EC	J 3S - Smart Software S	olutions GmbH	3.5.16.0	31939 ECU						
C Group by category Display all versions (for experts only) Display outdated versions C Group by category Display all versions (for experts only) Display outdated versions Mame: 11939_ECU Vendor: 11939_ECU Version: 31.6.0 Order Number: 7? Description: 31939_ECU Append selected device as last child of J1939_Hanager (Version: as ledet another target node in the navigator while bits window is open.)											
@ Group by category Display all versions (for experts only) Display outdated versions @ Mane: 11939-ECU Versions 3.5.6.0 Order Number; 7?? Description: 31939 ECU Person: 31939 ECU Append selected device as last child of 11939 Manager Order vanisher target node in the navigator while this window is open.)						 	 	 			
Image: 1399-ECU Image: 1399-ECU Vendor: 5: 3195-ECU Image: 1399-ECU Version: 5: 5: 5: 5: 5: 5: 5: 5: 5: 5: 5: 5: 5:	Group by category Dis	play all versions (for expert	s only) 🗌 Displa	iy outdated	d versions						
Append selected device as last child of 11393_Hanager () You can select another target node in the newspator while this window is open.)	Mame: J1939_ECU										
Version: 33. 51.60 Image: Comparison of	Categories: J1939 EC	ftware Solutions GmbH U									
Append selected device as last child of 11393_Hanager (You can select another target node in the navigator while this window is open.)	Version: 3.5.16.0									5	2
Append selected device as last child of J1339_Manager (You can select another target node in the navigator while this window is open.)	Order Number: ??? Description: J1939 E0	zu									-
Append selected device as last child of J1939_Hanager (You can select another target node in the navigator while this window is open.)											
Append selected device as last child of J1339_Hanager (You can select another target node in the navigator while this window is open.)											
Append selected device as last child of J1339_Manager (You can select another target node in the navigator while this window is open.)											
Append selected device as last child of J1939_Manager (You can select another target node in the navigator while this window is open.)											
1939_Hanager Image: International Control of the newloater while this window is open.) Image: Image	Append selected device as	last child of		_							
(You can select another target node in the navigator while this window is open.)	J1939_Manager										
Lidd Danies Place	(You can select another	target node in the navigato	r while this window	v is open.)						_	
	L					 	 	 	Add D	ievice	Close

Figure 117 Add Device ウィンドウ J1939_ECU の選択



(12) 「Add Device」ウィンドウを閉じると、「J1939_Manager」下に「J1939_ECU」が追加されています。 なお、同じデバイスを複数追加した場合は図のように「デバイス名_(番号)」の名称で追加されます。デバイ ス名称は各デバイスのコンテキストメニューのプロパティからおこなうことができます。

• •	X Device X Cx2500F	wm POU_CtrlA PLC_PRG			
SampleProject_Cx2500	Communication Settings	Scan Network Gateway - Device -			
E PLC Logic	Applications				
- O Application - O DUT_SAMPLE - O UT_SAMPLE - O O	Backup and Restore	· · · · · · · · · · · · · · · · · · ·			
Library Manager LC_PRG (PRG)	Files	Gateway		•	
POU_CtrlA (PRG)	Log	Gateway-1	MyDevice (active)		
Task Configuration	PLC Settings	IP-Address:	Device Name:		
■- 🥩 MainTask 	PLC Shell	Port:	MyDevice Device Address:		
POU_CtrlA	Users and Groups	1217	0301.5002		
POU_FB	Access Rights		Target ID: 17E5 0001		
PersistentVars	Symbol Rights		Target Type: 4096		
Visualization Manager	Licensed Software Metrics		Target Vendor: TOKYO KEIKI INC.		
Cx2500Ain (Cx2500Ain)	Task Deployment		Target Version: 3.5.18.30		
Cx2500Din (Cx2500Din)	Status				
Cx2500Frqin (Cx2500Frqin)	Information				
Cx2500Pwm (Cx2500Pwm)					>

Figure 118 デバイスウィンドウ CAN デバイス追加



8.3. <u>J1939</u>

8.3.1. <u>CANbus</u>

バスマスタ CANbus エディタ画面について説明します。バスマスタのエディタ画面は、エディタウィンドウ にあるバスマスタをダブルクリックすると表示されます。

Device CANbus X	줴 J1939_Manager - 궤 J1939_CX2500 - 궤 J1939_ECU = CAN_Ctrl	•
General	CANbus : n/a	
Log	Last diagnostic message	Acknowledge
CANbus IEC Objects	Diagnosis Info	
	Current Network ID	
Status	Current Baudrate	
	- Bus state	
Information	Bus Alarm	
	Bus Error counter	
	Last Driver Open Error	
	Driver Instances	
	Bus Load	
	Tx Counter	
	Tx Error Counter	
	- Rx Counter	
	Rx Error Counter	
	Lost Counter	
	<	>
<		>

Figure 119 CANbus エディタ画面

Table 56 CANbus タブ一覧

タブ名称	摘要
General	CAN バスに関する設定をおこなうことができる。
Log	デバッグ中にバスマスタ CANbus の動作ログを確認できる。
CANbus IEC Objects	このタブ内で定義されているオブジェクト名称を使うことでユーザーアプリケーションから
	CANbus の一部情報にアクセスできる。
Status	デバッグ中にバスステータス情報を確認できる。
Information	バスマスタ CANbus のバージョン情報などを確認できる。



8.3.1.1. <u>General タブ</u>

バスマスタ CANbus の General タブでは、使用する CX2500 のチャネルとボードレートを設定して下さい。

Device CANbus X	J1939_Manager	J1939_CX2500	J1939_ECU		•
General	General				_
Log	Network	0		CAN	
CANbus IEC Objects	Baud rate (kbit/s)	250	\checkmark		
Status					
Information					
(
					7

Figure 120 CANbus General タブ

Table 57	CANbus	General タブ	設定項目
	0/ 11003		ᇝᇨᆽᆸ

設定項目	摘要
Network	フィールドバスで使用する CX2500 の CAN チャネル番号(0~4)を入力する。
Baud rate(kbit/s)	選択タブからアプリケーションで使用するボードレートを選択する。
	なお、設定可能なボードレートは Table 47 と同じとなる。



8.3.1.2. <u>Status タブ</u>

このタブでは、デバッグ中、下図のように CAN バスステータス情報が表示されます。



Figure 121 CANbus Status タブ(デバッグ中の表示例)



8.3.2. <u>J1939_Manager</u>

プロトコルマネージャーJ1939_Manager エディタ画面について説明します。プロトコルマネージャーのエデ ィタ画面は、エディタウィンドウにあるプロトコルマネージャーをダブルクリックすると表示されます。

CANbus	📊 J1939_Manager 🗙 📊 J1939_CX2500 🛛 🕅 J1939_ECU 📄 CAN_Ctrl 🔍 🔻
General	Databases
Log	Install Uninstall Set as default J1939Default
J1939 I/O Mapping	
J1939 IEC Objects	
Status	
Information	
<	>

Figure 122 J1939_Manager エディタ画面

タブ名称	摘要
General	DBC ファイルの登録/削除をおこなうことができる。ファイルを登録することで J1939 フィ
	ールドバスでの各 CAN デバイスで、送受信メッセージを簡単に設定できる。
Log	デバッグ中にプロトコルマネージャーJ1939_Managerの動作ログを確認できる。
J1939 I/O Mapping	フィールドバスを制御する周期時間となるバスサイクルを設定できる。
J1939 IEC Objects	このタブ内で定義されているオブジェクト名称を使うことでユーザーアプリケーションから
	J1939_Manager の一部情報にアクセスできる。
Status	デバッグ中に J1939_Manager の動作良否を確認できる。
Information	プロトコルマネージャーJ1939_Managerのバージョン情報などを確認できる。

Table 58 J1939_Manager タブ一覧



8.3.2.1. <u>General タブ</u>

このタブでは、DBC ファイルの登録/削除をおこなうことができます。DBC ファイルを登録することで、J1939 フィールドバスでの各 CAN デバイスで、送受信メッセージを簡単に設定することができます。登録の際はタブ 画面上の「Install」ボタンを押してファイルをインストール(登録)できます。登録した DBC ファイルを削除す る際は、同画面上の削除対象ファイルを選択した上で「Uninstall」ボタンを押すことで削除できます。

Device CANbus	M J1939_Manager X M J1939_CX2500 M J1939_ECU CAN_Ctrl
General	Databases
Log	Install Uninstall Set as default J1939Default
J1939 I/O Mapping	
J1939 IEC Objects	
Status	
Information	
< Comparison of the second sec	· · · · · · · · · · · · · · · · · · ·

Figure 123 J1939_Manager General タブ



CCOT-24-016 Rev.1 8.3.2.2. <u>J1939 I/O Mapping タブ</u>

このタブでは、フィールドバス内の送受信メッセージやステータス情報を制御する周期時間を設定して下さい。

Device CANbus	J1939_Manager	X 11939_CX2500	J1939_ECU		-
General	Bus Cycle Options Bus cycle task	MainTask	✓ Recreate	required tasks	
Log					
J1939 I/O Mapping					
J1939 IEC Objects					
Status					
Information					
<					>

Figure 124 J1939_Manager J1939 I/O Mapping タブ



CCOT-24-016 Rev.1 8.3.3. <u>CAN デバイス</u>

J1939の CAN デバイスエディタ画面について説明します。CAN デバイスのエディタ画面は、エディタウィンドウにある CAN デバイスをダブルクリックすると表示されます。

CANbus	J1939_Manager	1939_CX2500 J1939_ECU 🗙	CAN_Ctrl	-
General	General			
TX Signals	Preferred address 0	-	SAE J1939	
Log	Local device			
J1939 IEC Objects	A ECU NAME			
Statue	NAME (64 bit): 16# 0			
Status	Arbitrary Address Capable			
Information	Industry Group	0: Global, applies to all industries	~	
	Vehicle System Instance	0		
	Vehicle System	0		
	Reserved			
	Function	0		
	Function Instance	0		
	ECU Instance	0		
	Manufacturer Code	0		
	Identity Number	0		
	4 Communication Watchd			
	Enable Communication W	atchdog		
	Watchdog Time (in ms):	1500		
<				>

Figure 125 CAN デバイス(J1939) エディタ画面

タブ名称	摘要
General	J1939 プロトコルにおけるパラメータ(CAN-ID など)を設定できる。
TX Signals	CAN デバイスに関わる送受信メッセージの登録および送信設定ができる。
P2P RX Signals	General で Local device にチェックを入れると表示される。他の ECU が受信する必要が
	ある全ての PGN が表示される。
Log	デバッグ中に CAN デバイスの動作ログを確認できる。
J1939 I/O Mapping	上記 Signals タブで定義したメッセージのデータに対して、ユーザーアプリケーションで使
	う変数を紐づけることができる。メッセージが登録されていないとこのタブは表示されな
	ιν _°
J1939 IEC Objects	このタブ内で定義されているオブジェクト名称を使うことでユーザーアプリケーションから
	CAN デバイスの一部情報にアクセスできる。
Status	デバッグ中、J1939 プロトコルにおける診断ステータス情報(DM1 - Diagnostic Message1)
	を確認できる。
Information	CAN デバイスのバージョン情報などを確認できる。

Table 59 J1939 CAN デバイス タブ一覧



8.3.3.1. <u>General タブ</u>

このタブは、J1939 プロトコルにおけるパラメータを設定できます。各種ユーザーアプリケーション所望の値になるように設定して下さい。

Device CANbus	🕂 J1939_Manager 🕂	J1939_CX2500		
General	General			
TX Signals	Preferred address 0	▲ ▼	SAE J1939	
Log	Local device			
J1939 IEC Objects	ECU NAME			
Status	NAME (64 bit): 16# 0			
	Arbitrary Address Capable			
Information	Industry Group	0: Global, applies to all industries	\checkmark	
	Vehicle System Instance	0		
	Vehicle System	0		
	Reserved			
	Function	0 🗘		
	Function Instance	0 🗘		
	ECU Instance	0		
	Manufacturer Code	0		
	Identity Number	0		
	Communication Watchd	og		
	Enable Communication W	Vatchdog		
	Watchdog Time (in ms):	1500		
5				>

Figure 126 CAN デバイス(J1939) General タブ



区分	名称	摘要
General	Preferred address	CAN デバイスのソースアドレスを入力する。
	Local device	チェックの有無によって、TxSignals タブで登録できるメッセージが
		変わる。
		☑ :CX2500 が他 ECU(デバイス)へ送信するメッセージを登録できる。
		□ :CX2500 が他 ECU から受信するメッセージを登録できる。
ECU NAME ^{%17}	NAME	ECU NAME を入力する。もしくは、以降のパラメータを入力することにより
		自動で数値が設定される。
	Arbitrary Address	チェックを入れると、他 ECU でアドレスが競合した場合に、CAN デバイス
	Capable	を別のアドレスに変更することを試みる。
	Industry Group	ECU NAME に関する設定。所望の値を入力する。
	Vehicle System	
	Instance	
	Vehicle System	
	Reserved	
	Function	
	Function Instance	
	ECU Instance	
	Manufacturer Code	
	Identity Number	
Communication	Enable Communication	チェックを入れると、CAN デバイスに関するメッセージが Watchdog Time
Watchdog ^{※18}	Watchdog	の時間内に送受信されたかを監視する。時間内に送受信されていない場
		合は、デバイスウィンドウ上の CAN デバイスのアイコン表示が△になる。
	Watchdog Time	送受信可否監視時間[ms]を入力する。

※17 ECU NAME については、J1939 プロトコル規格書などを参照。

※18 Local Device が有効の場合は表示されない。



8.3.3.2. <u>TX Signals タブ</u>

TX Signals タブでは、送受信メッセージの登録・削除をおこなうことができます。ただし、General タブの設 定項目 Local Device の設定により登録できるメッセージの種類が異なります。

Table 61 TX Signals タブの役割

Local Device 設定	TX Signal タブ役割
	CX2500 が他の ECU(デバイス)へ送信するメッセージを登録できる。
	CX2500 が他の ECU(デバイス)から受信するメッセージを登録できる。

J1939_Manager CANbu	us CAN_Ctrl] J1939_ECU	Device J1939_CX2500 X	
General	Enable PGN/SPN	Name Length	Туре	
TX Signals				
P2P RX Signals				
Log				
J1939 IEC Objects				
Status				
Information				
	Add PG	ional Delete		

Figure 127 CAN デバイス(J1939) TX Signals タブ



【メッセージの登録】

ここでは、メッセージ登録の手順を示します。メッセージの登録は送受信に限らず同様です。

(1) TX Signals タブ画面にて、「Add PG」ボタンを押して下さい。

J1939_Manager CANb	us 📄 CA	N_Ctrl	J1939_ECU	Device	J1939_CX2500	×	
General	Enable PC	N/SPN Name	e Length	Туре			
TX Signals							
P2P RX Signals							
Log							
J1939 IEC Objects							
Status							
Information							
	Add PG	Add Signal	Delete				
			2,01000				

Figure 128 メッセージの登録 Add PG ボタン



(2)「Add Parameter Group」ウィンドウが表示されます。このウィンドウにはタブが2つあります。事前にプロトコルマネージャーでDBCファイルを登録していれば、「Datebase」タブにDBCファイルに登録されている CANメッセージの一覧が表示されます。所望のメッセージを選択して「Add PG」ボタンを押して下さい。もし、DBCファイルが無かったり、DBCファイル内に登録されていないメッセージを登録したい場合は「Custom」タブで所望の値を入力することでメッセージを登録することができます。

abase TEST	r_11939	Filter	
d a parameter	group	│ All	
GN/SPN	Name	Description	
61444	ID_EEC1		٦
61450	ID_EGF1		
64656	ID_EFIMRPI2		
64712	ID_EEC13		
64777	ID_HRLFC		
64789	ID_BCH1		
64891	ID_AT151		
64892	ID_DPFC1		
64946	ID_AT1IMG		
65190	ID_IMT1		
65219	ID_ETC5		
65226	ID_DM1_SINGLE		
65247	ID_EEC3		
CEDE7	ID_LFC		

Figure 129 Add Parameter Group ウィンドウ Database タブ



.dd Parameter Group	_		\times
Database Custom			
Database Custom PGN 61440 PDU Specific (Bit 0-7) 0 PDU Format (Bit 8-15) 240 Datapage (Bit 16) 0 Datapage (Bit 17) 0 Seneral			
Ad	d PG	Clo	se

Figure 130 Add Parameter Group ウィンドウ Custom タブ



(3) TX Signals にメッセージが登録されます。登録することで、CAN デバイスの I/O Mapping タブで作業がお こなうことができたり、送信メッセージの場合は送信方法設定ができるようになります。また、登録したメ ッセージを選択して「Add Signal」ボタンを押すことで、メッセージデータの信号ラベルを登録することが できます。信号ラベルを登録すると、メッセージデータ内の指定領域において、ユーザーアプリケーション でそのデータを設定・取得することができるようになります。(DBC ファイルであらかじめ登録していれば、 メッセージ登録した段階で信号ラベルが既に登録されています)

	Enable	PGN/SPN	Name	Length	Type	~	General	
		61444	TD FFC1	8 Bytes	Broadcast		PGN	61444
Signals		0	EG PONT TRO	8 Bits	Divudcust		Name	ID_EEC
No. 144		0	EG_SPEED	16 Bite			Description	-
.og		U	LG_SPEED	10 0103			Length (Bytes)	8
1939 IEC Objects								
atus								
ormation								
						N	ame	
						NN	ame ame of Parameter G	iroup

Figure 131 TX Signals タブ メッセージ登録完了後



【送信方法設定】

デバイスに送信メッセージを登録できる設定の場合、各メッセージの送信方法の設定もおこなうことができま す。所望のメッセージを選択して表示されるメニュー内の値を編集することで設定できます。

ieneral	Enal	ble	PGN/SPN	Name	Length	Type	~	General	
	-		65220	TO INCHATOL	RButor	Broadcact		PGN	65329
X Signals		V	05529		o bytes	broaucast		Name	ID_INOMAT01
			0	EG_CTRL_MODE	2 Bits			Description	
2P RX Signals			0	OVR_PRI	2 Bits			Length (Bytes)	8
	-		0	P_GAIN	8 Bits		~	Transmission Settings	
og			0	I_GAIN	8 Bits			Priority 07	6
			0	OILPUMP_PRS	8 Bits			Destination Address	255
1939 I/O Mapping			0	REQ_RPM	16 Bits			TransmissionMode	Cyclic
			0	CTRL TRG	2 Bits			Cycle Time Factor	0
1939 IEC Objects			0	MSG CNT	4 Bits				
			0	MSG CHKSUM	4 Bite				
tatus			57244	TD CM1	9 Puter	838			
			57344	ID_CHI	8 bytes	FZF			
formation		~	65219	ID_EICS	8 Bytes	Broadcast			
	-								
							Cyc	cle Time Factor	
							Onl	ly valid for cylic Transmission Mode	; determines the number
							PLC	C cycles a Parameter Group should	be sent.

Figure 132 TX Signals タブ 送信方法設定メニュー

Table 62	送信方法設定メニュー	-
----------	------------	---

区分	名称	摘要	
General	PGN ^{%19}	パラメータグループ番号	
	Name ^{%19}	パラメータグループの名	称
	Description ^{%19}	パラメータグループの説	明
	Length(Bytes) ^{%19}	パラメータグループのメ	ッセージデータの長
Transmission	Priority ^{^{%19}}	送信優先度	
Settings	Destination Address ^{%19}	ターゲットアドレス(P2P >	メッセージのみ有効)
	Transmission Mode	送信方法を選択タブから	選択する。
		項目	摘要
		Change Of State	メッセージ内のデータがユーザーアプリケーショ
			ンで変更された時送信する。
		Cyclic ^{%20}	周期的に送信する。
		On Request	Request PGN メッセージを受信した時に送信す
			る。
		Application Triggered	非対応
	Cycle Time Factor ^{※20}	Transmission Mode を C	yclic にした場合に有効になる。メッセージの送信周
		期を設定する。	
		送信周期時間:「J1939 E	3us Cycle Task[ms]」 × 「Cycle Time Factor」

※19 これらの設定項目については、メッセージ登録した時点で既に設定されているため、再編集は不要となる。



※20 フィールドバスに限らず、他のタスクや CX2500 内部処理のため、Cyclic で設定した周期から定常的に遅れて送信されるこ とが想定されます。その際は、タスクを短周期にしたり、Cycle Time Factor を微調整して対応をお願い致します。

8.3.3.3. P2P RX Signals タブ

このタブは、General で Local device にチェックを入れると表示されます。このデバイスの他にも別の ECU(デバイス)が受信する必要がある PGN がある場合、このタブで受信メッセージを登録することができます。メッセージの登録手順は TX Signals タブと同様です。

J1939_Manager CANt	bus 📄 CAN_Ctrl 🕅 J1939_ECU 🗙 🕅 Device 🖷 J1939_CX2500	
General	Enable PGN/SPN Name Length Type	
TX Signals		
P2P RX Signals		
Log		
J1939 IEC Objects		
Status		
Information		
	Adu Po Adu bignai Delete	

Figure 133 CAN デバイス(J1939) P2P RX Signals タブ



8.3.3.4. <u>J1939 I/O Mapping タブ</u>

このタブは、Signals タブでメッセージを登録した場合に表示されます。各メッセージのデータが一覧表示されており、IOドライバと同様にこれらに変数を紐づけてユーザーアプリケーションで使用することができます。

Seneral	Find		Filter Show all			• - A	dd FB fo	r IO Channel →	Go to Instance
IX Signals	Variable	Mapping	Channel	Address	Туре	Default Value	Unit	Description	
.og	B_CCCT	***	EG_PCNT_TRQ	%IW80	INT		%		
1929 I/O Mapping	₩ EG_SPEED	**	EG_SPEED	%ID41	REAL		min-1		
1939 I/O Happing									
1939 IEC Objects									
Status									
information									

Figure 134 CAN デバイス(J1939) J1939 I/O Mapping タブ

8.3.3.5. <u>Status タブ</u>

このタブは J1939 プロトコルにおける診断情報を表示します。CX2500 が診断メッセージ(DM1 – Diagnostic Message1)を受信した場合、メッセージ内容に従って下図のように診断情報が表示されます。診断メッセージデータの詳細については、J1939 プロトコルの規格書を参照して下さい。

	Diagnostic Lamps (DM1)	
Signals		
3		
39 I/O Mapping	Malfunction Stop Warning Protect	
939 IEC Objects		
atus	J1939 : A diagnostic message is pending. The error has been deared.	
ormation	Last diagnostic message	Acknowledge
	Active DTC SPN 110 FMI Current below normal or open circuit OC 1 Conversion Method 1	

Figure 135 CAN デバイス(J1939) Status タブ(デバッグ中の表示例)



8.4. CANopen

8.4.1. CANbus

CANopen をフィールドバスで使う際のバスマスタ CANbus の設定・モニタ等は J1939 を使う場合と同様で す。8.3.1 項を参照して下さい。

8.4.2. CANopen_Manager

プロトコルマネージャーCANopen_Manager エディタ画面について説明します。プロトコルマネージャーの エディタ画面は、エディタウィンドウにあるプロトコルマネージャーをダブルクリックすると表示されます。

プロトコルマネージャーはリモートデバイス(CANopen マスタ)として CX2500 を使用する場合に紐づけが必要になります。ローカルデバイス(CANopen スレーブ)として CX2500 を使用する場合は CANopen Manager は不要です。その際は、バスマスタ CANbus から直接 CAN デバイスを紐づけて下さい。

CANbus 📊 J1939_Ma	nager 🏦 ECMCAN 🎢 Device 🏦 CTRL_IO_MAIN 🎢 11939_ECU 🦷 CANopenDeviceExample 🦯 🖓 CANopen_Manager 🗙 🔻
General	General
Log	Node-ID 127 🗧 Check and Fix Configuration
CANopen I/O Mapping	Autostart CANopen Manager 🛛 Polling of optional slaves
CANopen IEC Objects	Start slaves NMT error behavior Restart Slave V
Status	NMTstart all (if possible)
Information	☐ Guardung
	Node-ID 127
	Producer time (ms) 200
	# SYNC # TIME
	Enable SYNC producing Enable TIME producing
	COB-ID (Hex) 16# 80 + COB-ID (Hex) 16# 100 +
	Cycle period (µs) 1000 💠 Producer time (ms) 1000 💠
	Window length (µs) 1200 🗘
	Enable SYNC consuming

Figure 136 CANopen Manager エディタ画面

Table 63	CANopen_Manager	タブー	-覧
----------	-----------------	-----	----

タブ名称	摘要
General	CANopen プロトコルに関するパラメータの設定ができる。
Log	デバッグ中にプロトコルマネージャーCANopen_Managerの動作ログを確認できる。
CANopen I/O Mapping	フィールドバスを制御する周期時間となるバスサイクルを設定できる。
CANopen IEC Objects	このタブ内で定義されているオブジェクト名称を使うことでユーザーアプリケーションから
	CANopen_Manager の一部情報にアクセスできる。
Status	デバッグ中に CANopen_Manager の動作良否を確認できる。
Information	プロトコルマネージャーCANopen_Managerのバージョン情報などを確認できる。



8.4.2.1. <u>General タブ</u>

このタブは、CANopen プロトコルにおけるパラメータを設定できます。各種ユーザーアプリケーション所望 の値になるように設定して下さい。

GANbus 🔐 J1939_Ma	nager 👔 ECMCAN 📊 Device 👔 CTRL_IO_MAIN 👘 J1939_ECU 🕅 CANopenDeviceExample 🦯 🕅 CANopen_Manager 🗙
General	General
Log	Node-ID 127 Check and Fix Configuration
CANopen I/O Mapping	🗹 Autostart CANopen Manager 🛛 Polling of optional slaves
CANopen IEC Objects	Start slaves NMT error behavior Restart Slave
Status	MT start all (if possible) Guarding
Information	Enable heartbeat producing
	Node-ID 127
	Producer time (ms) 200
	# SYNC
	Enable SYNC producing Enable TIME producing
	COB-ID (Hex) 16# 80 + COB-ID (Hex) 16# 100 +
	Cycle period (µs) 1000 🗘 Producer time (ms) 1000 🗘
	Window length (µs)
	Enable SYNC consuming

Figure 137 CANopen Manager General タブ

Κεικι

CCOT-24-016 Rev.1

Table 64 CANopen_Manager General タブ設定項目

区分	名称	摘要				
General	Node-ID	CANopen マスタデバイスとなる CX2500 のノード番号を入力する。				
	Autostart CANopen	チェックを入れること。チェックを入れることでプロトコルマネージャーが自動				
	Manager	で起動する。				
	Polling of optional slaves	チェックを入れると、スレーブデバイスが応答しない場合に 1 秒ごとにマスタ				
		からスレーブへ問い合わせる。				
	Start slaves	チェックを入れること。チェックを入れると、プロトコルマネージャーが紐づけ				
		たスレーブデバイスの起動設定処理を自動でおこなう。				
	NMT error behavior	NMT(Network Management Object)エラー時のスレーブデバイスの動作を選				
		択タブから選択する。				
		Restart slave :NMT エラー時スレーブデバイスを再起動する。				
		Stop slave : NMT エラー時スレーブデバイスを停止する。				
	NMT start all(if possible)	チェックを入れると、プロトコルマネージャーが NMT Start All コマンドを使用				
		して全てのスレーブデバイスを開始する。				
Guarding Enable heartbeat		チェックを入れると、マスタデバイスが Producer time の間隔でハートビートを				
	producing	送信する。				
	Node-ID	ハートビート用のノード ID を入力する。				
	Producer time(ms)	ハートビート送信間隔[ms]を入力する。				
SYNC	Enable SYNC producing	チェックを入れると、マスタデバイスが Cycle period の間隔で SYNC メッセー				
		ジを送信する。				
	COB-ID(Hex)	SYNC メッセージの ID を入力する。				
	Cycle period(µs)	SYNC メッセージの送信間隔[µs]を入力する。				
	Window length(µs)	同期 PDO のタイムフレーム長[µs]を入力する。				
	Enable SYNC consuming	チェックを入れると、スレーブデバイスのいずれかが SYNC 生成をおこなう必				
		要がある。				
TIME	Enable TIME producing	チェックを入れると、プロトコルマネージャーは TIME メッセージを送信する。				
	COB-ID(Hex)	TIME メッセージの ID を入力する。				
	Producer time(ms)	TIME メッセージ送信間隔[ms]を入力する。				



8.4.2.2. <u>CANpen I/O Mapping タブ</u>

このタブでは、フィールドバス内の送受信メッセージやステータス情報を制御する周期時間を設定して下さい。

CANbus 11939_Ma	nager 📑 ECM	CAN Device		IO_MAIN J1939_ECU	CANopenDeviceExample	CANopen_Manager 🗙 🔻
General	Bus Cycle Options					
Log	Bus cycle task	MainTask	~	Recreate required tasks		
CANopen I/O Mapping						
CANopen IEC Objects						
Status						
Information						

Figure 138 CANopen Manager CANopen I/O Mapping タブ



8.4.3. CAN デバイス

CANopen の CAN デバイスエディタ画面について説明します。CAN デバイスのエディタ画面は、エディタウ ィンドウにある CAN デバイスをダブルクリックすると表示されます。

CANopen の CAN デバイスは、EDS ファイルのデータやデバイスによって下記タブ名などが異なる場合があることに留意して下さい。

neral	Receive PDOs (Master => Slave)			Transmit PDOs (Slave => Master)		
	+ Add PDO + Add Mapping 🖋 Edit 🗙 Delete 🛧 Move Up 🔋 + Add PDO + Add Mapping 🖋 Edit 🗙 Delete 🛧 Move Up					
Os	Name	Object	Bit lend	Name	Object	Bit ler
0s	✓ 16#1400: RPDO communicatio	16#201 (\$NODEID+16	64	✓ 16#1800: TPDO communication	16#181 (\$NODEID+16	64
-	DigOutput8	16#6200:16#01	8	DigInput8	16#6000:16#01	8
1	DigOutput8	16#6200:16#02	8	DigInput8	16#6000:16#02	8
	DigOutput8	16#6200:16#03	8	DigInput8	16#6000:16#03	8
Nopen I/O Mapping	DigOutput8	16#6200:16#04	8	DigInput8	16#6000:16#04	8
	DigOutput8	16#6200:16#05	8	DigInput8	16#6000:16#05	8
Nopen IEC Objects	DigOutput8	16#6200:16#06	8	DigInput8	16#6000:16#06	8
	DigOutput8	16#6200:16#07	8	DigInput8	16#6000:16#07	8
tus	DigOutput8	16#6200:16#08	8	DigInput8	16#6000:16#08	8
rmation						
	<		>	<		

Figure 139 CAN デバイス(CANopen) エディタ画面

タブ名称	摘要
General	CANopen プロトコルにおけるデバイスパラメータを設定できる。
PDOs	各デバイスの PDO 設定ができる。
SDOs	CX2500 がマスタデバイスの場合、CANopen_Manager に紐づけたスレーブデバイスは
又は	SDOs タブとなり、SDO 設定をおこなうことができる。
Object Dictionary	CX2500 がスレーブデバイスの場合、Object Dictionary タブとなり、スレーブデバイス
	(CX2500)の Object Dictionary 設定をおこなうことができる。
Log	デバッグ中に CAN デバイスの動作ログを確認できる。
CANopen I/O Mapping	上記 Object タブで定義したメッセージのデータに対して、ユーザーアプリケーションで使
	う変数を紐づけることができる。メッセージが登録されていないとこのタブは表示されな
	い。
CANopen IEC Objects	このタブ内で定義されているオブジェクト名称を使うことでユーザーアプリケーションから
	CAN デバイスの一部情報にアクセスできる。
Status	CANopen プロトコルにおけるステータス情報を表示する。
Information	CAN デバイスのバージョン情報などを確認できる。

Table 65 CANopen CAN デバイス タブ一覧



8.4.3.1. <u>General タブ</u>

このタブは、CANopen プロトコルにおけるデバイスパラメータを設定できます。このタブは、CX2500 が CANopen マスタかスレーブかによって設定画面及び項目が異なります。CODESYS にデフォルトでインストー ル(登録)されているスレーブデバイスとは別に、CANopen デバイス(EDS ファイル)を新しく CODESYS-IDE に 追加する場合は、別紙「CX2500Codesys_UserManual_ForSetup」4.4 節の要領で EDS ファイルのインストー ル(登録)をおこなってください。

selleral	General	
PDOs	Node-ID 1	CANopen
DOs	Enable expert settings	
og	Enable SYNC producing	
CANopen I/O Mapping	✓ Guarding	
CANopen IEC Objects	Guard time (ms) 0	Producer time (ms) 200
Status	Life time factor 0	✓ Heartbeat consuming (1/1 active)
nformation	Emergency (EMCY)	⊿ TIME
	Enable emergency (EMCY)	Enable TIME producing
	COB-ID \$NODEID+16#80	COB-ID (Hex) 16# 100
		Enable TIME consuming
	✓ Checks at Startup	
	Check vendor ID Check produc	t number 🔲 Check revision number

Figure 140 CAN デバイス(CANopen マスタ) General タブ



Table 66 (CAN デバイス	CANopen マスタ	設定項目(デフォルト時)
------------	----------	-------------	--------------

区分	名称	摘要
General	Node-ID	スレーブデバイスのノード ID を入力する。
	Enable expert settings	チェックを入れると、対象デバイスの EDS ファイル内に定義されている全設定
		が表示される。
	Enable SYNC producing	CANopen_Manager の Enable SYNC progucing が無効化されている時、チェック
		を入れると対象のスレーブデバイスで SYNC 生成をが有効になる。
Guarding	Enable nodeguarding	チェックを入れると、スレーブデバイスが応答しない場合にマスタデバイスはノ
		ードガードメッセージを送信する。
	Guard time(ms)	ノードガードメッセージの送信周期[ms]を入力する。
	Life time factor	ー定時間(Guard time×Life time factor)スレーブデバイスが応答しない場合、
		デバイスはノードガードエラーを検知する。
	Enable heartbeat	チェックを入れると、デバイスは Producer time で設定した周期でハートビートメ
	producing	ッセージを送信する。
	Producer time(ms)	ハートビートメッセージの送信周期[ms]を入力する。
	Heartbeat consuming	ボタンを押すとハートビート監視するスレーブデバイスを設定できる。
EMCY	Enable Emergency	チェックを入れると、デバイスで内部エラーが発生した場合に EMCY メッセージ
		を送信する。
	COB-ID	EMCY メッセージの ID を入力する。
TIME	Enable TIME producing	チェックを入れると、デバイスは TIME メッセージを送信する。
	COB-ID(Hex)	TIME メッセージの ID を入力する。
	Enable TIME consuming	チェックを入れると、デバイスは TIME メッセージの処理をおこなう。
Checks at	Check Vendor ID	デバイス起動時にベンダーIDをEDSファイルの設定と合っているか確認する。
Startup	Check product number	デバイス起動時に製品番号が EDS ファイルの設定と合っているか確認する。
	Check revision number	デバイス起動時にリビジョン番号が EDS ファイルの設定と合っているか確認す
		る。



General	General			
Object Dictionary	Node ID 1	•	CANODED	
PDOs	Device profile 401	\$		
Log	Edit I/O Area	Edit SDO Parame	ter Area	
CANopen I/O Mapping	EDS File			
CANopen IEC Objects	Vendor name 3S -	Smart Software Solutions GmbH		
Status	Vendor number 801	÷		
Information	Product name CAN	lopenDeviceExample		
	Product number 30	÷		
	Revision number 4	÷		
	Install to Device Rep	pository Export EDS F	ile	

Figure 141 CAN デバイス(CANopen スレーブ) General タブ

Table 67	CAN デバイス	CANopen スレーブ	設定項目
----------	----------	--------------	------

区分	名称	摘要
General	Node ID	スレーブデバイスのノード ID を入力する。
	Device profile	デバイスのプロファイル番号を入力する。
	Edit I/O Area	ボタンを押すと、IO 領域の編集をおこなうことができる。編集ウィンドウにつ
		いては CODESYS オンラインヘルプを参照。
	Edit SDO Parameter Area	ボタンを押すと、SDO パラメータの編集をおこなうことができる。編集ウィン
		ドウについては CODESYS オンラインヘルプを参照。
EDS File	Vendor name	ベンダー名
	Vendor number	ベンダー番号
	Product name	製品名
	Product number	製品番号
	Revision number	リビジョン番号
	Install to Device	ボタンを押すと、EDS ファイルをインポートし、CAN デバイスに EDS ファイル
	Repository	の設定を反映することができる。
	Export EDS File	ボタンを押すと、エディタ画面で設定した情報を元に EDS ファイルを生成・
		エクスポートする。



CCOT-24-016 Rev.1 8.4.3.2. PDOs タブ

このタブでは、CAN デバイスの PDO 設定をおこなうことができます。

CX2500 がマスタデバイスの時、CANopen_Manager に紐づけたスレーブデバイスの PDOs タブは以下の通 りになります。PDOs タブには、EDS ファイルから読み取られた PDO メッセージが自動追加されて一覧表示さ れます。この場合の PDS タブでは PDO メッセージの ID や送受信の種類について編集をおこなうことができま す。編集は所望のメッセージを選択した上で「Edit」ボタンを押して下さい。すると、PDO Properties ウィンド ウが表示されるので、所望の値を設定し「OK」ボタンを押して下さい。



Figure 142 CAN デバイス(CANopen) リモートデバイス PDOs タブ

COB-ID	\$NODEID+	16#200		
	= 16#201 (513)		
Inhibit time (x 100µs)	0	*		
Fransmissiontype	Asynchronous - device-profile-specific (Type 255 ${\sim}$			
Number of syncs	1	*		
Event time (x 1ms)	0	*		

Figure 143 リモートデバイス PDO Properties ウィンドウ



Table 68	リモートデバイス	PDO Properties ウィンドウ	設定項目
----------	----------	----------------------	------

項目	摘要			
COB-ID	対象の PDO メッセー	−ジの ID を設定する。		
RTR	送信 PDO メッセー	ジのみ表示される。チェックを入れると、リモートフレームを使用		
	して PDO 問い合わ ⁻	せをおこなう。		
Inhibit time($\times 100 \mu$ s)	非対応			
Transmissiontype	PDO の送信方法を選択タブから設定する。			
	Transmissiontype	PDO メッセージの送信法		
	Acyclic-	同期送信		
	synchronous			
	Cyclic-	バスサイクル×Number if syncs の周期で送信		
	synchronous			
	Synchronous-	送信 PDO メッセージのみ。リモート送信要求時のみ送信(同		
	only RTR	期)。		
	Asynchronous-	送信 PDO メッセージのみ。リモート送信要求時のみ送信(非		
	only RTR	同期)。		
	Asynchronous-	非対応		
	manufacturer			
	specific			
	Asynchronous-	CiA デバイスプロファイルに従って送信。		
	device profile			
Number if syncs	Transmissiontype が	「Cyclic-synchronous 設定時のみ有効化される。PDO の送信間		
	隔(バスサイクル時	間×Number if syncs)を入力する。		
Event time(× 1ms)	非対応			
Process by CANopen Manager	チェックを入れること。チェックを外すと、CANopen_Manager(マスタデバイス)が			
	PDO の処理をおこなわないため、送受信もおこなわれない。			



CX2500 がローカルデバイス(CANopen スレーブ)の時の PDOs タブは以下の通りとなります。この場合の PDOs タブでは主に Table 69 の操作ができます。

General	Receive PDOs (Master => Slave)			Transmit PDOs (Slave => Master)		
	🕂 Add PDO 🕂 Add Mapping 💉 Edi	it 💢 Delete 🔺 Move Up	Ŧ	🕂 Add PDO 🕂 Add Mapping 💉 Edit	🗙 Delete 🔺 Move Up	
bject Dictionary	Name	Object	Bitlength	Name	Object	Bitlengt
DOs	✓ 16#1400: RPDO communicati	16#201 (\$NODEID+1	64	✓ 16#1800: TPDO communicatio	16#181 (\$NODEID+16	64
	DigOutput8	16#6200:16#01	8	DigInput8	16#6000:16#01	8
og	DigOutput8	16#6200:16#02	8	DigInput8	16#6000:16#02	8
	DigOutput8	16#6200:16#03	8	DigInput8	16#6000:16#03	8
ANopen I/O Mapping	DigOutput8	16#6200:16#04	8	DigInput8	16#6000:16#04	8
	DigOutput8	16#6200:16#05	8	DigInput8	16#6000:16#05	8
ANopen IEC Objects	DigOutput8	16#6200:16#06	8	DigInput8	16#6000:16#06	8
	DigOutput8	16#6200:16#07	8	DigInput8	16#6000:16#07	8
tatus	DigOutput8	16#6200:16#08	8	DigInput8	16#6000:16#08	8
nformation						

Figure 144 CAN デバイス(CANopen) ローカルデバイス PDOs タブ

操作名	対応するボタン名称	編集ウィンドウ		
PDO メッセージの追加	Add PDO	Figure 145		
PDO メッセージ編集	Edit	Figure 146		
PDO メッセージデータの追加	Add Mapping	Figure 147		
PDO メッセージデータの編集	Edit	Figure 147		

Table 69 ローカルデバイス PDOs タブ 主な操作



 \times

Select PDO

Name	Object	Bit Length			^
16#1401: RPDO communication parameter					
I6#1402: RPDO communication parameter					
16#1403: RPDO communication parameter					
16#1404: RPDO communication parameter					
16#1405: RPDO communication parameter					
16#1406: RPDO communication parameter					
16#1407: RPDO communication parameter					
16#1408: RPDO communication parameter					
16#1409: RPDO communication parameter					
16#140A: RPDO communication parameter					
16#140B: RPDO communication parameter					
16#140C: RPDO communication parameter					
16#140D: RPDO communication parameter					
16#140E: RPDO communication parameter					
16#140F: RPDO communication parameter					
16#1410: RPDO communication parameter					
16#1411: RPDO communication parameter					~
			0K	Cancel	

Figure 145 Select PDO ウィンドウ(PDO メッセージの追加)


DO Properties		>
COB ID	\$NODEID+16#200 = 16#201 (513)	
Transmissiontype	Asynchronous - device-profile-specific (Type 255) $$ $$ $$ $$	
Number of syncs	1	
Number of syncs	1 (subindex 3, TxPDOs only): Dµs) 0 -	
Number of syncs	1 (subindex 3, TxPDOs only): Dµs) 0 + (subindex 5, TxPDOs only):	

Figure 146 PDO Properties ウィンドウ(PDO メッセージの編集)

項目	摘要	
COB-ID	対象の PDO メッセー	−ジの ID を設定する。
Transmissiontype	PDO の送信方法をi	選択タブから設定する。
	Transmissiontype	PDO メッセージの送信法
	Acyclic-	同期送信
	synchronous	
	Cyclic-	バスサイクル×Number if syncs の周期で送信
	synchronous	
	Synchronous-	送信 PDO メッセージのみ。リモート送信要求時のみ送信(同
	only RTR	期)。
	Asynchronous-	送信 PDO メッセージのみ。リモート送信要求時のみ送信(非
	only RTR	同期)。
	Asynchronous-	非対応
	manufacturer	
	specific	
	Asynchronous-	CiA デバイスプロファイルに従って送信。
	device profile	
Support inhibit time	非対応	
Support event time	非対応	

Table 70 ローカルデバイス PDO Properties ウィンドウ設定項目



dex	Name	Data Type	Default Value	Access Type	
16#1003	Predefined Error Field				
16#1005	Sync COB-ID	UNSIGNED32	16#80	rw	
16#1006	Communication Cycle Period	UNSIGNED32	16#0	rw	
16#100C	Guard Time	UNSIGNED 16	16#0	rw	
16#100D	Life Time Factor	UNSIGNED8	0	rw	
16#1014	COB-ID EMCY	UNSIGNED32	\$NODEID+16#80	rw	
16#1016	Consumer Heartbeat Time				
16#1017	Producer Heartbeat Time	UNSIGNED 16	16#0	rw	
16#6200	Write Output 8 Bit				
16#6200:16#01	DigOutput8	UNSIGNED8	0	rww	
16#6200:16#02	DigOutput8	UNSIGNED8	0	rww	
16#6200:16#03	DigOutput8	UNSIGNED8	0	rww	
16#6200:16#04	DigOutput8	UNSIGNED8	0	rww	
16#6200:16#05	DigOutput8	UNSIGNED8	0	rww	
16#6200:16#06	DigOutput8	UNSIGNED8	0	rww	
16#6200:16#07	DigOutput8	UNSIGNED8	0	rww	
16#6200:16#08	DigOutput8	UNSIGNED8	0	rww	

Figure 147 Select PDO Mapping ウィンドウ(PDO メッセージデータの追加)



CCOT-24-016 Rev.1 8.4.3.3. SDOs/Object Dictionary タブ

CX2500 がマスタデバイスの時、CANopen_Manager に紐づけたスレーブデバイスの SDOs タブは以下の通 りになります。SDOs タブには、EDS ファイルから読み取られた SDO メッセージが自動追加されて一覧表示さ れます。SDOs タブでは、SDO のオブジェクト追加や送信順序(図中の Move Up/Move Down ボタン)を設定す ることができます。なお、オブジェクト追加は「Add SDO」ボタンを押すと、「Select Item from Object Directory」 ウィンドウが開くので、EDS ファイルで登録され表示されているオブジェクトを選択して「OK」ボタンを押す ことで追加されます。

- Ad	d SDO 💉 Edit 🗙 D	elete 🕆 Move U;	p 🔱 Move	Down	
Line	Index:Subindex	Name	Value	Bit Length	Comment
1	16#1005:16#00	Sync COB-ID	16#80	32	

Figure 148 リモートデバイス SDOs タブ

elect Item from (Object Directo	ory					
Index:Subindex	x Nan	ne		AccessType	Туре	Default	
■ 16#1003	Pred	efined Error F	Field				
16#1005:16	#00 Sync	COB-ID		RW	UDINT	16#80	
16#1006:16	5#00 Com	munication C	cle Period	RW	UDINT	16#0	
16#100C:16	5#00 Guar	d Time		RW	UINT	16#0	
16#100D:16	5#00 Life 1	Time Factor		RW	USINT	0	
16#1014:16	5#00 COB	-ID EMCY		RW	UDINT	\$NODEID+16#80	
± 16#1016	Cons	umer Heartb	eat Time				
16#1017:16	#00 Prod	ucer Heartbe	at Time	RW	UINT	16#0	
ⁱ 16#1400	RPDO	O communicat	tion parameter	r			
■ 16#1800	TPDO) communicat	ion parameter				
16#6200	Write	e Output 8 Bit	t				
± 16#6412	Write	e Analogue O	utput 32 Bit				
Name	Unknown Obj	ect					
index	16#0	÷	Bit length	8	÷		
Subindex	16#0	•	Value	0			
						ОК	Cance

Figure 149 Select Item from Object Directory ウィンドウ



CX2500 がローカルデバイス(CANopen スレーブ)の時、CAN デバイスには Object Dictionary タブが表示さ れます。Object Dictionary タブには、EDS ファイルから読み取られたオブジェクトが自動追加されて一覧表示 されます。

Object Dictionary タブでは、新しいオブジェクトの追加・編集をおこなうことができます。新しいオブジェクト追加の際は、「Add Object」ボタンを押すと「Add Object」ウィンドウが開くので、所望のオブジェクトを選択して「Add Object」ボタンを押して下さい。

ieral	Add Object 🕂 Add	Subobject 🗙 Delete 📋 Clone 🖄 li	mport from EDS			
ect Dictionary	Index	Name	Data Type	Default Value	<u>₽</u> 2↓ 🖻	
	16#1000	Device Type	UNSIGNED32	16#191	✓ Access	
s	16#1001	Error Register	UNSIGNED8	16#0	Access Type	rw
	16#1003	Predefined Error Field			PDO Mapping	False
	16#1005	Sync COB-ID	UNSIGNED32	16#80	Refuse Read	False
	16#1006	Communication Cycle Period	UNSIGNED32	16#0	Refuse Write	False
open I/O Mapping	16#1008	Manufacturer device name	VISIBLE_STRING	'CANopenDeviceExample	✓ Address	10 #1000
	16#100C	Guard Time	UNSIGNED 16	16#0	Index	16#1006
open IEC Objects	16#100D	Life Time Factor	UNSIGNED8	0	• Gelleral	Communication Cycle Pr
	16#1014	COB-ID EMCY	UNSIGNED32	\$NODEID+16#80	Y Type	commanication cycle i
IS	· i6#1016	Consumer Heartbeat Time			Data Type	UNSIGNED32
mation	16#1017	Producer Heartbeat Time	UNSIGNED 16	16#0	Object Type	VAR
inación	16#1018	Identity Object			✓ Value	
	16#1200	ServerSdoParameter			Default Value	16#0
	16#1400	RPDQ communication parameter			High Limit	
	16#1600	RPDO mapping parameter			Low Limit	
	16#1800	TPDO communication parameter				
	E 16#1000	TPDO manping parameter				
	- D 16#3000	TestString				
	16#6000	Read Input 8 Bit	Totocc_ondito			
	16#6200	Write Output 9 Bit				
	15#6412	Write Analogue Output 22 Bit				

Figure 150 ローカルデバイス Object Dictionary タブ

dex 16# 6000	÷0	
ndex	Name	
16#6000	Read Input 8 Bit	
16#6002	Polarity Input 8 Bit	
16#6003	Filter Constant Input 8 Bit	
16#6005	Global Interrupt Enable Digital 8 Bit	
16#6006	Interrupt Mask Any Change 8 Bit	
16#6007	Interrupt Mask Low to High 8 Bit	
16#6008	Interrupt Mask High to Low 8 Bit	
16#6020	Read Input 1 Bit	
16#6021	Read Input 1 Bit	
16#6022	Read Input 1 Bit	
16#6023	Read Input 1 Bit	
16#6024	Read Input 1 Bit	
16#6025	Read Input 1 Bit	
16#6026	Read Input 1 Bit	
16#6027	Read Input 1 Bit	
16#6030	Polarity Input 1 Bit	
16#6031	Polarity Input 1 Bit	
16#6032	Polarity Input 1 Bit	
16#6033	Polarity Input 1 Bit	
16#6034	Polarity Input 1 Bit	
16#6035	Polarity Input 1 Bit	

Figure 151 Object Dictionary タブ Add Object ウィンドウ



8.4.3.4. CANopen I/O Mapping タブ

このタブは、PDOs・SDOs タブでメッセージを登録した場合に表示されます。各メッセージのデータが一覧 表示されており、IO ドライバと同様にこれらに変数を紐づけてユーザーアプリケーションで使用することがで きます。

Seneral	Find		Filter Show all		-	Add FB for IO	Channel	. + Go to Instance
20.04	Variable	Mapping	Channel	Address	Туре	Default Value	Unit	Description
-005	B- *		DigOutput8	%QB70	USINT			
DOs	· · · · ·		DigOutput8	%QB71	USINT			
	B-**		DigOutput8	%QB72	USINT			
og	18- * ø		DigOutput8	%QB73	USINT			
	· · · · · ·		DigOutput8	%QB74	USINT			
ANopen I/O Mapping	· · **		DigOutput8	%QB75	USINT			
			DigOutput8	%QB76	USINT			
ANopen IEC Objects	18- * ø		DigOutput8	%QB77	USINT			
	😟 🦄		DigInput8	%IB300	USINT			
tatus	iii - 🍫		DigInput8	%IB301	USINT			
oformation	· · · · · ·		DigInput8	%IB302	USINT			
monilation	18 - Mp		DigInput8	%IB303	USINT			
	B-*		DigInput8	%IB304	USINT			
	😑 - 🏘		DigInput8	%IB305	USINT			
	😟 – 🍫		DigInput8	%IB306	USINT			
	😟 - 🍫		DigInput8	%IB307	USINT			

Figure 152 CAN デバイス(CANopen) CANopen I/O Mapping タブ

8.4.3.5. <u>Status タブ</u>

このタブは CANopen プロトコルにおけるステータス情報を表示します。表示されるステータス情報については、CANopen プロトコルの規格書などを参照して下さい。



Figure 153 CAN デバイス(CANopen) Status タブ



9. <u>デバッグ機能について</u>

9.1. 基本画面

ログイン(6.11.1 項)完了後、CODESYS-IDE はデバッグモード(ログイン中と同義)へ遷移します。下図は、遷移後のデバッグ画面の一例です。

● SampleProject_Cx2500.project* - CODESYS File Edit View Project Build Online Debug 習 論 副 過 い い 炎 陥 喩 × 鍋 袋 巻 公	Tools Window Help 및 및 및 및 및 관 [1] [1] (1] (1) (1)	Application [Device: F	PLC Logic]	456	I 4 I 4 I 6	¢ ∰ 3 3√	_	× 0 •
Devices 🗸 🕂 🖌 🕅 Device	PLC_PRG X							-
SampleProject_Cx2500 Device.Applicat	ion.PLC_PRG							
Expression		Туре	Value	Prepared value	Address	Comment		b
PLC Logic / ucVal0		USINT	33					
Application [run] vcVal1		USINT	10					
PLC_PRG (PRG)								
B Configuration								
- DIC PRG								
- 🖓 🕤 Cx2500Ain (Cx2500Ain) 😑 2 💿 IP uc	Val0 33 < 255 THEN							
- 😌 👔 Cx2500Cntin (Cx2500Cntin 3 🔹 u	cVal0 33 := ucVal0 33 + 1;							
- 🖓 🛐 Cx2500Din (Cx2500Din) 🗏 4 ELSE								
Cx2500Do (Cx2500Do)	.cval0_33 := 0;							
Cx2500Frqin (Cx2500Frqin 7	•							
9 🔍 RETUR	N							
2								
POUs								100
Assesses - Total () error(s) () warning(s) () message(s)								
	0 error(s) (1) 0 warning(s) (1) 0 mess	age(s) 🗙 🕷						• + •
	· · · · · · · · · · · · · · · · · · ·	-9-(-)				Desired	01	Desiries
Pescription						Project	Object	Position
I ne application is up to date Ruild complete 0 errors - 0 warpings - Ready for download								
build complete o errors, o warnings : Ready for download	8							
Messages - Total 0 error(s), 0 warning(s), 0 message(s)	ch 1 🚮 Breakpoints							
Device user: Anonymous Last build: 😋 0 🕧 0	Precompile RUN	Program le	oaded	Program	n unchanged	Project use	r: (nobody)	6)

Figure 154 デバッグモード メイン画面

#	機能名	摘要
1	Variables Monitor	このエリアでは、表示しているソースファイルで定義した変数値をモニタしたり、強制的に
		値を書き込むことができる。
2	Code Monitor	このエリアでは、表示しているソースファイルで記述したコード及び現在の変数値をモニタ
		できる。ブレークポイントなども設定可能。
3	Login	アイコンを押すとアプリケーションを書き込み、デバッグを開始する。
4	Logout	アイコンを押すとデバッグを終了する。
5	Start	アイコンを押すとアプリケーションは動作を開始する。
6	Stop	アイコンを押すとアプリケーションは動作を停止する。
7	Watch	デバッグ中に監視したい変数とその値等を確認できる。
8	Breakpoints	アプリケーション上に設定したブレークポイントの情報を確認できる。

Table 71 デバッグモード メイン画面 機能概要



9.2. アプリケーションの動作開始・停止

アプリケーションの動作開始・停止は下記の手順でおこなうことができます。

9.2.1. 動作開始

下記にデバッグモードにおけるアプリケーションの動作開始の手順を示します。

① 下図の通り、デバッグモードでアプリケーションが停止していることを確認して下さい。

[2] 22 문 문 (2017) Devices	K MA & MA & III N N N III IIII III IIII III III IIII III III III III III III III III	Device C MainTask	e: PLC Logic) •	99 99 > = 99 10 B) POU_Ctria	x 41 41 41 5	(* 第 국 주	
G Berke (connected) (CX200) G B Four (connected) (CX200) G	Expression VAA RetainVal0 TimeCnt	Type ULINT USINT UDINT	Value 0 0 0	Prepared value	Address	Comment RETAIN VALUE	Ĩ
POU_FB (PRG) Wight Task Configuration V PersistentVars	2 TimeCnt 0 := SysTimeCore	:.SysTimeGetMs();	A	₩			
- 6 명 C-25004ri (C-25004r) - 6 명 C-2500cfin (C-2500cfin - 6 명 C-2500th (C-25000ri - 6 명 C-2500th (C-25000ri - 6 명 C-2500Fin (C-2500Finin) - 6 명 C-2500Finin (C-2500Finin)	<pre>6 GVL.g_udVal0_0 := 100; 7 0 RetainVal0_0 := 100; 9 0 PersistentVars.PersistVal0_00;</pre>	//Call RETA = 200; //Call PERSI:	IN value STENT valueRET	URN			
C G CASSOLIN (CASSOLIN) C G CASSOLIN (CASSOLIN)	 GVL.g_udVal0_g := 100; RetaitVal0_g := 100; PersistentVars.PersistVal0200; 	//Call RSTA = 200; //Call PERST:	IIN value STENF value ne r	URN)			100
- G ≦ C∠2500-m (C∠2500-m) - G ≡ C C∠2500-m (C∠2500-m) - G ≡ C C C C C C C C m) - G ≡ C C C C C C C C m) - G ≡ C C C C C C C C C m) - G ≡ C C C C C C C C C C C C C C C C C C	 GVL.g_udVal0_0 := 100; RetaitVal0_0 := 100; PersistantVars.PersistVal0200; PersistantVars.PersistVal0200; G 0 error(s) @ 0 warning(s) @ 0 	<pre>//Call RETA = 200; //Call PERSIS message(s) × *</pre>	IN value STENF value	183			100

Figure 155 デバッグモード 動作停止状態

② ツールバーの「)」アイコン(運転開始)を押して下さい。

🖆 🚅 🛃 🕼 🗠 여 👃 🖻 🛍	Chillie Debig Ious Wildow Help X 44 십 년 1 비 외 외 관 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	🛛 🕶 🗍 Application [C	Device: PLC Logic] 🔻	9 9 • • • • • • • • • • • • • • • • • •	×	¢ ≢ ≣ + ∿	
SampleProject_Cx2500 SampleProject_Cx2500 Device [connected] (CX2500) Device	Device-Application.POU_CtrIA Expression ValA RetainVal0 TimeCnt	Type ULINT USINT UDINT	Value 0 0 0	Prepared value	Address	Comment RETAIN VALUE	8
	<pre>1 TimeCnt 0 := SysTi 0 OVL.g_udVal0 0 := - 0 RetainVal0 1 := 1007 0 RetainVal0 1 := 1007</pre>	meCore.SysTimeGetMs(); 100; //Call J (200) := 200; //Call PE	ETAIN value RSISTENT value	TURN			
< 2 Cervices POUs Messages - Total 0 error(s), 0 warring(s), 0 to Cervices - Total 0 error(s), 0 warring(s), 0 to Cervices - Cervices - Cer	nessage(s)) 💿 0 message(s) 🔀 💥					100 (C
					Project	Object	Desition

Figure 156 デバッグモード 運転開始アイコンの位置



③ 画面の動作ステータスが「RUN」となりアプリケーションの動作が開始されます。

SampleProject File Edit V	_Cx2500.project - CODES	Online Debug Tools Window Help						- :	×
1	In a X BB	× 桷 结 桷 结 別 別 別 間 簡• 合 []	Application [Devic	e: PLC Logic] • 🤍		= 4 <u>=</u> 4 <u>=</u> + <u>=</u> 1	3 e 쪬 글 장		•
Devices	~ ∓ X	Library Manager	vice 🛛 🚷 MainTask	POU_FB	POU_CtHA	×			-
SampleProje	ect_Cx2500 •	Device.Application.POU_CtrlA							
🖹 😏 📆 Devi	ice [connected] (CX2500)	Expression	Туре	Value	Prepared value	Address	Comment		B.
	Logic	🛊 ValA	ULINT	0					
=-O	Application [run]	RetainVal0	USINT	100			RETAIN VALUE		
	DUT_SAMPLE	TimeCnt	UDINT	1735301					
	GVL								
	Library Manager								
	POU CHA (PRG)								
	POU FB (PRG)			A 7	/				
	Task Configuration		Time Control in						
	T PersistentVars	3	yar imede cria (),						
-0 🗊	Cx2500Ain (Cx2500Ain)	4							
- 😏 🗊	Cx2500Cntin (Cx2500Cntir	5 GVL.g_udVal0 100 := 100;							
- 😏 🗃	Cx2500Din (Cx2500Din)	6							
- 😏 🕤	Cx2500Do (Cx2500Do)	7							
-0 🕤	Cx2500Frgin (Cx2500Frgin	8 RetainVal0[100] := 100;	//Call RETA	IN value	541				
- 😔 🗐	Cx2500Pwm (Cx2500Pwm)	Persistentvars.Persistvalo 200 :=	200; //call PERSI:	SIEWI VALDERETUR	enu .				
<	>								
👷 Devices 👔 PC	DUs								100 🙉
Maccapat - Total D	arror(e) 0 warning(e) 0 m	errana(r)							
incasages - rotaro	croc(a), o warning(a), o m								• • •
		• Overfor(s) • Owarning(s) • Ome	ssage(s) × 🕅						
Description						Project	Object	Position	
T Manager Tab		Conservation (1)							
EI messages - Tota	si u error(s), u warning(s),	o message(s) March ± Mit Breakpoints							
Device use	r: Anonymous	Last build: O 0 🕈 0 Precompile 🗸 RUN	Progra	m loaded	Progra	m unchanged	Project us	ser: (nobody)	6 G

Figure 157 デバッグモード 動作中状態



9.2.2. <u>動作停止</u>

下記にデバッグモードにおけるアプリケーションの動作停止の手順を示します。

① 下図の通り、デバッグモードでアプリケーションが動作中であることを確認して下さい。

Devices - a x	Library Manager	Tim Device State Main	ask 🏹 POU_FI	B POU_CtHA	×		
Comparison (C.C.200 C.200 Comparison (C.C.200 Comparison (C.C.200 Comparison (C.C.200 Comparison (C.C.200 Comparison (C.C.200 C.200 C.200 Comparison (C.C.200 C.200 C.200 Comparison (C.C.200 C.200 C.200	DeveceAppRationPOU_tTRA Epyresion 9 VMA 9 RetainVal0 9 TimeCnt 1 0 TimeCnt 1733301 := SysTi	Type ULINT USINT UDINT meCore.SysTimeGetMs();	Value 0 100 1735301	Prepared value	Address	Comment RETAIN VALUE	ľ
Colorent (Cr2500An) Cr2500An (Cr2500An) Cr2500Cht (Cr2500Cht Cr2500Cht (Cr2500Cht Cr2500Dh (Cr2500Cht Cr2500Dh (Cr2500Cht Cr2500Dh (Cr2500Cht Cr2500Phrm (Cr2500Phrm Cr2500Phrm Cr250Phrm C	<pre>6 GVL.g_udVal0 100 := 7 9 BetainVal0 100 := 100; 9 FersistentVars.PersistVal0</pre>	100; //Call R 200; //Call PES	STAIN value SISTENT value <mark>RET</mark>	URN			

Figure 158 デバッグモード 動作中状態

2 「	┃」アイコン(運転停止)を押して下さい。
-----	----------------------

SampleProject_Cx2500.project - CODES File Edit View Project Build	YS Online Debug Tools Window Help		_			- 1	5 ×
19 년 년 19 10 11 18 19 19 19 19 19 19 19 19 19 19 19 19 19	< 🗛 🍕 🍓 🐫 🗏 🧏 🦄 🦓 🖓 🛗 🛅 - 🕤 🕮	Application [Device: PLC Logi	d • % % → <mark>=</mark> %	CE 6∃ 6∃ →E 5	이 빠르 1		
Devices • 3 × SampleProject_Cx2500 • Device [connected] (CX2500) PICLogic Capacitation (run]	Library Manager The PLC_PRG The Device Application.POU_CtrlA Expression ValA	re 🏹 MainTask 🏹 🖓 Type Value ULINT 0	POU_FB POU_CtHA	Address	Comment		iii
DUT_SAMPLE GNL GN	RetainVal0 TimeCnt	USINT 100 UDINT 1735301			RETAIN VALUE		
■ Task Configuration ● ③ CL300AIR (CL300AIR)	<pre>> TimeCotd 1328001 := SysTimeCore.sys > OVL.g_udVal0 100 := 100; > OVL.g_udVal0 100 := 100; > PersistentVal8 := PersistVal0 1000 := 20</pre>	TimeGetHs(); //Call RETAIN value 00; //Call PERSISTENT val	u*BETURN				
< >> Sevices POUs							100
Messages - Total 0 error(s), 0 warning(s), 0 me	essage(s)						→ ∓ X
Description	 O error(s) O warning(s) O mess 	age(s) × 🕅		Project	Object	Position	
Messages - Total 0 error(s), 0 warning(s), 0	0 message(s) 戻 Watch 1 🧑 Breakpoints						
Device user: Anonymous	Last build: 😋 0 😗 0 Precompile 🗸 🛛 RUN	Program loaded	Progr	am unchanged	Project a	iser: (nobody)	() Q

Figure 159 デバッグモード 運転停止アイコンの位置



③ 画面の動作ステータスが「STOP」となりアプリケーションの動作が停止します。

Devices • # 1	Library Manager PLC_PRG	🔐 Device 🛛 🕵 Main	rask 👔 POU_	FB POU_CtrlA	×			•
C D Desize [connected] (CX250 C C Desize [connected] (CX250 C C C C Desize [connected] (CX250 C C C C C C C C C C C	Expression	Type ULINT USINT UDINT	Value 0 0	Prepared value	Address	Comment RETAIN VALUE		
PersistentVars Grading Concentration	<pre>the GVL.g_udVal0 0 := 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</pre>	= 100; //Call R	ETAIN velue					
	tr 0 Gentla, 1 < 0	= 100; //Call R 10000 := 200; //Call PS () ● 0 message(a) × ¥	ETAIN value SISTENT value <mark>er</mark>	1.83			100	r New York Street Stree

Figure 160 デバッグモード 動作停止状態



9.3. <u>シングルサイクル</u>

CODESYS-IDE では、アプリケーションを1サイクルだけ実行する機能があります。

デバッグモード中にメニューバーの「Debug」 → 「Single Cycle」を選択すると、各タスクを1サイクル実行 しアプリケーションが停止します。

🗄 🍯 📓 🚔 In o 🙏 🖻 💼 🤇	< M 2	+	Start Stop	F5 Shift+En	pplication [Devi	e: PLC Logic] • C	8 08 • # 96 []	1 93 63 *N (3 ↓ ■ ■	12/			
Devices 🗸 🗸 🛪			Single Cycle	Ctrl+F5	S MainTas	POU_FB	POU_CtrlA	×					
SampleProject_Cr2000 SampleProject_Cr2000 B Device [connected] (Cl2500) B P.C Lopic B P.C Lopic B P.C Lopic D DIT_SAMPLE	Devic Express	90 B	New Breakpoint Edit Breakpoint Toggle Breakpoint Disable Breakpoint Enable Breakpoint	F9	pe NT NT NT	Value 200 100 2635447	Prepared value	Address	Comment RETAIN VALU	E			10
Lbrary Manager PLC_PRG (PRG) POU_CtrlA (PRG)		() () () () () () () () () () () () () (Step Over Step Into Step Out	F10 F8 Shift+F10			-						
Task Configuration TersistentVars Griff Cx2500Ain (Cx2500Ain)	1 2 3 4	1 00 P	Run to Cursor Set Next Statement		eGetMs();								
G Cx2500Cntin (Cx2500Cntin G Cx2500Dn (Cx2500Dn) G Cx2500Dn (Cx2500Dn) G Cx2500Dn (Cx2500Dn) G Cx2500Prim (Cx2500Prim) G Cx2500Prim (Cx2500Prim)	5 6 7 9 10	5	Write Values Force Values Unforce Values Toggle Flow Control Mode	Ctrl+F7 F7 Alt+F7	//Call RET //Call PERSI	NIN value STENT value							
	11		Core Dump Display Mode Check Memory for Active App	ication	- <u>-</u> +	1; RETURN							
Devices POUs												100	1
Aessages - Total 0 error(s), 1 warning(s), 7 m	essage(s)											- 4	×
Build			- O error(s) 😗 1 warning	s) 0 7 messag	ie(s) 🗙 🕅								
Description		_								Project	Object	Position	^
 Memory area 2 contains Retain data: size Memory area 3 contains Persistent data: : Build complete 0 arrors 1 marring - Re 	: 4072 byte size: 8168 b arbs for door	s, hig iytes,	hest used address: 1, largest cont highest used address: 45, largest	guous memory ga contiguous memo	p: 4071 bytes (99 9 ry gap: 8123 bytes	6) 99 %6)							•
Messages - Total 0 error(s), 1 warning(s), Device user: Anonymous Last but	7 message(s) 50	Watch 1 🖉 Breakpoints Precompile 🧹	Pro	oram loaded	Prog	ram unchanged	Project	user: (nobody)	INS	Ln 6 Col 1	Ch1 (2

Figure 161 シングルサイクルの選択

9.4. <u>変数モニタ</u>

デバッグモードでは、各 POU の Variable Monitor(変数宣言部)や Code Monitor(コード部)、グローバル変数 リスト画面で変数の現在値が表示されます。下図は、その一例として ST の画面を挙げます。LD や FBD の場合 は、実行中(TRUE)のアイテム(接点など)・接続ラインが黒→青色に変わります。

Expression ucVal Idx bFlg	ni 1	Туре						
 ucVa Idx bFlg 	all		Value	Prepared value	Address	Comment	^	
IdxbFlg		USINT	10					
bFlg		UINT	11					
	1	BOOL	FALSE			Declaration of "BOOL"		
ucVal	alO	USINT	7			Declaration of "UINT"		
<pre>rVal</pre>		REAL	1			Declaration of "REAL"		
🔷 sStr		STRING	'Tokyokeiki'			Declaration of "STRING"	- 1	
usBo	ox	UINT (UINT#2001	200			Declaration of "UINT(range : 200-1000)"		
e 🚸 usAn	ry	ARRAY [010] OF U				Declaration of "ARRAY(UINT)"		
e < stSm	np0	ST SAMPLE STRUCTO				Declaration of "STRUCT"		
2 3 0 4 0	//Array FOR Idx 11 := 0 TO 10 DO usAry[Idx 11] ??? := Idx 11	* 2;						
5 6 7 8 9 10	END_FOR (* Access *) //Struct atSmp0.ucMember0[0] := 0; c75m0 = 10 kmbar0	- 10.						
12 0	stSmp0.urMemberou 10 :	= 10;						
	Scompo. derenarrayo [0] H := 11;							

Figure 162 デバッグモード POU 画面



9.4.1. <u>変数値表記方式の変更</u>

変数モニタでは値の表記形式(2・10・16進数)を変更できます。ここでは、2進数→16進数への変更を例に手順を示します。

① メニューバーから、「Debug」→「Display Mode」を選択すると、表記方式の候補が表示されるので所望のものを選択して下さい。下図の✔印は、現在の表記方式が Decimal(10 進数)であることを意味します。

19 - 19 - 19 - 19 - 19 - 19 - 19 - 19 - 19	XIA	*	Start Stop	F5 Shift+F8	pplication [Device:	PLC Logic] 🔹 🔍	• • • • • • • • • • • • • • • • • • •	1 43 43 +3 Ş	◆ 罰 π'	₹/		
Devices 🗢 4 3	ĸ /1		Single Cycle	Ctrl+F5	MainTask	POU_F8	POU_CHA	GVL				-
SampleProject_Cr2500 G Device [connected] (Cr2500 G Device [connected] (Cr2500 G PLC Logc G Application [stop] * DUT_SAMPLE G OT_SAMPLE	Devi	0 0 0	New Breakpoint Edit Breakpoint Toggle Breakpoint Disable Breakpoint Enable Breakpoint	F9	ре INT IT OL	Value 10 11 FALSE	Prepared value	Address	Comment Declaration of	'800L"		^ b
GVL GVL Charay Manager CLC PRG (PRG) POU_CtrlA (PRG) POU_CtrlA (PRG) CLC PRG (PRG) CLC PR	* * * * *	日日日日の	Step Over Step Into Step Out Run to Cursor Set Next Statement	F10 F8 Shift+F10	NT AL RING IT (UDIT#2001 RAY [010] OF U SAMPLE_STRUCTO	7 1 'Tokyokelki' 200			Declaration of Declaration of Declaration of Declaration of Declaration of	'UINT" 'REAL" 'STRING" 'UINT(range 'ARRAY(UIN 'STRUCT"	: : 200-1000)' т)*	
-G CL2500An (Cx2500An) -G CL2500Chin (Cx2500Ch) -G Cx2500Ch (Cx2500Ch) -G Cx2500Ch (Cx2500Ch) -G Cx2500Ch (Cx2500Ch) -G Cx2500Ch (Cx2500Ch) -G Cx2500Ch (Cx2500Ch) -G Cx2500Ch (Cx2500Ch)	ntir E ajn m)		Show Next Statement Write Values Force Values Unforce Values Toggle Flow Control Mode Core Dump	Ctrl+F7 F7 Alt+F7	21							^
	1		Display Mode		Binary							
<	> 11	0.0	Check Memory for Active Appl stSmp0.ucMemArray0[0] stSmp0.ucMemArray0[1] 22	:= 11; := 22;	Decimal Hexadecim	st						100 🔍 🗸
Messages - Total 0 error(s), 1 warning(s), 7	message(s)											• * ×
Build			O error(s) I warning(s) 🧿 7 messa	ige(s) 🗙 🕅							
Description Memory area 2 contains Retain data: s Memory area 3 contains Persistent dat Bild coerclate == 0 arcor. 1 sections :	ize: 4072 by a: size: 8168 Deady for dr	bytes	ghest used address: 1, largest contig , highest used address: 45, largest o 4	guous memory g contiguous mem	ap: 4071 bytes (99 %) ory gap: 8123 bytes (95	%)				Project	Object	Position ^

Figure 163 Display Mode の選択

② 選択後、表記が変更されます。



Figure 164 変数モニタ 16 進数表記



9.4.2. <u>ウォッチ</u>

通常、デバッグモード時の変数値は 9.4 節の通り POU や変数リストでモニタできます。ただ、別々の POU に ある変数などを同時に見ることが難しい場合があります。その場合はウォッチリストにモニタしたい変数を登録 することで実現できます。ウォッチリストは最大 4 つ利用できます。

		r PLC_PRG	🗙 📄 Device 🏾 🎲 MainTa	k CP POU_FB	POU_CtrlA	🕵 Task1		
SampleProject_Cx2500	Device.Application	.PLC_PRG						
E Connected] (CX2	Expression		Туре	Value	Prepared value	Address	Comment	
E PLC Logic	🔳 🔿 usAry		ARRAY [010] OF	J			Declaration of "ARRAY(UINT)"	
Application [stop	🗄 🚸 stSmp0		ST_SAMPLE_STRU	то			Declaration of "STRUCT"	
DUT_SAMPLE	🗄 🍦 stSmp1		ST_SAMPLE_STRU	Τ1			Declaration of "STRUCT(EXT)"	
GVL	ucTmp0		USINT	0			Declaration of "USINT"	
Library Manager	🛞 < ucTmpAry		ARRAY [03] OF U	S			Declaration of "ARRAY(USINT)"	
	🛞 🧳 unSmp		UN_SAMPLE0				Declaration of "UNION"	
POU_CEIA (PRG) < enEnum0		EN_SAMPLE_ENUN	ENUM_SP0			Declaration of "EUNM"	
B · G 양 MainTas 레 PLC_PR - 레 POL_PE - G 양 Task1 네 POL_FE 파 PerstentVars	G H 3 FOR Idad	0 := 0 TO 10 ; ry[Idx 0] 0	DO] := Idx 0 * 2;					
Poug Poug	G = 3 ● FOR Ids G = 3 ● FOR Ids H = 5 ■ FOR Ids G = 5 ■ FOR I	0 := 0 TO 10 : ry[Idx 0] 0 ess *) t ucMember0[0 := 0 ulMember0[0]	DO := Idx 0 * 2;					100
Poly Poly	G 2 //Array G 3 0 TOR Ids B 3 0 TOR Ids B 0 02A B 200 FOR C 4 Acce C 7 (* Acce C 5) 0 0 atSmp0.1	0 := 0 TO 10 ry[Idx 0] 0 ess ') t ucMember0 0 := 0 ulMember0	DO] := Idx 0 * 2;); 0 := 10;		_			100
Poly Pol	G 4 2 //Array G 5 708 104 usA END FOR 104 usA END FOR 1 0 4 stmp0. 1 0 stSmp0. Application	0 := 0 TO 10 ry[Idx 0] 0 ess ') t ucMember0 0 := 0 ulMember0	DO := Idx 0 + 2; ; 0 := 10; Value Prepared	alue Execution po	nt	Address	Comment	100
G & ManTad G & ManTad G & C_R	G = //Array G = 3 F78 IG4 + 19 = 4 uak END_F08 + 0 (* Acce 0 (* Acce 1 0 estmp0, 11 estmp0, 10 estmp0, 10 estmp0, 10 estmp0, 11	0 := 0 TO 10 : ry[Idx 0] 0 ess *) t ucMember0 0 := 0 ULIMT	DO] := Idx 0 * 2; 0 := 10; Value Prepared 0	alue Execution poi	nt 9	Address	Comment	100
	G → //Arrey G → POR Ids/ U → ZND_FOR → C + Accession C +	0 := 0 TO 10 ry[Idx 0 0 0 t ucMember00 := 0 ulMember0 Type ULINT ARRAY [0.3] O	D0 1 = Tdx 0 + 2; 1 0 = 10; Value Prepared 0	alue Execution poi Cyclic Monitori Cyclic Monitori	nt g	Address	Comment Declaration of *ARR	100
enters () Pol. Pol. Cran. Pol. Cran. Pol. Cran. Pol	G = //Array G = 0 FOR IGA UaA E ZND_FOR 0 4 Control Co	0 := 0 TO 10 ry [dx] 0 0 t 0 t uddeaber0(Type ULNT USINT	0 1 == Tata o + 27 12 13 14 14 14 14 14 14 14 14 14 14	alue Execution poi Cyclic Monitori Cyclic Monitori Cyclic Monitori	nt 9 9	Address	Comment Declaration of "ARR	100
evices () POUS POU	G → //Arrey G → POR 1dd U → POR 1dd U → U → U → U → U → U → U → U → U → U →		0 1 = 1 dx 0 + 27 1 1 0 1 = 10; 10 10 10 0 0 0 0 0 0	alue Execution poi Cyclic Monitori Cyclic Monitori Cyclic Monitori Cyclic Monitori	nt 0 0 9 9	Address	Comment Declaration of "ARR	100
C C	G = //Array G = PCR 104 + CR 1	6 := 0 10 10 ry[Idx[6 6 6 ess ') t 10 10 t uclember(1= 0 10 Type uclemt 488AY(0.5) 0 USBNT USBNT USBNT 10	D0 := Idx 0 + 2; v lic 0 + 2; Value Prepared 0 0 0 0 0	alue Execution poi Cyclic Monitori Cyclic Monitori Cyclic Monitori Cyclic Monitori Cyclic Monitori	nt 0 0 9 0 0	Address	Comment Declaration of "ARR	100

Figure 165 ウォッチリストの位置

メイン画面に表示されていない場合は、メニューバーの「View」→「Watch」からウォッチリストを選択して下さい。



9.4.2.1. <u>変数の登録</u>

ウォッチリストへ変数を登録する手順を示します。

① デバッグモードにてウォッチリストに登録したい変数にカーソルを合わせ、右クリックして下さい。

≌ ⊜ k∩ ci } t <u>6</u>	s×ias≤sesi∎ ***a i≊	i I 🏧 + 📑 I 🛗 Application [D	evice: PLC Logic] 🔹 😋	¢ ∰ → <u>≡</u> ≪ [0)를 수필 수필 - 음 (일	¢ 국 🎶	
vices • 4	Library Manager PLC_PR	G 🕅 Device 🚱 Main	fask 👔 POU_FB	POU_CtrlA	🗙 🚷 Task1]	_
B G Device [connected] (CX250	Expression	Type	Value	Prepared value	Address	Comment	
= 1 PLC Logic	/ ValA	ULINT	0				
= O Application [stop]	RetainVal0	USINT	0			RETAIN VALUE	
GVI GVI	TimeCnt	UDINT	0				
Library Mapager							
PLC PRG (PRG)							
POU_CHA (PRG)	5 GVL.g_udVal0 0	:= 100;					
POU_FB (PRG)	6						
🖃 🧱 Task Configuratio		((0-11.7)					
🖹 🕞 🥩 MainTask	Retainvalo := 100;	Zal0[200] := 200: //Call PE	SIMIN VALUE				
- @ PLC_PRG	10	1000 1- 2007 // Outr 11					
- an equilated	11						
E1 . 00_ca		< 10 THEN					
≅-⊕ 🍪 Task1	H 12 O IF ValA 0						
≅-⊕⊗ Task1 -⊕ POU_FB	12 0 IF ValA 0	:= ValA 0	- 1;				
= ⊖ S Task1 — ⊕ POU_FB — ▼ PersistentVars	12 0 IF ValA 0 13 0 ValA 0 14 ELSE ValA 0	:= ValA 0	- 1;				
= ⊖ S Task1 — ● POU_F8 — ■ PresistentVars — © @ c>snniks (c>snniks		:= ValA 0 := ValA 0	- 1; + 5 ;				
Pevices Pous		:= ValA 0 := ValA 0	- 1; + 5 ;				100
Devices Pouls	H = 12 ● IF ValA ● ValA ● ValA ● ValA ● ValA ● Is ● ValA ● Is	:= ValA 0	- 1; + 5 ;				100
Devices Polls		i = Vala 0 i = Vala 0	- 1; + 5 ; dvalue Execution poi	nt	Addres	s Comment	100 🕅

■ Messages - Total O enro(b), 0 varming(b), 12 message(b) P Wath 1 @ Predpoints Device user: Anonymous Last build 0 0 0 Precomple Processie Program loaded Program unchanced Protect user: (nobody) 0 G Figure 166 リストに登録したい変数の選択

② 表示されるコンテキストメニューから「Add to Watchlist」を選択して下さい。

File Edit View Project Build Online Debug To 10 20 20 20 20 20 20 20 20	ols Window Help 비 케 캐 隆 隆)•	📑 🏥 Application [De	vice: PLC Logic] 🔹 🔾	\$ \$\$ → = % (I 9I 4I 4I 용	¢ ∰ = ™	۲
Devices - A X Ibrary Manage	r PLC_PRG	👔 Device 🛛 🚷 MainTa	ssk 👔 POU_FB	POU_CtrlA	🗙 🎲 Task1		
SampleProject_Cx2500 Control Device_Application B-G Device [connected] (CX250 Expression	.POU_CtrlA	Туре	Value	Prepared value	Address	Comment	
Application [stop] Application [stop		Cut Copy Paste Delete				RETAIN VALUE	
	dVal0 0 :=	Select All Browse	•				
G G G G G G G	0 A 0 A 0 A 0	Add to Watchlist Advanced Input Assistant Refactoring Edit Declaration Header.	•				
Le END_IF	4	Move Down					100
Watch 1	*	Move Up					→ 4
Expression Application	Type ∖ ĕ	New Breakpoint Toggle Breakpoint		it	Address	Comment	
	+3	Run to Cursor Set Next Statement					
		Unforce All Values of 'De Display Mode	vice.Application'				
Messages - Total 0 error(s), 0 warning(s), 12 message(s) 🕎 Watch	1 Dreakpoints						

Figure 167 Add Watchlist の選択



③ ウォッチリストに所望の変数が登録されます。

	👻 🕂 🗙 🎢 Library Mana	ger 👔 PLC	PRG R Device	e 🎲 MainTask	POU_FB	POU_CtrlA	🗙 🎲 Task1		
SampleProject_Cx2500	Device.Application	on.POU_CtrlA							
= II PLC Logic	Expression			Туре	Value	Prepared value	Address	Comment	
- O Applicati	ion [stop]			ULINT	0				
🖲 🛄 DUT_S	SAMPLE Provide			USINT	0			RETAIN VALUE	
- 🎒 GVL	# limeCnt			UDINT	0				
- 📶 Library	y Manager								
PLC_P	RG (PRG)		1.00		à 🔻				
] POU_(CtrlA (PRG) 5 GVL.g	udvalu 0	:= 100;						
	FB (PRG) 7								
Task C	Configuration 8 🖉 Retain	Val0 := 10	0;	//Call RETAIN	value				
=- 54 G	ManTask 9 Persis	tentVars.Persi	stVal0200 := 20	0; //Call PERSIST	NT value				
-0	PLC_PRG 10								
- 0.4	Tark1 12 12 0 TP Val		< 10 PUPN						
		1A 0	:= ValA	0	- 1:				
7									
- Persis	tentvars La Labora								
Perss R R Cy2500Ain	(CV2500Ain [*] 15 Va	1A 0	:= ValA	0	+5;				
- Carell Cy2500Ain	tentvars (⊂>>sno.in. ¥ 16 END_IF	1A <u>0</u>	:= ValA	0	+5;				
Res POUs	ICV2500Ain' 16 Va 16 END_IF 17 RETURN	1A 0	:= ValA	0	+5;				100
ices POUs	(~		:= ValA	0	+ 5 ;				100
Ces POUs	Application	1A 0	:= ValA	0 Prepared value	Execution point	nt	Address	Comment	100
Ces [] POUs ces [] POUs con POU_CtrlA.ValA	Application Device.Application	1A 0 Type ULINT	:= ValA Value 0	0 Prepared value	Execution point	nt 9	Address	Comment	100



9.4.2.2. <u>変数の削除</u>

ウォッチリストから変数を削除する手順を示します。

① デバッグモードにてウォッチリストから削除したい変数にカーソルを合わせ、右クリックして下さい。

evices ¥ 4 3	Library Manager PLC_P	RG 🗙 🖷 Device 😒 MainTask	POU_FB	POU_CHA	🔮 Task1]	
SampleProject_Cx2500	 Device.Application.PLC_PRG 						
= 😏 🔟 Device [connected] (CX250	Expression	Туре	Value	Prepared value	Address	Comment	^
- EH PLC Logc	usBox	UINT (UINT#2001	200			Declaration of "UINT(range : 200-1000)"	
= O Application [stop]	🗄 🌵 usAry	ARRAY [0., 10] OF U				Declaration of "ARRAY(UINT)"	
* OUT_SAMPLE	/	ST SAMPLE STRUCTO				Declaration of "STRUCT"	
- 🧭 GVL	🖹 🌵 stSmp1	ST SAMPLE STRUCT1				Declaration of "STRUCT(EXT)"	- 1
Library Manager	● ucTmp0	USINT	0			Declaration of "USINT"	
PLC_PRG (PRG)	H de ucTmoAry	APPAY ID. 3] OF US				Declaration of "ARRAY(USINT)"	
POU_CHA (PRG)	🗎 🌒 unSmp	LIN SAMPLED				Declaration of "UNION"	
DOU_FB (PRG)	anEnum)	EN CAMPLE EN MO	EN IN COD			Declaration of "ELINM"	- 1
- ④ POU_CH# = · ⑤ 윻 Task1 - ④ POU_FB	B 3 POR Idx 0 := 0 TO 3 4 usAry[Idx 0] 6 5 END_POR 6	0 DO					
	■ 3 ● TOR Ida ● := 0 TO :	0 DO = := Idx O * 2; = 0; O := 10; O := 11;					00
	<pre>B 3 FOR Idd 1 := 0 T0 1 4 UskryIdd 0 5 5 5 5 5 5 7 7 7 7 7 7 7 7 7 7 7 7 7</pre>	0 DO := Idx 0 * 2; 0; 0; 0; 10; 10;					00 (
	B 0 PDB. Idm.] = 0 TO] 4 0 UASY Idm.] 0 5 PDD. TSB TO] 0 7 0 (* Access *) 0 10 0 # ZSP v. USB 0 11 0 # ZSP v. USB 0 12 # ZSP v. USB v. USB v. USB 0 Application Top 10 10	0 00 1 = Ids 0 + 2; 0; 0; 0; 1 = 10; 0; 0; 0; 0; 0; 0; 0; 0; 0;	2 Execution pc	pint	Address	2 Comment	00 (
⊕ Pour, cel ⊕ Pour, sel ⊕ Pour,	B 0 TOX Edg 1 </td <td>0 00 := Idx • 2; 0; 0; := 10; 0; := 10; Where Prepared value 0</td> <td>e Execution pr Cyclic Monitor</td> <td>vint ng</td> <td>Address</td> <td>Comment</td> <td>•</td>	0 00 := Idx • 2; 0; 0; := 10; 0; := 10; Where Prepared value 0	e Execution pr Cyclic Monitor	vint ng	Address	Comment	•

Figure 169 リストから削除したい変数の選択

② 表示されるコンテキストメニューから「Delete」を選択して下さい。

Devices	- a :	🗙 🁔 Library Manager	PLC_PRG	X PDevice (🖇 MainTask	POU_FB	POU_CHA	🕑 Taski		
= 🗿 SampleP	roject_Cx2500 💌	 Device.Application.PLC_P 	RG							
8-9 🗊 🛛	evice [connected] (CX250	Expression		Туре		Value	Prepared value	Address	Comment	^]
- 1	PLC Logic	/ usBox		UINT ()	JINT#2001	200			Declaration of "UINT(rang	e : 200-1000)"
=-(Application [stop]	🖲 🌵 usAry		ARRAY	[010] OF U				Declaration of "ARRAY(UI	NT)"
	* C DUT_SAMPLE	🖹 🌵 stSmp0		ST SAM	PLE STRUCTO				Declaration of "STRUCT"	
	GVL	🖹 🍦 stSmp1		ST_SAM	PLE STRUCT1				Declaration of "STRUCT(E	XT)"
	Library Manager	ucTmp0		USINT	-	0			Declaration of "USINT"	·
	PLC_PRG (PRG)	🛞 🆸 ucTmpAry		ARRAY	[03] OF US				Declaration of "ARRAY(US	INT)"
	POU_CtriA (PRG)	🛞 🌵 unSmp		UN_SA	MPLE0				Declaration of "UNION"	
	POU_FB (PRG)	enEnum0		EN_SAI	MPLE_ENUMO	ENUM_SP0			Declaration of "EUNM"	~
-01 -01 <	Cx2500Ain (Cx250)	Copy Parte Delete Select All	10 := 0; 10 0	:= 10;						100
		Browse	•							100 (5%
Watch 1	24	Innut Assistant		Vel	Descendentland	f financia		Address	Comment	• • •
expression			_	value	Prepared value	Execution poin	-	Address	Comment	
# P00_00	10 IG	New Breakpoint		,		Cyclic Monitorini	2			
		Toggle Breakpoint								
		Write Values								
		Force Values								
		Unforce Values								
		Unforce Values Add all Forces to Watchlist								
🖪 Messages - T	'otal 0 error(s), 0 warn	Unforce Values Add all Forces to Watchlist	akpoints							

Figure 170 Delete の選択



③ これで変数がウォッチリストから削除されます。

	X Library Manager P	LC_PRG 🗙 👘 Device 🚷 MainTask	POU_FB	POU_CtrlA	S Task1		
SampleProject_CX2500 Device [connected] (CX250	Device_Application.PLC_PRG					A	
PLC Logic	Expression	lype	Value	Prepared value	Address	Comment	
- O Application [stop]	/ usBox	UINT (UINT#2001	200			Declaration of "UINT(range : 200-1000)"	
B 🖨 DUT. SAMPLE	🗷 🤣 usAry	ARRAY [010] OF U				Declaration of "ARRAY(UINT)"	
GVL	🗷 🧳 stSmp0	ST_SAMPLE_STRUCT0				Declaration of "STRUCT"	- 6
Library Manager	🗄 🌵 stSmp1	ST_SAMPLE_STRUCT1				Declaration of "STRUCT(EXT)"	
	ucTmp0	USINT	0			Declaration of "USINT"	
	🗷 🌵 ucTmpAry	ARRAY [03] OF US				Declaration of "ARRAY(USINT)"	
	🗄 < unSmp	UN_SAMPLE0				Declaration of "UNION"	
POD_FB (PRO)	enEnum0	EN_SAMPLE_ENUMD	ENUM_SP0			Declaration of "EUNM"	~
C (22 a press) (a press)	0 (* Access *)						
Image: Second secon	<pre>> //Struct 10 stSmp0.ucMember0 11 stSmp0.ulMember0 12 stSmp0.ucMemArray0</pre>	:= 0; 0 := 10; [0]0 := 11;				1	00 🛃
← ⊕ ∰ Cx2500Ain (Cx2500Ain ← ⊕ ∰ Cx2500Cintin (Cx25000 ← ⊕ ∰ Cx2500Din (Cx25000 → Pevices ♪ POUs h 1	<pre>v //Struct 10 stSmp0.ucMember00 11 stSmp0.ulMember00 12 stSmp0.ucMemArray0</pre>	:= 0; 0 := 10; 0 0 := 11;				1	00 🛃
G III Cx2500An (Cx2500An G III Cx2500Cnth (Cx2500Cn G III Cx2500Ch (Cx25000 Ar III Cx2500Ch (Cx25000 Ar III Cx2500Ch (Cx2500Ch Ar III Cx2500Ch Ar III Cx250Ch Ar III CX250C	 9 //Struct 10 @ stSmp0.ucHember0@ 11 @ stSmp0.ucHember0@ 12 @ stSmp0.ucHemArray0 	:= 0; 0 := 10; (0) 0 := 11; Value Prepared valu	e Execution poir	nt	Address	2 Comment	00 🛃
	9 //Struct 10 # stSmp0.ucMember013 11 # stSmp0.ucMember013 12 # stSmp0.ucMember013 Application Type	:= 0; 0 := 10; (0) 0 := 11; Value Prepared valu	e Execution poir	nt	Address	; Comment	00 g

Figure 171 変数削除後

9.5. 値の書き込み・強制

ユーザーは、デバッグモード中に変数値を任意で設定できます。デバッグモード中の値の設定方法としては以 下2通りの手法があります。値の設定は、実機の動作に重大な影響を与える可能性があります。設定の際は、必 ずアプリケーション動作に与える影響を念入りに考慮・検討を重ね、問題無いと判断した上で実施して下さい。

9.5.1~9.5.3 項に変数宣言部での設定・解除手順を記載しております。値の書き込み・強制はウォッチリストでも同様に設定可能です。

名称	摘要
値の書き込み	アプリケーション動作開始時若しくは動作中に値の書き込みをした時に 1 度だけ対象の変数
	に値を設定します。この方法の場合、次のサイクルやアプリケーション動作によっては、変数
	値が上書きされる可能性があります。
値の強制	毎サイクル変数に値を書き込みます。これにより、変数の値を強制解除(9.5.3 項参照)するま
	で保持し続けます。

Table 72 デバッグモード中の変数値設定方法



9.5.1. <u>値の書き込み</u>

ここでは、変数への値の書き込み手順を示します。

① 書き込みたい変数の変数宣言部にて Prepared value 列に所望の値を入力します。

SampleProject_Cx2500.project* - CODESYS							-	o ×
File Edit View Project Build Online	Debug Tools Window Help							₹2
9 🚅 🖬 🚑 🗠 ⊲ X 🗈 🙉 X 👪 🕅	6 🚳 64 II 🐄 🔊 🖄 📾 185 -	Di Itali Application [[Device: PLC Logic1 .	os os 🖕 🚽 🥸 so	10= == d= += 2	[이뻐글] 것		
			Jerred Fee cogie;	-9-9 1 1 1	14	- - page •• •o		
Devices 👻 📮	X ry Manager PLC_PRG	🖷 Device 🛛 🔝 Ma	inTask	FB / POU_Cti	IA 🗙 🔂 Taski	Task Configuration	Cx25	500Ain
SampleProject_Cx2500	Device Application POU CtrlA							
E-G Device [connected] (CX2500)	Everation	Turne	Value	Drepaged value	Address	Comment		T
- I PLC Logic	Expression	lype	value	Prepared value	Address	Comment		-
- O Application [stop]	🖗 ValA	ULINT	135	1000				
DUT_SAMPLE	RetainValu	USINT	100			RETAIN VALUE		
🎑 GVL	@ limeCnt	UDINT	13422645					
- 📶 Library Manager								
PLC_PRG (PRG)								
POU_CtrlA (PRG)								
POU_FB (PRG)	1							
= 🧱 Task Configuration	2 TimeCnt 13422645	:= SysTimeCore.SysT	imeGetMs();					
🖻 🚭 🥩 MainTask	3							
PLC_PRG	4							
DOU CITA	5 GVL.g_udVal0 10	0 := 100;						
E- S S Task1								
DOU FB	B RetainVal0100	100.	(/Call DETAIN	178 7 118				
PersistentVars	9 PersistentVars, Pe	rsistVal0 200 := 200	: //Call PERSISTE	NT value				
- (x2500Ain (Cx2500Ain)	10							
	11							
- G fill Cx2500Din (Cx2500Din)	12 IF Valk	135 <135>	> 10 TH	EN				
- (2 m Cx2500Do (Cx2500Do)	13 ValA	135 <135>	:= Val	A	135 <135>	- 1;		
Cx2500Ergin (Cx2500Ergin)								
Devices POUs	15 ValA	135 <135>	:= Val	A	135<135>	+ 500 ;	[100
essages - Total 0 error(s), 0 warning(s), 12 message(s)								• 4 X
uild	O error(s) O warning(s)	🕽 7 message(s) 🗙 💥						
Description						Project	Object	Position
Memory area 1 contains Data, Input, Output and Nor	nsafe data: size: 86016 bytes, highest used a	address: 9676, largest contig	uous memory gap: 7634	0 bytes (88 %)				
Memory area 2 contains Retain data: size: 4072 byte	s, highest used address: 1, largest contiguou	is memory gap: 4071 bytes ((99 %)					
Memory area 3 contains Persistent data: size: 8168 b	oytes, highest used address: 45, largest conti	iguous memory gap: 8123 by	rtes (99 %)					
Build complete 0 errors, 0 warnings : Ready for dow	inload							
Messages - Total 0 error(s), 0 warning(s), 12 message	(s) 🔛 Watch 1 🔊 Breakpoints							

Figure 172 値の書き込み Prepare value への値の入力

② メニューバーから「Debug」 \rightarrow 「Write Values」を選択して下さい。

											1.
🖹 ൙ 🖬 🚑 🗠 🗠 🖇 🛍 🗙 🖡	42 >	Start	F5	pplication [Devic	e: PLC Logic] • 🥰	\$ \$\$ } = % [0	E CI CI AI S	(中國)言	21		
		Stop	Shift+F8								
Devices -	4	Single Cycle	Ctrl+F5	MainTas	sk 0 POU_FB	POU_CtrlA	🗙 🔮 Task1	Task Config	uration	Cx:	2500Ain -
SampleProject_Cx2500	- 10	New Breakpoint									
B G Device [connected] (CX2500)	一马	Edit Breakpoint		De.	Value	Prepared value	Address	Comment			1
B DLC Logic		Toggle Breakpoint	F9	INT	135	1000					
Application [stop]	0	Disable Breakpoint		INT	100	1000		RETAIN VALUE			
DUT_SAMPLE		Enable Breakpoint		INT	13422645						
GVL Ubrary Manager	C III	Step Over	F10								
PLC PRG (PRG)	6-	Step Into	Eg								
POU_Ctria (PRG)	0.0	Step Into	Chift. 510								
POU_FB (PRG) Task Configuration		Step Out	200141-00			A 4					-
Task Configuration		Kun to Cursor		Core.SysTime(GetMs();						
B - G S MainTask		Set Next Statement									
⇒ S S MainTask		Show Next Statement									
- @ POU_CHA		Write Values	Ctrl+F7								
		Force Values	F7		((C-11 DEPATH -	1					
T PersistentVars		Unforce Values	Alt+F7	2001 1= 2001 /	/Call PERSISTENT	r talte					
G (Cx2500Ain (Cx2500Ain)	-07	Toggle Flow Control Mode									
		Core Dump									
- 🕞 🗑 Cx2500Din (Cx2500Din)				5>	> 10 THEN	N					
- 5 🗑 Cx2500Do (Cx2500Do)		Display Mode		35>	:= ValA	1	15 <135>	- 17			
Cx2500Erain (Cx2500Erain)	_	Check Memory for Active App	lication	1355	:= Vala	1	15<135>	+ 500			
2 Devices In POUs		IS PAT TP									100
Messages - Total 0 error(s), 0 warning(s), 12 message	pe(s)										- 4 X
Build		 O error(s) O warning 	(s) 🚯 7 messa	ige(s) 🗙 💥							
Description									Project	Object	Position *
	nd Nonsafe	data: size: 86016 bytes, highest u	sed address: 96	76, largest contiguous	memory gap: 76340 b	oytes (88 %)					
9 Memory area 1 contains Data, Input, Output ar		hest used address: 1. largest cont	iguous memory o	ap: 4071 bytes (99 %	i)						
 Memory area 1 contains Data, Input, Output ar Memory area 2 contains Retain data: size: 407; 	2 bytes, his	hiene and and can the Rear call									
 Memory area 1 contains Data, Input, Output ar Memory area 2 contains Retain data: size: 4073 Memory area 3 contains Persistent data: size: 8 	2 bytes, hij 168 bytes,	, highest used address: 45, largest	contiguous mem	ory gap: 8123 bytes (99 %)						
 Memory area 1 contains Data, Input, Output ar Memory area 2 contains Retain data: size: 4073 Memory area 3 contains Persistent data: size: 8 Build complete 0 errors, 0 warnings : Ready for 	2 bytes, hi 1168 bytes, ir download	, highest used address: 45, largest d	contiguous mem	ory gap: 8123 bytes (99 %)						
Memory area 1 contains Data, Input, Output ar Memory area 2 contains Retain data: size: 407; Memory area 3 contains Persistent data: size: 828 Build complete 0 errors, 0 warnings: Ready fr Messages - Total 0 error(s), 0 warning(s), 12 me	2 bytes, hi 168 bytes or download isage(s)	, highest used address: 45, largest d Watch 1 () Breakpoints	contiguous mem	ory gap: 8123 bytes (99 %)						

Figure 173 Write Values の選択



③ 所望の変数へ値が書き込まれます。

SampleProject_Cx2500.project* - CODESYS							– 🗆 ×
File Edit View Project Build Onlin	ne Debug Tools Window Help						₹2
🛅 🚅 📓 🚭 🗠 🗠 🐰 🖿 🛍 🗙 🖊	s 🕼 🍪 🌿 🔳 🐄 🦄 🖓 🐘 🛅	- 🕤 🛗 Application [D	evice: PLC Logic] •	Q\$ Q\$ > = % [0	in en en en s	히히罵言 장	
Devices -	4 X ry Manager PLC_PRG	🖬 Device 🏼 🍪 Ma	nTask 👔 POU	FB POU_CtrlA	🗙 🎲 Taski	Task Configuration	Cx2500Ain -
SampleProject_Cx2500	Device_Application_POU_CtrlA						
E 🕞 👔 Device [connected] (CX2500)	Expression	Type	Value	Prepared value	Address	Comment	3
E DLC Logic	@ ValA	LUINT	1000				
Application [stop]	RetainVal0	USINT	100			RETAIN VALUE	
B-CO DUT_SAMPLE	🛊 TimeCnt	UDINT	13422645				
GVL							
Library Manager							
PLC_PRG (PRG)							
POU_CtriA (PRG)				A V			
POU_FB (PRG)	2 TimeCnt 13422645	:= SysTimeCore.SysTi	meGetMs():				^
Task Configuration	3	- oformecorciofor					
All a c and	4						
	5 GVL.g_udVal0	<pre>00 := 100;</pre>					
POU_CHA	6						
	7						
一回 P00_F8	8 @ RetainVal0 100 :=	100;	//Call RETAIN	value			
Persistentivars	9 Persistentvars.P	ersistval0[200] := 200;	//call PERSIST	WT Value			
	11						
	12 • IF ValA 10	> 10 THEN					
Cx2500Da (Cx2500Da)	13 ValA 1	000 := ValA	1000	- 1;			
Cx2500Erain (Cx2500Erain)	↓ □ 14 ELSE						
Revices POUs	15 Vala 1	000 := ValA	1000	+ 500 ;			100 🔞 🗸
	12 PMD TP						
Messages - Total U error(s), U warning(s), 12 messag	e(s)						₩ # X
Build	O error(s) O warning(s)	🟮 7 message(s) 🗙 💥					
Description						Project	Object Position ^
Memory area 1 contains Data, Input, Output and	d Nonsafe data: size: 86016 bytes, highest used	address: 9676, largest contig	Jous memory gap: 7634	0 bytes (88 %)			
Memory area 2 contains Retain data: size: 4072	bytes, highest used address: 1, largest contiguo	us memory gap: 4071 bytes (9	19 %)				
Memory area 3 contains Persistent data: size: 8:	168 bytes, highest used address: 45, largest con	tiguous memory gap: 8123 by	tes (99 %)				
Build complete 0 errors, 0 warnings : Ready for	r download						~
Messages - Total 0 error(s), 0 warning(s), 12 mes	sage(s) 💹 Watch 1 🧑 Breakpoints						
Device user: Anonymous Last bu	ild: 🔿 0 😗 0 Precompile 🗸 😘	STOP	Program loaded	Prog	ram unchanged	Project use	r: (nobody) 🛛 🜍 🛆

Figure 174 値書き込み完了後

9.5.2. <u>値の強制</u>

ここでは、変数の値の強制手順を示します。値を強制した後は 9.5.3 項のように強制を解除しないと通常のア プリケーション動作による変数値に戻らないことに留意して下さい。

① 値を強制したい変数の変数宣言部にて Prepared value 列に所望の値を入力します。

SampleProject_Cx2500.project* - CODESYS							-	o ×
File Edit View Project Build Online	e Debug Tools Window Help							₹2
🖹 🚅 🔲 🚑 🗠 a 🕹 🛍 X 🛤	44 44 44 日 11 11 11 11 11 11 11 11 11 11 11 11 1	- 🕤 🛗 Application (D	evice: PLC Logic] • G	6 刘 🕨 = 🛠 10	∃ 9 <u>∃</u> 4 <u>∃</u> + <u>∃</u> 2	3 호 蒙 글 장		
Devices 👻	4 🗙 Iry Manager 📄 PLC_PRG	Device 🚯 Mair	יTask וס POU_FI	B POU_CtrlA	🗙 🛃 Task1	Task Configuration	i i cx	2500Ain
SampleProject_Cx2500	Device.Application.POU_CtrlA							
E G M Device [connected] (CX2500)	Expression	Type	Value	Prepared value	Address	Comment		B
E BU PLC Logic	de Való	LIL TNT	1000	2000				
Application [stop]	PetainVal	LISINT	1000	2000	-	PETAIN VALUE		
B 🚞 DUT_SAMPLE	TimeCot	LIDINT	13422645			RETAIN VALUE		
- 🥔 GVL		00111	13122013					
Library Manager								
PLC_PRG (PRG)								
POU_Ctria (PRG)								
POU_FB (PRG)	1							
Task Configuration	2 TimeCnt 13422645	:= SysTimeCore.SysTi	meGetMs();					
🖻 😏 🍪 MainTask	3							
@] PLC_PRG	5 GVT a udValo	- 100.						
	6	100,						
🖻 🐨 🍪 Task1	7							
DOU_FB	8 🛛 RetainVal0 100 :=	100;	//Call RETAIN	value				
T PersistentVars	9 PersistentVars.P	ersistVal0 200 := 200;	//Call PERSISTEN	T value				
- 🕞 📺 Cx2500Ain (Cx2500Ain)	10							
- 🗇 📺 Cx2500Cntin (Cx2500Cntin)	11							
- 🐨 🚮 Cx2500Din (Cx2500Din)	E 12 • IF ValA	1000 <2000>	> 10 THE	N				
-G 🕤 Cx2500Do (Cx2500Do)	13 ValA	1000 <2000>	:= ValA	10	00 <2000>	- 1;		
Cx2500Erain (Cx2500Erain)	✓ □ 14 ELSE							
Cevices POUs		1000 <2000>	:= Vala	10	00 <2000>	+ 500 ;		100 🔍
Messages - Total 0 error(s), 0 warning(s), 12 message((s)							• # >
Build	 O error(s) O warning(s) 	😗 7 message(s) 🗙 💥						
Description						Proje	ct Object	Position
Memory area 1 contains Data, Input, Output and	Nonsafe data: size: 86016 bytes, highest used	address: 9676, largest contigu	ous memory gap: 76340	bytes (88 %)				
Memory area 2 contains Retain data: size: 4072 b	vtes, highest used address: 1, largest contiguo	us memory gap: 4071 bytes (9	9 %)	,				
Memory area 3 contains Persistent data: size: 816	58 bytes, highest used address: 45, largest con	tiguous memory gap: 8123 byt	es (99 %)					
Build complete 0 errors, 0 warnings : Ready for o	download	-						
Messages - Total 0 error(s), 0 warning(s), 12 messa	age(s) 📓 Watch 1 🔊 Breakpoints							
Device user: Anonymous Last buil	ld: 🔿 0 🕐 0 Precompile 🧹 🔒	STOP F	Program loaded	Prog	am unchanged	Project	ser: (nobody)	0
								v

Figure 175 値の強制 Prepare value への値の入力



② メニューバーから「Debug」 \rightarrow 「Force Values」を選択して下さい。

SampleProject_Cx2500.project* - CODESYS File Edit View Project Puild Only	D-I	Tools Window Us	la.							_		2
File Edit View Project Build Onlin	Let Det	Start	IP ES		Notesta - O	· **	- c- d 0		а.		۲	2
		Store	Shift+E8	pplication [Device	PLC Logicj •	9 🦋 🕨 💷 🐄 (,= *= <u>~</u> = 4) ~ ma m	14			
	- 1	Single Cycle	Ctrl+Es									
Devices -	4	single Cycle	Ctri+F5	MainTask	POU_FB	POU_CtrlA	🗙 🍰 Task1	Task Config	guration	CX 🕅 CX	2500Ain	1
SampleProject_Cx2500	- 10	New Breakpoint										
B-G Device [connected] (CX2500)	- 5	Edit Breakpoint		pe	Value	Prepared value	Address	Comment				1
PLC Logic		Toggle Breakpoint	F9	NT	1000	2000						Г
=	0	Disable Breakpoint		INT	100		_	RETAIN VALUE				
B DUT_SAMPLE	•	Enable Breakpoint		INT	13422645							
GVL	-		Fre	-								
Library Manager	L,≡	step Over	F10									
	ゆ国	Step Into	F8									
	œ'≣	Step Out	Shift+F10									1
Tack Configuration	+3	Run to Cursor		Core.SysTimeGe	tMs();							1
B € S S MainTack	3	Set Next Statement										
	-	Show Next Statement										
- DU Ctria	~	Show Next Statement		0;								
E-G STask1		Write Values	Ctrl+F7	_								
POU_FB		Force Values	F7		/Call RETAIN V	alue						
T PersistentVars		Unforce Values	Alt+F7	0 := 200; //0	all PERSISTEN	r value						
- 🗇 📆 Cx2500Ain (Cx2500Ain)	1370	Toggle Flow Control Mode										
- 🗇 í Cx2500Cntin (Cx2500Cntin)		Core Dump		•								
🕞 👔 Cx2500Din (Cx2500Din)		North Martin		00>	> 10 THEN	N						
		Display Mode		▶ 2000>	:= ValA	1	100 <2000>	- 1;				
Cx2500Erain (Cx2500Erain)	_	Check Memory for Active Ap	plication	000>	:= ValA	1	100 <2000>	+ 500				
Devices POUs		IS PMD TP				-					100	R
Messages - Total 0 error(s), 0 warning(s), 12 messag	e(s)										• •	>
Build		O error(s) O warnin	g(s) 🜖 7 messa	ge(s) 🗙 💥								
Description									Project	Object	Position	T
Memory area 1 contains Data, Input, Output an	d Nonsafe	data: size: 86016 bytes, highest	used address: 96	76, largest contiguous n	emory gap: 76340 b	oytes (88 %)						
Memory area 2 contains Retain data: size: 4072	bytes, hig	hest used address: 1, largest cor	ntiguous memory g	ap: 4071 bytes (99 %)								
Memory area 3 contains Persistent data: size: 8	168 bytes	, highest used address: 45, larges	t contiguous mem	ory gap: 8123 bytes (99	9 %)							1
Build complete 0 errors, 0 warnings : Ready for	r downloa	d										ľ
Messages - Total 0 error(s), 0 warning(s), 12 mes	sage(s)	🚰 Watch 1 🎒 Breakpoints										
			_	-								-

Figure 176 Force Values の選択

③ 所望の変数へ値が書き込まれ、強制されます。強制中は図のようなアイコンが変数の隣に表示されます。

SampleProject_Cx2500.project* - CODESYS						- 0 ×
File Edit View Project Build Online	Debug Tools Window Help					T 2
🖹 🖨 🔚 🕼 🗠 🖓 🐴	(승 🐴 (슬) 📕 🐄 🦄 🦄 📾 🛅 -	🕤 🔛 Application [Device: Pl	LC Logic] 🔹 👒 🐗 🕨 🔳 帐 🛛	(I 91 41 41 8 + 🛒	= 7/	
Devices 🗸	x y Manager PLC_PRG	🖬 Device 🎲 MainTask	POU_FB POU_Ctrl/	🗙 😒 Task1 🛛 😹 Task	Configuration	Cx2500Ain -
SampleProject_Cx2500	Device.Application.POU_CtrlA					
🖹 🎲 Device [connected] (CX2500)	Expression	Type	Value Prepared value	Address Commer	at .	The second se
E I PLC Logic	(ValA	ULINT (E 2000			Ē
Application [stop]	RetainVal	USINT	100	PETAIN V	ALLIE	
I CONT_SAMPLE	TimeCnt	LIDINT	13422645	DE IGAN		
🧭 GVL		00111				
Library Manager						
PLC_PRG (PRG)						
POU_CtrIA (PRG)						
POU_FB (PRG)	1 2 TimeCot 1949944E	- SustingCore SustingCort				^
Task Configuration	3	- Systimecore.Systimesecr	La () ,			
AlinTask	4					
PLC_PRG	5 GVL.g_udVal0 10	0 := 100;				
POU_CMA	6					
=−G ⊗ Task1	7					
	8 RetainVal0 100 :=	100; //C	all RETAIN value			
Persistentivars	9 PersistentVars.Pe	rsistVal0200 := 200; //Cal	11 PERSISTENT value			
Cx2500Ah (Cx2500Ah)	11					
C C Custoppia (Custoppia)	H 12 IF Valk	> 10 THEN				
C (cuasoopa (cuasoopa)	13 ValA	2000 := ValA6	2000 - 1;			
Cx250050 (Cx250050)						
POUs	15 ValAF	2000 := ValA	2000 + 500 ;			100 🙉 🗸
Messages - Total 0 error(s), 0 warning(s), 12 message(s	s)					→ ∓ ×
Build	• O error(s) • O warning(s)	🕽 7 message(s) 🗙 💥				
Description					Project	Object Position ^
Memory area 1 contains Data, Input, Output and N	Nonsafe data: size: 86016 bytes, highest used a	ddress: 9676, largest contiguous men	nory gap: 76340 bytes (88 %)			
Memory area 2 contains Retain data: size: 4072 by	ytes, highest used address: 1, largest contiguou	is memory gap: 4071 bytes (99 %)				
Memory area 3 contains Persistent data: size: 816	8 bytes, highest used address: 45, largest conti	guous memory gap: 8123 bytes (99 %	a)			
Build complete 0 errors, 0 warnings : Ready for d	lownload					~
Messages - Total 0 error(s), 0 warning(s). 12 messa	ge(s) R Watch 1 Reakpoints					
Device user: Anonymous Last build: 📀 0	• 0 Precompile 🗸 🔒 STOP	Program loaded - Force active	Program unchanged	Project user: (nobody) INS Ln	8 Col 56 Ch 43 🔮 🔇

Figure 177 値の強制完了後



9.5.3. <u>値の強制解除</u>

変数値を強制した場合、値を元に戻すには強制を解除する必要が有ります。値の強制解除をすると、強制され ている変数全ての解除が行われます。ここでは、その手順を示します。

① メニューバーから「Debug」→「Unforce Values」を選択して下さい。

	kaj 2 ▶ =	Start Stop	F5 Shift+F8	pplication (Device: PLC Logic) • 역 👹 🕨 👜 🤻 (도 역 역 역 역 역 위 團 (국 유) 후 (國) (국) 한	
Devices -	4	Single Cycle	Ctn+F5	MainTask 🕜 POU_FB 📄 POU_CtrlA 🗙 🍪 Task1 🥁 Task Configuration 🕋 Cx2500Ain	
SampleProject_Cr2500 Giff Device [connected] (CX2500) Giff Prot Logic Application [stop] Call Connected [CX2500] Ca		New Breakpoint Edit Breakpoint Toggle Breakpoint Disable Breakpoint Enable Breakpoint	F9	Per Value Prepared value Address Comment NT 2000 PT 23 RETAIN VALUE NT 1542245	1
		Step Over Step Into Step Out Run to Cursor Set Next Statement Show Next Statement Write Values Force Values Inforce Values	F10 F8 Shift+F10 Ctrl+F7 F7	Core.SysTimeGetHs(); ;; //Cell ESTAIN velue	
- ⊕ ∰ Cx2500Ain (Cx2500Ain) - ⊕ ∰ Cx2500Cntin (Cx2500Cntin)	12*	Toggle Flow Control Mode Core Dump	AIL+F7	a) - 200, //Cal Publicat Value	
- ♥ PersistentVars - ↔ ∰ Cx2500An (Cx2500An) - ↔ ∰ Cx2500Cntin (Cx2500Cntin) - ↔ ∰ Cx2500D (Cx2500Dn) - ↔ ∰ Cx2500DF (Cx2500De) - ↔ ∰ Cx2500Eroin (Cx2500Eroin)	12*	Toggle Flow Control Mode Core Dump Display Mode Check Memory for Active A	Application	<pre>> > 10 THEN > 10 THEN > = Valk@ = 500 ;</pre>	
↓ T Persistent/Vars ↓ ① Cx2500Ari (Cx2500Ari) ↓ ② ① Cx2500Ari (Cx2500Crit) ↓ ③ ② Cx2500Ari (Cx2500Crit) ↓ ③ ③ Cx2500Ari (Cx2500Crit) ↓ ③ ⑦ Cx2500Ari (Cx2500Crit) ↓ ◎ ③ Cx2500Ari (Cx2500Crit) ↓ ◎ ③ Cx2500Ari (Cx2500Crit) ↓ ◎ ③ Cx2500Ari (Cx2500Crit) ↓ ◎ ○ □ ○ □ ○ □ ○ □ ○ □ ○ □ ○ □ ○ □ ○ □ ○	D.	Toggle Flow Control Mode Core Dump Display Mode Check Memory for Active A	Application	<pre>> 10 THEN > 10 THEN > 10 THEN : = Valk@ = 1; := Valk@ + 500 ; 100</pre>	R
	pe(s)	Toggle Flow Control Mode Core Dump Display Mode Check Memory for Active A	Alter/	<pre>> 10 THES > 10 THES > 10 THES > 12 VALUE 2000 - 1; := VALUE 2000 + 500 ; 100</pre>	#
	2* 2e(s)	Toggle Flow Control Mode Core Dump Display Mode Check Memory for Active A	Autory Application ing(s) 0 7 messa	<pre>> 10 THEN > 10 THEN > 10 THEN > 12 THEN > 12 THEN > 12 THEN > 13 THEN > 14 THEN > 10 THEN ></pre>	\$
T Persetentivers C (2500Ar) C.2500Ar) C.250Ar)	ge(s) nd Nonsafe 2 bytes, hig	Toggle Flow Control Mode Core Dump Display Mode Check Memory for Active A 1 a BNN TB 0 0 error(s) ① 0 warm data: size: 96016 bytes, highe hest used address: 1, largest c	Application ing(s) • 7 messa st used address: 96 ontiguous memory g	10 THESE 10 THESE 11 THESE 12 THESE 13 THESE 14 THESE 15 THESE 16 THESE 17 THESE 18 THESE 19 THESE 10 THESE 10 THESE 11 THESE 12 THESE 13 THESE 14 THESE 10 THESE	a a ion
Thestatentives Construction of the second	ge(s) vd Nonsafe 2 bytes, hig 1168 bytes,	Toggle How Control Mode Core Dump Display Mode Check Memory for Active A 1d aven TP 0 0 entrof(s) 0 0 warm data: size: 86016 bytes, highe hest used address: 1, largest c highest used address: 1, largest c	Application ing(s) • 7 messa st used address: 96 ontiguous memory g est contiguous mem	In the construction of	Q a ion

Figure 178 Unforced Values の選択

② 値が強制されている変数の強制が解除されます。解除と共に強制中のアイコンは消失します。

 SampleProject_Cx2500.project* - CODESYS 									-	
File Edit View Project Build On	line D	ebug loois Window Help	-21.1 (00)							₹ 2
1 🖼 🖬 1 📾 to or % 🌆 🖻 X t	PA 3,5 6	12 14 III III III III III III III III	Application [De	vice: PLC Logic] •	99 99) = % []	고 아프 아프 아프 공동	후 元 품'	24		
Devices	- 4 X	ary Manager PLC_PRG	🕅 Device 🏾 🍪 Main	Task 🔂 POU_F	B POU_CtrlA	🗙 🍪 Task1	Task Conf	iguration	Cx2	(500Ain
SampleProject_Cx2500	• ^	Device.Application.POU_CtrlA	\							
= - Co m Device [connected] (CX2500)		Expression	Туре	Value	Prepared value	Address	Comment			3
= El PLC Logic	- 1	🚸 ValA	ULINT	2000						
Application [stop]	•	RetainVal0	USINT	23			RETAIN VALUE			
OUT_SAMPLE		TimeCnt	UDINT	13422645						
Library Manager										
POU CtriA (PRG)										
POU FB (PRG)		1								
= 🔣 Task Configuration		2 TimeCnt 13422645	:= SysTimeCore.SysTime	neGetMs();						
🖹 😳 🥩 MainTask		3								
			100							
- @ POU_CtrlA		e dvi.g_ddvaio	100,							
🖹 🐨 🍪 Task1		7								
POU_FB		8 💿 RetainVal0 23 :	= 100;	//Call RETAIN	value					
T PersistentVars		9 PersistentVars.F	PersistVal0200 := 200;	//Call PERSISTE	NT value					
- 5 [] Cx2500Ain (Cx2500Ain)		10								
Cx2500Cntin (Cx2500Cntin)		12 TF Vala 20	DO THEN							
Cx2500Din (Cx2500Din)		13 Vala	2000 := ValA	2000 -	1:					
	~	ELSE								
POUs		15 ValA :	2000 := ValA	2000 +	500 ;					100
	()	L 16 REATE TRI								
Messages - Total 0 error(s), 0 warning(s), 12 messa	ge(s)									• 4
Build		O error(s) O warning(s)	O 7 message(s) X X							_
Description								Project	Object	Position
Memory area 1 contains Data, Input, Output a	nd Nonsa	fe data: size: 86016 bytes, highest used	d address: 9676, largest contigue	ous memory gap: 76340	bytes (88 %)					
Memory area 2 contains Retain data: size: 407	2 bytes, ł	highest used address: 1, largest contigue	ous memory gap: 4071 bytes (99	9%)						
Memory area 3 contains Persistent data: size:	8168 byte	es, highest used address: 45, largest cor	ntiguous memory gap: 8123 byte	is (99 %)						
Build complete 0 errors, 0 warnings : Ready f	or downlo	ad								
Messages - Total 0 error(s), 0 warning(s), 12 me	essage(s)	💭 Watch 1 🔊 Breakpoints								
Device user: Anonymous Last build: C	0 😗 0	Precompile 🧹 🚡 STOP	Program loaded	Pro	gram unchanged	Project use	er: (nobody)	INS	Ln 14 Col 5	Ch 5 🔇

Figure 179 変数強制解除完了後



9.6. <u>リセット</u>

アプリケーションをリセットすることで変数などを初期化できます。CODESYS-IDE でできるリセットは 3 種類(Reset Warm・Reset Cold・Reset Origin)あります。各リセット等で初期化されるデータについては Table 73 を参照して下さい。

操作名	通常変数	保持変数	持続変数	アプリケーション	RTC 時刻
動作停止	\bigcirc	\bigcirc	\bigcirc	\bigcirc	\bigcirc
Reset Warm	~	\bigcirc	\bigcirc	\bigcirc	\bigcirc
(ウォームリセット)	^	\bigcirc	\bigcirc	\bigcirc	\bigcirc
Reset Cold	×	×	\bigcirc	\bigcirc	\bigcirc
(コールドリセット)	^	^	\bigcirc	\bigcirc	\bigcirc
Reset Origin	*	~	~	~	\bigcirc
(PLC 初期化)	^	^	^	^	\bigcirc
Download	*	~	\bigcirc	_: 21	\bigcirc
(アプリ書込)	^	^	\bigcirc		\bigcirc
電源再起動	×	\bigcirc	\bigcirc	\bigcirc	\bigcirc

Table 73 各操作における初期化有無 O:初期化されない(値が保持される)、×:初期化される

※21 新しいアプリケーションを書き込むので、初期化ではなくアプリケーションが更新されます。



9.6.1. <u>リセット手順</u>

CODESYS-IDE でおこなえるリセットの手順を示します。ここでは Reset Warm を例に示しますが、Reset Cold や Reset Origin についても手順は同様になります。

なお、Reset Origin については Table 73 の通りリセット実施後アプリケーションも CX2500 から削除されま す。再びアプリケーションを動作させるには、ログイン中の場合はログアウトをおこない、再度ログインをおこ なう必要が有ります。

① デバッグモード中にメニューバーから「Online」 \rightarrow 「Reset Warm」を選択して下さい。

SampleProject_Cx2500.project - CODES	YS											_	o ×
File Edit View Project Build	On	line Debug To	ols Window	Help									₹2
월 🖨 📕 🎒 (M	0; 0;	Login Logout		Alt+F8 Ctrl+F8) (##	Application [Device	e PLC Logic] 🔹 🔇	8 03 > = 43 [[j≣ e≣ e≣ +≣ \$	¢ ∭ 	18./		
Devices 👻 🕂 🗙		Download Multicle Develop			Devic	e 🚱 MainTask	POU_FB	POU_CtrlA	🗙 🥵 Task1]			
SampleProject_Cx2500		Multiple Downloa	3 G										
B-G Device [connected] (CX2500)		Reset Warm				Туре	Value	Prepared value	Address	Comment			1
PLC Logic		Reset Cold				ULINT	2						
= Q Application [stop]		Reset Origin				USINT	100			RETAIN VALUE			
DUT_SAMPLE		Simulation				UDINT	665271						
GVL		Security		,									
Library Manager					-								
	_	Assign Server App	olications on Do	wnload									
		1						·					
- Case Configuration		2 TimeCnt	665271 :=	SysTimeC	ore.Sys	TimeGetMs();							
		3											
B POLI CHA		4											
⊟-⊕ 😪 Task1		S GVL.g_u	dVal0 100	:= 100	;								
B POU FB		2											
T PersistentVars		8 RetainV	a10 100 := 10	0;		//Call RETA	IN value						
- Cx2500Ain (Cx2500Ain)		9 Persiste	entVars.Pers	istVal0 200	:= 20	0; //Call PERSIS	TENT value						
Cx2500Cntin (Cx2500Cntin		10			_								
- 🗇 🕤 Cx2500Din (Cx2500Din)		11											
- G 🗊 Cx2500Do (Cx2500Do)	B	12 💿 IF ValA	2	< 1	10 THEN								
- 🕞 🗊 Cx2500Frgin (Cx2500Frgin		13 Vali	A <u>2</u>		ValA	2] - 1;						
G f Cx2500Pwm (Cx2500Pwm)		15 DLSE	2		Vala	2	1						
(16 END IF	<u>~</u>		Vain	-]+5,						
Sill Davison C DOLla	1	17 RETURN											100
Z Devices 10 Pous													100 200
Messages - Total 0 error(s), 0 warning(s), 12 r	nessa	ge(s)											- 4 ×
Build		- 000	error(s) 💿 0 w	arning(s)	7 mess	age(s) 🗙 💥					_		
Description											Project	Object	Position 2
Size of global data: 8916 bytes													
Managanan - Tatal Diagram(a) Discontine(a)	12	constants		te									
E Pessages - rotar d'error(s), o warning(s),	ild e	Bage(s) Bad Watch	nile 4			har an and a standard	Dece		Destant	(ashed)	1110	1.4.6.14	chi 🗿

Figure 180 Reset Warm の選択



② 確認ウィンドウが表示されるので、「はい(Y)」ボタンを押して下さい。

Converting Convertence Converting Converting Converting Converting Converting Conve	
COUPEYIS	^
C C C2500Pum (Cx2500Pum) C C2500Pum (Cx2500Pum) S S	00 69
Z urenes 1700 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2	- A X
Build • O error(s) • O warning(s) • 7 message(s) × 🔆	,

Figure 181 Reset Warm の実施前確認ウィンドウ

③ Reset Warm により、値がリセットされてアプリケーション動作が停止します。これでリセット完了です。

Devices	- a x	Library Ma	nager	Device 😥 MainTas	k 🕜 POUJ	FB	X 😒 Task1				
= 💮 SampleProject_C	×2500 💌	Device Applica	tion.POU_CtrlA								
😑 😳 🚮 Device (c	onnected] (CX2500)	Expression		Type	Value	Prepared value	Address	Comment			b
PLC Log	c	ValA		ULINT	0						
	lication [stop]	RetainVal0		USINT	100			RETAIN VALUE			
	DUT_SAMPLE	TimeCnt		UDINT	0						
	PLC_PRG (PRG) POU_CtHA (PRG) POU_FB (PRG)					V					
e-19	Task Configuration	1		SusTimeCarMa().							
8-	🖓 😂 MainTask	2 1 11me	cnt 0 := Systimecon	e.SysiimeGetMs();							
	PLC_PRG	4									
		5 💿 GVL.	g_udVal0 0 := 100;								
-	AD DOLLER	e									
T	PareistantVare	7 Deta	1-17-10 100 - 100 -	(/Call DET	ATM INC.						
- 6 F (22	00Ain (Cy2500Ain)	9 Pers	istentVars.PersistVal0 200	:= 200: //Call PERS	AIN VALUE						
-0 m cx25	00Cntin (Cx2500Cntin	10									
-G 🖬 Cx25	00Din (Cx2500Din)	11									
-0 🖬 Cx25	00Do (Cx2500Do)	12 IF V	alA 0 < 10	THEN	_						
- 🕤 🕤 Cx25	00Frqin (Cx2500Frqin	13 0	ValA 0 :=	ValA 0	- 1;						
- 🖓 🚮 Cx25	00Pwm (Cx2500Pwm)	15 0	Vala 0 :=	Vala 0	+ 5 :						
<	>	16 END	IF								
POUs		17 🔘 <u>RETU</u>	RN								100 🔞 .
Manager Tabel Carro	(-) 0										- 0. 1
messages - rotal o error	(s), o warning(s), 12 me	essage(s)									• • •
Build		• []	U error(s) 😗 U warning(s) 😈	/ message(s)					_	_	
Description									Project	Object	Position
Size of global data:	3916 bytes										1

Figure 182 Warm Reset 後の POU



9.7. <u>ブレークポイント</u>

ブレークポイントは、デバッグモード中にプログラムを任意の位置で強制的に停止させる機能です。プログ ラムの途中でユーザーが自由に停止できるため、処理中の変数値変化等を詳細に確認することができます。ブ レークポイントは最大4つまで同時に設定(有効化)可能です。なお、デバッグモードを終了(ログアウト)する と、設定したブレークポイントは次のログイン時に無効化されること(次のログインまでのアプリケーションを 変更していた場合はブレークポイント削除)に留意して下さい。

CODESYS で利用できるブレークポイントには下記の2種類あります。それぞれの設定方法については後述 します。

名称	摘要	設定方法(参照先)
通常ブレークポイント	条件問わず、設定した位置の処理を行うときに毎回停止する。	9.7.1 項
条件付きブレークポイント	通常のブレークポイントと異なり、ユーザーが設定した特定条	9.7.2 項
	件を満たす場合のみ停止する。	

Table 74 ブレークポイント種類

Table 75 ブレークポイントステータス

アイコン	状態
	ブレークポイント設定可能
0	ブレークポイント設定中(無効状態)
٠	通常ブレークポイント設定中(有効状態)
0	条件付きブレークポイント設定中(有効状態)
•	ブレークポイント停止中



9.7.1. ブレークポイントの設定(通常)

通常のブレークポイントの設定手順を下記に示します。

 デバッグモードにして、ブレークポイントを設定したい行を選択し右クリックを押して下さい。選択された 行は図のように黄色に変わります。

Construction Percent (PC_PR6 ×) PC_PR6 ×) PC_PR6 ×) PC_PR6 ×) Percent (PC_PR6 ×) <th>e Edit View Project Build ☞ 🖬 🚭 ∽ ా 🐰 🛍 💼</th> <th>Online Debug Tools Window Help × 🏘 🍪 🍓 🍓 📕 🐄 🆄 🆄 ا📾 </th> <th>🖫 - 📑 🛗 Application [[</th> <th>Device: PLC Logic] •</th> <th>oş oğ → = ≪ ((</th> <th>I GI CI +I S</th> <th>(1) (1) (1) (1) (1) (1) (1) (1) (1) (1)</th> <th></th> <th></th>	e Edit View Project Build ☞ 🖬 🚭 ∽ ా 🐰 🛍 💼	Online Debug Tools Window Help × 🏘 🍪 🍓 🍓 📕 🐄 🆄 🆄 ا📾	🖫 - 📑 🛗 Application [[Device: PLC Logic] •	oş oğ → = ≪ ((I GI CI +I S	(1) (1) (1) (1) (1) (1) (1) (1) (1) (1)				
Average Product (2:200) Overcappendix (2:200) Overcappendix (2:200) • Complexity (2:200) <t< th=""><th>ces 👻</th><th># X M Device PIC_PRG X</th><th>🛐 POU_CtrlA 🛛 🚷 Main</th><th>Task</th><th></th><th></th><th></th><th></th><th></th></t<>	ces 👻	# X M Device PIC_PRG X	🛐 POU_CtrlA 🛛 🚷 Main	Task							
Image: Specific S	SampleProject_Cx2500 C Classical Action (Compared Compared Comp	Device.Application.PLC_PRG									
	Device [connected] (CX2500)	Expression	Туре	Value	Prepared value	Address	Comment		^		
Application (1509) isk UNT 0 Image: Constraint of the constraint	H PLC Logic	ucVal1	USINT	10							
Origination Origination Origination Declaration of ToOL* 0 <t< td=""><td></td><td>Idx</td><td>UINT</td><td>0</td><td></td><td></td><td></td><td></td><td></td></t<>		Idx	UINT	0							
Unit User Val USER 0 Declaration of 'UNI' IP (C_pRG) IP (C_pRG) </td <td>COLONAL COLOR</td> <td>bFlg</td> <td>BOOL</td> <td>FALSE</td> <td></td> <td></td> <td>Declaration of "BOOL"</td> <td></td> <td></td>	COLONAL COLOR	bFlg	BOOL	FALSE			Declaration of "BOOL"				
Uble Findspie Well REAL Pedaration of 'REAL' Well REAL Pedaration of 'REAL' Pedaration of 'REAL' Well Real Pedaration of 'REAL' Pedaration of 'REAL' Well Real Pedaration of 'REAL' Pedaration of 'REAL' Well Pedaration of 'REAL' Pedaration of 'REAL' Well Pedaration of 'REAL' Well Pedaration of 'REAL' Well Well Pedaration of 'REAL' Well Well Pedaration of 'REAL' Well Well Well Well Well Well Well Well Well Well Well Well Well Well Well Well	Git Ubran Manager	ucVal0	USINT	0			Declaration of "UINT"				
1 1 //Arrsy/ 1 000000000000000000000000000000000000		rVal	REAL	1			Declaration of "REAL"		~		
Build O terror(s) O terror(s)<td>⊕ 1.0.7.966 ⊕ 1.0.7.95 ⊕ 1.0.7.95 ⊕ 1.0.7.95 ⊕ 1.0.7.95 ⊕ 1.0.7.95 ⊕ 1.0.7.95 ⊕ 1.0.2004 ⊕ 0.20004 ⊕ 0.20004 ⊕ 0.200004 ⊕ 0.200004 ⊕ 0.200004 ⊕ 0.200000 ⊕ 0.200000 ⊕ 0.200000 ⊕ 0.200000 ⊕ 0.200000 ⊕ 0.200000 ⊕ 0.200000 ⊕ 0.200000 ⊕ 0.2000000 ⊕ 0.2000000</td><td>6 ZBD_FCR 0 etSmp_fork 10 etSmp_fork 12 etSmp_fork 13 etSmp_fork 14 etSmp_fork 15 etSmp_fork 16 etSmp_fork 17 etSmp_fork 18 etSmp_fork 19 etSmp_fork 10 etSmp_fork 11 etSmp_fork 12 etSmp_fork 13 etSmp_fork 14 etSmp_fork 15 etSmp_fork 16 etSmp_fork 17 etSmp_fork 18 etSmp_fork 19 etSmp_fork 10 etSmp_fork 11 etSmp_fork 12 etSmp_fork 13 etSmp_fork 14 etSmp_fork 15 etSmp_fork 16 etSmp_fork</td><td>= 0; = 11; = 22; = 0; = 10;</td><td>_</td><td></td><td>_</td><td></td><td></td><td>100</td>	⊕ 1.0.7.966 ⊕ 1.0.7.95 ⊕ 1.0.7.95 ⊕ 1.0.7.95 ⊕ 1.0.7.95 ⊕ 1.0.7.95 ⊕ 1.0.7.95 ⊕ 1.0.2004 ⊕ 0.20004 ⊕ 0.20004 ⊕ 0.200004 ⊕ 0.200004 ⊕ 0.200004 ⊕ 0.200000 ⊕ 0.200000 ⊕ 0.200000 ⊕ 0.200000 ⊕ 0.200000 ⊕ 0.200000 ⊕ 0.200000 ⊕ 0.200000 ⊕ 0.2000000 ⊕ 0.2000000	6 ZBD_FCR 0 etSmp_fork 10 etSmp_fork 12 etSmp_fork 13 etSmp_fork 14 etSmp_fork 15 etSmp_fork 16 etSmp_fork 17 etSmp_fork 18 etSmp_fork 19 etSmp_fork 10 etSmp_fork 11 etSmp_fork 12 etSmp_fork 13 etSmp_fork 14 etSmp_fork 15 etSmp_fork 16 etSmp_fork 17 etSmp_fork 18 etSmp_fork 19 etSmp_fork 10 etSmp_fork 11 etSmp_fork 12 etSmp_fork 13 etSmp_fork 14 etSmp_fork 15 etSmp_fork 16 etSmp_fork	= 0; = 11; = 22; = 0; = 10;	_		_			100		
Dottion Image: Solid State		Presseges - rotal certor(s), o warming(s), y	Messages - Total 0 error(s), 0 warning(s), 7 message(s)								
Description Project Object Object Object Description Object Objec		Dono Lui	• ven	sar(a) 🗣 a warning(s) v message(s)	n					
Memory area 3 contains Persistent data: size: 3168 bytes, highest used address: 45, largest contiguous memory gap: 8123 bytes (99 %) Build semplate: a group of usering and a group in the size of		Description					Project	Object	Position		
build complete o entrings - Ready for domination		Memory area 3 contains Persistent dat Build complete 0 errors, 0 warnings :	a: size: 8168 bytes, highest use Ready for download	d address: 45, largest (contiguous memory gap: 81	23 bytes (99 %)					
Im Messages - Total 0 error(s). 0 warning(s). 7 message(s)		Messages - Total 0 error(s), 0 warning(s), 7 message(s) Watch 1	Breakpoints							

Figure 183 ブレークポイントを設定したい行の選択

② 表示されるコンテキストメニューから「Toggle Breakpoint」を選択して下さい。

- ₽ X	PLC_PRG X POU_CtriA	ainTask 🍸	T Cx2500Din T De	evice R Cx2500D	o 🗃 Visua	ization		_
SampleProject_Cx2500	Device.Application.PLC_PRG							
Device [connected] (CX2500)	Expression	Type	Value	Prepared value	Address	Comment		
E-O Application [ston]	ucVal1	USINT	10					
	🛊 Idx	UINT	11					
- GN	bFlg	BOOL	FALSE			Declaration of "BOOL"		
Ubrary Manager	ucVal0	USINT	7			Declaration of "UINT"		
	da ri/al	DEVI	1	A V		Declaration of "REAL"		
- ① POU/F9 (PKG) = ② Task Configuration = ③ ③ ManTask - ① POU_CYA = ② DOU = ② POU_F9 = ③ POU_F9 = ④ POU_F9 = Ø POU_F9	<pre> //Struct //Struct //Struct //Struct //Struct //Struct //Struct //Struct //Struct(Ext) //Struct(Ext) //Struct(Ext) // stSmpl.uckemArray(0[1] //Struct(Ext) // stSmpl.uckemArray(1] //Struct(Ext) // stSmpl.uckemArray(1] //Struct(Ext) // stSmpl.uckemArray(1] // // stSmpl.uckemArray(1] // // stSmpl.uckemArray(1] // // // stSmpl.uckemArray(1] //</pre>	11; 22; 11; 22; 11; 22; 2;	Cut Copy Paste Delete Select All Advanced Input Assistant	, ,				
- 🕞 🛐 Cx2500Cntin (Cx2500Cntin)	<pre>22</pre>		New Breakpoint					
- 😳 📆 Cx2500Din (Cx2500Din)	23 stSmpl.ulMember0 10		loggle Breakpoint					100
🕞 🛗 Cx2500Do (Cx2500Do)	Messages - Total 0 error(s), 0 warning(s), 0 message(s	a +1	Run to Cursor					-
😳 🕤 Cx2500Frqin (Cx2500Frqin)		- 3	Set Next Statement		¥			
🔂 🚹 Cx2500Pwm (Cx2500Pwm)		-	Unforce All Values of 'De	wise Application!	\$			
	Description		oniorce Air values of De	wiceApplication	Project	Object	Position	
			Display Mode	,				

Figure 184 Toggle Breakpointの選択



③ アイコンがブレークポイント設定中(有効状態)に変わると設定完了です。

		nou cela Valo state	Test					
E SamoleProtect (x250)	Device Application PLC_PRG X	POU_CHA Ss Man	l dsk.					
	Expression • ucrial: • idx • brig • ucrial: • rval • rval • ucary idade : - • TO 10 • ucary idade : - • • TO 10 • ucary idade : - • • TO 10 • ucary idade : - • • TO 10 • • Ucary idade : - • • • • • • • • • • • • • • • • • •	Type USINT UNIT BOOL USINT REAL DO 1 = 104 0 + 27 0 + 27 0 = 107 0 = 107 0 = 107 0 = 107	Value 10 0 24450 0 1	Prepared value	Address	Comment Declaration of "BOOL" Declaration of "UNT" Declaration of "RBAL"		
	Messages - Total 0 error(s), 0 warning(s), 7 mes	sage(s)						- 4 ×
	Build	- O err	or(s) 🕐 0 warning	(s) 🟮 7 message(s) 🗙	₩			
	Description					Projec	t Object	Position ^
	Memory area 3 contains Persistent data: size	ze: 8168 bytes, highest used	address: 45, largest	contiguous memory gap: 81	23 bytes (99 %)			

Figure 185 通常ブレークポイント設定完了後



CCOT-24-016 Rev.1 9.7.2. ブレークポイントの設定(条件付き)

ここでは、条件付きブレークポイントの設定手順について示します。

① 条件付きブレークポイントを設定したい行を選択し、右クリックして下さい。

• #)		POU_CtrlA Se Main	ıTask				
SampleProject_Cx2500	Device_Application_PLC_PRG						
(CX2500) Device [connected] (CX2500) 후 ബ PLC Logic	Expression	Туре	Value	Prepared value	Address	Comment	
Oplication [stop] DUT_SAMPLE OVL	forest forest	UINT BOOL	0 FALSE			Declaration of "BOOL"	
Library Manager International PLC_PRG (PRG)	vcvalu vrval	REAL	0			Declaration of "REAL"	
- ④ PkanTask ● RC_PRG ● PLC_PRG ● PLC_PRG	>///////////////////////////////////	0 0 1 = 117 1 = 227 0 0 0 1 = 117 1 = 107 1 = 107 0 0 0 0 1 = 107 1 = 107 0 1 = 107 1 = 107 1 = 107 0 0 0 0 0 0 0 0 0 0 0 0 0					100
	Messages - Total 0 error(s), 0 warning(s), 7 me	ssage(s)			~		-
	Description	• Oen	ror(s) v 0 warning(s) v message(s)	~	Project	Object Posi

Figure 186 ブレークポイントを設定したい行の選択

② 表示されるコンテキストメニューから「New Breakpoint...」を選択して下さい。

SampleProject_Cx2500.project* - CODESYS File Edit View Project Build O	; nline Debug Tools Window Help 桷 华。 🎂 🏡 川 🍿 🦄 🦄 🛅 🏭 - 🔓 崔	Application [Dev	rice: PLC Logic]	- • • • • • • • • • • • • • • • • • • •]] 두글 손글 +길 {	00 0 ∭ ₩ ™	- 0 >	<
Devices 👻 🗭		rlA 🏾 🚷 MainTa	sk					•
SampleProject_Cx2500	Device.Application.PLC_PRG							
B- 😳 🕤 Device [connected] (CX2500)	Expression	Type	Value	Prepared value	Address	Comment	^	Ŀ
PLC Logic	a ucVal1	USINT	10					
Application [stop]	/ Idx	UINT	0					
DUT_SAMPLE	bFlg	BOOL	FALSE			Declaration of "BOOL"		
GVL	/ ucVal0	USINT	0			Declaration of "UINT"		
Library Manager	/ rVal	REAL	1			Declaration of "REAL"		
PLC_PRG (PRG)	· · · · ·	nunu	•	A 🔻			*	_
G QU_FB (PRG) G Trace_Manifask T PersistentVars G QU_FB (PRG) G QU_FB (PRG)	<pre>7 (* Access *) 9 //Struct 10 * Stap0.uldenber() * * is 11 * stap0.uldenber() 0 12 * stap0.uldenber() 0 13 * stap0.ucdenbray0()] 0 := ; 14 * * * * * * * * * * * * * * * * * * *</pre>	:= 10; 1; 2; := 10; 1; 2; := 10;	20 × 10 m v	Cut Copy Paste Delete Select All Advanced Inout Assistant			100	
	Messages - Total 0 error(s), 0 warning(s), 7 message(s)			input Assistanta			▼ ₽	×
	Build	- O error	(s) 🕐 📶	New Breakpoint				
	Description Memory area 3 contains Persistent data: size: 8166 Build complete 0 errors, 0 warnings : Ready for d	l bytes, highest used a ownload	ddress: ₽	Toggle Breakpoint Run to Cursor Set Next Statement	Application'	Project	Object Position	^
<	> Messages - Total 0 error(s), 0 warning(s), 7 message	(s) 💭 Watch 1 🗐	Breakpo	omorce All values of 'Device.	Application	_		
Device user: Anonymous	st build 🔿 0 🕐 0 Precompile 🥒 Stop	Prog	ramioa	Display Mode	•	Project user:	(nohody)	0

Figure 187 New Breakpoint の選択



③ 「Breakpoint Properties」ウィンドウが表示されるので、「Condition」タブを選択してブレークポイント停止条件を設定します。条件を設定したら「OK」ボタンを押して下さい。なお、ここでは例としてブレークポイントを設定する行に5回入った時に停止する条件を設定しています。

Breakpoint Properties	\times
Condition Location Execution Point Settings Tasks Break only when the breakpoint is hit in one of the following tasks	
MainTask Task1	
Hit Count Break when the hit count is equal to 5	
Condition Break when true	
Enable breakpoint immediately OK Cancel	

Figure 188 Breakpoint Properties ウィンドウ Condition タブ

設定項目	摘要
Tasks	チェックを入れたタスクが全て1度以上実行されたのちにブレークポイントで停止す
	る。
Hit Count	タブで選択した回数(条件)分指定行に入った時にブレークポイントで停止する。
Condition	「Break when true」にチェックを入れると、その右隣の欄に入力した変数値が trueの
	時にブレークポイントで停止する。
Enable breakpoint immediately	チェックを入れるとブレークポイントが設定後即有効化される。

Table 76 Breakpoint Properties ウィンドウ Condition タブの設定項目



④ 図のようにブレークポイントが設定されれば完了です。



Figure 189 条件付きブレークポイント設定完了後



9.7.3. ブレークポイントの無効化設定

ここでは設定して有効化状態(指定位置でプログラムが停止する)のブレークポイントを無効化(指定位置でプ ログラムは停止しない)する手順を示します。

① 無効化したいブレークポイントがある行を選択し右クリックして下さい。

Edit View Project Build Or	nline Debug Tools Window Help 🚧 🎲 💑 🏂 🗍 🖷 🐄 🎢 🆄 📾	🖸 🔛 Application [E	levice: PLC Logic]	og og → = ≪10] ⊊∃ ¢∃ +I §	종 이 第 국 장		`
s v 4 :	X Device PLC_PRG X	POU_CtrlA 🛛 🚱 Main	Task					
SampleProject_Cx2500	Device_Application.PLC_PRG							
Device [connected] (CX2500)	Expression	Type	Value	Prepared value	Address	Comment		^
Real PLC Logic	🔹 ucVal1	USINT	10					- 1
= 🚫 Application [stop]	ldx	LIINT	0					-
DUT_SAMPLE	bElg	ROOL	FALSE			Declaration of "BOOL"		
- 🙆 GVL	🔹 ucVal0	USINT	0			Declaration of "UINT"		
Library Manager	e rVal	REAL	1			Declaration of "REAL"		
• ¶ Tac_VanTek • ₽ resistentivas • € • € • C.2500.nh (C.2500.hr) • € • C.2500.nh (C.2500.hr) • € • C.2500.nh (C.2500.hr) • € • C.2500.hr (C.2500.hr)	10 #timp0:ucbtesber0 1 11 stimp0:ucbtesber0 1 12 stimp0:ucbtesber0 1 13 stimp0:ucbtesber0 1 14 stimp0:ucbtesber0 1 15 stimp0:ucbtesber0 1 16 //Struct(Stt) 1 17 stimp1:ucbtesber0 1 18 stimp1:ubtesber1 1 19 stimp1:ubtesber1 1 Massaee: Tubt0 Purefd 0 1 1	0; 0; 0; 0; 0; 0; 0; 1= 10; 1=						100
	Messages - Total U error(s), U warning(s), 7 me	essage(s)			×			•
	Build	• 0 0 en	or(s) 🐨 0 warning	s) v message(s)	*			
	Description					Project	Object P	Positio
	Memory area 3 contains Persistent data:	size: 8168 bytes, highest used	address: 45, largest	contiguous memory gap: 81	23 bytes (99 %)			
	 Henory area o containa Perasterit data. I 							
	Build complete 0 errors, 0 warnings : Re	ady for download						

Figure 190 ブレークポイントを無効化したい行の選択

② 表示されるコンテキストメニューから「Disable Breakpoint」を選択して下さい。

ces 🗸 🗸 🗸 🗸		OU_CtriA	MainTask					
SampleProject_Cx2500	Device Application.PLC_PRG							
G Device [connected] (CX2500) G Of PLC Logic G Application [stop]	Expression ucVal1		Type USINT	Value 10	Prepared value	Address	Comment	^
DUT_SAMPLE GVL GVL Ubrary Manager			UINT BOOL USINT REAL	FALSE 0			Declaration of "BOOL" Declaration of "UINT" Declaration of "REAL"	
 평 Task Configuration 중 중 MainTask 에 Pic_C PRG 에 Pic_C PRG 에 PicU_C PRG 에 PicU_C PIA 에 PicU_C PIA 에 PicU_FB 에 Trace_MainTask 	<pre></pre>	:= Idx	0 * 2; Cut					
	<pre>11 # stSmp0.ulMember() 12 # stSmp0.ulMember() 13 # stSmp0.ucMemArray(0[1] 14 15 16 //Struct(Ext) 17 # stSmp1.ulMember[] 18 # stSmp1.ulMember[] 19 # stSmp1.ulMember[] 10 # stSmp1.ulMember[] 11 # stSmp1.ulMember[] 12 # stSmp1.ulMember[] 13 # stSmp1.ulMember[] 14 # stSmp1.ulMember[] 15 # stSmp1.ulMember[] 16 # stSmp1.ulMember[] 17 # stSmp1.ulMember[] 18 # stSmp1.ulMember[] 19 # stSmp1.ulMember[] 19 # stSmp1.ulMember[] 10 # stSmp1.ulMember[] 10 # stSmp1.ulMember[] 11 # stSmp1.ulMember[] 12 # stSmp1.ulMember[] 13 # stSmp1.ulMember[] 14 # stSmp1.ulMember[] 15 # stSmp1.ulMember[] 15 # stSmp1.ulMember[] 16 # stSmp1.ulMember[] 17 # stSmp1.ulMember[] 18 # stSmp1.ulMember[] 19 # stSmp1.ulMember[] 19 # stSmp1.ulMember[] 10 # stSmp1.ulMember[] 10 # stSmp1.ulMember[] 10 # stSmp1.ulMember[] 11 # stSmp1.ulMember[] 12 # stSmp1.ulMember[] 14 # stSmp1.ulMember[] 15 # stSmp1.</pre>	● ■ □ □ ■ □	Paste Delete Select All Advanced Input Assistant		•			100
	<pre>11 # st3mp0.illember0; 12 # st3mp0.uclema.rray0[0][8] 13 # st3mp0.uclema.rray0[1][8] 14 15 //Struct(Ext) 16 # st3mp1.uclember[8] # 0; 18 # st3mp1.ullember0; 19 # st3mp1.ullember0; 19 # st3mp1.ullember0; 10 # st3mp1.ullember0</pre>	2 1 1 1 1 1 1 1 1 1 1 1 1 1	Paste Delete Select All Advanced Input Assistant Edit Breakpoint		•			100
■ Tensentifiers ● G: 20000ch (Cx300kh) ● G: 22000ch (Cx300kh) ● G: Cx3000ch (Cx300kh) ● G: Cx3000ch (Cx300kh) ● G: Cx3000ch (Cx300kh) ● G: Cx3000ch (Cx300kh)	11 stSmp0.ulthemberQue 12 stSmp0.ulthemberQue 13 stSmp0.ulthemberQue 14 stSmp0.ulthemberQue 15 stSmp0.ulthemberQue 16 //Struct(Ext) 17 stSmp1.ulthemberQue 18 stSmp1.ulthemberQue 19 stSmp1.ulthemberQue 10 stSmp1.ulthemberQue 11 stSmp1.ulthemberQue 12 stSmp1.ulthemberQue 13 stSmp1.ulthemberQue 14 stSmp1.ulthemberQue 15 stSmp1.ulthemberQue 16 stSmp1.ulthemberQue 16 stSmp1.ulthemberQue 17 stSmp2.ulthemberQue 18 stSmp2.ulthemberQue 17 stSmp2.ulthemberQue <td>2 := × ; 3 3 3 9 €</td> <td>Paste Delete Select All Advanced Input Assistant Edit Breakpoint Togele Breakpoint</td> <td></td> <td>• ge(s) ></td> <td>*</td> <td></td> <td>100 g</td>	2 := × ; 3 3 3 9 €	Paste Delete Select All Advanced Input Assistant Edit Breakpoint Togele Breakpoint		• ge(s) >	*		100 g

Figure 191 Disable Breakpoint の選択



③ 図のようにブレークポイントのアイコンが無効化状態になれば完了です。なお、無効化状態から有効化状態 に変えたいときは、同様の手順で行います。ただし、手順②では「Enable Breakpoint」を選択して下さい。



Figure 192 ブレークポイント無効化完了後



9.7.4. ブレークポイントの設定削除

設定したブレークポイントを削除する手順を下記に示します。ブレークポイントの無効化(9.7.3 項)と異なり、 ブレークポイントを削除するとこれまでのブレークポイント到達数などの情報も削除されてしまうことに留意 して下さい。

① 削除したいブレークポイントがある行を選択した状態で右クリックして下さい。

1) 📽 🖬 🕼 ko 🗠 i 🗞 🛍 🛍	Online Debug Tools Window H × [44] 상품 4월 상품 제 1월	Help	Device: PLC Logic]	cs cg → = ≪ []] 6]] 6]] +]] (0 ¢ 開 市	r 10-		*
- CampleDroject (V2500	Device Application DI C. DDC		1 dok						
	Expression • uc/all • uc/all • ldx • bFig • uc/all • yr/al	Type USINT UINT BOOL USINT REAL	Value 10 0 FALSE 0 1	Prepared value	Address	Comment Declaration of Declaration of Declaration of	of "BOOL" of "UINT" of "REAL"		^))
- ■ POU_CtriA (PRG) - ● POU_CtriA (PRG) - ● POU_F (PRG) - ● ■ Task Configuration = - ● POU_F (PRG) - ● ■ PLC_PRG	↓ → //Array → FOR Idx 0 := 0 ↓ → Ary[Idx] END_FOR	TO 10 DO] 0 := Idx 0 * 2;		A V					
관 POU_CHA 	7 (* Access *) 10 ************************************	0 1= 07 0 (0)[0] := 11; 0(1)[0] := 22; 0 := 0; 0 := 10; 100[0] := 10;							100
응 (C.2007) (응 (C.2007) (응 (C.2007) (유명) (C.2007) (유명) (C.2007) (유명) (C.2007) (유명) (C.2007) (유명) (C.2007) (유명) (C.2007) (유명) (C.2007) (유명) (C.2007) (유명) (C.2007) (1 4 Access + 1 1 4 Stateget underseeft 12 4 Stateget underseeft 13 4 Stape underseeft 14 4 Stape underseeft 15 4 Stape underseeft 16 //Stape underseeft 17 4 Stape underseeft 18 17 4 19 15 10 10 4 Stape underseeft 10 5 4 11 5 5 12 4 5 13 4 5 14 //State (fit) 10 15 4 5 16 10 10 17 4 5 18 10 10 19 10 10 10 4 10 10 10 10 10 10 10 10 10								100
응 (C.2000+ (전) (C.2000+ (전) (C.2000+ (전) (C.2000+ (C.2000+ (C.2000+ (C.2000+ (C.2000+ (C.2000+ (C.2000+ (C.2000+ (C.2000+ (C.2000+ (C.2000+ (C.2000+ (C.2000+)) (C.2000+) (C.	an) (1 <u>Access</u>) (1) (1) (2) (2) (2) (2) (2) (2) (3) (3) (4) (4) (4) (4) (4) (4) (4) (4	2 (m 10) 1 = 107 1 = 227 2 = 227 2 = 107 2 = 007 2	or(s) 🖲 0 warning(s) • 7 message(s) ×	×				100 🙊
응 (0.250%) (전 2004) (전 2004) (전 2004) (전 2004) (전 2004) (전 2006) (전 2006) (전 2006) (전 2006) (전 2006) (전 2006) (전 2007e) (전 2007e) (전 2007e) (전 2007e) (전 2007e) (전 2007e) (전 2007e)	an) an) an) an) an) an) an) an)	(1) (1) (2) (1) (1) (1) (1) (1) (1) (1) (1) (1) (1	or(s) 💿 0 warning(d address: 45, largest (s) 0 7 message(s) X	≹ 23 bytes (99 %)		Project	Object	100 🙉 🗸 Versition d'
● ● ○ ○ Cata ● ● ○ ▼ Taki ● ○ ○ Taki ● ○ ○ Cata ● ○ ○ ○ Cata ● ○ ○ ○ Cata ● ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○	a (* Access *) b (* Access *) b * attraps_uckeesed c * attraps_uckeesed d //Stract(Ext) c Messages-Total 0 error(s), 0 warmsg Build Decorption Messages -Total 0 error(s), 0 warmsg	Immedia 1 = 107 0(0)[3] := 11; 0(0)[3] := 22; 0(1)[3] := 22; 1 = 107 0(1)[3] := 22; 1 = 107 0(1)[3] := 503; - 0 0 erit 1 (dds: ster: 8569 bytes, highest using gs; Ready for download - 0 0 erit 1 (dds: ster: 8569 bytes, highest using gs; Ready for download - 0 0 erit	or(s) ♥ 0 warning(d address: 45, largest r Ø Breakpoints	a) 🌒 7 messagela) 🗙	¥ 23 bytes (99 %)		Project	Object	100 🙉

② 表示されるコンテキストメニューから「Toggle Breakpoint」を選択して下さい。

SampleProject_Cx2500.project* - CODESYS							_	0	×
File Edit View Project Build Onlin	ne Debug Tools Window Help								7
🖹 📽 🔛 🚳 🗠 🗠 🕹 🛅 🛍 🗶 👭	仏 西 仏 州 州 州 州 岡 笹・白 笹	Application [Dev	ice: PLC Logic] 🔹 👒	💖 🕨 = 🔧 (ji fi di ti 8	이 國 로 장			
Devices	Device PLC_PRG X POU_Ctr	IA 🚯 MainTa:	sk 🛛						
= 3 SampleProject_Cx2500	Device Application.PLC_PRG								
E 😳 👔 Device [connected] (CX2500)	Expression	Type	Value	Prepared value	Address	Comment		^	b
B I PLC Logic	(a) urVal1	USINT	10						
= 🔘 Application [stop]	a Idv	LINT	10					_	-
B C DUT_SAMPLE	bEla	BOOL	FALSE			Declaration of "BOOL"			
- 🥔 GVL	a urVal0	LISTNT	0			Declaration of "LINT"			
Library Manager	e rVal	DEAL	1			Declaration of "REAL"			
PLC_PRG (PRG)			-	~				×	-
POU_Ctria (PRG)	1								^
POU_FB (PRG)	2 //Array								
Task Configuration		ix 0 * 2:							
B-G 😒 MainTask	S END FOR								
PLC_PRG	6 -								
POU_CHA	7								
E-test Task1	8 (* Access *)								
···현의 POU_FB	9 //Struct	3 Cut		1					
Trace_MainTask	<pre>10 stSmp0.ucMember0 0 := 0;</pre>	🗈 Сору							
PersistentVars	12 stSmp0.ulMember0 0	B Paste							
Cx2500Ain (Cx2500Ain)	13 astSmp0.ucMemArray0[1] 0 := 2	× Delete							
Cx2500Cntin (Cx2500Cntin)	14								
Cx2500Din (Cx2500Din)	15	Select All							
Cx2500Do (Cx2500Do)	16 //Struct(Ext)	Advanced		•					
Cx2500Frqin (Cx2500Frqin)	<pre>17</pre>								
🕄 🄟 Cx2500Pwm (Cx2500Pwm)	18 stSmpl.ulMemberl 0	요출 Input Assista	nt					100	2 .
	Manager Tabl () and () () () () ()	Uh Edit Breakpoi	nt						1
	Messages - Total 0 error(s), 0 warning(s), 7 message(s)	Toggle Break	point		~			• •	^
	Build	O Disable Break	point		*			_	
	Description					Proje	ct Object	Position	^
	Memory area 3 contains Persistent data: size: 8168	*.≡ Run to Curso	r	gap: 81	23 bytes (99 %)				1
	Build complete 0 errors, 0 warnings : Ready for do	Set Next State	ement						
< >	Messages - Total 0 error(s), 0 warning(s), 7 messagel	Unforce All V	alues of 'Device.Applica	tion'					~

Figure 194 Toggle Breakpoint の選択



③ 図のようにブレークポイントアイコンがブレークポイント設定可能状態になっていれば、ブレークポイント の削除は完了です。

› Edit View Project Build Onlin 参 🖬 🎒 い つ 券 🗈 💼 📉 🏘	ne Debug Tools Window Help 🕯 🏠 🐴 🏠 📕 📬 🎢 🎢 🎼 🛅 - 🖸	ິງ" 🏥 Application [Dev	ice: PLC Logic] 🝷	o; o; + = ≪ [(] 두] 스] +] 등	(*) (종(국) (*)			•
res 🗸 🕂 🗙	Device PLC_PRG X	DU_CtrlA 🛛 😸 MainTa	sk						
3 SampleProject_Cx2500	Device.Application.PLC_PRG								
🗐 🐨 👔 Device [connected] (CX2500)	Expression	Type	Value	Prepared value	Address	Comment			^
PLC Logic	expression • unital	type	value	Frepared value	Address	comment			
Application [stop]		USINI	10						
Gin Dut_sample		UINI	GALSE			Dedatation of "POO			- 1
🎑 GVL	unital	BOOL	PALSE			Declaration of BOO	-		- 1
- 前 Library Manager		USINI	0			Declaration of UIN	-		
PLC_PRG (PRG)	🖗 rvai	REAL	1			Declaration of REAL			~
- ④ POU_CHA - ④ ♥ Takt - ④ PTact_HanTak - ● ① Cx2500An (Cx2500An) - ○ ① Cx2500An (Cx2500An) - ○ ① Cx2500Cntin (Cx2500Cnin) - ○ ② Cx2500Cn (Cx2500Cn) - ○ ③ Cx2500Cn (Cx2500Cn) - ○ ⑤ Cx2500Cn (Cx2500Cn) - ○ ⑥ Cx2500Cn (Cx2500Cn) - ○ ⑥ Cx2500Cn (Cx2500Cn) - ○ ⑧ Cx250Cn (Cx2500Cn) - ○ ⑧ Cx250Cn (Cx250Cn) - ○ ◎ Cx250Cn (Cx250Cn) - ○ ◎ Cx250Cn (Cx250Cn) - ○ ◎ Cx250Cn (Cx250Cn) - ○ ○ (Cx250Cn) - ○ ○ (Cx250Cn) - ○ (Cx	<pre>7 (* Access *) 7 (* access *) 7</pre>	:= 10; := 22; ; ; ; ; = 10;						10	0 14
	10 at 2mpl walanimawili 01	- 11.							
	Messages - Total 0 error(s), 0 warning(s), 7 messa	sge(s)							• 4
	Build	• O error	(s) 🕐 0 warning(s	i) 🚺 7 message(s) 🗙	*				
	Description					Pro	oject	Dbject Pos	sition
	Memory area 3 contains Persistent data: size	: 8168 bytes, highest used a	ddress: 45, largest o	ontiquous memory gap: 81	23 bytes (99 %)				
	Build complete 0 errors, 0 warnings : Ready	for download							

Figure 195 ブレークポイント削除完了後



CCOT-24-016 Rev.1 9.7.5. <u>ブレークポイント中の操作</u>

ブレークポイント停止中、メニューバーの「Debug」から下記の操作ができます。

	2	Start	F5	plication (De	vice: PLC Lonic)	- 01 08 42 ITE	92 da +2	2 1 0 1 1 1	= 1.5×	,
		Stop	Shift+F8	pircuton (pe	ficer recebyic)			4 1 . 1 May 1		
vices 🗸 🖬 🗙		Single Cycle	Ctrl+F5	MainTa	ssk					
SampleProject_Cx2500	10	New Breakpoint								
= 😏 🛐 Device [connected] (CX2500)	18	Edit Breakpoint			Value	Prepared value	Address	Comment		^
PLC Logic	1	Toggle Breakpoint	F9		10	riepared value	HUGIESS	comment		
= 🔘 Application [halt on breakpo	- 0	Disable Breakpoint			10					
DUT_SAMPLE		Enable Breakpoint			FALSE			Declaration	of "BOOL"	
GVL	l H	- Lindole dicakpoint		NT	7			Declaration	of "UINT"	
Library Manager	- Ç≡	Step Over	F10	u .	1			Declaration	of "RFAL"	~
PLC_PRG (PRG)	F 93	Step Into	F8			* *				
POU_CtrlA (PRG)	e_	Step Out	Shift+F10							
POU_FB (PRG)	+ <u>=</u>	Run to Cursor								
Task Configuration	3	Set Next Statement								
-dB) PLC PRG		Show Next Statement		:= 10;						
POU_CHA		Million Maleron	04157							
🖻 😏 🎲 Task1		winte values	Curi+F7							
-d] POU_FB		Force values	FZ.							
Trace_MainTask		Unforce Values	Alt+F7							
T PersistentVars	0.	Toggle Flow Control Mode								
- 😳 🛗 Cx2500Ain (Cx2500Ain)		Core Dump	•	:= 10;						
Cx2500Cntin (Cx2500Cntin)		Display Mode	•							
Cx25000in (Cx25000in)		Check Memory for Active Appli	cation							
Cx250006 (Cx250006)	-	22 stSmpl.ucHember0 0	:= 0;							100 15
• (Cx2500Pwm (Cx2500Pwm)	Brank									- 0
	Annic	ration: Application [Device: Pic Logic]								GA EM
	POL	Location	Instance Path	Tarke	Condition	Hit Count Condition	Cur	ent Hit Count	Watched Values Last Lindate	ed.
	0.0	C PPC Line 10 Column 1 (Impl)	(anu)	(anu)	Break always	Brank alwaye	e cui	count	mattines made cast opdat	
		C DDG Line 17 Colume 1 (Impl)	(any)	(any)	Break always	Break when the hit must is an	al to 5 5			
			(urif)	(arry)	break amays	break when are the court of equ				

Figure 196 ブレークポイント 操作コマンド

Table 77 ブレークポイント中の操作

操作名	摘要
Step Over	ブレークポイントで停止している行の処理が実行される。この行に関数など別 POU 呼び出
	しで合った場合、呼び出し先の処理をすべて実行した上で元の POU の次の行で停止する。
Step Into	ブレークポイントで停止している行の処理が実行される。Step Overと異なり、この行が関数
	など別 POU の呼び出しであった場合はその呼び出し先にジャンプし、その最初の行で停止
	する。
Step Out	ブレークポイント以降の全ての処理を実行し次のプログラム(POU)の先頭で停止、もしくは
	呼び出し元 POU の次の行で停止する。
Run to Cursor	ブレークポイント以外の任意の行を選択した状態でこの操作をおこなった時、その行のまえ
	までアプリケーションの動作が行われて停止する。
Set Next Statement	ブレークポイント以外の任意の行を選択した状態でこの操作をおこなった時、ブレークポイ
	ントからその行までの処理は実行されずに一気にジャンプして停止する。
Show Next Statement	現在の停止位置にフォーカスされる。



CCOT-24-016 Rev.1 9.7.6. <u>ブレークポイントステータス</u>

ブレークポイントステータス画面で設定しているブレークポイントのリストを確認できます。また、各ブレー クポイントが動作中何度機能したか確認することもできます。

下図のようにブレークポイントステータス画面が表示されていない場合、メニューバーから「View」→「Breakpoints」を選択すると画面が表示されます。

P SampleProject_Cx2500.project* - CODESYS File Edit View Project Build Onlin	ne Debug Tools Window Help	📑 🕮 Application [Device: PLC Logic]	• 09 09 • = 42 (J= 93	ei +1 8 + ∭	- ⊡ × ₹
Devices	Device PLC_PRG X Device_Application.PLC_PRG Expression ucvali dx bFig ucval dx f(Access *) //Struct f (' Access *) //Struct stSmp0.ucMember00 0 1= l1 = stSmp0.ucMember010 1= stSmp0.ucMember0112 stSmp0.ucMember011 stSmp0.ucMember01	Туре ман Туре USINT UUNT BOOL USINT DEAL 10 := 10; := 11; := 22;	Value 10 11 PALSE 7 1	Prepared value A	ddress Comment Declaratio Declaration	n of 'BOOL"
- • • • • • • • • • • • • • • • • • • •	16 //Struct(Ext) 17	0; 10 := 10; 1:= 11; 1:= 22; 0;				100
	Breakpoints				•	÷4× ΣNew X Σ IB ISIAN ® A
	POU Location PLC_PRG Line 10, Column 1 (Impl) PLC_PRG Line 17, Column 1 (Impl)	Instance Path Tasks (any) (any) (any) (any)	Condition Break always Break always	Hit Count Condition Break always Break when the hit count is equal to	Current Hit Count 6 5 5 5	Watched Values Last Updated
< >	Messages - Total 0 error(s), 0 warning(s), 7 r	message(s) 🔀 Watch 1	ற Breakpoints			
Device user: Anonymous 🛛 Last build: 📀	0 🕐 0 Precompile 🧹 HALT ON BP	Program loaded		Program unchanged	Project user: (nobody)	INS Ln 10 Col 1 Ch 1 🐶

Figure 197 ブレークポイントステータス画面


CCOT-24-016 Rev.1 9.8. <u>デバイスログ</u>

デバイスログは、デバッグ中のデバイス状態をモニタすることができます。これにより、デバイス(PLC)の内部イベントや例外を含むエラー等を確認可能です。デバイスログは、デバイスウィンドウから「Log」を選択することでログ画面が表示され、確認することができます。各ログ画面中のボタンについては Table 78 を参照して下さい。

Device X PLC_PRG	POU	_CtrlA 🕼 Task1	MainTask		•
Communication Settings	Compon	ents 🕶 🛉 🛓 🛓	9 0 2 2 0 11 0 0 Search in messages		1^
la - l'anti-se	Severity	Time Stamp	Description	PlcLog	-
Applications	0	31.03.2011 14:22:46	Application [Application] loaded	UTC Time 🛒 📄	- 11
Jackup and Restore	0	31.03.2011 14:19:52	Application [Application] loaded	CmpAppEmbedded	-
	0	31.03.2011 14:19:14	Setting router 1 address to (0000)	CmpRouter	
iles	0	31.03.2011 14:19:14	Setting router 0 address to (0002)	CmpRouter	
	0	31.03.2011 14:19:14	Network interface MyCom at router 1 registered	CmpRouter	
.og	0	31.03.2011 14:19:14	Successfully opened COM interface (1)	CmpBlkDrvCom	
	0	31.03.2011 14:19:14	CAA Async Man could not be initialized, tasks are not available	CmpCAAAsyncMan	
² LC Settings	0	31.03.2011 14:19:14	Creating events failed	CmpTraceMgr	
N C Shall	0	31.03.2011 14:19:14	Feature bootproject run in flash activated	CmpAppEmbedded	
CC Shell	0	31.03.2011 14:19:14	Network interface ether 0 at router 0 registered	CmpRouter	
Jsers and Groups	0	31.03.2011 14:19:14	Network interface: 192.168.100.2, subnetmask 255.255.255.0	CmpBlkDrvUdp	
	0	31.03.2011 14:19:14	Running as network server	CmpChannelMgr	
Access Rights	0	31.03.2011 14:19:14	1 channels available, each of the size 2048 Bytes	CmpChannelServer	
lymbol Rights					
icensed Software Metrics					
ask Deployment					
itatus K					>

Figure 198 Device タブ Log 画面

Table 78 デバイスログ 各部概要

#	名称	摘要
1	定期更新ボタン	このボタンを押した状態にした時(上図のように背景が青くなる)、デバイスからのロ
		グを定期的に画面に表示・更新する。ただし、ログは一定量超過すると削除され
		る。
2	コンポーネント選択タブ	ログ画面に表示されているものの内、このタブで選択したコンポーネント(機能)の情
		報のみ表示する。
3	情報選択ボタン	各矢印ボタンを押すことで、ログ画面に表示される情報をひとつずつ確認できる。
4	Warning	このボタンを押した状態にした時、ログ画面に Warning を表示する。
5	Error	このボタンを押した状態にした時、ログ画面に Error を表示する。
6	Exception	このボタンを押した状態にした時、ログ画面に Exception を表示する。
7	Information	このボタンを押した状態にした時、ログ画面に Information を表示する。
8	エラー検索エリア	検索エリアに値を入力した時、その値と合致するログのみ画面に表示する。
9	ログインポートボタン※22	このボタンを押すと、エクスポートしたログをインポートし、ログの確認ができる。
10	ログエクスポートボタン※22	このボタンを押すと、画面に表示されるログをエクスポートする。

※22 このボタンはログ画面上の右上部(図上青枠部)をクリックすると表示される。



9.9. <u>トレース</u>

トレースは、デバッグモード中に変数値の変化をグラフ上で視覚的にモニタできる機能です。モニタ履歴は 保存することができ、保存したファイルを読み込むことで履歴の確認をおこなうことができます。

9.9.1. <u>トレースの作成</u>

プロジェクトにトレースを作成します。作成手順の例を以下に示します。

① デバイスウィンドウにて、「Application」にカーソルを合わせ右クリックして下さい。

	생 위 위 위 비미 18+ 6	* 西 Application [Device: PLC Logic] ・ 啓 岡 → 田 裕 〔日 空田 白田 谷田 谷 中 男	5 F 1 7	
→ #	X Device X PLC_PRG	á 👔 POU_CtrlA 🎡 Task1 🛞 MainTask	 ToolBox 	~ 7
SampleProject_Cx2500	 Communication Settings 	🗘 Components - 🗟 🛊 🛬 🖞 0 💿 0 😰 0 💿 0 Search in messages	· ^	
Application	Applications	Severity Time Stamp Description The device is disconnected		
GVL Application	Backup and Restore			
Library Manager	Files			
POU_CtrlA (PRG)	Log			
Task Configuration	PLC Settings			
MainTask	PLC Shell			
E-S Task1	Users and Groups			
Devictorities	Access Rights			
Persisterity als	Symbol Rights			
Cx2500Ain (Cx2500Ain) Cx2500Cntin (Cx2500Cntin)				
Cx2500An (Cx2500An) Cx2500An (Cx2500An) Cx2500Cntin (Cx2500Cntin) Cx2500Din (Cx2500Din) Cx2500Din (Cx2500Din)	Licensed Software Metrics		*	
() 「そちらにドルギョン () Cx2500An) () Cx2500Cntn (Cx2500Cntn) () Cx2500Cn (Cx2500Cn) () Cx2500Cn (Cx2500Cn) () Cx2500Cn (Cx2500Cn) () Cx2500Cn (Cx2500Cn)	Licensed Software Metrics		>	
(* 753ubit/145 (x25004n) (x25000n(x25000n) (x25000n(x25000n) (x25000n(x25000n) (x25000n(x25000n) (x25000n(x25000n) (x25000n(x25000n) (x25000n(x25000nn) (x25000nn(x25000nn)	Licensed Software Metrics	ng(s), 0 message(s)	`	•
Image: Construction Image: Constreal construction <	Licensed Software Metrics	ng(s), 0 message(s)	, ,	·
Cr35004 (C35004r) Cr35004r) Cr35004r(C35004r) Cr35004r(C35004r) Cr35004r(C35004r) Cr35004 (C35004r) Cr35004 (C35004r) Cr35004r(C35004r) Cr35004r(C35004r) Cr35004r(C35004r)	Licensed Software Metrics K Messages - Total 0 error(s), 0 warni Build Description	rg(s), 0 message(s) • O 0 error(s) • 0 warning(s) • 0 message(s) × ×	> Project Ob	ject Positio
Image: Construction Image: Constrest Construction <	Licensed Software Metrics Messages - Total 0 error(s), 0 warni Build Description The application is up to date	ng(s), 0 message(s) • O 0 error(s) • 0 warning(s) • 0 message(s) × •	Project Ob	ject Positic
Cr3500m (Cr3504r) Cr2500m (Cr3500m) Cr25000m (Cr25000m) Cr25000m (Cr25000m) Cr25000m (Cr2500mm) Cr2500mm (Cr2500mm) Cr2500mm (Cr2500mm)	Licensed Software Metrics < Messages - Total 0 error(s), 0 warri Build Description The application is up to date Build complete 0 errors, 0 war	ng(s), 0 message(s)	Project Ob	ject Positie

Figure 199 トレース作成 Application の選択

② 表示されるコンテキストメニューから「Add Object」→「Trace..」を選択して下さい。

File Edit View Project Build Online Debug	Tools Window	v Help □□= □===============================	Applicati	on [Device: PLC Logic	1•0 00→ = ≪10=11±1=3	0 罰 〒	₹.		*
Destors Des	Communicatio	Axis Group Can table CNC program CNC program CNC settings Communication Man DUT External File Global Variable List Global Variable List	Applicati POU_C Compo Severity ager asklocal)	ria Device PLC Logic	C C C + C C + C C +	nessages	ToolBox		- 0
Gx2500De (Cx2500De) Gx2500Frqin (Cx2500Frqin) Gx2500Frqin (Cx2500Pvm)	< ⊶ ⊷ Messages - ⊕	Image Pool Interface POU	lue.			`			• #
	Descriptio The app Build or	Redundancy Configu Symbol Configuration Text List	ration	• O error	(s) warning(s) o Omessage(s) X 🕅		Project	Object	Position
		Trend Recording Man Unit Conversion	ager						

Figure 200 Trace の選択



③ 「Add Trace」ウィンドウが表示されるので、下記のようにトレース名とモニタしたいタスク名を入力・選択 し、「Add」ボタンを押して下さい。

Add Trace	>
A tool to monitor varia	ables graphically.
Name of the Trace	
Trace_MainTask	
Task for Trace Recording	
MainTask 🖉	~

Figure 201 Add Trace ウィンドウ

④ 下図のようにデバイスウィンドウにトレースが追加され、また、トレース画面が表示されます。これでトレースの作成は完了です。



Figure 202 トレース追加完了後



9.9.2. <u>トレースの設定</u>

ここではトレースを開始するための設定をおこないます。

① トレース画面右上の「Configuration」をクリックして下さい。





② 「Trace Configuration」ウィンドウが表示されるので、各種所望の値を設定し「OK」ボタンを押して下さい。

Trace Configuration			×
Trace Record	Record Settings	_	
□ Trace_MainTask	Enable Trigger		
Ai0	Trigger variable 🔻		
	Trigger edge	<u> </u>	
	Posttrigger (samples	0 5s	
	Trigger Level		
	Task	MainTask v	
	Record condition	Ai0>1000	
	Comment		
Presentation (diagrams)			
Time axis			
Diagram 1			
Y axis			
Ai0	Resolution	ms 🗸	
	Automatic restart		
	Advanced		
Add Variable	Reset Display	y settings	neel
		UK Ca	ncer

Figure 204 Trace Configuration ウィンドウ

設定項目	摘要
Enable Trigger	チェックを入れるとトレース開始をトリガ起因にできる。Trigger variable、Trigger edge を満
	たした時にトレースを開始できる。
Trigger variable	トリガ条件となる変数を入力、もしくは「…」ボタンを押して選択する。
Trigger edge	トリガ条件となるエッジの種類を選択する。
Post trigger(samples)	トリガにてトレース開始した際のサンプリング数を設定する。
Trigger level	Trigger variable が BOOL 型の変数出ない時、トリガエッジがかかる数値を設定する。
Task	トレースするタスクを設定する。
Record condition	トレース開始およびトレース継続する条件を設定できる。
Comment	このトレースについてのコメントを記入できる。
Resolution	トレースする時間の分解能を設定できる。
Automatic restart	チェックを外した状態にする。
Advanced	ボタンを押すと「Advanced Trace Settings」ウィンドウが表示される。詳細は Table 80 を
	参照。

Table 79 Trace Configuration ウィンドウ トレース設定 設定項目



Advanced Trace Settings			\times
Measure in every n-th cycle	1	~	100ms
Recommended runtime buffer size (samples)		21	2s
Override runtime buffer size			
Trace editor buffer size per variable (samples)	10001		16m40s
ок		Cance	el

Figure 205 Advanced Trace Settings ウィンドウ

設定項目	摘要
Measure in every n-th cycle	トレースのサンプリング時間を選択タブから設定できる。数値をタブ
	から選択すると、その右隣に対応するサンプリング時間(=設定値×
	タスク周期時間)が表示される。
Recommended runtime buffer size(samples)	ユーザー設定不要。上記サンプリング時間に従い、ランタイム内で
	必要となるバッファサイズが表示される。
Override runtime buffer size	チェックを入れると、ランタイムのバッファサイズを変更できる。大きさ
	は、上記バッファサイズの2倍以上で設定する必要がある。
Trace editor buffer size per variable(samples)	各変数のサンプリング最大数を設定できる。

Table 80 Trace Configuration Advanced 設定項目

③ トレース画面でモニタしたい変数を登録します。アナログ入力の Ch.0 に割り当てた変数 Ai0 を例におこな います。トレース画面右上の「Add Variable」をクリックして下さい。



Figure 206 トレース画面 Add Variables の選択



④ 「Trace Configuration」ウィンドウが表示されるので、下記のように所望の値を設定して「OK」ボタンを押 して下さい。

race Record	Variable settings	
Trace MainTask	Variable - Ai0	
Ai0	Graph color Blue	~
	Line type / Line	~
	Point type • Dot	~
	Activate minimum warning	
	Critical lower limit 0	
	Warning minimum color Black	~
	Activate maximum warning	
resentation (diagrams)	Critical upper limit 0	
 Diagram 1 	Warning maximum color Red	~
Y axis		
Shown variables		
Ai0		
<u>I Variable</u>	Reset Display settings	OK Cancel

Figure 207 Trace Configuration 変数登録ウィンドウ

設定項目	摘要
Variable	モニタ画面に追加したい変数を記入、もしくは「…」ボタンを押して所望の変数を選択す
	る。
Graph color	対象の変数の線色を設定できる。
Line type	対象の変数の線種を設定できる。
Point type	対象の変数の点種を設定できる。
Activate minimum warning	チェックを入れると、変数値がグラフの下限値を下回った時に警告が表示される。
Critical lower limit	上記警告を表示するグラフの下限値を入力することで設定できる。
Warning minimum color	上記警告表示の色を設定できる。
Activate maximum color	チェックを入れると、変数値がグラフの上限値を上回った時に警告が表示される。
Critical upper limit	上記警告を表示するグラフの上限値を入力することで設定できる。
Warning maximum color	上記警告表示の色を設定できる。

Table 81 Trace Configuration 変数登録 設定項目



⑤ トレース画面右上に追加した変数が表示されれば登録は完了です。



Figure 208 トレース画面 変数登録後



CCOT-24-016 Rev.1 9.9.3. トレースの開始・終了

ここではトレースの開始・終了について示します。開始・停止手順は下記の通りです。

【開始手順】

① デバッグモード中にメニューバーから「Trace」 \rightarrow 「Download Trace」を選択して下さい。



Figure 209 Download Trace の選択

② 9.9.2 項で設定したトレース開始条件を満たすとトレースが自動で始まります。



Figure 210 トレース中の画面



CCOT-24-016 Rev.1 【終了手順】

① トレース中に、メニューバーから「Trace」 \rightarrow 「Stop Trace」を選択して下さい。



Figure 211 Stop Trace の選択

② トレースが停止します。メイン画面右下のステータスバーに「Trace stopped」と表示されます。



Figure 212 トレース停止後の画面



9.9.4. <u>トレースの保存</u>

ここではトレースの保存手順を示します。



① トレースを停止した状態で、メニューバーから「Trace」→「Save Trace」を選択して下さい。

Figure 213 Save Trace の選択

② 「Save Trace」ウィンドウが表示されるので、任意の保存場所とファイル名を入力して「保存」ボタンを押 して下さい。これでトレースの保存は完了です。

Save Trace					×				₹
← → * ↑ ↓ PC > ダウンロード :	>		ڻ ×	、 ダウンロードのき	- 2.2.2.2.2.2.2.2.2.2.2.2.2.2.2.2.2.2.2.	8 ¢ 155 17	1.87		
整理 ▼ 新しいフォルダー					iii • ()	500Ain			
	2) 5	更新日時 2023/03/09 16:55 2022/09/07 15:29	権損 ファイル フォルダー ファイル フォルダー	サイズ 保存(S)	、 、 キャンセル 、		Config Add Va	aration riable Ai0	
Cx2500Do (Cx2500Do) Cx2500Frqin (Cx2500Frqin) Cx2500Frqin (Cx2500Pwm)	1374.0	21	n41s	• • •	• • •	2m42s	•		
	Messages - Total 0 error(s), 0 warning(s),	7 message(s)							• a >
	Build	- Oe	rror(s) 🕐 0 warning(s)	7 message(s)	××				
	Description						Project	Object	Position 4
	Memory area 3 contains Persistent da Build complete 0 errors, 0 warnings	ata: size: 8168 bytes, highest us : Ready for download	ed address: 45, largest cor	ntiguous memory gap:	8123 bytes (99 %)				
	and complete o errors, o warnings								
3	Messages - Total 0 error(s), 0 warning	(s), 7 message(s) Watch 1	Breakpoints						

Figure 214 トレースファイルの保存



CCOT-24-016 Rev.1 9.9.5. トレース履歴の読み込み

ここでは、保存したトレースファイルの読み込み手順を示します。なお、トレース中にトレースファイルを読 み込んだ場合、トレースは停止し採取中であったデータは削除されることに留意して下さい。

① メニューバーから「Trace」 \rightarrow 「Load Trace」を選択して下さい。



Figure 215 Load Trace の選択

 ② 「Load Trace」ウィンドウが表示されるので、読み込みたいトレースファイルを選択して「開く」ボタンを 押して下さい。

I Load Trace					\times				₹	
> ·	/ロード >		ٽ ~	、 ダウンロードの検索		5 ¢ 🅦	= ' ³ √			
整理 ▼ 新しいフォルダー				[EE •		500Ain				
保存 ^ 名前		更新日時	種類	サイズ			Config Add Va	uration sriable		
● OneDrive ~今日	(1)							Ai0		
PC 🔚 Sa	impleTrace1.trace	2024/03/12 14:50	Wireshark capture file	51 KB						
	り前 (2)									
Desktop	DDESYS	2023/03/09 16:55	ファイル フォルダー							
🕹 ダウンロード	sus	2022/09/07 15:29	ファイル フォルダー							
🗐 ドキュメント										
📰 ピクチャ										
📕 ビデオ										
🎝 ミュージック										
Windows (C:)										
ファイル名(N): !	SampleTrace1.trace		~	Trace file (*.trace)	~					
				開<(O)	キャンセル					
- Cx2500De (Cx2500De)	-10									
🔂 🚮 Cx2500Frqin (Cx2500Frc	ain)									
	m) 0	1s 2s 3s	4s 5s	6s 7s	85	9s	10s			
- 😌 📓 Cx2500Pwm (Cx2500Pw		warning(s), 7 message(s)							→ ‡	,
🖓 🗊 Cx2500Pwm (Cx2500Pw	Messages - Total 0 error(s), 0			A = (1) b (¥					į
🤤 🚮 Cx2500Pwm (Cx2500Pw	Messages - Total 0 error(s), 0 Build	- C	0 error(s) 🕐 0 warning(s)	T message(s)	*					
- G 🗊 Cx2500Pwm (Cx2500Pw	Messages - Total 0 error(s), 0 Build Description	• •	0 error(s) 0 warning(s)	7 message(s) X	~		Project	Object	Position	1
- G ∰ Cx2500Pvm (Cx2500Pw	Messages - Total 0 error(s), 0 Build Description Memory area 3 contains P	• C	0 error(s) 0 warning(s) st used address: 45, largest cor	Tiguous memory gap: 812	23 bytes (99 %)		Project	Object	Position	1
Ci2500Pirm (Ci2500Pirm	Messages - Total 0 error(s), 0 Build Description Memory area 3 contains P Build complete 0 errors,	• C ersistent data: size: 8168 bytes, highe 0 warnings : Ready for download	• 0 error(s) • 0 warning(s) st used address: 45, largest cor	Tiguous memory gap: 812	23 bytes (99 %)		Project	Object	Position	-

Figure 216 トレースファイルの選択



③ 読み込みが完了し、トレース画面上にデータが表示されたら読み込みは完了です。



Figure 217 トレースファイル読み込み完了後



CCOT-24-016 Rev.1 9.10. タスクステータス監視

各タスクがアプリケーション内でどの程度処理に時間を要しているか確認することができます。 確認は、デバッグモード中にデバイスウィンドウの「Task Configuration」をダブルクリックすると表示され る画面でできます。各項目に関する説明を Table 82 に示します。

Monitor Varia	ble Usage	Properties									
Task	Status	IEC-Cyde Count	Cycle Count	Configure	Last Cycle Time (µs)	Average Cycle Time (µs)	Max. Cycle Time (µs)	Min. Cycle Time (µs)	Jitter (µs)	Min. Jitter (µs)	Max. Jitter (µs)
MainTask	Valid	15086	15419	100 ms	12000	12345	24000	12000		-	
Hask1	Valid	1512	1547	1000 ms	0	566	850000	0	-		

Figure 218 Task Configuration 画面

項目名	摘要
Task	アプリケーションで作成したタスク名が表示される。
Status	タスクのステータスが表示される。
IEC-Cycle Count	アプリケーション起動後、動作中にタスクが呼び出された回数が表示される。
Cycle Count	タスクが呼び出された回数。ただし、上記と異なりアプリケーション起動後にア
	プリケーション・タスクが動作を停止していてもサイクル数は加算されることに
	留意。
Configured Cycle Period	ユーザーが設定したタスクを呼び出す周期が表示される。
Last Cycle Time(µs) ^{%23}	最後に測定されたタスクの実行時間が表示される。
Average Cycle Time(µs) ^{%23}	全てのサイクルにおけるタスクの平均実行時間が表示される。
Max. Cycle Time(µs) ^{%23}	全てのサイクルにおけるタスクの最長実行時間が表示される。
Min. Cycle Time(µs) ^{%23+%24}	全てのサイクルにおけるタスクの最短実行時間が表示される。
Jitter(µs)	非対応。
Min. Jitter(µs)	非対応。
Max. Jitter(µs)	非対応。

Table 82 タスクステータス監視 項目一覧

※23 タスクの実行時間が 1ms 未満の場合、製品仕様上画面には 0 が表示されることに留意。また、これらの数値はアプリケーションが停止している時も含まれる(CODESYS-IDE の仕様)。

※24 Min. Cycle Time(µs)が 0ms より大きい値の時に実行時間 0ms(1ms 未満)が計測された場合 Min. Cycle Time(µs)は 0 にな る。ただ、CODESYS-IDE の仕様上、Min. Cycle Time(µs)=0 の時には 0 より大きい(1ms 以上)実行時間が観測された場合 に、その値に表示を更新してしまう。これにより、実行時間が短い時に Min. Cycle Time(µs)の値が例えば 0⇔2000µs を行 き来するような場合があるが CODESYS-IDE の仕様である。



CCOT-24-016 Rev.1 9.11. シミュレーション(オフラインデバッグ)

CODESYS-IDE では CX2500 と実際に接続せずとも、シミュレーションモードとして仮想環境でデバッグを おこなうことができます。シミュレーションモードへの移行とログイン手順について下記に示します。

なお、シミュレーションモードで CX2500 と実際に接続してデバッグをおこなうことはできません。その際 にはシミュレーションモードを終了してから実施して下さい。シミュレーションモードの終了は手順①の 「Simulation」を再度選択すると終了できます。

① ログインしていない状態で、メニューバーから「Online」→「Simulation」を選択して下さい。

SampleProject_Cx2500.project* - Colored	ODESYS				- σ	×
File Edit View Project Bu	ild Online Debug Tools Window Help					7 2
Devices V R X	Login Alt+F8 Logout Ctrl+F8 Create Boot Application Download um Multiple Download	Application [Device: PLC Logic] • 🧐 🖓	한 → = 색 [대 약대 약표 *표 용 수 第 POU_CtriA 같은 Task 1 같을 Task Confe	ur.▼ ToolBox		- a x
	-pli Reset Warm Reset Cold Okc Reset Origin Simulation Security					
POUCE (04.6) P	Settings C Stell ers and Groups cess Rights mobility(s) set Software Metrics c	Estrury 22 V B-Address Iscalitot Port 2217	MyDevice (active) Device Name: MyDevice Device Address: 5031.4002 Target Div 1755 8001 Target Type: 409 Target Vendor: TORYO KERLINC,			
Messages - Total 0 error(s), 0 warning(s),	, 12 message(s)					• • • ×
Build	- O error(s) 😗 0 warning(s)	0 7 message(s) 🗙 💥				
Description Size of global data: 8916 bytes Total allocated memory size for code : Memory area 0 contains Code: size: Memory area 1 contains Data, Input,	and data: 67884 bytes 1044480 bytes, highest used address: 58208, largest contig 0.duput and Nonsafe data: size: 86016 bytes, highest used	uous memory gap: 986272 bytes (94 %) address: 9676, largest contiguous memory gap: 76340 bytes	i (88 %)	Project	Object P	osition
Memory area 2 contains Retain data:	: size: 4072 bytes, highest used address: 1, largest contiguo	us memory gap: 4071 bytes (99 %)				
Messages - Total 0 error(s), 0 warning	(s), 12 message(s) 📓 Cross Reference List					

Figure 219 Simulation の選択

② 下図のようなシミュレーションモードに切り替わります。

Semichant Settings Communication Settings Communication Settings Communication Settings Settings Settings Settings Files Settings Communication Settings Settings Files Settings Communication Settings Settings Files Settings Communication Settings Settings Files Settings Fi	Communication Settings Cost Hebsork Cadeway
Application Application DUT_SAPRE Out_Sapre Ibery Manager Ibery Manag	Applications Bedup and Restore Files Log AC Settings P.C Shell Users and Groups Access Rights Symbol Rights Licensed Software Metrics Task Deployment Ratus
Image: Constraint of the second s	Bedup and Restore Files Log Catway Catway
Image: Construction Price Image: Construction Log Image: Construction Log Image: Construction Construction Image: Construction P.C. Shell Image: Construction P.C. Shell Image: Construction P.C. Shell Image: Construction P.C. Shell Image: Construction Construction Image: Construction P.C. Shell Image: Construction Construction	File Cateway Log Cateway R.C Settings Cateway Loss staffields Cateway Symbol Rights Cateway Licensed Software Metrics Cateway Task. Deployment Cateway Status Cateway
Image: Second	Log camulation mode> R.C Settings camulation mode> R.C Shell Users and Groups Access Rights Symbol Rights Ucensed Software Metrics Task Deployment Tasks Symbol Rights
Image: Constraint of the second se	P.C Settings P.C Shell Users and Groups Access Rights Symbol Rights Lucensed Software Metrics Task. Deployment Stess
• @ MainTak PLC Shell • @ ToU_CRIA Users and Groups • @ ToU_TRI Users and Groups • @ ToU_TRI Access Rights • Ca30000 (cc3000h) Symbol Rights • Ca30000 (cc3000h) Users ad Groups • Ca30000 (cc3000h) Symbol Rights • Ca30000 (cc3000h) Users ad Shrware Metrics • Ca30000 (cc300h) Task Deployment	PLC Shell Users and Groups Access Rights Symbol Rights Ucensed Software Metrics Task Deployment Status
Image: Construction Users and Groups Image: Construction Access Rights Image: Construction Access Rights Image: Construction Symbol Rights Image: Construction Symbol Rights Image: Construction Licensed Software Metrics Image: Construction Task Deployment	Users and Groups Access Rights Symbol Rights Ucensed Software Metrics Task Deployment Status
Course Course Access Rights Acc	Access Rights Symbol Rights Licensed Software Metrics Task: Deployment Status
Co2000An (C2000An) Symbol Rights Co2000Cn (C2000Cn) Licensed Software Metrics Co2000Cn (C2000Cn) Licensed Software Metrics Co2000Cn (C2000Cn) Task Deployment	Symbol Rights Licensed Software Metrics Task Deployment Satus
CA23000n (CA2300h) CA23000n (CA2300h) CA2300n (CA2300h) Task Deployment	Licensed Software Metrics Task Deployment Ratus
Cx2500Frgn (Cx2500Frgn) Task Deployment	Task Deployment Satus
10 CV2500Dum (CV2500Dum)	Status
Status	>v
> Automation V	Information
Devices 1 POLIs <	C C C C C C C C C C C C C C C C C C C

Figure 220 シミュレーションモード移行完了後の画面



③ メニューバーから「Online」 \rightarrow 「Login」を選択して下さい。

File Edit View Project Build O	Inline Debug Tools Window Help	Not the second	₹2
₩ 1 # # # ₩ 4 (⊕ # ₩ 1 9	Login Ait+F8	I Mapplication [Device: PLC Logic] • 🧐 🧐 🕨 🔳 📽 나프 약크 약크 약크 양 수 姚피 뉴 학	/
extess • • • <th>Create Boot Application Download Online Change Source Download to Connected Device Multiple Download. Reset Vilm Reset Cold Simulation Security Operating Mode Actign Server Applications on Download Users and Groups Access Rights Symbol Rights Licensed Software Metrics Task Deployment Status</th> <th>Pervice x Pervice x Perv</th> <th>ooBox - 0)</th>	Create Boot Application Download Online Change Source Download to Connected Device Multiple Download. Reset Vilm Reset Cold Simulation Security Operating Mode Actign Server Applications on Download Users and Groups Access Rights Symbol Rights Licensed Software Metrics Task Deployment Status	Pervice x Pervice x Perv	ooBox - 0)
fessages - Total 0 error(s), 0 warning(s), 12 mess	sage(s)		- ù >
uild	O error(s) O warning(s)	7 message(s) X K	Project Object Desition (
Ascription		· · · · · · · · · · · · · · · · · · ·	roject Object Position

Figure 221 シミュレーションモード Login の選択

④ ログインの書き込み確認ウィンドウが表示されますので、「Yes」ボタンを押して下さい。

CODESYS	>	\times
2	The application 'Sim.Device.Application' does not exist on device 'Device'. Do you want to create it and proceed with download?	
	Yes No Details	

Figure 222 シミュレーションモード ログイン確認画面



⑤ デバッグモードに遷移します。これでシミュレーションモードでのデバッグ準備は完了です。デバッグ手法 については通常の場合と同様です。なお、シミュレーションモードの場合、デバイスは図のようなアイコン が表示されます。機能ドライバも同様で、シミュレーションモードでは周期的な入力値の取得はおこなえま せん。

 SampleProject_Cx2500.project* - CODES File Edit View Project Build 	YS Online Debug Tools Wind	ow Help	- 0 ×
X 🗿 🖷 😓 🗠 🗠 🚭 🔚 🏝 X	< 🏘 🍇 🍓 🍇 🏢 🦮 🦮	객 🔚 🛅 • 🔓 🖽 Application [Device: PLC Logic] • 🧐 🥨 🕨 🔳 % [프 오페 스페 스페 스페 스케 이미 (Device: PLC Logic] •	
Devices - 4 ×	Library Manager	PLC_PRG 🖉 Device 🗙 🍪 MainTask 👔 POU_FB 👔 POU_CtrlA 🔯 Task1 🙀 Task Configuration	
SampleProject_Cx2500 ✓ A Device [connected] (CX250	Communication Settings	Scan Network Gateway + Device +	
PLC Logic	Applications		
B- C DUT_SAMPLE	Backup and Restore		
- 💼 Library Manager - 📄 PLC_PRG (PRG)	Files	Galegar	
POU_CtrlA (PRG)	Log	<simulation mode=""></simulation>	
Task Configuration	PLC Settings		
	PLC Shell		
연양 Task1	Users and Groups		
POU_FB	Access Rights		
Cx2500Ain (Cx2500Ain Cx2500Cntin (Cx2500C	Symbol Rights		
Cx2500Din (Cx2500Din)	Licensed Software Metrics		
Devices POUs	Task Deployment		,
fessages - Total 0 error(s), 0 warning(s), 12 m	essage(s)		- 4 ;
Build	- O error(s) 🕚	0 warning(s) 🛛 7 message(s) 🗙 🔆	
Description		Project	Object Position
Memory area 1 contains Data, Input, Outp	out and Nonsafe data: size: 86016 byte	s, highest used address: 12256, largest contiguous memory gap: 73760 bytes (85 %)	
Memory area 2 contains Retain data: size:	4072 bytes, highest used address: 1,	argest contiguous memory gap: 4071 bytes (99 %)	
memory area 3 contains Persistent data: s Build complete 0 errors, 0 warnings : Rea	ize: 8168 pytes, nighest used address: adv. for download	45, largest contiguous memory gap: 8123 bytes (99 %)	
Messages - Total 0 error(s) 0 warning(s) 1	2 mercane(c) V Watch 1 A Break	nointe	
Device user: Anonymous Last	build: O 0 0 Precompile 🗸	To STOP SIMULATIO Program loaded Program unchanged Project use	r: (nobody)

Figure 223 シミュレーションモード ログイン完了後画面



9.12. Visualization

Visualization は、プログラムの変数値をランプやメーターなどの仮想のアイテムと CODESYS-IDE 上で紐 づけることによって、視覚的にデバッグすることができる機能です。なお、本製品は Web ブラウザ上に表示さ せる Web Visualization には対応していません。

9.12.1. Visualization 作成

ここでは、Visualization を作成する手順を示します。

① デバイスウィンドウにて、「Application」にカーソルを合わせ右クリックして下さい。



Figure 224 Visualization Applicationの選択



② 表示されるコンテキストメニューから「Add Object」→「Visualization...」を選択して下さい。

	1979 GB 85 10 11		PRG X POLICHA	(Device PLC Logic) • • • • • • • • • • • • • • • • • • •	π ₩		
SampleProject_Cx2500 Device (CX2500) Device (CX2500)	1 PROGRAM 2 VAR 3 ucV	PLC_I	RG SSINT := 10; TT := 0; BOOL := FALSE; //	/Declaration of "BOOL"			
Library M Paste			USINT := 0; // REAL := 1.0; //	/Declaration of "UINT" /Declaration of "REAL"			100 1
PLC_PRG Delete		÷	AND \$14 1.0.1	n t i. v nameron			
	ject dec dec ect ect With pplication from device 13 stSmp0.1 14 15 //Struc 17 stSmp1.1		Avis Group Cam table CNC program CNC settings DUT DuT External File Global Variable List Global Variable List (tasklocal) Image Pool Interface POL				
Cx2500Pwm (Cx2500Pwm)	https://www.project.oc/2500.pro	POU for Implicit Checks					
	20 stSmpl.	uc Ø	Redundancy Configuration Symbol Configuration				100 🙊
	Messages - Total 0 error	(s)	Text List				- ÷ ;
	Build	49	Trace	or(s) 🕚 0 warning(s) 🜒 7 message(s) 🗙 💥			
<	> Description	10 C	Unit Conversion		Project	Object	Position
Modules 👾 Devices	Memory area 1 cor	° (8)	Visualization	16 bytes, highest used address: 10432, largest contiguous memory gap: 75584			

Figure 225 Visualization の選択

③ 「Add Visualization」ウィンドウが表示されるので、Name に任意の名称を入力し VisuSymbols にチェッ クを入れた状態で「Add」ボタンを押して下さい。

Add Visualization		×
Creates a visualization o	bject	
Name:		
Visualization		
Symbol libraries	Active	
VisuSymbols (System)		
A visualization symbol library is graphics and graphical objects. library is assigned the library is	a CODESYS libra If the visualizatio added into the P	ny with on symbol OUs library
toolbox when a visualization ed	itor is the active	e shown in the editor.
	Add	Cancel

Figure 226 Add Visualization ウィンドウ



④ Visualization と Visualization Manager がデバイスウィンドウに追加され、メイン画面上に Visualization エディタ画面が表示されれば作成は完了です。



Figure 227 Visualization 追加完了後画面



9.12.2. 変数とアイテムの紐づけ例

ここでは、HMI 画面の作成、つまり、Visualization 上で以下の変数とアイテムを紐づけてデバッグの準備を おこなう例を示します。今回、変数はそれぞれの機能ドライバで定義していますが、通常のコード上の変数につ いても紐づけの方法は同様になります。

なお、他の Visualization アイテムについては CODESYS オンラインヘルプを参照して下さい。

Table 83 紐づけする変数とアイテムの組み合わせ

変数名	アイテム
Di0(デジタル入力 Ch.0 の入力値)	Power Switch
Do0(デジタル出力 Ch.0 の出力値) ^{※25}	Lamp

※25 本例では、Di0 が TRUE(ON 状態)の時に Do0 が ON にするプログラムを POU で作成していることとする。

 まず、Di0の紐づけをおこないます。Visualization エディタ画面にて、Visualization Toolbox から Lamps/Switches/Bitmaps を選択し、表示される Visualization アイテムから Power Switch をエディタ画面 にドラッグ&ドロップして下さい。



Figure 228 Visualization アイテム Power Switch の選択



② エディタ画面上に追加されたアイテムを選択すると、アイテムプロパティウィンドウが表示されます。その ウィンドウの「Position」の「Variable」に Di0 と入力(もしくは「...」ボタンを押して Di0 を選択)して下さい。



Figure 229 アイテムと変数 Di0 の紐づけ

③ 次に、Do0 の紐づけをおこないます。Visualization エディタ画面にて、Visualization Toolbox から Lamps/Switches/Bitmaps を選択し、表示される Visualization アイテムから Lamp をエディタ画面にドラ ッグ&ドロップして下さい。



Figure 230 Visualization アイテム Lamp の選択



 ④ エディタ画面上に追加されたアイテムを選択すると、アイテムプロパティウィンドウが表示されます。その ウィンドウの「Position」の「Variable」に Do0 と入力(もしくは「...」ボタンを押して Do0 を選択)して下 さい。これで紐づけは完了です。



Figure 231 アイテムとDo0の紐づけ



9.12.3. Visualization 実行例

ここでは、9.12.3 項で作成した Visualization をデバッグモードで実行する例を示します。

① デバイスと接続しログインし、アプリケーションの動作を開始して下さい。



Figure 232 デバッグモード中画面

② 動作を開始して CX2500 のデジタル入力 Ch.0 の入力状態を変化させると下記のようにスイッチとランプが 状態に応じて変化することが確認できます。



Figure 233 Di0 の状態変化に伴う Visualization アイテムの様子



10. <u>Tips</u>

10.1. プログラム中に IDE と通信できなくなった時は・・・

プログラム中やデバッグ中に IDE と通信できなくなった時は、以下のような可能性が考えられます。製品破損認定の前に、それぞれの場合に従って確認して下さい。

【通信できなくなった時に考えうる原因と対処】

- 通信ラインの接触不良
 - ▶ 取扱説明書「CX2500Codesys_UserManual_ForSetup」の 8.1 節にある対処法を試して下さい。
- その他
 - ▶ 取扱説明書「CX2500Codesys_UserManual_ForSetup」の 7.2 節を参考にユーザーの CX2500 にダ ウンロードされているバージョンのランタイムを弊社 HP からダウンロードし、CX2500 にランタ イムを書き込んでください。
 - ▶ または、10.1.1 項のセーフモードを実施してください。



10.1.1. <u>セーフモード起動</u>

CX2500を再起動しても IDE と通信できない場合、CX2500 をセーフモードで起動する必要があります。セ ーフモードで起動すると、下記の通りユーザーアプリケーションは保持されていますが、CX2500 は通電中、 新たなアプリケーションの書き込みを待ち続けます。

下図の通り、CX2500の起動前に指定の入力(条件 A)を ON 状態にした上で起動(イグニッション入力を ON) にするとセーフモードで起動できます。セーフモードで起動した場合、デバイスログにセーフモードで起動し た旨のメッセージが表示されます。



¢ Co	ompone	ents 🕶 🚹 🏦 🚇 💻	1 0 2 2 E 0 1 10 0 Search in messages	og	•
Sever	rity	Time Stamp	Description	Component	_
•)	09.04.2011 20:20:15	CX2500 booted safe mode! Please flash your application!	CmpAppEmbedded	
			Finance 005 デバノフログ セーフエードお動性の かいちージ		•

Figure 235 デバイスログ セーフモード起動時のメッセージ



11. <u>Revision history</u>

Rev	Date	Substantive changes
0	2024.06.11	Initial release
1	2024.10.1	● 6.9.3 項 当社独自ライブラリ
		・新規追加。
		● 6.9.4 項
		・旧 6.9.2.1 項の内容であり、項目番号が左記より変更。
		● 6.9.5 項
		・旧 6.9.3 項の内容であり、項目番号が左記より変更。
		● 7.2.3.4 項 アプリケーションで使用しない機能ドライバの設定
		・新規追加。
		● 7.12.3 項
		・Write 関数の返り値の誤記を修正。(SENDING_ERROR の削除)
		・IsSendingActive 関数の返り値の誤記を修正。
		● 7.12.4.2 項
		・マスクレシーバー例の表中の誤記を修正。
		・使用ライフラリ変更につき、関数他関連機能を変更。
		・旧 Table.55 削除。