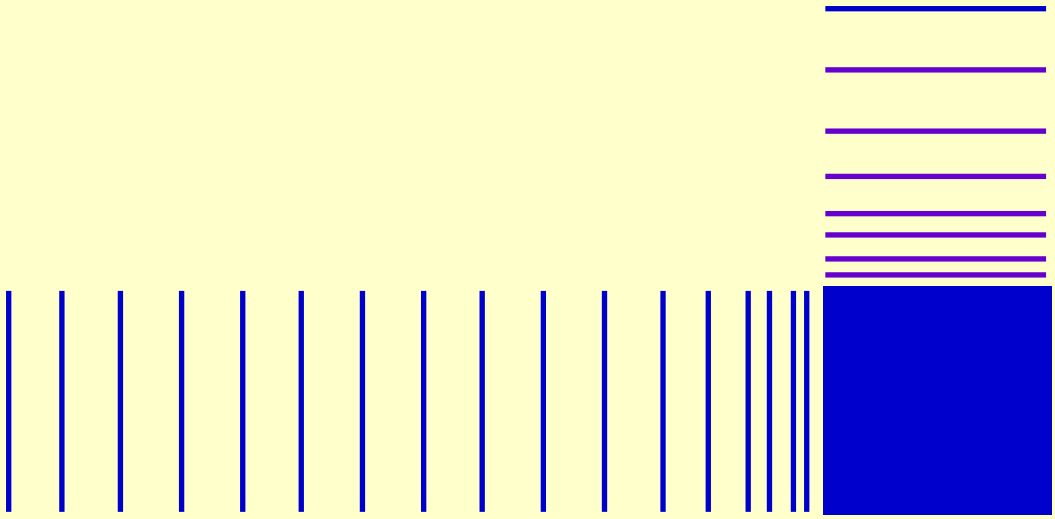
CODESYS®向け FL-net 通信ライブラリ

FL-net For CODESYS®

ユーザーズガイド



HFLN-01-03

この製品を輸出される場合には、外国為替及び外国貿易法の規制
並びに米国輸出管理規則など外国の輸出関連法規をご確認のうえ、
必要な手続きをお取りください。
なお、不明な場合は、弊社担当営業にお問い合わせください。

２０２１年 ４月（第１版）ＨＦＬＮ－０１－０１（廃版）
２０２１年 ９月（第２版）ＨＦＬＮ－０１－０２（廃版）
２０２２年 ７月（第３版）ＨＦＬＮ－０１－０３

ご注意

- このソフトウェアをご使用になる前に、このマニュアルの記載内容をよく読み、書かれている指示や注意を十分理解してください。

- このマニュアルの記載内容について理解できない内容、疑問点または不明点がございましたら、最寄りの当社営業までお知らせください。

- 当社提供ソフトウェアを改変して使用した場合には、発生した事故や損害につきましては、当社は責任を負いかねますのでご了承ください。

- 当社提供以外のソフトウェアを使用した場合の信頼性については、当社は責任を負いかねますのでご了承ください。

- このソフトウェアが万一故障したり、誤動作やプログラムに欠陥があった場合でも、ご使用されるシステムの安全が十分に確保されるよう、保護・安全回路は外部に設け、人身事故・重大な災害に対する安全対策などが十分確保できるようなシステム設計としてください。

はじめに

本マニュアルは、「FL-net For CODESYS®（S-763A-97P）」の使い方などについて記述したものです。本ソフトウェアをご使用になる前に、このマニュアルをよくお読みください。

＜マニュアル構成＞

このマニュアルは、次のような構成となっています。

第1章 FL-net For CODESYS®とは

第2章 FL-net For CODESYS®のインストール

第3章 サポート機能

第4章 ライブラリ関数

第5章 プロファイル情報

＜商標について＞

・Microsoft®、Windows®は、米国Microsoft Corporationの米国およびその他の国における登録商標または商標です。

・CODESYS®は、ドイツCODESYS GmbHの登録商標です。

・Intel®は、米国およびその他の国におけるIntel Corporationの商標です。

・上記以外にこのマニュアルに記載されている他社製品名（ソフトウェア、ハードウェア）は、各社の登録商標、商標、または商品です。

＜セキュリティに関する推奨事項＞

FL-net通信は、クローズドなネットワークでの使用を推奨します。本ライブラリでFL-net通信を行う装置を外部ネットワークに接続する場合、ファイアウォールや不正アクセスを検知できる仕組みの導入など、ネットワーク全体で十分なセキュリティ対策を講じてください。

目次

第1章 FL-net For CODESYS®とは

「FL-net For CODESYS®」は、ドイツ CODESYS GmbH（以下、CODESYS 社と称す）のソフトウェア PLC（Programmable Logic Controller）である CODESYS®上で FL-net 通信を実現するためのライブラリインタフェースです。通信プロトコルは、FL-net Ver.3/クラス 1 に準拠しています。CODESYS®開発環境に本ライブラリをインストールすることで、CODESYS®リアルタイム実行環境（ランタイム環境）上の PLC プログラムで動作する FL-net 通信処理の実装が可能になります。

■ CODESYS®とは

CODESYS®は、CODESYS社が開発したソフトウェアPLCです。国際標準規格IEC 61131-3で定義されるプログラミング言語に対応した開発環境と、開発環境で製作した制御用アプリケーションを動作させるためのリアルタイム実行環境で構成されます。「FL-net For CODESYS®」は、以下のCODESYS®バージョン（x64）に対応しています。

・開発環境 CODESYS® Development System ：V3.5 SP16 Patch 4
・リアルタイム実行環境 CODESYS® Control RTE ：V3.5 SP16 Patch 4

以下に、本ライブラリを用いた PLC プログラム作成の概要図を示します。

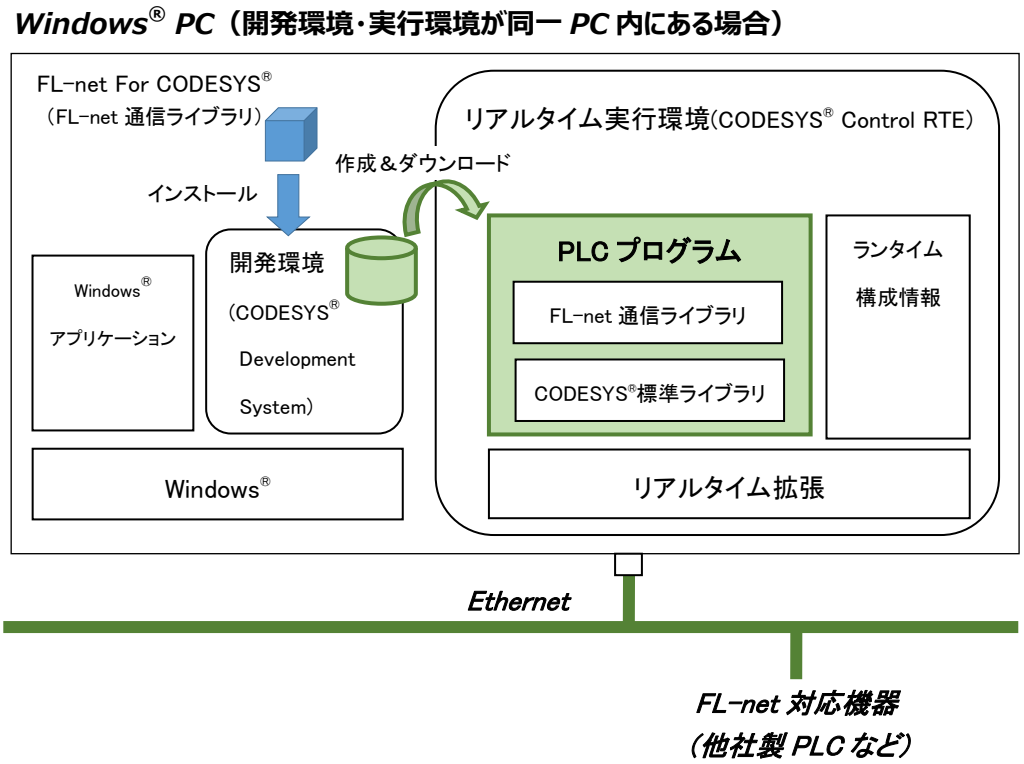


図 1「FL-net For CODESYS®」概要

＜関連・参考ドキュメント＞

本書で用いる主な用語の定義などについては、下記ドキュメントをご参照ください。

No.	ドキュメント名	備考
1	JEM1479 FA コントロールネットワーク標準 -プロトコル仕様	2012 年 9 月 27 日 改定(第 3 回)
2	JEM-TR213 FA コントロールネットワーク[FL-net(OPCN-2)] -実装ガイドライン	2011 年 7 月 12 日 改定(第 4 回)
3	HF-W100E/IoT スタートアップガイド	(*1)
4	HF-W2000/IoT モデル 58/55/50 HF-W400E/IoT スタートアップガイド	(*1)

(*1) 本ライブラリに対応した CODESYS®を搭載する IoT 対応産業用コントローラ HF-W/IoT のソフトウェア PLC の使い方などについて記述したものです。以下の URL からご参照ください。
https://www.hitachi-ip.co.jp/products/hfw/products/iot_ctr/download.html

第2章 FL-net For CODESYS®のインストール

2.1. インストール作業の前に

「FL-net For CODESYS®」のインストールに際して、インストール作業で必要な項目 および FL-net 通信ライブラリの動作環境について説明します。

インストール作業で必要な項目を以下に示します。作業を開始する前に予め確認しておいてください。

＜インストール作業で必要になる項目＞

項目	内容
Name	任意の名前を入力ください。
Serial number	本ソフトウェアのインストールで必要な 12 桁のシリアルナンバーです。 本製品購入時に、弊社より通知されたものを入力ください。

ソフトウェア開発・実行環境を以下に示します。

＜ソフトウェア開発環境＞

項目	内容
CODESYS®開発環境	CODESYS® Development System（V3.5 SP16 Patch 4）

＜ソフトウェア実行環境＞

項目	内容
FL-net 製品カテゴリ/機能クラス	FL-net Ver.3/クラス 1
OS（Windows®）	Microsoft® Windows® 10 Iot Enterprise（64bit）
CODESYS®リアルタイム実行環境	CODESYS® Control RTE（V3.5 SP16 Patch 4）
ネットワーク設定	FL-net ネットワークへの参加では、LAN ポート 1 つを占有します。また、以下の設定が必要です。 ・Intel® Pro/1000 互換のイーサネットコントローラを搭載した LAN ポートをお使いください。また、CODESYS® Control RTE に同梱のドライバ（CmpEt1000Drv）を適用してください。 ・TCP/IP のプロパティ設定で、固定 IP アドレスを割り当ててください。

留意事項

装置に DVD ドライブが搭載されていない場合は、USB 接続の外付け光ディスクドライブ（DVD メディアを読み込めるドライブ）を用意して装置に接続してください。

2.2. FL-net For CODESYS®のインストール

FL-net For CODESYS®のインストール手順について説明します。なお、インストールはコンピューターの管理者アカウントでログオンして行ってください。

① Administrator 権限を持つアカウントでログオンします。

② 「ファイル名を指定して実行」ウィンドウを開きます。
［スタート］ボタンを右クリックし、表示されたメニューより「ファイル名を指定して実行」をクリックします。

③ セットアッププログラムを起動します。名前のボックスに以下を入力して[Enter]キーを押します。
"D:¥HFLnet.msi"

※ DVD 媒体でのインストール手順として、DVD ドライブを D ドライブと仮定したパスを指定しています。下線部のパスはセットアッププログラムが格納されている場所に応じて変更してください。

④ 「FL-net For CODESYS®」用のセットアップウィザード画面が表示されます。「Next」ボタンをクリックします。

⑤ 画面に従いインストールします。Name には任意の名前を、Serial number には本ライブラリ購入時に弊社から提供されたキーを入力します。
「ユーザーアカウント制御」画面が表示される場合は、「はい」ボタンをクリックします。

⑥ FL-net For CODESYS®のインストールが完了したことを示す画面が表示されます。「Close」ボタンをクリックしてセットアッププログラムを終了します。

⑦ Windows®を再起動します。

2.3. FL-net For CODESYS®のアンインストール

FL-net For CODESYS®のアンインストール手順について説明します。なお、アンインストールはコンピューターの管理者アカウントでログオンして行ってください。

① 「スタート」メニューから「コントロールパネル」を開きます。［スタート］ボタンをクリックし、［Windows システムツール］-［コントロールパネル］をクリックします。

② 「プログラムのアンインストール」を選択します。

③ インストールされているプログラムのリストから、「FL-net For CODESYS®」を選択し、アンインストールを実行します。

④ 「FL-net For CODESYS®」の削除を確認するメッセージが表示されます。「はい」ボタンをクリックします。
「ユーザーアカウント制御」画面が表示される場合は、「はい」ボタンをクリックします。

⑤ Windows®を再起動します。

2.4. ライブラリの追加

CODESYS®の開発環境に FL-net 通信ライブラリを追加する手順を説明します。以下の手順を行う必要があります。

(1) ライブラリのリポジトリに FL-net 通信ライブラリ（パッケージ形式）をインストール

(2) ライブラリマネージャーに FL-net 通信ライブラリを追加

2.4.1. ライブラリインストール方法（パッケージ形式）

ライブラリのインストールを一度でも実施している場合、以下の手順は不要です。「2.4.2 ライブラリ追加方法」の手順から実施してください。

① CODESYS®開発環境上部の「ツール」-「パッケージマネージャー」をクリックします。

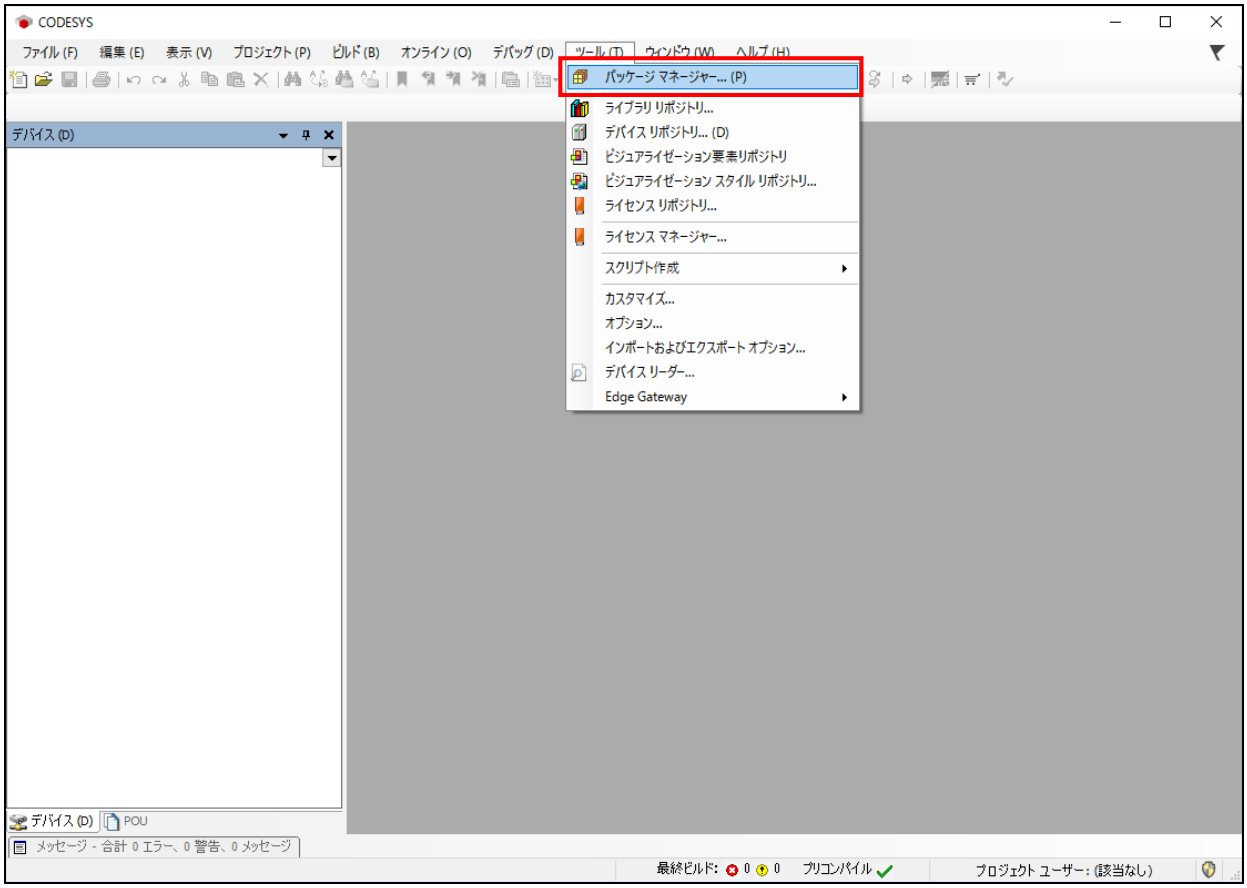


図 2 パッケージマネージャー表示

② 表示されるパッケージマネージャーダイアログで「インストール」ボタンを押下します。

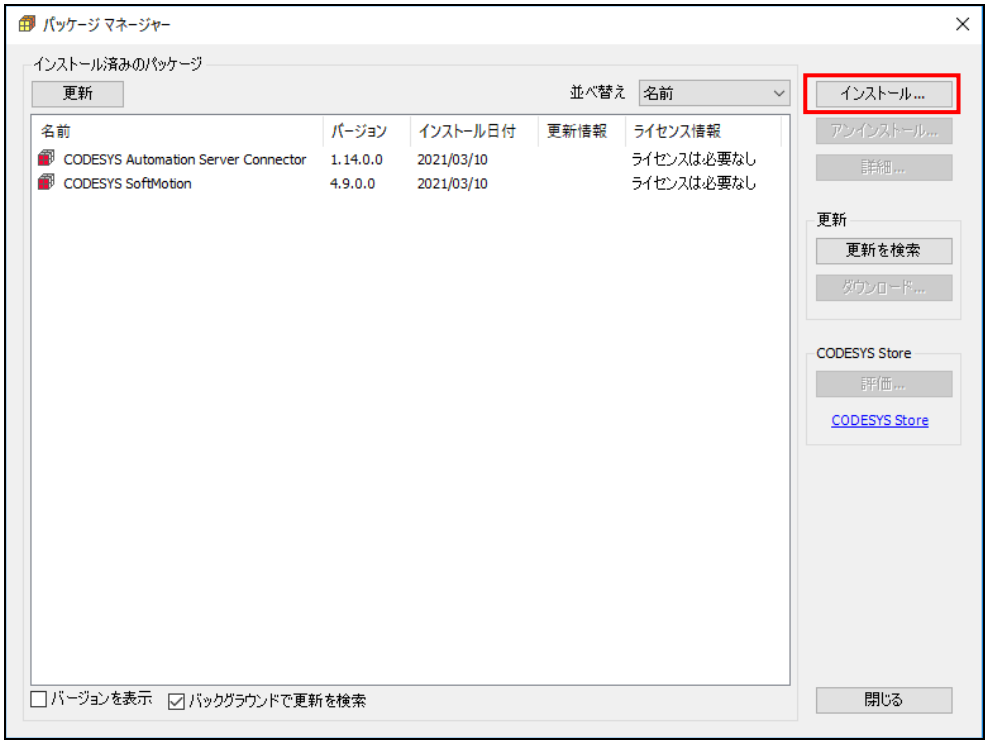


図 3 パッケージマネージャーダイアログ

③ 「C:¥Program Files¥HFLnet¥Library」フォルダ内の「CmpHFLnet.package」ファイルを指定し、「開く(O)」ボタンを押下します。

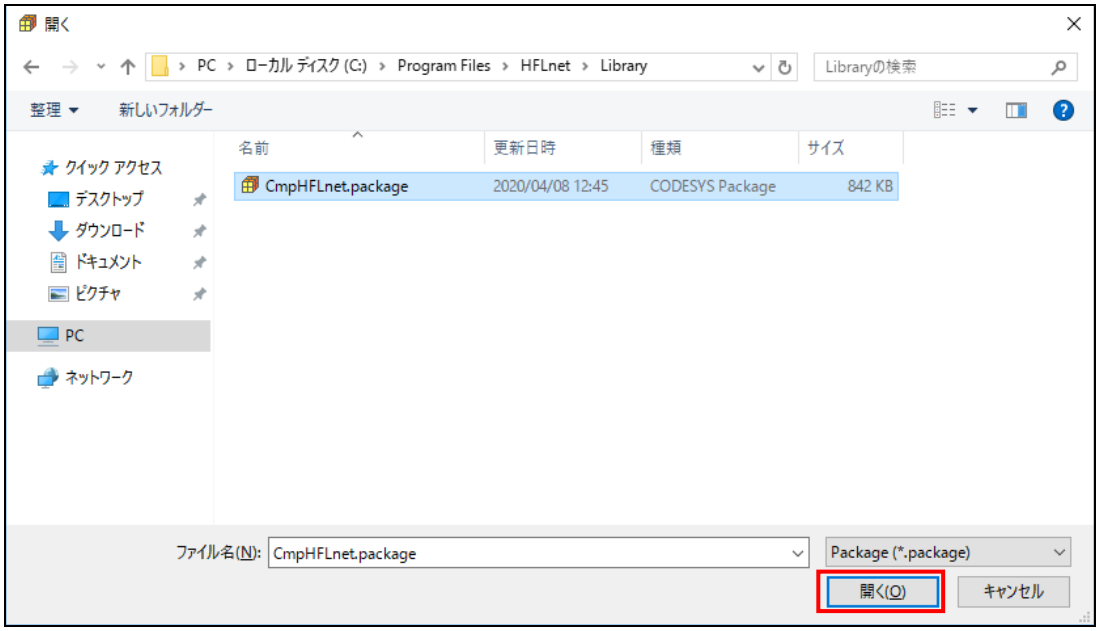


図 4 ライブラリパッケージインストール

④　パッケージのインストール画面が表示されます。画面に従いインストールを進めます。License Agreement の画面が表示されますので、内容を確認し「私は、上記の使用許諾契約を読んで理解したので同意します。」にチェックを入れ「Next」ボタンを押下します。

図 5 インストール画面(License Agreement)

⑤　セットアップの種類を選択します。デフォルトの「代表的なセットアップ」を選択したまま「Next」ボタンを押下します。「ユーザーアカウント制御」画面が表示される場合は、「はい」ボタンをクリックします。

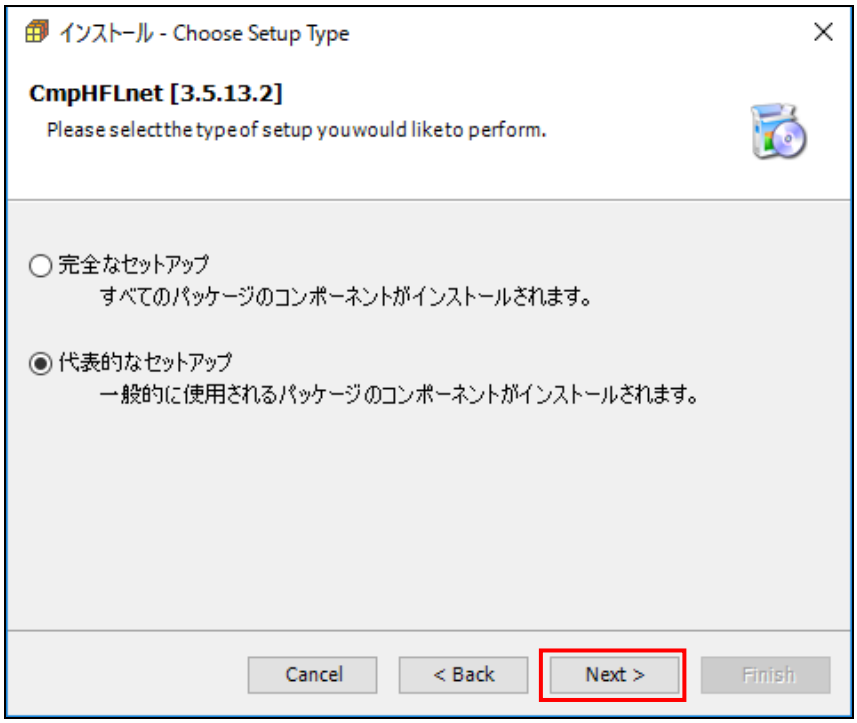


図 6 インストール画面(Choose Setup Type)

⑥　パッケージのインストールを開始します。1～2 分後、下記画面が表示されインストールが正常終了したことを確認し「Next」ボタンを押下します。

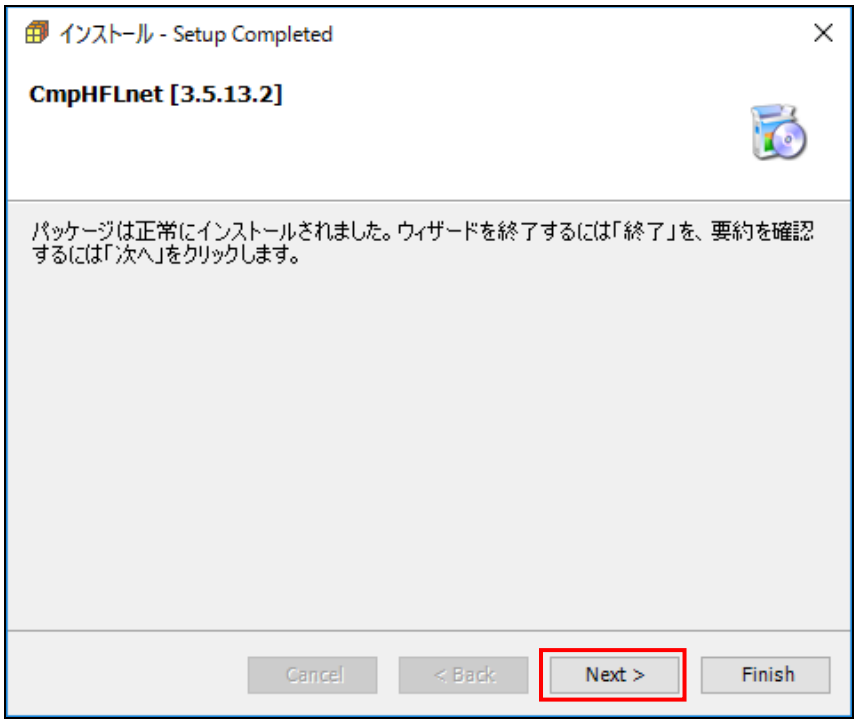


図 7 インストール画面(Setup Completed)

⑦　インストールの概要を確認し、「Finish」ボタンを押下して終了します。

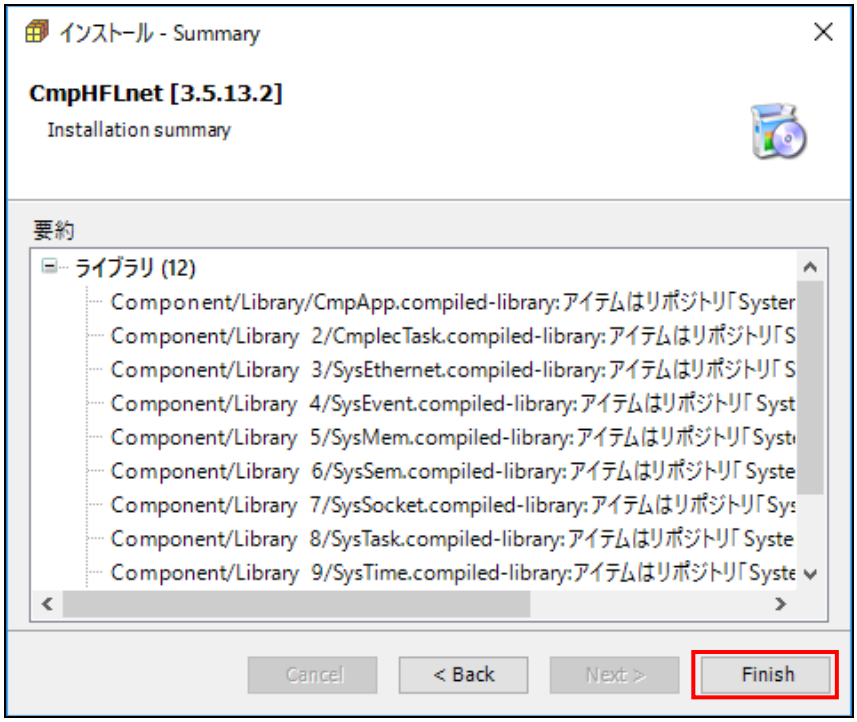


図 8 インストール画面(Summary)

⑧　パッケージマネージャーダイアログで「CmpHFLnet」が表示されることを確認します。

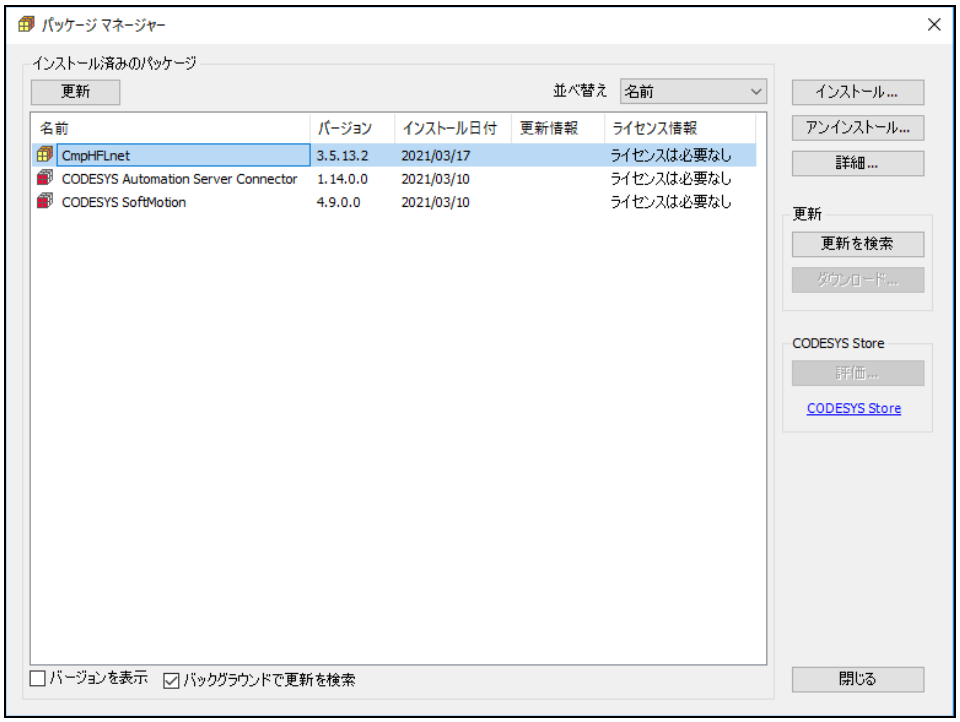


図 9 パッケージマネージャーダイアログ(インストール後)

2.4.2. ライブラリ追加方法

① デバイス欄の「ライブラリマネージャー」をダブルクリックします。

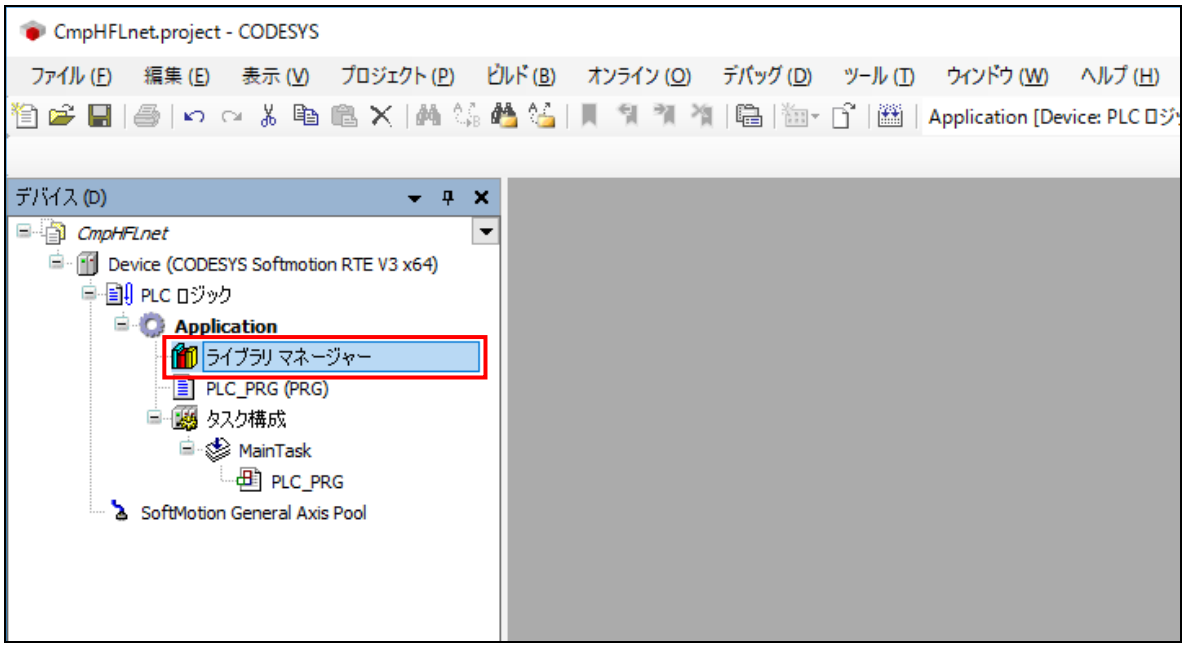


図 10 ライブラリマネージャー表示

② ライブラリマネージャー内の「ライブラリの追加」を押下します。

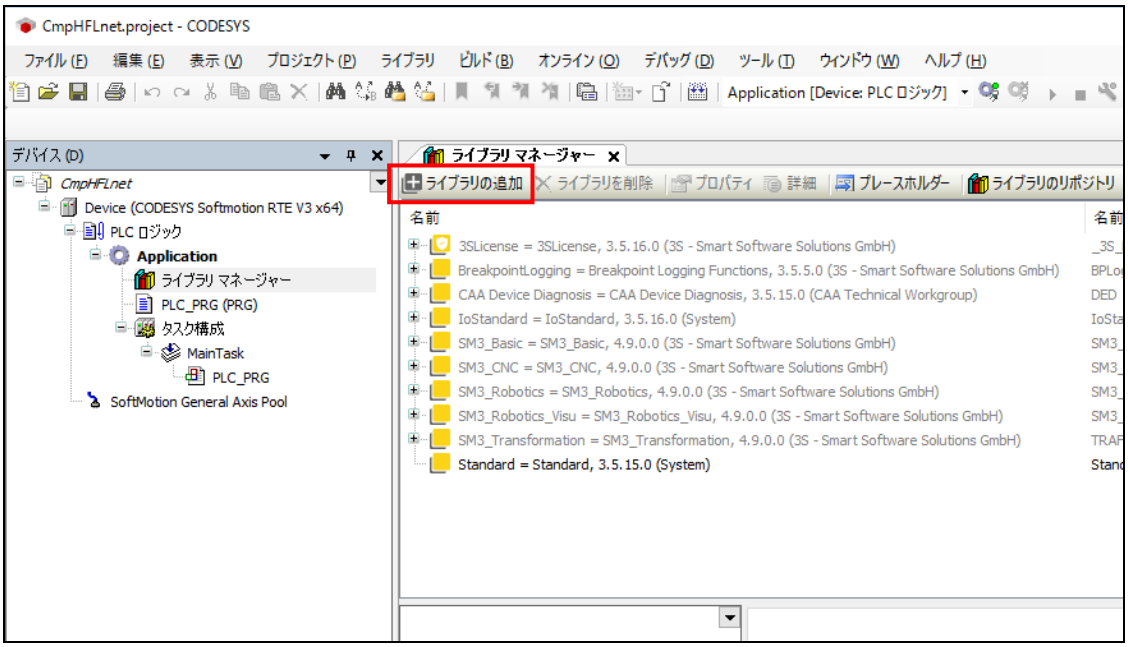


図 11 ライブラリ追加ダイアログ表示

③　ダイアログが表示されたら、左下部の「上級者向け...」ボタンを押下します。

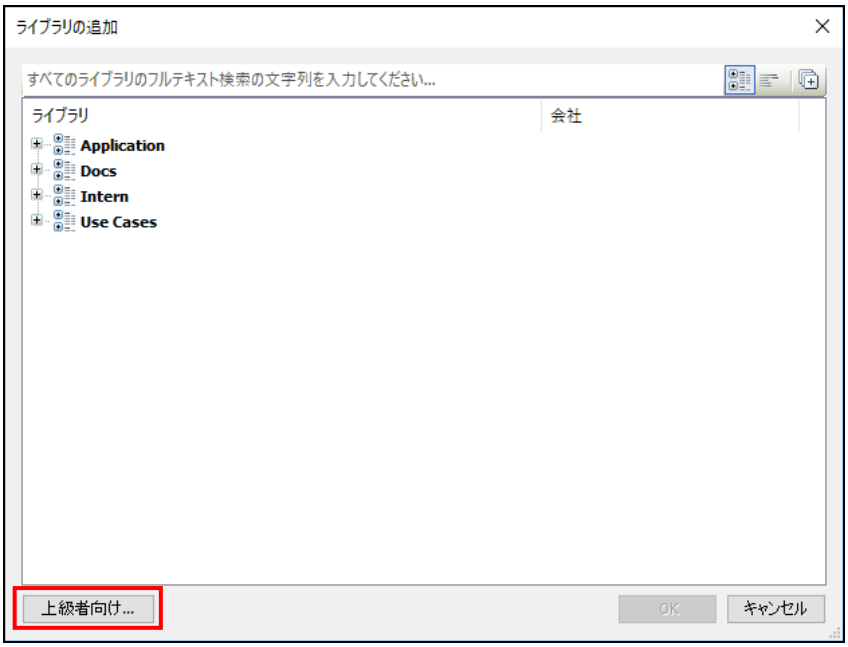


図 12 ライブラリ追加ダイアログ

④　ダイアログが切り替わったら、「System ⇒ SysLibs」内の「CmpHFLnet」を選択し、OK ボタンを押下します。

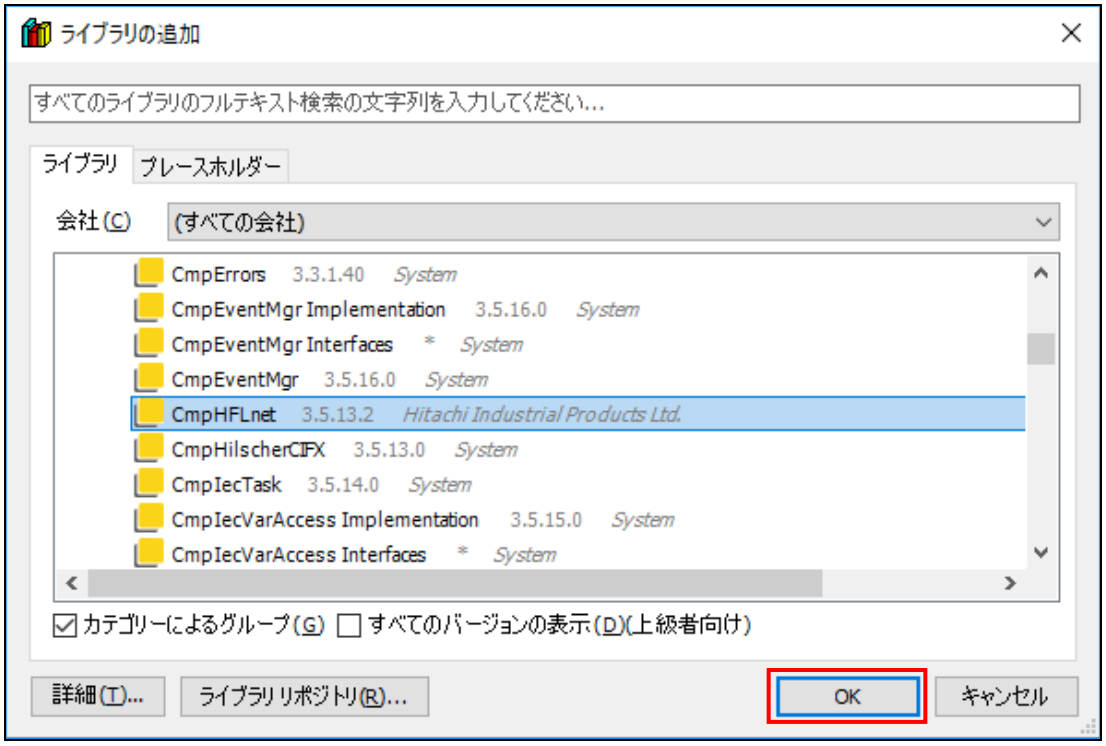


図 13 ライブラリ追加

⑤　なお、検索バーを利用し、ライブラリの検索をすることも可能です。

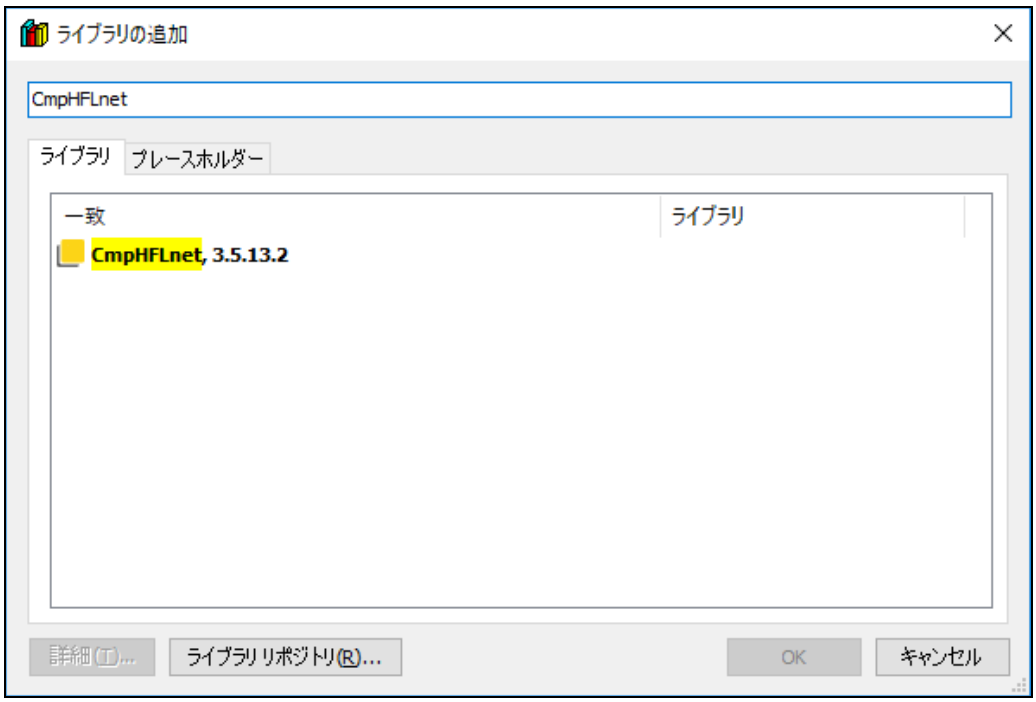


図 14 ライブラリ追加(検索バー使用時)

⑥　ライブラリマネージャーに以下のライブラリが追加されたら完了となります。

・CmpHFLnet, 3.5.13.2 (Hitachi Industrial Products Ltd.)

※　会社情報には「, (カンマ)」が使用できないため省略しています。

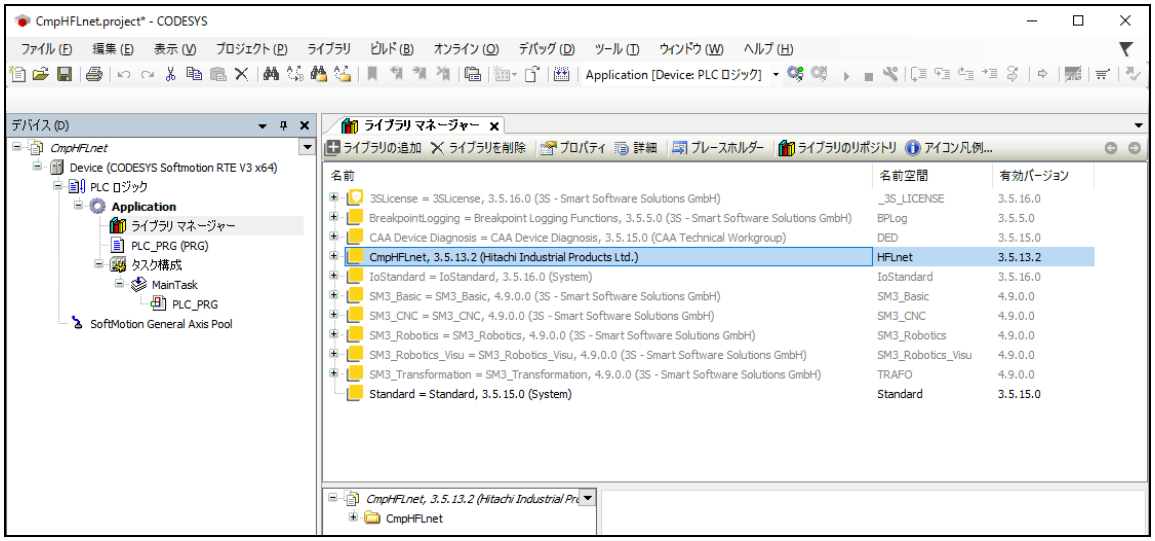


図 15 ライブラリ追加

第3章 サポート機能

3.1. サポート機能一覧

「FL-net For CODESYS®」が提供する機能の一覧と関連するライブラリ関数を以下に示します。

No.	機能		内容・関連ライブラリ関数
1	コンフィギュレーション用パラメータ設定		FL-net ネットワークに参加するための各種パラメータの設定を行います。 【関連ライブラリ関数】 ・HFA_SetConfigParam（4.2.1(1)節）
2	ネットワーク参加		FL-net ネットワークへ参加します。 【関連ライブラリ関数】 ・HFA_LinkIn（4.2.1(2)節） ・HFA_GetLinkInStatus（4.2.1(11)節）
3	ネットワーク離脱		FL-net ネットワークから離脱します。 【関連ライブラリ関数】 ・HFA_LinkOut（4.2.1(3)節）
4	コモンメモリ情報管理		コモンメモリ内のデータを管理します。 【関連ライブラリ関数】 ・HFA_ReadCommon（4.2.1(4)節） ・HFA_WriteCommon（4.2.1(5)節）
5	ネットワーク情報管理		FL-net ネットワークの情報を管理します。 【関連ライブラリ関数】 ・HFA_GetNodeStatus（4.2.1(6)節） ・HFA_GetNetworkStatus（4.2.1(7)節） ・HFA_GetMyNodeLog（4.2.1(8)節） ・HFA_ClearMyNodeLog（4.2.1(9)節） ・HFA_SetControlEquipment（4.2.1(10)節）
6	メッセージ伝送	サーバ機能	下記メッセージについて応答します（実装クラス 1 の必須メッセージ応答機能をサポート）。 ・ネットワークパラメータリード ・プロファイルリード ・メッセージ折返し ・ログデータリード ・ログデータクリア
		クライアント機能	－（非サポート）

第4章 ライブラリ関数

4.1. 概要

本章では、本ライブラリのインタフェース（I/F）について記載します。本ライブラリは、CODESYS®開発環境で作成した PLC プログラム上で動作するものです。以下に、本ライブラリを用いた場合の I/F 仕様の概要図を示します（太線箇所の I/F 仕様です）。

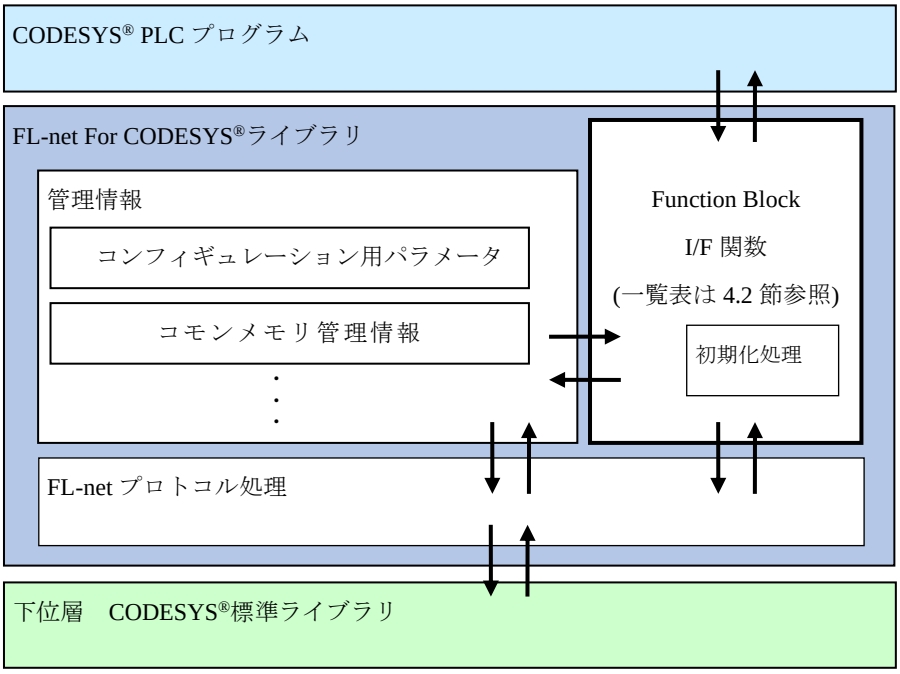


図 16 ライブラリ関数I/F仕様概要

留意事項

・本ライブラリで、デバイスが参加可能な FL-net ネットワークは 1 つまでです。

・FL-net 通信で使用する I/F 関数は、ファンクションブロック(FB)形式で定義されています。PLC プログラム作成の際、変数宣言部で FB のインスタンス化をすることで、I/F 関数が使用可能になります。

・本ライブラリはインスタンス化した際、ライブラリ内部に管理情報を持ちます。そのため、CODESYS®プロジェクト内で扱うインスタンスの数は必ず 1 つまでにしてください。

・関数（FUN）で FB のインスタンス化を行うと、処理終了時にインスタンスがスタックから廃棄されます。そのため、関数（FUN）で本ライブラリのインスタンス化は行わないでください。

・関数（FUN）や複数のプログラム（POU）で FL-net のライブラリ関数を使用したい場合は、グローバル（GVL）変数にて本ライブラリのインスタンス化を行い、その変数を使用してください。

・本ライブラリ内部で動作している FL-net フレーム受信用の管理タスクは、優先度 1（0～31 の内）で動作しています。そのため、FL-net 通信より優先させたいタスクがある場合は優先度 0 を、FL-net 通信を優先させたい場合は優先度 2 以降の値に設定することを推奨します。

4.2. ライブラリ関数一覧

ライブラリ関数の一覧を下表に示します。また、次ページに関数の戻り値の型を示します。

No.	関数名	説明	記載節
1	HFA_SetConfigParam	FL-net のコンフィギュレーション用パラメータ設定	4.2.1(1)
2	HFA_LinkIn	FL-net ネットワークへの参加	4.2.1(2)
3	HFA_LinkOut	FL-net ネットワークからの離脱	4.2.1(3)
4	HFA_ReadCommon	コモンメモリ読出し	4.2.1(4)
5	HFA_WriteCommon	コモンメモリ書込み	4.2.1(5)
6	HFA_GetNodeStatus	ノード管理情報パラメータ読出し	4.2.1(6)
7	HFA_GetNetworkStatus	ネットワークステータス読出し	4.2.1(7)
8	HFA_GetMyNodeLog	自ノードログ情報読出し	4.2.1(8)
9	HFA_ClearMyNodeLog	自ノードログ情報クリア	4.2.1(9)
10	HFA_SetControlEquipment	運転／停止状態設定	4.2.1(10)
11	HFA_GetLinkInStatus	FL-net ネットワーク参加状態の読出し	4.2.1(11)

＜ライブラリ関数の戻り値＞

ライブラリ関数の戻り値の型を下表に示します。なお、HFA_LINK_STATUS 型の戻り値となるのは HFA_GetLinkInStatus 関数のみで、それ以外の関数の戻り値の型は HFA_RESULT 型です。

【HFA_RESULT】

No.	名称	値	内容
1	Normal	0	正常終了
2	NotLinkIn	2	自ノード未参加
3	DestNodeLinkOut	3	送信先ノード未参加
4	AlreadyLinkIn	4	既に参加中
5	SubjectNodeLinkOut	5	対象ノード未参加
6	AlreadyLinkWait	7	既に参加待機中
7	CommonDisable	8	コモンメモリデータ無効
8	InvalidParam	-1	引数異常
9	SystemError	-100	システムエラー

【HFA_LINK_STATUS】

No.	名称	値	内容
1	LinkWait	0	参加待機中
2	LinkIn	1	参加中
3	LinkOut	2	離脱中

4.2.1. I/F 仕様詳細

（1）HFA_SetConfigParam －FL-net のコンフィギュレーション用パラメータを一括で設定

＜名　　前＞　HFA_SetConfigParam

＜形　　式＞　HFA_RESULT result := HFA_SetConfigParam(Param)

```
VAR_INPUT
  Param : POINTER TO HFA_CONFIG_PARAM;
END_VAR
```

＜機能説明＞

コンフィギュレーション用パラメータを一括で設定します。コンフィギュレーション用パラメータの設定は FL-net ネットワーク未参加時に設定可能です。設定する項目は省略できませんので、全てのメンバに適切な値を設定してください。1 つでも設定に失敗すると引数異常になり、コンフィギュレーション用パラメータの設定は更新されません。

Param :FL-net のコンフィギュレーション用パラメータ。

＜戻り値＞

以下の値を返します。

No.	値	内容
1	0	正常終了
2	-1	引数異常
3	4	既に参加中
4	7	既に参加待機中

【HFA_CONFIG_PARAM 構造体】

No.	名称	型	値の範囲	内容
1	NodeNo	BYTE	1～254	ノード番号
2	Common1Addr	WORD	16#0000～16#01FF	コモンメモリ領域 1 アドレス（ワード単位）
3	Common1Words	WORD	16#0000～16#0200	コモンメモリ領域 1 サイズ（ワード単位）
4	Common2Addr	WORD	16#0000～16#1FFF	コモンメモリ領域 2 アドレス（ワード単位）
5	Common2Words	WORD	16#0000～16#2000	コモンメモリ領域 2 サイズ（ワード単位）
6	TokenWatchTime	BYTE	1～255	トークン監視時間 （1ms 単位）
7	MinFrameInterval	BYTE	0～50	最小許容フレーム間隔 （100μs 単位）
8	NodeName	STRING(10)	ASCII 文字で最大 10 文字	ノード名(設備名)
9	LocalIP	ARRAY [0..3] OF BYTE	4 バイトの整数	ローカル IP アドレス
10	IntTaskCycle	DWORD	100～5000	[オプション] FL-net フレーム受信用管理タスク実行周期 （1μs 単位。デフォルトは 1000μs を設定）

留意事項

・ノード番号は、他ノードと重複しない番号を指定する必要があります。ノード番号の重複を検出した場合、FL-net ネットワークへの参加が失敗します。

・コモンメモリ領域は、他ノードと重複しない送信領域にする必要があります。他ノードの領域と重複した場合、コモンメモリの送信領域はゼロになります。

・トークン監視時間は、コモンメモリの送信領域および最小許容フレーム間隔の設定に応じて、値を調整してください。

・FL-net ネットワーク上に通信性能が遅い機器が存在する場合は、最小許容フレーム間隔の値を必要に応じて調整してください。

・オプションの設定として、FL-net 通信ライブラリ内でフレーム受信のチェックを行う、管理用タスクの実行周期を設定できます。応答性能のチューニングが必要となった場合に、ご使用ください。なお、本パラメータについては、初回の HFA_SetConfigParam 関数実行時のみでしか、設定の反映が行えません。また、許容範囲外の値が設定された場合は、デフォルトの 1000μs を設定します。

＜使用例＞

ST（Structured Text）言語にて、コンフィギュレーション用パラメータを設定する例です。

```
【変数宣言部】
PROGRAM PLC_PRG
VAR
    Param : HFA_CONFIG_PARAM;
    FLnet : HFLnet;
    result : HFA_RESULT;
END_VAR

【処理部】
// Set param
Param.Common1Addr := 0;
Param.Common1Words := 10;
Param.Common2Addr := 0;
Param.Common2Words := 10;
Param.LocalIP[0] := 192;
Param.LocalIP[1] := 168;
Param.LocalIP[2] := 250;
Param.LocalIP[3] := 1;
Param.MinFrameInterval := 0;
Param.TokenWatchTime := 250;
Param.NodeNo := 1;
Param.NodeName := 'TargetNode';

result := FLnet.HFA_SetConfigParam(ADR(Param));
```

（2）HFA_LinkIn －FL-net ネットワークへの参加

＜名　　前＞　HFA_LinkIn

＜形　　式＞　HFA_RESULT result := HFA_LinkIn()

＜機能説明＞

FL-net ネットワークへ参加します。設定値を変更する場合は、以下の関数で設定します。

・設定値一括変更･･･HFA_SetConfigParam 関数

＜戻り値＞

以下の値を返します。

No.	値	内容
1	0	正常終了
2	-1	引数異常
3	4	既に参加中
4	7	既に参加待機中
5	-100	システムエラー。以下の原因が考えられます。 ・タスク生成エラー

＜使用例＞

ST 言語にて、FL-net ネットワークに参加する例です。

```
【変数宣言部】
PROGRAM PLC_PRG
VAR
      Param : HFA_CONFIG_PARAM;
      FLnet : HFLnet;
      result : HFA_RESULT;
END_VAR

【処理部】
// Set Param
// …( (1)HFA_SetConfigParam の＜使用例＞参照 )

// Set Config
result := FLnet.HFA_SetConfigParam(ADR(Param));

// Link in
IF result = HFA_RESULT.Normal THEN
       result := FLnet.HFA_LinkIn();
END_IF
```

（3）HFA_LinkOut －FL-net ネットワークからの離脱

＜名　　前＞　HFA_LinkOut

＜形　　式＞　HFA_RESULT result := HFA_LinkOut()

＜機能説明＞

FL-net ネットワークから離脱します。

＜戻り値＞

以下の値を返します。

No.	値	内容
1	0	正常終了
2	2	自ノード未参加（参加待機中の場合は含まない）

＜使用例＞

ST 言語にて、FL-net ネットワークから離脱する例です。

```
【変数宣言部】
PROGRAM PLC_PRG
VAR
    FLnet : HFLnet;
    result : HFA_RESULT;
END_VAR

【処理部】
// Link out
result := FLnet.HFA_LinkOut();
```

（４）HFA_ReadCommon1 －コモンメモリ領域(領域 1、領域 2)からのデータ読出し
HFA_ReadCommon2

＜名　　前＞　HFA_ReadCommon1(領域 1 の場合)／HFA_ReadCommon2 (領域 2 の場合)

＜形　　式＞　HFA_RESULT result := HFA_ReadCommon1(StartAddr, Words, Data)

```
VAR_INPUT
  StartAddr     : WORD;
  Words         : WORD;
  Data          : POINTER TO BYTE;
END_VAR                    ※ 領域 2 用は、関数名以外の形式共通
```

＜機能説明＞

コモンメモリ領域（領域 1、領域 2）のデータを読出します。領域のアドレス指定は、絶対アドレスとし、コモンメモリ全体のアドレス（領域 1 の場合：16#0000～16#01FF、領域 2 の場合：16#0000～16#1FFF）を指定します。

StartAddr　：　読出し開始アドレス。ワード単位での指定。
Words　　：　読出しサイズ。ワード単位での指定。
Data　　：　読出し先バッファの先頭ポインタ。
(注）呼び出し元 PLC プログラムでは、必ず読出しサイズ以上の領域を確保してください。

＜戻り値＞

以下の値を返します。

No.	値	内容
1	0	正常終了
2	-1	引数異常

留意事項

・FL-net ネットワーク未参加状態でコモンメモリを読出した場合、内部メモリの値を読出します。

＜使用例＞

ST 言語にて、コモンメモリ領域 1 のデータを読出す例です。

```
【変数宣言部】
PROGRAM PLC_PRG
VAR
        FLnet : HFLnet;
        Addr : WORD;
        Words : WORD;
        Data : ARRAY [0..100] OF BYTE;
        result : HFA_RESULT;
END_VAR

【処理部】
// Set address and size
Addr := 0;
Words := 10; // 10[word] = 20[byte]

// Read common memory area 1
result := FLnet.HFA_ReadCommon1(Addr, Words, ADR(Data));
```

（５）HFA_WriteCommon1 －コモンメモリ領域(領域 1、領域 2)へのデータ書込み
HFA_WriteCommon2

＜名　　前＞HFA_WriteCommon1(領域 1 の場合)／HFA_WriteCommon2 (領域 2 の場合)

＜形　　式＞HFA_RESULT result := HFA_WriteCommon1(StartAddr, Words, Data)

```
VAR_INPUT
  StartAddr   : WORD;
  Words       : WORD;
  Data        : POINTER TO BYTE;
END_VAR
```

※ 領域 2 用は、関数名以外の形式共通

＜機能説明＞

コモンメモリ領域（領域 1、領域 2）にデータを書込みます。書込み可能な範囲は、自ノード送信領域（HFA_SetConfigParam 関数で設定）のみです。自ノード送信領域以外に書込みを指定した場合、戻り値=引数異常となり、コモンメモリは更新されません。領域のアドレス指定は、絶対アドレスとし、コモンメモリ全体のアドレス（領域 1 の場合：16#0000～16#01FF、領域 2 の場合：16#0000～16#1FFF）を指定します。

StartAddr　：　先頭アドレス。ワード単位での指定。
Words　　：　書込みサイズ。ワード単位での指定。
Data　　：　書込みデータの先頭ポインタ。

(注)呼び出し元 PLC プログラムでは、必ず書込みサイズ以上の領域を確保してください。

＜戻り値＞

以下の値を返します。

No.	値	内容
1	0	正常終了
2	-1	引数異常

留意事項

・FL-net ネットワーク未参加状態でコモンメモリに書込んだ場合、内部メモリの値が更新されます。

＜使用例＞

ST 言語にて、コモンメモリ領域 1 にデータを書き込む例です。

【変数宣言部】

```
PROGRAM PLC_PRG
VAR
        FLnet : HFLnet;
        Addr : WORD;
        Words : WORD;
        Data : ARRAY [0..100] OF BYTE;
        result : HFA_RESULT;
END_VAR
```

【処理部】

```
// Set address and size
Addr := 0;
Words := 10; // 10[word] = 20[byte]

// To set up some data
Data[0] := 1;

// Write common memory area 1
result := FLnet.HFA_WriteCommon1(Addr, Words, ADR(Data));
```

（6）HFA_GetNodeStatus －ノード毎の管理情報パラメータ読出し

＜名　　前＞ HFA_GetNodeStatus

＜形　　式＞ HFA_RESULT result := HFA_GetNodeStatus (NodeNo, Node)

```
VAR_INPUT
  NodeNo        : BYTE;
  Node          : POINTER TO HFA_NODE;
END_VAR
```

＜機能説明＞

ノード毎の管理情報パラメータを読出します。読出しを行う対象ノードに応じて、取得可能な項目が異なります。

NodeNo　：　読出し対象ノード番号（0～254）。
　　0 を指定することで、自ノードの情報を読出すことができます。

Node　　：　ノード情報

＜戻り値＞

以下の値を返します。

No.	値	内容
1	0	正常終了
2	-1	引数異常
3	2	自ノード未参加
4	5	対象ノード未参加

＜使用例＞

ST 言語にて、ノード管理情報を読出す例です。

```
【変数宣言部】
PROGRAM PLC_PRG
VAR
    FLnet : HFLnet;
    Node : HFA_NODE;
    NodeNo : BYTE;
    result : HFA_RESULT;
END_VAR

【処理部】
// When the node number is 0, it means its own node
NodeNo := 0;

// Get node status
result := FLnet.HFA_GetNodeStatus(NodeNo, ADR(Node));
```

【HFA_NODE 構造体】

No.	名称	型	値の範囲	内容
1	VendorName	STRING(10)	HITACHI-IP（固定）	ベンダ名称
2	MakerType	STRING(10)	S-763A-97P（固定）	製造業者形式
3	NodeName	STRING(10)	ASCII 文字で最大 10 文字	ノード名称
4	Common1Addr	WORD	16#0000～16#01FF	コモンメモリ領域 1 先頭アドレス（ワード単位）
5	Common1Words	WORD	16#0000～16#0200	コモンメモリ領域 1 データサイズ（ワード単位）
6	Common2Addr	WORD	16#0000～16#1FFF	コモンメモリ領域 2 先頭アドレス（ワード単位）
7	Common2Words	WORD	16#0000～16#2000	コモンメモリ領域 2 データサイズ（ワード単位）
8	TokenTimeout	BYTE	1～255	トークン監視時間（1ms 単位）
9	MaxRefreshCycle	WORD	0～65535	リフレッシュサイクル許容時間（1ms 単位）
10	RefreshCycle	WORD	0～65535	リフレッシュサイクル実測値（1ms 単位）
11	MinFrameInterval	BYTE	0～50	最小許容フレーム間隔（100μs 単位）
12	UpperStatus	WORD	次ページ参照	上位層の状態
13	LinkStatus	BYTE	次ページ参照	FA リンクの状態
14	ProtocolVersion	BYTE	16#80 固定	プロトコルタイプ
15	MyNodeStatus	WORD	次ページ参照	自ノードの状態

以降に、HFA_NODE 構造体の［No.12 上位層の状態］、［No.13 FA リンクの状態］、［No.15 自ノードの状態］の詳細を示します。

［No.12 上位層の状態(Upper Layer Status)］

ビット	値	内容
0～11	全ビット 0（固定※）	上位層のエラー内容（上位層で定義）
12	0（固定）	予備
13,14	00（固定※）	上位層のエラー情報 00：NORMAL 01：WARNING 10：ALARM 11：ALARM
15	0 または 1	上位層の動作情報（HFA_SetControlEquipment 関数の設定値） 0：STOP 1：RUN

(※) 本ライブラリでは、上位層に関する情報を設定するインタフェースは非サポートのため 0 固定です。

ビット列で表すと以下となります（‘X’はユーザーによる設定値で決定)。

ビット	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
値	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

［No.13 FA リンクの状態(FA Link Status)］

ビット	値	内容
0	0 または 1 （HFA_GetNodeStatus 関数でノード管理情報を読出した場合。ただし、ネットワークに送信するフレームについては 0 固定）	ノードの参加状態 0：離脱 1：参加中
1	0 または 1 （HFA_GetNodeStatus 関数でノード管理情報を読出した場合。ただし、ネットワークに送信するフレームについては 0 固定）	ノードの通信無効検知の有無 0：検知なし 1：検知あり
2,3	0（固定）	予備
4	0（設定用 I/F 非サポートのため 0 固定）	上位層動作信号エラー 0：エラーなし 1：エラーあり
5	0 または 1	コモンメモリの送信領域の有無 0：無効（送信領域無[受信専用]） 1：有効（送信領域有）
6	1（未完了のケースがないため 1 固定）	コモンメモリ(アドレス・サイズ)設定完了 0：未完了 1：完了
7	0 または 1	アドレス重複検知状態 0：重複なし 1：重複あり

ビット列で表すと以下となります（‘X’はユーザーによる設定値で決定)。

ビット	7	6	5	4	3	2	1	0
値	X	1	X	0	0	0	X	X

［No.15 自ノードの状態］

ビット	値	内容
0	0 または 1	ノード番号重複フラグ 0：重複なし 1：重複あり
1	0 または 1	トークン監視時間エラー 0：エラーなし 1：エラーあり
2	0 または 1	受信待ち状態 0：待ちなし 1：受信待ち
3	0 または 1	初期化エラー 0：エラーなし 1：エラーあり
4	0 または 1	参加状態 0：離脱 1：参加中
5～7	000（固定）	予備
8～15	自ノードステータス番号	FA リンクプロトコル仕様の状態遷移番号。 詳細は、以下参照。

ビット列で表すと以下となります（‘X’はユーザーによる設定値で決定）。

ビット	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
値	X	X	X	X	X	X	X	X	0	0	0	X	X	X	X	X

※自ノードステータス番号の値

値	状態	内容
1	上位層からの初期化待ち	ネットワーク加入に必要なパラメータの設定を上位層から設定されるのを待つ状態。
2	加入トークン検出待ち	ネットワークが稼働中かチェックを行う状態。
3	トリガ送信または、受信待ち	新規加入の同期をとるためのトリガによる同期を取るための状態。
4	参加要求受付	新規加入時、トリガによって同期をとった参加要求受付時間に参加要求フレームによる ネットワーク情報の確立を行う状態。
5	トークン周回の 3 周待ち	途中加入時、トークンによってネットワークの情報を収集する状態。
6	参加要求送信待ち	途中加入時、ネットワークの情報の収集が終了後、参加要求フレームの送信を行う状態。
7	トークン待ち	自ノード宛のトークンの受信を待ち、他ノードの監視を行う状態。
8	トークン保持	自ノード宛のトークンを受信してから、次ノード宛のトークンを送信するまでの状態。

（7）HFA_GetNetworkStatus －ネットワークステータスの読出し

＜名　　前＞　HFA_GetNetworkStatus

＜形　　式＞　HFA_RESULT result := HFA_GetNetworkStatus(Network, Node)

```
VAR_INPUT
  Network      : POINTER TO HFA_NETWORK;
  Node         : POINTER TO ARRAY [1..254] OF BYTE;
END_VAR
```

＜機能説明＞

ネットワークステータスを読出します。FL-net ネットワーク未参加の状態で本関数をコールした場合、戻り値=自ノード未参加となり、ステータスの読出しは行いません。

Network　：ネットワーク管理情報パラメータ。

Node　　：ノード参加状態。FL-net ネットワークへのノード参加状態を、ノード番号の昇順（1～254）で 1 バイト毎に格納します。BYTE 型配列の添え字がノード番号と対応し、参加状態の値は以下となります。

- 0=未参加
- 1=参加中
- 2=通信無効検知（未参加）

(注)呼び出し元 PLC プログラムでは、必ず 254 バイト以上の領域を確保してください。

＜戻り値＞

以下の値を返します。

No.	値	内容
1	0	正常終了
2	-1	引数異常
3	2	自ノード未参加

＜使用例＞

ST 言語にて、ネットワークステータスを読出す例です。

```
【変数宣言部】
PROGRAM PLC_PRG
VAR
    FLnet : HFLnet;
    Network : HFA_NETWORK;
    Nodes : ARRAY [1..254] OF BYTE;
    result : HFA_RESULT;
END_VAR

【処理部】
// Get network status
result := FLnet.HFA_GetNetworkStatus(ADR(Network), ADR(Nodes));
```

【HFA_NETWORK 構造体】

No.	名称	型	値の範囲	内容
1	TokenOwner	BYTE	1～254	トークン保持ノード番号
2	MinFrameInterval	BYTE	0～50	最小許容フレーム間隔 （100μs 単位）
3	MaxRefreshCycle	WORD	0～65535	リフレッシュサイクル許容時間 （1ms 単位）
4	RefreshCycle	WORD	0～65535	リフレッシュサイクル測定時間（現在値） （1ms 単位）
5	RefreshCycleHigh	WORD	0～65535	リフレッシュサイクル測定時間（最大値） （1ms 単位）
6	RefreshCycleLow	WORD	0～65535	リフレッシュサイクル測定時間（最小値） （1ms 単位）

（8）HFA_GetMyNodeLog －自ノードログ情報の読出し

＜名　　前＞ HFA_GetMyNodeLog

＜形　　式＞ HFA_RESULT result := HFA_GetMyNodeLog(pLog)

```
VAR_INPUT
  pLog   : POINTER TO HFA_LOG;
END_VAR
```

＜機能説明＞

自ノードのログ情報を読出します。

pLog　:ログ情報。

＜戻り値＞

以下の値を返します。

No.	値	内容
1	0	正常終了
2	-1	引数異常

＜使用例＞

ST 言語にて、自ノードのログ情報を読出す例です。

```
【変数宣言部】
PROGRAM PLC_PRG
VAR
    FLnet : HFLnet;
    MyLog : HFA_LOG;
    result : HFA_RESULT;
END_VAR

【処理部】
// Get log
result := FLnet.HFA_GetMyNodeLog(ADR(MyLog));
```

【HFA_LOG 構造体】

No.	名称	型	内容
1	Protocol	LOG_Protocol	プロトコル情報
2	Frame	LOG_Frame	フレーム情報
3	Cyclic	LOG_Cyclic	サイクリック情報
4	Message	LOG_Message	メッセージ情報
5	Ack	LOG_Ack	ACK 情報
6	Token	LOG_Token	トークン情報
7	Link	LOG_Link	リンク情報
8	Node	LOG_Node	参加ノード情報

【LOG_Protocol 構造体】

No.	名称	型	内容
1	SendData	DWORD	ソケット送信通算回数
2	SendSockErr	DWORD	ソケット送信エラー回数
3	SendNetErr	DWORD	Ether 送信エラー回数（未使用）
4	RecvData	DWORD	ソケット受信通算回数
5	RecvSockErr	DWORD	ソケット受信エラー回数
6	RecvNetErr	DWORD	Ether 受信エラー回数（未使用）

【LOG_Frame 構造体】

No.	名称	型	内容
1	SendToken	DWORD	トークン送信回数
2	SendCyclic	DWORD	サイクリックフレーム送信回数
3	SendPeerToPeer	DWORD	1 対 1 メッセージ送信回数
4	SendBroadcast	DWORD	1 対 n メッセージ送信回数
5	RecvToken	DWORD	トークン受信回数
6	RecvCyclic	DWORD	フレーム受信回数
7	RecvPeerToPeer	DWORD	1 対 1 メッセージ受信回数
8	RecvBroadcast	DWORD	1 対 n メッセージ受信回数

【LOG_Cyclic 構造体】

No.	名称	型	内容
1	RecvCyclicErr	DWORD	送受信エラー回数
2	CyclicAddrErr	DWORD	アドレスサイズエラー回数
3	CyclicCbnErr	DWORD	CBN エラー回数
4	CyclicTbnErr	DWORD	TBN エラー回数
5	CyclicBsizeErr	DWORD	Bsize エラー回数

【LOG_Message 構造体】

No.	名称	型	内容
1	Resend	DWORD	再送回数
2	RetryOut	DWORD	再送オーバー回数
3	RecvErr	DWORD	受信エラー回数
4	SeqVerErr	DWORD	通番バージョンエラー回数
5	SeqVerify	DWORD	通番再送認識回数

【LOG_Ack 構造体】

No.	名称	型	内容
1	AckErr	DWORD	ACK エラー回数
2	VersionErr	DWORD	通番バージョンエラー回数
3	SeqNoErr	DWORD	通番番号エラー回数
4	NodeNoErr	DWORD	ノード番号エラー回数（未使用）
5	TransactionCodeErr	DWORD	メッセージ番号（TCD）エラー回数

【LOG_Token 構造体】

No.	名称	型	内容
1	Conflict	DWORD	多重化回数
2	Destroy	DWORD	破棄回数
3	Retry	DWORD	再発行回数
4	KeepTimeout	DWORD	保持タイムアウト回数
5	WatchTimeout	DWORD	監視タイムアウト回数

【LOG_Link 構造体】

No.	名称	型	内容
1	ElapseTime	DWORD	通算稼働時間（秒単位）
2	WaitFrame	DWORD	フレーム待ち回数
3	LinkIn	DWORD	加入回数
4	LinkOut	DWORD	離脱回数
5	LinkOutBySkip	DWORD	離脱回数(ノードスキップによる)
6	AnotherLinkOut	DWORD	他ノード離脱回数

【LOG_Node 構造体】

No.	名称	型	内容
1	LinkIn	ARRAY[0..255] OF BYTE	参加認識ノード一覧

（9）HFA_ClearMyNodeLog －自ノードログ情報のクリア

＜名　　前＞　HFA_ClearMyNodeLog

＜形　　式＞　HFA_RESULT result := HFA_ClearMyNodeLog()

＜機能説明＞

自ノードのログ情報をクリアします。

＜戻り値＞

以下の値を返します。

No.	値	内容
1	0	正常終了

＜使用例＞

ST 言語にて、自ノードのログ情報をクリアする例です。

```
【変数宣言部】
PROGRAM PLC_PRG
VAR
      FLnet : HFLnet;
      result : HFA_RESULT;
END_VAR

【処理部】
// Clear log
result := FLnet.HFA_ClearMyNodeLog();
```

（１０） HFA_SetControlEquipment －自ノードの運転／停止状態の設定

＜名　　前＞　HFA_SetControlEquipment

＜形　　式＞　HFA_RESULT result := HFA_SetControlEquipment(RunMode)

```
VAR_INPUT
  RunMode   : BOOL;
END_VAR
```

＜機能説明＞

自ノードの運転／停止状態を設定します。

本関数がコールされない場合のデフォルト値は、運転状態となります。

RunMode　：　自ノードの運転／停止状態を設定する運転モード。

・FALSE＝停止状態

・TRUE＝運転状態

＜戻り値＞

以下の値を返します。

No.	値	内容
1	0	正常終了

＜使用例＞

ST 言語にて、自ノードの運転／停止状態を設定する例です。

```
【変数宣言部】
PROGRAM PLC_PRG
VAR
    FLnet : HFLnet;
    result : HFA_RESULT;
END_VAR

【処理部】
// Operate
result := FLnet.HFA_SetControlEquipment(TRUE);

// Stop
result := FLnet.HFA_SetControlEquipment(FALSE);
```

（１１）HFA_GetLinkInStatus － FL-net ネットワーク参加状態の読出し

＜名　　前＞ HFA_GetLinkInStatus

＜形　　式＞ HFA_LINK_STATUS result := HFA_GetLinkInStatus(NodeNo)

```
VAR_INPUT
  NodeNo   : BYTE;
END_VAR
```

＜機能説明＞

FL-net ネットワーク参加状態を読出します。

NodeNo ： 読出し対象ノード番号。（0～254）

0 を指定することで、自ノードの参加状態を読出すことができます。

＜戻り値＞

以下の値を返します。

No.	値	内容
1	0	参加待機中
2	1	参加中
3	2	離脱中

＜使用例＞

ST 言語にて、FL-net ネットワーク参加状態を読出す例です。

```
【変数宣言部】
PROGRAM PLC_PRG
VAR
    FLnet : HFLnet;
    status : HFA_LINK_STATUS;
END_VAR

【処理部】
// Get link status
status := FLnet.HFA_GetLinkInStatus(0);
```

4.3. ライブラリ関数使用の流れ

ライブラリ関数の使用例（コモンメモリ読み書き操作時）を以下に示します。

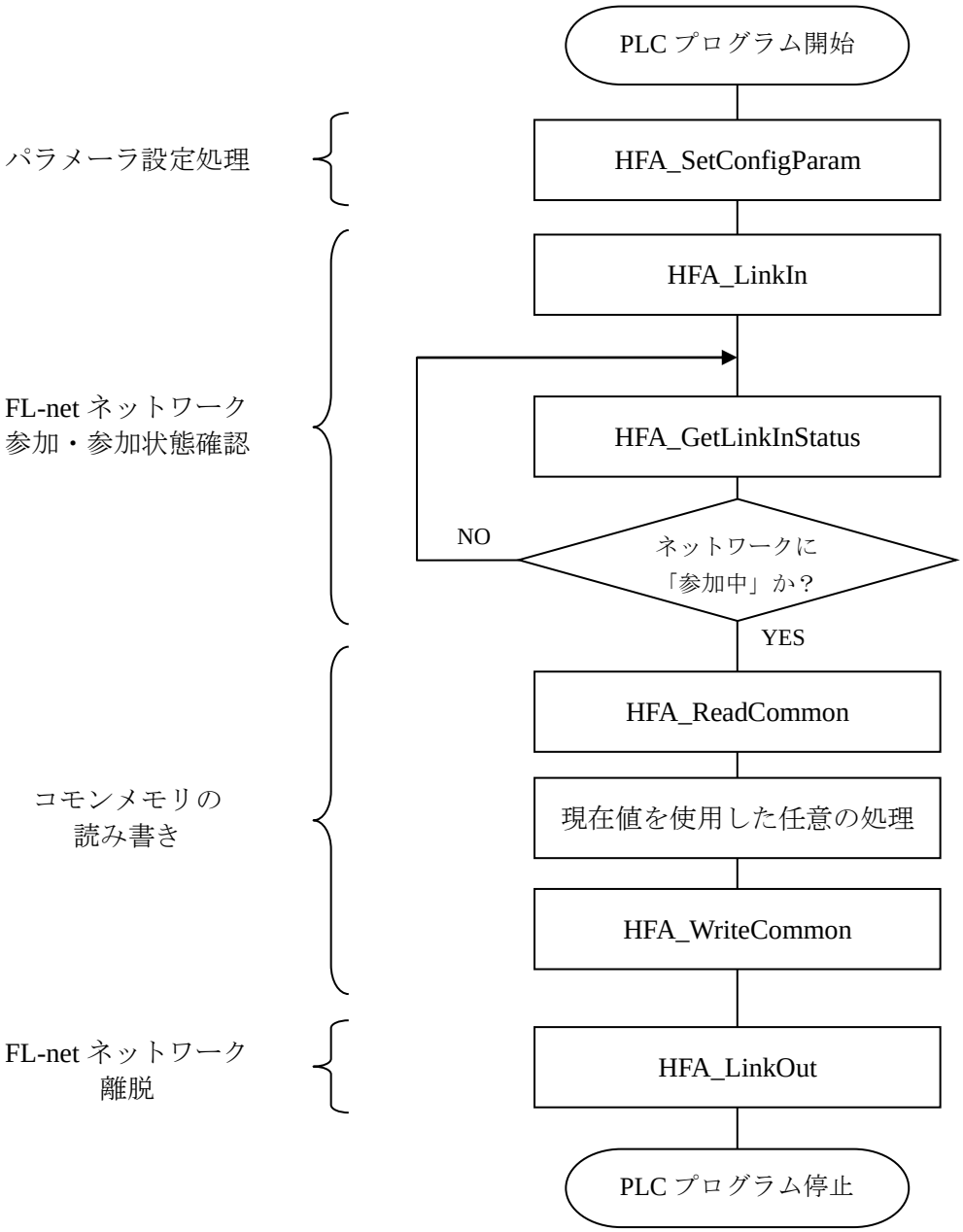


図 17 ライブラリ関数使用の流れ

第5章 プロファイル情報

「FL-net For CODESYS®」のプロファイル情報を以下に示します。

No.	パラメータ名称	名称文字 （長さ,文字）	データタイプ[型]	パラメータ内容 （長さ,内容）
1	デバイスプロファイル 共通仕様バージョン	6,“COMVER”	INTEGER	1,1
2	システムパラメータ 識別文字	2,“ID”	PrintableString	7,“SYSPARA”
3	システムパラメータ 改変番号	3,“REV”	INTEGER	1,0
4	システムパラメータ 変更日付	7,“REVDATE”	[INTEGER] [INTEGER] [INTEGER]	2,2019 1,9 1,1
5	デバイス種別	10,“DVCATEGORY”	PrintableString	8,“COMPUTER”
6	ベンダ名	6,“VENDOR”	PrintableString	10,“HITACHI-IP”
7	製品形名	7,“DVMODEL”	PrintableString	10,“S-763A-97P”