OMRON プログラマブルコントローラ
CJ シリーズ用プロバイダ

Version 1.2.8

ユーザーズ ガイド

February 6, 2023

【備考】

【改版履歴】

バージョン	日付	内容
1.0.0	2016-03-02	初版.
1.0.1	2016-05-27	エラーコード変更. 4 バイト以上のデータを取得した際のエンディアン変換を CX-Programmer に合わせ修正.
1.02	2016-06-22	VT オプションの省略時デフォルト値を変更. BIT → ビットアドレス指定時:BIT, チャネルアドレス指定時:I2
	2016-08-10	変数の読み書きに伴い発生する通信のパケットサイズが大きくなった場合に通信が失敗するケースがある旨記載.
	2016-10-31	Conn オプションで ETH 指定できるよう変更.
1.1.0	2018-06-08	Retry オプション追加.
1.2.0	2018-07-05	TCP 対応.
1.2.1	2019-02-07	TCP のパケット分割対応.
1.2.2	2019-08-19	拡張データメモリ(EM バンク 10～18)の読み書き処理 バグ修正
1.2.3	2020-01-29	互換用の BitOpt オプション追加(付録 C).
1.2.4	2020-06-22	排他制御処理修正. NSJ コントローラの接続する際の注意点を追加. Conn オプションに Src IP Address, Src Port No 要素を明記. 中継異常応答, エラー応答からの詳細エラー手順を明記. 付録 E 追加
	2020-10-01	動作確認機種の追加
1.2.5	2021-02-09	切断処理の改善.
1.2.6	2021-06-18	想定外データ取得時のエラーコードを追加
	2022-05-30	パフォーマンス計測結果を追加
1.2.7	2022-08-02	チャネル指定時の値書込み処理の修正 Mode オプション STATUS の説明分を修正
1.2.8	2023-02-06	パケット分割が発生するとメモリ破壊が発生する可能性がある障害を修正.

【動作確認機器】

機種	バージョン	注意事項
CJ2H	Ver1.2	
CS1H-CPU64H		
CS1W-EIP21		EtherNet/IP ユニット

目次

1. はじめに

本書は，OMRON 製 PLC(CJ シリーズ)に対しデータの書込み/読出しを行う CAO プロバイダのユーザーズガイドです．本書で扱う CAO プロバイダ(CaoProvOmronCJ.dll)を CJ プロバイダと呼びます．

第 2 章に CJ プロバイダの概要，変数の詳細を記載しています．

CJ プロバイダで実装している通信コマンドの対応状況及びデータ列については，通信先となる PLC(CJ シリーズ)に依存します．通信の詳細については OMRON の"sbca-304r-30_cs1_cj1_cp1_com_cmd.pdf"並びに"sbca-350l-22_cj2h-cpu6-eip_cj2m-cpu.pdf"を参照してください．

2. プロバイダの概要

2.1. 概要

CJ プロバイダは，OMRON 製 PLC(CJ シリーズ)に対し Ethernet(UDP/TCP)接続で FINS コマンドを用いてデータの書込み/読出しを行う CAO プロバイダです．そのファイル形式は DLL(Dynamic Link Library)であり，CAO エンジンから使用時に動的にロードされます．CJ プロバイダを使用するにあたっては ORiN2SDK をインストールするか，下表を参照して手作業でレジストリ登録を行う必要があります．

表 2-1 CJ プロバイダ

ファイル名	CaoProvOmronCJ.dll
ProgID	CaoProv.OMRON.CJ
レジストリ登録	regsvr32 CaoProvOmronCJ.dll
レジストリ登録の抹消	regsvr32 /u CaoProvOmronCJ.dll

2.2. メソッド・プロパティ

2.2.1. CaoWorkspace::AddController メソッド

CJ プロバイダは AddController 時に通信用の接続パラメータを参照し，通信の接続を行います．(TCP クライアントとして動作します)

書式　AddController(<bstrCtrlName:BSTR>,<bstrProvName:BSTR>,
<bstrPCName:BSTR>,<bstrOption:BSTR>))

bstrCtrlName : [in] コントローラ名
bstrProvName : [in] プロバイダ名．固定値 =" CaoProv.OMRON.CJ"
bstrPcName : [in] プロバイダの実行マシン名
bstrOption : [in] オプション文字列

以下にオプション文字列に指定するリストを示します．

表 2-2 CaoWorkspace::AddController のオプション文字列

オプション	説明
Conn=<接続パラメータ>	必須．通信形態と接続パラメータ． (参照 2.2.1.1)
ConnTimeout[=<タイムアウト時間>]	TCP 接続時のタイムアウト時間．(ミリ秒) (デフォルト:500)
Timeout[=<タイムアウト時間>]	送受信時のタイムアウト時間．(ミリ秒) （デフォルト:3000）
FinsParam=<DNA>:<DA1>:<DA2> [:<GCT>[:<SNA>[:<SA1>[:<SA2>:[<SID>]]]]]	必須(一部省略可)．FINS コマンドのヘッダ構成パラメータ． (参照 2.2.1.2)
Retry [=<リトライ回数>]	送受信時の通信リトライ回数の設定をします． (デフォルト:0)
KeepAlive[=<キープアライブ時間>]	TCP 通信時のキープアライブ時間の設定をします．(ミリ秒) (デフォルト:60000)
PutOpt:[=<書込み時動作>]	チャネルアドレスへ BIT, BOOL, UI1, I1 で書き込む際の，データの扱い方法を指定します． (参照 付録 D)

2.2.1.1. Conn オプション

以下に Conn オプションの接続パラメータ文字列を示します. ここで角括弧("[ ]")内は省略可能を示します. また, 各パラメータの解説中の下線部はオプションを指定しなかったときのデフォルト値になります.

(CJ プロバイダでは Ethernet デバイスの UDP/TCP 接続をサポートします)

Ethernet デバイス

"Conn=ETH:<Dest IP Address>[:<Dest Port No>[:<Src IP Address>[:<Src Port No>]]]"

"Conn=UDP:<Dest IP Address>[:<Dest Port No>[:<Src IP Address>[:<Src Port No>]]]"

"Conn=TCP: <Dest IP Address>[:<Dest Port No>[:<Src IP Address>[:<Src Port No>]]]"

< Dest IP Address >	:	接続先 IP アドレス. 例:"127.0.0.1", "192.168.0.1"
<Dest Port No>	:	接続先ポート番号. 9600, 5006, 5007, …任意指定可能
<Src IP Address>	:	接続元 IP アドレス. (複数 NIC 用途) IP アドレスを自動で判別する場合は"255.255.255.255"を指定してください. 例:"127.0.0.1", "192.168.0.1", "255.255.255.255"
<Src Port No>	:	接続元ポート番号. (複数 NIC 用途) [1][2] 例:0, 9600, 5006, 5007, …任意指定可能

[1] NSJ コントローラと接続する際は PC 側とポートが同じである必要がありますので, Dest Port No と Src Port No を合わせてください. NSJ 用拡張ユニット(NSJW-ETN21)を使用する場合は Src Port No を設定する必要はありません.

[2] ネットワーク先の PLC に接続する場合は Dest Port No と Src Port No を合わせてください. ネットワーク先の PLC に接続する場合は付録 E を参照してください.

2.2.1.2. FinsParam オプション

以下にFinsParamオプションの接続パラメータ文字列を示します. ここで角括弧("[ ]")内は省略可能を示します. また, 各パラメータの解説中の下線部はオプションを指定しなかったときのデフォルト値になります.

"FinsParam=<DNA>:<DA1>:<DA2>[:<GCT>[:<SNA>[:<SA1>[:<SA2>:[<SID>]]]]]"

<DNA> : 相手先ネットワークアドレス(Destination network address)
0 : 自ネットワーク
1～127 :相手先ネットワークアドレス

<DA1> : 相手先ノードアドレス(Destination node number)
0 :自 PLC 内通信
1～32 : Controller Link ネットワークの場合
1～254 : Ethernet の場合
255 : 一斉同報データ送信

<DA2> : 相手先号機アドレス(Destination unit address)
0 : CPU ユニット
254 : 該当 Controller Link ユニット/Ethernet ユニット
16～31 : CPU 高機能ユニット
225 : INNER ボード

<GCT> : 許容ブリッジ通過数(Gateway Count)
2, 7

<SNA> : 発信元ネットワークアドレス(Source network address)
0 : 自ネットワーク
1～127 : 相手先ネットワークアドレス

<SA1> : 発信元ノードアドレス(Source node number)
0 : 自 PLC 内通信
1～32 : Controller Link ネットワークの場合
1～254 : Ethernet の場合

<SA2> : 発信元号機アドレス(Source unit address)
0 : CPU ユニット
16～31 : CPU 高機能ユニット

<SID> : サービス ID(Service ID)
0～255 : 任意の値

2.2.2. CaoController::AddVariable メソッド

CaoController クラスの AddVariable メソッドは，PLC(CJ シリーズ)に対しデータの書込み/読出しを行うための変数オブジェクトを作成するためのメソッドです．

本メソッドでは変数名に 2.2.2.1 に記す書式で指定することで PLC(CJ シリーズ)内のアクセス対象とする IO メモリのエリア種別/アドレスを決定します．

また，オプションでデータの型変換や書込み/読み出しするデータ数の指定が可能です．[(3)]

書式　AddVariable(<bstrVariableName:VT_BSTR>[,<bstrOption:VT_BSTR>])

<bstrVariableName> : [in] 変数名

<bstrOption> : [in] オプション文字列

以下にオプション文字列に指定するリストを示します．

表 2-3 CaoController::AddVariable のオプション文字列

オプション	説明
Mode[=<アクセスモード>]	タイマ，カウンタ，タスクフラグにアクセスする際のモードを指定． (参照 2.2.2.2)
VT[=<変数型>]	Put/Get するデータ型を指定． (参照 2.2.2.3)
Elem[=<要素数>]	Put/Get するデータの要素数を指定． データ型が VT_BSTR の場合は文字列のバイト数指定になる． 要素数は 10 進数での指定の他，16 進数での指定も可能．(0x0A, &h0A, 0AH) （デフォルト:1）
Array[=< True or False >]	Put/Get するデータが 1 要素の場合に配列として扱うかを指定． （デフォルト:False）
ID[=<ID 初期値>]	変数の ID 初期値を指定．(10 進数) （デフォルト:0）

[3] データの読み書きに伴い発生する通信のパケットサイズが大きくなると，FINS コマンドの制限で通信が失敗する場合があります．1 度に読み書きするデータの要素数を減らすことで回避できます．

2.2.2.1. 変数名によるアクセス対象のエリア種別/アドレス指定

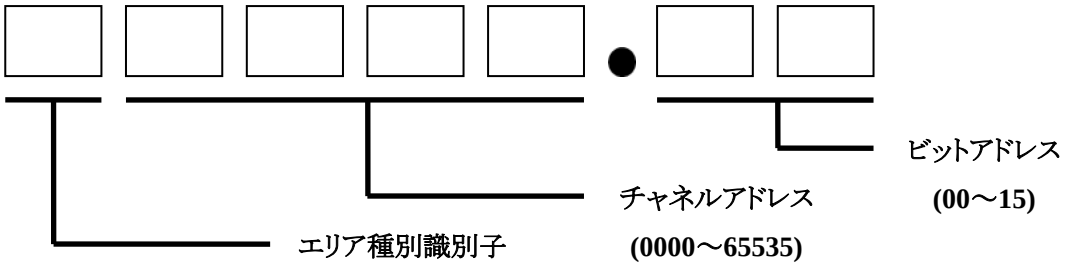
エリア種別識別子(アルファベット 0～4 文字), チャネルアドレス(10 進数の数値 1～4 桁), ビットアドレス(10 進数の数値 1～2 桁)で構成されます.

アクセス対象 IO メモリのデータ種類がビットの場合はチャネルアドレスの後に"."(ドット)を挟んでビットアドレスを指定します. チャネル(ワード)の場合はビットアドレスを指定しません.

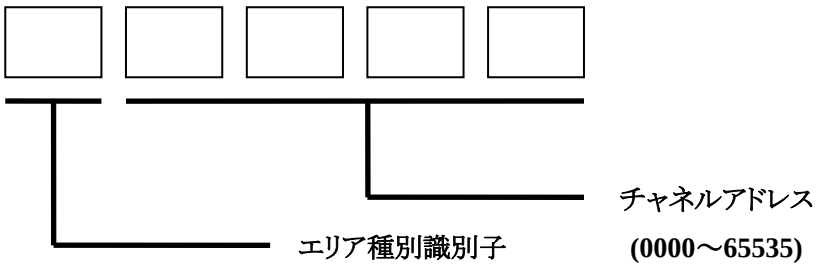
タイマ, カウンタ, タスクフラグに関してはチャネル(ワード)アドレス指定書式のみを使用し, データ種類は後述の Mode オプションによって指定します. (参照 2.2.2.2)

メモリ種別及びアドレスを指定するための書式を以下に示します.

ビットアドレス指定書式



チャネル(ワード)アドレス指定書式



IO メモリエリア種別毎の識別子一覧を以下に示します.

表 2-4 エリア種別識別子一覧

エリア種別		識別子
チャネル I/O	CIO	なし
内部補助リレー	WR	W
保持リレー	HR	H
特殊補助リレー	AR	A
タイマ	TIM	T
カウンタ	CNT	C
データメモリ	DM	D
拡張データメモリ	EM バンク 0	E0_
	EM バンク 1	E1_
	EM バンク 2	E2_
	EM バンク 3	E3_
	EM バンク 4	E4_
	EM バンク 5	E5_
	EM バンク 6	E6_
	EM バンク 7	E7_
	EM バンク 8	E8_
	EM バンク 9	E9_
	EM バンク A	EA_
	EM バンク B	EB_
	EM バンク C	EC_
	EM バンク D	ED_
	EM バンク E	EE_
	EM バンク F	EF_
	EM バンク 10	E10_
	EM バンク 11	E11_
	EM バンク 12	E12_
	EM バンク 13	E13_
	EM バンク 14	E14_
	EM バンク 15	E15_

	EM バンク 16	E16_
	EM バンク 17	E17_
	EM バンク 18	E18_
タスクフラグ	TK	TK

2.2.2.2. Mode オプション

タイマ, カウンタ, タスクフラグを対象とする際にデータ種類を指定するために使用します.

（デフォルト:NORMAL）

NORMAL : データ種別をビットとして扱う.
タイマ, カウンタ, タスクフラグ以外の場合は本設定を無視する.

CURRENT : タイマ, カウンタ時のみ有効.
データ種別を現在値として扱う.

STATUS : タスクフラグ時のみ有効.
データ種別をビットとして扱う.

2.2.2.3. VT オプション

Put/Get するデータ型を指定します.

指定可能なデータ型の一覧を以下に示します.

(デフォルト:ビットアドレス指定時=BIT, チャネルアドレス指定時=I2)

表 2-5 VT オプションで指定可能なデータ型一覧

VT	データ型	使用可能種類	説明
BIT	VT_UI1	ビット/チャネル	データを 0/1 の 2 値に変換して書込み/読出し Val==0:0, Val!=0:1
BOOL	VT_BOOL	ビット/チャネル	データを 0/1 の 2 値に変換して書込み/読出し Val==VARIANT_FALSE:0, Val= VARIANT_TRUE:1
BSTR	VT_BSTR	チャネル	BSTR を ASCII として書込み/読出し
I1	VT_I1	チャネル	1 バイトデータとして書込み/読出し
I2	VT_I2	チャネル	2 バイトデータとして書込み/読出し
I4	VT_I4	チャネル	4 バイトデータとして書込み/読出し
I8	VT_I8	チャネル	8 バイトデータとして書込み/読出し
UI1	VT_UI1	チャネル	1 バイトデータとして書込み/読出し
UI2	VT_UI2	チャネル	2 バイトデータとして書込み/読出し
UI4	VT_UI4	チャネル	4 バイトデータとして書込み/読出し
UI8	VT_UI8	チャネル	8 バイトデータとして書込み/読出し
R4	VT_R4	チャネル	4 バイトデータとして書込み/読出し

R8	VT_R8	チャネル	8 バイトデータとして書込み/読出し

2.2.2.4. Elem オプション

ELEM オプションで指定できる最大値はデータの種類と VT オプションの組み合わせで決まります.

ELEM オプションの最大値は以下の通りです.

データ種類	VT	PutOpt	読み込み最大長	書き込み最大長
ビット	BIT,BOOL	TRUE/FALSE	1998	1994
チャネル	BIT,BOOL,I1,UI1	FALSE	1998	997
		TRUE	1998	1994
	I2,UI2	TRUE/FALSE	999	997
	I4,UI4,R4	TRUE/FALSE	499	498
	I8,UI8	TRUE/FALSE	249	249

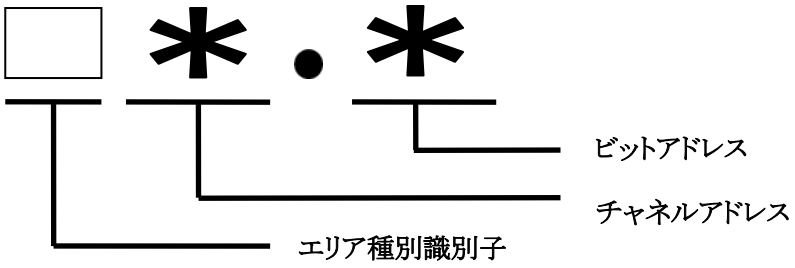
2.2.2.5. アドレスのワイルドカード指定

変数を登録する際にアクセス先のアドレスを固定化せず，後から動的に変更できるようアドレス部分を"*"（アスタリスク）でワイルドカード指定することが可能です．

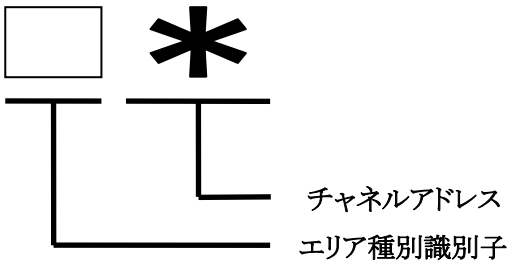
アドレスは put/get 時の変数の ID で決定します． (参照 2.2.5)

アドレスをワイルドカード指定するための書式を以下に示します．

ビットアドレス指定書式



チャネル（ワード）アドレス指定書式

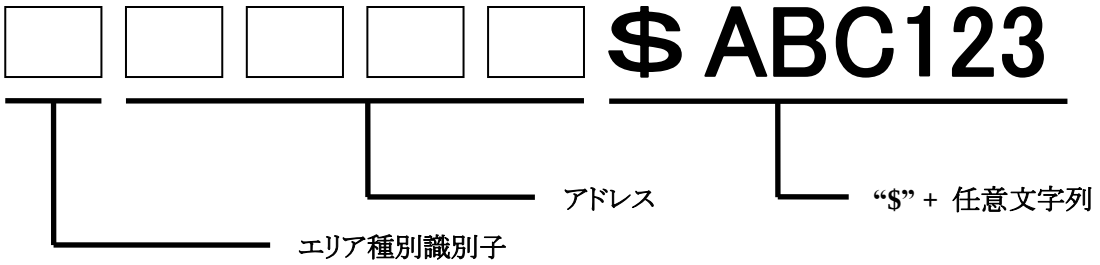


2.2.2.6. 変数名への任意文字列付与

アクセスするエリア種別やアドレスを変えずに複数変数を登録(オプションのみ変更したい場合等に有用)するために"$"(ダラー)を挟んで任意の文字列を付与することが可能です.

任意文字列を付与するための書式を以下に示します.

ビット/チャネル(ワード)アドレス共通指定書式



2.2.3. CaoVariable:put_Value プロパティ

引数渡された値をオプション指定に従い変換した後，変数名で指定したメモリ領域に対し書込みをするFINS コマンド（コマンドコード：01 02）を送出します．

2.2.4. CaoVariable:get_Value プロパティ

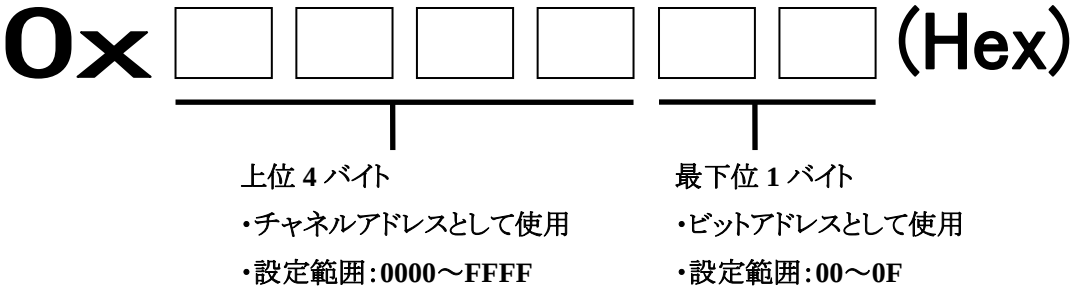
変数名で指定したメモリ領域からオプションで指定されたサイズになるように読出しするFINSコマンド（コマンドコード：01 01）を送出します．読み出した結果はオプションで指定された型に変換し返されます．

2.2.5. CaoVariable:put_ID プロパティ

変数の ID を設定します．

この ID は変数名によるアドレスをワイルドカード指定した際に参照されます．(参照 2.2.2.5)

ワイルドカード指定されていない場合は単純な ID として機能します．ワイルドカード指定された場合は ID 値は以下の意味と設定範囲を持ちます．



※ チャネルアドレスのみ指定したい場合は最下位 1 バイトを 0x00 として ID をセット

2.2.6. CaoVariable:get_ID プロパティ

現在設定されている変数の ID を取得します．

2.3. 変数一覧

2.3.1. CaoController クラス

表 2-6 CaoController クラス ユーザ変数一覧

変数名	データ型	説明	属性	
			get	put
任意	変数型依存	PLC(CJ シリーズ)内のアクセス対象とする IO メモリのエリア種別/アドレスを指定する. (参照 2.2.2)	○	○

表 2-7 CaoController クラス システム変数一覧

変数名	データ型	説明	属性	
			get	put
@MAKER_NAME	VT_BSTR	メーカ名="OMRON"を返す.	○	－
@VERSION	VT_BSTR	バージョン情報.	○	－
@LAST_ERROR	VT_VARIANT | VT_ARRAY	直前の通信異常応答詳細情報. [0] VT_UI2 ： 発生コマンドコード [1] VT_BOOL： 中継異常 [2] VT_BOOL： 本体停止異常 [3] VT_BOOL： 本体継続異常 [4] VT_UI1 ： メインレスポンスコード [5] VT_UI1 ： サブレスポンスコード [6] VT_UI1 ： 異常発生ネットワークアドレス [7] VT_UI1 ： 異常発生ノードアドレス	○	－

2.4. エラーコード

CJ プロバイダでは，以下の固有エラーコードが定義されています.

「中継異常発生」,「エラー応答」が返された場合，システム変数"@LAST_ERROR"を読み出すことで直前の通信のエラー情報を取得することが可能です.

OriN2 共通エラーについては,「OriN2 プログラミングガイド」のエラーコードの章を参照してください.

表 2-8 固有エラーコード

エラー名	エラー番号	説明
中継異常発生	0x80100000	通信経路に異常があり，中継異常の応答があった場合に返されます.
エラー応答	0x80100001	接続先からエラー応答を受けた場合に返されます.
受信データ欠落	0x80100002	受信データが欠落しており，解析ができなかった場合に返されます.
異なるコマンド応答	0x80100003	送信した FINS コマンドと異なるコマンド応答を受けた場合に返されます.
不正なコマンド	0x80100004	不正な FINS/TCP コマンドを受信した場合に返されます.
不正なノード	0x80100005	不正なノードアドレスを指定した場合に返されます.
不正なヘッダ	0x80100100	FINS/TCP ヘッダ部のヘッダが'FINS'(ASCII コード)でない場合に返されます.
データ長範囲外	0x80100101	FINS/TCP ヘッダ部のデータ長が範囲外である場合に返されます.
サポート外のコマンド	0x80100102	FINS/TCP ヘッダ部のコマンドがサポート外である場合に返されます.
全コネクション使用中	0x80100103	コネクションが全て使用中である場合に返されます.
ノード接続中	0x80100104	指定したノードアドレスとはすでに接続中である場合に返されます.
プロテクトされたノード	0x80100105	プロテクトされたノードアドレスに対して，指定外の IP アドレスでアクセスした場合に返されます.
ノード範囲外	0x80100106	クライアントのノードアドレスが範囲外である場合に返されます.

同一ノード	0x80100107	クライアントとサーバで，同一のノードアドレスを使用した場合に返されます．
想定外エラー	0x801001FF	想定外のデータを受信した場合に返されます．

中継異常応答，エラー応答が発生した際は@LAST_ERROR 変数を使用して詳細エラーを調べることにより原因を調査することができます．

解決する手順としては以下のようになります．

1. 中継異常応答，エラー応答が発生．
2. @LAST_ERROR 変数を使用して詳細情報を取得
3. メインレスポンスコード，サブレスポンスコードからエラー詳細を確認[4]
4. 本体停止異常，本体継続異常，ネットワーク中継異常時の処置方法を確認[4]

[4] エラー詳細は通信コマンドリファレンスマニュアル(“sbca-304r-30_cs1_cj1_cp1_com_cmd.pdf”)の「FINS コマンド一覧　終了コード一覧」を参照してください．

付録A. 変数名設定例

以下にいくつかの変数名の設定例を紹介します.

例1） チャネル I/O(CIO)のチャネルアドレス 0x0010, ビット 12 にアクセス

16.12 又は **0016.12** （桁の 0 埋めの有無に関わらず同じアドレスが指定されたものとして扱います）

例2） 拡張データメモリ EM バンク 4 のチャネルアドレス 0x00FF にアクセス

E4_255 又は **E4_0255**

例3） 内部補助リレー(WR)にビットアクセス出来るようアドレスをワイルドカード指定

W*.*

アドレス 0x0010, ビット 12 へアクセスするよう ID 設定

ID=4108 (0x001012)

例4） 保持リレー(HR)にチャネルアクセス出来るようアドレスをワイルドカード指定

H*

アドレス 0x0010 へアクセスするよう ID 設定

ID=4096 (0x001000)

例5） カウンタ(CNT)のアドレス 0x00A0 のアップフラグと現在値を取得するために変数を 2 つ登録

C0160$flag と **C0160$val** （"$"以降は解析されず別変数として登録できます）

付録B. 変数書き込み時の PLC(CJ シリーズ)内のメモリ割り付け

put_Value プロパティを用いてメモリ書き込みを行った際に，PLC（CJ シリーズ）内のメモリにどのようにマッピングされるか例を示します．(PLC（CJ シリーズ）内のメモリが全て 0 クリアされている前提で説明します)

例2） 保持リレー(HR) チャネルアドレス 0x0010 の Bit5，Bit7 を ON (1) にする

・設定用変数を 2 つ登録する

AddVariable

<変数 1>

Name ： H0016.05

Option ： Mode=NORMAL, VT=BOOL, Elem=1, Array=FALSE

<変数 2>

Name ： H0016.07

Option ： Mode=NORMAL, VT=BOOL, Elem=1, Array=FALSE

・各変数で値を書き込む

put_Value

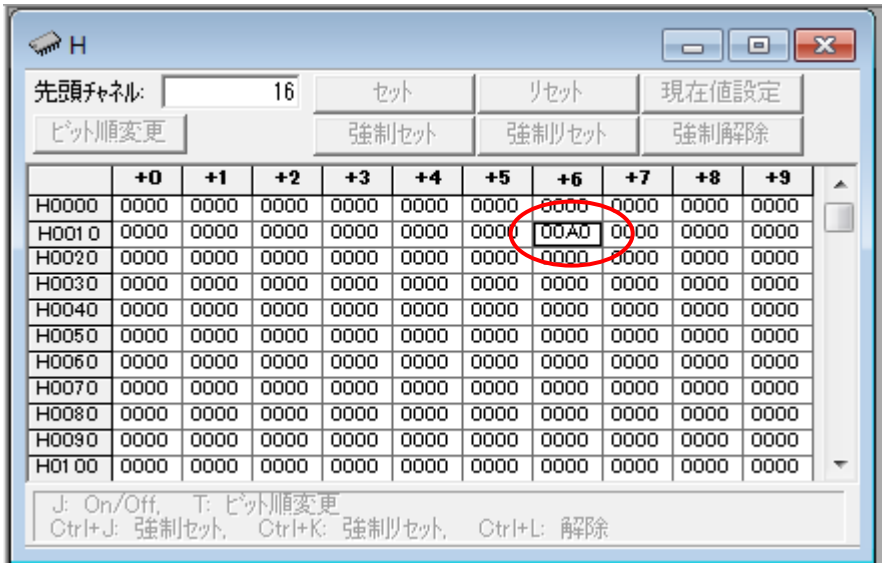
<変数 1>

VT_BOOL ： TRUE

<変数 2>

VT_BOOL ： TRUE

・PLC（CJ シリーズ）のメモリ内の状態（CX-Programmer を使用）



H0016 の値が 00A0（0000000010100000）

・確認のために同チャネルアドレスを読み出す変数を登録する

AddVariable

<変数 3>

Name : H0016

Option : Mode=NORMAL, VT=UI2, Elem=1, Array=FALSE

・値を読み出す

get_Value

<変数 3>

VT_UI2 : 160 (0x00A0)

例3) データメモリ(DM) チャネルアドレス 0x0100 に 1 バイト分データを書き込む

・設定用変数を 1 つ登録する

AddVariable

<変数 4>

Name : D0100

Option : Mode=NORMAL, VT=UI1, Elem=1, Array=FALSE

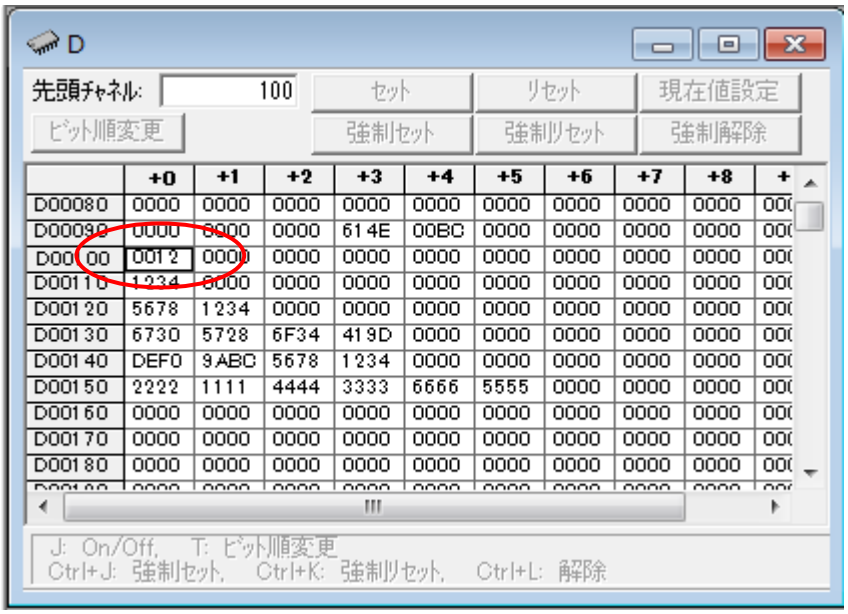
・値を書き込む

put_Value

<変数 4>

VT_UI1 :18 (0x12)

・PLC(CJ シリーズ)のメモリ内の状態(CX-Programmer を使用)



D0100 の値が 0012(0000000000010010)

・確認のために同チャネルアドレスを読み出す

get_Value

<変数 4>

VT_UI1 :18 (0x12)

例4) データメモリ(DM) チャネルアドレス 0x0110 に 2 バイト分データを書き込む

・設定用変数を 1 つ登録する

AddVariable

<変数 5>

Name : D0110

Option : Mode=NORMAL, VT=UI2, Elem=1, Array=FALSE

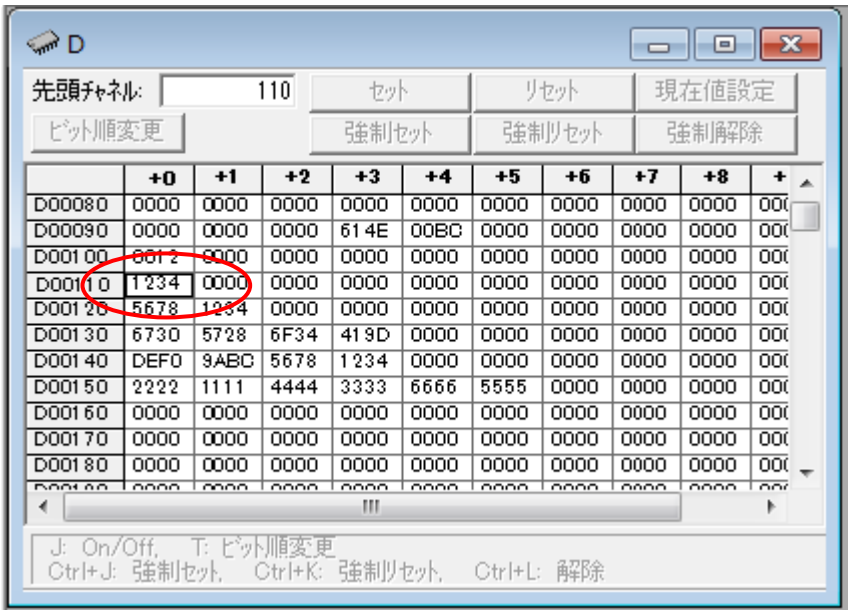
・値を書き込む

put_Value

<変数 5>

VT_UI2 :4660 (0x1234)

・PLC(CJ シリーズ)のメモリ内の状態(CX-Programmer を使用)



	+0	+1	+2	+3	+4	+5	+6	+7	+8	+
D00080	0000	0000	0000	0000	0000	0000	0000	0000	0000	000
D00090	0000	0000	0000	614E	00BC	0000	0000	0000	0000	000
D00100	6812	0000	0000	0000	0000	0000	0000	0000	0000	000
D00110	1234	0000	0000	0000	0000	0000	0000	0000	0000	000
D00120	5678	1234	0000	0000	0000	0000	0000	0000	0000	000
D00130	6730	5728	6F34	419D	0000	0000	0000	0000	0000	000
D00140	DEF0	9ABC	5678	1234	0000	0000	0000	0000	0000	000
D00150	2222	1111	4444	3333	6666	5555	0000	0000	0000	000
D00160	0000	0000	0000	0000	0000	0000	0000	0000	0000	000
D00170	0000	0000	0000	0000	0000	0000	0000	0000	0000	000
D00180	0000	0000	0000	0000	0000	0000	0000	0000	0000	000

D0110 の値が 1234(0001001000110100)

・確認のために同チャネルアドレスを読み出す

get_Value

<変数 5>

VT_UI2 :4660 (0x1234)

例5） データメモリ(DM) チャネルアドレス 0x0120 に 4 バイト分データを書き込む

・設定用変数を 1 つ登録する

AddVariable

<変数 6>

Name ： D0120

Option ： Mode=NORMAL, VT=I4, Elem=1, Array=FALSE

・値を書き込む

put_Value

<変数 6>

VT_I4 :305419896 (0x12345678)

・PLC(CJ シリーズ)のメモリ内の状態(CX-Programmer を使用)

D0120 の値が 5678(0101011001111000)

D0121 の値が 1234(0001001000110100)

・確認のために同チャネルアドレスを読み出す

get_Value

<変数 6>

VT_I4 :305419896 (0x12345678)

例6) データメモリ(DM) チャネルアドレス 0x0140 に 8 バイト分データを書き込む

・設定用変数を 1 つ登録する

AddVariable

<変数 7>

Name : D0140

Option : Mode=NORMAL, VT=I8, Elem=1, Array=FALSE

・値を書き込む

put_Value

<変数 7>

VT_I8 :1311768467463790320 (0x123456789ABCDEF0)

・PLC(CJ シリーズ)のメモリ内の状態(CX-Programmer を使用)

D0140 の値が DEF0(1101111011110000)

D0141 の値が 9ABC(1001101010111100)

D0142 の値が 5678(0101011001111000)

D0143 の値が 1234(0001001000110100)

・確認のために同チャネルアドレスを読み出す

get_Value

<変数 7>

VT_I8 :1311768467463790320 (0x123456789ABCDEF0)

例7) データメモリ(DM) チャネルアドレス 0x0150 に 4 バイト分データを 3 つ書き込む

・設定用変数を 1 つ登録する

AddVariable

<変数 8>

Name : D0150

Option : Mode=NORMAL, VT=I4, Elem=3, Array=TRUE

・値を書き込む

put_Value

<変数 8>

VT_ARRAY | VT_I4 :

286335522 (0x11112222),

858997828 (0x33334444),

1431660134 (0x55556666)

・PLC(CJ シリーズ)のメモリ内の状態(CX-Programmer を使用)

D0150 の値が 2222(1101111011110000)
D0151 の値が 1111(1001101010111100) } 要素 0

D0152 の値が 4444(0101011001111000)
D0153 の値が 3333(0001001000110100) } 要素 1

D0154 の値が 6666(0001001000110100)
D0155 の値が 5555(0001001000110100) } 要素 2

・確認のために同チャネルアドレスを読み出す

get_Value

<変数 8>

VT_ARRAY | VT_I4 :

286335522 (0x11112222),

858997828 (0x33334444),

1431660134 (0x55556666)

付録C. チャネルアドレスの VT=BIT 指定時の互換処理について

チャネルアドレス指定時に VT=BIT を指定した場合の挙動が v1.2.3 から 1 バイト分の値を取得し, 0 なら 0 を 0 以外なら 1 を取得する仕様に修正されました. v1.2.3 未満の処理に戻すには, AddController オプションで"BitOpt"オプションを指定する必要があります.

詳細は, 表 2-9 を参照してください.

表 2-9 CaoWorkspace::AddController のオプション文字列

オプション	説明
BitOpt[=< True or False >]	チャネルアドレスからの取得で VT=BIT を指定した時の挙動を指定. ・FALSE (最新動作) 1 バイト分のデータを取得し, 0 の場合に 0 を返し, 0 以外の場合に 1 を返します. ・TRUE (v1.2.3 未満のバージョンの動作) : 1 バイト分のデータを取得し, その値をそのまま返します. (デフォルト:FALSE)

付録D. チャネルアドレスの VT=BIT, VT=BOOL, VT=UI1, VT=I1 指定時の書き込み処理について

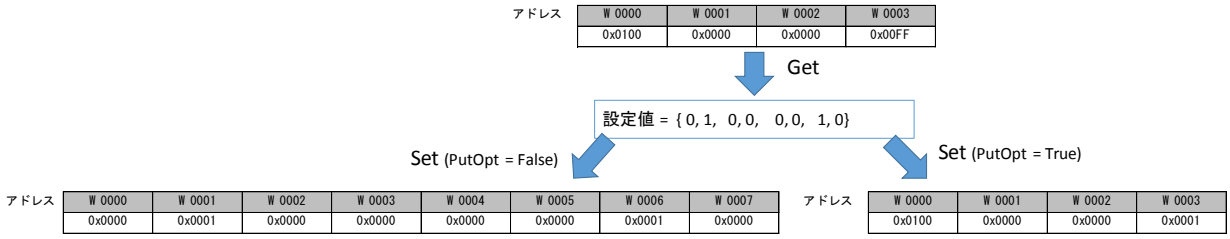
チャネルアドレス指定時に VT=BIT, VT=BOOL, VT=UI1, VT=I1 を指定した場合の挙動を v1.2.7 から選択できるようになりました. v1.2.7 以降に作成する場合は本オプションに True を指定してください.

表 2-10 CaoWorkspace::AddController のオプション文字列

オプション	説明
PutOpt[=< True or False >]	・FALSE (v1.2.7 未満のバージョン動作) データを書き込む際に上位バイトを 0 で補完して書き込みます. ・TRUE (v1.2.7 以降で追加された動作) 下位バイトから順にデータを格納して書き込みます. (デフォルト:FALSE)

以下にそれぞれの動作イメージを記述します.

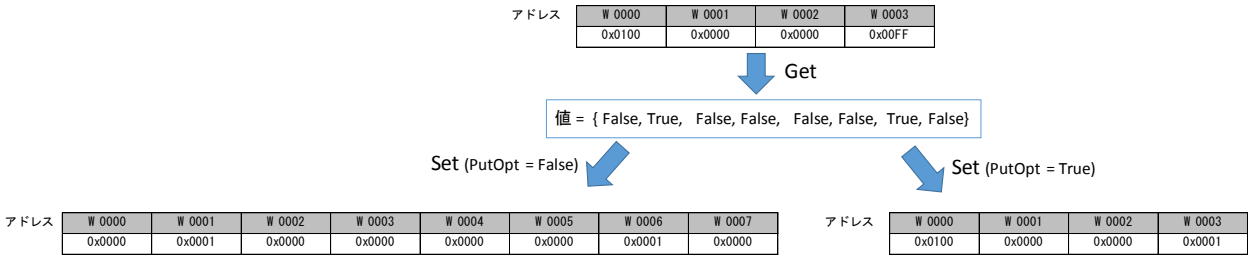
<変数>
Name : W0
Option : VT=BIT, Elem=8



FALSE 時: 読み込み時は, 1 チャネルデータを上位バイトと下位バイトデータに分け, それぞれ 0 以外であれば 1 とします. 書き込み時は, 1 ビットデータを 1 チャネルデータとして扱い, 上位バイトは 0 で補完してデバイスに書き込みます.

TRUE 時: 読み込み時は, 1 チャネルデータを上位バイトと下位バイトデータに分け, それぞれ 0 以外であれば 1 とします. 書き込み時は, 2 ビットデータを 1 チャネルデータとして扱い, 下位バイトから順にデータを格納してデバイスに書き込みます.

<変数>
Name : W0
Option : VT=BOOL, Elem=8



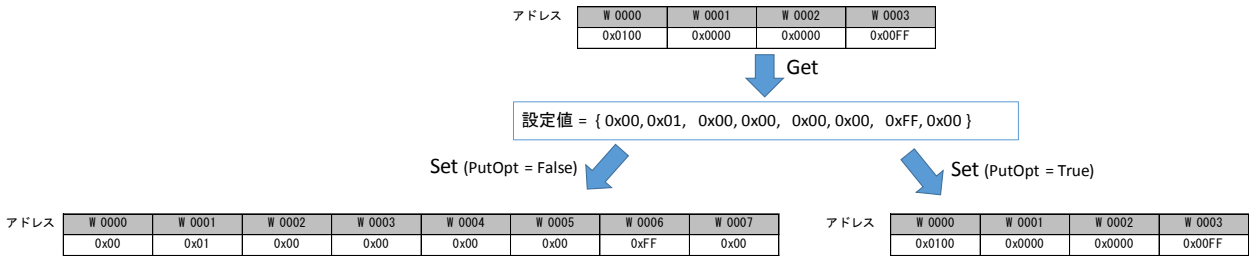
FALSE 時: 読み込み時は, 1 チャネルデータを上位バイトと下位バイトデータに分け, それぞれ 0 以外であれば True とします. 書き込み時は, 1 ブールデータを 1 チャネルデータとして扱い, 上位バイトは 0 で補完してデバイスに書き込みます.

TRUE 時: 読み込み時は, 1 チャネルデータを上位バイトと下位バイトデータに分け, それぞれ 0 以外であれば True とします. 書き込み時は, 2 ブールデータを 1 チャネルデータとして扱い, 下位バイトから順にデータを格納してデバイスに書き込みます.

<変数>

Name : W0

Option : VT=UI1 or I1, Elem=8



FALSE 時: 読み込み時は, 1 チャネルデータを上位バイトと下位バイトデータに分けて値を取得します. 書き込み時は, 1 バイトデータを 1 チャネルデータとして扱い, 上位バイトは 0 で補完してデバイスに書き込みます.

TRUE 時: 読み込み時は, 1 チャネルデータを上位バイトと下位バイトデータに分けて値を取得します. 書き込み時は, 2 バイトのデータを 1 チャネルデータとして扱い, 下位バイトから順にデータを格納してデバイスに書き込みます.

付録E. ネットワーク上の PLC へのアクセスについて

付録E.1. PLC の設定

ネットワーク上の PLC へアクセスするためにはルーチングテーブルの作成が必要になります.

また, プロトコル上の仕様で FinsUDP 通信でのみアクセス可能です. その際, 接続元ポート番号を接続先ポート番号と合わせる必要があります.

ルーチングテーブルの設定には CX-Integrator を使用します[5].

以下の構成例で説明します.

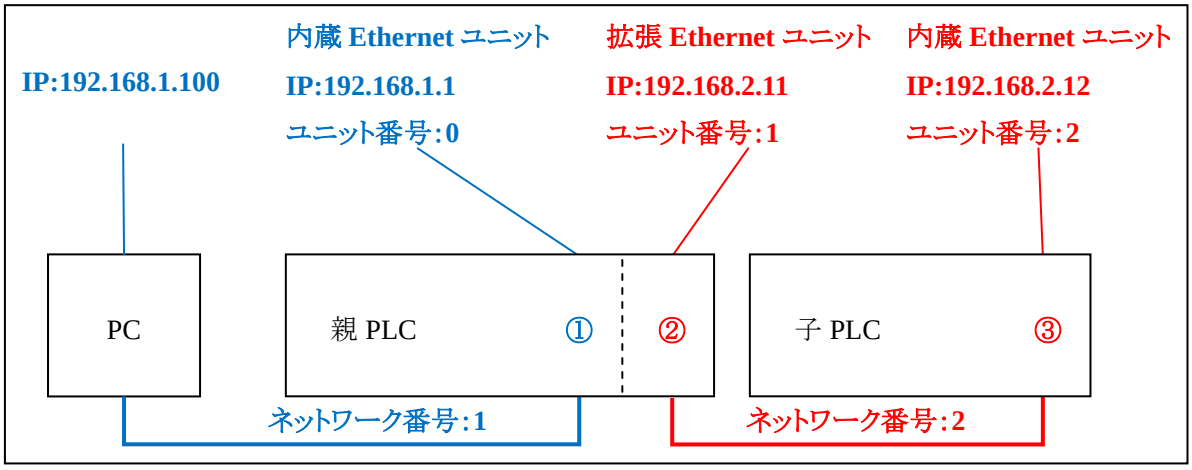


表 2-11 対象ユニットの説明

対象ユニット	説明
①	親 PLC の内蔵 Ethernet ユニット
②	親 PLC の拡張 Ethernet ユニット
③	子 PLC の内蔵 Ethernet ユニット

[5] CX Integrator による設定方法の手順については CX-Integrator Help を参照してください.

親 PLC のルーチングテーブルを以下のように設定します.

子 PLC のルーチングテーブルを以下のように設定します.

付録E.2. 接続確認

ルーチングテーブルを設定した PLC に対して接続確認の手順を示します.

表 2-12 接続確認

手順	内容
1	親 PLC への接続を確認します. 以下のように AddController オプションを設定します. Conn=UDP:192.168.1.1:9600:255.255.255.255:9600, FinsParam=1:1:0:2:1:100:0 接続後, 親 PLC から値を取得できるか確認します. エラーが発生する場合は親 PLC のルーチングテーブルを見直してください.
2	親 PLC を介して子 PLC から値を取得できるか確認します. 以下のように AddController オプションを設定します. Conn=UDP:192.168.1.1:9600:255.255.255.255:9600, FinsParam=2:12:0:2:1:100:0 接続後, 子 PLC から値を取得できるか確認します. エラーが発生する場合は子 PLC のルーチングテーブルを見直してください.

付録F. パフォーマンス測定結果

以下の条件でパフォーマンス測定を行った結果を記載します.

表 2-13 パフォーマンス測定条件

使用機種	OMRON SYSMAC CJ2H CPU64-EIP
ネットワーク	100BASE-TX
PC の CPU	Intel(R) Core(TM) i7-7700 3.60GHz
PC のオペレーティングシステム	Windows10 20H2 64bit
測定条件	2 バイト整数のタグにアクセスする変数に対して 100 回値を取得して取得にかかった時間の平均値を実行時間として採用

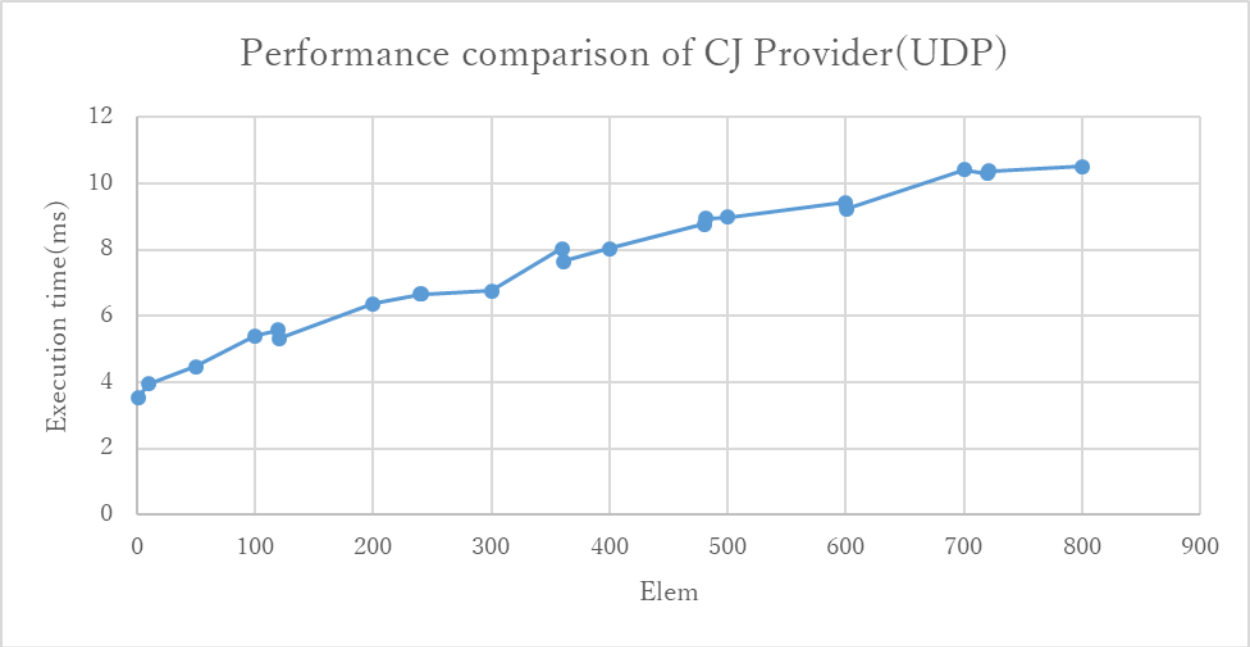


図 2-1 パフォーマンス測定結果