Mitsubishi 汎用シーケンサ
MELSEC-A シリーズ用プロバイダ

Version 1.1.3

ユーザーズ ガイド

March 27, 2020

【備考】

本書で使用している画像の一部は「A 対応 Ethernet インタフェースユニット ユーザーズマニュアル(詳細編)」,「計算機リンク/マルチドロップリンクユニット ユーザーズマニュアル」から引用しています.

【改版履歴】

バージョン	日付	内容
1.0.0	2017-10-10	初版.
1.1.0	2018-08-03	UDP オプション, Retry オプション追加
1.1.1	2018-10-30	メモリリーク バグ修正
1.1.2	2019-12-25	配列分割サイズ指定対応 通信プロトコルコマンド対応表追加
1.1.3	2020-03-27	AddController 失敗時の処理修正

【動作確認機器】

機種	バージョン	注意事項
A2ACPU		Ethernet, シリアル接続に対応.

目次

1. はじめに 4

2. プロバイダの概要 5

2.1. 概要 5
2.2. 使用上の注意点 5
2.2.1. CPU ユニットとの接続について 5
2.2.2. データコードの設定について 7
2.2.3. TCP 通信使用時の再接続処理にについて 7
2.2.4. 外部機器との通信について 8
2.3. メソッド・プロパティ 11
2.3.1. CaoWorkspace::AddController メソッド 11
2.3.1.1. Conn オプション 12
2.3.1.2. ProcessingPoints オプション 13
2.3.1.3. Format オプション 14
2.3.1.4. サンプルプログラム 17
2.3.2. CaoController::AddVariable メソッド 19
2.3.2.1. Path オプション 20
2.3.2.2. Param オプション 21
2.3.2.3. VT オプション 22
2.3.2.4. Elem オプション 23
2.3.2.5. Array オプション 24
2.3.2.6. サンプルプログラム 25
2.4. エラーコード 27

3. 通信プロトコルコマンド対応表 28

1. はじめに

本書は，三菱電機製汎用シーケンサ(MELSEC-A 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(CaoProvMELSECAnA.dll)を MELSEC-A プロバイダと呼びます．

MELSEC-A プロバイダは三菱電機製シーケンサ MELSEC-A シリーズの A 対応 Ethernet インタフェースユニット，計算機リンクユニット，を対象としたMELSECコミュニケーションプロトコル(以下 MC プロトコル)通信を行います．

第 2 章に MELSEC-A プロバイダの概要，変数の詳細を記載しています．

MELSEC-A 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，通信先となる汎用シーケンサ(MELSEC-A シリーズ)に依存します．[1]

通信の詳細や設定方法については「A 対応 Ethernet インタフェースユニット ユーザーズマニュアル(詳細編)」，「計算機リンク/マルチドロップリンクユニット ユーザーズマニュアル」を参照してください．

[1] MELSEC-FX1/2/3 シリーズについては同 MC プロトコルを使用しているため理論上は動作するものですが，動作確認はできておりません．

2. プロバイダの概要

2.1. 概要

MELSEC-A プロバイダは，三菱電機製シーケンサに対して MC プロトコルの QnA 互換 1E フレーム，QnA 互換 1C フレーム形式 1，2，3，4 を用いてデータの書き込み/読み出しを行う 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．MELSEC-A 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 MELSEC-A プロバイダ

ファイル名	CaoProvMELSECAnA.dll
ProgID	CaoProv.MELSEC.AnA
レジストリ登録	regsvr32 CaoProvMELSECAnA.dll
レジストリ登録の抹消	regsvr32 /u CaoProvMELSECAnA.dll

2.2. 使用上の注意点

2.2.1. CPU ユニットとの接続について

PC と CPU ユニットの接続につきましては，「A 対応 Ethernet インタフェースユニット ユーザーズマニュアル(詳細編)」，「計算機リンク/マルチドロップリンクユニット ユーザーズマニュアル」を参照して下さい．

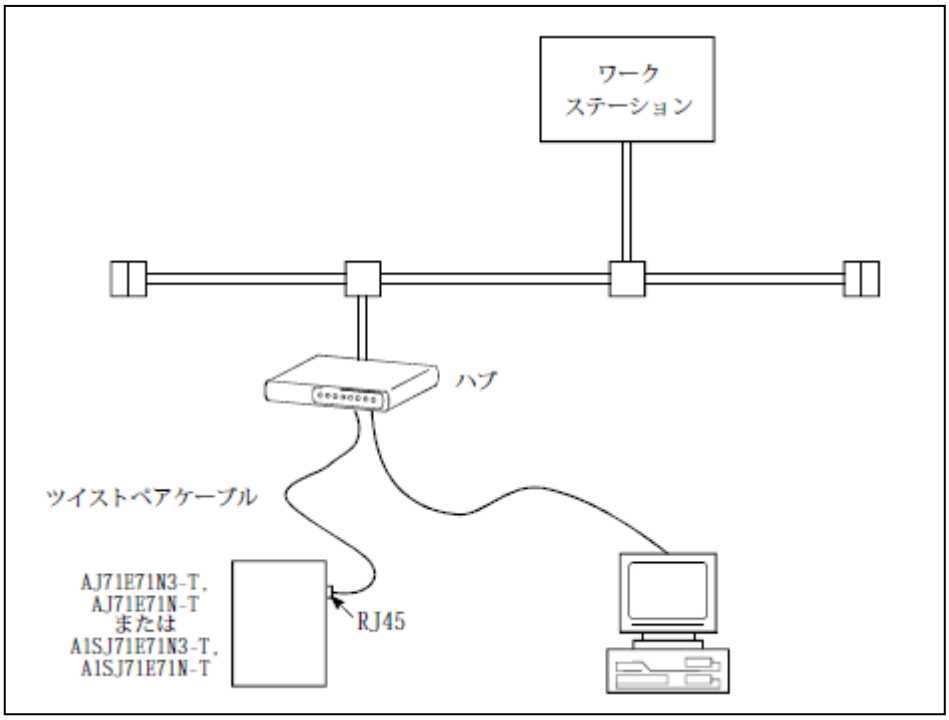


図 1　Ethernet 接続例

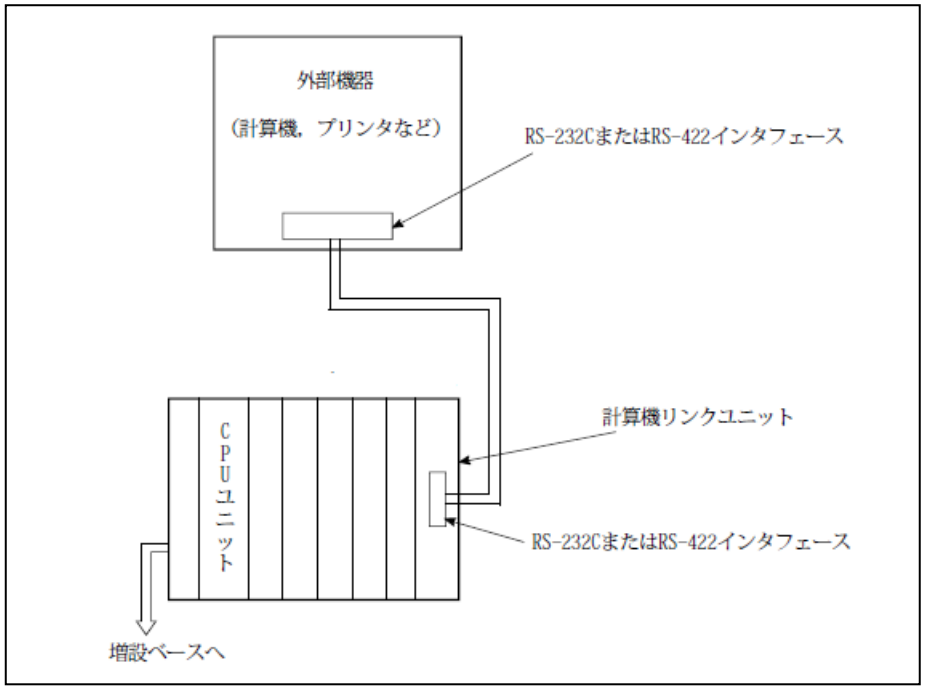


図 2　シリアル接続例

2.2.2. データコードの設定について

Ethernet 接続では MC プロトコルの QnA 互換 1E プロトコルのバイナリコードを用いて交信を行います.
データコードをバイナリに設定する必要があります.

4.3.2 交信条件の設定

交信条件設定スイッチ *1	スイッチ名称	設定項目	設定内容	工場出荷時
OFF ON SW1 SW2 SW3 SW4 SW5 SW6 SW7 SW8	SW1	TCPタイムアウトエラー時の回線処理選択	TCP ULPタイムアウトエラー発生時の回線処理を選択する。 OFF：TCP ULPタイムアウトエラーの発生により回線をクローズする。 ON ：TCP ULPタイムアウトエラーが発生しても回線をクローズしない。	OFF
	SW2	データコード設定	他ノードとの交信データのデータコード種別を選択する。(3.3項参照) OFF：バイナリコードにより交信を行う。 ON ：ASCIIコードにより交信を行う。	OFF
	SW3	使用不可（OFF固定）		OFF
	SW4			OFF
	SW5			OFF
	SW6			OFF
	SW7	CPU交信タイミング設定	シーケンサCPUがRUN中に，他ノードからのデータ書込みの許可／禁止を選択する。(シーケンサCPU内データの読出し／書込み交信用) OFF：シーケンサCPU RUN中には，他ノードからの書込みを禁止する。 ON ：シーケンサCPU RUN中でも，他ノードからの書込みを行う。	OFF
	SW8	イニシャルタイミング設定	イニシャル処理を起動するタイミングを選択する。 OFF：クイックスタート（遅延時間なしで起動） 一つのネットワークで全体構成されている場合に設定する。 ON ：ノーマルスタート（20秒の遅延時間後に起動） 複数のネットワークで全体構成されている場合に設定する。	OFF

*1 ハードウェアバージョンが「B版」以降のA1SJ71E71N-B5の交信条件設定スイッチは，下記のようになります。

図 3 Ethernet ユニット設定方法

シリアル接続では MC プロトコルの QnA 互換 1C プロトコルのフレーム形式 1，2，3，4 を ASCII コードで交信を行います.

データコードは ASCII のみのためデータコードの設定は不要です.

2.2.3. TCP 通信使用時の再接続処理にについて

TCP 通信使用時は以下の点に注意してください.

セッションの再接続に十分な時間が確保できない場合は UDP 通信を使用してください.

・ セッション切断後，再度オープン待ち状態にする場合は，十分に時間をあけてください.

2.2.4. 外部機器との通信について

外部機器との通信を行う場合には，シーケンサのラダーを作成しないと通信できません．

Ethernet 接続では以下のようにラダーを作成しポートを開ける必要があります．

以下の例では IP アドレスを「192.168.2.1」に設定しコネクション 1 に UDP/IP を「1025」ポートで開放，コネクション 2 に TCP/IP Unpassive を「1026」ポートで開放します．

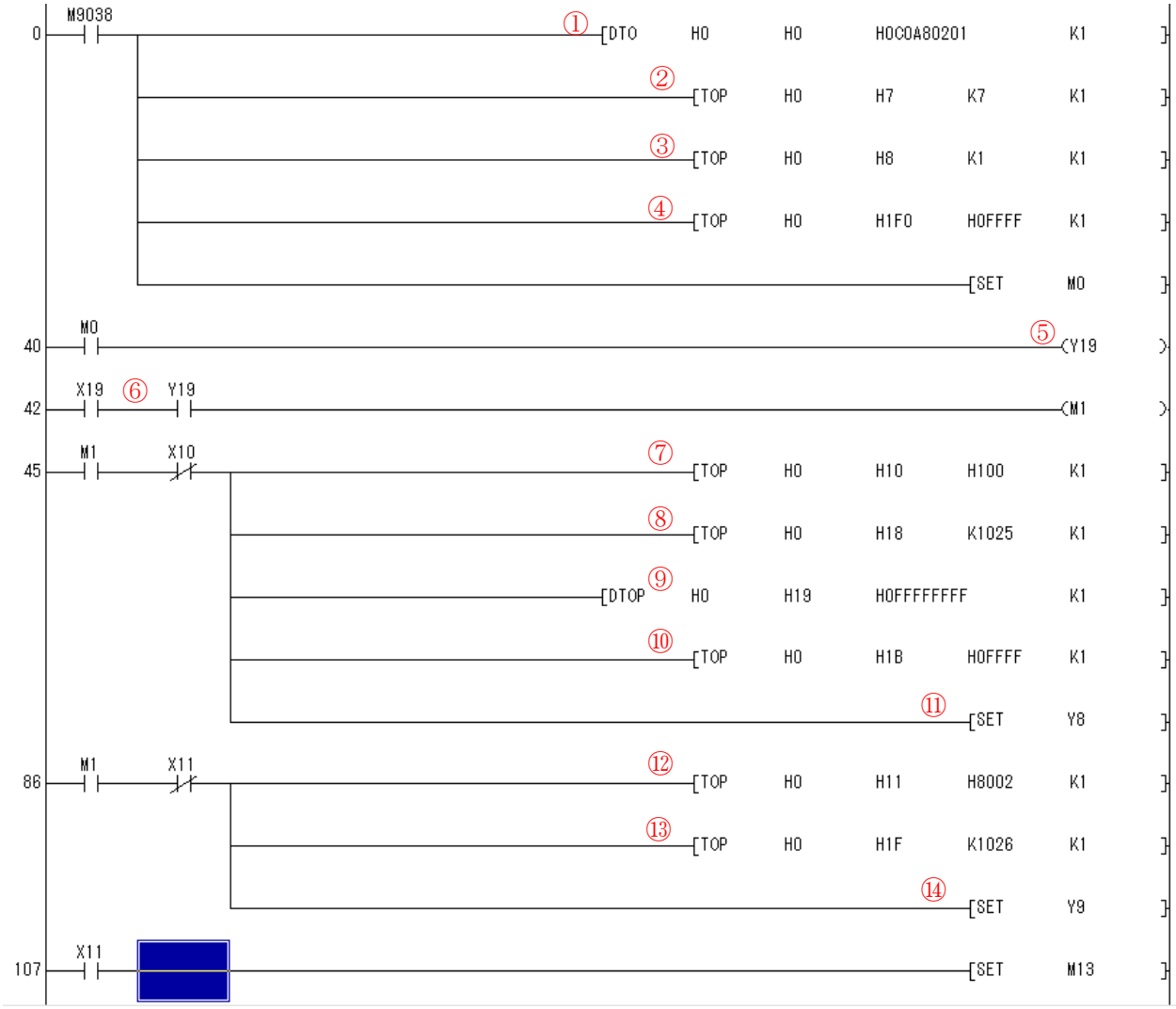


図 4　Ethernet 接続設定方法

① IP アドレスを「192.168.2.1」に設定

② 生存確認開始間隔タイマを 14 秒(7×2 秒)に設定

③ 生存確認間隔タイマを 2 秒(1×2 秒)に設定

④ STOP 中交信指示を全コネクション「可」(0xFFFF)に設定

⑤ イニシャル要求信号(Y19)を ON

⑥ イニシャル完了信号(X10)が ON した後，オープン処理に移行

⑦ コネクション 1 のプロトコルに UDP/IP(0x100)を設定
⑧ コネクション 1 の自ポート番号を「1025」に設定
⑨ ブロードキャスト通信を行うために，他 IP アドレスを「0xFFFFFFFF」に設定
⑩ ブロードキャスト通信を行うために，他ポート番号を「0xFFFF」に設定
⑪ コネクション 1 のオープン要求信号(Y8)を ON
⑫ コネクション 2 のプロトコルに TCP/IP Unpassive(0x8002)を設定
⑬ コネクション 2 の自ポートに「1026」を設定
⑭ コネクション 2 オープン要求信号(Y9)を ON

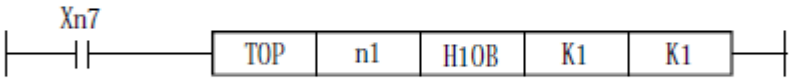
シリアル接続では CD 端子チェックの有無により以下のラダーを作成する必要があります.

(3) RS-232C CD端子チェック設定について

RS-232C CD端子チェック設定（バッファメモリのアドレス10BHで設定，3.10項参照）による計算機リンクユニットの，CD信号に対する動作は次のとおりです。

	CD端子チェックあり	CD端子チェックなし
全二重通信	計算機リンクユニットはCD信号（受信キャリア検出）のON状態で送受信処理を行う。データ交信時は，CD信号がOFFすると，計算機リンクユニットの伝送シーケンスを初期化する。	計算機リンクユニットは，CD信号のON/OFF状態に関係なく全二重通信方式で送受信処理を行う。 （計算機リンクユニットは，CD信号のON/OFFチェックを行わず，CD信号がONと同じ処理を行う。） CD信号をON/OFFさせられない外部機器とのデータ交信が可能。

RS-232C CD端子チェック設定で，“CD端子チェックなし”を設定するときは，次に示すシーケンスプログラムを組み込んでください。
“CD端子チェックあり”を設定するときは，計算機リンクユニット立上がり時のデフォルト値が「0」（CD端子チェックあり）のため，次に示すシーケンスプログラムの組込みは不要です。



＊RS-232C CD端子チェック設定については，第5章～第7章の各計算機リンク機能説明項の中(5.2.1項(2)②，6.2.4項(2)⑥，7.2.6項(2)④)でも説明しています。

図 5　シリアル接続設定方法

2.3. メソッド・プロパティ

2.3.1. CaoWorkspace::AddController メソッド

MELSEC-A プロバイダでは Controller オブジェクトの生成時に，イーサネット（TCP，UDP）通信またはシリアル通信の初期処理を行います．接続時にオプション文字列でデバイスを指定します．

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.MELSEC.AnA”
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[2]	説明
Conn=<接続パラメータ>	必須． 通信形態とその接続パラメータを設定します． (参照 2.3.1.1)
ProcessingPoints[=<一交信最大処理点数>]	一交信で行う最大処理点数を設定します． (参照 2.3.1.2)
Format[=<形式>]	シリアル通信に使用する形式(1～4)を設定します． (デフォルト:1) シリアル通信時のみ有効なオプションです． (参照 2.3.1.3)
SumCheck[=<有効/無効>]	シリアル通信のサムチェックの有効(1)/無効(0)を設定します． (デフォルト:1) シリアル通信時のみ有効なオプションです．
Timeout[=<タイムアウト時間>]	送受信時のタイムアウト時間をミリ秒で設定します． (デフォルト:3000)
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)

[2] 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．

2.3.1.1. 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(“[ ]”)内のパラメータ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指定を省略した時のデフォルト値を示します.

【Ethernet デバイス】

“Conn=ETH:<Dest IP Address>:<Dest Port No>”

“Conn=TCP:<Dest IP Address>:<Dest Port No>”

“Conn=UDP:<Dest IP Address>:<Dest Port No>”

< Dest IP Address > : 接続先の IP アドレス.

< Dest Port No > : 接続先のポート番号.

【シリアルデバイス】

“Conn=com:<COM Port>[:<BaudRate>[:<Parity>:<DataBits>:<StopBits>]]”

<COM Port> : COM ポート番号. ‘1’-COM1, ‘2’-COM2, …

<BaudRate> : 通信速度.

300, 600, 1200, 2400, 4800, 9600, 19200

<Parity> : パリティ. ‘N’-NONE, ‘E’-EVEN, ‘O’-ODD

<DataBits> : データビット数. ‘7’-7bit, ‘8’-8bit.

<StopBits> : ストップビット数. ‘1’-1bit, ‘2’-2bit.

(例 1) “com:1” 通信ポート COM1 (, 19200bps, None, 8bits, 1bit)

(例 2) “com:2:9600” 通信ポート COM2, 9600bps (, None, 8bits, 1bit)

(例 3) “com:3:4800:N:8:2” 通信ポート COM3, 4800bps, None, 8bits, 2bit

2.3.1.2. ProcessingPoints オプション

一交信で行う最大処理点数を指定します. この処理点数を超過すると分割して交信を行います[3].

例えば, Ethernet デバイス接続時にデフォルト設定でワードデバイスを 700 点交信する場合, 256 + 256 + 188 の 3 回更新を行います.

各点数は偶数点数単位でのみ設定可能です. また 256 点までしか設定できません.

【Ethernet デバイス】

"[ProcessingPoints=[<BitDevInBitUnit>:<BitDevInWordUnit>:<WordDevInWordUnit>]]"

< BitDevInBitUnit > : ビットデバイスをビット(1 点)単位で交信します.
デフォルトは 256 点です.

< BitDevInWordUnit > : ビットデバイスをワード(16 点)単位で交信します.
デフォルトは 128 ワード(2048 点)です.

< WordDevInWordUnit > : ワードデバイスをワード(1 点)単位で交信します.
デフォルトは 256 ワードです.

【シリアルデバイス】

"[ProcessingPoints=[<BitDevInBitUnit>:<BitDevInWordUnit>:<WordDevInWordUnit>]]"

< BitDevInBitUnit > : ビットデバイスをビット(1 点)単位で交信します.
デフォルトは 160 点です.

< BitDevInWordUnit > : ビットデバイスをワード(16 点)単位で交信します.
デフォルトは 10 ワード(160 点)です.

< WordDevInWordUnit > : ワードデバイスをワード(1 点)単位で交信します.
デフォルトは 64 ワードです.

(例) "Path=X0, Elem=5" X0 から X4 の値をビット値として取得します
(例) "Path=D10, Elem=0x10" D10 から D25 の値をワード値として取得します
(例) "Path=D10, Elem=&H10" D10 から D25 の値をワード値として取得します
(例) "Path=D10, Elem=10H" D10 から D25 の値をワード値として取得します
(例) "Path=M100, VT=I2, Elem=2" M100 から M131 の値をワード単位で取得します

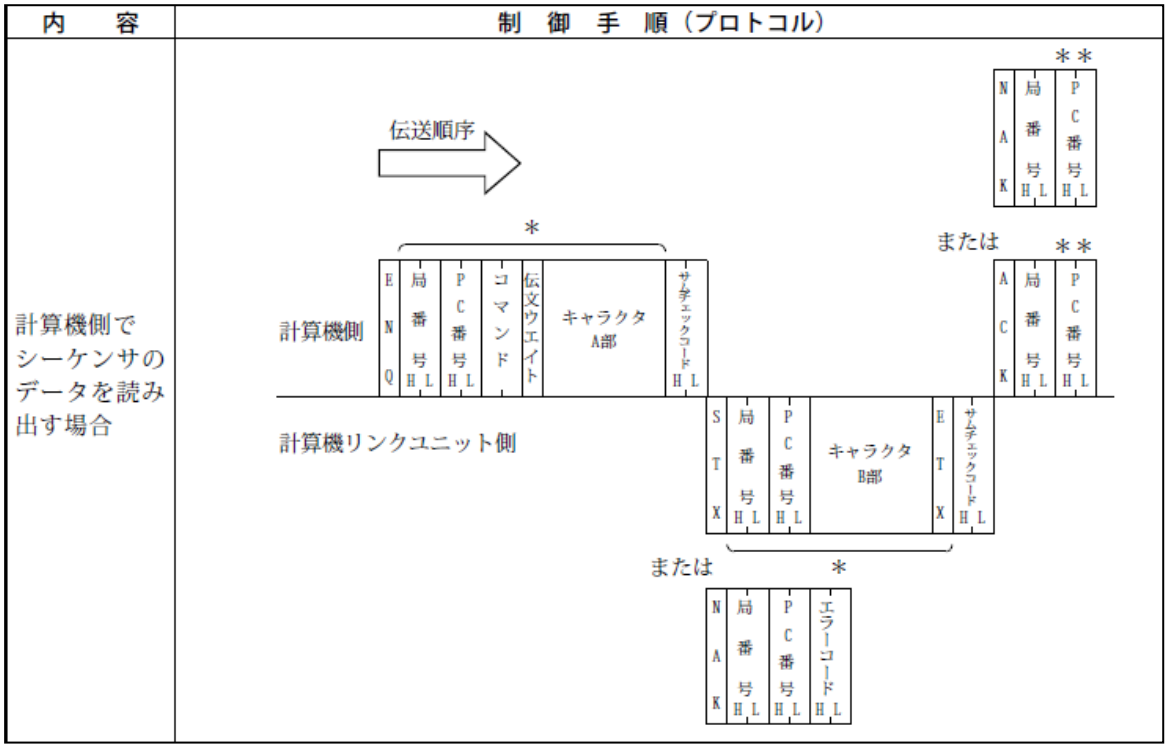
[3] 一度に交信できる最大処理点数はユニット毎に異なります. 詳細はユニットのユーザーズマニュアルを参照ください.

2.3.1.3. Format オプション

ASCII コードによる交信用の設定です. 各形式の違いは形式 1 を基準に考えると下記になります. 詳細はマニュアルをご覧ください.

Form (形式)	説明
2	各伝文にブロック番号を付加した形式
3	各伝文を STX, ETX で囲んだ形式
4	各伝文に CR, LF を付加した形式

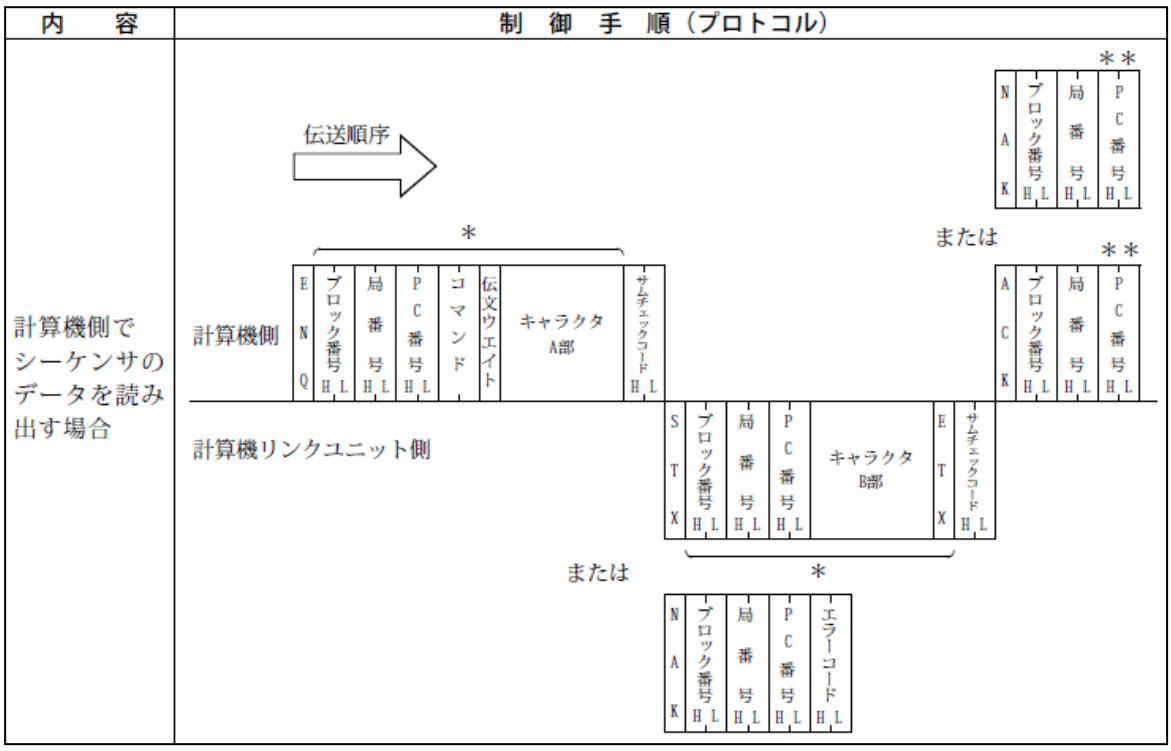
【形式 1】



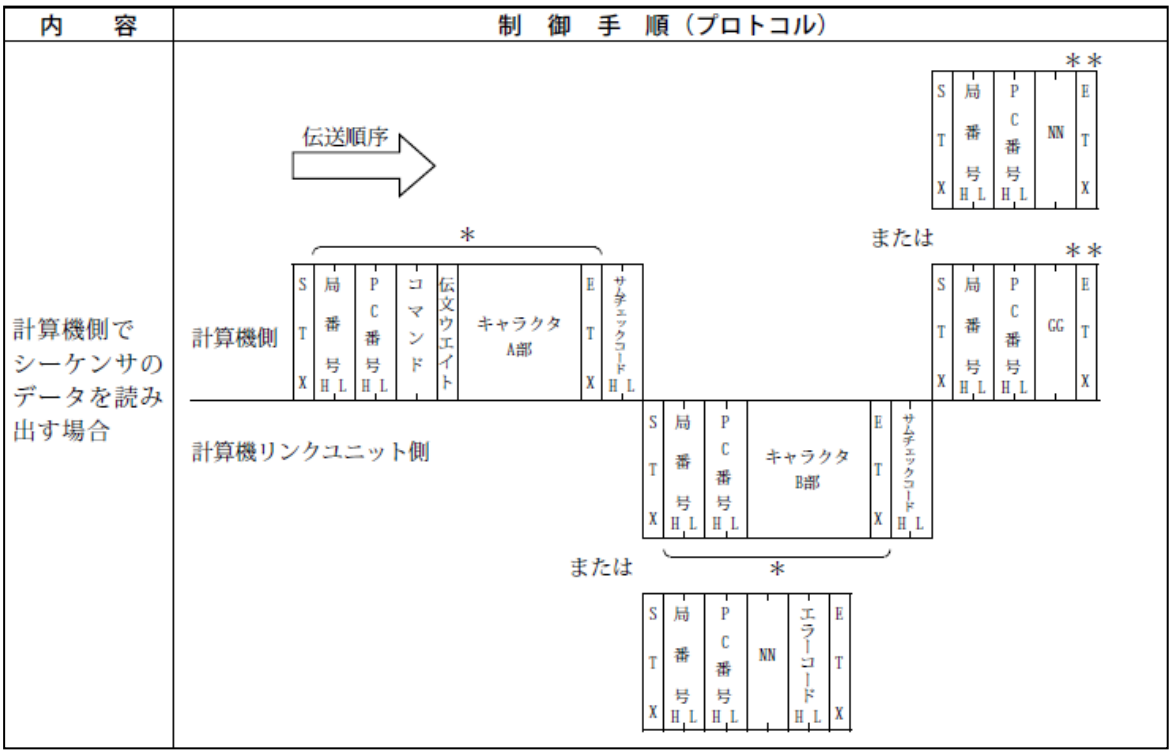
サムチェック「あり」に設定した場合は, 上記図の＊印部分のキャラクタに対してのみサムチェックを行います.

計算機側でシーケンサのデータを読み出す場合, 上記図中の＊＊印部分のACK/NAK伝文の送信は省略できます.

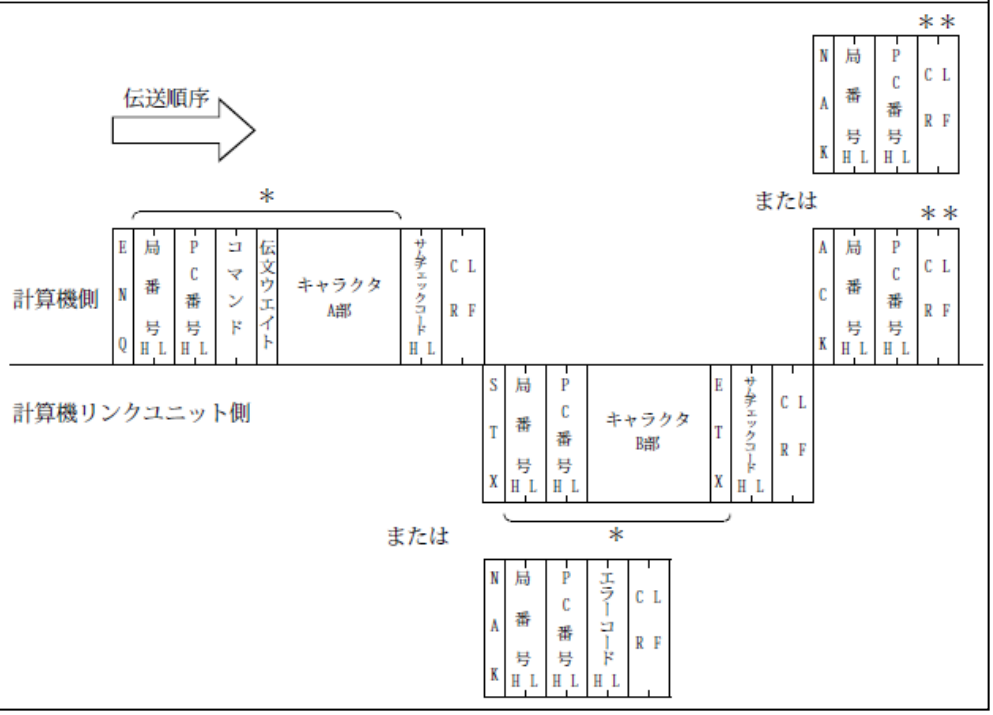
【形式 2】


内　容
制　御　手　順（プロトコル）
計算機側で
シーケンサの
データを読み
出す場合
伝送順序
＊
＊＊
または
計算機側
計算機リンクユニット側
ENQ
NAK
ACK
STX
ETX
ブロック番号
局番号
PC番号
コマンド
伝文ウエイト
キャラクタ
A部
キャラクタ
B部
サムチェックコード
エラーコード
H L

【形式 3】


内　容
制　御　手　順（プロトコル）
計算機側で
シーケンサの
データを読み
出す場合
伝送順序
＊
＊＊
または
計算機側
計算機リンクユニット側
STX
ETX
局番号
PC番号
NN
GG
コマンド
伝文ウエイト
キャラクタ
A部
キャラクタ
B部
サムチェックコード
エラーコード
H L

【形式 4】

内　容	制　御　手　順（プロトコル）
計算機側でシーケンサのデータを読み出す場合	

2.3.1.4. サンプルプログラム

AddControllerのサンプルプログラムを以下に示します.

【Ethernet】

```
    HRESULT hr = S_OK;
    ICaoEngine* pEng = NULL;
    ICaoWorkspaces *pWss = NULL;
    ICaoWorkspace *pWs = NULL;
    ICaoController *pCtrl = NULL;

    // CaoEngine の生成
    hr = CoCreateInstance(CLSID_CaoEngine,
                          NULL,
                          CLSCTX_LOCAL_SERVER,
                          IID_ICaoEngine,
                          (void **)&pEng);
    if (FAILED(hr)) {
        goto EndProc;
    }
    // CaoWorkspace コレクションの取得
    hr = pEng->get_Workspaces(&pWss);
    if (FAILED(hr)) {
        goto EndProc;
    }
    // CaoWorkspace の取得
    hr = pWss->Item(CComVariant(0L), &pWs);
    if (FAILED(hr)) {
        goto EndProc;
    }

    // CaoController の生成
    hr = pWs->AddController(CComBSTR(L"MELSEC_AnA"),
                          CComBSTR(L"CaoProv.MELSEC.AnA"),
                          CComBSTR(L""),
                          CComBSTR(L"Conn=ETH:192.168.2.1:1026"),
                          &pCtrl);
    if (FAILED(hr)) {
        goto EndProc;
    }

    // ここに必要な処理を入れる
    // 値の設定,取得など

EndProc:
    if (pCtrl) pCtrl->Release();
    if (pWs) pWs->Release();
    if (pWss) pWss->Release();
    if (pEng) pEng->Release();
```

【シリアル】

```
    HRESULT hr = S_OK;
    ICaoEngine* pEng = NULL;
    ICaoWorkspaces *pWss = NULL;
    ICaoWorkspace *pWs = NULL;
    ICaoController *pCtrl = NULL;

    // CaoEngine の生成
    hr = CoCreateInstance(CLSID_CaoEngine,
                          NULL,
                          CLSCTX_LOCAL_SERVER,
                          IID_ICaoEngine,
                          (void **)&pEng);
    if (FAILED(hr)) {
        goto EndProc;
    }
    // CaoWorkspace コレクションの取得
    hr = pEng->get_Workspaces(&pWss);
    if (FAILED(hr)) {
        goto EndProc;
    }
    // CaoWorkspace の取得
    hr = pWss->Item(CComVariant(0L), &pWs);
    if (FAILED(hr)) {
        goto EndProc;
    }

    // CaoController の生成
    hr = pWs->AddController(CComBSTR(L"MELSEC_AnA"),
                          CComBSTR(L"CaoProv.MELSEC.AnA"),
                          CComBSTR(L""),
                          CComBSTR(L"Conn=com:7"),
                          &pCtrl);
    if (FAILED(hr)) {
        goto EndProc;
    }

    // ここに必要な処理を入れる
    // 値の設定,取得など

EndProc:
    if (pCtrl) pCtrl->Release();
    if (pWs) pWs->Release();
    if (pWss) pWss->Release();
    if (pEng) pEng->Release();
```

2.3.2. CaoController::AddVariable メソッド

シーケンサに対しデータの書込み/読出しを行う変数オブジェクトを作成します.

書式 AddVariable(<bstrName:BSTR > ,<bstrOption:BSTR>)

bstrName : [in] 変数名. 管理する為の任意で一意な文字列.

bstrOption : [in] オプション文字列

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

オプション(2)	意味
Path=<先頭デバイス>	必須. アクセス対象のデバイスメモリの先頭デバイスを「デバイスコード＋デバイス番号」で指定します. (参照 2.3.2.1)
Param[=<変数パラメータ>]	変数のパラメータを設定します. (参照 2.3.2.2)
VT[=<変数型>]	デバイスメモリに入出力する場合に使用するデータの型を指定します. (参照 2.3.2.3)
Elem[=<要素数>]	変数の数を指定します. 16 進数で指定する場合は以下のフォーマットで入力してください. 0x[0-9,A-F]+, &H[0-9,A-F]+, [0-9,A-F]+H (デフォルト:1) (参照 2.3.2.4)
Array[=< True or False >]	一要素の読み込み時も配列の形式で値を取得するかどうかを指定します. (デフォルト:FALSE) (参照 2.3.2.5)

2.3.2.1. Path オプション

Path オプションにデバイスコードとデバイス番号を指定することで，対象のデバイスにアクセスすることが出来ます．

<デバイス> ：

ビットデバイス ： X，Y，M，B，F，M，TS，TC，CS，CC

ワードデバイス ： TN，CN，D，W，R

<デバイス番号> ： デバイスで指す変数のアドレス．アドレスは 10 進数，16 進数で指定します．

※ワード単位指定のときビットデバイスのデバイス番号は，必ず 16 で割り切れるデバイス番号にしてください．

(例) “Path=X1F” 入力 X15 値にアクセスします．

”Path=CC255” カウンタ(コイル)CC255 値にアクセスします．

表 2-4 デバイス一覧

デバイス		デバイスコード	種別	アドレス指定方法
入力		X	ビット	16 進数
出力		Y		
内部リレー (ラッチリレー，ステップリレーを含む)		M		10 進数
リンクリレー		B		16 進数
アナンシェータ		F		10 進数
特殊リレー		M		
タイマ	接点	TS		
	コイル	TC		
	現在値	TN	ワード	
カウンタ	接点	CS	ビット	
	コイル	CC		
	現在値	CN	ワード	
データレジスタ		D		
リンクレジスタ		W		16 進数
ファイルレジスタ		R		10 進数
特殊レジスタ		D		

2.3.2.2. Param オプション

以下に Param 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(“[ ]”)内のパラメータは省略可能を示します.

各要素は設定値を 16 進数で指定桁数分設定する必要があります. 不足桁数は 0 埋めしてください[4].

【Ethernet デバイス】

“[Param=[<PCNo>[:<CPU Timer>]]]”

< PCNo >	: PC 番号(2 桁). デフォルトは FF_H（自局）です.
< CPU Timer >	: CPU 監視タイマ（単位 250ms）(4 桁). デフォルトは 0000_H（無限待ち）です. 設定値 0001_H＝0.25 秒, $000A_H$＝2.5 秒.

(例) “Path=X0, Param=FF:0000” X0 値を取得します

【シリアルデバイス】

“[Param=[<StationNo>:<PCNo>:<Wait Time>[:<BlockNo >]]]”

< StationNo >	: 局番号(2桁). (00_H～$1F_H$) デフォルトは00_Hです.
< PCNo >	: PC 番号(2 桁). デフォルトは FF_H です.
< Wait >	: 伝文ウェイト(2桁). (00_H～$0F_H$) デフォルトは$0A_H$ (100ms)です. <伝文ウェイト時間> = <設定値> × 10ms
< BlockNo >	: ブロック番号(2 桁). デフォルトは 00_H です.

(例) “Path=X10”, “Param=01:FF:0A:00” 局番号 1 の X10 値を取得します

[4] 各要素の設定値についてはユニットのユーザーズマニュアルを参照ください.

2.3.2.3. VT オプション

読み書きするデータ型と一要素あたりの点数を指定します(1 点 = 1 bit).

VT オプションの省略時は, Path オプションに指定したデバイスによりデフォルトの値が設定されます. Path オプションにビットデバイスを指定した場合は“BIT”が, ワードデバイスを指定した場合は“I2”がデフォルトの値となります.

“[VT=[<VT オプション文字列>]]”

(例) “Path=X0000, VT=I2” X0000 から X000F の値をワード値(2Byte)として読み書きします

表 2-5 指定可能な VT オプションの一覧

VT オプション	データ型	点数/要素数	意味
BIT	VT_I2	1 点	ビット単位(1 点単位)で読み書きします. 注)ビットデバイス(X,Y,M 等)のみに指定できます.
BOOL	VT_BOOL	1 点	ビット単位(1 点単位)で読み書きします. 注)ビットデバイス(X,Y,M 等)のみに指定できます.
I1	VT_I1	8 点	8 点単位で読み書きします. 注)Elem オプションで奇数個の要素数を指定し書き込みをした場合, 偶数個の要素として扱い, 追加した 8 点分を 0 埋めし書き込みを行います.
UI1	VT_UI1	8 点	8 点単位で読み書きします. 注)Elem オプションで奇数個の要素数を指定し書き込みをした場合, 偶数個の要素として扱い, 追加した 8 点分を 0 埋めし書き込みを行います.
I2	VT_I2	16 点	ワード単位(16 点単位)で読み書きします.
UI2	VT_UI2	16 点	16 点単位で読み書きします.
I4	VT_I4	32 点	32 点単位で読み書きします.
UI4	VT_UI4	32 点	32 点単位で読み書きします.
R4	VT_R4	32 点	32 点単位で読み書きします.
R8	VT_R8	64 点	64 点単位で読み書きします.
BSTR	VT_BSTR	8 点	ASCII(1 文字:8 bit)の文字列を読み書きします. 注)Elem オプションで指定された要素数より短い文字列が書き込みされた場合は, 残りの点を 0 埋めします.

2.3.2.4. Elem オプション

要素数を 10 進数, または 16 進数で指定します. 10 進数で指定する場合はそのまま数値を指定してください. 16 進数で指定する場合は 0x[0-9,A-F]+, &H[0-9,A-F]+, または[0-9,A-F]+H の形式で指定してください. Elem オプションを省略した場合のデフォルトの値は 1 となります.

Elem オプションで一度に交信できるサイズを超過して設定した場合は分割して交信を行います. 一度に交信できるサイズは 2.3.1.2 を参照してください.

"[Elem =[<要素数>]]"

(例) "Path=X0, Elem=5" X0 から X4 の値をビット値として取得します
(例) "Path=D10, Elem=0x10" D10 から D25 の値をワード値として取得します
(例) "Path=D10, Elem=&H10" D10 から D25 の値をワード値として取得します
(例) "Path=D10, Elem=10H" D10 から D25 の値をワード値として取得します
(例) "Path=M100, VT=I2, Elem=2" M100 から M131 の値をワード単位で取得します

2.3.2.5. Array オプション

Elem オプションで 1 指定し，尚且つ VT オプションで BSTR 以外を指定した場合，読み込んだ値を配列の形式で取得するかどうかを指定します．True を指定した場合は配列の形式で，False を指定した場合は指定したデータ型の形式となります．Array オプションを省略した場合のデフォルトの値は False となります．

“[Array=[< True or False >]]”

(例) “Path=X0, VT=BOOL, Elem=1, Array=True” X0 の値を BOOL 型の配列として取得します
(例) “Path=X0, VT=BOOL, Elem=1, Array=False” X0 の値を BOOL 型として取得します

2.3.2.6. サンプルプログラム

AddVariableのサンプルプログラムを以下に示します.

例) ビットデバイスMの配列(要素数10)に値を設定する.

```
HRESULT hr = S_OK;
ICaoEngine* pEng = NULL;
ICaoWorkspaces *pWss = NULL;
ICaoWorkspace *pWs = NULL;
ICaoController *pCtrl = NULL;
ICaoVariable *pVar = NULL;
CComVariant vntGet;

// CaoEngine の生成
hr = CoCreateInstance(CLSID_CaoEngine,
                      NULL,
                      CLSCTX_LOCAL_SERVER,
                      IID_ICaoEngine,
                      (void **)&pEng);
if (FAILED(hr)) {
    goto EndProc;
}

// CaoWorkspace コレクションの取得
hr = pEng->get_Workspaces(&pWss);
if (FAILED(hr)) {
    goto EndProc;
}

// CaoWorkspace の取得
hr = pWss->Item(CComVariant(0L), &pWs);
if (FAILED(hr)) {
    goto EndProc;
}

// CaoController の生成
hr = pWs->AddController(CComBSTR(L"MELSEC_AnA"),
                        CComBSTR(L"CaoProv.MELSEC.AnA"),
                        CComBSTR(L""),
                        CComBSTR(L"Conn=ETH:192.168.2.1:1026"),
                        &pCtrl);
if (FAILED(hr)) {
    goto EndProc;
}

// 変数の生成
hr = pCtrl->AddVariable(CComBSTR(L"BIT_DEVICE_M"), CComBSTR(L"Path=M16, Elem=10"), &pVar);
if (FAILED(hr)) {
    goto EndProc;
}
```

```
    // 値の設定, 取得
    CComVariant vntPut
    vntPut.vt     = (VT_I2 | VT_ARRAY);
    vntPut.parray = SafeArrayCreateVector(VT_I2, 0, 10);
    pVar->put_Value(vntPut);
    pVar->get_Value(&vntGet);

EndProc:
    if (pVar) pVar->Release();
    if (pCtrl) pCtrl->Release();
    if (pWs) pWs->Release();
    if (pWss) pWss->Release();
    if (pEng) pEng->Release();
```

2.4. エラーコード

MELSEC-A 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．また，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 2-6 固有エラーコード

エラー名	エラー番号	説明
内部エラー	0x80100000	Ethernet またはシリアル通信における内部エラー．
シーケンサエラー	0x8010yyxx	シーケンサでエラーが発生した場合は，シーケンサのエラーコードを yyxx の箇所に入れて返します．xx が 0x5B の場合，異常コードが yy の個所に格納されます．xx が 0x5B 以外の場合は yy の個所は 0x00 が格納されます． エラーコードの内容についてはシーケンサのリファレンスを参照してください．
サムチェックエラー	0x80100100	シーケンサから受信した伝文のサムチェックが不正
受信データフォーマット異常	0x80110000	シーケンサから受信した伝文が想定外の異常なフォーマットであった場合に返ります．
受信データ欠落	0x80110001	シーケンサから受信した伝文の応答パケットのサイズが十分なサイズを満たしていない場合に返ります．

3. 通信プロトコルコマンド対応表

本プロバイダで実装している Variable 変数で対応しているコマンド一覧を表 3-1 示します.

表 3-1 通信コマンド対応表

機能		コマンド	説明	指定例
一括読出し	ビット単位	BR	ビットデバイスを 1 点単位で読み出す.	Path=X0, VT=BIT Path=X0, VT=BOOL
	ワード単位	WR	ビットデバイスを 16 点単位で読み出す.	Path=X0, VT=I1
			ワードデバイスを 1 点単位で読み出す.	Path=D0, VT=I2
一括書込み	ビット単位	BW	ビットデバイスを 1 点単位で書き込む.	Path=X0, VT=BIT Path=X0, VT=BOOL
	ワード単位	WW	ビットデバイスを 16 点単位で書き込む.	Path=X0, VT=I1
			ワードデバイスを 1 点単位で書き込む.	Path=D0, VT=I2