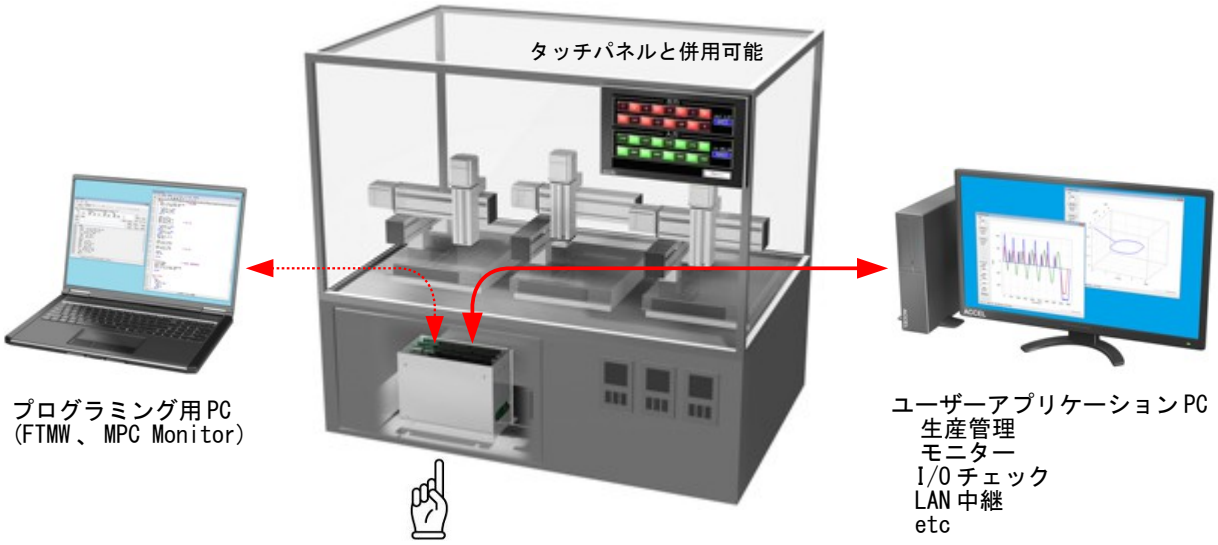
Technical Information		Ref No: ti2k-151109	Last Modify 151116
Title	MEWNET プロトコル通信 VB.net 用ライブラリ		

概要

MEWNET プロトコルで MPC の点データ、変数、配列変数、実 I/O、メモリー I/O の読み書きが可能です。VB 用の通信ライブラリを用いてパソコンから容易にアクセスできます。
ライブラリはソースで公開しているので、ユーザーの仕様に合わせてカスタマイズできます。
[開発ツールダウンロード] http://www.departonline.jp/acceleng/dev_uty.php

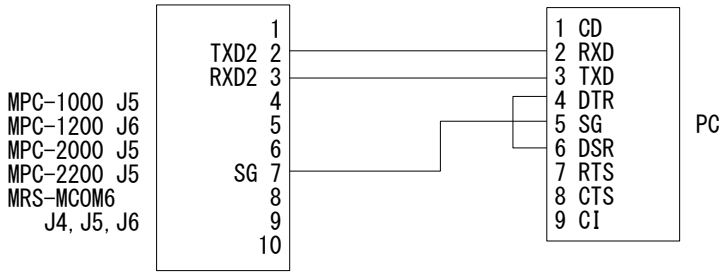
イメージ



MPC プログラムの MEWNET 宣言例

```
MEWNET 38400 1       /* Ch1  タッチパネル用
MEWNET 38400 2       /* Ch2 PC 用
```

Ch2 接続例

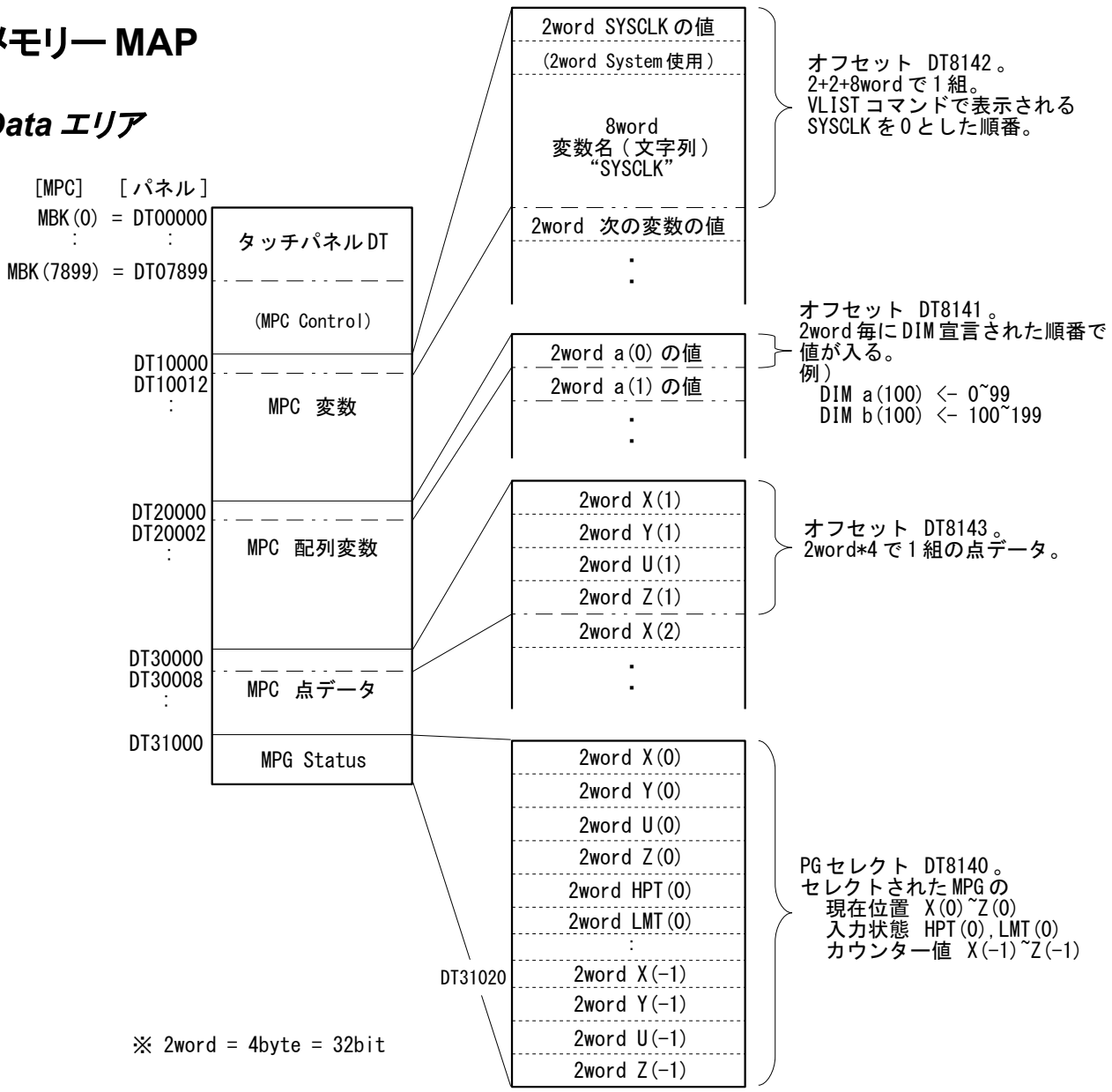


USB シリアルコンバーター「USB-RS」で直結できます（推奨）。
USB-RS のデバイスドライバは高速通信に設定されています。

USB-RS5(CEP-113E) は JP1 のパターンをカットします。
(MPC-2000 J5 へ接続する場合はカット不要です）

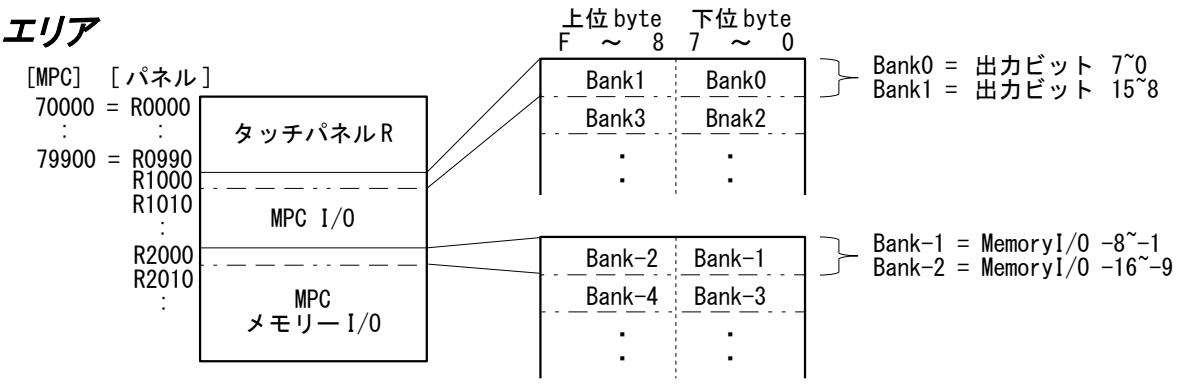
メモリー MAP

Data エリア



オフセットには各エリアの先頭番号を入れます。例えば変数の場合、DT8142=0 なら DT10000 は SYSCLK、DT8142=10 とすると DT10000 は SYSCLK から+10 の変数になります。
点データは 最小値=1 で、DT8143=1 なら DT30000 は P(1)になります。
上図は各先頭を初期値とした場合です。

I/O エリア



I/O エリアは連続したアドレスです。オフセット設定はありません。

MEWNET プロトコルを使用する利点

MEWNET プロトコルは MPC に組み込まれている通信手順です。MPC 側はプログラム冒頭で MEWNET 宣言をするだけでスレーブとなり、ホスト(パソコン)からのコマンドに応答します。汎用の RS-232 通信のような PRINT,INPUT 等によるユーザープログラムを記述する必要がありません。

mewnet.vb の機能

シリアルポート オープン、クローズ

```
''' <summary>
''' シリアルポート オープン
''' </summary>
''' <param name="Portnum">ポート番号</param>
''' <remarks></remarks>
Public Sub Serial_Open(ByVal Portnum As Integer)

''' <summary>
''' シリアルポート クローズ
''' </summary>
''' <remarks></remarks>
Public Sub Serial_Close()
```

MBK Data エリア操作

```
''' <summary>
''' MBK Data パラレル出力
''' </summary>
''' <param name="dat">出力データ</param>
''' <param name="adr">アドレス</param>
''' <param name="siz">サイズ</param>
''' <remarks></remarks>
Public Sub mbk_dt_out(ByVal dat As Integer, ByVal adr As Integer, ByVal siz As Integer)

''' <summary>
''' MBK Data パラレル入力
''' </summary>
''' <param name="adr">アドレス</param>
''' <param name="siz">サイズ</param>
''' <returns>入力データ</returns>
''' <remarks></remarks>
Public Function mbk_dt_in(ByVal adr As Integer, ByVal siz As Integer) As Integer

''' <summary>
''' MBK Data 文字列出力
''' </summary>
''' <param name="str">出力文字列</param>
''' <param name="adr">アドレス</param>
''' <remarks></remarks>
Public Sub mbk_write_str(ByVal str As String, ByVal adr As Integer)

''' <summary>
''' MBK Data 文字列入力
''' </summary>
''' <param name="adr">アドレス</param>
''' <returns>入力文字列</returns>
''' <remarks></remarks>
Public Function mbk_read_str(ByVal adr As Integer) As String
```

```
    ''' <summary>
    ''' MBK Data 一括出力
    ''' </summary>
    ''' <param name="topadr">先頭アドレス</param>
    ''' <param name="cnt">書き込み数 Int,WrdはMax50, LngはMax25 (MPC側の都合)</param>
    ''' <param name="siz">サイズ</param>
    ''' <param name="dat">書き込むデータを入れた配列</param>
    ''' <remarks></remarks>
    Public Sub mbk_bulkwrite(ByVal topadr As Integer, ByVal cnt As Integer, ByVal siz As Integer,
ByRef dat() As Integer)


    ''' <summary>
    ''' MBK Data 一括入力
    ''' </summary>
    ''' <param name="topadr">読み込み先頭アドレス</param>
    ''' <param name="cnt">読み込み数</param>
    ''' <param name="siz">サイズ</param>
    ''' <param name="res">結果格納配列</param>
    ''' <remarks></remarks>
    Public Sub mbk_bulkread(ByVal topadr As Integer, ByVal cnt As Integer, ByVal siz As Integer,
ByRef res() As Integer)
```

MBK I/O エリア操作

```
    ''' <summary>
    ''' MBK I/O ビット オン
    ''' </summary>
    ''' <param name="prt">ポート番号</param>
    ''' <remarks></remarks>
    Public Sub mbk_io_on(ByVal prt As Integer)

    ''' <summary>
    ''' MBK I/O ビット オフ
    ''' </summary>
    ''' <param name="prt">ポート番号</param>
    ''' <remarks></remarks>
    Public Sub mbk_io_off(ByVal prt As Integer)

    ''' <summary>
    ''' MBK I/O ビット入力
    ''' </summary>
    ''' <param name="prt">ポート番号</param>
    ''' <returns>1:On、0:Off</returns>
    ''' <remarks></remarks>
    Public Function mbk_io_sw(ByVal prt As Integer) As Integer

    ''' <summary>
    ''' MBK I/O パラレル出力
    ''' </summary>
    ''' <param name="dat">出力データ</param>
    ''' <param name="adr">アドレス</param>
    ''' <param name="siz">サイズ</param>
    ''' <remarks></remarks>
    Public Sub mbk_io_out(ByVal dat As Integer, ByVal adr As Integer, ByVal siz As Integer)

    ''' <summary>
    ''' MBK I/O パラレル入力
    ''' </summary>
    ''' <param name="adr">アドレス</param>
    ''' <param name="siz">サイズ</param>
    ''' <returns>入力データ</returns>
    ''' <remarks></remarks>
    Public Function mbk_io_in(ByVal adr As Integer, ByVal siz As Integer) As Integer
```

実I/O 操作

```
''' <summary>
''' 実I/O ビット入力
''' </summary>
''' <param name="prt">アドレス</param>
''' <returns>入力データ</returns>
''' <remarks></remarks>
Public Function rio_sw(ByVal prt As Integer) As Integer

''' <summary>
''' 実 I/O パラレル入力
''' </summary>
''' <param name="adr">アドレス</param>
''' <returns>入力データ</returns>
''' <remarks></remarks>
Public Function rio_in(ByVal adr As Integer, ByVal siz As Integer) As Integer

''' <summary>
''' 実I/O ビット オン
''' </summary>
''' <param name="prt">ポート番号</param>
''' <remarks></remarks>
Public Sub rio_on(ByVal prt As Integer)

''' <summary>
''' 実I/O ビット オフ
''' </summary>
''' <param name="prt">ポート番号</param>
''' <remarks></remarks>
Public Sub rio_off(ByVal prt As Integer)

''' <summary>
''' 実I/O パラレル出力
''' </summary>
''' <param name="dat">出力データ</param>
''' <param name="adr">アドレス</param>
''' <param name="siz">サイズ</param>
''' <remarks></remarks>
Public Sub rio_out(ByVal dat As Integer, ByVal adr As Integer, ByVal siz As Integer)
```

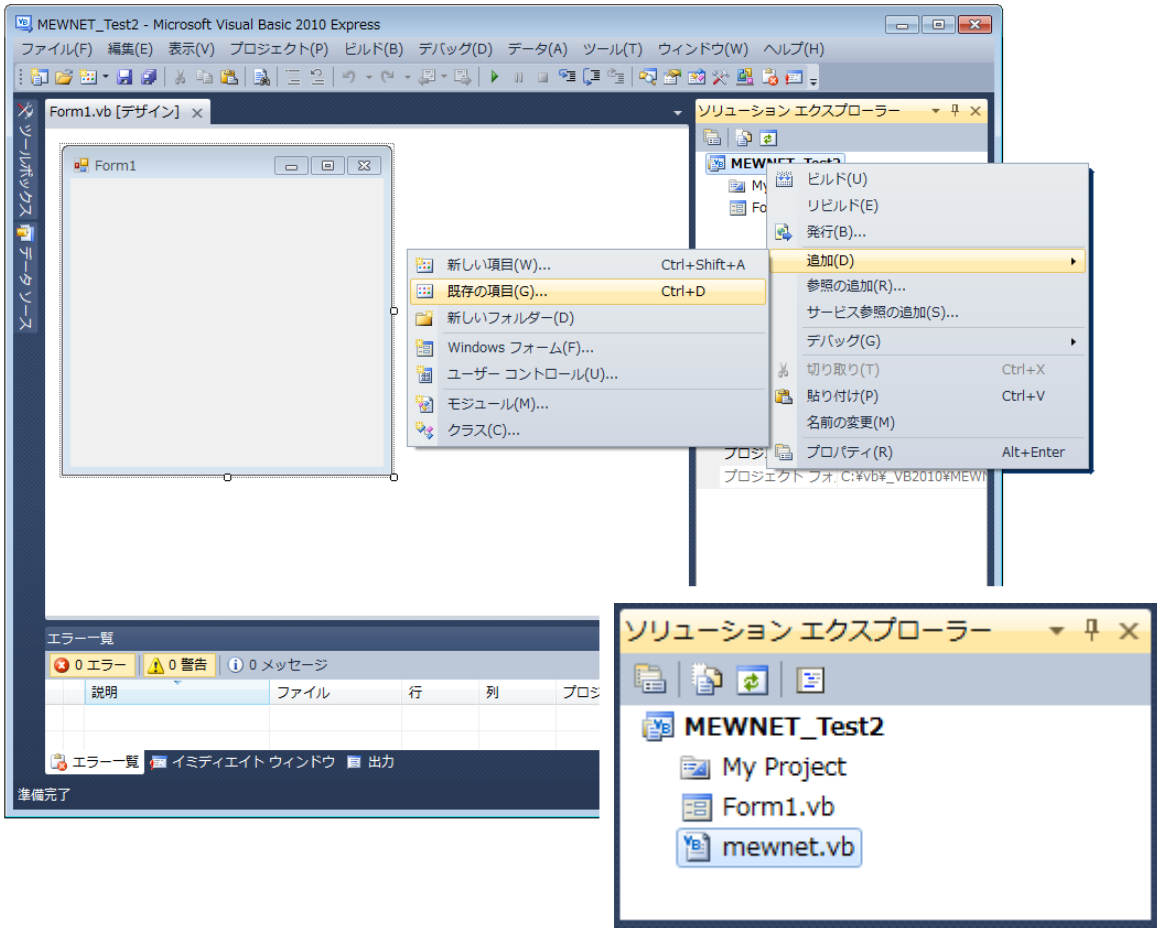
メモリーI/O 操作

```
''' <summary>
''' メモリーI/O ビット入力
''' </summary>
''' <param name="prt">アドレス</param>
''' <returns>入力データ</returns>
''' <remarks></remarks>
Public Function mio_sw(ByVal prt As Integer) As Integer

''' <summary>
''' メモリーI/O パラレル入力
''' </summary>
''' <param name="adr">アドレス</param>
''' <returns>入力データ</returns>
''' <remarks></remarks>
Public Function mio_in(ByVal adr As Integer, ByVal siz As Integer) As Integer

''' <summary>
''' メモリーI/O ビット オン
''' </summary>
''' <param name="prt">ポート番号</param>
''' <remarks></remarks>
Public Sub mio_on(ByVal prt As Integer)
```

```
    ''' <summary>
    ''' メモリーI/O ビット オフ
    ''' </summary>
    ''' <param name="prt">ポート番号</param>
    ''' <remarks></remarks>
    Public Sub mio_off(ByVal prt As Integer)

    ''' <summary>
    ''' メモリーI/O パラレル出力
    ''' </summary>
    ''' <param name="dat">出力データ</param>
    ''' <param name="adr">アドレス</param>
    ''' <param name="siz">サイズ</param>
    ''' <remarks></remarks>
    Public Sub mio_out(ByVal dat As Integer, ByVal adr As Integer, ByVal siz As Integer)
```

Visual Studio へ追加

(VB2010Expressの例)
　プロジェクトを新規作成→保存し、Form1.vbと同じフォルダーにmewnet.vbをコピーします。
ソリューションエクスプローラーでプロジェクトを逆クリック→追加→既存の項目でmewnet.vbを追加します。



追加後

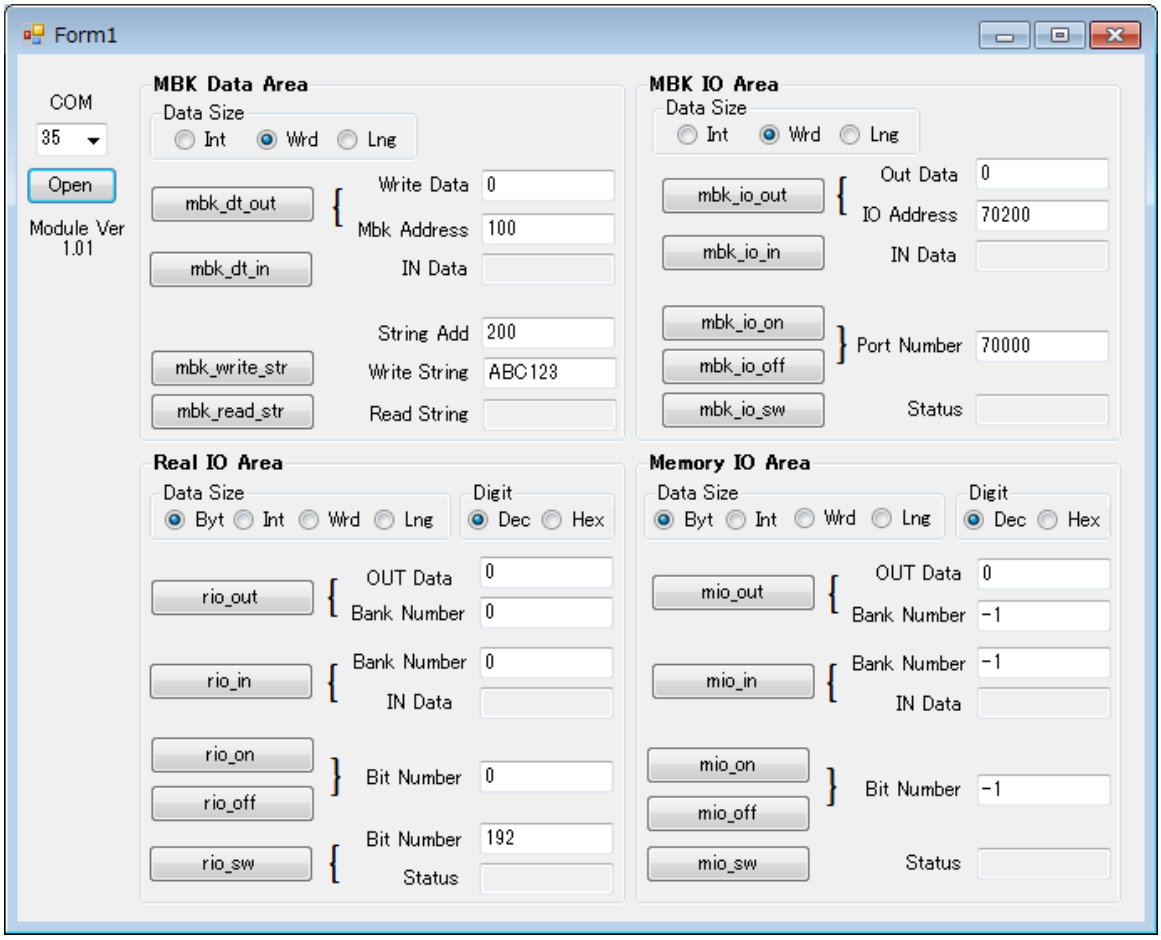
Public Class Form1の下に MEWNET を定義します。

```
Public Class Form1
    Private MEWNET As New mewnet

End Class
```

※サンプルアプリケーションのプロジェクトはホームページに掲載しています。

サンプルアプリケーション 1: MEWNET_ProtocolTest

mewnet.vb の動作・使用方法確認用のアプリケーションです。機能別に動作をチェックできます。



VB コード抜粋

シリアルポート オープン

```
Try
    MEWNET.Serial_Open(ComboBox1.Text)
Catch ex As Exception
    MessageBox.Show(ex.Message, Application.ProductName)
End Try
```

MBKエリア パラレル出力

```
Dim siz As Integer
If RadioButton_dtInt.Checked Then siz = MEWNET.Size.Int
If RadioButton_dtWrd.Checked Then siz = MEWNET.Size.Wrd
If RadioButton_dtLng.Checked Then siz = MEWNET.Size.Lng
Try
    MEWNET.mbk_dt_out(TextBox1.Text, TextBox2.Text, siz)
Catch ex As Exception
    MessageBox.Show(ex.Message, Application.ProductName)
End Try
```

MBKエリア パラレル入力

```
Dim siz As Integer
If RadioButton_dtInt.Checked Then siz = MEWNET.Size.Int
If RadioButton_dtWrd.Checked Then siz = MEWNET.Size.Wrd
If RadioButton_dtLng.Checked Then siz = MEWNET.Size.Lng
Try
    TextBox3.Text = MEWNET.mbk_dt_in(TextBox2.Text, siz)
Catch ex As Exception
    MessageBox.Show(ex.Message, Application.ProductName)
End Try
```

MBKエリア ビットオン

```
Try
    MEWNET.mbk_io_on(TextBox4.Text)
Catch ex As Exception
    MessageBox.Show(ex.Message, Application.ProductName)
End Try
```

MBK エリア ビット入力

```
Try
    TextBox5.Text = MEWNET.mbk_io_sw(TextBox4.Text)
Catch ex As Exception
    MessageBox.Show(ex.Message, Application.ProductName)
End Try
```

実IO ビットオン

```
Try
    MEWNET.rio_on(TextBox14.Text)
Catch ex As Exception
    MessageBox.Show(ex.Message, Application.ProductName)
End Try
```

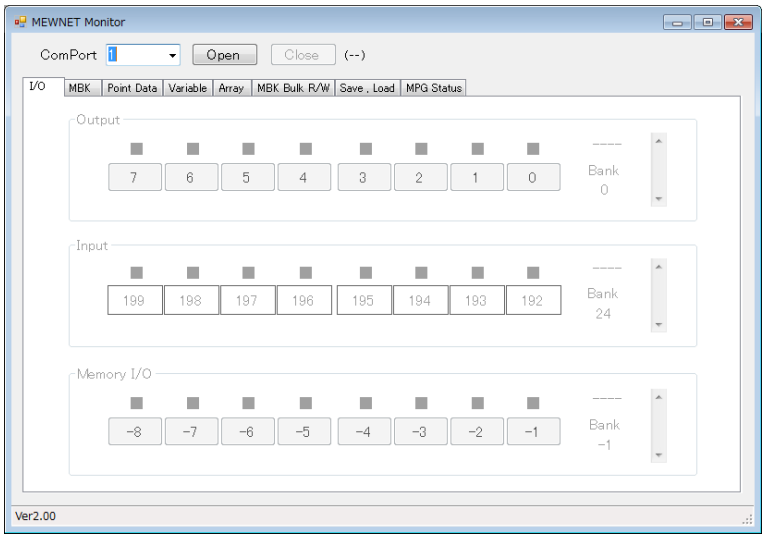
実IO ビット入力

```
Try
    TextBox16.Text = MEWNET.rio_sw(TextBox15.Text)
Catch ex As Exception
    MessageBox.Show(ex.Message, Application.ProductName)
End Try
```

サンプルアプリケーション 2: MEWNET Monitor

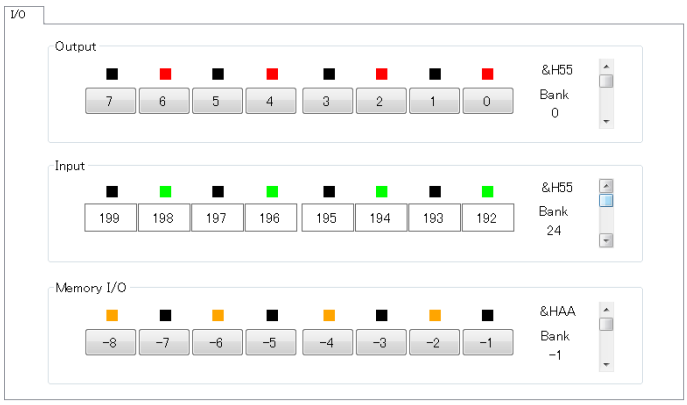
MEWNET_Monitor.exe は mewnet.vb のサンプルアプリケーションですが、実用的に I/O チェッカやデバッガとしても使えます。プログラムを止めることなく、点データ・MBK データの保存・読込みができるので、ログの取得や段取替えのデータ入れ替えなどにも応用できます。

起動画面



COM ポート番号を選択して Open ボタンを押すとモニターを開始します。

I/O タブ

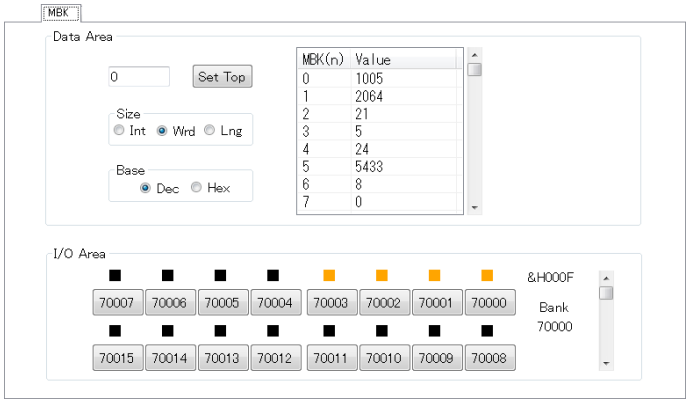


出力のモニタ、ON/OFF

入力のモニタ

メモリ I/O のモニタ、ON/OFF

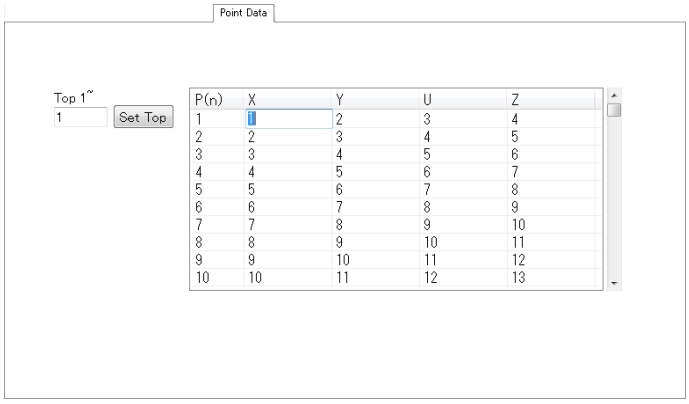
MBK タブ



MBK(タッチパネル DT) エリアモニタ、変更

I/O(タッチパネル R) エリアのモニタ、ON/OFF

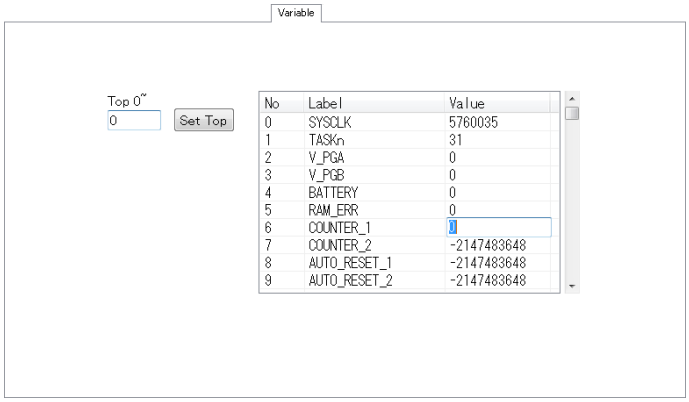
Point Data タブ



点データのモニタ、変更

データ欄をクリックすると変更できます。

Variable タブ

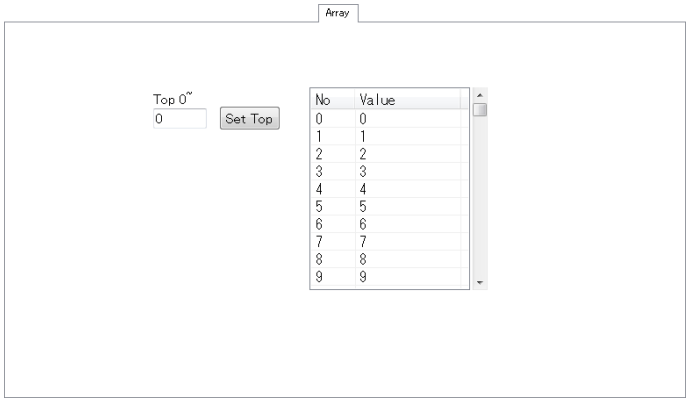


変数のモニタ、変更

VLIST で表示される変数の一覧で、 SYSLCK を No0 としています。

Value をクリックすると変更できます。

Array タブ



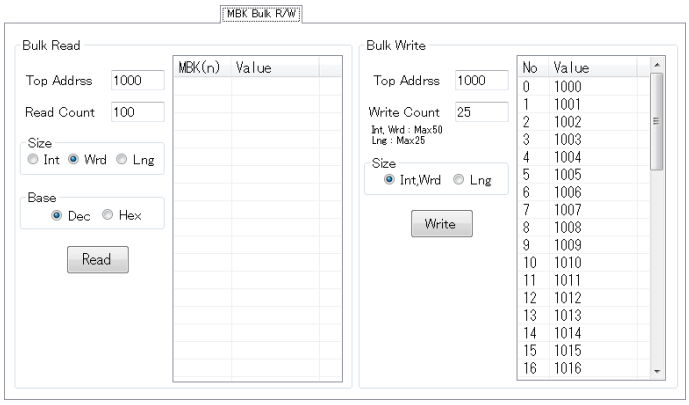
配列変数のモニタ、変更

DIM で宣言された順番です。

例）プログラムで
  DIM a(100)
  DIM b(100)
と宣言すると
a() は No0 ～ 99 、 b() は No100 ～ 199
となります。

Value をクリックすると変更できます。

MBK Bulk R.W タブ



MBK(DT) エリアー括書込み、一括読込み

書込む値は Value をクリックすると変更できます。

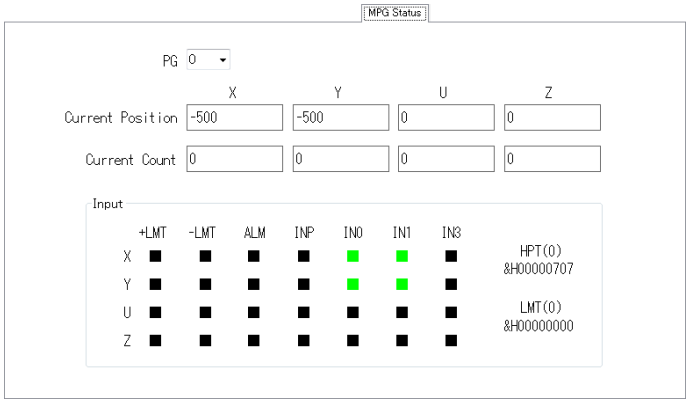
Save,Load タブ

点データと MBK(DT) エリアの保存、読込み

保存は範囲を指定、読込みはファイル単位です。

点データの保存、読込みはオフセットを切替ながら作業します。タッチパネルで点データを表示していると、表示箇所が変ることがあります。

MPG Status タブ



選択した MPG の
現在位置、カウンタ
各入力の状態です。
変更はできません。

MPG-2314 以外の入力は無効です。

サンプルアプリケーション 3: 加減速簡易実測

４次元配列の点データエリアは計測値などの記録エリアとして便利に使えます。通常、記録したデータは FTMW や MPC Monitor などでパソコンに保存して後処理をしますが、ここでは点データの SAVE を応用してデータ取り出しとグラフ描画をするアプリケーションを作ってみました。取り出したデータは表計算ソフト用にカンマ区切りの CSV でも保存できます。

MPC のプログラム実行を止めずにデータを取り出す時だけパソコンを接続することもできます(静電気、ノイズ印加に注意してください)。

MPC のプログラム

これは MPC-1200 用ですが PG 指定を変えれば MPG-2314 でも動作します。このプログラムは自分が出力したパルス数を単位時間あたりで点データに記録します。例えば　interval=2 なら 2msec ごとにその間のパルス量を点データに記録します。

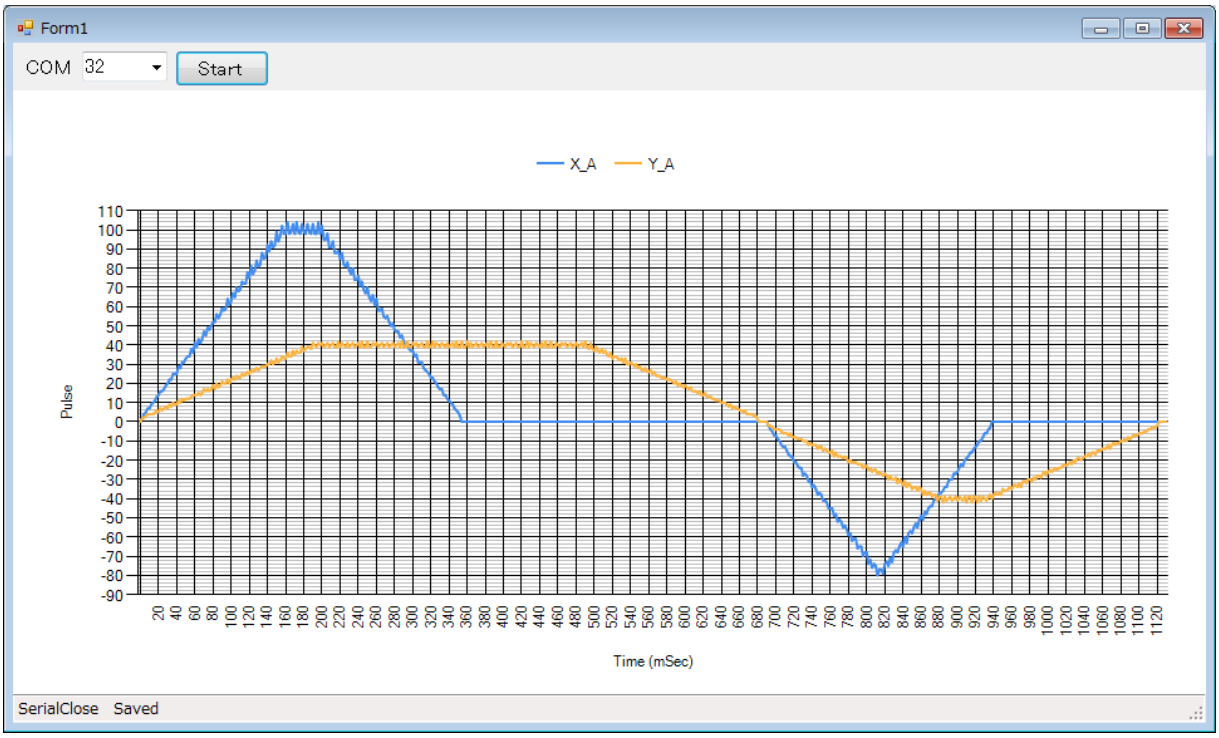
```
  MEWNET 38400 1
  QUIT_FORK 1 *main
  END

*main
  FILL P(0) 20000              /* 点データ初期化
  PG 17                        /* MPC-1200 パルス発生
  ACCEL X_A 50000 4000 1000    /* X 軸加減速
  ACCEL Y_A 20000 2000 1000    /* Y 軸加減速

  DO
    WAIT SW(192)==1
    WAIT SW(192)==0
    CLRPOS X_A|Y_A             /* 現在位置クリア
    QUIT_FORK 2 *record        /* 記録開始
    RMVS 10000 10000           /* CW パルス出力
    WAIT RR(X_A|Y_A)==0
    TIME 10
    RMVS -5000 -5000           /* CCW パルス出力
    WAIT RR(X_A|Y_A)==0
    TIME 10
    QUIT 2                     /* 記録終了
  LOOP

*record                        /* 記録タスク
  PG 17
  oldposX=0
  oldposY=0
  p_=1000
  SYSCLK=0
  interval=2                   /* 記録間隔 msec
  DO
    WAIT SYSCLK%interval==0
    X(p_)=SYSCLK               /* 経過時間記録
    nowposX=X(0)               /* X 現在位置
    nowposY=Y(0)               /* Y 現在位置
    difposX=nowposX-oldposX    /* X 差分
    oldposX=nowposX
    difposY=nowposY-oldposY    /* Y 差分
    oldposY=nowposY
    Y(p_)=difposX              /* X 差分記録
    U(p_)=difposY              /* Y 差分記録
    p_=p_+1
    WAIT SYSCLK%interval<>0
    SWAP
  LOOP
```

P(1000)を先頭にして、X()が経過時間、Y()がＸ軸のパルス数、U()がＹ軸のパルス数です。実行すると点データは次のようになります(これは通常の点データ.P2K のフォーマットです)

```
SETP 1000   0 0 0 0
SETP 1001   2 2 2 0
SETP 1002   4 4 2 0
SETP 1003   6 5 3 0
SETP 1004   8 7 4 0
SETP 1005   10 7 4 0
SETP 1006   12 9 4 0
SETP 1007   14 10 4 0
SETP 1008   16 11 5 0
SETP 1009   18 13 6 0
      （以下略）
```

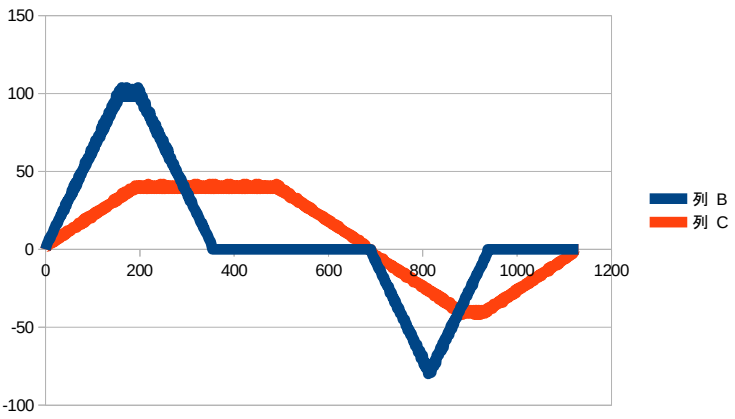
VB サンプルプログラムの実行結果

MPC プログラムを RUN 後、アプリケーションを実行しました。横軸は経過時間、縦軸は 2msec ごとのパルス数で、加減速の様子がわかります。RMVS は直線補間をしない、Ｘ軸は１回目の動作では最高速が出ているが、２番目は移動量が少なくて最高速まで到達していないことがわかります。



下左はこのアプリケーションで保存したデータです(拡張子 CSV)。経過時間,X パルス量,Y パルス量の順です。右はそれを表計算ソフトに読み込んでグラブウィザードでグラフ化したものです。

```
0, 0, 0
2, 2, 2
4, 4, 2
6, 5, 3
8, 7, 4
10, 7, 4
12, 9, 4
14, 10, 4
16, 11, 5
18, 13, 6
20, 14, 5
（以下略）
```



VB コード

```
Imports System.Windows.Forms.DataVisualization.Charting

Public Class Form1

    Private MEWNET As New mewnet

    Private Sub Form1_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load
        Panel2.Dock = DockStyle.Fill
        Chart1.Dock = DockStyle.Fill
        For i = 1 To 99
            ComboBox_ComNum.Items.Add(i)
        Next
        ComboBox_ComNum.Text = "Select"
        ChartClear(Chart1)
        ToolStripStatusLabel1.Text = ""
        ToolStripStatusLabel2.Text = ""

    End Sub

    ''' <summary>
    ''' Start ボタン
    ''' </summary>
    ''' <param name="sender"></param>
    ''' <param name="e"></param>
    ''' <remarks></remarks>
    Private Sub Button1_Click(sender As System.Object, e As System.EventArgs) Handles Button1.Click
        Static PointDataName As String = ""
        Dim res(399) As Integer
        Dim buf As String = ""

        Try
            ChartClear(Chart1)
            MEWNET.Serial_Open(ComboBox_ComNum.Text)
            ToolStripStatusLabel1.Text = "SerialOpen"

            Dim StartOfset As Integer = 1000
            Dim Range As Integer = 200 * 100
            Dim PintNo As Integer = StartOfset

            Dim ChartMaxY As Integer = 0
            Dim ChartMimY As Integer = 0
            Dim chart_x_cnt As Integer = 0
            For OfSet = StartOfset To StartOfset + Range - 1 Step 100
                MEWNET.mbk_dt_out(OfSet, 8143, MEWNET.Size.Wrd)  ' Offset Value
                MEWNET.mbk_bulkread(30000, res.Length, MEWNET.Size.Lng, res)

                Dim i As Integer
                For i = 0 To 99
                    If (res(i * 4) = 0) And (OfSet > StartOfset + 10) Then
                        Exit For
                    End If
                    'buf += "SETP " & PintNo
                    buf += res(i * 4).ToString          ' X
                    buf += "," & res(i * 4 + 1)         ' Y
                    buf += "," & res(i * 4 + 2)         ' U
                    'buf += " " & res(i * 4 + 3)        ' Z
                    buf += vbCrLf
                    PintNo += 1

                    ' Chartにプロットする
                    Chart1.Series("X_A").Points.AddY(res(i * 4 + 1))
                    Chart1.Series("Y_A").Points.AddY(res(i * 4 + 2))
                    If ChartMaxY < res(i * 4 + 1) Then ChartMaxY = res(i * 4 + 1)
                    If ChartMaxY < res(i * 4 + 2) Then ChartMaxY = res(i * 4 + 2)
                    If ChartMimY > res(i * 4 + 1) Then ChartMimY = res(i * 4 + 1)
                    If ChartMimY > res(i * 4 + 2) Then ChartMimY = res(i * 4 + 2)

                    If (chart_x_cnt Mod 10) = 0 Then
                        Chart1.ChartAreas(0).AxisX.CustomLabels.Add((chart_x_cnt + 1) * 2, 0, _
                                                                    res(i * 4).ToString)
```

```
                End If
                chart_x_cnt = chart_x_cnt + 1
                ToolStripStatusLabel2.Text = chart_x_cnt.ToString

            Next
            If i < 100 Then Exit For
            Application.DoEvents()
        Next
        'グラフY軸を最大最小値にあわせる
        Chart1.ChartAreas(0).AxisY.Maximum = (ChartMaxY ¥ 10) * 10 + 10
        Chart1.ChartAreas(0).AxisY.Minimum = (ChartMimY ¥ 10) * 10 - 10

        MEWNET.Serial_Close()
        ToolStripStatusLabel1.Text = "SerialClose"
        ToolStripStatusLabel2.Text = ""

        SaveFileDialog1.Title = "保存"
        SaveFileDialog1.FileName = PointDataName
        SaveFileDialog1.Filter = "CSV File|*.CSV"
        If SaveFileDialog1.ShowDialog() = DialogResult.OK Then
            PointDataName = SaveFileDialog1.FileName
            Dim sw As New System.IO.StreamWriter(PointDataName, _
                                False, System.Text.Encoding.GetEncoding("shift_jis"))
            sw.Write(buf)
            sw.Close()
            ToolStripStatusLabel2.Text = " Saved"
        End If

    Catch ex As Exception
        MessageBox.Show(ex.Message & vbCrLf, Application.ProductName)
    End Try

End Sub


''' <summary>
''' Chart設定
''' </summary>
Private Sub ChartClear(ByVal cht As Chart)

    Dim customLabel1 As New CustomLabel

    With cht
        .Titles.Clear()                         'タイトルの初期化
        .Titles.Add("")
        .Titles.Item(0).Font = New Font("ＭＳ ゴシック", 12, FontStyle.Bold)

        .BackGradientStyle = GradientStyle.None '背景グラデーション
        .BorderSkin.PageColor = SystemColors.ButtonFace

        '外形をデフォルトに
        .BorderSkin.SkinStyle = BorderSkinStyle.None
        .Legends.Clear()                        '凡例の初期化
        .Legends.Add("Legend1")
        .Legends("Legend1").Alignment = StringAlignment.Center
        .Legends("Legend1").Docking = Docking.Top

        .Series.Clear()                         '系列(データ関係)の初期化
        .Annotations.Clear()                    'グラフの注釈
        .DataSource = Nothing

        .Series.Add("X_A")
        .Series("X_A").ChartType = SeriesChartType.Line
        .Series("X_A").BorderWidth = 2
        .Series("X_A").LegendText = "X_A"

        .Series.Add("Y_A")
        .Series("Y_A").ChartType = SeriesChartType.Line
        .Series("Y_A").BorderWidth = 2
        .Series("Y_A").LegendText = "Y_A"

        .ChartAreas.Clear()                     '軸メモリ・3D 表示関係の初期化
        .ChartAreas.Add("ChartArea1")
```

```
            .ChartAreas(0).AxisX.CustomLabels.Add(customLabel1)
            .ChartAreas(0).AxisX.LabelAutoFitMaxFontSize = 9
            .ChartAreas(0).AxisX.LabelAutoFitMinFontSize = 9

            .ChartAreas(0).AxisX.MajorGrid.Enabled = True
            .ChartAreas(0).AxisX.MajorGrid.Interval = 10
            .ChartAreas(0).AxisX.MajorGrid.LineColor = Color.Black
            .ChartAreas(0).AxisX.MajorGrid.IntervalOffset = 1
            .ChartAreas(0).AxisX.MajorTickMark.Enabled = False
            '.ChartAreas(0).AxisX.Interval = 5
            '.ChartAreas(0).AxisX.MinorGrid.Enabled = True
            .ChartAreas(0).AxisX.MinorGrid.LineColor = Color.Silver
            .ChartAreas(0).AxisX.MinorTickMark.Enabled = False
            .ChartAreas(0).AxisX.Title = "Time (mSec)"

            .ChartAreas(0).AxisY.Maximum = 200
            .ChartAreas(0).AxisY.Minimum = -10
            .ChartAreas(0).AxisY.MajorGrid.Interval = 10
            .ChartAreas(0).AxisY.Interval = 10
            .ChartAreas(0).AxisY.MinorGrid.Enabled = True
            .ChartAreas(0).AxisY.MinorGrid.LineColor = Color.Silver
            .ChartAreas(0).AxisY.Title = "Pulse"
        End With

    End Sub

End Class
```

サンプルアプリケーション 4: ILNumerics

　市販の数学ライブラリ「ILNumerics」を使った加減速と軌跡を描くツールです。点配列に記録した単位時間当たりの出力パルス数の差分および座標値を動作後に読み込んで描画します。
　このサンプリは無償の NuGet 版の ILNumerics を用いました。ILNumerics についてはネット上に多くの情報が載っていますので検索してください。

MPC のプログラム

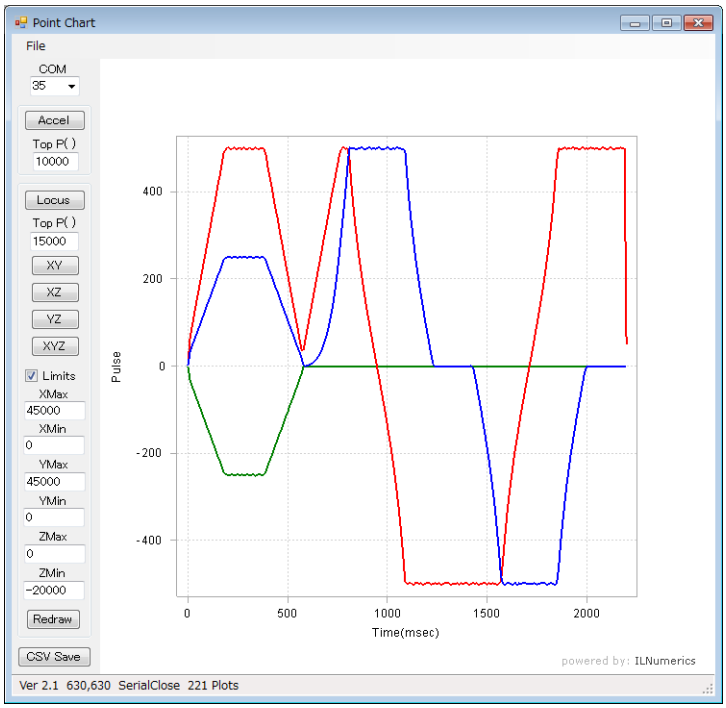
　直線円弧連続補間移動です。(ユーザーズマニュアル コマンドリファレンス MOVT 参照)

```
  MEWNET 38400 2                        /* 通信宣言
  QUIT_FORK 1 *main
  END
*main
  FILL P(10000) 10000 0                 /* 点データ初期化
  PG 0
  GOSUB *HOME
  GOSUB *SET_POINT                      /* 作業点と移動形態の作成

  QUIT_FORK 2 *PgRecord                 /* 記録開始
  CLRPOS
  ACCEL ALL_A 50000
  MOVL P(1000)                          /* スタートポイントへ移動（3軸移動）
  WAIT RR(ALL_A)==0
  DS_DACL
  FOR pnt=1001 TO X(2000)               /* P(1001)～P(1004)連続移動
   MOVT X_A|Y_A P(pnt) X(1000+pnt)      /* P(pnt)は移動先、X(1000+pnt)は円弧または直線指定
  NEXT
  EN_DACL
  WAIT RR(X_A|Y_A)==0
  TIME 10
  QUIT 2                                /* 記録終了
  END

*SET_POINT
  SETP 1000 10000 20000 0 -10000        /* スタートポイント
  SETP 1001 30000 20000 20000 20000     /* XY=円弧終点座標、UZ=円弧中心座標
  SETP 1002 30000 10000 0 0             /* XY=移動先座標
  SETP 1003 10000 10000 20000 10000     /* XY=円弧終点座標、UZ=円弧中心座標
  SETP 1004 10000 20000 0 0             /* XY=移動先座標
  SETP 2000 1004 50 0 0                 /* X=最終点番号、Y=FEED
  SETP 2001 CW 0 0 0                    /* P(1000) から P(1001) は CW
  SETP 2002 0 0 0 0                     /* P(1001) から P(1002) は 直線
  SETP 2003 CW 0 0 0                    /* P(1002) から P(1003) は CW
  SETP 2004 0 0 0 0                     /* P(1003) から P(1004) は 直線
  RETURN
*HOME                                   /* 原点復帰
  ACCEL X_A|Y_A|Z_A 10000 1000 10000
  RMVS Z_A -5000                        /* Z退避移動
  WAIT RR(Z_A)==0
  SHOM Z_A IN0_OFF|IN1_ON|CW            /* ORG入力だけ
  HOME Z_A POS_L                        /* Z原点復帰
  WAIT RR(ALL_A)==0
  RMVS 10000 10000                      /* XY退避移動
  WAIT RR(ALL_A)==0
  SHOM X_A|Y_A IN0_OFF|IN1_ON|CCW       /* ORG入力だけ
  HOME X_A|Y_A NEG_L                    /* XY原点復帰
  WAIT RR(ALL_A)==0
  RETURN
```

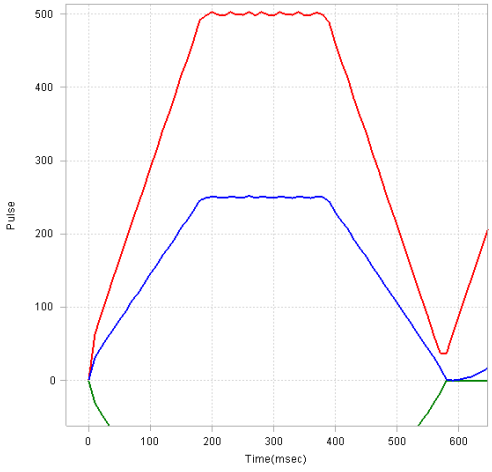
```
*PgRecord                           /* 記録タスク
  PG pgnum
  FILL P(10000) 10000 0
  oldposX_=X(0)
  oldposY_=Y(0)
  oldposZ_=Z(0)
  pointnum_=10000                   /* 加減速記録先頭点番号
  locusnum_=15000                   /* 軌跡記録先頭点番号
  SYSCLK=0
  interval_=10                      /* 記録間隔msec
  DO
    WAIT SYSCLK%interval_==0
    nowclk_=SYSCLK
    nowposX_=X(0)                   /* X現在位置
    nowposY_=Y(0)                   /* Y現在位置
    nowposZ_=Z(0)                   /* Y現在位置
    difposX_=nowposX_-oldposX_      /* X差分
    oldposX_=nowposX_
    difposY_=nowposY_-oldposY_      /* Y差分
    oldposY_=nowposY_
    difposZ_=nowposZ_-oldposZ_      /* Z差分
    oldposZ_=nowposZ_
    /*   点番号     時間      X差分    Y差分      Z差分
    SETP pointnum_ nowclk_ difposX_ difposY_ difposZ_
    pointnum_=pointnum_+1
    /*   点番号     時間     X現位置   Y現位置   Z現位置
    SETP locusnum_ nowclk_ nowposX_ nowposY_ nowposZ_
    locusnum_=locusnum_+1
    WAIT SYSCLK%interval_<>0
  LOOP
```

実行結果

加減速のグラフ。青：X軸、赤：Y軸、緑：Z軸。



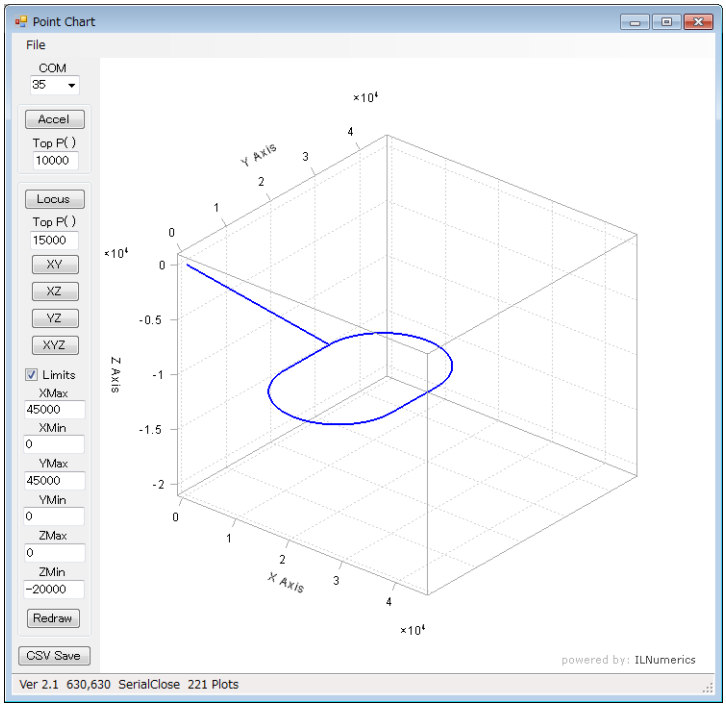
部分拡大ができます



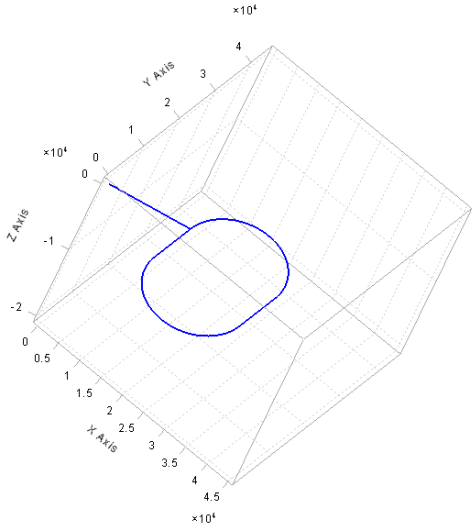
(マルチタスクの影響でジッタ状の揺らぎがあります)

軌跡

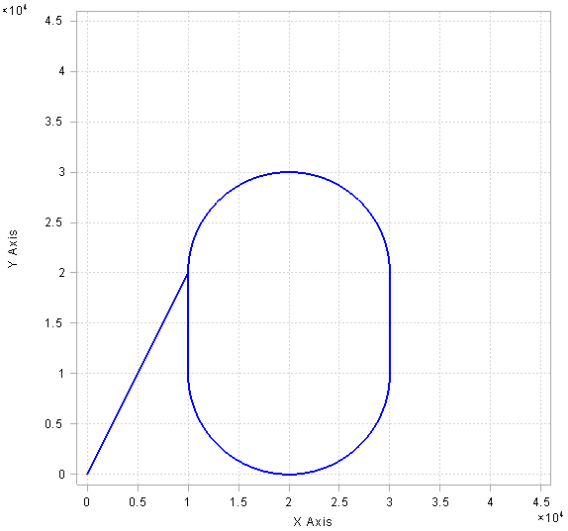
XYZ View



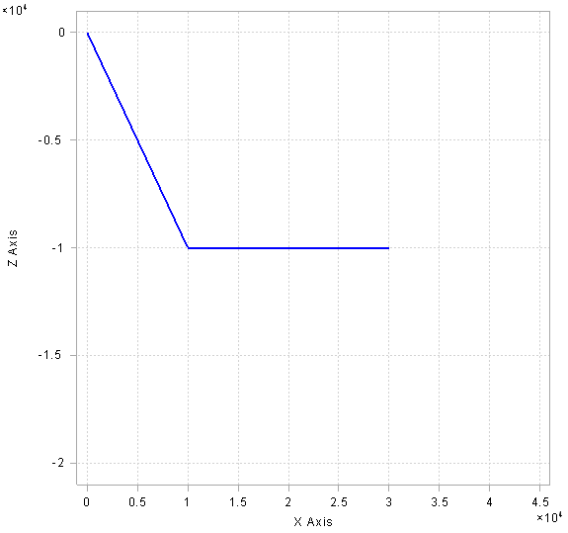
マウスドラッグで視点を変えることができます。



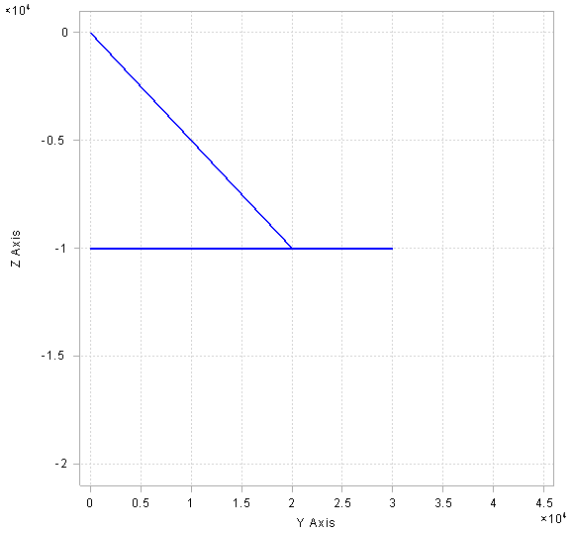
XY View



XZ View



YZ View



読み込んだデータを CSV で保存、それをオフラインで開くことができます。

MPC 内の加減速のデータ

```
SETP 10000   0 1 1 -1
SETP 10001   10 31 63 -31
SETP 10002   20 45 90 -45
SETP 10003   30 58 115 -58
SETP 10004   40 70 141 -71
SETP 10005   50 83 165 -82
SETP 10006   60 94 190 -95
SETP 10007   70 109 216 -108
SETP 10008   80 119 240 -120
SETP 10009   90 132 264 -132
SETP 10010   100 146 291 -146
SETP 10011   110 157 314 -157
SETP 10012   120 171 342 -171
SETP 10013   130 182 364 -182
SETP 10014   140 194 389 -194
(略)
```

保存した加減速の CSV データ

```
0,1,1,-1
10,31,63,-31
20,45,90,-45
30,58,115,-58
40,70,141,-71
50,83,165,-82
60,94,190,-95
70,109,216,-108
80,119,240,-120
90,132,264,-132
100,146,291,-146
110,157,314,-157
120,171,342,-171
130,182,364,-182
140,194,389,-194
(略)
```

MPC 内の座標値のデータ

```
SETP 15000   0 24 47 -24
SETP 15001   10 55 110 -55
SETP 15002   20 100 200 -100
SETP 15003   30 158 315 -158
SETP 15004   40 228 456 -229
SETP 15005   50 311 621 -311
SETP 15006   60 405 811 -406
SETP 15007   70 514 1027 -514
SETP 15008   80 633 1267 -634
SETP 15009   90 765 1531 -766
SETP 15010   100 911 1822 -912
SETP 15011   110 1068 2136 -1069
SETP 15012   120 1239 2478 -1240
SETP 15013   130 1421 2842 -1422
SETP 15014   140 1615 3231 -1616
(略)
```

保存した座標値の CSV データ

```
24,47,-24
55,110,-55
100,200,-100
158,315,-158
228,456,-229
311,621,-311
405,811,-406
514,1027,-514
633,1267,-634
765,1531,-766
911,1822,-912
1068,2136,-1069
1239,2478,-1240
1421,2842,-1422
1615,3231,-1616
(略)
```