パソコンによる３軸の制御

田頭　章司

Study ｏｎ　Ｔｈｒｅｅ　Ａｘｉｓ　Ｃｏｎｔｒｏｌ ｂｙ　ａ　Personal　Computer

Ｓｈｏｊｉ　Ｔａｇａｓｈｉｒａ

１．　はじめに

パソコンがますます家庭に入ってくる時代となりました。また、その一方でパソコンのハードウェアが高度となってきました。
パソコンを使って、３軸（Ｘ・Ｙ・Ｚ）軸を制御方法の１つの手段として、銘板彫刻器を試作することとし、まず最初の段階として、ソフト開発を行うこととしました。

パソコンを使って外部のハードウェアを制御する方法としては、パラレル・ポート制御とシリアル・ポート制御があります。

パラレル・ポート制御は比較的簡単に行えますが、いちいちパソコン本体のカバーをはずしてＰＣカードを増設しなければいけないといった面倒な面があります。

シリアル・ポート制御はシリアルコネクタにＲＳ－２３２Ｃを接続するか、また最近パソコンには必ずついているＵＳＢコネクタを接続するだけで簡単な設置でできます。

今回は、シリアル・ポート制御を用いて、３軸の制御を試してみました。

今回、Ｗｉｎｄｏｗｓ　ＭｅのＯＳ上でアプリケーション開発ソフト　Ｖｉｓｕａｌ　Ｂａｓｉｃ　６．０（以下ＶＢと呼ぶ）を用いシリアル・ポート制御ＲＳ－２３２Ｃ制御でＸ・Ｙ軸を、ＵＳＢ軸を制御することとしました。

シリアル・ポート制御およびＶＢでのプログラムの開発方法について述べたいと思います。

2.　制御機器の構成

（1）　ポジショニングシステム

Ｘ．Ｙ　ポジショナ
ユニタック社　　ＳＤ５０－３０　型
Ｚ　ポジショナ
ユニタック社　　ＳＤ６５Ａ　型
Ｘ．Ｙ　ポジショナコントロール
ユニタック社　　ＣＰＣＵ２Ａ　型
Ｚ　ポジショナコントロール
ユニタック社　　ＳＤＤ２　型

（2）　ソフトウェア
ＯＳ
Ｗｉｎｄｏｗｓ　Ｍｅ

アプリケーション
Ｖｉｓｕａｌ　Ｂａｓｉｃ　Ｖｅｒ６．０

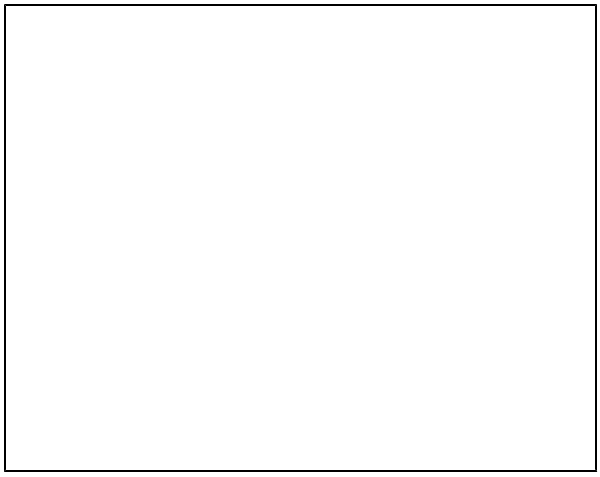


写真１　　Ｘ．Ｙ　ポジショナ・コントロール

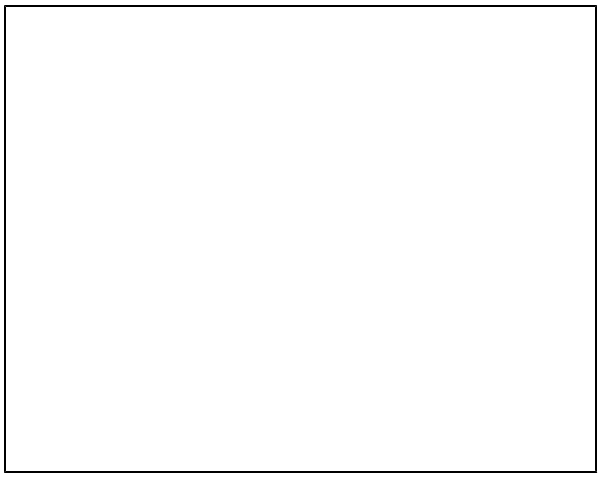


写真２　　Ｚ　ポジショナ・コントロール

３．　シリアル・ポートの制御方法

パソコンのシリアル・ポートを汎用的に使用することを考えた場合主に３つの方法があると思います。

（1）　ＭＳＣＯＭＭ３２．ＯＣＸ使った制御
（2）　ＡＰＩ　ＥＩＡ３２通信。
（3）　Ｉ／Ｏの直接制御。

今制御においては、ＲＳ－２３２Ｃによる制御については、１．ＭＳＣＯＭＭ３２．ＯＣＸを使ったシリアル・ポート制御とした。

ＵＳＢによる制御については、ＡＰＩ　ＥＩＡ３２使ったシリアル・ポート制御とした。

３．１　ＭＳＣＯＭＭ３２．ＯＣＸを使ったシリアル・ポート制御について

ＭＳＣＯＭＭ３２.ＯＣＸはボーレートに制限がなく、１１５，２００［ｂｐｓ］まで使うことが出来、

ます。
受信割り込みが使えるなど､外部ＲＳ２３２Ｃ機器との通信用インターフェスとして、便利であります。

３．２　ＡＰＩを使ったシリアル．ポート制御について

ＡＰＩはアプリケーション・プログラミング・インターフェース（Application Programing･Interface）の略で、Ｗｉｎｄｏｗｓ環境で実行するアプリケーションで使う、環境、メッセージ、データ構造、データ型、ステートメーションで構成されている。
ＶＢにない機能や、システムに係わる制御、高速化したい場合など使えるＡＰＩが多数用意されている。　［１］

４．　ＶＢでのプログラム開発

４．１　ＭＳＣＯＭＭ３２．ＯＣＸを使った制御

（１）　ツール．バーの「プロジェクト（Ｐ)」.コンポーネント（Ｏ)」からコンポーネント．ダイアログ．ボックスを開く。
（２）　コントロール．タブ内の一覧から、Ｍｉｃｒｏｓｏｆｔ.Ｃｏｍｍ　Ｃｏｎｔｏｒｏｌ６．０を選択
（３）　ツール．ボックスに図のようなアイコンがあることを確認。
４）　これで、ツール．ボックスのＭＳＣＯＭＭ．コントロールをダブルクリックしフォームに貼り付けるとと使用できる。

図１．ＭＳＣＯＭＭ　アイコン

４．２　ＤＬＬファイルの組み込み

（１）　ＤＬＬファイルとは
Ｗｉｎｄｏｗｓで使用できるすべてのアプリケーションが共通に使えるライブラリ。
ダイナミック・リンク・ライブラリ（「Ｄｙｎａｍｉｃ・Ｌｉｎｋ・Ｌａｉｂｒａｒｙ)はＷｉｎｄｏｗｓで動作する､すべてのアプリケーションが共通に使えるライブラリです。
ＶＢで使う場合、「こういうＤＬＬを使いますよ」と宣言するだけで､Ｗｉｎｄｏｗｓが組み込みや取り外しを自動的に行ってくれます。
アプリケーション側は一切気にせず、単なるサブルーチン集として使うことができる。

（２）　ＤＬＬのメリット
ＤＬＬは多重に組み込まれないので､メモリを効果的に使える。
ＤＬＬの開発は主にＣ言語を使用して開発するので、Ｃ言語を使用した。（ＶＣ＋＋）
（３）　ＤＬＬ開発言語
Ｖｉｓｕａｌ　Ｃ＋＋（ＶＣ＋＋）
ボーランド社のＣ＋＋（Ｂｕｌｄｅｒ）などがありまり､Ｂｏｒｌａｎｄ　Ｃ＋＋はＷｅｂサイトから無償で入手できるとのことですが､ＶＣ＋＋が手持ちにあるので、ＶＣ＋＋を使用した。
（４）　コンパイラ・ユーティリィ
ＤＬＬ作成の コンパイラ．ユーティリィは［１］掲載のソフトを使用しました。
（５）　ＲＳ－２３２Ｃ・ＵＳＢシリアル制御における　　組み込みファイル
Ｍｓｖｄｖｍ６０．ＤＬＬ
Ｐｓｃｏｎ．ＤＬＬ
（６）　組み込みＤＬＬファイル
ＭＳＶＢＶＭ６０．ＤＬＬ
ｐｓｃｏｎ．ＤＬＬ　（今回作成　DLL）
ＶＢ６ＪＰ．ＤＬＬ

ＵＳＢ接続、Ｍｏｄｕｌｅ１の１部分

```
'---------------------------------------------------------
'USB 接続の為に必要な処理
'---------------------------------------------------------
-'USB ドライバ DLL ファンクション
Declare Function Pscon_Open Lib "Pscon.
DLL" () As Long
Declare Function Pscon_Open_mask Lib
"Pscon.DLL"
 (ByVal flag As Long, ByVal Class As
Byte, ByVal SubClass As Byte, ByVal Vendor
 As Long, ByVal Product As Long, ByVal bcd
Device As Byte) As Long
Declare Function Pscon_Close Lib "Pscon.DLL"
(ByVal hUSB As Long) As Long
Declare Function Pscon_OpenPipe Lib "Pscon.
DLL" (ByVal hUSB As Long, ByVal Interface_
num As Byte, ByVal pipe_num As Byte) As
Long
```

```
Declare Function Pscon_ResetPipe Lib
"Pscon.DLL"（ByVal hFile As Long）As Long
Declare Function Pscon_ResetDevice Lib
"Pscon.DLL"（ByVal hUSB As Long）As Long
Declare Function Pscon_ClassRequest Lib
"Pscon.DLL"（ByVal hUSB As Long, ByVal dir_
in As Long, ByVal recipient As Byte, ByVal
bRequest As Byte, ByVal wValue As Long,
ByVal wIndex As Long, ByVal wLength As
Long, ByVal data As Long）As Long
Declare Function Pscon_VendorRequest Lib
"Pscon.DLL"（ByVal hUSB As Long, ByVal dir_
in As Long, ByVal recipient As Byte, ByVal bRe
quest As Byte, ByVal wValue As Long, ByVal
wIndex As Long, ByVal wLength As Long,
ByVal data As Long）As Long
’各種ハンドル
Global hUSB As Long          ’ＵＳＢハンドル
Global hPIPE As Long         ’リードパイプ
Global wPIPE As Long          'ライトパイプ
’オープンするＵＳＢのベンダＩＤとプロダクト
ID
Global Const Vendor = &H11D4
  ’ベンダＩＤ
Global Const Product = &H1
’マスク条件設定ビット
Global Const UU_MASK_NO = 0
Global Const UU_MASK_CLASS = 1
Global Const UU_MASK_SUBCLASS = 2
Global Const UU_MASK_VENDOR = 4
Global Const UU_MASK_PRODUCT = 8
Global Const UU_MASK_BCDDEVICE = 16
'-------------------------------------------------------
'画面の情報を保存するための変数の型を定義する
'-------------------------------------------------------
Public Type SampleInfo
      minus1 As String * 10
      plus1 As String * 10
      minus2 As String * 10
      plus2 As String * 10
      kurikaesiL As String * 10
      speed3s As String * 10
      speed3f As String * 10
      speed3r As String * 10
      speed4s As String * 10
      speed4f As String * 10
      speed4r As String * 10
          'USB 用に追加した変数
    usbplus1      As String * 10
     usbminus1    As String * 10
```

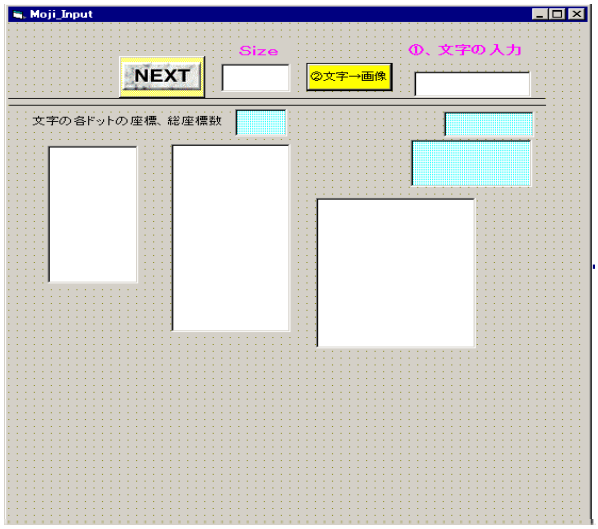
［３］

４．３
フォーム・ウィンドウ

ユーザ・インターフェースを構築するためのウィンドウとして図－２のような入力の文字をドットの座標点を取得するための入力画面フォームと図－３のようなＸ・Ｙ・Ｚ　ポジショナコントロールするためのフォーム２画面を切り替えることとしました。
また異なったオブジェクトを参照するようにしました。Ｆｏｒｍ－１（入力画面）でえた座標点のデータをＦｏｒｍ－２（Ｘ・Ｙ・Ｚ　ポジショナコントロール画面）にデータが送られるようにしました。
［２］

４．３．１　入力画面

文字入力より文字のドットの座標点を得るようにした。
各々のコマンドについて説明します。
①文字の入力面に文字を入力し、②文字→画像のコマンドをクリックすることにより、各ドットの座標点をえるようにした。
Ｓｉｚｅ面に数値を入力することにより、文字の大きさが自由に調整できるようにした。
文字より座標点をえる方法
（１）　テキストボックス入れた文字をビット・マップ（ｂｍｐ）ファイルに変換し、データを実行ファイルのあるフォルダに保存する。
（２）　テキストボックス入れた文字をピクチャーに表示。
（３）　ビット・マップ（ｂｍｐ）ファイルの作成。
（４）　ビット・マップ（ｂｍｐ）ファイルをピクチャーボックスに読み込み。
（５）　色情報を入れる。
（６）　元の画像の色情報を圧縮する。
（７）圧縮した後の高さ・幅を得る。
（８）　座標格納数を得る。
（９）　各ポイントでの色情報を得る。
以上の方法で座標点を得た。

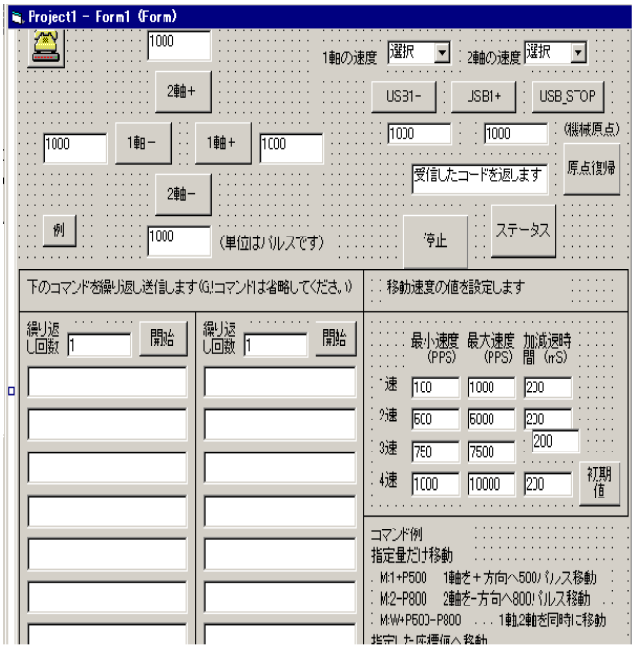


図．２　文字入力　Ｆｏｒｍ

３．１．１　文字入力→ドット変換の１部

```
Dim col As Long        '色情報をいれる
Dim xydata(999) As Long
Dim xtemp As Long     '圧縮した後の高さ
Dim ytemp As Long     '圧縮した後の幅
Dim t As Integer
t = 0
      '元の画像の情報を圧縮する
xtemp = Picture2.Width / 15
ytemp = Picture2.Height / 15
   For y = 0 To ytemp
     For x = 0 To xtemp
 '２の画像から各ポイントでの色情報を取得
   col = Picture1.Point(x * 15 - 7, y * 15 - 7)
       If col = 0 Then
  Text2.Text = Text2.Text + "(" + str(x)  + ","
                Text2.Text = Text2.Text + str(y)  +
")"
   For i = 0 To 14
   '３のピクチャーボックスに文字を描く
  Picture2.PSet (x * 15 - 7 + i, y * 15 - 7), co
```

４．３．２　ポジショナコントロール



図．３　ポジショナコントロールＦｏｒｍ

（１）　Ｘ・Ｙ軸ポジショナコントロールのコントロールについては、左側部分に配置
Ｚ軸ポジショナコントロールのコントロールについては、右側半分に配置

（２）　Ｘ・Ｙ・Ｚ軸移動コマンドについて

【１軸＋】・【１軸－】のテキストボックスに移動パルス数により、Ｘ軸の原点位置を調整
５００パルスで１［ｍｍ］
【２軸＋】・【２軸－】のテキストボックスに移動パルス数により、Ｙ軸の原点位置を調整
５００パルスで１［ｍｍ］
　１軸の速度・２軸の速度のテキストボックスの【▼】をクリックすることにより、Ｘ・Ｙ軸の速度を４段階に変化
　Ｚ軸ポジショナコントロールのコントロール
【ＵＳＢ＋】コマンドにより、Ｚ軸をＵＰ
【ＵＳＢ－】ＤＯＷＮ
【例】　Ｆｏｒｍ１　文字入力画面よりのＸ・Ｙ座標点よりデータを受け取り、Ｘ・Ｙ軸ポジショナコントロールの駆動
Ｚ軸ポジショナ駆動
【停止】　Ｘ・Ｙ２軸を同時に停止
【原点復帰】　Ｘ・Ｙ軸を原点に復帰
【ＵＳＢ＿ＳＴＯＰ】　Ｚ軸の停止
【【開始】　テキスト・ボックスに入れたコマンドにより、Ｘ・Ｙ軸の駆動

（３）　移動速度の設定

　移動速度の値を設定しますのテキストボックスに速度を１速～４速まで
最小速度・最大速度・ 加減速時間の値を入力
この値は以後画面情報ファイル
（ｓａｍｐｌｅ　Ｉｎｆｏ）として保存

３．２．２　Ｆｏｒｍ　Ｌｏａｄプログラムの１部

```
'-----------------------------------------------------------------
"■　Form_Load
'　（機能）
'　　Comm1 ポートをオープンする
"-----------------------------------------------------------------
Private Sub Form_Load()

    If MSComm1.PortOpen = False Then        '
 Comm1Port が使用されていなければ,
        MSComm1.CommPort = 1
 ' Port 1 を指定
        MSComm1.Settings = "9600,N,8,1"
' 9600pbs, パリティ無 データビット 8Bit ストップビット 1Bit.
 SComm1.InputLen = 0
  ' 受信バッファのクリア
   MSComm1.PortOpen = True
  ' ポートを Open する
       Else                                '
Comm1Port が使用されていれば,
警告メッセージを出す。
        Print MSComm1.CommPort
        MsgBox ("ComPort1 が使えません。他のプログラムを閉じてから再度立ち上げてください。")
    End If
    '-------------------------------------------------
'USB 接続を行う：パイプを開く
'-------------------------------------------------
 Dim l As Long
     hUSB = Pscon_Open_mask(UU_MASK_VENDOR + UU_MASK_PRODUCT, 0, 0, Vendor, Product, 0)

  ' ベンダー ID, プロダクト ID の確認
   If hUSB = -1 Then
  ' エラーハンドラだと
      MsgBox "USB が見つかりません", 16, "制御エラー"
 ' エラーメッセージを出す
        GoTo JP1
    End If

    wPIPE = Pscon_OpenPipe(hUSB, 0, 1)
      ' USB の送信ハンドルを開く
    If wPIPE = -1 Then
 ' ｴﾗｰハンドラだと
        MsgBox "USB 送信のパイプを開けませんでした", 16, "制御エラー"
  ' エラーメッセージを出す
        GoTo JP1
    End If

    hPIPE = Pscon_OpenPipe(hUSB, 0, 0)
                  ' USB の受信ハンドルを開く
    If hPIPE = -1 Then
 ' エラーハンドラだと
        MsgBox "USB 受信のパイプを開けませんでした", 16, "制御エラー"
 ' エラーメッセージを出す

        GoTo JP1
    End If

    l = Pscon_ResetDevice(hUSB)
 ' デバイスにリセットコマンドを送出する。
JP1:                                    ［３］
```

５．終わりに

　ＶＢはインタプリタであるため､実行速度ははっきりいって遅いところもあります。
またコンパイルして実行ファイルを作ってもあまり速くはないようでした。
従来のＮ88ＢＡＳＩＣの数倍程度、Ｃの数分程度ではと思います。
　ＶｉｓｕａｌＢａｓｉｃは本当によくできていると思います｡ＶＢの中がブラックボックスであっても､簡単に制御プログラムを作ることができます。
　今回は【ＲＳ－２３２Ｃサブルーチン】［move_ｓｔａｒｔ］と
【ＵＳＢサブルーチン】[move_ｓｔａｒｔｓｕｂ］はどちらか一方の処理が終わってから､次の処理を行うようにしないといけないところもありますが、一応はある成果が出たと思います。
　今後の課題として､各ドット点で上下さして､彫刻するのではなく､ＣＡＤデータをＤＸＦファイルに変換して､文字だけでなくどのようなダータでも彫刻できるようなシステムにしたいと思います。

今システムを元に学生の制御実習に活用したいと思います。

最後に本システムの開発にあたり、ハード面、ソフト面、両面において、株式会社［ユニタック］の皆様にはお世話になりました。

【参考文献】

［1］　ＣＱ出版社
ＶＢと製作で学ぶ初めての
パソコン応用工作

［2］　技術評論社
ＶｉｓｕａｌＢａｓｉｃ　６.０
（中級編）

【参考プログラム】

［3］
Ｘ．Ｙ　ポジショナ　ＳＤ５０－３０
サンプルプログラム