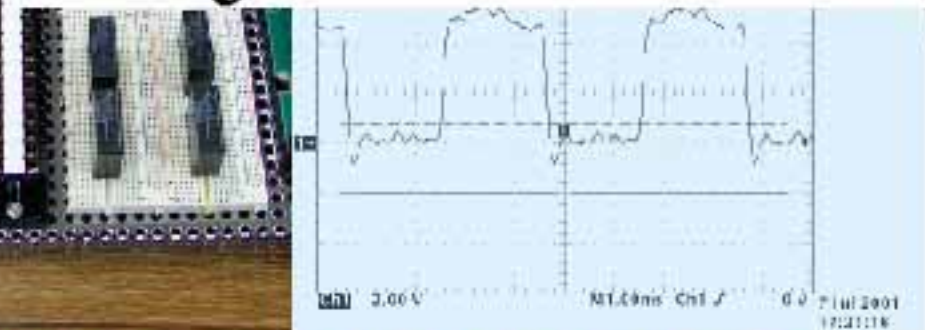
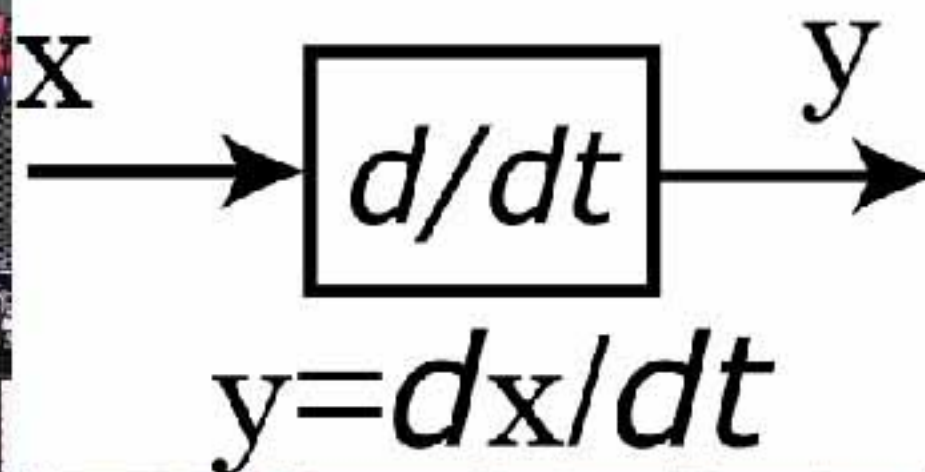
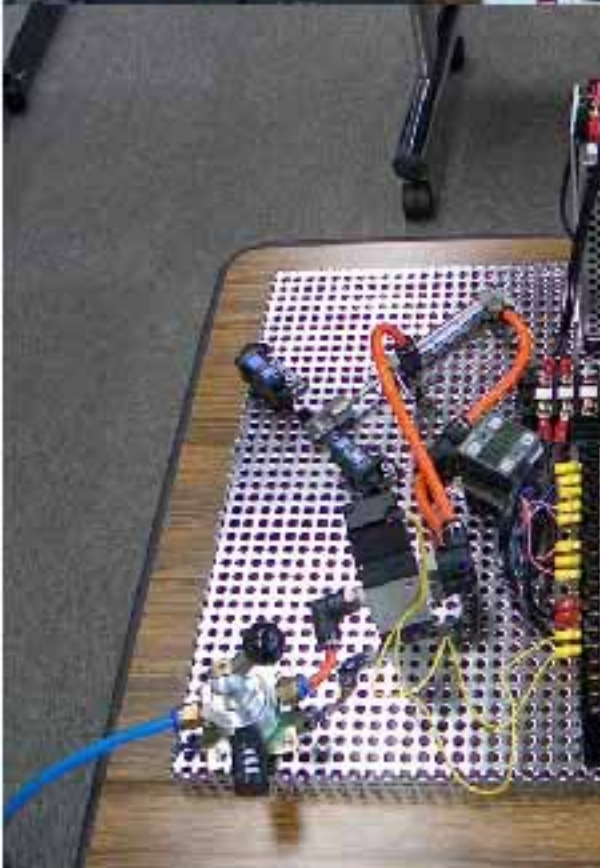
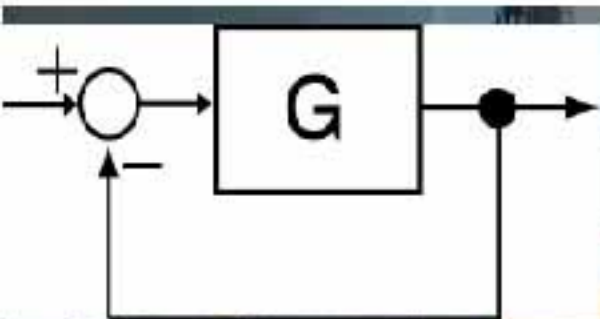


Visual Basicによる計測・制御



関東職業能力開発促進センター

TEL : 045-391-9537

FAX : 045-391-0141

Visual Basic による計測・制御

1.	はじめに	1 p
2.	Visual Basic による計測・制御	1 p
3.	Visual Basic について	1 p
4.	Visual Basic によるプログラミング	4 p
5.	GPIB からのデータ集録	8 p
6.	RS232C からのデータ集録	15 p
7.	A/D ボードからのデータ集録	25 p

付録

1.GPIB ボード資料

2.A/D ボード資料

1. はじめに

パソコンは、電子計算機と呼ばれ、計算処理装置としての使い方が一般的であった。しかし、インターネットやワードプロセッサの浸透に伴い、生産現場や実験室からの報告書作成にパソコンが使われるような状況にあり、データをそのままワードプロセッサ等で利用する環境が求められている。特に自動計測の分野では、その重要性が高まっている。本実習では、パソコンを利用した自動計測を中心とした機器制御の基礎技術の修得を目指す。

現在、パソコンの OS で最も多く使用される Windows 上で開発に使用されるプログラム言語のひとつである Visual Basic を使用した機器制御をおこなう。また、機器とパソコンを接続するインターフェースとして、GPIB、RS232C、A/D ボードを介した制御プログラムを作成する。

2. Visual Basic による計測・制御

Windows 上で稼動するプログラム言語を考えた場合、様々な言語のなかで価格対効果が良く、メーカーによるサポートも多く提供されているため、使用者が多い言語が Visual Basic である。Visual Basic を使った機器制御をおこなう場合、様々なハードウェアを直接制御する必要がある。このとき、ハードウェアを製造しているメーカーが作成しているプログラムを組み込んで使用することが多い。このため、メーカーが組み込むプログラムに合わせてプログラムを作成しなければならない点に注意を払う必要がある。

また、プログラムそのものを作成するためには、Visual Basic の使い方ではなく、ハードウェアのマニュアルから機器のプログラムを使って操作する方法を修得する必要がある。マニュアルが無い場合は、ハードウェアのメーカーからマニュアルを取り寄せる必要がある。

3. Visual Basic について

Windows のような GUI(グラフィカル・ユーザー・インターフェース)を持つ OS 上でプログラムを作成する場合は、操作をおこなうインターフェースを作成する部分と演算処理をおこなう部分に分かれる。

Visual Basic では全体を統括するプロジェクト内に、

- (1) Form 上に、制御器や表示器を配置して、ユーザーインターフェース部を作成する。
- (2) 制御器および表示器に対応したイベントプロシージャ上で演算処理内容を記述する。

Form や各種イベントのデザインをプログラムと考えることができる。

Visual Basic の起動

スタート＞プログラム＞Microsoft Visual Basic 6.0＞Microsoft Visual Basic6.0 を選択して Visual Basic を呼び出す。



デスクトップ上にアイコンを作成するとデスクトップ上のアイコンをダブルクリックすることで、Visual Basic を起動することができる。(図 01)

図 01. Visual Basic
のアイコン

起動すると、新規作成する際のメニュー画面が表示される。(図 02)

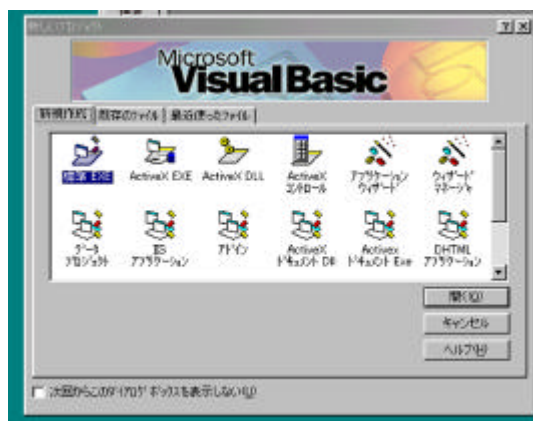


図 02. メニュー画面

メニュー画面から、「標準 EXE」を選択して「開く」ボタンをクリックする。するとプロジェクトのデザインをおこなう画面が立ちあがる。(図 03)

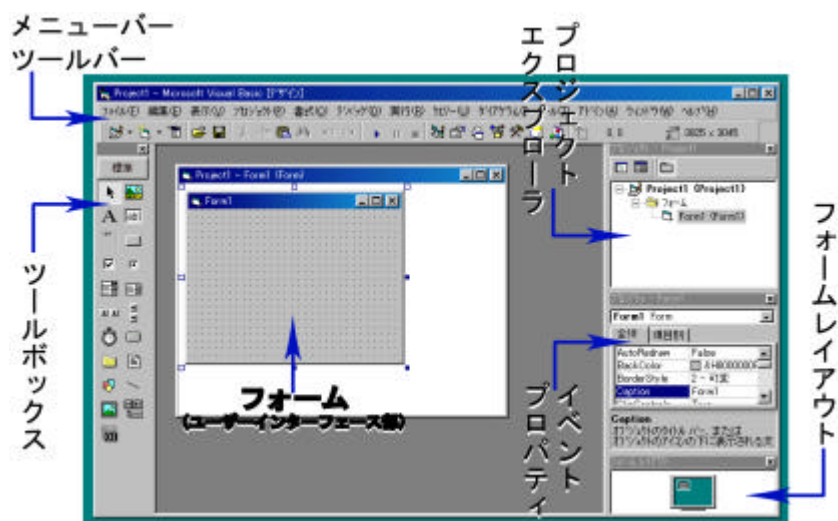


図 03. プロジェクト作成画面

メニューバーとツールバー：

ファイルの保存、プログラムの実行等の実施。

ツールボックス：

プロジェクトに組み込む各オブジェクトを、Form 上にデザインするためのツール。

プロジェクトエクスプローラ

プロジェクトおよびプロジェクト内で使用される各項目を階層構造で表示する。

イベントプロパティ：

作成したオブジェクトの設定をおこなう。

フォームレイアウト：

作成したフォームを実行する際に、ウィンドウ上で表示する位置設定をおこなう。

フォーム（Form：ユーザーインターフェース）：

プロジェクト内の各オブジェクトをユーザーが操作するための画面を作成する。

Form 上にマウスのカーソルを移動させ、右クリックすると、Form のメニュー画面が表示される。メニュー画面から、「コードの表示」を選択する。

(図 04)

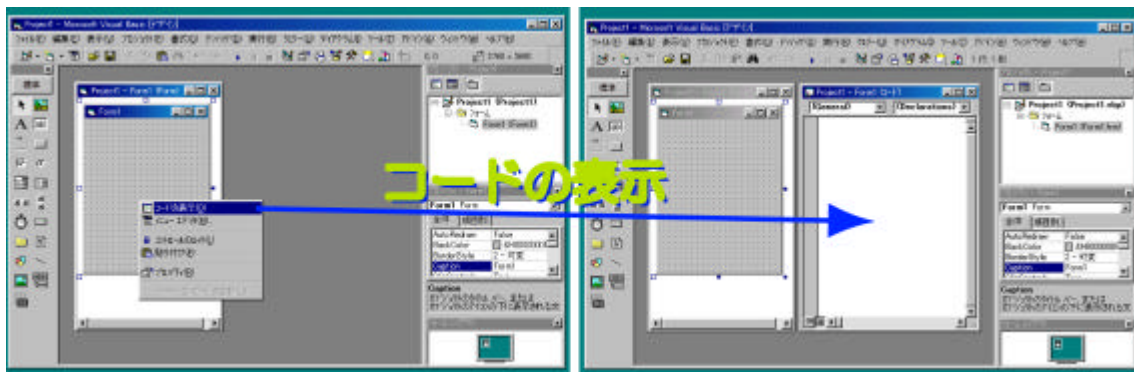


図 04. コードの表示

フォーム上に配置される、各オブジェクトによって処理が生じる状態を、「イベントの発生」と呼んでおり、イベントの制御方法をコードウィンドウ上に記述する。このコードウィンドウ上に記述する行為が従来のプログラミングの作成行為にあたる。イベントを発生させる状況を設定すると、自動的にコードが作成され、コードウィンドウ上に記述される。コードウィンドウ上の記述とフォーム上のオブジェクトとの関係を設定するのが、イベントプロパティとなる。

※イベントプロパティでの変更は、フォームウィンドウ上には変更結果が適応されるが、コードウィンドウ上には変更結果が適応されない。

4. Visual Basic のプログラム作成

Form 上にユーザーの操作およびユーザーへ表示する画面のデザインをおこなう。また、ユーザーの操作によって発生する演算処理および表示状況の整理をおこなう。

課題 1.

ボタンを押すとメッセージを表示するプログラムを作成する。

画面上でボタン表示される制御器を、ツールボックスから選択して Form 上に配置する。(図 05)

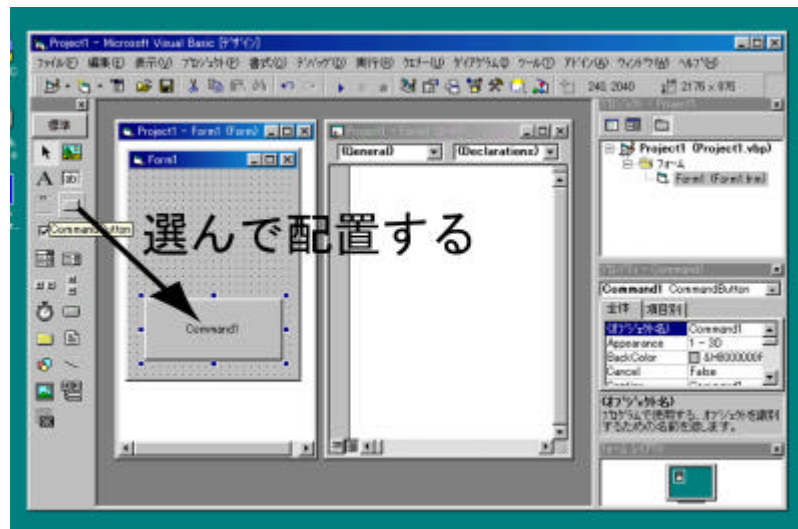


図 05. ボタンの配置

画面上に文字を表示するための表示器を、ツールボックスから選択して Form 上に配置する。(図 06)

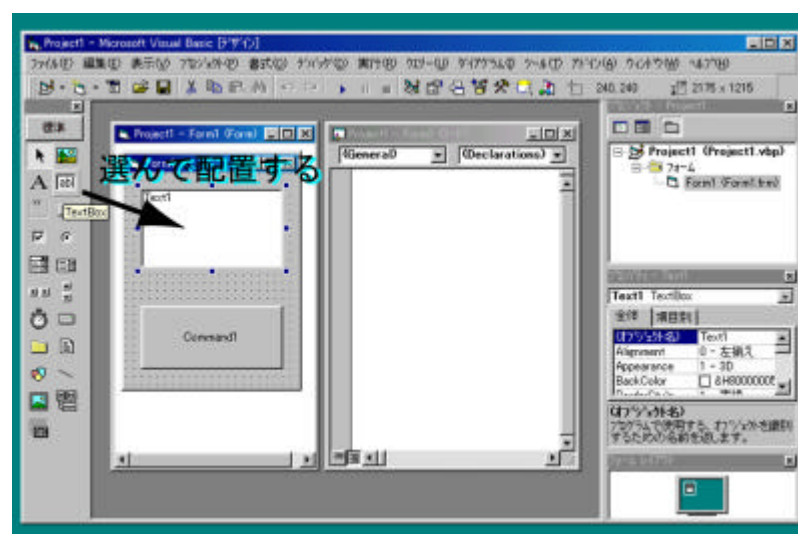


図 06. 画面上に文字を表示するエリアの配置

フォームおよびフォーム上に制御器として配置したボタンにマウスのカーソルを合わせて左クリックをすると、ボタンのプロパティが表示される。(図 07)

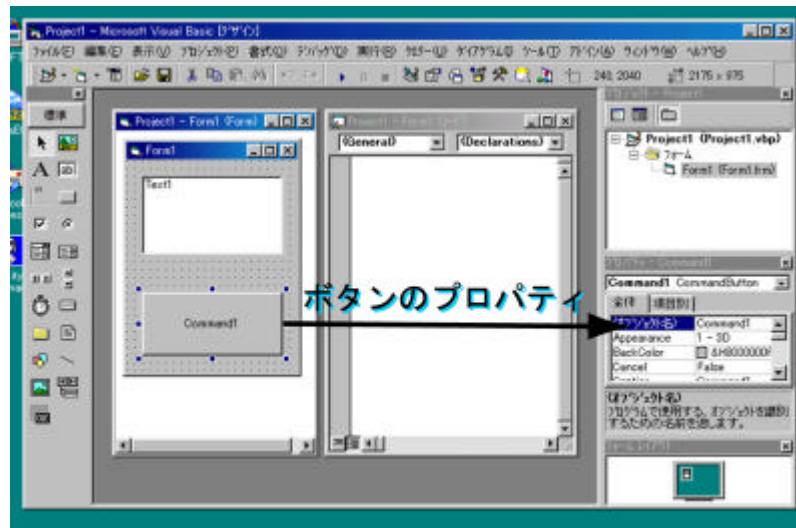


図 07. ボタンのプロパティ

プロパティのオブジェクト名を変更すると、ボタン上でマウスを左クリックした時に生じるイベントを処理するプログラムの変数値が変更される。プロパティの Caption を変更すると、ボタン上に描かれている名称が変更される。

プロパティ上では他に色等の変更が可能である。

プロパティの変更がすべて完了したら、ボタン上にカーソルを合わせてダブルクリックすると、ボタンによるイベントが生じたときに演算処理をおこなうプログラムを記述するコードが表示される。(図 08)

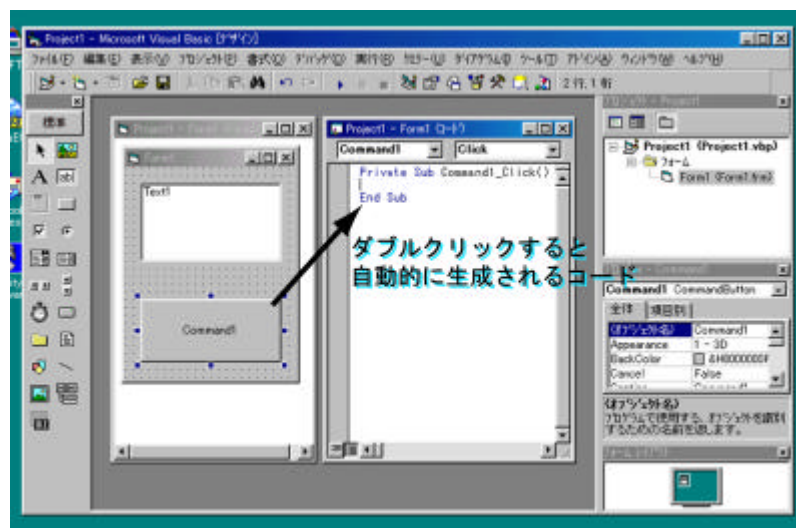


図 08. コードの自動生成

自動生成されたコード（Private Sub～End Sub）に演算処理内容の記述をおこなう。（図 09）

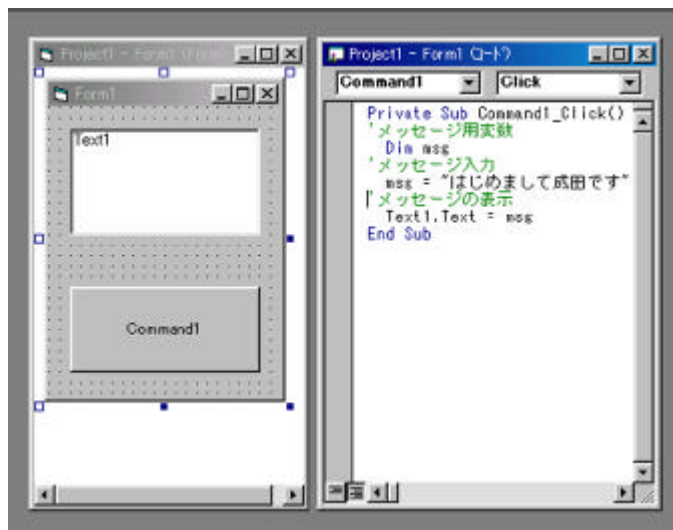


図 09. コードへ演算処理内容の記述

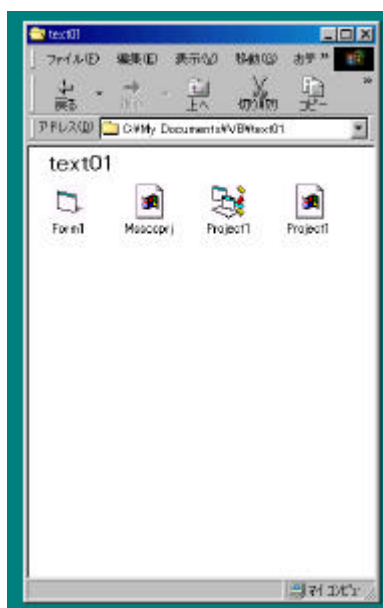


図 10. プロジェクトの保存

記述が完了すれば、例題 1 における演算処理内容の完成となる。完成したプロジェクトおよび Form を、ファイル＞プロジェクトの保存とファイル＞Form の保存から保存する。プロジェクトおよび Form を保存する場合、同名のファイルが存在することを避けるため、新しくディレクトリを作成して保存する方法が有効である。（図 10）

保存の完了後、プロジェクトを実行する。実行後、ボタン上でマウスをクリックするとメッセージが表示される。(図 11)

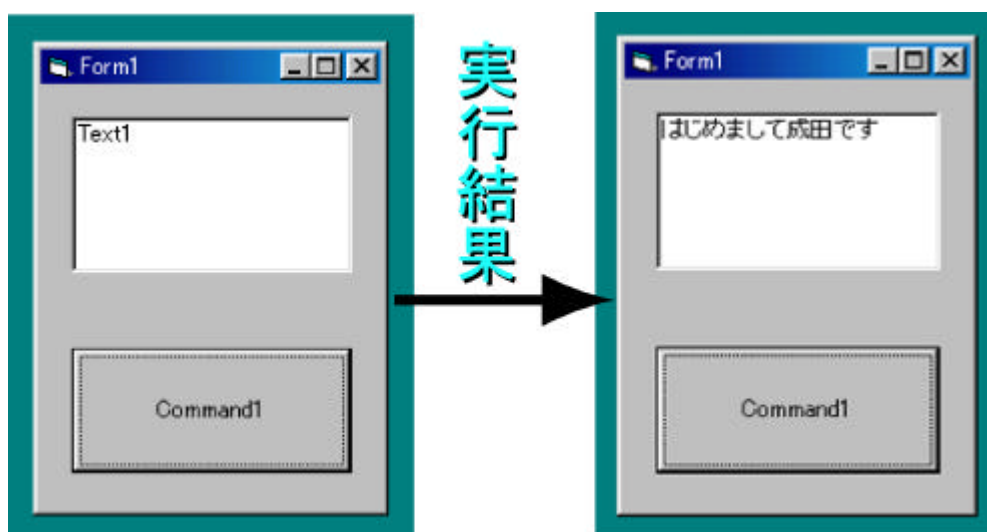


図 11. 完成したプロジェクトの実行結果

5. GPIB からのデータ集録

GPIB インターフェースは、General Purpose Interface Bus として、ヒューレットパッカード社から提唱されたインターフェースの規格です。アメリカ電気電子技術者協会（IEEE）により、IEEE488-1975 として規格化されていることから、IEEE488 インターフェースとも呼ばれる。1987 年に計測制御の命令や接続可能な機器数の拡張を含めた、IEEE488.2-1987 が制定されている。従来の規格は IEEE488.1-1987 の形で規格化されている。現在は、1992 年に改訂された IEEE488.2-1992 が採用されていると考えられる。しかし、IEEE488 および IEEE488.1 についても無くなったわけではないので、IEEE488 や IEEE488.1 を採用した計測装置も残っている。

GPIB インターフェースを Windows 上で稼働させるためには、インターフェースを製造しているメーカーのドライバソフトが必要である。Windows のバージョンによっては、ドライバソフトと OS が合わずハードウェアが稼働しない場合がある。

本実習では、ナショナルインスツルメンツ社製のノート用 GPIB インターフェースカードを使用する。(図 13)

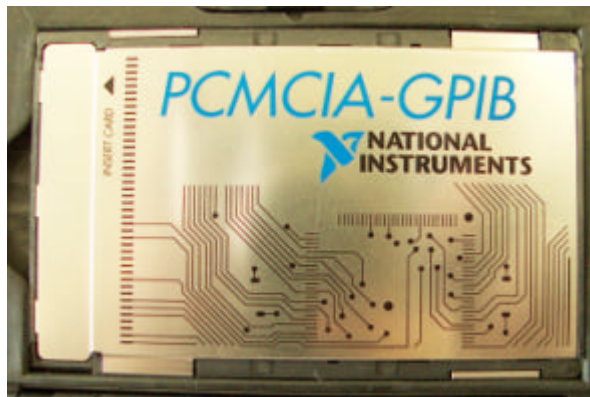


図 13. ノート用 GPIB インターフェース
ADVANTEST 社製 R6551 デジタルマルチメータを用いる。(図 14)



図 14. ADVANTEST 社製デジタルマルチメータ

まず、 GPIB インターフェースをケーブルで接続する。(図 15)

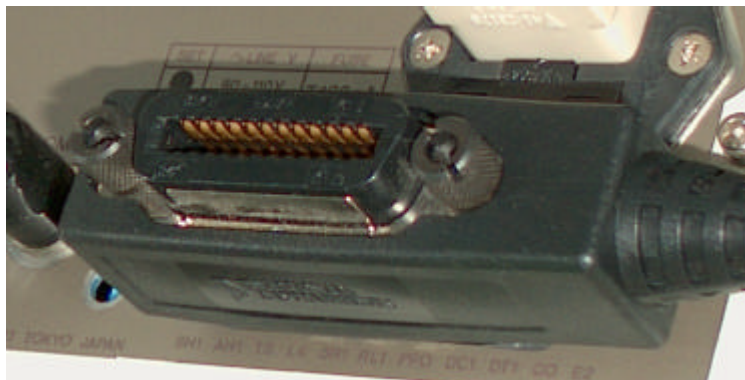


図 15. 機器との接続

また、 GPIB インターフェースは複数の機器を接続できるため、各機器に割り当てる番号を機器毎に設定する必要がある。(図 16)



図 16. 機器上での GPIB アドレス設定

各種計測器を使った自動計測をおこなうには、まず計測器を稼働させるための命令を理解する必要がある。IEEE488.1 では、命令については記述されておらず、IEEE488.2 においても一部の命令が共通化されているだけで、使用する命令は各メーカーや機器の種類等によって異なる。

ADVANTEST 社 R 6 5 5 1 デジタルマルチメータの場合、

F 1 : 直流電圧計測

R E 5 : 表示桁数 5 1/2 桁

H 1 : ヘッダ ON, ASCII コード

D L 0 : デリミタ (区切り記号) CR+LF と EO I 出力

命令間はカンマ (,) で接続し、機器へ送信をおこなうと、計測結果が ASCII コードのデータとして帰ってくる。使用する機種がどのような命令を持ち、命令を送出した結果どのようなデータが帰ってくるのかについて、マニュアルで確認する必要がある。(特に多機能な計測機器は、命令数が多く確認内容が多いので注意が必要である)

GPIB ドライバのインストール

GPIB インターフェースを購入すると、メーカーのドライバソフトが添付されていることが多い。添付されていない場合、インターネット上からダウンロードする等の方法で、ドライバソフトを手に入れる必要がある。Windows の場合、OS のバージョンが変わった場合、新バージョン対応のドライバソフトを別途手に入れる必要があることが多い。通常、新 OS 対応のドライバソフトは、新 OS 発売後一定の期間が過ぎて完成することが多いので、計測制御に使用するパソコンの新 OS への切り替えは、ドライバの開発終了後におこなう必要がある。

パソコンのカードスロットに、図 13 のカードを入れると、プラグアンドプレイが稼動して、GPIB インターフェースカードが認識されるので、画面の指示に従って、ナショナルインスツルメンツ社のドライバソフトをインストールする。

インストール完了後、マイコンピュータにカーソルを合わせてマウスを左クリックしてメニュー画面を表示させる。(図 17)

メニュー画面からプロパティを選択し、システムのプロパティを表示、デバイスマネージャを選択する。デバイスマネージャで、PCMCIA-GPIB の稼動状態を確認する。(図 18)

図 17. マイコンピュータのメニュー画面

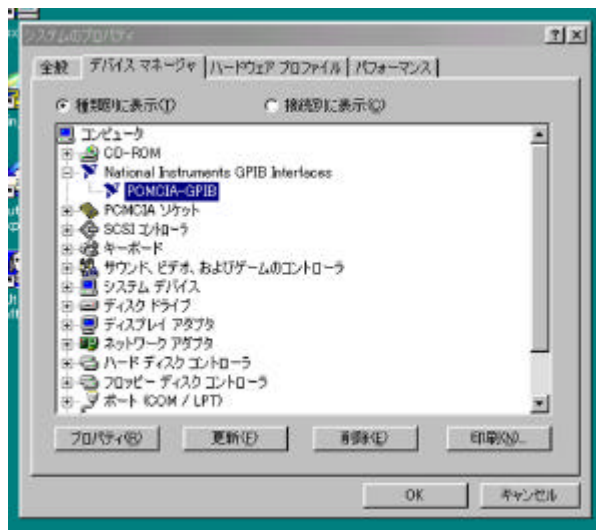
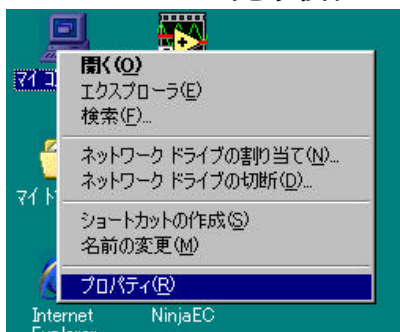


図 18. デバイスマネージャによる確認

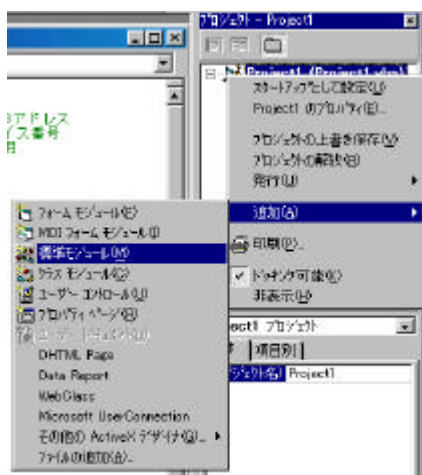
正常に稼動していない場合、デバイスマネージャ上にエラーマークの表示によって確認できる。

GPIB 用プロジェクトの作成

課題 2

GPIB インターフェースを介したデジタルマルチメータの制御プログラムを Visual Basic で作成する。

まず、Visual Basic 用の GPIB ドライバソフトを追加モジュールとして組み込む。モジュールプログラムは、GPIB インターフェースボードを製造しているメーカーから提供される。（インターネット上からダウンロードが必要な場合もある）メーカーによっては、計測器用のドライバソフトを作成し配布している場合もある。



追加モジュールは、プロジェクトエクスプローラ上で右クリックし、メニュー画面を呼び出し、追加＞「標準モジュール」を選択する。

(図 19)

「標準モジュールの追加」のメニュー画面から、既存のファイルから、メーカーから提供されている GPIB ボード用のモジュールソフトを組み込む。(図 20)

図 19. 標準モジュールの追加

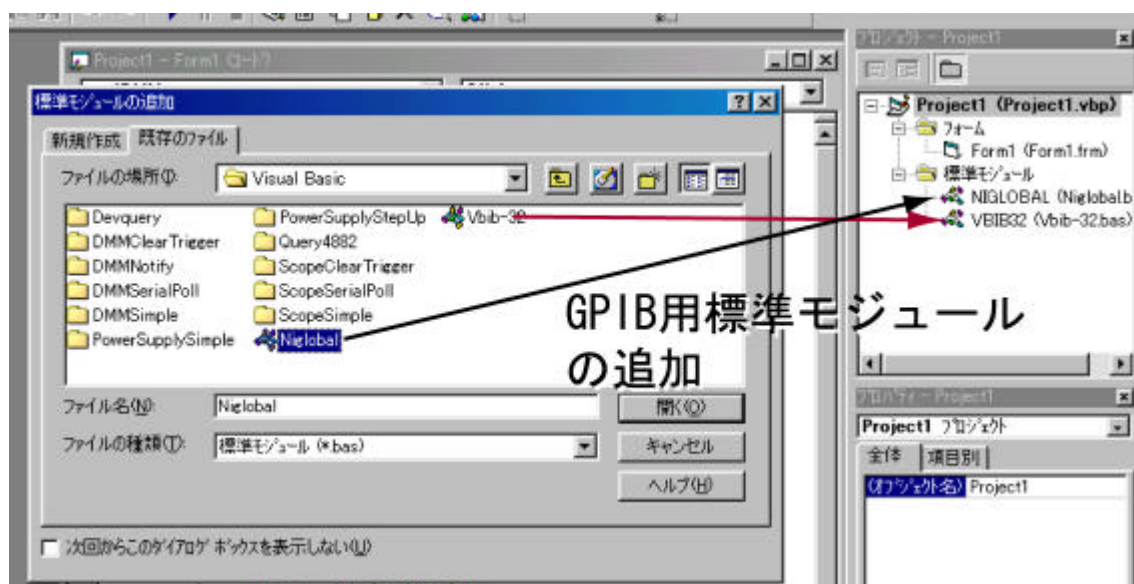


図 20. GPIB 用標準モジュールの追加

自動計測プログラム作成

画面上のボタンを押すとデジタルマルチメータで、電圧計測をおこなうプログラムの作成する。

ボタンを Form 上に作成し、オブジェクト名をイベントプロパティで、cmdDMM に変更後、右クリックをおこないコードを自動生成する。

```
Private Sub cmdDMM_Click()
```

変数宣言

```
Dim addr As Integer 'マルチメータの GPIB アドレス
```

```
Dim dmm As Integer 'マルチメータのデバイス番号
```

```
Dim rd As String '読み込みデータ保存用
```

```
Dim Spol As Integer 'ポーリング用
```

```
addr=2 ' デジタルマルチメータのアドレスを 2 に設定
```

```
Call ibdev(0, addr, 0, T10s, 1, 0, dmm) ' GPIB 機器の初期化コマンドの送信
```

```
Call ibclr(dmm) ' GPIB デバイスの初期化
```

```
' デジタルマルチメータへの制御コマンドの送出
```

```
Call ibwrt(dmm, "F1,RE5,H1")
```

```
' 測定終了の検出
```

```
Do
```

```
Call ibrsp(dmm, Spol)
```

```
DoEvents
```

```
Loop While(Spol And 16)=16
```

```
' デジタルマルチメータからのデータ読み込み
```

```
rd = Space$(20)
```

```
Call ibrd(dmm,rd)
```

```
' 取得データからの測定値の抜き出し
```

```
Dim MyString, Head, DMMDData
```

```
MyString = rd ' 文字列読み込み
```

```
Head = Mid ( MyString, 1, 3 ) ' ヘッダを Head に代入
```

```
DMMDData = Mid ( MyString, 4, 11 ) ' データを DMMDData に代入
```

```
' 取得データのテキストへの出力
```

```
DMM_Data.Text=DMMDData
```

```
' GPIB 機器をローカルに設定 (通信終了)
```

```
Call ibloc(dmm)
```

```
End Sub
```

完成したプロジェクトを以下に示す。(図 21)

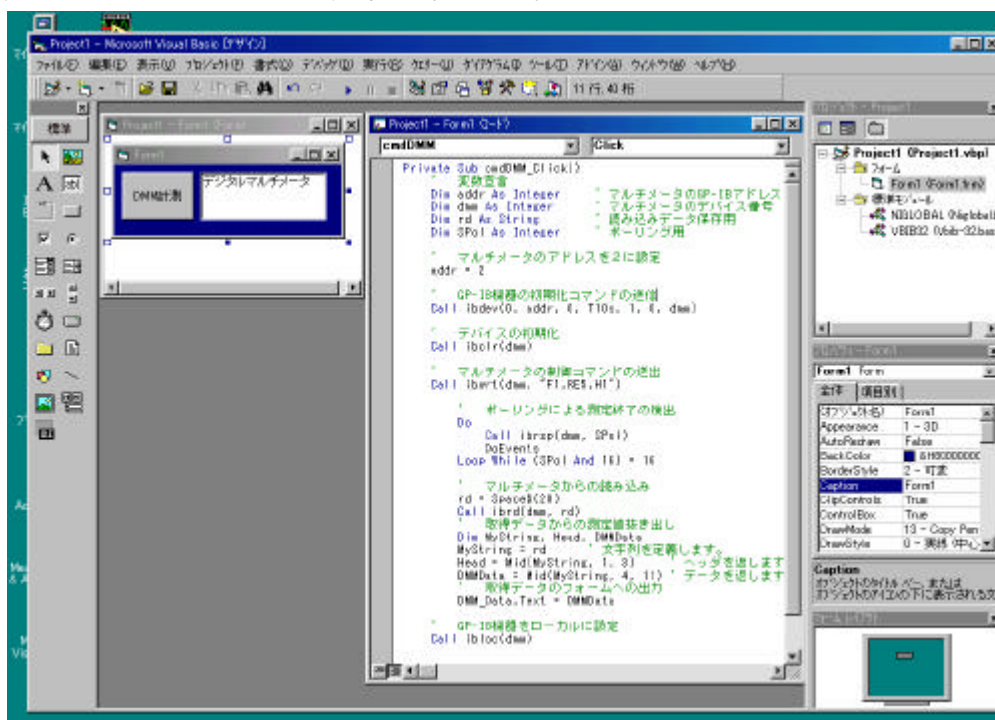
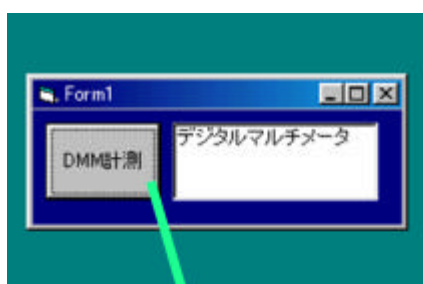


図 21. GPIB を介したデータ集録プロジェクト



実行すると、デジタルマルチメータで電圧計測した結果がパソコン上に表示される。(図 22)

ボタンを押すと
計測結果が表示される

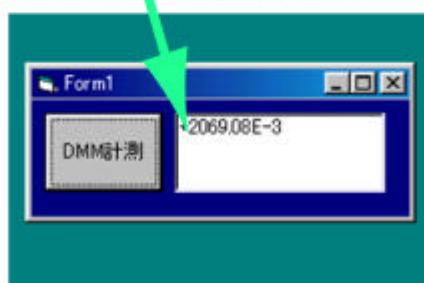


図 22. 実行結果

GPIB で使用する関数について（NI 社製ボードの場合）

Call ibdev (①, ②, ③, ④, ⑤, ⑥, ⑦)

または、

⑦=ibdev (①, ②, ③, ④, ⑤, ⑥)

GPIB ボードの設定を初期化する関数

- ① コントローラの GPIB アドレス
- ② 通信対象となる機器の GPIB アドレス
- ③ 通信対象となる機器の GPIB 二次アドレス
- ④ タイムアウト時間
- ⑤ EOI モード
- ⑥ アドレス
- ⑦ プログラムデバイス信号取得

Call ibclr (⑦)

⑦ で記述された ibdev に従って、ボードの初期化を実行する関数

Sta=ibclr(⑦)

ER は、ibsta データを出力する。

Ibsta は、GPIB コントロールの状態を示す。

Call ibwrt (⑦, ⑨)

⑦ で指定されている機器へ、⑨のデータを送信する。

Call ibrsp(⑦, ⑩)

⑦ 指定されている機器のシリアルポール反応を⑩へ読み出す。

シリアルポールによるステータスは、更新してデバイスの状態がクリアになるのを待つ必要があります。

Call ibrd (⑦, ⑪)

⑦ 指定されている機器からのデータを受信して⑪へ出力する。

⑪データを文字列で取得する。

ibloc (⑦)

⑦ 指定されている機器をローカルに戻す。

6. RS232C からのデータ集録

シリアルインターフェースとしては、最も一般的に使用され、パソコンに標準で装備されているインターフェースが RS232C である。規格の流れとしては、Recommended Standard から Electronics Industries Association および Telecommunications Industries Association へと変化しており、現在パソコンに搭載されているインターフェースは、EIA/TIA 規格に準じて搭載されていることが多い。

ただ、パソコンに搭載されているインターフェースは、通常モデムと接続することを前提として設計されており、接続する機器が古い場合、従来の RS 規格で設計されていることがあり、パソコン側のインターフェースポートと電圧の違いから焼損する可能性がある。(変換ケーブル等を使用すると発生しやすくなるので、実際に使用するシリアルポートの電気的特性を確認する必要がある)

また、信号線の接続は、Dsub9pin を使ってストレートもしくはクロス接続する形をとることとなっているが、機器によっては接続形式がことなることがある。(表 1) (制御線が使用されている場合と使用されていない場合があり、使用されていない場合には配線そのものを変更する必要がある)

表 1. シリアルポートのピン配置				
9Pin	25Pin	略称	名称	
1	8	DCD	Data Carrier Detect	キャリア検出
2	3	RxD	Receive Data	受信データ
3	2	TxD	Transmit Data	送信データ
4	20	DTR	Data Terminar Ready	データ端末レディ
5	7	SG	Signal Ground	信号グラウンド
6	6	DSR	Data Set Ready	データセットレディ
7	4	RTS	Request To Send	送信要求
8	5	CTS	Clear To Send	送信可能
9	22	RI	Ring Indicator	被呼表示

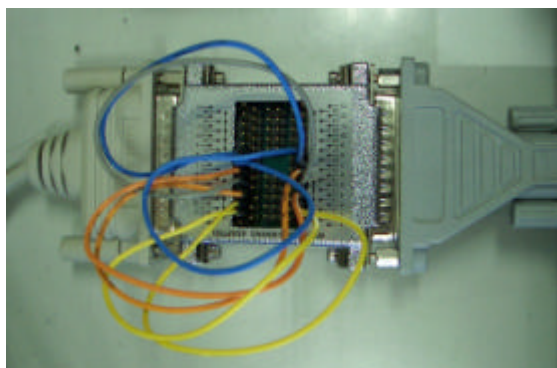


図 23. MINI-Wiring Adapter

接続配線を汎用でおこなうために、本実習では、MINI-Wiring Adapter 等を使用する。
(図 23)

機器間を接続するケーブルは、専用になっていることが多い。(接続するピン番号やコネクタの形状が異なることもある)

課題 3.

ソニーテクトロニクス社製のデジタルオシロスコープ TDS210 を使って、RS232C を介した計測プログラムを作成する。(図 24)

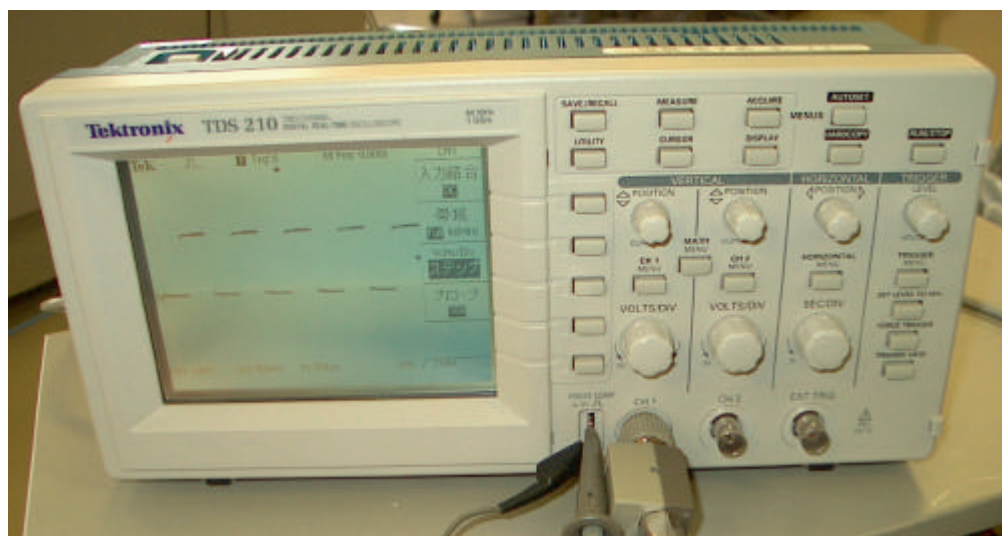


図 24. ソニーテクトロニクス社製 TDS210

ソニーテクトロニクス社の機器を接続する場合には表 2 に従って、RS232C ケーブルの配線をおこなう必要がある。(表 2)

表 2. RS232C 配線接続図

RS-232Cによる接続 D-Sub 9ピン)

コンピュータ側		オシロスコープ側	
ピン番号	信号名	ピン番号	信号名
1	DCD	1	DCD
2	RxD	2	RxD
3	TxD	3	TxD
4	DTR	4	DTR
5	GND	5	GND
6	DSR	6	DSR
7	RTS	7	RTS
8	CTS	8	CTS
9	RI	9	RI

パソコンのシリアルインターフェースと計測器を接続し、Visual Basic で制御およびデータを集録するプログラムを作成する場合、機器毎に通信方法が異なることが多いため、通信手順等について、機器のマニュアルから確認する必要がある。

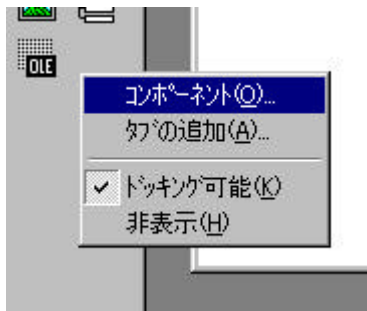


図 25. ツールボックスのメニュー

まず、Visual Basic でシリアルポート通信 Microsoft Comm Control 6.0 をツールボックスに組み込む必要がある。標準ツールボックス上でマウスの右クリックしてメニュー画面を立ち上げ、コンポーネントを選択する。(図 25)

コントロールの中から Microsoft Comm Control 6.0 を選択し、「OK」のボタンを押す。(図 26)

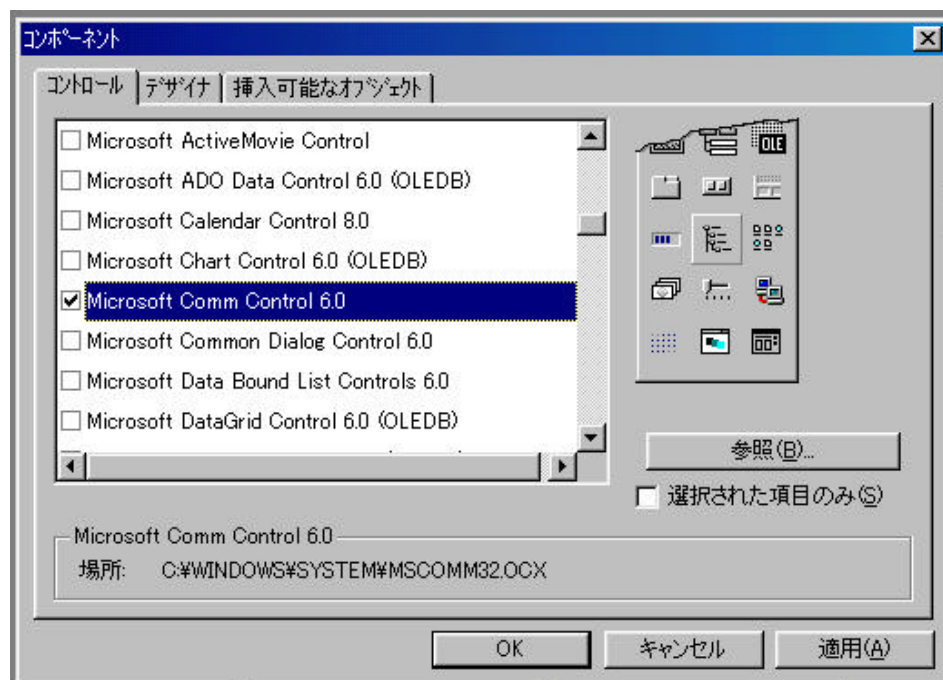


図 26. コントロールの追加



図 27. 追加されたアイコン

「OK」のボタンを押すと、ツールボックスに電話をイメージしたアイコンが現れる。(図 27)

追加されたツールを使うには、ツールボックスから MSComm を選択し、Form 上に配置する。(図 28)

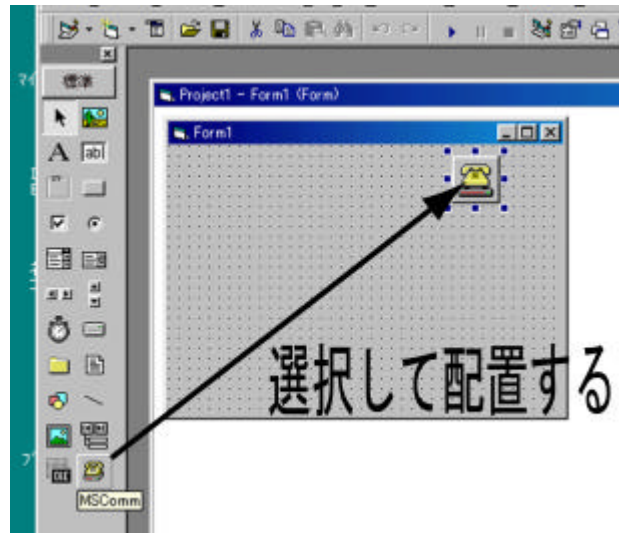


図 28. MSComm の配置

RS232C ポートを使用して通信をおこなう場合、送信側および受信側の通信条件を合わせる必要がある。この時、計測器側の通信条件を変更する方法がないことが多いので、パソコン側で通信条件を計測器に合わせて設定する必要がある。(表 3)

表 3. TDS210 の通信条件

通信条件		
通信速度	9600	bps
データ長	8	bit
パリティ	none	
ストップビット	1	bit

Form イベント発生時に、MSComm コントロールの初期設定コードを作成する。

```
Private Sub Form_Load()
    'MSComm コントロールの初期設定
    mscPort.CommPort = 1
    mscPort.Settings = "9600,n,8,1"
    mscPort.Handshaking = comRTS
    mscPort.RTSEnable = True
    mscPort.SThreshold = 1
    mscPort.RThreshold = 1
End Sub
```

シリアル通信ポートは 2 重に使用することができないため、なんらかのトラブルでプログラムが停止した場合、再度通信ポートを使用しようとした場合、エラーが発生する。こういったトラブルを防ぐため、通信ボタンを使って、ポートの状態を確認およびポートの接続をおこなうプログラムを作成する。

```
Private Sub Form_Unload(Cancel As Integer)
```

```
    通信ポートの閉鎖
```

```
    If mscPort.PortOpen Then
```

```
        mscPort.PortOpen = False
```

```
    End If
```

```
    終了
```

```
End
```

```
End Sub
```

```
Private Sub rsport_Click()
```

```
    mscPort.PortOpen = Not mscPort.PortOpen
```

```
    If mscPort.PortOpen = True Then
```

```
        rsport.Caption = "通信停止"
```

```
    Else
```

```
        rsport.Caption = "通信開始"
```

```
    End If
```

```
End Sub
```

通信ポートの確認終了後、機器に対して命令の送信をおこなうプログラムを作成する。このとき、命令の後に制御コード（LF）を付けてデータを送信する必要がある。

```
Private Sub cmddata_Click()
```

```
    通信ポートの確認
```

```
    If mscPort.PortOpen = True Then
```

```
        mscPort.Output = cmdSend.Text & Chr(&HA)
```

```
    End If
```

```
End Sub
```

命令が送出されると、機器側から計測データが送出される。このとき、シリアルポートからのデータ受信による割り込みによるデータ受信プログラムを作成する。また、受信データより、測定データ部分を抜き出すプログラムを作成する。

```
Private Static Sub mscPort_OnComm()  
    Dim rsBuffer As Variant  
    Dim rsData, PointData  
    Select Case mscPort.CommEvent  
        Case comEvCD  
        Case comEvCTS  
        Case comEvDSR  
        Case comEvRing  
        Case comEvReceive  
            rsBuffer = mscPort.Input  
            rsData = rsData & rsBuffer  
        Case comEvSend  
        Case comEvEOF  
        Case Else  
            MsgBox "通信エラーが発生しました", vbExclamation, "通信エラー"  
    End Select  
    PointData = Mid(rsData, InStr(rsData, "CURVE ") + 6)  
    txtData.Text = PointData  
End Sub
```

データを配列関数として扱えるように変換する。

```
Private Sub trData_Click()  
    Dim im As Long  
    Dim i1, i2, i3 As Integer  
    Dim strData  
    Dim P_Data, posData  
    Dim pos(30000) As Long  
    Dim intData(30000) As Long  
    Dim k(10000)  
    im = 1  
    i1 = 1  
    i2 = 1  
    P_Dat = 1  
    strData = txtData.Text  
    P_Data = Len(strData)  
    Do While InStr(strData, ",") <> 0  
        strData = Mid(strData, InStr(strData, ",") + 1)  
        im = im + 1  
    Loop  
    strData = txtData.Text  
    P_Data = Mid(strData, 1, P_Data - 1)  
    posData = 1  
    For i1 = 1 To im  
        pos(i1) = InStr(posData + 1, P_Data, ",")  
        posData = pos(i1)  
    Next i1  
    pos(im) = Len(P_Data)  
    k(1) = Mid(P_Data, 1, pos(1) - 1)  
    intData(1) = CInt(k(1))  
    For i2 = 2 To im - 1  
        k(i2) = Mid(P_Data, pos(i2 - 1) + 1, pos(i2) - pos(i2 - 1) - 1)  
        intData(i2) = CInt(k(i2))  
    Next i2  
    k(im) = Mid(P_Data, pos(im - 1) + 1, pos(im) - pos(im - 1))  
    intData(im) = CInt(k(im))  
    For i3 = 1 To im  
        Print intData(i3)  
    Next i3  
End Sub
```


以上で、RS232C を介したデータ集録プログラムが完成となる。データ集録用 Form および実行結果を以下に示す。

実行用 Form (図 29)

実行結果 (図 30)

図 29. RS232C からのデータ集録用 Form

-12,-23,-23,-23,-23,-23,-23,-23,-24,-23,-23,-23,-24,-24,-23,-23,-23,-24,-

図 30. 実行結果

RS-232C に使用する関数（Microsoft Comm Control 6.0：プロパティ名 mscPort）

シリアルポート設定

CommPort	機器が接続されているシリアルポートの番号を示す。 1～16までの整数で入力します。 mscPort.CommPort=1
Settings	通信速度，パリティ，データ長，ストップビット mscPort.Settings="9600,n,8,1"
Handshaking	ハードウェアの通信条件 mscPort.Handshaking=comRTS
RTSEnable	RTS(Request To Send：送信要求)線を使用するための設定 mscPort.RTSEnable=True
SThreshold	送信バッファ内のデータが指定されたバイト数より小さくなると onComm イベントを発生させるための設定。 mscPort.SThreshold=1
Rthreshold	受信データがバッファ内に指定されたバイト数以下受信されると、 onComm イベントを発生させるための設定。 mscPort.RThreshold=1

シリアルポートで発生するイベント処理

onComm	シリアルポートになんらかのイベントが発生した場合に発生する イベントが onComm イベントです。
CommEvent	onComm イベントが発生した場合、シリアルポートの状態を数値で 返すイベント
comEvSend	送信バッファ内のバイト数が SThreshold で指定された値よりも小 さくなると発生するイベント。
comEvReceive	RThreshold で指定されたバイト数を受信した場合、発生するイベ ント。Input プロパティでバッファからデータが削除されない限り、 発生し続けるイベント。
comEvCTS	Clear To Send の信号線の状態が変化した時に発生するイベント。
comEvDSR	Data Set Ready ラインの状態が変化した時に発生するイベント。 このイベントは、High から Low に変化した時にのみ発生します。
comEvCD	Carrier Detect ラインの状態が変化した場合に発生するイベント。
comEvRing	RI 線に信号が検出された場合に発生するイベント。 このイベントは、同期通信では使用しますが、非同期通信では使用さ れないので、通信する対象機器によっては信号線そのものが使用され ていない場合があります。
comEvEOF	EOF (End Of File) 文字を受信した場合に発生するイベント。 EOF は ASCII コードで 26h

シリアルポートのオープン／クローズ

PortOpen シリアルポートをオープンするかクローズにするかを設定する。
 mscPort.PortOpen=True

シリアルポートの送信バッファへの書込み

Output シリアルポートの送信バッファにデータを書込む。
 mscPort.Output=cmdSed.Text

シリアルポートの受信バッファからの読み込み

Input シリアルポートの受信バッファからデータを読み込む
 rsBuffer=mscPort.Input

表3. ASCII コード (Visual Basic ヘルプより抜粋)

Ctrl	Dec	Hex	Char	Code	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
^@	0	00		NUL	32	20	sp	64	40	@	96	60	`
^A	1	01	␣	SOH	33	21	!	65	41	A	97	61	a
^B	2	02	␣	SIX	34	22	"	66	42	B	98	62	b
^C	3	03	♥	EIX	35	23	#	67	43	C	99	63	c
^D	4	04	♦	EOI	36	24	\$	68	44	D	100	64	d
^E	5	05	♣	ENQ	37	25	%	69	45	E	101	65	e
^F	6	06	♠	ACK	38	26	&	70	46	F	102	66	f
^G	7	07	•	BEL	39	27	'	71	47	G	103	67	g
^H	8	08	⬢	BS	40	28	(	72	48	H	104	68	h
^I	9	09	○	HI	41	29	)	73	49	I	105	69	i
^J	10	0A	◻	LF	42	2A	*	74	4A	J	106	6A	j
^K	11	0B	◻	VI	43	2B	+	75	4B	K	107	6B	k
^L	12	0C	♀	FF	44	2C	,	76	4C	L	108	6C	l
^M	13	0D	♪	CR	45	2D	-	77	4D	M	109	6D	m
^N	14	0E	♪	SD	46	2E	.	78	4E	N	110	6E	n
^O	15	0F	※	SI	47	2F	/	79	4F	O	111	6F	o
^P	16	10	▶	SLE	48	30	0	80	50	P	112	70	p
^Q	17	11	◀	CS1	49	31	1	81	51	Q	113	71	q
^R	18	12	↑	DC2	50	32	2	82	52	R	114	72	r
^S	19	13	::	DC3	51	33	3	83	53	S	115	73	s
^T	20	14	☞	DC4	52	34	4	84	54	T	116	74	t
^U	21	15	☞	NAK	53	35	5	85	55	U	117	75	u
^V	22	16	■	SYN	54	36	6	86	56	V	118	76	v
^W	23	17	≡	EIB	55	37	7	87	57	W	119	77	w
^X	24	18	↑	CAN	56	38	8	88	58	X	120	78	x
^Y	25	19	↓	EM	57	39	9	89	59	Y	121	79	y
^Z	26	1A	→	SIB	58	3A	:	90	5A	Z	122	7A	z
^[	27	1B	+	ESC	59	3B	;	91	5B	[	123	7B	{
^\	28	1C	⌞	FS	60	3C	<	92	5C	\	124	7C	
^]	29	1D	+	GS	61	3D	=	93	5D	]	125	7D	}
^^	30	1E	▲	RS	62	3E	>	94	5E	^	126	7E	~
^_	31	1F	▼	US	63	3F	?	95	5F	_	127	7F	Δ†

† ASCII コードの 127 は、Del のコードです。MS-DOS では、ASCII コードの8 (BackSpace) と同じ効果があります。
 Del のコードの呼び出しには、Ctrl + BackSpace キーを使います。

7. A/D ボードからのデータ集録

直接、パソコンにアナログ信号をデジタル信号に変換するボードを装着して、データを集録する方法がある。こういった A/D ボードを利用したデータ集録は、個々の A/D ボード毎にデータ集録方法が異なるため、別個にドライバソフトを用意しておく必要がある。ボードによっては、インターネット上からダウンロードする必要があることもある。(古い A/D ボードの場合は、新しい OS で稼働させるためのドライバソフトが無い場合もある)

本実習では、ナショナルインスツルメンツ社製データ集録ボード DAQCard-500 を使用する。(図 31)



図 31. DAQCard-500

DAQCard を購入するとドライバソフトが添付されるが、OS が変わった場合には、ドライバを更新する必要がある。新しい OS 用のドライバソフトは、インターネット (DAQ ボード用サポートページ) からダウンロードするかメーカーから購入必要がある。

(図 32 : <http://www.ni.com/support/jp/daqsupp.htm>)



図 32. DAQ オンラインサポートページ

DAQCard-500 のピンの割付表

DAQCard-500 のコネクタピン

1	アナログ入力用 GND
2	0 c h アナログ入力
3	1 c h アナログ入力
4	2 c h アナログ入力
5	3 c h アナログ入力
6	4 c h アナログ入力
7	5 c h アナログ入力
8	6 c h アナログ入力
9	7 c h アナログ入力
10	+ 5 [DCV] 出力
11	A/D 変換完了信号
12	0 c h デジタル入力
13	1 c h デジタル入力
14	2 c h デジタル入力
15	3 c h デジタル入力
16	0 c h デジタル出力
17	1 c h デジタル出力
18	2 c h デジタル出力
19	3 c h デジタル出力
20	1 MHz 方形波出力
21	デジタル入出力 GND
22	カウンタ入力
23	カウンタ出力
24	カウンタスタート信号
25	スタート信号付クロック入力
26	スタート信号付カウンタ入力
27	デジタル入出力 GND
28	未使用
29	未使用
30	未使用

OS が変更された場合、ダウンロードしたドライバソフト（Nidaq69Jpn.exe）を解凍後、ドライバソフトのインストールおこなう。（図 33）

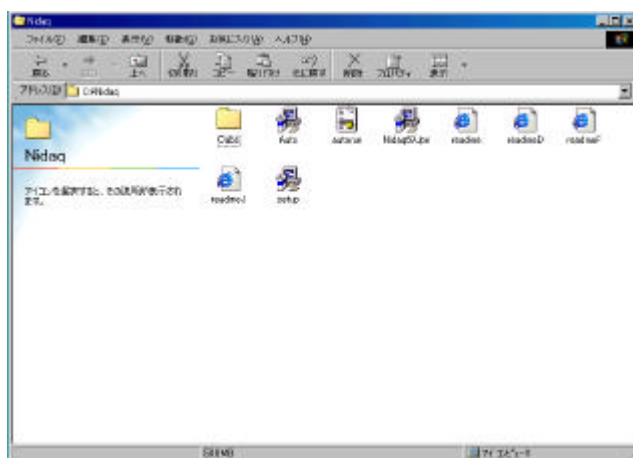


図 33. ダウンロードしたドライバソフト

ナショナルインスツルメンツ社の場合、Lab VIEW, Visual C++, Borland C++, Borland Delphi, Microsoft Internet Explorer 等がワンセットになっているため、今回は Visual Basic のドライバのみインストールする。（図 34）



図 34. NI-DAQ のデータ集録ドライバインストール画面



図 35. インストールをおこなったドライバソフト

A/D ボードからデータを集録するためには、プロジェクトエクスプローラ上で、A/D ボード用のドライバソフトの標準モジュールを追加する。(図 36)

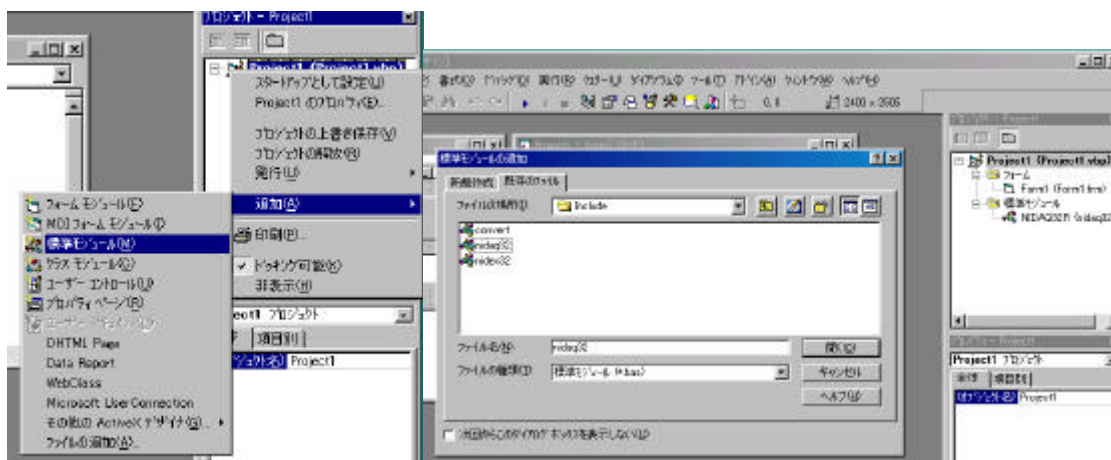


図 36. A/D ボード用標準モジュールの追加

課題 4. A/D ボードからのデータ集録プログラムの作成 (図 37)

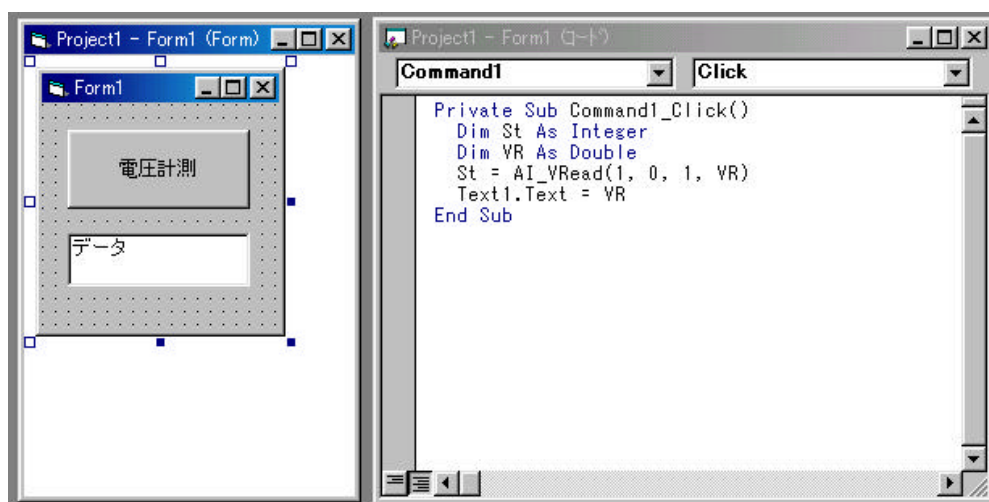


図 37. A/D ボードからのデータ集録プログラムと Form

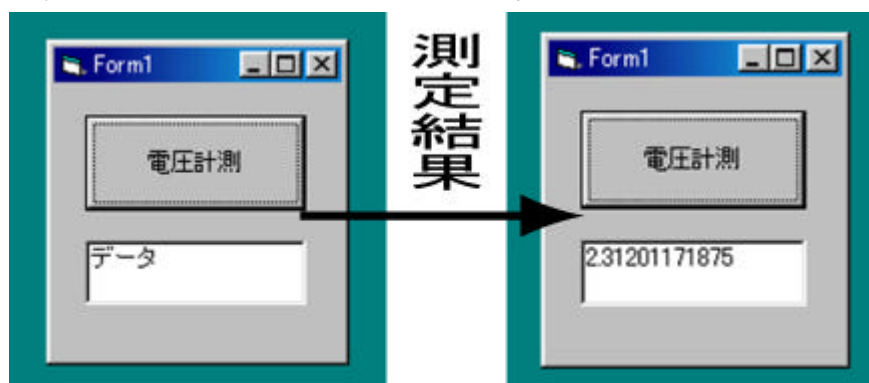


図 38. A/D ボードからのデータ集録プログラムの実行画面

アナログ入力に使用する関数

AI_Vread (①, ②, ③, ④)

- ① デバイス番号 : データ集録ボードの番号
- ② チャンネル数 : アナログ入力ポートのチャンネル数
- ③ インデックス番号 : アナログ入力ポートから取りこんだデータ指標
- ④ 計測データ : A/D変換されたデータ

課題 5.

A/D変換されたデータを 3000 取りこんで、Picture として表示するプログラムの作成する。

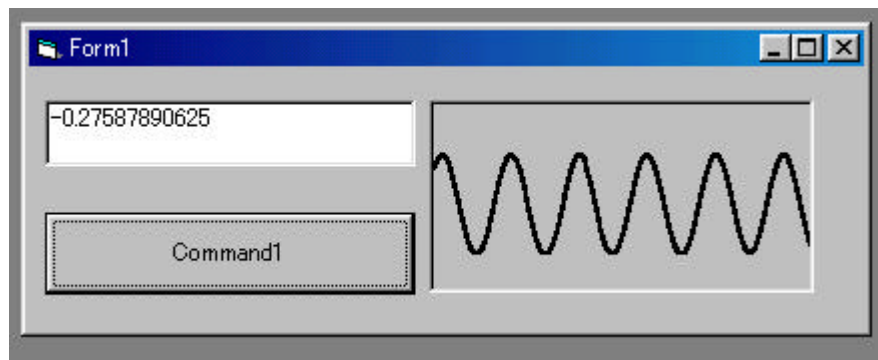


図 39. フォーム画面作成例

```
Private Sub Command1_Click()  
    Dim Vr(5000) As Double  
    For i = 1 To 3001  
        Call AI_VRead(1, 0, i, Vr(i))  
        Text1.Text = Vr(i)  
    Next i  
    Dim Y1(5000) As Double  
    Dim X1(5000) As Double  
    Picture1.Cls  
    For i = 1 To 3000  
        X1(i) = i * 1  
        Y1(i) = 2000 - 100 * (Vr(i) + 10)  
        Picture1.Circle (X1(i), Y1(i)), 10  
    Next i  
End Sub
```

今回使用したデータ集録ボード DAQCard-500 は、4 ビットの入力端子と 4 ビットの出力端子があります。今回使用したボードの場合、入力と出力が固定なので設定をおこなう必要はありませんが、各ポートをチャンネル毎に出力にするか入力にするかを設定する必要があります。

デジタル出力に使用する関数

DIG_Out_Port (①, ②, ③)

- ① デバイス番号：データ集録ボードの番号
- ② 出力ポート ：出力に使用しているポートのチャンネル番号
- ③ 出力データ ：ポートへ出力するデータ

課題 6. ボードのデジタル信号出力線の 0 チャンネルである 16 番端子からオープンコレクタ形式で出力信号を取り出し、LED を発光させるプログラムを作成する。

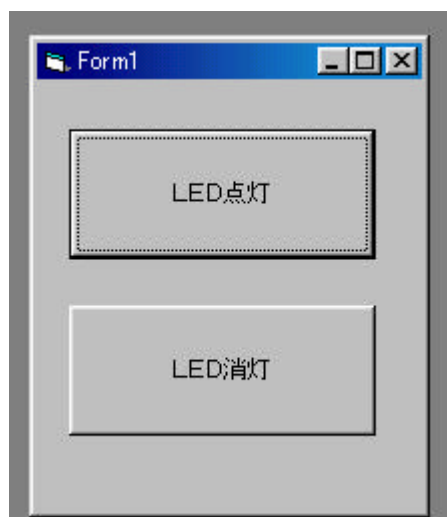


図 40. フォーム画面作成例

```
Private Sub Command1_Click()  
    Call DIG_Out_Port(1, 0, 15)  
End Sub
```

```
Private Sub Command2_Click()  
    Call DIG_Out_Port(1, 0, 0)  
End Sub
```