データ収集及びデジタル制御学生実験における Excel VBA の活用

（平成13年３月31日受付）

青　木　　清
杉　田　　徹

１．はじめに

近年の医療においてはコンピュータの利用が増加の一途をたどっており、診療放射線技師にとってコンピュータを自由に使いこなせることは非常に大切なこととなっている。このため、本学の学生実験においてもコンピュータを用いた実験を増やしつつある。こうした実験の中でも、コンピュータと外部機器との間でデータの交換をしたり外部機器を制御したりする場合、目的に応じた最適の動作をさせるためには、自分でプログラムを作成する必要が生じてくる。従来、このようなプログラミングにはコンピュータ付属の BASIC を用いることが多かった。しかし BASIC の場合、データを得ることは容易であっても、その結果を表やグラフの形にまとめるという作業には適していない。こうした作業には、最近は Microsoft 社の Excel を使うことが多い。Excel の場合はプログラミング言語として VBA (Visual Basic for Applications)が用意されており、BASIC と同様なプログラミングが可能であるばかりでなく、プログラムの中で Excel の機能を利用することもできるようになっている。従って VBA を利用すれば、外部とのデータ交換や機器の制御ばかりでなく、得られた結果を Excel の表にしたりグラフにしたりすることが容易にできると考えられる。そこで今回、我々は Excel VBA を用いて、データ収集やデジタル制御を行う学生実験の構築を試みた。

２．実験の構成

この実験のねらいは、コンピュータでのデータ収集や外部機器の制御が自分でプログラムを作ることにより自由に行え、さらに得られたデータを Excel の機能を用いて見やすい形に表現できることを学生に体験させることにある。このため、実験には次の３つの内容を盛り込んだ。

一つ目の実験は、ADC を利用した外部電圧の読み込みである。これには複数のチャネルを持つ ADC ボードを用い、各チャネルごとの電圧を表とグラフの形で表示させる。学生が電圧を変化させると、コンピュータ画面上の電圧も同時に変化するようにする。ただし、入力電圧が連続的に変化しても、測定電圧は ADC の分解能を単位とした不連続な値をとる。これにより、コン

ピュータの内部ではデータがデジタル的に処理されていることを学生は実感できるはずである。

二つ目の実験は、デジタル入出力ボードを利用したデジタル信号の入力および出力である。スイッチで設定した8チャネルのデジタル信号を読み込んで0～255の数値で表現させたり、逆に0～255の数値を入力して8チャネルのデジタル信号を出力し、これを8個のLEDの点滅により表示させる。学生はこれらの実験から、デジタル量の入出力方法を学ぶだけでなく、8ビットの信号で0～255の数値を表現できるという2進数の概念を視覚的に理解できる。

三つ目の実験では、一つ目と二つ目の実験を組み合わせることにより、温度の制御を実現する。発熱体としては、外部から制御の状態が観察しやすいよう、電球を用いる。電球なら極めて短時間の発熱でも光という形で観測でき、学生が理解しやすいからである。温度センサーとしてはサーミスタを用い、温度を電圧に変換する。この電圧をADCボードから読みとり、温度に変換して目標温度と比較する。測定温度が目標温度より低ければ電球ONの信号を、高ければOFFの信号をデジタル入出力ボードから出力する。このようにして電球を点滅させることにより温度の制御を行う。温度変化の様子はExcelの機能を利用して容易にグラフ表示できる。

3．入出力ボード

コンピュータを用いてデータ収集や制御を行う場合、特に重要となるのは使用する入出力ボードである。本実験の場合、ADCボードとしてNational Instruments社のPCI-6023Eを、デジタル入出力ボードとしてやはりNational Instruments社のPCI-6503を用いた。PCI-6023Eは12ビットの分解能を持ち、差動入力を8チャネル備えている。測定範囲は±10Vから±0.05Vまでの4種類が選択できる[1]。PCI-6503は入出力ポート（TTLレベル）を3つ備えており、ポートごとに8チャネルの信号を入力または出力できる[2]。どちらのボードもPCIバスに接続して使用する。National Instruments社からはこれらのボードを動かすためのソフトも提供されており、ボード購入時に付属してくる。ユーザーはプログラムの中で関数の形でこれらのソフトを利用できる[3]。このように使用環境が整っていることが、今回National Instruments社の製品を選んだ最大の理由である。

なお、今回利用したコンピュータはEPSON社のオフィスType-SSであり、薄型ではあるがPCIボードを2枚収容できる。使用OSはWindows NT4.0、利用ソフトはMicrosoft社のExcel 2000である。

4．VBAプログラムと動作結果

4．1　電圧の測定

電圧測定の実験ではADCボードを介して8チャネルの電圧を読み取り、それをリアルタイ

ムでExcelの表とグラフに表示する。Excelのシートにはあらかじめ0～7のチャネル番号と読み込みスタートボタン及び読み込みストップボタンを配置しておく。スタートボタンをマウスでクリックすると電圧読み込みプログラムが動作し、シートに測定電圧を書き込むとともに、その電圧を棒グラフで表示する。この動作はストップボタンがクリックされるまで繰り返され、電圧が変化した場合はすぐに画面上の表示も変化する。ストップボタンがクリックされると電圧の読み込みは終了し、再びスタートボタンがクリックされるのを待つ。図1にはこの電圧読み込みルーチンのフローチャートを、図2にはVBAプログラムを示す。グラフの作成はExcelのChartWizardを用いており、非常に簡単に行える。しかも図1からわかるように、電圧読み込み前にグラフ設定を行って

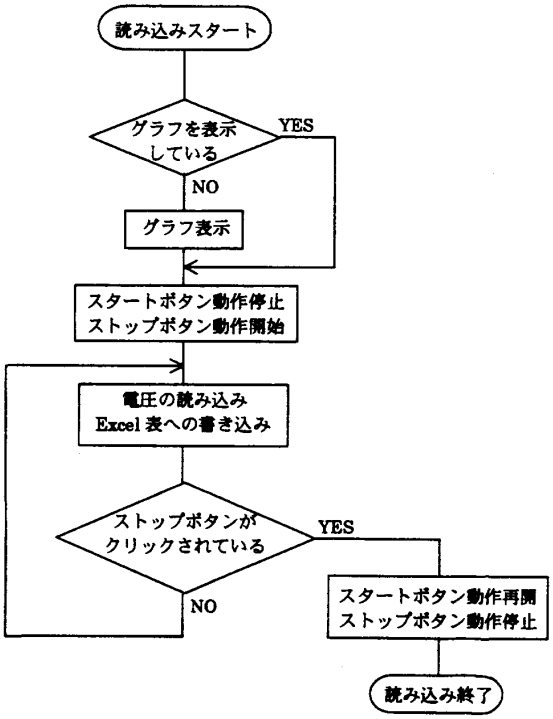

図1　電圧読み込みルーチンのフローチャート

```
Option Explicit
Option Base 0
'
Dim flgStop As Boolean

Private Sub cmdStart_Click()

    '変数の宣言
    Dim iStatus As Integer
    Dim iDevice As Integer
    Dim iChan As Integer
    Dim iGain As Integer
    Dim dVoltage As Double
    Dim Graph1 As ChartObject

    '変数の初期値設定
    iDevice = 1
    iGain = 1
    flgStop = False

    'グラフ設定
    If Sheet1.ChartObjects.Count = 0 Then
        Set Graph1 = Sheet1.ChartObjects.Add(130, _
                     20, 250, 200)
        Graph1.Chart.ChartWizard
            Source:=Sheet1.Range("A1:B9"), _
            Gallery:=xlColumn, _
            Format:=1, _
            PlotBy:=xlColumns, _
            CategoryLabels:=1, _
            SeriesLabels:=1, _
            HasLegend:=False, _
            Title:="電圧測定", _
            CategoryTitle:="チャネル番号", _
            ValueTitle:="電圧(V)"
        Graph1.Chart.Axes(xlValue).MaximumScale = 5
    End If

    'スタートボタンの一時的動作停止
    cmdStop.Enabled = True
    cmdStart.Enabled = False
    cmdStop.Activate

    '電圧の読み書き
    Do
        For iChan = 0 To 7
            iStatus = AI_VRead(iDevice, iChan, iGain, _
                      dVoltage)
            If (iStatus = 0) Then
                Sheet1.Cells(iChan + 2, 2) = Format( _
                      dVoltage, "   #0.000")
            Else
                MsgBox "Read error!  Error code=" + _
                      Str(iStatus), , "エラー"
                Exit Sub
            End If
        Next
        DoEvents
    Loop Until flgStop = True

    'スタートボタンの動作再開
    cmdStart.Enabled = True
    cmdStop.Enabled = False

End Sub

Private Sub cmdStop_Click()
    flgStop = True
    cmdStart.Enabled = True
    cmdStop.Enabled = False
    cmdStart.Activate
End Sub
```

図2　電圧測定のVBAプログラム（左下は右上に続く。）

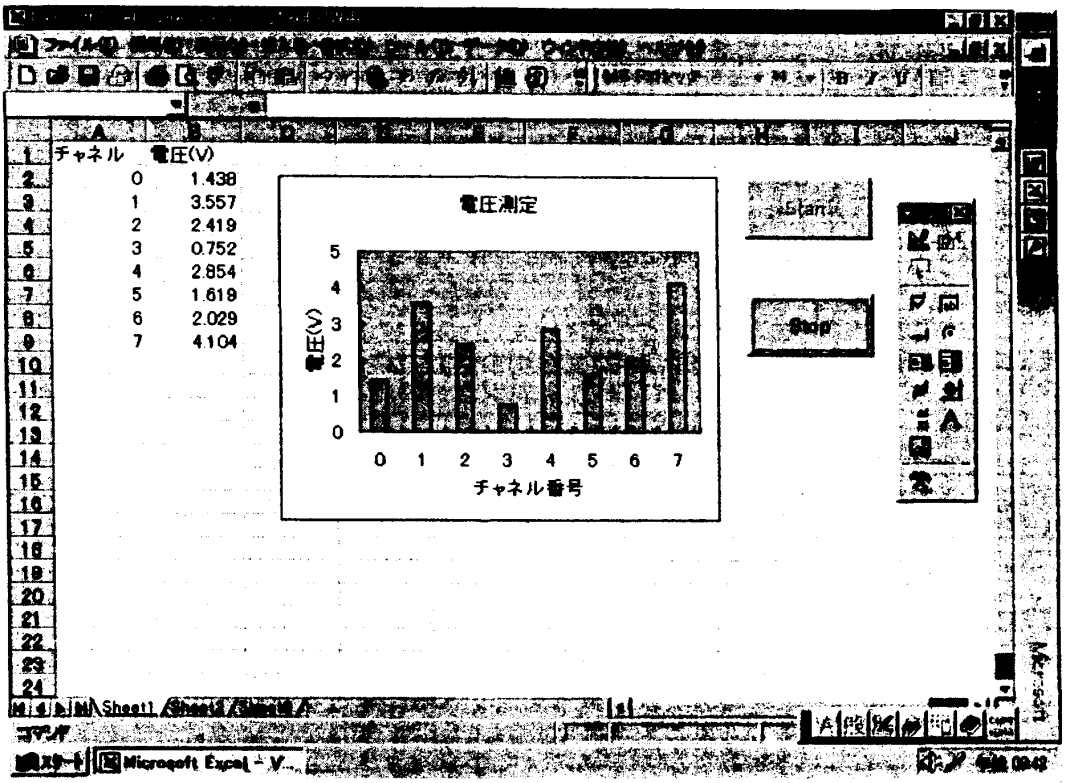

図3　電圧測定プログラム実行中の画面

おくと、以後はグラフ表示の指示をしなくてよい。読み込んだ電圧の表示は自動的に行われる。これがExcel VBAの特徴であり、BASICに比べてプログラミングが非常に簡単になる。なお、このプログラムではグラフの電圧軸の最大値を5 [V]に設定しているが、これを省略すると、最大値は表示値にあわせて自動的に変更される。電圧の読み込みを行うのはAI_VReadという関数であり[4]、これはNational Instruments社から提供されている。この関数では、2番目の引数iChanにより測定チャネルを、3番目の引数iGainにより測定範囲を変えている。もし読み込みに失敗したときは、iStatusの値から原因を知ることが出来る。図3にはプログラム実行中の画面を示す。このように各チャネルの電圧が表とグラフでリアルタイムに表示されるので、電圧変化を視覚的に観察することができる。

4．2　デジタル信号の読み取りと出力

デジタル信号の入出力を行うVBAプログラムを図4に示す。プログラム中のDIG_Prt_Configは入出力ポートを設定する関数、DIG_Out_Prtはデジタル信号出力関数、DIG_In_Prtはデジタル信号入力関数である[4]。これらはいずれもNational Instruments社から提供されている。

Excelシート上にはあらかじめデジタル信号の読み取りボタンと出力ボタンを配置しておく。プログラム実行中に読み取りボタンをクリックすると、入出力ボードより8チャネルのTTL

```
Option Explicit
Option Base 0

'共通変数の宣言
Dim iDevice As Integer
Dim iPort As Integer
Dim iMode As Integer
Dim iDir As Integer
Dim iStatus As Integer
Dim iPattern As Long

Private Sub cmdOut_Click()

    '変数の初期値設定
    iDevice = 2     'digital board
    iPort = 0       'port A
    iMode = 0       'no-handshake
    iDir = 1        'data out

    '出力ポートの設定
    iStatus = DIG_Prt_Config(iDevice, iPort, _
              iMode, iDir)
    If (iStatus <> 0) Then
        MsgBox "Config error!  Error code=" + _
               Str(iStatus), , "エラー"
        Exit Sub
    End If

    'デジタル信号の出力
    iPattern = Sheet1.Cells(3, 2)
    If ((iPattern < 0) Or (iPattern > 255)) Then
        MsgBox "正しい数値(0-255)を入力してください", _
               , "入力ミス"
        Sheet1.Cells(3, 2).Activate
        Exit Sub
    End If
    iStatus = DIG_Out_Prt(iDevice, iPort, iPattern)
    If (iStatus <> 0) Then
        MsgBox "Data out error!  Error code=" + _
               Str(iStatus), , "エラー"
        Exit Sub
    End If
    Sheet1.Cells(3, 2).Activate

End Sub
```

```
Private Sub cmdRead_Click()

    '変数の初期値設定
    iDevice = 2     'digital board
    iPort = 1       'Port B
    iMode = 0       'no-handshake
    iDir = 0        'read

    '入力ポートの設定
    iStatus = DIG_Prt_Config(iDevice, iPort, _
              iMode, iDir)
    If (iStatus <> 0) Then
        MsgBox "Config error!  Error code=" + _
               Str(iStatus), , "エラー"
        Exit Sub
    End If

    'デジタル信号の読み取り
    iStatus = DIG_In_Prt(iDevice, iPort, iPattern)
    If (iStatus = 0) Then
        Sheet1.Cells(1, 2) = iPattern
    Else
        MsgBox "Read error!  Error code=" + _
               Str(iStatus), , "エラー"
        Exit Sub
    End If
    Sheet1.Cells(3, 2).Activate

End Sub
```

図４　デジタル信号の読み取りと出力を行うVBAプログラム

レベル信号を読み取り、シートの1行目に10進数で表示する。この実験での外部デジタル信号の設定は８個のスイッチで行い、学生が自由に変えられるようにしてある。シートの３行目には、出力したいデジタル信号に対応する０～255の数値を入力しておく。出力ボタンをクリックすると、この数値に対応する８チャネルのデジタル信号が入出力ボードより出力され、８個のLEDの点滅により表示される。

４．３　温度の制御

電圧読み込み実験とデジタル信号出力実験とを組み合わせることにより、温度制御の実験を実現できる。使用する関数もこれまでと同じである。作成したプログラムを図５に、実行後の画面を図６に示す。プログラムにおける電圧読み込みは前と同じくAI_VRead関数により行う。読み込む電圧はサーミスタで測定温度を電圧に変換したものであり、１［℃］が0.01［V］になるようにしてある。従って、プログラム中で［V］単位の測定電圧を100倍すれば、［℃］単位の温度が得られる。この測定温度をExcelシートの1行目に書き込むとともに、前もって３行

```
Option Explicit
Option Base 0

Dim flgStop As Boolean

Private Sub cmdStart_Click()

    '変数の宣言
    Dim dTemp As Double
    Dim dTempAim As Double
    Dim iStatus As Integer
    Dim iDeviceA As Integer
    Dim iChan As Integer
    Dim iGain As Integer
    Dim dVoltage As Double
    Dim iDeviceD As Integer
    Dim iPort As Integer
    Dim iMode As Integer
    Dim iDir As Integer
    Dim iPattern As Long
    Dim StartTime As Single
    Dim MesrTime As Single
    Dim iWriteCount As Integer

    '変数の初期値設定
    flgStop = False
    iDeviceA = 1     'Analog board
    iChan = 0        'Analog input channel number
    iGain = 10       'Gain=10
    iDeviceD = 2     'Digital board
    iPort = 0        'Port A
    iMode = 0        ' No handshaking
    iDir = 1         'output

    '目標温度読み取り
    dTempAim = Sheet1.Cells(3, 2)
    If dTempAim < 0 Then
        MsgBox "正しい目標温度を入力してください。", _
               , "入力ミス"
        Sheet1.Cells(3, 2).Activate
        Exit Sub
    End If

    '出力ポートの設定
    iStatus = DIG_Prt_Config(iDeviceD, iPort, _
              iMode, iDir)
    If (iStatus <> 0) Then
        MsgBox "Config error!  Error code=" + _
               Str(iStatus), , "エラー"
        Exit Sub
    End If

    'スタートボタンの一時的動作停止
    cmdStop.Enabled = True
    cmdStart.Enabled = False
    cmdStop.Activate

    'シート初期化
    With Sheet1
        .Range("D:E").ClearContents
        .Columns("D").ColumnWidth = 10
        .Columns("E").ColumnWidth = 7
        .Cells(1, 4) = "経過時間(s)"
        .Cells(1, 5) = "温度(℃)"
    End With

    '温度制御
    iWriteCount = 0
    Do
        iStatus = AI_VRead(iDeviceA, iChan, _
                  iGain, dVoltage)
        If (iStatus <> 0) Then
            MsgBox "Voltage read error! Error code=" _
                   + Str(iStatus), , "エラー"
            Exit Do
        End If
        MesrTime = Timer
        dTemp = dVoltage * 100#
        Sheet1.Cells(1, 2) = Format(dTemp, _
                "  #0.000")
        If iWriteCount = 0 Then StartTime = _
                MesrTime
        If (MesrTime >= (StartTime + _
                iWriteCount)) Then
            Sheet1.Cells(iWriteCount + 2, 4) = Format _
                    ((MesrTime - StartTime), " ##0.0")
            Sheet1.Cells(iWriteCount + 2, 5) = Format _
                    (dTemp, "  #0.000")
            Sheet1.Cells(5, 2) = Format((MesrTime - _
                    StartTime), " ##0.0")
            iWriteCount = iWriteCount + 1
        End If
        If (dTemp < dTempAim) Then iPattern = 1 Else _
                iPattern = 0
        iStatus = DIG_Out_Prt(iDeviceD, iPort, iPattern)
        If iStatus <> 0 Then
            MsgBox "Digital out error!  Error code=" _
                   + Str(iStatus), , "エラー"
            Exit Do
        End If
        DoEvents
    Loop Until flgStop = True

    '制御終了時の消灯
    iPattern = 0
    iStatus = DIG_Out_Prt(iDeviceD, iPort, iPattern)
    If (iStatus <> 0) Then
        MsgBox "Digital out error!  Error code=" + _
               Str(iStatus), , "エラー"
        Exit Sub
    End If

    'スタートボタンの動作再開
    cmdStart.Enabled = True
    cmdStop.Enabled = False
    Sheet1.Cells(3, 2).Activate

End Sub

Private Sub cmdStop_Click()
    flgStop = True
    cmdStart.Enabled = True
    cmdStop.Enabled = False
End Sub
```

図5　温度制御のVBAプログラム

目に入力しておいた目標温度と比較する。その結果、測定温度が目標温度より低ければ数値1を、高ければ数値0をDIO_Out_Prt関数を使って出力する。そして、出力が1のときは発熱体である電球が点灯し、出力が0のときは消灯するようにしてある。さらに、温度変化を記録しておくため、1秒おきに経過時間と測定温度をそれぞれ4列目と5列目に書き込んでいる。測定時間が長くなるとこれらの書き込みが画面から見えなくなるため、図6のように5行目に経過時間を表示するようにしてある。

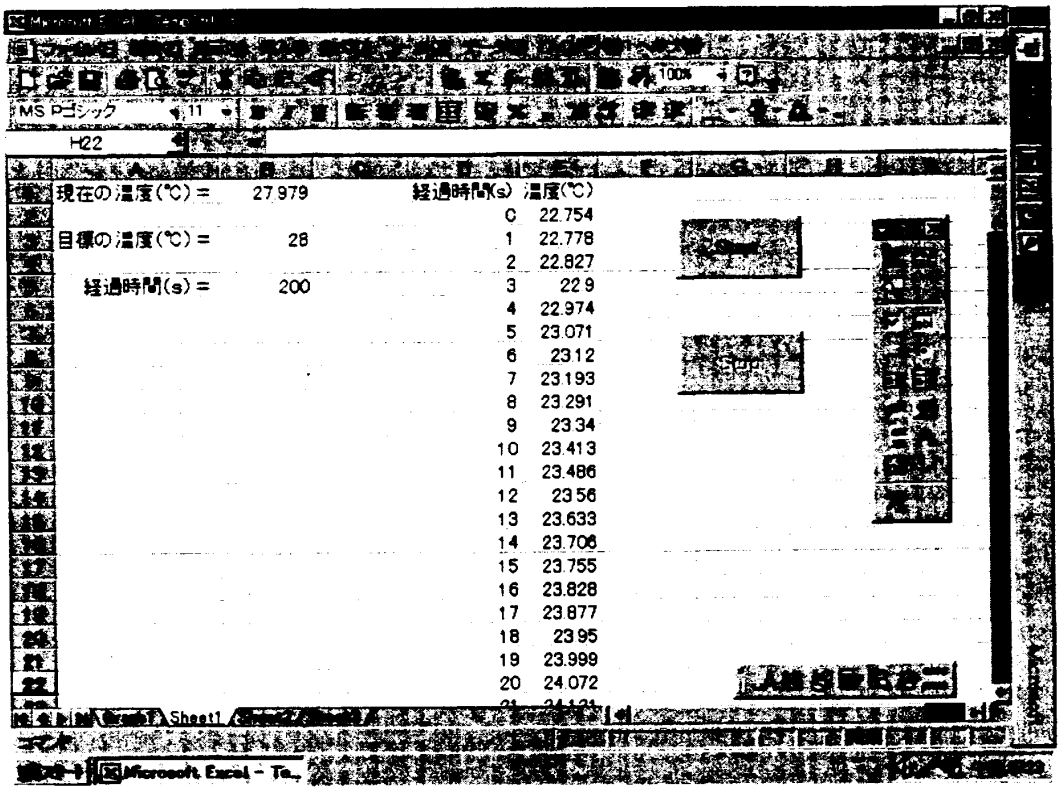

図６　温度制御プログラム実行後の画面

目標温度を気温より少し高い値に設定してこのプログラムを実行すると、最初しばらくは電球が点灯し続け、目標値に達すると煩雑に点滅するようになる。発熱が発光と対応しているため、この制御の様子は非常に分かりやすい。このようにして得られた温度変化の例を図７に示す。これは目標温度が28［℃］の場合で、正確に目標値に制御されている様子がわかる。この図もExcelで作成したものであり、測定データがExcelシート上に書き込んであるため、このようなグラフもすぐに作ることが出来る。

温度測定において、ADCボードの測定範囲は±0.5［V］(iGain＝10)に設定してある。従って、12ビットのADCなので、電圧分解能は0.24［mV］すなわち温度で0.024［℃］となる。このため、温度変化が小さい範囲で測定値をグラフ化すると、測定温度が0.024［℃］間隔でプロットされているのが見える。このようにして、アナログ量のデジタル化において常に生じる不連続を、視覚的に理解することが出来る。

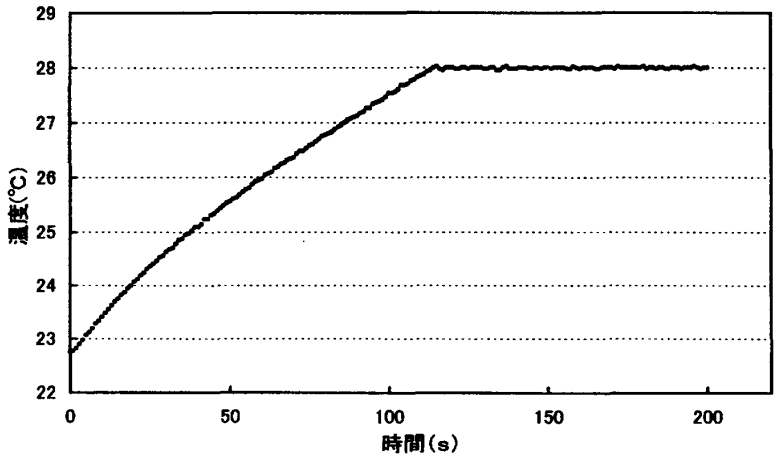

図７　目標温度28℃のときの温度制御特性

4．むすび

データ収集とデジタル制御を行う学生実験において、動作プログラムを Excel VBA で作成した。実験内容は以下の 3 つである。

1） ADC ボードを利用して電圧を読み込む。

2）デジタル入出力ボードを利用してデジタル信号の読み取りと出力を行う。

3）サーミスタと電球を利用して温度制御を行う。ここではサーミスタで温度を電圧に変え、この電圧を ADC ボードより読み込む。そして、測定温度が目標温度より低ければ電球の ON 信号を、高ければ OFF 信号をデジタル入出力ボードより出力し、温度を制御する。

ボードを動作させるのに必要なソフトウェアはメーカーから提供されている関数を利用した。プログラムを Excel VBA で作ったため、測定中の電圧変化のグラフ表示や温度制御特性のグラフ作成が非常に容易に行えた。また、アナログ量のデジタル化で必ず生じる不連続も、グラフから視覚的に実感することが出来た。

この実験の実施学年は 3 年次である。学生は 1、2 年次での実験レポート作成にかなり Excel を使用しており、既になじみのあるソフトとなっている。従ってこの実験により、これまでの使用経験とつながる形でプログラミングを体験できる。これによりプログラミングを身近なものに感じ、コンピュータをより自由に使いこなせるようになることが期待される。

［参考文献］

1） *6023E/6024E/6025E User Manual,* National Instruments Corporation (1999).

2） *PCI-DIO-96/PXI-6508/PCI-6503 User Manual,* National Instruments Corporation (1998).

3） *NI-DAQ User Manual for PC Compatibles, Version 6.7,* National Instruments Corporation (2000).

4） *NI-DAQ Function Reference Help, Version 6.7,* National Instruments Corporation (2000).