

# 初めての自動計測 Excel/VBA 編

キーサイトテクノロジー株式会社

2020.04.24

石井 幹



# こんな場面を想定したセミナーです

## 本セミナーのゴール

計測器とコンピュータの接続方法を知りたい

Excelのマクロを使用して、ある程度定型化している  
測定ルーチンを自動化したい

本セミナーによって・・・

- 計測器とコンピュータの通信に必要な基本的なツールが使用できるようになります
- 簡単な計測器の遠隔制御やデータの取り込みなどが出来るようになります

# 初めての自動計測 Excel/VBA 編

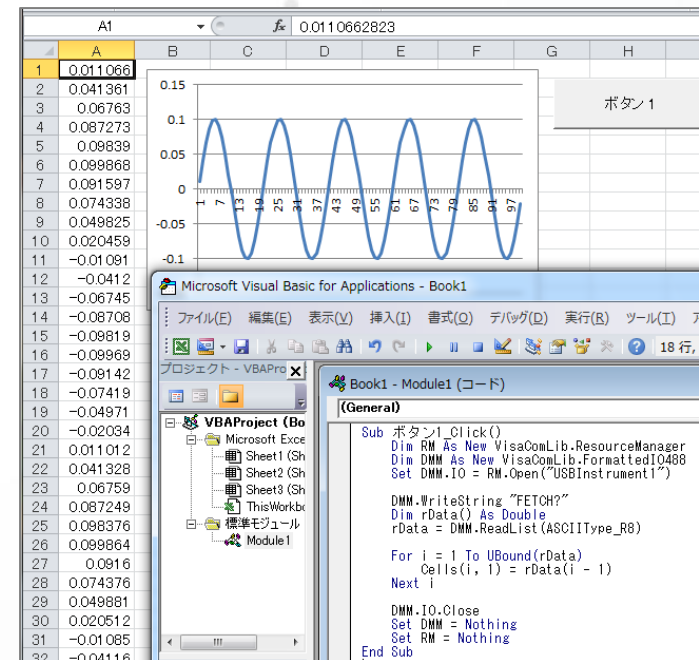
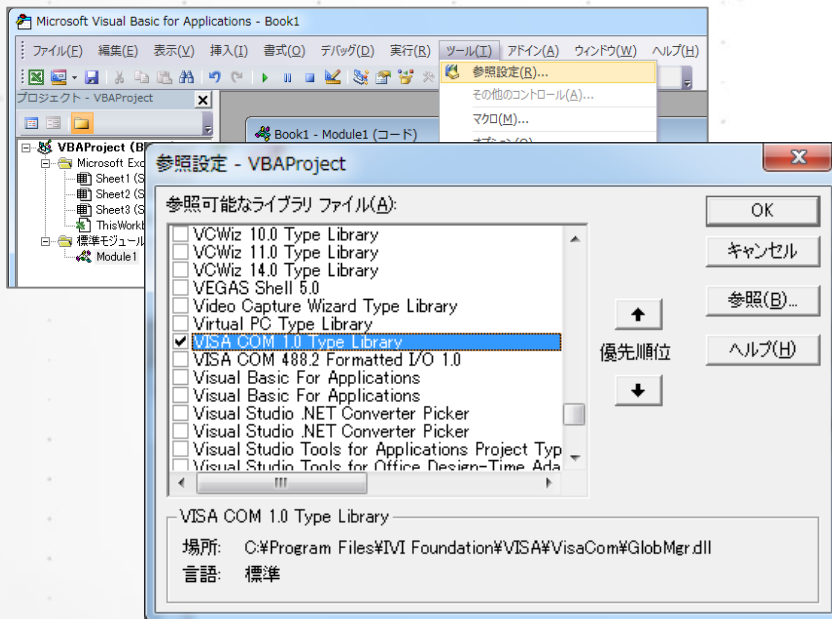
## 目次

- Excel と IO Libraries Suite で実現する自動計測
  - Excel/VBA はプログラム開発環境
  - 自動計測は難しいのか？
  - 業界標準のコマンド体系 SCPI
  - 業界標準のライブラリ VISA
  - IO Libraries Suite の紹介
  - Excel/VBA で実現する自動計測の特徴
- Excel/VBA 簡単なサンプルプログラム
  - Excel/VBA から VISA COM を使用するための準備
  - DMM を制御する簡単なサンプルプログラム
  - DC 電源を制御する簡単なサンプルプログラム
- Excel/VBA 実用的なサンプルプログラム
  - DC 電源と DMM を制御するサンプルプログラム
- 計測技術情報ライブラリの紹介

# Excel と IO Libraries Suite で実現する自動計測

## EXCEL/VBA はプログラム開発環境

- Excel には、標準で VBA (Visual Basic for Application) の開発環境が付属しています。
  - 標準の Excel だけで、VBA プログラムの作成、デバッグ、実行が可能
  - VBA を使用することで、Excel の機能拡張やカスタマイズが可能
  - Excel/VBA は一般書籍やインターネット上から、情報の入手が可能
- Excel/VBA には、測定器を制御する機能はありません



# Excel と IO Libraries Suite で実現する自動計測

## 自動計測は難しいのか？

最新の技術を取り入れることで、自動計測のハードルは非常に低くなります

自動計測のハードル	ハードルを下げる最新技術
最近のPCに標準搭載されていない I/Fを使用 (GPIB, RS-232Cなど)	USB, LANなどを搭載した計測器
メーカー・型番ごとにコマンド体系が異なる	業界標準コマンドSCPI
I/Fごとに通信ライブラリが異なる	業界標準ライブラリVISA
プログラミング言語の習得	VBAは初心者向けの文献がインターネットに多数

# Excel と IO Libraries Suite で実現する自動計測

## 業界標準のコマンド体系 SCPI

- SCPI (Standard Command for Programmable Instrument ; スキッピー)とは？
  - 計測器を制御するコマンド体系です
  - 業界標準化団体 (IVI-Foundation) が仕様を決めて、計測器ベンダ等が実装をしています (キーサイトの多くの計測器はSCPIコマンドで制御することが可能です)
- SCPIのメリット
  - 一度作成したプログラムの他の計測器などへの応用が容易
    - 例えば、「マルチメータに対して直流電圧を10 Vのレンジで測定して結果を返して下さい」というコマンドは、キーサイトのマルチメータだけでなく、他ベンダのSCPI対応マルチメータにも適用可能です
    - 計測器のモデル入れ替えや代替機運用時でも、プログラムの修正は不要 (または非常に軽微な修正のみ)
  - 一度習得してしまえば、他の計測器制御への応用が簡単

# Excel と IO Libraries Suite で実現する自動計測

## 業界標準のライブラリ VISA

- VISA (Virtual Instruments Software Architecture; ビサ)とは？
  - 計測器とPCを接続するインターフェースのドライバ・ソフトウェアです
  - 業界標準化団体が仕様を決めて、計測器ベンダ等が実装をしています (IO Libraries Suiteには、キーサイトが実装したVISAが含まれています)
  - LAN, GPIB, USB, RS-232C, VXI, PXIなどのインターフェースをサポート
- VISAのメリット
  - インターフェースに依存しないプログラムを作成可能
    - GPIB接続していた機器をUSB接続に変えても、プログラムの修正は不要 (または非常に軽微な修正のみ)
  - ライブラリ提供ベンダに依存しないプログラムを作成可能
    - キーサイト提供VISA上で動作するプログラムを、最小限の修正で別ベンダ提供VISA上で動作させることが可能

VISA: C 言語の関数群として提供

VISA COM: Windows の COM として提供

VISA.NET: Windows の .NET アセンブリとして提供



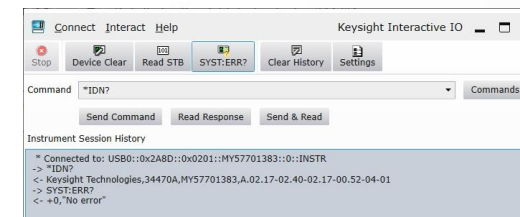
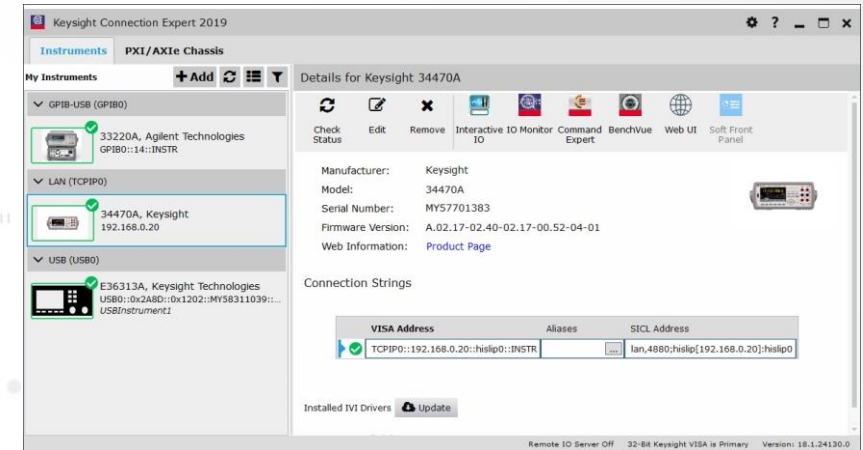
	Excel/VBA Visual Basic 6	Visual Basic.net C#	Visual C++
VISA	△	△	◎
VISA COM	◎	◎	△
VISA.NET	×	◎	◎

# Excel と IO Libraries Suite で実現する自動計測

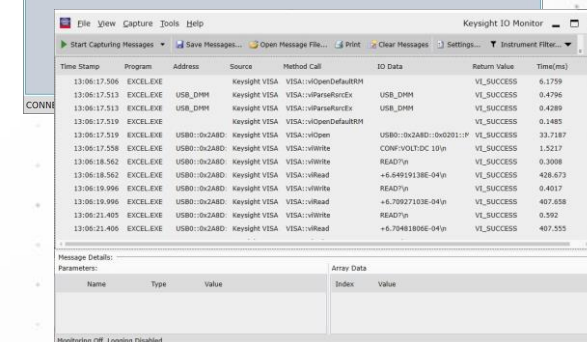


## IO LIBRARIES SUITE の紹介

- IO Libraries Suite とは、
  - キーサイトの WEB からダウンロードしていただける無償のソフトウェア
  - PC を測定器を接続するために使用するソフトウェア
- IO Libraries Suite が提供する3つの機能
  - プログラムが測定器と通信するための VISA COM などのライブラリ
  - キーサイト製インタフェースをWindows で使用するためのドライバ
  - PC と測定器の接続を確認するためのユーティリティプログラム
- IO Libraries Suite に含まれる VISA COM ライブラリは
  - 測定器にコマンドを送り、測定器から応答を受け取れます
  - Excel/VBA から呼び出せます



インタラクティブIO



IOモニタ

# Excel と IO Libraries Suite で実現する自動計測

## EXCEL/VBA と VISA COM で実現する自動計測の特徴

- 簡単に自動計測を実現する3つの方法の比較

	コマンドの情報	ライブラリの情報	プログラムの文法	制御できる測定器	制御フロー
BenchVue ベンチアプリケーション	不要	不要	不要	1台	不可
BenchVue テストフロー	不要	不要	要 (簡易)	複数台	可 (簡易)
Excel/VBA と VISA COM	要	要	要	複数台	可 (高度)

- Excel/VBA と VISA COM を使用する場合、コマンドやライブラリの情報に加え、プログラムの文法の知識が必要です。
- ただし、Excel/VBA では、複数の測定器に対して、繰り返しや条件分岐などの高度な制御フローを実現できません。

# Excel/VBAとVISA COMを使用した自動測定

簡単なサンプルプログラム

## デモンストレーション

The screenshot shows the Microsoft Visual Basic for Applications editor for a VBA Project (psTest.xlsm). The code in the 'ボタン1\_Click' event procedure is as follows:

```
Public Declare Sub Sleep Lib "KERNEL32.dll" (ByVal dwMilliseconds As Long)
Sub ボタン1_Click()
    Dim RM As New VisaComLib.ResourceManager
    Dim PS As New VisaComLib.ResourceManager
    Set PS.IO = RM.Open("USBInstrument2")

    PS.WriteString "VOLT 0"
    PS.WriteString "CURR 0.1"
    PS.WriteString "OUTPUT ON"

    Dim outVolt As Double
    Dim measVolt As Double
    Dim measCurr As Double
    Dim row As Integer
    row = 1
    For outVolt = 4.5 To 5.5 Step 0.1
        PS.WriteString "VOLT " & outVolt
        Sleep (1000)
        PS.WriteString "MEAS:VOLT?"
        measVolt = PS.ReadNumber(ASCIIType_R8)
        PS.WriteString "MEAS:CURR?"
        measCurr = PS.ReadNumber(ASCIIType_R8)
        Cells(row, 1) = outVolt
        Cells(row, 2) = measVolt
        Cells(row, 3) = measCurr
        DoEvents
        row = row + 1
    Next outVolt
    PS.WriteString "OUTPUT OFF"

    Dim psErr As String
    PS.WriteString "SYST:ERR?"
    psErr = Replace(PS.ReadString, vbCrLf, "")
    Cells(row, 1) = "Power supply ERROR = " & psErr

    PS.IO.Close
    Set PS = Nothing
    Set RM = Nothing
End Sub
```

The screenshot shows the Keysight Connection Expert 2019 software interface. The 'My Instruments' pane lists the following instruments:

- 33220A, Agilent Technologies GPIB0::14::INSTR
- 34470A, Keysight 192.168.0.20
- E36313A, Keysight Technologies USB::0x2ABD::0x1202::MYS83110391... USBInstrument1

The 'Details for Keysight 34470A' pane shows the following information:

- Manufacturer: Keysight
- Model: 34470A
- Serial Number: MYS7701383
- Firmware Version: A.02.17-02.40-02.17-00.52-04-01
- Web Information: Product Page

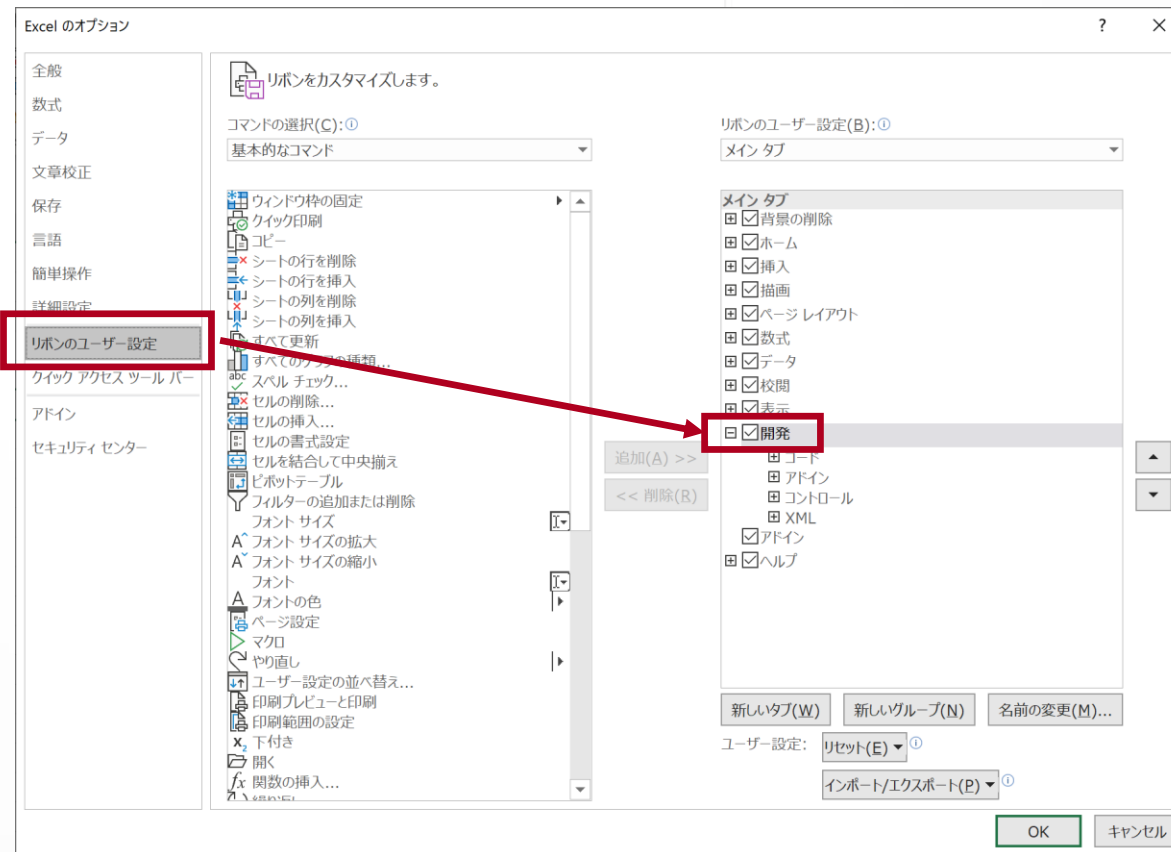
The 'Connection Strings' pane shows the following VISA Address:

VISA Address: TCP::192.168.0.20::hslp0::INSTR

# Excel/VBA 簡単なサンプルプログラム

## EXCEL/VBA からVISA COM を使用するための準備

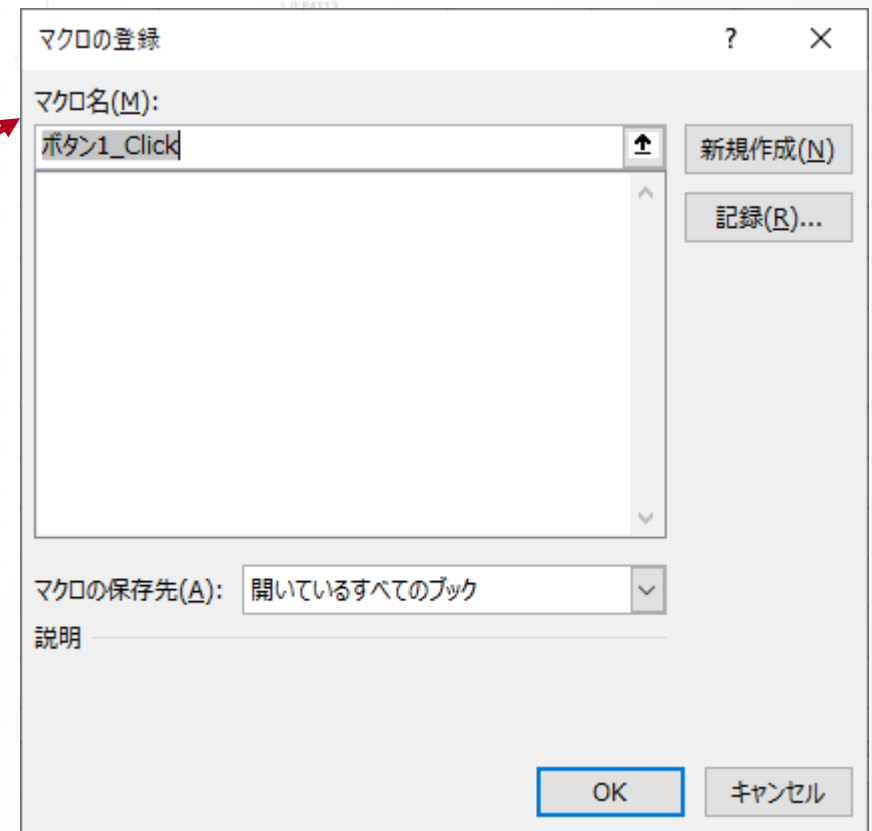
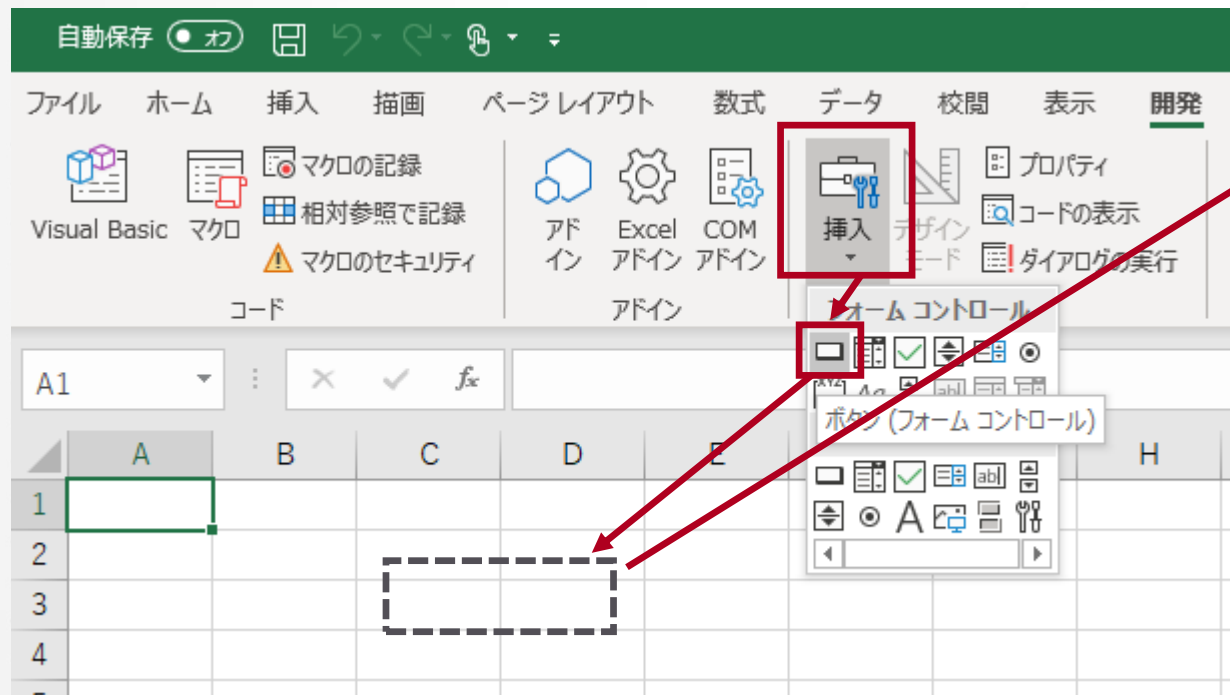
- Excel で VBA を使用するには、「開発」タブを表示させると便利です
- 「開発」タブは、「ファイル」 > 「オプション」 から「リボンのユーザ設定」を選択し、「開発」にチェックを入れる



# Excel/VBA 簡単なサンプルプログラム

## EXCEL/VBA からVISA COM を使用するための準備

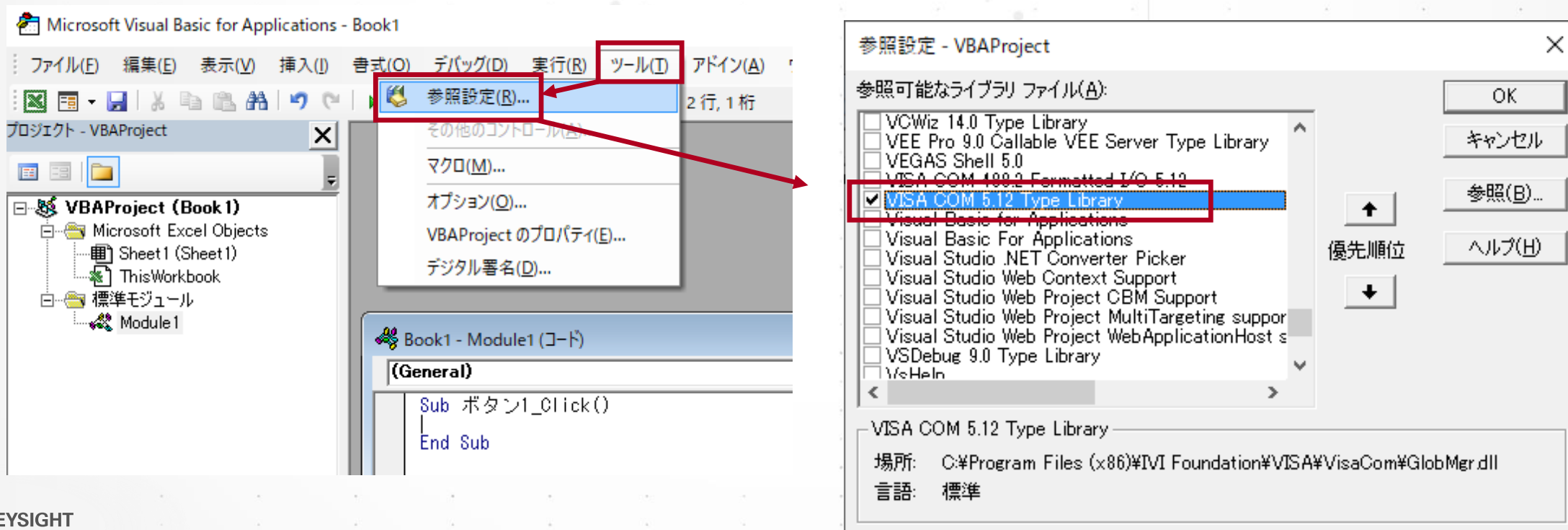
- Excel を起動し、「開発」タブから、「挿入」 > 「ボタン」 を実行
- シート上でボタンを配置する場所をドラッグ
- 「マクロの登録」で「新規作成」を選択



# Excel/VBA 簡単なサンプルプログラム

## EXCEL/VBA からVISA COM を使用するための準備

- Visual Basic for Applications から、「ツール」>「参照設定」を実行
- VISA COM x.xx Type Library にチェックを入れて「OK」
  - x.xx はバージョンです。お使いの IO Libraries Suite のバージョンにより異なります。この例では 5.12 です。
  - 表示されるVISA COM Type Library を参照に追加してください。



# VISA COMにおける3つのオブジェクトの関係

ここではクラスから生成されたインスタンスをオブジェクトと呼んでいます。

## プログラムの最初

3つのオブジェクト生成

```
Dim RM = New VisaComLib.ResourceManager  
Dim DMM = New VisaComLib.FormattedIO488  
Set DMM.IO = RM.Open("USB_DMM")
```

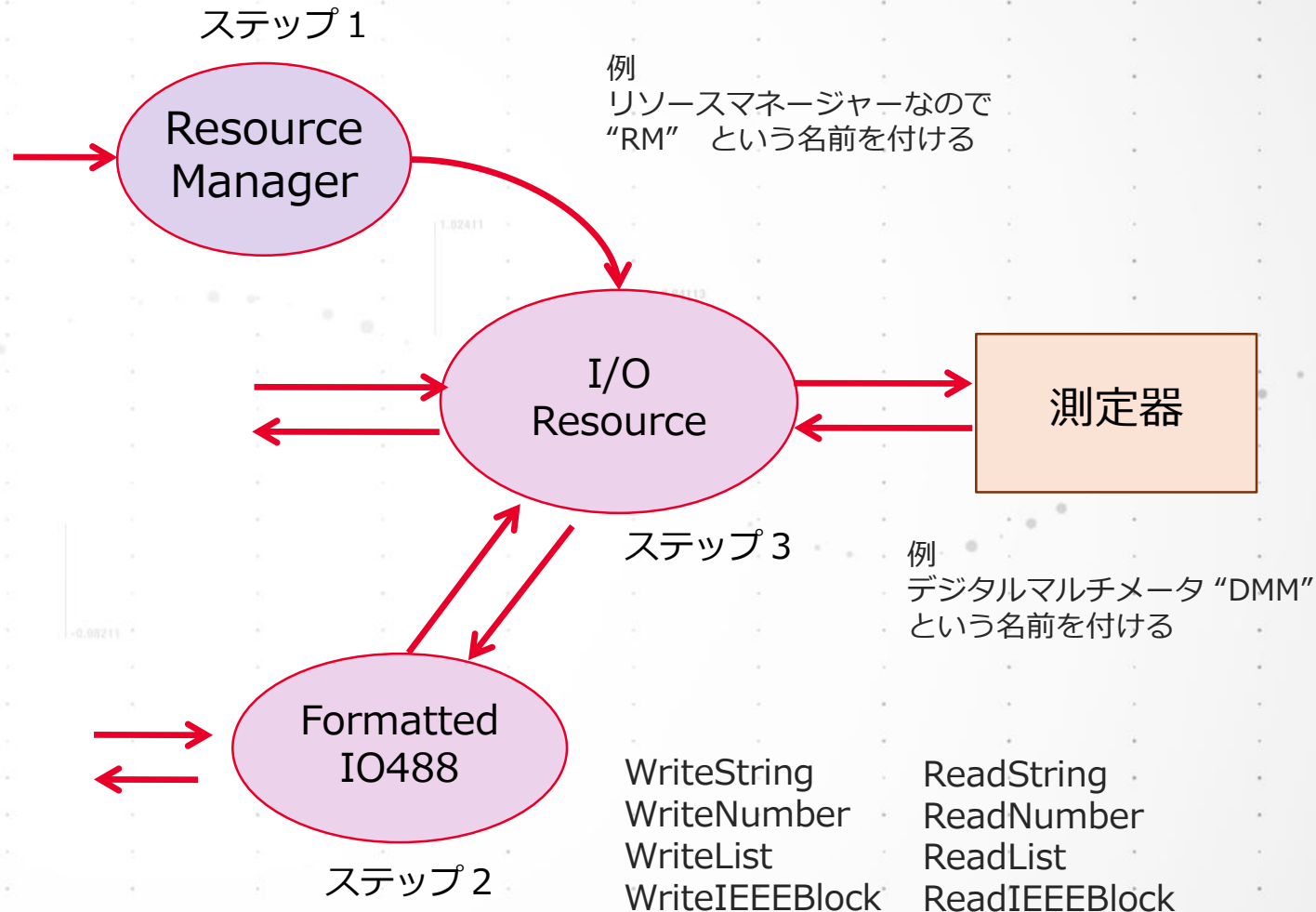
## プログラム中

```
DMM.IO.Timeout = 10000  
DMM.WriteString ("*IDN?")
```

## プログラムの最後

```
DMM.IO.Close()  
Set DMM = Nothing  
Set RM = Nothing
```

セッションの終了  
オブジェクトの開放



# Excel/VBA 簡単なサンプルプログラム

## DMMを制御する簡単なサンプルプログラム

- DMM を使用して、1秒ごとに10回、DC 電圧を測定し、結果をA列に保存するプログラム

```
Public Declare Sub Sleep Lib "KERNEL32.dll" (ByVal dwMilliseconds As Long)
Sub ボタン1_Click()
    Dim RM As New VisaComLib.ResourceManager
    Dim DMM As New VisaComLib.FormattedIO488
    Set DMM.IO = RM.Open("USBInstrument1")

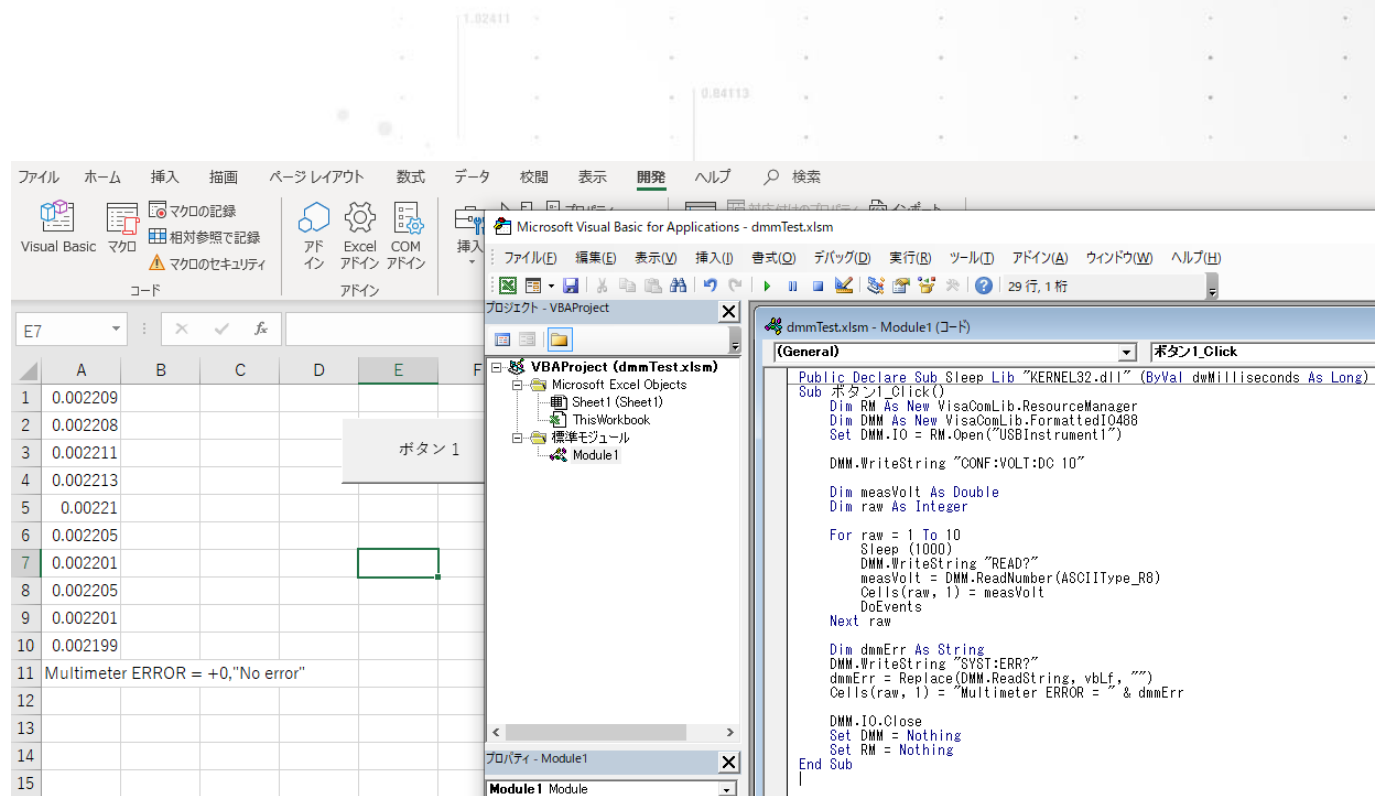
    DMM.WriteString "CONF:VOLT:DC 10"

    Dim measVolt As Double
    Dim raw As Integer

    For raw = 1 To 10
        Sleep (1000)
        DMM.WriteString "READ?"
        measVolt = DMM.ReadNumber(ASCIIType_R8)
        Cells(raw, 1) = measVolt
        DoEvents
    Next raw

    Dim dmmErr As String
    DMM.WriteString "SYST:ERR?"
    dmmErr = Replace(DMM.ReadString, vbLf, "")
    Cells(raw, 1) = "Multimeter ERROR = " & dmmErr

    DMM.IO.Close
    Set DMM = Nothing
    Set RM = Nothing
End Sub
```



# Excel/VBA 簡単なサンプルプログラム

## DC 電源を制御する簡単なサンプルプログラム

- DC 電源 を使用して、4.5Vから5.5Vまで、1秒ごとに 0.1V ステップで電圧を上昇させ、出力の設定電圧をA列、測定電圧をB列、測定電流をC列、に保存するプログラム

```
Public Declare Sub Sleep Lib "KERNEL32.dll" (ByVal dwMilliseconds As Long)
```

```
Sub ボタン1_Click()
```

```
Dim RM As New VisaComLib.ResourceManager  
Dim PS As New VisaComLib.FormattedIO488  
Set PS.IO = RM.Open("USBInstrument2")
```

```
PS.WriteString "VOLT 0"  
PS.WriteString "CURR 0.1"  
PS.WriteString "OUTPUT ON"
```

```
Dim outVolt As Double  
Dim measVolt As Double  
Dim measCurr As Double  
Dim raw As Integer
```

```
raw = 1  
For outVolt = 4.5 To 5.5 Step 0.1  
PS.WriteString "VOLT " & outVolt  
Sleep (1000)  
PS.WriteString "MEAS:VOLT?"  
measVolt = PS.ReadNumber(ASCIIType_R8)  
PS.WriteString "MEAS:CURR?"  
measCurr = PS.ReadNumber(ASCIIType_R8)  
Cells(raw, 1) = outVolt  
Cells(raw, 2) = measVolt  
Cells(raw, 3) = measCurr  
DoEvents  
raw = raw + 1  
Next outVolt
```

```
PS.WriteString "OUTPUT OFF"
```

```
Dim psErr As String  
PS.WriteString "SYST:ERR?"  
psErr = Replace(PS.ReadString, vbLf, "")  
Cells(raw, 1) = "Power supply ERROR = " & psErr
```

```
PS.IO.Close  
Set PS = Nothing  
Set RM = Nothing
```

```
End Sub
```

The screenshot displays the Microsoft Visual Basic for Applications environment. The left pane shows the VBAProject for psTest.xlsm, with the code for the ボタン1\_Click() subroutine. The right pane shows the code in the (General) tab. The background shows the Excel spreadsheet with the following data:

	A	B	C	D	E	F
1	4.5	4.50123	5.23E-05			
2	4.6	4.600846	4.96E-05			
3	4.7	4.701798	4.94E-05			
4	4.8	4.801414	5E-05			
5	4.9	4.901699	5.04E-05			
6	5	5.001315	5.05E-05			
7	5.1	5.10093	5.19E-05			
8	5.2	5.201883	5.24E-05			
9	5.3	5.301499	5.36E-05			
10	5.4	5.401114	5.46E-05			
11	5.5	5.501399	5.57E-05			
12	Power supply ERROR = +0,"No error"					
13						
14						
15						
16						
17						

# Excel/VBA 実用的なサンプルプログラム

## DC 電源と DMM を制御するサンプルプログラム

- DC 電源 を使用して、4.5Vから5.5Vまで、1秒ごとに 0.1V ステップで電圧を上昇させ、DC 電源 の設定電圧と測定電圧をA列とB列に、DMM での測定電圧をC列、に保存するプログラム

```
Public Declare Sub Sleep Lib "KERNEL32.dll" (ByVal dwMilliseconds As Long)
Sub ボタン1_Click()
    Dim RM As New VisaComLib.ResourceManager
    Dim DMM As New VisaComLib.FormattedIO488
    Dim PS As New VisaComLib.FormattedIO488
    Set DMM.IO = RM.Open("USBInstrument1")
    Set PS.IO = RM.Open("USBInstrument2")

    DMM.WriteString "CONF:VOLT:DC 10"

    PS.WriteString "VOLT 0"
    PS.WriteString "CURR 0.1"
    PS.WriteString "OUTPUT ON"

    Dim outVolt As Double
    Dim measV1 As Double
    Dim measV2 As Double
    Dim row As Integer
    row = 1
    For outVolt = 4.5 To 5.5 Step 0.1
        PS.WriteString "VOLT " & outVolt
        Sleep (1000)
        PS.WriteString "MEAS:VOLT?"
        measV1 = PS.ReadNumber(ASCIIType_R8)
        DMM.WriteString "READ?"
        measV2 = DMM.ReadNumber(ASCIIType_R8)
        Cells(row, 1) = outVolt
        Cells(row, 2) = measV1
        Cells(row, 3) = measV2
        DoEvents
        row = row + 1
    Next outVolt
    PS.WriteString "OUTPUT OFF"

    Dim psErr As String
    PS.WriteString "SYST:ERR?"
    psErr = Replace(PS.ReadString, vbLf, "")
    Cells(row, 1) = "Power supply ERROR = " & psErr
    row = row + 1
    Dim dmmErr As String
    DMM.WriteString "SYST:ERR?"
    dmmErr = Replace(DMM.ReadString, vbLf, "")
    Cells(row, 1) = "Multimeter ERROR = " & dmmErr

    PS.IO.Close
    DMM.IO.Close
    Set PS = Nothing
    Set DMM = Nothing
    Set RM = Nothing
End Sub
```

The screenshot shows the Microsoft Visual Basic for Applications environment. The VBA code editor on the right contains the same code as shown in the previous block. The Excel spreadsheet on the left displays the results of the program execution. The spreadsheet has columns A, B, and C, and rows 1 through 11. A button labeled 'ボタン1' is visible in cell D7.

	A	B	C	D	E	F
1	4.5	4.50123	2.330542			
2	4.6	4.600846	2.382139			
3	4.7	4.701798	2.4343			
4	4.8	4.801414	2.48584			
5	4.9	4.901699	2.537991			
6	5	5.001983	2.58954			
7	5.1	5.10093	2.641084			
8	5.2	5.201883	2.693219			
9	5.3	5.301499	2.744775			
10	5.4	5.401114	2.796316			
11	5.5	5.502068	2.848465			
12	Power supply ERROR = +0,"No error"					
13	Multimeter ERROR = +0,"No error"					
14						
15						
16						
17						
18						
19						
20						
21						
22						

# 参考資料

## 自動計測関連 アプリケーションノート

- IO Libraries Suite 2017 簡易取扱説明書

[https://www.keysight.com/upload/cmc\\_upload/All/IO\\_Lib\\_18\\_0\\_quick\\_JP\\_20170523.pdf](https://www.keysight.com/upload/cmc_upload/All/IO_Lib_18_0_quick_JP_20170523.pdf)

- 初心者向け Excel / VBA による測定器制御 プログラム 超入門編

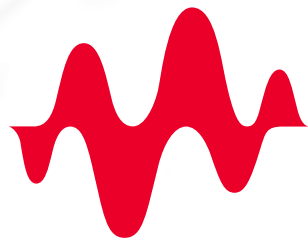
<https://www.keysight.com/jp/ja/assets/7018-04791/application-notes/5992-0691.pdf>

- Excel 2010/VBA による測定器制御プログラム入門

<https://www.keysight.com/jp/ja/assets/7018-07250/application-notes/5991-2363.pdf>

- Excel / VBA による測定器制御プログラム入門 中級編

<https://www.keysight.com/jp/ja/assets/7018-04790/application-notes/5992-0690.pdf>



**KEYSIGHT**  
TECHNOLOGIES

4.50221