Simple Measure

Version 1.1

スクリプトの使い方

Automatic Measurement Systems

URL: http://www.geocities.jp/automeasuresystem/

内容

概要

Simple Measure の C#スクリプトの具体的な使用方法を説明します。
C#スクリプトと C#マクロは実行形態が異なるだけで、機能面ではほぼ同じです。

スクリプトを使用することによって、統計処理などのデータの加工や、測定項目の繰り返し処理、複数の測定器を使用し、校正しながら測定するなどの操作が行えるようになり、より幅広い用途への使用が可能となります。
また、DAQmx などの独自のインターフェースに対応することも可能になります。

サンプルスクリプト

実用的なスクリプトをいくつか作成しました。基本的な処理のみ行う内容です。用途によって使いやすくなるように改良して使用してください。

スクリプトのソースコードは、Measure シートの Parameter フィールドに設定しますが、そこに直接記述しないようにしてください。Script シートなどを作成して、そこに記述し参照するようにします。複数の箇所に同じスクリプトを記述すると後からの修正が困難になります。
このドキュメントでは説明のために直接記述してあります。該当の箇所を青の網掛けで表示しています。

スクリプトの引数は Measure シートの Parameter2 に記述します。その属性が、数値になっている場合、エラーが発生する場合があります。そのような時は、Parameter2 の値を文字列になるようにセルの属性を修正してください。

スクリプトの仕様や使用できるオブジェクトなどは「Simple Measure 取扱説明書」を参照してください。

Excel のリボンに[開発] タブを表示する

初期状態では表示されていませんので、次の操作を行って表示するように設定してください。

Excel 2016
[ファイル]［オプション］[リボンのユーザー設定］を開き、［メイン タブ］の下の［開発］チェック ボックスをオンにします。

Excel 2007
[オプション］[基本設定]で「 Excel の使用に関する基本オプション」の枠の「[開発]タブをリボンに表示する」にチェックを入れます。

Jump

機能

Measure シートの指定された行にジャンプします。

引数（Parameter2）

ジャンプ先の行番号

スクリプト

```
// Jump
// 引数: ジャンプ先の行番号
// Parameter に行番号を指定する
// MeasureId から行番号を取得するには Excel の Math()関数を使用する
using System;
namespace SimpleMeasure
{
    public partial class MeasureScript : MeasureScriptBase
    {
        public override object[] Main(string value)
        {
            // Status.Index は 1 から始まるインデックスで、ここで設定した値は
            // この処理が終わったとき、インクリメントするので、
            // 戻る位置 - 1 を Status.Index に設定する。
            // 0 を設定した場合、最初の行に移動する。
            // Config[MeasureLibConfig.Measure_Start_Index] には、
            // デフォルトで 2 が設定されている。
            int offset = (int)Config[MeasureLibConfig.Measure_Start_Index];
            Status.Index = int.Parse(value.ToString()) - offset;
            Console.WriteLine("Jump {0}", Status.Index);
            return null;
        }
    }
}
```

使用例

次の例では、4 行目から 3 行目、MeasureID = 2 にジャンプします。

	A	B	C	D	E	F	G	H
1	Measure Id	Label	Method	Device	Parameter	Parameter2	Report Id	Description
2	1	●ジャンプ	;	;				無限ループ
3	2		Write	Wait	1000			
4	3		Script	Dummy	// Jump	3		MeasureId=2 に戻る"
5	4	終了	Write	End				

このサンプルでは、ここにスクリプトコードを貼り付けます。
本番では他のシートに記述したものを参照するようにしてください。

Loop

機能

Measure シートの指定された行に指定回数ジャンプします。

引数（Parameter2）

ジャンプ先行番号

ループカウンタ変数名

繰り返し数

0 以上の整数

スクリプト

```
// Loop
// 引数: ジャンプ先の行番号, 変数名, ループ回数
using System;
namespace SimpleMeasure
{
    public partial class MeasureScript : MeasureScriptBase
    {
        public override object[] Main(string value)
        {
            // Status.Index は 1 から始まるインデックスで、ここで設定した値は
            // この処理を終わったとき、インクリメントするので、
            // 戻る位置 - 1 を Status.Index に設定する。
            // 0 を設定した場合、最初の行に移動する。

            string[] param = value.ToString().Split(',');
            string v = param[1].Trim();
            // 変数が設定されていなかった場合は、定義する
            if (!Var.ContainsKey(v)) { Var.Add(v, 0); }
             Console.WriteLine("Looop Counter i = " + (int)Var[v]);

            Var[v] = (int)Var[v] + 1;
            // 指定ループ回数に達しているか?
            if ((int)Var[v] < int.Parse(param[2]))
            {
                // 内部のオフセット値を取得
                int offset = (int)Config[MeasureLibConfig.Measure_Start_Index];
                // オフセット値を除いた値を Status.Index に設定
                Status.Index = int.Parse(param[0]) - offset;
            }
            // Console.WriteLine("Index = {0}", Status.Index);
            return null;
        }
    }
}
```

使用例

次の例では、6 行目から 4 行目、MeasureID = 3 に 10 回ループします。使用する変数は「LoopCounter」です。3 行目のスクリプトについては「int 変数の初期化」を参照してください。

	A	B	C	D	E	F	G	H
1	Measure Id	Label	Method	Device	Parameter	Parameter2	Report Id	Description
2	1	●ループ	;	;				変数を登録し、指定回数繰り返す
3	2		Script	Dummy	// int 変数の初期化	LoopCounter		変数の初期化
4	3		Write	Wait	1000			
5	4		Read	Var	LoopCounter			
6	5		Script	Dummy	// Loop	4, LoopCounter, 10		MeasureId=3 へ戻る
7	6	終了	Write	End				

Int 変数の初期化

機能

Int 変数を登録して 0 に初期化します。すでに登録されていた場合は、値を 0 に設定します。

引数（Parameter2）

変数名[, 変数名, ・・・]

スクリプト

```
// int 変数の初期化
// 引数: 変数名リスト
using System;
namespace SimpleMeasure
{
    public partial class MeasureScript : MeasureScriptBase
    {
        public override object[] Main(string value)
        {
            string[] param = value.ToString().Split(',');
            foreach (object o in param)
            {
                string v = o.ToString().Trim();
                Console.WriteLine("Variable Reset: " + v);
                if (!Var.ContainsKey(v))
                {
                    Var.Add(v, 0);
                }
                else
                {
                    Var[v] = 0;
                }
            }
            return null;
        }
    }
}
```

使用例

次の例では、2 行目で、int 変数の i, j, k を登録し、0 で初期化します。4～6 行目で変数の値を Data シートに読み出しています。

	A	B	C	D	E	F	G	H
1	Measure Id	Label	Method	Device	Parameter	Parameter2	Report Id	Description
2	1	●int 変数の初期化	;	;				
3	2		Script	Dummy	// int 変数の初期化	i, j, k		変数の初期化
4	3		Read	Var	i			
5	4		Read	Var	j			
6	5		Read	Var	k			
7	6	終了	Write	End				

無限ループ測定

機能

直前に測定されたデータ、LastData を、指定されたシートに保存します。そして、指定された行にジャンプし繰り返し測定を行います

カンマ区切りのデータは "C" 以降のフィールドに分割され記録されます。

保存されるデータの書式は次のようになります。

インデックス, 測定日時, 測定値[,測定値・・・]

引数（Parameter2）

ジャンプ先の行番号, データを保存するシート名

スクリプト

```
// 無限ループ測定
// 引数: ジャンプ先の行番号, 書き込むテーブル
// 引数で指定されたテーブルの 2 行目以降にデータを書き込む
using System;
namespace SimpleMeasure
{
    public partial class MeasureScript : MeasureScriptBase
    {
        private static int CellId = 1;  // 書き込み先のセル・インデックス
        public override object[] Main(string value)
        {
            // データを書き込むフィールド
            string[] fld = new string[]{"C", "D", "E", "F", "G", "H", "I", "J", "K", "L"};
            string[] param = value.Split(',');

            // 書き込み先の Worksheet オブジェクトを作成
            dynamic sheet = ExcelBook.Worksheets[param[1].Trim()];
            // セルに書き込む
            sheet.Range["A" + (++CellId).ToString()].Value = (CellId - 1).ToString();
            sheet.Range["B" + CellId.ToString()].Value
                = DateTime.Now.ToString("yyyy/MM/dd HH:mm:ss.ff");

            string[] dat = LastData.Split(',');
            // データの書き込み
            for (int i = 0; i < dat.Length; ++i)
            {
                sheet.Range[fld[i] + CellId.ToString()].Value = dat[i];
            }
            Console.WriteLine("Id = {0}, {1}", CellId - 1, LastData);

            // オフセット値を除いた値を Status.Index に設定
            int offset = (int)Config[MeasureLibConfig.Measure_Start_Index];
            Status.Index = int.Parse(param[0]) - offset;

            return null;
        }
    }
}
```

使用例

次の例では、7 行目から 5 行目にジャンプし、測定を繰り返します。測定データは「LoopData」シートに保存されます。

3 行目の Parameter は説明のために省略してあります。実際には測定器の初期化コマンドを記述してください。5 行目で測定値を読み取っています。

·	A	B	C	D	E	F	G	H
1	Measure Id	Label	Method	Device	Parameter	Parameter2	Report Id	Description
2	1	●無限ループ測定	;	;				
3	2	初期化	Write	Oscillo	*RST; 初期化コマンド			測定器の初期化
4	3	初期化	Write	Wait	3000			初期化後の待ち時間
5	4	測定	ReadString	Oscillo	:Measure:Meas1:Value?			測定データの読み取り
6	5	測定	Write	Wait	1000			待ち時間
7	6	測定	Script	Dummy	// 無限ループ測定	5,LoopData		
8	7	終了	Write	End				

DateTime 付 Read

機能

データを読み取るときに、年月日と時間をデータに追加します。Method の Read の代わりに使用します。

引数（Parameter2）

[リモートコマンド]（省略可能）

スクリプト

```
// DateTime 付 Read
// 引数: リモートコマンド
using System;
namespace SimpleMeasure
{
    public partial class MeasureScript : MeasureScriptBase
    {
        public override object[] Main(string value)
        {
            if (value != "")
            {
                DevObj.CurrentDevice.Write(value);
                System.Threading.Thread.Sleep(100);
            }
            string[] dat = DevObj.CurrentDevice.ReadString().Trim().Split(',');
            object[] result = new object[dat.Length + 1];

            // 戻り値の作成
            result[0] = DateTime.Now.ToString("yyyy/MM/dd HH:mm:ss.ff");
            LastData = result[0].ToString();
            for (int i = 0; i < dat.Length; ++i)
            {
                result[i + 1] = dat[i].Trim();
                LastData += "," + result[i + 1];
            }
            Console.WriteLine(LastData);
            return result;
        }
    }
}
```

使用例

次の例では、5 行目で測定値を読み取り、DateTime をデータに追加します。読み取る前にリモートコマンド”:Measure:Meas1:Value?”を Oscillo に送っています。3 行目の Parameter は説明のために省略してあります。実際には測定器の初期化コマンドを記述してください。

データが保存される、Data シートの B フィールドはセルの書式設定で、「年月日 時分秒」に設定してください。書式が標準の場合、正しく表示されません。

	A	B	C	D	E	F	G	H
1	Measure Id	Label	Method	Device	Parameter	Parameter2	Report Id	Description
2	1	● DateTime 付 Read	;	;				
3	2	初期化	Write	Oscillo	*RST;:初期化コマンド			測定器の初期化
4	3	初期化	Write	Wait	3000			初期化後の待ち時間
5	4	測定	Script	Oscillo	// DateTime 付 Read	:Measure:Meas1:Value?		測定
6	5	終了	Write	End				

DateTime 付 ReadString

機能

データを読み取るときに、年月日と時間をデータに追加します。Method の ReadString の代わりに使用します。

引数（Parameter2）

リモートコマンド。省略可能。

スクリプト

```
// DateTime 付 ReadString
// 引数: リモートコマンド
using System;
namespace SimpleMeasure
{
    public partial class MeasureScript : MeasureScriptBase
    {
        public override object[] Main(string value)
        {
            if (value != "")
            {
                DevObj.CurrentDevice.Write(value);
                System.Threading.Thread.Sleep(100);
            }
            string dat = DevObj.CurrentDevice.ReadString().Trim();
            object[] result = new object[2];

            // 戻り値の作成
            result[0] = DateTime.Now.ToString("yyyy/MM/dd HH:mm:ss.ff");
            result[1] = dat;
            LastData = result[0].ToString() + "," + result[1];
            Console.WriteLine(LastData);
            return result;
        }
    }
}
```

使用例

次の例では、5 行目で測定値を読み取り、DateTime をデータに追加します。読み取る前にリモートコマンド”:Measure:Meas1:Value?”を Oscillo に送っています。3 行目の Parameter は説明のために省略してあります。実際には測定器の初期化コマンドを記述してください。

データが保存される、Data シートの B フィールドはセルの書式設定で、「年月日 時分秒」に設定してください。書式が標準の場合、正しく表示されません。

	A	B	C	D	E	F	G	H
1	Measure Id	Label	Method	Device	Parameter	Parameter2	Report Id	Description
2	1	● DateTime 付 ReadString	;	;				
3	2	初期化	Write	Oscillo	*RST;:初期化コマンド			測定器の初期化
4	3	初期化	Write	Wait	3000			初期化後の待ち時間
5	4	測定	Script	Oscillo	// DateTime 付 ReadString	:Measure:Meas1:Value?		測定
6	5	終了	Write	End				

パラメータ付き Read

機能

データを読み取るときに、引数に指定した値も Data シートに記録します。Method の Read の代わりに使用します。

引数（Parameter2）

追加するパラメータの数、パラメータ[, パラメータ]*、[リモートコマンド]

スクリプト

```
// パラメータ付き Read
// 引数:  引数の数, パラメータ[, パラメータ]*, [リモートコマンド]
using System;
namespace SimpleMeasure
{
    public partial class MeasureScript : MeasureScriptBase
    {
        public override object[] Main(string value)
        {
           string[] cnt = value.Split(new char[] { ',' }, 2);
           string[] arg = null;
           int arglen = int.Parse(cnt[0]);
           if (1 < cnt.Length)
           {
                arg = cnt[1].Split(new char[] { ',' }, arglen + 1);
           }

           if (arg != null && arglen < arg.Length)
           {
                DevObj.CurrentDevice.Write(arg[arglen]);
                System.Threading.Thread.Sleep(100);
           }
           string[] dat = DevObj.CurrentDevice.ReadString().Trim().Split(',');
           object[] result = new object[arglen + dat.Length];

           // 戻り値の作成
           LastData = "";
           if (0 < arglen)
           {
                result[0] = arg[0].Trim();
                LastData = result[0].ToString();
           }
           for (int i = 1; i < arglen; ++i)
           {
                result[i] = arg[i].Trim();
                LastData = "," + result[i].ToString();
           }
           for (int i = 0; i < dat.Length; ++i)
           {
                result[i + arglen] = dat[i].Trim();
                LastData += "," + result[i + arglen];
           }
           Console.WriteLine(LastData);
           return result;
        }
    }
```

}

使用例

次の例では、5 行目で測定値を読み取り、Parameter2 で指定されている “abc”をデータに追加します。読み取る前にリモートコマンド” :Measure:Meas1:Value?”を Oscillo に送っています。3 行目の Parameter は説明のために省略してあります。実際には測定器の初期化コマンドを記述してください。

	A	B	C	D	E	F	G	H
1	Measure Id	Label	Method	Device	Parameter	Parameter2	Report Id	Description
2	1	● パラメータ付 Read	;	;				
3	2	初期化	Write	Oscillo	*RST;: 初期化コマンド			
4	3	初期化	Write	Wait	3000			
5	4	測定	Script	Oscillo	// パラメータ付き Read	1, abc, :Measure:Meas1:Value?		
6	5	終了	Write	End				

バイナリーで読み書き

機能

バイナリーデータの読み書きを行います。

このサンプルでは文字列をバイト配列に変換して書き込み、バイト配列で読み取ったデータを文字列に変換します。ReadByte(int count)の count の値は 100 に設定しています。読み取るデータが長い場合はこの値を大きくしてください。

引数（Parameter2）

Binary Write: 問い合わせコマンド。

Binary Read: なし。

スクリプト

```
// Binary Write
// 引数:  問い合わせコマンド
using System;
namespace SimpleMeasure
{
    public partial class MeasureScript : MeasureScriptBase
    {
        public override object[] Main(string value)
        {
            Console.WriteLine("Parameter=" + value);

            // 文字列をバイト配列に変換
            Console.WriteLine("書き込むバイナリーデータ");
            byte[] cmd = new byte[value.Length];
            for (int i = 0; i < value.Length; i++)
            {
                cmd[i] = (byte)((int)value[i]);
                Console.Write("[{0}]", ((int)cmd[i]).ToString("x2"));
            }
            Console.WriteLine();

            // バイナリーで書き込み
            DevObj.CurrentDevice.Write(cmd);
            return null;
        }
    }
}

// Binary Read
// 引数:  なし
using System;
namespace SimpleMeasure
{
    public partial class MeasureScript : MeasureScriptBase
    {
        public override object[] Main(string value)
        {
            Console.WriteLine("読み取ったバイナリーデータ");

            // バイナリーで読み取り
            byte[] dat = DevObj.CurrentDevice.ReadByte(100);
```

```
            // バイト配列を文字列に変換
            string result = "";
            for (int i = 0; i < dat.Length; i++)
            {
                result += ((char)dat[i]).ToString();
                Console.Write("[{0}]", ((int)dat[i]).ToString("x2"));
            }
            Console.WriteLine();
            Console.WriteLine(result);
            return new object[]{result};
        }
    }
}
```

使用例

次の例では、3 行目の Parameter2 で指定されている “?Frq”を Osc にバイナリーで書き込みます。4 行目で Osc からバイナリーで読み取り、文字列に変換します。

	A	B	C	D	E	F	G	H
1	Measure Id	Label	Method	Device	Parameter	Parameter2	Report Id	Description
2	1	●バイナリーで読み書き	;	;				
3	2	Binary Write	Script	Osc	// Binary Write	?Frq		
4	3	Binary Read	Script	Osc	// Binary Read			
5	4	終了	Write	End				

DAQmx を使用する

機能

DAQmx デバイスの NI USB-6501 にデータを書き込みます。
出力は内部の 4.7kΩでプルアップされたオープンコレクタに設定されます。
このサンプルでは DAQmx .NET API を使用していますが、NI I/O Trace ではサポートされていません。
送受信 データのキャプチャは行えません。
参考
http://zone.ni.com/reference/ja-XX/help/370466AD-0112/daqhelp/troubleshooting/

引数（Parameter2）

出力するビット番号。Port0 の 0-7 と Port1 の 0-7 まで 16 ビットを連続で 0-15 を指定。

参照設定の追加

Excel VBA モジュール SimpleMeasure のメソッド Private Sub SetConfig()内で、asm(10)、asm(11)への代入を追加してください。（すでに 10,11 を使用している場合は、別のインデックスにしてもかまいません。）
このライブラリは、DAQmx のドライバをインストールするときに、「アプリケーション開発サポート」で「.NET Framework 言語サポート」を指定しないとインストールされません。また、インストールされたライブラリの場所は、デフォルトでの設定です。変更した場合はその場所を指定してください。
次のコードではファイルパスの途中で改行が入っていますが、ソースコードでは改行を入れないでください。

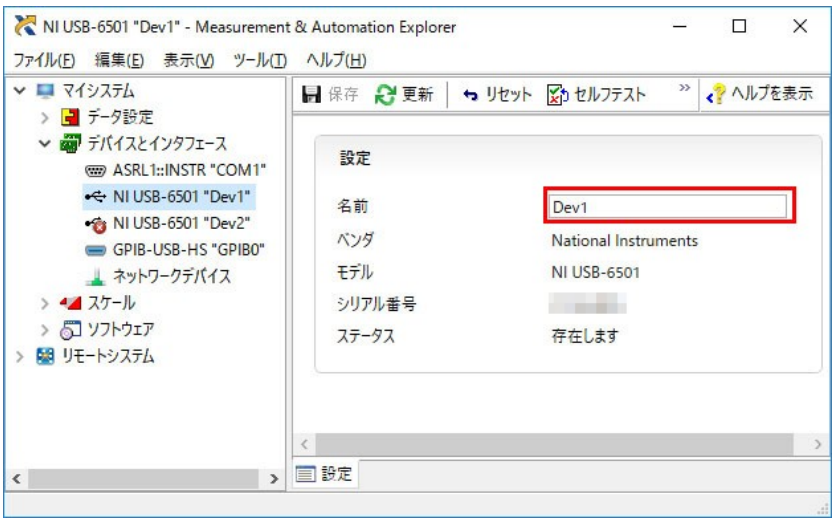
モジュール：SimpleMeasure

```
Private Sub SetConfig()
... 中略

    'Script で使用するアセンブリリスト
    'SimpleMeasure.dll はこのリストに指定する必要なし。
    Dim asm(15) As Variant
    asm(0) = "System.dll"
    asm(1) = "System.Core.dll"
    asm(2) = "System.Data.dll"
    asm(3) = "System.Data.DataSetExtensions.dll"
    asm(4) = "System.Drawing.dll"
    asm(5) = "System.Windows.dll"
    asm(6) = "System.Windows.Forms.dll"
    asm(7) = "System.Xml.dll"
    asm(8) = "System.Xml.Linq.dll"
    asm(9) = "Microsoft.CSharp.dll"
    asm(10) = "C:¥Program Files (x86)¥National Instruments¥
MeasurementStudioVS2012¥DotNET¥Assemblies¥Current¥NationalInstruments.Common.dll"
    asm(11) = "C:¥Program Files (x86)¥National Instruments¥
MeasurementStudioVS2012¥DotNET¥Assemblies¥Current¥NationalInstruments.DAQmx.dll"
    MesObj.AddConfig "Script.References", asm
... 中略
```

DAQmx のデバイス名の確認

NI MAX を起動し、デバイスとインターフェースを展開すると次の図のように名前のところにデバイス名（Dev1）が表示されます。編集して名前を変更することもできます。



スクリプト

DAQmx デバイスの指定はデバイス名を使用します。次のコードでは ”Dev1” となります。

```
// DAQmx Test
using System;
using NationalInstruments.DAQmx;
namespace SimpleMeasure
{
    public partial class MeasureScript : MeasureScriptBase
    {
        public override object[] Main(string value)
        {
            string dev = "Dev1";
            Task tsk = new Task();
            DOChannel ch = tsk.DOChannels.CreateChannel(dev + "/Port0:1", "",
                ChannelLineGrouping.OneChannelForAllLines);
            // ch.OutputDriveType = DOOutputDriveType.ActiveDrive;  // Active Drive
            DigitalSingleChannelWriter writer = new DigitalSingleChannelWriter(tsk.Stream);
            writer.WriteSingleSamplePort(true, (UInt16)Math.Pow(2, int.Parse(value)));
            return null;
        }
    }
}
```

使用例

次の例では、NI USB-6501 のポート 0 の 0 から 3 ビットへ、順にひとつずつ HI レベルに設定します。設定するビットは Parameter2 で設定します。

·	A	B	C	D	E	F	G	H
1	Measure Id	Label	Method	Device	Parameter	Parameter2	Report Id	Description
2	1	DAQmx Test	Script	Dummy	// DAQmx Test	0		Port0 のビット 0 に書込み
3	2	DAQmx Test	Write	Wait	1000			1 秒待つ
4	3	DAQmx Test	Script	Dummy	// DAQmx Test	1		Port のビット 1 に書込み
5	4	DAQmx Test	Write	Wait	1000			1 秒待つ
6	5	DAQmx Test	Script	Dummy	// DAQmx Test	2		Port0 のビット 2 に書込み
7	6	DAQmx Test	Write	Wait	1000			1 秒待つ
8	7	DAQmx Test	Script	Dummy	// DAQmx Test	3		Port0 のビット 3 に書込み
9	8	終了	Write	End				