[APNOTE08]

リモート XBee デバイス計測値を定期的に DB に保管、クライアント PC のエクセルから集計(イベント方式)

ABS-9000 DeviceServer
APNOTE08 Rev A.1.1
2010/03/31

オールブルーシステム (All Blue System)
ウェブページ: www.allbluesystem.com
コンタクト:contact@allbluesystem.com

1　　イントロダクション

複数のリモート XBee[1] デバイスから任意のタイミングで送信される I/O データフレーム(A/D 変換値, DIO 値)を受信して、サーバーのデータベースに保管します。データベースの保管されたデータを、随時クライアントPCから集計するシステムについて説明します。

XBee デバイスは、Digi International Inc. 社製の IEEE 802.15.4 RF モジュールを使用します。計測用のリモート XBee モジュールには、外付けのマイクロコントローラ等を使用せず、XBee 自身が持つ A/D 変換機能と DIO 機能のみを使用します。

DeviceServer の COM ポートに接続された XBee デバイスを経由して、複数の XBee デバイスから送信される I/O データフレーム(以下、I/O データ)を受信します。受信した I/O データは、DeviceServer 中のデータベースにデバイス毎にタイムスタンプを付けて保管します。データベースは DeviceServerに組み込まれた Firebird DBMS を使用します。データ保管用のデータベースとして外部の Oracle 10g サーバーを使用することもできます。

クライアントPC のエクセルから、デバイス名とI/O データの収集範囲を指定して、データベースに保管されたデータを取得し、集計やグラフの作成等を行うことができます。

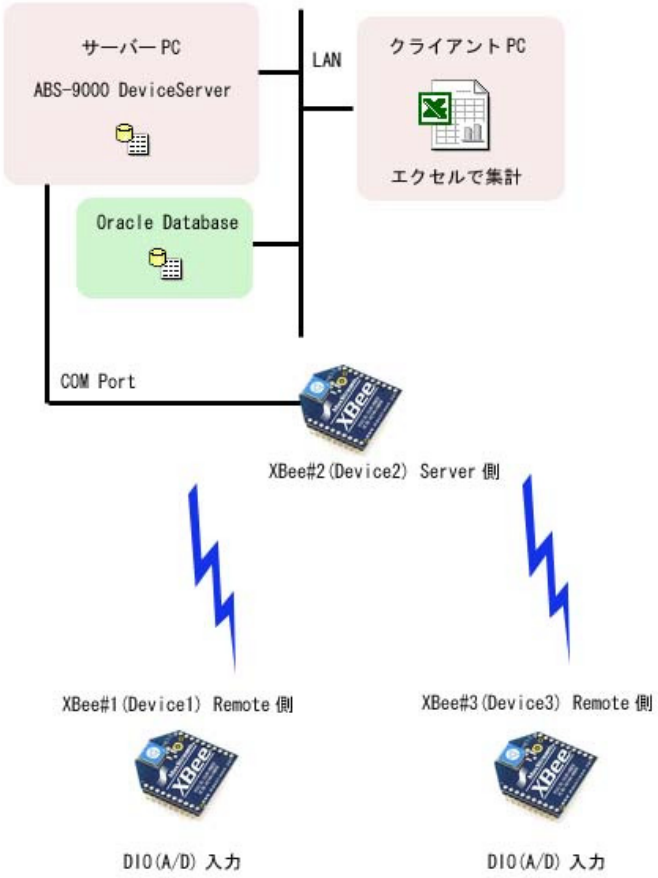
2　　必要な機材・リソース

必要なシステムやデバイス等	説明
ABS-9000 DeviceServerの動作しているPC	DeviceServer の動作する PC が必要になります。
ABS-9000 DeviceServerの動作しているPC の Windows OS	タスクスケジュールコマンド(schtasks) を使用するため、DeviceServer の動作している OS が WindowsXP もしくは Windows2003 である必要があります。
XBee デバイス	Digi International Inc. 社製 XBee IEEE 802.15.4 デバイスが 2台以上必要です。DeviceServer のCOM ポート直接接続する XBee デバイスが１台、その他はリモート計測側の XBee デバイスになります。 DeviceServer は、XBee デバイスのファームウエアバージョンの “10CD” にのみ対応しています。(必要に応じて XBee ファームウエアの更新を行ってください)
Oracle10g サーバー（オプション）	DeviceServer の動作するPC もしくは 別 PC にOracle10g サーバーと、DeviceServer の動作するPCに Oracleクライアントが必要です。 (保管対象のデータベースをFirebird からOracleに変更する時だけ、必要に

[1] XBee XBee® and XBee□PRO® are registered trademarks of Digi, Inc.

	なります) 詳細は “DeviceServer ユーザーマニュアル”を参照してください。

3 システム構成図



4 システム動作概要

- リモートXBee デバイス側のI/O データ(DIO,A/D)の送信先を、DeviceServer のCOMに接続された XBee デバイスアドレスに設定します。リモート XBee デバイスでは、予め設定しておいたサンプリングレートまたは Change Detect イベント毎に、I/O データパケットが DeviceServerに接続された XBee デバイスへ送信されます。
- DeviceServer に接続された XBee デバイスでI/O データを受信すると、XBEE_IO_DATA イベントハンドラが DeviceServer で実行されます。XBEE_IO_DATA イベントハンドラ中で、DIO 入力値と A/D 変換値を取り込んで、データベースに保管します。I/O データはリモート XBee デバイス毎にタイムスタンプ情報と共に保管されます。
- リモートクライアントPCもしくはサーバーPC のエクセルから、DeviceServer 経由でデータベースに保管された計測値を取り込みます。集計対象デバイスと収集範囲を指定して、計測データをエクセルのワークシートの

セルにロードして、集計作業やグラフ作成などを行います。

5 設定手順

5.1 XBee デバイス初期設定

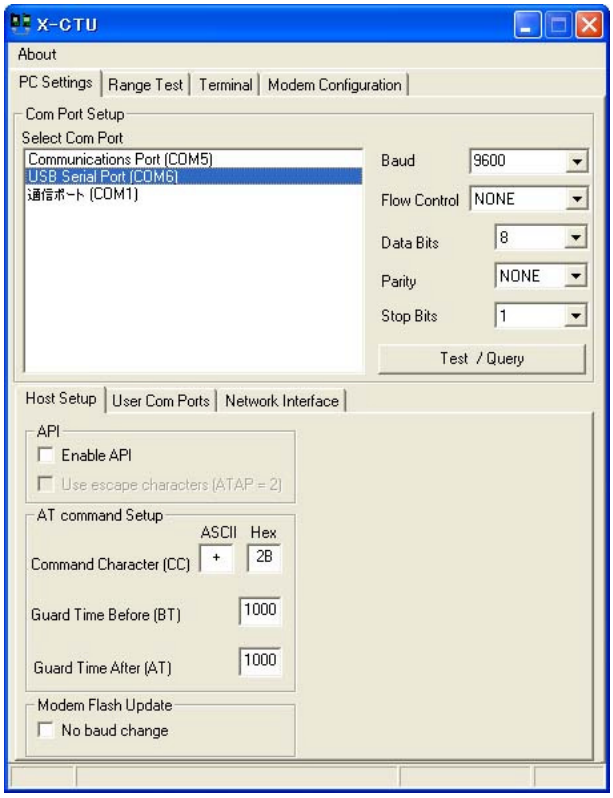
XBee デバイスを DeviceServer に接続するために、初期設定が必要です。DeviceServer の COM ポートに接続する XBee デバイスとリモート計測側の XBee デバイスで下記の設定を行ってください。

最初に DeviceServerとの接続に必要な最低限の設定を COM ポート経由で行います。Sparkfun Electoronics 社製の XBee Explorer USB 等を使用して、仮想 USB ポート経由で接続して設定してください。(これ以外の方法で COM ポート接続する場合も手順は同じです)

XBee デバイス デフォルト値から変更が必要な設定値	
API モード	1 (default は 0)
PAN(Personal Area Network) ID	任意の値 (default は0x3332) デフォルトの値のままだと、予期しないデバイスからのフレームを受信したり、間違ってデバイスを操作する恐れがありますので、適当な任意の値を設定するようにしてください。このマニュアルでは 0xAB90 を使用しています。
16bit Source Address	同一PAN ID 内でユニークな値 (default は 0x0000) この値は、ここで設定しなくても後から XBee 管理プログラムで設定することが可能ですが、デバイス一覧から選択したデバイスがどのデバイスであるかを見分けることが容易になるように便宜的にここで設定します。 全てのデバイス間で違った値を設定してください。 (0x0000,0xFFFF,0xFFFE を除く) 例えば、0x0001, 0x0002,0x0003 等。

上記３つの 初期設定のコマンドをXBee に送信するために、XBee デバイスを PC のCOM ポートに接続して Digi international Inc. 社製の X-CTU プログラム、または汎用のターミナルエミュレータプログラム等を使用します。XBee とCOM ポートのボーレートは初期設定の 9600 bps にして下さい。(ターミナルエミュレータを使用する場合は、ローカルエコー ON, 受信時の改行 CR + LF にするとコマンド実行の結果が見やすくなります)

X-CTU プログラムを起動して、COM ポートを選択します。ここでは、USB Serial Port(COM6) を選択しています。

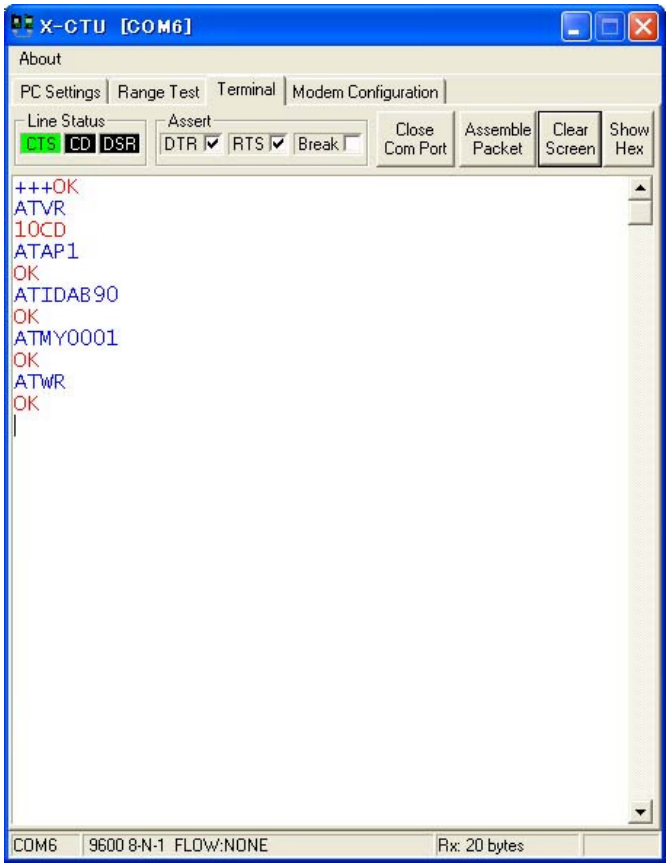


Terminal タブを選択してターミナル画面を表示します。キーボードから、"+++" を入力して、コマンドモードに入ります。コマンドモードに入ると "OK" が表示されますので、続けて以下のコマンド文字列を入力してください。コマンド入力の時間がかかりすぎると、自動的にコマンドモードから抜けてしまいますので、その場合は、"+++" を入力して最初からコマンドを入力し直して下さい。

```
ATVR
ATAP1
ATIDAB90
ATMY0001
ATWR
```

最初に ATVR でファームウエアバージョンを表示しています。"10CD" 以降になっていることを確認してください。ATAP1 は、API モードを "1" に設定しています。ATIDAB90 は PAN_ID を 0xAB90 に設定しています。もし別の PAN_ID を使用する場合は適宜変更してください。次に、ATMY0001 で、デバイスの16 bit Source Address を "0x0001" に設定しています。この部分は、デバイスごとにユニークな値になるように変更して下さい。

最後に、ATWR で、設定値を不揮発メモリに書き込みます。実際に入力した時の画面表示は以下の様になります。



X-CTU プログラムを終了します。 その後、XBee Explorer USB に接続する XBee デバイスを切り替えて、 使用する全ての XBee デバイスについて同様に初期設定を行って下さい。

このときに、設定した16bit Source Address の値をデバイス機器にマーキングしておくと、後で XBee デバイス管理プログラムでデバイスを選択するときに、識別し易くなります。

システム構成図上の３つのデバイスを接続する場合の設定値例は、以下の様になります。

デバイス	API モード(ATコマンド)	PAN_ID(ATコマンド)	16bit Address(ATコマンド)
XBee#1(リモート)	1(ATAP1)	0xAB90(ATIDAB90)	0x0A01(ATMY0A01)
XBee#2(サーバー)	1(ATAP1)	0xAB90(ATIDAB90)	0x0B02(ATMY0B02)
XBee#3(リモート)	1(ATAP1)	0xAB90(ATIDAB90)	0x0C03(ATMY0C03)

5.2 XBee デバイスを DeviceServer に接続

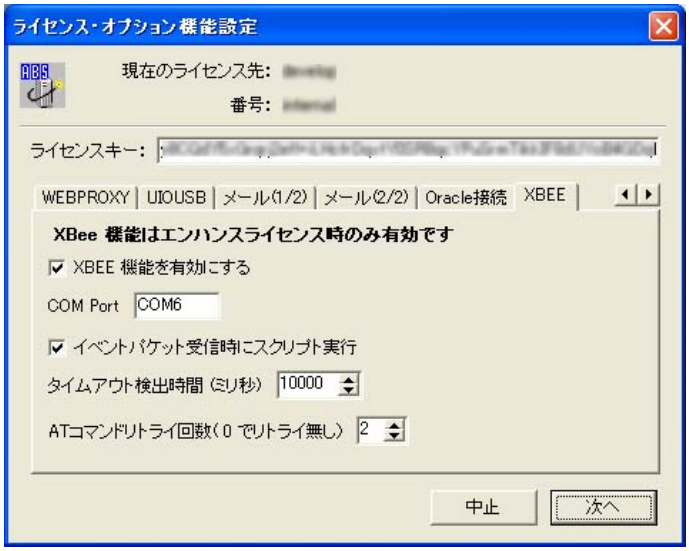
XBee デバイスの初期設定後に、XBee#2(サーバー) のみを PC (DeviceServer) の COM ポートに接続します。その他のリモート XBee デバイスも電源を入れて通信可能な状態にしておきます。

デバイスが PC に接続されたら、DeviceServer から XBee デバイスを使用可能にするために COM ポートの設定を行います。サーバー設定プログラム(ServerInit.exe)を起動して、XBEE タブを選択して COM ポート番号を設定して “XBEE 機能を有効にする” にチェックをつけてください。

また、I/O データ受信時に XBEE_IO_DATA イベントハンドラを実行するために、”イベントパケット受信時にスクリ

プト実行” にもチェックを付けてください。

サーバー設定プログラムの “次へ” を押して”完了”ボタンが表示されるまで進めて設定を完了して下さい。



5.3 マスター登録と XBee 詳細設定

XBee デバイスを DeviceServer のマスターファイルに登録します。XBee デバイスの初期設定で設定しなかった NodeIdentifier と DIO、サンプリングレート等の詳細設定もここで行います。

システム構成図上の３つのデバイスを接続する場合の設定値例は、以下の様になります。

デバイス 16bitAddress	Node Identifier	Sample Before TX	SampleRate (ms)	DIO	ADC	Destination Address
0x0A01(リモート)	Device1	1	5000(*1)	(*2)	(*2)	(*3)
0x0B02(サーバー)	Device2	1	0	Disabled	Disabled	0x0
0x0C03(リモート)	Device3	1	10000(*1)	(*2)	(*2)	(*3)

(*1)各々のリモート XBee デバイスで、5秒と10秒毎にI/O データ送信を行う場合

(*2)リモートXBee に接続する H/W に合わせて、DIO 設定を “DI” や “ADC”, “Disabled” 等に設定します。

(*3)リモートXBee デバイスの Destination Address(High/Low) は サーバー側 XBee デバイスのシリアル番号(64bitアドレス)もしくは、16bit アドレスを指定します。

XBee デバイス管理プログラムを使用して、同一 PAN ID をもつ XBee デバイスを DeviceServer に登録したり、デバイス自身の設定内容を変更します。プログラムメニューから “ALL BLUE SYSTEM” -> “クライアント起動”を選択・実行します。ログインするときは、管理者特権をもったユーザー（例えば DeviceServer セットアップ時に管理者アカウントとして登録したユーザー等）でログインしてください。デスクトッププログラムが起動したら、 “XBee” ツールボタンを選択してXBee デバイス管理プログラムを起動します。

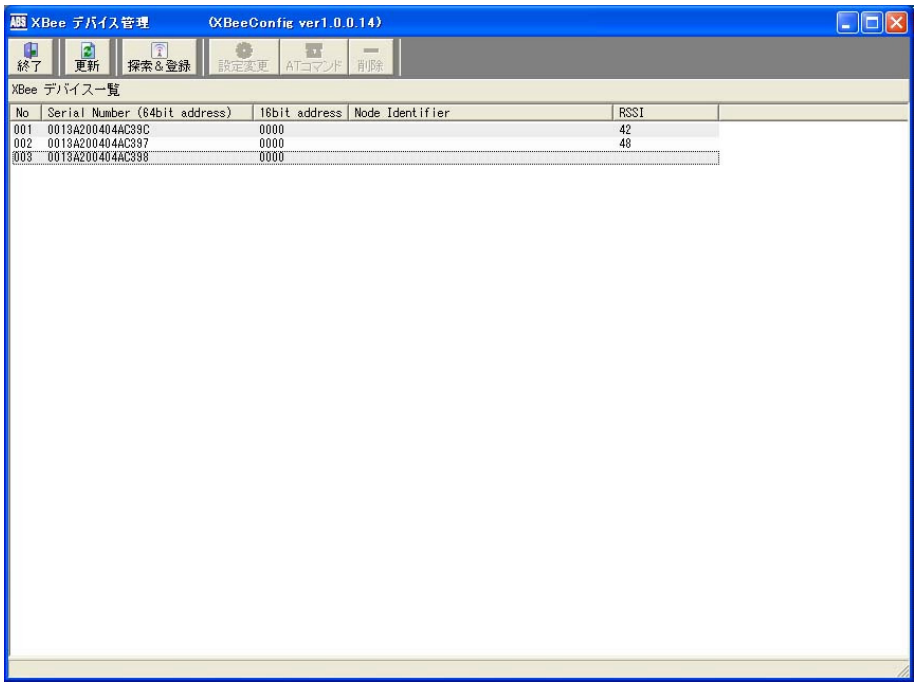
DeviceServer では 登録済みの XBee デバイスをマスターファイルに記録しています。

XBee デバイス登録は、”探索＆登録” ボタンを押すことで、同一 PAN ID のリモートデバイスを見つけて、自動的にマスターファイルに登録します。既に、登録済みのXBee デバイスの場合は最新の情報でマスターファイルの内容が更新されます。DeviceServer に COMポートで直接接続された XBee デバイスについても同様に自動登録されます。

XBee デバイス管理プログラムの “探索＆登録” ツールボタンを押します。

登録確認ダイアログが表示されますので、”OK” を押します。

DeviceServer の COM ポートに直接接続された XBee デバイスで “Node discover” が実行され、付近にある同一 PAN ID の XBee デバイス情報を取得して、自動的にマスターファイルに登録が行われます。XBee デバイス管理プログラムのデバイス一覧には、登録済みのXBee デバイスが表示されます。



（デバイスの探索＆登録が完了したときの画面）

XBee デバイスの詳細設定を変更するために、XBee デバイス管理プログラムのデバイス一覧から対象デバイスを選択して、”設定変更” ツールボタンを押します。初期設定時に 16 bit Source Address を設定した場合は、その値がデバイス一覧に表示されていますので、変更対象の XBee デバイスを確認することができます。

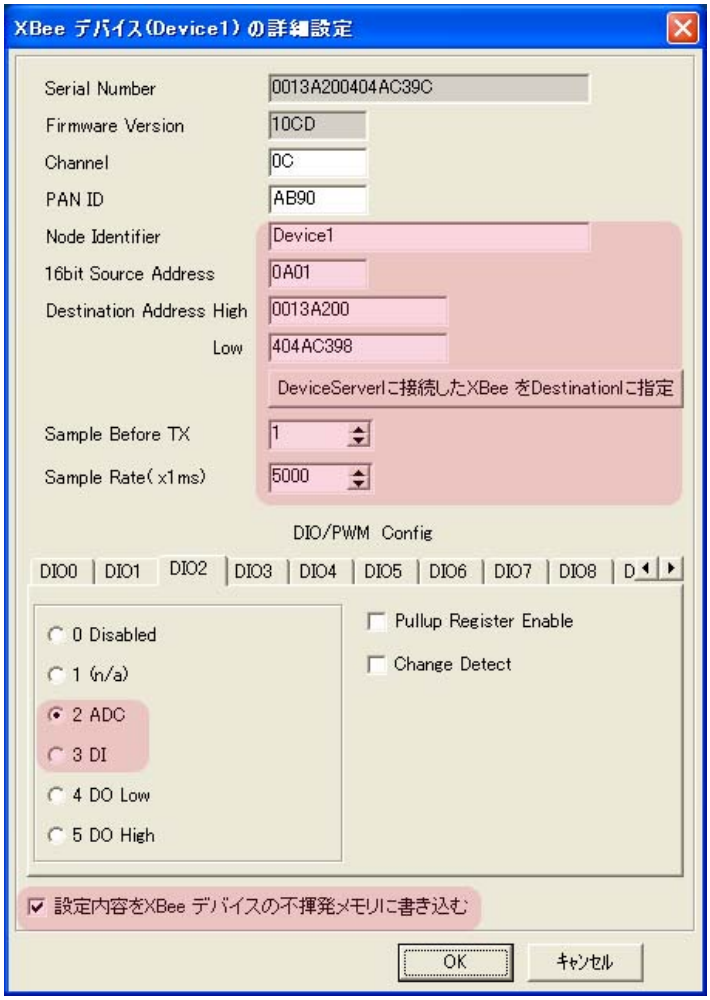
選択したXBee デバイスと通信を行って、現在のデバイス情報を取り込みます。

もしエラーが発生した場合は、選択したXBee デバイスとの間で通信ができない状態になっていますので、通信経路や電源等を確認してください。

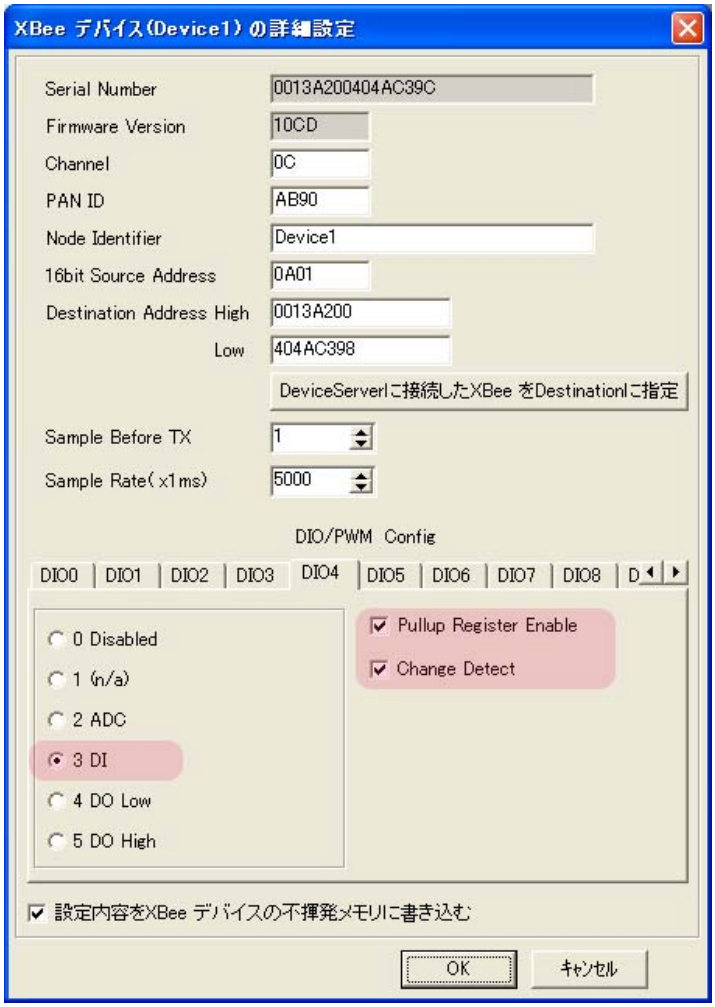
XBee デバイスの現在の設定値を取り込んだ後、詳細設定変更ダイアログが表示されます。
最初に、XBee デバイスの Node Identifier の設定を行って下さい。Node Identifier に指定可能な文字は ASCIIで20文字までです。また、ダイアログに表示されている 16 bit Source Address が、対象のデバイスであるかどうかの確認も行ってください。ここで 16 bit Source Address を任意の値に変更することも可能です。

リモート計測側の XBee デバイス(Device1, Device3) の場合のみ、ハードウエアの構成に合わせて DIO/PWM Config のタブから DIOポートの設定を “DI”, “ADC”,”Disabled” 等に設定します。

同様に、リモート計測側の XBee デバイス(Device1, Device3) の場合のみ、サンプリングとI/Oデータ送信先アドレスの設定を行います。”DeviceServerに接続したXBee をDestinationに指定” ボタンを押すと、Destination Address High と Destination Address Low をDeive2 の64ビットアドレス(Serial Number)に自動的に設定することができます。 Sample Before TX を 1 にして、Sample Rate を Device1 は 5000(5秒) Device3 は 10000(10秒)に設定します。



DIO 入力値変化のイベントで I/O データを送信する場合は、Change Detect も有効に設定します。ここでは、DIO4 に外付けのスイッチが接続されていて、そのスイッチ状態の変化で I/O データ送信を行う様に設定しています。



最後に、”設定内容を XBee デバイスの不揮発メモリに書き込む” にチェックを付けて “OK” を押してください。

XBeeデバイスの Node Identifier を変更した場合は、XBee デバイス管理プログラムのデバイス一覧に表示されている、マスターファイルも更新しておく必要があります。
XBee デバイス管理プログラムの “探索＆登録” ツールボタンを押します。

デバイスのNode Identifier または 16bit Source Address以外の詳細設定を変更する場合には、マスターファイルの更新は必要ありません。

5.4 サーバー設定(Oracle 接続)

計測データ保管用のデータベースに Oracle10g サーバーを使用する場合は、プログラムメニューから”サーバー設定” を選択・実行して下記の項目を設定します。このほかにもOracle サーバーに保管用テーブル作成と、オラクルクライアントの設定が必要となります。詳細は “DeviceServer ユーザーマニュアル” を参照してください。

サーバー設定プログラム(Oracle 使用時のみ設定が必要)	
設定が必要な項目	**設定内容**
Oracle接続機能を有効にする	チェックを付ける
Oracleユーザー名	Oracle サーバーの設定に合わせて下さい
Oracleパスワード	Oracle サーバーの設定に合わせて下さい
ホスト接続文字列	Oracle クライアントの設定に合わせて下さい
同時接続数	4を設定します。Oracle クライアントライセンスによっては、設定可能な接続数に制限がある場合があります。(1 以上の値を設定する必要があります)

5.5 スクリプト設定

⚠ 注意

スクリプト中に日本語を記述するときは、スクリプトファイルを UTF-8N 形式で保存してください。Shift_JISや UTF-8 BOM付き形式などで保存すると、DeviceServer でエラーが発生します。Windows付属のワードパッドやメモ帳ではこの形式で保存できませんので、別途 UTF-8N 形式で保存可能なエディタソフト（*1）を使用してください。

（*1）TeraPad 等のソフトウエアがよく使用されています。

5.5.1 XBEE_DEV_LIST スクリプト作成

リモート XBee デバイス一覧を取得するためのスクリプトを作成します。エクセル中の VBA からこのスクリプトが実行されます。DeviceServer のマスターファイルに登録された XBee デバイスを検索してスクリプト呼び出し元にデバイス一覧を返します。

ファイル名(XBEE_DEV_LIST.lua) で DeviceServerのスクリプトフォルダに保管します。

```
--[[
*********************************************************************
アクイジション対象のリモートXBeeデバイスのリストをリターン値に返す
DeviceServer のマスターに登録された XBee デバイスを対象に選択している。
もし、個別のデバイスを指定したい場合は、xbee_all_list()を使用しないで
下記の様にデバイスを個々にリターン値に返す様にする。
XBee の NodeIdentifier が "DeviceA","DeviceB" の２つのデバイスを
アクイジション対象にする場合は、スクリプトを下記の様にする。
script_result の第２パラメータ（キー名）はユニークな文字列であれば
なにを指定しても構わない。
script_result(g_taskid,"1","DeviceA")
script_result(g_taskid,"2","DeviceB")
*********************************************************************
```

```
]]

local stat,serial,addr,name,is_local = xbee_all_list();
for key,val in ipairs(name) do
  script_result(g_taskid,tostring(key),val)
end;

--[[
script_result(g_taskid,"1","Device1")
script_result(g_taskid,"2","Device2")
script_result(g_taskid,"3","Device3")
]]
```

5.5.2 XBEE_IO_DATA イベントハンドラ作成

リモート XBee デバイスから、I/O データが送信されたときに、そのデータをデータベースに格納するスクリプトを作成します。

データベースには、デバイス毎に取り込みシーケンス番号(AcuisitionNumber) をつけて保存されます。また、I/O データを受信したタイムスタンプもデータベース中に保管します。

データベースに保管するキー名	値
ACQNUM-<DeviceName>	データ取り込みカウンタ値 (AcquisitionNumber) 1 以上の整数。 1回の取り込み毎に、インクリメントされていく。デバイス別に保持している。
ACQAT-<DeviceName>-<AcquisitionNumber>	<YYYY/MM/DD HH:MM:SS> 日付と時間を表したタイムスタンプ値。 2001/1/31 10:20:30にデータを取得した場合は '2001/01/31 10:20:30' になる
ACQADC-<DeviceName>-<AcquisitionNumber>	<ADC0>,<ADC1>,<ADC2>,<ADC3>,<ADC4>,<ADC5> ADCxx の値は 0 から 1023 までの整数が入る ADCxx のサンプリング値が無い場合は、"NA" 文字列が入る
ACQDIO-<DeviceName>-<AcquisitionNumber>	<DIO0>,<DIO1>,<DIO2>,<DIO3>,<DIO4>,<DIO5>,<DIO6>,<DIO7>,<DIO8> DIOxx の値は 0 または 1の整数が入る。 (1 は High, 0 は Low レベル) DIOxx のサンプリング値が無い場合は、"NA" 文字列が入る

<DeviceName> には、XBee デバイスの NodeIdentifierが入ります。

<YYYYMMDDHHMMSS> には、西暦日付と時刻から成るタイムスタンプ文字列が入ります。

（例 2008/1/1 6:50:01 pm は、"20080101185001" ）

スクリプト中の set_permanent_data(),inc_permanent_data() 部分を set_oracle_data(),inc_oracle_data() に変更すると、保管先のデータベースを Firebird から、Oracle DBMS にすることができます。

I/O データ中に複数のサンプルデータが含まれていた場合(Sample Before TX に1 よりも大きな値を設定した場合)には、常に最終のサンプルデータだけをデータベースに保管するようにしています。スクリプトを修正して全てのサンプル値を保管するようにすることも可能です。

ファイル名(XBEE_IO_DATA.lua) で DeviceServerのスクリプトフォルダに保管します。

```
file_id = "XBEE_IO_DATA"
--------------------------------------------------------------------------
-- XBeeデバイスのI/O 値を、データベースに格納する。
-- I/O データは、下記のキー名で DeviceServer のデータベースに格納する。
-- データフレーム中のサンプル数が 1 より多い場合でも、
-- 常に最後のサンプルデータだけを取得対象にしている。
-- 詳細説明はアプリケーションノート (APNOTE08) を参照の事
--
-- データベースキー名： ACQNUM-<DeviceName>
-- データベース値　　： データ取り込みカウンタ値 (AcquisitionNumber) 1 以上の整数
--                      1回の取り込み毎に、インクリメントされていく。デバイス別に保持している。
--
-- データベースキー名： ACQAT-<DeviceName>-<AcquisitionNumber>
-- データベース値　　： <YYYY/MM/DD HH:MM:SS> 日付と時間を表したタイムスタンプ値。2001/1/31 10:20:30に
--                      データを取得した場合は '2001/01/31 10:20:30' になる
--
-- データベースキー名： ACQADC-<DeviceName>-<AcquisitionNumber>
-- データベース値　　：<ADC0>,<ADC1>,<ADC2>,<ADC3>,<ADC4>,<ADC5>
--                      ADCxx の値は 0 から 1023 までの整数が入る
--                      ADCxx のサンプリング値が無い場合は、"NA" 文字列が入る
--
-- データベースキー名： ACQDIO-<DeviceName>-<AcquisitionNumber>
-- データベース値　　：<DIO0>,<DIO1>,<DIO2>,<DIO3>,<DIO4>,<DIO5>,<DIO6>,<DIO7>,<DIO8>
--                      DIOxx の値は 0 または 1の整数が入る。(1 は High, 0 は Low レベル)
--                      DIOxx のサンプリング値が無い場合は、"NA" 文字列が入る
--------------------------------------------------------------------------
--[[

XBEE_IO_DATA スクリプト起動時に渡される追加パラメータ
```

```
--------------------------------------------------------------------------------
キー値                    値                                          値の例
--------------------------------------------------------------------------------
APIType                   フレームデータ中のAPI Type(16進数2桁)       83
SourceAddress             フレームデータ中のSourceAddress
                          16bit アドレスの場合(16進数4桁)             0A01
                          64bit アドレスの場合(16進数16桁)            0013A200404AC397
NodeIdentifier            XBee デバイスの NodeIdentifier。
                          DeviceServer に保持されたマスターファイルを使用して、
                          SourceAddress から変換した値が設定される。  Device1
                          マスターにNodeIdentifier未登録の場合は"" が設定される
RSSI                      フレームデータ中のRSSI(16進数2桁)           45
Options                   フレームデータ中Options(16進数2桁)          00
SAMPLE_COUNT              I/O データのサンプル数                      1
SAMPLE_DIO                I/O データ中のサンプル対象となったDIOビット番号リスト
                          (10進数、カンマ区切り)                      0,1,4
SAMPLE_ADC                I/O データ中のサンプル対象となったADCビット番号リスト
                          (10進数、カンマ区切り)                      2,3
SAMPLE_<Sample#>_<"DIO"|"ADC">_<Bit#>
                          I/O サンプルデータ値。
                          DIO の場合は、High で "1"、Low で "0"。     1
                          ADC の場合は 10進数。                       1023
                          <Sample#> には 最大、SAMPLE_COUNT まで 1から順番にインクリメントされた値が入る。
                          <"DIO"|"ADC">は、I/O サンプルデータが ADCもしくは、DIO のどちらであるかを示す。
                          <Bit#>は、サンプルデータのビット番号。10進数。
]]


------------------------------------------------------------
-- DeviceServerに登録済みのデバイスからのパケットのみを取り込む
------------------------------------------------------------
local device_name = g_params["NodeIdentifier"];
if (device_name ~= "") then
  --------------------------
  -- AcquisitionNumber 取得
  --------------------------
  local key_acq_number = "ACQNUM-" .. device_name;
  local stat,acq_number = inc_permanent_data(key_acq_number);
  --local stat,acq_number = inc_oracle_data(key_acq_number);
  if not stat then error() end;
```

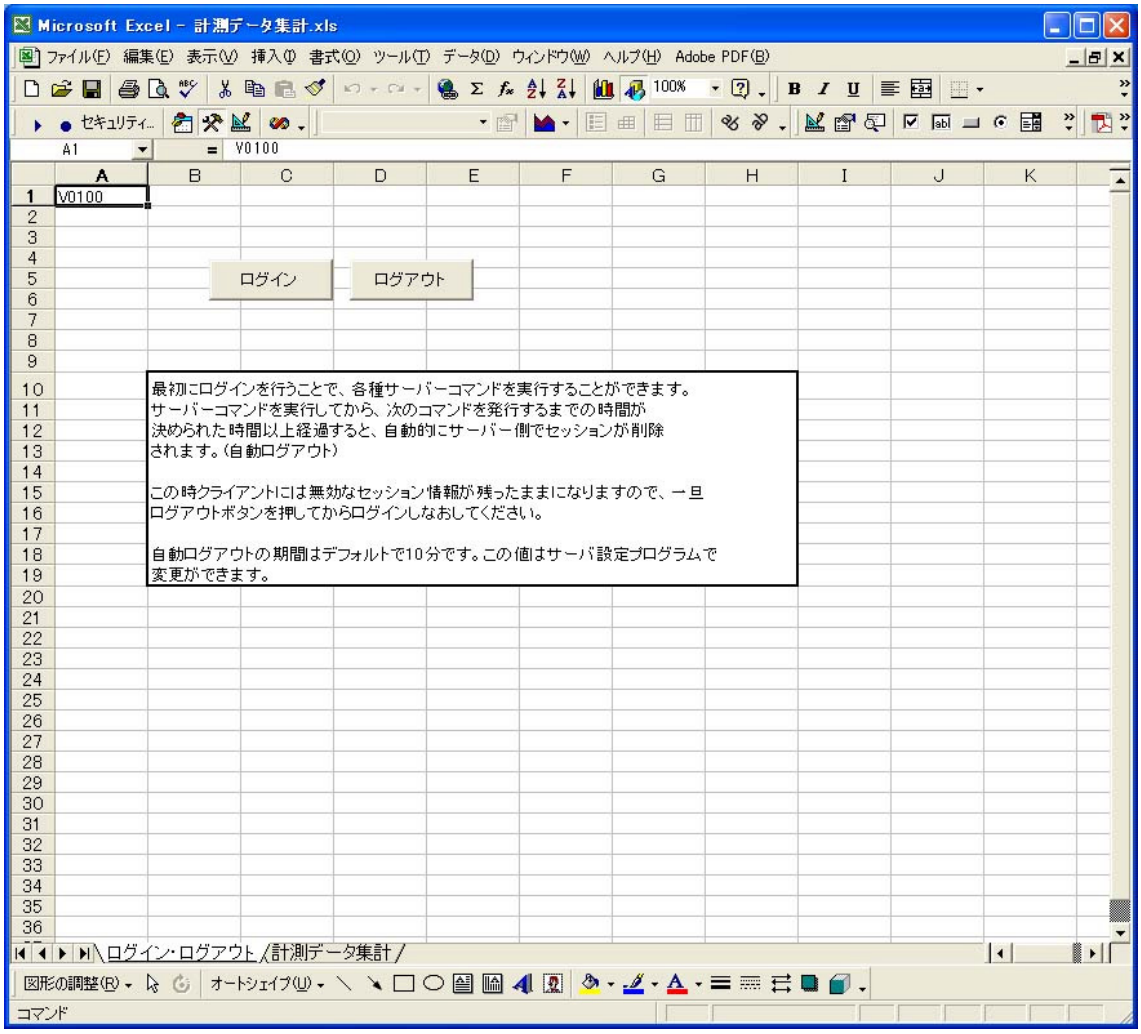
```
  log_msg("acquisition device = " .. device_name .. " number = " .. tostring(acq_number),file_id);

  --------------------------
  -- タイムスタンプ取得
  --------------------------
  local now = os.date "*t";
  local timestamp =
string.format("%4.4d/%2.2d/%2.2d %2.2d:%2.2d:%2.2d",now["year"],now["month"],now["day"],now["hour"],now["
min"],now["sec"]);
  if not set_permanent_data("ACQAT-" .. device_name .. "-" .. tostring(acq_number),timestamp) then error()
end
  --if not set_oracle_data("ACQAT-" .. device_name .. "-" .. tostring(acq_number),timestamp) then error() end

  --------------------------
  -- ADC データ取得
  --------------------------
  local value_str = "";
  local key_name_prefix;
  local key_name;
  key_name_prefix = "SAMPLE_" .. g_params["SAMPLE_COUNT"] .. "_ADC_";
  for bit = 0,5,1 do
    key_name = key_name_prefix .. tostring(bit);
    if g_params[key_name] then
      value_str = value_str .. g_params[key_name]
    else
      value_str = value_str .. "NA"
    end;
    if (bit ~= 5) then value_str = value_str .. "," end;
  end;
  if not set_permanent_data("ACQADC-" .. device_name .. "-" .. tostring(acq_number),value_str) then error()
end;
  --if not set_oracle_data("ACQADC-" .. device_name .. "-" .. tostring(acq_number),value_str) then error()
end;

  --------------------------
  -- DIO データ取得
  --------------------------
  value_str = "";
  key_name_prefix = "SAMPLE_" .. g_params["SAMPLE_COUNT"] .. "_DIO_";
```

```
  for bit = 0,8,1 do
    key_name = key_name_prefix .. tostring(bit);
    if g_params[key_name] then
      value_str = value_str .. g_params[key_name]
    else
      value_str = value_str .. "NA"
    end;
    if (bit ~= 8) then value_str = value_str .. "," end;
  end;
  if not set_permanent_data("ACQDIO-" .. device_name .. "-" .. tostring(acq_number),value_str) then error()
end;
  --if not set_oracle_data("ACQDIO-" .. device_name .. "-" .. tostring(acq_number),value_str) then error()
end;
else
  log_msg("*ERROR* unknown device, address = " .. g_params["SourceAddress"],file_id);
end;
```

6 エクセルからデータ取得・集計

クライアントPC のエクセルから、データベースに保存された計測データを取り込んで集計を行います。
データの取得とデータベース保管は、DeviceServer サービスプログラムで、バックグランドで実行されていますので、クライアントからの集計は任意のタイミングで行えます。

以降で説明している内容は、添付のエクセルファイル(計測データ集計.xls)のワークシートを実行したものです。全てのマクロ(VBA スクリプト)とワークシートシートはエクセルファイル(計測データ集計.xls)内に記述されていますので、詳細はエクセルをデザインモードにして、”Visual Basic Editor” で開いてください。以下に、データ取り込み用のワークシートのマクロ実行画面と、マクロ(VBA スクリプト)の動作について説明します。



（計測データ集計.xls を開いた画面。起動時にエクセルのマクロを有効にして下さい）

クライアントPC でエクセルを実行する場合には、予め XASDLCMD.DLL をクライアントPC のシステムフォルダにコピーしておく必要があります。エクセルを実行するPC が DeviceServer の動作しているPC と同一の場合は DLL のコピーは必要ありません。詳細は “DeviceServerユーザーマニュアル” の ”インストール” 章の “ユーザーアプリケーションを利用する場合” の項目を参照してください。

6.1 DeviceServer にログイン

計測データ集計.xls をエクセルで開いて、”ログイン・ログアウト” ワークシートを選択します。

ログインボタンを押して、DeviceServer にログインします。この時に指定するユーザーは、DeviceServer に登録済みのユーザーを指定してください。スケジューラでスクリプト実行用に作成したユーザー以外も指定することができます。ホスト名には、DeviceServer の動作しているPC のホスト名を入力します。

ワークシート内のログインボタンを押した時に実行されるマクロ(VBA)部分は以下の様に記述されています。ログインに成功するとLastSessionToken にセッショントークン文字列が入っています。未ログインの場合は空文字列が入っています。

```
Private Sub LoginBtn_Click()
  If LastSessionToken <> "" Then
    MsgBox ("現在ログイン中です。一旦ログアウトしてからやり直してください")
    Exit Sub
  End If
  LoginForm.LoginPasswordEdit.Text = ""
  LoginForm.Show
End Sub
```

ログインフォームの"OK" ボタンを押した時に実行されるマクロ(VBA)部分は以下の様に記述されています。

```
Private Sub CommandButton1_Click()
 If Not LoginUser(ServerEdit.Text, LoginNameEdit.Text, LoginPasswordEdit.Text) Then
    MsgBox ("ログインに失敗しました")
  End If
  LoginForm.Hide
End Sub
```

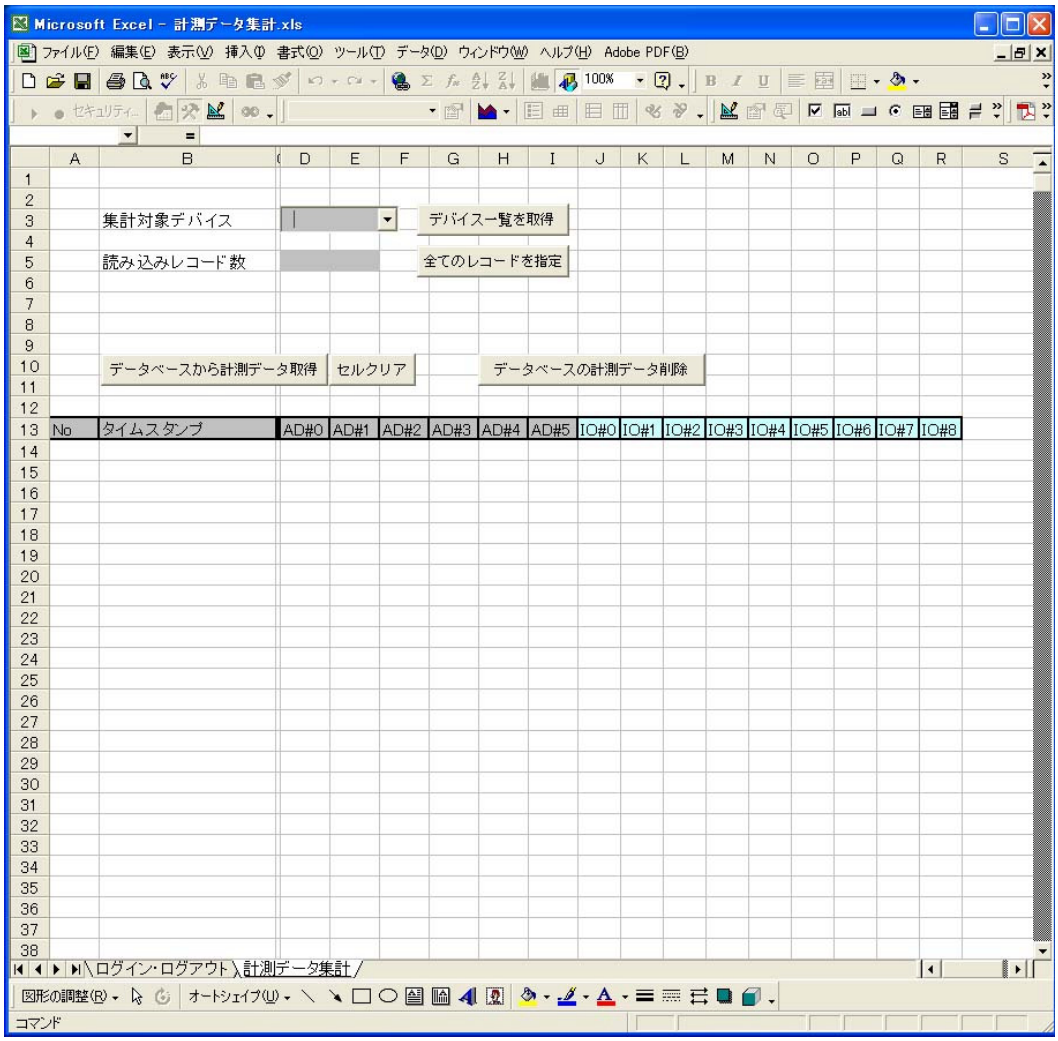
標準モジュールに定義された LoginUser() は以下の様になっています。

```
' この関数を直接コールしないで、代わりに LoginUser() を使用してください
Public Declare Function SX_LoginUser Lib "XASDLCMD.dll " (ByVal Host As String, ByVal Port As Integer, ByVal
UserName As String, ByVal Password As String, ByVal Session As String) As Integer

Public LastSessionToken As String
Public LastLoginName As String
Public LastLoginHost As String
```

```
Public Function LoginUser(ByVal Host As String, ByVal UserName As String, ByVal Password As String) As Boolean
  Dim NewSession As String * 128
  If SX_LoginUser(Host, DefPort, UserName, Password, NewSession) = 0 Then
    LastLoginHost = Host
    LastLoginName = UserName
    LastSessionToken = NewSession
    LoginUser = True
  Else
    LoginUser = False
  End If
End Function
```

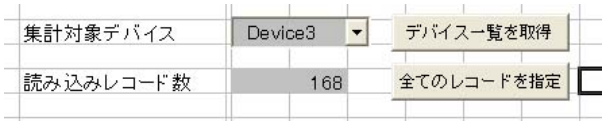
6.2 データ取り込み範囲指定

ログインに成功したら、”計測データ集計” ワークシートを選択します。



最初にデータベースから計測データを取得する条件を設定します。デバイス名と読み込みレコード数を指定します。デバイス名はコンボボックスになっていますので、直接 XBee デバイスの名前(NodeIdentifier)を指定することができます。”デバイス一覧を取得” ボタンを押すと、“XBEE_DEV_LIST スクリプト作成” で作成したスクリプトを DeviceServerで実行して、コンボボックスのリストに最新のデバイス名をロードすることができます。

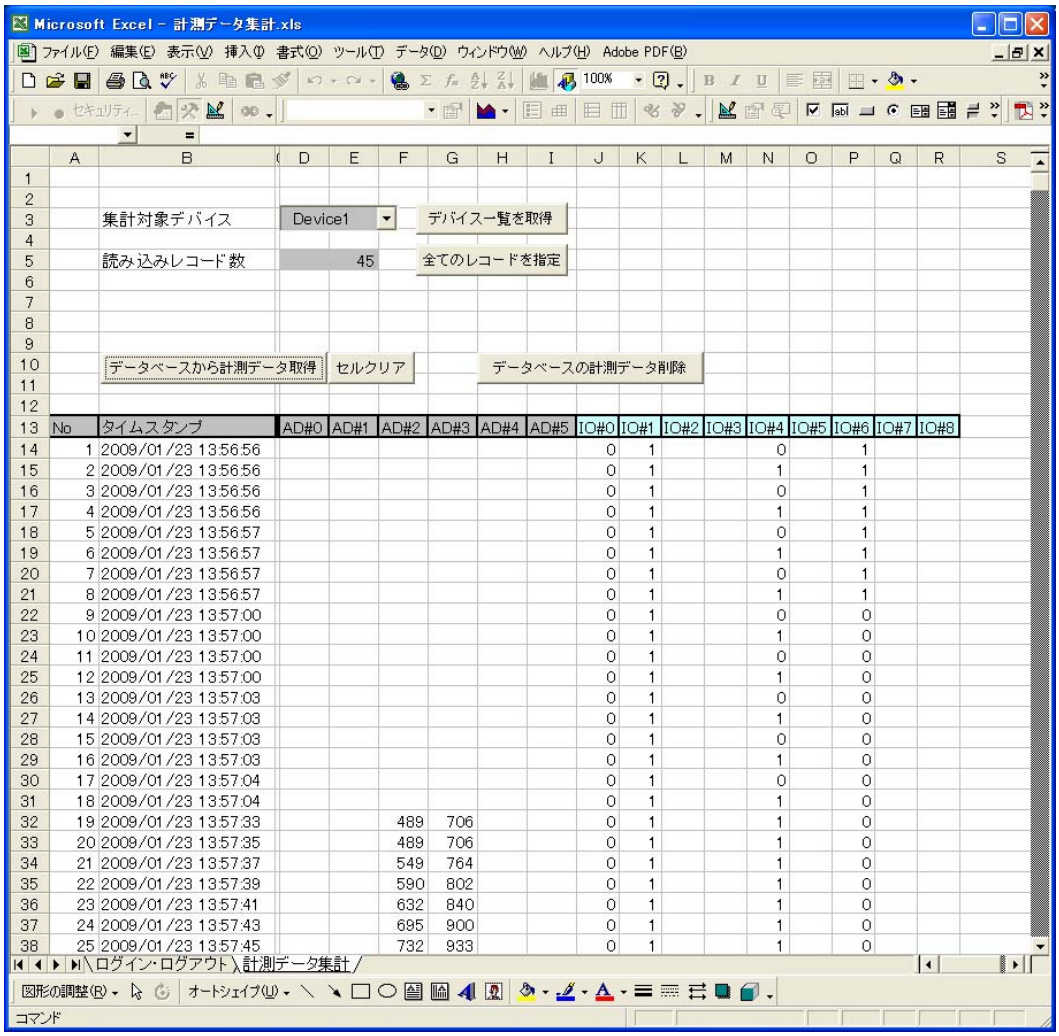
データベースに保存された I/O データは、デバイス毎に 1 から順番に収集番号(AcquisitionNumber) が振られていています。読み込みレコード数に設定した収集番号までの I/O データをエクセルにロードします。 “全てのレコードを指定” ボタンを押すと、最後の収集番号が読み込みレコード数に設定され、全てのレコードがロード対象となります。



（XBeeデバイス名 “Device3” の I/O データを168 番目まで取得する例）

6.3 データ取得と集計

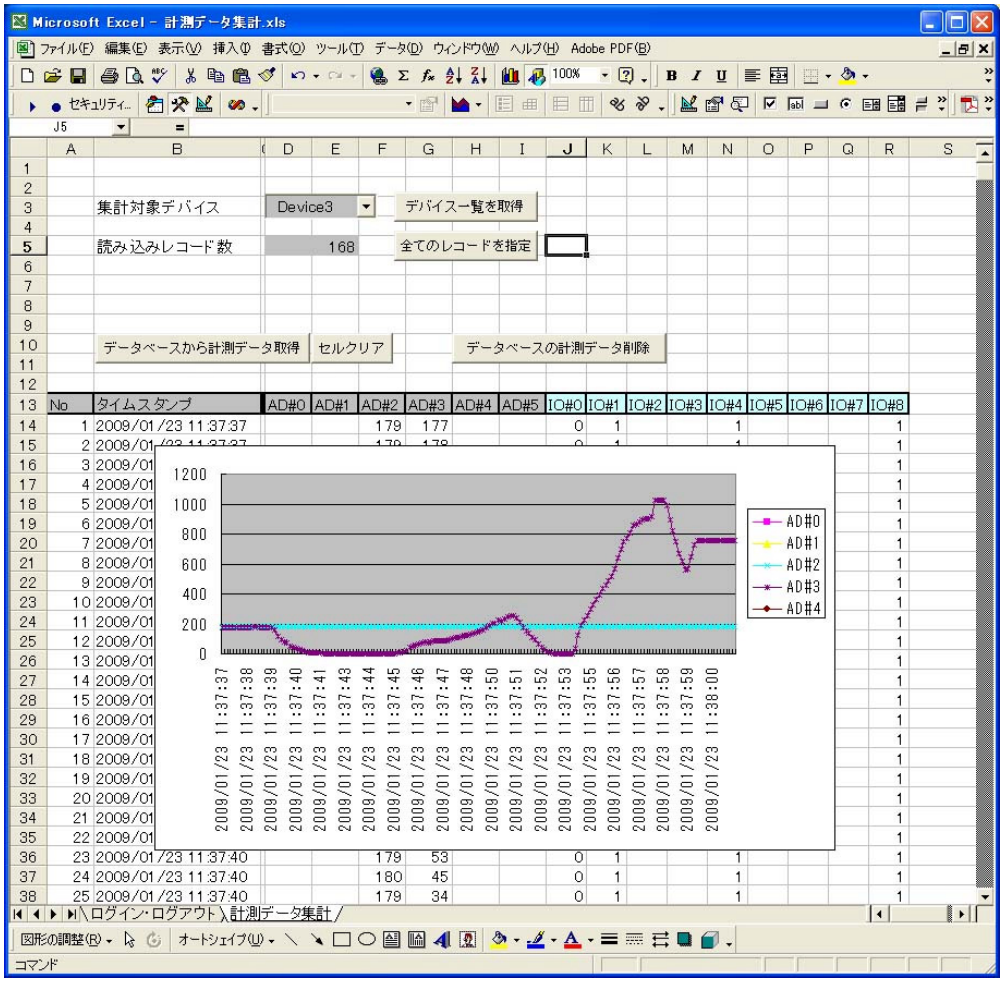
集計対象条件を入力した後に、”データベースから計測データ取得” ボタンを押します。集計対象に指定された計測データを DeviceServer のデータベースから取得して、ワークシートのセルに入力されます。



計測データが見つからない場合は、セルは空になります。DIO, A/D 変換値はそれぞれのチャンネル毎にセルに値が入ります。Change Detect で XBee デバイスから I/O データが送信された場合には、A/D チャンネルのデータは含まれないため、全てのA/D チャンネルのセルは空になります。

計測データをセルに取り込んだ後は、エクセルで自由に加工して集計を行うことができます。

計測データの A/D 変換値をグラフ化した例は以下の様になります。



”データベースから計測データ取得” ボタンを押した時に実行されるマクロ(VBA)部分は、以下の様に記述されています。マクロ(VBA)中の SX_get_permanent_data() を SX_get_oracle_data() に変更すると、計測データの取得先のデータベースを Firebird から、Oracle DBMS にすることができます。

CsvToList() 関数等、詳しい内容はエクセルファイル(計測データ集計.xls)の内容を参照してください。

```
Private Sub DoSummarizeBtn_Click()
  Call fetch_data
End Sub
Public Sub fetch_data()
  Dim TimeStampStr As String
  Dim KeyName As String
  Dim i As Integer
  Dim cntr As Integer
  Dim MaxSample As Integer
  Dim SharedValue As String * 256
  Dim ChackStr As String
```

```
Dim TmpVal As Integer



MaxSample = ActiveSheet.Range("D5").Value
For i = 1 To MaxSample

  ' Acqusition Number
  ActiveSheet.Cells(start_line + i - 1, start_column) = I

  ' タイムスタンプ取得
  KeyName = "ACQAT-" & DeviceListCbx.Text & "-" & Format(i)
  If SX_get_permanent_data(LastSessionToken, LastLoginHost, DefPort, KeyName, SharedValue) <> 0 Then
  'If SX_get_oracle_data(LastSessionToken, LastLoginHost, DefPort, KeyName, SharedValue) <> 0 Then
    MsgBox (SVR_ERR_MSG)
    Exit Sub
  End If
  ActiveSheet.Cells(start_line + i - 1, start_column + 1) = SharedValue

  ' A/D データ取得
  KeyName = "ACQADC-" & DeviceListCbx.Text & "-" & Format(i)
  If SX_get_permanent_data(LastSessionToken, LastLoginHost, DefPort, KeyName, SharedValue) <> 0 Then
  'If SX_get_oracle_data(LastSessionToken, LastLoginHost, DefPort, KeyName, SharedValue) <> 0 Then
    MsgBox (SVR_ERR_MSG)
    Exit Sub
  Else
    CheckStr = Trim(Left(SharedValue, InStr(SharedValue, vbNullChar) - 1))
    If CheckStr <> "" Then
      ' A/D データリスト(CSV)の各フィールドをセルに代入
      Call CsvToList(SharedValue)
      For cntr = 0 To (result_csv_clmcnt - 1)
        If result_csv(cntr) <> "NA" Then
          ActiveSheet.Cells(start_line + i - 1, start_column + 3 + cntr) = result_csv(cntr)
        End If
      Next cntr
    End If
  End If

  ' DIO データ取得
  KeyName = "ACQDIO-" & DeviceListCbx.Text & "-" & Format(i)
```

```
    If SX_get_permanent_data(LastSessionToken, LastLoginHost, DefPort, KeyName, SharedValue) <> 0 Then
    'If SX_get_oracle_data(LastSessionToken, LastLoginHost, DefPort, KeyName, SharedValue) <> 0 Then
      MsgBox (SVR_ERR_MSG)
      Exit Sub
    Else
      CheckStr = Trim(Left(SharedValue, InStr(SharedValue, vbNullChar) - 1))
      If CheckStr <> "" Then
        ' A/D データリスト(CSV)の各フィールドをセルに代入
        Call CsvToList(SharedValue)
        For cntr = 0 To (result_csv_clmcnt - 1)
          If result_csv(cntr) <> "NA" Then
            ActiveSheet.Cells(start_line + i - 1, start_column + 9 + cntr) = result_csv(cntr)
          End If
        Next cntr
      End If
    End If
  Next i
  MsgBox ("データ取得 完了しました")
End Sub
```

7 このドキュメントについて

7.1 著作権および登録商標

7.2 連絡先

オールブルーシステム (All Blue System)

ウェブページ http://www.allbluesystem.com

メール contact@allbluesystem.com

7.3 このドキュメントの使用について

このドキュメントは、ABS-9000 DeviceServer の一般的な使用方法と応用例について解説してあります。お客様の個別の問題について、このドキュメントに記載された内容を実際のシステムに利用するときには、ここに記載されている以外にも考慮する事柄がありますので、ご注意ください。特に安全性やセキュリティ、長期間にわたる運用を想定してシステムを構築する必要があります。

オールブルーシステムでは ABS-9000 DeviceServer の使用や、このドキュメントに記載された内容を使用することによって、お客様及び第三者に損害を与えないことを保証しません。 ABS-9000 DeviceServer を使用したシステムを構築するときは、お客様の責任の下で、システムの構築と運用が行われるものとします。

8 更新履歴

REV A.1.1 2010/3/31

XBee デバイスを DeviceServer 全てのライセンスで扱える様に記述を変更

REV A.1.0 2009/1/24

初版作成