TW-IOT システム
ユーザーズマニュアル

テクノウェーブ株式会社

1. はじめに

□ 注意事項

- お客様の不注意、誤操作により発生した製品、パソコン、その他の故障、及び事故につきまして弊社は一切の責任を負いませんのでご了承ください。
- 本製品または、付属のソフトウェアの使用による要因で生じた損害、逸失利益または第三者からのいかなる請求についても、弊社は一切その責任を負えませんのでご了承ください。
- 本システムの構成要素となるサーバーのインストールや基本的な設定等に関するお問い合わせやご質問については、サポートの対象外とさせていただきますのでご了承ください。

□ マニュアル内の表記について

本マニュアル内ではハードウェアの各電気的状態について下記のように表記いたします。

表 1　電気的状態の表記方法

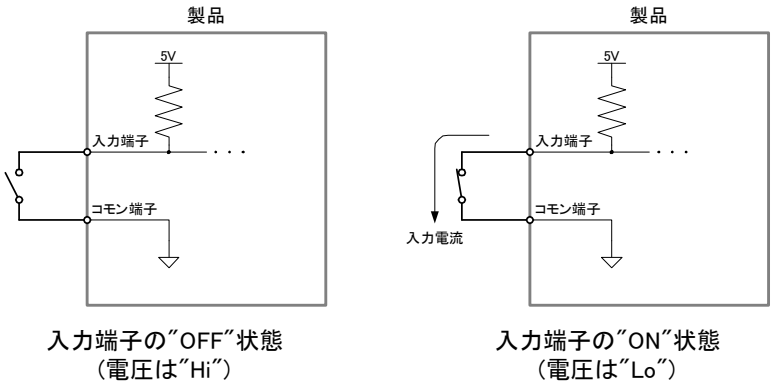
表記	状態
“ON”	電流が流れている状態、スイッチが閉じている状態、オープンコレクタ(オープンドレイン)出力がシンク出力している状態。
“OFF”	電流が流れていない状態、スイッチが開いている状態、オープンコレクタ(オープンドレイン)出力がハイインピーダンスの状態。
“Hi”	電圧がロジックレベルのハイレベルに相当する状態。
“Lo”	電圧がロジックレベルのローレベルに相当する状態。

数値について「0x」、「&H」 、「H’」はいずれもそれに続く数値が 16 進数であることを表します。“0x10”、“&H1F”、“H’20”などはいずれも 16 進数です。

本マニュアル内でシェルのコマンドラインを例示する場合、一般ユーザー権限で行うものには、「>」プロンプト、管理者権限で行うものには「#」プロンプトをコマンドの先頭に記述しています。

デジタル入力端子の状態

デジタル入力端子は十分な入力電流が流れている状態を″ON″、入力電流が流れていないか十分でない場合を″OFF″とします。



入力端子の″OFF″状態
（電圧は″Hi″）

入力端子の″ON″状態
（電圧は″Lo″）

図 1　デジタル入力端子の"OFF"状態と"ON"状態

デジタル出力端子の状態

デジタル出力端子は出力電流が流れている状態を"ON"、流れていない状態を"OFF"とします。

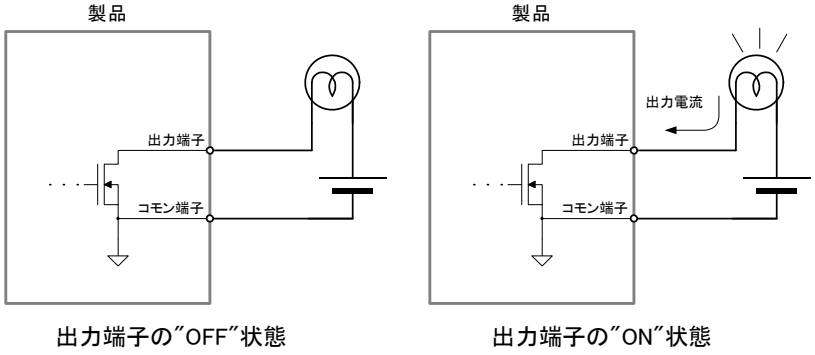


図 2　デジタル出力端子の"OFF"状態と"ON"状態

2. システム概要

□ TW-IOTシステムについて

「TW-IOT システム」は、弊社製品の「LANX-I2219」(以下、製品またはデバイス)、サーバー、WEB ブラウザから構成される Internet of Tihngs(IoT)システムです。

本システムを構成する製品およびプログラムの機能は表 2を参照してください。

表 2 システムの構成要素

構成要素	機能
LANX-I2219	「HTTP クライアント・ファームウェア」をダウンロードすることで、HTTP を使用してデジタル入力、アナログ入力、カウンタなどのデータをサーバーへ送信します。また WEB ブラウザからのデジタル出力、アナログ出力、PWM 出力などの制御命令を、サーバーを経由して受信することが可能となります。
サーバー	デバイスから送信されたデータをデータベースに蓄積します。また WEB ブラウザからの制御命令をデバイスに送信することが可能です。 Apache®、MySQL®、PHPのインストールと設定が必要です。
WEB ブラウザ	デバイスからサーバーに送信されたデータを表示します。また、サーバーを経由してデバイスのデジタル出力、アナログ出力、PWM 出力などを制御することが可能です。

□ TW-IOTシステムの特徴

- WEB ブラウザとサーバーとの通信に Comet および Ajax を使用することでサーバーに負荷をかけずに低レイテンシを実現
- データの表示に WEB ブラウザを使用することで、パソコン、スマートフォン、タブレットなどの様々なプラットフォームを使用可能
- WEBブラウザからデバイスの制御を行う際にサーバーを経由するため、多対多通信を実現することが可能(図 3)
- デバイスから送信されたデータはサーバーのデータベースに蓄積されるので、過去のデータを取得し、比較することが可能

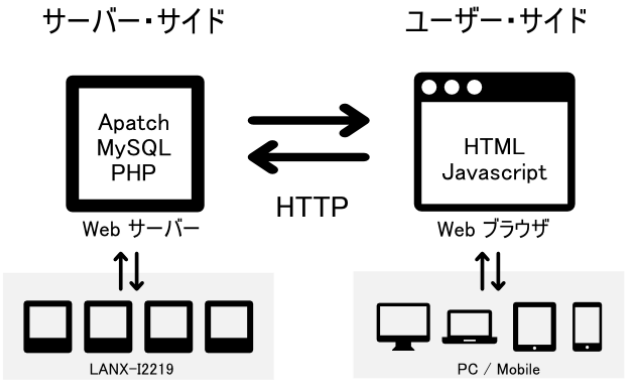


図 3 接続概念図

Apache は、Apache Software Foundation の米国およびその他の国における登録商標または商標です。
MySQL は、Oracle Corporation およびその子会社、関連会社の米国およびその他の国における登録商標です。

3. デバイスの準備

□ HTTPクライアント・ファームウェアについて

「HTTP クライアント・ファームウェア」は、LANX-I2219 用の追加ファームウェアです。HTTP を使用したサーバーへのデータのアップロードや、命令のダウンロードが可能となります。

「HTTP クライアント・ファームウェア」では以下の機能をサポートします。

- デジタル入出力
- アナログ入出力
- ソフトウェアカウンタ
- ハードウェアカウンタ
- PWM パルス出力

製品の機能、設定などの基本事項につきましては「LANX-I2219 ユーザーズマニュアル」に記載しています。合わせてご参照ください。

□ HTTPクライアント・ファームウェアの関連ファイル

使用するファイルは製品付属 CD の「¥X2219_AdditionalFirm¥TW-IOT」フォルダに収められています。また、以下の URL からも最新版をダウンロード可能です。

http://www.techw.co.jp/x2219/x2219_support1.htm#download

また、HTTPクライアント・ファームウェアのソースファイルは、製品のユーザーファーム開発用ファイル[1]に含まれていますので、ユーザーファームの開発環境があればカスタマイズすることが可能です。

ユーザーファーム開発の詳細は「USBX-I2219/LANX-I2219 ユーザーファーム開発マニュアル」を参照してください。

□ LANX2219Toolsのインストール

HTTP クライアント・ファームウェアを使用するためには、「LANX2219Tools」をインストールする必要があります。製品添付 CD の「¥TOOL¥LANX2219Tools¥setup.exe」を起動し、インストールを行ってください。詳しくは製品のユーザーズマニュアルを参照してください。

[1] 製品付属 CD では「¥TWFA_UserFirm¥Projects¥HttpClientFirm」フォルダに含まれます。また、製品のサポートページからダウンロード可能な「ユーザーファーム開発用ファイル」にも含まれています。

□ HTTPクライアント・ファームウェアのダウンロード

HTTP クライアント・ファームウェアは製品出荷時にはインストールされていません。使用するためには、以下の手順でファームウェアファイルを製品にダウンロードする必要があります。

1. 製品の電源を切った状態でディップスイッチの 2 番を"ON"にし、フラッシュ書換えモードにします。
2. 製品の電源を入れ LAN ケーブルを接続し、パソコンと通信可能な状態にします。
3. 「M3069FlashWriter」を起動します。[スタート]メニュー→[すべてのプログラム](または、[プログラム])→[テクノウェーブ]から[LANX2219Tools]を選択します。
4. メニュー画面が表示されますので[M3069FlashWriter]のボタンを押してください。
5. [参照]ボタンを押し、製品付属CDの「¥X2219_AdditionalFirm¥TW-IOT」フォルダ、または、ダウンロードファイルの解凍フォルダから「HttpClient.S」ファイルを選択します(図 4)。
6. [書込み]ボタンを押してファームウェアを書き込みます。接続に失敗する場合は「M3069FlashWriter」の「ヘルプ」ボタンを押してオンラインヘルプを参照してください。
7. 続けて「装置番号設定」および「HTTPクライアント・ファームウェアの動作設定」を行うことができます。

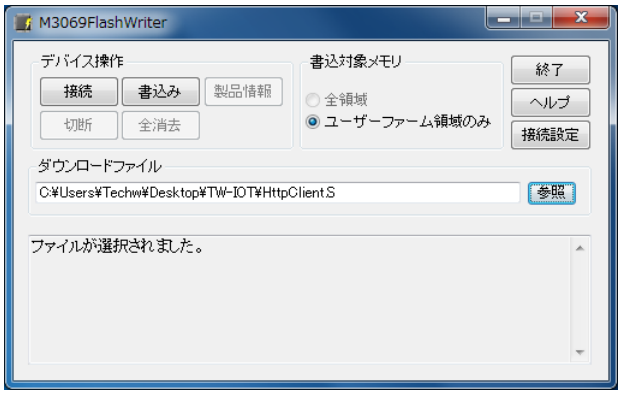


図 4 HTTPクライアント・ファームウェアのダウンロード

□ 装置番号設定

本システムを利用する場合は、必ず装置番号を指定する必要があります。製品出荷時には装置番号「1」が設定されています。複数の製品を同時に制御する場合、それぞれの製品に識別のためのユニークな装置番号を付与してください。

1. 製品の電源を切った状態でディップスイッチ 2 番を"ON"にし、フラッシュ書換えモードにします。
2. 製品の電源を入れ LAN ケーブルを接続し、パソコンと通信可能な状態にします。
3. 「装置番号設定ツール」を起動します。[スタート]メニュー→[すべてのプログラム](または、[プログラム])→[テクノウェーブ]から[LANX2219Tools]を選択します。
4. メニュー画面が表示されますので[装置番号設定ツール]のボタンを押してください。「装置番号設定ツール」が表示されます(図 5)。
5. [接続]ボタンを押して製品に接続します。
6. [新しい番号]に 1～65535 の範囲の数値を入力します。
7. [自動加算]にチェックを入れておくと、書込みを行う度に[新しい番号]が 1 ずつ増加します。
8. [書込み]ボタンを押して装置番号を設定します。
9. 続けて「HTTPクライアント・ファームウェアのダウンロード」および「HTTPクライアント・ファームウェアの動作設定」を行うことができます。

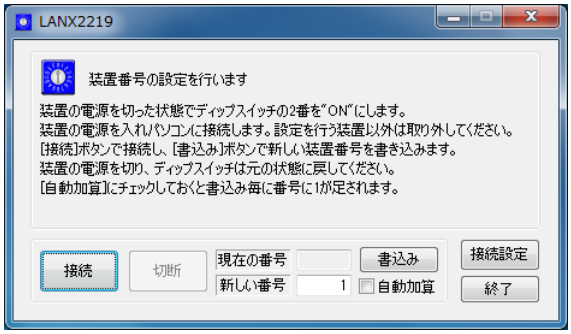


図 5　装置番号設定ツールの起動画面

□ HTTPクライアント・ファームウェアの動作設定

HTTP クライアント・ファームウェアの動作に関する設定は、INI ファイル形式のテキストファイルとして作成し、専用のツール「M3069IniWriter」で製品に書き込みます。

設定ファイルでは、以下で説明するセクションを記述し、機能毎に設定を行います。設定の必要のないセクションはなくてもかまいません。

COMMONセクション

HTTP クライアント・ファームウェア共通の動作設定を行います。

表 3　COMMON セクションのパラメータ

パラメータ名	説明	デフォルト値
daq_enable	"0"とするとサーバーへデータの送信を行いません。	1
ctrl_enable	"0"とするとサーバーからの制御コマンドを受け付けません。WEB ブラウザから出力の制御などを行うことができなくなります。	1
log	"1"とするとシリアルポート 0 からテキスト形式のログが出力されます。	0
time	"1"とすると SERVER セクションの[host_name]で指定するサーバーと時刻同期を行います。 "2"とすると NTP サーバーと時刻同期を行います。NTP サーバーは「LANX2219Tools」の「ネットワーク設定ツール(LANMConfig)」で指定できます。	0

SERVERセクション

通信先となるサーバーに関する動作設定を行います。

表 4　SERVER セクションのパラメータ

パラメータ名	説明	デフォルト値
host_name	通信先サーバーを IP アドレス、または、ドメイン名で指定します。	
host_port	通信先サーバーのポート番号を指定します。	80
daq_url	データのポスト先 URL のドメインを除いた部分を指定します。	
ctrl_url	WEB ブラウザからのコマンドを受け取るスクリプト URL のドメインを除いた部分を指定します。	
proxy_name	プロキシサーバーの URL を指定します。	
proxy_port	プロキシサーバーにアクセスする際のポート番号を指定します。	8080

IDセクション

WEB ページに対する認証に関する動作設定を行います。このセクションで設定された値は Digest 認証が必要なページにアクセスする場合に使用されます。

表 5　ID セクションのパラメータ

パラメータ名	説明	デフォルト値
name	Digest 認証のユーザー名を指定します。	
password	Digest 認証のパスワードを指定します。	

DAQセクション

データのポスト間隔に関する動作設定を行います。

表 6　DAQ セクションのパラメータ

パラメータ名	説明	デフォルト値
min_period	データの最小ポスト間隔をミリ秒単位で指定します。MONITOR_MASK セクションで指定した監視対象ビットに変化がある場合でも、前回のポストから min_period[msec]が経過するまでデータをポストしません。 値は 10～100,000 の範囲で指定します。	1000
max_period	データのポスト間隔をミリ秒単位で指定します。サーバーへ最後にデータをポストしてから MONITOR_MASK セクションで指定した監視対象ビットに変化が無い場合でも、max_period[msec]が経過する毎にデバイスはサーバーへデータをポストします。 値は 10～100,000 の範囲で指定します。	30000

DAQセクションのmin_periodパラメータに″30″[msec]、max_periodパラメータに″100″[msec]を指定し、MONITOR_MASKセクションのIaパラメータに″0x0F″を指定した場合、データは図 6の「送信」のタイミングでデバイスからサーバーに送信されます。

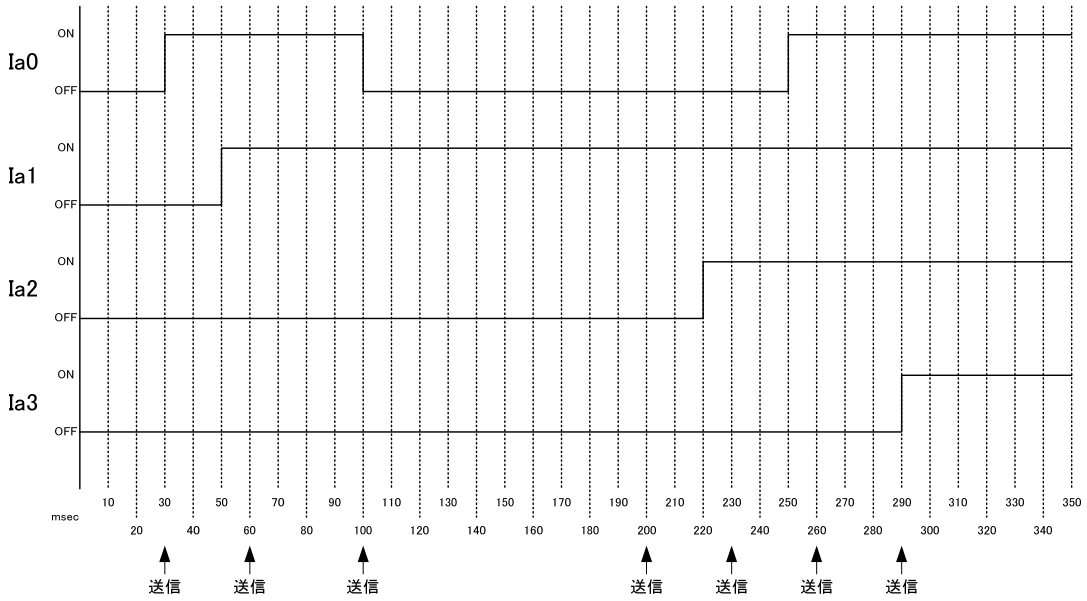


図 6　データ送信タイミング例

MONITOR_MASKセクション

入力ポートの監視に関する動作設定を行います。

表 7　MONITOR_MASK セクションのパラメータ

パラメータ名	説明	デフォルト値
Ia	監視するポートを 0x00-0xFF で指定します。0～7 ビットがそれぞれ入力ポート 0～7 に対応しており、1 が立っているビットに対応するポートが監視対象となります。監視対象のポートに変化を検出すると、デバイスはそのポート情報を含むデータをサーバーへポストします。	0
Ib		0
Ic		0

PWM0 セクション

PWM チャンネル 0 に関する初期設定を記述します。

表 8　PWM0 セクションのパラメータ

パラメータ名	説明	デフォルト値
enable	"1"とすると PWM0 出力が有効になります。	0
frequency	パルスの繰り返し周波数を Hz 単位で指定します。50～1000000 の範囲としてください。	1000
duty	ON デューティを%単位で指定します。	50
phase	初期位相を%単位で指定します。50%は 180°,100%は 360° に相当します。	0
start	"1"とすると起動と同時にパルスを出力します。	0

PWM1/PWM2 セクション

PWM0 セクションと同様に PWM チャンネル 1 と 2 の初期設定を行います。PWM1、PWM2 はハードウェアカウンタと同じハードウェアを使用します。ハードウェアカウンタのチャンネル 1、または、2 を使用すると、それぞれ PWM1、PWM2 が使用できなくなります。

CLK1 セクション

ハードウェアカウンタのチャンネル 1 に関する初期設定を行います。ハードウェアカウンタ 1 を有効にすると PWM1 は禁止されます(ハードウェアカウンタが優先されます)。

表 9　CLK1 セクションのパラメータ

パラメータ名	説明	デフォルト値
enable	"1"とするとハードウェアカウンタ 1 を有効にします。	0
start	"1"とすると起動と同時にカウントを開始します。	0

CLK2 セクション

ハードウェアカウンタのチャンネル 2 に関する初期設定を行います。ハードウェアカウンタ 2 を有効にすると PWM2 は禁止されます(ハードウェアカウンタが優先されます)。

表 10　CLK2 セクションのパラメータ

パラメータ名	説明	デフォルト値
enable	"1"とするとハードウェアカウンタ 2 を有効にします。	0
mode	ハードウェアカウンタ 2 のカウントモードを設定します。"2"とすると 90° 位相差の 2 相パルスをカウントするモードになります。CLK1 端子に B 相、CLK2 端子に A 相を接続して使用します。	0
start	"1"とすると起動と同時にカウントを開始します。	0

PC0 セクション

パルスカウンタ(ソフトウェアカウンタ)のチャンネル 0 に関する初期設定を行います。

表 11　PC0 セクションのパラメータ

パラメータ名	説明	デフォルト値
mode	パルスカウンタのチャンネル 0、1 のカウントモードを指定します。 "2"とするとパルスカウンタ 1 との組み合わせで 90° 位相差の 2 相パルスをカウントするモードになります。この場合、PC0、PC1 端子に A 相信号、Ib6 端子に B 相信号を入力してください。 "2"以外ではチャンネル 0、1 とも、単相カウントになります。	1
start	"1"とすると起動と同時にカウントを開始します。2 相カウントの場合はチャンネル 1 も同時に開始されます。	0

PC1セクション

パルスカウンタ(ソフトウェアカウンタ)のチャンネル 1 に関する初期設定を行います。

表 12　PC1 セクションのパラメータ

パラメータ名	説明	デフォルト値
start	"1"とすると起動と同時にカウントを開始します。カウントモードは PC0 セクションの設定によって決まります。PC0 セクションの mode が 2 の場合は 2 相カウント、それ以外では単相カウントとなります。	0

PC2 セクション

パルスカウンタ(ソフトウェアカウンタ)のチャンネル 2 に関する初期設定を行います。

表 13　PC2 セクションのパラメータ

パラメータ名	説明	デフォルト値
mode	パルスカウンタのチャンネル 2、3 のカウントモードを指定します。 "2"とするとパルスカウンタ 3 との組み合わせで 90° 位相差の 2 相パルスをカウントするモードになります。この場合、PC2、PC3 端子に A 相信号、Ib7 端子に B 相信号を入力してください。 "2"以外ではチャンネル 2、3 とも、単相カウントになります。	1
start	"1"とすると起動と同時にカウントを開始します。2 相カウントの場合はチャンネル 3 も同時に開始されます。	0

PC3 セクション

パルスカウンタ(ソフトウェアカウンタ)のチャンネル 3 に関する初期設定を行います。

表 14　PC3 セクションのパラメータ

パラメータ名	説明	デフォルト値
start	"1"とすると起動と同時にカウントを開始します。カウントモードは PC2 セクションの設定によって決まります。PC2 セクションの mode が 2 の場合は 2 相カウント、それ以外では単相カウントとなります。	0

初期設定ファイルの例

```
[COMMON]
time=1          ;通信先サーバーと時刻同期を行う

[SERVER]
host_name="192.168.0.10"    ;通信先サーバーの IP アドレス
daq_url="/device-post.php" ;データのポスト先 URL のドメインを除いた部分
ctrl_url="/device-get.php" ;コマンドを受信するスクリプト URL のドメインを除いた部分

[DAQ]
max_period=2000 ;データをサーバーへポストする最大間隔
min_period=10   ;データをサーバーへポストする最小間隔

[MONITOR_MASK]
Ic=0xFF ;Ic 端子の任意のポートで状態変化を検出するとデータをサーバーへポスト

[PC0]
start=1 ;単相モードでカウント開始

[PC1]
start=1 ;単相モードでカウント開始

[PC2]
start=1 ;単相モードでカウント開始

[PC3]
start=1 ;単相モードでカウント開始
```

HTTPクライアント・ファームウェア動作設定の書込み

1. 製品の電源を切った状態でディップスイッチの 2 番を"ON"にし、フラッシュ書換えモードに設定します。
2. 製品の電源を入れ LAN ケーブルを接続し、パソコンと通信可能な状態にします。
3. 「M3069IniWriter」を起動します。[スタート]メニュー→[すべてのプログラム]（または、[プログラム]）→[テクノウェーブ]から[LANX2219Tools]を選択します。
4. メニュー画面が表示されますので[M3069IniWriter]のボタンを押します。
5. [編集]画面で設定ファイルを編集します。[ファイルを開く]ボタンで既に作成したファイルを開くこともできます。
6. [デバイスと接続]ボタンを押して製品と接続します。接続に失敗する場合は「M3069IniWriter」のツールバーから「ヘルプ」→「ヘルプ」を選択してオンラインヘルプを参照してください。
7. [操作対象ブロック]は"EB1"を選択してください。
8. [デバイスへ書込み]ボタンを押して設定を書き込みます。
9. 続けて「HTTPクライアント・ファームウェアのダウンロード」および「装置番号設定」を行うことができます。

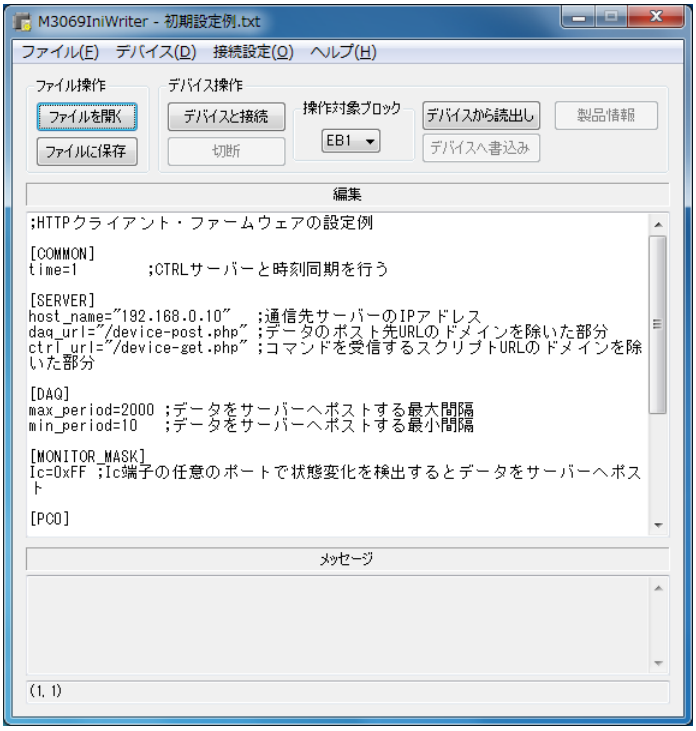


図 7　初期設定の書込み

□ ディップスイッチの設定

HTTP クライアント・ファームウェアを使用する場合、製品のディップスイッチを以下のように設定した状態で製品を起動します。

表 15　ディップスイッチの設定

番号	設定
1	ON
2	OFF

4. サーバーの準備

□ サーバー設定の概要と注意事項

TW-IOTシステムには、表 16に示すソフトウェアがインストールされたサーバーが必要です。また、パソコンをサーバーとして使用する場合は、パソコンのIPアドレスを固定とする必要があります。

表 16 必要なソフトウェアと対象バージョン

ソフトウェア	対象バージョン
Apache	2.4 以降
MySQL	5.6 以降
PHP	5.3 以降

□ Windows® OSをサーバーとして使用する場合

Windows OSをサーバーとして使用する場合、XAMPPパッケージ(バージョン 1.8.3 以降)を用いると表 16のソフトウェアを一括でインストールすることができます。

XAMPPパッケージは、「https://www.apachefriends.org/jp/index.html」から「Windows向けXAMPP」をダウンロードし、画面の指示に従ってインストールを行ってください。

1. 製品付属 CD の「¥X2219_AdditionalFirm¥TW-IOT¥htdocs」フォルダ、または、ダウンロードファイルの「¥htdocs」フォルダを XAMPP がインストールされたフォルダ(デフォルトで「C:¥xampp」)にコピーしてください。
2. [スタート]メニュー→[すべてのプログラム](または、[プログラム])→[XAMPP]から[XAMPP Control Panel]を選択します(図 8)。

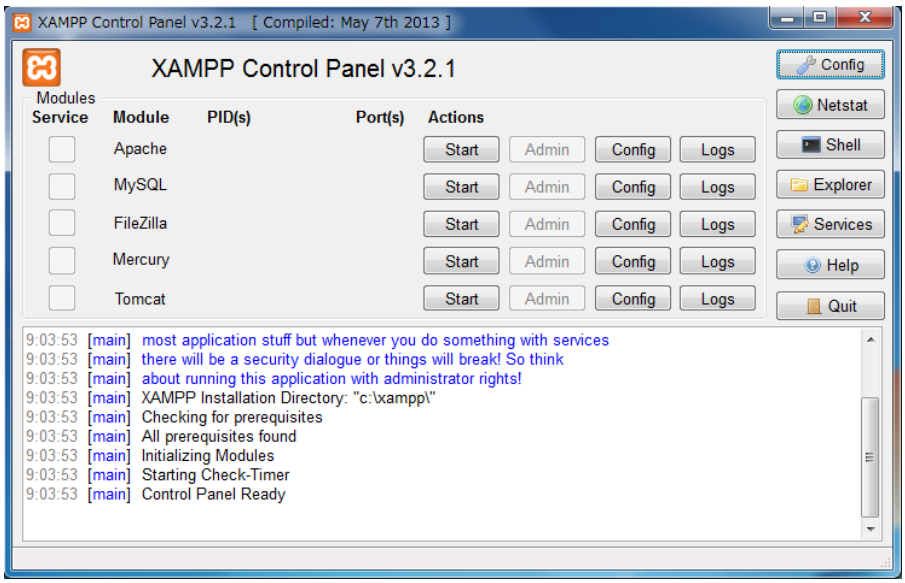


図 8 XAMPP Control Panel起動画面

3. 「Apache」および「MySQL」の[Start]ボタンを押してサーバー機能をスタートさせます。
4. WEBブラウザから「http://localhost/phpmyadmin」にアクセスし、メニューから[インポート](または[その他]→[インポート])を選択してください。

Windows は米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。

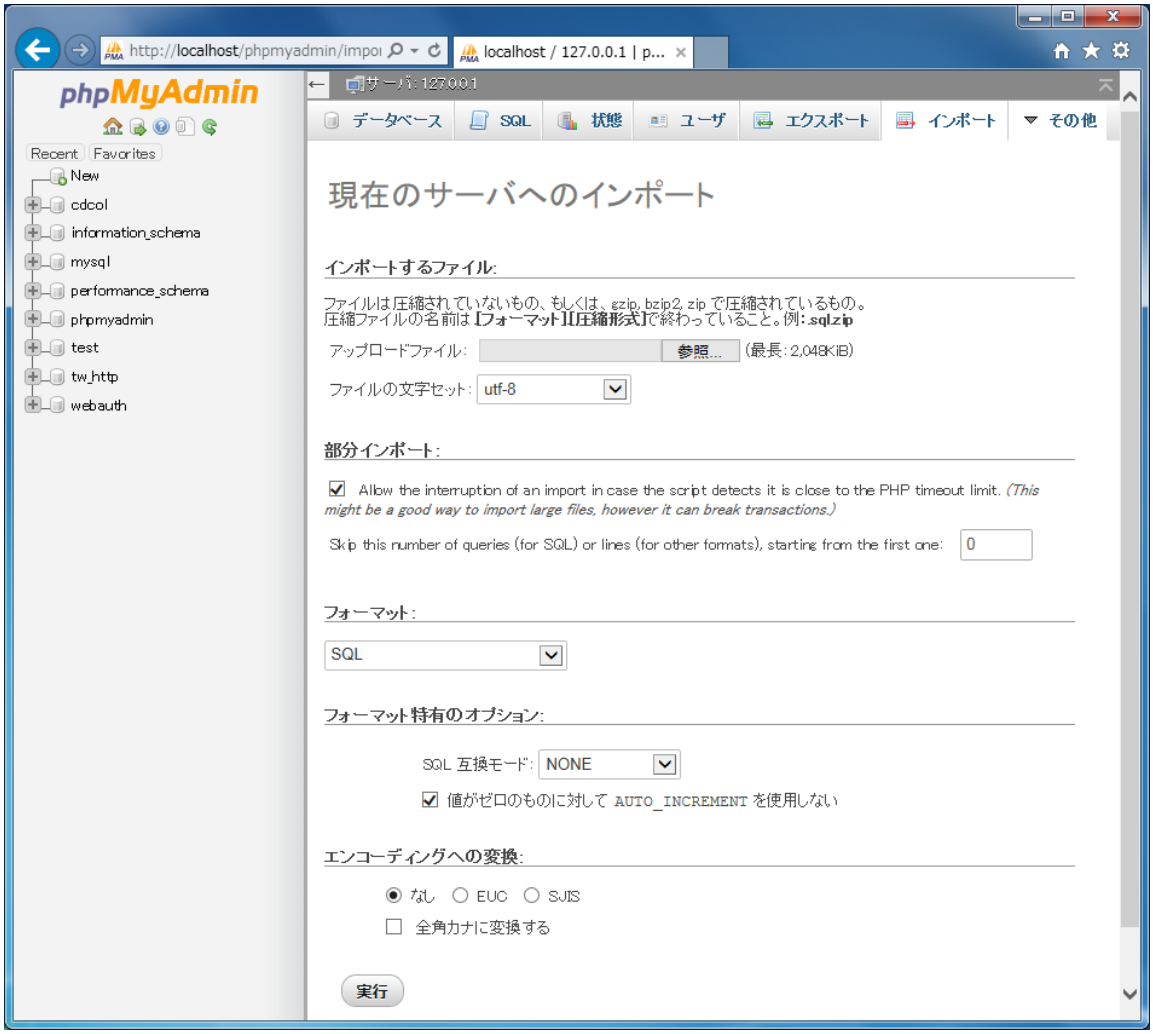


図 9　インポートファイルの選択画面

5. 「インポートするファイル」の［参照］（または［ファイルを選択］）ボタンを押して、「¥htdocs¥tw_http.sql」を選択し、[実行]ボタンを押してください。
6. MySQL の root アカウントにパスワードを設定した場合は「¥htdocs¥constants.php」ファイル内の[$pass]の部分を変更してください。

```
$user = "root";
$pass = "";
```

Windows OSがインストールされたパソコンのIPアドレスを固定する

1. Windows 7 の場合[コントロールパネル]→[ネットワークの状態とタスクの表示](アイコン表示の場合は[ネットワーク共有センター])→[アダプター設定の変更]の順にクリックします。
2. 使用する接続を右クリックし、プロパティを開きます。Windows 7 の場合「インターネット プロトコル バージョン 4(TCP/IPv4)」を選択しプロパティを表示します。
3. 表示された画面の「次の IP アドレスを使う」を選択し、「IP アドレス」に"192.168.0.10"、「サブネットマスク」に"255.255.255.0"、「デフォルトゲートウェイ」に"192.168.0.1"、「優先 DNS サーバー」に"192.168.0.1"と入力します。入力が終わったら「OK」ボタンを押して画面を閉じます。

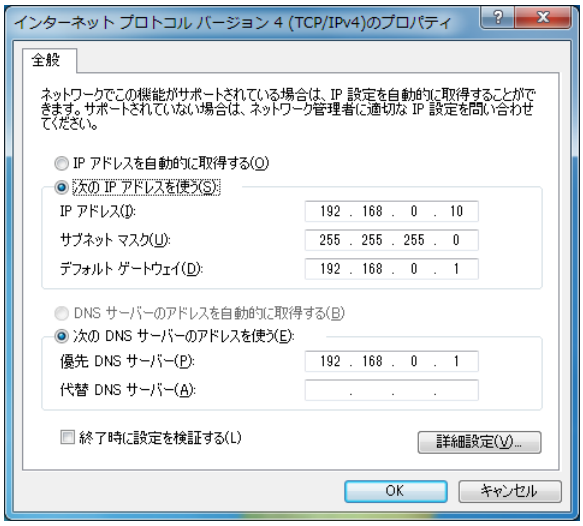


図 10 固定 IP アドレスの設定画面(Windows 7)

□ Linux® OSをサーバーとして使用する場合

Linux OSをサーバーとして使用する場合、LAMPPパッケージを用いると表 16のソフトウェアを一括でインストールすることができます。

本マニュアルでは「Ubuntu® 14.04 LTS」を使用した場合の手順を記載しております。

1. 製品付属 CD の「¥X2219_AdditionalFirm¥TW-IOT¥ServerFiles」ディレクトリを任意の場所にコピーします。
2. [Ctrl]+[Alt]+[T]キーを押してターミナルを起動します。
3. 「sudo su」コマンドで管理者権限に切り替えます。

```
>sudo su
```

4. 以下のコマンドを入力し、「tasksel」をインストールします。

```
#apt-get install tasksel
```

5. インストールが終了したら、以下のコマンドを入力し「tasksel」を起動します（図 11）。

```
#tasksel
```

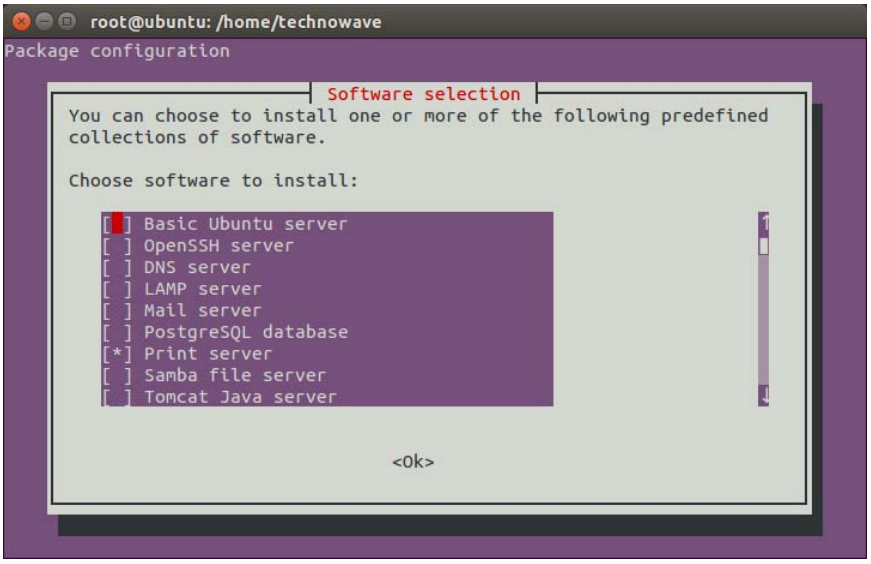


図 11 tasksel起動画面

6. [カーソルキー（矢印キー）]で「LAMP server」まで移動し、[スペースキー]で「LAMP Server」を選択します（選択されると[*]が表示されます）。[Tab キー]で「<Ok>」へ移動し[Enter キー]で「LAMP server」のインストールを開始します。
7. インストールの途中で、MySQLで使用する「root」ユーザーのパスワード設定に関する画面（図 12）が表示されますので、必要な場合はパスワードを設定してください。パスワードを設定しない場合は空欄のまま「<Ok>」を押してください。

Linux は、Linus Torvalds 氏の日本およびその他の国における登録商標または商標です。
Ubuntu は、Canonical Ltd.の登録商標です。

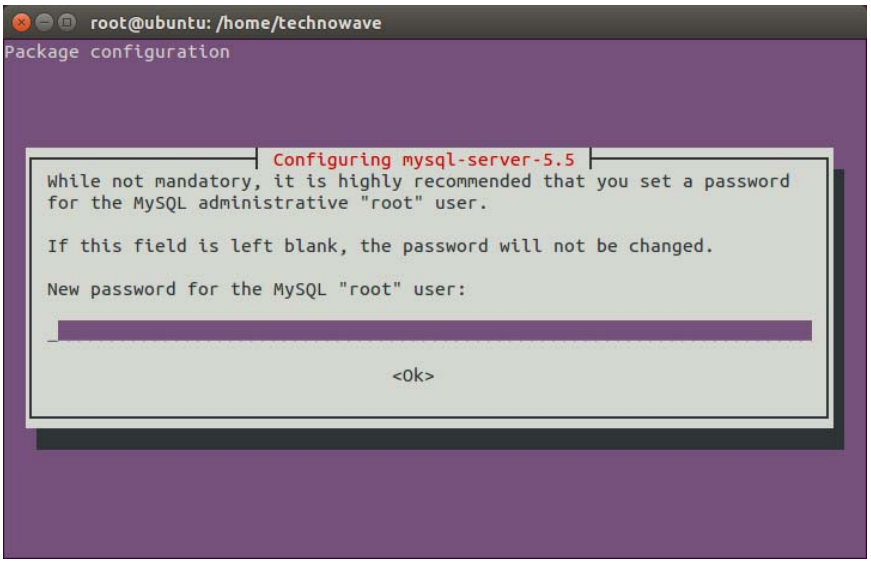


図 12 パスワード入力画面

8. インストールが終了したら、ターミナル上でコピーした「¥ServerFiles」ディレクトリ内に移動し、以下のコマンドを入力して本システムに必要なファイルのインストールを行います。

```
#./install.sh
```

9. デフォルトではドキュメントルートなどに必要なファイルが配置されます。ファイルの配置を、ドキュメントルート以外にしたい場合や、ドキュメントルートが表示とは異なる場合などは、必要に応じてディレクトリの絶対パスを入力してください。変更を行わない場合は空白の状態で[Enter キー]を押してください。

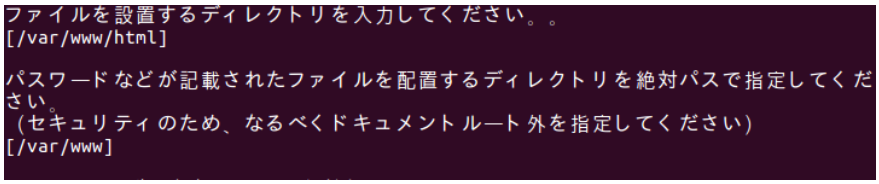


図 13 ドキュメントルート指定画面

10. 「MySQLのユーザー名」に"root"、「MySQLのパスワード」と「Enter password」に図 12で設定したパスワードを入力してください。図 12でパスワードを設定していない場合は未入力の状態で[Enterキー]を押してください。

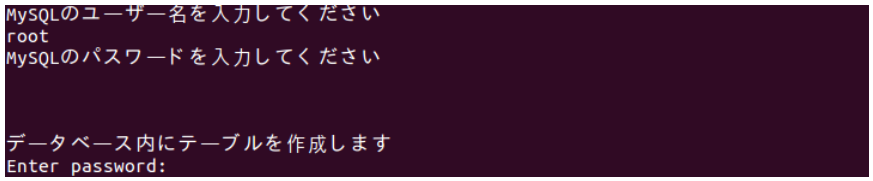


図 14 MySQLユーザー名/パスワード入力画面

Linux OSがインストールされたパソコンのIPアドレスを固定する

以下のコマンドを入力して、IP アドレスを固定します。

```
#/sbin/ifconfig eth0 192.168.0.10
```

"eth0"は IP アドレスを固定したいインタフェースに合わせて変更してください。

5. XMLファイルの準備

サーバーはデバイスから送信されたデータをデータベースに蓄積し、そのデータを WEB ブラウザに転送します。しかし、サーバーに送信されたデータはそのままの状態では単なるバイト列でしかありません。そこで、このバイト列を適当な位置で切り分け、10 進数やブール値などへ変換、さらに、変換した各項目へ独自の名前を付けるなど、変換ルールを定義する必要があります。本システムではこれらのルールを定義した XML ファイルを用意します。

XML ファイルは制御するデバイスに合わせて用意する必要がありますが、複数のデバイスで同様の制御を行う場合は、1 つのファイルを共有することができます。

作成したXMLファイルはサーバー内のドキュメントルート以下に配置してください。「4.サーバーの準備」で必要なファイルをコピー、または、インストールしている場合、ドキュメントルートの「¥views」フォルダにサンプルプログラム「sample.xml」およびLANX-I2219 の全データの変換ルールを記述した「I2219.xml」が収められていますので、必要に応じて参照してください。

□ XMLファイルの作成

リスト 1はアナログ入力「AD0」とアナログ出力「DA0」の値を取得するXMLの例です。

リスト 1 デバイスのアナログ入出力のデータを電圧値に変換

```
<?xml version="1.0" encoding="UTF-8"?>

<ArryStruct> ...①
  <field name="ad0"> ...②
    <type scale="0.0048828125">unsigned</type> ...③
    <address>4</address> ...④
    <length>10</length> ...⑤
    <direction>r</direction>
    <unit>V</unit>
    <description>"LANX-I2219 のアナログ入力チャンネル 0"</description>
  </field>

  <field name="da0">
    <type scale="0.01953125">unsigned</type>
    <address>42</address>
    <length>8</length>
    <direction>rw</direction>
    <unit>V</unit>
    <description>"LANX-I2219 のアナログ出力チャンネル 0"</description>
  </field>
</ArryStruct>
```

① XML のルート要素です。ルート要素内に、デバイスからサーバーへ送信されたデータを個々の値に変換するルールを設定します。

② <field>要素内に変換ルールを記述します。一つの<ArryStruct>内に<field>要素は複数存在しても問題ありません。<field>要素の[name]属性には、必ずユニークな値を指定してください。指定された値は変換結果を取り出す際に連想配列のキーとなります。

③ <type>要素には、値を取得する際のデータ型を指定します。また、<type>要素は属性を指定して、値を変換することができます。例えば、AD0 はそのままでは 10 ビットの値ですが、[scale]属性を使用して"0.0048828125"(=5/2^{10})を指定することによって 0V～5Vの電圧値に変換して取得することができます。 [scale]以外の属性については表 18を参照してください。

④ <address>要素には、値が格納されているデータの先頭アドレスをバイト単位で指定します。AD0 の値を取得する場合、値はデータのアドレス 4 から始まるので、<address>に"4"を指定します。各値の先頭アドレスは表 29(32ページ)を参照してください。

⑤ <length>要素には、データのサイズをビット単位で指定します。AD0 の値を取得する場合、値はアドレス 4 のMSBから 10 ビット分格納されているので<length>には"10"を指定します。各値のデータサイズは「7.制御方法」を参照してください。

表 17　XMLに設定できる要素

<ArryStruct>

要素	説明
<field>	各要素の親要素です。[name]属性を使用して独自の名前を設定することができます。
<type> </type>	"signed"（符号付整数）、"unsigned"（符号無し整数）、"fixed"（固定小数点数）、"float"（単精度浮動小数点数）、"boolean"（真偽値）のいずれかを指定します。この項目は必須です。
<address> </address>	値が格納されているデータの先頭アドレスをバイト単位で指定します。この項目は必須です。
<length> </length>	データのサイズをビット単位で指定します。この項目は<type>で"boolean"、および、"float"を指定する場合を除いて必須となります。
<direction> </direction>	<address>で指定されたアドレスの書込み許可を指定します。"r"を指定した場合は読出しのみとなり、"rw"を指定した場合は読み書き可能となります。指定が無い場合は"rw"となります。
<unit> </unit>	値の単位を指定します。データの変換には影響しません。
<description> </description>	値の説明を記入します。データの変換には影響しません。
</field>	

</ArryStruct>

表 18　<type>要素に設定できる属性

<type

属性	デフォルト値	説明
scale	1	値に掛け合わせるスケールを指定します。<type>が"signed"、"unsigned"、"fixed"、"float"の場合に指定できます。
offset	0	値に足し合わせるオフセットを指定します。<type>が"signed"、"unsigned"、"fixed"、"float"の場合に指定できます。
array	false	"true"を指定すると値を配列として取得します。<address>で指定された先頭アドレスから１要素<length>サイズで、要素数[elements]個の配列を取得します。
elements	1	配列の要素数を指定します。[array]が"true"の場合に指定できます。
incremental	false	"true"を指定すると１つ前の値からの増分値を取得します。
bit_address	0	値の読出しビット位置を指定します。<type>が"boolean"の場合に指定できます。
low_active	false	"true"を指定すると"boolean"で取得した値を反転し、"0"を"true"、"1"を"false"に変換します。
radix_point	8	<type>で"fixed"を指定した場合に、最上位桁からの小数点の位置を指定します。

>

□　XMLファイルの作成例

センサーなどの外部機器を接続した場合、XMLによって変数に独自の名前を付け、値の変換などを自動で処理させることができます。リスト 2はAD0 に湿度センサー「CHS-GSS」(TDK社)を接続し、PC0 にパルス出力式人感センサーを接続した場合のサンプルです。

リスト 2　湿度センサーの入力値と人感センサーのパルスカウント値を取得

```
<?xml version="1.0" encoding="UTF-8"?>

<ArryStruct>
  <field name="Relative Humidity">
    <type scale="0.1220703125" offset="-25">unsigned</type>
    <address>4</address>
    <length>10</length>
    <direction>r</direction>
    <description>"AD0 に接続された湿度センサーの値"</description>
  </field>

  <field name="Motion Sensor">
    <type>unsigned</type>
    <address>12</address>
    <length>32</length>
    <direction>rw</direction>
    <description>"PC0 に接続された人感センサーが反応した回数"</description>
  </field>
</ArryStruct>
```

6. WEBページの準備

WEBブラウザからデータの閲覧や制御を行う場合、制御するデバイスに対応したWEBページが必要となります。作成されたWEBページファイルはサーバー内のドキュメントルート以下に配置してください。「4.サーバーの準備」で必要なファイルをコピー、または、インストールされている場合、ドキュメントルートの「¥views」フォルダにサンプルページが収められていますので、必要に応じて参照してください。

□ WEBページの作成準備

デバイスの制御を行う際に必要なライブラリ

表 19はWEBページからデバイスの制御を行う際に必要となるライブラリです。

WEBページからデバイスの制御を行うには「tw_xml.js」(以下、tw_xmlライブラリ)が必要です。tw_xmlライブラリを使用することで、JavaScript®を使用したデータの送受信、および、XMLに基づいたデータの変換を行うことができます。

tw_xmlライブラリの動作にはjQueryが必要です。tw_xmlライブラリ、および、jQueryを使用するには、WEBページ内〈script〉の先頭にリスト 3のように記述してください。また、tw_xmlライブラリはWEBページと同じディレクトリに配置してください。jQueryについては、インターネット経由で読み込まれます。

表 19 WEBページからデバイスの制御を行う際に必要なライブラリ

ライブラリ名	説明	付属 CD 内の格納フォルダ
tw_xml.js	データの送受信、および、データの変換を行う専用ライブラリ	「X2219_AdditionalFirm¥TW-IOT¥htdocs¥views」フォルダ
jQuery ライブラリ	tw_xml ライブラリで必要な外部ライブラリ	インターネット経由で読み込まれます

リスト 3 ライブラリの読込み指定

```
<script type="text/javascript" src="//code.jquery.com/jquery-latest.min.js"></script>
<script type="text/javascript" src="tw_xml.js"></script>
```

JavaScript は、Oracle Corporation およびその子会社、関連会社の米国およびその他の国における登録商標です。

□ デバイスとXMLとの関連付け

特定のデバイスに対してデータの送受信を行う場合、事前にそのデバイスの装置番号と対応する XML とを関連付けておく必要があります。また、複数のデバイスを操作する場合は、全ての装置番号に対して、対応する XML との関連付けを行う必要があります。

デバイスとXMLとの関連付けを行うには、表 20の関数を使用します。

表 20 デバイスとXMLの関連付けを行う関数

関数名	説明
techw.addDevice()	デバイスと XML との関連付けを行います。

表 21 techw.addDevice() の関数宣言

関数宣言
techw.addDevice(xmlUrl, deviceId)

リスト 4 デバイスと XML の関連付けの例

```
techw.addDevice("./sample.xml", 1); //装置番号 1 のデバイスと sample.xml との対応関係を追加
```

□ データの取得

WEBブラウザからサーバーに蓄積されたデータを取得するには、表 22の関数を使用します。また、表 23はデータ取得に関するサンプルとして用意されているプログラムです。

表 22 データの取得に使用する関数

関数名	説明
techw.getData()	サーバーに蓄積された最新のデータを取得します。
techw.getRecord()	サーバーに蓄積されたデータから特定期間のデータを取得します。

表 23 データの取得に関するサンプルプログラム

ファイル名	説明
sample.html	デバイスからサーバーに送信された最新データをリアルタイムに表示します
sampleRecord.html	サーバー内のデータベースに蓄積された過去のデータを表示します

最新データの取得

WEB ブラウザからサーバーに蓄積された最新のデータを取得するには techw.getData() 関数を使用します。

表 24 techw.getData() の関数宣言

関数宣言
techw.getData(url, callback, error)

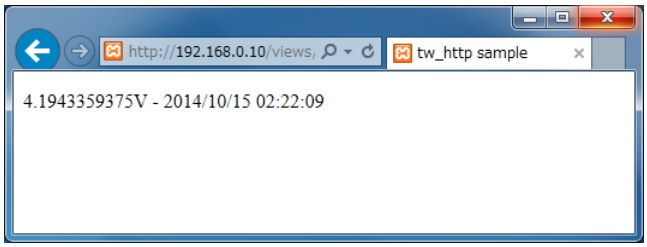
データの読出し、および、変換が正常に終了した場合、callback 引数に指定されたコールバック関数が呼び出されます。その際、コールバック関数の引数には変換されたデータとデータの取得時間が渡されます。また、エラーが発生した場合、error 引数に指定されたエラー関数が呼び出されます。

techw.getData() 関数は、初回呼び出し以外はサーバー側でデータが更新されるの待ってデータを取得するので、サーバーに不要な負荷をかけることはありません。

リスト 5 sample.xml を用いたサーバーから最新データの取得例

```
deviceId = 1;

window.onload = function(){
  techw.addDevice("./sample.xml", deviceId); //装置番号 1 のデバイスと XML の対応関係を追加
  techw.getData("../client-get.php", callback, error); //データを取得
}

function callback(data, time){
  //得られた結果を表示
  document.getElementById('result').innerHTML = data[deviceId]["ad0"] + "V - " + time;
  //続けて最新データの取得
  techw.getData("../client-get.php", callback, error);
}

function error(msg, code){ //エラーハンドラ
  document.getElementById('error').innerHTML = msg + ":" + code;
}
```

techw.getData() 関数を使用したサンプルプログラム「sample.html」の表示例



特定期間のデータの取得

サーバーに保存されたデータの中から、特定期間のデータを取得するには techw.getRecord() 関数を使用します。

表 25 techw.getRecord() の関数宣言

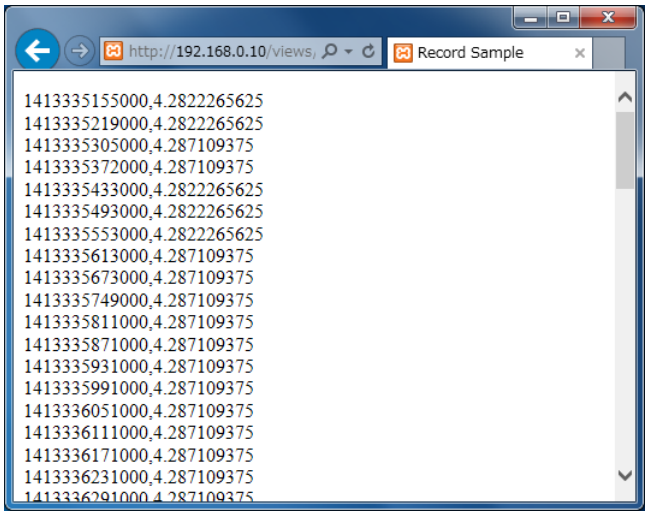
関数宣言
techw.getRecord(url, callback, error, start, end, interval, average, aveTime)

データの読出し、および、変換が正常に終了した場合、callback 引数に指定されたコールバック関数が呼び出されます。その際、コールバック関数の引数には変換されたデータが渡されます。また、エラーが発生した場合、error 引数に指定されたエラー関数が呼び出されます。

リスト 6 sample.xml を用いたサーバーから特定期間のデータの取得例

```
deviceId = 1;

var now = new Date();
var end = getStrDate(now); //現在の日時

var yesterday = new Date(Date.now() - 24*60*60*1000);
var start = getStrDate(yesterday); //24 時間前の日時

function getStrDate(date){ //日時を文字列で返す
  return date.getFullYear() + "/" + (now.getMonth()+1) + "/" + date.getDate()
  + " " + date.getHours() + ":" + date.getMinutes() + ":" + date.getSeconds();
}
window.onload = function(){
  techw.addDevice("./sample.xml", deviceId);  //装置番号 1 のデバイスと XML の対応関係を追加
  techw.getRecord("../client-get.php", callback, error, start, end, 60); //データを取得
}

function callback(data){
  for(var i=0; i<data[deviceId]["ad0"].length; i++){
    //データを表示
    document.getElementById('result').innerHTML += data[deviceId]["ad0"][i] + "<br>";
  }
}

function error(msg, code){ //エラーハンドラ
  document.getElementById('result').innerHTML = msg + ":" + code;
}
```

techw.getRecord() 関数を使用したサンプルプログラム「sampleRecord.html」の表示例



□ デバイスの操作

サーバーを経由してデバイスの出力ポートを操作するには表 26の関数を使用します。また、表 27は出力ポートの操作に関するサンプルとして用意されているプログラムです。

表 26 デバイスの操作に使用する関数

関数名	説明
techw.postData()	サーバーを経由してデバイスへデータを送信します。

表 27 データの取得に関するサンプルプログラム

ファイル名	説明
samplePost.html	サーバーを経由してデバイスの DA0 出力を操作します

表 28 techw.postData() の関数宣言

関数宣言
techw.postData(url, callback, error, data, dataName, deviceId, index, mask)

データがサーバーを経由してデバイスへ正常に転送された場合、callback 引数に指定されたコールバック関数が呼び出されます。一定時間経過してもデータが転送されない場合、error 引数に指定されたエラー関数が呼び出されます。また、<direction>に"r"が指定されているデータを書き換えようとした場合もエラー関数が呼び出されます。

リスト 7 sample.xml を用いたデバイスの操作例

```
deviceId = 1;

window.onload = function(){
  techw.addDevice("./sample.xml", deviceId); //装置番号 1 と XML の対応関係を追加
}

function postData(){
  var data = document.form.post.value; //フォームの値を取得
  //データを送信
  techw.postData("../client-post.php", callback, error, [data], "da0", deviceId);
}

function callback(msg){
  var now = new Date();
  document.getElementById('result').innerHTML = now.getFullYear()
  + "/"+ (now.getMonth() + 1) + "/" + now.getDate() + " " + now.getHours() + ":"
  + now.getMinutes() + ":" + now.getSeconds() + " - " + "送信成功"; //送信成功時の表示
}

function error(msg, code){
  document.getElementById('result').innerHTML = msg + ":" + code; //エラー発生時の表示
}
```

techw.postData() 関数を使用したサンプルプログラム「samplePost.html」の表示例

7. 制御方法

□ データレジスタのアドレスマップ

HTTPクライアント・ファームウェアが書き込まれたデバイスは、全てのI/Oポートの情報が入った表29のような構造体を保持しており、その情報をサーバーへサンプル 1のようなバイト列として送信します。「R/W」の項目は"R"が読出しのみ、"R/W"が読み書き可能なことを表しています。また、2 バイト以上のデータについては最上位バイトから格納されています。

表 29　データレジスタのアドレスマップ

アドレス	機能	R/W
0	Ia	R
1	Ib	R
2	Ic	R
3	N/A	-
4	AD0	R
5		R
6	AD1	R
7		R
8	AD2	R
9		R
10	AD3	R
11		R
12	PC0	R/W
13		R/W
14		R/W
15		R/W
16	PC1	R/W
17		R/W
18		R/W
19		R/W
20	PC2	R/W
21		R/W
22		R/W
23		R/W
24	PC3	R/W
25		R/W
26		R/W
27		R/W
28	TimerCnt0	R/W
29		R/W
30	TimerCnt1	R/W
31		R/W
32	TimerCnt2	R/W
33		R/W
34	N/A	-
35	UserStatus	R/W
36	Od_Mask	R/W
37	Od	R/W
38	Oe_Mask	R/W
39	Oe	R/W
40	Of_Mask	R/W
41	Of	R/W
42	DA0	R/W
43	DA1	R/W
44	PwmFreq0	R/W
45		R/W
46		R/W
47		R/W
48	PwmDuty0	R/W
49		R/W
50		R/W
51		R/W
52	PwmPhase0	R/W
53		R/W
54		R/W
55		R/W
56	PwmFreq1	R/W
57		R/W
58		R/W
59		R/W
60	PwmDuty1	R/W
61		R/W
62		R/W
63		R/W
64	PwmPhase1	R/W
65		R/W
66		R/W
67		R/W
68	PwmFreq2	R/W
69		R/W
70		R/W
71		R/W
72	PwmDuty2	R/W
73		R/W
74		R/W
75		R/W
76	PwmPhase2	R/W
77		R/W
78		R/W
79		R/W
80	NumOfPulse0	R/W
81		R/W
82		R/W
83		R/W
84	NumOfPulse1	R/W
85		R/W
86		R/W
87		R/W
88	NumOfPulse2	R/W
89		R/W
90		R/W
91		R/W
92	N/A	-
93	TimerStart	R/W
94	N/A	-
95	PCStart	R/W

サンプル 1　デバイスから送信されるバイト列(16 進表記)

```
0000000076C0E1C0E800E880000000000000000000000000000000000000000000000000FF00FF00F
F000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000
```

□ デジタル入出力

入力端子の状態はアドレスマップの Ia(アドレス 0)、Ib(アドレス 1)、Ic(アドレス 2)レジスタから読み出します。

Ia0～Ia7 端子を例にとると、各入力端子の状態は図 15のように格納され、"ON"の端子と対応するビットは"1"、"OFF"の端子と対応するビットは"0"となります。

データビット	7	6	5	4	3	2	1	0
	Ia7	Ia6	Ia5	Ia4	Ia3	Ia2	Ia1	Ia0

図 15 Ia レジスタ

出力端子の状態を変更するにはアドレスマップの Od_Mask(アドレス 36)、Od(アドレス 37)、Oe_Mask(アドレス 38)、Oe(アドレス 39)、Of_Mask(アドレス 40)、Of(アドレス 41)レジスタに書込みを行います。

Od0～Od7 端子を例にとると、各出力端子は図 16のように格納され、"1"を書き込んだビットと対応する端子は"ON"に、"0"を書き込んだビットと対応する端子は"OFF"になります。

Od_Mask レジスタの各ビットは、Od レジスタの各ビットと対応するマスク値となっています。マスク値が"0"の場合、対応する出力端子は書込みの影響を受けません。例えば Od_Mask レジスタに"0x0f"、Od レジスタに"0xff"を書き込んだ場合、Od0～Od3 の端子は"ON"となりますが、Od4～Od7 の端子は対応するマスクビットが"0"となっているため書込み前の状態を維持します。

データビット	7	6	5	4	3	2	1	0
	Od7	Od6	Od5	Od4	Od3	Od2	Od1	Od0

図 16 Od レジスタ

データビット	7	6	5	4	3	2	1	0
	Od7 Mask	Od6 Mask	Od5 Mask	Od4 Mask	Od3 Mask	Od2 Mask	Od1 Mask	Od0 Mask

図 17 Od_Mask レジスタ

□ アナログ入出力

アナログ入力値を取得するにはアドレスマップの AD0(アドレス 4、アドレス 5)、AD1(アドレス 6、アドレス 7)、AD2(アドレス 8、アドレス 9)、AD3(アドレス 10、アドレス 11)レジスタを読み出します。

AD0 端子を例にとると、AD変換結果は図 18のように格納されます。入力電圧と読み出された値の関係は表 30のようになります。

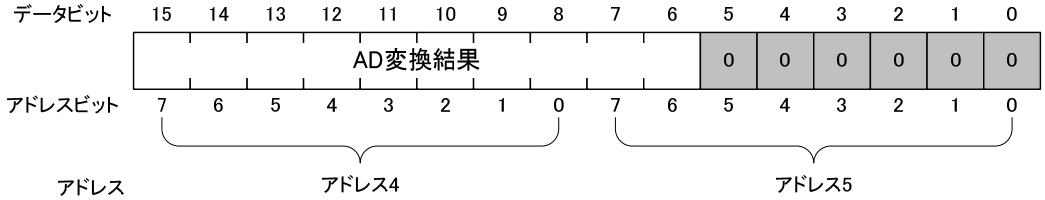


図 18　AD0 レジスタ

表 30　アナログ入力電圧と変換結果の関係

入力電圧値([V])	読み出される値
5-LSB	65472
2.5	32768
0	0

・LSB = 5 / 1024 [V]
・表は理論値を示しています。

アナログ出力値を変更するにはアドレスマップの DA0(アドレス 42)、DA1(アドレス 43)レジスタに書込みを行います。

DA0 端子を例にとると、出力値は図 19のように格納されます。表 31は書込み値と出力電圧の関係です。

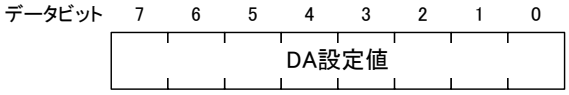


図 19　DA0 レジスタ

表 31　アナログ入力電圧と変換結果の関係

書込み値	出力電圧([V])
255	5-LSB
128	2.5
0	0

・LSB = 5 / 256 [V]
・表は理論値を示しています。

□ PWM出力

PWM パルスは、チャンネル 0～2 の最大 3 チャンネルが出力可能です。PWM 出力には製品搭載マイコンに内蔵されている 16 ビットタイマという機能を利用します。約 48Hz～1MHz までのパルス出力が可能で、デューティや初期位相の調整機能があります。また、出力パルス数を指定することも可能です。

PWM出力を行うには、HTTPクライアント・ファームウェアの初期設定（11ページ）で使用チャンネルのenableパラメータを"1"とし、PWMパルスを出力可能な状態にしておく必要があります。出力周波数、デューティ、初期位相、出力パルス数の設定、パルス出力の開始/停止はレジスタを通して制御することができます。

パルス設定用レジスタへのアクセス方法

各チャンネルには周波数、デューティ、初期位相、出力パルス数を設定するためのレジスタが用意されています。チャンネル 0 を例にとると、周波数、デューティ、初期位相、出力パルス数の設定は、それぞれ PwmFreq0（アドレス 44～47）、PwmDuty0（アドレス 48～51）、PwmPhase0（アドレス 52～55）、NumOfPulse0（アドレス 80～83）レジスタに書き込むことで行います。

各レジスタは 32 ビットで表現され、対応する 4 つのレジスタに 8 ビットずつ分かれてマップされています。PwmFreq0 レジスタを例にとると、設定値は図 20のように格納されています。

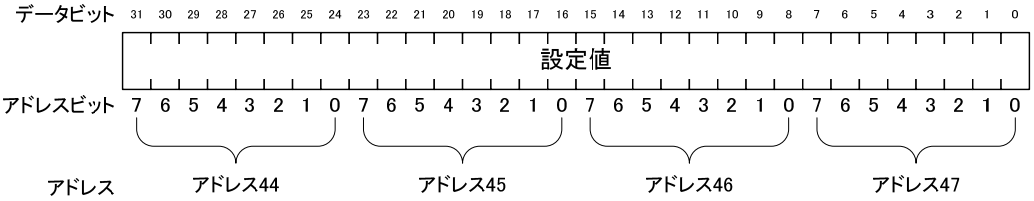


図 20　PwmFreq0 レジスタ

パルスの設定

周波数はチャンネルに応じて PwmFreq0、PwmFreq1、PwmFreq2 レジスタに対して Hz 単位で書込みを行ってください。

ディーティの設定は PwmDuty0、PwmDuty1、PwmDuty2 レジスタへ、初期位相は PwmPhase0、PwmPhase1、PwmPhase2 レジスタへ書き込むことで行います。デューティと初期位相は 0～100 までの値を与えます。

各レジスタの設定値と出力パルスの関係は図 21のようになります。

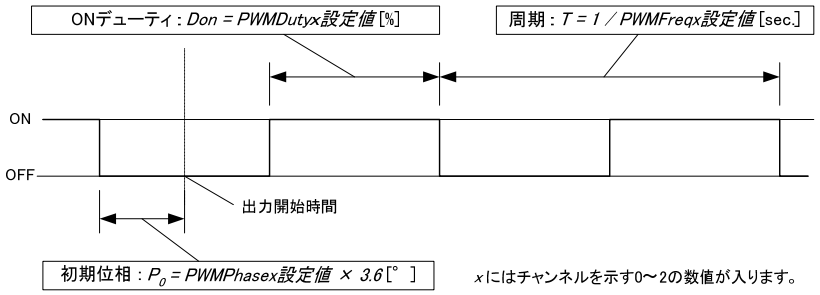


図 21　レジスタ設定値と出力パルスの関係

周波数とディーティは PWM パルス出力中に変更することが可能です。初期位相は停止している場合のみ変更可能です。

PWM パルスは基準クロックを分周して作られるため、周波数、デューティ、初期位相などのパラメータは離散的な値となります。そのため、希望の値と実際に設定できた値が異なっている場合があります。実際に設定できた値は、書込みを行った後にレジスタの値を読み出すことで取得することができます。

出力パルス数の設定

デフォルトの状態ではPWM出力は一度出力を開始すると、レジスタを操作して停止するまで出力を続けます。出力するパルス数を設定するには、出力開始前に各チャンネルに応じてNumOfPulse0、NumOfPulse1、NumOfPulse2 レジスタに出力したいパルス数を書き込みます。出力開始後、指定数のパルスを出力し終えたチャンネルは自動的に停止します[2]。

これらのレジスタに 0 を書き込むことはできません。0 を指定するとエラーとなります。

また、パルスを出力している間はこのレジスタの値を読むことにより、残りのパルス数を知ることができます。

パルス出力の開始/停止

パルス出力の開始/停止はTimerStart（アドレス 93）レジスタに書き込むことで行います。ビット 0～2 がチャンネル 0～2 に対応しています（図 22）。"1"を書き込んだビットと対応するチャンネルは出力が開始され、"0"を書き込んだビットと対応するチャンネルは出力が停止します。

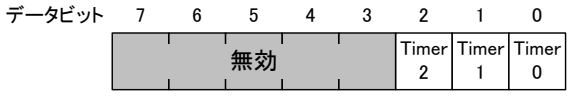


図 22　TimerStart レジスタ

次で説明するハードウェアカウンタも 16 ビットタイマの機能を使用します。ハードウェアカウンタのチャンネル 1、2 を使用した場合には PWM 出力のチャンネル 1、2 は使用できません。

[2] 出力パルスの OFF 期間が短すぎると ON 状態で停止する場合や、指定数を超えてしまう場合があります。OFF の時間を 50 μ sec 以上確保するようにしてください。

□ ソフトウェアカウンタ（パルスカウンタ）

ソフトウェアカウンタ（パルスカウンタ）は製品搭載マイコンの外部割込みを利用したカウンタ機能です。単相カウントで使用する場合はチャンネル 0～4 の最大 4 チャンネル、2 相カウントで使用する場合は、2 チャンネルずつ対で使用し、チャンネル 0 と 1、チャンネル 2 と 3 の組み合わせで最大 2 チャンネル使用可能です。

ソフトウェアカウンタを利用するにはHTTPクライアント・ファームウェアの初期設定（11ページ）で各チャンネルのカウントモードを設定しておきます。カウンタ値の読出し、書込み（クリア）、開始/停止はレジスタを通して制御することができます。

カウンタ値の読出しと書込み

カウンタ値の読出し、書込みはアドレスマップの PC0（アドレス 12～15）、PC1（アドレス 16～19）、PC2（アドレス 20～23）、PC3（アドレス 24～27）レジスタを読み書きすることで行います。

カウントモードが単相の場合、カウント動作はそれぞれのチャンネルが独立に行いますので、PC0～PC3 レジスタから読み出した値がそのままカウンタ値となります。

カウントモードが 2 相の場合、2 つのチャンネルを使用します。2 相カウントの結果は 2 つのチャンネルのカウンタ値を合計することで得られます。例えば、チャンネル 0 とチャンネル 1 で 2 相カウントを行う場合、PC0 レジスタの値と PC1 レジスタの値の合計が、2 相カウントの結果となります。

カウンタ値は 32 ビットで表現され、各端子のカウンタ値は対応する 4 つのレジスタに 8 ビットずつ分かれてマップされています。

PC0 端子を例にとると、カウンタ値は図 23のように格納されています。

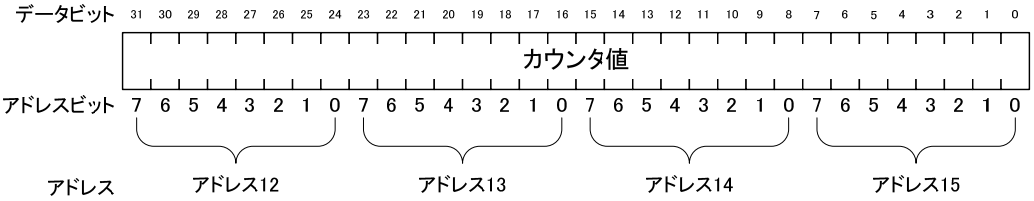


図 23　PC0 レジスタ

カウント動作の開始/停止

カウント動作の開始/停止はアドレスマップのPCStart（アドレス 95）レジスタに書き込むことで行います。PCStartレジスタのビット 0～3 が、それぞれソフトウェアカウンタのチャンネル 0～3 に対応しています（図 24）。"1"を書き込んだビットと対応するチャンネルはカウントを開始し、"0"を書き込んだビットと対応するチャンネルはカウントを停止します。

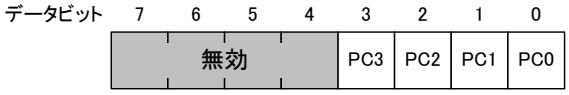


図 24　PCStart レジスタ

□ ハードウェアカウンタ

ハードウェアカウンタは、製品搭載マイコンに内蔵されている 16 ビットタイマという機能を利用したカウンタ機能です。ハードウェアによるカウント動作を行いますので高速に動作します。

ハードウェアカウンタはチャンネル 1 とチャンネル 2 の 2 チャンネルが使用可能で、チャンネル 2 は 2 相カウントモードにも設定可能です。[3]

ハードウェアカウンタを利用するには、HTTPクライアント・ファームウェアの初期設定(11ページ)で使用チャンネルのenableパラメータを"1"とし、カウントモードを設定しておきます。カウンタ値の読出し、書込み(クリア)、開始/停止はレジスタを通して制御することができます。

カウンタ値の読出しと書込み

カウンタ値の読出し、書込みはアドレスマップの TimerCnt1(アドレス 30、アドレス 31)、TimerCnt2(アドレス 32、アドレス 33)レジスタを読み書きすることで行います。

カウンタ値は 16 ビットで表現され、各端子のカウンタ値は対応する 2 つのレジスタに 8 ビットずつ分かれてマップされています。

TimerCnt1 端子を例にとると、カウンタ値は図 25のように格納されています。

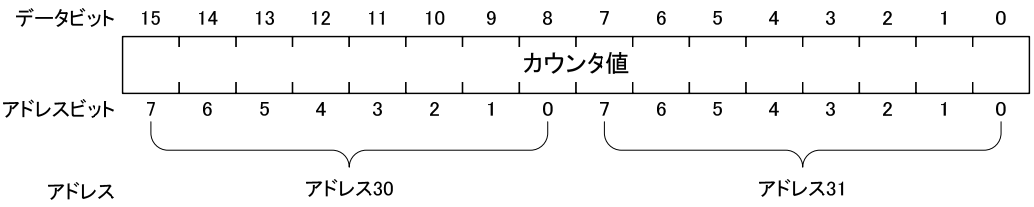


図 25 TimerCnt1 レジスタ

カウント動作の開始/停止

カウント動作の開始/停止はアドレスマップのTimerStartレジスタ(36ページ、図 22)に書き込むことで行います。TimerStartレジスタのビット 1、ビット 2 が、それぞれハードウェアカウンタのチャンネル 1、チャンネル 2 に対応しています。"1"を書き込んだビットと対応するチャンネルはカウントを開始し、"0"を書き込んだビットと対応するチャンネルはカウントを停止します。

ビット 0 はハードウェアカウンタの機能では使用しません。

□ その他

UserStatusレジスタ

アドレスマップにマップされている UserStatus(アドレス 35)レジスタは、ユーザーに開放されているレジスタです。主に設定状態や初期化情報などを格納するために用います。

[3] 2 相カウント時に使用する入力端子はチャンネル 1 と共用になっているため、チャンネル 2 で 2 相カウントする場合はチャンネル 1 が(実質)使用できなくなります。

8. tw_xmlライブラリ 関数リファレンス

□ エラー関数とエラーコード

error()

関数宣言
error(errMessage, errCode)

errMessage ： エラーメッセージ
errCode　　： エラーコード

tw_xmlライブラリ関数のエラーハンドラとして登録する関数のプロトタイプです。errCode引数に渡されるエラーコードの 1000 番未満はHTTPに関するエラーコード、1000 番以降は独自に定義されたエラーコードになります。主なエラーコードは表 32を参照してください。

表 32　エラーコード一覧

エラーコード	意味
403	アクセスが拒否されました。
404	スクリプトが見つかりません。
500	サーバー内部でエラーが発生しました。
1000 NullDataError	サーバーから受信したデータが空です。
1001 NoDeviceError	デバイスが登録されていません。techw.addDevice()関数を使用してデバイスを登録してください。
1002 XmlError	XML ファイルの取得でエラーが発生しました。エラーの詳細はメッセージを参照してください。
1003 InternalServerError	サーバー内部のスクリプトの処理でエラーが発生しました。エラーの詳細はメッセージを参照してください。
1004 InvalidValueError	送信データの値が不正です。
1005 WritingProhibitedError	書込みが禁止されているアドレスに値を書き込もうとしました。
1006 IncompletexmlError	XML に値の変換に必要な項目が不足しているため変換できません。
1007 InvalidDataNameError	データ送信の際、指定されたデータ名が XML の中に見つかりません。
1008 InvalidArgumentError	関数呼出しの引数が不正です。

□ デバイス登録関数

techw.addDevice()

関数宣言
techw.addDevice(xmlUrl, deviceId)

xmlUrl　： XML ファイルの URL
deviceId ： XML ファイルに対応するデバイスの装置番号

サーバーからデータを取得するデバイスの装置番号と、データを変換するための XML ファイルの URL との対応関係を登録します。ここで登録された装置番号と XML は techw.getData()、techw.getRecord()、techw.postData()で使用されるため、これらの関数を実行する前に必ず本関数を呼び出してください。

□ データ取得関数

techw.getData()

関数宣言
techw.getData(url, callback, error)

url : データ取得用スクリプトの URL
callback : データ取得成功時に呼ばれる関数
error : エラー発生時に呼ばれる関数

techw.addDevice()で登録したデバイスの最新のデータをサーバーから取得し、XML の記述に従って変換します。続けてデータの取得を行うには、callback 引数に techw.getData()関数が呼び出されている関数を指定してください。
取得に成功した場合は callback 引数に指定したコールバック関数が呼び出されます。callback の詳細は次項を参照してください。

techw.getData() 関数の引数に指定するcallback() 関数

関数宣言
callback(data, time)

data : 受信したデータ (「データ構造」を参照)
time : 日時

データ構造
data[装置番号][データ名]([配列の添え字]).データ本体

XML の<type>要素の[array]属性が"true"の場合、データにアクセスするための[配列の添え字]を指定してください。この配列は新しいデータを読み出す毎に更新されます。

techw.getRecord ()

関数宣言
techw.getRecord(url, callback, error, start, end, interval, average, aveTime)

url : データ取得用スクリプトの URL
callback : データ取得成功時に呼ばれる関数
error : エラー発生時に呼ばれる関数
start : データ取得開始日時を"YYYY/MM/DD hh:mm:ss"の形で指定
end : データ取得終了日時を"YYYY/MM/DD hh:mm:ss"の形で指定
interval : データ取得の間隔を分単位で指定
average : 平均値を求める場合"true"を指定
aveTime : 平均値を求める間隔を分単位で指定

techw.addDevice()で登録したデバイスの指定期間のデータをサーバーから取得し、対応する XML に従ってデータを変換します。ただし、XML で<type>要素に"boolean"が指定されている<field>は変換されません。
average 引数に"true"が指定されている場合、aveTime 引数で指定された時間内のデータの平均値を取得します。また、サーバーから取得するデータは aveTime 引数で指定された時間間隔となります。
変換が成功した場合は callback 引数に指定したコールバック関数が呼び出されます。callback の詳細は次項を参照してください。

techw.getRecord() 関数の引数に指定するcallback() 関数

関数宣言
callback(data)

data : 受信したデータ（「データ構造」を参照）

データ構造
data[装置番号][データ名]([配列の添え字])[time:日時, data:データ本体]

XML の<type>要素の[array]属性が"true"の場合、データにアクセスするための[配列の添え字]を指定してください。この配列は新しいデータを読み出す毎に更新されます。

□ デバイス操作関数

techw.postData ()

関数宣言
techw.postData(url, callback, error, data, dataName, deviceId, index, mask)

url : データ取得用スクリプトの URL
callback : データ取得成功時に呼ばれる関数
error : エラー発生時に呼ばれる関数
data : 送信するデータの配列
dataName : XML に記されている<field>要素の[name]属性
deviceId : データ送信先デバイスの装置番号
index : 配列データの書込み開始位置
mask : マスクビットを指定する場合"true"を指定

deviceId 引数のデバイスに対応する XML の記述に従ってデータを変換し、サーバーに送信します。index 引数は、XML の<type>要素の[array]属性が"true"の場合にのみ有効です。index 引数に指定が無い場合、配列データの先頭から書き込まれます。また、XML の<type>要素で"boolean"が指定されているデータに書込みを行う場合、mask 引数を指定してください。mask 引数の"1"が指定されているビットのみ操作されます。
送信に成功した場合は callback 引数に指定したコールバック関数が呼び出されます。callback の詳細は次項を参照してください。

techw.postData() 関数の引数に指定するcallback() 関数

関数宣言
callback(msg, dataName, index)

msg : 受信したメッセージ
dataName : 送信データの<field>要素の[name]属性
index : 送信したデータが配列の場合、配列データの書込み開始位置

サポート情報

製品に関する情報、最新のファームウェア、ユーティリティなどは弊社ホームページにてご案内しております。また、お問い合わせ、ご質問などは下記までご連絡ください。

テクノウェーブ(株)
URL：http://www.techw.co.jp
E-mail：support@techw.co.jp

(1) 本書、および本製品のホームページに掲載されている応用回路、プログラム、使用方法などは、製品の代表的動作・応用例を説明するための参考資料です。これらに起因する第三者の権利(工業所有権を含む)侵害、損害に対し、弊社はいかなる責任も負いません。

(2) 本書の内容の一部または全部を無断転載することをお断りします。

(3) 本書の内容については、将来予告なしに変更することがあります。

(4) 本書の内容については、万全を期して作成いたしましたが、万一ご不審な点や誤り、記載もれなど、お気づきの点がございましたらご連絡ください。

改訂記録

年月	版	改訂内容
2014 年 10 月	初	