

組み込み型 LAN 内蔵 H8CPU ボード

ConDuLan

操作ガイド — ROM ファイル編

本書に出てくる製品名などは各社の商標または登録商標です.

はじめに

組み込み型 LAN 内蔵 H8CPU ボード ConDuLan は電源を供給するだけで 40 点のデジタル入力を Web ラウザで監視できるなど手軽に組み込み Web サーバを構築することができます。

ConDuLan の Web ページは内蔵 ROM ファイルを変更することにより自由にデザインすることができます。また、内蔵 POST 処理や SSI の動作も ROM ファイルによって指定することができますので目的に応じたシステムを柔軟に構成することができます。

この操作ガイドは内蔵 ROM ファイルを修正して ConDuLan の Web ページデザインを変更するために必要な基礎知識をご紹介します。

ConDuLan に用意されている LAN-シリアル変換を機能させるには ROM ファイルを修正する必要がありますので、その方法もをご紹介します。

Web ページのための ROM ファイルは HTML で記述されています。修正は通常のテキストエディタでおこなえますが、HTML に関する知識が必要となります。HTML についての詳細は市販されている HTML のハンドブックなどを参照ください。

目次

1. ROM ファイル.....	1
1.1 ROM ファイル構成.....	2
1.2 ROM ファイルを生成する(romtar).....	3
1.3 ConDuLan へ ROM ファイルを書き込む.....	4
1.4 一般的なプログラムで ROM ファイルを生成する.....	5
1.5 ROM ファイルの容量制限.....	6
2. ROM ファイルの簡単な修整をする.....	7
2.1 トップページを背景色を変えてみる.....	8
2.2 AD 変換ページを別ウィンドウで開く.....	9
3. Web ページリンクと内蔵 CGI.....	10
3.1 簡単な Web ページを追加してみる.....	11
3.2 内蔵 CGI.....	12
3.3 内蔵 CGI 名.....	13
4. ConDuLan 内蔵 SSI.....	14
4.1 組み込み SSI #exec cgi=.....	15
4.2 ConDuLan の設定を表示する SSI.....	16
4.3 バージョンを表示する SSI.....	17
4.4 ページリフレッシュ関連 SSI.....	18
4.5 日付と時刻を表示する SSI.....	19
4.6 AD 変換 SSI.....	20
4.7 DA 変換値を表示する SSI.....	21
4.8 デジタル入出力構成を表示する SSI.....	22
4.9 デジタル入力値を表示する SSI.....	23
4.10 デジタル出力値を表示する SSI.....	24
5. 内蔵 SSI の使用例.....	25
5.1 トップページを SSI(#exec)の例.....	26
5.2 トップページを SSI(#include)の例.....	27
5.3 AD 変換 SSI の例.....	28
5.4 リフレッシュ SSI 使用例.....	29
5.5 デジタル入力 SSI 使用例.....	30
5.6 デジタル入力 SSI のデータ変更例.....	31
6. フォームの送信.....	32
6.1 DA 変換フォーム.....	33
6.2 DA 変換フォーム(現在値表示).....	34
6.3 デジタルポート構成フォーム(機能).....	35
6.4 デジタルポート構成フォーム(記述).....	36
6.5 デジタルポート構成フォーム(SSI 使用).....	37

6.6 デジタル出力フォーム(HIGH 出力).....	38
6.7 デジタル出力フォーム(LOW 出力).....	39
6.8 デジタル出力フォームの記述.....	40
6.9 デジタル出力フォームの記述(SSSI 使用).....	41
6.10 デジタル出力フォームの値.....	42
6.11 デジタル出力フォームの書式.....	43
6.12 デジタル出力フォームの書式変更例.....	44
7. GET によるクエリー.....	45
7.1 GET クエリーのための ROM ファイル変更.....	46
7.2 GET クエリーの Web 画面.....	47
7.3 form 送信と Web アクセス制限の関係.....	48
8. LAN-シリアル変換.....	49
8.1 LAN-シリアル通信動作許可.....	50
8.2 LAN-シリアル設定ページへのリンク.....	51
8.3 LAN-シリアルホストプログラム.....	52
8.4 LAN-シリアル通信設定.....	53
8.5 LAN-シリアル注意事項.....	54
9. etc の ROM ファイル.....	55
おわりに.....	56

1. ROM ファイル

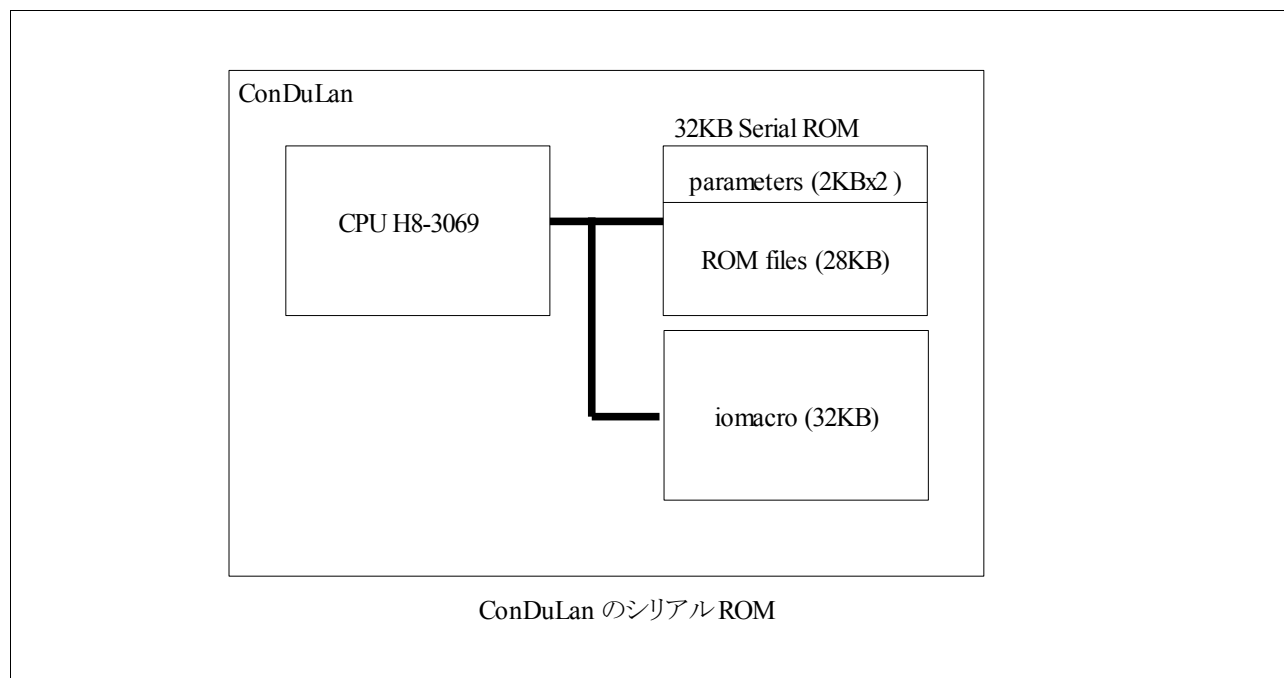
ConDuLan は内蔵する Web のページ情報やシステムの静的な構成情報を ROM ファイルというファイルの形式で内蔵しています。ROM ファイルをインストールするには PC 上で作成した複数のファイルを ConDuLan の内蔵 ROM へ一括して書き込みます。内蔵するアプリケーションプログラムがファイルとしてアクセスできるようになっていますが、ConDuLan 内ではファイルの修正はできませんので ROM ファイルと呼ばれています。

ConDuLan には 32KB のシリアル ROM が 2 個実装されていて、そのうちの 1 個には IP アドレスなどのパラメータと ROM ファイルが格納されています。もう一方の ROM はスクリプト言語 iomacro のソースプログラムをインストールするために用意されています。

ROM ファイルは 32KB シリアル ROM 内の 28KB の領域に置かれています。残りの 4KB は 2KB のパラメータを二重化して格納しています。

ROM ファイルを変更してインストールすることにより ConDuLan 内蔵の Web ページを自由にデザインすることができます。

この章では ROM ファイルを ConDuLan へインストールする方法をご紹介します。



1.1 ROM ファイル構成

ConDuLan の ROM ファイルには容量を超えない限りどのようなファイル構成でも可能ですが、通常2つのディレクトリ *etc* と *http* から構成されています。

ディレクトリ *etc* の中にはアプリケーションプログラムの固定的な設定情報が含まれ、一方ディレクトリ *http* の中には ConDuLan の Web ページファイルが含まれます。

ConDuLan が Web サーバとして構築されているとき、ファイル名指定なしの

```
http://192.168.0.254/
```

のようなページアクセスがあると、ディレクトリ *http* 内の ファイル *index.html* が参照されます。

また、Web サーバとして応答できるページファイルはディレクトリ *http* 内のファイルに限定されています。

では ConDuLan にインストールされている ROM ファイルを調べてみましょう。添付されている CD の *romfile* というディレクトリにある *universal* というディレクトリを PC へコピーしましょう。コピーしたディレクトリ *universal* には *etc* と *http* という2本のディレクトリがあり、それぞれ複数のファイルが含まれています。ファイル構成は次のような構成になっています。

```
etc --- val_pctl_1to0
        val_pctl_0to1
        ssi_pval_1
        ssi_pval_0
        ssi_pctl_1to0
        ssi_pctl_0to1
        html_post
        html_lan
        html_get
        html_access
        html_telnetd
        vender
        system_name

http --- iomonitor.html
        iocontrol.html
        ioconfig.html
        index.html
        ad.html
        da.html
        logo.gif
        credit.inc
        poweredby.inc
        version.inc
        showconfig.html
```

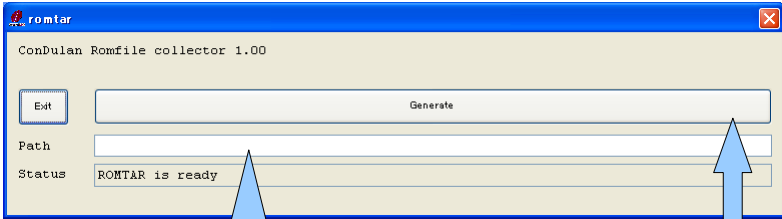
1.2 ROM ファイルを生成する(romtar)

PC にコピーしたディレクトリ *universal* を ROM ファイルとして ConDuLan にインストールするにはすべてのファイルを1本のファイルにまとめる必要があります。このためのファイル書式は一般に TAR と呼ばれている仕様に基づいています。TAR は Linux など Unix 系の OS では通常標準で用意されています。Windows の場合は添付 CD に用意してあるツール *romtar* で TAR ファイルにまとめることができます。

次の手順で ROM ファイル *universal.tar* を生成してみましょう。

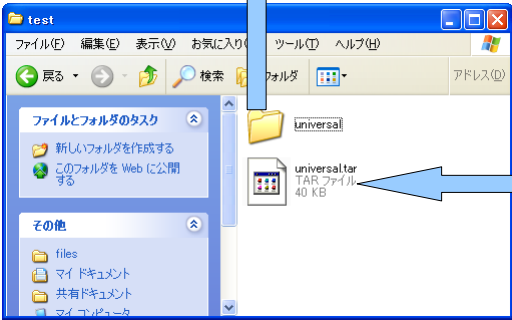
- CD の tools フォルダにある *romtar* をダブルクリックして起動します。
- PC にコピーしたディレクトリ *universal* を *romtar* のウィンドウにドラッグします。
- 「Generate」をクリックすると
- *universal.tar* という TAR ファイルが生成されます。これが ROM ファイルです。

1. CD の *tools* ディレクトリにある *romtar* をダブルクリックして起動する。



2. *etc* と *http* フォルダを含んでいるフォルダ *universal* を *romtar* のウィンドウにドラッグする。

3. *universal* をドラッグしてから *Generate* をクリックする。



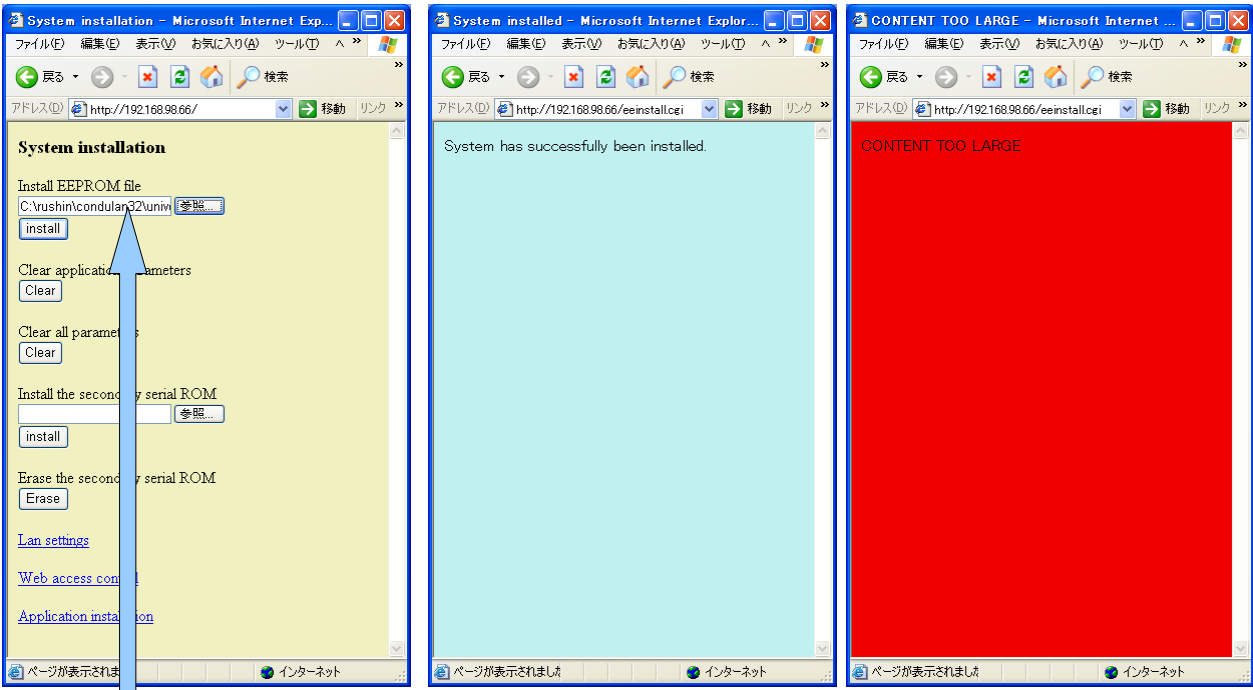
4. *universal.tar* という名の TAR ファイルが生成される

romtar の使用方法

1.3 ConDuLan へ ROM ファイルを書き込む

作成した ROM ファイル *universal.tar* を次の手順で ConDuLan へインストールしてみましょう。

- ConDuLan をインストールモードで起動する
(インストールスイッチを押した状態で電源投入する)
- PC の Web ブラウザから ConDuLan のトップページを開く
(工場出荷状態では <http://192.168.0.254>)
- Install EEPROM file へ *universal.tar* への完全パス名を入力する。
(参照...をクリックして *universal.tar* を検索してもよい)
- install をクリックすると ROM ファイルのインストールが開始される。
(数秒かかる)
- インストールが正常に終了したら ConDuLan の電源を断-入して再起動する。



インストールする ROM ファイルを指定して *install* をクリックする。

インストールに成功した場合の画面

ROM ファイルのインストール画面

ファイルが大きすぎてインストールに失敗した場合の画面

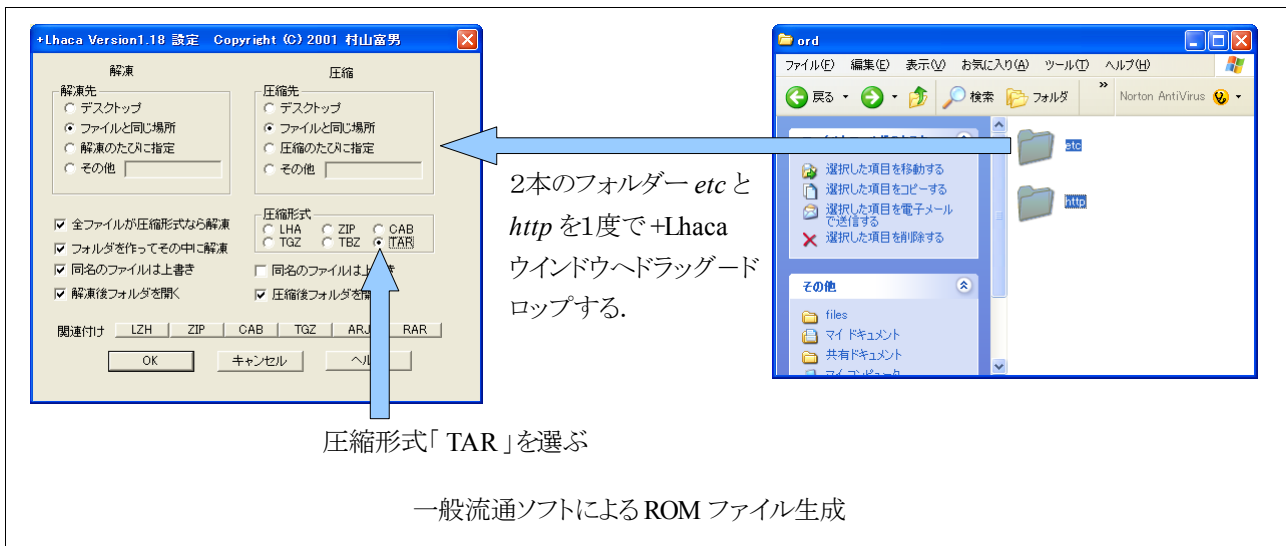
1.4 一般的なプログラムでROM ファイルを生成する

Windows では+Lhaca

<<http://www.vector.co.jp/soft/win95/util/se166893.html>>

などフリーのソフトウェアが流通していますので、ROM ファイル生成にこれを利用することもできます。

生成ファイルは *etc.tar* などの名前になりますので、生成後必要に応じて名前を変更してお使いください。



Unix 系の場合は *etc* と *http* があるディレクトリで

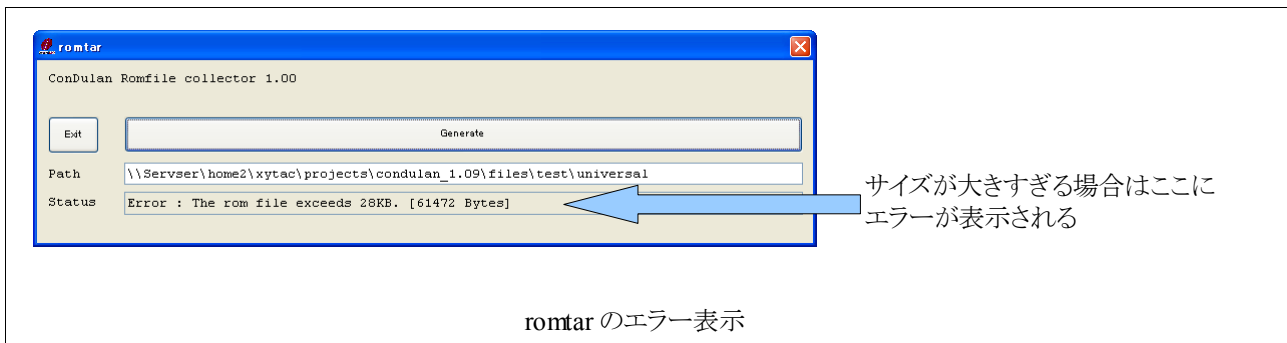
```
tar cf universal.tar etc http
```

などとしてファイル生成します。

1.5 ROM ファイルの容量制限

ConDuLan の ROM ファイルは 32KB のシリアル ROM 中 28KB の領域に格納されます。併せて内蔵 RTOS のファイル管理情報も格納されますので、実質的な ROM ファイル総容量は 27KB 程度となります。この容量を超えた場合は ROM ファイルのインストールでエラーとなります。各ファイルを一本のファイルにまとめた TAR ファイルは実際のファイルサイズの総計より相当大きなサイズになっていますが、TAR ファイル受信後 ConDuLan が個別ファイルに分解しますので、各ファイルサイズの総計が 27KB 程度に収まっていればインストール可能です。

ツール *romtar* の場合には ROM ファイル生成時にインストール可能サイズを超えるとエラーとなり、TAR ファイルは生成されません。



2. ROM ファイルの簡単な修整をする

ROM ファイルの利用例として ConDuLan の ROM ファイルを変更し, Web ページがどのように変わるか試してみます.

すでにご紹介したように CD の *universal* ディレクトリを PC にコピーしておきます. 修正はテキストエディタを使用しますが, Windows をご使用でテキストエディタをお持ちでない場合はメモ帳でもかまいません.

修正した ROM ファイルを ConDuLan にインストールしても, CD の *romfile* ディレクトリにある *universal.tar* をインストールしなおせば元に戻せます.

2.1 トップページの背景色を変えてみる

ROM ファイル修整の例として ConDuLan 内蔵 Web トップページ の背景色を変更してみます。

CD からコピーしたファイルのうち、ディレクトリ *http* にある *index.html* をテキストエディタで開いてください。

ファイルの2行目に

```
<body bgcolor="#f0f0c0">
```

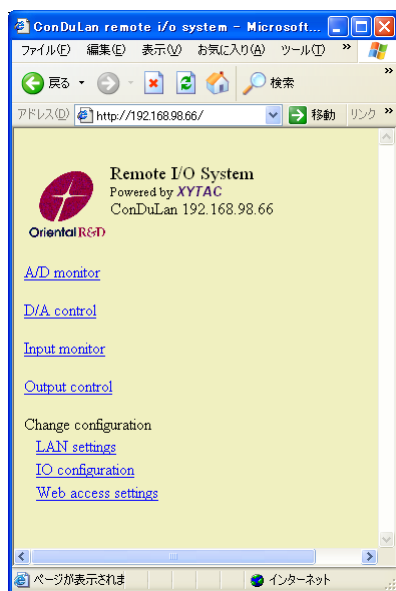
という行があります。

これはこのページの背景色を番号 *#f0f0c0* で指定した色で表示することを示しています。この値をたとえば *#66cc66* に修正してみます。

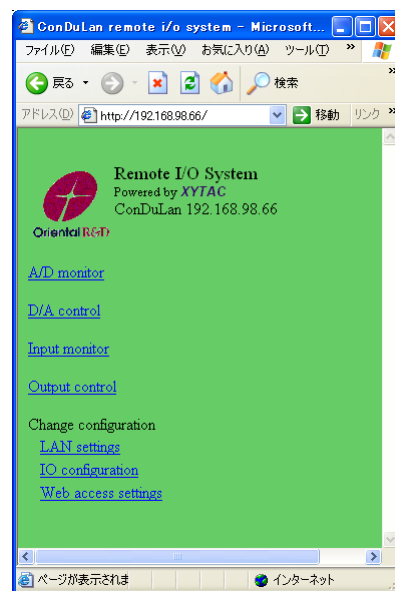
```
<body bgcolor="#66cc66">
```

テキストエディタを終了して ROM ファイルを作り ConDuLan へインストールしてみます。

背景色を変更した ROM ファイルをインストールしたら ConDuLan の電源を一度遮断し、再投入してください。Web ブラウザでトップページを表示すると背景色に変更されたことを確認できます。



本来のトップページ



背景色を変更した結果

ROM ファイルの変更例 (背景色)

2.2 AD 変換ページを別ウィンドウで開く

もうひとつ Web ページ自身の修正でデザインを変更する例を試してみましょう。

今回もディレクトリ *http* のファイル *index.html* を修正します。

ファイルの中央辺りにある

```
<p><a href="ad.html">A/D monitor</a></p>
```

の行を

```
<p><a href="ad.html" target="_blank">A/D monitor</a></p>
```

に変更します (*target="_blank"* を追加)。

もう一度 ROM ファイルを作りインストールしてから ConDuLan の電源を再投入してみましょう。トップページの *A/D Monitor* をクリックすると、今までは AD 変換の Web ページが同じ Web ウィンドウで表示されていましたが、今回の修正で別ウィンドウが開かれそこに表示されるはずです。

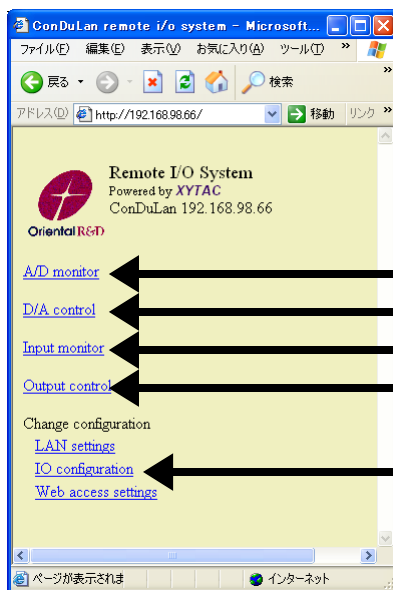
このように Web ページを記述しているファイルを HTML 規約に従って修正することによってさまざまにデザインを変更することができます。JavaScript などを利用すればさらに高度な Web ページを記述することも可能です。

3. Web ページリンクと内蔵 CGI

ConDuLan のトップページには ROM ファイルに含まれる各ページファイルを指定するリンクが含まれています。たとえば *A/D monitor* の表示を Web ブラウザでクリックすれば AD 変換用 Web ページである */ad.html* という ROM ファイルが表示されます。あらかじめ HTML で記述されたファイルを用意しておけば、他の Web ページからクリックすることにより表示させることができます。

Web ページに表示する ROM ファイルはディレクトリ *http* 以下に含まれている必要があります。またシンボリックリンクやショートカットは使えませんのでご注意ください。

トップページの *LAN settings* をクリックすると、あらかじめ ConDuLan に内蔵されている LAN 設定用 CGI が Web データを生成し表示します。このような内蔵 CGI は LAN 設定と Web アクセス制限設定の2本が用意されています。



以下のような記述でそれぞれの ROM ファイルへのリンクとなっている

```
<p><a href="ad.html">A/D monitor</a></p>
```

```
<p><a href="da.html">D/A control</a></p>
```

```
<p><a href="iomonitor.html">Input monitor</a></p>
```

```
<p><a href="iocontrol.html">Output control</a></p>
```

```
<a href="ioconfig.html">IO configuration</a>
```

トップページのリンク

3.1 簡単な Web ページを追加してみる

では ConDuLan に簡単な Web ページを追加する方法をご紹介します。

テキストエディタでたとえば

```
TEST
```

とだけ記述されたファイルを用意します。ファイル名は *test.html* で *http* ディレクトリに置きます(以後 *http/test.html* のように記述します)。

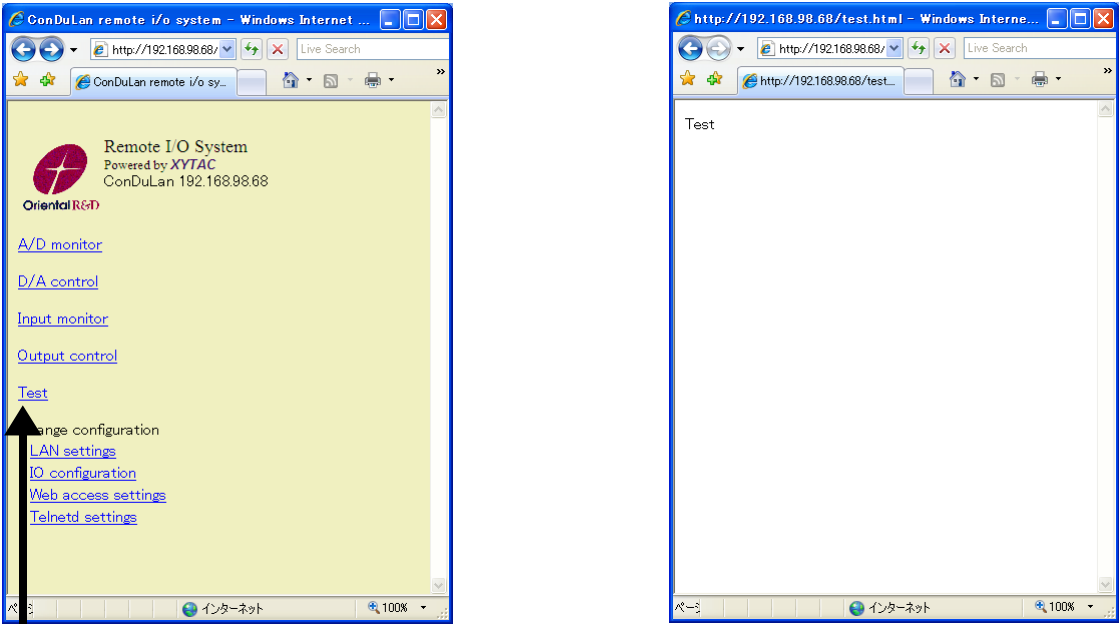
トップページである *http/index.html* ファイルの

```
<p><a href="iocontrol.html">Output control</a></p>
```

の記述の次に

```
<p><a href="test.html">Test</a></p>
```

を追加します。これでトップページには Test という表示が追加され、そこをクリックすると ROM ファイル *http/test.html* の内容 TEST が表示されるようになります。



`<p>Test</p>`

追加したリンク
クリックすると右図の追加ページが表示される

追加したページ

ROM ファイルの変更例 (Web ページ追加)

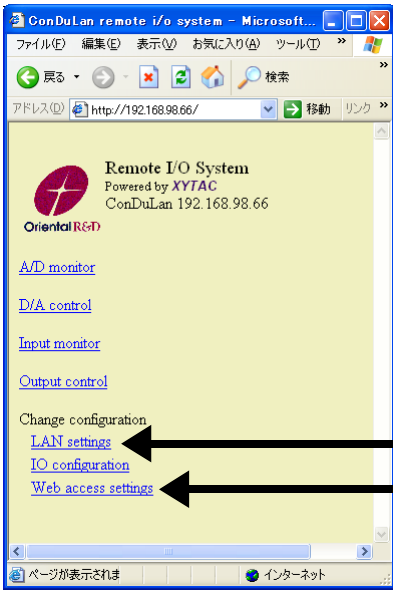
3.2 内蔵 CGI

トップページ用 ROM ファイル `http/inex.html` を調べると ROM ファイルに含まれないファイルへのリンクが2種類見つかります。

```
<a href="lan_settings.html">LAN settings</a>
```

```
<a href="access_control.html">Web access settings</a>
```

これらは ConDuLan 内蔵 CGI へのリンクとなっています。ConDuLan は `lan_settings.html` ページへの要求を受けると LAN 設定用のページを生成し表示します。このページは ConDuLan インストールモードにある LAN 設定ページと同じ内容です。同様に `access_control.html` ページの要求で Web アクセス制限の設定ページを生成して表示します。



以下のような記述で LAN 設定 CGI と Web アクセス制限 CGI へのリンクを用意することができる。

```
<a href="lan_settings.html">LAN settings</a>  
<a href="access_control.html">Web access settings</a>
```

トップページにある CGI へのリンク

3.3 内蔵 CGI 名

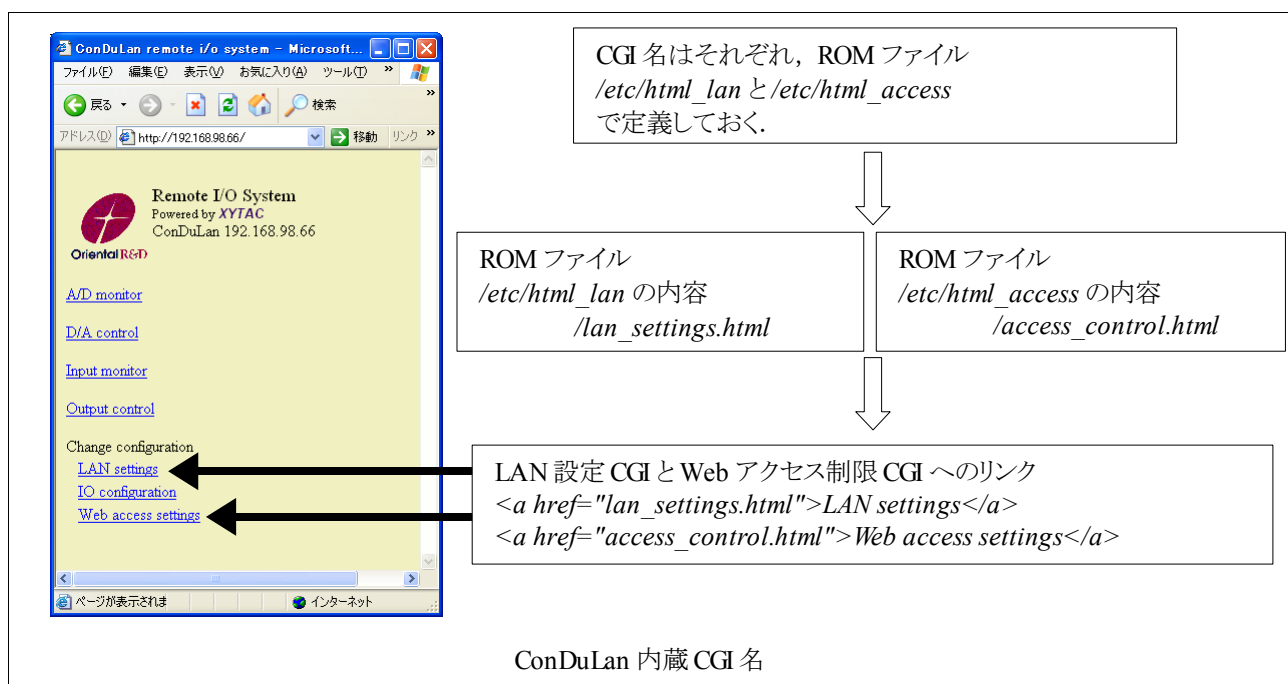
ConDuLan にインストールされている LAN 設定と Web アクセス制限設定の2つの CGI 名は、それぞれ *lan_settings.html* と *access_control.html* でしたが、これらの名前を変更することができます。

内蔵 CGI の名前は2つの ROM ファイル */etc/html_lan* と */etc/html_access* に書き込んでおきます。CD から PC にコピーしたファイルを見ると内容は */lan_settings.html* と */access_control.html* になっています。

たとえば ROM ファイル */etc/html_lan* の内容を */lan_settings.html* から */lan.html* という記述に変更すると LAN 設定 CGI は *lan.html* という名前で起動します。

また、ファイル */etc/html_lan* という ROM ファイルが存在しなかったり、存在していても内容に */* で始まる URL が記述されていない場合は LAN 設定 CGI を起動することはできません。この場合は LAN 設定はインストールモードだけでおこなえることになります。

Web アクセス制限 CGI の名前も同様に定義しますので、インストールモード以外ではパスワードの変更などをできないようにすることも可能です。



4. ConDuLan 内蔵 SSI

今までの修正例は Web ページファイルを直接修正するという静的な変更でした。これでもずいぶん柔軟な Web ページ構築が可能ですが、SSI を利用するとさらに多彩な Web ページを構築することができます。Web の表示ファイル (*index.html* など) に SSI を記述すると ConDuLan 内蔵の該当関数が起動され、その結果を SSI 記述の代わりに表示します。

ConDuLan には次のような SSI 機能が用意されています。

```
<!--#include file="rom_file_name" -->
<!--#echo var="environment_variable" -->
<!--#exec cgi="cgi_name arg0 arg1 ..." -->
```

```
<!--#include file="rom_file_name" -->
```

はこの SSI を指定された ROM ファイルの内容で置き換えます。複数の Web ページに共通の内容を埋め込む場合などに使用します。

```
<!--#echo var="environment_variable" -->
```

はこの SSI を指定の環境変数で置き換えます。

環境変数には

<i>DOCUMENT_URI</i>	SSI のあるページの URI
<i>DATE_LOCAL</i>	現在の日付と時刻

があります。*DATE_LOCAL* につきましては ConDuLan に RTC など時間情報が付加されている場合にだけ正常動作します。工場出荷の ConDuLan 単体では正常動作いたしません。

```
<!--#exec cgi="cgi_name arg0 arg1 ..." -->
```

は登録された関数を実行します *cgi_name* は登録した SSI 名です。SSI による関数呼び出しでは引数を渡すことができます。SSI の記述を関数の生成する文字列で置き換えます。

この章では ConDuLan にあらかじめ内蔵されている SSI の動作についてご説明します。使用例は次章をご参照ください。

4.1 組み込み SSI #exec cgi=

ConDuLan に組み込まれている #exec 型の SSI には次のようなものがあります。

<i>cgi_ipaddress</i>	IPアドレス
<i>cgi_subnetmask</i>	サブネットマスク
<i>cgi_gateway</i>	ゲートウェイアドレス
<i>cgi_http</i>	HTTP ポート番号
<i>cgi_version</i>	RTOS バージョン
<i>cgi_name</i>	名前
<i>cgi_refresh</i>	ページリフレッシュ指定
<i>cgi_refresh_interval</i>	ページリフレッシュ間隔(秒)
<i>cgi_date</i>	日付
<i>cgi_time</i>	時刻
<i>cgi_ad</i>	AD変換結果
<i>cgi_da</i>	DA出力値
<i>cgi_pcnf</i>	デジタルポート構成情報
<i>cgi_pval</i>	デジタル入力値
<i>cgi_pctl</i>	デジタル出力値

記述方法は

```
<!--#exec cgi=" cgi_ad 0" -->
```

のように <!--#exec cgi= に続いて”で囲んだ SSI を書きます。SSI に引数を与える場合は SSI 名の後に空白を置き、引数文字列を書きます。

上記の例 <!--#exec cgi=" cgi_ad 0" --> では *cgi_ad* という名前の SSI が起動します。この場合は <!--#exec cgi=" cgi_ad 0" --> の記述の部分にチャンネル0の AD の値を表示します。

4.2 ConDuLan の設定を表示する SSI

ConDuLan には設定情報を表示する5本の SSI が用意されています。

それぞれ次のように記述します。

<code><!--#exec cgi=" ipaddress" --></code>	IP アドレスの表示
<code><!--#exec cgi=" subnetmask" --></code>	サブネットマスクの表示
<code><!--#exec cgi=" gateway" --></code>	デフォルトゲートウェイの IP アドレスの表示
<code><!--#exec cgi=" http" --></code>	HTTP 待ち受けポート番号の表示
<code><!--#exec cgi=" name" --></code>	ConDuLan に設定された名前の表示

これらの表示は Lan Settings メニューで設定した内容を表示します。

4.3 バージョンを表示する SSI

ConDulan の RTOS とアプリケーションファームウェアのバージョン番号を表示する SSI が用意されています。

それぞれ次のように記述します。

```
<!--#exec cgi="version os" -->          RTOS バージョン表示  
<!--#exec cgi="version application" --> アプリケーションバージョン表示
```

4.4 ページリフレッシュ関連 SSI

Web ブラウザが一定時間間隔で ConDuLan の Web ページを要求する方法を説明します。通常 Web ブラウザは取得したページのヘッダに

```
<meta http-equiv="refresh" content="10">
```

の記述があると10秒後に同じページ情報を要求するように作られています(10秒は例です)。

ConDuLan の Web ページ情報にこのような記述をしておけば指定の時間間隔で同じページの最新情報を取得できます。

ConDuLan の SSI ではさらにページ要求の時間間隔を可変にする方法を提供します。

Web ページ情報に SSI

```
<!--#exec cgi="cgi_refresh" -->
```

が含まれていると、クライアントから要求された時間(10秒を例とする)を使って

```
<meta http-equiv="refresh" content="10">
```

を生成します。このときの要求時間は GET のクエリーで

```
http://192.168.0.254/ad.html?refresh_interval=10
```

のように `refresh_interval=`指定時間(秒)を指定します。

指定時間のクエリーがなかったり指定時間が0だった場合はリフレッシュステートメントは生成しません。

また Web ページに指定された時間間隔は SSI

```
<!--#exec cgi="cgi_refresh_interval" -->
```

で表示させることができます。

以上の組み合わせを利用するとページの表示をユーザ指定の時間間隔で繰り返すことができます。

この手順は ROM ファイル `/http/ad.html` で使用されています。使用例は次章でご説明します。

4.5 日付と時刻を表示する SSI

ConDuLan には時計が内蔵されていませんので、日付と時刻を扱うことはできません。しかし追加基板に RTC を実装するなどの方法で時計を内蔵した場合には、現在時刻をシステム全体で共有する手順が用意されています。ConDuLan に時刻設定の機能が追加された場合には Web ページに日付および時刻情報を埋め込むことができます。そのために2本の SSI が用意されています。

`<!--#exec cgi="cgi_ad" -->` 日付を表示する YYYY/MM/DD

`<!--#exec cgi="cgi_time" -->` 時刻を表示する hh/mm/ss

時計機能が内蔵されていない ConDulan では表示日時は空白となります。

4.6 AD 変換 SSI

ConDuLan 内蔵の10ビット AD コンバータを起動し, AD 変換の結果を表示する SSI が用意されています.

たとえばチャンネル0の AD を起動し, その変換結果を表示するには

```
<!--#exec cgi="cgi_ad 0" -->
```

と記述します. チャンネル番号は0から7までの8チャンネルでそれ以外の値を指定した場合には空白が表示されます.

表示の値は0から1023となり, 内蔵レファレンス電圧を使用している場合は0が0V, 1023が4.1Vに相当します.

アナログ入力ピンは P70-P77(CN1-20,22,24,26,28,30,32,34)です.

4.7 DA 変換値を表示する SSI

ConDuLan 内蔵の8ビット DA コンバータに出力されている値を表示する SSI が用意されています。DA チャンネルは0と1の2チャンネルがあり、出力値は0から255のいずれかになります。内蔵レファレンス電圧を使用している場合は0が0V、255が4.1Vに相当します。

たとえばチャンネル0の DA 出力値を表示する場合は

```
<!--#exec cgi="cgi_da 0" -->
```

と記述します。

アナログ出力ピンは P76,P77(CN1-32,34)です。この2本のピンはAD入力と兼用で、電源投入後は入力として使用されます。

実際に DA 出力するにはフォームを POST 要求します。フォームを POST する手順については後述します。

4.8 デジタル入出力構成を表示する SSI

ConDuLan の各デジタル信号の入出力ポートの設定をプルダウンメニューとして表示する SSI が用意されています。

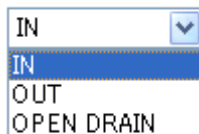
たとえば次の記述はポート pa0 の入出力設定のプルダウンメニューを表示します。あらかじめ選択されている設定が最新の設定状態となります。

```
<!--#exec cgi="cgi_pcnf pa0" -->
```

この記述では表示は次のようになります (IN が選択されている)。

A screenshot of a web browser showing a dropdown menu. The menu is currently displaying the option 'IN'. The dropdown arrow is visible on the right side of the menu box.

このメニュー内の右側にある下向き矢印をクリックすると次のようにデジタル入出力の設定プルダウンメニューが表示されます。

A screenshot of a web browser showing a dropdown menu that has been expanded. The menu box is open, showing three options: 'IN' (which is highlighted in blue), 'OUT', and 'OPEN_DRAIN'. The dropdown arrow is visible on the right side of the menu box.

この SSI は次のような表記に変換されています。

```
<select name="pa0">  
<option value="in" selected>IN</option>  
<option value="out">OUT</option>  
<option value="open">OPEN_DRAIN</option>  
</select>
```

このプルダウンメニューは通常後述するフォームと組み合わせて使用します。

4.9 デジタル入力値を表示する SSI

ConDuLan の各デジタル信号の入力状態を表示する SSI が用意されています。ConDuLan のデジタル入力の値を表示するには

```
<!--#exec cgi="cgi_pval pa0" -->
```

のように記述します。この例では pa0 の信号の状態を表示します。

表示は信号が LOW のとき 0, HIGH のとき 1 となります。

この SSI が実際に生成する文字列は

信号が LOW のときには

```
<td bgcolor="#ff9999">0</td>
```

信号が HIGH のときには

```
<td bgcolor="#33ccff">1</td>
```

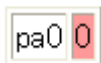
となります。これは Web ページ内でデジタル入力を表にして表示する場合に利用できます。

bgcolor= の部分は表示の背景色となります。

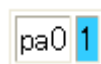
たとえば

```
<table border="1">  
<tr><td>pa0</td><!--#exec cgi="cgi_pval pa0" --></tr>  
</table>
```

と記述しておけば、入力が LOW の場合次のように表示されます。



また、入力が HIGH の場合は次のようになります。



上記記述で pa0 の部分を変更すれば信号名称を自由に変更することができます。

またこの SSI が生成する表記は ROM ファイルを変更することで自由に設定することができます。入力が LOW の場合は ROM ファイル中の/etc/ssi_pval_0 の内容が表示されます。同様に入力が HIGH の場合には ROM ファイル中の/etc/ssi_pval_1 の内容が表示されます。この2本のファイルの内容を書き換えて ROM ファイルをインストールすればデジタル入力値の表示部分を自由に変更することができます。詳しくは後述するデジタル入力 SSI 使用例を参照ください。

4.10 デジタル出力値を表示する SSI

ConDuLan の各デジタル信号の出力値を表示する SSI が用意されています。次の記述はポート pa0 に出力している値を表示します。

```
<!--#exec cgi="cgi_pctl pa0" -->
```

この SSI は通常フォーム送信に使用されるため出力される表記は

```
value="L to H" style="background-color:#ffff00"
```

あるいは

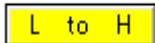
```
value="H to L" style="background-color:#ff6347"
```

のようになります。

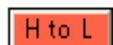
次のような記述で使用されることを想定しています。

```
<form action=/iocontrol.html method="post" enctype="text/plain">
<tt>
<input type="submit" name=pa0 <!--#exec cgi="cgi_pctl pa0" -->><br>
</tt>
</form>
```

この使用方法では LOW を出力しているとき



あるいは, HIGH を出力しているとき



のように表示されます。

上記の表示文字列や書式(背景色など)は ROM ファイルの内容を変更することで自由に設定できます。出力が LOW のときの表示文字列と書式はそれぞれ `val_pctl_0to1` と `ssi_pctl_0to1` の ROM ファイルで定義します。また, 出力が HIGH のときの表示文字列と書式はそれぞれ `val_pctl_1to0` と `ssi_pctl_1to0` の ROM ファイルで定義します。詳しくは次章でご説明します。

5. 内蔵 SSI の使用例

ConDuLan 内蔵の SSI についてご紹介してきましたが、実際の使用例を ConDuLan の ROM ファイルで見てください。

ConDuLan にインストールされている ROM ファイルにはディレクトリ `http` に以下のファイルが含まれています。

<code>http ---</code>	<code>iomonitor.html</code>	デジタル入力監視ページ
	<code>iocontrol.html</code>	デジタル出力制御ページ
	<code>ioconfig.html</code>	デジタル入出力構成ページ
	<code>index.html</code>	トップページ
	<code>ad.html</code>	AD 入力ページ
	<code>da.html</code>	DA 出力ページ
	<code>logo.gif</code>	ORD のロゴイメージファイル
	<code>credit.inc</code>	ページ情報制作クレジット
	<code>poweredby.inc</code>	開発協力組織
	<code>version.inc</code>	Web ページのバージョン
	<code>showconfig.inc</code>	構成情報表示ページ

すでに説明したように CD の `romfile` ディレクトリにあるファイルを PC にコピーして上記のファイルメモ帳などのテキストエディタで見てください (`logo.gif` はイメージデータですのでテキストエディタでは開かないで下さい)。

上記のうち拡張子が `html` で終わる 6 ファイルが Web ページファイルとなります。

残りの 3 ファイル `logo.gif`, `credit.inc`, `poweredby.inc` は Web ページの一部として表示されます。

5.1 トップページの SSI(#exec)の例

ConDuLan の Web トップページには ConDuLan に設定されている名前が SSI によって表示されています。

SSI 記述

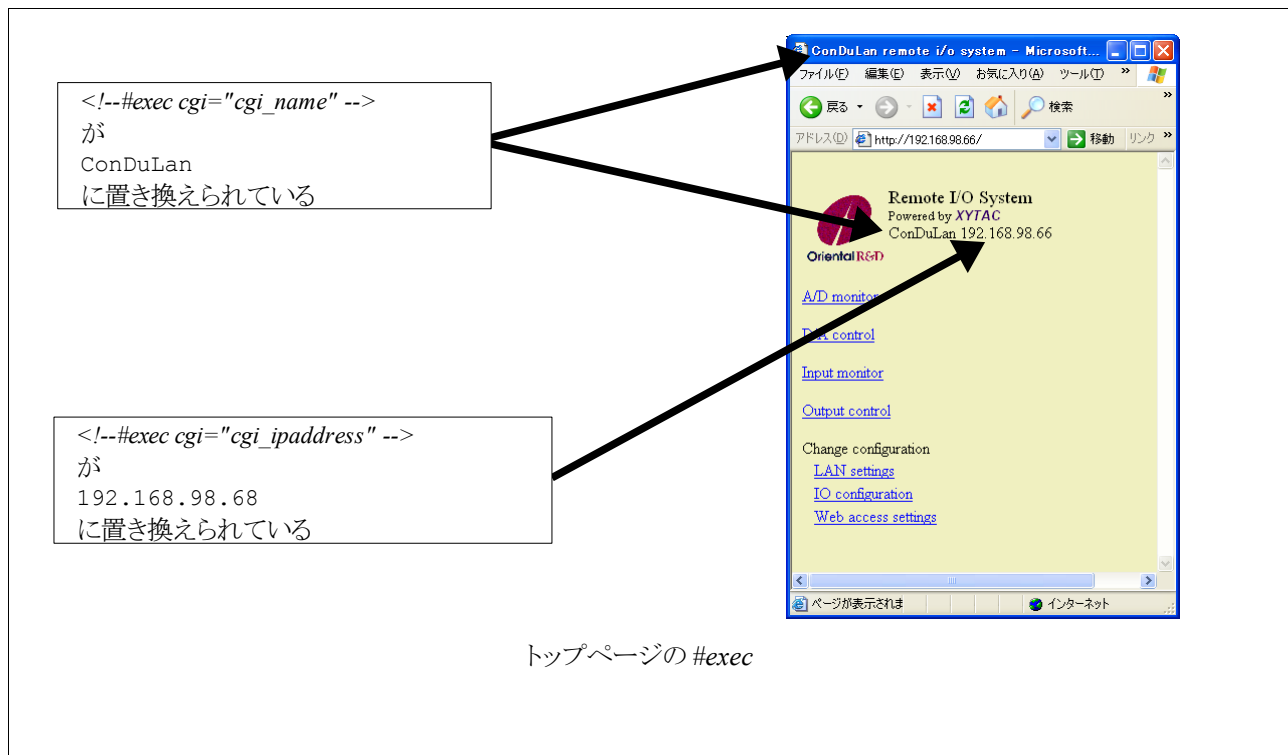
```
<!--#exec cgi="cgi_name" -->
```

は Web 表示では「ConDuLan」に置き換えられています。

また、この ConDuLan に設定されているアドレスは SSI 記述

```
<!--#exec cgi="cgi_ipaddress" -->
```

によって 192.168.98.68 と表示されています。



5.2 トップページの SSI(#include)の例

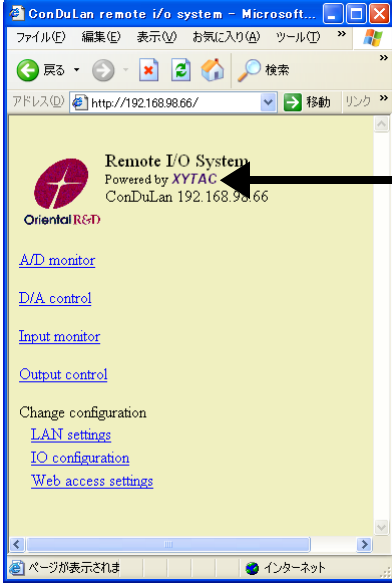
トップページには別の ROM ファイルの内容を組み込んで表示する SSI が使用されています。

ROM ファイル中の SSI 記述

```
<!--#include file="poweredby.inc" -->
```

は *poweredby.inc* という ROM ファイルの内容を表示します。ROM ファイル *poweredby.inc* には XYTAC と記述されていますので、XYTAC と表示されます。

Web の複数のページに共通で使用する文字列などは、このように一つのファイルとして ROM ファイルに用意しておいて *#include* による SSI 記述で表示させることができます。



`<!--#include file="poweredby.inc" -->`
が
XYTAC
に置き換えられている

トップページの *#include*

5.3 AD 変換 SSI の例

AD 変換用 Web ページ *ad.html* には AD 変換を実行してその結果を表示する SSI

```
<!--#exec cgi="cgi_ad 0" -->
```

が全チャンネル分含まれています(上記記述はチャンネル0の例). AD 変換 Web ページを開くと各 AD チャンネルに対応した SSI が順に起動され, それぞれ AD 変換を実行して結果を表示します.

AD 変換ページの ROM ファイルは *http/ad.html*, リンク先は */ad.html* です.

The screenshot shows a web browser window titled "ConDuLan A/D monitor" with the URL "http://192.168.98.68/adhtml?refresh_interval=10". The page content includes:

Remote I/O System
ConDuLan 192.168.98.68
A/D monitor

Channel	Value
CH0	500
CH1	474
CH2	465
CH3	467
CH4	473
CH5	480
CH6	469
CH7	467

Repeat interval in second
10
repeat
Refresh page
HOME

Two callout boxes are present:

- AD 変換 SSI(チャンネル0)
`<!--#exec cgi="cgi_ad 0" -->`
- AD 変換 SSI(チャンネル7)
`<!--#exec cgi="cgi_ad 7" -->`

AD 変換ページの SSI

5.4 リフレッシュ SSI 使用例

AD 変換用 Web ページにはページを再表示をするためのリフレッシュ用 SSI も使用されています。Web ページを要求するときに `refresh_interval=` の書式でクエリーを送信した場合、そのページに

```
<!--#exec cgi="cgi_refresh" -->
```

の記述があると、たとえばクエリーが `refresh_interval=10` の場合

```
<meta http-equiv="refresh" content="10">
```

に置き換えられます。この記述は Web 表示はされませんが、Web ブラウザが 10 秒後に同じ Web ページを要求して最新の情報を表示するよう指示します。

現在のリフレッシュ時間を表示する SSI は次のように記述して表示することができます。

```
<!--#exec cgi="cgi_refresh_interval" -->
```

リフレッシュ時間をクエリー要求するフォームで使用します。

このほか自 ページへのリンク (手動リフレッシュ用) とホームページへのリンクも含まれています。

繰り返し要求すると URI に `?refresh_interval=10` というクエリーが付加される (10秒の場合の例)

自 ページへのリンクとトップページへのリンク

```
<p><a href=
```

5.5 デジタル入力 SSI 使用例

デジタル入力監視ページ *iomonitor.html* にはデジタル入力を表示する SSI

```
<!--#exec cgi="cgi_pval pa0" -->
```

が40ポート分含まれています(上記記述はポート *pa0* の例)。

この SSI が生成する表示はデジタル入力値が LOW か HIGH かに対応してそれぞれ ROM ファイル *etc/ssi_pval_0* と *etc/ssi_pval_1* で定義しておきます。

デジタル入力監視ページの ROM ファイルは */http/iomonitor.html*, リンク先は */iomonitor.html* です。

Port2	Port5	Port7	Port8	A	B
Port2-3	0	PortA-0	0	0	
Port2-4	0	PortA-1	0	0	
Port2-5	0	PortA-2	1	0	
Port2-6	0	PortA-3	0	0	
Port2-7	0	PortA-4	0	0	
Port5-0	1	Port2-5	0		
Port5-1	1	PortA-6	1		
Port5-2	1	PortA-7	0		
Port5-3	1	Port4-0	1		
Port7-0	1	Port4-1	1		
Port7-1	1	Port4-2	1		
Port7-2	0	Port4-3	1		
Port7-3	1	Port4-4	0		
Port7-4	1	Port4-5	1		
Port7-5	1	Port4-6	0		
Port7-6	1	Port4-7	0		
Port7-7	1	PortB-0	1		
Port8-0	0	PortB-1	0		
Port8-1	0	PortB-2	1		
Port8-4	1	PortB-3	1		

デジタル入力値を表示する SSI
`<!--#exec cgi="cgi_pval pa0" -->`

この SSI は入力が LOW のとき次の記述を生成する
`<td bgcolor="#ff9999">0</td>`

この記述は次の ROM ファイルで定義されている
/etc/ssi_pval_0

デジタル入力値を表示する SSI
`<!--#exec cgi="cgi_pval pb0" -->`

この SSI は入力が HIGH のとき次の記述を生成する
`<td bgcolor="#33ccff">1</td>`

この記述は次の ROM ファイルで定義されている
/etc/ssi_pval_1

デジタル入力ページの SSI

5.6 デジタル入力 SSI のデータ変更例

デジタル入力 SSI の表示を定義している2本の ROM ファイル *etc/ssi_pval_0* と *etc/ssi_pval_1* の内容を変更して、表示がどのように変わるか見てみましょう。

ROM ファイル *etc/ssi_pval_0* と *etc/ssi_pval_1* の内容をそれぞれ

`<td bgcolor="#fff999">0</td>` を `<td bgcolor="#fff000">OFF</td>` へ

`<td bgcolor="#33ccff">1</td>` を `<td bgcolor="#0000ff">ON</td>` へ

変更して ConDuLan にインストールしてみます。デジタル入力のページを開くと今まで 0 と表示されていたところが OFF と表示され、1 と表示されていたところが ON と表示されるようになります。また表示の背景色も変更されています。

このようにデジタル入力の表示は ROM ファイルを書き換えることによって自由に変更することができます。

デジタル入力値が LOW のときの表示定義
etc/ssi_pval_0
を
`<td bgcolor="#fff000">OFF</td>`
に変更した表示結果

デジタル入力値が HIGH のときの表示定義
etc/ssi_pval_1
を
`<td bgcolor="#0000ff">ON</td>`
に変更した表示結果

デジタル入力 SSI のデータ変更例

6. フォームの送信

いままでは SSI を利用して ConDuLan に含まれるプログラムを起動する方法を見てきました。SSI では AD 変換結果など情報を表示するには便利ですが、ConDuLan の設定を変更したり外部への出力信号を操作するようなことには適していません。通常そのような操作には PC から Web サーバへフォームを送信します。

たとえば DA 出力用 Web ページでチャンネル0の *set* をクリックすると PC から ConDuLan の */da.html* へ POST が要求されます。そのときの送信データは *da0=255* のような書式になります (出力255の場合)。ConDuLan は *da0=255* のデータを持つ POST 要求を受けるとチャンネル0の DA から指定の値 255 を出力します。このように POST 要求で起動するデータの名前(上記の *da0* の部分)には次のような DA チャンネルとデジタルポートがあります。

da0, da1
p23, p24, p25, p26, p27
p40, p41, p42, p43, p44, p45, p46, p47
p50, p51, p52, p53
p70, p71, p72, p73, p74, p75, p76, p77
p80, p81, p82, p83, p84
pa0, pa1, pa2, pa3, pa4, pa5, pa6, pa7
pb0, pb1, pb2, pb3

また、これらの POST 要求が正常に動作するページは

/ioconfig.html
/iocontrol.html
/da.html

の3種だけです。他のページにこの POST を要求しても動作しません。

これらの POST 動作する URI は ROM ファイル */etc/html_post* に記述しておきます。ConDuLan ではあらかじめ上記3ページの URI が記述されています。

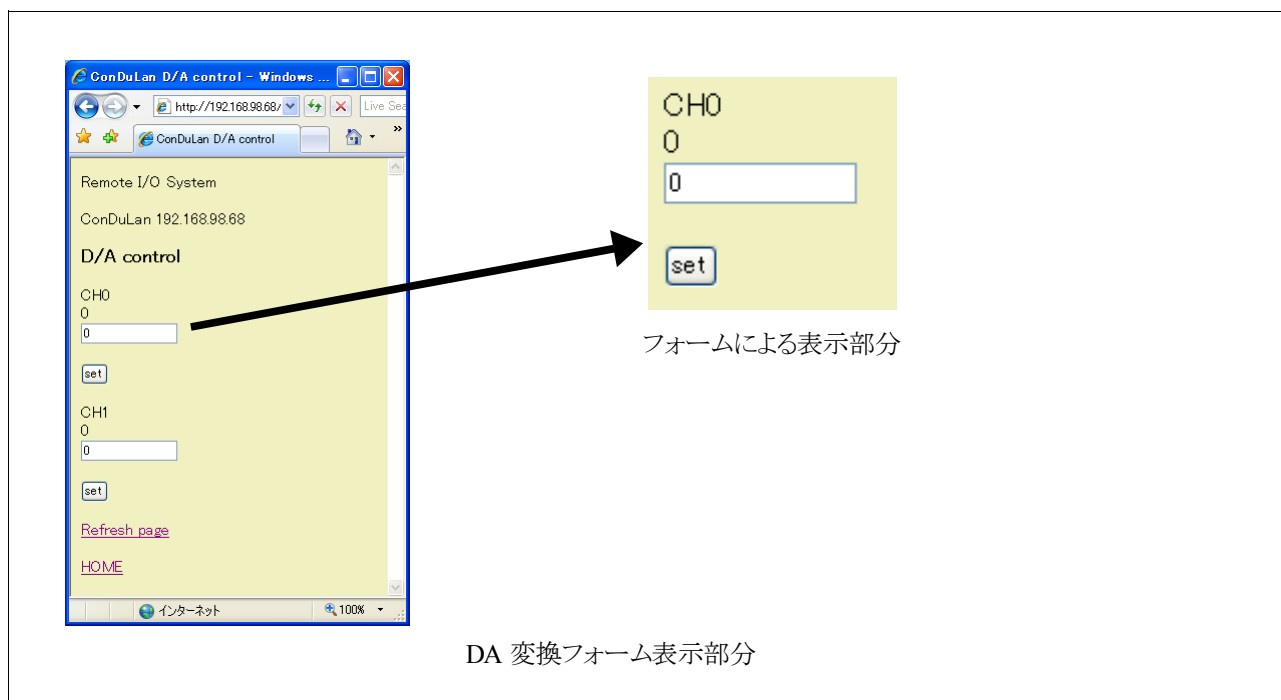
6.1 DA 変換フォーム

DA 変換を実行するにはチャンネル0用に $da0=255$ あるいはチャンネル1用に $da1=255$ のデータを POST します(出力値が255の場合). データを POST するためには通常 HTML の form タグを使用します.

たとえば次の記述はページ `da.html` に $da0=xxx$ を POST するためのフォームです.

```
<form action=/da.html method="post" enctype="text/plain">
<p>
  CH0<br>
  0<br>
  <input type="text" name="da0" value=0 size="16" maxlength="16"><br>
</p>
<input type="submit" value="set">
</form>
```

表示部分は下のようになります. 数値ウインドウに 255 を入力し, `set` をクリックすれば $da0=255$ が ConDuLan へ POST 要求として送られ, チャンネル0の DA 出力が 255(4.1V)となります.



6.2 DA 変換フォーム(現在値表示)

DA 変換フォームで数値ウインドウに表示される値が現在出力されている DA の値となっていれば便利なので、前章のフォーム記述中の `value=0` の部分を DA 出力値 SSI に書き直してみます。同様に `0
` も DA 出力値 SSI に書き直します(下の例では `action=` の部分も変更されています)。

```
<form action=
```

6.3 デジタルポート構成フォーム(機能)

次に各デジタルポート構成の Web ページを見てみましょう。デジタルポート構成ページはトップページから IO configuration をクリックして表示します。ROM ファイルは `/http/ioconfig.html` です。

各デジタルポートの設定は下図のような多数のプルダウンメニューで構成されています。

ページ下部の `set` をクリックすると各ポートの設定が `p23=IN` のようなデータの連続で ConDuLan に送られます。要求は POST となります。

ポート名はすでに説明した名称となります。値は

IN	入力
IN_PULLUP	内部プルアップ付き入力
OUT	出力
OPEN_DRAIN	LOW 出力 (HIGH はハイインピーダンス)

のいずれかになります。

ポートによっては設定できない項目があり、その場合にはプルダウンメニューには表示されません。

各ポートの構成メニュー

v で各ポートの構成要素を表示する

set をクリックすると POST 要求する

デジタルポート構成ページ

6.4 デジタルポート構成フォーム(記述)

デジタルポート構成フォームも DA 出力と同様 HTML の form タグを使用して

```
<form action=/ioconfig.html method="post" enctype="text/plain"><tt>
<p><input type="submit" value="set"></p></tt></form>
```

という記述を使用します。

上記2行の間に各ポート設定のプルダウンメニューを記述しておきます。

すべてを書きますと長くなってしまいますのでポート40の設定例を示します。

```
<form action=/ioconfig.html method="post" enctype="text/plain"><tt>
  40 <select name="p40">
    <option value="in" selected>IN</option>
    <option value="pullup">IN_PULLUP</option>
    <option value="out">OUT</option>
    <option value="open">OPEN_DRAIN</option>
  </select>
<p><input type="submit" value="set"></p></tt></form>
```

このフォームは下のような入力フォームを表示します。右はプルダウンしたところです。メニューからたとえば OUT を選んで set をクリックすると p40=OUT のデータが ConDulan へ POST 要求されます (OUT を選んだ場合)。ConDuLan 内蔵の POST 処理が p40=OUT を受け付けるとポート40を出力モードへ変更し、シリアル ROM へパラメータ p40=OUT を保存します。次回の電源投入時にはこの保存パラメータにしたがって自動的にポート40を出力に設定します。



デジタルポート構成ページ

6.5 デジタルポート構成フォーム (SSI 使用)

デジタルポート構成フォームに使用するプルダウンメニューは長い記述を必要とします。ConDuLan にはこのプルダウンメニューを生成する SSI が用意されていますので、それを利用してフォームを短く記述できます。以下の記述はポート40からポート43についての設定例です。

```
<form action=
```

6.6 デジタル出力フォーム(HIGH 出力)

次にデジタル出力 Web ページを見てみましょう。デジタル出力ページはトップページから Output control をクリックして表示します。ROM ファイルは `/http/iocontrol.html` です。

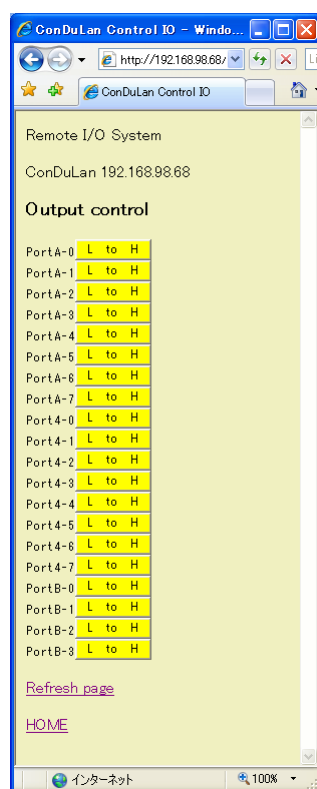
デジタル出力ページは下のような多数のクリックメニューが表示されています。デジタル出力を可能にするため、必要なデジタルポート構成をあらかじめ出力 (OUT) に設定しておきます。

表示が L to H のポートは現在出力が LOW で、クリックすると HIGH を出力するようになっています。このとき ConDuLan に POST されるフォームは、たとえばポート A0 の場合は、

pa0=L to H

となります(空白が含まれます)。

ConDuLan は POST 要求のデータに pa0=L to H が含まれている場合はポート A0 に HIGH を出力します。



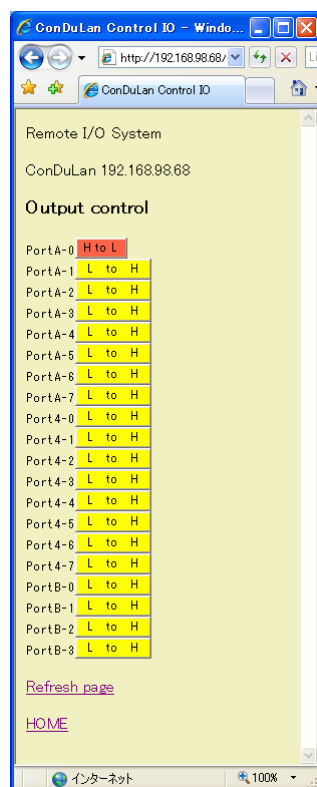
デジタル出力ページ(H出力)

6.7 デジタル出力フォーム(Low 出力)

次にデジタル出力Webページでは HIGH の出力状態にあるポートは H to L 表示となります。下の表示はポートA0が HIGH を出力している状態です。ここをクリックすると ConDuLan に pa0=H to L

が POST されます(空白が含まれます)。

ConDuLan は POST 要求のデータに pa0=H to L が含まれている場合はポートA0に LOW を出力します。



デジタル出力ページ(L出力)

6.8 デジタル出力フォームの記述

デジタル出力の Web ページに表示されているフォーム部分の記述は以下のようになります。この例ではポートA0が HIGH の状態で *H to L* が背景色 #ff6347 (赤) で表示されています。他のポートは LOW 状態で表示は背景色 #ffff00 (黄) で *L to H* が表示されています。記述中 *value* の値が表示され、またクリックにより ConDulan への POST データに使用されます。

```
<form action=/iocontrol.html method="post" enctype="text/plain">
<tt>
PortA-0<input type="submit" name=pa0 value="H to L" style="background-color:#ff6347"><br>
PortA-1<input type="submit" name=pa1 value="L to H" style="background-color:#ffff00"><br>
PortA-2<input type="submit" name=pa2 value="L to H" style="background-color:#ffff00"><br>
PortA-3<input type="submit" name=pa3 value="L to H" style="background-color:#ffff00"><br>
PortA-4<input type="submit" name=pa4 value="L to H" style="background-color:#ffff00"><br>
PortA-5<input type="submit" name=pa5 value="L to H" style="background-color:#ffff00"><br>
PortA-6<input type="submit" name=pa6 value="L to H" style="background-color:#ffff00"><br>
PortA-7<input type="submit" name=pa7 value="L to H" style="background-color:#ffff00"><br>
Port4-0<input type="submit" name=p40 value="L to H" style="background-color:#ffff00"><br>
Port4-1<input type="submit" name=p41 value="L to H" style="background-color:#ffff00"><br>
Port4-2<input type="submit" name=p42 value="L to H" style="background-color:#ffff00"><br>
Port4-3<input type="submit" name=p43 value="L to H" style="background-color:#ffff00"><br>
Port4-4<input type="submit" name=p44 value="L to H" style="background-color:#ffff00"><br>
Port4-5<input type="submit" name=p45 value="L to H" style="background-color:#ffff00"><br>
Port4-6<input type="submit" name=p46 value="L to H" style="background-color:#ffff00"><br>
Port4-7<input type="submit" name=p47 value="L to H" style="background-color:#ffff00"><br>
PortB-0<input type="submit" name=pb0 value="L to H" style="background-color:#ffff00"><br>
PortB-1<input type="submit" name=pb1 value="L to H" style="background-color:#ffff00"><br>
PortB-2<input type="submit" name=pb2 value="L to H" style="background-color:#ffff00"><br>
PortB-3<input type="submit" name=pb3 value="L to H" style="background-color:#ffff00"><br>
</tt>
</form>
```

6.9 デジタル出力フォームの記述 (SSI 使用)

前項のデジタル出力フォーム記述は現在の出力状態によって変化するため、ROM ファイルは固定の記述ではなく SSI を利用します。フォーム記述中の

```
value="H to L" style="background-color:#ff6347"
```

の部分は ConDuLan 内蔵 SSI

```
<!--#exec cgi="cgi_pctl pa0" -->
```

が生成します (ポート A0 の場合)。

このように現在のデジタル出力に対応して Web ページを変化させるために、デジタル出力の ROM ファイル `http/iocontrol.html` は次のようなフォーム記述で作られています。

```
<form action=<!--#echo var="DOCUMENT_URI" --> method="post" enctype="text/plain">
<tt>
PortA-0<input type="submit" name=pa0 <!--#exec cgi="cgi_pctl pa0" -->><br>
PortA-1<input type="submit" name=pa1 <!--#exec cgi="cgi_pctl pa1" -->><br>
PortA-2<input type="submit" name=pa2 <!--#exec cgi="cgi_pctl pa2" -->><br>
PortA-3<input type="submit" name=pa3 <!--#exec cgi="cgi_pctl pa3" -->><br>
PortA-4<input type="submit" name=pa4 <!--#exec cgi="cgi_pctl pa4" -->><br>
PortA-5<input type="submit" name=pa5 <!--#exec cgi="cgi_pctl pa5" -->><br>
PortA-6<input type="submit" name=pa6 <!--#exec cgi="cgi_pctl pa6" -->><br>
PortA-7<input type="submit" name=pa7 <!--#exec cgi="cgi_pctl pa7" -->><br>
Port4-0<input type="submit" name=p40 <!--#exec cgi="cgi_pctl p40" -->><br>
Port4-1<input type="submit" name=p41 <!--#exec cgi="cgi_pctl p41" -->><br>
Port4-2<input type="submit" name=p42 <!--#exec cgi="cgi_pctl p42" -->><br>
Port4-3<input type="submit" name=p43 <!--#exec cgi="cgi_pctl p43" -->><br>
Port4-4<input type="submit" name=p44 <!--#exec cgi="cgi_pctl p44" -->><br>
Port4-5<input type="submit" name=p45 <!--#exec cgi="cgi_pctl p45" -->><br>
Port4-6<input type="submit" name=p46 <!--#exec cgi="cgi_pctl p46" -->><br>
Port4-7<input type="submit" name=p47 <!--#exec cgi="cgi_pctl p47" -->><br>
PortB-0<input type="submit" name=pb0 <!--#exec cgi="cgi_pctl pb0" -->><br>
PortB-1<input type="submit" name=pb1 <!--#exec cgi="cgi_pctl pb1" -->><br>
PortB-2<input type="submit" name=pb2 <!--#exec cgi="cgi_pctl pb2" -->><br>
PortB-3<input type="submit" name=pb3 <!--#exec cgi="cgi_pctl pb3" -->><br>
</tt>
</form>
```

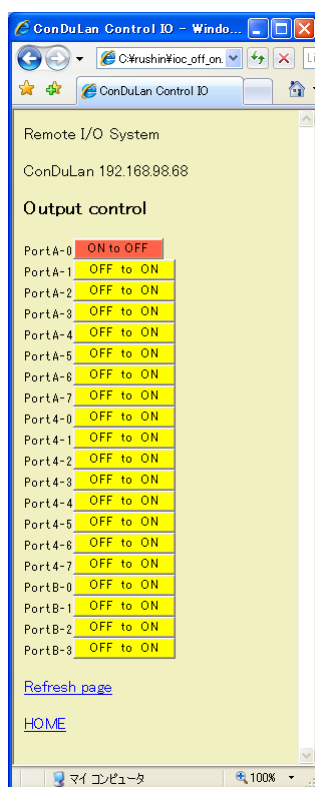
6.10 デジタル出力フォームの値

デジタル出力フォーム部分 *value=* で指定する値は LOW のとき *L to H* で HIGH のとき *H to L* でしたが他の値を使用することができます。

この2つの値はそれぞれ ROM ファイル */etc/val_pctl_0to1* と */etc/val_pctl_1to0* で定義しています。この ROM ファイルの内容を変更して ConDuLan にインストールすると下の図のような Web ページとなります。

この例では *L to H* (*/etc/val_pctl_0to1*) を *OFF to ON* へ変更し、*H to L* (*/etc/val_pctl_1to0*) を *ON to OFF* へ変更しました。

この変更により、デジタル出力の表示画面が変化し、またデジタル出力 POST データで使用する値の定義も *pa0=ON to OFF* 変更されます。



デジタル出力ページ(表示文字変更)

6.11 デジタル出力フォームの書式

今までの説明ではデジタル出力 SSI

```
<!--#exec cgi="cgi_pctl pa0" -->
```

が生成する書式は

```
value="L to H" style="background-color:#ffff00"
```

のようになっていました。

値部分の L to H は ROM ファイル/etc/val_pctl_0to1 で定義していますが, SSI が生成する記述全体の書式は ROM ファイル/etc/ssi_pctl_0to1 で定義します。出力値が HIGH の時の書式は ROM ファイル/etc/ssi_pctl_1to0 で定義します。

実際に ROM ファイル/etc/ssi_pctl_0to1 の内容を見ると

```
value="%s" style="background-color:#ffff00"
```

という内容になっています。

また ROM ファイル/etc/ssi_pctl_1to0 の内容を見ると

```
value="%s" style="background-color:#ff6347"
```

という内容になっています。

Web ページが表示されるときは SSI によって, この定義中の %s の部分に出力の値表記が書き込まれます。表示の背景色を変更する場合などに利用します。

6.12 デジタル出力フォームの書式変更例

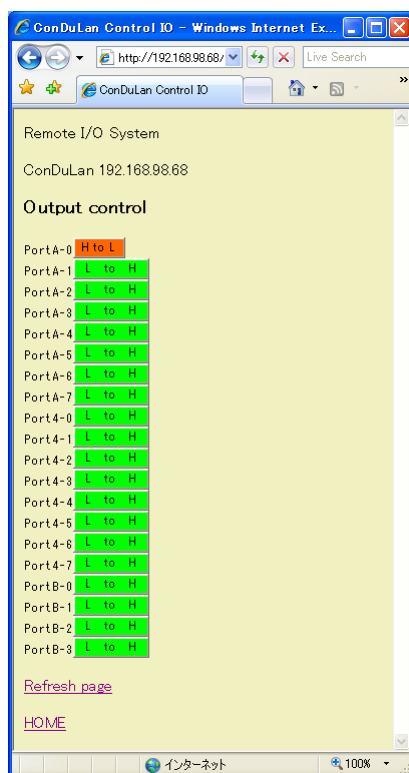
次にデジタル出力表示の背景色を変更する例を見てみましょう。デジタル出力が LOW でクリックすると HIGH になるための表示書式は ROM ファイル/etc/ssi_ctl_0to1 に書かれていますので、その背景色 #ffff00 を #00ff00 に変更し、出力 HIGH のときの書式ファイル/etc/ssi_ctl_1to0 の背景色 #ff6347 を #ff6600 に変更してみます。それぞれの ROM ファイルの内容は

```
/etc/ssi_ctl_0to1 value="%s" style="background-color:#00ff00"
```

```
/etc/ssi_ctl_1to0 value="%s" style="background-color:#ff6600"
```

のようになります。

出力値表示部分の背景色は次のようになります。



デジタル出力ページ(表示書式変更)

7. GET によるクエリー

ポート出力や DA 出力などは PC からの POST によるフォーム送信で実行されることを説明してきました。HTTP プロトコルは GET 操作でも URI にクエリーとしてデータを付加することが可能なので、理論的には GET でも ConDuLan へデータを渡すことができます。

この章では GET により ConDuLan へフォームを送る方法をご紹介します。

7.1 GET クエリーのための ROM ファイル変更

POST によるフォーム送信では *form* と */form* の間にある

```
form action=<!--#echo var="DOCUMENT_URI" --> method="post" enctype="text/plain">
```

の記述が POST 動作を指示していました。GET でフォームを送信するにはこの中の *post* の部分を *get* に変更して

```
form action=<!--#echo var="DOCUMENT_URI" --> method="get" enctype="text/plain">
```

のように記述します。

たとえば DA 出力の ROM ファイル */http/da.html* の *form* 部分にある *post* を *get* へ変更します。DA チャンネル0の *set* をクリックすると ConDuLan へは POST によるフォーム送信ではなく、次のようなクエリー付きの GET 要求が送信されます (値が255の場合)。

```
http://192.168.98.68/da.html?da0=255
```

ConDuLan 内蔵のクエリー処理は *da0=255* により DA チャンネル0に255を出力するよう要求されたことを検出します。

ただし、すべての Web ページがこのクエリー処理をおこなうわけではありません。POST 処理の場合に POST 動作するページを ROM ファイル */etc/html_post* で指定したように GET のクエリー処理をおこなう Web ページを ROM ファイル */etc/html_get* で指定します。ディレクトリ *etc* にある ROM ファイル *http/html_get* をテキストエディタで開くと次のような内容が書かれているはずで

```
#get cgi is applied to following urls  
#ioconfig.html  
#iocontrol.html  
#da.html
```

URL 記述の先頭の *#* を取り除いて

```
#get cgi is applied to following urls  
/ioconfig.html  
/iocontrol.html  
/da.html
```

に変更し ROM ファイルをインストールすると DA 変換を GET 手順でおこなうことができます。

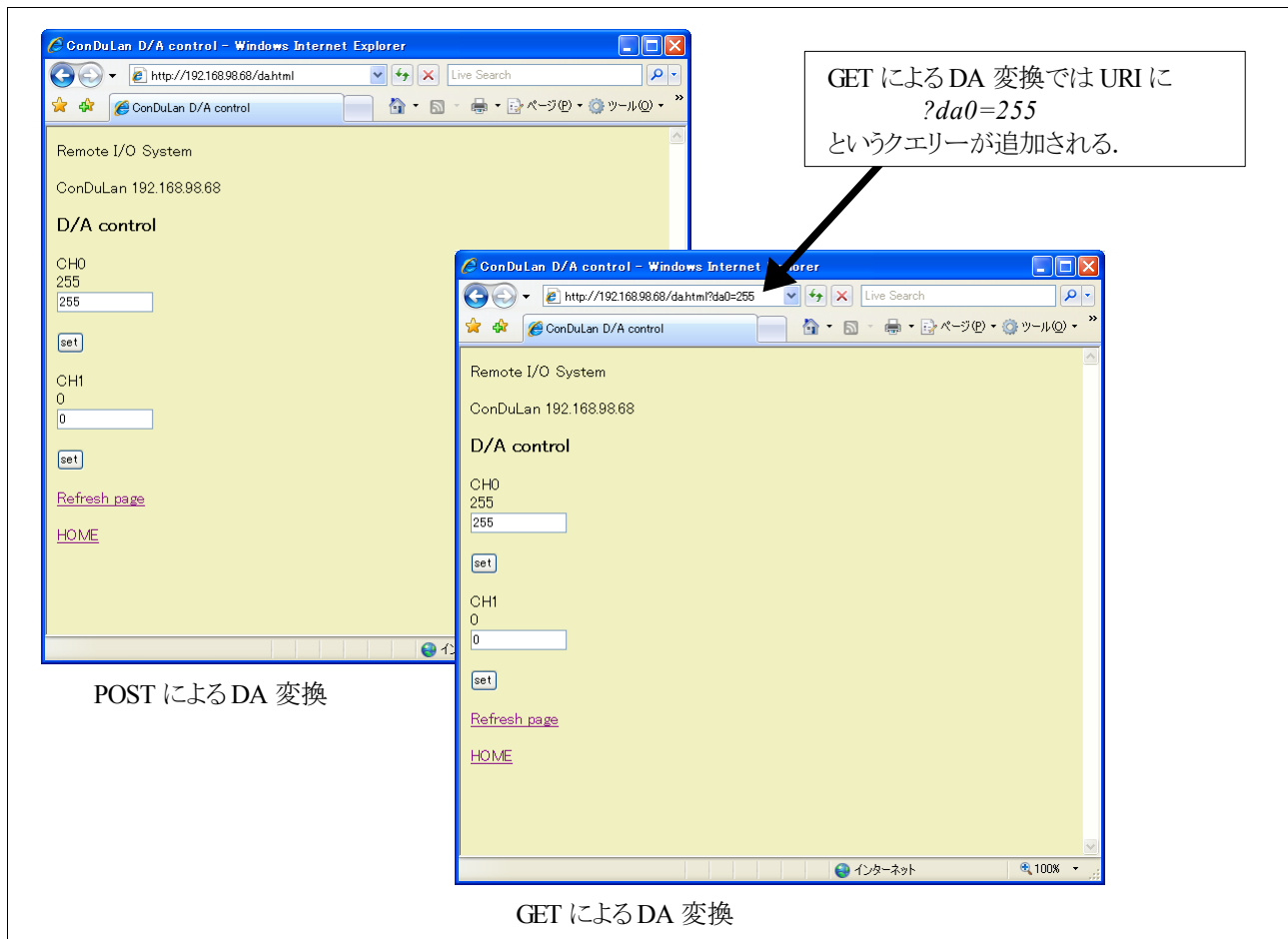
DA 変換の *form* 部分は次のようになります (チャンネル0の部分)。

```
<form action=/da.html method="get" enctype="text/plain">  
<p>  
CH0<br>  
0<br>  
<input type="text" name="da0" value=0 size="16" maxlength="16"><br>  
</p>  
<input type="submit" value="set">  
</form>
```

7.2 GET クエリーの Web 画面

DA 出力制御は POST によるフォーム送信も GET によるクエリー送信も操作上はほとんど同じです。

下の2つの画面は POST による DA 出力と GET による DA 出力の画面です。



7.3 form 送信と Web アクセス制限の関係

DA 変換のように ConDuLan へフォームを送る手段について POST と GET の2種類を説明してきました。この2種類の使い分けは ConDuLan の Web アクセス制限と関連しています。

ConDuLan の Web サーバはパスワード認証や IP アドレスなどでアクセスを制限することができます。

すべてのアクセスを制限するには GET と POST の両方を制限対象に設定します。

デジタル出力などの操作は禁止するが、現在のデジタル出力値は誰でも読み出せるようにするなど、「操作は制限ありー読み出しは制限なし」の場合は、GET は制限なし、POST はアクセス制限ありと設定します。もちろんデジタル出力などは POST で動作するように ROM ファイルを設定しておきます (ConDuLan の初期インストール状態です)。

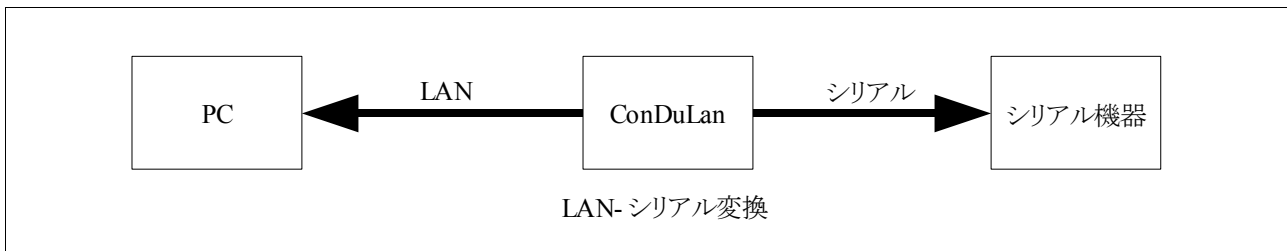
デジタル出力は制限なしで誰でも実行できるようにしておき、デジタルポートの構成 (各ポートを入力にするか出力にするかなど) はアクセス制限ありとする場合は、GET 制限なし、POST 制限ありとして、デジタル出力は GET によるクエリーで操作するようにしておきます。ポート構成操作は POST で動作するようにしておきます。

このように ConDuLan に対する操作を POST とするか GET とするかで Web アクセス制限を有効にするか誰でも操作できるようにするかに設定することができます。

なお、ConDuLan 内蔵の LAN 設定と Web アクセス制限の2つの CGI は設定を GET クエリーでおこなっていますが、POST アクセスに制限が設定されている場合はアクセス制限が有効になる特殊な動作をします。

8. LAN-シリアル変換

ConDuLan には PC から telnet 手順で LAN 接続するとシリアルポート経由の通信ができる LAN-シリアル変換機能が用意されています。この機能は Windows のハイパーターミナルなどでシリアル機器と通信する場合に便利です。



8.1 LAN-シリアル通信動作許可

ConDuLan の LAN-シリアル通信は初期状態では動作しないように設定されています。動作を許可するには ROM ファイルの *etc/html_telnetd* に LAN-シリアル通信の設定 Web ページ名を用意する必要があります。ConDulan の ROM ファイル *etc/html_telnetd* には次のような記述がありますので

```
#url for telnetd settings
#/telnetd_settings.html
```

次のように変更して LAN-シリアル設定ページを */telnetd_settings.html* にします。

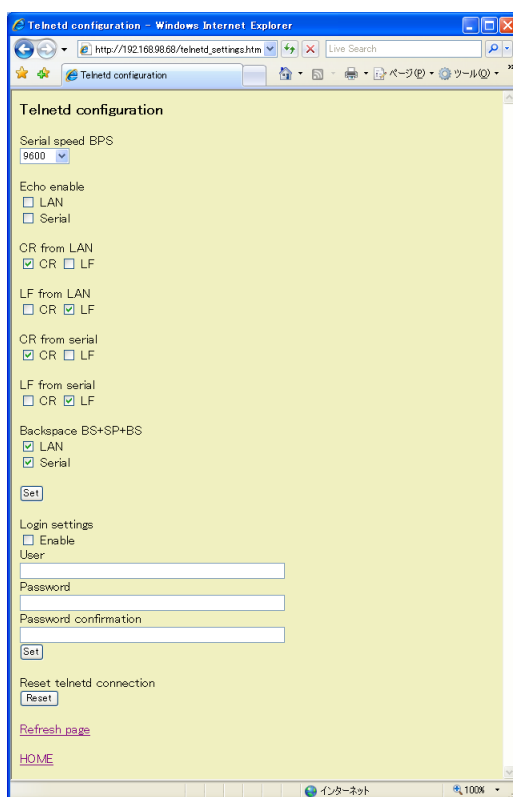
```
#url for telnetd settings
/telnetd_settings.html
```

この ROM ファイルをインストールすれば LAN-シリアル通信をおこなうことができます。

LAN-シリアル通信に必要な設定は Web ブラウザで次のようにアクセスします。

```
http://192.168.0.254/telnetd_settings.html
```

設定画面は ConDuLan 内蔵アプリケーションによる CGI で次のような表示となります。



LAN-シリアル通信設定画面

8.2 LAN-シリアル設定ページへのリンク

LAN-シリアル通信設定ページ名は ROM ファイル *etc/html_telnetd* で指定しますが, ConDuLan のトップページに LAN-シリアル設定ページへのリンクが用意されていれば便利です.

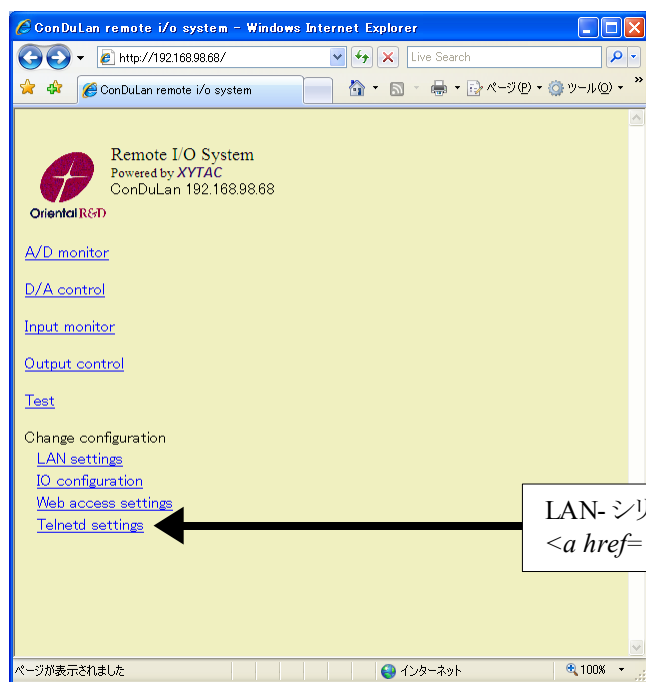
たとえばトップページの最下段に *Telnetd_settings* という表示で LAN-シリアル設定ページへのリンクを用意してみます (ページ名は */telnetd_settings.html* とします). トップページの ROM ファイル *http/index.html* 中の下部にある

```
<tr><td width="5"></td><td><a href="access_control.html">Web access settings</a></td></tr>
```

の下に

```
<tr><td width="5"></td><td><a href="telnetd_settings.html">Telnetd settings</a></td></tr>
```

を追加します. トップページは次のようになります.



LAN- シリアル通信設定へのリンク

```
<a href="telnetd_settings.html">Telnetd settings</a>
```

LAN- シリアル通信設定へのリンク

8.3 LAN-シリアルホストプログラム

ConDuLan の LAN-シリアル通信はポート 23 で TCP コネクションを待ち受けていますので、ホストが Unix あるいは Linux の場合は telnet を使用することができます。ホストが Windows の場合はハイパーターミナルを接続方法 TCP/IP (Winsock) で使用できます。

8.4 LAN-シリアル通信設定

ConDuLan の LAN-シリアル通信設定にはさまざまな環境で動作できるよう多くの設定を変更できるようになっています。以下に示す項目を設定できます。

- シリアル 転送速度 2400-57600BPS
- エコーバック LAN, シリアル独立設定
- 改行コード変換 CR および LF を変換, LAN, シリアル独立設定
- バックスペース動作 スルーあるいは BS+スペース+BS, LAN, シリアル独立設定
- ログイン認証 ユーザおよびパスワード, LAN 側のみ
- コネクション切断 Web メニューより切断, LAN 側のみ

The screenshot shows the 'Telnetd configuration' web page in Internet Explorer. The page contains several configuration sections:

- Serial speed BPS:** A dropdown menu currently set to 9600. A callout points to this dropdown with the label 'シリアル転送速度'.
- Echo enable:** Two checkboxes, 'LAN' and 'Serial', both of which are unchecked. A callout points to the 'LAN' checkbox with the label 'エコーバック'.
- CR from LAN:** Two checkboxes, 'CR' (checked) and 'LF' (unchecked). A callout points to the 'CR' checkbox with the label '改行コード変換'.
- LF from LAN:** Two checkboxes, 'CR' (unchecked) and 'LF' (checked). A callout points to the 'LF' checkbox with the label '改行コード変換'.
- CR from serial:** Two checkboxes, 'CR' (checked) and 'LF' (unchecked). A callout points to the 'CR' checkbox with the label '改行コード変換'.
- LF from serial:** Two checkboxes, 'CR' (unchecked) and 'LF' (checked). A callout points to the 'LF' checkbox with the label '改行コード変換'.
- Backspace BS+SP+BS:** Two checkboxes, 'LAN' (checked) and 'Serial' (checked). A callout points to the 'LAN' checkbox with the label 'バックスペース'.
- Login settings:** An 'Enable' checkbox (unchecked) followed by input fields for 'User', 'Password', and 'Password confirmation'. A callout points to the 'Enable' checkbox with the label 'ログイン認証'.
- Reset telnetd connection:** A 'Reset' button. A callout points to this button with the label 'コネクション切断'.

At the bottom of the screenshot, the text 'LAN-シリアル通信設定画面' is centered.

8.5 LAN-シリアル注意事項

LAN コネクション

複数の PC から同時に LAN-シリアル接続はできません。PC からコネクションを終了してから別の PC よりコネクションをする必要があります。通信経路障害など何らかの理由で終了手続きができない場合は LAN-シリアル設定画面より *Reset* をクリックすることにより強制的にコネクションを終了できます。

シリアル制御線

シリアルポートの制御線 (CTS,RTS など) は使用できません。

シリアルフローコントロール

フローコントロールはできません。

シリアル接続

RX,TX,GND だけが接続されています。

iomacro 使用時

iomacro のプリントポートとして使用する場合は LAN-シリアル機能は使用しないで下さい。

9. etc の ROM ファイル

最後に *etc* ディレクトリにおかれている ROM ファイルの役割をまとめておきます。

<i>val_pctl_1to0</i>	デジタル出力を LOW にするときの文字(H to L)
<i>val_pctl_0to1</i>	デジタル出力を HIGH にするときの文字(L to H)
<i>ssi_pval_1</i>	デジタル入力が HIGH のときの表示文字
<i>ssi_pval_0</i>	デジタル入力が LOW のときの表示文字
<i>ssi_pctl_1to0</i>	デジタル出力を LOW にするときの表示書式
<i>ssi_pctl_0to1</i>	デジタル出力を HIGH にするときの表示書式
<i>html_post</i>	POST 処理を許可する Web ページ名
<i>html_lan</i>	LAN 設定 CGI の名称
<i>html_get</i>	GET でクエリー処理を許可する Web ページ名
<i>html_access</i>	Web アクセス制限設定 CGI の名称
<i>html_telnetd</i>	LAN-シリアル通信設定 CGI の名称
<i>vender</i>	現在使用されません。Web ページ作成ベンダー名覚書です。
<i>system_name</i>	現在使用されません。Web ページ名称覚書です。

おわりに

ROMファイルの修正で可能な ConDulan のWebデザイン変更手順について説明してきました。さらに柔軟で多くの機能を実装するために内蔵簡易スクリプト言語 `iomacro` を使用することができます。詳細は「操作ガイド — — — `iomacro` 編」をご参照ください。

ConDuLan

操作ガイド ー ー ー ROMファイル編

発行年月日 2008年3月24日 第02版C

発行 オリエンタルアールアンドディー株式会社

著作 オリエンタルアールアンドディー株式会社

Printed in Japan

オリエンタルアールアンドディー株式会社

<http://www.ord.co.jp>