Control ACQ196/ACQ216 from MS Windows

MS Windows User Guide

Rev	***Date***	***Description***
1	20060912	First release
2	20061102	No edit /etc/postshot.d

Document created using OpenOffice.Org www.openoffice.org.
PDF rendition by extendedPDF, www.jdisoftware.co.uk

This document and D-TACQ Software comprising platform Linux port, Linux kernel modules and most applications are released under GNU GPL/FDL:

Document:

Software:

Table of Contents

1 Introduction

D-TACQ ACQ196CPCI/ACQ216CPCI cards may be configured as self contained networked appliances. This allows control, monitoring and data transfer using standard TCP/IP on Ethernet. Control of the cards is possible with no device drivers required, and, in some circumstances with no software required.

1.1 Scope

This document covers control and data handling for ACQ196, ACQ216 cards in the networked data acquisitions mode,

1.2 References

1. D-TACQ_2G_UserGuide.pdf (2GUG)
2. emweb : Embedded Web Pages Guide (eweb.pdf)
3. Dt100rcUserGuide.pdf

1.3 Definitions:

1. `ACQxxx` : generic term covers D-TACQ 2G cards including ACQ196CPCI, ACQ216CPCI
2. Host: host computer (in the embedded development sense) - provides a control and data storage gateway to the card. In the context of this document, this is assumed to be a PC running MS-Windows. The control and store functionality may be split over multiple Hosts.
3. Target: target computer (in the embedded development sense) - the ACQxxx card.
4. `target-ip` : the IP address of an ACQxxx target device.

2 Basic Ethernet Connectivity

2.1 Initial login session on serial console

It may be necessary to log in on the console to configure initial Ethernet settings.

ACQxxx features an RS232 console port (38400 baud, 8 bit, no parity).

Connect using a 9Way_D null modem cable and terminal emulation software.

D-TACQ recommends kermit or kermit-95.

It is possible to use Hyperterminal, but we cannot recommend this product.

For computers that no longer have serial ports, we recommend using a low-cost USB serial port adapter (less than £20 from Maplin), these hot plug and work out the box (at least on Linux hosts).

2.2 ip address selection

ACQxxx is set at the factory to use DHCP by default.

This can be changed by modifying the boot file /ffs/rc.local ([1] 6.2), either by editing with the embedded vi editor or (for those unfamiliar with vi) :

Use command /sbin/set.static_ip to change to a static ip address:

eg

```
/sbin/set.static_ip IP0 10.0.0.196
```

2.3 remote terminals

It is useful to be able to log into the ACQxxx card. This can be done via RS232 on the serial console, or via SSH over ethernet.

puTTY is a good freeware Windows ssh implementation:

http://www.chiark.greenend.org.uk/~sgtatham/putty/

2.4 embedded web pages

A set of diagnostic web pages is provided at http://ipaddress/cgi-bin.

See [1] # 7.3. and [2].

2.5 dt100rc

dt100rc is D-TACQ's cross platform reference "Remote Control" application. The application will run on Windows and Linux, requires Java 1.5 runtime. See [3] .

3 Control Via Web Service

3.1 precompiled programs

Precompiled SOAP client executables are available from:

http://www.d-tacq.com/swrel/SOAPI_WIN-0609122040.zip

For documentation please see

http://www.d-tacq.com/resources/SOAPI/html/index.html

A simple example:

set ACQ2XX_WS_URL=http://www.d-tacq.com:6666

acq2sh 'acqcmd setInternalClock 500000'

Two example configuration .BAT batch files are provided:

```
rem this is a batch file setInit.bat
set ACQ2XX_WS_URL=http://192.168.0.154:6666

acq2sh acqcmd setChannelMask
1111111111111111111111111111111111111111111111111111111111110000000000000000
00000000000000000
acq2sh acqcmd setMode SOFT_TRANSIENT 50000
rem set these to trigger from TRG LEMO
acq2sh set.route d3 in lemo out fpga
rem acq2sh set.trig DI3 falling
rem set an initial shot number. This will clock up one on setArm
acq2sh set.dtacq_drv shot 9999
```

```
rem this is a batch file
set ACQ2XX_WS_URL=http://192.168.0.154:6666
acq2sh acqcmd setArm
```

3.2 Using MS Visual Studio and VB .NET 1.0:

The acq2sh webservice is published at

`http://target-ip/cgi-bin/acq2sh.wsdl`

The MS .NET v1.0 environment is able to automatically generate client side stubs from this definition.

1. MS Visual Studio .NET: File New Project : VB.NET console application.
2. Add New Web reference : enter `http://target-ip/cgi-bin/acq2sh.wsdl`, wait for VS to parse the wsdl.
3. Create an instance of the Web reference class in your application:

```
example: accessing the web service from VB.NET
Module Module1
    Sub Main()
        'autogenerated from web reference http://acq196_001/cgi-bin/acq2sh.wsdl
        Dim service As New acq196_001.acq2xx
        Dim result As String
        Dim command As String
        'a more practical example might read commands from script file
        command = "acqcmd setInternalClock 500000;acqcmd setArm"
        result = service.acq2sh(command)
        System.Console.Out.WriteLine("command: " + command)
        System.Console.Out.WriteLine("result : " + result)
    End Sub
End Module
```

Multiple commands may be executed in one network transaction, separate the commands with semicolon (';').

4 Strategy for remote status polling.

A very common use case is as follows:

- Enable a hardware start trigger and arm the board.
- Poll for capture completion.

However, there may be a long delay before the hardware trigger (if it comes at all).

So the polling process can be active for a prolonged period. There are several ways to do this polling

4.1 Simple demand poll from the host:

acq2sh acqcmd getState

Good: command returns "immediately".

Bad: big trade off between responsiveness and network/target loading. The target is capable of about 10 SOAP transactions per second, and a fast polling action may cause the action we are waiting for (eg data transform) to be slowed down as a result

4.2 Target side polling

acq2sh acqcmd --until ST_STOP

This command will poll rapidly on the target, returning when done.

Good: rapid response to state change

Bad: network transaction subject to possibly long delay. Network timeouts can occur. Impossible to tell directly if indeed the equipment is still functioning.

4.3 Target side poll with timeout

acq2sh acqcmd -t TIMEOUT --until ST_STOP

The command will run for max TIMEOUT seconds before returning, or it will return immediately the state is reached.

This gives the best possible combination of low rate network traffic and fast response to state change.

Recommended setting:

acq2sh 'acqcmd -t 2 --until ST_STOP;acqcmd getState'

The 2s timeout gives a reasonable heartbeat response with minimal network/target loading. The second call to getState is necessary to give a state indication in the case of timeout.

5 Data upload via FTP

5.1 Install and configure FTP server

1. Install an FTP server (we recommend FileZilla : http://filezilla.sourceforge.net/)
2. Set up a non-privileged user for ftp user (we use user dt100, pass dt100
3. Run FileZilla interface, set dt100 as an authorized user, set password and home directory.

   eg we used C:\Documents And Settings\All Users\Documents\dt100ftphome

5.2 Configure ACQxxx for automated post-shot data upload.

5.2.1 Post Shot Processing Summary

After the shot, the embedded control software runs a script : `/usr/local/bin/acq200.pp`

It is possible for the user to modify the post shot operation by editing this file. However, in many cases, the required effect can be achieved without any editing at all.

Standard processing does some clean up - for example, identifies the exact position of any event (trigger). Then the data is transformed from [sample][channel] to [channel][sample] order. After the transform, any file in the directory /etc/postshot.d is executed.

The is an opportunity for remote clients to leave "stored procedures" to be executed when the data is available.

Depending on data set size, the "transform" can take up to 60s to execute. Frequently it is a requirement to view at least a snapshot of the data with minimum latency after the shot. The postprocessing script caters for this by executing any "stored procedure" files in /`etc/postshot0.d` *before* the transform is executed.

5.2.2 Stored Procedure rules

- any file in /etc/postshot.d is executed after Transform
- any file in /etc/postshot0.d is executed before Transform.
- files are typically shell scripts, but can also be custom xscale executables.
- if the file has execute permission, it is executed in its own process in the normal way.
- if the file does not have execute permission, the file is included "sourced" as part of the same shell process running `acq200.pp`. Clearly, the contents of this file must be valid shell script commands. The entire environment of `acq200.pp`, including any shell functions is inherited by the stored procedure, so this can be very brief.
- Please avoid using any spaces in the file name - the file system will let you, but the acq200.pp will likely have unpredictable behaviour

5.2.3 Example: file per channel using ftp client app

1. Create /root/.netrc with login information. (see example below)
2. Set protection on /root/.netrc : chmod 600 /root/.netrc
3. Configure a custom post shot script in /etc/postshot.d
4. When the in-place files have been tested, make a non-volatile copy as follows:

   cp /root/.netrc /ffs/dropbear/root/.netrc
5. Store a non-volatile copy of the stored procedure in

   /ffs/dropbear/etc/postshot.d

   - or -

   create stored procedure on the fly before the shot. (see following example).

```
Sample .netrc

machine 192.168.0.1
login dt100
password dt100

Sample stored procedure file:
/etc/postshot.d/my-ftp-upload-file
doFtpUpload  192.168.0.165
```

5.2.4 Example single block file upload using curl

Example shows use of curl to upload all channel data to a single file on the ftp server.

The data in the single file is optimally ordered - blocks of data comprising the full time-series for each channel, blocks in logical channel order. The curl program does not use .netrc, and login credentials are supplied on the command line.

```
From a remote client:

# assume ftp server  172.16.68.1
# assume UID dt100 / pass dt100

acq2sh -u URL 'echo blockFtpUpload 172.16.68.1 dt100:dt100
          >/etc/postshot.d/post-shot-ftp'

# creates this file:
/etc/postshot.d/post-shot-ftp:
blockFtpUpload 192.168.0.165 dt100:dt100

# data is stored in a single file on the ftp host: $SHOT.dat where $SHOT was
formatted using "%06d". The data order is:
ch01[nsamples],ch02[nsamples], .... ch64[nsamples]
```

6 Appendix: Editing configuration files.

The ACQxxx makes extensive use of text based shell scripts for configuration. The scripts can be customised either by using :

- embedded editor (vi).
- use ftp to copy to a pc, edit there using an editor you are comfortable with, then copy back. The ftp copy should be made in TEXT mode, so that any CRLF sequences added by Windows text editors are converted back to native UNIX LF's.
- sometimes dedicated commands are available to change configurations eg `/sbin/set.static_ip`