

CMG-DCM / AM

Data Communications Modules

User's guide

Part No. MAN-DCM-0001

Designed and manufactured by
Güralp Systems Limited
3 Midas House, Calleva Park
Aldermaston RG7 8EA
England

Proprietary Notice: The information in this manual is proprietary to Güralp Systems Limited and may not be copied or distributed outside the approved recipient's organisation without the approval of Güralp Systems Limited. Güralp Systems Limited shall not be liable for technical or editorial errors or omissions made herein, nor for incidental or consequential damages resulting from the furnishing, performance, or usage of this material.

Issue F 2006-12-13

Table of Contents

1 Introduction..... 5
1.1 Inside the DCM..... 8

2 Installing the CMG-DCM..... 11
2.1 Overview..... 11
2.2 Power supply..... 11
2.3 Connecting to a single computer..... 12
2.4 Connecting to a local network..... 15
2.5 Connecting digitizers and external hardware..... 18
2.6 Setting up the DCM..... 19

3 Usage..... 24
3.1 General notes..... 24
3.2 The DCM as a data store..... 24
3.3 The DCM as a GCF data source..... 28
3.4 The DCM as a network data hub..... 32
3.5 Troubleshooting DCM installations..... 38
3.6 CMG-AM Authentication Modules..... 43

4 Tools..... 51
4.1 Summary..... 51
4.2 Process Overview..... 54
4.3 Data Viewer..... 55
4.4 Data/Status Summary..... 57
4.5 Disk files..... 59
4.6 Flash files..... 60
4.7 Disk tools..... 60
4.8 Camera..... 62
4.9 Recent Log Entries..... 62

5 Data transfer..... 64
5.1 Scream! server..... 64
5.2 SeedLink..... 66
5.3 DSS..... 68
5.4 CD1 (CD1.0) Sender..... 70
5.5 CD1.1..... 72
5.6 CNSN configuration (cnsn.cfg)..... 74

5.7 SEED name mappings..... 75
5.8 AutoDRM..... 78
5.9 HTTP server..... 79

6 Configuration..... 80
6.1 General..... 81
6.2 Serial ports..... 82
6.3 Disk..... 86
6.4 SEED recorder..... 91
6.5 Ethernet port..... 94
6.6 Static routes..... 95
6.7 DNS setup..... 96
6.8 Incoming mail setup..... 97
6.9 Outgoing mail setup..... 98
6.10 Remote access..... 99
6.11 PPP..... 100
6.12 mgetty configuration..... 102
6.13 Administrators..... 103

7 Configuring digitizers..... 105
7.1 General digitizer settings..... 106
7.2 Digitizer output control..... 107
7.3 Trigger criteria..... 109
7.4 Auxiliary (“Mux”) channels..... 112
7.5 Sensor mass control..... 113

8 Inside the DCM..... 114
8.1 File systems..... 116
8.2 Command line tools..... 117
8.3 Configuration..... 118
8.4 Monitoring..... 121
8.5 Updating the DCM..... 123

9 Connector pinouts..... 127
9.1 Modular DCM units..... 127
9.2 Integrated DCM units..... 129

10 Sensor and digitizer types..... 130
10.1 Sensor response codes..... 130
10.2 Digitizer type codes..... 131

11 Revision history..... 132

1 Introduction

The CMG-DCM is a versatile Linux-based module for storing and transmitting data captured using Güralp Systems Limited's range of seismic measuring equipment. GCF data can be gathered from up to three compatible digitizers or digital instruments, and stored in its on-board Flash memory, from where it is written from time to time to a USB hard disk or to another device on your network. Once data is on the hard disk, you can connect to the DCM in various ways to retrieve it.

Depending on your site requirements, the DCM may be supplied in several formats. Although these look different, they all share the same internal features.

- A *stand-alone* DCM, housed in a high-impact copolymer polypropylene case, may be connected to a digitizer through mil-spec connector cables and installed in a ground station or other location remote from the digitizer and seismometer. The stand-alone DCM can accept data from up to three devices connected to it through RS232 ports.



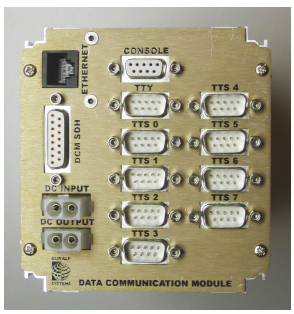
A stand-alone DCM unit is normally supplied with a high-capacity removable disk for data storage. These disks allow the DCM to be used as part of completely autonomous installation in cases where communications links are poor or non-existent. Manufactured and tested at Güralp Systems, the disks are compatible with USB and FireWire standards and include an internal temperature sensor and heater for use in cold environments.



You can retrieve data from a DCM's disk at any time by removing it from the module and attaching it to a PC using a USB or FireWire cable. Swapping disks does not cause any risk to data.

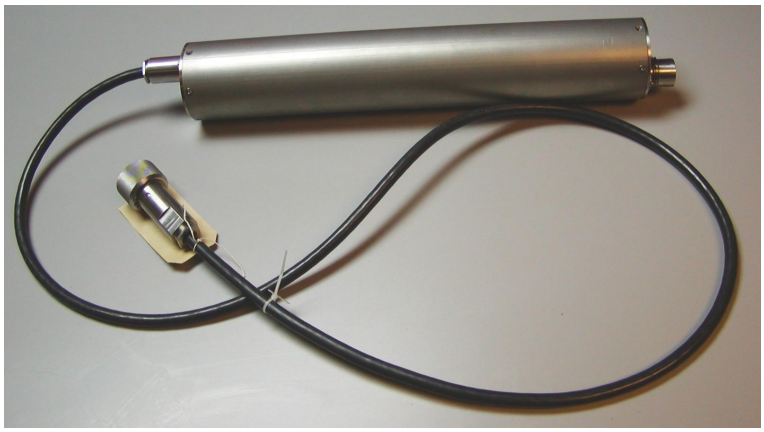
- A stand-alone DCM can also be supplied in a form factor suitable for mounting in a standard 19” rack. Such a DCM will normally come equipped with eight additional serial ports TTS0-7, and an optional State of Health (SOH) interface for monitoring or tamper evidencing.





Up to four modules may be mounted side-by-side in a 19” rack.

- An *integrated* or *modular* DCM may be combined with a digitizer and seismometer to produce a single unit capable of measuring seismic data, storing, and passing it on over a network:



This form of DCM uses an integrated 26-pin connector to communicate, rather than having separate ports.

A DCM in any of these forms may be supplied in the configuration most suitable to your requirements, depending on how you wish it to be connected to your network, and how its USB interface is to be used. The two sets of options are independent.

- Using the *10BaseT Ethernet* network option, you can connect the DCM to any TCP/IP network.
- Alternatively, an internal *PCMCIA modem* may be supplied, through which you can use a dial-up or other communications link to connect to your home network. Most 56k analogue and ISDN-type modems are supported.
- DCMs can also be supplied with internal *satellite modems.*
- As a fourth option, an *802.11b Wi-Fi* module may be installed.
- If the USB *client* option is installed, you can connect the DCM's USB port to a computer and communicate with it as if it were on a private network.
- If the USB *host* option is installed, you can connect the USB port to an additional peripheral, such as a camera or external hard disk.

Whichever set of options you choose, you can use any free RS232 port to connect the DCM to a single computer for local monitoring and configuration, or through it to a wider network using PPP/SLIP.

1.1 Inside the DCM

The DCM's configuration is maintained by an internal database. All of the flexibility of the unit is provided through this configuration database. Before you deploy the DCM, you will need to configure it for your needs, either using its on-board Web server or over a direct serial link.

The two main areas which need to be configured determine the different services which the DCM provides from its serial and network ports.

Serial port services

You can use either the DCM's console or its Web configuration interface to configure the serial data ports. The **serial.x.service** configuration option determines which service each port provides.

- The simplest service is *getty*. This creates a console port, which you can use to log in to the Linux operating system of the DCM directly.

- The *mgetty-r* service is similar, but is more suited for serial links including modems, in which the link is not always active. This is the default option.

- Any of the serial ports on the DCM can be directed to transmit GCF (Güralp Compressed Format) data by setting the service for that port to *gcf_out*. You can then run Scream! or other software on your computer to receive the data. You will not be able to access the Web configuration interface using this service, although you may be able to configure attached digitizers from Scream!.

- Likewise, any serial port can be set to listen for incoming data (*e.g.* from a connected digitizer or another DCM) by setting the service to *gcf_in*. The DCM will automatically combine this data with any other streams it is receiving, and send it on using all *gcf_out* ports and any other transmission methods which have been configured.

- You can log in to the Linux operating system over a modem link by setting the service to *mgetty*. *mgetty* listens for incoming calls from your computer and sets up a login session for you. You may need to configure `minicom` or `hypertrm` to communicate with your own modem before you can do this.

- Both *mgetty* and *mgetty-r* can recognize incoming PPP connections and will automatically run a PPP daemon if you attempt to connect to the DCM in this way.

- Specialist services *cnsn_in*, *cnsn_out* and *dss_sum* are also provided.

Network services

Because the DCM is a fully-fledged Linux machine, you will need to set up networking before you can use it. Advanced networking is beyond the scope of this manual. For a basic setup, you will need to perform the following actions:

1. Assign the DCM an IP address. By default, the DCM will try and use DHCP (Dynamic Host Configuration Protocol) to find its own IP address. However, this requires a DHCP server on your network. If you do not have a DHCP server, you will have to set the IP address yourself.

2. Find out the IP address the DCM is using (if you have not set it yourself).

3. If necessary, configure your own computer's IP address so that it can communicate with the DCM.

See Section 2.4, page 15, for full details.

Once you have connected the DCM to your local network or to the Internet, you can use a wide range of methods to retrieve data from it.

- All DCM units feature an on-board Web server, which allows you to monitor and configure the station from any browser (and download data by HTTP, if enabled.)

- For maintenance, you can log in to the operating system directly over the secure shell protocol, SSH, and use all the standard Unix/Linux tools.

- A GCF server can be set up to transmit data to any application that supports the GCF format (such as Scream!, Antelope, or Earthworm.)

- The DCM can act as a CD1.0 or CD1.1 transmitter to a specified client, or to multiple clients as an option.

- Optionally, the DCM can also receive and process AutoDRM

messages.

In addition, the DCM can be configured to act as a data client, receiving GCF and CD1.0/1.1 data streams, combining these streams and storing or retransmitting them as appropriate. See Chapter 3, page 24, for full details.

2 Installing the CMG-DCM

2.1 Overview

The DCM is effectively a low-power Linux computer dedicated to seismic data flow tasks. Resources permitting, the DCM can perform any computational task you require. Because of this flexibility, the DCM must be configured for your particular purposes before it can be used:

1. If you only need to use the DCM as a data store, you can gain console access over the serial port for configuration.

2. If you are going to use the DCM as part of a TCP/IP network, you must set up its IP address and other networking parameters before you can connect to it. Whilst the network is inoperative, you will need to connect to the DCM console over a direct serial connection. We recommend that you configure the DCM correctly, to the best of your knowledge, *before* shipping it into the field.

3. If you want to connect to the DCM using TCP/IP over modem links, you must set up PPP over these links. Configuring PPP is beyond the scope of this manual: please see the Linux `man` page for `ppp` for details, or the Linux PPP HOWTO.

4. Once communication with the DCM is working, you should set up the serial ports on the DCM to provide the services you require.

5. When all ports are configured, you can use the DCM to set up attached Güralp digitizers and digital instruments.

2.2 Power supply

The *stand-alone* DCM receives its power from lines in its data connectors. All three of these ports are Güralp standard 10-pin combined serial/power interfaces. Cables are available from Güralp Systems which enable you to connect a 9-pin D serial interface and 2-way power connector to any of these ports.

You can power the DCM from whichever data port is most convenient, whether or not it is being used for receiving or transmitting data. However, you should always supply power to the *DATA OUT* plug if

possible, since it is easy to accidentally short the pins of a male connector. Attached digitizers and sensors are powered from the same supply as the DCM. Any 12 V DC power supply may be used; if using batteries, we recommend that you use a unit with low-voltage disconnect to avoid excessive drain.

A *rack-mounted* DCM has a separate *POWER IN* connector on the back panel, which should be attached to a 12 V DC supply. There is also a *POWER OUT* connector on this panel, which allows you to daisy-chain several DCM units together. The *POWER OUT* connector is provided merely for convenience: lines from the *POWER IN* connector pass straight through.

An *integrated* DCM receives its 12 V DC power from two pins in its single connector. See Chapter 9, page 127, for the positions of these pins.

2.3 Connecting to a single computer

In autonomous or temporary installations, you will only need to connect the DCM to one computer (*e.g.* a laptop) for initial configuration. Follow these instructions to make the DCM communicate with your computer.

Note that your computer must either have an RS232 (serial) port for initial connection, or be running its own DHCP server. DHCP servers are available for Windows, but are not supplied as standard.

If you want to connect the DCM to a local area network, follow the instructions in Section 2.4, page 15, instead.

Connecting over a serial link

The *DATA OUT* port can be used to connect a DCM to a single computer through a direct serial link. This link can be used to log in to the Linux operating system of the DCM and transfer files. It is most useful for maintenance and troubleshooting over a low-bandwidth connection, and for initial setup.

To communicate over a serial link:

1. Connect an RS232 reverse serial cable between the DCM and your computer. A suitable cable should have been supplied with the unit.

2. Run a terminal emulation program on the computer. The built-in programs `minicom` for Linux, and `hypertrm` for Microsoft

Windows, can be used for this.

3. Configure the baud rate of the serial link. By default, the DCM uses a baud rate of 115200, with 8 data bits, no parity bit, and one stop bit, and *without* flow control.

4. Log in with your username and password, or `root` if you have not yet created one. See http://www.faqs.org/docs/Linux-HOWTO/Text-Terminal-HOWTO.html for full information on how to set up terminal emulation under Linux.

5. You now have access to the DCM's console and all of its functions. You should change your password, if you have not done so already, with the command `passwd`

For full details on how you can configure the serial ports of the DCM, see Section 6.2, page 82.

Connecting over Ethernet

You can use a “crossover” Ethernet cable to connect the DCM directly to your computer. This has the same effect as connecting the DCM and your computer to a separate hub using standard Ethernet cables: the two machines will consitute a network segment, and each will have its own IP address on that segment. See Section 2.4, page 15, for details on how to configure TCP/IP on the DCM.

If your computer has two network interfaces, and one is connected to a local area network, you may be able to set up a network bridge between the DCM's segment and the rest of the network. For example, if you are using Microsoft Windows XP:

1. Connect to the DCM's console over a serial link.

2. Because you are connecting the DCM to a single computer, you will probably not be able to use DHCP. You should configure the DCM to disable DHCP and use an IP address in a private range such as `192.168.0.`*x*:

   ```
   gcfgdbset net.eth.0 static
   gcfgdbset net.eth.0.address 192.168.0.2
   gcfgdbset net.eth.0.netmask 255.255.255.0
   ```

   Here, `192.168.0.2` should be replaced with the IP address you want the DCM to use.

Allow a short time for the DCM to reconfigure the network.

3. On your computer, select **Start → Control Panel → Network Connections**. Right-click on the interface connected to the DCM, and select **Properties**.

4. Click on **Internet Protocol (TCP/IP)**, then **Properties**.

5. Select *Use the following IP address*, and fill in an address on the same subnet as the DCM (*e.g.* `192.168.0.49`). Click **OK**, then **OK** in the *Connection Properties* window.

6. Check that you can connect to the DCM using its new IP address by opening its Web site at

   `http://192.168.0.2/`

7. *Advanced usage:* If you later want to connect your computer to a local area network, you can make the DCM visible to that network by setting up a network bridge. *Before* doing this, you must make sure that the IP addresses you have chosen for the DCM and your computer are suitable for the local network you want to connect it to.

   To set up the bridge, open the *Network Connections* window, and select the connections corresponding to the DCM and your network. Click **Bridge connections**, and follow the instructions in the wizard.

If you are using Linux or another operating system, you should see its own documentation for more details.

Using an internal modem

If you have ordered a DCM with an internal PCMCIA modem, it will automatically listen for incoming PPP connections. This is most useful if the DCM is integrated into a borehole instrument, which can then be connected to a modem at a central data collection point. The modem within the DCM appears on the Web configuration interface as a serial port, allowing you to change the service it provides and the data transfer settings it uses. For example, if you do not want the internal modem to use PPP, you can configure it to use *mgetty* instead and log in directly.

Connecting over USB

If your DCM is equipped with the USB *client* interface, you can

connect it directly to any computer with a USB socket. When this is done, the computer treats the module as if it were a standard network card. For example, to connect the DCM to a PC running Microsoft Windows, you should:

1. Link the DCM's USB port to any available USB socket on the computer.

2. After a short wait, Windows should report that a new USB device has been detected. When prompted to install drivers for it, you should use the ones provided on CD by Güralp Systems.

Once the "network card" is installed, the two machines form a network segment, and each one has its own IP address on that segment. See the next section for details on how to configure TCP/IP on the DCM. If you wish, you can set up a network bridge between this connection and a local area network or the Internet as described above.

2.4 Connecting to a local network

The DCM is normally supplied with a 10BaseT Ethernet port. This port can be used to connect to your local network.

When the DCM starts up, it will try to find a DHCP server on your network to assign it an IP address.

If you use DHCP on your network, and you want to access the DCM's Web server or console over the network, you may be able to find out from the DHCP server which address it has given the DCM, in which case you can access it directly. Otherwise, you will need to connect to the DCM over a serial link to find out its IP address.

If you do not use DHCP on your network, you will have to set the DCM to use a static IP address.

1. Make a temporary serial connection to the DCM from a local PC. You should see the message

   ```
   DCM login:
   ```

2. Enter `root` and the administrator password. If you have not been given a password, the default setting is `rootme`. The DCM will reply with a prompt:

   ```
   [root@DCM ~]#
   ```

   This verifies that the unit is working properly. You should

change the password as soon as you can with the command `passwd`

3. Connect the DCM's *NETWORK* port to your network, and power cycle it. Log in again.

4. If your network *does not* use DCHP, type

```
gcfgdbset net.eth.0 static
```

to make the DCM use a static IP address instead, and set the address with

```
gcfgdbset net.eth.0.netmask 255.255.255.0
gcfgdbset net.eth.0.address your-address
```

Here, *your-address* should be replaced with the IP address you want the DCM to use. The IP address you choose must be unique on your network.

If you are connecting the DCM to a machine on the same network, you do not need to configure any more options at this stage.

If your network uses DHCP, you will not need to perform this step. Instead, ensure the DCM is also using DHCP with

```
gcfgdbset net.eth.0 dhcp/bootp
```

5. Issue the command `ifconfig`

The DCM will reply with technical information on its current network setup. Each interface is listed separately. The *NETWORK* port uses the interface `eth0`:

```
eth0      Link encap:Ethernet  HWaddr 08:00:20:C0:FF:E2
          inet addr:192.168.48.187  Bcast:192.168.48.255
Mask:255.255.255.0
          UP BROADCAST NOTRAILERS RUNNING MULTICAST
MTU:1500  Metric:1
          RX packets:88045 errors:36 dropped:0 overruns:36
frame:0
          TX packets:16308 errors:0 dropped:0 overruns:0
carrier:0
          collisions:149 txqueuelen:100
          RX bytes:12358214 (11.7 MiB)  TX bytes:0 (0.0 B)
          Interrupt:76
```

If the interface is correctly set up, its IP address will be shown after `inet addr:` (here, 192.168.48.187.)

6. If you are connecting the DCM to a single computer which does not run a DHCP server, you will need to configure that computer to use an IP address on the same subnet as the DCM before they will be able to reach each other (see “Connecting over Ethernet“ on page 13.)

You should now be able to connect to the DCM's Web setup interface by typing its IP address into any browser, *e.g.*

https://192.168.0.2/

Scream! and similar software applications should also be configured to use this address to communicate with the module, if they are intended to communicate over the network.

Alternatively, you can continue using the `gcfgdbset` command to set other configuration options by name. In Chapter 6, each option's name is given in *italics*. The command syntax to use is

`gcfgdbset` *option-name new-value*

The options will take effect immediately whenever possible. Some changes may take some time to complete, since services may need to be restarted. The `gcfgdbset` command performs only simple checks on the new value, so you should check the syntax of the option carefully.

Wi-Fi

Optionally, a DCM may be supplied with 802.11 (“Wi-Fi”) hardware in place of Ethernet. There are two modes in which an 802.11 network may operate:

- In *ad hoc* mode, data packets are sent out over the wireless connection indiscriminately and are received by all machines on the network. This is the simplest style of Wi-Fi network, but consumes more power and cannot easily be extended to large networks.

- In *infrastructure* mode, dedicated 802.11 hubs collect data packets, collate, and distribute them efficiently around the network. These hubs may also act as transparent extensions to existing (wired) TCP/IP networks.

The DCM is capable of running in either situation, depending on your requirements. As far as the computers on the network are concerned, there is no difference between a TCP/IP network running over Ethernet

and one using 802.11 connections, or a mixture of the two.

Once the 802.11 system is running, you will need to run a DHCP service on your network, or manually set the IP address of the DCM as above.

Connecting to the Internet

If the local network is already connected to the Internet through some other means (perhaps through a gateway machine), you can create a default route with

```
gcfgdbset net.eth.0.gateway network-gateway
```

where *network-gateway* should be replaced with the IP address of the gateway on your local network.

The DCM can perform more complex routeing if required. See “Static routes”, page 80, for more details.

As supplied, the DCM module will accept requests to its on-board Web server and logins over SSH. In addition, the DCM can be requested by Scream! or other GCF-compatible software to send GCF streams to your computer.

If the instrument is located on a private network, you may be able to connect to it from the wider Internet by using a feature implemented by SSH known as *tunnelling.* You can use this technique to connect to the DCM through a chain of intermediate machines which support SSH. Once this chain is set up, you can treat the connection as if it were a direct link between the DCM and your computer. Many standard Internet protocols may be fed through SSH in this manner. Whether this is possible will depend on the precise configuration of your local area network. For more details, please see the documentation for SSH clients such as `ssh` and `putty`.

2.5 Connecting digitizers and external hardware

The DCM is designed for use with Güralp Systems digitizers, which communicate over RS232 or RS422 serial links using the GCF protocol. There are three RS232 ports available on a stand-alone DCM and ten on a rack-mounted model, whilst on a DCM bonded to a digitizer unit only the *DATA OUT* port is available (the digitizer is connected directly to the *PORT A* interface).

The three serial ports on a stand-alone DCM are labelled *DATA OUT, PORT A*, and *PORT B.* In a typical setup, one or both of the latter two

ports are connected to Güralp DM-24 digitizers, whilst the *DATA OUT* port connects the DCM to a computer or serial modem. However, the labels are provided merely for convenience: internally all three serial ports behave identically. Thus, if the DCM is connected to a network over Ethernet, you could use all three ports to communicate with digitizers. Conversely, if you have only one digitizer attached to a stand-alone DCM, either or both of the remaining two ports could be used to transmit data gathered by the module.

A rack-mounted DCM has a *CONSOLE* port with a female connector, corresponding to the *Data out* port of the stand-alone model, and nine serial ports (with male connectors) for communication with digitizers.

If the DCM has a USB *host* interface, you can connect it to any USB peripheral supported by its Linux operating system. For example, you may want to attach a camera to the DCM, or an additional USB mass storage device. If there is no hard disk inside the module, it will automatically search for suitable storage on the USB interface. The DCM may alternatively have been supplied as a USB *client*; if this is the case, the USB connection can be used to connect the module to a single computer (see below).

2.6 Setting up the DCM

Once a DCM is installed, it must be configured to your particular requirements. The easiest way to set up a DCM is through its on-board Web server, which gives you access to all the module's configuration options. You can use this Web interface over any network connection using the HTTP or HTTPS protocols. Alternatively, you can log in to the DCM's Linux operating system and issue commands directly: see Section 8.2, page 117.

Web setup

The DCM provides a Web interface which you can use to set up the system. To access this from anywhere on the local network, open any web browser and navigate to the IP address of the DCM. For example, if the DCM has the address `10.82.0.1` you would enter

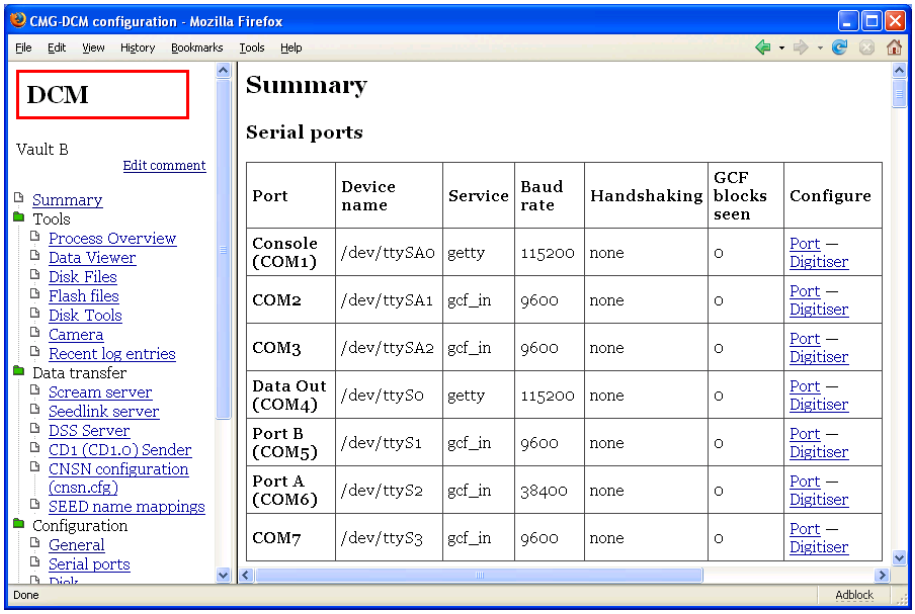
`http://10.82.0.1/` or `https://10.82.0.1/`

https is a secure variant of the *http* protocol, which ensures that network traffic cannot be read in transit. The DCM as supplied accepts connections only over *https*, although this may be altered if required (see “HTTP server”, page 79.)

Your browser may ask you for a username and password to access the

Web configuration interface. An initial *root* username and password will have been supplied by Güralp Systems; however, you should check with the DCM's administrator what username you should use. On a clean install of the Linux distribution, the password for `root` is `rootme`.

Once you have logged in, you will see a Web page similar to this:



The work area on the right displays a summary table describing the current setup of the DCM's serial ports, and showing how many blocks of data have been received on each one.

Under the *Configure* heading are two links for each serial port. **Port** provide quick access to the DCM's configuration settings for the port, and **Digitizer** allows you to configure a Güralp Systems digitizer attached to the port REFERENCE???

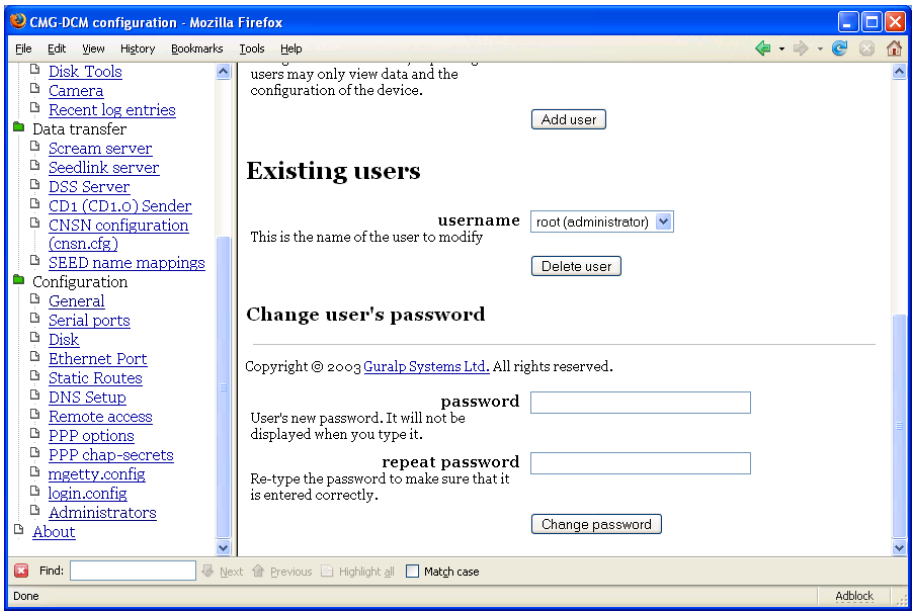
Scrolling down in this window, you will see

- a list of all the files currently stored in the on-board Flash memory, with a link to download each one;
- a summary of the current status of the DCM's removeable hard disk;
- the status of the tamper-proofing subsystem (used mainly on integrated DCMs and AMs; standard DCMs do not have the tamper lines exposed, so you can ignore this section);

- the current Linux network and DNS configuration;
- unique IDs for the hardware boards making up the DCM; and
- the name and version number of every software package currently installed on the unit.

The links in the menu on the left lead to various pages where you can change the configuration of the DCM.

As a first step, you change your administrator password. Scroll down the left-hand menu if necessary, and click on **Configuration – Administrators.**



At the bottom of this page is a section headed *Existing users.* In this section, enter your new password in the *password* and *repeat password* boxes, and click **Change password**.

If you want to add any other named accounts, you can do this on the same page (see “”, page .)

Next, you should configure the DCM and attached digitizers for your own needs. Several pages of configuration options are available under **Configuration** in the left-hand menu. When you make changes to any page, make sure you click the **Save changes** button at the bottom of the page before you move to a new page.

Examples of how to configure a DCM for several common applications are given in the next chapter, whilst a full reference is provided in

Chapter 6. Some important points to note are:

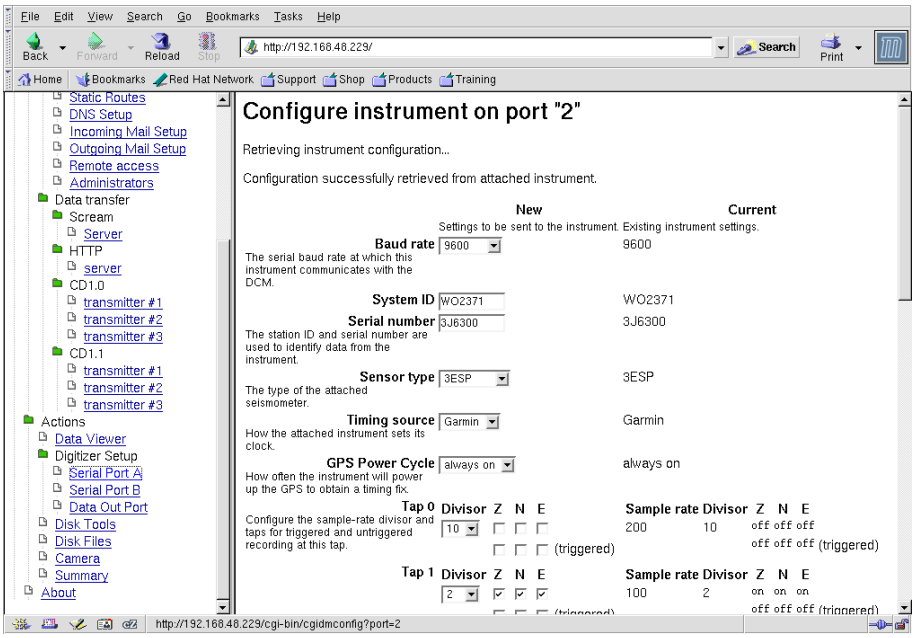
- The DCM uses internal serial and Ethernet ports to communicate with the digitizer and the network. Altering the serial or network settings may make the instrument unreachable over the network. You should review carefully any changes you make to these settings before committing them.

- It is possible to disable the DCM's *ssh*, *http*, and *https* servers using the links under *Data transfer*. Before doing this, you should ensure that you will still have access to the DCM by other means, *e.g.* by logging in directly over *ssh*.

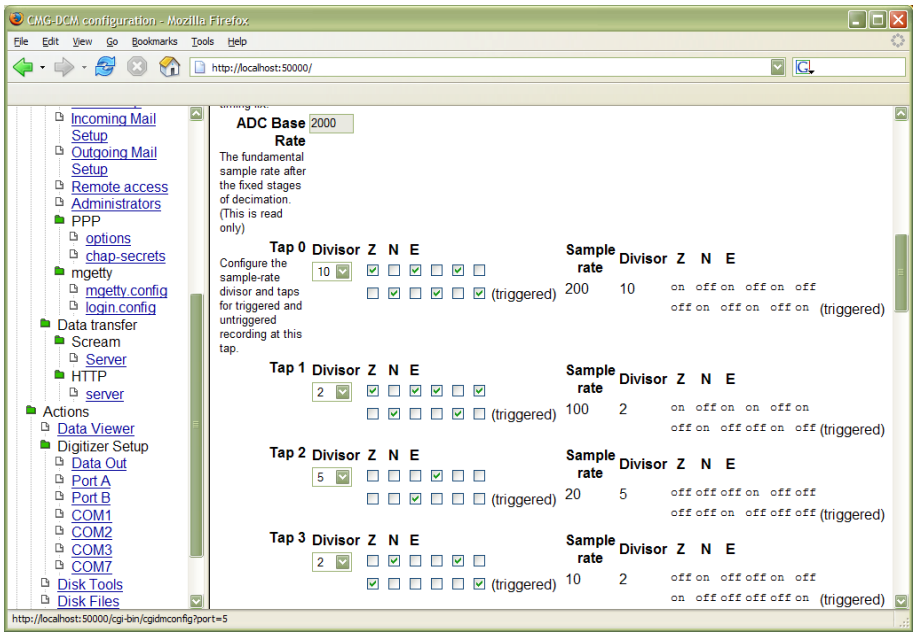
Configuring digitizers with the DCM

You can change the configuration of attached DM24 digitizers through the DCM's web page interface. To do this, click on one of the **Digitizer** links beneath *Configure* in the serial port table.

The DCM will then retrieve the current configuration from the DM24, which will take a few seconds. This done, a page will appear allowing you to alter the digitizer's settings:



You can change all of the digitizer's configuration options from this page. If you have attached a 6-channel digitizer, the page will reflect this:



When you are done, click **Configure instrument** to submit the changes to the digitizer and reboot it.

See “Digitizer output control”, page 107, for full details of what you can do.

Configuring digitizers with Scream!

If you prefer, you can use Güralp Systems' Scream! software to configure and calibrate digitizers and sensors through the DCM module. You can do this either over the network, or using a serial port.

To connect over the network, start the DCM's Scream! server (see ??? REFERNECE) and connect to the DCM using Scream!'s *Network Control* window. The digitizer should appear under an entry for the DCM in Scream!'s main window.

To connect over a serial link, set the service for the serial port to *gcf_out* (see ???REFERENCE), and connect the serial port to your computer. The digitizer should appear in Scream!'s main window. The DCM may not appear in this case, because it does not send its own GCF blocks.

For full details on how to use Scream!, please refer to its own documentation or the extensive on-line help.

3 Usage

The DCM can be integrated with any system where seismic data needs to be collected, or converted from one form into another. It is designed to operate as transparently as possible, and once connected and configured for a particular role in a system it should not require further maintenance.

The rest of this chapter gives detailed installation and usage notes for several common DCM installations. Between them, they highlight many important features of the DCM. For full configuration information, please refer to the next chapter.

3.1 General notes

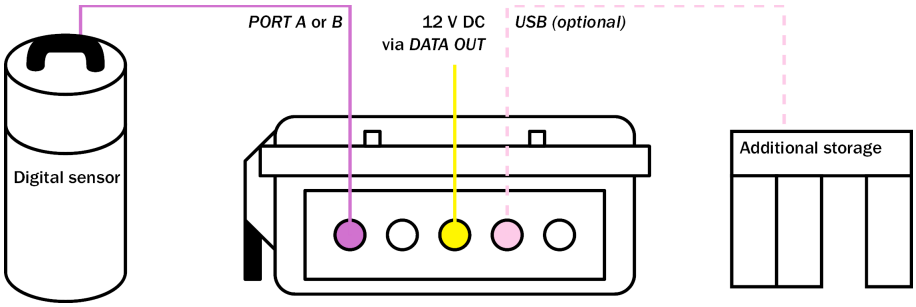
Stand-alone DCM modules can be supplied with a touch-screen and minibrowser as an option. This allows you to perform configuration tasks on-site. However, the touch-screen imposes some environmental restrictions on the unit.

For all other DCM units, you will need console or network access to the DCM to configure it for your installation. This is especially important if the DCM is not to be part of a TCP/IP network whilst in use.

As supplied, the *DATA OUT* port runs the *getty* service, which you can use to access the console of the DCM over a serial link. Alternatively, you can temporarily connect a computer to the *NETWORK* port. It is recommended that you keep a serial or modem connection to the DCM available for maintenance and troubleshooting, even if the link is too slow for general use.

3.2 The DCM as a data store

The simplest way to use a DCM is as a storage medium for digital data.



1. Connect the DCM to a digitizer or digital sensor.

2. Connect the *DATA OUT* port to a source of 12 V DC power.

3. Connect the unit to your local network as described in Section 2.4, page 15, and perform any necessary configuration of the DCM and digitizer. You will be prompted for a username and password: log in as `root` and use the password supplied by Güralp Systems. If you have not been given a password, use the default password `rootme`. You should change this password as soon as you can.

4. The DCM will immediately begin recording data into Flash memory as it is received, and every so often data will be moved onto the internal hard disk. At this point you can leave the DCM running without assistance.

   Data is stored in the Flash memory as a number of files in Güralp Compressed Format (GCF) unless you have specified another option. Once the memory becomes 75% full, files are automatically moved to the hard disk until it is less than 50% full. (See Section 8.1, page 116, for details of the algorithm used.)

   You can check that the DCM is receiving data either by monitoring the *Summary* page of the on-board Web interface (see Section 4.1, page) or from a command prompt using the command

   ```
   gnblocks
   ```

   This command displays details about all of the DCM's serial ports, and the number of GCF blocks received so far on each one. To see information about a single port, type

   ```
   gnblocks port-number
   ```

5. If the internal hard disk is missing or becomes full, and you have chosen the USB *host* option, the DCM will automatically look on the USB interface for an external USB mass storage device.

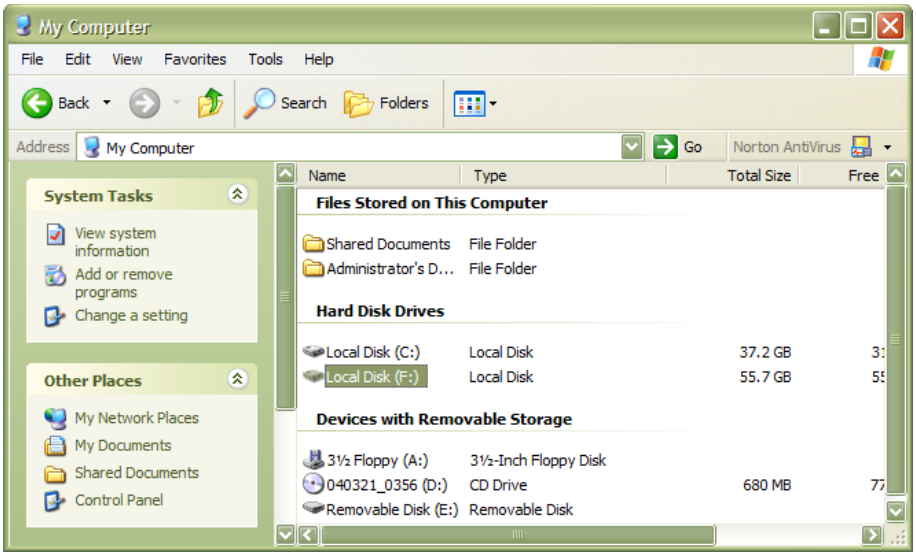
6. To replace the internal hard disk, unclip the cover of the DCM and hold down the lever button to bring the disk out from its housing.



Slide the disk out and replace with another Lacie U&I drive, or any brand of IDE / USB or IEEE 1394 2.5" drive you specify (at the time of manufacture). You can do this at any time without losing data.

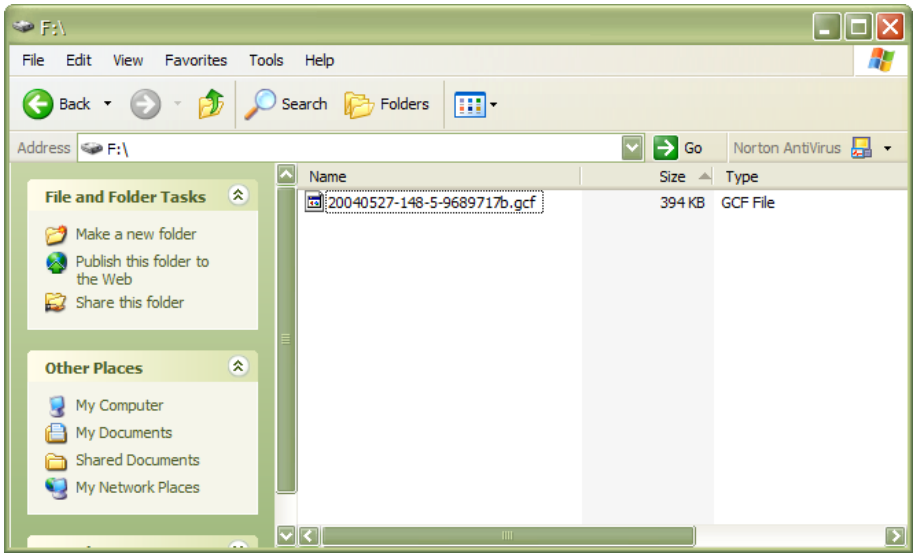
7. Plug the hard disk into any computer that supports the USB Mass Storage standard using a standard USB cable. Newer Linux distributions and Microsoft Windows XP have this enabled by default.

   In Windows XP, you should see a series of **Found new hardware** messages indicating that the drive has been recognised. A new disk drive icon should appear in *My Computer*:



This process may take several minutes to complete.

8. Double click on the drive's entry to browse the files inside and copy them to your data store.

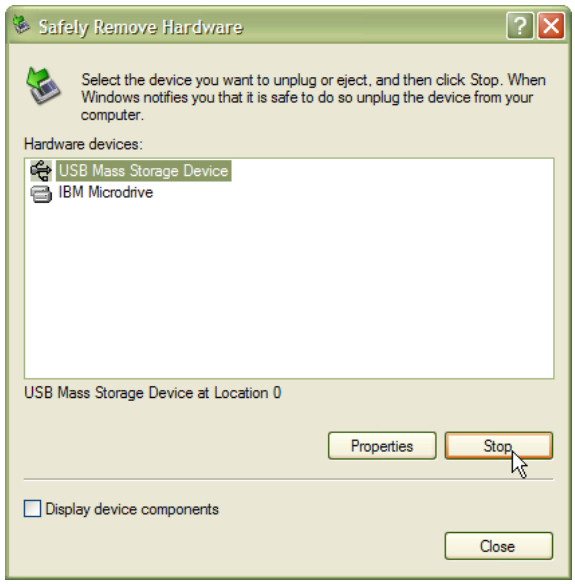


Alternatively, you can open the files directly from the USB disk using Scream! or other GCF-compatible software.

9. When you want to remove the USB disk, double-click on the **Safely Remove Hardware** button:



10. Choose the USB port attached to the disk, and click **Stop**.

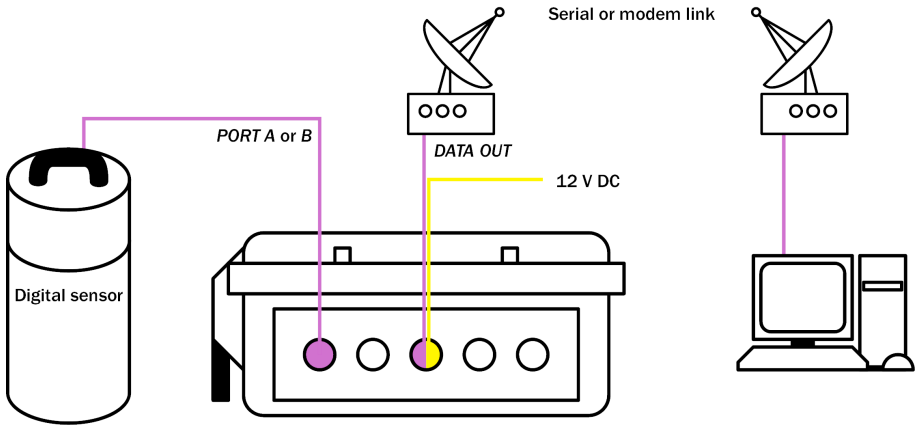


11. In the next window, check that the correct hardware is shown, and click **OK** to confirm.



12. You can now remove the hard disk from the computer and reinstall it in the DCM if required.

3.3 The DCM as a GCF data source



With a sufficiently fast serial link, you can instruct the DCM to send incoming data directly to a GCF-compatible client. For example, Güralp Systems' Scream! software allows you to display and record incoming data, as well as change the settings of attached digitizers. You can do this in addition to recording data on a local hard disk, or you can leave the hard disk uninstalled and operate the DCM entirely over the network link.

This example shows a module communicating with a local PC over a dedicated radio link. You could also use a simple serial cable to connect the DCM to the PC.

1. Connect the DCM to a digitizer or digital sensor.

2. Connect the *DATA OUT* port to a source of 12 V DC power.

3. Either use a computer connected to the *NETWORK* port to browse to the DCM's Web site, or log in to the Linux console over the serial link.

4. Find out the port number of the DCM's *DATA OUT* port. On the Web interface, the *DATA OUT* port is listed by name in the serial port table.

   If you are using the Linux console, use the command `gnblocks` to display the port number.

5. On the Web interface, click the **Port** configuration link for the *DATA OUT* port, and set the **serial.x.service** option to *gcf_out*.

   If using the Linux console, use the command `gcfgdbset serial.x.service gcf_out` (see Section 8.2, page 117.)

   Note: Beyond this point you will *not* be able to use the serial link to access the Web configuration interface of the DCM. If you do want to be able to do this, you should configure the serial link for PPP (see “PPP”, page 100) and run a Scream! *server* on the DCM. You will need to assign the DCM its own IP address on your local network.

6. If you connected to the DCM by PPP, you will lose the network connection at this point, because the DCM is now using the serial link directly.

7. Open Scream!'s main window, and look under *Local* in the tree on the left for the serial port which is communicating with the

DCM. If it is not shown, you may have to 'unfold' the tree to reveal it. The digitizer(s) attached to the DCM will appear underneath the entry for the serial port, and you can configure them from within Scream!. See Scream!'s documentation for more details.

8. If no digitizer is shown, it may be that your software is not configured correctly for the serial link. As supplied, the DCM uses a baud rate of 115200 on its *CONSOLE* port, with 8 data bits, no parity bits and 1 stop bit, and no flow control.

For more information on how to use Scream! to display and manage data streams, please see Scream!'s own documentation or the extensive on-line help.

Accessing the DCM command line through gcf_out

The *gcf_out* service provides a command-line terminal designed to be compatible with Güralp Systems' SAM serial storage device. To access this terminal from Scream!, right-click on the DCM's icon and select *Terminal...*. You will see a banner like

```
DCM DCM CRSM Command Mode
0 blocks in buffer | 16360 blocks free
```

The SAM commands you can use are described in the SAM Operator's Manual.

You can also log in to the DCM's Linux operating system through the terminal. Enter the command

```
GETTY
```

and press `ENTER`. You will be presented with the `login:` prompt.

If you are not using Scream!, and cannot make the unit enter SAM compatibility mode, you can interrupt the GCF data stream and drop straight to the `login:` prompt by connecting direct to the serial port and typing

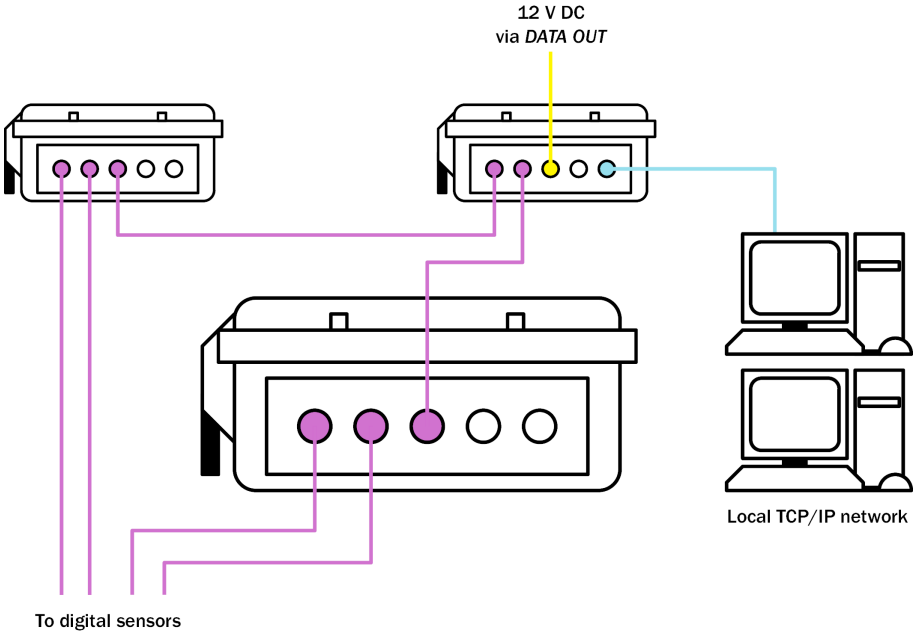
```
forcegetty
```

It may take several attempts for this to succeed.

Sensor arrays

Another possible reason for setting the *DATA OUT* port's service to *gcf_out* is to allow several DCM units to work together and aggregate

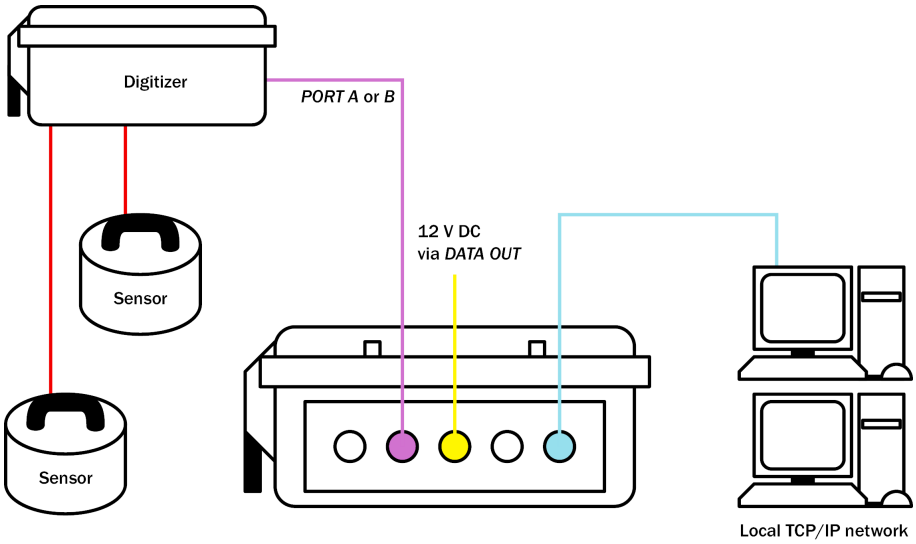
the inputs from an array of sensors.



Here, *DATA OUT* ports with *gcf_out* as the service output data to the serial inputs of another DCM or AM module, which aggregates them all and sends them out over a network connection.

The DCM can handle multiple data sources with ease using the *gcf_out* service. In practice, however, it is almost always preferable to set up a local area network and use PPP over any serial links, so that GCF, Web and SSH traffic can share the same connection. Setting up a fully networked array is slightly more difficult than using *gcf_in* and *gcf_out* services, but provides much more flexibility.

3.4 The DCM as a network data hub



The most flexible way to operate the DCM is as a fully independent machine on your local area network. To do this:

1. Connect Güralp digitizers to *PORT A* and *PORT B* as necessary, using the serial data cable provided.

2. If you are using the DCM's USB capabilities, connect your external hardware or computer to the *USB* socket.

   Note: In the USB standard, a device is either a *host* (a computer) or a *client* (a peripheral.) The DCM's USB port can act as either, depending on the options specified at manufacture. It cannot do both at the same time.

   If your DCM is a *host*, you can attach additional USB peripherals such as hard disks to the *USB* socket. If it is a *client*, you can attach the *USB* socket to a computer, and it will appear as a network interface.

3. Connect a Güralp combined serial/power cable to the *DATA OUT* port. Make up a connector if necessary, and attach the power lines to a 12 V DC power supply.

4. Connect the 9-pin serial socket to a computer for configuration.

5. Connect an Ethernet cable to the *NETWORK* socket, and set up the network as described in Section 2.4, page 15.

6. You should now be able to connect to the DCM's Web setup

interface by typing its IP address into any browser, *e.g.*

https://192.168.0.2/

By default you can use either *http:* or *https:* URLs to access the DCM's Web site. HTTPS is a secure variant of HTTP, which we recommend you use in preference to avoid passwords being sent over the network in clear text. Scream! and similar software applications should also be configured to use this IP address to communicate with the module. If the DCM is using a static IP address, you may assign it a name on your network (although you will still need to use the numeric IP address in Scream!.)

The DCM also runs a SSH (secure shell) server, which you can use to access its Linux command line over the network, just as if you had connected to it over a direct serial link. The standard Linux program `ssh` and the freeware Windows program `putty` are popular SSH clients.

If you do not wish to use the DCM's Web site, you can continue using the `gcfgdbset` command to set other configuration options by name. For the option names, you should refer to Chapter 4, "Configuration options" in the DCM manual, where each name is given in *italics* with a description of its action.

The command syntax to use is

`gcfgdbset` *option-name new-value*

The options will take effect immediately whenever possible. Some changes may take some time to complete, since services may need to be restarted. The `gcfgdbset` command performs only simple checks on the new value, so you should check the syntax of the option carefully.

Communicating with digitizers

You can configure attached digitizers from the DCM Web interface by clicking on the *Configure* – **Digitizer** link in the serial port table (see Section 2.6, page 22.)

You can also connect to the console of attached digitizers with the terminal command

`minicom -n` *port-number*

Alternatively, you can send individual commands to a digitizer with

the command

```
gcli port-number command
```

In these commands, *port-number* is the port number and *port-name* the port device name to send the command to. To find out these, issue the command `serialmap`:

```
Library version: libserialmap Version 1.0.5 with LIBGCONFIGDB
3 serial ports
 Port  0, Key   7000, name Data out port, device /dev/ttySA0, baud
115200
 Port  1, Key   7001, name Port B, device /dev/ttySA1, baud 9600
 Port  2, Key   7002, name Port A, device /dev/ttySA2, baud 19200
```

Each port will be listed, with its number (*e.g.* `0`), name (*e.g.* `Data out port`), and device (*e.g.* `/dev/ttySA0`).

Data storage and retrieval

All data received by the DCM is periodically written from Flash memory to the on-board hard disk. The time period used can be configured from the **Configuration – Disk** page of the DCM's Web site.

You can list the files on the hard disk

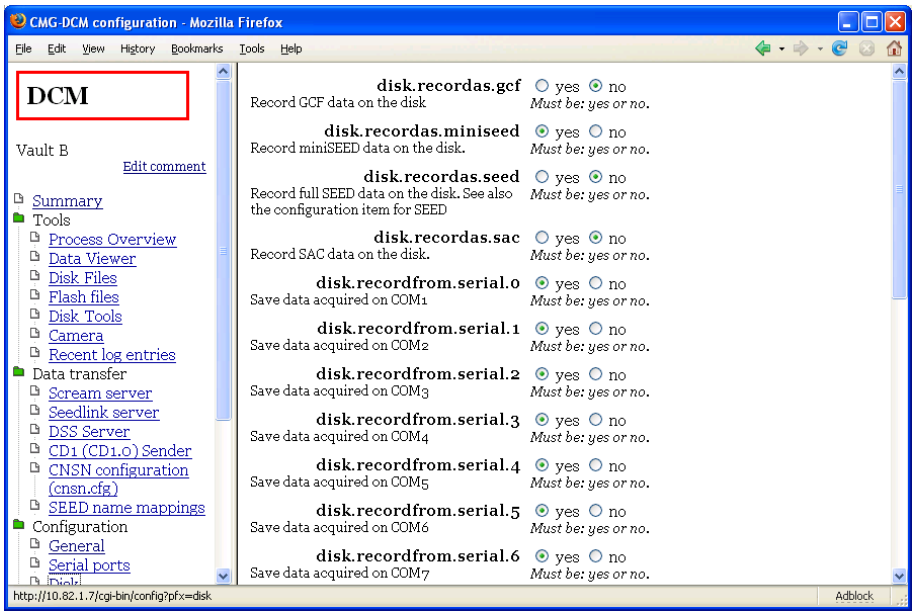
- over the Web, by selecting **Tools – Disk files** from the menu, or
- from the command line with `gfat32 ls`

If you are using the Web page interface, each file appears as a link: click its name to download the file.

Alternatively, you can physically remove the internal hard disk, and attach it to any computer supporting either Firewire or the USB Mass Storage Protocol. It will appear on the computer as an extra hard drive. The DCM uses a journalling file system compatible with Windows' FAT32, so you cannot lose or corrupt data by removing the hard disk, even if the DCM is in the middle of writing to it.

Using miniSEED format

You can instruct the DCM to record data to Flash (and, later, its hard disk) in miniSEED format by browsing to the **Configuration – Disk** page and selecting *yes* for the *disk.recordas.miniseed* option.



The DCM allows you to record in several formats simultaneously, if you wish; you should make sure that you have the capacity to store, or the bandwidth to transfer, all the data the DCM will produce in each format you have enabled.

Using SAC format

You can instruct the DCM to record data to Flash (and, later, its hard disk) in SAC format by setting *disk.recordas.sac* to *yes*.

Compiling SEED volumes

The DCM can also compile full SEED volumes in real time. To enable this:

1. Edit the `/etc/seed.cfg` file. The DCM will only write SEED volumes for streams mentioned in this file. See “SEED recorder”, page 91, for more details.

2. Browse to the **Configuration – Disk** page, and set *disk.recordas.seed* to *yes*, or issue the command

   ```
   gcfgdbset disk.recordas.seed yes
   ```

3. Click **Save changes** to save the configuration and start writing SEED volumes. One volume will be written for each stream every three hours. The DCM will always start a new volume if it is power cycled; the aborted file will still be a valid SEED volume.

Real-time data transmission

You can use the *DATA OUT* port (or any of the DCM's serial ports) in several ways, depending on the *service* you have chosen to run on the port.

- By default, the *DATA OUT* port runs the *getty* service, which allows you to connect to the DCM's console.
- The *mgetty* and *mgetty_raw* services are similar, but can deal with connections made through modems.
- The *gcf_out* service causes the DCM to combine the incoming GCF streams and send them on transparently, as for a Güralp SAM or CRM.
- The *gcf_in* service turns the port into an extra digital data input port. The DCM's *PORT A* and *PORT B* run the *gcf_in* service.

To change the service running on a port, click the *Configure* – **Port** link in the serial port table, and set the *serial.x.service* option to the required value. Alternatively, issue the command

```
gcfgdbset serial.x.service service
```

Scream! server

In a straightforward vault installation, we recommend that you leave the *DATA OUT* port running the *getty* service for emergency console access, and instruct the DCM to act as a server for incoming data on your network. This can be done from the **Data transfer – Scream server** page on the Web site:

1. Enable the Scream! server on the DCM by setting *datatransfer.scream.server* to *on* and *datatransfer.scream.server.allowserialaccess* to *yes.* Make a note of the port number *datatransfer.scream.server.port*, or change it to another. Click **Save settings**.
2. Start up Scream!, and select **Windows – Network Control...** from the main menu. Switch to the *My Client* tab.
3. If it is not already selected, check **Receive Data** to start Scream! listening.
4. Right-click anywhere in the **Servers** list box, and select **Add UDP Server...**. Enter the IP address and port number of the

DCM, separated by a colon **:** (*e.g.* 192.168.0.2:12345)

5. Test communications by right-clicking on the newly-added server, and selecting **GCFPING**. A message appears in the *Control* pane logging the ping being sent. If communication is good, and the server is enabled for client requests, you will receive a *GCFACKN* message from the server which will also appear in the *Control* pane.

6. Request data by right-clicking on the server and selecting **GCFSEND:L** (or **GCFSEND:B**) from the pop-up menu.(**L** is used for little-endian and **B** for big-endian byte order, and are distinguished for compatibility.) Streams should soon begin to appear in Scream!'s main window.

7. To stop the link, right-click as before and select **GCFSTOP** from the pop-up menu. If you do not **GCFSTOP**, the server will continue to transmit to a client that is no longer listening. You should ensure that the server replies with a *GCFACKN* message If an acknowledgement does not appear in the *Control* pane, repeat the **GCFSTOP** command.

CD1.0 and CD1.1

The DCM can be configured to transmit incoming data to a specified client using the CD1.0 or CD1.1 protocols. If you intend to send CD1.0 or CD1.1 data to a National Data Centre (NDC), the installation will need to be assigned a unique code by the International Seismic Centre or with the US's National Earthquake Information Center.

To enable CD1.0 or CD1.1 transmitters, browse to the **Data transfer – CD1 (CD1.0) Sender** or **Data transfer – CD1.1 Sender** page. Fill in the required data fields and click *Save settings*.

Using an Authentication Module (AM), you can cryptographically sign all outgoing CD1.0 or CD1.1 subframes as they pass through. This is particularly powerful where the module is physically part of the sensor casing, as it allows you to generate authenticated data at source, even within a borehole.

AutoDRM

AutoDRM (Automatic Data Request Manager) is an optional service which allows you to request data over e-mail. In response to a request, the AutoDRM can either send an e-mail in return, or establish an FTP connection to you.

Not all DCMs have these packages installed. If you are unsure, or wish

to upgrade, contact Güralp Systems.

In order for the AutoDRM system to work, the module must be able to send and receive e-mail messages. To set the system up, you will need to fill in the details of your network's SMTP server under **Configuration – Outgoing mail**, and of your POP or IMAP server under **Configuration – Incoming mail**. Remember to click **Save changes** after you have filled in each screen.

Switch to the **Data transfer – AutoDRM** page, and enable *datatransfer.autodrm*. Click **Save changes** on this page to start the service.

To test the AutoDRM system, send the following e-mail to the DCM:

```
From: your@e-mail.address
To: autodrm@your.DCM

BEGIN GSE2.0
MSG_TYPE request
MSG_ID unique-identifier
HELP
EMAIL your@e-mail.address
STOP
```

where *your@e-mail.address* is the e-mail address to send the return message and *unique-identifier* is a string which you can use to identify your request. `autodrm@`*your.DCM* should be replaced with an e-mail address which will be delivered to the `autodrm` mailbox on the DCM.

The DCM should send a mail back to you containing help on the commands you can give the AutoDRM system.

3.5 Troubleshooting DCM installations

I cannot connect to the DCM's DATA OUT port using a terminal program.

Press `ENTER` a few times to initiate communication.

If the *DATA OUT* port gives a `login` prompt, you should log in with your username and password. If you have not been given a different username and password, try logging in as `root` with the password `rootme`. You should change this password as soon as you can with the command `passwd`

If the *DATA OUT* port gives an `ok` prompt, the DCM is running the

gcf_out service on that port and has provided you with a FORTH compatibility interface. Type

```
GETTY
```

and press `ENTER` to gain access to the login prompt.

If the *DATA OUT* port does not respond when you press `ENTER`, or produces garbage, check that your terminal program is using the same baud rate as the DCM. By default, the DCM uses a baud rate of 115200, with 8 data bits, no parity bit and 1 stop bit, and no flow control.

If the *DATA OUT* port produces a large quantity of characters, it may be sending GCF data. Try typing

```
forcegetty
```

repeatedly. If the DCM is running the *gcf_out* service on this port, it will stop transmitting and give you access to the login prompt.

If the *DATA OUT* port is not responding at all, check the power supply to the DCM.

I cannot see the DCM's Web site over HTTP or HTTPS.

There is a problem with the network setup. Connect to the DCM's *DATA OUT* port using a terminal program and change the network settings to suit your network. In particular:

- If the DCM has a static IP address, use `ifconfig` to verify that the DCM is using the correct IP address. If it is not, change it with

  ```
  gcfgdbset net.eth.0 static
  gcfgdbset net.eth.0.netmask 255.255.255.0
  gcfgdbset net.eth.0.address 192.168.0.2
  ```

  (replacing `192.168.0.2` with the IP address you want the DCM to use.)

- If the DCM is using DHCP, use `ifconfig` to verify that the DCM is using the IP address you expected. If it is not, change the settings of your DHCP server or connect to the correct IP address for the DCM

- Check that your local PC can route to the DCM's IP address. For example, if you are using a cross-over Ethernet cable, the two hosts must share a subnet.

- Check that the DCM's HTTP(S) server is enabled by issuing one of

```
gcfgdbset net.remoteaccess.allow.http yes
gcfgdbset net.remoteaccess.allow.https yes
```

  as necessary.

I cannot connect to the DCM's Scream! server.

Make sure you are using UDP to connect to the DCM. Scream! servers between computers (including the DCM) always use UDP. The **Add TCP server...** option in Scream! is intended for hardware serial-to-IP converters only.

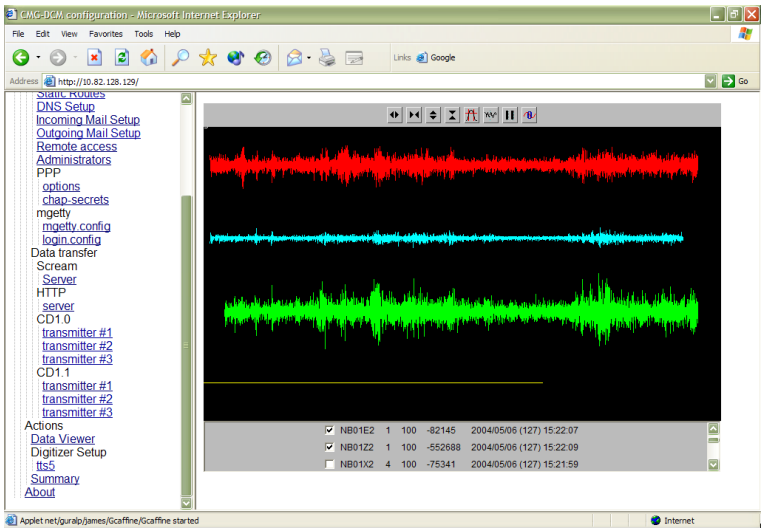
I cannot GCFPING the DCM's Scream! server.

- Check the *Receive UDP data* checkbox in Scream!'s *Network Control* window.

- Open the DCM's Web site, and check that the option *datatransfer.scream.server* is set to *on*, and that *datatransfer.scream.server.port* is the port you expected.

I can GCFPING the DCM's Scream! server, but no data appears.

Check the DCM is receiving data by selecting **Actions → Data Viewer** on its Web site. This is a Java applet which provides some of the functionality of Scream!, allowing you to check that data is being received correctly.



The streams being received at the DCM are listed in the bottom section of the applet. Click on a checkbox to add that stream to the main

viewer window.

If you do not want to use the DCM's Web site, you can find out the number of GCF blocks the DCM has received with the command `gnblocks`. Each serial port will be listed, with its name, number, key number and device name (as for `serialmap`) but including the number of blocks received on that port:

```
Key 0x007005: Blocks       3287 (Port  5, name Port A (COM6),
                                 device /dev/ttyS2, baud 9600)
```

The DCM is not receiving any data streams.

- Check the connection between the digitizer and the DCM by trying to log in to the digitizer's console using any of the methods described in Section 3.4 (page 33.) Press `ENTER` a few times to initiate communication.

- If the digitizer gives an `ok` prompt when you press `ENTER`, check that you have configured the digitizer to output real time data streams. Streams will not appear until a whole GCF block (1024 bytes) is ready for transmission, so a 5 sps stream may not appear until the digitizer has been working for 4 minutes. In addition, you can configure a digitizer to output only triggered streams, in which case it will not appear until a trigger occurs. (There is an exception to this: if you have put the digitizer in the *FILING* or *DUAL* filing mode, it will send heartbeat messages to Scream! clients every so often. The DCM will not show these messages in the *Data Viewer.*)

- If the digitizer does not respond when you press `ENTER`, or produces garbage, check that the DCM is using the same baud rate as the digitizer. By default, digitizers use a baud rate of 9600, with 8 data bits, no parity bit and 1 stop bit, and no flow control. To change the DCM's settings, *exit the terminal program,* and either

- access the DCM Web site, click on the *Configure* – **Port** link in the serial port table, and change the settings, or

- issue configuration commands such as

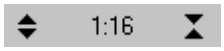
  ```
  serial.5.service gcf_in
  serial.5.baudrate 9600
  serial.5.handshaking none
  ```

  To obtain the port number (here 5) corresponding to a named port, use the command `gnblocks`.

- If the digitizer is not responding at all, check the power supply to the digitizer and the cable between it and the DCM.

The DCM is receiving streams, but they do not contain any data.

If the data you see is zero:

- Increase the Y axis scale using the 

icon in Scream!'s *Waveview window* or the corresponding icon in the DCM's *Data Viewer*.

- Check the mass position outputs to see if the masses are locked. A properly unlocked and centred mass should show a mass position within 1000 counts of zero; a locked mass will give a value greater than 32,000 (or less than −32,000) counts. If your sensor has remote mass locking, you can unlock the masses by navigating to **Actions ➔ Digitizer setup** and clicking *Unlock sensor*.

  Locking or unlocking the sensor mass typically takes several minutes to complete.

- Check the connection between the sensor and digitizer and try again.

If you cannot see the data in the stream, remove any constant offsets by clicking on the **Zero streams** icon  in Scream!'s *Waveview window* or the DCM's *Data Viewer*.

The DCM receives streams, but gaps appear in the data some minutes after boot-up.

Check that the baud rate between the digitizer and the DCM is sufficient for all the data streams you want to transmit. If it is not, the digitizer's output buffer will gradually fill up until no more data can be stored. Increase the baud rate of the digitizer through the DCM (or using Scream!), *then* set the baud rate of the DCM's input port to the same value.

If you are using triggered output streams, be especially careful to allow a high enough baud rate to transmit data from all possible output streams simultaneously, or you will observe gaps when an event triggers the digitizer.

The DCM receives streams, but 2-minute gaps appear in the data at 4-hour intervals.

After a reboot, the DCM takes around 2 minutes to begin transmitting. The DCM runs a `guardian` process which monitors the health of the system. In some circumstances `guardian` will need to reboot the DCM to attempt to resolve a problem. If the reboot does not help, the DCM will soon find itself in the same position, and `guardian` will reboot it again.

You can check the time since the last reboot with the command `uptime`, which will respond with a line like

```
14:30:32 up 34 min, load average: 1.24, 1.32, 1.10
```

In this example, the DCM last rebooted 34 minutes ago.

The most common circumstance where `guardian` will reboot a DCM is when the operator has instructed it to record data, but it cannot do so. This may be because

- the Flash memory is full, and all connected USB disks are also full,
- the Flash memory is full, and no USB disk is present;
- the Flash memory is full, and a USB disk is inaccessible for some other reason (*e.g.* it is unformatted, incorrectly partitioned, or faulty); or
- the Flash memory is being filled up faster than the DCM can empty it. This may happen if the individual files are longer than 25% of the total Flash memory in each of the two memory banks (*e.g.* 16 Mb, for DCMs with 128 Mb of storage.) If you encounter this problem, try making the watch files shorter by changing the value of *disk.recordinterval* (see Section 6.3, page 86.)

3.6 CMG-AM Authentication Modules

The CMG-AM is a variant of the CMG-DCM which includes strong cryptographic capabilities, including an on-boad Spyrus crypto-token.

This token can be used to sign all outgoing data at source. The signature matches the data as produced, so if anyone alters the data, the signature will no longer be correct.

Before your DCM can start sending authenticated data, it must

- generate a cryptographic key pair;
- request a certificate from a trusted Certificate Authority, and be assigned one;
- run a data server or transmitter using a protocol designed with provision for authenticated seismic data (*e.g.* CD1.0 and 1.1);
- using the certificate and key, begin authenticating the data using the method specified in the protocol.

These steps can be executed remotely, but must be carried out separately for each installation.

Hardware authentication

Where a Spyrus crypto-token is installed, the program `spyrus` should be used to set it up.

1. From a computer on the local network, use a suitable program to open a *ssh* session with the AM. For example, from a Linux computer:

   ```
   ssh 10.82.0.129
   ```

   where `10.82.0.129` should be replaced with the IP address of the AM on your network.

2. Log in using the username and password you use to access the AM's Web interface. (If you do not use a username and password to access the AM's Web interface, you should obtain one from your network administrator.)

3. Initialise the Spyrus card with the command

   ```
   spyrus zeroize
   ```

   followed by

   ```
   spyrus start
   ```

   You should see information messages coming from the Spyrus card, as follows:

   ```
   Initial state 8(Zeroized)
   info::state: Zeroized
   ```

```
info::logging into zeroized card
info::state: Uninitialized
info::logging into unitialized card
info::loading initialization values
info::state: Initialized
info::changing SSO password
info::state: SSO Initialized
info::logging into the card as SSO
info::loading the trusted certificate
info::state: LAW Initialized
info::logging into the card as SSO
info::setting the user's password
info::state: User Initialized
info::logging into the card as User
info::state: Standby
info::Final state 6(Standby)
```

If the AM reports an error, the spyrus package is not installed or cannot function. In this case, you should use `openssl` to generate keys in software as described below.

4. Now log in using

   ```
   spyrus login
   ```

5. Create a text file containing information about the station, sensor, etc., in the following format:

   ```
   organizationName:your-organization
   organizationalUnitName:station-name
   organizationalUnitName:operator-name
   localityName:HPAXX
   commonName:HPA0-02
   commonName:HPA0-01
   ```

   This will be used to request the certificate. You should replace the information after the colons with the correct data for the specific pit installation (available from the NDC or the station administrator.) Check that the field names (before the colons) are correctly capitalized, and do not leave spaces around the colons.

   You can have several `organizationalUnitName` and `commonName` records, in which case the information is entered into the database from bottom to top, so that the most recent entries appear first in the file.

   (To create a text file in Linux, either use a command like `cat > filename` and enter the data directly, ending with `CTRL-D`, or use the text editor `vi`. Using `vi` will allow you to edit the file should you make a mistake in entering the data. For more information, see the Linux manual page for `vi`. Alternatively,

you can create text files on the local computer and transfer them to the AM module using `scp` or a similar secure transfer program.)

6. Change into the CD1.1 transmitter's configuration directory using

   ```
   cd /etc/libcd11
   ```

   The following steps create a key pair and certificate request within the token, which need to be placed in this directory for the CD1.1 transmitter to be able to sign outgoing data.

7. Issue the command

   ```
   spyrus newreq -s filename -i 1 -r slot01.req -p slot01.pub
   -x
   ```

   where *filename* is the name of the file you created in step 5. This will generate a certificate request in the file `slot01.req` and a public key in the file `slot01.pub`. The private key is kept within the token itself, and cannot be extracted from it. Any attempt to compromise the token will cause it to shut down and become unusable.

8. The file `slot01.req` is a certificate request for the key pair generated. You should send this file by e-mail to the Certification Authority, so that they can generate a valid certificate from it.

9. When you receive the certificate, install it in the `/etc/libcd11` directory as `slot01.crt`. Also create the key ID file `slot01.kid`. (The key ID file is simply a text file containing the key ID as a single decimal number. You can use any key ID number as long as it is unique for each key. It is used in the key bucket file, described below.)

10. Now load the certificate into the token using the command

    ```
    spyrus loadcert -c slot01.crt
    ```

    The token will check that the certificate matches its own key pair, and should respond with

    ```
    info::No index specified searching for matching key
    info::Key in slot 1 matches certificate
    ```

If it reports that the key does not match the certificate, you may have attempted to load a certificate valid for the wrong token. Check the certificates you have received, and try again. Otherwise, you may have to generate a new certificate request and re-send to the Certification Authority.

If you issue `spyrus loadcert` without specifying a slot number, as above, any running CD1.1 transmitters will be interrupted, and you will need to restart them.

11. The token is now ready to start signing outgoing CD1.1 subframes. However, you will need to configure the format of these subframes by editing the `/etc/cd11sf.cfg` configuration file. You can do this either directly, or using the Web configuration interface (see Chapter 6, page 80.)

12. Any further key changes can be handled automatically over AutoDRM. However, occasionally you may want to supersede an existing key, or create a new key for a separate stream.

    Keys are handled by a system of *key buckets*. Each key bucket consists of a list of keys and activation times. Once the activation time for a new key passes, the previous key is superseded, and subsequent subframes are signed by the new key. You can have a different key bucket active for each stream, or even several key buckets for the same stream.

    Key buckets are stored in the files `0.bkt`, `1.bkt`, etc., within the `/etc/keybuckets` directory. Each line in a key bucket file has the format

    key-id:*days-since-epoch*:*seconds-since-day-start*

    where *days-since-epoch* is the number of days elapsed since November 17, 1989. The CD1.1 transmitter scans this file in order, and stops when it finds a key with an activation time in the past (relative to the time-stamp of the data being transmitted.) Thus, to supersede an existing key, you must place the new entry *before* the old one in the file, so that the CD1.1 transmitter will not continue signing subframes with the old key.

    To make the CD1.1 transmitter sign *all* subframes with a new key, even when backfilling, you should add the line

    key-id:0:0

to the beginning of the file.

13. Restart the CD1.1 transmitter with the command `killall -HUP cd11sf`. (Using the `-HUP` option makes the command send a hangup signal to the CD1.1 transmitter rather than killing it outright.)

Software authentication

Some AM units are not supplied with hardware crypto-tokens. These units can still perform authentication using the `openssl` package. To set up an AM using `openssl`:

1. Change into the CD1.1 transmitter's configuration directory using

   ```
   cd /etc/libcd11
   ```

2. Generate the DSA parameters file with

   ```
   openssl dsaparam -out slot01.prm key-length
   ```

   where *key-length* is the size of the key you wish to be generated (normally 1024.)

3. Generate the public and private keys with

   ```
   openssl dsaparam -in slot01.prm -out slot01.key -genkey
   ```

4. Create a configuration file in the format

   ```
   [req]
   default_bits                    = key-length
   distinguished_name              = req_dn
   [req_dn]
   organizationName              = Enter organization name (eg,
   company)
   organizationName_value        = organization-name
   0.organizationalUnitName        = Enter organizational unit
   name (eg, section)
   0.organizationalUnitName_value  = organizational-unit-name
   1.organizationalUnitName        = Enter organizational unit
   name (eg, section)
   1.organizationalUnitName_value  = organizational-unit-name
   localityName                    = Enter your station name
   localityName_value              = station-name
   commonName                    = Enter your site name
   commonName_value              = site-name
   ```

   You should *only* replace the words highlighted *like this* in the above file. The lines ending `Enter your site name`, *etc.,*

are used by `openssl` to generate prompt strings and need not be changed. If you need to enter two separate values for the same key (as `organizationalUnitName` above) you should prefix the pairs with `0.`, `1.`, etc., as shown.

5. Generate a certificate request with

   ```
   openssl req -new -key slot01.key -days validity-period
   -config config-file -out slot01.req
   ```

   where *validity-period* is the number of days' validity you want for the request, and *config-file* is the configuration file you created in the previous step.

   The file `slot01.req` is a certificate request for the key pair generated. You should send this file by e-mail to the Certification Authority, so that they can generate a valid certificate from it.

6. When you receive the certificate, install it in the `/etc/libcd11` directory as `slot01.crt`. Also create the key ID file `slot01.kid`. (The key ID file is simply a text file containing the key ID as a single decimal number. You can use any key ID number as long as it is unique for each key. It is used in the key bucket file, described below.)

7. The AM is now ready to start signing outgoing CD1.1 subframes. However, you will need to configure the format of these subframes by editing the `/etc/cd11sf.cfg` configuration file. If you intend to use CNSN authentication, you should also edit the `/etc/cnsn.cfg` configuration file. The AM's configuration files can be edited using its Web page interface or with an editor.

8. Any further key changes can be handled automatically over AutoDRM. However, occasionally you may want to supersede an existing key, or create a new key for a separate stream.

   Keys are handled by a system of *key buckets*. Each key bucket consists of a list of keys and activation times. Once the activation time for a new key passes, the previous key is superseded, and subsequent subframes are signed by the new key. You can have a different key bucket active for each stream, or even several key buckets for the same stream.

   Key buckets are stored in the files `0.bkt`, `1.bkt`, etc., within the `/etc/keybuckets` directory. Each line in a key bucket file

has the format

key-id:*days-since-epoch*:*seconds-since-day-start*

where *days-since-epoch* is the number of days elapsed since November 17, 1989. The CD1.1 transmitter scans this file in order, and stops when it finds a key with an activation time in the past (relative to the time-stamp of the data being transmitted.) Thus, to supersede an existing key, you must place the new entry *before* the old one in the file, so that the CD1.1 transmitter will not continue signing subframes with the old key.

To make the CD1.1 transmitter sign *all* subframes with a new key, even when backfilling, you should add the line

key-id:0:0

to the beginning of the file.

9. If you are running it, restart the CD1.1 transmitter with the command `killall -HUP cd11sf`. (Using the `-HUP` option makes the command send a hangup signal to the CD1.1 transmitter rather than killing it outright.)

10. If you are running it, restart the CNSN transmitter with the command `killall -HUP cnsnauth`

4 Tools

The menu bar of the DCM's Web site is divided into four sections.

- At the top is a banner identifying the DCM, with an optional comment. There is also a link to the **Summary** page, which is the one you see when you first open the DCM's Web site.

- The next section, **Tools**, contains links which perform various actions on the DCM. This includes viewing data, transferring files from Flash memory and the on-board USB disk, and troubleshooting.

  Once a DCM has been configured for use in a station, operators will generally only need to use the options in this section.

- **Data Transfer** contains all the settings which relate to the DCM's primary function as a seismic data module. Here you will find servers and clients for all the seismic protocols supported by your DCM. See Chapter 5, page 64, for details.

- **Configuration** contains the remaining DCM settings, including hardware setup, networking and user management. See Chapter 6, page 80, for details.

4.1 Summary

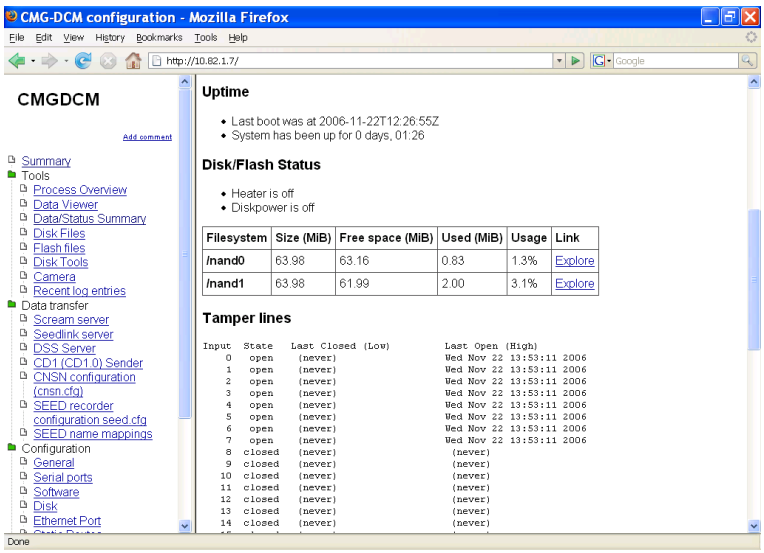
This link displays a page summarizing the current setup of the DCM. When you first log in to the module over its Web interface, this is the page you are initially presented with. It is divided into a several sections:

Serial ports

This table shows all the serial ports connected to the DCM, with information about each one. Each row also contains lnks to configuration pages for the port and any digitizer connected to it.

The table is also shown when you choose **Configuration – Serial ports**. For full details on the table and on serial port configuration options, see section 6.2, page 82.

Uptime



This section displays the time the DCM last rebooted and calculates how long it has been running since that time.

Disk/Flash Status

This section reports the current status of the DCM's storage media.

- First, the DCM reports whether the disk is currently powered, and whether the heater is switched on.

- Next is a table showing each file system in use by the DCM, with its size and current usage (in Mb and as a percentage.) To see the files in each file system, click **Explore**.

A standard DCM has two Flash partitions mounted at `/nand0` and `/nand1`. These are used alternately to record data. If the DCM has recently used its USB disk, this will also be shown in the table.

The DCM does not power up the USB interface specially to build this table, so if you remove the USB disk the DCM may continue to include it in the table until it attempts a disk operation.

Tamper lines

This section, if present, reports the status of the tamper lines relayed to the DCM over an external State of Health interface:

```
Input  State   Last Closed (Low)          Last Open (High)
    0   open    (never)                   Wed Jul  7 16:04:48 2004
    1   open    (never)                   Wed Jul  7 16:04:48 2004
    2   open    (never)                   Wed Jul  7 16:04:48 2004
```

```
 3   open    (never)                  Wed Jul  7 16:04:48 2004
 4   open    (never)                  Wed Jul  7 16:04:48 2004
 5  closed  Wed Jul  7 16:04:48 2004   (never)
 6   open    (never)                  Wed Jul  7 16:04:48 2004
 7  closed  Wed Jul  7 16:04:48 2004   (never)
 8  closed  Wed Jul  7 16:04:48 2004   (never)
 9  closed   (never)                    (never)
10  closed   (never)                    (never)
11  closed   (never)                    (never)
12  closed   (never)                    (never)
13  closed   (never)                    (never)
14  closed   (never)                    (never)
15  closed   (never)                    (never)
```

The columns in the table give, in turn, the current state of each switch and when it was last observed to be in each state. The above example shows a typical reading for a set of 9 tamper switches that have not triggered: the normally-closed switches show the current time under “Last Closed”, and the normally-open switches under “Last Open”. All other fields show `(never)`, indicating that no switches have been observed in the “wrong” state since the DCM was last booted.

Network configuration

This section displays basic information about the DCM's network setup. It is identical to the output from the Linux `ifconfig` program. A typical reading might look like this:

```
eth0 Link encap:Ethernet HWaddr 00:D0:1F:34:EB:08
inet addr:192.168.0.46 Bcast:192.168.0.255 Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:3619 errors:0 dropped:0 overruns:0 frame:0
TX packets:2453 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:100
RX bytes:290005 (283.2 kb) TX bytes:417231 (407.4 kb)
Interrupt:42 Base address:0x8300
```

For further information, see the Linux manual page for `ifconfig`.

DNS configuration

This section reports the current status of the DCM's domain name resolution service. This is done by presenting the contents of the standard Linux `/etc/resolv.conf` file.

```
# eth0 begin
domain guralp.local
nameserver 192.168.0.1
nameserver 192.168.0.2
# eth0 end
```

For further information, see the Linux manual page for `resolv.conf`.

Unique IDs

This section contains unique identifiers for each of the hardware boards inside the DCM.

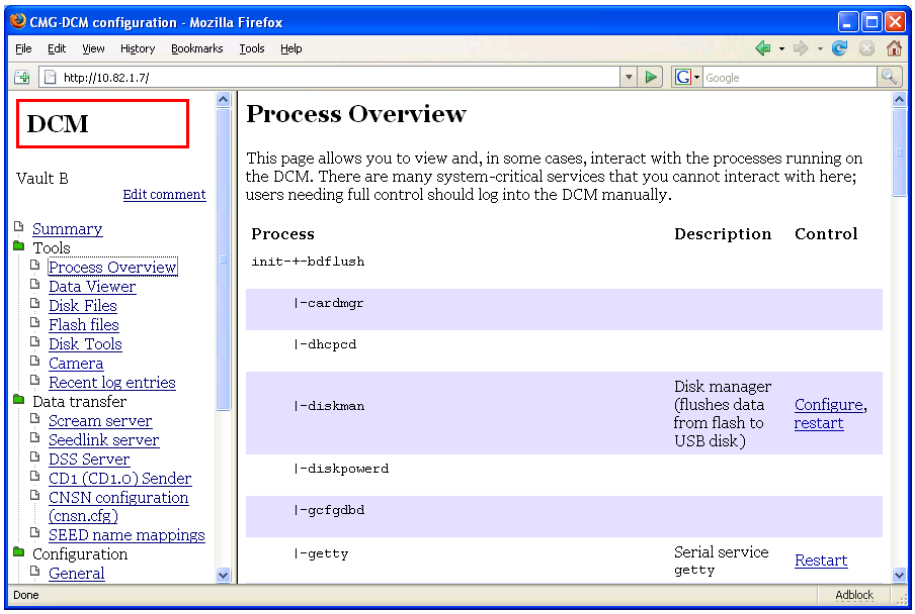
If you have a problem with the DCM, or it seems to be damaged, we may ask you to provide us with these IDs. You should write them down and keep them safe in case you cannot access the Web server in this event.

Software Versions

The final section lists the software currently installed on the DCM's Linux operating system, together with each program's version number. If you need to contact Güralp Systems about any of the installed programs, you should quote the version number in your correspondence.

4.2 Process Overview

Clicking **Tools – Process Overview** displays a page containing a list of all the processes currently running on the DCM. The list is generated using the standard command `ps`.



Some processes are required for the Linux operating system to function. You cannot change or restart these processes from the **Process Overview** page. Other processes are shown with a short description and control options.

If a **Restart** link is shown beside a process, clicking it will kill the

process and run a new one. This is useful if a process appears to have stopped responding.

Restarting some processes may cause data transfer to be interrupted. For network services, clients will normally have to reconnect to the DCM to resume data transmission. In some cases, data may be lost. If you are unsure, contact Güralp Systems.

If a **Configure** link is shown beside a process, clicking it will take you to the DCM configuration options which relate to that process. For some processes, more than one link is present. For example:

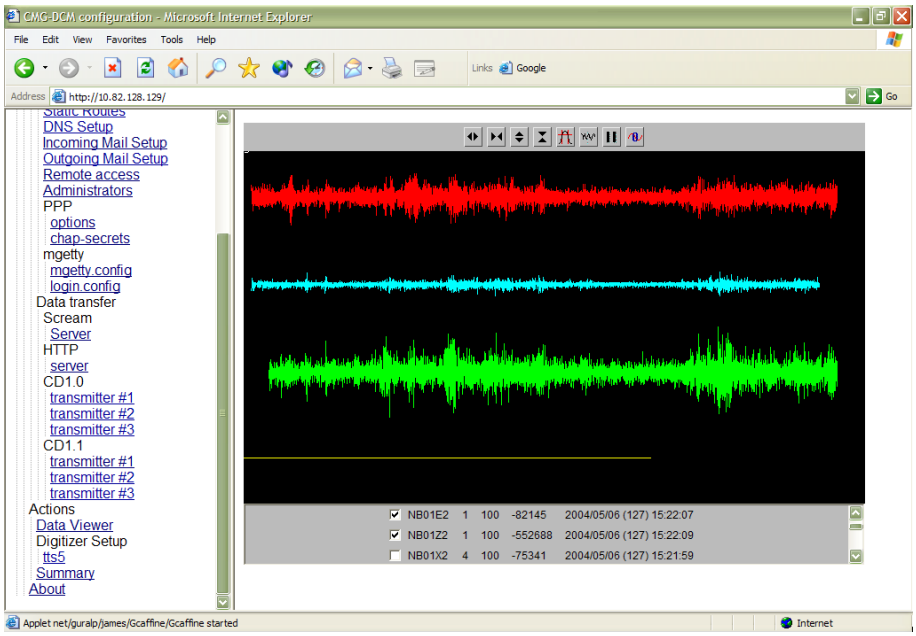
- Clicking the **Configure** link beside `screamserver` takes you to the Scream! server configuration page.
- Clicking the **Configure** link beside `seedrecorder` edits the SEED configuration file, whilst **Recording options** links to the *Disk* configuration page.
- Clicking the **Mappings** link beside `msrecorder` or `slserver` (the MiniSEED and SeedLink services, respectively) takes you to the *SEED name mappings* page.

4.3 Data Viewer

The *Data viewer* page uses a simple Java applet to enable you to check that the DCM is receiving data correctly. Your browser must have Java installed for you to be able to do this.

The data viewer is not intended to be a fully functional data visualisation tool. Scream! and similar software packages offer a full range of facilities for displaying and manipulating seismic data, either as it arrives or from stored data files.

The main part of the display shows a selection of the data streams being recorded, whilst the bars at the top and bottom allow you to control what is displayed, and in what manner:



If you cannot see any data in the Data Viewer, you should check first that

- some streams have been checked in the lower portion of the applet, and
- the data has been offset correctly. You can automatically zero the streams by clicking on the icon depicting a “0” and a sine wave in the icon bar. Otherwise, you can zoom out until the streams are visible.

The icon bar

The bar at the top of the applet allows you to alter how the data is displayed. If you are familiar with Güralp Systems' Scream! package, you will recognise the icons in use here. There are:

- four icons allowing you to alter the X and Y scaling factors of the graph (from left to right: magnify X, reduce X, magnify Y, and reduce Y)
- a bandpass filter toggle (not currently available),
- a toggle to display the endpoints of each GCF block as a white line,
- a pause button, which will cause the viewer to stop scrolling as it receives new data, and

- a zeroing button, which attempts to centre all the selected streams in the display.

The stream list

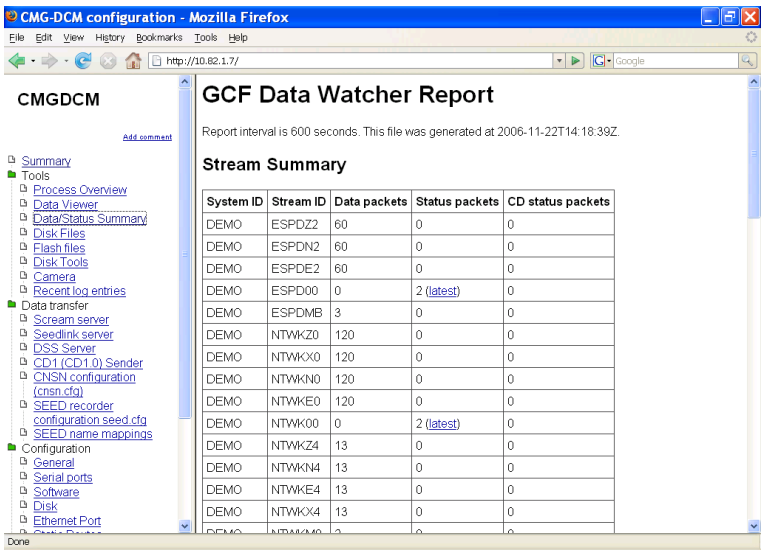
Below the display is a list of all the streams currently being recorded by the DCM. Each has a checkbox indicating whether that stream is currently being displayed.

To the right of the checkboxes, you are given information about each stream, similar to that found in the main window of Scream!. The information provided is:

- The ID of the stream (six characters long);
- The compression code corresponding to the bit depth of the data. 0 = 8-bit data, 1 = 16-bit data, 2 = 24-bit data, and 3 = 32-bit data;
- A recent FIC for the data. The FIC gives a rough idea of the magnitude of the signal being received.
- Finally, the time the most recent data block was received, including, in brackets, the day number within the year (1 = January 1, etc.)

4.4 Data/Status Summary

This link displays information about the data streams being received by the DCM.



This page is updated every 10 minutes. If you make a change to your installation, you will need to wait 10 minutes before it will be reflected in this page.

At the top of the page is a *Stream Summary* table. This table lists all the data streams which the DCM has received in the last 10 minutes.

- The first two columns are the *System ID* and *Stream ID* for the stream.

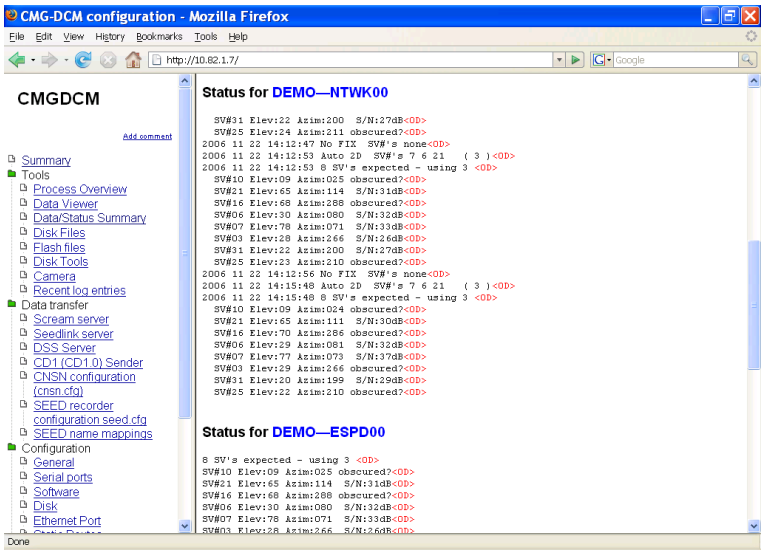
- The third column is the number of data blocks which were received in the last 10 minutes. The number you should expect here depends on the sample rate and the compression level of your data.

  For example, a 100 samples/s stream at 16-bit compression (moderate noise/activity), will contain 5 seconds of data in each block, so 120 blocks of data would be generated in 10 minutes.

- The fourth column is the number of status blocks received in the last 10 minutes. Click on **Latest** to show the content of the most recent status block for that stream.

- The last column is the number of CD status blocks received in the last 10 minutes. Digitizers do not output CD status blocks by default, so this column will be zero unless you have reconfigured your digitizer.

Below the stream table, the DCM prints out the content of the latest block produced by each status stream. You can use this to check on the health of your digitizer.



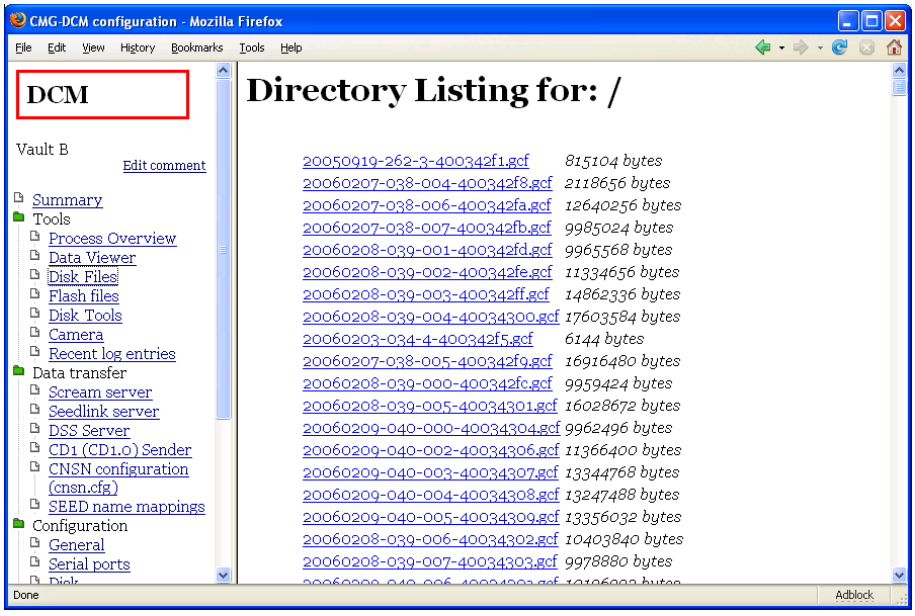
Finally, at the bottom of the page, is a table headed *Shared Memory Diagnostics.* This table lists the number of blocks which have passed through the DCM's internal shared memory areas. One shared memory space is allocated to each input port, but extra areas may also exist.

The entries in this table give the key index of each shared memory area, its name (if it has one – this may be taken from its allocated port), and the number of data, status and CD blocks received.

The “Bad blocks” column reports how many corrupt blocks have been detected in this shared memory area. A “bad block” is either one which could not be parsed, or one whose checksums do not match the data. Bad blocks are noted, but not discarded.

4.5 Disk files

This link allows you to browse through any files currently on the DCM's primary hard disk.



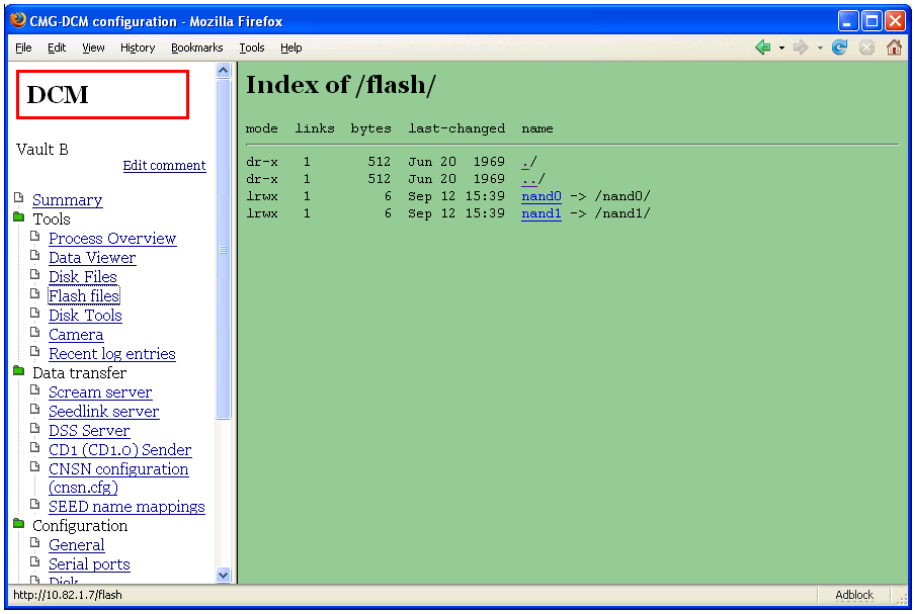
The name of each file is shown, together with its size. If there are directories on the disk, they are shown at the end of the list. The amount of free space on the disk is also displayed.

Click on a link to download a file or enter a directory.

If no suitable storage medium can be found, the module will report the error `Failed to open USB disk`.

4.6 Flash files

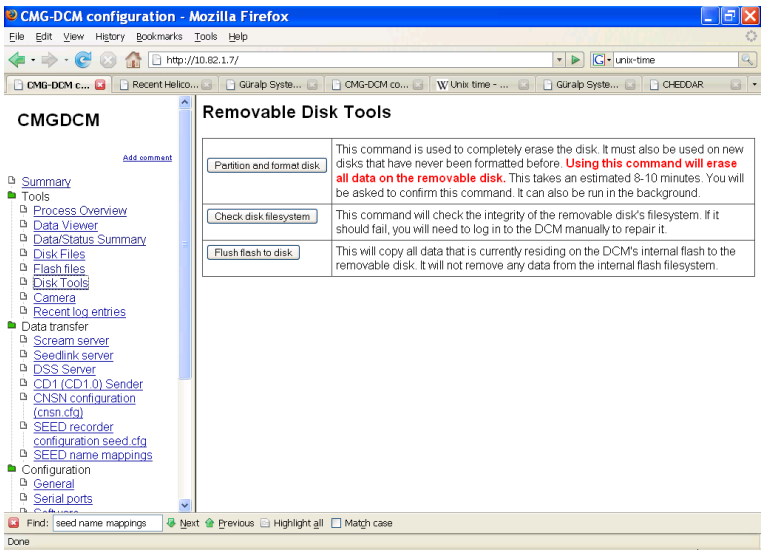
This link opens a page similar to **Disk files**, but which explores the DCM's internal Flash memory.



The two directories `nand0` and `nand1` denote the two regions of Flash memory used by the DCM to store data. Click on one of them to view the files.

4.7 Disk tools

The buttons on the **Disk tools** page allow you to perform some important actions on the DCM's primary hard disk.



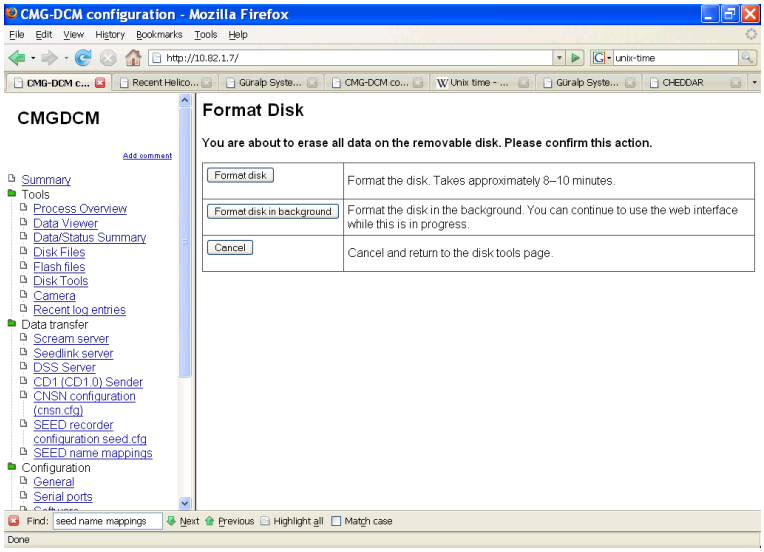
By default, the module looks for a connected USB hard disk to use as its primary storage medium. If no suitable storage medium can be found, you will see the message `Failed to find a USB disk` when you attempt to perform any of these actions.

Partition and format disk

Before using the CMG-DCM you should ensure that its primary hard disk is ready to receive data by partitioning and formatting it. The DCM uses a special journalling format for the hard disk which is designed to maintain the integrity of your data at all times.

The hard disk is guaranteed to contain a FAT32-compatible filesystem, even if a write operation fails or is aborted suddenly.

To format the disk, click **Partition and format disk.** You will be taken to a confirmation page.



- Click **Format disk** to format the disk and display progress. This operation takes around 10 minutes. You *must not* close the browser window or navigate elsewhere whilst you are doing this. Closing the browser window will abort the operation and leave you with a partially-formatted disk.

- If you want to close the browser window or perform other tasks, click **Format disk in background**. The DCM will format the disk but will not display its progress. You should expect it to finish in around 10 minutes.

  Whilst the disk is formatting, you will not be able to access it from other parts of the Web page (*e.g.* **Disk files**). If you try, the

DCM will report `Disk is already in use`.

- Click *Cancel*, or navigate elsewhere, to cancel the format.

Check disk filesystem

Clicking on this button will verify the integrity of the hard disk's filesystem. It is recommended that you do this immediately after installing the device.

Flush flash

Clicking on this button will cause the DCM to dump the current contents of its Flash memory to the hard disk, thus synchronizing it with the most current data. If you want to remove or replace the DCM's hard disk, using this tool will ensure that the outgoing disk is up-to-date.

Flush flash does *not* remove data from the DCM's on-board Flash memory. If you *Flush flash* and then swap hard disks, the data remaining in memory will later be written out to the new hard disk, causing some overlap between it and the old disk.

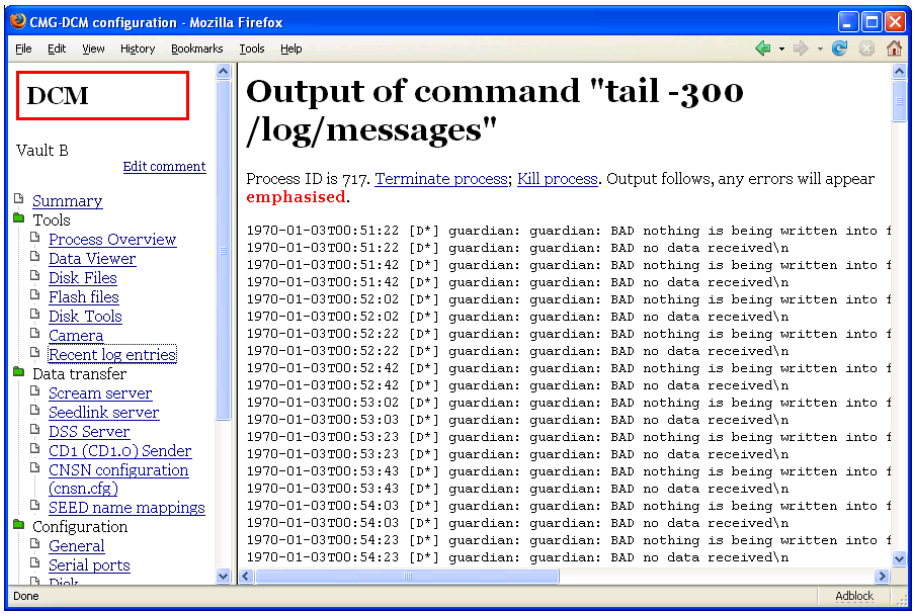
Whilst the DCM is copying the contents of the Flash memory to disk, you will be shown a log of its progress. The USB interface allows data transfer at a speed of around 100 Kb/s, so large files may take several minutes to complete. If an error occurs at any point, it will be marked in red.

4.8 Camera

If you have attached a compatible camera to the external USB port of the DCM, clicking on this link will show the current view from the installation site. A new image is retrieved every 12 seconds, or whenever you reload the page. Güralp Systems can supply a compatible camera with the DCM, although any STV0680-based device can be used.

4.9 Recent Log Entries

This link displays the most recent 300 entries written to the DCM's internal log file.



Each entry includes, in order:

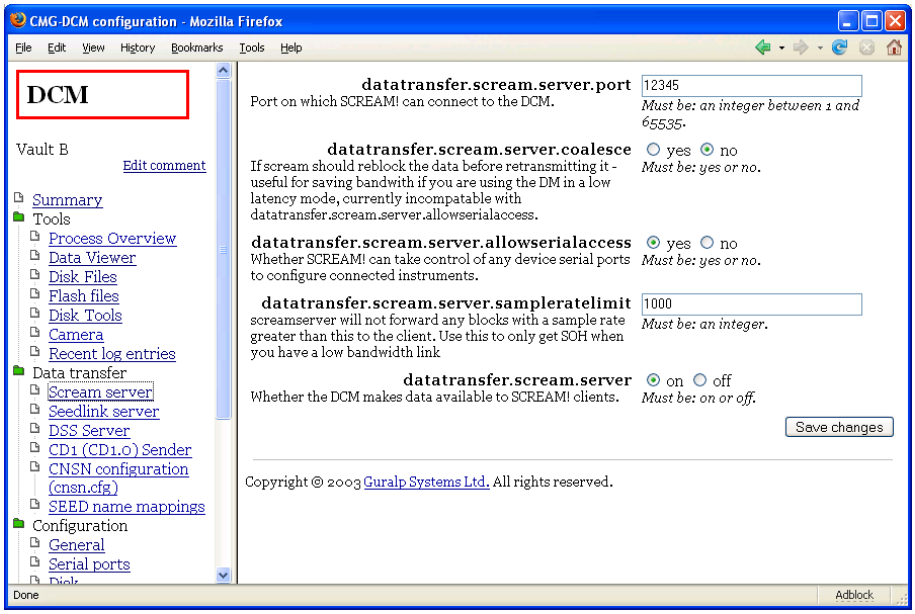
- the date and time of the log message,
- a code `[Xx]` describing the type of program and the importance of the message (*e.g.* all errors have E as the second letter: see the Linux `man` page for `syslog` for full details),
- the name of the program (*e.g.* `guardian_log`) which generated the message,
- the text of the message.

5 Data transfer

You can retrieve data from the DCM using a variety of standard seismic network protocols and formats, including Güralp Systems' own Scream! software. The options beneath **Data transfer** are used to configure these network services.

For data transfer options which use the serial ports of the DCM (*e.g.* DSS summary mode, etc.) you should use the **Configuration – Serial ports** pages to set the *serial.x.service* option instead (see Section 6.2, page 82.)

5.1 Scream! server



Scream! is a Microsoft Windows software package designed to configure and retrieve data from Güralp digitizers. The options on this page allow you to control how the DCM responds to requests from Scream!. Depending on your requirements, the DCM can act either as an instrument in itself, or pass through requests directly to connected equipment.

datatransfer.scream.server : Select *on* to allow Scream! to connect to the module over a TCP/IP connection and receive data as if it were a connected instrument. The connection can be established over an Ethernet, wireless, or PPP link. If you do not want to transfer data directly to Scream! clients, select *off*.

datatransfer.scream.server.allowserialaccess : The Scream! software allows you to configure digitizers connected to it by serial links. Select

yes if you want to be able to use Scream! to configure digitizers attached to the DCM's serial ports. The software is able to differentiate between several instruments connected to the DCM, so enabling this option will allow you to configure all attached digitizers from within Scream!.

datatransfer.scream.server.port : The network port number which Scream! clients should use to connect to the DCM. You can use any port which neither the PC nor the DCM is using for other purposes. See your Scream! configuration for the current port setting; the default is 1567.

datatransfer.scream.server.udp_push : A list of clients to send UDP data, separated by commas.

Normally, clients connect to the DCM's Scream! server by sending a special UDP packet called `GCFSEND`, or by making a TCP connection to the server. In both of these cases the client is "pulling" the data from the DCM. Any client can connect to the DCM, but the client needs to know the DCM's IP address.

In some cases, *e.g.* when the DCM has a dynamic IP address, you may want the DCM to "push" UDP data to clients. To do this, you should add the clients to this list.

Each entry in the list is either the IP address or the hostname of the client. The DCM will send data to port 1567 by default. If you want to use a different port, use *IP-address*:*port* or *hostname*:*port*.

For example, the value

```
192.168.2.2, 82.68.239.4:8888, screamclient.remote.net
```

would cause the DCM to push data to port 1567 at `192.168.2.2` and `screamclient.remote.net`, and to port 888 at `82.68.239.4`.

If UDP "push" is active, other clients can still connect to the DCM and "pull" data from it as normal.

datatransfer.scream.server.sampleratelimit : The fastest sample rate allowed for transmission over this connection. Using this option, you can prevent high-rate streams from being transmitted to Scream!, whilst allowing them to be transmitted by other means (*e.g.* CD1.1, etc.)

Setting this option to zero will filter out all data streams but preserve status channels, allowing you to use Scream! to monitor the status of

your instrumentation whilst minimizing the bandwidth used.

datatransfer.scream.server.coalesce : Some digitizers are configured to produce GCF blocks at higher than normal rates. This may be done, for example, to reduce the latency in data transmission. Blocks produced this way may not all be full, so space will be wasted in the recording medium.

Selecting this option makes the DCM build full GCF blocks from the data it receives (a process known as *re-blocking*) before transmitting them.

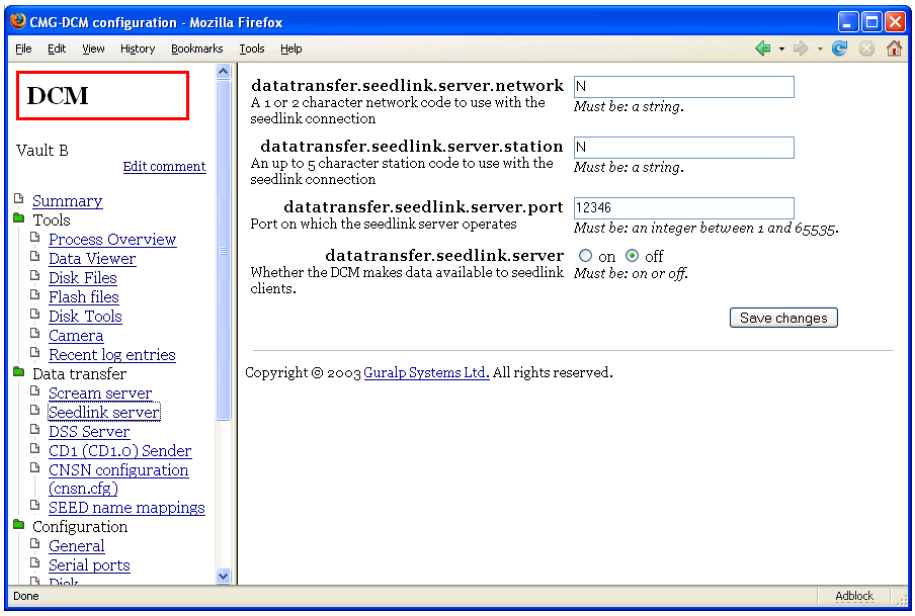
If you enable *datatransfer.scream.server.coalesce*, you will not be able to connect to the terminal of any connected digitizers using the Scream! server (*i.e.* the DCM will behave as if *datatransfer.scream.server.allowserialaccess* was *no.*)

You can still configure the digitizers using the Web page (Section 7, page 105) or using command line tools like `minicom` and `gcli` (see Section 8.2, page 117).

5.2 SeedLink

SeedLink is another member of the SEED / MiniSEED family of formats, designed for transferring seismic data over a network. The DCM's `slserver` package provides a SeedLink server.

SeedLink expects streams to be named using station, channel, and network codes according to the FSDN SEED naming convention. Before using the SeedLink server, you will need to define these codes for every stream you expect to receive. This is done on the **Seed name mappings** page (see Section 5.7, page 75.)



The following configuration options are available:

datatransfer.seedlink.server.network : The default 1- or 2-character SEED network code to use for this station.

datatransfer.seedlink.server.station : The default SEED station code to use, up to 5 characters long.

datatransfer.seedlink.server : Select `on` to enable the server, `off` to disable it.

datatransfer.seedlink.server.port : The network port on which to listen for incoming connections.

Before you can start the SeedLink server, you will need to tell it which streams to output, and what SEED codes to use for each one. This is done on the *Seed name mappings* page (see Section 5.7, page 75.)

You should make sure that each stream uses the same SEED network and station codes as you have configured for the server on this page.

When you have finished editing these options, click **Save changes** to commit them.

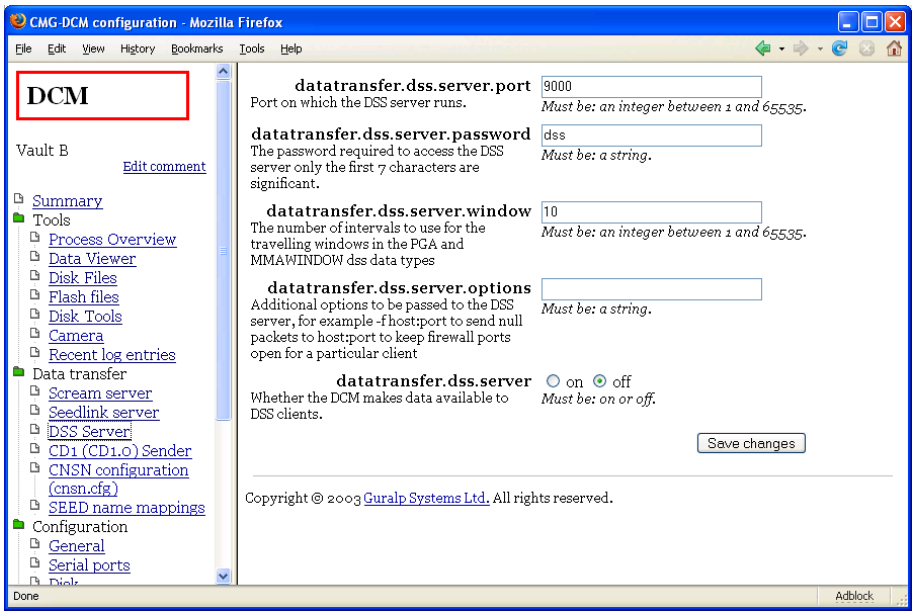
The SeedLink server will not immediately use the new options, because clients may still be connected. You will need to restart the SeedLink server at a convenient time. Clients will have to reconnect to the new instance of `slserver` before they can continue receiving data.

To restart the SeedLink server, browse to the **Tools – Process Overview** page and find the entry for the `slserver` process. Click the **Restart** link in this entry.

5.3 DSS

DSS (Data Subscription Service) is a packet format which enables data and statistics to be requested from a seismic installation. A DSS server is designed to handle many concurrent requests from clients with varying levels of privilege, and may prioritize requests according to their origin and urgency.

Güralp Systems' data modules include a package, `libdss`, which is designed to communicate with installations using DSS as either a server or a client. A daemon utility, `dssserver`, is also available which receives requests on a network port and replies to them.



The dssserver program

`dssserver` initially listens on a network port for DSS_REG registration packets from clients. Any incoming connnections other than DSS_REG are refused with a DSS_REF packet. Receiving a valid DSS_REG packet, dssserver sends a DSS_ACK packet and adds the client to an internal client list, identified by its source IP address and port number.

Once a client is registered, `dssserver` will accept other DSS command packets from it.

`dssserver` fulfils requests by following this procedure:

- Receiving a valid DSS_REQ packet, `dssserver` will send a DSS_ACK packet immediately and add the request to an internal subscription list.

- Receiving a valid DSS_DEL packet, `dssserver` will find the corresponding request in the subscription list, remove it, and reply with a DSS_ACK packet. If the relevant data has already been transmitted, `dssserver` will still send the DSS_ACK packet to notify the client that the request has been dealt with.

- `dssserver` keeps a record of the subscriptions for each client so that it can respond to DSS_LRQ packets. When a subscription is fulfilled, it is removed from the relevant list.

- Other valid DSS commands are dealt with according to the standard.

- When data arrives from connected digitizers, `dssserver` examines the current queue of data requests, and generates the required data or averages separately for each client, in accordance with these requests. This data is placed in a *train,* which is a data structure whose length is determined by the reporting interval. Every train includes data from the vertical, N/S, and E/W component of a particular instrument.

- As soon as the received or generated data fills the buffer, it is ready for dispatch. `dssserver` forms the data into a DSS_DAT message and sends it to the client, identifying it with the relevant *Data Identification Word*.

- Every 10 minutes or thereabouts, dssserver checks the client list for any clients which have not reported to it in the past 10 minutes, and purges the inactive clients from the list.

`dssserver` does not implement advanced prioritization or queuing schemes. All clients connect with the same password.

Server configuration

DSS expects streams to be named using station, channel, and network codes according to the FSDN SEED naming convention. Before using the SeedLink server, you will need to tell it which streams to output, and what SEED codes to use for each one. This is done on the *Seed name mappings* page (see Section 5.7, page 75.)

The options on the **Data transfer – DSS** page are:

datatransfer.dss.server : Set this to *on* to run the server; *off* to disable it.

datatransfer.dss.server.port : The port number on which to listen for incoming DSS packets. The default is 9000.

datatransfer.dss.server.password : The password all clients should use when registering with `dssserver`.

datatransfer.dss.server.window : The number of DSS intervals to use for travelling windows (used in the PGA and MMAWINDOW data types). The default is 10.

datatransfer.dss.server.options : Any additional options to send to the DSS server process (for advanced usage.)

Remember to click *Save changes* before you browse away from the page, if you want to keep the changes you have made.

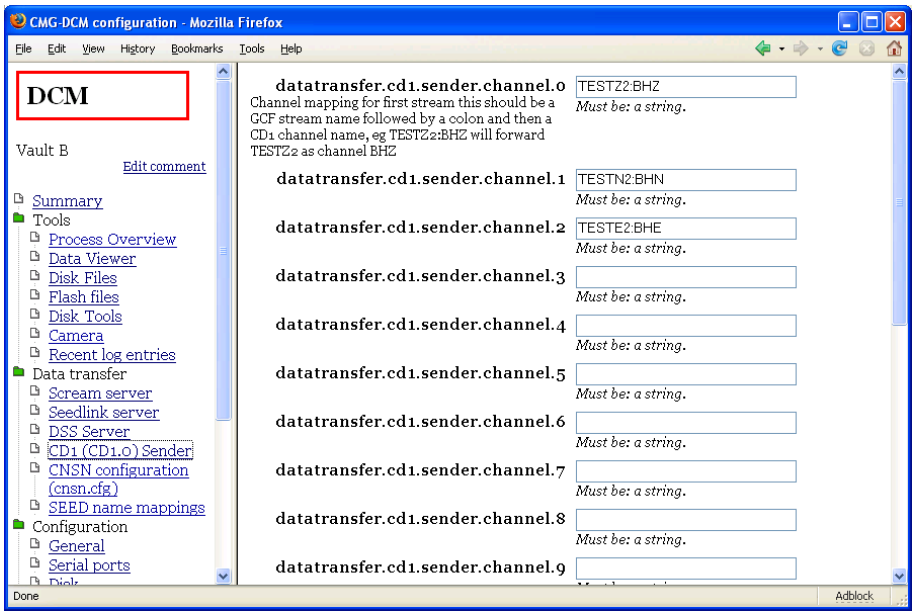
The DSS server will not immediately use the new options, because clients may still be connected. You will need to restart the DSS server at a convenient time. Clients will have to reconnect to the new instance of `dssserver` before they can continue receiving data.

To restart the DSS server, browse to the **Tools – Process Overview** page and find the entry for the `dsserver` process. Click the **Restart** link in this entry.

5.4 CD1 (CD1.0) Sender

Continuous Data Format, CD1.0, is a standard method for sending seismic data over a TCP/IP network. The DCM module can send data in CD1.0 format to a specific CD1.0-compatible device or NDC.

Some installations of the DCM have multiple CD1.0 senders. In this case, there will be several *sender* pages, all with the same options.



datatransfer.cd1.sender : Select *on* to activate the CD1.0 sender. If you do not want the DCM to transmit CD1.0 data, select *off* and ignore the remaining settings.

datatransfer.cd1.sender.sitename : The CD1.0 site name for the DCM.

datatransfer.cd1.sender.receiver : The IP address or hostname of the remote CD1.0 device.

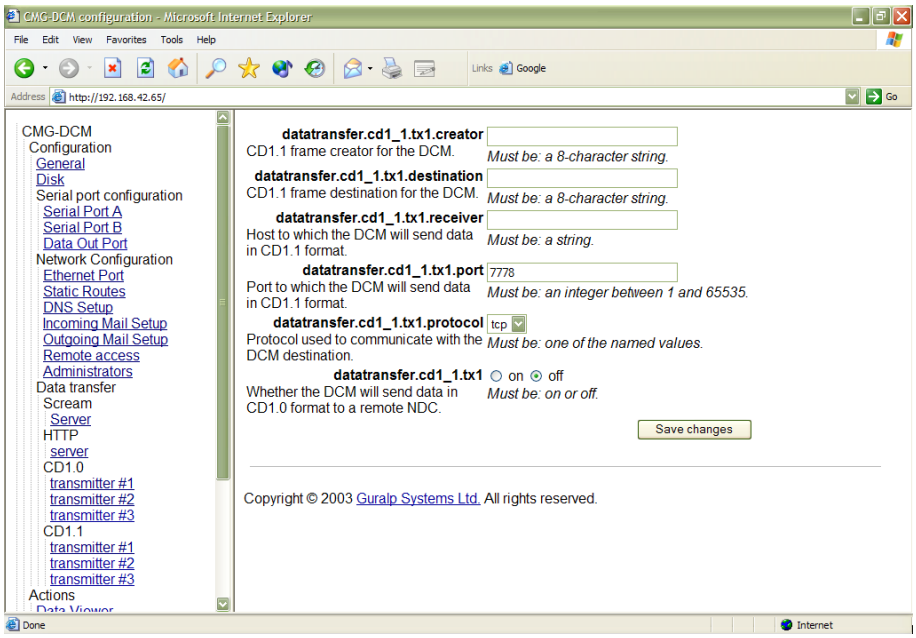
datatransfer.cd1.sender.port : The network port number to which the DCM will send CD1.0 data.

datatransfer.cd1.sender.compression : Select *on* to enable Canadian compression in the outgoing data stream.

datatransfer.cd1.sender.channel.x : The DCM needs to know the CD1.0 channel name for each of the streams you want to transmit. When data comes in on a stream that the DCM recognizes, it will send it on with this new name. You can configure up to 12 channels to be transmitted by one CD1.0 sender.

Channel names are normally chosen to conform to the FSDN SEED naming convention, but you can use different names if you wish. The CD1.0 sender does not use the current *Seed name mappings*.

5.5 CD1.1



The CD1.1 format is an evolution of CD1.0 used by many NDCs, using different settings from CD1.0. The DCM has a separate transmitter for this format.

datatransfer.cd1_1.tx : Select *on* to instruct the DCM to provide data in this format. If you do not want the DCM to transmit CD1.1 data, select *off* and ignore the remaining settings.

datatransfer.cd1_1.creator : An 8-character string specifying the source of the data (*i.e.* the instrument or array to which the DCM is attached.)

datatransfer.cd1_1.destination : An 8-character string specifying the intended destination of the data.

datatransfer.cd1_1.port : The network port number to which the DCM will send CD1.1 data.

datatransfer.cd1_1.protocol : The network protocol the DCM should use to send CD1.1 data—currently only *tcp* is supported.

datatransfer.cd1_1.receiver : The IP address or hostname of the remote CD1.1 device.

For the CD1.1 sender to be able to build frames, it needs to have access to detailed status information from the digitizer. This is provided in a special digitizer stream ending `CD`. This stream is not output by

default: to make the DM24 output a CD stream, connect to its console and issue the command

```
+MONITOR
```

This can be done from the DCM command line (see Section 8.2, page 117) with the command

```
gcli n '+monitor'
```

where *n* is the number of the serial port attached to the digitizer.

CD1.1 subframe configuration

CD1.1 data is organised into subframes, which have some flexibility in their format. A configuration file, `/etc/cd11sf.cfg`, is provided allowing you to specify the exact format to be used in outgoing CD1.1 subframes. Clicking **CD1.1 subframe configuration** brings up a page in the work area which enables you to edit the file and its attributes directly.

The Web interface does *not* check that the content of the files will be understood. You should ensure that the file is valid before committing any changes.

Each line in `/etc/cd11sf.cfg` describes a single CD1.1 stream in the format

data-stream:*status-stream*:*location*:*site*:*instrument*:*channel*:*prefix*:*key-bucket*

where the fields, separated by colons, are:

data-stream : the digitizer's data stream ID;

status-stream : the digitizer's status stream ID;

location : the CD1.1 location code for the array;

site : the CD1.1 site code for the instrument;

channel : the CD1.1 channel name;

prefix : a prefix for the location to store the files in the file tree of the DCM.

key-bucket : the key-bucket code, which tells the DCM which key to use for that particular stream.

The cryptographic hardware required to produce authenticated CD1.1 streams is installed only in Authentication Modules (AMs). If you do not have this hardware, you can transmit unauthenticated CD1.1 data by using a key-bucket code of `-1`.

prefix : A prefix used to determine where to place generated CD1.1 subframes. For example, a prefix of `/data/HPA1.` will produce files in the `/data/` directory beginning with `HPA1.` and followed by a unique timestamp. Several streams may produce files using the same prefix; indeed, this is recommended.

Blank lines, and any lines beginning with `#`, are ignored.

When you have finished editing the file, clicking *Save changes* will write the changes to disk. The changes will not take effect, however, until the CD1.1 service is restarted. You can restart all running CD1.1 services by clicking *Restart CD1.1*.

5.6 CNSN configuration (cnsn.cfg)

This option is only functional on Authentication Modules (AMs) with an on-board cryptographic token.

In this system, a signature is generated from incoming data, which is transmitted together with the compressed data. At the receiving end, the signature is regenerated and compared with the one sent to establish authenticity.

To use the CNSN system, you will need to set exactly one *serial.x.service* option to *cnsn_in*, and one to *cnsn_out* (see Section 6.2, page 82.) You will also need to edit the file `/etc/cnsn.cfg`, which defines the streams.

Each line in `/etc/cnsn.cfg` describes a single CNSN stream in the format

method:*key-id*:*high-byte*:*low-byte*:*site*:*channel*[:*location*]

where the fields, separated by colons, are:

method : The compression/signature method, one of `canadian_compression_after_signing`, `canadian_compression_before_signing`, `canadian_as`, `canadian_bs`, `steim_as`, `steim_bs`, or `none`

key-id : The ID of the key to use for the stream, as used in

`/etc/libcd11/slot`*`xx`*`.kid` and `/etc/cd11sf.cfg`

`high-byte` and *`low-byte`* : The two CNSN stream identifier bytes, expressed as decimal numbers.

`site` : the site code.

`channel` : the channel code.

`location` : optionally, the location code.

Blank lines, and any lines beginning with `#`, are ignored. When you have finished, click **Save changes**.

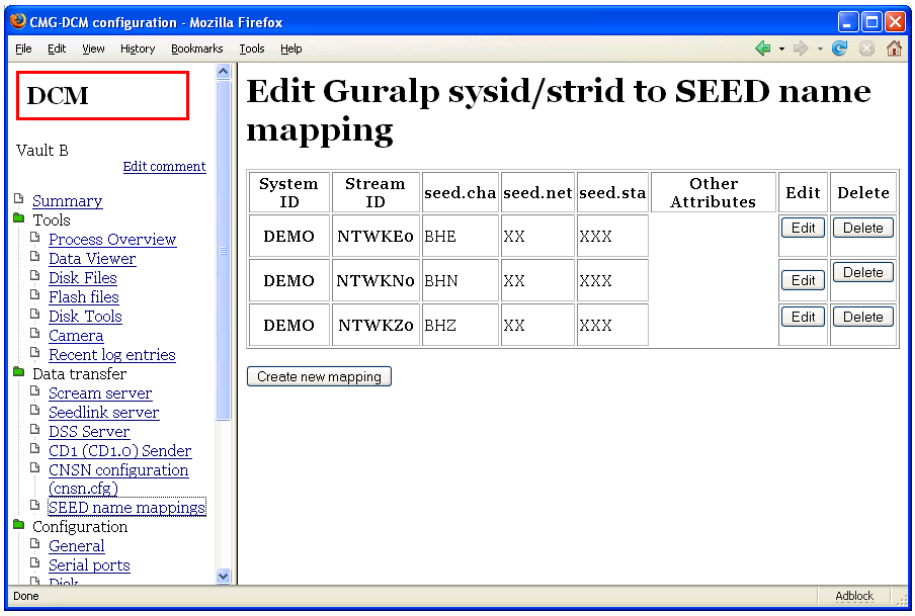
The Web interface does *not* check that the content of the file will be understood. You should ensure that the file is valid before committing any changes.

5.7 SEED name mappings

Several seismic network formats use the FSDN SEED naming convention to identify channels.

Before you can connect Güralp Systems instruments to a network using this convention, you will need to define mappings between the raw stream names from the digitizer (*e.g.* `DEMOZ2`) and the FSDN SEED names for the same streams (*e.g.* `GB:GSL:BHZ`).

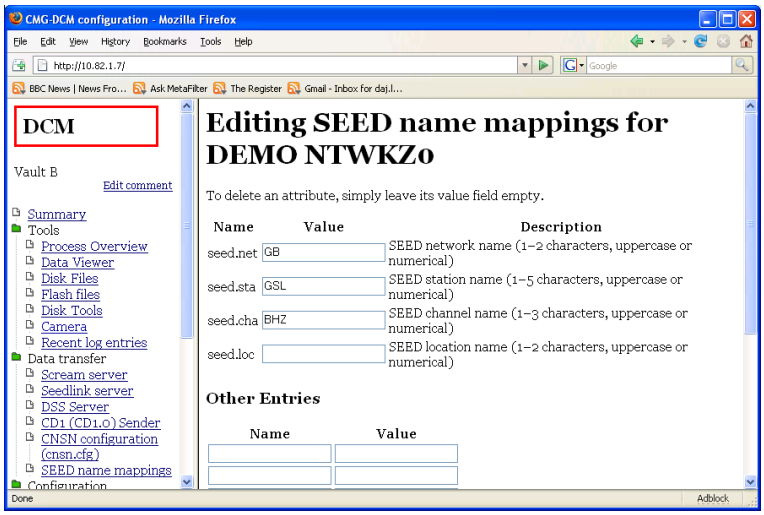
This mapping is defined in an XML file stored at `/etc/seedmap2.xml`. Clicking on **Data transfer – SEED name mappings** opens a page allowing you to view and edit the mapping.



The table on this page shows all the streams which have SEED name mappings. If a mapping is defined for a stream, services which use SEED names will automatically transmit that stream.

- the *System ID* and *Stream ID* for each stream which has a mapping;
- the 3-character SEED channel name in use for the stream;
- the 1- or 2-character SEED network name in use for the stream;
- the 3-character SEED station name in use for the stream;
- any other attributes you have defined for the stream and
- buttons allowing you to **Edit** and **Delete** the entry.

Clicking **Edit** brings up a page where you can edit these attributes.



- Enter values into the *seed.net*, *seed.sta* and *seed.cha* boxes to change the mapping. If one of these values is missing, services which use the *SEED name mappings* will ignore the stream.

- If you are using SEED location codes, enter a value into the *seed.loc* box to set the location code for this stream. Location codes should be used in cases where several instruments with the same conventional SEED name are located at the same station.

- Finally, you must enable output of this stream using each of the protocols you are using. This is done by adding an entry to the *Other Entries* section with the name `protocol.enabled` and the value `true`.

  For example, to make a stream available over SeedLink, enter `seedlink.enabled` under *Name*, and `true` under *Value.*

  To stop transmitting a stream, change the *Value* of this entry to `false`, or remove it.

- You can add other information about your instruments into this file. For example, you might want to program the DCM with response and sensitivity parameters for your instrument.

  To add additional information, enter the name of a quantity (*e.g.* sensitivity) in one of the *Name* boxes, and its value (*e.g.* 3000) in the *Value* box beside it.

  This information is not currently used by any DCM services.

Click **Save changes** to commit your changes, or **Cancel** to return to the

table without changing anything.

The *SEED name mappings* are used by MiniSEED, SeedLink and DSS. SEED names are also used by the full SEED recorder, but this requires additional data. You should supply this data in SEED's own configuration file `/etc/seed.cfg` (see above.)

CD1.0 and CD1.1 protocols use channel names which are very similar to SEED names. However, there may be instances where the CD1.0 and CD1.1 names need to differ from the FSDN SEED name, or where you do not want to transmit streams automatically. Because of this, the CD1.0 and CD1.1 services do not look for mappings in this file.

5.8 AutoDRM

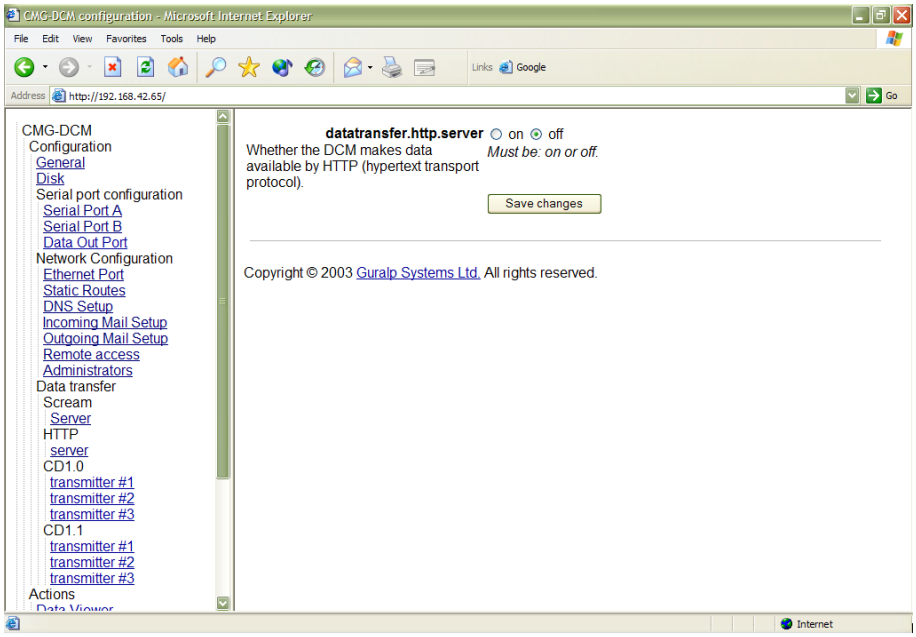
AutoDRM is an automated system for handling requests for seismic data over e-mail, and for fulfilling those requests either by returning e-mails or by establishing an FTP connnection. In order for the AutoDRM system to work, the module must be able to send and receive e-mail messages: this can be configured using the *Incoming* and *Outgoing mail setup* configuration pages. See “Incoming mail setup”, page 97, for more details.

The DCM's AutoDRM capabilities are compatible with the GSE2.0 and 2.1 message standards developed for GSE-EISMS and GSETT-3.

datatransfer.autodrm : Select *on* to enable the DCM to handle AutoDRM requests.

datatransfer.autodrm.maxpoolsize : The maximum pool size to use whilst processing AutoDRM requests, in megabytes.

5.9 HTTP server



In addition to configuring the DCM and instruments attached to it, you can also instruct the on-board Web server to provide data to network clients.

datatransfer.http.server : Select *on* if you wish to be able to retrieve data from the DCM over the Web (*i.e.* HTTP). Once the HTTP server is enabled, you can browse through the DCM's hard disk using the URL `http://mydcm/cgi-bin/explorefs?path=/`.

You will need to use your username and password to access this service.

6 Configuration

All DCM modules feature an on-board Web server, which you can use to set up the unit. When you first connect to the module, you will be shown a menu tree in a panel on the left, with a summary of its current status on the right.

The configuration options are accessed towards the bottom of the menu tree, under **Configuration**. There are several pages of configuration options.

Once you have changed settings on any page, click **Save changes** at the bottom of the page to make the DCM apply them. If you move to a different page without clicking **Save changes**, your changes will be lost.

This chapter describes all the settings available to users of the DCM. The name of each setting is shown on the Web page interface, and can also be used to access the setting using the `gcfgdbset` command (see Section 8.2, page 117.)

For each setting, you will have the option to

- enable or disable the setting (*on/off*, *yes/no*, etc);
- choose from a drop-down list of options; or
- enter your own information in a text box.

In each case, the Web interface describes the values the DCM will accept.

If you are using `gcfgdbset`, and you try to change a setting to an unsupported value, the program will report an error.

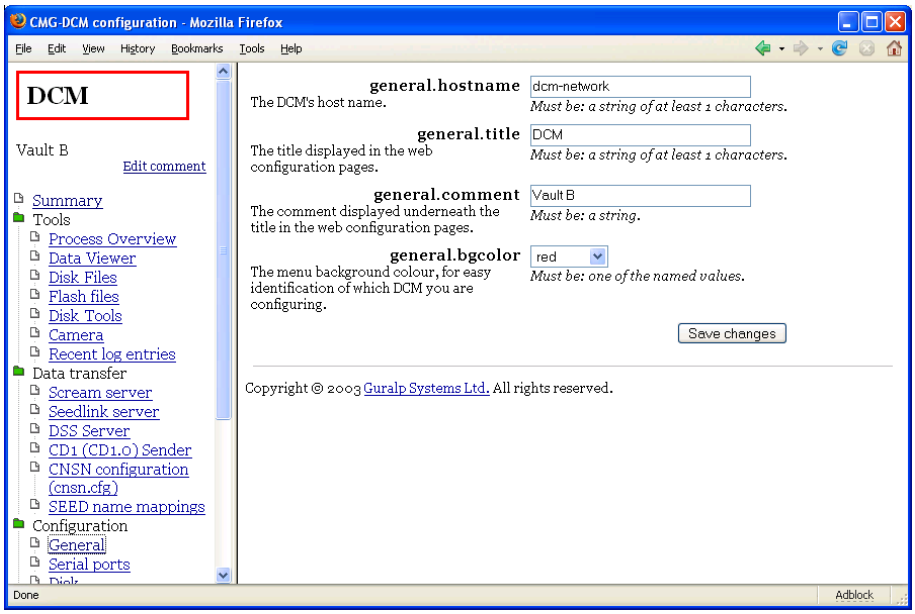
Note: The DCM is a flexible unit which can be supplied with a range of feature sets depending on your requirements. A stand-alone DCM unit normally has most of these options available to the user. However, some of the settings described below may not be relevant to your installation, and thus will not be available. If you are unsure which packages you will need, contact Güralp Systems.

A DCM may be supplied which supports multiple instances of a particular service (*e.g.* several CD1.1 transmitters.) In this case, each instance will have its own configuration page, which will appear as described below for that service. Users of installations which include

several types of DCM or AM (Authentication Module) will find that each unit offers a different range of configuration options, according to its specialization as a component in the system.

6.1 General

This page contains settings which help you identify the DCM.



general.hostname : This is the host name the DCM will use to identify itself on the network. It is normally the first part of the module's fully-qualified domain name. For example, a DCM at *myinst.bignet.org* would have its host name set to *myinst*.

general.title : This is the title displayed inside the coloured border at the top of the left-hand menu.

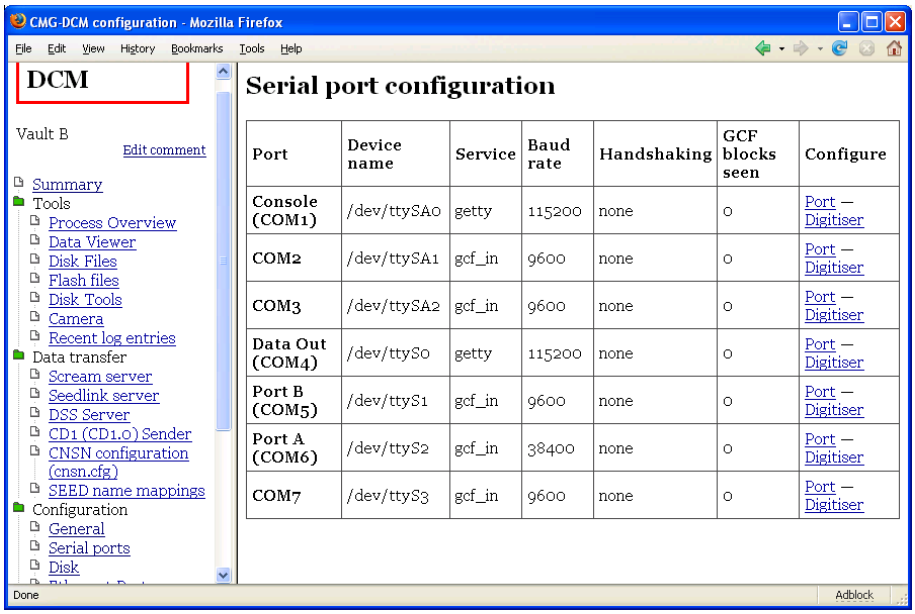
general.comment : This is an optional comment displayed beneath the title in the left-hand menu.

general.bgcolor : Select a value from the drop-down box to choose the colour of the title border. For example, you might choose to assign a particular colour to all DCMs at one location.

If you change *general.title*, *general.comment*, or *general.bgcolor*, the left-hand menu will not change in your browser when you **Save changes**. You will need to reload the page, or open it in a new browser window.

6.2 Serial ports

Clicking on **Configuration – Serial ports** displays the serial port table.



Serial port configuration

Port	Device name	Service	Baud rate	Handshaking	GCF blocks seen	Configure
Console (COM1)	/dev/ttySA0	getty	115200	none	0	Port — Digitiser
COM2	/dev/ttySA1	gcf_in	9600	none	0	Port — Digitiser
COM3	/dev/ttySA2	gcf_in	9600	none	0	Port — Digitiser
Data Out (COM4)	/dev/ttyS0	getty	115200	none	0	Port — Digitiser
Port B (COM5)	/dev/ttyS1	gcf_in	9600	none	0	Port — Digitiser
Port A (COM6)	/dev/ttyS2	gcf_in	38400	none	0	Port — Digitiser
COM7	/dev/ttyS3	gcf_in	9600	none	0	Port — Digitiser

Every serial port on the DCM's internal board is listed in this table.

Port : This column shows the DCM's name for each port. A stand-alone DCM has three RS232 serial ports brought out on 10-pin mil-spec connectors, labelled *PORT A, PORT B* and *DATA OUT*.

The labelling is provided for convenience; you can change it if it does not match your site requirements. Each port may be used for communication in either or both directions.

Note that the standard DCM does not expose the *Console (COM1)*, *COM2* or *COM3* ports. The external ports begin at *COM4.*

Device name : This column gives the Linux device name for each port. You may need this if you want to write your own scripts.

Service : The service currently running on the port (by default, *getty*.)

Baud rate : The current baud rate of the port (by default, 115200.)

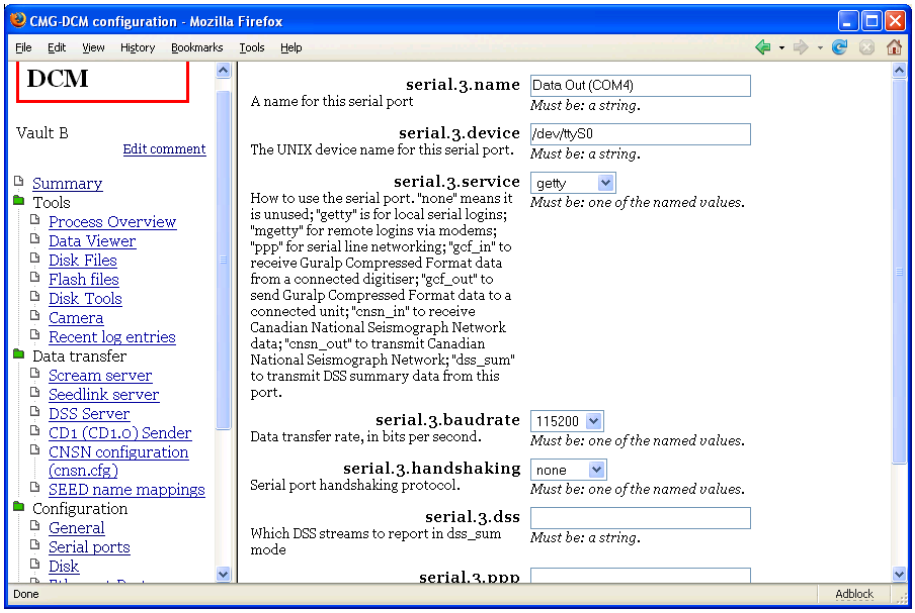
Handshaking : The handshaking protocol currently being used on the port, if any (by default, *none*).

GCF blocks seen : The number of GCF blocks which have been received on this port. This will only be non-zero for ports set to *gcf_in*.

Configure : This column contains links allowing you to configure each port, and the digitizer attached to it (if there is one.)

Configure – Port

When you click on a *Configure* – **Port** link, the port's configuration page is displayed.



Each serial port has its own configuration screen, with a number of options beginning *serial.x* (where *x* is the internal number of the serial port). The configuration options for all the ports are as follows:

serial.x.name : The name of this serial port, as shown in the serial port table.

serial.x.device : The pathname of the Linux device corresponding to this serial port. You should not need to change this option.

serial.x.service : How the DCM will use each serial port. There are five options:

- *getty* : The port will listen for logins from a connected computer. Once logged in, the user will have direct command-line access to the DCM's Linux operating system.

- *mgetty* : The port will listen for logins over a connected serial modem. Again, a logged-in user will be presented with the DCM's Linux command line.

- *mgetty-r* : The port will listen for logins over a connected serial

modem which is operating in raw mode (*i.e.* which does not support the standard *AT* modem commands.) This option in runs the Linux command `mgetty -r`. A port running the *mgetty-r* service will also listen for incoming PPP frames, and if it detects one will configure the PPP link according to the `/etc/mgetty+sendfax/login.config` file (see Section 6.11, page 100.)

- *ppp* : The port will provide a PPP link to a connected computer. Once set up on the computer, the DCM will appear as if on a local TCP/IP network. You can then log in to it using *ssh* or use its Web-based administration system, as well as receiving data.

  Before a PPP link will function, you will need to specify the PPP options and (if using CHAP) provide a `chap-secrets` file. This can be done through the DCM configuration interface: see see Section 6.11, page 100. Also see the Linux manual page for `pppd(8)` for an explanation of how PPP is implemented.

- *gcf_in* : The port will receive data from a GCF-compatible digitizer.

- *gcf_out* : The port will send GCF data directly to the connected device as it is received. A number of programs are available for PCs which handle incoming GCF data streams, such as Scream!, Antelope, and Earthworm. The DCM transparently merges all incoming data streams, so if you are using Scream!, you can also configure and control attached digitizers using its own interface.

- *cnsn_in* : The module will use this port to receive raw data using the CNSN protocols. If you use this service, there must be exactly one *cnsn_in* and one *cnsn_out* port.

- *cnsn_out* : The module will this port to transmit authenticated data using the CNSN protocols. If you use this service, there must be exactly one *cnsn_in* and one *cnsn_out* port.

- *dss_sum* : The port will transmit DSS summary data. Which data is transmitted depends on the value of *serial.x.dss*, below.

serial.x.baudrate : This option alters the speed of communication across each serial link, in bits per second.

For ports connected to digitizers, you should ensure that the baud rate is high enough to allow all the data to be transmitted at the rates you have chosen. As an example, for three streams transmitting at 100 Hz, a rate of 9600 baud is usually sufficient. Modern modems can

normally operate at rates up to 57600 baud (~56 kbits/s), although the telephone or transmission lines may not support such a high rate. The same is true of radio telemetry links.

The DCM's serial ports operate using frames of 8 data bits, no parity bits, and one stop bit.

serial.x.handshaking : The flow-control (handshaking) protocol used across each serial link. There are three options:

- *off* : Transmit data across the serial link without handshaking, *i.e.* assume that the link is always ready to send or receive data.
- *rts/cts* : Use the Ready To Send/Clear To Send handshaking method, where two separate lines within the serial cable are used to control the flow of data. This is the most reliable method since accomplished in hardware, but is not feasible for long-distance or complex connections.
- *xon/xoff* : Send the two special characters `^Q` (17) and `^S` (19) as part of the data stream to request that data transfer be started and stopped. This method requires only a single connection, but assumes that the special characters will be received correctly every time.

serial.x.dss : If you have set the **serial.x.service** option to *dss_sum*, this option lists the DSS data types that will be transmitted. If the port is not running the *dss_sum* service, this setting will be ignored.

serial.x.ppp : If you have set the **serial.x.service** option to *ppp*, the DCM will use the standard Linux command `pppd` to manage the PPP service for this port. The **serial.x.ppp** option allows you to add command-line options to `pppd` which are specific to this port. The command-line options are fully described in the Linux documentation for `pppd`.

If the port is not running the *ppp* service, this setting will be ignored.

Any options which apply to all serial ports running the *ppp* service should be placed in the `/etc/ppp/options` file instead (see Section 6.11, page 100.

serial.x.ack_nak_timeout : If you have set the **serial.x.service** option to *gcf_out*, this option sets the length of time, in milliseconds, that the DCM will wait for a response to each GCF block sent from this port, before giving up and sending the next one.

This option has the same effect as the `MS-GAP` command on a DM24.

If the port is not running the *gcf_out* service, this setting will be ignored.

Configure – Digitizer

Clicking on *Configure* – **Digitizer** opens a page allowing you to reconfigure the digitizer attached to the port, if there is one.

When the page loads, the DCM attempts to retrieve the current configuration of the digitizer attached to that port. This will take a few seconds, after which the message `Configuration successfully retrieved from attached instrument` should appear together with a form detailing the available settings.

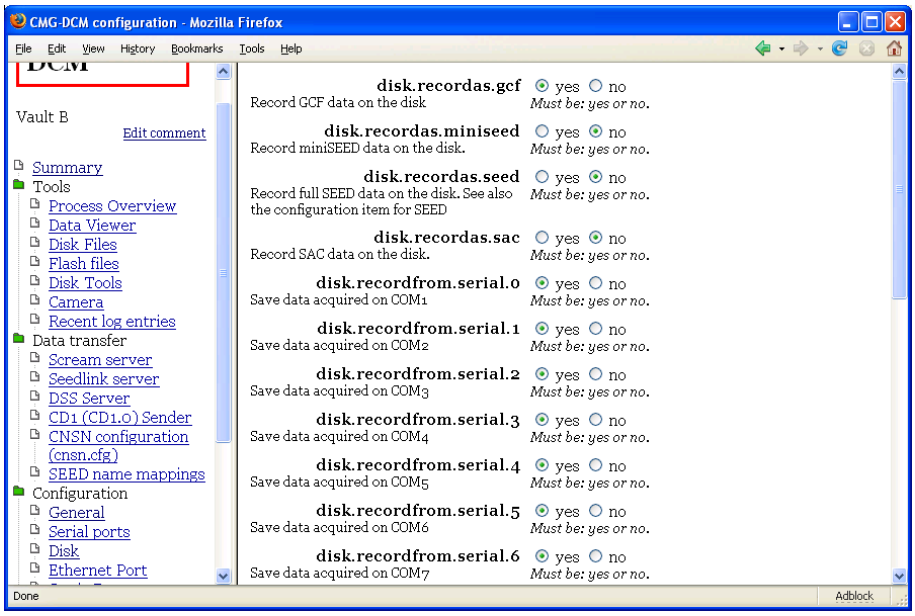
If the DCM cannot find a digitizer attached to the port, it will give the message `Unable to retrieve configuration from attached instrument`. In this case you should

- check that an instrument is connected to the port you clicked, and that it is powered up;
- check that the baud rate of the serial port (see Section 6.2, page 82) matches the output baud rate of the digitizer; and
- check that the DCM is running the *gcf_in* service on the port.

For a full description of the digitizer settings you can change, see Chapter 7, page 105.

6.3 Disk

On this page you can set how the DCM uses its hard disk, and what data it records.



If you want to check that the hard disk is working, explore its contents, or format it, you should use the **Tools – Disk tools** page instead.

At the top of the window are two sets of options, **disk.recordas...** and **disk.recordfrom...**.

Data recording formats: disk.recordas

The **disk.recordas** section tells the DCM which formats to record data on the disk.You can record simultaneously from any number of ports and in any number of formats. If you record in several formats simultaneously, you should make sure that you have the capacity to store all the data the DCM will produce, or the bandwidth to transfer it.

For example, if you set all the **disk.recordfrom** options to *yes,* **disk.recordas.gcf** to *yes*, and the other formats to *no*, the DCM will record GCF data from all serial ports on the disk. This is the default behaviour.

Files are not immediately written to the disk, but into Flash memory; the options **disk.usagemode** and **disk.writeinterval**, below, control how often the DCM flushes new data to the disk, and what it will do if the disk becomes full.

disk.recordas.gcf : Set this to *yes* to have the DCM record Güralp Compressed Format data files. We recommend that you leave this option set to *yes*, because Güralp Systems digitizers output natively in this format. If you record in a different format, and do not keep the original GCF data, information will be lost.

disk.recordas.miniseed : Set this to *yes* to have the DCM record files in miniSEED format.

MiniSEED uses the FSDN SEED channel naming convention. Before enabling this option, you will need to tell the DCM which data streams to output, and what SEED channel codes to use for each one. This is done on the **Data transfer – SEED name mappings** page: see Section 5.7, page 75.

disk.recordas.seed : Set this to *yes* to have the DCM record full SEED volumes on the hard disk.

This option will have no effect until you create a SEED recorder configuration file. See Section 6.4, page 91, for information on these options.

disk.recordas.sac : Set this to *yes* to have the DCM record files in SAC format.

disk.recordas.ascii : Set this to *yes* to have the DCM record files in ASCII format. *Enabling this option produces very large files, and is not supported in the default distribution.* If you need ASCII output, you must install the **asciirecorder** package from the Linux command line, using the command

```
ipkg install asciirecorder
```

The ASCII files have a simple format. After a header line, each line contains two decimal numbers separated by a tab.

The first number is the UNIX time of the sample (*i.e.* the number of seconds since 00:00:00 UTC on January 1, 1970). The second is the absolute sample value at that time. Samples are not guaranteed to be present in time order.

Data sources: disk.recordfrom

Each serial port has an option in the **disk.recordfrom** section which enables recording from that port. If you set the **disk.record.from** option for a port to *yes,* the DCM will record all the data coming from the port on the disk.

disk.recordfrom.serial.*n* : Each of these options refers to a serial port on the DCM by its number. Select *yes* to have the DCM record data coming into this port.

disk.recordfrom.log : Set this to *yes* to make the DCM record its own

system log onto the disk. The system log is split up into time blocks in the same way as the other data (see **disk.recordinterval**, below.)

Status blocks from digitizers are part of the GCF data, not part of the log. To record these, you need to enable **disk.recordas.gcf**. If you also enable **disk.recordsplit** (see below), each status stream will be recorded in its own file.

disk.recordfrom.coalesce : The DM24 digitizer can be configured to produce GCF blocks more quickly than normal. This may be done, for example, to reduce the latency in data transmission. Blocks produced this way may not all be full, so space will be wasted. You may also experience problems with recording bandwidth.

Enabling this option makes the DCM build full GCF blocks from the data it receives (a process known as *re-blocking*) before saving it to disk. You should only need to enable this option if you are using the low latency features of the DM24 digitizer. It is disabled by default.

Other disk options

disk.power : To save power, the on-board hard disk can be set to power up only when required. If power consumption is an issue and you expect the hard disk to be used only occasionally, you should set this option to *automatic*. Otherwise, setting it to *always-on* will ensure that the hard disk is always ready to receive data. If you intend to operate an internal disk continuously, you should take care not to let it overheat, since the interior of the DCM is well insulated.

disk.usagemode : This setting varies the way the DCM uses its hard disk storage once it has been filled up.

- The *use-once* option causes the DCM to stop recording data to the hard disk, and discard any new data that arrives. It may still send new data over the network, if you have so configured it, but the hard disk will be untouched once full. Thus the *start* point of the stored data is known, but the end point depends on the disk capacity. This is most useful for installations where you are expecting to replace the hard disk periodically, or where you need to take time-synchronized readings at several sites.

- The *recycle* option causes the DCM to delete the oldest data files on the hard disk to make space for incoming data. This way, the *end* point of the stored data is known (it always includes the most recent block of data), but the start point depends on the disk capacity.

disk.recordinterval : The DCM writes all the data streams it has been instructed to record into a file in its Flash memory. With the default setting active, the system starts a new file every 3 hours, alternating between the two Flash memory banks (see Section 8.1, page 116.) The 3-hour interval is called a *watch*.

For those formats which can only deal with single streams, the DCM opens a file for each stream, starting a new set of files every watch.

You can make watches shorter by setting *disk.recordinterval*, which is expressed in seconds. The options are

- 10800 seconds, or 3 hours;
- 3600 seconds, or 1 hour;
- 1800 seconds, or 30 minutes.

The recording interval needs to be chosen with care, bearing in mind how you will use the DCM.

- If files are larger than 25% of the Flash memory capacity, the DCM may encounter problems, because the disk manager only starts transferring files when the Flash memory is over 75% full.

  If a large file fills up the Flash memory before it is finished, the DCM's watchdog process may force a reboot.

- In any case, files recorded by the DCM must never exceed 16 Mb owing to internal limitations.

- Each recording process keeps files open in Flash memory whilst it is writing to them. For MiniSEED and SAC files, and also for GCF files if the **disk.recordsplit** option is active (see below), a file is opened for every stream the DCM encounters.

  The Linux operating system enforces a limit of 1024 files which can be open at once.

  You will only approach this limit if you are running several recording processes on a large number of streams. If you do, you may find that decreasing **disk.recordinterval** helps.

- The Flash filesystem can store up to 262144 files. It is unlikely you will approach this limit under normal conditions.

Files are given names in the format

*yyyyjjj*T*hhmmss*-*wwwww*-*WWWWW*-*ssssssss*.*eee*

where *yyyy* is the current year, *jjj* is the day of the year, *hhmmss* is the start time of the watch, *wwwww* is the watch number, *WWWWW* is the number of watches, *ssssssss* is an 8-digit serial number (for uniqueness), and *eee* is an appropriate file extension for the format used.

For file types which do not support multiple streams, and also for GCF files if *disk.recordsplit* is enabled (see below), the *Stream ID* *SSSSS* is also included. This produces names in the format

*yyyyjjj*T*hhmmss*-*wwwww*-*WWWWW*-*ssssssss*-*SSSSS*.*eee*

disk.recordsplit : By default, GCF files saved to Flash memory contain all the GCF blocks which have been received on the active ports.

Setting *disk.recordsplit* to *yes* instructs the DCM to look through the incoming GCF data and save a separate file for each incoming stream.

If you enable *disk.recordsplit*, you should make sure the DCM does not produce too many files: see above for details.

disk.heater : A heater is provided in the internal enclosure of the DCM to bring the on-board hard disk up to operating temperature when required. If you are using an on-board hard disk, you should set this option to *automatic* and choose a suitable operating temperature below. If you are not using an internal hard disk, you should disable the heater by setting this option to *off*.

disk.temperature : The minimum operating temperature of the on-board hard disk, in degrees Celsius. If you have set the internal heating to *automatic*, you should enter a suitable operating temperature here. The hard disk will not function properly below 0°C. With the heater active, the DCM is designed to operate in ambient temperatures up to 40 degrees below zero.

6.4 SEED recorder

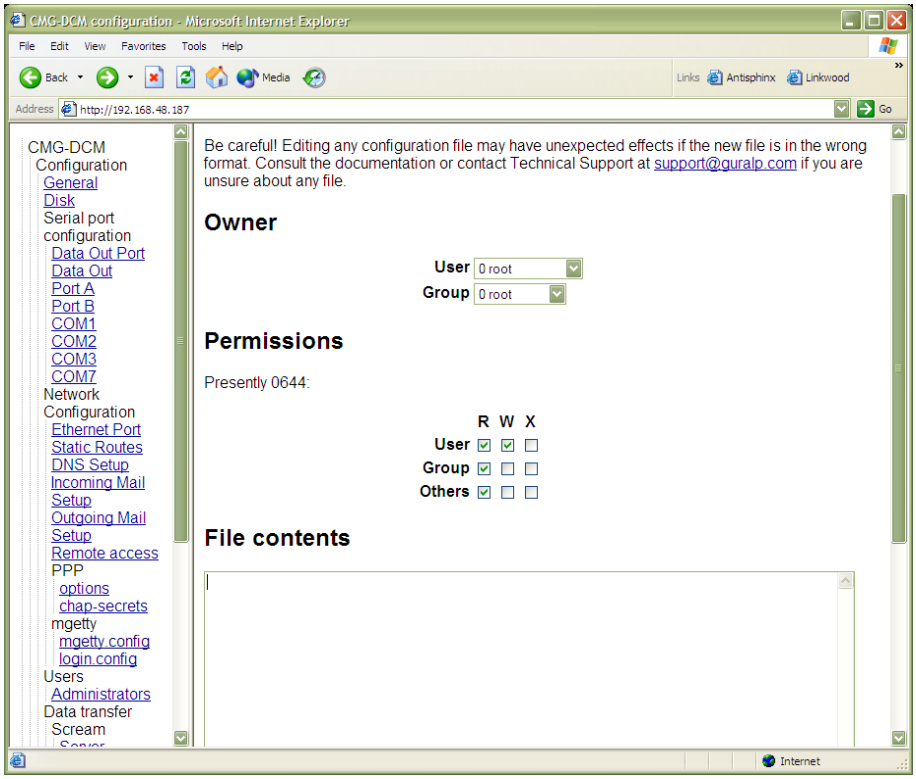
The DCM can compile full SEED volumes in real time. To enable this, select *yes* for the *disk.recordas.seed* option under **Configuration – Disk** (see above.)

A configuration file, `/etc/seed.cfg`, is provided allowing you to provide technical information about the streams to be recorded in

SEED format. The DCM will only write SEED volumes for streams mentioned in this file.

Clicking **SEED configuration** brings up a page in the work area which enables you to edit the file and its attributes directly.

The Web interface does *not* check that the content of the files will be understood. You should ensure that the file is valid before committing any changes.



The format of the file is as follows:

```
HPA1Z2:::::CMG-3_120S_50HZ:CMG-DM24mk3:V9.800E-01:1.6E-07:46
HPA1N2:::::CMG-3_120S_50HZ:CMG-DM24mk3:V9.400E-1:1.6E-07:46
HPA1E2:::::CMG-3_120S_50HZ:CMG-DM24mk3:V1.050E00:1.6E-07:46
```

Each line represents an incoming stream from the digitizer; the fields are separated by colons, and are in turn:

- the digitizer's stream ID;
- a SEED network name for the array (or leave blank to have the DCM generate one automatically);
- a SEED station name for the station (or leave blank to have the DCM generate one automatically);

- a SEED location name for the instrument (or leave blank to have the DCM generate one automatically);

- a SEED channel name for the stream (or leave blank to have the DCM generate one automatically);

- a case-sensitive code for the sensor type (see Section 10.1, page 130);

- a case-sensitive code for the digitizer type (see Section 10.2, page 131)

- the sensitivity of the sensor, prefixed with a `V` for a sensitivity in V/m/s, or `A` for a sensitivity in V/m/s^2 (in the example, around 1 V/m/s).

  Note that the unit follows directly after the sensitivity, with no colon.

  The sensitivity is given on the sensor's calibration sheet. The sensitivity of a sensor integrated with a digitizer is quoted as a *single-ended* sensitivity, whilst stand-alone digitizers are provided with their *differential* sensitivity. A differential sensitivity is quoted as, *e.g.* 2 × 3000 V/m/s. In this file, you should use the doubled value, *i.e.* 6000 V/m/s, for a stand-alone digitizer. For integrated digitizers, you should use the single-ended value quoted.

- the sensitivity of the digitizer, in volts per count (here 1.6×10^{-7} V/count = 0.16 μV/count). This is given on the digitizer's calibration sheet.

- the decimation sequence code. This is an integer between 0 and 239 which describes the sequence of filters in use by the digitizer. If you have a recent digitizer (with firmware newer than version 0.91) this information is included in the incoming GCF streams, so you can leave this field blank. The codes corresponding to each possible decimation sequence are available from the support section of the Güralp Systems Web-site.

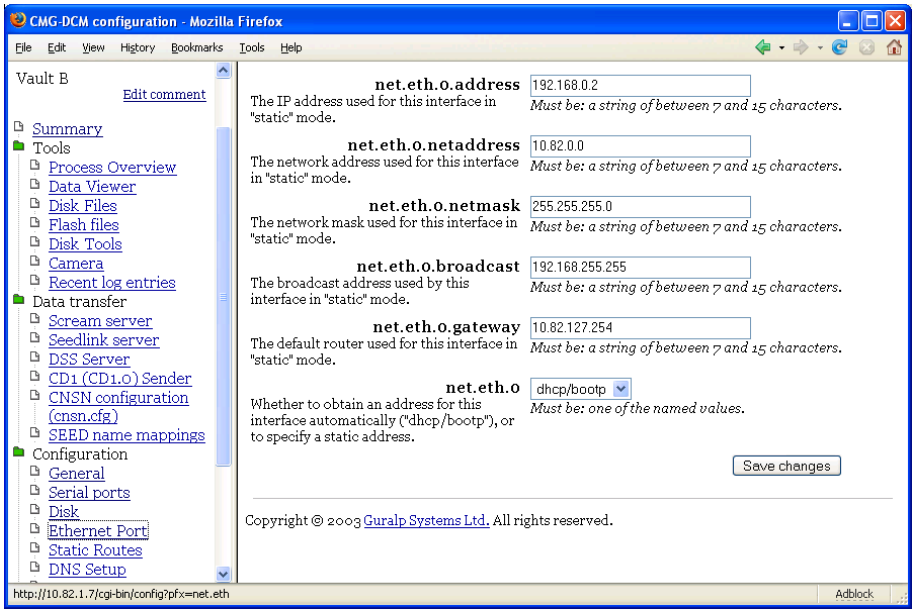
When you have finished editing the file, clicking **Save changes** will write the changes to disk. The changes will not take effect, however, until the SEED service is restarted.

To restart the SEED recorder, browse to the **Tools – Process Overview**

page and find the entry for the `seedrecorder` process. Click the **Restart** link in this entry.

6.5 Ethernet port

The **Configuration – Ethernet Port** page contains the configuration settings for the DCM's network interface.



The settings described below begin *net.eth.0* because they refer to the first Ethernet interface. If there are several Ethernet interfaces on your DCM, there will be an **Ethernet Port** page for each one.

net.eth.0 : If you know the IP address the DCM is going to use over this interface, you should set this option to *static* and fill in the details below. Alternatively, you can select *dhcp/bootp* and have the DCM automatically request an address over the network. If you choose the *dhcp/bootp* option, you should ensure that a gateway computer exists on your network that can receive these requests, and that the DNS (nameserver) entry for the DCM is kept current. You can then ignore the remaining settings on this page.

net.eth.0.address : If the above option is set to *static*, you should fill in the IP address of the interface in this box. Valid IP addresses consist of four numbers between 0 and 255, separated by periods. If you intend to attach the DCM to a private network, you should use an IP address in one of the private ranges (10.*b.c.d*; 172.16-31.*c.d*; 192.168.*c.d*) to avoid clashing with addresses on the wider Internet.

If the *net.eth.0* option is set to *dhcp/bootp*, this configuration setting

has no effect.

If you are configuring the DCM over an SSH connection, changing any of the remaining options will cause you to lose contact with it. You will have to log in to the DCM again (using the same IP address) to continue configuring.

net.eth.0.broadcast : The broadcast IP address used by the interface. This can often be derived from the IP address by replacing numbers at the end with 255.

net.eth.0.gateway : The IP address of the gateway machine (router) on your network.

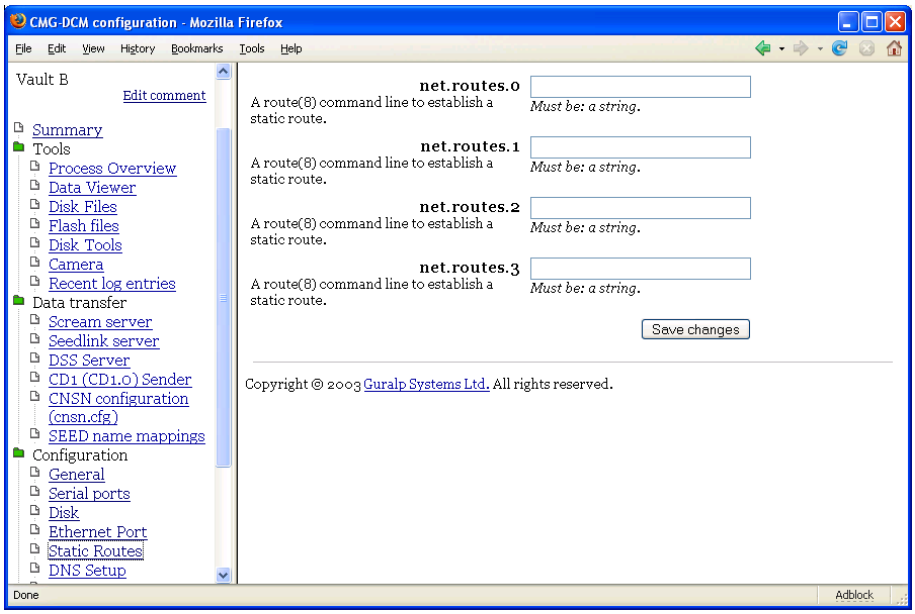
net.eth.0.netmask : The network mask to use for the IP interface. A DCM with an IP address in the domain 192.168.1.*x* would normally use a netmask of 255.255.255.0.

net.eth.0.netaddress : The network address of the DCM. A DCM with an IP address in the domain 192.168.1.*x* would normally use a network address of 192.168.1.0.

6.6 Static routes

In addition to the standard IP routeing mechanisms, you can define up to 4 additional static routes from the DCM to other parts of the network.

These routes are set on the **Configuration – Static Routes** page.



The routes are specified as arguments to the standard Unix `route` command. For example, if you wanted the DCM to route traffic to and from IP addresses beginning `192.168.0.` over the `eth0` interface, you would include the line

```
add -net 192.168.0.0 netmask 255.255.255.0 dev eth0
```

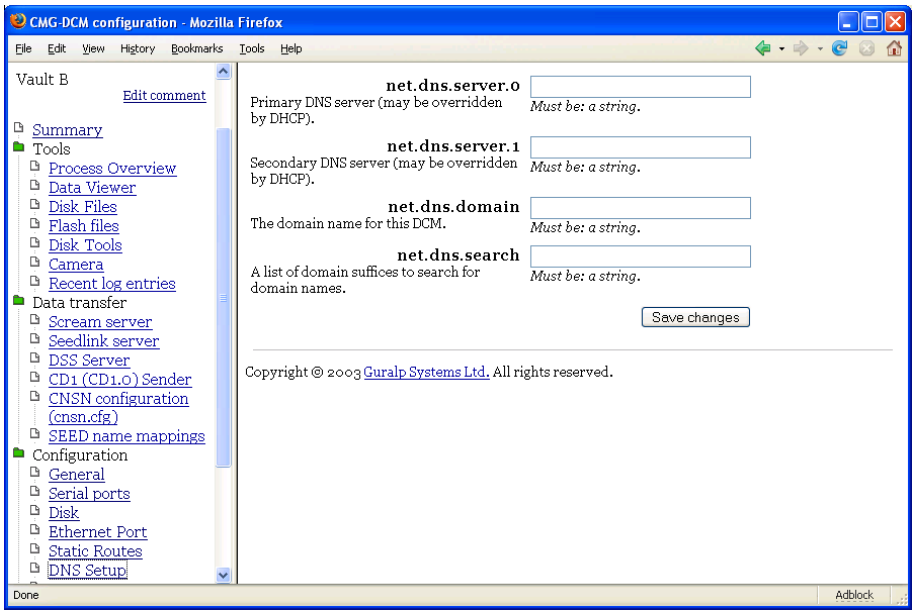
in the list.

For more information on the `route` command, please see its Linux manual page.

6.7 DNS setup

The settings on the **Configuration – DNS setup** refer to the Domain Name Service (DNS) used by the DCM, which translates numeric IP addresses into names.

It is not essential that you set up DNS for the DCM to work, or to update the software.



net.dns.domain : The name of the domain containing the module. For example, a DCM at *myinst.bignet.org* would have its domain set to *bignet.org*.

The rest of the settings on this page determine how the DCM determines the addresses of other hosts on the network from their host names.

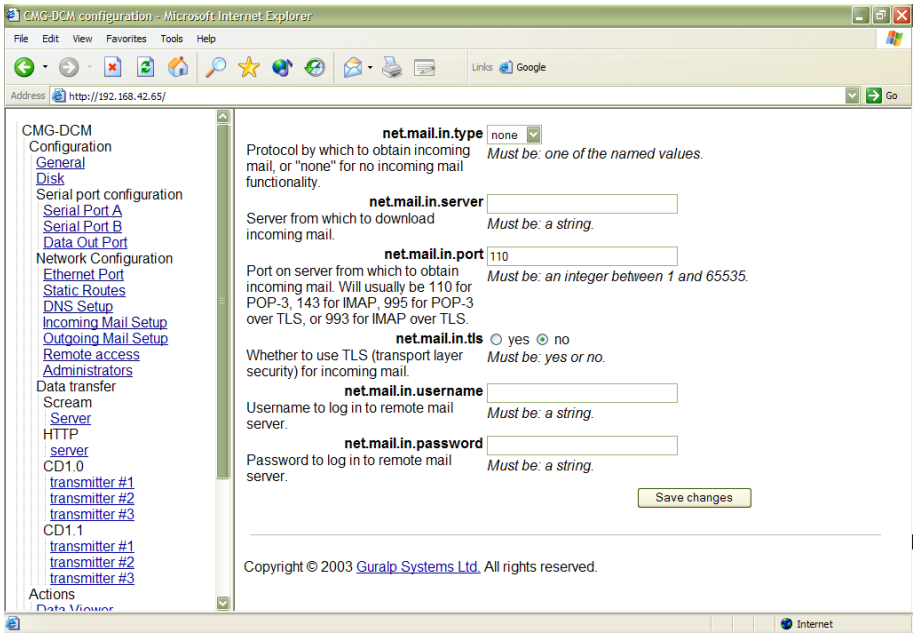
net.dns.search : If the DCM encounters an unqualified hostname in any of its configuration settings, or in a command, it will look for that name in each of the domains specified here in turn. You should give the domains to search in their correct order, separated by commas. The DCM's own domain (as given in the *net.dns.domain* setting) will always be searched first, so you need not specify it here.

net.dns.server.0 : The IP address of the primary nameserver used to resolve host names. If you are using *dhcp/bootp*, the DNS service is negotiated automatically, and so this setting may be ignored.

net.dns.server.1 : The IP address of the secondary nameserver. Again, you may be able to ignore this setting if you are using *dhcp/bootp*.

6.8 Incoming mail setup

This page enables you to set up the DCM to retrieve e-mail from a POP3 or IMAP server on the network, either unencrypted or secured using TLS (transport-layer security.)



If you do not need the DCM to be able to fetch mail, select *None* from the *net.mail.in.type* drop-down menu and ignore the remaining settings.

Not all DCMs have mail capabilities installed.

net.mail.in.password : The password needed to log in to the mail server.

net.mail.in.port : The number of the port to connect to on the mail server. This generally depends on the protocol used to retrieve mail:

- For *pop3* mail, this port will normally be 110, or 995 for secure POP3 over TLS (transport-layer security).
- For *imap* mail, this port will normally be 143, or 993 for secure IMAP over TLS.

net.mail.in.server : The hostname or IP address of the mail server.

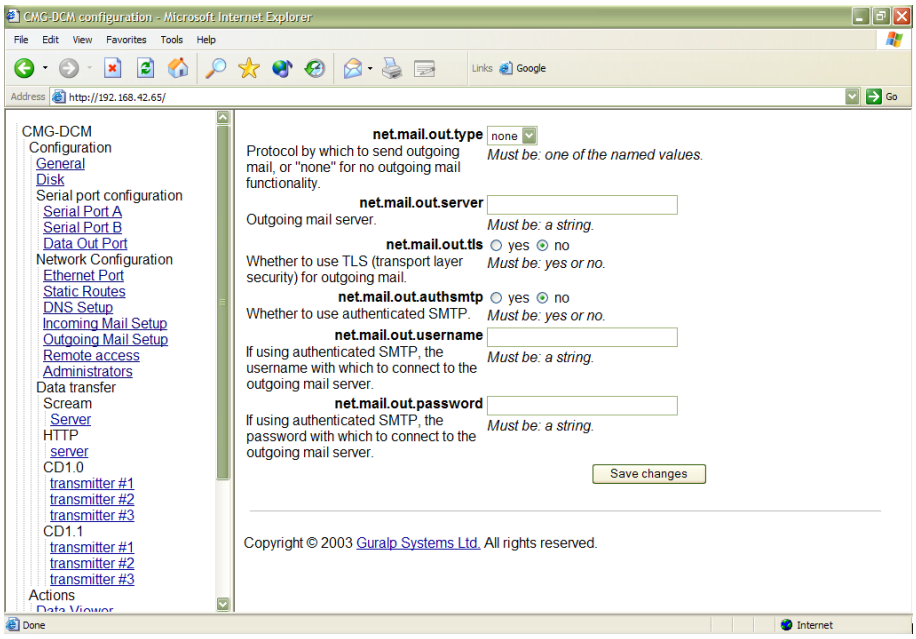
net.mail.in.tls : Select *yes* to encrypt the incoming mail channel with TLS, *no* to leave communications unencrypted.

net.mail.in.type : Select *pop-3* or *imap* to use that protocol to retrieve mail from the server. If you do not need the DCM to be able to receive mail, select *None*.

net.mail.in.username : The username needed to log in to the mail server.

6.9 Outgoing mail setup

This page enables you to send mail from the DCM using the SMTP protocol. Both authenticated (*i.e.* sending a username to the server) and unauthenticated SMTP are supported, as is encryption using TLS (transport-layer security.)



If you do not need the DCM to be able to send mail, select *None* from

the *net.mail.out.type* drop-down menu and ignore the remaining settings.

Not all DCMs have mail capabilities installed.

net.mail.out.authsmtp : Select *yes* and give a username and password in the fields below to use authenticated SMTP for outgoing mail.

net.mail.out.password : The password needed to log in to the outgoing mail server, if using authenticated SMTP.

net.mail.out.server : The hostname or IP address of the outgoing mail server.

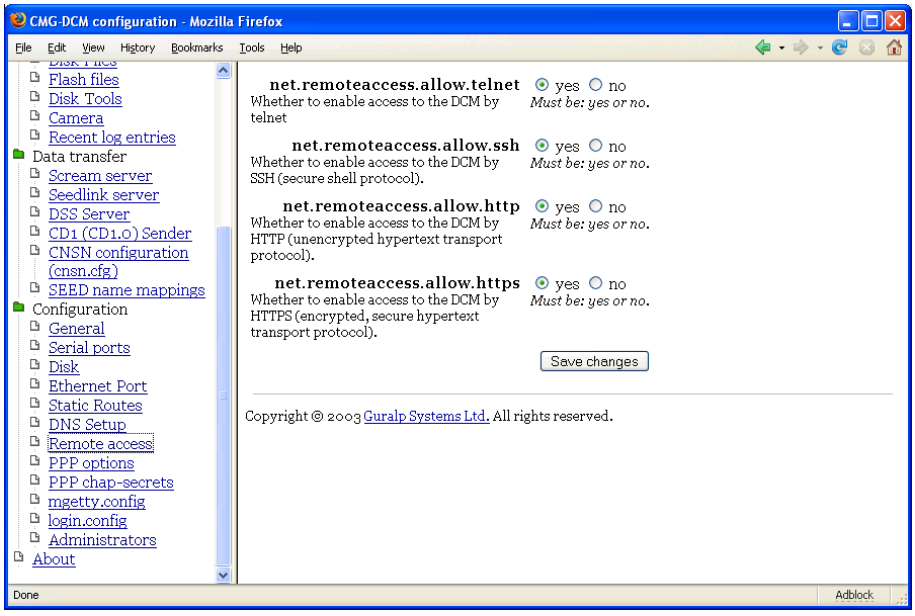
net.mail.out.tls : Select *yes* to encrypt the outgoing mail channel with TLS, *no* to leave communications unencrypted.

net.mail.out.type : Select *smtp* to use that protocol to send mail to the server. Currently only SMTP is supported. If you do not need the DCM to be able to send mail, select *None*.

net.mail.out.username : The username needed to log in to the outgoing mail server, if using authenticated SMTP.

6.10 Remote access

The settings on this page determines which ways you can use to access the DCM over a TCP/IP network.



If all these methods are disabled, you may still be able to connect to the DCM over a serial link using *getty* (see *serial.x.service*, above), or over a direct modem connection. Your settings will come into effect as soon as you click *Save settings,* so you should ensure that you have a backup means of communication with the DCM before disabling these methods.

net.remoteaccess.allow.telnet : Select *yes* to allow telnet clients to log in to the DCM's Linux operating system, or *no* to disallow connections over telnet. Select *no* to disable it. By default, this method is disabled for security reasons.

net.remoteaccess.allow.ssh : Select *yes* to allow SSH clients to log in to the DCM's Linux operating system, or *no* to disallow connections over SSH. By default, this method is enabled.

net.remoteaccess.allow.http : Select *yes* to allow access to the Web-based administration system by unencrypted HTTP. (A username and password will still be required, but they are sent over the network insecurely.) Select *no* to disable it. By default, this method is disabled for security reasons.

net.remoteaccess.allow.https : Select *yes* to allow access to the Web-based administration system by HTTPS (secure HTTP), or *no* to disable it. By default, this method is enabled.

6.11 PPP

These pages allow you to alter the PPP configuration files, `/etc/ppp/options` and `/etc/ppp/chap-secrets`.

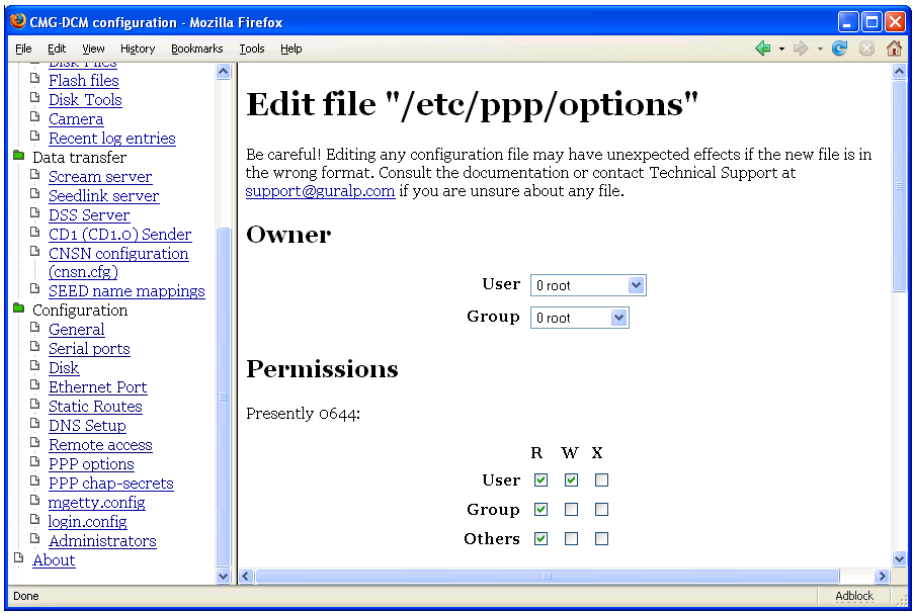
These two files are required by the Linux PPP server, and affect all *ppp* services on the DCM. Separate PPP options for each port may be specified under *Serial port configuration*; see above.

The `/etc/ppp/chap-secrets` file is only necessary if you are using CHAP to authenticate PPP connections.

Clicking on either of these entries brings up a page in the work area which enables you to edit the file and its attributes directly, including

- the ownership of the file;
- the permissions (read, write, or execute) of the file for its own user, its own group, and all others; and

- the content of the file.



The Web interface does *not* check that the content of the files will be understood. You should ensure that the file is valid before committing any changes, referring to the Linux manual page for `pppd(8)` if you are unsure.

When you have finished editing a file, clicking **Save changes** will write the changes to disk. The changes will not take effect, however, until the PPP services are restarted. You can restart all running PPP services by clicking *Restart PPP*; whilst this occurs, the network will be briefly unavailable.

options

The file `/etc/ppp/options` sets the default options for the PPP daemon. These will be applied to all PPP connections and to any of the serial ports which are running the *ppp* service. The file is treated as a list of words, each either an option or an argument to a previous option. For a list of the available options, and a full explanation of the format of the `/etc/ppp/options` file, see the Linux manual page for `pppd(8)`.

You can specify additional options for each serial port separately using the *serial.x.ppp* configuration option, which can be found on the *Serial port configuration* page for that port.

chap-secrets

The file `/etc/ppp/chap-secrets` contains secrets for `pppd` to use

when it authenticates itself to other systems, and also when it authenticates other systems to itself. Each line in a secrets file contains (at least) the name of a client, the name of the server, and a secret specific to that particular combination of client and server.

For a full explanation of the format and usage of the `/etc/ppp/chap-secrets` file, see the Linux manual page for `pppd(8)`.

6.12 mgetty configuration

The next two entries in the menu allow you to alter the *mgetty* configuration files, `/etc/mgetty+sendfax/mgetty.config` and `/etc/mgetty+sendfax/login.config`.

Clicking on either of these entries brings up a page in the work area which enables you to edit the file and its attributes directly, in the same manner as the PPP configuration files above.

The Web interface does *not* check that the content of the files will be understood. You should ensure that the file is valid before committing any changes, referring to the Linux manual page for `mgetty(8)` if you are unsure.

When you have finished editing a file, clicking **Save changes** will write the changes to disk. The changes will not take effect, however, until all *mgetty* services are restarted. You can restart all running *mgetty* services by clicking *Restart mgetty*; whilst this occurs, the network will be briefly unavailable.

mgetty.config

The file `/etc/mgetty+sendfax/mgetty.config` is the main configuration file for *mgetty*.

For full details on the format of the file and the options available, see the explanatory comments within the file, or the Linux info documentation for `mgetty`.

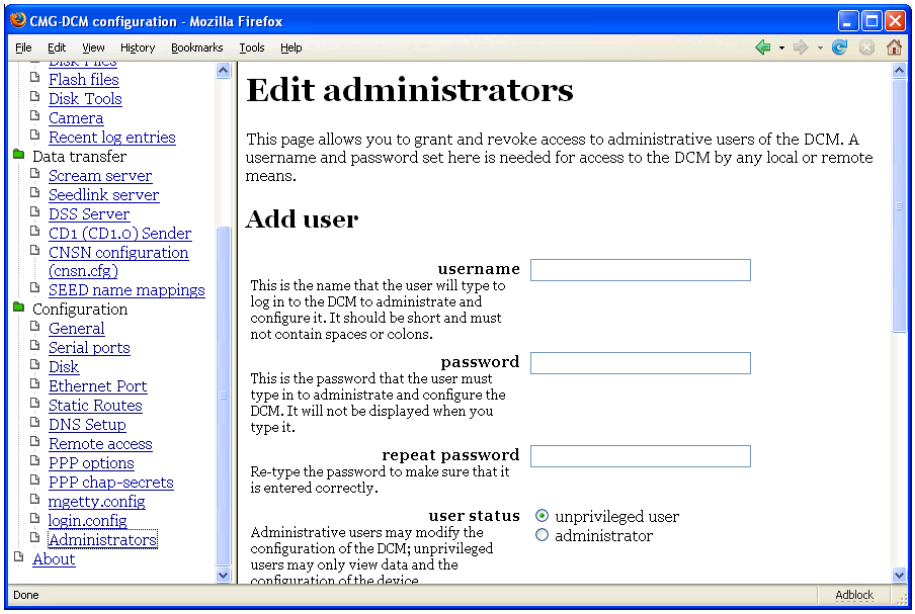
login.config

Normal user logins are handled by the program `/bin/login`. However, you may want to run a different program to handle logins by certain users. You can do this by editing the file `/etc/mgetty+sendfax/login.config`.

For full details on the format of the file, see the explanatory comments within the file, or the Linux info documentation for `mgetty`.

6.13 Administrators

This page allows you to add new users to the DCM, and change the passwords of existing users.



Users of the DCM can be one of two kinds.

- *Unprivileged users* can log in or view the Web interface of the DCM, including its current configuration, but cannot change any of the options, nor configure any instruments attached to the DCM

- *Administrators* may change settings or configure attached digitizers as they please.

You may want to give different users separate accounts for tracking or other purposes, or to allow users to set their own preferences when logging in. Alternatively, you may prefer to restrict access only to those with administrative requirements, by removing any unprivileged user accounts. If you want to access the DCM by *any* means, locally or remotely, you must have a username and password on the list given here.

If this is the first time you have logged in to the DCM, there will be a single user, `root`, whose password is factory-set. To add a new user, fill in their username and password in the top two boxes of the page. Repeat the password in the next box, to ensure it is entered correctly (since it will not be visible on screen), and click on **Add user**. A username may not contain spaces or colons.

You should be shown an acknowledgement screen, indicating that the new user account has been created. The user will be able to log in with that username and password by any of the methods currently available to the device, and from any location. More complex access controls cannot be performed through the Web administration system, although you can of course use the features of the DCM's underlying Linux operating system to implement them if required.

To remove a user account, choose their username from the drop-down box below *Existing accounts* and click **Delete user**. If you delete your own account, you will no longer be able to log in to the DCM, although you may finish what you are doing in the current session.

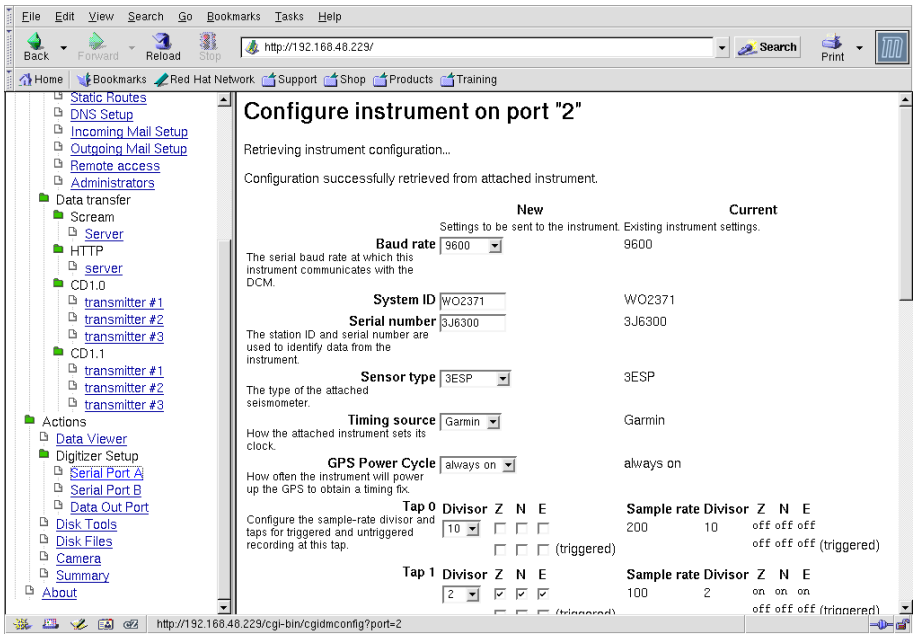
You can also change the password for any account, for example if you believe it to have been compromised, or if you are assigning it to a new person. To do this, choose their username from the same drop-down box and repeat the new password in the two boxes at the bottom of the page. Again, the password will not be visible on screen. Click *Change password* to make the change.

7 Configuring digitizers

To configure a digitizer, open the **Configuration – Serial Ports** page and find the entry for the serial port attached to the digitizer.

Click the *Configure* – **Digitizer** link to the right of this entry.

Alternatively, click the same link in the serial port table shown on the front (**Summary**) page.



This page allows you to set up Güralp instruments attached to any of the DCM's serial ports.

Assuming that the DCM has detected an instrument, its current configuration is now shown on the right side of the page for reference. In the centre of the page, the settings are repeated in a form that you can change. Once you have finished making changes, click *Configure instrument*. The DCM will then attempt to alter the settings on the digitizer to reflect your choices; this done, you should see the message `New configuration successfully saved to attached instrument`.

If the DCM is connected to a PC running Güralp Systems' Scream! software, you can also configure the digitizers from within Scream!. See the User Guide for your digitizer model for more details.

7.1 General digitizer settings

Baud rate : The speed at which the digitizer will communicate with the DCM, in bytes per second. This must match the baud rate the DCM is using for the serial port linked to this digitizer. The DCM's baud rates can be altered on the various *Serial port configuration* pages: see Section 6.2, page 82.

You should ensure that the baud rate is high enough to allow all the data to be transmitted at the rates you have chosen. As an example, for three streams transmitting at 100 Hz, a rate of 9600 baud is usually sufficient. Modern modems can normally operate at rates up to 57600 baud (~56 kbits/s), although the telephone or transmission lines may not support such a high rate. The same is true of radio telemetry links.

System ID and serial number : Together, these two fields uniquely identify data originating from a particular instrument.

Every data or status block sent by the digitizer will contain them as the first two 32-bit fields in the header.

On delivery of the digitizer from the factory, the *System ID* is set to the Güralp Systems works order number, and the *Serial number* is set to the serial number for that digitizer.

You can set the *System ID* to any combination of up to 5 letters (A – Z) and numbers. The *Serial number* can be up to 4 characters long, also using letters and numbers only. For example, you may wish to set the *System ID* to a more easily-recognised value, such as an abbreviation of your institution's name.

Sensor type : If the sensor attached to the digitizer is a Güralp velocity sensor, mass control functions (such as sensor locking, unlocking and centering) may be performed through the digitizer and DCM.

Different types of sensor have different functions available. This field allows you to change the type announced by the digitizer.

Timing source : The digitizer needs to be able to time-stamp accurately all data that passes through it. It can set its clock either by receiving time signals from the GPS satellite network using an attached Garmin GPS unit, or by taking time information from a central site via the DCM (*stream sync* mode). In *stream sync* mode, the digitizer expects to receive GCF packets from the central timing source (which may have its own GPS unit, or take signals from one of the radio time standards). The DCM can recognise GCF timing packets and will pass them on to all connected digitizers.

GPS Power Cycle : If you have selected *Garmin GPS* as the timing source, above, this setting determines how often the attached instrument will power up the GPS receiver to obtain an accurate timing signal. Between timing fixes the instrument will run on its internal clock, saving power at a small expense in accuracy. If your instrument has ready access to a power source, you should select *always on*.

7.2 Digitizer output control

The analogue-to-digital converters on a DM24 output data sampled at 2000 Hz, which is then filtered and reduced to a lower rate (*decimated*) using an on-board digital signal processing (DSP) unit. The DSP has four filtering-decimation stages, which run one after the other. Each can be programmed to reduce the sampling rate by a factor between 1 and 10. The output of each stage is called a *tap*.

Each of the taps may be configured for a different decimation factor by choosing values from the drop-down menus on the left. If you are using a mouse wheel to select values from a drop-down menu, ensure you remove the focus from the drop-down menu before scrolling the window, or you may inadvertently change the setting.

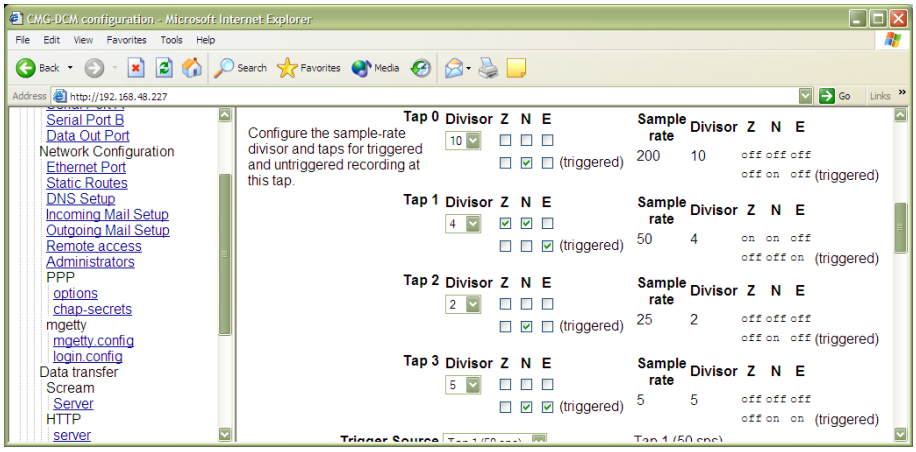
Not all digitizers support the full range of taps and decimation factors. For example, the Güralp DM24 allows you to select decimation factors of 2, 4, 5, 8, and 10 only, and does not allow the decimation factor of Tap 0 to be altered from its default setting of 10. In addition, no combination of decimation factors may be used which produces a non-integer data rate (in Hz). A full list of possible tap combinations for the DM24 is given in Section 7.3, below.

To the right of each decimation factor menu is a grid of six check-boxes marked *Z*, *N*, and *E.* These boxes mark which streams of data to record at each sample rate. Three streams of data are measured by the seismometer, corresponding to movement along each of three perpendicular axes. Although all the streams are decimated by the same set of successive scale factors, you can decide at which stage(s) of processing each stream outputs data. A tick in one of the check boxes will produce an output for a particular channel (column) at the corresponding sample rate (grid).

Each grid also has two rows, which differentiate between constant and triggered output. If a box in the upper row is ticked, that stream will produce output constantly at the corresponding sample rate. If the box below it is ticked, that stream will only produce output at that rate *if* a particular set of trigger criteria are also met. If the constant-output check box is ticked, the other will be ignored.

The table to the right shows the current setting of the digitizer.

For example:



In this example, under normal conditions,

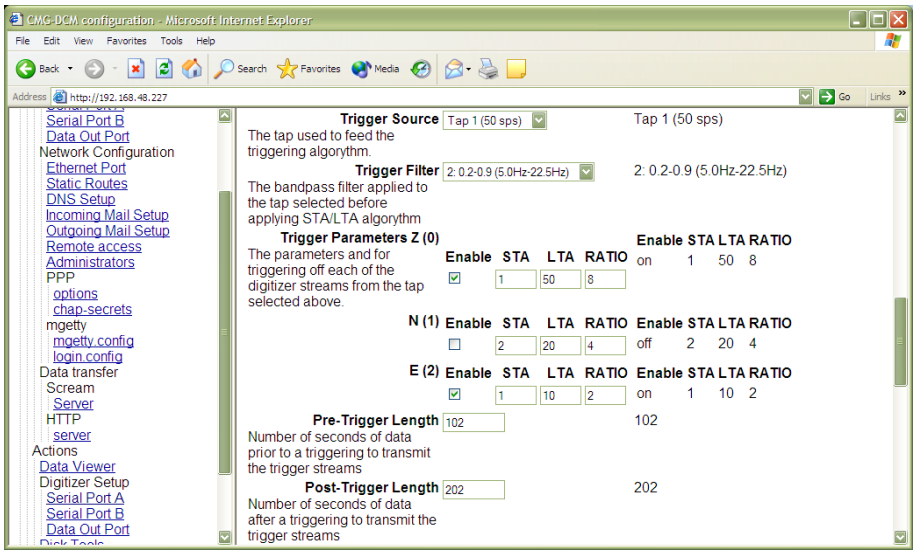
- the *Z* and *N* data streams will be output by Tap 1 at a rate of 2000/10/4 = 50 Hz, and
- the *E* data stream will not be output.

Additionally, when the trigger criteria are met,

- the *N* data stream will be output by Taps 0, 2 and 3 at rates of 2000/10 = 200 Hz, 2000/10/4/2 = 25 Hz and 2000/10/4/2/5 = 5 Hz, and
- the *E* data stream will be output by Taps 1 and 3 at rates of 2000/10/4 = 50 Hz and 2000/10/4/2/5 = 5 Hz.

The next section allows you to alter the criteria that the digitizer uses when deciding whether a trigger event has occurred.

7.3 Trigger criteria



The triggering algorithm applies a simple short-term average / long-term average calculation to the triggering stream. It works by identifying sections of an incoming data stream when the signal amplitude increases. The purpose of taking a short term average, rather than triggering on signal amplitude directly, is to make it less likely that spurious spikes will trigger the device. Averaging also introduces an element of frequency selectivity into the triggering process.

You can select which tap is tested for the trigger from the **Trigger source** drop-down menu. The tap does *not* have to be selected for data output for you to be able to use it here.

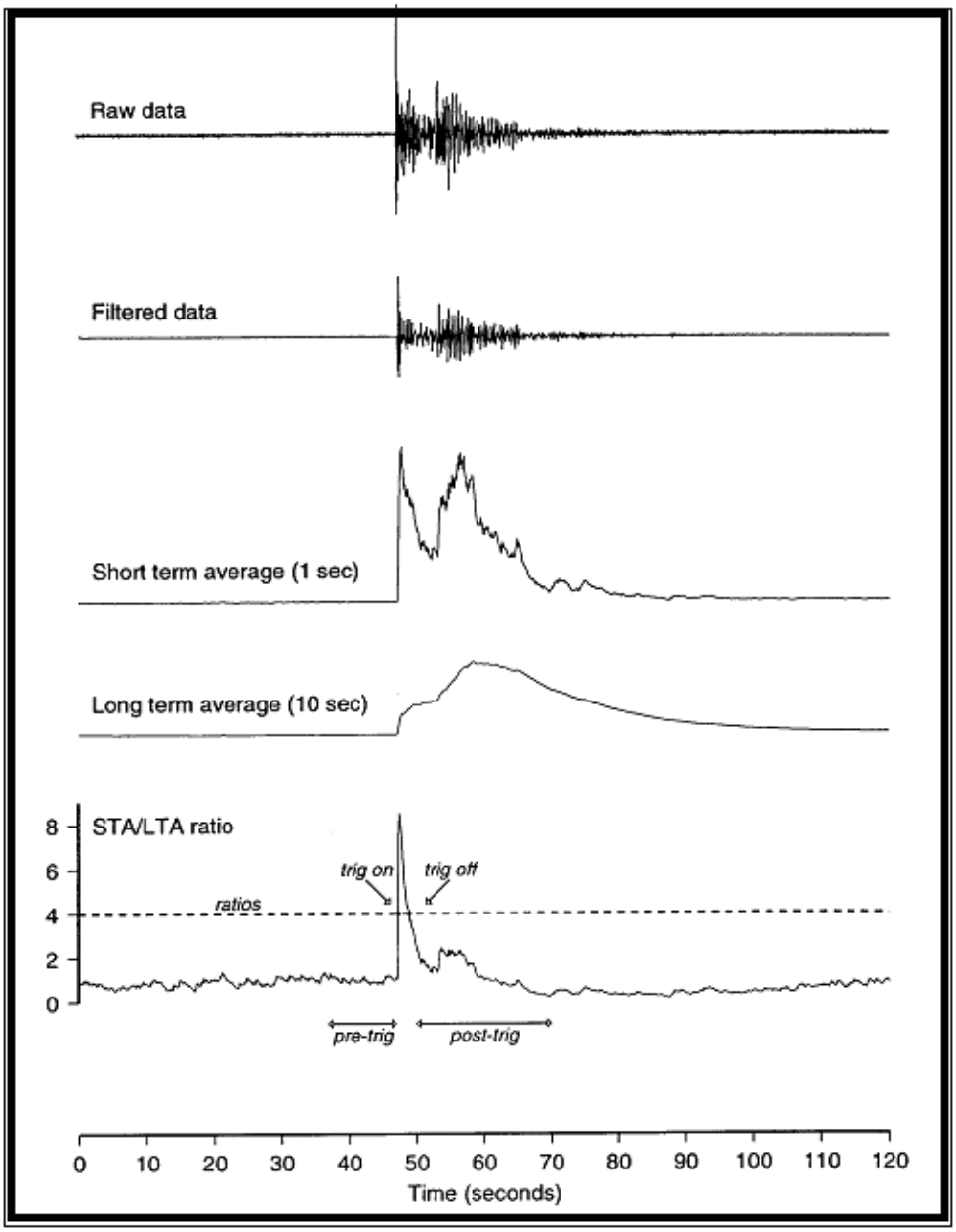
The next option, **Trigger filter**, allows you to apply a bandpass filter at this stage (see below.)

Any or all of the channels available at the tap you have selected may be used to determine a trigger. The next part of the window lists the channels, each with **Enable, STA, LTA** and **Ratio** settings. The **Enable** boxes determine which channels are considered for triggering. If *any* of the checked channels passes the trigger condition, the trigger will activate, and will not detrigger until *all* of the checked channels have fallen below their respective **Ratio** values.

The **STA** and **LTA** columns allow you to set the intervals over which the two averages are calculated, in seconds. Typically, the time interval for the short term average should be about as long as the signals you want to trigger on, while the long term average should be taken over a much longer interval. Both the STA and LTA values are recalculated continually, even during a trigger.

The **Ratio** column determines by what factor the STA and LTA must differ for the trigger to be passed. Finding the ratio most suited to your needs is best done by experiment. Too high a value will result in events being missed, while too low a value will result in spurious non-seismic noise triggering the system. Like the averages, their ratio is continuously recalculated for all components. Note that none of the boxes are allowed to be empty, and so you will need to enter the new value before removing the old one. Alternatively, you can use the up and down cursor keys to change the values.

For example, setting the **STA** to 1 second, the **LTA** to 10 seconds and the **Ratio** to 4 would give rise to the following trigger behaviour:



Usually, the values of the **STA** and **LTA** periods, and of the **Ratio**, will be the same for all checked channels. For convenience, Scream! will automatically fill in other values to match ones you enter. If you want to use different values for some channels, you should uncheck **Common values** before altering them.

If you are using Scream!, you can use the *Control* window to change the values of the **STA** and **LTA** periods, together with the **Ratio**, without restarting the digitizer. See the documentation for Scream! for more details.

Since it is not generally advisable to trigger from broadband data, the digitizer provides a set of standard bandpass filters to apply to the data streams before they are tested for the trigger condition. This filtering serves to maximise sensitivity within a the frequency band of interest, and filter out noise outside this band. You can select which bandpass filter to use from the **Trigger filter** drop-down menu. The corner frequencies of the pass band of the filter are determined by the Nyquist frequency, which is given by the sampling rate of the triggering data. The three filter options have pass bands between 10 % and 90 %, between 20 % and 90 % and between 50% and 90% of the data's Nyquist frequency, respectively.

The possible filter configurations are shown in the following table:

Tap #	Rate (samples/s)	Bandwidth 1 (Hz)	Bandwidth 2 (Hz)	Bandwidth 5 (Hz)
0	200	10 – 90	20 – 90	50 – 90
1	100	5 – 45	10 – 45	25 – 45
	50	2.5 – 22.5	5 – 22.5	12.5 – 22.5
	40	2 – 18	4 – 18	10 – 18
	25	1.25 – 11.25	2.5 – 11.25	6.25 – 11.25
	20	1 – 9	2 – 9	5 – 9
2	50	2.5 – 22.5	5 – 22.5	12.5 – 22.5
	25	1.25 – 11.25	2.5 – 11.25	6.25 – 11.25
	20	1 – 9	2 – 9	5 – 9
	10	0.5 – 4.5	1 – 4.5	2.5 – 4.5
	8	0.4 – 3.6	0.8 – 3.6	2 – 3.6
	5	0.25 – 2.25	0.5 – 2.25	1.25 – 2.25
	4	0.2 – 1.8	0.4 – 1.8	1 – 1.8

Tap #	Rate (samples/s)	Bandwidth 1 (Hz)	Bandwidth 2 (Hz)	Bandwidth 5 (Hz)
3	2	0.1 – 0.9	0.2 – 0.9	0.5 – 0.9
	25	1.25 – 11.25	12.5 – 11.25	6.25 – 11.25
	10	0.5 – 4.5	1 – 4.5	2.5 – 4.5
	5	0.25 – 2.25	0.5 – 2.25	1.25 – 2.25
	4	0.2 – 1.8	0.4 – 1.8	1 – 1.8
	2	0.1 – 0.9	0.2 – 0.9	0.5 – 0.9
	1	0.05 – 0.45	0.1 – 0.45	0.25 – 0.45

As can be seen, the filter you choose defines the set of permissible sample rates.

7.4 Auxiliary (“Mux”) channels

Güralp digitizers provide a range of slow-rate auxiliary channels for reporting the system's state of health and other diagnostic information, known as multiplexed (“Mux”) channels. The number of Mux channels depends on the model and configuration of your digitizer. Generally, three channels are used to report the sensor mass position, and another measures the internal temperature of the digitizer. In addition to these, up to 12 Mux channels may be supplied for the user's own purposes. Some digitizers have a separate *AUXILIARY* port which can be used to access these channels.

You can choose which, if any, of these channels should be transmitted to the DCM in the next section of the page. If one of the check-boxes is ticked, the digitizer will output data on that channel to the DCM, which will then store or transmit it with the rest of the data, according to the way it is configured.

Since it is an optional feature, the digitizer may not use the *Pressure* Mux channel to report pressure data. If this is the case, that channel may be used for another purpose. Likewise, the three channels marked *Spare* may not be used, depending on the optional features present in the instrument.

For more information on the format of data packets transmitted on the Mux channels, please refer to the documentation supplied with your digitizer.

7.5 Sensor mass control

There are three commands which can be relayed to an instrument through the DCM to control the position of its sensor mass. For each instrument the buttons which perform these commands (**Centre instrument**, **Lock instrument**, and **Unlock instrument**) can be found at the bottom of the *Digitizer Setup* page. The type of sensor you have installed determines which, if any, of these commands have any effect.

- The *CMG-40T* sensor type has no mass lock or centring capabilities, so all three buttons are inactive.
- The *CMG-3ESP* sensor type has manual mass lock and remote centring, so only the *Centre instrument* button is active.
- The *CMG-3T* sensor type is an automated analogue instrument, with all three commands available.
- The *CMG-3TD* sensor type is a fully digital borehole instrument, with all three commands available.

Lock and **Unlock instrument** are provided so that you can secure the seismometer's sensor mass for safe transportation. Once installed and unlocked, you should **Centre instrument** to move the mass to the correct position to start measuring data. During the execution of any of these commands, the system will display the three components of the mass's current position and update them once per second per component. When the masses are correctly centered all three readings should be less than ±1,000 counts. Locking or unlocking the sensor mass typically takes several minutes to complete.

8 Inside the DCM

The DCM is a fully-functional, Linux-based computer system especially designed for handling seismic data. It can collect and store data from several sources and, if required, output it in your preferred format to other locations on your network or on the Internet. This is done in the following manner.

Firstly, the DCM receives some data from an instrument connected to it. This can be any of

- a digitizer connected through a serial link,
- a computer running a Scream! server,
- a CD1.0 or CD1.1 transmitter (optionally), or
- another DCM or AM.

All the received data is stored in files in the on-board Flash memory. There are two banks of Flash memory available, which are accessible as `/nand0` and `/nand1` in the Linux file tree. Data is normally stored as GCF (Güralp Compressed Format) files.

As an option, you may be able to configure the DCM to use the *miniSEED* or *sac* formats instead (see Section 6.3, page 86.)

In *automatic* mode, when the Flash memory becomes more than 75% full, the oldest data files are moved to the DCM's primary hard disk until it is less than 50% full. If you prefer, you can configure the DCM to write to the hard disk at set intervals.

Writing to the hard disk is performed robustly, so that no data will be lost if a write is aborted (see Section 8.1, page 116.) This means that you can safely swap hardware in and out at any time. Stand-alone DCM modules use off-the-shelf Lacie hard disks, which can be easily removed and installed in most conditions. You can specify other models of IDE / USB or IEEE 1394 2.5” disk at manufacture. If an internal disk is not present, and the module has a USB *host* interface, it will look for hard disks connected to its external USB port.



Once the data is stored on the DCM, whether in Flash memory or on the hard disk, it can be retrieved

- by a remote computer running Güralp Systems' Scream!, or other GCF-compatible software;
- by another DCM or AM, also using GCF;
- by setting up a CD1.0 or CD1.1 transmitter on the DCM;
- by direct file transfer (using SSH, HTTP, HTTPS, etc.,)
- optionally, by requesting the data using SeedLink or AutoDRM.

A PC running Güralp Systems' Scream! software can not only collect data from the DCM, but also configure the module and any instruments attached to it.

You may need to enable and configure some of these methods before you can use them: see Chapter 6, page 80, for more details.

Most installations of the DCM will not require any more complex setting up than the Web configuration system can offer. However, in some cases you may need to take advantage of the flexibility offered by the underlying Linux operating system.

8.1 File systems

The DCM uses the standard Unix/Linux file naming conventions. The operating system resides in two blocks of Flash memory, mounted on `/` and `/boot`; when the module is powered up, a separate boot loader loads the rest of the operating system.

Once the operating system is loaded, the main Flash memory blocks (where present) are mounted on `/nand0` and `/nand1`. Incoming data, which may be from several sources, is combined into a single stream and placed in one of these blocks (whichever is less full). When in use, you can expect each to be between 50% and 75% full, with several da:ta files present. If the DCM is using GCF as its storage file format (recommended; see Section 6.3, page 86) then each file will be named after the timestamp on its first packet of data, in the following fashion:

`file-yyyymmdd-jjj-s-cccccccc.gcf` where `yyyymmdd` represents the date of the earliest data packet in the file, `jjj` the number of full days elapsed since midnight on January 1, `s` the time segment within the day (each day is divided into eight 3-hour segments), and `cccccccc` a unique hexadecimal code included to ensure filenames do not coincide.

When one of the Flash memory blocks approaches capacity, or after a fixed time period (if you have configured it to do this) the DCM will automatically move them onto the primary USB hard disk. This may be either an internal Lacie hard disk, or an external drive connected to the module through a USB *client* interface. This disk uses a specially-designed journalled filesystem, which is designed to maintain the integrity of your data at all times. Even if a write operation fails or is aborted suddenly, the disk will still contain a valid filesystem with all previously-saved data intact, which can be read using any driver that supports FAT32.

There is a set of specialised commands which allow you to perform basic tasks on this filesystem:

`gfat32df` : Displays the size of the filesystem, and how much free space remains, in a format similar to this:

```
FAT32 filesystem has 15 G bytes free
Partition is 37 G bytes (78140097 blocks of 512 bytes)
```

If no suitable storage medium can be found, you will see the message `Failed to find a USB disk`.

`diskman` : Ensures that the `/nandx` partitions do not become full by

moving files when necessary. This program ordinarily runs constantly in the background. However, a user can use the command `diskman -f` to force the Flash memory to be entirely copied to the USB disk. Typing `diskman -f` is identical to clicking the *Flush flash* button on the *Disk tools* page. It does *not* remove data from the Flash memory. If you issue `diskman -f` and then swap hard disks, the data remaining in memory will later be written out to the new hard disk, causing some overlap between it and the old disk.

Whilst the DCM is copying the contents of the Flash memory to disk, you will be shown a log of its progress. The USB interface can transfer data at a speed around 100 Kb/s, so large files may take several minutes to complete.

Once a file has been moved from the Flash memory to disk, any further data received which would otherwise be appended to that file will instead be placed in a new file in the Flash memory. Because of this, a stream may occasionally be fragmented. The *automatic* options are chosen to minimize this likelihood by only moving the oldest files, and by keeping files in Flash memory for a reasonable period of time. If you choose to transfer files to disk more often than this, more files will be fragmented.

`gfat32 ls` : List the files present on the hard disk, with the size of each file.

`gfat32 cpf` *`filename-on-disk`* *`destination-filename`* : Copy a file from the disk into temporary storage in the Linux filesystem (*e.g.* in your home directory.) Once the file is in the Linux file system, you can modify or convert it using your own scripts running on the DCM, or use programs such as `scp` to transfer it to a remote machine.

`gfat32 cpt` *`source-filename`* *`filename-on-disk`* : Copy a file from the Linux file system onto the disk.

`gfat32 mv` *`filename`* *`new-filename`* : Rename a file on the disk.

8.2 Command line tools

The DCM module's Linux operating system can be accessed over a network via SSH. There are many programs available for your computer which implement this protocol: `ssh` is included as part of most Linux and Unix distributions, whilst for Windows `putty` is a reliable free client. `ssh` is essentially a secure version of programs like `rlogin` and `telnet`, and provides a simple command line interface to the device. Access to the DCM by SSH is enabled by default, although

you can disable it using the *net.remoteaccess.ssh* configuration option.

In addition, if you have so configured it, you can connect directly to an RS232 port running the *getty* service.

Once the connection has been set up and you have logged in, you will then be shown a command prompt:

```
~ #
```

By default, your account uses the standard Bourne shell, `sh`. If you prefer, the more advanced shells `ash` and `bash` are also available. Many standard Unix programs are also present: `ls`, `cat`, `more`, `sed`, etc.

The following sections describe how to operate a DCM from the Linux command line, including descriptions of all commands unique to the DCM. Any of these commands can be included in your own shell script files, which can be run as services on the DCM or remotely using a `ssh` connection as required. If you need to compile your own C or FORTRAN programs to be run on the DCM, please contact Güralp Systems for assistance.

8.3 Configuration

Configuring the DCM is automated by a suite of command line tools. These maintain a configuration database and check that all the relevant Linux files are kept up to date. If you alter the standard Linux configuration files, you should bear in mind that these tools will overwrite them without checking that they match the information in the database. Because of this you should use the tools wherever possible rather than editing the files directly. The DCM's Web-based configuration system is just a front-end to these tools.

`gcfgdbls` `prefix` : Enter this command to find out which configuration options begin with the prefix `prefix` (case sensitive—all the configuration options are in lower case). The options are listed in alphabetical order. For example:

```
~ # gcfgdbls serial.0
 serial.0.baudrate
 serial.0.handshaking
 serial.0.ppp
 serial.0.service
```

`gcfgdbset` *`option-name`* *`new-value`* : Enter this command to set the value of the named option to `new-value`. The database will perform a simple type check on your value (for example, to check that

certain options are numbers), but will not otherwise make sure that your change makes sense.

`gcfgdbget` *option-name-or-prefix* : Enter this command to find the current value of the named option. Instead of a single option name, you can also use a prefix (as described above) to find out the values of a range of options. The remaining tools also allow you to use prefixes in place of full option names.

Each option in the database can be marked either as “clean” or as “dirty”. This flag tells the DCM whether the database is currently in sync with the state of the device. Whenever you alter the value of an option using the `gcfgdbset` tool, or using the Web interface, the option is marked as “dirty”; the DCM then alters its configuration, and marks the option as “clean” again, to signify that the change completed successfully. Three more commands are provided to allow you to access this flag:

`gcfgdbmark clean` *option-name-or-prefix* : Enter this command to mark the named configuration option as “clean” in the database.

`gcfgdbmark dirty` *option-name-or-prefix* : Enter this command to mark the named configuration option as “dirty” in the database.

`gcfgdbisdirty` *option-name-or-prefix* (*option-names-or-prefixes...*) : This command finds out whether any of the named options is marked as “dirty”. If none are marked, the command exits successfully; otherwise, it exits with a failure code. Thus you can use `gcfgdbisdirty` in your own shell scripts, to perform actions depending on the status of the DCM's configuration options.

Note that marking an option as “dirty” will not necessarily lead to any action being taken. For example, you cannot force a service to be restarted using these commands.

Each of the four tools can also take two command-line options:

`gcfgdb`*xxx* `-h` : Displays a short usage reminder about the command `gcfgdb`*xxx*. All other arguments will be ignored.

`gcfgdb`*xxx* `-c config-database other-arguments` : Normally, all the `gcfgdb`*xxx* commands work on the system configuration database. However, you can supply the file name of an alternative database using the `-c` option, in which case the command will be

performed on that database instead.

For a detailed description of the configuration options in the database, see Chapter 6, page 80.

Digitizer console access

The command `gcli` allows you to pass commands directly to the digitizer's console. The syntax is:

`gcli` *`port-number`* `[-f] [-r]` *`command`*

where *`port-number`* refers to the serial port connected to the digitizer. You can use `serialmap` to discover which serial port has which number, as described below. *`command`* is a DM24 FORTH command, which may contain several words. For information on the commands available on the DM24, please see its own documentation.

`gcli` will wait for the digitizer command to finish before exiting, and will only output any response from the digitizer afterwards. If you need to issue a command which monitors a value, you will need to connect to the digitizer's console directly using Scream! or `minicom`.

If you want to pass special characters to `gcli`, using the `-f` option allows you to use backslash sequences as used by the C command `printf` (*e.g.* `\t` = tab character, `\n` = newline, `\r` = carriage return, `\a` = bell, etc.) You can use `\r` to issue several commands to the digitizer in one session, since the carriage return will cause the digitizer to act on the previous command.

Passing the `-r` option causes the digitizer to reboot automatically once the command is completed. This is useful if you are using `gcli` to change the configuration of the digitizer, since many options require a restart for any changes to take effect.

If you need an interactive session with the digitizer, you can use the Linux terminal program `minicom`, which has been configured specially to cooperate with the DCM's various serial services. You can open a session with a digitizer by issuing the command

`minicom -n` *`port-number`*

If the port you specify is set to *gcf_in*, the DCM will automatically interrupt the data flow from the digitizer to allow you to enter commands.

When you have finished your session, press `CTRL-a` then `q`. `minicom`

will ask you if you want to quit without resetting the connection. Choose `yes` to return the digitizer to data mode.

8.4 Monitoring

Data flow

You can check that the DCM is receiving data either by monitoring the *Summary* page of the on-board Web interface (see Section 4.1, page 51), or from a command prompt using the command `gnblocks`:

```
Key 0x007000: Blocks          0 (Port  0, name Data out port,
device /dev/ttySA0, baud 115200)
Key 0x007001: Blocks          0 (Port  1, name Port B, device
/dev/ttySA1, baud 9600)
Key 0x007002: Blocks        149 (Port  2, name Port A, device
/dev/ttySA2, baud 38400)
```

This command shows, for each port:

- the *Key* number (in hexadecimal) of the process on that port which deals with incoming blocks,
- the number of blocks received by that process,
- the internal port number of the port,
- the name you have assigned to it,
- the port's Linux device name, and
- the baud rate currently in operation on the port.

You can query a single port by using the port number or key as arguments to the `gnblocks` command:

```
gnblocks 2
gnblocks 0x7002
gnblocks 28674
```

(In the last example, 28674 is the key ID `0x7002` expressed in decimal: hex $7002 = 7 \times 16^3 + 2 = 28674$.)

Another way to find out the index, key ID, name or device name of a particular serial port is to issue the command `serialmap`. A line will be output for each serial port, in the form

```
Port  0, Key 7000, name tts0, device /dev/ttyS0, baud 115200
```

where `port` and `baud` are in decimal, and `Key` in hexadecimal. The related command `serialmap -k` returns the key ID in decimal.

Digitizer status

The Güralp DM24mk3 digitizer outputs status information as a separate stream. If you have a DM24mk3, you can monitor this stream with the command

```
dm24mk3cds -s port-number
```

Whilst this program is running, it will output any status blocks it receives on that port directly onto your terminal:

```
HPA1CD: Wed Jun 30 14:51:13 2004
        gps_fix=2 (0x32)
        gps_mode=A (0x41)
        gps_control=255 (on)
        gps_power=255 (on)
        gps_offset=167 ticks
        busy_counter=0 ticks
        locking=0 unlocking=0 centering=0
        calibration  V  N  E
```

The information given, after the stream ID (which will always end in `CD`), is as follows:

- a time-stamp for the status block;
- whether the GPS has obtained a fix (0 = has not received any data, 1 = has not obtained a fix, 2 = has obtained a 2D fix, and 3 = has obtained a 3D fix);
- the mode the GPS is running in (A = automatic, and M = manual);
- whether or not the system clock is being controlled by the GPS (255 = on);
- whether the GPS is currently powered up (255 = on);
- the current measured offset between GPS and the internal clock, in units of 500 ns;
- the current value of the “busy” counter, which counts down towards zero whilst certain digitizer processes (such as calibration) are active;
- whether the sensor is currentlty being locked, unlocked, or

centred;

- which, if any, of the channels are currently being calibrated (the channel appears with a + if this is the case).

Tamper lines

The command `tamper` provides information about the current state of the DCM tamper lines. Issued with no arguments, it will output the status of all tamper lines once every 10 seconds. Including the `-w` option causes it to exit after printing the status once only:

```
[root@dcm-87AD9C933DE1 ~]# tamper -w
Input  State   Last Closed (Low)          Last Open (High)
    0   open    (never)                   Mon Jan  5 21:50:37 1970
    1   open    (never)                   Mon Jan  5 21:50:37 1970
    2   open    (never)                   Mon Jan  5 21:50:37 1970
    3   open    (never)                   Mon Jan  5 21:50:37 1970
    4   open    (never)                   Mon Jan  5 21:50:37 1970
    5   open    (never)                   Mon Jan  5 21:50:37 1970
    6   open    (never)                   Mon Jan  5 21:50:37 1970
    7   open    (never)                   Mon Jan  5 21:50:37 1970
    8  closed   (never)                    (never)
    9  closed   (never)                    (never)
   10  closed   (never)                    (never)
   11  closed   (never)                    (never)
   12  closed   (never)                    (never)
   13  closed   (never)                    (never)
   14  closed   (never)                    (never)
   15  closed   (never)                    (never)
```

The output of the `tamper` command is included on the *Summary* page under *Actions* on the DCM Web site.

8.5 Updating the DCM

Over the Internet

The easiest way to ensure that your DCM has all the latest software packages is to update it over the Internet.

The DCM is provided with a simple command-line script, `upgrade`, which checks for new versions of all packages in the distribution on the Güralp Systems Web site and installs them as necessary. It also rebuilds the configuration database to work with the new packages, resets it, and reboots the machine.

Running `upgrade` will restore the DCM to its factory settings, so make sure you will still be able to communicate with it before you issue the command, either

- over a serial link to the *DATA OUT* port (115200 baud, 8 data bits, no parity bit, 1 stop bit, no flow control), or
- over a DHCP-enabled network (using an IP address provided by the DHCP server.)

From the hard disk

If your DCM does not have access to the Internet, you can update its software from a connected USB disk. You will need to make a copy of all the files in the directory http://www.guralp.net/cmgdcm/feeds, including all subdirectories, in a directory called /cmgdcm/feeds on the hard disk. On Linux, you can do this with the command

```
wget -np -nH -m http://www.guralp.net/cmgdcm/feeds
```

Once the files are on the hard disk, you can install it in a DCM and transfer the packages with the command

```
upgrade gfat32
```

This facility is available with versions 2.10 and greater of the `upgrade` package.

Removing support packages

The DCM as shipped includes a package `gsl-backdoor` which enables Güralp Systems to respond to support issues by remote administration (*e.g.* installing updated firmware or packages.) If you prefer not to allow Güralp Systems access to the DCM you should use the command

```
ipkg remove gsl-backdoor
```

to remove the package. This will not affect the operation of the system in any other way; however, it may prevent Güralp Systems' engineers from being able to assist you in the event of problems.

`ipkg` can also be used for other package management tasks; however, if you remove software from the DCM, or replace packages with versions incompatible with the rest of the system, you risk leaving the unit in an unrecoverable state. We recommend that you use only `upgrade` wherever possible, to ensure that you have a fully tested set of packages.

The firmware

Reinstalling the firmware from scratch is a more involved process, and you should only need to do it if the root or boot partitions of the DCM become corrupted. In this case, you can use the boot loader (which is resident in hardware) to update them. You will need a second DCM or Linux computer attached to the console port in order to do this.

Note that the *firmware* is not the same as the *distribution.* If you want to ensure that your DCM has all the latest software, you should use the `upgrade` command described above. Reinstalling the firmware will reset any changes you have made to the system.

The procedure for installing new firmware depends on which revision of the DCM design you have. All recent DCM units use "MkII" firmware. If you are unsure which hardware type you are using, contact Güralp Systems.

MkII DCM or AM

Mark II DCMs and AMs have a 256 kb bootloader, a 1 Mb kernel image and a 64 Mb file system image. Currently all surface DCM units are the Mark II design.

1. Download the latest revision of the root (and boot, if appropriate) files from the Güralp Systems website at `http://www.guralp.net/software/modules/DCM/`

2. Obtain the `flashdcm` tool from the same site. This is a Linux utility which enables you to access the DCM's firmware. If you are using a second DCM, `flashdcm` will already be present on it.

3. Note the baud rate of the DCM's console port (115200 by default).

4. Power down the DCM.

5. On the second computer or DCM, issue the command

   `flashdcm -r` *`root-file`* `-b` *`boot-file`* `-s` *`baud-rate`* `-p` *`root-password`* `-d` *`port`*

   where *`root-file`* and *`boot-file`* are the images you wish to transmit, *`baud-rate`* is the baud rate of the DCM's console port, *`root-password`* is the password of the user `root` on the DCM (*not* any other administrative account), and *`port`* is the port number of the serial interface (on the second computer or DCM)

which you will be using.

If the device has several serial interfaces, the command `serialmap` may help you determine which port number corresponds to which device:

```
[root @ dcm] # serialmap
Library version: libserialmap Version 1.0.5 with
LIBGCONFIGDB
4 serial ports
 Port  0, Key   7000, name ttyS0, device /dev/ttyS0, baud
38400
 Port  1, Key   7001, name ttyS1, device /dev/ttyS1, baud
19200
 Port  2, Key   7002, name ttyS2, device /dev/ttyS2, baud
19200
 Port  3, Key   7003, name ttyS3, device /dev/ttyS3, baud
19200
```

`flashdcm` also has an `-e` option, which wipes clean the Flash memory used for the filesystem *before* providing the new firmware. Some surface DCM units require you to do this when installing a root image. If you use the `-e` option to install a root image, you must also provide a new boot image, since the boot image in memory will be erased.

6. Power up the DCM. The second computer or DCM will detect messages coming from the bootloader of the DCM, and automatically interrupt the boot process to provide the new firmware.

7. Wait for the new firmware to be uploaded. The DCM will reboot automatically at the end of transmission, and the `flashdcm` process will terminate. This may take several minutes.

At this point, you will have a minimal installation of the DCM software, and can proceed to commissioning the system.

9 Connector pinouts

9.1 Modular DCM units

PORT A and B

This is a standard 10-pin mil-spec socket (02E-12-10S). The pinout is such that the port can be connected to the serial output of a DM24 digitizer using a straight-through cable.

Pin	Function
A	Power 0 V
B	Power +10 to +35 V
C	RS232 RTS
D	RS232 CTS
E	RS232 DTR
F	RS232 DSR
G	RS232 ground
H	RS232 CD
J	RS232 transmit
K	RS232 receive

DATA OUT port

This is a standard 10-pin mil-spec plug (02E-12-10P). The pinout is the same as the serial output of a DM24 digitizer, allowing you to insert a DCM into a pre-existing installation and maintain connectivity.

Pin	Function
A	Power 0 V
B	Power +10 to +35 V
C	RS232 CTS
D	RS232 RTS

E	RS232 DTR
F	RS232 DSR
G	RS232 ground
H	RS232 CD
J	RS232 receive
K	RS232 transmit

USB connector

This is a standard 6-pin mil-spec socket (02E-10-06S).

Pin	Function
A	+5 V DC (USB Type A pin 1)
B	Data –ve (USB Type A pin 2)
C	Data +ve (USB Type A pin 3)
D	0 V (USB Type A pin 4)
E	Shielding
F	Switched power +10 to +35 V

NETWORK connector

This is a standard 6-pin mil-spec plug (02E-10-06P).

Pin	Function
B	Data transmit +ve (RJ45 pin 1)
C	Data receive +ve (RJ45 pin 3)
E	Data receive –ve (RJ45 pin 6)
F	Data transmit –ve (RJ45 pin 2)

9.2 Integrated DCM units

DM/AM module output

This is a standard 32-pin mil-spec plug (02E-18-32P).

Pin	Function
D	Power +10 to +30 V
E	Power 0 V
J	Data transmit +ve
K	Data transmit –ve
L	Data receive +ve
M	Data receive –ve
R	AM console receive
S	AM console transmit
T	AM console ground
U	Tamper switch 5
V	GPS 1pps signal
W	GPS RS232 transmit
X	GPS RS232 receive
Y	GPS RS232 ground
Z	DM console transmit
a	DM console receive
b	DM console ground
c	Tamper switch 0
d	Tamper switch 1
e	Data ground
f	Tamper switch ground
g	Tamper switch 2
h	Tamper switch 3
j	Tamper switch 4

10 Sensor and digitizer types

These codes are used in situations where the DCM needs to know the response properties of sensors and digitizers, *e.g.* when compiling full SEED volumes.

If you are unsure about the code you should use, contact Güralp Systems.

10.1 Sensor response codes

Sensor	Sensor type code	Units (V/A)
CMG-5T or 5TD, DC – 100 Hz response	CMG-5_100HZ	A
CMG-40T-1 or 6T-1, 1 s – 100 Hz response	CMG-40_1HZ_50HZ	V
	CMG-40_1S_100HZ	V
CMG-40T-1 or 6T-1, 2 s – 100 Hz response	CMG-40_2S_100HZ	V
CMG-40T-1 or 6T-1, 10 s – 100 Hz response	CMG-40_10S_100HZ	V
CMG-40, 20 s – 50 Hz response	CMG-40_20S_50HZ	V
CMG-40, 30 s – 50 Hz response	CMG-40_30S_50HZ	V
CMG-3T or 3ESP, 30 s – 50 Hz response	CMG-3_30S_50HZ	V
CMG-40, 60 s – 50 Hz response	CMG-40_60S_50HZ	V
CMG-3T or 3ESP, 60 s – 50 Hz response	CMG-3_60S_50HZ	V
CMG-3T or 3ESP, 100 s – 50 Hz response	CMG-3_100S_50HZ	V
CMG-3T or 3ESP, 120 s – 50 Hz response	CMG-3_120S_50HZ	V
CMG-3T, 360 s – 50 Hz response	CMG-3_360S_50HZ	V
CMG-3TB or 3V / 3ESP borehole, 30 s – 50 Hz response	CMG-3B_30S_50HZ	V
CMG-3TB or 3V / 3ESP borehole, 100 s – 50 Hz response	CMG-3B_100S_50HZ	V
CMG-3TB or 3V / 3ESP borehole, 120 s – 50 Hz response	CMG-3B_120S_50HZ	V
CMG-3TB or 3V / 3ESP borehole, 360 s – 50 Hz response	CMG-3B_360S_50HZ	V
CMG-3TB or 3V / 3ESP borehole, 360 s – 100 Hz response	CMG-3B_360S_100HZ	V

10.2 Digitizer type codes

Digitizer	Digitizer type code
CMG-DM24 mk2 (3- or 6- channel)	`CMG-DM24`
CMG-3TD or 5TD using DM24 mk2 module (pre-2005)	`CMG-DM24`
CMG-3TD or 5TD using DM24 mk3 module (2006 and later)	`CMG-DM24mk3`
CMG-DM24S12 (including AMS)	`CMG-DM24`
CMG-DM24 mk3	`CMG-DM24mk3`
CMG-6TD	`CMG-6TD`

11 Revision history

2006-12-13	F	Added **datatransfer.scream.server.udp_push**
2006-11-21	E	Added new recording options; data viewer.
2006-10-06	D	Added *protocol*`.enabled` to `seedmap.xml`
2006-09-29	C	Revised for new user interface organization. Added new sensor type codes. Introduced `seedmap2.xml` and Process Overview.
2006-05-18	B	Added revision history, DSS, SeedLink and CNSN.
2005-12-13	A	New document