

AROW BFTP

AROW Data Diode

PYTHON SOFTWARE SETUP AND USE

Applicable Products

AROW Data Diodes -All Models

Part Number: AROW-MAN-0602

Document Reference

AROW-MAN-0602

Publication Date

Jan 2019 Issue 9

Published by

Somerdata Ltd.
Underwood Business Park
Wells
BA5 1AF
UK

Sales & Customer Support

Phone: +44 (0)1179 634050
E-Mail: sales@somerdata.com
 support@somerdata.com
Website: www.somerdata.com

AROWBFTP *Data Diode Management Application*



REVISION HISTORY

Issue	Date	Notes
9	Jan 2019	Software Version 2.0.0. Python 3 and Unicode support
8	July 2016	Software Version 1.3.1 –added Multicast support
7	Nov 2015	Software Version 1.3.0 Send refactoring, addition of high capacity transfer management
6	May 2015	Software Version 1.2.1 Refactoring or code, major enhancements and functional changes
5	Oct 2014	Software Version 1.1.1 Clarification of synchronisation
4	Feb 2014	Software Version 1.1.0 Web browser data status added UDP streaming description added INI file description added
3	Dec 2013	Hot file recovery function described
2	21 st Jan 2013	Additional application information, additional register functions, corrections and streaming examples.
1	15 Aug 2012	Initial Issue

CONTENTS

CONTENTS	1-1
1. INTRODUCTION	1-1
WHAT'S IN THIS USER GUIDE	1-1
USER GUIDE AVAILABILITY	1-1
2. PRODUCT DESCRIPTION	2-1
INTRODUCTION	2-1
BLOCK DIAGRAM	2-1
3. INSTALLATION	3-1
PREPARATION	3-1
NETWORK ADDRESSING	3-2
RECORD MAC ADDRESSES (AROW HA)	3-2
NETWORK CONNECTIONS	3-3
4. CONFIGURATION	4-1
INTRODUCTION	4-1
SETTING THE CONTROL PORT IP ADDRESS	4-1
SETTING THE DATA PORT IP ADDRESS	4-2
WINAROW	4-3
USING WINAROW.	4-3
CHANGING THE DATA PORT.	4-4
WINAROW OPERATION	4-4
5. OPERATION	5-1
INTRODUCTION	5-2
INSTALLATION	5-3
CYTHON	5-3
PATHS AND ENVIRONMENT	5-4
SOURCE AND DESTINATION PATHS	5-4
SETUP EXAMPLE RUNNING FROM PYTHON SOURCE CODE	5-4
OPTIONS	5-5
OPTIONS:	5-6
INITIALISATION FILE	5-10

COMPENSATING FOR DATA LOSSES	5-11
SUPERFRAME	5-11
REDUNDANCY (OPTION –R)	5-11
RATE LIMITING (OPTION –L)	5-12
INCREASE THE IP STACK BUFFER SIZE	5-13
LOAD SHARING	5-13
RECOVERY FROM INTERRUPTION	5-13
HOT FILE RECOVERY	5-14
HIGH-SIDE BUFFER OVERRUNS	5-15
HIGH CAPACITY TRANSFERS AND LEVELLING	5-15
FILE SYSTEM RESTRICTIONS	5-17
FILE SYMLINKS AND SUPPORT APPLICATIONS	5-17
WEBSITE VIEWING	5-18
TCP STREAMING	5-18
UDP STREAMING	5-19
MULTICAST UDP STREAMING	5-20
MULTICAST TESTING	5-20
MULTICAST DEMONSTRATION	5-21
AROWBFTP V2 AND UNICODE	5-22
DIRECT DATA STREAMING	5-22
WEBCAM	5-24
LARGE FILE PROCESSING	5-26
VIRTUAL MACHINES	5-26
6. CONTROL AND STATUS	6-1
INTRODUCTION	6-1
CONNECTING TO AROW's CONTROL AND STATUS PORT	6-2
PROTOCOL	6-2
COMMANDS	6-3
STATUS REQUESTS	6-4
CONTROL AND STATUS PYTHON SCRIPT	6-5
SETTING THE DATA PORT IP ADDRESS	6-5
RESETTING THE TCP STACK.	6-6
CONTROL AND STATUS MESSAGE FORMAT	6-7

MEMORY MAP	6-7
REGISTERS	6-8
GRAPHICAL DATA MONITORING	6-15
LOGGING	6-17
LOGFILE MANAGEMENT	6-18
7. SUPPORT	7-1
WHAT TO DO IF YOU HAVE A PROBLEM	7-1
SERVICING, MAINTENANCE AND REPAIRS	7-1
IF YOU NEED SUPPORT	7-1
SUPPORT REQUESTS	7-2
RETURNS	7-3
SOMERDATA CONTACT INFORMATION	7-4
END-OF-LIFE DISPOSAL	7-4
WASTE ELECTRICAL & ELECTRONIC EQUIPMENT (WEEE)	7-4
8. WARRANTY	8-1
INTRODUCTION	8-1
WARRANTY: TERMS AND CONDITIONS	8-1
9. NOTICES	9-1
IN THIS SECTION	9-1
GENERAL INFORMATION	9-1
SOMERDATA AND THE ENVIRONMENT	9-1
CURRENT COMPLIANCE ACTIVITIES	9-2
RESTRICTION OF USE OF HAZARDOUS SUBSTANCES (RoHS)	9-2
DECLARATION OF CONFORMITY	A
10. INDEX	10-1

1. INTRODUCTION

In this Section

WHAT'S IN THIS USER GUIDE	1-1
USER GUIDE AVAILABILITY	1-1

What's in this User Guide

This User Guide covers SomerData's **AROWBFTP** application software for the AROW Data Diode Optical Wormhole. For Hardware installation and setup, refer to publication AROW-0601.

*Section 2 – **PRODUCT DESCRIPTION*** gives an overview of the application capabilities and limitations.

*Section 3 – **INSTALLATION*** describes the process of installing AROWBFTP.

*Section 4 – **CONFIGURATION*** describes the process of configuring AROW for operation

*Section 5 – **OPERATION*** describes how to use AROW.

*Section 6 – **CONTROL AND STATUS*** describes the available software controls and status registers.

*Section 7 – **SUPPORT*** describes the procedure and contact details for obtaining customer support on this product.

*Section 8 – **WARRANTY*** your rights and obligations in support of this product.

*Section 9 – **NOTICES*** statutory documentation and certificates.

*Section 10 – **INDEX***

User Guide Availability

Printed copies of Hardware and Software User Guides are supplied with the original products on request.

Additional printed copies, including the Programmer's Reference Guide can be supplied on request. Please contact your local supplier or SomerData for ordering details.

Electronic copies (Adobe Acrobat files) are included on the SomerData electronic delivery medium that is supplied with the original products.

The User Guide library, which also includes product data sheets, can be accessed by browsing the `\Documents\` folder for the required document.

Additional and updated copies are available through our website. or can be supplied on request. Please contact your local supplier or SomerData for ordering details.

2. PRODUCT DESCRIPTION

In this Section

INTRODUCTION	2-1
BLOCK DIAGRAM	2-1

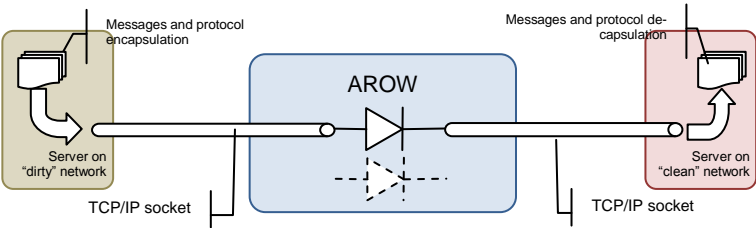
Introduction

AROWBFTP is a set of Python scripts designed to support the AROW Data Diode Optical Wormhole.

AROW is a 1U rack mountable high availability, TCP/IP based Gigabit Ethernet uni-directional network access device (Data Diode).

Internal simplex optical links between the unsafe network (dirty) and the secure and restricted network (clean) ensure a reliable data link in one direction and guarantees no reverse data path.

Block diagram



AROWBFTP consists of a set of Python scripts that can be run from a command line interface.

The software manages data transfers between the low and high sides of the data diode, ensuring data redundancy, data integrity checks, managing deletions and keeping the high side data set in synchronisation with the low-side data set.

By definition, no data transfer acknowledgments can be sent across the Data Diode from the high side to the low side . The absence of acknowledgements is compensated by redundant data transmissions, extra layers of error detection, and a framing scheme that allows connectionless TCP/IP protocol to be used.

The use of Python scripting ensures portability over operating systems (Windows, Linux, MacOSX,...)

3. INSTALLATION

In this Section

PREPARATION	3-1
NETWORK ADDRESSING	3-2
RECORD MAC ADDRESSES	3-2
NETWORK CONNECTIONS	3-3

Preparation

It is not the purpose of this document to teach Python and some familiarity with the use of Python scripts in Network System Administration is assumed.

However, unless customisation is required, there is no requirement for detailed Python programming skills.

Clearly, a pre-requisite of this application is that Python is installed and working on the target systems (a minimum of Python 2.7.10 for versions below V2, a minimum of Python 3.6 for version 2 and later.). See the Operation section for a guide to Python installation.

Pre-built binaries for major OS versions are available that incorporate all dependencies and can be run stand-alone.

There are two script sets, one on the low-side data server ('AROWSend') and one on the high-side data server ('AROWReceive').

Note that in this context, 'data server' simply means the system(s) on which the data files to be transferred and synchronised are stored.

In a typical system setup this could be a pair of network servers or a pair of Data storage servers with appropriate network access, privileges and security. Please note that the scripts themselves are not secured for access control, by design. Scripts should be run under access controls appropriate to the Operating System and User Access Controls in place.

If the High Availability (redundant) AROW is used, making use of its automatic failover, then the network servers may include redundant servers themselves, with failover control software running on each.

AROW features complete network and data isolation, and includes buffers to compensate for transient network congestion.

Network Addressing

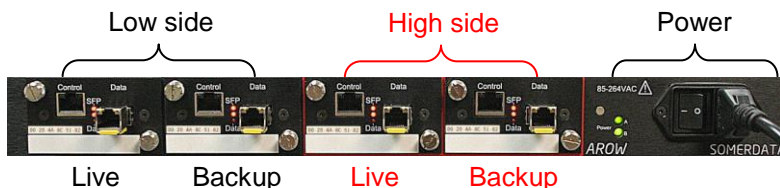
AROW is designed to work in IPV4 environments only. The failover requirements mean that static address resolution is required, a full AROW system can require up to 6 addresses (one for each of the 4 control ports and one for each of the two data ports). Note that AROW HA's unique IPClone system automatically activates a backup port on failover taking over the previously assigned address to the backup MAC address. Thus the low-side and high-side data port IP addresses remain constant.

Control port IP address assignments are factory-set and can be re-assigned during installation.

Data port addresses are set via the Control port, it is only necessary to initially set the Live Data ports (Low and High). For AROW HA, backup port assignment is handled internally.

During the physical installation process ensure that all MAC addresses are recorded

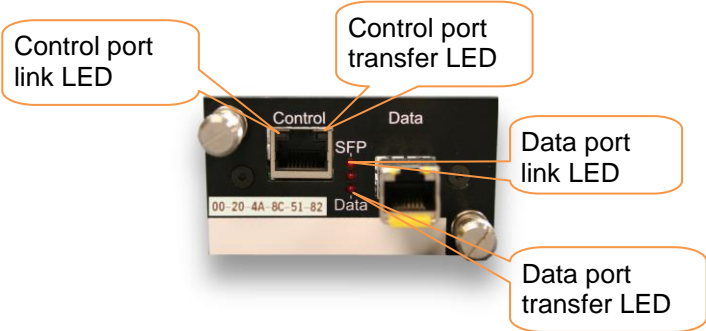
Record MAC addresses (AROW HA)



	Control port MAC address
Low live	00:20:4A:
Low backup	00:20:4A:
High Live	00:20:4A:
High backup	00:20:4A:

Network connections

Connect Low side Control and Data network ports to the dirty network and High side network ports to the clean network. High side data connections must be on low-contention Gigabit Ethernet networks. 10/100 Ethernet is NOT supported. Any hold-ups on the high side network will cause AROW's buffers to fill. If hold-ups on the high side network are greater than on the low side, the buffers will overrun, causing loss of data. Buffer state can be monitored by reading status (see section 6 CONTROL AND STATUS). Redundant system backup Control and Data connections should be to backup switch/routers.



Control port can be connected to 10/100 ports. When a good connection is made, the left LED on the control port RJ45 socket will turn green to indicate the link is up. The right LED will flash with Ethernet traffic. For security against tampering, these ports should only be connected during setup or diagnostic testing.

Data ports must be connected to 1000base-T ports. These can be either optical or copper. Before plugging in a network cable, the module LED labelled "SFP" will flash (If it is off, this indicates that no SFP module is detected).

When a physical Gigabit Ethernet link has been established, this LED will remain constantly on.

4. CONFIGURATION

In this Section

INTRODUCTION	4-1
SETTING THE CONTROL PORT IP ADDRESS	4-1
SETTING THE DATA PORT IP ADDRESS	4-2
WINAROW	4-3
USING WINAROW.	4-3
CHANGING THE DATA PORT.	4-4
WINAROW OPERATION	4-4

Introduction

To integrate AROW into a control network, the IP address for each Control and each Data port must be set.

Setting the control port IP address

The control port IP address can only be set from a machine on the same sub network. You will need the MAC address for each control port to be set. This is printed on each module panel.

After setting it is recommended to write the IP address on the white strip of the module panel in semi-permanent marker, and to keep a note in your System Administration journal.

For Windows, open a Windows command prompt (Start, Run, enter command or CMD depending on your operating system). For Linux/Posix, open a command line terminal.

From the command prompt enter the new ARP entry for the IP address you want to set as shown below:

```
ARP -S 192.168.xxx.xxx 00-20-4A-xx-xx-xx  
(for linux: arp -s 192.168.xxx.xxx 00:20:4A:xx:xx:xx)
```

Hit return

Open a new command prompt.

Type `ping -t 192.168.xxx.xxx`

(linux: `ping 192.168.xxx.xxx`)

The “pings” will fail to start with but will give an indication of progress.

Back to the first command prompt telnet to the same IP address **using port 1**.

e.g. Telnet 192.168.xxx.xxx 1

Hit return. (message ‘failed to connect’ should appear within 2 to 3 seconds)

The “Pings” should start to work.

At the next command prompt telnet to the same IP address **using port 9999**.

Telnet 192.168.xxx.xxx **9999**

Hit return. You will be prompted to "Press Enter to go into Setup Mode"

Hit return again as soon as you see the prompt to access the configuration choices. The prompt will time out after ~ 3 seconds.

Select 0 for server configuration.

Manually enter the IP Address. This permanently assigns the IP address,

Manually enter the gateway address (optional)

Manually enter the host bits for the subnet mask

Select 9 to save and exit

Repeat these steps for each module’s control port.

Setting the data port IP address

For each Data port, set the IP address using the "control_arow.py" script e.g. :

`./control_arow.py -H 192.168.xxx.xxx -i 10.0.0.yyy`

-H 192.168.xxx.xxx defines the control port that will be used to set the data port IP address and -i 10.0.0.yyy defines the data port IP address.

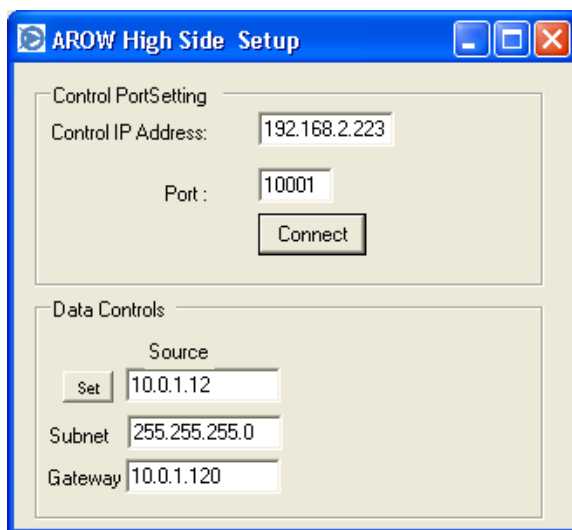
WinAROW

You can also use the WinAROW application (Windows only) to set and monitor the AROW data ports.

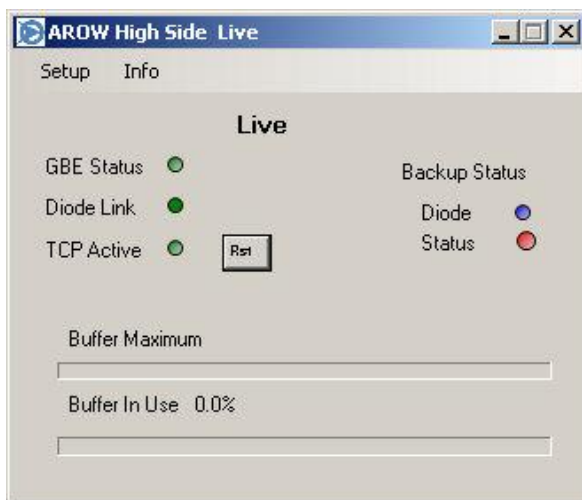
This is a Graphical User Interface application that constantly monitors the target data port and stream.

Using WinAROW.

Start the application from the desktop or installation folder. If this is the first time of running, you will be offered a Setup window. Set the Control Port IP address as recorded above.



Note that since each AROW module is fully independent you will need an instance of WinAROW for each module control address. The default Port is 10001 and should not normally need changing. Select Connect and the Main window should appear, populated with the current settings. Screenshots may



appear slightly differently, depending on your AROW configuration

Changing the Data Port.

Open Setup and enter the Data Port IP address, Subnet and Gateway that you require. Press set.

WinAROW Operation

The main window shows the state of the connected module and is dependent on the build state of AROW. If configured as a High Availability dual redundant system, this will show whether it is in a live or backup position and whether it is currently acting as live or backup. In Single or Dual channel systems, the backup information is not displayed.

Also shown is the condition of the attached data link (GBE/Optical) and the state of the internal buffer. This can be useful to monitor performance in high data flow situations and may inform any decision about the data throttling option in the main data python program. (AROWSend.py). Both the current and maximum states of the buffer are indicated. Note that there will be data loss if the buffer reaches maximum.

AROW is now ready to be used.

5. OPERATION

In this Section

INTRODUCTION	5-2
INSTALLATION	5-3
CYTHON	5-3
PATHS AND ENVIRONMENT	5-4
SOURCE AND DESTINATION PATHS	5-4
SETUP EXAMPLE	5-4
OPTIONS	5-5
OPTIONS:	5-6
INITIALISATION FILE	5-10
COMPENSATING FOR DATA LOSSES	5-11
SUPERFRAME	5-11
REDUNDANCY (OPTION –R)	5-11
RATE LIMITING (OPTION –L)	5-12
AROW-S	Error! Bookmark not defined.
AROW-G	Error! Bookmark not defined.
Processing Rate	5-12
INCREASE THE IP STACK BUFFER SIZE	5-13
LOAD SHARING	5-13
RECOVERY FROM INTERRUPTION	5-13
HOT FILE RECOVERY	5-14
HIGH-SIDE BUFFER OVERRUNS (AROW-S)	5-15
HIGH CAPACITY TRANSFERS AND LEVELLING	5-15
FILE SYSTEM RESTRICTIONS	5-17
FILE SYMLINKS AND SUPPORT APPLICATIONS	5-17
WEBSITE VIEWING	5-18
TCP STREAMING	5-18
UDP STREAMING	5-19
MULTICAST UDP STREAMING	5-20
MULTICAST TESTING	5-20
MULTICAST DEMONSTRATION	5-21

DIRECT DATA STREAMING	5-22
WEBCAM	5-24
LARGE FILE PROCESSING	5-26
VIRTUAL MACHINES	5-26

Introduction

AROWBFTP is written in Python and distributed under the GPLv3.0 licence, allowing relatively simple, auditable and portable code.

It has been successfully tested with Windows XP/7/8/10, MacOSX 10.3.9 and Linux 32 and 64-bit versions. It should also work on other operating systems that support Python, perhaps with a few minor modifications.

Pre-built binaries for Linux32/64bit and Windows 32/64 bit are supplied, as well as the source code.

When running from sources, it is necessary to install a Python interpreter, if possible the latest version available through the site <http://www.python.org>. N.B. AROWBftp prior to version 2 has been tested with Python 2.7.10, AROWBftp V2 is suitable for Python 3.6 onwards only.

Under Windows, the Win32 add-ons are also required. A simple solution is to install ActivePython (<http://www.activestate.com/Products/ActivePython>), that understands the Python interpreter and the PyWin32 add-ons, but this version of Python is not always the most recent. The add-ons for Win32 can be downloaded separately from <http://sourceforge.net/projects/pywin32>.

AROWBftp is based also on some third-party modules provided with the source code, that may use updates.

AROWBftp is divided into Send and Receive scripts and requires two host platforms, for example network servers.

AROWSend should be installed on the source network server. AROWReceive should be installed on the destination network data server or dedicated platform.

Prepare or create parent folders for the source files on the low side and received files on the high side. Ensure that these folders have appropriate permissions including write permissions.

The purpose of AROWBftp is to maintain the File Tree structure across the data diode.

It monitors when files have changed, but the detailed management of which files and data are made available is different for every deployment and will change over time, so is considered a Network Administration task.

Nevertheless, the purpose of AROWBFTP is to allow users on the protected network to view files as if they were on the unprotected network.

↳ path: <http://pypi.python.org/pypi/path.py>

↳ xfl: <http://www.decalage.info/en/python/xfl>

↳ plx: <http://www.decalage.info/en/python/plx>

Installation

Installing AROWBftp from sources consists of simply copying the provided files into any secured directory. It is recommended that the active scripts be restricted to read-only permissions. Consider using a virtual environment if there are other python-based applications on the system.

If using the pre-built binaries, it is only necessary to grant executable permissions (Linux) or Administrative rights (Windows.)

Cython

Some parts of the script can be enhanced by using Cython. This should also be installed for your platform. Subfolders containing the Cython source are provided with setup.py scripts to create the necessary object files. Note that Cython source files have the extension .pyx. Object files may have the extension .so, or pyd depending on the platform. Run 'setup.py build_ext -inplace' from the AROWSocket folder.

The purpose of including cythonisable files is to allow the maximum performance to be obtained, and all specification figures have been taken using this setup.

Note that if you do not have Cython on your platform, you will need to change the AROWReceive.py script to use the BFTPSocket.py script instead when running from sources.

Paths and Environment

Setting a PATH environment to the Python interpreter means that the program can be launched from a command window from its home directory by simply typing 'python AROWSend.py' (or AROWReceive.py), with suitable parameters as discussed below.

The normal use of AROWBFTP is to transfer files across a unidirectional link over a AROW Data Diode. For that, the contents of a platform source directory are copied to a target platform, and synchronised to changes if the source directory is modified over the course of time.

Application data can include flat-file mail databases, cached web sites, copy anti-virus signatures and software updates (Windows, Linux,...) from the internet through a secure network.

Distributed databases are usually not suitable directly. For these it may be necessary to export a file from the database (backup), transmit across the diode, and then import into a database on the protected side.

Place a copy of the AROWBFTP send and receive directories (including AROWSend.py/AROWReceive.py and all of its dependencies) into a path on each machine, transmit and receive, where the Python interpreter can find it.

Source and Destination paths

The base source and destination paths are set with the script parameters. These **MUST** exist – AROWBftp does not create them. They must be created with appropriate access rights (under Linux, consider creating a special AROW users group with read/write access).

It is not necessary to copy files and folders into the send directory, make symlinks to the directory trees to be transmitted and AROWBftp will find the sources.

On the receive side, it is not necessary to access the receive directory directly, again, symlinks can be used to direct users to their appropriate tree/files.

Setup Example running from python source code

As an example, the source server 192.168.1.2 transmits the directory « source » to the destination server 192.168.2.3 into the directory « reception ».

1) On the machine set to receive, under Windows :

```
'AROWRecv.py -r -f reception -a 192.168.2.3 -p 9876'
```

under Linux/UNIX :

```
'python AROWRecv.py -r -f reception -a 192.168.2.3 -p 9876'
```

2) On the transmit machine, under Windows

```
'AROWSend.py -s source -a 192.168.1.2 -p 9876 -b'
```

Under Unix/Linux:

```
'python AROWSend.py -s source -a 192.168.1.2 -p9876 -b'
```

Option -b permits transmission of files in a loop with a user definable pause between loops. This ensures that high-side files are constantly updated.

Note the different syntax for Windows and Linux.

Running from binaries

Under Windows, create a shortcut to the executable and add parameters, then run the shortcut.

e.g.

```
[path to]\AROWSendX86.exe -b -S TestSend -a 192.168.1.2 -p 9876 -l 1000000 -P 120 -R 1
```

```
[path to]\AROWReceiveX86.exe -r -f e:\\receive -a 192.168.2.3 -p 9876
```

Under Linux, just add the parameters

eg. ./AROWReceive -r -f e:\\receive -a 192.168.2.3 -p 9876

Options

AROWBFTP requires command line parameters to be entered for its configuration. If no command line is provided, the contents of an .ini file may be used to set parameters. The set of options is described here.

To see help for options : enter AROWSend.py -h or AROWRecv.py -h

Usage: AROWxxxx.py [options]

Mandatory options are IP Address and Port number and a source/receive directory.

Options:

Common (AROWReceive and AROWSend)

Option code	Meaning	Notes
-h, --help	Help	Show help message and exit
-b, --loop	Cycle continuously	Loop the sending files. In this mode, the designated sending directory will be regularly scanned at an interval determined by the -P parameter
-a	address	IP address : IP or machine (host) name can be used
-p	port	TCP port (the data address/port must be the same as set on AROW using the setup utility)
-d	debug	Debug mode, causes extra print statements to be sent to the console and log
-c	continue	Current file recovery see 'Recovery from File Interruption'
-u	UDP Streaming port	set the port to be used in UDP streaming mode

-t	TCP Streaming port	Set the port to be used in TCP streaming mode.
-w	Browser port	Set the port to be used for browser access (default 8080)
-m	Multicast Group Address	Set the address to be used to multicast - default 224.1.1.2
-n	Multicast Port	Multicast stream source port – default 5001 send, 5002 receive
-i	TTL value for Multicast Streaming	TTL value to set for Multicast reception (default 1)

AROWSend Only

-R, -- redundancy	Redundancy	Set the number of times each file is transmitted after adding to the send folder or subsequent modification, default is 1 (no redundancy).
-P	Pause	Pause between loops (in seconds). This should not be too frequent, to allow data to be processed on the receiving machine and to limit unnecessary network traffic. The pause depends on the amount of data normally expected to be sent. At GBE rates, for example, 1 GB of data will take over 10 seconds.
-s	synchro	Synchronise the tree. There are two variants of this. -s causes the entire contents of the send folder, including sub-folders and files, to be sent. However if files are removed, the delete instruction will not be sent until a week has elapsed. Note that the interval for this is hard-coded ie deletions will only be transmitted weekly.
-S	Synchro Strict	Synchronise the tree with deletions, causes the entire tree structure under the send directory to be sent and will also allow deleted file information to be sent, causing the receive directory to also delete its files (subject to the -x option on AROWReceive). Use this option to maintain an exact folder copy of the low-side on the high-side of the diode

-e, --send	emit	Send the file, a single send of the named file will occur, then the script will exit
-l	limit flow rate	Rate limit (Kbps). The flow rate across AROW can be limited to deal with heavy traffic situations or very large files etc. AROW includes a substantial hardware buffer but depends on the receiving machine TCP/IP connection performance to remove data at a rate greater than or equal to the send rate.
-j	Control the levels to be transmitted	Very high capacity transfers can be very resource intensive since a directory tree structure is maintained in memory. Setting the level option allows AROWBftp to create multiple sending trees, each separately maintained in temporary file storage. A level of 2 to 5 covers most scenarios. See separate section.
-o	Offline delay	The time (in seconds) between a file being deleted on the low side and that deletion request being sent to the high side –default 7 days

AROWReceive only

-x --nodelete	Prevent deletion	Turns off automatic deletion on the receiving side. Note – deletions are sent to maintain an exact tree structure between low and high sides. Use of this option invalidates this.
-f --folder	Receive folder	AROWReceive destination directory path – use the fully qualified path or the directory will be created under the current working directory. If the directory does not exist it will be created with the same permissions as the current user.
-g --temp	Temp folder	Set the temporary folder path – defaults to the system temp folder

Initialisation file

After the first run using command line parameters, an initialisation file is created that stores the software start parameters. This can be edited separately or copied as a backup to restore settings. Note that the INI file will be used if no command line parameters are given. Subsequent software releases may remove the command line parameter option completely

Compensating for data losses

Since the link is unidirectional, the transmitter does not receive any acknowledgment and it therefore cannot be warned in case of data losses on the receiving side. TCP is a connection-oriented protocol and would normally re-try if data were not successfully received. AROW includes a substantial hardware buffer to support re-tries but in the event of buffer saturation data will be lost. In this case the only information available is that 1 or more frames were not successfully received.

The AROWBftp programs also includes FIFO queues to decouple the packet transmission from the processing, and also includes worker threads to background-process data without holding up transmission.

Here are various solutions employed to prevent data loss :

SuperFrame

Data is transmitted with a header of fixed size that contains a unique marker sequence and frame length indicator. On reception, this information can be used to determine the integrity of the data, to re-align data from a continuous stream to a data packet form and to perform Cyclic Redundancy Checksums on the data. Together with file name and date stamping, substantial information is available to ensure the correct reconstruction of files streamed across AROW.

Redundancy (Option -R)

Data is transmitted several times, in case it gets lost. Inside AROWBFTP this redundancy is implemented at each file level ; the files are simply re-transmitted with a limit of R tries .

It is important to note that the use of redundancy is secure, but it is better to avoid the need in the general case. If a file frame must be retransmitted to be received completely, the transmission rate is divided by R or more.

Note also that the program in receive mode does not need to receive all the file in one go – frames are stored and indexed in a temporary file, and this is used to reconstruct the file from frames even if they are received out of order, perhaps because

the receive program was re-started during a transmission, or because one or more errored frames were received. Of course, the redundancy setting (-R) is important here – if the file is transmitted only a couple of times, then the receive program may not get enough information to reconstruct the file in the event of errors.

Rate Limiting (Option -l)

The transmission flow should be limited to a value that does not permit loss of data on the first transmission, under the normal utilisation case of the receiving machine. The value of this limit depends therefore strongly on the machine quality, the operating system, the applications running, and the size of files transmitted and particularly the write speed of the receiving file system.

Too high a limit will result in buffer overruns(the data cannot be processed on the receive side quickly enough) giving data corruption and loss. The state of the receive side buffer can be interrogated using the supplied `contol_arow.py` script. See Control and Status section.

Processing Rate

The receive processing rate is very dependent on destination machine configuration and the type and number of files to be processed.

The Python receive script includes a (hard-coded) length-limited buffer (python deque). The value of this limit is set to typical usage. Increasing the deque length requires an equivalent increase in free system RAM, since swap memory is very slow by comparison. However if enormous files (> 128MB) are to be transferred, it may be necessary to increase the deque limit, which will require extra RAM.

Note too that the interval between files will need to be sufficient to allow processing to complete at the average rate limit of the receiving system.

The checking and copying process of files can take considerable processing and memory resources, so overall,

copying a file across the diode and checking it may take 1.5 to 2 times longer than a direct transfer might be expected to take.

Some experimentation is required to determine the practical rate limit of your receiving system, with the files and system resources you wish to use.

In general, the AROW diode will be much faster than data processing on the receiving device.

A typical value with a reasonable PC, a fast disk (>1Gbps write rate) and a dedicated GBE link is of the order of 500-600 Mbits/s . (set flow limit -l 600000).

To limit the transmission rate, use the option -l :

```
python AROWSend.py -s source -a  
192.168.1.3 -b -l 600000
```

Increase the IP stack Buffer size

This method permits an empirical solution to reduce the risk of data loss and therefore employs an amount of 'tuning' that comes with experience and running time. It is however dependant on operating system, and must be configured outside of AROWBFTP, and is not available to all platforms.

Load Sharing

You can use more diode links and divide up the files to be transmitted. Each link will require a dedicated ethernet interface. Since AROWBFTP is GPL-licensed, there is no limit to the number or instances that may be run.

Recovery from interruption

The synchronisation with file recovery mode (option -c) permits the stopping and resuming of tree synchronisation without having to retransmit all the data from the start.

The previously transmitted files are not transmitted again.

In case the high-side detects an incident that prevents good reception of data and the low-side continues the transmission of data, the recovery file must be corrected to take account of the

elements not received. This setting necessitates stopping the normal processes of transmission.

The Python script `xfl_reset.py` provides a means of automatic correction.

This script permits the changing of the status of files identified as from a particular date of transmission (all files identified as transmitted after such date will be considered for re-transmission) .

If a XML file is retrieved from the reception side, it is possible to determine the state of reception by comparing the received tree structure with this file. This option requires that the relative transmit path and receive path are completely identical. The correction script then identifies and corrects the state of files not received , and by using the recovery file. it will then either copy the file on the low side or use the list of files identified for correction via `xfl_reset` directly on the low-side.

NB : the removal of synchronisation files and restart of the transmitter allows the whole of the original tree to be transmitted again.

Hot file recovery

It is not necessary to restart AROWSend in order to incorporate changes to the file manifest. The program regularly scans the XML file modified by `xfl_reset.py` and resends files that have been so marked.

The `xfl_reset.py` script offers 4 modes of operation.

1. By date – all files transmitted since the entered date will be marked for transmission again
2. By file name
3. By comparison with a received file manifest. differences in the manifest are used to reset attributes to cause files to be re-transmitted
4. By regular expression comparison. Files can be compared to a mask or pattern and re-sent if the pattern matches, for example all *.doc files, or all text files containing 'AROW' in the title.

High-side Buffer overruns

The main buffer is 2Gbit deep. This is enough to store over 2 seconds worth of data at full gigabit Ethernet wire speed.

If data gets congested on the high side server, packets may be dropped. This is expected TCP behaviour and is dealt with by resending the lost packet. While data is being resent, the buffer will continue being filled with data from the optical connection. When lost data is resent, data queued up in the buffer can resume being sent out. The buffer will gradually recover to normal levels depending on network slack

When transferring data at maximum data rates, this recovery time will take a long time, possibly minutes. Too many resends at high data rates will cause the buffer to overflow, causing holes to appear in the data. Refer to the rate-limiting section for possible remedies.

High Capacity Transfers and Levelling

When moving very large quantities of file tree structured-data across the diode, host resources consumed can be quite extensive.

By large capacity we imply >100GB of data to be transferred.

AROWBftp works by maintaining a copy of the file structure, timestamp, crc signature and number of times sent in a XML tree structure. Normally this is maintained in memory, but as the source tree grows, perhaps with many millions of files, the RAM and temporary file storage required can also grow very large.

V1.3.0 of the software introduces the concept of directory levelling with the option `-j` and an integer qualifier (0-10).

When this is invoked, instead of the entire tree being scanned in one process, the tree is broken into levels and each level scanned. A temporary file is created for each level's tree and stored in the temp directory. As each level scan is completed, the tree is transmitted, then the program moves to other trees in the level and scans and sends each in turn. The program arranges that higher levels only send files down to the chosen level. So for example if a levelling value of 2 is chosen, only files at level 1 will be sent, then a new scan for each of the folders discovered at level 1 is made and these with their sub-folders and files are sent in sequential operations.

The consequence is that only a relatively small XML tree is held in memory at any one time and can be discarded after use since a file copy is made and stored for use in subsequent scan comparisons.

The disadvantage is that the overall scan-and-send process takes longer, since tree structures must be written to and read from disk rather than from RAM. It also requires that System Administrators know and understand the quantity and structure of data to be sent.

For very complex data structures with very many files, the maximum levelling should be used, for simpler structures or structures with not many files per level, then lower or no levelling should be used.

Here are some examples of structures and level recommendations. Note that the size of files is immaterial, this only affects the transfer time.

No. of Folders	No. of sub-folder levels	No. of files per folder	Likely RAM usage (w/o levelling)	Level recommended
1	1	<10000	50MB	1
1	3	<10000	500MB	2-3
1	10	<10000	5GB	5-7
50	10	>10000	>5GB	5-10

1000	100	<1000	>5GB	3-5
------	-----	-------	------	-----

For any particular situation, monitoring the send server resource usage will indicate whether the level settings are appropriate.

File System Restrictions

It is important to note that the underlying file system plays a crucial role in determining the order and availability of files to be transmitted. In particular, large file trees require a significant amount of time to index from power-up before the index is made available to AROWBftp. It may be the case that the program will miss files to be sent if they have not been already indexed by the host file system. Make sure that the host system is fully ‘settled’ before trying to run large capacity transfers. If you can’t see the files in the host file system (using a command line or visual such as Windows Explorer or Nautilus) then AROWBftp will not see them either.

File Symlinks and Support Applications

Obviously it is not always practical or desirable to move or copy source or destination folders to the AROWBftp scanned directories. The answer to this is to use symlinks (Symbolic Links). These are links to files or folder structures that carry information about those files allowing the link to be treated as if it were the actual files/folders.

Symlink implementation varies on each operating system so you will need to consult documentation on your os to determine exact procedures.

Windows NTFS has had symlink support since Windows Vista, and this has been improved and expanded from Windows 7. Linux has always used symlinks.

A type of symlink is also available on Windows XP, called *junctions*. These can be tricky and quite dangerous to data since direct deletion will not only delete the link but also the files that were pointed to. Use with care, and manage the creation and deletion of junctions with a tool like Winbolic.

Placing symlinks in your send and receive folders means that AROWBftp can scan and reconstruct folders and files as if they were actually in these folders.

Website viewing

Websites are by design interactive and require the user to ask for data and information via their browser. Clearly this is not allowed or available from a network protected by AROW, so a different strategy must be employed.

For websites with mostly static content, offline web cacheing tools are appropriate. Tools such as HTTrack, available for most operating systems, allow websites to be downloaded as a tree structure, creating files with all the appropriate links on a local storage system, thus creating a browseable offline cache.

These tools necessarily require some administration, to set up the levels of browsing, provide regular refreshes and so on, but when used with AROW enable users on the protected network to browse most of the functionality of a website, although of course no interaction that requires contacting the original site can take place.

Simply create a link to the cached files, and AROW will transfer them to the protected network. They can then be used to reconstruct and populate a proxy browser cache or simply browsed directly with the browser in offline or file mode.

TCP Streaming

AROWBftp includes the ability to connect to a TCP stream and mix-in TCP data with file data.

AROWBftp acts as a server on each side of the data diode, allowing client streams to connect directly.

By default, AROWSend uses port 10002 to accept data from TCP sources and AROWReceive uses port 10000 to accept clients to output TCP data.

A system of priority queueing is used to give preference to stream frames over file and heartbeat frames, however processing overhead necessarily means that TCP stream rates are more restricted than in Direct Streaming mode.

File transfer is thus delayed when TCP streaming is used.

An example of the use of this function using a webcam is as follows:

If you have a webcam running under Direct Show, install ffmpeg and then try this from a command window:

```
'ffmpeg -list_devices true -f dshow -l dummy'
```

to get a list of the direct show names for your video and audio devices.

Run AROWSend in the normal way.

From a command line on the send server:

```
'ffmpeg -f dshow -l video="<yourcameraname>" -f mpegts  
tcp://127.0.0.1:10002'
```

Run AROWRecv in the normal way.

From a command line on the receive server:

```
'nc -vvn 127.0.0.1 10000 | mplayer -'
```

After a short delay (<10secs) the webcam image should appear on mplayer.

Note that for the purposes of this example we have shown a Windows server as the sending host and a Linux server as the receiving host, with appropriate syntax for each. (Under Linux simply find the appropriate camera device under /dev and change the ffmpeg syntax appropriately).

If changed files are detected by AROWSend, these will be transmitted in the usual way, but any Stream frames detected will be given priority, thus maintaining a 'real-time' data flow.

Stream data can be disconnected or interrupted at any time without interrupting AROWBftp's file transfer operation.

However, high volumes of Stream data will delay file transfers so causing them to take longer.

UDP Streaming

AROWBftp includes the ability to connect to a UDP unicast stream and mix-in UDP data with file data and other stream data.

AROWBftp acts as a server on each side of the data diode, allowing client streams to connect directly.

By default, AROWSend uses port 8002 to accept data from UDP sources and AROWReceive uses port 8001 to accept clients to output UDP data. These can be changed using command line parameters and/or the configuration .ini file.

A system of priority queueing is used to give preference to stream frames over file and heartbeat frames, however processing overhead necessarily means that UDP stream rates are more restricted than in Direct Streaming mode.

File transfer is thus delayed when UDP streaming is used.

The example for TCP streaming above can be used with UDP, simply substitute udp for tcp, and use the appropriate port settings.

Multicast UDP Streaming

AROWBftp includes the ability to join a multicast group and encapsulate the data, then create a multicast stream on the receive side.

In order to do this, the application creates a multicast server in AROWSend, taking command line parameters that define the multicast address, port and TTL value. (see options table).

Multicast packets are serialised, queued and sent across the AROW data diode, then de-serialised and presented to a new multicast address as set by the AROWReceive command line parameters.

For testing purposes, tools such as iperf and socat are useful, and for demonstration, VLC or similar media players.

Multicast Testing

The Linux utility iperf can be used to generate multicast data as follows:

```
iperf -c 224.1.1.1 -u -T 1 -t 3600 -i 1
```

This will generate repetitive 0-9 decimal data bytes (0x30-0x39) with a 36-byte header, totalling 1470 byte datagrams. The multicast source address is 224.1.1.1 (set AROW send option -m to be the same) and uses the default port of 5001. The TTL value is 1 and the duration of generation is 3600 seconds. See iperf docs for more options and information.

AROWSend joins the group automatically and formats the datagrams for transmission across AROW.

On the receive side, socat can be used to receive and display or dump the data.

`socat udp4-recvfrom:5002,ip-add-membership=224.1.1.2:0.0.0.0,fork - > test.out`
assuming AROWReceive is set to its default address and port.
Output is directed to the `test.out` file which can then be checked for accuracy against the source.

A note for Windows users: some of the network tools mentioned here have been ported to Windows, others can be invoked from cygwin, a linux like environment for windows.

Multicast Demonstration

A non-quantitative demonstration of Multi-casting can be made using the [VLC media player](#) and a mp4 or similar video file.

1. On the sending side ,in the Media menu, choose “Stream”
2. In the Open Media dialog file tab, click “add” and choose the mp4 file you want to stream and click “Open”
3. At the bottom, click the “Stream” button
4. This opens the “Stream Output” dialog showing the source file you have chosen. Click Next to set destination.
5. In “Destinations”, choose “RTP /MPEG Transport Stream” and click the “Add” button
6. In the “Address” box, enter the required multicast address (eg 224.1.1.1) and set the port to 5001
7. In transcoding options, choose the appropriate settings for your video and PC’s codecs e.g. “Video H.264 + MP3 (MP4)”. Click the options (screwdriver and spanner) button immediately to the right of the dropdown. In encapsulation, choose MPEG-TS. In video codec, optionally set the bitrate to 4000kb/s
8. Once the options are set, click “Save”. Then click Next for “Option Setup” and select “Stream all elementary streams” then click stream.

To view the stream on the receive side, open VLC media player, with AROWReceive running.

1. Choose Media/Open Network Stream

2. In address, enter rtp://@221.1.1.2:5002 – choose the correct address and port you entered when setting up AROWReceive.
3. Click “Play”

AROWBftp V2 and Unicode

Version 2 of AROWBftp uses Python 3's native Unicode support to enable the correct transmission and rendering of complex font structures.

AROWBftp runs in a local command window (Windows) or Terminal program (Linux). For the correct rendering of complex languages such as Chinese, Korean, Japanese, etc., in some versions of Windows, it is necessary to ensure that the appropriate language packs are installed.

For mixed language systems ,e.g. English and Korean, it is necessary to enable Unicode support explicitly, by making all the appropriate fonts available (e.g. BatangChe etc.) by going to Control Panel/Appearance and Personalization/Fonts/Font settings and unchecking the 'Hide fonts based on language settings'. Also open the Region and Language window, select the 'Administrative' tab and select your desired language in the 'Language for non-Unicode programs' section. This will enable Windows Command line console to render e.g. Korean characters.

Of course if your locale is set to e.g. Korean, this should already be in place.

Direct Data Streaming

It is possible to stream data directly through AROW without using AROWBftp. See AROW-MAN-0601 for connection information. Direct (TCP, UDP unicast and multi-cast) connection is supported

on unprotected and protected sides of AROW (TCP-only on High Availability AROW 0601). Simply connect as a client to each server socket. Data received on the unprotected low side will be made available on the protected high side as soon as the high-side client connects. AROW-S is optimised for this process.

Note however that on AROW-S there is no intrinsic TCP rate control – the high side cannot communicate with the low side to limit transmission rate. Consequently, the internal buffer will overflow if the low-side data rate exceeds the high-side data rate, or if data in excess of the available buffer size is streamed into the low-side.

However, there exist a number of streaming control programs, especially on Linux/Unix systems that can be used to manage TCP streams. In some cases these useful tools have been ported to Windows and Mac as well.

Netcat is the ubiquitous and extremely powerful tool that can be used directly to stream data across AROW, and is available as a Windows port, Ncat.

A simple example of the use of this tool is as follows:

A windows source on the unprotected, low, side of AROW can send data to a Linux sink on the protected, high, side of AROW: from a terminal on the high side

```
'nc -<options> <source ip address> <source port> | mplayer -'  
e.g. 'nc -vvn 10.0.0.3 9876 | mplayer -'
```

the high side will wait for standard input

then from a command window on the low side

```
'type <some file> | ncat <source address> <port>'  
e.g. 'type audio.mp3 | ncat 192.168.2.9 9876'
```

the file will be streamed and played out on the high side. Note that there is no rate limiting in this example.

In this case, an alternative is to use ffmpeg.

This tool contains a number of very useful functions, including Transport Layer streaming and Rate Limiting.

With the same setup, we can stream video from a file or other source such as webcam or realtime netcast across AROW.

Mplayer can accept and detect video/audio content automatically. So on the high side, as before, use netcat to accept data from the AROW socket connection and pipe to Mplayer:

```
'nc <-options> 'source ipaddress> <source port> | <application>
```

```
e.g. 'nc -vvn 10.0.0.3 9876 | mplayer - '
```

then from a command window on the low side (and with ffmpeg.exe in the source folder)

```
'ffmpeg <options> <input> <output filter> <output destination>'
```

```
e.g. 'ffmpeg -re -i video.mp4 -f mpegts tcp://192.168.2.9:9876'
```

The options we have chosen rate-limit the source to (an approximation) of the original source, declare the input as video.mp4, force the output to be converted to a mpeg transport layer, and then sent as tcp to AROW's ip address and port.

The same principle can be used for virtually any audio/video source.

Webcam

If you have a Webcam running under Direct Show, try this:

```
ffmpeg -list_devices true -f dshow -l dummy
```

to get a list of the direct show names for your video and audio devices.

Then :

```
'ffmpeg -f dshow -i video="video device name": audio="audio device name" -f mpegts tcp://192.168.2.9:9876'
```

will send your video device pictures across AROW to mplayer.

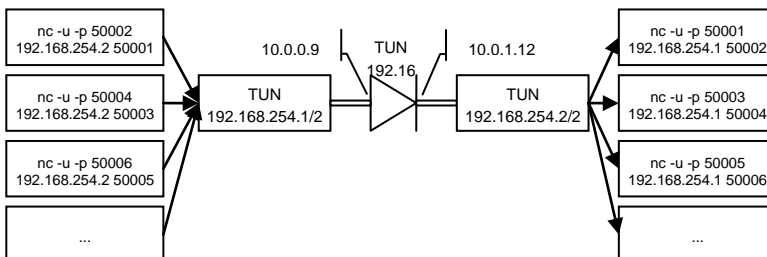
Of course in practise these simple examples would probably be replaced by a TCP stream aggregator such as Data Turbine or similar, but the principle of a simple TCP connection holds good.

Note that for mixed sources (for example, files and tcp streams) it is better to use AROWBftp which includes rate management as well as error detection and stream identification and reconstruction.

If you wish to write your own TCP stream application, be aware that you **MUST** implement application layer framing and data length management. See the Python code for how AROWBftp accomplishes this.

UDP over TCP (Linux Only)

Multiple parallel UDP streams of data can be sent through AROW using *socat* to create virtual network interface ports using tcp socket connections to AROW as the "wire". This is especially useful for the High Availability version of AROW that does not support UDP directly.



Socat is a Linux application that includes tunnelling a network interface over a TCP socket connection.

Socat can be found here: <http://www.dest-unreach.org/socat>

Assuming the low side host has a connection to the 10.0.0.0/24 network, create the low side of the tunnel:

```
# socat -d -d TCP:10.0.0.9:9876 TUN:192.168.254.1/24,up
```

Assuming the high side host has a connection to the 10.0.1.0/24 network, create the high side of the tunnel:

```
# socat -d -d TCP:10.0.1.12:9876 TUN:192.168.254.2/24,up
```

(Note, -d -d is optional and is useful for confirming a network connection to AROW.)

Now we should have a "tun" interface in "ip addr" or "ifconfig" of each of the hosts.

Sending UDP over that tunnel is as simple as opening the listening program on the high side host, setting the IP address to 192.168.254.1 and setting source and destination ports as wished. On the low side host, start the sending application with the destination address as 192.168.254.2 and source and destination port set appropriately. For instance:

On the high side host:

```
$ netcat -u -p 50002 192.168.254.1 50001
```

And on the low side host:

```
$ netcat -u -p 50001 192.168.254.2 50002
```

Now any packets sent from the low side netcat instance will be received by the netcat instance on the high side. Packets sent from the high side will be silently dropped.

Unfortunately, this option is not available under cygwin so this technique cannot be used on Windows installations.

Large File Processing

There is no limit to file size that AROWBftp can process (except of course the limit of the host operating systems). Speed of processing is determined by the host system(s) capabilities, especially file system read and write times. The checksum calculation for very large files can take a long time and will delay file transmission.

Large files will therefore tend to hog system resources. The effect of this on users is increased latency- changes to other files may appear to be delayed if very large (Multi-Gigabyte) files are being transmitted. See also the section on [large capacity transfers](#) and levelling.

When first establishing the system, monitor the logs and use the web interface to check the actual data rate achieved, and set the rate limit appropriately.

Virtual Machines

AROWBftp and the AROW system can be operated using a virtual machine. However the nature of the network connection virtualisation means that performance will be restricted. This is most noticeable with large contiguous data, (files or streaming) when the throughput performance can be degraded by 2 to 3 times.

As an example, using identical hardware and operating with VirtualBox [™] hosting 32-bit Ubuntu Linux, actual throughput was measured to be >900Mbps, virtual throughput < 300 Mbps

6. CONTROL AND STATUS

In this Section

INTRODUCTION	6-1
CONNECTING TO AROW'S CONTROL AND STATUS PORT	6-2
PROTOCOL	6-2
COMMANDS	6-3
STATUS REQUESTS	6-4
CONTROL AND STATUS PYTHON SCRIPT	6-5
SETTING THE DATA PORT IP ADDRESS	6-5
RESETTING THE TCP STACK.	6-6
CONTROL AND STATUS MESSAGE FORMAT	6-7
MEMORY MAP	6-7
REGISTERS	6-8
GRAPHICAL DATA MONITORING	6-15
LOGGING	6-17
LOGFILE MANAGEMENT	6-18

Introduction

Control is via an Ethernet interface, so that controlling software can be run from a PC anywhere on a network to control and interrogate the status of the unit.

The external interface is 10BASE-T/100TX Ethernet with auto sensing.

Any application wishing to control AROW must open a TCP/IP socket connection with port 10001 of the control interface. Only one socket connection can be made at a time so when the application has finished, it must release the port. Any application attempting to connect to a connected port will be refused.

All AROW messages are 9 Bytes long: The first Byte is the Start of Message marker byte, the second read or write flag byte, the 3rd and 4th are the addressing word, the 5rd to 8th are the data payload bytes and the last byte is the end marker byte.

Any command sent is reflected back by AROW to confirm that the command has been received and acted upon.

There is no unsolicited status. All status must be collected by sending a status request command.

Connecting to AROW's Control and Status Port

Connection to AROW's control and status Port are made with a standard TCP/IP socket. This socket connection is to port 10001

Only one Application can open a socket at a time so if a second application attempts to create a socket connection it will be refused.

When the application has finished controlling AROW, it should release the socket connection so that another application can have access to it.

Protocol

Messages between AROW and the controlling application should follow a request/response model. The application should request that a command be executed or that a status word be collected. AROW responds with the executed command or the status requested.

Commands

Only one command should be sent at one time. The application should wait until the response to the command has been received before moving on to the next command or status request.

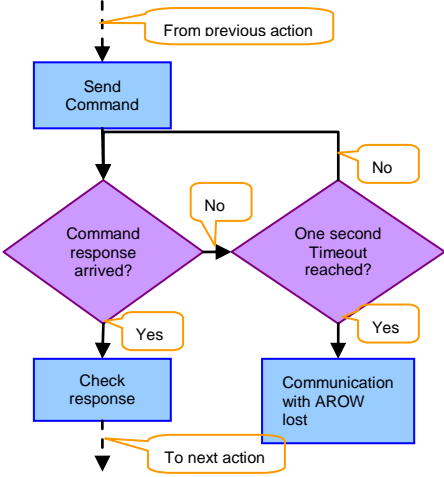


Figure 6-1: Command Flow Diagram

Status requests

Only one status request should be made at one time. The application should wait until the status of the request has been received before moving on to the next command or status request.

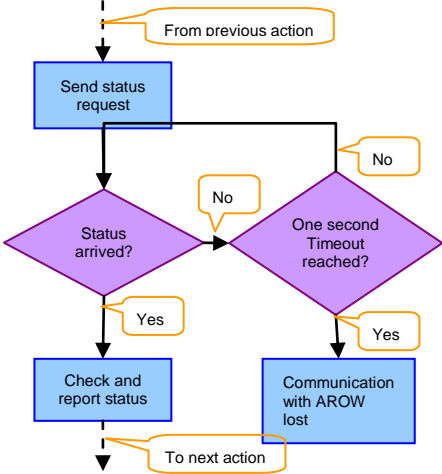


Figure 6-2: Status Flow Diagram

Control and status Python script

A python script is provided as an easy starting point to setting up and controlling AROW.

use this script to set the data port IP address, collect various status and perform resets.

The following is the "-h" output.

```
./control arow tcp.py -h
usage: control arow tcp.py [-h] [--version] [-t] [-o] [-m]
                        [-D] [-s] [-i [IPADDR]] [-M]
                        [-a ADDR] [-d DATA] [-H HOST]

Controls and collects status from AROW

optional arguments:
  -h, --help            show this help message and exit
  --version            show program's version number and
                        exit
  -t, --tcp_rst         reset the tcp stack
  -o, --optics rst     reset the optical link
  -m, --hightide rst   reset the ddr buffer high tide mark
                        and the buffer overrun counters
  -D, --defaults       set all NV registers to default
                        values
  -s, --status         collect status
  -i [IPADDR], --ipaddr [IPADDR]
                        Sets (or gets when no IP address
                        provided) the IP address of the data
                        port. dotted ip address string
  -M, --mac            gets the MAC address of the data
                        port.
  -a ADDR, --addr ADDR hexadecimal (`0010` for example)
                        representation of the address of the
                        register to access
  -d DATA, --data DATA hexadecimal (`00100000` for example)
                        representation of the register to
                        write. If no -d is specified, a read
                        access is implied
  -H HOST, --HOST HOST IP address of control port.
                        Default is '192.168.2.220'
```

Setting the data port IP address

Assuming AROW's control port is 192.168.2.221,

```
./control arow tcp.py -H 192.168.2.221 -i 10.0.0.51
```

will set the data port IP address to 10.0.0.51.

Resetting the tcp stack.

If AROW's data port ever becomes unresponsive, its TCP stack can be reset with the following command.

```
./control_arow_tcp.py -H 192.168.2.221 -t
```

Control and Status message format

All AROW messages are 9 bytes long: the 1st byte is the "Start of Message" marker, the 2nd byte is the read/write flag byte, the 3rd and 4th bytes are the register address, the 5th to 8th are the Data payload bytes and the last byte is the End of message marker.

Message field name	SOM	Rd_wr	addr	Data	EOM
Number of bits	8	8	16	32	8
value	0x53	0x00 for read, 0x01 for write	Register address	Data	0x0d

Bytes are sent Least significant bit first.

Where a field is more than 8 bits, the MSbyte is sent first.

Registers that are not implemented will return all 1s in the data field.

Any message that doesn't conform to the format above will be ignored and discarded.

Memory map

Address range	Type
0x0000 to 0x0FFF	System control and status (0x0100-0x010F is Non volatile)
0x1000 to 0x1FFF	TCP/IP stack (0x1000-0x100F is Non volatile)
0x2000 to 0x2FFF	MAC core (0x2400-0x241F and 0x2700-0x270F are non volatile)
0x3000 to 0xFFFF	Reserved

Registers

System registers

Table 6-1 Part Number register (0x0000)

Valid from firmware V0R1

Bit	field	mode	Default Value	Description
31:0	Part_num	RO	0xa0a00112	Part number field

Table 6-2 Version and Revision register (0x0001)

Valid from firmware V0R1

Bit	field	mode	Default Value	Description
31:16	ver_num	RO	0x0000	Version number field
15:0	Rev_num	RO	0x0001	Revision number field

Table 6-3 Serial number low (0x0002)

Valid from firmware V0R1

Bit	Field	mode	Default Value	Description
31:0	Ser_num_L	RO	0x89abcdef	Low Dword of serial number

Table 6-4 Serial Number high (0x0003)

Valid from firmware V0R1

Bit	Field	mode	Default Value	Description
31:0	Ser_num_H	RO	0x01234567	High DWord of Serial Number Field

Table 6-5 commit number (0x0004)

Valid from firmware V0R2

Bit	Field	mode	Default Value	Description
27:0	commit	RO	0x0000000	Revision system (git) commit number

Table 6-6 hardware status (0x0005)

Valid from firmware V0R5

Bit	Field	mode	Default Value	Description
0	Module is High	RO	-	Module is in a high side slot. 0 = module is in a low side (dirty) slot.
1	Module in live slot	RO	-	Module is in a live slot. 0 = module is in a backup slot
2	Module is live	RO	-	Module is currently the live module. (even if it is in a backup slot). 0= the other module is a live module and we are backup.
3	Live-backup link is up	RO	-	Link between live and backup modules is up.
16	Stand-alone module	RW	0x0	1 = Module is live and is for use in non redundant systems. 0 = Module to be used in a redundant system; it expects another module alongside it and will produce or receive both a live and backup stream of optical data.
17	Reset live/backup link	RW	0x0	Reset the Live/backup duplex link

Table 6-7 optical link (0x0100)

Valid from firmware V0R4

Bit	Field	mode	Default Value	Description
0	Live Link up	RO	-	Live Optical Link has been established
1	Backup Link up	RO	-	Backup Optical Link has been established
16	Reset Live Optical link	RW	0x0	Reset the live optical link
17	Reset backup Optical link	RW	0x0	Reset the backup optical link
others	reserved	RO	0x0	

TCP/IP stack registers

Table 6-8 MAC address Low (0x1000)

Valid from firmware V0R1

Bit	Field	Mode	Default Value	Description
31:0	MAC_ADD_low	RO	0xC25D-D000	LOW DWord of Mac Address Field for data port

Table 6-9 MAC address Low (0x1001)

Valid from firmware V0R1

Bit	Field	Mode	Default Value	Description
15:0	MAC_ADD_high	RO	0x0050	High Word of Mac Address Field for data port

Table 6-10 IP address (0x1002)

Valid from firmware V0R1

Bit	Field	Mode	Default Value	Description
31:0	Ip_address	RW	0x0000	IP address of data port

Table 6-11 tcp_ip controls and flags(0x1003)

Valid from firmware V0R1

Bit	Field	Mode	Default Value	Description
0	TCP est	RO	0x0	TCP socket connection has been established with the data port.
1	GBE SFP present	RO	0x0	Gig Ethernet SFP module has been detected
2	Link-up	RO	0x0	Gig Ethernet link is up
3	DDR full	RO	0x0	Buffer is full - loosing data.
16	Reset tcp/ip stack	RW	0x0	Reset the tcp/ip stack. Normal operation is 0x0
17	Reset fifo overrun byte cnt	RW	0x0	Reset the byte counter that counts the number of bytes that were lost because of a Fifo full condition.
18	Reset ddr overrun byte count	RW	0x0	Reset the byte counter that counts the number of bytes that were lost because of a buffer full condition.
19	Reset high tide mark	RW	0x0	Reset the DDR buffer high tide mark
others	Reserved	RO	0x0	

Table 6-12 FIFO overrun bytes counter (0x1010)

Valid from firmware V0R1

Bit	Field	mode	Default Value	Description
15:0	Fifo Overrun bytes counter	RO	0x0000	Number of bytes that have been lost because of a fifo full event. Pegs at 0xFFFF. Reset with "reset FIFO overrun byte cnt" register

Table 6-13 DDR overrun bytes counter (0x1011)

Valid from firmware V0R1

Bit	Field	mode	Default Value	Description
31:0	DDR Overrun bytes counter	RO	0x00000000	Number of bytes that have been lost because of a buffer full event. Pegs at 0xFFFFFFFF. Reset with "reset DDR overrun byte cnt" register

Table 6-14 bytes in buffer (0x1012)

Valid from firmware V0R1

Bit	Field	mode	Default Value	Description
31:0	Bytes currently in buffer	RO	0x00000000	Number of bytes currently in the buffer. Maximum is 0x07FFFFFF in the prototype, 0xFFFFFFFF in the final product.

Table 6-15 high tide(0x1013)

Valid from firmware V0R1

Bit	Field	mode	Default Value	Description
31:0	High tide mark	RO	0x00000000	Number of bytes in the buffer when the buffer was at its fullest. Maximum is 0x0FFFFFFF . Reset with "reset high tide mark" register

MAC core registers

Provides network connectivity statistics, refer to Xilinx UG777, Base address is 0x2000

Graphical Data Monitoring

From version 1.1.0, the AROWSend and AROWReceive scripts incorporate a simple HTTP server that can be used to access static and dynamic information about the software.

The script StatsServer.py in conjunction with a XSLT stylesheet and javascript file allows a browser to connect to the application and display statistical information.

This includes setup options chosen at script launch, a heartbeat monitor, and a graphical representation of throughput rates.

AROW

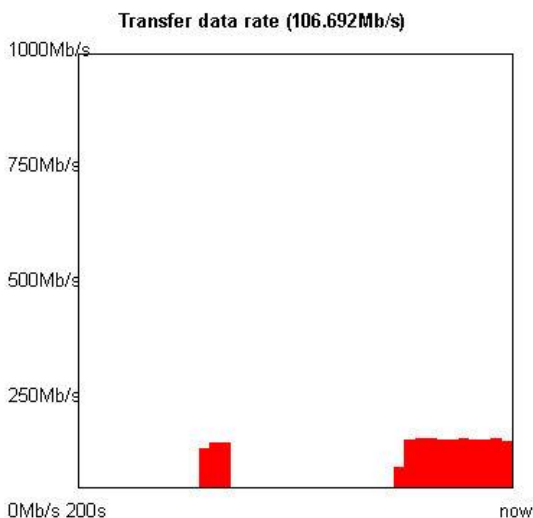
Setup

Address	Port	UDP Port	TCP Port	Heartbeat
192.168.2.39	9876	8002	10002	37 : 10:01:21

Modes

Recovery	Debug	Flow Limit (kbps)	Loop	Pause
False	False	1000000	True	50

Throughput



When either script is running, point a browser at <http://localhost:8080> to obtain a screen similar to that shown here. The port number can be set as a startup parameter.

The Setup section shows the IP addresses and ports from the start parameters or ini file plus a heartbeat display (sent or received depending on the script accessed). This gives a regular (5 seconds by default) indication that the script is running and communicating with AROW.

The Modes section shows the static modes selected at launch time.

The Throughput section includes a current rate value and a historical graph of throughput in the form of a moving histogram.

On the Receive side, file synchronisation is tracked, showing a quick display of the number of files , whether they are synchronised to the low side, if any are missing or if there are extra files on the high side that are no longer on the low side.

Logging

AROWBFTP maintains an application log detailing major events and status information. These are AROWSend.log and AROWReceive.log, found in the launch directory of the application. Note that debug mode creates many log entries, creating a large file quickly.

The log shows any missing files that have been detected.

AROWBFTP uses the python Logging module. This can be readily extended and customised to provide logging to syslog, smtp, http etc, by the addition of standard handlers, as well as varying the level of logging according to destination.

Main features in the log:

- all events are timestamped
- all events are levelstamped
- plain text to file

Common Logging Events

- Application configuration and version information
- Source and destination information – enables multiple instances
- Successful and lost file information
- Scan timing
- Heartbeat information

You can configure the level of critical events, for example lost files, to trigger a syslog or email alert. Contact us for advice on how to do this.

Logfile Management

The logfile is set to a maximum of 1MB, with 3 backup files on a rotating basis. This is set in the main part of the script and can be easily changed.

7. SUPPORT

In this Section

WHAT TO DO IF YOU HAVE A PROBLEM	7-1
SERVICING, MAINTENANCE AND REPAIRS	7-1
IF YOU NEED SUPPORT	7-1
SUPPORT REQUESTS	7-2
RETURNS	7-3
SOMERDATA CONTACT INFORMATION	7-4
END-OF-LIFE DISPOSAL	7-4
WASTE ELECTRICAL & ELECTRONIC EQUIPMENT (WEEE)	7-4

What to do if you have a problem

Firstly, please ensure that you have followed the installation, connection and operation instructions in the appropriate User Guide.

Also, check the Troubleshooting section (where appropriate) to eliminate common problems.

Servicing, Maintenance and Repairs

Please contact your supplier or SomerData for all questions relating to maintenance and repairs.

Any unauthorised attempt to open, modify or otherwise repair the product will invalidate the SomerData warranty and may result in the product being left in an irreparable condition.

If you need Support

For warranty, technical and application support issues, you should initially contact your supplier to check whether your SomerData product is covered by warranty, extended warranty or maintenance contract.

At SomerData, we will make our best efforts to provide prompt and friendly support by phone, fax and e-mail.

Diagnosing a problem will require your co-operation and we expect you to provide a detailed description of the problem in the form of a detailed Fault Report.

Support Requests

When contacting SomerData for support, please provide as much information as possible about the problem or issue for which you require assistance.

We will be able to deal with your request more efficiently if you provide the following details (where available) in your Fault Report:

- Part Number or Model Number
(for example AROW-0112)
- Serial Number (for example 2018/01/001)
- Software Version (for example 2.0)
- Details of any symptoms or error messages
- Diagnostics information (if available)
- Sequence of events/actions or other circumstances that triggered the problem
- How you are able to identify that there is a problem
- How you have been able to measure, log or otherwise display the problem
- Details of the host PC (if appropriate) including: operating system; hardware configuration; other software applications (e.g. analysis or processing programs) that are running at the same time
- Sample data files (if appropriate)

When we acknowledge your support request, you will be given a *Support Tracking Number* (STN), which should be quoted in all further correspondence relating to that specific support request.

Returns

Please do not return any products to SomerData without first contacting SomerData and obtaining a Return Merchandise Authorisation (RMA) Number. Goods returned without a RMA will be subject to a handling charge and returned unopened.

This will ensure that the processing of any repair or upgrade is handled efficiently and in accordance with any agreed action.

If the SomerData product is under warranty, repairs are free-of charge. If not, there will be a repair charge, which will comprise an initial evaluation fee and quotation, followed by repair and parts (if authority is given to carry-out the repair).

Pack the item in its original packaging. If the original packaging is not available, it must be packed in such a way to avoid transit damage. Damage sustained in transit is not covered under warranty.

Returned goods should be accompanied by documentation that indicates the RMA Number along with a detailed fault report and contact details (name, organisation, phone, fax and e-mail).

Mark the RMA Number on the outside of the package.

Ship the item by insured, prepaid carrier to the above address.

Items being returned from outside the European Community must be accompanied by a Commercial Invoice. This should include a description of the goods, value for Customs Purposes and state that the goods are being temporarily returned to the UK for repair. SomerData will not accept liability for UK importation costs resulting from inadequate documentation.

SomerData Contact Information

Address: Somerdata Limited
Underwood Business Park
Wells
BA5 1AF
England

Phone: UK 01179-634050
International +44 1179-634050

E-Mail: support@somerdata.com

Website: www.somerdata.com

End-of-Life Disposal

Your SomerData product may be returned to SomerData at the end of its life provided that the product is free from radiation or biological contaminants and that no other legislation forbids the return. The end user is responsible for all shipping costs and the Returns procedure should be followed.

Waste Electrical & Electronic Equipment (WEEE)

In the UK, Somerdata products may be recycled free of charge at any local authority recycling centre as long as the SomerData logo appears on the product and the following WEEE producer registration number is quoted: WEE/HA0074UR/PRO.



8. WARRANTY

In this Section

INTRODUCTION	8-1
WARRANTY: TERMS AND CONDITIONS	8-1

Introduction

This section describes SomerData’s Warranty Terms and Conditions.

Where the SomerData product has been supplied via an authorised Reseller, the Warranty and Support is between SomerData and its Reseller. Local Warranty and Support arrangements should be agreed between the Reseller and the Customer.

The following information is subject to change without notice.

Warranty: Terms and Conditions

General Warranty Terms – these may be superseded by specific contractual conditions per item or per contract.

SomerData warrants all goods supplied by it to be free from defects in material and workmanship, under normal use, care, storage and service, for a maximum period of twelve months from the date of delivery (per Incoterms 2010) by SomerData.

This warranty is limited to the repair or replacement, as SomerData may elect and at an establishment authorised by it, of such items as shall appear to SomerData, upon inspection to have been defective in material or workmanship.

All decisions relating to the validity and processing of Warranty claims shall be at the sole discretion of SomerData.

This warranty does not apply to normal maintenance service or to normal replacement of service goods.

Any claim under this warranty shall expire unless made in writing immediately after the appearance of a claimed defect.

This warranty excludes damage from incorrect installation, unauthorised modification, negligence, misuse or abuse or any item of equipment which has been serviced or worked on by anyone other than SomerData or its authorised representative.

SomerData will repair or replace, at its option, any product purchased from SomerData which, under normal conditions of use and service, proves to be defective in material or workmanship.

No charge will be made for labour or parts with respect to defects covered by this warranty, provided that the work is done by SomerData.

This warranty does not cover expenses incurred in the removal or reinstallation of any SomerData products, whether or not proven defective

Replacement or repairs furnished under this warranty are subject to the same terms and conditions of the original warranty.

9. NOTICES

In this Section

GENERAL INFORMATION	9-1
SOMERDATA AND THE ENVIRONMENT	9-1
CURRENT COMPLIANCE ACTIVITIES	9-2
RESTRICTION OF USE OF HAZARDOUS SUBSTANCES (RoHS)	9-2

General Information

Copyright © 2012 Somerdata Ltd. All Rights Reserved.

This publication is protected by copyright and all rights are reserved. No part of it may be reproduced or transmitted by any means or in any form, without prior consent in writing from SomerData.

The information in this User Guide has been carefully checked and is believed to be accurate. However, SomerData assumes no responsibility for any inaccuracies that may be contained in this publication.

In no event will SomerData be liable for direct, indirect, special, exemplary, incidental, or consequential damages resulting from any defect or omission in this manual, even if advised of the possibility of such damages.

In the interest of continued product development, SomerData reserves the right to make improvements in this publication and the products it describes at any time, without notice or obligation.

All product names mentioned herein are used for identification purposes only, and may be the trademarks or registered trademarks of their respective companies.

Somerdata and the Environment

SomerData is committed to design and introduce products that conform to applicable environmental legislation and standards.

One of our missions is to integrate environmental stewardship into the business of providing quality products, services, and customer support at the best value.

In order to achieve this, SomerData has established a strategic team to focus on the importance of meeting our environmental obligations in the design, manufacture and support of our products.

We have developed a broad appreciation of the impact of these directives on our entire business model, from technical processes for materials, to finished goods manufacturing.

Current Compliance Activities

The Company's current environmental compliance commitment has been structured to meet the following European Union directives:

- Restriction of use of Hazardous Substances or RoHS Compliance (EU Directive 2002/95/EC)
- Waste Electrical & Electronic Equipment or WEEE Compliance (EU Directive 2002/96/EC)

Our goal is to meet or exceed compliance obligations of these EU directives.

Restriction of use of Hazardous Substances (RoHS)

Somerdata has also established a RoHS qualification process to help ensure that products meet stringent reliability and quality requirements, as well as regulatory compliance requirements.

The maximum allowable hazardous substance at a homogeneous material level under the EU RoHS Directive is shown in the following table.

From 1st July 2006 all SomerData manufactured products used lead-free soldering

Substances	Maximum Concentration Values (ppm)
Lead and its compounds	1000
Mercury and its compounds	1000
Hexavalent Chromium (Cr+6)	1000
Cadmium and its compounds	100
PolyBrominated Biphenyls (PBBs)	1000
PolyBrominated Diphenyl Ethers (PBDEs)	1000

Declaration of Conformity

Name of Manufacturer: Somerdata Ltd.
Address of Manufacturer: Somerdata Ltd
1 Riverside Business Park
Bristol

BS4 4ED
United Kingdom

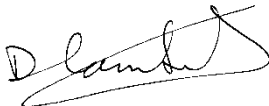
Equipment description: AROW Network Data Diode
Model: AROW-100

Conforms to the following Product Specifications:

Safety: IEC 950

EMC: 89/336/EEC EN55022 Harmonised Standard

The product complies with the requirements of the Electromagnetic Compatibility Directive 89/336/EEC as amended and the Low Voltage Directive 73/23/EEC and carries the CE marking accordingly.



Signed:
Position: Technical Director
Date: 19th June 2012

10. INDEX

Address for SomerData Support7-4

Configuration4-1

 Setting Control Port IP Address4-1

 Setting Data Port IP Address4-2

Contact Details7-4

Control and Status6-1

 Graphical Data Monitoring6-15

 Memory Map Format.....6-7

 Message Format6-7

 Python Script.....6-5

 Registers6-8

 Using Web Browser6-15

Controlling AROW

 Connecting6-2

 Protocol6-2

Customer Support7-1

Cython5-3

Declaration of Conformitya

Direct Data Streaming5-22

General Information9-1

Index10-1

INSTALLATION3-1

 Network Addressing.....3-2

 Preparation3-1

Introduction1-1

Large Files5-26

Logging6-17

 File Management6-18

Multicast UDP Data5-20

Notices

 Declaration of Conformitya

 General Information9-1

 SomerData and the Environment9-2

Operation5-1

 Command Line Options5-5

 Data Losses5-11

 High Capacity and Levelling5-15

 INI file5-10

 Recovering from Interruption5-13

 Software Installation.....5-3

 Streaming Data5-18

 Streaming Multicast UDP Data5-20

 Streaming UDP Data5-19

 Support

 Websites.....5-18

 Symlinks and Support Applications5-17

Product Description2-1

 Block Diagram.....2-1

 Introduction2-1

SomerData

 Contact Details.....7-4

SomerData and the Environment9-2

Support

 End of life disposal.....7-4

 WEEE.....7-5

SUPPORT7-1

UDP Data.....5-19

UDP over TCP -SOCAT5-25

Unicode.....5-22

User Guide Availability1-2

Virtual Machines5-26

Warranty8-1

 Introduction8-1

 Terms and Conditions.....8-1

WEEE7-5

What's in this User Guide1-1