Rockland Technical Note 051

Rockland ODAS v6 Data File Format

2026-01-12

Rockland Scientific International Inc.
520 Dupplin Rd
Victoria, BC, CANADA, V8Z 1C1
www.rocklandscientific.com

Copyright Notice

info@rocklandscientific.com
tel: +1 250 370 1688

Version History

Date	Description
2020-11-05	Initial version
2026-01-12	Added Bad Buffer content. Updated with data file versions up to 6.3. Changed title from Rockland Data File Anatomy to Rockland ODAS v6 Data File Format.

1 Overview

Rockland instruments generate several types of data files, see Table 1. This technical note only discusses ODAS version 6 (`.p`) data files created by the majority of Rockland instruments. These binary files are generated by the data acquisition software and are given the `.p` extension. Files with extension `.p` can be found on Rockland instruments using differing data acquisition systems including CF2 Persistor data loggers, Rockland Data Loggers (RDL), non-RDL Real Time instruments (ODAS-RT software), and older data loggers no longer supported. The `.p` files have been revised over time and the different versions of these `.p` files are discussed in this technical note.

Rockland offers several tools for opening and manipulating `.p` files, including functions from the ODAS Matlab Library, as well as the standalone applications Zissou Essentials and Zissou Premium. The information in this technical note acts as a specification for the `.p` file format; enabling the writing of custom scripts to read and manipulate the raw data contained in these `.p` files.

Table 1: Rockland Data Files including `.p` discussed in this technical note. Other data files generated by Rockland instruments are listed for reference.

File Type	Description & Source of Information about file type
`.p`	Files created by the majority of Rockland Instruments. Technical Note - 051
`.q` and `.mri`	Files created by In Situ Data Processing (ISDP) on most Rockland Instruments. Future Technical Note & TN-050
`.eng` and `.dat`	Profiling float based instruments using the MAPLe data logger create two types of data files including `.eng` and `.dat`. Information is provided in documents associated with these instruments.

1.1 Data File Versions

This document covers Rockland Data File version 6 and does not cover earlier versions. Older `.p` file versions (previous to version 6.0) are no longer supported by Rockland data processing software, however most earlier versions can be made compatible with current data processing software by patching an updated setup.cfg file into the older `.p` file. Please contact support@rocklandscientific.com if you require support patching an updated setup.cfg into old `.p` files. Please see table 2 for a list of supported data file versions with brief descriptions. Differences between these versions are discussed in more detail in sections 2 and 3.

Table 2: ODAS data file versions with descriptions.

Version	Release Date	Description
1.0	Prior to 2010	Original version.
6.0	2010	Version created by the following data acquisition software: ODAS-RT (Non-RDL Real Time), ODAS5IR (CF2 persistor based instruments). Additional updates to ODAS-RT and ODAS5IR are not expected.
6.1	2020-Nov-13	File version created by Rockland Data Logger (RDL) based instruments running OS versions 4.11 and earlier.
6.2	2023-Sep-11	File Timestamps modified so that each record uses a calculated timestamp. See Section 2.4.3 for more information. Included with RDL OS version 4.12 up to and including RDL OS version 4.16
6.3	2024-Jul-30	Version 6.3 data files fix the timing error in version 6.2 data files. See Section 2.4.4 for more information. Version 6.3 data files include a new feature enabling easy stitching together of files. Included with RDL OS version 4.17 and newer.

1.2 Version 1.0 Data Files

In most cases, it is possible to upgrade pre-v6 data files into v6 data files by patching a v6 configuration file (`setup.cfg`) into the pre-v6 data file. Please see data processing software documentation for instructions on patching configuration files into `.p` files.

2 File Structure

The file is structured as a series of records, the first of which includes a header and the configuration string. Subsequent records are data records constructed by prepending a header to a data block.

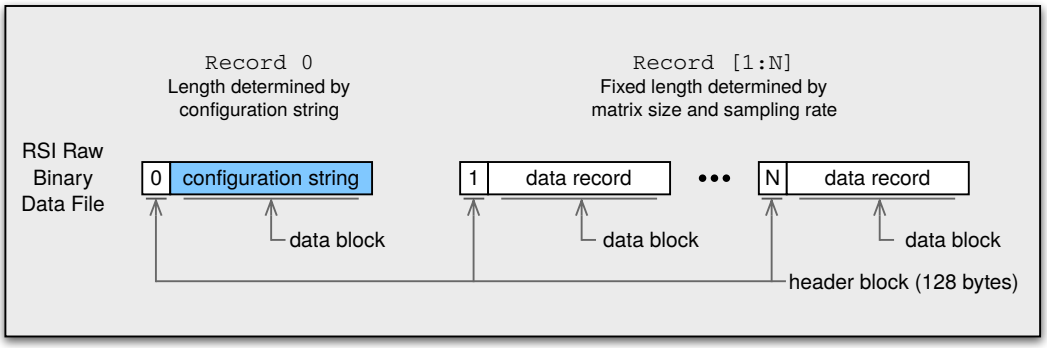

Figure 1: Visualization of the record composition of the Rockland data files.

2.1 Header

Both the configuration and data records start with a 128 byte header comprised of 64, 16-bit words. This header contains the information needed to correctly read the data file. Most entries in the header are self explanatory, see the table below, but some entries require the additional details described in section 2.1.1.

Table 3: Description of fields that constitute a record header.

Location	Description
1	File number, the three or four-digit number appended to the name of the data file
2	Record number
3	Record number of the input on the RS-232 port, for non-RDL real-time telemetering instruments (ODAS-RT only). Used to manage records in corresponding data files logging RS-232 based auxilary sensors such as GPS. Not used.
4	Year, see section 2.4
5	Month, see section 2.4
6	Day, see section 2.4
7	Hour, see section 2.4
8	Minute, see section 2.4
9	Second, see section 2.4
10	Millisecond, see section 2.4
11	Rockland data file version (MSB: major version, LSB: minor version)
12	Configuration string size in bytes
13	Data Acquisition Software Product ID (0=legacy 1=odas5ir, 2=odasrt, 3=odas4ir), see note 2
14	Build number, not used.
15	Time zone as minutes from UTC
16	Buffer status (1 if special character check fails, otherwise 0), see section 3
17	Restarted (1 if data acquisition was restarted due to buffer status = 1. Otherwise 0), see section 3.1.1
18	Record header size in bytes (128)
19	Data record size in bytes (header + data block)
20	Number of records written to the current file
21	Truncated frequency of the data acquisition clock (Hz), see note 5
22	Fractional part of the frequency of the data acquisition clock (to 0.001 Hz), see note 5
23-28	not used
29	Number of fast columns in the address `[matrix]`, see note 6
30	Number of slow columns in the address `[matrix]`, see note 6
31	Number of rows in the address `[matrix]`, see note 6
32-62	Not used
63	Profile (0=Vertical, 1=Horizontal). Default value is 0, not used, see note 7
64	Data type (0=unknown, 1=little endian, 2=big endian), see 8

2.1.1 Header Entry Notes

1. For discussion of the date-time entries (locations 4 – 10) and the differences between data file versions, please see section 2.4.
2. The entry in location 13 indicates the Data Acquisition Software Product ID (0=legacy 1=odas5ir, 2=odasrt, 3=odas4ir). This parameter is used in 6.0 data files but is no longer used in data file versions 6.1 and newer. For data file versions 6.1 and newer the entry in location 13 is permanently set to 1.
3. The entry in location 16 indicates a “Bad Buffer”. For a discussion of “Bad Buffers” and the differences between data file versions, please see section 3.
4. For details on the entry in location 17 please see section 3.1.1.
5. Entries in locations 21 and 22 give the sampling rate. Please note this is the actual sample rate and may differ slightly from the requested sample rate. It may differ from the requested rate if it is not a whole-number divisor of the 38.4MHz data acquisition clock. If so, the data acquisition software selects the nearest rate. The data acquisition clock is accurate to approximately 1.5 part per million, with a temperature stability of 0.5 part per million. The sampling rate of fast channels is the actual sampling rate divided by the number of columns in the address `[matrix]`. The number of columns in the address matrix equals the sum of entires in locations 29 and 30.
6. Entries in locations 29 to 31 are the dimensions of the address `[matrix]`. They are used to de-multiplex the data records into individual channels.
7. The profile flag (entry in location 63) indicates the direction an instrument travels. It is no longer used.
8. The final entry in location 64, gives the endian format of the data file. Programs that read a data file directly should use this item to read the data correctly.

2.2 Configuration Record

The first record within a data file is the configuration record. The configuration record consists of a header and a copy of the configuration file (`setup.cfg`) used for data acquisition. The configuration record contains the address matrix. The copy is a single string of ASCII characters – the configuration string. The record number is always 0. The size of this record depends on the size of the configuration string.

To read the configuration record, open the data file and read the first 64 words (128 bytes). Use the last word, entry in location 64, to determine the endian format of the data file. Close the file and reopen it with the correct endian, if necessary. Use entries in locations 18 and 12 to read the first header and the configuration string.

2.3 Data Record

Following the configuration record you will find data records. Data records consist of a header and a data block. The header is described in section 2.1 and the data block consists of raw data from an instrument. Data words (samples of 2-byte numbers) are stored in the order provided by the address matrix. The number of words within a data block is a multiple of the size of the address matrix. The total size of a data record (header + data block) is stored in entry location 19 of the record header. All data records will be the same size. By default, data records are approximately 1 second in length; however the record duration (seconds) can be specified in the setup.cfg file.

To read the data records, start by reading the configuration record. The file pointer will be at the start of the first record. Use entries in locations 18 and 19 to read the header and data for the first data record. Continue reading data records until you reach the end-of-file.

2.4 Time

The definition of the date-time entries (in locations 4 – 10) vary depending on the data file version.

2.4.1 Version 6.0 Time

For version 6.0 data files, the date-time entries (in locations 4 – 10) record the time at which the record is written to storage. For version 6.0 data files Rockland software only uses the date-time of the first record in the data file and then calculates timestamps for each subsequent record using the actual sample rate.

2.4.2 Version 6.1 Time

Version 6.1 uses the same method as 6.0; the date-time entries (in locations 4 – 10) record the time at which the record is written to storage. Version 6.1 has a known bug with the date-time entries. The RDL typically captures up to a few minutes of data records (in the FIFO) before writing to storage. This caused the date-time to only be updated every few minutes (whenever the data records were written to storage). A fix for this bug was included in version 6.2 data files which are created by RDL instruments running OS Patch 4.12 and newer.

2.4.3 Version 6.2 Time

Version 6.2 data file date-time entries (in locations 4 – 10) are derived from two different sources depending on the record. The initial record date-time originates from the system time which is set by a RTC (real time clock) at time of boot. Note that a host (e.g. glider or AUV) can set the system time after booting the instrument. Subsequent records have artificial date-time entries

which are derived from the previous record time and the expected duration of a record. The accuracy of these times is determined by the accuracy of the data acquisition clock with has an accuracy of <2 ppm.

Please note a firmware bug caused a count-by-one error in the data acquisition clock. Zissou Premium version 1.1.0 and newer and Zissou Essentials version 2.2.1 and newer correct for this error when processing v6.1 and v6.2 data files. Please contact support@rocklandscientific.com if more information is required on this matter.

2.4.4 Version 6.3 Time

Version 6.3 date-time entries are the same as version 6.2 date-time entries with the one difference being that the firmware bug was corrected. Current data processing software only applies corrections to impacted files and will leave version 6.3 files unchanged.

2.4.5 Contiguous Data Files

Version 6.1, 6.2, and 6.3 data files maintain contiguous data from one file to the subsequent file in three scenarios. The three scenarios include when a new file is created by using the `odas restart` command, when a new file is created because the `maxfilesize` parameter has been reached, and when a new file is created because the `maxfiletime` parameter has been reached. No data has been lost between files in these three scenarios. In some cases, unfortunately, the timestamps may suggest a time gap between files, however this is not correct, the data is contiguous. In scenarios where the instrument is power cycled, or the data acquistion is stopped and then started separately, contiguous data is not maintained.

With version 6.1 data files, the header date-time value is set to the system time at which the data is saved. This time can be minutes after the time when data was collected due to the impact of data buffering. This leads to misleading start times. As mentioned above, later data file versions use calculated date-time values in place of the system time. As such, the header date-time entries should be accurate and data from contiguous data files can be directly joined together.

Please note that for long periods of continuous data acquisition, the log file (`logfile.txt`) event time may become mis-aligned with the data records as the log file event times are set by the system clock and the records are calculated using the data acquisition clock, and the drift of these two clocks will differ.

For cases outside of the three scenarios described above (`maxfilesize`, `maxfiletime`, and `odas restart`), the system time will be used to set the time of the first record of new data files. For example, this includes when cyclic sampling is employed or when the platform issues commands to end and re-start data acquisition as is common on many glider platforms.

3 Bad Buffers

"Bad buffer" is the term used to describe an error found in a record of data. The error could be a missing or erroneous data record, or missing or erroneous data from a specific channel. Bad buffers can be generated for multiple reasons including when a channel does not issue a response in time, an issue has occurred with the hardware or software on an instrument, corruption has occurred during transfer of data files, and other reasons. A very small number of bad buffers are expected to occur on all Rockland instruments including RDL instruments on very rare occasions. Please contact support@rocklandscientific.com to assst with troubleshooting bad buffers.

3.1 v6.0 Bad Buffers (non-RDL Real Time and CF2 Persistor Instruments)

Version 6.0 data files record a special character from channel 255 in every record. The size of the record is known based on the size of the address matrix. If one or more channels do not respond in a record the special character will not occur at the expected interval and the record will be labeled a bad buffer. Only the entirety of a record can be labeled a bad buffer even though a single channel may be the cause. It may be possible to identify which channel is the source of the bad buffer by using the calibrate all function in the data acquisition software.

The special character is 32752 in raw counts (hexidecimal value: 0X7FF0). The special character value was chosen as it is near but not equal to the maximum positive value for raw counts of 32768. It is therefore an unlikely number, but also is not the maximum positive value which is often observed when a channel is saturated.

Data file v6.0 bad buffers can be cleaned up in the Rockland ODAS Matlab Library data processing software by replacing them with average channel values only when there are sufficient special characters at expected intervals in the neighbourhood of the bad buffer. If too many subsequant bad buffers occur the data processing software will not be able to clean up the bad buffers.

3.1.1 A note about bad buffers and non-RDL real time instruments

For ODAS-RT software used on non-RDL Real Time instruments, the software will re-start data acquisition when a large number of bad buffers are detected. A re-start event will be indicated in header entry in location 17 (see table 2)

3.2 v6.1 and Newer Bad Buffers (RDL Instruments)

The meaning of bad buffers changed with version 6.1 data files, as the RDL enabled identification of an error with a specific channel within a data record. Version 6.1 and newer data files no

longer flag the entire record as a bad buffer as is the case with version 6.0 data files. Accordingly, the special character channel 255 is removed from version 6.1 data files and newer. When missing or erroneous data is detected from a specific channel by the RDL, the negative special character -32753 raw counts (hexidecimal value: 0X800F) is input as a data point to replace the missing or erroneous data. For version 6.1 data files and newer, a single record can contain multiple bad buffers. This method ensures that each record is always the correct size.

Features providing cleaning of bad buffers for both 6.0 and versions 6.1 and newer may be available in Zissou Software. Currently Zissou Software will notify the user of the presence of bad buffers as well as recommendations about when to be concerned about the quantity of bad buffers. If you require support for version 6.1 and newer bad buffers occuring on RDL instruments, please contact support@rocklandscientific.com.

End of Document