

cubeaux(1)

Name

cubeaux - read out auxiliary recording channels from Cube files

Synopsis

```
cubeaux [-v | --verbose] [--include-pattern=PATTERN]...  
        [--wnro-correction=AUTO | OFF | FORCE] [--channel=CHANNEL]...  
        [--output-dir=DIRECTORY [--force-overwrite]]  
        file | directory...
```

```
cubeaux [-h | --help] [--version] [--sysinfo]
```

Description

Cubeaux is used to read out auxiliary or secondary recording channels from one or more Cube *files*. Typical examples for secondary channels are logger temperature, operational voltage or the geographic location of the logger. But custom build externally attached sensors (the eponymous "auxiliary" channels) are a possibility as well. Which channels are available depends on the used recorder hardware/firmware/sensor combination. Common to all secondary channels is a much lower sampling rate compared to the primary recording channels.

If a *directory* is given as argument, **cubeaux** searches recursively for Cube files inside the *directory*. The search for Cube files can be restricted to only consider files with a name matching the patterns given by the `--include-pattern` option.

After the initial search for available Cube files has completed, the program will begin to index the data contained in the respective files. This step is necessary so that the program later knows, which Cube files belonging to the same continuous trace and their correct chronological order.

The data of the selected (use the `--channel` option) secondary channels is written directly to standard output (i.e., text console) or saved to an output directory (option `--output-dir`).

Options

The program pretty much follows the usual Unix command line syntax. Some command line options have two variants, one long and an additional short one (for convenience). These are shown below, separated by commas. However, most options only have a long variant. The '=' for options that take a parameter is required and cannot be replaced by whitespace.

-h, --help

Summarize the most important command line options and exit.

--version

Print the **cubeaux** release information and exit.

--sysinfo

Report system settings relevant for **cubeaux** and exit.

-v, --verbose

This option increases the amount of information given to the user during program execution. By default, (i.e. without this option) **cubeaux** only reports warnings and errors. (See the diagnostics section below.)

--include-pattern=*PATTERN*

Only process files whose filename matches the given *PATTERN*. Files with a name not matching the search *PATTERN* will be ignored. This option is useful to speed up recursive searches through large subdirectory trees and can be given more than once in the same command line.

You can use the two wild card characters (***** and **?**) when specifying a *PATTERN* (e.g. ***.123**). Or, alternatively, you can also use a predefined filter called **GIPP** that can be used to exclude all files not following the usual GIPP naming convention for files recorded by Cubes.



The search *PATTERN* is only applied to the filename part and not to the full pathname of a file. It will not match directories.

--wnro-correction=*AUTO* | *OFF* | *FORCE*

Data loggers receive the recording time from the Global Positioning System (*GPS*). Unlike the commonly used Gregorian calendar, *GPS* (internally) expresses date/time information using two integer numbers. The first number counts the weeks since start of the *GPS* system on January 6th, 1980. The second number gives the seconds relative to the beginning of the week.

Satellites transmit the week number as a 10-bit long integer, which will become zero again every 1024 weeks. This integer overflow is called week number rollover (*WNRO*) and is a common issue with all GPS receivers.

Usually, the **cubeaux** utility will detect *WNRO* and just correct the date automatically by adding 1024 weeks to the "wrong" recording time. However, in cases where the automatic detection and correction fails, the handling of *WNRO* can be manually controlled using this command line option.

AUTO

Automatically detect and correct *WNRO* problems. This is the default mode.

OFF

Disable any (automatic) *WNRO* corrections for the input data. Time information will be used as recorded by the GPS receiver.

FORCE

Force apply *WNRO* corrections to the input data.

Please note that the `--wnro-correction` option only affects the handling of *WNRO* caused errors. General quality control of the timing accuracy can independently be adjusted using the `--timing-control` command line option.



Unless there is a good reason for it, the "*WNRO* correction" should be left in the default automatic mode. Use *OFF* or *FORCE* arguments to overwrite in case the automatic *WNRO* detection fails.

--channel=CHANNEL

Select an auxiliary/secondary recording channel for a read-out. The following channels can be selected:

AUXILIARY

Data recorded by custom build, externally attached hardware. Usually used to monitor things like solar panel voltage, temperature, humidity, sensor orientation (gyroscope), etc.



Special, custom-made hardware is required to record auxiliary channels! The measurements are sent to the Cube via a serial interface, where they will be threaded into the normal Cube data stream.

GNSS

Returns the geographic location of the Cube logger during the recording. The output is tabular and looks as follows:

```
2024-11-12 11:14:44 c0123 +52.2510 +20.9286 75.4 4 2D
2024-11-12 11:14:45 c0123 +52.2510 +20.9285 77.2 4 2D
2024-11-12 11:14:46 c0123 +52.2511 +20.9284 76.5 5 2D
2024-11-12 11:14:47 c0123 +52.2512 +20.9283 79.8 5 3D
2024-11-12 11:14:48 c0123 +52.2512 +20.9282 77.3 5 3D
2024-11-12 11:14:49 c0123 +52.2511 +20.9271 76.9 7 3D
```

The columns are recording *date* and *time*, *Cube ID*, *latitude*, *longitude*, *elevation*, *satellite count* and *GPS fix type*.



A more detailed view of the information recorded by the *GNSS receiver* can be viewed with the `cubeinfo --format=GPS` command.

TEMPERATURE

Temperature inside the Cube enclosure. Example:

```
2024-01-09 11:58:30 c0123 24.55 Celsius
2024-01-09 11:58:40 c0123 24.79 Celsius
2024-01-09 11:58:50 c0123 24.84 Celsius
```

The availability of temperature information varies depending on logger hardware and firmware release! Some (mostly older) loggers do not contain a temperature sensor.

VOLTAGE

Operating voltage of the Cube logger. Depending on the used power source, this is the voltage of the internal D cells batteries or the voltage of an external attached power supply. Example:

```
2024-08-23 08:00:01 c0123 2.90 Volt
2024-08-23 16:00:01 c0123 2.76 Volt
2024-08-24 00:00:01 c0123 2.70 Volt
2024-08-24 08:00:01 c0123 2.66 Volt
2024-08-24 16:00:01 c0123 2.66 Volt
2024-08-25 00:00:01 c0123 2.62 Volt
```



The maximum reported voltage will be clipped either at about 16 Volts or 20 Volts depending on the build-in sensor.

The availability of voltage information varies depending on logger hardware and firmware release! Some (mostly older) loggers do not provide voltage values in the recorded data stream at all!

ALL

The combination of *AUXILIARY*, *GNSS*, *TEMPERATURE* and *VOLTAGE* channels. Selecting all of the above is useful for having a quick peek into a Cube file to learn what auxiliary recording channels it contains. To report *ALL* available secondary channels is also the default if the `--channel` command line option is unused.



If you intend to further process the data, it probably makes more sense to read out the desired secondary channel individually.

--output-dir=*DIRECTORY*

Save the resulting output to files in this *DIRECTORY*. The directory must already exist and be writable! Already existing files in that directory will not be overwritten unless the option `--force-overwrite` is used as well.

--force-overwrite

If this option is used, already existing files in the output directory will be overwritten without mercy!

The default behavior, however, is **not** to overwrite already existing files. Instead, a new file is created with an additional number before the file extension.

Environment

The following environment variables can optionally be used to influence the behavior of the GIPPTool utilities.

GIPPTOOLS_HOME

This environment variable can be used to set the GIPPtools installation directory.

In particular, the Java classes that make up the GIPPtools are read from *JAR* files in the **java** subdirectory located inside **GIPPTOOLS_HOME**. The start scripts for the individual GIPPtool utilities can be found inside the **bin** subdirectory.

GIPPTOOLS_JAVA

The utilities of the GIPPtools are written in the programming language Java and consequently need a Java Runtime Environment (*JRE*) to execute. Use this variable to specify the path to the *JRE*, which should be used.

GIPPTOOLS_OPTS

You can use this environment variable for additional fine-tuning of the Java runtime environment. It is typically used to set the Java heap size available to GIPPtool programs.

GIPPTOOLS_LEAP

The GIPPtools require up-to-date leap second information to correctly interpret Cube files. Usually, this information is read from the **leap-seconds.list** file located in the **config** subdirectory of the GIPPtools installation directory (**GIPPTOOLS_HOME**). This environment variable can be used to provide a more up-to-date leap second list to GIPPtool programs.

It is usually not necessary to define any of those variables as suitable values should be selected automatically. However, if the automatic detection build into the start script fails, or you need to choose between different GIPPtool or Java runtime releases installed on your computer, these environment variables might become helpful to troubleshoot the situation.

Diagnostics

During execution, the **cubeaux** utility will produce user feedback. In general, user messages are classified as *INFO*, *WARNING* or *ERROR*.

INFO messages usually report about the progress of the program run, give statistical information or write a final summary. They are only displayed when the **--verbose** command line option is used.

More important are *WARNING* messages. In general, they warn about any issues that may influence the outcome in unexpected ways. Although the program will continue with execution, you certainly should check the results carefully. You might not have gotten what you (thought you) asked for.

Finally, *ERROR* messages inform about problems that cannot be resolved automatically. Program execution usually stops and the user must fix the cause of the error first.

Exit codes

Use the following program exit codes when calling **cubeaux** from scripts or other programs to see if **cubeaux** finished successfully. Any non-zero code indicates an *ERROR*!

0

Success.

64

Command line syntax or usage error.

65

Input data was incorrectly formatted.

66

An input file did not exist or was not readable.

70

Error in internal program logic.

74

I/O error.

99

Other, unspecified errors.

Examples

1. Peek into a Cube file to find out what secondary recording channels it contains.

```
cubeaux --verbose ./cube-in/12200000.123
```

The program will output the data of all secondary channels recorded by Cube #123.

2. Read out the geographic position of Cube #123 during the recording.

To restrict the output to a single secondary channel the command line option `--channel` can be used.

```
cubeaux --include-pattern='*.123' --channel=GNSS ./cube-in
```

The program will search for files recorded by Cube #123 in the `cube-in` directory and output tabular data with *date*, *time*, *Cube ID*, *latitude*, *longitude*, *elevation*, *satellite count* and *GPS fix type* information.

3. Get the operating voltage of all Cubes.

The following command line will search the `cube-in` subdirectory for Cube files and write operating voltage (over time) for each Cube into the `voltage-out` subdirectory.

```
cubeaux --channel=VOLTAGE --output-dir=./voltage-out/ ./cube-in/
```

Files

\$GIPPTOOLS_HOME/bin/cubeaux

The **cubeaux** "program". Usually just a copy of or a symbolic link pointing to the standard GIPPTools start script.

\$GIPPTOOLS_HOME/bin/gipptools

The standard start script used to run all GIPPTool utilities.

See also

gipptools(1), cube2ascii(1), cube2mseed(1), cube2segy(1), cubeaux(1), cubeevent(1), cubeinfo(1), cubeinspect(1), mseed2ascii(1), mseed2mseed(1), mseed2pdas(1), mseed2segy(1), mseedcut(1), mseedinfo(1), mseedrecover(1), mseedrename(1)

Bugs and caveats

None so far.