# PowerSpy CSV format



## Introduction

The PowerSpy CSV data format make heavy use of the first element to communicate meta data about the whole buffer. This element can have two types of content:

- a space separated list of **key:value** pairs
- the word TIME

The second option (TIME) is only supported for legacy reasons, as explained in the section below on "FGCspy Compatibility", while the first option is the standard approach for normal PowerSpy CSV files. The order of **key:value** is not significant.

| **Buffer Parameters** | **Default value** | **Notes** |
|---|---|---|
| type | "analog" | Buffer type: "analog" \| "digital" \| "frequency" \| "parametric" \| "table". This is not case sensitive. |
| source | "FILE" | Data source type: "FGC", "CCRT", "SVC", "DIM", or "PM". Used to group the monitors. |
| device | file name | If the data is coming from a local file, the default value will be the file name *without the .csv suffix (if present) and with spaces, colons or commas replaced by underscores, full stops and semicolons respectively*.<br><br>The device will appear in the monitor title (DEVICE : BUF_NAME (cycleSelector) ) and will also prefix the signal name in the tooltips (DEVICE : BUF_NAME (cycleSelector) : SIGNAL_NAME : ACQ_INDEX). The webserver will always specify the device in the JSON data it returns to the PowerSpy. If cycleSelector is NONE, then it is dropped from the BUF_NAME field. |
| name | empty string | The buffer name will appear in the monitor title (DEVICE : BUF_NAME). |
| cycleSelector | 0 | Cycle selector that applied to the acquisition of the data. For non-cyclic acquisitions, this is 0. For cyclic acquisition, it can be 1-N for cyclic or the cycle selector name as a string. |
| **Time Series Buffer Parameters** | **Default value** | **Notes** |
| epoch | 0 | The oldest integer part between timeOrigin and firstSampleTime. For example:  timeOrigin:1668442670.000000000 firstSampleTime: 1668442668.000000099  epoch = 1668442668 or more generalized: epoch = MIN(time_origin_s, timestamp_s |
| timeOrigin | TIME1 | Time origin for all the signals in the buffer in UTC Unix time. This is the time that will appear as 0.0 when the PowerSpy is in relative time mode. In this mode, the time origin will be subtracted from the time of every sample before display.<br><br>If the epoch is provided, then it should be subtracted form the timeOrigin  (i.e. timeOrigin = timeOrigin - epoch) |
| firstSampleTime | TIME1 | Time of first sample in UTC Unix time. If the value is zero, it will appear to be at the start of 1970.<br>If the epoch is provided, then it should be subtracted form the firstSampleTime  (i.e. firstSampleTime = firstSampleTime - epoch) |

| | | |
|---|---|---|
| period | TIME2 - TIME1 | Sampling period (s). |
| **Table Buffer Parameters** | **Default value** | **Notes** |
| subtype | - | This is to specify the type of the table. Possibilities are "default" \| "event-log" \| "cycle-log" \| "timing-log" \| "error-log" \| "bode" \| "group-delay" \| comparison" \| "properties".<br>Properties is a case of table containing FGC properties and values which can be send directly to the converter. |
| columns | see 'Notes' | This field contains the styling for the columns and is valid only if subtype is not specified. The options are:<br>• Upper/Lower case letter : Bold/normal<br>• L : Align left<br>• C : Align center<br>• R : Align right<br><br>The number of letters must be the same as the number of columns.<br><br>This letter for each column defaults to normal text weight (no bold) and justified to left: 'l' |

## Buffers Containing Analog or Digital Signals

```
BUF_PARS,SIGNAL1_PARS,SIGNAL2_PARS,...,SIGNALM_PARS
TIME1,SIGNAL1_VALUE1,SIGNAL2_VALUE1,...,SIGNALM_VALUE1
TIME2,SIGNAL1_VALUE2,SIGNAL2_VALUE2,...,SIGNALM_VALUE2
TIME3,SIGNAL1_VALUE3,SIGNAL2_VALUE3,...,SIGNALM_VALUE3
...
TIMEN,SIGNAL1_VALUEN,SIGNAL2_VALUEN,...,SIGNALM_VALUEN
```

- The first line contains the buffer parameters' field as well as the fields for the signal parameters.
- The first column contains the moments of time common to all signals
- The rest are the values for the signals for each moment of time

The Signal parameters are:

| Signal Parameters | Mandatory | Default value | Notes |
|---|---|---|---|
| name | Yes | - | The name of the signal. |
| step | No | - | If this string is found in the field, trailing step interpolation is applied for this signal. Case-insensitive. |
| timeOffset | No | - | If a numeric value is found in the field, it will be used as the time offset for the signals in seconds. The time of a sample will be calculated from firstSampleTime + period * sample_index + timeOffset. So a positive time offset will move the point later and a negative time offset will move it earlier. It is recommended, but not required, to prefix positive values with "+". |

The Signal parameters should be given in such order that the signal **name** is given first. The rest can be in arbitrary order.

**TIMEN** is the time stamp of the $N^{th}$ sample given in UTC UNIX time (e.g. 1458137213.240000).

**SIGNALM_PARSN** defines the value for the $N^{th}$ sample of signal M. This must be a number. For digital signals, it should be "0" or "1" only. For analog signals, it can be any number that fits into a double precision floating point value.

### Analog Buffer

```
source:fgc device:SYSTEM_NAME name:BUFFER_NAME,SIGNAL1 +0.002 STEP,SIGNAL2 STEP

1458137212.000000, 10.1   , -5, 1

1458137212.000100, 11.5E+3, -3, 2

1458137212.000200,-12.2E-2,  1, 3
```

### Digital buffer

```
source:ccrt device:SYSTEM_NAME name:BUFFER_NAME type:digital,SIGNAL1,SIGNAL2 -0.5,SIGNAL3 +1.5

1458137212.000000,0,1,0

1458137212.000100,0,0,1
```

```
1458137212.000200,1,1,1
```

# Buffers Containing a Table

## Without specified subtype

### Case 1

If the first field contains buffer parameters (more precisely a string: 'type:table'), then the first column of the file is omitted even if it contained data

```
type:table ...,HEADING1,HEADING2,...,HEADINGM
,COLUMN1_PARS1,COLUMN2_PARS1,...,COLUMNM_PARS1
,COLUMN1_PARS2,COLUMN2_PARS2,...,COLUMNM_PARS2
,COLUMN1_PARS3,COLUMN2_PARS3,...,COLUMNM_PARS3
...
,COLUMN1_PARSN,COLUMN2_PARSN,...,COLUMNM_PARSN
```

- The first line contains the buffer parameters (first field) and the headings for each column. The headings may be left out, in which case the table will not have headings
- The first column is not read at all (leading comma)
- The rest of the columns contain column data
- If a column field contains a comma (,) or a double quote ("), the whole field must be put inside double quotes ("field") and all the double quotes must be changed to double double quotes (e.g. "I like 9**""** nails")

### Case 2

Otherwise, the whole first field is assumed to be a heading*****, in which case the first column is read (no leading comma)

```
HEADING1,HEADING2,...,HEADINGM
COLUMN1_PARS1,COLUMN2_PARS1,...,COLUMNM_PARS1
COLUMN1_PARS2,COLUMN2_PARS2,...,COLUMNM_PARS2
COLUMN1_PARS3,COLUMN2_PARS3,...,COLUMNM_PARS3
...
COLUMN1_PARSN,COLUMN2_PARSN,...,COLUMNM_PARSN
```

***** The file must have an extension '.csv' otherwise it is assumed to be of subtype 'text'. Also, if the field contains just the word 'time' (case insensitive), it is assumed to be an analog log.

## Subtype: Event Log

```
type:table subtype:eventlog...,Property,Action,Value,Status
TIMESTAMP1,PROPERTY1,ACTION1,VALUE1,STATUS1
TIMESTAMP2,PROPERTY2,ACTION2,VALUE2,STATUS2
TIMESTAMP3,PROPERTY3,ACTION3,VALUE3,STATUS3
...
TIMESTAMPN,PROPERTYN,ACTIONN,VALUEN,STATUSN
```

- The first line contains the buffer parameters (first field) and the headings for each column. The headings may be left out, in which case they default to Property, Action, Value and Status
- The first column contains the timestamps for the table rows. If any is missing or invalid, all the timestamps are omitted.
- The rest of the columns contain column data
- If a column field contains a comma (,) or a double quote ("), the whole field must be put inside double quotes ("field") and all the double quotes must be changed to double double quotes (e.g. "I like 9**""** nails")
- The Event Logs are automatically sorted by time stamps

## Subtype: Text

### Case 1

If the first field contains buffer parameters ('type:table' and 'subtype:text'), then the first row of the file is omitted even if it contained data

```
type:table subtype:text..., { OMITTED }
ROW1
ROW2
ROW3
...
ROWN
```

### Case 2

If the parameters of Case 1 are not present*, the first row of the file is taken as a first row of data

```
ROW1
ROW2
ROW3
...
ROWN
```

* If the file name has an extension '.csv', it is considered to be a table without specified subtype.

## Examples

### Example 1 - Table without subtype

```
type:table columns:lCl, Price, Profit, ,
,100,12,
,450,53, Price and profit depends on the freshness of the product
,25, 3, May not arrive on time
```

| Price | Profit | |
|---|---|---|
| 100 | **12** | |
| 450 | **53** | Price and profit depends on the freshness of the product |
| 25 | **3** | May not arrive on time |

The same data without the first column in the file:

```
Price, Profit, ,
100,12,
450,53, Price and profit depends on the freshness of the product
25, 3, May not arrive on time
```

| Price | Profit | |
|---|---|---|
| 100 | 12 | |
| 450 | 53 | Price and profit depends on the freshness of the product |
| 25 | 3 | May not arrive on time |

### Example 2 - Event Log

```
type:table subtype:eventlog, , , ,
1464722837.172000,STATUS.ST_UNLATCHED,START_EVENT,SET_BIT,+
1464722837.272000,STATUS.ST_UNLATCHED,START_EVENT,CLR_BIT,+
1464722837.242000,STATE.PC,RUNNING,SET,
```

| Timestamp | Property | Value | Action | Status |
|---|---|---|---|---|
| 2016/05/31 21:27:17.172000 | STATUS.ST_UNLATCHED | START_EVENT | SET_BIT | FREQUENT |
| 2016/05/31 21:27:17.242000 | STATE.PC | RUNNING | SET | |
| 2016/05/31 21:27:17.272000 | STATUS.ST_UNLATCHED | START_EVENT | CLR_BIT | FREQUENT |

### Example 3 - Text

```
type:table subtype:text, "Whatever is put here is not read",
A long time ago in a
Galaxy far, far away...
```

A long time ago in a

Galaxy far, far away...

The same data without buffer parameters (file type extension cannot be .csv)

```
A long time ago in a
Galaxy far, far away...
```

A long time ago in a

Galaxy far, far away...

## FGCspy Compatibility

The PowerSpy CSV format is backwards compatible with existing FGCspy files.

The BUF_DATA field will be "TIME", which is ignored, and first sample time can be zero, in which case Spy data will appear to be from the beginning of 1970.

PowerSpy will set the TimeOrigin to equal the first sample time.

PowerSpy uses a heuristic algorithm to enable the trailing step interpolation:

- If the signal name contains "REF" or "ERR" then trailing step interpolation will be enabled.

## Downloading CSV to a local file

The PowerSpy interface allows the user to download all signals or just the visible signals.

The name of the file will be:

> Device_name_Buffer_name_YYYY-MM-DD_HH-MM-SS.csv

where the time is the time origin in local time.

If history level is greater than one and a monitor contains multiple acquisitions, only a single CSV file will be downloaded containing all the CSV data separated by 1 newline character. Each acquisition will have its own header row. Example:

```
source:FGC type:analog device:RFNA.866.04.ETH1 name:I_MEAS timeOrigin:1582901270.25 firstSampleTime:1582901269.25
period:0.0001,I_MEAS,I_MEAS_FLTR,I_REF_DELAYED STEP,I_ERR STEP
1582901269.793300,0.0000000e+0,0.0000000e+0,0.0000000e+0,0.0000000e+0
[10K hidden rows]
1582901269.793500,0.0000000e+0,0.0000000e+0,0.0000000e+0,0.0000000e+0

source:FGC type:analog device:RFNA.866.04.ETH1 name:I_MEAS timeOrigin:1582901272.14 firstSampleTime:1582901271.14
period:0.0001,I_MEAS,I_MEAS_FLTR,I_REF_DELAYED STEP,I_ERR STEP
1582901271.140000,0.0000000e+0,0.0000000e+0,0.0000000e+0,0.0000000e+0
[10K hidden rows]
1582901271.140200,0.0000000e+0,0.0000000e+0,0.0000000e+0,0.0000000e+0
```

# Error table logs

Error tables are a special type of table generated by the FGC Logger, to report any failure during the reading of a log.

They conform to the same format defined for table logs.

The current error table format expects the following columns:

| Column | Format | Can be empy | Description |
|---|---|---|---|
| TIMESTAMP | Float value, as a string | No | Timestamp of error |
| FAILED_LOG | String | No | Log whose error refers to |
| ERROR_TYPE | Enum value, as a string | No | Error type (FGC_ERROR or EXCEPTION) |
| ERROR_CODE | Integer, as a string | Yes | FGC error code |
| ERROR_MESSAGE | String | Yes | FGC error message |
| ERROR_INFO | String | Yes | Exception information |

Two main types of errors are expected:

- FGC_ERROR: Errors reported by the FGC. In this case, ERROR_CODE and ERROR_MESSAGE should specify the FGC error. ERROR_INFO can give extra information.
- EXCEPTION: Other errors faced by the FGC logger during the reading/decoding/storing of the logs. ERROR_INFO should contain exception details.

# References


- RFC 4180: Common Format and MIME Type for CSV Files