ENGLISH

Export Block Descriptor

REFERENCE GUIDE

RG-0009-00-EN
Publication date 30/08/2023

Important User Information

Disclaimer
The information in this document is for informational purposes only. Please inform HMS Networks of any inaccuracies or omissions found in this document. HMS Networks disclaims any responsibility or liability for any errors that may appear in this document.

HMS Networks reserves the right to modify its products in line with its policy of continuous product development. The information in this document shall therefore not be construed as a commitment on the part of HMS Networks and is subject to change without notice. HMS Networks makes no commitment to update or keep current the information in this document.

The data, examples and illustrations found in this document are included for illustrative purposes and are only intended to help improve understanding of the functionality and handling of the product. In view of the wide range of possible applications of the product, and because of the many variables and requirements associated with any particular implementation, HMS Networks cannot assume responsibility or liability for actual use based on the data, examples or illustrations included in this document nor for any damages incurred during installation of the product. Those responsible for the use of the product must acquire sufficient knowledge in order to ensure that the product is used correctly in their specific application and that the application meets all performance and safety requirements including any applicable laws, regulations, codes and standards. Further, HMS Networks will under no circumstances assume liability or responsibility for any problems that may arise as a result from the use of undocumented features or functional side effects found outside the documented scope of the product. The effects caused by any direct or indirect use of such aspects of the product are undefined and may include e.g. compatibility issues and stability issues.

Table of Contents

1. Export Block Descriptor

Export block descriptors or EBDs are a system to describe data to be exported from an Ewon device. The data can then be used in different situations.

For example, it can be :

- **Included in the body of an email or as an attachment.**
- **Sent to an FTP server with an FTP PUT**
- **Retrieved by an FTP client with an FTP GET**
- **Included in a custom HTML page on the Ewon device**
- **Accessed in BASIC with OPEN "exp:..."**

An Export Block Descriptor is used to describe what data to retrieve and how to present it.

1.1. Export Block Descriptor Fields

An Export Block Descriptor is a string of characters describing the Ewon data to export in a predefined syntax. Typically, the Export Block Descriptor will include information about:

- **Which data to export (Event log, Historical logging, etc)**
- **How to format the exported data (Binary, Text, Html table, Graphic)**
- **Start time**
- **End time**
- **Which Tag is concerned**
- **…**

Example of an Export Block Descriptor:

```
$dtHL $ftT $st_m10 $et_0 $tnMyTag $fnData.csv
```

The export syntax is composed of a sequence of fields and their values. A **field** is an identifier starting with **$** and followed by 2 lower case letters (case sensitive).

- **The first letter of the parameter value immediately follows the last letter of the field.**
- **The parameter includes all characters up to the first space found or until a $ or [is detected.**
- **The parameter can also be placed between quotes ("). In that case the parameter value is the value between the quotes.**

The Export Block Descriptor fields are defined as below:

Fields	Description
$dt	Data type
$ft	Export format
$st	Start time
$et	End time
$tn	Tag name
$ts	Timestamp format
$ut	Update last time
$ic	Intrasecond counter
$ct	Compression type
$se	Script expression
$fl	Group filter
$fn	File name

NOTICE

Not all fields apply to every export block descriptor. The value of the data type field (**$dt**) determines which fields can be included in the EBD.

2. Export Fields Syntax Definition

The syntax for the different fields is defined in the following chapters.

2.1. $dt [Data Type]

The **$dt** field defines what data to export from the Ewon.

The **$dt** parameter can take one of the following values:

$dt Parameter	Description	Binary	Graph	Text	HTML	JSON
AH	Alarm history			T*	H	
AR	Alarm Real time			T*	H	
CF	Config	B*		T		
ES	estat file			T*	H	
EV	Event file			T*	H	
FW	Firmware	B*				
HL	Historical Logging	B*	G	T	H	
HT	Historical Table			T*	H	
HS	Historical String	B		T*	H	
IS	Instant Strings	B*				
IV	Instant values	B*		T		
KPI	Key Performance Indicator			T	H	J*
PG	Program			T*		
PP	PPP dump file	B				
RL	Real time logging	B*	G	T	H	
SC	Communications configuration file			T	H	
SE	Script Expression	B*		T	H	
SS	Scheduled status			T*	H	
SV	System Variable			T		
TL	Tag list			T*	H	
UF	User file	B*		T	H	
RE	Real Time diagnostic			T*		
TR	TAR File	B*				

NOTICE
The asterisk (*) in the previous table denotes the default value of the $ft (export format). For example, for the DataType HL (Historical Logging), the default export format will be B (Binary) if you do not specify a $ft in your Export Block Descriptor (using [$dtHL] is equivalent to [$dtHL $ftB]).

2.2. $ft [Format]

The **$ft** field defines how to format the data exported. The following formats are available:

$ft Parameter	Format description
B	Binary
G	Graph
T	Text
H	HTML Table
J	JSON

- **Binary**: The data is sent in a raw binary format, not modified by the export module.
- **Graph**: The data is used to produce a PNG image representing a graph of the values (historical trend or real time graph).
- **Text**: The data is formatted as a CSV file. This means that each record is represented with each field on a line separated by a semicolon (;). The string fields are written between quotes, each line is ending by a CRLF (0x0D, 0x0A) sequence.
- **HTML**: The data is placed in a simple HTML table. This format is useful for inserting data in custom HTML pages.
- **JSON**: This file format applies only to **$dtKPI**.

NOTICE
Not all file types are available for every data type.

2.3. $st [Start Time] and $et [End Time]

These 2 fields are used to limit the time range of an export operation. $st and $et provide the start and end time of the export. The parameter format is the same for both fields. There are 3 different formats for the **$st** and **$et** parameter:

- Relative Time
- Absolute Time
- From last $ut (more information can be found in the $ut [Update Time] section).

2.3.1. $st, $et with relative time

Syntax:

```
$st_([s]|[m]|[h]|[d])100 _ = back
```

(h,m,s,d = Hour, min, sec, day. 100 is the amount)

This represents a time relative to the current time expressed in days, hours, minutes or seconds. If no letter is specified minutes are used.

$st with relative time examples:

$st_m10	**10 minutes in the past**
$et_0	0 minutes in the past (= now)
$st_d2	2 days in the past

2.3.2. $st, $et with absolute time

Syntax:

```
$stDDMMYYYY[[_HHMMSS][[_mmm][[_I][[_T]]]]]
```

$st parameters:

DDMMYYYY	Means Day, Month, Year, 8 characters. This parameter is required.
HHMMSS	Means Hour, Minute, Second, 6 characters. This parameter is optional (0 used by default)
mmm	Means milliseconds (000 to 999) 3 characters. This parameter is optional but if present, HHMMSS must also be specified.
I	Means intra sec counter. This value is present when receiving a historical logging from the Ewon. It can be specified in export request to allow precise repositioning in the historical file. This parameter is optional, but if present, HHMMSS and mmm must also be specified.
T	Means Tag id. As for I, this parameter is used for precise positioning in historical file. This parameter is optional, but if specified, HHMMSS, mmm and I must be present also. When *ALL* Tags are specified, the Tag values are exported in chronological order. However, for the same timestamp, there can be more than one Tag value. To reposition correctly in the file, it is necessary to provide the last Tag output during a previous export.

$st with absolute time examples:

$st01012000_120000	1 jan 2000 at 12 AM
$st01012000_120000_010	1 jan 2000 at 12 AM + 10 msec

2.3.3. $st, $et with Last Time

By adding the **$ut** command in an Export Block Descriptor, you can ask the Ewon to **remember the time of the last point exported**. This time can be used for the next export. The last time is reset when the Ewon boots.

Syntax:

```
$stL
```

Where "**L**" is the time parameter meaning last time.

2.4. $ts [TIME FORMAT]

This field determines whether time fields are presented in the Ewon device's local time or in UTC. It also determines what date and time format is used for time strings.

The following options are available:

$ts Parameter	Time Format
No modifier	Local time (local time format) -*string format* "DD/MM/YYYY HH:MM:SS" -*Example:* "19/09/2018 12:48:12"
O	Local time (UTC format) -*string format* "DD/MM/YYYY HH:MM:SS" -*Example:* "19/09/2018 10:48:12"
U	ISO 8601 ZULU (UTC format) -*string format* "YYYY-MM-DDTHH:MM:SSZ" -*Example:* "2018-09-19T10:48:12Z"
L	ISO 8601 local (local time format) -*string format* "YYYY-MM-DDTHH:MM:SS±000" -*Example:* "2018-09-19T12:48:12+0200"

2.5. $ut [UPDATE TIME]

This field has no parameter.

Using **$ut** means that at the end of this export, the time of the last point exported must be saved in the Ewon so that it can be used as a reference time for a later call.

Example:

```
$stL$et_0$ut
```

This sequence will specify a time range from last time to current time and will ask to update the last time at the end of the export.

- The last time is stored on a per Tag basis if one Tag is specified for the export.
- A global last time can also be saved if "ALL Tags" is specified in an export.

2.6. $tn [TAG NAME]

This field is used to specify a Tag name. It is required for *graph export format*. The parameter specified is the name of the Tag. When a **$tn** field can be specified for an export and no **$tn** is given, then the command is executed for all the Tags.

Example:

```
$tnMyTag
```

MyTag is the name of the Tag

2.7. $ic [INTRASECOND COUNTER]

If this field is used, the intra second counter field is included in the output for historical records.

2.8. $ct [COMPRESSION FORMAT]

This field is only applicable when sending a file from the Ewon to an FTP server, a web server or as an attachment to an email.

The compression format is gzip: http://www.gzip.org. The unique argument to add after the field "**$ct**" is "**G**".

Example:

```
Putftp "test2.txt.gz","[$dtUF $ctG $uf/test.txt]"
```

Or

```
SENDMAIL "destinator@provider.net", "", "Subject", "Mail body &amp;
[$dtUF $ctG $uf/usr/test.txt $fntest2.txt.gz]"
```

NOTICE

If you give the destination file only the "**.gz**" extension (and not "**.txt.gz**" for example), the destination file will be correctly exported, but you will have to indicate the extension when uncompressing ("**.txt**" in the above case).

You can then use a tool such as *Winrar** to extract the file; it will be extracted in a folder named "**test2.txt**".

2.9. $se [SCRIPT EXPRESSION]

This field is only required for **$dtSE** export data. The **$se** parameter specifies the "*script expression*" to compute.

Usually, the **$se** parameter will be inserted between quotes because if a **$** is found in the expression it will be considered as the end of the parameters.

Example:

```
$dtSE $se"A$"
```

Exports the content of A$

2.10. $fl [GROUP FILTER]

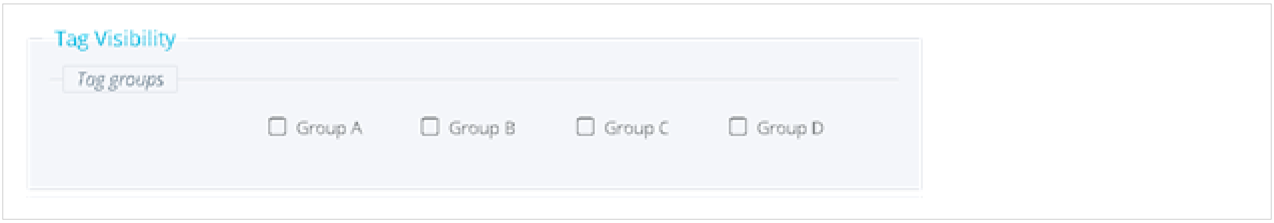
The group filter can be used to export the data for the tags belonging to one or more Tag Groups.

Example:

```
ACX or BDAX or X
```

- If the filter includes A, B, C, or D, then only the tags belonging to those groups are included in the output.
- An additional filter of **X** is available for historical records. If the filter includes X, then tags without historical logging enabled are also output.
  This is provided in case recording has been disabled but tags have been previously recorded in the file.

Group filter can be added during the TAG creation:



2.11. $fn [FILE NAME]

This field is used for specifying a file name for the export data (destination name).

Usually this file name is used to specify the output of the data, for example when sending an attachment to an email. In this case, the **$fn** file name gives the name of the attachment.

When doing a PUTFTP, then **$fn** does not need to be specified. The PUTFTP command manages the name of the destination file.

Example:

```
PUTFTP "MyFileWithANewName.txt", "[$dtUF $uf/myfile.txt]"
```

3. Data Type Descriptions and Syntax

A Data type defines what is exported from the Ewon.

The data type is defined by the **$dt** field followed by *uppercase letters*.

The **$dt** field is mandatory for each Export Block Descriptor. Usually the *$ft* (format) is also present to define the output format of your data although a default format is defined for each data type.

For each Data type, a set of other fields may be provided. Some are mandatory and others are optional.

NOTICE
If you specify an unused field (neither mandatory nor optional), it will then be ignored.

This section will describe the syntax for each data type with the specific features for each of them.

3.1. $dtAH [Alarm History]

3.1.1. Export Content

The Alarm History outputs historical alarm data from the File system for **one** or **all** the Tags.

The output format can be Text or HTML Table.

A time range can also be specified for this export.

3.1.2. Detailed Example

```
$dtAH $ftH $st01012001 $tsL
```

$dtAH	Data Type Alarm History Logging
$ftH	Output Format requested is HTML Table
$st01012001	1st of January 2001
$et	If not specified > until the end of file
$tn	If not specified > all the Tags
$tsL	Format the time string in ISO 8601 local (local time format)

3.1.3. Used Fields

Fields	Value if not specified	Mandatory	Optional
$dt		X	
$ft	Text		X
$st	01/01/1970		X
$et	31/12/2030		X
$ts	No modifier		X
$tn	All Tags		X
$ut	No Time Update		X
$fn	Export Block Descriptor		X
$ct	No Compression		X

3.1.4. Special Parameters and Fields

$st [start Time] $et [End Time]

- If Last time is specified (**$stL** or **$etL**): there is a last time logged for each Tag and a last time logged for all Tags.
- If you specify a given Tag, its own last time will be used.
- If a specific Tag is not requested, the export is performed for all tags and a global last time is used.

$ft [Data Format]

Acceptable values		
Text	HTML	

- The Text format will output a comma-separated file. The separator is '**;**' to avoid any confusion with decimal points.
- If all the Tags are output, they will be output in a chronological order in the file.
  Line content of output file: **"EventDate";"TagName";"Status";"UserAck";"Description"**

$ts [Timestamp Format]

The **$ts** field determines whether the event dates are presented in the Ewon device's local time or UTC and the timestamp's format.

Modifier	EventDate	Example
No modifier	Local Time	"EventDate";"TagName";"Status";"Type";"UserAck";"Description" "19/09/2018 09:59:11";"Tag_alarm1";"ALM";"LO";"";"" "19/09/2018 10:02:21";"Tag_alarm1";"ALM";"LO";"";""
$tsO	UTC Time	"EventDate";"TagName";"Status";"Type";"UserAck";"Description" "19/09/2018 07:59:11";"Tag_alarm1";"ALM";"LO";"";"" "19/09/2018 08:02:21";"Tag_alarm1";"ALM";"LO";"";""
$tsU	UTC Time	"EventDate";"TagName";"Status";"Type";"UserAck";"Description" "19/09/2018 07:59:11";"Tag_alarm1";"ALM";"LO";"";"" "19/09/2018 08:02:21";"Tag_alarm1";"ALM";"LO";"";"" "2018-09-19T08:02:21Z";"Tag_alarm1";"ALM";"LO";"";""
$tsL	Local Time	"EventDate";"TagName";"Status";"Type";"UserAck";"Description" "2018-09-19T09:59:11+0200";"Tag_alarm1";"ALM";"LO";"";"" "2018-09-19T10:02:21+0200";"Tag_alarm1";"ALM";"LO";"";""

$tn [Tag Name]

- If this Tag is not specified, **all the Tags** will be selected for export.
- If this Tag is specified, the Tag with the given name will be selected.

3.2. $dtAR [Alarm Real-time]

3.2.1. Export Content

The **Alarm Real-Time** outputs the Alram Real-Time data for **one or all** the Tags.

The output format can be **Text or HTML Table**.

If only **one** Tag is specified, 1 or 0 lines will be appended to the output header line (Time range is not applicable here).

3.2.2. Detailed Example

```
$dtAR $ftT $tsL
```

$dtAR	Data Type Alarm Real Time
$ftT	Output format requested is CSV
$tn	If not specified > all the Tags
$tsL	Format the time string in ISO 8601 local (local time format)

3.2.3. Used Fields

Fields	Value if not specified	Mandatory	Optional
$dt		X	
$ft	Binary		X
$tn	All Tags		X
$ts	No modifier		x
$fn	Export Block Descriptor		X
$ct	No Compression		X

3.2.4. Special Parameters and Fields

$ft [Data Format]

Acceptable values		
Text	HTML	

- The Text format will output a comma-separated file. The separator is '**;**' to avoid any confusion with decimal point.
- If all the Tags are output, they will be output in a chronological order in the file.
  Line content of output file:
  "TagId";"AlarmTime";"TagName";"AlStatus";"AlType";"StatusTime";"UserAck";"Description"

$tn [Tag Name]

- If this Tag is not specified, **all the Tags** will be selected for export.
- If this Tag is specified, the Tag with the given name will be selected.

$ts [Timestamp Format]

The result of this EBD contains 2 time elements:

- **AlarmTime**: timestamp of the beginning of alarm in string format
- **StatusTime**: timestamp of the AlStatus that is currently shown

Modifier	AlarmTime	StatusTime	Example
No modifier	Local Time	Local Time	"TagId";"AlarmTime";"TagName";"AlStatus";"AlType";"StatusTime";"UserAck";"Description";"AlHint" 54;"19/09/2018 13:34:18";"Tag_alarm1";"ALM";"HIHI";"19/09/2018 13:34:18";"";"";""
$tsO	UTC Time	UTC Time	"TagId";"AlarmTime";"TagName";"AlStatus";"AlType";"StatusTime";"UserAck";"Description";"AlHint" 54;"19/09/2018 11:34:18";"Tag_alarm1";"ALM";"HIHI";"19/09/2018 11:34:18";"";"";""
$tsU	UTC Time	UTC Time	"TagId";"AlarmTime";"TagName";"AlStatus";"AlType";"StatusTime";"UserAck";"Description";"AlHint" 54;"2018-09-19T11:34:18Z";"Tag_ alarm1";"ALM";"HIHI";"2018-09-19T11:34:18Z";"";"";""
$tsL	Local Time	Local Time	"TagId";"AlarmTime";"TagName";"AlStatus";"AlType";"StatusTime";"UserAck";"Description";"AlHint" 54;"2018-09-19T13:34:18+0200";"Tag_ alarm1";"ALM";"HIHI";"2018-09-19T13:34:18+0200";"";"";""

3.3. $dtES [Export Estat]

3.3.1. Export Content

$dtES exports the **estat.htm** file. This file lists the current status from the main Ewon features.

The output format can be **Text** or **HTML**.

3.3.2. Detailed Example

```
sendmail "user@user.be","","eWON estat file","&amp;
[$dtES$ftH$fnestat.htm]"
```

$dtES	Data Type Export Estat
$ftH	Will export the file in htm format
$fn	Will give to the file the required name

Will attach the Ewon *estat.htm* file to the email.

3.3.3. Used Fields

Fields	Value if not specified	Mandatory	Optional
$dt		X	
$ft	Text		X
$fn	Export Block Descriptor		X

3.4. $dtEV [Event File]

3.4.1. Export Content

The **Event File** outputs event data from the **File System**.

The output format can be **Text or HTML Table**. A time range can also be specified for this export.

3.4.2. Detailed Example

```
$dtEV $ftT $st_m30 $tsL
```

$dtEV	Data Type Alarm Real Time
$ftT	Output format requested is CSV
$st_m30	Last 30 minutes
$et	If not specified > until Now
$tsL	Format the time string in ISO 8601 local (local time format)

3.4.3. Used Fields

Fields	Value if not specified	Mandatory	Optional
$dt		X	
$ft	Text		X
$st	01/01/1970		X
$et	31/12/2030 < > NOW		X
$ts	No modifier		X
$fn	Export Block Descriptor		X
$ct	No Compression		X

3.4.4. Special Parameters and Fields

$ft [Data Format]

Acceptable values		
Text	HTML	

- The Text format will output a comma-separated file. The separator is ’**;**’ to avoid any confusion with decimal point.
- Line content of output file: **"EventTimeInt";"EventTimeStr";"Event"**

EventTimeInt	Time provided as an integer (number of seconds since 1/1/1970)
EventTimeStr	Date and Time as Text

$ts [Timestamp]

This EBD exports 2 time elements:

- **EventTimeInt:** Integer format (number of seconds since 01/01/1970 - UNIX Epoch time).
- **EventTimeStr:** Timestamp in string format (syntax depending on $ts modifier).

The **$ts** field impacts the format of the EventTimeStr element.

The difference between the **TimeInt** presentation and the **TimeStr** presentation will vary depending on whether the Ewon device is configured to “*Record data in UTC*”.

Table 1. Event File with “Record data in UTC” **enabled**:

Modifier	EventTimeInt	EventTimeStr	Example
No modifier	UTC Time	Local Time	"EventTimeInt";"EventTimeStr";"EventStr";"ThreadStr";"- ThreadId";"Event" 1537359559;"19/09/2018 14:19:19";"main-Real Time Clock updated";"unact";79301;1073762139
$tsO	UTC Time	UTC Time	"EventTimeInt";"EventTimeStr";"EventStr";"ThreadStr";"- ThreadId";"Event" 1537359559;"19/09/2018 12:19:19";"main-Real Time Clock updated";"unact";79301;1073762139
$tsU	UTC Time	UTC Time	"EventTimeInt";"EventTimeStr";"EventStr";"ThreadStr";"- ThreadId";"Event" 1537359559;"2018-09-19T12:19:19Z";"main-Real Time Clock updated";"unact";79301;1073762139
$tsL	UTC Time	Local Time	"EventTimeInt";"EventTimeStr";"EventStr";"ThreadStr";"- ThreadId";"Event" 1537359559;"2018-09-19T14:19:19+0200";"main-Real Time Clock updated";"unact";79301;1073762139.

Table 2. Event File with “Record data in UTC” **not enabled**:

Modifier	EventTimeInt	EventTimeStr
No modifier	Local Time	Local Time
$tsO	Local Time	UTC Time
$tsU	Local Time	UTC Time
$tsL	Local Time	Local time

3.5. $dtHL [Historical Logging]

3.5.1. Export Content

The Historical logging outputs the historical data from the File system for numeric (Analog and Boolean) tags.

String tags are not included.

The output format can be TEXT, HTML Table, or BINARY.

The **GRAPH** format is also available **if only one Tag** is specified.

A time range can also be specified for this export.

3.5.2. Detailed Example

```
$dtHL $ftT $st_h4 $et_m0 $tsL $tnA1 $ic
```

$dtHL	Data Type Historical Logging
$ftT	Output Format requested is CSV
$st_h4	Start Time is Current Time – 4 hours
$et_0	End Time is Current Time – 0 minutes < > NOW
$tsL	Format the time string in ISO 8601 local (local time format)
$tnA1	Tagname “A1” History to Output
$ic	Include Intra Sec Counter

3.5.3. Used Fields

Fields	Value if not specified	Mandatory	Optional
$dt		X	
$ft	Binary		X
$st	01/01/1970		X
$et	31/12/2030		X
$ts	No modifier		X
$tn	All Tags		X
$fl	All Tags Groups		X
$ut	No Time Update		X
$fn	Export Block Descriptor		X
$ct	No Compression		X
$ic	Do not output Intra Sec Counter		X

3.5.4. Special Parameters and Fields

$st [Start Time] $et [End Time]

- If Last time is specified (**$stL** or **$etL**): there is a last time logged for each Tag and a last time logged for all Tags.
- If you specify a given Tag, its own last time will be used.
- If a specific Tag is not requested, the export is performed for all Tags with historical logging and a global last time is used.
- If the output format is graph: **$et_0** should be used instead of default value, otherwise the graph would span up to 31/12/2030.
- For *binary* or *text output*, the default value can be kept.

$ts [Time Format]

The **$dtHL** exports 2 time elements:

- **TimeInt:** integer format (number of seconds since 01/01/1970 - UNIX Epoch time).
- **TimeStr:** timestamp in string format (syntax depending on $ts modifier).

The **$ts** field impacts the format of the **TimeStr** element.

The difference between the **TimeInt** presentation and the **TimeStr** presentation will vary depending on whether the Ewon device is configured to "*Record data in UTC*".

Table 3. Historical Logging with "Record data in UTC" **enabled**:

Modifier	TimeInt	TimeStr	Example
No modifier	UTC Time	Local Time	"TimeInt";"TimeStr";"IsInitValue";"Value";"IQuality" 1537354078;"19/09/2018 12:47:58";0;556;3 1537354092;"19/09/2018 12:48:12";0;500;3
$tsO	UTC Time	UTC Time	"TimeInt";"TimeStr";"IsInitValue";"Value";"IQuality" 1537354078;"19/09/2018 10:47:58";0;556;3 1537354092;"19/09/2018 10:48:12";0;500;3
$tsU	UTC Time	UTC Time	"TimeInt";"TimeStr";"IsInitValue";"Value";"IQuality" 1537354078;"2018-09-19T10:47:58Z";0;556;3 1537354092;"2018-09-19T10:48:12Z";0;500;3
$tsL	UTC Time	Local Time	"TimeInt";"TimeStr";"IsInitValue";"Value";"IQuality" 1537354078;"2018-09-19T12:47:58+0200";0;556;3 1537354092;"2018-09-19T12:48:12+0200";0;500;3

Table 4. Historical Logging with "Record data in UTC" **not enabled**:

Modifier	TimeInt	TimeStr
No modifier	Local Time	Local Time
$tsO	Local Time	UTC Time
$tsU	Local Time	UTC Time
$tsL	Local Time	Local Time

$ft [Data Format]

Acceptable values			
Binary	Text	HTML	Graph

- The Graph format is only allowed if a Tag has been specified.
- The Text format will output a comma-separated file. The separator is '**;**' to avoid any confusion with decimal points.
- If all the Tags are output, they will be output in a chronological order in the file.

$ut [Update Time]

- If only one Tag is specified, the time of the last point for that Tag will be memorized.
- All the Tags can be output individually and last time is saved for each point.
- Another memory is available if **$ut** is requested for ALL the Tags.

$tn [Tag Name]

- If this Tag is not specified, **all the Tags** will be selected for export.
- If this Tag is specified, the Tag with the given name will be selected.

$fl [Group Filter]

The group selection is only available with binary, text and HTML formats (not allowed for Graphic format **$ftG**).

$ic [Intrasecond Counter]

The **IntraSecCounter** is only available with binary, text, and HTML formats (not allowed for Graphic format $ftG).

Table 5. [$dtHL] examples:

$dtHL	Export all the Tags records in Binary format.
$dtHL $ftT	Export all the Tags records in ($ftT) Text format (likeCSV file).
$dtHL $ftT $tnTemp	Export all the values of the ($tnTemp) Tag named “Temp” in ($ftT) Text format.
$dtHL $ftB $flAB	Export all the values of ($flAB) tags belonging to group A and B in ($ftB) Binary format.
$dtHL $ftT $tnTemp $st_h1 $et_s0 $tsU	Export all the Tag records ($st_h1) from 1 hour to ($et_s0) with the TimeStr field in ISO 8601 ZULU (UTC time) format.
$dtHL $ftT $st_h1 $et_s0	Export the values ($st_h1) from 1 hour to ($et_s0) **now**.
$dtHL $ftT $tnTemp $st_h1 $et_s0 $ic	Export the values ($st_h1) from 1 hour to ($et_s0) now of ($tnTemp) Tag named “Temp” in ($ftT) Text format ($ic) including the intra sec counter.
$dtHL $ftT $flCD $st_h1 $et_s0	Export the values ($st_h1) from 1 hour to ($et_s0) now of ($flCD) Tags belonging to group C and D in ($ftT) Text format.

3.5.5. Binary Data Structure

Binary File Format for Historical Logging Values Tags:

- **ircall.bin:**
  The *ircall.bin* file contains the binary values of all recorded Tags defined in the Ewon device.
  This file is an image of the Tag memory of the Ewon device and is present on every device type.

IMPORTANT
The processor of the Ewon uses the Big Endian format for memory access (most significant byte first) like Motorola processor. Thus, this access method is also applicable for the *ircall.bin* file. In PC world (Intel processor), the access method is Little Endian! We need to reverse all bytes read (and words if necessary) from files in order to correctly interpret it in a PC program

Table 6. Header

Length (bits)	Field Name	Description
#16	Firmware Minor	Firmware version of the Ewon (Minor)
#16	Firmware Major	Firmware version of the Ewon (Major)
#16	Unused	Unused data (Dummy)
#16	Record Size	Length of the record structure

This structure contains 4 short integers (16bits):
- The firmware version of the Ewon (Major and Minor)
- Unused data (Dummy)
- The length of the record structure (RecordSize).
Example of implementation in C++:

```
typedef struct
{
time_t LogTime;
unsigned shortIntraSecCounter;
unsigned short MSec;
unsigned int InitValue:1;
unsigned int TagId:31;
float TagValue;
}
HistoricalRecord_t;
```

After the Header, the Ewon data can be found. Each record is encoded in a structure of 16 bytes defined as follows:

Table 7. Data Element

Length (bits)	Field Name	Description
32	LogTime	Integer format (number of seconds since 01/01/1970 - UNIX Epoch time). See $ts (page 5) section.
2	TagQuality	Quality of the Tag 0: Bad 1: Uncertain 3: Good
4	TagType	Tag type 0: Boolean 1: Float32 2: Integer32 3: unsignedInteger32
10	MSec	Set to 0. *deprecated: This was the number of MSec to add to LogTime in order to get the complete timestamp of the record.*

Length (bits)	Field Name	Description
16	IntraSecCounter	This value is incremented for each point logged during the same second (incremented even if TagId is different).
31	TagID	Unique ID of the Tag (== value in Var_lst.txt), never the same for 2 tags, even if tag deleted.
1	InitValueFlag	TRUE if the point was log due to a restart of the system.
32	TagValue	Actual value logged at the beginning of the intervalCoded as Float32, Integer32 or Unsigned32.

Example of implementation in C++:

```
typedef struct
{
time_t LogTime;
unsigned shortIntraSecCounter;
unsigned short MSec:10;
unsigned short Type:4;
unsigned short IrcQuality:2;
unsigned int InitValue:1;
unsigned int TagId:31;
float TagValue;
}
HistoricalRecordV6_t;
```

NOTICE
Due to the *BigEndian to LittleEndian* swap of bytes and Words, the structure elements are in reverse order.

3.6. $dtHS [Historical String]

3.6.1. Export Content

The **Historical Strings** outputs the historical data from the File system for String tags.

The output format can be **Text, HTML Table, or Binary**.

3.6.2. Detailed Example

```
$dtHS $ftT $st_h4 $et_m0 $tsL $tnA1 $ic
```

$dtHS	Data Type Historical Strings
$ftT	Output format requested is CSV
$st_h4	Start Time is Current Time – 4 hours
$et_0	End Time is Current Time – 0 minutes <> NOW
$tnA1	If not specified > all the Tags
$tsL	Format the time string in ISO 8601 local (local time format)
$ic	Include Intra Sec Counter

3.6.3. Used Fields

Fields	Value if not specified	Mandatory	Optional
$dt		X	
$ft	Binary		X
$st	01/01/1970		X
$et	31/12/2030		X
$tn	All Tags		X
$ts	No modifier		x
$fl	All Tag Groups		X
$fn	Export Block Descriptor		X
$ct	No Compression		X
$ic	Do not output IntraSec Counter		X

3.6.4. Special Parameters and Fields

$ft [Data Format]

Acceptable values		
Binary	Text	HTML

- In the Text format, values of strings are surrounded by double quotes (**“**). Special characters (quote, double quote, non-printable) are escaped with a ‘****’.
- If all the Tags are output, they will be output in a chronological order in the file.

$st [Start Time] $et [End Time]

Last time (**$stL** or **$etL**) and update time (**$ut)** is *not available* for Historical Strings

$tn [Tag Name]

- If this Tag is not specified, **all the Tags** will be selected for export.
- If this Tag is specified, the Tag with the given name will be selected.

$ts [Time Format]

The **$dtHS** exports 2 time elements.

- **TimeInt:** integer format (number of seconds since 01/01/1970 - UNIX Epoch time).
- **TimeStr:** timestamp in string format (syntax depending on **$ts** modifier).

The **$ts** field impacts the format of the **TimeStr** element.

The difference between the **TimeInt** presentation and the **TimeStr** presentation will vary depending on whether the Ewon device is configured to “*Record data in UTC*”.

Table 8. Historical Strings with “Record data in UTC” **enabled**:

Modifier	AlarmTime	StatusTime	Example
No modifier	UTC Time	Local Time	"TimeInt";"TimeStr";"IsInitValue";"Value";"IQuality" 1537354078;"19/09/2018 12:47:58";0;"Message1";3 1537354092;"19/09/2018 12:48:12";0;"Message2";3
$tsO	UTC Time	UTC Time	"TimeInt";"TimeStr";"IsInitValue";"Value";"IQuality" 1537354078;"19/09/2018 10:47:58";0;"Message1";3 1537354092;"19/09/2018 10:48:12";0;"Message2";3
$tsU	UTC Time	UTC Time	"TimeInt";"TimeStr";"IsInitValue";"Value";"IQuality" 1537354078;"2018-09-19T10:47:58Z";0;"Message1";3 1537354092;"2018-09-19T10:48:12Z";0;"Message2";3
$tsL	UTC Time	Local Time	"TimeInt";"TimeStr";"IsInitValue";"Value";"IQuality" 1537354078;"2018-09-19T12:47:58+0200";"Message1";3 1537354092;"2018-09-19T12:48:12+0200";0;"Message2";3

Table 9. Historical Logging with “Record data in UTC” **not enabled**:

Modifier	TimeInt	TimeStr
No modifier	Local Time	Local Time
$tsO	Local Time	UTC Time
$tsU	Local Time	UTC Time
$tsL	Local Time	Local Time

$fl [Group Filter]

The filter can be used as for the instant values (**$dtIV**), with an additional option '**X**'.

- The $fl can be any string containing A, B, C, D, X.
  Example: ACX or BDAX or X
- If no filter is specified, all the Tags with an enabled Historical Logging are output.
- If filter includes "**X**", then Tags without Historical Logging enabled are also included, this is provided in case recording has been disabled but Tags have been previously recorded in the file.
- If filter includes any of the "**A, B, C, D**", then only the Tags that belong to those groups are included in the output.

Table 10. [$dtHS] examples:

$dtHS	Export all the String Tags records in Binary format
$dtHS $ftT	Export all the String Tags records in ($ftT) Text format
$dtHS $ftT $tnMessage	Export all the values of the ($tnMessage) Tag named “Message” in ($ftT) Text format
$dtHS $ftB $flAB	Export all the values of ($flAB) string tags belonging to group A and B in ($ftB) Binary format
$dtHS $ftT $st_h1 $et_s0 $tsU	Export all the Tag records ($st_h1) from 1 hourto ($et_s0) **now** with (**$tsU**) the TimeStr field in ISO 8601 ZULU (UTC time) format.
$dtHS $ftT $st_h1 $et_s0	Export the string tag values ($st_h1) from 1 hour to ($et_s0) **now**.
$dtHS $ftT $tnMessage $st_h1 $et_s0 $ic	Export the values ($st_h1) from 1 hour to ($et_s0) now of ($tnMessage) Tag named “Message” in ($ftT) Text format ($ic) including the intra sec counter
$dtHS $ftT $flCD $st_h1 $et_s0	Export the values ($st_h1) from 1 hour to ($et_s0) now of ($flCD) string Tags belonging to group C and D in ($ftT) Text format

3.6.5. Binary Data Structure

Binary File Format for Historical Strings Values Tags:

Table 11. FWVersion = 0x11223344

Field Name	Byte	Value
FWVersion	#0	0x11
	#1	0x22
	#2	0x33
	#3	0x44

Table 12. Header

Byte Start Position	Length	Field Name	Description
#0	4	FWVersion	
#4	4	FileFormatVersion	
#8	Data Table String Tags		

- First element starts at **byte #8**

Table 13. Data Element

Byte Start Position	Length	Field Name	Description
#0	4	LogTime	Integer format (number of seconds since 01/01/1970 - UNIX Epoch time). See $ts (page 5).
#4	2	IntraSecCounter	This value is incremented for each point logged during the same second (incremented even if TagId is different).
#6	4	TagID	Unique ID of the Tag (= value in Var_lst.txt), never the same for 2 tags, even if tag deleted.
#10	1	TagQualty	Quality of the Tag 0: Bad 1: Uncertain 3: Good
#11	0: Bad 1: Good	InitValueFlag	TRUE if the point was log due to a restart of the system.
#12	4	StringSize	
#16	StringSize	StringByteStream	

3.7. $dtHT [Historical Table]

3.7.1. Export Content

The Historical Table is a representation of the IRCALL.BIN (incremental recording).

This representation provides a recordings representation as a table where columns are Tag names and rows are recording times.

3.7.2. Detailed Example

```
$dtHT $ftT $st_h4 $et_m0 $tsL $flAB $in10
```

$dtHT	Data Type Historical Table
$ftT	Output Format requested is CSV
$st_h4	Start Time is Current Time – 4 hours
$et_0	End Time is Current Time – 0 minutes < > NOW
$tsL	Format the time string in ISO 8601 local (local time format)
$flAB	Filter to Instant Value groups A and B
$in10	Interval fixed to 10 seconds

3.7.3. Used Fields

Fields	Value if not specified	Mandatory	Optional
$dt		X	
$st	01/01/1970		X
$et	31/12/2030		X
$ts	No modifier		X
$fl	All Tags are displayed		X
$in	Interval from the *ircall.bin* file		X
$ct	No Compression		X

3.7.4. Special Parameters and Fields

$ft [Data Format]

Acceptable values		
Text	HTML	

- Text format will output a comma-separated file. The separator is '**;**' to avoid confusion with decimal point.

$fl [Group Filter]

- The filter can be used like for the instant values (**$dtIV)**, with an additional option '**X**'
- The **$fl** can be any string containing A, B, C, D, X.
  Example: ACX or BDAX or X
- If no filter is specified, all the Tags with an enabled Historical Logging are output.
- If filter includes "**X**", then Tags without Historical Logging enabled are also included, this is provided in case recording has been disabled but Tags have been previously recorded in the file.
- If filter includes any of the "**A, B, C, D**", then only the Tags that belong to those groups are included in the output.

$in [Fixed Interval]

The Historical Table allows you to present the data at fixed intervals of time rather than including every recorded time.

For fixed interval, **$in** parameter must be used. The interval is defined in *seconds*.

Example: **$in10** to output one value every 10 seconds

If **$in** is not specified, then the output time is defined by the time in the recording file.

Example:

Let's assume that we have 2 Tags logged with the following time and values (for clarity the date has been omitted):

Time	Tag	Value
10:01:00	Tag 1	1
10:10:00	Tag 1	1.5
10:10:00	Tag 2	1
10:11:00	Tag 1	2
10:12:00	Tag 1	3
10:21:00	Tag 2	2
10:30:00	Tag 1	4

1. If the fixed interval is not requested, then the following output will be produced

	Time	Tag 1	Tag 2
1	10:01:00	1	Undef
2	10:10:00	1.5	1
3	10:11:00	2	1
4	10:12:00	3	1
5	10:21:00	3	2
6	10:30:00	4	2

NOTICE
At line 1: Tag2 is Undef, because no values are available in the log file.

At line 2: Tag1 and Tag2 are updated on the same line, although there are 2 records in the incremental recording file, only 1 line is produced.

When no interval is specified - except for the case when multiple Tags changed at the same time- the output contains one line for every record that has been logged.

2. If an interval of 10 minutes has been requested (**$in600**), then the following output would be produced.

	Time	Tag 1	Tag 2
1	10:01:00	1	Undef
2	10:11:00	2	1
3	10:21:00	3	2

NOTICE
The output starts with the first time found in the file then it increases by 10 minutes.

There is no record with time equal (or higher) to 10:31, so the last line is 10:21.

3. If an interval of 10 minutes is requested and the start time is 10:00, then the following output would be produced.

	Time	Tag 1	Tag 2
1	10:00:00	Undef	Undef
2	10:10:00	1.5	1
3	10:20:00	3	1
4	10:30:00	4	2

NOTICE
On the first line, no values are available for Tag1 or Tag2 before 10:01:00 (for tag Tag1) in the recording file, so the values are Undef.

3.8. $dtIS [Instant String Values]

3.8.1. Export Content

The **Instant String Values** exports the current values of **String** tags.

The output is only available in **Binary** format.

Table 14. The Instant String Values file contains the following information for each Tag:

TagID	ID of the Tag
AlStatus	Current alarm status of the Tag
Altype	Type of the current alarm
TagQuality	0x0000:Bad 0x0040:Uncertain 0x00C0:Good
StringSize	String Size
StringByteStream	String Value

3.8.2. Detailed Example

$dtIS $flA $fnMyStrings.bin

$dtIS	Data Type Instant String Values
$flA	Include only tags from Tag group A
$fn	Output file named MyStrings.bin

NOTICE
Alarming is currently not supported by the String Tags.

3.8.3. Used Fields

Fields	Value if not specified	Mandatory	Optional
$dt		X	
$ft	Binary		X
$fl	All tags		X
$fn	Export Block Descriptor		X

3.8.4. Binary Data Structure

Binary File Format for Instant String values Tags:

Endianness: Big endian (MSB First).

Examples:

Table 15. FWVersion = 0x11223344

Field Name	Byte	Value
FWVersion	#0	0x11
	#1	0x22
	#2	0x33
	#3	0x44

Table 16. Header

Byte Start Position	Length	Field Name	Description
#0	4	FWVersion	
#4	4	FileFormatVersion	
#8	Data Table String Tags		

Table 17. Data Element

Byte Start Position	Length	Field Name	Description
#0	4	TagID	
#4	4	AlStatus	**Alarm status :** 0: None 2: ALM 3: ACK 4: RTN
#8	4	AlType	**Alarm Type:** 0: None 1: High 2: Low 3: Level 4: HighHigh 5: LowLow
#12	2	TagQuality	0x0000: Bad 0x0040: Uncertain 0x00C0: Good
#14	2	Reserved	
#16	4	StringSize	
#20	StringSize	StringByteStream	

3.9. $dtIV [Instant Values]

1. Instant values - General information

   Instant Values means values of the Tags at the current time. The instant values file contains the following information for each Tag:

TagId	ID of the Tag
TagName	Name of the Tag (in text mode)
Value	Current value of the Tag
AlStatus	Current alarm status of the Tag
AlType	Type of the current alarm

   - The file containing the **Instant Values** for every Tag is available in **text format**.
   - The **binary format** instant value file only contains the Instant Values for numerical tags. To export String Tags, you must use the Instant Value Strings (**$dtIS**) data type.
   - The Instant Values file normally contains **all the Tags**, but there is an *additional feature* that allows obtaining only the Instant Values from *specific Tags*.

   a. Alarm status code values:

      The table below lists the different values that the field **AlStatus** can have; depending on the Alarm State and of the action the user has performed on it:

Alarm Status	Alarm Status Value	Alarm status explanations
NONE	0	Tag is not in alarm status
PRETRIGGER	1	Tag is in pretrigger alarm statusWarning: we assume there is no alarm if AlStatus value <= Alarm Pretrigger
ALM	2	Tag’s alarm status is active
ACK	3	Tag’s alarm has been acknowledged
RTN	4	Tag’s alarm returns from an active status

   b. Alarm type values

      The table below lists the different values that the field AlType can have:

Alarm Type	Alarm Type Value	Alarm type explanations
NONE	0	The Tag value is inside of the limits beyond which the alarm is triggered
HIGH	1	The Tag value **exceeds** the value entered in the **Alarm Level High** field from the Tag configuration page
LOW	2	The Tag value is **less** than the value entered in the **Alarm Level Low** field from the Tag configuration page
LEVEL	3	The Tag value **matches** the **Boolean Alarm Level** value defined in the Tag configuration page
HIGH_HIGH	4	The Tag value **exceeds** the value entered in the **Alarm Level HighHigh** field from the Tag configuration page
LOW_LOW	5	The Tag value is **less** than the value entered in the **Alarm Level LowLow** field from the Tag configuration page

3.9.1. Export Content

The **$dtIV Tag** exports either the entire content of the **Instant Value** file (txt or binary format) or only a part of it, depending on the parameters that have been defined with the **$fl** field.

3.9.2. Detailed Example

```
$dtIV $flAB
```

$dtIV $flAB	Will export all the Tags belonging to group A or B
$dtIV $flA	Will export all the Tags belonging to group A
$dtIV $fl	Will export no Tag (useless)
$dtIV $flABCD	Will export all the Tags belonging to group A or B or C or D (but missing Tags that belong to no group)
$dtIV	Will export all the Tags regardless of group definition

Binary file format:

3.9.3. Used Fields

Fields	Value if not specified	Mandatory	Optional
$dt		X	
$fl	All tags		X
$ft	Text		X
$fn	Export Block Descriptor		X

```
$ft
```

Acceptable Values	
Binary	Text

The Text format includes both numeric and string tags.

The values of strings are surrounded by double quotes (**"**). Special characters (quote, double quote, non-printable) are escaped with a '\'.

```
"TagId";"TagName";"Value";"AlStatus";"AlType";"Quality"
1;"Float_0";5;0;0;65472
2;"Float_1";6;0;0;65472
3;"Float_2";7;0;0;65472
2003;"Str_100";"ABCDEF\"GH\'IJKLMNOPQRSTUVWXYZ";0;0;65472
2004;"Str_101";"ABCDEFGHIJKLMNOPQRSTUVWXYZ";0;0;65472 2005;"Str_102";"AT
char @";0;0;65472
```

The Binary format includes only numeric tags.

$fl [Group or Groups]

The **$fl** (for filter) field must be directly followed by a list of one or more groups A, B, C or D (that have been checked in the Tag's configuration).

There must be no other character in the filter and all the groups must be in uppercase.

Example:

```
$dtIV $flAB
```

It will export all the Tags belonging to group A or B.

3.9.4. Binary Data Structure

Binary File Format for Instant Values Tags:

The **inst_val.bin** file contains the current values of all tags defined in the Ewon.

The file starts with a **Header** of 20 bytes that can be represented by the following **C** structure:

Table 18. Header

Byte Start Position	Field Name	Description
Revision #32	Revision (32 bits)	Revision of the inst_val file: **1**: before firmware 6 **2**: and above: since firmware 6
RecordSize #32	RecordSize (32 bits)	Size of the Record structure representing each tags information
NumberOfTag #32	NumberOfTags (32 bits)	Number of tags recorded in inst_val file
RecFlag #32	RecFlag (32 bits)	Internal use
Reserved #32	Reserved (32 bits)	Internal use

Table 19. Data Element

Byte Start Position	Field Name	Description
TagId #32	TagID (32 bits)	The tag ID
TagValue #32	TagValue (32 bits)	The tag value
AlarmStatus #32	AlarmStatus (32 bits)	Depending of the TagType it can be coded as: - Float32 - Integer32 - Unsigned32
AlarmType #32	AlarmType (32 bits)	
TagType #16	TagType (16 bits)	
TagQuality #16	TagQuality (16 bits)	Quality of the Tag 0: Bad 1: Uncertain 3: Good

3.10. $dtKPI [Key Performance Indicators]

3.10.1. Export Content

The **$dtKPI** exports the current value of those tags configured as KPI tags on the Ewon device.

Tags can be marked as KPI tags on the *Tag Setup Screen*.

3.10.2. Detailed Example

$dtKPI$ftJ

$dtKPI	Data Type Key Performance Indicators
$ftJ	Output Format requested is JSON

3.10.3. Used Fields

Fields	Value if not specified	Mandatory	Optional
$dt		X	
$ft	JSON		X
$fn	Export Block Descriptor		X
$ct	No Compression		X

3.10.4. Special Parameters and Fields

$ft [Data Format]

Acceptable Values		
JSON	Text	HTML

The KPI data type includes a special output file type of JSON.

Example output in JSON format:

```
{
"label": [
"Name",
"Value",
"Description"
],
"model": [
[
"OEE",
"92.099998",
""
],
[
"Parts_produced",
"86",
"Good parts"
    ]
  ]
}
```

3.11. $dtPP [Dump PPP]

3.11.1. Export Content

$dtPP exports the *dump.ppp* file (binary format).

The output format can only be of **Binary** type.

TIP
Rename it **.pcap** if you want to open it using *Wireshark*.

3.11.2. Detailed Example

```
sendmail "user@user.be","","eWON PPP dump","&[$dtPP$fndump.ppp]"
```

$dtPP	Data Type Dump PPP
$fn	Will give the required name to the file

3.11.3. Used Fields

Fields	Value if not specified	Mandatory	Optional
$dt		X	
$fn	Export Block Descriptor		X

3.12. $dtRE [Real Time Diagnostic]

3.12.1. Export Content

$dtRE exports the **Real Time Diagnostic data** (equivalent to the real-time log).

The output format can be **Text** only.

3.12.2. Detailed Example

```
sendmail "user@user.be","","eWON Real Time Log","&[$dtRE$fndiag.txt]"
```

$dtRE	Data type Real Time Diagnostic
$fn	Will give to the file the required name

It will attach to an email the file *diag.txt* holding the Real Time Diagnostic of the Ewon.

3.12.3. Used Fields

Fields	Value if not specified	Mandatory	Optional
$dt		X	
$ft	Text		X
$fn	Export Block Descriptor		X
$ts	No modifier		X

3.13. $dtRL [Real time Logging]

3.13.1. Export Content

The Real-time logging outputs the real-time log data from the File system for **one** Tag.

The output format can be Text, HTML Table, Binary or Graph.

A time range can also be specified for this export.

3.13.2. Detailed Example

```
$dtRL $ftT $st_h4 $et_m0 $tsL $tnA1
```

$dtRL	Data Type Real Time Logging
$ftG	Output Format requested is GRAPH
$st_m10	Start Time is Current Time – 10 minutes
$et_0	End Time is Current Time – 0 minutes < > NOW
$tsL	Format the time string in ISO 8601 local (local time format)
$tnA1	Tagname “A1” History to Output
A1	Name of the Tag

3.13.3. Used Fields

Fields	Value if not specified	Mandatory	Optional
$dt		X	
$tn		X	
$ft	Binary		X
$st	01/01/1970		X
$et	31/12/2030		X
$ut	No Time Update		X
$fn	Export Block Descriptor		X
$ct	No Compression		X

3.13.4. Special Parameters and Fields

$st [Start Time] $et [end Time]

- If the output format is "**graph**", **$et_0** should be used instead of default value, otherwise the graph would span up to 31/12/2030.
- For *binary* or *text output*, the default value can be kept.

$ft [Data Format]

Acceptable values			
Binary	Text	HTML	Graph

- The Text format will output a comma-separated file. The separator is '**;**' to avoid any confusion with decimal point.

$tn [Tag Name]

NOTICE
The Tag **must** be specified for this export (Real Time Logging enabled).

$ts [Timestamp Format]

The **$dtRL** exports 2 time elements.

- **TimeInt**: integer format (number of seconds since 01/01/1970 - UNIX Epoch time).
- **TimeStr**: timestamp in string format (syntax depending on $ts modifier).

The **$ts** field impacts the format of the **TimeStr** element.

The difference between the **TimeInt** presentation and the **TimeStr** presentation will vary depending on whether the Ewon device is configured to "*Record data in UTC*".

Table 20. Historical Logging with “Record data in UTC” enabled:

Modifier	TimeInt	TimeStr	Example
No modifier	UTC Time	Local Time	"TimeInt";"TimeStr";"Value" 1537355066;"19/09/2018 13:04:26";506 1537355076;"19/09/2018 13:04:36";506
$tsO	UTC Time	UTC Time	"TimeInt";"TimeStr";"Value" 1537355066;"19/09/2018 11:04:26";506 1537355076;"19/09/2018 11:04:36";506
$tsU	UTC Time	UTC Time	"TimeInt";"TimeStr";"Value" 1537355067;"2018-09-19T11:04:27Z";506 1537355077;"2018-09-19T11:04:37Z";506
$tsL	UTC Time	Local Time	"TimeInt";"TimeStr";"Value" 1537355066;"2018-09-19T13:04:26+0200";506 1537355076;"2018-09 19T13:04:36+0200";506

Table 21. Historical Logging with “Record data in UTC” not enabled:

Modifier	TimeInt	TimeStr	Example
No modifier	Local Time	Local Time	"TimeInt";"TimeStr";"Value" 1537355066;"19/09/2018 13:04:26";506 1537355076;"19/09/2018 13:04:36";506
$tsO	Local Time	UTC Time	"TimeInt";"TimeStr";"Value" 1537355066;"19/09/2018 11:04:26";506 1537355076;"19/09/2018 11:04:36";506
$tsU	Local Time	UTC Time	"TimeInt";"TimeStr";"Value" 1537355067;"2018-09-19T11:04:27Z";506 1537355077;"2018-09-19T11:04:37Z";506
$tsL	Local Time	Local Time	"TimeInt";"TimeStr";"Value" 1537355066;"2018-09-19T13:04:26+0200";506 1537355076;"2018-09-19T13:04:36+0200";506

3.14. $dtSC [Export COM Config]

3.14.1. Export Content

$dtSC exports the **Communications Configuration** file (*comcfg.txt*).

The output format can be **Text** or **HTML**.

3.14.2. Detailed Example

```
sendmail "user@user.be","","eWON COM config
file","&[$dtSC$ftH$fncomcfg.htm]"
```

$dtSC	Data Type COM Config file
$ftH	Will export the file in HTML format
$fn	Will give to the file the required name

It will attach the Ewon *comcfg.htm* file to the email.

3.14.3. Used Fields

Fields	Value if not specified	Mandatory	Optional
$dt		X	
$ft	Text		X
$fn	Export Block Descriptor		X

3.15. $dtSE [Script Expression]

3.15.1. Export Content

This export provides a means to get the content of a **Script Expression**.

The Script Expression is a standard Ewon Basic-like expression returning a **String**, an **Integer** or a **Float**.

The evaluation of the expression will always occur between 2 script executions, for example between 2 OnTimer executions, or between 2 Cycles of the cyclic sections.

3.15.2. Detailed Example

```
$dtSE $se"A$"
```

$dtSE	Data Type Script Expressions

3.15.3. Used Fields

Fields	Value if not specified	Mandatory	Optional
$dt		X	
$se		X	
$ft	Binary		X
$fn	Export Block Descriptor		X

3.15.4. Special Parameters and Fields

$ft [Data Format]

Acceptable values		
Text	HTML	Binary

- Binary and Text format means that the output is the content of the Script Expression itself.
- HTML output supposes that the content of the script expression is a comma-separated data (string between quotes, items separated by '**;**' and end of lines marked with CRLF (0x0d, 0x0a)). The exported output is an HTML table containing these data.
- **$se** defines the script expression to output. Usually this expression is typed between quotes because $ characters are considered as separator otherwise.

3.16. $dtSS [Scheduled Status]

3.16.1. Export Content

The **Scheduled Status** are actions that are executed in a scheduled manner, for example: PutFTP, Send Mail, Send SMS.

When one of these actions is requested, it does not occur immediately, but it is queued for a sequential execution.

This export allows checking the content of this queue and giving the **status of all the actions in queue**: "*in progress*", "*executed (success)*" and "*executed with error*".

3.16.2. Detailed Example

```
$dtSS
```

$dtSS	Data Type Scheduled Status

3.16.3. Used Fields

Fields	Value if not specified	Mandatory	Optional
$dt		X	
$ft	Text		X
$fn	Export Block Descriptor		X
$ts	No modifier		X
$ct	No Compression		X

3.16.4. Special Parameters and Fields

$ft[Data Format]

Acceptable values		
Text	HTML	

- The Text format will output a comma-separated file. The separator is ’**;**’ to avoid any confusion with decimal points.
- Line content of output file: **"ActionId","ActionType","StatusCode","StatusText","Start","End"**

$ts [Timestamp Format]

This result of this EBD contains 2 time elements:

- **Start:** Timestamp of the start time in string format
- **End:** Timestamp of the end time in string format

The time format field applies to both elements.

Modifier	Start Time	End Time	Example
No modifier	Local Time	Local Time	41;"Send Mail";0;"Success";"19/09/2018 11:25:01";"19/09/2018 11:25:38" 42;"Send Mail";0;"Success";"19/09/2018 11:25:10";"19/09/2018 11:26:17"
$tsO	UTC Time	UTC Time	41;"Send Mail";0;"Success";"19/09/2018 11:25:01";"19/09/2018 11:25:38" 42;"Send Mail";0;"Success";"19/09/2018 11:25:10";"19/09/2018 11:26:17"
$tsU	UTC Time	UTC Time	41;"Send Mail";0;"Success";"2018-09-19T11:25:01Z";"2018-09-19T11:25:38Z" 42;"Send Mail";0;"Success";"2018-09-19T11:25:10Z";"2018-09-19T11:26:17Z"
$tsL	Local Time	Local Time	41;"Send Mail";0;"Success";"2018-09-19T13:25:01+0200";"2018-09-19T13:25:38+0200" 42;"Send Mail";0;"Success";"2018-09-19T13:25:10+0200";"2018-09-19T13:26:17+0200"

3.17. $dtSV [System Variable]

3.17.1. Export Content

$dtSV returns the value of a defined Ewon system variable.

A typical use is when the user wants to **include the Ewon online IP address in an email** by using the *sendmail Basic syntax*.

The output format can only be of **Text type**.

3.17.2. Detailed Example

```
sendmail "user@user.be","","Ip","The Ewon online IP'address is:
[$dtSV$seOnlineIpAddr]"
```

$dtSV	Data Type System Variable
$se	Will export a System Expression
OnlineIpAddr	The current Ewon online IP address (ie. 192.168.10.15)

It will include the Ewon online IP address in the body from a sent email.

3.17.3. Used Fields

Fields	Value if not specified	Mandatory	Optional
$dt		x	
$se		X	

3.18. $dtTR [TAR file]

3.18.1. Export Content

$dtTR exports the Ewon file(s) inside a **TAR** formated file.

The data to include in the TAR file can be defined using a single file list, a directory and wildcard '*****', or/and another export block descriptor.

3.18.2. Detailed Example

```
$dtTR $fnmytar.tar $td{/usr/*}
```

$dtTR	Data Type TAR file
$td	Data {/usr/*} the complete /usr directory
$fn	mytar.tar

Put the complete **/usr** directory in the *mytar.tar* file.

3.18.3. Used Fields

Fields	Value if not specified	Mandatory	Optional
$dt		X	
$fn	Text	X	
$td	Data	X	
$ft	B		X
$ct	No Compression		X

3.18.4. Special Parameters and Fields

$fn [Destination Filename]

The **$fn** is used to define a Name for the Output File.

Example:

$fnMyDataFile.tar will produce a TAR file with the name "*MyDataFile.tar*".

$td [TAR Data]

The data consists in a list of items separated by ',' (comma).

The items are specified between "{ }" (curly brackets).

```
$td {item1},{item2},...,{itemX}
```

Each item is one of the following:

- A /usr file name (complete path to file)
- A /usr directory name (complete path to directory) followed by *
- An export block descriptor

If the path represents a directory followed by * then the whole tree is exported.

TAR Format & eTAR Modified Format:

The TAR file produced by the Ewon could be:

- a standard TAR file, compliant to the USTAR (Uniform Standard Tape Archive) format.
- a modified TAR file, called eTAR.

Standard TAR file can be opened by most of Packager Program like Winzip, WinRar, ...

Due to technical reasons, the Ewon produces an eTar format when the package holds file(s) belonging to the Ewon root directory. This eTAR file is viewed as a "corrupted file" by Packager Program. But, you can use our eTar.exe tools to reformat this eTAR as a valid TAR file.

TIP
You can find this **eTar.exe** program on http://cdn.ewon.biz/software/divers/etar.zip.

Examples:

```
$dtTR $fnmytar.tar $td{/usr/file1.txt}
```

Will make a TAR file named "mytar.tar" containing the file /usr/file1.txt

```
$dtTR $fnmytar.tar $td{/usr/MyFile1.txt},{/usr/MyFile2.txt}
```

Will make a TAR file named "mytar.tar" containing the files /usr/MyFile1.txt and /usr/MyFile2.txt

```
$dtTR $fnmytar.tar $td{$dtCF $ftT $fnMyConfig.txt}
```

Will make an eTAR file named "mytar.tar" containing the Ewon configuration file named "MyConfig.txt"

```
$dtTR $fnmytar.tar $td{/usr/file1.txt},{$dtCF $ftT $fnMyConfig.txt}
```

Will make an eTAR file named "mytar.tar" containing the Ewon configuration file named "MyConfig.txt" and the file /usr/file1.txt

```
$dtTR $fnmytar.tar $td{/usr/*}
```

Will make a TAR file named "mytar.tar" containing all the /usr directory.

```
$dtTR $fnmytar.tar.gz $ctG $td{/usr/*}
```

Will make a compressed TAR file named "mytar.tar.gz" containing all the /usr directory

```
$dtTR $fnmytar.tar $td{/usr/*},{$dtPG $fnprogram.bas},{$dtCF $ftT
$fnconfig.txt},{$dtSC $ftT $fncomcfg.txt}
```

Will make an eTAR file named "mytar.tar" containing all the /usr directory, the program file named "program.bas", the configuration file named "config.txt" and the communication configuration file named "comcfg.txt"

```
putftp "Test_TAR.tar","[$dtTR $td{/usr/Page1.shtm},{/usr/Page2.shtm}]"
```

Will put by FTP the file "Test_TAR.tar" containing the files Page1.shtm and Page2.shtm

NOTE
For FTP action, the filename is the first parameter of the **PutFTP** instruction, then the **$fn** parameter is not required in the TAR command.

NOTICE
It is forbidden to include an item that describes a TAR format itself.

The TAR is not recursive.

Forbidden example:

```
$dtTR $td{ $TR ……….. }
```

3.19. $dtUF [User File]

3.19.1. Export Content

The **User File** export returns the content of a file in the User File area (**/usr/ directory** – or sub-directory).

When the file is exported, SSI tags such as **<%#ParamSSI>** and **<#TagSSI>** included in the user file are replaced by the actual values.

3.19.2. Detailed Example

```
$dtUF $uf/ufdir/uf1.txt
```

$dtUF	Data Type User File
$uf/ufdir/uf1.txt	Will export the uf1.txt file located in the /usr/ufdir directory

3.19.3. Used Fields

Fields	Value if not specified	Mandatory	Optional
$dt		X	
$uf		X	
$ft	Binary		X
$fl	SSI are parsed		X
$fn	Export Block Descriptor		X
$ct	No Compression		X

3.19.4. Special Parameters and Fields

$ftB [Data Format]

File type binary (default). Other types are unavailable.

$flNOSSI [Group Filter No SSI]

The **$flNOSSI** can be used to disable SSI parsing in **$dtUF**

When the **$dtUF** export block descriptor is used to export a user file, then Ewon will parse the user file during export for any SSI tag (tags starting with <%#). In some cases, this behavior is not wanted (in case the file may contain the <%# sequence, but no SSI are used).

IMPORTANT
NOSSI must be entered in caps (case sensitive)

Example:

```
$dtUF $uf/usr/MyFile.bin $flNOSSI $fnOutFile.bin
```

$uf [User File Name]

This field is the name of the user file that you want to export (source name).

The file name can be preceded by the name of the subdirectory inside the /usr directory:

```
/myfile.txt (myfile.txt is in the /usr directory)
```

```
/mydir/myfile.txt (myfile.txt is in the /usr/mydir directory)
```

The complete path can also be specified:

```
/usr/myfile.txt (myfile.txt is in the /usr directory)
```

```
/usr/mydir/myfile.txt (myfile.txt is in the /usr/mydir subdirectory)
```

NOTE
The first "**/**" is optional.

Example:

```
Putftp "/test.txt", " [$dtUF $uf/myfile.txt] "
```

$fn [Destination File Name]

This field is used for specifying a file name to the export data (destination name). Usually this file name is used to specify the output of the data, for example when sending an attachment to an email.

In this case, the **$fn** file name gives the name of the attachment:

```
SENDMAIL "MailReceiver@YourMail.com", "", "Mail Subject", "& [$dtUF $uf/
myfile.txt $fnNewName.txt]"
```

The above example will attach to an email a file named “**NewName.txt**” that is a copy of the file “**/usr/myfile.txt**”.

There is also one special use of the **$fn**: when a user file (**$dtUF$fn**) is exported and you do not specify the source name (**$uf**); in that case, the **$fn** parameter is used as source and as destination file name.

Using only **$fn** in a send mail string:

```
SENDMAIL "MailReceiver@YourMail.com", "", "Mail Subject", "Mail text &
[$dtUF$fnmyfile.txt]"
```

The above syntax will attach a file with its name (and not with the EBD syntax as name). When doing a PUTFTP, then **$fn** does not need to be specified, because the PUTFTP command manages the name of the destination file:

```
PUTFTP "MyFileWithANewName.txt", "[$dtUF $uf/myfile.txt]"
```

3.20. Additional Exports Available

$dtTL	Tag List
$dtPG	Program
$dtCF	Configuration File

These are all the files from the Ewon configuration.

They are equivalent to the file available through the *Ewon FTP* server.