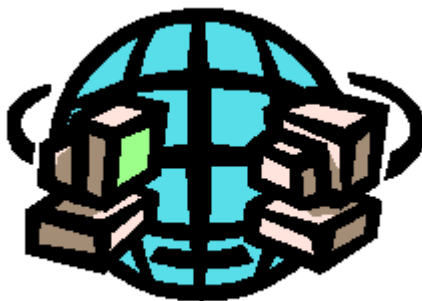


eWON 500-2001-4001-4002

User Guide

Version 4.3



COOL INTERNET TELECONTROL SOLUTION

1 Introduction.....	14
2 The eWON welcome page	15
3 eWON Login.....	16
4 eWON Configuration Interface.....	17
4.1 Overview	17
4.2 Users Setup	18
4.2.1 To edit a user	18
4.2.2 To create a new user.....	18
4.2.3 Logon parameters	19
4.2.4 User's rights	19
4.2.5 User's directory restriction	20
4.2.6 Tag pages restriction	21
4.2.7 To delete a user	21
4.2.8 Callback settings.....	21
4.2.9 Validation of the User's settings.....	21
4.3 System Setup	22
4.3.1 Main system setup	23
4.3.1.1 eWON identification (and miscellaneous)	23
4.3.1.2 Setup for outgoing actions.....	24
4.3.1.3 Resetting eWON.....	26
4.3.1.3.1 Reset Request from the General Configuration page.....	26
4.3.2 SNMP Setup.....	27
4.3.2.1 Communities	28
4.3.2.2 Hosts.....	28
4.3.3 Virtual Com Port Setup	29
4.3.3.1 Introduction.....	29
4.3.3.2 The different VCOM modes	29
4.3.3.2.1 RAW TCP.....	29
4.3.3.2.2 TELNET RFC2217	29
4.3.3.2.3 MODBUS/TCP MODBUS/RTU Gateway.....	30
4.3.3.3 PC Virtual COM Port driver.....	30
4.3.3.3.1 MbsTepCOM	30
4.3.3.3.2 Serial/IP	30
4.3.3.4 eWON VCOM Configuration.....	30
4.3.3.4.1 Introduction.....	30
4.3.3.4.2 Web configuration	31
4.3.4 Diagnostic	33
4.3.4.1 Overview	33
4.3.4.2 Events logging	33

4.3.4.2.1 Overview	33
4.3.4.2.2 Event logging Configuration page.....	34
4.3.4.3 PPP Dump	35
4.3.4.3.1 Overview	35
4.3.4.3.2 PPP Dump Configuration	35
4.3.5 Clock Setup section	36
4.3.6 COM Setup section.....	36
4.3.6.1 Ethernet	37
4.3.6.2 Modem	38
4.3.6.2.1 Modem configuration (PSTN modem).....	38
4.3.6.2.2 Modem configuration (GSM/GPRS modem)	39
4.3.6.2.3 Leased line configuration	42
4.3.6.3 Dial up (PPP).....	43
4.3.6.4 Callback.....	46
4.3.6.5 Router	49
4.3.6.6 IP Services	51
4.3.6.7 Default Config	52
4.3.7 STORAGE CONFIGURATION	52
4.3.7.1 Overview	52
4.3.8 The various types of information stored	52
4.3.8.1 Historical data.....	52
4.3.8.1.1 Overview	52
4.3.8.1.2 Partitioning configuration.....	52
4.3.8.2 Retentive values	52
4.3.8.2.1 Overview	52
4.3.8.2.2 Partitioning configuration.....	52
4.3.8.3 Communication Configuration.....	53
4.3.8.3.1 Overview	53
4.3.8.3.2 Partitioning configuration.....	53
4.3.8.4 /usr partition.....	53
4.3.8.4.1 Overview	53
4.3.8.4.2 Partitioning configuration.....	53
4.3.8.5 /sys partition	53
4.3.8.5.1 Overview	53
4.3.8.5.2 Partitioning configuration.....	53
4.3.8.6 Storage size options.....	53
4.3.8.7 Storage management from the eWON Web interface.....	54
4.3.8.7.1 Configure	54
4.3.8.7.2 Erase & Format	55
4.3.8.8 RESET Request	56
4.4 Tag Setup.....	57
4.4.1 Tag definition: Introduction.....	57

4.4.1.1 Tags monitoring and handling	58
4.4.2 Tag definition: Setup.....	58
4.4.2.1 Tag main edit window	60
4.4.2.1.1 Publish as Modbus TCP	64
4.4.2.1.1.1 Defining the Modbus TCP address	64
4.4.2.1.1.2 Modbus TCP rules.....	65
4.4.2.2 Tag "Alarm Action" edit window	65
4.4.2.2.1 Email on alarm configuration.....	66
4.4.2.2.2 SMS on alarm configuration.....	66
4.4.2.2.2.1 SMS_RECIPIENT: syntax	66
4.4.2.2.3 FTP on alarm configuration	69
4.4.2.2.3.1 File content:.....	69
4.4.2.2.4 SNMP Trap on alarm configuration.....	69
4.5 Pages configuration	69
4.6 IO servers setup	70
4.7 Script Setup	70
4.7.1 The Edit script link	71
4.7.2 The Script control link.....	73
4.7.3 The RUN/STOP link	73
5 Configuring the eWON by a file upload	74
6 The eWON IO Servers	75
6.1 Introduction.....	75
6.2 IO servers setup	75
6.2.1 Standard IO server configuration page	76
6.3 Modbus IO server	77
6.3.1 Introduction	77
6.3.2 Setup	77
6.3.2.1 Setup for eWON Server	77
6.3.2.2 Setup for eWON IO server and Gateway - COM Setup.....	78
6.3.2.3 Topic configuration	78
6.3.2.4 Advanced parameters.....	79
6.3.2.4.1 Additional advanced parameters	80
6.3.3 Tag name convention	81
6.3.3.1 ValueName	81
6.3.3.2 Slave Address	82
6.3.3.3 IP Address	82
6.3.3.4 Device specific information.....	83
6.4 NETMPI IO Server	84
6.4.1 Introduction	84
6.4.2 Setup	84

6.4.3 Tag name convention	85
6.4.3.1 ValueName	85
6.4.3.2 Device Address	85
6.5 UNITE IO Server.....	86
6.5.1 Introduction	86
6.5.2 Setup	86
6.5.2.1 Communication Setup	86
6.5.2.2 Topic configuration	88
6.5.3 Tag name convention	89
6.5.3.1 Value Name	89
6.5.3.2 The device address syntax.....	90
6.6 DF1 IO Server	91
6.6.1 Introduction	91
6.6.2 Setup	91
6.6.2.1 Communication Setup	91
6.6.2.2 Topic configuration	92
6.6.3 Tag name convention	93
6.6.3.1 Value Name	94
6.6.3.1.1 General Description.....	94
6.6.3.1.2 Output File Items	94
6.6.3.1.3 Input File Items	94
6.6.3.1.4 Status File Items.....	94
6.6.3.1.5 Binary File Items.....	95
6.6.3.1.6 Timer File Items.....	95
6.6.3.1.7 Counter File Items	95
6.6.3.1.8 Control File Items	95
6.6.3.1.9 Integer File Items	95
6.6.3.1.10 Floating File Items	95
6.6.3.1.11 ASCII File Items	95
6.6.3.2 Destination Device Type and Address	95
6.7 FINS IO Server	96
6.7.1 Introduction	96
6.7.2 Setup	96
6.7.2.1 Communication Setup	96
6.7.2.2 Topic Configuration	97
6.7.2.3 Gateway Configuration	98
6.7.3 Tag Name Convention	99
6.7.3.1 Value Name	99
6.7.3.1.1 General Description.....	99
6.7.3.2 Global Device Address	99
6.8 S5-AS511 IO Server.....	100

6.8.1 Introduction	100
6.8.2 Setup	100
6.8.3 Communication setup	101
6.8.3.1 Supported Devices	101
6.8.4 Tag name convention	101
6.8.4.1 ValueName	101
6.9 EWON IO Server.....	103
6.9.1 Introduction	103
6.9.2 Standard eWON I/O Item Names.....	103
6.9.2.1 Tag name convention.....	103
6.9.2.2 Analog Input Value Range (eWON4002/eWON1002)	105
6.9.2.2.1 Configurable analog input AI1 to AI4	105
6.9.2.2.2 PT100 input AI5 and AI6.....	107
6.9.3 Setup	108
6.9.3.1 Configuration of the counter pulse length.....	109
6.9.3.2 Energy management setup.....	109
6.10 MEM IO Server	110
6.10.1 Introduction	110
6.10.2 Setup	110
6.10.3 Tag name convention	110
7 eWON Monitoring Web Interface.....	111
7.1 Real-time screen.....	112
7.1.1 Change Tag value	112
7.1.2 Alarm state.....	112
7.1.3 Real time graph	113
7.1.4 Historical window.....	113
7.2 Historical Trending screen	113
7.3 Real-time Alarm screen.....	115
7.4 Historical Alarm screen	117
7.5 Files transfer.....	118
8 Retrieving Data from eWON	119
8.1 List of eWON files.....	119
8.2 Files Format	120
8.3 FTP transfer	120
8.3.1 FTP Software tools	120
8.3.2 FTP session.....	121
8.3.3 Via eWON web site	121
9 Programming the eWON	122

9.1 BASIC language definition	122
9.1.1 Introduction	122
9.1.2 Program flow	122
9.1.3 Character string	126
9.1.4 Command.....	126
9.1.5 Integer	127
9.1.6 Real	127
9.1.7 Alphanumeric character.....	127
9.1.8 Function.....	127
9.1.9 Operators priority	127
9.1.10 Type of Variables	128
9.1.10.1 Integer variable.....	128
9.1.10.2 Real variable.....	128
9.1.10.3 Alphanumeric string.....	128
9.1.10.4 Characters arrays	128
9.1.10.5 Real arrays	129
9.1.11 TagName variable	129
9.1.12 Limitations of the BASIC	129
9.2 List of the keywords.....	130
9.2.1 Syntax convention	130
9.2.2 # (bit extraction operator)	130
9.2.3 ABS	130
9.2.4 ALMACK.....	130
9.2.5 ALSTAT	131
9.2.6 AND	131
9.2.7 ASCII.....	132
9.2.8 BINS	132
9.2.9 BNOT.....	132
9.2.10 CFGSAVE.....	133
9.2.11 CHR\$	133
9.2.12 CLEAR.....	133
9.2.13 CLOSE	133
9.2.14 CLS	134
9.2.15 DAY	134
9.2.16 DEC	134
9.2.17 DIM.....	134
9.2.18 DOW	135
9.2.19 DOY	135

9.2.20 DYNDNS	136
9.2.21 END	136
9.2.22 ERASE	136
9.2.23 EOF	137
9.2.24 FCNV	137
9.2.25 FOR NEXT STEP	138
9.2.26 GET	138
9.2.26.1 /usr Syntax [function] – Binary mode	138
9.2.26.2 /usr Syntax [function] – Text mode	139
9.2.26.3 COM Syntax [function] – Binary mode	139
9.2.26.4 TCP/UDP Syntax [function] – Binary mode	140
9.2.27 GETFTP	140
9.2.28 GETIO	140
9.2.29 GETSYS, SETSYS	141
9.2.30 GO	144
9.2.31 GOSUB RETURN	144
9.2.32 GOTO	144
9.2.33 HALT	145
9.2.34 HEX\$	145
9.2.35 IF THEN ELSE ENDIF	145
9.2.35.1 Short IF Syntax	145
9.2.35.2 Long IF syntax	145
9.2.36 INSTR	146
9.2.37 INT	146
9.2.38 IOMOD	146
9.2.39 IORCV	147
9.2.40 IOSEND	148
9.2.41 LEN	148
9.2.42 LOGEVENT	149
9.2.43 LOGIO	149
9.2.44 LTRIM	149
9.2.45 MOD	150
9.2.46 MONTH	150
9.2.47 NOT	150
9.2.48 NTPSync	150
9.2.49 ONxxxxxx	151
9.2.50 ONALARM	151
9.2.51 ONCHANGE	152

9.2.52 ONERROR	152
9.2.53 ONPPP.....	152
9.2.54 ONSMS.....	153
9.2.55 ONSTATUS	153
9.2.56 ONTIMER	154
9.2.57 OPEN.....	154
9.2.57.1 Introduction to file management.....	154
9.2.57.2 OPEN general syntax	154
9.2.57.3 Different File/stream types.....	155
9.2.57.3.1 FILE open /usr	155
9.2.57.3.2 TCP stream open Syntax [command]	155
9.2.57.3.3 COM port open Syntax [command].....	156
9.2.57.3.4 EXP export bloc descriptor open Syntax [command]	157
9.2.58 OR.....	158
9.2.59 PI.....	158
9.2.60 PRINT - AT.....	159
9.2.61 PRINT #	159
9.2.62 PUT	160
9.2.62.1 File Syntax[Command] – Binary mode	160
9.2.62.2 File Syntax[Command] – Text mode	160
9.2.62.3 COM Syntax[Command] – Binary mode	161
9.2.62.4 TCP/UDP Syntax[Command] – Binary mode	161
9.2.63 PUTFTP	161
9.2.64 REBOOT	161
9.2.65 REM.....	162
9.2.66 RTRIM	162
9.2.67 SENDMAIL	162
9.2.68 SENDSMS	163
9.2.69 SENDTRAP	164
9.2.70 SETIO.....	165
9.2.71 SETTIME.....	165
9.2.72 SGN.....	165
9.2.73 SQRT	166
9.2.74 STR\$	166
9.2.75 TIMES\$	166
9.2.76 TGET.....	167
9.2.77 TSET.....	167
9.2.78 VAL	168
9.2.79 XOR.....	168

9.3 Debug a BASIC program	169
9.4 BASIC Errors Codes	169
9.5 Configuration Fields	170
9.5.1 SYS Config	171
9.5.2 Com Section	172
9.5.3 Tag Section	175
9.5.3.1 Send on alarm notification patterns*	177
9.5.3.2 Setting a Tag value, deleting a Tag and acknowledging an alarm	177
9.5.4 User Section	178
10 User defined Web Site	179
10.1 Introduction	179
10.2 SSI Syntax	179
10.2.1 HTML Page extension	179
10.2.2 Special eWON SSI Tags	179
10.2.2.1 TagSSI HTML Tag	180
10.2.2.2 ParamSSI HTML Tag	181
10.2.3 VarSSI HTML Tag	182
10.2.4 ExeSSI HTML Tag	182
10.3 bASP Syntax	182
10.3.1 Building page content by using bASP	182
10.3.2 WEB context variables	183
10.3.2.1 Web context variables for request parameters	183
10.3.2.2 Web context variable directly inserted with SSI	184
10.3.2.3 Improving WEB performance using Web context variables and bASP	185
10.3.2.4 Using Web context variables with FORMS	186
10.4 Special FORMS	187
10.4.1 Update Tag Value (and acknowledge)	187
10.4.1.1 Examples	188
10.4.1.2 Acknowledge Tags alarms	189
10.4.2 Produce an export data block	189
10.4.3 Execute an eWON script	190
10.4.3.1 Syntax	190
10.4.3.2 Examples	191
11 Export Block Descriptor	192
11.1 Export block descriptor	192
11.2 Export fields syntax definition	193
11.2.1 \$dt [Data Type]	193
11.2.2 \$ft [Format]	194

11.2.3 \$st [Start Time] and \$et [End Time]	195
11.2.3.1 \$st, \$et with relative time	195
11.2.3.2 \$st, \$et with absolute time	195
11.2.3.3 \$st , \$et with Last time.....	196
11.2.4 \$ut [Update Time]	196
11.2.5 \$tn [Tag Name]	197
11.2.6 \$ct [compression format]	197
11.2.7 \$se [Script Expression]	197
11.3 Data Types description and syntax	197
11.3.1 \$dtHL [Historical Logging]	198
11.3.1.1 Export content.....	198
11.3.1.2 Detailed Example	198
11.3.1.3 Fields used	198
11.3.1.4 Special parameters and fields	199
11.3.2 \$dtRL [Real time Logging]	200
11.3.2.1 Export content.....	200
11.3.2.2 Detailed Example	200
11.3.2.3 Fields used	201
11.3.2.4 Special parameters and fields	201
11.3.3 \$dtAH [Alarm History]	202
11.3.3.1 Export content.....	202
11.3.3.2 Detailed Example	202
11.3.3.3 Fields used	202
11.3.3.4 Special parameters and fields	203
11.3.4 \$dtAR [Alarm Real time]	204
11.3.4.1 Export content.....	204
11.3.4.2 Detailed Example	204
11.3.4.3 Fields used	204
11.3.4.4 Special parameters and fields	204
11.3.5 \$dtEV [Event file]	205
11.3.5.1 Export content.....	205
11.3.5.2 Detailed Example	205
11.3.5.3 Fields used	205
11.3.5.4 Special parameters and fields	206
11.3.6 \$dtSS [Scheduled Status]	207
11.3.6.1 Export content.....	207
11.3.6.2 Detailed Example	207
11.3.6.3 Fields used	207
11.3.6.4 Special parameters and fields	207

11.3.7 \$dtSE [Script Expression]	208
11.3.7.1 Export content.....	208
11.3.7.2 Detailed Example	208
11.3.7.3 Fields used	208
11.3.7.4 Special parameters and fields	208
11.3.8 \$dtUF [User File]	209
11.3.8.1 Export content.....	209
11.3.8.2 Detailed Example	209
11.3.8.3 Fields used	209
11.3.8.3.1 \$fn [File Name]	210
11.3.8.3.2 \$uf [User File]	210
11.3.8.3.3 Special parameters and fields	211
11.3.9 \$dtIV [Instant Values]	212
11.3.10 Instant value - general information	212
11.3.10.1 Alarm status code values.....	212
11.3.10.2 Alarm type values	213
11.3.10.3 Writing Instant Values to eWON	213
11.3.10.4 Binary file format	213
11.3.10.5 Export content.....	214
11.3.10.5.1 Detailed examples	214
11.3.10.5.2 Fields used	214
11.3.10.6 \$fl [Group or Groups]	214
11.3.11 \$dtSV	215
11.3.11.1 Export content.....	215
11.3.11.2 Detailed Example	215
11.3.11.3 Fields used	215
11.3.12 \$dtPP	216
11.3.12.1 Export content.....	216
11.3.12.2 Detailed Example	216
11.3.12.3 Fields used	216
11.3.13 \$dtES	217
11.3.13.1 Export content.....	217
11.3.13.2 Detailed Example	217
11.3.13.3 Fields used	217
11.3.14 \$dtSC	218
11.3.14.1 Export content.....	218
11.3.14.2 Detailed Example	218
11.3.14.3 Fields used	218
11.3.15 Additional exports available	218
12 Upgrading the eWON firmware	219

12.1 Purpose	219
12.2 Upgrading the eWON firmware with eBuddy	219
12.3 Upgrading the eWON firmware by a direct upload.....	222
13 Appendix.....	225
13.1 Access to the eWON Technical Support.....	225
13.2 eWON configuration and files storage.....	225
13.2.1 Flash file system.....	225
13.2.2 Non-volatile configuration	225
13.3 Tips for Internet setup.....	226
13.3.1 Finding the IP address of a given host.....	226
13.4 Finding your PC IP address.....	226
13.5 Resetting the eWON	227
13.5.1 Overview	227
13.5.2 Reset sequence	227
13.5.2.1 First level reset	227
13.5.2.2 Second level reset	227
13.5.3 Second level initialization diagnostics	228
13.5.4 Entering level 2 initialization without request.....	228
13.5.5 What to do in case of error?	228
13.5.6 Important remark	228
13.6 The eWON firmware upgrade process	229
13.6.1 Overview of the upgrade process	229
13.6.2 eWON firmware upgrade step-by-step	231
13.6.2.1 Upgrading from version 3 to version 4	231
13.6.2.2 Upgrading from version 4 to version 4	231
13.6.2.3 Monitoring of eWON's behavior during the flashing operation	232
13.7 Table of comparisons between eWON types	233

1 Introduction

The aim of this guide is to provide you with exhaustive information about the multiple eWON Version 4 features.

Providing the gains in terms of optimization and performances that firmware version 4 brings, we incite you to migrate from version 3 to version 4 if not already done.

Among the sum of new items that are embedded in version 4, – the ability to customize the way you store your configuration and user files, the ability to receive SMS, the ability to monitor potential failures and dysfunctions by means of the brand new diagnosis functions – may decide you to take the decision to upgrade from v3 to v4.

The upgrade process is described very precisely at the end of this manual, at chapter “The eWON firmware upgrade process” on page 229.

This guide describes the version 4 features for all eWON types. You will then find from place to place in this guide some references to a matrix table you can find at this end of this guide, in which you can quickly check whether the described feature concerns your eWON type or not.

Please refer to chapter “Table of comparisons between eWON types” on page 233.

Clicking on the link inside of the table that matches the feature you was consulting, will bring you back to the good chapter.

i.e. If you click on the link “Historical data logging” at the beginning from chapter “Tag main edit window”, you will then be directed to the table matrix, in which you will check that Historical data logging feature is possible with eWON 4001 and 4002 models only.

If you click again on the matching link inside of the table, you will be redirected back to the “Tag main edit window” chapter.

Users who prefer to consult the printed manual should keep a printed copy of the eWON's comparison table apart from the entire guide to quickly check the different features that are available or not, depending on the type of eWON they are working with.

Now let's discover all that is possible for you to realize, by using your eWON V4 on a daily basis.

2 The eWON welcome page



Figure 1: eWON welcome page

The eWON welcome page will invite you to log on. The Name just below the eWON logo is the name of your eWON, which allows you to identify the eWON you are connected to (see next chapter). A link to the ACT'L site is also provided if you want to know more about the company or download eWON application notes or software updates. Clicking on the PDA icon at the top left corner of the page allows you to reach the web pages that have been designed to be displayed on a PDA screen.



Figure 2: eWON PDA welcome page

3 eWON Login

The default factory defined login of the eWON is:

Login	adm
Password	adm

Table 1: eWON default login and password

IMPORTANT! Password IS case sensitive but user name is not.

It is recommended to change the password of the "adm" user to protect it against intrusion. You will be able to change the user configuration in the Users setup screen of your eWON (please refer to chapter "Users Setup" on page 18).

4 eWON Configuration Interface

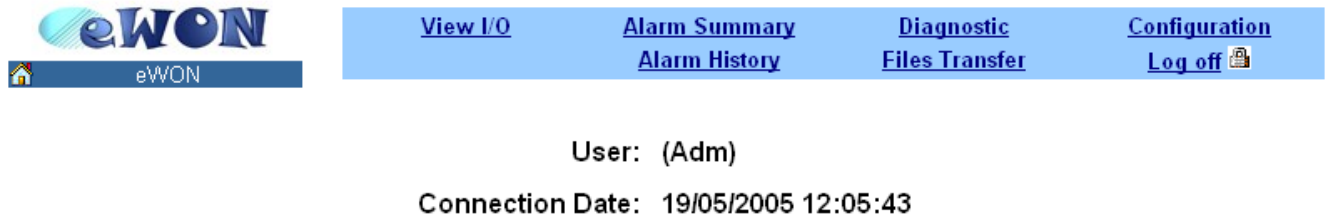


Figure 3: eWON SCADA page

Just after you have logged in successfully, the eWON SCADA will be displayed (as explained in the next chapter). But before analyzing the SCADA page, you need to configure your eWON. Click on the **Configuration** menu item. The following Configuration web page then appears:

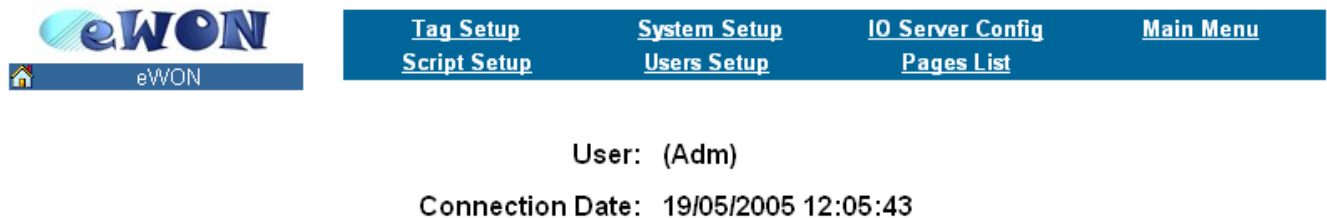


Figure 4: eWON configuration links

4.1 Overview

Configuration includes the following points:

- **User's list**

The user's list is used by the eWON to restrict access to the various features of the eWON according to the user's rights. It also allows logging the user's actions (like alarm acknowledge...).

See also chapter "Users Setup" on page 18

- **Tags list**

The eWON monitors and manages variables (called *Tags*). A Tag can be *Boolean* or *float*, and Tags are produced by IO servers. The configuration of a Tag defines its IO server and all its monitoring parameters (historical logging (Warning: not for all eWONs versions - see table page 233), alarm levels, etc.

See also chapter "Tag definition: Introduction" on page 56

- **System configuration**

Communication and global configuration of the eWON are defined here.

See also chapter "System Setup" on page 22

- **IO Server configuration**

As explained in chapter "Tag definition: Introduction" on page 56, each Tag is produced by an IO server. An IO server can interface the physical eWON IO or the Modbus remote IO, etc. Some of these IO servers require some configuration.

See also chapter "IO servers setup" on page 69

- **Pages definitions**

It is possible to organize the Tags by groups. These groups are called "pages" and they help keeping a clearer organization of the Tags and also managing per user rights for the Tags. Two pages, called "Default" and "System" are automatically defined, and up to 10 pages can be user defined.

See also chapter "Pages configuration" on page 68

- **Scripts Configuration**

The Basic language that is embedded in the eWON allows you to create your own scripts that you will use to get access to the data that are stored in the eWON, in order to perform real-time monitoring or programming actions on alarms,...

See also chapter "Script Setup" on page 69

4.2 Users Setup

The **Users Setup** page allows building the list of authorized eWON users.

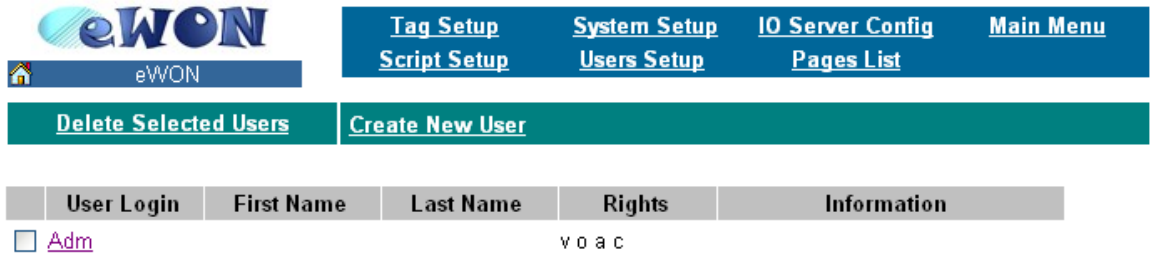


Figure 5: eWON Users Setup page

The user's page lists all eWON users and attributes. The first time you connect to the eWON, the default user is Adm. It is recommended to modify the default password of the Administrator user.

4.2.1 To edit a user

Click on the user's name link in the **User Login** column.

4.2.2 To create a new user

Click on the **Create New user** link.

In both cases, you will get the following edit window:

First Name:	<input type="text"/>	Last Name:	<input type="text"/>
User Login:	<input type="text"/>	Password:	<input type="password"/>
		Confirm Password:	<input type="password"/>
Information:	<input type="text"/>		
Rights			
All <input type="button" value="v"/>	Tag Page allowed (Default is always allowed)		
All <input type="button" value="v"/>	User Directory allowed (/usr/ is always allowed)		
<input checked="" type="checkbox"/>	View IO		
<input checked="" type="checkbox"/>	Force Outputs		
<input checked="" type="checkbox"/>	Acknowledge Alarms		
<input checked="" type="checkbox"/>	Change Configuration		
Callback			
Enabled	<input type="checkbox"/> (Callback must also be globally enabled in System Config)		
Callback phone number value is	Mandatory <input type="button" value="v"/>	(Defines if user can change phone number)	
Callback phone number	<input type="text"/>	(User's login and password are used to connect to server)	
<input type="button" value="Add/Update"/>		<input type="button" value="Cancel"/>	

Figure 6: eWON user's configuration page

4.2.3 Logon parameters

First Name and **Last Name** are detailed (and optional) information about the user, while **User Login** and **Password** are mandatory (they are used during the logon procedure).

Logon procedures using the **User Login** and **Password** are:

- Main eWON access logon
- FTP server access
- User defined page Digest access authentication (please refer to chapter “eWON identification (and miscellaneous)” on page 23)
- PPP Dial up connection

Warning: depending on eWON's version (c.f. table at the end of this manual page 233)

The same user name and password will be used for the different access.

4.2.4 User's rights

The following rights can be selected for user:

Rights	
All <input type="button" value="v"/>	Tag Page allowed (Default is always allowed)
All <input type="button" value="v"/>	User Directory allowed (/usr/ is always allowed)
<input checked="" type="checkbox"/>	View IO
<input checked="" type="checkbox"/>	Force Outputs
<input checked="" type="checkbox"/>	Acknowledge Alarms
<input checked="" type="checkbox"/>	Change Configuration

Figure 7: User Rights: user web & Tag pages

View IO	allows accessing the SCADA real time IO screen
Force outputs	allows changing the eWON outputs
Acknowledge alarms	allows acknowledging alarm
Change configuration	allows accessing the configuration part of the eWON

Table 2: user's rights explanations

There are two additional user's rights provided to restrict user rights access to:

- The directories in the user defined web site (please refer to chapter “User defined Web Site” on page 179)
- The pages of Tags

4.2.5 User's directory restriction

When the user web site is built, HTML (or SHTML) pages can be placed in subdirectories. The root directory of the user defined web site is **/usr** (from the FTP directory structure). Every user has access to that directory.

/usr is considered as DEFAULT directory for the web site.

Suppose for this explanation that the following FTP directory structure is defined:

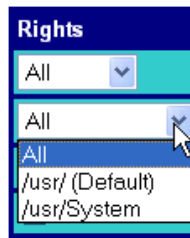
/usr/update contains HTML files to update Tags

/usr/admin contains pages for system administration

There are 10 user defined pages (please refer to chapter "Pages configuration" on page 68) and one default page. Suppose for the explanation that the following pages have been defined:

Page 1	update
Page 2	admin

When editing users rights, the following list would be proposed:



The following security schemes would be possible:

Dir. Right selected	Access description
All	HTML Pages in any directory will be accessible.
Default	Only the /usr directory pages will be accessible.
update	/usr and /usr/update directory pages will be accessible. Note: /usr is always accessible. When a subdirectory is accessible, all its subdirectory are also accessible. Example: /usr/update/image would also be accessible.
admin	/usr and /admin directory pages will be accessible. Note: this is obviously not useful because /usr/update will not be accessible which is probably not what is required for an "administrator". In the "administrator" case, the preferred selection is probably "All".

Table 3: user's security schemes - 1

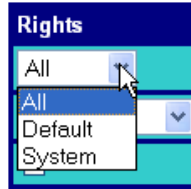
The proposed directories list is derived from the PAGES LIST. It is of the user's responsibility to create the same pages as the directories in the FTP structure (otherwise security setup will not be possible).

4.2.6 Tag pages restriction

There are 10 user defined pages (please refer to chapter "Pages configuration" on page 68) and one default page. The security mechanism follows the following rules:

- User has always access to the "Default" page
- User can have access to all pages
- User can have access to only 1 page

Example: taking the same pages as in previous example, the following selection appears:



The following security schemes would be possible:

Page Right selected	Access description
All	Access to all Tag pages is granted.
Default	Only the "Default" page is accessible.
update	"Default" and "update" pages would be accessible.
admin	"Default" and "admin" pages would be accessible.

Table 4: user's security schemes - 2

4.2.7 To delete a user

Click on the check box just next to the user's login of the user that you want to delete and click on the **Delete Selected Users** link.

- Note 1: you can select for deletion more than one user at the same time
- Note 2: the Adm user won't be deleted
- Note 3: password is CASE SENSITIVE

4.2.8 Callback settings

Warning: depending on eWON's version (c.f. Table on page 233)

When the callback is enabled as a global parameter in the **COM Setup** page, the user can initiate a User's callback and specify when the trigger will be started for the callback.

The callback phone number can be forced (from this menu), then the user can only initiate a call to a fixed phone number; or User Defined, then he can modify the proposed phone number. In the first case, the list box is set to **Mandatory**, and in the second case, it is set to **User Defined**.

The callback phone number can be specified here. Remember that the user's login and password that have been defined for the eWON access are used as login and password on the remote server called for callback.

Callback	
Enabled	<input type="checkbox"/> (Callback must also be globally enabled in System Config)
Callback phone number value is	Mandatory <input type="button" value="v"/> (Defines if user can change phone number)
Callback phone number	<input type="text"/> (User's login and password are used to connect to server)

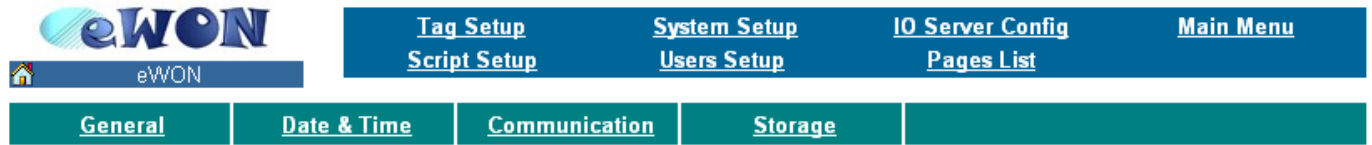
Figure 8: User's callback settings

4.2.9 Validation of the User's settings

When all the required fields are filled-in, you can either click on the **Add/Update** button (to validate your changes) or on the **Cancel** button (to undo your changes).

4.3 System Setup

The system setup page (accessed by clicking on the **System Setup** item from the configuration menu) allows setting all eWON system parameters and looks as follows:



Select a system page

Figure 9: eWON system configuration page

This section has a high impact on the eWON behavior (mainly on the communication point of view). You should fill it in carefully.

The System setup page is divided in four sections:

- **The Main system setup (General)**
 Defines all the eWON global settings, except for settings regarding the eWON communication.
 The user can modify the Email, FTP and NTP parameters, along with reset and format requests.
- **The clock setup section (Date & Time)**
 Used to update the system's Real Time Clock.
- **The Communication setup section**
 Includes all the communication settings of the eWON. These settings are separated from the Main settings and even stored at a different place inside the device in order to be able to format the eWON flash file system without affecting the communication settings (see also chapter "eWON configuration and files storage" on page 225).
 A menu allows the user to reset the communication parameters to their default values.
- **The storage setup section**
 Used to set up the way the memory resources from the eWON are used.

4.3.1 Main system setup

4.3.1.1 eWON identification (and miscellaneous)

eWON Identification:	<input type="text" value="eWON"/>
General Information: <small>(Transmitted in eMails)</small>	<input type="text"/>
User defined home page	<input type="text" value="http://your_device_ip/usr/"/> <input type="button" value="(clear for eWON default home page)"/>
Enable user pages security:	<input type="checkbox"/>

Figure 10: eWON main system setup - eWON identification

Control	Description
eWON Identification	The name of the eWON. This information is added inside all eWON Email alarm notification. Check that this identification and the next one allow you to identify eWON without any doubt.
General Information	As for eWON identification. You can put a free text; but it can be very useful to indicate here the eWON geographical location and phone number.
User defined home page	<p>When a user defined web site is used. The home page of the eWON web site can be replaced by a user defined web page. If your default home page is a viewON synopsis, you need to select "viewON synopsis" in the Combo box and enter the name of the synopsis in the Text box.</p> <p>If your default home page is a classical HTML page, you need to select "http://your_device_ip/usr" in the Combo box and enter the name of the page in the Text box.</p> <p>See also the chapter "User defined Web Site" on page 179 to get detailed information about the user web site creation.</p>
Enable user page security	If a user web site is defined, then default user logon page is not displayed and there is no session, but using Digest Access Authentication can also ensure security. This checkbox will enable the DAA security when the user wants to access a user-defined page. See also the chapter "User defined Web Site" on page 179 to get detailed information about the user web site creation.

Table 5: eWON Identification controls

4.3.1.2 Setup for outgoing actions

Regarding its Internet connectivity features, the eWON has basically two modes of operation:

- **A server mode (Web server and FTP server)**
- **A client mode (Email client, FTP client and NTP client)**

In the server mode, the eWON is waiting for a client to connect with its Web browser or with its FTP client.

In the client mode, the eWON needs to connect to a server. This connection requires knowing at least the IP address of the server and the Port number for the required service. Sometimes a username and a password are also required.

Note: Except in special case the Port number is usually the default value proposed by the eWON. This setup section is used to define the eWON's CLIENT MODE configuration.

eWON Identification:	eWON	
General Information: <small>(Transmitted in eMails)</small>		
User defined home page	http://your_device_ip/usr/	(clear for eWON default home page)
Enable user pages security:	<input type="checkbox"/>	

SETUP FOR ALARM ACTIONS		
Action retrig. interval:	86400 sec	Action occurs again if condition is still true
Try action	1 time(s)	Number of times the action is retried in case of error (>=1)
Action retry interval:	120 sec	Interval before retry in case of error (>=10)

SETUP FOR OUTGOING ACTIONS (operations for which eWON must connect to the Internet/Intranet)		
EMAIL Config	Configure Mail Transfert	
SMTP Server Address:	relay.proximus.be	Usually something like smtp.domain.com or mail.domain.com (can be an IP address)
SMTP Server Port:	25	The default value is 25. It must only be changed in very special cases.
E-Mail "From" User name :		this will be used to send eMails, it must be compatible with your account name on the SMTP server.
User name:		Fill this only if SMTP requires authentication, otherwise leave empty.
Password:		Password for SMTP authentication (only if above field used).
NTP Config	Configure update of eWON clock with an NTP time server	
Enable NTP clock update	<input type="checkbox"/>	
NTP Server Address:		
NTP Server Port:	123	The default value is 123.
GMT Offset	-2 Hour(s)	
Update time interval	1440 Minutes	
FTP Config	Configure FTP connection of eWON to an FTP server	
FTP Server Address:		
FTP Server Port:	1742	The default value is 21.
User Name:		
Password:		
Use Passive Mode	<input type="checkbox"/>	
RESET Request	This command will Reset eWON after every changes in the config have been stored	
Reset request	<input type="checkbox"/>	

Figure 11: eWON main system setup - Setup for outgoing actions

Control	Description
SETUP FOR ALARM ACTIONS	
Action retrig interval	If the action on alarm first failed, then a new action on alarm will be triggered, only if the alarm condition is still true AND if the alarm has not been acknowledged yet. The default value for this parameter is one day (86400 seconds).
Try action	This parameter defines the number of times the action will be retried in case of errors. The value of this number must be greater than 1.
Action retry interval	This parameter defines the interval between two actions attempts if an error occurs. The value for this parameter must be greater than 10.

Table 6: setup for alarm actions controls

Control	Description
SETUP FOR OUTGOING ACTIONS	
SMTP server address	The IP address of the SMTP server where the Email notification will be sent. Put SMTP server IP address. If you only know the name of the SMTP server (like smtp.domain.com) see the chapter "Finding the IP address of a given host" on page 226. Note that it is possible to send mails towards an Exchange server when eWON is located inside from an Intranet, providing the IMC (Internet Mail Connector) ad-in is installed on the Exchange Server, and this service is configured to accept incoming mails sent by eWON.
SMTP Server port	Usually the value is 25. In case of doubt, contact your Internet provider or your Network Administrator to check it.
Email "From" User name	The name of the eWON Email account. For instance: ewon@compuserve.com. Will appear in the FROM field of the message sent.
User name	SMTP AUTHENTICATION: place here the user name. (leave empty if no authentication is required)
Password	SMTP AUTHENTICATION: place here the password. (leave empty if no authentication is required)
Enable NTP clock update	To update the eWON date & time the eWON is able to make a connection automatically on a NTP (Network Time Protocol) timeserver. If you want to use that functionality, check the checkbox and fill the next edit boxes.
NTP server address	The IP address of the NTP (Network Time Server Protocol) server. You can easily find a list of NTP servers by using any Web search engine. If you are working on your own network without a real connection to Internet, we recommend NTP server software http://www.haytech.com.au/TimeServer/TimeServer.htm . If you only know the name of the NTP server (like canon.inria.fr) see the chapter "Finding the IP address of a given host" on page 226. Note: eWON does not consider the DST data (Daylight Saving Time).
NTP Server Port	Usually 123. In case of doubt, contact your Internet provider or your Network Administrator to check it.
GMT Offset	Enter here the offset in hours between your local time zone and the GMT time zone. This information is required for correct automatic time update.
Update Time Interval	Interval in minutes for automatic connection to the NTP Server. Default is 1440= 1 day.
FTP server address	The name of the FTP server where the Put FTP command is issued (alarm action or Script direct PUTFTP command). Enter FTP server IP address. If you only know the name of the FTP server (like domain.com) see the chapter "Finding the IP address of a given host" on page 226.
FTP server port	Usually the value is 21. In case of doubt, contact your Internet provider or your Network Administrator to check it.
User name	The FTP client user name defined on the FTP server.
Password	The password for the given FTP client.
Use Passive Mode	When checked, all FTP transactions are performed in passive mode.

Table 7: eWON setup for outgoing actions controls

Click on **Update Config** or **Cancel** button when you have filled-in this first part of the eWON configuration.

4.3.1.3 Resetting eWON

Resetting eWON is sometimes necessary (i.e. to validate some configuration changes).

eWON version 4 offers two ways to perform a reset:

- From the **General configuration page** (chapter “Reset Request from the General Configuration page” on page 26)
- By using the "REBOOT" Basic command, as described in chapter “REBOOT” on page 161.

4.3.1.3.1 Reset Request from the General Configuration page

This way of resetting eWON did not change since version 3. You always can reset eWON by validating the checkbox that stands at the very bottom of the main (general) configuration of eWON web site, on which you access this way from the **Main Menu** navigation bar: **Configuration/System setup/General/General (Main submenu)**:

RESET Request		This command will Reset eWON after every changes in the config have been stored	
Reset request	<input type="checkbox"/>		
<input type="button" value="Update Config"/> <input type="button" value="Cancel"/>			

Figure 12: eWON General system setup - Reset Request

Any change that you enter in the General configuration page (or for example changing the eWON IP address) requires that you perform a first level reset of eWON for this change to be validated. Activating the **Reset request** checkbox, then clicking on the **Update Config** button triggers the reset process. You will have to wait that eWON comes back to its normal state after restarting to get access to it again.

4.3.2 SNMP Setup

The second page from the **System setup** is the **SNMP setup**. All the global SNMP management is done here.

SNMP Config				
Communities				
Community 1	<input type="text"/>	Read <input type="checkbox"/>	Write <input type="checkbox"/>	
Community 2	<input type="text"/>	Read <input type="checkbox"/>	Write <input type="checkbox"/>	
Community 3	<input type="text"/>	Read <input type="checkbox"/>	Write <input type="checkbox"/>	
Community 4	<input type="text"/>	Read <input type="checkbox"/>	Write <input type="checkbox"/>	
Community 5	<input type="text"/>	Read <input type="checkbox"/>	Write <input type="checkbox"/>	
Hosts				
Accept SNMP from any host <input type="checkbox"/>		If disabled, then access is only allowed for hosts below that have "Allow Access" checked.		
Host 1	Ip Address <input type="text"/>	Community <input type="text"/>	Trap <input type="checkbox"/>	Allow Access <input type="checkbox"/>
Host 2	Ip Address <input type="text"/>	Community <input type="text"/>	Trap <input type="checkbox"/>	Allow Access <input type="checkbox"/>
Host 3	Ip Address <input type="text"/>	Community <input type="text"/>	Trap <input type="checkbox"/>	Allow Access <input type="checkbox"/>
Host 4	Ip Address <input type="text"/>	Community <input type="text"/>	Trap <input type="checkbox"/>	Allow Access <input type="checkbox"/>
Host 5	Ip Address <input type="text"/>	Community <input type="text"/>	Trap <input type="checkbox"/>	Allow Access <input type="checkbox"/>
<input type="button" value="Update Config"/> <input type="button" value="Cancel"/>				

Figure 13: eWON SNMP setup main menu page

4.3.2.1 Communities

The communities are defined here. These are acting like a "login-password" feature. Please refer to the SNMP standard for detailed explanations. Up to five different communities can be set up in the eWON. Each community has different read and write attributes. In the eWON, each of them can be specified for read and/or write right. The standard *public* community is defined here as read-only.

4.3.2.2 Hosts

The hosts that can access the communities and/or receive the generated SNMP traps have to be specified. If the checkbox Access SNMP from Any Host is selected, any IP address will be granted to browse the SNMP tree of the eWON.

The traps are always sent to known IP addresses, defined in the following fields. Each host is determined by its IP address and by the community he is working on. For each host, the user can specify whether he can receive traps and/or browse the SNMP tree.

A MIB file describing the SNMP structure and OID of the eWON is available on our web site.

Traps can originate from three different events:

System traps	At cold boot and at soft reset, the eWON sends system traps with identification 0 and 3 respectively.
Basic scripting traps	The <i>sendtrap</i> Basic function can send a trap at user request.
Alarm event traps	The eWON automatically sends a trap on alarm, containing alarm information.

Table 8: events that generate a SNMP trap

The parameters for Basic traps are as follows:

Param 0	Text string [0...255]
Param 1	Integer 32bits

Table 9: Basic SNMP traps parameters

The parameters for Alarm traps are as follows:

Param 0	Tag name	(Text [0..63])
Param 1	Alarm message	(Text [0..255])
Param 2	Value of the Tag in alarm	(Integer 32bits)
Param 3	Alarm status	(Integer)
Param 4	Alarm Type	(Integer)

Table 10: Alarm traps parameters

4.3.3 Virtual Com Port Setup

4.3.3.1 Introduction

VCOM (or more precisely « Virtual COM port ») is a technology that consists in using the eWON's serial port as if it was a standard COM port of a PC running Windows.

Actually, a piece of software is installed on the PC; this software is a kind of driver that simulates additional COM ports on your PC. The PC is communicating with the eWON and the eWON serial port outputs all data transferred through the PC virtual serial port, the PC virtual port receives all data received by the eWON's serial port.

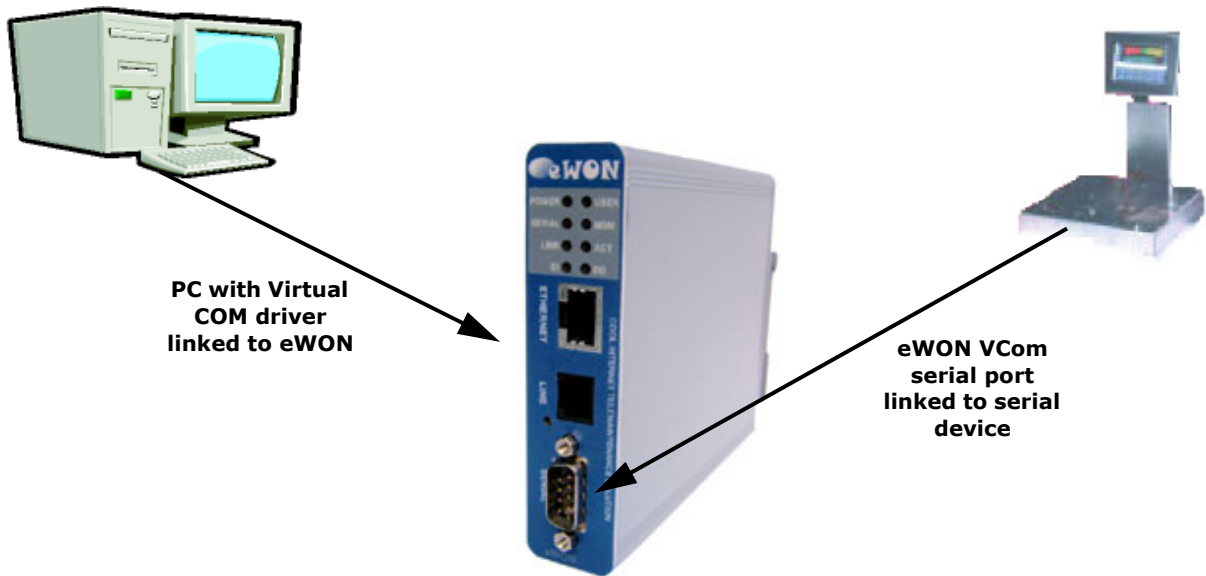


Figure 14: Virtual COM Port link used to read data on serial devices through eWON

4.3.3.2 The different VCOM modes

The eWON supports two kinds of VCOM mode and an additional mode which is not exactly classified as a VCOM mode although it has the same purpose.

4.3.3.2.1 RAW TCP

The first mode is called RAW TCP. This is a basic mode where the PC opens a TCP/IP socket to the eWON on a predefined port. This socket is used to exchange data from and towards the eWON's serial port.

The Serial port configuration (baud rate, parity, etc.) must be defined in the eWON's configuration. The serial port signals (RTS, DTR, etc.) are not exchanged between the PC and the eWON.

This mode can also be used to create simple TCP/IP applications that need to communicate through the eWON's serial port, as the only requirement is to open a TCP/IP socket on the eWON.

4.3.3.2.2 TELNET RFC2217

This is a more complex communication protocol between the PC and the eWON. In addition to the RAW mode this mode allows to control remotely the eWON's serial port. Every configuration change that is applied to the PC virtual COM port is automatically applied to eWON serial port. For example if you open a HyperTerminal on the PC's virtual serial port and you change the serial port's baud rate, the eWON's serial port baud rate will change accordingly.

Another feature of the TELNET RFC2217 mode is its ability to change the modem line status. This means that the RTS/CTS, DTR, DCD (etc.) levels of the eWON's physical port are reflected to the PC virtual port and vice-versa.

This protocol is called TELNET RFC 2217 because it has been standardized and described in an RFC specification. It means that any client supporting the RFC2217 protocol can use the eWON as a virtual port server.

4.3.3.2.3 MODBUS/TCP MODBUS/RTU Gateway

Although it is possible to use this technology to transfer almost any type of data through the virtual serial port, some protocols require special handling for efficient operation. ModbusRTU is one of these protocols and VCOM technology does not apply well to that protocol. For Modbus RTU communication it is recommended to use the MbsTcpCOM software and use the eWON as a ModbusTCP to ModbusRTU gateway.

4.3.3.3 PC Virtual COM Port driver

There are different options for creating virtual serial ports on the PC side.

4.3.3.3.1 MbsTcpCOM

This software provided by ACT'L is used to create ONE virtual COM port on the PC, the virtual COM port must only be used by software that are supposed to communicate on the PC serial port in ModbusRTU.

The ModbusRTU slave is actually connected to the eWON's serial port.

The PC software believes it is talking directly to the device through its serial port while it actually "talks" to its virtual serial port that transfers all the data to the eWON by actually translating every requests in ModbusTCP (without interpreting them).

This software is downloadable for free from <http://www.ewon.biz>.

4.3.3.3.2 Serial/IP

Tactical Software provides this software: <http://www.tactical-sw.com>

This software is not for free. It can be purchased directly from the Tactical software website.

An 30 days-evaluation version is available for download from this site.

The software supports both RAW TCP and Telnet RFC 2217 modes and works on all Windows version from Windows 95 to Windows XP.

The software can create up to 256 virtual serial ports that communicate with multiple eWONs.

4.3.3.4 eWON VCOM Configuration

4.3.3.4.1 Introduction

All serial ports can be used for VCOM.

Depending of the eWON type, you have 1, 2 or 4 serial ports.

The COM1 is always the serial port 1 of the eWON.

The COM2 is always the MODEM port (even if there is no modem present on your eWON, the COM2 exist but is useless).

The COM3, if present, is linked to the serial port 2 (SER2), the full RS-232 port on eWON4002.

The COM4, if present, is linked to the serial port 3 (SER3).

As COM2 is the modem, it should only be used for debug purpose because when the port is used by VCOM it is not available to PPP or SMS communication.

4.3.3.4.2 Web configuration

You get access to the COM1 (or COM2) VIRTUAL PORT CONFIGURATION page by following this path from the **Main Menu** navigation bar: **Configuration/System Setup/General/Virtual Com** (from the **General** menu):

COM Port: SER1 Port (COM:1)

COM1 VIRTUAL PORT CONFIGURATION		
General		
Port Type:	Disabled <input type="button" value="v"/>	Disable if not used to avoid port conflict.
TCP Port	23 <input type="text"/>	Default to 23 (telnet).
Poll signal interval	100 <input type="text"/> MSec	Check Modem line interval. Default to 100 MSec.
Debug	<input type="checkbox"/>	Warning: will slow down system.
Access Management		
Always accept new client	<input checked="" type="checkbox"/>	If a new client connects, the previous client connection is closed.
Inactivity Timeout	0 <input type="text"/> MSec (0=disabled)	Close TCP connection if no TCP data received after this time.
Line Parameters		
Baud Rate:	9600 <input type="button" value="v"/>	Init values for RFC 2217. Permanent values for Raw TCP.
Data Size	8 <input type="button" value="v"/>	
Parity	None <input type="button" value="v"/>	
Stop Bit(s)	1 <input type="button" value="v"/>	
HW Mode	Full Duplex NO Handshaking <input type="button" value="v"/>	

Figure 15: eWON Virtual COM port configuration page

You may first choose the COM port you want to configure in the **COM Port**: scrolling menu. You can then define the following parameters:

Control	Description
Port Type	If Disabled is selected, then VCOM will not be used on this port.
TCP Port	The PC will have to connect to that TCP/IP port to communicate through the eWON serial port. REM: if multiple VCOM are defined on the same eWON, they must all use a different TCP/IP port.
Poll Signal Interval	This parameter is only used in TELNET RFC2217 mode. In that mode the eWON will scan the modem port for changes in modem line input levels (CTS, DSR, DCD, RING) thus this parameters defines the scan rate. The default value of 100msec should be faster enough for most applications. Debug: Will enable debug logging in the real time event log (\$dtRE, see "Export Block Descriptor" on page 192).

Table 11: eWON virtual COM port configuration controls

Always accept new clients	<p>When a PC is connected to the eWON, then a socket is opened by the PC to the eWON. If for example the PC suddenly switches off, the eWON will not know that the socket should be closed, and when the PC will try to connect again, the eWON will refuse its connection. This option is provided in order to avoid that situation; when checked, the eWON will always be ready for a new connection, and the new connection will replace the previous connection.</p> <p>This means also that if one PC is connected to the VCOM port, and that another PC tries to connect with this option checked, the new PC connection will be accepted and the existing PC connection will be closed.</p>
Inactivity Timeout	<p>There is another way to avoid that an unused but opened socket prevents access to the eWON (if the previous option has NOT BEEN checked).</p> <p>If you set this option with a value different of 0, then the eWON will close its VCOM socket if there is no communication for a given amount of time.</p> <p>If serial communication is supposed to occur all the time this option can be useful, but if silence in communication is expected, then this option will obviously not apply.</p>
Line parameters	<p>Except for HW Mode, which is only configurable through the WEB interface, these parameters are only used when the port is configured in RAW TCP mode, because in TELNET RFC2217 mode the PC virtual port will usually change them. In any case, this will define the initial or default values.</p>
HW Mode	<p>REM: This mode can not be controlled remotely by RFC 2217*</p>

Table 11: eWON virtual COM port configuration controls

***HW Mode: further information will be added soon.**

When using modem port for VCOM, the following must be considered:

- **Modem serial port is normally owned by PPP:**
- **If an SMS transfer is in progress and a VCOM client tries to connect, the VCOM connection will fail.**
- **If a VCOM client is connected, and an SMS must be sent, the SMS sending will fail.**
- **When an SMS transfer or a VCOM connection ends, the PPP is again the owner of the modem serial port.**

4.3.4 Diagnostic

4.3.4.1 Overview

This part of eWON configuration allows you to fine-tuning the way you monitor eWON, concerning the events, that gives the user the ability to diagnose quickly and efficiently any trouble that could happen.

The second choice in the **Diagnostic** menu is **PPP Dump**, which allows you to log the history of any PPP communication to and from eWON.

4.3.4.2 Events logging

4.3.4.2.1 Overview

Event logging Configuration gives you the ability to define the reporting level you want to get a diagnostic from most important part from eWON's features.

The three different reporting levels that can be defined are:

Trace	The events with level "Trace", "Warning" and "Error" will be logged
Warning	The events with level "Warning" and "Error" will be logged
Error	Only the critical events will be logged

Table 12: reporting levels - explanations

Warning: if for example you define the "Error" reporting level for "IP communication", then you won't be able to retrieve the reporting information concerning the "Warning" and "Trace" levels in the events log file, that means you have to wonder exactly what are the eWON features you want to keep an eye about potential issues or not.

4.3.4.2.2 Event logging Configuration page

Event logging Configuration			
Event Class	Reporting Level		
	Error	Warning	Trace
Initialisation	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
Configuration	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
IO Server	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
Modem Communication	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
IP Communication	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
Serial Communication	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
Kernel	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
Web Interface	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
Security	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
Other Applications	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>

Figure 16: Event logging configuration page

Control	Description
Initialisation	Allows you to define the level of monitoring about the events concerning eWON boot.
Configuration	Allows you to define the level of monitoring about the events concerning eWON configuration.
IO Server	Allows you to define the level of monitoring about the events concerning the IO Servers that eWON manages.
Modem Communication	Allows you to define the level of monitoring about the events concerning eWON's modem communications (incoming and outgoing).
IP Communication	Allows you to define the level of monitoring about the events concerning eWON's IP communications.
Serial Communication	Allows you to define the level of monitoring about the events concerning eWON's serial communications.
Kernel	Allows you to define the level of monitoring about the events concerning eWON's kernel.
Web Interface	Allows you to define the level of monitoring about the events concerning eWON's Web Interface.
Security	Allows you to define the level of monitoring about the events concerning eWON's security
Other Applications	Allows you to define the level of monitoring about the events concerning eWON's features that are distinct from all the one that are listed above in this table.

Table 13: events logging configuration controls

4.3.4.3 PPP Dump

Warning: depending on eWON's version (c.f. Table on page 233)

4.3.4.3.1 Overview

WARNING: this configuration is volatile, which means that the dump.ppp file will be cleared each time when eWON is rebooted.

The *dump.ppp* file containing the logged data can be used in the following ways:

- it can be sent as an attachment to an email or by using the \$dtPP Export Block Descriptor
- it can be found in the eWON FTP root and then downloaded locally or on another FTP folder
- it can be opened and analyzed by using EtherReal, which is a tool used to analyze TCP frames (<http://www.ethereal.com/>)

4.3.4.3.2 PPP Dump Configuration

PPP Dump Configuration		
Log Incoming call	<input type="checkbox"/>	
Log Outgoing call	<input type="checkbox"/>	
Log Size (bytes)	<input type="text" value="0"/>	This is allocated on eWON memory
Append to log	<input type="checkbox"/>	Do not clear log between connections, append until full.
Log following connections	<input type="text" value="0"/>	0 for all. Log the following N connections (appending or clearing)
<input type="button" value="Update Config"/> <input type="button" value="Cancel"/>		
Clear log now	<input type="button" value="Clear Log"/>	Clear log

Figure 17: PPP Dump configuration page

Control	Description
Log Incoming call	Logs communications when eWON is acting as a PPP server
Log Outgoing call	Logs communication when eWON is acting as PPP Client (connects to a server)
Log Size (bytes)	Number of bytes allocated for logging PPP communications. When the log is full, logging stops (this does not prevents communication from continuing) Maximum log size is 1MByte (Log Size between 50000 and 100000 are sufficient for debug purpose)
Append to log	Prior to append to log, you must clear log manually using the Clear Log button in the PPP Dump configuration page
Log following connections	eWON will log only the N next connections, the number of connections will be decreased each time a new connection is logged; when the last connection has been logged, then the counter will be set to -1 (to prevent further connections to be logged). When the value of this counter is 0, then all connections are logged. You can manually set the counter to -1, in order to suspend connection logging, but if you want to release the log buffer, then you should disable the Log Incoming call and the Log Outgoing call options. In case you want to log multiple connections, you can choose to Append data to the log or not.
clear log now	Clicking on this button will clear the PPP dump log.

Table 14: PPP dump configuration controls

4.3.5 Clock Setup section

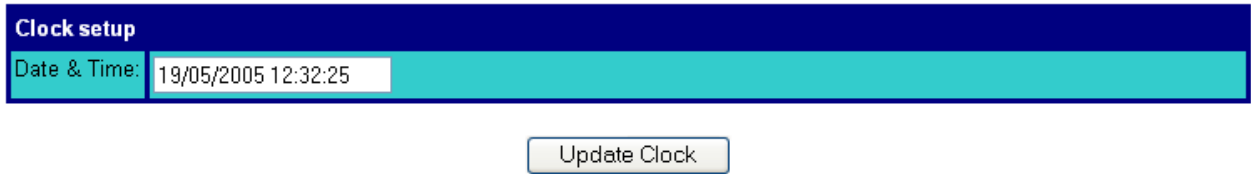


Figure 18: eWON clock setup field

The eWON's real time clock can be updated manually with this dialog window. Fill in the edit box with date and time and click on **Update Clock** button.

An event will be added to the event log signaling the time update and the time offset between old and new time.

Important: Updating the time may result in duplicate points stored in non-chronological order in the eWON files (alarms, events and historical).

4.3.6 COM Setup section

Warning: depending on eWON's version (c.f. Table on page 233)

The third page of the **System setup** is the communication setup. All Ethernet, modem and PPP TCP/IP connectivity parameters are defined here. This page is divided into 7 sub-menus, which classifies the COM setup into functional systems:

Ethernet, Modem, Dialup (PPP), Callback, Router (Filter), IP Services and Default Config (Communication setup only).



Figure 19: eWON COM setup main menu page - eWONs that embeds a modem

- **Important: This configuration is not saved in the eWON flash file system, its means that formatting the eWON will not erase this configuration. This allows formatting the eWON remotely and ensuring that communication is still possible after formatting. Nevertheless, the *Default Config* menu allows the user to return to default well known communication values.**

4.3.6.1 Ethernet

This tab contains the Ethernet setup of the eWON, setting up the LAN side communication.

ETHERNET SETUP	
Address Setup	
eWON Ethernet IP address	10.0.120.71
eWON Ethernet IP mask	255.255.0.0
eWON Ethernet IP Gateway	0.0.0.0
eWON Use BOOTP	<input type="checkbox"/> (Ethernet address, mask and gateway will be provided by BOOTP)
WARNING: if "Use BOOTP" is enabled and no BOOTP server is present, use button to unlock eWON during boot sequence.	
DNS Setup	
Leave blank (or 0.0.0.0) if no DNS or if DNS automatically allocated by PPP server	
Primary DNS IP address	0.0.0.0
Secondary DNS IP address	0.0.0.0

Figure 20: eWON Ethernet setup

Control	Description
Address Setup	
eWON Ethernet IP address	IP address of the eWON on the LAN side. Use this IP address to connect to the eWON using a LAN connection. It is 4 numbers from 0 to 255 separated by a dot.
eWON Ethernet IP mask	eWON Ethernet server subnet mask, used to determine the address range of the LAN connection (your network).
eWON Ethernet IP gateway	eWON Ethernet server gateway, which is the IP address used to forward information to other networks. When a gateway IP address is set, ALL outgoing actions (except SMS) will pass by the Ethernet gateway (no dial out using phone line will occur). That means that if you want to redirect some actions through Ethernet and some other by phone (PPP), you then must leave this field empty, or "0.0.0.0".
eWON Use BOOTP	Forces eWON to wait its IP address from a BootP Server (see chapter "TCP/IP Bootstrap Protocol (BOOTP)" on page 234).
DNS Setup	
Primary DNS IP address	IP address, It is 4 numbers from 0 to 255 separated by a dot, of the primary Domain Name Server of your domain or ISP provider.
Secondary DNS IP address	IP address, It is 4 numbers from 0 to 255 separated by a dot, of the secondary Domain Name Server of your domain or ISP provider.

Table 15: eWON Ethernet setup configuration page

- Versions older than 4.0 S8:
Click on the **Update Ethernet Setup** to validate your changes (eWON reboot is required).
- From version 4.0 S8:
No reboot if required if you only update the eWON Ethernet IP gateway.

4.3.6.2 Modem

Warning: depending on eWON's version (c.f. Table on page 233)

4.3.6.2.1 Modem configuration (PSTN modem)

This tab contains the Modem setup of the eWON and stands in a distinct page from Dialup Setup since version 4. Its appearance differs depending on the type of modem that eWON embeds; if a PSTN modem, then the modem settings page will only content the **Modem Init String** field (see figure below):

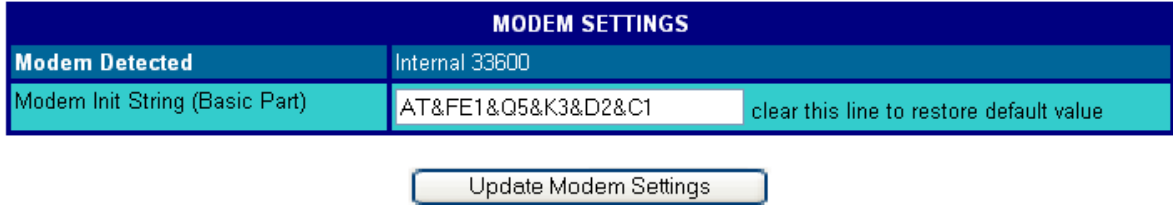


Figure 21: Modem settings for eWON with PSTN modem

Control	Description
Modem Detected	Textual description of the eWON internal modem detected, if available. The text that displays is always " <i>Internal</i> ", followed by the modem type or speed (i.e. "33600" in the screenshot above).
Modem Init String (Basic Part)	This string is used to configure and to initialize the modem. Any change in this string could prevent communications to succeed, for that reason only qualified users should modify this string. It can be useful to modify this string i.e. if you install eWON in a country where the telephonic network is different than in your country Clearing this string will result to applying a well known default initialization string (after you have validated your choice by clicking on the Update Modem Settings button).

Table 16: eWON PSTN modem settings controls

4.3.6.2.2 Modem configuration (GSM/GPRS modem)

If eWON embeds a GSM/GPRS modems, then the modem page will have a very different look:

MODEM SETTINGS			
Modem Detected	Internal BIBAND GSM		
GSM Pin Code (reboot required)	1234	Signal Level: 30	Network: Home network
Modem Init String (Basic Part)	AT&FE0&D2&C1+IFC=2,2 <small>clear this line to restore default value</small>		
GPRS Settings			
PDP context definition		<input type="checkbox"/> Enable this config	
Access Point Name	<input type="text"/> WARNING: case sensitive		
Quality Of Service Profile (Requested)		<input type="checkbox"/> Enable this config	
precedence	0: Subscribed <input type="button" value="v"/> (default: Network Subscribed Value)		
delay	0: Subscribed <input type="button" value="v"/> (default: Network Subscribed Value)		
reliability	0: Subscribed <input type="button" value="v"/> (default: Network Subscribed Value)		
peak	0: Subscribed <input type="button" value="v"/> (default: Network Subscribed Value)		
mean	0: Subscribed <input type="button" value="v"/> (default: Network Subscribed Value)		
Quality Of Service Profile (Minimum Acceptable)		<input type="checkbox"/> Enable this config	
precedence	0: Subscribed <input type="button" value="v"/> (default: Network Subscribed Value)		
delay	0: Subscribed <input type="button" value="v"/> (default: Network Subscribed Value)		
reliability	0: Subscribed <input type="button" value="v"/> (default: Network Subscribed Value)		
peak	0: Subscribed <input type="button" value="v"/> (default: Network Subscribed Value)		
mean	0: Subscribed <input type="button" value="v"/> (default: Network Subscribed Value)		

Figure 22: Modem settings for eWON with GSM/GPRS modem

Control	Description
Modem Settings	
Modem Detected	Textual description of the eWON internal modem detected, if available. The text that displays is always " <i>Internal</i> ", followed by the modem type or speed (i.e. "BIBAND GSM" in the screenshot above).
GSM Pin Code (reboot required)	Enter here the currently valid PIN code on the SIM card you have inserted in eWON. This PIN code will be taken in account only after you have power OFF/power ON eWON.
Signal Level	Indicates you the current signal level for your GSM/GPRS communication. This signal range must be between 20 and 31 (signal levels lower than 18 could work, but the communications could be slower or even interrupted). Check your local environment and your antenna isolation/power if you get a 0 or a 99. Contact us at sales@ewon.biz if you need any information about how to order an antenna that fits your needs.
Network	This field indicates if you are able to connect to the GSM/GPRS network. You should read " Home network " in order to communicate safely.
Modem Init String (Basic Part)	This string is used to configure and to initialize the modem. Any change in this string could prevent communications to succeed, for that reason only qualified users should modify this string. It can be useful to modify this string i.e. if you install eWON in a country where the telephonic network is different than in your country. Clearing this string will result to applying a well known default initialization string (after you have validated your choice by clicking on the Update Modem Settings button).

Table 17: GSM Modem configuration controls

GPRS Settings	
<p>In order to generate a GPRS outgoing communication, the user must enter "GPRS" in the Server Phone Number field from the Outgoing Calls config part from the Dial Up Configuration page c.f. chapter "Dial up (PPP)" on page 42.</p> <p>Notes:</p> <ul style="list-style-type: none"> • Each part of the GPRS configuration can be enabled or not. • If your GPRS provider requires authentication when login in the GPRS network, the login and password must be entered in the Dial Up Configuration page c.f. chapter "Dial up (PPP)" on page 42. • Some network operators use authentication to identify the mobile phone connecting to their GPRS network. 	
PDP context definition	Packet Data Protocol Context:
Access Point Name	(APN): Enter the Internet address of your access point. Your service provider supplies the address.

Table 18: eWON GPRS main settings controls

Quality Of Service Profile (Requested)	
<p>Check with your network operator before changing any of the QoS (Quality of Service) settings. The settings Precedence, Delay, Peak rate, Reliability and Mean Rate are all set by default to "Subscribed", which means that your operators default QoS (Quality of Service) values will apply.</p>	
precedence	a numeric parameter which specifies the precedence class
delay	a numeric parameter which specifies the delay class
reliability	a numeric parameter which specifies the reliability class.
peak	a numeric parameter which specifies the peak throughput class
mean	a numeric parameter which specifies the mean throughput class
Quality Of Service Profile (Minimum Acceptable)	
precedence	a numeric parameter which specifies the precedence class
delay	a numeric parameter which specifies the delay class
reliability	a numeric parameter which specifies the reliability class.
peak	a numeric parameter which specifies the peak throughput class
mean	a numeric parameter which specifies the mean throughput class

Table 19: eWON GPRS quality of services (QOS) settings controls

4.3.6.2.3 Leased line configuration

PSTN modem types MT5634SMI-xx have the ability to work in Leased Line mode. In this mode, you are able to connect by phone two eWONs with simple wires. The two eWONs must have a MT5634SMI-xx modem.

One side acts as the client and you must place the LLCLT command in the init string. The other side acts as the server and you must place the LLSRV command in the init string. The init string MUST start by LLCLT or LLSRV for LeasedLine mode operation but the init string can continue with other params.

Example:

LLSRV;AT&FE1&Q5&K3&D2&C1

Or:

LLSRV

or

LLCLT

or

...

The link is established by the CLIENT. This Client eWON can be configured with following parameters:

Modem Init String	LLCLT
Call direction allowed	Outgoing Only
Primary Server ' Phone number	0
Primary Server ' User Name	adm
Primary Server ' Password	adm

Note: Username and password of an user on the Server eWON.

The Server eWON can be configured with following parameters:

Modem Init String	LLSRV
Call direction allowed	Incoming Only

- When the dialout occurs, it takes up to 2 minutes to synchronize both eWONs.
- The Server modem leased line (LLSRV) cannot generate outgoing calls, or it would generate an error.
- The Client modem leased line (LLCLT) cannot receive incoming calls.

4.3.6.3 Dial up (PPP)

Warning: depending on eWON's version (c.f. Table on page 233)

This tab contains the PPP setup of the eWON, setting up the PPP (Point to Point) settings for the server and client functions of the eWON engine.

DIAL UP CONFIGURATION	
Global Dialup Config	
Call direction allowed	Incoming & Outgoing ▾
Use incoming for outgoing	<input type="checkbox"/> Use connected client connection (if any) for outgoing operations
Incoming Calls Config	direction allowed must be 'Incoming' or 'Both'
Idle time before hanging up	240 seconds (min. 60 sec.)
Enable compression	<input checked="" type="checkbox"/>
eWON PPP server IP address	202.0.0.240
eWON PPP server IP mask	255.255.255.0
eWON PPP server gateway	0.0.0.0
PPP client allocated IP address	202.0.0.1
Outgoing Calls Config	direction allowed must be 'Outgoing' or 'Both'
Dial-out timeout	180 seconds
Idle time before hanging up	120 seconds (min. 60 sec.)
Delay between dialout retry	60 secondes
Max outgoing call duration	60 minutes (0 for no limit)
Hang up if no outgoing action after	-1 minutes (if -1 hangup occurs after "idle time")
Enable compression	<input checked="" type="checkbox"/>
Require secure authentication (CHAP)	<input type="checkbox"/> (otherwise allow PAP - send your password as clear text)
Primary Server	
Server Phone Number	00000 (Or phone number = GPRS)
User Name	
Password	
Secondary Server	Leave blank if not defined
Server Phone Number	(Or phone number = GPRS)
User Name	
Password	
Calls Budget management	Applies to outgoing calls only
Allocated budget	24 Hours (0 for no limit).
Reset budget period	168 Hour(s)
Current period budget: 24:00:00	Hour(s), (empty for 'no change')

Figure 23: eWON PPP Dialup configuration window

Control	Description
Global Dialup Config	
Call direction allowed	<p>This list box lists all modes of allowed connections types. This means: disabled, incoming, outgoing and both incoming and outgoing.</p> <p>Disabled means that no call will be answered and that no outgoing call will be initiated, under any circumstance (this does not concern the SMS actions, that means you are able to send a SMS from eWON, even if "Disabled" is selected).</p> <p>Incoming means that incoming calls will be answered, but that no outgoing call will ever be initiated (except for SMS).</p> <p>Outgoing means that outgoing calls will be initiated, but that no incoming call will ever be answered.</p> <p>Incoming & Outgoing means that incoming calls will be answered, and that outgoing calls will be initiated on request by the system.</p>
Use incoming for outgoing	<p>When checked, this ensures that when an incoming call is undergoing, no external event, such as alarm Email, will drop the line in order to initiate a new connection. If an alarm has to be sent through the PPP connection (FTP, Email, ...), the current PPP link will be used. Be aware that SMS alarms will always drop the line, whatever the value of this checkbox. Also remember that if this box is unchecked and that an Email for example has to be sent while connection has been established by a user in order to browse the eWON, if the Email can be sent through the Ethernet link, the PPP link will NOT be dropped.</p>

Table 20: eWON PPP Global Dialup configuration controls

Incoming Calls Config	
Idle time before hanging up	After this amount of time without data transfer on the PPP link between the eWON (any type of PPP packet) and a remote host (Computer).
Enable compression	Enable the compression negotiation request when an incoming call occurs. This includes all compression modes known by the eWON PPP engine (Van Jacobson, header compression,...).
eWON PPP server IP address	PPP server Internet protocol (IP) address of the eWON. Use this IP address to connect to the eWON using a RAS connection. It is 4 numbers from 0 to 255 separated by a dot.
eWON PPP server IP mask	eWON PPP server subnet mask, used to determine the address range of the PPP connection.
EWON PPP server gateway	eWON PPP server gateway of the which is the IP address used to forward information to other networks.
PPP client allocated IP address	Type the IP address that the eWON will allocate to the RAS client to establish the communication.

Table 21: incoming calls configuration controls

Control	Description
Outgoing Calls Config	
Table 22: outgoing calls configuration controls	
Dial-out timeout	Time allowed for the whole establishment of the PPP link to be up. This means modem call, modem negotiation, PPP negotiation and logon. This time includes all trials on each server. This is thus a global time.
Idle time before hanging up	After this amount of time without data transfer on the PPP link between the eWON (any type of PPP packet) and a remote host (Computer).
Delay between dialout retry	In case of an unsuccessful attempt to establish the outgoing communication, then eWON will retry to establish it again. This parameter will allow you to define the amount of time eWON will wait to try and establish again the outgoing action.
Max outgoing call duration	The maximum amount of time of the outgoing call. When this amount of time is reached, then eWON stops the PPP communication.
Hang up if no outgoing action after	When attempting to perform an outgoing call, an issue can prevent any action to occur. This allows you to define the time after which eWON will give up the outgoing action in case of no activity on the line.
Enable compression	Enable the compression negotiation request when an incoming call occurs. This includes all compression modes known by the eWON PPP engine (Van Jacobson, header compression,...).
Require secure authentication (CHAP)	If this box is checked, then the eWON explicitly requests CHAP authentication for the PPP link. If the other side cannot do CHAP, no connection will be made. If this box is leaved unchecked, then PAP (clear text password) is used.
Primary & secondary servers	The following parameters are the same for both servers. Two different servers can be set up. This ensures that if a server is down, the eWON can find a way out for PPP link. At the startup of the establishment of a connection, the primary server is always dialed first. If the connection cannot be established, the eWON tries the second server. If it fails, then it toggles back to the primary server. This is done until the dial-out timeout is reached.
Server phone number	Complete phone number of the server. A coma (,) can be used to insert a pause (can be useful i.e. if you have to go through a pabx).
User Name	User name of your ISP login for PPP link establishment.
Password	Password linked to the above login for PPP link establishment.

Control	Description
Calls Budget management	
Allocated Budget	This is the allocated time budget for outgoing calls. When a communication initiated by the eWON is undergoing, the Current period budget (remaining time) is decremented. When the allocated budget period is elapsed (next field), a new period is started, with this basic call timing package, which is therefore called Allocated budget.
Reset budget period	This is the time allowed in order to use the budget. After the call period is over, a new period is started and the new period timer is reset to this value. In the above example, this means that the current budget period is reset to 24 hours each 168 hours. The reset period is restored to its value each time one of the three configuration fields is modified.
Current period budget	This is the remaining call budget for the current period, expressed in hours:min:sec. A new budget can be provided; this restarts a new reset period. For example, in the above example 24 hours are remaining, if we force 100 hours as call budget, the reset period is reset. This means that for the newly started period, 100 hours are allocated. A new period will start again in a delay of 168 hours.

Table 23: calls budget management controls

Click on **Update Dialup Setup** button when you have filled this part of the eWON configuration.

4.3.6.4 Callback

Warning: depending on eWON's version (c.f. Table on page 233)

This tab contains the configuration controls that the eWON will use to perform callback operations.

CALLBACK CONFIGURATION	
General Callback Config	
Callback Enabled	<input checked="" type="checkbox"/> ('Outgoing' calls must be enabled in Dialup configuration)
Callback delay	30 seconds
Wait for user login for	1200 seconds
Dialup account	Primary dialup server (User callback is valid only if 'On User's request mode is selected')
Select one callback method: RING or USER'S REQUEST	
Callback on RING <input checked="" type="radio"/> Callback occurs when RING is detected	
Number of RINGS	5 (minimum 2)
Plus number of RINGS then On Hook	10 (minimum 5)
Callback on USER'S REQUEST <input type="radio"/> User must log on and request callback	
IP address publishing	
Publish IP address EMail	user@ewon.biz (Empty means no address publishing by Email)
No-ip Username (see http://no-ip.com)	myewon@no-ip.com (Empty means no address publishing by No-ip)
Dynamic DNS password	
Dynamic DNS Host name	myewon
Dynamic DNS Domain name	no-ip.com
Calls Budget management	See Dialup configuration

Figure 24: eWON PPP Callback configuration

Control	Description
General Callback Config	
Callback enabled	If this box is checked, then the eWON callback feature is enabled. This means that the eWON can be triggered by an external event in order to call a given phone number in order to establish a PPP link. This permits to let the eWON's phone line pay for the call, with a budget maintained by the eWON itself. This also ensures more security if you are calling back a private server. Do not forget to enable outgoing calls in the dialup configuration.
Callback delay	Once the eWON has been triggered, it will wait for this amount of time before dialing out. This is useful in order to release the phone line or perform any other action.
Wait for user login for	Once the eWON has been triggered, it will wait for this amount of time in order for the user to log-in. Otherwise, the call is dropped. It is clear that this delay has to be greater than the sum of the callback delay and the call establishment.
Dialup account	This list box permits to choose the server used for callback. The "User's request account" option can not be chosen when the trigger mode is "Ring", because the eWON does not know the server data at this time.

Table 24: general callback configuration controls

Control	Description
IP address publishing	
Callback on ring	This is a mutual exclusive option with Callback on User's request. Selecting this option disables the User Callback mode. When the eWON sees an incoming call, it will trigger its callback task.
Number of RINGS	(minimum value is 2): This is the number of rings needed before the eWON callback function is triggered. Assuming a value of 3 has been entered, the eWON will generate a callback if someone calls the eWON and lets the phone ring 5 times.
Plus number of RINGS then On Hook	(minimum value is 5): This is the number of rings that are necessary to avoid the callback trigger. Assuming a value of 5 has been entered, if the user tries to dial the eWON directly in, this can be achieved by ringing the phone more than the sum from the two values. With the current example, the eWON will pick up the line after more than 3+5=8 rings are detected.
Callback on User's request	<p>When this option is selected, the eWON accepts the incoming call and then the user can request a callback to a defined server.</p> <p>When connecting to an eWON with Callback on user's request, you will get the following webpage.</p> <p>You will start the callback by using the Callback button or abort (hang up the phone line) by using the "Close PPP Connection" button.</p> <p>The callback can be performed with the parameters specified in the user's configuration page.</p> <p>The delay the eWON will wait before initiating the callback can be specified.</p>

Table 25: user's callback controls

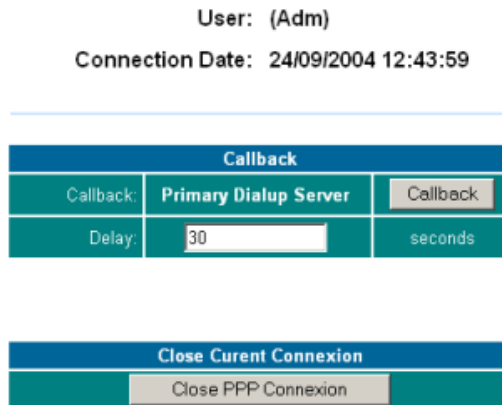


Figure 25: Callback on user's request at logon

Note: When connecting to an eWON with Callback on user's request, you will get the above webpage.

Control	Description
Selection of the callback method	
Publish IP address EMail	This field allows you to define the destination address by default that will receive the callback notification.
No-ip Username (see http://www.no-ip.com/)	No-ip is a service that allows you to publish on Internet a domain name that matches the IP address of your PC (fix or dynamic type) e.g. <i>myewon@no-ip.com</i> .
Dynamic DNS password	This field allows you to enter a valid password for the Dynamic DNS server you want to use.
Dynamic DNS Host name	This field allows you to enter a valid host name for the Dynamic DNS server you want to use e.g. <i>myewon</i> .
Dynamic DNS Domain name	This field allows you to enter a valid domain name for the Dynamic DNS server you want to use e.g. <i>no-ip.com</i> .

Table 26: IP address publishing controls

You will start the callback by using the Callback button or abort (hang up the phone line) by using the "Close PPP Connection" button. Please check the callback configuration that has been defined in the user's configuration (see screenshot below), especially if you enable the "Callback on user's request" checkbox in the callback configuration window.

Callback	
Enabled	<input type="checkbox"/> (Callback must also be globally enabled in System Config)
Callback phone number value is	Mandatory <input type="button" value="v"/> (Defines if user can change phone number)
Callback phone number	<input type="text"/> (User's login and password are used to connect to server)

Figure 26: eWON User callback setup

4.3.6.5 Router

Warning: depending on eWON's version (c.f. Table on page 233)

This tab contains the router configuration of the eWON, setting up the PPP (Point to Point) settings for the server and client functions of the eWON engine, in order to define the way you will use your eWON to accessing remote devices, and access to them through eWON as if they were on your local site.

ROUTER CONFIGURATION	
Ip Forwarding	
Enable IP forwarding	<input checked="" type="checkbox"/> Allow data to travel between Ethernet and PPP [1]
Enable NAT	<input type="checkbox"/> (IP forwarding is required for NAT) [2]
Enable Transparent Forwarding (TF)	<input type="checkbox"/> Connect to a LAN device using eWON (PPP) IP address. [2]
Security	
Authenticated routing	<input type="checkbox"/> Accept PPP packets only from authenticated user (for TF). [2]
Dial On Demand [2]	
<input type="radio"/> Accept dial on demand from NO ONE EXCEPT from: <input checked="" type="radio"/> Accept dial on demand from ANYONE EXCEPT from:	
IP Range	From: 0.0.0.0 To: 0.0.0.0
IP Range	From: 0.0.0.0 To: 0.0.0.0
IP Range	From: 0.0.0.0 To: 0.0.0.0
IP Range	From: 0.0.0.0 To: 0.0.0.0
[1] Updated only when eWON boots. [2] Updated every new PPP connection. NAT and Transparent Forwarding are only used for outgoing connections.	

Update Router Setup

Figure 27: eWON Router configuration

Control	Description
Enable IP forwarding	<p>If this box is checked, then the eWON IP forwarding feature is enabled. This means that a link can be performed between PPP and Ethernet IP packets. Please note that this is dangerous because it can connect your LAN to the internet directly.</p> <p>This feature can be used in order to connect a device to the internet through the eWON. For example, if an automaton is connected to the Ethernet LAN of the eWON and has the eWON IP address as gateway address, the eWON will perform any thing needed in order for the device to send its packets (dialout, IP translation ...). Example: a device sends a mail on the internet, while the eWON performs the PPP dial-out.</p> <p>Another use is to access a device located on the Ethernet LAN of the eWON, the user dials the eWON directly and then gains access to its distant device through its LAN IP address. Example: a user on a LAN with IP address range 192.168.0.xxx can access its device on the eWON LAN, with eWON IP address 10.0.0.81 and the distant device with IP 10.0.0.82 and having the eWON as a gateway. The eWON will for example assign the IP address 202.0.0.1 to the PPP adapter of the PC and take for its PPP adapter the IP address 202.0.0.240.</p>
Enable NAT	<p>This feature enables the Network Address Translation (NAT). If the device to reach is on the same LAN as the eWON, and if the user has correctly defined the address IP and the port for this device, then the eWON will redirect the packets towards it. This means that only the packets for the concerned port will be redirected towards the selected IP address. IP Forwarding must be activated for NAT to be active.</p>
Enable Transparent forwarding	<p>If this box is checked, the transparent forwarding feature of the eWON is activated. For this to work, the IP forwarding must also be enabled. With simple IP forwarding, it is not possible to access a device located on the eWON Ethernet LAN through the internet. For example if the eWON is connected to the internet through PPP (for example on alarm), it publishes its IP address by Email. In order to get access to its device, the user cannot simply type the IP address of the device like this was done with IP forwarding and direct eWON call.</p> <p>In this case, transparent forwarding is the only solution: when accessing the eWON, the user can request to perform transparent forwarding once he logs in. He can then see the window that is illustrated by the screenshot below this table. If the device is on the same network as the eWON and gets the eWON as gateway, and if the user configured the IP address of the device at eWON login, the eWON routes all packets to this device.</p> <p>This means that all IP packets, on all ports except 81 will be routed transparently to the selected IP address. In order to get access to the eWON web server, the user has to connect to port 81 of the eWON IP address. For example http://10.0.0.53:81 will access the eWON main page, while 10.0.0.53 will access the IP address 10.0.0.56 of the device located onto the LAN if the user selected this IP at logon. The user can clear transparent forwarding by getting access to the port 81 and using the "Clear transp forwarding" button. Note that FTP cannot be routed, since it includes the target IP address in its frame in text mode.</p>
Authenticated routing	<p>If this box is checked, the eWON will only accept to forward packets coming from the user who initiated the communication.</p>
Dial On Demand	<p>It is possible to establish remote connections towards the ISP by using the Dial On Demand feature. If selected, the eWON will then try to connect to the ISP each time a connection will be required (i.e. packets to send). Those fields allow you EITHER to exclude OR to select IP addresses ranges that are allowed or refused for Dial On Demand. Four different IP addresses ranges can be defined.</p> <p>Note: those fields will be updated at each every new PPP connection.</p>

Table 27: router configuration controls

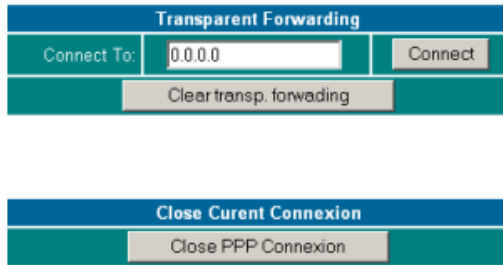


Figure 28: eWON PPP Router logon screen

4.3.6.6 IP Services

This page allows you to redefine the standard eWON's ports. The main interest of this feature is i.e. to be able to go through a firewall when all the ports with value less than 1000 are blocked for access by this firewall.

IP SERVICES SETUP		
HTTP Web Server		
Primary TCP port	80	default to 80
Secondary TCP port	81	not affected by transparent forwarding (default to 81).
FTP Server		
TCP command Port	21	default to 21 (Data port is this port -1)
Modbus/TCP Server		
Modbus/TCP Port	502	default to 502
Ethernet/IP Server		
Ethernet/IP Port	44818	default to 44818
These settings are only updated when the eWON boots.		
<input type="button" value="Update IP Services Setup"/>		

Figure 29: IP services configuration

Control	Description
HTTP Web Server	
Primary TCP port	Allows you to redefine eWON's Primary TCP Port (of which default value is set to 80).
Secondary TCP	Allows you to redefine eWON's Secondary TCP Port (of which default value is set to 81).
FTP Server	
TCP command Port	Allows you to redefine the TCP command Port from eWON's FTP Server (of which default value is set to 21).
Modbus/TCP Server	
Modbus/TCP Port	Allows you to redefine the Modbus/TCP Port from eWON's Modbus/TCP Server (of which default value is set to 502).
Ethernet/IP Server	
Ethernet/IP Port	Allows you to redefine the Ethernet/IP Port from eWON's Ethernet/IP Server (of which default value is set to 44818).

Table 28: IP services configuration controls

Validate your changes by clicking on the **Update IP Services Setup** button. The changes in those fields will be taken in account only after eWON is rebooted.

4.3.6.7 **Default Config**

Using this menu, the user can return to a default well known configuration of the communication setup of the eWON. This will NOT modify anything on the other configuration of the eWON. The IP address will not be modified. This option requires a forced reboot of the eWON (software or hardware).

Also remember that the communication setup is NOT affected by a format of the eWON.

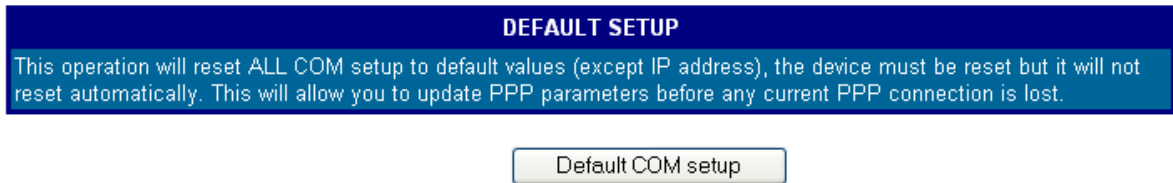


Figure 30: eWON Reset to default page

4.3.7 **STORAGE CONFIGURATION**

4.3.7.1 **Overview**

eWON stores configuration and recording data in its flash memory.

eWON flash memory is divided into areas of different sizes that can be erased and reformatted individually during a partitioning operation.

4.3.8 **The various types of information stored**

4.3.8.1 **Historical data**

4.3.8.1.1 **Overview**

A special file system is used for storing historical data. This file system is located in its own partition.

This partition contains a fixed number of 3 files that contain circular data:

- **Incremental recordings**
- **Events logging**
- **Alarm history logging**

Each of those files receives a predefined maximum amount of space.

When this amount of space is reached, then the older data are erased, in order to free space for the new data.

4.3.8.1.2 **Partitioning configuration**

The size allocated to the historical data partition can be configured by the user (from 1 MB to 3 MB).

In any case the Events logging and Alarm history logging size remains the same, only the Incremental recording size changes.

4.3.8.2 **Retentive values**

4.3.8.2.1 **Overview**

This is a fixed flash memory block that contains the retentive values.

The dimension of that block does not need to be modified.

4.3.8.2.2 **Partitioning configuration**

Retentive values use a fixed memory size.

4.3.8.3 Communication Configuration

4.3.8.3.1 Overview

The communication configuration needs to be saved in a distinct block in order to allow formatting any other data in the eWON without risking to loose contact with the device (Ethernet IP address, PPP configuration etc.).

This configuration uses a fixed memory size and is stored with a special mechanism that prevents losing configuration even if power is lost at any time during the configuration update.

The only risk is to lose the last modification made after last save occurred.

Communication Configuration consists in all the configuration information that appears in the ComCfg.txt file.

4.3.8.3.2 Partitioning configuration

Communication configuration uses a fixed memory size.

4.3.8.4 /usr partition

4.3.8.4.1 Overview

This partition uses a different file system, allowing to create a larger number of files and to use a larger total flash memory.

This file system is also very robust in case of power lost during the time when operations are performed on the files.

This partition can be used through the eWON's FTP server or using the eWON's BASIC scripts.

If the /sys partition does not exist, the program and the configuration are then stored in this partition (see below: /sys partition).

4.3.8.4.2 Partitioning configuration

The size allocated to the /usr partition can be configured by the user (from 1 MB to 3 MB).

4.3.8.5 /sys partition

4.3.8.5.1 Overview

This partition uses the same file system as the /usr partition described above.

Its function is to store the program and the configuration.

Configuration includes all the configuration information that appears in the config.txt file.

When this partition is formatted, the configuration and the program are erased, but the current configuration and program are still in memory.

If the eWON is rebooted at that moment, it will use after restarting a default configuration and an empty program.

When the configuration or the program is saved, the eWON uses this partition to save a config.sys and a program.sys files.

These files are used internally by the eWON and should not be modified by the user.

4.3.8.5.2 Partitioning configuration

The size allocated to the /sys partition can be configured by the user (0 or 1 MB).

If the partition size is set to 0 MB, then the partition is not created, and all /sys data are saved in the "/usr" partition.

4.3.8.6 Storage size options

The following options are available for the storage configuration (see below Configure chapter).

Mode ID	\sys size (MBytes)	\usr size (MBytes)	Incremental Recordings size (MBytes)	Nb IRC (number of points)	Nb Events	Nb Alarm
1-2-1	1	2	1	16384	762	8192
1-1-2	1	1	2	73728	762	8192
0-3-1	0	3	1	16384	762	8192
0-2-2	0	2	2	73728	762	8192
0-1-3	0	1	3	139264	762	8192

Table 29: storage size options list

4.3.8.7 Storage management from the eWON Web interface

There are three choices that can be accessed by following this way from the **Main Menu** navigation bar from eWON: **Configuration/System Setup/Storage**.

4.3.8.7.1 Configure

There are five different ways to configure the storage from the different files from eWON, depending on the way you are going to use it (refer to the configuration table below the following screenshot).

STORAGE CONFIGURATION						
Mem Org	/usr size (MB)	Recording size (MB)	Number of points (Hist. Rec.)	Number of events	Number of alarm hist.	/sys size (MB)
<input checked="" type="radio"/>	2	1	16384	762	8192	1
<input type="radio"/>	1	2	73728	762	8192	1
<input type="radio"/>	3	1	16384	762	8192	0
<input type="radio"/>	2	2	73728	762	8192	0
<input type="radio"/>	1	3	139264	762	8192	0

The table shows the possible memory organisations.
 If the memory organisation is changed, the new organisation will only be applied the **next time the eWON boots**.
 Changing the memory organisation result in a **complete format of the eWON**. Everything except the COM configuration will be erased.
 If the /sys partition size is NOT 0, the program and config files will be stored there.
 If the /sys partition size IS 0, the program and config files will be stored in the /usr partition
 The current configuration is displayed in **red**
Undo change by setting back memory organisation to current organisation before rebooting (in that case reboot is not required).

Apply change

Figure 31: Storage Configuration panel

Storage configuration	/usr	Recording	/sys	Typical use of your eWON
1	2	1	1	You want to store a lot of user files on your eWON, but you won't create a lot of Tags.
2	1	2	1	You mainly use your eWON to read values on PLCs or serial devices, so that you will create a lot of Tags, and you won't store a lot of user files on eWON.
3	3	1	0	You will store a lot of user files on your eWON, so that you decide to reserve the maximal amount of memory for it. You won't have to create a lot of Tags. The configuration files will be stored in the /usr folder.
4	2	2	0	You will store a lot of user files AND you will define a lot of Tags in your eWON. The configuration files will be stored in the /usr folder.
5	1	3	0	You won't store a lot of user files on your eWON, but you will define a lot of Tags on it. The configuration files will be stored in the /usr folder.

Table 30: storage configuration examples of use

IMPORTANT NOTE: The storage configuration you define in this panel will only be applied at next eWON reboot.

The current configuration is displayed in **RED**.

Any change in the configuration can be undone, providing that eWON has not been rebooted in the meantime.

Imagine you change your configuration (i.e. you select configuration 5 while you are currently running configuration 2... As long as you have not rebooted eWON, you can choose again configuration2. There will be no change applied in the Storage Configuration when eWON reboots. You have just to check that you select the line where characters display in red, and validate by clicking on the **Apply change** button.

4.3.8.7.2 Erase & Format

Erase & Format		
Format All partitions	<input type="checkbox"/>	Format /usr and /sys (if present)
Format /sys partition	<input type="checkbox"/>	
Format /usr partition	<input type="checkbox"/>	
Erase config files		
Erase config	<input type="checkbox"/>	All configuration except COM configuration (you will be unlogged)
Erase program	<input type="checkbox"/>	Basic program
Format Historical Recording File System		
Erase "Historical Recording" file	<input type="checkbox"/>	Erase ircall.bin
Erase "Events" file	<input type="checkbox"/>	Erase events.txt file
Erase "Alarms History" file	<input type="checkbox"/>	Erase hst_alm.txt file
Clear scheduled action		
Clear pending actions	<input type="checkbox"/>	Clear pending actions (except action currently 'in progress') from sstat.htm
Confirm		
Password required:	<input type="text"/>	Enter your password

If the partition containing the "config.sys" and "program.sys" file is erased and the eWON is reset before a config or program is changed the default config or program will be loaded.

Figure 32: Erase & Format panel

This control panel function is to allow you to erase some parts of the eWON storage area.

• Format All partitions

Control	Description
Format All partitions	Validating this checkbox will result in formatting /usr and /sys partitions.
Format /sys partition	Validating this checkbox will result in formatting /sys partition.
Format /usr partition	Validating this checkbox will result in: Formatting only /usr if /sys is not stored in it Formatting /usr AND /sys if you have chosen a storage configuration with /sys set to "0"

Table 31: Format All partitions controls

• Erase config files

Control	Description
Erase config	Validating this checkbox will result in erasing the eWON configuration, except for its communication information (comcfg.txt). Clicking on Execute after having selecting this checkbox will disconnect you from your current eWON session.
Erase program	Validating the checkbox will result in erasing the eWON Basic script file program.bas.

Table 32: Erase config files controls

• **Format Historical Recording File System**

Control	Description
Format Historical Recording File System	Validating this checkbox will result in erasing the 3 files that stores eWON internal history, which means all of the three following controls together in this table.
Erase "Historical Recording" file	Validating the checkbox will result in erasing the binary format ircall.bin file that contains the binary values of all the Tags that are defined in eWON.
Erase "Events" file	Validating the checkbox will result in erasing the text format events.txt file that contains the history of all of the (maximum) 762 last events that have been logged in eWON.
Erase "Alarms History" file	Validating the checkbox will result in erasing the text format hst_alm.txt file that contains the history of the alarms for the Tags that have been defined in the eWON.

Table 33: Format Historical Recording File System controls

• **Clear scheduled actions**

Control	Description
Clear pending actions	Validating this checkbox will result in erasing from the sstat.htm file the eWON actions that are not currently in progress, that means all the actions that are complete (whatever successful or not). Note: as sstat.htm is a "virtual" file, that means that its information are stored in the volatile memory from eWON, clearing this file does not impact on the memory file system from eWON. So that this command is an exception in this page, it has been placed here for obvious ergonomic reasons.

• **Confirm**

Control	Description
Password required	Your password is required in order to confirm your changes. Those changes will be validated when you click on the Execute button.

Table 34: Clear scheduled action control

4.3.8.8 RESET Request

eWON version 4 allows you to perform a reset from a second place inside of its web interface.

This reset page can be reached by following this way from the **Main Menu** navigation bar: **Configuration/System Setup/Storage/Reset** (from the **Storage** menu):

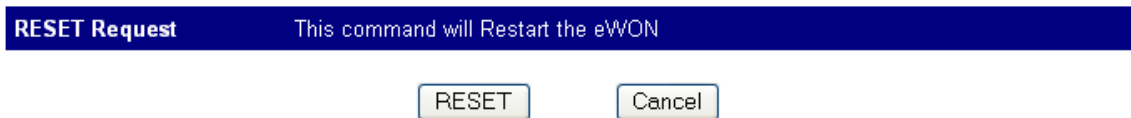


Figure 33: eWON RESET Request page from the Storage menu

If for any reason you want to restart eWON, then you have just to click on the **RESET** button, and the reset process will begin.

eWON will be accessible again when restart is complete.

The access to information on the different ways to reset eWON, please refer to chapter "Resetting eWON" on page 26.

4.4 Tag Setup

4.4.1 Tag definition: Introduction

The eWON SCADA features are based on:

- **The configurable monitoring of Tags**
- **The execution of scripts**

All the variables monitored by the eWON are defined as "Tags". A Tag is a Boolean or Float value changing with time and coming from a data-source.

Typical data-sources are:

- **eWON internal Inputs/Outputs**
- **Remote Modbus Input/Outputs**
- **eWON memory Inputs/Outputs (updated by script)**

The data source is called an "IO Server". An IO Server is the interface between the changing value and the eWON monitoring engine. It is a kind of driver. Any variable from any data source must have a common representation for all IO Servers in order to define common interface in the eWON.

The *data-source* representation in the eWON uses 3 fields for the definition of a Tag:

- **The IO Server Name**
- **The Topic name**
- **The Item Name**

A Tag's data-source will be uniquely identified with these 3 parameters.

IO Server name	Is a kind of driver name. For each IO Server there is a specific Topic Name and Item Name syntax. The following drivers are available: MODBUS, EWON, MEM, NETMPI, UNITE (Unitelway), DF1, FINS and S5-AS511.
Topic Name	Is used to group items inside an IO Server, for example the memory IO Server uses the blank topic ("") and the retentive topic ("ret"). All Tags of the MEM IO Server defined in the "ret" topic will have their value saved and restored when the eWON boots. All IO servers do not use Topic Name. In that case Topic Name must be empty.
Item Name	The item name is a string of characters; its syntax is specific to each IO Server. The Item Name describes the physical variable to monitor using the IO Server.

Table 35: Tag's data-source parameters

For example, the MODBUS IO Server needs to poll registers or coils from a slave, so it uses an item name representation to define the register type, register address and slave address.

(Example "40001,5" => Where 4 means read write register, 0001 is the register number and 5 is the slave Modbus address).

A description of the different IO Server syntax is given in Annex. Once a Tag is configured with its Server Name, Topic Name and Item Name, it is given a Tag name that will be used everywhere in the eWON.

4.4.1.1 Tags monitoring and handling

The eWON engine can handle the following operations on the Tags:

Operation	Description
Alarm monitoring	Check for low and high alarm levels or Boolean alarm level and management of alarm acknowledgement, alarm historical logging and action on alarm (Email, SMS, etc.)
Historical logging	Tags can be monitored and changes on a Tag can be saved in the Flash File System. Storage can be based on change threshold or at regular interval.
Real Time logging	A Tag history can be kept in memory for an amount of time with a given time interval. This operation is volatile and does not involve any flash storage.
Modbus TCP publishing	All Tags can be given a Modbus address and can be read using Modbus TCP protocol by and external modbus TCP master.
Tag grouping	Tags can be organized by page for easier handling and viewing.
Script Access	All Tags values and attributes can be read or changed from script.


Table 36: Tags operations handled by the eWON

4.4.2 Tag definition: Setup

The Tag Setup page, obtained by clicking on the **Tag Setup** item of the **Configuration** menu, allows building the eWON Tag's dictionary. The Tag name dictionary is the eWON central database where you have to define all Input / Output (internal – available on the eWON – or external – through extension bus) that you want the eWON to monitor. If your Input/Output is not defined by a Tag, you will not be able to address it.

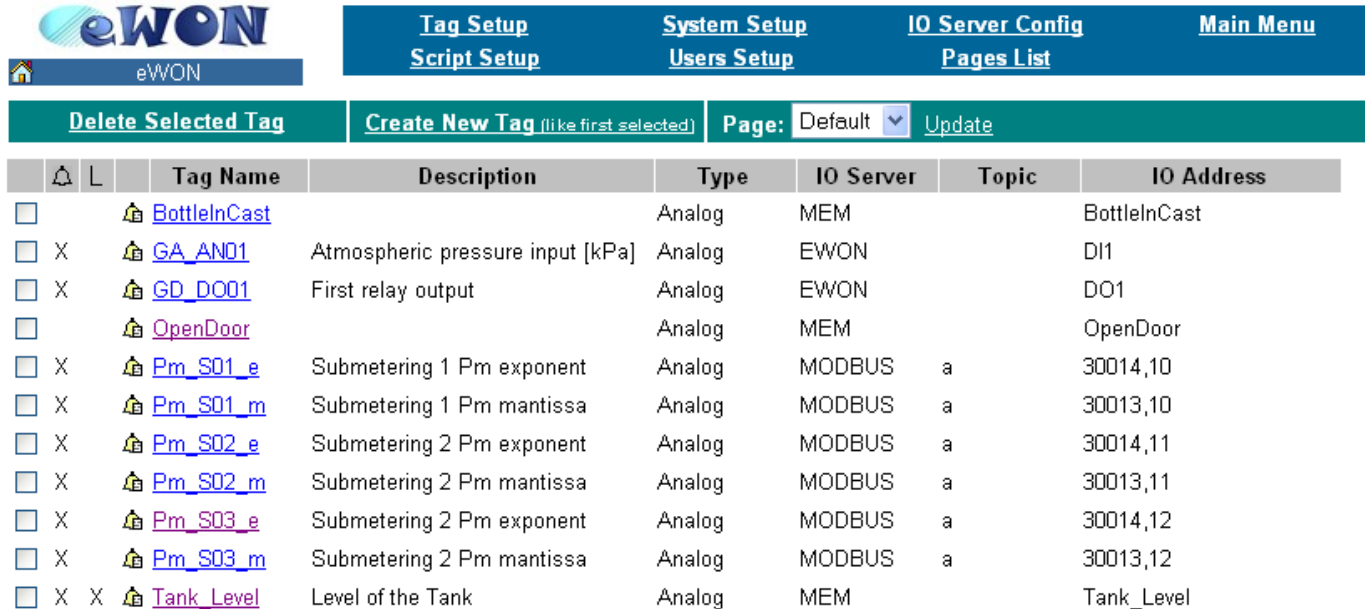
The Tag configuration includes two parts:

- **The complete Tag configuration except for the actions to execute in case of an alarm.**
- **The alarm action list.**

The first part of the configuration is accessed through the Tag hyperlink; the second part is accessed with the  icon next to the Tag hyperlink.


Note: In the header menu of the Tag edition, a combo box appears with a list of pages (please refer to chapter "Pages configuration" on page 68). Only the Tags of the selected page will be displayed in the list, except if "ALL" is selected.

Important: If a Tag is created in a page different than the page being displayed, it will not appear in the list, although it is present.



<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Tag Name	Description	Type	IO Server	Topic	IO Address
<input type="checkbox"/>			BottleInCast		Analog	MEM		BottleInCast
<input type="checkbox"/>	X		GA_AND1	Atmospheric pressure input [kPa]	Analog	EWON		DI1
<input type="checkbox"/>	X		GD_DO01	First relay output	Analog	EWON		DO1
<input type="checkbox"/>			OpenDoor		Analog	MEM		OpenDoor
<input type="checkbox"/>	X		Pm_S01_e	Submetering 1 Pm exponent	Analog	MODBUS	a	30014,10
<input type="checkbox"/>	X		Pm_S01_m	Submetering 1 Pm mantissa	Analog	MODBUS	a	30013,10
<input type="checkbox"/>	X		Pm_S02_e	Submetering 2 Pm exponent	Analog	MODBUS	a	30014,11
<input type="checkbox"/>	X		Pm_S02_m	Submetering 2 Pm mantissa	Analog	MODBUS	a	30013,11
<input type="checkbox"/>	X		Pm_S03_e	Submetering 2 Pm exponent	Analog	MODBUS	a	30014,12
<input type="checkbox"/>	X		Pm_S03_m	Submetering 2 Pm mantissa	Analog	MODBUS	a	30013,12
<input type="checkbox"/>	X	X	Tank_Level	Level of the Tank	Analog	MEM		Tank_Level

Figure 34: eWON Tag Setup page

- **To EDIT a Tag**
Click on the link in the *Tag name* column.
- **To EDIT a Tag ALARM ACTION**
Click on the icon  next to name of the Tag link.
- **To DELETE a Tag**
Click on the check box near the Tag name and then click on the *Delete Selected Tag* link.
- **To ADD a Tag**
Click on the *Create New Tag* link.

If you want to create a new Tag that has nearly the same property as another Tag in the list, then validate the checkbox that stands at the left from the "source" Tag's name and then click on the **Create New Tag** link. Doing that, all field properties of the new Tag will be automatically filled with the selected Tag's properties. Note: "the first selected" means that if several Tags are selected, then the Tag that will be cloned will be the first among the Tags that have been selected.

4.4.2.1 Tag main edit window

You will get access to the Tag main setup page by clicking on the **Create New Tag** link, or by editing an existing Tag when clicking on its name (the content of this page differs depending on your eWON's version: [Historical Data Logging](#) and [Real Time Logging](#) is not available on all the eWONs):

Tag Name:	<input type="text"/>	Page:	Default <input type="button" value="v"/>
Tag Description:	<input type="text"/>		
I/O Server Setup			
Server Name:	MEM <input type="button" value="v"/>	Topic Name:	<input type="text"/>
Address	<input type="text"/>	Type:	Analog <input type="button" value="v"/> Force Read Only: <input type="checkbox"/>
eWON value = IO Server Value * <input type="text" value="1"/> + <input type="text" value="0"/>			
Tag Visibility			
Published value	eWON value * <input type="text" value="1"/> + <input type="text" value="0"/> <small>REMARK: Value published is unsigned 16Bits for ModbusTCP and signed bits for SNMP</small>		
Modbus TCP	<input checked="" type="checkbox"/> Enabled		
Register	<input type="text" value="1"/>	<input type="checkbox"/> Consider as float register	
SNMP	<input checked="" type="checkbox"/> Enabled		
OID	<input type="text" value="1"/>	Value published: .1.3.6.1.4.1.8284.2.1.3.1.11.1.4. OID (Integer32)	
Instant Value			
<input type="checkbox"/> Group A <input type="checkbox"/> Group B <input type="checkbox"/> Group C <input type="checkbox"/> Group D			
Alarm Setup <input checked="" type="checkbox"/> Alarm Enabled			
Alarm Level Low:	<input type="text" value="0"/>	Alarm Level High:	<input type="text" value="0"/> Value Deadband: <input type="text" value="0"/>
Alarm Level LowLow :	<input type="text"/>	Alarm Level HighHigh :	<input type="text"/> Leave empty HiHi if unused and LoLo if unu
Boolean Alarm Level:	<input type="button" value="v"/> 0		
Activation Delay:	<input type="text" value="0"/> sec	Auto acknowledge:	<input type="checkbox"/> (Auto ACK on RTN)
Alarm Hint:	<input type="text"/>		
<input type="button" value="Add/Update Only"/> <input type="checkbox"/> Then Edit Notification <input type="button" value="Cancel"/>			

Figure 35: eWON Tag main configuration page - eWONs that does not allow logging feature

Tag Name:	<input type="text"/>	Page:	Default <input type="button" value="v"/>
Tag Description:	<input type="text"/>		
I/O Server Setup			
Server Name:	MEM <input type="button" value="v"/>	Topic Name:	<input type="text"/>
Address	<input type="text"/>	Type:	Analog <input type="button" value="v"/> Force Read Only: <input type="checkbox"/>
eWON value = IO Server Value * 1 <input type="text"/> + 0 <input type="text"/>			
Tag Visibility			
Published value	eWON value * 1 <input type="text"/> + 0 <input type="text"/> REMARK: value published is unsigned 16Bits for ModbusTCP and signed 32 bits for SNMP		
Modbus TCP	<input checked="" type="checkbox"/> Enabled		
Register	1 <input type="text"/>	<input type="checkbox"/> Consider as float register	
SNMP	<input checked="" type="checkbox"/> Enabled		
OID	1 <input type="text"/>	Value published: .1.3.6.1.4.1.8284.2.1.3.1.11.1.4.OID (Integer32)	
Instant Value			
<input type="checkbox"/> Group A <input type="checkbox"/> Group B <input type="checkbox"/> Group C <input type="checkbox"/> Group D			
Alarm Setup <input checked="" type="checkbox"/> Alarm Enabled			
Alarm Level Low:	0 <input type="text"/>	Alarm Level High:	0 <input type="text"/> Value Deadband: 0 <input type="text"/>
Alarm Level LowLow :	<input type="text"/>	Alarm Level HighHigh :	<input type="text"/> Leave empty HiHi if unused and LoLo if unused
Boolean Alarm Level:	0 <input type="button" value="v"/>		
Activation Delay:	0 <input type="text"/> sec	Auto acknowledge:	<input type="checkbox"/> (Auto ACK on RTN)
Alarm Hint:	<input type="text"/>		
Historical Logging <input checked="" type="checkbox"/> Historical Logging Enabled			
Logging Deadband:	-1 <input type="text"/> (put a negative value to disable deadband logging)		
Logging Interval:	0 <input type="text"/> Seconds (set to 0 will enable only Deadband logging)		
Real Time Logging <input checked="" type="checkbox"/> Real Time Logging Enabled			
Time Span:	600 <input type="text"/> Seconds		
Logging Interval:	10 <input type="text"/> Seconds		
<input type="button" value="Add/Update Only"/> <input type="checkbox"/> Then Edit Notification <input type="button" value="Cancel"/>			

Figure 36: eWON Tag main configuration page - eWONs that allow logging feature

Control	Description
Tag general properties	
Tag name	<p>Name of the Tag The name of the Tag will be used for any reference to the Tag when using export function or script function. -This Information will be included on the alarm Email Note: the Tag's name cannot contain: spaces \$ character " characters Maximum length from Tag's name is 64 characters</p>
Tag Description	<p>A free text to describe the meaning of the Tag. Useful to clarify the meaning from the alarm. -This Information will be included in the Email you can send on alarm.</p>

Table 37: Tag's general properties controls

I/O Server Setup	
Server Name	<p>The IO server name is the Data source of the Tag name. Six data sources are available: 'EWON' for all EWON internal IO (please refer to chapter "ABLOGIX IO Server" on page 104) 'MODBUS' for IO located on the extension bus (please refer to chapter "Modbus IO server" on page 77) 'MEM' for virtual IO used by script function (please refer to chapter "MEM IO Server" on page 113) 'NETMPI' for IO located on the extension bus (please refer to chapter "NETMPI IO Server" on page 84) 'UNITE' for IO located on the extension bus (please refer to chapter "UNITE IO Server" on page 86) 'DF1' for IO located on the extension bus (please refer to chapter "DF1 IO Server" on page 91) 'FINS' for IO located on the extension bus (please refer to chapter "FINS IO Server" on page 97) 'S5-AS511' for IO located on the extension bus (please refer to chapter "S5-AS511 IO Server" on page 101)</p>
Topic Name	<p>Used to apply a common configuration to several Tags. Please refer to chapter "Tag definition: Introduction" on page 56</p>
Address	<p>Please refer to chapter "Tag definition: Introduction" on page 56 Note: for memory Tag (MEM IO server) this field can be left empty. Although when editing a memory Tag, the Tag Name will be found here, this field is "don't care".</p>
Type	<p>Defines the Tag name type: Analog or Boolean. Analog enables the Tag to take float values coded on 4 bytes. Boolean only return value 0 or 1. If the IO server returns a value equal to 0, the Boolean value is 0. If the IO server returns a value different from 0 the output value is 1.</p>
Force Read Only	<p>Allows disabling the Update function in the <i>View IO</i> page. This is useful if you want to monitor a read/write Tag. The Tag is still read/write for BASIC operations.</p>
eWON value	<p>Defines the offset and scale factor to be applied to the IO value coming from the server. The offset and scales are float values and negative values are accepted. The Tag value will be: $TAGval = IOSEVERval * scale + offset$.</p>

Table 38: IO server configuration controls

Tag Visibility	
Published value	
Modbus TCP visibility	Each Tag in the eWON can be accessed by a modbus TCP master. If the Tag must be visible: Enable this checkbox.
Register	Address of the register, starting with 1. Only the register address has to be specified, the type of Tag (coil, contact, Input register or Holding register) is obtained from the Tag type (Analog or Boolean) and the Tag Read Only or Read/Write property (obtained from the IO server).
Consider as float	If this option is checked, then 2 consecutive 16 bits registers will be reserved and the value will be output as a 4 bytes IEEE float in those 2 registers (standard Modbus float representation). If the Tag is published as integer it may need to be scaled to fit the 16 bits modbus register. This operation will be applied to the Tag value to publish it.
SNMP Visibility	If this is checked, the Tag can be seen by the SNMP manager. The OID of the Tag can be defined. The base OID is already defined, the only parameter is the end of the OID.
Instant Value	Instant value means values of Tags at a given time. The Tag's instant values are stored in the inst_val file (available in txt and binary format from the Files Transfer link from the Main menu navigation bar from eWON). The 4 checkboxes that match the groups you want to choose will work accordingly with the \$ft Export Block Descriptor Tag that is described in chapter "\$dtIV [Instant Values]" on page 212 from this manual. You will find there detailed information about Instant Value too.

Table 39: Tag visibility controls

Alarm Setup	
'Alarm Enabled'	Check if you want to generate an alarm on the current Tag name.
Alarm level low	Low "warning" threshold value for alarm detection.
Alarm level high	High "warning" threshold value for alarm detection.
Alarm level lowlow	Low "danger" threshold value for alarm detection.
Alarm level hihi	High "danger" threshold value for alarm detection.
Value Dead band	The dead band is the difference between the alarm level and the RTN level (Return To Normal). i.e.: if alarm value is 20°C with a DeadBand=1, the alarm is triggered when the Temperature cross this 20°C boundary. On the other hand, the AlarmStatus will be RTN when the temperature pass below 19°C (20-1).
Boolean Alarm level	The alarm value ('0' or '1') of a Boolean Tag name –not applicable for analog Tag name-.
Activation delay	Time in seconds for which the Tag has to be out of threshold before declaring the Tag is in an alarm state. (This is mainly to avoid non significant alarms)
Auto acknowledge	If checked, the alarm will be automatically acknowledged when then alarm state goes to RTN value. Thus, the alarm is directly ended.
Alarm Hint	Information related to the alarm action - This Information will be included on the alarm Email-

Table 40: alarm setup configuration controls

• **Historical and real-time logging:**

Those fields display only for some eWON's versions (c.f. "Tags monitoring and handling" on page 57)

Historical and real-time logging	
Historical logging enabled	If checked, the Tag values will be logged. Warning: this is a non-volatile logging; data are stored in the flash file system. All the data are stored in the same file, the maximum number of values that can be saved is from 16384 to 139264, depending on the way you have setup the resources storage in the eWON (when maximum size is reached, then the older data will be erased first). If one Tag is changing very often and another Tag is changing rarely, the first value will obviously overwrite the second Tag's values because of the circular storage mechanism.
Logging dead band	Defines the dead band of the Tag's incremental recording (use negative value to disable it)
Logging Interval	Defines the interval, in seconds, for the Tag recording (set to zero to disable logging interval). Can be used at the same time as <i>logging dead band</i> .
Real time enabled	If checked, the Tag values will be logged in memory. Real time logging is different from historical logging because data are saved in a circular memory buffer. The other difference with historical time logging is that incremental recording is not possible, only fixed interval recording can be performed.
Time span	Defines total logging window time in seconds.
Logging Interval	Defines the interval, in seconds, of the Tag recording

Table 41: Historical and real-time logging controls

4.4.2.1.1 Publish as Modbus TCP

This feature allows accessing Tag Value through Modbus/TCP. In that configuration, eWON acts as a slave. Actually, there is no data map as in most PLC, instead of that you define Tag by Tag its Modbus TCP address.

4.4.2.1.1.1 Defining the Modbus TCP address

There are 4 types of Modbus variable, Contact (RO), Coil (RW), Input Register (RO), Holding register (RW). The type selected for the Tag you want to publish will depend on its type:

Tag property	Modbus data type
Boolean, RO	Contact
Boolean, RW	Coil
Analog, RO	Input register
Analog, RW	Holding register

Table 42: the 4 Modbus variable types

Usually, the RO or RW property of a Tag is obvious. But in case of doubt, you can confirm the type by checking the View IO for the Tag:

- **Remove the "Force Read Only" option in the Tag configuration (disabled by default).**
- **Check the View IO page, if the Tag has an update field, it means it is a RW Tag otherwise the Tag is RO.**
- **The address selected will be the address of the Modbus 16 bit register. The address range starts with register 1 (in the Modbus frame, eWON register 1 is transmitted as 0).**
- **Publish as float**

The eWON supports accessing Tag values as float registers. The float is published with the IEEE representation and the value can be read on 2 consecutive registers, with the first register starting at the user defined address.


4.4.2.1.1.2 Modbus TCP rules

- When accessing Modbus registers or coils that are not mapped to an eWON register, the returned value is 0.
- Maximum number of registers readable in 1 request: 25
- Maximum number of coils readable in 1 request: 2000

4.4.2.2 Tag "Alarm Action" edit window

Alarm action can be one of the following:

- Send an Email
- Send an SMS
- Send a file to an FTP server
- Send an SNMP Trap

The Alarm action window is accessed with the icon  next to the Tag's link.

Each of the four actions can be triggered with the following events:

- The alarm occurs (ALM) - low, lowlow, high or hihi (The alarm will also be triggered when changing the alarm level)
- Acknowledging (ACK)
- Return to normal level (RTN)
- End of Alarm (END)

Alarm Notification for tag: <u>Tank Level</u>	
Email upon	<input checked="" type="checkbox"/> ALM <input type="checkbox"/> ACK <input type="checkbox"/> RTN <input type="checkbox"/> END
Short Message:	<input type="checkbox"/> Format as short message
eMail TO:	ewon@actl.be <small>ex: usr1@dom.ci,usr2@dom.ci</small>
eMail CC:	
eMail Subject:	eWON COM config file
eMail Attachment(s):	&[\$dtSC\$ftH\$fncomecfg.htm] <small>ex: &[\$dtEV]</small>
SMS upon	<input checked="" type="checkbox"/> ALM <input type="checkbox"/> ACK <input type="checkbox"/> RTN <input type="checkbox"/> END
SMS Destination:	0423456789,ucp,0475161622,provider <small>ex: 0456334433,ucp,0567112200,pass DestPhone,Protocol,ServerPhone,pass with Protocol = ucp or tap</small>
SMS Subject:	Tank Level to be checked
Put FTP upon	<input type="checkbox"/> ALM <input type="checkbox"/> ACK <input type="checkbox"/> RTN <input type="checkbox"/> END
Destination File Name:	
File Content :	
SNMP Trap upon	<input type="checkbox"/> ALM <input type="checkbox"/> ACK <input type="checkbox"/> RTN <input type="checkbox"/> END
Trap Subject:	

Figure 37: Alarm notification setup

4.4.2.2.1 Email on alarm configuration

Configuration required if an Email must be sent in case of alarm. If Email must be sent through PPP, the system configuration (Main and COM) must also be setup.

Alarm action Properties	Description
Email upon	Checks the alarm states triggering an Email (ALM, ACK, RTN, END).
Short Message	In some case it is useful to have the whole message sent in the subject. For example if you need to route Email to SMS. Usually this checkbox is disabled.
Email To	List of TO Email address comma (, or ;) separated.
Email CC	List of CC Email address comma (, or ;) separated.
Email subject	Will be the subject of the Email (except if short message is selected).
Email attachment	Body text of the Email. This text can include Export Block Descriptor inline with text or as attachment. Attachments to include in the Email must follow the syntax: &[EXPORT_BLOC_DESCRIPTOR_1] &[EBD_2]... There can be as many attachments as required. EXPORT_BLOC_DESCRIPTOR syntax is described in chapter "Export Block Descriptor" on page 192 Example: &[\$dtRTGA_AN01\$ftG] &[\$dtEV\$ftT] Will export Real time data of GA_AN01 as a graph and the event log file as a text file.

Table 43: Email on alarm configuration controls

See also the SCRIPT function "SENDMAIL" on page 162.

4.4.2.2.2 SMS on alarm configuration

Warning: sending a SMS is possible only from an eWON that embeds a modem (c.f. Table on page 233)

Alarm action Properties	Description
SMS upon	Checks the alarm states triggering a SMS (ALM, ACK, RTN, END).
SMS Destination	A list of SMS_RECIPIENT; SMS_RECIPIENT;... See below for the SMS_RECIPIENT syntax.
SMS Subject	Will appear at the beginning of the SMS message.

Table 44: SMS on alarm configuration controls

See also the SCRIPT function called "SENDSMS" on page 163

4.4.2.2.2.1 SMS_RECIPIENT: syntax

The SMS_DESTINATION defines the phone number of the SMS recipient.

In order to reach a SMS recipient, a SMS server must be called and the correct protocol must be used with that server. Server phone number depends on the GSM operator and the protocol used will be one of the 2 standard UCP or TAP protocol. A table with the SMS protocols and server phone numbers is maintained on <http://www.ewon.biz/DocSMS.htm>.

To introduce timeout in the number composition, use '+' in the number instead of ","

SPECIAL CASE FOR FRANCE USERS: as ucp and tap server is not available in France, the InfoZ protocol is available for eWONs embedding an analog modem.

In that case the server phone number is 0. If a number like has to be dialed to access the network it can be entered before the 0.

Example: if 0 must be dialed to leave PABX the syntax is 0407886633,ifz,00

See also www.infoz.fr for details about the InfoZ service.

The syntax for SMS_RECIPIENT is:

- **DDDD,TTT,MMM,PPP**

Or

- **DDDD,TTT,MMM**

DDDD	Destination phone number	
TTT	Protocol type, must be one of the 4 following values:	
	ucp	It is possible to add a word datasize and parity specification. The generic syntax is ucpDP : with Datasize D= 8 or 7 with Parity P= n: none, o: odd, e: even Examples: ucp7o ucp7e ucp7n ucp8n (default value)
	tap	It is possible to add a word datasize and parity specification. The generic syntax is tapDP : with Datasize D= 8 or 7 with Parity P= n: none, o: odd, e: even Examples: tap7o tap7e tap7n tap8n (default value)
	gsm	
	ifz	
MMM	Server phone number, see your GSM provider or for example: http://www.woodstone.nu/salive/ PagerSettings.html If the previous field is GSM or IFZ, the server can be set to "0"	40 char max
PPP	If a password sometimes required by the GSM provider.	30 char max

Table 45: the SMS_recipient syntax

Examples:

SMS on alarm "SMS Destination" syntax	Explanations
0407886633,ucp,0475161622,proximus	ucp protocol requires the use of a password. In this case, the password is "proximus" See in the above table for the word datasize and parity specification
0407886633,tap,0475161621	tap protocol does not require to enter a password. See in the above table for the word datasize and parity specification
0407886633,gsm,0	Syntax for sending a SMS from an eWON with a GSM/GPRS modem Note: we advise you to exclusively use this syntax to send a SMS from an eWON that embeds a GSM modem (not tap or ucp protocols).
407886633,ifz,0	Syntax for sending a SMS towards a GSM modem (France)

Table 46: SMS destination syntax examples

All the above strings are valid. Concerning the three last examples in the list, the last "," is not mandatory as there is no password.

Important: Password is case sensitive.

When defining a phone number with a modem the "," is often used to insert a pause during the dialing operation. As the eWON uses the coma as a separator, the pause dial sign is replaced by a +.

For example the 0+0407886633 would dial a 0, then insert a 1 second pause, then dial 0407...

The gsm protocol attribute can be used in order to send the SMS directly through the GSM network, not using an SMS server. In this case, there is no server needed. This is valid only if eWON embeds a GSM or GPRS modem.

BENELUX only: SEMASCRIP

It is possible to send Semascript/ SemaDigit to a Semaphone inside of the Benelux area, by using the Belgacom server.

Example:

You want to send a semadigit to number 0498373101...

You must call the server at phone number 0458500001 (0+0458500001 if you must first dial a 0 on a PABX)

Keep only the last 7 digits of the semaphone destination number, and use TAP protocol with "7e1":

```
sendsms "8373101,tap7e1,0458500001","0498373101"
```

if eWON is behind a PABX and 0 must be dialed first use:

```
sendsms "8373101,tap7e1,0+0458500001","0498373101"
```

FRANCE only: ALPHAPAGE

As for SEMASCRIP, it is possible to send ALPHAPAGE messages to ALPHAPAGE pagers inside of the France area, by using the emessage server: the server is managed by a German firm called emessage, (<http://www.emessage.de/en/index.html>).

As for the SEMASCRIP users, the ALPHAPAGE users have chosen to keep on using a pager, with which the broadcasting coverage is better than with the GSM network (the users are sure to receive the message, that is not always the case with SMS).

The used protocol is TAP. The operating mode is the same as for SEMASCRIP:

You want to send an ALPHAPAGE message to number 0612345678...

You must call the server at phone number 0836601212 (0+0836601212 if you must first dial a 0 on a PABX)

Keep only the last 7 digits of the alphapage destination number, and use TAP protocol with "7e":

```
sendms "2345678,tap7e,0836601212"
```

if eWON is behind a PABX and 0 must be dialed first use:

```
sendms "2345678,tap7e,0+0836601212"
```

See also:

The SCRIPT function "SENDSMS" on page 163

The SCRIPT function "ONSMS" on page 153

4.4.2.2.3 FTP on alarm configuration

Alarm action Properties	Description
Put FTP upon	Checks the alarm states triggering the Put FTP (ALM, ACK, RTN, END)
Destination file name	Name of the file to create on the FTP server. The name can contain path specification.
File content	The file content can be static or dynamic (see below)

Table 47: alarm action properties

4.4.2.2.3.1 File content:

If a standard text is put in the File Content field, the file built will receive that static text as content. If the File content has the following form, one or more file(s) will be written with a dynamic content:

[EXPORT_BLOC_DESCRIPTOR_1][EBD_2]...

The number of EXPORT_BLOC_DESCRIPTOR is not limited. EXPORT_BLOC_DESCRIPTOR syntax is described in chapter "Export Block Descriptor" on page 192.

If the \$fn field is used with multiple Export blocks, the "Destination File Name" property MUST be empty.

See also:

The Basic function called "PUTFTP" on page 161.

4.4.2.2.4 SNMP Trap on alarm configuration

If a trap has to be sent on an alarm, the checkbox corresponding to triggering event has to be selected.

The edit box allows entering a specific text that will be displayed in the Trap event of the SNMP manager. The text string is limited to 256 chars.

All of the hosts that have been defined in the **SNMP Setup** menu (see chapter "SNMP Setup" on page 26) will receive the generated alarm Trap.

4.5 Pages configuration

Page definitions are used in eWON for two purposes:

- **Restrict user rights to specific directories in the user definable web site.**

See also chapter "Users Setup" on page 18.

- **Organize Tags in pages to ease viewing and restrict user access to specific Tags.**

See also chapter "Users Setup" on page 18.

The *Pages list* setup that is accessed from the configuration navigation bar from eWON looks as follows:

Pages list	
Default Page	Default
User Page 1	System
User Page 2	
User Page 3	
User Page 4	
User Page 5	
User Page 6	
User Page 7	
User Page 8	
User Page 9	
User Page 10	

If you change the name of a page, all users who had access to the old page will have access to the new page, if you clear the name of a page users who had access to that page will have access to the default page instead.

Figure 38: Pages list setup

You may enter up to 10 user defined pages.

When Tags are defined in a specific page and the name of the page is changed, then the same Tags will belong to the renamed page. In other words, the Tag actually belongs to a page number, **EXCEPT** if the name of the page is changed to <nothing> (empty field). In that case, all the Tags that belonged to the previous page will return in the default page. All the users who had access to that page only will have access to "All pages".

Any text can be entered for the page name, but if a page name is used for directory restriction, it must comply with directory syntax.

4.6 IO servers setup

Please refer to chapter "The eWON IO Servers" on page 75 for a complete description of IO server and their configuration.

4.7 Script Setup

The *Script setup* page you can reach by clicking on the **Script Setup** link from the configuration navigation bar is used to create, modify or erase the scripts in the eWON and to test them.

Basic is the language that is used to write these scripts.

Further information about the structure and the syntax of this language is given in chapter "Programming the eWON" on page 122.

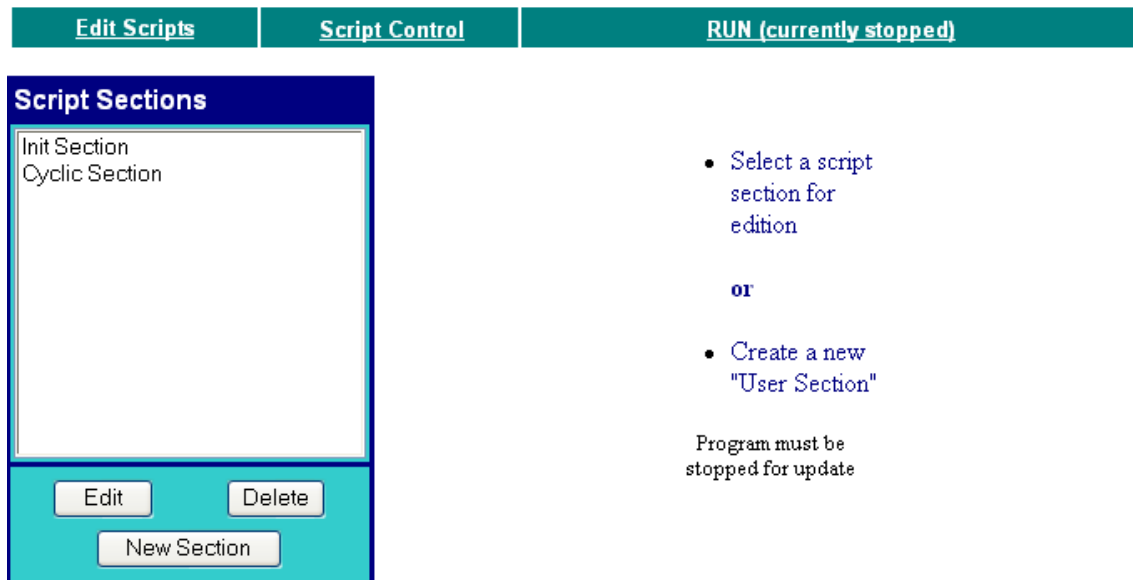


Figure 39: eWON script section page

The 3 available links in the script setup page are:

- The **Edit Scripts** link: to create, modify or delete the scripts
- The **Script Control** link: to test the scripts and enter one-shot commands
- The **Run/Stop** link: to run or stop the Basic scripts

4.7.1 The Edit script link

The 2 predefined scripts that are present in the eWON are:

- The **Init** section, which is executed once at the startup of the eWON.
- The **Cyclic** section, which is executed cyclically by the eWON.

This page is used to create, delete or edit script sections.

- The **cycle time of the eWON BASIC is not pre-determined!**
- **This cycle depends on script itself.**
- **If well programmed, you can achieve cycle time under one second.**
- **To edit a script section**

Select the script you want to display the code of from the **Script Sections** list, and then click on the **Edit** button, in order to display it in the edition window:

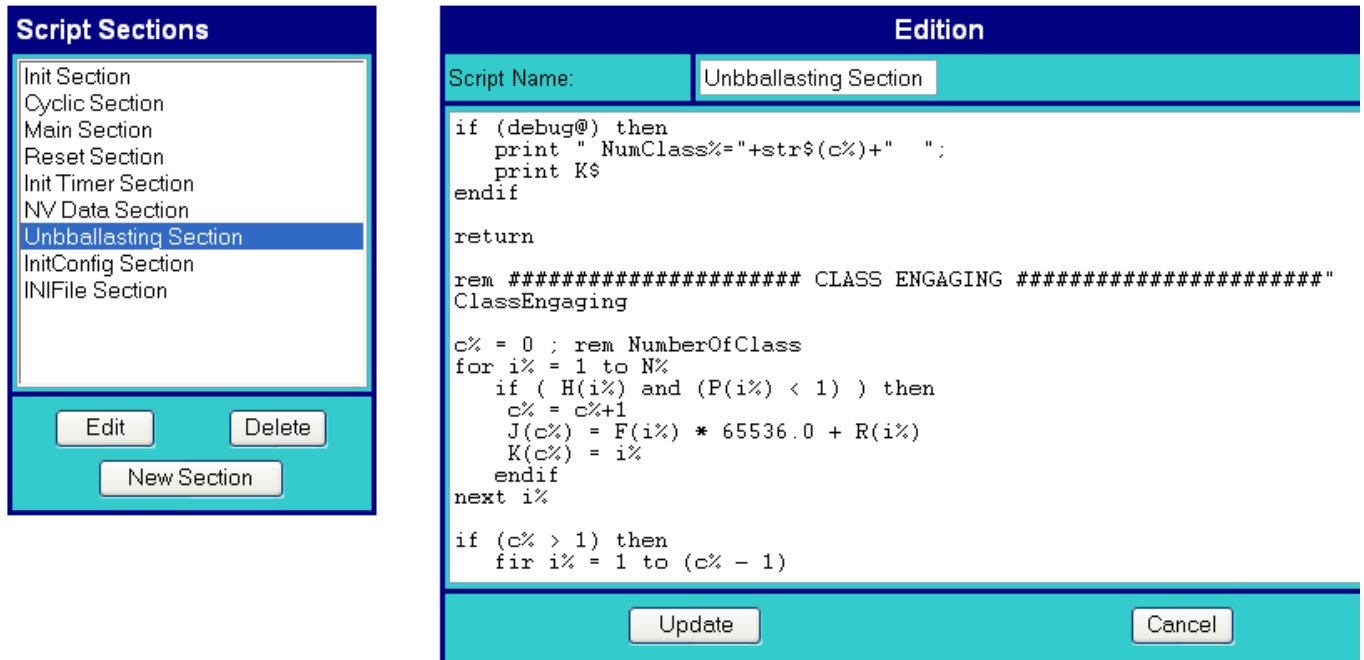


Figure 40: eWON Edit script section page

Click on the **Update** button to validate your modifications or the **Cancel** button to quit without saving.

- **To erase a script section:**
Select the script section to delete and click on the **Delete** button.
- **To create a new script section:**
Click on the **New Section** button.

Enter the name of the script in the **Script Name** field and the code to execute in the blank edition area.

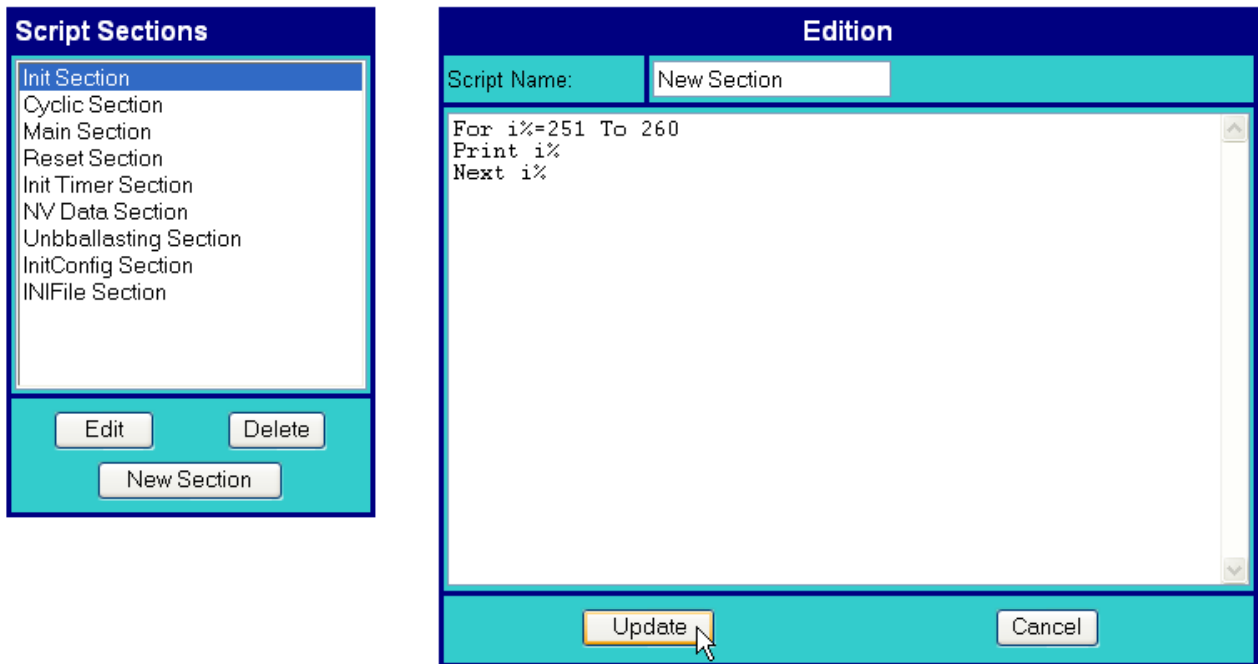


Figure 41: eWON new script section page

Click then on the **Update** button to validate your script, or on the **Cancel** button to quit without saving.

4.7.2 The Script control link

This page is used to monitor the result of the script execution and to execute some commands manually. You can use this page to easily debug your program.

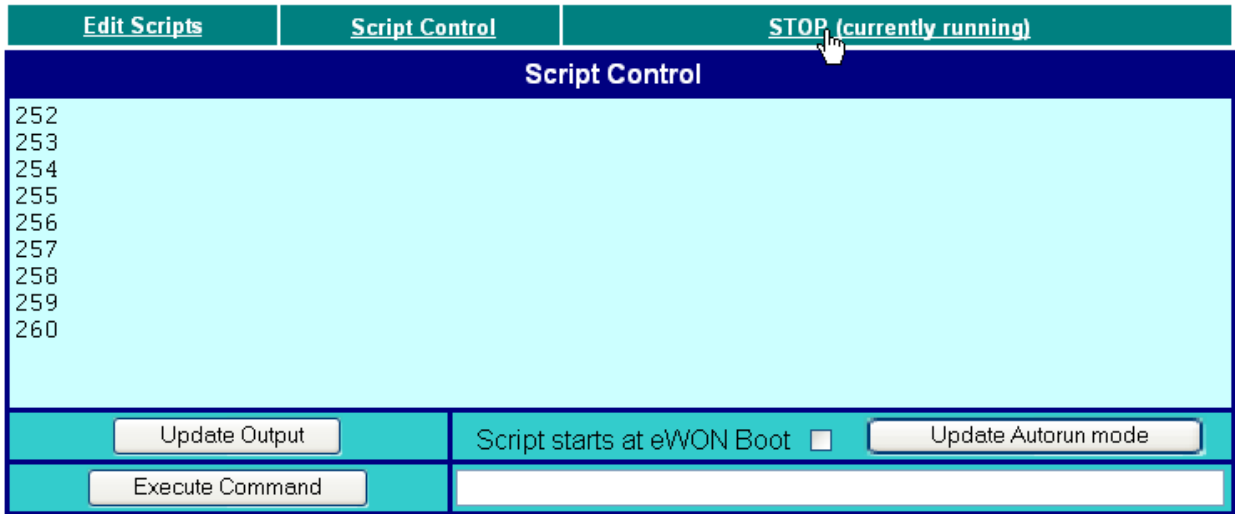


Figure 42: eWON script control page

To display the result of the script execution, 2 actions must be performed:

- Start the script execution by clicking on the **RUN** link.
- Click on the **Update Output** button to update the displaying of the result screen.

It's also possible to execute one-shot commands. Just fill in the blank field with the command to execute and then click on the **Execute Command** button.

Validate the **Script starts at eWON Boot** checkbox to start the script automatically when eWON boots. This change will be validated after you have clicked on the **Update Autorun mode** button.

4.7.3 The RUN/STOP link

This link is used to start or stop the script execution.

5 Configuring the eWON by a file upload

It is possible to configure eWON by uploading some files with a FTP client program.

If you need to configure the eWON, you will put the config.bin file or the two config.txt and comcfg.txt files on the root directory of the eWON.

You could also put the program.bas directly on the eWON. You can edit/modify the script Basic application in your favorite text editor, save as text file with the name program.bas and upload it in the eWON.

The files config.txt and comcfg.txt are interpreted by eWON. The eWON will use only the parameters that are known by it. In addition, you can send a config.txt file containing only the parameter you want to modify.

For instance, if the *Config.txt* file only contains the following lines, only the eWON identification will be changed.

```
:System
Identification:New_Identification
```

The *config.txt* file contains three sections: *System*, *TagList* and *UserList*. A section must only be declared if at least one field of that group is present in the file. A field must always appear after its section declaration.

A section is declared on a separated line, preceded by a column (See example above).

Each user and Tag appears on a separated line, with its field separated by a semi-column.

Example: the following config.txt file updates the eWON identification, defines a first user named "user1", a second user named "user2", and a Tag named "tag1". Notice that for user1, the CBMode is not specified and takes the value 0 (Mandatory), while for user2, CBMode takes the value 1 (User Defined).

```
:System
Identification:eWON
Information:
:TagList
"Id";"Name";"Description";"ServerName";"Address";"tagvalue"
1;"tag1";"first Tag";"MEM";"tag1";"47"
:UserList
"Id";"FirstName";"LastName";"Login";"Password";"Information";"CBMode"
1;"Jacques";"Dupond";"user1";"thePassword";"first user";
2;"Albert";"Deux";"user2";"thePassword2";"second user";"1"
```

When creating a Tag or a user, any field that is not specified will take the default value. Be aware of the fact that when creating a user, the full rights are allowed on the Tags by default (v.o.a.c).

The fields found in the *config.txt* and *comcfg.txt* file can also be used with the **GETSYS** and **SETSYS** functions explained in chapter "Programming the eWON" on page 122. A table describing all the fields can be found in chapter "Configuration Fields" on page 170.

6 The eWON IO Servers

6.1 Introduction

This introduction repeats some information already introduced in chapter "Tag definition: Introduction" on page 56.

An IO Server is the interface between a changing value and the eWON monitoring engine. It is a kind of driver. Any variable from any *data source* must have a common representation for all IO Servers in order to define common interface in the eWON.

The *data-source* representation in the eWON uses 3 fields for the definition of a Tag:

- The IO Server Name
- The Topic name
- The Item Name

A Tag's data-source will be uniquely identified with these 3 parameters:

IO Server name:	Is a kind of driver name. For each IO Server there is a specific Topic Name and Item Name syntax. Example: MODBUS, EWON, MEM
Topic Name:	Is used to group items inside an IO Server, for example the memory IO Server uses the blank topic ("") and the retentive topic ("ret"). All Tags of the MEM IO Server defined in the "ret" topic will have their value saved and restored when eWON boots. All IO servers do not use a Topic Name. In that case the Topic Name field must be left empty.
Item Name:	The item name is a string of characters; its syntax is specific to each IO Server. The Item Name describes the physical variable to monitor that uses the IO Server.

For example, the MODBUS IO Server needs to poll registers or coils from a slave, so it uses an item name representation to define the *register type*, *register address and slave address*. (Example "40001,5" => Where 4 means "read write register", 0001 is the register number and 5 is the slave Modbus address).

Table 48: Tags data-source parameters

6.2 IO servers setup

Some of the IO servers are configurable.

The IO setup window proposes a list of IO servers:



Figure 43: eWON IO servers scrolling list

Click on the **Edit** hyper link or select another IO server to display its edition window.

There are 3 possible cases regarding the IO server configuration:

- The IO server is not configurable
- The IO server has a dedicated configuration page (ex: MODBUS, UNITE, NETMPI, DF1, ...)
- The IO server uses the standard IO server configuration page.

The configuration of the Standard IO server is described in next chapter.

6.2.1 Standard IO server configuration page

When no dedicated configuration page is defined for configuring an IO server, the standard configuration page is used.

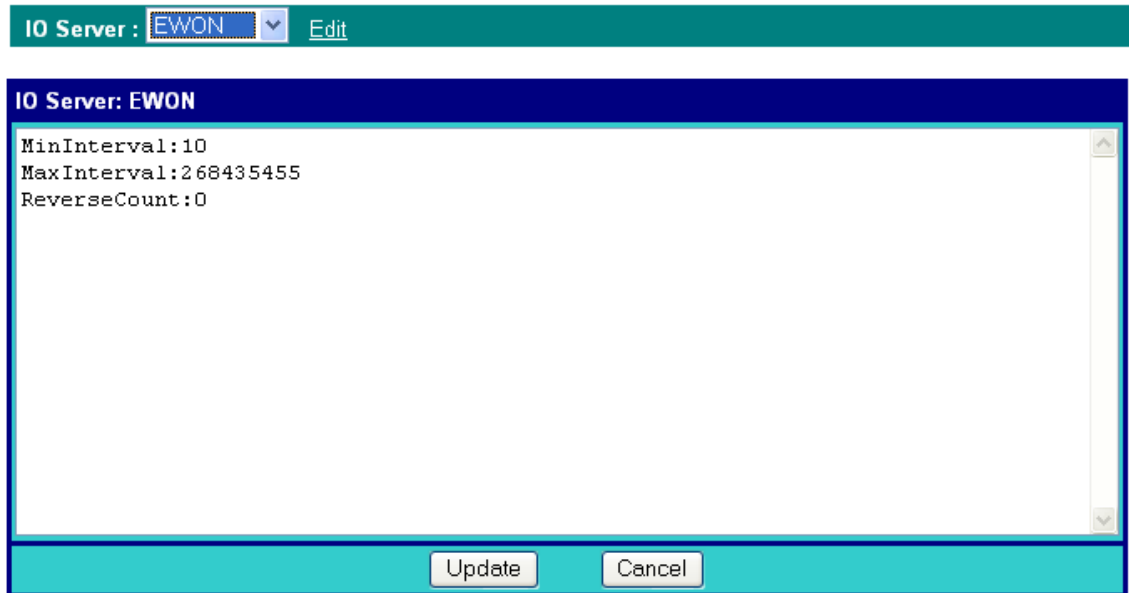


Figure 44: Standard IO server configuration page

As you can see in the example above, the standard configuration screen is a simple text edition area. Each parameter is entered on a separate line, the parameter value is separated from the parameter name by a ':'.

The generic format of a line is:

PARAM_NAME:PARAM_VALUE

Warning: Extra space must be removed.

When using this configuration, you must respect the correct syntax of each parameter and the values for each parameter.

The list of valid parameters, and there corresponding valid values is listed in the corresponding IO server documentation (see following chapters).

Any error that would occur when eWON applies the configuration you have entered would be written to the event file. Please refer to chapter "Files transfer" on page 118 to see how to get access to the events file.

6.3 Modbus IO server

6.3.1 Introduction

The MODBUS IO Server setup is the standard remote IO communication setup of the eWON. It is used to configure:

- eWON as a Modbus TCP slave
- eWON as a Modbus RTU master and as a Modbus TCP master.

The first feature (Modbus TCP slave) is specific to the MODBUS IO server; it is actually designed to provide access to eWON Tag values and, unlike all the other IO servers, for interfacing field values with the eWON.

The second feature (MODBUS Master) is the actual IO server feature that provides an interface to the field values as a common IO server.

The eWON MODBUS IO server will give access to values in equipments having a MODBUS interface.

The interface can be:

- RS485 – MODBUS RTU protocol will be used
- ETHERNET/PPP – MODBUS TCP protocol will be used.

eWON can mix access to MODBUS RTU and MODBUS TCP, depending on the way the Tag address is defined.

6.3.2 Setup

6.3.2.1 Setup for eWON Server

Figure 45: Modbus setup for eWON server configuration

This page defines the eWON configuration when used as a Modbus TCP slave.

As described in the Tag configuration paragraph, each Tag can be published to Modbus TCP so that a Modbus TCP can read their values.

This setup screen defines the eWON address, and globally enables or disables the Modbus TCP slave feature.

eWON Server Properties	Description
eWON server enabled	Globally enables or disables the Modbus TCP Server feature. If disabled, then any request from a Modbus TCP master will be rejected, even if Tags are published.
Modbus TCP Unit address	This feature is used by some gateway but can usually be left to 1 because Modbus TCP appears as a point to point connection.

Table 49: eWON server configuration - eWON as Modbus TCP slave

6.3.2.2 Setup for eWON IO server and Gateway - COM Setup

SETUP FOR eWON IO Server & Gateway (The eWON is Master of RS485 Modbus and ModbusTCP Gateway)	
COM Setup	
Baud Rate:	9600 ▼
Parity	None ▼
Stop Bit(s)	1 ▼
HW Mode	Full Duplex NO Handshaking ▼
Reply Timeout	1000 MS
Others:	8 data bits, RTU mode

Figure 46: Modbus communications configuration

If more than one Serial port are available, you must choose on which COM the modbus request will be sent.

This configuration part defines the RS485 setup. The four first fields are used to define the baud rate, parity, number of stop bits and the reply timeout (in msec – usually 1000 msec).

Warning: When there are multiple IO servers using potentially the serial line, then the unused IO server baudrate must be set to Disabled.

Example: if Modbus and UniTE IO servers are available, at least one of them must have its baud rate set to Disabled. If not the case, one of the IO servers will not be able to use the serial line and it will be disabled, with an error written in the event log.

6.3.2.3 Topic configuration

REM: Leave 'Global Slave Address' empty to define it tag by tag.

Topic A: <input checked="" type="checkbox"/> Enabled	
Topic Name	A
Global Slave Address	Slave Address (Unit Id): <input type="text"/> IP Address (Blank for RTU): <input type="text"/>
Poll Rate	2000 MS
Topic B: <input checked="" type="checkbox"/> Enabled	
Topic Name	B
Global Slave Address	Slave Address (Unit Id): <input type="text"/> IP Address (Blank for RTU): <input type="text"/>
Poll Rate	2000 MS
Topic C: <input type="checkbox"/> Enabled	
Topic Name	C
Global Slave Address	Slave Address (Unit Id): <input type="text"/> IP Address (Blank for RTU): <input type="text"/>
Poll Rate	2000 MS

Figure 47: Modbus topics configuration

Three topics can be used for the IO Server. These topics are used to give a common property to a group of MODBUS Tag like:

- **Enable/Disable**
- **Poll rate**
- **Slave address (Modbus RTU)**
- **Unit address and TCP/IP address (Modbus TCP)**

Modbus Server Properties	Description
Topic enabled	Enables or disables polling of all the Tags in the topic.
Slave address	This slave address is a global parameter for all the Tags of the topic. If the slave is connected with Modbus RTU, Slave address must be entered and IP address must be blank. If the slave is Modbus TCP, its unit address and its IP address must be entered.
Poll rate	This defines rate to which the Tag will be refreshed. In a complex application, we can imagine that some Tags must be refreshed every second – typically for digital input - and other every minute – typically: temperature-.

Table 50: Modbus topic configuration

Warning:

Any slave address that is defined in the Topic configuration overwrites the slave address configured per Tag.

If a Tag is defined with Tag Address: 40010,5 and the global address of the topic is 5 and 10.0.0.81, the Tag is entered as Modbus RTU but it is polled as Modbus TCP. So if you need to address slaves Tag by Tag, leave the topic address configuration empty.

6.3.2.4 Advanced parameters

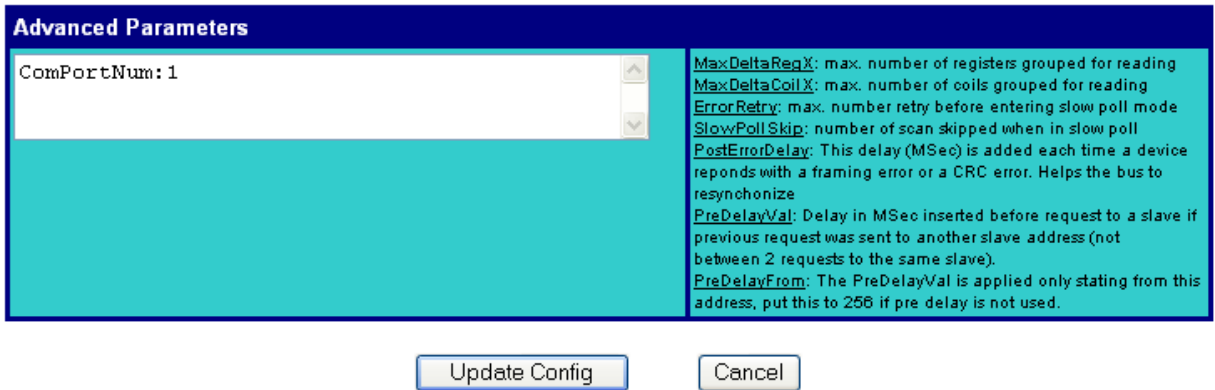


Figure 48: Modbus advanced parameters configuration

Extended parameters have been added to accommodate various special operating conditions. They are entered in the edit box at the bottom of the configuration page, conforming to the syntax below. Each parameter has a default value, so the advanced parameter edit box must only be filled with the parameters for which default values must be changed. (c.f. "Standard IO server configuration page" page 76).

Parameter name	Description
PreDelayFrom	Used in conjunction with the next parameter (PreDelayVal), starting at that slave address, the eWON will insert a delay when switching from one slave address to another. If the PreDelayVal feature is not used, then the value for PreDelayFrom must be set to 256.
PreDelayVal	Used to define the delay (in Msec) to be inserted before request to a slave if previous request was sent to another slave address (not between 2 requests to the same slave). The PreDelayVal is place only for slaves with an address higher than PreDelayFrom. (in Msec)
ErrorRetry	Defines the number of errors before the device enters in slow poll mode. (Minimum 1)
SlowPollSkip	Defines the number of times the slave is skipped when in slow poll mode. This delay depends on the poll rate.
GatewayIPCTimeout	Maximum event lock while waiting for response to modbus gateway request from the modbus IO server (router) (in Msec – minimum 1000).
PostErrorDelay	This delay is added each time a device responds with a framing error or a CRC error, in order to help the bus to perform its re synchronization. (in Msec)
MaxDeltaCoilX (X= A,B,C)	Maximum number of Coils that can be grouped in a request (per topic)
MaxDeltaRegX (X= A,B,C)	Maximum number of registers that can be grouped in a request (per topic)

Table 51: extended IO server configuration controls

6.3.2.4.1 Additional advanced parameters

Some parameters don't display in the Modbus IO server configuration window:

• **gwdestaddr**

When the eWON is used as a Modbus gateway, it uses the UnitID from the ModbusTCP request to determine the Modbus RTU destination address.

It is possible to bypass this mechanism and force all requests received by the eWON in ModbusTCP to be forwarded to a single ModbusRTU device (EXCEPT for requests with UnitID corresponding to the eWON's Modbus TCP Unit address

(usually 100) when the eWON Server is enabled - see explanations about this precise point in the above configuration fields descriptions).

Every requests are forwarded to the device with Slave address defined by the Modbus IO server advanced parameter called **gwdestaddr**.

If the advanced parameters textfield contains the following entry:

gwdestaddr:10

Then all requests will be forwarded to the slave with address 10.

REMARK: the eWON will change the address before sending the request to the slave, then it will change it back when sending the response to the master (Modbus TCP master). So the **gwdestaddr** will never appear in any communication between the Master and the eWON.

• **FastSrv**

FastSrv is a mode which allows a supervisor to read in Modbus TCP more easily the Modbus tags published by eWON.

This mode completely changes the tag's addressing, and when activated, the Modbus addresses are no more compatible.

You have just to enter "FastSrv:1" in the Advanced Parameters text area to activate it. When done, the Modbus tags can be read as follows:

X	Integer (with scale factors and offset defined)
X+2048	Float (X+2048: 1 ST float, X+2048+2: 2 nd float, etc.)
X+4096	Alarm status
X+6144	Alarm type

Notes:

- If the tag is binary read, its value is worth 0 if tag==0, and 1 if tag <>0
- Writing 0 in AlarmStatus acknowledges the alarm (will be logged by eWON as acknowledged by administrator)
- It is not possible to write a coil in the float area (coil address range: X+2048 to X+4094)
- It is not possible to address more than 1024 registers in float.

Click the **Update Config** button to validate your inputs or use the **Cancel** button to discard changes

6.3.3 Tag name convention

IO Server configuration		
IO Server Name	MODBUS	
Topic Name	A or B or C	
Item Name	ValueName,SlaveAddress	PLC Address is defined Tag by Tag on serial link (RTU Master)
	ValueName,SlaveAddress,IPAddress	PLC Address is defined Tag by Tag on TCP link
	ValueName	Topic PLC address is used

6.3.3.1 ValueName

The Modbus IO Server Tags can be classified following ranges of values. Two types of ranges are used. The two following tables describe the different ranges of value, for each of the two standards.

• **First standard:**

Modbus Type	IO Type	Access	Register address
Coil	Digital Output	R/W	1 → 9999
Contact	Digital Input	R	10001 → 19999
Input Register	Analog Input	R	30001 → 39999
Holding Register	Analog Output	R/W	40001 → 49999
Output Coil*	Digital Output	W	50001 → 59999
Output Registers*	Digital Input	W	60001 → 69999

Table 52: Modbus IO server Tag name convention: first standard

• **Second standard:**

Modbus Type	IO Type	Access	Register address
Coil	Digital Output	R/W	+1 → +65535
Contact	Digital Input	R	+100001 → +165535
Input Register	Analog Input	R	+300001 → +365535
Holding Register	Analog Output	R/W	+400001 → +465535
Output Coil*	Digital Output	W	+500001 → +565535
Output Registers*	Digital Input	W	+600001 → +665535

Table 53: Modbus IO server Tag name convention: second standard

The second standard allows more than 9999 values in each range. Notice the "+" in before the register value.

* The two last ranges are used with non-standard equipments that do not allow reading of some of their values. In this case, specifying the address in the "write only" ranges informs the eWON not to read the values after setting them, which is normally done in the other cases. If those registers are read, the returned value will always be 0.

After the numerical value, the characters F, L, I, D or W can be used to specify how to read the value. The following table describes the different character meaning.

Character	Description
W	Reads 1 register considered as 16 bits unsigned integer (DEFAULT IF NOT SPECIFIED)
I	Reads 1 register considered as 16 bits signed integer
D	Reads 2 regs R1, R2 as a DWORD R1 is Less significant, R2 is most significant (32 bits, unsigned)
E	Reads 2 regs R1, R2 as a DWORD R2 is Less significant, R1 is most significant (32 bits, unsigned)
L	Reads 2 regs R1, R2 as a LONG R1 is Less significant, R2 is most significant (32 bits, signed)
M	Reads 2 regs R1, R2 as a LONG R2 is Less significant, R1 is most significant (32 bits, signed)
F	Reads 2 regs R1, R2 as a FLOAT R1 is Less significant, R2 is most significant (32 bits, signed)
H	Reads 2 regs R1, R2 as a FLOAT R2 is Less significant, R1 is most significant (32 bits, signed)

Table 54: the characters used to specify how to read the value

When reading a 32 bits value, two consecutive registers or coils are read and combined, i.e. 40001F,11 to access in float representation the reg 1 on the slave 11.

Reading high 32 bits values involve a lost of precision in the mantissa because internally all values are considered as float by the eWON.

STATUS TAG:

The STATUS Tag is a special Tag that returns information about the current state of the communication for a given device. As for other Tags, the status Tag ItemName is composed of:

Status,Address

If address is omitted, the Topic address will be used. i.e.: status,11 points to the status of the slave 11

You can define a status Tag for each address used.

If you use the status MODBUS address, the Tag must be configured as analog:

Tag value	Meaning
0	Communication not initialized. Status UNKNOWN. If no Tag is polled on that device address, the communication status is unknown.
1	Communication OK.
2	Communication NOT OK.

Table 55: MODBUS status values

6.3.3.2 Slave Address

This is the address of the slave device you want to access.

It is a number from 0 to 255.

Example:

30001,11	Polls a RTU device at address 11.
-----------------	-----------------------------------

6.3.3.3 IP Address

This is the IP address of the device on an Ethernet network. It is composed of 4 numbers separated by a dot.

Example:

30001,11,10.0.0.50	Polls a device configured with IP address 10.0.0.50 and with Modbus slave address 11.
---------------------------	---

6.3.3.4 Device specific information

Warning for new users of WAGO modules:

Keep in mind that coil read and write don't use the same address (offset of 0x200); please consult the Wago™ documentation.

Example:

If you use Wago™ systems with two digital inputs and two digital outputs, inputs have addresses 1 and 2 and outputs have the same. The only way to distinct them it's the read-only access or R/W access.

Tags: station 11

Tag	Modbus address	Comment
MB_DigIn1	10001,11	Digital input module 1 -- read-only
MB_DigIn2	10002,11	Digital input module 2 -- read-only
MB_DigOut1	00001,11	Digital output module 1 for writing - Encode all leading zeroes!
MB_DigOut1Read	10513,11	Digital output module 1 for reading only
MB_DigOut2	00002,11	Digital output module 2 for writing Encode all leading zeroes!
MB_DigOut2Read	10514,11	Digital output module 2 for reading only

Table 56: Wago™ modules - addresses examples

In *View I/O page*, you can change the value of MB_DigOut1 with the update link (set to 1), and if you do that, you view that the value read is always 0.

Why?

Because the eWON reads the value at the WAGO address 1 (thus, DigIn1)! If you want to read the state of the DigOut1, you must read it at WAGO address 513!

The same remark is applied for analog Modbus registers. It's the documented behavior of Wago™-Modbus modules; keep it in mind.

6.4 NETMPI IO Server

6.4.1 Introduction

NETMPI IO server is used to communicate with SIEMENS PLC. The eWON will connect to the PLC's MPI interface by mean of the eLINK adapter.

The eLINK adapter is an interface with an MPI interface on one side and an Ethernet TCP/IP interface on the other side. The eLINK interface will be configured with an IP that the eWON will use to poll data. In addition to the IP address, the eWON will need the MPI address of the PLC on the MPI bus.

Using that eLINK interface, it is possible to poll different types of item in the PLC. These items types are described below.

6.4.2 Setup

IO Server : **NETMPI**

NETMPI CONFIGURATION

This IO server configures interface with eLINK device

REM: Leave 'Global Device Address' empty to define it tag by tag.

Topic A: Enabled

Topic Name	A
Global Device Address	<input type="text"/> (MPI_Addr,eLINK_IP_Addr)
Poll Rate	<input type="text"/> MS (default: 2000)

Topic B: Enabled

Topic Name	B
Global Device Address	<input type="text"/> (MPI_Addr,eLINK_IP_Addr)
Poll Rate	<input type="text"/> MS (default: 2000)

Topic C: Enabled

Topic Name	C
Global Device Address	<input type="text"/> (MPI_Addr,eLINK_IP_Addr)
Poll Rate	<input type="text"/> MS (default: 2000)

Figure 49: NETMPI configuration window

As for the Modbus and Unite protocols, it is possible to define topics, so that a single configuration can be applied to several Tags.

6.4.3 Tag name convention

IO Server configuration		
IO Server Name	NETMPI	
Topic Name	A	
	B	
	C	
Item Name	ValueName.RemoteStationAddress.eLinkIpAddress	PLC Address is defined Tag by Tag
	ValueName	Topic PLC Address is used

Table 57: NETMPI - IO server configuration

The Item Name can contain the PLC address where the value is polled, or not. If address is not specified at the Tag level, the global topic address will be used. If Topic address is not defined either, the Tag will be disabled.

6.4.3.1 ValueName

DBxBy	Data block x Byte offset y (0-255)
DBxWy	Data block x Word offset y (0 - 65535)
DBxFy	Data block x Float offset y (+-3.4e38)
DBxDy	Data block x DWord offset y (-2147483648->2147483647)
MxB	Memento x as byte
MxW	Memento x as word
MxF	Memento x as real
MxD	Memento x as double
Tx	Timer x (0-65535)
Cx	Counter x (0-65535)

Table 58: value names for NETMPI addresses

• Bit access modifier:

In any of the items above, it is possible to access a single bit (except for float items).

#x must be appended to the Tag name. (Bit index goes from 0..31)

The syntax can be used for reading bits and for writing them as well.

Example:

DB1W13#3 represents bit 3 of W13 in DB 1

REM:

Bit index range depends on item referenced (for W, bit index maximum value is 15)

6.4.3.2 Device Address

The device address is either appended to the ValueName in the Item Name definition, or entered in the Topic global address fields.

The device address is composed of: RemoteStationAddress,eLinkIpAddress:

• **RemoteStationAddress:** is the PLC's MPI address

• **eLinkIpAddress:** is the eLINK IP address.

Important:

If the PLC address is defined at the Topic level, it can be omitted in the Tag definition. In that case the Tag name will only contain the "ValueName".

If the PLC address is specified at the Topic level, it will replace any address defined Tag by Tag.

6.5 UNITE IO Server

6.5.1 Introduction

The eWON RS485 link can be configured as an UNITELWAY SLAVE INTERFACE. When the BaudRate in the UNITE IO Server is set to a value different than "Disabled", the Unitelway slave module in the eWON will be enabled.

This Unitelway slave provides 2 features:

- **Poll items in a Unitelway capable device.**

The device can be the Unitelway master itself or a device addressable through the Unitelway master on the XWay network.

- **Forward XIP requests from TCP/IP XIP to Unitelway bus and thus, act as a gateway between XIP and Unitelway.**

Using that feature, it is for example possible to access a PLC connected to the eWON's Unitelway link by connecting PL7PRO using the XIP driver started with the eWON IP address as destination.

6.5.2 Setup

6.5.2.1 Communication Setup

IO Server : UNITE

XWAY - UNITELWAY CONFIGURATION	
eWON is acting as a XIP to Unitelway gateway and Unitelway IO slave	
COM Setup	
Baud Rate:	Disabled <input type="button" value="v"/> (default: 9600)
Parity	Odd <input type="button" value="v"/> (default: ODD)
Stop Bit(s)	1 <input type="button" value="v"/> (default: 1)
HW Mode	Half Duplex <input type="button" value="v"/> (default: Half duplex)
Master response timeout	<input type="text"/> MS (20..60000, default: 1000)
Rx message timeout	<input type="text"/> MS (1000..60000, default: 3000)
Tx message timeout	<input type="text"/> MS (1000..60000, default: 3000)
Force Unitelway V2	<input type="checkbox"/> (if not checked eWON will talk in V1)
Disable 0,254 translation	<input type="checkbox"/> (if not checked, the gateway uses the request's Network & Station destination address)
ADD	<input type="text"/> Link address of eWON on unitelway bus (default: 4)
XWAY Network.Station	<input type="text"/> . <input type="text"/> (0..127).(0..63)

Figure 50: XWAY-UNITELWAY configuration

If more than one Serial port are available, you must choose on which COM the modbus request will be sent.

The following parameters can be modified:

Baud Rate	Select the baud rate applying to your industrial network
Parity	The parity to apply: none / odd / even. This field is set by default to Odd, as in the main cases in a typical UniTE topology. However, eWON allows you to define a different parity type (Even or None), in case this is needful to comply with your industrial network installations.
Stop Bit(s)	Number of stop bits
Master Response Timeout	Maximum time the eWON will wait for a valid message from the Unitelway master. This value can be critical for a correct operation, depending on the responsiveness of the master. A value of 1000 should be selected to guarantee correct operation.
Rx message timeout (MSEC)	Maximum time between a request is posted and the response is received
Tx message timeout (MSEC)	Maximum time for a request to be sent
Force UnitelWay V2 (MSEC)	If checked, the eWON will initiate communication in V2 with the devices. When used with a TSX PLC, this check box can be left unchecked.
ADO	Link address base. EWON will respond to AD0 and AD0+1 on the Unitelway link. The eWON will act as an Unitelway slave, it will respond to 2 consecutive link addresses AD0 and AD0+1, doing this improves the throughput of data across the eWON when acting as a gateway.
Xway Network Station	Address of the eWON on the XWAY network. When acting as an XIP to Unitelway gateway, the eWON will only respond the XWay network station defined here. Any XIP frame addressed to another network station will be ignored.

Table 59: XWAY communication setup controls

Important: When there are multiple IO servers potentially using the serial line, the unused IO server must select "DISABLED" for the unused IO server baudrate.

Example: if Modbus and UniTE IO servers are available, at least one of them must have the baudrate configured to "Disabled". If not, one of the IO servers will not be able to use the serial line and it will be disabled, with an error written in the event log.

6.5.2.2 Topic configuration

REM: Leave 'Global Device Address' empty to define it tag by tag.

Topic A: <input type="checkbox"/> Enabled	
Topic Name	A
Global Device Address	<input type="text"/> (Network,Station,Gate[,Module,Channel]) 0,254,0 is default
Poll Rate	<input type="text"/> MS (default: 2000)

Topic B: <input type="checkbox"/> Enabled	
Topic Name	B
Global Device Address	<input type="text"/> (Network,Station,Gate[,Module,Channel]) 0,254,0 is default
Poll Rate	<input type="text"/> MS (default: 2000)

Topic C: <input type="checkbox"/> Enabled	
Topic Name	C
Global Device Address	<input type="text"/> (Network,Station,Gate[,Module,Channel]) 0,254,0 is default
Poll Rate	<input type="text"/> MS (default: 2000)

Figure 51: XWAY-UNITELWAY topics configuration

Three topics can be used for the IO Server. These topics are used to give a common property to a group of UNITE Tags such as:

- Enable/Disable
- Poll rate
- Device address

Topic configuration item	Description
Topic enabled	Enables or disable polling of al the Tags in the topic.
Device Address	This device address is a global parameter for all the Tags of the topic. See below for the Device Address syntax. If an address is specified here, it will replace (overload) the address defined Tag by Tag.
Poll rate	Defines the refresh rate of the Tag name. In a complex application, we can imagine that some Tag name must be refreshed every second – typically for digital input - and other every minute – typically: temperature-.

Table 60: UNITE - topics configuration

6.5.3 Tag name convention

IO Server configuration		
IO Server Name	UNITE	
Topic Name	A	
	B	
	C	
Item Name	ValueName,Network,Station,Gate,Module,Channel	PLC Address is defined Tag by Tag
	ValueName,Network,Station,Gate	PLC Address is defined Tag by Tag (and the gate requires a 5 level addressing)
	ValueName	Topic PLC Address is used

Table 61: UNITE - IO server configuration

The Item Name can contain the PLC address where the value is polled, or not. If address is not specified at the Tag level, the global topic address will be used. If Topic address is not defined either, then address 0,254,0 will be used.

6.5.3.1 Value Name

Value name follows the syntax below:

MWxW	Internal data word 16 bits (unsigned)
MWxl	Internal data word 16 bits (signed)
MWxD	Internal data word 32 bits as DWORD (unsigned)
MWxF	Internal data word 32 bits as IEEE float
MWxL	Internal data word 32 bits as LONG (signed)
SWxW	System data word 16 bits (unsigned)
SWxl	System data word 16 bits (signed)
SWxD	System data word 32 bits as DWORD (unsigned)
SWxL	System data word 32 bits as LONG (signed)
Mx	Internal data bit
Sx	System data bit

Table 62: value names for UNITE addresses

Notes:

SW type cannot be formatted as float

eWON allows you to optimize the requests in case you need to read a lot of Tags that have been created on the UniTelWay device. Imagine you have 100 Tags to read, eWON will group the Tags within a predefined limit in order to make the less as reading operations as possible. The number of Tags that can be read depends of the types of words or bits that have to be read:

SW and MW types: by groups of 50

S and M types: by groups of 200

It is possible to read one bit from a word. The syntax to add is as follows:

#0 to #31

That means, if you want to read the fourth bit from an internal data word 16 bits unsigned that you address MW0, you have to add "#4" at the end of the address: MW0#4.

The type of words for which this syntax can be applied are:

MWxW, MWxI, MWxD, MWxL, SWxW, SWxI, SWxD and SWxL (please report to the table above).

6.5.3.2 The device address syntax

The Device Address is used in the topic definition or in the Tag definition. If used in the Tag definition, it will be separated from the value name by a coma (',')

- **Network,Station,Gate**

or

- **Network,Station,Gate,Module,Channel**

The second case applies to addresses with 5 levels:

- **Network: 0..127**
- **Station: 0..63**
- **Gate:**
- **Module:**
- **Channel:**

Module and channel can be omitted if not required.

If address is not specified, 0,254,0 will be used.

Important: If an address is specified in a Topic definition it will replace any address defined Tag by Tag.

6.6 DF1 IO Server

6.6.1 Introduction

The eWON serial link can be configured as a DF1 INTERFACE. When the Baud Rate in the DF1 IO Server is set to a value different than "Disabled", the DF1 module in the eWON will be enabled.

This DF1 module provides 2 features:

- Poll items in SLC controllers using PCCC requests.

- Forward PCCC requests from EIP (TCP/IP) to DF1 bus and thus, act as an adapter between EIP and DF1.

Using that feature, it is for example possible to access a PLC connected to the eWON's DF1 link by connecting RSLogix 500 using RSLinx TCP driver started with the eWON IP address as destination.

The DF1 IO Server can be configured in 2 modes:

- Full Duplex mode (eWON serial link must be configured in RS232 mode)
- Half Duplex slave mode

6.6.2 Setup

6.6.2.1 Communication Setup

IO Server : DF1 Edit

DF1 CONFIGURATION

eWON is acting as a EIP to DF1 adapter and DF1 IO slave

COM Setup

Baud Rate:	Disabled (default: 9600)
Parity	None (default: NO)
Stop Bit(s)	1 (default: 1)
Frame Error Detection	CRC (default: CRC)
HW Mode	Half Duplex (default: Full Duplex)
Master response timeout	<input type="text"/> MS (20..60000, default: 1000)
Rx message timeout	<input type="text"/> MS (1000..60000, default: 3000)
Tx message timeout	<input type="text"/> MS (1000..60000, default: 3000)
eWON DF1 Address	<input type="text"/> Device address of eWON on DF1 link (0..254, default: 4)
Destination DF1 Address	<input type="text"/> Device address of destination on DF1 link when EIP is used (0..254, default: 1)

Figure 52: DF1 communications setup

If more than one Serial port are available, you must choose on which COM the modbus request will be sent.

The following parameters can be modified:

Baud Rate	
Parity	The parity to apply: none / odd / even
Stop Bit(s)	Number of stop bits
Frame Error Detection	Cyclic Redundancy Check (CRC) or Block Check Character (BCC)
HW mode	Full Duplex no handshaking or Half duplex
Master Response Timeout	Maximum time the eWON will wait for a valid message from the DF1 master. This value can be critical for a correct operation, depending on the responsiveness of the master. A value of 1000 should be selected to guarantee correct operation
Rx message timeout (MSEC)	Maximum time between a request is posted and the response is received
Tx message timeout (MSEC)	Maximum time for a request to be sent
eWON DF1 address	Device Address of eWON on DF1 link When eWON will act as a DF1 slave, it will respond to 2 consecutive link addresses; doing this improves the throughput of data across the eWON when acting as a gateway. (eWON DF1 address and eWON DF1 address +1)
Destination DF1 address	Device Address of Destination on DF1 link when EIP is used

Table 63: DF1 communication setup controls

6.6.2.2 Topic configuration

REM: Leave 'Destination Device Address' empty to define it tag by tag.

Topic A: <input checked="" type="checkbox"/> Enabled	
Topic Name	A
Destination Device Type and Address	<input type="text"/> SLC500-Device Address (0..254)
Poll Rate	<input type="text"/> MS (default: 2000)
Topic B: <input checked="" type="checkbox"/> Enabled	
Topic Name	B
Destination Device Type and Address	<input type="text"/> SLC500-Device Address (0..254)
Poll Rate	<input type="text"/> MS (default: 2000)
Topic C: <input checked="" type="checkbox"/> Enabled	
Topic Name	C
Destination Device Type and Address	<input type="text"/> SLC500-Device Address (0..254)
Poll Rate	<input type="text"/> MS (default: 2000)

Figure 53: DF1 topics configuration

Three topics can be used for the IO Server. These topics are used to give a common property to a group of DF1 Tags like:

- **Enable/Disable**
- **Poll rate**
- **Destination Device Type and Address**

Topic configuration item	Description
Topic enabled	Enables or disables polling of all the Tags in the topic
Destination Device Type and Address	The Destination Device Type and Address is a global parameter for all the Tags of the topic. See below for the Device Address syntax. If an address is specified here, it will replace (overload) the address defined Tag by Tag
Poll rate	Defines the refresh rate of the Tag name. In a complex application, we can imagine that some Tag name must be refreshed every second – typically for digital input - and other every minute – typically: temperature-

Table 64: DF1 topics configuration setup

6.6.2.2.1 DF1 : serial link

The Destination Device Type and address is always of type :

- *SLC500-x* (where x is the address of your Device (0*254))

i.e.: `Topic A Destination = SLC500-1`

6.6.2.2.2 DF1 : Ethernet routing

Thanks to the ABLogix IOserver, eWON is now able to poll data on SLC500 devices by its ethernet link. Even SLC500 connected behind your ControlLogix Network became available for polling.

With these use of DF1 IOserver, the Serial config can be left unconfigured (baudrate=disabled), only one Topic enabled is required.

You need to use the same Device syntax than in ABLogix IOserver.

IP Address, Port, Link.

- **IP Address** = address on your Ethernet network (i.e.: 10.0.0.50)
- **Port** = value from 1 to 3 representing: 1 = Backplane, 2 = Channel A, 3 = Channel B
- **Link** could be:
 - *Slot*: representing the Slot on the Backplane (0=CPU)
 - *Node ID*: value from 0 to 99 (for ControlNet and DH+)
 - *IP address*

To reach a SLC500 with Ethernet address 10.0.0.60:

Topic A Destination = **10.0.0.60**

To reach a SLC500 connected behind a ControlLogix :

Topic A Destination = **10.0.0.80,1,3,2,45**

- **10.0.0.80** = IP address of the ControlLogix
- **1** = BackPlane
- **3** = Slot 3 (DH+ card)
- **2** = Channel A (of the Card present in Slot 3)
- **45** = NodeID of the SLC500 (in the DH+ network)

6.6.3 Tag name convention

IO Server configuration		
IO Server Name	DF1	
Topic Name	A	
	B	
	C	
Item Name	ValueName, DeviceType-DeviceAddress	Device Type and Address are defined Tag by Tag Only device Type SLC500 is supported Device Address is a number between 0..254
	ValueName	Topic PLC Address is used

Table 65: DF1 IO server configuration

The Item Name can contain the PLC address where the value is polled, or not. If address is not specified at the Tag level, the global topic address will be used.

6.6.3.1 Value Name

6.6.3.1.1 General Description

The general format of value names for data from SLC-500 controllers matches the naming convention used by the programming software. The format is shown below (The parts of the name shown between square brackets are optional).

General Value Name Format: X: [file] element [.field] [/bit]

X: Identifies the file type. The table below summarizes the valid file types, default file number for each type and the fields allowed.

X	File Type	Default file Number	Fields
O	Output	0	
I	Input	1	
S	Status	2	
B	Binary	3	
T	Timer	4	.PRE, .ACC, .EN, .TT, .DN
C	Counter	5	.PRE, .ACC, .CU, .CD, .DN, .OV, .UN, .UA
R	Control	6	.LEN, .POS, .EN, .DN, .ER, .UL, .IN, .FD
N	Integer	7	
F	Floating	8	
A	ASCII	none	

Table 66: value names for DF1 addresses

- **File:** File number must be 0-255 decimal
- **Element:** Element number within the file
- **Field:** Valid only for Counter, Timer and Control files
- **/bit:** Valid for all types except Floating

6.6.3.1.2 Output File Items

Output File Item Format: O[n]:e.s[/b]

- "n" represents the file number and is optional. Value is always zero
- "e" indicates the element number in the file (0..30)
- "s" indicates the sub-element number (0..255)
- "b" specifies the bit (0..15) decimal

6.6.3.1.3 Input File Items

Input File Item Format: I[n]:e.s[/b]

- "n" represents the file number and is optional. Value is always one
- "e" indicates the element number in the file (0..30)
- "s" indicates the sub-element number (0..255)
- "b" specifies the bit (0..15) decimal

6.6.3.1.4 Status File Items

Status File Item Format: S[n]:e[/b]

- "n" represents the file number and is optional. If not specified, it is assumed to be two
- "e" indicates the element number in the file (0..255)
- "b" specifies the bit (0..15) decimal

6.6.3.1.5 Binary File Items

Binary File Item Format: B[n]:e/b

- "n" represents the file number and is optional. If not specified, it is assumed to be three, otherwise must be between 3 and 255 decimal
- "e" indicates the element number in the file (0..255)
- "b" specifies the bit (0..15) decimal

Note:

The format B[n]/b is not supported.

6.6.3.1.6 Timer File Items

Timer File Item Format: T[n]: e [.f] [/b]

- "n" represents the file number and is optional. If not specified, it is assumed to be four, otherwise must be between 4 and 255 decimal
- "e" indicates the element number (3 words per element) in the file (0..255)
- "f" identifies one of the valid values for timer fields specified in the table above.
If omitted it is assumed to be the word containing the status bits
- "b" specifies the bit (0..15) decimal

6.6.3.1.7 Counter File Items

Counter File Item Format: C[n]: e [.f] [/b]

- "n" represents the file number and is optional. If not specified, it is assumed to be five, otherwise must be between 5 and 255 decimal
- "e" indicates the element number (3 words per element) in the file (0..255)
- "f" identifies one of the valid values for counter fields specified in the table above.
If omitted it is assumed to be the word containing the status bits
- "b" specifies the bit (0..15) decimal

6.6.3.1.8 Control File Items

Counter File Item Format: C[n]: e [.f] [/b]

- "n" represents the file number and is optional. If not specified, it is assumed to be six, otherwise must be between 6 and 255 decimal
- "e" indicates the element number (3 words per element) in the file (0..255)
- "f" identifies one of the valid values for counter fields specified in the table above.
If omitted it is assumed to be the word containing the status bits
- "b" specifies the bit (0..15) decimal

6.6.3.1.9 *Integer File Items*

Integer File Item Format: N[n]:e[/b]

- "n" represents the file number and is optional. If not specified, it is assumed to be seven, otherwise must be between 7 and 255 decimal
- "e" indicates the element number in the file (0..255)
- "b" specifies the bit (0..15) decimal

6.6.3.1.10 *Floating File Items*

Floating File Item Format: F[n]:e

- "n" represents the file number and is optional. If not specified, it is assumed to be eight, otherwise must be between 8 and 255 decimal
- "e" indicates the element number in the file (0..255)

6.6.3.1.11 *ASCII File Items*

ASCII File Item Format: An:e[/b]

- "n" represents the file number and is not optional
- "e" indicates the element number in the file (0...255)
- "b" specifies the bit (0..15) decimal

6.6.3.2 *Destination Device Type and Address*

The Device Address is used in the topic definition or in the Tag definition. If used in the Tag definition, it will be separated from the value name by a coma (',').

- **Format: DeviceType-DeviceAddress**

Only device Type SLC500 is supported. Device Address is a number between 0 and 254.

6.7 FINS IO Server

6.7.1 Introduction

The FINS IO Server includes the configuration of:

- The eWON as a FINS Hostlink client (master) to give access on values in CSCJ series OMRON equipments reachable using eWON serial port(s).
- The eWON as a FINS TCP/UDP client (master) to give access on values in CSCJ OMRON equipments reachable using eWON Ethernet interface.
- The eWON as a FINS TCP/UDP server acting as a gateway between the Ethernet/PPP interface and the serial interface (used to connect remotely programming/monitoring software to OMRON FINS supporting equipments reachable using eWON serial ports).

The FINS IO Server is designed to provide simultaneous access to OMRON equipments on its serial interface, and Ethernet interface. The correct protocol will depend on the topic the Tag belongs to. UDP and TCP protocols can be mixed as well on the Ethernet interface.

When the BaudRate in the FINS IO Server is set to a value different than "Disabled", the serial Hostlink Client will be enabled.

The FINS IO Server can be configured in 3 modes:

- Full Duplex mode (eWON serial link must be configured in RS232 mode) without HW handshaking
- Full Duplex mode with HW handshaking
- Half Duplex slave mode

6.7.2 Setup

6.7.2.1 Communication Setup

COM Setup	
COM Port:	SER1 Port (COM:1) ▾
Baud Rate:	Disabled ▾ (default: 9600)
Parity	Even ▾ (default: EVEN)
Databits	7 ▾ (default:7)
Stop Bit(s)	2 ▾ (default: 2)
HW Mode	Full Duplex NO Handshaking ▾ (default: full duplex no Handshaking)
Reply Timeout	<input type="text"/> MS (50..50000, default: 3000)
Ethernet FINS network	<input type="text"/> (0..127, default: 0)
Ethernet FINS node	<input type="text"/> (0..254, default: 0)
Serial FINS network	<input type="text"/> (0..127, default: 0)
Serial FINS node	<input type="text"/> (0..254, default: 0)

Figure 54: FINS IO server COM setup

The following parameters can be modified:

Baud Rate	Disabled, 1200, 2400, 4800, 9600, 19200, 38400, 57600
Parity	None, Odd, Even
Data Bits	7, 8
Stop Bit(s)	1,2
HW mode	Full Duplex no HW handshaking, Full Duplex HW handshaking, Half Duplex

Table 67: FINS IO server COM setup configuration fields

ReplyTimeout	Maximum time the eWON will wait for a valid FINS message response (applicable for ethernet and serial interface).
Ethernet FINS network	Source Network Address (SNA) filled in a FINS request message originating from the eWON and sent out on the ethernet interface.
Ethernet FINS node	Source Node Address (SA1) filled in a FINS request message originating from the eWON and sent out on the ethernet interface. It uniquely identifies the eWON on the ethernet network.
Serial FINS network	Source Network Address (SNA) filled in a FINS request message originating from the eWON and sent out on the serial interface.
Serial FINS node	Source Node Address (SA1) filled in a FINS request message originating from the eWON and sent out on the serial interface. It uniquely identifies the eWON on the serial network.

Table 67: FINS IO server COM setup configuration fields

6.7.2.2 Topic Configuration

REM : Leave 'Global Slave Address' empty to define it tag by tag..

Topic A: <input checked="" type="checkbox"/> Enabled	
Topic Name	A
Protocol:	Fins Serial (default: SERIAL)
Global Device Address	<input type="text"/> Fins Network,Fins Node,HostLink or Ip
Polling Rate	<input type="text"/> MS (default: 2000)
Topic B: <input checked="" type="checkbox"/> Enabled	
Topic Name	B
Protocol:	Fins Serial (default: SERIAL)
Global Device Address	<input type="text"/> Fins Network,Fins Node,HostLink or Ip
Polling Rate	<input type="text"/> MS (default: 2000)
Topic C: <input checked="" type="checkbox"/> Enabled	
Topic Name	C
Protocol:	Fins Serial (default: SERIAL)
Global Device Address	<input type="text"/> Fins Network,Fins Node,HostLink or Ip
Polling Rate	<input type="text"/> MS (default: 2000)

Figure 55: FINS IO server topic configuration

Three (3) topics can be used for the IO Server. These topics are used to give a common property to a group of FINS Tags like:

- Enable/Disable
- Protocol
- Global Device Address
- Polling Rate

Topic configuration item	Description
Topic enabled	Enables or disables polling of all the Tags in the topic.
Protocol	Protocol used for the tags belonging to this topic: FINS Serial, FINS UDP, FINS TCP.
Global Device Address	See below for the Device Address Syntax. If an address is specified here, it will replace (overload) the address-defined Tag by Tag.
Poll rate	Defines the refresh rate of the Tag name. In a complex application, we can imagine that some Tag names must be refreshed every second - typically for digital input - and other every minute - typically: temperature-.

Table 68: FINS IO server topic configuration fields

6.7.2.3 Gateway Configuration

The following parameters can be modified:

Gateway Configuration	
Server port	<input type="text"/> (default: 9600)
FINS TCP server node	<input type="text"/> (0..254, default: 1)
Routing Entry	FINS Destination Network: <input type="text"/> Relay Node: <input type="text"/> (0..127, 0..254)
Routing Entry	FINS Destination Network: <input type="text"/> Relay Node: <input type="text"/> (0..127, 0..254)
Routing Entry	FINS Destination Network: <input type="text"/> Relay Node: <input type="text"/> (0..127, 0..254)

Figure 56: FINS IO server gateway configuration

Server Port	Identifies the FINS UDP/TCP port used by the FINS service (same port defined as well for UDP than TCP).
FINS TCP Server Node	EWON FINS server node address used during FINS TCP session establishment (exchange of the FINS node address messages, and allocation of a FINS TCP Client node if necessary).
Routing Entry 1..3	For each defined destination network, gives the matching relay destination node. This is used to fill in the host link unit ID in the hostlink frame which encapsulates the FINS message sent out on the serial interface.

Table 69: FINS IO server gateway configuration fields

6.7.3 Tag Name Convention

IO Server configuration		
IO Server Name	FINS	
Topic Name	A	
	B	
	C	
Item Name	ValueName, FINS Network, FINS Node, Hostlink or Ip Address	If FINS Serial has been chosen at topic level, Hostlink value has to be defined. If FINS UDP or FINS TCP has been chosen at topic level, IP address has to be defined.
	ValueName	Topic PLC Address is used.

Table 70: FINS IO server - Tag name convention table

The Item Name can contain the PLC address where the value is polled, or not. If address is not specified at the Tag level, the global topic address will be used.

6.7.3.1 Value Name

6.7.3.1.1 General Description

The format of value names for data from CSCJ OMRON controllers is shown below. Its is based on the naming convention used by the CX Programmer programming software. The format is shown below (The parts of the name shown in square brackets are optional).

General Value Name Format: X[bank number:]word address[#bit address]*

- **X: Identifies the Memory area acronym.**
- **[bank number:] is only supported by the E memory area. Values 0 to max values for memory area.**
- **[#bit address] is only supported by A, D, CIO, H and W memory areas. Values 0 to 15.**

**items between brackets "[]" are optional (the brackets should not be used!)*

Supported memory areas:

X	Memory area
A	Auxiliary area
C	Counter area
CIO	Core I/O area
D	Data Memory area
E	Extended Data Memory area
H	Holding area
T	Timer Area
W	Work Area

Table 71: FINS IO server supported memory areas table

6.7.3.2 Global Device Address

The global device address is used in the topic definition or in the Tag definition. If it is used in the Tag definition, it will be separated from the value name by a coma.

6.8 S5-AS511 IO Server

6.8.1 Introduction

This IO server is intended for use with Siemens S5 PLCs communicating via the front programming port using AS511 protocol. The AS511 protocol is specific for each Siemens device. This IO Server has been designed to operate with a set range for Siemens equipment. Use of the IO Server on devices other than those listed is not recommended and not supported.

The Siemens S5 PLC family has a unique memory structure. Data within the PLC are not located at fixed locations within the PLC's memory space. This memory space is continuously updated and revised as you create and modify your PLC logic. When these revisions occur the location of key data elements such as flags, timers, counters, I/O, and data blocks can move around in the PLC's memory. The Siemens S5 IO Server has been designed to read the location of these memory elements when the driver first begins operation upon detecting a communication error or after a request (which is not a read or write request) has been transmitted to the PLC. If you change your PLC configuration you must restart the Siemens S5 IO Server or simply unplug/re plug the cable connection. Both of these actions will cause the Siemens S5 IO Server to reacquire the location of all PLC memory elements.

6.8.2 Setup

IO Server : S5-AS511

S5 AS511 CONFIGURATION

This IO server configures interface with SIEMENS S5 CPUs

REM: Leave 'Global Device Address' empty to define it tag by tag.

Topic A: Enabled

Topic Name	A
Global Device Address	IO Port (COM:1) (Serial port)
Poll Rate	MS (default: 2000)

Topic B: Enabled

Topic Name	B
Global Device Address	IO Port (COM:1) (Serial port)
Poll Rate	MS (default: 2000)

Topic C: Enabled

Topic Name	C
Global Device Address	IO Port (COM:1) (Serial port)
Poll Rate	MS (default: 2000)

Figure 57: S5-AS511 IO server setup window

6.8.3 Communication setup

The AS511 link uses a RS-232 Current Loop functioning mode.
The communication parameters are fixed to the following values:

- 9600 Baud (Fixed)
- Even Parity (Fixed)
- 8 Data Bits (Fixed)
- 1 Stop Bit (Fixed)
- Full duplex no handshaking (Fixed)

6.8.3.1 Supported Devices

<ul style="list-style-type: none"> • Siemens S5 - 90U • Siemens S5 - 95U • Siemens S5 - 100U - 100 • Siemens S5 - 100U - 101 • Siemens S5 - 100U - 103 • Siemens S5 - 101U 	<ul style="list-style-type: none"> • Siemens S5 - 115U - 941 • Siemens S5 - 115U - 942 • Siemens S5 - 115U - 943 • Siemens S5 - 115U - 944 • Siemens S5 - 115U - 945 • Siemens S5 - 135U - 921 	<ul style="list-style-type: none"> • Siemens S5 - 135U - 922 • Siemens S5 - 135U - 928 • Siemens S5 - 155U - 946 • Siemens S5 - 155U - 947
--	--	--

Table 72: Siemens S5 devices supported by the eWON AS-511 IO server

6.8.4 Tag name convention

IO server Name	S5-AS511	
Topic name	A or B or C	
Item Name	ValueName, ComPortNum	COM port is defined Tag by Tag
	ValueName	Topic COM port is used (or default)

Table 73: S5-AS511 IO server Tag name convention table

ComPortNum: Port number used to access the PLC; if not specified, the default COM port is used.

- If a port address is specified in the topic, it is used and overloads per Tag address.
- If no address is specified, neither in the topic, neither at the Tag level, then the default port is used (Default port is the ECIA Port - COM:1).

6.8.4.1 ValueName

DBxLy	Data block x Word offset y, left byte of word (0 - 255)
DBxRy	Data block x Word offset y, right byte of word (0 - 255)
DBxWy	Data block x Word offset y, full word (0 - 65535)
DBxDy	Data block x DWord offset y (-2147483648->2147483647)
Tx or Kx	Timer x (0-65535)
Cx or Zx	Counter x (0-65535)
MxB or FxB	Memento as Byte
MxW or FxW	Memento as Word
MxD or FxD	Memento as DWord
Ix or Ex	Input
Qx or Ax	Output

Table 74: S5-AS511 IO server value names

• **Bit access modifier:**

In any of the items above, it is possible to access a single bit.
 #x must be appended to the Tag name (Bit index goes from 0..31).
 The syntax can be used for reading bits and for writing them as well.

Example:

DB1W13#3 represents bit 3 of W13 in DB 1

REM:

Bit index range depends on item referenced (for W, bit index maximum value is 15)

NOTES:

- **For DB the smallest element is a WORD and we count addresses in Word.**
 This means that DB1W0 and DB1W1 WILL NOT overlap.
- **Timers and Counters are always Words and addresses are counted in WORD also.**
 This means that T0 and T1 WILL NOT overlap.
- **For M, I, Q addresses are counted in BYTES.**
 This means that M0W and M1W WILL overlap.
- **Both syntax (German and English) can be used for I, Q, C**
 which in German gives: E, A, Z.

• **Status register:**

The STATUS Tag is a special Tag that returns information about the current state of the communication for a given device.
 As for other Tags, the status Tag ValueName is composed of:

Status, ComPortNum

- You can define a status Tag for each COM port used.
- If you use the status address, the Tag must be configured as analog.

6.8.4.2 Tag value Meaning:

0	Communication not initialized. Status UNKNOWN. If no Tag is polled on that device address, the communication status is unknown.
1	Communication OK.
2	Communication NOT OK.

Table 75: Tag value meaning

6.9 ABLOGIX IO Server

6.9.1 Introduction

ABLogix IO server is used to communicate with Allen Bradley Logix Series PLCs on the Ethernet link.

With this IO Server, the eWON can poll ABLogix PLCs to read data.

6.9.2 Setup

IO Server : **ABLOGIX**

LOGIX 5000 CONFIGURATION

This IO server configures interface with AB LOGIX 5000 devices

REM: Leave 'Global Device Address' empty to define it tag by tag.

Topic A: Enabled

Topic Name	A	
Global Device Address	<input type="text"/>	IP Address[,Port Address, Link Address]*
Poll Rate	<input type="text"/>	MS (default: 2000)

Topic B: Enabled

Topic Name	B	
Global Device Address	<input type="text"/>	IP Address[,Port Address, Link Address]*
Poll Rate	<input type="text"/>	MS (default: 2000)

Topic C: Enabled

Topic Name	C	
Global Device Address	<input type="text"/>	IP Address[,Port Address, Link Address]*
Poll Rate	<input type="text"/>	MS (default: 2000)

Figure 58: ABLOGIX IO server setup window

You can define 3 topics to organize your polling strategy. Each Topic can be enabled/disabled separately, have a Poll Rate and are linked to a specific Device.

- **Poll rate:**
Is expressed in MilliSeconds and a default value of 2000 will be used if leaved empty.
- **Global Device Address:**
Have the following syntax: IP Address, Port, Link.
 - **IP Address** = address on your Ethernet network (i.e.: 10.0.0.50)
 - **Port** = value from 1 to 3 representing: 1 = Backplane, 2 = Channel A, 3 = Channel B
 - **Link** could be:
 - *Slot*: representing the Slot on the Backplane (0=CPU)
 - *Node ID*: value from 0 to 99 (for ControlNet and DH+)
 - *IP address*

REM: In the case of using a ControlLogix as Gateway, the Device Address could be like following:
IP Address, Port, Link[, Port,Link][, Port,Link]...

6.9.3 Tag name convention

IO Server Configuration		
IO server Name	ABLOGIX	
Topic name	A or B or C	
Item Name	ValueName	Topic PLC address is used
	ValueName, IP address[, Port, Link]	PLC address is defined Tag by Tag

Table 76: ABLOGIX IO server Tag name convention table

6.9.3.1 ValueName

ValueName follows the syntax below:

[PROGRAM:ProgName.]SymbolicTagName

- **ProgName is the name of the program where the Tag is.**
If no ProgName, the Tag is in the global scope (tag is controller type)
- **SymbolicTagName:**
 - Symbol of the tag.
Only atomic type following are supported: BOOL, SINT, INT, FLOAT, DINT, BIT ARRAY
 - Bit selection with <SymbolicTagName>/bit
where bit is the bit number (from 0 to 31)
ex: controlbit/4 read the bit 4 of the controlbit register
 - TIMER, CONTROL, COUNTER predefined types with <SymbolicTagName>.acc (or ctl or pre)
ex: MyVar.acc read the counter of MyVar
 - Element of a table
<Symbol_1>[idx_1].<Symbol_2>[idx_2].<Symbol_3>[idx_3].<Symbol>
With 3 index maximum.
ex: table1[2].subtable[6].element read data named 'element' on the index 6 of the 'sub table' from the index 2 of the 'table1'
 - Element of a structure
<Symbol_1>.<Symbol_2>
ex: CounterObj.init read the 'init' part of the structure 'CounterObj'

6.10 EWON IO Server

6.10.1 Introduction

The eWON IO server is used to interface the eWON INPUTS and OUTPUTS. Depending on your eWON model, you have:

- **Digital inputs**
 - 1 on eWON500/2001/4001
 - 9 on eWON4002
- **Digital outputs**
 - 1 on eWON500/2001/4001
 - 3 on eWON4002
- **Analog inputs**
 - None on eWON500/2001/4001
 - 6 on eWON4002

Additionally, there are a number of Tags that can be addressed with this IO server and which are computed by the eWON IO Server. These additional Tags are used for energy management. In Energy management, the following requirements are taken into accounts:

- **Using digital inputs as counter inputs**
- **Count for a given interval and latch computed result (also save it in historical)**
- **Reject the measurement interval if too long or too short**
- **Adjust eWON's Real Time Clock based on a digital input**

These Tags will be computed if the energy module is enabled. There is no topic name to define for the eWON IO server.

6.10.2 Standard eWON I/O Item Names

6.10.2.1 Tag name convention

The following Tags are available for standard eWON Inputs & Outputs access:

IO Server configuration		Comment		
IO Server Name	EWON			
Topic Name	Empty			
Item Name	DI#	Digital Input (1)		Boolean
	CI#	Counter Input (1)	0 to 2.147.400.000(2)	Analog
	FI#	Counter Input	0 to 255	Analog
	LI#	Latched Counter Input (1)		Analog
	DO#	Digital Output (1)		Boolean
	AI#	Analog Input (1)		Analog
	BI#	Button Input (1)		Boolean

Table 77: eWON IO server configuration parameters - no topic defined

NOTE:

The button input (BI1) can be used during normal eWON operation, if it is pressed for more that 4 seconds while eWON is booting; the flash file system will be erased (please refer to chapter "The eWON firmware upgrade process" on page 229).

(1) The number of item depends of eWON type. See table Table 79 on page 107.

(2) From firmware 4.1S3 to 4.3, the CIx value wrapped at 1.000.000.

Before 4.1S3, the CIx was misinterpreted as negative when value passes over 2³¹.

IO Server configuration		Comment	
IO Server Name	EWON		
Topic Name	SYS		
Item Name	SN_LO	Serial number, low part (see example of use below)	Analog
	SN_HI	Serial number, high part (see example of use below)	Analog
	SYS_UP	Number of seconds since Power Up	Analog
	GSM_REG	GSM Status: 1: Home network 2: Searching registration 3: Registration denied 4: unknown registration 5: Roaming 100: Not applicable 101: Registration in progress usually: 1 or 5: means registered Other: not registered	Analog
	GSM_LEV	GSM level (antenna reception)	Analog

Table 78: eWON IO server configuration parameters - SYS topic

Example of use of SN_LO and SN_HI items in a Basic program:

```

a% = Int(SN_HI@ * 65536) + Int(SN_LO@)
Rem Product code
b% = a% Mod 256
Rem Sequential number
c% = Int(a% / 256) Mod 1024
Rem Week number
d% = Int(a% / 262144) Mod 64
Rem Year number
e% = Int(a% / 16777216) Mod 128
Print "SN: ";e%;" ";d%;"-";c%;"- "b%
    
```

For instance, you can define a Tag for the digital input 1 as follows:

Server name: EWON - **Topic:** empty - **Address:** DI1

eWON500 eWON2001 eWON4001	DI1	Status of input 1 on connector Input/output (bottom side)
	CI1	Counter on input 1 on connector Input/output (bottom side)
	LI1	Latched counter on input 1 on connector Input/output
	DO1	Digital command of output 1 on connector Input/output
	BI1	Status of button on front face
eWON4002	DI1... DI8	Status of input 1 to 8 on connector DI1-DI8 (top side)
	CI1... CI8	Counter on input 1 to 8 on connector DI1-DI8 (top side)
	LI1... LI8	Latched counter on input 1 to 8 on connector DI1-DI8 (top side)
	DI9	Status of input 9 on connector Input/output (bottom side)
	CI9	Counter on input 9 on connector Input/output (bottom side)
	LI9	Latched counter on input 9 on connector Input/output (bottom)
	DO1... DO2	Digital command of relay 1 and 2 on relay connector (bottom side)
	DO3	Digital command of output 1 on connector Input/output (bottom)
	BI1	Status of button on front face
	AI1... AI4	Value of analog input 1 to 4
AI5... AI6	Value of analog input 5 to 6 (PT100 probes)	

Table 79: Available Items for eWON product type

6.10.2.2 Energy configuration with the Llx Tags

The following parameters can be added to the eWON IO server in order to activate the energy support (enter *Energy:1*, validate and the other parameters will be automatically added):

Parameter	Default Value		
<i>Energy</i>	0 or 1	Enabled or not	Once Energy is defined, the eWON will automatically add the other parameters with default value. When 0, no energy computation at all are performed, saving eWON CPU resources.
<i>Debug</i>	0 or 1	Not automatically added	When :1 all synchronisations are logged in the Real time event log.
<i>RefTime</i>		This is the reference time to compute interval. Leaving this empty will start at 1/1/1970 00:00. You may enter a date with hour in the form 08/07/2002 11:15:00 to set a new reference (usually not required).	
<i>IntTime</i>	15	This is the integration time in MINUTES.	
<i>IntToIS</i>	5	This is a tolerance on the integration time in SECONDS. If the measurment interval is shorter or longer than this number of seconds the period is considered as valid. For example IntTime=15 IntToIS =5 means "interval is valid if between 11.14:55 and 11.15:05".	
<i>SyncIO</i>	5	This is the IO number for clock synchronization. 0 means no synchro available, 1..8 is the IO number. Synchronisation is based on the counter input associated with the digital input, each time the IO changes the eWON will try to perform a synchronization (see also <i>SyncToIS</i>)	
<i>SyncToIS</i>	5	This is the tolerance for accepting the synchronization pulse in SECONDS. If the synchronization pulse arrives outside the interval, the pulse is rejected. <i>ERROR REPORT</i> : the first time the pulse is rejected, an error is logged in the event log. Next errors are not logged (except in Real time log if Debug is enabled); When sync are accepted again, a trace is logged in the event file for the first accepted synchronization. REM: Because the internal clock has a precision of 1 second, the time is only updated if the absolute value of the offset is greater than 1 (2 or more). Otherwise the offset jitters between -1 and +1 all the time.	

Figure 59: Energy IO server parameters

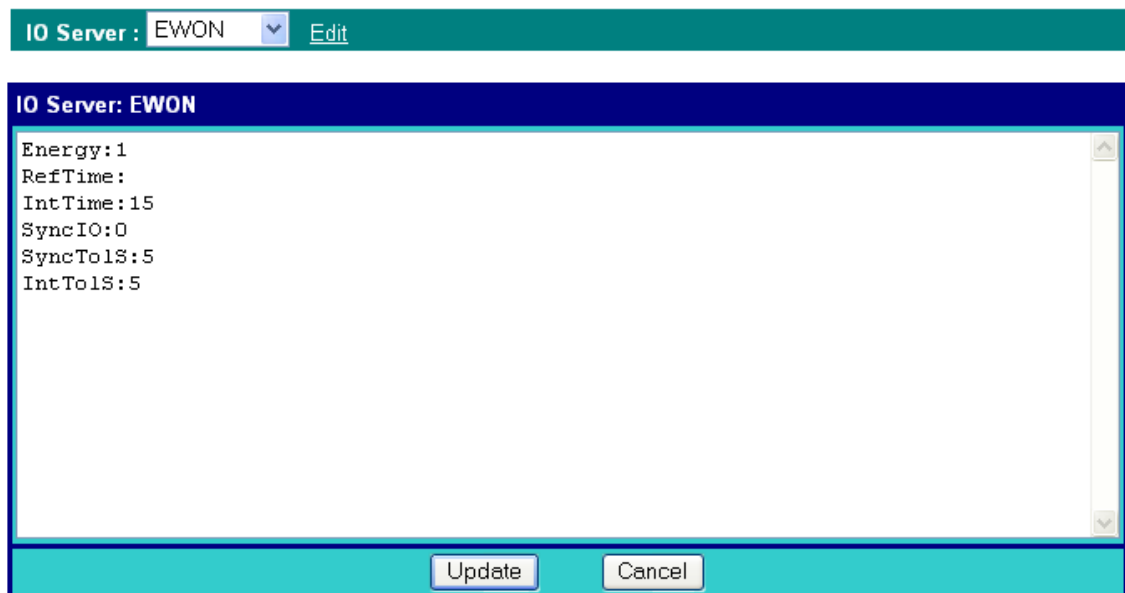


Figure 60: eWON IO server default parameters

6.10.2.2.1 ENERGY TAGS

The following tags are available for energy management:

LI1..LI8	Latched input	When the Integration Period expires, the number of pulses counted during the integration period are logged in the corresponding LIx.
ST1..ST4	Status variables:	
ST1	This is the absolute time when counter were latched. Expressed in seconds since 1/1/1970 minus 0x30000000 HEX (or 805306368 decimal). This huge value is subtracted to maintain precision in the Float storage.	
Latch time - 0x30000000		
ST2	0 means period is valid (within tolerance), 1 means period is invalid.	
Period status		
ST3	Length of the period in second.	
Period length sec		
ST4	This 32 bit counter counts is increased by 1 after each integration period. It can be used with ONCHANGE to perform operations when period expires.	
Period Num		

REM1: Energy tags are logged after each new period IF Logging is enabled, even if value has not changed since previous period.

REM2: Even if Deadband is -1 and Interval is 0. If not you will have additional points and even maybe duplicate points.

6.10.2.2.2 32 BIT COUNTER TAGS:

The eWON IO Server provides now 8 32 bit counter tags named: CI1..CI8. These counter are writable but writing in these register affect the LIx inputs of the Energy IO server module.

6.10.2.3 Analog Input Value Range (eWON4002/eWON1002)

The Analog Inputs (AI1 to AI6) come from a 12 bits ADC. Then, the values read on a AI# tag go from 0 to 4095.

6.10.2.3.1 Configurable analog input AI1 to AI4

- In 0-10V mode

AI#	Volt
0	0
4095	10

If you need to read the tag value converted in Volt, you can set a factor of $0.0024420 = 10/4095$ with an offset of 0.

Tag Name: Tag_AI1 Page: Default

Tag Description:

I/O Server Setup

Server Name: EWON Topic Name:

Address: AI1 Type: Analog Force Read Only:

eWON value = IO Server Value * 0.002442 + 0

Figure 61: Tag's configuration: reading the tag value in Volt

- In 0-20 mA mode

The shunt resistor for measuring current is a 220 Ohm.

AI#	mA
0	0
1802	20

If you need to read the tag value converted in mA, you can set a factor of $0.011099 = 20/1802$ with an offset of 0.

Tag Name: Tag_AI_2 Page: Default

Tag Description:

I/O Server Setup

Server Name: EWON Topic Name:

Address: AI1 Type: Analog Force Read Only:

eWON value = IO Server Value * 0.011099 + 0

Figure 62: Tag's configuration: reading the tag value in mA

6.10.2.3.2 PT100 input AI5 and AI6

These two entries measure a Resistance value in Ohm.

AI#	Ohm
0	162.13
4095	79.625

As you can see, the slope is negative, and if you need to read the tag value converted in Ohm, you can set a factor of $-0.02014774 = -82.505/4095$ with an offset of 162.13.

The screenshot shows a configuration window for a tag named 'Tag_PT100'. The 'I/O Server Setup' section is highlighted in blue. It contains the following fields:

- Server Name: EWON
- Topic Name: (empty)
- Address: AI5
- Type: Analog
- Force Read Only:
- Conversion formula: $eWON\ value = IO\ Server\ Value * -0.02014774 + 162.13$

Figure 63: Tag's configuration: reading the tag value in mA

If you need to convert in °C: $temperature = AI * (-0.0394075) + 161.376$.

The default range of PT100 in eWON is -50°C to 150°C.

Full range of ADC is: -52.92°C to 161.376°C (79.625 Ohm to 162.13 Ohm).

Note:

- eWON4002 prior to S/N 0517-xxxx-89 have a PT100 range from 0°C to 131.9°C and must apply the following formulas:
 $Ohm = IOValue * (-0.012402) + 150.8$
 $°C = IOValue * (-0.0322) + 131.9$
- eWON4002 from S/N 0517-xxxx-89 to S/N 0521-xxxx-89 have a PT100 range from -56°C to 151°C and must apply the following formulas:
 $Ohm = IOValue * (-0.0195068) + 158.2$
 $°C = IOValue * (-0.050677) + 151.2$

6.10.3 Setup

The Energy IO Server is based on the eWON IO Server. The configuration consists in advanced parameters from this latter IO Server. The following screen shot shows the standard empty eWON IO Server configuration page.

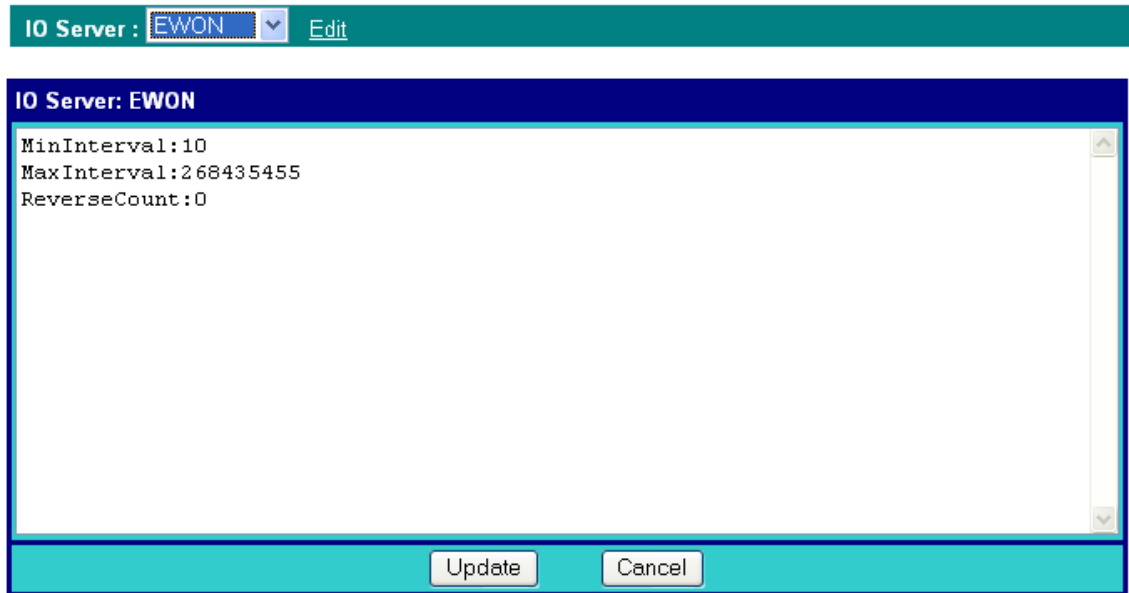


Figure 64: eWON IO server configuration page

6.10.3.1 Configuration of the counter pulse length

Pulse length for counter can be configured on all eWON types. The configuration must be entered in the eWON IO server edit area. The following parameters are used:

MinInterval:	default=10, min=10
MaxInterval:	default=268435455, max=268435455
MinInterval and MaxInterval are entered in milliseconds	
ReverseCount:	default=0
The pulse length must be between MinInterval and MaxInterval to be accepted, the measurement resolution is 5 msec (the precision of the pulse length measured is 5 msec)	
ReverseCount can be used if pulses entered in the eWON are reversed, reversed means that signal is normally high on the eWON input and it goes low when the pulse occurs.	

Table 80: counter pulse length configuration

Example of configuration:

MinInterval:40
 MaxInterval:1000
 ReverseCount:0

Counts pulse with length longer than 40 msec and shorter than 1000 msec, other pulses are ignored.

6.11 MEM IO Server

6.11.1 Introduction

The MEMORY IO Server is not a real IO server because values do not come from a peripheral. Memory Tags (Tags defined with the MEM IO server) are rather sorts of variables that can be modified by user input or by a BASIC application.

These Tags are very useful for combining different Tags and consider the result as an actual Tag, thus having data logging capabilities and alarm management capabilities like for all other Tags.

6.11.2 Setup

There is no setup for the MEM eWON IO server.

6.11.3 Tag name convention

If no topic is specified, the Tag is a standard memory Tag. Its value is set to 0 when the eWON boots and the Tag is read/write, it can be updated through user actions with script or web pages.

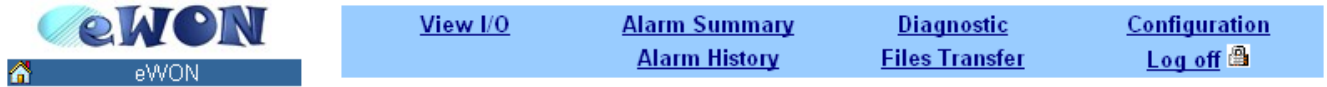
IO Server configuration		
IO Server Name	MEM	
Topic Name		If no topic is specified, the Tag is a standard memory Tag. Its value is set to 0 when the eWON boots and the Tag is read/write, it can be updated through user actions with script or web pages
	RET	If topic is set to RET, the Tag is retentive, each change is saved to flash and when the eWON boots, the last value of the Tag is restored. This feature can be used to use Tags as parameters.
Item Name	DON'T CARE	The MEM "Item Name" is "don't care", the Tag name is automatically copied in the MEM address by the eWON, user can change it but value will immediately be overwritten again by the Tag name.

Table 81: MEM IO server Tag name convention

Important: RET values are saved to FLASH memory, this is not an immediate operation and the FLASH memory can only be written and erased a limited (even if huge 1.000.000) number of times. This means that you should not use a BASIC script for changing retentive values at very high rate. 1 Tag/Sec. is a maximum AVERAGE rate (occasional higher rates are no prohibited at all).

7 eWON Monitoring Web Interface

If you have configured the eWON with some Tag name, you can now come back on the eWON SCADA menu (**Main Menu**). The SCADA menu of the eWON looks like this:



User: (Adm)

Connection Date: 19/05/2005 12:05:43

Figure 64: eWON SCADA menu

The little home icon under the eWON logo can be accessed from any page of the eWON Scada and will always lead your browser directly to this main menu of the Scada.

You can always click on the eWON logo to obtain the definition and system information about the eWON you are connected to. A second navigator window will be opened giving you a summary of the main information about the current eWON:

eWON Information	
Identification	eWON
Additional Info	
IP Address	10.0.120.11
Version Codename	EW_4_1S2
Revision Number	4.1
Serial Number	0508-0003-88
System Info	
IO Revision	1
Modem type	Internal 33600 (2)
Free Config Mem.	124617
Free Prog. Mem.	129111

REM: This page was opened in a new window. You may close this window to return to the previous view.

Figure 65: eWON information page

7.1 Real-time screen

Click on the **View IO** item from the SCADA menu to obtain the list of all eWON Tag names and associated real-time values. To refresh the value, click again on **View IO** item.

	Tag Name	Value	New Value		Description
	BottleInCast	0	<input type="text" value="0"/>	Update	
	GA_AN01	0			Atmospheric pressure input [kPa]
	GD_DO01	1	<input type="text" value="1"/>	Update	First relay output
	OpenDoor	0	<input type="text" value="0"/>	Update	
	Pm_S01_e	0			Submetering 1 Pm exponent
	Pm_S01_m	0			Submetering 1 Pm mantissa
	Pm_S02_e	0			Submetering 2 Pm exponent
	Pm_S02_m	0			Submetering 2 Pm mantissa
	Pm_S03_e	0			Submetering 2 Pm exponent
	Pm_S03_m	0			Submetering 2 Pm mantissa
<input type="checkbox"/>	Tank_Level	0	<input type="text" value="0"/>	Update	Level of the Tank

Figure 66: eWON real-time screen

7.1.1 Change Tag value

As shown in the example above on the second row from the Tag list, if the Tag name is an output and if the user has the right to 'force output', an edit box and an Update link is available.

To change the value of the output just fill the related edit box with the new wanted value and click on the related Update link. If the Tag is of Boolean type, a combo box with values 0 (zero) and 1 is then displayed.

7.1.2 Alarm state

If the Tag name is in an alarm state, a yellow bell appears at its left side:



You can click on this picture to access directly the Alarm summary screen.

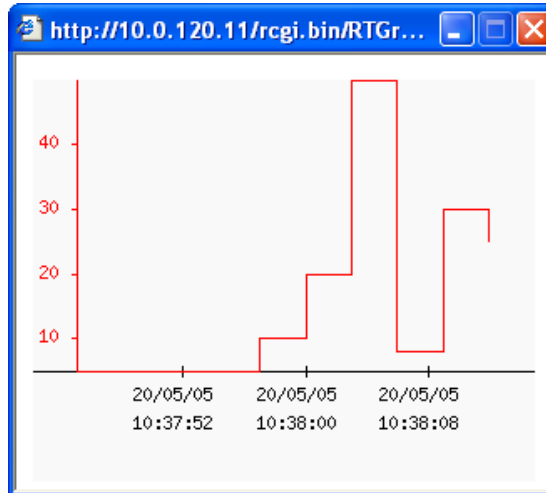
Tag description: The description of the Tag is also displayed for each Tag.

7.1.3 Real time graph

If a real time recording is enabled for a Tag, a small icon will appear next to its name:



This small icon is an hyperlink to show the graph picture. When the link is clicked, another window will open and after a small delay (about 3 seconds), a graph will be displayed:



This graph displays the whole real time window logged. It is possible with user defined web pages to limit the display range of the graph (please refer to chapter “User defined Web Site” on page 179).

Important: If the main explorer window hides the real time window, clicking again on the real time graph hyperlink will not bring it forward. You should bring it forward manually.

7.1.4 Historical window

A check box may also appear in the first column of the View I/O screen meaning that you can choose this Tag to view the historical trend. It appears only if the historical logging has been enabled in the Tag description (see chapter “Tag Setup” on page 56).

7.2 Historical Trending screen

Warning: depending on eWON's version - c.f. table at the end of this manual.

From the real-time screen, users can select Tags (providing that historical logging has been enabled for those Tags) by clicking on the related checkbox and request a graph for these Tags by clicking on the **Show Graph for selection** link above the real-time tabs. Up to 4 Tags can be selected at the same time to be plotted on the graph.

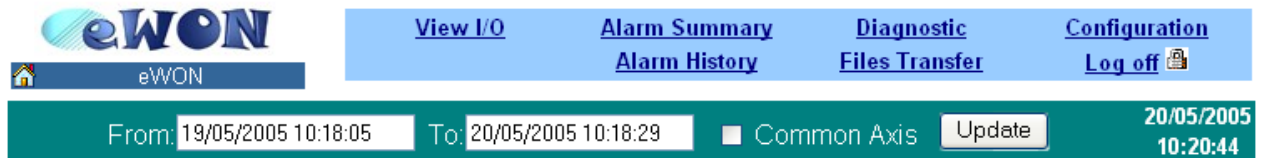


Figure 67: eWON historical trend configuration - 1

The user has then to select the correct time range for the Tag's selection and click on the **Update** link to obtain a historical trend. The common axis checkbox allows visualizing all Tag names on the same axis. If the scales are different, it is cleaner to display each curve with its own scale (by unchecking the **Common Axis** checkbox).

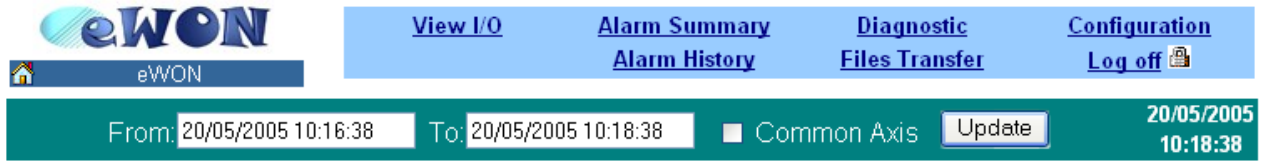
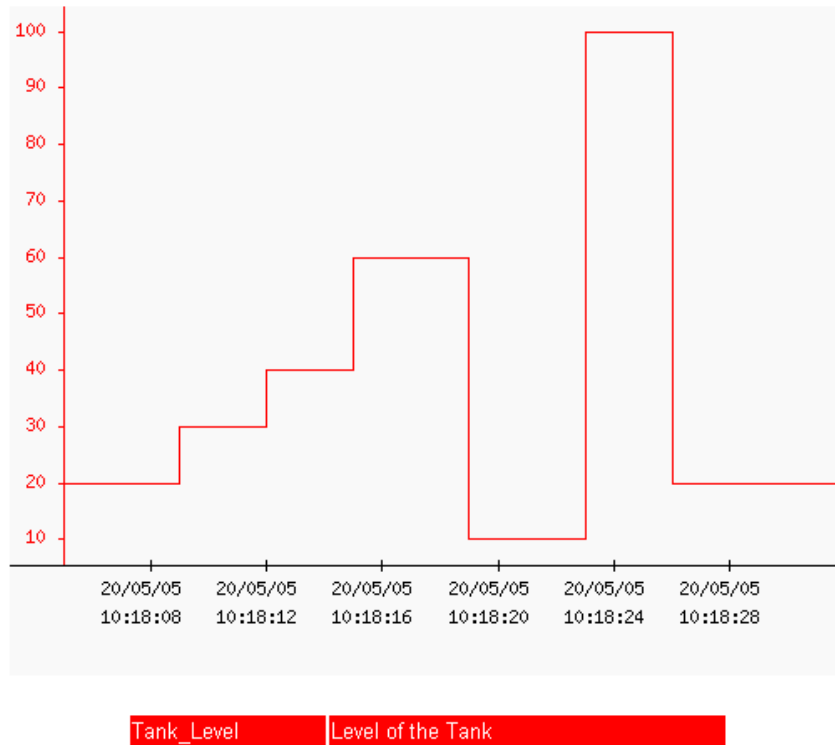


Figure 68: eWON historical trend configuration - 2



Tank_Level Level of the Tank

Figure 69: eWON historical trend graph

7.3 Real-time Alarm screen

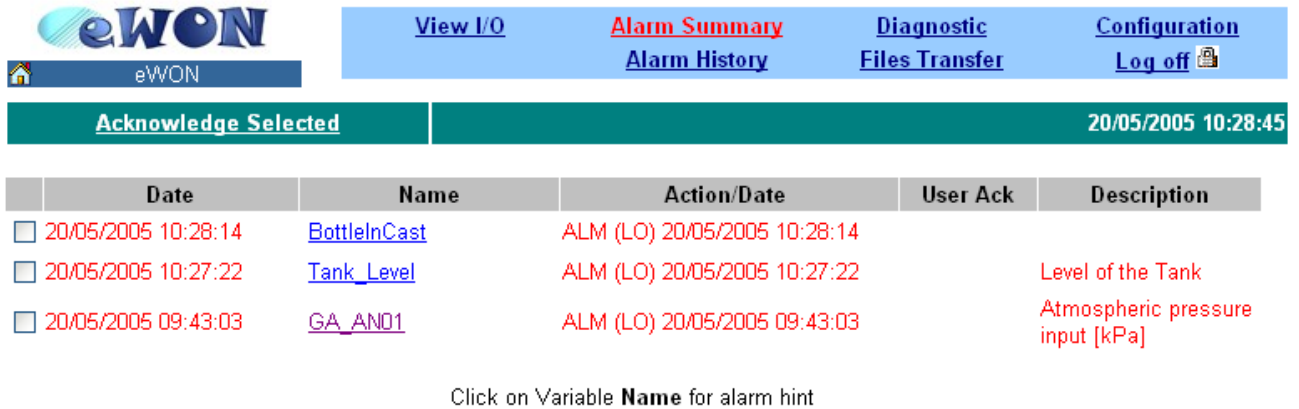


Figure 70: eWON real-time alarm screen

The real-time alarm page lists all Tag names currently in an alarm state.

- The 'Date' tab displays the eWON date and hour at which the Tag was in alarm.
- The 'Name' tab displays the Tag in alarm.
- The 'Action/Date' tab displays the last status of the Tag and the date and hour of this state.
- The 'User Ack' tab displays the user who acknowledged the alarm.
- The 'Description' tab displays the description of the Tag in alarm.

There are several types of Alarm status:

ALM (HI)	ALARM status, the current value is in warning high position
ALM (HIHI)	ALARM status, the current value is in insecure high position
ALM (LO)	ALARM status, the current value is in warning low position
ALM (LOLO)	ALARM status, the current value is in insecure low position
ALM	ALARM status, the present value is out of defined threshold (Boolean Tag)
RTN	Return to Normal status, the present value is inside the defined threshold but has been out of threshold before and hasn't be acknowledged
ACK	Acknowledgment status, the present value is out of defined threshold but someone has acknowledged the alarm

Table 82: alarm status types

It is possible to define an alarm hint:



Alarm Hint	
Variable Name	GA_AN01
Description	Atmospheric pressure input [kPa]
Alarm Hint	

REM: This page was opened in a new window. You may close this window to return to the alarm list.

Figure 71: eWON alarm notification: pre-defined hint

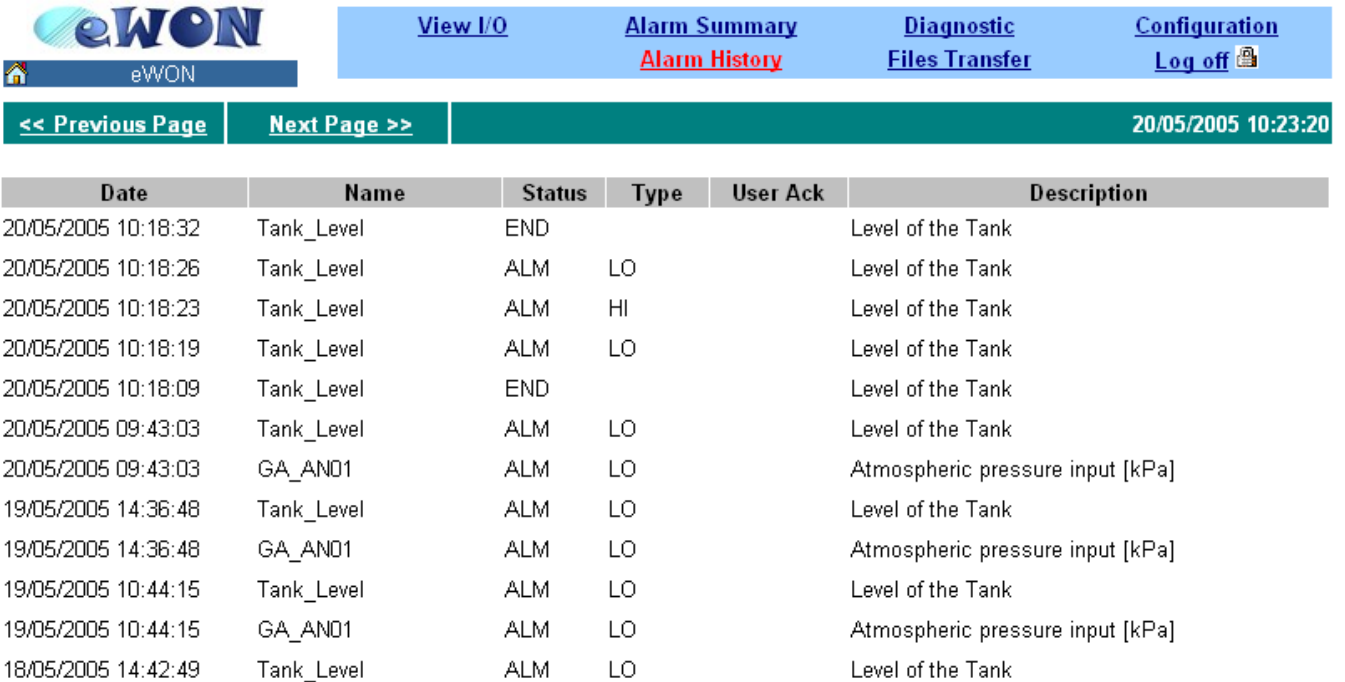
This alarm hint can be useful to help the users that are allowed to take actions regarding the alarms.

7.4 Historical Alarm screen

The historical alarm screen is used to find the alarms generated in the past and to know who acknowledged a specified alarm. All the alarms are stacked from the top to the bottom.

The information given is:

- The date and the hour of the event.
- The Tag name.
- The type of the event: ALM, RTN, ACK or END (RTN and ACK).
- The user who acknowledged the alarm.
- The description of the Tag.



Date	Name	Status	Type	User Ack	Description
20/05/2005 10:18:32	Tank_Level	END			Level of the Tank
20/05/2005 10:18:26	Tank_Level	ALM	LO		Level of the Tank
20/05/2005 10:18:23	Tank_Level	ALM	HI		Level of the Tank
20/05/2005 10:18:19	Tank_Level	ALM	LO		Level of the Tank
20/05/2005 10:18:09	Tank_Level	END			Level of the Tank
20/05/2005 09:43:03	Tank_Level	ALM	LO		Level of the Tank
20/05/2005 09:43:03	GA_AN01	ALM	LO		Atmospheric pressure input [kPa]
19/05/2005 14:36:48	Tank_Level	ALM	LO		Level of the Tank
19/05/2005 14:36:48	GA_AN01	ALM	LO		Atmospheric pressure input [kPa]
19/05/2005 10:44:15	Tank_Level	ALM	LO		Level of the Tank
19/05/2005 10:44:15	GA_AN01	ALM	LO		Atmospheric pressure input [kPa]
18/05/2005 14:42:49	Tank_Level	ALM	LO		Level of the Tank

Figure 72: eWON alarm history page

The **Previous Page** link is used to go back to the past and the **Next page** link is used to go to the present. When no more alarm is available for display the following message appears: "No more history page".

7.5 Files transfer

This link gives you access to a page where the eWON files are listed:



File Name	Description
Events.htm	Events log as table
sstat.htm	Scheduled status as table
estat.htm	System status as table
rt_alm.txt	Real time alarms
inst_val.txt	Instantaneous values as text
inst_val.bin	Instantaneous values (binary)
events.txt	Events log
hst_alm.txt	Alarms history
var_lst.txt	Variables list and details
var_lst.csv	Variables list and details
program.bas	Program
ewonfwr.edf	Firmware
dump.ppp	PPP Dump
config.bin	Binary config
config.txt	Text config
comcfg.txt	Text COM config
ircall.bin	All historical logs
irc_GA_AN01.txt	GA_AN01 Historical log
irc_GD_DO01.txt	GD_DO01 Historical log
irc_Pm_S01_e.txt	Pm_S01_e Historical log
irc_Pm_S01_m.txt	Pm_S01_m Historical log
irc_Pm_S02_e.txt	Pm_S02_e Historical log
irc_Pm_S02_m.txt	Pm_S02_m Historical log
irc_Pm_S03_e.txt	Pm_S03_e Historical log
irc_Pm_S03_m.txt	Pm_S03_m Historical log
irc_Tank_Level.txt	Tank_Level Historical log

Figure 73: files transfer page

8 Retrieving Data from eWON

8.1 List of eWON files

The eWON flash file system contains the following files (R: read, W: write). Detailed information about files format are contained in the Technical Notes you can download from the eWON web site (**Support/Documentation/Technical notes** on <http://www.ewon.biz>):

- TN02: eWON files format
- TN03: ircAll.bin format
- TN12: var_lst.txt format

File Name	Type	Description
Events.htm	R	eWON events occurred (as log in, log out, error) - html format
sstat.htm	R	All the scheduled actions status for the current session - html format
estat.htm	R	Current status from the eWON - html format
rt_alm.txt	R	Real-time alarms list
inst_val.txt	R/W	Contents the instant values from the Tags that have been defined in eWON - text format
inst_val.bin	R/W	Instant values from eWON's Tags - binary format
events.txt	R	eWON events occurred (as log in, log out, error) - text format
hst_alm.txt	R	Historical alarms list - text format
var_lst.txt	R/W	List of all eWON Tag names which are logged - text format
var_lst.csv	R/W	List of all eWON Tag names which are logged (csv format)
program.bas	R/W	All source code of eWON basic program
ewonfwr.edf	W	eWON's firmware file
dump.ppp	R	ppp dump file (can be analyzed by using a software that manages the .ppp format)
config.bin	R/W	eWON configuration - binary format
config.txt	R	eWON configuration - text format
comcfg.txt	R	eWON COM configuration - text format
ircall.bin	R	All the binary values from all Tags that have been defined in eWON
icr_XXXXX.txt	R	Incremental recording file. One file per each of the Tags that are listed inside the var_lst.txt file
remote.bas	W	Single shot execution section

Table 83: eWON files list

- Files *ewonfwr.edf* and *config.bin* are binary files and cannot be modify by users.
The *config.bin* file is Read/Write to allow users to "copy/paste" configuration from one eWON to another.
- The *program.bas* file is Read/Write; that's allowing you to design your program offline with your text editor and to upload it to the eWON.
- *Remote.bas* is not readable. When a *remote.bas* file is "dropped" in the eWON root directory, this program is executed 1 time. This feature can be used to execute single shot action in an automated way.
- The following files only appear if the user owns the "Config rights":
 - Program.bas
 - ewonfwr.edf
 - config.bin
 - config.txt
- During FTP session, all files are read-only, except for users who own the "Config rights".

8.2 Files Format

Files format are text semicolon (;) separated files that may be inserted inside a program such as Excel™. The first row gives the columns names, and the next ones the values of the different columns.

For example, the content of the *rt_alm.txt* file:

```
"TagId";"AlarmTime";"TagName";"AlStatus";"AlType";"StatusTime";"UserAck";"Description"
1;"7/09/04 13:19:12";"Level tank A";"ALM";"LOW";"7/09/04 13:19:12";"";"Fuel level on Tank A"
```

8.3 FTP transfer

FTP stands for the classical File Transfer Protocol used on the Internet.

8.3.1 FTP Software tools

A lot of FTP software tools are available on the market. A very professional freeware FTP software called **SMART FTP** is available on the eWON web site for download <http://www.ewon.biz/> (/Support/Resource Links). This tool allows you to easily manage file retrieving by drag and drop. Please refer to this product's user manual in order to connect to the eWON.

The settings needed in order to open a TCP/IP connection to the eWON are (with factory settings):

IP address	10.0.0.53
Login	adm
Password	adm
Port	21 (default FTP port)
Anonymous unchecked (you are connecting with the above login and password)	

Table 84: eWON factory TCP/IP connection parameters

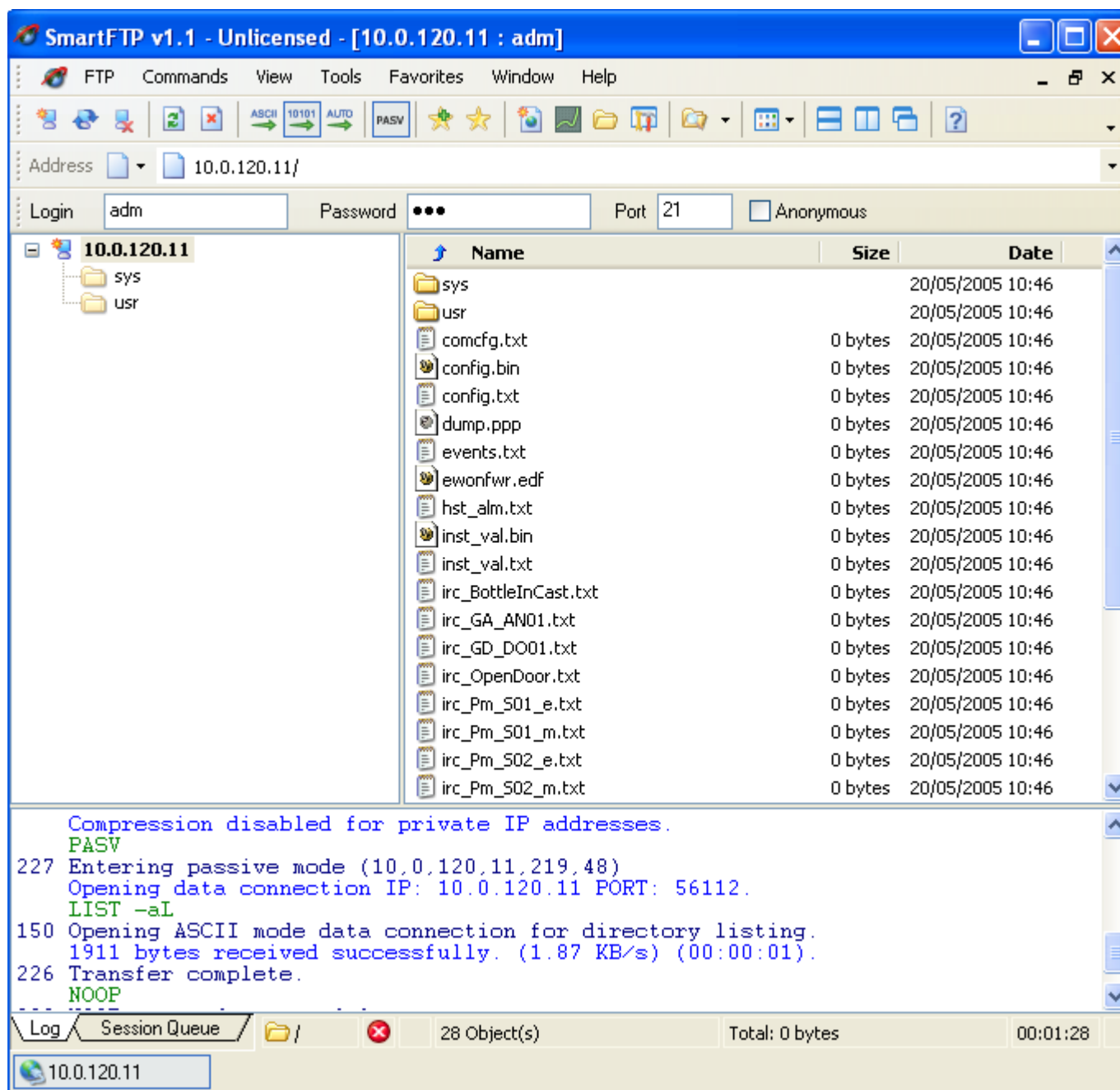


Figure 74: eWON FTP site page

8.3.2 FTP session

Using a common FTP session and using all standard FTP command, you can also easily retrieve data from the eWON. The main interest of this method is to automate the upload by a FTP scripting executed automatically.

8.3.3 Via eWON web site

One of the Web pages available within the eWON web site is called *Files Transfer*. This page contains a table with a list of hyperlinks. The Hyperlinks can be used to download files directly from the eWON web site.

9 Programming the eWON

9.1 BASIC language definition

9.1.1 Introduction

The program of the eWON is based on syntax close to the BASIC, with many specific extensions.

9.1.2 Program flow

IMPORTANT:

- It is very important to understand how the eWON executes its program.
- You should understand the difference between how the eWON stores the program and how it is executed.
- The eWON has a program task that extracts BASIC requests from a queue and executes the requests.

A request can be:

- A single command, example: `MyVar=1`
- A branch to a label, example: `goto MyLabel`
- A list of commands (program block)

In the first case, the command is executed then the BASIC task is ready again for the next request.

In the second case, the BASIC task goes to label `MyLabel` and the program executes until the `END` command is encountered or until an error occurs.

Suppose the eWON has no program, and you create:

An INIT SECTION with:

```
CLS
MyVar=0
```

A CYCLIC SECTION with:

```
FOR V%=0 TO 10
  MyVar = MyVar+1
NEXT V%
PRINT MyVar
```

Then create a new section:

MY NEW SECTION with:

```
MyNewSection:
  MyVar=0
  pPRINT "MyVar is Reset"
```

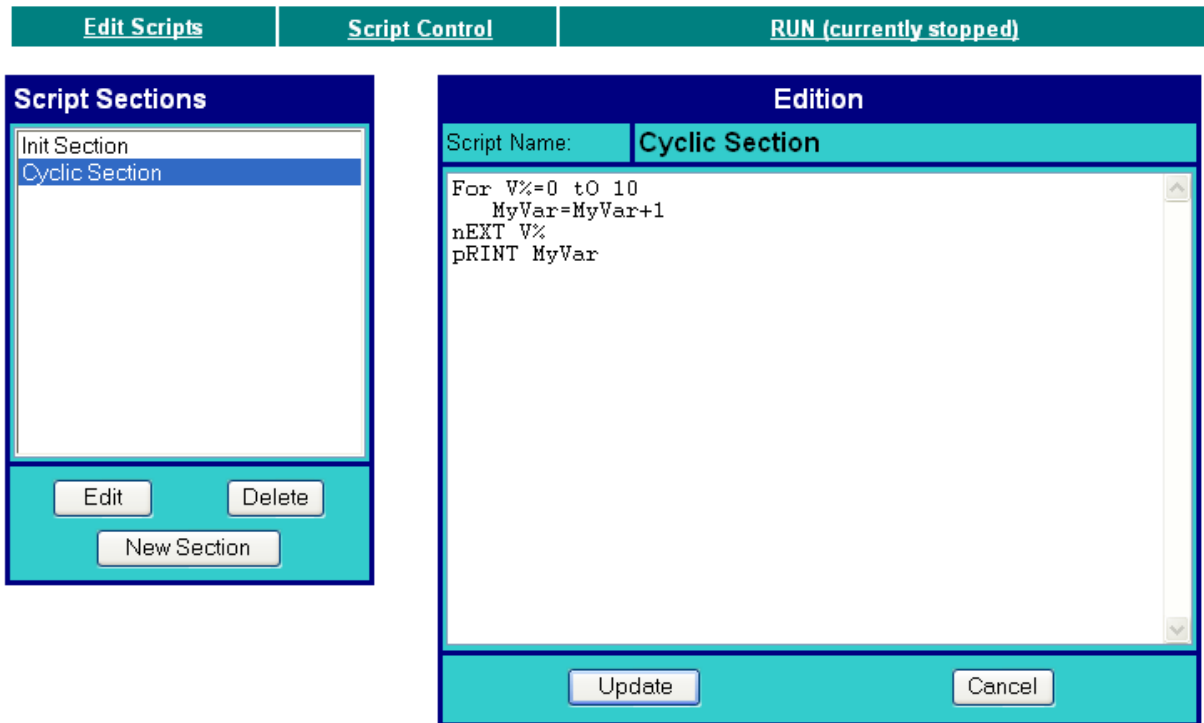


Figure 75: eWON cyclic section code

If you download the corresponding **program.bas** file using an FTP client (see "FTP transfer" on page 120), you will obtain the following program:

```
rem --- eWON start section: MY NEW SECTION
rem --- eWON user (start)
MyNewSection:
  MyVar=0
  pRINT "MyVar is Reset"
rem --- eWON user (end)
end
rem --- eWON end section: MY NEW SECTION
rem --- eWON start section: Cyclic Section
ewon_cyclic_section:
rem --- eWON user (start)
for V%=0 to 10
  MyVar=MyVar+1
nEXT V%
pRINT MyVar
rem --- eWON user (end)
end
rem --- eWON end section: Cyclic Section
rem --- eWON start section: Init Section
ewon_init_section:
rem --- eWON user (start)
CLS
M

yVar=0
rem --- eWON user (end)
end
rem --- eWON end section: Init Section
```

As you can see, the code you have entered is present, but the eWON has added some remarks and labels in order to allow edition and to provide program flow control.

For each section in the editor, the eWON has added an END statement at the end to prevent the program from continuing to the next section. The example also shows that any label is global to the whole program and should not be duplicated.

We can also see here that there is not correlation between the section name and the label used in that section.

The section name is only a way to organize program listing during edition in the eWON.

The section names can contain spaces while the program labels can't.

When the program starts (click **RUN** from the web site for example), the eWON posts 2 commands in the Queue:

Queue pos	Content	Type
...		
3		
2	goto ewon_cyclic_section	CYCLIC_SECTION
1	goto ewon_init_section	INIT_SECTION

Table 85: BASIC Queue - 1

The eWON BASIC task will read the request in the queue that has the lowest index and will execute it until an END is found or until an error occurs.

The first command is "goto ewon_init_section". The following lines will be executed:

```
ewon_init_section:
rem --- eWON user (start)
CLS
MyVar=0
rem --- eWON user (end)
end
```

The END command on the last line will end the program and the BASIC task will check in the queue for a new request:

Queue pos	Content	Type
...		
3		
2		
1	goto ewon_cyclic_section	CYCLIC_SECTION

Table 86: BASIC Queue - 2

The first available command is "goto ewon_cyclic_section", it will also be executed until the END is found.

When this END is executed the BASIC task will detect that the section it has just executed was a CYCLIC_SECTION and it will post a new "goto ewon_cyclic_section" request in the queue.

This is how the program is continuously executed forever while the BASIC is in RUN mode.

There are a number of actions that can be programmed to occur upon event, like ONTIMER:

```
TSET 1,10
ONTIMER 1,"goto MyLabel"
```

Suppose you add the above lines in the INIT SECTION, it will start timer 1 with an interval of 10 seconds and program a "goto MyLabel" request when timer 1 ellapses. What actually happens when the ONTIMER occurs is that the eWON posts the "goto MyLabel" request in the BASIC queue.

Queue pos	Content	Type
...		
3		
2	goto MyLabel	
1	goto ewon_cyclic_section	CYCLIC_SECTION

Table 87: BASIC Queue - 3

When the CYCLIC SECTION will be finished, the timer request will be extracted from the queue and then executed. If the CYCLIC SECTION takes a long time to execute, then the time can elapse more than once during its execution, this could lead to more timer action to be posted in the queue:

Queue pos	Content	Type
...		
5		
4	goto MyLabel	
3	goto MyLabel	
2	goto MyLabel	
1	goto ewon_cyclic_section	CYCLIC_SECTION

Table 88: BASIC Queue - 4

The basic queue can hold more than 100 requests, but if TIMER goes too fast or if other events like ONCHANGE are used the queue can overflow, in that case an error is logged in the events file and requests are dropped. You can also see that the ONTIMER request is not executed with the exact precision of a timer, depending on the current load of the BASIC when the timer elapses. When an ASP block has to be executed for the delivery of a WEB page to a client, the ASP block is also put in the queue Example: if ASP block contains the following lines:

```
FromWebVar = Var1!
PRINT #0;TIME$
```

Queue pos	Content	Type
...		
3	FromWebVar = Var1! PRINT #0;TIME\$	
2	goto MyLabel	
1	goto ewon_cyclic_section	CYCLIC_SECTION

Table 89: BASIC Queue - 5

If a request in the queue contains more than 1 BASIC line, what actually happens is the following:

- The block is appended to the end of the program as a temporary section:

```
ewon_one_shot_section::
FromWebVar = Var1!
PRINT #0;TIME$
END
```

- The temporary label is called (goto ewon_one_shot_section).
- When the execution is done, the temporary section is deleted from the program.

As a consequence, we have the following:

- Any global variable, or label can be used in REMOTE.BAS or ASP blocks; you can call subroutines in your ASP blocks and share common variables with the program.
- If a section is being executed when the ASP section is posted, all the requests in the queue must first be executed. This may have an impact on the responsiveness of the WEB site when ASP is used.
- When using ASP you would better group your blocks to avoid posting too many different requests in the queue. By doing this you will reduce queue extraction and BASIC context switches.
- If a big amount of ASP request (or long ASP request) is posted to the BASIC by the WEB server, it may slow down normal execution of the BASIC.
- Sections are never interrupted by other sections: this is always true, when a program sequence is written, it will never be broken by another execution (of timer or WEB request or anything else).

9.1.3 Character string

A character string can contain any set of characters.

When creating an alphanumeric string with a quoted string the ' or " delimiter can be used:

Examples:

```
"abcd"
' abdc'
"abc`def' ghi "
```

Are 3 valid quoted strings.

A character string can be stored either in an alphanumeric type variable, or in an alphanumeric variable array.

9.1.4 Command

A command is an instruction that has 0 or several comma (,) separated parameters.

There are 2 exceptions to the comma separator: PRINT and PUT.

Examples:

```
GOTO Label
PRINT
CLS
SETSYS TAG, "name", "Power"
SETSYS TAG, "SAVE"
```


9.1.5 Integer

An integer is a number between -2147483648 and +2147483647. This number can be stored in an integer variable. When a parameter of integer type is specified for a function or a command and the variable past is of real type, the eWON automatically converts the real value to an integer value. When the expected value is of integer type and the past value is a character string, the eWON generates an error.

9.1.6 Real

A real number is a number in floating point representation of which value is between -3.4028236 10E38 and +3.4028234 10E38. Value of this type can be stored in a variable of real type or in an array of reals.

Precision: a Real number has approximately 10 significant digits.

This means that conversion of a number with more than 10 significant digits to real will lead to a lost of precision.

When a function expects a real number and an integer is passed, the eWON automatically converts the integer into a real value. If the function waits for a real and a character string is passed, the eWON generates an error.

9.1.7 Alphanumeric character

An alphanumeric character is one of the ASCII characters. Each ASCII character has a numerical representation between 0 and 255. The ASCII basic function returns the ASCII code of a character, and the CHR\$ function converts the ASCII code to a string of 1 character.

9.1.8 Function

A function is a BASIC command having 0 or several parameters and returning a result that can be of integer, real or string type. Examples:

```
ASCII "HOP"
GETSYS TAG, "NAME"
PI
```

9.1.9 Operators priority

When these operations appear in expressions, they have the following priority:

- **Bracket terms: maximum priority**
- **All functions except NOT and - (inversion)**
- **Inversion of sign -**
- ***, /, ^, MOD (modulo function)**
- **+, -**
- **=, >, <, <=, >=, <>**
- **NOT, BNOT**
- **AND, OR, XOR: minimum priority**
- **^ operator is the Power operator**
i.e.: $2^4 = 2*2*2*2$

The expressions are ordered by decreasing order of priority.

See also:

"NOT" on page 150, "BNOT" on page 132, "AND" on page 131, "OR" on page 158, "XOR" on page 168, "MOD" on page 150

9.1.10 Type of Variables

9.1.10.1 Integer variable

Syntax

a%

a is a letter from "a" to "z". The name of the variable is followed by the "%" letter to indicate that it is an integer variable. An integer variable can contain a number of type integer.

9.1.10.2 Real variable

Syntax

abcdef

abcdef is the name of the variable that can contain up to 200 characters. Variable names are case insensitive:

AbCdEf and ABCDEF represent the same variable.

The variable name can only contain the letters "a" to "z"; all other characters are not allowed. The variable name can contain alphabetical characters, numbers and "_" underscore, but name must begin with an alphabetical character. Others characters are not allowed. Examples:

MyVar = 12.3 (valid)

My Var = 12.3 (invalid)

My_Var = 12.3 (valid)

Var1 = 12.3 (valid)

1Var = 12.3 (invalid)

A real variable can contain a real number.

9.1.10.3 Alphanumeric string

Syntax

a\$

a is a letter from "a" to "z". The name of the variable is followed by the "\$" letter to indicate that it is a string. A string can contain any number of characters. Its size is modified each time the content of the variable is modified.

It is possible to address parts of a string with the TO keyword:

A\$(4 TO 10)	returns a string with chars 4 to 10
A\$(4 TO)	returns a string with character 4 to end of string
A\$(4 TO LEN(A\$))	same result

9.1.10.4 Characters arrays

The number of dimensions is only limited by the memory size of the BASIC.

When the DIM command is called, the array is created and replaces any other DIM or VAR existing with the same name.

To erase an array you can either use the CLEAR command that erases all variables, or change the dimension of the array to 1 element with another call to DIM if you don't want to clear everything but need to release memory.

An array of which name is a\$(E1,E2,E3) and an alphanumeric variable of which name is a\$ can exist simultaneously.

A characters array contains E1*E2*E3 *... characters.

Syntax [Command]

DIM a\$(E1 [, E2 [, E3 [, ...]]])

a\$ is the name of the variable array created, its name only contains one active character of "a" until "z".

E1 is the number of characters for the first dimension. E2, E3, E4 are optional and are present if the array must have 2, 3, 4,...dimensions.

Examples:

DIM A\$(10,20)	
DIM Z(6)	
A\$(4, 3 TO 3)	same kind of access with arrays

9.1.10.5 Real arrays

When the DIM command is called, the array is created and replaces any existing array with the same name.

To erase an array you can either use the CLEAR command that erases all variables, or bring back the dimension of the array to 1 element if you don't want to clear everything but need to release memory.

In order to assign a value, type a(x, y, z)=value.

An array of which name is a(E1,E2,E3) and a real variable of which name is a CAN exist simultaneously. A real array contains E1*E2*E3 *... reals.

Syntax [Command]

```
DIM a(E1 [, E2 [, E3 [, ...]]])
```

a is the name of the array variable created, its name contains one character from "a" to "z". E1 is the number of real for the first dimension. E2, E3, E4 are optional and are present if the array must have 2, 3, 4,... dimensions.

The number of dimensions is only limited by the BASIC memory size.

Example :

```
DIM d(5,5)
d(1,6)=6
```

9.1.11 TagName variable

Syntax

TagName@

TagName is the name of the Tag. This syntax allows direct access to the Tag value. This syntax can be used for reading or writing to the Tag. This syntax should replace in many cases the GETIO SETIO commands. Only in some cases it is useful to use the GETIO or SETIO commands in order to build the Tag name in the program (If a Tag name begins with a number, it cannot be accessed in Basic using the @ syntax, instead the GETIO, SETIO function must be used).

Example: Tag1, Tag2, Tag3..

Repetitive operations can be performed with "Tag"+STR\$(I%) for example.

9.1.12 Limitations of the BASIC

- Names of integer and string variables are one character long.
- The maximum number of integer variables is 26 (names are 'a%' to 'z%').
- The maximum number of string variables is 26 (names are 'a\$' to 'z\$').
- This limitation is not applicable to real variable because real names are 200 characters long (maximum).
If more strings are required Arrays of any dimensions can be allocated.
- The eWON BASIC script is limited by the memory reserved for it (128 k).
Users have to share this memory space between the code and the data.

9.2 List of the keywords

The commands and functions used to program the eWON are listed below in alphabetical order.

9.2.1 *Syntax convention*

In the following keyword usage description, the following convention is used to represent the parameters:

Parameter	Type
E1,E2,..	Integer
S1,S2,..	String
CA	Character (if string passed, first char is evaluated)

Table 90: BASIC keywords syntax convention

9.2.2 # (*bit extraction operator*)

Syntax [function]

E1 # E2

- E1= integer word
- E2 = bit position (0 to 31)

Purpose:

The # function is used to extract a bit from an integer variable (and only from an integer).

Example:

```
i%=5      :Rem Binary 0101
a%=i%#0   :Rem a%=1
b%=i%#1   :Rem b%=0
c%=i%#2   :Rem c%=1
```

9.2.3 ABS

Syntax [function]

ABS E1

Purpose:

The function returns the absolute value of E1. E1 can be a value or a Tag name. See also "Operators priority" on page 127. If the value is negative, you have to add use ().

Example:

```
ABS (-10.4)
```

Returns 10.4

9.2.4 ALMACK

Syntax [function]

ALMACK S1, S2

- S1 is the TagName of the Tag to be acknowledged
- S2 is the UserName of the user that will acknowledge the alarm. If this field is the empty field "", then the "adm" login is assumed for acknowledgement.

Purpose:

The function acknowledges the alarm status of a given Tag.

Example:

```
ALMACK "MyTag", "TheMighty"
```

9.2.5 ALSTAT

Syntax [function]

ALSTAT S1

- S1 is the name of the Tag or TagId.

Purpose:

Returns the S1 Tag alarm status. The returned values are:

Values	Description
0	No alarm
1	Pretrigger: no active alarm but physical signal active
2	In alarm
3	In alarm but acknowledged
4	Returns to normal but not acknowledged

Table 91: Values returned by the ALSTAT command

Example:

```
a% = ALSTAT "MyLittleTag"
```

9.2.6 AND

Syntax [Operator]

E1 AND E2

Purpose:

Do a bit-wise AND between E1 and E2. Also have a look at the priority of the operators.

Examples:

```
1 AND 2
```

Returns 0

```
2 AND 2
```

Returns 2

```
3 AND 1
```

Returns 1

```
MyFirstTag@ AND 3
```

Keeps first 2 bits.

See also:

"Operators priority" on page 127, "OR" on page 158, "XOR" on page 168

9.2.7 ASCII

Syntax [function]

ASCII CA

Purpose:

The function returns the ASCII code of the first character of the chain CA. If the chain is empty, the function returns 0.

Example:

```
a% = ASCII "HOP"
```

Returns the ASCII code of the character H

See also:

"CHR\$" on page 133

9.2.8 BIN\$

Syntax [function]

BIN\$ E1

Purpose:

The function returns a string of 32 characters that represents the binary value of E1. It does not work on negative values.

Example:

```
A$= BIN$ 5
```

REM A\$ is worth " 00000000000000000000000000000000101 " after this affectation

See also:

"HEX\$" on page 145

9.2.9 BNOT

Syntax [function]

BNOT E1

Purpose:

This function returns the "bitwise negation" or one's complement of the integer E1.

Example:

```
a%=5  
b%=BNOT a%  
print BIN$(b%)
```

Will display 111111111111111111111111111111111010

See also:

"Operators priority" on page 127

9.2.10 CFGSAVE

Syntax [Command]

CFGSAVE

Purpose:

Writes the eWON configuration to flash. This command is necessary after SETSYS command on SYS, TAG or USER records because using SETSYS will modify the configuration in memory.

The modification will be effective as soon as the SETSYS XXX,"save" (where XXX stands for SYS, USER or TAG), but the config is not saved to the eWON flash file system.

See also:

"GETSYS, SETSYS" on page 141

9.2.11 CHR\$

Syntax [Function]

CHR\$ E1

Purpose:

The function returns a character string with only one character corresponding to the ASCII code of E1. E1 must be contained in the 0..255 range.

Example :

```
A$= CHR$ 48
```

REM A\$ is worth " 0 " after this affectation

```
B$=CHR$(getio(MyTag))
```

See also:

"ASCII" on page 132

9.2.12 CLEAR

Syntax [Command]

CLEAR

Purpose:

Erases all variables from the eWON. All DIM are erased. This command cannot be canceled.

9.2.13 CLOSE

Syntax [Command]

CLOSE I1

I1 is the file number (1-4)

Purpose:

This command closes the file with file number I1. If the file is opened for write, it is actually written to the flash file system. The function can be called even if the file is not opened.

See also:

"EOF" on page 137, "GET" on page 138, "OPEN" on page 154, "PUT" on page 160

9.2.14 CLS

Syntax [Command]

CLS

Purpose:

This command erases the virtual screen of the eWON, visible in the Script control page.

See also:

"PRINT - AT" on page 159

9.2.15 DAY

Syntax [Function]

DAY E1 / S1

- E1 is a date in integer format (number of seconds since 1/1/1970)
- S1 is a date in String format ("18/09/2003 15:45:30")

Purpose:

This function returns an integer corresponding to the value of the day of the month (1--31) that matches a defined time variable.

REM: Do not call the function with a float variable of value (or this would result to error "invalid parameter").

Example 1:

```
a$ = TIME$
a% = DAY a$
```

Example 2:

```
b% = getsys prg, "TIMESEC"
a% = DAY b%
```

See also:

"DOW" on page 135, "DOY" on page 135, "MONTH" on page 150

9.2.16 DEC

Syntax [Function]

DEC S1

- S1 is the string to convert from HEX to DEC

Purpose:

This function returns an integer corresponding to the hexadecimal value of parameter.

The string is not case sensitive (a023fc = A023FC). The string can be of any length.

Example :

```
A$= HEX$ (1234)
I%=DEC (A$)
```

REM Now I%=1234

See also:

"HEX\$" on page 145

9.2.17 DIM

Purpose:

The DIM function permits to create variables of array type. Two types of array are available: the characters arrays and the real arrays.

9.2.18 DOW

Syntax [Function]

DOW E1 / S1

- E1 is a date in integer format (number of seconds since 1/1/1970)
- S1 is a date in String format ("18/09/2003 15:45:30")

Purpose:

This function returns an integer corresponding to the value of the day of the week (0--6; Sunday = 0) that matches a defined time variable.

REM: Do not call the function with a float variable of value (or this would result to error "invalid parameter").

Example 1:

```
a$ = TIME$
a% = DOW a$
```

Example 2:

```
b% = getsys prg, "TIMESEC"
a% = DOW b%
```

See also:

"DAY" on page 134, "DOY" on page 135, "MONTH" on page 150

9.2.19 DOY

Syntax [Function]

DOY E1 / S1

- E1 is a date in integer format (number of seconds since 1/1/1970)
- S1 is a date in String format ("18/09/2003 15:45:30")

Purpose:

This function returns an integer corresponding to the value of the current day in the year (0-365) that matches a defined time variable.

REM: Do not call the function with a float variable of value (or this would result to error "invalid parameter").

Example 1:

```
a$ = TIME$
a% = DOY a$
```

Example 2:

```
b% = getsys prg, "TIMESEC"
a% = DOY b%
```

See also:

"DAY" on page 134, "DOW" on page 135, "MONTH" on page 150

9.2.20 **DYNDNS**

Syntax

DYNDNS

Purpose:

The command has no parameter and asks a NO-IP dynamic PPP IP address update to the Dynamic DNS server you have set in the IP address publishing part from the Callback Configuration eWON page.

It will be used to synchronize a Dynamic DNS server such as No-IP with the eWON PPP IP address.

9.2.21 **END**

Syntax [Command]

END

Purpose:

Indicates the end of the program. This command can also be used to stop the execution of a section.

If the program is in RUN mode, this command will suspend the execution until another section is ready to run (ONCHANGE, CYCLIC etc.).

Example:

```
PRINT " START "  
END  
PRINT " SUB "
```

See also:

"HALT" on page 145

9.2.22 **ERASE**

Syntax [Command]

ERASE Filename

Purpose:

Erase the specified file in the /usr directory. That means this command will not work for a different directory than the "/usr" directory. Omitting "/usr/" before the filename will result to a syntax error.

The file and directory names are case sensitive.

Example:

```
ERASE "/usr/myfile.shtm"
```

9.2.23 EOF

Syntax [function]

EOF E1

- E1 is the file number (1-4)

Purpose:

Returns 1 when end of file is reached.

EOF always returns 1 for files opened for write.

Example:

```
PRINT "open file"
OPEN "file:/usr/myfile.txt" FOR TEXT INPUT AS 1
ReadNext:
IF EOF 1 THEN GOTO ReadDone
A$ = GET 1
PRINT A$
GOTO ReadNext
ReadDone:
PRINT "close file"
CLOSE 1
```

See also:

"CLOSE" on page 133, "GET" on page 138, "OPEN" on page 154, "PUT" on page 160

9.2.24 FCNV

Syntax [function]

FCNV E1, E2

- E1 is the string to be converted to an IEEE float representation.
- E2 is the parameter for the conversion, with comporment as follows:

x==1: convert A\$(1 to 4) to a float IEEE representation with:
A\$(1)== Mantissa MSB
A\$(2)== Mantissa
A\$(3)== Mantissa LSB
A\$(4)== Exponent+ Sign
x==2: convert A\$(1 to 4) to a float IEEE representation with:
A\$(1)== Exponent+ Sign
A\$(2)== Mantissa LSB
A\$(3)== Mantissa
A\$(4)== Mantissa MSB

Table 92: Convention for converting a string to its IEEE representation

Purpose:

Converts a string to its IEEE representation on 32 bits.

Example:

```
ieee = 0.0
A$="1234"
A$(1) = Chr$(140)
A$(2) = Chr$(186)
A$(3) = Chr$(9)
A$(4) = Chr$(194)
ieee = FCNV A$,2
Print ieee
rem ieee = -34.432176
```

9.2.25 FOR NEXT STEP

Syntax

```
FOR a% = E1 TO E2 STEP E3
NEXT a%
```

- a% is an integer variable used as a counter.
- E1, E2, E3 are integer values/variables

Purpose:

The instructions between the lines containing the FOR and the NEXT are executed until a% = E2. The loop is always executed at least 1 time, even if E1 is greater than E2. During first loop execution, a% equals E1. FOR and NEXT cannot be on the same line of program. Do not exit the FOR/NEXT loop by a GOTO statement because, in this case, after a certain number of executions, the memory of the eWON is full.

Example:

```
FOR a%=10 TO 20 STEP 2
    PRINT a%
NEXT a%
```

9.2.26 GET

The GET command works completely differently if the file is opened in Binary mode or in Text mode. The file syntax has been extended in version 3 of the eWON to allow access to the serial port and to TCP and UDP socket. The command description describes operation for /usr (Text and Binary modes), COM (always binary) and TCP-UDP (always binary)

9.2.26.1 /usr Syntax [function] – Binary mode

```
GET E1, E2/S1
```

- E1 is the file number (1-8)
- E2 is the number of bytes to read from the file

Or

S1 if S1 is used, the function returns file specific information.

S1 value	Returned information
"SIZE"	File total size

Purpose:

Returns a string of char with the data read. Moves the file read pointer to the character following the last character read (or to end of file).

- Get 1, 1 will return max 1 char
- Get 1, 5000 will return max 5000 char
- Get 1 without param is equivalent to Get 1,2048

Example:

```
OPEN "file:/usr/myfile.bin" FOR BINARY INPUT AS 1
A$ = GET 1,10 REM read 10 bytes
PRINT A$
CLOSE 1
```

9.2.26.2 /usr Syntax [function] – Text mode

```
GET E1 [, E2]
```

- **E1 is the file number (1-4)**

E2 optional: buffer size. When a data is read from the file, it must be read in a buffer to be interpreted. The buffer must be able to hold at least the whole item and the CRLF at the end of the line if the item is the last of the line. The default buffer size is 1000 bytes, if your file contains items that may be bigger than 1000 bytes, you should specify this parameter, and otherwise you only have to specify the E1 parameter: file number.

Purpose:

Returns a STRING or a FLOAT according to the data read from the file. If the data read is surrounded with quotes, it is returned as a STRING, if the data read is not surrounded with quotes, it is returned as a FLOAT. The function never returns an INTEGER.

The function moves the file read pointer to the next item. For string items, the ' quote or " quote can be used.

The separator between items is the ';' character. When a CRLF (CHR\$(13)+CHR\$(10)) is found it is also skipped.

Example:

```
123;"ABC"
1.345;"HOP"
DIM A$(2,20)
DIM A[2]
OPEN "/myfile.txt" FOR TEXT INPUT AS 1
I%=1
ReadNext:
  A[I%] = GET 1
  A$(I%) = GET 1
  I% = I%+1
  GOTO ReadNext
IF EOF 1 THEN GOTO ReadDone
ReadDone:
CLOSE 1
```

9.2.26.3 COM Syntax [function] – Binary mode

```
GET E1,E2
```

```
CLOSE
```

- **E1: File number**
- **E2: maximum number of bytes to read from the serial port.**

Purpose:

Returns a string with the data read from the serial port buffer.

If there are no data to read from the buffer the returned string is empty.

If E2 is specified and the buffer contains more than E2 bytes, the function returns with E2 bytes.

If E2 is specified and the buffer contains less than E2 bytes, then the function returns with the content of the buffer.

Then function always returns immediately.

REM: Attempting to USE a serial port used by an IO server is not allowed and returns an error.

Example:

```
OPEN "COM:2,... AS 1"
A$=GET 1,100
CLOSE 1
```

9.2.26.4 TCP/UDP Syntax [function] – Binary mode

GET E1, E2

- E1 is the file number returned by OPEN.
- E2: maximum number of bytes to read from the socket.

Purpose:

Returns a string with the data read from the TCP/UDP socket.

If there are no data to read from the buffer the returned string is empty.

If E1 is specified and the buffer contains more than E1 bytes, the function returns with E1 bytes.

If E1 is specified and the buffer contains less than E1 bytes, then the function returns with the content of the buffer.

If the other party has closed the socket or if the socket is in error at the TCP/IP stack level, the function exits with error (See ONERROR on page 152)

Then function always returns immediately.

See also:

“CLOSE” on page 133, “EOF” on page 137, “OPEN” on page 154, “PUT” on page 160

9.2.27 GETFTP

Syntax [function]

GETFTP F1, F2

- F1 is the name of the file to retrieve on the FTP server.
- F2 is the name to assign to the file on the eWON

Purpose:

Retrieves a file on an FTP server

Example:

```
GETFTP "server_file_name.txt", "/usr/ewon_file_name.txt"
```

9.2.28 GETIO

Syntax [function]

GETIO S1 / E1

- S1 is the name of the tag
 - E1 is the ID of the tag
- Purpose:**

Returns the value of the S1 Tag. This value is a FLOAT.

Example 1:

```
A = GETIO "MyTag"
```

Example 2:

```
A = GETIO 12 rem if TagID =12
```

This function is equivalent A = MyTag@

Warning:

The MYTAG Basic variable is distinct than the memory Tag "MYTAG".

9.2.29 GETSYS, SETSYS

The GetSys and SetSys function are used to set or get some special parameters of the eWON. There are 5 types of parameters:

Group	Description
PRG	Program parameters like the time in milliseconds or the type of action that started the program
SYS	Edition of the eWON system parameters
COM	Edition of the eWON communication parameters
USER	Edition of the eWON users list
TAG	Edition of the eWON Tag list

Table 93: GETSYS and SETSYS parameters

Each group has a number of fields that can be read or written.

• **PRG** group fields:

Field name			Description
ACTIONID	RW	I	After execution of a scheduled action like: SendSMS SendMail PutFTP SENDTRAP TCP/UDP Connect (see OPEN command) The ACTIONID returns the ID of the action just executed. When the ONACTION event is executed, this ActionId is stored in EVTINFO. Writing to this field is useful to read the current value of an action.
ACTIONSTAT	RO	I	Current status of the action with ActionID given by ACTIONID. If ACTIONSTAT must be checked, ACTIONID must first be initialized Possible values of ACTIONSTAT are: -1: in progress -2: ID not found 0: done with success >0: finished with error = error code The eWON maintains a list with the status of the last 20 scheduled actions executed. When more actions are executed, the older status is erased and its ACTIONSTAT may return -2, meaning it is not available anymore
EVTINFO	RO	I	The value of this field is updated before executing the ONXXXXX (ONSTATUS, ONERROR, etc.), see the different ONXXXXX function for the meaning of the EVTINFO parameter
TIMESEC	RO	I	Returns the time elapsed since 1/1/1970 in seconds. (Useful for computing time differences) Warning: when you assign this value to a float variable the number is too big and rounding will occur You should use an integer variable (ex: a%) to store this value
MSEC	RO	I	Time in MSEC since eWON has booted Max value is 134217727 then it wraps to 0
RUNSRC	RO	I	When program is started, the source of the execution is given by this parameter: 1: Started from the Web site 'Script Control' window 2: Started by the FTP server because program has been updated 3: A 'GO' command has been executed from the script 4: Automatic program start at eWON boot

Table 94: PRG group fields

PPPIP	RW	S/I	This parameter returns the string corresponding to the current PPP IP address. When the eWON is offline, the value returned is "0.0.0.0". When the eWON is online the value returned is the dotted IP address allocated for the PPP connection The parameter can be written in order to disconnect the eWON. The only value accepted when writing in this parameter is 0 (setsys prg, "PPPIP", 0)
TRFWD	RW	S	Transparent forwarding IP address. The parameter can be used to write or read the routing parameter. The parameter is only active when the PPP connection is established
SERNUM	RW	S	Returns a string with the eWON serial number string
PRIOH	RW	I	Used for changing the script priority Currently not documented
PRION	RW	I	Used for changing the script priority Currently not documented
PRIOT	RW	I	Used for changing the script priority Currently not documented
RESUMENEXT	R/W	I	Controls the OnError action. Possible values are a combination of: 1: Resume Next mechanism is enabled 4: Do not execute ONERROR 8: Do not show error on virtual screen This parameter is useful when testing the existence of a variable, file or other Example: Testing the existence of a file can be done by opening it and see if it generated an error The Error result is accessible through LSTERR
LSTERR	RO	I	Contains the code from the last Basic error that occurred

Table 94: PRG group fields

Notes:

RO means read only

R/W means read/write

I,R,S means Integer, real, string

• **SYS fields group:**The fields edited with this group are the one found in the *config.txt* file under the section *System*.

The fields are described in the section "Configuration fields".

• **COM fields group:**The fields edited with this group are the one found in the *comcfg.txt*. The fields are described in the section "Configuration fields".

It is possible to tune the modem detection too. See Com Section on page 172.

• **TAG fields group:**The fields edited with this group are the one found in the *config.txt* file under the section *TagList*.

The fields are described in the section "Configuration fields".

See *Tag Section* on page 175.• **USER fields group:**The fields edited with this group are the one found in the *config.txt* file under the section *UserList*.

The fields are described in the section "Configuration fields".

See *User Section* on page 178.

- A block must be loaded for edition with the SETSYS command and a special field called "load". According to the source, this block will be either the eWON system configuration, the eWON COM configuration, or one Tag configuration, or one user configuration.
- Then each field of this configuration can be accessed by the GETSYS or SETSYS commands.
This edition works on the record loaded values but does not actually affect the configuration.
- When edition is finished, the SETSYS command is called with a special field called "save" and the edited block is saved (this is only necessary if the record has changed).
At that time the record edited content is checked and the configuration is actually updated and applied.
- The CFGSAVE command can be called to actually save the updated configuration to flash.
- SETSYS TAG,"load",XXXXXX

The TAG load case is particular because it allows to load a Tag defined by its name, its ID or its Index.

If there are 6 Tags defined in the config, each Tag can be accessed by its index (0 to 5), its ID (the first item of a Tag definition when reloading the config.txt file, the ID of a Tag is never reused during the live of a configuration until the eWON is formatted) or finally by its name.

Method	XXX param	Example	Example explanation
Tag name access	Tagname	Setsys Tag,"load","MyTagName"	Loads Tag with name MyTagName
Index access	-Index	Setsys Tag,"load",- 4	Loads Tag with index 4
TagId access	Id	Setsys Tag,"load",50	Loads Tag with id 50

Table 95: Setsys Tag,"load" examples

• **Recognized field values per group**

The fields values are the same fields as those returned by the FTP get config.txt command. See the Chapter "Files Format" on page 120 for a list of these fields.

Syntax

GETSYS SSS, S1

SETSYS SSS, S1, S2 / E2

- SSS is the source block: PRG, SYS, TAG, USR - This parameter must be typed as is (it could not be replaced by a string)!
- S1 is the field name you want to read or modify. S1 can be the action "load" or "save"
- S2 /E2 is the value to assign to the field, of which type depends on the field itself

Example :

```
A% = GETSYS PRG, "TIMESEC"
REM Suppose Tag_1 exists and is memory Tag
SETSYS TAG,"load","Tag_1"
A$ = GETSYS TAG,"Name" : REM A$="Tag_1"
SETSYS TAG,"ETO","ewon actl@ewon.biz" : REM EmailTo field of Tag_1
SETSYS TAG,"save" : REM save data in the config => update Tag_1
SETSYS TAG,"Name","Tag_2"
SETSYS TAG,"save" : REM update or create Tag_2 with Tag_1 cfg
```

See also:

"CFGSAVE" on page 133, "Configuration Fields" on page 170

9.2.30 GO

Syntax [Command]

GO

Purpose:

Start program execution (*RUN*). This is equivalent to clicking *RUN* in the script control window.

This command is mainly useful for remote eWON operation through the use of REMOTE.BAS FTP transfer.

See also:

"HALT" on page 145), ("REBOOT" on page 161)

9.2.31 GOSUB RETURN

Syntax

GOSUB Label

Label:Expression

RETURN

Purpose:

When the GOSUB line is executed, the program continues at "Label" line. Then the program executes up to the RETURN line.

The RETURN command modifies the program pointer to the line immediately following the GOSUB Line.

IMPORTANT: if the gosub line contains instruction after the GOSUB command, they won't be executed on return.

It is possible to create a new section containing the Label. Sections are useful in order to divide the program into smaller code snippets and help the reader to get a clear view of the software. At the end of every section there is an invisible END but jumps are possible from section to section.

Example:

```
GOSUB NL3
PRINT " End "
END
NL3 : PRINT " Beginning "
RETURN
REM Display " Beginning " then " End "
GOSUB NL3 :print "Never"
PRINT " End "
END
NL3 : PRINT " Beginning "
RETURN
REM Display " Beginning " then " End " => "Never"
is never printed
```

9.2.32 GOTO

Syntax [Command]

GOTO Label

Purpose:

The execution of the program continues to the line indicated by Label.

The GOTO command also allows starting the program without erasing all variables.

The Label statement cannot be empty.

Example:

```
GOTO Label
Print " Hop "
REM the program continues at line Label (Hop is not printed)
Label:
```

9.2.33 HALT

Syntax [Command]

HALT

Purpose:

Stops program execution. This is equivalent to clicking **STOP** in the script control window.

This command is mainly useful for remote eWON operation through the use of REMOTE.BAS FTP transfer.

See also:

"GO" on page 144, "REBOOT" on page 161

9.2.34 HEX\$

Syntax [function]

HEX\$ E1

Purpose:

The function returns a chain of 8 characters that represents the hexadecimal value of the E1 number.

Example:

```
a$= HEX$ 255
REM A$ is worth " 000000FF " after this affectation
```

See also:

"BIN\$" on page 132

9.2.35 IF THEN ELSE ENDIF

This sequence of commands now supports two different syntaxes: the short IF syntax and the long IF syntax.

9.2.35.1 Short IF Syntax

```
IF N THEN EXPRESSION1 [ELSE EXPRESSION2 [ENDIF]]
```

Purpose:

The condition is the result of an operation returning an N integer. If N is 0, the condition is considered as false and the eWON executes at the following line or to the ELSE "expression2" if present. If N is different from 0, the condition is considered as true and the eWON executes "expression1". If more than one instruction has to be executed, separate them with ':'. If N is an expression or a test, use ().

The ELSE Expression2 is optional and the finishing ENDIF is also optional.

IMPORTANT the short IF syntax is used as soon as an item is found after the THEN statement.

Even putting a REM statement on the IF N THEN line will make the eWON consider it as a short IF statement.

9.2.35.2 Long IF syntax

```
IF N THEN
```

```
    Expression1
```

```
ELSE
```

```
    Expression2
```

```
ENDIF
```

The ELSE Expression2 is optional but ENDIF is mandatory.

You can mix short and long IF syntax in your code, but don't forget that anything typed after the THEN statement will lead to a short IF syntax interpretation.

Example:

```
IF (a<10) THEN PRINT"A is lower than 10": SETIO"MyTag",1
IF (a<10) THEN
PRINT"A is lower than 10": MyTag@=1
ELSE
    PRINT"A is bigger than 10": MyTag@=0
ENDIF
```

9.2.36 INSTR

Syntax [Function]

```
INSTR I1, S1, S2
```

- I1 is the index in the string to search (valid value goes from 1 to LEN S1)
- S1 is the string to be search for S2
- S2 is the string to search for in S1

Purpose:

The function returns an integer equal to the position of string S2 in string S1.

The returned index is 0 based. If string S2 is not contained in S1, the function returns 0.

The I1 parameter should be 1 to search the whole S1 string. If I1 is >0 then string S1 is searched starting at offset I1.

The value returned is still referenced to the beginning of S1, example:

INSTR 1, "AAABBC", "BB" = 4 and INSTR 3, "AAABBC", "BB" = 4 also.

9.2.37 INT

Syntax [function]

```
INT F1
```

Purpose:

Extract the integer part of the number. There is no rounding operation.

Example 1:

```
A = INT(10.95)
REM A equals 10.00
```

Example 2:

```
A% = 10.95
REM A equals 10 --- automatic type conversion
```

9.2.38 IOMOD

Syntax [function]

```
IOMOD S1 / E1
```

- S1 is the name of the Tag (A = IOMOD "MyTag")
- E1 is the ID of the Tag (A = IOMOD 12 : rem if TagID =12)

Purpose:

Returns '1' if the S1 Tag has been modified in the eWON since the last call to the IOMOD command.

The call to this function resets Tag change internal flag to 0. i.e. if the variable doesn't change anymore, the next call to IOMOD will return 0.

You can achieve an equivalent behavior with the use of ONCHANGE instruction.

Example:

```
a% = IOMOD " MYTAG "
IF a% THEN PRINT " mytag has changed "
```

See also:

"ONCHANGE" on page 152

9.2.39 IORCV

Syntax [function]

IORCV S1

or

IORCV S1, I1

- S1 is the STRING IOServerName
- I1 is an additional parameter (= 0 OR = 1 OR = -1)

Purpose:

Function for reading IO server response to an IOSEND request.

• **First case:**

- a% in examples below represents the Request number returned by IOSEND.
- a% is the result of the IOSEND command.

Examples:

```
a$ = IORCV a%
```

```
a$ = IORCV a%,0
```

a\$="XXXXXXXX"	where XXXXXXXXX is the result of the request
a\$="#FREE"	slot a% is free
a\$="#RUN"	slot a% is in progress
a\$="#ERR"	slot a% is done with error

• **Second case:**

```
a$ = IORCV a%,-1
```

Same as for a\$=IORCV a%,0 but the slot is not freed if a request is done.

• **Third case:**

```
a$ = IORCV a%,1
```

Returns the status of the IORCV command in INTEGER format.

The returned status can contain the following values:

b%= -2	slot a% is free
b%= -1	slot a% is in progress
b%= 0	slot a% is done with success
b% > 0	slot a% is done with error
b% < -2	slot a% is done with error - code type: warning Such warning codes mean "Read failed" on the serial link These warnings are flagged as internal and thus are not added in the event log The warning codes can be very long; ie. -536893114

See also:

"IOSEND" on page 148

9.2.40 IOSEND

Syntax [function]

IOSEND S1, S2, S3

Purpose:

Sends a request by using the IO server's protocol.

Parameters are:

- **STRING IOServerName:** IO Server name as it appears in the Tag configuration page
- **STRING Address:** Slave address as described in the eWON User manual for each IO server section
- **STRING IoCommand:** Array of bytes with a protocol command, the content depends on the IO server.

Returns a request number that must be used in IORCV for reading the response to the request.

Note:

The request result is read by using the IORCV function and uses a polling mechanism.

That means that you need to use IORCV in order to check with the request received with IOSEND that the slot is free.

There are three transmission slots available, using IORCV allows you to free them before the three slots are busy.

Requests are interlaced with gateway requests sent to the IO server and with normal IO server polling operations.

Example:

```
a% = IOSEND IOServerName,Address,IoCommand
```

See also:

"IORCV" on page 147

9.2.41 LEN

Syntax [function]

LEN S1

Purpose:

The function returns the number of characters in a string.

Example:

```
a$= "Hop "  
A% = LEN A$  
REM a% equal 3
```

9.2.42 LOGEVENT

Syntax [command]

LOGEVENT S1 [,S2]

- S1 is the phrase to log
- S2 is the type of logging. This parameter is optional and can take the following ranges of values:

Range of values	Description
0 .. 99	Error
-99 .. -1	Warning
100 .. 199	Trace

Table 96: LOGEVENT - ranges of values

If the logging level is not specified, it is considered to be an error.

Purpose:

Appends an event to the log file. The current time is automatically used for event logging.

Example:

```
logevent "Save this in log", 120
REM Would append 978353046;"01/01/2001 12:44:06";"Save this in log" to the
log file.
```

9.2.43 LOGIO

Syntax [command]

LOGIO S1

Purpose:

Force historical logging of Tag name given by S1.

The Tag must have historical logging enabled

(Warning: not available on all eWON's versions - see "Table of comparisons between eWON types" on page 233).

The point is logged at the time the LOGIO command is issued with its current value.

Note: If the Tag is configured for historical logging with logging dead band equal to -1 and time interval equal to 0, no point will be logged automatically and it is possible to program a purely scripted logging.

Example:

```
LOGIO "mytag"
```

9.2.44 LTRIM

Syntax[Command]

LTRIM S1

- S1 is a string.

Purpose:

LTRIM returns a copy of a string with the leftmost spaces removed.

Example:

```
b$ = LTRIM a$
```

See also:

"RTRIM" on page 162

9.2.45 MOD

Syntax [Operator]

E1 MOD E2

Purpose:

Compute the remainder of the division of E1 by E2

Example:

```
1 MOD 2
REM returns 1
2 MOD 2
REM returns 0
```

See also:

"Operators priority" on page 127

9.2.46 MONTH

Syntax [Function]

MONTH E1

- E1 is a date in integer format (number of seconds since 1/1/1970)
- S1 is a date in String format ("18/09/2003 15:45:30")

Purpose:

This function returns an integer corresponding to the value of the month (1--12) that matches a defined time variable.

Warning: Do not call the function with a float variable of value (or this would result to error "invalid parameter").**Example 1:**

```
a$ = TIME$
a% = MONTH a$
```

Example 2:

```
b% = getsys prg, "TIMESEC"
a% = MONTH b%
```

See also:

"DAY" on page 134, "DOW" on page 135, "DOY" on page 135

9.2.47 NOT

Syntax [function]

NOT E1

Purpose:

The function returns '1' if E1 is equal to '0' otherwise the function returns 0.

Example:

```
IF NOT a% THEN PRINT " A% is worth 0 "
```

See also:

"Operators priority" on page 127

9.2.48 NTPSync

Syntax [function]

NtpSync

Purpose:

Posts a request for clock re synchronization (even if this feature is disabled in the configuration).

9.2.49 ONxxxxxx

There are some ONxxxxxx commands listed below. These commands are used to register a BASIC action to perform in case of special conditions. For every ONxxxxxx command, the action to execute is a string that is used as a BASIC command line. When the condition occurs, the command is queued in an execution queue and is executed when its turn comes. These functions are:

ONxxxx command	Description
ONTIMER	Executed when one of the timers expires
ONCHANGE	Executed when a Tag value changes
ONALARM	Executed when a Tag alarm state changes
ONERROR	Executed when an error occurs during BASIC execution
ONSTATUS	Executed when a scheduled action is finished (success or failure)
ONPPP	Executed when the PPP connection goes online or offline

Table 97: the various "ONXXXX" functions

When the command line programmed is executed, a special parameter is set in SETSYS PRG,"EVTINFO". The value of the parameter depends on the ONxxxxxx function and it can be checked with the GETSYS command.

Warning:

For all ONxxxx command, if the last parameter is omitted, the action is canceled.

Example:

```
ONTIMER 1
REM will cancel any action programmed on TIMER 1.
```

See also:

"GETSYS, SETSYS" on page 141, ONxxxxxx (following chapters)

9.2.50 ONALARM

Syntax [command]

ONALARM S1,S2

- S1 is the Tag to scan for alarm state change
- S2 is the command line to execute in case of alarm state change.

Purpose:

Executes the S2 command line when alarm state on Tag with Tag name given by S1 changes. The EVTINFO parameter (see GETSYS page 141) is set to the Tag id when command is called. Note: ONALARM will execute the command when the alarm status gets the value "2", that means that ONALARM **DOES NOT DETECT** the "pre trigger" status (value=1).

Example:

```
ONALARM "MyTag", "goto MyTagAlarm"
```

See also:

"ALSTAT" on page 131, "GETSYS, SETSYS" on page 141, "ONxxxxxx" on page 151, "ONCHANGE" on page 152

9.2.51 ONCHANGE

Syntax [command]

ONCHANGE S1, S2

- S1 is the Tag name to scan for value change
- S2 is the command line to execute in case of value change.

Purpose:

Executes S2 command line when the value of Tag with Tag name given by S1 changes. The EVTINFO parameter (see GETSYS page 141) is set to the Tag id when command is called.

Example:

```
ONCHANGE "MyTag", "goto MyTagChange"
```

See also:

"GETSYS, SETSYS" on page 141, "ONxxxxx" on page 151

9.2.52 ONERROR

Syntax [command]

ONERROR S1

- S1 is the command line to execute when an error occurs during program execution.

Purpose:

The EVTINFO parameter (See GETSYS, SETSYS on page 141) is set to the code of the error.

Example:

```
ONERROR "goto TrapError"
```

See also:

"GETSYS, SETSYS" on page 141, "ONxxxxx" on page 151

9.2.53 ONPPP

Syntax [command]

ONPPP S1

- S1 is the command line to execute when the PPP connection goes online or offline.

Purpose:

The EVTINFO parameter (see GETSYS page 141) is set to one of the following values:

EVTINFO Value	Situation
1	The PPP connection has gone ONLINE
2	The PPP has gone OFFLINE

Table 98: ONPPP - EVTINFO values

Example:

```
ONPPP "goto PppAction"
END
PppAction:
I%=GETSYS PRG,"EVTINFO"
IF I%=1 then
PRINT "Online with address";GETSYS PRG,"PPPPIP"
ELSE
PRINT "PPP Going offline"
ENDIF
END
```

See also:

"GETSYS, SETSYS" on page 141, "ONxxxxx" on page 151

9.2.54 ONSMS

Syntax [command]

ONSMS S1

- S1 is the command line to execute when eWON receives a SMS.

Purpose:

A typical use of the ONSMS syntax is allowing eWON to send a read SMS receipt to the SMS sender.

You can read the received SMS with GETSYS PRG function with:

- **smsRead:**
hold 1 if there is a new SMS (reading smsRead load the other parameters)
hold 0 if the SMS queue is empty
- **smsFrom:**
String holding the phone number of the sender
- **smsDate:**
String holding the Date of SMS reception
- **smsMsg:**
String holding the SMS message

Example:

```
InitSection:
ONSMS "Goto HSms"
HSms:
a% = getsys prg,"SmsRead"
if (a%<>0) then
s% = s%+1
print "SMS Nr: ";s%
f$ = getsys prg,"smsfrom"
print "From: ";f$
print getsys prg,"smsdate"
a$ = getsys prg,"smsmsg"
print "Message: ";a$
b$ = f$+",gsm,0"
c$ = "Received message: "+a$
sendsms b$,c$
goto HSms
endif
end
```

9.2.55 ONSTATUS

Syntax [command]

ONSTATUS S1

- S1 is the command line to execute when a scheduled action is finished.

Purpose:

The EVTINFO parameter (see GETSYS page 141) is set to the ACTIONID of the finished action when command is called. This function can be used to track success or failure of scheduled actions.

Example:

```
ONSTATUS "goto Status"
```

See also:

"GETSYS, SETSYS" on page 141, "ONxxxxx" on page 151, "PUTFTP" on page 161, "SENDMAIL" on page 162, "SENDSMS" on page 163, "SENDTRAP" on page 164

9.2.56 ONTIMER

Syntax [command]

ONTIMER E1,S1

- E1 is the timer number (see TSET page 167)
- S1 is the command line to execute when timer expires.

Purpose:

Executes S1 command line when E1 expires.

The EVTINFO parameter (see GETSYS page 141) is set to the timer number when command is called.

Example :

```
ONTIMER 1,"goto Timer1"
ONTIMER 1, "LOGIO 'mytag' "
```

See also:

"GETSYS, SETSYS" on page 141, "ONxxxxx" on page 151, "TSET" on page 167

9.2.57 OPEN

9.2.57.1 Introduction to file management

Files accessed in BASIC can be of 4 different types:

- Files from the /usr directory (or from other directories such as /sys)
- Serial communication link
- TCP or UDP socket
- Export Block Descriptor

9.2.57.2 OPEN general syntax

There are two different modes of operation for the file access:

- **BINARY mode: file is read by blocks of bytes**
- **TEXT mode: files are read or written as CSV files**

See the "GET" on page 138 and "PUT" on page 160 commands for a detailed difference between the BINARY and TEXT mode outputs.

There are 3 operation types:

Parameter value	Description
INPUT	The file must exist it is opened for a read only operation The file pointer is set to the beginning of the file
OUTPUT	The Path must exist. If the file exists it is erased first. The file is opened for write only operation
APPEND	The Path must exist. The file must not exist. If the file does not exist, it is created (like with OUTPUT type), if the file exists, it is opened and the write pointer is located at the end of the file. The file is opened for write only operation

Table 99: OPEN read and write operations parameters

When binary mode is used, the data written to the file are strings of characters that are considered as stream of bytes.

The GET command returns the amount of bytes requested.

When Text mode is used, the operation is completely different: the PUT operation is more like a PRINT command directed to file, the data are formatted as text, and each data is separated by a ';' in the output file (strings are exported between quotes).

The GET command works like a READ command, the file is read sequentially and each GET returns one of the ';' separated element, the type of the data returned depends on the type of data read.

In both modes, files are read sequentially until end of file is reached. End of file can be tested with the EOF function.

The eWON user flash file system allows up to 8 files to be simultaneously opened for read (even twice the same file), and 1 file opened for write. If a file is opened for read it cannot be opened for write at the same time (and vice versa).

Running the program will close any previously opened files (not GOTO).

9.2.57.3 Different File/stream types

9.2.57.3.1 FILE open /usr

Syntax [command]

`OPEN S1 FOR BINARY|TEXT INPUT|OUTPUT|APPEND AS E1`

E1 is the file number. After the OPEN operation, the file is referenced by its file number and not by its file name.

There are 8 file numbers available. Once a file number is assigned to a file, it is allocated to that file, until the CLOSE command is issued.

S1 describes the access to a file that is located on eWON directories. S1 must respect the following syntax:

- **"file:directory/filename"**

This allows to read or write files in the /usr and /sys directories. You will not be able to access the files in the root (virtual files like config.txt) with this command.

Example	Comment
<pre>OPEN "file:/usr/test.dat" FOR BINARY INPUT AS 1 A\$=GET 1,4 CLOSE 1</pre>	<p>Opens file 1 Reads 4 bytes</p>
<pre>OPEN "file:/sys/test.dat" FOR BINARY INPUT AS 1 A\$=GET 1,4 CLOSE 1</pre>	<p>Opens file 1 Reads 4 bytes</p>

- If S1 does not begin by "file:", "tcp", "com", or "exp", then the file will be considered as being part of the /usr directory.

Example	Comment
<pre>OPEN "test.dat" FOR BINARY INPUT AS 1 A\$=GET 1,4 CLOSE 1</pre>	<p>Open the /usr/test.dat file Reads 4 bytes</p>

This syntax was the old (ver 3) syntax and is kept for compatibility purpose.

9.2.57.3.2 TCP stream open Syntax [command]

`OPEN S1 FOR BINARY INPUT|OUTPUT AS E1`

Note:

This command works only with BINARY

S1 must respect the following syntax:

E1 is the file number. After the OPEN operation, the file is referenced by its file number and not by its file name.

There are 8 file numbers available. Once a file number is assigned to a file, it is allocated to that file, until the CLOSE command is issued.

"tcp:address:port"

Address can be a dotted IP address like 10.0.0.1 or a valid resolvable internet name like ftp.ewon.be

Port must be a valid port number from 1 to 65535.

WARNING - scheduled action: when the OPEN command is used to open a TCP connection, the command returns before the connection is actually opened. A scheduled action is posted because opening the socket may require a dial out or take up to more than a minute, and the BASIC cannot be stopped during that time.

In order to know if the connection is established, the user has 2 options:

- Check the scheduled action status by checking the PRG,ACTIONSTAT (See GETSYS, SETSYS on page 141).
 - Read the file with GET: as long as the file is not actually opened, the function returns #CLOSED#.
- When the function stops sending #CLOSED# the file can be read and written for socket operations.

Example :

Example	Comment
<pre>OPEN "tcp:10.0.0.1:25" FOR BINARY OUTPUT AS 1 PUT 1,CHR\$(13)+CHR\$(10) A\$=GET 1 CLOSE 1</pre>	<p>Opens socket to 10.0.0.1 port 25 for read access. Write CRLF then read response.</p>

9.2.57.3.3 COM port open Syntax [command]

OPEN S1 FOR BINARY INPUT|OUTPUT AS E1

Note:

This command works only with BINARY

S1 will be as follows:

"com:n,b,dpsh"

- where n is 1 to 4 (the port number, 1 is Front panel serial port, 2 is Modem Port)
- where b is the baud rate
- where d is the number of bits "7" or "8"
- where p is the parity: "e","o" or "n"
- where s is the number of stop bit "1" or "2"
- where h is the handshaking "h": half duplex, "r": yes Rts/Cts, "n": No

This command will open the serial port to port 1 or 2 with the given line parameters.

The configuration can be accessed using *GETSYS SYS* & *SETSYS SYS* to update the serial configuration and to enable/disable Modbus if needed.

E1 is the file number. After the OPEN operation, the file is referenced by its file number and not by its file name.

There are 8 file numbers available. Once a file number is assigned to a file it is allocated to that file until the CLOSE command is issued.

REM: Attempting to USE a serial port used by an IO server is not allowed and returns an error.

9.2.57.3.4 EXP export bloc descriptor open Syntax [command]

```
OPEN S1 FOR TEXT|BINARY INPUT AS E1
```

Note:

This command works only with INPUT

S1 will be as follows: "exp:XXXXX", where XXXXX is an Export Block Descriptor.

E1 is the file number. After the OPEN operation, the file is referenced by its file number and not by its file name.

There are 8 file numbers available. Once a file number is assigned to a file it is allocated to that file until the CLOSE command is issued. When the export block has been read (or not if you close before end) you must call CLOSE to release memory.

Warning: You cannot use the PUT command with a EXP: file

Example 1:

```
OPEN "exp:$dtAR $ftT" FOR TEXT INPUT AS 1
Loop:
A$ = Get 1
PRINT A$
If A$ <>" " then GOTO Loop
CLOSE 1
```

In that case the "a\$ = get 1" can be called until it returns an empty string to read the content of the Export Block Descriptor; the data are then read by blocks of maximum 2048 bytes. If you want to reduce or increase that size, you can call "a\$ = get 1,y", where y is the maximum number of bytes you want the function to return (do not put y=0).

Example 2:

```
OPEN "exp:$dtUF $ftT $fn/myfile.txt" FOR TEXT INPUT AS
1
A$ = Get 1
PRINT A$
CLOSE 1
```

See also:

"CLOSE" on page 133, "EOF" on page 137, "GET" on page 138, "PUT" on page 160

9.2.58 OR

Syntax [Operator]

E1 OR E2

Purpose:

Does a bit-by-bit OR between the 2 integers E1 and E2.

WARNINGS:

- When executed on float elements (float constant or float variable), the OR functions returns the logical OR operation.
- When executed on integer elements (integer constant or integer variable - like i%), the OR function returns the bitwise OR operation
- This is NOT true for AND and XOR
- This is historical and is left for compatibility with existing programs

Examples:

```
1 OR 2 REM returns 3
```

```
2 OR 2 REM returns 2
```

```
3 OR 1 REM returns 3
```

• Logical OR:

```
var1=0.0
var2=0.0
ORResult = var1 OR var2
Print ORresult
rem ORResult = 0.0
```

```
var1=0.0
var2=12.0
OR Result = var1 OR var2
Print ORresult
rem ORResult = 1.0
```

See also:

"Operators priority" on page 127, "AND" on page 131, "XOR" on page 168

9.2.59 PI

Syntax [function]

PI

Purpose:

The function returns 3.14159265

9.2.60 PRINT - AT

Syntax [Command]

PRINT CA

This command displays the text CA followed by a new line.

PRINT CA;

This command displays the text CA without a new line.

PRINT AT E1, E2 CA

This command displays the text CA at the E1 column and at the E2 line.

PRINT CA1;CA2 [;CA3 . . .]

Display the CA1, CA2 text etc. one following the other (don't pass to next line).

Purpose:

The eWON has a virtual "screen" that can be used in order to inspect the content of values while the program is running, or in order to debug an expression...

Example:

```
PRINT " HOP1 "; HOP2 "
```

See also:

"CLS" on page 134

9.2.61 PRINT

Syntax [Command]

PRINT #x, CA

With x defined as follows:

Value	Description
0	User's WEB page
1	Virtual screen

Table 100: valid values for print redirection

CA is described in the PRINT description above.

Purpose:

The PRINT command sends output to the virtual screen. With the PRINT # command, output can be routed to another destination.

When running ASP code, the print command can be used to build the content of the page sent to the user.

If you print to Web page, the Print command add a "
" at the end of line to pass to the next line.

If you don't want to pass to next line, you need to add a ";" (semicolon) at your Print.

Example:

PRINT A\$;

Example:

```
PRINT #0,A$ REM sends A$ to the user's web page
PRINT #1,A$ REM works like PRINT A$ by sending to the virtual screen.
```

9.2.62 PUT

The put command works completely differently if the file is opened in Binary mode or in Text mode. The file must be opened for OUTPUT or for APPEND operation (APPEND for /usr files only).

• COM, TCP-UDP, /usr

The file syntax has been extended in version 3 of the eWON to allow access to the serial port and to TCP and UDP socket.

The command description describes operation for /usr (Text and Binary modes), COM (always binary) and TCP-UDP (always binary)

9.2.62.1 File Syntax[Command] – Binary mode

```
PUT E1, S1[;S2...]
```

- E1 is file number (1-8)
- S1 is the string of char to append to the file. The number of bytes written depends on the length of the string.
- S2...: (optional) additional data to write

Important: the delimiter between the file number and the first item is a ',' but the separator between the first item and the optional next item is a ';'. This is close to the PRINT syntax.

The length of a BASIC line limits the number of items.

Example:

```
OPEN "/myfile.bin" FOR BINARY OUTPUT AS 1
PUT 1,"ABCDEF";"GHIJKLMN"
CLOSE 1
REM Now reopens and append
OPEN "/myfile.bin" FOR BINARY APPEND AS 1
PUT 1,"OPQRSTUVWXYZ"
CLOSE 1
```

9.2.62.2 File Syntax[Command] – Text mode

```
PUT E1, V1[;V2...][;]
```

- E1 is file number (1-8)
- V1 is an element of type STRING, INTEGER or FLOAT
- V2...optional: additional data to write (STRING, INTEGER or FLOAT)

The data are converted to text before being written to file. If data is of STRING type it is written between quotes ("), otherwise not.

A ',' is inserted between each data written to the file.

If the PUT command line ends with a ';', the sequence of data can continue on another BASIC line. If the PUT command line ends without the ';' character, the line is considered as finished and a CRLF (CHR\$(13)+CHR\$(10)) is added at the end of the file.

Example:

```
OPEN "/myfile.txt" FOR TEXT OUTPUT AS 1
PUT 1,123;"ABC";
PUT 1,"DEF"
PUT 1,345.7;"YYY";"ZZZ"
CLOSE 1
```

Produces this file:

```
123;"ABC";"DEF"
345.7;"YYY";"ZZZ"
```

REM: there is a CRLF at the end of the last line, PUT 1,345.7;"YYY";"ZZZ"; would avoid that.

9.2.62.3 COM Syntax[Command] – Binary mode

PUT 1, S1

- S1: string of data to write to serial port.

Purpose:

Writes the S1 string to the serial port. The function returns only after all the data have been actually sent.

Warnings:

- The string can contain any byte by using the CHR\$ function.
- Serial port cannot be used by an IO server in the same time, or it would result to a "IO Error".

9.2.62.4 TCP/UDP Syntax[Command] – Binary mode

PUT E1, S1

- E1: is the file number returned by the OPEN function.
- S1: string of data to write to the socket.

Purpose:

Writes the S1 string to the socket

The function returns only after all the data have been actually transferred to the stack.

Warnings:

- The socket must be opened. The OPEN command returns immediately but generates a scheduled action. The PUT command will generate an IO error until the socket is actually opened (See OPEN on page 154).
- When data are transferred to the TCP/IP stack, it does not mean that the data have been received by the socket end point. It may take minutes before the data are considered as undeliverable and the socket is put in error mode.
- The string can contain any byte by using the CHR\$ function.

See also:

"CLOSE" on page 133, "EOF" on page 137, "GET" on page 138, "OPEN" on page 154.

9.2.63 PUTFTP

Syntax[command]

PUTFTP S1, S2

- S1 is the destination file name
- S2 is the destination file content. This content follows a special syntax allowing including EXPORT_BLOCK_DESCRIPTOR contents.

See Export block descriptor on page 192

Purpose:

This command posts a scheduled action request for a PUTFTP generation.

When the function returns, the GETSYS PRG,"ACTIONID" returns the ID of the scheduled action and allows tracking this action. It is also possible to program an ONSTATUS action that will be called when the action is finished (with or without success).

Example:

```
REM Post an FTP file to the configured FTP server with event log
PUTFTP "/ewon1/events", "$dtEV"
```

The path /ewon1/ specified is the path on the server side. It has nothing to do with the path on the /usr/ eWON side!

See also:

"GETSYS, SETSYS" on page 141, "ONxxxxxx" on page 151, "ONSTATUS" on page 153.

9.2.64 REBOOT

Syntax [Command]

REBOOT

Purpose:

This Basic keyword provides a very easy way to reboot eWON.

A typical use of this command is by simply entering it into a file you name "remote.bas" then saving locally and uploading this file on the eWON FTP site to replace the existing remote.bas file. eWON then directly reboots.

9.2.65 REM

Syntax [command]

REM free text

Purpose:

This command enables the insertion of a line of comment in the program. The interpreter does not consider the line.

Example:

```
PRINT a%
REM we can put whatever comment we want here
a%=2: REM Set a% to 2
```

9.2.66 RTRIM

Syntax[Command]

RTRIM S1

• S1 is a copy of a string.

Purpose:

RTRIM returns a copy of a string with the rightmost spaces removed.

Example:

```
b$ = RTRIM a$
```

See also:

"LTRIM" on page 149

9.2.67 SENDMAIL

Syntax[command]

SENDMAIL S1, S2, S3, S4

- S1 is the E-mail address of the recipients (TO). Multiple recipients can be entered separated by ','.
- S2 is the E-mail address of the recipient *Carbon Copies* (CC). Multiple recipients can be entered separated by ','.
- S3 is the subject of the message.
- S4 is the content of the message.

Purpose:

This command posts a scheduled action request for an Email generation. When the function returns, the GETSYS PRG, "ACTIONID" returns the ID of the scheduled action and allows tracking this action. It is also possible to program an ONSTATUS action that will be called when the action is finished (with or without success). The S4 message content follows a special syntax that allows sending attachments and inserting Export data inside the content itself. (See also chapter "Export block descriptor" on page 192). The content field (S4) syntax can content any number of [EXPORT_BLOCK_DESCRIPTOR], these blocks will be replaced by their actual content.

Example:

```
S4 = "Event Log data [$dtEV] And a real time table: [$dtRL $ftT $tnMyTag]"
Rem will generate an Email with [$dtEV] and [$dtRL...] replaced by the actual data.
```

If instead of putting [EXPORT_BLOCK_DESCRIPTOR] you put &[EXPORT_BLOCK_DESCRIPTOR], then the same data is attached to the Email. The position in the S4 field where the &[...] is placed does not matter, the attachment &[...] descriptor will NOT appear in the content itself, but will produce the given attachment.

Example:

```
S4 = "Event Log data are attached to this mail &[&$dtEV]"
Rem will generate an Email with "Event Log data are attached to this mail " as content
and an attachment with the events log.
```

See Email on alarm configuration on page 65 for the syntax of this field.

Example:

```
SENDMAIL "ewon@actl.be", "", "Subject", "Message"
```

See also:

"GETSYS, SETSYS" on page 141, "ONxxxxx" on page 151, "ONSTATUS" on page 153.

9.2.68 SENDSMS

Syntax[command]

SENDSMS S1,S2

- S1 is the SMS recipients list. Please refer to chapter "SMS on alarm configuration" on page 65, for syntax of this field.
- S2 is the content of the message (maximum 140 characters).

Purpose:

This command posts a scheduled action request for an SMS generation.

When the function returns, the GETSYS PRG,"ACTIONID" returns the ID of the scheduled action and allows tracking this action.

It is also possible to program an ONSTATUS action that will be called when the action is finished (with or without success).

Example:

```
REM send an SMS to 2 recipients.
D$ = "0407886633,ucp,0475161622,proximus;"
D$ = D$+"0407886634,ucp,0475161622,proximus"
SENDSMS D$, "", "Message from eWON"
```

See also:

"GETSYS, SETSYS" on page 141, "ONxxxxx" on page 151, "ONSTATUS" on page 153.

9.2.69 SENDTRAP

Syntax[command]

SENDTRAP I1,S1

- I1 is the first trap parameter (INTEGER)
- S1 is the second trap parameter (STRING)

Purpose:

This command posts a scheduled action request for an SNMP TRAP generation.

The first parameter is sent on OID .1.3.6.1.4.1.8284.2.1.4.2

The second parameter is sent in OID .1.3.6.1.4.1.8284.2.1.4.1

--
-- Script information
--
ewonScript OBJECT IDENTIFIER ::= { prodEwon 4 }
scpUserNotif OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This is the text of the last trap sent by the Script"
::= { ewonScript 1 }
scpUserNotifI OBJECT-TYPE
SYNTAX Integer32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This is a free parameters for script generated traps"
::= { ewonScript 2 }

Table 101: Part of MIB regarding BASIC TRAP

When the function returns, the GETSYS PRG,"ACTIONID" returns the ID of the scheduled action and allows tracking this action. It is also possible to program an ONSTATUS action that will be called when the action is finished (with or without success).

Example:

```
REM send a trap with NotifI = 10 and Notif = Trap message
SENDTRAP 10,"Trap message"
```

See also:

"GETSYS, SETSYS" on page 141, "ONxxxxx" on page 151, "ONSTATUS" on page 153, "SNMP Setup" on page 26.

9.2.70 SETIO

Syntax [command]

SETIO S1, F1

- S1 is the name of the Tag or Tag TagId.
- F1 is the value to give to the Tag.

Purpose:

Modifies the value of a Tag. The Tag must be writable (not for the read-only Tags).

Note:

In many cases this function is efficiently replaced by the TagName@ syntax. For example SETIO "MyTag", 10.2 is equivalent to MyTag@=10.2

Example:

```
SETIO "MYTAG", 10.123
```

9.2.71 SETTIME

Syntax [Command]

SETTIME S1

- S1 is the new date / time to set.
- S1 can contain only the time. In that case the date is not modified.
- S1 can contain only a date. In that case the time is set to 00:00:00

Purpose:

Updates the eWON's real time clock.

Note:

An event is generated in the events log.

Example

```
REM The following are valid time updates
SETTIME "1/1/2000": REM Time is set to 01/01/2000 00:00:00
SETTIME "01/12/2000 12:00": REM Time is set to 01/12/2000 12:00:00
PRINT TIME$: REM suppose it returns "15/01/2000 07:38:04"
SETTIME "12:00": REM Time is set to 15/01/2000 12:00:00
```

See also:

"TIME\$" on page 166

9.2.72 SGN

Syntax [function]

SGN F1

Purpose:

Returns the sign of F1.

- If F1 is > 0, the function returns 1.
- If F1 = 0, the function returns 0.
- If F1 is < 0, the function returns -1.

Example:

```
SGN (-10) REM returns -1
SGN (-10.6) REM returns -1
SGN 10 REM returns 1
```

9.2.73 Sqrt

Syntax [function]

Sqrt F1

Purpose:

Returns the square root of F1.

Example:

```
Sqrt 16 :REM returns 4
```

9.2.74 STR\$

Syntax [function]

STR\$ F1/E1

Purpose:

The function returns the character string related to an E1 or F1 number.

Example:

```
a%=48  
a$= STR$ a%  
REM A$ is worth " 48 " after this affectation
```

See also:

"VAL" on page 168

9.2.75 TIME\$

Syntax[function]

TIME\$

Purpose:

Returns the string with the current date and time. The output format is:

25/10/2004 15:45:55

The number of characters in the returned string is constant.

Note:

The GETSYS command provides a mean to return the time as a number of seconds since 1/1/1970.

Example:

```
PRINT TIME$
```

See also:

"SETTIME" on page 165

9.2.76 TGET

Syntax[function]

TGET E1

- E1 is the number of the timer (1 to 4).

Purpose:

Returns N (>0) if the timer expires and then resets the value (N is the number of times the timer has expired) TGET returns. Returns '0' if the timer did not expired since the last call to TGET.

Example:

```
REM timer 1 minute
TSET 1,60
Label1:
IF NOT TGET 1 GOTO LABEL1
```

See also:

"ONTIMER" on page 154, "TSET" on page 167.

9.2.77 TSET

Syntax[Command]

TSET E1, E2

- E1 is the number of the timer (1 to 4).
- E2 is the value in seconds of the timer.

Purpose:

Initializes the timer E1 at an E2 time base (in second). The timer is read by TGET.

Example:

```
REM timer 1 minute
TSET 1,60
Label1:
IF NOT TGET 1 GOTO LABEL1
```

To stop a timer, you must put the value 0:

```
TSET 1,0
```

See also:

"ONTIMER" on page 154, "TGET" on page 167.

9.2.78 VAL

Syntax [function]

VAL S1

Purpose:

The function evaluates the character string and returns the corresponding expression.

Note:

VAL is a function that usually takes an expression and returns a Real after expression evaluation. This VAL function can also evaluate an expression that returns a string.

Example :

```
a$= "12"
a% = VAL (" 10"+ a$)
REM a% equal 1012
a$="abc"
b$="efg"
c$=val ("a$+b$")
REM c$ equal "abcefg"
```

See also:

"STR\$" on page 166.

9.2.79 XOR

Syntax [Operator]

E1 XOR E2

Purpose:

This command returns the bitwise XOR comparison of E1 and E2.

a XOR b returns 1 if [a is true](#) or if [b is true](#), but NOT IF both of them are true.

Example :

```
1 XOR 2 returns 1
2 XOR 2 returns 0
```

See also:

"Operators priority" on page 127, "AND" on page 131, "OR" on page 158.

9.3 Debug a BASIC program

To debug your basic program, you can use the "Script control" screen of the "Script setup" page that is described in chapter "The Script control link" on page 72.

9.4 BASIC Errors Codes

These codes are returned in ONERROR:

Error Name	Error Code
syntax error	0
'(or)' expected	1
no expression present	2
'=' expected	3
not a variable	4
invalid parameter	5
duplicate label	6
undefined label	7
THEN expected	8
TO expected	9
too many nested FOR loops	10
NEXT without FOR	11
too many nested GOSUBs	12
RETURN without GOSUB	13
Out of memory	14
invalid var name	15
variable not found	16
unknown operator	17
mixed string&num operation	18
Dim index error	19
',' expected	20
Number expected	21
Invalid assignment	22
Quote too long	23
Var or keyword too long	24
No more data	25
reenter timer	26
label not found	27
Operation failed	28
ENDIF expected	29
ENDIF without IF	30
ELSE without IF	31
Math error	32
IO Error	33
End of file	34
val in val	35

Table 102:

9.5 Configuration Fields

This section describes the fields found in the *config.txt* file.

All the fields are readable and writable using GETSYS and SETSYS(unless otherwise specified). The file is separated in the several sections (System, UserList and TagList).

One of the three sections must first be *loaded* with the "SETSYS SYS, xxx" command, where xxx is one of SYS, USER or TAG.

Example :

Setting the eWON Identification parameter and printing the Information (parameter = Identification, Information)

```
SETSYS SYS, "LOAD"  
SETSYS SYS, "Identification", "10.0.0.53"  
PRINT GETSYS SYS, "Information"  
SETSYS SYS, "SAVE"
```

9.5.1 SYS Config

The following table describes the fields accessible from the system configuration. The last column gives the ewon configuration web page where the parameter appears. The web pages are found under **Configuration**.

Name	Description	Web Page
Identification	Identification of the eWON (appears on the logon web page logon below the logo)	System Setup/General/General
Information	Complementary information about the eWON	System Setup/General/General
SmtpServerPort	SMTP server port	System Setup/General/General
SmtpServerAddr	SMTP server address	System Setup/General/General
SmtpUserName	SMTP user name	System Setup/General/General
AIRe trigInt	Interval after which an alarm will be re triggered if the condition is still true (only for alarms that have not been acknowledged)	System Setup/General/General
NtpEnable	1 if NTP service is enabled, 0 otherwise	System Setup/General/General
NtpServerAddr	NTP Server address as a chain of char	System Setup/General/General
NtpServerPort	NTP server port	System Setup/General/General
NtpInterval	Interval between NTP connections	System Setup/General/General
PrgAutorun	1 if script starts at eWON boot time. See script control page	Script Setup
FormatRequest	1 if a format has been requested, 0 otherwise	System Setup/General/General
MbsBaudRate	Modbus baud rate. 0 if disabled, positive value otherwise	IO Server Config → Modbus
Mbs2StopBit	1 if Modbus IO server uses two stop bits, 0 if it uses 1 stop bit	IO Server Config → Modbus
MbsParity	0 for none, 1 for even, 2 for odd	IO Server Config → Modbus
MbsReplyTO	Modbus reply time out	IO Server Config → Modbus
MbsPR(x)	x = 1..3, Modbus topic 1..3 polling rate (expressed in Msec)	IO Server Config → Modbus
TimeZoneOffset	Time zone (expressed in HOUR)	System Setup/General/General
MbsAddress	Modbus address	IO Server Config → Modbus
MbsSlaveEn	1 if Modbus slave mode enabled	IO Server Config → Modbus
DecSeparator	Decimal separator: 44 = "," 46 = "."	
Page(x)	x = 1..11, User Page as defined in the Page Lists config page	Pages List
IOSrv(x)	x = 0..9	
IOSrvData(x)	x = 0..9	"Corresponding IO Server Config"
SecureUsr	1 to Enable user security pages	System Setup/General/General
HomePage	User defined home page	System Setup/General/General
MbsSMB(x)	x=1..3, Modbus Topic x	

Table 103: system configuration fields

MbsSIP(x)	x=1..3, Modbus Topic x IP address	IO Server Config → Modbus
FtpServerPort	FTP Server port	System Setup/General/General
FtpServerAddr	FTP server address	System Setup/General/General
FtpUserName	FTP login name	System Setup/General/General
FtpPassword	FTP password	System Setup/General/General
SmtptAllowB64		
MbsEn(x)	x=1..3, Modbus Topic x enabled (1 if enabled, 0 otherwise)	IO Server Config → Modbus
FTPC_SDTO		
FTPC_SCTO		
FTPC_ACTO		
FTPC_RDTO		
DNS_SRTO		
SnmpCom(x)	x=1..5, SNMP Community x	System Setup/General/SNMP
SnmpR	x=1..5, SNMP Community x Read enabled	System Setup/General/SNMP
SnmpW	x=1..5, SNMP Community x Write enabled	System Setup/General/SNMP
SnmpAlwAll	Accepts SNMP packet from any host (1 = enabled)	System Setup/General/SNMP
SnmpHIp(x)	x=1..5, SNMP Host x IP Address	System Setup/General/SNMP
SnmpHCom(x)	x=1..5, SNMP Host x Community	System Setup/General/SNMP
SnmpHTrap(x)	x=1..5, SNMP Host x Trap enabled	System Setup/General/SNMP
SnmpHAlw(x)	x=1..5, SNMP Host x access allowed	System Setup/General/SNMP
MbsBits*	Modbus Bits (7 or 8)	N/A
AlMaxTry	Number of times an action is retried in case of error	System Setup/General/General
AlRetryInt	Interval between action trials in case of error	System Setup/General/General

Table 103: system configuration fields

The MbsBits parameter specifies how the Modbus IO Server will read the bytes. The possible values are 7 and 8 bits/byte.

9.5.2 Com Section

This section describes the fields found in the *comcfg.txt* file. All the fields are readable and writable (unless otherwise specified) using GETSYS and SETSYS with the COM parameter.

Example:

Setting the First ISP phone number (parameter = PPPC1Phone1) to number 0123456789:

```
SETSYS COM, "LOAD"
SETSYS COM, "PPPC1Phone1", 0123456789
SETSYS COM, "SAVE"
```

The following table describes the fields accessible from the communication configuration.
The last column gives the ewon configuration web page where the parameter appears. The web pages are found under **System Setup**.

Name	Description	Web Page
EthIp	Ethernet IP address	Communication/Ethernet
EthMask	Ethernet IP Mask	Communication/Ethernet
EthGW	Ethernet Gateway	Communication/Ethernet
ModemInitStr	Modem Initialization String	Communication/Modem
PPPServerIP	PPP Server IP Address	Communication/Dial UP (PPP)
PPPServerMask	PPP Server IP Mask	Communication/Dial UP (PPP)
PPPServerGW	PPP Server Gateway	Communication/Dial UP (PPP)
PPPClntIp	PPP Client IP Address	Communication/Dial UP (PPP)
PPPCCompress	Enable PPP Client Compression (enabled =1)	Communication/Dial UP (PPP)
PPPCIPhone1	ISP1 Phone number	Communication/Dial UP (PPP)
PPPCUserName1	ISP1 User Name	Communication/Dial UP (PPP)
PPPCPassword1	ISP1 Password	Communication/Dial UP (PPP)
PIN	PIN code (for GSM modem usage only)	Communication/Modem
RTEnIpFwrd	Enable IP forwarding (1 = enabled)	Communication/Router (Filter)
DialInOut	Enable Dial in / out / both (1 / 2 / 3)	Communication/Dial UP (PPP)
InEqualOut	Enable Usage of dial in connection to dial out (enabled = 1)	Communication/Dial UP (PPP)
DialTO	Dial out timeout	Communication/Dial UP (PPP)
CIIdle	Client mode idle timeout before hang up	Communication/Dial UP (PPP)
SrvIdle	Server mode idle timeout before hang up	Communication/Dial UP (PPP)
EthDns1	Ethernet DNS 1 IP Address	Communication/Ethernet
EthDns2	Ethernet DNS 2 IP Address	Communication/Ethernet
PPPSrvCompress	Enable PPP Server compression (enabled = 1)	Communication/Dial UP (PPP)
PPPCINeedChap	Enable CHAP authentication requirement (enabled = 1)	Communication/Dial UP (PPP)
PPPCIPhone2	ISP2 Phone number	Communication/Dial UP (PPP)
PPPCUserName2	ISP2 User Name	Communication/Dial UP (PPP)
PPPCPassword2	ISP2 Password	Communication/Dial UP (PPP)
CallAlloc	Allocated Budget	Communication/Dial UP (PPP)
CallAllocRst	Budget Reset Period	Communication/Dial UP (PPP)
CBEnabled	Enable callback (enabled = 1)	Communication/Callback

Table 104: communication configuration fields

CBDelay	Delay after rings before callback (in seconds)	Communication/Callback
CBIdleTime	Callback mode idle timeout before hang up	Communication/Callback
CBPubEMail	Email address where to send the IP address when callback	Communication/Callback
CBDDnsType	Dynamic DNS Type	Communication/Callback
CBDDnsUName	Dynamic DNS User Name	Communication/Callback
CBDDnsPass	Dynamic DNS Password	Communication/Callback
CBDDnsHName	Dynamic DNS Host Name	Communication/Callback
CBDDnsDName	Dynamic DNS Domain Name	Communication/Callback
CBType	Callback type (0 = Callback on ring, 1 = Callback on User's Request)	Communication/Callback
CBNbRing	Minimal number of rings to detect callback	Communication/Callback
CBTo	ISP to use when calling back (1 / 2)	
RTEnTransFw	Enable transparent forwarding (enabled = 1)	Communication/Router (Filter)
RTEnAuthRt	Enable user authentication when forwarding (enabled = 1)	Communication/Router (Filter)
RTEnableNat	Enable Network Address Translation (enabled = 1)	Communication/Router (Filter)
ModDetCnt	Number of time the eWON tries to detects the modem in case of error (default = 1)	N/A
ModExpType*	Expected modem type (default = -1)	N/A
ModFrcType*	Forced modem type (default = -1)	N/A
SSAM**	Server Access Selection Mode	N/A
CBNbRingOH	Number of rings more than the minimal for callback	Communication/Callback

Table 104: communication configuration fields

* The following table describes the possible modem type values:

Type	Value
Not used	-1
No modem	0
14400 baud	1
33600 baud	2
56600	3
ISDN	4
Unknown	5
Wavecom Wismo Q2403 GSM/GPRS	0X83***

Table 105: modem type values

***Modem type can be found in the eWON Information page you open by clicking on the eWON Logo. In the above case: "Internal BIBAND GSM (131)"

** The following table describes the SSAM possible values:

Description	Value
The last server that worked will be used for next call	-1
Return to first	0 (default)
Always use server 1	1
Always use server 2	2

Table 106: SSAM values

9.5.3 Tag Section

This section describes the configuration fields for a single Tag. The fields are readable and writable using GETSYS and SETSYS with the TAG parameters and the Tag name.

Example:

Printing the alarm status and setting the value (to 45) of the Tag "testTag".

```
SETSYS TAG, "LOAD", "testTag"
PRINT GETSYS TAG, "alstat"
SETSYS TAG, "TAGVALUE", 45
SETSYS TAG, "DoSetVal", 1
SETSYS TAG, "SAVE"
```

The following table describes the fields accessible from the Tag configuration. The web pages are found under *Tag Setup/Tag Name*.

Name	Description
Id	Tag id. Not editable through the web page (only for program usage)
Name	Tag name
Description	Tag description
ServerName	IO Server the Tag gets the value from
TopicName	Topic the Tag takes its basic configuration from
Address	Tag address
Coef	Tag value multiplier coefficient
Offset	Tag value offset
LogEnabled	Enabled Tag value logging (enabled = 1)
AIEnabled	Enable Tag alarm (enabled = 1)
Type	0 = Boolean, 1 = analog
AIBool	Boolean Tag alarm level
MemTag	Is memory Tag (1 = memory Tag, 0 = other)
MbsTcpEnabled	Modbus TCP Enable

Table 107: Tag configuration fields

MbsTcpFloat	Consider as float value (2 subsequent registers)	
SnmpEnabled	Enable SNMP (enabled = 1)	
RTLogEnabled	Enable real time logging (enabled = 1)	
AIAutoAck	Enable alarm auto-acknowledging (enabled = 1)	
ForceRO	Force read-only Tag	
SnmpOID	SNMP OID	
AIHint	Alarm hint	
AIHigh	Alarm high level (warning level)	
AILow	Alarm low level (warning level)	
AITimeDB	Alarm interval deadband	
AILevelDB	Alarm level deadband	
PageId	Page the Tag is published on	
RTLogWindow	Real-time logging time span	
RTLogTimer	Real-time logging interval	
LogDB	Historical logging deadband	
LogTimer	Historical logging interval	
AILoLo	Alarm low-low level (danger level)	
AIHiHi	Alarm high-high level (danger level)	
MbsTcpRegister	Enabled access to the Tag as a Modbus register (enabled = 1)	
MbsTcpCoef	Tag value Modbus TCP publishing multiplier coefficient	
MbsTcpOffset	Tag value Modbus TCP publishing offset	
EEN*	Enable Email	alarm notification config
ETO	Email alarm recipient(s) (coma separated)	alarm notification config
ECC	Email alarm carbon-copy recipient(s)	alarm notification config
ESU	Email alarm subject	alarm notification config
EAT	Email alarm attachment (as Export Block Descriptor)	alarm notification config
ESH	Enable Email sent as SMS (enabled = A)	alarm notification config
SEN*	Enable SMS	alarm notification config
STO	SMS alarm recipient	alarm notification config
SSU	SMS alarm subject	alarm notification config
TEN*	Enable trap (SNMP)	alarm notification config

Table 107: Tag configuration fields

TSU	Trap (SNMP) subject	alarm notification config
FEN*	Enable FTP	alarm notification config
FFN	FTP destination file name	alarm notification config
FCO	FTP file content (as Export Block Descriptor)	alarm notification config
AIStat	Alarm status (0 = no alarm, 1 = in alarm)	View IO page
ChangeTime	Last change time	View IO page
TagValue	Tag current value	View IO page
DoDelete	Delete the Tag (0 = do not delete, 1 = delete)	
DoAck	Acknowledge the Tag (0 = do not acknowledge, 1 = acknowledge)	
DoSetVal	Set to 1 to be able to modify TagValue	

Table 107: Tag configuration fields

9.5.3.1 Send on alarm notification patterns*

In the table below are listed the different pattern values you will find in the in the “:TagList” section from the config.txt file, in the EEN, SEN, TEN and FEN columns, depending on the way you configure the send on alarm action for the Tag (that means, depending on which alarm status will trigger the send on alarm action):

ALM	ACK	RTN	END	Values
				0
x				8
	x			16
		x		32
			x	2

If you activate several of the send on alarm actions checkboxes, the result of the value will be an addition of selected fields’ values:

Example :

If you activate “ALM” and “END” to trigger an SMS sending, the value of the “SEN” field will be 10.

9.5.3.2 Setting a Tag value, deleting a Tag and acknowledging an alarm

A Tag value can be set using the following sequence (shown for a Tag MM1):

```
SETSYS TAG, "LOAD", "MM1"
SETSYS TAG, "TAGVALUE", 1234
SETSYS TAG, "DoSetVal", 1
SETSYS TAG, "SAVE"
```

There are other ways to change a Tag’s value. Examples:

```
MM1@ = 1234
setio "MM1", 1234
```

Let's the MM1 Tag is in alarm state. It is then possible to acknowledge its alarm with the following command

```
SETSYS TAG, "LOAD", "MM1"
SETSYS TAG, "DoAck", 1
SETSYS TAG, "SAVE"
```

It is possible to delete a Tag with:

```
SETSYS TAG, "LOAD", "MM1"
SETSYS TAG, "DoDelete", 1
SETSYS TAG, "SAVE"
```

9.5.4 User Section

This section describes the configuration fields for a *single* user.

The fields are readable and writable using GETSYS and SETSYS with the USER parameters and the user name.

Example :

Changing password of user "pierre"

```
SETSYS USER, "LOAD", "pierre"
SETSYS USER, "password", "new_password"
```

The following table describes the fields accessible from the User configuration.

The web pages are found under *Users Setup/[The name from the User]*.

Name	Description
Id	User Id (only for programs simplicity)
FirstName	User first name
LastName	User last name
Login	User login
Password	User password
Information	User information
Right	Combination of bits for user rights on Tags (acknowledge, view, write, ...)
EMA	User Email address
SMS	User SMS number
AccessPage	The page the user is allowed to access
AccessDir	The directory (and subdirectories) the user is allowed to access
CBEn	Allow the user to use callback (allowed = 1)
CBMode	Callback phone number is: 0 = mandatory, 1 = user defined
CBPhNum	Callback phone number

Table 108: Users configuration fields

10 User defined Web Site

10.1 Introduction

The eWON can host a web site containing user define web pages. The hosting management can be done like for common web site using the eWON FTP server.

When connecting to the eWON FTP server, the root directory contains a /usr directory. The pages from the custom Web site can be located in that directory or in subdirectories of /usr.

A file /usr/index.htm

Will be accessed using **http://10.0.0.53/usr/index.htm**

(If 10.0.0.53 is the eWON IP address)

Static web pages stored in the eWON have a limited interest. That is why the eWON provides 3 ways to make its web content dynamic:

- **Build /usr files using BASIC at regular intervals or upon context change.**
- **Use the SSI (server side include) syntax.**
- **Use bASP (Basic Active Server Page).**

The SSI Stands for Server Side Include; this method is used for creating web sites in which the server updates parts of the HTML page before sending the page to the client.

- **The eWON uses that technique to allow dynamic update of the pages. The following type of data can be dynamically replaced by the eWON in the user page:**
- **Tag Value.**
- **Creation of HTML table for different types of data (historical, event, etc.).**
- **Creation of Graphs containing Real time values or Historical values.**
- **Script expression.**
- ...

The bASP consists in putting BASIC program block in your web pages saved on the server, when the page is delivered to the WEB client. The BASIC block is executed. The BASIC block can access parameters passed to the page and it can output HTML content directly to the page delivered to the client.

The eWON also provides 3 types of FORMS to perform the following actions in user-defined pages:

- **Modify Tag values**
- **Acknowledge alarm**
- **Execute script action**

10.2 SSI Syntax

10.2.1 HTML Page extension

The normal processing for an SSI page is the following:

- **The user requests a page from the server by entering the page address, example: `http://myewon/usr/index.shtm` (there are many other ways to request a page of course).**
- **The server (eWON) checks that the given page is available in the file system.**
- **The server checks the page extension (here .shtm), if this extension is NOT .shtm the page is transferred as-is to the client. This applies to .htm files for example, but also .jpg, .gif, etc.**
- **If the page's extension is .shtm, then server will parse the page and replace all the special HTML Tags (see below) by there current value.**

So the SSI extension for the pages in the eWON is **.shtm** (case sensitive). This extension is recognized by most HTML editors.

10.2.2 Special eWON SSI Tags

There are 4 special HTML Tags defined to provide SSI functionality in the eWON:

```
<%#TagSSI, YYYYY%>
<%#ParamSSI, XXXXXX%>
<%#VarSSI, YYYYY%>
<%#ExeSSI, XXXXXX%>
```

These 4 special HTML Tags follow a generic syntax for SSI Tags, i.e. they start by **<%** and end by **%>**. The consequence of this is that HTML editor will not check the syntax inside these Tag and they will be completely ignored by the editors.

10.2.2.1 TagSSI HTML Tag

The TagSSI is used to insert the current value of a Tag into the page:

```
<%#TagSSI,TagName%>
```

TagName is the name of the Tag that must be inserted.

Notes:

- Do not insert any space in the TagSSI:
- TagSSI is case sensitive:

```
<%#TagSSI,TagName%>
```

Below is the example of a user webpage named "TEMP_page.shtml" which displays the current value from the Tag "TEMP":

```
<html>
  <body>
    <p>
      Last Value of TEMP: <%#TagSSI,TEMP%>
    </p>
  </body>
</html>
```

```
<%#tagssi,TagName%>
```

10.2.2.2 ParamSSI HTML Tag

The ParamSSI is used to insert the content of an export block into the page (See also chapter "Export Block Descriptor" on page 192)

`<##ParamSSI,Parameter%>`

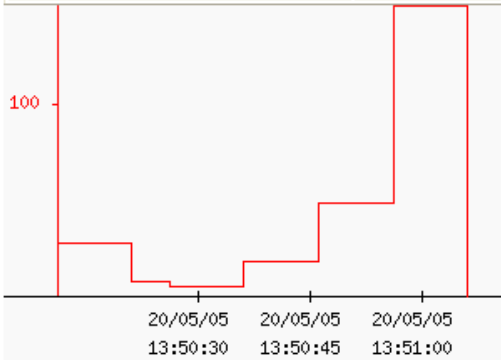
"Parameter" follows the "Export Block Descriptor" syntax described in this manual. The resulting output is directly inserted into the HTML page returned. This format is useful to return HTML Tables into the page or return Script expression (\$dtSE) into the page.

With this method, the exported data is inserted directly into the page. If what you need is a hyperlink to an exported data, the **ParamForm** described in chapter "Produce an export data block" on page 189 should be used:

```

2         <body>
3             <##ParamSSI,[$dtRL$ftH$st_m40$et_m0$tnTank_Level]%>
4             
5         </body>
6 </html>
    
```

TimeInt	TimeStr	Value
1116597012	20/05/2005 13:50:12	30.000000
1116597017	20/05/2005 13:50:17	30.000000
1116597022	20/05/2005 13:50:22	10.000000
1116597027	20/05/2005 13:50:27	8.000000
1116597032	20/05/2005 13:50:32	8.000000
1116597037	20/05/2005 13:50:37	20.000000
1116597042	20/05/2005 13:50:42	20.000000
1116597047	20/05/2005 13:50:47	50.000000
1116597052	20/05/2005 13:50:52	50.000000
1116597057	20/05/2005 13:50:57	150.000000
1116597062	20/05/2005 13:51:02	150.000000
1116597067	20/05/2005 13:51:07	3.000000



10.2.3 VarSSI HTML Tag

See:

"Web context variable directly inserted with SSI" on page 184"

10.2.4 ExeSSI HTML Tag

See:

"Building page content by using bASP" on page 182"

10.3 bASP Syntax

The bASP allows to insert BASIC blocks in the WEB page in order to build the page on the fly while the page is transmitted to the client's web browser.

The big advantages of this technique are:

- It can generate pure HTML, compatible with any WEB client (not even javascript enabled).
- The BASIC code is inserted directly in the WEB page, so it can be created at the same time the rest of the page layout is designed.
- It is very flexible.

10.3.1 Building page content by using bASP

The page created will contain special Tags with BASIC block embedded.

Example:

```
<html>
  <head>
    <title>SSI Demo</title>
  </head>
  <body>
    <p>
      Time:<%#ExeSSI,PRINT #0,TIME$%><BR>
      Array of data:<BR>
      <%#ExeSSI,
        for i%=1 to 3
          print #0,"TData(";i%;" )";a(i%)
        next i%
      %>
    </p>
  </body>
</html>
```

In this example the page actually generated by the eWON would be:

```
<html>
  <head>
    <title>SSI Demo</title>
  </head>
  <body>
    <p>
      Time:05/06/2002 14:09:57<BR><BR>
      Array of data:<BR>
      TData(1) 10<BR>
      TData(2) 20<BR>
      TData(3) 30<BR>
    </p>
  </body>
</html>
```

If a(1)=10, a(2)=20, a(3)=30.

The following remarks can be made about this example:

- The syntax for the bASP inclusion is:

```
<%#ExeSSI,Basic_block%>
```

(Basic block can span multiple lines)

- **PRINT #0,xxx** is an extended syntax of the PRINT command that will route printed data to the client WEB page. As for the common PRINT command, if the sequence of data to print does not ends with a ";", a
 is added to the data to force a new line.
- The execution of the blocks occurs as the page is sent to the user. The blocks nearer from the top of page are executed first.
- The user will never see the Basic Block, he will only see the content of its execution.

SYNTAX:

The bASP syntax is:

```
<%#ExeSSI, Basic_block%>
```

When the block contains multiple BASIC lines, the leading space are not trimmed, they are part of the line submitted to the BASIC interpreter and should be avoided.

10.3.2 WEB context variables

When the client submits a WEB request to the eWON web server:

```
GET /usr/index.shtm
```

Then the web server will create a context to send a response to the client. This context will live until whole data in the index.shtm page have been sent to the client. This context can be used by the BASIC bASP to save temporary data and the access the FORMS and REQUEST parameters.

SYNTAX:

The context variable in basic are ending with a "!", a context variable must always be in **lowercase**. The context variables are always STRING variables (like A\$).

10.3.2.1 Web context variables for request parameters

WEB servers support the syntax where client passes parameters to the server in the URL. In this syntax, each parameter is separated by a &, the first parameter is separated from the file URL by a ? and each parameter has the format **param_name=param_value** (param_name in lowercase).

Example:

```
http://10.0.0.53/usr/index.shtm?param1=1234&param2=ABCD
```

Requests the index.shtm page with:

```
param1=1234
```

```
param2=ABCD
```

The context variables **param1** and **param2** will automatically be created. This means that any bASP program block in the index.shtm page will have access to the 2 basic variables **param1!** and **param2!**

Example:

```
<html>
  <head>
    <title>SSI Demo</title>
  </head>
  <body>
    <p>
      Param1:<%#ExeSSI,PRINT #0,param1!%><BR>
      Param2:<%#ExeSSI,PRINT #0,param2!%><BR>
    </p>
  </body>
</html>
```

In this example we print the 2 parameters to the WEB client. The resulting output would be (for the request above):

```
<html>
  <head>
    <title>SSI Demo</title>
  </head>
  <body>
    <p>
      Param1:1234<BR><BR>
      Param2:ABCD<BR><BR>
    </p>
  </body>
</html>
```

10.3.2.2 Web context variable directly inserted with SSI

There is another syntax for inserting Web context variables in the client web page. A special SSI Tag called VarSSI can be used:

```
<%#VarSSI,variable_name%>
<%#VarSSI,variable_name,default_value%>
```

Where variable_name is the Web context variable (without the ending "!").

In the second syntax, you can have a default value (usefull in case of Form parameters).

The previous example can be written differently like follows:

```
<html>
  <head>
    <title>SSI Demo</title>
  </head>
  <body>
    <p>
      Tag Name:<%#ExeSSI,
      SETSYS TAG,"load","Tag1"
      PRINT #0,GETSYS TAG,"Name"%>
      Tag Desc:<%#ExeSSI,
      SETSYS TAG,"load","Tag1"
      PRINT #0,GETSYS TAG,"Description"%>
    </p>
  </body>
</html>
```

The output is exactly the same, yet there is a strong performance issue between these two implementations:

In the first implementation using ExeSSI, we have 2 bASP blocks that are posted to the eWON basic queue (see also "Program flow" on page 122). It means that the current execution of the program will have to be interrupted twice; the blocks must be inserted in the BASIC program and executed before the output is available to the client.

In the second implementation using VarSSI, the Web context variable are directly read from the context, there is no intervention of the eWON BASIC in that operation. It is much faster.

10.3.2.3 Improving WEB performance using Web context variables and bASP

The example above introduces how the WEB performance can be improved.

Suppose to following example:

```
<html>
  <head>
    <title>SSI Demo</title>
  </head>
  <body>
    <p>
      Tag Name:<#ExeSSI,
      SETSYS TAG,"load","Tag1"
      PRINT #0,GETSYS TAG,"Name"%>
      Tag Desc:<#ExeSSI,
      SETSYS TAG,"load","Tag1"
      PRINT #0,GETSYS TAG,"Description"%>
    </p>
  </body>
</html>
```

In this example, we print the config Name and Description fields for a Tag called "Tag1". There are 2 bASP blocks and we call SETSYS TAG,"load" twice because at the end of the first bASP block there may be a context switch with another BASIC request loading the TAG edition area with another Tag value (depending of the rest of the application, but let's take the worst case).

These 2 SETSYS TAG,"load" take some time and we have 2 BASIC context switch to generate the page.

The following implementation returns the same result with a significantly improved performance:

```
<html>
  <head>
    <title>SSI Demo</title>
  </head>
  <body>
    <p>
      <#ExeSSI,
      SETSYS TAG,"load","Tag1"
      tag1_name!= GETSYS TAG,"Name"
      tag1_desc!= GETSYS TAG,"Description"%>
      Tag Name:<#VarSSI,tag1_name%><BR>
      Tag Desc:<#VarSSI,tag1_desc%><BR>
    </p>
  </body>
</html>
```

In this case Tag1 is loaded only once in the edition area and we generate only one BASIC context switch to execute the only bASP block.

10.3.2.4 Using Web context variables with FORMS

The Web context variables provide the easiest way to use FORM fields when the FORM is posted.

As described below, there is a special form call ExeScriptForm that allows to request execution of a script command.

Example :

```
<form method="POST" action="/rcgi.bin/ExeScriptForm">
<input type="hidden" name="Command" value="goto UseForm">
</form>
```

- When the form is posted, the "goto UseForm" request is posted in the BASIC queue.
- If other fields are added in this form, there content can be accessed using Web context variables.

Example :

```
<form method="POST" action="/rcgi.bin/ExeScriptForm">
<input type="text" name="edit1" size="20">
<input type="hidden" name="Command" value="goto UseForm">
</form>
```

And the BASIC contains the following code at UseForm label:

```
UseForm:
  REM save Edit1 parameter
  A$ = edit1!
  PRINT "Edit1 value entered was: ";A$
  END
```

Warning:

That Web context variable must be lowercase.

10.4 Special FORMS

Some special forms are handled by the eWON to allow:

- Update of eWON Tag values
- Hyperlink to Graph, Text or HTML table
- Execute Script action
- Acknowledge Alarm

10.4.1 Update Tag Value (and acknowledge)

The purpose of this form is to:

- Update the value of a Tag.
- Acknowledge a Tag alarm.

Syntax

Form name: UpdateTagForm

Form fields name:

TagName
TagValue
ResultPageOk

Or:

TagName1
TagValue1
TagName2
TagValue2
TagNameN
TagValueN
ResultPageOk

The first field name syntax can be used to update only 1 Tag per form, while the second syntax can be used if it is required to update more than one Tag at the time. With the second syntax, the number of Tags that may be updated with one FORM is not limited. The eWON will check items with an increment of 1 until not found.

Note:

If TagName1, TagName2 and TagName4 are defined, only TagName1 and TagName2 will be updated because TagName3 is missing.

TagName	Is a form field that will define the name of the Tag to update. Usually it will be a hidden field.
TagValue	Is a field that will hold the new value of the Tag. This may be a list box or a text edit field or any other type, and its initial value may be filled with a TagSSI (see examples), or maybe with a TagSSI and Java Script in case of list box.
ResultPageOk	Is a field that is also usually hidden. This field is optional and defines the page to show when the update has been performed.

Table 109: Tags to update fields

WARNING:

The "ResultPageOk" URL must be specified from the eWON root: i.e.: /usr/xxxx

10.4.1.1 Examples

• Single Tag update

```
<form method="POST" action="/rcgi.bin/rcgi.bin/UpdateTagForm">
<input type="hidden" name="TagName" value="Pressure">
<p>Pressure:
<input type="text" name="TagValue" size="20" value="">
</p>
</form>
```

This example shows how to update one Tag called "Pressure". There is one hidden form field for the Tag name and its value is "Pressure" which is the name of the Tag to update.

There is another form field called TagValue, it is a text edit field where the user can enter the new value of the Tag. When the form is shown, the initial value of the TagValue field is empty.

• Single Tag update with initial value

```
<form method="POST" action="/rcgi.bin/UpdateTagForm">
<input type="hidden" name="TagName" value="Pressure">
<p>Pressure:
<input type="text" name="TagValue" size="20" value="<##TagSSI,Pressure%>">
</p>
</form>
```

The only difference between this example and the previous one is the initial value of the TagValue form field. In this case a TagSSI as been placed in the "value" attribute of the text field. The eWON will replace this placeholder with the current value of the pressure field when the page is displayed.

• Single Tag update with no-default result page

```
<form method="POST" action="/rcgi.bin/UpdateTagForm">
<input type="hidden" name="TagName" value="Pressure">
<p>Pressure:
<input type="text" name="TagValue" size="20" value="">
</p>
<input type="hidden" name="ResultPageOk" value="/usr/x.shtm">
</form>
```

Again, this is the same example as the first one, here an additional field has been inserted to define which page to show when the FORM update has been executed correctly. The additional field is hidden, its name is ResultPageOk and its value is the page to display in case of update success.

• Multiple Tags update

```
<form method="POST" action="/rcgi.bin/UpdateTagForm">
<input type="hidden" name="TagName1" value="Pressure">
<p>Pressure:
<input type="text" name="TagValue1" size="20" value="">
</p>
<input type="hidden" name="TagName2" value="Speed">
<p>Pressure:
<input type="text" name="TagValue2" size="20" value="">
</p>
</form>
```

This example shows the syntax for updating more than one Tag at the time. For each TagNameX there must be a corresponding TagValueX. The first field must have index 1, then the next must follow with an increment of 1.

10.4.1.2 Acknowledge Tags alarms

The same FORM can be used to acknowledge a Tag alarm; the syntax is exactly the same as for Tag update, the only difference is for the content of the **TagValue**.

For Tag alarm acknowledgement, the TagValue field will contain the keyword "ack" optionally followed by the ",**UserName**" who will be logged in alarm history.

Example :

```
<input type="hidden" name="TagValue" value="ack,adm">
```

This example shows a hidden field used to request the acknowledgement of the Tag by the Admin user ("adm"):

```
<input type="hidden" name="TagValue" value="ack">
```

Would yield to the same result because the default user is the Administrator (if none is specified for acknowledgement).

10.4.2 Produce an export data block

Using a <%#ParamSSI,xxx>, it is possible to insert an export data in a page. But sometimes what is needed is a hyperlink to an exported block, for example:

- **Hyperlink to the event file**
- **Hyperlink to a Real time graph picture (will appear as a picture in the page).**

Inserting a picture in a page is not possible with the <%#ParamSSI,xxx>, because the binary content of the picture would be included in the page. A picture in a page is an hyperlink to the file of that picture. Instead of pointing to a file, we can point to a FORM that provides the picture data. The syntax for the export form hyperlink source is the following:

SourceURL

```
"/rcgi.bin/ParamForm?AST_Param=XXXXXXXXXX"
```

(ParamForm is case sensitive).

Where XXXXXXXXXX is the Export Block Descriptor (Please refer to chapter "Export Block Descriptor" on page 192). The URL source can be used anywhere an URL is required, the MIME type of the data returned will match the actual type: PNG, HTML or PLAIN TEXT.

Examples :

- **Real time graph hyperlink**

```
<br>
```

- **Hyperlink to a text file containing real time data**

```
<a href="/rcgi.bin/ParamForm?AST_Param=$$dtRL$tnPressure$st_s20$ftT" > Upload Text file</a><br>
```

10.4.3 Execute an eWON script

The purpose of this form is to execute an eWON script command line by posting a FORM from the web server. Using this feature, you can have a button in the Web page that starts execution of a script section.

The form allows executing a sequence of one or more script commands. Each command is executed as if it was typed and executed from the **Script Control** window. If more than one command is specified, they are executed in sequence.

The length of the command line is limited to 250 chars.

10.4.3.1 Syntax

Form name: ExeScriptForm (case sensitive)

Form fields name
 Command
 ResultPageOk

Or:

Command1
 Command2
 ...
 CommandN
 ResultPageOk

If there is only one script command to execute, the first syntax can be used. If more than one command must be issued, the second syntax applies. The second syntax requires that the CommandX fields starts with Command1 and goes on by increment of 1.

Note:

If Command1, Command2 and Command4 are defined, only Command1 and Command2 will be executed because Command3 is missing.

ResultPageOk is a field that is also usually hidden. This field is **optional** and defines the page to show when the update has been performed.

WARNING:

The page URL must be specified from the eWON root: i.e.: /usr/xxxx

As for execution of command directly from the **Script Window**, and when the program is running, the command is executed between 2 sections executions. This means that the command will NEVER be executed between 2 lines of instructions inside a script.

For example, the commands will be executed between 2 executions of a cyclic section, or after the complete execution of an OnTimer section. In other words, the commands are always executed after a script END command of a section being executed.

When the FORM is posted, the result page returned by the eWON web server is not sent after execution of the command. The command is actually posted for execution but there is a queue of commands that can be more or less filled, and a section execution may be in progress also. This means that if your result page contains fields that should be updated by your command, it is normal that these fields are not yet updated at the time the response page is sent by the eWON web server.

Any script command can be issued with this form, including Goto commands (Do not use Gosub because the rest of the line after Gosub is not executed).

10.4.3.2 Examples

- Single command execution:

```
<form method="POST" action="/rcgi.bin/ExeScriptForm">  
<input type="hidden" name="Command" value="a$='close':MyTag@=1">  
</form>
```

Note:

Please note the use of the (a\$='close') single quote instead of the common a\$="close" syntax that would conflict with the HTML quotes.

- Single command execution with no-default result page

```
<form method="POST" action="/rcgi.bin/ExeScriptForm">  
<input type="hidden" name="Command" value="a$='close':MyTag@=1">  
<input type="hidden" name="ResultPageOk" value="usr/x.shtm">  
</form>
```

Compared to the previous example, the page displayed after execution is the user page x.shtm.

- Multiple commands execution:

```
<form method="POST" action="/rcgi.bin/ExeScriptForm">  
<input type="hidden" name="Command1" value="a$='close' ">  
<input type="hidden" name="Command2" value="MyTag@=1">  
</form>
```

In this example the same commands are executed as in the first example. One of the main differences is that in the first example, the 2 commands (a\$='close' and MyTag@=1) are executed together (no instruction can be inserted between these 2 instructions). In the second case, another command or another section may be executed between the 2 commands.

11 Export Block Descriptor

Exports are used to export eWON data.

Export block can be used in the following situations:

- **Attach eWON data to an eMail**
- **Include eWON data into an eMail content**
- **Make a FTP PUT of eWON data from the eWON to a FTP server**
- **Make a FTP GET from a FTP client out of the eWON FTP server**
- **Include data in an eWON HTML custom page.**
- **Access data in Basic with OPEN “exp:.....”**

In all these cases, an Export Block Descriptor will be used to describe the data to export.

11.1 Export block descriptor

An Export Block Descriptor is a string of characters describing the eWON data to export with a precise syntax.

Typically, the Export Block Descriptor will answer the following questions:

- **What eWON data to export (Event log, Historical logging, etc.)?**
- **How to format the data to export (Binary, Text, Html table, Graphic)?**
- **From what time?**
- **To what time?**
- **What Tag is concerned?**
- **...?**

This list is NOT complete, and this information is not required for all type of data to export, but it gives an idea of what we describe in an Export block descriptor.

Example of Export blocks descriptor:

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

The export syntax is composed of a sequence of fields followed by its value.

A field is a 3 characters identifier starting with \$ and followed by 2 lower cap letters (case sensitive).

- **The first letter of the parameter value follows immediately the second letter of the field.**
- **The parameter is considered up to the first space found or until a \$ or a [is detected.**
- **The parameter can also be placed between quotes (“”). In that case the parameter value is the value between the quotes.**

The following fields are defined:

Fields	Description
\$dt	Data type
\$ft	Export format
\$st	Start time
\$et	End time
\$tn	Tag name
\$ut	Update last time
\$ct	Compression type
\$fl	Group filter

Table 110: Export Block Descriptor fields description

11.2 Export fields syntax definition

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

11.2.1 \$dt [Data Type]

The \$dt field defines what data to export from the eWON.

The \$dt parameter is composed of 2 upper case letters (case sensitive) that can take one of the following values:

\$dt Parameter	Description	Binary	Graph	Text	Html
AH	Alarm history			T*	H
AR	Alarm Real time			T*	H
CF	Config	B*			
ES	estat file			T	H
EV	Event file			T*	H
FW	Firmware	B*			
HL	Historical Logging	B*	G	T	H
IV	Instant values	B*		T	
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

Table 111: \$dt parameters description

11.2.2 \$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

Table 112: \$ft parameters description

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

11.2.3 \$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, \$et parameter:

- Relative time
- Absolute time
- From last \$ut (see also “\$ut [Update Time]” on page 196).

11.2.3.1 \$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, hour, minutes or seconds.

If no letter is specified minutes are considered.

Examples:

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

Table 113: \$st with relative time examples

11.2.3.2 \$st, \$et with absolute time

Syntax:

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

Where:

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 an 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. The parameters is optional, but if specified, HHMMSS, mmm and I must be present also.

Table 114: \$st parameters

When ALL Tags are specified, Tag values are output in chronological order.

For the same time there can be 2 Tag values.

In order to reposition correctly in the file, it is necessary to provide the last Tag output during a previous export.

Examples:

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

Table 115: \$st with absolute time examples

11.2.3.3 **\$st , \$set with Last time**

By adding the \$st command in an Export Block Descriptor, you can ask the eWON to memorize 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`

L is the time parameter meaning last time.

11.2.4 **\$ut [Update Time]**

This field has no parameter, it means that at the end of this exports, 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 :

```
$stL$set_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 Tag" is specified in an export.

11.2.5 \$tn [Tag Name]

This field is used to specify a Tag name. It is required for graph commands. 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 Tags.

Example:

```
$tnMyTag
```

(MyTag is the name of the Tag)

11.2.6 \$ct [compression format]

This field is required when sending a file from the eWON to an FTP server, or as an attachment to a mail. The compression format is gzip (<http://www.gzip.org>). So that the unique argument to add after the field "\$ct" is "G".

Example:

```
Putftp "test2.txt.gz", "[&#215; $ctG $uf/test.txt]"
```

Or:

```
SENDMAIL "destinator@provider.net", "", "Subject", "Mail body &[#215; $uf/usr/test.txt $ctG $fn/test2.txt.gz]"
```

Note:

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

**You can download a free trial version of this tool at the following address: <http://www.rarlab.com/download.htm>.*

11.2.7 \$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 within quotes because if a \$ is found in the expression it will be considered as end of parameters.

Example:

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

(Exports the content of A\$)

11.3 Data Types description and syntax

Data type defines what is exported from the eWON. The data type is defined by the \$dt field followed by 2 uppercase letter. The \$dt field is mandatory for each "Export Block Descriptor" and usually the \$ft (Format) field will also be 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 must be provided (some are mandatory and others are optional).

Note:

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.

11.3.1 \$dtHL [Historical Logging]

11.3.1.1 Export content

The Historical logging outputs the data from the File system for ONE or ALL fields.

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.

11.3.1.2 Detailed Example

```
$dtHL $ftT $st_h4 $et_m0 $tnA1
```

\$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
\$tnA1	Tag history to output

Table 116: \$dtHL detailed example

11.3.1.3 Fields used

Fields	Value if not specified
Mandatory	
\$dt	
Optional	
\$ft	Binary
\$st	01/01/1970
\$et	31/12/2030
\$tn	All
\$ut	No time update
\$fn	Export block descriptor
\$ct	Compression type

Table 117: \$dt - fields used

11.3.1.4 Special parameters and fields

\$st \$et

If Last time is specified: (\$stL or \$etL): there is a last time logged for each Tag plus 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 Tag (concerned by historical logging) and another last time memory is used.

If output format is graph: \$et_0 should be used instead of default value because otherwise the graph spans up to 31/12/2030. For binary or text output, the default value can be kept.

\$ft

Acceptable values			
Binary	Text	HTML	Graph

Table 118: [\$dtHL] \$ft acceptable values

Graph format is only allowed if 1 Tag has been specified.

Text format will output a comma-separated file.

The separator is ',' to avoid confusion with decimal point.

If all Tags are output they will be output in a chronological order in the file.

\$ut

If only one Tag is specified, the time of the last point for that Tag will be memorized.

All Tags can be output individually and last time is saved for each point.

Another memory if available if \$ut is requested for ALL Tags.

\$tn

If this Tag is not specified, ALL Tag will be selected for export.

Otherwise, the Tag with the given name will be selected.

11.3.2 \$dtRL [Real time Logging]

11.3.2.1 Export content

The Real time logging outputs the 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.

11.3.2.2 Detailed Example

```
$dtRL $ftG $st_m10 $et_m0 $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
\$tnA1	Tag log to output
A1	Name of the Tag

Table 119: \$dtRL - detailed example

11.3.2.3 Fields used

Fields	Value if not specified
Mandatory	
\$dt	
\$tn	
Optional	
\$ft	Binary
\$st	01/01/1970
\$et	31/12/2030
\$ut	No time update
\$fn	Export block descriptor
\$ct	Compression type

Table 120: \$dtRL - fields used

11.3.2.4 Special parameters and fields

\$st \$et

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

\$ft

Acceptable values			
Binary	Text	HTML	Graph

Table 121: [\$dtRL] \$ft - acceptable values

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

\$tn

A Tag MUST be specified for this export (does not work on ALL Tags).

11.3.3 \$dtAH [Alarm History]

11.3.3.1 Export content

The Alarm History outputs the data from the File system for ONE or ALL Tags.

The output format can be TEXT or HTML Table.

A time range can also be specified for this export.

11.3.3.2 Detailed Example

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

\$dtAL	data type Alarm history logging
\$ftH	output format requested is HTML table
\$st01012001	1 st of January 2001
\$et	not specified → until the end of file
\$tn	not specified → all Tags

Table 122: \$dtAH - detailed example

11.3.3.3 Fields used

Fields	Value if not specified
Mandatory	
\$dt	
Optional	
\$ft	Text
\$st	01/01/1970
\$et	31/12/2030
\$tn	All
\$ut	No time update
\$fn	Export block descriptor
\$ct	Compression type

Table 123: \$dtAH - fields used

11.3.3.4 Special parameters and fields

\$ft

Acceptable values	
Text	HTML

Table 124: [\$dtAH] \$ft - acceptable values

Text format will output a comma-separated file.
 The separator is ',' to avoid confusion with decimal point.
 If all Tags are output they will be output in a chronological order in the file.
 Line content of output file:

`"EventDate"; "TagName"; "Status"; "UserAck"; "Description"`

\$tn

If this Tag is not specified, ALL Tags will be selected for export. Otherwise, the Tag with the given name will be selected.

11.3.4 \$dtAR [Alarm Real time]

11.3.4.1 Export content

The Alarm Real time outputs the real time data for ONE or ALL of 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).

11.3.4.2 Detailed Example

```
$dtAR $ftT
```

\$dtAR	data type Alarm Real time
\$ftT	output format requested is CSV
\$tn	not specified → all Tags

Table 125: \$dtAR \$ft - detailed example

11.3.4.3 Fields used

Fields	Value if not specified
Mandatory	
\$dt	
Optional	
\$ft	Text
\$tn	All
\$fn	Export block descriptor

Table 126: \$dtAR - fields used

11.3.4.4 Special parameters and fields

\$ft

Acceptable values	
Text	HTML

Table 127: [\$dtAR] \$ft -acceptable values

Text format will output a comma-separated file.

The separator is ',' to avoid confusion with decimal point.

If all 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

If this field is not specified, ALL Tags will be selected for export.

Otherwise, the Tag with the given name will be selected.

11.3.5 \$dtEV [Event file]

11.3.5.1 Export content

The Event file outputs the data from the File system. The output format can be TEXT or HTML Table. A time range can also be specified for this export.

11.3.5.2 Detailed Example

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

\$dtEV	data type events logging
\$ftT	output format requested is CSV
\$st_m30	last 30 minutes
\$et	not specified → until now

Table 128: \$dtEV - detailed example

Will output a CSV file containing the events during the last 30 minutes.

11.3.5.3 Fields used

Fields	Value if not specified
Mandatory	
\$dt	
Optional	
\$ft	Text
\$st	01/01/1970
\$et	31/12/2030 ⇔ NOW
\$fn	Export block descriptor
\$ct	Compression type

Table 129: \$dtEV - fields used

11.3.5.4 Special parameters and fields

\$ft

Acceptable values	
Text	HTML

Table 130: [\$dtEV] \$ft - acceptable values

Text format will output a comma-separated file.
 The separator is ',' to avoid 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

Table 131: EventTime types

11.3.6 \$dtSS [Scheduled Status]

11.3.6.1 Export content

The scheduled actions 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 sequential execution. This exports allows checking the content of this queue and giving the status of all the actions in the queue: "in progress", "executed (success)" and "executed with error".

11.3.6.2 Detailed Example

\$dtSS

11.3.6.3 Fields used

Fields	Value if not specified
Mandatory	
\$dt	
Optional	
\$ft	Text
\$fn	Export block descriptor

Table 132: \$dtSS - fields used

11.3.6.4 Special parameters and fields

\$ft

Acceptable values	
Text	HTML

Table 133: [\$dtSS] \$ft - acceptable values

Text format will output a comma-separated file.

The separator is ',' to avoid confusion with decimal point.

Line content of output file:

"ActionId", "ActionType", "StatusCode", "StatusText", "Start", "End"

11.3.7 \$dtSE [Script Expression]

11.3.7.1 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, and INTEGER or a FLOAT.

The evaluation of the expression will always occur between 2 scripts execution, for example between 2 ONTIMER executions, or between 2 cycles of the cyclic sections.

11.3.7.2 Detailed Example

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

11.3.7.3 Fields used

Fields	Value if not specified
Mandatory	
\$dt	
\$se	
Optional	
\$ft	Binary
\$fn	Export block descriptor

Table 134: \$dtSE - fields used

11.3.7.4 Special parameters and fields

\$ft

Acceptable values		
Text	HTML	Binary

Table 135: [\$dtSE] \$ft - acceptable values

Binary and Text format means 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)).

Then 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.

11.3.8 \$dtUF [User File]

11.3.8.1 Export content

The User File export returns the content of a file in the User File area (/usr/ directory – or subdirectory). When the file is exported, the <##ParamSSI> and <##TagSSI> Tags are replaced by the actual values.

11.3.8.2 Detailed Example

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

\$dtUF	User file
\$fn/ufdir/uf1.txt	Will export the uf1.txt file located in the /usr/ufdir directory

11.3.8.3 Fields used

Fields	Value if not specified
Mandatory	
\$dt	Must always precede "UF": dtUF (means Data Type= User File)
\$fn	
\$uf	
Optional	
\$ft	Binary
\$ct	Compression type

Table 136: \$dtUF - fields used

11.3.8.3.1 \$fn [File Name]

This field is used for specifying a file name to the export data.

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, because the PUTFTP command manages the name of the destination file:

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

There is one special case: when a user file (\${dtUF}) is exported; in that case, the file name is the name of the user file (i.e. not only the destination file name but also the source file name).

Examples:

Using \$fn in a send mail string:

```
SENDMAIL "MailReceiver@YourMail.com", "", "Mail Subject", "Mail text &[${dtUF}$fn/myfile.txt]"
```

Use the above syntax if you want the attached file to keep its name.

```
SENDMAIL "MailReceiver@YourMail.com", "", "Mail Subject", "&[${dtUF} $uf/myfile.txt $fn/myfile2.txt]"
```

Use the above syntax if you want the attached file to be received with a different name.

11.3.8.3.2 \$uf [User File]

This field is required when sending a file from the eWON to an FTP server, in case the file names on the eWON and the server must be different.

The \$uf parameter specifies the "destination name" of the file which will be sent to the FTP Server.

Another use of \$uf is when sending a mail with an attachment, in case you want the attachment to have a different name from its name in the eWON (see example in chapter "\$fn [File Name]" on page 210)

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

• Syntax 1:

```
/myfile.txt
```

(myfile.txt is in the /usr directory)

• Syntax 2:

```
/mydir/myfile.txt
```

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

Note:

The first "/" is optional.

The complete path can also be specified:

• **Syntax 1:**

```
/usr/myfile.txt
```

(myfile.txt is in the /usr directory)

• **Syntax 2:**

```
/usr/mydir/myfile.txt
```

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

Note:

The first "/" is mandatory.

Example:

```
Putftp "/test.txt", "[${UF} $uf/myfile.txt] "
```

11.3.8.3.3 Special parameters and fields

\$ft

Acceptable value
Binary

Table 137: [\${UF}] \$ft - acceptable values

11.3.9 \$dtIV [Instant Values]

11.3.10 Instant value - general information

Instant value means values of Tags at a given time. The file of instant value contains for each Tag the following information:

TagId	Id of the Tag
TagName	Name of the Tag (in text mode)
Value	Current value of the Tag
AIStatus	Current alarm status of the Tag
AIType	Type of the current alarm

Table 138: \$dtIV - instant value file's informations

- The file containing the instant values for every Tag is available in binary or text format; you can download the instant values file directly from the root of the eWON root file list, or you can address it by using an Export Block Descriptor.
- 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.
- In the Tag definition, there is a new config in the "Tag visibility section".
- There is a group of 4 check boxes, each of them being associated to a group called A, B, C, D (4 groups).
- Every Tag can belong to no group, one group, or more than one group.
- These groups are only used when reading the instant values using an Export Block Descriptor; in that case there is an additional field in the Export Block Descriptor. That allows for requesting the instant values for Tags belonging to one or more groups.

IMPORTANT:

Regardless of the group definition for each of the Tags, the inst_val.txt and inst_val.bin files (see below) always return the instant value for ALL Tags.

These groups have nothing in common with the A, B, C topics of the IO servers.
They are defined in the context of the Instant Values!

• **Root file access**

In the eWON root folder (FTP access or file transfer) you will find the following 2 files:

inst_val.txt	instant values in text mode
inst_val.bin	instant values in binary mode

11.3.10.1 Alarm status code values

The table below lists the different values that the field **AIStatus** 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 status Warning: we assume there is no alarm if AIStatus 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

Table 139: inst_val.txt file - alarm status code values

11.3.10.2 Alarm type values

The table below lists the different values that the field **AlType** can have, depending on the type of threshold that has been stepped over by the Tag value, depending on the configuration set in the Tag's configuration page:

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

Table 140: inst_val.txt file - alarm type values

11.3.10.3 Writing Instant Values to eWON

The instant values file can also be written to eWON. The file must be written by FTP to the ftp root folder and must be written to file. All the Tags value present in the file will be used to change the corresponding Tag in the eWON. If a Tag is no found, then it will be ignored.

- **Writing in binary format:**
 - The file format must comply exactly with the definition (see below) and all Tags are identified by their Tag ID.
- **Writing in text format:**
 - When writing the instant value in text format, there are different possibilities to address the Tag:
 - If a "TagName" column is present, then the Tags will be accessed by their name (even if a "TagId" column is present)

Example :

```
"TagId";"TagName";"Value";"AlStatus";"AlType"
1;"M1";10.000000;0;0
2;"M2";20.000000;0;0
```

If a "TagName" column is NOT present, the Tags will be accessed by their id:

```
"TagId";"Value";"AlStatus";"AlType"
1;10.000000;0;0
2;20.000000;0;0
```

WARNING:

Remember that the Tag Id is not an index, but a unique number that has been allocated to the Tag when created, and remaining never reused unless the configuration is erased and a new configuration is created.

11.3.10.4 Binary file format

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

```
struct InstantValueHeader
{
    int    Rev;
    int    RecSize;           //Record size
    int    NbTag;            //Number of Tags exported
    int    RecFlag;          //Reserve (must be set to 0)
    int    Reserved2;
}
```

Then there is a record number for each Tag (the record number can be obtained from the header (NbTag):

```
struct InstantValueRecord
{
    int    TagId;
    float Value;
    int    AlStatus;
    int    AlType;
    int    Reserved;
}
```

WARNING:

All data in these records are stored in BigEndian

Until more information is available in this chapter, please refer to "Technical Note 03" for more information about eWON's data representation of floats and big endian format (<http://www.ewon.biz /Support/Technical Notes>).

11.3.10.5 Export content

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

11.3.10.5.1 Detailed examples

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

Table 141: \$dtIV - detailed examples

11.3.10.5.2 Fields used

Fields	Value if not specified
Optional	
\$fl	

Table 142: \$dtIV - fields used

11.3.10.6 \$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 groups must be in uppercase.

Example:

```
$dtIV $flAB
```

Will export all Tags belonging to group A or B.

11.3.11 \$dtSV

11.3.11.1 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.

11.3.11.2 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)

Table 143: \$dtSV - detailed example

Will include the eWON online IP address in the body from a sent eMail.

11.3.11.3 Fields used

Fields	Value if not specified
Mandatory	
\$se	System expression At this time, only "OnlineIpAddress" is available

Table 144: \$dtSV - fields used

11.3.12 \$dtPP

11.3.12.1 Export content

\$dtPP exports the dump.ppp file (binary format): the file in which the online eWON activity is logged. The output format can only be of BINARY type.

11.3.12.2 Detailed Example

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

\$dtPP	Data type PPP dump
\$fn	Will give to the file the required name

Table 145: \$dtPP - detailed example

Will attach the eWON PPP dump file to a sent on alarm Email.

11.3.12.3 Fields used

Fields	Value if not specified
Optional	
\$fn	File name

Table 146: \$dtPP - fields used

11.3.13 \$dtES

11.3.13.1 Export content

\$dtES exports the estat.htm file : the file that lists the current status from the main eWON features.
The output format can be TEXT or HTML.

11.3.13.2 Detailed Example

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

\$dtES	Data type estat file
\$ftH	Will export the file in htm format
\$fn	Will give to the file the required name

Table 147: \$dtES - detailed example

Will attach the eWON estat.htm file to a sent on alarm Email.

11.3.13.3 Fields used

Fields	Value if not specified
Optional	
\$ft	File type
\$fn	File name

Table 148: \$dtES - fields used

11.3.14 \$dtSC

11.3.14.1 Export content

\$dtSC exports the communications configuration file (comcfg.txt): the file that lists the current status from the main eWON communication features.

The output format can be TEXT or HTML.

11.3.14.2 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 htm format
\$fn	Will give to the file the required name

Table 149: \$dtSC - detailed example

Will attach the eWON comcfg.htm file to a sent on alarm Email.

11.3.14.3 Fields used

Fields	Value if not specified
Optional	
\$ft	File type
\$fn	File name

Table 150: \$dtES - fields used

11.3.15 Additional exports available

\$dtTL	Tag List
\$dtPG	Program
\$dtCF	Configuration File
\$dtFW	FirmWare

Table 151: additional exports available

These are all files from the eWON configuration. They are equivalent to the file available through the eWON FTP server.

12 Upgrading the eWON firmware

12.1 Purpose

There are two ways to upgrade the eWON firmware: by using eBuddy, the ACT'L utility designed to set up the eWON, or by directly uploading a new firmware on the eWON by means of a FTP client.

12.2 Upgrading the eWON firmware with eBuddy

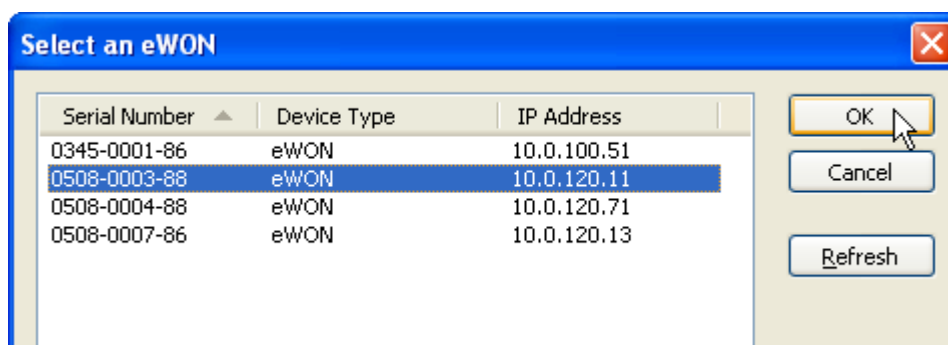
eBuddy allows you to update your eWONs with the latest firmware versions that are available for download from the eWON web site.

eBuddy downloads the firmwares and it stocks them in a folder on the local machine.

When you use the *Update Firmware* feature, eBuddy compares the firmware on the selected eWON with the latest firmwares, and it proposes to you to upgrade your eWON with one of the stored and up-to-date firmwares.

That means you first have to update eBuddy itself prior to upgrade the eWON firmware (please refer to TN 28 eBuddy User Guide you can download from Support/Documentation/Technical notes on <http://www.ewon.biz>).

The *Update Firmware* link in the eBuddy Wizard home page is used if you want to update the firmware from one of the eWONs on your network. Click on the link to launch the wizard:

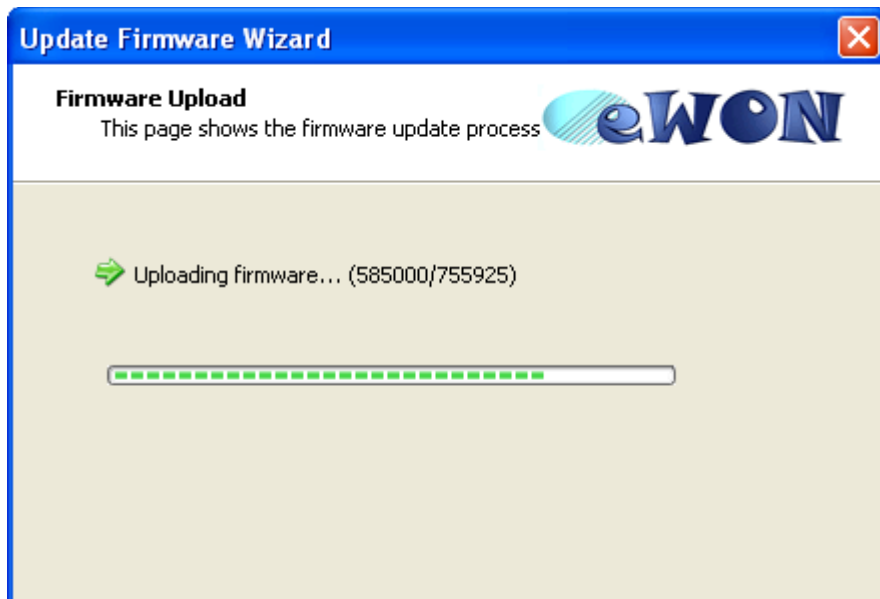


Select an eWON, either by entering directly its serial number or by choosing it in the *Select an eWON* dialog box; enter the login and password for the eWON, then click on **Next** in the *Update Firmware Wizard*.

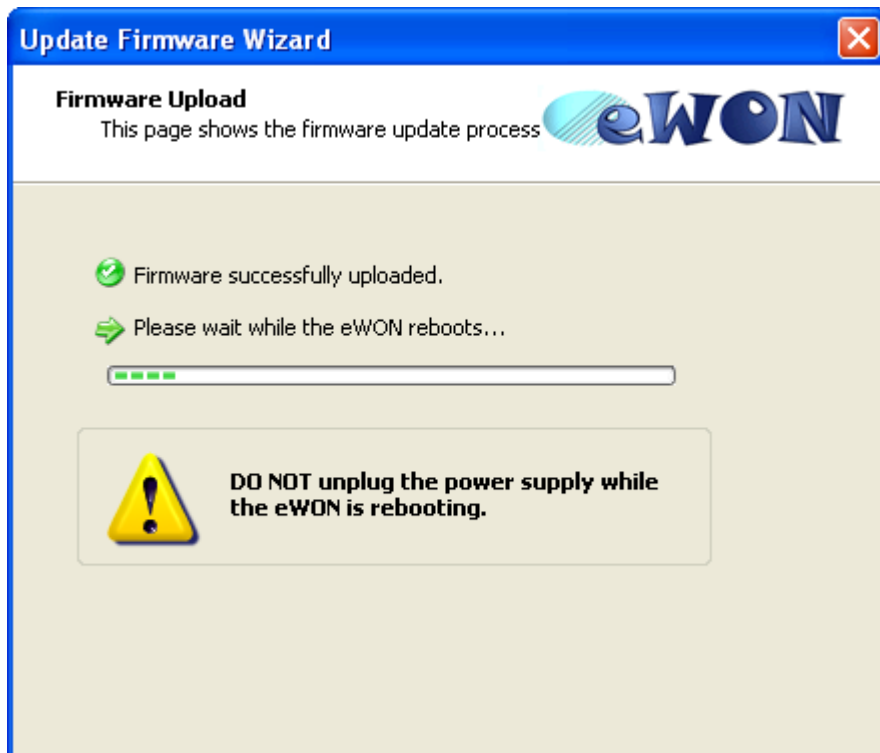


Select the firmware **Language** and **Version** you want to apply, then click **Next**.

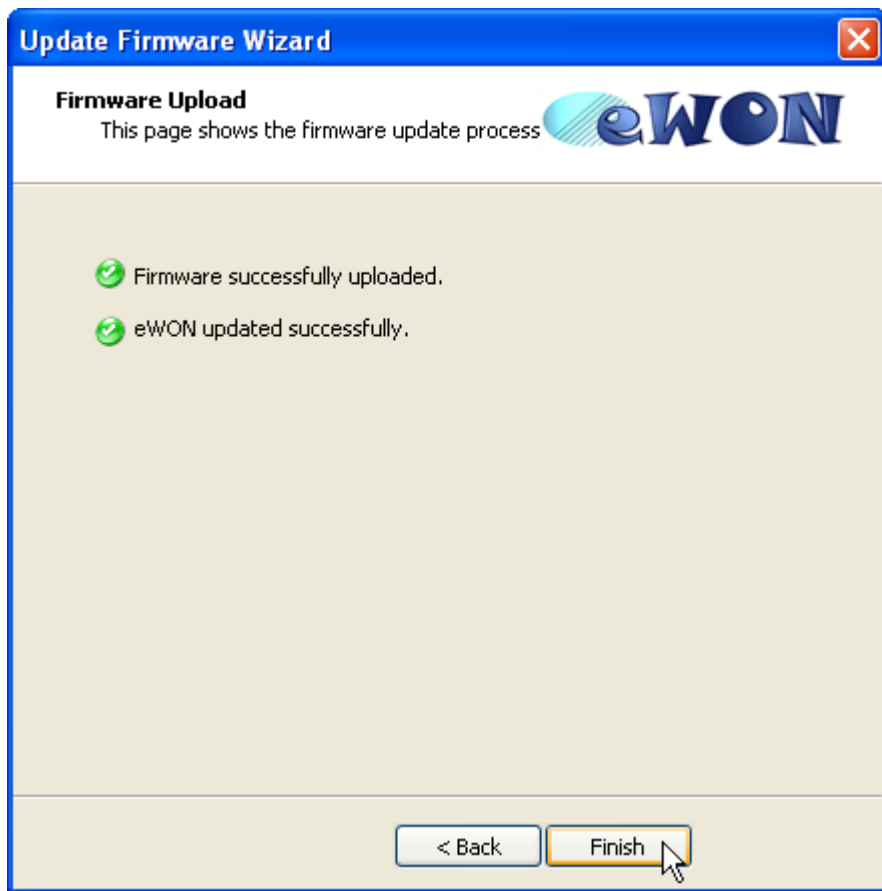
The new firmware is uploaded to the selected eWON:



Click on Next when upload is complete, the following dialog box then appears. Please DO NOT power OFF the eWON when it restarts, or this could lead to make it unusable.



Click on the **Finish** Button when the upgrade is complete and exit from the Wizard.



12.3 Upgrading the eWON firmware by a direct upload

You can easily upgrade the eWON firmware using the SmartFTP tool available on our Web site <http://www.ewon.biz>.

To upgrade the eWON firmware, please follow the following instructions:

- Start the SmartFTP program.
- Type in the eWON IP address in the "URL" field, the user name in the "Login" field and the user password in the "Password" field.

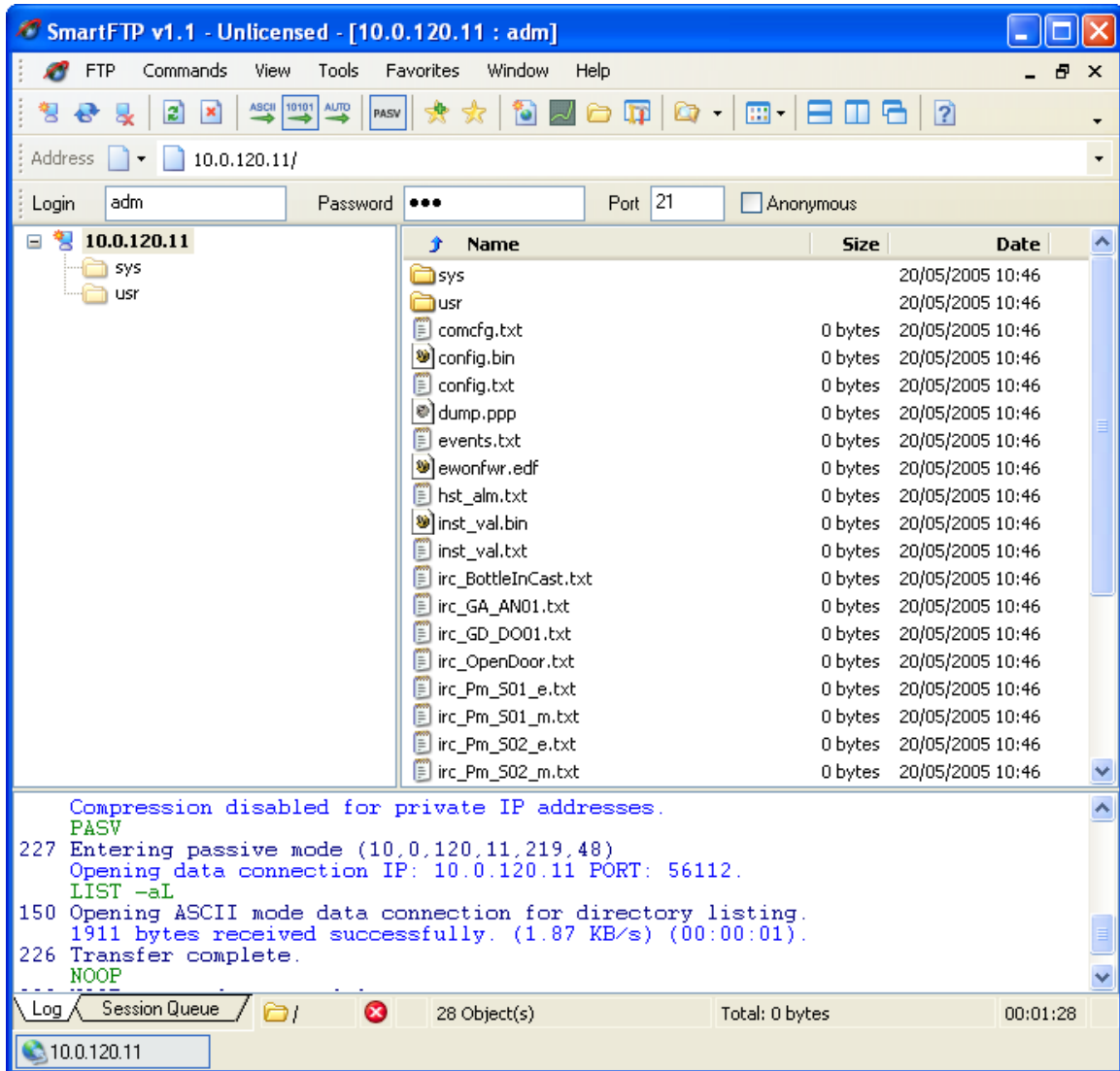


Figure 76: Connection to eWON through SmartFTP

- Click on the "Connect" icon in the SmartFTP window to connect to the eWON with the parameters you entered. A new window appears showing all the files that are present in the WON.

- Simply drag the new firmware to be uploaded (from your local hard disk or from another FTP folder) and drop it to the root of the eWON file system (here the window is named "10.0.120.11"). A confirm window will appear asking you the action to be performed.

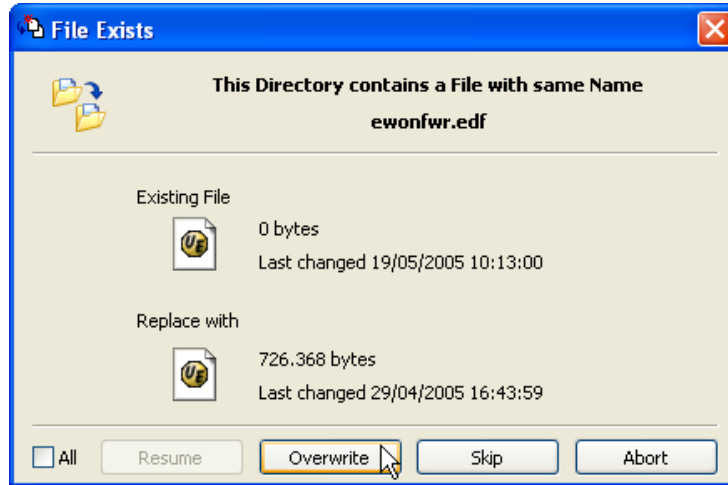


Figure 77: eWON firmware overwrite confirmation window

- Click on the *Overwrite* button and wait the fill in (blue) of the progress bar in the left bottom side of the window.

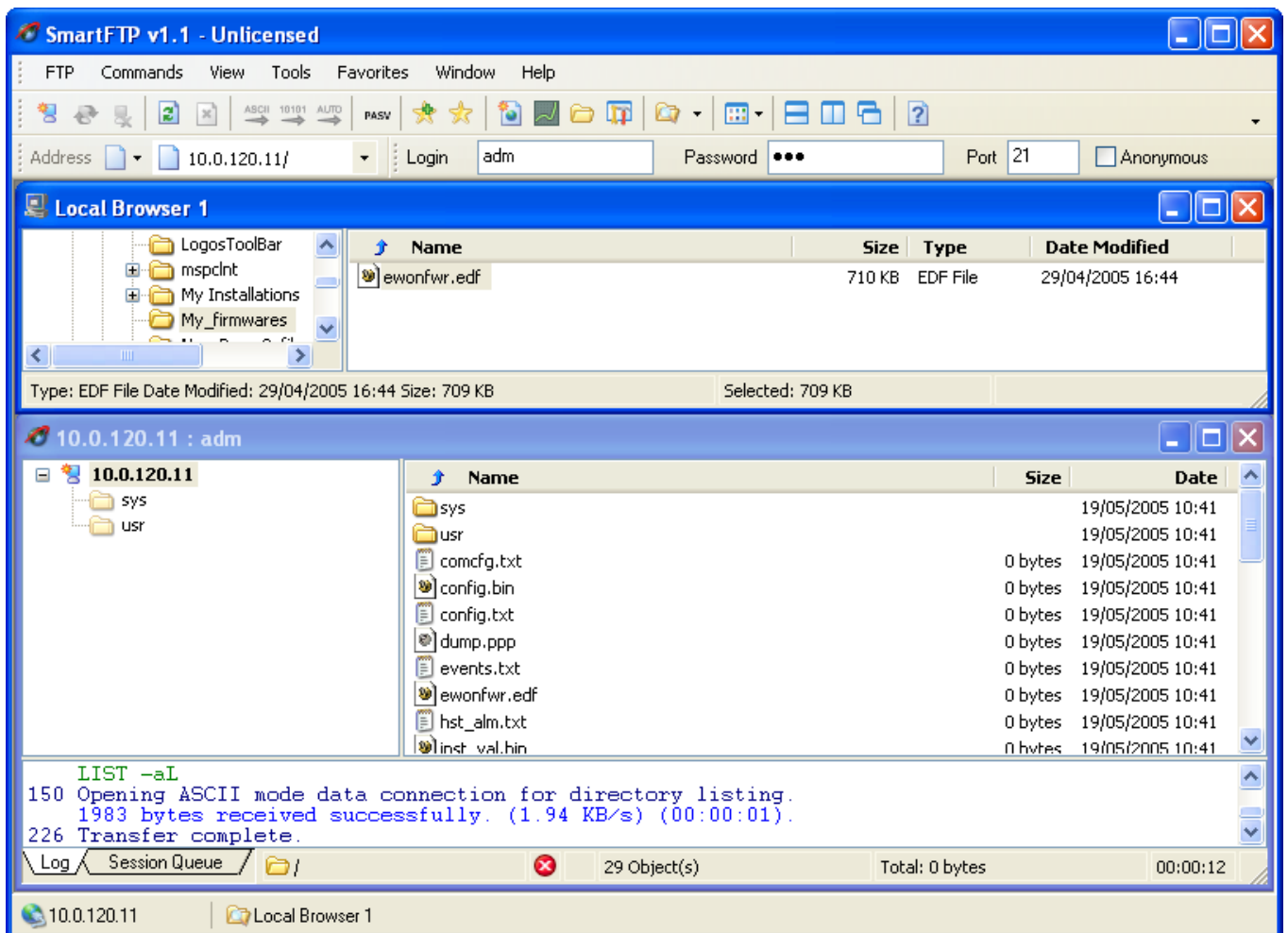


Figure 78: eWON upload firmware action

- Click on the “Disconnect” icon to exit the connection to the eWON and quit the SmartFTP program.

WARNING:

When you click on the “Disconnect” icon, the eWON begins a flash memory programming operation (about one minute long).

Do NOT remove the power from the eWON during this operation.

If you have a look at the eWON STATUS led, it will be blinking about once a second.

When it finishes blinking, the eWON will reboot.

The reboot operation can be observed on the Ethernet link led, going off and then back on again.

Please do NOT remove power until you can get access to the eWON using your web browser or FTP browser.

Failure to do so can lead to the destruction of the eWON and a factory return would be unavoidable.

13 Appendix

13.1 Access to the eWON Technical Support

An eWON technical support is provided on our Web site (<http://www.ewon.biz/>). Just fill in the support request sheet (/Support/Support request), or send your problem description to support@ewon.biz. Our support team will provide you with technical information to help you solving any issue you could encounter to configure your eWON (even if we think that this User Manual answers the more exhaustively as possible to such questions), or to integrate it or collecting information on your industrial networks.

13.2 eWON configuration and files storage

This chapter explains how data storage is organized in the eWON.

13.2.1 Flash file system

The eWON uses a robust flash file system that uses a 1Mbyte flash area.
The following files are saved in this flash file system:

File	Type	Max Size	Description
Configuration (Except COM configuration)		128K	Configuration file: <ul style="list-style-type: none"> • System setup • Pages setup • IO Servers setup • Tags setup • Users setup Note: The content of the COM setup page is not saved here
Program		128K	Script program
/usr directory		1 to 3 MByte	Content of the whole /usr directory and its subdirectories The size of the /usr directory can be defined by the user since version 4
Event file	C	128K	Event file
Alarm History	C	128K	Alarm history file Max number of alarm history: 8192
Historical logging	C	16384, 73728 or 139264 points	Historical logging file

Table 152: Flash file system - files type and size

Important:

“C” file type means the file is a circular file. This kind of file has 2 sizes, the standard size and the maximum size. When maximum size is reached, the oldest 64K of data are erased and new data starts to be written. This means that the actual size of data that has to be considered for a circular file is the standard size, because maximum size is not permanent.

Formatting the Flash file system means erasing all the data in these files.

13.2.2 Non-volatile configuration

There are 2 additional blocks of flash memory used for:

- **Non-volatile COM configuration:** this block contains the "System COM" setup.
- **Retentive values.** This file is also erased when the flash file system is formatted.

13.3 Tips for Internet setup

13.3.1 Finding the IP address of a given host

The eWON does not provide DNS (Domain Name Server) resolution. It means that the usual way to reference nodes on Internet, which is "by name", will not work with the eWON. Instead you have to specify the IP address of the destination node.

In order to find the address corresponding to a given name, you can use the **ping -a NodeName** command.

This command will return miscellaneous information including the IP address of **NodeName** that you need.

Example:

```
C:\>ping -a microsoft.com
Pinging microsoft.com [207.46.197.101] with 32 bytes of data:
Destination host unreachable.
Destination host unreachable.
Destination host unreachable.
Destination host unreachable.
Ping statistics for 207.46.197.101:
Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

The microsoft.com IP address reported is in italic for this example.

13.4 Finding your PC IP address

Under Windows 95/98, the **WINIPCFG** command (executed from a command prompt or from the **START/RUN** menu) will return your Ethernet and PPP adapter IP address.

If not currently connected through PPP, the PPP IP address is N/A.

WINIPCFG DOS command does not exist anymore on Millennium, Windows NT, 2000 and XP systems.

The **IPCONFIG** command can be used instead.

This command must be executed from a command prompt and displays as text the IP address of all the TCP/IP adapters that are detected.

Example:

```
C:\>ipconfig
Windows 2000 IP Configuration
Ethernet adapter Local Area Connection:
Connection-specific DNS Suffix . :
    IP Address. . . . . : 10.0.0.11
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . :
```

13.5 Resetting the eWON

13.5.1 Overview

In some situation it may be desired or required to initialize the eWON.

The eWON has a non volatile memory which is used to store configurations and acquired data. Non volatile information is basically divided into 2 groups:

- **Communication configuration**
- **The rest of the non volatile data (file system, Tag Config, user defined web site, etc.)**

The reset button is located between the **SERIAL** and **LINE** connectors.

A very small hole is located in the box; you will need a thin tool like a paper clip to push the button.

The button must be pressed while the eWON is powered up.

The button must be maintained until correct initialization level has been reached (see below):



Figure 79: eWON - reset hole and "USER" led

13.5.2 Reset sequence

There are 2 initialization levels:

- **The first level (the more usual) will force a format of the eWON.**
- **The second level will reset the eWON in a state corresponding to the "out of the box" configuration. By doing this you will also perform a thorough self test of the eWON.**

13.5.2.1 First level reset

When the eWON boots with the SW switch pressed, after 4 seconds, the "USER" led will start blinking at a 1 flash per second rate.

As soon as the led starts blinking at that rate, release the switch. The Format request will be set.

The blinking will continue during 15 seconds, so there is no need to hurry, but you must release the switch before the 15 seconds elapse.

13.5.2.2 Second level reset

If the SW switch is maintained for more than 15 seconds after blinking starts, the "USER" led will stop blinking and will become "solid red", at that moment the "Out of the box reset" is registered, the SW switch can be released and all the data of the eWON will be erased, including its communication, IP address,... parameters.

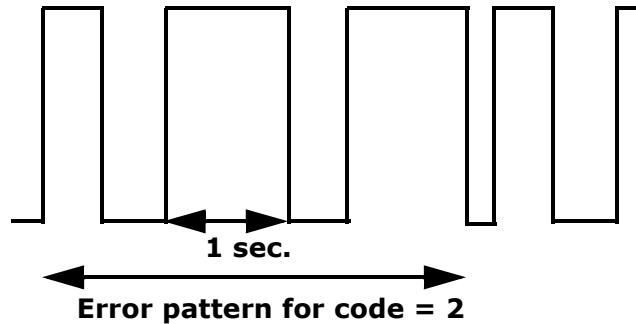
The eWON configuration tool **eBuddy** will be required to define your eWON working IP address.

Note: if you have waited too long and the "USER" led becomes "solid red" while you only needed to format the eWON, the only solution is to keep the SW switch pressed and remove the eWON power, this will avoid entering the "Full Erase" procedure.

13.5.3 Second level initialization diagnostics

When the second level initialization is requested, the eWON will also perform a self test. When the initialization ends, the test result is displayed via the "USER" led. If the test is successful, the following pattern will be displayed on the led. The led will flash for 200msec every 1.5 seconds. The pattern is repeated indefinitely until the eWON is manually rebooted (Power OFF/Power ON).

If an error is detected during the test, the led will show the error code with the following pattern:



This example shows the case where error code is 2. The pattern is repeated indefinitely until the eWON is manually rebooted (power OFF/Power ON).

The pattern starts with a short blink of 200msec indicating the pattern start, then N blinks, each during 1 second are displayed. Then the pattern is repeated, etc.

The number of 1 second blinks is defined by the type of error detected:

Number of blinks	Error meaning
1	RAM Test error
2	Flash erase error
3	Flash write error
4	Real Time Clock error
5	Flash identification failed
6	IO CPU not responding

Table 153: eWON USER led blinks meaning

13.5.4 Entering level 2 initialization without request

The eWON may enter level 2 test without request in the following case:

- If the IO CPU is not responding at boot time

In that special case you may detect the typical led blinking of the level 2 initialization, without having requested that mode.

13.5.5 What to do in case of error?

If an error is detected during the self test, run the test again to confirm the result. If test error persists, then contact the factory for return and repair procedure.

13.5.6 Important remark

When performing a level 2 self test, it is important to let the test run until the end, if you stop the test before the end, the flash memory may contain random data that may cause unexpected operation of the eWON when it will start. For this reason, a self test must always be continued until the "USER" led displays the test result.

In case if trouble you can perform the level 2 initialization again to return to a normal situation.

13.6 The eWON firmware upgrade process

13.6.1 Overview of the upgrade process

Very important notes:

Applying a new firmware on the eWON requires following a very precise process that is described in chapter “eWON firmware upgrade step-by-step” on page 231.

Please do not attempt to upgrade your eWON prior reading this chapter if you are not so familiar with this process, otherwise you could severely damage your eWON.

Versions 4.xx apply to eWON 500, 2001, 4001 and 4002. That means, NOT TO eWONs 1000, 2000 and 4000.

It is possible to come back to version 3 after you have installed the firmware version 4 on your eWON, providing you apply a level 2 reset to your eWON prior to “downgrading” it to version 3.

The choice of the firmware you will install on your eWON depends on the current version your eWON embeds:

- If your eWON is still running a version 3 firmware, you should use the firmware that permits you to upgrade from v3 to v4.
- In case you are already running a version 4, and that you want:
 - either upgrade from your current v4 version to the latest version
 - or reapply your current firmware, because you think that your eWON requires such an operation, Then you should use a v4 to v4 firmware.

All the V4 firmwares are available for download from <http://www.ewon.biz/> (Support/Download software), depending:

- On your eWON's P-Code (86, 87, 88 or 89),
- On the firmware version your eWON is currently running (V3.8 or V4.xx).

Note:

You can always check your eWON's current version, by clicking on the eWON logo that is available from any standard eWON's website page (at the top left corner, as illustrated on the screenshot below)

- On the language with which you want to monitor your eWON (English or French).

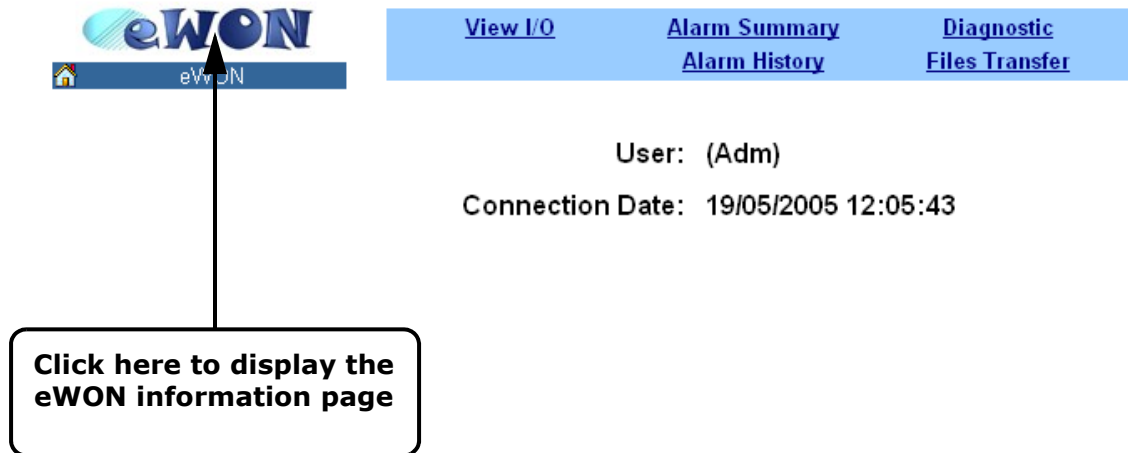


Figure 80: Link to get access to the eWON info page

eWON Information	
Identification	eWON
Additional Info	
IP Address	10.0.120.11
Version Codename	EW_4_1S2
Revision Number	4.1
Serial Number	0508-0003-88
System Info	
IO Revision	1
Modem type	Internal 33600 (2)
Free Config Mem.	124617
Free Prog. Mem.	129111

Figure 81: eWON information page

13.6.2 eWON firmware upgrade step-by-step

13.6.2.1 Upgrading from version 3 to version 4

Firmwares versions 3 and 4 are very different, that is the reason why the flash action erases all of the existing files from the eWON (configuration and user files), and the eWON IP address too.

- The step-by-step process described below is the standard process that we advise you to follow to be sure to stay on the safe side when performing the eWON's firmware upgrade from version 3 to version 4:
- Download the required eWON's firmware version from the eWON website <http://www.ewon.biz/> (/Support/Download software) to your local disk. **Double check that you download the correct firmware version (version 3 to version 4)**. The file you will download will be a zipped (compressed format) file, that contains the *ewonfwr.edf* file.
- Open a FTP session on your eWON, by using a FTP client software, such as SMART FTP, that is available for download on the eWON web site <http://www.ewon.biz/> (/Support/Resource Links). Please refer to chapter "FTP transfer" on page 120.
- Backup your current eWON user and configuration files, by downloading them from the eWON FTP site to your local disk, because all of those files will be erased during the upgrade firmware process. You will be able to upload them on your eWON when the upgrade process is complete. The files you may want to keep are:
 - the eWON configuration files (*config.txt* and *comcfg.txt*)
 - the file containing your Basic script (*program.bas*)
 - the files that contain the information about the Tags you have created (*var_lst.txt*, *var_lst.csv*, *ircall.bin*)
 - the historical logs for each of the Tags for which you have selected this option
 - the user website you have created, which is stored in the *usr* folder (and its subfolders)
- Verify the files you have just downloaded, in order to check that no information could have been lost during the FTP transfer
- Verify that the version 4 firmware you have downloaded in step 1 IS NOT in the same folder as your eWON files you have just downloaded (in order to be sure not to upload again the firmware in the eWON when you will upload your user and configuration files, at the end of the process)
- You are now ready to migrate from version 3 to version 4... Upload the firmware *ewonfwr.edf* file using the FTP tool, by dragging it from the local folder on your hard disk on which you have stored it, towards the root of the eWON's FTP site. Nothing will happen until the FTP connection is alive
- In order to initiate the writing operation of the new firmware on the eWON flash memory, select the "Disconnect" command from the FTP client software. The upgrade process now begins...
- **WARNING:** Process takes some time and eWON cannot be disconnected during that time, please refer to chapter "Monitoring of eWON's behavior during the flashing operation" on page 232
- When eWON comes back to its usual state, you now have to define its IP address, as its settings are now the factory settings (IP address=10.0.0.53). You will have to use the eBuddy tool that is provided for free on <http://www.ewon.biz> (/Support/Download software).
 - Please refer to the Technical Note 28 "eBuddy User Guide" you will find on <http://www.ewon.biz> (/Support/Documentation/Technical notes), which describes the eWON IP address definition process with eBuddy.

Your eWON is now a firmware version 4 eWON, you now just have to browse it with your Internet explorer to discover its new features.

13.6.2.2 Upgrading from version 4 to version 4

There is no loss of files neither than configuration settings when upgrading from a version 4.xx to a newer 4.xx version, then performing a backup of your configuration and user files is not absolutely mandatory, as it is the case when upgrading from version 3 to 4. However, you can do this backup if you want to be sure to stay on the safe side, because i.e. a casual power loss during writing operation in the flash memory may cause severe damages to your eWON, as it is the case for every computer.

The steps to follow to upgrade from a version 4 to a version 4 firmware are then:

- Download the required eWON's firmware version from the eWON website <http://www.ewon.biz/> (/Support/Download software) to your local disk. **Double check that you download the correct firmware version (i.e. version 4.01 to version 4.02)**
- Open a FTP session on your eWON, by using a FTP client software, such as SMARTFTP, that is available for download on the eWON web site <http://www.ewon.biz/> (/Support/Resource Links). Please refer to chapter "FTP transfer" on page 120.
- Upload the firmware *ewonfwr.edf* file using the FTP tool, by dragging it from the local folder on your hard disk on which you have stored it, towards the root of the eWON's FTP site. Nothing will happen until the FTP connection is alive (while you are connected)
- In order to initiate the writing operation of the new firmware on the eWON flash memory, select the "Disconnect" command from the FTP client software. The upgrade process now begins. The behavior of the USER led is identical as for upgrade from version 3 to 4 (please refer to the chapter that follows: "Monitoring of eWON's behavior during the flashing operation" on page 232.
- When flashing process is complete, you are now able to browse your eWON website and FTP site, running the newly firmware version, without having to redefine the IP address or having to set again the communication settings from the eWON.

13.6.2.3 Monitoring of eWON's behavior during the flashing operation

You are able to monitor the eWON behavior during flashing operation, by observing the **USER** led that is located at the top right of the led panel from the eWON.

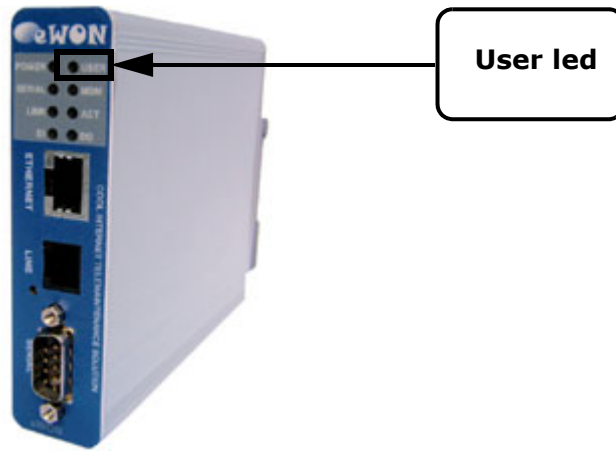
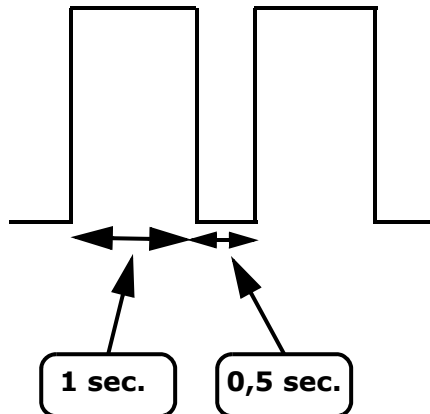


Figure 82: eWON USER led

As soon as you have selected the "Disconnect" command from the FTP client software, you can observe the following **USER** led behavior:

- The **USER** led first lights off, then turns to solid red during about 2 seconds.
- The led will blink red during 1 second every 1,5 seconds.
- The pattern is repeated during approximately 45 seconds.



- Then the red led lights off, and stays without any activity during at least 25 seconds.
- Then the led blinks green again at the rate of about 2 times per second, that is the usual behavior of an eWON that is in order to work.

13.7 Table of comparisons between eWON types

Feature	eWON 500	eWON 2001	eWON 4001	eWON 4002
“Dial up (PPP)” on page 42	NO (no modem)	YES	YES	YES
“Callback” on page 45	NO (no modem)	YES	YES	YES
“PPP Dump” on page 34	NO (no modem)	YES	YES	YES
“COM Setup section” on page 35	NO (no modem)	YES	YES	YES
Historical Data Logging “Tag main edit window” on page 59	NO	NO	YES	YES
Real Time Logging “Tag main edit window” on page 59	NO	NO	YES	YES
“Modem” on page 37	NO (no modem)	YES	YES	YES
“Router” on page 48	NO (no modem)	YES	YES	YES
“SMS on alarm configuration” on page 65	NO (no modem)	YES	YES	YES

Table 154: Features available depending on the eWON type

13.8 Duplicate IP detection (User Led blinking Red)

Since firmware 4.3, eWON performs a check on the IP network in order to avoid duplicate IP. eWON makes this test at power up to see if its IP address is not conflicting with another device.

If eWON is configured with the same IP address than another device, eWON will not start and the USER LED blinks continuously with the following pattern:

short red light + pause + long red light + pause

Resolution:

- 1) Isolate your eWON from the conflicting device
The best way is to use a direct crossed IP cable to directly link your PC with the eWON (without a hub)
- 2) Reboot the eWON
- 3) Change the eWON IP address (with eBuddy, you can download from <http://www.ewon.biz> *Support/Download Software*)

When eWON is blocked, the duplicate IP test is made again every 10 minutes. If the conflicting device is not present anymore, then eWON will start normally.

13.9 TCP/IP Bootstrap Protocol (BOOTP)

Since firmware 4.3, eWON is BootP manageable.
 You can force eWON to ask its IP address to a BootP Server.
 To correctly set the IP mask and the Gateway, your BootP server must comply with the RFC-1048.

ETHERNET SETUP	
Address Setup	
eWON Ethernet IP address	10.0.120.71
eWON Ethernet IP mask	255.255.0.0
eWON Ethernet IP Gateway	0.0.0.0
eWON Use BOOTP	<input type="checkbox"/> (Ethernet address, mask and gateway will be provided by BOOTP)
	WARNING: if "Use BOOTP" is enabled and no BOOTP server is present, use button to unlock eWON during boot sequence.
DNS Setup	Leave blank (or 0.0.0.0) if no DNS or if DNS automatically allocated by PPP server
Primary DNS IP address	0.0.0.0
Secondary DNS IP address	0.0.0.0

Update Ethernet Setup

Figure 83: Ethernet Setup Page

At each startup, eWON will receive its IP address from the Server.
 eWON will wait for the server unless it is not present.
 eWON will perform a new attempt at encreasing interval (1 minute interval max.) endlessly.

While waiting for its IP address, eWON is in startup phase and thus it is not functioning! During this time, eWON will blink continuously the USER LED with the following pattern:

short red light + pause + long green light + pause

Resolution:

- 1) Pushing on the Reset button (on the front face from the eWON) will skip the BootP request, then eWON will use the IP address which is configured in the Ethernet configuration page
 The Duplicate IP test is skipped too!
- 2) Make a Second Level Reset (see chapter "Second level reset" on page 227) to force eWON to 10.0.0.53 IP address and to disable the BootP mechanism

eWON is now accessible.