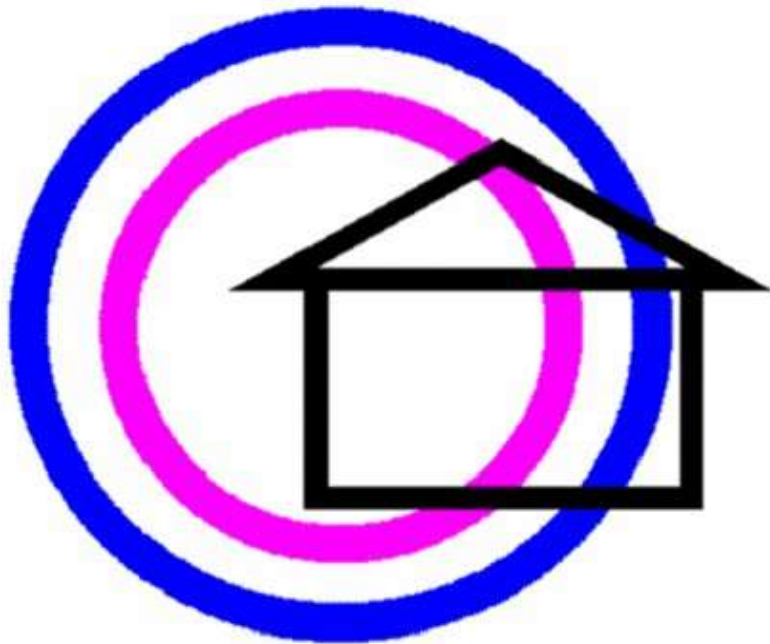


Zdanetix



The versatile Home Automation system **Reference Manual**

Helmut Müller

Version 0.1

First draft, march 2015

<http://www.achemtronic.eu>

Contents

1. Presentation.....	5
1.1 General configuration	5
1.2 The Raspberry Pi in short.....	6
1.3 Overall structure of the software	6
1.4 Terminology	7
1.5 The database.....	8
1.5.1 Overview	8
1.5.2 The domains	8
1.5.3 The channel tables	9
1.5.4 The input/output matrix	10
2 I/O hardware configuration.....	10
2.1 General.....	10
2.1.1 Raspberry Pi	10
2.1.2 Serial line	10
2.2 System and configuration parameters	10
2.3 The MCP23S17 I/O expander	11
2.4 The MCP3xxx series A/D converters	12
2.5 Serial line I/O	13
3 The user interface	13
3.1 Main menu	13
3.2 Login.....	14
3.3 Clock	15
3.4 Data edition.....	15
3.4.1 Numeric programs.....	16
3.4.2 Analogic programs.....	17
3.4.3 Numeric channel tables.....	17
3.4.4 Analogic channel tables.....	19
3.4.5 The catch-all editor	19
3.5 Graphics	21
3.5.1 The channels list	22
3.5.2 Display configuration	22
3.5.3 Configuration parameters	22
3.5.4 Named lists of channels	23
3.6 User controls	24
3.7 Journalling	24
3.7.1 Viewing the contents of a journalling domain	25
3.7.2 Managing the message library	25
3.8 States.....	26
3.9 Simulations.....	27

4	Processing units	28
4.1	Overview.....	28
4.1.1	Numeric channels.....	28
4.1.2	Analogic channels.....	28
4.2	Numeric channels.....	29
4.2.1	Channel table parameter names.....	29
4.2.2	progdchan - Programmed channel.....	30
4.2.3	clonechan - Copy the value of a channel from its associated channel.....	30
4.2.4	dcombo - Logical combination of 3 channels.....	30
4.2.5	dtoggle - Channel toggling	30
4.2.6	copychan - Copy the value of a channel to its associated channel	30
4.2.7	monitdchan - Monitored channel	31
4.2.8	dpulse - Pulsed numeric channel	31
4.2.9	dchange - Actions on numeric channel state change.....	31
4.2.10	dlogchan - Numeric channel daily logging.....	31
4.2.11	copyseq - Copy the value of a channel to its seq channel.....	31
4.3	Analogic channels.....	31
4.3.1	Channel table parameter names.....	32
4.3.2	achange - Actions on analogic channel state change.....	32
4.3.3	alogchan - Analogic channel daily logging.....	33
4.3.4	avalue - Analogic channel value evaluation	33
4.3.5	blink - Blinking a numeric output channel.....	33
4.3.6	progachan - Programmed channel.....	33
5	Application examples	34
5.1	Heating regulation	34
5.2	Garage door automation	35
5.3	Lighting control	35
5.4	Intrusion detection	36
6	Installation	36
6.1	Hardware.....	36
6.2	Software.....	36
7	Application development.....	36

The **Zdanetix** project (Russian **здание**, *building*) is an effort to develop a simple, multifunctional Hardware/Software home automation application, that can be used for small business premises or private houses.

Its main objectives are as following :

- acquire, record and process a limited number of numeric and analogic data (a few dozens)
- elaborate numeric and analogic controls while combining these data with commands and setpoints entered by the user
- heating, lighting, access, etc. control and/or monitoring
- data logging and journalling
- provide a friendly user interface accessible through the Internet, using a web browser, without the need to install additional software on the client side
- be simple and open enough to be easily portable to small low-end configurations.
- be localization friendly, at least for the languages using an 8-bit latin alphabet

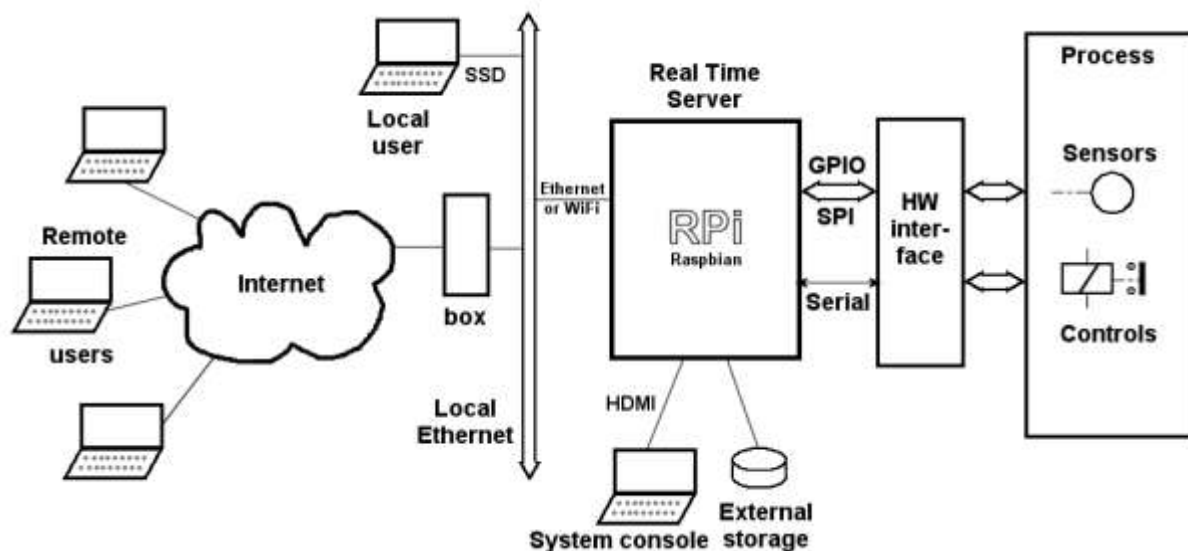
The web site <http://www.achemtronic.eu> is largely dedicated to this software and presents a demonstration of an emulated subset of the user interface.

1. Presentation

1.1 General configuration

For performance and portability reasons, the software is written in C. It can be ported to any system configuration with adequate computing resources (CPU, memory, mass storage, system software) and a decent C compiler and library. It runs well on a Windows system, which is its current development environment (the free version of Visual Studio), but, because of a combination of features usually not available on most other systems, the preferred deployment platform is the **Raspberry Pi**, model B or B+ (abbreviated as RPi).

Therefore, unless stated otherwise, any further reference to non-standard hardware features (GPIO, etc.) will refer to this platform.



The overall configuration implements a real-time process control system featuring :

- on one side : multiple connections to sensors and controls of a specific process, which are managed by a dedicated *hardware interface* that connects to one or several of the computer's I/O ports (USB, COM, SPI, GPIO, etc.)
- on the other side : a web server connected to a local Ethernet and/or the Internet, providing the navigator based user interface
- in the middle : the application software.

A local system console, which may be useful for privileged maintenance transactions, is optional in normal operation and can be emulated on a PC through the network.

Video and sound functionalities are not currently supported by the software.

1.2 The Raspberry Pi in short

The Raspberry Pi (while keeping an eye on some of its recent competing clones) presents a unique combination of features that makes it a platform of choice for home control applications :

- its low cost, which encourages to deploy several of them, each dedicated to a specific small application or an independent subset of the whole process, avoiding the necessity to overload one device with too much functionality or activity
- its adequate performance for relatively slow and low-end process control applications
- its small footprint (physical dimensions and power requirements)
- its embeddability
- its large and evergrowing user community

Besides its Ethernet port, the feature of interest is its GPIO port, with its various subfunctions :

- a serial port, connectable to a microcontroller used as process control connections expander
- an I2C port, interfacing a real-time clock module
- a SPI port, connected to I/O expanders and ADC inputs
- individual GPIO pins, used for specific functions like providing addresses to multiplexors

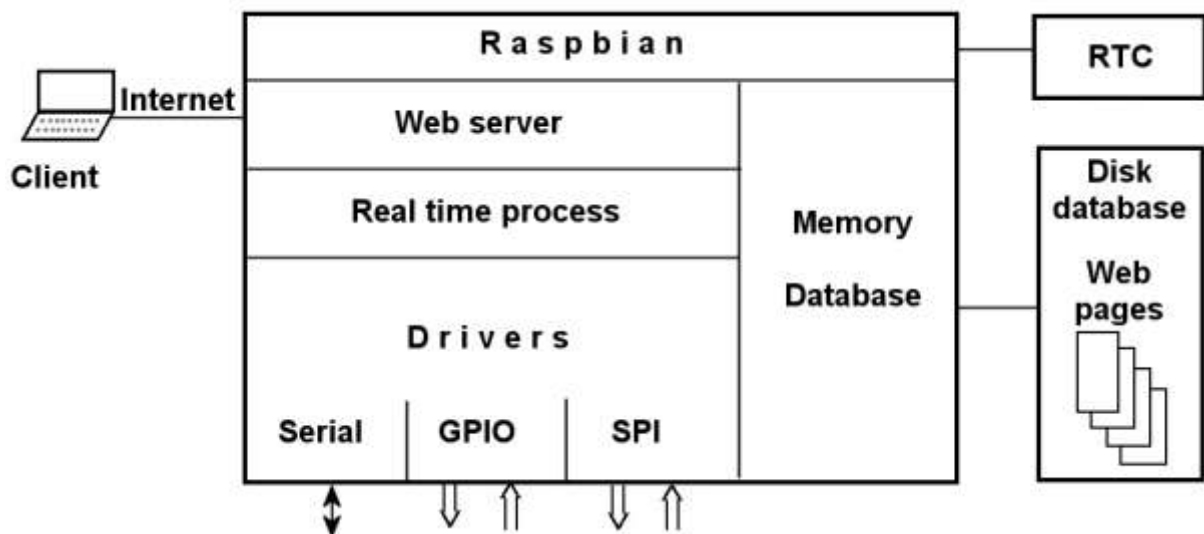
However, it lacks some important features which must be provided by the interfacing hardware :

- analogic I/O
- a real time clock, if time must be preserved across reboots and losses of Internet connection

There is already a wealth of literature and Internet information available on the Raspberry Pi and Raspbian, therefore the present document will only talk about whatever is necessary to work with the application.

Note : When referring to GPIO pins in the following, except otherwise stated, the BCM2835 pin designations are used, not the Raspberry Pi pin header layout.

1.3 Overall structure of the software



The application software is built around a database stored on external mass storage (disk or SD card for the RPi) with a subset in memory, for the information that must be immediately available. This subset is built up from scratch during the startup phase.

Except for the operating system (Linux Raspbian for the RPi), in this version, the application software is bundled in a single executable with the following functions :

- a web server executing several identical threads
- an interface between the web server and the data base, similar to a CGI or PHP procedure in standard web server configurations
- a process management loop executing as a unique background thread, which does all the data processing and calls the drivers.
- drivers that support the various I/O protocols

1.4 Terminology

This is a list of terms used in this document, along with their accepted meanings. Some of them may have more than one meaning, depending on the context. If a term refers to any of those, the referred meaning is specified in parentheses.

Application	combination of the <i>system</i> and the <i>process</i> (1)
Channel	unit of data interchange with the <i>process</i> (1) <i>Numeric or digital channel</i> 1-bit value, 0 or 1 <i>Analogic channel</i> multi-bit value (usually an integer number of bytes) <i>Input or output channel</i> defines the direction of the data interchange. Set up on a channel basis, depending on the needs of the <i>application</i> <i>Virtual channel</i> hidden from the hardware, used by software as both input and output <i>Channel number</i> a number uniquely identifying the channel, like an address <i>Channel name</i> a text string associated with a channel, usually telling its usage
Computer	the part of the <i>system</i> that runs the software, manages the data and supports the <i>user interface</i>
Current value	a value supplied by the <i>process</i> (1) at a given time (<i>now</i>). May be <i>raw</i> or <i>physical</i> (<i>French</i> : mesure, <i>German</i> : Istwert). Frequently compared to a <i>setpoint</i> .
Data base	collection of data used by the <i>system</i> , both on <i>disk</i> and memory
Disk	catch-all name for whatever external mass storage is available on the <i>system</i> : flash memory, USB key, SD card, hard disk, etc.
Drivers	part of the <i>software</i> that drives the <i>hardware interface</i>
Hardware interface	the hardware subsystem that interconnects the <i>system</i> and the <i>process</i> (1)
Node	<ol style="list-style-type: none">1. record on <i>disk</i> containing the values of <i>system parameters</i>2. structure in memory initialized with those values
Physical value	a <i>raw value</i> converted to a value in some physical unit. For <i>numeric values</i> , this may be the <i>raw value</i> or its complement For <i>analogic values</i> , this may imply a more or less complicated transformation of the <i>raw value</i> (currently only linear)
Presentation value	a <i>physical value</i> after a <i>rescaling</i> designed to display or print it in a meaningful format and range
Process	<ol style="list-style-type: none">1. the environment controlled by the <i>system</i>;2. part of the application software that manages the <i>process</i>(1), not including the <i>drivers</i>3. a unit of activity in the computer (may be used for a thread)
Program	<ol style="list-style-type: none">1. a subset of the <i>software</i>(1)2. a collection of values, usually organized on a time-of-day basis, which are used as timed <i>setpoints</i>
Quarter hour	abbreviation for quarter of an hour, a 15 minutes time interval. Within a day, they are numbered from 0 to 95.
Raw or hardware value	any value exchanged between the <i>computer</i> and the <i>hardware interface</i>
Rescaling	the operation of multiplying or dividing a value by a scaling factor and possibly applying an offset. This may be needed for storage or display requirements and is done identically for all values of a given type.
Setpoint	a value set by the <i>user</i> or computed by the <i>system</i> and used as reference (<i>French</i> : consigne, <i>German</i> : Sollwert). Frequently compared to a <i>current value</i> .
Software	<ol style="list-style-type: none">1. the collection of application programs, including the <i>process</i>(2)2. the data management and the <i>user interface</i>

State word	one of the entries in a system wide <i>state words table</i> contained in a Javascript module included by HTML pages using this feature, and used as a channel state indicator and in messages telling the occurrence of an event on a channel. Most used state words : <i>on, off, open, closed, low, high ...</i>
Stored value	the <i>physical value</i> as stored on <i>disk</i> . Possibly subject to <i>rescaling</i> to fit the size of a storage cell
System	the hardware/software combination that controls a <i>process(1)</i> (includes the <i>computer</i> and the <i>hardware interface</i>)
System or configuration parameter	system wide value. Usually initialized at startup from a value on disk (see <i>node</i>), but may be hard coded or modifiable on-line (exceptionally, for debugging/testing).
User	1. person that interacts with the <i>application</i> ; 2. user agent (browser)
User interface	part of the <i>software</i> supporting the <i>user(1)</i> interaction with the <i>application</i> . Mostly implemented by a collection of dynamic web pages.

A *channel* can be *numeric* (term used interchangeably with *digital*) or *analogic* (sometimes abbreviated to *analog*). The primary identification of a channel is its *channel number*. The software supports a *channel space* of 126 channels, where channels 1 to 63 are numeric, and channels 64 to 126 are analogic. 0 and 127 are reserved values and cannot be used as actual channel numbers.

The input or output direction of a numeric channel is a priori undefined, until it is configured by the user or dictated by the layout of the hardware interface (in which case the user configuration must obviously follow).

Analogic channels are all inputs in the current version, whose hardware interface does not support analogic outputs, as this is not yet required by the current target applications (but upgrading the functionality should not be a big problem).

A channel may be either *physical*, that is associated with an actual hardware I/O channel, or *virtual*, dedicated to some special internal use by the software, in which case it may act as both input and output, as its value is fully controlled by the software.

1.5 The database

1.5.1 Overview

The system's **data base** has one part in active memory (RAM) and another part on external mass storage (disk or SD card).

The part on external mass storage is divided into independent areas called **domains**, each containing a specific type of data : *programs(2)*, channel tables, channel names, message texts, journals etc.

Every domain is identified by a **domain letter** (A-Z), some domains (those that are user editable) by a name. Domains are spread across a number of physical files and are random access, divided into constant length records, subdivided into variable length fields (all records in a domain have the same structure). Two domains may map the same storage space, allowing different views of the data.

The standard record size is currently 128 bytes (the same for all by convenience, not by necessity).

The contents is basically 8-bit text, the contents of a byte being occasionally interpreted as a binary value.

The part in active memory includes the input/output matrix (**I/O matrix**) and the **memory channel tables**.

These structures are set up dynamically at startup time.

The I/O matrix contains an image of the raw values of all channels and is used for data interchange with the process. It supplies the current values for input channels processing and holds the setpoints for output channels.

The memory channel tables contain binary copies of a subset of the channel tables on disk and dynamic values read from the process (mainly the raw and/or the physical value) or resulting from processing. The tables are organized as a linked list, ordered by priority. In order to optimize processing speed and memory usage, only the channels specifically configured by the user have a table in memory (the others have no specific processing or are unused).

1.5.2 The domains

The domains are designed independent in size and structure. Each domain is associated with a data structure that specifies its area on disk, its size in records and the subdivision of its records into fields.

Short access time being a significant requirement, random access was preferred and a same constant record size was set to 128 bytes. As the number of records in a domain is frequently related to the number of channels and some waste of storage space is no longer a problem with current disks or SD cards, all domains (except system and journal domains) have a size of 256 records.

When the software requires that the same data area be viewed as two different structures (one for editing and one for processing), this area is mapped by two different domains.

List of domains

Letter	Editable	Contents
@	yes	Privileged system access (<i>overmaps the whole database</i>)
A	yes	Numeric programs full (<i>overmaps B</i>)
B	-	Numeric programs 1-byte
C	yes	Analogic programs full (<i>overmaps D</i>)
D	-	Analogic programs 1-byte
E	yes	Channel tables full (<i>overmaps F</i>)
F	-	Channel tables by fields
G	yes	User profiles
H	yes	Constants and parameters
I	yes	Values for simulation full (<i>overmaps J</i>)
J	-	Values for simulation 1-byte
K	yes	Macros
L	-	Sequence Control Block info full (<i>overmaps M, function currently not implemented</i>)
M	-	Sequence Control Block info by fields (<i>function currently not implemented</i>)
N	-	Sequential programs (<i>function currently not implemented</i>)
O	yes	Channel and program names dictionary
P	yes	General messages dictionary
Q	-	Journal 0
R	-	Journal 1
S	-	Journal 2
T	-	Journal 3
U-Z	-	Reserved

In this list, 'yes' in normal face means that editing is performed by the standard **catch-all editor** and '**yes**' in bold face that it is done by the dedicated **program editor** or **channel table editor**.

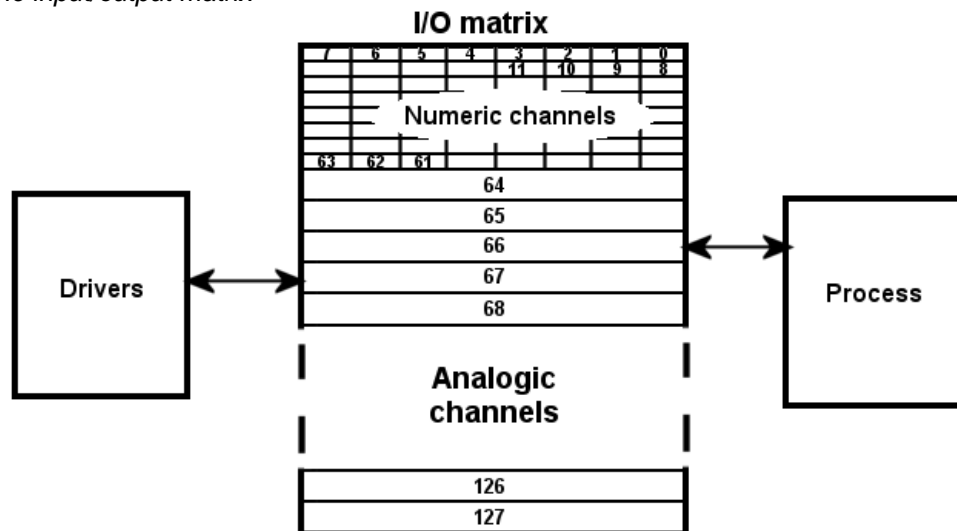
1.5.3 The channel tables

Each possible channel is associated with a collection of static and permanent parameters describing its usage, whose values are stored in an individual channel table in the dedicated E *domain* on disk. The channel number is used as record number for direct access to this table.

A parameter in this table specifies if the channel is idle or involved in some kind of more or less sophisticated processing, in which case a table, containing a subset of the table on disk and a few additional dynamic variables, is 'loaded', that is allocated and initialized in memory, at startup.

A memory channel table cannot be loaded or unloaded at run time : a restart of the application is required.

1.5.4 The input/output matrix



The **I/O matrix** is a permanent central software interface between the *drivers* and the *process(2)*, fully disconnecting their respective activities. It is a table holding the *current raw values* of all channels. On one side, the drivers periodically refresh the values of input channels and copy the values of output channels to the *hardware interface*, on the other side, these values are read or written by the process. This allows the process to run even if the hardware is disabled or missing, which is required for development, debugging and the maintenance of the *database*. In addition, it makes the *raw values* easily accessible for display or modification and facilitates the implementation of test or simulation functions and the addition of drivers supporting new interfacing hardware.

2 I/O hardware configuration

2.1 General

I/O interfacing may be extended to various devices as long as they feature a hardware/software interface connectable to the computer.

2.1.1 Raspberry Pi

The current supported interfaces on the Raspberry Pi are its GPIO pins (except the pins reserved for special functions or protocols) and the interfacing circuits from Microchip supporting the SPI protocol :

- MCP23S17 16-bit I/O expander
- MCP300x and MCP320x series Analogic/Digital Converters (ADC)

A few other peripherals, like a Real Time Clock (RTC) and the Serial Line are directly supported by the Debian Operating System (through the I2C protocol and as terminal device `/dev/ttyAMA0`).

2.1.2 Serial line

For hosts, like Windows PC, which do not feature a GPIO, the I/O hardware interface is managed by a dedicated microcontroller which communicates with the host via a serial line. A simple character oriented protocol provides the means of interchanging digital and analogic data.

2.2 System and configuration parameters

Most hardware configuration parameters are stored in domain H : « Constants and Parameters » as character strings of up to 16 chars. There are « primary » or « master » parameters, which may be immediate (literal) or reference a value or a collection of « secondary » parameters. All primary parameters are collected in the master parameter list which is stored in contiguous master records in domain H. The first master record is 254 by convention, other records are allocated in descending order. Each record contains 8 fields of 16 chars, field 0 being the check field, which must contain the record number, the other 7 fields containing each one of the following :

- the value of a literal parameter as a char string representing an alphabetic name (like a file name) or a numeric decimal value (or hex if starting with 0x). This value is truncated by the first semi-colon, which allows to append a short comment in the unused space if any. Numeric conversion stops at the first non-numeric digit; the numeric value associated with a non-numeric string is zero.

- the 3-digit number of a record if the parameter stands for
 - a record of secondary parameters also stored in domain H.
 - the base record of a cluster of records in another domain, which is user dependent
- a domain/record/field address (format : Drrrff) for any need to reference a literal value stored in another domain (ex : its length exceeds 16 char). This is transparent : one may change a literal parameter to a referenced one, or conversely, any time, provided that the literal value cannot be interpreted as an address.

Any modification of a system parameter requires a restart to take effect.

List of primary parameters :

index	rec	field	name	contents
0	254	0		254 (check)
1		1	S_TOFBASE	temperature evaluation offset base
2		2	S_TRACE	trace masks node
3		3	S_FTRACE	trace file name
4		4	S_SIOPORT	COM port number
5		5	S_SIOSPEED	COM line speed
6		6	S_SPIREC	first SPI node
7		7	S_ADCREC	first ADC node
8	253	0		253 (check)
9		1		reserved
10		2		reserved
11		3	S_LANG	list of supported languages
12		4	S_LOCPATH	path to locale scripts and files
13		5	S_HOME	list of home pages
14		6	S_SCALING	static values scaling params start record in domain H
15		7		reserved
16	252	0		252 (check)
17		1	S_GRAPH	<i>subset</i> graph menus start record in domain O
18		2	S_GRAFCHAN	<i>channel</i> graph menus start record in domain O
19		3	S_NUMMENU	<i>numeric programs</i> menu start record in domain O
20		4	S_ANALMENU	<i>analogic programs</i> menu start record in domain O

2.3 The MCP23S17 I/O expander

The MCP23S17 I/O expander device provides 16 I/O pins. Up to 8 such devices may be addressed on a single SPI bus, that is 128 pins, but with the current design limit of 64 numeric I/O, only a maximum of 4 devices are supported. The MCP23S17 features two 8-pin **banks**, named A and B, which are treated by software as independent devices. Therefore, if n is the number of devices present in the configuration, numbered from 0 to n-1, the valid hardware I/O addresses are in the continuous range 0..(2*n)-1, with an even address for bank A and an odd address for bank B (in other words, if a device/bank address is represented on 3 bits, bits 2 and 1 are the device and bit 0 is the bank).

Each pin can be independently configured as input or output by setting an associated bit in the 'direction' hardware register within the device (0=out, 1=in). In order to support this independence, a small memory database is created in memory and initialized at startup, composed of a collection of *nodes*. Each node contains the information needed for 'bridging' the information flow between the hardware and the application software layer. This information is a table of bytes, each representing a property of the 8 channels of one bank, whose usage as input or output depends on the needs of the application and must be as independent as possible from the channels assignment as seen by the application. Therefore, ideally, this database should allow to associate any I/O pin with any bit in the I/O matrix. To keep things simple and effective, there is one restriction : a bank maps a byte within the I/O matrix pin for bit.

Each node supports from 1 to 8 bits within the same I/O matrix byte and supplies the following information:

Field	Name	
0	SPINODID	the SPI node identifier for validity check. This value must be equal to the record number of the node in domain H. If the check fails, the node is assumed to contain inconsistent data and ignored.
1	MAPIDX	the byte index in the I/O matrix (0..7)
2	SPIADDR	the hardware I/O address (device and bank address, 0..7)
3	MASKIN	the mask of input pins (1=yes)
4	MASKOUT	the mask of output pins (1=yes) As a pin cannot be both input and output, MASKIN and MASKOUT cannot have any same bits set. This applies collectively for all nodes with the same SPIADDR. The generation of the database is terminated if and as soon as conflicting information is found. Having no bit set in both MASKIN and MASKOUT is meaningless.
5	PULLUP	the mask of pins with internal pullup resistor (1=yes) This is only meaningful for input pins.
6	POLARITY	the mask of the polarity of I/O pins (1=inverted)
7	-	unused

The record number of the first node is in system parameter S_SPIREC. Subsequent nodes are allocated (and scanned at startup) in ascending record numbers until a node fails to pass the validity check (field SPINODID).

There can be more than one node describing the same bank or associating it to another byte of the I/O matrix, provided this causes no I/O assignment conflict.

2.4 The MCP3xxx series A/D converters

For analog channels, the bridging process is described by **memory objects** which are initialized by the processing of descriptor *nodes* on *disk*. Each object describes from 0 (if empty) to 16 channels that have the same value acquisition method (i.e they represent the same physical value and have some common parameters). Each descriptor node configures a cluster (a contiguous subset) of channels within an object, providing the linking information directing a value got from the hardware to the target software channel (its associated word in the I/O matrix). Nodes are processed in ascending record numbers. If several nodes provide configuration info for the same channel, only the first one is used, later redundant info is ignored.

Information provided by a node :

Field	Name	
0	ADCNODID	the ADC node identifier for validity check. This value must be equal to the record number of the node in domain H. If the check fails, the node is considered as a terminating node which ends the ADC initialization process.
1	ADCFIRST	number of first analog channel in the I/O matrix (64..126)
2	ADCMPLX	input multiplexor addr of ADCFIRST (0..15) This address is incremented for the next channel in the cluster.
3	ADCSIZE	size of the cluster of similar channels, starting at ADCFIRST ADCFIRST+ADCSIZE < 127 ADCMPLX+ADCSIZE <= 16
4	ADCTYPE	Number of the channel on a multichannel ADC device (each channel is usually dedicated to one type of physical input : electrical, temperature etc.) ADC device dependent, range=0..7 for MCP3xx8
5	ADCDEVICE	ADC device type. Includes signal <i>rescaling</i> : 0 reserved for testing 1 10-bit>>2 (MCP30xx to 8-bit) 2 12-bit>>4 (MCP32xx to 8-bit) 3 10-bit raw (MCP30xx) 4 12-bit raw (MCP32xx)
6	ADCDELAY	delay between input mpx address write and data read (W/R), to allow the signal to settle
7	ADCFLIMIT	filter limit (for the filter algorithm)

The record number of the first node is in system parameter S_ADCREC. Subsequent nodes are allocated (and scanned at startup) in ascending record numbers until a node fails to pass the validity check (field ADCNODID).

As the MCP3xxx series do not support device addressing and there is only one chip select available on the RPi for this functionality, only one ADC device is supported.

Input multiplexing

Between the current or temperature sensor and the analogic input of the ADC device, there is a conditioning circuit adapting the sensor signal to the signal type and level expected by the ADC. Up to 16 sensors of the same type may share the same conditioning circuit and ADC channel when they are connected via an input multiplexor. This mutiplexor is switched by a group of 4 GPIO outputs of the Raspberry Pi making up a multiplexor address. Two such groups are available for selecting the inputs to two groups of different conditioning circuits differing by the delay between input select and input read (current signals should be scanned as frequently as possible but as the conditioning circuit includes a rectifier and filter, 50 or 60 Hz RMS signals may take a second or two to settle; temperature signals are DC signals that vary very slowly, there is no need to hurry in scanning them, but as their conditioning circuit includes only DC amplifiers, the time to settle may be very short).

GPIO pins used for MPX addressing :

Address bits	3	2	1	0
GPIO pins group 0	25	24	23	22
GPIO pins group 1	27	18	17	4

Input filter algorithm

In a home environment, most analogic signals vary moderately slowly (electric consumption, except for switching on and off) to very slowly (temperature). It can therefore be assumed that small fast signal variations are random perturbations or jitter that should be filtered out.

The following algorithm is provided. Let :

<i>NEW</i>	the new value provided by the hardware
<i>new</i>	the filtered <i>NEW</i> passed to the application
<i>old</i>	the previous <i>new</i>
<i>delta</i>	the signed difference between <i>NEW</i> and <i>old</i>
<i>sum(delta)</i>	<i>delta</i> cumulated
<i>flimit</i>	a value provided by the user, delimiting the 'small' delta range below 0 disables filtering

If *delta* is :

- large : one is twice the other or more : *new* is updated immediately to *NEW*
- medium : larger than the user programmed *flimit* : *new* becomes $(NEW+old)/2$
- small : smaller than the user programmed *flimit* : *delta* is cumulated (summed up). When the absolute value of *sum(delta)* exceeds *flimit*, *new* is modified by -1 or +1 and *sum(delta)* is cleared.

2.5 Serial line I/O

Currently not supported on the RPi.

3 The user interface

3.1 Main menu

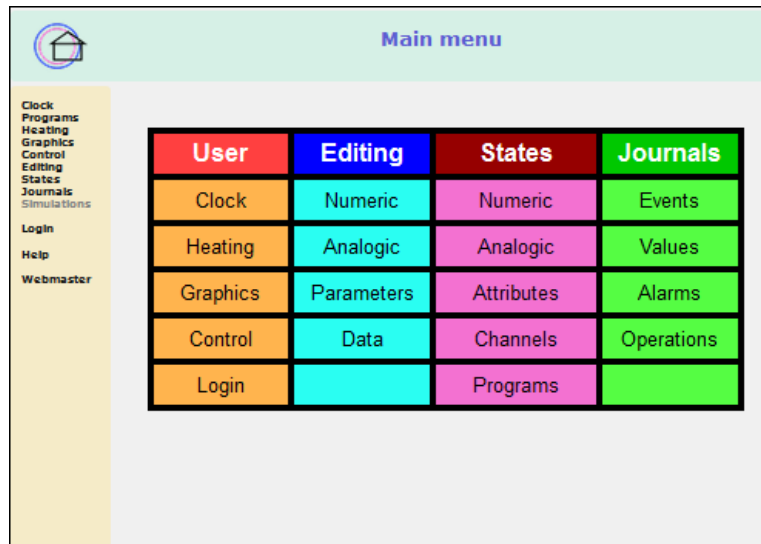
Clock
Programs
Heating
Graphics
Control
Editing
States
Journals

Login

Help

The user interface is made of a collection of dynamic web pages, featuring all the same alphabetic first level menu in their left column. Clicking on one of the entries of this menu either scrolls down a second level menu immediately below the entry or loads another page if there is no second level menu. This menu makes all possible options available within one or two clicks.

The main menu page, which acts as a home page, provides a one click access to the most frequent options.



3.2 Login

User privileges

Each user is assigned a level of privilege, associated with one of the four following categories :

<i>Visitor</i>	no privilege, no login required, has read-only access to a subset of the data
<i>Read access</i>	can view data that is hidden to visitors, but cannot modify it
<i>Write access</i>	may modify data, except if reserved to the administrator
<i>Administrator</i>	full access

Privileged users must login to activate their privileges. The system supports up to 7 privileged users. Only the administrator has access to the user authorization domain, which is not visible to any other user, and can create or delete any user and set privileges (including self, beware !). There should be only one user with administrator privileges, who must be on top of the user's list (user number 1).

Each user transaction requires its own level of privilege and is otherwise rejected with the message « **Privileged operation** ». Write or administrator access is required for any modification of the database.

Login procedure

This page has no complete left column menu, as it is opened on top of the calling page to which control returns after the login is completed.

When clicking on the **Submit** button after having entered a valid username/password combination, the long name of the user is displayed and the caption of the Submit button changes to **Exit**. Clicking on the Exit button validates the new user and the page closes.

If the username/password combination is invalid, the message « **Unknown user** » is displayed. After 3 failures in a row, the login operation is given up.

Logout

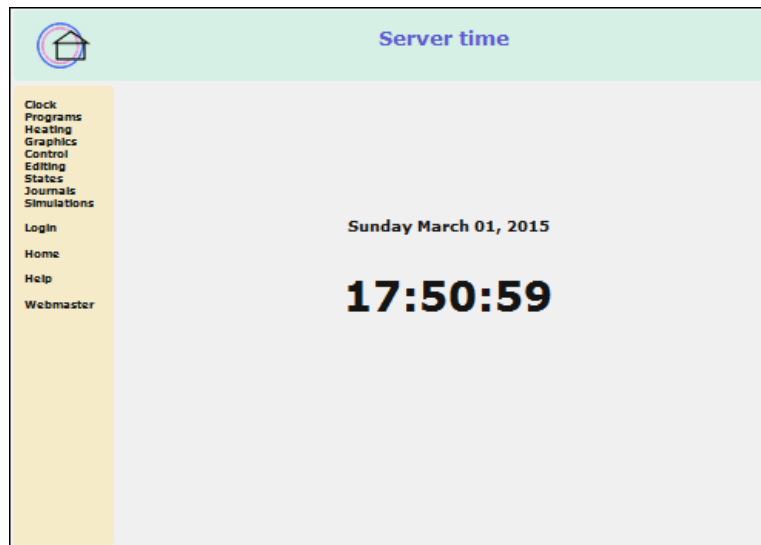
The Login page is also used for logging out. In fact, when the login page is loaded, it immediately logs out the current user, meaning that the user must perform a successful login to recover their access rights. Clicking on the Submit button with both fields blank completes Logout.

Some menu entries may appear grayed out, or even hidden. To make them active, the user must reload the calling page after login (or load another page).

A login is only valid for the current browser session. Exiting the browser terminates the login.

An automatic logout may occur if a session times out.

3.3 Clock



The server uses its own local time for all its internal needs like managing timed events, delays and timestamps for journal entries. This time may differ from the user's time if no time synchronization mechanism exists and obviously if the user is not in the same time zone.

The display format of the date may be language dependent.

The time is shown with seconds. On heavy server or network load, the updating may occasionally miss a second or two.

3.4 Data edition

The database is maintained with several editing interfaces depending on the type of data to be edited. Formatted editing is used for :

- programs, i.e. collections of 96 binary values are binary stored in byte records : *numeric and analogic program editors*, with a click-on-value interface
- channel tables, whose parameters must be easily identifiable : *numeric and analogic channel tables editors* (menu entry : Parameters). There, numeric values must be entered.

Unformatted editing is used for everything else:

- raw text fields with no specific identification : *catch-all editor* (menu entry: Data)

3.4.1 Numeric programs

Numeric program 15

Kitchen heating

Min.	0	3	6	9	12	15	18	21
Day 00-15								
Night 15-30								
All 30-45								
All 45-60								

Controls state selection

Choice OFF ON Cascade Ignore

Clear Paste Copy

A **numeric program** is a time-of-day ordered list of values that are used to set numeric outputs or to compare with numeric inputs all along a day. This function is modelled on the standard electro-mechanical timer shown below.



The standard functionality features 96 values, each valid during a *quarter hour* (15 minutes). Any program may be associated with any numeric channel to make up a timer (output) or an event monitor (input).

This page allows to program the 96 values individually or bulk (the same value for all, or for day or night, or for a whole hour).

To display a page :

- open the **Editing** submenu and click on the *Numeric* entry. Then enter the program number (1..255) in the topmost white field and hit the Enter key twice.
- alternately, a menu allows to select a program by its name within a list up to 20 programs : open the **Programs** submenu and select the *Numeric* entry, then click on the wanted program

name (this list is made up by the system administrator)

To program a value :

- first select this value on the selection bar (this value is reflected in the choice box and in the leftmost column),
- then click on the appropriate quarter hour box, or, for bulk programming, on the hour caption bar or one of the boxes of the leftmost column.

In addition to ON-OFF, the following values may be selected:

- *Cascade*: the value is got from another source
- *Ignore*: no automatic check or action is performed by the system : the channel is left alone.

The 3 buttons allow to *Clear* a program (set all values to *Ignore*) or to make a temporary *Copy* of the program before a modification in order to restore it (*Paste*) to cancel this modification. This is also usable to duplicate a program.

The blinking box denotes the current server time.

The action menu on the top of the left column provides the following functions :

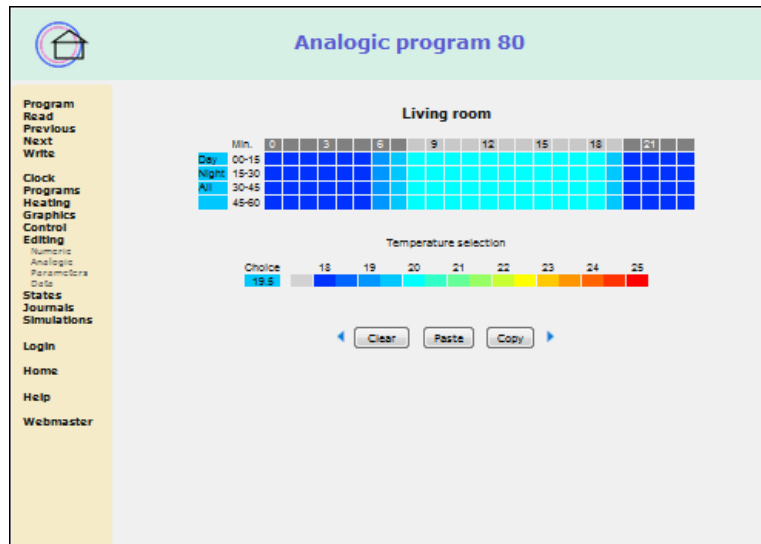
- Program** selection of a new program by entering its number in the top most box
- Read** refreshes the grid of the current program
- Previous** displays the previous program
- Next** displays the next program
- Write** writes the current program to disk, overwriting the old one. Failing to **Write** causes the loss of any modification since the previous Write

3.4.2 Analogic programs

An **Analogic program** is a time-of-day ordered list of values that are used as successive values of an analogic setpoint all along a day.

The standard functionality features 96 values, each valid during a *quarter hour* (15 minutes). Any program may be associated with any analogic channel to provide a setpoint to an external device or to a regulation loop.

This page allows to program the 96 values individually or bulk (the same value for all, or for day or night, or for a whole hour).



To display a page :

- open the **Editing** submenu and click on the *Analogic* entry. Then enter the program number (1..255) in the topmost white field and hit the Enter key twice.
- alternately, you may select a program by its name within a list up to 20 programs : open the **Programs** submenu and select the *Analogic* entry, then click on the wanted program name.

To program a value :

- first select this value as a color on the selection bar (this value is reflected in the choice box and in the leftmost column),
- then click on the appropriate quarter hour box, or, for bulk programming, on the hour bar or one of the boxes of the leftmost column.

In this example, it is possible to choose a temperature comprized between 17.5 and 25.0 °C by half degree increments. This is the default range and should be appropriate for a temperature regulation of a living or working space. Other temperature ranges are available by clicking on the arrow heads left or right of the buttons (the temperature captions line changes).

The 3 buttons allow to *Clear* a program (gray out all values), or to make a temporary *Copy* of the program before a modification in order to restore it (*Paste*) to cancel this modification. This is also usable to duplicate a program.

The blinking box denotes the current server time.

The top-left action menu operates exactly as for numeric programs.

3.4.3 Numeric channel tables

This interface provides the editing capability of all parameters associated with any of the 63 possible numeric channels (numbered 1 to 63). These parameters are all numeric decimal values, stored as character strings in the database on disk in the domain E.

To display a page, use the following procedure :

- open the **Editing** submenu and click on the *Parameters* entry. The following empty page is displayed :

Channel type	
Priority	
Associated Hardware channel	
Associated channel 1 / Program	
Associated channel 2	
Messages base	
Processing Units mask	
Static bits mask	
Dynamic bits mask	
Set point value	
Sequence / Channel 3 / Time unit	
Channel 4 / Increment	

Enter a number in the range 1 to 127

Enter a number in the range 1 to 127

- Enter a channel number (1 to 63 for numeric channels) in the topmost white field and hit the Enter key twice. This fills the right column of the grid with the current values of the parameters of the selected channel.

After the first hit on Enter, the following status message is displayed :

Read : get the new record. **Write** : write the current record to disk

This allows to copy the currently displayed values to the selected channel, providing a channel table duplication function. The second Enter skips this function.

Channel type	
Priority	3
Associated Hardware channel	8
Associated channel 1 / Program	0
Associated channel 2	13
Messages base	10
Processing Units mask	1152
Static bits mask	1542
Dynamic bits mask	130
Set point value	0
Sequence / Channel / Time unit	0

Enter a number in the range 1 to 127

Enter a number in the range 1 to 127

Parameters

Channel type
Priority
Associated Hardware channel
Associated channel 1 / Program
Associated channel 2
Messages base
Processing Units mask
Static bits mask
Dynamic bits mask
Set point value
Sequence / Channel / Time unit

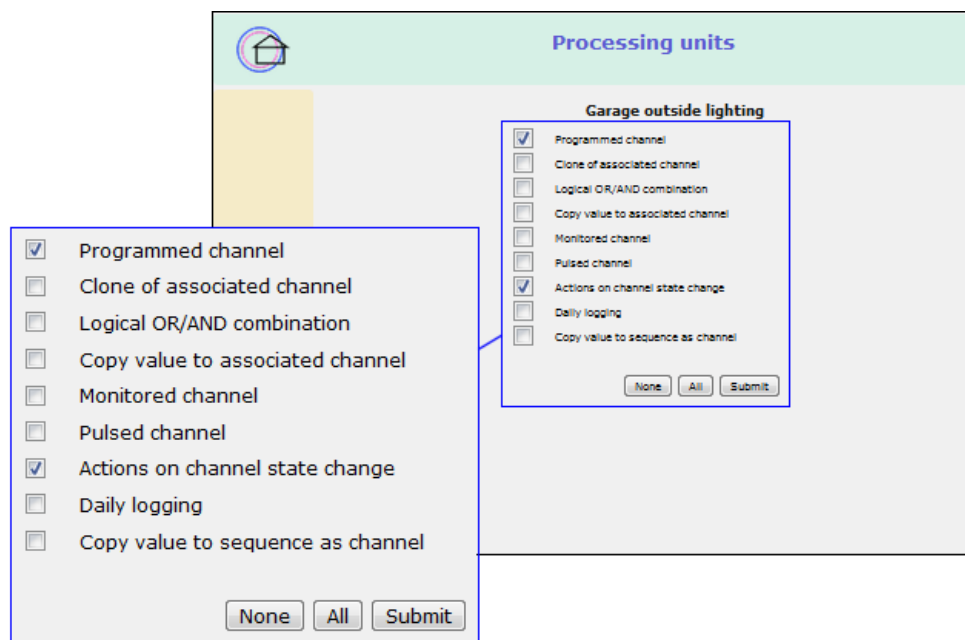
- Edit any value in one of the fields in the right column. Do not hit Enter, use the Tab key or the mouse to change fields.
- When ready to save the new values, click on **Write** in the topleft action menu. This makes the new values permanent. Failing to do this action before proceeding with another operation causes the modifications to be lost. Clicking on **Read** restores the displayed values to their state after the last successful Write operation.

- watch the space below the table for any prompt/status (in black) or warning/error (in red) message. The « **Privileged operation** » message indicates that you did not **Login** or your user privilege level is not high enough and your action was rejected.


The action menu on the top of the left column provides the following functions :

Table	selection of a new channel by entering its number in the topmost box
Read	refreshes the parameter values of the current channel
Previous	displays the parameter values of the previous channel
Next	displays the parameter values of the next channel
Write	writes the parameter values of the current channel to disk

Many of the parameter values represent the encoded selection of a collection of attributes of the channel, which would be cumbersome to the user to be handled numerically. To make it more user friendly, a detailed clear text list of attributes is displayed by clicking on the blue arrowhead at the right of a field, for example for the « Processing Units mask » :



Functions of the buttons :

Check boxes	individual selection of one or several items as attributes of the channel
None	deselects all checkboxes
All	selects all checkboxes
Submit	copy the checked boxes mask value to the calling page and exit
	cancel and exit (browser button)

This page has no left-column menu as it *must* exit to the calling page.

There is no attributes incompatibility detection. Correct selection is under the user's responsibility.

3.4.4 Analogic channel tables

The procedure is the same as for numeric channel tables, except that the channel number ranges from 64 to 127 and the parameter list is slightly modified and extended.

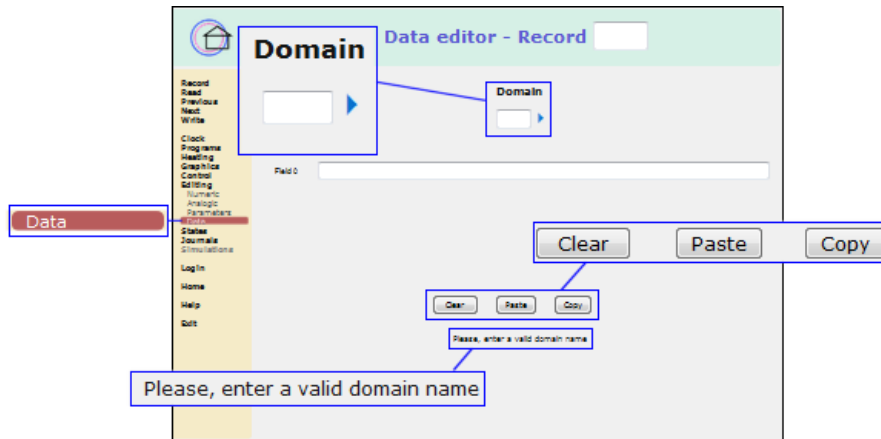
3.4.5 The catch-all editor

This is the general low level maintenance tool of the database, used in all cases not handled by any of the previous editors. It considers the database as a collection of independent domains and reformats its interface according to the field structure of the domain, providing an input field for each of the 8 first data fields within the record currently edited. All fields are alphanumeric. There is no field captions other than Field 0 ... Field 7 as the field contents depend on domain and usage.

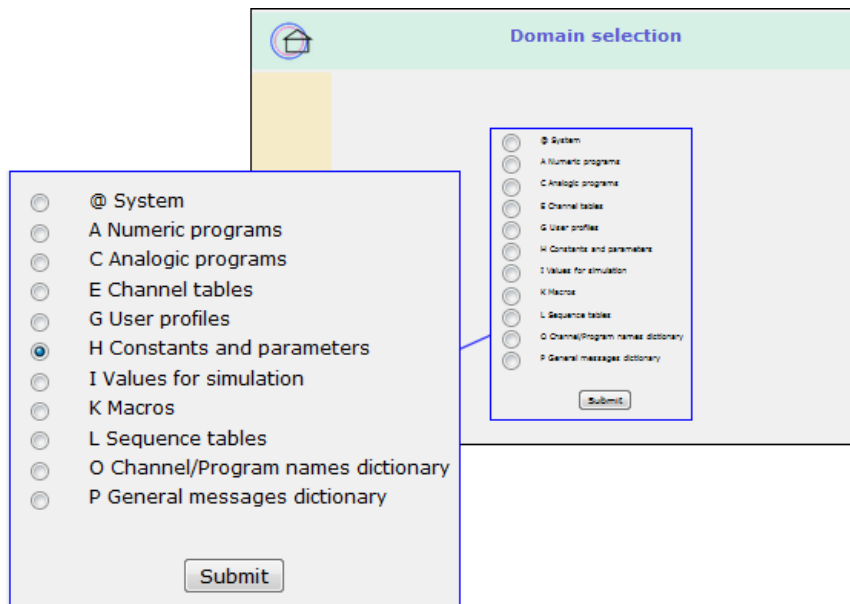
A few domains and fields beyond the eighth are not editable. Domains editable with other editors should not be modified with the catch-all editor.

Editing session

- click on the Data item in the **Editing** submenu. The empty editor page is displayed



- enter the letter of the domain to edit or click on the blue arrow head on the right of the domain field for opening a domain selection page :



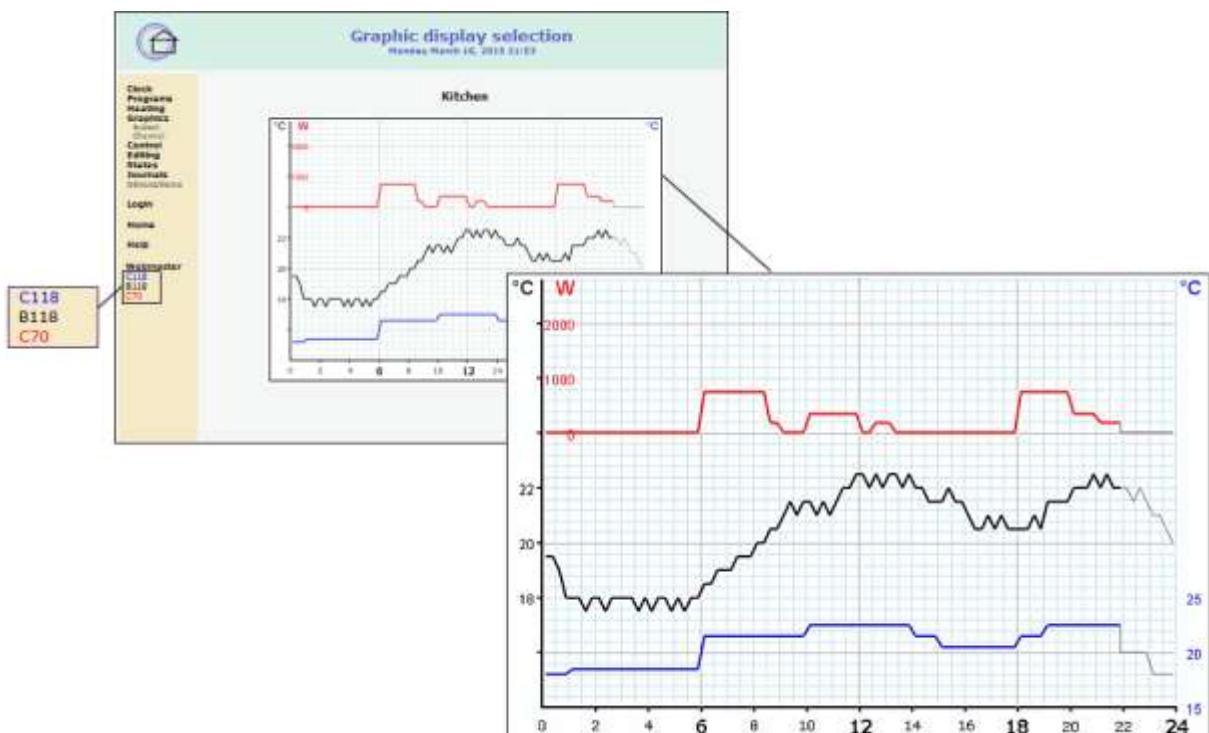
- click the radio button of the domain to select and Submit. The selection page exits, the letter is loaded into the domain field and up to 8 empty edit boxes appear.
- enter the record number and hit the Enter key twice. The boxes are filled with the contents of the fields of the selected record.

Field	Value
Field 0	95 act time %
Field 1	8
Field 2	20
Field 3	%
Field 4	-2500

- for editing operations and saving the modifications, do as with the channel table editor.

3.5 Graphics

This feature displays up to 4 curves, each with its own vertical graduation, on the same chart. It uses the daily logging function and its collection of 96 values for each channel which has this functionality enabled as processing unit and in its static bits mask. A list of up to 4 channels is appended as argument to the page URL and the corresponding daily values are displayed as curves with each its own colour, which is also the colour of the graduation and of its channel number which appears on a clickable list in the left column. Clicking on one of these numbers displays the table page of the channel.



The curves represent the evolution of today's physical values from 00h00 to now (on the example above : 22h00). The parts of the curves that follow 'now' are yesterday's values, in light gray. There is one value for every quarter hour (15 minutes interval, 96 values for a whole day). For technical reasons, there is no time alignment at startup, meaning that the timing on the chart may be shifted by up to one value. The chart is automatically refreshed every half-quarter.

3.5.1 The channels list

The list appended to the page URL is composed of a hash character (#) followed by up to 5 channel identifiers separated by underscores. Example :

.../celsiusA.htm#A216_b118_C118_C70

A channel identifier is an alphabetic source letter followed by a channel number. The source letter indicates the origin of the data :

- A Chart caption. The channel number is the index of the entry in domain O
- B Setpoint. The channel number is the index of the entry in domain A
- C Daily logging values. Nothing if the channel is not correctly configured.
- D Simulation values. The channel number is the index of the entry in domain I
- E Same as C
- F-Z Ignored

As the same channel may provide data from different sources, it may appear more than once in the list, with different source letters, but only two occurrences are fully supported, one with an upper case source letter, the other with a lower case source letter.

3.5.2 Display configuration

Each curve and its graduation are configured by a set of 8 parameters contained in a record of the system parameter domain H. The *Channel type* entry in the *channel table* is a relative index within a cluster of such records (which starts at the record number in system parameter S_SCALING). The first 4 records have names suggesting their expected use :

Index	Name
0	(Reserved)
1	Temperature
2	Numeric variable
3	Percent
4	Electric power

But these and all the next ones can be freely reprogrammed.

A *lower case* source letter adds 8 to the *Channel type*. For example, if channel 70 has a channel type of 1, the channel type would be 1 for B70 and 9 for b70. This must be carefully considered when allocating channel type numbers and records to avoid collisions. Otherwise, the source letter is case insensitive.

3.5.3 Configuration parameters

Field	Contents	Example
0	channel plotting type	1
1	slope	5
2	offset	225
3	physical unit name or symbol	°C
4	graduation range min	-225
5	graduation range max	300
6	horizontal position of vertical scale	1
7	visible part of the graduation	70

channel plotting type

- 1 Slant - a point is joined to the next by one slanting segment
- 2 Step - a point is joined to the next by a vertical segment followed by a horizontal segment

slope / offset

As the raw physical values are stored in bytes, these 2 parameters are used to rescale these values according to the formula :

$$\text{displayed_value} = (\text{slope} * \text{raw_value}) - \text{offset}$$

Note the minus sign before the *offset*.

For a representation as integers, the values entered for these parameters must be magnified 10 times.

physical unit name or symbol

Any unit name like °C % kW m³, etc. or blank for no display.

The symbol is displayed on top of the grid, regardless of the position of the graduation.

graduation ranges min / max

The software uses these values to select a graduation that is easy to read and reevaluates the displayed values accordingly. Negative as well as positive values are possible.

The values entered for these parameters must be magnified 10 times.

horizontal position of vertical graduation

0 default = channel plotting type

1 far left

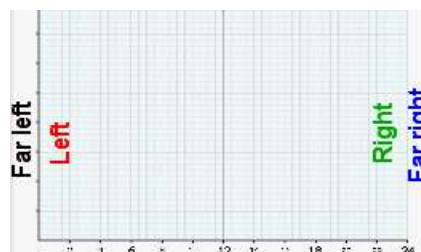
2 left

3 right

4 far right

5 left enlarged

6 far right enlarged



visible part of the graduation

A double digit decimal number :

units : lowest visible value (0..7)

tens : highest visible value (>= lowest)

The vertical axis can receive up to 8 equidistant graduation numbers, termed 0 to 7, bottom up. Use this feature if only a few of them should be visible or to avoid overwriting.

Example (as used with the diagram above) :

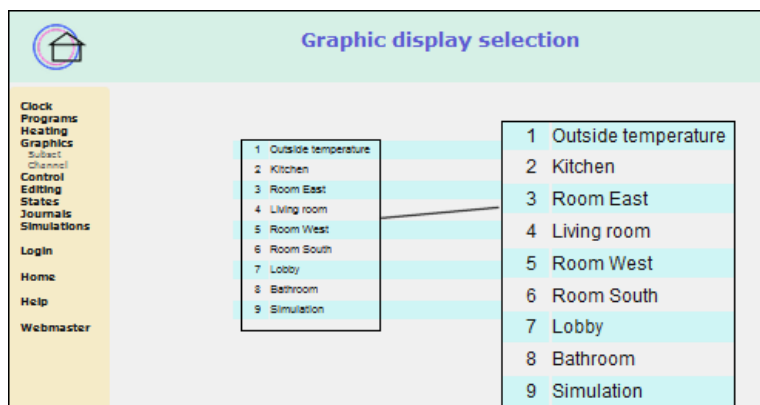
Channel Id	b118	C118	C70
Type	1	1	4
Rec. index	9	1	4
Field 0	1	1	1
1	5	5	100
2	225	225	0
3	°C	°C	W
4	160	150	-50000
5	500	250	30000
6	4	1	5
7	20	42	75

3.5.4 Named lists of channels

Two user programmable sets of named lists are available as clickable menus :

Subset : display a collection of related channels

Channel : display one channel



There are 2 sets for commodity, but in fact they are identical in operation, differing only by their names in the main menu and the number of their base records in domain O (system parameters S_GRAPH and S_GRAFCHAN).

To add an entry to one of the 2 sets, use the catch-all editor, open domain O and select the first blank record at the end of the set. then enter the name of the new list in field 0 and the channel list (without the leading hash character) in field 1. (A record is blank if its first character in field 0 is a space).

Of course, it is always possible to modify a list any time, but you cannot remove a list without closing the gap, as a blank record acts as a set terminator.

3.6 User controls

This page displays a list of numeric output channels that can be directly set or reset by the user. The colour of the box shows the current state of the channel : blue=OFF, red=ON. Clicking on a box toggles the state of the channel. Clicking on the channel name opens the channel table page.

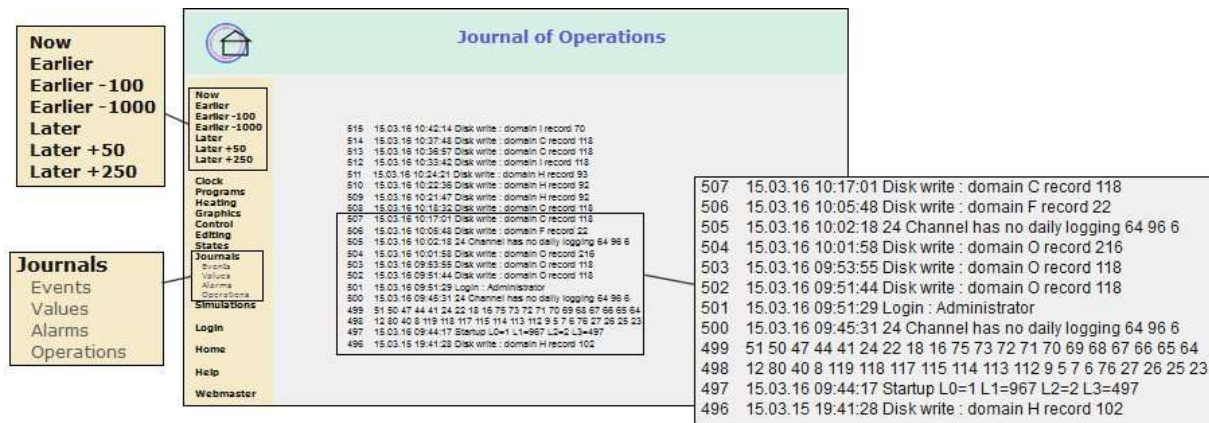
Channel Number	Channel Name	State
9	Room East heating	OFF
10	Living Room heating	ON
11	Room West heating	OFF
12	Room South heating	OFF
13	Lobby heating	OFF
14	Bathroom heating	ON
23	Entrance door	ON
24	Entrance lighting	ON
38	Garage door	OFF
39	Garage lighting	ON
40	Garden watering	OFF

The contents of the list is user defined by checking the box « User control » in the entry « Static bits mask » of the channel table. It is the user's responsibility to evaluate if this function would conflict with other functions configured on the channel.

The list may contain up to 32 channels ordered by increasing channel numbers, in two pages (commands **Previous** and **Next**).

3.7 Journalling

The system generates or witnesses *events*, which it has the possibility to record in one or several of 4 journalling domains. The logged information can be accessed any time by the user by calling one of 4 journal pages.



An event is a significant change in the state of the system, for example :

- system startup
- the change of state of a numeric channel
- the crossing of a threshold of an analogic channel
- the action of an operator (login or modification of the database)
- the change of date for channels with daily logging

For commodity, the 4 journals are named *Events*, *Values*, *Alarms*, *Operations*. This provides a rough sorting by event type, but imposes no restriction to which journals an event is directed to, except that the Operations journal is only accessible to the system administrator.

3.7.1 Viewing the contents of a journalling domain

When a journal page loads, it displays the 20 most recent messages in reverse chronological order. Click on **Earlier** to get the 20 immediately older messages and on **Later** to get the 20 immediately more recent messages (if possible). As a journalling domain can contain up to one million entries (depending on the size of the associated file), commands are provided for moving the message index back and forth by several hundreds of messages. Click on **Now** to get back to the most recent messages.

3.7.2 Managing the message library

Each recordable event is associated with a number, which is the record number of its associated text in domain P.

- system message numbers are hard coded and allocated in decreasing values, starting at 255
- the number associated with a channel is in entry *Message base* of its channel table. If several messages are needed for the same channel, they are collected into a message cluster allocated upwards and starting at *Message base*.

Each message in domain P is in fact a message format string including formatting directives, which are percent characters followed by one letter representing an action on an associated argument (similar to the *printf()* function of the C standard library). The server parses the string and appends to it the string argument corresponding to each directive found and sends the complete string to the *client(2)*, which formats and displays it. The arguments are separated by carets (^). Percent signs and carets must not appear alone in the format string, they must be escaped by a percent sign. In a few cases, where the resulting length of the string would cause overflow problems, the original format string is itself patched.

The system administrator has the responsibility to populate the domain P with format strings with the correct syntax.

List of available formatting commands :

Specific means that the associated argument is not supplied by the system (like date and time) and is dependent on the context in which the message logging call is performed. These letters require the knowledge of the ordered list of available specific arguments.

Letter	Action	Specific
^	escaped caret. Passed as an argument containing '^ ;'	
%	escaped percent. Passed as an argument containing '%'	
d	argument converted to decimal string	yes
H	current system date as YYMMDD	
h	current system time as hh :mm :ss	
i	argument converted to decimal string	yes
K	must be followed by 4 dummy characters : %K.... The six chars are overwritten with the system date YYMMDD	
n	number of the used format string	
N	no action, does not generate an appended argument. Provided for allowing the <i>client(2)</i> to insert a local argument	
p	tells the client to skip the associated argument	yes
s	special. Used to pass ^ and % as arguments. Do not use.	
u	must be followed by a double digit decimal number (possibly with a leading 0) and associated with a numeric argument. The sum of this number and the argument is the index into the <i>state words table</i> (supplied by the client). %udd is then replaced on the client side by the indexed state word (<i>off, on, open, closed, low, high...</i>)	yes
Other	Reserved, must not be used	

Example : Assume 2 specific arguments with the values 7 and 31 and the following format string at number 61 :

```
1%H%h %n Room %i hygrometry %d%%
```

Before storing it in Journal 0, the server would convert it to :

```
1%H%h %n Room %i hygrometry %d%s^150317^13:47^61^7^31^%^
```

The client could reformat and display it as :

```
17/03/15 13:47 61 Room 7 hygrometry 31%
```

Journal files selector mask

The first char of a message text may have a special meaning, allowing to select from 0 to 4 journal files to receive the entry. If this char is :

- '0', this message is ignored (used for testing or for disabling a given message)
- in the range '1'..'?', the four low order bits of its code are used as a mask of the journal files to receive the entry
- else, the message is sent to journal 0

Standard specific arguments

Numeric channels

- channel number
- channel state : OFF=0, ON=1

Analogic channels

- channel number
- channel transition : normal->low=0, low->normal=1, high->normal=2, normal->high=3
- channel full setpoint (including the additional temporary increment/decrement)

3.8 States

States

Numeric
Analogic
Attributes
Log values
Channels
Programs
Language
Home
Version

The **States** submenu provides a number of functions mostly dedicated to maintenance and development.

Numeric : A table of the current state of the numeric channels 1 to 59 (channels 60 to 63 are not visible on this chart). Blue=OFF, red=ON. You can change the state of a channel by clicking on its box. The new state depends on the selected action by clicking on the selection bar (OFF ON or Toggle). Of course, clicking on an input

channel or an output channel directly controlled by the system is not meaningful. The table is refreshed every 10 seconds with the true state of the channels.

Analogic : A table of the current *raw values* of the analogic channels. Green boxes denote channels that have a channel table containing a *physical value*, which can be displayed for a few seconds by clicking on the box.

Attributes : A table displaying for each channel a bit pattern representing 5 of its attributes. Bits are numbered from right to left and stand for the following attributes :

Bit	Attribute
4	Channel has a daily logging
3	Channel has a channel table
2	Channel with user function
1	Channel write locked
0	Channel numeric state

Log values : By selecting a channel by its number in the top most box, its current 96 daily logging *physical* values are displayed, provided this channel has a channel table and this function enabled.

Channels : A list with on each line the channel number followed by the name assigned to the channel or blank if no assignment. This provides a synopsis of all assignments and an aid to channel allocation during application development. Click on **Previous** or **Next** to display a new set of 20 channels.

Programs : As for **Channels**, a list with on each line the program number followed by the name assigned to the program or blank if no assignment. This provides a synopsis of all assignments and an aid to program allocation during application development.

Languages : This page displays a list of available languages.

Clicking on a language switches the user interface to the selected language.

Depending on the implementation, there are two possible language modes :

Static

- Only one language is concurrently supported.
- This language is selected at installation via a configuration option.
- This language selection page is not available.
- This mode is expected to be the general case, the only mode supported by some implementations.

dynamic

- The language can be switched during operation.
- There may be only one language at a time, valid for all users.
- Administrator privileges are required to switch the language.
- Text modifications made by the user to the database remain in the language used.
- Existing entries in the journal files are not changed.
- This mode applies to advanced specific cases or demo configurations.

The language list is limited to 7 languages.

It displays a subset of the available languages and is user definable and modifiable.

In fact, a user community is expected to use no more than two or three languages, or frequently only one, restricting this functionality to a mere demo object.

As some browsers keep javascript files in cache even when the calling HTML page is reloaded, switching to a new language may require the cache to be manually cleared (command available as CTRL/F5 on most browsers, but not available to a web page for obvious reasons).

Home : This page displays a clickable list of web pages. The selected page becomes the page that is loaded when clicking on the **Home** entry of the main menu.

Version : This page displays an information message about the current server system.

3.9 Simulations

This entry of the main menu is only available to the system administrator. It provides access to the editing capabilities of domain I, the simulation values domain, and to possible other simulation scenarios (temporary and unsupported). Editing operations are identical as for programs (submenu **Editing**).

4 Processing units

4.1 Overview

Every channel may be configured to be the subject of one or several operations performed repeatedly by processing units (**PU**). PU are organized in a hierarchy, that is, if several PU apply to the same channel, they are executed in a strict **order** to make the latest results of earlier PU available to later PU.

Some PU are specific to numeric channels or to analogic channels, others are common to both, but are independent in the order of execution.

For each channel, the user decides the list of associated PU by crossing the appropriate check boxes on the PU page of the User Interface.

PU may conflict with each other or use a resource for different purposes : care must be taken not to apply these PU to the same channel simultaneously.

Each PU is rescheduled at some given time interval, according to its **sched** index :

Sched	Rescheduling interval
0	100 ms (clock tick)
1	1 second
2	9 seconds (100 times every 15 min)
3	1 minute
4	15 minutes (quarter hour)

4.1.1 Numeric channels

The value of a numeric channel is the value of its associated bit in the *I/O matrix*.

This bit can be either the image of a numeric HW input, which is volatile and likely to change state any time (*physical channels*), or manipulated by software (*virtual channels*). Having both is not very meaningful and should be avoided (or restricted to testing/debugging).

When a PU is processed, the channel value found in the I/O matrix is used without any assumption on when and how it was set.

For numeric channels, *hardware value* and *physical value* are the same (or complemented if negative logic).

List of current processing units :

Order	Sched	Name	Function
0-1			Reserved
2	4	progdchan	Programmed channel
3	0	clonechan	Copy the value of a channel from its associated channel
4	0	dcombo	Logical combination of 2 channels
5	0	dtoggle	Channel toggling
6			Reserved
7	0	copychan	Copy the value of a channel to its associated channel
8	4	monitdchan	Monitored channel
9	0	dpulse	Pulsed numeric channel
10	0	dchange	Actions on numeric channel state change
11	4	dlogchan	Daily logging
12	0	copyseq	Copy the value of a channel to its seq channel
13-15			Reserved

4.1.2 Analogic channels

The *raw value* (or *hardware value*) of an analogic channel is the value in its associated word in the I/O matrix. It usually comes from a hardware sensor and in a few cases from a software source.

Its *physical value* is usually computed by applying a (most frequently linear) transformation to the raw value, but may also be copied from another channel or supplied by software. The *raw* and the *physical* values are integers, possibly rescaled to fit some storage size (byte or short integer). The physical values, although usually not directly shown to the user, are chosen to be consistent within a group of similar channels (i.e. handling quantities of the same kind and range) and to be directly comparable to channel independent threshold values. For example, the supported temperature range is as following :

Physical (unsigned integer)	1	45	85	127
Temperature (°C)	-22	0	20	41

Its *display value* may be its physical value or a linear transformation of the physical value in order to produce a presentation value meaningful to the user in terms of decimals and physical units. This transformation uses channel independent parameters (in the example : 0.5 and -22.5) and is performed by the client executing a Javascript procedure using physical values as input.

An analogic channel may also be a host, that is a repository of parameters and variables for a Processing Unit elaborating a more or less complicated behaviour involving other channels. In that case, it loses its analogic channel properties and some of the contents of its channel table are reassigned to different PU specific purposes.

List of current processing units :

Order	Sched	Name	Function
0			Reserved
1	0	avalue	Analogic channel value evaluation
2	4	progachan	Programmed channel
3	0	clonechan	Clone of associated channel (<i>same as numeric</i>)
4	0	dcombo	Logical combination of 2 channels (<i>same as numeric</i>)
5-6			Reserved
7	0	copychan	Copy the value of a channel to its associate channel
8-9			Reserved
10	0	achange	Actions on analogic channel state change
11	4	alogchan	Analogic channel daily logging
12	0	copyseq	Copy value to channel specified in Seq (<i>same as numeric</i>)
13	0	blink	Blinking a numeric output channel (<i>hosted</i>)
14-15			Reserved

4.2 Numeric channels

4.2.1 Channel table parameter names

The parameter names in the following tables are used in the description of the Processing Units. Some entries may have more than one name, they are synonyms identifying the same parameter used in different contexts .The description uses one of them, depending on the relevant context.

Name	Contents
Chan	Channel number
Ctype	Channel type
Cprio/Cnext	Priority
HWchan	Associated Hardware channel
Chan1/Prog	Associated channel 1 / Program
Chan2	Associated channel 2
MSGclus	Messages base
PUMask	Processing Units mask
Stabits	Static bits mask
Dynbits	Dynamic bits mask
Progval	Set point value
Chan3/Seq	Sequence / Channel 3 / Time unit
Chan4/Delta	Channel 4 / Increment

In the following, the owner of the PU is called the *master* if its value is used as input or the *target* if its value is set by the PU.

Names of the bits in the static bits mask

Name	Function
D_INITIAL	Value at startup
D_OFFLOG	Log ON→OFF transition
D_ONLOG	Log OFF→ON transition
D_HYST	Logical value of hysteresis zone
D_NOWRITE	Channel write protected
D_DAYLOG	Daily logging
D_USERCTL	User control
D_COMBO	Logical channel combination (0=OR, 1=AND)
D_LDSETVAL	Load static value from disk

4.2.2 progchan - Programmed channel

A channel may be time programmed, with 96 different values in a day, one per quarter hour. The entry Prog of the channel table contains the number of a program within the program domain. This number is encoded as following :

bits	meaning
7-0	program number (2..252, if == 1, the feature is disabled. Note : 0 defaults to channel)
7	calendar usage flag : if (program >= 128) the value returned by the calendar function (not yet fully supported) is added to the program number (implemented : 0=working days, 1=saturdays, 2=sundays; considered : 3=user defined days)
	Good practice for programmed channels intending to use this feature : organize the upper half of the program domain in 4-program clusters aligned on multiples of 4.

4.2.3 clonechan - Copy the value of a channel from its associated channel

Chan1 → target

The value of the associated channel Chan1 is copied to the value of the target, which becomes a clone of Chan1. This allows to make more than one clone from the same associate.

4.2.4 dcombo - Logical combination of 3 channels

This unit combines 3 channels (Chan1, Chan2, Chan4) to generate the value of the target. The logical combination depends on flag D_COMBO (Logical channel combination in the static bits mask)

D_COMBO	HWchan	Logical combination
0	<128	Chan1 OR Chan2 OR Chan4
0	>=128	Chan1 XOR Chan2 XOR Chan4
1	x	Chan1 AND Chan2 AND Chan4

Action depends on the type of the target :

- numeric : the value becomes the new value of the target
- analogic : if value=0, the target value is cleared, else, it remains unaffected

If an input channel is not used, its number must be set to 0 for OR (always returns 0) and 128 for AND (always returns 1). For XOR, it may be either, depending on the wanted result.

4.2.5 dtoggle - Channel toggling

This unit has two flavors, depending on having one or two input channels (Chan1, Chan2) configured :

- one channel (Chan2 set to 0) :
 - a 0 → 1 transition on Chan1 toggles the target.
- two channels :
 - a 0 → 1 transition on Chan1 sets the target (precedence)
 - a 0 → 1 transition on Chan2 resets the target

In any case, 1 → 0 transitions are ignored.

4.2.6 copychan - Copy the value of a channel to its associated channel

master → Chan2

The physical value of the master becomes the physical value of Chan2.

If Chan2 is numeric and the master is analogic and in the hysteresis zone, the last master's state outside the hysteresis zone applies.

If Chan2 is numeric and has bit 7 set, the master's value is complemented before use.

This works also if Chan2 has no channel table.

It may be used to duplicate an analogic channel if both the master and Chan2 are analogic.

Copying numeric to analogic is a special case : Chan2 is left unchanged if the master is set and cleared otherwise.

4.2.7 *monitdchan - Monitored channel*

A channel may be time monitored, with 96 different values in a day, one per quarter hour. This is similar to progchan, except that instead of setting, the value of the channel is compared to the programmed value. If they are equal, Chan4 (the « alarm output ») is reset, else Chan4 is set and the « Action on change » function is called to evaluate the system's reaction.

Chan2 is used as an enable-disable input.

4.2.8 *dpulse - Pulsed numeric channel*

On normal operation, this channel delivers a pulse of specified constant width when the logical combination of the 2 control channels (Chan1 and Chan2) goes high, regardless whatever this combination does afterwards (going low during the pulse or staying high after the pulse).

To deliver a new pulse, both input and output must have gone low.

This PU is incompatible with those using the same resources.

Resources:

HWChan	pulsed output			
Chan1	control input channel 1	logical combination		
Chan2	control input channel 2	selected by D_COMBO		
Progval	pulse length (in time units specified in Delta)			
Delta	time unit (sched index)			
D_COMBO	Chan1 Chan2 logical combination selector (0=OR, 1=AND)			
D_HYST	output channel value when idle			
	0	true pulse	input: ____----	output: ____--__
	1	delayed command	input: ____----	output: -----__

4.2.9 *dchange - Actions on numeric channel state change*

These actions depend on the values of flags D_ONLOG, D_OFFLOG, D_ONSEQ, D_OFFSEQ

If D_ONLOG or D_OFFLOG set, the corresponding state change (OFF→ON or ON→OFF) triggers the insertion of a status message into the log file(s), with, as parameters, the channel number and its final state.

If D_ONSEQ or D_OFFSEQ set, the corresponding state change starts the associated sequence.

Associated state words :

0 ON → OFF	0 OFF	2 OPEN	4 OUT OF ORDER	10 END
1 OFF → ON	1 ON	3 CLOSED	5 WORKING	11 START

4.2.10 *dlogchan - Numeric channel daily logging*

Every 9th second, in the logging buffer, the byte indexed by the current quarter hour is incremented if the channel is set. This results in a value in the range 1..100 representing the % of 15 mn time during which the channel was set, which increases the precision of the graphical display.

4.2.11 *copyseq - Copy the value of a channel to its seq channel*

master → Chan3

This is the same as copychan, except that the Chan3 (Seq) number is used as associate channel number. This would be incompatible with sequence management in numchange and pulsed channel

4.3 Analogic channels

4.3.1 Channel table parameter names

The parameter names in the following tables are used in the description of the Processing Units. Some entries may have more than one name, they are synonyms identifying the same parameter used in different contexts. The description uses one of them, depending on the relevant context.

Name	Contents
Chan	Channel number
Ctype	Channel type
Cprio/Cnext	Priority
HWchan	Associated Hardware channel
Chan1/Prog	Associated channel 1 / Program
Chan2	Associated channel 2
MSGclus	Messages base
PUMask	Processing Units mask
Stabits	Static bits mask
Dynbits	Dynamic bits mask
Progval	Set point value
Chan3/Seq	Sequence / Channel 3 / Time unit
Chan4/Delta	Channel 4 / Increment
Coefa	Conversion coefficient A
Coefb	Conversion coefficient B
Coefc	Conversion coefficient C
Hyst	Hysteresis

In the following, the owner of the PU is called the *master* if its value is used as input or the *target* if its value is set by the PU.

Names of the bits in the static bits mask

Name	Function
A_NLOWLOG	Log normal→LOW crossing
A_LOWNLOG	Log LOW→normal crossing
A_NHIGLOG	Log normal→HIGH crossing
A_HIGNLOG	Log HIGH→normal crossing
A_NOWRITE	Channel write protected
A_DAYLOG	Daily logging
A_USERCTL	User control
A_OUTCHAN	Action on threshold crossing
A_OUTMODE	Action mode on threshold crossing
A_COMBO	Logical channel combination (0=OR, 1=AND)
A_LDSETVAL	Load setpoint value from disk

4.3.2 achange - Actions on analogic channel state change

An analogic channel is considered a three state device depending on its value being below, within (=normal) or above the normal range (also called hysteresis zone), which is comprised between a LOW threshold (setpoint - hysteresis) and a HIGH threshold (setpoint).

Each of the 4 possible transitions LOW ↔ Normal ↔ HIGH may conditionally cause any or all of the following:

- a log message to be issued
- a sequence to be started
- one or two numeric channels to be set or cleared

The first of a pair of numeric channels is Chan2. The second is Chan2+1

Action mode is selected by flag A_OUTMODE :

A_OUTMODE 0=one channel, 1=two channels, according to the following truth table :

A_OUTMODE	0	1	
	Chan2	Chan2	Chan2+1
HIGH	OFF	OFF	ON
Hyst zone	no action	OFF	OFF
LOW	ON	ON	OFF

The action outside the hyst zone is inverted if Chan2 has bit7 set.

Flag A_OUTHYST retains the last transition normal → LOW (=0) or normal → HIGH (=1) when the channel is back to the normal (hysteresis) zone. This is assumed to be the 'numeric value' of the analogic channel in situations where a binary ON/OFF value is required.

4.3.3 *alogchan - Analogic channel daily logging*

A new value is written to the log buffer every quarter hour. It is calculated by adding up the current value every 9 sec (100 times during the quarter hour) and dividing by 100 at the end of the quarter hour.

4.3.4 *avalue - Analogic channel value evaluation*

Linear conversion : the hardware value is converted into a physical value with the formula :

$$y = ((a*x)/c) - \text{tofbase} + b \text{ with } y \geq 0$$

If c=0, c=1 is used.

As this is integer arithmetic, a/c allows to define fractional multipliers in the range :

{ 0, 1/255=0.003921, .. 255/1 }

4.3.5 *blink - Blinking a numeric output channel*

This is an example of an analogic channel hosting a non-analogic function.

It is triggered by the AND or OR logical combination of 2 numeric inputs, produces a given number of pulses on a numeric output (with user programmable parameters pulse length and interval), followed by a longer pulse on the same output, then by a "final action" pulse on a second numeric output.

Example of time diagram : assume 4 pulses and a last long pulse :

blink sequence _____ -- _____ -- _____ -- _____ -- _____ -----
final action _____ -----

An alternate configuration makes the blinking last forever.

Resources :

HWchan	final action output channel
Chan1	control input channel 1
Chan2	control input channel 2
Chan3	pulsed output channel
Delta	timelength multiplier for the last warning pulse
Ctype	number of pulses before final action
Coefa	warning pulse length (in ticks)
Coefb	interval between warning pulses (in seconds)
Coefc	timelength of the final action pulse (in ticks)
D_COMBO	Chan1 Chan2 logical combination selector (0=OR, 1=AND)

4.3.6 *progachan - Programmed channel*

A channel may be time programmed, with 96 different values in a day, one per quarter hour. After rescaling, the current value is written to the Setval entry of the channel table.

The entry Prog of the channel table contains the number of a program within the program domain of the data base. This number is encoded as following:

bits meaning
 7-0 program number (1..255, 0 defaults to the channel number)
 7 calendar usage flag : if set, the value returned by the calendar is added to the program number (supported : 0=working days, 1=saturdays, 2=sundays ; considered : 3=user defined days)
 A set of 4 consecutive program entries must be allocated for each channel using this feature (the same set may be shared by several channels with identical programming requirements)
 Note: the programs in the range 1..127 are not affected and should be used if this feature is not required.

5 Application examples

5.1 Heating regulation



Assume a house with a living room, three bedrooms and a kitchen, all equipped with electric heaters. You want to have an independent temperature regulation system for each room.

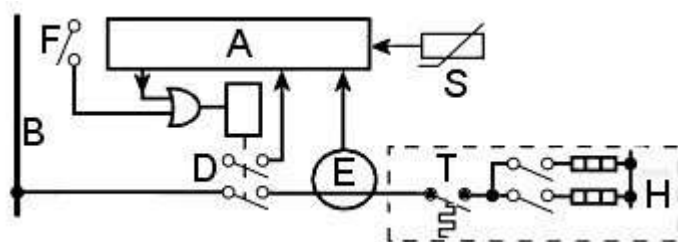
The user interface would be based on a schematic layout of the house, showing a set of relevant information for each room :

Name of the room
Current temperature
 Required temperature
 Electric consumption

The lobby and the bathroom have a heater, but no independent regulation. There is an outside temperature sensor on the terrace.

Each heater tells its current state by a number (0 to 7) and a corresponding color.

Each room is provided with the following heating circuitry :



The heater H is connected to the mains B through a power switch D and a current sensor E.
 The power switch D is operated either by the computer A or manually via the switch F (logical OR).
 The heater has a local thermostat T and one or several heating elements that may be locally switched ON or OFF.
 The power switch D has an auxiliary contact whose state is reported to the computer.
 The current sensor E supplies an analogic value, proportional to the current.
 An independent sensor S reports the room temperature to the computer .
 The state of a heater is represented by a 3-bit value :

- bit 0 : command issued by the computer to the power switch
- bit 1 : state of the power switch (0=open, 1=closed)
- bit 2 : current flowing through the heater (0=no, 1=yes)

Some of these 8 states are normal operation, others show a malfunction :

- switched by an output channel either programmed or controlled by an input channel triggered by a dusk photocell

5.4 Intrusion detection

- detected by some sensing device (electromagnetic, volumetric, infrared) triggering a monitored input channel
- action by one or several output channels (lighting, sound alarm, door locking..) controlled by the input channel
- event journalling

6 Installation

6.1 Hardware

T.B.S.

6.2 Software

T.B.S.

7 Application development

T.B.S.