General Description

The module receives all commands that can also be enclosed in a single message. The module also manages chained up sms till a maximum of 5. To each module the module responds with the message OK or ERROR if the message was not received correctly or if it contains programming errors.

The GSA module can manage:

4 Input opto-isolated named **I1. I2, I3, I4.**

Each input is filtered with a digital filter of 10 milliseconds, then to detect a signal on the inputs is necessary the level remains high for at least this time value.

1 Input for 220VAC power supply monitoring named **Ia**

4 Output relais 10A named **Q1. Q2, Q3, Q4**

2 Analogue Input for temperature sensors (any PTC with KTY81/210 sensor) named **A1, A2**

The module only recognizes the numbers in its own SIM book and must be written in international notation (+39 xxxxx). When you call or send a SMS to the GSA module you need to show your number.

You can store up to 6 numbers in the SIM phonebook.
It is possible to communicate with the GSA module sending commands by SMS and receiving SMS messages according to a predetermined format discussed below.

The module has the ability to execute a program written by the user and stored into the module through SMS messages. The program can enable or disable the outputs in response to events that the module can detect (changes in inputs and outputs and temporal events)

PREPAID SIM

Given the differents methods of transmission for the credit by GSM Operators must indicate to the module how to retrieve the remaining credit.
The module will store the string of credit received from the network and send it to the phone that will be required by a specific command.
First you need to store in the sim card number to call to get the credit (either text or string immediately, not obviously a voicemail!).
This number must be stored under the name CREDIT in SIM phonebook. How to send credit will be defined by a specific command (see below).

Command Execution

Is possible to send multiple commands to GSA SEPARATED BY COMMA (',')

Settings for sending credit
GSM operators can send the string remaining credit in various ways. This command is used to define the mode. (Consult your operator to know how to send credit)

CREDITMODE= mod:"sms" Where mod is the mode and sms is any string to be sent.

Call mode
CREDITMODE = 0
A voice call is made to the number "CREDIT". The operator ends the call and sends an SMS with the string of credit remaining (Typical vodafone dial the number 404)

Mode SMS
CREDITMODE = 1 "sms to send"
"sms to send" text message is sent to number CREDIT. The operator will answer with an SMS showing the remaining credit (Typical of TIM with sms "PRE CRE SIN number 40916)

Mode instant message
CREDITMODE = 2
Sends a service call to the number CREDIT. The operator immediately responds with the string of credit (typical number for those operators that use such calls is ”* number #”, ie starting with * and end with #)

Setting Date and Time
The module takes the time and date directly from the GSM network if the mobile operator has provided the service delivery date and time. Vodafone and TIM, for example, in January 2011 don't have this service available.

TIME=dd/mm/yy hh:mn:ss	hh = hour; mn = minutes; ss = seconds
	dd = day(01-31); mm = month (01-12); yy = year (09..)

Power supply falling down or raise up

Pw=1	Enable sending SMS status when power is absent
Pw=0	Disable sending SMS status when power is absent

When enabled the state is sent to the first number in the phonebook.

Output Set or Reset

Qx=0 x = 1..4

Qx=1

Description allocation input and output

X="string" X is the I/O number shows as below:

1	Analog Input A1
2	Analog Input A2
3	Digital Input I1
4	Digital Input I2
5	Digital Input I3
6	Digital Input I4
7	Digital Output Q1
8	Digital Output Q2
9	Digital Output Q3
10	Digital Output Q4

"string" is the string you want to show in sms status in spite as default string MAXIMUM 10 CHARACTERS (Excess characters will be truncated)

If you intend to describe the I/O strings be careful if you do not want to send multiple messages linked with the SMS state.
The calculation can be characterized as follows:

SMS status length =

(Each I/O length + 7) +

(Each Analog Input length + 9) + 21

example: 2 characters each I/O:

(2 + 7) each I/O = 8*(2+7) = 72

(2 + 9) each analog input = 2*(2+9) = 22

+ 21

Total 72 + 22 + 21 = 115

If exceeded the 160 characters would be sent 2 sms.

Command assignment of an offset to temperature probes.

If you have a thermometer accurate than 1% you can add an offset to temperature sensors in order to make the measure more accurate.

OffsetX= yy.yy	where X =	1 or 2 e and represents the number of probe to which to add an offset
	yy.yy	Represents the value (a negative value simply insert -). The decimal point must always be the point.

These values are written in nonvolatile memory and will be replaced only by sending a new command.

Storing a temporal event

You can store up to 32 times (hours and dates) that may serve as events that involve the actions. For example, you can activate an output after a time.

Txx= dd:mm:yy [hh/mm/ss]	xx = 01-32 (both characters must be inserted)
	hh = hour; mm = minutes; ss = seconds
	dd = day(01-31); mm = month (01-12); yy = year (10..)

If the date is missed the event time does not consider the course date.
The time can not be omitted.

You can store up to 32 events **T01..T32**

Example

T01=21/05/09 15:20:00

T01=15:20:00

Storing an weekly event

You can store up to 32 events per week which can serve as events that involve the actions. For example, you can activate an output after a time.

Wxx= YYY hh:mm:ss	xx = 01-32 (both characters must be inserted)
	hh = hour; mm = minutes; ss = seconds
	YYY = 3 characters showing the day per week:

mon = Monday

tue = Tuesday

wed = Wednesday

thu = Thursday

fri = Friday

sat = Saturday

sun = Sunday

You can store up to 32 events **W01..W32**

Example W01=sun 15:20:00

Storing a single SMS

You can store up to 10 messages (0 .. 9) of maximum 50 characters length that can be sent only through the programming.

SMSx="string to memorize" (x = 0..9)

Example: SMS1="Cold Water Temperature Alarm"

The characters in the string are between two quotes.
The commands sent are stored in nonvolatile memory, so the absence of power does not erase commands even when the battery module is off.

Any ring from a number in phonebook sends the module status message

Reading the program currently running

Sending this command GSA module responds with the programming string currently in use

PRG

Program Execution

Sending this command GSA module runs or stops current program stored in the module

RUN = 0 **stop program**

RUN = 1 **run program**

NOTE:

Since there is no command to read events by the GSA module, you should prepare a single message when you define the events and program to execute. In this way, the request command above will also include events stored.
(Eg at the end of document)

Request Status via SMS
You can request the status of the module by sending even this command:
STATUS
The module will respond by sending the state instead of "ok" to the receipt.

Mode activation GATE
This mode is used to activate an output without using the call and then spend money in order to activate an output.
In this mode, program execution could be compromised. Take care not to use the output running the same program that uses the gate mode (the results would be unpredictable)
Activating this mode the only way to ask the state of the module is through the **STATUS** command via SMS.

When this mode is active the status SMS will show "Progr: GATE" or "Prog: RUN & GATE" if is running even a program.

<u>Command syntax:</u>

<u>Pulse output</u>

RING = Qx: nn
x = output number to activate (1 .. 4)
nn = duration in tenths of a second

Output toggle

RING = Qx
x = output number to activate (1 .. 4)
For each ring, the output is inverted.

Disable mode: RING= OFF

<u>Mode activation: Dual-mode output</u>

This mode works like the previous one but uses two outputs: the first ring an output is activated for a time and the second ring different output is activated for a second time. This is very useful for example to handle hand-pump with remote control start and stop.

TOGGLE = Qx: Qy: nn

x (1 .. 4)	Output to be activated on the first ring
y (1 .. 4)	Output to be activated on the second ring
nn	Pulse in tenths of a second

NOTE:
Activating this mode the only way to ask the state of the module is through the **STATUS** command via SMS.

Programming

A program is a series of events that are monitored and which follows an action from the module.
Actions by module may be accounted for in two easy steps: changing the status of output and /or sending a message.
By default, the module comes with a default program that monitors the input **I1** and the case goes to high logic level sends an SMS with the message "Active alarm input I1".
This string number preconfigured is 0.

WARNING:

A message that contains a program may not contain individual commands and vice versa.

Handled events are:

- Input or Output state change
- Power supply absent or restored
- Achievement of a predetermined time
- Achieve a predetermined date
- Analog Input threshold exceeded or not

Actions can be:

- Output Set o Reset
- SMS sending to a phone number
- SMS sending to the first phonebook number with confirmation otherwise to the next until the end of the numbers are available
- Sending Status of Input Output and GSM modem

Programming Syntax

Start and end programming sequence

&	Start sequence: is the first character to inform that the next strings are a program
!	End sequence

Whatever is between these two commands is interpreted as a module programming

A program step is always composed of two parts separated by commas (','): the first part is the event to check and the second the action to take.
Each program step is separated by a semicolon (';').

EXAMPLE:

```
&
PI1, Q1=1; PI2, Q1=0;            two program steps
!
```

Event is considered active if the clock module is greater than or equal to the date / time of the event.

<u>I/O Status Changing</u>

PIx, PQx	Rising edge Input Ix (1..4) or Output Qx (1..4)
PNIx, PNQx	Falling edge Input Ix (1..4) or Output Qx (1..4)

<u>Achievement of a fixed time event</u>

<u>Txx where xx = 1 .. 16 temporal events (always write both characters)</u>

<u>Achievement of a fixed weekly time event</u>

Wxx where x = 1..16 weekly temporal events (always write both characters)

<u>Threshold of an analog input</u>

A1 > **xxx**	analog input 1	**xxx = -99 .. 999 (celsius degree)**
A1 < xxx	analog input 1	
A2 > xxx	analog input 2	
A2 < xxx	analog input 2	

<u>Analog range exeeded</u>

A1 >< **xxx:yyy**	analog input 1	**xxx, yyy = -99 .. 999 con xxx < yyy**
A2 >< **xxx:yyy**	analog input 2	

This instruction checks whether the analog value out of range target. The separation between the two values is the character ':'

<u>Sending selected SMS number to a phone book</u>

SENDxx	sends sms number xx (0-9) to the first phone book number
SENDCxx	sends sms number xx (0-9) with call confirm

Basics of SENDCxx:

When the module sends an SMS with confirmation call, it takes the first number of SIM phone book and perform a call number. If this number closes the call within 60 seconds the module sends the SMS. If by chance the number is not answered within 60 seconds or is unreachable, the module passes to the next number of phone book and so on until the end of it. If at the end the message has not yet been sent by default GSA module sends the message to the first number in the phone book.

PROGRAMMIMG EXAMPLES

Set up of 5 evens:

T01=20/12/09 00:00:00, T02=07/01/10 12:00:00, T03=16:00:00, T04=18:00:00, T05=19:23:00

(T1 and T2 are date events, while T3 and T5 are only time events.)

```
&
T01, Q1=0;
T02 Q1=1;
 !
```

From Date an Time T1 Q1 will be OFF and from date and time T2 become ON

By factory default GSA module contains the below string:

SMS0 “Alarm Input I1 active”

and the below program:

```
&
PI1,SENDC0;
!
```

Example of Program included events in one SMS:

T01= 10:00:00,T2=19:05:00,W01=thu 10:00:00,W02=thu 19:00:00, &T1,q1=0;T2,q1=1;W1,Q2=0;W2,q2=1;!

Note that it is indifferent to write capital letters and number event with one or two characters

T01 e T02 are two temporal events

W01 e W02 are two weekly events

Program simply put Q1 = OFF from event T1 and Q1 = ON from event T2.

The same for Q2 with weekly events W01 e W02