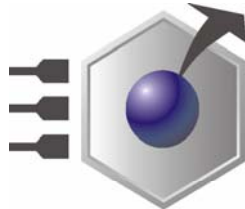


***EmbeddedDNA***<sup>®</sup>



***An0061***

**ALPHABOX: The Guardian**

Rev.1.0

Dec. 2005

COPYRIGHT 1994-2005 Eurotech S.p.A. All Rights Reserved.



Via J. Linussio 1  
33020 AMARO (UD)  
ITALY  
Phone: +39 0433 485 411  
Fax: +39 0433 485 499

Website: [www.eurotech.it](http://www.eurotech.it)  
E-mail: [sales@eurotech.it](mailto:sales@eurotech.it)

#### NOTICE



**Although all the information contained herein has been carefully verified, Eurotech S.p.A. assumes no responsibility for errors that might appear in this document, or for damage to property or persons resulting from an improper use of this manual and of the related software. Eurotech S.p.A. reserves the right to change the contents and form of this document, as well as the features and specifications of its products at any time, without notice.**

Trademarks and registered trademarks appearing in this document are the property of their respective owners

---

## Conventions

The following table lists conventions used throughout this guide.

Icon	Notice Type	Description
	<b>Information note</b>	Important features or instructions
	<b>Warning</b>	Information to alert you to potential damage to a program, system or device or potential personal injury

LSB: Least Significant Byte

MSB: Most Significant Byte

R: Read

W: Write

WO: Write Only

(This page is intentionally left blank.)

# Contents

---

Conventions .....	3
<b>Contents .....</b>	<b>5</b>
<b>Chapter 1 Monitoring features: the Guardian .....</b>	<b>7</b>
Foreword about the Guardian monitoring system feature .....	8
The Guardian Registers .....	8
Accessing the registers .....	10
Register descriptions .....	11
<i>REV_L &amp; REV_H</i> .....	11
<i>COMMAND</i> .....	12
RESTART .....	12
SAVE_DEFAULT .....	12
RUNTIME_INVALIDATE .....	12
ERASE_DEFAULTS .....	12
SHUTDOWN .....	12
ENABLE_PWM .....	13
DISABLE_PWM .....	13
EXCLUDE_KEY .....	13
IRQ_ACK .....	13
POST_ERROR .....	13
CPU_TEMP_ERROR .....	13
WRITE_TEMP_OFFSET .....	13
READ_FLASH .....	13
READ_EEPROM .....	13
WRITE_EEPROM .....	13
SCRATCH .....	14
STATUS .....	14
SHUTDOWN_DELAY .....	14
Counter Registers .....	15
CNT_TRIG_REG .....	15
CNT_REG_H & CNT_REG_L .....	15
PWM Registers .....	16

PWM_TB_REG .....	16
PWM_DUTY_REG_H &, PWM_DUTY_REG_L .....	16
<i>VDD Registers</i> .....	17
VDD_MIN_LOW_REG & VDD_MIN_HIGH_REG .....	17
VDD_MAX_LOW_REG & VDD_MAX_HIGH_REG .....	17
VDD_HIGH_REG .....	18
VDD_LOW_REG .....	18
<i>VIN Registers</i> .....	19
VIN_MIN_LOW_REG, VIN_MIN_HIGH_REG .....	19
VIN_MAX_LOW_REG, VIN_MAX_HIGH_REG .....	19
VIN_LOW_REG .....	19
VIN_HIGH_REG .....	20
<i>VCC3 Registers</i> .....	21
VCC3_MIN_LOW_REG & VCC3_MIN_HIGH_REG .....	21
VCC3_MAX_LOW_REG & VCC3_MAX_HIGH_REG .....	21
VCC3_ALW_MIN_LOW_REG, VCC3_ALW_MIN_HIGH_REG .....	22
VCC3_ALW_MAX_LOW_REG, VCC3_ALW_MAX_HIGH_REG .....	22
VCC3_LOW_REG .....	22
VCC3_HIGH_REG .....	23
VCC3_ALWAYS_LOW_REG .....	23
VCC3_ALWAYS_HIGH_REG .....	23
<i>V12 Registers</i> .....	24
V12_MIN_LOW_REG & V12_MIN_HIGH_REG .....	24
V12_MAX_LOW_REG & V12_MAX_HIGH_REG .....	24
V12_LOW_REG .....	24
V12_HIGH_REG .....	24
<i>Temperature Registers</i> .....	26
MICRO_TEMP_MIN_LOW_REG & MICRO_TEMP_MIN_HIGH_REG .....	26
MICRO_TEMP_MAX_LOW_REG & MICRO_TEMP_MAX_HIGH_REG .....	26
MICRO_TEMP_LOW_REG .....	27
MICRO_TEMP_HIGH_REG .....	27
OUT_OF_RANGE_STATUS_REG .....	27
<i>Flash Registers</i> .....	28
FLASH_ADDR_H_REG .....	28
FLASH_ADDR_L_REG .....	28
FLASH_DATA_REG .....	28
<i>EEPROM Registers</i> .....	29
EEPROM_ADDR_REG .....	29
EEPROM_DATA_REG .....	29
EEPROM_STATUS_REG .....	29
Storage of the events: the Log .....	30
<i>Map of the <math>\mu</math>C flash memory</i> .....	30
<i>Map of the EEPROM</i> .....	30
<i>Alarm log Structure</i> .....	30
<i>How to read the Log</i> .....	31
Power supply CPU logic.....	31

---

## Chapter 1 Monitoring features: the Guardian

---



The Alphabox system comes with a new integrated intelligence called "Guardian", this allows the user to monitor the systems operating parameters and manage the command execution logic even if the system is in standby mode.

The "Guardian", allows for the following:

- Verification of voltages and internal temperature; signalling alarms to the CPU
- Storage of the system status LOG on the EEPROM
- Manages the power-up and power-down of the system

---

## Foreword about the Guardian monitoring system feature

The Guardian is based on a micro-controller ( $\mu$ C) C8051F310 (8051 compatible) featuring:

- Counter input
- PWM signal generator
- Countdown monitor
- Voltages enabling logic
- Command execution logic
- Internal voltage monitor
- Internal temperature monitor

Every second the Guardian checks that the internal voltage and temperature values are within the permitted ranges, if these conditions are not realized the event is stored in an EEPROM and an interrupt request can be activated. (The description is valid for: Guardian: 1.7, Bootloader: 1.3 or higher).

---

## The Guardian Registers

The Guardian is managed with status and control registers implemented within it. These registers are 1 byte long as follows:

Register Name	Description	Address
REV_L	Firmware revision, LSB	0x00
REV_H	Firmware revision, MSB	0x01
COMMAND_REG	Register command	0x02
SCRATCH_REG	Register scratch	0x03
CNT_TRIG_REG	Counter trigger control	0x04
CNT_REG_H	Counter, MSB	0x05
CNT_REG_L	Counter, LSB	0x06
PWM_TB_REG	PWM time base control	0x07
PWM_DUTY_REG_H	PWM duty cycle control, MSB	0x08
PWM_DUTY_REG_L	PWM duty cycle control, LSB	0x09
SHUTDOWN_DELAY	Retarded shutdown control	0x0A
STATUS	$\mu$ C status register	0x0B
VDD_MIN_LOW_REG	VDD minimum limit, LSB	0x0C
VDD_MIN_HIGH_REG	VDD minimum limit, MSB	0x0D
VDD_MAX_LOW_REG	VDD maximum limit, LSB	0x0E
VDD_MAX_HIGH_REG	VDD maximum limit, MSB	0x0F
VIN_MIN_LOW_REG	VIN minimum limit, LSB	0x10
VIN_MIN_HIGH_REG	VIN minimum limit, MSB	0x11

Continued over...

Register Name	Description	Address
VIN_MAX_LOW_REG	VIN maximum limit, LSB	0x12
VIN_MAX_HIGH_REG	VIN maximum limit, MSB	0x13
VCC3_MIN_LOW_REG	VCC3 minimum limit, LSB	0x14
VCC3_MIN_HIGH_REG	VCC3 minimum limit, MSB	0x15
VCC3_MAX_LOW_REG	VCC3 maximum limit, LSB	0x16
VCC3_MAX_HIGH_REG	VCC3 maximum limit, MSB	0x17
V12_MIN_LOW_REG	12V minimum limit, LSB	0x18
V12_MIN_HIGH_REG	12V minimum limit, MSB	0x19
V12_MAX_LOW_REG	12V maximum limit, LSB	0x1A
V12_MAX_HIGH_REG	12V maximum limit, MSB	0x1B
MICRO_TEMP_MIN_LOW_REG	Micro-controller temperature minimum limit, LSB.	0x1C
MICRO_TEMP_MIN_HIGH_REG	Micro-controller temperature minimum limit, MSB.	0x1D
MICRO_TEMP_MAX_LOW_REG	Micro-controller temperature maximum limit, LSB.	0x1E
MICRO_TEMP_MAX_HIGH_REG	Micro-controller temperature maximum limit, MSB.	0x1F
VCC3_ALW_MIN_LOW_REG	VCC_Always minimum limit, LSB	0x20
VCC3_ALW_MIN_HIGH_REG	VCC_Always minimum limit, MSB	0x21
VCC3_ALW_MAX_LOW_REG	VCC_Always maximum limit, LSB	0x22
VCC3_ALW_MAX_HIGH_REG	VCC_Always maximum limit, MSB	0x23
OUT_OF_RANGE_STATUS_REG	Status of the monitored values	0x24
MICRO_TEMP_LOW_REG	Micro-controller temperature value, LSB.	0x25
MICRO_TEMP_HIGH_REG	Micro-controller temperature value, MSB.	0x26
FLASH_ADDR_H_REG	Flash reading address, LSB	0x27
FLASH_ADDR_L_REG	Flash reading address, MSB	0x28
FLASH_DATA_REG	Flash reading data	0x29
VDD_LOW_REG	VDD value, LSB	0x2A
VDD_HIGH_REG	VDD value, MSB	0x2B
VIN_LOW_REG	VIN value, LSB	0x2C
VIN_HIGH_REG	VIN value, MSB	0x2D
VCC3_LOW_REG	VCC3 value, LSB	0x2E
VCC3_HIGH_REG	VCC3 value, MSB	0x2F
V12_LOW_REG	V12 value, LSB	0x30
V12_HIGH_REG	V12 value, MSB	0x31
VCC3_ALW_LOW_REG	VCC3_ALWAYS value, LSB	0x32
VCC3_ALW_HIGH_REG	VCC3_ALWAYS value, MSB	0x33
EEPROM_ADDR_REG	EEPROM address	0x34
EEPROM_DATA_REG	EEPROM data	0x35
EEPROM_STATUS_REG	EEPROM status	0x36

## Accessing the registers

Accessing the registers is done through reading and writing operations in the I/O address space of the ISA BUS. The 8-bit I/O port (mapped at the default address of 0x324h) is available for accessing the registers.

For transmitting a byte to the  $\mu\text{C}$ , write it in the GUARDIAN\_INTERFACE register. At the same time as this transmission the  $\mu\text{C}$  will transmit a response byte to the CPU in the GUARDIAN\_INTERFACE register. A read operation in this register will allow you to get the received byte.

Read and write operations follow a protocol that controls the transmission of byte sequences. Each sequence must be separated by a time interval of >10ms.

In more detail, the read operation of a register is made following this transmission / reception sequence:

	1st Byte	2nd Byte	3rd Byte	4th Byte
<b>TX (to the <math>\mu\text{C}</math>)</b>	'R' (0x52)	<i>Addr</i>	<i>Dummy</i>	<i>Dummy</i>
<b>RX (from the <math>\mu\text{C}</math>)</b>	<i>Dummy</i>	'R' (0x52)	<i>Addr</i>	<i>Data</i>

The *Addr* and *Data* bytes are respectively the address of the register to be read and the read data. The  $\mu\text{C}$  performs an echo with the first 2 bytes received; in this way CPU can validate the transmitted data.

The write operation of a register is made following this transmission / reception sequence:

	1 <sup>st</sup> Byte	2 <sup>nd</sup> Byte	3 <sup>rd</sup> Byte	4 <sup>th</sup> Byte
<b>TX (to the <math>\mu\text{C}</math>)</b>	'W' (0x57)	<i>Addr</i>	<i>Dummy</i>	<i>Dummy</i>
<b>RX (from the <math>\mu\text{C}</math>)</b>	<i>Dummy</i>	'W' (0x57)	<i>Addr</i>	<i>Data</i>

The *Addr* and *Data* bytes are respectively the address of the register to be written and the data to write. The  $\mu\text{C}$  performs an echo with the first 3 bytes received; in this way CPU can validate the transmitted data.

In the case that a sequence starts badly (i.e. using bytes different from 'R' or 'W') the  $\mu\text{C}$  replies with the byte 'N' (0x4E) at the next transmission and then it waits for a new sequence.

Registers can be accessed using **Read**, **Write** or **Write-Only** operations.

Some registers are set with a default value when the firmware is started (and the Guardian  $\mu\text{C}$  is powered on). This default value can be:

- A value set by Eurotech (described in more detail in the following paragraphs) if there are not any default values to load in the registers stored in the non-volatile memory of the  $\mu\text{C}$
- A value previously stored by the application software in the non-volatile memory of the  $\mu\text{C}$  (Refer to the SAVE\_DEFAULT command)

---

## Register descriptions

### REV\_L & REV\_H

The REV\_L and REV\_H registers contain the  $\mu$ C firmware revision level:

“H” is the high byte contained in REV\_H

“L” is the low byte contained in REV\_L

REV_L									0x00
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	REV_L7 REV_L0								
Default	Depends on the revision								
Access	R								
Allowable values: 0 to 254 (255 reserved)									

REV_H									0x01
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	REV_H7	REV_H6 REV_H0							
Default	0	Depends on the revision							
Access	R	R							
Allowable values: 0 to 127									

Bit 7 of REV\_H is 1 during the Bootloader program execution.

This allows the CPU to verify which program (Bootloader or runtime) is running by reading bit 7.

## COMMAND

The Guardian can execute the commands sent from the CPU.

A command is given by writing its corresponding code in the COMMAND register:

	COMMAND							0x02
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name	COMMAND7 to COMMAND0							
Default	0							
Access	R/W							

When the command execution ends, the register assumes the value: NO\_COMMAND.

The CMD\_DONE register signals to the CPU that the previous execution has been completed.

The CMD\_DONE register is activated when a command to execute is loaded in the COMMAND register. It is deactivated when the command has been executed.

Table 1. Command set

Name	Code	Description
NO_COMMAND	0x00	-
RESTART	0x01	Guardian firmware re-start
SAVE_DEFAULT	0x02	Save register default values in the non volatile memory
RUNTIME_INVALIDATE	0x03	Runtime firmware Invalidation
ERASE_DEFAULTS	0x04	Erase the user default values
SHUTDOWN	0x05	Starts the shutdown procedure
ENABLE_PWM	0x06	Enable the PWM out
DISABLE_PWM	0x07	Disable the PWM out
EXCLUDE_KEY	0x08	Disable the Key signal control
IRQ_ACK	0x09	Acknowledge IRQ
POST_ERROR	0x0A	Reserved
CPU_TEMP_ERROR	0x0B	Save the CPU over-temperature error in the memory
WRITE_TEMP_OFFSET	0x0C	Write the temperature offset in the flash
READ_FLASH	0x0D	Read the logs from the $\mu$ C Flash
READ_EEPROM	0x0E	Read the EEPROM
WRITE_EEPROM	0x0F	Write the EEPROM

### RESTART

Activates a software reset of the  $\mu$ C that will restart the firmware running in the Guardian.

### SAVE\_DEFAULT

Saves the values of all the write accessible Guardian registers into the non-volatile memory of  $\mu$ C. Stored values will then be reloaded during the next start up.

### RUNTIME\_INVALIDATE

Erases the runtime firmware validity key.

This command is used during the  $\mu$ C firmware update procedure

### ERASE\_DEFAULTS

Erases the default register values previously stored into the flash memory of the  $\mu$ C.

At the next start up the registers will load the default values.

### SHUTDOWN

Powers down the CPU module after a number of seconds as defined in the SHUTDOWN\_DELAY register.

**ENABLE\_PWM**

Enables the PWM output with the parameters contained in the registers PWM\_TB\_REG, PWM\_DUTY\_REG\_H and PWM\_DUTY\_REG\_L.

**DISABLE\_PWM**

Disables the PWM output.

In order to change the PWM run-time output parameters it is necessary to send a DISABLE\_PWM command, modify the registers PWM\_TB\_REG PWM\_DUTY\_REG\_H and PWM\_DUTY\_REG\_L, and send an ENABLE\_PWM command.

**EXCLUDE\_KEY**

Disables the KEY signal status control

**IRQ\_ACK**

Disable the interrupt line

**POST\_ERROR**

Reserved

**CPU\_TEMP\_ERROR**

Reserved.

**WRITE\_TEMP\_OFFSET**

Save the temperature offset of the  $\mu$ C in the flash

**READ\_FLASH**

Read a byte from the flash of the  $\mu$ C.

The reading address must be programmed in the FLASH\_ADDR\_L\_REG and FLASH\_ADDR\_H\_REG registers before the command execution.

The read data is stored in the READ\_FLASH register.

**READ\_EEPROM**

Read a byte from the EEPROM connected to the  $\mu$ C.

The reading address must be programmed in the EEPROM\_ADDR\_REG register before the command execution.

The read data is stored in the EEPROM\_DATA\_REG register.

**WRITE\_EEPROM**

Write a byte in the EEPROM connected to the  $\mu$ C.

The writing address and the data to write must be programmed respectively in the EEPROM\_ADDR\_REG and EEPROM\_DATA\_REG registers before the command execution.

## SCRATCH

SCRATCH									0x03
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	SCRATCH7 to SCRATCH0								
Default	X								
Access	R/W								

This scratch read / write accessible register has no effect on the power monitor. It has been developed to be used by the user as a service / debugging register.

## STATUS

STATUS									0x0B
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	IRQ5_EN	IO_SEL1	IO_SEL0	KEY_EX	SHUTD	ODOV	PWM	WDT	
Default	0x00								
Access	R/W								

STATUS register defines the current  $\mu$ C status:

Status	Definition
WDT	CPU has been reset from the watchdog timer
PWM	PWM out is active
OFLOW	Counter is in overflow
SHUTD	Shutdown procedure is running
KEY_EX	Key input is disabled
IO_SEL0	Decode logic base register address selection
IO_SEL1	Decode logic base register address selection
IRQ5_EN	IRQ5 enabling

PWM bit can be used to enable/disable PWM output at the next  $\mu$ C starting.

IO\_SEL0 and IO\_SEL1 bits can be used to modify the I/O space addresses of the Alphabox internal register.

IO_SEL0	IO_SEL1	Address
0	0	0x320..0x327 (default)
0	1	0x310..0x317
1	0	0x330..0x337
1	1	0x340..0x347

IRQ5\_EN bit can be used to enable the IRQ5 in the CPU.

## SHUTDOWN\_DELAY

SHUTDOWN_DELAY									0x0A
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	SHUTDOWN_DELAY 7 to SHUTDOWN_DELAY0								
Default	0x05								
Access	R/W								

SHUTDOWN\_DELAY register defines how many seconds the  $\mu$ C has to wait between the receipt of the SHUTDOWN command and the CPU shutdown.

## Counter Registers

### CNT\_TRIG\_REG

CNT_TRIG_REG								0x04
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name	CNT_TRIG_REG7 ÷ CNT_TRIG_REG0							
Default	0x20							
Access	R/W							

The CNT\_TRIG\_REG register allows the selection of the counter trigger parameters as follows:

- 0x10: trigger on transition 1→0
- 0x20: trigger on transition 0→1
- 0x30: trigger on both transitions

### CNT\_REG\_H & CNT\_REG\_L

CNT\_REG\_H and CNT\_REG\_L contain respectively the counters MSB and LSB

CNT_REG_H								0x05
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name	CNT_REG_H7 to CNT_REG_H 0							
Default	0							
Access	R/W							

CNT_REG_L								0x06
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name	CNT_REG_L7 to CNT_REG_L0							
Default	0							
Access	R/W							



Before setting the Counter select which input will be used. In order to read/set the correct value, read/write before the CNT\_REG\_L and then the CNT\_REG\_H registers.

## PWM Registers

### PWM\_TB\_REG

PWM\_TB\_REG register allows you to select the PWM frequency time base.

PWM_TB_REG								0x07
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name	PWM_TB_REG7 to PWM_TB_REG0							
Default	0x08							
Access	R/W							

Allowable values are:

- 0x00: 31.79 Hz
- 0x01: 95.37 Hz

### PWM\_DUTY\_REG\_H &, PWM\_DUTY\_REG\_L

PWM_DUTY_REG_H								0x08
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name	PWM_DUTY_REG_H7 to PWM_DUTY_REG_H0							
Default	0x80							
Access	R/W							

PWM_DUTY_REG_L								0x09
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name	PWM_DUTY_REG_L7 to PWM_DUTY_REG_L0							
Default	0x00							
Access	R/W							

PWM\_DUTY\_REG\_H and PWM\_DUTY\_REG\_L registers define the PWM duty cycle according to the following equation:

$$\text{Duty cycle} = (65536 - (\text{PWM\_DUTY\_REG\_L} + \text{PWM\_DUTY\_REG\_H} * 256)) / 65536$$

The maximum allowed duty cycle (100 %) is obtained with:  
 PWM\_DUTY\_REG\_L = PWM\_DUTY\_REG\_H = 0

The minimum allowed duty cycle (0.0015%) is obtained with:  
 PWM\_DUTY\_REG\_L = PWM\_DUTY\_REG\_H = 0xFF.

Default values set Duty cycle = 50%.

## VDD Registers

### VDD\_MIN\_LOW\_REG & VDD\_MIN\_HIGH\_REG

VDD_MIN_LOW_REG									0x0C
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	VDD_MIN_LOW_REG7 to VDD_MIN_LOW_REG0								
Default	0xE0								
Access	R/W								

VDD_MIN_HIGH_REG									0x0D
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	VDD_MIN_HIGH_REG7 to VDD_MIN_HIGH_REG0								
Default	0x02								
Access	R/W								

VDD\_MIN\_LOW\_REG and VDD\_MIN\_HIGH\_REG registers contain respectively the MSB and LSB of the minimum allowed threshold on VDD for alarm condition.

By reading the VDD\_MIN\_LOW\_REG and VDD\_MIN\_HIGH\_REG registers it is possible to monitor the internal minimum threshold VDD voltage calculating it using the following formula:

$$VDD = \text{'Read value in decimal'} * ((V_{ref} = 3.3V) / 1023) * 2 = \text{'Read value'} * 0.0064516$$

For example, reading 1AA (in hexadecimal that is 426 in decimal) the VDD will be:  
 $426 * (3.3 / 1023) * 2 = 2.75 \text{ V}$

### VDD\_MAX\_LOW\_REG & VDD\_MAX\_HIGH\_REG

VDD_MAX_LOW_REG									0x0E
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	VDD_MAX_LOW_REG7 to VDD_MAX_LOW_REG0								
Default	0x2D								
Access	R/W								

VDD_MAX_HIGH_REG									0x0F
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	VDD_MAX_HIGH_REG7 to VDD_MAX_HIGH_REG0								
Default	0x03								
Access	R/W								

VDD\_MAX\_LOW\_REG and VDD\_MAX\_HIGH\_REG registers contain respectively the MSB and LSB of the maximum allowed threshold on VDD for alarm condition.

By reading the VDD\_MAX\_LOW\_REG and VDD\_MAX\_HIGH\_REG registers it is possible to monitor the internal maximum threshold VDD voltage calculating it using the following formula:

$$VDD = \text{'Read value in decimal'} * ((V_{ref} = 3.3V) / 1023) * 2 = \text{'Read value'} * 0.0064516$$

**VDD\_HIGH\_REG**

	VDD_HIGH_REG							0x2B
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name								
Default								-
Access								R

Reading the VDD\_LOW\_REG and VDD\_HIGH\_REG registers it is possible to monitor the VDD Internal voltage calculating it using the following formula:

$$\text{VDD} = \text{'Read value in decimal'} * ((V_{\text{ref}} = 3.3\text{V}) / 1023) * 2 = \text{'Read value'} * 0.0064516$$

**VDD\_LOW\_REG**

	VDD_LOW_REG							0x2A
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name								
Default								-
Access								R

## VIN Registers

### VIN\_MIN\_LOW\_REG, VIN\_MIN\_HIGH\_REG

VIN_MIN_LOW_REG									0x10
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	VIN_MIN_LOW_REG7 to VIN_MIN_LOW_REG0								
Default	0xC8								
Access	R/W								

VIN_MIN_HIGH_REG									0x11
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	VIN_MIN_HIGH_REG7 to VIN_MIN_HIGH_REG0								
Default	0x00								
Access	R/W								

VIN\_MIN\_LOW\_REG and VIN\_MIN\_HIGH\_REG registers contain respectively the MSB and LSB of the minimum allowed threshold on VIN for alarm condition.

By reading the VIN\_MIN\_LOW\_REG and VIN\_MIN\_HIGH\_REG registers it is possible to monitor the internal minimum threshold VIN voltage calculating it using the following formula:

$$VIN = \text{'Read value in decimal'} * ((V_{ref} = 3.3V) / 1023) * 13 = \text{'Read value'} * 0.041935$$

### VIN\_MAX\_LOW\_REG, VIN\_MAX\_HIGH\_REG

VIN_MAX_LOW_REG									0x12
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	VIN_MAX_LOW_REG7 to VIN_MAX_LOW_REG0								
Default	0x5A								
Access	R/W								

VIN_MAX_HIGH_REG									0x13
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	VIN_MAX_HIGH_REG7 to VIN_MAX_HIGH_REG0								
Default	0x03								
Access	R/W								

VIN\_MAX\_LOW\_REG and VIN\_MAX\_HIGH\_REG registers contain respectively the MSB and LSB of the maximum allowed threshold on VIN for alarm condition.

By reading the VIN\_MAX\_LOW\_REG and VIN\_MAX\_HIGH\_REG registers it is possible to monitor the internal maximum threshold VIN voltage calculating it using the following formula:

$$VIN = \text{'Read value in decimal'} * ((V_{ref} = 3.3V) / 1023) * 13 = \text{'Read value'} * 0.041935$$

### VIN\_LOW\_REG

VIN_LOW_REG									0x2C
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name									
Default	-								
Access	R								

**VIN\_HIGH\_REG**

	VIN_HIGH_REG								0x2D
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name									
Default	-								
Access	R								

By reading the VIN\_LOW\_REG and VIN\_HIGH\_REG registers it is possible to monitor the internal VIN voltage calculating it using the following formula:

$$\text{VIN} = \text{'Read value in decimal'} * ((V_{\text{ref}} = 3.3\text{V}) / 1023) * 13 = \text{'Read value'} * 0.041935$$

## VCC3 Registers

### VCC3\_MIN\_LOW\_REG & VCC3\_MIN\_HIGH\_REG

VCC3_MIN_LOW_REG									0x14
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	VCC3_MIN_LOW_REG7 to VCC3_MIN_LOW_REG0								
Default	0xD1								
Access	R/W								

VCC3_MIN_HIGH_REG									0x15
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	VCC3_MIN_HIGH_REG7 to VCC3_MIN_HIGH_REG0								
Default	0x01								
Access	R/W								

VCC3\_MIN\_LOW\_REG and VCC3\_MIN\_HIGH\_REG registers contain respectively the MSB and LSB of the minimum allowed threshold on VCC3 for alarm condition.

By reading the VCC3\_MIN\_LOW\_REG and VCC3\_MIN\_HIGH\_REG registers it is possible to monitor the internal minimum threshold VCC3 voltage calculating it using the following formula:

$$VCC3 = \text{'Read value in decimal'} * ((V_{ref} = 3.3V) / 1023) * 2 = \text{'Read value'} * 0.0064516$$

### VCC3\_MAX\_LOW\_REG & VCC3\_MAX\_HIGH\_REG

VCC3_MAX_LOW_REG									0x16
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	VCC3_MAX_LOW_REG7 to VCC3_MAX_LOW_REG0								
Default	0x2E								
Access	R/W								

VCC3_MAX_HIGH_REG									0x17
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	VCC3_MAX_HIGH_REG7 to VCC3_MAX_HIGH_REG0								
Default	0x02								
Access	R/W								

VCC3\_MAX\_LOW\_REG and VCC3\_MAX\_HIGH\_REG registers contain respectively the MSB and LSB of the maximum allowed threshold on VCC3 for alarm condition.

By reading the VCC3\_MAX\_LOW\_REG and VCC3\_MAX\_HIGH\_REG registers it is possible to monitor the internal maximum threshold VDD voltage calculating it using the following formula:

$$VCC3 = \text{'Read value in decimal'} * ((V_{ref} = 3.3V) / 1023) * 2 = \text{'Read value'} * 0.0064516$$

**VCC3\_ALW\_MIN\_LOW\_REG, VCC3\_ALW\_MIN\_HIGH\_REG**

VCC3_ALW_MIN_LOW_REG								0x20
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name	VCC3_ALW_MIN_LOW_REG7 to VCC3_ALW_MIN_LOW_REG0							
Default	0xD1							
Access	R/W							

VCC3_ALW_MIN_HIGH_REG								0x21
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name	VCC3_ALW_MIN_HIGH_REG7 to VCC3_ALW_MIN_HIGH_REG0							
Default	0x01							
Access	R/W							

VCC3\_ALW\_MIN\_LOW\_REG and VCC3\_ALW\_MIN\_HIGH\_REG registers contain respectively the MSB and LSB of the minimum allowed threshold on VCC3 Always for alarm condition.

By reading the VCC3\_ALW\_MIN\_LOW\_REG and VCC3\_ALW\_MIN\_HIGH\_REG registers it is possible to monitor the internal minimum threshold VCC3 Always voltage calculating it using the following formula:

$$VCC3 \text{ Always} = \text{'Read value in decimal'} * ((V_{ref} = 3.3V) / 1023) * 2 = \text{'Read value'} * 0.0064516$$

**VCC3\_ALW\_MAX\_LOW\_REG, VCC3\_ALW\_MAX\_HIGH\_REG**

VCC3_ALW_MAX_LOW_REG								0x22
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name	VCC3_MAX_LOW_REG7 to VCC3_MAX_LOW_REG0							
Default	0x2E							
Access	R/W							

VCC3_ALW_MAX_HIGH_REG								0x23
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name	VCC3_ALW_MAX_HIGH_REG7 to VCC3_ALW_MAX_HIGH_REG0							
Default	0x02							
Access	R/W							

VCC3\_ALW\_MAX\_LOW\_REG and VCC3\_ALW\_MAX\_HIGH\_REG registers contain respectively the MSB and LSB of the maximum allowed threshold on VCC3 Always for alarm condition.

By reading the VCC3\_ALW\_MAX\_LOW\_REG and VCC3\_ALW\_MAX\_HIGH\_REG registers it is possible to monitor the internal maximum threshold VCC3 Always voltage calculating it using the following formula:

$$VCC3 \text{ Always} = \text{'Read value in decimal'} * ((V_{ref} = 3.3V) / 1023) * 2 = \text{'Read value'} * 0.0064516$$

**VCC3\_LOW\_REG**

VCC3_LOW_REG								0x2E
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name								
Default	-							
Access	R							

**VCC3\_HIGH\_REG**

	VCC3_HIGH_REG								0x2F
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name									
Default	-								
Access	R								

By reading the VCC3\_LOW\_REG and VCC3\_HIGH\_REG registers it is possible to monitor the internal VCC3 voltage calculating it using the following formula:

$$\text{VCC3} = \text{'Read value in decimal'} * ((V_{\text{ref}} = 3.3\text{V}) / 1023) * 2 = \text{'Read value'} * 0.0064516$$

**VCC3\_ALWAYS\_LOW\_REG**

	VCC3_ALWAYS_LOW_REG								0x32
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name									
Default	-								
Access	R								

**VCC3\_ALWAYS\_HIGH\_REG**

	VCC3_ALWAYS_HIGH_REG								0x33
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name									
Default	-								
Access	R								

By reading the VCC3\_ALWAYS\_LOW\_REG and VCC3\_ALWAYS\_HIGH\_REG registers it is possible to monitor the internal VCC3\_ALWAYS voltage, calculating it with the following formula:

$$\text{VCC3\_ALWAYS} = \text{'Read value in decimal'} * ((V_{\text{ref}} = 3.3\text{V}) / 1023) * 2 = \text{'Read value'} * 0.0064516$$

## V12 Registers

### V12\_MIN\_LOW\_REG & V12\_MIN\_HIGH\_REG

V12_MIN_LOW_REG									0x18
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	V12_MIN_LOW_REG7 to V12_MIN_LOW_REG0								
Default	0x00								
Access	R/W								

V12_MIN_HIGH_REG									0x19
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	V12_MIN_HIGH_REG7 to V12_MIN_HIGH_REG0								
Default	0x00								
Access	R/W								

V12\_MIN\_LOW\_REG and V12\_MIN\_HIGH\_REG registers contain respectively the MSB and LSB of the minimum allowed threshold on V12 for alarm condition.

By reading the V12\_MIN\_LOW\_REG and V12\_MIN\_HIGH\_REG registers it is possible to monitor the internal minimum threshold V12 voltage calculating it using the following formula:

$$V12 = \text{'Read value in decimal'} * ((V_{ref} = 3.3V) / 1023) * 4.32 = \text{'Read value'} * 0.013954$$

### V12\_MAX\_LOW\_REG & V12\_MAX\_HIGH\_REG

V12_MAX_LOW_REG									0x1A
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	V12_MAX_LOW_REG7 to V12_MAX_LOW_REG0								
Default	0x88								
Access	R/W								

V12_MAX_HIGH_REG									0x1B
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	V12_MAX_HIGH_REG7 to V12_MAX_HIGH_REG0								
Default	0x03								
Access	R/W								

V12\_MAX\_LOW\_REG and V12\_MAX\_HIGH\_REG registers contain respectively the MSB and LSB of the maximum allowed threshold on V12 for alarm condition.

By reading the V12\_MAX\_LOW\_REG and V12\_MAX\_HIGH\_REG registers it is possible to monitor the internal maximum threshold V12 voltage calculating it using the following formula:

$$V12 = \text{'Read value in decimal'} * ((V_{ref} = 3.3V) / 1023) * 4.32 = \text{'Read value'} * 0.013954$$

### V12\_LOW\_REG

V12_LOW_REG									0x30
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name									
Default	-								
Access	R								

### V12\_HIGH\_REG

	V12_HIGH_REG							0x31
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name								
Default	-							
Access	R							

By reading the V12\_LOW\_REG and V12\_HIGH\_REG registers it is possible to monitor the internal +12V voltage. This option is available if the system is has a DC/DC converter (+5V to +12V) present. The following formula has to be used:

$$V12 = \text{'Read value in decimal'} * ((V_{ref} = 3.3V) / 1023) * 4.32 = \text{'Read value'} * 0.013954$$

## Temperature Registers

### MICRO\_TEMP\_MIN\_LOW\_REG & MICRO\_TEMP\_MIN\_HIGH\_REG

MICRO_TEMP_MIN_LOW_REG									0x1C
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	MICRO_TEMP_MIN_LOW_REG7 to MICRO_TEMP_MIN_LOW_REG0								
Default	0x00								
Access	R/W								

MICRO_TEMP_MIN_HIGH_REG									0x1D
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	MICRO_TEMP_MIN_HIGH_REG7 to MICRO_TEMP_MIN_HIGH_REG0								
Default	0x00								
Access	R/W								

MICRO\_TEMP\_MIN\_LOW\_REG and MICRO\_TEMP\_MIN\_HIGH\_REG registers contain respectively the MSB and LSB of the minimum allowed threshold on micro-controller temperature value for alarm condition.

By reading the MICRO\_TEMP\_MIN\_LOW\_REG and MICRO\_TEMP\_MIN\_HIGH\_REG registers it is possible to calculate the internal minimum allowed threshold on micro-controller temperature value using this formula:

$$\text{'}\mu\text{C temperature in }^{\circ}\text{C'} = (\text{'Read value in decimal'} * (3300/1023) - 897) / 3.35$$

For example, reading 11F (in hexadecimal, that is 287 in decimal) the  $\mu\text{C}$  temperature will be:  
 $(287 * (3300 / 1023) - 897) / 3,35 = 8.6 \text{ }^{\circ}\text{C}$

### MICRO\_TEMP\_MAX\_LOW\_REG & MICRO\_TEMP\_MAX\_HIGH\_REG

MICRO_TEMP_MAX_LOW_REG									0x1E
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	MICRO_TEMP_MAX_LOW_REG7 to MICRO_TEMP_MAX_LOW_REG0								
Default	0xFF								
Access	R/W								

MICRO_TEMP_MAX_HIGH_REG									0x1F
Bit	B7	B6	B5	B4	B3	B2	B1	B0	
Bit name	MICRO_TEMP_MAX_HIGH_REG7 to MICRO_TEMP_MAX_HIGH_REG0								
Default	0xFF								
Access	R/W								

MICRO\_TEMP\_MAX\_LOW\_REG and MICRO\_TEMP\_MAX\_HIGH\_REG registers contain respectively the MSB and LSB of the maximum allowed threshold on micro-controller temperature value for alarm condition.

By reading the MICRO\_TEMP\_MIN\_LOW\_REG and MICRO\_TEMP\_MIN\_HIGH\_REG registers it is possible to calculate the internal minimum allowed threshold on temperature micro-controller temperature value using this formula:

$$\text{'}\mu\text{C temperature in }^{\circ}\text{C'} = (\text{'Read value in decimal'} * (3300/1023) - 897) / 3.35$$

**MICRO\_TEMP\_LOW\_REG**

MICRO_TEMP_LOW_REG								0x25
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name								
Default	0x00							
Access	R/W							

**MICRO\_TEMP\_HIGH\_REG**

MICRO_TEMP_HIGH_REG								0x26
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name								
Default	0x00							
Access	R/W							

By reading the MICRO\_TEMP\_LOW\_REG and MICRO\_TEMP\_HIGH\_REG registers it is possible to calculate the internal temperature value of the micro-controller using this formula:

$$\text{'}\mu\text{C temperature in }^{\circ}\text{C'} = (\text{'Read value in decimal'} * (3300/1023) - 897) / 3.35$$

**OUT\_OF\_RANGE\_STATUS\_REG**

OUT_OF_RANGE_STATUS_REG								0x24
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name	-	TEMP_OUT_OF_RANGE	V12_OUT_OF_RANGE	VCC3_OUT_OF_RANGE	VIN_OUT_OF_RANGE	VCC3_ALW_OUT_OF_RANGE	VDD_OUT_OF_RANGE	-
Default	0x00							
Access	R/W							

The read operation of the OUT\_OF\_RANGE\_STATUS\_REG register defines the status of all the monitored parameters with respect to their threshold value:

- Bit = 0 means "value inside the range"
- Bit = 1 means "alarm condition"

## Flash Registers

### FLASH\_ADDR\_H\_REG

FLASH_ADDR_H_REG								0x27
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name								
Default	0x00							
Access	R/W							

This register contains the MSB of the reading address for the flash page. This page is used for storing the log.

### FLASH\_ADDR\_L\_REG

FLASH_ADDR_L_REG								0x28
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name								
Default	0x00							
Access	R/W							

This register contains the LSB of the reading address for the flash page. This page is used for storing the log.

### FLASH\_DATA\_REG

FLASH_DATA_REG								0x29
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name								
Default	0x00							
Access	R/W							

This register contains the read Byte from the Flash that is stored at the address identified by FLASH\_ADDR\_L\_REG and FLASH\_ADDR\_H\_REG

## EEPROM Registers

### EEPROM\_ADDR\_REG

EEPROM_ADDR_REG								0x34
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name								
Default	0x00							
Access	R/W							

This register contains the read / write address of the EEPROM used to store logs.

### EEPROM\_DATA\_REG

EEPROM_DATA_REG								0x35
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name								
Default	0x00							
Access	R							

This register can contain:

- The byte read from the EEPROM at the address identified by EEPROM\_ADDR\_REG after a read command
- The data to write at the address identified by EEPROM\_ADDR\_REG before the write command

### EEPROM\_STATUS\_REG

EEPROM_STATUS_REG								0x36
Bit	B7	B6	B5	B4	B3	B2	B1	B0
Bit name								
Default	0x00							
Access	R/W							

This register contains the status of the last read/write operation done in the EEPROM used to store the logs.

## Storage of the events: the Log

Once every second the Guardian checks that the values being monitored are within specified ranges, if for any reasons these conditions are not reached, the system sets an alarm event, this event is stored using 5-bytes in the EEPROM and a log is created. Then IRQ5 is activated on the ISA bus (if previously enabled) in order to inform the CPU about the failed condition. If this condition persists the CPU will be informed automatically at one-minute intervals.

When the  $\mu\text{C}$  is turned on the time is reset to zero and a Log of all the monitored values is initiated, also monitored values will be stored in the  $\mu\text{C}$  flash memory (using the *At startup* ID codes showed in Table 2) even if no alarm is detected.

### Map of the $\mu\text{C}$ flash memory

The  $\mu\text{C}$  flash memory follows this structure:

Address	Content
0x0000 ÷ 0x33FF	Reserved
0x3400 ÷ 0x35FF	Log
0x3600 ÷ 0x37FF	Log
0x3800 ÷ 0x3FFF	Reserved

### Map of the EEPROM

The first byte of the EEPROM is a pointer to the first free location.

The first event starts at the fifth location, following this structure:

Address	Content
0x00	Pointer to the first free location
0x01	Reserved
0x02	Reserved
0x03	Reserved
0x04	Reserved
0x05	Log 1
...	
0x0A	Log 2
...	
0x0F	Log 3
...	

### Alarm log Structure

Each alarm is formed by 5 bytes and stored in the flash memory, following this rule:

Byte #	Content
1	Identifies the alarm type based on its ID code (see the "Alarm ID codes" paragraph below)
2	Contains the MSB of the value of the monitored feature
3	Contains the LSB of the value of the monitored feature
4	Contains the hour when the alarm event happened
5	Contains the minute when the alarm event happened

Table 2. Alarm ID codes

Alarm condition	Alarm condition	At start-up	Description
VDD_ALARM	0x01	0x81	VDD value is out of range
VCC3_ALWAYS_ALARM	0x02	0x82	VCC3 Always value is out of range
VIN_ALARM	0x03	0x83	VIN value is out of range
VCC3_ALARM	0x04	0x84	VCC3 value is out of the range
V12_ALARM	0x05	0x85	V12 value is out of the range
MICRO_TEMP_ALARM	0x06	0x86	µC temperature value is out of range
CPU_TEMP_ALARM	0x07	0x87	Reserved
MICRO_WTD_ALARM	0x08	0x88	Watchdog has restarted the µC
CPU_WTD_ALARM	0x09	0x89	Watchdog has restarted the CPU
POST_ALARM	0x0A	0x8A	Reserved
EEPROM_ALARM	0x0B	0x8B	Error in the EEPROM

Bytes 2 and 3 of the alarm log are null in the last five error codes.

### How to read the Log

You have to follow these steps in order to read the Log that has been created in case of a start-up or an alarm condition:

1. Write the address to be read in the register EEPROM\_ADDR\_REG
2. Give the command READ\_EPROM writing its corresponding code in the COMMAND register
3. Read the EEPROM\_DATA\_REG register

---

## Power supply CPU logic

The Guardian firmware will not enable the POWERON line until the KEY (hardware key) input is activated. Disabling the KEY input will immediately disable the POWERON line unless Guardian receives an EXCLUDE\_KEY command: in this second case the KEY input status will not affect the POWERON, and the only way to disable the CPU power line is sending the command SHUTDOWN to the Guardian.