20G025A00 E3 •2017-04-20

# G25A

## Intel Xeon D CPU Board

### 3U CompactPCI Serial

# User Manual

# Contents

## Figures



## Tables

# About this Document

This user manual is intended only for system developers and integrators, it is not intended for end users.

It describes the design, functions and connection of the product. The manual does not include detailed information on individual components (data sheets etc.).

G25A product page with up-to-date information and downloads:
www.men.de/products/g25a/

## History

| Issue | Comments | Date |
|---|---|---|
| E1 | First issue | 2016-05-11 |
| E2 | Restructuring, error correction, added chapters Hardware/Software Interface, UEFI Firmware, added G229 description | 2017-01-23 |
| E3 | General update and improvements; reworked RTC descriptions; added SGPIO description | 2017-04-20 |

## Conventions

Indicates important information or warnings concerning situations which could result in personal injury, or damage or destruction of the component.

Indicates important information concerning electrostatic discharge which could result in damage or destruction of the component.

Indicates important information or warnings concerning proper functionality of the product described in this document.

The globe icon indicates a hyperlink that links directly to the Internet. When no globe icon is present, the hyperlink links to specific information within this document.

| | |
|---|---|
| *Italics* | Folder, file and function names are printed in *italics*. |
| ***Comment*** | Comments embedded into coding examples are shown in green text. |
| IRQ#<br>/IRQ | Signal names followed by a hashtag "#" or preceded by a forward slash "/" indicate that this signal is either active low or that it becomes active at a falling edge. |
| In/Out | Signal directions in signal mnemonics tables generally refer to the corresponding board or component, "in" meaning "to the board or component", "out" meaning "from the board or component". |
| `0xFF` | Hexadecimal numbers are preceded by "`0x`". |
| `0b1111` | Binary numbers are preceded by "`0b`". |

# Product Safety

## Electrostatic Discharge (ESD)

Computer boards and components contain electrostatic sensitive devices. Electrostatic discharge (ESD) can damage components. To protect the PCB and other components against damage from static electricity, you should follow some precautions whenever you work on your computer.

- Power down and unplug your computer system when working on the inside.
- Hold components by the edges and try not to touch the IC chips, leads, or circuitry.
- Use a grounded wrist strap before handling computer components.
- Place components on a grounded antistatic pad or on the bag that came with the component whenever the components are separated from the system.
- Only store the product in its original ESD-protected packaging. Retain the original packaging in case you need to return the product to MEN for repair.

# Legal Information

## Changes

MEN Mikro Elektronik GmbH ("MEN") reserves the right to make changes without further notice to any products herein.

## Warranty, Guarantee, Liability

MEN makes no warranty, representation or guarantee of any kind regarding the suitability of its products for any particular purpose, nor does MEN assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including, without limitation, consequential or incidental damages. TO THE EXTENT APPLICABLE, SPECIFICALLY EXCLUDED ARE ANY IMPLIED WARRANTIES ARISING BY OPERATION OF LAW, CUSTOM OR USAGE, INCLUDING WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE OR USE. In no event shall MEN be liable for more than the contract price for the products in question. If buyer does not notify MEN in writing within the foregoing warranty period, MEN shall have no liability or obligation to buyer hereunder.

The publication is provided on the terms and understanding that:

1. MEN is not responsible for the results of any actions taken on the basis of information in the publication, nor for any error in or omission from the publication; and
2. MEN is not engaged in rendering technical or other advice or services.

MEN expressly disclaims all and any liability and responsibility to any person, whether a reader of the publication or not, in respect of anything, and of the consequences of anything, done or omitted to be done by any such person in reliance, whether wholly or partially, on the whole or any part of the contents of the publication.

## Conditions for Use, Field of Application

The correct function of MEN products in mission-critical and life-critical applications is limited to the environmental specification given for each product in the technical user manual. The correct function of MEN products under extended environmental conditions is limited to the individual requirement specification and subsequent validation documents for each product for the applicable use case and has to be agreed upon in writing by MEN and the customer. Should the customer purchase or use MEN products for any unintended or unauthorized application, the customer shall indemnify and hold MEN and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim or personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that MEN was negligent regarding the design or manufacture of the part. In no case is MEN liable for the correct function of the technical installation where MEN products are a part of.

## Qualified Personnel

The product/system described in this documentation may be operated only by personnel qualified for the specific task in accordance with the relevant documentation, in particular its warning notices and safety instructions. Qualified personnel are those who, based on their training and experience, are capable of identifying risks and avoiding potential hazards when working with these products/systems.

## Conformity

MEN products are no ready-made products for end users. They are tested according to the standards given in the Technical Data and thus enable you to achieve certification of the product according to the standards applicable in your field of application.

## RoHS

Since July 1, 2006 all MEN standard products comply with RoHS legislation.

Since January 2005 the SMD and manual soldering processes at MEN have already been completely lead-free. Between June 2004 and June 30, 2006 MEN's selected component suppliers have changed delivery to RoHS-compliant parts. During this period any change and status was traceable through the MEN ERP system and the boards gradually became RoHS-compliant.

## WEEE Application

The WEEE directive does not apply to fixed industrial plants and tools. The compliance is the responsibility of the company which puts the product on the market, as defined in the directive; components and sub-assemblies are not subject to product compliance.
In other words: Since MEN does not deliver ready-made products to end users, the WEEE directive is not applicable for MEN. Users are nevertheless recommended to properly recycle all electronic boards which have passed their life cycle.

Nevertheless, MEN is registered as a manufacturer in Germany. The registration number can be provided on request.



**Germany**

MEN Mikro Elektronik GmbH
Neuwieder Straße 3-7
90411 Nuremberg
Phone +49-911-99 33 5-0

info@men.de
www.men.de

**France**

MEN Mikro Elektronik SAS
18, rue René Cassin
ZA de la Châtelaine
74240 Gaillard
Phone +33-450-955-312

info@men-france.fr
www.men-france.fr

**USA**

MEN Micro Inc.
860 Penllyn Blue Bell Pike
Blue Bell, PA 19422
Phone 215-542-9575

sales@menmicro.com
www.menmicro.com

**China**

MEN Mikro Elektronik (Shanghai) Co., Ltd.
Room 808-809, Jiaxing Mansion, No. 877 Dongfang Road
200122 Shanghai
Phone +86-21-5058-0961

sales@men-china.cn
www.men-china.cn

# 1 Product Overview

## 1.1 Product Description

### High Performance for Virtualization Applications

The G25A is a high-performance multicore CPU platform based on Intel's Xeon D-1500 System-on-a-Chip (SoC). Paired with I/O cards, the G25A can be ideally used for transferring data from and to storage media, the Internet via LTE, wireless, copper or optical Ethernet. Its up to 16 processor cores and built in Intel® VT-d and VT-x support make the board ideally suited for virtualization applications.

The G25A offers high data bandwidth on both the front and the rear I/O interfaces. Two 10 Gigabit Ethernet interfaces are available on RJ45 or M12 connectors at the front, for example.

USB 3.0, SATA revision 3.x, PCI Express 3.0 and Gigabit Ethernet interfaces on front and rear complement the I/O possibilities of the board.

### Ruggedized Memory and Mass Storage

The memory configuration of the G25A includes a state-of-the-art fast DDR4 DRAM with ECC which is soldered to the board to guarantee optimum shock and vibration resistance. A microSD card device offers space for user applications or as a local boot medium.

### System Security and Board Management

For system security, a Trusted Platform Module is assembled on the board. A board management controller provides thermal supervision of the processor and a watchdog for the operating system.

### Perfect for Embedded and Harsh Environments

The G25A operates in Windows and Linux environments as well as under real-time operating systems that support Intel's multi-core architecture.

The G25A comes with a tailored heat sink within 4 HP height. All components are soldered for protection against shock and vibration according to applicable DIN, EN or IEC industry standards. The G25A is also ready for coating so that it can be used in humid and dusty environments and has a guaranteed minimum standard availability of 7 years.

### G229 I/O Extension Board

The G229 offers one m.2 slot, three USB 3.0 interfaces, one RS232 interface, one RS422/485 interface, one VGA via PCI Express Mini card and up to three flexible slots for two additional RS232 or RS422/485 interfaces. It is combined with the G25A to form a space-saving solution with a bigger heat sink.

## 1.2 External Interfaces

*Figure 1. Front interfaces - G25A*

02G025A00 02G025A01 02G025A02

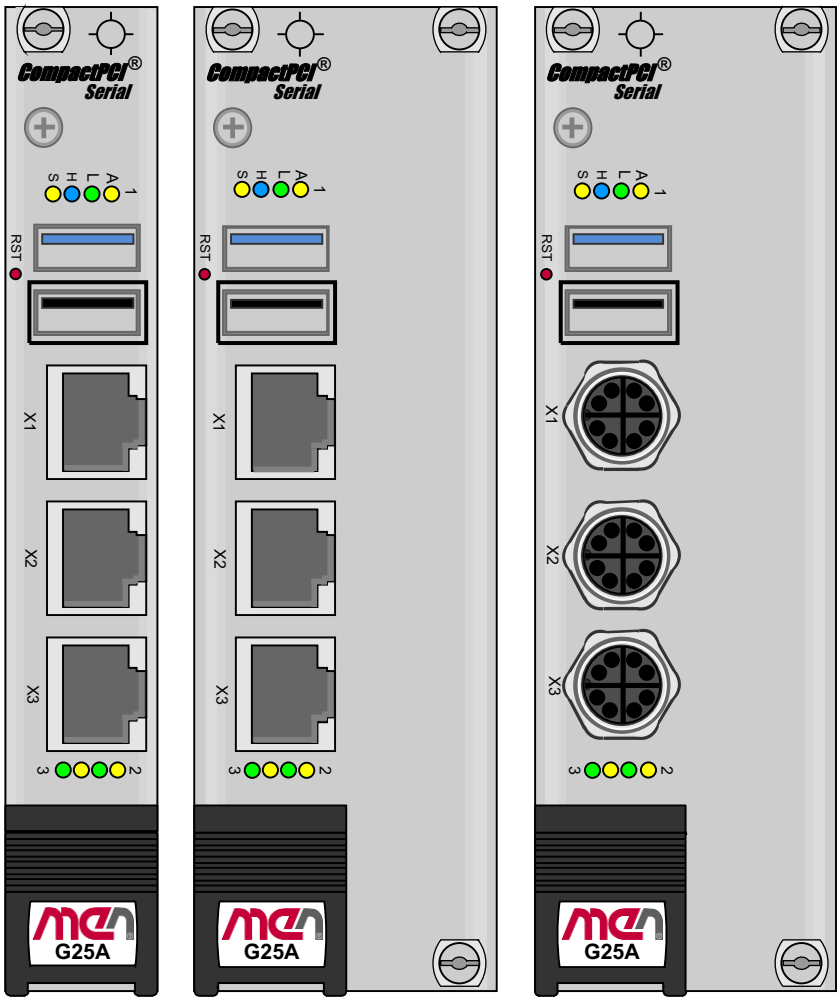
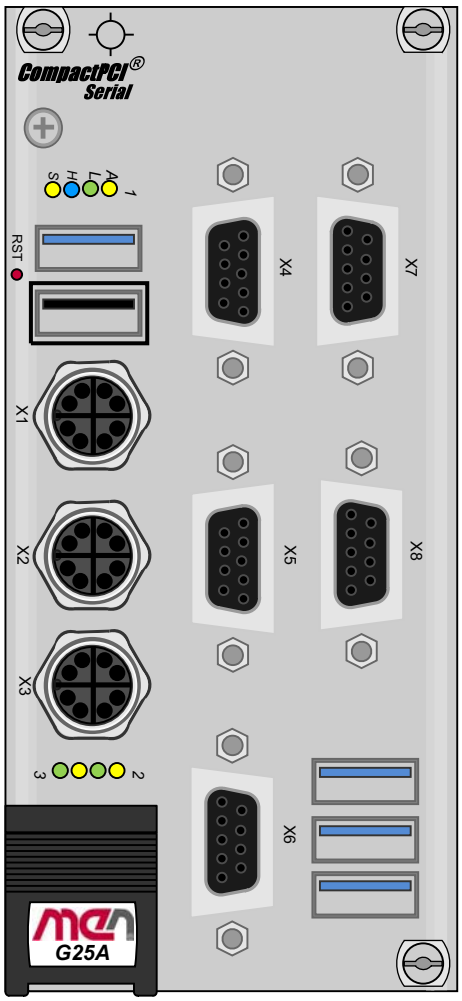


*Figure 2. Front interfaces – G25A combined with G229 peripheral board*




Up to 5 UART interfaces:

X4: RS232 or RS422/485 (UART2)

X5: RS232 or RS422/485 (UART3)

X6: RS232 or RS422/485 (UART4)
or VGA

X7: RS232 (UART0)

X8: RS422/485 (UART1)

USB 3.0

USB 3.0

USB 3.0

## 1.3 Board Layout

*Figure 3. Board layout – top view*

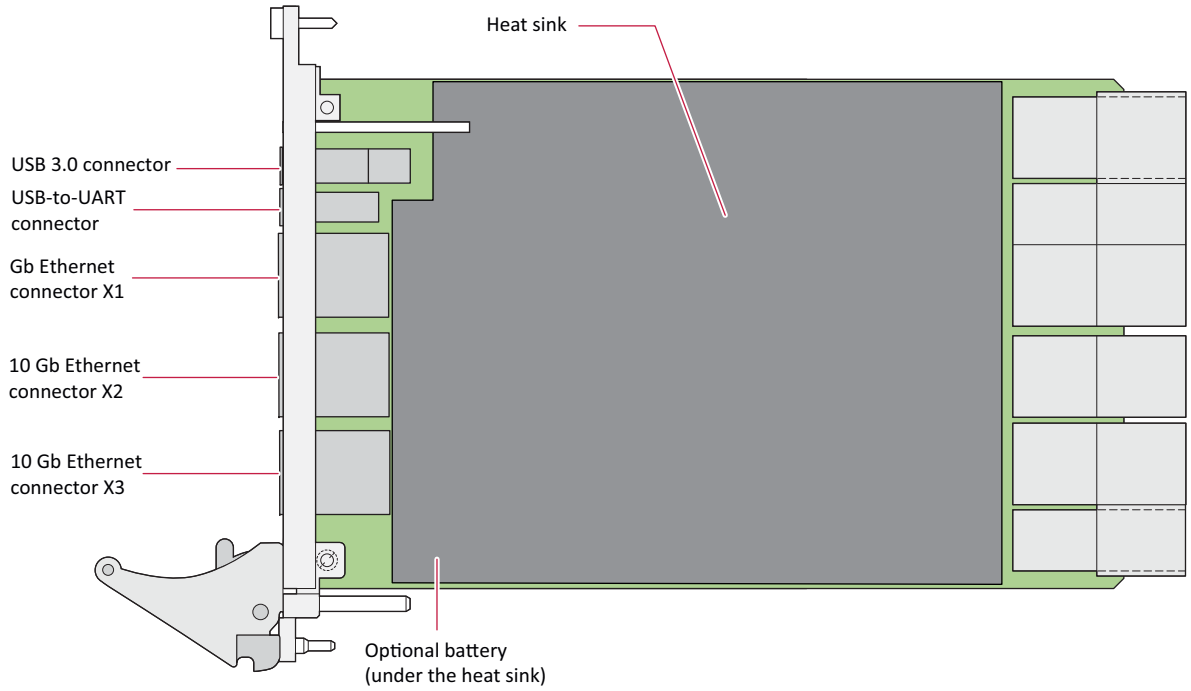
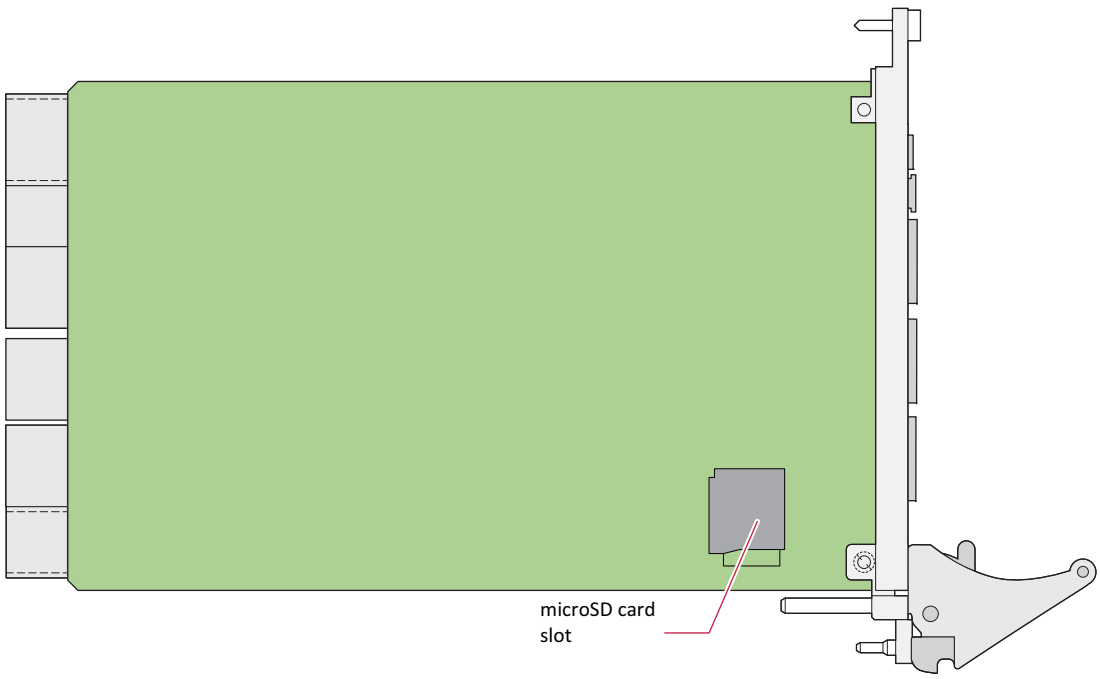



*Figure 4. Board layout – bottom view*



## 1.4 Block Diagram

*Figure 5. Block diagram - standard G25A*

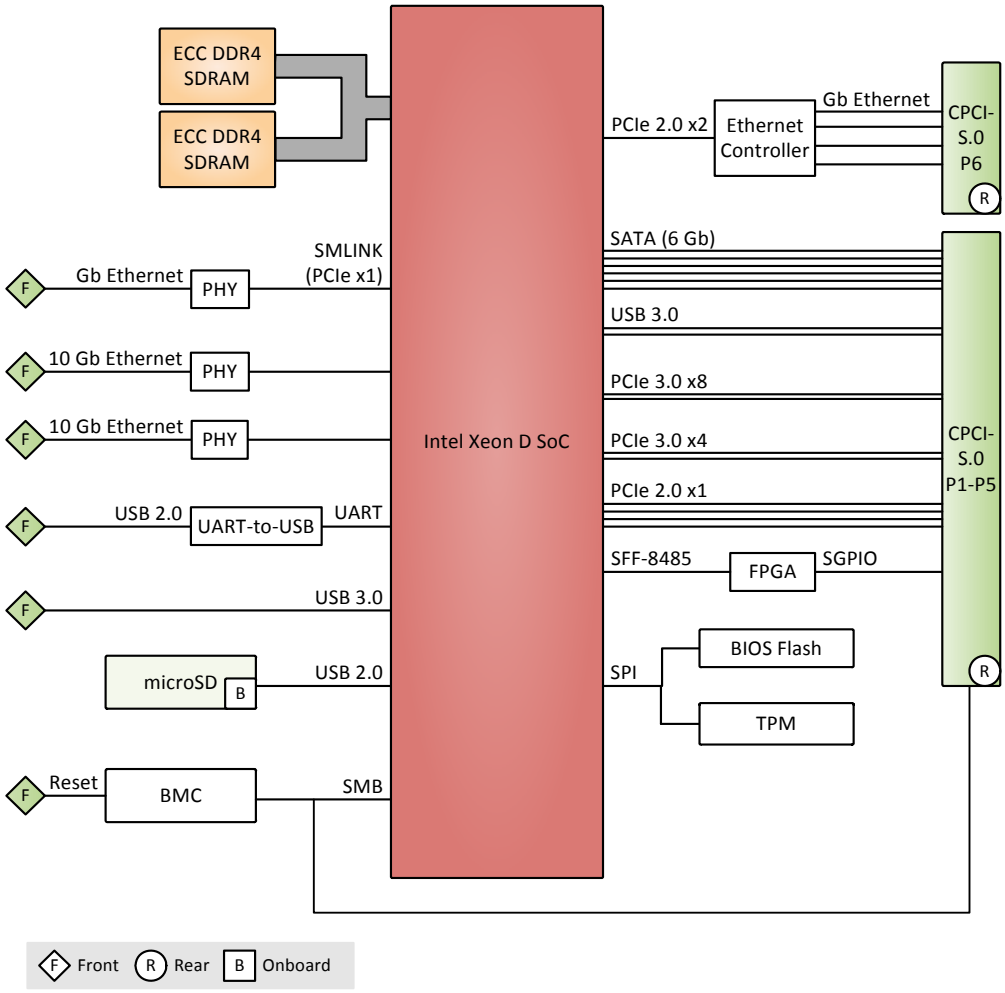
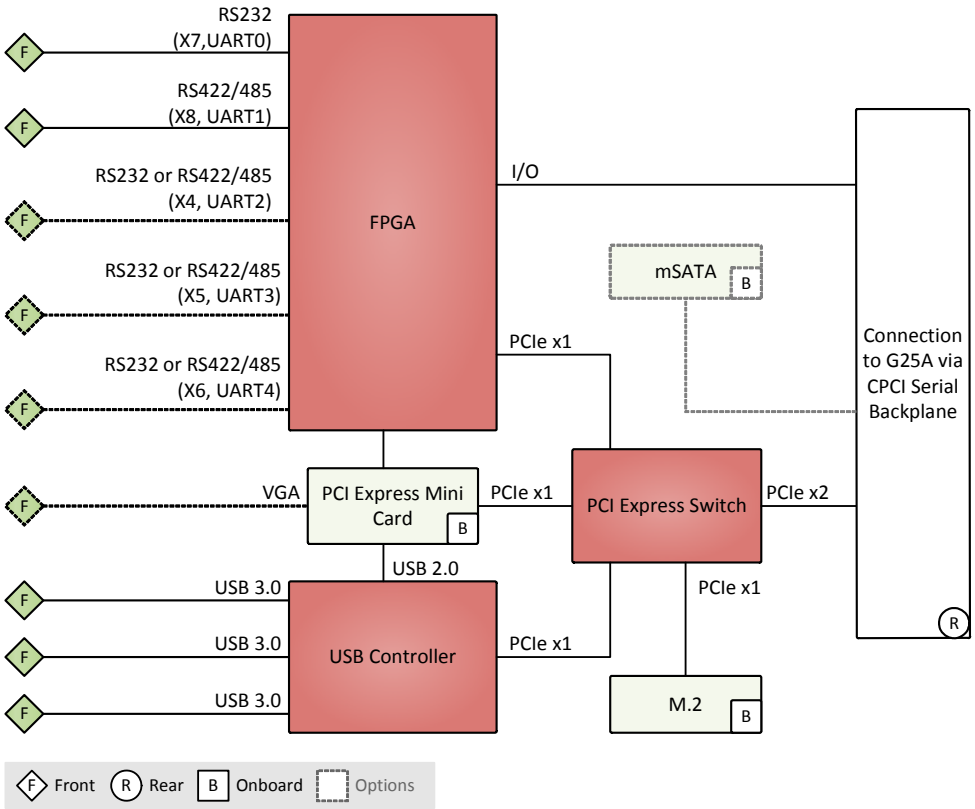


*Figure 6. Block diagram - G25A combined with G229 peripheral board*



## 1.5 Technical Data

### CPU

- The following CPU types are available:
  - Intel Xeon D-1539, 8 Cores
  - Intel Xeon D-1577, 16 Cores
  - Intel Pentium D-1519, Quad Core
- Intel Virtualization Technology (Intel VT)
  - VT-d
  - VT-x
- Intel Turbo Boost Technology
- Intel Hyper-Threading Technology

### Security

- Trusted Platform Module (TPM 2.0)

### Memory

- System Memory
  - Soldered DDR4, ECC support
  - 8 GB, 16 GB, or 32 GB
- Boot Flash
  - 16 MB

### Mass Storage

- The following mass storage device can be assembled:
  - microSD card

### Front Interfaces

- USB
  - One Type A connector, USB 3.0
  - One configuration port implemented as a USB 2.0 device interface
- Ethernet
  - Two RJ45 connectors, 10GBASE-T, or
  - Two 8-pin M12 connectors, X-coded, 10GBASE-T
  - One RJ45 connector, 1000BASE-T, or
  - One 8-pin M12 connector, X-coded, 1000BASE-T
  - Two link and activity LEDs per Ethernet channel
- Status LED
- Hot-plug LED
- Reset button

### *Rear Interfaces*

- SATA
  - Six channels, SATA Revision 3.x, RAID level 0/1/5/10 support
- USB
  - Two channels, USB 3.0
- Ethernet
  - Four channels, 1000BASE-T
- PCI Express
  - Four x1 links, PCIe 2.x
  - Two x4 links, PCIe 3.x
  - Two x8 links, PCIe 3.x

### *G229 Peripheral Board*

- Mass Storage
  - m.2 disk
- Graphics
  - None, or
  - VGA PCI Express MiniCard MPX-750, operating temperature 0°C to +40°C instead of one legacy serial I/O interface
- USB
  - Three Type A connectors, USB 3.0
- RS232
  - D-Sub connector at front panel
  - Handshake lines: RTS, CTS
- RS422/485
  - D-Sub connector at front panel
  - Full duplex
- Legacy serial I/O
  - None, or
  - Up to three interfaces via SA-Adapters:
  - RS232, not optically isolated, -40..+85°C screened, conformal coating
  - RS232, optically isolated, -40..+85°C screened, conformal coating
  - RS422/485, full duplex, optically isolated, -50..+85°C screened, conformal coating

Please contact MEN sales for ordering information.

### *Supervision and Control*

- Board controller
- Watchdog timer
- Temperature measurement
- Real-time clock with supercapacitor or battery backup
  - Data retention of supercapacitor: 72 h

### Backplane Standard

- Compliance with CompactPCI Serial PICMG CPCI-S.0 Specification
- System or peripheral slot

### Electrical Specifications

- Supply voltage
  - +12 V (9.5 to 15.5 V)
- Power consumption
  - 4.5 A typ, 6 A max. (depending on CPU and board configuration)

### Mechanical Specifications

- Dimensions
  - 3U, 4 HP, or
  - 3U, 8 HP, or
  - 3U, 12 HP together with G229 peripheral board
- Weight: 228 g (model 02G025A00 without heat sink)

### Environmental Specifications

- Temperature range (operation)
  - EN 50155 class T1, T2, T3 or TX
  - Depends on system configuration (CPU, hard disk, heat sink...)
- Temperature range (storage): -40°C to +85°C
- Cooling concept
  - Air-cooled, airflow 1.5 m/s
  - Conduction-cooled in MEN CCA frame
- Humidity: EN 60068-2-30, EN 50155
- Altitude: -300 m to +2000 m
- Shock: EN 50155 category 1 class B
- Vibration: EN 50155 category 1 class B
- Conformal coating; optional

### Reliability

- MTBF: 311 585 h @ 40°C according to IEC/TR 62380 (RDF2000) (model 02G025A00)

### Safety

- Electrical Safety
  - EN 62368-1 (former EN 60950-1)
- Flammability (PCBs)
  - UL 94 V-0

### EMC

- EN 55022 class B, EN 50121-3-2 (radiated and conducted emission)
- EN 55024, EN 50121-3-2 (immunity)

***Software Support***

- Windows
- Linux

See the MEN website for supported operating system versions, available software and more details on supported functions:
www.men.de/products/g25a/#downl

***BIOS***

- AMI Aptio

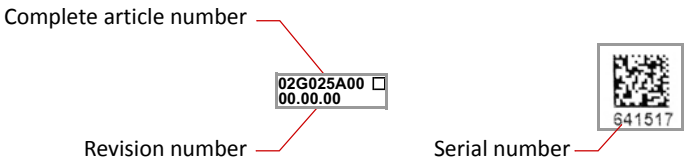
## 1.6 Product Identification

MEN documentation may describe several different models and design revisions of the G25A. You can find the article number, design revision and serial number affixed to the G25A.

- **Article number:** Indicates the product family and model. This is also MEN's main ordering number. To be complete it must have 9 characters.
- **Revision number:** Indicates the design revision of the product.
- **Serial number:** Unique identification assigned during production.

If you need support, you should communicate these numbers to MEN.

*Figure 7. Product labels*



# 2 Getting Started

## 2.1 Configuring the Hardware

Check your hardware requirements before installing the board in a system. Modifications are difficult or impossible to do when the board is integrated in a system.

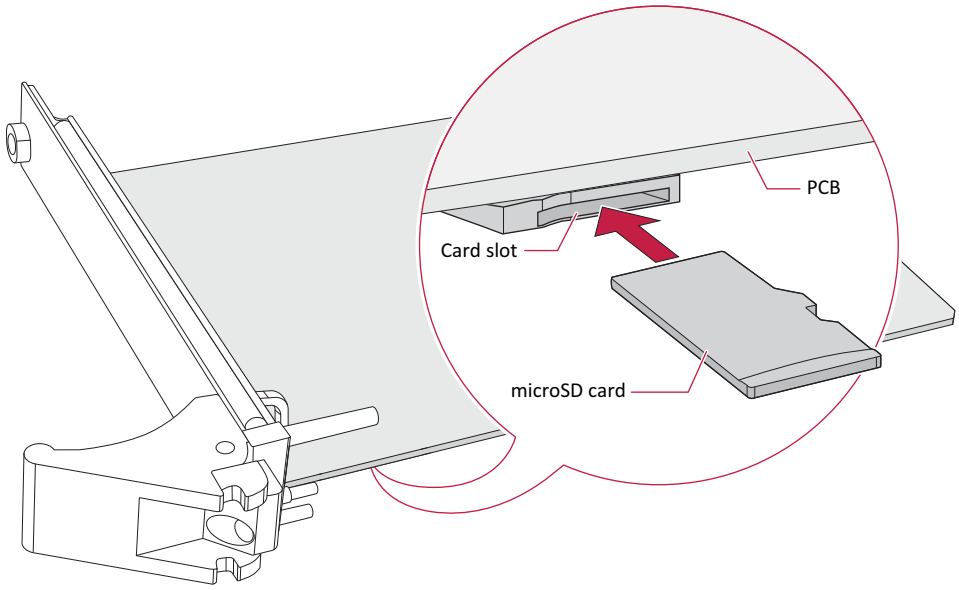
MEN offers suitable accessory articles for G25A.
See the MEN website for ordering information:
www.men.de/products/g25a/#ord

### 2.1.1 Installing a microSD Card

See Figure 4, Board layout – bottom view on page 14 for the position of the microSD slot.

The following steps are necessary:

- Power down your system and remove the G25A from the system.
- Put the board on a flat surface.
- Insert the microSD card into the slot with the contacts at the top.




- Make sure that it clicks into place properly.
- For extracting the card push it down and pull it out.

## 2.2 Connecting and Starting

You can use the following check list when installing the board in a system for the first time and with minimum configuration.

» Power down the system.

» Remove all boards from the CompactPCI Serial system.

» Insert the G25A into the system slot of your system, making sure that the backplane connectors are properly aligned.

Note: The system slot of every system is marked by a triangle with a plus sign behind it on the backplane and/or at the front panel: ⏃
It also has red guide rails.

» Connect a terminal to the RS232 interface at the front panel (USB connector).

See Chapter 3.9.3 USB-to-UART Interface on page 40.

Note: An FTDI driver must be installed on the PC your terminal is running on.

» Set your terminal to the following protocol:

- 115 200 baud data transmission rate
- 8 data bits
- 1 stop bit
- No parity

» Power up the system.

» You can access the UEFI firmware of the G25A via Console Redirection and start up the setup menu by hitting the <Esc> key.

» Now you can make configurations in the UEFI firmware.

See Chapter 4 UEFI Firmware on page 51.

### 2.2.1 Configuring the UEFI Firmware for RAID

For setting up a RAID, make the following UEFI firmware settings:

- IntelRCSetup – PCH Configuration – PCH SATA Configuration – Configure SATA as [RAID]
- IntelRCSetup – PCH Configuration – PCH SATA Configuration SATA Port xyz – Hot Plug [Enabled]
- Advanced – CSM Configuration – Option ROM Execution – Storage [Legacy]

You can now enter the RAID Option ROM if discs are present with "CTRL+ I".

For booting from Intel RAID, make the following setting:

- Boot – Boot Option #1 [Intel RAID]

## 2.3 Troubleshooting at Start-up

If you have any problems at start-up of the G25A, you can check if the front-panel status LED gives an error flash code.

See Chapter 3.3.3 Status LEDs on page 34.

You can also start the board with firmware default settings for troubleshooting.

## 2.4 Installing Operating System Software

By default, no operating system is installed on the G25A.

- Please refer to the respective manufacturer's documentation on how to implement the operating system.
- See the MEN website for all available software: www.men.de/products/g25a/#downl

### 2.4.1 Installing Windows on USB Devices

The microSD card of the G25A is connected via USB. A standard Windows operating system usually does not support direct installation on USB memory devices.

There are different solutions:

- Add a hard drive (SATA, mSATA) on a peripheral board or side card
- Switch to an Embedded Windows (like Windows Embedded Standard 7). These Embedded Windows operating systems support being installed on and booted from a USB device.

## 2.5 Installing Driver Software

For a detailed description on how to install driver software, please refer to the respective documentation of the software package to be installed.

See the MEN website for all available software: www.men.de/products/g25a/#downl

### 2.5.1 MDIS System Package

The G25A is supported by the MDIS framework.

MDIS stands for MEN Driver Interface System and is a framework for device drivers for almost any kind of I/O hardware. It greatly simplifies system configuration, also in combination with specialized board BSPs.

See the MEN website for more information on MDIS.

## 2.6 Using the G25A under Windows

This chapter describes how to use Windows software together with the G25A. A detailed step-by-step description is given where needed.

### 2.6.1 Accessing SMBus/I2C Devices

MEN provides the Windows ERTC/SMB Support Package (13Y021-70) for accessing SMBus devices, e.g., board management functions.

» Install the MDIS5 System Package for Windows.

» Install the Windows ERTC/SMB Support Package 13Y021-70.

See the MEN website for detailed information and documentation for the 13Y021-70.

- All necessary drivers are installed automatically.

All available devices are visible in the device manager.

The package provides the *xm01bc_ctrl* tool to access the board management controller (BMC) and the *smb2_eetemp tool* for accessing the temperature sensor, for example. To list all possible parameters for the tool in the command line:

```
C:\> xm01bc_ctrl <device name> <device name> is a place holder for the name of the device
```

***More information on supported functions and hardware implementation***

- See Chapter 3.3 Supervision and Management on page 33.
- See Chapter 3.5 Real-Time Clock (RTC) on page 36.
- See Chapter 5.2 SMBus/I2C Devices on page 59.

### 2.6.2 Managing RTC Time Adjustments

During the boot process, the CPU firmware gets the time from the system RTC (ERTC) and sets the CRTC accordingly. In the next step, the operating system (OS) gets the time from the CRTC and sets the system time accordingly. Now the OS system time is updated independently of the CRTC via periodic clock interrupts. Thus, over time (i.e. as the system runs), the system time may become out of sync with the CRTC/ERTC time.

If the system time is adjusted (e.g. by the user), the CRTC time will not be automatically adjusted by the time management, because the OS is not aware of the additional ERTC. The ERTC time will not be updated and is out of date. During the next system boot, the OS would use the outdated time.

MEN provides a dedicated ERTC driver to manage system time adjustments.

See the MEN website for the User Manual " Windows ERTC/SMB Support Package".

### 2.6.3 Configuring the UART Interfaces

MEN's driver installation package (Installset) for Windows (13F215-77) allows easy configuration through the Device Manager.

To do this, open the *Properties* page of each G25A UART device via the Windows Device Manager and select the *Port Interface* tab.

See the MEN website for the Windows Installset and user manual:
www.men.de/software/13f215-77/

See the block diagram and the front panel drawing in Chapter 1 Product Overview to find out the interface names and numbers.

## 2.7 Using the G25A under Linux

This chapter describes how to use Linux software together with the G25A. A detailed step-by-step description is given where needed.

### 2.7.1 Accessing Board Management Functions

There are two ways to access board management functions, e.g., the board management controller (BMC), under Linux:

- Using MEN software tools for Linux.
- Using standard Linux I2C tools.

***More information on supported functions and hardware implementation***

- See Chapter 3.3 Supervision and Management on page 33.
- See Chapter 5.4 BMC API (Application Programming Interface) on page 62 for a detailed description of the BMC API.
- See Chapter 5.2 SMBus/I2C Devices on page 59.

#### 2.7.1.1 MEN Tools

MEN provides a number of MDIS software tools for accessing G25A functions via the SMBus, which are included in the MDIS5 System Package for Linux,13MD05-90:

- *xm01bc_ctrl* is the tool for accessing the BMC
- *smb2_eetemp* is the tool for accessing the temperature sensor

In the following you can find an exemplary description of how to access the BMC:

» Install the MDIS5 System Package for Linux.

- See the MEN website for details on the MEN Driver Interface System, and all available software packages.
- See the MDIS5 System Package for Linux documentation for a detailed description of the configuration of SMBus devices.
  We recommend that you read the MDIS5 under Linux User Manual, especially Chapter A 10.1.6 Adding SMB Devices, before continuing.

» Load the driver for the BMC using the following commands:

```
$ sudo modprobe men_mdis_kernel
$ sudo modprobe men_ll_xm01bc
```

» Print a list of all possible parameters of the *xm01bc_ctrl* tool:

```
$ sudo xm01bc_ctrl <device name> <device name> is a place holder for the name of the device

Usage: xm01bc_ctrl [<opts>] <device> [<opts>]

Function: Control XM01BC PIC

Options:
  device      device name
  -v          show voltage values
  -s          do voltage supervision (requires option -v)

  -r=0xdead   perform SW warm reset (!! dangerous !!)
  -R=0xdead   perform SW cold reset (!! dangerous !!)

  -e          show error counters
  -c          clear error counters
  -n          get number of error counters

  -f          show firmware revision
  -F          show firmware revision extended

  -w          show last reset reason
  -x          show last error
  -y          show power failure flag
  -z          clear failure registers

  -o          show operating hours counter
  -p          show power cycle counter

  -l          get LED state
  -L=<state>  set LED state

  -u          get power resume mode
  -a          get EXT_PWR_OK resume mode
  -b          get RESET_IN mode

  -h          get hardware variant ID

  -q          exit QM-Mode (for production tests only)

(c) 2008 by MEN mikro elektronik GmbH
```

» For example, if you want to look up the voltage values, use the following command:

```
$ sudo xm01bc_ctrl -v xm01bc_1
```

#### 2.7.1.2 Standard Linux I2C Tools

Carry out the following steps to access the BMC.

This procedure is also applicable to other SMBus devices.

» Find out the number of the SMBus the BMC is located on by listing the I2C devices:

```
$ sudo i2cdetect -l (small l not number 1)

[...]
i2c-x smbus     SMBus adapter                   SMBus adapter
[...]

For example:
i2c-9 smbus     SMBus adapter                   SMBus adapter
```

» Look up the SMBus address of the BMC in Chapter 5.2 SMBus/I2C Devices on page 55. In this example, the BMC address is 0x4D (7-bit notation).

» Display the devices of the SMBus (i2c-9) to look for the BMC address 0x4D:

```
$ sudo i2cdetect -y 9

     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- 08 -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- 1f
20: 20 21 22 -- -- -- -- 27 -- -- -- -- -- -- -- --
30: -- -- 32 -- -- -- -- 37 -- -- -- -- -- -- -- --
40: -- -- -- -- 44 -- -- -- -- -- -- -- -- 4d -- --
50: -- -- -- -- -- -- 56 57 -- -- -- -- -- -- -- --
60: 60 -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: 70 -- -- -- -- -- -- --
```

» In this example, the BMC has the address 0x4D, i.e. it is located on SMBus number 9.

» There might be several SMBuses in the list. If you do not find the BMC address on the first SMBus, list the devices on the other SMBuses. For SMBus 1 (i2c-1), for example, use this command:

```
$ sudo i2cdetect -y 1
```

» When you have found out the address of the BMC and the number of the SMBus you can dump the registers of the BMC using this command:

```
$ sudo i2cdump -y 9 0x4d
```

» Read out values using command *i2cget*:

```
$ sudo i2cget -y 9 0x4d
```

In the following you can find some examples showing how to use the watchdog:

- Setting the watchdog time to 10 seconds (0x64):

```
$ sudo i2cset -y 9 0x4d 0x14 0x64 w
```

- Enabling the watchdog (if it's not enabled inside the BIOS firmware):

```
$ sudo i2cset -y 9 0x4d 0x11 0x00
```

- Triggering the watchdog:.

```
$ sudo i2cset -y 9 0x4d 0x13 0x00
```

See Chapter 5.3 BMC API (Application Programming Interface) on page 56 for a detailed description of the BMC API.

### 2.7.2 Managing RTC Time Adjustments

During the boot process, the CPU firmware gets the time from the system RTC (ERTC) and sets the CRTC accordingly. In the next step, the operating system (OS) gets the time from the CRTC and sets the system time accordingly. Now the OS system time is updated independently of the CRTC via periodic clock interrupts. Thus, over time (i.e. as the system runs), the system time may become out of sync with the CRTC/ERTC time.

If the system time is adjusted (e.g. by the user), the CRTC time will not be automatically adjusted by the time management, because the OS is not aware of the additional ERTC. The ERTC time will not be updated and is out of date. During the next system boot, the OS would use the outdated time.

MEN provides a dedicated ERTC driver to manage system time adjustments.

See the MEN website for the Application Note "Using the System RTC (ERTC) on MEN CPUs under Linux".

### 2.7.3 Configuring the UART Interfaces

MEN provides a Linux driver that allows to configure the interface mode and baud rate.It is included in the 13MD05-90 MDIS5 system package for Linux.

See the MEN website for the MDIS system package:
www.men.de/software/13md05-90/

The *baud_base* parameter must be set to 1843200. Use the following command:

```
# modprobe men_lx_z25 baud_base=1843200
```

MEN's Linux driver supports the following values for the *mode* parameter:

| | |
|---|---|
| **se** | single ended (RS232) |
| **df_fdx** | differential, full duplex (RS422) |
| **df_hdxe** | differential, half duplex, with echo (RS485) |
| **df_hdx** | differential, half duplex, no echo (RS485) |

Note: You can check using *dmesg* whether all */dev/ttySx* interfaces are present.

Most Linux kernels only support 4 UARTs by default. If you need more than 4 UARTs, add parameter *8250.nr_uarts=16* to your kernel boot line in the bootloader or adjust kernel parameter *CONFIG_NR_8250_UARTS* and recompile the kernel.

See the block diagram and the front panel drawing in Chapter 1 Product Overview to find out the interface names and numbers.

### 2.7.4 Controlling SATA SGPIO Functions

The G25A implementation does **not support** the standard Linux tools *sgpio* and *ledmon*.

There are two ways to access SGPIO functions under Linux:

- Using the MEN software tool included in the MDIS5 System Package for Linux, 13MD05-90.
- Using standard Linux I2C tools.

***More information on supported functions and hardware implementation***

- See Chapter 5.3 SATA SGPIO FPGA on page 60 for a register description.
- See Chapter 2.7.1.2 Standard Linux I2C Tools on page 28 for a more detailed description of how to access SMBus devices using standard Linux tools.

#### 2.7.4.1 MEN Tool

The MEN MDIS tool provided to access SMBus devices is called *smb2_ctrl* (MEN article number 13Y004-06).

> See the HTML documentation included with the tool in the MDIS5 framework for a detailed description.

For example, to enable the locate LED of drive 0:

```
$ sudo smb2_ctrl smb2_2 wbd -a=0xe0 -o=0x00 -d=0x40
```

#### 2.7.4.2 Standard Linux I2C Tools

You can use *i2cset* to control the SATA SGPIO functions. The basic parameters to be used for the command are:

```
$ sudo i2cset -y <SMBus number> <device address> <drive number> <data>
```

Take care to use the correct SMBus number applicable for your system.

***Examples***

- Enabling activity LED of drive 1:

  ```
  $ sudo i2cset -y 1 0x70 0x01 0x80
  ```

- Disabling activity LED of drive 1:

  ```
  $ sudo i2cset -y 1 0x70 0x01 0x00
  ```

- Enabling locate LED of drive 1:

  ```
  $ sudo i2cset -y 1 0x70 0x01 0x40
  ```

- Enabling hot-plug LED of drive 2:

  ```
  $ sudo i2cset -y 1 0x70 0x02 0x20
  ```

- Disabling *SATA_DETECT* bit:

  ```
  $ sudo i2cset -y 1 0x70 0xFE 0x80
  ```

- Enabling *SATA_DETECT* bit:

  ```
  $ sudo i2cset -y 1 0x70 0xFE 0x00
  ```

> ***More information on supported functions and hardware implementation***
>
> - See Chapter 5.3 SATA SGPIO FPGA on page 60 for a register description.
> - See Chapter 2.7.1.2 Standard Linux I2C Tools on page 28 for a more detailed description of how to access SMBus devices using standard Linux tools.

# 3 Functional Description

## 3.1 Power Supply

The G25A board is supplied with +12V only. The voltage range is +9.5 V up to +15.5 V (absolute maximum voltage). The voltage is monitored within these limits.

## 3.2 Processor Core

The following CPU types are available:

*Table 1. Processor core options on G25A*

| Processor Type | Core Frequency | Cores | Power Consumption | Cache |
|---|---|---|---|---|
| Xeon D-1577 | 1.3 GHz | 16 | 45 W | 24 MB |
| Xeon D-1559 | 1.5 GHz | 12 | 45 W | 18 MB |
| Xeon D-1548 | 2 GHz | 8 | 45 W | 12 MB |
| Xeon D-1539 | 1.6 GHz | 8 | 35 W | 12 MB |
| Xeon D-1529 | 1.3 GHz | 4 | 25 W | 6 MB |
| Xeon D-1527 | 2.2 GHz | 4 | 35 W | 6 MB |
| Xeon D-1519 | 1.5 GHz | 4 | 25 W | 6 MB |
| Xeon D-1509 (on request) | 1.5 GHz | 2 | 19 W | 3 MB |
| Xeon D-1508 | 2.2 GHz | 2 | 25 W | 3 MB |

### 3.2.1 Thermal Considerations

The power dissipation of G25A heavily depends on its processor and I/O configuration and on the workload.

Power dissipation of a G25A equipped with a high-end 45 W processor is up to 72 W (45 W for the high-end processor and 27 W for the rest of the board).

The G25A provides a very high computing power on a small space. For this reason, it is vital to provide sufficient airflow (1.5 m/s). A suitable heat sink is provided to meet thermal requirements.

If you use any other heat sink than that supplied by MEN, or no heat sink at all, warranty on functionality and reliability of the G25A may cease. Please contact MEN if you have any questions or problems regarding thermal behavior.

## 3.3 Supervision and Management

The G25A provides an intelligent board management controller (BMC) with the following main features:

- System watchdog
- Operating hours counter
- Power cycle counter
- Voltage supervision
- Error state logging
- Temperature measurement

### 3.3.1 Watchdog

The watchdog device monitors the CPU board on operating system level. If enabled, the watchdog must be triggered by application software. If the trigger is overdue, the watchdog initiates a board reset and in this way can put the system back into operation when the software hangs.

The watchdog unit can be enabled or disabled, as required and the watchdog timeout can be set in 100-ms steps from 100 ms up to 1:49:10 (hh:mm:ss) - 65536 steps.

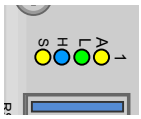
### 3.3.2 Temperature Measurement

The G25A uses a temperature device to measure the local CPU board temperature. The temperature device is connected to the BMC via I2C, and is supported by the firmware.

### 3.3.3 Status LEDs

The G25A has two general status LEDs at the front panel which are controlled by the board controller using SMBus commands.

*Table 2. General status LEDs at front panel*

| Appearance | Label | Color | Function |
|---|---|---|---|
|  | *S* | Yellow | Board status LED |
| | *H* | Blue | Hot-Swap LED |

#### 3.3.3.1 Board Status LED

The yellow status LED shows G25A status messages. The LED is controlled by a GPIO pin of the board controller. It is switched on when the G25A firmware starts, switched off when the G25A is switched off and flashing when the board is in stand-by (S3) status.

During normal operation the LED can be switched on and off using software.

Please refer to the ACPI Specification for more details on the power states S0 to S5:
Advanced Configuration and Power Interface Specification Version 6.1
January, 2016
Unified EFI Forum
uefi.org/specifications

In case of an error, the LED displays the following error messages by repeatedly flashing *n* times and then pausing for one second:

*Table 3. Error codes signaled by board management controller via LED flashes*

| Number of Flashes | Error |
|---|---|
| 1 | +3.3 V failure |
| 2 | Input voltage failure |
| 3 | External power supply failure |
| 4 | CPU too hot |
| 5 | G25A firmware timeout |
| >5 | Internal error |

#### 3.3.3.2 Hot-Swap LED

The hot-swap LED signals the hot-swap status.

*Table 4. Hot-swap LED*

| Hot-Swap LED | Description |
|---|---|
| Off | The hot-swap switch is closed. |
| Starts flashing | The hot-swap switch is opened while the system is running (start of hot-swap sequence). |
| Lights continuously | The system has been shut down (end of hot-swap sequence). |
| | The hot-swap switch is open during power-up. The board controller delays the power-up sequence until the hot-swap switch is closed. |
| Stops flashing | The hot-swap switch has been closed while the hot-swap sequence is in progress. The board controller no longer waits for system shutdown. |
| Inactive | The CPU is plugged into a system slot or is in S0 or S3 state. The board controller ignores the hot-swap switch. |

If the hot-swap switch is closed after system shutdown, the board controller initiates Power Resume

Please refer to the ACPI Specification for more details on the power states S0 to S5:
Advanced Configuration and Power Interface Specification Version 6.1
January, 2016
Unified EFI Forum
uefi.org/specifications

#### 3.3.3.3 Software Support

Supervision and management functions are accessible by software.

- See Chapter 2.6.1 Accessing SMBus/I2C Devices on page 24 and Chapter 2.7.1 Accessing Board Management Functions on page 26 for details on using MEN driver software.
- See Chapter 5.4 BMC API (Application Programming Interface) on page 62 for a detailed documentation of the BMC API.

## 3.4 Reset

The G25A is equipped with a reset button which is recessed within the front panel and requires a tool, e.g. paper clip to be pressed, preventing the button from being inadvertently activated.

## 3.5 Real-Time Clock (RTC)

The G25A includes a real-time clock connected to the processor as the system RTC (ERTC) RX-8571. The RTC has an accuracy of approximately 1.7 seconds/day (11 minutes/year) at 25°C. The real-time clock device is connected to the CPU via SMBus.

- **ERTC (External Real-Time Clock)** is a real-time clock additionally connected to the processor as a system RTC.
- **CRTC (Chipset Real-Time Clock)** is the real-time clock of the processor.

For data retention during power off the RTC is backed up by a supercapacitor. The supercapacitor gives an autonomy of approx. 72 hours (+/-20%) when fully charged. Worst case derating is -30% after 1.67 years at +40°C board temperature.

For retention of time/date data after a power off of more than 8-10 hours the RTC can optionally be backed by a battery.

### 3.5.1 Software Support

Please note that the real-time clock integrated in the processor is **not used**. Configuring the date and time through the means provided by the operating system **does not set** the system RTC.

You can set the system date and time through the UEFI firmware.

If you use dedicated MEN driver software supporting the system RTC, you can use the functions provided there to set the system RTC also via software.

- See Chapter 2.6.2 Managing RTC Time Adjustments on page 24 for more information on how to configure the RTC under Windows.
- See Chapter 2.7.2 Managing RTC Time Adjustments on page 29 for more information on how to configure the RTC under Linux.

## 3.6 Trusted Platform Module (TPM)

A trusted platform module for authenticating the hardware to ensure platform integrity is available on the G25A. The TPM module is compliant to the TPM v2.0 specification.

## 3.7 Memory

### 3.7.1 DRAM System Memory

The DRAM system memory of G25A is scalable.

- See Chapter 1.5 Technical Data on page 17.
- See the MEN website for available standard configurations: www.men.de/products/g25a/#ord

### 3.7.2 Boot Flash

The G25A provides 16 MB boot Flash memory. It contains the UEFI firmware and the Intel Management Engine firmware.

## 3.8 Mass Storage

### 3.8.1 microSD Card Slot

Within its housing, the G25A provides one microSD card slot.

Supported standards:

- Secure Digital 1.0 specification (microSD)
- Secure Digital 2.0 specification (microSDHC)

Supported speed classes:

- Class 2 / 4 / 6 / 8 / 10
- UHS-I / UHS-II

MEN offers suitable accessory articles for G25A.
See the MEN website for ordering information:
www.men.de/products/g25a/#ord

See Chapter 2.1.1 Installing a microSD Card on page 21 for information on how to install a microSD card.

### 3.8.2 Serial ATA (SATA)

The G25A supports

- AHCI operation
- RAID operation

  See Chapter 4.3.2.5 MEN Menu – Sub-Menu SATA Settings on page 55 for firmware configuration options.

- SGPIO for SATA0 to SATA5

#### 3.8.2.1 SGPIO

SGPIO functions allow to read out the geographical address of SATA drives in the system or to switch shuttle LEDs, e.g., a locate LED.

#### 3.8.2.2 Connection

- See the CompactPCI Serial standard PICMG CPCI-S.0 for the exact position of the SATA ports on the rear I/O connectors.
- See Chapter 3.14 CompactPCI Serial on page 48 for the position of the SATA interfaces in a CompactPCI Serial system.

#### 3.8.2.3 Using the G25A in a CompactPCI Serial Peripheral Slot

If the G25A is used in a peripheral slot, the interfaces on the CompactPCI Serial connectors are switched off.

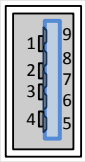
## 3.9 USB

### 3.9.1 Front Connection

*Table 5. Connector types – USB 3.0*

| Connector | Type |
|---|---|
| On G25A | 9-pin USB 3.0 Standard-A receptacle according to Universal Serial Bus Specification |
| Mating | 9-pin USB 3.0 Standard-A plug according to Universal Serial Bus Specification |

*Table 6. Pin assignment – USB 3.0*




| | | | |
|---|---|---|---|
| 1 | +5V | 9 | StdA_SSTX+ |
| 2 | D- | 8 | StdA_SSTX- |
| 3 | D+ | 7 | GND |
| 4 | GND | 6 | StdA_SSRX+ |
| | | 5 | StdA_SSRX- |

*Table 7. Signal mnemonics – USB 3.0*

| Signal | Direction | Function |
|---|---|---|
| +5V | out | +5 V power supply |
| GND | - | Digital ground |
| D+, D- | in/out | USB 2.0 differential pair |
| StdA_SSTX+, StdA_SSTX- | out | SuperSpeed transmitter differential pair |
| StdA_SSRX+, StdA_SSRX- | in | SuperSpeed receiver differential pair |

### 3.9.2 Rear Connection

- Refer to the CompactPCI Serial standard PICMG CPCI-S.0 for the exact position of the USB ports on the rear I/O connectors.
- See Chapter 3.14 CompactPCI Serial on page 48 for the exact position of the USB ports in a CompactPCI Serial system.

#### 3.9.2.1 Using the G25A in a Peripheral Slot

If the G25A is used in a peripheral slot, the USB interface on the CompactPCI Serial connectors is switched off.

### 3.9.3 USB-to-UART Interface

The USB signals of this interface are converted to UART signals using a bridge chip, e.g., when you connect a PC or other remote station using a USB port. The remote computer does not need a UART port.

The interface supports:

- Data rates up to 4 Mbit/s
- 512-byte transmit buffer
- 512-byte receive buffer

See the MEN website for a service cable for USB:
www.men.de/products/g25a/#ord

## 3.10 Ethernet

Implementation:

- 10 Gigabit Ethernet interfaces:
  - 10 Gigabit Ethernet MACs integrated in the System-on-a-Chip combined with an Intel X557 Dual PHY
- Gigabit Ethernet interfaces:
  - Gigabit Ethernet MAC integrated in the SOC combined with an Intel I218LM PHY

Note: The 10 Gigabit interfaces (10GBASE-T) of the G25A do not support Fast Ethernet interfaces (10/100BASE-T).

Supported cable lengths:

- 1 Gigabit Ethernet interfaces:
  - 100 m (M12 und RJ45)
- 10 Gigabit Ethernet interfaces:
  - 55 m (M12)
  - 45 m (RJ45)

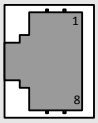
### 3.10.1 Front Connection

#### 3.10.1.1 RJ45

*Table 8. Connector types – Ethernet (RJ45)*

| Connector | Type |
|---|---|
| On G25A | Modular 8/8-pin receptacle according to FCC68 |
| Mating | Modular 8/8-pin plug according to FCC68 |

*Table 9. Pin assignment – Ethernet (RJ45)*

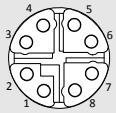
| | | 1000BASE-T/ 10GBASE-T |
|---|---|---|
|  | 1 | BI_DA+ |
| | 2 | BI_DA- |
| | 3 | BI_DB+ |
| | 4 | BI_DC+ |
| | 5 | BI_DC- |
| | 6 | BI_DB- |
| | 7 | BI_DD+ |
| | 8 | BI_DD- |

#### 3.10.1.2 M12

*Table 10. Connector types – Ethernet M12*

| Connector | Type |
|---|---|
| On G25A | 8-pin M12 receptacle X-coded, e.g., Phoenix Contact SACC-CI-M12FSX-8CON-L90 |
| Mating | 8-pin M12 plug X-coded |

*Table 11. Pin assignment – Ethernet (8-pin M12)*

| | | 10GBASE-T/ 1000BASE-T |
|---|---|---|
|  | 1 | BI_DA+ |
| | 2 | BI_DA- |
| | 3 | BI_DB+ |
| | 4 | BI_DB- |
| | 5 | BI_DD+ |
| | 6 | BI_DD- |
| | 7 | BI_DC- |
| | 8 | BI_DC+ |

### 3.10.2 Rear Connection

- Refer to the CompactPCI Serial standard PICMG CPCI-S.0 for the exact position of the Ethernet ports on the rear I/O connectors.
- See Chapter 3.14 CompactPCI Serial on page 48 for the exact position of the Ethernet ports in a CompactPCI Serial system.

### 3.10.3 Ethernet Signal Mnemonics

*Table 12. Signal mnemonics – Ethernet*

| Signal | Direction | Function |
|---|---|---|
| BI_Dx+/- | in/out | Differential pairs of data lines for 1000BASE-T or 10GBASE-T |

### 3.10.4 Ethernet MAC Addresses

The unique MAC address is set at the factory and should not be changed. Any attempt to change this address may create node or bus contention and thereby render the board inoperable.

The naming of the interfaces may differ depending on the operating system. The MAC addresses on G25A are:

*Table 13. Ethernet MAC addresses*

| Interface | Position | Base Address |
|---|---|---|
| X1 | Front | `0x 00 C0 3A D0 80 00` |
| X2 | Front | `0x 00 C0 3A D0 90 00` |
| X3 | Front | `0x 00 C0 3A D0 A0 00` |
| Rear interface 1 | Slot 2 on a standard CompactPCI Serial backplane | `0x 00 C0 3A D0 B0 00` |
| Rear interface 2 | Slot 3 on a standard CompactPCI Serial backplane | `0x 00 C0 3A D0 C0 00` |
| Rear interface 3 | Slot 4 on a standard CompactPCI Serial backplane | `0x 00 C0 3A D0 D0 00` |
| Rear interface 4 | Slot 5 on a standard CompactPCI Serial backplane | `0x 00 C0 3A D0 E0 00` |

"00 C0 3A" is the MEN vendor code. The last six digits form the unique MAC address for each board. The serial number is added by the last three digits in the range:

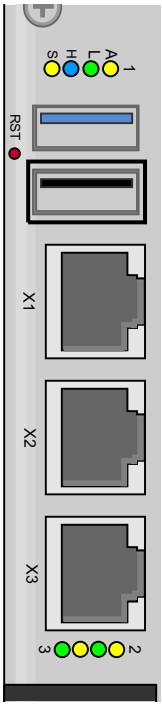
Serial number 42 (X1): `0x8000 + 0x002A = 0x802A.`

See Chapter 1.6 Product Identification on page 20.

### 3.10.5 Ethernet Status LEDs

See Chapter 1.2 External Interfaces on page 12 for the position of the LEDs.

*Table 14. Ethernet status LEDs at front panel*

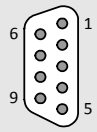
| Appearance | Label | Color | Function |
|---|---|---|---|
|  | *1/A* | Yellow | Activity LED<br>▪ On: Tx/Rx activity<br>▪ Off: No activity<br>▪ Blinking: Tx/Rx activity |
| | *1/L* | Green | Linkup LED<br>▪ On: Link up<br>▪ Off: No link<br>▪ Blinking: n/a |
| | *2* | Yellow | ▪ On: Link up<br>▪ Off: No link<br>▪ Blinking: Tx/Rx activity |
| | *2* | Green | ▪ On: 10 Gb link<br>▪ Off: Gb link<br>▪ Blinking: n/a |
| | *3* | Yellow | ▪ On: Link up<br>▪ Off: No link<br>▪ Blinking: Tx/Rx activity |
| | *3* | Green | ▪ On: 10 Gb link<br>▪ Off: Gb link<br>▪ Blinking: n/a |

## 3.11 RS232 (on G229 Peripheral Board)

*Table 15. Connector types – 9-pin D-Sub plug*

| Connector | Type |
|---|---|
| On G25A | 9-pin D-Sub plug according to DIN41652/MIL-C-24308, with thread bolt UNC4-40 |
| Mating | 9-pin D-Sub receptacle according to DIN41652/MIL-C-24308, available for ribbon cable (insulation piercing connection), hand-soldering connection or crimp connection |

*Table 16. Pin assignment – RS232 (9-pin D-Sub)*




| | | | | |
|---|---|---|---|---|
| 1 | - | 6 | - |
| 2 | RXD | 7 | RTS# |
| 3 | TXD | 8 | CTS# |
| 4 | - | 9 | - |
| 5 | GND | | |

*Table 17. Signal mnemonics – RS232*

| Signal | Direction | Function |
|---|---|---|
| GND | - | Ground |
| CTS# | in | Clear to send |
| RTS# | out | Request to send |
| RXD | in | Receive data |
| TXD | out | Transmit data |

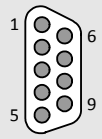
## 3.12 RS422/485 (on G229 Peripheral Board)

*Table 18. Connector types – 9-pin D-Sub receptacle*

| Connector | Type |
|---|---|
| On G25A | 9-pin D-Sub receptacle according to DIN41652/MIL-C-24308, with thread bolt UNC4-40 |
| Mating | 9-pin D-Sub plug according to DIN41652/MIL-C-24308, available for ribbon cable (insulation piercing connection), hand-soldering connection or crimp connection |

### 3.12.1 Full-Duplex Interface

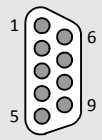
*Table 19. Pin assignment – RS422/485 full duplex (9-pin D-Sub)*




| | | | |
|---|---|---|---|
| 1 | - | 6 | I-VCC |
| 2 | - | 7 | - |
| 3 | TX+ | 8 | TX- |
| 4 | RX+ | 9 | RX- |
| 5 | I-GND | | |

### 3.12.2 Half-Duplex Interface

*Table 20. Pin assignment – RS422/485 half duplex (9-pin D-Sub)*




| | | | |
|---|---|---|---|
| 1 | - | 6 | I-VCC |
| 2 | - | 7 | - |
| 3 | RX/TX+ | 8 | RX/TX- |
| 4 | - | 9 | - |
| 5 | I-GND | | |

### 3.12.3 Signal Mnemonics

*Table 21. Signal mnemonics – RS422/485*

| Signal | Direction | Description |
|---|---|---|
| I-GND | - | Isolated reference potential |
| I-VCC | out | Isolated power supply |
| RX+/- | in | Receive data |
| TX+/- | out | Transmit data |

## 3.13 PCI Express

### Rear I/O connection

The following PCI Express links are available at the G25A's rear I/O connectors:

- Two PCI Express 3.0 x8 links
- Two PCI Express 3.0 x4 links
- Four PCIe 2.0 x1 links

Refer to the CompactPCI Serial standard PICMG CPCI-S.0 for the exact position of the PCI Express ports on the rear I/O connectors. See Chapter 3.14 CompactPCI Serial on page 48.

If the G25A is used in a peripheral slot, the interfaces on the CompactPCI Serial connectors are switched off.

### Connection of Gigabit Ethernet controller

The controller for the four Gigabit Ethernet channels at the rear is permanently connected via one PCIe 2.0 x2 link.

## 3.14 CompactPCI Serial

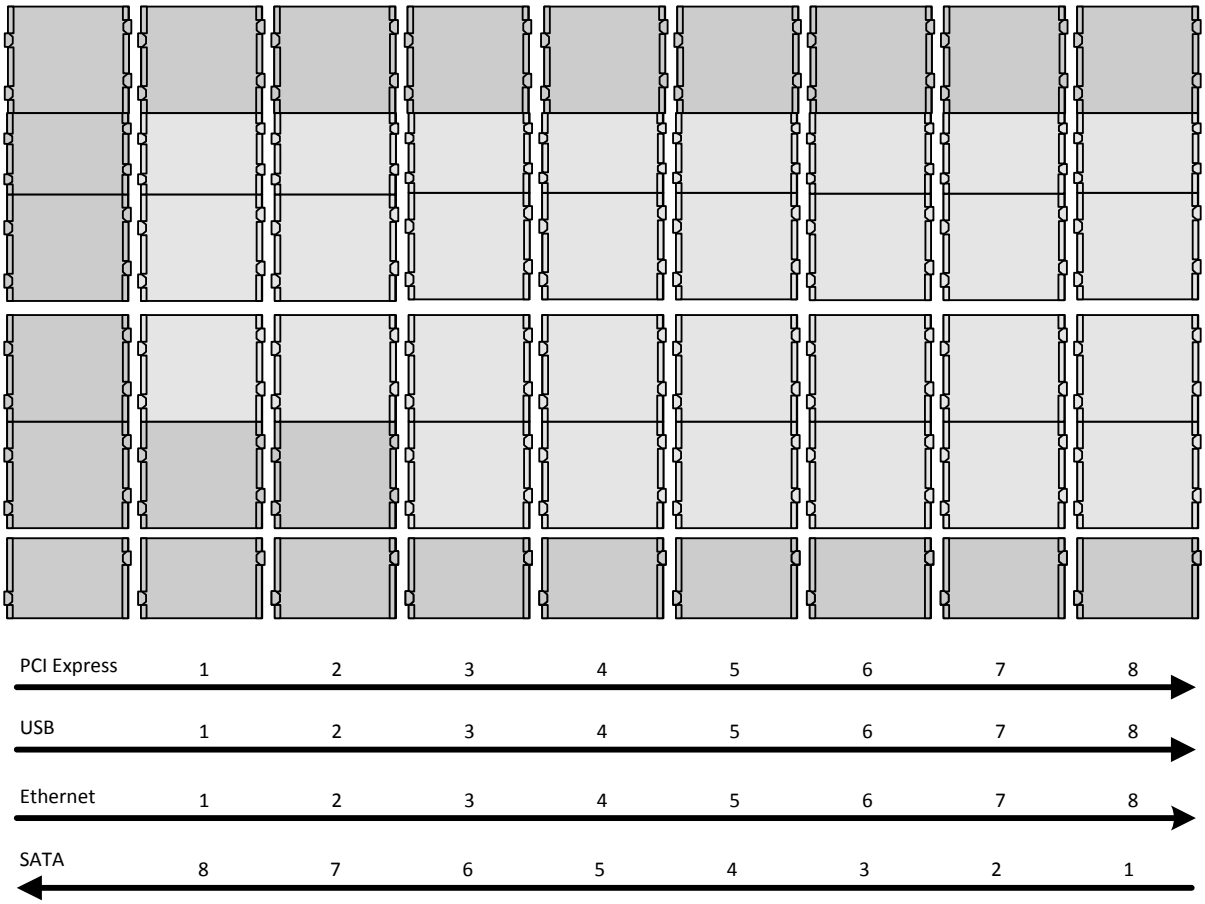
Refer to the CompactPCI Serial standard PICMG CPCI-S.0 for detailed information regarding the rear I/O connectors.

- CompactPCI Serial Specification PICMG CPCI-S.0 Revision 2.0: 2015; PCI Industrial Computers Manufacturers Group (PICMG) www.picmg.org
- Introduction to CompactPCI Serial on Wikipedia: en.wikipedia.org/wiki/CompactPCI_Serial

### 3.14.1 Backplane Filling Order

The CompactPCI Serial standard supports a maximum of 2 PCI Express x8 links (fat pipe), 6 PCI Express x4, 8 SATA, 8 USB and 8 Ethernet interfaces.

*Table 22. CompactPCI Serial backplane filling order*

| System Slot 1 | Fat Pipe Periph. Slot 2 | Fat Pipe Periph. Slot 3 | Periph. Slot 4 | Periph. Slot 5 | Periph. Slot 6 | Periph. Slot 7 | Periph. Slot 8 | Periph. Slot 9 |
|---|---|---|---|---|---|---|---|---|
|  | | | | | | | | |
| **Implementation on the G25A** | | | | | | | | |
| | PCI Express 3.0 x8 | PCI Express 3.0 x8 | PCI Express 3.0 x4 | PCI Express 3.0 x4 | PCI Express 2.0 x1 | PCI Express 2.0 x1 | PCI Express 2.0 x1 | PCI Express 2.0 x1 |
| | USB 3.0 | USB 3.0 | | | | | | |
| | 1GBASE-T Ethernet | 1GBASE-T Ethernet | 1GBASE-T Ethernet | 1GBASE-T Ethernet | | | | |
| | | | SATA | SATA | SATA | SATA | SATA | SATA |

### 3.14.2 Using the G25A as a Peripheral Board

It is possible to use more than one G25A board within a CPCI-S.0 system to build a redundant system or a cluster with more processing power. The communication between the boards is done via Ethernet in this case and the other high-speed interfaces cannot be used. The G25A cannot be booted via SATA in such a configuration.

In peripheral mode the following interfaces available on the CPCI-S.0 connector are disabled by the board firmware automatically:

- PCI Express
- USB
- SATA
- SMBus
- SGPIO
- All control signals which are only available for system boards

The board firmware detects if the board is inserted in a peripheral slot by monitoring the *SYS_EN#* pin on CompactPCI Serial connector P1.

# 4 UEFI Firmware

The G25A is equipped with the UEFI-based Aptio firmware from AMI. For more user-friendliness it has been modified by MEN.

The firmware settings can be entered using a setup menu.

## 4.1 Accessing the Firmware

Carry out the following steps to enter the setup menu:

» Power up the system.

» Wait until the following message appears on the screen:

```
Press <DEL> or <ESC> to enter setup
Press <F7> to enter Boot Menu
```

» Press the <Esc> or <DEL>[1] key.

» A dialog appears for entering the password, if passwords have been set.

» Enter the user or the administrator password.

See Chapter 4.3.5 Security Menu on page 56.

» The setup menu appears. You can now navigate through the tabs and menus via the keyboard.

## 4.2 Setup Menus

The UEFI firmware has 7 menus which can be selected via the tabs at the top of the screen.

On the right side of the screen, you can find information regarding the functions of the function keys and the selected menu item.

In the following, only the MEN-specific MEN menu and the most important features in the other tabs are described. Default values are printed in bold type in the following tables.

## 4.3 Setup Modes

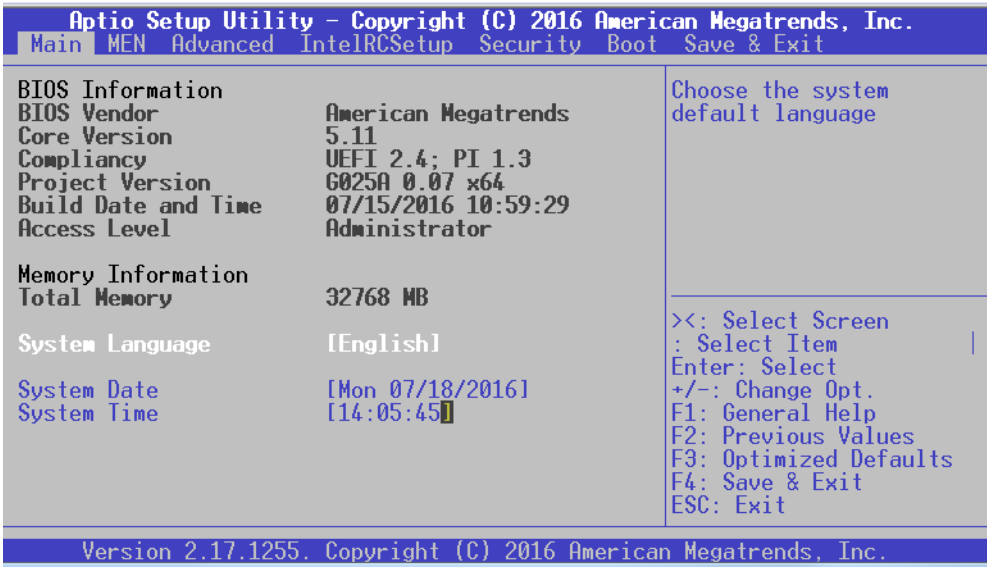
There are two setup modes for both the user and the administrator profile. Using the standard mode, menu items can be hidden which normally do not have to be modified. The mode can be switched under the MEN tab.

See Chapter 4.3.2 MEN Menu on page 52.

[1] When accessing the G25A via a serial console only the <Esc> key can be used.

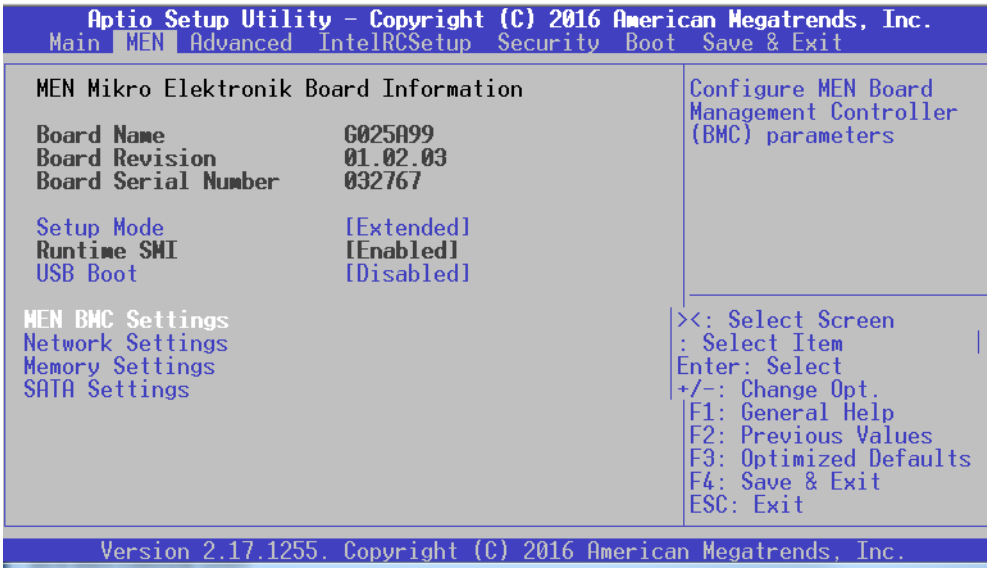
### 4.3.1 Main Menu

In the Main menu you can look up system parameters and set system language, date and time.




### 4.3.2 MEN Menu

The MEN menu has been adapted in order to simplify access to important functions. You can find information regarding board name, board revision and serial number as well as several sub-menus.



#### 4.3.2.1 MEN Menu – Sub-Functions

*Table 23. Functions in the MEN Menu*

| Function | Options |
|---|---|
| Setup Mode | **Standard**<br>Extended |
| Runtime SMI | **Enabled**<br>Disabled |
| USB Boot | **Enabled**<br>Disabled |

#### 4.3.2.2 MEN Menu – Sub-Menu BMC Settings

In the sub-menu BMC Settings you can find board information that is read out via the Board Management Controller (BMC), as well as the following sub-functions:

*Table 24. Sub-menu BMC Settings*

| Sub-menu | Function | Options |
|---|---|---|
| MEN BMC Settings | WatchDog | **Disabled**<br>1 min<br>2 min<br>5 min<br>10 min<br>15 min<br>20 min<br>30 min |
| | Power Resume Mode | **On**<br>Off<br>Former State |
| | EXT_PWRGD Mode | **Check at Start-Up only**<br>Check always |
| | EXT_PS_ON Mode | **Always on**<br>Switched |
| | RESET_IN | **Enabled**<br>Blocked |
| | MEN BMC Registers | For information: displays values from the BMC, e.g. error information |

#### 4.3.2.3 MEN Menu – Sub-Menu Network Settings

In the sub-menu Network Settings you can activate and deactivate the Ethernet controllers and the PXE boot functionality.

*Table 25. Sub-Menu Network Settings*

| Sub-menu | Function | Options |
|---|---|---|
| MEN Network Settings | Front X1 - 1x 1GbE | |
| | Controller (I218) | **Enabled**<br>Disabled |
| | PXE Boot | Enabled<br>**Disabled** |
| | Front X2/3 - 2x 10GbE | |
| | Controller (X557) | **Enabled**<br>Disabled |
| | PXE Boot | Enabled<br>**Disabled** |
| | Backplane - 4x 1GbE | |
| | Controller (I350) | **Enabled**<br>Disabled |
| | PXE Boot | Enabled<br>**Disabled** |

#### 4.3.2.4 MEN Menu – Sub-Menu Memory Settings

*Table 26. Sub-Menu Memory Settings*

| Sub-menu | Function | Options |
|---|---|---|
| MEN Memory Settings | ECC Support | **Auto**<br>Disabled<br>Enabled |
| | Refresh Rate | **Regular**<br>Custom |
| | Self Refresh Mode | **Acc Self Refresh**<br>2x Self Refresh |

#### 4.3.2.5 MEN Menu – Sub-Menu SATA Settings

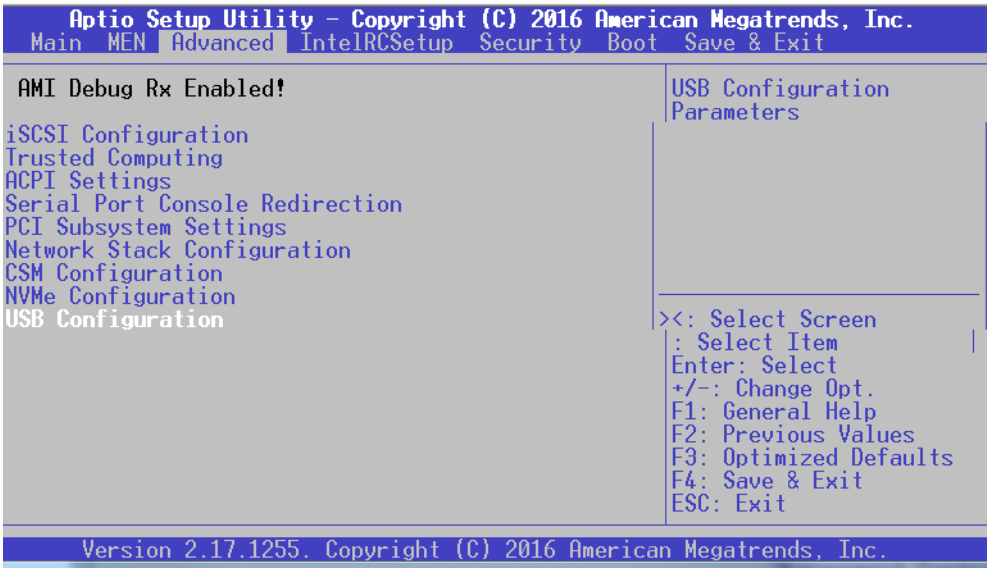
In the sub-menu SATA Settings you can set SATA speed and enable the SGPIO interface.

*Table 27. Sub-Menu SATA Settings*

| Sub-menu | Function | Options |
|---|---|---|
| MEN SATA Settings | SATA Speed Limit | **Unlimited**<br>Gen3<br>Gen2<br>Gen1 |
| | SATA Delay | **0** |
| | SGPIO Interface | **Enabled**<br>Disabled |

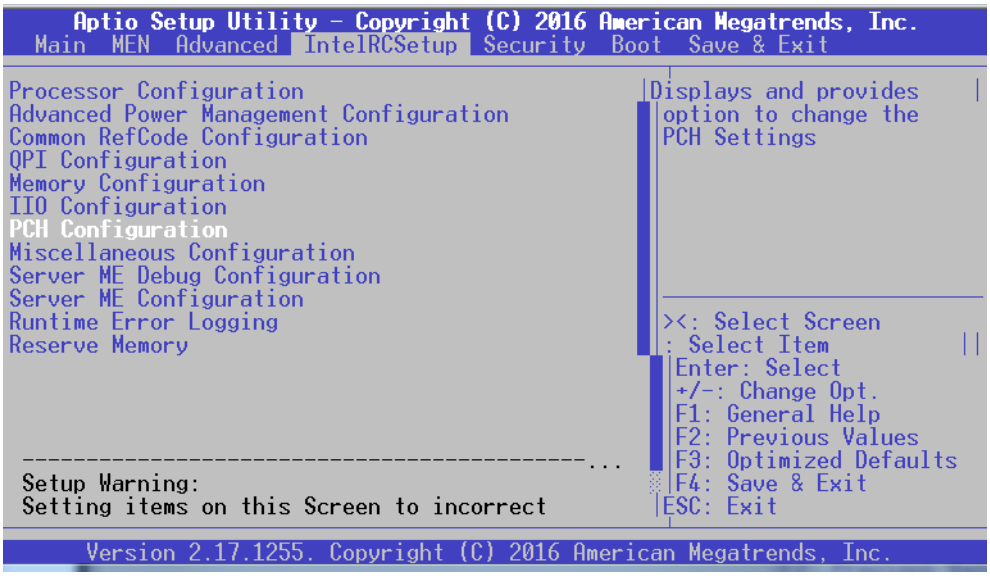
### 4.3.3 Advanced Menu

In the Advanced menu you can change settings regarding Trusted Computing, Console Redirection or for configuring the Network Stack.



### 4.3.4 IntelRCSetup Menu

Most of the functions in this menu normally do not have to be changed during operation. If you use a RAID in your system, however, some settings have to be made here..




Setting items on this screen to incorrect values can lead to malfunction of the system.

### 4.3.5 Security Menu

In the Security menu you can enter passwords for the user profiles User and Administrator.

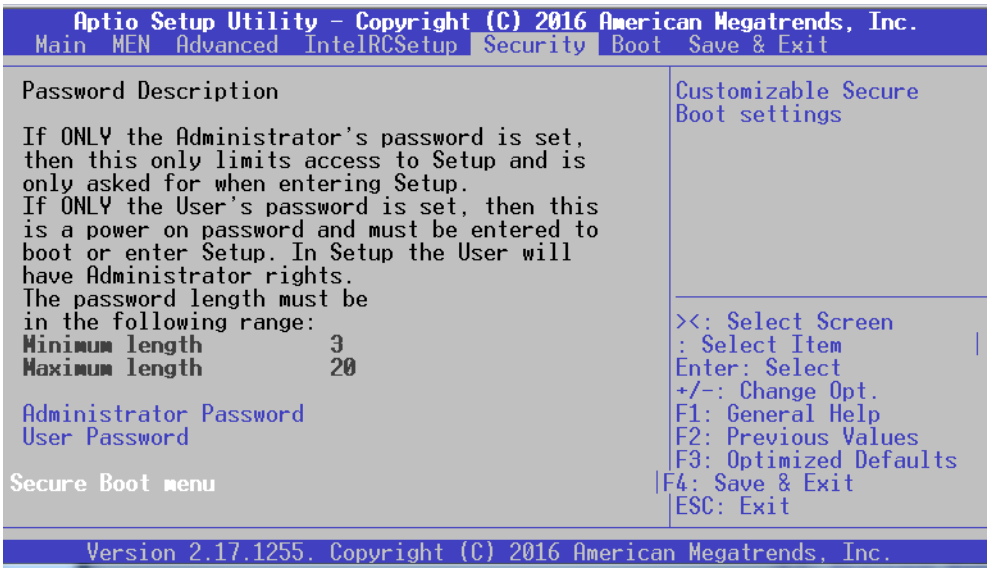


*Table 28. Setting passwords*

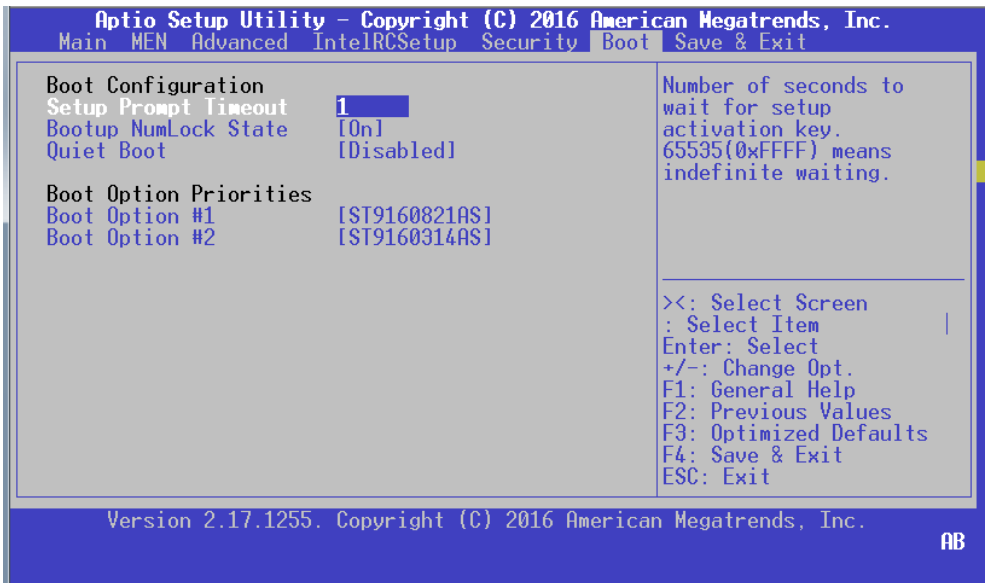
| Function | Description |
|---|---|
| Administrator Password | Sets, changes or deletes the administrator password. If an administrator password is set the systems asks for this first. For deleting the password enter nothing and press the enter key. For entering a new password enter it twice and press the enter key. |
| User Password | Sets, changes or deletes the user password. If a user password is set the system asks for this first. For deleting the password enter nothing and press the enter key. For entering a new password enter it twice and press the enter key. |

*Table 29. Security modes*

| Setting | Description |
|---|---|
| No password is set | Booting the system and opening the UEFI setup is not protected. |
| Only administrator password is set | Booting the system is not protected. For opening the UEFI setup, the administrator password is required. |
| Only user password is set | Booting the system is not protected. The user password is required for opening the UEFI setup. The user password has to be entered at every system start. |
| Administrator and user password is set | Booting the system is not protected. The administrator or the user password is required for opening the UEFI setup. The user password allows only limited access to the setup. The administrator password allows full access to the setup. |

### 4.3.6 Boot Menu

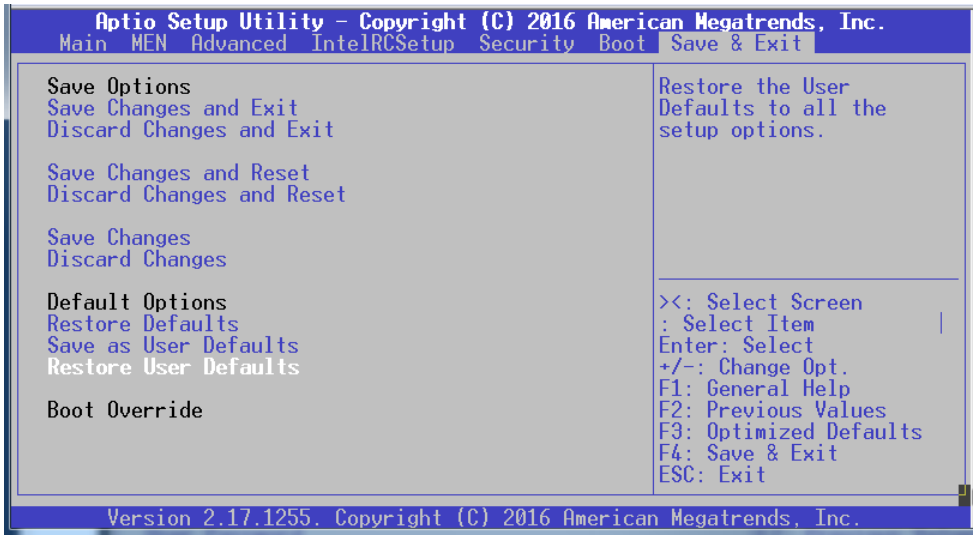
In the boot menu you can make settings regarding boot behavior and boot order.




### 4.3.7 Save and Exit Menu

In the Save and Exit menu you can set how to handle changes made within the setup and exit the setup.

Using the *Boot Override* function you can boot from a medium different from the one set in the boot order.



# 5 Hardware/Software Interface

This chapter is intended for software developers or board integrators who need deeper knowledge of the implementation details of the G25A interfaces and its internal connections.

## 5.1 PCI Express Root Port Interrupt Mapping

*Table 30. PCI Express Root Port Interrupt Mapping for Downstream Devices*

| Port | INTA# | INTB# | INTC# | INTD# |
|---|---|---|---|---|
| 1 | INTA# | INTB# | INTC# | INTD# |
| 2 | INTB# | INTC# | INTD# | INTA# |
| 3 | INTC# | INTD# | INTA# | INTB# |
| 4 | INTD# | INTA# | INTB# | INTC# |
| 5 | INTA# | INTB# | INTC# | INTD# |
| 6 | INTB# | INTC# | INTD# | INTA# |
| 7 | INTC# | INTD# | INTA# | INTB# |
| 8 | INTD# | INTA# | INTB# | INTC# |

## 5.2 SMBus/I2C Devices

*Table 31. SMBus/I2C devices*

| 8-Bit Address | 7-Bit Address | Function | MDIS Device Name |
|---|---|---|---|
| 0x9A | 0x4D | Board Management Controller (BMC) | *xm01bc_1* |
| 0x3E | 0x1F | Board temperature sensor | - |
| 0x64 | 0x32 | System RTC (ERTC) RX-8571, see also Chapter 3.5 Real-Time Clock (RTC) on page 36 | - |
| 0x6E | 0x37 | EEPROM: protected register | - |
| 0x88 | 0x44 | Intel Management Engine (ME) | - |
| 0xAE | 0x57 | Board information EEPROM with thermal sensor | *smb2_2* |
| 0xE0 | 0x70 | SATA SGPIO FPGA, see Chapter 5.3 SATA SGPIO FPGA on page 60 | *smb2_2* |

***Note on 8-Bit/7-Bit Addressing***

- **8-bit addressing** is compliant to the Windows nomenclature. The last bit, which is used as the read/write bit, is added to the address (0 = write, 1 = read).
  If you use MDIS driver software, use 8-bit addresses, with any OS.
- **7-bit addressing** is used, e.g., under Linux. A '0' is added at the beginning of the address so that all consecutive address bits are moved one bit to the right.
  If you use standard I2C commands under Linux, use 7-bit addresses.

## 5.3 SATA SGPIO FPGA

*Table 32. SMBus/IC2 – SATA SGPIO FPGA – address map*

| Address Offset | Register |
|---|---|
| 0x00 | SATA0 Register (SATA drive 0) |
| 0x01 | SATA1 Register (SATA drive 1) |
| 0x02 | SATA2 Register (SATA drive 2) |
| 0x03 | SATA3 Register (SATA drive 3) |
| 0x04 | SATA4 Register (SATA drive 4) |
| 0x05 | SATA5 Register (SATA drive 5) |
| 0x06 | SATA6 Register (SATA drive 6) |
| 0x07 | SATA7 Register (SATA drive 7) |
| 0xFE | Control Register |
| 0xFF | Handle State Register |

*Table 33. SMBus/IC2 – SATA SGPIO FPGA – SATA0 to SATA7 Register (0x00 to 0x07)*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | *LED_ACT* | *LED_LOC* | *LED_HOT* | - | | | | |
| Access | R | R/W | R/W | - | | | | |
| Reset | 0 | 0 | 0 | 00000 | | | | |

| Bit Field | | Description |
|---|---|---|
| 7 | *LED_ACT* | Activity LED of respective SATA drive<br>▪ 0: LED off<br>▪ 1: LED on |
| 6 | *LED_LOC* | Locate LED of respective SATA drive<br>▪ 0: LED off<br>▪ 1: LED on |
| 5 | *LED_HOT* | Hot-plug LED of respective SATA drive<br>▪ 0: LED off<br>▪ 1: LED on |
| 4:0 | Reserved | |

*Table 34. SMBus/IC2 – SATA SGPIO FPGA – Control Register (0xFE)*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | *SATA_ DETECT* | *SMBALERT _EN* | *SMBALERT* | - | | | | |
| Access | R/W | R/W | R/W | - | | | | |
| Reset | 1 | 0 | 0 | 00000 | | | | |

| Bit Field | | Description |
|---|---|---|
| **7** | *SATA_DETECT* | Enable or disable passing output pins HANDLE_STATE to the CPU<br>▪ 0: Disable all HANDLE_STATE pins<br>▪ 1: Enable HANDLE_STATE pins |
| **6** | *SMBALERT_EN* | Enable or disable passing output interrupt *I2C_IRQ_N* to the CPU<br>▪ 0: Disable passing SMBALERT interrupt<br>▪ 1: Enable passing SMBALERT interrupt |
| **5** | *SMBALERT* | **Read access:**<br>SMBALERT interrupt bit<br>▪ 0: SMBALERT interrupt is inactive<br>▪ 1: SMBALERT interrupt is active<br>**Write access:**<br>SMBALERT interrupt acknowledge bit<br>▪ 0: Ignored<br>▪ 1: Acknowledge SMBALERT interrupt |
| **4:0** | Reserved | |

*Table 35. SMBus/IC2 – SATA SGPIO FPGA – Handle State Register (0xFF)*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| **Name** | *HANDLE_STATE7* | *HANDLE_STATE6* | *HANDLE_STATE5* | *HANDLE_STATE4* | *HANDLE_STATE3* | *HANDLE_STATE2* | *HANDLE_STATE1* | *HANDLE_STATE0* |
| **Access** | R | R | R | R | R | R | R | R |
| **Reset** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit Field | | Description |
|---|---|---|
| **7:0** | *HANDLE_STATE [7:0]* | Upstream bits from the connected SATA drive<br>▪ 0: SATA drive in slot x connected (handle switch closed)<br>▪ 1: SATA drive in slot x **not** connected (handle switch open) |

## 5.4 BMC API (Application Programming Interface)

The G25A uses a generic command interface for communication between the CPU (host) and the BMC. Application software uses command packets to communicate with the BMC.

The application software controls the BMC via I2C/SMBus. The device address is `0x4D` (in 7-bit, non-shifted notation) or `0x9A`/`0x9B` (in 8-bit, shifted notation, write/read).

### 5.4.1 BMC Command Packets

#### 5.4.1.1 Command Packet Protocol

From a logical point of view, the command protocol has the following characteristics:

- Commands are always initiated by the host. The BMC never sends packets without the host requesting it to do so.
- Packets are either
  - unidirectional from host to BMC, without an answer from the BMC
  - bidirectional, with an answer from the BMC
- Each command has a unique identifier, consisting of the command opcode and a packet type:

*Table 36. BMC API – Packet types*

| Packet Type | Description | Request Data Host > BMC | Response Data BMC > Host | Error Signaling |
|---|---|---|---|---|
| *PT_SB* | Send command only | None | No response | - |
| *PT_RBD* | Send command, get one data byte from BMC | None | 1 byte | Response byte = `0xFF` |
| *PT_WBD* | Send command, send one data byte to BMC | 1 byte | No response | - |
| *PT_RWD* | Send command, get two data bytes from BMC | None | 2 bytes | Response byte = `0xFFFF` |
| *PT_WWD* | Send command, send two data bytes to BMC | 2 bytes | No response | - |

The packet types are directly mapped to the corresponding SMBus “bus protocols” as defined in the System Management Bus Specification.

*Table 37. BMC API – Packet types mapping on SMBus*

| Packet Type | SMBus Protocol |
|---|---|
| *PT_SB* | Send byte |
| *PT_RBD* | Read byte |
| *PT_WBD* | Write byte |
| *PT_RWD* | Read word |
| *PT_WWD* | Write word |

#### 5.4.1.2 Watchdog Control Commands

*Table 38. BMC API – Watchdog commands*

| Command | Packet Type | Opcode | Functional Description |
|---|---|---|---|
| *WDOG_ON* | *PT_SB* | 0x11 | Enable watchdog |
| *WDOG_OFF* | *PT_WBD* | 0x12 | Disable watchdog |
| *WDOG_TRIG* | *PT_SB* | 0x13 | Trigger watchdog |
| *WDOG_TIME_SET* | *PT_WWD* | 0x14 | Set watchdog timeout value |
| *WDOG_TIME_GET* | *PT_RWD* | 0x14 | Get watchdog timeout value |
| *WDOG_STATE_GET* | *PT_RBD* | 0x17 | Get watchdog state |
| *WDOG_ARM* | *PT_SB* | 0x18 | Arm watchdog and BIOS timeouts |
| *ARM_STATE* | *PT_RBD* | 0x19 | Get watchdog arming state |

***Command WDOG_ON***

| **Opcode:** 0x11 | **Packet Type:** ***PT_SB*** |
|---|---|

***Command WDOG_OFF***

| **Opcode:** 0x12 | | | | | | | | **Packet Type:** ***PT_WBD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | 0x69 | | | | | | | |

***Command WDOG_TRIG***

| **Opcode:** 0x13 | **Packet Type:** ***PT_SB*** |
|---|---|

### *Commands WDOG_TIME_SET and WDOG_TIME_GET*

*Command WDOG_TIME_SET*

| **Opcode: 0x14** | | | | | | | | **Packet Type: *PT_WWD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data 0** | *WD_TOUT* (LSB) | | | | | | | |
| **Data 1** | *WD_TOUT* (MSB) | | | | | | | |

*Command WDOG_TIME_GET*

| **Opcode: 0x14** | | | | | | | | **Packet Type: *PT_RWD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data 0** | *WD_TOUT* (LSB) | | | | | | | |
| **Data 1** | *WD_TOUT* (MSB) | | | | | | | |

| **Bit Field** | **Description** |
|---|---|
| *WD_TOUT* | Trigger timeout, in steps of 100 ms<br>▪ 0x0001: 100 ms<br>▪ 0x0002: 200 ms<br>▪ ...<br>▪ 0xFFFF: Error |

### *Command WDOG_STATE_GET*

| **Opcode: 0x17** | | | | | | | | **Packet Type: *PT_RBD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | *WD_STATE* | | | | | | | |

| **Bit Field** | **Description** |
|---|---|
| *WD_STATE* | Watchdog state<br>▪ 0x00: Off<br>▪ 0x01: On<br>▪ 0xFF: Error |

***Command WDOG_ARM***

| Opcode: 0x18 | Packet Type: *PT_SB* |
|---|---|

***Command WDOG_ARM_STATE***

| Opcode: 0x19 | | | | | | | | Packet Type: *PT_RBD* |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | *ARM_STATE* | | | | | | | |

| Bit Field | Description |
|---|---|
| *ARM_STATE* | Watchdog arming state<br>▪ 0x00: Not armed<br>▪ 0x01: Armed<br>▪ 0xFF: Error |

### 5.4.1.3 Power Resume Mode Commands

These commands allow configuring the behavior of the G25A in case the power is reapplied after a power failure and input voltages return to their allowed limits.

This setting is only used when the G25A is in Master mode. When the G25A is in Slave mode, the BMC always starts the power-up sequence.

See Chapter 5.4.1.12 Board Controller Mode on page 77.

The setting is persistent, i.e. it is stored in non-volatile memory.

The default resume mode after factory programming is "On".

*Table 39. BMC API – Power resume mode commands*

| Command | Packet Type | Opcode | Functional Description |
|---|---|---|---|
| *RESUME_MODE_SET* | *PT_WBD* | 0x20 | Set power resume mode |
| *RESUME_MODE_GET* | *PT_RBD* | 0x20 | Get power resume mode |

*Table 40. BMC API – Power resume modes*

| Resume Mode | System State at Power Loss | Resume Action |
|---|---|---|
| On | On | Start power-up sequence |
| | Off | Start power-up sequence |
| Off | On | Stay in S4/S5 state |
| | Off | Stay in S4/S5 state |
| Former | On | Start power-up sequence |
| | Off | Stay in S4/S5 state |

S0 to S5 are the power states as defined in the ACPI specification, or an equivalent state

Please refer to the ACPI Specification for more details on the power states S0 to S5:
Advanced Configuration and Power Interface Specification Version 6.1
January, 2016
Unified EFI Forum
uefi.org/specifications

#### *Commands RESUME_MODE_SET and RESUME_MODE_GET*

*Command RESUME_MODE_SET*

| **Opcode: 0x20** | | | | | | | | **Packet Type: *PT_WBD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | *RES_MODE* | | | | | | | |

*Command RESUME_MODE_GET*

| **Opcode: 0x20** | | | | | | | | **Packet Type: *PT_RBD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | *RES_MODE* | | | | | | | |

| Bit Field | Description |
|---|---|
| *RES_MODE* | Resume mode<br>▪ 0x00: Off<br>▪ 0x01: On<br>▪ 0x02: Former |

#### 5.4.1.4 External Power Supply Failure Mode

These commands allow configuring the behavior of the G25A upon assertion of an external power supply fail signal.

Modes:

- Ignore: Assertion of external power failure signal is completely ignored.
- Treat as error: Assertion of external power failure is treated as an error; i.e. event is counted as an error and G25A is reset.

The setting is persistent, i.e. it is stored in non-volatile memory.

The default external power supply fail signal mode after factory programming is "Ignore".

*Table 41. BMC API – External power supply failure mode commands*

| Command | Packet Type | Opcode | Functional Description |
|---|---|---|---|
| *EXT_PWR_FAIL_MODE_SET* | *PT_WBD* | `0x21` | Set external power supply failure mode |
| *EXT_PWR_FAIL_MODE_GET* | *PT_RBD* | `0x21` | Get external power supply failure mode |

***Commands EXT_PWR_FAIL_MODE_SET and EXT_PWR_FAIL_MODE_GET***

*Command EXT_PWR_FAIL_MODE_SET*

| **Opcode:** `0x21` | | | | | | | | **Packet Type:** ***PT_WBD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | *EXT_PWR_FAIL_MODE* | | | | | | | |

*Command EXT_PWR_FAIL_MODE_GET*

| **Opcode:** `0x21` | | | | | | | | **Packet Type:** ***PT_RBD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | *EXT_PWR_FAIL_MODE* | | | | | | | |

| Bit Field | Description |
|---|---|
| *EXT_PWR_FAIL_MODE* | External power supply failure mode<br>▪ `0x00`: Ignore<br>▪ `0x01`: Treat as error<br>▪ `0xFF`: Error |

#### 5.4.1.5 Reset Signal Blocking

These commands allow blocking of G25A reset inputs. The setting is persistent, i.e. it is stored in non-volatile memory.

In a system with master and slave CPU boards, normally the slave boards will get a reset whenever the master board resets. With "Reset Signal Blocking" configuration it is possible to decide at runtime for the slave boards whether they should get a reset whenever the master board resets or whether the slave board should operate independently. Additionally with this functionality it is possible to disable external reset for the master board where needed.

The default mode after factory programming is "Reset enabled".

*Table 42. BMC API – Reset signal blocking commands*

| Command | Packet Type | Opcode | Functional Description |
|---|---|---|---|
| *RESET_IN_MODE_SET* | *PT_WBD* | 0x22 | Set reset input mode |
| *RESET_IN_MODE_GET* | *PT_RBD* | 0x22 | Get reset input mode |

***Commands RESET_IN_MODE_SET and RESET_IN_MODE_GET***

*Command RESET_IN_MODE_SET*

| **Opcode: 0x22** | | | | | | | | **Packet Type: *PT_WBD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | *RESET_IN_MODE* | | | | | | | |

*Command RESET_IN_MODE_GET*

| **Opcode: 0x22** | | | | | | | | **Packet Type: *PT_RBD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | *RESET_IN_MODE* | | | | | | | |

| Bit Field | Description |
|---|---|
| *RESET_IN_MODE* | Reset input mode<br>▪ 0x00: Reset enabled<br>▪ 0x01: Reset masked<br>▪ 0xFF: Error |

#### 5.4.1.6 External Power Supply Control

In Master mode, the BMC uses the EXT_PS_ON signal to switch the external power supply on and off. In Slave mode, the BMC does not control the EXT_PS_ON signal.

*Table 43. BMC API – External power supply control commands*

| Command | Packet Type | Opcode | Functional Description |
|---|---|---|---|
| *EXT_PS_ON_MODE_SET* | *PT_WBD* | 0x23 | Set *EXT_PS_ON* mode |
| *EXT_PS_ON_MODE_GET* | *PT_RBD* | 0x23 | Get *EXT_PS_ON* mode |

***Commands EXT_PS_ON_MODE_SET and EXT_PS_ON_MODE_GET***

*Command EXT_PS_ON_MODE_SET*

| **Opcode:** 0x23 | | | | | | | | **Packet Type:** ***PT_WBD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | *EXT_PS_ON_MODE* | | | | | | | |

*Command EXT_PS_ON_MODE_GET*

| **Opcode:** 0x23 | | | | | | | | **Packet Type:** ***PT_RBD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | *EXT_PS_ON_MODE* | | | | | | | |
| *EXT_PS_ON_MODE* | External power supply on/off mode<br>▪ 0x00: Invalid<br>▪ 0x01: Always<br>▪ 0x02: Switched<br>▪ 0xFF: Error | | | | | | | |

#### 5.4.1.7 Software Reset

These commands allow performing CPU resets under application software control.

Different types of resets are available:

- *SW_RESET* issues a “warm reset”.
- *SW_COLD_RESET* issues a “cold reset”.
- *SW_RTC_RESET* issues a “cold reset”, together with an RTC reset.

Resets will be performed by writing the data word `0xDEAD`, see below.

*Table 44. BMC API – Software reset commands*

| Command | Packet Type | Opcode | Functional Description |
|---|---|---|---|
| *SW_RESET* | *PT_WWD* | `0x31` | Initiate software reset (warm reset) |
| *SW_COLD_RESET* | *PT_WWD* | `0x32` | Initiate cold reset |
| *SW_RTC_RESET* | *PT_WWD* | `0x35` | Initiate cold reset combined with RTC reset |

***Command SW_RESET***

| **Opcode:** `0x31` | | | | | | | | **Packet Type:** ***PT_WWD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data 0** | `0xAD` | | | | | | | |
| **Data 1** | `0xDE` | | | | | | | |

***Command SW_COLD_RESET***

| **Opcode:** `0x32` | | | | | | | | **Packet Type:** ***PT_WWD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data 0** | `0xAD` | | | | | | | |
| **Data 1** | `0xDE` | | | | | | | |

***Command SW_RTC_RESET***

| **Opcode:** `0x35` | | | | | | | | **Packet Type:** ***PT_WWD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data 0** | `0xAD` | | | | | | | |
| **Data 1** | `0xDE` | | | | | | | |

#### 5.4.1.8 Power Button

These commands allow initiating power button events.

*Table 45. BMC API – Power button commands*

| Command | Packet Type | Opcode | Functional Description |
|---|---|---|---|
| *PWRBTN* | *PT_WBD* | 0x33 | Perform pressing of power button |
| *PWRBTN_OVRD* | *PT_WBD* | 0x34 | Perform power button override, i.e. assert the power button for more than 4 seconds to initiate system shutdown |

***Command PWRBTN***

| **Opcode:** 0x33 | | | | | | | | **Packet Type:** ***PT_WBD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | 0x69 | | | | | | | |

***Command PWRBTN_OVRD***

| **Opcode:** 0x34 | | | | | | | | **Packet Type:** ***PT_WBD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | 0x69 | | | | | | | |

#### 5.4.1.9 Voltage Supervision

The voltage supervision commands the customer application to monitor different voltages on the G25A.

*Table 46. BMC API – Voltage supervision commands*

| Command | Packet Type | Opcode | Functional Description |
|---|---|---|---|
| *VOLT_LOW(0)* | *PT_RWD* | 0x40 | Get lower limit of +3.3 V (in mV) |
| *VOLT_LOW(1)* | *PT_RWD* | 0x41 | Get lower limit of +5 V (in mV) |
| *VOLT_LOW(2)* | *PT_RWD* | 0x42 | Get lower limit of +12 V (in mV) |
| *VOLT_LOW(3)* | *PT_RWD* | 0x43 | Get lower limit of +5 V standby (in mV) |
| *VOLT_LOW(4)* | *PT_RWD* | 0x44 | Get lower limit of battery voltage (in mV) |
| *VOLT_HIGH(0)* | *PT_RWD* | 0x50 | Get upper limit of +3.3 V (in mV) |
| *VOLT_HIGH(1)* | *PT_RWD* | 0x51 | Get upper limit of +5 V (in mV) |
| *VOLT_HIGH(2)* | *PT_RWD* | 0x52 | Get upper limit of +12 V (in mV) |
| *VOLT_HIGH(3)* | *PT_RWD* | 0x53 | Get upper limit of +5 V standby (in mV) |
| *VOLT_HIGH(4)* | *PT_RWD* | 0x54 | Get upper limit of battery voltage (in mV) |
| *VOLT_ACT(0)* | *PT_RWD* | 0x60 | Get actual value of +3.3 V (in mV) |
| *VOLT_ACT(1)* | *PT_RWD* | 0x61 | Get actual value of +5 V (in mV) |
| *VOLT_ACT(2)* | *PT_RWD* | 0x62 | Get actual value of +12 V (in mV) |
| *VOLT_ACT(3)* | *PT_RWD* | 0x63 | Get actual value of +5 V standby (in mV) |
| *VOLT_ACT(4)* | *PT_RWD* | 0x64 | Get actual value of battery voltage (in mV) |
| *NUM_VOLTS* | *PT_RBD* | 0x8E | Get number of supervised voltages |

***Command VOLT_LOW(x)***

| **Opcode: 0x40 + x** | | | | | | | | **Packet Type: *PT_RWD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data 0** | Lower limit of voltage x (LSB) | | | | | | | |
| **Data 1** | Lower limit of voltage x (MSB) | | | | | | | |

***Command VOLT_HIGH(x)***

| **Opcode: 0x50 + x** | | | | | | | | **Packet Type: *PT_RWD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data 0** | Upper limit of voltage x (LSB) | | | | | | | |
| **Data 1** | Upper limit of voltage x (MSB) | | | | | | | |

***Command VOLT_ACT(x)***

| **Opcode:** 0x60 + x | | | | | | | | **Packet Type:** *PT_RWD* |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data 0** | Actual value of voltage x (LSB) | | | | | | | |
| **Data 1** | Actual value of voltage x (MSB) | | | | | | | |

***Command NUM_VOLTS***

| **Opcode:** 0x8E | | | | | | | | **Packet Type:** *PT_RBD* |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | Number of supervised voltages | | | | | | | |

#### 5.4.1.10 Error Counters

The error counter commands allow querying and clearing error counters.

The BMC provides error counters for each type of error that can occur. Using this information, the application software can determine how often certain errors have occurred, but it is not possible to determine the chronological order of the errors.

You can determine the actual number of error counters using *NUM_ERR_CNTRS*, up to a theoretical maximum of 255 error counters.

All counters are set to zero during factory programming or using command *ERR_CNT_CLR*.

*Table 47. BMC API – Error counters*

| Counter | Error Condition / Error Clearing |
|---|---|
| 1 | External BMC watchdog timeout (application software timeout) |
| 2 | Internal BMC watchdog timeout |
| 3 | Internal brown-out (BMC undervoltage) |
| 4 | External power failure |
| 5 | BIOS life sign timeout |
| 6 | Processor too hot |
| 7 | Shutdown while too hot |
| 8 | Internal power failure |
| 9 | Handshake timeout |
| 10 | Platform reset timeout |
| 11 | Error cleared using system reset |
| 12 | Error cleared using power cycling |
| 13 | Error cleared using power cycling with resume reset |
| 14 | Error cleared using power cycling with RTC reset |
| 15 | Error could not be corrected |

*Table 48. BMC API – Error counter commands*

| Command | Packet Type | Opcode | Functional Description |
|---|---|---|---|
| *ERRCNT_01* | *PT_RBD* | `0x70` | Get error counter 1 |
| *ERRCNT_xx* | | `0x70` + x | Get error counter xx |
| *ERRCNT_15* | | `0x7E` | Get error counter 15 |
| *ERR_CNT_CLR* | *PT_WBD* | `0x7F` | Clear error counters |
| *NUM_ERR_CNTRS* | *PT_RBD* | `0x8D` | Get number of error counters |

***Command ERRCNT_xx (1 to 15)***

| **Opcode: `0x70` + x** | | | | | | | **Packet Type: *PT_RBD*** | |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | Value of error counter number xx | | | | | | | |

### *Command ERRCNT_xx (16 to 32)*

| **Opcode:** 0xB0 + x | | | | | | | | **Packet Type:** *PT_RBD* |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | Value of error counter number xx | | | | | | | |

### *Command ERR_CNT_CLR*

This command clears all error counters.

| **Opcode:** 0x7F | | | | | | | | **Packet Type:** *PT_WBD* |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | 0x69 | | | | | | | |

### *Command NUM_ERR_CNTRS*

| **Opcode:** 0x8D | | | | | | | | **Packet Type:** *PT_RBD* |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | Number of error counters | | | | | | | |

#### 5.4.1.11 Firmware Revision

The firmware revision commands allow querying the separate parts of the BMC firmware revision.

*Table 49. BMC API – Firmware version commands*

| Command | Packet Type | Opcode | Functional Description |
|---|---|---|---|
| *GETREV_WORD0* | *PT_RWD* | 0x80 | Get firmware revision major part |
| *GETREV_WORD1* | *PT_RWD* | 0x81 | Get firmware revision minor part |
| *GETREV_WORD2* | *PT_RWD* | 0x82 | Get firmware revision maintenance part |
| *GETREV_WORD3* | *PT_RWD* | 0x83 | Get firmware revision build part |
| *GETREV_WORD4* | *PT_RWD* | 0x84 | Get firmware revision verification marker |

***Command GETREV_WORD0***

| **Opcode: 0x80** | | | | | | | | **Packet Type: *PT_RWD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data 0** | Firmware Revision Major Part (LSB) | | | | | | | |
| **Data 1** | Firmware Revision Major Part (MSB) | | | | | | | |

***Command GETREV_WORD1***

| **Opcode: 0x81** | | | | | | | | **Packet Type: *PT_RWD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data 0** | Firmware Revision Minor Part (LSB) | | | | | | | |
| **Data 1** | Firmware Revision Minor Part (MSB) | | | | | | | |

***Command GETREV_WORD2***

| **Opcode: 0x82** | | | | | | | | **Packet Type: *PT_RWD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data 0** | Firmware Revision Maintenance Part (LSB) | | | | | | | |
| **Data 1** | Firmware Revision Maintenance Part (MSB) | | | | | | | |

***Command GETREV_WORD3***

| **Opcode: 0x83** | | | | | | | | **Packet Type: *PT_RWD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data 0** | Firmware Revision Build Part (LSB) | | | | | | | |
| **Data 1** | Firmware Revision Build Part (MSB) | | | | | | | |

***Command GETREV_WORD4***

| Opcode: 0x84 | | | | | | | | Packet Type: *PT_RWD* |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data 0** | Firmware Revision Verification Marker (LSB) | | | | | | | |
| **Data 1** | Firmware Revision Verification Marker (MSB) | | | | | | | |

### 5.4.1.12 Board Controller Mode

This command allows determining if the CPU is operated as a master or slave.

*Table 50. BMC API – Board controller mode command*

| Command | Packet Type | Opcode | Functional Description |
|---|---|---|---|
| *BOARD_MODE* | *PT_RBD* | 0x8B | Get board controller mode (Master/ Slave) |

***Command BOARD_MODE***

| Opcode: 0x8B | | | | | | | | Packet Type: *PT_RBD* |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | *BOARD_CTRL_MODE* | | | | | | | |

| Bit Field | Description |
|---|---|
| *BOARD_CTRL_MODE* | Board controller mode<br>▪ 0x00: Invalid<br>▪ 0x01: Master<br>▪ 0x02: Slave<br>▪ 0xFF: Error |

#### 5.4.1.13 Geographical Address

*Table 51. BMC API – Backplane slot geographical address command*

| Command | Packet Type | Opcode | Functional Description |
|---|---|---|---|
| *CPCI_SLOT_ADDRESS* | *PT_RBD* | 0x8C | Get CompactPCI peripheral slot address |

***Command SLOT_ADDRESS***

| **Opcode: 0x8C** | | | | | | | | **Packet Type: *PT_RBD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | *SLOT_ADDRESS* | | | | | | | |

| Bit Field | Description |
|---|---|
| *SLOT_ADDRESS* | CompactPCI backplane slot geographical board address<br>▪ 0x00 – 0x07: Information from GA[2:0] backplane pins<br>▪ 0xFF: Error |

#### 5.4.1.14 Hardware Board Type

This command allows the BMC to query the board type, i.e. a unique ID that MEN assigns to each hardware board the generic BMC is implemented on. The board type is programmed into the BMC during production. The setting is persistent, i.e. is stored in a non-volatile memory.

***Command HW_BOARD_GET***

| **Opcode: 0x8F** | | | | | | | | **Packet Type: *PT_RWD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data 0** | *BOARD* (LSB) | | | | | | | |
| **Data 1** | *BOARD* (MSB) | | | | | | | |

| Bit Field | Description |
|---|---|
| *BOARD* | Unique MEN board ID |

#### 5.4.1.15 Last Error

This command allows querying the last error.

*Table 52. BMC API – Last error command*

| Command | Packet Type | Opcode | Functional Description |
|---|---|---|---|
| *ERR_LAST* | *PT_RBD* | 0x90 | Get last error |

***Command ERR_LAST***

| **Opcode: 0x90** | | | | | | | | **Packet Type: *PT_RBD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | *LAST_ERR_CODE* | | | | | | | |

| Bit Field | Description |
|---|---|
| *LAST_ERR_CODE* | Last error<br>▪ 0x00: Initial value; no error was registered by the BMC since the Last Error Register was cleared<br>▪ 0x01: +3.3 V voltage failure<br>▪ 0x02: Input voltage failure<br>▪ 0x03: External power supply failure<br>▪ 0x04: CPU too hot<br>▪ 0x05: BIOS life sign timeout<br>▪ 0x06: System reset timeout<br>▪ 0x07: Platform reset failure<br>▪ 0x08: Chipset handshake failure<br>▪ 0x09: System power OK failure<br>▪ 0xFF: Error |

#### 5.4.1.16 Power Failure Flags

This command allows querying the power failure flags of the G25A.

*Table 53. BMC API – Power failure flags command*

| Command | Packet Type | Opcode | Functional Description |
|---|---|---|---|
| *ERR_PWR_FLAGS* | *PT_RBD* | 0x91 | Get power failure flags |

***Command ERR_PWR_FLAGS***

Whenever a power failure occurs, the respective flag is set to 1 until the Power Failure Flag Register is cleared.

| **Opcode: 0x91** | | | | | | | | **Packet Type: *PT_RBD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | *BATT* | - | *EXT* | *SYS_ PWROK* | *12V* | *5V_ STDBY* | *5V* | *33V* |

| Bit Field | Description |
|---|---|
| Initial Value | 0x00: No power failure was registered by the BMC since the Power Failure Flag Register was cleared. |
| *BATT* | Battery failure |
| *EXT* | External power supply failure |
| *SYS_PWROK* | System power OK failure |
| *12V* | +12 V input voltage failure |
| *5V_STDBY* | +5 V standby voltage failure |
| *5V* | +5 V input voltage failure |
| *33V* | +3.3 V voltage failure |

#### 5.4.1.17 Reset Reason

This command allows querying the reason of the last reset. The BMC maintains a Reset Reason Register that stores the reason for the last reset issued by the BMC.

*Table 54. BMC API – Reset reason command*

| Command | Packet Type | Opcode | Functional Description |
|---|---|---|---|
| *ERR_RST_RSN* | *PT_RBD* | 0x92 | Get reason of last reset |

***Command ERR_RST_RSN***

| **Opcode: 0x92** | | | | | | | | **Packet Type: *PT_RBD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | *RST_REASON* | | | | | | | |

| Bit Field | Description |
|---|---|
| *RST_REASON* | Reason of last reset<br>▪ 0x00: Initial value; no reset was issued by the BMC since the Reset Reason Register was cleared<br>▪ 0x01: Regular reset<br>▪ 0x02: External BMC watchdog timeout (application software timeout)<br>▪ 0x03: Internal BMC watchdog timeout<br>▪ 0x04: Internal brown-out reset (BMC undervoltage)<br>▪ 0x05: External reset<br>▪ 0x06: Platform reset<br>▪ 0x07: Software warm reset<br>▪ 0x08: Software cold reset<br>▪ 0x09: Software cold reset with RTC reset<br>▪ 0x0A: Power failure<br>▪ 0x0B: Chipset handshaking timeout<br>▪ 0x0C: PLT_RST timeout<br>▪ 0x0D: BIOS life sign timeout |

#### 5.4.1.18 Clear Error Registers

This command allows clearing the Reset Reason Register, Last Error Register and Power Failure Flag Register, collectively called 'error registers'.

*Table 55. BMC API – Clear error registers command*

| Command | Packet Type | Opcode | Functional Description |
|---|---|---|---|
| *ERR_REG_CLR* | *PT_WBD* | 0x9F | Clear error registers |

***Command ERR_REG_CLR***

| **Opcode: 0x9F** | | | | | | | | **Packet Type: *PT_WBD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | 0x69 | | | | | | | |

#### 5.4.1.19 Power Cycle Counter

The power cycle counter counts the number of power cycles of the external power supply, i.e. the number of times the system changes from S5 into S0 state.
S0 to S5 are the power states as defined in the ACPI specification, or an equivalent state

Please refer to the ACPI Specification for more details on the power states S0 to S5:
Advanced Configuration and Power Interface Specification Version 6.1
January, 2016
Unified EFI Forum
uefi.org/specifications

The counter is set to zero during factory programming.

*Table 56. BMC API – Power cycle counter command*

| Command | Packet Type | Opcode | Functional Description |
|---|---|---|---|
| *PWRCYCL_CNT* | *PT_RWD* | 0x93 | Get power cycle counter |

***Command PWRCYCL_CNT***

| **Opcode: 0x93** | | | | | | | | **Packet Type: *PT_RWD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data 0** | *PWR_CYCLES* (LSB) | | | | | | | |
| **Data 1** | *PWR_CYCLES* (MSB) | | | | | | | |

| Bit Field | Description |
|---|---|
| *PWR_CYCLES* | Number of power cycles on the external power supply |

#### 5.4.1.20 Operating Hours Counter

This command allows querying the operating hours counter. The operating hours counter counts the number of hours and minutes the board has been (at least partly) powered on, i.e. when the system is in S3 or S0 state.
S0 to S5 are the power states as defined in the ACPI specification, or an equivalent state

Please refer to the ACPI Specification for more details on the power states S0 to S5:
Advanced Configuration and Power Interface Specification Version 6.1
January, 2016
Unified EFI Forum
uefi.org/specifications

The counter is set to zero during factory programming.

*Table 57. BMC API – Operating hours counter command*

| Command | Packet Type | Opcode | Functional Description |
|---|---|---|---|
| *OP_HRS_CNT* | *PT_RWD* | 0x94 | Get Operating Hours Counter |

***Command OP_HRS_CNT***

| **Opcode: 0x94** | | | | | | | | **Packet Type: *PT_RWD*** |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data 0** | *OP_TIME* (LSB) | | | | | | | |
| **Data 1** | *OP_TIME* (MSB) | | | | | | | |

| Bit Field | Description |
|---|---|
| *OP_TIME* | Number of hours the board has been powered on |

#### 5.4.1.21 Status LED Control

This command allows controlling status LEDs, depending on implementation on the product.

*Table 58. BMC API – Status LED control command*

| Command | Packet Type | Opcode | Functional Description |
|---|---|---|---|
| *LED_CTRL_SET* | *PT_WBD* | 0xA0 | Set LED state |
| *LED_CTRL_GET* | *PT_RBD* | 0xA0 | Get LED state |

***Command LED_CTRL_SET***

| **Opcode: 0xA0** | | | | | | | **Packet Type: *PT_WBD*** | |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | - | | | | | | *HTSWP* | *STA* |

***Command LED_CTRL_GET***

| **Opcode: 0xA0** | | | | | | | **Packet Type: *PT_RBD*** | |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Data** | - | | | | | | *HTSWP* | *STA* |

| Bit Field | Description |
|---|---|
| *HTSWP* | Hot swap LED |
| *STA* | Status LED at front panel |

### 5.4.2 Example BMC API Usage

- See Chapter 2.7.1 Accessing Board Management Functions on page 26 for how to access board management functions under Linux.
- See Chapter 2.6.1 Accessing SMBus/I2C Devices on page 24 for how to access board management functions under Windows.

# 6 Maintenance

## 6.1 Lithium Battery

This board might contain a lithium battery. There is a danger of explosion if the battery is incorrectly replaced!

- Replace only with the same or equivalent type.
- Manufacturer: Renata
- Type: CR1025
- Capacity: 30 mAh
- The battery has to be UL listed.

Used batteries have to be disposed of according to the local regulations concerning the disposal of hazardous waste.

*Figure 8. Position of lithium battery on G25A*