Connect. Accelerate. Outperform.™

Chassis Management for Mellanox Switch Systems with Sysfs

User Manual

Rev 1.0

www.mellanox.com

NOTE:
THIS HARDWARE, SOFTWARE OR TEST SUITE PRODUCT (“PRODUCT(S)”) AND ITS RELATED DOCUMENTATION ARE PROVIDED BY MELLANOX TECHNOLOGIES “AS-IS” WITH ALL FAULTS OF ANY KIND AND SOLELY FOR THE PURPOSE OF AIDING THE CUSTOMER IN TESTING APPLICATIONS THAT USE THE PRODUCTS IN DESIGNATED SOLUTIONS. THE CUSTOMER'S MANUFACTURING TEST ENVIRONMENT HAS NOT MET THE STANDARDS SET BY MELLANOX TECHNOLOGIES TO FULLY QUALIFY THE PRODUCT(S) AND/OR THE SYSTEM USING IT. THEREFORE, MELLANOX TECHNOLOGIES CANNOT AND DOES NOT GUARANTEE OR WARRANT THAT THE PRODUCTS WILL OPERATE WITH THE HIGHEST QUALITY. ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT ARE DISCLAIMED. IN NO EVENT SHALL MELLANOX BE LIABLE TO CUSTOMER OR ANY THIRD PARTIES FOR ANY DIRECT, INDIRECT, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES OF ANY KIND (INCLUDING, BUT NOT LIMITED TO, PAYMENT FOR PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY FROM THE USE OF THE PRODUCT(S) AND RELATED DOCUMENTATION EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Mellanox Technologies
350 Oakmead Parkway Suite 100
Sunnyvale, CA 94085
U.S.A.
www.mellanox.com
Tel: (408) 970-3400
Fax: (408) 970-3403

Mellanox Technologies, Ltd.
Hakidma 26
Ofer Industrial Park
Yokneam 2069200
Israel
www.mellanox.com
Tel: +972 (0)74 723 7200
Fax: +972 (0)4 959 3245

Table of Contents

Document Revision History

Table 1 - Document Revision History

Revision	Date	Description
1.0	September 8, 2015	First release

About this Manual

This manual describes using sysfs in order to control Mellanox switch HW.

Audience

This manual is intended for developers creating management software over Mellanox switches using Mellanox SDK.

Document Conventions

The following lists conventions used in this document.

NOTE: Identifies important information that contains helpful suggestions.

CAUTION: Alerts you to the risk of personal injury, system damage, or loss of data.

WARNING: Warns you that failure to take or avoid a specific action might result in personal injury or a malfunction of the hardware or software. Be aware of the hazards involved with electrical circuitry and be familiar with standard practices for preventing accidents before you work on any equipment.

1 Introduction

Mellanox SDK support hardware (HW) management and control using a virtual file system provided by the Linux kernel called sysfs.

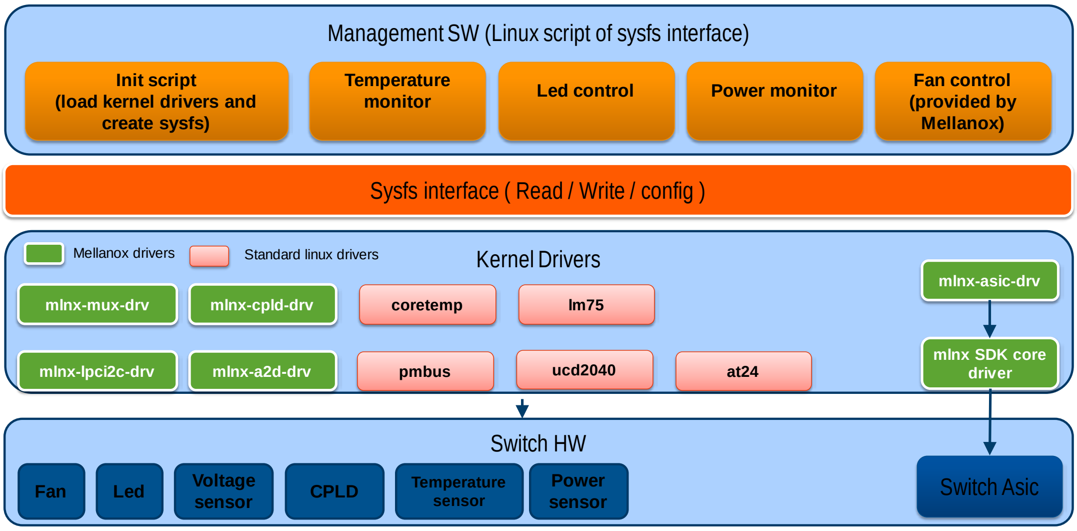
The sysfs file system enumerates the devices and buses attached to the system in a file system hierarchy that can be accessed from user space. It is designed to handle the device and driver specific options that have previously resided in /proc/, and to encompass the dynamic device addition previously offered by devfs.

The major advantage of working with sysfs is that it makes HW hierarchy easy to understand and control without having to learn about HW component location and the buses through which they are connected.

1.1 Software Components

Figure 1 presents the software (SW) hierarchy and layer separation for sysfs support.

Figure 1 - Sysfs Layout



1.2 Hierarchy and Structure

The DVS-OS supports the default hierarchy structure of sysfs under the directory /sys/bus/i2c/devices/. This path is used by existing applications that use auto-discovery to find existing HW components. Two examples for such applications are:

- libsysfs – the libraries provide a consistent and stable interface for querying system device information exposed through sysfs.
- systool – a utility built upon libsysfs that lists devices by bus, class, and topology.

The disadvantage of using this path is that the hierarchy model includes the BUS type and location model which is subject to change between different system types. For example: /sys/bus/i2c/devices/2-0060/led1.

To resolve this limitation, the DVS-OS also supports a new virtual hierarchy structure that is not HW dependent. This hierarchy is a collection of soft links to the default sysfs structure. This document delves inot the way to work with this hierarchy in order to control the HW.

Chassis attributes information exported through sysfs can be utilized by a number of standard Linux tools. So, for example, the following are tools from the Linux packages lm-sensors and fancontrol, which are capable of operating on top of Mellanox sysfs infrastructure:

- pwmconfig – tests the pulse width modulation (PWM) outputs of sensors and configures fancontrol
- fancontrol – automated software based fan speed regulation
- sensors – print sensors information

1.3 Sysfs Initialization and Driver Registration

As describe in the previous sections, sysfs structure provide access to HW drivers. These drivers need to be initialized before using sysfs. In addition, Mellanox virtual hierarchy also needs to be created in order to use it.

Mellanox SDK provides a simple way to initialize the aforementioned drivers using a shell script located at /etc/mlnx/msn2700. This script supports initialization and de-initialization of driver and virtual hierarchy structure.

- For initialization, run the command /etc/mlnx/msn2700 start
- For de-initialization, run the command /etc/mlnx/msn2700 stop

NOTE: SDK must be initialized before running msn2700 start.

2 Virtual SysFS Hierarchy

Mellanox virtual hierarchy supports the following HW control:

Table 2 - Mellanox Hierarchy Node Support

Node Path	Purpose
/bsp/module	Gets information on module status (e.g. insert or removed)
/bsp/fan	Sets/gets fans speed and minimum allow speed
/bsp/thermal	Gets information from thermal sensors
/bsp/power	Gets information from power sensors
/bsp/led	Sets/gets LED color
/bsp/cpld	Gets CPLD version information
/bsp/eeprom	Gets raw data from EEPROM in system modules
/bsp/system	Controls system reset

Detailed information on each of these nodes can be found in the following sections.

2.1 Module Control

2.1.1 Read Fan Status

Node name	/bsp/module/fan<fan module number>_status		
Description	Read fan module status		
Access	Read only		
Release version	1.0		
Arguments	Name	Data type	Values
	Status	Integer	0 – not present 1 – present
Example	Get fan module 1 status: **cat /bsp/module/fan1_status**		

2.1.2 Read Power Supply Status

Node name	/bsp/module/psu<power supply module number>_status		
Description	Read power supply module status		
Access	Read only		
Release version	1.0		
Arguments	Name	Data type	Values
	Status	Integer	0 – not present 1 – present
Example	Get power supply 1 status: **cat /bsp/module/psu1_status**		

2.1.3 Read Power Supply Power Status

Node name	/bsp/module/psu<power supply module number>_pwr_status		
Description	Read power supply power status		
Access	Read only		
Release version	1.0		
Arguments	Name	Data type	Values
	Status	Integer	0 – no power or cable not present 1 – power present
Example	Get power supply 1 status: **cat /bsp/module/psu1_pwr_status**		

2.2 Fan Control

2.2.1 Read Fan Speed

Node name	/bsp/fan/fan<fan module number>_<fan number>_get		
Description	Read fan speed information in RPM		
Access	Read only		
Release version	1.0		
Arguments	Name	Data type	Values
	Fan speed	Integer	0-23000 RPM
Example	Get fan module 1/fan 0 speed: **cat /bsp/fan/fan1_0_speed_get**		

2.2.2 Set Fan Speed

Node name	/bsp/fan/fan<fan module number>_set		
Description	Set fan speed in parentage of maximum speed		
Access	Write only		
Release version	1.0		
Arguments	Name	Data type	Values
	Fan speed	Integer	0-255 (PWM duty cycles)
Example	Set fan module 1 to 60% (153 PWM duty cycles) of top speed.: **echo 60 > /bsp/fan/fan1_speed_set** **Note:** Setting fan speed is allowed per fan module and not per fan unit inside a module. **Note:** PWM to percent mapping is 0-100% to o-255 PWM.		

2.2.3 Get Fan Min Speed

Node name	/bsp/fan/fan<fan module number>_min		
Description	Get fan minimum allow speed in parentage of maximum speed		
Access	Read only		
Release version	1.0		
Arguments	Name	Data type	Values
	Fan speed	Integer	0-23000 RPM
Example	Get fan module 1 minimum speed: **cat /bsp/fan/fan1_speed_min**		

2.3 Thermal Control

2.3.1 Read Switch ASIC Temperature

Node name	/bsp/thermal/asic		
Description	Read value of switch module ASIC temperature		
Access	Read only		
Release version	1.0		
Arguments	Name	Data type	Values
	Temperature	Integer	Degrees in Celsius
Example	Get switch module ASIC temperature: **cat /bsp/thermal/asic**		

2.3.2 Read Switch CPU Temperature

Node name	/bsp/thermal/cpu_<core0 | core1 | pack>		
Description	Read value of CPU module temperature		
Access	Read only		
Release version	1.0		
Arguments	Name	Data type	Values
	Temperature	Integer	Degrees in Celsius
Example	Get CPU Core 0 temperature: **cat /bsp/thermal/cpu_core0**		

2.3.3 Read Switch Board Temperature

Node name	/bsp/thermal/board_amb		
Description	Read value of switch board ambient temperature		
Access	Read only		
Release version	1.0		
Arguments	Name	Data type	Values
	Temperature	Integer	Degrees in Celsius
Example	Get switch board ambient temperature: **cat /bsp/thermal/board_amb**		

2.3.4 Read Switch Port Temperature

Node name	/bsp/thermal/port_amb		
Description	Read value of switch port ambient temperature		
Access	Read only		
Release version	1.0		
Arguments	Name	Data type	Values
	Temperature	Integer	Degrees in Celsius
Example	Get switch board ambient temperature: **cat /bsp/thermal/port_amb**		

2.3.5 Read Switch Power Supply Temperature

Node name	/bsp/thermal/psu<psu module number>		
Description	Read value of power supply temperature		
Access	Read only		
Release version	1.0		
Arguments	Name	Data type	Values
	Temperature	Integer	Degrees in Celsius
Example	Get switch power supply 1 temperature: **cat /bsp/thermal/psu1**		

2.4 LED Control

2.4.1 Control Fan Status LED

Node name	/bsp/led/led_fan<fan module number>		
Description	Read/write fan module status LED		
Access	Read / Write		
Release version	1.0		
Arguments	Name	Data type	Values
	LED color	Integer	green_blink_fast red_blink_fast green_blink red_blink green red
Example	Get fan module 1 status LED color: **cat /bsp/led/fan1** Set fan module 1 LED color to green: **echo green > /bsp/led/led_fan1**		

2.4.2 Get Fan LED Capabilities

Node name	/bsp/led/led_fan<fan module number>capability		
Description	Read fan module status LED		
Access	Read only		
Release version	1.0		
Arguments	Name	Data type	Values
	LED capabilities	Integer	green_blink_fast red_blink_fast green_blink red_blink green red
Example	Get fan module 1 capabilities: **cat /bsp/led/fan1_capability**		

2.4.3 Control Power Supply Status LED

Node name	/bsp/led/led_psu< power supply module number>		
Description	Set/get power supply module status LED		
Access	Read / Write		
Release version	1.0		
Arguments	Name	Data type	Values
	LED color	Integer	green_blink_fast red_blink_fast green_blink red_blink green red
Example	Get power supply module 1 status LED color: **cat /bsp/led/psu1** Set power supply module 1 LED color to green: **echo green > /bsp/led/led_psu1**		

2.4.4 Get Fan LED Capabilities

Node name	/bsp/led/led_psu< power supply module number>capability		
Description	Set/get power supply module status LED		
Access	Read only		
Release version	1.0		
Arguments	Name	Data type	Values
	LED capabilities	Integer	green_blink_fast red_blink_fast green_blink red_blink green red
Example	Get power supply module 1 capabilities: **cat /bsp/led/psu1_capability**		

2.4.5 Control System Status LED

Node name	/bsp/led/led_system		
Description	Set/get system status LED		
Access	Read / Write		
Release version	1.0		
Arguments	Name	Data type	Values
	LED color	Integer	green_blink_fast red_blink_fast green_blink red_blink green red
Example	Get system status LED color: **cat /bsp/led/led_system** Set system LED color to green: **echo green > /bsp/led/psu1**		

2.4.6 Get Fan LED Capabilities

Node name	/bsp/led/led_system_capability		
Description	Set/get system status LED		
Access	Read only		
Release version	1.0		
Arguments	Name	Data type	Values
	LED capabilities	Integer	green_blink_fast red_blink_fast green_blink red_blink green red
Example	Get system status LED capabilities: **cat /bsp/led/led_system_capability**		

2.5 CPLD Control

2.5.1 Read CPLD Version Number

Node name	/bsp/cpld/< cpld_brd | cpld_mgmt | cpld_port>_version		
Description	Read cpld version number		
Access	Read only		
Release version	1.0		
Arguments	Name	Data type	Values
	CPLD version	Integer	>= 0
Example	Get management board CPLD version: **cat /bsp/cpld/cpld_brd_version**		

2.6 EEPROM Control

2.6.1 Read CPU EEPROM Data

Node name	/bsp/eeprom/cpu_info		
Description	Read CPU raw data in hex format		
Access	Read only		
Release version	1.0		
Arguments	Name	Data type	Values
	EEPROM information	Hex	Hex dump format of memory
Example	Get CPU EEPROM information: **cat /bsp/eeprom/cpu_info**		

2.6.2 Read Fan Module EEPROM Data

Node name	/bsp/eeprom/fan<fan module number>_info		
Description	Read fan module raw data in hex format		
Access	Read only		
Release version	1.0		
Arguments	Name	Data type	Values
	EEPROM information	Hex	Hex dump format of memory
Example	Get fan module 1 EEPROM information: **cat /bsp/eeprom/fan1_info**		

2.6.3 Read Power Supply Module EEPROM Data

Node name	/bsp/eeprom/psu<power supply module number>_info		
Description	Read power supply module raw data in hex format		
Access	Read only		
Release version	1.0		
Arguments	Name	Data type	Values
	EEPROM information	Hex	Hex dump format of memory
Example	Get power supply module 1 EEPROM information: **cat /bsp/eeprom/psu1_info**		

2.6.4 Read System Chassis EEPROM Data

Node name	/bsp/eeprom/vpd_info		
Description	Read system chassis raw data in hex format		
Access	Read only		
Release version	1.0		
Arguments	Name	Data type	Values
	EEPROM information	Hex	Hex dump format of memory
Example	Get system chassis EEPROM information: **cat /bsp/eeprom/vpd_info**		

2.7 Control System Reset

2.7.1 Reset the System

Node name	/bsp/system/power_cycle		
Description	Reset the system		
Access	Write only		
Release version	1.0		
Arguments	Name	Data type	Values
	Reset	Hex	1 – reset the system
Example	Reset the switch system: **cat /bsp/system/power_cycle**		

3 Thermal Control

DVS-OS is equipped with a user application for controlling temperature on MSN2700 systems. The application is based on sysfs as already detailed in the previous chapters. In order to use the application, sysfs and SDK must be initialized (see section 1.3 “Sysfs Initialization and Driver Registration”).

3.1.1 Thermal Algorithm

The thermal algorithm controls the switch temperature by monitoring the switch ASIC. The algorithm has the following parameters:

- Critical temperature – a temperature that ASIC should not reach
- Hot temperature – the lower threshold for increasing fan speed
- Cold temperature – the high threshold for decreasing fan speed
- Min fan speed – the minimum allowed fan speed (60% by default)

By default, the application monitors temperature every 15 second.

The following is an example of the thermal control algorithm:

1. If one of the fans is unresponsive, set all others to 100%
2. If ASIC temperature is above critical temperature, set fans to 100%
3. If ASIC temperature is greater than hot temperature, increase fan speed by 5% (if not already at 100%)
4. If temperature below cold temperature, decrease speed by 5% but not below min fan speed

3.1.2 Starting and Stopping Thermal Algorithm

To start the thermal algorithm, run the command /usr/sbin/thermal_watch_start.sh.

The script features two optional parameters:

- Min fan speed (which by default is set at 60%).

NOTE: It is not recommended to tamper with this configuration.

- Polling interval in seconds (which by default is set at 15 seconds).

To stop the thermal algorithm, run the command /usr/sbin/thermal_watch_stop.sh.

Another way to activate the thermal control is to use virtual sysfs hierarchy under: /bsp/thermal_zone1/thermal_control_activate and /bsp/thermal_zone1/thermal_control_deactivate.