Linux Hwmon Documentation

The kernel development community

Jul 14, 2020

CONTENTS

CHAPTER

ONE

THE LINUX HARDWARE MONITORING KERNEL API

Guenter Roeck

1.1 Introduction

This document describes the API that can be used by hardware monitoring drivers that want to use the hardware monitoring framework.

This document does not describe what a hardware monitoring (hwmon) Driver or Device is. It also does not describe the API which can be used by user space to communicate with a hardware monitoring device. If you want to know this then please read the following file: Documentation/hwmon/sysfs-interface.rst.

For additional guidelines on how to write and improve hwmon drivers, please also read Documentation/hwmon/submitting-patches.rst.

1.2 The API

Each hardware monitoring driver must #include <linux/hwmon.h> and, in most cases, <linux/hwmon-sysfs.h>. linux/hwmon.h declares the following register/unregister functions:

```
struct device *
hwmon_device_register_with_groups(struct device *dev, const char *name,
                                  void *drvdata,
                                  const struct attribute_group **groups);

struct device *
devm_hwmon_device_register_with_groups(struct device *dev,
                                       const char *name, void *drvdata,
                                       const struct attribute_group␣
↪**groups);

struct device *
hwmon_device_register_with_info(struct device *dev,
                                const char *name, void *drvdata,
                                const struct hwmon_chip_info *info,
                                const struct attribute_group **extra_
↪groups);
```

(continues on next page)

(continued from previous page)

```
struct device *
devm_hwmon_device_register_with_info(struct device *dev,
                                     const char *name,
                                     void *drvdata,
                                     const struct hwmon_chip_info *info,
                                     const struct attribute_group **extra_
↪groups);

void hwmon_device_unregister(struct device *dev);

void devm_hwmon_device_unregister(struct device *dev);
```

hwmon_device_register_with_groups registers a hardware monitoring device. The first parameter of this function is a pointer to the parent device. The name parameter is a pointer to the hwmon device name. The registration function wil create a name sysfs attribute pointing to this name. The drvdata parameter is the pointer to the local driver data. hwmon_device_register_with_groups will attach this pointer to the newly allocated hwmon device. The pointer can be retrieved by the driver using dev_get_drvdata() on the hwmon device pointer. The groups parameter is a pointer to a list of sysfs attribute groups. The list must be NULL terminated. hwmon_device_register_with_groups creates the hwmon device with name attribute as well as all sysfs attributes attached to the hwmon device. This function returns a pointer to the newly created hardware monitoring device or PTR_ERR for failure.

devm_hwmon_device_register_with_groups is similar to hwmon_device_register_with_groups. However, it is device managed, meaning the hwmon device does not have to be removed explicitly by the removal function.

hwmon_device_register_with_info is the most comprehensive and preferred means to register a hardware monitoring device. It creates the standard sysfs attributes in the hardware monitoring core, letting the driver focus on reading from and writing to the chip instead of having to bother with sysfs attributes. The parent device parameter cannot be NULL with non-NULL chip info. Its parameters are described in more detail below.

devm_hwmon_device_register_with_info is similar to hwmon_device_register_with_info. However, it is device managed, meaning the hwmon device does not have to be removed explicitly by the removal function.

hwmon_device_unregister deregisters a registered hardware monitoring device. The parameter of this function is the pointer to the registered hardware monitoring device structure. This function must be called from the driver remove function if the hardware monitoring device was registered with hwmon_device_register_with_groups or hwmon_device_register_with_info.

devm_hwmon_device_unregister does not normally have to be called. It is only needed for error handling, and only needed if the driver probe fails after the call to devm_hwmon_device_register_with_groups or hwmon_device_register_with_info and if the automatic (device managed) removal would be too late.

All supported hwmon device registration functions only accept valid device names. Device names including invalid characters (whitespace, '*', or '-') will be rejected. The 'name' parameter is mandatory.

1.3 Using devm_hwmon_device_register_with_info()

hwmon_device_register_with_info() registers a hardware monitoring device. The parameters to this function are

struct device *dev	Pointer to parent device
const char *name	Device name
void *drvdata	Driver private data
const struct hwmon_chip_info *info	Pointer to chip description.
const struct attribute_group **extra_groups	Null-terminated list of additional non-standard sysfs attribute groups.

This function returns a pointer to the created hardware monitoring device on success and a negative error code for failure.

The hwmon_chip_info structure looks as follows:

```
struct hwmon_chip_info {
        const struct hwmon_ops *ops;
        const struct hwmon_channel_info **info;
};
```

It contains the following fields:

- **ops:** Pointer to device operations.
- **info:** NULL-terminated list of device channel descriptors.

The list of hwmon operations is defined as:

```
struct hwmon_ops {
      umode_t (*is_visible)(const void *, enum hwmon_sensor_types type,
                            u32 attr, int);
      int (*read)(struct device *, enum hwmon_sensor_types type,
                  u32 attr, int, long *);
      int (*write)(struct device *, enum hwmon_sensor_types type,
                   u32 attr, int, long);
};
```

It defines the following operations.

- **is_visible:** Pointer to a function to return the file mode for each supported attribute. This function is mandatory.
- **read:** Pointer to a function for reading a value from the chip. This function is optional, but must be provided if any readable attributes exist.
- **write:** Pointer to a function for writing a value to the chip. This function is optional, but must be provided if any writeable attributes exist.

Each sensor channel is described with struct hwmon_channel_info, which is defined as follows:

```
struct hwmon_channel_info {
        enum hwmon_sensor_types type;
        u32 *config;
};
```

It contains following fields:

- **type:** The hardware monitoring sensor type.

  Supported sensor types are

hwmon_chip	A virtual sensor type, used to describe attributes which are not bound to a specific input or output
hwmon_temp	Temperature sensor
hwmon_in	Voltage sensor
hwmon_curr	Current sensor
hwmon_power	Power sensor
hwmon_energy	Energy sensor
hwmon_humidity	Humidity sensor
hwmon_fan	Fan speed sensor
hwmon_pwm	PWM control

- **config:** Pointer to a 0-terminated list of configuration values for each sensor of the given type. Each value is a combination of bit values describing the attributes supposed by a single sensor.

As an example, here is the complete description file for a LM75 compatible sensor chip. The chip has a single temperature sensor. The driver wants to register with the thermal subsystem (HWMON_C_REGISTER_TZ), and it supports the update_interval attribute (HWMON_C_UPDATE_INTERVAL). The chip supports reading the temperature (HWMON_T_INPUT), it has a maximum temperature register (HWMON_T_MAX) as well as a maximum temperature hysteresis register (HWMON_T_MAX_HYST):

```
static const u32 lm75_chip_config[] = {
        HWMON_C_REGISTER_TZ | HWMON_C_UPDATE_INTERVAL,
        0
};

static const struct hwmon_channel_info lm75_chip = {
        .type = hwmon_chip,
        .config = lm75_chip_config,
};

static const u32 lm75_temp_config[] = {
```

(continues on next page)

(continued from previous page)

```
        HWMON_T_INPUT | HWMON_T_MAX | HWMON_T_MAX_HYST,
        0
};

static const struct hwmon_channel_info lm75_temp = {
        .type = hwmon_temp,
        .config = lm75_temp_config,
};

static const struct hwmon_channel_info *lm75_info[] = {
        &lm75_chip,
        &lm75_temp,
        NULL
};

The HWMON_CHANNEL_INFO() macro can and should be used when possible.
With this macro, the above example can be simplified to

static const struct hwmon_channel_info *lm75_info[] = {
        HWMON_CHANNEL_INFO(chip,
                           HWMON_C_REGISTER_TZ | HWMON_C_UPDATE_INTERVAL),
        HWMON_CHANNEL_INFO(temp,
                           HWMON_T_INPUT | HWMON_T_MAX | HWMON_T_MAX_HYST),
        NULL
};

The remaining declarations are as follows.

static const struct hwmon_ops lm75_hwmon_ops = {
        .is_visible = lm75_is_visible,
        .read = lm75_read,
        .write = lm75_write,
};

static const struct hwmon_chip_info lm75_chip_info = {
        .ops = &lm75_hwmon_ops,
        .info = lm75_info,
};
```

A complete list of bit values indicating individual attribute support is defined in include/linux/hwmon.h. Definition prefixes are as follows.

HWMON_C_xxxx	Chip attributes, for use with hwmon_chip.
HWMON_T_xxxx	Temperature attributes, for use with hwmon_temp.
HWMON_I_xxxx	Voltage attributes, for use with hwmon_in.
HWMON_C_xxxx	Current attributes, for use with hwmon_curr. Notice the prefix overlap with chip attributes.
HWMON_P_xxxx	Power attributes, for use with hwmon_power.
HWMON_E_xxxx	Energy attributes, for use with hwmon_energy.
HWMON_H_xxxx	Humidity attributes, for use with hwmon_humidity.
HWMON_F_xxxx	Fan speed attributes, for use with hwmon_fan.
HWMON_PWM_xxxx	PWM control attributes, for use with hwmon_pwm.

1.4 Driver callback functions

Each driver provides is_visible, read, and write functions. Parameters and return values for those functions are as follows:

```
umode_t is_visible_func(const void *data, enum hwmon_sensor_types type,
                        u32 attr, int channel)
```

Parameters:

data: Pointer to device private data structure.

type: The sensor type.

attr: Attribute identifier associated with a specific attribute. For example, the attribute value for HWMON_T_INPUT would be hwmon_temp_input. For complete mappings of bit fields to attribute values please see include/linux/hwmon.h.

channel: The sensor channel number.

Return value: The file mode for this attribute. Typically, this will be 0 (the attribute will not be created), S_IRUGO, or ‘S_IRUGO | S_IWUSR’.

```
int read_func(struct device *dev, enum hwmon_sensor_types type,
              u32 attr, int channel, long *val)
```

Parameters:

dev: Pointer to the hardware monitoring device.

type: The sensor type.

attr: Attribute identifier associated with a specific attribute. For example, the attribute value for HWMON_T_INPUT would be hwmon_temp_input. For complete mappings please see include/linux/hwmon.h.

channel: The sensor channel number.

val: Pointer to attribute value.

Return value: 0 on success, a negative error number otherwise.

```
int write_func(struct device *dev, enum hwmon_sensor_types type,
               u32 attr, int channel, long val)
```

Parameters:

dev: Pointer to the hardware monitoring device.

type: The sensor type.

attr: Attribute identifier associated with a specific attribute. For example, the attribute value for HWMON_T_INPUT would be hwmon_temp_input. For complete mappings please see include/linux/hwmon.h.

channel: The sensor channel number.

val: The value to write to the chip.

Return value: 0 on success, a negative error number otherwise.

1.5 Driver-provided sysfs attributes

If the hardware monitoring device is registered with hwmon_device_register_with_info or devm_hwmon_device_register_with_info, it is most likely not necessary to provide sysfs attributes. Only additional non-standard sysfs attributes need to be provided when one of those registration functions is used.

The header file linux/hwmon-sysfs.h provides a number of useful macros to declare and use hardware monitoring sysfs attributes.

In many cases, you can use the exsting define DEVICE_ATTR or its variants DEVICE_ATTR_{RW,RO,WO} to declare such attributes. This is feasible if an attribute has no additional context. However, in many cases there will be additional information such as a sensor index which will need to be passed to the sysfs attribute handling function.

SENSOR_DEVICE_ATTR and SENSOR_DEVICE_ATTR_2 can be used to define attributes which need such additional context information. SENSOR_DEVICE_ATTR requires one additional argument, SENSOR_DEVICE_ATTR_2 requires two.

Simplified variants of SENSOR_DEVICE_ATTR and SENSOR_DEVICE_ATTR_2 are available and should be used if standard attribute permissions and function names are feasible. Standard permissions are 0644 for SENSOR_DEVICE_ATTR[_2]_RW, 0444 for SENSOR_DEVICE_ATTR[_2]_RO, and 0200 for SENSOR_DEVICE_ATTR[_2]_WO. Standard functions, similar to DEVICE_ATTR_{RW,RO,WO}, have _show and _store appended to the provided function name.

SENSOR_DEVICE_ATTR and its variants define a struct sensor_device_attribute variable. This structure has the following fields:

```
struct sensor_device_attribute {
        struct device_attribute dev_attr;
        int index;
};
```

You can use to_sensor_dev_attr to get the pointer to this structure from the attribute read or write function. Its parameter is the device to which the attribute is attached.

SENSOR_DEVICE_ATTR_2 and its variants define a struct sensor_device_attribute_2 variable, which is defined as follows:

```
struct sensor_device_attribute_2 {
        struct device_attribute dev_attr;
        u8 index;
        u8 nr;
};
```

Use to_sensor_dev_attr_2 to get the pointer to this structure. Its parameter is the device to which the attribute is attached.

CHAPTER

TWO

PMBUS CORE DRIVER AND INTERNAL API

2.1 Introduction

[from pmbus.org] The Power Management Bus (PMBus) is an open standard power-management protocol with a fully defined command language that facilitates communication with power converters and other devices in a power system. The protocol is implemented over the industry-standard SMBus serial interface and enables programming, control, and real-time monitoring of compliant power conversion products. This flexible and highly versatile standard allows for communication between devices based on both analog and digital technologies, and provides true interoperability which will reduce design complexity and shorten time to market for power system designers. Pioneered by leading power supply and semiconductor companies, this open power system standard is maintained and promoted by the PMBus Implementers Forum (PMBus-IF), comprising 30+ adopters with the objective to provide support to, and facilitate adoption among, users.

Unfortunately, while PMBus commands are standardized, there are no mandatory commands, and manufacturers can add as many non-standard commands as they like. Also, different PMBUs devices act differently if non-supported commands are executed. Some devices return an error, some devices return 0xff or 0xffff and set a status error flag, and some devices may simply hang up.

Despite all those difficulties, a generic PMBus device driver is still useful and supported since kernel version 2.6.39. However, it was necessary to support device specific extensions in addition to the core PMBus driver, since it is simply unknown what new device specific functionality PMBus device developers come up with next.

To make device specific extensions as scalable as possible, and to avoid having to modify the core PMBus driver repeatedly for new devices, the PMBus driver was split into core, generic, and device specific code. The core code (in pmbus_core.c) provides generic functionality. The generic code (in pmbus.c) provides support for generic PMBus devices. Device specific code is responsible for device specific initialization and, if needed, maps device specific functionality into generic functionality. This is to some degree comparable to PCI code, where generic code is augmented as needed with quirks for all kinds of devices.

2.2 PMBus device capabilities auto-detection

For generic PMBus devices, code in pmbus.c attempts to auto-detect all supported PMBus commands. Auto-detection is somewhat limited, since there are simply too many variables to consider. For example, it is almost impossible to autodetect which PMBus commands are paged and which commands are replicated across all pages (see the PMBus specification for details on multi-page PMBus devices).

For this reason, it often makes sense to provide a device specific driver if not all commands can be auto-detected. The data structures in this driver can be used to inform the core driver about functionality supported by individual chips.

Some commands are always auto-detected. This applies to all limit commands (lcrit, min, max, and crit attributes) as well as associated alarm attributes. Limits and alarm attributes are auto-detected because there are simply too many possible combinations to provide a manual configuration interface.

2.3 PMBus internal API

The API between core and device specific PMBus code is defined in drivers/hwmon/pmbus/pmbus.h. In addition to the internal API, pmbus.h defines standard PMBus commands and virtual PMBus commands.

2.3.1 Standard PMBus commands

Standard PMBus commands (commands values 0x00 to 0xff) are defined in the PMBUs specification.

2.3.2 Virtual PMBus commands

Virtual PMBus commands are provided to enable support for non-standard functionality which has been implemented by several chip vendors and is thus desirable to support.

Virtual PMBus commands start with command value 0x100 and can thus easily be distinguished from standard PMBus commands (which can not have values larger than 0xff). Support for virtual PMBus commands is device specific and thus has to be implemented in device specific code.

Virtual commands are named PMBUS_VIRT_xxx and start with PMBUS_VIRT_BASE. All virtual commands are word sized.

There are currently two types of virtual commands.

- READ commands are read-only; writes are either ignored or return an error.
- RESET commands are read/write. Reading reset registers returns zero (used for detection), writing any value causes the associated history to be reset.

Virtual commands have to be handled in device specific driver code. Chip driver code returns non-negative values if a virtual command is supported, or a negative error code if not. The chip driver may return -ENODATA or any other Linux error

code in this case, though an error code other than -ENODATA is handled more efficiently and thus preferred. Either case, the calling PMBus core code will abort if the chip driver returns an error code when reading or writing virtual registers (in other words, the PMBus core code will never send a virtual command to a chip).

2.3.3 PMBus driver information

PMBus driver information, defined in struct pmbus_driver_info, is the main means for device specific drivers to pass information to the core PMBus driver. Specifically, it provides the following information.

- For devices supporting its data in Direct Data Format, it provides coefficients for converting register values into normalized data. This data is usually provided by chip manufacturers in device datasheets.
- Supported chip functionality can be provided to the core driver. This may be necessary for chips which react badly if non-supported commands are executed, and/or to speed up device detection and initialization.
- Several function entry points are provided to support overriding and/or augmenting generic command execution. This functionality can be used to map non-standard PMBus commands to standard commands, or to augment standard command return values with device specific information.

2.4 API functions

2.4.1 Functions provided by chip driver

All functions return the command return value (read) or zero (write) if successful. A return value of -ENODATA indicates that there is no manufacturer specific command, but that a standard PMBus command may exist. Any other negative return value indicates that the commands does not exist for this chip, and that no attempt should be made to read or write the standard command.

As mentioned above, an exception to this rule applies to virtual commands, which must be handled in driver specific code. See “Virtual PMBus Commands” above for more details.

Command execution in the core PMBus driver code is as follows:

```
if (chip_access_function) {
        status = chip_access_function();
        if (status != -ENODATA)
                return status;
}
if (command >= PMBUS_VIRT_BASE) /* For word commands/registers only */
        return -EINVAL;
return generic_access();
```

Chip drivers may provide pointers to the following functions in struct pmbus_driver_info. All functions are optional.

```
int (*read_byte_data)(struct i2c_client *client, int page, int reg);
```

Read byte from page <page>, register <reg>. <page> may be -1, which means "current page".

```
int (*read_word_data)(struct i2c_client *client, int page, int phase,
                      int reg);
```

Read word from page <page>, phase <pase>, register <reg>. If the chip does not support multiple phases, the phase parameter can be ignored. If the chip supports multiple phases, a phase value of 0xff indicates all phases.

```
int (*write_word_data)(struct i2c_client *client, int page, int reg,
                       u16 word);
```

Write word to page <page>, register <reg>.

```
int (*write_byte)(struct i2c_client *client, int page, u8 value);
```

Write byte to page <page>, register <reg>. <page> may be -1, which means "current page".

```
int (*identify)(struct i2c_client *client, struct pmbus_driver_info *info);
```

Determine supported PMBus functionality. This function is only necessary if a chip driver supports multiple chips, and the chip functionality is not pre-determined. It is currently only used by the generic pmbus driver (pmbus.c).

2.4.2 Functions exported by core driver

Chip drivers are expected to use the following functions to read or write PMBus registers. Chip drivers may also use direct I2C commands. If direct I2C commands are used, the chip driver code must not directly modify the current page, since the selected page is cached in the core driver and the core driver will assume that it is selected. Using pmbus_set_page() to select a new page is mandatory.

```
int pmbus_set_page(struct i2c_client *client, u8 page, u8 phase);
```

Set PMBus page register to <page> and <phase> for subsequent commands. If the chip does not support multiple phases, the phase parameter is ignored. Otherwise, a phase value of 0xff selects all phases.

```
int pmbus_read_word_data(struct i2c_client *client, u8 page, u8 phase,
                         u8 reg);
```

Read word data from <page>, <phase>, <reg>. Similar to i2c_smbus_read_word_data(), but selects page and phase first. If the chip does not support multiple phases, the phase parameter is ignored. Otherwise, a phase value of 0xff selects all phases.

```
int pmbus_write_word_data(struct i2c_client *client, u8 page, u8 reg,
                          u16 word);
```

Write word data to <page>, <reg>. Similar to i2c_smbus_write_word_data(), but selects page first.

```
int pmbus_read_byte_data(struct i2c_client *client, int page, u8 reg);
```

Read byte data from <page>, <reg>. Similar to i2c_smbus_read_byte_data(), but selects page first. <page> may be -1, which means "current page".

```
int pmbus_write_byte(struct i2c_client *client, int page, u8 value);
```

Write byte data to <page>, <reg>. Similar to i2c_smbus_write_byte(), but selects page first. <page> may be -1, which means "current page".

```
void pmbus_clear_faults(struct i2c_client *client);
```

Execute PMBus "Clear Fault" command on all chip pages. This function calls the device specific write_byte function if defined. Therefore, it must _not_ be called from that function.

```
bool pmbus_check_byte_register(struct i2c_client *client, int page, int␣
↪reg);
```

Check if byte register exists. Return true if the register exists, false otherwise. This function calls the device specific write_byte function if defined to obtain the chip status. Therefore, it must _not_ be called from that function.

```
bool pmbus_check_word_register(struct i2c_client *client, int page, int␣
↪reg);
```

Check if word register exists. Return true if the register exists, false otherwise. This function calls the device specific write_byte function if defined to obtain the chip status. Therefore, it must _not_ be called from that function.

```
int pmbus_do_probe(struct i2c_client *client, const struct i2c_device_id␣
↪*id,
                   struct pmbus_driver_info *info);
```

Execute probe function. Similar to standard probe function for other drivers, with the pointer to struct pmbus_driver_info as additional argument. Calls identify function if supported. Must only be called from device probe function.

```
void pmbus_do_remove(struct i2c_client *client);
```

Execute driver remove function. Similar to standard driver remove function.

```
const struct pmbus_driver_info
      *pmbus_get_driver_info(struct i2c_client *client);
```

Return pointer to struct pmbus_driver_info as passed to pmbus_do_probe().

2.5 PMBus driver platform data

PMBus platform data is defined in include/linux/pmbus.h. Platform data currently only provides a flag field with a single bit used:

```
#define PMBUS_SKIP_STATUS_CHECK (1 << 0)

struct pmbus_platform_data {
        u32 flags;              /* Device specific flags */
};
```

2.5.1 Flags

PMBUS_SKIP_STATUS_CHECK During register detection, skip checking the status register for communication or command errors.

Some PMBus chips respond with valid data when trying to read an unsupported register. For such chips, checking the status register is mandatory when trying to determine if a chip register exists or not. Other PMBus chips don't support the STATUS_CML register, or report communication errors for no explicable reason. For such chips, checking the status register must be disabled.

Some i2c controllers do not support single-byte commands (write commands with no data, i2c_smbus_write_byte()). With such controllers, clearing the status register is impossible, and the PMBUS_SKIP_STATUS_CHECK flag must be set.

CHAPTER

THREE

KERNEL DRIVER INSPUR-IPSPS1

Supported chips:

- Inspur Power System power supply unit

Author: John Wang <wangzqbj@inspur.com>

3.1 Description

This driver supports Inspur Power System power supplies. This driver is a client to the core PMBus driver.

3.2 Usage Notes

This driver does not auto-detect devices. You will have to instantiate the devices explicitly. Please see Documentation/i2c/instantiating-devices.rst for details.

3.3 Sysfs entries

The following attributes are supported:

curr1_input	Measured input current
curr1_label	“iin”
curr1_max	Maximum current
curr1_max_alarm	Current high alarm
curr2_input	Measured output current in mA.
curr2_label	“iout1”
curr2_crit	Critical maximum current
curr2_crit_alarm	Current critical high alarm
curr2_max	Maximum current
curr2_max_alarm	Current high alarm
fan1_alarm	Fan 1 warning.
fan1_fault	Fan 1 fault.
fan1_input	Fan 1 speed in RPM.
in1_alarm	Input voltage under-voltage alarm.

Continued on next page

Table 1 – continued from previous page

in1_input	Measured input voltage in mV.
in1_label	“vin”
in2_input	Measured output voltage in mV.
in2_label	“vout1”
in2_lcrit	Critical minimum output voltage
in2_lcrit_alarm	Output voltage critical low alarm
in2_max	Maximum output voltage
in2_max_alarm	Output voltage high alarm
in2_min	Minimum output voltage
in2_min_alarm	Output voltage low alarm
power1_alarm	Input fault or alarm.
power1_input	Measured input power in uW.
power1_label	“pin”
power1_max	Input power limit
power2_max_alarm	Output power high alarm
power2_max	Output power limit
power2_input	Measured output power in uW.
power2_label	“pout”
temp[1-3]_input	Measured temperature
temp[1-2]_max	Maximum temperature
temp[1-3]_max_alarm	Temperature high alarm
vendor	Manufacturer name
model	Product model
part_number	Product part number
serial_number	Product serial number
fw_version	Firmware version
hw_version	Hardware version
mode	Work mode. Can be set to active or standby, when set to standby, PSU

CHAPTER

FOUR

HOW TO GET YOUR PATCH ACCEPTED INTO THE HWMON SUBSYSTEM

This text is a collection of suggestions for people writing patches or drivers for the hwmon subsystem. Following these suggestions will greatly increase the chances of your change being accepted.

4.1 1. General

- It should be unnecessary to mention, but please read and follow:
  - Documentation/process/submit-checklist.rst
  - Documentation/process/submitting-drivers.rst
  - Documentation/process/submitting-patches.rst
  - Documentation/process/coding-style.rst
- Please run your patch through 'checkpatch –strict'. There should be no errors, no warnings, and few if any check messages. If there are any messages, please be prepared to explain.
- Please use the standard multi-line comment style. Do not mix C and C++ style comments in a single driver (with the exception of the SPDX license identifier).
- If your patch generates checkpatch errors, warnings, or check messages, please refrain from explanations such as "I prefer that coding style". Keep in mind that each unnecessary message helps hiding a real problem, and a consistent coding style makes it easier for others to understand and review the code.
- Please test your patch thoroughly. We are not your test group. Sometimes a patch can not or not completely be tested because of missing hardware. In such cases, you should test-build the code on at least one architecture. If run-time testing was not achieved, it should be written explicitly below the patch header.
- If your patch (or the driver) is affected by configuration options such as CONFIG_SMP, make sure it compiles for all configuration variants.

4.2 2. Adding functionality to existing drivers

- Make sure the documentation in Documentation/hwmon/<driver_name>.rst is up to date.
- Make sure the information in Kconfig is up to date.
- If the added functionality requires some cleanup or structural changes, split your patch into a cleanup part and the actual addition. This makes it easier to review your changes, and to bisect any resulting problems.
- Never mix bug fixes, cleanup, and functional enhancements in a single patch.

4.3 3. New drivers

- Running your patch or driver file(s) through checkpatch does not mean its formatting is clean. If unsure about formatting in your new driver, run it through Lindent. Lindent is not perfect, and you may have to do some minor cleanup, but it is a good start.
- Consider adding yourself to MAINTAINERS.
- Document the driver in Documentation/hwmon/<driver_name>.rst.
- Add the driver to Kconfig and Makefile in alphabetical order.
- Make sure that all dependencies are listed in Kconfig.
- Please list include files in alphabetic order.
- Please align continuation lines with ‘(‘on the previous line.
- Avoid forward declarations if you can. Rearrange the code if necessary.
- Avoid macros to generate groups of sensor attributes. It not only confuses checkpatch, but also makes it more difficult to review the code.
- Avoid calculations in macros and macro-generated functions. While such macros may save a line or so in the source, it obfuscates the code and makes code review more difficult. It may also result in code which is more complicated than necessary. Use inline functions or just regular functions instead.
- Limit the number of kernel log messages. In general, your driver should not generate an error message just because a runtime operation failed. Report errors to user space instead, using an appropriate error code. Keep in mind that kernel error log messages not only fill up the kernel log, but also are printed synchronously, most likely with interrupt disabled, often to a serial console. Excessive logging can seriously affect system performance.
- Use devres functions whenever possible to allocate resources. For rationale and supported functions, please see Documentation/driver-api/driver-model/devres.rst. If a function is not supported by devres, consider using devm_add_action().
- If the driver has a detect function, make sure it is silent. Debug messages and messages printed after a successful detection are acceptable, but it must not print messages such as “Chip XXX not found/supported”.

Keep in mind that the detect function will run for all drivers supporting an address if a chip is detected on that address. Unnecessary messages will just pollute the kernel log and not provide any value.

- Provide a detect function if and only if a chip can be detected reliably.
- Only the following I2C addresses shall be probed: 0x18-0x1f, 0x28-0x2f, 0x48-0x4f, 0x58, 0x5c, 0x73 and 0x77. Probing other addresses is strongly discouraged as it is known to cause trouble with other (non-hwmon) I2C chips. If your chip lives at an address which can't be probed then the device will have to be instantiated explicitly (which is always better anyway.)
- Avoid writing to chip registers in the detect function. If you have to write, only do it after you have already gathered enough data to be certain that the detection is going to be successful.

  Keep in mind that the chip might not be what your driver believes it is, and writing to it might cause a bad misconfiguration.
- Make sure there are no race conditions in the probe function. Specifically, completely initialize your chip and your driver first, then register with the hwmon subsystem.
- Use devm_hwmon_device_register_with_info() or, if your driver needs a remove function, hwmon_device_register_with_info() to register your driver with the hwmon subsystem. Try using devm_add_action() instead of a remove function if possible. Do not use hwmon_device_register().
- Your driver should be buildable as module. If not, please be prepared to explain why it has to be built into the kernel.
- Do not provide support for deprecated sysfs attributes.
- Do not create non-standard attributes unless really needed. If you have to use non-standard attributes, or you believe you do, discuss it on the mailing list first. Either case, provide a detailed explanation why you need the non-standard attribute(s). Standard attributes are specified in Documentation/hwmon/sysfs-interface.rst.
- When deciding which sysfs attributes to support, look at the chip's capabilities. While we do not expect your driver to support everything the chip may offer, it should at least support all limits and alarms.
- Last but not least, please check if a driver for your chip already exists before starting to write a new driver. Especially for temperature sensors, new chips are often variants of previously released chips. In some cases, a presumably new chip may simply have been relabeled.

CHAPTER

FIVE

NAMING AND DATA FORMAT STANDARDS FOR SYSFS FILES

The libsensors library offers an interface to the raw sensors data through the sysfs interface. Since lm-sensors 3.0.0, libsensors is completely chip-independent. It assumes that all the kernel drivers implement the standard sysfs interface described in this document. This makes adding or updating support for any given chip very easy, as libsensors, and applications using it, do not need to be modified. This is a major improvement compared to lm-sensors 2.

Note that motherboards vary widely in the connections to sensor chips. There is no standard that ensures, for example, that the second temperature sensor is connected to the CPU, or that the second fan is on the CPU. Also, some values reported by the chips need some computation before they make full sense. For example, most chips can only measure voltages between 0 and +4V. Other voltages are scaled back into that range using external resistors. Since the values of these resistors can change from motherboard to motherboard, the conversions cannot be hard coded into the driver and have to be done in user space.

For this reason, even if we aim at a chip-independent libsensors, it will still require a configuration file (e.g. /etc/sensors.conf) for proper values conversion, labeling of inputs and hiding of unused inputs.

An alternative method that some programs use is to access the sysfs files directly. This document briefly describes the standards that the drivers follow, so that an application program can scan for entries and access this data in a simple and consistent way. That said, such programs will have to implement conversion, labeling and hiding of inputs. For this reason, it is still not recommended to bypass the library.

Each chip gets its own directory in the sysfs /sys/devices tree. To find all sensor chips, it is easier to follow the device symlinks from /sys/class/hwmon/hwmon*.

Up to lm-sensors 3.0.0, libsensors looks for hardware monitoring attributes in the “physical” device directory. Since lm-sensors 3.0.1, attributes found in the hwmon “class” device directory are also supported. Complex drivers (e.g. drivers for multifunction chips) may want to use this possibility to avoid namespace pollution. The only drawback will be that older versions of libsensors won’t support the driver in question.

All sysfs values are fixed point numbers.

There is only one value per file, unlike the older /proc specification. The common scheme for files naming is: <type><number>_<item>. Usual types for sensor

chips are “in” (voltage), “temp” (temperature) and “fan” (fan). Usual items are “input”(measured value), “max”(high threshold, “min”(low threshold). Numbering usually starts from 1, except for voltages which start from 0 (because most data sheets use this). A number is always used for elements that can be present more than once, even if there is a single element of the given type on the specific chip. Other files do not refer to a specific element, so they have a simple name, and no number.

Alarms are direct indications read from the chips. The drivers do NOT make comparisons of readings to thresholds. This allows violations between readings to be caught and alarmed. The exact definition of an alarm (for example, whether a threshold must be met or must be exceeded to cause an alarm) is chip-dependent.

When setting values of hwmon sysfs attributes, the string representation of the desired value must be written, note that strings which are not a number are interpreted as 0! For more on how written strings are interpreted see the “sysfs attribute writes interpretation” section at the end of this file.

[0-*]	denotes any positive number starting from 0
[1-*]	denotes any positive number starting from 1
RO	read only value
WO	write only value
RW	read/write value

Read/write values may be read-only for some chips, depending on the hardware implementation.

All entries (except name) are optional, and should only be created in a given driver if the chip has the feature.

5.1 Global attributes

name The chip name. This should be a short, lowercase string, not containing whitespace, dashes, or the wildcard character ‘*’ . This attribute represents the chip name. It is the only mandatory attribute. I2C devices get this attribute created automatically.

RO

update_interval The interval at which the chip will update readings. Unit: millisecond

RW

Some devices have a variable update rate or interval. This attribute can be used to change it to the desired value.

5.2 Voltages

in[0-*]_min Voltage min value.

Unit: millivolt

RW

in[0-*]_lcrit Voltage critical min value.

Unit: millivolt

RW

If voltage drops to or below this limit, the system may take drastic action such as power down or reset. At the very least, it should report a fault.

in[0-*]_max Voltage max value.

Unit: millivolt

RW

in[0-*]_crit Voltage critical max value.

Unit: millivolt

RW

If voltage reaches or exceeds this limit, the system may take drastic action such as power down or reset. At the very least, it should report a fault.

in[0-*]_input Voltage input value.

Unit: millivolt

RO

Voltage measured on the chip pin.

Actual voltage depends on the scaling resistors on the motherboard, as recommended in the chip datasheet.

This varies by chip and by motherboard. Because of this variation, values are generally NOT scaled by the chip driver, and must be done by the application. However, some drivers (notably lm87 and via686a) do scale, because of internal resistors built into a chip. These drivers will output the actual voltage. Rule of thumb: drivers should report the voltage values at the "pins" of the chip.

in[0-*]_average Average voltage

Unit: millivolt

RO

in[0-*]_lowest Historical minimum voltage

Unit: millivolt

RO

in[0-*]_highest Historical maximum voltage

Unit: millivolt

RO

in[0-*]_reset_history Reset inX_lowest and inX_highest

WO

in_reset_history Reset inX_lowest and inX_highest for all sensors

WO

in[0-*]_label Suggested voltage channel label.

Text string

Should only be created if the driver has hints about what this voltage channel is being used for, and user-space doesn't. In all other cases, the label is provided by user-space.

RO

in[0-*]_enable Enable or disable the sensors.

When disabled the sensor read will return -ENODATA.

- 1: Enable
- 0: Disable

RW

cpu[0-*]_vid CPU core reference voltage.

Unit: millivolt

RO

Not always correct.

vrm Voltage Regulator Module version number.

RW (but changing it should no more be necessary)

Originally the VRM standard version multiplied by 10, but now an arbitrary number, as not all standards have a version number.

Affects the way the driver calculates the CPU core reference voltage from the vid pins.

Also see the Alarms section for status flags associated with voltages.

5.3 Fans

fan[1-*]_min Fan minimum value

Unit: revolution/min (RPM)

RW

fan[1-*]_max Fan maximum value

Unit: revolution/min (RPM)

Only rarely supported by the hardware. RW

fan[1-*]_input Fan input value.

Unit: revolution/min (RPM)

RO

fan[1-*]_div Fan divisor.

Integer value in powers of two (1, 2, 4, 8, 16, 32, 64, 128).

RW

Some chips only support values 1, 2, 4 and 8. Note that this is actually an internal clock divisor, which affects the measurable speed range, not the read value.

fan[1-*]_pulses Number of tachometer pulses per fan revolution.

Integer value, typically between 1 and 4.

RW

This value is a characteristic of the fan connected to the device's input, so it has to be set in accordance with the fan model.

Should only be created if the chip has a register to configure the number of pulses. In the absence of such a register (and thus attribute) the value assumed by all devices is 2 pulses per fan revolution.

fan[1-*]_target Desired fan speed

Unit: revolution/min (RPM)

RW

Only makes sense if the chip supports closed-loop fan speed control based on the measured fan speed.

fan[1-*]_label Suggested fan channel label.

Text string

Should only be created if the driver has hints about what this fan channel is being used for, and user-space doesn't. In all other cases, the label is provided by user-space.

RO

fan[1-*]_enable Enable or disable the sensors.

When disabled the sensor read will return -ENODATA.

- 1: Enable
- 0: Disable

RW

Also see the Alarms section for status flags associated with fans.

5.4 PWM

pwm[1-*] Pulse width modulation fan control.

Integer value in the range 0 to 255

RW

255 is max or 100%.

pwm[1-*]_enable Fan speed control method:

- 0: no fan speed control (i.e. fan at full speed)
- 1: manual fan speed control enabled (using pwm[1-*])
- 2+: automatic fan speed control enabled

Check individual chip documentation files for automatic mode details.

RW

pwm[1-*]_mode

- 0: DC mode (direct current)
- 1: PWM mode (pulse-width modulation)

RW

pwm[1-*]_freq Base PWM frequency in Hz.

Only possibly available when pwmN_mode is PWM, but not always present even then.

RW

pwm[1-*]_auto_channels_temp Select which temperature channels affect this PWM output in auto mode.

Bitfield, 1 is temp1, 2 is temp2, 4 is temp3 etc⋯Which values are possible depend on the chip used.

RW

pwm[1-*]_auto_point[1-*]_pwm / pwm[1-*]_auto_point[1-*]_temp / pwm[1-*]_auto_poin
Define the PWM vs temperature curve.

Number of trip points is chip-dependent. Use this for chips which associate trip points to PWM output channels.

RW

temp[1-*]_auto_point[1-*]_pwm / temp[1-*]_auto_point[1-*]_temp / temp[1-*]_auto_poi
Define the PWM vs temperature curve.

Number of trip points is chip-dependent. Use this for chips which associate trip points to temperature channels.

RW

There is a third case where trip points are associated to both PWM output channels and temperature channels: the PWM values are associated to PWM output channels while the temperature values are associated to temperature channels. In that case, the result is determined by the mapping between temperature inputs and PWM outputs. When several temperature inputs are mapped to a given PWM output, this leads to several candidate PWM values. The actual result is up to the chip, but in general the highest candidate value (fastest fan speed) wins.

5.5 Temperatures

temp[1-*]_type Sensor type selection.

Integers 1 to 6

RW

- 1: CPU embedded diode
- 2: 3904 transistor
- 3: thermal diode
- 4: thermistor
- 5: AMD AMDSI
- 6: Intel PECI

Not all types are supported by all chips

temp[1-*]_max Temperature max value.

Unit: millidegree Celsius (or millivolt, see below)

RW

temp[1-*]_min Temperature min value.

Unit: millidegree Celsius

RW

temp[1-*]_max_hyst Temperature hysteresis value for max limit.

Unit: millidegree Celsius

Must be reported as an absolute temperature, NOT a delta from the max value.

RW

temp[1-*]_min_hyst Temperature hysteresis value for min limit. Unit: millidegree Celsius

Must be reported as an absolute temperature, NOT a delta from the min value.

RW

temp[1-*]_input Temperature input value.

Unit: millidegree Celsius

RO

temp[1-*]_crit Temperature critical max value, typically greater than corresponding temp_max values.

Unit: millidegree Celsius

RW

temp[1-*]_crit_hyst Temperature hysteresis value for critical limit.

Unit: millidegree Celsius

Must be reported as an absolute temperature, NOT a delta from the critical value.

RW

temp[1-*]_emergency Temperature emergency max value, for chips supporting more than two upper temperature limits. Must be equal or greater than corresponding temp_crit values.

Unit: millidegree Celsius

RW

temp[1-*]_emergency_hyst Temperature hysteresis value for emergency limit.

Unit: millidegree Celsius

Must be reported as an absolute temperature, NOT a delta from the emergency value.

RW

temp[1-*]_lcrit Temperature critical min value, typically lower than corresponding temp_min values.

Unit: millidegree Celsius

RW

temp[1-*]_lcrit_hyst Temperature hysteresis value for critical min limit.

Unit: millidegree Celsius

Must be reported as an absolute temperature, NOT a delta from the critical min value.

RW

temp[1-*]_offset Temperature offset which is added to the temperature reading by the chip.

Unit: millidegree Celsius

Read/Write value.

temp[1-*]_label Suggested temperature channel label.

Text string

Should only be created if the driver has hints about what this temperature channel is being used for, and user-space doesn't. In all other cases, the label is provided by user-space.

RO

temp[1-*]_lowest Historical minimum temperature

Unit: millidegree Celsius

RO

temp[1-*]_highest Historical maximum temperature

Unit: millidegree Celsius

RO

temp[1-*]_reset_history Reset temp_lowest and temp_highest

WO

temp_reset_history Reset temp_lowest and temp_highest for all sensors

WO

temp[1-*]_enable Enable or disable the sensors.

When disabled the sensor read will return -ENODATA.

- 1: Enable
- 0: Disable

RW

Some chips measure temperature using external thermistors and an ADC, and report the temperature measurement as a voltage. Converting this voltage back to a temperature (or the other way around for limits) requires mathematical functions not available in the kernel, so the conversion must occur in user space. For these chips, all temp* files described above should contain values expressed in millivolt instead of millidegree Celsius. In other words, such temperature channels are handled as voltage channels by the driver.

Also see the Alarms section for status flags associated with temperatures.

5.6 Currents

curr[1-*]_max Current max value

Unit: milliampere

RW

curr[1-*]_min Current min value.

Unit: milliampere

RW

curr[1-*]_lcrit Current critical low value

Unit: milliampere

RW

curr[1-*]_crit Current critical high value.

Unit: milliampere

RW

curr[1-*]_input Current input value

Unit: milliampere

RO

curr[1-*]_average Average current use

Unit: milliampere

RO

curr[1-*]_lowest Historical minimum current

Unit: milliampere

RO

curr[1-*]_highest Historical maximum current Unit: milliampere RO

curr[1-*]_reset_history Reset currX_lowest and currX_highest

WO

curr_reset_history Reset currX_lowest and currX_highest for all sensors

WO

curr[1-*]_enable Enable or disable the sensors.

When disabled the sensor read will return -ENODATA.

- 1: Enable
- 0: Disable

RW

Also see the Alarms section for status flags associated with currents.

5.7 Power

power[1-*]_average Average power use

Unit: microWatt

RO

power[1-*]_average_interval Power use averaging interval. A poll notification is sent to this file if the hardware changes the averaging interval.

Unit: milliseconds

RW

power[1-*]_average_interval_max Maximum power use averaging interval

Unit: milliseconds

RO

power[1-*]_average_interval_min Minimum power use averaging interval

Unit: milliseconds

RO

power[1-*]_average_highest Historical average maximum power use

Unit: microWatt

RO

power[1-*]_average_lowest Historical average minimum power use

Unit: microWatt

RO

power[1-*]_average_max A poll notification is sent to power[1-*]_average when power use rises above this value.

Unit: microWatt

RW

power[1-*]_average_min A poll notification is sent to power[1-*]_average when power use sinks below this value.

Unit: microWatt

RW

power[1-*]_input Instantaneous power use

Unit: microWatt

RO

power[1-*]_input_highest Historical maximum power use

Unit: microWatt

RO

power[1-*]_input_lowest Historical minimum power use

Unit: microWatt

RO

power[1-*]_reset_history Reset input_highest, input_lowest, average_highest and average_lowest.

WO

power[1-*]_accuracy Accuracy of the power meter.

Unit: Percent

RO

power[1-*]_cap If power use rises above this limit, the system should take action to reduce power use. A poll notification is sent to this file if the cap is changed by the hardware. The *_cap files only appear if the cap is known to be enforced by hardware.

Unit: microWatt

RW

power[1-*]_cap_hyst Margin of hysteresis built around capping and notification.

Unit: microWatt

RW

power[1-*]_cap_max Maximum cap that can be set.

Unit: microWatt

RO

power[1-*]_cap_min Minimum cap that can be set.

Unit: microWatt

RO

power[1-*]_max Maximum power.

Unit: microWatt

RW

power[1-*]_crit Critical maximum power.

If power rises to or above this limit, the system is expected take drastic action to reduce power consumption, such as a system shutdown or a forced powerdown of some devices.

Unit: microWatt

RW

power[1-*]_enable Enable or disable the sensors.

When disabled the sensor read will return -ENODATA.

- 1: Enable

- 0: Disable

RW

Also see the Alarms section for status flags associated with power readings.

5.8 Energy

energy[1-*]_input Cumulative energy use

Unit: microJoule

RO

energy[1-*]_enable Enable or disable the sensors.

When disabled the sensor read will return -ENODATA.

- 1: Enable
- 0: Disable

RW

5.9 Humidity

humidity[1-*]_input Humidity

Unit: milli-percent (per cent mille, pcm)

RO

humidity[1-*]_enable Enable or disable the sensors

When disabled the sensor read will return -ENODATA.

- 1: Enable
- 0: Disable

RW

5.10 Alarms

Each channel or limit may have an associated alarm file, containing a boolean value. 1 means than an alarm condition exists, 0 means no alarm.

Usually a given chip will either use channel-related alarms, or limit-related alarms, not both. The driver should just reflect the hardware implementation.

`in[0-*]_alarm`, `curr[1-*]_alarm`, `power[1-*]_alarm`, `fan[1-*]_alarm`, `temp[1-*]_alarm`	Channel alarm • 0: no alarm • 1: alarm RO

OR

`in[0-*]_min_alarm`, `in[0-*]_max_alarm`, `in[0-*]_lcrit_alarm`, `in[0-*]_crit_alarm`, `curr[1-*]_min_alarm`, `curr[1-*]_max_alarm`, `curr[1-*]_lcrit_alarm`, `curr[1-*]_crit_alarm`, `power[1-*]_cap_alarm`, `power[1-*]_max_alarm`, `power[1-*]_crit_alarm`, `fan[1-*]_min_alarm`, `fan[1-*]_max_alarm`, `temp[1-*]_min_alarm`, `temp[1-*]_max_alarm`, `temp[1-*]_lcrit_alarm`, `temp[1-*]_crit_alarm`, `temp[1-*]_emergency_alarm`	Limit alarm • 0: no alarm • 1: alarm RO

Each input channel may have an associated fault file. This can be used to notify open diodes, unconnected fans etc. where the hardware supports it. When this boolean has value 1, the measurement for that channel should not be trusted.

fan[1-*]_fault / temp[1-*]_fault Input fault condition

- 0: no fault occurred
- 1: fault condition

RO

Some chips also offer the possibility to get beeped when an alarm occurs:

beep_enable Master beep enable

- 0: no beeps
- 1: beeps

RW

in[0-*]_beep, curr[1-*]_beep, fan[1-*]_beep, temp[1-*]_beep, Channel beep

- 0: disable
- 1: enable

RW

In theory, a chip could provide per-limit beep masking, but no such chip was seen so far.

Old drivers provided a different, non-standard interface to alarms and beeps. These interface files are deprecated, but will be kept around for compatibility reasons:

alarms Alarm bitmask.

RO

Integer representation of one to four bytes.

A ‘1’ bit means an alarm.

Chips should be programmed for ‘comparator’ mode so that the alarm will ‘come back’ after you read the register if it is still valid.

Generally a direct representation of a chip’s internal alarm registers; there is no standard for the position of individual bits. For this reason, the use of this interface file for new drivers is discouraged. Use individual *_alarm and *_fault files instead. Bits are defined in kernel/include/sensors.h.

beep_mask Bitmask for beep. Same format as ‘alarms’with the same bit locations, use discouraged for the same reason. Use individual *_beep files instead. RW

5.11 Intrusion detection

intrusion[0-*]_alarm Chassis intrusion detection

- 0: OK
- 1: intrusion detected

RW

Contrary to regular alarm flags which clear themselves automatically when read, this one sticks until cleared by the user. This is done by writing 0 to the file. Writing other values is unsupported.

intrusion[0-*]_beep Chassis intrusion beep

0: disable 1: enable

RW

5.12 Average sample configuration

Devices allowing for reading {in,power,curr,temp}_average values may export attributes for controlling number of samples used to compute average.

samples	Sets number of average samples for all types of measurements. RW
in_samples power_samples curr_samples temp_samples	Sets number of average samples for specific type of measurements. Note that on some devices it won’t be possible to set all of them to different values so changing one might also change some others. RW

5.12.1 sysfs attribute writes interpretation

hwmon sysfs attributes always contain numbers, so the first thing to do is to convert the input to a number, there are 2 ways todo this depending whether the number can be negative or not:

```
unsigned long u = simple_strtoul(buf, NULL, 10);
long s = simple_strtol(buf, NULL, 10);
```

With buf being the buffer with the user input being passed by the kernel. Notice that we do not use the second argument of strto[u]l, and thus cannot tell when 0 is returned, if this was really 0 or is caused by invalid input. This is done deliberately as checking this everywhere would add a lot of code to the kernel.

Notice that it is important to always store the converted value in an unsigned long or long, so that no wrap around can happen before any further checking.

After the input string is converted to an (unsigned) long, the value should be checked if its acceptable. Be careful with further conversions on the value before checking it for validity, as these conversions could still cause a wrap around before the check. For example do not multiply the result, and only add/subtract if it has been divided before the add/subtract.

What to do if a value is found to be invalid, depends on the type of the sysfs attribute that is being set. If it is a continuous setting like a tempX_max or inX_max attribute, then the value should be clamped to its limits using clamp_val(value, min_limit, max_limit). If it is not continuous like for example a tempX_type, then when an invalid value is written, -EINVAL should be returned.

Example1, temp1_max, register is a signed 8 bit value (-128 - 127 degrees):

```
long v = simple_strtol(buf, NULL, 10) / 1000;
v = clamp_val(v, -128, 127);
/* write v to register */
```

Example2, fan divider setting, valid values 2, 4 and 8:

```
unsigned long v = simple_strtoul(buf, NULL, 10);

switch (v) {
case 2: v = 1; break;
case 4: v = 2; break;
case 8: v = 3; break;
default:
        return -EINVAL;
}
/* write v to register */
```

CHAPTER

SIX

USERSPACE TOOLS

6.1 Introduction

Most mainboards have sensor chips to monitor system health (like temperatures, voltages, fans speed). They are often connected through an I2C bus, but some are also connected directly through the ISA bus.

The kernel drivers make the data from the sensor chips available in the /sys virtual filesystem. Userspace tools are then used to display the measured values or configure the chips in a more friendly manner.

6.2 Lm-sensors

Core set of utilities that will allow you to obtain health information, setup monitoring limits etc. You can get them on their homepage http://www.lm-sensors.org/ or as a package from your Linux distribution.

If from website: Get lm-sensors from project web site. Please note, you need only userspace part, so compile with "make user" and install with "make user_install" .

General hints to get things working:

0) get lm-sensors userspace utils
1) compile all drivers in I2C and Hardware Monitoring sections as modules in your kernel
2) run sensors-detect script, it will tell you what modules you need to load.
3) load them and run "sensors" command, you should see some results.
4) fix sensors.conf, labels, limits, fan divisors
5) if any more problems consult FAQ, or documentation

6.3 Other utilities

If you want some graphical indicators of system health look for applications like: gkrellm, ksensors, xsensors, wmtemp, wmsensors, wmgtemp, ksysguardd, hardware-monitor

If you are server administrator you can try snmpd or mrtgutils.

CHAPTER

SEVEN

HARDWARE MONITORING KERNEL DRIVERS

7.1 Kernel driver ab8500

Supported chips:

- ST-Ericsson AB8500

  Prefix: 'ab8500'

  Addresses scanned: -

  Datasheet: http://www.stericsson.com/developers/documentation.jsp

Authors:

- Martin Persson <martin.persson@stericsson.com>
- Hongbo Zhang <hongbo.zhang@linaro.org>

7.1.1 Description

See also Documentation/hwmon/abx500.rst. This is the ST-Ericsson AB8500 specific driver.

Currently only the AB8500 internal sensor and one external sensor for battery temperature are monitored. Other GPADC channels can also be monitored if needed in future.

7.2 Kernel driver abituguru

Supported chips:

- Abit uGuru revision 1 & 2 (Hardware Monitor part only)

  Prefix: 'abituguru'

  Addresses scanned: ISA 0x0E0

  Datasheet: Not available, this driver is based on reverse engineering. A "Datasheet" has been written based on the reverse engineering it should be available in the same dir as this file under the name abituguru-datasheet.

Note: The uGuru is a microcontroller with onboard firmware which programs it to behave as a hwmon IC. There are many different revisions of the firmware and thus effectivly many different revisions of the uGuru. Below is an incomplete list with which revisions are used for which Motherboards:

- uGuru 1.00 ~ 1.24 (AI7, KV8-MAX3, AN7)[1]
- uGuru 2.0.0.0 ~ 2.0.4.2 (KV8-PRO)
- uGuru 2.1.0.0 ~ 2.1.2.8 (AS8, AV8, AA8, AG8, AA8XE, AX8)
- uGuru 2.2.0.0 ~ 2.2.0.6 (AA8 Fatal1ty)
- uGuru 2.3.0.0 ~ 2.3.0.9 (AN8)
- uGuru 3.0.0.0 ~ 3.0.x.x (AW8, AL8, AT8, NI8 SLI, AT8 32X, AN8 32X, AW9D-MAX)[2]

Authors:

- Hans de Goede <j.w.r.degoede@hhs.nl>,
- (Initial reverse engineering done by Olle Sandberg <ollebull@gmail.com>)

7.2.1 Module Parameters

- **force: bool** Force detection. Note this parameter only causes the detection to be skipped, and thus the insmod to succeed. If the uGuru can't be read the actual hwmon driver will not load and thus no hwmon device will get registered.
- **bank1_types: int[]** Bank1 sensortype autodetection override:
  - -1 autodetect (default)
  - 0 volt sensor
  - 1 temp sensor
  - 2 not connected
- **fan_sensors: int** Tell the driver how many fan speed sensors there are on your motherboard. Default: 0 (autodetect).
- **pwms: int** Tell the driver how many fan speed controls (fan pwms) your motherboard has. Default: 0 (autodetect).
- **verbose: int** How verbose should the driver be? (0-3):
  - 0 normal output

[1] For revisions 2 and 3 uGuru's the driver can autodetect the sensortype (Volt or Temp) for bank1 sensors, for revision 1 uGuru's this does not always work. For these uGuru's the autodetection can be overridden with the bank1_types module param. For all 3 known revison 1 motherboards the correct use of this param is: bank1_types=1,1,0,0,0,0,0,2,0,0,0,0,2,0,0,1 You may also need to specify the fan_sensors option for these boards fan_sensors=5

[2] There is a separate abituguru3 driver for these motherboards, the abituguru (without the 3 !) driver will not work on these motherboards (and visa versa)!

- 1 + verbose error reporting
- 2 + sensors type probing info (default)
- 3 + retryable error reporting

Default: 2 (the driver is still in the testing phase)

Notice: if you need any of the first three options above please insmod the driver with verbose set to 3 and mail me <j.w.r.degoede@hhs.nl> the output of: dmesg | grep abituguru

7.2.2 Description

This driver supports the hardware monitoring features of the first and second revision of the Abit uGuru chip found on Abit uGuru featuring motherboards (most modern Abit motherboards).

The first and second revision of the uGuru chip in reality is a Winbond W83L950D in disguise (despite Abit claiming it is "a new microprocessor designed by the ABIT Engineers"). Unfortunately this doesn't help since the W83L950D is a generic microcontroller with a custom Abit application running on it.

Despite Abit not releasing any information regarding the uGuru, Olle Sandberg <ollebull@gmail.com> has managed to reverse engineer the sensor part of the uGuru. Without his work this driver would not have been possible.

7.2.3 Known Issues

The voltage and frequency control parts of the Abit uGuru are not supported.

uGuru datasheet

First of all, what I know about uGuru is no fact based on any help, hints or datasheet from Abit. The data I have got on uGuru have I assembled through my weak knowledge in "backwards engineering". And just for the record, you may have noticed uGuru isn't a chip developed by Abit, as they claim it to be. It's really just an microprocessor (uC) created by Winbond (W83L950D). And no, reading the manual for this specific uC or mailing Windbond for help won't give any useful data about uGuru, as it is the program inside the uC that is responding to calls.

Olle Sandberg <ollebull@gmail.com>, 2005-05-25

Original version by Olle Sandberg who did the heavy lifting of the initial reverse engineering. This version has been almost fully rewritten for clarity and extended with write support and info on more databanks, the write support is once again reverse engineered by Olle the additional databanks have been reverse engineered by me. I would like to express my thanks to Olle, this document and the Linux driver could not have been written without his efforts.

Note: because of the lack of specs only the sensors part of the uGuru is described here and not the CPU / RAM / etc voltage & frequency control.

Hans de Goede <j.w.r.degoede@hhs.nl>, 28-01-2006

Detection

As far as known the uGuru is always placed at and using the (ISA) I/O-ports 0xE0 and 0xE4, so we don't have to scan any port-range, just check what the two ports are holding for detection. We will refer to 0xE0 as CMD (command-port) and 0xE4 as DATA because Abit refers to them with these names.

If DATA holds 0x00 or 0x08 and CMD holds 0x00 or 0xAC an uGuru could be present. We have to check for two different values at data-port, because after a reboot uGuru will hold 0x00 here, but if the driver is removed and later on attached again data-port will hold 0x08, more about this later.

After wider testing of the Linux kernel driver some variants of the uGuru have turned up which will hold 0x00 instead of 0xAC at the CMD port, thus we also have to test CMD for two different values. On these uGuru's DATA will initially hold 0x09 and will only hold 0x08 after reading CMD first, so CMD must be read first!

To be really sure an uGuru is present a test read of one or more register sets should be done.

Reading / Writing

Addressing

The uGuru has a number of different addressing levels. The first addressing level we will call banks. A bank holds data for one or more sensors. The data in a bank for a sensor is one or more bytes large.

The number of bytes is fixed for a given bank, you should always read or write that many bytes, reading / writing more will fail, the results when writing less then the number of bytes for a given bank are undetermined.

See below for all known bank addresses, numbers of sensors in that bank, number of bytes data per sensor and contents/meaning of those bytes.

Although both this document and the kernel driver have kept the sensor terminoligy for the addressing within a bank this is not 100% correct, in bank 0x24 for example the addressing within the bank selects a PWM output not a sensor.

Notice that some banks have both a read and a write address this is how the uGuru determines if a read from or a write to the bank is taking place, thus when reading you should always use the read address and when writing the write address. The write address is always one (1) more than the read address.

uGuru ready

Before you can read from or write to the uGuru you must first put the uGuru in "ready" mode.

To put the uGuru in ready mode first write 0x00 to DATA and then wait for DATA to hold 0x09, DATA should read 0x09 within 250 read cycles.

Next CMD _must_ be read and should hold 0xAC, usually CMD will hold 0xAC the first read but sometimes it takes a while before CMD holds 0xAC and thus it has to be read a number of times (max 50).

After reading CMD, DATA should hold 0x08 which means that the uGuru is ready for input. As above DATA will usually hold 0x08 the first read but not always. This step can be skipped, but it is undetermined what happens if the uGuru has not yet reported 0x08 at DATA and you proceed with writing a bank address.

Sending bank and sensor addresses to the uGuru

First the uGuru must be in "ready" mode as described above, DATA should hold 0x08 indicating that the uGuru wants input, in this case the bank address.

Next write the bank address to DATA. After the bank address has been written wait for to DATA to hold 0x08 again indicating that it wants / is ready for more input (max 250 reads).

Once DATA holds 0x08 again write the sensor address to CMD.

Reading

First send the bank and sensor addresses as described above. Then for each byte of data you want to read wait for DATA to hold 0x01 which indicates that the uGuru is ready to be read (max 250 reads) and once DATA holds 0x01 read the byte from CMD.

Once all bytes have been read data will hold 0x09, but there is no reason to test for this. Notice that the number of bytes is bank address dependent see above and below.

After completing a successful read it is advised to put the uGuru back in ready mode, so that it is ready for the next read / write cycle. This way if your program / driver is unloaded and later loaded again the detection algorithm described above will still work.

Writing

First send the bank and sensor addresses as described above. Then for each byte of data you want to write wait for DATA to hold 0x00 which indicates that the uGuru is ready to be written (max 250 reads) and once DATA holds 0x00 write the byte to CMD.

Once all bytes have been written wait for DATA to hold 0x01 (max 250 reads) don't ask why this is the way it is.

Once DATA holds 0x01 read CMD it should hold 0xAC now.

After completing a successful write it is advised to put the uGuru back in ready mode, so that it is ready for the next read / write cycle. This way if your program / driver is unloaded and later loaded again the detection algorithm described above will still work.

Gotchas

After wider testing of the Linux kernel driver some variants of the uGuru have turned up which do not hold 0x08 at DATA within 250 reads after writing the bank address. With these versions this happens quite frequent, using larger timeouts doesn't help, they just go offline for a second or 2, doing some internal callibration or whatever. Your code should be prepared to handle this and in case of no response in this specific case just goto sleep for a while and then retry.

Address Map

Bank 0x20 Alarms (R)

This bank contains 0 sensors, iow the sensor address is ignored (but must be written) just use 0. Bank 0x20 contains 3 bytes:

Byte 0: This byte holds the alarm flags for sensor 0-7 of Sensor Bank1, with bit 0 corresponding to sensor 0, 1 to 1, etc.

Byte 1: This byte holds the alarm flags for sensor 8-15 of Sensor Bank1, with bit 0 corresponding to sensor 8, 1 to 9, etc.

Byte 2: This byte holds the alarm flags for sensor 0-5 of Sensor Bank2, with bit 0 corresponding to sensor 0, 1 to 1, etc.

Bank 0x21 Sensor Bank1 Values / Readings (R)

This bank contains 16 sensors, for each sensor it contains 1 byte. So far the following sensors are known to be available on all motherboards:

- Sensor 0 CPU temp
- Sensor 1 SYS temp
- Sensor 3 CPU core volt
- Sensor 4 DDR volt
- Sensor 10 DDR Vtt volt
- Sensor 15 PWM temp

Byte 0: This byte holds the reading from the sensor. Sensors in Bank1 can be both volt and temp sensors, this is motherboard specific. The uGuru however does seem to know (be programmed with) what kindoff sensor is attached see Sensor Bank1 Settings description.

Volt sensors use a linear scale, a reading 0 corresponds with 0 volt and a reading of 255 with 3494 mV. The sensors for higher voltages however are connected through a division circuit. The currently known division circuits in use result in ranges of: 0-4361mV, 0-6248mV or 0-14510mV. 3.3 volt sources use the 0-4361mV range, 5 volt the 0-6248mV and 12 volt the 0-14510mV .

Temp sensors also use a linear scale, a reading of 0 corresponds with 0 degree Celsius and a reading of 255 with a reading of 255 degrees Celsius.

Bank 0x22 Sensor Bank1 Settings (R) and Bank 0x23 Sensor Bank1 Settings (W)

Those banks contain 16 sensors, for each sensor it contains 3 bytes. Each set of 3 bytes contains the settings for the sensor with the same sensor address in Bank 0x21 .

Byte 0: Alarm behaviour for the selected sensor. A 1 enables the described behaviour.

Bit 0: Give an alarm if measured temp is over the warning threshold (RW)[1]

Bit 1: Give an alarm if measured volt is over the max threshold (RW)[2]

Bit 2: Give an alarm if measured volt is under the min threshold (RW)[2]

Bit 3: Beep if alarm (RW)

Bit 4: 1 if alarm cause measured temp is over the warning threshold (R)

Bit 5: 1 if alarm cause measured volt is over the max threshold (R)

Bit 6: 1 if alarm cause measured volt is under the min threshold (R)

[1] This bit is only honored/used by the uGuru if a temp sensor is connected

[2] This bit is only honored/used by the uGuru if a volt sensor is connected Note with some trickery this can be used to find out what kinda sensor is detected see the Linux kernel driver for an example with many comments on how todo this.

Bit 7:

- Volt sensor: Shutdown if alarm persist for more than 4 seconds (RW)
- Temp sensor: Shutdown if temp is over the shutdown threshold (RW)

Byte 1:

- Temp sensor: warning threshold (scale as bank 0x21)
- Volt sensor: min threshold (scale as bank 0x21)

Byte 2:

- Temp sensor: shutdown threshold (scale as bank 0x21)
- Volt sensor: max threshold (scale as bank 0x21)

Bank 0x24 PWM outputs for FAN’s (R) and Bank 0x25 PWM outputs for FAN’s (W)

Those banks contain 3 “sensors”, for each sensor it contains 5 bytes.

- Sensor 0 usually controls the CPU fan
- Sensor 1 usually controls the NB (or chipset for single chip) fan
- Sensor 2 usually controls the System fan

Byte 0: Flag 0x80 to enable control, Fan runs at 100% when disabled. low nibble (temp)sensor address at bank 0x21 used for control.

Byte 1: 0-255 = 0-12v (linear), specify voltage at which fan will rotate when under low threshold temp (specified in byte 3)

Byte 2: 0-255 = 0-12v (linear), specify voltage at which fan will rotate when above high threshold temp (specified in byte 4)

Byte 3: Low threshold temp (scale as bank 0x21)

byte 4: High threshold temp (scale as bank 0x21)

Bank 0x26 Sensors Bank2 Values / Readings (R)

This bank contains 6 sensors (AFAIK), for each sensor it contains 1 byte.

So far the following sensors are known to be available on all motherboards:

- Sensor 0: CPU fan speed
- Sensor 1: NB (or chipset for single chip) fan speed
- Sensor 2: SYS fan speed

Byte 0: This byte holds the reading from the sensor. 0-255 = 0-15300 (linear)

Bank 0x27 Sensors Bank2 Settings (R) and Bank 0x28 Sensors Bank2 Settings (W)

Those banks contain 6 sensors (AFAIK), for each sensor it contains 2 bytes.

Byte 0: Alarm behaviour for the selected sensor. A 1 enables the described behaviour.

Bit 0: Give an alarm if measured rpm is under the min threshold (RW)

Bit 3: Beep if alarm (RW)

Bit 7: Shutdown if alarm persist for more than 4 seconds (RW)

Byte 1: min threshold (scale as bank 0x26)

Warning for the adventurous

A word of caution to those who want to experiment and see if they can figure the voltage / clock programming out, I tried reading and only reading banks 0-0x30 with the reading code used for the sensor banks (0x20-0x28) and this resulted in a _permanent_ reprogramming of the voltages, luckily I had the sensors part configured so that it would shutdown my system on any out of spec voltages which proprably safed my computer (after a reboot I managed to immediately enter the bios and reload the defaults). This probably means that the read/write cycle for the non sensor part is different from the sensor part.

7.3 Kernel driver abituguru3

Supported chips:

- Abit uGuru revision 3 (Hardware Monitor part, reading only)

  Prefix: 'abituguru3'

  Addresses scanned: ISA 0x0E0

  Datasheet: Not available, this driver is based on reverse engineering.

  Note: The uGuru is a microcontroller with onboard firmware which programs it to behave as a hwmon IC. There are many different revisions of the firmware and thus effectivly many different revisions of the uGuru. Below is an incomplete list with which revisions are used for which Motherboards:

  - uGuru 1.00 ~ 1.24 (AI7, KV8-MAX3, AN7)
  - uGuru 2.0.0.0 ~ 2.0.4.2 (KV8-PRO)
  - uGuru 2.1.0.0 ~ 2.1.2.8 (AS8, AV8, AA8, AG8, AA8XE, AX8)
  - uGuru 2.3.0.0 ~ 2.3.0.9 (AN8)
  - uGuru 3.0.0.0 ~ 3.0.x.x (AW8, AL8, AT8, NI8 SLI, AT8 32X, AN8 32X, AW9D-MAX)

The abituguru3 driver is only for revison 3.0.x.x motherboards, this driver will not work on older motherboards. For older motherboards use the abituguru (without the 3 !) driver.

Authors:

- Hans de Goede <j.w.r.degoede@hhs.nl>,
- (Initial reverse engineering done by Louis Kruger)

7.3.1 Module Parameters

- **force: bool** Force detection. Note this parameter only causes the detection to be skipped, and thus the insmod to succeed. If the uGuru can't be read the actual hwmon driver will not load and thus no hwmon device will get registered.
- **verbose: bool** Should the driver be verbose?
  - 0/off/false normal output
  - 1/on/true + verbose error reporting (default)

  Default: 1 (the driver is still in the testing phase)

7.3.2 Description

This driver supports the hardware monitoring features of the third revision of the Abit uGuru chip, found on recent Abit uGuru featuring motherboards.

The 3rd revision of the uGuru chip in reality is a Winbond W83L951G. Unfortunately this doesn't help since the W83L951G is a generic microcontroller with a custom Abit application running on it.

Despite Abit not releasing any information regarding the uGuru revision 3, Louis Kruger has managed to reverse engineer the sensor part of the uGuru. Without his work this driver would not have been possible.

7.3.3 Known Issues

The voltage and frequency control parts of the Abit uGuru are not supported, neither is writing any of the sensor settings and writing / reading the fanspeed control registers (FanEQ)

If you encounter any problems please mail me <j.w.r.degoede@hhs.nl> and include the output of: dmesg | grep abituguru

7.4 Kernel driver abx500

Supported chips:

- ST-Ericsson ABx500 series

  Prefix: 'abx500'

  Addresses scanned: -

  Datasheet: http://www.stericsson.com/developers/documentation.jsp

Authors: Martin Persson <martin.persson@stericsson.com> Hongbo Zhang <hongbo.zhang@linaro.org>

7.4.1 Description

Every ST-Ericsson Ux500 SOC consists of both ABx500 and DBx500 physically, this is kernel hwmon driver for ABx500.

There are some GPADCs inside ABx500 which are designed for connecting to thermal sensors, and there is also a thermal sensor inside ABx500 too, which raises interrupt when critical temperature reached.

This abx500 is a common layer which can monitor all of the sensors, every specific abx500 chip has its special configurations in its own file, e.g. some sensors can be configured invisible if they are not available on that chip, and the corresponding gpadc_addr should be set to 0, thus this sensor won't be polled.

7.5 Kernel driver power_meter

This driver talks to ACPI 4.0 power meters.

Supported systems:

- Any recent system with ACPI 4.0.

  Prefix: 'power_meter'

  Datasheet: http://acpi.info/, section 10.4.

Author: Darrick J. Wong

7.5.1 Description

This driver implements sensor reading support for the power meters exposed in the ACPI 4.0 spec (Chapter 10.4). These devices have a simple set of features–a power meter that returns average power use over a configurable interval, an optional capping mechanism, and a couple of trip points. The sysfs interface conforms with the specification outlined in the "Power" section of Documentation/hwmon/sysfs-interface.rst.

7.5.2 Special Features

The power[1-*]_is_battery knob indicates if the power supply is a battery. Both power[1-*]_average_{min,max} must be set before the trip points will work. When both of them are set, an ACPI event will be broadcast on the ACPI netlink socket and a poll notification will be sent to the appropriate power[1-*]_average sysfs file.

The power[1-*]_{model_number, serial_number, oem_info} fields display arbitrary strings that ACPI provides with the meter. The measures/ directory contains symlinks to the devices that this meter measures.

Some computers have the ability to enforce a power cap in hardware. If this is the case, the power[1-*]_cap and related sysfs files will appear. When the average power consumption exceeds the cap, an ACPI event will be broadcast on the netlink event socket and a poll notification will be sent to the appropriate power[1-*]_alarm file to indicate that capping has begun, and the hardware has taken action to reduce power consumption. Most likely this will result in reduced performance.

There are a few other ACPI notifications that can be sent by the firmware. In all cases the ACPI event will be broadcast on the ACPI netlink event socket as well as sent as a poll notification to a sysfs file. The events are as follows:

power[1-*]_cap will be notified if the firmware changes the power cap. power[1-*]_interval will be notified if the firmware changes the averaging interval.

7.6 Kernel driver ad7314

Supported chips:

- Analog Devices AD7314

  Prefix: ‘ad7314’

  Datasheet: Publicly available at Analog Devices website.

- Analog Devices ADT7301

  Prefix: ‘adt7301’

  Datasheet: Publicly available at Analog Devices website.

- Analog Devices ADT7302

  Prefix: ‘adt7302’

  Datasheet: Publicly available at Analog Devices website.

7.6.1 Description

Driver supports the above parts. The ad7314 has a 10 bit sensor with 1lsb = 0.25 degrees centigrade. The adt7301 and adt7302 have 14 bit sensors with 1lsb = 0.03125 degrees centigrade.

7.6.2 Notes

Currently power down mode is not supported.

7.7 Kernel driver adc128d818

Supported chips:

- Texas Instruments ADC818D818

  Prefix: 'adc818d818'

  Addresses scanned: I2C 0x1d, 0x1e, 0x1f, 0x2d, 0x2e, 0x2f

  Datasheet: Publicly available at the TI website http://www.ti.com/

Author: Guenter Roeck

7.7.1 Description

This driver implements support for the Texas Instruments ADC128D818. It is described as 'ADC System Monitor with Temperature Sensor'.

The ADC128D818 implements one temperature sensor and seven voltage sensors.

Temperatures are measured in degrees Celsius. There is one set of limits. When the HOT Temperature Limit is crossed, this will cause an alarm that will be reasserted until the temperature drops below the HOT Hysteresis. Measurements are guaranteed between -55 and +125 degrees. The temperature measurement has a resolution of 0.5 degrees; the limits have a resolution of 1 degree.

Voltage sensors (also known as IN sensors) report their values in volts. An alarm is triggered if the voltage has crossed a programmable minimum or maximum limit. Note that minimum in this case always means 'closest to zero'; this is important for negative voltage measurements. All voltage inputs can measure voltages between 0 and 2.55 volts, with a resolution of 0.625 mV.

If an alarm triggers, it will remain triggered until the hardware register is read at least once. This means that the cause for the alarm may already have disappeared by the time the alarm is read. The driver caches the alarm status for each sensor until it is at least reported once, to ensure that alarms are reported to user space.

The ADC128D818 only updates its values approximately once per second; reading it more often will do no harm, but will return 'old' values.

In addition to the scanned address list, the chip can also be configured for addresses 0x35 to 0x37. Those addresses are not scanned. You have to instantiate

the driver explicitly if the chip is configured for any of those addresses in your system.

7.8 Kernel driver adm1021

Supported chips:

- Analog Devices ADM1021

  Prefix: 'adm1021'

  Addresses scanned: I2C 0x18 - 0x1a, 0x29 - 0x2b, 0x4c - 0x4e

  Datasheet: Publicly available at the Analog Devices website

- Analog Devices ADM1021A/ADM1023

  Prefix: 'adm1023'

  Addresses scanned: I2C 0x18 - 0x1a, 0x29 - 0x2b, 0x4c - 0x4e

  Datasheet: Publicly available at the Analog Devices website

- Genesys Logic GL523SM

  Prefix: 'gl523sm'

  Addresses scanned: I2C 0x18 - 0x1a, 0x29 - 0x2b, 0x4c - 0x4e

  Datasheet:

- Maxim MAX1617

  Prefix: 'max1617'

  Addresses scanned: I2C 0x18 - 0x1a, 0x29 - 0x2b, 0x4c - 0x4e

  Datasheet: Publicly available at the Maxim website

- Maxim MAX1617A

  Prefix: 'max1617a'

  Addresses scanned: I2C 0x18 - 0x1a, 0x29 - 0x2b, 0x4c - 0x4e

  Datasheet: Publicly available at the Maxim website

- National Semiconductor LM84

  Prefix: 'lm84'

  Addresses scanned: I2C 0x18 - 0x1a, 0x29 - 0x2b, 0x4c - 0x4e

  Datasheet: Publicly available at the National Semiconductor website

- Philips NE1617

  Prefix: 'max1617' (probably detected as a max1617)

  Addresses scanned: I2C 0x18 - 0x1a, 0x29 - 0x2b, 0x4c - 0x4e

  Datasheet: Publicly available at the Philips website

- Philips NE1617A

  Prefix: ‘max1617’ (probably detected as a max1617)

  Addresses scanned: I2C 0x18 - 0x1a, 0x29 - 0x2b, 0x4c - 0x4e

  Datasheet: Publicly available at the Philips website

- TI THMC10

  Prefix: ‘thmc10’

  Addresses scanned: I2C 0x18 - 0x1a, 0x29 - 0x2b, 0x4c - 0x4e

  Datasheet: Publicly available at the TI website

- Onsemi MC1066

  Prefix: ‘mc1066’

  Addresses scanned: I2C 0x18 - 0x1a, 0x29 - 0x2b, 0x4c - 0x4e

  Datasheet: Publicly available at the Onsemi website

Authors:

- Frodo Looijaard <frodol@dds.nl>,
- Philip Edelbrock <phil@netroedge.com>

7.8.1 Module Parameters

- read_only: int Don’t set any values, read only mode

7.8.2 Description

The chips supported by this driver are very similar. The Maxim MAX1617 is the oldest; it has the problem that it is not very well detectable. The MAX1617A solves that. The ADM1021 is a straight clone of the MAX1617A. Ditto for the THMC10. From here on, we will refer to all these chips as ADM1021-clones.

The ADM1021 and MAX1617A reports a die code, which is a sort of revision code. This can help us pinpoint problems; it is not very useful otherwise.

ADM1021-clones implement two temperature sensors. One of them is internal, and measures the temperature of the chip itself; the other is external and is realised in the form of a transistor-like device. A special alarm indicates whether the remote sensor is connected.

Each sensor has its own low and high limits. When they are crossed, the corresponding alarm is set and remains on as long as the temperature stays out of range. Temperatures are measured in degrees Celsius. Measurements are possible between -65 and +127 degrees, with a resolution of one degree.

If an alarm triggers, it will remain triggered until the hardware register is read at least once. This means that the cause for the alarm may already have disappeared!

This driver only updates its values each 1.5 seconds; reading it more often will do no harm, but will return 'old' values. It is possible to make ADM1021-clones do faster measurements, but there is really no good reason for that.

7.8.3 Netburst-based Xeon support

Some Xeon processors based on the Netburst (early Pentium 4, from 2001 to 2003) microarchitecture had real MAX1617, ADM1021, or compatible chips within them, with two temperature sensors. Other Xeon processors of this era (with 400 MHz FSB) had chips with only one temperature sensor.

If you have such an old Xeon, and you get two valid temperatures when loading the adm1021 module, then things are good.

If nothing happens when loading the adm1021 module, and you are certain that your specific Xeon processor model includes compatible sensors, you will have to explicitly instantiate the sensor chips from user-space. See method 4 in Documentation/i2c/instantiating-devices.rst. Possible slave addresses are 0x18, 0x1a, 0x29, 0x2b, 0x4c, or 0x4e. It is likely that only temp2 will be correct and temp1 will have to be ignored.

Previous generations of the Xeon processor (based on Pentium II/III) didn't have these sensors. Next generations of Xeon processors (533 MHz FSB and faster) lost them, until the Core-based generation which introduced integrated digital thermal sensors. These are supported by the coretemp driver.

7.9 Kernel driver adm1025

Supported chips:

- Analog Devices ADM1025, ADM1025A

  Prefix: 'adm1025'

  Addresses scanned: I2C 0x2c - 0x2e

  Datasheet: Publicly available at the Analog Devices website

- Philips NE1619

  Prefix: 'ne1619'

  Addresses scanned: I2C 0x2c - 0x2d

  Datasheet: Publicly available at the Philips website

The NE1619 presents some differences with the original ADM1025:

- Only two possible addresses (0x2c - 0x2d).
- No temperature offset register, but we don't use it anyway.
- No INT mode for pin 16. We don't play with it anyway.

Authors:

- Chen-Yuan Wu <gwu@esoft.com>,

- Jean Delvare <jdelvare@suse.de>

7.9.1 Description

(This is from Analog Devices.) The ADM1025 is a complete system hardware monitor for microprocessor-based systems, providing measurement and limit comparison of various system parameters. Five voltage measurement inputs are provided, for monitoring +2.5V, +3.3V, +5V and +12V power supplies and the processor core voltage. The ADM1025 can monitor a sixth power-supply voltage by measuring its own VCC. One input (two pins) is dedicated to a remote temperature-sensing diode and an on-chip temperature sensor allows ambient temperature to be monitored.

One specificity of this chip is that the pin 11 can be hardwired in two different manners. It can act as the +12V power-supply voltage analog input, or as the a fifth digital entry for the VID reading (bit 4). It's kind of strange since both are useful, and the reason for designing the chip that way is obscure at least to me. The bit 5 of the configuration register can be used to define how the chip is hardwired. Please note that it is not a choice you have to make as the user. The choice was already made by your motherboard's maker. If the configuration bit isn't set properly, you'll have a wrong +12V reading or a wrong VID reading. The way the driver handles that is to preserve this bit through the initialization process, assuming that the BIOS set it up properly beforehand. If it turns out not to be true in some cases, we'll provide a module parameter to force modes.

This driver also supports the ADM1025A, which differs from the ADM1025 only in that it has "open-drain VID inputs while the ADM1025 has on-chip 100k pull-ups on the VID inputs". It doesn't make any difference for us.

7.10 Kernel driver adm1026

Supported chips:

- Analog Devices ADM1026

  Prefix: 'adm1026'

  Addresses scanned: I2C 0x2c, 0x2d, 0x2e

  Datasheet: Publicly available at the Analog Devices website

  http://www.onsemi.com/PowerSolutions/product.do?id=ADM1026

Authors:

- Philip Pokorny <ppokorny@penguincomputing.com> for Penguin Computing
- Justin Thiessen <jthiessen@penguincomputing.com>

7.10.1 Module Parameters

- **gpio_input: int array (min = 1, max = 17)** List of GPIO pins (0-16) to program as inputs
- **gpio_output: int array (min = 1, max = 17)** List of GPIO pins (0-16) to program as outputs
- **gpio_inverted: int array (min = 1, max = 17)** List of GPIO pins (0-16) to program as inverted
- **gpio_normal: int array (min = 1, max = 17)** List of GPIO pins (0-16) to program as normal/non-inverted
- **gpio_fan: int array (min = 1, max = 8)** List of GPIO pins (0-7) to program as fan tachs

7.10.2 Description

This driver implements support for the Analog Devices ADM1026. Analog Devices calls it a "complete thermal system management controller."

The ADM1026 implements three (3) temperature sensors, 17 voltage sensors, 16 general purpose digital I/O lines, eight (8) fan speed sensors (8-bit), an analog output and a PWM output along with limit, alarm and mask bits for all of the above. There is even 8k bytes of EEPROM memory on chip.

Temperatures are measured in degrees Celsius. There are two external sensor inputs and one internal sensor. Each sensor has a high and low limit. If the limit is exceeded, an interrupt (#SMBALERT) can be generated. The interrupts can be masked. In addition, there are over-temp limits for each sensor. If this limit is exceeded, the #THERM output will be asserted. The current temperature and limits have a resolution of 1 degree.

Fan rotation speeds are reported in RPM (rotations per minute) but measured in counts of a 22.5kHz internal clock. Each fan has a high limit which corresponds to a minimum fan speed. If the limit is exceeded, an interrupt can be generated. Each fan can be programmed to divide the reference clock by 1, 2, 4 or 8. Not all RPM values can accurately be represented, so some rounding is done. With a divider of 8, the slowest measurable speed of a two pulse per revolution fan is 661 RPM.

There are 17 voltage sensors. An alarm is triggered if the voltage has crossed a programmable minimum or maximum limit. Note that minimum in this case always means 'closest to zero'; this is important for negative voltage measurements. Several inputs have integrated attenuators so they can measure higher voltages directly. 3.3V, 5V, 12V, -12V and battery voltage all have dedicated inputs. There are several inputs scaled to 0-3V full-scale range for SCSI terminator power. The remaining inputs are not scaled and have a 0-2.5V full-scale range. A 2.5V or 1.82V reference voltage is provided for negative voltage measurements.

If an alarm triggers, it will remain triggered until the hardware register is read at least once. This means that the cause for the alarm may already have disappeared! Note that in the current implementation, all hardware registers are read whenever

any data is read (unless it is less than 2.0 seconds since the last update). This means that you can easily miss once-only alarms.

The ADM1026 measures continuously. Analog inputs are measured about 4 times a second. Fan speed measurement time depends on fan speed and divisor. It can take as long as 1.5 seconds to measure all fan speeds.

The ADM1026 has the ability to automatically control fan speed based on the temperature sensor inputs. Both the PWM output and the DAC output can be used to control fan speed. Usually only one of these two outputs will be used. Write the minimum PWM or DAC value to the appropriate control register. Then set the low temperature limit in the tmin values for each temperature sensor. The range of control is fixed at 20 °C, and the largest difference between current and tmin of the temperature sensors sets the control output. See the datasheet for several example circuits for controlling fan speed with the PWM and DAC outputs. The fan speed sensors do not have PWM compensation, so it is probably best to control the fan voltage from the power lead rather than on the ground lead.

The datasheet shows an example application with VID signals attached to GPIO lines. Unfortunately, the chip may not be connected to the VID lines in this way. The driver assumes that the chips is connected this way to get a VID voltage.

7.11 Kernel driver adm1031

Supported chips:

- Analog Devices ADM1030

  Prefix: ‘adm1030’

  Addresses scanned: I2C 0x2c to 0x2e

  Datasheet: Publicly available at the Analog Devices website

  http://www.analog.com/en/prod/0%2C2877%2CADM1030%2C00.html

- Analog Devices ADM1031

  Prefix: ‘adm1031’

  Addresses scanned: I2C 0x2c to 0x2e

  Datasheet: Publicly available at the Analog Devices website

  http://www.analog.com/en/prod/0%2C2877%2CADM1031%2C00.html

Authors:

- Alexandre d’ Alton <alex@alexdalton.org>
- Jean Delvare <jdelvare@suse.de>

7.11.1 Description

The ADM1030 and ADM1031 are digital temperature sensors and fan controllers. They sense their own temperature as well as the temperature of up to one (ADM1030) or two (ADM1031) external diodes.

All temperature values are given in degrees Celsius. Resolution is 0.5 degree for the local temperature, 0.125 degree for the remote temperatures.

Each temperature channel has its own high and low limits, plus a critical limit.

The ADM1030 monitors a single fan speed, while the ADM1031 monitors up to two. Each fan channel has its own low speed limit.

7.12 Kernel driver adm1177

Supported chips:

- Analog Devices ADM1177 Prefix: 'adm1177' Datasheet: https://www.analog.com/media/en/technical-documentation/data-sheets/ADM1177.pdf

Author: Beniamin Bia <beniamin.bia@analog.com>

7.12.1 Description

This driver supports hardware monitoring for Analog Devices ADM1177 Hot-Swap Controller and Digital Power Monitors with Soft Start Pin.

7.12.2 Usage Notes

This driver does not auto-detect devices. You will have to instantiate the devices explicitly. Please see /i2c/instantiating-devices for details.

7.12.3 Sysfs entries

The following attributes are supported. Current maxim attribute is read-write, all other attributes are read-only.

in0_input Measured voltage in microvolts.

curr1_input Measured current in microamperes. curr1_max_alarm Overcurrent alarm in microamperes.

7.13 Kernel driver adm1275

Supported chips:

- Analog Devices ADM1075

  Prefix: 'adm1075'

  Addresses scanned: -

  Datasheet: www.analog.com/static/imported-files/data_sheets/ADM1075.pdf

- Analog Devices ADM1272

  Prefix: 'adm1272'

  Addresses scanned: -

  Datasheet: www.analog.com/static/imported-files/data_sheets/ADM1272.pdf

- Analog Devices ADM1275

  Prefix: 'adm1275'

  Addresses scanned: -

  Datasheet: www.analog.com/static/imported-files/data_sheets/ADM1275.pdf

- Analog Devices ADM1276

  Prefix: 'adm1276'

  Addresses scanned: -

  Datasheet: www.analog.com/static/imported-files/data_sheets/ADM1276.pdf

- Analog Devices ADM1278

  Prefix: 'adm1278'

  Addresses scanned: -

  Datasheet: www.analog.com/static/imported-files/data_sheets/ADM1278.pdf

- Analog Devices ADM1293/ADM1294

  Prefix: 'adm1293', 'adm1294'

  Addresses scanned: -

  Datasheet: http://www.analog.com/media/en/technical-documentation/data-sheets/ADM1293_1294.pdf

Author: Guenter Roeck <linux@roeck-us.net>

7.13.1 Description

This driver supports hardware monitoring for Analog Devices ADM1075, ADM1272, ADM1275, ADM1276, ADM1278, ADM1293, and ADM1294 Hot-Swap Controller and Digital Power Monitors.

ADM1075, ADM1272, ADM1275, ADM1276, ADM1278, ADM1293, and ADM1294 are hot-swap controllers that allow a circuit board to be removed from or inserted into a live backplane. They also feature current and voltage readback via an integrated 12 bit analog-to-digital converter (ADC), accessed using a PMBus interface.

The driver is a client driver to the core PMBus driver. Please see Documentation/hwmon/pmbus.rst for details on PMBus client drivers.

7.13.2 Usage Notes

This driver does not auto-detect devices. You will have to instantiate the devices explicitly. Please see Documentation/i2c/instantiating-devices.rst for details.

The ADM1075, unlike many other PMBus devices, does not support internal voltage or current scaling. Reported voltages, currents, and power are raw measurements, and will typically have to be scaled.

The shunt value in micro-ohms can be set via device tree at compile-time. Please refer to the Documentation/devicetree/bindings/hwmon/adm1275.txt for bindings if the device tree is used.

7.13.3 Platform data support

The driver supports standard PMBus driver platform data. Please see Documentation/hwmon/pmbus.rst for details.

7.13.4 Sysfs entries

The following attributes are supported. Limits are read-write, history reset attributes are write-only, all other attributes are read-only.

inX_label	“vin1” or “vout1” depending on chip variant and configuration. On ADM1075, ADM1293, and ADM1294, vout1 reports the voltage on the VAUX pin.
inX_input	Measured voltage.
inX_min	Minimum Voltage.
inX_max	Maximum voltage.
inX_min_alarm	Voltage low alarm.
inX_max_alarm	Voltage high alarm.
inX_highest	Historical maximum voltage.
inX_reset_history	Write any value to reset history.
curr1_label	“iout1”
curr1_input	Measured current.
curr1_max	Maximum current.
curr1_max_alarm	Current high alarm.
curr1_lcrit	Critical minimum current. Depending on the chip configuration, either curr1_lcrit or curr1_crit is supported, but not both.
curr1_lcrit_alarm	Critical current low alarm.
curr1_crit	Critical maximum current. Depending on the chip configuration, either curr1_lcrit or curr1_crit is supported, but not both.
curr1_crit_alarm	Critical current high alarm.
curr1_highest	Historical maximum current.
curr1_reset_history	Write any value to reset history.
power1_label	“pin1”
power1_input	Input power.
power1_input_lowest	Lowest observed input power. ADM1293 and ADM1294 only.
power1_input_highest	Highest observed input power.
power1_reset_history	Write any value to reset history. Power attributes are supported on ADM1075, ADM1272, ADM1276, ADM1293, and ADM1294.
temp1_input	Chip temperature.
temp1_max	Maximum chip temperature.
temp1_max_alarm	Temperature alarm.
temp1_crit	Critical chip temperature.
temp1_crit_alarm	Critical temperature high alarm.
temp1_highest	Highest observed temperature.
temp1_reset_history	Write any value to reset history. Temperature attributes are supported on ADM1272 and ADM1278.

7.14 Kernel driver adm9240

Supported chips:

- Analog Devices ADM9240

  Prefix: 'adm9240'

  Addresses scanned: I2C 0x2c - 0x2f

  Datasheet: Publicly available at the Analog Devices website

  http://www.analog.com/UploadedFiles/Data_Sheets/79857778ADM9240_0.pdf

- Dallas Semiconductor DS1780

  Prefix: 'ds1780'

  Addresses scanned: I2C 0x2c - 0x2f

  Datasheet: Publicly available at the Dallas Semiconductor (Maxim) website

  http://pdfserv.maxim-ic.com/en/ds/DS1780.pdf

- National Semiconductor LM81

  Prefix: 'lm81'

  Addresses scanned: I2C 0x2c - 0x2f

  Datasheet: Publicly available at the National Semiconductor website

  http://www.national.com/ds.cgi/LM/LM81.pdf

Authors:

- Frodo Looijaard <frodol@dds.nl>,
- Philip Edelbrock <phil@netroedge.com>,
- Michiel Rook <michiel@grendelproject.nl>,
- Grant Coady <gcoady.lk@gmail.com> with guidance from Jean Delvare <jdelvare@suse.de>

7.14.1 Interface

The I2C addresses listed above assume BIOS has not changed the chip MSB 5-bit address. Each chip reports a unique manufacturer identification code as well as the chip revision/stepping level.

7.14.2 Description

[From ADM9240] The ADM9240 is a complete system hardware monitor for microprocessor-based systems, providing measurement and limit comparison of up to four power supplies and two processor core voltages, plus temperature, two fan speeds and chassis intrusion. Measured values can be read out via an I2C-compatible serial System Management Bus, and values for limit comparisons can be programmed in over the same serial bus. The high speed successive approximation ADC allows frequent sampling of all analog channels to ensure a fast interrupt response to any out-of-limit measurement.

The ADM9240, DS1780 and LM81 are register compatible, the following details are common to the three chips. Chip differences are described after this section.

7.14.3 Measurements

The measurement cycle

The adm9240 driver will take a measurement reading no faster than once each two seconds. User-space may read sysfs interface faster than the measurement update rate and will receive cached data from the most recent measurement.

ADM9240 has a very fast 320us temperature and voltage measurement cycle with independent fan speed measurement cycles counting alternating rising edges of the fan tacho inputs.

DS1780 measurement cycle is about once per second including fan speed.

LM81 measurement cycle is about once per 400ms including fan speed. The LM81 12-bit extended temperature measurement mode is not supported.

7.14.4 Temperature

On chip temperature is reported as degrees Celsius as 9-bit signed data with resolution of 0.5 degrees Celsius. High and low temperature limits are 8-bit signed data with resolution of one degree Celsius.

Temperature alarm is asserted once the temperature exceeds the high limit, and is cleared when the temperature falls below the temp1_max_hyst value.

7.14.5 Fan Speed

Two fan tacho inputs are provided, the ADM9240 gates an internal 22.5kHz clock via a divider to an 8-bit counter. Fan speed (rpm) is calculated by:

rpm = (22500 * 60) / (count * divider)

Automatic fan clock divider

- User sets 0 to fan_min limit
  - low speed alarm is disabled
  - fan clock divider not changed

  - auto fan clock adjuster enabled for valid fan speed reading

- User sets fan_min limit too low
  - low speed alarm is enabled
  - fan clock divider set to max
  - fan_min set to register value 254 which corresponds to 664 rpm on adm9240
  - low speed alarm will be asserted if fan speed is less than minimum measurable speed
  - auto fan clock adjuster disabled
- User sets reasonable fan speed
  - low speed alarm is enabled
  - fan clock divider set to suit fan_min
  - auto fan clock adjuster enabled: adjusts fan_min
- User sets unreasonably high low fan speed limit
  - resolution of the low speed limit may be reduced
  - alarm will be asserted
  - auto fan clock adjuster enabled: adjusts fan_min
- fan speed may be displayed as zero until the auto fan clock divider adjuster brings fan speed clock divider back into chip measurement range, this will occur within a few measurement cycles.

7.14.6 Analog Output

An analog output provides a 0 to 1.25 volt signal intended for an external fan speed amplifier circuit. The analog output is set to maximum value on power up or reset. This doesn't do much on the test Intel SE440BX-2.

Voltage Monitor

Voltage (IN) measurement is internally scaled:

nr	label	**nominal** mV	**maximum** mV	**resolution** mV
0	+2.5V	2500	3320	13.0
1	Vccp1	2700	3600	14.1
2	+3.3V	3300	4380	17.2
3	+5V	5000	6640	26.0
4	+12V	12000	15940	62.5
5	Vccp2	2700	3600	14.1

The reading is an unsigned 8-bit value, nominal voltage measurement is represented by a reading of 192, being 3/4 of the measurement range.

An alarm is asserted for any voltage going below or above the set limits.

The driver reports and accepts voltage limits scaled to the above table.

7.14.7 VID Monitor

The chip has five inputs to read the 5-bit VID and reports the mV value based on detected CPU type.

7.14.8 Chassis Intrusion

An alarm is asserted when the CI pin goes active high. The ADM9240 Datasheet has an example of an external temperature sensor driving this pin. On an Intel SE440BX-2 the Chassis Intrusion header is connected to a normally open switch.

The ADM9240 provides an internal open drain on this line, and may output a 20 ms active low pulse to reset an external Chassis Intrusion latch.

Clear the CI latch by writing value 0 to the sysfs intrusion0_alarm file.

Alarm flags reported as 16-bit word

bit	label	comment
0	+2.5 V_Error	high or low limit exceeded
1	VCCP_Error	high or low limit exceeded
2	+3.3 V_Error	high or low limit exceeded
3	+5 V_Error	high or low limit exceeded
4	Temp_Error	temperature error
6	FAN1_Error	fan low limit exceeded
7	FAN2_Error	fan low limit exceeded
8	+12 V_Error	high or low limit exceeded
9	VCCP2_Error	high or low limit exceeded
12	Chassis_Error	CI pin went high

Remaining bits are reserved and thus undefined. It is important to note that alarm bits may be cleared on read, user-space may latch alarms and provide the end-user with a method to clear alarm memory.

7.15 Kernel driver ads7828

Supported chips:

- Texas Instruments/Burr-Brown ADS7828

  Prefix: ‘ads7828’

  Datasheet: Publicly available at the Texas Instruments website:

  http://focus.ti.com/lit/ds/symlink/ads7828.pdf

- Texas Instruments ADS7830

  Prefix: 'ads7830'

  Datasheet: Publicly available at the Texas Instruments website:

  http://focus.ti.com/lit/ds/symlink/ads7830.pdf

Authors:

- Steve Hardy <shardy@redhat.com>
- Vivien Didelot <vivien.didelot@savoirfairelinux.com>
- Guillaume Roguez <guillaume.roguez@savoirfairelinux.com>

7.15.1 Platform data

The ads7828 driver accepts an optional ads7828_platform_data structure (defined in include/linux/platform_data/ads7828.h). The structure fields are:

- **diff_input: (bool) Differential operation** set to true for differential mode, false for default single ended mode.
- **ext_vref: (bool) External reference** set to true if it operates with an external reference, false for default internal reference.
- **vref_mv: (unsigned int) Voltage reference** if using an external reference, set this to the reference voltage in mV, otherwise it will default to the internal value (2500mV). This value will be bounded with limits accepted by the chip, described in the datasheet.

If no structure is provided, the configuration defaults to single ended operation and internal voltage reference (2.5V).

7.15.2 Description

This driver implements support for the Texas Instruments ADS7828 and ADS7830.

The ADS7828 device is a 12-bit 8-channel A/D converter, while the ADS7830 does 8-bit sampling.

It can operate in single ended mode (8 +ve inputs) or in differential mode, where 4 differential pairs can be measured.

The chip also has the facility to use an external voltage reference. This may be required if your hardware supplies the ADS7828 from a 5V supply, see the datasheet for more details.

There is no reliable way to identify this chip, so the driver will not scan some addresses to try to auto-detect it. That means that you will have to statically declare the device in the platform support code.

7.16 Kernel driver adt7410

Supported chips:

- Analog Devices ADT7410

  Prefix: 'adt7410'

  Addresses scanned: None

  Datasheet: Publicly available at the Analog Devices website

  http://www.analog.com/static/imported-files/data_sheets/ADT7410.pdf

- Analog Devices ADT7420

  Prefix: 'adt7420'

  Addresses scanned: None

  Datasheet: Publicly available at the Analog Devices website

  http://www.analog.com/static/imported-files/data_sheets/ADT7420.pdf

- Analog Devices ADT7310

  Prefix: 'adt7310'

  Addresses scanned: None

  Datasheet: Publicly available at the Analog Devices website

  http://www.analog.com/static/imported-files/data_sheets/ADT7310.pdf

- Analog Devices ADT7320

  Prefix: 'adt7320'

  Addresses scanned: None

  Datasheet: Publicly available at the Analog Devices website

  http://www.analog.com/static/imported-files/data_sheets/ADT7320.pdf

Author: Hartmut Knaack <knaack.h@gmx.de>

7.16.1 Description

The ADT7310/ADT7410 is a temperature sensor with rated temperature range of -55°C to +150°C. It has a high accuracy of +/-0.5°C and can be operated at a resolution of 13 bits (0.0625°C) or 16 bits (0.0078°C). The sensor provides an INT pin to indicate that a minimum or maximum temperature set point has been exceeded, as well as a critical temperature (CT) pin to indicate that the critical temperature set point has been exceeded. Both pins can be set up with a common hysteresis of 0°C - 15°C and a fault queue, ranging from 1 to 4 events. Both pins can individually set to be active-low or active-high, while the whole device can either run in comparator mode or interrupt mode. The ADT7410 supports

continuous temperature sampling, as well as sampling one temperature value per second or even just get one sample on demand for power saving. Besides, it can completely power down its ADC, if power management is required.

The ADT7320/ADT7420 is register compatible, the only differences being the package, a slightly narrower operating temperature range (-40°C to +150°C), and a better accuracy (0.25°C instead of 0.50°C.)

The difference between the ADT7310/ADT7320 and ADT7410/ADT7420 is the control interface, the ADT7310 and ADT7320 use SPI while the ADT7410 and ADT7420 use I2C.

7.16.2 Configuration Notes

Since the device uses one hysteresis value, which is an offset to minimum, maximum and critical temperature, it can only be set for temp#_max_hyst. However, temp#_min_hyst and temp#_crit_hyst show their corresponding hysteresis. The device is set to 16 bit resolution and comparator mode.

7.16.3 sysfs-Interface

temp#_input	temperature input
temp#_min	temperature minimum setpoint
temp#_max	temperature maximum setpoint
temp#_crit	critical temperature setpoint
temp#_min_hyst	hysteresis for temperature minimum (read-only)
temp#_max_hyst	hysteresis for temperature maximum (read/write)
temp#_crit_hyst	hysteresis for critical temperature (read-only)
temp#_min_alarm	temperature minimum alarm flag
temp#_max_alarm	temperature maximum alarm flag
temp#_crit_alarm	critical temperature alarm flag

7.17 Kernel driver adt7411

Supported chips:

- Analog Devices ADT7411

  Prefix: ‘adt7411’

  Addresses scanned: 0x48, 0x4a, 0x4b

  Datasheet: Publicly available at the Analog Devices website

Author: Wolfram Sang (based on adt7470 by Darrick J. Wong)

7.17.1 Description

This driver implements support for the Analog Devices ADT7411 chip. There may be other chips that implement this interface.

The ADT7411 can use an I2C/SMBus compatible 2-wire interface or an SPI-compatible 4-wire interface. It provides a 10-bit analog to digital converter which measures 1 temperature, vdd and 8 input voltages. It has an internal temperature sensor, but an external one can also be connected (one loses 2 inputs then). There are high- and low-limit registers for all inputs.

Check the datasheet for details.

7.17.2 sysfs-Interface

in0_input	vdd voltage input
in[1-8]_input	analog 1-8 input
temp1_input	temperature input

Besides standard interfaces, this driver adds (0 = off, 1 = on):

adc_ref_vdd	Use vdd as reference instead of 2.25 V
fast_sampling	Sample at 22.5 kHz instead of 1.4 kHz, but drop filters
no_average	Turn off averaging over 16 samples

7.17.3 Notes

SPI, external temperature sensor and limit registers are not supported yet.

7.18 Kernel driver adt7462

Supported chips:

- Analog Devices ADT7462

  Prefix: 'adt7462'

  Addresses scanned: I2C 0x58, 0x5C

  Datasheet: Publicly available at the Analog Devices website

Author: Darrick J. Wong

7.18.1 Description

This driver implements support for the Analog Devices ADT7462 chip family.

This chip is a bit of a beast. It has 8 counters for measuring fan speed. It can also measure 13 voltages or 4 temperatures, or various combinations of the two. See the chip documentation for more details about the exact set of configurations. This driver does not allow one to configure the chip; that is left to the system designer.

A sophisticated control system for the PWM outputs is designed into the ADT7462 that allows fan speed to be adjusted automatically based on any of the three temperature sensors. Each PWM output is individually adjustable and programmable. Once configured, the ADT7462 will adjust the PWM outputs in response to the measured temperatures without further host intervention. This feature can also be disabled for manual control of the PWM's.

Each of the measured inputs (voltage, temperature, fan speed) has corresponding high/low limit values. The ADT7462 will signal an ALARM if any measured value exceeds either limit.

The ADT7462 samples all inputs continuously. The driver will not read the registers more often than once every other second. Further, configuration data is only read once per minute.

7.18.2 Special Features

The ADT7462 have a 10-bit ADC and can therefore measure temperatures with 0.25 degC resolution.

The Analog Devices datasheet is very detailed and describes a procedure for determining an optimal configuration for the automatic PWM control.

The driver will report sensor labels when it is able to determine that information from the configuration registers.

7.18.3 Configuration Notes

Besides standard interfaces driver adds the following:

- PWM Control
- pwm#_auto_point1_pwm and temp#_auto_point1_temp and
- pwm#_auto_point2_pwm and temp#_auto_point2_temp -
  - point1: Set the pwm speed at a lower temperature bound.
  - point2: Set the pwm speed at a higher temperature bound.

The ADT7462 will scale the pwm between the lower and higher pwm speed when the temperature is between the two temperature boundaries. PWM values range from 0 (off) to 255 (full speed). Fan speed will be set to maximum when the temperature sensor associated with the PWM control exceeds temp#_max.

7.19 Kernel driver adt7470

Supported chips:

- Analog Devices ADT7470

  Prefix: 'adt7470'

  Addresses scanned: I2C 0x2C, 0x2E, 0x2F

  Datasheet: Publicly available at the Analog Devices website

Author: Darrick J. Wong

7.19.1 Description

This driver implements support for the Analog Devices ADT7470 chip. There may be other chips that implement this interface.

The ADT7470 uses the 2-wire interface compatible with the SMBus 2.0 specification. Using an analog to digital converter it measures up to ten (10) external temperatures. It has four (4) 16-bit counters for measuring fan speed. There are four (4) PWM outputs that can be used to control fan speed.

A sophisticated control system for the PWM outputs is designed into the ADT7470 that allows fan speed to be adjusted automatically based on any of the ten temperature sensors. Each PWM output is individually adjustable and programmable. Once configured, the ADT7470 will adjust the PWM outputs in response to the measured temperatures with further host intervention. This feature can also be disabled for manual control of the PWM's.

Each of the measured inputs (temperature, fan speed) has corresponding high/low limit values. The ADT7470 will signal an ALARM if any measured value exceeds either limit.

The ADT7470 samples all inputs continuously. A kernel thread is started up for the purpose of periodically querying the temperature sensors, thus allowing the automatic fan pwm control to set the fan speed. The driver will not read the registers more often than once every 5 seconds. Further, configuration data is only read once per minute.

7.19.2 Special Features

The ADT7470 has a 8-bit ADC and is capable of measuring temperatures with 1 degC resolution.

The Analog Devices datasheet is very detailed and describes a procedure for determining an optimal configuration for the automatic PWM control.

7.19.3 Configuration Notes

Besides standard interfaces driver adds the following:

- PWM Control
- pwm#_auto_point1_pwm and pwm#_auto_point1_temp and
- pwm#_auto_point2_pwm and pwm#_auto_point2_temp -
  - point1: Set the pwm speed at a lower temperature bound.
  - point2: Set the pwm speed at a higher temperature bound.

The ADT7470 will scale the pwm between the lower and higher pwm speed when the temperature is between the two temperature boundaries. PWM values range from 0 (off) to 255 (full speed). Fan speed will be set to maximum when the temperature sensor associated with the PWM control exceeds pwm#_auto_point2_temp.

The driver also allows control of the PWM frequency:

- pwm1_freq

The PWM frequency is rounded to the nearest one of:

- 11.0 Hz
- 14.7 Hz
- 22.1 Hz
- 29.4 Hz
- 35.3 Hz
- 44.1 Hz
- 58.8 Hz
- 88.2 Hz
- 1.4 kHz
- 22.5 kHz

7.19.4 Notes

The temperature inputs no longer need to be read periodically from userspace in order for the automatic pwm algorithm to run. This was the case for earlier versions of the driver.

7.20 Kernel driver adt7475

Supported chips:

- Analog Devices ADT7473

  Prefix: 'adt7473'

  Addresses scanned: I2C 0x2C, 0x2D, 0x2E

  Datasheet: Publicly available at the On Semiconductors website

- Analog Devices ADT7475

  Prefix: 'adt7475'

  Addresses scanned: I2C 0x2E

  Datasheet: Publicly available at the On Semiconductors website

- Analog Devices ADT7476

  Prefix: 'adt7476'

  Addresses scanned: I2C 0x2C, 0x2D, 0x2E

  Datasheet: Publicly available at the On Semiconductors website

- Analog Devices ADT7490

  Prefix: 'adt7490'

  Addresses scanned: I2C 0x2C, 0x2D, 0x2E

  Datasheet: Publicly available at the On Semiconductors website

Authors:

- Jordan Crouse
- Hans de Goede
- Darrick J. Wong (documentation)
- Jean Delvare

7.20.1 Description

This driver implements support for the Analog Devices ADT7473, ADT7475, ADT7476 and ADT7490 chip family. The ADT7473 and ADT7475 differ only in minor details. The ADT7476 has additional features, including extra voltage measurement inputs and VID support. The ADT7490 also has additional features, including extra voltage measurement inputs and PECI support. All the supported chips will be collectively designed by the name "ADT747x" in the rest of this document.

The ADT747x uses the 2-wire interface compatible with the SMBus 2.0 specification. Using an analog to digital converter it measures three (3) temperatures and two (2) or more voltages. It has four (4) 16-bit counters for measuring fan speed. There are three (3) PWM outputs that can be used to control fan speed.

A sophisticated control system for the PWM outputs is designed into the ADT747x that allows fan speed to be adjusted automatically based on any of the three temperature sensors. Each PWM output is individually adjustable and programmable. Once configured, the ADT747x will adjust the PWM outputs in response to the measured temperatures without further host intervention. This feature can also be disabled for manual control of the PWM' s.

Each of the measured inputs (voltage, temperature, fan speed) has corresponding high/low limit values. The ADT747x will signal an ALARM if any measured value exceeds either limit.

The ADT747x samples all inputs continuously. The driver will not read the registers more often than once every other second. Further, configuration data is only read once per minute.

7.20.2 Chip Differences Summary

ADT7473:

- 2 voltage inputs
- system acoustics optimizations (not implemented)

ADT7475:

- 2 voltage inputs

ADT7476:

- 5 voltage inputs
- VID support

ADT7490:

- 6 voltage inputs
- 1 Imon input (not implemented)
- PECI support (not implemented)
- 2 GPIO pins (not implemented)
- system acoustics optimizations (not implemented)

7.20.3 Sysfs Mapping

in	ADT7490	ADT7476	ADT7475	ADT7473
in0	2.5VIN (22)	2.5VIN (22)	•	•
in1	VCCP (23)	VCCP (23)	VCCP (14)	VCCP (14)
in2	VCC (4)	VCC (4)	VCC (4)	VCC (3)
in3	5VIN (20)	5VIN (20)		
in4	12VIN (21)	12VIN (21)		
in5	VTT (8)			

7.20.4 Special Features

The ADT747x has a 10-bit ADC and can therefore measure temperatures with a resolution of 0.25 degree Celsius. Temperature readings can be configured either for two's complement format or "Offset 64" format, wherein 64 is subtracted from the raw value to get the temperature value.

The datasheet is very detailed and describes a procedure for determining an optimal configuration for the automatic PWM control.

7.20.5 Fan Speed Control

The driver exposes two trip points per PWM channel.

- point1: Set the PWM speed at the lower temperature bound
- point2: Set the PWM speed at the higher temperature bound

The ADT747x will scale the PWM linearly between the lower and higher PWM speed when the temperature is between the two temperature boundaries. Temperature boundaries are associated to temperature channels rather than PWM outputs, and a given PWM output can be controlled by several temperature channels. As a result, the ADT747x may compute more than one PWM value for a channel at a given time, in which case the maximum value (fastest fan speed) is applied. PWM values range from 0 (off) to 255 (full speed).

Fan speed may be set to maximum when the temperature sensor associated with the PWM control exceeds temp#_max.

At Tmin - hysteresis the PWM output can either be off (0% duty cycle) or at the minimum (i.e. auto_point1_pwm). This behaviour can be configured using the pwm[1-*]_stall_disable sysfs attribute. A value of 0 means the fans will shut off. A value of 1 means the fans will run at auto_point1_pwm.

The responsiveness of the ADT747x to temperature changes can be configured. This allows smoothing of the fan speed transition. To set the transition time set the value in ms in the temp[1-*]_smoothing sysfs attribute.

7.20.6 Notes

The nVidia binary driver presents an ADT7473 chip via an on-card i2c bus. Unfortunately, they fail to set the i2c adapter class, so this driver may fail to find the chip until the nvidia driver is patched.

7.21 Kernel driver amc6821

Supported chips:

Texas Instruments AMC6821

Prefix: 'amc6821'

Addresses scanned: 0x18, 0x19, 0x1a, 0x2c, 0x2d, 0x2e, 0x4c, 0x4d, 0x4e

Datasheet: http://focus.ti.com/docs/prod/folders/print/amc6821.html

Authors: Tomaz Mertelj <tomaz.mertelj@guest.arnes.si>

7.21.1 Description

This driver implements support for the Texas Instruments amc6821 chip. The chip has one on-chip and one remote temperature sensor and one pwm fan regulator. The pwm can be controlled either from software or automatically.

The driver provides the following sensor accesses in sysfs:

temp1_input	ro	on-chip temperature
temp1_min	rw	"
temp1_max	rw	"
temp1_crit	rw	"
temp1_min_alarm	ro	"
temp1_max_alarm	ro	"
temp1_crit_alarm	ro	"
temp2_input	ro	remote temperature
temp2_min	rw	"
temp2_max	rw	"
temp2_crit	rw	"
temp2_min_alarm	ro	"
temp2_max_alarm	ro	"
temp2_crit_alarm	ro	"
temp2_fault	ro	"
fan1_input	ro	tachometer speed
fan1_min	rw	"
fan1_max	rw	"
fan1_fault	ro	"
fan1_div	rw	Fan divisor can be either 2 or 4.
pwm1	rw	pwm1
pwm1_enable	rw	regulator mode, 1=open loop, 2=fan controlled by remote
pwm1_auto_channels_temp	ro	1 if pwm_enable==2, 3 if pwm_enable==3
pwm1_auto_point1_pwm	ro	Hardwired to 0, shared for both temperature channels.
pwm1_auto_point2_pwm	rw	This value is shared for both temperature channels.
pwm1_auto_point3_pwm	rw	Hardwired to 255, shared for both temperature channels.
temp1_auto_point1_temp	ro	Hardwired to temp2_auto_point1_temp which is rw. Below
temp1_auto_point2_temp	rw	The low-temperature limit of the proportional range. Below

Continued on next page

Table 1 – continued from previous page

temp1_auto_point3_temp	rw	Above this temperature fan runs at maximum speed. It can
temp2_auto_point1_temp	rw	Must be between 0 degree C and 63 degree C and it define
temp2_auto_point2_temp	rw	The low-temperature limit of the proportional range. Below
temp2_auto_point3_temp	rw	Above this temperature fan runs at maximum speed. It can

7.21.2 Module parameters

If your board has a BIOS that initializes the amc6821 correctly, you should load the module with: init=0.

If your board BIOS doesn't initialize the chip, or you want different settings, you can set the following parameters:

- init=1,
- pwminv: 0 default pwm output, 1 inverts pwm output.

7.22 Kernel driver amd_energy

Supported chips:

- AMD Family 17h Processors

  Prefix: 'amd_energy'

  Addresses used: RAPL MSRs

  Datasheets:

  - Processor Programming Reference (PPR) for AMD Family 17h Model 01h, Revision B1 Processors

    https://developer.amd.com/wp-content/resources/55570-B1_PUB.zip

  - Preliminary Processor Programming Reference (PPR) for AMD Family 17h Model 31h, Revision B0 Processors

    https://developer.amd.com/wp-content/resources/56176_ppr_Family_17h_Model_71h_B0_pub_Rev_3.06.zip

Author: Naveen Krishna Chatradhi <nchatrad@amd.com>

7.22.1 Description

The Energy driver exposes the energy counters that are reported via the Running Average Power Limit (RAPL) Model-specific Registers (MSRs) via the hardware monitor (HWMON) sysfs interface.

1. Power, Energy and Time Units MSR_RAPL_POWER_UNIT/ C001_0299: shared with all cores in the socket
2. Energy consumed by each Core MSR_CORE_ENERGY_STATUS/ C001_029A: 32-bitRO, Accumulator, core-level power reporting

3. Energy consumed by Socket MSR_PACKAGE_ENERGY_STATUS/ C001_029B: 32-bitRO, Accumulator, socket-level power reporting, shared with all cores in socket

These registers are updated every 1ms and cleared on reset of the system.

Note: If SMT is enabled, Linux enumerates all threads as cpus. Since, the energy status registers are accessed at core level, reading those registers from the sibling threads would result in duplicate values. Hence, energy counter entries are not populated for the siblings.

7.22.2 Energy Caluclation

Energy information (in Joules) is based on the multiplier, 1/2^ESU; where ESU is an unsigned integer read from MSR_RAPL_POWER_UNIT register. Default value is 10000b, indicating energy status unit is 15.3 micro-Joules increment.

Reported values are scaled as per the formula

scaled value = ((1/2^ESU) * (Raw value) * 1000000UL) in uJoules

Users calculate power for a given domain by calculating dEnergy/dTime for that domain.

7.22.3 Energy accumulation

Current, Socket energy status register is 32bit, assuming a 240W 2P system, the register would wrap around in

2^32*15.3 e-6/240 * 2 = 547.60833024 secs to wrap(~9 mins)

The Core energy register may wrap around after several days.

To improve the wrap around time, a kernel thread is implemented to accumulate the socket energy counters and one core energy counter per run to a respective 64-bit counter. The kernel thread starts running during probe, wakes up every 100secs and stops running when driver is removed.

A socket and core energy read would return the current register value added to the respective energy accumulator.

7.22.4 Sysfs attributes

Attribute	Label	Description

- For index N between [1] and [nr_cpus]

energy[N]_input	EcoreX	Core Energy X = [0] to [nr_cpus - 1] Measured input core energy

- For N between [nr_cpus] and [nr_cpus + nr_socks]

energy[N]_input	Esocket X	Socket Energy X = [0] to [nr_socks -1] Measured input socket energy

7.23 Kernel driver asb100

Supported Chips:

- Asus ASB100 and ASB100-A “Bach”

  Prefix: ‘asb100’

  Addresses scanned: I2C 0x2d

  Datasheet: none released

Author: Mark M. Hoffman <mhoffman@lightlink.com>

7.23.1 Description

This driver implements support for the Asus ASB100 and ASB100-A “Bach”. These are custom ASICs available only on Asus mainboards. Asus refuses to supply a datasheet for these chips. Thanks go to many people who helped investigate their hardware, including:

Vitaly V. Bursov Alexander van Kaam (author of MBM for Windows) Bertrik Sikken

The ASB100 implements seven voltage sensors, three fan rotation speed sensors, four temperature sensors, VID lines and alarms. In addition to these, the ASB100-A also implements a single PWM controller for fans 2 and 3 (i.e. one setting controls both.) If you have a plain ASB100, the PWM controller will simply not work (or maybe it will for you···it doesn’t for me).

Temperatures are measured and reported in degrees Celsius.

Fan speeds are reported in RPM (rotations per minute). An alarm is triggered if the rotation speed has dropped below a programmable limit.

Voltage sensors (also known as IN sensors) report values in volts.

The VID lines encode the core voltage value: the voltage level your processor should work with. This is hardcoded by the mainboard and/or processor itself. It is a value in volts.

Alarms: (TODO question marks indicate may or may not work)

- 0x0001 => in0 (?)
- 0x0002 => in1 (?)
- 0x0004 => in2
- 0x0008 => in3
- 0x0010 => temp1[1]

[1] This alarm will only trigger if the hysteresis value is 127C. I.e. it behaves the same as w83781d.

- 0x0020 => temp2
- 0x0040 => fan1
- 0x0080 => fan2
- 0x0100 => in4
- 0x0200 => in5 (?)[2]
- 0x0400 => in6 (?)[2]
- 0x0800 => fan3
- 0x1000 => chassis switch
- 0x2000 => temp3

TODO:

- Experiment with fan divisors > 8.
- Experiment with temp. sensor types.
- Are there really 13 voltage inputs? Probably not…
- Cleanups, no doubt…

7.24 Kernel driver asc7621

Supported chips:

Andigilog aSC7621 and aSC7621a

Prefix: 'asc7621'

Addresses scanned: I2C 0x2c, 0x2d, 0x2e

Datasheet: http://www.fairview5.com/linux/asc7621/asc7621.pdf

Author: George Joseph

Description provided by Dave Pivin @ Andigilog:

Andigilog has both the PECI and pre-PECI versions of the Heceta-6, as Intel calls them. Heceta-6e has high frequency PWM and Heceta-6p has added PECI and a 4th thermal zone. The Andigilog aSC7611 is the Heceta-6e part and aSC7621 is the Heceta-6p part. They are both in volume production, shipping to Intel and their subs.

We have enhanced both parts relative to the governing Intel specification. First enhancement is temperature reading resolution. We have used registers below 20h for vendor-specific functions in addition to those in the Intel-specified vendor range.

Our conversion process produces a result that is reported as two bytes. The fan speed control uses this finer value to produce a "step-less" fan PWM output. These two bytes are "read-locked" to guarantee that once a high or low byte is read, the other byte is locked-in until after the next read of any register. So to get an atomic

[2] The min and max registers for these values appear to be read-only or otherwise stuck at 0x00.

reading, read high or low byte, then the very next read should be the opposite byte. Our data sheet says 10-bits of resolution, although you may find the lower bits are active, they are not necessarily reliable or useful externally. We chose not to mask them.

We employ significant filtering that is user tunable as described in the data sheet. Our temperature reports and fan PWM outputs are very smooth when compared to the competition, in addition to the higher resolution temperature reports. The smoother PWM output does not require user intervention.

We offer GPIO features on the former VID pins. These are open-drain outputs or inputs and may be used as general purpose I/O or as alarm outputs that are based on temperature limits. These are in 19h and 1Ah.

We offer flexible mapping of temperature readings to thermal zones. Any temperature may be mapped to any zone, which has a default assignment that follows Intel's specs.

Since there is a fan to zone assignment that allows for the "hotter" of a set of zones to control the PWM of an individual fan, but there is no indication to the user, we have added an indicator that shows which zone is currently controlling the PWM for a given fan. This is in register 00h.

Both remote diode temperature readings may be given an offset value such that the reported reading as well as the temperature used to determine PWM may be offset for system calibration purposes.

PECI Extended configuration allows for having more than two domains per PECI address and also provides an enabling function for each PECI address. One could use our flexible zone assignment to have a zone assigned to up to 4 PECI addresses. This is not possible in the default Intel configuration. This would be useful in multi-CPU systems with individual fans on each that would benefit from individual fan control. This is in register 0Eh.

The tachometer measurement system is flexible and able to adapt to many fan types. We can also support pulse-stretched PWM so that 3-wire fans may be used. These characteristics are in registers 04h to 07h.

Finally, we have added a tach disable function that turns off the tach measurement system for individual tachs in order to save power. That is in register 75h.

7.24.1 aSC7621 Product Description

The aSC7621 has a two wire digital interface compatible with SMBus 2.0. Using a 10-bit ADC, the aSC7621 measures the temperature of two remote diode connected transistors as well as its own die. Support for Platform Environmental Control Interface (PECI) is included.

Using temperature information from these four zones, an automatic fan speed control algorithm is employed to minimize acoustic impact while achieving recommended CPU temperature under varying operational loads.

To set fan speed, the aSC7621 has three independent pulse width modulation (PWM) outputs that are controlled by one, or a combination of three, temperature zones. Both high- and low-frequency PWM ranges are supported.

The aSC7621 also includes a digital filter that can be invoked to smooth temperature readings for better control of fan speed and minimum acoustic impact.

The aSC7621 has tachometer inputs to measure fan speed on up to four fans. Limit and status registers for all measured values are included to alert the system host that any measurements are outside of programmed limits via status registers.

System voltages of VCCP, 2.5V, 3.3V, 5.0V, and 12V motherboard power are monitored efficiently with internal scaling resistors.

Features

- Supports PECI interface and monitors internal and remote thermal diodes
- 2-wire, SMBus 2.0 compliant, serial interface
- 10-bit ADC
- Monitors VCCP, 2.5V, 3.3V, 5.0V, and 12V motherboard/processor supplies
- Programmable autonomous fan control based on temperature readings
- Noise filtering of temperature reading for fan speed control
- 0.25C digital temperature sensor resolution
- 3 PWM fan speed control outputs for 2-, 3- or 4-wire fans and up to 4 fan tachometer inputs
- Enhanced measured temperature to Temperature Zone assignment.
- Provides high and low PWM frequency ranges
- 3 GPIO pins for custom use
- 24-Lead QSOP package

7.24.2 Configuration Notes

Except where noted below, the sysfs entries created by this driver follow the standards defined in “sysfs-interface”.

temp1_source

0	(default) peci_legacy = 0, Remote 1 Temperature peci_legacy = 1, PECI Processor Temperature 0
1	Remote 1 Temperature
2	Remote 2 Temperature
3	Internal Temperature
4	PECI Processor Temperature 0
5	PECI Processor Temperature 1
6	PECI Processor Temperature 2
7	PECI Processor Temperature 3

temp2_source

0	(default) Internal Temperature
1	Remote 1 Temperature
2	Remote 2 Temperature
3	Internal Temperature
4	PECI Processor Temperature 0
5	PECI Processor Temperature 1
6	PECI Processor Temperature 2
7	PECI Processor Temperature 3

temp3_source

0	(default) Remote 2 Temperature
1	Remote 1 Temperature
2	Remote 2 Temperature
3	Internal Temperature
4	PECI Processor Temperature 0
5	PECI Processor Temperature 1
6	PECI Processor Temperature 2
7	PECI Processor Temperature 3

temp4_source

0	(default) peci_legacy = 0, PECI Processor Temperature 0 peci_legacy = 1, Remote 1 Temperature
1	Remote 1 Temperature
2	Remote 2 Temperature
3	Internal Temperature
4	PECI Processor Temperature 0
5	PECI Processor Temperature 1
6	PECI Processor Temperature 2
7	PECI Processor Temperature 3

temp[1-4]_smoothing_enable / temp[1-4]_smoothing_time Smooths spikes in temp readings caused by noise. Valid values in milliseconds are:

- 35000
- 17600
- 11800
- 7000
- 4400
- 3000
- 1600
- 800

temp[1-4]_crit When the corresponding zone temperature reaches this value, ALL pwm outputs will got to 100%.

temp[5-8]_input / temp[5-8]_enable The aSC7621 can also read temperatures provided by the processor via the PECI bus. Usually these are "core" temps and are relative to the point where the automatic thermal control circuit starts throttling. This means that these are usually negative numbers.

pwm[1-3]_enable

0	Fan off.
1	Fan on manual control.
2	Fan on automatic control and will run at the minimum pwm if the temperature for the zone is below the minimum.
3	Fan on automatic control but will be off if the temperature for the zone is below the minimum.
4-254	Ignored.
255	Fan on full.

pwm[1-3]_auto_channels Bitmap as described in sysctl-interface with the following exceptions···

Only the following combination of zones (and their corresponding masks) are valid:

- 1
- 2
- 3
- 2,3
- 1,2,3
- 4
- 1,2,3,4
- Special values:

0	Disabled.
16	Fan on manual control.
31	Fan on full.

pwm[1-3]_invert When set, inverts the meaning of pwm[1-3]. i.e. when pwm = 0, the fan will be on full and when pwm = 255 the fan will be off.

pwm[1-3]_freq PWM frequency in Hz Valid values in Hz are:

- 10
- 15
- 23
- 30 (default)
- 38
- 47

- 62
- 94
- 23000
- 24000
- 25000
- 26000
- 27000
- 28000
- 29000
- 30000

Setting any other value will be ignored.

peci_enable Enables or disables PECI

peci_avg Input filter average time.

- 0 0 Sec. (no Smoothing) (default)
- 1 0.25 Sec.
- 2 0.5 Sec.
- 3 1.0 Sec.
- 4 2.0 Sec.
- 5 4.0 Sec.
- 6 8.0 Sec.
- 7 0.0 Sec.

peci_legacy

0	Standard Mode (default) Remote Diode 1 reading is associated with Temperature Zone 1, PECI is associated with Zone 4
1	Legacy Mode PECI is associated with Temperature Zone 1, Remote Diode 1 is associated with Zone 4

peci_diode Diode filter

0	0.25 Sec.
1	1.1 Sec.
2	2.4 Sec. (default)
3	3.4 Sec.
4	5.0 Sec.
5	6.8 Sec.
6	10.2 Sec.
7	16.4 Sec.

peci_4domain Four domain enable

0	1 or 2 Domains for enabled processors (default)
1	3 or 4 Domains for enabled processors

peci_domain Domain

0	Processor contains a single domain (0) (default)
1	Processor contains two domains (0,1)

7.25 Kernel driver aspeed-pwm-tacho

Supported chips: ASPEED AST2400/2500

Authors: <jaghu@google.com>

7.25.1 Description:

This driver implements support for ASPEED AST2400/2500 PWM and Fan Tacho controller. The PWM controller supports upto 8 PWM outputs. The Fan tacho controller supports up to 16 tachometer inputs.

The driver provides the following sensor accesses in sysfs:

fanX_input	ro	provide current fan rotation value in RPM as reported by the fan to the device.
pwmX	rw	get or set PWM fan control value. This is an integer value between 0(off) and 255(full speed).

7.26 Broadcom BCM54140 Quad SGMII/QSGMII PHY

Supported chips:

- Broadcom BCM54140

  Datasheet: not public

Author: Michael Walle <michael@walle.cc>

7.26.1 Description

The Broadcom BCM54140 is a Quad SGMII/QSGMII PHY which supports monitoring its die temperature as well as two analog voltages.

The AVDDL is a 1.0V analogue voltage, the AVDDH is a 3.3V analogue voltage. Both voltages and the temperature are measured in a round-robin fashion.

7.26.2 Sysfs entries

The following attributes are supported.

in0_label	“AVDDL”
in0_input	Measured AVDDL voltage.
in0_min	Minimum AVDDL voltage.
in0_max	Maximum AVDDL voltage.
in0_alarm	AVDDL voltage alarm.
in1_label	“AVDDH”
in1_input	Measured AVDDH voltage.
in1_min	Minimum AVDDH voltage.
in1_max	Maximum AVDDH voltage.
in1_alarm	AVDDH voltage alarm.
temp1_input	Die temperature.
temp1_min	Minimum die temperature.
temp1_max	Maximum die temperature.
temp1_alarm	Die temperature alarm.

7.27 Kernel driver bel-pfe

Supported chips:

- BEL PFE1100

  Prefixes: ‘pfe1100’

  Addresses scanned: -

  Datasheet: https://www.belfuse.com/resources/datasheets/powersolutions/ds-bps-pfe1100-12-054xa.pdf

- BEL PFE3000

  Prefixes: ‘pfe3000’

  Addresses scanned: -

  Datasheet: https://www.belfuse.com/resources/datasheets/powersolutions/ds-bps-pfe3000-series.pdf

Author: Tao Ren <rentao.bupt@gmail.com>

7.27.1 Description

This driver supports hardware monitoring for below power supply devices which support PMBus Protocol:

- BEL PFE1100

  1100 Watt AC to DC power-factor-corrected (PFC) power supply. PMBus Communication Manual is not publicly available.

- BEL PFE3000

  3000 Watt AC/DC power-factor-corrected (PFC) and DC-DC power supply. PMBus Communication Manual is not publicly available.

The driver is a client driver to the core PMBus driver. Please see Documentation/hwmon/pmbus.rst for details on PMBus client drivers.

7.27.2 Usage Notes

This driver does not auto-detect devices. You will have to instantiate the devices explicitly. Please see Documentation/i2c/instantiating-devices.rst for details.

Example: the following will load the driver for an PFE3000 at address 0x20 on I2C bus #1:

```
$ modprobe bel-pfe
$ echo pfe3000 0x20 > /sys/bus/i2c/devices/i2c-1/new_device
```

7.27.3 Platform data support

The driver supports standard PMBus driver platform data.

7.27.4 Sysfs entries

curr1_label	“iin”
curr1_input	Measured input current
curr1_max	Input current max value
curr1_max_alarm	Input current max alarm
curr[2-3]_label	“iout[1-2]”
curr[2-3]_input	Measured output current
curr[2-3]_max	Output current max value
curr[2-3]_max_alarm	Output current max alarm
fan[1-2]_input	Fan 1 and 2 speed in RPM
fan1_target	Set fan speed reference for both fans
in1_label	“vin”
in1_input	Measured input voltage
in1_crit	Input voltage critical max value
in1_crit_alarm	Input voltage critical max alarm
in1_lcrit	Input voltage critical min value
in1_lcrit_alarm	Input voltage critical min alarm
in1_max	Input voltage max value
in1_max_alarm	Input voltage max alarm
in2_label	“vcap”
in2_input	Hold up capacitor voltage
in[3-8]_label	“vout[1-3,5-7]”
in[3-8]_input	Measured output voltage
in[3-4]_alarm	vout[1-2] output voltage alarm
power[1-2]_label	“pin[1-2]”
power[1-2]_input	Measured input power
power[1-2]_alarm	Input power high alarm
power[3-4]_label	“pout[1-2]”
power[3-4]_input	Measured output power
temp[1-3]_input	Measured temperature
temp[1-3]_alarm	Temperature alarm

Note:

- curr3, fan2, vout[2-7], vcap, pin2, pout2 and temp3 attributes only exist for PFE3000.

7.28 Kernel driver bt1-pvt

Supported chips:

- Baikal-T1 PVT sensor (in SoC)

  Prefix: ‘bt1-pvt’

  Addresses scanned: -

  Datasheet: Provided by BAIKAL ELECTRONICS upon request and under NDA

Authors: Maxim Kaurkin <maxim.kaurkin@baikalelectronics.ru> Serge Semin <Sergey.Semin@baikalelectronics.ru>

7.28.1 Description

This driver implements support for the hardware monitoring capabilities of the embedded into Baikal-T1 process, voltage and temperature sensors. PVT IP-core consists of one temperature and four voltage sensors, which can be used to monitor the chip internal environment like heating, supply voltage and transistors performance. The driver can optionally provide the hwmon alarms for each sensor the PVT controller supports. The alarms functionality is made compile-time configurable due to the hardware interface implementation peculiarity, which is connected with an ability to convert data from only one sensor at a time. Additional limitation is that the controller performs the thresholds checking synchronously with the data conversion procedure. Due to these in order to have the hwmon alarms automatically detected the driver code must switch from one sensor to another, read converted data and manually check the threshold status bits. Depending on the measurements timeout settings (update_interval sysfs node value) this design may cause additional burden on the system performance. So in case if alarms are unnecessary in your system design it’s recommended to have them disabled to prevent the PVT IRQs being periodically raised to get the data cache/alarms status up to date. By default in alarm-less configuration the data conversion is performed by the driver on demand when read operation is requested via corresponding _input-file.

7.28.2 Temperature Monitoring

Temperature is measured with 10-bit resolution and reported in millidegree Celsius. The driver performs all the scaling by itself therefore reports true temperatures that don’t need any user-space adjustments. While the data translation formulae isn’t linear, which gives us non-linear discreteness, it’s close to one, but giving a bit better accuracy for higher temperatures. The temperature input is mapped as follows (the last column indicates the input ranges):

```
temp1: CPU embedded diode       -48.38C - +147.438C
```

In case if the alarms kernel config is enabled in the driver the temperature input has associated min and max limits which trigger an alarm when crossed.

7.28.3 Voltage Monitoring

The voltage inputs are also sampled with 10-bit resolution and reported in millivolts. But in this case the data translation formulae is linear, which provides a constant measurements discreteness. The data scaling is also performed by the driver, so returning true millivolts. The voltage inputs are mapped as follows (the last column indicates the input ranges):

```
in0: VDD                (processor core)                0.62V - 1.168V
in1: Low-Vt             (low voltage threshold)         0.62V - 1.168V
```

(continues on next page)

(continued from previous page)

```
in2: High-Vt            (high voltage threshold)        0.62V - 1.168V
in3: Standard-Vt        (standard voltage threshold)    0.62V - 1.168V
```

In case if the alarms config is enabled in the driver the voltage inputs have associated min and max limits which trigger an alarm when crossed.

7.28.4 Sysfs Attributes

Following is a list of all sysfs attributes that the driver provides, their permissions and a short description:

Name	Perm	Description
update_interval	RW	Measurements update interval per sensor.
temp1_type	RO	Sensor type (always 1 as CPU embedded diode).
temp1_label	RO	CPU Core Temperature sensor.
temp1_input	RO	Measured temperature in millidegree Celsius.
temp1_min	RW	Low limit for temp input.
temp1_max	RW	High limit for temp input.
temp1_min_alarm	RO	Temperature input alarm. Returns 1 if temperature input went below min limit, 0 otherwise.
temp1_max_alarm	RO	Temperature input alarm. Returns 1 if temperature input went above max limit, 0 otherwise.
temp1_offset	RW	Temperature offset in millidegree Celsius which is added to the temperature reading by the chip. It can be used to manually adjust the temperature measurements within 7.130 degrees Celsius.
in[0-3]_label	RO	CPU Voltage sensor (either core or low/high/standard thresholds).
in[0-3]_input	RO	Measured voltage in millivolts.
in[0-3]_min	RW	Low limit for voltage input.
in[0-3]_max	RW	High limit for voltage input.
in[0-3]_min_alarm	RO	Voltage input alarm. Returns 1 if voltage input went below min limit, 0 otherwise.
in[0-3]_max_alarm	RO	Voltage input alarm. Returns 1 if voltage input went above max limit, 0 otherwise.

7.29 Kernel driver coretemp

Supported chips:

- All Intel Core family

  Prefix: 'coretemp'

  CPUID: family 0x6, models

  - 0xe (Pentium M DC), 0xf (Core 2 DC 65nm),
  - 0x16 (Core 2 SC 65nm), 0x17 (Penryn 45nm),
  - 0x1a (Nehalem), 0x1c (Atom), 0x1e (Lynnfield),
  - 0x26 (Tunnel Creek Atom), 0x27 (Medfield Atom),
  - 0x36 (Cedar Trail Atom)

  Datasheet:

  Intel 64 and IA-32 Architectures Software Developer's Manual Volume 3A: System Programming Guide

  http://softwarecommunity.intel.com/Wiki/Mobility/720.htm

Author: Rudolf Marek

7.29.1 Description

This driver permits reading the DTS (Digital Temperature Sensor) embedded inside Intel CPUs. This driver can read both the per-core and per-package temperature using the appropriate sensors. The per-package sensor is new; as of now, it is present only in the SandyBridge platform. The driver will show the temperature of all cores inside a package under a single device directory inside hwmon.

Temperature is measured in degrees Celsius and measurement resolution is 1 degree C. Valid temperatures are from 0 to TjMax degrees C, because the actual value of temperature register is in fact a delta from TjMax.

Temperature known as TjMax is the maximum junction temperature of processor, which depends on the CPU model. See table below. At this temperature, protection mechanism will perform actions to forcibly cool down the processor. Alarm may be raised, if the temperature grows enough (more than TjMax) to trigger the Out-Of-Spec bit. Following table summarizes the exported sysfs files:

All Sysfs entries are named with their core_id (represented here by 'X').

tempX_input	Core temperature (in millidegrees Celsius).
tempX_max	All cooling devices should be turned on (on Core2).
tempX_crit	Maximum junction temperature (in millidegrees Celsius).
tempX_crit_alarm	Set when Out-of-spec bit is set, never clears. Correct CPU operation is no longer guaranteed.
tempX_label	Contains string "Core X", where X is processor number. For Package temp, this will be "Physical id Y", where Y is the package number.

On CPU models which support it, TjMax is read from a model-specific register. On other models, it is set to an arbitrary value based on weak heuristics. If these heuristics don’ t work for you, you can pass the correct TjMax value as a module parameter (tjmax).

Appendix A. Known TjMax lists (TBD): Some information comes from ark.intel.com

Process	Processor	TjMax(C)
22nm	Core i5/i7 Processors i7 3920XM, 3820QM, 3720QM, 3667U, 3520M i5 3427U, 3360M/3320M i7 3770/3770K i5 3570/3570K, 3550, 3470/3450 i7 3770S i5 3570S/3550S, 3475S/3470S/3450S i7 3770T i5 3570T i5 3470T	105 105 105 105 103 103 94 94 91
32nm	Core i3/i5/i7 Processors i7 2600 i7 660UM/640/620, 640LM/620, 620M, 610E i5 540UM/520/430, 540M/520/450/430 i3 330E, 370M/350/330 i3 330UM	98 105 105 90 rPGA, 105 BGA 105
32nm	Core i7 Extreme Processors 980X	100
32nm	Celeron Processors U3400 P4505/P4500	105 90
32nm	Atom Processors S1260/1220 S1240 Z2460 Z2760 D2700/2550/2500 N2850/2800/2650/2600	95 102 90 90 100 100
45nm	Xeon Processors 5400 Quad-Core X5492, X5482, X5472, X5470, X5460, X5450 E5472, E5462, E5450/40/30/20/10/05 L5408 L5430, L5420, L5410	85 85 95 70
45nm	Xeon Processors 5200 Dual-Core X5282, X5272, X5270, X5260 E5240 E5205, E5220 L5240 L5238, L5215	90 90 70, 90 70 95
45nm	Atom Processors D525/510/425/410 K525/510/425/410 Z670/650 Z560/550/540/530P/530/520PT/520/515/510PT/510P Z510/500 N570/550 N475/470/455/450 N280/270 330/230 E680/660/640/620 E680T/660T/640T/620T E665C/645C E665CT/645CT CE4170/4150/4110 CE4200 series CE5300 series	100 100 90 90 90 100 100 90 125 90 110 90 110 110 unknown unknown
45nm	Core2 Processors Solo ULV SU3500/3300 T9900/9800/9600/9550/9500/9400/9300/8300/8100 T6670/6500/6400 T6600 SU9600/9400/9300 SP9600/9400 SL9600/9400/9380/9300 P9700/9600/9500/8800/8700/8600/8400/7570 P7550/7450	100 105 105 90 105 105 105 105 90
45nm	Core2 Quad Processors Q9100/9000	100
45nm	Core2 Extreme Processors X9100/9000 QX9300	105 100
45nm	Core i3/i5/i7 Processors i7 940XM/920 i7 840QM/820/740/720	100 100
45nm	Celeron Processors SU2300 900	100 105
65nm	Core2 Duo Processors Solo U2200, U2100 U7700/7600/7500 T7800/7700/7600/7500/7400/7300/7250/7200/7100 T5870/5670/5600/5550/5500/5470/5450/5300/5270 T5250 T5800/5750/5200 L7700/7500/7400/7300/7200	100 100 100 100 100 85 100
65nm	Core2 Extreme Processors X7900/7800	100
65nm	Core Duo Processors U2500/2400 T2700/2600/2450/2400/2350/2300E/2300/2250/2050 L2500/2400/2300	100 100 100
65nm	Core Solo Processors U1500/1400/1300 T1400/1350/1300/1250	100 100
65nm	Xeon Processors 5000 Quad-Core X5000 E5000 L5000 L5318	90-95 80 70 95
65nm	Xeon Processors 5000 Dual-Core 5080, 5063, 5060, 5050, 5030 5160, 5150, 5148, 5140, 5130, 5120, 5110 L5138	80-90 80 100
65nm	Celeron Processors T1700/1600 560/550/540/530	100 100

7.30 Kernel driver da9052

Supported chips:

- Dialog Semiconductors DA9052-BC and DA9053-AA/Bx PMICs

  Prefix: 'da9052'

  Datasheet: Datasheet is not publicly available.

Authors: David Dajun Chen <dchen@diasemi.com>

7.30.1 Description

The DA9052/53 provides an Analogue to Digital Converter (ADC) with 10 bits resolution and track and hold circuitry combined with an analogue input multiplexer. The analogue input multiplexer will allow conversion of up to 10 different inputs. The track and hold circuit ensures stable input voltages at the input of the ADC during the conversion.

The ADC is used to measure the following inputs:

Channel 0	VDDOUT - measurement of the system voltage
Channel 1	ICH - internal battery charger current measurement
Channel 2	TBAT - output from the battery NTC
Channel 3	VBAT - measurement of the battery voltage
Channel 4	ADC_IN4 - high impedance input (0 - 2.5V)
Channel 5	ADC_IN5 - high impedance input (0 - 2.5V)
Channel 6	ADC_IN6 - high impedance input (0 - 2.5V)
Channel 7	XY - TSI interface to measure the X and Y voltage of the touch screen resistive potentiometers
Channel 8	Internal Tjunc. - sense (internal temp. sensor)
Channel 9	VBBAT - measurement of the backup battery voltage

By using sysfs attributes we can measure the system voltage VDDOUT, the battery charging current ICH, battery temperature TBAT, battery junction temperature TJUNC, battery voltage VBAT and the back up battery voltage VBBAT.

7.30.2 Voltage Monitoring

Voltages are sampled by a 10 bit ADC.

The battery voltage is calculated as:

    Milli volt = ((ADC value * 1000) / 512) + 2500

The backup battery voltage is calculated as:

    Milli volt = (ADC value * 2500) / 512;

The voltages on ADC channels 4, 5 and 6 are calculated as:

    Milli volt = (ADC value * 2500) / 1023

7.30.3 Temperature Monitoring

Temperatures are sampled by a 10 bit ADC. Junction and battery temperatures are monitored by the ADC channels.

The junction temperature is calculated:

    Degrees celsius = 1.708 * (TJUNC_RES - T_OFFSET) - 108.8

The junction temperature attribute is supported by the driver.

The battery temperature is calculated:

    Degree Celsius = 1 / (t1 + 1/298) - 273

where t1 = (1/B)* ln((ADCval * 2.5)/(R25*ITBAT*255))

Default values of R25, B, ITBAT are 10e3, 3380 and 50e-6 respectively.

7.31 Kernel driver da9055

Supported chips:

- Dialog Semiconductors DA9055 PMIC

  Prefix: ‘da9055’

  Datasheet: Datasheet is not publicly available.

Authors: David Dajun Chen <dchen@diasemi.com>

7.31.1 Description

The DA9055 provides an Analogue to Digital Converter (ADC) with 10 bits resolution and track and hold circuitry combined with an analogue input multiplexer. The analogue input multiplexer will allow conversion of up to 5 different inputs. The track and hold circuit ensures stable input voltages at the input of the ADC during the conversion.

The ADC is used to measure the following inputs:

- Channel 0: VDDOUT - measurement of the system voltage
- Channel 1: ADC_IN1 - high impedance input (0 - 2.5V)
- Channel 2: ADC_IN2 - high impedance input (0 - 2.5V)
- Channel 3: ADC_IN3 - high impedance input (0 - 2.5V)
- Channel 4: Internal Tjunc. - sense (internal temp. sensor)

By using sysfs attributes we can measure the system voltage VDDOUT, chip junction temperature and auxiliary channels voltages.

7.31.2 Voltage Monitoring

Voltages are sampled in a AUTO mode it can be manually sampled too and results are stored in a 10 bit ADC.

The system voltage is calculated as:

    Milli volt = ((ADC value * 1000) / 85) + 2500

The voltages on ADC channels 1, 2 and 3 are calculated as:

    Milli volt = (ADC value * 1000) / 102

7.31.3 Temperature Monitoring

Temperatures are sampled by a 10 bit ADC. Junction temperatures are monitored by the ADC channels.

The junction temperature is calculated:

    Degrees celsius = -0.4084 * (ADC_RES - T_OFFSET) + 307.6332

The junction temperature attribute is supported by the driver.

7.32 Kernel driver dell-smm-hwmon

7.32.1 Description

On many Dell laptops the System Management Mode (SMM) BIOS can be queried for the status of fans and temperature sensors. Userspace utilities like `sensors` can be used to return the readings. The userspace suite i8kutils can also be used to read the sensors and automatically adjust fan speed (please notice that it currently uses the deprecated `/proc/i8k` interface).

7.32.2 `sysfs` interface

Temperature sensors and fans can be queried and set via the standard `hwmon` interface on `sysfs`, under the directory `/sys/class/hwmon/hwmonX` for some value of `X` (search for the `X` such that `/sys/class/hwmon/hwmonX/name` has content `dell_smm`). A number of other attributes can be read or written:

Name	Perm	Description
fan[1-3]_input	RO	Fan speed in RPM.
fan[1-3]_label	RO	Fan label.
pwm[1-3]	RW	Control the fan PWM duty-cycle.
pwm1_enable	WO	Enable or disable automatic BIOS fan control (not supported on all laptops, see below for details).
temp[1-10]_input	RO	Temperature reading in milli-degrees Celsius.
temp[1-10]_label	RO	Temperature sensor label.

7.32.3 Disabling automatic BIOS fan control

On some laptops the BIOS automatically sets fan speed every few seconds. Therefore the fan speed set by mean of this driver is quickly overwritten.

There is experimental support for disabling automatic BIOS fan control, at least on laptops where the corresponding SMM command is known, by writing the value `1` in the attribute `pwm1_enable` (writing `2` enables automatic BIOS control again). Even if you have more than one fan, all of them are set to either enabled or disabled automatic fan control at the same time and, notwithstanding the name, `pwm1_enable` sets automatic control for all fans.

If `pwm1_enable` is not available, then it means that SMM codes for enabling and disabling automatic BIOS fan control are not whitelisted for your hardware. It is

possible that codes that work for other laptops actually work for yours as well, or that you have to discover new codes.

Check the list `i8k_whitelist_fan_control` in file `drivers/hwmon/dell-smm-hwmon.c` in the kernel tree: as a first attempt you can try to add your machine and use an already-known code pair. If, after recompiling the kernel, you see that `pwm1_enable` is present and works (i.e., you can manually control the fan speed), then please submit your finding as a kernel patch, so that other users can benefit from it. Please see Documentation/process/submitting-patches.rst for information on submitting patches.

If no known code works on your machine, you need to resort to do some probing, because unfortunately Dell does not publish datasheets for its SMM. You can experiment with the code in this repository to probe the BIOS on your machine and discover the appropriate codes.

Again, when you find new codes, we'd be happy to have your patches!

7.32.4 Module parameters

- **force:bool** Force loading without checking for supported models. (default: 0)
- **ignore_dmi:bool** Continue probing hardware even if DMI data does not match. (default: 0)
- **restricted:bool** Allow fan control only to processes with the `CAP_SYS_ADMIN` capability set or processes run as root when using the legacy `/proc/i8k` interface. In this case normal users will be able to read temperature and fan status but not to control the fan. If your notebook is shared with other users and you don't trust them you may want to use this option. (default: 1, only available with `CONFIG_I8K`)
- **power_status:bool** Report AC status in `/proc/i8k`. (default: 0, only available with `CONFIG_I8K`)
- **fan_mult:uint** Factor to multiply fan speed with. (default: autodetect)
- **fan_max:uint** Maximum configurable fan speed. (default: autodetect)

7.32.5 Legacy `/proc` interface

> **Warning:** This interface is obsolete and deprecated and should not used in new applications. This interface is only available when kernel is compiled with option `CONFIG_I8K`.

The information provided by the kernel driver can be accessed by simply reading the `/proc/i8k` file. For example:

```
$ cat /proc/i8k
1.0 A17 2J59L02 52 2 1 8040 6420 1 2
```

The fields read from /proc/i8k are:

```
1.0 A17 2J59L02 52 2 1 8040 6420 1 2
|   |   |       |  | | |    |    | |
|   |   |       |  | | |    |    | +------- 10. buttons status
|   |   |       |  | | |    |    +--------- 9.  AC status
|   |   |       |  | | |    +-------------- 8.  fan0 RPM
|   |   |       |  | | +------------------- 7.  fan1 RPM
|   |   |       |  | +--------------------- 6.  fan0 status
|   |   |       |  +----------------------- 5.  fan1 status
|   |   |       +-------------------------- 4.  temp0 reading (Celsius)
|   |   +---------------------------------- 3.  Dell service tag (later␣
↪known as 'serial number')
|   +------------------------------------- 2.  BIOS version
+----------------------------------------- 1.  /proc/i8k format version
```

A negative value, for example -22, indicates that the BIOS doesn't return the corresponding information. This is normal on some models/BIOSes.

For performance reasons the /proc/i8k doesn't report by default the AC status since this SMM call takes a long time to execute and is not really needed. If you want to see the ac status in /proc/i8k you must explictitly enable this option by passing the power_status=1 parameter to insmod. If AC status is not available -1 is printed instead.

The driver provides also an ioctl interface which can be used to obtain the same information and to control the fan status. The ioctl interface can be accessed from C programs or from shell using the i8kctl utility. See the source file of i8kutils for more information on how to use the ioctl interface.

7.33 Kernel driver dme1737

Supported chips:

- SMSC DME1737 and compatibles (like Asus A8000)

  Prefix: 'dme1737'

  Addresses scanned: I2C 0x2c, 0x2d, 0x2e

  Datasheet: Provided by SMSC upon request and under NDA

- SMSC SCH3112, SCH3114, SCH3116

  Prefix: 'sch311x'

  Addresses scanned: none, address read from Super-I/O config space

  Datasheet: Available on the Internet

- SMSC SCH5027

  Prefix: 'sch5027'

  Addresses scanned: I2C 0x2c, 0x2d, 0x2e

  Datasheet: Provided by SMSC upon request and under NDA

- SMSC SCH5127

  Prefix: 'sch5127'

  Addresses scanned: none, address read from Super-I/O config space

  Datasheet: Provided by SMSC upon request and under NDA

Authors: Juerg Haefliger <juergh@gmail.com>

7.33.1 Module Parameters

- **force_start: bool** Enables the monitoring of voltage, fan and temp inputs and PWM output control functions. Using this parameter shouldn't be required since the BIOS usually takes care of this.
- **probe_all_addr: bool** Include non-standard LPC addresses 0x162e and 0x164e when probing for ISA devices. This is required for the following boards: - VIA EPIA SN18000

7.33.2 Description

This driver implements support for the hardware monitoring capabilities of the SMSC DME1737 and Asus A8000 (which are the same), SMSC SCH5027, SCH311x, and SCH5127 Super-I/O chips. These chips feature monitoring of 3 temp sensors temp[1-3] (2 remote diodes and 1 internal), 8 voltages in[0-7] (7 external and 1 internal) and up to 6 fan speeds fan[1-6]. Additionally, the chips implement up to 5 PWM outputs pwm[1-3,5-6] for controlling fan speeds both manually and automatically.

For the DME1737, A8000 and SCH5027, fan[1-2] and pwm[1-2] are always present. Fan[3-6] and pwm[3,5-6] are optional features and their availability depends on the configuration of the chip. The driver will detect which features are present during initialization and create the sysfs attributes accordingly.

For the SCH311x and SCH5127, fan[1-3] and pwm[1-3] are always present and fan[4-6] and pwm[5-6] don't exist.

The hardware monitoring features of the DME1737, A8000, and SCH5027 are only accessible via SMBus, while the SCH311x and SCH5127 only provide access via the ISA bus. The driver will therefore register itself as an I2C client driver if it detects a DME1737, A8000, or SCH5027 and as a platform driver if it detects a SCH311x or SCH5127 chip.

7.33.3 Voltage Monitoring

The voltage inputs are sampled with 12-bit resolution and have internal scaling resistors. The values returned by the driver therefore reflect true millivolts and don't need scaling. The voltage inputs are mapped as follows (the last column indicates the input ranges):

DME1737, A8000:

```
in0: +5VTR      (+5V standby)           0V - 6.64V
in1: Vccp       (processor core)        0V - 3V
in2: VCC        (internal +3.3V)        0V - 4.38V
in3: +5V                                0V - 6.64V
in4: +12V                               0V - 16V
in5: VTR        (+3.3V standby)         0V - 4.38V
in6: Vbat       (+3.0V)                 0V - 4.38V
```

SCH311x:

```
in0: +2.5V                              0V - 3.32V
in1: Vccp       (processor core)        0V - 2V
in2: VCC        (internal +3.3V)        0V - 4.38V
in3: +5V                                0V - 6.64V
in4: +12V                               0V - 16V
in5: VTR        (+3.3V standby)         0V - 4.38V
in6: Vbat       (+3.0V)                 0V - 4.38V
```

SCH5027:

```
in0: +5VTR      (+5V standby)           0V - 6.64V
in1: Vccp       (processor core)        0V - 3V
in2: VCC        (internal +3.3V)        0V - 4.38V
in3: V2_IN                              0V - 1.5V
in4: V1_IN                              0V - 1.5V
in5: VTR        (+3.3V standby)         0V - 4.38V
in6: Vbat       (+3.0V)                 0V - 4.38V
```

SCH5127:

```
in0: +2.5                               0V - 3.32V
in1: Vccp       (processor core)        0V - 3V
in2: VCC        (internal +3.3V)        0V - 4.38V
in3: V2_IN                              0V - 1.5V
in4: V1_IN                              0V - 1.5V
in5: VTR        (+3.3V standby)         0V - 4.38V
in6: Vbat       (+3.0V)                 0V - 4.38V
in7: Vtrip      (+1.5V)                 0V - 1.99V
```

Each voltage input has associated min and max limits which trigger an alarm when crossed.

7.33.4 Temperature Monitoring

Temperatures are measured with 12-bit resolution and reported in millidegree Celsius. The chip also features offsets for all 3 temperature inputs which - when programmed - get added to the input readings. The chip does all the scaling by itself and the driver therefore reports true temperatures that don't need any user-space adjustments. The temperature inputs are mapped as follows (the last column indicates the input ranges):

```
temp1: Remote diode 1 (3904 type) temperature   -127C - +127C
temp2: DME1737 internal temperature             -127C - +127C
temp3: Remote diode 2 (3904 type) temperature   -127C - +127C
```

Each temperature input has associated min and max limits which trigger an alarm when crossed. Additionally, each temperature input has a fault attribute that returns 1 when a faulty diode or an unconnected input is detected and 0 otherwise.

7.33.5 Fan Monitoring

Fan RPMs are measured with 16-bit resolution. The chip provides inputs for 6 fan tachometers. All 6 inputs have an associated min limit which triggers an alarm when crossed. Fan inputs 1-4 provide type attributes that need to be set to the number of pulses per fan revolution that the connected tachometer generates. Supported values are 1, 2, and 4. Fan inputs 5-6 only support fans that generate 2 pulses per revolution. Fan inputs 5-6 also provide a max attribute that needs to be set to the maximum attainable RPM (fan at 100% duty- cycle) of the input. The chip adjusts the sampling rate based on this value.

7.33.6 PWM Output Control

This chip features 5 PWM outputs. PWM outputs 1-3 are associated with fan inputs 1-3 and PWM outputs 5-6 are associated with fan inputs 5-6. PWM outputs 1-3 can be configured to operate either in manual or automatic mode by setting the appropriate enable attribute accordingly. PWM outputs 5-6 can only operate in manual mode, their enable attributes are therefore read-only. When set to manual mode, the fan speed is set by writing the duty-cycle value to the appropriate PWM attribute. In automatic mode, the PWM attribute returns the current duty-cycle as set by the fan controller in the chip. All PWM outputs support the setting of the output frequency via the freq attribute.

In automatic mode, the chip supports the setting of the PWM ramp rate which defines how fast the PWM output is adjusting to changes of the associated temperature input. Associating PWM outputs to temperature inputs is done via temperature zones. The chip features 3 zones whose assignments to temperature inputs is static and determined during initialization. These assignments can be retrieved via the zone[1-3]_auto_channels_temp attributes. Each PWM output is assigned to one (or hottest of multiple) temperature zone(s) through the pwm[1-3]_auto_channels_zone attributes. Each PWM output has 3 distinct output duty-cycles: full, low, and min. Full is internally hard-wired to 255 (100%) and low and min can be programmed via pwm[1-3]_auto_point1_pwm and pwm[1-3]_auto_pwm_min, respectively. The thermal

thresholds of the zones are programmed via zone[1-3]_auto_point[1-3]_temp and zone[1-3]_auto_point1_temp_hyst:

pwm[1-3]_auto_point2_pwm	full-speed duty-cycle (255, i.e., 100%)
pwm[1-3]_auto_point1_pwm	low-speed duty-cycle
pwm[1-3]_auto_pwm_min	min-speed duty-cycle
zone[1-3]_auto_point3_temp	full-speed temp (all outputs)
zone[1-3]_auto_point2_temp	full-speed temp
zone[1-3]_auto_point1_temp	low-speed temp
zone[1-3]_auto_point1_temp_hyst	min-speed temp

The chip adjusts the output duty-cycle linearly in the range of auto_point1_pwm to auto_point2_pwm if the temperature of the associated zone is between auto_point1_temp and auto_point2_temp. If the temperature drops below the auto_point1_temp_hyst value, the output duty-cycle is set to the auto_pwm_min value which only supports two values: 0 or auto_point1_pwm. That means that the fan either turns completely off or keeps spinning with the low-speed duty-cycle. If any of the temperatures rise above the auto_point3_temp value, all PWM outputs are set to 100% duty-cycle.

Following is another representation of how the chip sets the output duty-cycle based on the temperature of the associated thermal zone:

Temperature	Duty-Cycle Rising Temp	Duty-Cycle Falling Temp
full-speed	full-speed	full-speed
•	< linearly adjusted duty-cycle >	•
low-speed	low-speed	low-speed
•	min-speed	low-speed
min-speed	min-speed	min-speed
•	min-speed	min-speed

7.33.7 Sysfs Attributes

Following is a list of all sysfs attributes that the driver provides, their permissions and a short description:

Name	Perm	Description
cpu0_vid	RO	CPU core reference voltage in millivolts.
vrm	RW	Voltage regulator module version number.
in[0-7]_input	RO	Measured voltage in millivolts.
in[0-7]_min	RW	Low limit for voltage input.
in[0-7]_max	RW	High limit for voltage input.
in[0-7]_alarm	RO	Voltage input alarm. Returns 1 if voltage input is or went outside the associated min-max range, 0 otherwise.
temp[1-3]_input	RO	Measured temperature in millidegree Celsius.
temp[1-3]_min	RW	Low limit for temp input.
temp[1-3]_max	RW	High limit for temp input.
temp[1-3]_offset	RW	Offset for temp input. This value will be added by the chip to the measured temperature.
temp[1-3]_alarm	RO	Alarm for temp input. Returns 1 if temp input is or went outside the associated min-max range, 0 otherwise.
temp[1-3]_fault	RO	Temp input fault. Returns 1 if the chip detects a faulty thermal diode or an unconnected temp input, 0 otherwise.
zone[1-3]_auto_channels_temp	RO	Temperature zone to temperature input mapping. This attribute is a bitfield and supports the following values: • 1: temp1 • 2: temp2 • 4: temp3
zone[1-3]_auto_point1_temp_hyst	RW	Auto PWM temp point1 hysteresis. The output of the corresponding PWM is set to the pwm_auto_min value if the temp falls below the auto_point1_temp_hyst value.
zone[1-3]_auto_point[1-3]_temp	RW	Auto PWM temp points. Auto_point1 is the low-speed temp, auto_point2 is the full-speed temp, and auto_point3 is the

7.33.8 Chip Differences

Feature	dme1737	sch311x	sch5027	sch5127
temp[1-3]_offset	yes	yes		
vid	yes			
zone3	yes	yes	yes	
zone[1-3]_hyst	yes	yes		
pwm min/off	yes	yes		
fan3	opt	yes	opt	yes
pwm3	opt	yes	opt	yes
fan4	opt		opt	
fan5	opt		opt	
pwm5	opt		opt	
fan6	opt		opt	
pwm6	opt		opt	
in7				yes

7.34 Kernel driver drivetemp

7.34.1 References

ANS T13/1699-D Information technology - AT Attachment 8 - ATA/ATAPI Command Set (ATA8-ACS)

ANS Project T10/BSR INCITS 513 Information technology - SCSI Primary Commands - 4 (SPC-4)

ANS Project INCITS 557 Information technology - SCSI / ATA Translation - 5 (SAT-5)

7.34.2 Description

This driver supports reporting the temperature of disk and solid state drives with temperature sensors.

If supported, it uses the ATA SCT Command Transport feature to read the current drive temperature and, if available, temperature limits as well as historic minimum and maximum temperatures. If SCT Command Transport is not supported, the driver uses SMART attributes to read the drive temperature.

7.34.3 Sysfs entries

Only the temp1_input attribute is always available. Other attributes are available only if reported by the drive. All temperatures are reported in milli-degrees Celsius.

temp1_input	Current drive temperature
temp1_lcrit	Minimum temperature limit. Operating the device below this temperature may cause physical damage to the device.
temp1_min	Minimum recommended continuous operating limit
temp1_max	Maximum recommended continuous operating temperature
temp1_crit	Maximum temperature limit. Operating the device above this temperature may cause physical damage to the device.
temp1_lowest	Minimum temperature seen this power cycle
temp1_highest	Maximum temperature seen this power cycle

7.35 Kernel driver ds1621

Supported chips:

- Dallas Semiconductor / Maxim Integrated DS1621

  Prefix: 'ds1621'

  Addresses scanned: none

  Datasheet: Publicly available from www.maximintegrated.com

- Dallas Semiconductor DS1625

  Prefix: 'ds1625'

  Addresses scanned: none

  Datasheet: Publicly available from www.datasheetarchive.com

- Maxim Integrated DS1631

  Prefix: 'ds1631'

  Addresses scanned: none

  Datasheet: Publicly available from www.maximintegrated.com

- Maxim Integrated DS1721

  Prefix: 'ds1721'

  Addresses scanned: none

  Datasheet: Publicly available from www.maximintegrated.com

- Maxim Integrated DS1731

  Prefix: 'ds1731'

  Addresses scanned: none

  Datasheet: Publicly available from www.maximintegrated.com

Authors:

- Christian W. Zuckschwerdt <zany@triq.net>
- valuable contributions by Jan M. Sendler <sendler@sendler.de>
- ported to 2.6 by Aurelien Jarno <aurelien@aurel32.net> with the help of Jean Delvare <jdelvare@suse.de>

7.35.1 Module Parameters

- polarity int Output' s polarity:
  - 0 = active high,
  - 1 = active low

7.35.2 Description

The DS1621 is a (one instance) digital thermometer and thermostat. It has both high and low temperature limits which can be user defined (i.e. programmed into non-volatile on-chip registers). Temperature range is -55 degree Celsius to +125 in 0.5 increments. You may convert this into a Fahrenheit range of -67 to +257 degrees with 0.9 steps. If polarity parameter is not provided, original value is used.

As for the thermostat, behavior can also be programmed using the polarity toggle. On the one hand ("heater"), the thermostat output of the chip, Tout, will trigger when the low limit temperature is met or underrun and stays high until the high limit is met or exceeded. On the other hand ("cooler"), vice versa. That way "heater" equals "active low" , whereas "conditioner" equals "active high" . Please note that the DS1621 data sheet is somewhat misleading in this point since setting the polarity bit does not simply invert Tout.

A second thing is that, during extensive testing, Tout showed a tolerance of up to +/- 0.5 degrees even when compared against precise temperature readings. Be sure to have a high vs. low temperature limit gap of al least 1.0 degree Celsius to avoid Tout "bouncing" , though!

The alarm bits are set when the high or low limits are met or exceeded and are reset by the module as soon as the respective temperature ranges are left.

The alarm registers are in no way suitable to find out about the actual status of Tout. They will only tell you about its history, whether or not any of the limits have ever been met or exceeded since last power-up or reset. Be aware: When testing, it showed that the status of Tout can change with neither of the alarms set.

Since there is no version or vendor identification register, there is no unique identification for these devices. Therefore, explicit device instantiation is required for correct device identification and functionality (one device per address in this address range: 0x48..0x4f).

The DS1625 is pin compatible and functionally equivalent with the DS1621, but the DS1621 is meant to replace it. The DS1631, DS1721, and DS1731 are also pin compatible with the DS1621 and provide multi-resolution support.

Additionally, the DS1721 data sheet says the temperature flags (THF and TLF) are used internally, however, these flags do get set and cleared as the actual temperature crosses the min or max settings (which by default are set to 75 and 80 degrees respectively).

7.35.3 Temperature Conversion

- DS1621 - 750ms (older devices may take up to 1000ms)
- DS1625 - 500ms
- DS1631 - 93ms..750ms for 9..12 bits resolution, respectively.
- DS1721 - 93ms..750ms for 9..12 bits resolution, respectively.
- DS1731 - 93ms..750ms for 9..12 bits resolution, respectively.

Note: On the DS1621, internal access to non-volatile registers may last for 10ms or less (unverified on the other devices).

7.35.4 Temperature Accuracy

- DS1621: +/- 0.5 degree Celsius (from 0 to +70 degrees)
- DS1625: +/- 0.5 degree Celsius (from 0 to +70 degrees)
- DS1631: +/- 0.5 degree Celsius (from 0 to +70 degrees)
- DS1721: +/- 1.0 degree Celsius (from -10 to +85 degrees)
- DS1731: +/- 1.0 degree Celsius (from -10 to +85 degrees)

Note: Please refer to the device datasheets for accuracy at other temperatures.

7.35.5 Temperature Resolution:

As mentioned above, the DS1631, DS1721, and DS1731 provide multi-resolution support, which is achieved via the R0 and R1 config register bits, where:

7.35.6 R0..R1

R0	R1	
0	0	9 bits, 0.5 degrees Celsius
1	0	10 bits, 0.25 degrees Celsius
0	1	11 bits, 0.125 degrees Celsius
1	1	12 bits, 0.0625 degrees Celsius

Note: At initial device power-on, the default resolution is set to 12-bits.

The resolution mode for the DS1631, DS1721, or DS1731 can be changed from userspace, via the device 'update_interval' sysfs attribute. This attribute will normalize the range of input values to the device maximum resolution values defined in the datasheet as follows:

Resolution	Conversion Time	Input Range
(C/LSB)	(msec)	(msec)
0.5	93.75	0⋯.94
0.25	187.5	95⋯187
0.125	375	188..375
0.0625	750	376..infinity

The following examples show how the 'update_interval' attribute can be used to change the conversion time:

```
$ cat update_interval
750
$ cat temp1_input
22062
$
$ echo 300 > update_interval
$ cat update_interval
375
$ cat temp1_input
22125
$
$ echo 150 > update_interval
$ cat update_interval
188
$ cat temp1_input
22250
$
$ echo 1 > update_interval
$ cat update_interval
94
$ cat temp1_input
22000
$
$ echo 1000 > update_interval
$ cat update_interval
750
$ cat temp1_input
22062
$
```

As shown, the ds1621 driver automatically adjusts the 'update_interval' user input, via a step function. Reading back the 'update_interval' value after a write operation provides the conversion time used by the device.

Mathematically, the resolution can be derived from the conversion time via the following function:

g(x) = 0.5 * [minimum_conversion_time/x]

where:

- ‘x’ = the output from ‘update_interval’
- ‘g(x)’ = the resolution in degrees C per LSB.
- 93.75ms = minimum conversion time

7.36 Kernel driver ds620

Supported chips:

- Dallas Semiconductor DS620

  Prefix: ‘ds620’

  Datasheet: Publicly available at the Dallas Semiconductor website

  http://www.dalsemi.com/

Authors: Roland Stigge <stigge@antcom.de> based on ds1621.c by Christian W. Zuckschwerdt <zany@triq.net>

7.36.1 Description

The DS620 is a (one instance) digital thermometer and thermostat. It has both high and low temperature limits which can be user defined (i.e. programmed into non-volatile on-chip registers). Temperature range is -55 degree Celsius to +125. Between 0 and 70 degree Celsius, accuracy is 0.5 Kelvin. The value returned via sysfs displays post decimal positions.

The thermostat function works as follows: When configured via platform_data (struct ds620_platform_data) .pomode == 0 (default), the thermostat output pin PO is always low. If .pomode == 1, the thermostat is in PO_LOW mode. I.e., the output pin PO becomes active when the temperature falls below temp1_min and stays active until the temperature goes above temp1_max.

Likewise, with .pomode == 2, the thermostat is in PO_HIGH mode. I.e., the PO output pin becomes active when the temperature goes above temp1_max and stays active until the temperature falls below temp1_min.

The PO output pin of the DS620 operates active-low.

7.37 Kernel driver emc1403

Supported chips:

- SMSC / Microchip EMC1402, EMC1412

  Addresses scanned: I2C 0x18, 0x1c, 0x29, 0x4c, 0x4d, 0x5c

  Prefix: ‘emc1402’

  Datasheets:

  - http://ww1.microchip.com/downloads/en/DeviceDoc/1412.pdf
  - http://ww1.microchip.com/downloads/en/DeviceDoc/1402.pdf

- SMSC / Microchip EMC1403, EMC1404, EMC1413, EMC1414

  Addresses scanned: I2C 0x18, 0x29, 0x4c, 0x4d

  Prefix: 'emc1403', 'emc1404'

  Datasheets:

  - http://ww1.microchip.com/downloads/en/DeviceDoc/1403_1404.pdf
  - http://ww1.microchip.com/downloads/en/DeviceDoc/1413_1414.pdf

- SMSC / Microchip EMC1422

  Addresses scanned: I2C 0x4c

  Prefix: 'emc1422'

  Datasheet:

  - http://ww1.microchip.com/downloads/en/DeviceDoc/1422.pdf

- SMSC / Microchip EMC1423, EMC1424

  Addresses scanned: I2C 0x4c

  Prefix: 'emc1423', 'emc1424'

  Datasheet:

  - http://ww1.microchip.com/downloads/en/DeviceDoc/1423_1424.pdf

Author: Kalhan Trisal <kalhan.trisal@intel.com

7.37.1 Description

The Standard Microsystems Corporation (SMSC) / Microchip EMC14xx chips contain up to four temperature sensors. EMC14x2 support two sensors (one internal, one external). EMC14x3 support three sensors (one internal, two external), and EMC14x4 support four sensors (one internal, three external).

The chips implement three limits for each sensor: low (tempX_min), high (tempX_max) and critical (tempX_crit.) The chips also implement an hysteresis mechanism which applies to all limits. The relative difference is stored in a single register on the chip, which means that the relative difference between the limit and its hysteresis is always the same for all three limits.

This implementation detail implies the following:

- When setting a limit, its hysteresis will automatically follow, the difference staying unchanged. For example, if the old critical limit was 80 degrees C, and the hysteresis was 75 degrees C, and you change the critical limit to 90 degrees C, then the hysteresis will automatically change to 85 degrees C.
- The hysteresis values can't be set independently. We decided to make only temp1_crit_hyst writable, while all other hysteresis attributes are read-only. Setting temp1_crit_hyst writes the difference between temp1_crit_hyst and temp1_crit into the chip, and the same relative hysteresis applies automatically to all other limits.
- The limits should be set before the hysteresis.

7.38 Kernel driver emc2103

Supported chips:

- SMSC EMC2103

  Addresses scanned: I2C 0x2e

  Prefix: 'emc2103'

  Datasheet: Not public

Authors: Steve Glendinning <steve.glendinning@smsc.com>

7.38.1 Description

The Standard Microsystems Corporation (SMSC) EMC2103 chips contain up to 4 temperature sensors and a single fan controller.

Fan rotation speeds are reported in RPM (rotations per minute). An alarm is triggered if the rotation speed has dropped below a programmable limit. Fan readings can be divided by a programmable divider (1, 2, 4 or 8) to give the readings more range or accuracy. Not all RPM values can accurately be represented, so some rounding is done. With a divider of 1, the lowest representable value is 480 RPM.

This driver supports RPM based control, to use this a fan target should be written to fan1_target and pwm1_enable should be set to 3.

The 2103-2 and 2103-4 variants have a third temperature sensor, which can be connected to two anti-parallel diodes. These values can be read as temp3 and temp4. If only one diode is attached to this channel, temp4 will show as "fault". The module parameter "apd=0" can be used to suppress this 4th channel when anti-parallel diodes are not fitted.

7.39 Kernel driver emc6w201

Supported chips:

- SMSC EMC6W201

  Prefix: 'emc6w201'

  Addresses scanned: I2C 0x2c, 0x2d, 0x2e

  Datasheet: Not public

Author: Jean Delvare <jdelvare@suse.de>

7.39.1 Description

From the datasheet:

“The EMC6W201 is an environmental monitoring device with automatic fan control capability and enhanced system acoustics for noise suppression. This ACPI compliant device provides hardware monitoring for up to six voltages (including its own VCC) and five external thermal sensors, measures the speed of up to five fans, and controls the speed of multiple DC fans using three Pulse Width Modulator (PWM) outputs. Note that it is possible to control more than three fans by connecting two fans to one PWM output. The EMC6W201 will be available in a 36-pin QFN package.”

The device is functionally close to the EMC6D100 series, but is register-incompatible.

The driver currently only supports the monitoring of the voltages, temperatures and fan speeds. Limits can be changed. Alarms are not supported, and neither is fan speed control.

7.39.2 Known Systems With EMC6W201

The EMC6W201 is a rare device, only found on a few systems, made in 2005 and 2006. Known systems with this device:

- Dell Precision 670 workstation
- Gigabyte 2CEWH mainboard

7.40 Kernel driver f71805f

Supported chips:

- Fintek F71805F/FG

  Prefix: ‘f71805f’

  Addresses scanned: none, address read from Super I/O config space

  Datasheet: Available from the Fintek website

- Fintek F71806F/FG

  Prefix: ‘f71872f’

  Addresses scanned: none, address read from Super I/O config space

  Datasheet: Available from the Fintek website

- Fintek F71872F/FG

  Prefix: ‘f71872f’

  Addresses scanned: none, address read from Super I/O config space

  Datasheet: Available from the Fintek website

Author: Jean Delvare <jdelvare@suse.de>

Thanks to Denis Kieft from Barracuda Networks for the donation of a test system (custom Jetway K8M8MS motherboard, with CPU and RAM) and for providing initial documentation.

Thanks to Kris Chen and Aaron Huang from Fintek for answering technical questions and providing additional documentation.

Thanks to Chris Lin from Jetway for providing wiring schematics and answering technical questions.

7.40.1 Description

The Fintek F71805F/FG Super I/O chip includes complete hardware monitoring capabilities. It can monitor up to 9 voltages (counting its own power source), 3 fans and 3 temperature sensors.

This chip also has fan controlling features, using either DC or PWM, in three different modes (one manual, two automatic).

The Fintek F71872F/FG Super I/O chip is almost the same, with two additional internal voltages monitored (VSB and battery). It also features 6 VID inputs. The VID inputs are not yet supported by this driver.

The Fintek F71806F/FG Super-I/O chip is essentially the same as the F71872F/FG, and is undistinguishable therefrom.

The driver assumes that no more than one chip is present, which seems reasonable.

7.40.2 Voltage Monitoring

Voltages are sampled by an 8-bit ADC with a LSB of 8 mV. The supported range is thus from 0 to 2.040 V. Voltage values outside of this range need external resistors. An exception is in0, which is used to monitor the chip's own power source (+3.3V), and is divided internally by a factor 2. For the F71872F/FG, in9 (VSB) and in10 (battery) are also divided internally by a factor 2.

The two LSB of the voltage limit registers are not used (always 0), so you can only set the limits in steps of 32 mV (before scaling).

The wirings and resistor values suggested by Fintek are as follow:

in	pin name	use	R1	R2	divider	expected raw val.
in0	VCC	VCC3.3V	int.	int.	2.00	1.65 V
in1	VIN1	VTT1.2V	10K	•	1.00	1.20 V
in2	VIN2	VRAM	100K	100K	2.00	~1.25 V[1]
in3	VIN3	VCHIPSET	47K	100K	1.47	2.24 V[2]
in4	VIN4	VCC5V	200K	47K	5.25	0.95 V
in5	VIN5	+12V	200K	20K	11.00	1.05 V
in6	VIN6	VCC1.5V	10K	•	1.00	1.50 V
in7	VIN7	VCORE	10K	•	1.00	~1.40 V[1]
in8	VIN8	VSB5V	200K	47K	1.00	0.95 V
in10	VSB	VSB3.3V	int.	int.	2.00	1.65 V[3]
in9	VBAT	VBATTERY	int.	int.	2.00	1.50 V[3]

These values can be used as hints at best, as motherboard manufacturers are free to use a completely different setup. As a matter of fact, the Jetway K8M8MS uses a significantly different setup. You will have to find out documentation about your own motherboard, and edit sensors.conf accordingly.

Each voltage measured has associated low and high limits, each of which triggers an alarm when crossed.

7.40.3 Fan Monitoring

Fan rotation speeds are reported as 12-bit values from a gated clock signal. Speeds down to 366 RPM can be measured. There is no theoretical high limit, but values over 6000 RPM seem to cause problem. The effective resolution is much lower than you would expect, the step between different register values being 10 rather than 1.

The chip assumes 2 pulse-per-revolution fans.

An alarm is triggered if the rotation speed drops below a programmable limit or is too low to be measured.

[1] Depends on your hardware setup.

[2] Obviously not correct, swapping R1 and R2 would make more sense.

[3] F71872F/FG only.

7.40.4 Temperature Monitoring

Temperatures are reported in degrees Celsius. Each temperature measured has a high limit, those crossing triggers an alarm. There is an associated hysteresis value, below which the temperature has to drop before the alarm is cleared.

All temperature channels are external, there is no embedded temperature sensor. Each channel can be used for connecting either a thermal diode or a thermistor. The driver reports the currently selected mode, but doesn't allow changing it. In theory, the BIOS should have configured everything properly.

7.40.5 Fan Control

Both PWM (pulse-width modulation) and DC fan speed control methods are supported. The right one to use depends on external circuitry on the motherboard, so the driver assumes that the BIOS set the method properly. The driver will report the method, but won't let you change it.

When the PWM method is used, you can select the operating frequency, from 187.5 kHz (default) to 31 Hz. The best frequency depends on the fan model. As a rule of thumb, lower frequencies seem to give better control, but may generate annoying high-pitch noise. So a frequency just above the audible range, such as 25 kHz, may be a good choice; if this doesn't give you good linear control, try reducing it. Fintek recommends not going below 1 kHz, as the fan tachometers get confused by lower frequencies as well.

When the DC method is used, Fintek recommends not going below 5 V, which corresponds to a pwm value of 106 for the driver. The driver doesn't enforce this limit though.

Three different fan control modes are supported; the mode number is written to the pwm<n>_enable file.

- 1: Manual mode You ask for a specific PWM duty cycle or DC voltage by writing to the pwm<n> file.
- 2: Temperature mode You define 3 temperature/fan speed trip points using the pwm<n>_auto_point<m>_temp and _fan files. These define a staircase relationship between temperature and fan speed with two additional points interpolated between the values that you define. When the temperature is below auto_point1_temp the fan is switched off.
- 3: Fan speed mode You ask for a specific fan speed by writing to the fan<n>_target file.

Both of the automatic modes require that pwm1 corresponds to fan1, pwm2 to fan2 and pwm3 to fan3. Temperature mode also requires that temp1 corresponds to pwm1 and fan1, etc.

7.41 Kernel driver f71882fg

Supported chips:

- Fintek F71808E

  Prefix: 'f71808e'

  Addresses scanned: none, address read from Super I/O config space

  Datasheet: Not public

- Fintek F71808A

  Prefix: 'f71808a'

  Addresses scanned: none, address read from Super I/O config space

  Datasheet: Not public

- Fintek F71858FG

  Prefix: 'f71858fg'

  Addresses scanned: none, address read from Super I/O config space

  Datasheet: Available from the Fintek website

- Fintek F71862FG and F71863FG

  Prefix: 'f71862fg'

  Addresses scanned: none, address read from Super I/O config space

  Datasheet: Available from the Fintek website

- Fintek F71869F and F71869E

  Prefix: 'f71869'

  Addresses scanned: none, address read from Super I/O config space

  Datasheet: Available from the Fintek website

- Fintek F71869A

  Prefix: 'f71869a'

  Addresses scanned: none, address read from Super I/O config space

  Datasheet: Not public

- Fintek F71882FG and F71883FG

  Prefix: 'f71882fg'

  Addresses scanned: none, address read from Super I/O config space

  Datasheet: Available from the Fintek website

- Fintek F71889FG

  Prefix: 'f71889fg'

  Addresses scanned: none, address read from Super I/O config space

  Datasheet: Available from the Fintek website

- Fintek F71889ED

  Prefix: ‘f71889ed’

  Addresses scanned: none, address read from Super I/O config space

  Datasheet: Should become available on the Fintek website soon

- Fintek F71889A

  Prefix: ‘f71889a’

  Addresses scanned: none, address read from Super I/O config space

  Datasheet: Should become available on the Fintek website soon

- Fintek F8000

  Prefix: ‘f8000’

  Addresses scanned: none, address read from Super I/O config space

  Datasheet: Not public

- Fintek F81801U

  Prefix: ‘f71889fg’

  Addresses scanned: none, address read from Super I/O config space

  Datasheet: Not public

  Note: This is the 64-pin variant of the F71889FG, they have the same device ID and are fully compatible as far as hardware monitoring is concerned.

- Fintek F81865F

  Prefix: ‘f81865f’

  Addresses scanned: none, address read from Super I/O config space

  Datasheet: Available from the Fintek website

Author: Hans de Goede <hdegoede@redhat.com>

7.41.1 Description

Fintek F718xx/F8000 Super I/O chips include complete hardware monitoring capabilities. They can monitor up to 9 voltages, 4 fans and 3 temperature sensors.

These chips also have fan controlling features, using either DC or PWM, in three different modes (one manual, two automatic).

The driver assumes that no more than one chip is present, which seems reasonable.

7.41.2 Monitoring

The Voltage, Fan and Temperature Monitoring uses the standard sysfs interface as documented in sysfs-interface, without any exceptions.

7.41.3 Fan Control

Both PWM (pulse-width modulation) and DC fan speed control methods are supported. The right one to use depends on external circuitry on the motherboard, so the driver assumes that the BIOS set the method properly.

Note that the lowest numbered temperature zone trip point corresponds to to the border between the highest and one but highest temperature zones, and vica versa. So the temperature zone trip points 1-4 (or 1-2) go from high temp to low temp! This is how things are implemented in the IC, and the driver mimics this.

There are 2 modes to specify the speed of the fan, PWM duty cycle (or DC voltage) mode, where 0-100% duty cycle (0-100% of 12V) is specified. And RPM mode where the actual RPM of the fan (as measured) is controlled and the speed gets specified as 0-100% of the fan#_full_speed file.

Since both modes work in a 0-100% (mapped to 0-255) scale, there isn't a whole lot of a difference when modifying fan control settings. The only important difference is that in RPM mode the 0-100% controls the fan speed between 0-100% of fan#_full_speed. It is assumed that if the BIOS programs RPM mode, it will also set fan#_full_speed properly, if it does not then fan control will not work properly, unless you set a sane fan#_full_speed value yourself.

Switching between these modes requires re-initializing a whole bunch of registers, so the mode which the BIOS has set is kept. The mode is printed when loading the driver.

Three different fan control modes are supported; the mode number is written to the pwm#_enable file. Note that not all modes are supported on all chips, and some modes may only be available in RPM / PWM mode. Writing an unsupported mode will result in an invalid parameter error.

- 1: Manual mode You ask for a specific PWM duty cycle / DC voltage or a specific % of fan#_full_speed by writing to the pwm# file. This mode is only available on the F71858FG / F8000 if the fan channel is in RPM mode.
- 2: Normal auto mode You can define a number of temperature/fan speed trip points, which % the fan should run at at this temp and which temp a fan should follow using the standard sysfs interface. The number and type of trip points is chip depended, see which files are available in sysfs. Fan/PWM channel 3 of the F8000 is always in this mode!
- 3: Thermostat mode (Only available on the F8000 when in duty cycle mode) The fan speed is regulated to keep the temp the fan is mapped to between temp#_auto_point2_temp and temp#_auto_point3_temp.

All of the automatic modes require that pwm1 corresponds to fan1, pwm2 to fan2 and pwm3 to fan3.

7.42 Kernel driver fam15h_power

Supported chips:

- AMD Family 15h Processors
- AMD Family 16h Processors

  Prefix: 'fam15h_power'

  Addresses scanned: PCI space

  Datasheets:

  - BIOS and Kernel Developer's Guide (BKDG) For AMD Family 15h Processors
  - BIOS and Kernel Developer's Guide (BKDG) For AMD Family 16h Processors
  - AMD64 Architecture Programmer's Manual Volume 2: System Programming

Author: Andreas Herrmann <herrmann.der.user@googlemail.com>

7.42.1 Description

1) Processor TDP (Thermal design power)

Given a fixed frequency and voltage, the power consumption of a processor varies based on the workload being executed. Derated power is the power consumed when running a specific application. Thermal design power (TDP) is an example of derated power.

This driver permits reading of registers providing power information of AMD Family 15h and 16h processors via TDP algorithm.

For AMD Family 15h and 16h processors the following power values can be calculated using different processor northbridge function registers:

- **BasePwrWatts:** Specifies in watts the maximum amount of power consumed by the processor for NB and logic external to the core.
- **ProcessorPwrWatts:** Specifies in watts the maximum amount of power the processor can support.
- **CurrPwrWatts:** Specifies in watts the current amount of power being consumed by the processor.

This driver provides ProcessorPwrWatts and CurrPwrWatts:

- power1_crit (ProcessorPwrWatts)
- power1_input (CurrPwrWatts)

On multi-node processors the calculated value is for the entire package and not for a single node. Thus the driver creates sysfs attributes only for internal node0 of a multi-node processor.

2) Accumulated Power Mechanism

This driver also introduces an algorithm that should be used to calculate the average power consumed by a processor during a measurement interval Tm. The feature of accumulated power mechanism is indicated by CPUID Fn8000_0007_EDX[12].

- **Tsample:** compute unit power accumulator sample period
- **Tref:** the PTSC counter period
- **PTSC:** performance timestamp counter
- **N:** the ratio of compute unit power accumulator sample period to the PTSC period
- **Jmax:** max compute unit accumulated power which is indicated by MaxCpuSwPwrAcc MSR C001007b
- **Jx/Jy:** compute unit accumulated power which is indicated by CpuSwPwrAcc MSR C001007a
- **Tx/Ty:** the value of performance timestamp counter which is indicated by CU_PTSC MSR C0010280
- **PwrCPUave:** CPU average power

i. Determine the ratio of Tsample to Tref by executing CPUID Fn8000_0007.

   N = value of CPUID Fn8000_0007_ECX[CpuPwrSampleTimeRatio[15:0]].

ii. Read the full range of the cumulative energy value from the new MSR MaxCpuSwPwrAcc.

   Jmax = value returned.

iii. At time x, SW reads CpuSwPwrAcc MSR and samples the PTSC.

   Jx = value read from CpuSwPwrAcc and Tx = value read from PTSC.

iv. At time y, SW reads CpuSwPwrAcc MSR and samples the PTSC.

   Jy = value read from CpuSwPwrAcc and Ty = value read from PTSC.

v. Calculate the average power consumption for a compute unit over time period (y-x). Unit of result is uWatt:

```
if (Jy < Jx) // Rollover has occurred
        Jdelta = (Jy + Jmax) - Jx
else
        Jdelta = Jy - Jx
PwrCPUave = N * Jdelta * 1000 / (Ty - Tx)
```

This driver provides PwrCPUave and interval(default is 10 millisecond and maximum is 1 second):

- power1_average (PwrCPUave)
- power1_average_interval (Interval)

The power1_average_interval can be updated at /etc/sensors3.conf file as below:

chip fam15h_power-* set power1_average_interval 0.01

Then save it with “sensors -s”.

7.43 Kernel driver ftsteutates

Supported chips:

- FTS Teutates

  Prefix: 'ftsteutates'

  Addresses scanned: I2C 0x73 (7-Bit)

Author: Thilo Cestonaro <thilo.cestonaro@ts.fujitsu.com>

7.43.1 Description

The BMC Teutates is the Eleventh generation of Superior System monitoring and thermal management solution. It is builds on the basic functionality of the BMC Theseus and contains several new features and enhancements. It can monitor up to 4 voltages, 16 temperatures and 8 fans. It also contains an integrated watchdog which is currently implemented in this driver.

To clear a temperature or fan alarm, execute the following command with the correct path to the alarm file:

```
echo 0 >XXXX_alarm
```

Specification of the chip can be found here:

- ftp://ftp.ts.fujitsu.com/pub/Mainboard-OEM-Sales/Services/Software&Tools/Linux_SystemMonitoring&Watchdog&GPIO/BMC-Teutates_Specification_V1.21.pdf
- ftp://ftp.ts.fujitsu.com/pub/Mainboard-OEM-Sales/Services/Software&Tools/Linux_SystemMonitoring&Watchdog&GPIO/Fujitsu_mainboards-1-Sensors_HowTo-en-US.pdf

7.44 Kernel driver g760a

Supported chips:

- Global Mixed-mode Technology Inc. G760A

  Prefix: 'g760a'

  Datasheet: Publicly available at the GMT website

  http://www.gmt.com.tw/product/datasheet/EDS-760A.pdf

Author: Herbert Valerio Riedel <hvr@gnu.org>

7.44.1 Description

The GMT G760A Fan Speed PWM Controller is connected directly to a fan and performs closed-loop control of the fan speed.

The fan speed is programmed by setting the period via ‘pwm1’ of two consecutive speed pulses. The period is defined in terms of clock cycle counts of an assumed 32kHz clock source.

Setting a period of 0 stops the fan; setting the period to 255 sets fan to maximum speed.

The measured fan rotation speed returned via ‘fan1_input’ is derived from the measured speed pulse period by assuming again a 32kHz clock source and a 2 pulse-per-revolution fan.

The ‘alarms’ file provides access to the two alarm bits provided by the G760A chip’s status register: Bit 0 is set when the actual fan speed differs more than 20% with respect to the programmed fan speed; bit 1 is set when fan speed is below 1920 RPM.

The g760a driver will not update its values more frequently than every other second; reading them more often will do no harm, but will return ‘old’ values.

7.45 Kernel driver g762

The GMT G762 Fan Speed PWM Controller is connected directly to a fan and performs closed-loop or open-loop control of the fan speed. Two modes - PWM or DC - are supported by the device.

For additional information, a detailed datasheet is available at http://natisbad.org/NAS/ref/GMT_EDS-762_763-080710-0.2.pdf. sysfs bindings are described in Documentation/hwmon/sysfs-interface.rst.

The following entries are available to the user in a subdirectory of /sys/bus/i2c/drivers/g762/ to control the operation of the device. This can be done manually using the following entries but is usually done via a userland daemon like fancontrol.

Note that those entries do not provide ways to setup the specific hardware characteristics of the system (reference clock, pulses per fan revolution, ⋯); Those can be modified via devicetree bindings documented in Documentation/devicetree/bindings/hwmon/g762.txt or using a specific platform_data structure in board initialization file (see include/linux/platform_data/g762.h).

fan1_target: set desired fan speed. This only makes sense in closed-loop fan speed control (i.e. when pwm1_enable is set to 2).

fan1_input: provide current fan rotation value in RPM as reported by the fan to the device.

fan1_div: fan clock divisor. Supported value are 1, 2, 4 and 8.

fan1_pulses: number of pulses per fan revolution. Supported values are 2 and 4.

fan1_fault: reports fan failure, i.e. no transition on fan gear pin for about 0.7s (if the fan is not voluntarily set off).

fan1_alarm: in closed-loop control mode, if fan RPM value is 25% out of the programmed value for over 6 seconds ‘fan1_alarm’ is set to 1.

pwm1_enable: set current fan speed control mode i.e. 1 for manual fan speed control (open-loop) via pwm1 described below, 2 for automatic fan speed control (closed-loop) via fan1_target above.

pwm1_mode: set or get fan driving mode: 1 for PWM mode, 0 for DC mode.

pwm1: get or set PWM fan control value in open-loop mode. This is an integer value between 0 and 255. 0 stops the fan, 255 makes it run at full speed.

Both in PWM mode (‘pwm1_mode’ set to 1) and DC mode (‘pwm1_mode’ set to 0), when current fan speed control mode is open-loop (‘pwm1_enable’ set to 1), the fan speed is programmed by setting a value between 0 and 255 via ‘pwm1’ entry (0 stops the fan, 255 makes it run at full speed). In closed-loop mode (‘pwm1_enable’ set to 2), the expected rotation speed in RPM can be passed to the chip via ‘fan1_target’ . In closed-loop mode, the target speed is compared with current speed (available via ‘fan1_input’) by the device and a feedback is performed to match that target value. The fan speed value is computed based on the parameters associated with the physical characteristics of the system: a reference clock source frequency, a number of pulses per fan revolution, etc.

Note that the driver will update its values at most once per second.

7.46 Kernel driver gsc-hwmon

Supported chips: Gateworks GSC Datasheet: http://trac.gateworks.com/wiki/gsc
Author: Tim Harvey <tharvey@gateworks.com>

7.46.1 Description:

This driver supports hardware monitoring for the temperature sensor, various ADC’ s connected to the GSC, and optional FAN controller available on some boards.

7.46.2 Voltage Monitoring

The voltage inputs are scaled either internally or by the driver depending on the GSC version and firmware. The values returned by the driver do not need further scaling. The voltage input labels provide the voltage rail name:

inX_input Measured voltage (mV). inX_label Name of voltage rail.

7.46.3 Temperature Monitoring

Temperatures are measured with 12-bit or 10-bit resolution and are scaled either internally or by the driver depending on the GSC version and firmware. The values returned by the driver reflect millidegree Celcius:

tempX_input Measured temperature. tempX_label Name of temperature input.

7.46.4 PWM Output Control

The GSC features 1 PWM output that operates in automatic mode where the PWM value will be scalled depending on 6 temperature boundaries. The tempeature boundaries are read-write and in millidegree Celcius and the read-only PWM values range from 0 (off) to 255 (full speed). Fan speed will be set to minimum (off) when the temperature sensor reads less than pwm1_auto_point1_temp and maximum when the temperature sensor equals or exceeds pwm1_auto_point6_temp.

pwm1_auto_point[1-6]_pwm PWM value. pwm1_auto_point[1-6]_temp Temperature boundary.

7.47 Kernel driver gl518sm

Supported chips:

- Genesys Logic GL518SM release 0x00

  Prefix: 'gl518sm'

  Addresses scanned: I2C 0x2c and 0x2d

- Genesys Logic GL518SM release 0x80

  Prefix: 'gl518sm'

  Addresses scanned: I2C 0x2c and 0x2d

  Datasheet: http://www.genesyslogic.com/

Authors:

- Frodo Looijaard <frodol@dds.nl>,
- Kyösti Mälkki <kmalkki@cc.hut.fi>
- Hong-Gunn Chew <hglinux@gunnet.org>
- Jean Delvare <jdelvare@suse.de>

7.47.1 Description

Important: For the revision 0x00 chip, the in0, in1, and in2 values (+5V, +3V, and +12V) CANNOT be read. This is a limitation of the chip, not the driver.

This driver supports the Genesys Logic GL518SM chip. There are at least two revision of this chip, which we call revision 0x00 and 0x80. Revision 0x80 chips support the reading of all voltages and revision 0x00 only for VIN3.

The GL518SM implements one temperature sensor, two fan rotation speed sensors, and four voltage sensors. It can report alarms through the computer speakers.

Temperatures are measured in degrees Celsius. An alarm goes off while the temperature is above the over temperature limit, and has not yet dropped below the hysteresis limit. The alarm always reflects the current situation. Measurements are guaranteed between -10 degrees and +110 degrees, with a accuracy of +/-3 degrees.

Rotation speeds are reported in RPM (rotations per minute). An alarm is triggered if the rotation speed has dropped below a programmable limit. In case when you have selected to turn fan1 off, no fan1 alarm is triggered.

Fan readings can be divided by a programmable divider (1, 2, 4 or 8) to give the readings more range or accuracy. Not all RPM values can accurately be represented, so some rounding is done. With a divider of 2, the lowest representable value is around 1900 RPM.

Voltage sensors (also known as VIN sensors) report their values in volts. An alarm is triggered if the voltage has crossed a programmable minimum or maximum limit. Note that minimum in this case always means ‘closest to zero’; this is important for negative voltage measurements. The VDD input measures voltages between 0.000 and 5.865 volt, with a resolution of 0.023 volt. The other inputs measure voltages between 0.000 and 4.845 volt, with a resolution of 0.019 volt. Note that revision 0x00 chips do not support reading the current voltage of any input except for VIN3; limit setting and alarms work fine, though.

When an alarm is triggered, you can be warned by a beeping signal through your computer speaker. It is possible to enable all beeping globally, or only the beeping for some alarms.

If an alarm triggers, it will remain triggered until the hardware register is read at least once (except for temperature alarms). This means that the cause for the alarm may already have disappeared! Note that in the current implementation, all hardware registers are read whenever any data is read (unless it is less than 1.5 seconds since the last update). This means that you can easily miss once-only alarms.

The GL518SM only updates its values each 1.5 seconds; reading it more often will do no harm, but will return ‘old’ values.

7.48 Kernel driver hih6130

Supported chips:

- Honeywell HIH-6130 / HIH-6131

  Prefix: 'hih6130'

  Addresses scanned: none

  Datasheet: Publicly available at the Honeywell website

  http://sensing.honeywell.com/index.php?ci_id=3106&la_id=1&defId=44872

Author: Iain Paton <ipaton0@gmail.com>

7.48.1 Description

The HIH-6130 & HIH-6131 are humidity and temperature sensors in a SO8 package. The difference between the two devices is that the HIH-6131 has a condensation filter.

The devices communicate with the I2C protocol. All sensors are set to the same I2C address 0x27 by default, so an entry with I2C_BOARD_INFO("hih6130", 0x27) can be used in the board setup code.

Please see Documentation/i2c/instantiating-devices.rst for details on how to instantiate I2C devices.

7.48.2 sysfs-Interface

temp1_input temperature input

humidity1_input humidity input

7.48.3 Notes

Command mode and alarms are not currently supported.

7.49 Kernel driver ibmaem

This driver talks to the IBM Systems Director Active Energy Manager, known henceforth as AEM.

Supported systems:

- Any recent IBM System X server with AEM support.

  This includes the x3350, x3550, x3650, x3655, x3755, x3850 M2, x3950 M2, and certain HC10/HS2x/LS2x/QS2x blades.

  The IPMI host interface driver ("ipmi-si") needs to be loaded for this driver to do anything.

Prefix: 'ibmaem'

Datasheet: Not available

Author: Darrick J. Wong

7.49.1 Description

This driver implements sensor reading support for the energy and power meters available on various IBM System X hardware through the BMC. All sensor banks will be exported as platform devices; this driver can talk to both v1 and v2 interfaces. This driver is completely separate from the older ibmpex driver.

The v1 AEM interface has a simple set of features to monitor energy use. There is a register that displays an estimate of raw energy consumption since the last BMC reset, and a power sensor that returns average power use over a configurable interval.

The v2 AEM interface is a bit more sophisticated, being able to present a wider range of energy and power use registers, the power cap as set by the AEM software, and temperature sensors.

7.49.2 Special Features

The "power_cap" value displays the current system power cap, as set by the AEM software. Setting the power cap from the host is not currently supported.

7.50 Kernel driver ibm-cffps

Supported chips:

- IBM Common Form Factor power supply

Author: Eddie James <eajames@us.ibm.com>

7.50.1 Description

This driver supports IBM Common Form Factor (CFF) power supplies. This driver is a client to the core PMBus driver.

7.50.2 Usage Notes

This driver does not auto-detect devices. You will have to instantiate the devices explicitly. Please see Documentation/i2c/instantiating-devices.rst for details.

7.50.3 Sysfs entries

The following attributes are supported:

curr1_alarm	Output current over-current alarm.
curr1_input	Measured output current in mA.
curr1_label	“iout1”
fan1_alarm	Fan 1 warning.
fan1_fault	Fan 1 fault.
fan1_input	Fan 1 speed in RPM.
fan2_alarm	Fan 2 warning.
fan2_fault	Fan 2 fault.
fan2_input	Fan 2 speed in RPM.
in1_alarm	Input voltage under-voltage alarm.
in1_input	Measured input voltage in mV.
in1_label	“vin”
in2_alarm	Output voltage over-voltage alarm.
in2_input	Measured output voltage in mV.
in2_label	“vout1”
power1_alarm	Input fault or alarm.
power1_input	Measured input power in uW.
power1_label	“pin”
temp1_alarm	PSU inlet ambient temperature over-temperature alarm.
temp1_input	Measured PSU inlet ambient temp in millidegrees C.
temp2_alarm	Secondary rectifier temp over-temperature alarm.
temp2_input	Measured secondary rectifier temp in millidegrees C.
temp3_alarm	ORing FET temperature over-temperature alarm.
temp3_input	Measured ORing FET temperature in millidegrees C.

7.51 Kernel Driver IBMPOWERNV

Supported systems:

- Any recent IBM P servers based on POWERNV platform

Author: Neelesh Gupta

7.51.1 Description

This driver implements reading the platform sensors data like temperature/fan/voltage/power for ‘POWERNV’ platform.

The driver uses the platform device infrastructure. It probes the device tree for sensor devices during the __init phase and registers them with the ‘hwmon’. ‘hwmon’ populates the ‘sysfs’ tree having attribute files, each for a given sensor type and its attribute data.

All the nodes in the DT appear under “/ibm,opal/sensors” and each valid node in the DT maps to an attribute file in ‘sysfs’. The node exports unique ‘sensor-id’ which the driver uses to make an OPAL call to the firmware.

7.51.2 Usage notes

The driver is built statically with the kernel by enabling the config CONFIG_SENSORS_IBMPOWERNV. It can also be built as module ‘ibmpowernv’.

7.51.3 Sysfs attributes

fanX_input	Measured RPM value.
fanX_min	Threshold RPM for alert generation.
fanX_fault	• 0: No fail condition • 1: Failing fan
tempX_input	Measured ambient temperature.
tempX_max	Threshold ambient temperature for alert generation.
tempX_highest	Historical maximum temperature
tempX_lowest	Historical minimum temperature
tempX_enable	Enable/disable all temperature sensors belonging to the sub-group. In POWER9, this attribute corresponds to each OCC. Using this attribute each OCC can be asked to disable/enable all of its temperature sensors. • 1: Enable • 0: Disable
inX_input	Measured power supply voltage (millivolt)
inX_fault	• 0: No fail condition. • 1: Failing power supply.
inX_highest	Historical maximum voltage
inX_lowest	Historical minimum voltage
inX_enable	Enable/disable all voltage sensors belonging to the sub-group. In POWER9, this attribute corresponds to each OCC. Using this attribute each OCC can be asked to disable/enable all of its voltage sensors. • 1: Enable • 0: Disable
powerX_input	Power consumption (microWatt)
powerX_input_highest	Historical maximum power
powerX_input_lowest	Historical minimum power
powerX_enable	Enable/disable all power sensors belonging to the sub-group. In POWER9, this attribute corresponds to each OCC. Using this attribute each OCC can be asked to disable/enable all of its power sensors. • 1: Enable • 0: Disable
currX_input	Measured current (milliampere)
currX_highest	Historical maximum current
currX_lowest	Historical minimum current
currX_enable	Enable/disable all current sensors belonging to the sub-group. In POWER9,

7.52 Kernel driver ina209

Supported chips:

- Burr-Brown / Texas Instruments INA209

  Prefix: 'ina209'

  Addresses scanned: -

  Datasheet: http://www.ti.com/lit/gpn/ina209

Author:

- Paul Hays <Paul.Hays@cattail.ca>
- Ira W. Snyder <iws@ovro.caltech.edu>
- Guenter Roeck <linux@roeck-us.net>

7.52.1 Description

The TI / Burr-Brown INA209 monitors voltage, current, and power on the high side of a D.C. power supply. It can perform measurements and calculations in the background to supply readings at any time. It includes a programmable calibration multiplier to scale the displayed current and power values.

7.52.2 Sysfs entries

The INA209 chip is highly configurable both via hardwiring and via the I2C bus. See the datasheet for details.

This tries to expose most monitoring features of the hardware via sysfs. It does not support every feature of this chip.

in0_input	shunt voltage (mV)
in0_input_highest	shunt voltage historical maximum reading (mV)
in0_input_lowest	shunt voltage historical minimum reading (mV)
in0_reset_history	reset shunt voltage history
in0_max	shunt voltage max alarm limit (mV)
in0_min	shunt voltage min alarm limit (mV)
in0_crit_max	shunt voltage crit max alarm limit (mV)
in0_crit_min	shunt voltage crit min alarm limit (mV)
in0_max_alarm	shunt voltage max alarm limit exceeded
in0_min_alarm	shunt voltage min alarm limit exceeded
in0_crit_max_alarm	shunt voltage crit max alarm limit exceeded
in0_crit_min_alarm	shunt voltage crit min alarm limit exceeded
in1_input	bus voltage (mV)
in1_input_highest	bus voltage historical maximum reading (mV)
in1_input_lowest	bus voltage historical minimum reading (mV)
in1_reset_history	reset bus voltage history
in1_max	bus voltage max alarm limit (mV)

Continued on next page

Table 2 – continued from previous page

in1_min	bus voltage min alarm limit (mV)
in1_crit_max	bus voltage crit max alarm limit (mV)
in1_crit_min	bus voltage crit min alarm limit (mV)
in1_max_alarm	bus voltage max alarm limit exceeded
in1_min_alarm	bus voltage min alarm limit exceeded
in1_crit_max_alarm	bus voltage crit max alarm limit exceeded
in1_crit_min_alarm	bus voltage crit min alarm limit exceeded
power1_input	power measurement (uW)
power1_input_highest	power historical maximum reading (uW)
power1_reset_history	reset power history
power1_max	power max alarm limit (uW)
power1_crit	power crit alarm limit (uW)
power1_max_alarm	power max alarm limit exceeded
power1_crit_alarm	power crit alarm limit exceeded
curr1_input	current measurement (mA)
update_interval	data conversion time; affects number of samples used to average res

7.52.3 General Remarks

The power and current registers in this chip require that the calibration register is programmed correctly before they are used. Normally this is expected to be done in the BIOS. In the absence of BIOS programming, the shunt resistor voltage can be provided using platform data. The driver uses platform data from the ina2xx driver for this purpose. If calibration register data is not provided via platform data, the driver checks if the calibration register has been programmed (ie has a value not equal to zero). If so, this value is retained. Otherwise, a default value reflecting a shunt resistor value of 10 mOhm is programmed into the calibration register.

7.52.4 Output Pins

Output pin programming is a board feature which depends on the BIOS. It is outside the scope of a hardware monitoring driver to enable or disable output pins.

7.53 Kernel driver ina2xx

Supported chips:

- Texas Instruments INA219

  Prefix: 'ina219' Addresses: I2C 0x40 - 0x4f

  Datasheet: Publicly available at the Texas Instruments website

  http://www.ti.com/

- Texas Instruments INA220

  Prefix: 'ina220'

Addresses: I2C 0x40 - 0x4f

Datasheet: Publicly available at the Texas Instruments website

http://www.ti.com/

- Texas Instruments INA226

  Prefix: 'ina226'

  Addresses: I2C 0x40 - 0x4f

  Datasheet: Publicly available at the Texas Instruments website

  http://www.ti.com/

- Texas Instruments INA230

  Prefix: 'ina230'

  Addresses: I2C 0x40 - 0x4f

  Datasheet: Publicly available at the Texas Instruments website

  http://www.ti.com/

- Texas Instruments INA231

  Prefix: 'ina231'

  Addresses: I2C 0x40 - 0x4f

  Datasheet: Publicly available at the Texas Instruments website

  http://www.ti.com/

Author: Lothar Felten <lothar.felten@gmail.com>

7.53.1 Description

The INA219 is a high-side current shunt and power monitor with an I2C interface. The INA219 monitors both shunt drop and supply voltage, with programmable conversion times and filtering.

The INA220 is a high or low side current shunt and power monitor with an I2C interface. The INA220 monitors both shunt drop and supply voltage.

The INA226 is a current shunt and power monitor with an I2C interface. The INA226 monitors both a shunt voltage drop and bus supply voltage.

INA230 and INA231 are high or low side current shunt and power monitors with an I2C interface. The chips monitor both a shunt voltage drop and bus supply voltage.

The shunt value in micro-ohms can be set via platform data or device tree at compile-time or via the shunt_resistor attribute in sysfs at run-time. Please refer to the Documentation/devicetree/bindings/hwmon/ina2xx.txt for bindings if the device tree is used.

Additionally ina226 supports update_interval attribute as described in Documentation/hwmon/sysfs-interface.rst. Internally the interval is the sum of bus and shunt voltage conversion times multiplied by the averaging rate. We

don’t touch the conversion times and only modify the number of averages. The lower limit of the update_interval is 2 ms, the upper limit is 2253 ms. The actual programmed interval may vary from the desired value.

7.53.2 General sysfs entries

in0_input	Shunt voltage(mV) channel
in1_input	Bus voltage(mV) channel
curr1_input	Current(mA) measurement channel
power1_input	Power(uW) measurement channel
shunt_resistor	Shunt resistance(uOhm) channel

7.53.3 Sysfs entries for ina226, ina230 and ina231 only

in0_lcrit	Critical low shunt voltage
in0_crit	Critical high shunt voltage
in0_lcrit_alarm	Shunt voltage critical low alarm
in0_crit_alarm	Shunt voltage critical high alarm
in1_lcrit	Critical low bus voltage
in1_crit	Critical high bus voltage
in1_lcrit_alarm	Bus voltage critical low alarm
in1_crit_alarm	Bus voltage critical high alarm
power1_crit	Critical high power
power1_crit_alarm	Power critical high alarm
update_interval	data conversion time; affects number of samples used to average results for shunt and bus voltages.

Note:

- Configure shunt_resistor before configure power1_crit, because power value is calculated based on shunt_resistor set.
- Because of the underlying register implementation, only one *crit setting and its alarm can be active. Writing to one *crit setting clears other *crit settings and alarms. Writing 0 to any *crit setting clears all *crit settings and alarms.

7.54 Kernel driver ina3221

Supported chips:

- Texas Instruments INA3221

  Prefix: ‘ina3221’

  Addresses: I2C 0x40 - 0x43

  Datasheet: Publicly available at the Texas Instruments website

http://www.ti.com/

Author: Andrew F. Davis <afd@ti.com>

7.54.1 Description

The Texas Instruments INA3221 monitors voltage, current, and power on the high side of up to three D.C. power supplies. The INA3221 monitors both shunt drop and supply voltage, with programmable conversion times and averaging, current and power are calculated host-side from these.

7.54.2 Sysfs entries

in[123]_label	Voltage channel labels
in[123]_enable	Voltage channel enable controls
in[123]_input	Bus voltage(mV) channels
curr[123]_input	Current(mA) measurement channels
shunt[123]_resistor	Shunt resistance(uOhm) channels
curr[123]_crit	Critical alert current(mA) setting, activates the corresponding alarm when the respective current is above this value
curr[123]_crit_alarm	Critical alert current limit exceeded
curr[123]_max	Warning alert current(mA) setting, activates the corresponding alarm when the respective current average is above this value.
curr[123]_max_alarm	Warning alert current limit exceeded
in[456]_input	Shunt voltage(uV) for channels 1, 2, and 3 respectively
in7_input	Sum of shunt voltage(uV) channels
in7_label	Channel label for sum of shunt voltage
curr4_input	Sum of current(mA) measurement channels, (only available when all channels use the same resistor value for their shunt resistors)
curr4_crit	Critical alert current(mA) setting for sum of current measurements, activates the corresponding alarm when the respective current is above this value (only effective when all channels use the same resistor value for their shunt resistors)
curr4_crit_alarm	Critical alert current limit exceeded for sum of current measurements.
samples	Number of samples using in the averaging mode. Supports the list of number of samples: 1, 4, 16, 64, 128, 256, 512, 1024
update_interval	Data conversion time in millisecond, following: update_interval = C x S x (BC + SC) • C: number of enabled channels • S: number of samples • BC: bus-voltage conversion time in millisecond • SC: shunt-voltage conversion time in millisecond Affects both Bus- and Shunt-voltage conversion time. Note that setting update_interval to 0ms sets both BC and SC to 140 us (minimum conversion

7.55 Kernel driver ir35221

Supported chips:

- Infineon IR35221

  Prefix: 'ir35221'

  Addresses scanned: -

  Datasheet: Datasheet is not publicly available.

Author: Samuel Mendoza-Jonas <sam@mendozajonas.com>

7.55.1 Description

IR35221 is a Digital DC-DC Multiphase Converter

7.55.2 Usage Notes

This driver does not probe for PMBus devices. You will have to instantiate devices explicitly.

Example: the following commands will load the driver for an IR35221 at address 0x70 on I2C bus #4:

```
# modprobe ir35221
# echo ir35221 0x70 > /sys/bus/i2c/devices/i2c-4/new_device
```

7.55.3 Sysfs attributes

curr1_label	"iin"
curr1_input	Measured input current
curr1_max	Maximum current
curr1_max_alarm	Current high alarm
curr[2-3]_label	"iout[1-2]"
curr[2-3]_input	Measured output current
curr[2-3]_crit	Critical maximum current
curr[2-3]_crit_alarm	Current critical high alarm
curr[2-3]_highest	Highest output current
curr[2-3]_lowest	Lowest output current
curr[2-3]_max	Maximum current
curr[2-3]_max_alarm	Current high alarm
in1_label	"vin"
in1_input	Measured input voltage
in1_crit	Critical maximum input voltage
in1_crit_alarm	Input voltage critical high alarm
in1_highest	Highest input voltage
in1_lowest	Lowest input voltage

Continued on next page

Table 3 – continued from previous page

in1_min	Minimum input voltage
in1_min_alarm	Input voltage low alarm
in[2-3]_label	“vout[1-2]”
in[2-3]_input	Measured output voltage
in[2-3]_lcrit	Critical minimum output voltage
in[2-3]_lcrit_alarm	Output voltage critical low alarm
in[2-3]_crit	Critical maximum output voltage
in[2-3]_crit_alarm	Output voltage critical high alarm
in[2-3]_highest	Highest output voltage
in[2-3]_lowest	Lowest output voltage
in[2-3]_max	Maximum output voltage
in[2-3]_max_alarm	Output voltage high alarm
in[2-3]_min	Minimum output voltage
in[2-3]_min_alarm	Output voltage low alarm
power1_label	“pin”
power1_input	Measured input power
power1_alarm	Input power high alarm
power1_max	Input power limit
power[2-3]_label	“pout[1-2]”
power[2-3]_input	Measured output power
power[2-3]_max	Output power limit
power[2-3]_max_alarm	Output power high alarm
temp[1-2]_input	Measured temperature
temp[1-2]_crit	Critical high temperature
temp[1-2]_crit_alarm	Chip temperature critical high alarm
temp[1-2]_highest	Highest temperature
temp[1-2]_lowest	Lowest temperature
temp[1-2]_max	Maximum temperature
temp[1-2]_max_alarm	Chip temperature high alarm

7.56 Kernel driver ir38064

Supported chips:

- Infineon IR38064

  Prefix: ‘ir38064’ Addresses scanned: -

  Datasheet: Publicly available at the Infineon webiste https://www.infineon.com/dgdl/Infineon-IR38064MTRPBF-DS-v03_07-EN.pdf?fileId=5546d462584d1d4a0158db0d9efb67ca

Authors:

- Maxim Sloyko <maxims@google.com>
- Patrick Venture <venture@google.com>

7.56.1 Description

IR38064 is a Single-input Voltage, Synchronous Buck Regulator, DC-DC Converter.

7.56.2 Usage Notes

This driver does not probe for PMBus devices. You will have to instantiate devices explicitly.

7.56.3 Sysfs attributes

curr1_label	“iout1”
curr1_input	Measured output current
curr1_crit	Critical maximum current
curr1_crit_alarm	Current critical high alarm
curr1_max	Maximum current
curr1_max_alarm	Current high alarm
in1_label	“vin”
in1_input	Measured input voltage
in1_crit	Critical maximum input voltage
in1_crit_alarm	Input voltage critical high alarm
in1_min	Minimum input voltage
in1_min_alarm	Input voltage low alarm
in2_label	“vout1”
in2_input	Measured output voltage
in2_lcrit	Critical minimum output voltage
in2_lcrit_alarm	Output voltage critical low alarm
in2_crit	Critical maximum output voltage
in2_crit_alarm	Output voltage critical high alarm
in2_max	Maximum output voltage
in2_max_alarm	Output voltage high alarm
in2_min	Minimum output voltage
in2_min_alarm	Output voltage low alarm
power1_label	“pout1”
power1_input	Measured output power
temp1_input	Measured temperature
temp1_crit	Critical high temperature
temp1_crit_alarm	Chip temperature critical high alarm
temp1_max	Maximum temperature
temp1_max_alarm	Chip temperature high alarm

7.57 Kernel driver isl68137

Supported chips:

- Renesas ISL68137

  Prefix: ‘isl68137’

  Addresses scanned: -

  Datasheet:

  Publicly available at the Renesas website https://www.renesas.com/us/en/www/doc/datasheet/isl68137.pdf

- Renesas ISL68220

  Prefix: ‘isl68220’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL68221

  Prefix: ‘isl68221’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL68222

  Prefix: ‘isl68222’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL68223

  Prefix: ‘isl68223’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL68224

  Prefix: ‘isl68224’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL68225

  Prefix: ‘isl68225’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL68226

  Prefix: ‘isl68226’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL68227

  Prefix: ‘isl68227’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL68229

  Prefix: ‘isl68229’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL68233

  Prefix: ‘isl68233’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL68239

  Prefix: ‘isl68239’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69222

  Prefix: ‘isl69222’

  Addresses scanned: -

  Datasheet:

Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69223

  Prefix: ‘isl69223’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69224

  Prefix: ‘isl69224’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69225

  Prefix: ‘isl69225’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69227

  Prefix: ‘isl69227’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69228

  Prefix: ‘isl69228’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69234

  Prefix: ‘isl69234’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69236

  Prefix: ‘isl69236’

  Addresses scanned: -

Datasheet:

Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69239

  Prefix: ‘isl69239’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69242

  Prefix: ‘isl69242’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69243

  Prefix: ‘isl69243’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69247

  Prefix: ‘isl69247’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69248

  Prefix: ‘isl69248’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69254

  Prefix: ‘isl69254’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69255

  Prefix: ‘isl69255’

Addresses scanned: -

Datasheet:

Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69256

  Prefix: ‘isl69256’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69259

  Prefix: ‘isl69259’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69260

  Prefix: ‘isl69260’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69268

  Prefix: ‘isl69268’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69269

  Prefix: ‘isl69269’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas ISL69298

  Prefix: ‘isl69298’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas RAA228000

  Prefix: ‘raa228000’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas RAA228004

  Prefix: ‘raa228004’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas RAA228006

  Prefix: ‘raa228006’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas RAA228228

  Prefix: ‘raa228228’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas RAA229001

  Prefix: ‘raa229001’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

- Renesas RAA229004

  Prefix: ‘raa229004’

  Addresses scanned: -

  Datasheet:

  Publicly available (after August 2020 launch) at the Renesas website

Authors:

- Maxim Sloyko <maxims@google.com>
- Robert Lippert <rlippert@google.com>
- Patrick Venture <venture@google.com>

- Grant Peltier <grant.peltier.jg@renesas.com>

7.57.1 Description

This driver supports the Renesas ISL68137 and all 2nd generation Renesas digital multiphase voltage regulators (raa_dmpvr2). The ISL68137 is a digital output 7-phase configurable PWM controller with an AVSBus interface. 2nd generation devices are grouped into 4 distinct configurations: '1rail' for single-rail devices, '2rail' for dual-rail devices, '3rail' for 3-rail devices, and 'hv' for high voltage single-rail devices. Consult the individual datasheets for more information.

7.57.2 Usage Notes

This driver does not probe for PMBus devices. You will have to instantiate devices explicitly.

The ISL68137 AVS operation mode must be enabled/disabled at runtime.

Beyond the normal sysfs pmbus attributes, the driver exposes a control attribute for the ISL68137.

For 2nd generation Renesas digital multiphase voltage regulators, only the normal sysfs pmbus attributes are supported.

7.57.3 ISL68137 sysfs attributes

avs(0|1)_enable	Controls the AVS state of each rail.
curr1_label	"iin"
curr1_input	Measured input current
curr1_crit	Critical maximum current
curr1_crit_alarm	Current critical high alarm
curr[2-3]_label	"iout[1-2]"
curr[2-3]_input	Measured output current
curr[2-3]_crit	Critical maximum current
curr[2-3]_crit_alarm	Current critical high alarm
in1_label	"vin"
in1_input	Measured input voltage
in1_lcrit	Critical minimum input voltage
in1_lcrit_alarm	Input voltage critical low alarm
in1_crit	Critical maximum input voltage
in1_crit_alarm	Input voltage critical high alarm
in[2-3]_label	"vout[1-2]"
in[2-3]_input	Measured output voltage
in[2-3]_lcrit	Critical minimum output voltage
in[2-3]_lcrit_alarm	Output voltage critical low alarm
in[2-3]_crit	Critical maximum output voltage
in[2-3]_crit_alarm	Output voltage critical high alarm
power1_label	"pin"

Continued on next page

Table 4 – continued from previous page

power1_input	Measured input power
power1_alarm	Input power high alarm
power[2-3]_label	“pout[1-2]”
power[2-3]_input	Measured output power
temp[1-3]_input	Measured temperature
temp[1-3]_crit	Critical high temperature
temp[1-3]_crit_alarm	Chip temperature critical high alarm
temp[1-3]_max	Maximum temperature
temp[1-3]_max_alarm	Chip temperature high alarm

7.57.4 raa_dmpvr2_1rail/hv sysfs attributes

curr1_label	“iin”
curr1_input	Measured input current
curr1_crit	Critical maximum current
curr1_crit_alarm	Current critical high alarm
curr2_label	“iout”
curr2_input	Measured output current
curr2_crit	Critical maximum current
curr2_crit_alarm	Current critical high alarm
in1_label	“vin”
in1_input	Measured input voltage
in1_lcrit	Critical minimum input voltage
in1_lcrit_alarm	Input voltage critical low alarm
in1_crit	Critical maximum input voltage
in1_crit_alarm	Input voltage critical high alarm
in2_label	“vmon”
in2_input	Scaled VMON voltage read from the VMON pin
in3_label	“vout”
in3_input	Measured output voltage
in3_lcrit	Critical minimum output voltage
in3_lcrit_alarm	Output voltage critical low alarm
in3_crit	Critical maximum output voltage
in3_crit_alarm	Output voltage critical high alarm
power1_label	“pin”
power1_input	Measured input power
power1_alarm	Input power high alarm
power2_label	“pout”
power2_input	Measured output power
temp[1-3]_input	Measured temperature
temp[1-3]_crit	Critical high temperature
temp[1-3]_crit_alarm	Chip temperature critical high alarm
temp[1-3]_max	Maximum temperature
temp[1-3]_max_alarm	Chip temperature high alarm

7.57.5 raa_dmpvr2_2rail sysfs attributes

curr[1-2]_label	“iin[1-2]”
curr[1-2]_input	Measured input current
curr[1-2]_crit	Critical maximum current
curr[1-2]_crit_alarm	Current critical high alarm
curr[3-4]_label	“iout[1-2]”
curr[3-4]_input	Measured output current
curr[3-4]_crit	Critical maximum current
curr[3-4]_crit_alarm	Current critical high alarm
in1_label	“vin”
in1_input	Measured input voltage
in1_lcrit	Critical minimum input voltage
in1_lcrit_alarm	Input voltage critical low alarm
in1_crit	Critical maximum input voltage
in1_crit_alarm	Input voltage critical high alarm
in2_label	“vmon”
in2_input	Scaled VMON voltage read from the VMON pin
in[3-4]_label	“vout[1-2]”
in[3-4]_input	Measured output voltage
in[3-4]_lcrit	Critical minimum output voltage
in[3-4]_lcrit_alarm	Output voltage critical low alarm
in[3-4]_crit	Critical maximum output voltage
in[3-4]_crit_alarm	Output voltage critical high alarm
power[1-2]_label	“pin[1-2]”
power[1-2]_input	Measured input power
power[1-2]_alarm	Input power high alarm
power[3-4]_label	“pout[1-2]”
power[3-4]_input	Measured output power
temp[1-5]_input	Measured temperature
temp[1-5]_crit	Critical high temperature
temp[1-5]_crit_alarm	Chip temperature critical high alarm
temp[1-5]_max	Maximum temperature
temp[1-5]_max_alarm	Chip temperature high alarm

7.57.6 raa_dmpvr2_3rail sysfs attributes

curr[1-3]_label	“iin[1-3]”
curr[1-3]_input	Measured input current
curr[1-3]_crit	Critical maximum current
curr[1-3]_crit_alarm	Current critical high alarm
curr[4-6]_label	“iout[1-3]”
curr[4-6]_input	Measured output current
curr[4-6]_crit	Critical maximum current
curr[4-6]_crit_alarm	Current critical high alarm
in1_label	“vin”
in1_input	Measured input voltage

Continued on next page

Table 7 – continued from previous page

in1_lcrit	Critical minimum input voltage
in1_lcrit_alarm	Input voltage critical low alarm
in1_crit	Critical maximum input voltage
in1_crit_alarm	Input voltage critical high alarm
in2_label	“vmon”
in2_input	Scaled VMON voltage read from the VMON pin
in[3-5]_label	“vout[1-3]”
in[3-5]_input	Measured output voltage
in[3-5]_lcrit	Critical minimum output voltage
in[3-5]_lcrit_alarm	Output voltage critical low alarm
in[3-5]_crit	Critical maximum output voltage
in[3-5]_crit_alarm	Output voltage critical high alarm
power[1-3]_label	“pin[1-3]”
power[1-3]_input	Measured input power
power[1-3]_alarm	Input power high alarm
power[4-6]_label	“pout[1-3]”
power[4-6]_input	Measured output power
temp[1-7]_input	Measured temperature
temp[1-7]_crit	Critical high temperature
temp[1-7]_crit_alarm	Chip temperature critical high alarm
temp[1-7]_max	Maximum temperature
temp[1-7]_max_alarm	Chip temperature high alarm

7.58 Kernel driver it87

Supported chips:

- IT8603E/IT8623E

  Prefix: ‘it8603’

  Addresses scanned: from Super I/O config space (8 I/O ports)

  Datasheet: Not publicly available

- IT8620E

  Prefix: ‘it8620’

  Addresses scanned: from Super I/O config space (8 I/O ports)

- IT8628E

  Prefix: ‘it8628’

  Addresses scanned: from Super I/O config space (8 I/O ports)

  Datasheet: Not publicly available

- IT8705F

  Prefix: ‘it87’

  Addresses scanned: from Super I/O config space (8 I/O ports)

Datasheet: Once publicly available at the ITE website, but no longer

- IT8712F

  Prefix: 'it8712'

  Addresses scanned: from Super I/O config space (8 I/O ports)

  Datasheet: Once publicly available at the ITE website, but no longer

- IT8716F/IT8726F

  Prefix: 'it8716'

  Addresses scanned: from Super I/O config space (8 I/O ports)

  Datasheet: Once publicly available at the ITE website, but no longer

- IT8718F

  Prefix: 'it8718'

  Addresses scanned: from Super I/O config space (8 I/O ports)

  Datasheet: Once publicly available at the ITE website, but no longer

- IT8720F

  Prefix: 'it8720'

  Addresses scanned: from Super I/O config space (8 I/O ports)

  Datasheet: Not publicly available

- IT8721F/IT8758E

  Prefix: 'it8721'

  Addresses scanned: from Super I/O config space (8 I/O ports)

  Datasheet: Not publicly available

- IT8728F

  Prefix: 'it8728'

  Addresses scanned: from Super I/O config space (8 I/O ports)

  Datasheet: Not publicly available

- IT8732F

  Prefix: 'it8732'

  Addresses scanned: from Super I/O config space (8 I/O ports)

  Datasheet: Not publicly available

- IT8771E

  Prefix: 'it8771'

  Addresses scanned: from Super I/O config space (8 I/O ports)

  Datasheet: Not publicly available

- IT8772E

  Prefix: ‘it8772’

  Addresses scanned: from Super I/O config space (8 I/O ports)

  Datasheet: Not publicly available
- IT8781F

  Prefix: ‘it8781’

  Addresses scanned: from Super I/O config space (8 I/O ports)

  Datasheet: Not publicly available
- IT8782F

  Prefix: ‘it8782’

  Addresses scanned: from Super I/O config space (8 I/O ports)

  Datasheet: Not publicly available
- IT8783E/F

  Prefix: ‘it8783’

  Addresses scanned: from Super I/O config space (8 I/O ports)

  Datasheet: Not publicly available
- IT8786E

  Prefix: ‘it8786’

  Addresses scanned: from Super I/O config space (8 I/O ports)

  Datasheet: Not publicly available
- IT8790E

  Prefix: ‘it8790’

  Addresses scanned: from Super I/O config space (8 I/O ports)

  Datasheet: Not publicly available
- SiS950 [clone of IT8705F]

  Prefix: ‘it87’

  Addresses scanned: from Super I/O config space (8 I/O ports)

  Datasheet: No longer be available

Authors:

- Christophe Gauthron
- Jean Delvare <jdelvare@suse.de>

7.58.1 Module Parameters

- **update_vbat: int** 0 if vbat should report power on value, 1 if vbat should be updated after each read. Default is 0. On some boards the battery voltage is provided by either the battery or the onboard power supply. Only the first reading at power on will be the actual battery voltage (which the chip does automatically). On other boards the battery voltage is always fed to the chip so can be read at any time. Excessive reading may decrease battery life but no information is given in the datasheet.
- **fix_pwm_polarity int** Force PWM polarity to active high (DANGEROUS). Some chips are misconfigured by BIOS - PWM values would be inverted. This option tries to fix this. Please contact your BIOS manufacturer and ask him for fix.

7.58.2 Hardware Interfaces

All the chips supported by this driver are LPC Super-I/O chips, accessed through the LPC bus (ISA-like I/O ports). The IT8712F additionally has an SMBus interface to the hardware monitoring functions. This driver no longer supports this interface though, as it is slower and less reliable than the ISA access, and was only available on a small number of motherboard models.

7.58.3 Description

This driver implements support for the IT8603E, IT8620E, IT8623E, IT8628E, IT8705F, IT8712F, IT8716F, IT8718F, IT8720F, IT8721F, IT8726F, IT8728F, IT8732F, IT8758E, IT8771E, IT8772E, IT8781F, IT8782F, IT8783E/F, IT8786E, IT8790E, and SiS950 chips.

These chips are 'Super I/O chips', supporting floppy disks, infrared ports, joysticks and other miscellaneous stuff. For hardware monitoring, they include an 'environment controller' with 3 temperature sensors, 3 fan rotation speed sensors, 8 voltage sensors, associated alarms, and chassis intrusion detection.

The IT8712F and IT8716F additionally feature VID inputs, used to report the Vcore voltage of the processor. The early IT8712F have 5 VID pins, the IT8716F and late IT8712F have 6. They are shared with other functions though, so the functionality may not be available on a given system.

The IT8718F and IT8720F also features VID inputs (up to 8 pins) but the value is stored in the Super-I/O configuration space. Due to technical limitations, this value can currently only be read once at initialization time, so the driver won't notice and report changes in the VID value. The two upper VID bits share their pins with voltage inputs (in5 and in6) so you can't have both on a given board.

The IT8716F, IT8718F, IT8720F, IT8721F/IT8758E and later IT8712F revisions have support for 2 additional fans. The additional fans are supported by the driver.

The IT8716F, IT8718F, IT8720F, IT8721F/IT8758E, IT8732F, IT8781F, IT8782F, IT8783E/F, and late IT8712F and IT8705F also have optional 16-bit tachometer counters for fans 1 to 3. This is better (no more fan clock divider mess) but not

compatible with the older chips and revisions. The 16-bit tachometer mode is enabled by the driver when one of the above chips is detected.

The IT8726F is just bit enhanced IT8716F with additional hardware for AMD power sequencing. Therefore the chip will appear as IT8716F to userspace applications.

The IT8728F, IT8771E, and IT8772E are considered compatible with the IT8721F, until a datasheet becomes available (hopefully.)

The IT8603E/IT8623E is a custom design, hardware monitoring part is similar to IT8728F. It only supports 3 fans, 16-bit fan mode, and the full speed mode of the fan is not supported (value 0 of pwmX_enable).

The IT8620E and IT8628E are custom designs, hardware monitoring part is similar to IT8728F. It only supports 16-bit fan mode. Both chips support up to 6 fans.

The IT8790E supports up to 3 fans. 16-bit fan mode is always enabled.

The IT8732F supports a closed-loop mode for fan control, but this is not currently implemented by the driver.

Temperatures are measured in degrees Celsius. An alarm is triggered once when the Overtemperature Shutdown limit is crossed.

Fan rotation speeds are reported in RPM (rotations per minute). An alarm is triggered if the rotation speed has dropped below a programmable limit. When 16-bit tachometer counters aren't used, fan readings can be divided by a programmable divider (1, 2, 4 or 8) to give the readings more range or accuracy. With a divider of 2, the lowest representable value is around 2600 RPM. Not all RPM values can accurately be represented, so some rounding is done.

Voltage sensors (also known as IN sensors) report their values in volts. An alarm is triggered if the voltage has crossed a programmable minimum or maximum limit. Note that minimum in this case always means 'closest to zero'; this is important for negative voltage measurements. On most chips, all voltage inputs can measure voltages between 0 and 4.08 volts, with a resolution of 0.016 volt. IT8603E, IT8721F/IT8758E and IT8728F can measure between 0 and 3.06 volts, with a resolution of 0.012 volt. IT8732F can measure between 0 and 2.8 volts with a resolution of 0.0109 volt. The battery voltage in8 does not have limit registers.

On the IT8603E, IT8620E, IT8628E, IT8721F/IT8758E, IT8732F, IT8781F, IT8782F, and IT8783E/F, some voltage inputs are internal and scaled inside the chip: * in3 (optional) * in7 (optional for IT8781F, IT8782F, and IT8783E/F) * in8 (always) * in9 (relevant for IT8603E only) The driver handles this transparently so user-space doesn't have to care.

The VID lines (IT8712F/IT8716F/IT8718F/IT8720F) encode the core voltage value: the voltage level your processor should work with. This is hardcoded by the mainboard and/or processor itself. It is a value in volts.

If an alarm triggers, it will remain triggered until the hardware register is read at least once. This means that the cause for the alarm may already have disappeared! Note that in the current implementation, all hardware registers are read whenever any data is read (unless it is less than 1.5 seconds since the last update). This means that you can easily miss once-only alarms.

Out-of-limit readings can also result in beeping, if the chip is properly wired and configured. Beeping can be enabled or disabled per sensor type (temperatures, voltages and fans.)

The IT87xx only updates its values each 1.5 seconds; reading it more often will do no harm, but will return 'old' values.

To change sensor N to a thermistor, 'echo 4 > tempN_type' where N is 1, 2, or 3. To change sensor N to a thermal diode, 'echo 3 > tempN_type'. Give 0 for unused sensor. Any other value is invalid. To configure this at startup, consult lm_sensors' s /etc/sensors.conf. (4 = thermistor; 3 = thermal diode)

7.58.4 Fan speed control

The fan speed control features are limited to manual PWM mode. Automatic "Smart Guardian" mode control handling is only implemented for older chips (see below.) However if you want to go for "manual mode"just write 1 to pwmN_enable.

If you are only able to control the fan speed with very small PWM values, try lowering the PWM base frequency (pwm1_freq). Depending on the fan, it may give you a somewhat greater control range. The same frequency is used to drive all fan outputs, which is why pwm2_freq and pwm3_freq are read-only.

7.58.5 Automatic fan speed control (old interface)

The driver supports the old interface to automatic fan speed control which is implemented by IT8705F chips up to revision F and IT8712F chips up to revision G.

This interface implements 4 temperature vs. PWM output trip points. The PWM output of trip point 4 is always the maximum value (fan running at full speed) while the PWM output of the other 3 trip points can be freely chosen. The temperature of all 4 trip points can be freely chosen. Additionally, trip point 1 has an hysteresis temperature attached, to prevent fast switching between fan on and off.

The chip automatically computes the PWM output value based on the input temperature, based on this simple rule: if the temperature value is between trip point N and trip point N+1 then the PWM output value is the one of trip point N. The automatic control mode is less flexible than the manual control mode, but it reacts faster, is more robust and doesn't use CPU cycles.

Trip points must be set properly before switching to automatic fan speed control mode. The driver will perform basic integrity checks before actually switching to automatic control mode.

7.58.6 Temperature offset attributes

The driver supports temp[1-3]_offset sysfs attributes to adjust the reported temperature for thermal diodes or diode-connected thermal transistors. If a temperature sensor is configured for thermistors, the attribute values are ignored. If the thermal sensor type is Intel PECI, the temperature offset must be programmed to the critical CPU temperature.

7.59 Kernel driver jc42

Supported chips:

- Analog Devices ADT7408

  Datasheets:

  http://www.analog.com/static/imported-files/data_sheets/ADT7408.pdf

- Atmel AT30TS00, AT30TS002A/B, AT30TSE004A

  Datasheets:

  http://www.atmel.com/Images/doc8585.pdf

  http://www.atmel.com/Images/doc8711.pdf

  http://www.atmel.com/Images/Atmel-8852-SEEPROM-AT30TSE002A-Datasheet.pdf

  http://www.atmel.com/Images/Atmel-8868-DTS-AT30TSE004A-Datasheet.pdf

- IDT TSE2002B3, TSE2002GB2, TSE2004GB2, TS3000B3, TS3000GB0, TS3000GB2,

  TS3001GB2

  Datasheets:

  Available from IDT web site

- Maxim MAX6604

  Datasheets:

  http://datasheets.maxim-ic.com/en/ds/MAX6604.pdf

- Microchip MCP9804, MCP9805, MCP9808, MCP98242, MCP98243, MCP98244, MCP9843

  Datasheets:

  http://ww1.microchip.com/downloads/en/DeviceDoc/22203C.pdf

  http://ww1.microchip.com/downloads/en/DeviceDoc/21977b.pdf

http://ww1.microchip.com/downloads/en/DeviceDoc/25095A.pdf

http://ww1.microchip.com/downloads/en/DeviceDoc/21996a.pdf

http://ww1.microchip.com/downloads/en/DeviceDoc/22153c.pdf

http://ww1.microchip.com/downloads/en/DeviceDoc/22327A.pdf

- NXP Semiconductors SE97, SE97B, SE98, SE98A

  Datasheets:

  http://www.nxp.com/documents/data_sheet/SE97.pdf

  http://www.nxp.com/documents/data_sheet/SE97B.pdf

  http://www.nxp.com/documents/data_sheet/SE98.pdf

  http://www.nxp.com/documents/data_sheet/SE98A.pdf

- ON Semiconductor CAT34TS02, CAT6095

  Datasheet:

  http://www.onsemi.com/pub_link/Collateral/CAT34TS02-D.PDF

  http://www.onsemi.com/pub/Collateral/CAT6095-D.PDF

- ST Microelectronics STTS424, STTS424E02, STTS2002, STTS2004, STTS3000

  Datasheets:

  http://www.st.com/web/en/resource/technical/document/datasheet/CD00157556.pdf

  http://www.st.com/web/en/resource/technical/document/datasheet/CD00157558.pdf

  http://www.st.com/web/en/resource/technical/document/datasheet/CD00266638.pdf

  http://www.st.com/web/en/resource/technical/document/datasheet/CD00225278.pdf

  http://www.st.com/web/en/resource/technical/document/datasheet/DM00076709.pdf

- JEDEC JC 42.4 compliant temperature sensor chips

  Datasheet:

  http://www.jedec.org/sites/default/files/docs/4_01_04R19.pdf

Common for all chips:

Prefix: 'jc42'

Addresses scanned: I2C 0x18 - 0x1f

Author: Guenter Roeck <linux@roeck-us.net>

7.59.1 Description

This driver implements support for JEDEC JC 42.4 compliant temperature sensors, which are used on many DDR3 memory modules for mobile devices and servers. Some systems use the sensor to prevent memory overheating by automatically throttling the memory controller.

The driver auto-detects the chips listed above, but can be manually instantiated to support other JC 42.4 compliant chips.

Example: the following will load the driver for a generic JC 42.4 compliant temperature sensor at address 0x18 on I2C bus #1:

```
# modprobe jc42
# echo jc42 0x18 > /sys/bus/i2c/devices/i2c-1/new_device
```

A JC 42.4 compliant chip supports a single temperature sensor. Minimum, maximum, and critical temperature can be configured. There are alarms for high, low, and critical thresholds.

There is also an hysteresis to control the thresholds for resetting alarms. Per JC 42.4 specification, the hysteresis threshold can be configured to 0, 1.5, 3.0, and 6.0 degrees C. Configured hysteresis values will be rounded to those limits. The chip supports only a single register to configure the hysteresis, which applies to all limits. This register can be written by writing into temp1_crit_hyst. Other hysteresis attributes are read-only.

If the BIOS has configured the sensor for automatic temperature management, it is likely that it has locked the registers, i.e., that the temperature limits cannot be changed.

7.59.2 Sysfs entries

temp1_input	Temperature (RO)
temp1_min	Minimum temperature (RO or RW)
temp1_max	Maximum temperature (RO or RW)
temp1_crit	Critical high temperature (RO or RW)
temp1_crit_hyst	Critical hysteresis temperature (RO or RW)
temp1_max_hyst	Maximum hysteresis temperature (RO)
temp1_min_alarm	Temperature low alarm
temp1_max_alarm	Temperature high alarm
temp1_crit_alarm	Temperature critical alarm

7.60 Kernel driver k10temp

Supported chips:

- AMD Family 10h processors:

  Socket F: Quad-Core/Six-Core/Embedded Opteron (but see below)

  Socket AM2+: Quad-Core Opteron, Phenom (II) X3/X4, Athlon X2 (but see below)

  Socket AM3: Quad-Core Opteron, Athlon/Phenom II X2/X3/X4, Sempron II

  Socket S1G3: Athlon II, Sempron, Turion II

- AMD Family 11h processors:

  Socket S1G2: Athlon (X2), Sempron (X2), Turion X2 (Ultra)

- AMD Family 12h processors: “Llano” (E2/A4/A6/A8-Series)
- AMD Family 14h processors: “Brazos” (C/E/G/Z-Series)
- AMD Family 15h processors: “Bulldozer” (FX-Series), “Trinity”, “Kaveri”, “Carrizo”, “Stoney Ridge”, “Bristol Ridge”
- AMD Family 16h processors: “Kabini”, “Mullins”
- AMD Family 17h processors: “Zen”, “Zen 2”
- AMD Family 18h processors: “Hygon Dhyana”
- AMD Family 19h processors: “Zen 3”

  Prefix: ‘k10temp’

  Addresses scanned: PCI space

  Datasheets:

  BIOS and Kernel Developer’s Guide (BKDG) For AMD Family 10h Processors:

  http://support.amd.com/us/Processor_TechDocs/31116.pdf

  BIOS and Kernel Developer’s Guide (BKDG) for AMD Family 11h Processors:

  http://support.amd.com/us/Processor_TechDocs/41256.pdf

  BIOS and Kernel Developer’s Guide (BKDG) for AMD Family 12h Processors:

  http://support.amd.com/us/Processor_TechDocs/41131.pdf

  BIOS and Kernel Developer’s Guide (BKDG) for AMD Family 14h Models 00h-0Fh Processors:

  http://support.amd.com/us/Processor_TechDocs/43170.pdf

  Revision Guide for AMD Family 10h Processors:

  http://support.amd.com/us/Processor_TechDocs/41322.pdf

  Revision Guide for AMD Family 11h Processors:

  http://support.amd.com/us/Processor_TechDocs/41788.pdf

  Revision Guide for AMD Family 12h Processors:

http://support.amd.com/us/Processor_TechDocs/44739.pdf

Revision Guide for AMD Family 14h Models 00h-0Fh Processors:

http://support.amd.com/us/Processor_TechDocs/47534.pdf

AMD Family 11h Processor Power and Thermal Data Sheet for Notebooks:

http://support.amd.com/us/Processor_TechDocs/43373.pdf

AMD Family 10h Server and Workstation Processor Power and Thermal Data Sheet:

http://support.amd.com/us/Processor_TechDocs/43374.pdf

AMD Family 10h Desktop Processor Power and Thermal Data Sheet:

http://support.amd.com/us/Processor_TechDocs/43375.pdf

Author: Clemens Ladisch <clemens@ladisch.de>

7.60.1 Description

This driver permits reading of the internal temperature sensor of AMD Family 10h/11h/12h/14h/15h/16h processors.

All these processors have a sensor, but on those for Socket F or AM2+, the sensor may return inconsistent values (erratum 319). The driver will refuse to load on these revisions unless you specify the "force=1" module parameter.

Due to technical reasons, the driver can detect only the mainboard's socket type, not the processor's actual capabilities. Therefore, if you are using an AM3 processor on an AM2+ mainboard, you can safely use the "force=1" parameter.

For CPUs older than Family 17h, there is one temperature measurement value, available as temp1_input in sysfs. It is measured in degrees Celsius with a resolution of 1/8th degree. Please note that it is defined as a relative value; to quote the AMD manual:

```
Tctl is the processor temperature control value, used by the platform to
control cooling systems. Tctl is a non-physical temperature on an
arbitrary scale measured in degrees. It does _not_ represent an actual
physical temperature like die or case temperature. Instead, it specifies
the processor temperature relative to the point at which the system must
supply the maximum cooling for the processor's specified maximum case
temperature and maximum thermal power dissipation.
```

The maximum value for Tctl is available in the file temp1_max.

If the BIOS has enabled hardware temperature control, the threshold at which the processor will throttle itself to avoid damage is available in temp1_crit and temp1_crit_hyst.

On some AMD CPUs, there is a difference between the die temperature (Tdie) and the reported temperature (Tctl). Tdie is the real measured temperature, and Tctl is used for fan control. While Tctl is always available as temp1_input, the driver exports Tdie temperature as temp2_input for those CPUs which support it.

Models from 17h family report relative temperature, the driver aims to compensate and report the real temperature.

On Family 17h and Family 18h CPUs, additional temperature sensors may report Core Complex Die (CCD) temperatures. Up to 8 such temperatures are reported as temp{3..10}_input, labeled Tccd{1..8}. Actual support depends on the CPU variant.

Various Family 17h and 18h CPUs report voltage and current telemetry information. The following attributes may be reported.

Attribute Label Description ================ ======= ================ in0_input Vcore Core voltage in1_input Vsoc SoC voltage curr1_input Icore Core current curr2_input Isoc SoC current ================ ======= ================

Current values are raw (unscaled) as reported by the CPU. Core current is reported as multiples of 1A / LSB. SoC is reported as multiples of 0.25A / LSB. The real current is board specific. Reported currents should be seen as rough guidance, and should be scaled using sensors3.conf as appropriate for a given board.

7.61 Kernel driver k8temp

Supported chips:

- AMD Athlon64/FX or Opteron CPUs

  Prefix: 'k8temp'

  Addresses scanned: PCI space

  Datasheet: http://www.amd.com/system/files/TechDocs/32559.pdf

Author: Rudolf Marek

Contact: Rudolf Marek <r.marek@assembler.cz>

7.61.1 Description

This driver permits reading temperature sensor(s) embedded inside AMD K8 family CPUs (Athlon64/FX, Opteron). Official documentation says that it works from revision F of K8 core, but in fact it seems to be implemented for all revisions of K8 except the first two revisions (SH-B0 and SH-B3).

Please note that you will need at least lm-sensors 2.10.1 for proper userspace support.

There can be up to four temperature sensors inside single CPU. The driver will auto-detect the sensors and will display only temperatures from implemented sensors.

Mapping of /sys files is as follows:

temp1_input	temperature of Core 0 and “place” 0
temp2_input	temperature of Core 0 and “place” 1
temp3_input	temperature of Core 1 and “place” 0
temp4_input	temperature of Core 1 and “place” 1

Temperatures are measured in degrees Celsius and measurement resolution is 1 degree C. It is expected that future CPU will have better resolution. The temperature is updated once a second. Valid temperatures are from -49 to 206 degrees C.

Temperature known as TCaseMax was specified for processors up to revision E. This temperature is defined as temperature between heat-spreader and CPU case, so the internal CPU temperature supplied by this driver can be higher. There is no easy way how to measure the temperature which will correlate with TCaseMax temperature.

For newer revisions of CPU (rev F, socket AM2) there is a mathematically computed temperature called TControl, which must be lower than TControlMax.

The relationship is following:

temp1_input - TjOffset*2 < TControlMax,

TjOffset is not yet exported by the driver, TControlMax is usually 70 degrees C. The rule of the thumb -> CPU temperature should not cross 60 degrees C too much.

7.62 Kernel driver lineage-pem

Supported devices:

- Lineage Compact Power Line Power Entry Modules

  Prefix: ‘lineage-pem’

  Addresses scanned: -

  Documentation:

  http://www.lineagepower.com/oem/pdf/CPLI2C.pdf

Author: Guenter Roeck <linux@roeck-us.net>

7.62.1 Description

This driver supports various Lineage Compact Power Line DC/DC and AC/DC converters such as CP1800, CP2000AC, CP2000DC, CP2100DC, and others.

Lineage CPL power entry modules are nominally PMBus compliant. However, most standard PMBus commands are not supported. Specifically, all hardware monitoring and status reporting commands are non-standard. For this reason, a standard PMBus driver can not be used.

7.62.2 Usage Notes

This driver does not probe for Lineage CPL devices, since there is no register which can be safely used to identify the chip. You will have to instantiate the devices explicitly.

Example: the following will load the driver for a Lineage PEM at address 0x40 on I2C bus #1:

```
$ modprobe lineage-pem
$ echo lineage-pem 0x40 > /sys/bus/i2c/devices/i2c-1/new_device
```

All Lineage CPL power entry modules have a built-in I2C bus master selector (PCA9541). To ensure device access, this driver should only be used as client driver to the pca9541 I2C master selector driver.

7.62.3 Sysfs entries

All Lineage CPL devices report output voltage and device temperature as well as alarms for output voltage, temperature, input voltage, input current, input power, and fan status.

Input voltage, input current, input power, and fan speed measurement is only supported on newer devices. The driver detects if those attributes are supported, and only creates respective sysfs entries if they are.

in1_input	Output voltage (mV)
in1_min_alarm	Output undervoltage alarm
in1_max_alarm	Output overvoltage alarm
in1_crit	Output voltage critical alarm
in2_input	Input voltage (mV, optional)
in2_alarm	Input voltage alarm
curr1_input	Input current (mA, optional)
curr1_alarm	Input overcurrent alarm
power1_input	Input power (uW, optional)
power1_alarm	Input power alarm
fan1_input	Fan 1 speed (rpm, optional)
fan2_input	Fan 2 speed (rpm, optional)
fan3_input	Fan 3 speed (rpm, optional)
temp1_input	
temp1_max	
temp1_crit	
temp1_alarm	
temp1_crit_alarm	
temp1_fault	

7.63 Kernel driver lm25066

Supported chips:

- TI LM25056

  Prefix: 'lm25056'

  Addresses scanned: -

  Datasheets:

  http://www.ti.com/lit/gpn/lm25056

  http://www.ti.com/lit/gpn/lm25056a

- National Semiconductor LM25066

  Prefix: 'lm25066'

  Addresses scanned: -

  Datasheets:

  http://www.national.com/pf/LM/LM25066.html

  http://www.national.com/pf/LM/LM25066A.html

- National Semiconductor LM5064

  Prefix: 'lm5064'

  Addresses scanned: -

  Datasheet:

  http://www.national.com/pf/LM/LM5064.html

- National Semiconductor LM5066

  Prefix: 'lm5066'

  Addresses scanned: -

  Datasheet:

  http://www.national.com/pf/LM/LM5066.html

- Texas Instruments LM5066I

  Prefix: 'lm5066i'

  Addresses scanned: -

  Datasheet:

  http://www.ti.com/product/LM5066I

Author: Guenter Roeck <linux@roeck-us.net>

7.63.1 Description

This driver supports hardware monitoring for National Semiconductor / TI LM25056, LM25066, LM5064, and LM5066/LM5066I Power Management, Monitoring, Control, and Protection ICs.

The driver is a client driver to the core PMBus driver. Please see Documentation/hwmon/pmbus.rst for details on PMBus client drivers.

7.63.2 Usage Notes

This driver does not auto-detect devices. You will have to instantiate the devices explicitly. Please see Documentation/i2c/instantiating-devices.rst for details.

7.63.3 Platform data support

The driver supports standard PMBus driver platform data.

7.63.4 Sysfs entries

The following attributes are supported. Limits are read-write; all other attributes are read-only.

in1_label	“vin”
in1_input	Measured input voltage.
in1_average	Average measured input voltage.
in1_min	Minimum input voltage.
in1_max	Maximum input voltage.
in1_min_alarm	Input voltage low alarm.
in1_max_alarm	Input voltage high alarm.
in2_label	“vmon”
in2_input	Measured voltage on VAUX pin
in2_min	Minimum VAUX voltage (LM25056 only).
in2_max	Maximum VAUX voltage (LM25056 only).
in2_min_alarm	VAUX voltage low alarm (LM25056 only).
in2_max_alarm	VAUX voltage high alarm (LM25056 only).
in3_label	“vout1” Not supported on LM25056.
in3_input	Measured output voltage.
in3_average	Average measured output voltage.
in3_min	Minimum output voltage.
in3_min_alarm	Output voltage low alarm.
curr1_label	“iin”
curr1_input	Measured input current.
curr1_average	Average measured input current.
curr1_max	Maximum input current.
curr1_max_alarm	Input current high alarm.
power1_label	“pin”
power1_input	Measured input power.

Continued on next page

Table 8 – continued from previous page

power1_average	Average measured input power.
power1_max	Maximum input power limit.
power1_alarm	Input power alarm
power1_input_highest	Historical maximum power.
power1_reset_history	Write any value to reset maximum power history.
temp1_input	Measured temperature.
temp1_max	Maximum temperature.
temp1_crit	Critical high temperature.
temp1_max_alarm	Chip temperature high alarm.
temp1_crit_alarm	Chip temperature critical high alarm.

7.64 Kernel driver lm63

Supported chips:

- National Semiconductor LM63

  Prefix: 'lm63'

  Addresses scanned: I2C 0x4c

  Datasheet: Publicly available at the National Semiconductor website

  http://www.national.com/pf/LM/LM63.html

- National Semiconductor LM64

  Prefix: 'lm64'

  Addresses scanned: I2C 0x18 and 0x4e

  Datasheet: Publicly available at the National Semiconductor website

  http://www.national.com/pf/LM/LM64.html

- National Semiconductor LM96163

  Prefix: 'lm96163'

  Addresses scanned: I2C 0x4c

  Datasheet: Publicly available at the National Semiconductor website

  http://www.national.com/pf/LM/LM96163.html

Author: Jean Delvare <jdelvare@suse.de>

Thanks go to Tyan and especially Alex Buckingham for setting up a remote access to their S4882 test platform for this driver.

http://www.tyan.com/

7.64.1 Description

The LM63 is a digital temperature sensor with integrated fan monitoring and control.

The LM63 is basically an LM86 with fan speed monitoring and control capabilities added. It misses some of the LM86 features though:

- No low limit for local temperature.
- No critical limit for local temperature.
- Critical limit for remote temperature can be changed only once. We will consider that the critical limit is read-only.

The datasheet isn't very clear about what the tachometer reading is.

An explanation from National Semiconductor: The two lower bits of the read value have to be masked out. The value is still 16 bit in width.

All temperature values are given in degrees Celsius. Resolution is 1.0 degree for the local temperature, 0.125 degree for the remote temperature.

The fan speed is measured using a tachometer. Contrary to most chips which store the value in an 8-bit register and have a selectable clock divider to make sure that the result will fit in the register, the LM63 uses 16-bit value for measuring the speed of the fan. It can measure fan speeds down to 83 RPM, at least in theory.

Note that the pin used for fan monitoring is shared with an alert out function. Depending on how the board designer wanted to use the chip, fan speed monitoring will or will not be possible. The proper chip configuration is left to the BIOS, and the driver will blindly trust it. Only the original LM63 suffers from this limitation, the LM64 and LM96163 have separate pins for fan monitoring and alert out. On the LM64, monitoring is always enabled; on the LM96163 it can be disabled.

A PWM output can be used to control the speed of the fan. The LM63 has two PWM modes: manual and automatic. Automatic mode is not fully implemented yet (you cannot define your custom PWM/temperature curve), and mode change isn't supported either.

The lm63 driver will not update its values more frequently than configured with the update_interval sysfs attribute; reading them more often will do no harm, but will return 'old' values. Values in the automatic fan control lookup table (attributes pwm1_auto_*) have their own independent lifetime of 5 seconds.

The LM64 is effectively an LM63 with GPIO lines. The driver does not support these GPIO lines at present.

The LM96163 is an enhanced version of LM63 with improved temperature accuracy and better PWM resolution. For LM96163, the external temperature sensor type is configurable as CPU embedded diode(1) or 3904 transistor(2).

7.65 Kernel driver lm70

Supported chips:

- National Semiconductor LM70

  Datasheet: http://www.national.com/pf/LM/LM70.html

- Texas Instruments TMP121/TMP123

  Information: http://focus.ti.com/docs/prod/folders/print/tmp121.html

- Texas Instruments TMP122/TMP124

  Information: http://www.ti.com/product/tmp122

- National Semiconductor LM71

  Datasheet: http://www.ti.com/product/LM71

- National Semiconductor LM74

  Datasheet: http://www.ti.com/product/LM74

Author: Kaiwan N Billimoria <kaiwan@designergraphix.com>

7.65.1 Description

This driver implements support for the National Semiconductor LM70 temperature sensor.

The LM70 temperature sensor chip supports a single temperature sensor. It communicates with a host processor (or microcontroller) via an SPI/Microwire Bus interface.

Communication with the LM70 is simple: when the temperature is to be sensed, the driver accesses the LM70 using SPI communication: 16 SCLK cycles comprise the MOSI/MISO loop. At the end of the transfer, the 11-bit 2's complement digital temperature (sent via the SIO line), is available in the driver for interpretation. This driver makes use of the kernel's in-core SPI support.

As a real (in-tree) example of this "SPI protocol driver" interfacing with a "SPI master controller driver", see drivers/spi/spi_lm70llp.c and its associated documentation.

The LM74 and TMP121/TMP122/TMP123/TMP124 are very similar; main difference is 13-bit temperature data (0.0625 degrees celsius resolution).

The TMP122/TMP124 also feature configurable temperature thresholds.

The LM71 is also very similar; main difference is 14-bit temperature data (0.03125 degrees celsius resolution).

7.65.2 Thanks to

Jean Delvare <jdelvare@suse.de> for mentoring the hwmon-side driver development.

7.66 Kernel driver lm73

Supported chips:

- Texas Instruments LM73

  Prefix: 'lm73'

  Addresses scanned: I2C 0x48, 0x49, 0x4a, 0x4c, 0x4d, and 0x4e

  Datasheet: Publicly available at the Texas Instruments website

  http://www.ti.com/product/lm73

Author: Guillaume Ligneul <guillaume.ligneul@gmail.com>

Documentation: Chris Verges <kg4ysn@gmail.com>

7.66.1 Description

The LM73 is a digital temperature sensor. All temperature values are given in degrees Celsius.

7.66.2 Measurement Resolution Support

The LM73 supports four resolutions, defined in terms of degrees C per LSB: 0.25, 0.125, 0.0625, and 0.3125. Changing the resolution mode affects the conversion time of the LM73's analog-to-digital converter. From userspace, the desired resolution can be specified as a function of conversion time via the 'update_interval' sysfs attribute for the device. This attribute will normalize ranges of input values to the maximum times defined for the resolution in the datasheet.

Resolution	Conv. Time	Input Range
(C/LSB)	(msec)	(msec)
0.25	14	0..14
0.125	28	15..28
0.0625	56	29..56
0.03125	112	57..infinity

The following examples show how the 'update_interval' attribute can be used to change the conversion time:

```
$ echo 0 > update_interval
$ cat update_interval
14
$ cat temp1_input
```

(continues on next page)

(continued from previous page)

```
24250

$ echo 22 > update_interval
$ cat update_interval
28
$ cat temp1_input
24125

$ echo 56 > update_interval
$ cat update_interval
56
$ cat temp1_input
24062

$ echo 85 > update_interval
$ cat update_interval
112
$ cat temp1_input
24031
```

As shown here, the lm73 driver automatically adjusts any user input for 'update_interval' via a step function. Reading back the 'update_interval' value after a write operation will confirm the conversion time actively in use.

Mathematically, the resolution can be derived from the conversion time via the following function:

g(x) = 0.250 * [log(x/14) / log(2)]

where 'x' is the output from 'update_interval' and 'g(x)' is the resolution in degrees C per LSB.

7.66.3 Alarm Support

The LM73 features a simple over-temperature alarm mechanism. This feature is exposed via the sysfs attributes.

The attributes 'temp1_max_alarm' and 'temp1_min_alarm' are flags provided by the LM73 that indicate whether the measured temperature has passed the 'temp1_max' and 'temp1_min' thresholds, respectively. These values _must_ be read to clear the registers on the LM73.

7.67 Kernel driver lm75

Supported chips:

- National Semiconductor LM75

  Prefix: 'lm75'

  Addresses scanned: I2C 0x48 - 0x4f

  Datasheet: Publicly available at the National Semiconductor website

  http://www.national.com/

- National Semiconductor LM75A

  Prefix: 'lm75a'

  Addresses scanned: I2C 0x48 - 0x4f

  Datasheet: Publicly available at the National Semiconductor website

  http://www.national.com/

- Dallas Semiconductor (now Maxim) DS75, DS1775, DS7505

  Prefixes: 'ds75', 'ds1775', 'ds7505'

  Addresses scanned: none

  Datasheet: Publicly available at the Maxim website

  http://www.maximintegrated.com/

- Maxim MAX6625, MAX6626, MAX31725, MAX31726

  Prefixes: 'max6625', 'max6626', 'max31725', 'max31726'

  Addresses scanned: none

  Datasheet: Publicly available at the Maxim website

  http://www.maxim-ic.com/

- Microchip (TelCom) TCN75

  Prefix: 'tcn75'

  Addresses scanned: none

  Datasheet: Publicly available at the Microchip website

  http://www.microchip.com/

- Microchip MCP9800, MCP9801, MCP9802, MCP9803

  Prefix: 'mcp980x'

  Addresses scanned: none

  Datasheet: Publicly available at the Microchip website

  http://www.microchip.com/

- Analog Devices ADT75

  Prefix: 'adt75'

  Addresses scanned: none

  Datasheet: Publicly available at the Analog Devices website

  http://www.analog.com/adt75

- ST Microelectronics STDS75

  Prefix: 'stds75'

  Addresses scanned: none

  Datasheet: Publicly available at the ST website

http://www.st.com/internet/analog/product/121769.jsp

- ST Microelectronics STLM75

  Prefix: 'stlm75'

  Addresses scanned: none

  Datasheet: Publicly available at the ST website

  https://www.st.com/resource/en/datasheet/stlm75.pdf

- Texas Instruments TMP100, TMP101, TMP105, TMP112, TMP75, TMP75B, TMP75C, TMP175, TMP275

  Prefixes: 'tmp100', 'tmp101', 'tmp105', 'tmp112', 'tmp175', 'tmp75', 'tmp75b', 'tmp75c', 'tmp275'

  Addresses scanned: none

  Datasheet: Publicly available at the Texas Instruments website

  http://www.ti.com/product/tmp100

  http://www.ti.com/product/tmp101

  http://www.ti.com/product/tmp105

  http://www.ti.com/product/tmp112

  http://www.ti.com/product/tmp75

  http://www.ti.com/product/tmp75b

  http://www.ti.com/product/tmp75c

  http://www.ti.com/product/tmp175

  http://www.ti.com/product/tmp275

- NXP LM75B, PCT2075

  Prefix: 'lm75b', 'pct2075'

  Addresses scanned: none

  Datasheet: Publicly available at the NXP website

  http://www.nxp.com/documents/data_sheet/LM75B.pdf

  http://www.nxp.com/docs/en/data-sheet/PCT2075.pdf

Author: Frodo Looijaard <frodol@dds.nl>

7.67.1 Description

The LM75 implements one temperature sensor. Limits can be set through the Overtemperature Shutdown register and Hysteresis register. Each value can be set and read to half-degree accuracy. An alarm is issued (usually to a connected LM78) when the temperature gets higher then the Overtemperature Shutdown value; it stays on until the temperature falls below the Hysteresis value. All temperatures are in degrees Celsius, and are guaranteed within a range of -55 to +125 degrees.

The driver caches the values for a period varying between 1 second for the slowest chips and 125 ms for the fastest chips; reading it more often will do no harm, but will return 'old' values.

The original LM75 was typically used in combination with LM78-like chips on PC motherboards, to measure the temperature of the processor(s). Clones are now used in various embedded designs.

The LM75 is essentially an industry standard; there may be other LM75 clones not listed here, with or without various enhancements, that are supported. The clones are not detected by the driver, unless they reproduce the exact register tricks of the original LM75, and must therefore be instantiated explicitly. Higher resolution up to 16-bit is supported by this driver, other specific enhancements are not.

The LM77 is not supported, contrary to what we pretended for a long time. Both chips are simply not compatible, value encoding differs.

7.68 Kernel driver lm77

Supported chips:

- National Semiconductor LM77

  Prefix: 'lm77'

  Addresses scanned: I2C 0x48 - 0x4b

  Datasheet: Publicly available at the National Semiconductor website

  http://www.national.com/

Author: Andras BALI <drewie@freemail.hu>

7.68.1 Description

The LM77 implements one temperature sensor. The temperature sensor incorporates a band-gap type temperature sensor, 10-bit ADC, and a digital comparator with user-programmable upper and lower limit values.

The LM77 implements 3 limits: low (temp1_min), high (temp1_max) and critical (temp1_crit.) It also implements an hysteresis mechanism which applies to all 3 limits. The relative difference is stored in a single register on the chip, which means that the relative difference between the limit and its hysteresis is always the same for all 3 limits.

This implementation detail implies the following:

- When setting a limit, its hysteresis will automatically follow, the difference staying unchanged. For example, if the old critical limit was 80 degrees C, and the hysteresis was 75 degrees C, and you change the critical limit to 90 degrees C, then the hysteresis will automatically change to 85 degrees C.
- All 3 hysteresis can't be set independently. We decided to make temp1_crit_hyst writable, while temp1_min_hyst and temp1_max_hyst are read-only. Setting temp1_crit_hyst writes the difference between temp1_crit_hyst and temp1_crit into the chip, and the same relative hysteresis applies automatically to the low and high limits.
- The limits should be set before the hysteresis.

7.69 Kernel driver lm78

Supported chips:

- National Semiconductor LM78 / LM78-J

  Prefix: 'lm78'

  Addresses scanned: I2C 0x28 - 0x2f, ISA 0x290 (8 I/O ports)

  Datasheet: Publicly available at the National Semiconductor website

  http://www.national.com/

- National Semiconductor LM79

  Prefix: 'lm79'

  Addresses scanned: I2C 0x28 - 0x2f, ISA 0x290 (8 I/O ports)

  Datasheet: Publicly available at the National Semiconductor website

  http://www.national.com/

Authors:

- Frodo Looijaard <frodol@dds.nl>
- Jean Delvare <jdelvare@suse.de>

7.69.1 Description

This driver implements support for the National Semiconductor LM78, LM78-J and LM79. They are described as 'Microprocessor System Hardware Monitors'.

There is almost no difference between the three supported chips. Functionally, the LM78 and LM78-J are exactly identical. The LM79 has one more VID line, which is used to report the lower voltages newer Pentium processors use. From here on, LM7* means either of these three types.

The LM7* implements one temperature sensor, three fan rotation speed sensors, seven voltage sensors, VID lines, alarms, and some miscellaneous stuff.

Temperatures are measured in degrees Celsius. An alarm is triggered once when the Overtemperature Shutdown limit is crossed; it is triggered again as soon as it drops below the Hysteresis value. A more useful behavior can be found by setting the Hysteresis value to +127 degrees Celsius; in this case, alarms are issued during all the time when the actual temperature is above the Overtemperature Shutdown value. Measurements are guaranteed between -55 and +125 degrees, with a resolution of 1 degree.

Fan rotation speeds are reported in RPM (rotations per minute). An alarm is triggered if the rotation speed has dropped below a programmable limit. Fan readings can be divided by a programmable divider (1, 2, 4 or 8) to give the readings more range or accuracy. Not all RPM values can accurately be represented, so some rounding is done. With a divider of 2, the lowest representable value is around 2600 RPM.

Voltage sensors (also known as IN sensors) report their values in volts. An alarm is triggered if the voltage has crossed a programmable minimum or maximum limit. Note that minimum in this case always means 'closest to zero'; this is important for negative voltage measurements. All voltage inputs can measure voltages between 0 and 4.08 volts, with a resolution of 0.016 volt.

The VID lines encode the core voltage value: the voltage level your processor should work with. This is hardcoded by the mainboard and/or processor itself. It is a value in volts. When it is unconnected, you will often find the value 3.50 V here.

If an alarm triggers, it will remain triggered until the hardware register is read at least once. This means that the cause for the alarm may already have disappeared! Note that in the current implementation, all hardware registers are read whenever any data is read (unless it is less than 1.5 seconds since the last update). This means that you can easily miss once-only alarms.

The LM7* only updates its values each 1.5 seconds; reading it more often will do no harm, but will return 'old' values.

7.70 Kernel driver lm80

Supported chips:

- National Semiconductor LM80

  Prefix: 'lm80'

  Addresses scanned: I2C 0x28 - 0x2f

  Datasheet: Publicly available at the National Semiconductor website

  http://www.national.com/

- National Semiconductor LM96080

  Prefix: 'lm96080'

  Addresses scanned: I2C 0x28 - 0x2f

  Datasheet: Publicly available at the National Semiconductor website

http://www.national.com/

Authors:

- Frodo Looijaard <frodol@dds.nl>,
- Philip Edelbrock <phil@netroedge.com>

7.70.1 Description

This driver implements support for the National Semiconductor LM80. It is described as a 'Serial Interface ACPI-Compatible Microprocessor System Hardware Monitor'. The LM96080 is a more recent incarnation, it is pin and register compatible, with a few additional features not yet supported by the driver.

The LM80 implements one temperature sensor, two fan rotation speed sensors, seven voltage sensors, alarms, and some miscellaneous stuff.

Temperatures are measured in degrees Celsius. There are two sets of limits which operate independently. When the HOT Temperature Limit is crossed, this will cause an alarm that will be reasserted until the temperature drops below the HOT Hysteresis. The Overtemperature Shutdown (OS) limits should work in the same way (but this must be checked; the datasheet is unclear about this). Measurements are guaranteed between -55 and +125 degrees. The current temperature measurement has a resolution of 0.0625 degrees; the limits have a resolution of 1 degree.

Fan rotation speeds are reported in RPM (rotations per minute). An alarm is triggered if the rotation speed has dropped below a programmable limit. Fan readings can be divided by a programmable divider (1, 2, 4 or 8) to give the readings more range or accuracy. Not all RPM values can accurately be represented, so some rounding is done. With a divider of 2, the lowest representable value is around 2600 RPM.

Voltage sensors (also known as IN sensors) report their values in volts. An alarm is triggered if the voltage has crossed a programmable minimum or maximum limit. Note that minimum in this case always means 'closest to zero'; this is important for negative voltage measurements. All voltage inputs can measure voltages between 0 and 2.55 volts, with a resolution of 0.01 volt.

If an alarm triggers, it will remain triggered until the hardware register is read at least once. This means that the cause for the alarm may already have disappeared! Note that in the current implementation, all hardware registers are read whenever any data is read (unless it is less than 2.0 seconds since the last update). This means that you can easily miss once-only alarms.

The LM80 only updates its values each 1.5 seconds; reading it more often will do no harm, but will return 'old' values.

7.71 Kernel driver lm83

Supported chips:

- National Semiconductor LM83

  Prefix: 'lm83'

  Addresses scanned: I2C 0x18 - 0x1a, 0x29 - 0x2b, 0x4c - 0x4e

  Datasheet: Publicly available at the National Semiconductor website

  http://www.national.com/pf/LM/LM83.html

- National Semiconductor LM82

  Addresses scanned: I2C 0x18 - 0x1a, 0x29 - 0x2b, 0x4c - 0x4e

  Datasheet: Publicly available at the National Semiconductor website

  http://www.national.com/pf/LM/LM82.html

Author: Jean Delvare <jdelvare@suse.de>

7.71.1 Description

The LM83 is a digital temperature sensor. It senses its own temperature as well as the temperature of up to three external diodes. The LM82 is a stripped down version of the LM83 that only supports one external diode. Both are compatible with many other devices such as the LM84 and all other ADM1021 clones. The main difference between the LM83 and the LM84 in that the later can only sense the temperature of one external diode.

Using the adm1021 driver for a LM83 should work, but only two temperatures will be reported instead of four.

The LM83 is only found on a handful of motherboards. Both a confirmed list and an unconfirmed list follow. If you can confirm or infirm the fact that any of these motherboards do actually have an LM83, please contact us. Note that the LM90 can easily be misdetected as a LM83.

Confirmed motherboards:

SBS	P014
SBS	PSL09

Unconfirmed motherboards:

Gigabyte	GA-8IK1100
Iwill	MPX2
Soltek	SL-75DRV5

The LM82 is confirmed to have been found on most AMD Geode reference designs and test platforms.

The driver has been successfully tested by Magnus Forsström, who I'd like to thank here. More testers will be of course welcome.

The fact that the LM83 is only scarcely used can be easily explained. Most motherboards come with more than just temperature sensors for health monitoring. They also have voltage and fan rotation speed sensors. This means that temperature-only chips are usually used as secondary chips coupled with another chip such as an IT8705F or similar chip, which provides more features. Since systems usually need three temperature sensors (motherboard, processor, power supply) and primary chips provide some temperature sensors, the secondary chip, if needed, won't have to handle more than two temperatures. Thus, ADM1021 clones are sufficient, and there is no need for a four temperatures sensor chip such as the LM83. The only case where using an LM83 would make sense is on SMP systems, such as the above-mentioned Iwill MPX2, because you want an additional temperature sensor for each additional CPU.

On the SBS P014, this is different, since the LM83 is the only hardware monitoring chipset. One temperature sensor is used for the motherboard (actually measuring the LM83's own temperature), one is used for the CPU. The two other sensors must be used to measure the temperature of two other points of the motherboard. We suspect these points to be the north and south bridges, but this couldn't be confirmed.

All temperature values are given in degrees Celsius. Local temperature is given within a range of 0 to +85 degrees. Remote temperatures are given within a range of 0 to +125 degrees. Resolution is 1.0 degree, accuracy is guaranteed to 3.0 degrees (see the datasheet for more details).

Each sensor has its own high limit, but the critical limit is common to all four sensors. There is no hysteresis mechanism as found on most recent temperature sensors.

The lm83 driver will not update its values more frequently than every other second; reading them more often will do no harm, but will return 'old' values.

7.72 Kernel driver lm85

Supported chips:

- National Semiconductor LM85 (B and C versions)

  Prefix: 'lm85b' or 'lm85c'

  Addresses scanned: I2C 0x2c, 0x2d, 0x2e

  Datasheet: http://www.national.com/pf/LM/LM85.html

- Texas Instruments LM96000

  Prefix: 'lm9600'

  Addresses scanned: I2C 0x2c, 0x2d, 0x2e

  Datasheet: http://www.ti.com/lit/ds/symlink/lm96000.pdf

- Analog Devices ADM1027

Prefix: ‘adm1027’

Addresses scanned: I2C 0x2c, 0x2d, 0x2e

Datasheet: http://www.onsemi.com/PowerSolutions/product.do?id=ADM1027

- Analog Devices ADT7463

  Prefix: ‘adt7463’

  Addresses scanned: I2C 0x2c, 0x2d, 0x2e

  Datasheet: http://www.onsemi.com/PowerSolutions/product.do?id=ADT7463

- Analog Devices ADT7468

  Prefix: ‘adt7468’

  Addresses scanned: I2C 0x2c, 0x2d, 0x2e

  Datasheet: http://www.onsemi.com/PowerSolutions/product.do?id=ADT7468

- SMSC EMC6D100, SMSC EMC6D101

  Prefix: ‘emc6d100’

  Addresses scanned: I2C 0x2c, 0x2d, 0x2e

  Datasheet: http://www.smsc.com/media/Downloads_Public/discontinued/6d100.pdf

- SMSC EMC6D102

  Prefix: ‘emc6d102’

  Addresses scanned: I2C 0x2c, 0x2d, 0x2e

  Datasheet: http://www.smsc.com/main/catalog/emc6d102.html

- SMSC EMC6D103

  Prefix: ‘emc6d103’

  Addresses scanned: I2C 0x2c, 0x2d, 0x2e

  Datasheet: http://www.smsc.com/main/catalog/emc6d103.html

- SMSC EMC6D103S

  Prefix: ‘emc6d103s’

  Addresses scanned: I2C 0x2c, 0x2d, 0x2e

  Datasheet: http://www.smsc.com/main/catalog/emc6d103s.html

Authors:

- Philip Pokorny <ppokorny@penguincomputing.com>,
- Frodo Looijaard <frodol@dds.nl>,
- Richard Barrington <rich_b_nz@clear.net.nz>,
- Margit Schubert-While <margitsw@t-online.de>,

- Justin Thiessen <jthiessen@penguincomputing.com>

7.72.1 Description

This driver implements support for the National Semiconductor LM85 and compatible chips including the Analog Devices ADM1027, ADT7463, ADT7468 and SMSC EMC6D10x chips family.

The LM85 uses the 2-wire interface compatible with the SMBUS 2.0 specification. Using an analog to digital converter it measures three (3) temperatures and five (5) voltages. It has four (4) 16-bit counters for measuring fan speed. Five (5) digital inputs are provided for sampling the VID signals from the processor to the VRM. Lastly, there are three (3) PWM outputs that can be used to control fan speed.

The voltage inputs have internal scaling resistors so that the following voltage can be measured without external resistors:

2.5V, 3.3V, 5V, 12V, and CPU core voltage (2.25V)

The temperatures measured are one internal diode, and two remote diodes. Remote 1 is generally the CPU temperature. These inputs are designed to measure a thermal diode like the one in a Pentium 4 processor in a socket 423 or socket 478 package. They can also measure temperature using a transistor like the 2N3904.

A sophisticated control system for the PWM outputs is designed into the LM85 that allows fan speed to be adjusted automatically based on any of the three temperature sensors. Each PWM output is individually adjustable and programmable. Once configured, the LM85 will adjust the PWM outputs in response to the measured temperatures without further host intervention. This feature can also be disabled for manual control of the PWM’ s.

Each of the measured inputs (voltage, temperature, fan speed) has corresponding high/low limit values. The LM85 will signal an ALARM if any measured value exceeds either limit.

The LM85 samples all inputs continuously. The lm85 driver will not read the registers more often than once a second. Further, configuration data is only read once each 5 minutes. There is twice as much config data as measurements, so this would seem to be a worthwhile optimization.

7.72.2 Special Features

The LM85 has four fan speed monitoring modes. The ADM1027 has only two. Both have special circuitry to compensate for PWM interactions with the TACH signal from the fans. The ADM1027 can be configured to measure the speed of a two wire fan, but the input conditioning circuitry is different for 3-wire and 2-wire mode. For this reason, the 2-wire fan modes are not exposed to user control. The BIOS should initialize them to the correct mode. If you’ve designed your own ADM1027, you’ ll have to modify the init_client function and add an insmod parameter to set this up.

To smooth the response of fans to changes in temperature, the LM85 has an optional filter for smoothing temperatures. The ADM1027 has the same config option but uses it to rate limit the changes to fan speed instead.

The ADM1027, ADT7463 and ADT7468 have a 10-bit ADC and can therefore measure temperatures with 0.25 degC resolution. They also provide an offset to the temperature readings that is automatically applied during measurement. This offset can be used to zero out any errors due to traces and placement. The documentation says that the offset is in 0.25 degC steps, but in initial testing of the ADM1027 it was 1.00 degC steps. Analog Devices has confirmed this "bug". The ADT7463 is reported to work as described in the documentation. The current lm85 driver does not show the offset register.

The ADT7468 has a high-frequency PWM mode, where all PWM outputs are driven by a 22.5 kHz clock. This is a global mode, not per-PWM output, which means that setting any PWM frequency above 11.3 kHz will switch all 3 PWM outputs to a 22.5 kHz frequency. Conversely, setting any PWM frequency below 11.3 kHz will switch all 3 PWM outputs to a frequency between 10 and 100 Hz, which can then be tuned separately.

See the vendor datasheets for more information. There is application note from National (AN-1260) with some additional information about the LM85. The Analog Devices datasheet is very detailed and describes a procedure for determining an optimal configuration for the automatic PWM control.

The SMSC EMC6D100 & EMC6D101 monitor external voltages, temperatures, and fan speeds. They use this monitoring capability to alert the system to out of limit conditions and can automatically control the speeds of multiple fans in a PC or embedded system. The EMC6D101, available in a 24-pin SSOP package, and the EMC6D100, available in a 28-pin SSOP package, are designed to be register compatible. The EMC6D100 offers all the features of the EMC6D101 plus additional voltage monitoring and system control features. Unfortunately it is not possible to distinguish between the package versions on register level so these additional voltage inputs may read zero. EMC6D102 and EMC6D103 feature additional ADC bits thus extending precision of voltage and temperature channels.

SMSC EMC6D103S is similar to EMC6D103, but does not support pwm#_auto_pwm_minctl and temp#_auto_temp_off.

The LM96000 supports additional high frequency PWM modes (22.5 kHz, 24 kHz, 25.7 kHz, 27.7 kHz and 30 kHz), which can be configured on a per-PWM basis.

7.72.3 Hardware Configurations

The LM85 can be jumpered for 3 different SMBus addresses. There are no other hardware configuration options for the LM85.

The lm85 driver detects both LM85B and LM85C revisions of the chip. See the datasheet for a complete description of the differences. Other than identifying the chip, the driver behaves no differently with regard to these two chips. The LM85B is recommended for new designs.

The ADM1027, ADT7463 and ADT7468 chips have an optional SMBALERT output that can be used to signal the chipset in case a limit is exceeded or the temperature sensors fail. Individual sensor interrupts can be masked so they won' t trigger SMBALERT. The SMBALERT output if configured replaces one of the other functions (PWM2 or IN0). This functionality is not implemented in current driver.

The ADT7463 and ADT7468 also have an optional THERM output/input which can be connected to the processor PROC_HOT output. If available, the autofan control dynamic Tmin feature can be enabled to keep the system temperature within spec (just?!) with the least possible fan noise.

7.72.4 Configuration Notes

Besides standard interfaces driver adds following:

- Temperatures and Zones

Each temperature sensor is associated with a Zone. There are three sensors and therefore three zones (# 1, 2 and 3). Each zone has the following temperature configuration points:

- **temp#_auto_temp_off**
  - temperature below which fans should be off or spinning very low.
- **temp#_auto_temp_min**
  - temperature over which fans start to spin.
- **temp#_auto_temp_max**
  - temperature when fans spin at full speed.
- **temp#_auto_temp_crit**
  - temperature when all fans will run full speed.

PWM Control

There are three PWM outputs. The LM85 datasheet suggests that the pwm3 output control both fan3 and fan4. Each PWM can be individually configured and assigned to a zone for its control value. Each PWM can be configured individually according to the following options.

- **pwm#_auto_pwm_min**
  - this specifies the PWM value for temp#_auto_temp_off temperature. (PWM value from 0 to 255)
- **pwm#_auto_pwm_minctl**
  - this flags selects for temp#_auto_temp_off temperature the behaviour of fans. Write 1 to let fans spinning at pwm#_auto_pwm_min or write 0 to let them off.

Note: It has been reported that there is a bug in the LM85 that causes the flag to be associated with the zones not the PWMs. This contradicts all the published documentation. Setting pwm#_min_ctl in this case actually affects all PWMs controlled by zone '#'.

PWM Controlling Zone selection

- **pwm#_auto_channels**
  - controls zone that is associated with PWM

Configuration choices:

Value	Meaning
1	Controlled by Zone 1
2	Controlled by Zone 2
3	Controlled by Zone 3
23	Controlled by higher temp of Zone 2 or 3
123	Controlled by highest temp of Zone 1, 2 or 3
0	PWM always 0% (off)
-1	PWM always 100% (full on)
-2	Manual control (write to 'pwm#' to set)

The National LM85's have two vendor specific configuration features. Tach. mode and Spinup Control. For more details on these, see the LM85 datasheet or Application Note AN-1260. These features are not currently supported by the lm85 driver.

The Analog Devices ADM1027 has several vendor specific enhancements. The number of pulses-per-rev of the fans can be set, Tach monitoring can be optimized for PWM operation, and an offset can be applied to the temperatures to compensate for systemic errors in the measurements. These features are not currently supported by the lm85 driver.

In addition to the ADM1027 features, the ADT7463 and ADT7468 also have Tmin control and THERM asserted counts. Automatic Tmin control acts to adjust the Tmin value to maintain the measured temperature sensor at a specified temperature. There isn't much documentation on this feature in the ADT7463 data sheet. This is not supported by current driver.

7.73 Kernel driver lm87

Supported chips:

- National Semiconductor LM87

  Prefix: 'lm87'

  Addresses scanned: I2C 0x2c - 0x2e

  Datasheet: http://www.national.com/pf/LM/LM87.html

- Analog Devices ADM1024

  Prefix: 'adm1024'

  Addresses scanned: I2C 0x2c - 0x2e

  Datasheet: http://www.analog.com/en/prod/0,2877,ADM1024,00.html

Authors:

- Frodo Looijaard <frodol@dds.nl>,
- Philip Edelbrock <phil@netroedge.com>,
- Mark Studebaker <mdsxyz123@yahoo.com>,
- Stephen Rousset <stephen.rousset@rocketlogix.com>,
- Dan Eaton <dan.eaton@rocketlogix.com>,
- Jean Delvare <jdelvare@suse.de>,
- Original 2.6 port Jeff Oliver

7.73.1 Description

This driver implements support for the National Semiconductor LM87 and the Analog Devices ADM1024.

The LM87 implements up to three temperature sensors, up to two fan rotation speed sensors, up to seven voltage sensors, alarms, and some miscellaneous stuff. The ADM1024 is fully compatible.

Temperatures are measured in degrees Celsius. Each input has a high and low alarm settings. A high limit produces an alarm when the value goes above it, and an alarm is also produced when the value goes below the low limit.

Fan rotation speeds are reported in RPM (rotations per minute). An alarm is triggered if the rotation speed has dropped below a programmable limit. Fan readings can be divided by a programmable divider (1, 2, 4 or 8) to give the readings more range or accuracy. Not all RPM values can accurately be represented, so some rounding is done. With a divider of 2, the lowest representable value is around 2600 RPM.

Voltage sensors (also known as IN sensors) report their values in volts. An alarm is triggered if the voltage has crossed a programmable minimum or maximum limit. Note that minimum in this case always means ‘closest to zero’; this is important for negative voltage measurements.

If an alarm triggers, it will remain triggered until the hardware register is read at least once. This means that the cause for the alarm may already have disappeared! Note that in the current implementation, all hardware registers are read whenever any data is read (unless it is less than 1.0 seconds since the last update). This means that you can easily miss once-only alarms.

The lm87 driver only updates its values each 1.0 seconds; reading it more often will do no harm, but will return ‘old’ values.

7.73.2 Hardware Configurations

The LM87 has four pins which can serve one of two possible functions, depending on the hardware configuration.

Some functions share pins, so not all functions are available at the same time. Which are depends on the hardware setup. This driver normally assumes that firmware configured the chip correctly. Where this is not the case, platform code must set the I2C client's platform_data to point to a u8 value to be written to the channel register.

For reference, here is the list of exclusive functions:

- in0+in5 (default) or temp3
- fan1 (default) or in6
- fan2 (default) or in7
- VID lines (default) or IRQ lines (not handled by this driver)

7.74 Kernel driver lm90

Supported chips:

- National Semiconductor LM90

  Prefix: 'lm90'

  Addresses scanned: I2C 0x4c

  Datasheet: Publicly available at the National Semiconductor website

  http://www.national.com/pf/LM/LM90.html

- National Semiconductor LM89

  Prefix: 'lm89' (no auto-detection)

  Addresses scanned: I2C 0x4c and 0x4d

  Datasheet: Publicly available at the National Semiconductor website

  http://www.national.com/mpf/LM/LM89.html

- National Semiconductor LM99

  Prefix: 'lm99'

  Addresses scanned: I2C 0x4c and 0x4d

  Datasheet: Publicly available at the National Semiconductor website

  http://www.national.com/pf/LM/LM99.html

- National Semiconductor LM86

  Prefix: 'lm86'

  Addresses scanned: I2C 0x4c

  Datasheet: Publicly available at the National Semiconductor website

http://www.national.com/mpf/LM/LM86.html

- Analog Devices ADM1032

  Prefix: ‘adm1032’

  Addresses scanned: I2C 0x4c and 0x4d

  Datasheet: Publicly available at the ON Semiconductor website

  http://www.onsemi.com/PowerSolutions/product.do?id=ADM1032

- Analog Devices ADT7461

  Prefix: ‘adt7461’

  Addresses scanned: I2C 0x4c and 0x4d

  Datasheet: Publicly available at the ON Semiconductor website

  http://www.onsemi.com/PowerSolutions/product.do?id=ADT7461

- Analog Devices ADT7461A

  Prefix: ‘adt7461a’

  Addresses scanned: I2C 0x4c and 0x4d

  Datasheet: Publicly available at the ON Semiconductor website

  http://www.onsemi.com/PowerSolutions/product.do?id=ADT7461A

- ON Semiconductor NCT1008

  Prefix: ‘nct1008’

  Addresses scanned: I2C 0x4c and 0x4d

  Datasheet: Publicly available at the ON Semiconductor website

  http://www.onsemi.com/PowerSolutions/product.do?id=NCT1008

- Maxim MAX6646

  Prefix: ‘max6646’

  Addresses scanned: I2C 0x4d

  Datasheet: Publicly available at the Maxim website

  http://www.maxim-ic.com/quick_view2.cfm/qv_pk/3497

- Maxim MAX6647

  Prefix: ‘max6646’

  Addresses scanned: I2C 0x4e

  Datasheet: Publicly available at the Maxim website

  http://www.maxim-ic.com/quick_view2.cfm/qv_pk/3497

- Maxim MAX6648

  Prefix: ‘max6646’

  Addresses scanned: I2C 0x4c

Datasheet: Publicly available at the Maxim website

http://www.maxim-ic.com/quick_view2.cfm/qv_pk/3500

- Maxim MAX6649

  Prefix: 'max6646'

  Addresses scanned: I2C 0x4c

  Datasheet: Publicly available at the Maxim website

  http://www.maxim-ic.com/quick_view2.cfm/qv_pk/3497

- Maxim MAX6654

  Prefix: 'max6654'

  Addresses scanned: I2C 0x18, 0x19, 0x1a, 0x29, 0x2a, 0x2b,

  0x4c, 0x4d and 0x4e

  Datasheet: Publicly available at the Maxim website

  https://www.maximintegrated.com/en/products/sensors/MAX6654.html

- Maxim MAX6657

  Prefix: 'max6657'

  Addresses scanned: I2C 0x4c

  Datasheet: Publicly available at the Maxim website

  http://www.maxim-ic.com/quick_view2.cfm/qv_pk/2578

- Maxim MAX6658

  Prefix: 'max6657'

  Addresses scanned: I2C 0x4c

  Datasheet: Publicly available at the Maxim website

  http://www.maxim-ic.com/quick_view2.cfm/qv_pk/2578

- Maxim MAX6659

  Prefix: 'max6659'

  Addresses scanned: I2C 0x4c, 0x4d, 0x4e

  Datasheet: Publicly available at the Maxim website

  http://www.maxim-ic.com/quick_view2.cfm/qv_pk/2578

- Maxim MAX6680

  Prefix: 'max6680'

  Addresses scanned: I2C 0x18, 0x19, 0x1a, 0x29, 0x2a, 0x2b,

  0x4c, 0x4d and 0x4e

  Datasheet: Publicly available at the Maxim website

  http://www.maxim-ic.com/quick_view2.cfm/qv_pk/3370

- Maxim MAX6681

  Prefix: ‘max6680’

  Addresses scanned: I2C 0x18, 0x19, 0x1a, 0x29, 0x2a, 0x2b, 0x4c, 0x4d and 0x4e

  Datasheet: Publicly available at the Maxim website http://www.maxim-ic.com/quick_view2.cfm/qv_pk/3370

- Maxim MAX6692

  Prefix: ‘max6646’

  Addresses scanned: I2C 0x4c

  Datasheet: Publicly available at the Maxim website http://www.maxim-ic.com/quick_view2.cfm/qv_pk/3500

- Maxim MAX6695

  Prefix: ‘max6695’

  Addresses scanned: I2C 0x18

  Datasheet: Publicly available at the Maxim website http://www.maxim-ic.com/datasheet/index.mvp/id/4199

- Maxim MAX6696

  Prefix: ‘max6695’

  Addresses scanned: I2C 0x18, 0x19, 0x1a, 0x29, 0x2a, 0x2b, 0x4c, 0x4d and 0x4e

  Datasheet: Publicly available at the Maxim website http://www.maxim-ic.com/datasheet/index.mvp/id/4199

- Winbond/Nuvoton W83L771W/G

  Prefix: ‘w83l771’

  Addresses scanned: I2C 0x4c

  Datasheet: No longer available

- Winbond/Nuvoton W83L771AWG/ASG

  Prefix: ‘w83l771’

  Addresses scanned: I2C 0x4c

  Datasheet: Not publicly available, can be requested from Nuvoton

- Philips/NXP SA56004X

  Prefix: ‘sa56004’

  Addresses scanned: I2C 0x48 through 0x4F

  Datasheet: Publicly available at NXP website

http://ics.nxp.com/products/interface/datasheet/sa56004x.pdf

- GMT G781

  Prefix: 'g781'

  Addresses scanned: I2C 0x4c, 0x4d

  Datasheet: Not publicly available from GMT

- Texas Instruments TMP451

  Prefix: 'tmp451'

  Addresses scanned: I2C 0x4c

  Datasheet: Publicly available at TI website

  http://www.ti.com/litv/pdf/sbos686

Author: Jean Delvare <jdelvare@suse.de>

7.74.1 Description

The LM90 is a digital temperature sensor. It senses its own temperature as well as the temperature of up to one external diode. It is compatible with many other devices, many of which are supported by this driver.

Note that there is no easy way to differentiate between the MAX6657, MAX6658 and MAX6659 variants. The extra features of the MAX6659 are only supported by this driver if the chip is located at address 0x4d or 0x4e, or if the chip type is explicitly selected as max6659. The MAX6680 and MAX6681 only differ in their pinout, therefore they obviously can't (and don't need to) be distinguished.

The specificity of this family of chipsets over the ADM1021/LM84 family is that it features critical limits with hysteresis, and an increased resolution of the remote temperature measurement.

The different chipsets of the family are not strictly identical, although very similar. For reference, here comes a non-exhaustive list of specific features:

LM90:

- Filter and alert configuration register at 0xBF.
- ALERT is triggered by temperatures over critical limits.

LM86 and LM89:

- Same as LM90
- Better external channel accuracy

LM99:

- Same as LM89
- External temperature shifted by 16 degrees down

ADM1032:

- Consecutive alert register at 0x22.

- Conversion averaging.
- Up to 64 conversions/s.
- ALERT is triggered by open remote sensor.
- SMBus PEC support for Write Byte and Receive Byte transactions.

ADT7461, ADT7461A, NCT1008:

- Extended temperature range (breaks compatibility)
- Lower resolution for remote temperature

MAX6654:

- Better local resolution
- Selectable address
- Remote sensor type selection
- Extended temperature range
- Extended resolution only available when conversion rate <= 1 Hz

MAX6657 and MAX6658:

- Better local resolution
- Remote sensor type selection

MAX6659:

- Better local resolution
- Selectable address
- Second critical temperature limit
- Remote sensor type selection

MAX6680 and MAX6681:

- Selectable address
- Remote sensor type selection

MAX6695 and MAX6696:

- Better local resolution
- Selectable address (max6696)
- Second critical temperature limit
- Two remote sensors

W83L771W/G

- The G variant is lead-free, otherwise similar to the W.
- Filter and alert configuration register at 0xBF
- Moving average (depending on conversion rate)

W83L771AWG/ASG

- Successor of the W83L771W/G, same features.
- The AWG and ASG variants only differ in package format.
- Diode ideality factor configuration (remote sensor) at 0xE3

SA56004X:

- Better local resolution

All temperature values are given in degrees Celsius. Resolution is 1.0 degree for the local temperature, 0.125 degree for the remote temperature, except for the MAX6654, MAX6657, MAX6658 and MAX6659 which have a resolution of 0.125 degree for both temperatures.

Each sensor has its own high and low limits, plus a critical limit. Additionally, there is a relative hysteresis value common to both critical values. To make life easier to user-space applications, two absolute values are exported, one for each channel, but these values are of course linked. Only the local hysteresis can be set from user-space, and the same delta applies to the remote hysteresis.

The lm90 driver will not update its values more frequently than configured with the update_interval attribute; reading them more often will do no harm, but will return 'old' values.

7.74.2 SMBus Alert Support

This driver has basic support for SMBus alert. When an alert is received, the status register is read and the faulty temperature channel is logged.

The Analog Devices chips (ADM1032, ADT7461 and ADT7461A) and ON Semiconductor chips (NCT1008) do not implement the SMBus alert protocol properly so additional care is needed: the ALERT output is disabled when an alert is received, and is re-enabled only when the alarm is gone. Otherwise the chip would block alerts from other chips in the bus as long as the alarm is active.

7.74.3 PEC Support

The ADM1032 is the only chip of the family which supports PEC. It does not support PEC on all transactions though, so some care must be taken.

When reading a register value, the PEC byte is computed and sent by the ADM1032 chip. However, in the case of a combined transaction (SMBus Read Byte), the ADM1032 computes the CRC value over only the second half of the message rather than its entirety, because it thinks the first half of the message belongs to a different transaction. As a result, the CRC value differs from what the SMBus master expects, and all reads fail.

For this reason, the lm90 driver will enable PEC for the ADM1032 only if the bus supports the SMBus Send Byte and Receive Byte transaction types. These transactions will be used to read register values, instead of SMBus Read Byte, and PEC will work properly.

Additionally, the ADM1032 doesn't support SMBus Send Byte with PEC. Instead, it will try to write the PEC value to the register (because the SMBus Send Byte trans-

action with PEC is similar to a Write Byte transaction without PEC), which is not what we want. Thus, PEC is explicitly disabled on SMBus Send Byte transactions in the lm90 driver.

PEC on byte data transactions represents a significant increase in bandwidth usage (+33% for writes, +25% for reads) in normal conditions. With the need to use two SMBus transaction for reads, this overhead jumps to +50%. Worse, two transactions will typically mean twice as much delay waiting for transaction completion, effectively doubling the register cache refresh time. I guess reliability comes at a price, but it' s quite expensive this time.

So, as not everyone might enjoy the slowdown, PEC can be disabled through sysfs. Just write 0 to the "pec" file and PEC will be disabled. Write 1 to that file to enable PEC again.

7.75 Kernel driver lm92

Supported chips:

- National Semiconductor LM92

  Prefix: 'lm92'

  Addresses scanned: I2C 0x48 - 0x4b

  Datasheet: http://www.national.com/pf/LM/LM92.html

- National Semiconductor LM76

  Prefix: 'lm92'

  Addresses scanned: none, force parameter needed

  Datasheet: http://www.national.com/pf/LM/LM76.html

- Maxim MAX6633/MAX6634/MAX6635

  Prefix: 'max6635'

  Addresses scanned: none, force parameter needed

  Datasheet: http://www.maxim-ic.com/quick_view2.cfm/qv_pk/3074

Authors:

- Abraham van der Merwe <abraham@2d3d.co.za>
- Jean Delvare <jdelvare@suse.de>

7.75.1 Description

This driver implements support for the National Semiconductor LM92 temperature sensor.

Each LM92 temperature sensor supports a single temperature sensor. There are alarms for high, low, and critical thresholds. There's also an hysteresis to control the thresholds for resetting alarms.

Support was added later for the LM76 and Maxim MAX6633/MAX6634/MAX6635, which are mostly compatible. They have not all been tested, so you may need to use the force parameter.

7.76 Kernel driver lm93

Supported chips:

- National Semiconductor LM93

  Prefix 'lm93'

  Addresses scanned: I2C 0x2c-0x2e

  Datasheet: http://www.national.com/ds.cgi/LM/LM93.pdf

- National Semiconductor LM94

  Prefix 'lm94'

  Addresses scanned: I2C 0x2c-0x2e

  Datasheet: http://www.national.com/ds.cgi/LM/LM94.pdf

Authors:

- Mark M. Hoffman <mhoffman@lightlink.com>
- Ported to 2.6 by Eric J. Bowersox <ericb@aspsys.com>
- Adapted to 2.6.20 by Carsten Emde <ce@osadl.org>
- Modified for mainline integration by Hans J. Koch <hjk@hansjkoch.de>

7.76.1 Module Parameters

- init: integer Set to non-zero to force some initializations (default is 0).
- disable_block: integer A "0" allows SMBus block data transactions if the host supports them. A "1" disables SMBus block data transactions. The default is 0.
- vccp_limit_type: integer array (2) Configures in7 and in8 limit type, where 0 means absolute and non-zero means relative. "Relative" here refers to "Dynamic Vccp Monitoring using VID"from the datasheet. It greatly simplifies the interface to allow only one set of limits (absolute or relative) to be in operation at a time (even though the hardware is capable of enabling both). There's not a compelling use case for enabling both at once, anyway. The default is "0,0".

- vid_agtl: integer A “0” configures the VID pins for V(ih) = 2.1V min, V(il) = 0.8V max. A “1” configures the VID pins for V(ih) = 0.8V min, V(il) = 0.4V max. (The latter setting is referred to as AGTL+ Compatible in the datasheet.) I.e. this parameter controls the VID pin input thresholds; if your VID inputs are not working, try changing this. The default value is “0”.

7.76.2 Hardware Description

(from the datasheet)

The LM93 hardware monitor has a two wire digital interface compatible with SMBus 2.0. Using an 8-bit ADC, the LM93 measures the temperature of two remote diode connected transistors as well as its own die and 16 power supply voltages. To set fan speed, the LM93 has two PWM outputs that are each controlled by up to four temperature zones. The fancontrol algorithm is lookup table based. The LM93 includes a digital filter that can be invoked to smooth temperature readings for better control of fan speed. The LM93 has four tachometer inputs to measure fan speed. Limit and status registers for all measured values are included. The LM93 builds upon the functionality of previous motherboard management ASICs and uses some of the LM85's features (i.e. smart tachometer mode). It also adds measurement and control support for dynamic Vccp monitoring and PROCHOT. It is designed to monitor a dual processor Xeon class motherboard with a minimum of external components.

LM94 is also supported in LM93 compatible mode. Extra sensors and features of LM94 are not supported.

7.76.3 User Interface

#PROCHOT

The LM93 can monitor two #PROCHOT signals. The results are found in the sysfs files prochot1, prochot2, prochot1_avg, prochot2_avg, prochot1_max, and prochot2_max. prochot1_max and prochot2_max contain the user limits for #PROCHOT1 and #PROCHOT2, respectively. prochot1 and prochot2 contain the current readings for the most recent complete time interval. The value of prochot1_avg and prochot2_avg is something like a 2 period exponential moving average (but not quite - check the datasheet). Note that this third value is calculated by the chip itself. All values range from 0-255 where 0 indicates no throttling, and 255 indicates > 99.6%.

The monitoring intervals for the two #PROCHOT signals is also configurable. These intervals can be found in the sysfs files prochot1_interval and prochot2_interval. The values in these files specify the intervals for #P1_PROCHOT and #P2_PROCHOT, respectively. Selecting a value not in this list will cause the driver to use the next largest interval. The available intervals are (in seconds):

#PROCHOT intervals: 0.73, 1.46, 2.9, 5.8, 11.7, 23.3, 46.6, 93.2, 186, 372

It is possible to configure the LM93 to logically short the two #PROCHOT signals. I.e. when #P1_PROCHOT is asserted, the LM93 will automatically assert

#P2_PROCHOT, and vice-versa. This mode is enabled by writing a non-zero integer to the sysfs file prochot_short.

The LM93 can also override the #PROCHOT pins by driving a PWM signal onto one or both of them. When overridden, the signal has a period of 3.56 ms, a minimum pulse width of 5 clocks (at 22.5kHz => 6.25% duty cycle), and a maximum pulse width of 80 clocks (at 22.5kHz => 99.88% duty cycle).

The sysfs files prochot1_override and prochot2_override contain boolean integers which enable or disable the override function for #P1_PROCHOT and #P2_PROCHOT, respectively. The sysfs file prochot_override_duty_cycle contains a value controlling the duty cycle for the PWM signal used when the override function is enabled. This value ranges from 0 to 15, with 0 indicating minimum duty cycle and 15 indicating maximum.

#VRD_HOT

The LM93 can monitor two #VRD_HOT signals. The results are found in the sysfs files vrdhot1 and vrdhot2. There is one value per file: a boolean for which 1 indicates #VRD_HOT is asserted and 0 indicates it is negated. These files are read-only.

Smart Tach Mode (from the datasheet):

```
If a fan is driven using a low-side drive PWM, the tachometer
output of the fan is corrupted. The LM93 includes smart tachometer
circuitry that allows an accurate tachometer reading to be
achieved despite the signal corruption.  In smart tach mode all
four signals are measured within 4 seconds.
```

Smart tach mode is enabled by the driver by writing 1 or 2 (associating the the fan tachometer with a pwm) to the sysfs file fan<n>_smart_tach. A zero will disable the function for that fan. Note that Smart tach mode cannot be enabled if the PWM output frequency is 22500 Hz (see below).

Manual PWM

The LM93 has a fixed or override mode for the two PWM outputs (although, there are still some conditions that will override even this mode - see section 15.10.6 of the datasheet for details.) The sysfs files pwm1_override and pwm2_override are used to enable this mode; each is a boolean integer where 0 disables and 1 enables the manual control mode. The sysfs files pwm1 and pwm2 are used to set the manual duty cycle; each is an integer (0-255) where 0 is 0% duty cycle, and 255 is 100%. Note that the duty cycle values are constrained by the hardware. Selecting a value which is not available will cause the driver to use the next largest value. Also note: when manual PWM mode is disabled, the value of pwm1 and pwm2 indicates the current duty cycle chosen by the h/w.

PWM Output Frequency

The LM93 supports several different frequencies for the PWM output channels. The sysfs files pwm1_freq and pwm2_freq are used to select the frequency. The frequency values are constrained by the hardware. Selecting a value which is not available will cause the driver to use the next largest value. Also note that this parameter has implications for the Smart Tach Mode (see above).

PWM Output Frequencies (in Hz): 12, 36, 48, 60, 72, 84, 96, 22500 (default)

Automatic PWM

The LM93 is capable of complex automatic fan control, with many different points of configuration. To start, each PWM output can be bound to any combination of eight control sources. The final PWM is the largest of all individual control sources to which the PWM output is bound.

The eight control sources are: temp1-temp4 (aka “zones” in the datasheet), #PROCHOT 1 & 2, and #VRDHOT 1 & 2. The bindings are expressed as a bitmask in the sysfs files pwm<n>_auto_channels, where a “1” enables the binding, and a “0” disables it. The h/w default is 0x0f (all temperatures bound).

0x01	Temp 1
0x02	Temp 2
0x04	Temp 3
0x08	Temp 4
0x10	#PROCHOT 1
0x20	#PROCHOT 2
0x40	#VRDHOT 1
0x80	#VRDHOT 2

The function y = f(x) takes a source temperature x to a PWM output y. This function of the LM93 is derived from a base temperature and a table of 12 temperature offsets. The base temperature is expressed in degrees C in the sysfs files temp<n>_auto_base. The offsets are expressed in cumulative degrees C, with the value of offset <i> for temperature value <n> being contained in the file temp<n>_auto_offset<i>. E.g. if the base temperature is 40C:

offset #	temp<n>_auto_offset<i>	range	pwm
1	0	•	25.00%
2	0	•	28.57%
3	1	40C - 41C	32.14%
4	1	41C - 42C	35.71%
5	2	42C - 44C	39.29%
6	2	44C - 46C	42.86%
7	2	48C - 50C	46.43%
8	2	50C - 52C	50.00%
9	2	52C - 54C	53.57%
10	2	54C - 56C	57.14%
11	2	56C - 58C	71.43%
12	2	58C - 60C	85.71%
•	•	> 60C	100.00%

Valid offsets are in the range 0C <= x <= 7.5C in 0.5C increments.

There is an independent base temperature for each temperature channel. Note, however, there are only two tables of offsets: one each for temp[12] and temp[34]. Therefore, any change to e.g. temp1_auto_offset<i> will also affect temp2_auto_offset<i>.

The LM93 can also apply hysteresis to the offset table, to prevent unwanted oscillation between two steps in the offsets table. These values are found in the sysfs files temp<n>_auto_offset_hyst. The value in this file has the same representation as in temp<n>_auto_offset<i>.

If a temperature reading falls below the base value for that channel, the LM93 will use the minimum PWM value. These values are found in the sysfs files temp<n>_auto_pwm_min. Note, there are only two minimums: one each for temp[12] and temp[34]. Therefore, any change to e.g. temp1_auto_pwm_min will also affect temp2_auto_pwm_min.

PWM Spin-Up Cycle

A spin-up cycle occurs when a PWM output is commanded from 0% duty cycle to some value > 0%. The LM93 supports a minimum duty cycle during spin-up. These values are found in the sysfs files pwm<n>_auto_spinup_min. The value in this file has the same representation as other PWM duty cycle values. The duration of the spin-up cycle is also configurable. These values are found in the sysfs files pwm<n>_auto_spinup_time. The value in this file is the spin-up time in seconds. The available spin-up times are constrained by the hardware. Selecting a value which is not available will cause the driver to use the next largest value.

Spin-up Durations: 0 (disabled, h/w default), 0.1, 0.25, 0.4, 0.7, 1.0, 2.0, 4.0

#PROCHOT and #VRDHOT PWM Ramping

If the #PROCHOT or #VRDHOT signals are asserted while bound to a PWM output channel, the LM93 will ramp the PWM output up to 100% duty cycle in discrete steps. The duration of each step is configurable. There are two files, with one value each in seconds: pwm_auto_prochot_ramp and pwm_auto_vrdhot_ramp. The available ramp times are constrained by the hardware. Selecting a value which is not available will cause the driver to use the next largest value.

Ramp Times: 0 (disabled, h/w default) to 0.75 in 0.05 second intervals

Fan Boost

For each temperature channel, there is a boost temperature: if the channel exceeds this limit, the LM93 will immediately drive both PWM outputs to 100%. This limit is expressed in degrees C in the sysfs files temp<n>_auto_boost. There is also a hysteresis temperature for this function: after the boost limit is reached, the temperature channel must drop below this value before the boost function is disabled. This temperature is also expressed in degrees C in the sysfs files temp<n>_auto_boost_hyst.

GPIO Pins

The LM93 can monitor the logic level of four dedicated GPIO pins as well as the four tach input pins. GPIO0-GPIO3 correspond to (fan) tach 1-4, respectively. All eight GPIOs are read by reading the bitmask in the sysfs file gpio. The LSB is GPIO0, and the MSB is GPIO7.

7.76.4 LM93 Unique sysfs Files

file	description
prochot<n>	current #PROCHOT %
prochot<n>_avg	moving average #PROCHOT %
prochot<n>_max	limit #PROCHOT %
prochot_short	enable or disable logical #PROCHOT pin short
prochot<n>_override	force #PROCHOT assertion as PWM
prochot_override_duty_cycle	duty cycle for the PWM signal used when #PROCHOT is overridden
prochot<n>_interval	#PROCHOT PWM sampling interval
vrdhot<n>	0 means negated, 1 means asserted
fan<n>_smart_tach	enable or disable smart tach mode
pwm<n>_auto_channels	select control sources for PWM outputs
pwm<n>_auto_spinup_min	minimum duty cycle during spin-up
pwm<n>_auto_spinup_time	duration of spin-up
pwm_auto_prochot_ramp	ramp time per step when #PROCHOT asserted
pwm_auto_vrdhot_ramp	ramp time per step when #VRDHOT asserted
temp<n>_auto_base	temperature channel base
temp<n>_auto_offset[1-12]	temperature channel offsets
temp<n>_auto_offset_hyst	temperature channel offset hysteresis
temp<n>_auto_boost	temperature channel boost (PWMs to 100%) limit
temp<n>_auto_boost_hyst	temperature channel boost hysteresis
gpio	input state of 8 GPIO pins; read-only

7.77 Kernel driver lm95234

Supported chips:

- National Semiconductor / Texas Instruments LM95233

  Addresses scanned: I2C 0x18, 0x2a, 0x2b

  Datasheet: Publicly available at the Texas Instruments website

  http://www.ti.com/product/lm95233

- National Semiconductor / Texas Instruments LM95234

  Addresses scanned: I2C 0x18, 0x4d, 0x4e

  Datasheet: Publicly available at the Texas Instruments website

  http://www.ti.com/product/lm95234

Author: Guenter Roeck <linux@roeck-us.net>

7.77.1 Description

LM95233 and LM95234 are 11-bit digital temperature sensors with a 2-wire System Management Bus (SMBus) interface and TrueTherm technology that can very accurately monitor the temperature of two (LM95233) or four (LM95234) remote diodes as well as its own temperature. The remote diodes can be external devices such as microprocessors, graphics processors or diode-connected 2N3904s. The chip' s TruTherm beta compensation technology allows sensing of 90 nm or 65 nm process thermal diodes accurately.

All temperature values are given in millidegrees Celsius. Temperature is provided within a range of -127 to +255 degrees (+127.875 degrees for the internal sensor). Resolution depends on temperature input and range.

Each sensor has its own maximum limit, but the hysteresis is common to all channels. The hysteresis is configurable with the tem1_max_hyst attribute and affects the hysteresis on all channels. The first two external sensors also have a critical limit.

The lm95234 driver can change its update interval to a fixed set of values. It will round up to the next selectable interval. See the datasheet for exact values. Reading sensor values more often will do no harm, but will return 'old' values.

7.78 Kernel driver lm95245

Supported chips:

- TI LM95235

  Addresses scanned: I2C 0x18, 0x29, 0x4c

  Datasheet: Publicly available at the TI website

  http://www.ti.com/lit/ds/symlink/lm95235.pdf

- TI / National Semiconductor LM95245

  Addresses scanned: I2C 0x18, 0x19, 0x29, 0x4c, 0x4d

  Datasheet: Publicly available at the TI website

  http://www.ti.com/lit/ds/symlink/lm95245.pdf

Author: Alexander Stein <alexander.stein@systec-electronic.com>

7.78.1 Description

LM95235 and LM95245 are 11-bit digital temperature sensors with a 2-wire System Management Bus (SMBus) interface and TruTherm technology that can monitor the temperature of a remote diode as well as its own temperature. The chips can be used to very accurately monitor the temperature of external devices such as microprocessors.

All temperature values are given in millidegrees Celsius. Local temperature is given within a range of -127 to +127.875 degrees. Remote temperatures are given

within a range of -127 to +255 degrees. Resolution depends on temperature input and range.

Each sensor has its own critical limit. Additionally, there is a relative hysteresis value common to both critical limits. To make life easier to user-space applications, two absolute values are exported, one for each channel, but these values are of course linked. Only the local hysteresis can be set from user-space, and the same delta applies to the remote hysteresis.

The lm95245 driver can change its update interval to a fixed set of values. It will round up to the next selectable interval. See the datasheet for exact values. Reading sensor values more often will do no harm, but will return 'old' values.

7.79 Kernel Driver Lochnagar

Supported systems:

- Cirrus Logic : Lochnagar 2

Author: Lucas A. Tanure Alves

7.79.1 Description

Lochnagar 2 features built-in Current Monitor circuitry that allows for the measurement of both voltage and current on up to eight of the supply voltage rails provided to the minicards. The Current Monitor does not require any hardware modifications or external circuitry to operate.

The current and voltage measurements are obtained through the standard register map interface to the Lochnagar board controller, and can therefore be monitored by software.

7.79.2 Sysfs attributes

temp1_input	The Lochnagar board temperature (milliCelsius)
in0_input	Measured voltage for DBVDD1 (milliVolts)
in0_label	"DBVDD1"
curr1_input	Measured current for DBVDD1 (milliAmps)
curr1_label	"DBVDD1"
power1_average	Measured average power for DBVDD1 (microWatts)
power1_average_interval	Power averaging time input valid from 1 to 1708mS
power1_label	"DBVDD1"
in1_input	Measured voltage for 1V8 DSP (milliVolts)
in1_label	"1V8 DSP"
curr2_input	Measured current for 1V8 DSP (milliAmps)
curr2_label	"1V8 DSP"
power2_average	Measured average power for 1V8 DSP (microWatts)
power2_average_interval	Power averaging time input valid from 1 to 1708mS
power2_label	"1V8 DSP"

Continued on next page

Table 9 – continued from previous page

in2_input	Measured voltage for 1V8 CDC (milliVolts)
in2_label	“1V8 CDC”
curr3_input	Measured current for 1V8 CDC (milliAmps)
curr3_label	“1V8 CDC”
power3_average	Measured average power for 1V8 CDC (microWatts)
power3_average_interval	Power averaging time input valid from 1 to 1708mS
power3_label	“1V8 CDC”
in3_input	Measured voltage for VDDCORE DSP (milliVolts)
in3_label	“VDDCORE DSP”
curr4_input	Measured current for VDDCORE DSP (milliAmps)
curr4_label	“VDDCORE DSP”
power4_average	Measured average power for VDDCORE DSP (microWatts)
power4_average_interval	Power averaging time input valid from 1 to 1708mS
power4_label	“VDDCORE DSP”
in4_input	Measured voltage for AVDD 1V8 (milliVolts)
in4_label	“AVDD 1V8”
curr5_input	Measured current for AVDD 1V8 (milliAmps)
curr5_label	“AVDD 1V8”
power5_average	Measured average power for AVDD 1V8 (microWatts)
power5_average_interval	Power averaging time input valid from 1 to 1708mS
power5_label	“AVDD 1V8”
curr6_input	Measured current for SYSVDD (milliAmps)
curr6_label	“SYSVDD”
power6_average	Measured average power for SYSVDD (microWatts)
power6_average_interval	Power averaging time input valid from 1 to 1708mS
power6_label	“SYSVDD”
in6_input	Measured voltage for VDDCORE CDC (milliVolts)
in6_label	“VDDCORE CDC”
curr7_input	Measured current for VDDCORE CDC (milliAmps)
curr7_label	“VDDCORE CDC”
power7_average	Measured average power for VDDCORE CDC (microWatts)
power7_average_interval	Power averaging time input valid from 1 to 1708mS
power7_label	“VDDCORE CDC”
in7_input	Measured voltage for MICVDD (milliVolts)
in7_label	“MICVDD”
curr8_input	Measured current for MICVDD (milliAmps)
curr8_label	“MICVDD”
power8_average	Measured average power for MICVDD (microWatts)
power8_average_interval	Power averaging time input valid from 1 to 1708mS
power8_label	“MICVDD”

Note: It is not possible to measure voltage on the SYSVDD rail.

7.80 Kernel driver ltc2945

Supported chips:

- Linear Technology LTC2945

  Prefix: 'ltc2945'

  Addresses scanned: -

  Datasheet:

  http://cds.linear.com/docs/en/datasheet/2945fa.pdf

Author: Guenter Roeck <linux@roeck-us.net>

7.80.1 Description

The LTC2945 is a rail-to-rail system monitor that measures current, voltage, and power consumption.

7.80.2 Usage Notes

This driver does not probe for LTC2945 devices, since there is no register which can be safely used to identify the chip. You will have to instantiate the devices explicitly.

Example: the following will load the driver for an LTC2945 at address 0x10 on I2C bus #1:

```
$ modprobe ltc2945
$ echo ltc2945 0x10 > /sys/bus/i2c/devices/i2c-1/new_device
```

7.80.3 Sysfs entries

Voltage readings provided by this driver are reported as obtained from the ADC registers. If a set of voltage divider resistors is installed, calculate the real voltage by multiplying the reported value with (R1+R2)/R2, where R1 is the value of the divider resistor against the measured voltage and R2 is the value of the divider resistor against Ground.

Current reading provided by this driver is reported as obtained from the ADC Current Sense register. The reported value assumes that a 1 mOhm sense resistor is installed. If a different sense resistor is installed, calculate the real current by dividing the reported value by the sense resistor value in mOhm.

in1_input	VIN voltage (mV). Voltage is measured either at SENSE+ or VDD pin depe
in1_min	Undervoltage threshold
in1_max	Overvoltage threshold
in1_lowest	Lowest measured voltage
in1_highest	Highest measured voltage

Continued on next page

Table 10 – continued from previous page

in1_reset_history	Write 1 to reset in1 history
in1_min_alarm	Undervoltage alarm
in1_max_alarm	Overvoltage alarm
in2_input	ADIN voltage (mV)
in2_min	Undervoltage threshold
in2_max	Overvoltage threshold
in2_lowest	Lowest measured voltage
in2_highest	Highest measured voltage
in2_reset_history	Write 1 to reset in2 history
in2_min_alarm	Undervoltage alarm
in2_max_alarm	Overvoltage alarm
curr1_input	SENSE current (mA)
curr1_min	Undercurrent threshold
curr1_max	Overcurrent threshold
curr1_lowest	Lowest measured current
curr1_highest	Highest measured current
curr1_reset_history	Write 1 to reset curr1 history
curr1_min_alarm	Undercurrent alarm
curr1_max_alarm	Overcurrent alarm
power1_input	Power (in uW). Power is calculated based on SENSE+/VDD voltage o
power1_min	Low lower threshold
power1_max	High power threshold
power1_input_lowest	Historical minimum power use
power1_input_highest	Historical maximum power use
power1_reset_history	Write 1 to reset power1 history
power1_min_alarm	Low power alarm
power1_max_alarm	High power alarm

7.81 Kernel drivers ltc2947-i2c and ltc2947-spi

Supported chips:

- Analog Devices LTC2947

  Prefix: 'ltc2947'

  Addresses scanned: -

  Datasheet:

  https://www.analog.com/media/en/technical-documentation/data-sheets/LTC2947.pdf

Author: Nuno Sá <nuno.sa@analog.com>

7.81.1 Description

The LTC2947 is a high precision power and energy monitor that measures current, voltage, power, temperature, charge and energy. The device supports both SPI and I2C depending on the chip configuration. The device also measures accumulated quantities as energy. It has two banks of register' s to read/set energy related values. These banks can be configured independently to have setups like: energy1 accumulates always and enrgy2 only accumulates if current is positive (to check battery charging efficiency for example). The device also supports a GPIO pin that can be configured as output to control a fan as a function of measured temperature. Then, the GPIO becomes active as soon as a temperature reading is higher than a defined threshold. The temp2 channel is used to control this thresholds and to read the respective alarms.

7.81.2 Sysfs entries

The following attributes are supported. Limits are read-write, reset_history is write-only and all the other attributes are read-only.

in0_input	VP-VM voltage (mV).
in0_min	Undervoltage threshold
in0_max	Overvoltage threshold
in0_lowest	Lowest measured voltage
in0_highest	Highest measured voltage
in0_reset_history	Write 1 to reset in1 history
in0_min_alarm	Undervoltage alarm
in0_max_alarm	Overvoltage alarm
in0_label	Channel label (VP-VM)
in1_input	DVCC voltage (mV)
in1_min	Undervoltage threshold
in1_max	Overvoltage threshold
in1_lowest	Lowest measured voltage
in1_highest	Highest measured voltage
in1_reset_history	Write 1 to reset in2 history
in1_min_alarm	Undervoltage alarm
in1_max_alarm	Overvoltage alarm
in1_label	Channel label (DVCC)
curr1_input	IP-IM Sense current (mA)
curr1_min	Undercurrent threshold
curr1_max	Overcurrent threshold
curr1_lowest	Lowest measured current
curr1_highest	Highest measured current
curr1_reset_history	Write 1 to reset curr1 history
curr1_min_alarm	Undercurrent alarm
curr1_max_alarm	Overcurrent alarm
curr1_label	Channel label (IP-IM)
power1_input	Power (in uW)
power1_min	Low power threshold

Continued on next page

Table 11 – continued from previous page

power1_max	High power threshold
power1_input_lowest	Historical minimum power use
power1_input_highest	Historical maximum power use
power1_reset_history	Write 1 to reset power1 history
power1_min_alarm	Low power alarm
power1_max_alarm	High power alarm
power1_label	Channel label (Power)
temp1_input	Chip Temperature (in milliC)
temp1_min	Low temperature threshold
temp1_max	High temperature threshold
temp1_input_lowest	Historical minimum temperature use
temp1_input_highest	Historical maximum temperature use
temp1_reset_history	Write 1 to reset temp1 history
temp1_min_alarm	Low temperature alarm
temp1_max_alarm	High temperature alarm
temp1_label	Channel label (Ambient)
temp2_min	Low temperature threshold for fan control
temp2_max	High temperature threshold for fan control
temp2_min_alarm	Low temperature fan control alarm
temp2_max_alarm	High temperature fan control alarm
temp2_label	Channel label (TEMPFAN)
energy1_input	Measured energy over time (in microJoule)
energy2_input	Measured energy over time (in microJoule)

7.82 Kernel driver ltc2978

Supported chips:

- Linear Technology LTC2972

  Prefix: 'ltc2972'

  Addresses scanned: -

  Datasheet: https://www.analog.com/en/products/ltc2972.html

- Linear Technology LTC2974

  Prefix: 'ltc2974'

  Addresses scanned: -

  Datasheet: https://www.analog.com/en/products/ltc2974

- Linear Technology LTC2975

  Prefix: 'ltc2975'

  Addresses scanned: -

  Datasheet: https://www.analog.com/en/products/ltc2975

- Linear Technology LTC2977

  Prefix: 'ltc2977'

Addresses scanned: -

Datasheet: https://www.analog.com/en/products/ltc2977

- Linear Technology LTC2978, LTC2978A

  Prefix: 'ltc2978'

  Addresses scanned: -

  Datasheet: https://www.analog.com/en/products/ltc2978
  https://www.analog.com/en/products/ltc2978a

- Linear Technology LTC2979

  Prefix: 'ltc2979'

  Addresses scanned: -

  Datasheet: https://www.analog.com/en/products/ltc2979

- Linear Technology LTC2980

  Prefix: 'ltc2980'

  Addresses scanned: -

  Datasheet: https://www.analog.com/en/products/ltc2980

- Linear Technology LTC3880

  Prefix: 'ltc3880'

  Addresses scanned: -

  Datasheet: https://www.analog.com/en/products/ltc3880

- Linear Technology LTC3882

  Prefix: 'ltc3882'

  Addresses scanned: -

  Datasheet: https://www.analog.com/en/products/ltc3882

- Linear Technology LTC3883

  Prefix: 'ltc3883'

  Addresses scanned: -

  Datasheet: https://www.analog.com/en/products/ltc3883

- Linear Technology LTC3884

  Prefix: 'ltc3884'

  Addresses scanned: -

  Datasheet: https://www.analog.com/en/products/ltc3884

- Linear Technology LTC3886

  Prefix: 'ltc3886'

  Addresses scanned: -

Datasheet: https://www.analog.com/en/products/ltc3886

- Linear Technology LTC3887

  Prefix: 'ltc3887'

  Addresses scanned: -

  Datasheet: https://www.analog.com/en/products/ltc3887

- Linear Technology LTC3889

  Prefix: 'ltc3889'

  Addresses scanned: -

  Datasheet: https://www.analog.com/en/products/ltc3889

- Linear Technology LTC7880

  Prefix: 'ltc7880'

  Addresses scanned: -

  Datasheet: https://www.analog.com/en/products/ltc7880

- Linear Technology LTM2987

  Prefix: 'ltm2987'

  Addresses scanned: -

  Datasheet: https://www.analog.com/en/products/ltm2987

- Linear Technology LTM4644

  Prefix: 'ltm4644'

  Addresses scanned: -

  Datasheet: https://www.analog.com/en/products/ltm4644

  - Linear Technology LTM4675

    Prefix: 'ltm4675'

    Addresses scanned: -

    Datasheet: https://www.analog.com/en/products/ltm4675

- Linear Technology LTM4676

  Prefix: 'ltm4676'

  Addresses scanned: -

  Datasheet: https://www.analog.com/en/products/ltm4676

- Linear Technology LTM4677

  Prefix: 'ltm4677'

  Addresses scanned: -

  Datasheet: https://www.analog.com/en/products/ltm4677

- Linear Technology LTM4678

  Prefix: 'ltm4678'

  Addresses scanned: -

  Datasheet: https://www.analog.com/en/products/ltm4678

- Analog Devices LTM4680

  Prefix: 'ltm4680'

  Addresses scanned: -

  Datasheet: http://www.analog.com/ltm4680

- Analog Devices LTM4686

  Prefix: 'ltm4686'

  Addresses scanned: -

  Datasheet: http://www.analog.com/ltm4686

- Analog Devices LTM4700

  Prefix: 'ltm4700'

  Addresses scanned: -

  Datasheet: http://www.analog.com/ltm4700

Author: Guenter Roeck <linux@roeck-us.net>

7.82.1 Description

- LTC2974 and LTC2975 are quad digital power supply managers.
- LTC2978 is an octal power supply monitor.
- LTC2977 is a pin compatible replacement for LTC2978.
- LTC2980 is a 16-channel Power System Manager, consisting of two LTC2977
- in a single die. The chip is instantiated and reported as two separate chips
- on two different I2C bus addresses.
- LTC3880, LTC3882, LTC3886, and LTC3887 are dual output poly-phase step-down
- DC/DC controllers.
- LTC3883 is a single phase step-down DC/DC controller.
- LTM2987 is a 16-channel Power System Manager with two LTC2977 plus
- additional components on a single die. The chip is instantiated and reported
- as two separate chips on two different I2C bus addresses.
- LTM4675 is a dual 9A or single 18A µModule regulator
- LTM4676 is a dual 13A or single 26A uModule regulator.
- LTM4686 is a dual 10A or single 20A uModule regulator.

7.82.2 Usage Notes

This driver does not probe for PMBus devices. You will have to instantiate devices explicitly.

Example: the following commands will load the driver for an LTC2978 at address 0x60 on I2C bus #1:

```
# modprobe ltc2978
# echo ltc2978 0x60 > /sys/bus/i2c/devices/i2c-1/new_device
```

7.82.3 Sysfs attributes

in1_label	“vin”
in1_input	Measured input voltage.
in1_min	Minimum input voltage.
in1_max	Maximum input voltage. LTC2974, LTC2975, LTC2977, LTC2980, LTC2978, LTC2979 and LTM2987 only.
in1_lcrit	Critical minimum input voltage. LTC2972, LTC2974, LTC2975, LTC2977, LTC2980, LTC2978, LTC2979 and LTM2987 only.
in1_crit	Critical maximum input voltage.
in1_min_alarm	Input voltage low alarm.
in1_max_alarm	Input voltage high alarm. LTC2972, LTC2974, LTC2975, LTC2977, LTC2980, LTC2978, LTC2979 and LTM2987 only.
in1_lcrit_alarm	Input voltage critical low alarm. LTC2972, LTC2974, LTC2975, LTC2977, LTC2980, LTC2978, LTC2979 and LTM2987 only.
in1_crit_alarm	Input voltage critical high alarm.
in1_lowest	Lowest input voltage. LTC2972, LTC2974, LTC2975, LTC2977, LTC2980, LTC2978, and LTM2987 only.
in1_highest	Highest input voltage.
in1_reset_history	Reset input voltage history.

Continued on next page

Table 12 – continued from previous page

in[N]_label	"vout[1-8]". • LTC2972: N=2-3 • LTC2974, LTC2975: N=2-5 • LTC2977, LTC2979, LTC2980, LTM2987: N=2-9 • LTC2978: N=2-9 • LTC3880, LTC3882, LTC3884, LTC23886 LTC3887, LTC3889, LTC7880, LTM4644, LTM4675, LTM4676, LTM4677, LTM4678, LTM4680, LTM4700: N=2-3 • LTC3883: N=2
in[N]_input	Measured output voltage.
in[N]_min	Minimum output voltage.
in[N]_max	Maximum output voltage.
in[N]_lcrit	Critical minimum output voltage.
in[N]_crit	Critical maximum output voltage.
in[N]_min_alarm	Output voltage low alarm.
in[N]_max_alarm	Output voltage high alarm.
in[N]_lcrit_alarm	Output voltage critical low alarm.
in[N]_crit_alarm	Output voltage critical high alarm.
in[N]_lowest	Lowest output voltage. LTC2972, LTC2974, LTC2975,and LTC2978 only.
in[N]_highest	Highest output voltage.
in[N]_reset_history	Reset output voltage history.

Continued on next page

Table 12 – continued from previous page

temp[N]_input	Measured temperature. • On LTC2972, temp[1-2] report external temperatures, and temp 3 reports the chip temperature. • On LTC2974 and LTC2975, temp[1-4] report external temperatures, and temp5 reports the chip temperature. • On LTC2977, LTC2979, LTC2980, LTC2978, and LTM2987, only one temperature measurement is supported and reports the chip temperature. • On LTC3880, LTC3882, LTC3886, LTC3887, LTC3889, LTM4664, LTM4675, LTM4676, LTM4677, LTM4678, LTM4680, and LTM4700, temp1 and temp2 report external temperatures, and temp3 reports the chip temperature. • On LTC3883, temp1 reports an external temperature, and temp2 reports the chip temperature.
temp[N]_min	Mimimum temperature. LTC2972, LTC2974, LCT2977, LTM2980, LTC2978, LTC2979, and LTM2987 only.
temp[N]_max	Maximum temperature.
temp[N]_lcrit	Critical low temperature.
temp[N]_crit	Critical high temperature.
temp[N]_min_alarm	Temperature low alarm. LTC2972, LTC2974, LTC2975, LTC2977, LTM2980, LTC2978, LTC2979, and LTM2987 only.
temp[N]_max_alarm	Temperature high alarm.
temp[N]_lcrit_alarm	Temperature critical low alarm.
temp[N]_crit_alarm	Temperature critical high alarm.
temp[N]_lowest	Lowest measured temperature. • LTC2972, LTC2974, LTC2975, LTC2977, LTM2980, LTC2978, LTC2979, and LTM2987 only. • Not supported for chip temperature sensor on LTC2974 and LTC2975.

Continued on next page

Table 12 – continued from previous page

temp[N]_highest	Highest measured temperature. Not supported for chip temperature sensor on LTC2974 and LTC2975.
temp[N]_reset_history	Reset temperature history. Not supported for chip temperature sensor on LTC2974 and LTC2975.
power1_label	“pin”. LTC3883 and LTC3886 only.
power1_input	Measured input power.
power[N]_label	“pout[1-4]”. • LTC2972: N=1-2 • LTC2974, LTC2975: N=1-4 • LTC2977, LTC2979, LTC2980, LTM2987: Not supported • LTC2978: Not supported • LTC3880, LTC3882, LTC3884, LTC3886, LTC3887, LTC3889, LTM4664, LTM4675, LTM4676, LTM4677, LTM4678, LTM4680, LTM4700: N=1-2 • LTC3883: N=2
power[N]_input	Measured output power.
curr1_label	“iin”. LTC3880, LTC3883, LTC3884, LTC3886, LTC3887, LTC3889, LTM4644, LTM4675, LTM4676, LTM4677, LTM4678, LTM4680, and LTM4700 only.
curr1_input	Measured input current.
curr1_max	Maximum input current.
curr1_max_alarm	Input current high alarm.
curr1_highest	Highest input current. LTC3883 and LTC3886 only.
curr1_reset_history	Reset input current history. LTC3883 and LTC3886 only.
curr[N]_label	“iout[1-4]”. • LTC2972: N-1-2 • LTC2974, LTC2975: N=1-4 • LTC2977, LTC2979, LTC2980, LTM2987: not supported • LTC2978: not supported • LTC3880, LTC3882, LTC3884, LTC3886, LTC3887, LTC3889, LTM4664, LTM4675, LTM4676, LTM4677, LTM4678, LTM4680, LTM4700: N=2-3 • LTC3883: N=2
curr[N]_input	Measured output current.

Continued on next page

Table 12 – continued from previous page

curr[N]_max	Maximum output current.
curr[N]_crit	Critical high output current.
curr[N]_lcrit	Critical low output current. LTC2972, LTC2974 and LTC2975 only.
curr[N]_max_alarm	Output current high alarm.
curr[N]_crit_alarm	Output current critical high alarm.
curr[N]_lcrit_alarm	Output current critical low alarm. LTC2972, LTC2974 and LTC2975 only.
curr[N]_lowest	Lowest output current. LTC2972, LTC2974 and LTC2975 only.
curr[N]_highest	Highest output current.
curr[N]_reset_history	Reset output current history.

7.83 Kernel driver ltc2990

Supported chips:

- Linear Technology LTC2990

  Prefix: 'ltc2990'

  Addresses scanned: -

  Datasheet: http://www.linear.com/product/ltc2990

Author:

- Mike Looijmans <mike.looijmans@topic.nl>
- Tom Levens <tom.levens@cern.ch>

7.83.1 Description

LTC2990 is a Quad I2C Voltage, Current and Temperature Monitor. The chip's inputs can measure 4 voltages, or two inputs together (1+2 and 3+4) can be combined to measure a differential voltage, which is typically used to measure current through a series resistor, or a temperature with an external diode.

7.83.2 Usage Notes

This driver does not probe for PMBus devices. You will have to instantiate devices explicitly.

7.83.3 Sysfs attributes

in0_input	Voltage at Vcc pin in millivolt (range 2.5V to 5V)
temp1_input	Internal chip temperature in millidegrees Celsius

A subset of the following attributes are visible, depending on the measurement mode of the chip.

in[1-4]_input	Voltage at V[1-4] pin in millivolt
temp2_input	External temperature sensor TR1 in millidegrees Celsius
temp3_input	External temperature sensor TR2 in millidegrees Celsius
curr1_input	Current in mA across V1-V2 assuming a 1mOhm sense resistor
curr2_input	Current in mA across V3-V4 assuming a 1mOhm sense resistor

The "curr*_input" measurements actually report the voltage drop across the input pins in microvolts. This is equivalent to the current through a 1mOhm sense resistor. Divide the reported value by the actual sense resistor value in mOhm to get the actual value.

7.84 Kernel driver ltc3815

Supported chips:

- Linear Technology LTC3815

  Prefix: 'ltc3815'

  Addresses scanned: -

  Datasheet: http://www.linear.com/product/ltc3815

Author: Guenter Roeck <linux@roeck-us.net>

7.84.1 Description

LTC3815 is a Monolithic Synchronous DC/DC Step-Down Converter.

7.84.2 Usage Notes

This driver does not probe for PMBus devices. You will have to instantiate devices explicitly.

Example: the following commands will load the driver for an LTC3815 at address 0x20 on I2C bus #1:

```
# modprobe ltc3815
# echo ltc3815 0x20 > /sys/bus/i2c/devices/i2c-1/new_device
```

7.84.3 Sysfs attributes

in1_label	“vin”
in1_input	Measured input voltage.
in1_alarm	Input voltage alarm.
in1_highest	Highest input voltage.
in1_reset_history	Reset input voltage history.
in2_label	“vout1”.
in2_input	Measured output voltage.
in2_alarm	Output voltage alarm.
in2_highest	Highest output voltage.
in2_reset_history	Reset output voltage history.
temp1_input	Measured chip temperature.
temp1_alarm	Temperature alarm.
temp1_highest	Highest measured temperature.
temp1_reset_history	Reset temperature history.
curr1_label	“iin”.
curr1_input	Measured input current.
curr1_highest	Highest input current.
curr1_reset_history	Reset input current history.
curr2_label	“iout1”.
curr2_input	Measured output current.
curr2_alarm	Output current alarm.
curr2_highest	Highest output current.
curr2_reset_history	Reset output current history.

7.85 Kernel driver ltc4151

Supported chips:

- Linear Technology LTC4151

  Prefix: ‘ltc4151’

  Addresses scanned: -

  Datasheet:

  http://www.linear.com/docs/Datasheet/4151fc.pdf

Author: Per Dalen <per.dalen@appeartv.com>

7.85.1 Description

The LTC4151 is a High Voltage I2C Current and Voltage Monitor.

7.85.2 Usage Notes

This driver does not probe for LTC4151 devices, since there is no register which can be safely used to identify the chip. You will have to instantiate the devices explicitly.

Example: the following will load the driver for an LTC4151 at address 0x6f on I2C bus #0:

```
# modprobe ltc4151
# echo ltc4151 0x6f > /sys/bus/i2c/devices/i2c-0/new_device
```

7.85.3 Sysfs entries

Voltage readings provided by this driver are reported as obtained from the ADIN and VIN registers.

Current reading provided by this driver is reported as obtained from the Current Sense register. The reported value assumes that a 1 mOhm sense resistor is installed.

in1_input	VDIN voltage (mV)
in2_input	ADIN voltage (mV)
curr1_input	SENSE current (mA)

7.86 Kernel driver ltc4215

Supported chips:

- Linear Technology LTC4215

  Prefix: 'ltc4215'

  Addresses scanned: 0x44

  Datasheet:

  http://www.linear.com/pc/downloadDocument.do?navId=H0,C1,C1003,C1006,C1163,P17572,D12697

Author: Ira W. Snyder <iws@ovro.caltech.edu>

7.86.1 Description

The LTC4215 controller allows a board to be safely inserted and removed from a live backplane.

7.86.2 Usage Notes

This driver does not probe for LTC4215 devices, due to the fact that some of the possible addresses are unfriendly to probing. You will have to instantiate the devices explicitly.

Example: the following will load the driver for an LTC4215 at address 0x44 on I2C bus #0:

```
$ modprobe ltc4215
$ echo ltc4215 0x44 > /sys/bus/i2c/devices/i2c-0/new_device
```

7.86.3 Sysfs entries

The LTC4215 has built-in limits for overvoltage, undervoltage, and undercurrent warnings. This makes it very likely that the reference circuit will be used.

in1_input	input voltage
in2_input	output voltage
in1_min_alarm	input undervoltage alarm
in1_max_alarm	input overvoltage alarm
curr1_input	current
curr1_max_alarm	overcurrent alarm
power1_input	power usage
power1_alarm	power bad alarm

7.87 Kernel driver ltc4245

Supported chips:

- Linear Technology LTC4245

  Prefix: 'ltc4245'

  Addresses scanned: 0x20-0x3f

  Datasheet:

  http://www.linear.com/pc/downloadDocument.do?navId=H0,C1,C1003,C1006,C1140,P19392,D13517

Author: Ira W. Snyder <iws@ovro.caltech.edu>

7.87.1 Description

The LTC4245 controller allows a board to be safely inserted and removed from a live backplane in multiple supply systems such as CompactPCI and PCI Express.

7.87.2 Usage Notes

This driver does not probe for LTC4245 devices, due to the fact that some of the possible addresses are unfriendly to probing. You will have to instantiate the devices explicitly.

Example: the following will load the driver for an LTC4245 at address 0x23 on I2C bus #1:

```
$ modprobe ltc4245
$ echo ltc4245 0x23 > /sys/bus/i2c/devices/i2c-1/new_device
```

7.87.3 Sysfs entries

The LTC4245 has built-in limits for over and under current warnings. This makes it very likely that the reference circuit will be used.

This driver uses the values in the datasheet to change the register values into the values specified in the sysfs-interface document. The current readings rely on the sense resistors listed in Table 2: “Sense Resistor Values”.

in1_input	12v input voltage (mV)
in2_input	5v input voltage (mV)
in3_input	3v input voltage (mV)
in4_input	Vee (-12v) input voltage (mV)
in1_min_alarm	12v input undervoltage alarm
in2_min_alarm	5v input undervoltage alarm
in3_min_alarm	3v input undervoltage alarm
in4_min_alarm	Vee (-12v) input undervoltage alarm
curr1_input	12v current (mA)
curr2_input	5v current (mA)
curr3_input	3v current (mA)
curr4_input	Vee (-12v) current (mA)
curr1_max_alarm	12v overcurrent alarm
curr2_max_alarm	5v overcurrent alarm
curr3_max_alarm	3v overcurrent alarm
curr4_max_alarm	Vee (-12v) overcurrent alarm
in5_input	12v output voltage (mV)
in6_input	5v output voltage (mV)
in7_input	3v output voltage (mV)
in8_input	Vee (-12v) output voltage (mV)
in5_min_alarm	12v output undervoltage alarm
in6_min_alarm	5v output undervoltage alarm
in7_min_alarm	3v output undervoltage alarm

Continued on next page

Table 13 – continued from previous page

in8_min_alarm	Vee (-12v) output undervoltage alarm
in9_input	GPIO voltage data (see note 1)
in10_input	GPIO voltage data (see note 1)
in11_input	GPIO voltage data (see note 1)
power1_input	12v power usage (mW)
power2_input	5v power usage (mW)
power3_input	3v power usage (mW)
power4_input	Vee (-12v) power usage (mW)

7.87.4 Note 1

If you have NOT configured the driver to sample all GPIO pins as analog voltages, then the in10_input and in11_input sysfs attributes will not be created. The driver will sample the GPIO pin that is currently connected to the ADC as an analog voltage, and report the value in in9_input.

If you have configured the driver to sample all GPIO pins as analog voltages, then they will be sampled in round-robin fashion. If userspace reads too slowly, -EAGAIN will be returned when you read the sysfs attribute containing the sensor reading.

The LTC4245 chip can be configured to sample all GPIO pins with two methods:

1) platform data – see include/linux/platform_data/ltc4245.h
2) OF device tree – add the "ltc4245,use-extra-gpios" property to each chip

The default mode of operation is to sample a single GPIO pin.

7.88 Kernel driver ltc4260

Supported chips:

- Linear Technology LTC4260

  Prefix: 'ltc4260'

  Addresses scanned: -

  Datasheet:

  http://cds.linear.com/docs/en/datasheet/4260fc.pdf

Author: Guenter Roeck <linux@roeck-us.net>

7.88.1 Description

The LTC4260 Hot Swap controller allows a board to be safely inserted and removed from a live backplane.

7.88.2 Usage Notes

This driver does not probe for LTC4260 devices, since there is no register which can be safely used to identify the chip. You will have to instantiate the devices explicitly.

Example: the following will load the driver for an LTC4260 at address 0x10 on I2C bus #1:

```
$ modprobe ltc4260
$ echo ltc4260 0x10 > /sys/bus/i2c/devices/i2c-1/new_device
```

7.88.3 Sysfs entries

Voltage readings provided by this driver are reported as obtained from the ADC registers. If a set of voltage divider resistors is installed, calculate the real voltage by multiplying the reported value with (R1+R2)/R2, where R1 is the value of the divider resistor against the measured voltage and R2 is the value of the divider resistor against Ground.

Current reading provided by this driver is reported as obtained from the ADC Current Sense register. The reported value assumes that a 1 mOhm sense resistor is installed. If a different sense resistor is installed, calculate the real current by dividing the reported value by the sense resistor value in mOhm.

in1_input	SOURCE voltage (mV)
in1_min_alarm	Undervoltage alarm
in1_max_alarm	Overvoltage alarm
in2_input	ADIN voltage (mV)
in2_alarm	Power bad alarm
curr1_input	SENSE current (mA)
curr1_alarm	SENSE overcurrent alarm

7.89 Kernel driver ltc4261

Supported chips:

- Linear Technology LTC4261

  Prefix: 'ltc4261'

  Addresses scanned: -

  Datasheet:

  http://cds.linear.com/docs/Datasheet/42612fb.pdf

Author: Guenter Roeck <linux@roeck-us.net>

7.89.1 Description

The LTC4261/LTC4261-2 negative voltage Hot Swap controllers allow a board to be safely inserted and removed from a live backplane.

7.89.2 Usage Notes

This driver does not probe for LTC4261 devices, since there is no register which can be safely used to identify the chip. You will have to instantiate the devices explicitly.

Example: the following will load the driver for an LTC4261 at address 0x10 on I2C bus #1:

```
$ modprobe ltc4261
$ echo ltc4261 0x10 > /sys/bus/i2c/devices/i2c-1/new_device
```

7.89.3 Sysfs entries

Voltage readings provided by this driver are reported as obtained from the ADC registers. If a set of voltage divider resistors is installed, calculate the real voltage by multiplying the reported value with (R1+R2)/R2, where R1 is the value of the divider resistor against the measured voltage and R2 is the value of the divider resistor against Ground.

Current reading provided by this driver is reported as obtained from the ADC Current Sense register. The reported value assumes that a 1 mOhm sense resistor is installed. If a different sense resistor is installed, calculate the real current by dividing the reported value by the sense resistor value in mOhm.

The chip has two voltage sensors, but only one set of voltage alarm status bits. In many many designs, those alarms are associated with the ADIN2 sensor, due to the proximity of the ADIN2 pin to the OV pin. ADIN2 is, however, not available on all chip variants. To ensure that the alarm condition is reported to the user, report it with both voltage sensors.

in1_input	ADIN2 voltage (mV)
in1_min_alarm	ADIN/ADIN2 Undervoltage alarm
in1_max_alarm	ADIN/ADIN2 Overvoltage alarm
in2_input	ADIN voltage (mV)
in2_min_alarm	ADIN/ADIN2 Undervoltage alarm
in2_max_alarm	ADIN/ADIN2 Overvoltage alarm
curr1_input	SENSE current (mA)
curr1_alarm	SENSE overcurrent alarm

7.90 Kernel driver max16064

Supported chips:

- Maxim MAX16064

  Prefix: 'max16064'

  Addresses scanned: -

  Datasheet: http://datasheets.maxim-ic.com/en/ds/MAX16064.pdf

Author: Guenter Roeck <linux@roeck-us.net>

7.90.1 Description

This driver supports hardware monitoring for Maxim MAX16064 Quad Power-Supply Controller with Active-Voltage Output Control and PMBus Interface.

The driver is a client driver to the core PMBus driver. Please see Documentation/hwmon/pmbus.rst for details on PMBus client drivers.

7.90.2 Usage Notes

This driver does not auto-detect devices. You will have to instantiate the devices explicitly. Please see Documentation/i2c/instantiating-devices.rst for details.

7.90.3 Platform data support

The driver supports standard PMBus driver platform data.

7.90.4 Sysfs entries

The following attributes are supported. Limits are read-write; all other attributes are read-only.

in[1-4]_label	“vout[1-4]”
in[1-4]_input	Measured voltage. From READ_VOUT register.
in[1-4]_min	Minimum Voltage. From VOUT_UV_WARN_LIMIT register.
in[1-4]_max	Maximum voltage. From VOUT_OV_WARN_LIMIT register.
in[1-4]_lcrit	Critical minimum Voltage. VOUT_UV_FAULT_LIMIT register.
in[1-4]_crit	Critical maximum voltage. From VOUT_OV_FAULT_LIMIT register.
in[1-4]_min_alarm	Voltage low alarm. From VOLTAGE_UV_WARNING status.
in[1-4]_max_alarm	Voltage high alarm. From VOLTAGE_OV_WARNING status.
in[1-4]_lcrit_alarm	Voltage critical low alarm. From VOLTAGE_UV_FAULT status.
in[1-4]_crit_alarm	Voltage critical high alarm. From VOLTAGE_OV_FAULT status.
in[1-4]_highest	Historical maximum voltage.
in[1-4]_reset_history	Write any value to reset history.
temp1_input	Measured temperature. From READ_TEMPERATURE_1 register.
temp1_max	Maximum temperature. From OT_WARN_LIMIT register.
temp1_crit	Critical high temperature. From OT_FAULT_LIMIT register.
temp1_max_alarm	Chip temperature high alarm. Set by comparing READ_TEMPERATURE_1 with OT_WARN_LIMIT if TEMP_OT_WARNING status is set.
temp1_crit_alarm	Chip temperature critical high alarm. Set by comparing READ_TEMPERATURE_1 with OT_FAULT_LIMIT if TEMP_OT_FAULT status is set.
temp1_highest	Historical maximum temperature.
temp1_reset_history	Write any value to reset history.

7.91 Kernel driver max16065

Supported chips:

- Maxim MAX16065, MAX16066

  Prefixes: ‘max16065’, ‘max16066’

  Addresses scanned: -

  Datasheet:

  http://datasheets.maxim-ic.com/en/ds/MAX16065-MAX16066.pdf

- Maxim MAX16067

  Prefix: ‘max16067’

  Addresses scanned: -

  Datasheet:

  http://datasheets.maxim-ic.com/en/ds/MAX16067.pdf

- Maxim MAX16068

  Prefix: ‘max16068’

  Addresses scanned: -

  Datasheet:

  http://datasheets.maxim-ic.com/en/ds/MAX16068.pdf

- Maxim MAX16070/MAX16071

  Prefixes: ‘max16070’, ‘max16071’

  Addresses scanned: -

  Datasheet:

  http://datasheets.maxim-ic.com/en/ds/MAX16070-MAX16071.pdf

Author: Guenter Roeck <linux@roeck-us.net>

7.91.1 Description

[From datasheets] The MAX16065/MAX16066 flash-configurable system managers monitor and sequence multiple system voltages. The MAX16065/MAX16066 can also accurately monitor (+/-2.5%) one current channel using a dedicated high-side current-sense amplifier. The MAX16065 manages up to twelve system voltages simultaneously, and the MAX16066 manages up to eight supply voltages.

The MAX16067 flash-configurable system manager monitors and sequences multiple system voltages. The MAX16067 manages up to six system voltages simultaneously.

The MAX16068 flash-configurable system manager monitors and manages up to six system voltages simultaneously.

The MAX16070/MAX16071 flash-configurable system monitors supervise multiple system voltages. The MAX16070/MAX16071 can also accurately monitor (+/-2.5%) one current channel using a dedicated high-side current-sense amplifier. The MAX16070 monitors up to twelve system voltages simultaneously, and the MAX16071 monitors up to eight supply voltages.

Each monitored channel has its own low and high critical limits. MAX16065, MAX16066, MAX16070, and MAX16071 support an additional limit which is configurable as either low or high secondary limit. MAX16065, MAX16066, MAX16070, and MAX16071 also support supply current monitoring.

7.91.2 Usage Notes

This driver does not probe for devices, since there is no register which can be safely used to identify the chip. You will have to instantiate the devices explicitly. Please see Documentation/i2c/instantiating-devices.rst for details.

WARNING: Do not access chip registers using the i2cdump command, and do not use any of the i2ctools commands on a command register (0xa5 to 0xac). The chips supported by this driver interpret any access to a command register (including read commands) as request to execute the command in question. This may result in power loss, board resets, and/or Flash corruption. Worst case, your board may turn into a brick.

7.91.3 Sysfs entries

in[0-11]_input	Input voltage measurements.
in12_input	Voltage on CSP (Current Sense Positive) pin. Only if the chip supports current sensing and if current sensing is enabled.
in[0-11]_min	Low warning limit. Supported on MAX16065, MAX16066, MAX16070, and MAX16071 only.
in[0-11]_max	High warning limit. Supported on MAX16065, MAX16066, MAX16070, and MAX16071 only. Either low or high warning limits are supported (depending on chip configuration), but not both.
in[0-11]_lcrit	Low critical limit.
in[0-11]_crit	High critical limit.
in[0-11]_alarm	Input voltage alarm.
curr1_input	Current sense input; only if the chip supports current sensing and if current sensing is enabled. Displayed current assumes 0.001 Ohm current sense resistor.
curr1_alarm	Overcurrent alarm; only if the chip supports current sensing and if current sensing is enabled.

7.92 Kernel driver max1619

Supported chips:

- Maxim MAX1619

  Prefix: ‘max1619’

  Addresses scanned: I2C 0x18-0x1a, 0x29-0x2b, 0x4c-0x4e

  Datasheet: Publicly available at the Maxim website

  http://pdfserv.maxim-ic.com/en/ds/MAX1619.pdf

Authors:

- Oleksij Rempel <bug-track@fisher-privat.net>,
- Jean Delvare <jdelvare@suse.de>

7.92.1 Description

The MAX1619 is a digital temperature sensor. It senses its own temperature as well as the temperature of up to one external diode.

All temperature values are given in degrees Celsius. Resolution is 1.0 degree for the local temperature and for the remote temperature.

Only the external sensor has high and low limits.

The max1619 driver will not update its values more frequently than every other second; reading them more often will do no harm, but will return 'old' values.

7.93 Kernel driver max16601

Supported chips:

- Maxim MAX16601

  Prefix: 'max16601'

  Addresses scanned: -

  Datasheet: Not published

Author: Guenter Roeck <linux@roeck-us.net>

7.93.1 Description

This driver supports the MAX16601 VR13.HC Dual-Output Voltage Regulator Chipset.

The driver is a client driver to the core PMBus driver. Please see Documentation/hwmon/pmbus.rst for details on PMBus client drivers.

7.93.2 Usage Notes

This driver does not auto-detect devices. You will have to instantiate the devices explicitly. Please see Documentation/i2c/instantiating-devices.rst for details.

7.93.3 Platform data support

The driver supports standard PMBus driver platform data.

7.93.4 Sysfs entries

The following attributes are supported.

in1_label	“vin1”
in1_input	VCORE input voltage.
in1_alarm	Input voltage alarm.
in2_label	“vout1”
in2_input	VCORE output voltage.
in2_alarm	Output voltage alarm.
curr1_label	“iin1”
curr1_input	VCORE input current, derived from duty cycle and output current.
curr1_max	Maximum input current.
curr1_max_alarm	Current high alarm.
curr2_label	“iin1.0”
curr2_input	VCORE phase 0 input current.
curr3_label	“iin1.1”
curr3_input	VCORE phase 1 input current.
curr4_label	“iin1.2”
curr4_input	VCORE phase 2 input current.
curr5_label	“iin1.3”
curr5_input	VCORE phase 3 input current.
curr6_label	“iin1.4”
curr6_input	VCORE phase 4 input current.
curr7_label	“iin1.5”
curr7_input	VCORE phase 5 input current.
curr8_label	“iin1.6”
curr8_input	VCORE phase 6 input current.
curr9_label	“iin1.7”
curr9_input	VCORE phase 7 input current.
curr10_label	“iin2”
curr10_input	VCORE input current, derived from sensor element.
curr11_label	“iin3”
curr11_input	VSA input current.
curr12_label	“iout1”
curr12_input	VCORE output current.
curr12_crit	Critical output current.
curr12_crit_alarm	Output current critical alarm.
curr12_max	Maximum output current.
curr12_max_alarm	Output current high alarm.
curr13_label	“iout1.0”
curr13_input	VCORE phase 0 output current.
curr14_label	“iout1.1”
curr14_input	VCORE phase 1 output current.

Continued on next page

Table 14 – continued from previous page

curr15_label	“iout1.2”
curr15_input	VCORE phase 2 output current.
curr16_label	“iout1.3”
curr16_input	VCORE phase 3 output current.
curr17_label	“iout1.4”
curr17_input	VCORE phase 4 output current.
curr18_label	“iout1.5”
curr18_input	VCORE phase 5 output current.
curr19_label	“iout1.6”
curr19_input	VCORE phase 6 output current.
curr20_label	“iout1.7”
curr20_input	VCORE phase 7 output current.
curr21_label	“iout3”
curr21_input	VSA output current.
curr21_highest	Historical maximum VSA output current.
curr21_reset_history	Write any value to reset curr21_highest.
curr21_crit	Critical output current.
curr21_crit_alarm	Output current critical alarm.
curr21_max	Maximum output current.
curr21_max_alarm	Output current high alarm.
power1_label	“pin1”
power1_input	Input power, derived from duty cycle and output current.
power1_alarm	Input power alarm.
power2_label	“pin2”
power2_input	Input power, derived from input current sensor.
power3_label	“pout”
power3_input	Output power.
temp1_input	VCORE temperature.
temp1_crit	Critical high temperature.
temp1_crit_alarm	Chip temperature critical high alarm.
temp1_max	Maximum temperature.
temp1_max_alarm	Chip temperature high alarm.
temp2_input	TSENSE_0 temperature
temp3_input	TSENSE_1 temperature
temp4_input	TSENSE_2 temperature
temp5_input	TSENSE_3 temperature
temp6_input	VSA temperature.
temp6_crit	Critical high temperature.
temp6_crit_alarm	Chip temperature critical high alarm.
temp6_max	Maximum temperature.
temp6_max_alarm	Chip temperature high alarm.

7.94 Kernel driver max1668

Supported chips:

- Maxim MAX1668, MAX1805 and MAX1989

  Prefix: 'max1668'

  Addresses scanned: I2C 0x18, 0x19, 0x1a, 0x29, 0x2a, 0x2b, 0x4c, 0x4d, 0x4e

  Datasheet: http://datasheets.maxim-ic.com/en/ds/MAX1668-MAX1989.pdf

Author:

David George <david.george@ska.ac.za>

7.94.1 Description

This driver implements support for the Maxim MAX1668, MAX1805 and MAX1989 chips.

The three devices are very similar, but the MAX1805 has a reduced feature set; only two remote temperature inputs vs the four available on the other two ICs.

The driver is able to distinguish between the devices and creates sysfs entries as follows:

- MAX1805, MAX1668 and MAX1989:

temp1_input	ro	local (ambient) temperature
temp1_max	rw	local temperature maximum threshold for alarm
temp1_max_alarm	ro	local temperature maximum threshold alarm
temp1_min	rw	local temperature minimum threshold for alarm
temp1_min_alarm	ro	local temperature minimum threshold alarm
temp2_input	ro	remote temperature 1
temp2_max	rw	remote temperature 1 maximum threshold for alarm
temp2_max_alarm	ro	remote temperature 1 maximum threshold alarm
temp2_min	rw	remote temperature 1 minimum threshold for alarm
temp2_min_alarm	ro	remote temperature 1 minimum threshold alarm
temp3_input	ro	remote temperature 2
temp3_max	rw	remote temperature 2 maximum threshold for alarm
temp3_max_alarm	ro	remote temperature 2 maximum threshold alarm
temp3_min	rw	remote temperature 2 minimum threshold for alarm
temp3_min_alarm	ro	remote temperature 2 minimum threshold alarm

- MAX1668 and MAX1989 only:

temp4_input	ro	remote temperature 3
temp4_max	rw	remote temperature 3 maximum threshold for alarm
temp4_max_alarm	ro	remote temperature 3 maximum threshold alarm
temp4_min	rw	remote temperature 3 minimum threshold for alarm
temp4_min_alarm	ro	remote temperature 3 minimum threshold alarm
temp5_input	ro	remote temperature 4
temp5_max	rw	remote temperature 4 maximum threshold for alarm
temp5_max_alarm	ro	remote temperature 4 maximum threshold alarm
temp5_min	rw	remote temperature 4 minimum threshold for alarm
temp5_min_alarm	ro	remote temperature 4 minimum threshold alarm

7.94.2 Module Parameters

- read_only: int Set to non-zero if you wish to prevent write access to alarm thresholds.

7.95 Kernel driver max197

Author:

- Vivien Didelot <vivien.didelot@savoirfairelinux.com>

Supported chips:

- Maxim MAX197

  Prefix: 'max197'

  Datasheet: http://datasheets.maxim-ic.com/en/ds/MAX197.pdf

- Maxim MAX199

  Prefix: 'max199'

  Datasheet: http://datasheets.maxim-ic.com/en/ds/MAX199.pdf

7.95.1 Description

The A/D converters MAX197, and MAX199 are both 8-Channel, Multi-Range, 5V, 12-Bit DAS with 8+4 Bus Interface and Fault Protection.

The available ranges for the MAX197 are {0,-5V} to 5V, and {0,-10V} to 10V, while they are {0,-2V} to 2V, and {0,-4V} to 4V on the MAX199.

7.95.2 Platform data

The MAX197 platform data (defined in linux/platform_data/max197.h) should be filled with a pointer to a conversion function, defined like:

```
int convert(u8 ctrl);
```

ctrl is the control byte to write to start a new conversion. On success, the function must return the 12-bit raw value read from the chip, or a negative error code otherwise.

Control byte format:

Bit	Name	Description
7,6	PD1,PD0	Clock and Power-Down modes
5	ACQMOD	Internal or External Controlled Acquisition
4	RNG	Full-scale voltage magnitude at the input
3	BIP	Unipolar or Bipolar conversion mode
2,1,0	A2,A1,A0	Channel

7.95.3 Sysfs interface

in[0-7]_input	The conversion value for the corresponding channel. RO
in[0-7]_min	The lower limit (in mV) for the corresponding channel. For the MAX197, it will be adjusted to -10000, -5000, or 0. For the MAX199, it will be adjusted to -4000, -2000, or 0. RW
in[0-7]_max	The higher limit (in mV) for the corresponding channel. For the MAX197, it will be adjusted to 0, 5000, or 10000. For the MAX199, it will be adjusted to 0, 2000, or 4000. RW

7.96 Kernel driver max20730

Supported chips:

- Maxim MAX20730

  Prefix: ‘max20730’

  Addresses scanned: -

  Datasheet: https://datasheets.maximintegrated.com/en/ds/MAX20730.pdf

- Maxim MAX20734

  Prefix: ‘max20734’

  Addresses scanned: -

  Datasheet: https://datasheets.maximintegrated.com/en/ds/MAX20734.pdf

- Maxim MAX20743

  Prefix: ‘max20743’

  Addresses scanned: -

  Datasheet: https://datasheets.maximintegrated.com/en/ds/MAX20743.pdf

Author: Guenter Roeck <linux@roeck-us.net>

7.96.1 Description

This driver implements support for Maxim MAX20730, MAX20734, and MAX20743 Integrated, Step-Down Switching Regulators with PMBus support.

The driver is a client driver to the core PMBus driver. Please see Documentation/hwmon/pmbus.rst for details on PMBus client drivers.

7.96.2 Usage Notes

This driver does not auto-detect devices. You will have to instantiate the devices explicitly. Please see Documentation/i2c/instantiating-devices.rst for details.

7.96.3 Sysfs entries

curr1_crit	RW/RO	Critical output current. Please see datasheet for supported limits. Read-only if the chip is write protected; read-write otherwise.
curr1_crit_alarm	RO	Output current critical alarm
curr1_input	RO	Output current
curr1_label	RO	‘iout1’
in1_alarm	RO	Input voltage alarm
in1_input	RO	Input voltage
in1_label	RO	‘vin’
in2_alarm	RO	Output voltage alarm
in2_input	RO	Output voltage
in2_label	RO	‘vout1’
temp1_crit	RW/RO	Critical temeperature. Supported values are 130 or 150 degrees C. Read-only if the chip is write protected; read-write otherwise.
temp1_crit_alarm	RO	Temperature critical alarm
temp1_input	RO	Chip temperature

7.97 Kernel driver max20751

Supported chips:

- maxim MAX20751

  Prefix: 'max20751'

  Addresses scanned: -

  Datasheet: http://datasheets.maximintegrated.com/en/ds/MAX20751.pdf

  Application note: http://pdfserv.maximintegrated.com/en/an/AN5941.pdf

Author: Guenter Roeck <linux@roeck-us.net>

7.97.1 Description

This driver supports MAX20751 Multiphase Master with PMBus Interface and Internal Buck Converter.

The driver is a client driver to the core PMBus driver. Please see Documentation/hwmon/pmbus.rst for details on PMBus client drivers.

7.97.2 Usage Notes

This driver does not auto-detect devices. You will have to instantiate the devices explicitly. Please see Documentation/i2c/instantiating-devices.rst for details.

7.97.3 Platform data support

The driver supports standard PMBus driver platform data.

7.97.4 Sysfs entries

The following attributes are supported.

in1_label	"vin1"
in1_input	Measured voltage.
in1_min	Minimum input voltage.
in1_max	Maximum input voltage.
in1_lcrit	Critical minimum input voltage.
in1_crit	Critical maximum input voltage.
in1_min_alarm	Input voltage low alarm.
in1_lcrit_alarm	Input voltage critical low alarm.
in1_min_alarm	Input voltage low alarm.
in1_max_alarm	Input voltage high alarm.
in2_label	"vout1"
in2_input	Measured voltage.
in2_min	Minimum output voltage.

Continued on next page

Table 15 – continued from previous page

in2_max	Maximum output voltage.
in2_lcrit	Critical minimum output voltage.
in2_crit	Critical maximum output voltage.
in2_min_alarm	Output voltage low alarm.
in2_lcrit_alarm	Output voltage critical low alarm.
in2_min_alarm	Output voltage low alarm.
in2_max_alarm	Output voltage high alarm.
curr1_input	Measured output current.
curr1_label	“iout1”
curr1_max	Maximum output current.
curr1_alarm	Current high alarm.
temp1_input	Measured temperature.
temp1_max	Maximum temperature.
temp1_crit	Critical high temperature.
temp1_max_alarm	Chip temperature high alarm.
temp1_crit_alarm	Chip temperature critical high alarm.
power1_input	Output power.
power1_label	“pout1”

7.98 Kernel driver max31722

Supported chips:

- Maxim Integrated MAX31722

  Prefix: ‘max31722’

  ACPI ID: MAX31722

  Addresses scanned: -

  Datasheet: https://datasheets.maximintegrated.com/en/ds/MAX31722-MAX31723.pdf

- Maxim Integrated MAX31723

  Prefix: ‘max31723’

  ACPI ID: MAX31723

  Addresses scanned: -

  Datasheet: https://datasheets.maximintegrated.com/en/ds/MAX31722-MAX31723.pdf

Author: Tiberiu Breana <tiberiu.a.breana@intel.com>

7.98.1 Description

This driver adds support for the Maxim Integrated MAX31722/MAX31723 thermometers and thermostats running over an SPI interface.

7.98.2 Usage Notes

This driver uses ACPI to auto-detect devices. See ACPI IDs in the above section.

7.98.3 Sysfs entries

The following attribute is supported:

temp1_input	Measured temperature. Read-only.

7.99 Kernel driver max31790

Supported chips:

- Maxim MAX31730

  Prefix: 'max31730'

  Addresses scanned: 0x1c, 0x1d, 0x1e, 0x1f, 0x4c, 0x4d, 0x4e, 0x4f

  Datasheet: https://datasheets.maximintegrated.com/en/ds/MAX31730.pdf

Author: Guenter Roeck <linux@roeck-us.net>

7.99.1 Description

This driver implements support for Maxim MAX31730.

The MAX31730 temperature sensor monitors its own temperature and the temperatures of three external diode-connected transistors. The operating supply voltage is from 3.0V to 3.6V. Resistance cancellation compensates for high series resistance in circuit-board traces and the external thermal diode, while beta compensation corrects for temperature-measurement errors due to low-beta sensing transistors.

7.99.2 Sysfs entries

temp[1-4]_enable	RW	Temperature enable/disable Set to 0 to enable channel, 0 to disable
temp[1-4]_input	RO	Temperature input
temp[2-4]_fault	RO	Fault indicator for remote channels
temp[1-4]_max	RW	Maximum temperature
temp[1-4]_max_alarm	RW	Maximum temperature alarm
temp[1-4]_min	RW	Minimum temperature. Common for all channels. Only temp1_min is writeable.
temp[1-4]_min_alarm	RO	Minimum temperature alarm
temp[2-4]_offset	RW	Temperature offset for remote channels

7.100 Kernel driver max31785

Supported chips:

- Maxim MAX31785, MAX31785A

  Prefix: 'max31785' or 'max31785a'

  Addresses scanned: -

  Datasheet: https://datasheets.maximintegrated.com/en/ds/MAX31785.pdf

Author: Andrew Jeffery <andrew@aj.id.au>

7.100.1 Description

The Maxim MAX31785 is a PMBus device providing closed-loop, multi-channel fan management with temperature and remote voltage sensing. Various fan control features are provided, including PWM frequency control, temperature hysteresis, dual tachometer measurements, and fan health monitoring.

For dual-rotor configurations the MAX31785A exposes the second rotor tachometer readings in attributes fan[5-8]_input. By contrast the MAX31785 only exposes the slowest rotor measurement, and does so in the fan[1-4]_input attributes.

7.100.2 Usage Notes

This driver does not probe for PMBus devices. You will have to instantiate devices explicitly.

7.100.3 Sysfs attributes

fan[1-4]_alarm	Fan alarm.
fan[1-4]_fault	Fan fault.
fan[1-8]_input	Fan RPM. On the MAX31785A, inputs 5-8 correspond to the second rotor of fans 1-4
fan[1-4]_target	Fan input target
in[1-6]_crit	Critical maximum output voltage
in[1-6]_crit_alarm	Output voltage critical high alarm
in[1-6]_input	Measured output voltage
in[1-6]_label	“vout[18-23]”
in[1-6]_lcrit	Critical minimum output voltage
in[1-6]_lcrit_alarm	Output voltage critical low alarm
in[1-6]_max	Maximum output voltage
in[1-6]_max_alarm	Output voltage high alarm
in[1-6]_min	Minimum output voltage
in[1-6]_min_alarm	Output voltage low alarm
pwm[1-4]	Fan target duty cycle (0..255)
pwm[1-4]_enable	0: Full-speed 1: Manual PWM control 2: Automatic PWM (tach-feedback RPM fan-control) 3: Automatic closed-loop (temp-feedback fan-control)
temp[1-11]_crit	Critical high temperature
temp[1-11]_crit_alarm	Chip temperature critical high alarm
temp[1-11]_input	Measured temperature
temp[1-11]_max	Maximum temperature
temp[1-11]_max_alarm	Chip temperature high alarm

7.101 Kernel driver max31790

Supported chips:

- Maxim MAX31790

  Prefix: 'max31790'

  Addresses scanned: -

  Datasheet: http://pdfserv.maximintegrated.com/en/ds/MAX31790.pdf

Author: Il Han <corone.il.han@gmail.com>

7.101.1 Description

This driver implements support for the Maxim MAX31790 chip.

The MAX31790 controls the speeds of up to six fans using six independent PWM outputs. The desired fan speeds (or PWM duty cycles) are written through the I2C interface. The outputs drive "4-wire" fans directly, or can be used to modulate the fan's power terminals using an external pass transistor.

Tachometer inputs monitor fan tachometer logic outputs for precise (+/-1%) monitoring and control of fan RPM as well as detection of fan failure. Six pins are dedicated tachometer inputs. Any of the six PWM outputs can also be configured to serve as tachometer inputs.

7.101.2 Sysfs entries

fan[1-12]_input	RO	fan tachometer speed in RPM
fan[1-12]_fault	RO	fan experienced fault
fan[1-6]_target	RW	desired fan speed in RPM
pwm[1-6]_enable	RW	regulator mode, 0=disabled, 1=manual mode, 2=rpm mode
pwm[1-6]	RW	fan target duty cycle (0-255)

7.102 Kernel driver max34440

Supported chips:

- Maxim MAX34440

  Prefixes: 'max34440'

  Addresses scanned: -

  Datasheet: http://datasheets.maximintegrated.com/en/ds/MAX34440.pdf

- Maxim MAX34441

  PMBus 5-Channel Power-Supply Manager and Intelligent Fan Controller

  Prefixes: 'max34441'

Addresses scanned: -

Datasheet: http://datasheets.maximintegrated.com/en/ds/MAX34441.pdf

- Maxim MAX34446

  PMBus Power-Supply Data Logger

  Prefixes: 'max34446'

  Addresses scanned: -

  Datasheet: http://datasheets.maximintegrated.com/en/ds/MAX34446.pdf

- Maxim MAX34451

  PMBus 16-Channel V/I Monitor and 12-Channel Sequencer/Marginer

  Prefixes: 'max34451'

  Addresses scanned: -

  Datasheet: http://datasheets.maximintegrated.com/en/ds/MAX34451.pdf

- Maxim MAX34460

  PMBus 12-Channel Voltage Monitor & Sequencer

  Prefix: 'max34460'

  Addresses scanned: -

  Datasheet: http://datasheets.maximintegrated.com/en/ds/MAX34460.pdf

- Maxim MAX34461

  PMBus 16-Channel Voltage Monitor & Sequencer

  Prefix: 'max34461'

  Addresses scanned: -

  Datasheet: http://datasheets.maximintegrated.com/en/ds/MAX34461.pdf

Author: Guenter Roeck <linux@roeck-us.net>

7.102.1 Description

This driver supports hardware monitoring for Maxim MAX34440 PMBus 6-Channel Power-Supply Manager, MAX34441 PMBus 5-Channel Power-Supply Manager and Intelligent Fan Controller, and MAX34446 PMBus Power-Supply Data Logger. It also supports the MAX34451, MAX34460, and MAX34461 PMBus Voltage Monitor & Sequencers. The MAX34451 supports monitoring voltage or current of 12 channels based on GIN pins. The MAX34460 supports 12 voltage channels, and the MAX34461 supports 16 voltage channels.

The driver is a client driver to the core PMBus driver. Please see Documentation/hwmon/pmbus.rst for details on PMBus client drivers.

7.102.2 Usage Notes

This driver does not auto-detect devices. You will have to instantiate the devices explicitly. Please see Documentation/i2c/instantiating-devices.rst for details.

For MAX34446, the value of the currX_crit attribute determines if current or voltage measurement is enabled for a given channel. Voltage measurement is enabled if currX_crit is set to 0; current measurement is enabled if the attribute is set to a positive value. Power measurement is only enabled if channel 1 (3) is configured for voltage measurement, and channel 2 (4) is configured for current measurement.

7.102.3 Platform data support

The driver supports standard PMBus driver platform data.

7.102.4 Sysfs entries

The following attributes are supported. Limits are read-write; all other attributes are read-only.

In

in[1-6]_label	“vout[1-6]”.
in[1-6]_input	Measured voltage. From READ_VOUT register.
in[1-6]_min	Minimum Voltage. From VOUT_UV_WARN_LIMIT register.
in[1-6]_max	Maximum voltage. From VOUT_OV_WARN_LIMIT register.
in[1-6]_lcrit	Critical minimum Voltage. VOUT_UV_FAULT_LIMIT register.
in[1-6]_crit	Critical maximum voltage. From VOUT_OV_FAULT_LIMIT register.
in[1-6]_min_alarm	Voltage low alarm. From VOLTAGE_UV_WARNING status.
in[1-6]_max_alarm	Voltage high alarm. From VOLTAGE_OV_WARNING status.
in[1-6]_lcrit_alarm	Voltage critical low alarm. From VOLTAGE_UV_FAULT status.
in[1-6]_crit_alarm	Voltage critical high alarm. From VOLTAGE_OV_FAULT status.
in[1-6]_lowest	Historical minimum voltage.
in[1-6]_highest	Historical maximum voltage.
in[1-6]_reset_history	Write any value to reset history.

Note: MAX34446 only supports in[1-4].

Curr

curr[1-6]_label	“iout[1-6]” .
curr[1-6]_input	Measured current. From READ_IOUT register.
curr[1-6]_max	Maximum current. From IOUT_OC_WARN_LIMIT register.
curr[1-6]_crit	Critical maximum current. From IOUT_OC_FAULT_LIMIT register.
curr[1-6]_max_alarm	Current high alarm. From IOUT_OC_WARNING status.
curr[1-6]_crit_alarm	Current critical high alarm. From IOUT_OC_FAULT status.
curr[1-4]_average	Historical average current (MAX34446/34451 only).
curr[1-6]_highest	Historical maximum current.
curr[1-6]_reset_history	Write any value to reset history.

Note:

- in6 and curr6 attributes only exist for MAX34440.
- MAX34446 only supports curr[1-4].

Power

power[1,3]_label	“pout[1,3]”
power[1,3]_input	Measured power.
power[1,3]_average	Historical average power.
power[1,3]_highest	Historical maximum power.

Note: Power attributes only exist for MAX34446.

Temp

temp[1-8]_input	Measured temperatures. From READ_TEMPERATURE_1 register. temp1 is the chip's internal temperature. temp2..temp5 are remote I2C temperature sensors. For MAX34441, temp6 is a remote thermal-diode sensor. For MAX34440, temp6..8 are remote I2C temperature sensors.
temp[1-8]_max	Maximum temperature. From OT_WARN_LIMIT register.
temp[1-8]_crit	Critical high temperature. From OT_FAULT_LIMIT register.
temp[1-8]_max_alarm	Temperature high alarm.
temp[1-8]_crit_alarm	Temperature critical high alarm.
temp[1-8]_average	Historical average temperature (MAX34446 only).
temp[1-8]_highest	Historical maximum temperature.
temp[1-8]_reset_history	Write any value to reset history.

Note:

- temp7 and temp8 attributes only exist for MAX34440.
- MAX34446 only supports temp[1-3].

Note:

- MAX34451 supports attribute groups in[1-16] (or curr[1-16] based on input pins) and temp[1-5].
- MAX34460 supports attribute groups in[1-12] and temp[1-5].
- MAX34461 supports attribute groups in[1-16] and temp[1-5].

7.103 Kernel driver max6639

Supported chips:

- Maxim MAX6639

  Prefix: 'max6639'

  Addresses scanned: I2C 0x2c, 0x2e, 0x2f

  Datasheet: http://pdfserv.maxim-ic.com/en/ds/MAX6639.pdf

Authors:

- He Changqing <hechangqing@semptian.com>
- Roland Stigge <stigge@antcom.de>

7.103.1 Description

This driver implements support for the Maxim MAX6639. This chip is a 2-channel temperature monitor with dual PWM fan speed controller. It can monitor its own temperature and one external diode-connected transistor or two external diode-connected transistors.

The following device attributes are implemented via sysfs:

Attribute	R/W	Contents
temp1_input	R	Temperature channel 1 input (0..150 C)
temp2_input	R	Temperature channel 2 input (0..150 C)
temp1_fault	R	Temperature channel 1 diode fault
temp2_fault	R	Temperature channel 2 diode fault
temp1_max	RW	Set THERM temperature for input 1 (in C, see datasheet)
temp2_max	RW	Set THERM temperature for input 2
temp1_crit	RW	Set ALERT temperature for input 1
temp2_crit	RW	Set ALERT temperature for input 2
temp1_emergency	RW	Set OT temperature for input 1 (in C, see datasheet)
temp2_emergency	RW	Set OT temperature for input 2
pwm1	RW	Fan 1 target duty cycle (0..255)
pwm2	RW	Fan 2 target duty cycle (0..255)
fan1_input	R	TACH1 fan tachometer input (in RPM)
fan2_input	R	TACH2 fan tachometer input (in RPM)
fan1_fault	R	Fan 1 fault
fan2_fault	R	Fan 2 fault
temp1_max_alarm	R	Alarm on THERM temperature on channel 1
temp2_max_alarm	R	Alarm on THERM temperature on channel 2
temp1_crit_alarm	R	Alarm on ALERT temperature on channel 1
temp2_crit_alarm	R	Alarm on ALERT temperature on channel 2
temp1_emergency_alarm	R	Alarm on OT temperature on channel 1
temp2_emergency_alarm	R	Alarm on OT temperature on channel 2

7.104 Kernel driver max6642

Supported chips:

- Maxim MAX6642

  Prefix: ‘max6642’

  Addresses scanned: I2C 0x48-0x4f

  Datasheet: Publicly available at the Maxim website

  http://datasheets.maxim-ic.com/en/ds/MAX6642.pdf

Authors:

Per Dalen <per.dalen@appeartv.com>

7.104.1 Description

The MAX6642 is a digital temperature sensor. It senses its own temperature as well as the temperature on one external diode.

All temperature values are given in degrees Celsius. Resolution is 0.25 degree for the local temperature and for the remote temperature.

7.105 Kernel driver max6650

Supported chips:

- Maxim MAX6650

  Prefix: 'max6650'

  Addresses scanned: none

  Datasheet: http://pdfserv.maxim-ic.com/en/ds/MAX6650-MAX6651.pdf

- Maxim MAX6651

  Prefix: 'max6651'

  Addresses scanned: none

  Datasheet: http://pdfserv.maxim-ic.com/en/ds/MAX6650-MAX6651.pdf

Authors:

- Hans J. Koch <hjk@hansjkoch.de>
- John Morris <john.morris@spirentcom.com>
- Claus Gindhart <claus.gindhart@kontron.com>

7.105.1 Description

This driver implements support for the Maxim MAX6650 and MAX6651.

The 2 devices are very similar, but the MAX6550 has a reduced feature set, e.g. only one fan-input, instead of 4 for the MAX6651.

The driver is not able to distinguish between the 2 devices.

The driver provides the following sensor accesses in sysfs:

fan1_input	ro	fan tachometer speed in RPM
fan2_input	ro	"
fan3_input	ro	"
fan4_input	ro	"
fan1_target	rw	desired fan speed in RPM (closed loop mode only)
pwm1_enable	rw	regulator mode, 0=full on, 1=open loop, 2=closed loop 3=off
pwm1	rw	relative speed (0-255), 255=max. speed. Used in open loop mode only.
fan1_div	rw	sets the speed range the inputs can handle. Legal values are 1, 2, 4, and 8. Use lower values for faster fans.

7.105.2 Usage notes

This driver does not auto-detect devices. You will have to instantiate the devices explicitly. Please see Documentation/i2c/instantiating-devices.rst for details.

7.105.3 Module parameters

If your board has a BIOS that initializes the MAX6650/6651 correctly, you can simply load your module without parameters. It won't touch the configuration registers then. If your board BIOS doesn't initialize the chip, or you want different settings, you can set the following parameters:

voltage_12V: 5=5V fan, 12=12V fan, 0=don't change prescaler: Possible values are 1,2,4,8,16, or 0 for don't change clock: The clock frequency in Hz of the chip the driver should assume [254000]

Please have a look at the MAX6650/6651 data sheet and make sure that you fully understand the meaning of these parameters before you attempt to change them.

7.106 Kernel driver max6697

Supported chips:

- Maxim MAX6581

  Prefix: 'max6581'

  Datasheet: http://datasheets.maximintegrated.com/en/ds/MAX6581.pdf

- Maxim MAX6602

  Prefix: 'max6602'

  Datasheet: http://datasheets.maximintegrated.com/en/ds/MAX6602.pdf

- Maxim MAX6622

  Prefix: 'max6622'

  Datasheet: http://datasheets.maximintegrated.com/en/ds/MAX6622.pdf

- Maxim MAX6636

  Prefix: ‘max6636’

  Datasheet: http://datasheets.maximintegrated.com/en/ds/MAX6636.pdf

- Maxim MAX6689

  Prefix: ‘max6689’

  Datasheet: http://datasheets.maximintegrated.com/en/ds/MAX6689.pdf

- Maxim MAX6693

  Prefix: ‘max6693’

  Datasheet: http://datasheets.maximintegrated.com/en/ds/MAX6693.pdf

- Maxim MAX6694

  Prefix: ‘max6694’

  Datasheet: http://datasheets.maximintegrated.com/en/ds/MAX6694.pdf

- Maxim MAX6697

  Prefix: ‘max6697’

  Datasheet: http://datasheets.maximintegrated.com/en/ds/MAX6697.pdf

- Maxim MAX6698

  Prefix: ‘max6698’

  Datasheet: http://datasheets.maximintegrated.com/en/ds/MAX6698.pdf

- Maxim MAX6699

  Prefix: ‘max6699’

  Datasheet: http://datasheets.maximintegrated.com/en/ds/MAX6699.pdf

Author:

Guenter Roeck <linux@roeck-us.net>

7.106.1 Description

This driver implements support for several MAX6697 compatible temperature sensor chips. The chips support one local temperature sensor plus four, six, or seven remote temperature sensors. Remote temperature sensors are diode-connected thermal transitors, except for MAX6698 which supports three diode-connected thermal transistors plus three thermistors in addition to the local temperature sensor.

The driver provides the following sysfs attributes. temp1 is the local (chip) temperature, temp[2..n] are remote temperatures. The actually supported per-channel attributes are chip type and channel dependent.

tempX_input	RO	temperature
tempX_max	RW	temperature maximum threshold
tempX_max_alarm	RO	temperature maximum threshold alarm
tempX_crit	RW	temperature critical threshold
tempX_crit_alarm	RO	temperature critical threshold alarm
tempX_fault	RO	temperature diode fault (remote sensors only)

7.107 Kernel driver max8688

Supported chips:

- Maxim MAX8688

  Prefix: 'max8688'

  Addresses scanned: -

  Datasheet: http://datasheets.maxim-ic.com/en/ds/MAX8688.pdf

Author: Guenter Roeck <linux@roeck-us.net>

7.107.1 Description

This driver supports hardware monitoring for Maxim MAX8688 Digital Power-Supply Controller/Monitor with PMBus Interface.

The driver is a client driver to the core PMBus driver. Please see Documentation/hwmon/pmbus.rst for details on PMBus client drivers.

7.107.2 Usage Notes

This driver does not auto-detect devices. You will have to instantiate the devices explicitly. Please see Documentation/i2c/instantiating-devices.rst for details.

7.107.3 Platform data support

The driver supports standard PMBus driver platform data.

7.107.4 Sysfs entries

The following attributes are supported. Limits are read-write; all other attributes are read-only.

in1_label	“vout1”
in1_input	Measured voltage. From READ_VOUT register.
in1_min	Minimum Voltage. From VOUT_UV_WARN_LIMIT register.
in1_max	Maximum voltage. From VOUT_OV_WARN_LIMIT register.
in1_lcrit	Critical minimum Voltage. VOUT_UV_FAULT_LIMIT register.
in1_crit	Critical maximum voltage. From VOUT_OV_FAULT_LIMIT register.
in1_min_alarm	Voltage low alarm. From VOLTAGE_UV_WARNING status.
in1_max_alarm	Voltage high alarm. From VOLTAGE_OV_WARNING status.
in1_lcrit_alarm	Voltage critical low alarm. From VOLTAGE_UV_FAULT status.
in1_crit_alarm	Voltage critical high alarm. From VOLTAGE_OV_FAULT status.
in1_highest	Historical maximum voltage.
in1_reset_history	Write any value to reset history.
curr1_label	“iout1”
curr1_input	Measured current. From READ_IOUT register.
curr1_max	Maximum current. From IOUT_OC_WARN_LIMIT register.
curr1_crit	Critical maximum current. From IOUT_OC_FAULT_LIMIT register.
curr1_max_alarm	Current high alarm. From IOUT_OC_WARN_LIMIT register.
curr1_crit_alarm	Current critical high alarm. From IOUT_OC_FAULT status.
curr1_highest	Historical maximum current.
curr1_reset_history	Write any value to reset history.
temp1_input	Measured temperature. From READ_TEMPERATURE_1 register.
temp1_max	Maximum temperature. From OT_WARN_LIMIT register.
temp1_crit	Critical high temperature. From OT_FAULT_LIMIT register.
temp1_max_alarm	Chip temperature high alarm. Set by comparing READ_TEMPERATURE_1 with OT_WARN_LIMIT if TEMP_OT_WARNING status is set.
temp1_crit_alarm	Chip temperature critical high alarm. Set by comparing READ_TEMPERATURE_1 with OT_FAULT_LIMIT if TEMP_OT_FAULT status is set.
temp1_highest	Historical maximum temperature.
temp1_reset_history	Write any value to reset history.

7.108 Kernel driver mc13783-adc

Supported chips:

- Freescale MC13783

  Prefix: ‘mc13783’

  Datasheet: https://www.nxp.com/docs/en/data-sheet/MC13783.pdf

- Freescale MC13892

  Prefix: ‘mc13892’

  Datasheet: https://www.nxp.com/docs/en/data-sheet/MC13892.pdf

Authors:

- Sascha Hauer <s.hauer@pengutronix.de>
- Luotao Fu <l.fu@pengutronix.de>

7.108.1 Description

The Freescale MC13783 and MC13892 are Power Management and Audio Circuits. Among other things they contain a 10-bit A/D converter. The converter has 16 (MC13783) resp. 12 (MC13892) channels which can be used in different modes. The A/D converter has a resolution of 2.25mV.

Some channels can be used as General Purpose inputs or in a dedicated mode with a chip internal scaling applied .

Currently the driver only supports the Application Supply channel (BP / BPSNS), the General Purpose inputs and touchscreen.

See the following tables for the meaning of the different channels and their chip internal scaling:

- MC13783:

Channel	Signal	Input Range	Scaling
0	Battery Voltage (BATT)	2.50 - 4.65V	-2.40V
1	Battery Current (BATT - BATTISNS)	-50 - 50 mV	x20
2	Application Supply (BP)	2.50 - 4.65V	-2.40V
3	Charger Voltage (CHRGRAW)	0 - 10V / 0 - 20V	/5 /10
4	Charger Current (CHRGISNSP-CHRGISNSN)	-0.25 - 0.25V	x4
5	General Purpose ADIN5 / Battery Pack Thermistor	0 - 2.30V	No
6	General Purpose ADIN6 / Backup Voltage (LICELL)	0 - 2.30V / 1.50 - 3.50V	No / - 1.20V
7	General Purpose ADIN7 / UID / Die Temperature	0 - 2.30V / 0 - 2.55V /	No / x0.9 / No
8	General Purpose ADIN8	0 - 2.30V	No
9	General Purpose ADIN9	0 - 2.30V	No
10	General Purpose ADIN10	0 - 2.30V	No
11	General Purpose ADIN11	0 - 2.30V	No
12	General Purpose TSX1 / Touchscreen X-plate 1	0 - 2.30V	No
13	General Purpose TSX2 / Touchscreen X-plate 2	0 - 2.30V	No
14	General Purpose TSY1 / Touchscreen Y-plate 1	0 - 2.30V	No
15	General Purpose TSY2 / Touchscreen Y-plate 2	0 - 2.30V	No

- MC13892:

Chan-nel	Signal	Input Range	Scal-ing
0	Battery Voltage (BATT)	0 - 4.8V	/2
1	Battery Current (BATT - BATTISNSCC)	-60 - 60 mV	x20
2	Application Supply (BPSNS)	0 - 4.8V	/2
3	Charger Voltage (CHRGRAW)	0 - 12V / 0 - 20V	/5 /10
4	Charger Current (CHRGISNS-BPSNS) / Touch-screen X-plate 1	-0.3 - 0.3V / 0 - 2.4V	x4 / No
5	General Purpose ADIN5 / Battery Pack Thermistor	0 - 2.4V	No
6	General Purpose ADIN6 / Backup Voltage (LI-CELL) Backup Voltage (LICELL)	0 - 2.4V / 0 - 3.6V	No x2/3
7	General Purpose ADIN7 / UID / Die Temperature	0 - 2.4V / 0 - 4.8V	No / /2
12	General Purpose TSX1 / Touchscreen X-plate 1	0 - 2.4V	No
13	General Purpose TSX2 / Touchscreen X-plate 2	0 - 2.4V	No
14	General Purpose TSY1 / Touchscreen Y-plate 1	0 - 2.4V	No
15	General Purpose TSY2 / Touchscreen Y-plate 2	0 - 2.4V	No

7.109 Kernel driver MCP3021

Supported chips:

- Microchip Technology MCP3021

  Prefix: ‘mcp3021’

  Datasheet: http://ww1.microchip.com/downloads/en/DeviceDoc/21805a.pdf

- Microchip Technology MCP3221

  Prefix: ‘mcp3221’

  Datasheet: http://ww1.microchip.com/downloads/en/DeviceDoc/21732c.pdf

Authors:

- Mingkai Hu
- Sven Schuchmann <schuchmann@schleissheimer.de>

7.109.1 Description

This driver implements support for the Microchip Technology MCP3021 and MCP3221 chip.

The Microchip Technology Inc. MCP3021 is a successive approximation A/D converter (ADC) with 10-bit resolution. The MCP3221 has 12-bit resolution.

These devices provide one single-ended input with very low power consumption. Communication to the MCP3021/MCP3221 is performed using a 2-wire I2C compatible interface. Standard (100 kHz) and Fast (400 kHz) I2C modes are avail-

able. The default I2C device address is 0x4d (contact the Microchip factory for additional address options).

7.110 Kernel driver menf21bmc_hwmon

Supported chips:

- MEN 14F021P00

  Prefix: 'menf21bmc_hwmon'

  Adresses scanned: -

Author: Andreas Werner <andreas.werner@men.de>

7.110.1 Description

The menf21bmc is a Board Management Controller (BMC) which provides an I2C interface to the host to access the features implemented in the BMC.

This driver gives access to the voltage monitoring feature of the main voltages of the board. The voltage sensors are connected to the ADC inputs of the BMC which is a PIC16F917 Mikrocontroller.

7.110.2 Usage Notes

This driver is part of the MFD driver named "menf21bmc" and does not auto-detect devices. You will have to instantiate the MFD driver explicitly. Please see Documentation/i2c/instantiating-devices.rst for details.

7.110.3 Sysfs entries

The following attributes are supported. All attributes are read only The Limits are read once by the driver.

in0_input	+3.3V input voltage
in1_input	+5.0V input voltage
in2_input	+12.0V input voltage
in3_input	+5V Standby input voltage
in4_input	VBAT (on board battery)
in[0-4]_min	Minimum voltage limit
in[0-4]_max	Maximum voltage limit
in0_label	"MON_3_3V"
in1_label	"MON_5V"
in2_label	"MON_12V"
in3_label	"5V_STANDBY"
in4_label	"VBAT"

7.111 Kernel driver mlxreg-fan

Provides FAN control for the next Mellanox systems:

- QMB700, equipped with 40x200GbE InfiniBand ports;
- MSN3700, equipped with 32x200GbE or 16x400GbE Ethernet ports;
- MSN3410, equipped with 6x400GbE plus 48x50GbE Ethernet ports;
- MSN3800, equipped with 64x1000GbE Ethernet ports;

Author: Vadim Pasternak <vadimp@mellanox.com>

These are the Top of the Rack systems, equipped with Mellanox switch board with Mellanox Quantum or Spectrume-2 devices. FAN controller is implemented by the programmable device logic.

The default registers offsets set within the programmable device is as following:

pwm1	0xe3
fan1 (tacho1)	0xe4
fan2 (tacho2)	0xe5
fan3 (tacho3)	0xe6
fan4 (tacho4)	0xe7
fan5 (tacho5)	0xe8
fan6 (tacho6)	0xe9
fan7 (tacho7)	0xea
fan8 (tacho8)	0xeb
fan9 (tacho9)	0xec
fan10 (tacho10)	0xed
fan11 (tacho11)	0xee
fan12 (tacho12)	0xef

This setup can be re-programmed with other registers.

7.111.1 Description

The driver implements a simple interface for driving a fan connected to a PWM output and tachometer inputs. This driver obtains PWM and tachometers registers location according to the system configuration and creates FAN/PWM hwmon objects and a cooling device. PWM and tachometers are sensed through the on-board programmable device, which exports its register map. This device could be attached to any bus type, for which register mapping is supported. Single instance is created with one PWM control, up to 12 tachometers and one cooling device. It could be as many instances as programmable device supports. The driver exposes the fan to the user space through the hwmon's and thermal's sysfs interfaces.

7.111.2 /sys files in hwmon subsystem

fan[1-12]_fault	RO	files for tachometers TACH1-TACH12 fault indication
fan[1-12]_input	RO	files for tachometers TACH1-TACH12 input (in RPM)
pwm1	RW	file for fan[1-12] target duty cycle (0..255)

7.111.3 /sys files in thermal subsystem

cur_state	RW	file for current cooling state of the cooling device (0..max_state)
max_state	RO	file for maximum cooling state of the cooling device

7.112 Kernel driver nct6683

Supported chips:

- Nuvoton NCT6683D

  Prefix: 'nct6683'

  Addresses scanned: ISA address retrieved from Super I/O registers

  Datasheet: Available from Nuvoton upon request

Authors:

Guenter Roeck <linux@roeck-us.net>

7.112.1 Description

This driver implements support for the Nuvoton NCT6683D eSIO chip.

The chips implement up to shared 32 temperature and voltage sensors. It supports up to 16 fan rotation sensors and up to 8 fan control engines.

Temperatures are measured in degrees Celsius. Measurement resolution is 0.5 degrees C.

Voltage sensors (also known as IN sensors) report their values in millivolts.

Fan rotation speeds are reported in RPM (rotations per minute).

7.112.2 Usage Note

Limit register locations on Intel boards with EC firmware version 1.0 build date 04/03/13 do not match the register locations in the Nuvoton datasheet. Nuvoton confirms that Intel uses a special firmware version with different register addresses. The specification describing the Intel firmware is held under NDA by Nuvoton and Intel and not available to the public.

Some of the register locations can be reverse engineered; others are too well hidden. Given this, writing any values from the operating system is considered too risky with this firmware and has been disabled. All limits must all be written from the BIOS.

The driver has only been tested with the Intel firmware, and by default only instantiates on Intel boards. To enable it on non-Intel boards, set the 'force' module parameter to 1.

7.112.3 Tested Boards and Firmware Versions

The driver has been reported to work with the following boards and firmware versions.

Board	Firmware version
Intel DH87RL	NCT6683D EC firmware version 1.0 build 04/03/13
Intel DH87MC	NCT6683D EC firmware version 1.0 build 04/03/13
Intel DB85FL	NCT6683D EC firmware version 1.0 build 04/03/13

7.113 Kernel driver NCT6775

Note: This driver supersedes the NCT6775F and NCT6776F support in the W83627EHF driver.

Supported chips:

- Nuvoton NCT6102D/NCT6104D/NCT6106D

  Prefix: 'nct6106'

  Addresses scanned: ISA address retrieved from Super I/O registers

  Datasheet: Available from the Nuvoton web site

- Nuvoton NCT5572D/NCT6771F/NCT6772F/NCT6775F/W83677HG-I

  Prefix: 'nct6775'

  Addresses scanned: ISA address retrieved from Super I/O registers

  Datasheet: Available from Nuvoton upon request

- Nuvoton NCT5573D/NCT5577D/NCT6776D/NCT6776F

  Prefix: 'nct6776'

Addresses scanned: ISA address retrieved from Super I/O registers

Datasheet: Available from Nuvoton upon request

- Nuvoton NCT5532D/NCT6779D

  Prefix: ‘nct6779’

  Addresses scanned: ISA address retrieved from Super I/O registers

  Datasheet: Available from Nuvoton upon request

- Nuvoton NCT6791D

  Prefix: ‘nct6791’

  Addresses scanned: ISA address retrieved from Super I/O registers

  Datasheet: Available from Nuvoton upon request

- Nuvoton NCT6792D

  Prefix: ‘nct6792’

  Addresses scanned: ISA address retrieved from Super I/O registers

  Datasheet: Available from Nuvoton upon request

- Nuvoton NCT6793D

  Prefix: ‘nct6793’

  Addresses scanned: ISA address retrieved from Super I/O registers

  Datasheet: Available from Nuvoton upon request

- Nuvoton NCT6795D

  Prefix: ‘nct6795’

  Addresses scanned: ISA address retrieved from Super I/O registers

  Datasheet: Available from Nuvoton upon request

- Nuvoton NCT6796D

  Prefix: ‘nct6796’

  Addresses scanned: ISA address retrieved from Super I/O registers

  Datasheet: Available from Nuvoton upon request

Authors:

Guenter Roeck <linux@roeck-us.net>

7.113.1 Description

This driver implements support for the Nuvoton NCT6775F, NCT6776F, and NCT6779D and compatible super I/O chips.

The chips support up to 25 temperature monitoring sources. Up to 6 of those are direct temperature sensor inputs, the others are special sources such as PECI, PCH, and SMBUS. Depending on the chip type, 2 to 6 of the temperature sources can be monitored and compared against minimum, maximum, and critical temperatures. The driver reports up to 10 of the temperatures to the user. There are 4 to 5 fan rotation speed sensors, 8 to 15 analog voltage sensors, one VID, alarms with beep warnings (control unimplemented), and some automatic fan regulation strategies (plus manual fan control mode).

The temperature sensor sources on all chips are configurable. The configured source for each of the temperature sensors is provided in tempX_label.

Temperatures are measured in degrees Celsius and measurement resolution is either 1 degC or 0.5 degC, depending on the temperature source and configuration. An alarm is triggered when the temperature gets higher than the high limit; it stays on until the temperature falls below the hysteresis value. Alarms are only supported for temp1 to temp6, depending on the chip type.

Fan rotation speeds are reported in RPM (rotations per minute). An alarm is triggered if the rotation speed has dropped below a programmable limit. On NCT6775F, fan readings can be divided by a programmable divider (1, 2, 4, 8, 16, 32, 64 or 128) to give the readings more range or accuracy; the other chips do not have a fan speed divider. The driver sets the most suitable fan divisor itself; specifically, it increases the divider value each time a fan speed reading returns an invalid value, and it reduces it if the fan speed reading is lower than optimal. Some fans might not be present because they share pins with other functions.

Voltage sensors (also known as IN sensors) report their values in millivolts. An alarm is triggered if the voltage has crossed a programmable minimum or maximum limit.

The driver supports automatic fan control mode known as Thermal Cruise. In this mode, the chip attempts to keep the measured temperature in a predefined temperature range. If the temperature goes out of range, fan is driven slower/faster to reach the predefined range again.

The mode works for fan1-fan5.

7.113.2 sysfs attributes

pwm[1-7]

- this file stores PWM duty cycle or DC value (fan speed) in range:

  0 (lowest speed) to 255 (full)

pwm[1-7]_enable

- this file controls mode of fan/temperature control:
  - 0 Fan control disabled (fans set to maximum speed)

  - 1 Manual mode, write to pwm[0-5] any value 0-255
  - 2 “Thermal Cruise” mode
  - 3 “Fan Speed Cruise” mode
  - 4 “Smart Fan III” mode (NCT6775F only)
  - 5 “Smart Fan IV” mode

pwm[1-7]_mode

- controls if output is PWM or DC level
  - 0 DC output
  - 1 PWM output

7.113.3 Common fan control attributes

pwm[1-7]_temp_sel Temperature source. Value is temperature sensor index. For example, select ‘1’ for temp1_input.

pwm[1-7]_weight_temp_sel Secondary temperature source. Value is temperature sensor index. For example, select ‘1’ for temp1_input. Set to 0 to disable secondary temperature control.

If secondary temperature functionality is enabled, it is controlled with the following attributes.

pwm[1-7]_weight_duty_step Duty step size.

pwm[1-7]_weight_temp_step Temperature step size. With each step over temp_step_base, the value of weight_duty_step is added to the current pwm value.

pwm[1-7]_weight_temp_step_base Temperature at which secondary temperature control kicks in.

pwm[1-7]_weight_temp_step_tol Temperature step tolerance.

7.113.4 Thermal Cruise mode (2)

If the temperature is in the range defined by:

pwm[1-7]_target_temp Target temperature, unit millidegree Celsius (range 0 - 127000)

pwm[1-7]_temp_tolerance Target temperature tolerance, unit millidegree Celsius

There are no changes to fan speed. Once the temperature leaves the interval, fan speed increases (if temperature is higher that desired) or decreases (if temperature is lower than desired), using the following limits and time intervals.

pwm[1-7]_start fan pwm start value (range 1 - 255), to start fan when the temperature is above defined range.

pwm[1-7]_floor lowest fan pwm (range 0 - 255) if temperature is below the defined range. If set to 0, the fan is expected to stop if the temperature is below the defined range.

pwm[1-7]_step_up_time milliseconds before fan speed is increased

pwm[1-7]_step_down_time milliseconds before fan speed is decreased

pwm[1-7]_stop_time how many milliseconds must elapse to switch corresponding fan off (when the temperature was below defined range).

7.113.5 Speed Cruise mode (3)

This modes tries to keep the fan speed constant.

fan[1-7]_target Target fan speed

fan[1-7]_tolerance Target speed tolerance

Untested; use at your own risk.

7.113.6 Smart Fan IV mode (5)

This mode offers multiple slopes to control the fan speed. The slopes can be controlled by setting the pwm and temperature attributes. When the temperature rises, the chip will calculate the DC/PWM output based on the current slope. There are up to seven data points depending on the chip type. Subsequent data points should be set to higher temperatures and higher pwm values to achieve higher fan speeds with increasing temperature. The last data point reflects critical temperature mode, in which the fans should run at full speed.

pwm[1-7]_auto_point[1-7]_pwm pwm value to be set if temperature reaches matching temperature range.

pwm[1-7]_auto_point[1-7]_temp Temperature over which the matching pwm is enabled.

pwm[1-7]_temp_tolerance Temperature tolerance, unit millidegree Celsius

pwm[1-7]_crit_temp_tolerance Temperature tolerance for critical temperature, unit millidegree Celsius

pwm[1-7]_step_up_time milliseconds before fan speed is increased

pwm[1-7]_step_down_time milliseconds before fan speed is decreased

7.113.7 Usage Notes

On various ASUS boards with NCT6776F, it appears that CPUTIN is not really connected to anything and floats, or that it is connected to some non-standard temperature measurement device. As a result, the temperature reported on CPUTIN will not reflect a usable value. It often reports unreasonably high temperatures, and in some cases the reported temperature declines if the actual temperature increases (similar to the raw PECI temperature value - see PECI specification for details). CPUTIN should therefore be be ignored on ASUS boards. The CPU temperature on ASUS boards is reported from PECI 0.

7.114 Kernel driver nct7802

Supported chips:

- Nuvoton NCT7802Y

  Prefix: 'nct7802'

  Addresses scanned: I2C 0x28..0x2f

  Datasheet: Available from Nuvoton web site

Authors:

  Guenter Roeck <linux@roeck-us.net>

7.114.1 Description

This driver implements support for the Nuvoton NCT7802Y hardware monitoring chip. NCT7802Y supports 6 temperature sensors, 5 voltage sensors, and 3 fan speed sensors.

Smart Fan™ speed control is available via pwmX_auto_point attributes.

7.114.2 Tested Boards and BIOS Versions

The driver has been reported to work with the following boards and BIOS versions.

Board	BIOS version
Kontron COMe-bSC2	CHR2E934.001.GGO
Kontron COMe-bIP2	CCR2E212

7.115 Kernel driver nct7904

Supported chip:

- Nuvoton NCT7904D

  Prefix: nct7904

  Addresses: I2C 0x2d, 0x2e

  Datasheet: Publicly available at Nuvoton website

  http://www.nuvoton.com/

Author: Vadim V. Vlasov <vvlasov@dev.rtsoft.ru>

7.115.1 Description

The NCT7904D is a hardware monitor supporting up to 20 voltage sensors, internal temperature sensor, Intel PECI and AMD SB-TSI CPU temperature interface, up to 12 fan tachometer inputs, up to 4 fan control channels with SmartFan.

7.115.2 Sysfs entries

Currently, the driver supports only the following features:

in[1-20]_input	Input voltage measurements (mV)
fan[1-12]_input	Fan tachometer measurements (rpm)
temp1_input	Local temperature (1/1000 degree, 0.125 degree resolution)
temp[2-9]_input	CPU temperatures (1/1000 degree, 0.125 degree resolution)
pwm[1-4]_enable	R/W, 1/2 for manual or SmartFan mode Setting SmartFan mode is supported only if it has been previously configured by BIOS (or configuration EEPROM)
pwm[1-4]	R/O in SmartFan mode, R/W in manual control mode

The driver checks sensor control registers and does not export the sensors that are not enabled. Anyway, a sensor that is enabled may actually be not connected and thus provide zero readings.

7.115.3 Limitations

The following features are not supported in current version:

- SmartFan control
- Watchdog
- GPIO
- external temperature sensors
- SMI
- min/max values
- many other···

7.116 Kernel driver npcm750-pwm-fan

Supported chips:

NUVOTON NPCM750/730/715/705

Authors:

<tomer.maimon@nuvoton.com>

7.116.1 Description:

This driver implements support for NUVOTON NPCM7XX PWM and Fan Tacho controller. The PWM controller supports up to 8 PWM outputs. The Fan tacho controller supports up to 16 tachometer inputs.

The driver provides the following sensor accesses in sysfs:

fanX_input	ro	provide current fan rotation value in RPM as reported by the fan to the device.
pwmX	rw	get or set PWM fan control value. This is an integer value between 0(off) and 255(full speed).

7.117 Kernel driver nsa320_hwmon

Supported chips:

- Holtek HT46R065 microcontroller with onboard firmware that configures it to act as a hardware monitor.

  Prefix: 'nsa320'

  Addresses scanned: none

  Datasheet: Not available, driver was reverse engineered based upon the Zyxel kernel source

Author:

Adam Baker <linux@baker-net.org.uk>

7.117.1 Description

This chip is known to be used in the Zyxel NSA320 and NSA325 NAS Units and also in some variants of the NSA310 but the driver has only been tested on the NSA320. In all of these devices it is connected to the same 3 GPIO lines which are used to provide chip select, clock and data lines. The interface behaves similarly to SPI but at much lower speeds than are normally used for SPI.

Following each chip select pulse the chip will generate a single 32 bit word that contains 0x55 as a marker to indicate that data is being read correctly, followed by an 8 bit fan speed in 100s of RPM and a 16 bit temperature in tenths of a degree.

7.117.2 sysfs-Interface

temp1_input	temperature input
fan1_input	fan speed

7.117.3 Notes

The access timings used in the driver are the same as used in the Zyxel provided kernel. Testing has shown that if the delay between chip select and the first clock pulse is reduced from 100 ms to just under 10ms then the chip will not produce any output. If the duration of either phase of the clock is reduced from 100 us to less than 15 us then data pulses are likely to be read twice corrupting the output. The above analysis is based upon a sample of one unit but suggests that the Zyxel provided delay values include a reasonable tolerance.

The driver incorporates a limit that it will not check for updated values faster than once a second. This is because the hardware takes a relatively long time to read the data from the device and when it does it reads both temp and fan speed. As the most likely case for two accesses in quick succession is to read both of these values avoiding a second read delay is desirable.

7.118 Kernel driver ntc_thermistor

Supported thermistors from Murata:

- Murata NTC Thermistors NCP15WB473, NCP18WB473, NCP21WB473, NCP03WB473, NCP15WL333, NCP03WF104, NCP15XH103

  Prefixes: ‘ncp15wb473’ , ‘ncp18wb473’ , ‘ncp21wb473’ , ‘ncp03wb473’ , ‘ncp15wl333’ , ‘ncp03wf104’ , ‘ncp15xh103’

  Datasheet: Publicly available at Murata

Supported thermistors from EPCOS:

- EPCOS NTC Thermistors B57330V2103

  Prefixes: b57330v2103

  Datasheet: Publicly available at EPCOS

Other NTC thermistors can be supported simply by adding compensation tables; e.g., NCP15WL333 support is added by the table ncpXXwl333.

Authors:

MyungJoo Ham <myungjoo.ham@samsung.com>

7.118.1 Description

The NTC (Negative Temperature Coefficient) thermistor is a simple thermistor that requires users to provide the resistance and lookup the corresponding compensation table to get the temperature input.

The NTC driver provides lookup tables with a linear approximation function and four circuit models with an option not to use any of the four models.

Using the following convention:

```
$    resistor
[TH] the thermistor
```

The four circuit models provided are:

1. connect = NTC_CONNECTED_POSITIVE, pullup_ohm > 0:

```
[pullup_uV]
    |    |
   [TH]  $ (pullup_ohm)
    |    |
    +----+-----------------------[read_uV]
    |
    $ (pulldown_ohm)
    |
   -+- (ground)
```

2. connect = NTC_CONNECTED_POSITIVE, pullup_ohm = 0 (not-connected):

```
[pullup_uV]
    |
   [TH]
    |
    +----------------------------[read_uV]
    |
    $ (pulldown_ohm)
    |
   -+- (ground)
```

3. connect = NTC_CONNECTED_GROUND, pulldown_ohm > 0:

```
[pullup_uV]
    |
```

(continues on next page)

(continued from previous page)

```
    $ (pullup_ohm)
    |
    +----+-----------------------[read_uV]
    |    |
   [TH]  $ (pulldown_ohm)
    |    |
   -+----+- (ground)
```

4. connect = NTC_CONNECTED_GROUND, pulldown_ohm = 0 (not-connected):

```
[pullup_uV]
    |
    $ (pullup_ohm)
    |
    +----------------------------[read_uV]
    |
   [TH]
    |
   -+- (ground)
```

When one of the four circuit models is used, read_uV, pullup_uV, pullup_ohm, pulldown_ohm, and connect should be provided. When none of the four models are suitable or the user can get the resistance directly, the user should provide read_ohm and _not_ provide the others.

7.118.2 Sysfs Interface

name		the mandatory global attribute, the thermistor name.
temp1_type	RO	always 4 (thermistor)
temp1_input	RO	measure the temperature and provide the measured value. (reading this file initiates the reading procedure.)

Note that each NTC thermistor has only _one_ thermistor; thus, only temp1 exists.

7.119 Kernel driver occ-hwmon

Supported chips:

- POWER8
- POWER9

Author: Eddie James <eajames@linux.ibm.com>

7.119.1 Description

This driver supports hardware monitoring for the On-Chip Controller (OCC) embedded on POWER processors. The OCC is a device that collects and aggregates sensor data from the processor and the system. The OCC can provide the raw sensor data as well as perform thermal and power management on the system.

The P8 version of this driver is a client driver of I2C. It may be probed manually if an "ibm,p8-occ-hwmon" compatible device is found under the appropriate I2C bus node in the device-tree.

The P9 version of this driver is a client driver of the FSI-based OCC driver. It will be probed automatically by the FSI-based OCC driver.

7.119.2 Sysfs entries

The following attributes are supported. All attributes are read-only unless specified.

The OCC sensor ID is an integer that represents the unique identifier of the sensor with respect to the OCC. For example, a temperature sensor for the third DIMM slot in the system may have a sensor ID of 7. This mapping is unavailable to the device driver, which must therefore export the sensor ID as-is.

Some entries are only present with certain OCC sensor versions or only on certain OCCs in the system. The version number is not exported to the user but can be inferred.

temp[1-n]_label OCC sensor ID.

[with temperature sensor version 1]

temp[1-n]_input Measured temperature of the component in millidegrees Celsius.

[with temperature sensor version >= 2]

temp[1-n]_type The FRU (Field Replaceable Unit) type (represented by an integer) for the component that this sensor measures.

temp[1-n]_fault Temperature sensor fault boolean; 1 to indicate that a fault is present or 0 to indicate that no fault is present.

[with type == 3 (FRU type is VRM)]

temp[1-n]_alarm VRM temperature alarm boolean; 1 to indicate alarm, 0 to indicate no alarm

[else]

temp[1-n]_input Measured temperature of the component in millidegrees Celsius.

freq[1-n]_label OCC sensor ID.

freq[1-n]_input Measured frequency of the component in MHz.

power[1-n]_input Latest measured power reading of the component in microwatts.

power[1-n]_average Average power of the component in microwatts.

power[1-n]_average_interval The amount of time over which the power average was taken in microseconds.

[with power sensor version < 2]

power[1-n]_label OCC sensor ID.

[with power sensor version >= 2]

power[1-n]_label OCC sensor ID + function ID + channel in the form of a string, delimited by underscores, i.e. "0_15_1". Both the function ID and channel are integers that further identify the power sensor.

[with power sensor version 0xa0]

power[1-n]_label OCC sensor ID + sensor type in the form of a string, delimited by an underscore, i.e. "0_system". Sensor type will be one of "system", "proc", "vdd" or "vdn". For this sensor version, OCC sensor ID will be the same for all power sensors.

[present only on "master" OCC; represents the whole system power; only one of this type of power sensor will be present]

power[1-n]_label "system"

power[1-n]_input Latest system output power in microwatts.

power[1-n]_cap Current system power cap in microwatts.

power[1-n]_cap_not_redundant System power cap in microwatts when there is not redundant power.

power[1-n]_cap_max Maximum power cap that the OCC can enforce in microwatts.

power[1-n]_cap_min Minimum power cap that the OCC can enforce in microwatts.

power[1-n]_cap_user The power cap set by the user, in microwatts. This attribute will return 0 if no user power cap has been set. This attribute is read-write, but writing any precision below watts will be ignored, i.e. requesting a power cap of 500900000 microwatts will result in a power cap request of 500 watts.

[with caps sensor version > 1]

power[1-n]_cap_user_source Indicates how the user power cap was set. This is an integer that maps to system or firmware components that can set the user power cap.

The following "extn" sensors are exported as a way for the OCC to provide data that doesn't fit anywhere else. The meaning of these sensors is entirely dependent on their data, and cannot be statically defined.

extn[1-n]_label ASCII ID or OCC sensor ID.

extn[1-n]_flags This is one byte hexadecimal value. Bit 7 indicates the type of the label attribute; 1 for sensor ID, 0 for ASCII ID. Other bits are reserved.

extn[1-n]_input 6 bytes of hexadecimal data, with a meaning defined by the sensor ID.

7.120 Kernel driver pc87360

Supported chips:

- National Semiconductor PC87360, PC87363, PC87364, PC87365 and PC87366

  Prefixes: 'pc87360', 'pc87363', 'pc87364', 'pc87365', 'pc87366'

  Addresses scanned: none, address read from Super I/O config space

  Datasheets: No longer available

Authors: Jean Delvare <jdelvare@suse.de>

Thanks to Sandeep Mehta, Tonko de Rooy and Daniel Ceregatti for testing.

Thanks to Rudolf Marek for helping me investigate conversion issues.

7.120.1 Module Parameters

- **init int** Chip initialization level:
  - 0: None
  - **1**: Forcibly enable internal voltage and temperature channels, except in9
  - 2: Forcibly enable all voltage and temperature channels, except in9
  - 3: Forcibly enable all voltage and temperature channels, including in9

Note that this parameter has no effect for the PC87360, PC87363 and PC87364 chips.

Also note that for the PC87366, initialization levels 2 and 3 don't enable all temperature channels, because some of them share pins with each other, so they can't be used at the same time.

7.120.2 Description

The National Semiconductor PC87360 Super I/O chip contains monitoring and PWM control circuitry for two fans. The PC87363 chip is similar, and the PC87364 chip has monitoring and PWM control for a third fan.

The National Semiconductor PC87365 and PC87366 Super I/O chips are complete hardware monitoring chipsets, not only controlling and monitoring three fans, but also monitoring eleven voltage inputs and two (PC87365) or up to four (PC87366) temperatures.

Chip	#vin	#fan	#pwm	#temp	devid
PC87360	•	2	2	•	0xE1
PC87363	•	2	2	•	0xE8
PC87364	•	3	3	•	0xE4
PC87365	11	3	3	2	0xE5
PC87366	11	3	3	3-4	0xE9

The driver assumes that no more than one chip is present, and one of the standard Super I/O addresses is used (0x2E/0x2F or 0x4E/0x4F)

7.120.3 Fan Monitoring

Fan rotation speeds are reported in RPM (revolutions per minute). An alarm is triggered if the rotation speed has dropped below a programmable limit. A different alarm is triggered if the fan speed is too low to be measured.

Fan readings are affected by a programmable clock divider, giving the readings more range or accuracy. Usually, users have to learn how it works, but this driver implements dynamic clock divider selection, so you don't have to care no more.

For reference, here are a few values about clock dividers:

divider	speed (RPM)	RPM (RPM)	speed (RPM)
1	1882	18	6928
2	941	37	4898
4	470	74	3464
8	235	150	2449

For the curious, here is how the values above were computed:

- slowest measurable speed: clock/(255*divider)
- accuracy around 3000 RPM: 3000^2/clock
- highest accurate speed: sqrt(clock*100)

The clock speed for the PC87360 family is 480 kHz. I arbitrarily chose 100 RPM as the lowest acceptable accuracy.

As mentioned above, you don't have to care about this no more.

Note that not all RPM values can be represented, even when the best clock divider is selected. This is not only true for the measured speeds, but also for the programmable low limits, so don't be surprised if you try to set, say, fan1_min to 2900 and it finally reads 2909.

7.120.4 Fan Control

PWM (pulse width modulation) values range from 0 to 255, with 0 meaning that the fan is stopped, and 255 meaning that the fan goes at full speed.

Be extremely careful when changing PWM values. Low PWM values, even non-zero, can stop the fan, which may cause irreversible damage to your hardware if temperature increases too much. When changing PWM values, go step by step and keep an eye on temperatures.

One user reported problems with PWM. Changing PWM values would break fan speed readings. No explanation nor fix could be found.

7.120.5 Temperature Monitoring

Temperatures are reported in degrees Celsius. Each temperature measured has associated low, high and overtemperature limits, each of which triggers an alarm when crossed.

The first two temperature channels are external. The third one (PC87366 only) is internal.

The PC87366 has three additional temperature channels, based on thermistors (as opposed to thermal diodes for the first three temperature channels). For technical reasons, these channels are held by the VLM (voltage level monitor) logical device, not the TMS (temperature measurement) one. As a consequence, these temperatures are exported as voltages, and converted into temperatures in user-space.

Note that these three additional channels share their pins with the external thermal diode channels, so you (physically) can't use them all at the same time. Although it should be possible to mix the two sensor types, the documents from National Semiconductor suggest that motherboard manufacturers should choose one type and stick to it. So you will more likely have either channels 1 to 3 (thermal diodes) or 3 to 6 (internal thermal diode, and thermistors).

7.120.6 Voltage Monitoring

Voltages are reported relatively to a reference voltage, either internal or external. Some of them (in7:Vsb, in8:Vdd and in10:AVdd) are divided by two internally, you will have to compensate in sensors.conf. Others (in0 to in6) are likely to be divided externally. The meaning of each of these inputs as well as the values of the resistors used for division is left to the motherboard manufacturers, so you will have to document yourself and edit sensors.conf accordingly. National Semiconductor has a document with recommended resistor values for some voltages, but this still leaves much room for per motherboard specificities, unfortunately. Even worse, motherboard manufacturers don't seem to care about National Semiconductor's recommendations.

Each voltage measured has associated low and high limits, each of which triggers an alarm when crossed.

When available, VID inputs are used to provide the nominal CPU Core voltage. The driver will default to VRM 9.0, but this can be changed from user-space. The

chipsets can handle two sets of VID inputs (on dual-CPU systems), but the driver will only export one for now. This may change later if there is a need.

7.120.7 General Remarks

If an alarm triggers, it will remain triggered until the hardware register is read at least once. This means that the cause for the alarm may already have disappeared! Note that all hardware registers are read whenever any data is read (unless it is less than 2 seconds since the last update, in which case cached values are returned instead). As a consequence, when a once-only alarm triggers, it may take 2 seconds for it to show, and 2 more seconds for it to disappear.

Monitoring of in9 isn't enabled at lower init levels (<3) because that channel measures the battery voltage (Vbat). It is a known fact that repeatedly sampling the battery voltage reduces its lifetime. National Semiconductor smartly designed their chipset so that in9 is sampled only once every 1024 sampling cycles (that is every 34 minutes at the default sampling rate), so the effect is attenuated, but still present.

7.120.8 Limitations

The datasheets suggests that some values (fan mins, fan dividers) shouldn't be changed once the monitoring has started, but we ignore that recommendation. We'll reconsider if it actually causes trouble.

7.121 Kernel driver pc87427

Supported chips:

- National Semiconductor PC87427

  Prefix: 'pc87427'

  Addresses scanned: none, address read from Super I/O config space

  Datasheet: No longer available

Author: Jean Delvare <jdelvare@suse.de>

Thanks to Amir Habibi at Candelis for setting up a test system, and to Michael Kress for testing several iterations of this driver.

7.121.1 Description

The National Semiconductor Super I/O chip includes complete hardware monitoring capabilities. It can monitor up to 18 voltages, 8 fans and 6 temperature sensors. Only the fans and temperatures are supported at the moment, voltages aren't.

This chip also has fan controlling features (up to 4 PWM outputs), which are partly supported by this driver.

The driver assumes that no more than one chip is present, which seems reasonable.

7.121.2 Fan Monitoring

Fan rotation speeds are reported as 14-bit values from a gated clock signal. Speeds down to 83 RPM can be measured.

An alarm is triggered if the rotation speed drops below a programmable limit. Another alarm is triggered if the speed is too low to be measured (including stalled or missing fan).

7.121.3 Fan Speed Control

Fan speed can be controlled by PWM outputs. There are 4 possible modes: always off, always on, manual and automatic. The latter isn't supported by the driver: you can only return to that mode if it was the original setting, and the configuration interface is missing.

7.121.4 Temperature Monitoring

The PC87427 relies on external sensors (following the SensorPath standard), so the resolution and range depend on the type of sensor connected. The integer part can be 8-bit or 9-bit, and can be signed or not. I couldn't find a way to figure out the external sensor data temperature format, so user-space adjustment (typically by a factor 2) may be required.

7.122 Kernel driver pcf8591

Supported chips:

- Philips/NXP PCF8591

  Prefix: 'pcf8591'

  Addresses scanned: none

  Datasheet: Publicly available at the NXP website

  http://www.nxp.com/pip/PCF8591_6.html

Authors:

- Aurelien Jarno <aurelien@aurel32.net>
- valuable contributions by Jan M. Sendler <sendler@sendler.de>,
- Jean Delvare <jdelvare@suse.de>

7.122.1 Description

The PCF8591 is an 8-bit A/D and D/A converter (4 analog inputs and one analog output) for the I2C bus produced by Philips Semiconductors (now NXP). It is designed to provide a byte I2C interface to up to 4 separate devices.

The PCF8591 has 4 analog inputs programmable as single-ended or differential inputs:

- **mode 0** [four single ended inputs] Pins AIN0 to AIN3 are single ended inputs for channels 0 to 3
- **mode 1** [three differential inputs] Pins AIN3 is the common negative differential input Pins AIN0 to AIN2 are positive differential inputs for channels 0 to 2
- **mode 2** [single ended and differential mixed] Pins AIN0 and AIN1 are single ended inputs for channels 0 and 1 Pins AIN2 is the positive differential input for channel 3 Pins AIN3 is the negative differential input for channel 3
- **mode 3** [two differential inputs] Pins AIN0 is the positive differential input for channel 0 Pins AIN1 is the negative differential input for channel 0 Pins AIN2 is the positive differential input for channel 1 Pins AIN3 is the negative differential input for channel 1

See the datasheet for details.

7.122.2 Module parameters

- input_mode int

  Analog input mode:

  - 0 = four single ended inputs
  - 1 = three differential inputs
  - 2 = single ended and differential mixed
  - 3 = two differential inputs

7.122.3 Accessing PCF8591 via /sys interface

The PCF8591 is plainly impossible to detect! Thus the driver won't even try. You have to explicitly instantiate the device at the relevant address (in the interval [0x48..0x4f]) either through platform data, or using the sysfs interface. See Documentation/i2c/instantiating-devices.rst for details.

Directories are being created for each instantiated PCF8591:

/sys/bus/i2c/devices/<0>-<1>/ where <0> is the bus the chip is connected to (e. g. i2c-0) and <1> the chip address ([48..4f])

Inside these directories, there are such files:

in0_input, in1_input, in2_input, in3_input, out0_enable, out0_output, name

Name contains chip name.

The in0_input, in1_input, in2_input and in3_input files are RO. Reading gives the value of the corresponding channel. Depending on the current analog inputs configuration, files in2_input and in3_input may not exist. Values range from 0 to 255 for single ended inputs and -128 to +127 for differential inputs (8-bit ADC).

The out0_enable file is RW. Reading gives "1" for analog output enabled and "0" for analog output disabled. Writing accepts "0" and "1" accordingly.

The out0_output file is RW. Writing a number between 0 and 255 (8-bit DAC), send the value to the digital-to-analog converter. Note that a voltage will only appears on AOUT pin if aout0_enable equals 1. Reading returns the last value written.

7.123 Kernel driver pmbus

Supported chips:

- Ericsson BMR453, BMR454

  Prefixes: 'bmr453', 'bmr454'

  Addresses scanned: -

  Datasheet:

  http://archive.ericsson.net/service/internet/picov/get?DocNo=28701-EN/LZT146395

- ON Semiconductor ADP4000, NCP4200, NCP4208

  Prefixes: 'adp4000', 'ncp4200', 'ncp4208'

  Addresses scanned: -

  Datasheets:

  http://www.onsemi.com/pub_link/Collateral/ADP4000-D.PDF

  http://www.onsemi.com/pub_link/Collateral/NCP4200-D.PDF

  http://www.onsemi.com/pub_link/Collateral/JUNE%202009-%20REV.%200.PDF

- Lineage Power

  Prefixes: 'mdt040', 'pdt003', 'pdt006', 'pdt012', 'udt020'

  Addresses scanned: -

  Datasheets:

  http://www.lineagepower.com/oem/pdf/PDT003A0X.pdf

  http://www.lineagepower.com/oem/pdf/PDT006A0X.pdf

http://www.lineagepower.com/oem/pdf/PDT012A0X.pdf

http://www.lineagepower.com/oem/pdf/UDT020A0X.pdf

http://www.lineagepower.com/oem/pdf/MDT040A0X.pdf

- Texas Instruments TPS40400, TPS544B20, TPS544B25, TPS544C20, TPS544C25

  Prefixes: 'tps40400', 'tps544b20', 'tps544b25', 'tps544c20', 'tps544c25'

  Addresses scanned: -

  Datasheets:

  http://www.ti.com/lit/gpn/tps40400

  http://www.ti.com/lit/gpn/tps544b20

  http://www.ti.com/lit/gpn/tps544b25

  http://www.ti.com/lit/gpn/tps544c20

  http://www.ti.com/lit/gpn/tps544c25

- Maxim MAX20796

  Prefix: 'max20796'

  Addresses scanned: -

  Datasheet:

  Not published

- Generic PMBus devices

  Prefix: 'pmbus'

  Addresses scanned: -

  Datasheet: n.a.

Author: Guenter Roeck <linux@roeck-us.net>

7.123.1 Description

This driver supports hardware monitoring for various PMBus compliant devices. It supports voltage, current, power, and temperature sensors as supported by the device.

Each monitored channel has its own high and low limits, plus a critical limit.

Fan support will be added in a later version of this driver.

7.123.2 Usage Notes

This driver does not probe for PMBus devices, since there is no register which can be safely used to identify the chip (The MFG_ID register is not supported by all chips), and since there is no well defined address range for PMBus devices. You will have to instantiate the devices explicitly.

Example: the following will load the driver for an LTC2978 at address 0x60 on I2C bus #1:

```
$ modprobe pmbus
$ echo ltc2978 0x60 > /sys/bus/i2c/devices/i2c-1/new_device
```

7.123.3 Platform data support

Support for additional PMBus chips can be added by defining chip parameters in a new chip specific driver file. For example, (untested) code to add support for Emerson DS1200 power modules might look as follows:

```
static struct pmbus_driver_info ds1200_info = {
      .pages = 1,
      /* Note: All other sensors are in linear mode */
      .direct[PSC_VOLTAGE_OUT] = true,
      .direct[PSC_TEMPERATURE] = true,
      .direct[PSC_CURRENT_OUT] = true,
      .m[PSC_VOLTAGE_IN] = 1,
      .b[PSC_VOLTAGE_IN] = 0,
      .R[PSC_VOLTAGE_IN] = 3,
      .m[PSC_VOLTAGE_OUT] = 1,
      .b[PSC_VOLTAGE_OUT] = 0,
      .R[PSC_VOLTAGE_OUT] = 3,
      .m[PSC_TEMPERATURE] = 1,
      .b[PSC_TEMPERATURE] = 0,
      .R[PSC_TEMPERATURE] = 3,
      .func[0] = PMBUS_HAVE_VIN | PMBUS_HAVE_IIN | PMBUS_HAVE_STATUS_INPUT
                 | PMBUS_HAVE_VOUT | PMBUS_HAVE_STATUS_VOUT
                 | PMBUS_HAVE_IOUT | PMBUS_HAVE_STATUS_IOUT
                 | PMBUS_HAVE_PIN | PMBUS_HAVE_POUT
                 | PMBUS_HAVE_TEMP | PMBUS_HAVE_STATUS_TEMP
                 | PMBUS_HAVE_FAN12 | PMBUS_HAVE_STATUS_FAN12,
};

static int ds1200_probe(struct i2c_client *client,
                        const struct i2c_device_id *id)
{
      return pmbus_do_probe(client, id, &ds1200_info);
}

static int ds1200_remove(struct i2c_client *client)
{
      return pmbus_do_remove(client);
}

static const struct i2c_device_id ds1200_id[] = {
      {"ds1200", 0},
```

(continues on next page)

(continued from previous page)

```
      {}
};

MODULE_DEVICE_TABLE(i2c, ds1200_id);

/* This is the driver that will be inserted */
static struct i2c_driver ds1200_driver = {
      .driver = {
                  .name = "ds1200",
                  },
      .probe = ds1200_probe,
      .remove = ds1200_remove,
      .id_table = ds1200_id,
};

static int __init ds1200_init(void)
{
      return i2c_add_driver(&ds1200_driver);
}

static void __exit ds1200_exit(void)
{
      i2c_del_driver(&ds1200_driver);
}
```

7.123.4 Sysfs entries

When probing the chip, the driver identifies which PMBus registers are supported, and determines available sensors from this information. Attribute files only exist if respective sensors are supported by the chip. Labels are provided to inform the user about the sensor associated with a given sysfs entry.

The following attributes are supported. Limits are read-write; all other attributes are read-only.

inX_input	Measured voltage. From READ_VIN or READ_VOUT register.
inX_min	Minimum Voltage. From VIN_UV_WARN_LIMIT or VOUT_UV_WARN_LIM
inX_max	Maximum voltage. From VIN_OV_WARN_LIMIT or VOUT_OV_WARN_LIM
inX_lcrit	Critical minimum Voltage. From VIN_UV_FAULT_LIMIT or VOUT_UV_FAU
inX_crit	Critical maximum voltage. From VIN_OV_FAULT_LIMIT or VOUT_OV_FAU
inX_min_alarm	Voltage low alarm. From VOLTAGE_UV_WARNING status.
inX_max_alarm	Voltage high alarm. From VOLTAGE_OV_WARNING status.
inX_lcrit_alarm	Voltage critical low alarm. From VOLTAGE_UV_FAULT status.
inX_crit_alarm	Voltage critical high alarm. From VOLTAGE_OV_FAULT status.
inX_label	“vin”, “vcap”, or “voutY”
currX_input	Measured current. From READ_IIN or READ_IOUT register.
currX_max	Maximum current. From IIN_OC_WARN_LIMIT or IOUT_OC_WARN_LIMI
currX_lcrit	Critical minimum output current. From IOUT_UC_FAULT_LIMIT register.
currX_crit	Critical maximum current. From IIN_OC_FAULT_LIMIT or IOUT_OC_FAU
currX_alarm	Current high alarm. From IIN_OC_WARNING or IOUT_OC_WARNING sta

Continued on next page

Table 16 – continued from previous page

currX_max_alarm	Current high alarm. From IIN_OC_WARN_LIMIT or IOUT_OC_WARN_LIM
currX_lcrit_alarm	Output current critical low alarm. From IOUT_UC_FAULT status.
currX_crit_alarm	Current critical high alarm. From IIN_OC_FAULT or IOUT_OC_FAULT sta
currX_label	“iin”, “iinY”, “iinY.Z”, “ioutY”, or “ioutY.Z”, where Y reflects the p
powerX_input	Measured power. From READ_PIN or READ_POUT register.
powerX_cap	Output power cap. From POUT_MAX register.
powerX_max	Power limit. From PIN_OP_WARN_LIMIT or POUT_OP_WARN_LIMIT re
powerX_crit	Critical output power limit. From POUT_OP_FAULT_LIMIT register.
powerX_alarm	Power high alarm. From PIN_OP_WARNING or POUT_OP_WARNING sta
powerX_crit_alarm	Output power critical high alarm. From POUT_OP_FAULT status.
powerX_label	“pin”, “pinY”, “pinY.Z”, “poutY”, or “poutY.Z”, where Y reflects t
tempX_input	Measured temperature. From READ_TEMPERATURE_X register.
tempX_min	Mimimum temperature. From UT_WARN_LIMIT register.
tempX_max	Maximum temperature. From OT_WARN_LIMIT register.
tempX_lcrit	Critical low temperature. From UT_FAULT_LIMIT register.
tempX_crit	Critical high temperature. From OT_FAULT_LIMIT register.
tempX_min_alarm	Chip temperature low alarm. Set by comparing READ_TEMPERATURE_
tempX_max_alarm	Chip temperature high alarm. Set by comparing READ_TEMPERATURE
tempX_lcrit_alarm	Chip temperature critical low alarm. Set by comparing READ_TEMPERA
tempX_crit_alarm	Chip temperature critical high alarm. Set by comparing READ_TEMPER

7.124 Kernel driver powr1220

Supported chips:

- Lattice POWR1220AT8

  Prefix: ‘powr1220’

  Addresses scanned: none

  Datasheet: Publicly available at the Lattice website

  http://www.latticesemi.com/

Author: Scott Kanowitz <scott.kanowitz@gmail.com>

7.124.1 Description

This driver supports the Lattice POWR1220AT8 chip. The POWR1220 includes voltage monitoring for 14 inputs as well as trim settings for output voltages and GPIOs. This driver implements the voltage monitoring portion of the chip.

Voltages are sampled by a 12-bit ADC with a step size of 2 mV. An in-line attenuator allows measurements from 0 to 6 V. The attenuator is enabled or disabled depending on the setting of the input’s max value. The driver will enable the attenuator for any value over the low measurement range maximum of 2 V.

The input naming convention is as follows:

driver name	pin name
in0	VMON1
in1	VMON2
in2	VMON3
in2	VMON4
in4	VMON5
in5	VMON6
in6	VMON7
in7	VMON8
in8	VMON9
in9	VMON10
in10	VMON11
in11	VMON12
in12	VCCA
in13	VCCINP

The ADC readings are updated on request with a minimum period of 1s.

7.125 Kernel driver pxe1610

Supported chips:

- Infineon PXE1610

  Prefix: 'pxe1610'

  Addresses scanned: -

  Datasheet: Datasheet is not publicly available.

- Infineon PXE1110

  Prefix: 'pxe1110'

  Addresses scanned: -

  Datasheet: Datasheet is not publicly available.

- Infineon PXM1310

  Prefix: 'pxm1310'

  Addresses scanned: -

  Datasheet: Datasheet is not publicly available.

Author: Vijay Khemka <vijaykhemka@fb.com>

7.125.1 Description

PXE1610/PXE1110 are Multi-rail/Multiphase Digital Controllers and compliant to

- Intel VR13 DC-DC converter specifications.
- Intel SVID protocol.

Used for Vcore power regulation for Intel VR13 based microprocessors

- Servers, Workstations, and High-end desktops

PXM1310 is a Multi-rail Controller and it is compliant to

- Intel VR13 DC-DC converter specifications.
- Intel SVID protocol.

Used for DDR3/DDR4 Memory power regulation for Intel VR13 and IMVP8 based systems

7.125.2 Usage Notes

This driver does not probe for PMBus devices. You will have to instantiate devices explicitly.

Example: the following commands will load the driver for an PXE1610 at address 0x70 on I2C bus #4:

```
# modprobe pxe1610
# echo pxe1610 0x70 > /sys/bus/i2c/devices/i2c-4/new_device
```

It can also be instantiated by declaring in device tree

7.125.3 Sysfs attributes

curr1_label	“iin”
curr1_input	Measured input current
curr1_alarm	Current high alarm
curr[2-4]_label	“iout[1-3]”
curr[2-4]_input	Measured output current
curr[2-4]_crit	Critical maximum current
curr[2-4]_crit_alarm	Current critical high alarm
in1_label	“vin”
in1_input	Measured input voltage
in1_crit	Critical maximum input voltage
in1_crit_alarm	Input voltage critical high alarm
in[2-4]_label	“vout[1-3]”
in[2-4]_input	Measured output voltage
in[2-4]_lcrit	Critical minimum output voltage
in[2-4]_lcrit_alarm	Output voltage critical low alarm
in[2-4]_crit	Critical maximum output voltage
in[2-4]_crit_alarm	Output voltage critical high alarm
power1_label	“pin”
power1_input	Measured input power
power1_alarm	Input power high alarm
power[2-4]_label	“pout[1-3]”
power[2-4]_input	Measured output power
temp[1-3]_input	Measured temperature
temp[1-3]_crit	Critical high temperature
temp[1-3]_crit_alarm	Chip temperature critical high alarm
temp[1-3]_max	Maximum temperature
temp[1-3]_max_alarm	Chip temperature high alarm

7.126 Kernel driver pwm-fan

This driver enables the use of a PWM module to drive a fan. It uses the generic PWM interface thus it is hardware independent. It can be used on many SoCs, as long as the SoC supplies a PWM line driver that exposes the generic PWM API.

Author: Kamil Debski <k.debski@samsung.com>

7.126.1 Description

The driver implements a simple interface for driving a fan connected to a PWM output. It uses the generic PWM interface, thus it can be used with a range of SoCs. The driver exposes the fan to the user space through the hwmon’s sysfs interface.

The fan rotation speed returned via the optional ‘fan1_input’ is extrapolated from the sampled interrupts from the tachometer signal within 1 second.

7.127 Kernel driver raspberrypi-hwmon

Supported boards:

- Raspberry Pi A+ (via GPIO on SoC)
- Raspberry Pi B+ (via GPIO on SoC)
- Raspberry Pi 2 B (via GPIO on SoC)
- Raspberry Pi 3 B (via GPIO on port expander)
- Raspberry Pi 3 B+ (via PMIC)

Author: Stefan Wahren <stefan.wahren@i2se.com>

7.127.1 Description

This driver periodically polls a mailbox property of the VC4 firmware to detect undervoltage conditions.

7.127.2 Sysfs entries

in0_lcrit_alarm	Undervoltage alarm

7.128 Kernel driver sch5627

Supported chips:

- SMSC SCH5627

  Prefix: 'sch5627'

  Addresses scanned: none, address read from Super I/O config space

  Datasheet: Application Note available upon request

Author: Hans de Goede <hdegoede@redhat.com>

7.128.1 Description

SMSC SCH5627 Super I/O chips include complete hardware monitoring capabilities. They can monitor up to 5 voltages, 4 fans and 8 temperatures.

The SMSC SCH5627 hardware monitoring part also contains an integrated watchdog. In order for this watchdog to function some motherboard specific initialization most be done by the BIOS, so if the watchdog is not enabled by the BIOS the sch5627 driver will not register a watchdog device.

The hardware monitoring part of the SMSC SCH5627 is accessed by talking through an embedded microcontroller. An application note describing the protocol for communicating with the microcontroller is available upon request. Please mail me if you want a copy.

7.129 Kernel driver sch5636

Supported chips:

- SMSC SCH5636

  Prefix: 'sch5636'

  Addresses scanned: none, address read from Super I/O config space

Author: Hans de Goede <hdegoede@redhat.com>

7.129.1 Description

SMSC SCH5636 Super I/O chips include an embedded microcontroller for hardware monitoring solutions, allowing motherboard manufacturers to create their own custom hwmon solution based upon the SCH5636.

Currently the sch5636 driver only supports the Fujitsu Theseus SCH5636 based hwmon solution. The sch5636 driver runs a sanity check on loading to ensure it is dealing with a Fujitsu Theseus and not with another custom SCH5636 based hwmon solution.

The Fujitsu Theseus can monitor up to 5 voltages, 8 fans and 16 temperatures. Note that the driver detects how many fan headers / temperature sensors are actually implemented on the motherboard, so you will likely see fewer temperature and fan inputs.

The Fujitsu Theseus hwmon solution also contains an integrated watchdog. This watchdog is fully supported by the sch5636 driver.

An application note describing the Theseus' registers, as well as an application note describing the protocol for communicating with the microcontroller is available upon request. Please mail me if you want a copy.

7.130 Kernel driver scpi-hwmon

Supported chips:

- Chips based on ARM System Control Processor Interface

  Addresses scanned: -

  Datasheet: http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.dui0922b/index.html

Author: Punit Agrawal <punit.agrawal@arm.com>

7.130.1 Description

This driver supports hardware monitoring for SoC's based on the ARM System Control Processor (SCP) implementing the System Control Processor Interface (SCPI). The following sensor types are supported by the SCP:

- temperature
- voltage
- current
- power

The SCP interface provides an API to query the available sensors and their values which are then exported to userspace by this driver.

7.130.2 Usage Notes

The driver relies on device tree node to indicate the presence of SCPI support in the kernel. See Documentation/devicetree/bindings/arm/arm,scpi.txt for details of the devicetree node.

7.131 Kernel driver sht15

Authors:

- Wouter Horre
- Jonathan Cameron
- Vivien Didelot <vivien.didelot@savoirfairelinux.com>
- Jerome Oufella <jerome.oufella@savoirfairelinux.com>

Supported chips:

- Sensirion SHT10

  Prefix: 'sht10'
- Sensirion SHT11

  Prefix: 'sht11'
- Sensirion SHT15

  Prefix: 'sht15'
- Sensirion SHT71

  Prefix: 'sht71'
- Sensirion SHT75

  Prefix: 'sht75'

Datasheet: Publicly available at the Sensirion website

http://www.sensirion.ch/en/pdf/product_information/Datasheet-humidity-sensor-SHT1x.pdf

7.131.1 Description

The SHT10, SHT11, SHT15, SHT71, and SHT75 are humidity and temperature sensors.

The devices communicate using two GPIO lines.

Supported resolutions for the measurements are 14 bits for temperature and 12 bits for humidity, or 12 bits for temperature and 8 bits for humidity.

The humidity calibration coefficients are programmed into an OTP memory on the chip. These coefficients are used to internally calibrate the signals from the sensors. Disabling the reload of those coefficients allows saving 10ms for each measurement and decrease power consumption, while losing on precision.

Some options may be set via sysfs attributes.

Notes:

- The regulator supply name is set to “vcc”.
- If a CRC validation fails, a soft reset command is sent, which resets status register to its hardware default value, but the driver will try to restore the previous device configuration.

7.131.2 Platform data

- checksum: set it to true to enable CRC validation of the readings (default to false).
- no_otp_reload: flag to indicate not to reload from OTP (default to false).
- low_resolution: flag to indicate the temp/humidity resolution to use (default to false).

7.131.3 Sysfs interface

temp1_input	temperature input
humidity1_input	humidity input
heater_enable	write 1 in this attribute to enable the on-chip heater, 0 to disable it. Be careful not to enable the heater for too long.
temp1_fault	if 1, this means that the voltage is low (below 2.47V) and measurement may be invalid.
humidity1_fault	same as temp1_fault.

7.132 Kernel driver sht21

Supported chips:

- Sensirion SHT21

  Prefix: ‘sht21’

  Addresses scanned: none

  Datasheet: Publicly available at the Sensirion website

  http://www.sensirion.com/file/datasheet_sht21

- Sensirion SHT25

  Prefix: ‘sht25’

  Addresses scanned: none

  Datasheet: Publicly available at the Sensirion website

  http://www.sensirion.com/file/datasheet_sht25

Author:

Urs Fleisch <urs.fleisch@sensirion.com>

7.132.1 Description

The SHT21 and SHT25 are humidity and temperature sensors in a DFN package of only 3 x 3 mm footprint and 1.1 mm height. The difference between the two devices is the higher level of precision of the SHT25 (1.8% relative humidity, 0.2 degree Celsius) compared with the SHT21 (2.0% relative humidity, 0.3 degree Celsius).

The devices communicate with the I2C protocol. All sensors are set to the same I2C address 0x40, so an entry with I2C_BOARD_INFO(“sht21” , 0x40) can be used in the board setup code.

7.132.2 sysfs-Interface

temp1_input

- temperature input

humidity1_input

- humidity input

eic

- Electronic Identification Code

7.132.3 Notes

The driver uses the default resolution settings of 12 bit for humidity and 14 bit for temperature, which results in typical measurement times of 22 ms for humidity and 66 ms for temperature. To keep self heating below 0.1 degree Celsius, the device should not be active for more than 10% of the time, e.g. maximum two measurements per second at the given resolution.

Different resolutions, the on-chip heater, and using the CRC checksum are not supported yet.

7.133 Kernel driver sht3x

Supported chips:

- Sensirion SHT3x-DIS

  Prefix: 'sht3x'

  Addresses scanned: none

  Datasheet: https://www.sensirion.com/file/datasheet_sht3x_digital

Author:

- David Frey <david.frey@sensirion.com>
- Pascal Sachs <pascal.sachs@sensirion.com>

7.133.1 Description

This driver implements support for the Sensirion SHT3x-DIS chip, a humidity and temperature sensor. Temperature is measured in degrees celsius, relative humidity is expressed as a percentage. In the sysfs interface, all values are scaled by 1000, i.e. the value for 31.5 degrees celsius is 31500.

The device communicates with the I2C protocol. Sensors can have the I2C addresses 0x44 or 0x45, depending on the wiring. See Documentation/i2c/instantiating-devices.rst for methods to instantiate the device.

There are two options configurable by means of sht3x_platform_data:

1. blocking (pull the I2C clock line down while performing the measurement) or non-blocking mode. Blocking mode will guarantee the fastest result but the I2C bus will be busy during that time. By default, non-blocking mode is used. Make sure clock-stretching works properly on your device if you want to use blocking mode.
2. high or low accuracy. High accuracy is used by default and using it is strongly recommended.

The sht3x sensor supports a single shot mode as well as 5 periodic measure modes, which can be controlled with the update_interval sysfs interface. The allowed update_interval in milliseconds are as follows:

0		single shot mode
2000	0.5 Hz	periodic measurement
1000	1 Hz	periodic measurement
500	2 Hz	periodic measurement
250	4 Hz	periodic measurement
100	10 Hz	periodic measurement

In the periodic measure mode, the sensor automatically triggers a measurement with the configured update interval on the chip. When a temperature or humidity reading exceeds the configured limits, the alert attribute is set to 1 and the alert pin on the sensor is set to high. When the temperature and humidity readings move back between the hysteresis values, the alert bit is set to 0 and the alert pin on the sensor is set to low.

7.133.2 sysfs-Interface

temp1_input:	temperature input
humidity1_input:	humidity input
temp1_max:	temperature max value
temp1_max_hyst:	temperature hysteresis value for max limit
humidity1_max:	humidity max value
humidity1_max_hyst:	humidity hysteresis value for max limit
temp1_min:	temperature min value
temp1_min_hyst:	temperature hysteresis value for min limit
humidity1_min:	humidity min value
humidity1_min_hyst:	humidity hysteresis value for min limit
temp1_alarm:	alarm flag is set to 1 if the temperature is outside the configured limits. Alarm only works in periodic measure mode
humidity1_alarm:	alarm flag is set to 1 if the humidity is outside the configured limits. Alarm only works in periodic measure mode
heater_enable:	heater enable, heating element removes excess humidity from sensor: • 0: turned off • 1: turned on
update_interval:	update interval, 0 for single shot, interval in msec for periodic measurement. If the interval is not supported by the sensor, the next faster interval is chosen

7.134 Kernel driver shtc1

Supported chips:

- Sensirion SHTC1

  Prefix: 'shtc1'

  Addresses scanned: none

  Datasheet: http://www.sensirion.com/file/datasheet_shtc1

- Sensirion SHTW1

  Prefix: 'shtw1'

  Addresses scanned: none

  Datasheet: http://www.sensirion.com/file/datasheet_shtw1

- Sensirion SHTC3

  Prefix: 'shtc3'

  Addresses scanned: none

  Datasheet: http://www.sensirion.com/file/datasheet_shtc3

Author:

Johannes Winkelmann <johannes.winkelmann@sensirion.com>

7.134.1 Description

This driver implements support for the Sensirion SHTC1, SHTW1, and SHTC3 chips, a humidity and temperature sensor. Temperature is measured in degrees celsius, relative humidity is expressed as a percentage.

The device communicates with the I2C protocol. All sensors are set to I2C address 0x70. See Documentation/i2c/instantiating-devices.rst for methods to instantiate the device.

There are two options configurable by means of shtc1_platform_data:

1. blocking (pull the I2C clock line down while performing the measurement) or non-blocking mode. Blocking mode will guarantee the fastest result but the I2C bus will be busy during that time. By default, non-blocking mode is used. Make sure clock-stretching works properly on your device if you want to use blocking mode.
2. high or low accuracy. High accuracy is used by default and using it is strongly recommended.

7.134.2 sysfs-Interface

temp1_input

- temperature input

humidity1_input

- humidity input

7.135 Kernel driver sis5595

Supported chips:

- Silicon Integrated Systems Corp. SiS5595 Southbridge Hardware Monitor

  Prefix: 'sis5595'

  Addresses scanned: ISA in PCI-space encoded address

  Datasheet: Publicly available at the Silicon Integrated Systems Corp. site.

Authors:

- Kyösti Mälkki <kmalkki@cc.hut.fi>,
- Mark D. Studebaker <mdsxyz123@yahoo.com>,
- Aurelien Jarno <aurelien@aurel32.net> 2.6 port

SiS southbridge has a LM78-like chip integrated on the same IC. This driver is a customized copy of lm78.c

Supports following revisions:

Version	PCI ID	PCI Revision
1	1039/0008	AF or less
2	1039/0008	B0 or greater

Note: these chips contain a 0008 device which is incompatible with the 5595. We recognize these by the presence of the listed "blacklist" PCI ID and refuse to load.

NOT SUPPORTED	PCI ID	BLACKLIST PCI ID
540	0008	0540
550	0008	0550
5513	0008	5511
5581	0008	5597
5582	0008	5597
5597	0008	5597
630	0008	0630
645	0008	0645
730	0008	0730
735	0008	0735

7.135.1 Module Parameters

force_addr=0xaddr	Set the I/O base address. Useful for boards that don't set the address in the BIOS. Does not do a PCI force; the device must still be present in lspci. Don't use this unless the driver complains that the base address is not set. Example: 'modprobe sis5595 force_addr=0x290'

7.135.2 Description

The SiS5595 southbridge has integrated hardware monitor functions. It also has an I2C bus, but this driver only supports the hardware monitor. For the I2C bus driver see i2c-sis5595.

The SiS5595 implements zero or one temperature sensor, two fan speed sensors, four or five voltage sensors, and alarms.

On the first version of the chip, there are four voltage sensors and one temperature sensor.

On the second version of the chip, the temperature sensor (temp) and the fifth voltage sensor (in4) share a pin which is configurable, but not through the driver. Sorry. The driver senses the configuration of the pin, which was hopefully set by the BIOS.

Temperatures are measured in degrees Celsius. An alarm is triggered once when the max is crossed; it is also triggered when it drops below the min value. Measurements are guaranteed between -55 and +125 degrees, with a resolution of 1 degree.

Fan rotation speeds are reported in RPM (rotations per minute). An alarm is triggered if the rotation speed has dropped below a programmable limit. Fan readings can be divided by a programmable divider (1, 2, 4 or 8) to give the readings more range or accuracy. Not all RPM values can accurately be represented, so some rounding is done. With a divider of 2, the lowest representable value is around 2600 RPM.

Voltage sensors (also known as IN sensors) report their values in volts. An alarm is triggered if the voltage has crossed a programmable minimum or maximum limit. Note that minimum in this case always means 'closest to zero'; this is important for negative voltage measurements. All voltage inputs can measure voltages between 0 and 4.08 volts, with a resolution of 0.016 volt.

In addition to the alarms described above, there is a BTI alarm, which gets triggered when an external chip has crossed its limits. Usually, this is connected to some LM75-like chip; if at least one crosses its limits, this bit gets set.

If an alarm triggers, it will remain triggered until the hardware register is read at least once. This means that the cause for the alarm may already have disappeared! Note that in the current implementation, all hardware registers are read whenever any data is read (unless it is less than 1.5 seconds since the last update). This means that you can easily miss once-only alarms.

The SiS5595 only updates its values each 1.5 seconds; reading it more often will do no harm, but will return 'old' values.

7.135.3 Problems

Some chips refuse to be enabled. We don't know why. The driver will recognize this and print a message in dmesg.

7.136 Kernel driver smm665

Supported chips:

- Summit Microelectronics SMM465

  Prefix: 'smm465'

  Addresses scanned: -

  Datasheet:

  http://www.summitmicro.com/prod_select/summary/SMM465/SMM465DS.pdf

- Summit Microelectronics SMM665, SMM665B

  Prefix: 'smm665'

  Addresses scanned: -

  Datasheet:

  http://www.summitmicro.com/prod_select/summary/SMM665/SMM665B_2089_20.pdf

- Summit Microelectronics SMM665C

  Prefix: 'smm665c'

  Addresses scanned: -

  Datasheet:

  http://www.summitmicro.com/prod_select/summary/SMM665C/SMM665C_2125.pdf

- Summit Microelectronics SMM764

  Prefix: 'smm764'

  Addresses scanned: -

  Datasheet:

  http://www.summitmicro.com/prod_select/summary/SMM764/SMM764_2098.pdf

- Summit Microelectronics SMM766, SMM766B

  Prefix: 'smm766'

  Addresses scanned: -

Datasheets:

http://www.summitmicro.com/prod_select/summary/SMM766/SMM766_2086.pdf

http://www.summitmicro.com/prod_select/summary/SMM766B/SMM766B_2122.pdf

Author: Guenter Roeck <linux@roeck-us.net>

7.136.1 Module Parameters

- **vref: int** Default: 1250 (mV)

  Reference voltage on VREF_ADC pin in mV. It should not be necessary to set this parameter unless a non-default reference voltage is used.

7.136.2 Description

[From datasheet] The SMM665 is an Active DC Output power supply Controller that monitors, margins and cascade sequences power. The part monitors six power supply channels as well as VDD, 12V input, two general-purpose analog inputs and an internal temperature sensor using a 10-bit ADC.

Each monitored channel has its own high and low limits, plus a critical limit.

Support for SMM465, SMM764, and SMM766 has been implemented but is untested.

7.136.3 Usage Notes

This driver does not probe for devices, since there is no register which can be safely used to identify the chip. You will have to instantiate the devices explicitly. When instantiating the device, you have to specify its configuration register address.

Example: the following will load the driver for an SMM665 at address 0x57 on I2C bus #1:

```
$ modprobe smm665
$ echo smm665 0x57 > /sys/bus/i2c/devices/i2c-1/new_device
```

7.136.4 Sysfs entries

This driver uses the values in the datasheet to convert ADC register values into the values specified in the sysfs-interface document. All attributes are read only.

Min, max, lcrit, and crit values are used by the chip to trigger external signals and/or other activity. Triggered signals can include HEALTHY, RST, Power Off, or Fault depending on the chip configuration. The driver reports values as lcrit or crit if exceeding the limits triggers RST, Power Off, or Fault, and as min or max otherwise. For details please see the SMM665 datasheet.

For SMM465 and SMM764, values for Channel E and F are reported but undefined.

in1_input	12V input voltage (mV)
in2_input	3.3V (VDD) input voltage (mV)
in3_input	Channel A voltage (mV)
in4_input	Channel B voltage (mV)
in5_input	Channel C voltage (mV)
in6_input	Channel D voltage (mV)
in7_input	Channel E voltage (mV)
in8_input	Channel F voltage (mV)
in9_input	AIN1 voltage (mV)
in10_input	AIN2 voltage (mV)
in1_min	12v input minimum voltage (mV)
in2_min	3.3V (VDD) input minimum voltage (mV)
in3_min	Channel A minimum voltage (mV)
in4_min	Channel B minimum voltage (mV)
in5_min	Channel C minimum voltage (mV)
in6_min	Channel D minimum voltage (mV)
in7_min	Channel E minimum voltage (mV)
in8_min	Channel F minimum voltage (mV)
in9_min	AIN1 minimum voltage (mV)
in10_min	AIN2 minimum voltage (mV)
in1_max	12v input maximum voltage (mV)
in2_max	3.3V (VDD) input maximum voltage (mV)
in3_max	Channel A maximum voltage (mV)
in4_max	Channel B maximum voltage (mV)
in5_max	Channel C maximum voltage (mV)
in6_max	Channel D maximum voltage (mV)
in7_max	Channel E maximum voltage (mV)
in8_max	Channel F maximum voltage (mV)
in9_max	AIN1 maximum voltage (mV)
in10_max	AIN2 maximum voltage (mV)
in1_lcrit	12v input critical minimum voltage (mV)
in2_lcrit	3.3V (VDD) input critical minimum voltage (mV)
in3_lcrit	Channel A critical minimum voltage (mV)
in4_lcrit	Channel B critical minimum voltage (mV)
in5_lcrit	Channel C critical minimum voltage (mV)
in6_lcrit	Channel D critical minimum voltage (mV)
in7_lcrit	Channel E critical minimum voltage (mV)
in8_lcrit	Channel F critical minimum voltage (mV)
in9_lcrit	AIN1 critical minimum voltage (mV)
in10_lcrit	AIN2 critical minimum voltage (mV)
in1_crit	12v input critical maximum voltage (mV)
in2_crit	3.3V (VDD) input critical maximum voltage (mV)
in3_crit	Channel A critical maximum voltage (mV)
in4_crit	Channel B critical maximum voltage (mV)
in5_crit	Channel C critical maximum voltage (mV)
in6_crit	Channel D critical maximum voltage (mV)
in7_crit	Channel E critical maximum voltage (mV)
in8_crit	Channel F critical maximum voltage (mV)

Continued on next page

Table 17 – continued from previous page

in9_crit	AIN1 critical maximum voltage (mV)
in10_crit	AIN2 critical maximum voltage (mV)
in1_crit_alarm	12v input critical alarm
in2_crit_alarm	3.3V (VDD) input critical alarm
in3_crit_alarm	Channel A critical alarm
in4_crit_alarm	Channel B critical alarm
in5_crit_alarm	Channel C critical alarm
in6_crit_alarm	Channel D critical alarm
in7_crit_alarm	Channel E critical alarm
in8_crit_alarm	Channel F critical alarm
in9_crit_alarm	AIN1 critical alarm
in10_crit_alarm	AIN2 critical alarm
temp1_input	Chip temperature
temp1_min	Mimimum chip temperature
temp1_max	Maximum chip temperature
temp1_crit	Critical chip temperature
temp1_crit_alarm	Temperature critical alarm

7.137 Kernel driver smsc47b397

Supported chips:

- SMSC LPC47B397-NC
- SMSC SCH5307-NS
- SMSC SCH5317

  Prefix: 'smsc47b397'

  Addresses scanned: none, address read from Super I/O config space

  Datasheet: In this file

Authors:

- Mark M. Hoffman <mhoffman@lightlink.com>
- Utilitek Systems, Inc.

November 23, 2004

The following specification describes the SMSC LPC47B397-NC[1] sensor chip (for which there is no public datasheet available). This document was provided by Craig Kelly (In-Store Broadcast Network) and edited/corrected by Mark M. Hoffman <mhoffman@lightlink.com>.

[1] And SMSC SCH5307-NS and SCH5317, which have different device IDs but are otherwise compatible.

7.137.1 Methods for detecting the HP SIO and reading the thermal data on a dc7100

The thermal information on the dc7100 is contained in the SIO Hardware Monitor (HWM). The information is accessed through an index/data pair. The index/data pair is located at the HWM Base Address + 0 and the HWM Base Address + 1. The HWM Base address can be obtained from Logical Device 8, registers 0x60 (MSB) and 0x61 (LSB). Currently we are using 0x480 for the HWM Base Address and 0x480 and 0x481 for the index/data pair.

Reading temperature information. The temperature information is located in the following registers:

Temp1	0x25	(Currently, this reflects the CPU temp on all systems).
Temp2	0x26	
Temp3	0x27	
Temp4	0x80	

Programming Example The following is an example of how to read the HWM temperature registers:

```
MOV     DX,480H
MOV     AX,25H
OUT     DX,AL
MOV     DX,481H
IN      AL,DX
```

AL contains the data in hex, the temperature in Celsius is the decimal equivalent.

Ex: If AL contains 0x2A, the temperature is 42 degrees C.

Reading tach information. The fan speed information is located in the following registers:

Tach1	0x28	0x29	(Currently, this reflects the CPU fan speed on all systems).
Tach2	0x2A	0x2B	
Tach3	0x2C	0x2D	
Tach4	0x2E	0x2F	

Important: Reading the tach LSB locks the tach MSB. The LSB Must be read first.

7.137.2 How to convert the tach reading to RPM

The tach reading (TCount) is given by: (Tach MSB * 256) + (Tach LSB) The SIO counts the number of 90kHz (11.111us) pulses per revolution. RPM = 60/(TCount * 11.111us)

Example:

```
Reg 0x28 = 0x9B
Reg 0x29 = 0x08
```

TCount = 0x89B = 2203

RPM = 60 / (2203 * 11.11111 E-6) = 2451 RPM

Obtaining the SIO version.

7.137.3 Configuration Sequence

To program the configuration registers, the following sequence must be followed: 1. Enter Configuration Mode 2. Configure the Configuration Registers 3. Exit Configuration Mode.

Enter Configuration Mode

To place the chip into the Configuration State The config key (0x55) is written to the CONFIG PORT (0x2E).

Configuration Mode

In configuration mode, the INDEX PORT is located at the CONFIG PORT address and the DATA PORT is at INDEX PORT address + 1.

The desired configuration registers are accessed in two steps:

a. Write the index of the Logical Device Number Configuration Register (i.e., 0x07) to the INDEX PORT and then write the number of the desired logical device to the DATA PORT.

b. Write the address of the desired configuration register within the logical device to the INDEX PORT and then write or read the config- uration register through the DATA PORT.

Note: If accessing the Global Configuration Registers, step (a) is not required.

Exit Configuration Mode

To exit the Configuration State the write 0xAA to the CONFIG PORT (0x2E). The chip returns to the RUN State. (This is important).

Programming Example

The following is an example of how to read the SIO Device ID located at 0x20:

> ; ENTER CONFIGURATION MODE MOV DX,02EH MOV AX,055H OUT DX,AL ; GLOBAL CONFIGURATION REGISTER MOV DX,02EH MOV AL,20H OUT DX,AL ; READ THE DATA MOV DX,02FH IN AL,DX ; EXIT CONFIGURATION MODE MOV DX,02EH MOV AX,0AAH OUT DX,AL

The registers of interest for identifying the SIO on the dc7100 are Device ID (0x20) and Device Rev (0x21).

The Device ID will read 0x6F (0x81 for SCH5307-NS, and 0x85 for SCH5317) The Device Rev currently reads 0x01

7.137.4 Obtaining the HWM Base Address

The following is an example of how to read the HWM Base Address located in Logical Device 8:

```
; ENTER CONFIGURATION MODE
MOV     DX,02EH
MOV     AX,055H
OUT     DX,AL
; CONFIGURE REGISTER CRE0,
; LOGICAL DEVICE 8
MOV     DX,02EH
MOV     AL,07H
OUT     DX,AL ;Point to LD# Config Reg
MOV     DX,02FH
MOV     AL, 08H
OUT     DX,AL;Point to Logical Device 8
;
MOV     DX,02EH
MOV     AL,60H
OUT     DX,AL   ; Point to HWM Base Addr MSB
MOV     DX,02FH
IN      AL,DX   ; Get MSB of HWM Base Addr
; EXIT CONFIGURATION MODE
MOV     DX,02EH
MOV     AX,0AAH
OUT     DX,AL
```

7.138 Kernel driver smsc47m192

Supported chips:

- SMSC LPC47M192, LPC47M15x, LPC47M292 and LPC47M997

  Prefix: 'smsc47m192'

  Addresses scanned: I2C 0x2c - 0x2d

  Datasheet: The datasheet for LPC47M192 is publicly available from

  http://www.smsc.com/

  The LPC47M15x, LPC47M292 and LPC47M997 are compatible for hardware monitoring.

Author:

- Hartmut Rick <linux@rick.claranet.de>
- Special thanks to Jean Delvare for careful checking of the code and many helpful comments and suggestions.

7.138.1 Description

This driver implements support for the hardware sensor capabilities of the SMSC LPC47M192 and compatible Super-I/O chips.

These chips support 3 temperature channels and 8 voltage inputs as well as CPU voltage VID input.

They do also have fan monitoring and control capabilities, but the these features are accessed via ISA bus and are not supported by this driver. Use the 'smsc47m1' driver for fan monitoring and control.

Voltages and temperatures are measured by an 8-bit ADC, the resolution of the temperatures is 1 bit per degree C. Voltages are scaled such that the nominal voltage corresponds to 192 counts, i.e. 3/4 of the full range. Thus the available range for each voltage channel is 0V ⋯255/192*(nominal voltage), the resolution is 1 bit per (nominal voltage)/192. Both voltage and temperature values are scaled by 1000, the sys files show voltages in mV and temperatures in units of 0.001 degC.

The +12V analog voltage input channel (in4_input) is multiplexed with bit 4 of the encoded CPU voltage. This means that you either get a +12V voltage measurement or a 5 bit CPU VID, but not both. The default setting is to use the pin as 12V input, and use only 4 bit VID. This driver assumes that the information in the configuration register is correct, i.e. that the BIOS has updated the configuration if the motherboard has this input wired to VID4.

The temperature and voltage readings are updated once every 1.5 seconds. Reading them more often repeats the same values.

7.138.2 sysfs interface

in0_input	+2.5V voltage input
in1_input	CPU voltage input (nominal 2.25V)
in2_input	+3.3V voltage input
in3_input	+5V voltage input
in4_input	+12V voltage input (may be missing if used as VID4)
in5_input	Vcc voltage input (nominal 3.3V) This is the supply voltage of the sensor chip itself.
in6_input	+1.5V voltage input
in7_input	+1.8V voltage input
in[0-7]_min, in[0-7]_max	lower and upper alarm thresholds for in[0-7]_input reading All voltages are read and written in mV.
in[0-7]_alarm	alarm flags for voltage inputs These files read '1' in case of alarm, '0' otherwise.
temp1_input	chip temperature measured by on-chip diode
temp[2-3]_input	temperature measured by external diodes (one of these would typically be wired to the diode inside the CPU)
temp[1-3]_min, temp[1-3]_max	lower and upper alarm thresholds for temperatures
temp[1-3]_offset	temperature offset registers The chip adds the offsets stored in these registers to the corresponding temperature readings. Note that temp1 and temp2 offsets share the same register, they cannot both be different from zero at the same time. Writing a non-zero number to one of them will reset the other offset to zero. All temperatures and offsets are read and written in units of 0.001 degC.
temp[1-3]_alarm	alarm flags for temperature inputs, '1' in case of alarm, '0' otherwise.
temp[2-3]_input_fault	diode fault flags for temperature inputs 2 and 3. A fault is detected if the two pins for the corresponding sensor are open or shorted, or any of the two is shorted to ground or Vcc. '1' indicates a diode fault.
cpu0_vid	CPU voltage as received from the CPU
vrm	CPU VID standard used for decoding CPU voltage

The *_min, *_max, *_offset and vrm files can be read and written, all others are read-only.

7.139 Kernel driver smsc47m1

Supported chips:

- SMSC LPC47B27x, LPC47M112, LPC47M10x, LPC47M13x, LPC47M14x, LPC47M15x and LPC47M192

  Addresses scanned: none, address read from Super I/O config space

  Prefix: 'smsc47m1'

  Datasheets:

  http://www.smsc.com/media/Downloads_Public/Data_Sheets/47b272.pdf

  http://www.smsc.com/media/Downloads_Public/Data_Sheets/47m10x.pdf

  http://www.smsc.com/media/Downloads_Public/Data_Sheets/47m112.pdf

  http://www.smsc.com/

- SMSC LPC47M292

  Addresses scanned: none, address read from Super I/O config space

  Prefix: 'smsc47m2'

  Datasheet: Not public

- SMSC LPC47M997

  Addresses scanned: none, address read from Super I/O config space

  Prefix: 'smsc47m1'

  Datasheet: none

Authors:

- Mark D. Studebaker <mdsxyz123@yahoo.com>,
- With assistance from Bruce Allen <ballen@uwm.edu>, and his fan.c program:
  - http://www.lsc-group.phys.uwm.edu/%7Eballen/driver/
- Gabriele Gorla <gorlik@yahoo.com>,
- Jean Delvare <jdelvare@suse.de>

7.139.1 Description

The Standard Microsystems Corporation (SMSC) 47M1xx Super I/O chips contain monitoring and PWM control circuitry for two fans.

The LPC47M15x, LPC47M192 and LPC47M292 chips contain a full 'hardware monitoring block' in addition to the fan monitoring and control. The hardware monitoring block is not supported by this driver, use the smsc47m192 driver for that.

No documentation is available for the 47M997, but it has the same device ID as the 47M15x and 47M192 chips and seems to be compatible.

Fan rotation speeds are reported in RPM (rotations per minute). An alarm is triggered if the rotation speed has dropped below a programmable limit. Fan readings can be divided by a programmable divider (1, 2, 4 or 8) to give the readings more range or accuracy. Not all RPM values can accurately be represented, so some rounding is done. With a divider of 2, the lowest representable value is around 2600 RPM.

PWM values are from 0 to 255.

If an alarm triggers, it will remain triggered until the hardware register is read at least once. This means that the cause for the alarm may already have disappeared! Note that in the current implementation, all hardware registers are read whenever any data is read (unless it is less than 1.5 seconds since the last update). This means that you can easily miss once-only alarms.

The lm_sensors project gratefully acknowledges the support of Intel in the development of this driver.

7.140 Kernel driver tc654

Supported chips:

- Microchip TC654 and TC655

  Prefix: 'tc654'Datasheet: http://ww1.microchip.com/downloads/en/DeviceDoc/20001734C

Authors:

- Chris Packham <chris.packham@alliedtelesis.co.nz>
- Masahiko Iwamoto <iwamoto@allied-telesis.co.jp>

7.140.1 Description

This driver implements support for the Microchip TC654 and TC655.

The TC654 uses the 2-wire interface compatible with the SMBUS 2.0 specification. The TC654 has two (2) inputs for measuring fan RPM and one (1) PWM output which can be used for fan control.

7.140.2 Configuration Notes

Ordinarily the pwm1_mode ABI is used for controlling the pwm output mode. However, for this chip the output is always pwm, and the pwm1_mode determines if the pwm output is controlled via the pwm1 value or via the Vin analog input.

Setting pwm1_mode to 1 will cause the pwm output to be driven based on the pwm1 value. Setting pwm1_mode to 0 will cause the pwm output to be driven based on the Vin input.

7.141 Kernel driver tc74

Supported chips:

- Microchip TC74

  Prefix: 'tc74'

  Datasheet: Publicly available at Microchip website.

7.141.1 Description

Driver supports the above part.

The tc74 has an 8-bit sensor, with 1 degree centigrade resolution and +- 2 degrees centigrade accuracy.

7.141.2 Notes

Currently entering low power standby mode is not supported.

7.142 Kernel driver thmc50

Supported chips:

- Analog Devices ADM1022

  Prefix: 'adm1022'

  Addresses scanned: I2C 0x2c - 0x2e

  Datasheet: http://www.analog.com/en/prod/0,2877,ADM1022,00.html

- Texas Instruments THMC50

  Prefix: ‘thmc50’

  Addresses scanned: I2C 0x2c - 0x2e

  Datasheet: http://www.ti.com/

Author: Krzysztof Helt <krzysztof.h1@wp.pl>

This driver was derived from the 2.4 kernel thmc50.c source file.

Credits:

thmc50.c (2.4 kernel):

- Frodo Looijaard <frodol@dds.nl>
- Philip Edelbrock <phil@netroedge.com>

7.142.1 Module Parameters

- **adm1022_temp3: short array** List of adapter,address pairs to force chips into ADM1022 mode with second remote temperature. This does not work for original THMC50 chips.

7.142.2 Description

The THMC50 implements: an internal temperature sensor, support for an external diode-type temperature sensor (compatible w/ the diode sensor inside many processors), and a controllable fan/analog_out DAC. For the temperature sensors, limits can be set through the appropriate Overtemperature Shutdown register and Hysteresis register. Each value can be set and read to half-degree accuracy. An alarm is issued (usually to a connected LM78) when the temperature gets higher then the Overtemperature Shutdown value; it stays on until the temperature falls below the Hysteresis value. All temperatures are in degrees Celsius, and are guaranteed within a range of -55 to +125 degrees.

The THMC50 only updates its values each 1.5 seconds; reading it more often will do no harm, but will return ‘old’ values.

The THMC50 is usually used in combination with LM78-like chips, to measure the temperature of the processor(s).

The ADM1022 works the same as THMC50 but it is faster (5 Hz instead of 1 Hz for THMC50). It can be also put in a new mode to handle additional remote temperature sensor. The driver use the mode set by BIOS by default.

In case the BIOS is broken and the mode is set incorrectly, you can force the mode with additional remote temperature with adm1022_temp3 parameter. A typical symptom of wrong setting is a fan forced to full speed.

7.142.3 Driver Features

The driver provides up to three temperatures:

temp1

- internal

temp2

- remote

temp3

- 2nd remote only for ADM1022

pwm1

- fan speed (0 = stop, 255 = full)

pwm1_mode

- always 0 (DC mode)

The value of 0 for pwm1 also forces FAN_OFF signal from the chip, so it stops fans even if the value 0 into the ANALOG_OUT register does not.

The driver was tested on Compaq AP550 with two ADM1022 chips (one works in the temp3 mode), five temperature readings and two fans.

7.143 Kernel driver tmp102

Supported chips:

- Texas Instruments TMP102

  Prefix: 'tmp102'

  Addresses scanned: none

  Datasheet: http://focus.ti.com/docs/prod/folders/print/tmp102.html

Author:

Steven King <sfking@fdwdc.com>

7.143.1 Description

The Texas Instruments TMP102 implements one temperature sensor. Limits can be set through the Overtemperature Shutdown register and Hysteresis register. The sensor is accurate to 0.5 degree over the range of -25 to +85 C, and to 1.0 degree from -40 to +125 C. Resolution of the sensor is 0.0625 degree. The operating temperature has a minimum of -55 C and a maximum of +150 C.

The TMP102 has a programmable update rate that can select between 8, 4, 1, and 0.5 Hz. (Currently the driver only supports the default of 4 Hz).

The driver provides the common sysfs-interface for temperatures (see Documentation/hwmon/sysfs-interface.rst under Temperatures).

7.144 Kernel driver tmp103

Supported chips:

- Texas Instruments TMP103

  Prefix: 'tmp103'

  Addresses scanned: none

  Product info and datasheet: http://www.ti.com/product/tmp103

Author:

Heiko Schocher <hs@denx.de>

7.144.1 Description

The TMP103 is a digital output temperature sensor in a four-ball wafer chip-scale package (WCSP). The TMP103 is capable of reading temperatures to a resolution of 1°C. The TMP103 is specified for operation over a temperature range of –40°C to +125°C.

Resolution: 8 Bits Accuracy: ±1°C Typ (–10°C to +100°C)

The driver provides the common sysfs-interface for temperatures (see Documentation/hwmon/sysfs-interface.rst under Temperatures).

Please refer how to instantiate this driver: Documentation/i2c/instantiating-devices.rst

7.145 Kernel driver tmp108

Supported chips:

- Texas Instruments TMP108

  Prefix: 'tmp108'

  Addresses scanned: none

  Datasheet: http://www.ti.com/product/tmp108

Author:

John Muir <john@jmuir.com>

7.145.1 Description

The Texas Instruments TMP108 implements one temperature sensor. An alert pin can be set when temperatures exceed minimum or maximum values plus or minus a hysteresis value. (This driver does not support interrupts for the alert pin, and the device runs in comparator mode.)

The sensor is accurate to 0.75C over the range of -25 to +85 C, and to 1.0 degree from -40 to +125 C. Resolution of the sensor is 0.0625 degree. The operating temperature has a minimum of -55 C and a maximum of +150 C. Hysteresis values can be set to 0, 1, 2, or 4C.

The TMP108 has a programmable update rate that can select between 8, 4, 1, and 0.5 Hz.

By default the TMP108 reads the temperature continuously. To conserve power, the TMP108 has a one-shot mode where the device is normally shut-down. When a one shot is requested the temperature is read, the result can be retrieved, and then the device is shut down automatically. (This driver only supports continuous mode.)

The driver provides the common sysfs-interface for temperatures (see Documentation/hwmon/sysfs-interface.rst under Temperatures).

7.146 Kernel driver tmp401

Supported chips:

- Texas Instruments TMP401

  Prefix: 'tmp401'

  Addresses scanned: I2C 0x4c

  Datasheet: http://focus.ti.com/docs/prod/folders/print/tmp401.html

- Texas Instruments TMP411

  Prefix: 'tmp411'

  Addresses scanned: I2C 0x4c, 0x4d, 0x4e

  Datasheet: http://focus.ti.com/docs/prod/folders/print/tmp411.html

- Texas Instruments TMP431

  Prefix: 'tmp431'

  Addresses scanned: I2C 0x4c, 0x4d

  Datasheet: http://focus.ti.com/docs/prod/folders/print/tmp431.html

- Texas Instruments TMP432

  Prefix: 'tmp432'

  Addresses scanned: I2C 0x4c, 0x4d

  Datasheet: http://focus.ti.com/docs/prod/folders/print/tmp432.html

- Texas Instruments TMP435

  Prefix: ‘tmp435’

  Addresses scanned: I2C 0x48 - 0x4f

  Datasheet: http://focus.ti.com/docs/prod/folders/print/tmp435.html

- Texas Instruments TMP461

  Prefix: ‘tmp461’

  Datasheet: http://www.ti.com/product/tmp461

Authors:

- Hans de Goede <hdegoede@redhat.com>
- Andre Prendel <andre.prendel@gmx.de>

7.146.1 Description

This driver implements support for Texas Instruments TMP401, TMP411, TMP431, TMP432, TMP435, and TMP461 chips. These chips implement one or two remote and one local temperature sensors. Temperature is measured in degrees Celsius. Resolution of the remote sensor is 0.0625 degree. Local sensor resolution can be set to 0.5, 0.25, 0.125 or 0.0625 degree (not supported by the driver so far, so using the default resolution of 0.5 degree).

The driver provides the common sysfs-interface for temperatures (see Documentation/hwmon/sysfs-interface.rst under Temperatures).

The TMP411 and TMP431 chips are compatible with TMP401. TMP411 provides some additional features.

- Minimum and Maximum temperature measured since power-on, chip-reset

  Exported via sysfs attributes tempX_lowest and tempX_highest.

- Reset of historical minimum/maximum temperature measurements

  Exported via sysfs attribute temp_reset_history. Writing 1 to this file triggers a reset.

TMP432 is compatible with TMP401 and TMP431. It supports two external temperature sensors.

TMP461 is compatible with TMP401. It supports offset correction that is applied to the remote sensor.

- Sensor offset values are temperature values

  Exported via sysfs attribute tempX_offset

7.147 Kernel driver tmp421

Supported chips:

- Texas Instruments TMP421

  Prefix: 'tmp421'

  Addresses scanned: I2C 0x2a, 0x4c, 0x4d, 0x4e and 0x4f

  Datasheet: http://focus.ti.com/docs/prod/folders/print/tmp421.html

- Texas Instruments TMP422

  Prefix: 'tmp422'

  Addresses scanned: I2C 0x4c, 0x4d, 0x4e and 0x4f

  Datasheet: http://focus.ti.com/docs/prod/folders/print/tmp421.html

- Texas Instruments TMP423

  Prefix: 'tmp423'

  Addresses scanned: I2C 0x4c and 0x4d

  Datasheet: http://focus.ti.com/docs/prod/folders/print/tmp421.html

- Texas Instruments TMP441

  Prefix: 'tmp441'

  Addresses scanned: I2C 0x2a, 0x4c, 0x4d, 0x4e and 0x4f

  Datasheet: http://www.ti.com/product/tmp441

- Texas Instruments TMP442

  Prefix: 'tmp442'

  Addresses scanned: I2C 0x4c and 0x4d

  Datasheet: http://www.ti.com/product/tmp442

Authors:

Andre Prendel <andre.prendel@gmx.de>

7.147.1 Description

This driver implements support for Texas Instruments TMP421, TMP422, TMP423, TMP441, and TMP442 temperature sensor chips. These chips implement one local and up to one (TMP421, TMP441), up to two (TMP422, TMP442) or up to three (TMP423) remote sensors. Temperature is measured in degrees Celsius. The chips are wired over I2C/SMBus and specified over a temperature range of -40 to +125 degrees Celsius. Resolution for both the local and remote channels is 0.0625 degree C.

The chips support only temperature measurement. The driver exports the temperature values via the following sysfs files:

temp[1-4]_input

temp[2-4]_fault

7.148 Kernel driver tmp513

Supported chips:

- Texas Instruments TMP512

  Prefix: 'tmp512'

  Datasheet: http://www.ti.com/lit/ds/symlink/tmp512.pdf

- Texas Instruments TMP513

  Prefix: 'tmp513'

  Datasheet: http://www.ti.com/lit/ds/symlink/tmp513.pdf

Authors:

Eric Tremblay <etremblay@distech-controls.com>

7.148.1 Description

This driver implements support for Texas Instruments TMP512, and TMP513. The TMP512 (dual-channel) and TMP513 (triple-channel) are system monitors that include remote sensors, a local temperature sensor, and a high-side current shunt monitor. These system monitors have the capability of measuring remote temperatures, on-chip temperatures, and system voltage/power/current consumption.

The temperatures are measured in degrees Celsius with a range of -40 to + 125 degrees with a resolution of 0.0625 degree C.

For hysteresis value, only the first channel is writable. Writing to it will affect all other values since each channels are sharing the same hysteresis value. The hysteresis is in degrees Celsius with a range of 0 to 127.5 degrees with a resolution of 0.5 degree.

The driver exports the temperature values via the following sysfs files:

temp[1-4]_input

temp[1-4]_crit

temp[1-4]_crit_alarm

temp[1-4]_crit_hyst

The driver read the shunt voltage from the chip and convert it to current. The readable range depends on the "ti,pga-gain" property (default to 8) and the shunt resistor value. The value resolution will be equal to 10uV/Rshunt.

The driver exports the shunt currents values via the following sysFs files:

curr1_input

curr1_lcrit

curr1_lcrit_alarm

curr1_crit

curr1_crit_alarm

The bus voltage range is read from the chip with a resolution of 4mV. The chip can be configurable in two different range (32V or 16V) using the ti,bus-range-microvolt property in the devicetree.

The driver exports the bus voltage values via the following sysFs files:

in0_input

in0_lcrit

in0_lcrit_alarm

in0_crit

in0_crit_alarm

The bus power and bus currents range and resolution depends on the calibration register value. Those values are calculate by the hardware using those formulas:

Current = (ShuntVoltage * CalibrationRegister) / 4096 Power = (Current * BusVoltage) / 5000

The driver exports the bus current and bus power values via the following sysFs files:

curr2_input

power1_input

power1_crit

power1_crit_alarm

The calibration process follow the procedure of the datasheet (without overflow) and depend on the shunt resistor value and the pga_gain value.

7.149 Kernel driver tps40422

Supported chips:

- TI TPS40422

  Prefix: 'tps40422'

  Addresses scanned: -

  Datasheet: http://www.ti.com/lit/gpn/tps40422

Author: Zhu Laiwen <richard.zhu@nsn.com>

7.149.1 Description

This driver supports TI TPS40422 Dual-Output or Two-Phase Synchronous Buck Controller with PMBus

The driver is a client driver to the core PMBus driver. Please see Documentation/hwmon/pmbus.rst for details on PMBus client drivers.

7.149.2 Usage Notes

This driver does not auto-detect devices. You will have to instantiate the devices explicitly. Please see Documentation/i2c/instantiating-devices.rst for details.

7.149.3 Platform data support

The driver supports standard PMBus driver platform data.

7.149.4 Sysfs entries

The following attributes are supported.

in[1-2]_label	“vout[1-2]”
in[1-2]_input	Measured voltage. From READ_VOUT register.
in[1-2]_alarm	voltage alarm.
curr[1-2]_input	Measured current. From READ_IOUT register.
curr[1-2]_label	“iout[1-2]”
curr1_max	Maximum current. From IOUT_OC_WARN_LIMIT register.
curr1_crit	Critical maximum current. From IOUT_OC_FAULT_LIMIT register.
curr1_max_alarm	Current high alarm. From IOUT_OC_WARN_LIMIT status.
curr1_crit_alarm	Current critical high alarm. From IOUT_OC_FAULT status.
curr2_alarm	Current high alarm. From IOUT_OC_WARNING status.
temp1_input	Measured temperature. From READ_TEMPERATURE_2 register on page 0.
temp1_max	Maximum temperature. From OT_WARN_LIMIT register.
temp1_crit	Critical high temperature. From OT_FAULT_LIMIT register.
temp1_max_alarm	Chip temperature high alarm. Set by comparing READ_TEMPERATURE_2 on page 0 with OT_WARN_LIMIT if TEMP_OT_WARNING status is set.
temp1_crit_alarm	Chip temperature critical high alarm. Set by comparing READ_TEMPERATURE_2 on page 0 with OT_FAULT_LIMIT if TEMP_OT_FAULT status is set.
temp2_input	Measured temperature. From READ_TEMPERATURE_2 register on page 1.
temp2_alarm	Chip temperature alarm on page 1.

7.150 Kernel driver tps53679

Supported chips:

- Texas Instruments TPS53647

  Prefix: 'tps53647'

  Addresses scanned: -

  Datasheet: http://www.ti.com/lit/gpn/tps53647

- Texas Instruments TPS53667

  Prefix: 'tps53667'

  Addresses scanned: -

  Datasheet: http://www.ti.com/lit/gpn/TPS53667

- Texas Instruments TPS53679

  Prefix: 'tps53679'

  Addresses scanned: -

  Datasheet: http://www.ti.com/lit/gpn/TPS53679 (short version)

- Texas Instruments TPS53681

  Prefix: 'tps53681'

  Addresses scanned: -

  Datasheet: http://www.ti.com/lit/gpn/TPS53681

- Texas Instruments TPS53688

  Prefix: 'tps53688'

  Addresses scanned: -

  Datasheet: Available under NDA

Authors: Vadim Pasternak <vadimp@mellanox.com> Guenter Roeck <linux@roeck-us.net>

7.150.1 Description

Chips in this series are multi-phase step-down converters with one or two output channels and up to 8 phases per channel.

7.150.2 Usage Notes

This driver does not probe for PMBus devices. You will have to instantiate devices explicitly.

Example: the following commands will load the driver for an TPS53681 at address 0x60 on I2C bus #1:

```
# modprobe tps53679
# echo tps53681 0x60 > /sys/bus/i2c/devices/i2c-1/new_device
```

7.150.3 Sysfs attributes

in1_label	“vin”
in1_input	Measured input voltage.
in1_lcrit	Critical minimum input voltage TPS53679, TPS53681, TPS53688 only.
in1_lcrit_alarm	Input voltage critical low alarm. TPS53679, TPS53681, TPS53688 only.
in1_crit	Critical maximum input voltage.
in1_crit_alarm	Input voltage critical high alarm.
in[N]_label	“vout[1-2]” • TPS53647, TPS53667: N=2 • TPS53679, TPS53588: N=2,3
in[N]_input	Measured output voltage.
in[N]_lcrit	Critical minimum input voltage. TPS53679, TPS53681, TPS53688 only.
in[N]_lcrit_alarm	Critical minimum voltage alarm. TPS53679, TPS53681, TPS53688 only.
in[N]_alarm	Output voltage alarm. TPS53647, TPS53667 only.
in[N]_crit	Critical maximum output voltage. TPS53679, TPS53681, TPS53688 only.
in[N]_crit_alarm	Output voltage critical high alarm. TPS53679, TPS53681, TPS53688 only.
temp[N]_input	Measured temperature. • TPS53647, TPS53667: N=1 • TPS53679, TPS53681, TPS53588: N=1,2
temp[N]_max	Maximum temperature.
temp[N]_crit	Critical high temperature.
temp[N]_max_alarm	Temperature high alarm.
temp[N]_crit_alarm	Temperature critical high alarm.
power1_label	“pin” .
power1_input	Measured input power.

Continued on next page

Table 18 – continued from previous page

power[N]_label	“pout[1-2]”. • TPS53647, TPS53667: N=2 • TPS53679, TPS53681, TPS53588: N=2,3
power[N]_input	Measured output power.
curr1_label	“iin”.
curr1_input	Measured input current.
curr1_max	Maximum input current.
curr1_max_alarm	Input current high alarm.
curr1_crit	Critical input current.
curr1_crit_alarm	Input current critical alarm.
curr[N]_label	“iout[1-2]” or “iout1.[0-5]”. The first digit is the output channel, the second digit is the phase within the channel. Per-phase telemetry supported on TPS53681 only. • TPS53647, TPS53667: N=2 • TPS53679, TPS53588: N=2,3 • TPS53681: N=2-9
curr[N]_input	Measured output current.
curr[N]_max	Maximum output current.
curr[N]_crit	Critical high output current.
curr[N]_max_alarm	Output current high alarm.
curr[N]_crit_alarm	Output current critical high alarm. Limit and alarm attributes are only available for non-phase telemetry (iout1, iout2).

7.151 Kernel driver twl4030-madc

Supported chips:

- Texas Instruments TWL4030

Prefix: ‘twl4030-madc’

Authors: J Keerthy <j-keerthy@ti.com>

7.151.1 Description

The Texas Instruments TWL4030 is a Power Management and Audio Circuit. Among other things it contains a 10-bit A/D converter MADC. The converter has 16 channels which can be used in different modes.

See this table for the meaning of the different channels

Channel	Signal
0	Battery type(BTYPE)
1	BCI: Battery temperature (BTEMP)
2	GP analog input
3	GP analog input
4	GP analog input
5	GP analog input
6	GP analog input
7	GP analog input
8	BCI: VBUS voltage(VBUS)
9	Backup Battery voltage (VBKP)
10	BCI: Battery charger current (ICHG)
11	BCI: Battery charger voltage (VCHG)
12	BCI: Main battery voltage (VBAT)
13	Reserved
14	Reserved
15	VRUSB Supply/Speaker left/Speaker right polarization level

The Sysfs nodes will represent the voltage in the units of mV, the temperature channel shows the converted temperature in degree Celsius. The Battery charging current channel represents battery charging current in mA.

7.152 Kernel driver ucd9000

Supported chips:

- TI UCD90120, UCD90124, UCD90160, UCD90320, UCD9090, and UCD90910

  Prefixes: **'ucd90120'**, **'ucd90124'**, **'ucd90160'**, **'ucd90320'**, **'ucd9090'**, 'ucd90910'

  Addresses scanned: -

  Datasheets:

  - http://focus.ti.com/lit/ds/symlink/ucd90120.pdf
  - http://focus.ti.com/lit/ds/symlink/ucd90124.pdf
  - http://focus.ti.com/lit/ds/symlink/ucd90160.pdf
  - http://focus.ti.com/lit/ds/symlink/ucd90320.pdf
  - http://focus.ti.com/lit/ds/symlink/ucd9090.pdf

- http://focus.ti.com/lit/ds/symlink/ucd90910.pdf

Author: Guenter Roeck <linux@roeck-us.net>

7.152.1 Description

From datasheets:

The UCD90120 Power Supply Sequencer and System Health Monitor monitors and sequences up to 12 independent voltage rails. The device integrates a 12-bit ADC with a 2.5V internal reference for monitoring up to 13 power supply voltage, current, or temperature inputs.

The UCD90124 is a 12-rail PMBus/I2C addressable power-supply sequencer and system-health monitor. The device integrates a 12-bit ADC for monitoring up to 13 power-supply voltage, current, or temperature inputs. Twenty-six GPIO pins can be used for power supply enables, power-on reset signals, external interrupts, cascading, or other system functions. Twelve of these pins offer PWM functionality. Using these pins, the UCD90124 offers support for fan control, margining, and general-purpose PWM functions.

The UCD90160 is a 16-rail PMBus/I2C addressable power-supply sequencer and monitor. The device integrates a 12-bit ADC for monitoring up to 16 power-supply voltage inputs. Twenty-six GPIO pins can be used for power supply enables, power-on reset signals, external interrupts, cascading, or other system functions. Twelve of these pins offer PWM functionality. Using these pins, the UCD90160 offers support for margining, and general-purpose PWM functions.

The UCD90320 is a 32-rail PMBus/I2C addressable power-supply sequencer and monitor. The 24 integrated ADC channels (AMONx) monitor the power supply voltage, current, and temperature. Of the 84 GPIO pins, 8 can be used as digital monitors (DMONx), 32 to enable the power supply (ENx), 24 for margining (MARx), 16 for logical GPO, and 32 GPIs for cascading, and system function.

The UCD9090 is a 10-rail PMBus/I2C addressable power-supply sequencer and monitor. The device integrates a 12-bit ADC for monitoring up to 10 power-supply voltage inputs. Twenty-three GPIO pins can be used for power supply enables, power-on reset signals, external interrupts, cascading, or other system functions. Ten of these pins offer PWM functionality. Using these pins, the UCD9090 offers support for margining, and general-purpose PWM functions.

The UCD90910 is a ten-rail I2C / PMBus addressable power-supply sequencer and system-health monitor. The device integrates a 12-bit ADC for monitoring up to 13 power-supply voltage, current, or temperature inputs.

This driver is a client driver to the core PMBus driver. Please see Documentation/hwmon/pmbus.rst for details on PMBus client drivers.

7.152.2 Usage Notes

This driver does not auto-detect devices. You will have to instantiate the devices explicitly. Please see Documentation/i2c/instantiating-devices.rst for details.

7.152.3 Platform data support

The driver supports standard PMBus driver platform data. Please see Documentation/hwmon/pmbus.rst for details.

7.152.4 Sysfs entries

The following attributes are supported. Limits are read-write; all other attributes are read-only.

in[1-12]_label	"vout[1-12]".
in[1-12]_input	Measured voltage. From READ_VOUT register.
in[1-12]_min	Minimum Voltage. From VOUT_UV_WARN_LIMIT register.
in[1-12]_max	Maximum voltage. From VOUT_OV_WARN_LIMIT register.
in[1-12]_lcrit	Critical minimum Voltage. VOUT_UV_FAULT_LIMIT register.
in[1-12]_crit	Critical maximum voltage. From VOUT_OV_FAULT_LIMIT register.
in[1-12]_min_alarm	Voltage low alarm. From VOLTAGE_UV_WARNING status.
in[1-12]_max_alarm	Voltage high alarm. From VOLTAGE_OV_WARNING status.
in[1-12]_lcrit_alarm	Voltage critical low alarm. From VOLTAGE_UV_FAULT status.
in[1-12]_crit_alarm	Voltage critical high alarm. From VOLTAGE_OV_FAULT status.
curr[1-12]_label	"iout[1-12]".
curr[1-12]_input	Measured current. From READ_IOUT register.
curr[1-12]_max	Maximum current. From IOUT_OC_WARN_LIMIT register.
curr[1-12]_lcritter.	Critical minimum output current. From IOUT_UC_FAULT_LIMIT regis-
curr[1-12]_crit	Critical maximum current. From IOUT_OC_FAULT_LIMIT register.
curr[1-12]_max_alarm	Current high alarm. From IOUT_OC_WARNING status.
curr[1-12]_crit_alarm	Current critical high alarm. From IOUT_OC_FAULT status. For each attribute index, either voltage or current is reported, but not both. If voltage or current is reported depends on the chip configuration.
temp[1-2]_input	Measured temperatures. From READ_TEMPERATURE_1 and READ_TEMPERATURE_2 registers.
temp[1-2]_max	Maximum temperature. From OT_WARN_LIMIT register.
temp[1-2]_crit	Critical high temperature. From OT_FAULT_LIMIT register.
temp[1-2]_max_alarm	Temperature high alarm.
temp[1-2]_crit_alarm	Temperature critical high alarm.
fan[1-4]_input	Fan RPM.
fan[1-4]_alarm	Fan alarm.
fan[1-4]_fault	Fan fault. Fan attributes are only available on chips supporting fan control (UCD90124, UCD90910). Attribute files are created only for enabled fans. Note that even though UCD90910 supports up to 10 fans, only up to four fans are currently supported.

7.153 Kernel driver ucd9200

Supported chips:

- TI UCD9220, UCD9222, UCD9224, UCD9240, UCD9244, UCD9246, and UCD9248

  Prefixes: 'ucd9220', 'ucd9222', 'ucd9224', 'ucd9240', 'ucd9244', 'ucd9246', 'ucd9248'

  Addresses scanned: -

  Datasheets:

  - http://focus.ti.com/lit/ds/symlink/ucd9220.pdf
  - http://focus.ti.com/lit/ds/symlink/ucd9222.pdf
  - http://focus.ti.com/lit/ds/symlink/ucd9224.pdf
  - http://focus.ti.com/lit/ds/symlink/ucd9240.pdf
  - http://focus.ti.com/lit/ds/symlink/ucd9244.pdf
  - http://focus.ti.com/lit/ds/symlink/ucd9246.pdf
  - http://focus.ti.com/lit/ds/symlink/ucd9248.pdf

Author: Guenter Roeck <linux@roeck-us.net>

7.153.1 Description

[From datasheets] UCD9220, UCD9222, UCD9224, UCD9240, UCD9244, UCD9246, and UCD9248 are multi-rail, multi-phase synchronous buck digital PWM controllers designed for non-isolated DC/DC power applications. The devices integrate dedicated circuitry for DC/DC loop management with flash memory and a serial interface to support configuration, monitoring and management.

This driver is a client driver to the core PMBus driver. Please see Documentation/hwmon/pmbus.rst for details on PMBus client drivers.

7.153.2 Usage Notes

This driver does not auto-detect devices. You will have to instantiate the devices explicitly. Please see Documentation/i2c/instantiating-devices.rst for details.

7.153.3 Platform data support

The driver supports standard PMBus driver platform data. Please see Documentation/hwmon/pmbus.rst for details.

7.153.4 Sysfs entries

The following attributes are supported. Limits are read-write; all other attributes are read-only.

in1_label	“vin”.
in1_input	Measured voltage. From READ_VIN register.
in1_min	Minimum Voltage. From VIN_UV_WARN_LIMIT register.
in1_max	Maximum voltage. From VIN_OV_WARN_LIMIT register.
in1_lcrit	Critical minimum Voltage. VIN_UV_FAULT_LIMIT register.
in1_crit	Critical maximum voltage. From VIN_OV_FAULT_LIMIT register.
in1_min_alarm	Voltage low alarm. From VIN_UV_WARNING status.
in1_max_alarm	Voltage high alarm. From VIN_OV_WARNING status.
in1_lcrit_alarm	Voltage critical low alarm. From VIN_UV_FAULT status.
in1_crit_alarm	Voltage critical high alarm. From VIN_OV_FAULT status.
in[2-5]_label	“vout[1-4]”.
in[2-5]_input	Measured voltage. From READ_VOUT register.
in[2-5]_min	Minimum Voltage. From VOUT_UV_WARN_LIMIT register.
in[2-5]_max	Maximum voltage. From VOUT_OV_WARN_LIMIT register.
in[2-5]_lcrit	Critical minimum Voltage. VOUT_UV_FAULT_LIMIT register.
in[2-5]_crit	Critical maximum voltage. From VOUT_OV_FAULT_LIMIT register.
in[2-5]_min_alarm	Voltage low alarm. From VOLTAGE_UV_WARNING status.
in[2-5]_max_alarm	Voltage high alarm. From VOLTAGE_OV_WARNING status.
in[2-5]_lcrit_alarm	Voltage critical low alarm. From VOLTAGE_UV_FAULT status.
in[2-5]_crit_alarm	Voltage critical high alarm. From VOLTAGE_OV_FAULT status.
curr1_label	“iin”.

Continued on next page

Table 19 – continued from previous page

curr1_input	Measured current. From READ_IIN register.
curr[2-5]_label	“iout[1-4]” .
curr[2-5]_input	Measured current. From READ_IOUT register.
curr[2-5]_max	Maximum current. From IOUT_OC_WARN_LIMIT register.
curr[2-5]_lcrit	Critical minimum output current. From IOUT_UC_FAULT_LIMIT register.
curr[2-5]_crit	Critical maximum current. From IOUT_OC_FAULT_LIMIT register.
curr[2-5]_max_alarm	Current high alarm. From IOUT_OC_WARNING status.
curr[2-5]_crit_alarm	Current critical high alarm. From IOUT_OC_FAULT status.
power1_input	Measured input power. From READ_PIN register.
power1_label	“pin”
power[2-5]_input	Measured output power. From READ_POUT register.
power[2-5]_label	“pout[1-4]” The number of output voltage, current, and power attribute sets is determined by the number of enabled rails. See chip datasheets for details.
temp[1-5]_input	Measured temperatures. From READ_TEMPERATURE_1 and READ_TEMPERATURE_2 registers. temp1 is the chip internal temperature. temp[2-5] are rail temperatures. temp[2-5] attributes are only created for enabled rails. See chip datasheets for details.
temp[1-5]_max	Maximum temperature. From OT_WARN_LIMIT register.
temp[1-5]_crit	Critical high temperature. From OT_FAULT_LIMIT register.
temp[1-5]_max_alarm	Temperature high alarm.
temp[1-5]_crit_alarm	Temperature critical high alarm.
fan1_input	Fan RPM. ucd9240 only.
fan1_alarm	Fan alarm. ucd9240 only.
fan1_fault	Fan fault. ucd9240 only.

7.154 Kernel driver vexpress

Supported systems:

- ARM Ltd. Versatile Express platform

  Prefix: 'vexpress'

  Datasheets:

  - "Hardware Description" sections of the Technical Reference Manuals for the Versatile Express boards:
    - http://infocenter.arm.com/help/topic/com.arm.doc.subset.boards.express/index.html
  - Section "4.4.14. System Configuration registers" of the V2M-P1 TRM:
    - http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.dui0447-/index.html

Author: Pawel Moll

7.154.1 Description

Versatile Express platform (http://www.arm.com/versatileexpress/) is a reference & prototyping system for ARM Ltd. processors. It can be set up from a wide range of boards, each of them containing (apart of the main chip/FPGA) a number of microcontrollers responsible for platform configuration and control. Theses microcontrollers can also monitor the board and its environment by a number of internal and external sensors, providing information about power lines voltages and currents, board temperature and power usage. Some of them also calculate consumed energy and provide a cumulative use counter.

The configuration devices are _not_ memory mapped and must be accessed via a custom interface, abstracted by the "vexpress_config" API.

As these devices are non-discoverable, they must be described in a Device Tree passed to the kernel. Details of the DT binding for them can be found in Documentation/devicetree/bindings/hwmon/vexpress.txt.

7.155 Kernel driver via686a

Supported chips:

- Via VT82C686A, VT82C686B Southbridge Integrated Hardware Monitor

  Prefix: 'via686a'

  Addresses scanned: ISA in PCI-space encoded address

  Datasheet: On request through web form (http://www.via.com.tw/en/resources/download-center/)

Authors:

- Kyösti Mälkki <kmalkki@cc.hut.fi>,

- Mark D. Studebaker <mdsxyz123@yahoo.com>
- Bob Dougherty <bobd@stanford.edu>
- (Some conversion-factor data were contributed by
- Jonathan Teh Soon Yew <j.teh@iname.com>
- and Alex van Kaam <darkside@chello.nl>.)

7.155.1 Module Parameters

force_addr=0xaddr	Set the I/O base address. Useful for boards that don't set the address in the BIOS. Look for a BIOS upgrade before resorting to this. Does not do a PCI force; the via686a must still be present in lspci. Don't use this unless the driver complains that the base address is not set. Example: 'modprobe via686a force_addr=0x6000'

7.155.2 Description

The driver does not distinguish between the chips and reports all as a 686A.

The Via 686a southbridge has integrated hardware monitor functionality. It also has an I2C bus, but this driver only supports the hardware monitor. For the I2C bus driver, see <file:Documentation/i2c/busses/i2c-viapro.rst>

The Via 686a implements three temperature sensors, two fan rotation speed sensors, five voltage sensors and alarms.

Temperatures are measured in degrees Celsius. An alarm is triggered once when the Overtemperature Shutdown limit is crossed; it is triggered again as soon as it drops below the hysteresis value.

Fan rotation speeds are reported in RPM (rotations per minute). An alarm is triggered if the rotation speed has dropped below a programmable limit. Fan readings can be divided by a programmable divider (1, 2, 4 or 8) to give the readings more range or accuracy. Not all RPM values can accurately be represented, so some rounding is done. With a divider of 2, the lowest representable value is around 2600 RPM.

Voltage sensors (also known as IN sensors) report their values in volts. An alarm is triggered if the voltage has crossed a programmable minimum or maximum limit. Voltages are internally scalled, so each voltage channel has a different resolution and range.

If an alarm triggers, it will remain triggered until the hardware register is read at least once. This means that the cause for the alarm may already have disappeared! Note that in the current implementation, all hardware registers are read whenever any data is read (unless it is less than 1.5 seconds since the last update). This means that you can easily miss once-only alarms.

The driver only updates its values each 1.5 seconds; reading it more often will do no harm, but will return 'old' values.

7.155.3 Known Issues

This driver handles sensors integrated in some VIA south bridges. It is possible that a motherboard maker used a VT82C686A/B chip as part of a product design but was not interested in its hardware monitoring features, in which case the sensor inputs will not be wired. This is the case of the Asus K7V, A7V and A7V133 motherboards, to name only a few of them. So, if you need the force_addr parameter, and end up with values which don't seem to make any sense, don't look any further: your chip is simply not wired for hardware monitoring.

7.156 Kernel driver vt1211

Supported chips:

- VIA VT1211

  Prefix: 'vt1211'

  Addresses scanned: none, address read from Super-I/O config space

  Datasheet: Provided by VIA upon request and under NDA

Authors: Juerg Haefliger <juergh@gmail.com>

This driver is based on the driver for kernel 2.4 by Mark D. Studebaker and its port to kernel 2.6 by Lars Ekman.

Thanks to Joseph Chan and Fiona Gatt from VIA for providing documentation and technical support.

7.156.1 Module Parameters

- **uch_config: int** Override the BIOS default universal channel (UCH) configuration for channels 1-5. Legal values are in the range of 0-31. Bit 0 maps to UCH1, bit 1 maps to UCH2 and so on. Setting a bit to 1 enables the thermal input of that particular UCH and setting a bit to 0 enables the voltage input.
- **int_mode: int** Override the BIOS default temperature interrupt mode. The only possible value is 0 which forces interrupt mode 0. In this mode, any pending interrupt is cleared when the status register is read but is regenerated as long as the temperature stays above the hysteresis limit.

Be aware that overriding BIOS defaults might cause some unwanted side effects!

7.156.2 Description

The VIA VT1211 Super-I/O chip includes complete hardware monitoring capabilities. It monitors 2 dedicated temperature sensor inputs (temp1 and temp2), 1 dedicated voltage (in5) and 2 fans. Additionally, the chip implements 5 universal input channels (UCH1-5) that can be individually programmed to either monitor a voltage or a temperature.

This chip also provides manual and automatic control of fan speeds (according to the datasheet). The driver only supports automatic control since the manual mode doesn't seem to work as advertised in the datasheet. In fact I couldn't get manual mode to work at all! Be aware that automatic mode hasn't been tested very well (due to the fact that my EPIA M10000 doesn't have the fans connected to the PWM outputs of the VT1211 :-().

The following table shows the relationship between the vt1211 inputs and the sysfs nodes.

Sensor	Voltage Mode	Temp Mode	Default Use (from the datasheet)
Reading 1		temp1	Intel thermal diode
Reading 3		temp2	Internal thermal diode
UCH1/Reading2	in0	temp3	NTC type thermistor
UCH2	in1	temp4	+2.5V
UCH3	in2	temp5	VccP (processor core)
UCH4	in3	temp6	+5V
UCH5	in4	temp7	+12V
+3.3V	in5		Internal VCC (+3.3V)

7.156.3 Voltage Monitoring

Voltages are sampled by an 8-bit ADC with a LSB of ~10mV. The supported input range is thus from 0 to 2.60V. Voltage values outside of this range need external scaling resistors. This external scaling needs to be compensated for via compute lines in sensors.conf, like:

compute inx @*(1+R1/R2), @/(1+R1/R2)

The board level scaling resistors according to VIA's recommendation are as follows. And this is of course totally dependent on the actual board implementation :-) You will have to find documentation for your own motherboard and edit sensors.conf accordingly.

Voltage	R1	R2	Divider	Raw Value
+2.5V	2K	10K	1.2	2083 mV
VccP	—	—	1.0	1400 mV[1]
+5V	14K	10K	2.4	2083 mV
+12V	47K	10K	5.7	2105 mV
+3.3V (int)	2K	3.4K	1.588	3300 mV[2]
+3.3V (ext)	6.8K	10K	1.68	1964 mV

Each measured voltage has an associated low and high limit which triggers an alarm when crossed.

7.156.4 Temperature Monitoring

Temperatures are reported in millidegree Celsius. Each measured temperature has a high limit which triggers an alarm if crossed. There is an associated hysteresis value with each temperature below which the temperature has to drop before the alarm is cleared (this is only true for interrupt mode 0). The interrupt mode can be forced to 0 in case the BIOS doesn't do it automatically. See the 'Module Parameters' section for details.

All temperature channels except temp2 are external. Temp2 is the VT1211 internal thermal diode and the driver does all the scaling for temp2 and returns the temperature in millidegree Celsius. For the external channels temp1 and temp3-temp7, scaling depends on the board implementation and needs to be performed in userspace via sensors.conf.

Temp1 is an Intel-type thermal diode which requires the following formula to convert between sysfs readings and real temperatures:

compute temp1 (@-Offset)/Gain, (@*Gain)+Offset

According to the VIA VT1211 BIOS porting guide, the following gain and offset values should be used:

Diode Type	Offset	Gain
Intel CPU	88.638 65.000	0.9528 0.9686[3]
VIA C3 Ezra	83.869	0.9528
VIA C3 Ezra-T	73.869	0.9528

Temp3-temp7 support NTC thermistors. For these channels, the driver returns the voltages as seen at the individual pins of UCH1-UCH5. The voltage at the pin (Vpin) is formed by a voltage divider made of the thermistor (Rth) and a scaling resistor (Rs):

```
Vpin = 2200 * Rth / (Rs + Rth)   (2200 is the ADC max limit of 2200 mV)
```

The equation for the thermistor is as follows (google it if you want to know more about it):

```
Rth = Ro * exp(B * (1 / T - 1 / To))   (To is 298.15K (25C) and Ro is the
                                        nominal resistance at 25C)
```

Mingling the above two equations and assuming Rs = Ro and B = 3435 yields the following formula for sensors.conf:

[1] Depending on the CPU (1.4V is for a VIA C3 Nehemiah).

[2] R1 and R2 for 3.3V (int) are internal to the VT1211 chip and the driver performs the scaling and returns the properly scaled voltage value.

[3] This is the formula from the lm_sensors 2.10.0 sensors.conf file. I don't know where it comes from or how it was derived, it's just listed here for completeness.

```
compute tempx 1 / (1 / 298.15 - (` (2200 / @ - 1)) / 3435) - 273.15,
              2200 / (1 + (^ (3435 / 298.15 - 3435 / (273.15 + @))))
```

7.156.5 Fan Speed Control

The VT1211 provides 2 programmable PWM outputs to control the speeds of 2 fans. Writing a 2 to any of the two pwm[1-2]_enable sysfs nodes will put the PWM controller in automatic mode. There is only a single controller that controls both PWM outputs but each PWM output can be individually enabled and disabled.

Each PWM has 4 associated distinct output duty-cycles: full, high, low and off. Full and off are internally hard-wired to 255 (100%) and 0 (0%), respectively. High and low can be programmed via pwm[1-2]_auto_point[2-3]_pwm. Each PWM output can be associated with a different thermal input but - and here's the weird part - only one set of thermal thresholds exist that controls both PWMs output duty-cycles. The thermal thresholds are accessible via pwm[1-2]_auto_point[1-4]_temp. Note that even though there are 2 sets of 4 auto points each, they map to the same registers in the VT1211 and programming one set is sufficient (actually only the first set pwm1_auto_point[1-4]_temp is writable, the second set is read-only).

PWM Auto Point	PWM Output Duty-Cycle
pwm[1-2]_auto_point4_pwm	full speed duty-cycle (hard-wired to 255)
pwm[1-2]_auto_point3_pwm	high speed duty-cycle
pwm[1-2]_auto_point2_pwm	low speed duty-cycle
pwm[1-2]_auto_point1_pwm	off duty-cycle (hard-wired to 0)

Temp Auto Point	Thermal Threshold
pwm[1-2]_auto_point4_temp	full speed temp
pwm[1-2]_auto_point3_temp	high speed temp
pwm[1-2]_auto_point2_temp	low speed temp
pwm[1-2]_auto_point1_temp	off temp

Long story short, the controller implements the following algorithm to set the PWM output duty-cycle based on the input temperature:

Thermal Threshold	Output Duty-Cycle (Rising Temp)	Output Duty-Cycle (Falling Temp)
•	full speed duty-cycle	full speed duty-cycle
full speed temp		
•	high speed duty-cycle	full speed duty-cycle
high speed temp		
•	low speed duty-cycle	high speed duty-cycle
low speed temp		
•	off duty-cycle	low speed duty-cycle
off temp		

7.157 Kernel driver w83627ehf

Supported chips:

- Winbond W83627EHF/EHG (ISA access ONLY)

  Prefix: 'w83627ehf'

  Addresses scanned: ISA address retrieved from Super I/O registers

  Datasheet: not available

- Winbond W83627DHG

  Prefix: 'w83627dhg'

  Addresses scanned: ISA address retrieved from Super I/O registers

  Datasheet: not available

- Winbond W83627DHG-P

  Prefix: 'w83627dhg'

  Addresses scanned: ISA address retrieved from Super I/O registers

  Datasheet: not available

- Winbond W83627UHG

  Prefix: 'w83627uhg'

  Addresses scanned: ISA address retrieved from Super I/O registers

  Datasheet: available from www.nuvoton.com

- Winbond W83667HG

  Prefix: 'w83667hg'

  Addresses scanned: ISA address retrieved from Super I/O registers

Datasheet: not available

- Winbond W83667HG-B

  Prefix: 'w83667hg'

  Addresses scanned: ISA address retrieved from Super I/O registers

  Datasheet: Available from Nuvoton upon request

- Nuvoton NCT6775F/W83667HG-I

  Prefix: 'nct6775'

  Addresses scanned: ISA address retrieved from Super I/O registers

  Datasheet: Available from Nuvoton upon request

- Nuvoton NCT6776F

  Prefix: 'nct6776'

  Addresses scanned: ISA address retrieved from Super I/O registers

  Datasheet: Available from Nuvoton upon request

Authors:

- Jean Delvare <jdelvare@suse.de>
- Yuan Mu (Winbond)
- Rudolf Marek <r.marek@assembler.cz>
- David Hubbard <david.c.hubbard@gmail.com>
- Gong Jun <JGong@nuvoton.com>

7.157.1 Description

This driver implements support for the Winbond W83627EHF, W83627EHG, W83627DHG, W83627DHG-P, W83627UHG, W83667HG, W83667HG-B, W83667HG-I (NCT6775F), and NCT6776F super I/O chips. We will refer to them collectively as Winbond chips.

The chips implement 3 to 4 temperature sensors (9 for NCT6775F and NCT6776F), 2 to 5 fan rotation speed sensors, 8 to 10 analog voltage sensors, one VID (except for 627UHG), alarms with beep warnings (control unimplemented), and some automatic fan regulation strategies (plus manual fan control mode).

The temperature sensor sources on W82677HG-B, NCT6775F, and NCT6776F are configurable. temp4 and higher attributes are only reported if its temperature source differs from the temperature sources of the already reported temperature sensors. The configured source for each of the temperature sensors is provided in tempX_label.

Temperatures are measured in degrees Celsius and measurement resolution is 1 degC for temp1 and and 0.5 degC for temp2 and temp3. For temp4 and higher, resolution is 1 degC for W83667HG-B and 0.0 degC for NCT6775F and NCT6776F. An alarm is triggered when the temperature gets higher than high limit; it stays on

until the temperature falls below the hysteresis value. Alarms are only supported for temp1, temp2, and temp3.

Fan rotation speeds are reported in RPM (rotations per minute). An alarm is triggered if the rotation speed has dropped below a programmable limit. Fan readings can be divided by a programmable divider (1, 2, 4, 8, 16, 32, 64 or 128) to give the readings more range or accuracy. The driver sets the most suitable fan divisor itself. Some fans might not be present because they share pins with other functions.

Voltage sensors (also known as IN sensors) report their values in millivolts. An alarm is triggered if the voltage has crossed a programmable minimum or maximum limit.

The driver supports automatic fan control mode known as Thermal Cruise. In this mode, the chip attempts to keep the measured temperature in a predefined temperature range. If the temperature goes out of range, fan is driven slower/faster to reach the predefined range again.

The mode works for fan1-fan4. Mapping of temperatures to pwm outputs is as follows:

```
temp1 -> pwm1
temp2 -> pwm2
temp3 -> pwm3 (not on 627UHG)
prog  -> pwm4 (not on 667HG and 667HG-B; the programmable setting is not
               supported by the driver)
```

7.157.2 /sys files

name this is a standard hwmon device entry, it contains the name of the device (see the prefix in the list of supported devices at the top of this file)

pwm[1-4] this file stores PWM duty cycle or DC value (fan speed) in range:

0 (stop) to 255 (full)

pwm[1-4]_enable this file controls mode of fan/temperature control:

- 1 Manual mode, write to pwm file any value 0-255 (full speed)
- 2 “Thermal Cruise” mode
- 3 “Fan Speed Cruise” mode
- 4 “Smart Fan III” mode
- 5 “Smart Fan IV” mode

SmartFan III mode is not supported on NCT6776F.

SmartFan IV mode is configurable only if it was configured at system startup, and is only supported for W83677HG-B, NCT6775F, and NCT6776F. SmartFan IV operational parameters can not be configured at this time, and the various pwm attributes are not used in SmartFan IV mode. The attributes can be written to, which is useful if you plan to configure the system for a different pwm mode. However, the information returned when reading pwm attributes is unrelated to SmartFan IV operation.

pwm[1-4]_mode controls if output is PWM or DC level

- 0 DC output (0 - 12v)
- 1 PWM output

7.157.3 Thermal Cruise mode

If the temperature is in the range defined by:

pwm[1-4]_target set target temperature, unit millidegree Celsius (range 0 - 127000)

pwm[1-4]_tolerance tolerance, unit millidegree Celsius (range 0 - 15000)

there are no changes to fan speed. Once the temperature leaves the interval, fan speed increases (temp is higher) or decreases if lower than desired. There are defined steps and times, but not exported by the driver yet.

pwm[1-4]_min_output minimum fan speed (range 1 - 255), when the temperature is below defined range.

pwm[1-4]_stop_time how many milliseconds [ms] must elapse to switch corresponding fan off. (when the temperature was below defined range).

pwm[1-4]_start_output minimum fan speed (range 1 - 255) when spinning up

pwm[1-4]_step_output rate of fan speed change (1 - 255)

pwm[1-4]_stop_output minimum fan speed (range 1 - 255) when spinning down

pwm[1-4]_max_output maximum fan speed (range 1 - 255), when the temperature is above defined range.

Note: last six functions are influenced by other control bits, not yet exported by the driver, so a change might not have any effect.

7.157.4 Implementation Details

Future driver development should bear in mind that the following registers have different functions on the 627EHF and the 627DHG. Some registers also have different power-on default values, but BIOS should already be loading appropriate defaults. Note that bank selection must be performed as is currently done in the driver for all register addresses.

Register(s)	Meaning
0x49	only on DHG, selects temperature source for AUX fan, CPU fan0
0x4a	not completely documented for the EHF and the DHG documentation assigns different behavior to bits 7 and 6, including extending the temperature input selection to SmartFan I, not just SmartFan III. Testing on the EHF will reveal whether they are compatible or not.
0x58	Chip ID: 0xa1=EHF 0xc1=DHG
0x5e	only on DHG, has bits to enable "current mode" temperature detection and critical temperature protection
0x45b	only on EHF, bit 3, vin4 alarm (EHF supports 10 inputs, only 9 on DHG)
0x552	only on EHF, vin4
0x558	only on EHF, vin4 high limit
0x559	only on EHF, vin4 low limit
0x6b	only on DHG, SYS fan critical temperature
0x6c	only on DHG, CPU fan0 critical temperature
0x6d	only on DHG, AUX fan critical temperature
0x6e	only on DHG, CPU fan1 critical temperature
0x50-0x55 and 0x650-0x657	marked as: • "Test Register" for the EHF • "Reserved Register" for the DHG

The DHG also supports PECI, where the DHG queries Intel CPU temperatures, and the ICH8 southbridge gets that data via PECI from the DHG, so that the southbridge drives the fans. And the DHG supports SST, a one-wire serial bus.

The DHG-P has an additional automatic fan speed control mode named Smart Fan (TM) III+. This mode is not yet supported by the driver.

7.158 Kernel driver w83627hf

Supported chips:

- Winbond W83627HF (ISA accesses ONLY) Prefix: 'w83627hf' Addresses scanned: ISA address retrieved from Super I/O registers
- Winbond W83627THF Prefix: 'w83627thf' Addresses scanned: ISA address retrieved from Super I/O registers

- Winbond W83697HF Prefix: 'w83697hf' Addresses scanned: ISA address retrieved from Super I/O registers
- Winbond W83637HF Prefix: 'w83637hf' Addresses scanned: ISA address retrieved from Super I/O registers
- Winbond W83687THF Prefix: 'w83687thf' Addresses scanned: ISA address retrieved from Super I/O registers Datasheet: Provided by Winbond on request(http://www.winbond.com/hq/enu)

Authors: Frodo Looijaard <frodol@dds.nl>, Philip Edelbrock <phil@netroedge.com>, Mark Studebaker <mdsxyz123@yahoo.com>, Bernhard C. Schrenk <clemy@clemy.org>

7.158.1 Module Parameters

- force_i2c: int Initialize the I2C address of the sensors
- init: int (default is 1) Use 'init=0' to bypass initializing the chip. Try this if your computer crashes when you load the module.

7.158.2 Description

This driver implements support for ISA accesses only for the Winbond W83627HF, W83627THF, W83697HF and W83637HF Super I/O chips. We will refer to them collectively as Winbond chips.

This driver supports ISA accesses, which should be more reliable than i2c accesses. Also, for Tyan boards which contain both a Super I/O chip and a second i2c-only Winbond chip (often a W83782D), using this driver will avoid i2c address conflicts and complex initialization that were required in the w83781d driver.

If you really want i2c accesses for these Super I/O chips, use the w83781d driver. However this is not the preferred method now that this ISA driver has been developed.

The w83627_HF_ uses pins 110-106 as VID0-VID4. The w83627_THF_ uses the same pins as GPIO[0:4]. Technically, the w83627_THF_ does not support a VID reading. However the two chips have the identical 128 pin package. So, it is possible or even likely for a w83627thf to have the VID signals routed to these pins despite their not being labeled for that purpose. Therefore, the w83627thf driver interprets these as VID. If the VID on your board doesn't work, first see doc/vid in the lm_sensors package[1]. If that still doesn't help, you may just ignore the bogus VID reading with no harm done.

For further information on this driver see the w83781d driver documentation.

[1] http://www.lm-sensors.org/browser/lm-sensors/trunk/doc/vid

7.158.3 Forcing the address

The driver used to have a module parameter named force_addr, which could be used to force the base I/O address of the hardware monitoring block. This was meant as a workaround for mainboards with a broken BIOS. This module parameter is gone for technical reasons. If you need this feature, you can obtain the same result by using the isaset tool (part of lm-sensors) before loading the driver:

Enter the Super I/O config space:

```
isaset -y -f 0x2e 0x87
isaset -y -f 0x2e 0x87
```

Select the hwmon logical device:

```
isaset -y 0x2e 0x2f 0x07 0x0b
```

Set the base I/O address (to 0x290 in this example):

```
isaset -y 0x2e 0x2f 0x60 0x02
isaset -y 0x2e 0x2f 0x61 0x90
```

Exit the Super-I/O config space:

```
isaset -y -f 0x2e 0xaa
```

The above sequence assumes a Super-I/O config space at 0x2e/0x2f, but 0x4e/0x4f is also possible.

7.158.4 Voltage pin mapping

Here is a summary of the voltage pin mapping for the W83627THF. This can be useful to convert data provided by board manufacturers into working libsensors configuration statements:

- W83627THF

Pin	Name	Register	Sysfs attribute
100	CPUVCORE	20h	in0
99	VIN0	21h	in1
98	VIN1	22h	in2
97	VIN2	24h	in4
114	AVCC	23h	in3
61	5VSB	50h (bank 5)	in7
74	VBAT	51h (bank 5)	in8

For other supported devices, you’ ll have to take the hard path and look up the information in the datasheet yourself (and then add it to this document please.)

7.159 Kernel driver w83773g

Supported chips:

- Nuvoton W83773G

  Prefix: 'w83773g'

  Addresses scanned: I2C 0x4c and 0x4d

  Datasheet: https://www.nuvoton.com/resource-files/W83773G_SG_DatasheetV1_2.pdf

Authors:

Lei YU <mine260309@gmail.com>

7.159.1 Description

This driver implements support for Nuvoton W83773G temperature sensor chip. This chip implements one local and two remote sensors. The chip also features offsets for the two remote sensors which get added to the input readings. The chip does all the scaling by itself and the driver therefore reports true temperatures that don' t need any user-space adjustments. Temperature is measured in degrees Celsius. The chip is wired over I2C/SMBus and specified over a temperature range of -40 to +125 degrees Celsius (for local sensor) and -40 to +127 degrees Celsius (for remote sensors). Resolution for both the local and remote channels is 0.125 degree C.

The chip supports only temperature measurement. The driver exports the temperature values via the following sysfs files:

temp[1-3]_input, temp[2-3]_fault, temp[2-3]_offset, update_interval

7.160 Kernel driver w83781d

Supported chips:

- Winbond W83781D

  Prefix: 'w83781d'

  Addresses scanned: I2C 0x28 - 0x2f, ISA 0x290 (8 I/O ports)

  Datasheet: http://www.winbond-usa.com/products/winbond_products/pdfs/PCIC/w83781d.pdf

- Winbond W83782D

  Prefix: 'w83782d'

  Addresses scanned: I2C 0x28 - 0x2f, ISA 0x290 (8 I/O ports)

  Datasheet: http://www.winbond.com

- Winbond W83783S

  Prefix: 'w83783s'

  Addresses scanned: I2C 0x2d

  Datasheet: http://www.winbond-usa.com/products/winbond_products/pdfs/PCIC/w83783s.pdf

- Asus AS99127F

  Prefix: 'as99127f'

  Addresses scanned: I2C 0x28 - 0x2f

  Datasheet: Unavailable from Asus

Authors:

- Frodo Looijaard <frodol@dds.nl>,
- Philip Edelbrock <phil@netroedge.com>,
- Mark Studebaker <mdsxyz123@yahoo.com>

7.160.1 Module parameters

- **init int** (default 1)

  Use 'init=0' to bypass initializing the chip. Try this if your computer crashes when you load the module.

- **reset int** (default 0) The driver used to reset the chip on load, but does no more. Use 'reset=1' to restore the old behavior. Report if you need to do this.

force_subclients=bus,caddr,saddr,saddr This is used to force the i2c addresses for subclients of a certain chip. Typical usage is force_subclients=0,0x2d,0x4a,0x4b to force the subclients of chip 0x2d on bus 0 to i2c addresses 0x4a and 0x4b. This parameter is useful for certain Tyan boards.

7.160.2 Description

This driver implements support for the Winbond W83781D, W83782D, W83783S chips, and the Asus AS99127F chips. We will refer to them collectively as W8378* chips.

There is quite some difference between these chips, but they are similar enough that it was sensible to put them together in one driver. The Asus chips are similar to an I2C-only W83782D.

Chip	#vin	#fanin	#pwm	#temp	wchipid	vendid	i2c	ISA
as99127f	7	3	0	3	0x31	0x12c3	yes	no
as99127f rev.2 (type_name = as99127f)					0x31	0x5ca3	yes	no
w83781d	7	3	0	3	0x10-1	0x5ca3	yes	yes
w83782d	9	3	2-4	3	0x30	0x5ca3	yes	yes
w83783s	5-6	3	2	1-2	0x40	0x5ca3	yes	no

Detection of these chips can sometimes be foiled because they can be in an internal state that allows no clean access. If you know the address of the chip, use a 'force' parameter; this will put them into a more well-behaved state first.

The W8378* implements temperature sensors (three on the W83781D and W83782D, two on the W83783S), three fan rotation speed sensors, voltage sensors (seven on the W83781D, nine on the W83782D and six on the W83783S), VID lines, alarms with beep warnings, and some miscellaneous stuff.

Temperatures are measured in degrees Celsius. There is always one main temperature sensor, and one (W83783S) or two (W83781D and W83782D) other sensors. An alarm is triggered for the main sensor once when the Overtemperature Shutdown limit is crossed; it is triggered again as soon as it drops below the Hysteresis value. A more useful behavior can be found by setting the Hysteresis value to +127 degrees Celsius; in this case, alarms are issued during all the time when the actual temperature is above the Overtemperature Shutdown value. The driver sets the hysteresis value for temp1 to 127 at initialization.

For the other temperature sensor(s), an alarm is triggered when the temperature gets higher then the Overtemperature Shutdown value; it stays on until the temperature falls below the Hysteresis value. But on the W83781D, there is only one alarm that functions for both other sensors! Temperatures are guaranteed within a range of -55 to +125 degrees. The main temperature sensors has a resolution of 1 degree; the other sensor(s) of 0.5 degree.

Fan rotation speeds are reported in RPM (rotations per minute). An alarm is triggered if the rotation speed has dropped below a programmable limit. Fan readings can be divided by a programmable divider (1, 2, 4 or 8 for the W83781D; 1, 2, 4, 8, 16, 32, 64 or 128 for the others) to give the readings more range or accuracy. Not all RPM values can accurately be represented, so some rounding is done. With a divider of 2, the lowest representable value is around 2600 RPM.

Voltage sensors (also known as IN sensors) report their values in volts. An alarm is triggered if the voltage has crossed a programmable minimum or maximum limit. Note that minimum in this case always means 'closest to zero'; this is important for negative voltage measurements. All voltage inputs can measure voltages between 0 and 4.08 volts, with a resolution of 0.016 volt.

The VID lines encode the core voltage value: the voltage level your processor should work with. This is hardcoded by the mainboard and/or processor itself. It is a value in volts. When it is unconnected, you will often find the value 3.50 V here.

The W83782D and W83783S temperature conversion machine understands about several kinds of temperature probes. You can program the so-called beta value in the sensor files. '1' is the PII/Celeron diode, '2' is the TN3904 transistor, and 3435 the default thermistor value. Other values are (not yet) supported.

In addition to the alarms described above, there is a CHAS alarm on the chips which triggers if your computer case is open.

When an alarm goes off, you can be warned by a beeping signal through your computer speaker. It is possible to enable all beeping globally, or only the beeping for some alarms.

Individual alarm and beep bits:

0x000001	in0
0x000002	in1
0x000004	in2
0x000008	in3
0x000010	temp1
0x000020	temp2 (+temp3 on W83781D)
0x000040	fan1
0x000080	fan2
0x000100	in4
0x000200	in5
0x000400	in6
0x000800	fan3
0x001000	chassis
0x002000	temp3 (W83782D only)
0x010000	in7 (W83782D only)
0x020000	in8 (W83782D only)

If an alarm triggers, it will remain triggered until the hardware register is read at least once. This means that the cause for the alarm may already have disappeared! Note that in the current implementation, all hardware registers are read whenever any data is read (unless it is less than 1.5 seconds since the last update). This means that you can easily miss once-only alarms.

The chips only update values each 1.5 seconds; reading them more often will do no harm, but will return 'old' values.

7.160.3 AS99127F PROBLEMS

The as99127f support was developed without the benefit of a datasheet. In most cases it is treated as a w83781d (although revision 2 of the AS99127F looks more like a w83782d). This support will be BETA until a datasheet is released. One user has reported problems with fans stopping occasionally.

Note that the individual beep bits are inverted from the other chips. The driver now takes care of this so that user-space applications don't have to know about it.

Known problems:

- Problems with diode/thermistor settings (supported?)
- One user reports fans stopping under high server load.
- Revision 2 seems to have 2 PWM registers but we don't know how to handle them. More details below.

These will not be fixed unless we get a datasheet. If you have problems, please lobby Asus to release a datasheet. Unfortunately several others have without success. Please do not send mail to us asking for better as99127f support. We have done the best we can without a datasheet. Please do not send mail to the author or the sensors group asking for a datasheet or ideas on how to convince Asus. We can't help.

7.160.4 NOTES

783s has no in1 so that in[2-6] are compatible with the 781d/782d.

783s pin is programmable for -5V or temp1; defaults to -5V, no control in driver so temp1 doesn't work.

782d and 783s datasheets differ on which is pwm1 and which is pwm2. We chose to follow 782d.

782d and 783s pin is programmable for fan3 input or pwm2 output; defaults to fan3 input. If pwm2 is enabled (with echo 255 1 > pwm2), then fan3 will report 0.

782d has pwm1-2 for ISA, pwm1-4 for i2c. (pwm3-4 share pins with the ISA pins)

7.160.5 Data sheet updates

- **PWM clock registers:**
  - 000: master / 512
  - 001: master / 1024
  - 010: master / 2048
  - 011: master / 4096
  - 100: master / 8192

7.160.6 Answers from Winbond tech support

```
>
> 1) In the W83781D data sheet section 7.2 last paragraph, it talks about
>    reprogramming the R-T table if the Beta of the thermistor is not
>    3435K. The R-T table is described briefly in section 8.20.
>    What formulas do I use to program a new R-T table for a given Beta?
>

We are sorry that the calculation for R-T table value is
confidential. If you have another Beta value of thermistor, we can help
to calculate the R-T table for you. But you should give us real R-T
Table which can be gotten by thermistor vendor. Therefore we will calculate
them and obtain 32-byte data, and you can fill the 32-byte data to the
register in Bank0.CR51 of W83781D.
```

(continues on next page)

(continued from previous page)

```
> 2) In the W83782D data sheet, it mentions that pins 38, 39, and 40 are
>    programmable to be either thermistor or Pentium II diode inputs.
>    How do I program them for diode inputs? I can't find any register
>    to program these to be diode inputs.

You may program Bank0 CR[5Dh] and CR[59h] registers.

============================== ============== ============== ============
      CR[5Dh]                  bit 1(VTIN1)    bit 2(VTIN2)   bit 3(VTIN3)

              thermistor                0              0              0
      diode                       1              1              1


(error) CR[59h]                bit 4(VTIN1)    bit 2(VTIN2)   bit 3(VTIN3)
(right) CR[59h]                bit 4(VTIN1)    bit 5(VTIN2)   bit 6(VTIN3)

      PII thermal diode          1              1              1
      2N3904  diode              0              0              0
============================== ============== ============== ============
```

7.160.7 Asus Clones

We have no datasheets for the Asus clones (AS99127F and ASB100 Bach). Here are some very useful information that were given to us by Alex Van Kaam about how to detect these chips, and how to read their values. He also gives advice for another Asus chipset, the Mozart-2 (which we don't support yet). Thanks Alex!

I reworded some parts and added personal comments.

Detection

AS99127F rev.1, AS99127F rev.2 and ASB100: - I2C address range: 0x29 - 0x2F - If register 0x58 holds 0x31 then we have an Asus (either ASB100 or AS99127F) - Which one depends on register 0x4F (manufacturer ID):

- 0x06 or 0x94: ASB100
- 0x12 or 0xC3: AS99127F rev.1
- 0x5C or 0xA3: AS99127F rev.2

Note that 0x5CA3 is Winbond's ID (WEC), which let us think Asus get their AS99127F rev.2 direct from Winbond. The other codes mean ATT and DVC, respectively. ATT could stand for Asustek something (although it would be very badly chosen IMHO), I don't know what DVC could stand for. Maybe these codes simply aren't meant to be decoded that way.

Mozart-2: - I2C address: 0x77 - If register 0x58 holds 0x56 or 0x10 then we have a Mozart-2 - Of the Mozart there are 3 types:

- 0x58=0x56, 0x4E=0x94, 0x4F=0x36: Asus ASM58 Mozart-2
- 0x58=0x56, 0x4E=0x94, 0x4F=0x06: Asus AS2K129R Mozart-2

- 0x58=0x10, 0x4E=0x5C, 0x4F=0xA3: Asus ??? Mozart-2

You can handle all 3 the exact same way :)

Temperature sensors

ASB100:

- sensor 1: register 0x27
- sensor 2 & 3 are the 2 LM75's on the SMBus
- sensor 4: register 0x17

Remark:

I noticed that on Intel boards sensor 2 is used for the CPU and 4 is ignored/stuck, on AMD boards sensor 4 is the CPU and sensor 2 is either ignored or a socket temperature.

AS99127F (rev.1 and 2 alike):

- sensor 1: register 0x27
- sensor 2 & 3 are the 2 LM75's on the SMBus

Remark:

Register 0x5b is suspected to be temperature type selector. Bit 1 would control temp1, bit 3 temp2 and bit 5 temp3.

Mozart-2:

- sensor 1: register 0x27
- sensor 2: register 0x13

Fan sensors

ASB100, AS99127F (rev.1 and 2 alike):

- 3 fans, identical to the W83781D

Mozart-2:

- 2 fans only, 1350000/RPM/div
- fan 1: register 0x28, divisor on register 0xA1 (bits 4-5)
- fan 2: register 0x29, divisor on register 0xA1 (bits 6-7)

Voltages

This is where there is a difference between AS99127F rev.1 and 2.

Remark:

> The difference is similar to the difference between W83781D and W83782D.

ASB100:

- in0=r(0x20)*0.016
- in1=r(0x21)*0.016
- in2=r(0x22)*0.016
- in3=r(0x23)*0.016*1.68
- in4=r(0x24)*0.016*3.8
- in5=r(0x25)*(-0.016)*3.97
- in6=r(0x26)*(-0.016)*1.666

AS99127F rev.1:

- in0=r(0x20)*0.016
- in1=r(0x21)*0.016
- in2=r(0x22)*0.016
- in3=r(0x23)*0.016*1.68
- in4=r(0x24)*0.016*3.8
- in5=r(0x25)*(-0.016)*3.97
- in6=r(0x26)*(-0.016)*1.503

AS99127F rev.2:

- in0=r(0x20)*0.016
- in1=r(0x21)*0.016
- in2=r(0x22)*0.016
- in3=r(0x23)*0.016*1.68
- in4=r(0x24)*0.016*3.8
- in5=(r(0x25)*0.016-3.6)*5.14+3.6
- in6=(r(0x26)*0.016-3.6)*3.14+3.6

Mozart-2:

- in0=r(0x20)*0.016
- in1=255
- in2=r(0x22)*0.016
- in3=r(0x23)*0.016*1.68

- in4=r(0x24)*0.016*4
- in5=255
- in6=255

PWM

- Additional info about PWM on the AS99127F (may apply to other Asus chips as well) by Jean Delvare as of 2004-04-09:

AS99127F revision 2 seems to have two PWM registers at 0x59 and 0x5A, and a temperature sensor type selector at 0x5B (which basically means that they swapped registers 0x59 and 0x5B when you compare with Winbond chips). Revision 1 of the chip also has the temperature sensor type selector at 0x5B, but PWM registers have no effect.

We don't know exactly how the temperature sensor type selection works. Looks like bits 1-0 are for temp1, bits 3-2 for temp2 and bits 5-4 for temp3, although it is possible that only the most significant bit matters each time. So far, values other than 0 always broke the readings.

PWM registers seem to be split in two parts: bit 7 is a mode selector, while the other bits seem to define a value or threshold.

When bit 7 is clear, bits 6-0 seem to hold a threshold value. If the value is below a given limit, the fan runs at low speed. If the value is above the limit, the fan runs at full speed. We have no clue as to what the limit represents. Note that there seem to be some inertia in this mode, speed changes may need some time to trigger. Also, an hysteresis mechanism is suspected since walking through all the values increasingly and then decreasingly led to slightly different limits.

When bit 7 is set, bits 3-0 seem to hold a threshold value, while bits 6-4 would not be significant. If the value is below a given limit, the fan runs at full speed, while if it is above the limit it runs at low speed (so this is the contrary of the other mode, in a way). Here again, we don't know what the limit is supposed to represent.

One remarkable thing is that the fans would only have two or three different speeds (transitional states left apart), not a whole range as you usually get with PWM.

As a conclusion, you can write 0x00 or 0x8F to the PWM registers to make fans run at low speed, and 0x7F or 0x80 to make them run at full speed.

Please contact us if you can figure out how it is supposed to work. As long as we don't know more, the w83781d driver doesn't handle PWM on AS99127F chips at all.

- Additional info about PWM on the AS99127F rev.1 by Hector Martin:

I've been fiddling around with the (in)famous 0x59 register and found out the following values do work as a form of coarse pwm:

0x80

- seems to turn fans off after some time(1-2 minutes)…might be some form of auto-fan-control based on temp? hmm (Qfan? this mobo is an old ASUS, it isn't marketed as Qfan. Maybe some beta pre-attempt at Qfan that was dropped at the BIOS)

0x81

- off

0x82

- slightly “on-ner” than off, but my fans do not get to move. I can hear the high-pitched PWM sound that motors give off at too-low-pwm.

0x83

- now they do move. Estimate about 70% speed or so.

0x84-0x8f

- full on

Changing the high nibble doesn’t seem to do much except the high bit (0x80) must be set for PWM to work, else the current pwm doesn’t seem to change.

My mobo is an ASUS A7V266-E. This behavior is similar to what I got with speedfan under Windows, where 0-15% would be off, 15-2x% (can’t remember the exact value) would be 70% and higher would be full on.

- Additional info about PWM on the AS99127F rev.1 from lm-sensors ticket #2350:

I conducted some experiment on Asus P3B-F motherboard with AS99127F (Ver. 1).

I confirm that 0x59 register control the CPU_Fan Header on this motherboard, and 0x5a register control PWR_Fan.

In order to reduce the dependency of specific fan, the measurement is conducted with a digital scope without fan connected. I found out that P3B-F actually output variable DC voltage on fan header center pin, looks like PWM is filtered on this motherboard.

Here are some of measurements:

0x80	20 mV
0x81	20 mV
0x82	232 mV
0x83	1.2 V
0x84	2.31 V
0x85	3.44 V
0x86	4.62 V
0x87	5.81 V
0x88	7.01 V
9x89	8.22 V
0x8a	9.42 V
0x8b	10.6 V
0x8c	11.9 V
0x8d	12.4 V
0x8e	12.4 V
0x8f	12.4 V

7.161 Kernel driver w83791d

Supported chips:

- Winbond W83791D

  Prefix: 'w83791d'

  Addresses scanned: I2C 0x2c - 0x2f

  Datasheet: http://www.winbond-usa.com/products/winbond_products/pdfs/PCIC/W83791D_W83791Gb.pdf

Author: Charles Spirakis <bezaur@gmail.com>

This driver was derived from the w83781d.c and w83792d.c source files.

Credits:

w83781d.c:

- Frodo Looijaard <frodol@dds.nl>,
- Philip Edelbrock <phil@netroedge.com>,
- Mark Studebaker <mdsxyz123@yahoo.com>

w83792d.c:

- Shane Huang (Winbond),
- Rudolf Marek <r.marek@assembler.cz>

Additional contributors:

- Sven Anders <anders@anduras.de>
- Marc Hulsman <m.hulsman@tudelft.nl>

7.161.1 Module Parameters

- **init boolean** (default 0)

  Use 'init=1' to have the driver do extra software initializations. The default behavior is to do the minimum initialization possible and depend on the BIOS to properly setup the chip. If you know you have a w83791d and you're having problems, try init=1 before trying reset=1.

- **reset boolean** (default 0)

  Use 'reset=1'to reset the chip (via index 0x40, bit 7). The default behavior is no chip reset to preserve BIOS settings.

- **force_subclients=bus,caddr,saddr,saddr** This is used to force the i2c addresses for subclients of a certain chip. Example usage is force_subclients=0,0x2f,0x4a,0x4b to force the subclients of chip 0x2f on bus 0 to i2c addresses 0x4a and 0x4b.

7.161.2 Description

This driver implements support for the Winbond W83791D chip. The W83791G chip appears to be the same as the W83791D but is lead free.

Detection of the chip can sometimes be foiled because it can be in an internal state that allows no clean access (Bank with ID register is not currently selected). If you know the address of the chip, use a 'force' parameter; this will put it into a more well-behaved state first.

The driver implements three temperature sensors, ten voltage sensors, five fan rotation speed sensors and manual PWM control of each fan.

Temperatures are measured in degrees Celsius and measurement resolution is 1 degC for temp1 and 0.5 degC for temp2 and temp3. An alarm is triggered when the temperature gets higher than the Overtemperature Shutdown value; it stays on until the temperature falls below the Hysteresis value.

Voltage sensors (also known as IN sensors) report their values in millivolts. An alarm is triggered if the voltage has crossed a programmable minimum or maximum limit.

Fan rotation speeds are reported in RPM (rotations per minute). An alarm is triggered if the rotation speed has dropped below a programmable limit. Fan readings can be divided by a programmable divider (1, 2, 4, 8, 16, 32, 64 or 128 for all fans) to give the readings more range or accuracy.

Each fan controlled is controlled by PWM. The PWM duty cycle can be read and set for each fan separately. Valid values range from 0 (stop) to 255 (full). PWM 1-3 support Thermal Cruise mode, in which the PWMs are automatically regulated to keep respectively temp 1-3 at a certain target temperature. See below for the description of the sysfs-interface.

The w83791d has a global bit used to enable beeping from the speaker when an alarm is triggered as well as a bitmask to enable or disable the beep for specific alarms. You need both the global beep enable bit and the corresponding beep bit to be on for a triggered alarm to sound a beep.

The sysfs interface to the global enable is via the sysfs beep_enable file. This file is used for both legacy and new code.

The sysfs interface to the beep bitmask has migrated from the original legacy method of a single sysfs beep_mask file to a newer method using multiple *_beep files as described in Documentation/hwmon/sysfs-interface.rst.

A similar change has occurred for the bitmap corresponding to the alarms. The original legacy method used a single sysfs alarms file containing a bitmap of triggered alarms. The newer method uses multiple sysfs *_alarm files (again following the pattern described in sysfs-interface).

Since both methods read and write the underlying hardware, they can be used interchangeably and changes in one will automatically be reflected by the other. If you use the legacy bitmask method, your user-space code is responsible for handling the fact that the alarms and beep_mask bitmaps are not the same (see the table below).

NOTE: All new code should be written to use the newer sysfs-interface specification as that avoids bitmap problems and is the preferred interface going forward.

The driver reads the hardware chip values at most once every three seconds. User mode code requesting values more often will receive cached values.

7.161.3 /sys files

The sysfs-interface is documented in the 'sysfs-interface' file. Only chip-specific options are documented here.

pwm[1-3]_enable	this file controls mode of fan/temperature control for fan 1-3. Fan/PWM 4-5 only support manual mode. • 1 Manual mode • 2 Thermal Cruise mode • 3 Fan Speed Cruise mode (no further support)
temp[1-3]_target	defines the target temperature for Thermal Cruise mode. Unit: millidegree Celsius RW
temp[1-3]_tolerance	temperature tolerance for Thermal Cruise mode. Specifies an interval around the target temperature in which the fan speed is not changed. Unit: millidegree Celsius RW

7.161.4 Alarms bitmap vs. beep_mask bitmask

For legacy code using the alarms and beep_mask files:

Signal	Alarms	beep_mask	Obs
in0 (VCORE)	0x000001	0x000001	
in1 (VINR0)	0x000002	0x002000	<== mismatch
in2 (+3.3VIN)	0x000004	0x000004	
in3 (5VDD)	0x000008	0x000008	
in4 (+12VIN)	0x000100	0x000100	
in5 (-12VIN)	0x000200	0x000200	
in6 (-5VIN)	0x000400	0x000400	
in7 (VSB)	0x080000	0x010000	<== mismatch
in8 (VBAT)	0x100000	0x020000	<== mismatch
in9 (VINR1)	0x004000	0x004000	
temp1	0x000010	0x000010	
temp2	0x000020	0x000020	
temp3	0x002000	0x000002	<== mismatch
fan1	0x000040	0x000040	
fan2	0x000080	0x000080	
fan3	0x000800	0x000800	
fan4	0x200000	0x200000	
fan5	0x400000	0x400000	
tart1	0x010000	0x040000	<== mismatch
tart2	0x020000	0x080000	<== mismatch
tart3	0x040000	0x100000	<== mismatch
case_open	0x001000	0x001000	
global_enable	•	0x800000	(modified via beep_enable)

7.162 Kernel driver w83792d

Supported chips:

- Winbond W83792D

  Prefix: 'w83792d'

  Addresses scanned: I2C 0x2c - 0x2f

  Datasheet: http://www.winbond.com.tw

Author: Shane Huang (Winbond) Updated: Roger Lucas

7.162.1 Module Parameters

- **init int** (default 1)

  Use 'init=0' to bypass initializing the chip. Try this if your computer crashes when you load the module.

- **force_subclients=bus,caddr,saddr,saddr** This is used to force the i2c addresses for subclients of a certain chip. Example usage is force_subclients=0,0x2f,0x4a,0x4b to force the subclients of chip 0x2f on bus 0 to i2c addresses 0x4a and 0x4b.

7.162.2 Description

This driver implements support for the Winbond W83792AD/D.

Detection of the chip can sometimes be foiled because it can be in an internal state that allows no clean access (Bank with ID register is not currently selected). If you know the address of the chip, use a 'force' parameter; this will put it into a more well-behaved state first.

The driver implements three temperature sensors, seven fan rotation speed sensors, nine voltage sensors, and two automatic fan regulation strategies called: Smart Fan I (Thermal Cruise mode) and Smart Fan II.

The driver also implements up to seven fan control outputs: pwm1-7. Pwm1-7 can be configured to PWM output or Analogue DC output via their associated pwmX_mode. Outputs pwm4 through pwm7 may or may not be present depending on how the W83792AD/D was configured by the BIOS.

Automatic fan control mode is possible only for fan1-fan3.

For all pwmX outputs, a value of 0 means minimum fan speed and a value of 255 means maximum fan speed.

Temperatures are measured in degrees Celsius and measurement resolution is 1 degC for temp1 and 0.5 degC for temp2 and temp3. An alarm is triggered when the temperature gets higher than the Overtemperature Shutdown value; it stays on until the temperature falls below the Hysteresis value.

Fan rotation speeds are reported in RPM (rotations per minute). An alarm is triggered if the rotation speed has dropped below a programmable limit. Fan readings can be divided by a programmable divider (1, 2, 4, 8, 16, 32, 64 or 128) to give the readings more range or accuracy.

Voltage sensors (also known as IN sensors) report their values in millivolts. An alarm is triggered if the voltage has crossed a programmable minimum or maximum limit.

Alarms are provided as output from "realtime status register". Following bits are defined:

bit	alarm on
0	in0
1	in1
2	temp1
3	temp2
4	temp3
5	fan1
6	fan2
7	fan3
8	in2
9	in3
10	in4
11	in5
12	in6
13	VID change
14	chassis
15	fan7
16	tart1
17	tart2
18	tart3
19	in7
20	in8
21	fan4
22	fan5
23	fan6

Tart will be asserted while target temperature cannot be achieved after 3 minutes of full speed rotation of corresponding fan.

In addition to the alarms described above, there is a CHAS alarm on the chips which triggers if your computer case is open (This one is latched, contrary to realtime alarms).

The chips only update values each 3 seconds; reading them more often will do no harm, but will return 'old' values.

7.162.3 W83792D PROBLEMS

Known problems:

- This driver is only for Winbond W83792D C version device, there are also some motherboards with B version W83792D device. The calculation method to in6-in7(measured value, limits) is a little different between C and B version. C or B version can be identified by CR[0x49h].
- The function of vid and vrm has not been finished, because I' m NOT very familiar with them. Adding support is welcome.
- The function of chassis open detection needs more tests.
- If you have ASUS server board and chip was not found: Then you will need to upgrade to latest (or beta) BIOS. If it does not help please contact

us.

7.162.4 Fan control

7.162.5 Manual mode

Works as expected. You just need to specify desired PWM/DC value (fan speed) in appropriate pwm# file.

7.162.6 Thermal cruise

In this mode, W83792D provides the Smart Fan system to automatically control fan speed to keep the temperatures of CPU and the system within specific range. At first a wanted temperature and interval must be set. This is done via thermal_cruise# file. The tolerance# file serves to create T +- tolerance interval. The fan speed will be lowered as long as the current temperature remains below the thermal_cruise# +- tolerance# value. Once the temperature exceeds the high limit (T+tolerance), the fan will be turned on with a specific speed set by pwm# and automatically controlled its PWM duty cycle with the temperature varying. Three conditions may occur:

(1) If the temperature still exceeds the high limit, PWM duty cycle will increase slowly.

(2) If the temperature goes below the high limit, but still above the low limit (T-tolerance), the fan speed will be fixed at the current speed because the temperature is in the target range.

(3) If the temperature goes below the low limit, PWM duty cycle will decrease slowly to 0 or a preset stop value until the temperature exceeds the low limit. (The preset stop value handling is not yet implemented in driver)

7.162.7 Smart Fan II

W83792D also provides a special mode for fan. Four temperature points are available. When related temperature sensors detects the temperature in preset temperature region (sf2_point@_fan# +- tolerance#) it will cause fans to run on programmed value from sf2_level@_fan#. You need to set four temperatures for each fan.

7.162.8 /sys files

pwm[1-7]

- this file stores PWM duty cycle or DC value (fan speed) in range:

  0 (stop) to 255 (full)

pwm[1-3]_enable

- this file controls mode of fan/temperature control:
  - 0 Disabled

  - 1 Manual mode
  - 2 Smart Fan II
  - 3 Thermal Cruise

pwm[1-7]_mode

- Select PWM or DC mode
  - 0 DC
  - 1 PWM

thermal_cruise[1-3]

- Selects the desired temperature for cruise (degC)

tolerance[1-3]

- Value in degrees of Celsius (degC) for +- T

sf2_point[1-4]_fan[1-3]

- four temperature points for each fan for Smart Fan II

sf2_level[1-3]_fan[1-3]

- three PWM/DC levels for each fan for Smart Fan II

7.163 Kernel driver w83793

Supported chips:

- Winbond W83793G/W83793R

  Prefix: 'w83793'

  Addresses scanned: I2C 0x2c - 0x2f

  Datasheet: Still not published

Authors:

- Yuan Mu (Winbond Electronics)
- Rudolf Marek <r.marek@assembler.cz>

7.163.1 Module parameters

- **reset int** (default 0)

  This parameter is not recommended, it will lose motherboard specific settings. Use 'reset=1' to reset the chip when loading this module.

- **force_subclients=bus,caddr,saddr1,saddr2** This is used to force the i2c addresses for subclients of a certain chip. Typical usage is force_subclients=0,0x2f,0x4a,0x4b to force the subclients of chip 0x2f on bus 0 to i2c addresses 0x4a and 0x4b.

7.163.2 Description

This driver implements support for Winbond W83793G/W83793R chips.

- **Exported features** This driver exports 10 voltage sensors, up to 12 fan tachometer inputs, 6 remote temperatures, up to 8 sets of PWM fan controls, SmartFan (automatic fan speed control) on all temperature/PWM combinations, 2 sets of 6-pin CPU VID input.
- **Sensor resolutions** If your motherboard maker used the reference design, the resolution of voltage0-2 is 2mV, resolution of voltage3/4/5 is 16mV, 8mV for voltage6, 24mV for voltage7/8. Temp1-4 have a 0.25 degree Celsius resolution, temp5-6 have a 1 degree Celsiis resolution.
- **Temperature sensor types** Temp1-4 have 2 possible types. It can be read from (and written to) temp[1-4]_type.
  - If the value is 3, it starts monitoring using a remote termal diode (default).
  - If the value is 6, it starts monitoring using the temperature sensor in Intel CPU and get result by PECI.

  Temp5-6 can be connected to external thermistors (value of temp[5-6]_type is 4).
- **Alarm mechanism** For voltage sensors, an alarm triggers if the measured value is below the low voltage limit or over the high voltage limit. For temperature sensors, an alarm triggers if the measured value goes above the high temperature limit, and wears off only after the measured value drops below the hysteresis value. For fan sensors, an alarm triggers if the measured value is below the low speed limit.
- **SmartFan/PWM control** If you want to set a pwm fan to manual mode, you just need to make sure it is not controlled by any temp channel, for example, you want to set fan1 to manual mode, you need to check the value of temp[1-6]_fan_map, make sure bit 0 is cleared in the 6 values. And then set the pwm1 value to control the fan.

  Each temperature channel can control all the 8 PWM outputs (by setting the corresponding bit in tempX_fan_map), you can set the temperature channel mode using temp[1-6]_pwm_enable, 2 is Thermal Cruise mode and 3 is the SmartFanII mode. Temperature channels will try to speed up or slow down all controlled fans, this means one fan can receive different PWM value requests from different temperature channels, but the chip will always pick the safest (max) PWM value for each fan.

  In Thermal Cruise mode, the chip attempts to keep the temperature at a predefined value, within a tolerance margin. So if tempX_input > thermal_cruiseX + toleranceX, the chip will increase the PWM value, if tempX_input < thermal_cruiseX - toleranceX, the chip will decrease the PWM value. If the temperature is within the tolerance range, the PWM value is left unchanged.

  SmartFanII works differently, you have to define up to 7 PWM, temperature trip points, defining a PWM/temperature curve which the chip will

follow. While not fundamentally different from the Thermal Cruise mode, the implementation is quite different, giving you a finer-grained control.

- **Chassis** If the case open alarm triggers, it will stay in this state unless cleared by writing 0 to the sysfs file "intrusion0_alarm".
- **VID and VRM** The VRM version is detected automatically, don't modify the it unless you do know the cpu VRM version and it's not properly detected.

7.163.3 Notes

Only Fan1-5 and PWM1-3 are guaranteed to always exist, other fan inputs and PWM outputs may or may not exist depending on the chip pin configuration.

7.164 Kernel driver w83795

Supported chips:

- Winbond/Nuvoton W83795G

  Prefix: 'w83795g'

  Addresses scanned: I2C 0x2c - 0x2f

  Datasheet: Available for download on nuvoton.com

- Winbond/Nuvoton W83795ADG

  Prefix: 'w83795adg'

  Addresses scanned: I2C 0x2c - 0x2f

  Datasheet: Available for download on nuvoton.com

Authors:

- Wei Song (Nuvoton)
- Jean Delvare <jdelvare@suse.de>

7.164.1 Pin mapping

Here is a summary of the pin mapping for the W83795G and W83795ADG. This can be useful to convert data provided by board manufacturers into working libsensors configuration statements.

- W83795G

Pin	Name	Register	Sysfs attribute
13	VSEN1 (VCORE1)	10h	in0
14	VSEN2 (VCORE2)	11h	in1
15	VSEN3 (VCORE3)	12h	in2
16	VSEN4	13h	in3

Continued on next page

Table 20 – continued from previous page

Pin	Name	Register	Sysfs attribute
17	VSEN5	14h	in4
18	VSEN6	15h	in5
19	VSEN7	16h	in6
20	VSEN8	17h	in7
21	VSEN9	18h	in8
22	VSEN10	19h	in9
23	VSEN11	1Ah	in10
28	VTT	1Bh	in11
24	3VDD	1Ch	in12
25	3VSB	1Dh	in13
26	VBAT	1Eh	in14
3	VSEN12/TR5	1Fh	in15/temp5
4	VSEN13/TR5	20h	in16/temp6
5/ 6	VDSEN14/TR1/TD1	21h	in17/temp1
7/ 8	VDSEN15/TR2/TD2	22h	in18/temp2
9/ 10	VDSEN16/TR3/TD3	23h	in19/temp3
11/ 12	VDSEN17/TR4/TD4	24h	in20/temp4
40	FANIN1	2Eh	fan1
42	FANIN2	2Fh	fan2
44	FANIN3	30h	fan3
46	FANIN4	31h	fan4
48	FANIN5	32h	fan5
50	FANIN6	33h	fan6
52	FANIN7	34h	fan7
54	FANIN8	35h	fan8
57	FANIN9	36h	fan9
58	FANIN10	37h	fan10
59	FANIN11	38h	fan11
60	FANIN12	39h	fan12
31	FANIN13	3Ah	fan13
35	FANIN14	3Bh	fan14
41	FANCTL1	10h (bank 2)	pwm1
43	FANCTL2	11h (bank 2)	pwm2
45	FANCTL3	12h (bank 2)	pwm3
47	FANCTL4	13h (bank 2)	pwm4
49	FANCTL5	14h (bank 2)	pwm5
51	FANCTL6	15h (bank 2)	pwm6
53	FANCTL7	16h (bank 2)	pwm7
55	FANCTL8	17h (bank 2)	pwm8
29/ 30	PECI/TSI (DTS1)	26h	temp7
29/ 30	PECI/TSI (DTS2)	27h	temp8
29/ 30	PECI/TSI (DTS3)	28h	temp9
29/ 30	PECI/TSI (DTS4)	29h	temp10
29/ 30	PECI/TSI (DTS5)	2Ah	temp11
29/ 30	PECI/TSI (DTS6)	2Bh	temp12
29/ 30	PECI/TSI (DTS7)	2Ch	temp13
29/ 30	PECI/TSI (DTS8)	2Dh	temp14

Continued on next page

Table 20 – continued from previous page

Pin	Name	Register	Sysfs attribute
27	CASEOPEN#	46h	intrusion0

- W83795ADG

Pin	Name	Register	Sysfs attribute
10	VSEN1 (VCORE1)	10h	in0
11	VSEN2 (VCORE2)	11h	in1
12	VSEN3 (VCORE3)	12h	in2
13	VSEN4	13h	in3
14	VSEN5	14h	in4
15	VSEN6	15h	in5
16	VSEN7	16h	in6
17	VSEN8	17h	in7
22	VTT	1Bh	in11
18	3VDD	1Ch	in12
19	3VSB	1Dh	in13
20	VBAT	1Eh	in14
48	VSEN12/TR5	1Fh	in15/temp5
1	VSEN13/TR5	20h	in16/temp6
2/ 3	VDSEN14/TR1/TD1	21h	in17/temp1
4/ 5	VDSEN15/TR2/TD2	22h	in18/temp2
6/ 7	VDSEN16/TR3/TD3	23h	in19/temp3
8/ 9	VDSEN17/TR4/TD4	24h	in20/temp4
32	FANIN1	2Eh	fan1
34	FANIN2	2Fh	fan2
36	FANIN3	30h	fan3
37	FANIN4	31h	fan4
38	FANIN5	32h	fan5
39	FANIN6	33h	fan6
40	FANIN7	34h	fan7
41	FANIN8	35h	fan8
43	FANIN9	36h	fan9
44	FANIN10	37h	fan10
45	FANIN11	38h	fan11
46	FANIN12	39h	fan12
24	FANIN13	3Ah	fan13
28	FANIN14	3Bh	fan14
33	FANCTL1	10h (bank 2)	pwm1
35	FANCTL2	11h (bank 2)	pwm2
23	PECI (DTS1)	26h	temp7
23	PECI (DTS2)	27h	temp8
23	PECI (DTS3)	28h	temp9
23	PECI (DTS4)	29h	temp10
23	PECI (DTS5)	2Ah	temp11
23	PECI (DTS6)	2Bh	temp12
23	PECI (DTS7)	2Ch	temp13
23	PECI (DTS8)	2Dh	temp14

Continued on next page

Table 21 – continued from previous page

Pin	Name	Register	Sysfs attribute
21	CASEOPEN#	46h	intrusion0

7.165 Kernel driver w83l785ts

Supported chips:

- Winbond W83L785TS-S

  Prefix: 'w83l785ts'

  Addresses scanned: I2C 0x2e

  Datasheet: Publicly available at the Winbond USA website

  http://www.winbond-usa.com/products/winbond_products/pdfs/PCIC/W83L785TS-S.pdf

Authors: Jean Delvare <jdelvare@suse.de>

7.165.1 Description

The W83L785TS-S is a digital temperature sensor. It senses the temperature of a single external diode. The high limit is theoretically defined as 85 or 100 degrees C through a combination of external resistors, so the user cannot change it. Values seen so far suggest that the two possible limits are actually 95 and 110 degrees C. The datasheet is rather poor and obviously inaccurate on several points including this one.

All temperature values are given in degrees Celsius. Resolution is 1.0 degree. See the datasheet for details.

The w83l785ts driver will not update its values more frequently than every other second; reading them more often will do no harm, but will return 'old' values.

7.165.2 Known Issues

On some systems (Asus), the BIOS is known to interfere with the driver and cause read errors. Or maybe the W83L785TS-S chip is simply unreliable, we don't really know. The driver will retry a given number of times (5 by default) and then give up, returning the old value (or 0 if there is no old value). It seems to work well enough so that you should not notice anything. Thanks to James Bolt for helping test this feature.

7.166 Kernel driver w83l786ng

Supported chips:

- Winbond W83L786NG/W83L786NR

  Prefix: 'w83l786ng'

  Addresses scanned: I2C 0x2e - 0x2f

  Datasheet: http://www.winbond-usa.com/products/winbond_products/pdfs/PCIC/W83L786NRNG09.pdf

Author: Kevin Lo <kevlo@kevlo.org>

7.166.1 Module Parameters

- **reset boolean** (default 0)

  Use 'reset=1'to reset the chip (via index 0x40, bit 7). The default behavior is no chip reset to preserve BIOS settings

7.166.2 Description

This driver implements support for Winbond W83L786NG/W83L786NR chips.

The driver implements two temperature sensors, two fan rotation speed sensors, and three voltage sensors.

Temperatures are measured in degrees Celsius and measurement resolution is 1 degC for temp1 and temp2.

Fan rotation speeds are reported in RPM (rotations per minute). Fan readings readings can be divided by a programmable divider (1, 2, 4, 8, 16, 32, 64 or 128 for fan 1/2) to give the readings more range or accuracy.

Voltage sensors (also known as IN sensors) report their values in millivolts. An alarm is triggered if the voltage has crossed a programmable minimum or maximum limit.

7.166.3 /sys files

pwm[1-2]

- this file stores PWM duty cycle or DC value (fan speed) in range:

  0 (stop) to 255 (full)

pwm[1-2]_enable

- this file controls mode of fan/temperature control:
- 0 Manual Mode
- 1 Thermal Cruise
- 2 Smart Fan II

- 4 FAN_SET

pwm[1-2]_mode

- Select PWM of DC mode
- 0 DC
- 1 PWM

tolerance[1-2]

- Value in degrees of Celsius (degC) for +- T

7.167 Kernel driver wm831x-hwmon

Supported chips:

- Wolfson Microelectronics WM831x PMICs

  Prefix: 'wm831x'

  Datasheet:

  - http://www.wolfsonmicro.com/products/WM8310
  - http://www.wolfsonmicro.com/products/WM8311
  - http://www.wolfsonmicro.com/products/WM8312

Authors: Mark Brown <broonie@opensource.wolfsonmicro.com>

7.167.1 Description

The WM831x series of PMICs include an AUXADC which can be used to monitor a range of system operating parameters, including the voltages of the major supplies within the system. Currently the driver provides reporting of all the input values but does not provide any alarms.

7.167.2 Voltage Monitoring

Voltages are sampled by a 12 bit ADC. Voltages in millivolts are 1.465 times the ADC value.

7.167.3 Temperature Monitoring

Temperatures are sampled by a 12 bit ADC. Chip and battery temperatures are available. The chip temperature is calculated as:

Degrees celsius = (512.18 - data) / 1.0983

while the battery temperature calculation will depend on the NTC thermistor component.

7.168 Kernel driver wm8350-hwmon

Supported chips:

- Wolfson Microelectronics WM835x PMICs

  Prefix: 'wm8350'

  Datasheet:

  - http://www.wolfsonmicro.com/products/WM8350
  - http://www.wolfsonmicro.com/products/WM8351
  - http://www.wolfsonmicro.com/products/WM8352

Authors: Mark Brown <broonie@opensource.wolfsonmicro.com>

7.168.1 Description

The WM835x series of PMICs include an AUXADC which can be used to monitor a range of system operating parameters, including the voltages of the major supplies within the system. Currently the driver provides simple access to these major supplies.

7.168.2 Voltage Monitoring

Voltages are sampled by a 12 bit ADC. For the internal supplies the ADC is referenced to the system VRTC.

7.169 Kernel driver xgene-hwmon

Supported chips:

- APM X-Gene SoC

7.169.1 Description

This driver adds hardware temperature and power reading support for APM X-Gene SoC using the mailbox communication interface. For device tree, it is the standard DT mailbox. For ACPI, it is the PCC mailbox.

The following sensors are supported

- **Temperature**
  - SoC on-die temperature in milli-degree C
  - Alarm when high/over temperature occurs
- **Power**
  - CPU power in uW
  - IO power in uW

7.169.2 sysfs-Interface

temp0_input

- SoC on-die temperature (milli-degree C)

temp0_critical_alarm

- An 1 would indicates on-die temperature exceeded threshold

power0_input

- CPU power in (uW)

power1_input

- IO power in (uW)

7.170 Kernel driver xdpe122

Supported chips:

- Infineon XDPE12254

  Prefix: 'xdpe12254'

- Infineon XDPE12284

  Prefix: 'xdpe12284'

Authors:

Vadim Pasternak <vadimp@mellanox.com>

7.170.1 Description

This driver implements support for Infineon Multi-phase XDPE122 family dual loop voltage regulators. The family includes XDPE12284 and XDPE12254 devices. The devices from this family complaint with:

- Intel VR13 and VR13HC rev 1.3, IMVP8 rev 1.2 and IMPVP9 rev 1.3 DC-DC converter specification.
- Intel SVID rev 1.9. protocol.
- PMBus rev 1.3 interface.

Devices support linear format for reading input voltage, input and output current, input and output power and temperature. Device supports VID format for reading output voltage. The below modes are supported: - VR12.0 mode, 5-mV DAC - 0x01. - VR12.5 mode, 10-mV DAC - 0x02. - IMVP9 mode, 5-mV DAC - 0x03. - AMD mode 6.25mV - 0x10.

Devices support two pages for telemetry.

The driver provides for current: input, maximum and critical thresholds and maximum and critical alarms. Critical thresholds and critical alarm are supported only for current output. The driver exports the following attributes for via the sysfs files, where indexes 1, 2 are for "iin" and 3, 4 for "iout":

curr[3-4]_crit

curr[3-4]_crit_alarm

curr[1-4]_input

curr[1-4]_label

curr[1-4]_max

curr[1-4]_max_alarm

The driver provides for voltage: input, critical and low critical thresholds and critical and low critical alarms. The driver exports the following attributes for via the sysfs files, where indexes 1, 2 are for "vin" and 3, 4 for "vout":

in[1-4]_crit

in[1-4_crit_alarm

in[1-4]_input

in[1-4_label

in[1-4]_lcrit

in[1-41_lcrit_alarm

The driver provides for power: input and alarms. Power alarm is supported only for power input. The driver exports the following attributes for via the sysfs files, where indexes 1, 2 are for "pin" and 3, 4 for "pout":

power[1-2]_alarm

power[1-4]_input

power[1-4]_label

The driver provides for temperature: input, maximum and critical thresholds and maximum and critical alarms. The driver exports the following attributes for via the sysfs files:

temp[1-2]_crit

temp[1-2]_crit_alarm

temp[1-2]_input

temp[1-2]_max

temp[1-2]_max_alarm

7.171 Kernel driver zl6100

Supported chips:

- Intersil / Zilker Labs ZL2004

  Prefix: 'zl2004'

  Addresses scanned: -

  Datasheet: http://www.intersil.com/data/fn/fn6847.pdf

- Intersil / Zilker Labs ZL2005

  Prefix: ‘zl2005’

  Addresses scanned: -

  Datasheet: http://www.intersil.com/data/fn/fn6848.pdf
- Intersil / Zilker Labs ZL2006

  Prefix: ‘zl2006’

  Addresses scanned: -

  Datasheet: http://www.intersil.com/data/fn/fn6850.pdf
- Intersil / Zilker Labs ZL2008

  Prefix: ‘zl2008’

  Addresses scanned: -

  Datasheet: http://www.intersil.com/data/fn/fn6859.pdf
- Intersil / Zilker Labs ZL2105

  Prefix: ‘zl2105’

  Addresses scanned: -

  Datasheet: http://www.intersil.com/data/fn/fn6851.pdf
- Intersil / Zilker Labs ZL2106

  Prefix: ‘zl2106’

  Addresses scanned: -

  Datasheet: http://www.intersil.com/data/fn/fn6852.pdf
- Intersil / Zilker Labs ZL6100

  Prefix: ‘zl6100’

  Addresses scanned: -

  Datasheet: http://www.intersil.com/data/fn/fn6876.pdf
- Intersil / Zilker Labs ZL6105

  Prefix: ‘zl6105’

  Addresses scanned: -

  Datasheet: http://www.intersil.com/data/fn/fn6906.pdf
- Intersil / Zilker Labs ZL9101M

  Prefix: ‘zl9101’

  Addresses scanned: -

  Datasheet: http://www.intersil.com/data/fn/fn7669.pdf
- Intersil / Zilker Labs ZL9117M

  Prefix: ‘zl9117’

Addresses scanned: -

Datasheet: http://www.intersil.com/data/fn/fn7914.pdf

- Ericsson BMR450, BMR451

  Prefix: 'bmr450', 'bmr451'

  Addresses scanned: -

  Datasheet:

http://archive.ericsson.net/service/internet/picov/get?DocNo=28701-EN/LZT146401

- Ericsson BMR462, BMR463, BMR464

  Prefixes: 'bmr462', 'bmr463', 'bmr464'

  Addresses scanned: -

  Datasheet:

    http://archive.ericsson.net/service/internet/picov/get?DocNo=28701-EN/LZT146256

Author: Guenter Roeck <linux@roeck-us.net>

7.171.1 Description

This driver supports hardware monitoring for Intersil / Zilker Labs ZL6100 and compatible digital DC-DC controllers.

The driver is a client driver to the core PMBus driver. Please see Documentation/hwmon/pmbus.rst and Documentation.hwmon/pmbus-core for details on PMBus client drivers.

7.171.2 Usage Notes

This driver does not auto-detect devices. You will have to instantiate the devices explicitly. Please see Documentation/i2c/instantiating-devices.rst for details.

Warning: Do not access chip registers using the i2cdump command, and do not use any of the i2ctools commands on a command register used to save and restore configuration data (0x11, 0x12, 0x15, 0x16, and 0xf4). The chips supported by this driver interpret any access to those command registers (including read commands) as request to execute the command in question. Unless write accesses to those registers are protected, this may result in power loss, board resets, and/or Flash corruption. Worst case, your board may turn into a brick.

7.171.3 Platform data support

The driver supports standard PMBus driver platform data.

7.171.4 Module parameters

7.171.5 delay

Intersil/Zilker Labs DC-DC controllers require a minimum interval between I2C bus accesses. According to Intersil, the minimum interval is 2 ms, though 1 ms appears to be sufficient and has not caused any problems in testing. The problem is known to affect all currently supported chips. For manual override, the driver provides a writeable module parameter, 'delay', which can be used to set the interval to a value between 0 and 65,535 microseconds.

7.171.6 Sysfs entries

The following attributes are supported. Limits are read-write; all other attributes are read-only.

in1_label	"vin"
in1_input	Measured input voltage.
in1_min	Minimum input voltage.
in1_max	Maximum input voltage.
in1_lcrit	Critical minimum input voltage.
in1_crit	Critical maximum input voltage.
in1_min_alarm	Input voltage low alarm.
in1_max_alarm	Input voltage high alarm.
in1_lcrit_alarm	Input voltage critical low alarm.
in1_crit_alarm	Input voltage critical high alarm.
in2_label	"vmon"
in2_input	Measured voltage on VMON (ZL2004) or VDRV (ZL9101M, ZL9117M) pin. Reported voltage is 16x the voltage on the pin (adjusted internally by the chip).
in2_lcrit	Critical minimum VMON/VDRV Voltage.
in2_crit	Critical maximum VMON/VDRV voltage.
in2_lcrit_alarm	VMON/VDRV voltage critical low alarm.
in2_crit_alarm	VMON/VDRV voltage critical high alarm. vmon attributes are supported on ZL2004, ZL9101M, and ZL9117M only.
inX_label	"vout1"

Continued on next page

Table 22 – continued from previous page

inX_input	Measured output voltage.
inX_lcrit	Critical minimum output Voltage.
inX_crit	Critical maximum output voltage.
inX_lcrit_alarm	Critical output voltage critical low alarm.
inX_crit_alarm	Critical output voltage critical high alarm. X is 3 for ZL2004, ZL9101M, and ZL9117M, 2 otherwise.
curr1_label	“iout1”
curr1_input	Measured output current.
curr1_lcrit	Critical minimum output current.
curr1_crit	Critical maximum output current.
curr1_lcrit_alarm	Output current critical low alarm.
curr1_crit_alarm	Output current critical high alarm.
temp[12]_input	Measured temperature.
temp[12]_min	Minimum temperature.
temp[12]_max	Maximum temperature.
temp[12]_lcrit	Critical low temperature.
temp[12]_crit	Critical high temperature.
temp[12]_min_alarm	Chip temperature low alarm.
temp[12]_max_alarm	Chip temperature high alarm.
temp[12]_lcrit_alarm	Chip temperature critical low alarm.
temp[12]_crit_alarm	Chip temperature critical high alarm.