History of the OpenBSD Hardware Sensors Framework

Constantine A. Murenin
University of Waterloo

EuroBSDCon 2008 — 16/19 October 2008 — Strasbourg, France
Outline

• Introduction
• Framework API and utilities
• Drivers
• I²C Bus Scan
• Summer of Code
• Conclusion
What is a sensor?

• Any device with a sensor-like data:
  • temperature
  • voltage
  • fan speed
  • ...
  • logical drive status
  • time offset
Are these common at all?

- many Super I/O chips have integrated hardware monitors
- Intel Core and AMD K8 / K10 have integrated thermal sensors
- IPMI in servers / ACPI in laptops
- SCSI enclosures
- 10GbE and 802.11
Why sensors framework?

- Monitoring environmental values can predict, detect, troubleshoot system failure. (Voltage, temperature, fan, logical drive status.)
- Unified interface, no configuration required, works out-of-the-box.
- Sensors are fun!
Drivers in 4.4 since 4.3

- 62nd driver: fins(4)
- 63rd: andl(4)
- 64th: kate(4)
- 65th: sdtemp(4) — JEDEC (JC -42.4) SO-DIMM
- 66th: adtfsm(4)
- 67th: km(4) — AMD Phenom, Opteron Barcelona
- 68th: vmt(4)
Latest drivers

• sdtemp(4) — SO-DIMM temperature sensors

• km(4) — AMD Family 10h processors (Phenom, Opteron Barcelona) and Family 11h (Turion X2 Ultra et al)

neither of these two are in Linux yet!
Design decisions

- Keep it simple, secure and usable
- Make it work by default
- Overengineering is useless — many devices have incomplete specifications
- No buttons™
How voltage sensors work?

- Most chips have sensors from 0 to 4 V
- Excess voltage removed by resistors
- Resistor “recommendations”
How voltage sensors read?

<table>
<thead>
<tr>
<th>function</th>
<th>maths</th>
<th>result</th>
</tr>
</thead>
<tbody>
<tr>
<td>original readin’</td>
<td>0x0cb</td>
<td>203</td>
</tr>
<tr>
<td>sensor voltage</td>
<td>203 * 16 mV</td>
<td>3.24 V</td>
</tr>
<tr>
<td>scale for +5 V</td>
<td>3.24 V * 1.68</td>
<td>5.44 V</td>
</tr>
<tr>
<td>scale for +12 V</td>
<td>3.24 V * 3.80</td>
<td>12.31 V</td>
</tr>
</tbody>
</table>
Resister recommendations

- Ignored by some motherboard designers
- Not given in documentation for some chips

Results:
- Voltage “doesn’t scale”
- Do the best with what you have
Framework API

/sys/sys/sensors.h

- struct sensor / struct sensordev, transport over sysctl(3)

- sensor description, e.g. “CPU” (optional)


- sensor state: unspec, ok, warn, crit, unknown
struct sensor {
    char desc[32];   /* sensor description, may be empty */
    struct timeval tv;  /* sensor value last change time */
    int64_t value;   /* current value */
    enum sensor_type type;  /* sensor type */
    enum sensor_status status; /* sensor status */
    int numt;      /* sensor number of .type type */
    int flags;     /* sensor flags */
};

struct sensordev {
    int num;      /* sensordev number */
    char xname[16];   /* unix device name */
    int maxnumt[SENSOR_MAX_TYPES];
    int sensors_count;
};
Adding sensors in attach()

```c
void
drv_attach(struct device *parent, struct device *self, void *aux)
{
    ...

    strlcpy(sc->sc_sensordev.xname, sc->sc_dev.dv_xname,
            sizeof(sc->sc_sensordev.xname));

    for (i = 0; i < n; i++) {
        sc->sc_sensors[i].type = SENSOR_TEMP;
        sensor_attach(&sc->sc_sensordev, &sc->sc_sensors[i]);
    }

    if (sensor_task_register(sc, drv_refresh, 5) == NULL) {
        printf(": unable to register the update task\n");
        return;
    }

    sensordev_install(&sc->sc_sensordev);

    printf("\n");
}
```
Sensor task refresh procedure

```c
void drv_refresh(void *arg)
{
    struct drv_softc *sc = arg;
    struct ksensor *s = sc->sc_sensors;
    ...

    for (i = 0; i < n; i++)
        s[i].value = ...;
}
```
Sensor tools in OpenBSD

- `sysctl(3)` HW_SENSORS / `sysctl(8)` hw.sensors
- `systat(1)` — semi-realtime sensor monitoring
- `sensorsd(8)` — sensor monitor
- `ntpd(8)` — timedelta minimiser
- `snmpd(8)` — SNMP daemon
- `ports/sysutils/symon` — remote monitoring
% sysctl hw.sensors

hw.sensors.km0.temp0=50.50 degC
hw.sensors.it0.temp0=32.00 degC
hw.sensors.it0.temp1=45.00 degC
hw.sensors.it0.temp2=92.00 degC
hw.sensors.it0.fan0=2528 RPM
hw.sensors.it0.volt0=1.34 VDC (VCORE_A)
hw.sensors.it0.volt1=1.92 VDC (VCORE_B)
hw.sensors.it0.volt2=3.42 VDC (+3.3V)
hw.sensors.it0.volt3=5.21 VDC (+5V)
hw.sensors.it0.volt4=12.54 VDC (+12V)
hw.sensors.it0.volt5=1.62 VDC (-5V)
hw.sensors.it0.volt6=4.01 VDC (-12V)
hw.sensors.it0.volt7=5.75 VDC (+5VSB)
hw.sensors.it0.volt8=3.23 VDC (VBAT)
sensorsd

• fills in your logs
• no manual configuration required for ‘smart’ sensors (those that keep state)
• most other sensors require very minimal configuration (“temp:low=15C:high=65C”)
Drivers

- Super I/O hardware monitors (lm, it, viaenv, viasio, nsclpcsio, fins etc)
- SMBus hardware monitors (too many to mention)
- Embedded temperature sensors (Ethernet, CPU etc)
- SCSI enclosures and IPMI (safte, ses, ipmi, esm)
- Various ACPI sensors (temperature, voltage, power)
- RAID logical drive status sensors (esm, ami, ciss, mfi, arc, softraid)
- time offset sensors (“timedelta” sensors)
Drivers by type

- i²c: 29
- Super I/O: 6
- timedelta: 6
- drive: 6
- misc: 17
- ACPI: 4
Drivers

- OpenBSD 3.4 — 3 drivers (lm, it, viaenv)
- OpenBSD 3.5 — 4 drivers (…, nsclpcsio)
- OpenBSD 3.6 — 5 drivers (…, lmtemp)
- OpenBSD 3.7 — 5 drivers
- OpenBSD 3.8 — 9 drivers (…, aps, viasio, safte, ses)
- OpenBSD 3.9 — 33 drivers: huge number of i²c sensors, i2c_scan, IPMI, drive status sensor introduced
- OpenBSD 4.0 — 42 drivers, timedelta introduced
Drivers

• OpenBSD 4.0-current as of 2006-12-23 — new framework revision, 44 drivers converted
• OpenBSD 4.1 — 46 drivers
• OpenBSD 4.2 — 51 drivers
• OpenBSD 4.3 — 61 drivers
• OpenBSD 4.4 — 68 drivers!
Many chips lack meaningful signatures
Open Firmware provides a list of devices (string, $i^2c$-address pairs)
Drivers match by string, e.g. “adt7467” or “ds1775”
I²C Bus Scan

/sys/dev/i2c/i2c_scan.c

- when there’s no Open Firmware (e.g. i386/amd64/etc)
- goes through a list of i²c-addresses where sensors live
  - for each address, the value of each register is cached on the first read, unless it is ignored entirely via blacklisting
  - the result of successful scan iteration is a string describing the chip (e.g. “w83793g”)
I²C Bus Scan (cont.)

- All signatures are located in i2c_scan.c, ensuring that there are no conflicts.
- OpenBSD-way: all of this is enabled by default.
- Result: code is tested on all machines that have i²c and don’t have Open Firmware.
- All supported i²c drivers are enabled in GENERICs and “just work”.
I²C Sandbox

• i2c_scan.c prints a register dump for unidentified sensors into dmesg

• we kindly ask all users to voluntarily send dmesg’s to dmesg@openbsd.org archive

• a sandbox driver wrapper can be easily written to parse the dumps, and test drivers

• streamlines i²c driver development and initial testing
NetBSD envsys / sysmon

- 31 drivers in NetBSD (vs. 68 in OpenBSD)
- more complicated API
- non-standard tools
- ‘drive’ sensors ported from OpenBSD
- 2007-11 envsys2 API introduced suspicious resemblance of OpenBSD’s sensor_attach API
Framework Timeline, Simplified

1999/2000: envsys / sysmon introduced into NetBSD, with lm(4) and viaenv(4)

2003-04-25: lm(4) and viaenv(4) are committed into OpenBSD by grange@ (Alexander Yurchenko), but with a much simpler sysctl-based interfacing, first appeared in OpenBSD 3.4

2004/2005: evolution by grange, dlg, kettenis and deraadt

2006-12-23: deraadt@ commits my patches, converting 44 device drivers and userland applications from one-level addressing to two-level addressing (e.g. hw.sensors.11 to hw.sensors.lm0.temp2)

Porting to FreeBSD

• Last of day of SoC applications, looked at the FreeBSD ideas page by chance, seeing that someone has requested a port of the sensors framework

• Applied, talked with several people, got accepted
Summer of Code 2007

- Ported the sensors API and documentation
- Drivers: lm, it, coretemp
- Userland applications: sysctl, sensorsd, systat
- Fixed several small bugs here and there
- Fixed one 10-year-old bug in OpenBSD and another 12-year-old bug in FreeBSD
GSoC2007/cnst-sensors

• Complete final patch released on 2007-09-13, but FreeBSD HEAD still frozen

• Hasso Tepper mailed me on 2007-09-25 thanking me for the FreeBSD port, and saying that with small adaptations the work will be committed into DragonFly really soon — took me by surprise

• Committed to DragonFly on 2007-10-02
Sensors in FreeBSD CVS

• Approved by re@ and committed into FreeBSD 8.0-CURRENT by Alexander Leidinger (netchild@) on 2007-10-14, same week when RELENG_7 branch was created and the long-term code freeze of CVS HEAD was over

• Backed out on 2007-10-15 per phk request
Sensors in FreeBSD

• The SoC project was done to phk’s satisfaction!

• However, few questions were posed:
  1. whether the framework is needed in FreeBSD in the first place
  2. whether this specific framework has a FreeBSD feel to it

• Huge discussions, many people liked it, many people hated it, others wanted more drivers.
Sensors in FreeBSD?

- separate sensors framework is less needed in FreeBSD due to phk’s “sysctl magic”
  - many people still want it, though...
- userland vs. kernel argument
  - then why coretemp(4) and k8temp(4)?
Sensors in FreeBSD Finale

• “ported to FreeBSD, committed to DragonFly”
• gained a lot of experience with drivers
• thanks to syrinx, netchild, rpaulo, rwatson, sam and many others
Conclusion

• 68 drivers in OpenBSD 4.4
• Framework is popular and in high demand
• Driver code is shared between NetBSD, OpenBSD, DragonFly BSD and FreeBSD
• Userland interface is compatible between OpenBSD and DragonFly BSD, and patched FreeBSD
Future Projects

- Write even more sensor drivers for OpenBSD (75+ drivers by OpenBSD 4.5?)
- Port sensors-detect.pl from lm_sensors
- Port i2c_scan.c to FreeBSD / DragonFly APIs
- Further improve sensorsd
- Fan-speed controlling