History of the OpenBSD Hardware Sensors Framework

Constantine A. Murenin
University of Waterloo

AsiaBSDCon 2009 — 12/15 March 2009 — Tokyo, Japan

Outline

- Introduction
- Framework API and utilities
- Drivers
- I^2C Bus Scan
- 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

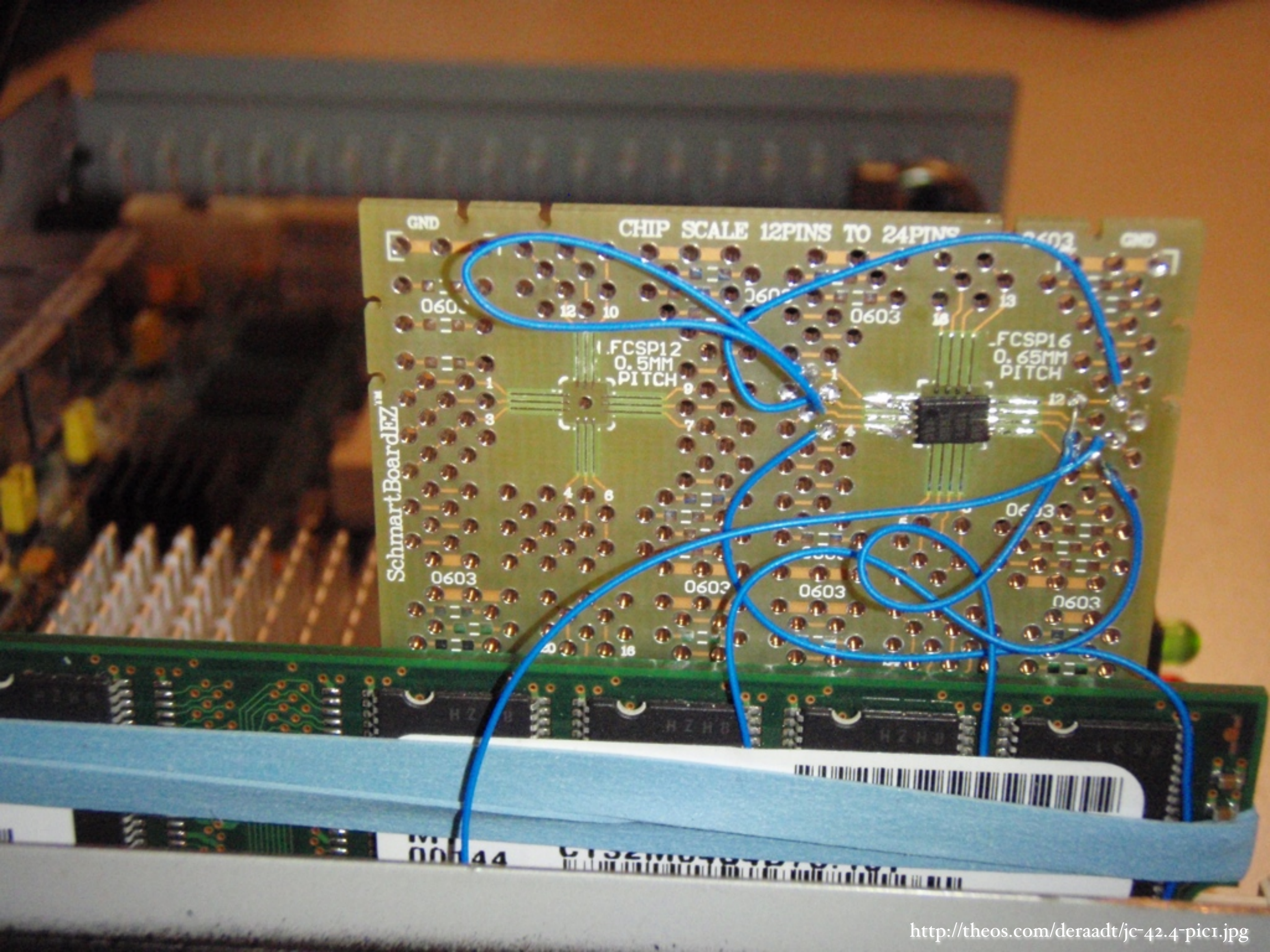
Why sensors framework?

- 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!

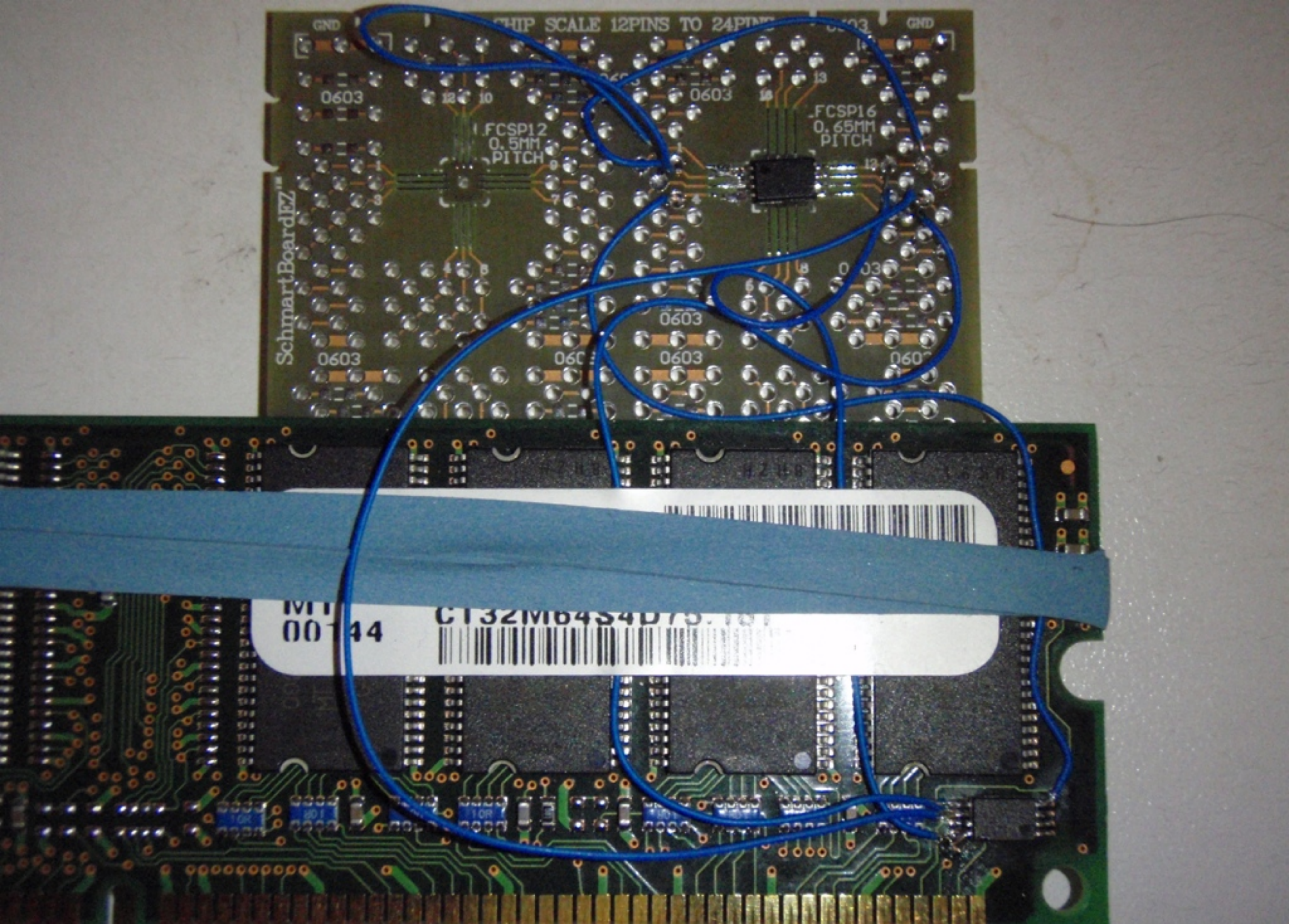
Uber cool drivers

- 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!



http://theos.com/deraadt/jc-42.4-pic1.jpg



http://theos.com/deraadt/jc-42.4-pic2.jpg

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?

function	*maths*	*result*
original readin'	0xcb	203
sensor voltage	203 * 16 mV	3.24 V
scale for +5 V	3.24 V * 1.68	5.44 V
scale for +12 V	3.24 V * 3.80	12.31 V

Resistor 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 type / unit: ‘temp’, ‘fan’, ‘volt’, ‘indicator’, ‘drive’, ‘timedelta’ etc
  - sensor state: unspec, ok, warn, crit, unknown

Adding sensors in attach()

```
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

```
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
- ports/sysutils/gkrellm — GUI 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)
```

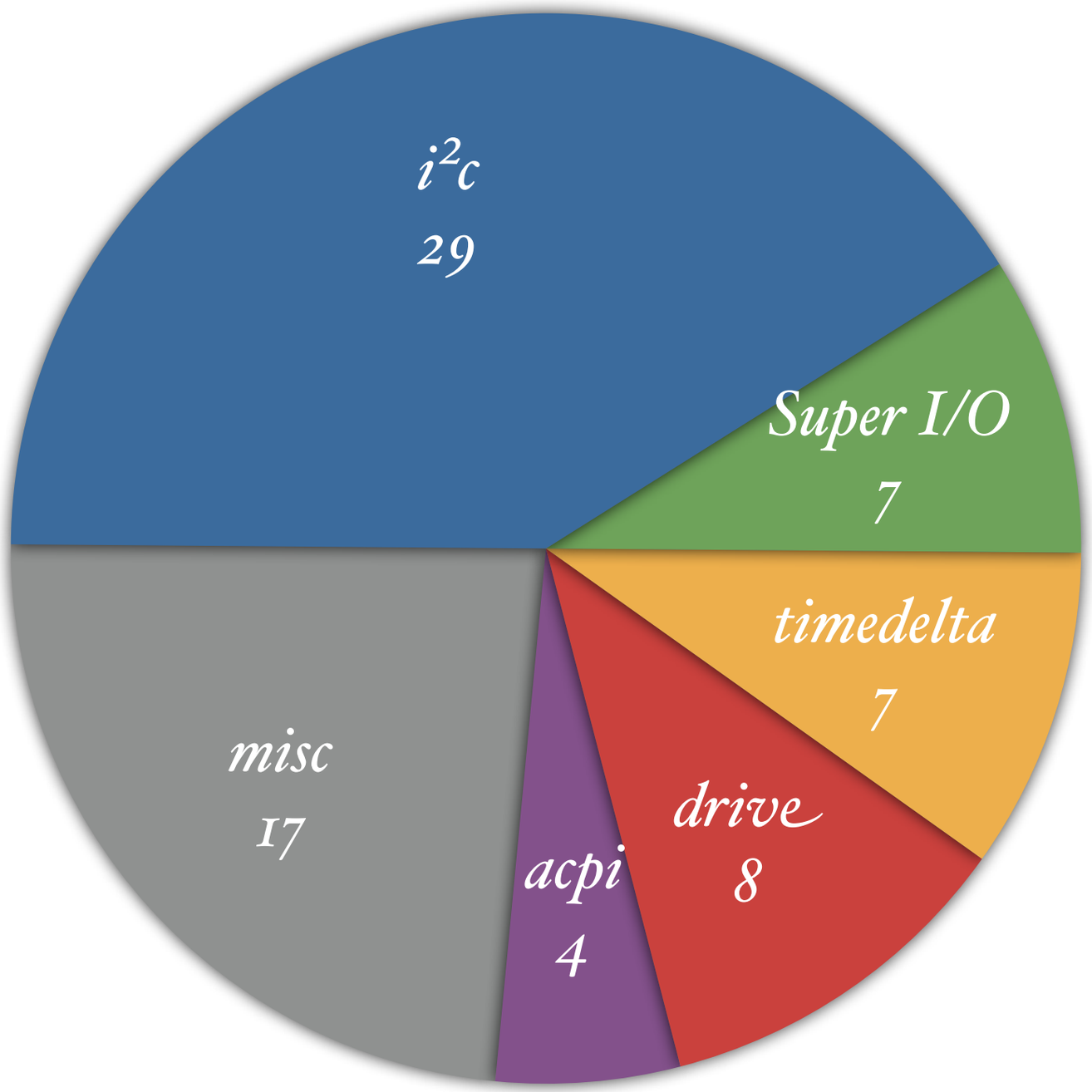
sensorsd

- 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”)

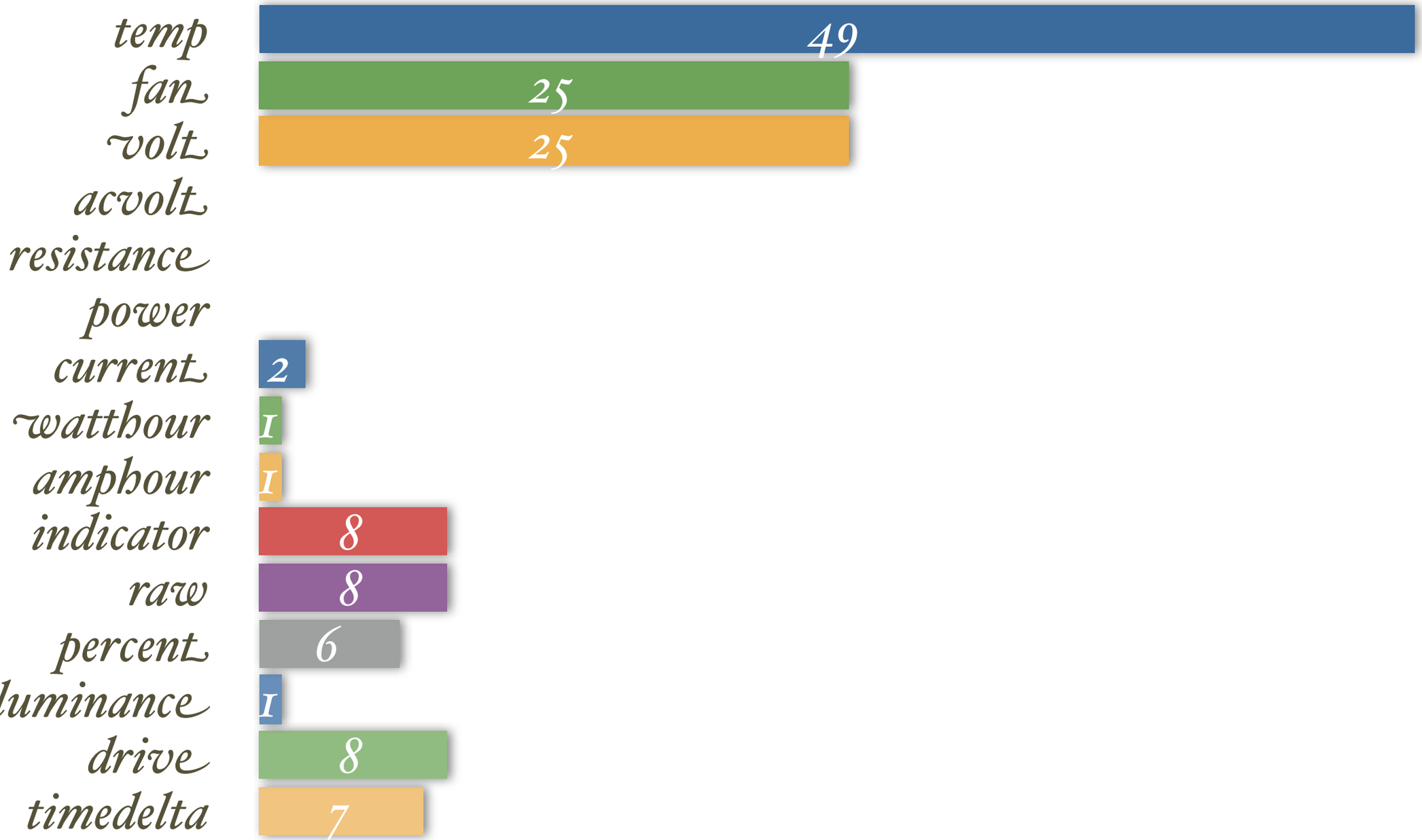
Drivers

- Super I/O hardware monitors (lm, it, viaenv, viasio, nsclpcsio, fins, schsio 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, cac, mpi)
- time offset sensors (“timedelta” sensors)

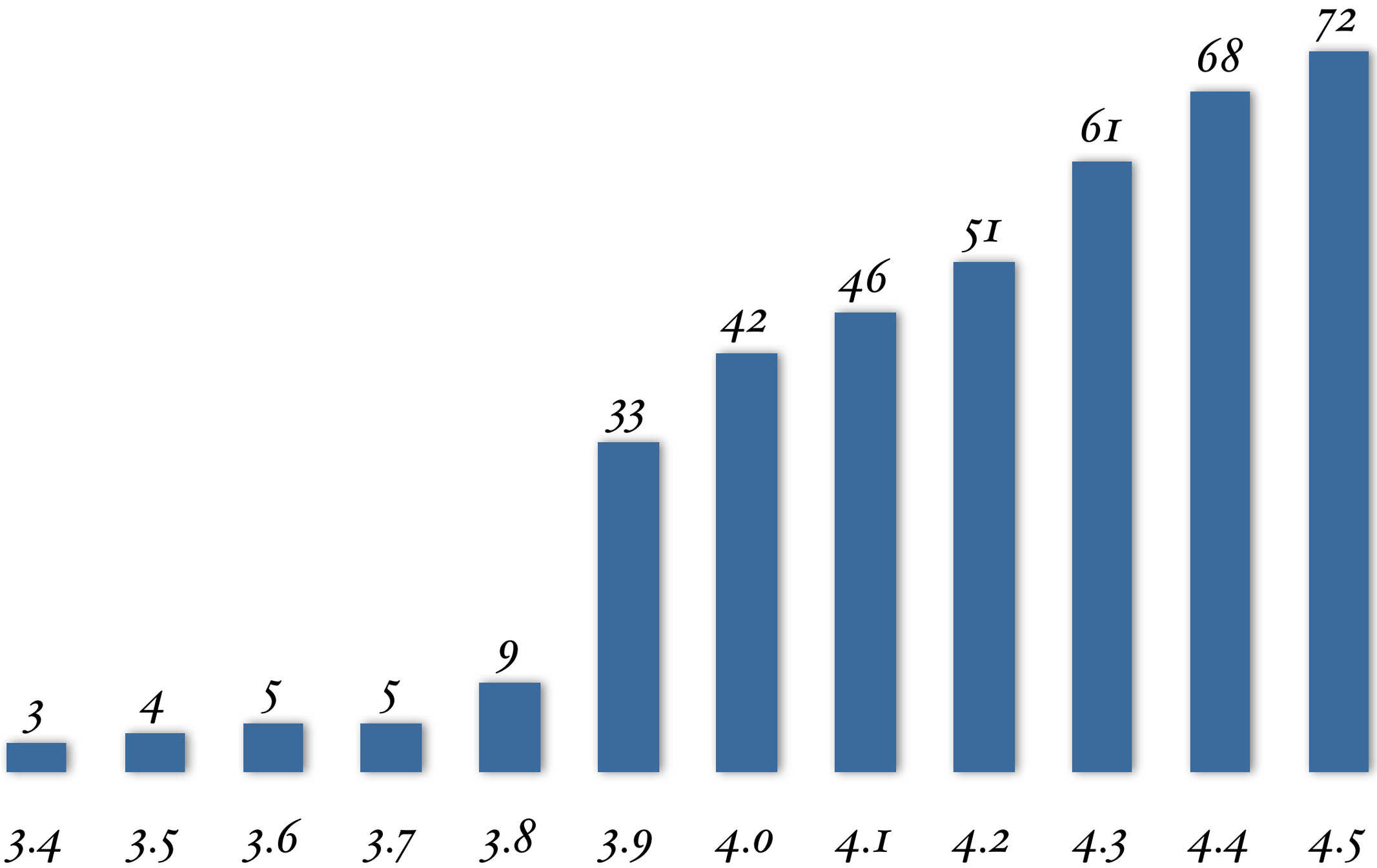
Drivers by category



Drivers by type



Drivers by release



I²C

- Many chips lack meaningful signatures
- Open Firmware provides a list of devices (string, i²c-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

- 32 drivers in NetBSD (vs. 72 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)

2007-09-13: final GSoC2007/cnst-sensors patch released for FreeBSD 7.0-CURRENT

Conclusion

- 72 drivers in OpenBSD 4.5
- 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 (76 drivers by OpenBSD 4.6?)
- Port sensors-detect.pl from lm_sensors
- Port i2c_scan.c to FreeBSD / DragonFly APIs
- Further improve sensorsd
- Fan-speed controlling

Questions?
Comments?

Constantine A. Murenin
<cnst@openbsd.org>