

Monitoring zdraví počítače.

Gratuluji, tím že jste se rozhodli navštívit moji přednášku, nebo si aspoň přečíst tento článek, jste prokázali, že vám zdraví vašeho počítače není lhostejné.

. Teplota procesoru i otáčky větráku budou zřejmě zajímat i Linuxáře overklokaře, nebo správce buildovacích mašin. Linuxáře kolejáky zase regulace otáček větráčku v závislosti na na teplotě, to proto aby nemuseli vypínat počítač přes noc. Tento příspěvek by měl sloužit jako úvod do problematiky, nikoliv jako výklad vyčerpávající jak duševně i obsahově.

Co všechno se monitoruje

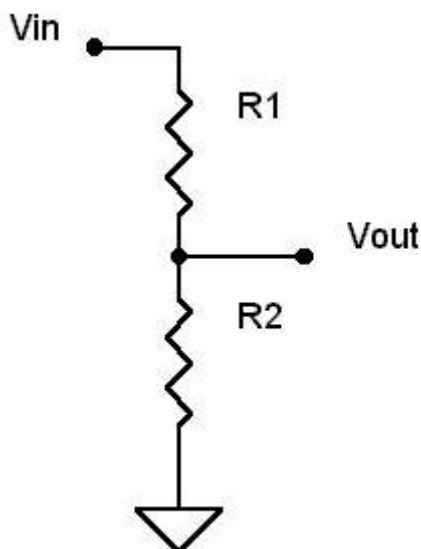
Zajímavé fyzikální veličiny z hlediska zdraví počítače zahrnují napětí, proud, otáčky, teploty a zrychlení

Napětí

Fyzikální veličinou jsou volty [V], monitorují se některé z napájecích větví zdroje a napětí, kterým se živí mikroprocesor. Typicky se jedná o +5V, +3.3V, +2.5V, +12V, záporná napětí -5V a -12V se monitorují jen ve výjimečných případech. Napětí se převádí z analogové oblasti do digitální klasickým A/D převodníkem. Každý převodník je charakterizován počtem bitů – tedy rozsahem a mV na jeden bit – přesností. Drtivá většina převodníků je osmibitových s LSB na bit 8mV. To znamená, že nejvyšší možné napětí, které lze měřit je $8\text{mV} * 255$, což jsou asi 2V. Pro měření většiny napětí je tedy třeba vstupní napětí v nějakém poměru snížit. Používají se proto odporové děliče. Mým spolužákům z FELu netřeba snad vysvětlovat oč se jedná. Ale pro ty ostatní snad postačí, že se jedná o dva odpory zapojené do série, jeden konec se zapojí na napětí co chceme snížit, druhý konec se připojí na zem. Ze spojeného prostředku se odeberá vydělené napětí v poměru odporů:

$$V_{out} = \frac{R_2}{R_1 + R_2} \cdot V_{in}$$

Další informace najde laskavý čtenář například zde:



http://encyclopedia.laborlawtalk.com/voltage_divider

Pokročilé čtenáře jistě napadne, že právě hodnoty obou odporů potřebujeme znát a zakomponovat je do převodního vzorečku. Naštěstí zkušenosti ukazují, že výrobci motherboardů naštěstí často ctí datasheety chipů, kde jsou odpory předepsané. Samozřejmě existují výjimky potvrzující pravidlo. Jak situaci řešit si povíme později.

Posledním druhem napětí, který se „měří“, je napájecí napětí procesoru. Uvozovky jsou namístě, nic se neměří, ale převádí se podle tabulky. Z procesoru vede několik drátů co se označují jako VID (tvoří tak několikabitové (5 nebo 6 bit číslo). Regulátor napětí (VRM) na desce podle hodnoty VID vyrobí odpovídající napětí (VCore) pro procesor. Pokud VID napětí změním v BIOSu, výstup z procesoru se nepoužije, ale BIOS zařídí vyslání jiného čísla do VRM. Někdy výrobci motherboardů použijí pro změnu VID speciální chip. Často bývá od firmy Attansic ze série **ATXP?**. Poslední jádra řady 2.6 podporují tento chip, takže VID můžete směle měnit i z Linuxu.

$$V_{out} = \frac{R_2}{R_1 + R_2} \cdot V_{in}$$

Proud

Proud se měří v bateriích notebooků, avšak nikdy jsem nevynašel jinou metodu než si dotyčnou informaci přečíst z ACPI modulu battery. (cat /proc/acpi/battery/BAT1/state)

Otáčky

V počítačích se měří otáčky větráčku, jednotkou jsou otáčky za minutu, anglicky často jako RPM (rotations per minute). Často jsou připojeny senzory k větráčku mikroprocesoru a chipsetu, méně často větráček ze zdroje nebo grafické karty. Otáčky se pohybují od 2000 do 5000 ot/min.

Měřicí chipy mají vstup, kam je připojen ten třetí kablík od větráčku. Pokud je větráček připojen na více než 5V, je opět nutné upravit napětíovou úroveň odporovým děličem.

Rychlost větráčku se měří čítačem. Čítač počítá pulsy ze vstupu, ale to pořád nestačí pro to, abychom mohli uživateli nabídnout počet otáček za minutu. Chip nemá stopky, ale vstupní signál se hradluje s hodinami, které typicky běží na 22kHz. Aby bylo možné postihnout velký rozsah otáček a měřit pokud možno co nejpřesněji byl přidán ještě jeden měřicí parametr – fan divisor. Dělitel. Dělitele musíme zvýšit abychom mohli měřit nižší otáčky přesněji a naopak snížit abychom mohli měřit rychle se točící větráčky. Dělitel nedělí vstupní signál ale ten hradlovací takže to pracuje opačně než by člověk čekal. Použitý vzorec vypadá asi takhle:

$$\text{RPM} = \text{hodiny}/(\text{div} \cdot \text{count})$$

K dispozici jsou dělitelé (div) pouze jako mocniny 2. Typicky chip nabízí 1, 2, 4 nebo 8.

Teploty

Teplota se v našich končinách měří ve stupních Celsia. Teplota se převádí na měřitelnou veličinu termistorem, který mění odpor v závislosti na teplotě. Změna odporu v obvodu způsobí změnu napětí v děliči a tak problém převedeme na měření napětí a ten už umíme vyřešit. Jak měřit teplotu v procesoru? I na to konstruktéři mysleli a zabudovali do procesoru diodu nebo celý transistor, jež mění své vlastnosti v závislosti na teplotě. Teplota měřená termistorem se typicky mění pozvolněji, diodou se mění skokově a hodně se zátěží CPU. Některé základní desky vydávají za teplotu procesoru jen teplotu termistoru pod patičí procesoru a dohání to konstantou kolem +10C, ale i tak nic moc...

S teplotou a převodními vztahy nebývají problémy, výrobci se většinou drží zaběhnutých řešení.

Zrychlení

Některé notebooky Thinkpad mají v sobě zabudovaný akcelerometr, který měří zrychlení notebooku. Pokud notebook padá na zem, ovladač zaparkuje disky. Komerční název této technologie je HDAPS. Dobrou zprávou je, že nějací geekové chtěli změřit jak rychle jezdí jejich auto a tak se driver posléze objevil v Linuxu...

Regulace rychlosti větráčku

Výrobci základních desek mají ve zvyku tyto technologie nazývat různě i když oba výrobci používají stejný chip. V podstatě existují dva přístupy. První nastaví pár bodů na křivce závislosti teploty na otáčkách. Druhý přístup reguluje otáčky větráčku tak, aby teplota zůstávala kolem zvolené teploty plus mínus několik stupňů. I v Linuxu jsou k dispozici prostředky pro nastavování těchto pokročilých vlastností HW monitoring chipů. Je však důležité prostudovat dokumentaci k ovladači, protože každý chip nabízí trochu něco jiného.

Chipy

Myslím, že nejlepší je rozdělit chipy podle sběrnic, ke kterým se připojují. Na PC připadají v úvahu sběrnice ISA a I2C/SMBus. Seznam chipů, které jsou v Linuxu podporovány nebo aspoň se o nich ví najdete na: <http://secure.netroedge.com/~lm78/supported.html> a na <http://secure.netroedge.com/~lm78/newdrivers.html>. Krom ovladače pro konkrétní chip je také potřeba ovladač pro sběrnici I2C/SMBus. Podporované chipsety naleznete tamtéž. Následuje stručná charakteristika sběrnic.

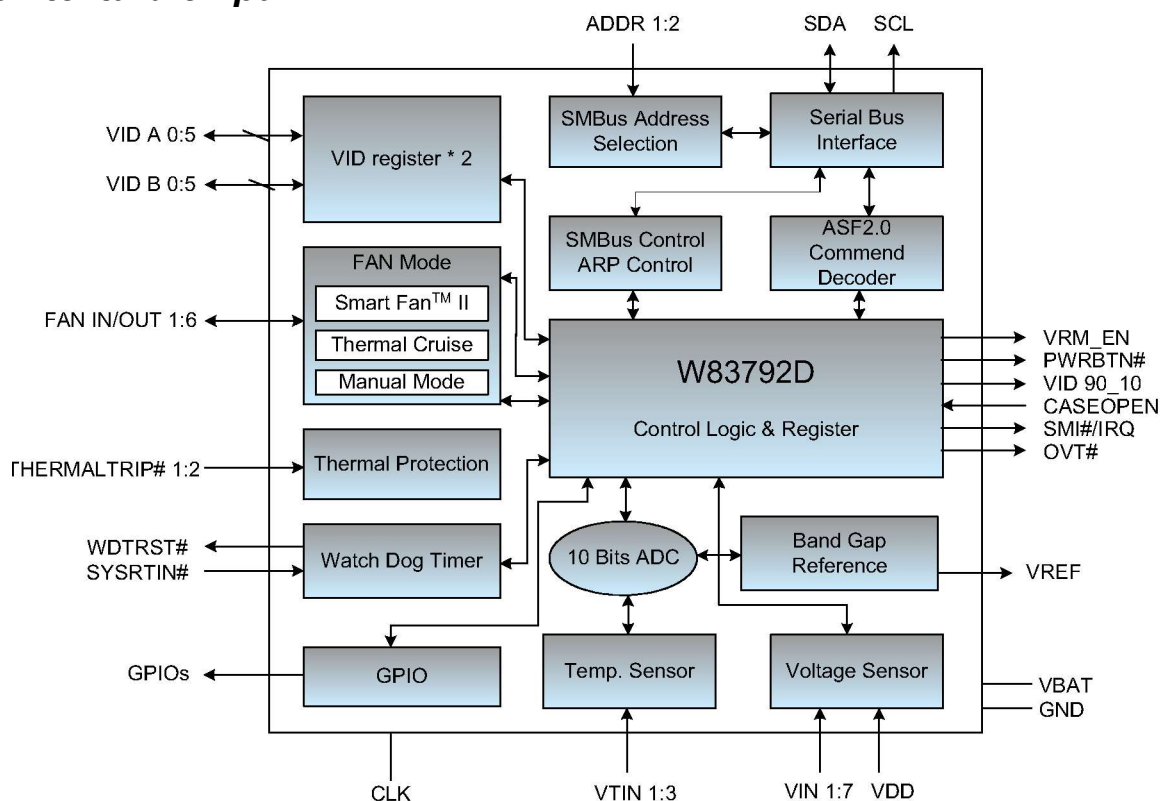
I2C

Je klasická dvou vodičová sběrnice vyvinutá firmou Philips. V počítačích PC se používá její podmnožina pod názvem SMBus. Elektricky se jedná o sběrnici s jedním datovým vodičem a hodinami, oboje s otevřeným kolektorem. Podrobnější informace budou dostupné na přednášce. Z hlediska programátora je situace mnohem jednodušší. Každý chip má svoji adresu a 256 možných osmibitových (nejčastěji) registrů. Každý registr má ve specifikaci chipu definovanou svoji funkci.

ISA

Sběrnice ISA, ač se to nezdá, je pořád v našich PC. Nepoužívá se sice už ve své široké verzi ale jako LPC – low pin count interface. Propojuje SuperIO a jižní můstek. SuperIO je chip obstarávající „legacy“ PC zařízení jako řadič floppy mechaniky, klávesnicový interface atd. Často je integrován i modul pro HW monitoring.

Architektura chipů



Na obrázku vidíte klasického zástupce pokročilých monitorovacích chipů pro serverové desky. Vedle šipek jsou uvedeny názvy jednotlivých vývodů, uvnitř pak jednotlivé bloky, které ulehčí programátorovi orientaci v rozhraních chipu. Pojdme se podívat například na výběr dělitele pro jednotlivé větráčky.

Jak je hezky vidět na obrázku, i v našem případě jsou registry osmibitové. Každý registr je rozdělen do shluků jednotlivých bitů, jejichž význam je definován výrobcem (v druhé tabulce).

8.17 FANIN Divisor Control Registers – Index 47h 5Bh 5Ch(Bank 0)

Mnemonic	Bank	Index	Attr	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
FAN Divisor 1	0	47	RW	FAN2_OB	FAN 2 Divisor			FAN1_OB	FAN 1 Divisor		
				0	1			0	1		
FAN Divisor 2	0	5B	RW	FAN4_OB	FAN 4 Divisor			FAN3_OB	FAN 3 Divisor		
				0	1			0	1		
FAN Divisor 3	0	5C	RW	FAN6_OB	FAN 6 Divisor			FAN5_OB	FAN 5 Divisor		
				0	1			0	1		
FAN 7 Divisor and Control	0	9E	RW	Manual	Trigger	Mask	Reserve	FAN7_OB	FAN 7 Divisor		
				0	0	0	0	0	1		

Reset Condition: Resume Reset, INIT, 5VDD posedge

BIT	NAME	ATTRIBUTE	DESCRIPTION
7,3	FAN_OB	R/W	Enable Fan as Output Buffer. Set to 1, FANOUT can drive logical high or logical low.
6-4, 2-0	FAN_DIV	R/W	FAN PWM Input Divisor. 000 - divided by 1; 001 - divided by 2(Default); 010 - divided by 4; 011 - divided by 8; 100 - divided by 16; 101 - divided by 32; 110 - divided by 64; 111 - divided by 128.

Chceme-li například změnit dělitele pro fan3 na 32, usíme zjistit adresu registru a hodnoty bitů které potřebujeme změnit. Z vrchní části tabulky zjistíme, že náš registr má číslo 0x5B a je nutné změnit bity 2-0. Jejich hodnotu určíme ze spodní tabulky. Ostatní bity je nutné zachovat, takže je musíme správně vymaskovat.

Pokud někdy budeme chtít provést takovou změnu z Linuxu, musíme vědět jakou má chip adresu a k jaké sběrnici je připojen. Nápomocné budou příkazy `i2cdetect` a `i2cset`. Více si pravděpodobně ukážeme na přednášce.

Jak v Linuxu ?

Podporu HW monitoring v Linuxu zastřešuje projekt `lm-sensors`. Domácí stránku naleznete na <http://www.lm-sensors.org>, jakožto i stránky pro uživatelskou podporu a mailinglist.

POZOR: ZAPOMEŇTE NA lm-sensors POKUD MÁTE IBM ThinkPad!!!! Thinkpad obsahuje servisní EEPROM, která při autodetekci změní svůj obsah (je naschvál špatně navržená), BIOS odmítne nastartovat a jediným řešením bez pájení motherboardu je jeho výměna!

Jak na to?

Jako každý hardware i tento potřebuje ovladače. Pro verzi jádra 2.4 jsou k dispozici ze stránek projektu (`lm_sensors-x.x.x.tar.gz` a `i2c-x.x.x.tar.gz`). Uživatelé jádra 2.6 mají situaci jednodušší - stačí jim zkompileovat jen správné moduly. Transparentní přístup k sensorům je uskutečněn pomocí knihovny `libsensors`, takže je nutné v obou případech nainstalovat tuto knihovnu společně s detekčními nástroji z balíku `lm-sensors`. (2.4 uživatelé `make install`, 2.6 uživatelé `make user_install`) Pak stačí spustit skript `sensors-detect` a nechat detekovat chipy na desce. Po detekci nahrajeme požadované moduly a spustíme příkaz: `sensors`

Časté problémy

Q: Popisky u napětí nesedí

A: Pokud některé napětí připomínají podezřelé hodnoty jako 4.90V, 3.23V apod. pak stačí v /

etc/sensors.conf změnit popisky. Pokud jsou napětí úplně mimo, pak je třeba vyzvídat buď u výrobce základní desky (Tyan má k dispozici na webových stránkách soubory sensors.conf) a nebo se obrátit na mailing list. Existuje také dekomparační metoda jejíž nástin je uveden na konci příspěvku.

Q: Záporné napětí nesedí

A: Výrobci nechají jen tak plavat aniž by nepoužité vstupy uzemnili. O jejich soudnosti si rozhodne každý sám.

Q: Teplota se moc nehýbe

A: Možná je špatně nastaven sensors. Zkuste místo termistoru vybrat diodu (v /etc/sensors.conf) Zkuste třeba začít kompilovat jádro a pozorovat změny teploty.

Q: Větráček ukazuje 0 RPM

A: Zkuste změnit dělitel větráčku na 4 nebo 8 (v /etc/sensors.conf)

Q: Sensory nenalezeny

A: Zkuste si projít dostupnou dokumentaci a pokud to nezabere, poradíme na mailing listu. Další věci co a jak ukážu na přednášce.

Pokud nic nezabere jedinou zbývající je možností je thermal zone v acpi. Teplotu procesoru získáte příkazem:

```
ruik@kiur:~/ow/$ acpi -V
    Battery 1: charged, 99%
    Thermal 1: ok, 54.0 degrees C
    AC Adapter 1: on-line
```

Lov na převodní funkce

Tak už víme, že nám výrobce základní desky ten převodní vztah nedá, protože tvrdí že je tajný. Existuje však metoda poslední záchrany a tou je dekompilace BIOSu. Ukážeme si jak takový vztah najít. Tato část je určena pro studenty FELU, pro ostatní je tato část nepovinná. Naším cílem je nalezení instrukcí, které uskutečňují převod žádané veličiny. Hledáme vlastně kód BIOSu, který se spouští když se díváme na sekci „Hardware Monitoring“

ALL YOUR CHIPS ARE BELONG TO US!

Co je třeba

1. znát assembler
2. mít Interactive Disassembler (IDA)
3. BIOS v souboru pro náš motherboard
4. Rozpakovač BIOSu

Ad 1. Pokud se chcete naučit programovat v Assembleru, můžete třeba použít moji knížku. Rudolf Marek, Učíme se programovat v jazyce Assembler pro PC.

Ad 2. IDA freeware lze stáhnout z simtelnet mirrorů:

<ftp://ftp.simtel.net/pub/simtelnet/msdos/disasm/ida37fw.zip>

Pěkné čtení o IDA je například zde: <http://fravia.anticrack.de/howtouse.htm>

Ad 3. Ten stáhneme ze stránek výrobce, potřebujeme 512kB image s BIOSem. Archív ze stránek rozpakujeme, nejčastěji se používá zip nebo rar. Ve výjimečných případech musíme spustit

instalátor ve wine.

Ad 4. Image BIOSu je podobný filesystému, je nutné ho rozbalit. Rozpakovač BIOSu najdeme například na http://www-user.tu.cottbus.de/~kannegv/programm/index_e.htm nebo taky na <http://biosgfx.euro.ru/> Musíme však předem vědět jestli máme BIOS AMI, AWARD nebo Phoenix.

V mém případě byl BIOS AWARD, to znamená že jsem použil tento program:
awardeco.arj 44668 Sun Dec 21 13:04:00 MET 2003

Co dál?

Rozpakujeme BIOS a pokud se to povede dostame podobný výpis.

```
AWARDECO * V.K. * 2000.10.07..2003.12.21
Position packed C target type unpacked filename
-----
000C0000 0001021D -lh5- 5000:0000 System BIOS -> 00020000 original.tmp
000D0246 000037B2 -lh5- 4001:0000 CPU micro code -> 00004826 cpucode.exe
000D3A20 00003FEB -lh5- 6000:0000 -> 00007CD0 AWARDEXT.ROM
000D7A34 00003096 -lh5- A800:0000 -> 00004D3A FILE1.ROM
000DAAF0 0000078F -lh5- A000:0000 -> 00000F90 AWARDEYT.ROM
000DB2A8 00000279 -lh5- 4002:0000 EPA pattern -> 00000642 awardepa.epa
000DB54A 00000D01 -lh5- 4003:0000 ACPI table -> 000026DB ACPITBL.BIN
000DC273 00001293 -lh5- 4007:0000 Virus ROM -> 00001F65 cav_shdw.bin
000DD52F 00008C10 -lh5- 4011:0000 LOGO1 ROM -> 0000E000 ba4019m1.lom
000F2000 00001000 Deco F200:0000 => 00001000 r00F2000.dec
000FE000 00002000 Boot FE00:0000 => 00002000 r00FE000.dec
```

Vytvoří se nové soubory s jednotlivými moduly BIOSu. Některé rozpakovače obsahují příznak typu modulu, pro Phoenix BIOS je dobrým kandidátem modul označený „T“.

```
PHOEDECO * V.K. * 1998.04.02..2003.11.02
dpl126.iwl
Position packed C unpacked type target filename
-----
...
000E2EA5 00002D5D 05 00006710 T 00000000 -> r00E2EA5.DEC
...
```

AMI BIOS má kód, který hledáme v sekci "Setup Client" (type 4). AWARD Bios mívá kód který hledáme ke konci první poloviny hlavního BIOSu tj na konci prvních 64kB toho 128kB modulu.

Prvním krokem budiž vytvoření disassemblovaných souborů BIOSu. Použijeme ndisasm z balíčku NASM. Zvolíme šestnáctibitový režim, protože BIOS funguje v real mode.

```
nasm -b 16 -a original.tmp > original.asm
```

Dobré je znát aspoň IO adresu monitoring chipu, můžeme ji získat z nějakého windows only programu, nebo ji zkusíme tipnout. V našem příkladě je to jednoduché hledáme zařízení na sběrnici ISA takže nemusíme hledat nejdřív něco jako „ovladač I2C sběrnice“.

```
ruik@kiur: /var/lib/dosemu/freedos/aw/pc87366/asm$ grep 0xecd0 *
```

```
original.asm:00011013 BAD0EC mov dx,0xecd0
original.asm:00011025 BAD0EC mov dx,0xecd0
original.asm:0001EC6D 7761 ja 0xecd0
```

To vypadá dobře, v registru DX se předává adresa instrukci OUT (out dx,al in dx,al), jsme tedy na stopě.

Někdy se nepoužívá base adresa ale přímo nějaká „bližší“. Například, pokud je base adresa 0x600, a

napětí začínají na offsetu 0x20, zkusme hledat 0x620. Někdy lze vsadit na konkrétní adresu, třeba na adresu bank switch registru:

```
ruik@kiur: /usr/lib/freedos/illwill/roms/sensors$ grep -A 3 0x629
*.asm
```

```
00001849 BA2906          mov dx,0x629
0000184C B001          mov al,0x1
0000184E EE          out dx,al
0000184F E6ED          out 0xed,al
```

Ano, je to dobré, nyní přijde na řadu IDA, protože ta umí najít zase funkce, které tyhle podprogramy volá (cross reference). AWARD BIOS mívá 128kB plus přídatné moduly. Kvůli segmentové relokači je dobré hlavní soubor rozdělit příkazem dd na dvě 64kB části a postupně je do IDA nahrát. Prvních 64kB nahrajeme na adresu 0xE0000, druhou na 0xF0000. Necháme pracovat autoanalýzu, která najde crossreference. Zaměříme se na kód kolem adresy 0x1849, protože je velká šance že ostatní funkce budou někde poblíž. Pokud autoanalýza kód **nerozkryje** stiskneme klávesu c a nebo u. Klávesu n použijeme pro přejmenování funkcí. Za chvíli dostaneme

```
seg001:1012 inportplusbase proc near ; CODE XREF: readval+4
seg001:1012 ; readval+10.p readval+1A
seg001:1012 push dx
seg001:1013 mov dx, 0ECD0h
seg001:1016 jmp short loc_1000_1035
```

Ted' už budeme potřebovat datasheet abychom věděli jak komunikovat s chipem. Na XREF reference jde klikat a podívat se jaké funkce ty IO funkce používají.

```
seg001:0C93 mov cl, 9
seg001:0C95 mov al, 1
seg001:0C97 call outportplusbase
seg001:0C9A call readval
seg001:0C9D jmp computeexit
```

Tenhle kód čte hodnotu vstupu napětí in1, ale není známá žádná reference pro tuhle funkci. To se klidně může stát, protože se volá z pole pointerů na funkce (jumtable). Ted' se také vyplatí hledat kolem dokola, protože jumtable bude někde poblíž. A je tomu tak:

```
seg001:0C52 push bx
seg001:0C53 push cx
seg001:0C54 push dx
seg001:0C55 call tablehash
seg001:0C58 mov ax, cs:[bx+0Ah]
seg001:0C5C or ax, ax
seg001:0C5E jz loc_1000_C7A
seg001:0C60 call ax
seg001:0C62 cbw ; expand 8 bit in AL to AX
seg001:0C63 mov cl, 10h
seg001:0C65 mul cl ; ;result in AX
seg001:0C67 mov cx, cs:[bx] ; use first constants
seg001:0C6A imul cx ; result in DX:AX
seg001:0C6C mov cx, cs:[bx+2] ; load div constant
seg001:0C70 idiv cx ; DX will be remainder
seg001:0C72 add ax, cs:[bx+4] ; add third constant to AX
seg001:0C76
seg001:0C76 loc_1000_C76: ; CODE XREF:
seg001:0C7D.j
seg001:0C76 pop dx
```

Už jsou tu vidět instrukce pro násobení a dělení (imul a idiv) tj dost možná se jedná o ten vzoreček co hledáme. Nepředbíhejme a spíš se podívejme jak to celé funguje. Inicializační funkce připraví offset do jumtable a pak se zavolá přes pointer ta funkce kterou jsme už našli (call ax), Zdá se že

konstanty pro násobení se také berou z nějaké tabulky, jejíž formát je následující:

```
seg001:0BCB sometable2 dw 1 ; imul
seg001:0BCD dw 1 ; idiv
seg001:0BCF dw 0 ; add
seg001:0BD1 dw 640h ; ???
seg001:0BD3 dw 73Ah ; ???
seg001:0BD5 in2 dw 0CA0h ; caller
```

Funkce tablehash vybírá správný pointer z pole struktur sometable2

```
seg001:0C41 tablehash proc near ; CODE XREF:
seg001:0C55
seg001:0C41 xor bx, bx
seg001:0C43 cmp ch, 6
seg001:0C46 jnb loc_1000_C4D ; jump if CH is equal or bigger
seg001:0C48 mov bl, ch
seg001:0C4A imul bx, 0Eh ; CH*0E
seg001:0C4D
seg001:0C4D loc_1000_C4D: ; CODE XREF:tablehash+5.j
seg001:0C4D add bx, 0BCBh ; add table start
seg001:0C51 retn
seg001:0C51 tablehash endp
```

Jakmile si uvědomíme všechny souvislosti zrekonstruujeme vzoreček a máme vyhráno. Pokud se jedná o termistor, často najedeme překladové pole, které má 256 byte a jednotlivé položky jsou žádané teploty. Pak je nutné funkci zrekonstruovat např. pomocí Openoffice.org nebo jiných matematických programů.

Celá tahle legrace se dá stihnout za večer, ale i tak přeji hodně štěstí. Někdy se to ale zkomplikuje, protože se base adresy nemusí objevit přímo ale jsou součástí dat. Pak musíme hledat všechny IO operace a dívat se odkud se berou parametry.

Závěr

Doufám, že i takhle stručně to stačilo. Jako studijní zdroj doporučuji stránky projektu lm-sensors. Ptát se můžete v mailinglistu na <http://lists.lm-sensors.org> nebo na IRC kanálu #linux-sensors v síti Freenode.net.