Linux extra / 1.časť

Nostalgia

Nemôžem na to nespomínať. Mal som asi 10 rokov, keď som si za pomoci svojho brata Janka postavil prvú kryštálku* (popis pozri v inom bloku). A vtedy som objavil nový svet. Svet elektrotechniky. Ale nie hocijaký, ale taký, ktorý som mohol sám ovládať a ovplyvniť. Mohol som si postaviť všakovaké prístroje a zariadeníčka - od spomínanej kryštálky, cez rôzne zosilňovače, farebnú hudbu, vreckové vysielačky až po celkom zložité zariadenia. Voľakedy sa tento jav označoval slovom *rádioamatérstvo*. Neskôr prišla doba prvých logických integrovaných obvodov. Začalo to obyčajnými hradlami NAND (4 v jednom puzdre = integrovaný obvod MH7400), z ktorých sme stavali prvé digitálne zariadenia (napríklad digitálne hodiny obsahovali až 150 týchto integrovaných obvodov). Potom prišli ďalšie a ďalšie logické obvody, pomocou ktorých sa dali postaviť už celkom inteligentné číslicové zariadenia. Moja diplomová práca v roku 1987 – *automatický telefónny záznamník hovorov* – obsahovala 42 integrovaných obvodov! A tak nastala doba elektroniky a digitálnej techniky. Cítil som, že toto je pre mňa ten správny smer! A keď o pár rokov prišli prvé procesory a s nimi počítače, začal som sa im venovať intenzívnejšie. Mal som pocit, že spojenie digitálneho hardvéru a trošky softvéru je to pravé orechové. Bohužiaľ, dnes je informatika skôr o sofvérových záležitostiach a problémoch, ale v kútiku duše stále cítim, že počítač pracuje naplno iba vtedy, keď niečo riadi, kontroluje a ovláda. Niečo, čo je k nemu pripojené zvonku a nieje jeho integrálnou súčasťou. Je mi smutno, keď zisťujem, že dnešní špičkoví informatici sú machri za klávesnicou, ale boja sa pozrieť dovnútra svojho počítača. A pripojiť nejaký jednoduchý obvod a chcieť po počítači, aby s ním komunikoval, to už je skoro nočná mora. (Ruku na srdce, koľkí z vás, čo čítajú tieto riadky a denne vysedávajú za klávesnicou, by boli schopní k pécečku pripojiť obyčajné tlačidlo a LED diódu? A teplomer? Mobil? Elektromotor alebo dokonca sústruh?). Škoda, že sa kamsi do nenávratna vytratili klasickí rádioamatéri, elektrotechnici a elektronici. Ich nedostatok je už vo svete cítiť. Chýbajú profesionáli, ktorí okrem softvéru dobre poznajú aj elektroniku v klasickom zmysle slova. Takýto špecialisti začínajú byť vo svete hľadaní a výrazne morálne aj finančne oceňovaní. Možno v niektorých z vás spí takýto budúci špecialista. A ja sa týmto seriálom pokúsim tie driemajúce duše zobudiť.

A tak vás vítam pri našom novom seriáli o Linuxe. Ako už názov naznačuje, seriál bude trochu špecifický. Budeme sa zaoberať spojením počítača s inými, možno povedať atypickými zariadeniami. Okrem toho, že tieto zariadenia hardvérovo pripojíme k počítaču, budeme ich aj softvérovo ovládať, akože inak, ako pod Linuxom. Začneme jednoduchými tlačidlami a svetelnými LED diódami, prejdeme na relé, malé niekoľkoriadkové LCD displeje, rôzne teplomery a iné merače a taktiež si vyskúšame pripojiť aj mobilný telefón.

Čo budeme potrebovať

Tak v prvom rade odvahu. Nebáť sa niečo „zbastliť“ a pripojiť to k počítaču je najzákladnejší predpoklad úspechu. Potom trpezlivosť a dôslednosť. Uvedomme si, že budeme pracovať s hardvérom, kde neplatí, že ak sa niečo nevydarí, tak použijeme tlačidlo *reset*. Ešte budeme potrebovať aspoň elementárne základy elektrotechniky v šírke učiva strednej školy a trochu všeobecnej zručnosti. A nakoniec nejaký materiál a softvér.
Z materiálu budeme potrebovať:

- počítač (akože inak) – nemusí byť najšpičkovejší, práve naopak. Pokusy môžeme robiť aj na staršej 486, či dokonca 386. A keď si to odskúšame, prenesieme to na ozajstný „pracovný“ počítač. Keďže sa stáva, že nie všetci majú doma starší počítač, tak si aspoň zožeňme od známych staršiu dosku so sériovým a paralelným portom. Tú zasunieme do počítača, aby sme v prípade nejakej radikálnej chyby nepoškodili porty na základnej doske počítača.
- Pájkovačku – obyčajnú, amatérsku, k tomu cínovú pájku, kolofóniu, kliešte, skrutkovač a iné bežné amatérske náradie. To na to, aby sme tých pár súčiastok mohli podľa danej schémy pospájať. Pokiaľ to bude možné, budeme pracovať metódou „vrabčieho hniezda“, teda vyhneme sa tvorbe plošných spojov. (Samozrejme, že ak ste v tejto veci zbehlí a máte k tvorbe dosiek plošného spoja možnosti, aktivite sa medze nekladú).
- Elektrotechnické súčiastky – podľa danej schémy. Stačia obyčajné, „šuplíkové“ zásoby. Odpory, tlačidlá, LED diódy, konektory a iné. Všetko v cene pár korún, dokonca aj LCD displej sa dá kúpiť za pár stoviek no a starší mobilný telefón tiež nejako lacnejšie zoženieme.

Zo softvéru budeme potrebovať:

- Linux – ľubovolnú distribúciu, ktorú už poznáme.
- Prekladač *gcc*, ktorý je štandardnou súčasťou každej distribúcie Linuxu

- Zdrojové kódy programov, ktorými budeme ovládať vyrobené zariadenia. Tieto kódy si stiahneme z Internetu alebo sa budú nachádzať na CD časopisu PC Revue.

Bezpečnosť pri práci
Musíme si uvedomiť, že počítač je elektrické zariadenie. My však budeme narábať s malými napätiami a prúdmi, takže prinajhoršom môžeme niečo poškodiť. Cez to všetko je opatrnosť na mieste! ***Nesmieme zabudnúť, že sa okrem zdroja v počítači nachádza ešte aj sieťové napätie 220V!*** To je privedené do zdroja počítača a u niektorých typov skriniek je privedené aj na vypínač, umiestnený na priečelí skrinky. Ak však nebudeme siahať tam, kam nemáme, ak budeme opatrne pracovať, nič zlého sa nestane!

Čo ak nie som zručný „amatér"?
Áno, môže sa stať, že niektorého čitateľa zaujme náš seriál tak, že by si chcel všetko vyskúšať, ale naozaj nemá dostatočné znalosti a skúsenosti so stavbou čo aj jednoduchých obvodov.
Čo v takomto prípade?
Stačí, ak požiada niektorého kamaráta – rádioamatéra (a nemusí mať o počítačoch ani šajnu), aby mu zapojenie zbastlil. A keby sa náhodou zaujímal, na čo to chcete, môžete ho zároveň pritiahnuť k Linuxu!

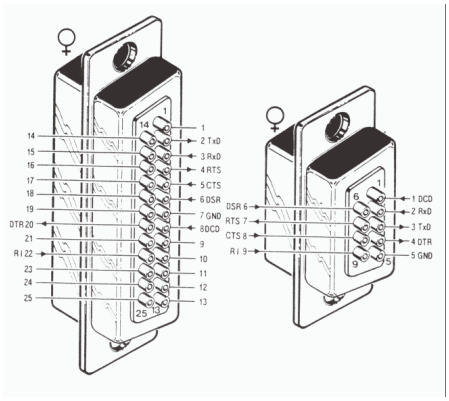
Ako budeme zariadenia pripájať
Ľubovolné zariadenie, ktoré chceme pripojiť k akémukoľvek počítaču, môžeme pripojiť dvoma spôsobmi:
- interne (z vnútra) – keď do volného slotu na základnej doske počítača zasunieme akúsi prídavnú kartu s príslušnou elektronikou
- externe (z vonka) – keď využijeme vonkajšie porty počítača

Vonkajšie porty počítača sú:
- sériový port – slúži prevažne na pripojenie externého modemu, voľakedy slúžil aj na pripojenie sériovej myši a tlačiarne
- paralelný port – slúži spravidla na pripojenie paralelnej tlačiarne
- GAME port – slúži na pripojenie joysticku a iných hracích konzol
- USB port – slúži na pripojenie USB zariadení, najčastejšie USB tlačiarne, digitálneho fotoaparátu, skenera poprípade myši a klávesnice
- PS/2 port – slúži na pripojenie klávesnice a myši

My budeme najčastejšie používať sériový a paralelný port.

Na úvod si veľmi stručne, len pre oboznámenie povedzme niečo o uvedených portoch z technického hľadiska:

Sériový port
Každý počítač má spravidla 1 až dva sériové porty. V prostredí MS Windows sú známe ako COM1 a COM2, v Linuxe ich označujeme ttyS0 a ttyS1.
Na skrinke počítača je tento port vyvedený pomocou konektora CANON25M alebo CANON9M. Tieto konektory majú kolíky a preto im hovorievame aj „samce", ktoré sa odborne označujú písmenom M (ako *male = muž*). Do nich sa zasúvajú podobné konektory – tzv. **protikusy**, ktoré majú prezmenu dutinky (hovoríme im samice). Samice sa označujú písmenom F, čo vychádza zo slova *female = žena*. Číslo za označením konektora udáva počet kontaktov (pinov), teda CANON25 má 25 kontaktov a CANON9 má 9 kontaktov. Príklad 25- a 9-pinového F(emale) konektoru (z letovacej strany!) je na obr.č.1:



Sériový port slúži na sériový prenos. Ten je definovaný určitým protokolom – normou, ktorá sa v odbornej literatúre označuje **RS 232**.

Zapojenie najdôležitejších kontaktov sériového portu je v tab.č.2:

Tab.č.2: Popis vývodov sériového portu RS232

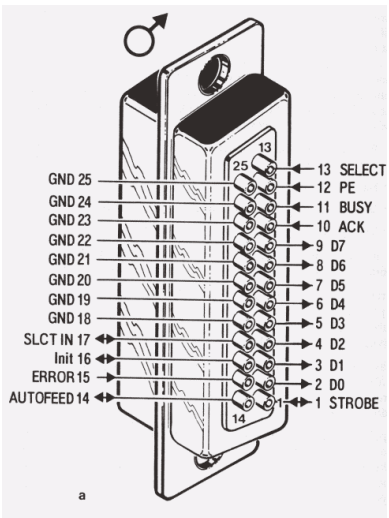
Vývod 25 pin	Vývod 9 pin	Stav	Označenie	Význam	Funkcia
2	3	výstup	TxD	Transmit Data	Vysielané dáta
3	2	vstup	RxD	Receive Data	Prijímané dáta
4	7	výstup	RTS	Request To Send	Výzva k vysielaniu
5	8	vstup	CTS	Clear To Send	Povolené odoslanie
6	6	vstup	DSR	Data Set Ready	Dátová sada pripravená
7	5		GND	Ground	Zem (kostra)
8	1	vstup	DCD	Data Carrier Detect	Detekcia nosnej
20	4	výstup	DTR	Data Terminal Ready	Dátováý terminál pripravený
22	9	vstup	RI	Ring Indicator	Indikátor zvonenia

Sériový port používa na prenos dát iba dva vodiče (okrem spoločnej zeme). Jeden na vysielanie (TxD) a druhý na príjem (RxT). Ostatné vývody sú použité na obsluhu komunikácie.

Elektrické vlastnosti sériového portu
Logická nula je indikovaná napätím od +3 do +25 Voltov. Logická jednotka je indikovaná napätím od –3 do –25 Voltov. Odoberaný prúd nesmie prekročiť 500 mA. Tieto charakteristiky hovoria, že sériové porty sú pomerne odolné k našim pokusom. Okrem toho môžeme z výstupov niektorých pinov využívať napätie na napájanie našich obvodov.
Maximálna dĺžka prívodných kabelov medzi sérivým portom a zariadením je 25 metrov.
S ostatnými pinmi sa podrobnejšie oboznámime pri realizácii konkrétnych zapojení.

Paralelný port
Každý počítač má spravidla iba jeden paralelný port. Ak ho využíva pre tlačiareň, máme smolu. Ale v dnešnej dobe sa tlačiarne začínajú pripájať na USB port, takže paralelný port zostáva volný pre naše pokusy. Paralelný port sa v MS Windows a DOS označuje LPT1, LPT2 a v Linuxe sa označuje lp0, lp1 atď.

Konektor na skrinke počítača je typu CANON25F, takže má dutinky (je to „samica“). Jeho protikusom je teda CANON25M. Príklad tohto konektora (z letovacej strany) je na obr.č.3:



Paralelný port tiež dodržuje dohodnutú normu, označovanú ako **Centronics**.
Popis jednotlivých pinov paralelného portu je v tab.č.4:

Tab.č.4: Popis vývodov paraleleného portu Centronics

Vývod	Stav	Označenie
2	výstup	D0
3	výstup	D1
4	výstup	D2
5	výstup	D3
6	výstup	D4
7	výstup	D5
8	výstup	D6
9	výstup	D7
15	vstup	Error
13	vstup	Select
12	vstup	PE
10	vstup	ACK
11	vstup inv.	Busy
1	vstup/výstup inv.	Strobe
14	vstup/výstup inv.	Auto Feed
16	vstup/výstup	Init
7	vstup/výstup inv.	SLCT IN

Paralelný port využíva na odosielanie a príjem dát osem obojsmerných dátových vodičov. Ostatné signály slúžia pri zabezpečení prenosu.

Elektrické vlastnosti paralelného portu
Paralelný port má klasickú TTL logiku, teda napätie 0 Voltov pre logickú nulu a 5 Voltov pre logickú jednotku. Maximálny prúd je asi 20mA.
Vzhľadom k uvedeným napätiam a prúdom je maximálna vzdialenosť zariadenia od konektoru na základnej doske počítača do 3,5 metra.

Tak si to zhrňme. Na základnej doske počítača alebo na vloženej karte s portami sa môžu nachádzať tieto konektory:
Sériový port:
- 25 pinový CANON s kolíkmi – CANON25M (samce)
- 9 pinový CANON s kolíkmi – CANON9M (samica)

Paralelný port:
- 25 pinový CANON s dutinkami – CANON25F

A teraz si musíme uvedomiť jednu dôležitú vec! Ak chceme do týchto konektorov pripojiť ľubovolné zariadenie, musíme mať na tom zariadení správny protikus!
Teda do sérového portu budeme potrebovať konektory CANON25F a CANON9F (samice) a do paralelného portu konektor CANON25M (samec).

Takže za domácu úlohu si zoženieme tieto spomínané konektory. Kde? Myslím, že skoro v každom obchode s výpočtovou technikou. A môžeme prípadne využiť aj konektory zo starších káblov.

Miroslav Oravec

*Kryštálka:

Kryštálka bolo najjednoduchšie, ale zato funkčné rádio na príjem rozhlasového vysielača s amplitúdovou moduláciou. Na jeho zostavenie stačila obyčajná slúchadlová vložka, čo sa ešte aj dnes montuje do bytových telefónov, jedna germániová dióda a ešte kúsok drôtiku. Potom už len stačilo drôtik priložiť na ľubovolný kovový predmet, napr. odkvap, radiátor, vodovodné potrubie a podobne a zo slúchadlovej vložky sa ozývalo vysielanie najbližšej a najsilnejšej rozhlasovej stanice.
K tomuto zariadeníčku bolo možné (čo bolo, aj stále je!) pripojiť jednoduchý rezonančný obvod a už sa dali stanice vyberať ladením. A to všetko bez nutnosti batérie!

A prečo názov kryštálka?
Voľakedy, v úplných začiatkoch polovodičov sa najprv používal malinký kryštál germánia, o ktorý sa opieral zlatý hrot. Na tomto prechode dochádzalo k demodulácii prijímaného signálu. Až neskôr sa kryštál s hrotom nahradil germániovou diódou.
(Keď si spomínam, ako som tíško čučieval pod perinou s uchom prilepeným k slúchadlu a počúval vysielanie z Banskej Bystrice, je mi akosi teplo u srdiečka a je mi trochu smutno za tým bezstarostným životom malého chalana...)

Linux extra / 2.časť

V minulej časti sme si povedali niečo o pripájaní externých zariadení k počítaču. Vieme, že najčastejšie budeme používať paralelný a sériový port a už vieme, aké konektory k tomu potrebujeme.
Dnes pristúpime k nášmu prvému praktickému cvičeniu. Zo začiatku sa ešte troška rozcvičíme na klávesnici, ale k záveru článku sa vrhneme na stavbu prvého externe pripojeného zariadenia.

Využitie
Veľmi často sa stáva, že máme v správe viac serverov a keďže bežia spoľahlivo (akože ináč!), nie je potrebné mať pri každom z nich pripojený monitor (z ekonomických dôvodov), a každú chvíľu sledovať výpisy obrazovky alebo študovať logy. Cez to všetko by bolo dobré, aby sme boli aspoň jednoducho opticky prípadne zvukovo informovaní o stave servera. Najvhodnejším vizuálnym nástrojom je LED dióda. Ak sa dobre popozeráme okolo seba, na klávesnici sa nachádzajú tri LED diódy, ktoré sú relatívne voľné a tak by sme ich mohli k tomuto alebo podobnému účelu použiť. Sú označené ***NumLock***, ***CapsLock*** a ***ScrollLock***.
Ich význam teraz nie je pre nás podstatný, dôležité je, že sa musíme naučiť ich jednoducho ovládať.
Bolo by dobre, keby existoval nejaký prostriedok, ktorý by dokázal rozsvecovať a zhasínať jednotlivé diódy na základe našich príkazov a ešte lepšie by bolo, keby vedeli tie diódy aj blikať, no nie?

ledblink
ledblink je ten správny program, ktorý potrebujeme! Nájdeme ho na internetovej stránke *Joerqa Mensmanna* na adrese http://www.bitplanet.de/unix/ledblink.html. Nachádzajú sa tu nielen balíčky zdrojových kódov v tvare *tar.gz*, ale aj RPM balíčky, bohužiaľ iba pre RedHat 7.2. My si stiahneme zdrojový balíček, lebo sme už v Linuxe skúsenejší a prekladu sa vôbec, ale vôbec nebojíme, však nie?

Inštalácia programu
Súbor ***ledblink-0.85.tar.gz*** si uložíme do vhodného podadresára. Ja si zvyknem pre takéto účely vytvoriť adresár */install/* v hlavnom koreni súborového stromu a v ňom vytváram príslušné podadresáre, v tomto prípade */install/ledblink/*.
Prejdeme teda do tohto podadresára a ztarovaný balíček rozbalíme príkazom

```
[root@doma ledblink]# tar xzvf ledblink-0.85.tar.gz
```

kedy sa vytvorí podadresár */ledblink-0.85/* (plná cesta je */install/ledblink/ledblink-0.85/*) a v ňom sa nachádzajú príslušné zdrojové kódy.
Prejdeme do tohto podadresára príkazom

```
[root@doma ledblink]# cd ledblink-0.85
```

a ideme kompilovať.

Tento program je veľmi jednoduchý (a preto tradičný prvý krok *./configure* vynecháme) a spustíme príkaz

```
[root@doma ledblink-0.85]# make
```

Preklad je veľmi krátky – iba trojriadkový a pristúpime k inštalácii príkazom

```
[root@doma ledblink-0.85]# make install
```

Výsledkom je vytvorenie tzv. binárky s názvom **ledblink**, ktorá sa nachádza v adresári */usr/local/bin/*.

Použitie programu
Program dokáže ovládať LED diódy na klávesnici alebo pripojené na paralelný port počítača. Dokáže tieto LED-ky rozsvietiť, zhasnúť alebo rozblikať (z toho aj názov *ledblink*). To blikanie je cyklické a môže byť modifikované, teda počet za sebou idúcich bliknutí nasledovaných krátkou pauzou môžeme nastavovať podľa našej potreby (ak ste teraz neporozumeli, ako som to myslel, netrápte sa, všetko si ukážeme).

Podmienkou je, aby program bežal s právami roota. Ako prvé musíme spustiť inštanciu programu, ktorej úlohou je odchytávať nasledujúce volania programu *ledblink* s príslušnými parametrami.

Potom syntaktický zápis príkazu je

ledblink -<*parameter1*> -<*parameterX*>

Parametre programu
Parametre programu sú uvedené v tabuľke č.1:

Tabuľka č.1 - parametre programu ledblink

Parameter	**Popis**
l<*meno_led_diódy*> (to nieje i, ale "el")	špecifikuje, pre ktorú LED diódu bude vytvorená inštancia programu. Následný parameter *meno_led_diódy* môže obsahovať tieto hodnoty: **num, caps, scroll, parport0** až **parport7.** Ak nezadáme meno diódy, defaultne sa predpokladá hodnota **scroll.**
b	rozbliká danú diódu. Ak zavoláme príkaz viackrát, počet bliknutí danej diódy sa zväčší o jedno.
d	znižuje počet bliknutí o jedno.
s	zastaví blikanie diódy
1 (jednotka)	rozsvieti diódu a nechá ju rozsvietenú (bez blikania)
0 (nula)	zhasne diódu
t	zmení stav diódy (ak svietila, tak zhasne, ak bola zhasnutá, tak sa rozsvieti)
k	zruší inštanciu programu *ledblink* pre danú diódu
q	nebude vypisovať hlásenie o inštancii a PID
v	vypisuje hlásenie o inštancii (skúste vvv)
D	ladiace parametre programu

Predstavme si, že chceme rozblikať svetielko na klávesnici, ktoré je označené ako *NumLock*.

Príkazom

[root@doma root]# ledblink –l num

vytvoríme inštanciu programu *ledblink* pre LED diódu *NumLock*.
Vytvorenie inštancie je potvrdené hlásením: ***ledblink running with PID 10920*** (kde číslo PID bude aktuálne podľa behu nášho systému).

Potom už stačí ovládať túto diódu príslušnými parametrami.

Zadajme príkaz

[root@doma root]# ledblink –l num -b

a vidíme, že dióda *NumLock* sa rozbliká!
Zatiaľ bliká (cyklicky) v rytme *„blik – pauza – blik – pauza“*.

Spusťme tento príkaz ešte raz:

[root@doma root]# ledblink –l num -b

Aha! Teraz dióda bliká v rytme *„blik – blik - pauza – blik – blik – pauza“*.
Ak tretíkrát spustíme tento príkaz, ledočka bude blikať blik – blik – blik – pauza - blik – blik – blik – pauza!
(Ak vás teraz napadá, ako to využiť, tak chvíľku počkajte!)

Teraz prezmenu skúsime počet bliknutí znížiť príkazom

[root@doma root]# ledblink –l num -d

Počet bliknutí sa zmenil na dva. Čo však v takom prípade, ak chceme ovládať aj inú LED diódu, napríklad *CapsLock*?

Žiadny problém! Sekvencia príkazov bude:

```
[root@doma root]# ledblink –l caps
[root@doma root]# ledblink –l caps –b
```

To isté môžeme urobiť aj s diódou *ScrollLock* a na klávesnici budú pekne poblikávať všetky tri svetielka!

Ak chceme blikanie niektorej diódy zastaviť, napríklad *NumLock*, použijeme príkaz

```
[root@doma root]# ledblink –l num –s
```

Svetielko zostane svietiť. Ak ho chceme zhasnúť, použijeme príkaz

```
[root@doma root]# ledblink –l num –0
```

a ak ho chceme zase rozsvietiť (bez blikania), použijeme príkaz

```
[root@doma root]# ledblink –l num –1
```

Zmena stavu sa dosahuje príkazom

```
[root@doma root]# ledblink –l num –t
```

teda ak svetielko svietilo, tak sa zhasne, a ak bolo zhasnuté, tak sa rozsvieti.

Ak budeme chcieť ukončiť ovládanie diódy, musíme odstrániť inštanciu programu *ledblink* pre danú diódu (napr. *NumLock*) príkazom

```
[root@doma root]# ledblink –l num –k
```

Ukončenie inštancie sa oznámi podobným hlásením ako na mojej obrazovke - ***Process 16090 killed***.

Praktické využitie
No takáto farebná hudba z klávesnice je síce sympatická, ale je to – ako môj brácho Janko hovorieva – „hračička“. Skutočný efekt dosiahneme, ak nájdeme naozaj praktické použitie.
A to si ukážeme na dvoch príkladoch :

Príklad č.1
Predstavme si, že používame na našom linuxovom serveríku *Sambu* (predpokladám, že vieme, čo to je) a chceme byť informovaní, či Samba beží alebo nie. Ak beží, tak nech svetielko *CapsLock* trvalo svieti, ak Samba v systéme nebeží, nech toto svetielko bliká v režime **_**_**_ (* = blik, _ = pauza).
A k tomuto účelu si napíšeme skript v *shelli*. Nech sa volá **sambatester** a jeho obsah je na výpise č.2:

```
#!/bin/sh

ledblink -l caps

while true; do
ps ax | grep smbd |grep -v grep
if [ $? = 0 ]
    then
ledblink -l caps -s
ledblink -l caps -1
    else
ledblink -l caps -s
ledblink -l caps -b
ledblink -l caps -b
fi
```

```
sleep 10

done
```

Na prvom riadku (okrem známeho #!/bin/sh) sa spustí inštancia programu *ledblink* pre LED diódu *CapsLock*. Potom v nekonečnom cykle pomocou príkazu ***ps ax*** testujeme, či je démon *smbd* spustený. Podmienkou ***if – fi*** vyhodnocujeme návratovú hodnotu predchádzajúceho príkazu.
Ak je démon *smbd* spustený (návratový kód = nula), tak sa zastaví blikanie (to kvôli tomu, ak náhodou predtým LED-ka blikala - parameter –s) a následne svetielko *CapsLock* trvalo rozsvietime (parameter –1).

Ak démon *smbd* nie je spustený (návratový kód nerovný nule), riadky za slovíčkom ***else*** spôsobia zastavenie prípadného blikania a zároveň spustia blikanie. Keďže parameter –b použijeme dvakrát, dosiahneme blikanie v požadovanom rytme **_**_. Nakoniec si náš skriptík oddýchne 10 sekúnd (*sleep 10*) a pokračuje znova.

Poznámka:
Toto je len vzorový skript. Má niekoľko chýb, ktoré by chcelo odladiť. Ale je ***funkčný****!*

Možno nás napadne, že stačí namiesto démona *smbd* napísať meno iného démona, napr. *httpd* a podobne a už sledujeme aktivitu, alebo lepšie povedané funkčnosť inej služby nášho servera.

Príklad č.2
Ak používame server na komunikáciu či už so svetom alebo len v lokálnej sieti, každý správca sa asi najčastejšie presvedča o funkčnosti pripojenia veľmi známym príkazom **ping**. Ak však nemáme pripojený monitor, tak chceme, aby sme o priechodnosti spojenia s našim náprotivkom boli informovaní svetielkom, napríklad *NumLock*.
Ak je spojenie funkčné, nech je LED-ka rozsvietená, ak je nefunkčné, tak nech zhasne.
Na to si napíšeme skript s názvom **pingtester**, ktorého obsah sa nachádza na výpise č.3:

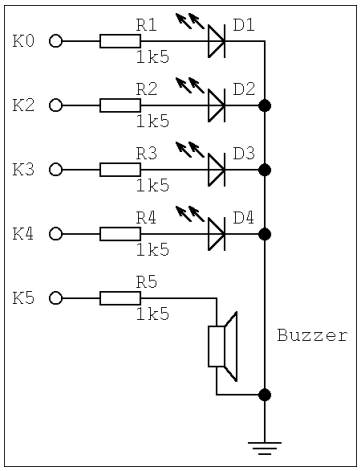
```
#!/bin/sh

ledblink -l num

while true; do

ping www.infoware.sk -c 1
if [ $? = 0 ]
    then
ledblink -l num -1
    else
ledblink -l num -0
fi

sleep 60

done
```

Výpis je veľmi podobný predchádzajúcemu. Na prvom riadku spustíme inštanciu pre diódu *NumLock*.
V nekonečnom cykle „pingáme“ na internetovú stránku www.infoware.sk. Parameter *–c 1* spôsobí, že sa vyšle iba jeden kontrolný paket a príkaz ping sa ukončí. Ak je spojenie funkčné (teda ping prešiel), jeho návratový kód sa rovná nule a pomocou parametra –1 („mínus jedna“) sa dióda rozsvieti. Ak sa ping nevráti, tak parameter –0 (nula) diódu zhasne. A aby sme pri testovaní nezahlcovali zbytočne sieť, nastavíme sleep na 60 sekúnd.

Pozor!
Podobne ako predchádzajúci príklad, aj tento je len náčrt možného riešenia. Buďme si vedomí, že Linux má mnoho možností a že všetko v Linuxe sa javí ako súbor a vo väčšine prípadov ako súbor textový! A na prácu s textom má Linux veľmi silné nástroje, počínajúc **Perlom** a končiac nástrojom **awk** (aj tieto v našom seriáli použijeme). Jednoducho krása! Žiadne registre...

Nezabudnime!
Obidva skripty musia mať nastavený atribút spustiteľnosti! Inak ich môžeme volať koľko chceme a nič sa nebude diať! *Len na pripomenutie – atribúty meníme príkazom chmod!*

Čo keď klávesnica nestačí...
Ak nám tri LED diódy nestačia, môžeme využiť ešte paralelný port počítača. K nemu môžeme pripojiť až osem LED diód. A čo je dôležité, nemusia to byť len LED diódy, ale rôzne relé, spínače alebo buzzer, tak ako to je aj v našom prípade.
Na obrázku č.4 je schéma nášho prvého zapojenia:



Zapojenie je skutočne jednoduché! Použijeme 5 rezistorov R1 až R5, ktorých hodnota nie je kritická a môže sa pohybovať v rozmedzí od 1,5 do 2,2 kiloohmov. LED diódy D1 až D4 sú bežne dostupné, prípadne šuplíkové. Môžeme použiť aj rôzne farby. Každá farba môže signalizovať inú službu alebo udalosť.

Namiesto piatej diódy použijeme akustický piezoelektrický menič so vstavaným multivibrátorom – takzvaný *buzzer*. To je súčiastka veľkosti mince a keď na jej vývody privedieme elektrické napätie, vydáva pisklavý tón pomerne dobrej intenzity. Úpravou vyššie spomínaných skriptov dosiahneme okrem optickej aj zvukovú signalizáciu. Tá je samozrejme podstatne účinnejšia. Predstavme si, že ak vypadne spojenie do Internetu, buzzer začne pípať a my hneď vieme, že sa deje niečo nedobré. Dokonca môžeme pípanie prerušovať, podobne ako blikanie svetielok a podľa rytmu pípania určiť, o akú poruchu sa jedná.

Jednotlivé vývody *K1* až *K5* pripojíme na konektor CANON25M podľa tabuľky č. 5:

Tabuľka č.5 - pripojenie na konektor

kontakt	**K1**	**K2**	**K3**	**K4**	**K5**
dátový vývod	*D0*	*D1*	*D2*	*D3*	*D4*
pin konektora	*2*	*3*	*4*	*5*	*6*

Na rozdiel od klávesnice, k paralelnému portu môžeme pripojiť naraz až 8 signalizačných súčiastok (diód, buzzerov, relé a iných). Jednotlivé dátové piny paralelného portu ovládame parametrom *parport0* až *parport7*.

Ak chceme aktivovať všetkých 5 kontroliek, zadáme túto postupnosť príkazov, ktoré vytvoria inštanciu programu *ledblink* pre jednotlivé piny:

```
ledblink –l parport0
ledblink –l parport1
ledblink –l parport2
ledblink –l parport3
ledblink –l parport4
```

Teraz môžeme zadávať príkazy na ovládanie jednotlivých kontroliek.

Ak chceme rozsvietiť diódu D1, rozblikať diódu D2 a rozpípať buzzer v rytme ***_***_***_, zadáme túto sériu príkazov:

```
ledblink –l parport0 -1
ledblink –l parport1 -b
ledblink –l parport4 -b
ledblink –l parport4 -b
ledblink –l parport4 -b
```

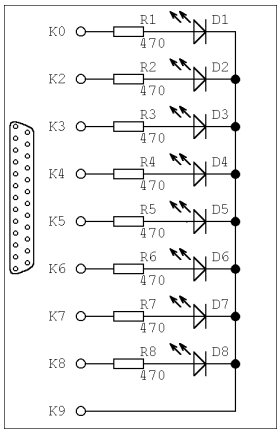
Jednoduché, nie?
Je zrejmé, že nápadov, ako využiť tento jednoduchý program pre rôzne účely, je mnoho. Od jednoduchého sledovania stavu servera, cez kontrolu došlých mejlov až po skutočné ovládanie rôznych zariadení pomocou relé. Verte mi, stačí si len premyslieť, ako na to! „Tak ako všetko v Linuxe“ (moja okrídlená veta!), aj tu môžeme modifikovať skripty viacerými spôsobmi, používať na ich spúšťanie napríklad program **cron** a podobne. Našej fantázii sa medze nekladú.
Technika je mocná čarodejnica!
Teším sa na budúce stretnutie!

Linux extra /3.časť

V minulej časti sme si ukázali ovládanie svetielok na klávesnici a potom sme si to isté vyskúšali s paralelným portom. Použili sme na to program ***ledblink***, ktorý sme kompilovali zo zdrojových kódov a na prácu s portom sme použili už preložené binárky. Ale ako je to v skutočnosti s ovládaním portu po programátorskej stránke? A to si dnes vysvetlíme.

Zapojenie s LED

V minulej časti sme použili zapojenie, kde boli štyri LED diódy a buzzer. Niektorým však tie diódy svietili veľmi slabo. Prečo?
Pri použití odporu o hodnote 1k5 (1500 Ohmov) je možné použiť iba nízkopríkonové diódy. To sú tie novšie a drahšie. Kto však chce použiť „šuplíkové“ zásoby, nech zníži hodnotu odporov na 470 Ohmov.
Pre dnešné naše účely je vhodné mať zapojenie, kde sa nachádza osem LED diód – obr.č.1:

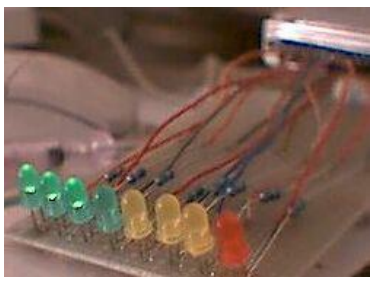


Podľa typu diód použijeme príslušné odpory – hodnoty nie sú nijako kritické – v rozsahu od 330 Ohmov do 2k2. Vývody K0 až K8 pripojíme na dátové piny D0 až D7 paralelného portu, vývod označený K9 pripojíme na spoločnú zem (spoločný vodič, „kostru“, GND), teda na ľubovolný z pinov 18 až 25 – stačí na jeden z nich. Popis vývodov je v tabuľke č.2:

Tabuľka č.2 - pripojenie na konektor

kontakt	**K1**	**K2**	**K3**	**K4**	**K5**	**K6**	**K7**	**K8**	**K9**
dátový vývod	*D0*	*D1*	*D2*	*D3*	*D4*	*D5*	*D6*	*D7*	*Zem*
pin konektora	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*18-25*

Ako také zapojenie môže vyzerať v praxi je vidieť na obrázku č.3:



Niekto môže použiť takzvaný LED-pás, čo je osem LED diód v jednom púzdre vedľa seba. Potom také praktické zariadenie vyzerá ako na obrázku č.4:

Vidíme, že súčiastky sú namontované priamo na konektore paralelného portu, čím sa stáva kompaktnejším. Má aj niekoľko nevýhod – diódy sú rovnakej farby, teda nemôžeme farebne rozlišovať rôzne stavové signály. Druhým drobným nedostatkom je, že konektor s diódami je pripojený na zadnej strane počítača, takže sa môže stať, že nebudeme na diódy vidieť. Variácie konštrukcie ponechávam na vašu zručnosť a fantáziu.

Softvérové ovládanie paraleleného portu.
Ak chceme softvérovo ovládať paralený port, musíme si uvedomiť, že sa jedná o osem- bitový register, ktorý môže dosahovať 2^8 = 256 hodnôt, teda od 0 (nula) do 255. Tieto hodnoty sa zobrazujú v binárnej (dvojkovej) sústave. V tabuľke č.5 sú uvedené hodnoty pre jednotlivé čísla:

Tabuľka č. 5: binárne hodnoty

pin	2	3	4	5	6	7	8	9
dióda	**D1**	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**
hodnota	1	2	4	8	16	32	64	128

Ak pomocou príslušného softvéru pošleme na paralelný port napr. číslo 213, tak sa rozsvieti dióda D8, D7, D5, D3 a D1. Prečo? Lebo 128+64+4+1 = 213. Posledný riadok v tabuľke znázorňuje rozsvietenie príslušných diód.

Adresa portu
Ešte jednu vec ako programátori musíme vedieť – kam máme príslušnú hodnotu poslať?
Paralelný port – tak ako každý port v počítači má svoju ***adresu***. Pre paralelný port sú vyhradené tieto dve adresy:
- 378h - pre prvý paralelný port (v Linuxe lp0, v DOS LPT1)
- 278h – pre druhý paralelný port (v Linuxe lp1, v DOS LPT2)

Programovanie paralelného portu
Programy na ovládanie paralelného portu môžeme napísať v ľubovolnom programovacom jazyku, napríklad v assembleri, v Pascale alebo v Céčku. Ako sa určitý program napíše pre prostredie Microsoftu, to je uvedené vedľa tohto článku v samostatnom odseku. My sa tu budeme venovať Linuxu.
V Linuxe je tiež možné používať assembler alebo pascal, ale Linux ako taký bol napísaný v jazyku C a preto aj my budeme v tomto jazyku pracovať. Nebojte sa, uvidíte, že to bude veľmi jednoduché!

Predstavme si, že chceme mať program, ktorý rozsievit všetky svetielka na paralelnom porte.
My vieme, že na tento port (*lp0*) musíme vyslať číslo 255

Na výpise č.6 je príklad programu v jazyku C pre Linux:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/io.h>

#define base 0x378          /* adresa printer portu */
#define value 255           /* numericka hodnota, poslana na printer port */
```

```
main(int argc, char **argv)
{
   if (ioperm(base,1,1))
    fprintf(stderr, "Nemozem najst paralelny port na adrese %x\n", base), exit(1);

   outb(value, base);
}
```

Preklad programu
Na rozdiel od iných operačných systémov Linux so sebou nesie aj prekladač **gcc** (*GNU C Compiler*). Preto môžeme hneď pristúpiť k prekladu programu.
Uvedený zdrojový text napíšeme v ľubovolnom textovom editore v Linuxe, napr. v programe *vi* alebo v editore, ktorý je súčasťou programu *Midnight Commander*.
Súbor uložíme s menom **lpt_test.c**.

Samotný preklad spustíme zadaním na príkazovom riadku:

[root@rubin root] # gcc –O lpt_test.c –o lpt_test

Program je to veľmi jednoduchý a preto nie sú potrebné žiadne špeciálne parametre prekladača.

Poznámka:
Aby nedošlo k mylnému zápisu príkazu, fonetický zápis znie:

gcc mínus veľké o (O) lpt_test bodka c mínus malé o (o) lpt_test

Načo slúžia dané prepínače kompilera gcc si naštudujte v manuálovej stránke gcc (man gcc).

Nakoniec nastavíme správne atribúty súboru – hlavne atribút spustiteľnosti **x**. Na prístup k portu musí byť program spustený s právami roota!

Je zrejmé, že tento program posiela na paralelný port číslo s hodnotou 255, teda rozsvieti všetky LED diódy. My však chceme, aby sme mohli hodnotu čísla, poslaného na port, meniť, teda zadávať ako parameter programu.

Na výpise č.7 je zdrojový kód takého programu:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/io.h>

#define base 0x378           /* adresa paralelneho portu*/

main(int argc, char **argv)
{
  int value;

  if (argc!=2)
    fprintf(stderr, "Chyba: Zly pocet argumentov. Tento program potrebuje jeden
argument, ktory je cislo medzi 0 a 255.\n"), exit(1);
  if (sscanf(argv[1],"%i",&value)!=1)
    fprintf(stderr, "Chyba: Parameter nie je cislo.\n"), exit(1);
  if ((value<0) || (value>255))
    fprintf(stderr, "chyba: Nespravna hodnota. Parameter musi byt medzi 0 a
255\n"), exit(1);
  if (ioperm(base,1,1))
    fprintf(stderr, "Chyba: Nemozem dosiahnut port na adrese %x\n", base),
exit(1);

  outb((unsigned char)value, base);
}
```

Nemusíme byť zdatní programátori, aby sme pochopili princíp tohto programu.
Vytvorený súbor uložíme, napríklad s názvom **lptout.c**
Pristúpime k prekladu príkazom

```
[root@rubin root] # gcc –O lpt_out.c –o lpt_out
```

Po preklade vznikne program s názvom **lpt_out**.

Použitie programu
Po preklade a nastavení atribútov prístupových práv používame program veľmi jednoducho:
Príkaz

```
[root@rubin root] # lpt_out 0
```

vypne všetky svetielka na zariadení, pripojenom na paralelný port.
Príkaz

```
[root@rubin root] # lpt_out 255
```

pre zmenu všetky LED diódy rozsvieti.

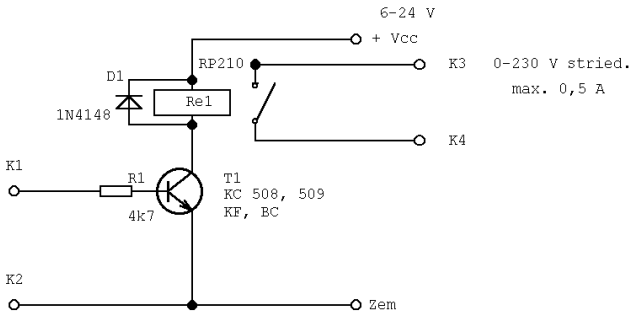
A príkaz

```
[root@rubin root] # lpt_out 1
```

rozsvieti iba jednu – prvú LED diódu.

Parametre programu môžeme zadávať aj v hexadecimálnom tvare, napríklad namiesto 255 zadáme 0xffh.

Spínač s relé
Svetielka sú krásne, ale čo v takom prípade, ak chceme niečo spínať? Vtedy namiesto LED diódy použijeme relé. Keďže relé má pomerne veľký odberový prúd, musíme ho ovládať pomocou tranzistora.
Schéma zapojenia s relé je na obrázku č.8:



Vývod K1 pripojíme na ľubovolný pin konektora, kde je dátový vodič paralelného portu (2 až 9), vývod K2 pripojíme na pin spoločnej zeme (18 až 25). Signál z dátového vodiča prechádza cez bázový odpor R1 o hodnote 4k7 do NPN tranzistoru T1. Relé R1 použijeme podľa potreby a hlavne podľa možnosti spínacích kontaktov. Ak chceme spínať napätie 230V, musíme vybrať vhodné relé. Veľmi dobré relátka sú staršieho typu RP 210, ktoré sa ešte stále dajú kúpiť za dobrú cenu, sú na napätie 6V, 12V a 24 V a dokážu zopnúť striedavé napätie 230V o maximálnom prúde 0,5A. Keď to spočítame (P = U.I), tak môžeme spínať spotrebiče do 100W, napríklad žiarovky. Ak by sme chceli spínať viacpríkonové spotrebiče, musíme zvoliť iný typ relé alebo malým relátkom môžeme spínať výkonový stykač. Keďže relé je cievka a tá môže vyvolávať nežiadúce účinky na tranzistor, pripojíme paralelne k relé ochrannú diódu D1. Vyhovuje typ 1N4148 alebo podobný. Tranzistor T1 môže byť takisto „šuplíkový“ typu KC 508, KC 509 alebo KF prípadne BC. Napájacie napätie celého obvodu závisí od použitého relé, teda ka máme relé na 12V, tak napätie Vcc bude tiež 12V.
Keďže paralelný port má osem výstupov, na jeden port môžeme pripojiť naraz až osem takýchto obvodov s relé a tak ovládať až osem rôznych zariadení.

Postavte si podobný elektronický obvod a vytvorte a skompilujte si program **lpt_out**!
Nielen že ho môžeme okamžite použiť – našej fantázii sa medze nekladú.
Ale my sa k nemu v budúcnosti vrátime a ukážeme si, ako pomocou tohto programu a elektronického obvodu dokážeme ovládať rôzne zariadenia a to aj na diaľku, pomocou Internetu.
Nebude to efektné, ak pomocou Internetu zapneme kávovar v kancelárii, kúrenie na chate alebo automatické polievanie prenajatej pláže v Miami?
A to všetko z ľubovolného miesta na svete, kde sa dostaneme k Internetu.
Ale o tom inokedy.
*A ešte maličkosť – všimli ste si, že jeden a ten istý obvod môžu ovládať dva programy – **ledblink** a **lpt_out**?*

Miroslav Oravec

========== túto časť prosím do samostatného stĺpca či odstavca, pokiaľ možno farebne ==========

Programovanie LPT portu v prostredí MS DOS a MS Windows

Borland Pascal:

Na ovládanie paralelého portu sa veľmi hodí známy programovací jazyk Pascal. Je pomerne výkonný a zrozumiteľný. Najznámejšie prekladače jazyka Pascal sú od firmy Borland. Môžeme použiť Borland Pascal ver.7 alebo staršie verzie tohto produktu pod označením Turbo Pascal 6.
Zdrojový text programu v tomto jazyku bude takýto:

```
Program lpt1_out;

Uses Dos;

Var
   addr:word;
   data:byte;
   e:integer;

Begin
   addr:=MemW[$0040:$0008];
   Val(ParamStr(1),data,e);
   Port[addr]:=data;
End.
```

Program preložíme s menom **lpt1_out.exe**.

Ako parameter zadávame ľubovolné číslo v rozsahu 0 až 255. Taktiež môžeme zadávať hodnoty v hexadecimálnom tvare, ale musíme použiť zápis so znakom $, napríklad namiesto 255 zadáme $FF.

Ak zadáme príkaz

lpt1_out 0

tak zostanú zhasnuté všetky LED diódy na paralelnom porte LPT1.

Ak zadáme príkaz

lpt1_out 255

tak sa všetky svetielka rozsvietia.

No a ak zadáme príkaz

lpt1_out 1

bude svietiť iba prvá dióda D1.

Ak by sme tento program chceli použiť pre druhý paralelný port LPT2, v zdrojovom texte zmeníme riadok *addr:=MemW($0040:$0008)* na riadok *addr:=MemW($0040:$000A).*

Assembler
Assembler je programovací jazyk, ktorý je počítačovému srdcu – mikroprocesoru – najbližší, a to doslova. Preto aj zápis je veľmi stručný:

```
MOV DX,0378H
MOV AL,n
OUT DX,AL
```

kde n je číslo, ktoré chceme na paralelný port poslať a zobraziť.

Borland C++ 3.1
V jazyku C++ bude ten istý príklad vyzerať takto:

```
#include <stdio.h>
#include <dos.h>
#include <conio.h>

/*********************************************/
/*  Tento program nastavuje paralelny port LPT1 */
/*********************************************/

void main (void)
{
clrscr();             /* vymaz obrazovky */
outportb(0x378,0xff); /* posli data na paralelny port */
getch();              /* cakaj na stlacenie klavesu pred skoncenim */
}
```

Tieto programy fungujú v prostredí MS DOS a MS Windows 95 alebo MS Windows 98. Pre verzie operačného systému na báze NT (MS Windows NT, 2000, XP) nemôžeme tieto programy použiť, lebo v týchto systémoch sa pristupuje k paralelnému portu iným spôsobom.

mior

Programovanie LPT portu v prostredí MS DOS a MS Windows

Borland Pascal:

Na ovládanie paralelého portu sa veľmi hodí známy programovací jazyk Pascal. Je pomerne výkonný a zrozumiteľný. Najznámejšie prekladače jazyka Pascal sú od firmy Borland. Môžeme použiť Borland Pascal ver.7 alebo staršie verzie tohto produktu pod označením Turbo Pascal 6.
Zdrojový text programu v tomto jazyku bude takýto:

```
Program lpt1_out;

Uses Dos;

Var
   addr:word;
   data:byte;
   e:integer;
```

```
Begin
   addr:=MemW[$0040:$0008];
   Val(ParamStr(1),data,e);
   Port[addr]:=data;
End.
```

Program preložíme s menom **lpt1_out.exe**.

Ako parameter zadávame ľubovolné číslo v rozsahu 0 až 255. Taktiež môžeme zadávať hodnoty v hexadecimálnom tvare, ale musíme použiť zápis so znakom $, napríklad namiesto 255 zadáme $FF.

Ak zadáme príkaz

lpt1_out 0

tak zostanú zhasnuté všetky LED diódy na paralelnom porte LPT1.

Ak zadáme príkaz

lpt1_out 255

tak sa všetky svetielka rozsvietia.

No a ak zadáme príkaz

lpt1_out 1

bude svietiť iba prvá dióda D1.

Ak by sme tento program chceli použiť pre druhý paralelný port LPT2, v zdrojovom texte zmeníme riadok *addr:=MemW($0040:$0008)* na riadok *addr:=MemW($0040:$000A).*

Assembler
Assembler je programovací jazyk, ktorý je počítačovému srdcu – mikroprocesoru – najbližší, a to doslova. Preto aj zápis je veľmi stručný:

```
MOV DX,0378H
MOV AL,n
OUT DX,AL
```

kde n je číslo, ktoré chceme na paralelný port poslať a zobraziť.

Borland C++ 3.1
V jazyku C++ bude ten istý príklad vyzerať takto:

```
#include <stdio.h>
#include <dos.h>
#include <conio.h>

/*********************************************/
/*  Tento program nastavuje paralelny port LPT1 */
/*********************************************/

void main (void)
{
clrscr();             /* vymaz obrazovky */
outportb(0x378,0xff); /* posli data na paralelny port */
```

```
getch();              /* cakaj na stlacenie klavesu pred skoncenim
*/
}
```

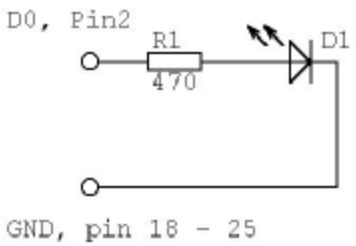
Tieto programy fungujú v prostredí MS DOS a MS Windows 95 alebo MS Windows 98. Pre verzie operačného systému na báze NT (MS Windows NT, 2000, XP) nemôžeme tieto programy použiť, lebo v týchto systémoch sa pristupuje k paralelnému portu iným spôsobom.

Linux extra /4.časť

V minulej časti sme si ukázali, ako vytvoriť program, ktorým dokážeme ovládať elektronický obvod, pripojený k paralelnému portu počítača. Ukázali sme si to v rôznych jazykoch, ale povedali sme si, že Linux je napísaný v jazyku C, tak aj my budeme pracovať v tomto jazyku. Ako sme zistili, ani nepotrebujeme podrobné znalosti programovania, aby sme sa dostali k úspechu. Stačí trochu technickej intuície a pochopenia. Na záver minulej časti sme si spomenuli, že cez Internet dokážeme ovládať kávovar, vzdialený aj desiatky kilometrov alebo dokonca polievať pláž v Miami. Niektorí ste sa posmievali, a preto ich dnes o tom presvedčíme!

Čo budeme potrebovať
No v prvom rade tú pláž! Potom hadicu, vodu, elektromagnetický vodný ventil..... **J**
Potom náš elektronický obvod. A samozrejme počítač s Linuxom.

Ak chceme skutočne niečo ovládať, spínať či riadiť, elektronický obvod by mal obsahovať nejaký spínací prvok, napríklad relé, triak či optický člen a podobne. Podrobnejšie zapojenie závisí od konkrétneho použitia.
Ak si ovládanie chceme iba odskúšať, postačí nám aj obvod s LED diódami, ba dokonca postačuje iba jedna jediná LED dióda a jeden odpor, tak ako je to na obrázku č.1:



Princíp činnosti
Najprv si povedzme, ako také ovládanie spotrebičov pomocou Internetu funguje. Najprv treba poznamenať, že slovo „Internetu“ je tu trochu zavádzajúce. Správnejšie by bolo povedať, že niečo chceme ovládať pomocou webu. Je to širší pojem, lebo nemusíme využívať rovno Internet, niekomu stačí aj podnikový Intranet alebo školská sieť. Takže k tomu ovládaniu.
Skladá sa z niekoľkých krokov:

- používateľ pomocou internetového prehliadača požiada web server o konkrétnu stránku
- server stránku vyhľadá , zašle ju prehliadaču a ten ju zobrazí
- na stránke sa nachádzajú určité ovládacie prvky (tlačidlá, formuláre, prepínače a podobne)
- kliknutím na ovládací prvok odošleme http (web) serveru informáciu, že chceme vykonať určitú činnosť, zviazanú s daným ovládacím prvkom na stránke
- server vyhodnotí informáciu a ak je korektná, vydá povel hardvérovému zariadeniu, aby vykonalo stanovenú činnosť

Z vyššie uvedeného popisu vidíme, že sa systém ovládania cez rozhranie web skladá z dvoch častí:

- z ovládania hardvéhového zariadenia
- z webového rozhrania

Ovládanie hardvérového zariadenia
Spomeňme si na náš program *lptout.c* – výpis č.2:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/io.h>

#define base 0x378          /* printer port adresa */

main(int argc, char **argv)
```

```
{
  int value;

  if (argc!=2)
    fprintf(stderr, "Chyba: Zle cislo argumentu. Tento program potrebuje jeden argument ktory je cislo medzi 0 a 255.\n"),
exit(1);
  if (sscanf(argv[1],"%i",&value)!=1)
    fprintf(stderr, "Chyba: Parameter nie je cislo.\n"), exit(1);
  if ((value<0) || (value>255))
    fprintf(stderr, "chyba: Nespravna hodnota. Parameter musi byt medzi 0 a 255\n"), exit(1);
  if (ioperm(base,1,1))
    fprintf(stderr, "Chyba: Nemozem dosiahnut port na adrese %x\n", base), exit(1);

  outb((unsigned char)value, base);
}
```

Pomocou príkazu

```
[root@rubin root] # gcc –O lptout.c –o lptout
```

po preklade vznikne program s názvom **lptout**.

Ak tento program zavoláme s parametrom rôznym od nuly (teda s hodnotou medzi 1 až 255), LED dióda v obvode podľa obrázku č.1 bude svietiť.
Ak použijeme parameter rovný nule, LED dióda zhasne.

Inštalácia súboru
Po preklade tento súbor príkazom

```
[root@rubin root] # cp lptout.c /usr/sbin/lptout
```

prekopírujeme do adresára /usr/sbin/.

Nasledujúcim príkazom

```
[root@rubin root] # chmod +s /usr/sbin/lptout
```

zabezpečíme, aby tento program mohol spúšťať každý používateľ s právami roota.

Týmto máme časť ovládania hardvéru za sebou.

Webové rozhranie
K webovému rozhraniu budeme potrebovať linuxový server s niekoľkými súčasťami.
Sú to:

- http server, napríklad ***Apache***. Ten je súčasťou každej distribúcie Linuxu a býva spravidla prednastavený, stačí už len doplniť pár maličkostí do konfiguračného súboru (týmto sa dnes zaoberať nebudeme, predpokladám, že http server máme funkčný a v prevádzke)
- web stránku, na ktorej sa budú nachádzať ovládacie tlačidlá
- nejaký program, ktorý bude vykonávať prostredníka medzi webovým rozhraním a hardvérovým ovládaním. Takýmto programom sa hovorieva **CGI skripty**. CGI skript (Common Gateway Interface) je externý program, ktorý je na základe požiadavky od prehliadača spustený ako samostatný proces a vykonáva konkrétnu činnosť. CGI skript môže byť napísaný v ľubovolnom programovacom jazyku, v PHP, v HTML alebo v príkazoch shellu. A práve toto my potrebujeme.

Naša ovládacia web stránka
Nebudem vás trápiť, na výpise č.3 je vzor našej stránky v jazyku HTML:

```
<html>
<head>
<title>Test ovladania paralelneho portu</title>
</head>
<body>
```

```
<h1>Test ovladania paralelneho portu</h1>
<p>
<form action="/cgi-bin/lpton.cgi" method="get" name="on">
<input type=submit value="Rozsviet LED diodu">
</form>
<p>
<form action="/cgi-bin/lptoff.cgi" method="get" name="off">
<input type=submit value="Zhasni LED diodu   ">
</form>
<p>
</body>
</html>
```

Stránku pomenujeme **index.html**.
Pozrime sa teraz na stránku podrobnejšie:
Okrem hlavičky a titulku obsahuje iba tri prvky:

- nadpis “Test ovladania paralelneho portu” (tag <h1>)
- tlačidlo s popisom “Rozsviet LED diodu” (tag form)
- tlačidlo s popisom “Zhasni LED diodu”

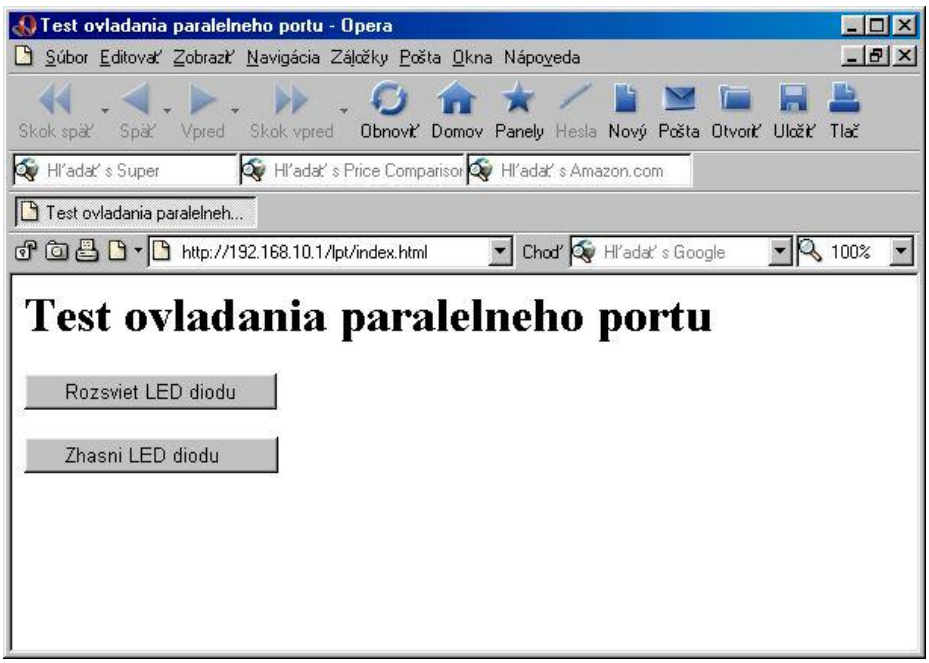
Stránku teraz uložíme do adresára http servera. Spravidla to býva adresár /var/www/html/, hlavne v distribúciách založených na RedHate alebo Fedore. Ak máte vo svojej distribúcii iný adresár, uložte súbor tam.
V tomto adresári vytvoríme podadresár /lpt/, ktorému nastavíme patričné prístupové práva:

```
[root@rubin root] # cd /var/www/html/
[root@rubin root] # mkdir lpt
[root@rubin root] # chmod go+rx lpt
```

Potom súbor *index.html* prekopírujeme a nastavíme príslušné prístupové práva:

```
[root@rubin root] # cp index.html /var/www/lpt/index.html
[root@rubin root] # chmod go+r /var/www/html/lpt/index.html
```

Pre kontrolu spustíme internetový prehliadač, do ktorého zadáme adresu v tvare *http://meno_servera/lpt/index.html* alebo namiesto mena servera môžeme zadať jeho IP adresu, teda napríklad *192.168.10.1/lpt/index.html* – obr.č.4:



Na obrazovke v okne prehliadača uvidíme vyššie spomenuté prvky – nadpis a dve ovládacie tlačidlá.
Kliknutím na tlačidlá sa ešte nič nevykoná, lebo nemáme vytvorené CGI skripty.

CGI skripty
Pozrime sa ešte raz na výpis č.5:
Kliknutím na tlačidlo s nápisom „Rozsviet LED diodu“ sa spustí CGI skript s menom *lpton.cgi* v podadresári /cgi-bin/.
Kliknutím na tlačidlo s nápisom „Zhasni LED diodu“ sa prezmenu spustí CGI skript s menom *lptoff.cgi* v tom istom adresári /cgi-bin/.
Adresár /cgi-bin/ je špeciálny podadresár patriaci http serveru Apache a jeho úplná cesta je /var/www/cgi-bin/. Slúži na uloženie a spúšťanie CGI skriptov.

Vytvorme si teraz jednotlivé CGI skripty.
Na výpise č.4 je obsah súboru *lpton.cgi*:

```
#!/bin/sh
# Parallel port CGI script
#
# Send HTTP headers
echo Content-type: text/html;charset=ISO-8859
echo
# Ovladaj - zapni
/usr/sbin/lptout 0xff
# web data
echo "<html><head></head><body>"
echo "Svetielko rozsvietene<br>"
echo "<a href=\"/lpt/index.html\">Chod nazad na hlavnu stranku</a>"
echo "</body></html>"
#
```

Všimnime si, že je napísaný v shelli – to spoznáme hneď podľa prvého riadku *#!/bin/sh*. Okrem toho obsahuje prvky jazyka HTML. To preto, aby sme ho mohli vidieť v prehliadači.

Pre nás je veľmi dôležitý riadok

```
/usr/sbin/lptout 0xff
```

Práve týmto riadkom voláme náš dobre známy program *lptout* z výpisu č.2!
To je to rozhranie – interfejs medzi webom a ovládaním hardvéru.

Na výpise č.6 je výpis obsahu programu lptoff.cgi:

```
#!/bin/sh
# Parallel port CGI script
#
# Send HTTP headers
echo Content-type: text/html;charset=ISO-8859
echo
# Ovladaj - zhasni
/usr/sbin/lptout 0x00
# web data
echo "<html><head></head><body>"
echo "Svetielko zhasnute<br>"
echo "<a href=\"/lpt/index.html\">Chod nazad na hlavnu stranku</a>"
echo "</body></html>"
#
```

Všimnime si, že sú úplne rovnaké, až na riadok

```
/usr/sbin/lptout 0x00
```

teda líšia sa iba v parametri programu *lptout*. My vieme, že 0x00 a 0xff je hexadecimálny zápis pre hodnoty 0 a 255.

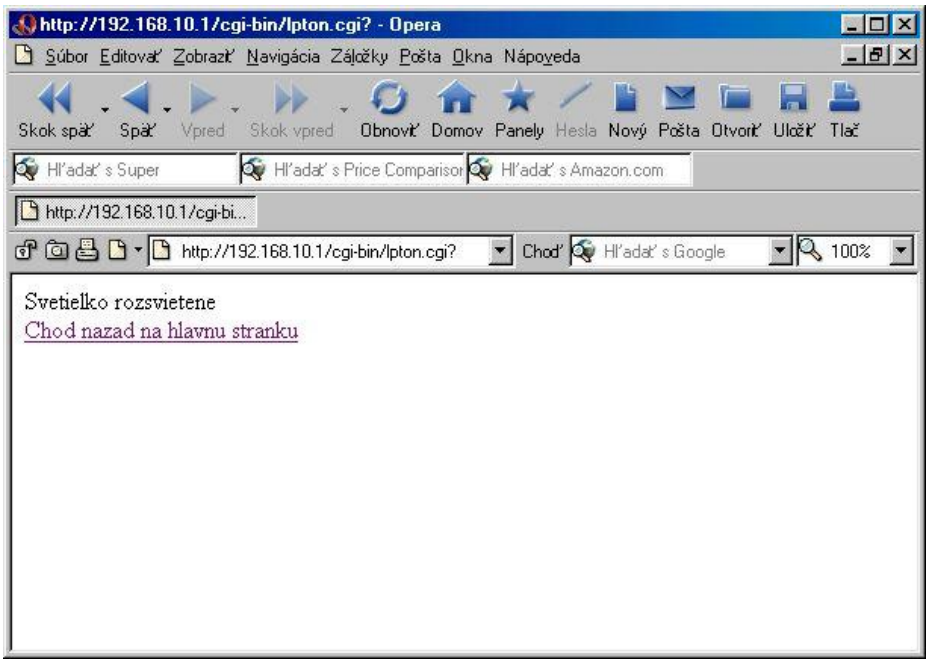
Obidva programy po napísaní musíme uložiť do adresára /var/www/cgi-bin/ a nastaviť príslušné práva. To dosiahneme sériou príkazov:

```
[root@rubin root] # cp lpton.cgi /var/www/cgi-bin/lpton.cgi
[root@rubin root] # cp lptoff.cgi /var/www/cgi-bin/lptoff.cgi
[root@rubin root] # chmod go+rx /var/www/cgi-bin/lpton.cgi
[root@rubin root] # chmod go+rx /var/www/cgi-bin/lptoff.cgi
```

A máme hotovo!

Teraz už len pripojiť náš obvod k paralelnému portu servera, kde máme tieto súbory uložené a kde je spustený web server Apache.

Znovu zavoláme stránku *index.html* tak, ako je to na obrázku č.4.
Teraz klikneme na prvé tlačidlo. Na obrazovke sa objaví okno č.7:

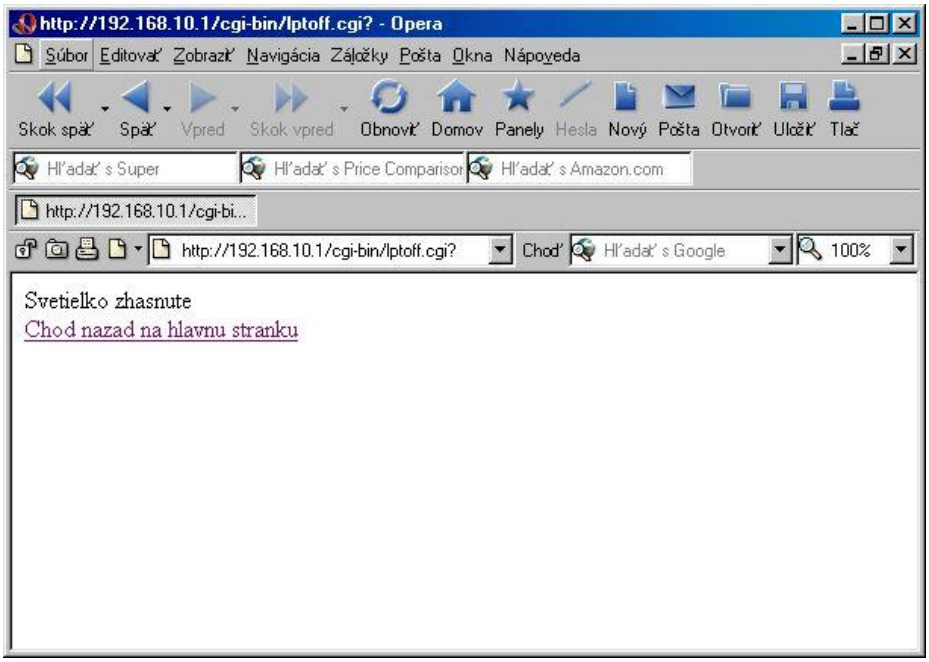


A LED dióda sa rozsvieti!

Keď klikneme na text „Chod nazad na hlavnu stranku“, vrátime sa k obrázku č.4.

Upozornenie!
Stav na paralelnom porte je stavový, To znamená, že jeho stav zostane zachovaný, pokým ho iným príkazom nezmeníme alebo nedôjde k resetu počítača. Preto LED dióda aj po návrate na hlavnú stránku trvalo svieti a to dovtedy, pokým ju nezhasneme!

Teraz prezmenu klikneme na druhé tlačidlo. Na obrazovke sa objaví okno č.8:



Tentoraz LED dióda zhasne!

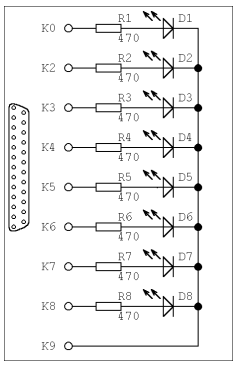
Teraz už len stačí vymeniť LED diódu za obvod s relé a vodným ventilom, premiestniť server na Miami a pekne z domu na Slovensku polievať pláž.
V konečnom dôsledku nás to vyjde lacnejšie ako živý záhradník!

Linux extra /5.časť

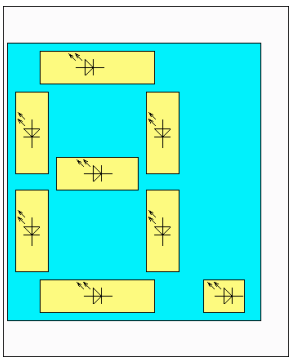
V minulej časti sme sa pohrali trochu s ovládaním zariadení pomocou webového servera a internetového prehliadača. Dokážeme zariadenie zapnúť alebo vypnúť, a to na rôznu vzdialenosť, ktorá je limitovaná len prepojením servera a ľubovolného počítača na Zemi, prípadne vo vesmíre.

Dnes sa budeme ešte chvíľu hrať s paralelným portom, lepšie povedané so svetielkami k nemu pripojenými. Tentoraz sa naučíme signalizovať rôzne stavy počítača. A je jedno, či ten počítač bude server alebo pracovná stanica – desktop. Dôležité je, aby mali v sebe nainštalovaný náš milovaný Linux.

Použijeme zapojenie, ktoré je na obrázku č.1:



Pre dnešné pokusy je lepšie, ak diódy nebudú umiestnené priamo na konektore paralelného portu, ale na viditeľnom mieste niekde na skrinke počítača.
Veľmi efektné je umiestnenie LED diód na prednom paneli, napríklad do plastovej záslepky po veľkej alebo malej disketovej mechanike.
Tí, ktorí nechcú zbytočne nič vyrábať a majú staršie počítačové skrine s číselným displejom, ktorý na starých počítačoch ukazoval rýchlosť procesora, môžu využiť jeden z týchto ukazovateľov. Je to vlastne sedem LED diód zapojených tak, aby sa dalo pomocou nich zobraziť ľubovolná číslica od 0 do 9. My však nebudeme zobrazovať čísla, len využijeme LED diódy - segmenty, z ktorých sa zobrazovač skladá. Posledná, ôsma LED dióda je umiestnená v bodke vedľa číslice – obrázok č.2:



Jedno z riešení je, že káble vyvedieme von zadnou stranou počítača a pripojíme na príslušný konektor.
No našej fantázii sa medze nekladú, technická realizácia zostáva plne na nás a my teraz pristúpime k viacerým softvérovým využitiam tohto zapojenia.

Na sieti sietí Internete nájdeme veľmi veľa programov, ktoré dokážu pomocou uvedenej elektronickej schémy zobrazovať rôzne informácie o stave počítača alebo siete.
My si dnes z celej plejády programov ukážeme možnosti troch dobrých programov:

- **meter**
- **ledstatus**
- **portato**

meter
Program *meter* zobrazuje zaťaženie systému, kde záťaž systému je zobrazovaná veličinou *load*.
Program je už dlhší čas vo verzii 0.2, čo neznačí, že by ho nemal kto vyvíjať, ale naopak – je tak jednoduchý, že niet čo na ňom zlepšovať.

Inštalácia
Súbor si stiahneme na určité miesto na disku a príkazom

```
[root@rubin install] # tar xzvf meter-0.2.tar.gz
```

ho rozbalíme. Vytvorí sa podadresár */meter-0.2/*.
Príkazom

```
[root@rubin install] # cd meter-0.2
```

prejdeme do tohto adresára a zadáme príkaz pre naozaj veľmi jednoduchú kompiláciu

```
[root@rubin meter-0.2] # make
```

Vytvorí sa príslušná binárka s názvom *meter*, ktorú prekopírujeme príkazom

```
[root@rubin meter-0.2] # cp meter /usr/bin/meter
```

do adresára /usr/bin/.

Použitie
Aj keď projekt *meter* je veľmi jednoduchý, má šesť rôznych režimov – módov činnosti a niekoľko ďalších nastavovacích parametrov.
Parametrom *–n* určíme, koľko LED diód budeme na zobrazovanie používať. Štandardná hodnota je 8.
Parametrom *–p* určujeme, ktorý paralelný port sa bude využívať. Štandardná hodnota je LPT1 s adresou 0x378.
Parameter *–t* predstavuje interval obnovy zobrazenia v mikrosekundách. Napríklad –t 1000000 predstavuje obnovu každú sekundu.

Voľbu použitého módu určíme parametrom *–m*. Ten môže dosahovať hodnôt od 0 do 5.

Pozor!
Treba si uvedomiť, že program beží na popredí, čím obsadí jednu konzolu. Preto je vhodnejšie spúšťanie na pozadí, čo dosiahneme znakom & za príkazom, napríklad

```
[root@rubin root] # meter –n 8 –t 500000 –m 3 &
```

Jednotlivé módy
Prvý mód, v skutočnosti **mód 0** (nula) je implicitný a zobrazuje záťaž procesora vysvietením príslušného počtu LED indikátorov, napríklad päť, ako je to na obrázku č.3:

Čím je záťaž väčšia, tým je dlhší rad rozsvietených svetielok.

Mód 1 spôsobuje beh jedného svetielka po páse hore –dole, tak ako to bolo u auta v dobre známom seriáli *Knight Rider*.
Spúšťa sa napríklad príkazom napríklad:

```
[root@rubin root] # meter –t 500000 –m 1 &
```

A priebeh je podobný ako na obrázku č.4:

Mód 2 je veľmi podobný módu 1, len je inverzný, teda nebehá svetielko, ale „tma“ – obr.č.5:

Módy 3 a 4 sú podobné predchádzajúcim, ale pulzné „pobehovanie“ svetielka alebo tmy nie je v obidvoch smeroch, ale len v jednom. Bohužiaľ to nie je dosť možné graficky zobraziť v časopise a tak to treba iba vyskúšať.
Mód 5 spôsobuje náhodný výber predchádzajúcich piatich módov.

Ledstatus
Projekt *ledstatus* vychádza z projektu *meter-0.2* a kopíruje zobrazovanie vyťaženia systému tak, ako to bolo voľakedy na staniciach spoločnosti Silicon Graphics.

Inštalácia je tiež veľmi jednoduchá – rozbalenie, skompilovanie a prekopírovanie do príslušného „výkonného“ adresára:

```
tar xzvf ledstatus-0.1.tgz
cd ledstatus
make
cp ledstatus /usr/bin/ledstatus
```

ledstatus pracuje ako démon, preto nie je potrebný beh na pozadí.
Okrem indikácie vyťaženia sa zobrazuje na poslednej LED dióde aj stav behu systému (beží – nebeží) a preto je vhodné, aby bola inej farby. Beh ostatných svetielok je podobný módu 0 u projektu *meter* – obr.č.6:

Veľmi jednoduché, efektné. Práve sa mi tlačí na jazyk porekadlo – *„V jednoduchosti je krása!“*

Portato
Peter Heist sa pri tvorbe projektu *portato* nechal inšpirovať vedecko-fantastickými filmami zo 60-tych rokov, kedy stav počítačov signalizovalo veľa - veľa svetiel, kde každé svetlo znamenalo inú sledovanú udalosť.
Preto pri stavbe tohto projektu je vhodné vytvoriť indikačný panel s popiskami čo - ktorá dióda signalizuje.
Už z popisu vidíme, je to projekt náročnejší a má oproti vyššie spomínaným aj inú filozofiu.
Stiahneme si program vo verzii 1.2, rozbalíme a skompilujeme:

```
tar xzvf portato-1.2.tar.gz
cd portato-1.2
make
make install
```

Posledný príkaz zabezpečí prekopírovanie vytvorenej príslušnej binárky *portato* do adresára */usr/local/bin/* a prekopírovanie konfiguračného súboru *portato.conf* do adresára */usr/local/etc/*.

Konfigurácia
Konfigurácia projektu *portato* sa vykonáva v súbore *portato.conf*.
Projekt *portato* dokáže sledovať 5 rôznych skupín činností:

- **aktivita procesora**
- **aktivita prístupu k pevným diskom počítača**
- **sledovanie činností prerušení**
- **aktivita prístupu k jednotlivým sieťovým rozhraniam**

Ø činnosť swapu

Tieto skupiny môžeme ešte podrobnejšie deliť na jednotlivé úlohy alebo komponenty počítača.

Aktivita procesora
Syntaktický zápis je ***cpu <type***>.
type môže dosahovať hodnôt:
user – sleduje user aktivitu procesora
system – sleduje system aktivitu procesora
nice – sleduje aktivitu procesora podľa nice procesov
activity – sleduje ľubovolnú aktivitu typu user, system alebo nice
idle – sleduje vyťaženosť procesora

Aktivita prístupu k pevným diskom počítača
Sledovaním činností pevných diskov dostaneme zrazu nie jednu, ale niekoľko kontrolných LED diód.
Syntaktický zápis je ***disk <číslo***>.
Číslo je číslo sledovaného disku a dosahuje hodnôt od 1 až 4. Ak dosadíme nulu, bude sledovaná aktivita ľubovolného disku.

Sledovanie činností prerušení
Ak budeme sledovať prerušenia, uvidíme napríklad každé stlačenie klávesu na klávesnici alebo prístup k zvukovému systému, k sériovým alebo paralelným portom a podobne.
Syntaktický zápis je ***intr <číslo***>.
Číslo je číslo prerušenia v systéme počítača od 1 do 15. Ich stav sa vyčítáva z */proc/interrupts*.
Najdôležitejšie prerušenia sú:

1 - klávesnica
3 – druhý sériový port
4 – prvý sériový port
5 – zvuková karta / druhý paralelný port
7 – zvuková karta / prvý paralelný port
8-9 - rezervované
10-12 – rôzne sieťové, zvukové alebo grafické karty
14 – radič pevných diskov

Aktivita prístupu k jednotlivým sieťovým rozhraniam
Taktiež sledovanie sieťových udalostí nás dokáže informovať o aktivite sieťovej karty, o vysielaní a príjme, prípadne kolízii paketov. Tieto údaje sa preberajú priamo z */proc/ net/dev*.
Syntaktický zápis je:

net_activity <*interface*> - vysielanie alebo príjem dát
net_collisions <*interface*> - kolízie dát
net_receive <*interface*> - príjem dát
net_transmit <*interface*> - vysielanie dát

<*interface*> je sieťové zariadenie, ktoré je systému známe a nachádza sa v */proc/net/dev*, napríklad *eth0, ppp1, sl0* a podobne.

Činnosť swapu
Sledovanie zápisu alebo čítania swapu tiež nie je na zahodenie.
Syntaktický zápis je ***swap <typ***>, kde typ môže byť:
in – zápis do swapu
out – čítanie swapu
activity – čítanie aj zápis

Keďže máme maximálne 8 LED diód, je dôležité správne sa rozhodnúť, čo chceme sledovať. Naše požiadavky nadefinujeme v konfiguračnom súbore *portato.conf*.

Na výpise č.7 je vzorový príklad súboru *portato.conf*:

```
port lpt 1
update 50 msec

pin1 cpu idle
pin2 cpu system
pin3 swap activity
pin4 net_activity eth0
pin5 net_activity eth1
pin6 disk 1
pin7 disk 2
pin8 intr 1
```

Na začiatku nadefinujeme číslo paralelného portu, kde budú LED diódy pripojené a interval obnovovania informácií. Potom jednotlivým pinom (svetielkam) priradíme sledovanú činnosť.

Program *portato* nebeží ako démon, preto ak si nechceme obsadiť jednu konzolu, musíme ho spustiť na pozadí pomocou znaku *ampersand* (&).
Takisto môžeme zabezpečiť automatické spúšťanie vytvorením patričných spúšťacích skriptov.

Musíme uznať, že máme na výber mnoho možností a dokážeme sledovať dostatočné funkcie systému bez toho, aby sme k počítaču mali pripojený monitor a klávesnicu.
Ak ešte pridáme do počítača ďalšiu prídavnú kartu s druhým paralelným portom, dostaneme až 16 sledovacích bodov, čo už naozaj musí stačiť každému.
Záleží len na nás, ako skôr spomínané programy a projekty využijeme, prípadne vzájomne skombinujeme.
A nabudúce sa začneme hrať so sériovým portom.

Uvedené programy, ako aj ďalšie iné veci súvisiace s týmto článkom, si môžete stiahnuť z mojej vynovenej web stránky na novej adrese **www.cevaro.sk** a prípadné otázky mi môžete posielať na adresu ***extra@cevaro.sk***.
(Ak vás zaujalo, prečo používam práve doménu *CEVARO*, tak si toto slovo prečítajte odzadu...)

Linux extra /6.časť

V minulých častiach sme sa venovali paralelnému portu počítača. Často býva tento port obsadený tlačiarňou. Počítač má však aj sériové porty. Bývajú spravidla dva a aj keď býva niekedy jeden obsadený myšou, zostáva nám ešte jeden voľný port na naše pokusy. A tak dnes sa budeme venovať sériovému portu.

Mnohokrát môže nastať prípad, že z rôznych dôvodov potrebujeme konkrétny server jednoducho vypnúť. (Teda, nie úplne tak jednoducho, že vypneme napájanie počítača, ale tak slušne, pomocou príkazu shutdown).
Čo k tomu potrebujeme?
To je jednoduché!
V prvom rade prístup ku klávesnici a potom rootovské heslo.

Čo však v takom prípade, keď máme server alebo viac serverov, ku ktorým nie je pripojená žiadna klávesnica? (to nie je nič nezvyklé vo veľkej serverovni plnej rackov)? Alebo nie sme v dosahu servera, ale je tam iba Paľo, ale tomu nechceme ani za nič povedať rootovské heslo?
Alebo tak ako u nás doma, kde máme linuxový server kvôli prepojeniu do internetu, ale moja internetujúca dcéra ešte nie je schopná (= nechce sa jej!) sadnúť za konzolu, nalogovať sa a „šutdavnúť“ server?
Vtedy by prišlo vhod nejaké čarovné tlačidlo, ktoré by dokázalo po stlačení samo server korektne vypnúť.

A že niečo také – samozrejme – existuje, to si práve teraz ukážeme.

Aby po stlačení tlačidla došlo ku korektnému vypnutiu linuxového servera pomocou príkazu ***shutdown***, musí sa celé zariadenie skladať z elektroniky a príslušného softvéru. A ako sme si už naznačili, celé to pripojíme na sériový port.

Elektronika
Postavíme si malé zariadenie, ktoré sa bude skladať z dvoch LED diód – zelenej a žltej a jedného tlačidla.
Žltá LED dióda pravidelným blikaním potvrdzuje, že je zariadenie funkčné a ovládací softvér je zavedený v pamäti počítača. Zelená dióda trvalým svitom signalizuje, že je možné stlačiť tlačidlo.
Ak na dobu dvoch sekúnd stlačíme tlačidlo, zelená dióda zhasne a server sa korektne vypne.
Keďže sériový port má na svojich vývodoch napätie približne + - 15 Voltov, musíme použiť aj niekoľko ochranných odporov.

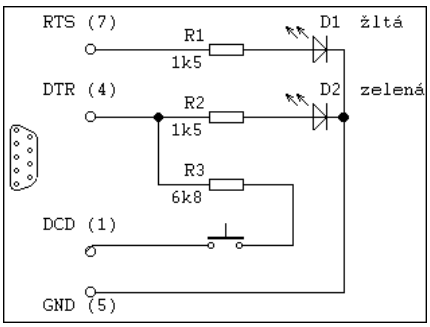
K zostrojeniu celého zariadenia budeme potrebovať tieto súčiastky:

- zelenú LED diódu
- žltú LED diódu
- 2 ks odpor s hodnotou 1,5 kOhmu
- 1 ks odpor s hodnotou 6,8 kOhmu
- 1 ks tlačidlo
- 1 ks konektor do sériového portu (samicu) s káblom

LED diódy použijeme nízkopríkonové, ale môžeme použiť aj staré obyčajné šuplíkové zásoby s tým, že ich svit bude trochu slabší alebo prípadne pozmeníme hodnoty odporov R1 a R2.
Na tlačidle naozaj nezáleží, je len dôležité, aby malo aspoň jeden spínací kontakt.
Konektor použijeme podľa toho, do akého protikusu ho budeme zasúvať. Voľakedy boli v počítačoch 2 sériové porty, jeden s konektorom CANON25M a jeden CANON9M. V poslednej dobe sa na základných doskách počítačov používajú iba dva konektory CANON9M, takže my použijeme konektor CANON9F.
Kábel môže byť podľa normy maximálne 25 metrov dlhý, ale nám istotne vystačí malý kúsok z chvostíka starej počítačovej myši.

Poznámka:
Hodnoty odporov nie sú kritické a môžu sa pohybovať v tolerancii 10%. Aj keď je sériový port relatívne odolný k pokusom, pri stavbe zariadenia buďme opatrní, aby sme si nepoškodili základnú dosku počítača!

Schéma zapojenia obvodu je na obrázku č.1:

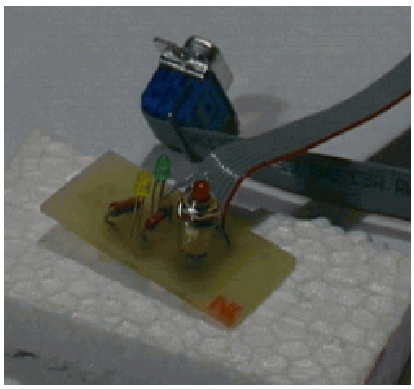


Zariadenie je pripojené na tieto piny sériového portu – tabuľka č.2:

Konektor		***Označenie***	***Popis***
9 pin	***25 pin***		
1	8	DCD	Data Carrier Detect
4	20	DTR	Data Terminal Ready
5	7	GND	Zem
7	4	RTS	Request To Send

(V tabuľke sú uvedené aj piny pre 25 – pinový konektor, keby sme potrebovali pripojiť toto zariadenie k staršiemu počítaču.)

Vzhľadom k jednoduchosti nie je potrebné vyrábať dosku plošného spoja, celé to môžeme postaviť metódou „vrabčieho hniezda“. Technológia stavby ako aj konečný tvar záleží len od našich schopností a možností. Jeden variant hotového prenosného zariadenia je na obrázku č.3:



(Ak chceme vytvoriť stacionárne zariadenie, teda plánujeme jeho používanie pravidelnejšie, môžeme využiť plastikové kryty z neobsadených šácht pre disketovú alebo CD mechaniku.)

Softvér
Ak hotové zariadenie pripojíme k počítaču, nič sa nedeje – LED diódy neblikajú a ani počítač nereaguje na stlačenie tlačidla. K tomu, aby zariadenie ožilo, potrebujeme ovládací softvér.
Ten sa volá *sled-0.3.tar.gz* a nájdete ho na mojej vynovenej stránke ***www.cevaro.sk*** v sekcii *Download*.
Jeho autorom je pán *Guido Socher*.

Inštalácia softvéru
Zdrojové kódy programu **sled** (***serial line led***) si uložíme na vhodné miesto v počítači.

Najprv musíme komprimovaný balíček rozbaliť príkazom:

```
[root@rubin install] # tar xzvf sled-0.3.tar.gz
```

Vytvorí sa nový podadresár s menom *sled-0.3*.

Prejdeme do tohto podadresára príkazom:

```
[root@rubin install] # cd sled-0.3
```

Ak sa nachádzame v uvedenom adresári, prejdeme k samotnej kompilácii.
Keďže sa jedna o naozaj jednoduchý program, nie je potrebná predkompilačná konfigurácia známym príkazom *.configure* a tak pristúpime rovno ku kompilácii príkazom:

```
[root@rubin sled-0.3] # make
```

Kompiláciou vzniknú binárne a pomocné súbory, ktoré umiestnime na určené miesta príkazom:

```
[root@rubin sled-0.3] # make install
```

V adresári */usr/sbin/* sa objaví binárka **sled** a v adresári */etc/rc.d/init.d/* bude súbor **sled_rc** na ovládanie behu programu *sled*.

Špecifikácia portu
Pred spustením samotného programu musíme špecifikovať port, ku ktorému je zariadenie pripojené. Nech je to druhý sériový port, v MS produktoch označovaný ako *COM2* a v Linuxe ako *ttyS1*.
Program *sled* bude hľadať zariadenie z názvom *led*, takže musíme vytvoriť symbolickú linku na *ttyS1*.
To vykonáme príkazom

```
[root@rubin root] # ln –s /dev/ttyS1 /dev/led
```

Prvé spustenie
Aby sme si overili, či naše zariadenie naozaj funguje, spustíme program *sled* s parametrom zariadenia *led*:

```
[root@rubin root] # sled /dev/led
```

Ak začne blikať žltá LED dióda a zelená sa rozsvieti, je to v poriadku.

Proces teraz killneme a vyskúšame ovládanie pomocou štartovacieho skriptu.

Štartovací skript
Obsah štartovacieho skriptu *sled_rc* je na výpise č.4:

```
#!/bin/sh

# Source function library.
. /etc/rc.d/init.d/functions


[ -x /usr/sbin/sled ] || exit 0
DEVICE=/dev/led
RETVAL=0
# See how we were called.
case "$1" in
  start)
        # Start daemons.
        echo -n "Starting sled: "
        daemon sled $DEVICE
       RETVAL=$?
        ;;
  stop)
        # Stop daemons.
        echo -n "Shutting down sled: "
       killproc sled
       RETVAL=$?
        ;;
  status)
       status sled
```

```
       RETVAL=$?
       ;;
  restart|reload)
       $0 stop
       $0 start
       RETVAL=$?
       ;;
  *)
        echo "Usage: sled_rc
{start|stop|restart|reload|status}"
        exit 1
esac

exit $RETVAL
```

Všimnime si, že štartovací skript *sled_rc* je štandardný skript na spúšťanie démonov. Jeho schopnosť spustenia programu *sled* si overíme príkazom

[root@rubin root] # /etc/rc.d/init.d/sled_rc start

a zastavenie príkazom

[root@rubin root] # /etc/rc.d/init.d/sled_rc stop

V distribúciách postavených na RedHate môžeme použiť aj príkaz

[root@rubin root] # service sled_rc start|stop

Automatické spúšťanie skriptu
Ak na počítači plánujeme používanie tohto programu, nesmieme zabudnúť, že sa tento musí samostatne spúšťať už pri štarte systému. Zvlášť je to potrebné u tých počítačov – serverov, ktoré nemajú ani klávesnicu, kde by sme ručné zavádzanie uplatňovali ťažko.
Pre automatické spúšťanie je potrebné vytvoriť príslušné symbolické linky do konkrétnych úrovní behu systému – takzvaných *runlevelov*. Jedná sa hlavne o runlevely 3 a 5.

Vytvoríme linku do runlevelu 3 takto:

[root@rubin root] # ln –s /etc/rc.d/init.d/sled_rc /etc/rc.d/rc3.d/S99sled

a do runlevelu 5 takto:

[root@rubin root] # ln –s /etc/rc.d/init.d/sled_rc /etc/rc.d/rc5.d/S99sled

Tým, že sme týmto linkám pridelili číslo *S99*, dosiahneme toho, že sa bude démon *sled* aktivovať ako posledný. Keď začne blikať žltá LED dióda, vieme, že systém je už kompletne naštartovaný (to je veľmi vhodné najmä vtedy, ak nemáme k serveru pripojený ani monitor, čo býva dosť často).

Modifikácie
Uvedený program aj elektronická schéma sú v popísanej forme funkčné. To by však človek nebol človekom, aby sa nesnažil určitý nápad modifikovať či redefinovať a využiť na niečo podobné alebo úplne iné k obrazu svojmu.

Modifikácia softvéru
Možno niekomu nevyhovuje, že po stlačení tlačidla dôjde k ukončeniu behu celého systému. Možno potrebujeme iba reštartovať niektorú službu (squid, httpd...) alebo len vymeniť cédečko v mechanike.
Vtedy stačí, ak v zdrojovom súbore ***sled.c*** nájdeme rutinu s názvom ***runshutdown*** a prípadne pozmeníme riadok, ktorý je na skrátenom výpise č.5 vyznačený tučným písmom:

```
/* switch off green led and run shutdown
 * the init scripts will finally kill us. We do not
 * terminate here
 */
```

```
int runshutdown(int fd,int *ledstate)
{
      setled( dtr_green, fd, 0, ledstate);
      system("/sbin/shutdown -t2 -h now");
      return (0);
}
```

Všimnime si, že sa jedná o klasický príkaz *shutdown.*
Na toto miesto napíšeme rutinu, ktorá vykoná akciu, očakávanú po stlačení tlačidla. A to nemusíme ani dobre ovládať jazyk C, v ktorom je program sled napísaný! Stačí, ak poznáme príkazy operačného systému alebo na uvedenom riadku zavoláme ďalší program.

Len pre ukážku:
Ak chceme, aby sa po stlačení tlačidla vykonal reštart sieťových nastavení (napríklad po „zaseknutí“ siete), riadok **system("/sbin/shutdown -t2 -h now");** nahradíme riadkom **system("service network restart");**

Modifikácia hardvéru
Okrem toho, že môžeme zmeniť softvérovú funkciu tlačidla, môžeme technicky zabezpečiť náhradu tlačidla iným vhodným elektronickým obvodom.
V takom prípade môžeme počítač ovládať pomocou telefónnej linky alebo mobilu, vysielačky či inak.

Takže popustite uzdu svojej fantázii! Upravte schému a program, dajte to na Internet a prispejete k rozvoju *open source* komunity! A o to nám predsa ide, nie?

Uvedené programy, ako aj ďalšie iné veci súvisiace s týmto článkom, si môžete stiahnuť z mojej vynovenej web stránky na novej adrese ***www.cevaro.sk****, sekcia Download a prípadné otázky mi môžete posielať na adresu* ***extra@cevaro.sk***.

Miroslav Oravec