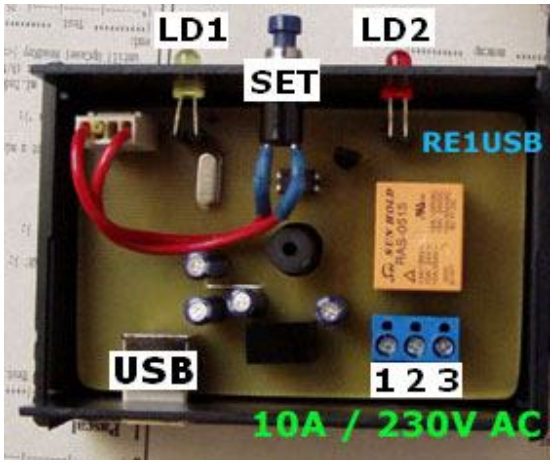
RE1USB/T1 – USB relé s teplotním čidlem

Parametry:

- **USB rozhraní: rychlost 9600bps, 8bitů bez parity, 1 nebo 2 stop bity.**
- **Zatavené teplotní čidlo DS18B20 na 2m kabílku**, rozsah měření od -55°C do 120°C.
- Výstupní svorkovnice - kontakty relé (1-spínací, 2-společný, 3-rozpínací).
- Cívka relé je galvanicky oddělena od USB portu, což zvyšuje odolnost proti rušení.
- Trvalá zatížitelnost relé: 10A / 250V AC.
- Sepnuté relé indikováno trvalým svitem červené LED **LD2.**
- Tlačítko **SET** – krátký stisk přepne výstupní relé.
- Žlutá LED **LD1** bliká v sekundovém intervalu, delší svit – potvrzení přijetí příkazu.



Základní funkce a ovládání

USB rozhraní zajišťuje sofistikovaný obvod FT232RL, čímž je zajištěna kompatibilita se všemi dostupnými operačními systémy. Po prvním připojení k počítači (nadřazenému systému) se zařízení identifikuje jako externí znakové zařízení s rychlostí přenosu 9600bps – virtuální COM.

Zařízení je zakrytováno – propojení s ovládaným spotřebičem (maximálně 10A / 230V AC), zátěží, je prostřednictvím modré svorkovnice uvnitř zařízení. Přístup ke svorkovnici po odšroubování krytu – viz obrázek na následující stránce.

1. Komunikace se zařízením – ovládání relé

Sepnutí relé: odesláním řetězce **R1=1s**

Rozepnutí relé: odesláním řetězce **R1=0s**

Přepnutí relé (toggle): odesláním řetězce **R1=Ts**

Okamžité přepnutí relé je možné docílit také stiskem tlačítka SET
(současně je zastaveno časování)

Přepne relé po uplynutí nadefinovaného času: **R1=Xs**

, kde X je čas (2 až 999999) vteřin

Příklady: **R1=9s** … relé Re1 se přepne po 9 vteřinách
R1=2s … relé se přepne po 2 vteřinách
R1=0s … okamžitě vypne relé, viz výše
R1=1s … ihned zapne relé, viz výše
R1=10s … relé se přepne za 10 vteřin
R1=120s … relé se přepne za 2 minuty
R1=3600s … relé se přepne za hodinu
R1=86400s … relé se přepne za 24 hodin

Zapne relé a po uplynutí nadefinovaného času X vypne: **R1=X,1s**

, kde X je čas (1 až 999999) vteřin

Vypne relé a po uplynutí nadefinovaného času X zapne: **R1=X,0s**

, kde X je čas (1 až 999999) vteřin

Příklady: **R1=10,1s** … ihned zapne relé a za deset vteřin vypne
R1=1,0s … ihned vypne relé a za vteřinu zapne
R1=3600,1s … ihned zapne relé a za hodinu vypne
R1=2,1s … ihned zapne relé a za 2 vteřiny vypne
R1=60,1s … ihned zapne relé a za minutu vypne
R1=86400,0s … vypne relé a za 24 hodin zapne

POZN: **RE1USB** rozlišuje velká a malá písmena, ukončení řetězce malým **s** (ASCII 73H)
Začátek řetězce velkým **R** (ASCII 52H). Všechny číslovky v odesílaných řetězcích musí být dle ASCII tabulky (30H pro 0, 31H pro 1, … 39H pro 9).

2. Komunikace se zařízením – čtení teploty (datový formát 1)

Dotaz na teplotu: odesláním řetězce **Rtp1s**

nebo jednoznakový příkaz: odesláním **?**

Odešle zpět kladnou teplotu ve tvaru: **t1=+24.9C**

Odešle zpět zápornou teplotu ve tvaru: **t1=-24.9C**

Obecný formát: **t1=zXX.XC**

Všechny znaky jsou ASCII („0“ = 30H, „1“ = 31H, … „9“ = 39H).

Pro teplotu 0°C platí **t1=0C**

Pro teplotu pod 10°C platí **t1=+4.9C** ,resp. **t1=-4.9C**

Obecný formát: **t1=zX.XC**

3. Komunikace se zařízením – čtení teploty (datový formát 2)

Dotaz na teplotu: odesláním řetězce **Rtemp1s**

Odešle zpět kladnou teplotu ve formátu: **zXXXC**

Pevná délka řetězce v celém měřeném rozsahu (pro 0°C je +000C, -0.3°C je –003C)

Všechny znaky jsou ASCII („0“ = 30H, „1“ = 31H, … „9“ = 39H).

4. Linux – Ovládání USB relé ze skriptu v jazyce Ruby

Že počítač USB zařízení vidí, otestujeme pomocí:

```
lsusb
```

Přidám uživatele do skupiny dialout, aby měl přístup k **/dev/ttyUSB0** a skript tak nemusel běžet jako root.

```
adduser xsouku04 dialout
```

Instalace Ruby a potřebných knihoven pro Ruby.

```
aptitude install ruby
aptitude install ruby-dev ruby-mkrf
gem install serialport
#!/usr/bin/ruby
#encoding: utf-8

require 'serialport'
# Příklad ovládání RE1USBt1
$port= SerialPort.new("/dev/ttyUSB0", 9600,  8, 1, SerialPort::NONE)
$port.read_timeout=5000 # pouze pro RE1USBt1, pokud nám nic
# nepošlou, čekáme maximálně 5 vteřin. Vyhnu se tak nekonečnému "zaseknutí"
# programu pokud je někde chyba, nebo pokud zařízení žádné události neposílá.
# Umožní mi to v mezičase  změnit stav relátka - pokud by bylo třeba
# Nastavení níže nebylo potřeba, ale jiné aplikace jej používají, nechám jej tedy
# v poznámce.
#$port.sync = true
```

```
def prectiOdpoved()
  buf=""
  begin
    precteny_znak=$port.getc
    if precteny_znak!=nil then buf+=precteny_znak end
  end while precteny_znak!=nil and precteny_znak!="*"
  # odpovědi končí znakem *, nil znamená, že došlo k timeoutu (něco je špatně)
  return buf
end

# poznámka pro Rubysty (nic důležitého):
#  Pokud bych nechtěl používat globální proměnou $port, mohl bych definovat
funkci:
#x = Proc.new do
#end
#x.call
# v tomto případě by byly uvnitř funkce přístupné i neglobální proměnné.

# následující 4 řádky u RE1USB vynechat - RE1USB nemá vstupy, pouze 1 výstup
$port.write("RESET=Ys")  #  jen pro RE4/5/8USB chceme info o sepnutí/rozepnutí
kontaktu
puts prectiOdpoved()
$port.write("RUN=1s")   # jen pro RE4/5/8USB chceme zasílat informace o změnách
vstupů
puts prectiOdpoved()




while true do
   operace= $port.getc
   if operace="1" then
                    $port.write("R1=1s") # zapnu rele 1
                  end
   if operace="0" then
                    $port.write("R1=0s") # vypnu rele 1
                  end
end
```

Další info na fóru http://www.odorik.cz/w/selfcontrol

Další info na fóru http://forum.odorik.cz/viewtopic.php?f=33&t=3122&start=20#p24768

www.selfcontrol.cz