Samova kuchařka

Komunikace výrobků ELSACO protokolem SAM a Epsnet

s využitím programu TESTER pro konfiguraci výrobků

a příklady ve FREDovi pro PL2

ELSACO

Jaselská 177, 280 02 KOLÍN 3
tel./fax: 321 727 753

Verze 2.05 07/2008

© 2008sdružení ELSACO
20.7.2008
Účelová publikace ELSACO

ELSACO, Jaselská 177, 280 02 Kolín 3
Tel./fax/modem: 321 727 753 / 321 727 759

Internet : **www.elsaco.cz**

Připomínky : vacek@elsaco.cz

Obsah kuchařky

1 Info o programu TESTER 5
2 Poznámky na úvod 7
2.1 Obecně k lince RS-485. 7
2.2 Zakončení RS-422/485 u SAMů 7
2.3 Napájení SAMů. 7
3 Panelový číslicový indikátor XDM–14A 8
3.1 Konfigurace XDM-14A 8
3.2 Zobrazení údajů na XDM-14A 9
3.3 Jak na XDM-14A ve FREDovi 10
3.4 COM1
u CCPU-03 a CCPU-21 12
4 Modul SAM–01 pro měření čtyř analogových vstupů 14
4.1 Konfigurace modulu SAM-01 14
4.1.1 INFO v TESTERu 14
4.1.2 Vyčtení verze FW TERMINALem 14
4.1.3 Nastavení starého SAMa TESTERem 14
4.1.4 Nastavení nového SAMa TESTERem 16
4.2 Vyčtení naměřených hodnot 17
4.3 SAM-01 ve FREDovi 18
4.4 SAM-01 a aktivní posílání hodnot. 20
4.4.1 Konfigurace XDM-14A 21
4.4.2 Konfigurace modulu SAM-01 22
4.4.3 Vlastní přenos dat 25
4.4.4 Stejné řešení jinými příkazy 25
5 Modul SAM–02 (4 logické vstupy a 4 relé) 27
5.1 Konfigurace modulu SAM-02 27
5.2 Základní příkazy pro SAM-02 28
5.2.1 Ovládání relé 28
5.2.2 Čtení vstupů 29
5.3 SAM-02 ve FREDovi 30
5.3.1 Spínání relé. 30
5.3.2 Čtení vstupů 31
5.3.3 Projekt 31
6 Modul SAM–04 (16 logických vstupů). 33
6.1 Konfigurace modulu SAM-04 33
6.2 Čtení okamžité hodnoty vstupů. 34
6.3 SAM-04 ve FREDovi 35
7 Interiérový teploměr RAI-01 s komunikací RS-485 38
7.1 Konfigurace RAI-01 38
7.1.1 Úvod 38
7.1.2 USB ovladač 38
7.1.3 INFO z TESTERu 38
7.1.4 SET v TESTERu 39
7.2 RAI-01 ve FREDovi 41
8 Čítač impulzů RCI-01 s komunikací RS-485. 43
8.1 Konfigurace RCI-01 43
8.2 RCI-01 a FRED 45
9 CCPU-03 – přenos dat mezi centrálami po lince RS-485 46

9.1 Propojení centrál 46
9.2 MPC komunikace; help 46
9.3 MPC monomaster. 47
9.4 MPC monomaster - přenos 32 dat 50
9.5 MPC multimaster 53

1 Info o programu TESTER

První verze **Kuchařky** (verze 1.xx) byla napsána ještě před vznikem firemního programu **TESTER** s využitím programu **TERMINAL** pro konfiguraci výrobků s protokolem SAM. To vyžadovalo jistou znalost a zručnost jak s TERMINALem, tak i s manuály výrobků, kde jste nalezli příkazy pro konfiguraci výrobku a museli jste vše provádět "ručně" - bylo to tedy mnohem složitější než nyní, s použitím programu TESTER. Ten práci, zejména vlastní konfiguraci (nastavení) výrobků, značně zjednodušuje - dá se říci, že ji zautomatizuje.

Samozřejmě, že je i nyní důležité zjistit si z manuálu výrobku jeho technické parametry a přesvědčit se na výrobním štítku zejména o hodnotách jeho napájení a zapojení svorek...

Program TESTER je volně ke stažení na našich stránkách *www.elsaco.cz* v sekci *Ke stažení / TESTER*. Program se neinstaluje, pouze kopíruje do zvoleného adresáře. Přitom je důležité zachovat jeho strukturu adresářů, která je na stránkách napovídána v tabulce *Ke stažení*.

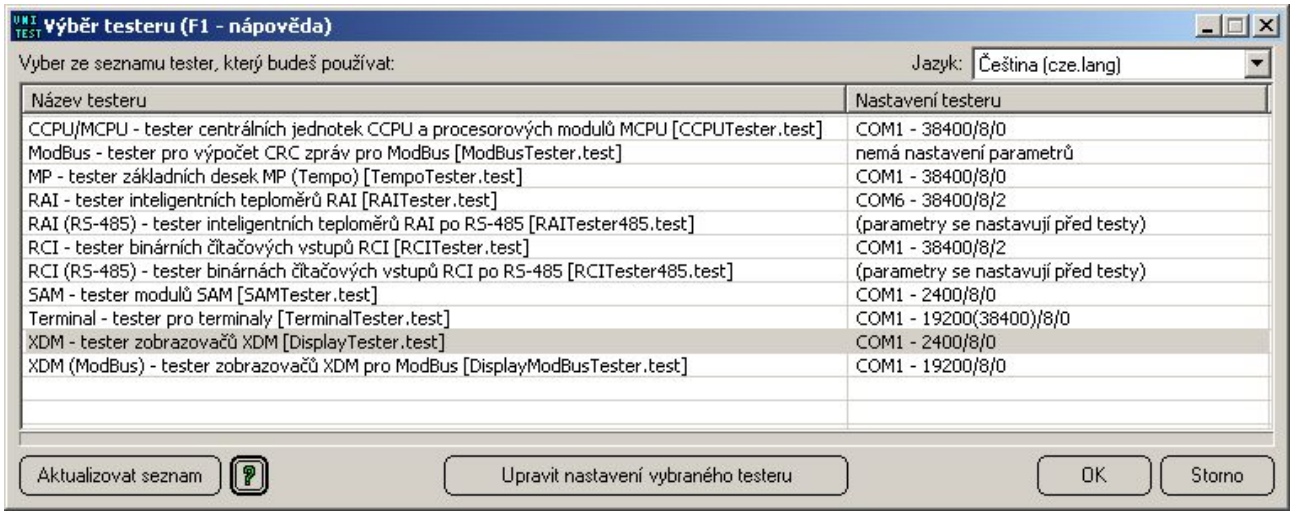
Po startu programu TESTER v nabízeném okně nejprve vyberete výrobek, který chcete konfigurovat, testovat nebo pomocí volby INFO si přečíst jeho nastavení.

Teprve potom má smysl kliknout na volbu **Upravit nastavení vybraného testeru** a nastavit komunikační parametry COMu PC. V podstatě nastavujete pouze číslo COMu PC, protože zmíněné výrobky mají ihned po zapnutí jistý definovaný režim pro svou konfiguraci, a to komunikační rychlost 2400 Bd, bez parity, 8 datových bitů, adresu 0... Ale to nás až tak nemusí zajímat, protože to TESTER ví. Nastavíme COM PC podle následného obrázku 2.



Obr. 2 Nastavení COMu PC

TESTER pozná, zda je volený COM k dispozici, nemějte proto otevřenou (aktivní) jinou aplikaci v PC, která tento COM používá. Pokud si nejste jisti, který COM



Obr. 1 Úvodní okno programu TESTER pro výběr testovaného výrobku

Protože se budeme zabývat výrobky XDM-14A, SAM-01, SAM-02, SAM-04, tj. výrobky komunikující po sériové lince RS-485 protokolem SAM, týká se následný popis pouze jich. Vybereme v úvodním okně jeden z nich, např. XDM - viz obr. 1.

Dole uprostřed úvodního okna programu TESTER máte k dispozici volbu **Upravit nastavení vybraného testeru**, pomocí které lze správně nastavit komunikační parametry.

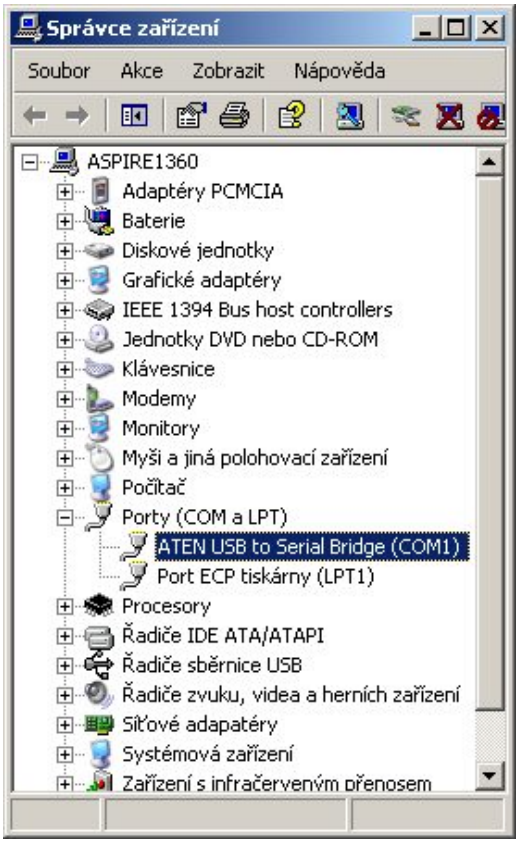
Výrobek musí být jednak připojen na předepsané napájení, jednak jej musíte mít připojen na linku RS-485 přes převodník na RS-232/USB k počítači na jeho COM/USB.

PC je k dispozici (zejména používáte-li USBéčko), otevřete **Ovládací panely** / **Systém**, na kartě **Vlastnosti systému** pak na záložce **Hardware** zvolte **Správce zařízení** a v okně **Správce** uvidíte, které COMy vaše PC zná - viz obrázek 3.

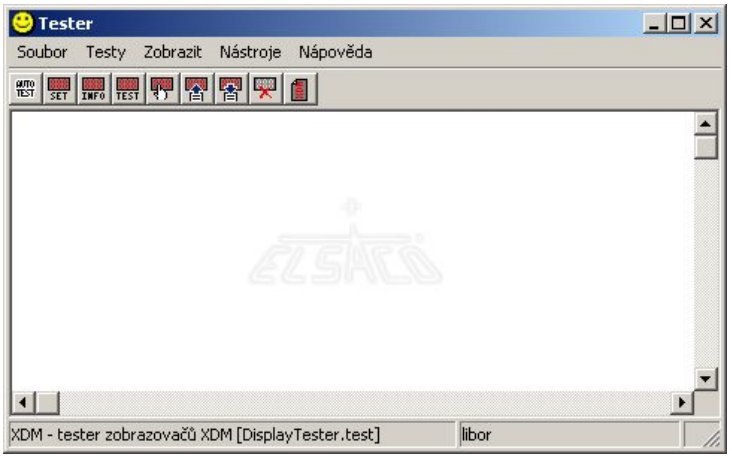
Volbu COMu schválíte a pak v úvodním okně TESTERu kliknete na **OK** pro přechod do příslušného testu výrobku.

My jsme měli zvolen test pro XDM, proto TESTER přechází do okna pro test XDM - viz obrázek 4.

Obecně okno pro test výrobku má nabídky přímo pro testovaný výrobek, jiné samozřejmě ne. Z nabídky volíme požadované, TESTER pak dává pokyny co



Obr. 3 Správce zařízení



Obr. 4 Test pro XDM



Obr. 6 Info o výrobku

máme udělat pro to, abychom získali co potřebujeme. Nemusíme proto znát příkazy, které musíme vykonat - poslat na linku.... Nemusíme je hledat v manuálu, pracně zadávat, nespleteme se ap.

Po každé volbě (u výrobků s protokolem SAM) jsme vyzváni k vypnutí a znovuzapnutí napájení - to proto, abychom se dostali do konfiguračního režimu výrobku. Nejvhodnější postup pro začátek práce s výrobkem je volit v testu výrobku **INFO** a získat tak informace o jeho nastavení. Po kliknutí na volbu **INFO** uvidíme zmíněné okno s výzvou - obr. 5:



Obr. 5 Okno výzvy

Všimněte si, že během tohoto času blikají LEDky na převodníku - TESTER vysílá cyklicky požadavek pro přechod do konfiguračního režimu.

Pokud provedeme požadované znovuzapnutí, přejde výrobek do konfiguračního režimu, tuto odpověď přijme TESTER a vyšle do výrobku zprávu, která požaduje vyčtení jeho nastavení, které je pak následně zobrazeno v dalším dialogovém okně - viz obr. 6.

Poznámka:

Provádíte-li znovuzapnutí výrobku vypínačem na zdroji, mějte v patrnosti, že zdroj díky vnitřním elektrolytům neztratí napětí na výstupu ihned. Proto musíte pár vteřin počkat, aby napájecí napětí výrobku opravdu "zmizelo". Vhodnější je ponechat zdroj stále zapnutý a odpojovat nebo přerušovat napájecí vodič výrobku. Jinak nemusí být odezva na znovuzapnutí výrobku stoprocentní a musíte ji opakovat.

Postup pro ostatní volby programu TESTER je obdobný, k dispozici máte rovněž HELP programu, který popisuje jak volby programu, tak jednotlivé testy.

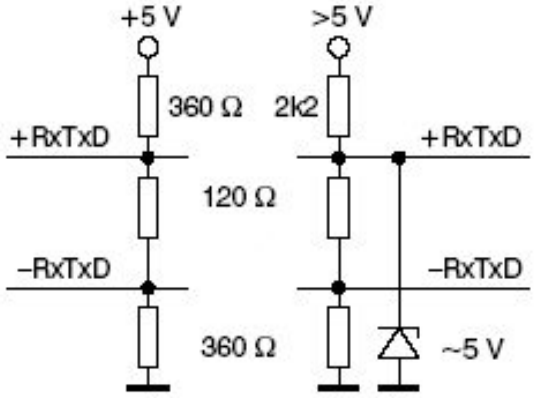
Po tomto krátkém seznámení s TESTERem přikročíme k práci s jednotlivými výrobky s protokolem SAM.

2 Poznámky na úvod

2.1 Obecně k lince RS-485

Většinou se lince RS-485 věnujeme až pokud nastanou problémy s komunikací. Dělejte to naopak - přečtěte si v manuálech, jak má být linka topologicky zapojena a ošetřena a pak se můžete zabývat pouze problémy s protokolem apod. Hodně starostí si tím ušetříte. Nejčastějším opomenutím bývá nesprávné zakončení linky; chcete si ověřit jak se např. pracuje s moduly SAM nebo XDM na stole, naskládáte HW, propojíte linku dvěma vodiči a očekáváte zázraky. Vězte, že pokud i na stole propojíte zhruba víc jak tři moduly linkou RS-485 bez zakončení (při rychlosti 38 kB), nebude komunikace stoprocentní, v horších případech bude vykazovat pouze chyby.

Sběrnice potřebuje zakončení pouze na dvou místech, pokud se dodržuje správná topologie, je to první a poslední jednotka. Zakončení zajišťuje jednak impedanční přizpůsobení, jednak definici klidového stavu linky. Celé to řeší právě kombinace odporů na napájecí napětí spolu s odporem mezi vodiče RS-485. Viz obrázek 7.



Obr. 7 Zakončení linky RS-485

Vytahovací odpory, které používáme typicky 360R (ale není to pravidlem), jsou pro správnou komunikaci nutné, bez nich se nedá zajistit správný klidový stav linky.

Odpor mezi vodiče rozhraní kritický není, stejně obvykle málokdo používá kabely s definovanou impedancí - a tak se nedá mluvit o nějakém přizpůsobení. Navíc používané komunikační rychlosti nejsou žádný fofr, takže nějaké zákmity při změně úrovně nemají dopad na vlastní přenos dat, ale zvedají úroveň vyzařovaného rušení.

Co se dá na sběrnici změřit? Použitelné údaje lze získat pouze při zastavené komunikaci. Potom úroveň napětí na lince vůči GND určují především zakončovací odpory a pokud použijete 3x 360R, naměříte na +485 cca 3,3 V, na -485 cca 1,6 V, mezi dráty taky cca 1,6 V. Tyto údaje by se neměly moc měnit ani při připojování dalších jednotek na linku.

Počet jednotek na lince RS-485 z HW hlediska závisí na použitých budičích a na odporech, které jsou používány na vstupech pro klidový stav. 32 jednotek není problém, dnešní budiče jich podporují 128.

V praxi než propojíme dvě zařízení na lince RS-485 mezi sebou, je vhodné nejprve tato zařízení jen napájet a změřit polaritu vodičů linky RS-485. Tak si ověřit, které vodiče jsou plusové, ty propojit mezi sebou, a které mínusové a ty pak propojit rovněž.

Tento postup zachovávejte i při připojování převodníku kvůli odposlechu či konfiguraci modulu.

A jak poznáme ve FREDovi zda probíhá komunikace po lince RS-485 v pořádku? Každý komunikační modul SERIALCOMM má vpravo dole výstupní pin **ERR** a pokud je na něm hodnota = 1 znamená to, že na poslední požadavek nepřišla správná odpověď. Proto je v projektu aplikace se sériovými moduly vhodné doplnit chybové hlášení o tomto stavu pomocí knihovních modulů HAVBIN.

2.2 Zakončení RS-422/485 u SAMů

Všechny SAMy mají vlevo od napájecích svorek 3 jumpery s popisem **ABC**. Z výroby jsou kolíky jumperů rozpojené (propojky jsou zaparkovány na horním kolíku).

Levá propojka **A** slouží k zařazování vytahovacího odporu proti zemi.

Prostřední propojka **B** slouží k zařazování impedančního přizpůsobení mezi komunikační vodiče.

Pravá propojka **C** slouží k zařazování vytahovacího odporu k plusu napájení.

2.3 Napájení SAMů

Hodnota napájecího napětí u SAMů je vyznačena na štítku. Starší SAMy měly volbu 12 nebo 24V DC, nové SAMy mohou být napájeny v rozmezí 10-30V DC. Na původních štítcích jsou pro hodnotu napájení zaškrtávací políčka **power** pro **12** a pro **24**V. Na starším SAMu je zaškrtnuta jedna hodnota, odpovídající požadovanému napájecímu napětí. Pokud je nový SAM osazen starším štítkem, jsou na něm zaškrtnuty obě hodnoty napájecích napětí, aby bylo jasné, že lze SAMa napájet kteroukoliv hodnotou z 12/24V, resp. z rozsahu 10-30V. Na nových štítcích nejsou zaškrtávací políčka pro hodnoty napájení, ale nad svorkami pro napájení (6+7) je uveden rozsah hodnot možného napájení, tj. popis **power 10-30V**. Modul **SAM-02** obsahuje relé, takže o hodnotě napájecího napětí rozhoduje hodnota napětí pro napájení cívky relé; proto tento modul má inadále volbu 12 / 24V.

Po připojení napájecího napětí trvale svítí na SAmu zelená LEDka POWER. Pouze u nového modulu SAM-01 tato LEDka bliká, podle toho také poznáme, že se jedná o nový typ modulu SAM-01.

3 Panelový číslicový indikátor XDM–14A

3.1 Konfigurace XDM-14A

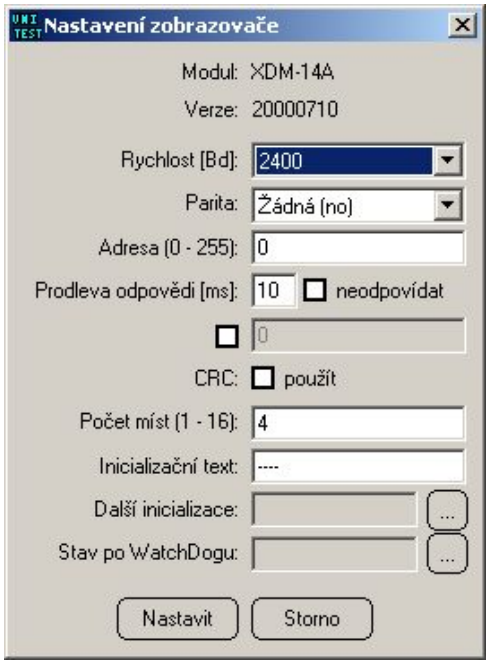
Na štítku XDMka přečteme napájecí napětí 9 - 30 V, na svorky 9+10 pak přivedeme např. 12 V DC. Propojíme svorky 1+3 a 2+4 abychom "udělali" z RS-422 rozhraní RS-485. RS-485 pak připojíme k převodníku RS-485 na RS-232/USB podle toho, co máme k dispozici na PC. Linku RS-232/USB připojíme k PC na COM/USB podle kapitoly 1.

Spustíme program TESTER, zvolíme výrobek XDM, případně upřeníme COM PC - opět viz kapitola 1.

Pokud máme XDM-14A přímo z výroby a vyčteme jeho nastavení volbou INFO (opět viz kapitola 1), dostaneme tuto informaci:



Obr. 9 XDM před konfigurací



Obr. 10 XDM před editací hodnot

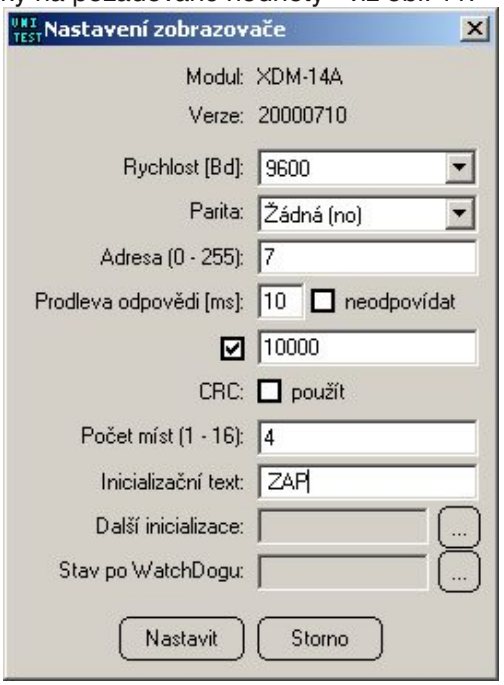
Chceme nastavit adresu 7, rychlost 9600 Bd a WD nastavit na 10 vteřin. Po zapnutí na napájení nechť se zobrazí na displeji nápis ZAP (viz manuál XDMka a inicializační text a jeho definice v TESTERu), pak budeme chtít zobrazovat údaje podle nadefinovaných zpráv projektem ve FREDovi; pokud by komunikace selhala, zobrazí se na displeji čtyři pomlčky (viz popis v manuálu pro XDM).



Obr. 8 Foto zobrazovače XDM-14A

V TESTERu volíme nabídku SET, vidíme následující dialogové okno - viz obr. 10 (pro XDM z výroby - jinak je vyčteno do formuláře aktuální nastavení).

V okně **Nastavení zobrazovače** zeditujeme položky na požadované hodnoty - viz obr. 11.

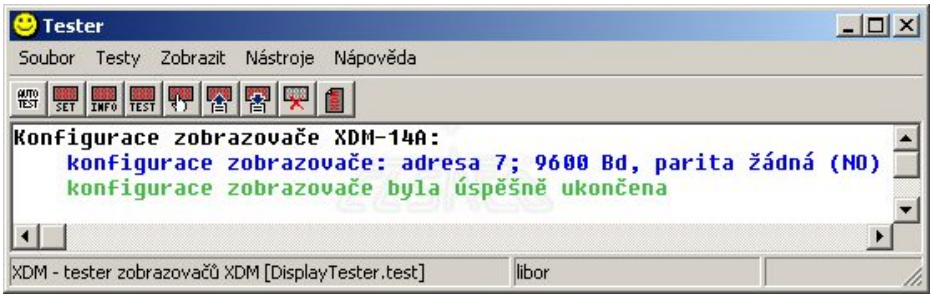


Obr. 11 XDM po editaci hodnot

Jak vidíte na obrázku formuláře pro nastavování zobrazovače XDM, lze nastavovat více položek; pro běžné používání XDM je popsané plně dostačující. Pomocí manuálu pro XDM a TESTERu si můžete vyzkoušet další...

Watchdog zadávejte do řádku pod popisem **Prodleva odpovědi** - zatím tam chybí nápis pro WD. Zadává se v milisekundách.

Po volbě **Nastavit** je do okna TESTERu vepsána zpráva o vykonaném, a to včetně úspěšnosti - viz obrázek 12.



Obr. 12 Průběh konfigurace - zprávy v okně TESTERu

Po zmíněném kliknutí na **Nastavit** se pak chová XDMko následně. Po zapnutí na napájení se objeví nápis ZAP, který je přepsán relací po RS-485 podle toho, co je zadáno ve zprávě. Pokud po 10 sec nepřijde další zpráva, objeví se na displeji XDMka pomlčky. Ty zmizí další správnou relací....

Pro jistotu si můžeme ještě v TESTERu konfiguraci ověřit volbou INFO, uvidíme následující dialogové okno - viz obr. 6 či 13.



Obr. 13 Info o provedeném nastavení XDMka

A potom další zkoušku s odpojenou linkou RS-485: po znovuzapnutí XDMka uvidíme nejprve nápis ZAP a po 10ti vteřinách, kdy vyprší WD, budou zobrazeny pomlčky.

Nyní bychom si měli vyzkoušet použití příkazů pro požadovaný výpis údajů na XDMku.

3.2 Zobrazení údajů na XDM-14A

Budeme potřebovat manuál buď přímo k XDMku nebo manuál **Komunikační protokol SAM** a program TERMINAL.EXE - vše je volně ke stažení na našich stránkách v sekci ***Ke stažení***. Pokud máte problém uvedené nalézt, využijte **Hledání na stránkách**.

Tak např. v manuálu **Komunikační protokol SAM** nalistujeme kapitolu 3.4.2 **Zobrazení údajů na displeji**. Nastudujeme složení zprávy a pokusíme se ji vyslat z PC programem TERMINAL přes COM a převodník do XDMka a tak zkontrolovat, zda zprávu skládáme správně dle manuálu.

Obecně má být zpráva v tomto tvaru:

"aaT<číslo><CR>

Uvozovky na počátku zprávy jsou následovány adresou XDMka (aa = adresa = **07** v našem příkladě), pak následuje písmeno velké **T,** dále zobrazovaný údaj **<číslo>** a znak **<CR>** pro konec zprávy (je to vlastně znak pro konec řádku, enter, dekadická hodnota 13).

<číslo> pro XDM-14A jsou čtyři jakékoliv zobrazitelné znaky na displeji XDM-14A. Vyjímku tvoří znak "tečka", který se do počtu znaků nepočítá. Bližší viz manuál XDMka. **<číslo>** tak může být např. 123,4 nebo i nápis AHOJ či VODA apod.

Pokud budeme zprávu chystat v TERMINALu, s úspěchem použijeme možnost nachystat si ji jako makro. Nadefinujeme si ji tak do příslušného řádku formuláře pro makro, dokonce můžeme pro dané makro zvolit, jak často má být zpráva na linku odesílána (např. každých 500 ms).

Nejprve se však musíme seznámit s TERMINALem a zjistit, jak se znaky zprávy v tomto programu zapisují do řádku formuláře. K tomu nám slouží HELP programu a v něm si je třeba všimnout, jak se zapisují řídící znaky - budeme totiž potřebovat vyslat na závěr zprávy znak **<CR>**, tj. ASCII kód s dekadickou hodnotou **13**, reps. hexahodnotou **0D**.

Quick Help je sice v angličtině, ale s ním si poradíme i my, paneloví angličtináři...

In macros you can use all characters from keyboard and any ASCII char if you use $xx or #xxx. Where $xx is hex and #xxx dec format of ascii code.

If you want to use # or $ char in macro you should type it twice ($$=$ and ##=#).

Takže pokud chceme zapsat hodnotu ASCII znaku ve tvaru hexa, předřadíme ASCII hodnotě znak $, pokud cheme znak zapsat dekadicky, předřadíme mu # (za ním musí být tři cifry!). A dáme si hned příklad pro naše **<CR>**, které zadáme jako:

$0D

nebo jako

#013

Takže řádek pro makro, které vyšle do XDMka zprávu pro zobrazení nápisu AHOJ bude vypadat takto:

"07TAHOJ$0D

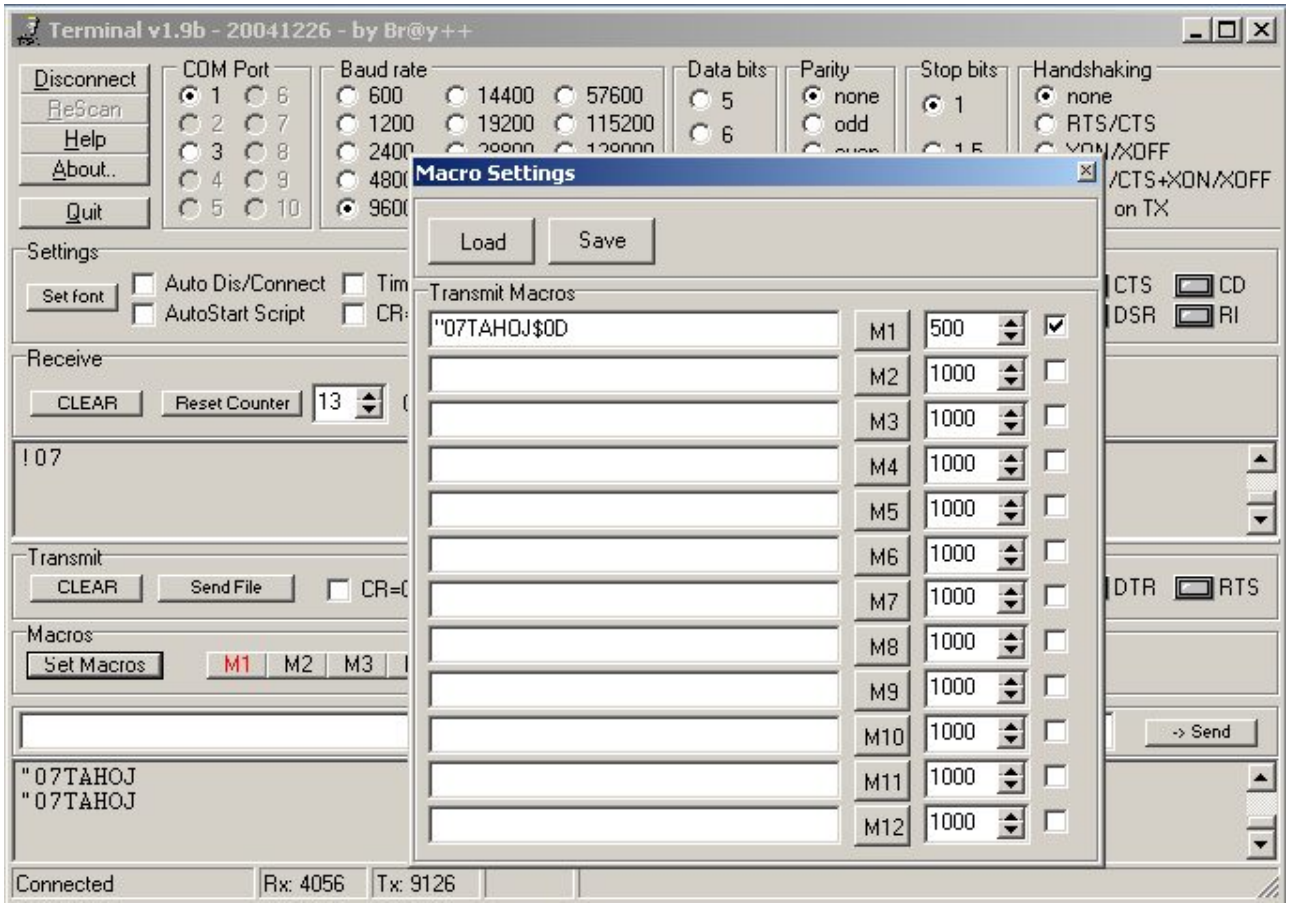
nebo

"07TAHOJ#013

Podobně si vyzkoušíme vyslání zprávy, která zobrazí nějakou čtyřcifernou hodnotu včetně desetinné tečky:

"07T456.7$0D

Tato zpráva zobrazí údaj **456,7**. Lze tedy říci, že už víme jak na zprávu pro zobrazení údajů na XDM-14A. Pro úplnost si ještě všimněme v manuálu, že existuje



Obr. 14 Program TERMINAL a makro pro XDM-14A

odpověď na zprávu tohoto typu a tato odpověď má tvar

!aa<CR>

pokud proběhlo všechno v pořádku, nebo

?aa<CR>

pokud příkaz nebyl proveden.

Sice bychom později v projektu ve FREDovi nemuseli vyhodnocovat, zda vše proběhlo OK (nakonec výsledek je vidět na displeji - nebo při chybě vidět není), ale pro pořádek a pro možnost vyhodnocení chyby i ve FREDovi je dobré vědět, jak má správná odpověď XDMka vypadat.

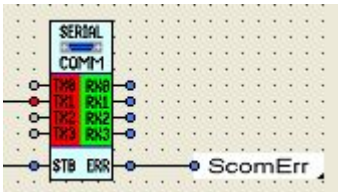


Obr. 16 Důkaz :-)

3.3 Jak na XDM-14A ve FREDovi

V knihovně FREDa pro PL2 HW modul XDM nenaleznete. Je však k dispozici univerzální modul SERIALCOMM, pomocí kterého lze vysílat a přijímat námi nadefinované komunkační relace. Proto jej v projektu (soubor XDM14.opf) ve FREDovi využijeme.

Nejprve si modul SERIALCOMM trochu připomeneme, např. otevřením HELPu a nápovědou přímo k tomuto modulu. Zde se zaměříme na definici jeho parametrů směrem k naší úloze. Cílem bude vypisovat hodnotu, reálné číslo na jedno desetinné místo. Bude se jednat o změřenou hodnotu přivedenou na vstup **Tx1** - viz obr. 15.



Obr. 15

Nastavme nyní komunikační parametry modulu SERIALCOMM. Parametr **rychlost** volíme = 2, proto budou relace uskutečněny ve vteřinové programové smyčce. Jedním průchodem pouze jedna relace, bude-li v projektu jeden modul SERIALCOMM, tak s ním bude komunikováno každou vteřinu, budou-li v projektu dva moduly SERIALCOMM, bude jedním prů-

chodem komunikováno pouze s jedním modulem SERIALCOMM, druhým průchodem pak s druhým modulem SERIALCOMM, ... Pokud bychom chtěli nadefinovat pořadí relací pro více modulů SERIALCOMM, docílíme toho využitím parametru **priorita**.

Komunikační rychlost nastavujeme parametrem **Baudrate**, a to podle tabulky v HELPu. V předchozích kapitolách jsme pro XDMko nadefinovali rychlost 9600 Bd (pro příjem i vysílání - jinak to pro XDM ani nejde). V tabulce pro tuto volbu nalezneme hodnotu parametru Baudrate = 55.

Dále parametr **Parity** ponecháme 0 (nulový, tj. bez parity; *dle konfigurace XDM - viz konfigurace v kap. 3.1 platná i pro další komunikační parametry*).

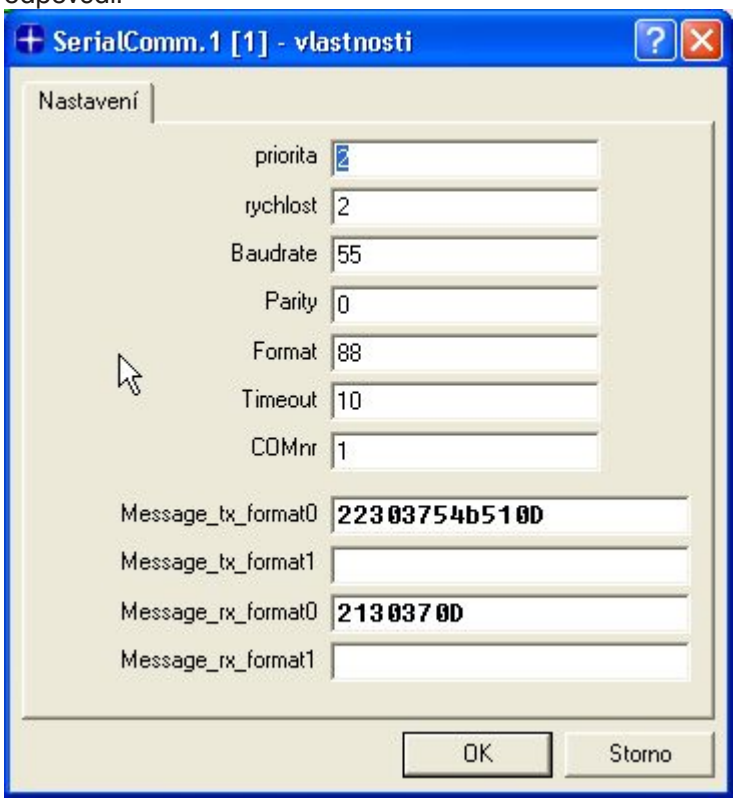
Parametr **Format** - ponecháme hodnotu 88 pro osmibitový formát dat pro příjem i vysílání.

Parametr **Timeout** - maximální doba mezi znaky; ponecháme hodnotu 10 ms.

Parametr **COMnr** nastavíme na 1, protože budeme využívat na centrále COM1 s galvanickým oddělením.

Tím nadefinujeme COMu1 centrály CCPU-21 komunikační parametry, ještě musíme stvořit vlastní komunikační zprávu - relaci.

K tomu slouží formátovací řetězce - viz HELP k modulu, a to zvlášť pro vyslání zprávy a zvlášť pro příjem odpovědi.



Obr. 17 Dialogové okno modulu SERIALCOMM

Formátovací řetězec je řada znaků, která definuje přesně bajt po bajtu jak má zpráva vypadat. Zadává se po otevření dialogového okna SERIALCOMMu do řádků pro parametry **Message_tx_format0**, **Message_tx_format1** pro zprávu, kterou chci vyslat na linku a do řádků pro parametry **Message_rx_format0**, **Message_rx_format1** pro zprávu, kterou přijímám jako odpověď.

Teď nás zajímá definice formátovacího řetězce pro zprávu, kterou chci na linku vyslat. Budeme jej zadávat do parametru **Message_tx_format0**. Pokud by byl řetězec delší než 32 znaků, naplníme tento parametr formátovacími znaky až dokonce a teprve pak pokračujeme v definici na dalším řádku parametrem **Message_tx_format1**.

Bez HELPu se neobejdeme ani po delších zkušenostech, proto si jej necháme otevřen v místech, kde je nadpis **Formátovací řetězce pro vyslání zprávy**.

Zopakujme si, jak má zpráva vypadat. Obecně má být zpráva v tomto tvaru:

"aaT<číslo><CR>

V pokusech jsme vysílali zprávu s hodnotou takto:

"07T456.7$0D

Zpráva začíná **uvozovkami**, pokračuje adresou **07** (viz konfigurace v kap. 3.1), následuje znak velké **T**, pak **hodnota** a na závěr posíláme znak **<CR>** s hexa hodnotou **0D**.

Podle HELPu znak, který chceme vyslat na linku zadáváme přímo jako hexačíslo ve tvaru 00 až FF (písmena velkými znaky).

Uvozovky mají kód 34 dekadicky, tj. 22 hexa. Vyslání prvního znaku (uvozovek) na linku zapíšeme tedy do řádku parametru **Message_tx_format0** jako **22**.

Pak musí následovat zápis adresy, která je dána dvěma čísly 07. Po převodu každé cifry na hexa tvar se jedná o hodnoty **30** a **37**.

Dále následuje znak velké T, který má hexa hodnotu **54**.

A dostáváme se k vyslání hodnoty **<číslo>**, kterou přivádíme na pin **Tx1** modulu **SERIALCOMM**. Na pin přivádíme reálné číslo, které potřebujeme pro XDMko převést na sérii ASCII znaků. Zobrazit (krom desetinné tečky) můžeme na XDM-14A pouze 4 cifry. V HELPu nalezneme nápovědu:

- příkazy apz, bpz, cpz, dpz slouží pro vyslání reálného čísla ze vstupů Tx0..3 jako ASCII znaky na linku
a pro Tx0, b pro Tx1, c pro Tx2, d pro Tx3
p = počet znaků celkem včetně desetinné tečky
z = počet cifer za desetinnou tečkou
maximálně 3 desetinná místa a celkem 8 cifer
je-li cifer pro přenos méně než je udáno ve formátu, jsou zleva doplněny mezerami
při překoročení formátu je na linku vysláno EE.EE (počet E podle formátu)
např. a52 pro vyslání hodnoty reálného čísla ze vstupu Tx0 jako ASCII znaky ve tvaru 12.45

Podle této nápovědy zadejme pro výpis hodnoty formátovací řetězec **b51**;
b - protože se jedná o vstup Tx1
5 - protože chceme zobrazit 4 cifry a desetinnou tečku

1 - protože chceme zobrazit údaj na jedno desetinné číslo.

No a zpráva musí končit znakem <CR>, který má hexa hodnotu **0D**.

Tím jsme vytvořili formátovací řetězec pro vyslání zprávy. Pokud je zpráva správně vyslána a XDMkem přijata, XDMko odpoví zprávou v obecném tvaru:

```
!aa<CR>
```

a v našem případě pak

```
!07<CR>
```

Tento řetězec znaků tedy musíme zadat obdobným způsobem do formátovacího řetězce **Message_rx_format0** pro příjem odpovědi. V HELPu nalezneme opět nápovědu za nadpisem **Formátovací řetězce pro příjem zprávy**.

A tvoříme obdobně. Prvním znakem má být vykřičník, který má hexakód **21**.

Následuje adresa 07, vyjádřena hexa jako dva bajty s kódy **30** a **37**.

Na závěr zprávy opět znak <CR>, tj. hexa **0D**.

Zadání do formátovacích řádků vidíte na obr. 17.

Projekt **XDM-14A.opf**, který tuto definici (krom jiného) obsahuje, je ke stažení na našich stránkách spolu s touto Kuchařkou.

Zobrazovač XDM-14A připojíme linkou RS-485 ke COMu1 centrály CCPU-21, projekt přeložíme a vyzkoušíme.

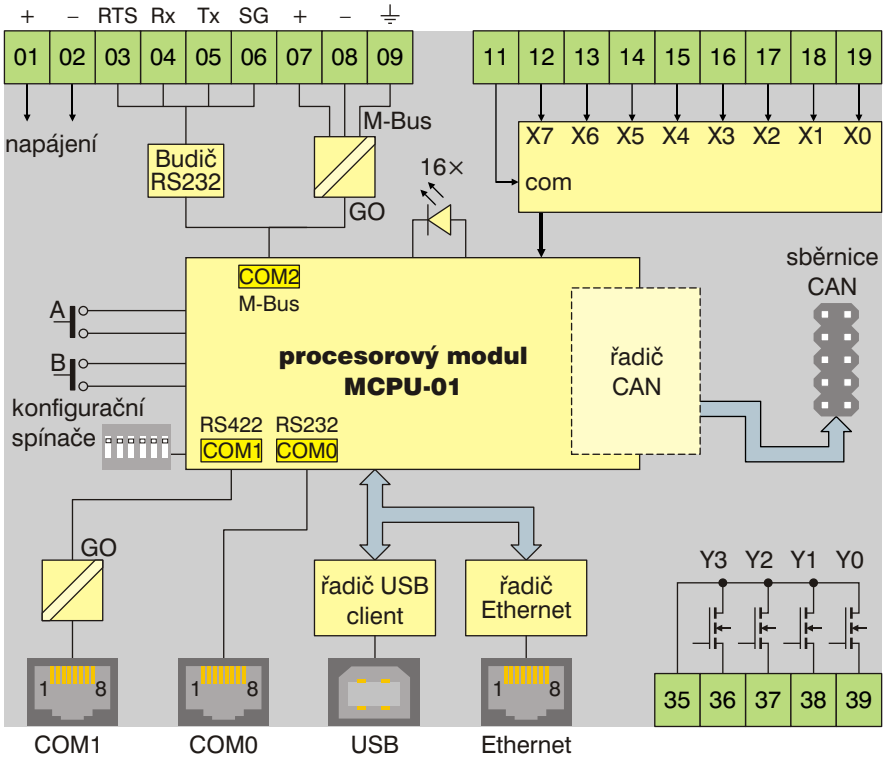
V projektu vidíte ještě druhý modul SERIALCOMM, který slouží k vyslání textu VODA na stejný zobrazovač. Všimněte si, že je určena priorita obou modulů a tak je prvním průchodem vyslán na XDMko text VODA a druhým průchodem programovou smyčkou pak naměřená hodnota. Každý modul SERIALCOMM lze pak blokovat signály, přiváděnými na pin STB modulu SERIALCOMM. Tak můžete posílat jednu, duhou nebo obě zprávy, případně obě zablokovat (a po 10 vteřinách se objeví na zobrazovači pomlčky).

3.4 COM1 u CCPU-03 a CCPU-21

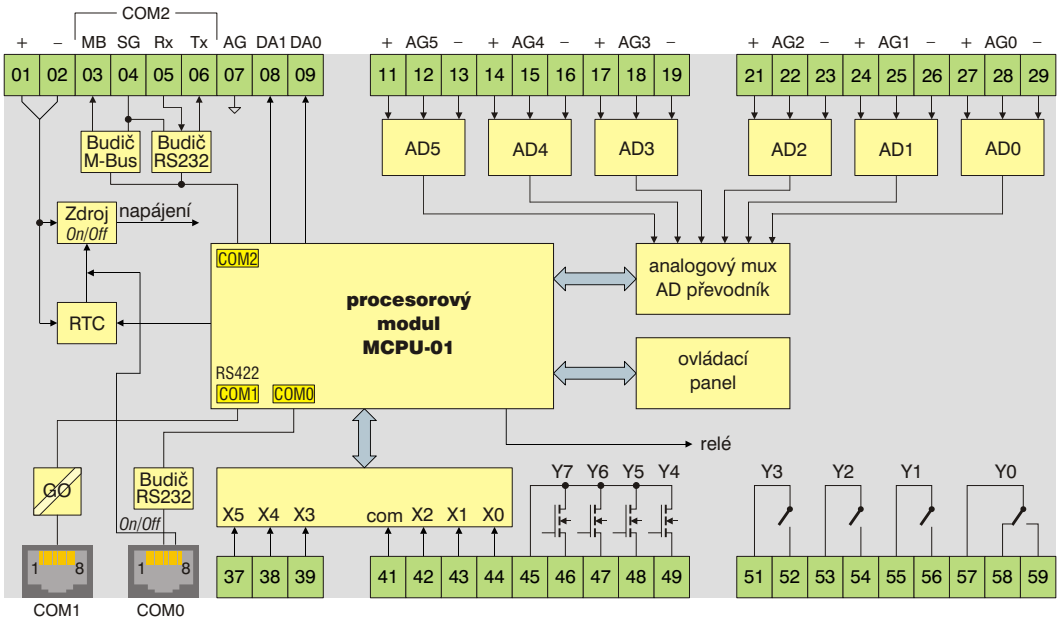
Pro pořádek ještě zapojení COMu1, které je u centrál CCPU-03 a CCPU-21 totožné. COM1 je vyveden na osmipinový konektor RJ45. Viz bloková schémata na obr. 18 a 20 a tabulka zapojení konektoru na obr. 20. Z linky RS-422 uděláme linku RS-485 propojením signálu TxD+ se signálem RxD+ a signálu TxD- se signálem RxD-.

COM1 RS-422	
1	+360R
2	TxD+
3	TxD-
4	GND
5	GND
6	RxD-
7	RxD+
8	-360R

Obr. 19 Zapojení konektoru pro COM1



Obr. 18 Blokové schéma CCPU-03



Obr. 20 Blokové schéma centrály CCPU-21

4 Modul SAM–01 pro měření čtyř analogových vstupů

4.1 Konfigurace modulu SAM-01

4.1.1 INFO v TESTERu

Ke konfiguraci opět použijeme výhodný firemní program **TESTER**. Na štítku modulu **SAM-01** si přečteme pro jaké napájení byl vyroben, na jeho svorkách 6+7 pro napájení je naznačena jeho polarita.

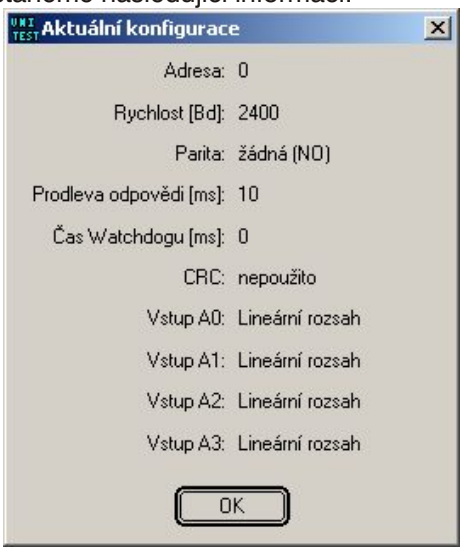
Na svorky pro linku RS-422 zapojíme propojky 2+4 a 3+5 pro převod na linku RS-485. Potom vodič +RxDTxD vedeme na označení stejného pinu převodníku RS-485 na USB/COM, obdobně pak -RxDTxD... Konektor USB/COM převodníku propojíme kabelem s příslušným USB/COMem počítače. Toto propojení je tedy obdobné jako v předchozí kapitole se zobrazovačem XDM-14A.

Zařízení a převodník zapneme, nastartujeme program **TESTER**, v úvodním okně zvolte testy pro moduly SAM, resp. zkontrolujte nastavení COMu PC jako v kap. 1. Pak zvolíme volbu **INFO**. Jsme vyzváni ke znovuzapnutí SAMu:



Obr. 21 Výzva před testem ap.

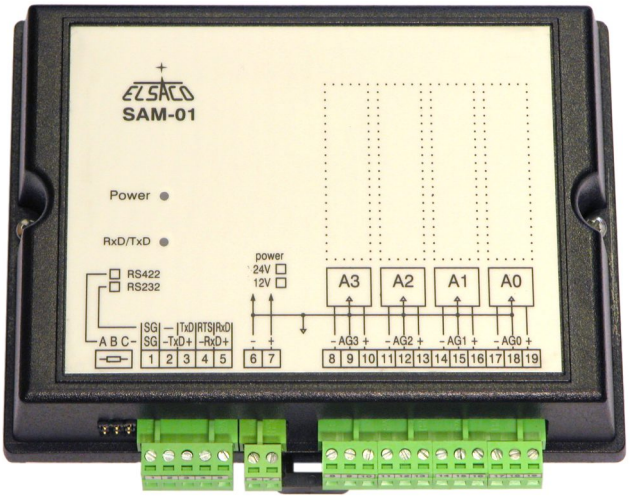
Provedeme znovuzapnutí SAMu a pokud máme SAM přímo z výroby (nebo po výmazu konfigurace) dostaneme následující informaci:



Obr. 22 SAM-01 nastavení z výroby

4.1.2 Vyčtení verze FW TERMINALem

Vývoj SAMů (tak jako i jiných výrobků) si vyžádal změnu firmware, případně i náhradu některých součástek, které se třeba přestali vyrábět apod. Tak pro "staré" SAMy nemusí být TESTER plně funkční, na což



Obr. 23 Foto modulu SAM-01

by nás měl sám TESTER upozornit. My si to musíme uvědomit, být opatrnější, případně si vyčíst verzi FW a porovnat ji s datem, od kterého se počítá SAM jako "nový". Jak uvidíte dál, bylo důležité verzi FW ze SAMa vyčíst. To nyní můžeme, protože díky volbě INFO v programu TESTER (viz předešlá kapitola) známe nastavení komunikačních parametrů SAMa.

Ponecháme SAM připojený přes převodník k PC jako před chvílí a spustíme program TERMINAL. V něm nastavíme komunikační parametry pro COM PC podle toho, co jsme vyčetli ze SAMa volbou INFO v kap. 4.1.1. V našem příkladě se jednalo o nastavení z výroby či po výmazu konfigurace, tj. adresa 0, rychlost 2400 Bd, ...

V manuálu **Konfigurační protokol SAM** nalezneme v kap. **2.1.5 Verze firmware** příkaz pro vyčtení. Ten si nachystáme jako makro do TERMINALu. Obecný tvar příkazu:

$aaF<CR>

kde aa je adresa modulu, u nás 00

Podle dříve zmíněných poravidel QuickHelpu TERMINALu bude zápis makra vypadat následovně:

$$00F$0D

Po jednorázovém spuštění makra je tato relace vyslána na linku a SAM odpoví:

!0020000710

Odpověď očekáváme ve tvaru:

!aarrrrmmdd<CR>

kde rrrrmmdd udává rok/měsíc/den revize FW

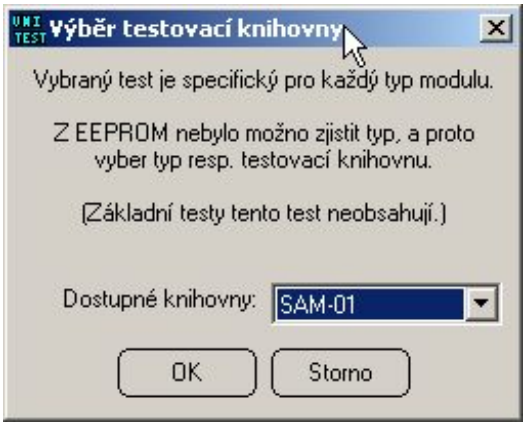
Takže náš SAM s adresou 0 odpověděl, že obsahuje FW ze dne 10.07.2000 - jedná se tedy o "starého" SAMa.

4.1.3 Nastavení starého SAMa TESTERem

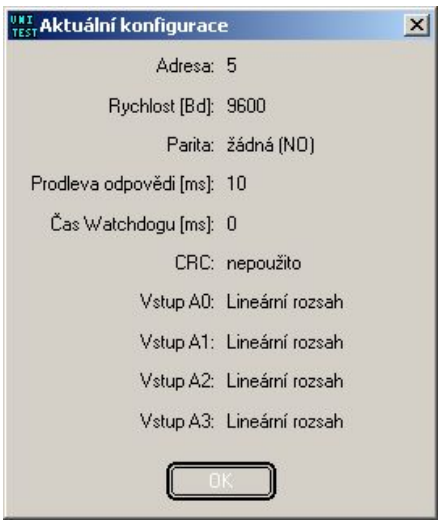
Pro náš příklad chtějme nastavit komunikační adresu 5, rychlost 9600 Bd, bez parity, prodlevu odpovědi ponecháme 10 ms, bez WD a bez CRCu. Lineární roz-



Obr. 26 Editace komunikačních parametrů



Obr. 27 Volba knihovny



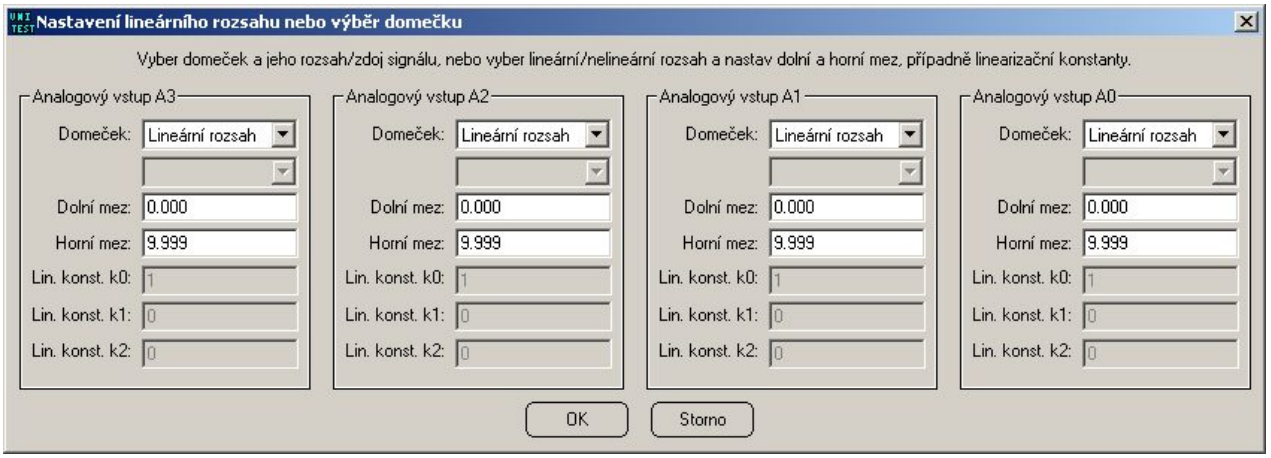
Obr. 28 INFO po konfiguraci

sahy měřených vstupů chtějme změnit podle odporových sítí uvnitř SAMa na rozsahy od 0,000 do 9,999 V. Měříme 0 až 10V - volba rozsahu - viz manuál SAMa.

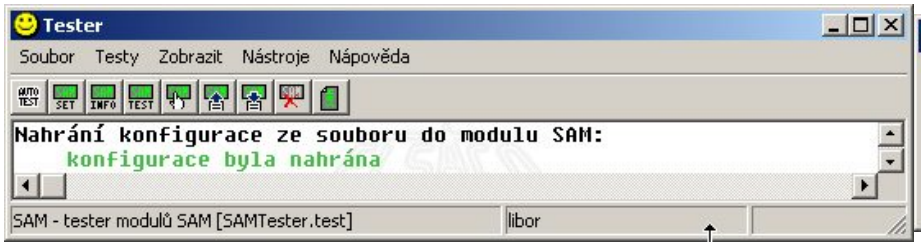
Pokud narazíte na "starý" modul SAM-01, dejte si pozor při volbě SET v TESTERu, stalo se mi, že došlo k chybnému přepsání rozsahů. Proto v takovém případě provádějte konfiguraci "starého" modulu SAM-01 pomocí volby **Testy** / **Správa konfiguračních souborů F10**, resp. rychlou volbou klinutím na ikonu úplně vpravo (*Offline tvorba a úprava konfiguračních souborů*).

Následně jste vyzváni k volbě knihovny - viz obr. 27.

Vyberte **SAM-01** a schvalte **OK**. V dalším okně proveďte požadovanou konfiguraci pro komunikační parametry - viz. obr. 26.



Obr. 24 Volba rozsahů pro konfiguraci modulu SAM-01



Obr. 29 Info po konfiguraci ze souboru



Obr. 25 Info o uložení

Vyplňte požadované a pak klikněte na tlačítko pro **Nastavení rozsahu** / **domečku** - to v řádku úplně vpravo, označené třemi tečkami.

Je nabídnuto další dialogové okno, ve kterém vyplníte rozsahy domečků podle manuálu pro SAM-01, viz obrázek 24.

Po správném vyplnění hodnot pro rozsahy jednotlivých vstupů schvalte tlačítkem **OK**. Dále pak v původním okně **Offline konfigurace modulu SAM-01** klikněte na tlačítko **Uložit do souboru**, v dalším okně pak zadejte název souboru, do kterého chcete nadefinovanou konfiguraci uložit a tu uložte. Info o uložení - viz obrázek 25.

Pak přejděte na volbu **TESTY** / **Nahrání konfigurace F7**, splňte požadované znovuzapnutí SAMa, vyberte právě uložený soubor a výběr schvalte. Konfigurace ze souboru je vyslána do SAMa a v okně TESTERu jste informováni o jejím provedení - viz obr. 29.

Po ukončené konfiguraci modulu SAM-01 se můžeme o výsledku přesvědčit pomocí volby **INFO** - postup jako na začátku této kapitoly, výsledek je vidět na obrázku 28.

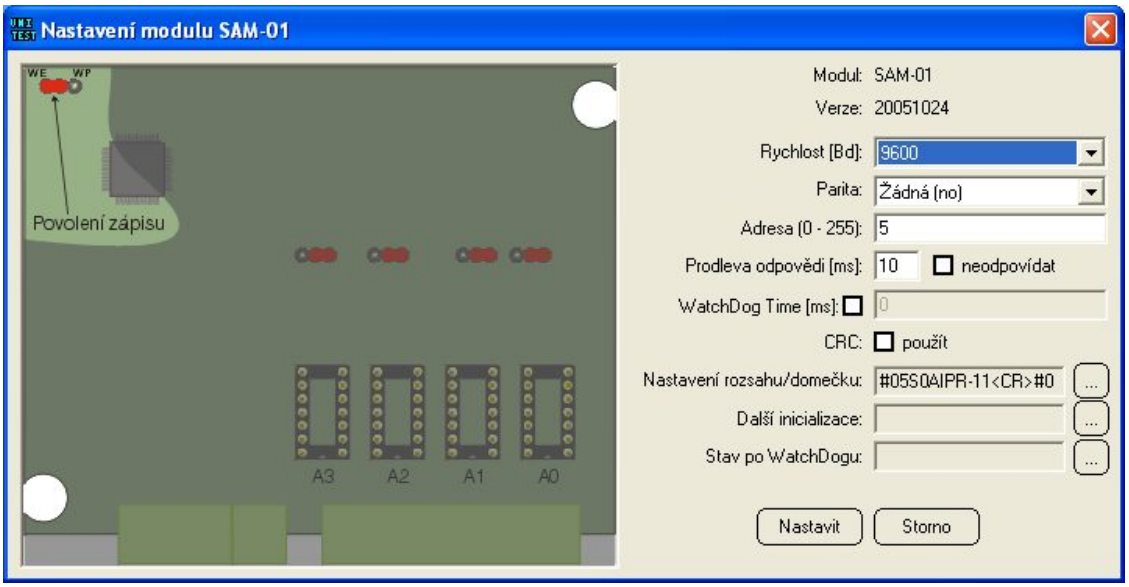
Nyní máme modul SAM-01 připraven pro náš další příklad aplikace, a to s adresou 5, komunikační rychlostí 9600 Bd a s rozsahy 0 až 10 V.

Základní komunikaci se SAMem si vyzkoušíme v kapitole **Vyčtení naměřených hodnot** opět s použitím programu TERMINAL.

4.1.4 Nastavení nového SAMa TESTERem

V tomto případě můžeme využít volby **SET** v programu TESTER přímo. Nejprve jste jako obvykle vyzváni k znovuzapnutí SAMa, po vykonání tohoto příkazu je otevřeno dialogové okno pro nastavení SAM-01, viz obr. 30. Podle požadovaného změníme ve formuláři jednotlivé parametry, V našem případě nastavujeme komunikační rychlost **9600 Bd**, **bez parity**, adresu **5**, prodlevu odpovědi ponecháme **10 ms**, WD nezařazujeme... Povšimněte si, že úplně vpravo nahoře vidíte označení testovaného modulu **SAM-01** a verzi FW **20051024** - jedná se o nového SAMa s FW ze dne 24.10.2005.

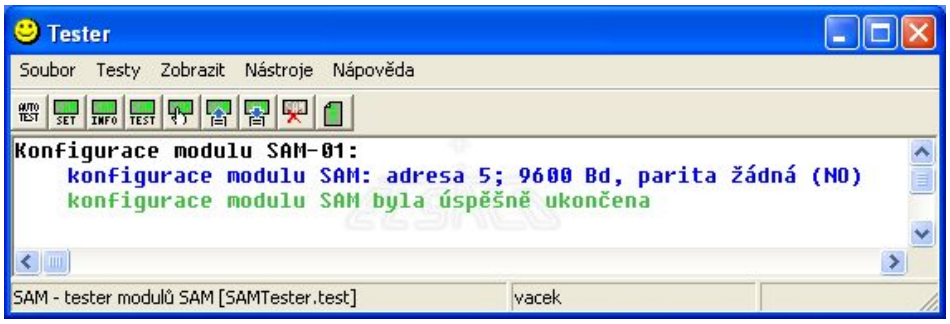
Nakonec jsem zjistil, že pokud po připojení SAM-01 na napájení zelená LEDka POWER bliká, jedná se o nový typ



Obr. 30 Nastavení parametrů pro modul SAM-01 v programu TESTER



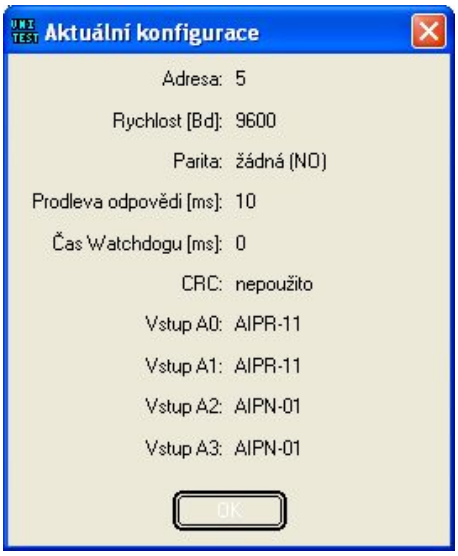
Obr. 31 Nastavení rozsahů/domečků pro SAM-01



Obr. 33 Info o provedené konfiguraci

modulu SAM-01; pokud svítí trvale, jedná se o starší typ modulu SAM-01.

Náš SAM obsahuje domečky (výměnné odporové sítě pro nastavení analogových rozsahů). Správné osazení domečky (fyzické proti této konfiguraci) můžeme zkontrolovat, případně opravit, pomocí dalšího dialogového okna, které vyvoláme kliknutím v tomto okně na tlačítko "..." vpravo vedle popisu **Nastavení rozsahu/domečku**. Máme-li nastavení rozsahu (obr. 31) správně nastaveno, schválíme tlačítkem **OK**. Ještě jednou zkontrolujeme formulář celkového nastavení (obr. 30) a to pak schválíme tlačítkem **NASTAVIT.** Předvolené konfigurace se nahraje do SAMa, o čemž jsme informování v okně TESTERU - viz obr. 33. Pro jistotu můžeme ještě použít v TESTERu volbu INFO pro kontrolu nastaveného SAMa - viz obr. 34.



Obr. 34 INFO nastaveného modulu SAM-01

4.2 Vyčtení naměřených hodnot

V manuálu **Komunikační protokol SAM** v kap. 3.3.2 **Čtení analogové hodnoty** zjistíme jak má vypadat komunikační požadavek:

#aan<CR>

kde

aa = adresa modulu

n = číslo vstupu (0 až 3 pro SAM-01)

A protože máme SAMa nakonfigurovaného s adresou 05 a chceme číst pro jednoduchost pouze naměřenou hodnotu z prvního vstupu, označeného A0, bude náš požadavek vypadat následovně:

#050<CR>

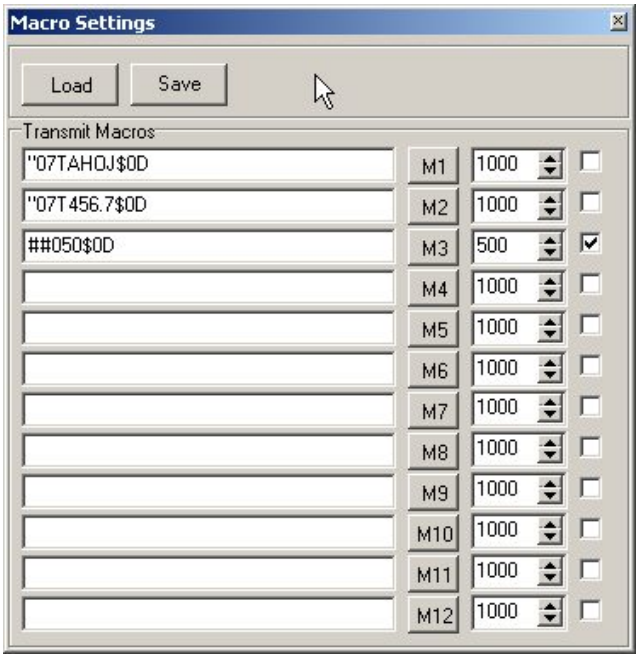
První pokusy pro ověření správnosti našeho zadávání požadavku provedeme opět pomocí programu TERMINAL. Nachystejme si požadavek jako makro **M3** podle pravidel Quick Helpu TERMINALu. Pokud si dobře vzpomínáte, v předchozí kapitole je kopie *Quick Helpu* v angličtině, ale v českém překladu jsme nezdůraznili, že pokud potřebujeme na linku TERMINALem vyslat znaky # nebo $ (které jsou určeny pro předeznačení typu čísla, tj. zda je za tímto znakem ASCII hodnota dekadicky nebo hexadecimálně), musíme je do řádku makra napsat 2x. To znamená, že když nyní potřebujeme vyslat znak #, musíme zapsat tento znak dvakrát po sobě: ##.

Pak do řádku pro nastavení třetího makra **M3** zapíšeme:

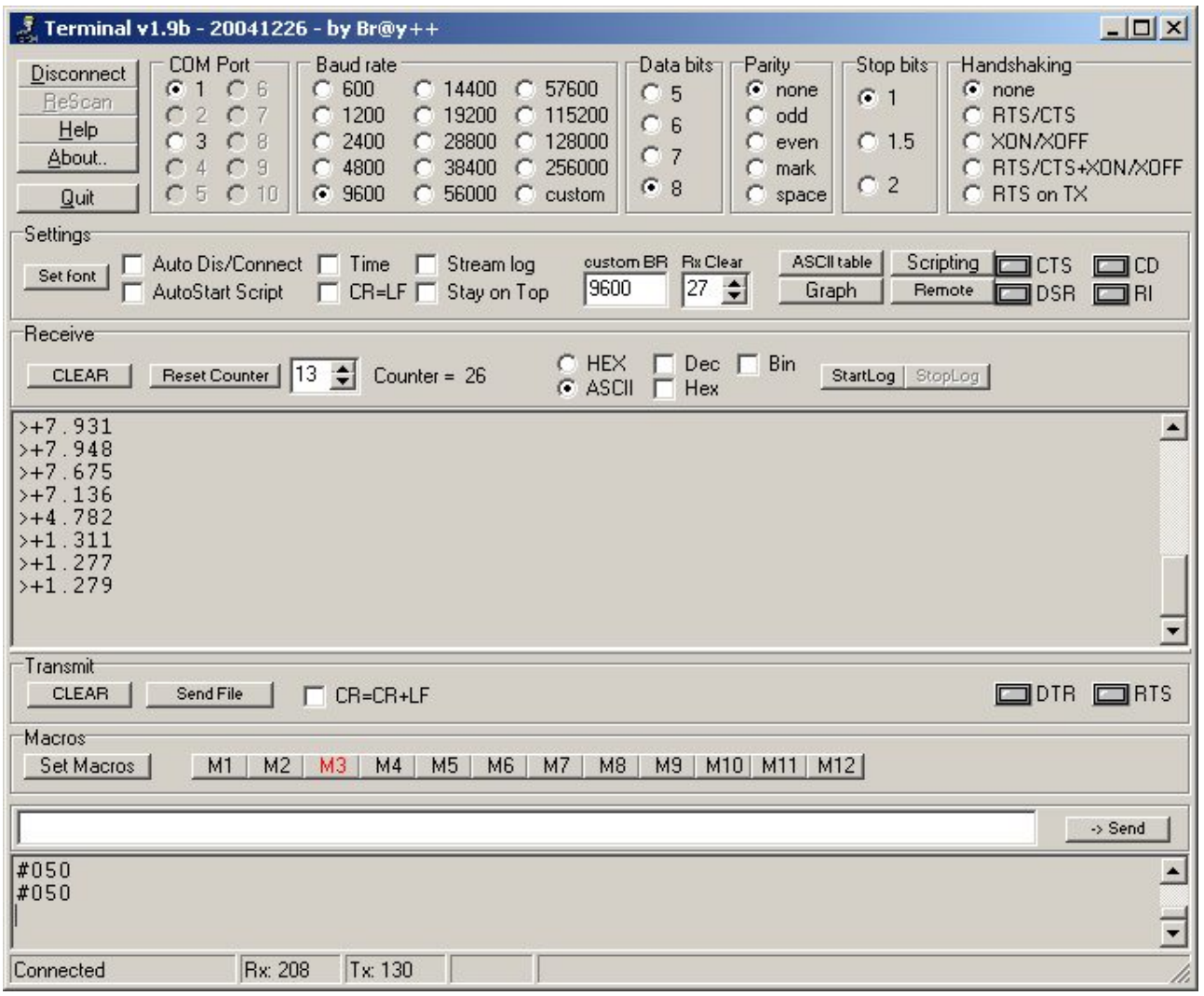
##050$0D

SAM máme připojen k PC přes převodník (tak jako v předchozí kapitole), nastartujeme program TERMINAL, nachystáme popsané makro **M3**, zadáme dobu 500 ms pro opakování zprávy a zatržením vedle této volby definujeme opakované vysílání požadavku na vyčtení vstupu A0 z modulu SAM-01 - viz obrázek 32.

V hlavním okně TERMINALu - viz obr. 36 - pak zkontrolujeme nastavení COMu PC, na který máme připojen převodník (a na něj pak SAMa). V našem případě se jedná o COM1 PC, komunikační rychlost 9600 Bd, bez parity, 8 datových bitů, 1 stop_bit... Kliknutím na **Connect** (vlevo nahoře okna TERMINALu) se popis



Obr. 32 Makra v TERMINALu



Obr. 36 Vyčítání naměřené hodnoty modulem SAM-01

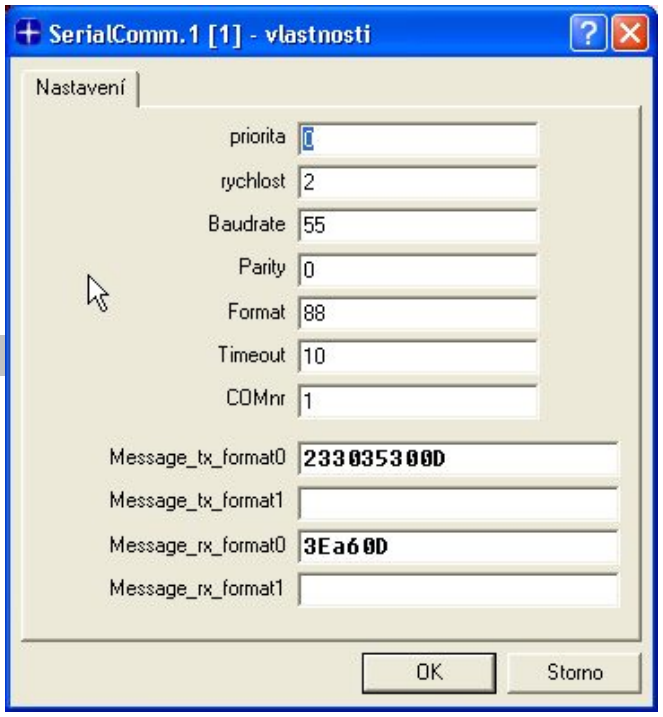
tohoto tlačítka změní na **Disconnect** a začnou probíhat relace, které můžeme číst v okně zpráv, a to zvlášť v dolní části okna vysílané požadavky **#050** a ve výše umístěné části okna pro příjem pak přijaté zprávy **>+7.931** ..... **>+1.279**, ve kterých se naměřená hodnota mění podle skutečnosti.

Můžeme říct, že už víme jak na SAM-01 a přikročit tak k dalšímu kroku - poradit si se SAMem i ve FREDovi.

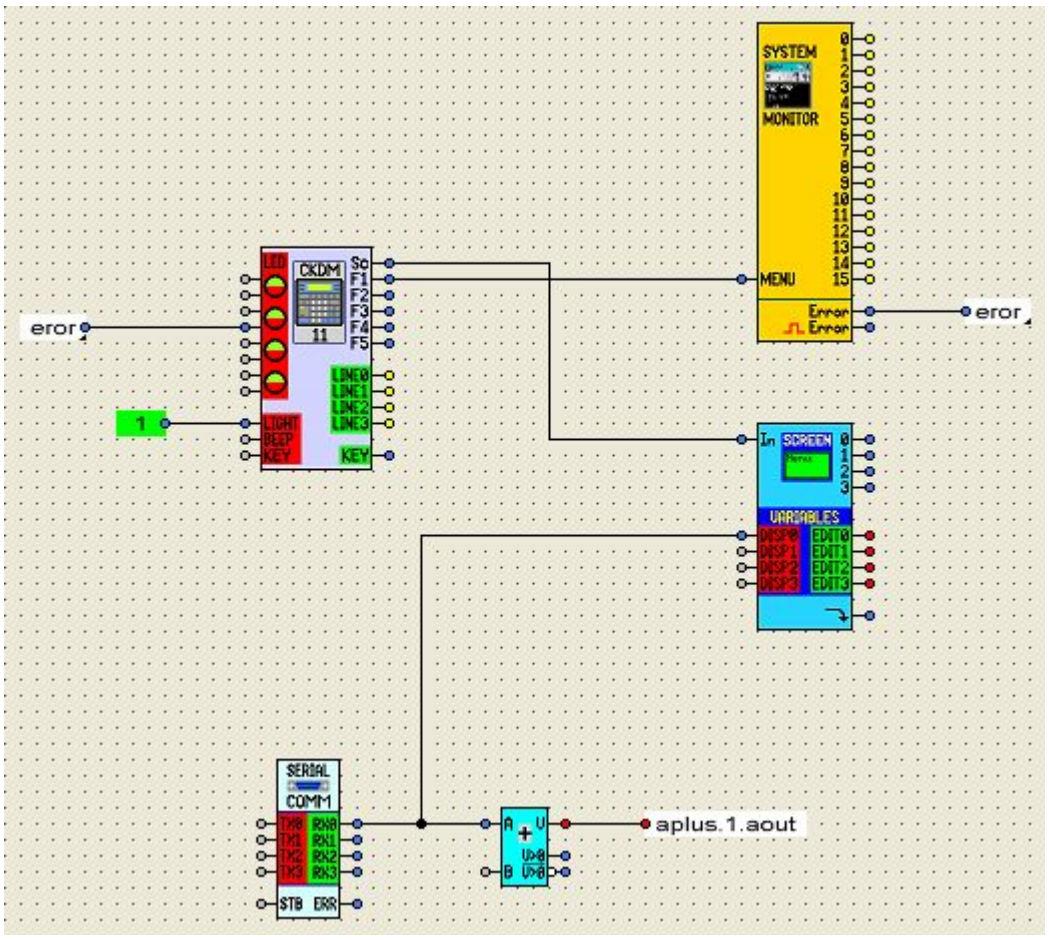
4.3 SAM-01 ve FREDovi

Také modul SAM-01 (tak jako ostatní SAMy) v knihovně PL2 FREDa nenalezneme. To však nevadí, protože už tušíme, že budeme k němu přistupovat obdobně jako k zobrazovači XDM-14A v minulé kapitole. Použijeme pro definici komunikačních relací modul SERIALCOMM a relace pro modul SAM-01 jsme si vysvětlili před chvílí.

Nastavme nejprve komunikační parametry modulu SERIALCOMM. Parametr **rychlost** volíme = 2, tak budou relace uskutečněny ve vteřinové programové smyčce. Jedním průchodem pouze jedna relace, bude-li v projektu jeden modul SERIALCOMM, tak s



Obr. 35 Relace pro SAM-01, modul SERIALCOMM



Obr. 37 Zkušební projekt pro SAM-01

ním bude komunikováno každou vteřinu, budou-li v projektu dva moduly SERIALCOMM, bude jedním průchodem komunikováno pouze s jedním modulem SERIALCOMM, druhým průchodem pak s druhým modulem SERIALCOMM, ... Pokud bychom chtěli nadefinovat pořadí relací pro více modulů SERIALCOMM, docílíme toho využitím parametru **priorita**.

Komunikační rychlost nastavujeme parametrem **Baudrate**, a to podle tabulky v HELPu. Pro SAMa jsme nadefinovali rychlost 9600 Bd (pro příjem i vysílání - pro SAMy opět volíme vždy stejnou rychlost příjmu i vysílání). V tabulce nalezneme pro tuto volbu hodnotu 55.

Dále parametr **Parity** ponecháme 0 (nulový - bez parity; dle konfigurace SAMa - *viz konfigurace v kap. 4.1, která samozřejmě platí i pro další komunikační parametry*).

Parametr **Format** - ponecháme hodnotu 88 pro osmibitový formát dat pro příjem i vysílání.

Parametr **Timeout** - maximální doba mezi znaky - ponecháme hodnotu 10 ms.

Parametr **COMnr** nastavíme na 1, protože budeme využívat na centrále COM1 s galvanickým oddělením.

Tím nadefinujeme COMu1 centrály CCPU-21 komunikační parametry, ještě musíme stvořit vlastní komunikační zprávy - relace pro požadavek a příjem.

K tomu slouží formátovací řetězce - opět viz HELP PL2 ve FREDovi, a to zvlášť pro vyslání zprávy a zvlášť pro příjem odpovědi.

Požadavek na vyčtení naměřené analogové hodnoty modulem SAM-01 a prvním vstupu A0 (jak jsme si řekli již dříve) vypadá následovně:

#aan<CR>

kde

aa = adresa modulu

n = číslo vstupu (od 0)

A protože máme SAMa nakonfigurovaného s adresou 05 a chceme číst pro jednoduchost pouze naměřenou hodnotu z prvního vstupu, označeného A0, bude náš požadavek vypadat následovně:

#050<CR>

Podle HELPu víme, že pokud chceme poslat přímo znak na linku, definujeme ho hexa_zápisem jeho ASCII hodnoty. Pokud nemáte příslušnou tabulku k dispozici, lze si vypomoci i naším firemním katalogem, ve kterém naleznete obdobnou tabulku u terminálu CDKM / SKDM, např. v Katalogu 5 tabulka Znaková sada. Pro znak **#** naleznete hexa_hodnotu **23**.

Po něm má následovat dvojciferně zadaná **adresa** 05, vyjádřeno hexa se jedná i bajty **30** a **35**.

Dále následuje číslo vstupu 0, hexa pak **30**.

A na konec relace dáváme znak <CR>, zapsáno ve tvaru hexa pak **0D**. Viz obrázek 35.

Tím jsme nadefinovali celý požadavek na vyčtení hodnoty prvního vstupu modulu SAM-01. Očekáváme však důležitou odpověď, která bude obsahovat naměřený údaj jako ASCII číslo (viz manuál pro SAM-01 nebo manuál *Komunikační protokol SAM*, *kap. 3.2.2 na str. 6*). Proto hledáme v HELPu informace o formátovacích řetězcích pro příjem zprávy. Tam se dočteme, že pro příjem **n** znaků ve formátu ASCII číslo slouží příkaz **an** až **dn**:

*příkaz **an** až **dn** (malá písmena **a, b, c, d** s cifrou) = přečte **n** znaků jako ASCII číslo a uloží jako reálné číslo na výstup Rx0..3*

*(př. **b6** přečte následujících 6 znaků jako ascii řetězec [např. +23.56], konvertuje na reálné číslo a uloží na výstup Rx1)*

Zkusme na základě všech výše uvedených poznatků poskládat formátovací řetězec pro příjem očekávané zprávy.

Přijímaná zpráva má podle manuálu obecný tvar

>+123.4<CR>

My jsme si nadefinovali rozsahy měřených analogových hodnot od 0,000 do 9,999 V - máme odporové sítě 0 - 10 V. Tak jak máme nadefinován příslušný vstupní rozsah, tak ve stejném tvaru bude vypadat odpověď ve zprávě. Proto její obecný tvar upravíme podle toho, jak jsme nakonfigurovali dříve příslušný rozsah vstupu A0:

>+1.234<CR>

V tabulce ASCII hodnot nalezneme hexa hodnotu pro většítko, je **3E**.

Pak následuje včetně plusu a desetinné tečky 6 znaků naměřené hodnoty. Podle HELPu tedy do formátovacího řetězce dále zapíšeme příkaz **a6** a na závěr zprávy doplníme hexakód pro znak <CR>. Výsledný formátovací řetězec pro příjem naší zprávy pak dopadne takto:

3Ea60D

Zápis zkontrolujeme na obrázku 35. Tak bude vyplněn formátovací řetězec v projektu **SAM01.opf**. Vytvoříme velmi jednoduchý projekt, pouze na zkoušku komunikace s modulem SAM-01 (tento projekt je ke stažení na našich stránikách spolu s Kuchařkou).

Projekt je vidět na obr. 37. V něm je nejdůležitější dříve popsaná konfigurace modulu SERIALCOMM, jejíž část je i na obr. 35. Za zmínku ještě stojí součtové hradlo APLUS, které je užito pouze pro správné zobrazování hodnoty reálného čísla (přijaté změřené analogové hodnoty) v ladícím (RUN) režimu ve výstupním labelu. Vstupy a výstupy modulu SERIALCOMM jsou "obecné" a jejich typ je definován až formátovacím řetězcem. Protože všechny typy hodnot (float, integer, double integer) jsou vnitřně vyjadřovány vždy 4bajtově a u modulu SERIALCOMM není definováno, že výstup je např. vždy typu float, musíme na výstupní pin SERIALCOMMu zapojit nějaký modul, kterým 4bajtová hodnota projde a na jeho výstupu bude již definována pevně jako float. Vybrali jsme si pro tento případ modul hradla **APLUS**. Na jeho výstupu pak můžeme v připojeném labelu v RUN režimu sledovat ono vykomunikované (přijaté) reálné číslo.

Pokud by nastaly problémy s komunikací je vhodné k lince RS-485 ponechat připojen převodník RS-485/RS232 nebo USB k PC, v PC spustit program TERMINAL a zprávy "odposlouchávat". Tak pak zjistíme, zda na lince zprávy probíhají, zda mají správný tvar podle manuálu či popisu zde v Kuchařce, atd.

4.4 SAM-01 a aktivní posílání hodnot

Od verze firmware 20041005 produktu SAM-01 má SAM tento příkaz:

#aaTddf

pro aktivní odeslání (tj. bez předchozího požadavku) změřené hodnoty na sériovou linku, čehož můžeme velice výhodně využít např. pro přímé odeslání naměřené hodnoty na zobrazovače typu XDM. V tomto případě se podrobně zmíníme o propojení modulu SAM-01 se zobrazovačem XDM-14A.

Ve zmíněném příkazu znamená

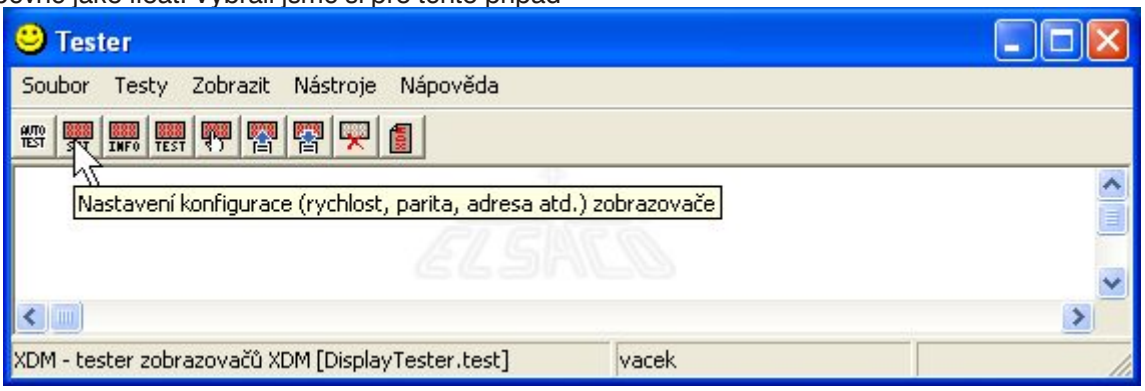
aa = adresa modulu
dd = adresa cílového displeje
f = podle bitů

- 0 a 1 – číslo kanálu
- 3 – se znaménkem
- 2 – 4 nebo 3 cifry

Konfiguraci obou výrobků provedeme pomocí programu TESTER, takže zmíněný příkaz uvádíme pouze pro informaci.

Náš úkol, který nyní budeme řešit, je propojit modul SAM-01 se zobrazovačem XDM-14A po sériové lince RS-485, komunikovat rychlostí 2400 Bd, bez parity. Ze SAMa budeme přenášet změřenou hodnotu z nultého kanálu (tj. jeho prvního vstupu).

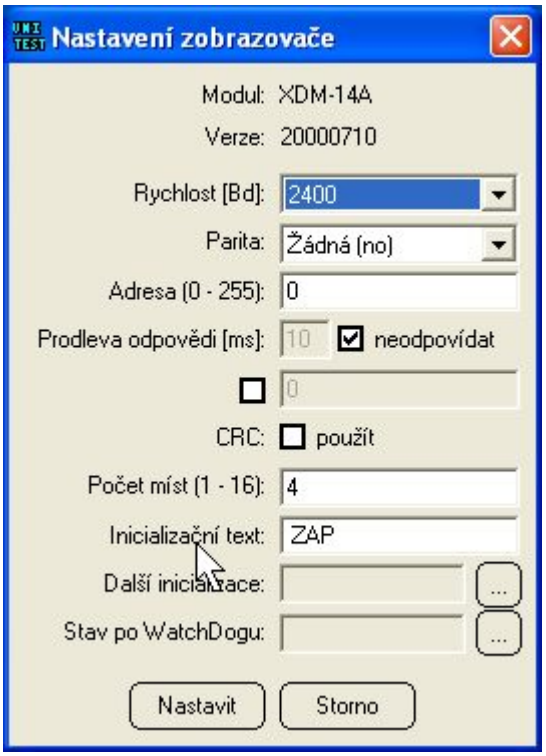
Výše zmíněný příkaz však vyšle pouze jednou onu změřenou hodnotu ze SAMa na linku, a to po restartu nebo po zapnutí jednotky. Pro soustavné zobrazování právě změřené hodnoty musíme použít příkazů víc. Jak to udělat? Stačí po definované prodlevě poslat týž příkaz znovu. Takže krom běžné konfigurace (jako je nastavení adresy, komunkační rychlosti, ...) použijeme zmíněný příkaz **T** pro odeslání změřené hodnoty ze SAMa, příkaz **W** (wait) pro definici prodlevy a nako-



Obr. 38 Start Nastavení konfigurace zobrazovače XDM



Obr. 39 Výzva vyp/zap



Obr. 40 Dialog pro nastavení XDM

Po zapojení a jeho kontrole nastartujeme program TESTER, v jeho úvodním okně zvolíme

XDM - tester zobrazovačů XDM (DisplayTester.test)

případně před touto volbou editujeme nastavení portu PC (COMu) podle potřeby. Jednalo by se o volbu tlačítkem

Upravit nastavení vybraného testeru

Pak volíme tlačítkem **OK** vstup do testeru zobrazovačů XDM - viz obr. 38 a kliknutím na ikonku **SET** volíme nastavení konfigurace.

Následuje výzva k vypnutí a po chvíli k znovuzapnutí zobrazovače (obr. 39), což je nutné pro přechod do konfiguračního režimu zobrazovače.

Provedeme požadované s prodlevou cca 7-10 sec.

Pak bude zobrazeno dialogové okno podle obrázku 40, ve kterém nastavíme požadované parametry.

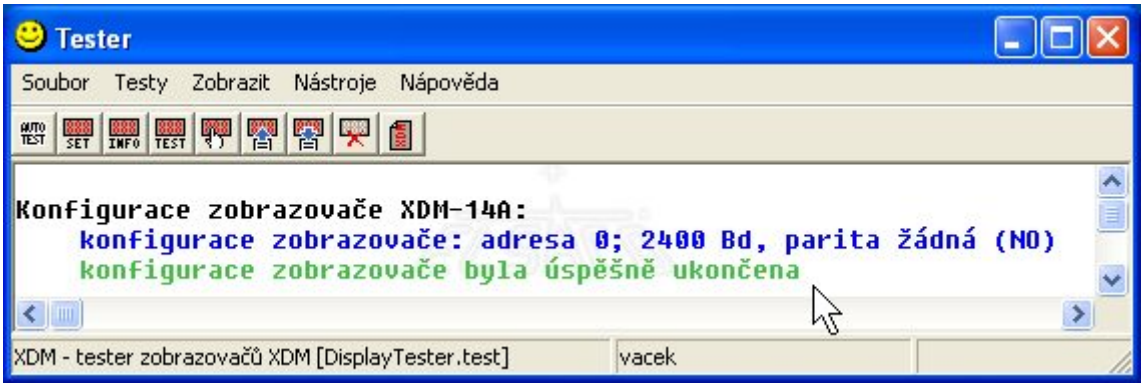
Komunikační rychlost 2400 Bd, bez parity, požadovanou adresu pro XDMko na sběrnici RS-485 - zde 0.

Zaškrtneme volbu "*neodpovídat*", protože bohatě bude stačit, že SAM-01 bude aktivně posílat zprávu se změřenou hodnotou. Potvrzení o jejím správném příjmu nemá smysl, relace se bude opakovat každou vteřinu.

Další řádek v okénku, kterému chybí popis WatchDog, můžeme, ale nemusíme vyplňovat.

Zde řádek vyplněn (a zaškrtnuto políčko) není, na obr. 40 vidíte šedou nulu. Znamená to, že v případě, že vypadnou relace mezi XDM a SAMem, zůstane na displeji zobrazovače zobrazena poslední komunikovaná hodnota.

Pokud políčko zaškrtneme a zadáme WD = 5000 ms, pak v případě ztráty komunikace budou za 5 sec



Obr. 41 Zpráva o ukončení konfogurace XDMka

nec příkaz pro restart jednotky, kterým zajistíme opakované vykonání předchozích příkazů.

4.4.1 Konfigurace XDM-14A

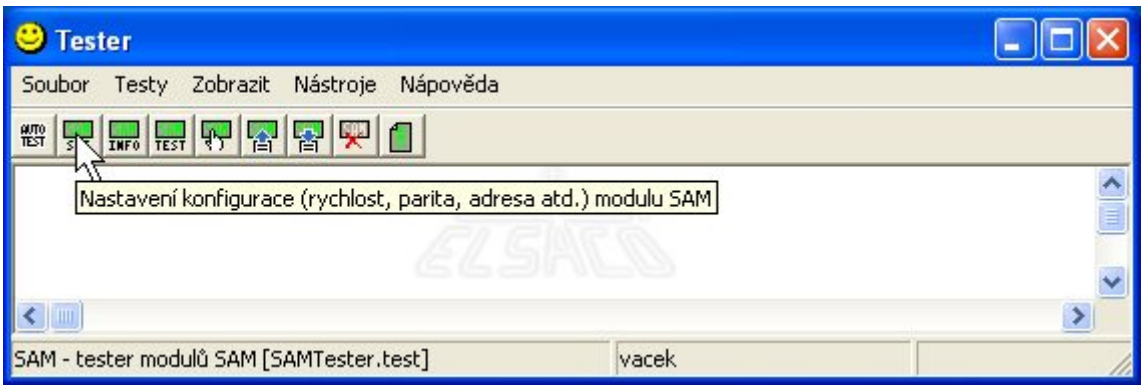
Fyzické propojení počítače, převodníku RS-232/RS-485 (mezi PC a XDM) a zobrazovačem XDM-14A bylo posáno v dřívějších kapitolách několikrát. Připomínáme jen kontrolu údajů na štítku, zejména napájení, typ sériového kanálu a rozmístění signálů na svorkovnici.

na displeji zobrazeny pomlčky, podle kterých uživatel pozná, že došlo ke ztrátě komunikace.

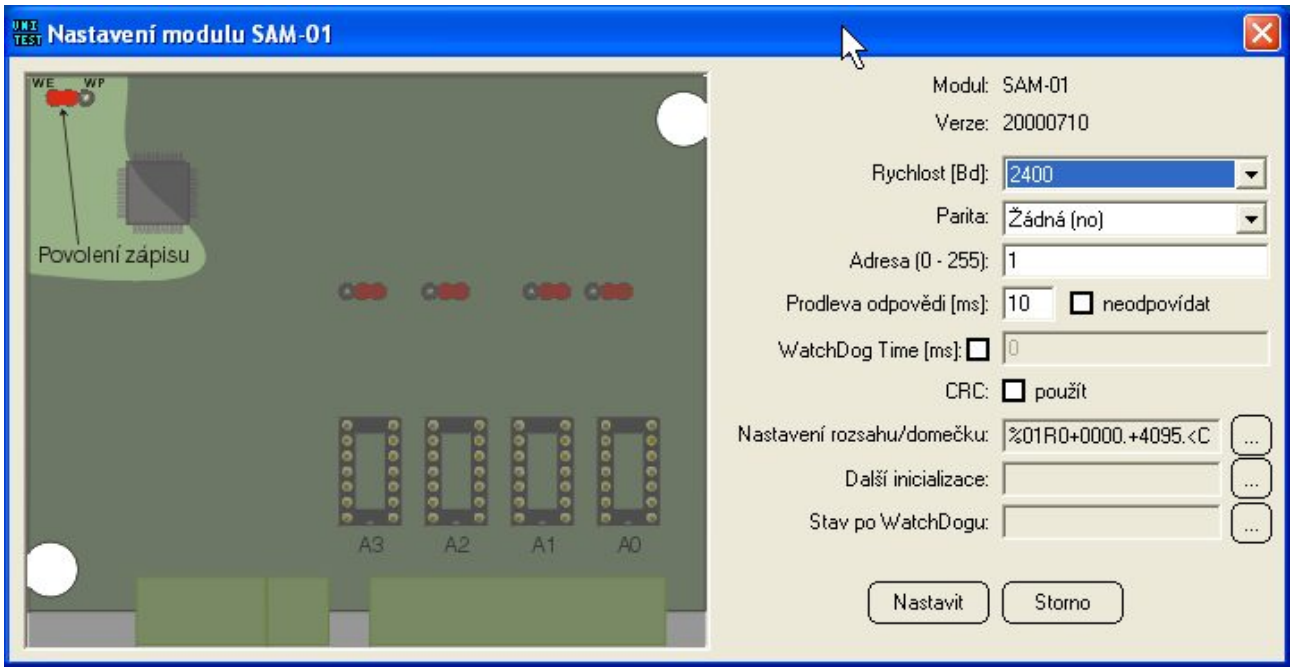
Zvolte podle situace, co je pro vaši aplikaci výhodnější.

Počet míst zadejte 4 (podle skutečnosti zobrazovače).

Inicializační text zadávejte v délce podle zadaného počtu míst na předchozím řádku. Text volte tak, aby něco naznačoval. Zde je po mezeře (aby celkový počet znaků byl 4) zadáno ZAP, takže po zapnutí napáje-



Obr. 42 Start Nastavení konfigurace modulu SAM



Obr. 43 Dialogové okno pro nastavení konfigurace modulu SAM-01

cího napětí (ještě než SAM-01 změří a zakomunikuje) bude na displeji napsáno ZAP.

Další nastavování XDMka pro náš vcelku jednoduchý příklad není potřeba.

Klikneme na tlačítko **Nastavit** a v hlavním okně TESTERu dostaneme zprávu o úspěšně ukončené konfiguraci zobrazovače - viz obr. 41.

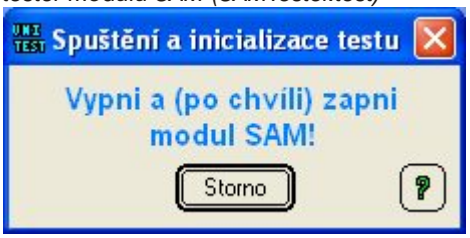
4.4.2 Konfigurace modulu SAM-01

Konfigurace modulu SAM-01 pro aktivní odesílání naměřených hodnot bude o něco složitější.

O fyzickém propojení modulu SAM-01 s počítačem, případně později se zobrazovačem XDM-14A, platí totéž, co v předchozí kapitole. O takovém zapojení jsme psali v této Kuchařce mnohokrát, už víme, že musíme dávat pozor na hodnotu napájecího napětí, označení signálů na svorkách, atd. Také umíme udělat ze čtyřdrátové RS-422 dvoudrátovou RS-485,

Propojíme tedy SAMa podle zvyklostí s počítačem přes převodník RS-232/RS-485 a nastartujeme program TESTER.

V úvodním okně TESTERu zvolíme
SAM - tester modulů SAM (SAMTester.test)



Obr. 44 Výzva pro znovuzapnutí před konfigurací

a opět případně upravíme definici portu (COMu) našeho počítače. To provádíme pomocí volby tlačítkem
Upravit nastavení vybraného testeru

Pokud pak z úvodního okna TESTERu pokračujeme tlačítkem **OK**, bude zobrazeno hlavní okno TESTERu pro moduly SAM - viz obr. 42.

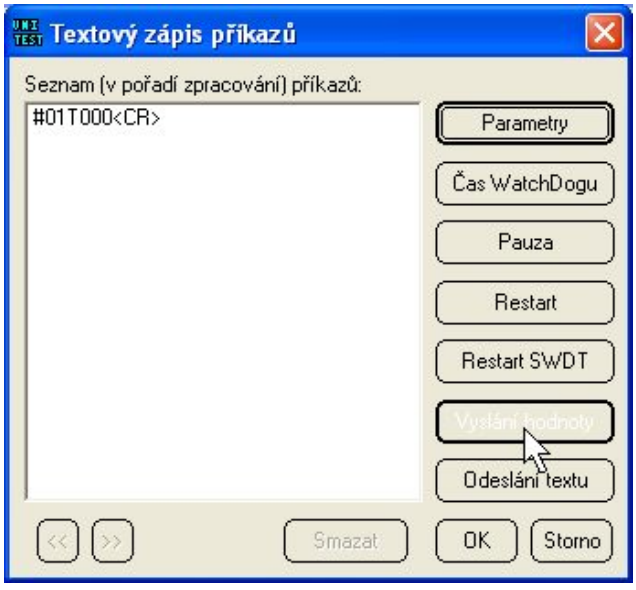
Tam opět pomocí ikonky **SET** přejdeme přímo na nastavování konfigurace připojeného modulu SAM-01.



Obr. 50 Okno pro textový zápis příkazů



Obr. 49 Okno pro zadání akt. odesílání hodnoty



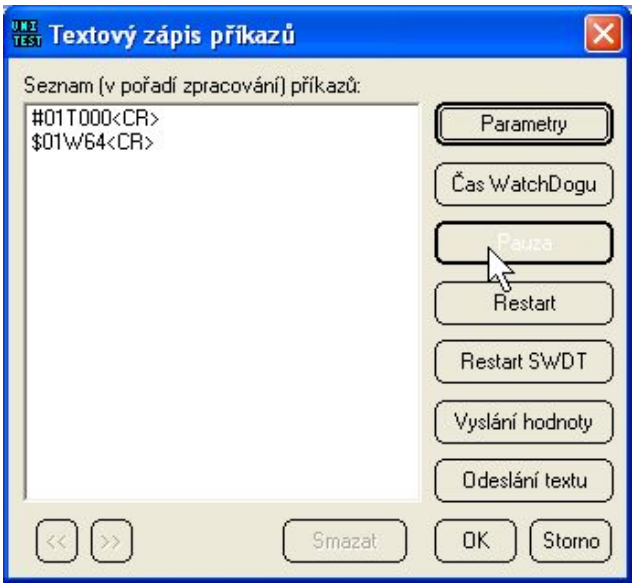
Obr. 45 Vytvořený příkaz pro akt. odesílání hodnoty

Opět naše práce začíná vypnutím a po chvíli znovuzapnutím modulu - viz výzva dle obr. 44.

Po provedení požadovaného bude zobrazeno dialogové okno pro nastavení modulu SAM-01 jako na obrázku 43.



Obr. 46 Okno pro zadání prodlevy



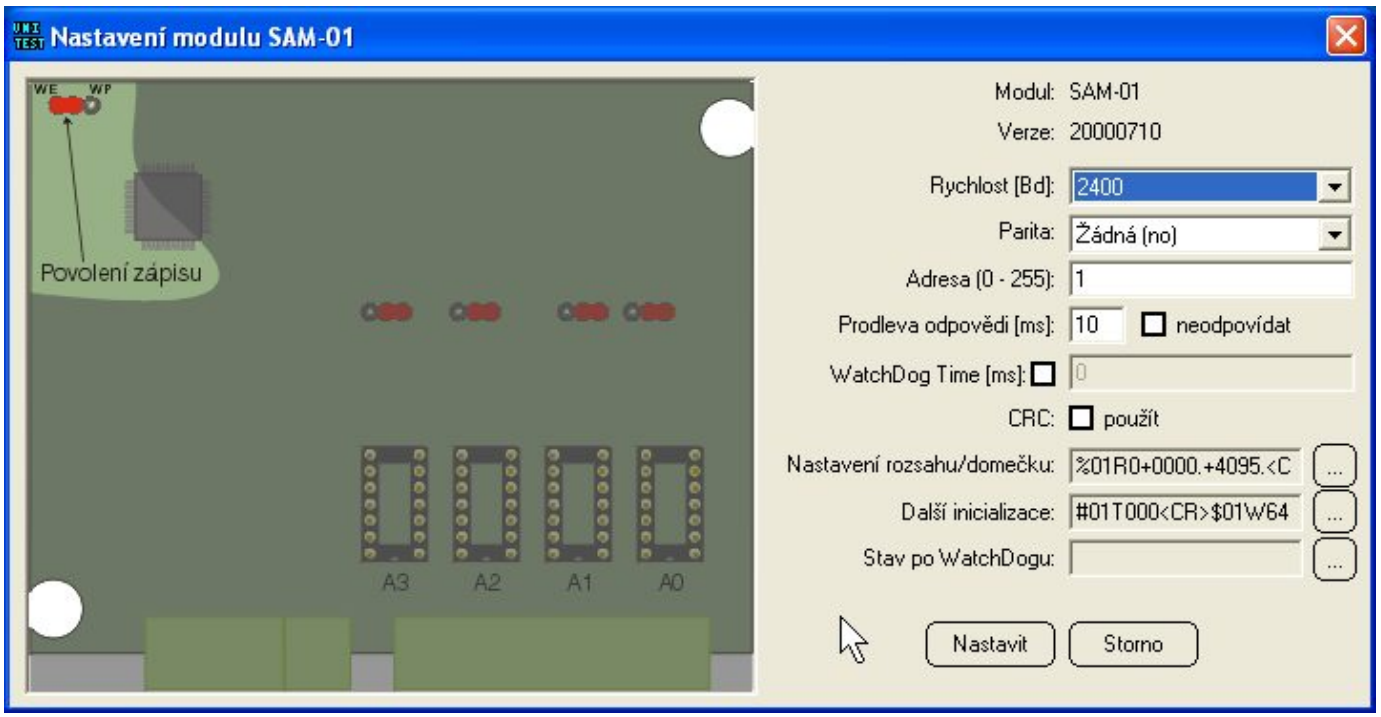
Obr. 47 Okno s vytvořenými příkazy odeslat + počkat



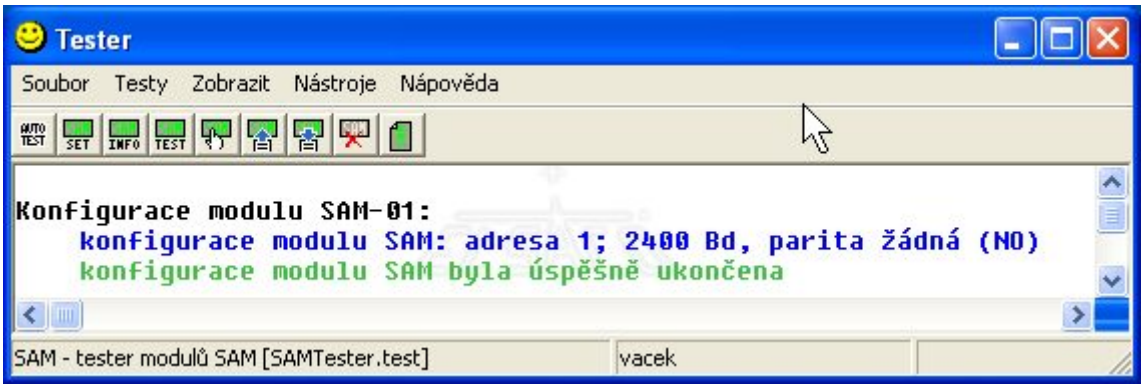
Obr. 48 Okno se všemi třemi příkazy

V tomto okně zadáme ve formuláři požadovanou přenosovou rychlost 2400 Bd, paritu žádnou, adresu SAMa na lince RS-485 volíme 1, prodlevu odpovědi ponecháme 10 ms, odpovídat nechceme - sami odesíláme. WatchDog ani CRC nechceme.

O nastavení rozsahu domečků se zde zmiňovat nebudeme, protože toto je popsáno např. na str. 16.



Obr. 52 Okno Nastavení modulu SAM-01 po zadání doplňujících příkazů v Další inicializaci



Obr. 51 Hlavní okénko TESTERu po ukončení konfigurace modulu SAM-01

Musíme se ale zmínit o dalším nastavení, protože bez něj by k pravidelnému odesílání změřené hodnoty nedošlo.

V řádku *Další inicializace* klikneme na tlačítko s tečkami.

Zobrazí se dialogové okno podle obr. 50, zatím prázdné, ale s tlačítky na své pravé straně. Pomocí nich zadáme do tohoto okna potřebné příkazy.

Jako první bude příkaz pro **Vyslání hodnoty**, druhým bude **Pauza** a posledním příkazem bude **Restart**.

Klikneme-li na tlačítko **Vyslání hodnoty**, zobrazí se další okénko - viz obr. 49. V něm zadáváme nejprve ze kterého kanálu (AD vstupu) modulu SAM-01 chceme změřenou hodnotu posílat protokolem na linku.

Lze zatrhnout políčko pro posílání hodnoty se znaménkem.

Pak volíme počet cifer, naše XDMko má 4.

Na posledním řádku zadáváme adresu displeje, v našem případě nulu (0).

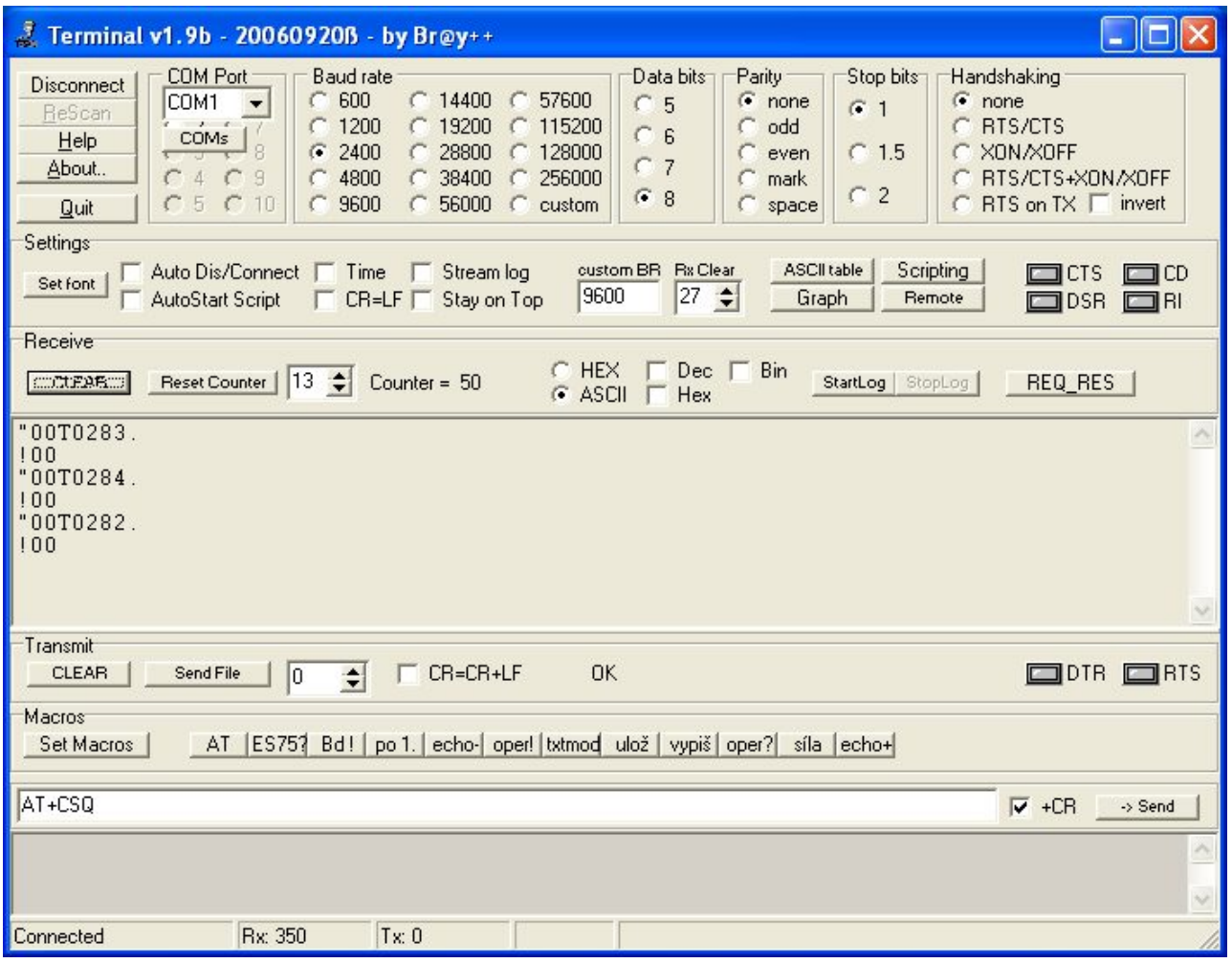
Klikneme na tlačítko **OK** a vidíme, že naše požadavky se automaticky přeložili do příkazu, který je zobrazen v okně **Textového zápisu příkazů** - viz obr. 45.

Poté klikneme v pravé části tohoto okna na tlačítko **Pauza** a dostaneme další okénko **Nastavení pozastavení činnosti** - viz obr. 46. Pomocí něj nastavíme prodlevu - pauzu - na jednu vteřinu tak, že do formuláře zadáme číslo 100. Čas se zadává v desítkách ms a ne ve stovkách, jak je omylem napovídáno v dialogovém okénku. 100 x 10 = 1000 ms = 1 sec.

Po kliknutí na tlačítko **Nastavit** je do dialogového okénka pro **Textový zápis příkazů** doplněn příkaz pro pauzu 1 sec - viz obr. 47.

Pro poslední příkaz stačí kliknout na tlačítko **Restart** - viz obr. 48. A v okénku vidíme i poslední příkaz pro restart SAMa.

Výsledkem by tedy mělo být odeslání změřené hodnoty ze vstupu 0, vteřinová pauza a znovu restart, a to stále dokola.



Obr. 53 Odposlech na lince RS-485; komunikace ze SAM-01 na XDM-14A

V okénku pro textový zápis příkazů klikneme na tlačítko **OK**, protože již obsahuje všechny tři požadované rozšiřující příkazy.

Aspoň zběžně zkontrolujeme požadavky na novou konfiguraci v dialogovém okně *Nastavení modulu SAM-01* - viz. obr. 52. Vše by mělo být podle našich požadavků, schválíme tlačítkem **Nastavit**, konfigurační příkazy se přenesou do modulu SAM a zapíší do jeho paměti FLASH.

Zda byl výsledek našeho snažení o novou konfiguraci úspěšný se dozvíme ve zprávě v hlavním okně TESTERu - viz obr. 51.

4.4.3 Vlastní přenos dat

Obě konfigurace máme úspěšně za sebou, proto můžeme na linku RS-485 připojit obě zařízení najednou. Když nyní k zařízením připojíme napájecí napětí, je nejprve na displeji vypsán text ZAP a po krátké době dojde ke komunikaci. Z modulu SAM-01 je každou vteřinu vyslána zpráva, která obsahuje změřenou hodnotu ze vstupu 0 modulu SAM-01. Zpráva je modulem XDM-14A přijata, vyhodnocena a je zobrazen údaj ze zprávy.

Pokud máme ještě připojen převodník RS-232/RS-485 k oběma zařízením a k počítači, můžeme v počítači spustit terminál a sledovat probíhající komunikační relace.

Odposlech samozřejmě využijeme zejména tehdy, kdy nás technika a naše příkazy nechtějí poslouchat. Pak podle manuálu *Komunikační protokol SAM* můžeme komunikaci sledovat, testovat ji různými příkazy (jednotlivá zařízení samostatně). Stačí na jednom z nich odpojit napájecí napětí nebo komunikační linku. K tomu slouží odnímatelné svorkovnice jak na modulu SAM-01, tak i na zobrazovači XDM-14A.

4.4.4 Stejné řešení jinými příkazy

Úkol odesílat z modulu SAM-01 pravidelně změřenou hodnotu do zobrazovače můžeme řešit i jinak. Zobrazovači XDM-14A ponecháme v našem příkladě konfiguraci stejnou. Jiná bude v modulu SAM-01.

Dostaňme se do konfigurace modulu SAM-01 pomocí programu TESTER jako na obr. 43.

Nyní zvolíme **WD** = 1000 ms a dál nebudeme pokračovat volbou **Další inicializace**, ale volbou **Stav po WatchDogu**. Klikneme tedy v tomto řádku na tlačítko se třemi tečkami, dostaneme okno podle obr. 50 pro textový zápis příkazů. Dále zvolíme tlačítkem **Vyslání hodnoty** zápis příkazu **T** - viz obr. 49 a 45. Pak pokračujeme tlačítkem **Restart SWDT** (*software (re-*

set) watch-dog time) pro zápis příkazu **S**. To je vše, obvyklými tlačítky konfiguraci zapíšeme do modulu a pak znovu propojíme jen modul SAM-01 se zobrazovačem XDM-14A. Zapojení funguje obdobně, SAM-01 vyšle změřenou hodnotu, XDM-14A ji zobrazí a pak se provede restart s nulováním času WD. Proto se hned vše cyklicky opakuje bez čekání (nastavený čas pro WD se uplatní pouze po zapnutí SAMa).

Počet relací za vteřinu lze snížit zvýšením prodlevy pro odpověď. Zadáme-li 200 ms, vyšle SAM-01 změřenou hodnotu ze vstupu 0 asi 5x za vteřinu.

5 Modul SAM–02 (4 logické vstupy a 4 relé)

5.1 Konfigurace modulu SAM-02

Ke konfiguraci opět použijeme výhodný firemní program TESTER. Na štítku modulu SAM-02 si přečteme pro jaké napájení byl vyroben, na jeho napájecích svorkách 6+7 je naznačena polarita napájení. Na svorky pro linku RS-422 zapojíme propojky 2+4 a 3+5 pro převod na linku RS-485.

Zapojení svorek je totožné se zapojením svorek na modulu SAM-01, svorky jsou odnímatelné, lze tedy (pokud děláte pokusy postupně na stole podle této Kuchařky) vyjmout zapojené vodiče se svorkami z modulu SAM-01 a zasunout je do modulu SAM-02. Jedná se o svorky 1 až 7 včetně propojek mezi nimi.

Potom vodič +RxDTxD vedeme na označení stejného pinu převodníku RS-485 na USB/COM, obdobně pak -RxDTxD... Konektor USB/COM převodníku propojíme kabelem s příslušným USB/COMem počítače. Toto propojení je tedy obdobné jako v předchozí kapitole se zobrazovačem XDM-14A.

Zařízení a převodník zapneme, nastartujeme program TESTER, v úvodním okně zvolte testy pro moduly SAM, resp. zkontrolujte nastavení COMu PC jako v kap. 1. Pak zvolíme volbu **INFO**. Jsme vyzvání ke znovuzapnutí SAMu:



Obr. 54 Výzva pro znovuzapnutí

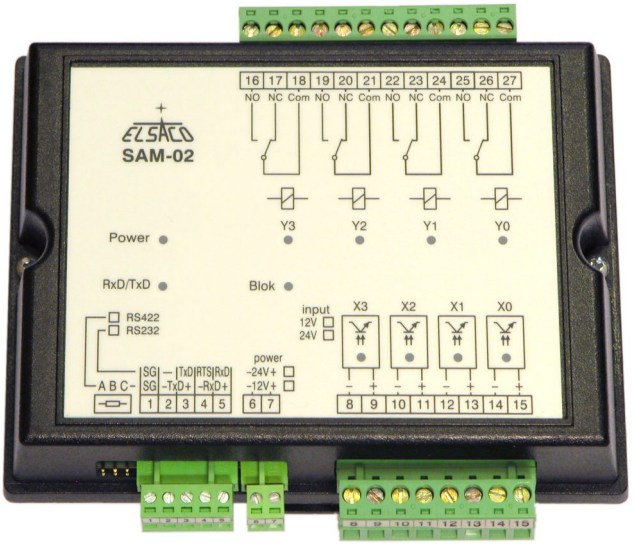
Během tohoto čekání na znovuzapnutí vidíme problikávat LEDky na převodníku - TESTER vysílá požadavek na přepnutí do konfiguračního režimu.

Provedeme znovuzapnutí SAMu a pokud máme SAM přímo z výroby (nebo po výmazu konfigurace) dostaneme následující informaci:



Obr. 55 SAM-02 z výroby - info

Tak máme ověřeno správné propojení - komunikace se SAMem funguje.

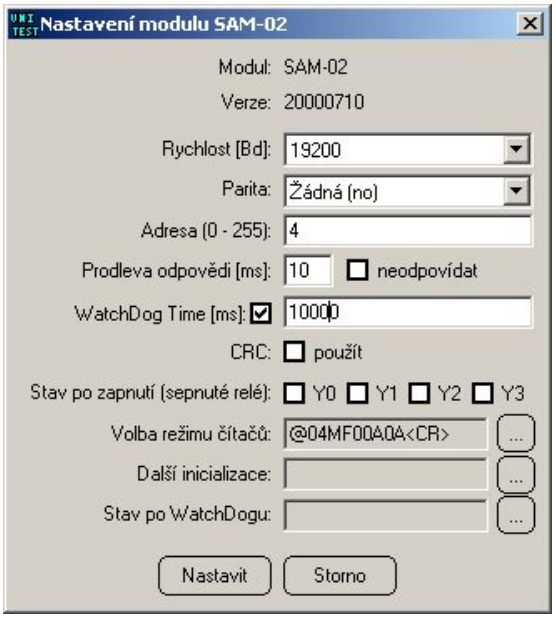


Obr. 57 Foto modulu SAM-02

Nyní budeme chtít nastavit, nakonfigurovat SAMa pro naše pokusy na stole. Chtějme prozměnu nastavit komunikační rychlost na lince RS-485 na 19200 Bd, komunikační adresu SAMa na 4 a WD na 10 sec.

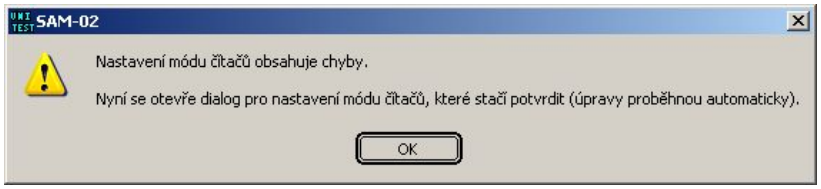
V TESTERu využijeme volbu SET. Nejprve opět dostaneme výzvu k znovuzapnutí - viz obr. 21.

Poté je nabídnuto okno pro editaci konfiguračních parametrů:

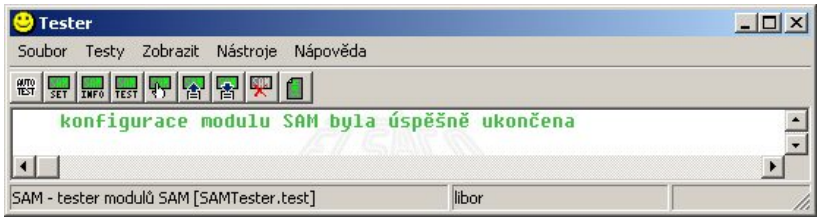


Obr. 56 SAM-02 konfigurace

Zeditujeme rychlost a adresu na požadované, ostatní ponecháme, protože v zadání další požadavky nebyly. Klikneme na tlačítko **Nastavit** a může se objevit toto hlášení - viz obr. 61. Schválíme tlačítkem **OK** a dostaneme se do okna pro nastavení čítačů, které jsme původně nepotřebovali. V okně je předchystána defaultní definice čítačů, kterou schválíme - viz obr. 58 - kliknutím na tlačítko **Nastavit**.



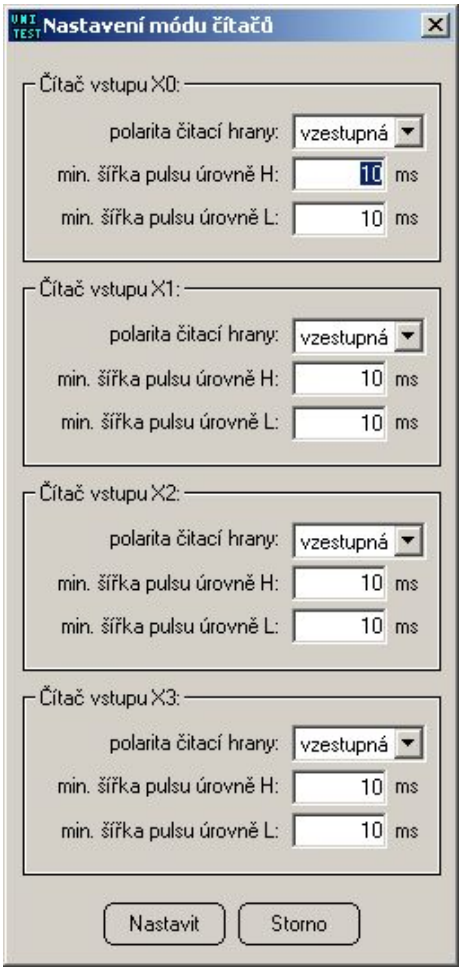
Obr. 61 Hlášení pro SAM-02



Obr. 59 Zpráva v hlavním okně TESTERu



Obr. 60 Vyčtené nastavení pro SAM-02



Obr. 58 Konfigurace čítačů

Tím jsme prošli potřebnou konfiguraci modului SAM-02 - o tom nás informuje zpráva v hlavním okně TESTERu - viz obr. 59.

A protože jsme opatrní, raději si znovu vyčteme pomocí volby **INFO**, jak je modul SAM-02 nastaven, abychom měli jistotu zejména v komunikační adrese a rychlosti. Opět dostaneme nejprve výzvu pro znovuzapnutí SAMa - viz obr. 21, po provedení znovuzapnutí SAMa se v dialogovém okně objeví vyžádané informace o jeho nastvení - viz obr. 60.

Po kontrole údajů v okně **Aktuální konfigurace** můžeme konstatovat, že máme nastavenu požadovanou adresu **4**, komunikační rychlost **19200** Bd a čas WD **10000** ms, tj. 10 vteřin.

5.2 Základní příkazy pro SAM-02

5.2.1 Ovládání relé

SAM-02 obsahuje 4 relé, které budeme chtít komunikačními relacemi (příkazy) ovládat. Reakci po relaci budeme vidět hned, proto začněme nejprve tímto příkazem.

V manuále **Komunikační protokol SAM** nalezneme kap. 3.2.1 **Nastavení stavu výstupů (SAM-02)** na str. 4.

Vidíme, že relace má obecný tvar:

@aan<CR>

kde
aa je adresa modulu
n číslo odpovídající po bitech výstupům Y,
bit 0 - Y0, ..., bit 3 - Y3

Odpověď je-li vše OK pak:

!aa<CR>

Modul SAM-02 máme připojen přes převodník k PC tak jako v předchozí kapitole, spustíme program TERMINAL a v něm nejprve zeditujeme parametry pro komunikaci po sériové lince PC. Měnili jsme rychlost, nastavíme ji tedy na 19200 Bd, ostatní parametry jsou vlastně stejné jako v pokusech se SAM-01. Zkontrolujte na obr. 62 v horní části hlavního okna TERMINALu.

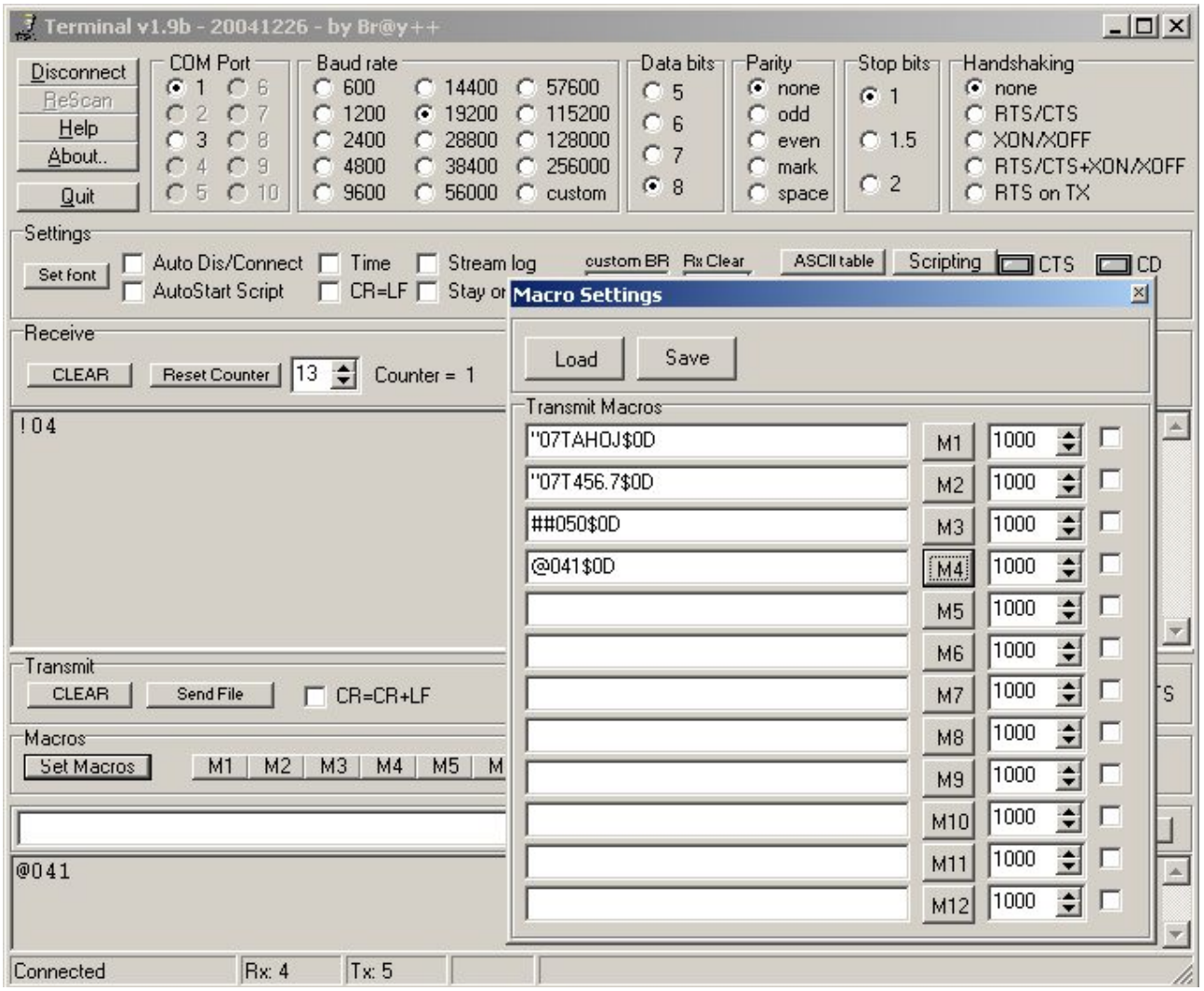
Abychom mohli pohodlně udělat pokus se SAMem, nadefinujeme si makro **M4** pro sepnutí prvního relé, tj. výstupu Y0. Zpráva by měla vypadat:

@041<CR>

a protože již víme, jak na znak <CR> z minulých pokusů, zapíšeme do řádku pro makro M4 (po kliku na tlačítko **Set Macros** se otevře okno Macro Settings - viz obr. 62):

@041$0D

Pak klikneme nahoře v hlavním okně TERMINALu na **Connect** (změní se na **Disconnect**) a na tlačítko



Obr. 62 SAM-02 v okně TERMINALu

makra **M4**. Je vyslána relace - vidíme její výpis ve spodní části okna TERMINALu (a v horní části okna vidíme přijatou odpověď **!04**), uslyšíme klapnout relé v modulu SAM-02 - vizuálně zkontrolujeme, že na něm svítí i LEDka **Y0**, zhasne LEDka **Blok** - a po dobu WD, tj. 10ti vteřin, tento stav trvá. Potom relé odpadne, LEDka **Y0** zůstává svítit jako poslední požadavek, ale protože nenásleduje další relace, rozsítí se červená LEDka **Blok** jako informace o chybějící další relaci. Pokud u makra **M4** zašrtneme volbu pro jeho opakování, opět po relaci dojde k sepnutí relé a zhasnutí červené LED **Blok**... Tak jsme si osahali relaci pro sepnutí relé, kterou můžeme měnit a spínat či odpínat relé modulu. Pokud chceme sepnout více relé naráz, sečítáme váhy bitů jednotlivých relé a tento bajt zapíšeme do relace místo jedničky. Tak např. pro znak **F** sepnou všechna relé.

Pro naše pokusy je vhodné zapojit kontakty relé tak, aby postupně od **Y0..Y3** napájeli při svém sepnutí vstupy **X0..X3**. To proto, abychom při dalším pokusu dokázali přečíst měnící se vstupní signál. Toto zapojení rovněž vyžaduje test modulu SAM-02 v programu TESTER. Když pak sepneme relé **Y0**, napájí jeho kontakt vstup **X0** a u něj se rozsvítí na modulu SAM-02



Obr. 63 Makra M4 a M5 v TERMINALu

LEDka. Ukončeme tedy pokusy s ovládáním relé podle našeho příkladu tak, že makro **M4** je nadefinováno pro sepnutí prvního relé **Y0**, jehož kontakt napájí vstup **X0**. U makra **M4** zašrtněme volbu pro jeho opakování po 1000 ms.

5.2.2 Čtení vstupů

A nyní zkusíme vytvořit makro **M5** pro vyčtení stavu vstupů **X0..X3** modulu SAM-02. Podle manuálu **Komunikační protokol SAM** v kap. 3.2.5 **Čtení okamžité hodnoty vstupů (SAM-02)** vypadá komunikační požadavek takto:

`@aaI<CR>`

a odpověď:

`!aaoi<CR>`

kde

aa = adresa modulu

o = aktuální stav výstupů Y0..Y3 (výstupní registr)

i = aktuální stav vstupů X0..X3 (vstupní registr)



Obr. 64

Vidíme, že na požadavek přečtení aktuálního stavu vstupů dostáváme v odpovědi po adrese nejprve aktuální stav výstupů (v jednom bajtu) a teprve v dalším bajtu aktuální stav vstupů. Jednotlivé váhy v bajtu pak vždy odpovídají jednotlivým bitům / vstupům / relé, a to postupně od nejnižších...

Adresa modulu SAM-02 byla nastavena na 4, proto požadavek na čtení vstupů ve tvaru pro řádek makra **M5** bude vypadat takto (viz obr. 63):

```
@04I$0D
```

Tak máme nachystána dvě makra, **M4** pro sepnutí relé Y0 a **M5** pro vyčtení vstupů. Pokud nejprve klikneme na tlačítko **M4**, dojde k sepnutí relé Y0, potom hned klikneme na tlačítko **M5**, dojde k vyčtení vstupů. Jak odpovědi na relace mají vypadat jsme si rovněž popsali, podívejme se nyní do okna TERMINALu jak opravdu vypadají - viz obr. 64.

První požadavek **M4** vidíme zobrazen jako **@041**, odpověď na něj je v prvním řádku jako **!04**. Druhý požadavek **M5** je zobrazen jako **@04I** a odpověď na něj jako **!0411**.

Zajímá nás nyní druhá odpověď, kterou máme určit stav výstupů. Po vykřičníku a adrese 04 následují dvě jedničky.

První jednička je stav výstupů - podle ní poznáme, že jsme do modulu SAM-02 dříve vyslali požadavek na sepnutí relé Y0. Druhá jednička je stav vstupů, je tedy vstup X0 aktivní - je na něm vstupní signál, který hlásí např. poruchu apod.

Makra si můžeme upravit, abychom si ověřili, že dokážeme sepnout jiná relé a vyčíst stavy jiných vstupů.

Můžeme opět konstatovat, že nás modul SAM-02 poslouchá a přikročit tak k jeho naprogramování ve FREDovi.

5.3 SAM-02 ve FREDovi

5.3.1 Spínání relé

Začneme opět ovládáním relé modulu SAM-02. Jak vypadá komunikační požadavek jsme si již řekli:

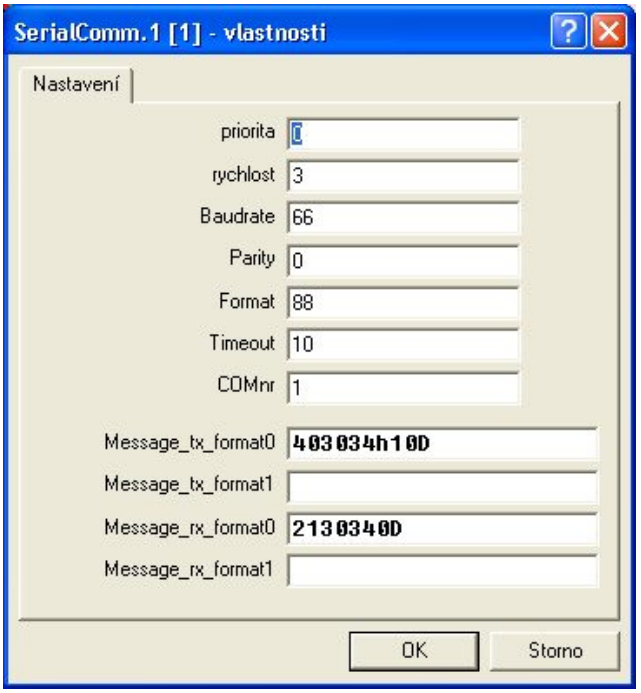
```
@aan<CR>
```

kde
aa je adresa modulu
n číslo odpovídající po bitech výstupům Y,
bit 0 - Y0, ..., bit 3 - Y3

Odpověď je-li vše OK pak:

```
!aa<CR>
```

Tyto obecné relace upřesníme pro náš příklad na adresu 04 a formátovací řetězec pro vyslání požadavku a pro příjem odpovědi zadáme do prvního modulu SERIALCOMM v projektu (projekt je opět součástí Kuchařky, která je ke stažení na našich stránkách; projekt se jmenuje **SAM02.opf**). Řetězec bude celkem jednoduchý, pokud budeme chtít na linku vysílat přímo definované znaky, zapíšeme je podle ASCII tabulky ve tvaru hexa.



Obr. 65 Požadavek pro sepnutí relé

Zavináč jako první znak má hodnotu 40, adresu 04 zapíšeme jako bajty 30 a 34.

Pak potřebujeme posílat bajt (integer hodnotu), který nám určí, která relé sepnou. V tom nám pomůže HELP PL2 s nápovědou pro formátovací řetězec pro vysílání:

příkazy hn, in, jn, kn slouží pro vyslání integer hodnoty ze vstupů Tx0..3 jako n znaků ve formátu ASCII hexa
např. h4 vyšle ve formátu ASCII hexa 4bytový string, vyjadřující hexadecimálně integer hodnotu na vstupu Tx0.
Direktivy h..k se hodí i pro práci s binárními hodnotami HW modulů SAM ve spojení s moduly BINtoINT a INTtoBIN.
Typický příkaz pro nastavení binárních výstupů SAMu-02 (adresa=1) bude 403031h10D, kde na vstup Tx0 bude připojen výstup BINtoINT (pro SAM-02 stačí, budou-li funkční první čtyři binární vstupy).

Použijme tedy v projektu modul BINtoINT, na jeho vstupy 0, 1, 2 a 3 přivedeme z projektu signály na sepnutí relé Y0, Y1, Y2 a Y3. Na jeho výstupu INT pak budeme mít ineger číslo, které přivedeme na vstup TX0 modulu SERIALCOMM (viz projekt obr. 67).

Pak podle HELPu bude následující direktiva ve formátovacím řetězci zapsána jako **h1**.

Posledním bajtem v řetězci bude <CR>, vyjádřeno hexa jako **0D**. Relace - požadavek:

```
403034h10D
```

A odpověď? Vcelku jednoduchá. Vykřičník, adresa a <CR>:

```
2130340D
```

Vše vidíte nadefinováno v dialogovém okně modulu SERIALCOMM na obr. 65.

5.3.2 Čtení vstupů

Obdobně se zachováme při čtení vstupů modulu SAM-02. Obecný požadavek na čtení:

@aaI<CR>

a odpověď:

!aaoi<CR>

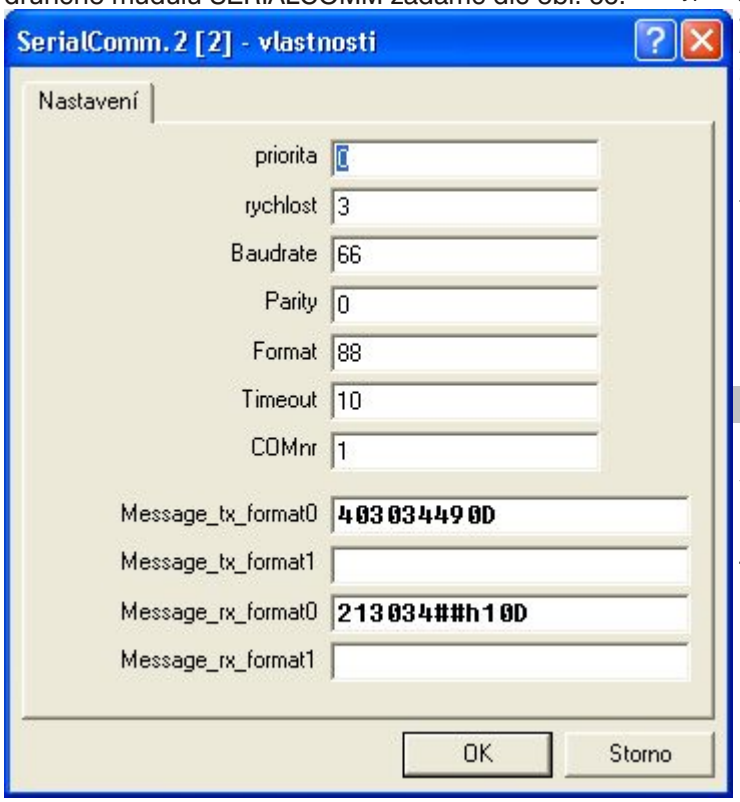
kde

aa = adresa modulu

o = aktuální stav výstupů Y0..Y3 (výstupní registr)

i = aktuální stav vstupů X0..X3 (vstupní registr)

Pro náš příklad opět upřesníme na adresu 04 a do druhého mudulu SERIALCOMM zadáme dle obr. 66:



Obr. 66 Požadavek na čtení vstupů

Vidíte, že vysílaný požadavek je vcelku jednoduchý, pouze požadované znaky převedeme do tvaru hexa opět podle ASCII tabulky.

S odpovědí je to poněkud složitější, první znaky nám problém neudělají. Vykřičník je **21** hexa, adresa opět dva bajty **30** a **34**, potom nás v podstatě nezajímá další bajt, ve kterém je integer hodnota stavy výstupů (která relé jsou sepnuta).

V HELPu pro přeskočení bajtu nalezneme použití do formátovacího řetězce pro příjem znak vězení #. Bude-li například vrácena jednička , bude v hexa tvaru vráceno 31. Proto musíme zapsat znaky # dva: **##**.

Viz HELP:

příkaz ## (2 znaky "vězení") = je ignorován 1bajt (znak) v příjmu zprávy

V HELPu také nalezneme jak se chovat k příjmu integer hodnoty:

příkaz hn až kn (malá písmena h, i, j, k s cifrou) = přečte n znaků ve formátu ASCII hexa a uloží jako hodnotu integer do výstupu Rx0..3

např. h4 přijme čtyři znaky ve formátu ASCII hexa a uloží je jako int do Rx0 jako první znak čísla může být + nebo - (význam znaménka)

nebo > či < (interpretuje se jako + či -; kvůli modulům SAM-01/02/04).

Typický formátovací řetězec pro vyčtení čítačů z modulu SAM-02 pak bude h52Ci42Cj42Ck40D, kde první hodnota bude buď kladná nebo záporná podle toho, jestli je na začátku > nebo <.

Protože očekáváme jeden bajt s integer hodnotou a v projektu jej chceme poslat na výstup RX0 druhého modulu SERIALCOMM bude do formátovacího řetězce zapsána direktiva **h1**.

A na konec zprávy oblíbený znak <CR>, v hexa vyjádření pak **0D**.

Celý řetězec pak vypadá takto (viz obr. 66.):

213034##h10D

5.3.3 Projekt

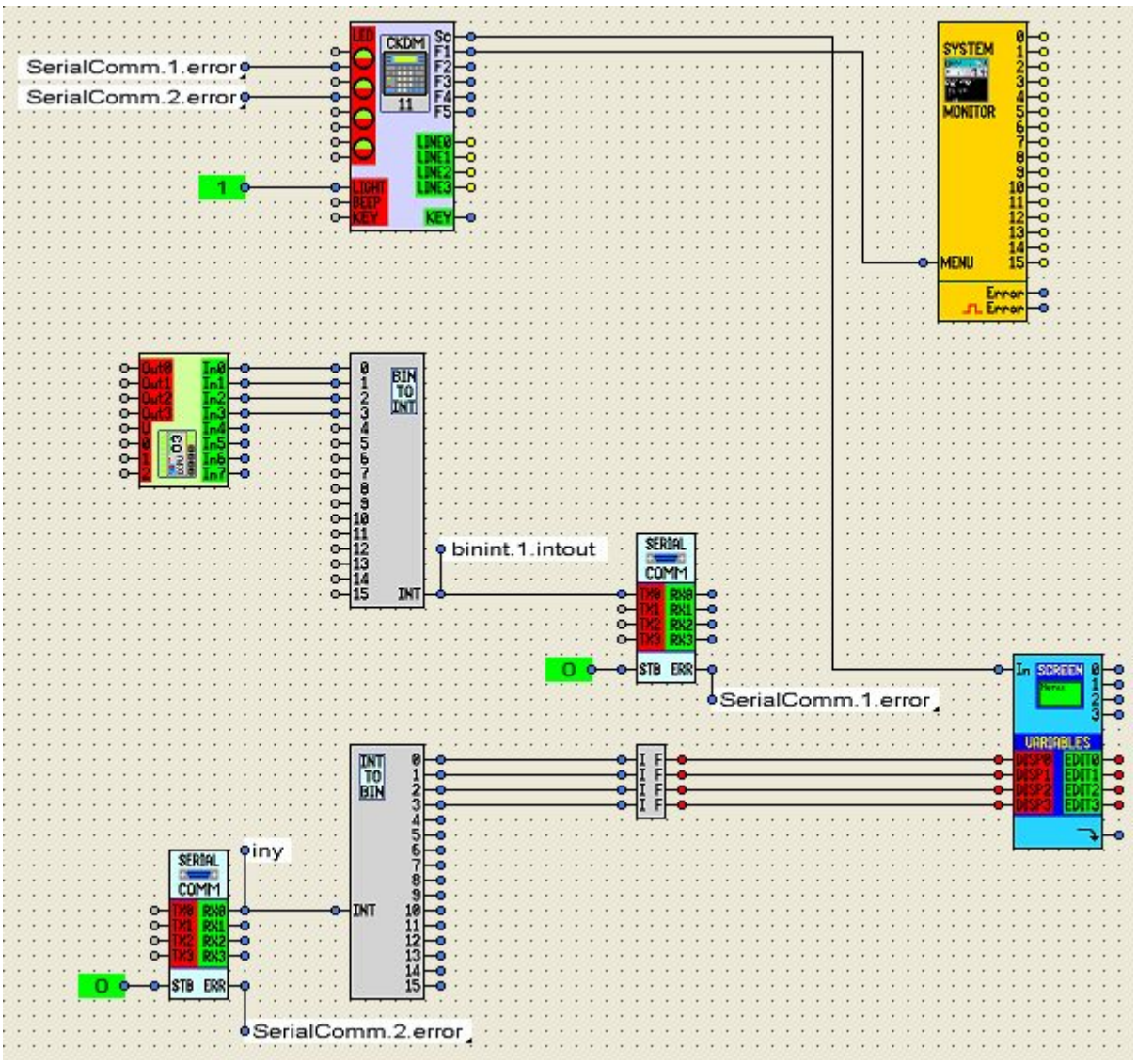
Pokud si prohlédnete projekt na obr. 67, všimněte si, že signály In0..3 z centrály CCPU-21 jsou vedeny do modulu BINtoINT, jehož výstup je zaveden do prvního modulu SERIALCOMM. Tak ovládáme relé modulu SAM-02.

Vyčtenou ineger hodnotu vstupů pak druhý modul SERIALCOMM výstupem RX0 posílá do modulu INTtoBIN, který ji váhově rozliší, a přes hradlo Int2float nabídneme tyto hodnoty modulu SCREEN pro jejich zobrazení.

Projekt přeložíme a můžeme si pak ověřit přímo na displeji centrály CCPU-21 jak nás poslouchá spínání relé a vyčítání stavu vstupů modulu SAM-02.

Pro rychlejší odezvu na změnu jsou moduly umístěny v nejrychlejší 100 ms programové smyčce - mají tedy nastaven parametr rychlost = 3.

Moduly SERIALCOMM pak mají nastaveny parametry pro komunikační rychlost 19200 Bd, protože tak byl dříve modul SAM-02 nakonfigurován, proto má parametr Baudrate hodnotu 66.



Obr. 67 Projekt se SAM-02

6 Modul SAM–04 (16 logických vstupů)

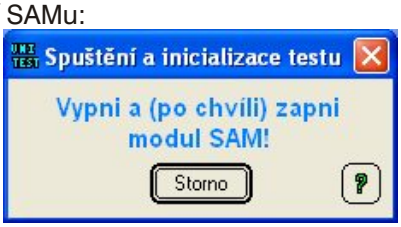
6.1 Konfigurace modulu SAM-04

Ke konfiguraci opět použijeme výhodný firemní program TESTER. Na štítku modulu SAM-04 si přečteme pro jaké napájení byl vyroben, na jeho napájecích svorkách 6+7 je naznačena polarita napájení. Na svorky pro linku RS-422 zapojíme propojky 2+4 a 3+5 pro převod na linku RS-485.

Zapojení svorek je totožné se zapojením svorek na modulu SAM-01/02, svorky jsou odnímatelné, lze tedy (pokud děláte pokusy postupně na stole podle této Kuchařky) vyjmout zapojené vodiče se svorkami z modulu SAM-01/02 a zasunout je do modulu SAM-04. Jedná se o svorky 1 až 7 včetně propojek mezi nimi.

Potom vodič +RxDTxD vedeme na označení stejného pinu převodníku RS-485 na USB/COM, obdobně pak -RxDTxD... Konektor USB/COM převodníku propojíme kabelem s příslušným USB/COMem počítače. Toto propojení je tedy obdobné jako v předchozích kapitolách při konfiguraci zobrazovače XDM-14A či modulů SAM-01/02.

Zařízení a převodník zapneme, nastartujeme program TESTER, v úvodním okně zvolte testy pro moduly SAM, resp. zkontrolujte nastavení COMu PC jako v kap. 1. Pak zvolíme volbu **SET**. Jsme vyzváni ke znovuzapnutí SAMu:



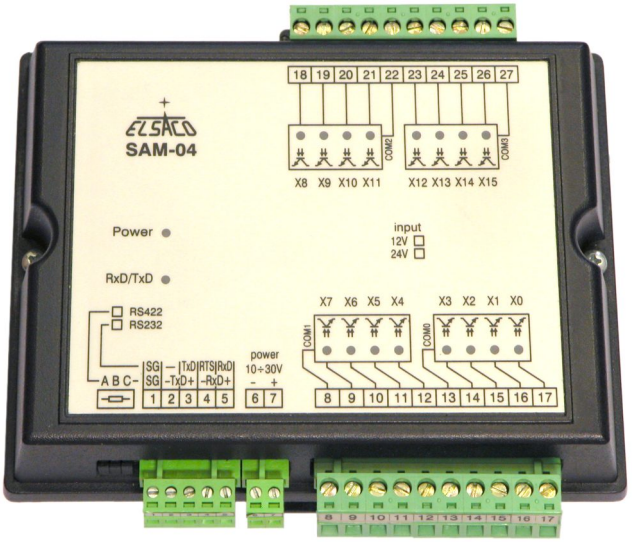
Obr. 69 Výzva k znovuzapnutí

Po znovuzapnutí SAMa se objeví dialogové okno pro jeho konfiguraci (obr. 70), ve kterém nastavíme požadované komunikační, případně i další, parametry SAMa. Volíme komunikační rychlost 9600 Bd, bez parity, adresu 7, prodlevu odpovědi 10 ms, bez WD a bez CRCu.

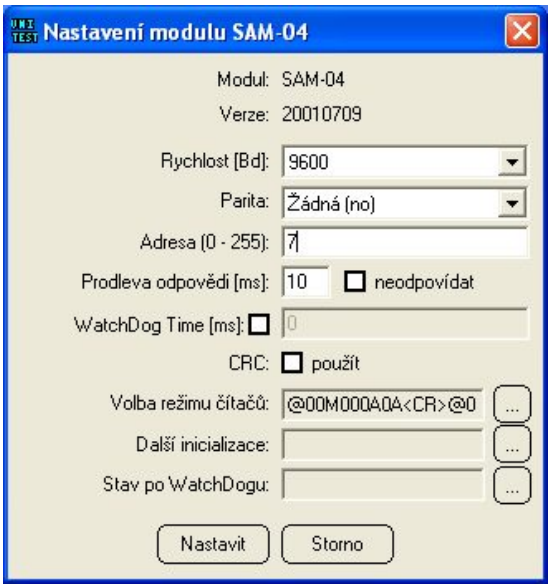
Pokud chceme využít čítačový režim, nadefinujeme jej v dalším dialogovém okně, které se otevírá po kliknutí na tlačítko "..." v řádku **Volba režimu čítačů** - viz obr. 72. V něm lze zvolit polarity čítacích hran (sestupná / vzestupná) a minimální šířky pulzů jak pro úroveň H, tak i L. Po editaci na požadované hodnoty schválíme kliknutím na tlačítko **NASTAVIT**.

Tím se vrátíme zpět do okna pro celkové nastavení SAMa (obr. 70), které ještě jednou zkontrolujeme, zda máme nastaveno požadované a schválíme kliknutím na tlačítko **NASTAVIT**.

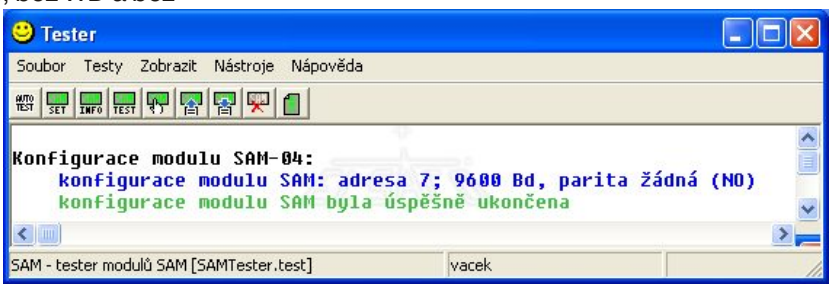
Navolená konfigurace modulu SAM-04 je nahrána do paměti a v hlavním okně TESTERu je pro nás vypsána informace o provedeném nastavení - obr. 71.



Obr. 68 Foto modulu SAM-04

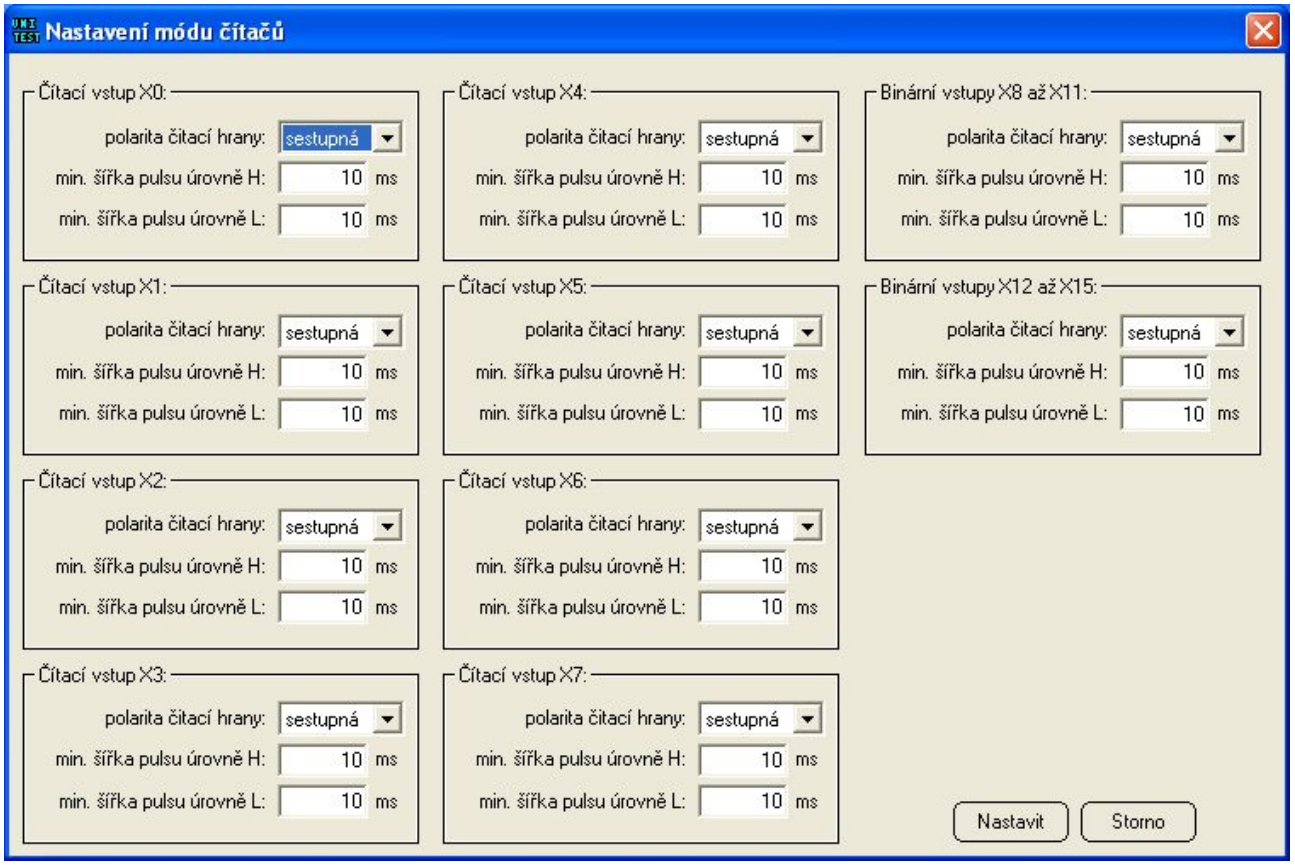


Obr. 70 Dialog pro nastavení modulu SAM-04



Obr. 71 Zpráva o provedeném nastavení modulu SAM-04

Správnost nastavení si jako vždy můžeme překontrolovat v TESTERu pomocí volby **INFO**. Opět jsme vyzváni k znovuzapnutí modulu a po vykonání tohoto příkazu dostáváme informační okno s vyčtenými parametry SAMa, které jsme právě nastavili - viz obrázek 73. Tak jsme si ověřili, že máme nastavenu komunikační rychlost 9600 Bd, bez parity a bez CRCu, adresu



Obr. 72 Volab režimu čítačů pro modul SAM-04

7, prodlevu odpovědi 10 ms. S těmito známými parametry pak můžeme přikročit k dalšímu, a to k seznámení se s relacemi pro vyčítání informací z modulu SAM-04.

6.2 Čtení okamžité hodnoty vstupů

V manuálu **Komunikační protokol SAM** se v kapitole **3.2.7** dozvíme, jak vypadá požadavek (relace) pro čtení okamžité hodnoty vstupů z modulu SAM-04:

`@aaI<CR>`

aa = adresa modulu

V odpovědi se dozvíme očekávaný stav vstupů:

`!aaccdd<CR>`

aa = adresa modulu
cc = stav vstupů 0-7
dd = stav vstupů 8-15

Takto vybaveni znalostmi spustíme program TERMINAL, nastavíme v něm komunikační parametry podle našeho SAMa (použitý COMport PC, 9600Bd, 8 datových bitů, bez parity, 1 stopbit, Handshaking none - viz obr. 74) a nadefinujeme makro pro čtení stavu vstupů z modulu SAM-04.

V našem příkladu chceme vysílat požadavek

`@07I<CR>`

což v řádku pro makro **M1** zapíšeme podle QuickHelpu Terminalu jako

`@07I$0D`



Obr. 73 Info o nastavení modulu SAM-04

Máme-li vše správně zapojeno, pak po kliknutí na tlačítko **M1** vyšleme požadavek na vyčtení vstupů relací, což vidíme v dolním okně **Transmit** programu TERMINAL, modul SAM-01 odpoví a jeho odpověď čteme v horním okně **Receive**:

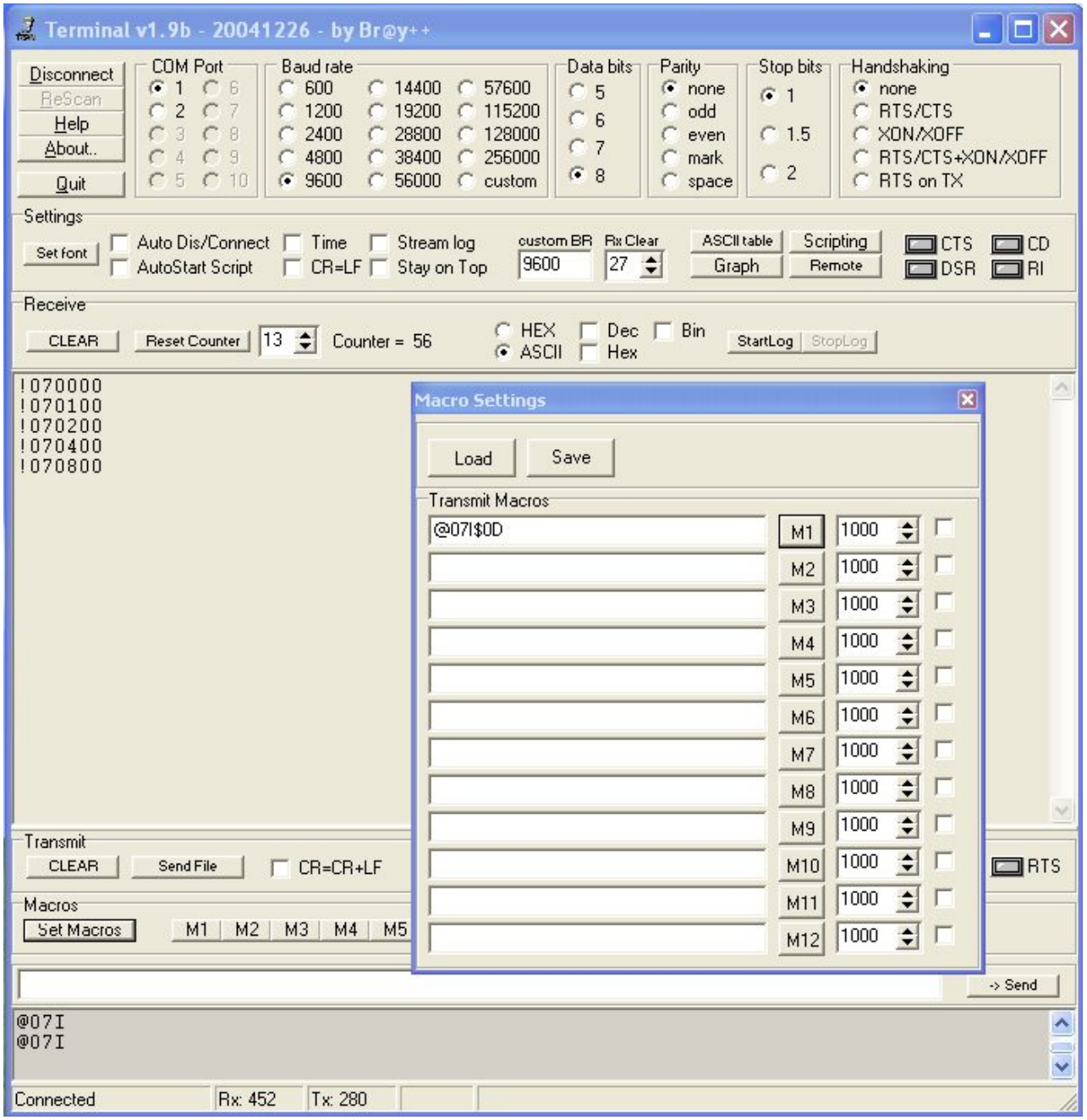
`!070000`

Po **!** a adrese **07** další **0000** znamená, že máme všechny vstupy v nule.

Přivedeme na vstup **X0** napětí pro logickou jedničku (opět její maximální hodnota je uvedena na štítku modulu - viz zaškrtávací políčka **input 12** / **24** V). Spustíme makro **M1** a čteme odpověď:

`!070100`

Bajt **cc** = 01, tzn., že **X0** = 1.



Obr. 74 Vyčítání vstupů modulu SAM-04 pomocí programu TERMINAL

Přepojíme napájení z X0 na X1 a opět spustíme makro M1. Odpověď:

```
!070200
```

Bajt **cc** = 02, vstup **X1** má váhu 2, je tedy v logické jedničce pouze on.

Podobně na obr. 74 vidíte odzkoušení vstupů **X2** s váhou **4** a **X3** s váhou **8**. Obdobně tento princip platí i pro ostatní vstupy s tím, že váhy pro bajty **cc** a **dd** postupně po jednotlivých bitech bajtu narůstají v řadě 1 2 4 8 16 32 64 a 128.

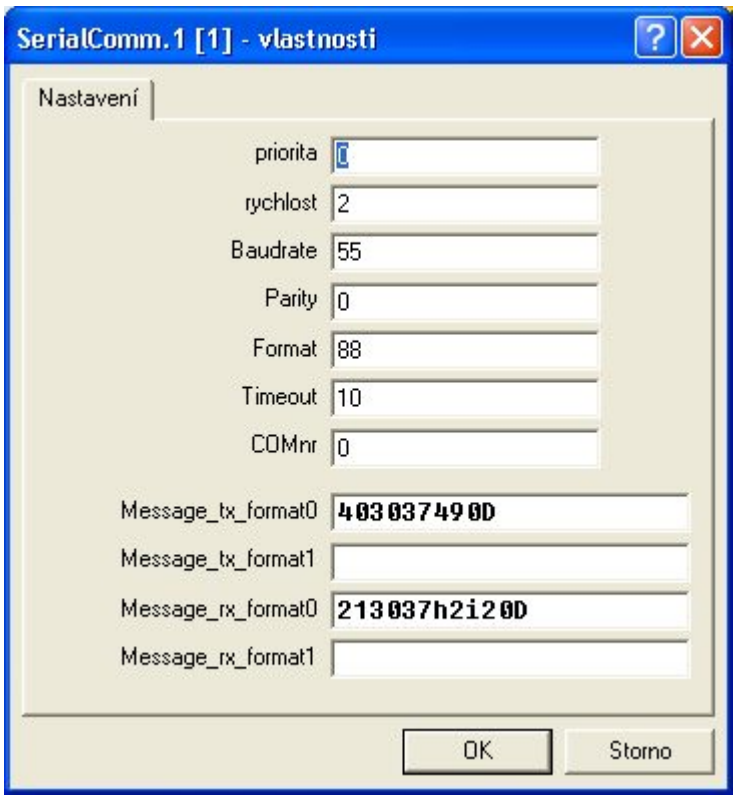
Na základě toho odzkoušení už víme, jaké relace budeme potřebovat stvořit pro modul SERIALCOMM ve FREDovi, můžeme tedy přikročit k dalšímu.

6.3 SAM-04 ve FREDovi

Opět budeme postupovat obdobně jako u předchozích SAMů. Seznámili jsme se se základními relacemi SAM-04, které se budeme snažit ve FREDovi správně nadefinovat pomocí formátovacího řetězce do knihovního modulu SERIALCOMM. Víme, že požadavek na čtení vstupů pro našeho SAMa má podle manuálu **Komunikační protokol SAM** vypadat:

```
@07I<CR>
```

Ve FREDovi pak musíme vytvořit formátovací řetězec pro požadavek (vysílanou zprávu z COMu centrály) podle HELPu. Ten napovídá, že konkrétně požadované znaky vyšleme na linku pomocí zápisu jednotlivých znaků ve tvaru hexa (velkými písmeny). Proto do



Obr. 75 Požadavek na čtení vstupů

řádku pro parametr **Message_tx_format0** v dialogovém okně modulu SERIALCOMM zapíšeme:

```
403037490D
```

Dále musíme nadefinovat formátovací řetězec pro odpověď, tj. pro zprávu, kterou SAM na požadavek odpovídá. Podle předchozí kapitoly víme, že očekáváme zprávu ve tvaru

```
!aaccdd<CR>
```

aa = adresa modulu
cc = stav vstupů 0-7
dd = stav vstupů 8-15

kterou opět musíme převést na správný formátovací řetězec do parametru **Message_rx_format0** stejného modulu SERIALCOMM.

S prvními znaky problém mít nebudeme. Přímo zapsané hexadecimální číslo **00** až **FF** (velkými písmeny) porovná přijatý znak, souhlasí-li, pokračuje dalším dekódováním, jinak končí a zprávu ignoruje.

Toto pravidlo se týká neměnných znaků ve zprávě, tj. uvozovacího vykřičníku, adresy a posledního znaku **<CR>** ve zprávě (celá definice zpráv viz obr. 75).

Bajty **cc** a **dd** se mění s ohledem na stav jednotlivých vstupů modulu SAM-04, na ty musíme jinak. Víme, že SAM stav 16ti vstupů předává vlastně ve tvaru ASCII hexa (16 bitů, integer hodnota). V HELPu se dočteme, že pro tento případ slouží ve formátovacím řetězci příkazy **hn** až **kn** (malá písmena **h**, **i**, **j**, **k** s cifrou). Pak je přečteno **n** znaků ve formátu ASCII hexa a odpovídající integer hodnota vyslána na odpovídající výstup **Rx0..3** (*hn pro Rx0, ...*).

Podle HELPu by tedy bylo nejjednodušší zvolit formát **h4**, ale můžeme také zvolit formát **h2i2** a rozebírat tak každý bajt **cc** a **dd** zvlášť. To si můžete vyzkoušet. Nyní zvolíme druhou variantu a poskládáme požadovaný řetězec:

```
213037h2i20D
```

Základní nastavení parametrů modulu SERIALCOMM už považujeme za samozřejmost:

```
priorita = 0
rychlost = 2
Baudrate = 55 (pro 9600Bd)
Parity = 0 (bez parity)
Format = 88 (pro 8 datových bitů)
Timeout = 10
COMnr = 0
(podle připojení SAMa na COM centrály)
```

Poznámka:
Pokud máte SAMa v provedení na RS-232, nezapomeňte (jako já zpočátku), že nestačí navolit parametr COMnr=0. Musíte také odklonit hlavní kanál pomocí systémového menu centrály z COMu0 na jiný!

Pokud na uvedené zapomenete, hlásí vám SERAILCOMM error=1, relace jsou velmi pomalé a k řádnému příjmu zprávy SERIALCOMMem vůbec nedojde.

S těmito znalostmi můžeme vytvořit projekt, obsahující komunikaci se SAMem-04, příklad je opět ke stažení v ZIPu **Samovy kuchařky** na našem webu a vidíte jej také na obrázku 76. Pozor, tento projekt je napsán pro SAMa s linkou RS-232, pro RS-485 musíte upravit parametr COMnr modulu SERIALCOMM.

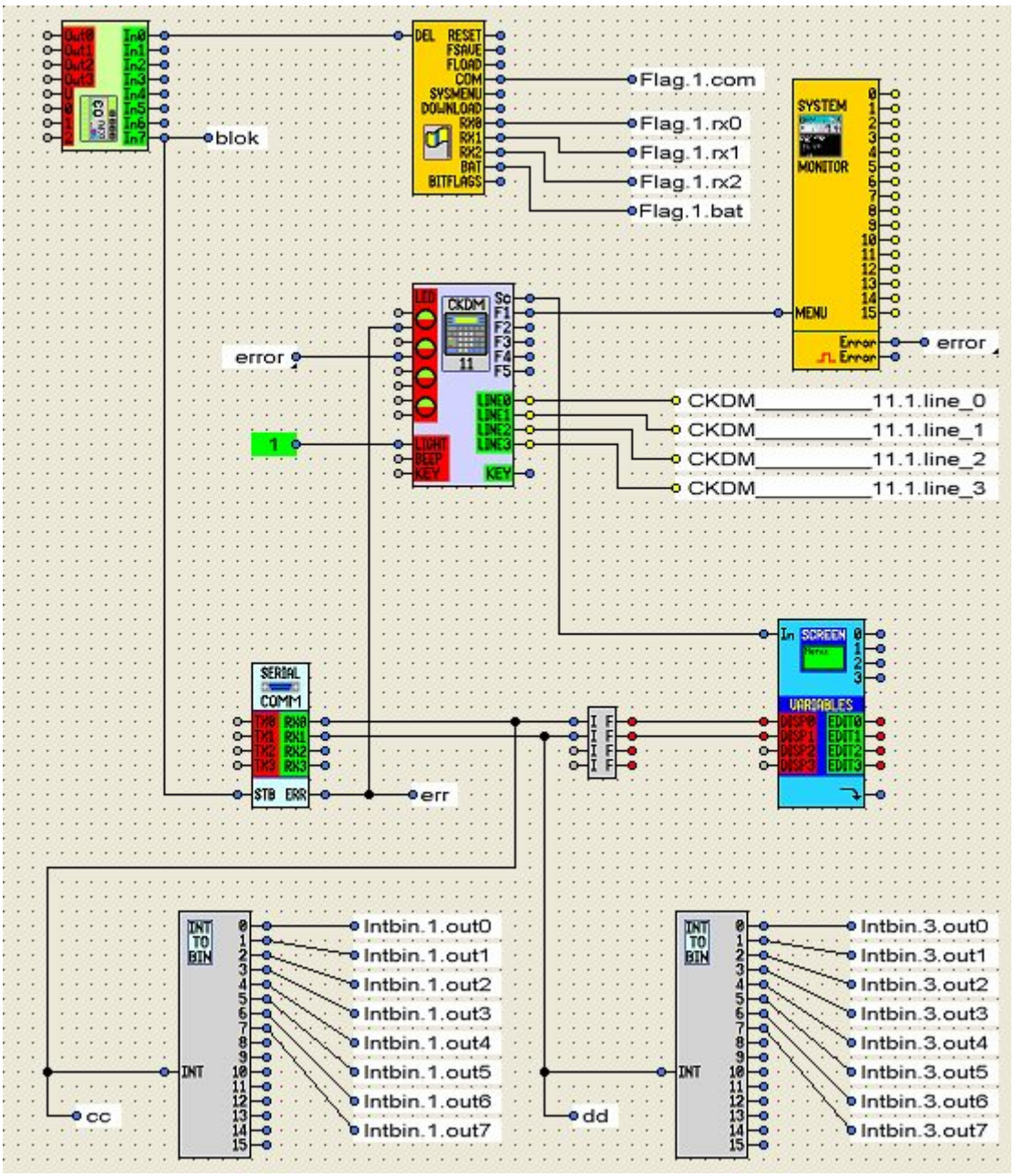
Všimněte si v něm i několika ladících prostředků.

Blokace komunikace je prováděna signálem **blok** z Inu 7 centrály CCPU-03.

Pro ověření, zda je funkční příjem na COMu centrály slouží modul FLAG. V ladícím režimu FREDa můžete tak sledovat na výstupech RX, zda váš COM centrály něco přijímá.

Dále v RUNu vidíte za modulem SERIALCOMM přijaté bajty **cc** a **dd** a za moduly INTtoBIN jejich převod na binární signály - tak získáte zpět logické stavy jednotlivých čtených signálů modulem SAM-04.

V případě, že vás váš projekt neposlouchá je opět vhodné do komunikační linky zařadit odposlech a pomocí programu TERMINAL tak zjistit, jak skutečně relace vypadají a podle toho dohledávat chybu.



Obr. 76 Projekt s modulem SAM-04

7 Interiérový teploměr RAI-01 s komunikací RS-485

7.1 Konfigurace RAI-01

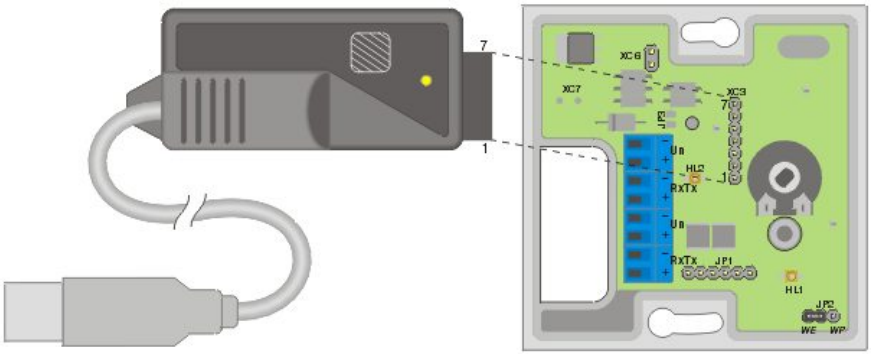
7.1.1 Úvod

Inteligentní interiérové teploměry **RAI-01** se vyrábí v provedení **A,** tj. s ovládacím kolečkem, které může sloužit pro nastavování korekce, hodnoty ap., a v provedení **B** - bez tohoto nastavování. Napájení RAI-01 může být v rozsahu 10 - 30 V DC.

Popis teploměru, blokové schéma, technické parametry a popis komunikace protokolem **Epsnet** naleznete v manuálu, který je ke stažení na našich stránkách; odkaz naleznete přímo na stránce výrobku RAI-01. Nás nyní zajímá komunikace s teploměrem a první čeho si všimneme je to, že se již nejedná o ASCII protokol SAM, ale o protokol Epsnet. Společné s předchozími přílady s moduly SAM je však práce s firemním programem TESTER pro konfiguraci. Odlišná pak bude až práce s teploměrem RAI-01 v aplikačním programu ve FREDovi - ale o tom později.



Obr. 77 Foto teploměru RAI-01B a RAI-01A



Obr. 78 Připojení programovacího kabelu USI-11 k teploměru RAI-01

Konfiguraci RAIčka provádíme tedy opět pomocí firemního programu **TESTER,** ale tentokrát budeme navíc potřebovat programovací kabel USI-11 (což je převodník USB - sériová linka TTL). Připojení kabelu USB-11 k teploměru RAI-01 je vidět na obrázku 78; při konfiguraci není nutné RAI-01 napájet, plně postačuje napájení z USB. Opačný konec kabelu USI-11 zasuneme do konektoru USB počítače.

7.1.2 USB ovladač

Doposud jsme výrobky (XDM, SAMy) připojovali k PC (kvůli možnosti konfigurace programem TESTER) většinou přes převodník RS-485 / RS-232, případně pokud PC nemělo standardní COM typu RS-232, ale USBéčko, vkládali jsme mezi zmíněný převodník RS-485/RS-232 ještě převodník RS-232/USB. Tento USBéčkový převodník jste používali již dříve a věděli jste, že v programu TESTER se musíte odkazovat na COM, který byl tomuto USBéčku přidělen systémem po instalaci příslušného USB ovladače (od výrobce převodníku).

Nyní nastává obdobný případ s programovacím kabelem USI-11. Na stránce výrobku RAI-01 na webu, se dozvíte, že pro naše převodníky s rozhraním USB používáme obvody FTDI a k nim máme na stránkách ke stažení ovladač.

Proto si stáhněte soubor **usb.zip**, ten rozbalte a řiďte se podrobnými pokyny v souboru **usb.txt** pro instalaci ovladače.

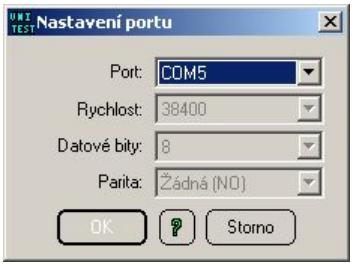
Je také možné, že jste již dříve používali přímo převodník USB / RS-485 pro připojení SAMů k PC. Pokud to byl převodník naší firmy, tj, s obvodem FTDI, pak nemusíte pro programovací kabel USI-11 tento USB ovladač instalovat, protože jste instalaci zřejmě provedli již dříve.

Po instalaci USB ovladače zjistíme pomocí Správce zařízení, který COM PC byl USBéčku přiřazen. Tak jako v kap.1 - viz obr. 3.

7.1.3 INFO z TESTERu

Předpokládejme, že USB ovladač máme nainstalován správně, zapojme programovací kabel USI-11 k teploměru RAI-01 podle obrázku 78 a spusťme program TESTER. Takjako dříve musíme nejprve v okně nabídky výrobků vybrat - tentokrát vybereme **RAI - tester inteligentních teploměrů RAI** (ne ten řádek pod tímto s popisem RS-485!). Viz obrázek 80.

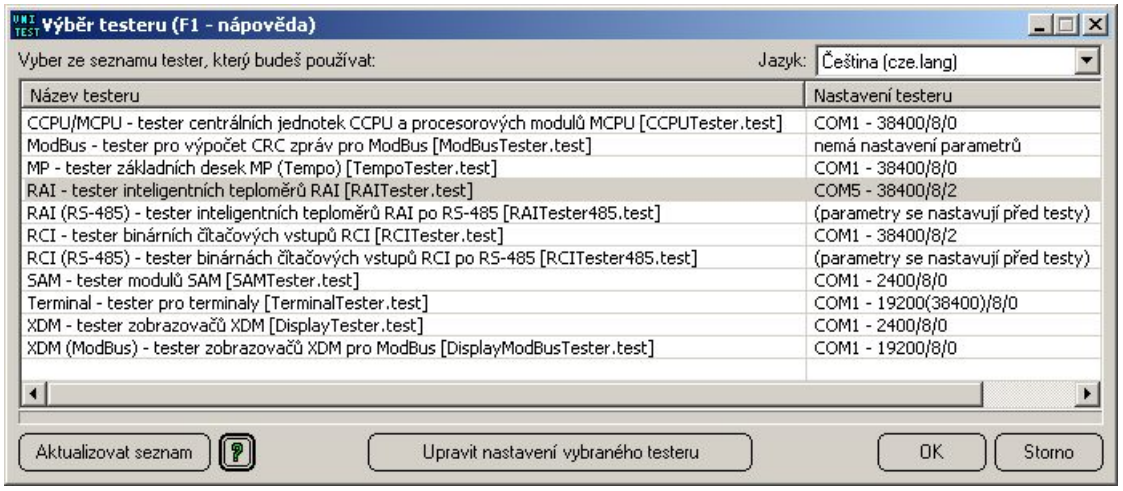
Zkontrolujeme, případně nastavíme, port / COM PC kliknutím na tlačítko **Upravit nastavení vybraného testeru** - viz obr. 79.



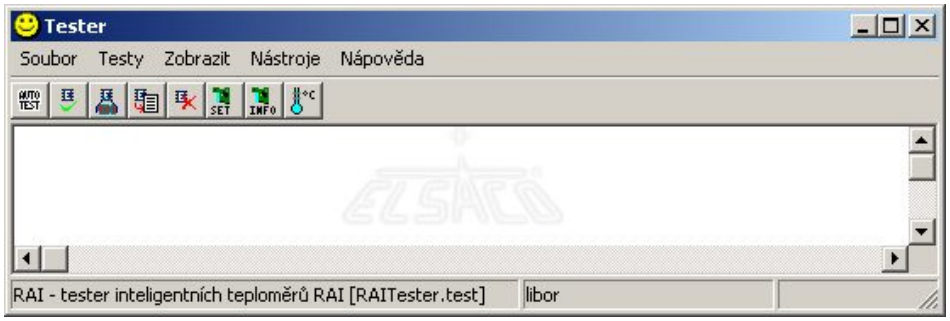
Obr. 79

Schválíme nastavení tlačítkem OK, další OK v okně pro Výběr testeru a dostáváme so do hlavního okna TESTERu - viz obrázek 81.

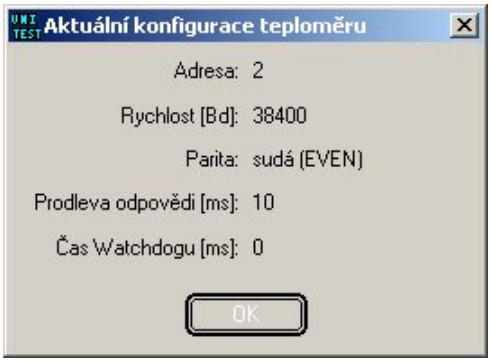
V nabídce ikon nalezněte volbu **INFO** nebo volte z nabídky **TESTY** / **Zobrazení aktuální konfigurace F8** a vyčkejte až se zobrazí dia-



Obr. 80 TESTER - výběr výrobku



Obr. 81 Hlavní okno TESTERu



Obr. 82 INFO o RAI-01

logové okno, ve kterém budou zobrazeny vyčtené informace z teploměru RAI-01. To vidíte na obrázku 82. Z výroby tam bude možná adresa 1. Tak jsme získali infoprmace, jak je teploměr nastaven a hlavně víme, že s ním přes TESTER umíme komunikovat. Můžeme proto přikročit k nastavení jeho adresy, případně komunikační rychlosti.

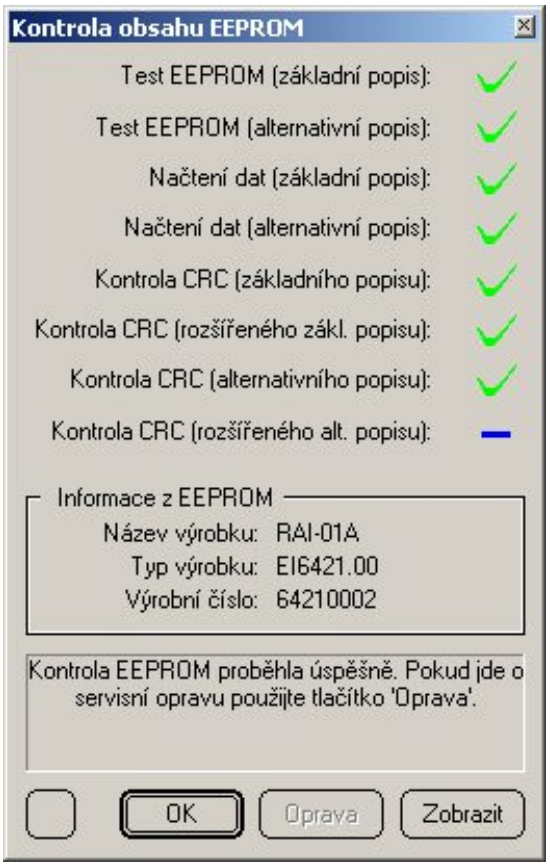
Během práce s TESTERem jste si možná všimli, že v hlavním okně TESTERu jsou vypisovány infomace o tom, co jsme právě provedli. Např.

```
Zobrazuji dialog s aktuální konfigurací
ukončeno uživatelem
```

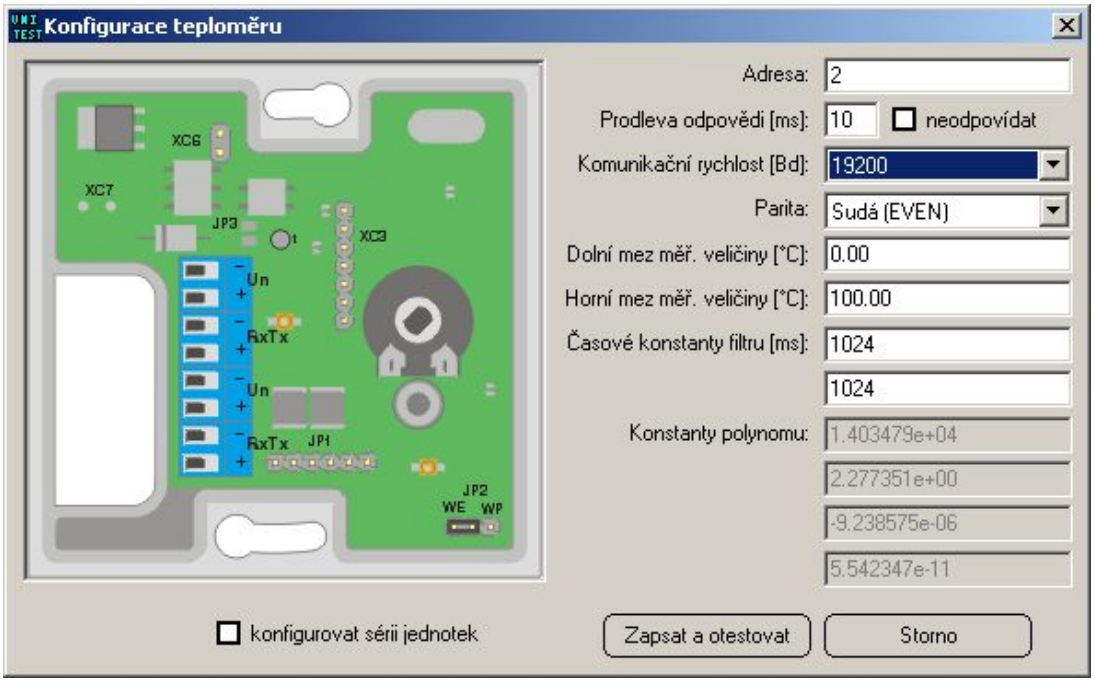
7.1.4 SET v TESTERu

Pro změnu konfigurace teploměru slouží volba pomocí ikonky **SET** nebo z řádku nabídek pak **TESTY** / **Konfigurace teploměru F7**. Chtějme nastavit adresu 2 a rychlost 19 200 Bd. Po kliknutí na SET je nejprve provedena kontrola obsahu eeprom v teploměru - viz obrázek 83. Proběhne v pořádku, schválíme tlačítkem OK.

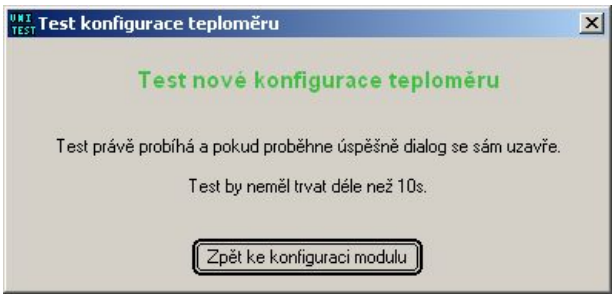
Teprve pak se dostáváme do okna pro konfiguraci parametrů teploměru - viz obrázek 85. Zeditujeme vybrané položky a schválíme tlačítkem **Zapsat a otestovat**.



Obr. 83 Kontrola obsahu EEPROM



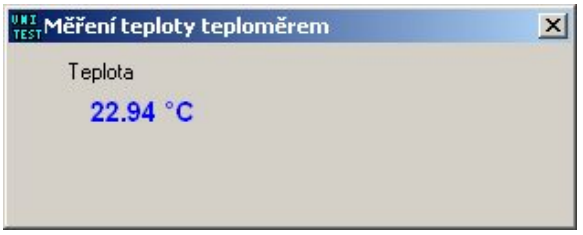
Obr. 85 Okno pro konfiguraci parametrů teploměru RAI-01



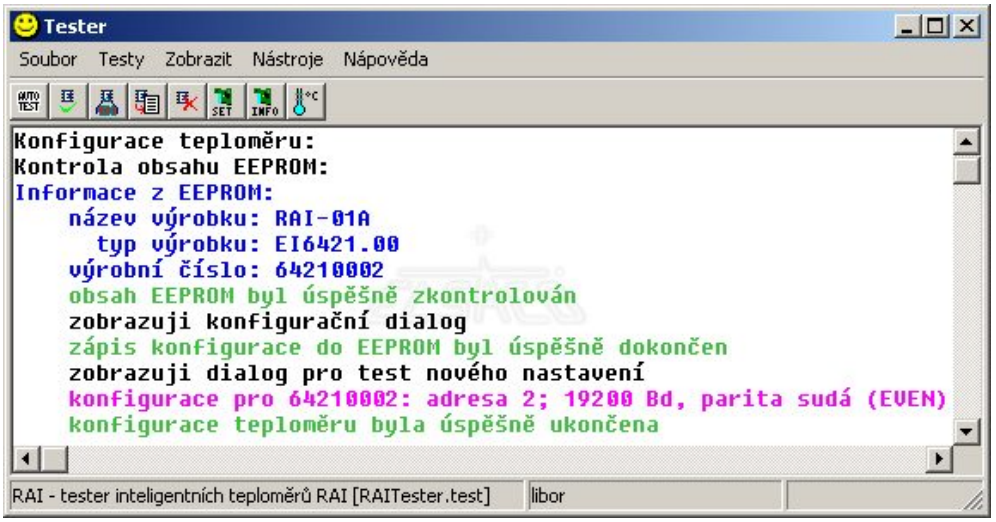
Obr. 84 Test nové konfigurace teploměru

Potom probíhá test nové konfigurace, během kterého jsme o tomto informování oknem dle obrázku 84. Pokud necháme test doběhnout, okno samovolně zmizí. V hlavním okně TESTERu si pak můžeme přečíst informace o právě dokončené konfiguraci teploměru - viz obrázek 86.

Tak máme teploměr připraven pro další pokusy ve FREDovi s adresou 2 a komunikační rychlostí 19 200 Bd. Ještě si můžeme ověřit, zda teploměr skutečně měří teplotu, a to pomocí volby **TESTY** / **Měření teploměrem F9** nebo kliknutím na ikonku s teploměrem. V tomto případě je lepší teploměr napájet, protože při napájení přes USB dostaneme značně zkreslené údaje. V novém okénku pak vidíme jak teploměr měří - viz obrázek 87. Tím naše pokusy s TESTERem a teplomě-



Obr. 87



Obr. 86 Zprávy v hlavním okně TESTERu o konfiguraci teploměru

rem mohou skončit a můžeme pokračovat problematikou komunikace s teploměrem protokolem Epsnet.

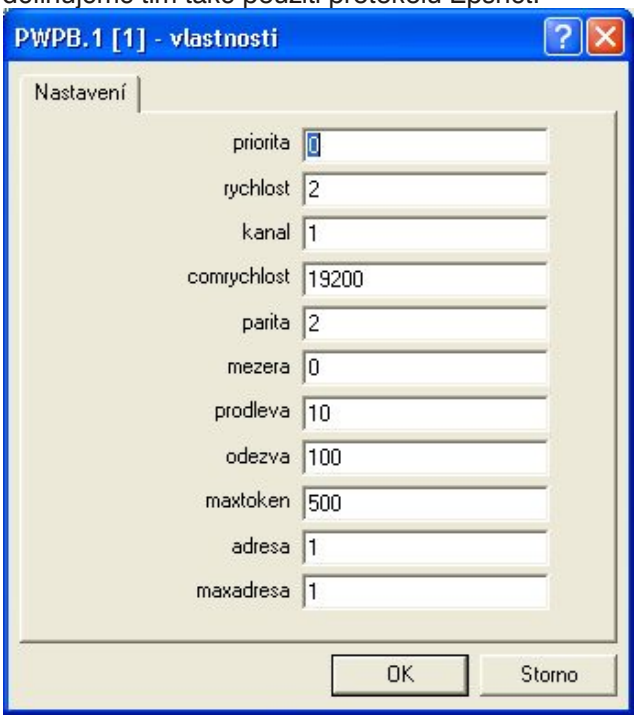
7.2 RAI-01 ve FREDovi

Jak již bylo naznačeno, komunikace bude probíhat protokolem Epsnet. K tomu potřebujeme podrobné informace o tom, kde se potřebné údaje v protokolu nachází. Zřejmě budeme ve FREDovi využívat tzv. MPC komunikaci (pro Epsnet, zjednodušený Profi-Bus). Sáhneme po manuálu pro RAI-01 a z něj vyčteme na str. 3 a 4, že procesní data jsou uložena jako reálná čísla ve čtyřbajtovém formátu **v bloku 2 s offsetem 0 pro teplotu a s offsetem 4 pro údaj polohy korekčního kolečka**.

Ostatní definice nás nezajímají - ty využívá TESTER a ty nemusíme umět.

Chceme jen vlastně vyčítat teplotu a polohu kolečka.

Teploměr budeme připojovat k centrále CCPU-21 na COM1 s GO. Proto musíme nejprve (krom povinných modulů) ve FREDovi položit na plochu modul **PWPB_MAIN**, pomocí kterého nadefinujeme pro COM1 RS-485 základní komunikační parametry a nadefinujeme tím také použití protokolu Epsnet.



Obr. 88 Definice modulu PWPB_MAIN

V HELPu můžeme o MPC komunikaci přečíst základní, v našem případě se bude jednat o napojení sériových modulů k centrále, která bude stanicí monomaster. Proto parametry modulu **PWPB_MAIN** nastavíme následovně (obr. 88):

```
priorita = 0
rychlost = 2
kanal = 1
comrychlost = 19200
parita = 2
mezera = 0
prodleva = 10
odezva = 100
maxtoken = 500
adresa = 1
maxadresa = 1
```

Protože chceme jen přijímat data, použijeme dále do projektu modul PWPB_RX.



Obr. 89 Definice pro modul PWPB_RX

V něm nadefinujeme parametry **blok, offset a delka** podle prvního odstavce této kapitoly (obr. 89):

```
priorita = 0
rychlost = 2
perioda = 5000
adresa = 2
blok = 2
offset = 0
delka = 8
```

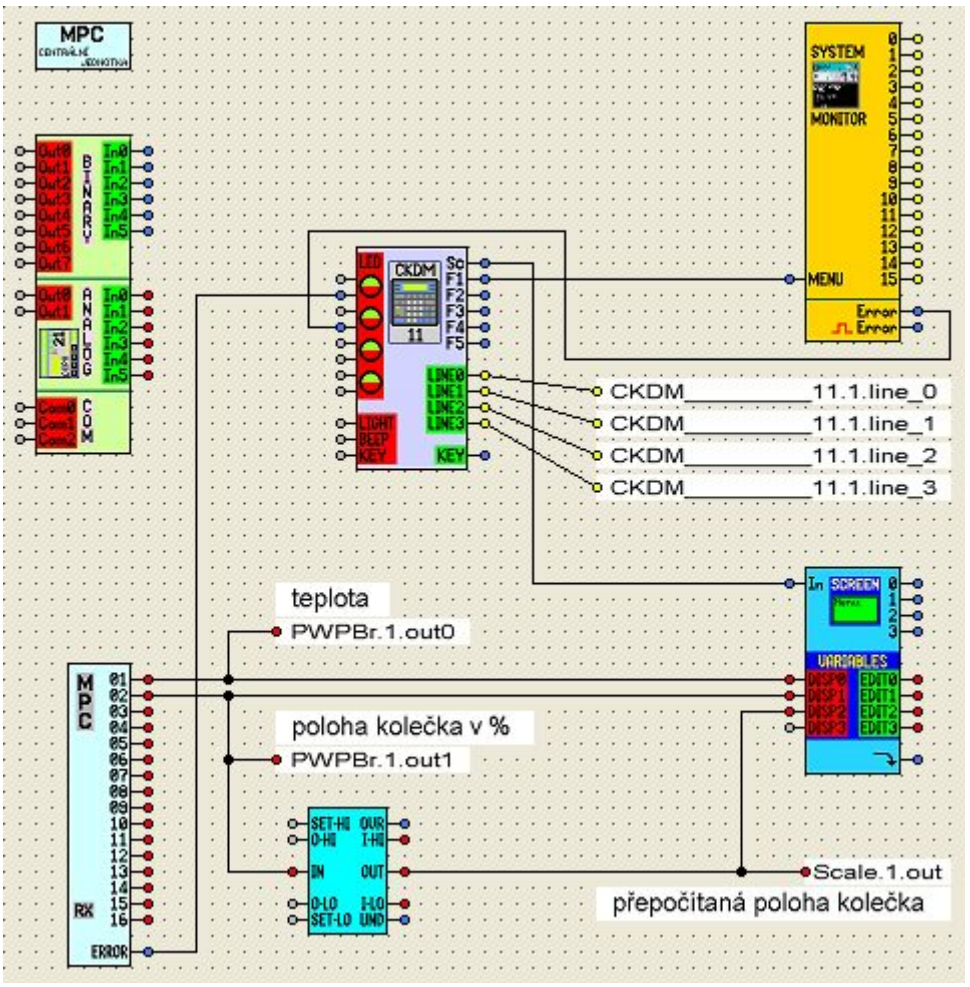
Adresu 2 jsme definovali TESTERem. Délka dat je 8 bajtů, 4 bajty teplota a 4 bajty poloha kolečka.

Projekt **RAI01.opf** je opět součástí ZIP balíčku Kuchařky a je volně ke stažení na našich stránkách *www.elsaco.cz*.

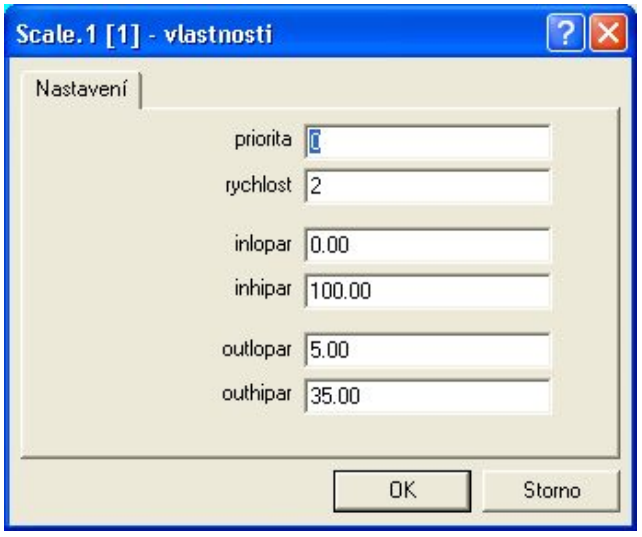
Projekt je na obrázku 90, v ladícím režimu ve FREDovi pak vidíte vyčítané hodnoty přímo na výstupních labelech modulu PWPB_RX. Protože polohu kolečka budeme chtít použít pro nastavování např. teploty v místnosti, použijeme pro přepočet modul SCALE a v něm nastavíme parametry pro přepočet rozsahu. Vstupní rozsah je 0 až 100, výstupní např. 5 až 35 (viz obr. 91).

Nastavená teplota v místnosti je pak po přepočtu vidět za modulem SCALE (v ladícím režimu v labelu *Scale.1.out*). Tak podobně pracujeme s vyčtenou polohou kolečka. Přepočtenou hodnotu pak nabídneme dál do programu, zrovna tak jako změřenou teplotu.

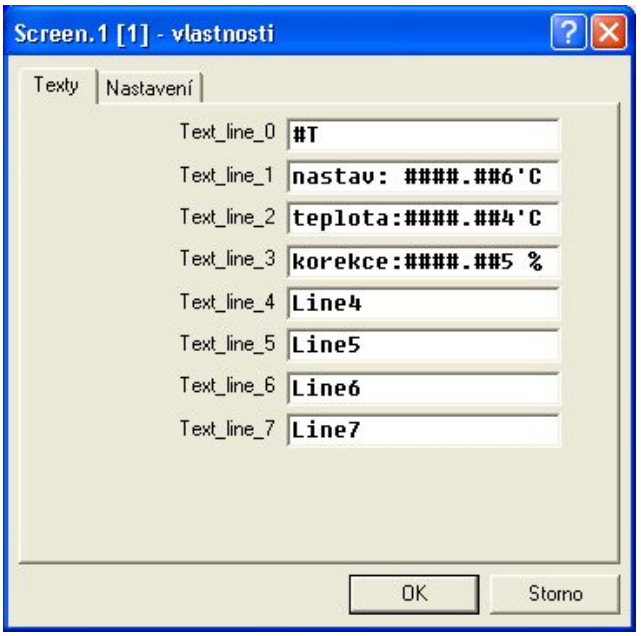
Zobrazení hodnot na displeji ovládacího panelu CKDM je definováno modulem SCREEN - viz obr. 92.



Obr. 90 projekt pro RAI-01



Obr. 91 Nastavení modulu SCALE



Obr. 92 Definice výpisů na displeji (SCREEN)

8 Čítač impulzů RCI-01 s komunikací RS-485

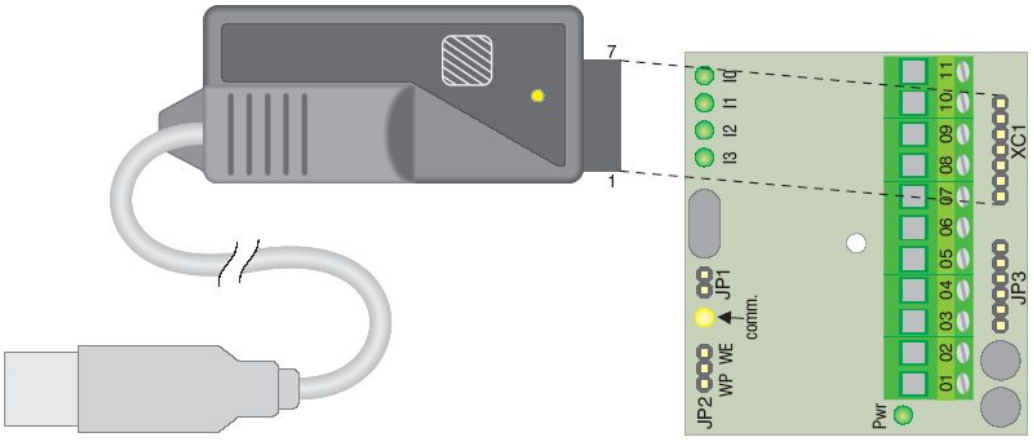
8.1 Konfigurace RCI-01

Ke konfiguraci RCIčka potřebujeme firemní program TESTER a programovací kabel USI-11. Popis konfigurace a testů RCIčka naleznete v HELPu TESTERu. Zde jen krátce.

Předpokládáme, že v PC máte nainstalován ovladač USB pro obvody FTDI, a proto lze připojit k PC programovací kabel USI-11 na port USB PC. Bližší viz kap. 6.1.2. Programovací kabel zasuneme do USB portu PC.



Obr. 93 Foto modulu RCI-01



Obr. 94 Připojení programovacího kabelu USI-11 k čítači RCI-01

Sundáme kryt RCIčka a vyjmeme tištěný spoj z krabičky. Stačí opatrně průchodkou zasunout šroubovák pod tištěný spoj a lehce páčit spoj směrem ode dna krabičky.

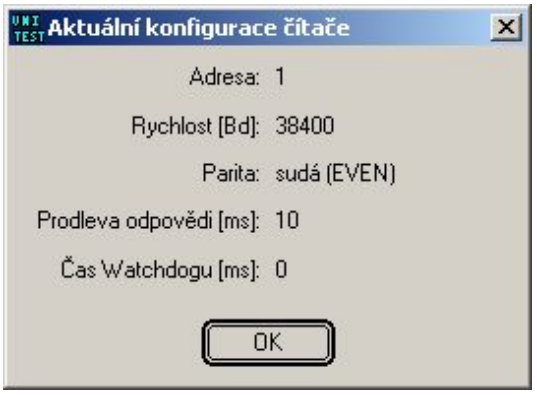
Programovací kabel zasuneme podle obrázku 94 do konektoru pro konfiguraci. Propojku EEPROM dáme do polohy WE, abychom umožnili zápis nové konfigurace do paměti EEPROM.

Spustíme program TESTER, vybereme test - řádek **RCI - tester binárních čítačových vstupů RCI**, případně volbou **Upravit nastavení vybraného testeru** zeditujeme **Nastavení portu** pro USBéčko PC, ve kterém máme zasunut programovací kabel. Obdoba kap. 6.1.3. Po schválení na tlačítko OK se dostaneme do hlavního okna TESTERu - to už známe z minulých pokusů.

Klikneme na ikonku **INFO** nebo volíme **Testy** / **Zobrazení aktuální konfigurace F8** a dostaneme okno s vyčtenými parametry z čítače RCI-01 - viz obrázek 95.

Tak máme ověřeno funkční napojení PC na RCIčko a můžeme přistoupit k nastavení pro aplikace. Chceme nastavit adresu 2 a komunikační rychlost 19 200 Bd.

Okno **Aktuální konfigurace čítače** zavřeme kliknutím na **OK** a volíme konfiguraci čítače kliknutím na ikonku **SET**.



Obr. 95 Info RCI-01, nastavení z výroby

Nejprve proběhne kontrola obsahu paměti EEPROM - viz obrázek 96 - pokračujeme schválením, klikneme na tlačítko **OK**.

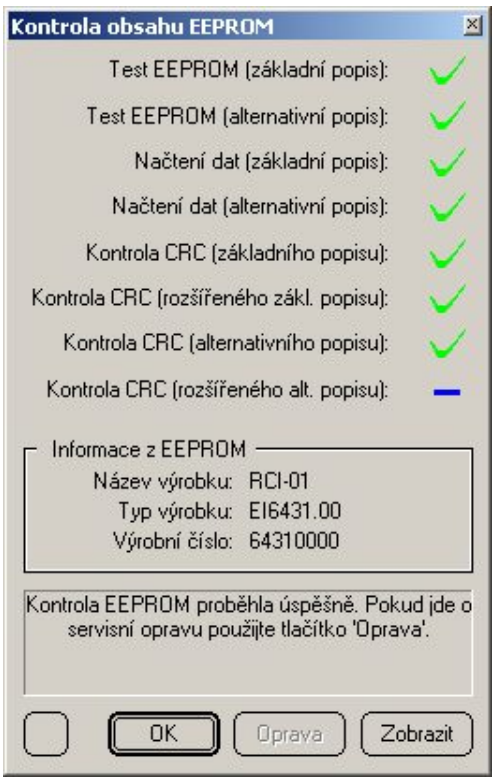
Teprve potom dostaneme dialogové okno pro nastavení parametrů čítače RCI-01.

V tomto okně zeditujeme adresu a komunikační rychlost na požadované hodnoty - viz obrázek 97.

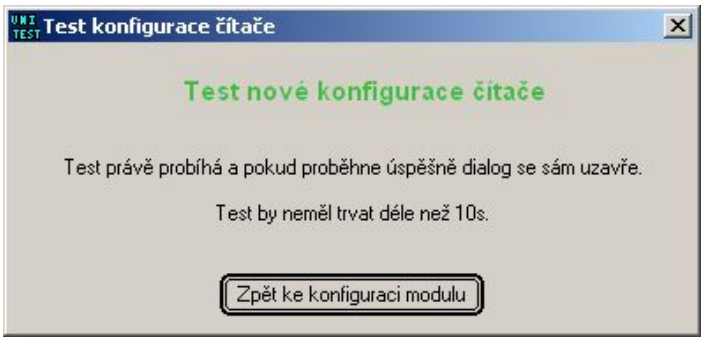
Zkontrolujeme, zda máme hodnoty správně vyplněny a schválíme tlačítkem **Zapsat a otestovat**.

Během testu nové konfigurace je zobrazeno dialogové okno dle obrázku 98. Po tetsuokno zmizí . konfigurace je hotova.

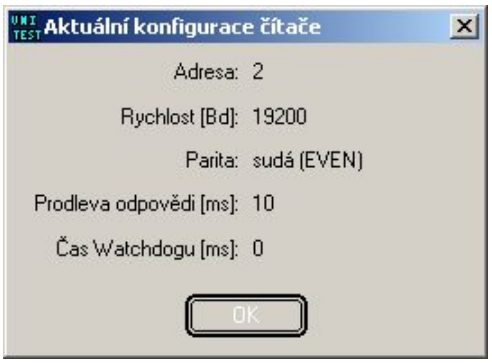
Průběh konfigurace je zapsán v hlavním okně TESTERu - viz obrázek 99.



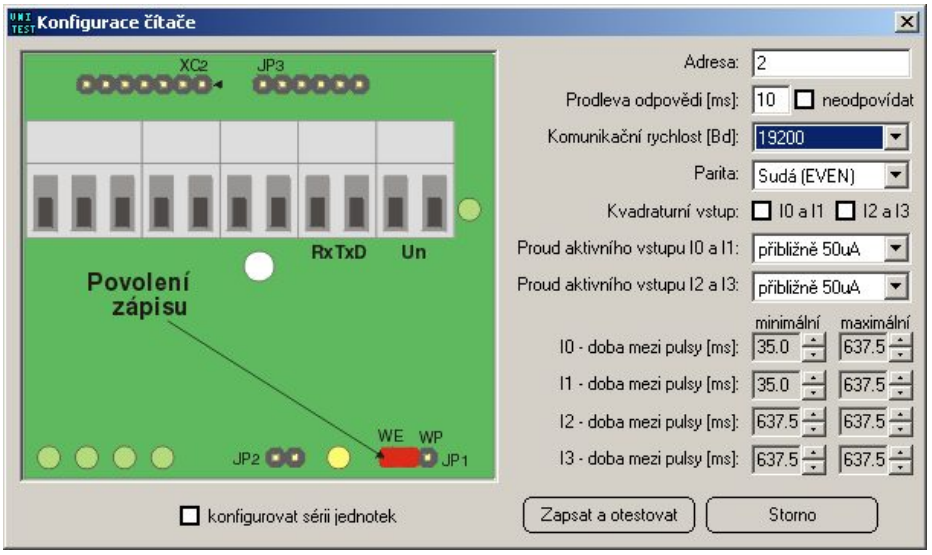
Obr. 96 Kontrola obsahu EEPROM



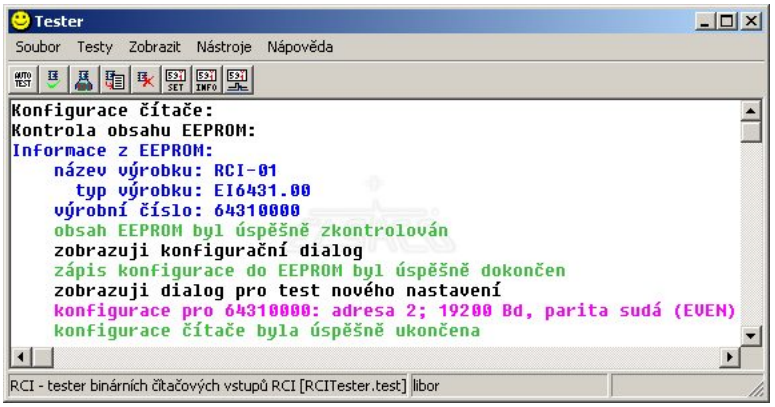
Obr. 98 Test nové konfigurace



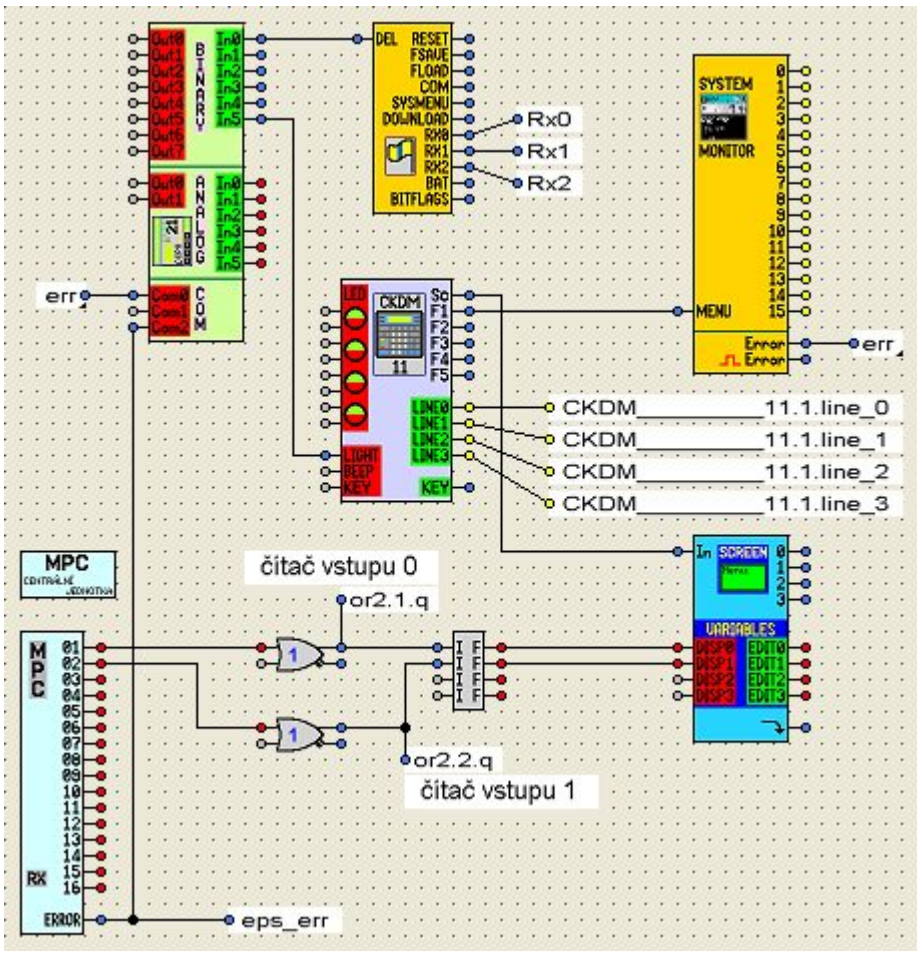
Obr. 100 Ověření konfigurace RCI-01



Obr. 97 Okno pro konfigurace RCI-01



Obr. 99 Průběh konfigurace RCI-01 v hlavním okně TESTERu



Obr. 101 Projekt pro čítač impulzů RCI-01

Její správnost lze ověřit znovu použitím volby INFO - viz obrázek 100. Je-li vše v pořádku nastaveno, nezapomeneme vrátit propojku do polohy WP, abychom zakázali změnu konfigurace zápisem do paměti EEPROM.

8.2 RCI-01 a FRED

Komunikace bude probíhat protokolem Epsnet. Ve FREDovi budeme definovat tzv. MPC komunikaci (pro Epsnet, zjednodušený ProfiBus). Sáhneme po manuálu pro RCI-01 a z něj vyčteme na str. 4, že procesní data, která nás nyní zajímají, jsou uložena celočíselně ve čtyřbajtovém formátu (long integer) **v bloku 2 a 3 s offsetem 6 pro čítač vstupu 0**.

Ostatní definice nás nezajímají - ty využívá TESTER a ty nemusíme umět. Chceme jen vyčítat hodnoty dvou čítačů, tj. 2x4 bajty.

RCI-01 připojíme k centrále CCPU-21 na COM1 s GO. Proto musíme nejprve (krom povinných modulů) ve FREDovi položit na plochu listu schématu modul **PWPB_MAIN**, pomocí kterého nadefinujeme pro COM1 / RS-485 základní komunikační parametry a nadefinujeme tím také použití protokolu Epsnet.

V HELPu PL2 je popis MPC komunikace, v našem případě se jedná o napojení sériových modulů k centrále, která bude stanicí monomaster. Proto parametry modulu **PWPB_MAIN** nastavíme následovně:

```
priorita = 0
rychlost = 2
kanal = 1
comrychlost = 19200
parita = 2
mezera = 0
prodleva = 10
odezva = 100
maxtoken = 500
adresa = 1
maxadresa = 1
```

Protože chceme jen přijímat data, použijeme dále do projektu modul PWPB_RX.

V něm nadefinujeme parametry **blok, offset a delka** podle prvního odstavce této kapitoly:

```
priorita = 0
rychlost = 2
perioda = 5000
adresa = 2
blok = 2
offset = 6
delka = 8
```

Adresu 2 jsme definovali TESTERem. Délka dat je 8 bajtů, 4 bajty pro hodnotu čítače impulzů vstupu 0 a 4 bajty pro vstup 1.

Projekt **RCI01.opf** (obr. 101) je opět součástí ZIP balíčku Kuchařky a je volně ke stažení na našich stránkách *www.elsaco.cz*. Pro ladění projektu je opět vhodný modul FLAG a jeho výstupy RX0..2, na kterých je v RUNu vidět, zda centrála přijímá relace a na kterém kanálu. COM0 je určen jako hlavní kanál, na něm probíhají relace ladícího (RUN) režimu, na kanálu 1 bychom měli vidět příjem MPC relací. A je-li vše OK, pak na výstupu Error modulu PWPB_RX musí být 0.

9 CCPU-03 – přenos dat mezi centrálami po lince RS-485

9.1 Propojení centrál

V praxi se vám může stát, že budete potřebovat přenést např. změřené teploty z jedné centrály do druhé. Obecně - přenést data... K tomu lze využít sériovou komunikační linku RS-485/RS-422, která je u centrály CCPU-03 vyvedena na konektor RJ-45 a je označena jako COM1. Tato linka je včetně galvanického oddělení a včetně vestavěného měniče napájení galvanicky oddělené strany.

Pokud nemáte zkušenosti s komunikací mezi centrálami, doporučujeme vyzkoušet si nejprve "na stole" jak vlastní fyzické propojení, tak i naprogramování jednoduché aplikace, např. podle této kapitoly.

CCPU-03 / COM1/ RS-422/485	
pin	*signál*
1	+360R
2	TxD+
3	TxD-
4	GND
5	GND
6	RxD-
7	RxD+
8	-360R

K propojení centrál (kanály COM1) lze použít osmižilový kabel. Jednu stranu kabelu "zaříznete" do konektoru RJ45, na druhé straně kabelu po odstranění obalové izolace kabelu odštípnete oba krajní vodiče a "zaříznete" opět do konektoru RJ45 tak, aby zapojení pinů bylo "zrcadlově", tj. pin 2 na pin 7 (TxD+ na RxD+), pin 3 na pin 6 (TxD- na RxD-) a pin 4 na pin 5 (GND na GND). Krajní piny 1 a 8 propojeny nejsou, protože jsme na jedné straně kabelu odštípli vodiče - nejsou tedy zapojeny do konektroru. Tak celkem rychle získáte "testovací" propojovací kabel pro pokusy "na stole". Hotový kabel pro jistotu prozvoníme.

Osmižilový kabel s konektory RJ45, křížené propojení, lze objednat i v naší firmě, a to pod označením EI9272.xxx, viz Katalog 2006 na str. 7-12.

Jak dál? Centrály máme propojeny, další bude záležitostí aplikačního programu ve FREDovi. Pro komunikaci mezi centrálami slouží tzv. MPC komunikace.

9.2 MPC komunikace; help

Multi Point Communication, tj. vícebodová (mezicentrálová) komunikace EPSNET / ProfiBus stanic PL2.

Každá stanice může být řídící (MASTER) nebo podřízená (SLAVE).

Stanice MASTER aktivně vysílá data nebo žádá o data.

Stanice SLAVE pouze odpovidá na dotazy. Dotaz může být jak žádost o čtení dat ze SLAVE stanice, tak žádost o uložení dat do SLAVE stanice.

Společné komunikační parametry se definují v modulu PWPB_MAIN.

Maximální délka jednoho bloku přenášených dat je 64 bajtů (lze definovat parametrem DELKA, 64 bajtů právě odpovídá množství dat zpřístupňovaných moduly PWPB_RX a PWPB_TX). V případě potřeby komunikace s jiným zařízením než s centrálou CCPU-02/03/21 (například se sériovými periferními jednotkami SBI, SBO, SBIO nebo SAIO), které může být do sítě připojeno jako SLAVE zařízení, je možné nastavit délku bloku přenášených dat i menší, podle specifikace komunikačních dat připojeného zařízení.

Data v síti Epsnet jsou definována číslem bloku, který má maximální délku 64 kB.

Parametrem OFFSET se dále definuje posun komunikovaných dat v daném bloku.

Definice jedněch přenášených dat na přijímací a vysílací straně si musí odpovídat (stejné číslo bloku a stejný offset v bloku). Jiná přenášená data (jiné veličiny) naopak musí mí definici (určení) jinou, aby se nepřekrývaly už s jinde definovanými daty a nedocházelo k jejich vzájemnému přepisování.

Sama vysílat umí pouze stanice MASTER (tedy stanice, u které je parametr MAXTOKEN >0).

Po spuštění komunikace zahajuje stanice MASTER vysílání přebráním tokenu (vysílacího oprávnění).

Pokud je stanic MASTER více, začíná ta s nejmenší adresou.

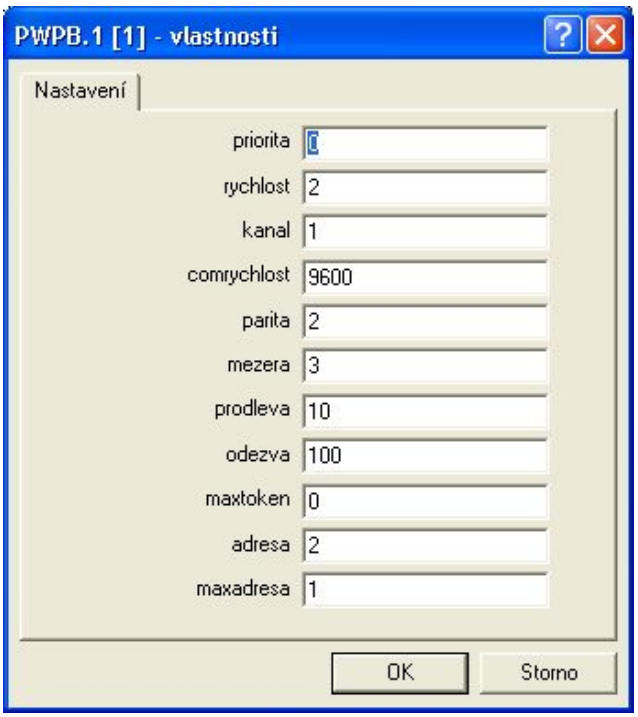
Prakticky je to po počáteční prodlevě ta stanice MASTER, jejíž hodnota výrazu ADRESA*ODEZVA (parametry stanice) je nejnižší. Aby tento systém fungoval v síti s více stanicemi MASTER, je třeba, aby měly všechny stanice MASTER parametr ODEZVA stejný.

TOKEN (vysílací oprávnění) si posílají stanice mezi sebou pomocí speciální krátké zprávy, a to postupně po adresách (modulo +1, tj. např. stanice s adresou 1 stanici s adresou 2, stanice 2 stanici 3, ...). Poslední takto obsluhovaná adresa je dána parametrem MAXADRESA v modulu PWPB_MAIN.

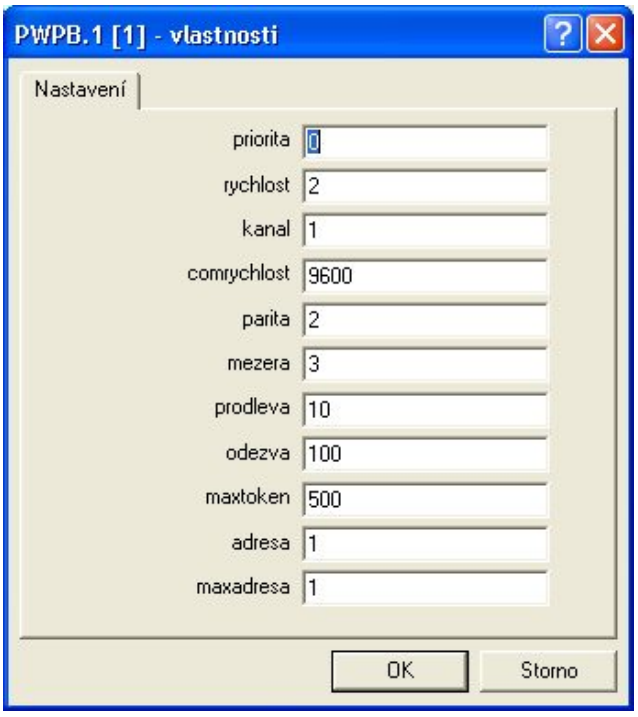
Stanice mající TOKEN vyřizuje po dobu trvání svého TOKENU komunikační požadavky podle definice svých modulů PWPB_RX a PWPB_TX.

Je tedy vhodné, aby doba trvání tokenu stanice (MAXTOKEN) byla alespoň tak dlouhá, aby byla stanice během jejího trvání schopna vyřídit všechny svoje komunikační požadavky.

Podle definice stanic master/slave v komunikační síti lze rozdělit síť na:



Obr. 102 Definice MPC pro stanici SLAVE



Obr. 103 Definice MPC pro stavici MASTER

- **multislave**
  Všechny stanice jsou podřízené, většinou dispečerskému PC, který je master.
  SLAVE odpovídá na požadavky dispečinku.
  Stanice slave musí mít nastaven parametr ODEZVA co nejdelší, nejlépe 30000.
- **monomaster**
  Jediná stanice MASTER, ostatní SLAVE.
  Síť stanic PL2, jedna z nich je řídící (tedy MASTER) - tato stanice vyřizuje centrálně všechny komunikační požadavky, tedy předávání dat mezi SLAVE stanicemi přímo není možné, pouze zprostředkovaně přenosem přes MASTER stanici.
  MASTER stanice by neměla mít nastaven parametr ODEZVA na dobu podstatně delší než na dobu nezbytně nutnou k dokončení příjmu odpovědi na komunikační požadavek, aby nebyla komunikace při náhodném výpadku zbytečně brzděna.
- **multimaster**
  Více stanic řídících.
  Takto lze nadefinovat, že několik stanic může aktivně vyřizovat komunikační požadavky.
  To umožňuje předávání dat nejen mezi libovolnou MASTER a SLAVE stanicí, a také mezi MASTER stanicemi mezi sebou navzájem. MASTER stanice obsluhuje jak požadavky na čtení, tak na zápis. Každá MASTER stanice si hlídá čerstvost svých komunikačních dat a v případě zjištěné prodlevy delší než polovina doby timeoutu (parametr PERIODA) pro daná data iniciuje komunikační relaci přenosu těchto dat.

9.3 MPC monomaster

V našem velmi jednoduchém případě zvolíme jednu centrálu jako MASTER a druhou jako SLAVE.

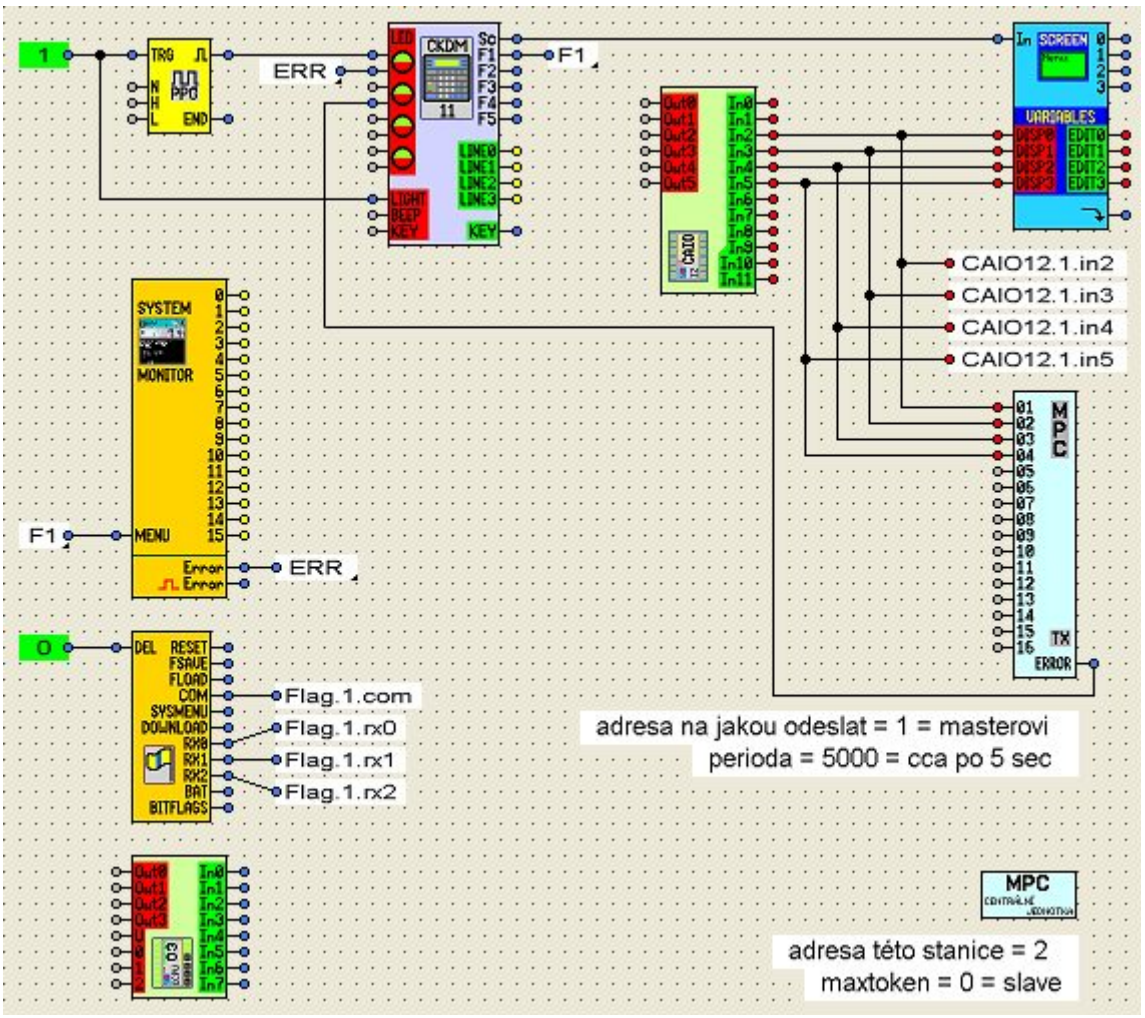
Stanici MASTER volíme s adresou 1 (pro MPC/Epsnet/ProfiBus), proto parametr MAXADRESA nastavujeme na 1 - TOKEN se předávat nemusí, zůstává na stanici MASTER. Stanici SLAVE pak volíme s adresou 2.

Společné parametry pro MPC komunikaci volíme v modulu PWPB_MAIN, a to v prvním projektu pro stanici SLAVE (obr. 102):

```
priorita = 0
rychlost = 2
kanal = 1
comrychlost = 9600
parita = 2
mezera = 3
prodleva = 10
odezva = 100
maxtoken = 0
adresa = 2
maxadresa = 1
```

a ve druhém projektu pro stanici MASTER (obr. 103):

```
priorita = 0
rychlost = 2
kanal = 1
comrychlost = 9600
parita = 2
mezera = 3
prodleva = 10
odezva = 100
maxtoken = 500
adresa = 1
maxadresa = 1
```



Obr. 104 Propojení centrál - SLAVE, projekt KUFR03PB.opf

Stanice MASTER má tedy na sběrnici s MPC adresu 1 a stanice SLAVE adresu 2; tyto adresy nesouvisí s adresami pro tzv. QQ protokol, které se nastavují na DIL přepínačích centrály nebo pomocí modulu SYSMON. Jedná se o adresy pro MPC protokol.

Zaměřme se nyní na stanici, která teploty měří a bude je předávat druhé centrále. Zvolme proto druhou centrálu jako MASTER - bude si o změřené údaje říkat požadavkem, a stanice s měřenými teplotami nechť je SLAVE. Stanici SLAVE doplníme modulem PWPB_TX, protože bude hodnoty vyslílat.

Parametry modulu PWPB_TX (obr. 105) definujme takto:

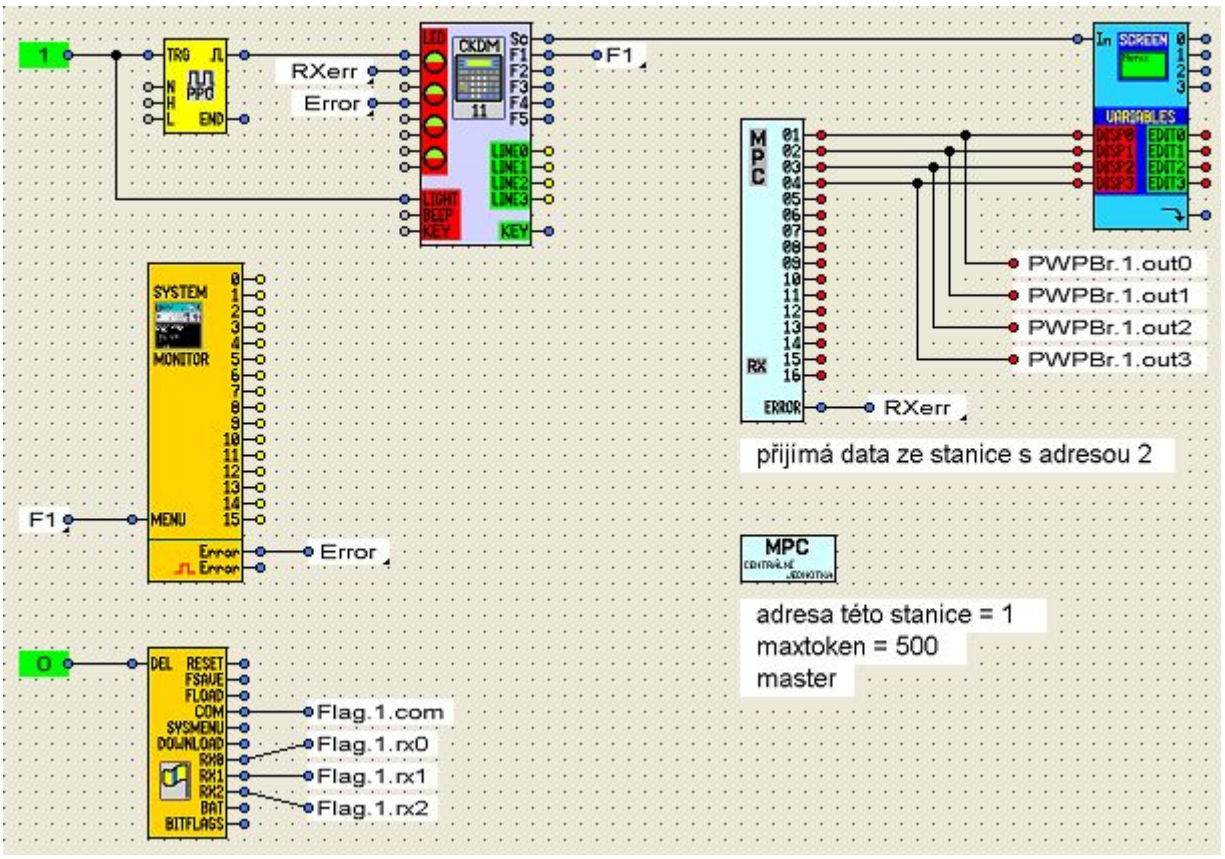
```
priorita = 0
rychlost = 2
perioda = 5000
adresa = 1
blok = 1
offset = 0
delka = 16
```

Délku dat 16 volíme podle počty přenášených hodnot (teplot). V testovacím projektu **KUFR03PB.opf** měříme a vysíláme 4 údaje po 4 bajtech (každá hodnota je přenášena ve 4 bajtech).



Obr. 105 Definice PWPB_TX pro SLAVE

Číslo bloku a offset volíme sami, nesmíme zapomenout, že se data nesmí překrývat a že moduly pro příjem a vysílání týchž dat musí mít čísla bloků a offset stejný.



Obr. 106 Propojení centrál - MASTER, projekt 03PBRX.opf

Poznámka:
Zmíněné platí pro propojení centrál, kdy si správu dat zajišťujeme sami.
Pokud bude např. centrála komunikovat s jednotkami SBI, SBIO, SBO, SAIO tímto protokolem (Epsnet/ProfiBus), zjistíme rozložení dat v jednotkách v jejich technickém manuálu. Tam např. čteme, že pro jednotky typu SAIO je blok 1 určen pro konfigurační data (definice mezí, filtrů apod.) a teprve blok 2 a 3 pro procesní data (naměřené filtrované hodnoty analogových vstupů, požadované hodnoty analogových výstupů).

V praxi při přenosu dat mezi centrálami to znamená, že pokud přenášíme maximálně 16 hodnot z jedné stanice, ponecháme číslo bloku 1 a offset 0. Délku ponecháme 64 (16x4=64 bajtů).

Teprve až je počet přenášených hodnot větší než 16, definujeme u odpovídajících modulů PWPB_TX a PWPB_RX (pro stejná data) např. číslo bloku 1, ale offset 64, protože v prvním bloku s offsetem 0 přenášíme prvních 16 hodnot.

Celková délka jednoho bloku dat může být maximálně 64 kB.

Další možností je, ponechat offset vždy 0 a zvýšit číslo bloku na 2, atd. Záleží, co je pro nás přehlednější.

V testovacím projektu (obr. 104) měříme 4 teploty modulem CAIO12 a naměřené hodnoty z pinů In2 až In5 zobrazujeme jednak pomocí modulu SCREEN na displeji ovládacího panelu CKDM-11, jednak hodnoty

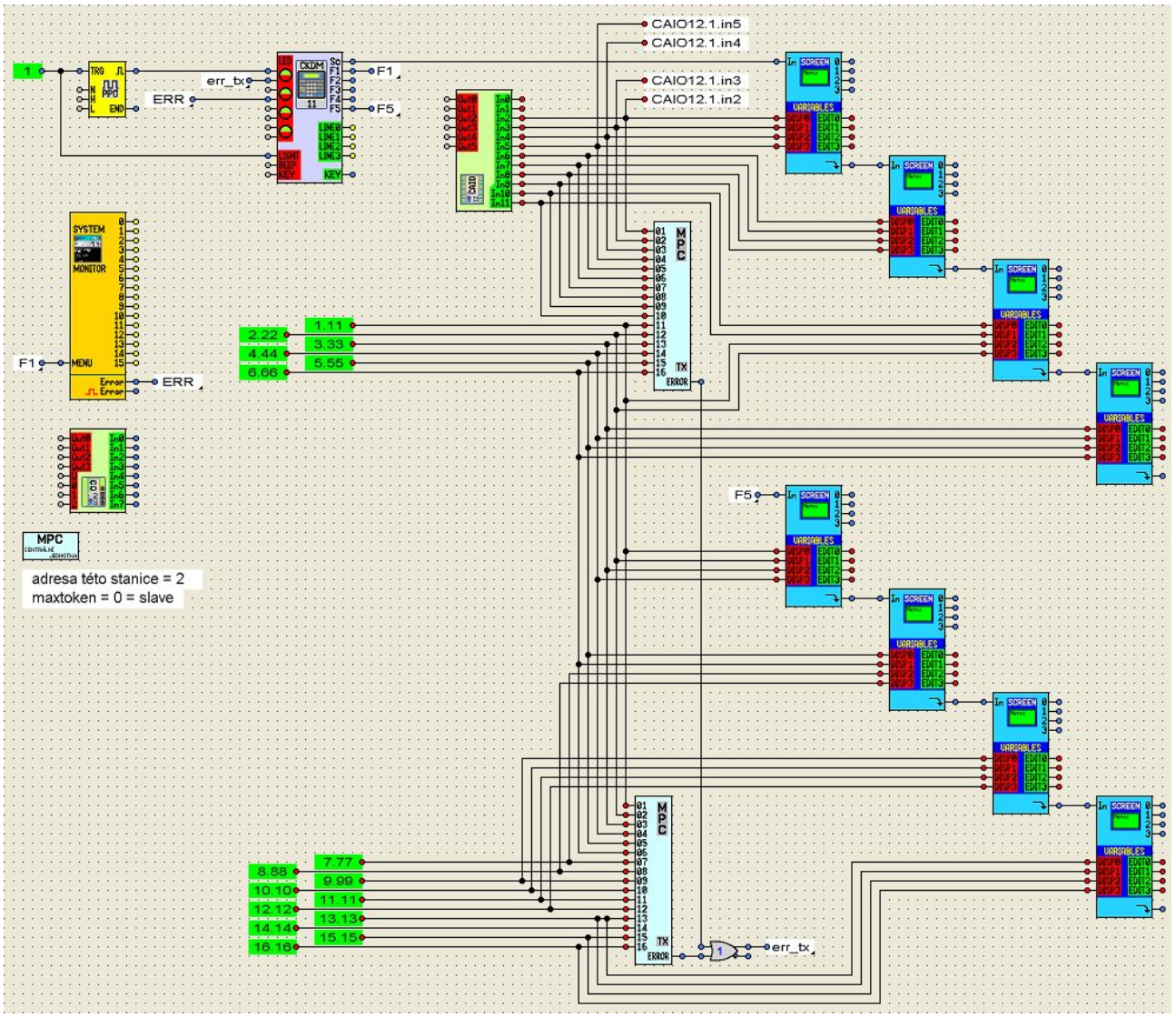


Obr. 107 Definice PWPB_RX pro stanici MASTER

z těchto pinů přivádíme na zmíněný modul PWPB_TX pro MPC komunikaci na druhou stanici.

Na výstupních labelech modulu FLAG můžeme pak v ladícím režimu FREDa sledovat, zda probíhají komunikační relace na kanále COM1. Viz obrázek 104. Projekt **KUFR03PB.opf** je ke stažení spolu s tímto manuálem v ZIP balíčku na našich stránkách, a to spolu s projektem **03PBRX.opf** do protější centrály.

Ve zmíněném druhém projektu pro centrálu (obr. 106), která se bude chovat jako MASTER, je obsažen modul PWPB_RX pro příjem 4 hodnot z předešlého projektu od centrály SLAVE.



Obr. 108 Propojení centrál 2 - SLAVE dává 32 hodnot, projekt 03PB_TX.opf

Parametry modulu PWPB_RX jsou nadefinovány takto (obr. 107):

```
priorita = 0
rychlost = 2
perioda = 5000
adresa = 2
blok = 1
offset = 0
delka = 16
initout1..16 = 0.0
```

Přijaté hodnoty lze sledovat v RUN režimu FREDa na výstupních labelech modulu PWPB_RX a na displeji.

Tak jsme docílili požadovaného. Jedna centrála měří 4 analogové hodnoty a po lince RS-485 protokolem Epsnet/ProfiBus (naše MPC) předává naměřené hodnoty druhé centrále.

Lze samozřejmě předávat jakékoliv hodnoty z projektu, třeba vypočtené apod.

Předání většího počtu hodnot si ukážeme v dalších dvou projektech.

9.4 MPC monomaster - přenos 32 dat

Původní projekty upravíme pro přenos většího počtu hodnot tak, abychom museli použít více než jeden modul PWPB_RX a PWPB_TX.

Projekt stanice SLAVE, která dává hodnoty pro přenos, doplníme dalším modulem PWPB_TX. Na všechny vstupy obou modulů PWPB_TX přivedeme signály, tj. zapojíme všechny jejich piny 01..16. Na první modul dopojíme analogové signály z modulu CAIO12, na zbytek přivedeme floatové konstanty. Obrázek projektu vidíte na obr. 108.

Oproti minulému projektu (toho, který nyní upravujeme) je k prvnímu modulu PWPB_TX připojeno všech 16 možných vstupů, tj. délka přenášených dat bude 16x4=64 bajtů. Proto parametry tohoto modulu budou nadefinovány takto:

```
priorita = 0
rychlost = 2
perioda = 5000
```

```
adresa = 1
blok = 1
offset = 0
delka = 64
```

Viz projekt **03PB_TX.opf** (obr. 108) v ZIP balíčku, ve kterém jsme oproti původnímu projektu **KUFR03-PB.opf** dále přidali pro zobrazování přenášených hodnot ještě řadu modulů SCREEN.

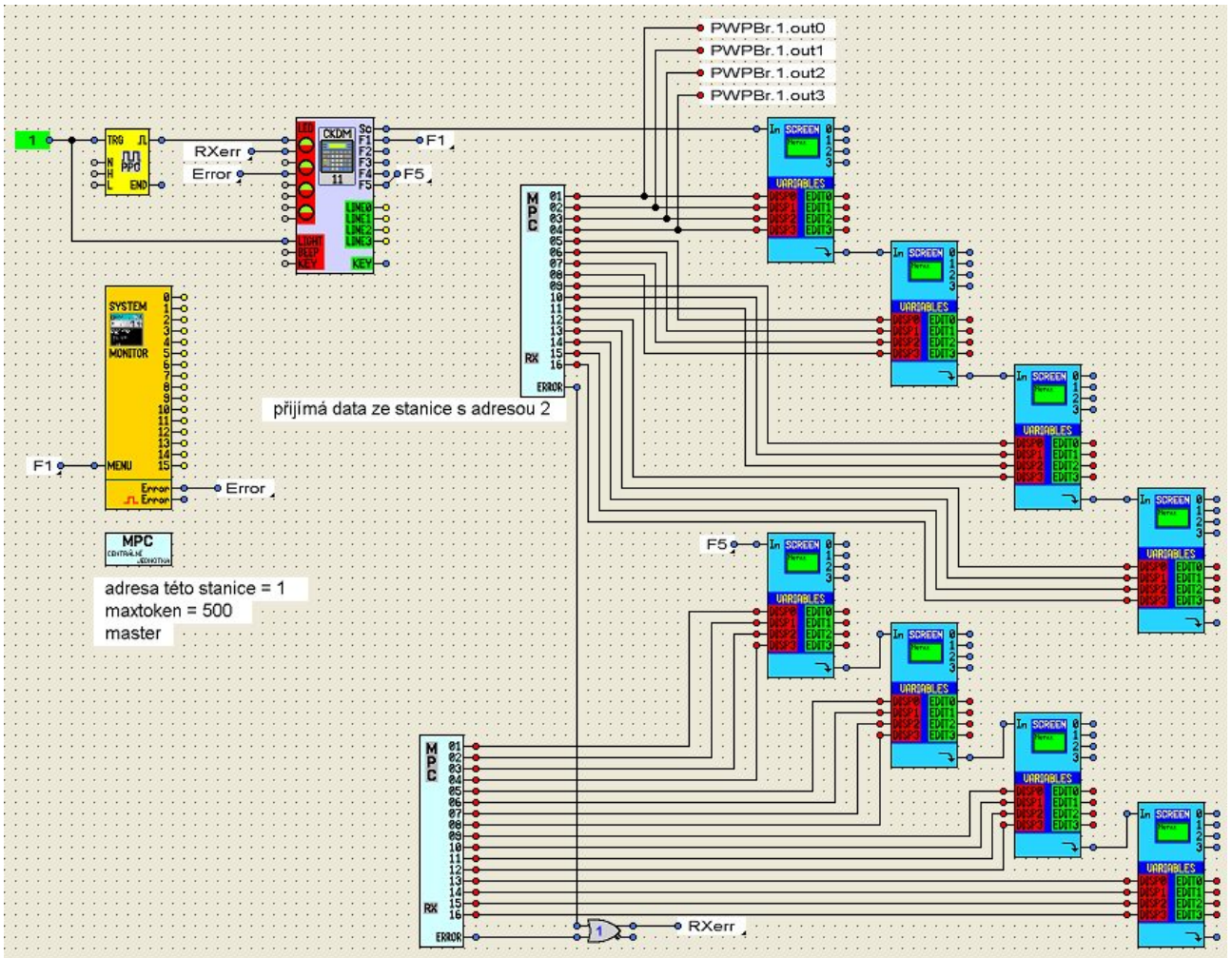
Modul PWPB_MAIN, definující společné vlastnosti MPC komunikace, ponecháváme beze změn:

```
priorita = 0
rychlost = 2
kanal = 1
comrychlost = 9600
parita = 2
mezera = 3
prodleva = 10
odezva = 100
maxtoken = 0
adresa = 2
maxadresa = 1
```

Druhý modul PWPB_TX bude mít parametry obdobné jako první, musíme však správně zařadit přenášená data do bloku dat. Protože jsme prve použili číslo bloku 1 s offsetem 0 a vímě, že jeden blok může obsahovat až 64 kB dat, budeme další data opět řadit do bloku číslo 1. Musíme však offset pro druhý modul PWPB_TX nastavit o 64 bajtů dat dál, tj. za data přenášená prvním modulem PWPB_TX. Délka dat tohoto bloku bude opět 64, protože máme zapojeno všech 16 vstupů/hodnot a jedn hodnota je přenášena ve 4 bajtech (16x4=64). Parametry druhého modulu PWPB_TX budou definovány takto:

```
priorita = 0
rychlost = 2
perioda = 5000
adresa = 1
blok = 1
offset = 64
delka = 64
```

Všechny hodnoty, které přenášíme do druhé centrály, můžeme zobrazit (pro kontrolu) pomocí kaskády modulů SCREEN (obr. 109).



Obr. 109 Propojení centrál 2 - stanice MASTER přijímá 32 hodnot, projekt 03PB_RX.opf

Chybová hlášení modulů PWPB_TX z jejich pinů Error sečteme a je-li hlášena chyba, dostaneme na výstupu součtového hradla "1", která ovládá horní červenou LEDku na panelu CKDM-11. Ta v normálu zeleně bliká díky zapojení PPG hradla.

Projekt pro SLAVE stanici máme nachystán, podívejme se nyní na upravený druhý projekt pro stanici MASTER.

Druhý projekt **03PB_RX.opf** (obr. 109) doplníme druhým modulem PWPB_RX a řadou modulů SCREEN pro zobrazení přijatých hodnot.

Protože přenášíme celkem 32 hodnot, a to dvojicí modulů pro přenos, jeden vždy pro 16 hodnot po 4 bajtech, bude opět parametr pro délku dat 16x4=64.

První modul PWPB_RX bude mít parametry:

```
priorita = 0
rychlost = 2
perioda = 5000
adresa = 2
blok = 1
offset = 0
delka = 64
initout1..16 = 0.0
```

A pro druhý modul PWPB_RX definujeme parametry:

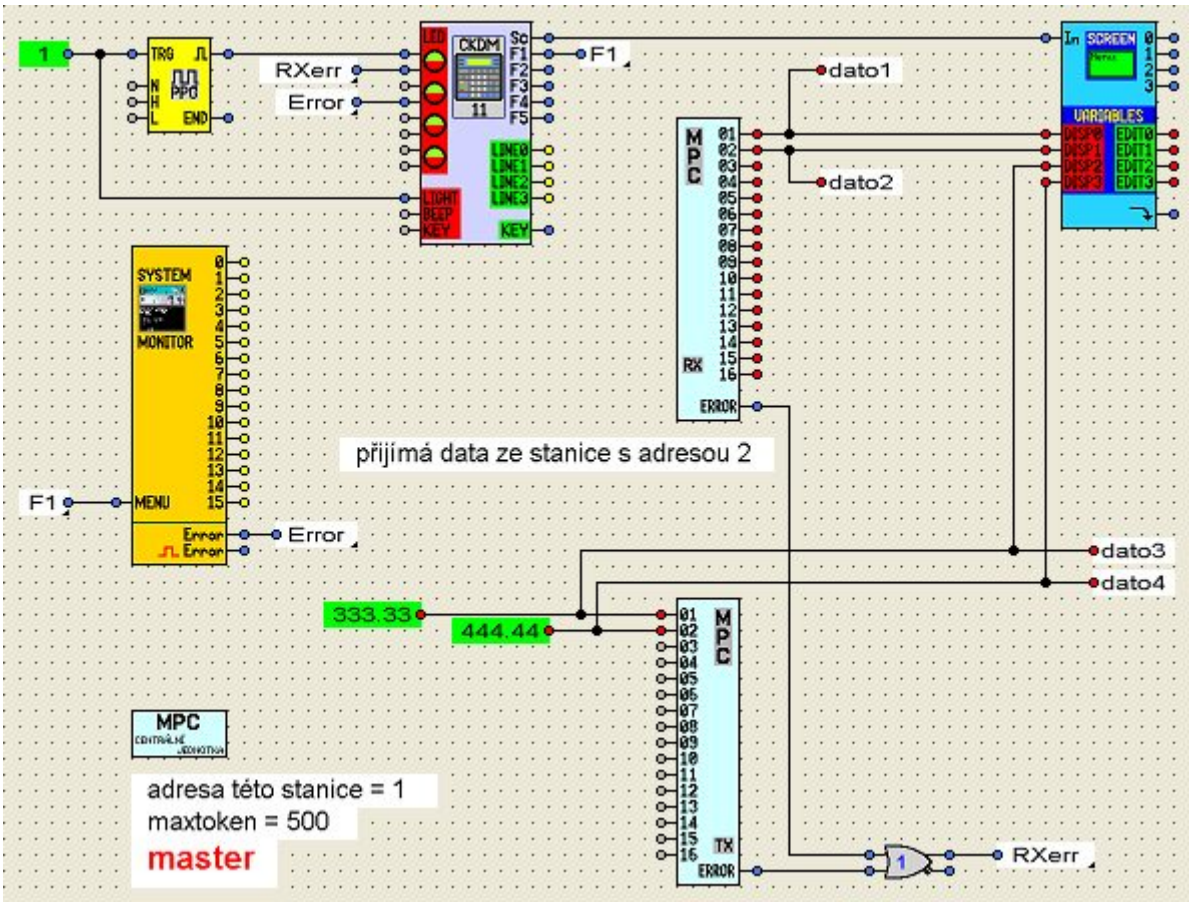
```
priorita = 0
rychlost = 2
perioda = 5000
adresa = 2
blok = 1
offset = 64
delka = 64
initout1..16 = 0.0
```

Parametr modulu PWPB_MAIN pro definici společných parametrů pro MPC zůstávají stejné:

```
priorita = 0
rychlost = 2
kanal = 1
comrychlost = 9600
parita = 2
mezera = 3
prodleva = 10
odezva = 100
maxtoken = 500
adresa = 1
maxadresa = 1
```

Oba projekty postupně přeložíme do jednotlivých centrál, které jsou propojeny zmíněným komunikačním kabelem (COM1).

Pak stačí na vysílací stanici SLAVE na displeji nalézt pro kontrolu zvolenou čtveřici vysílaných hodnot, na přijímací stanici nalistovat na displeji zobrazení příjmu stejné čtveřice hodnot a porovnat, zda data jsou stejná.



Obr. 110 Propojení centrál - MPC multimaster, stanice 1, projekt PB_adr1.opf

9.5 MPC multimaster

Mějme dále propojeny dvě centrály CCPU-03 na kanále COM1, tj. RS-422/485. Adresy centrál ponecháme 1 a 2, ale budeme chtít přenášet z každé centrály data na centrálu druhou. Zvolíme tentokrát obě centrály jako MASTER, obě budou dávat střídavě požadavky na přenos dat. Střídavé předávání požadavku je zabezpečeno předáváním tzv. TOKENu. Jedná se o režim MULTIMASTER a ten zabezpečíme v prvé řadě tím, že parametr MAXADRESA u obou centrál nadefinujeme na 2. Podle definice MPC bude zahajovat po zapnutí přenos stanice s nižší adresou (bude mít jako první TOKEN). Komunikaci zahájí tedy stanice s adresou 1, vyřídí své a předá TOKEN stanici s adresou 2, která pak v komunikaci pokračuje vyřizováním svých požadavků. A protože je MAXADRESA = 2, vrátí pak TOKEN stanici s adresou 1... a tak to pokračuje stále dokola.

Podívejme se nyní na projekt PB_ADR1.opf (obr. 110) pro stanici MASTER s adresou 1 pro MPC.

Nastavení parametrů modulu PWPB_MAIN:

```
priorita = 0
rychlost = 2
kanal = 1
comrychlost = 9600
parita = 2
mezera = 3
prodleva = 10
odezva = 300
maxtoken = 500
adresa = 1 [adresa této stanice]
maxadresa = 2
```

Vidíte, že MAXTOKEN > 0, tj. stanice bude MASTER a MAXADRESA = 2, tj. TOKEN se bude předávat postupně mezi stanicemi s adresami 1 a 2.

Modul PWPB_RX přijímá data D1 a D2 ze stanice s adresou 2, zvolme si tedy pro lepší pamatovatelnost, že data ze stanice s adresou 1 budeme řadit do bloku dat číslo 1 a data ze stanice s adresou 2 budeme řadit do bloku dat číslo 2.

Pak parametry modulu PWPB_RX v tomto projektu budou tyto:

```
priorita = 0
rychlost = 2
perioda = 5000
adresa = 2 [adresa stanice odkud beru data]
blok = 2
offset = 0
delka = 8 [2 data po 4 bajtech]
initout1..16 = 0.0
```

Modul PWPB_TX vysílá data D3 a D4 z této stanice do stanice s adresou 2, proto parametry budou tyto:

```
priorita = 0
rychlost = 2
perioda = 5000
adresa = 2 [adresa stanice kam posílám data]
blok = 1
offset = 0
delka = 8
```

Data D1..4 v projektech definujeme pomocí modulu CONSTF, data přivádíme na modul SCREEN pro jejich zobrazení pro kontrolu dat po komunikaci.

Porovnejme nyní projekt PB_ADR2.TXT pro druhou stanici MASTER s adresou 2 pro MPC.

Nastavení parametrů modulu PWPB_MAIN:

```
priorita = 0
rychlost = 2
kanal = 1
comrychlost = 9600
parita = 2
mezera = 3
prodleva = 10
odezva = 300
maxtoken = 500
adresa = 2 [adresa této stanice]
maxadresa = 2
```

Opět vidíte, že MAXTOKEN > 0, tj. stanice bude MASTER a MAXADRESA = 2, tj. TOKEN se bude předávat postupně mezi stanicemi s adresami 1 a 2.

Tento projekt je pro stanici, která vysílá data D1 a D2 ze stanice s adresou 2 na stanici s adresou 1, proto parametry modulu PWPB_TX budou vyplněny takto:

```
priorita = 0
rychlost = 2
perioda = 5000
adresa = 1
blok = 2
offset = 0
delka = 8
```

Data D3 a D4 tato stanice přijímá modulem PWPB_RX:

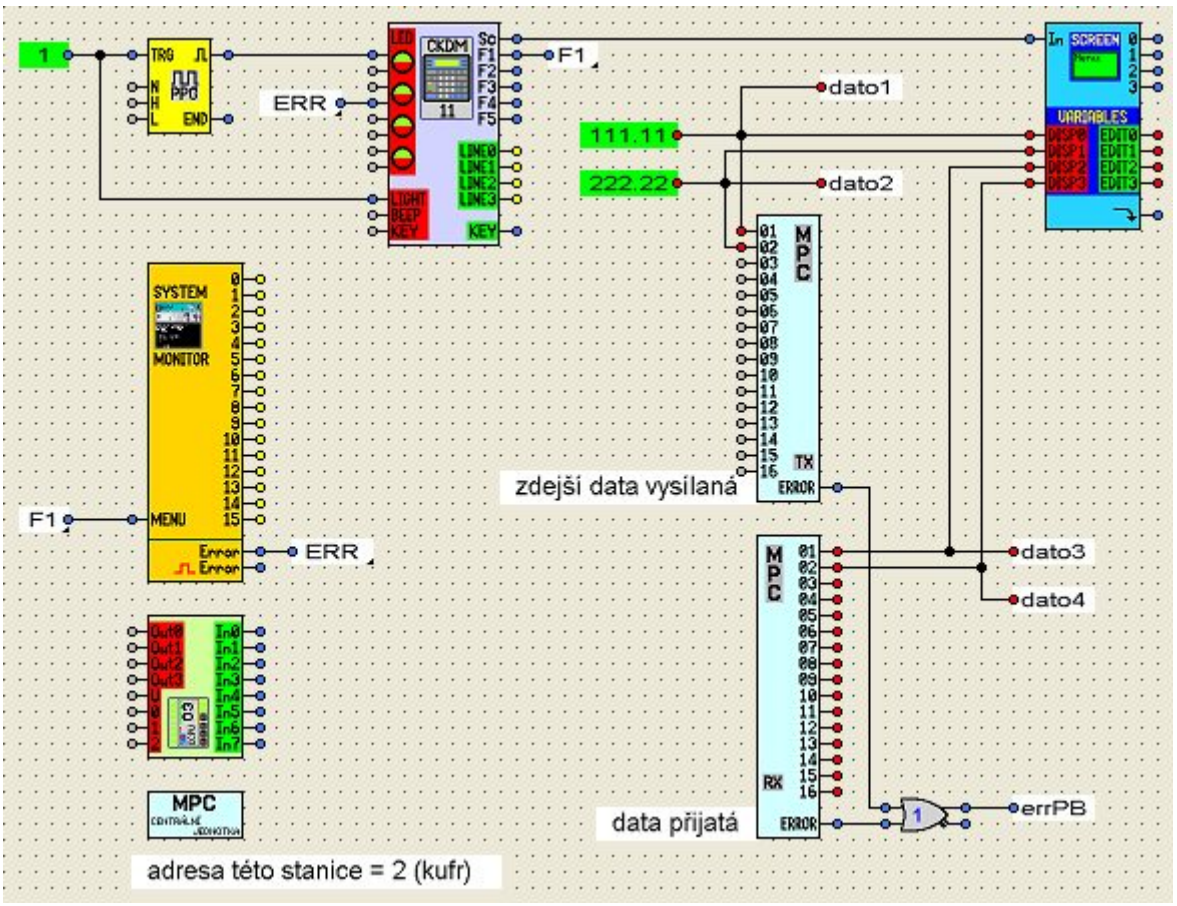
```
priorita = 0
rychlost = 2
perioda = 5000
adresa = 1 [adresa stanice odkud beru data]
blok = 1
offset = 0
delka = 8 [2 data po 4 bajtech]
initout1..16 = 0.0
```

Přehled o hodnotách dat, jak vysílaných, tak i přijatých, máme na displejích CKDM-11 u obou centrál.

Na displeji stanice s adresou 1 uvidíte:

```
Data prijata D1+D2:
  111.11    222.22
Data zdejsi D3+D4:
  333.33    444.44
```

A na displeji stanice s adresou 2:



Obr. 111 Propojení centrál - MPC multimaster, stanice 2, projekt PB_ADR2.opf

```
Data zdejsi D1+D2:
  111.11    222.22
Data prijata D3+D4:
  333.33    444.44
```

Podle popisu lze tak určit, která data na stanici vznikla a která jsou přijata z druhé stanice. Pro jednodušší kontrolu jsou data pojmenována a naplněna postupně zapamatovatelnými hodnotami :-)

D1 = 111.11
D2 = 222.22
D3 = 333.33
D4 = 444.44

Takže kontrola funkce přenosu dat by měla být jednoznačná a rychlá. V RUN režimu FREDa pak můžete měnit hodnoty floatových konstant v projektech a sledovat tak změnu přenášených dat na druhou stanici.

Pokud vám bude na ovládacím panelu občas nebo trvale svítit či blikat červeně LED s označením RUN, dochází k nějaké chybě komunikace. Většinou stačí časově komunikace sladit, např. dát větší hodnotu parametru ODEZVA apod.