department of signal processing

http://sp.utia.cz

Technická zpráva

Akademie věd České republiky
Ústav teorie informace a automatizace AV ČR, v.v.i.

# RCCOM

# Podpora komunikace s FPGA z prostředí Matlab a Matlab/Simulink

Roman Bartosinski
*bartosr@utia.cas.cz*

# Obsah



# Revize

| Revize | Datum | Autor | Popis změn v dokumentu |
|---|---|---|---|
| 0 | 17.2.2007 | R.B. | Vytvoření dokumentu pro RCCOM verze 1.1.2 |
| 1 | 20.12.2007 | R.B. | Doplnění dokumentu o popis ovládání z prostředí Matlab/Simulink |

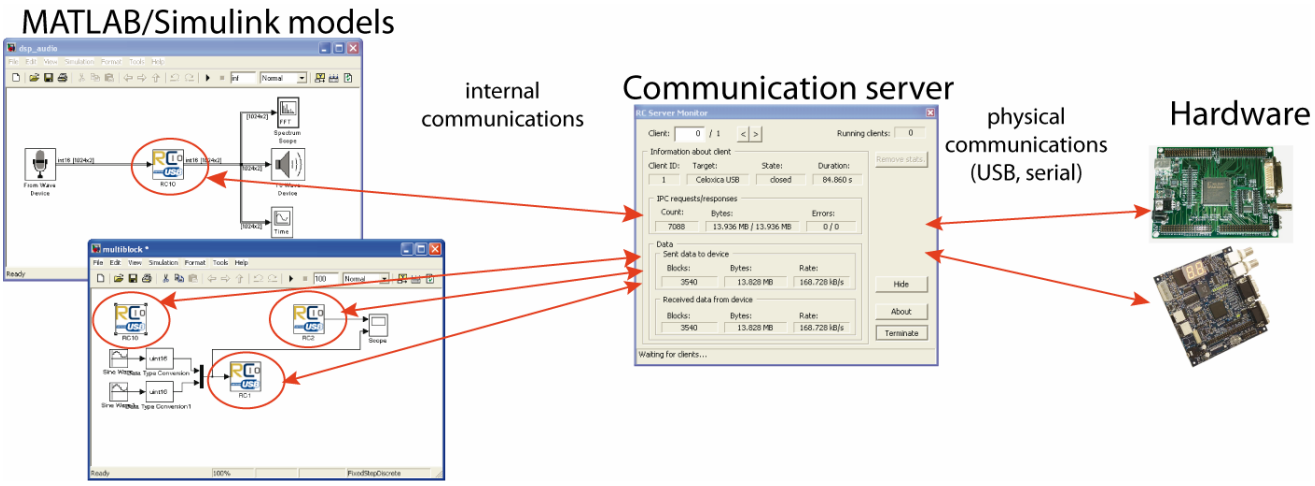
# 1. Úvod

RCCOM je knihovna pro simulační prostředí Matlab/Simulink, která umožňuje propojení softwarové simulace s hardwarem a tím i *hardware-in-the-loop* (HIL) simulaci. Primárně je tato knihovna určena pro připojení vývojových desek s FPGA od firmy Celoxica (desky RC10, RC250).
V této verzi knihovna podporuje komunikaci s vývojovou deskou pomocí sériového a USB rozhraní (USB rozhraní je podpořeno pouze pro karty firmy Celoxica). Komunikační server je složen z modulů a je připraven pro rozšiřování o podporu dalších komunikačních kanálů (ethernet, jtag) v podobě dynamicky linkovaných knihoven.

RCCOM objekt je rozšíření RCCOM knihovny pro snadnější přístup k hardwaru z Matlabu. Jde o soubor funkcí, které vytvoří a spravují speciální objekt *rccom*. Tento objekt je vytvořen podobně jako objekt *serial* nebo jiné objekty, které jsou dodávané v základním balíku Matlabu.
Tato zpráva obsahuje popis možností a použití knihovny v prostředí Matlab/Simulink a popis jednotlivých příkazů a vlastností objektu `rccom` prostředí Matlab.

# 2. Popis knihovny RCCOM

Knihovna umožňuje HIL simulaci a propojuje simulační prostředí Matlab/Simulink s funkcemi implementovanými v hardwaru, hlavně v obvodech FPGA. Účelem knihovny je urychlit vývoj a testování hardwarových výpočetních akcelerátorů. Blokové schéma propojení softwarové simulace s hardwarem je na obrázku 1.




Obrázek 1: Základní schéma datového toku

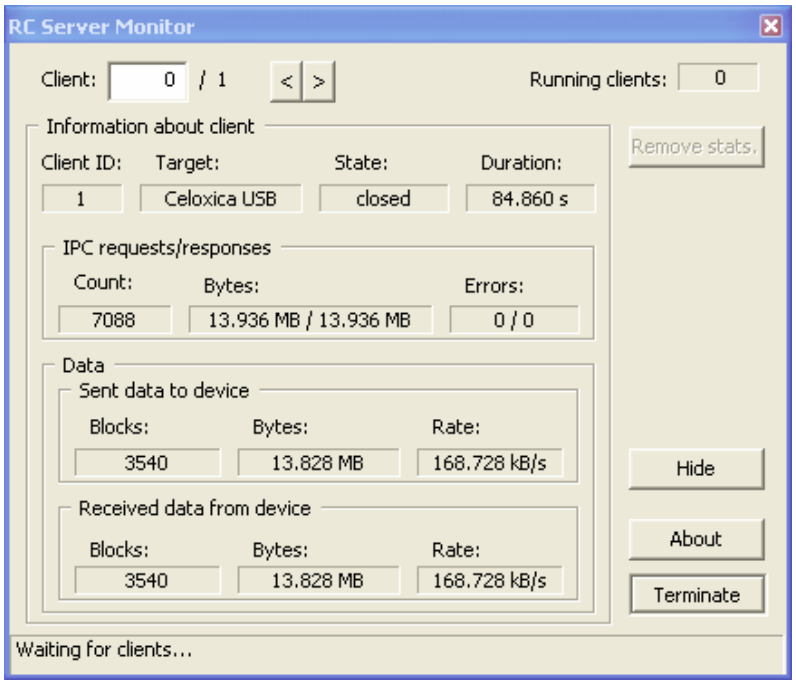
Komunikace je rozdělena na tři části (viz. obrázek 1) – část v hardwaru pro příjímání a vysílání dat, společný komunikační server v počítači a knihovna pro simulační nástroj Matlab/Simulink. V této zprávě je dále popis pouze komunikačního serveru a simulační knihovna, protože na straně FPGA se jedná o přímé čtení a zápis z/do daného rozhraní a to je přímo podpořeno v knihovnách PSL a PAL v jazyce Handel-C firmy Celoxica. Příklad zdrojového kódu s použitím funkcí v Handel-C je uveden u prvního příkladu v kapitole 6.

## 2.1 Komunikační server

Komunikační server je na pozadí běžící proces, který se vizuálně projevuje pouze jako ikona na hlavním panelu v oznamovací oblasti (obrázek 2) nebo monitorovací okno (obrázek 3).


Obrázek 2: Kopie části informační oblasti systému s ikonou RCCOM serveru

Server slouží k jednotnému přístupu z různých programů k hardwaru a monitorování komunikace. Současně tak lze komunikovat s jednou deskou z více simulačních modelů a případně dalších aplikací. Pro každé propojení server vytvoří samostatně běžící vlákno, které se stará o zpracování požadavků na přenos dat nebo nastavení hardwaru. A o celkovou správu propojení, tak aby nebyl ohrožen chod systému nebo aplikace, která komunikaci využívá.
Díky tomuto modelu komunikace lze stejným simulačním blokem (nebo aplikací) přistupovat k různým vývojovým deskám s různými komunikačními rozhraními. Zároveň je tak možné knihovnu rozšiřovat o podporu dalších komunikačních rozhraní bez nutnosti zásahu do vlastního modelu (nebo aplikace).
Vedlejší funkcí serveru je monitorování jednotlivých komunikačních kanálů. Dostupné informace se zobrazí při poklepání na ikoně RCCOM serveru (viz. obrázek 3).

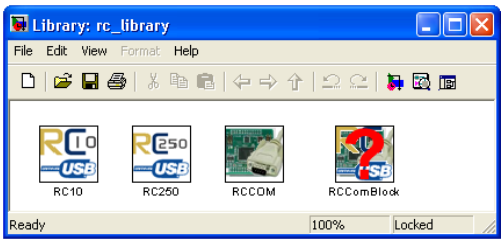



Obrázek 3: Okno statistik komunikačního serveru.

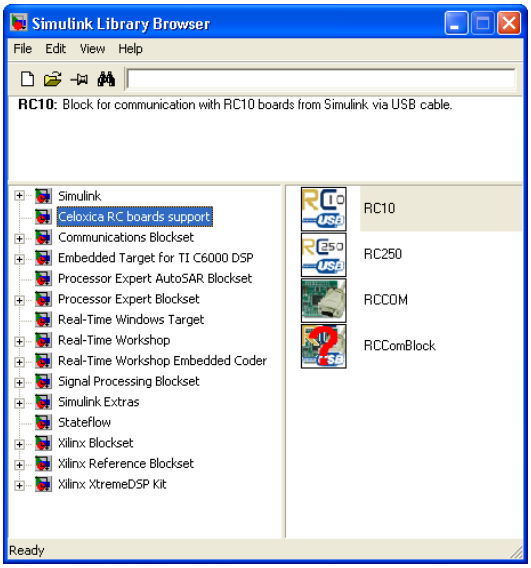
Monitorovací okno serveru obsahuje informace o jednotlivých otevřených i již uzavřených komunikačních kanálech. Dále informuje o stavu serveru a případně výpis poslední chyby komunikace. Pro jednotlivé kanály jsou zde informace o počtu přenesených dat, chyb přenosu a průměrných přenosových rychlostí pro čtení a zápis z/do hardwaru. V okně je také možné jednotlivé komunikační kanály násilně uzavřít, stejně jako je možné ukončit činnost celého serveru.

## 2.2 Simulační knihovna

Knihovna simulačních bloků (viz. obrázek 4) pro prostředí Matlab/Simulink je při instalaci přidána mezi ostatní knihovny bloků pod názvem „Celoxica RC boards support“ (viz. obrázek 5). V novější verzi pojmenované „RCCOM blockset“.




Obrázek 4: Knihovna bloků RCCOM pro Matlab/Simulink.

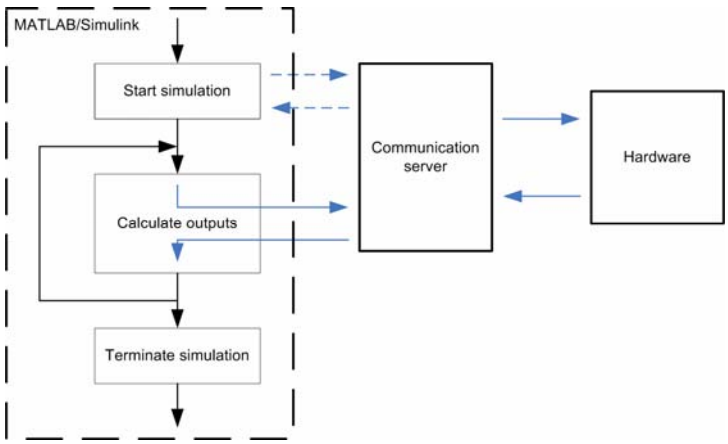



Obrázek 5: Prohlížeč knihoven bloků s přidanou knihovnou RCCOM.

Knihovna obsahuje čtyři bloky

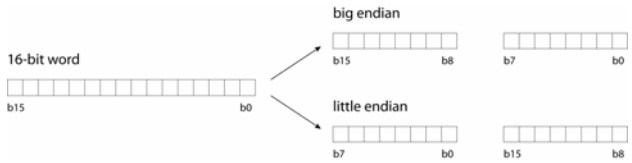
- blok pro komunikaci s vývojovou deskou Celoxica RC10 přes rozhraní USB,
- blok pro komunikaci s vývojovou deskou Celoxica RC250 přes rozhraní USB,
- blok pro komunikace přes sériové rozhraní,
- a univerzální blok (ten je v této verzi RCCOM knihovny přidán pouze pokusně a není ještě zcela podpořen).

Všechny bloky mají z hlediska simulace stejné chování, kdy se počet jejich vstupů a výstupů dynamicky mění podle aktuálního nastavení a připojení okolních bloků. Proto je popis chování z hlediska použití a simulace společný.




Obrázek 6: Komunikace s hardwarem při simulaci.

Z podstaty fungování Simulace v Matlabu/Simulinku jde o synchronní komunikaci (viz. obrázek 6), kdy se v každém kroku simulace vyšlou do hardwaru data přečtená na vstupu bloku reprezentujícího komunikaci a z hardwaru se zpět přečtou data představující výstup bloku. Pro fungující komunikaci musí být na straně Simulinku i na straně hardwaru shodně nastavený počet a typ předávaných dat. U bloků jsou v této verzi knihovny podporované pouze celočíselné typy – znaménková i neznaménková 8, 16 a 32 bitů široká data. K bloku lze připojovat signály s přenosem jednotlivých dat, vektorů nebo matic dat v jednom kroku. Protože fyzický přenos je prováděn po jednotlivých 8-bitových bytech, je třeba věnovat zvýšenou pozornost faktickému skládání dat při přenosu vektorů nebo matic a také pro jiné než 8-bitové typy.
Zjednodušení zpracování dat na straně hardwaru pomáhá možnost volby kódování dat širších než 8bitů (endianing) – buď od nižších bytů k vyšším (little endian) nebo naopak od vyšších bytů k nižším (big endian).



Obrázek 7: Možné kódování přenášených dat.

Při přenosu vektorů a matic jsou data skládána ve stejném pořadí jako je skládá Matlab. To znamená,že přenos probíhá po sloupcích v celé délce. Pro délku dat L a šířku vstupního nebo výstupního portu W je matice dat uspořádána takto

$$\begin{bmatrix} 1d1 & 1d2 & \cdots & 1dW \\ 2d1 & 2d2 & \cdots & 2dW \\ \vdots & \vdots & \ddots & \vdots \\ Ld1 & Ld2 & \cdots & LdW \end{bmatrix}$$

A do/z hardwaru jsou zapsána/čtena jako posloupnost dat

$$1d1,1d2,...,1dW,2d1,2d2,...,2dW,...,Ld1,Ld2,...,LdW \quad ,$$

kde *d* představuje jeden prvek dat se zadaným kódováním (little/big endian).
Data jsou do/z hardwaru zapisována/čtena se zadanou periodou vzorkování, která může být zděděna i z modelu, jak je běžné u většiny bloků. Primárně jsou bloky určeny pro diskrétní simulace, ale mohou být použity i pro spojité simulace. Navíc je možnost použít u bloků různé vzorkovací periody vstupů a výstupů, tzv. *multirate* bloky. Při využití této funkce je třeba mít na paměti, že se vzorkování vstupů a výstupů provádí podle jednoduchého pravidla rovnosti aktuálního simulačního času a násobku vzorkovací periody. Při tom může dojít k nechtěnému efektu, pokud je simulace zahájena v čase 0. S nastavením výstupní vzorkovací periody na dvojnásobek vstupní periody bude přenos dat fungovat tak, že se při simulaci v jednom časovém okamžiku pouze odešle vstup a v dalším časovém okamžiku se nejdříve odešle vstup do hardwaru a poté se přečte výstup z hardwaru. Ovšem při simulaci od času=0 se v čase 0 nejdříve odešle jeden vzorek vstupu a hned se přečte výstup hardwaru, i když ještě nebyl poslán druhý vzorek vstupu.
Protože je komunikace s hardwarem uskutečňována pomocí komunikačního serveru, lze k jedné desce přistupovat z více bloků v jednom modelu. Zde je potřeba uvědomit si pořadí přístupů k hardwaru, které je stejné jako je seřazení bloků při simulaci. Toto seřazení lze zobrazit z menu modelu – *Format→Block Displays→Sorted Order*.

## 2.3 Popis nastavení RCCOM bloků

Každý z bloků má několik nastavení, která lze změnit v okně parametrů bloku. Toto okno se otevře při poklepání na daném bloku. Na obrázcích 8, 9 a 10 jsou vyobrazeny okna nastavení bloků RC10, RC250 a RCCOM. Všechny bloky mají společné nastavení a vlastní speciální nastavení.

Mezi společné nastavení patří:

- **Šířka vstupního portu bloku** – Pro hodnotu 0 nemá blok vstupní port. Pokud je zadaná hodnota záporná, šířka portu je dynamicky určena z šířky připojeného signálu (předchozího bloku).
- **Délka vstupních dat** – (pro signály typu „frame“ je to délka rámce). Pokud je rovna nule, blok nemá vstupní port. Pro záporné hodnoty je délka určena z připojeného signálu od předchozího bloku. (Pro normální signály by měla být hodnota rovna 1.)
- **Typ vstupních dat** – Tato verze knihovny podporuje pouze přenos celočíselných typů (znaménkových i neznaménkových). Lze nastavit 'uint8','uint16','uint32', ‘int8’, ’int16’, ’int32’ a ‘auto’. Výběrem typu 'auto' bude typ určen z připojeného signálu, případně bude automaticky nastaven na 'uint8', pokud nelze typ určit podle okolí.
- **Šířka výstupního portu bloku** – Pro hodnotu 0 nemá blok výstupní port. Pokud je zadaná hodnota záporná, šířka portu je dynamicky určena z šířky připojeného signálu do následujících bloků.
- **Délka výstupních dat** – (pro signály typu „frame“ je to délka rámce). Pokud je rovna nule, blok nemá výstupní port. Pro záporné hodnoty je délka určena z připojeného signálu do následujících bloků. (Pro normální signály by měla být hodnota rovna 1.)
- **Typ výstupních dat** – Stejně jako u typu vstupních dat lze nastavit v této verzi knihovny pouze celočíselné typy a „auto“.
- **Data typu „frame“** – Data budou do bloku a z bloku posílána jako rámce definované v knihovně *Signal Processing Blockset*. Volba slouží pro jednodušší připojování modelů s využitím knihovny DSP. Při zaškrtnutí této volby lze přímo připojovat bloky knihovny DSP. Pro správnou funkci musí být DSP knihovna nainstalována s platnou licencí.
- **Používat kódování dat big endian** – Data jsou normálně posílaná v pořadí od nižších bytů do vyšších, jak jsou uložena v paměti (na architekturách i86). Při zaškrtnutí této volby budou data posílána v opačném pořadí, tj. od vyšších bytů k nižším.
- **Počet nul pro jednorázový zápis nebo proměnná Matlabu** – Umožňuje při startu simulace poslat určitý počet nul nebo dat do hardwaru. Pokud je zadané číslo, bude vyslán odpovídající počet nul. Pokud je zadané jméno proměnné, odešlou se všechna data proměnné.
- **Počet bytů pro jednorázové čtení nebo proměnná Matlabu** – Umožňuje při startu simulace načíst zadaný počet bytů a zahodit je nebo uložit do zadané proměnné. Pokud je zadané jméno proměnné, musí proměnná před spuštěním simulace již existovat a mít stejný počet a typ dat, jako se má načíst z hardwaru.
- **Změna pořadí jednorázového čtení a zápisu dat před simulací** – Pokud je požadováno jednorázové čtení i zápis, lze tímto nastavením změnit jejich pořadí. V normálním pořadí je nejdříve proveden zápis a poté čtení z hardwaru.
- **Vzorkovací perioda bloku** – Perioda vzorkování bloku, nebo -1 pro zdědění vzorkování z modelu. Pokud je povolení různé vzorkování vstupu a výstupu bloku, určuje tento parametr periodu vstupu.
- **„Multirate“ blok** – Povoluje různé vzorkování vstupu a výstupu bloku.
- **Vzorkovací perioda výstupu pro „multirate“ bloky** – Perioda vzorkování výstupu bloku

Speciální nastavení pro karty RCxx jsou následující:

- **Index připojené vývojové karty RC..** – Protože je možné mít k jednomu počítači připojeno více stejných karet RC10 nebo RC250 a všechny jsou řízené jedním ovladačem, musí být

tímto indexem adresovaná správná karta. Pokud je připojena pouze jedna karta k počítači, pak tato karta má index=0.

- **Konfigurovat FPGA při startu simulace** – Tato volba zpřístupňuje funkci rozhraní USB pro karty RCxx, která umožňuje nakonfigurování FPGA z určitého souboru v počítači.
- **Vymazat konfiguraci v FPGA před konfigurací** – Umožňuje vymazání obsahu FPGA před jeho novou konfigurací ze zadaného souboru.
- **Jméno souboru s konfigurací FGPA** – Cesta (i relativní) a jméno souboru s konfigurací FPGA.

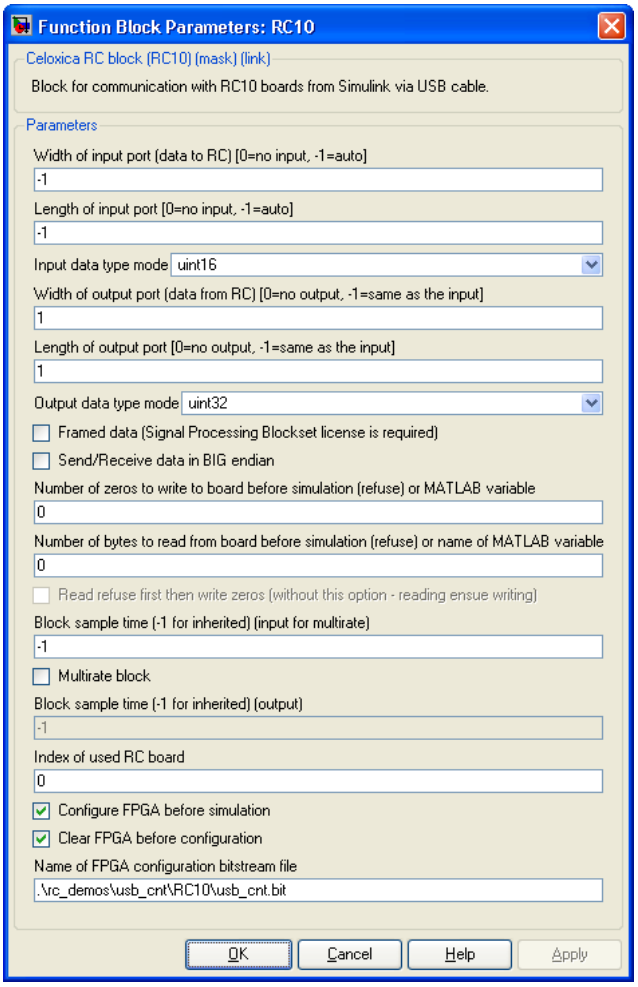
Speciální nastavení pro kartu RC250:

- **Frekvence prvního hodinového vstupu FPGA** - Číselná hodnota požadované frekvence hodin CLK0 pro FPGA. Pokud nelze nastavit požadovanou frekvenci, je při spuštění simulace vypsané varování v Matlabu s opravdu nastavenou frekvencí.
- **Frekvence druhého hodinového vstupu FPGA** – Stejné jako nastavení frekvence prvního hodinového vstupu.

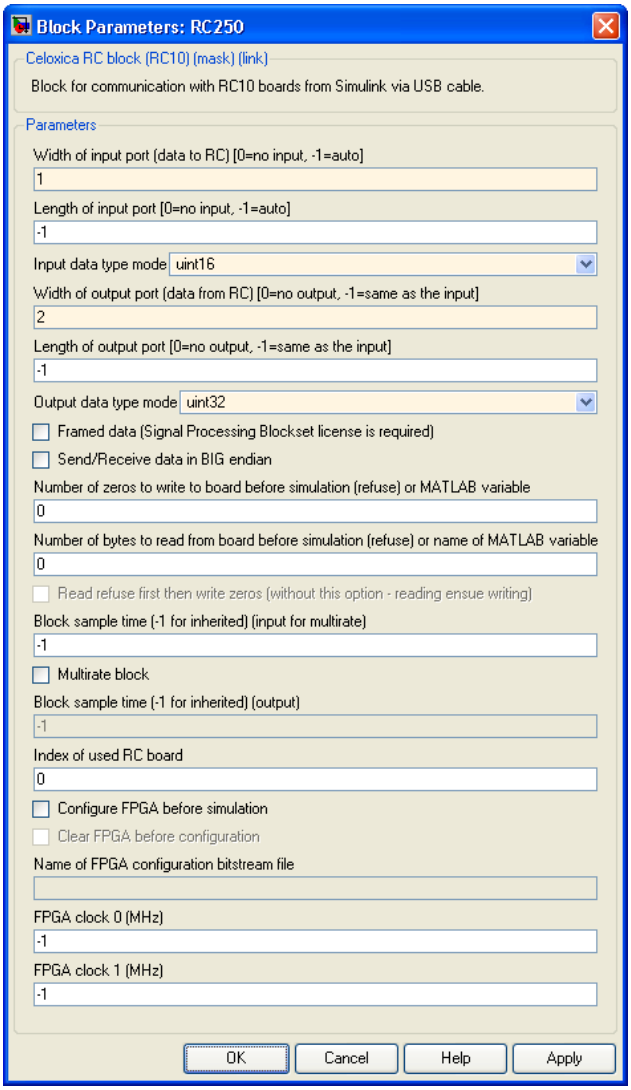
Speciální nastavení bloku RCCOM:

- **Index použitého portu sériové linky** – Číslo použitého sériového portu COMx. Standardně je přednastavena hodnota 1 pro použití COM1.
- **Nastavení COM portu** – Řetězec nastavující rychlost a další vlastnosti sériového přenosu. Formát řetězce je stejný jako u příkazu 'mode' na příkazové řádce. Základní parametry jsou:
  - baud - použitá přenosová rychlost,
  - parity - použitá parita,
  - data - počet bitů v přenášeném znaku,
  - stop - počet stop bitů.

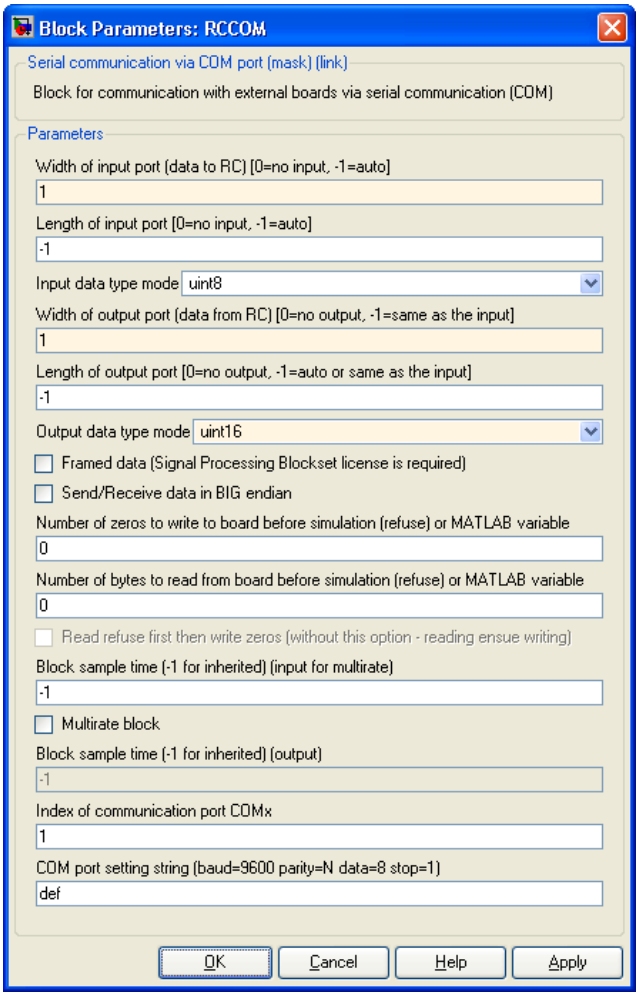
  Pokud je jako řetězec zadáno 'def', je nastavení ponecháno podle nastavení portu v systému. Pokud není zadán žádný řetězec, je použito přednastavení 9600,8,N,1.




Obrázek 8: Okno nastavení bloku RC10.




Obrázek 9: Okno nastavení bloku RC250.




Obrázek 10: Okno nastavení bloku RCCOM.

# 3. Známé problémy a jejich odstranění

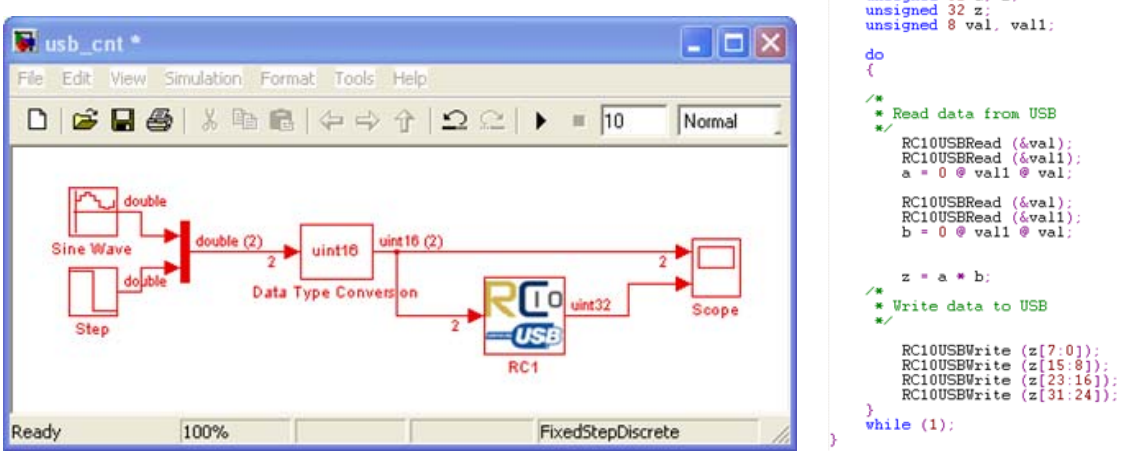
Některé známé možné potíže při používání bloků knihovny jsou popsány v sekci 2.2.

# 4. Popis příkladů pro Matlab/Simulink

Spolu s knihovnou jsou v adresáři */demos* přiloženy jednoduché příklady použití jednotlivých podporovaných funkcí knihovny.

## 4.1 Příklad USB_CNT

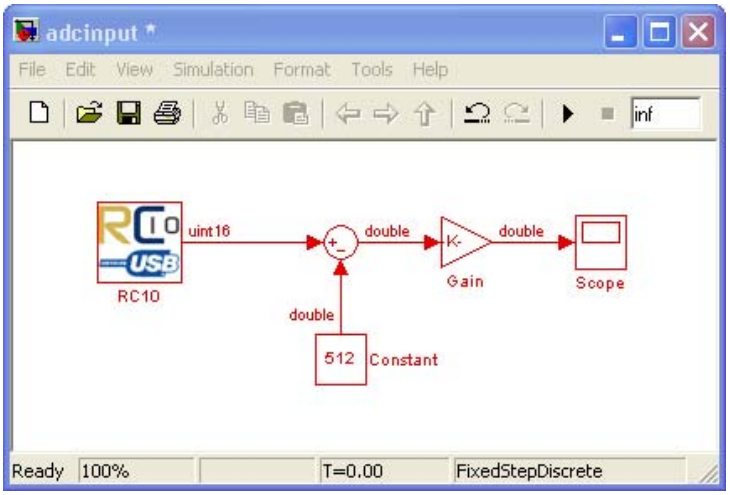
Příklad zápisu dvou 16-bitových čísel do karty RC10 a přečtení jednoho 32-bitového čísla z karty. V FPGA je implementován součin vstupních čísel.




Obrázek 11: Model příkladu *usb_cnt* a kód pro FPGA v Handel-C.

## 4.2 Příklad ADCINPUT

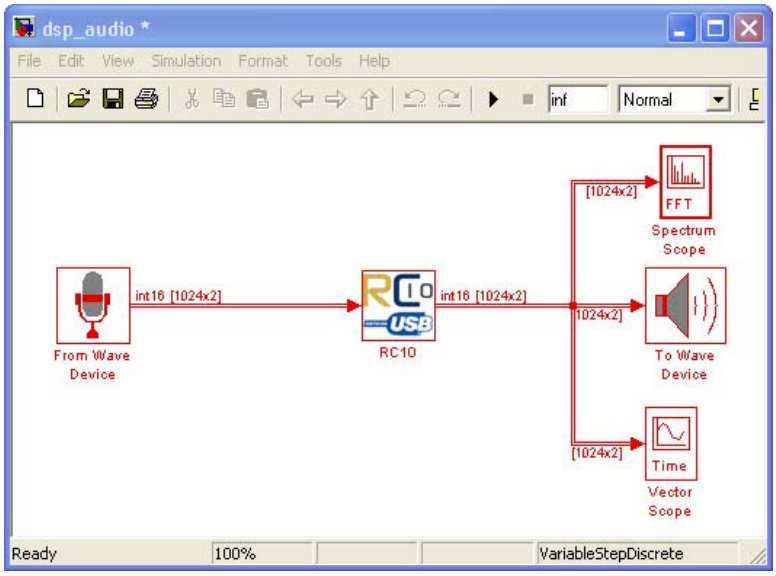
Pokus s vyčítáním dat z AD převodníku na kartě RC10. Aby příklad fungoval alespoň přibližně správně, je třeba mít nastavenu stejnou frekvenci v Simulinku i v HW pro čtení AD převodníku.




Obrázek 12: Model příkladu *adcinput*.

## 4.3 Příklad DSP_AUDIO

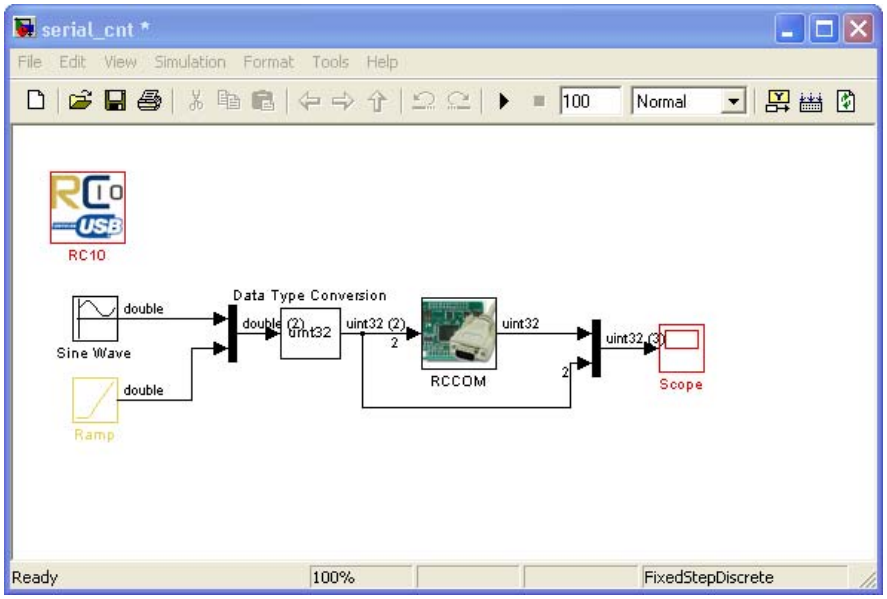
Pokus s daty v rámcích a signal processing blocksetem. Z mikrofonu jsou data načítána po rámcích 2X1024 vzorků typu int16 (stereo audio). Data jsou v RC10 filtrována a poslána zpět s následující úpravou – pokud je křížový ovladač vlevo, je levý kanál vypnut; pokud je ovladač vpravo, je pravý kanál vypnut. Při stisku ovladače jsou vypnuty oba kanály.




Obrázek 13: Model příkladu *dsp_audio*.

## 4.4 Příklad SERIAL_CNT

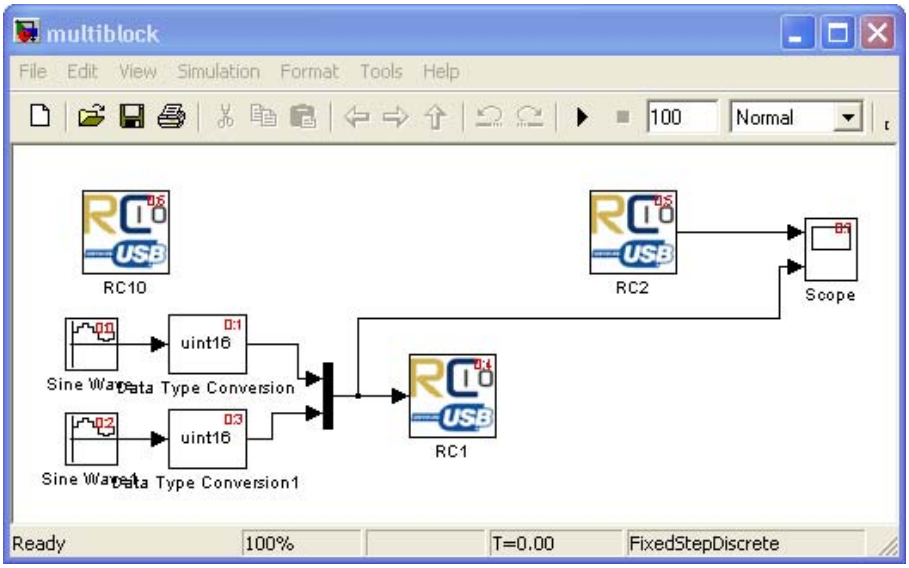
Jednoduchý testovací příklad, který komunikuje s deskou pomocí sériového portu – přečte vždy dvě 32bitová čísla a zpět posílá jedno 32bitové číslo. V FPGA je implementován součet vstupních čísel. Blok RC10 je v tomto příkladu použit pouze k počáteční konfiguraci FPGA a protože nemá žádný vstup ani výstup, tak se poté simulaci neúčastní.




Obrázek 14: Model příkladu *serial_cnt*.

## 4.5 Příklad MULTIBLOCK

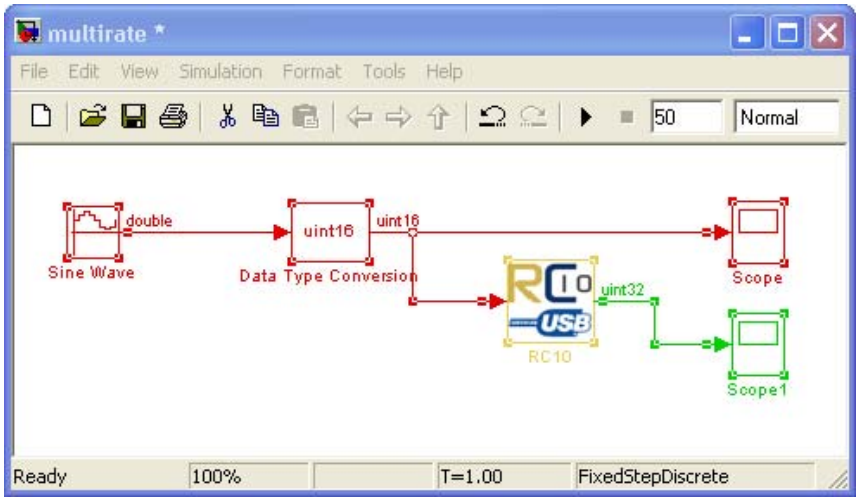
Stejný příklad jako *usb_cnt* jen s rozdílem rozdělení práce mezi více bloky. Konfigurace, čtení i zápis jsou vykonávány v samostatných blocích. U modelů s více bloky je třeba dávat pozor na pořadí bloků při simulaci.




Obrázek 15: Model příkladu *multiblock*.

## 4.6 Příklad MULTIRATE

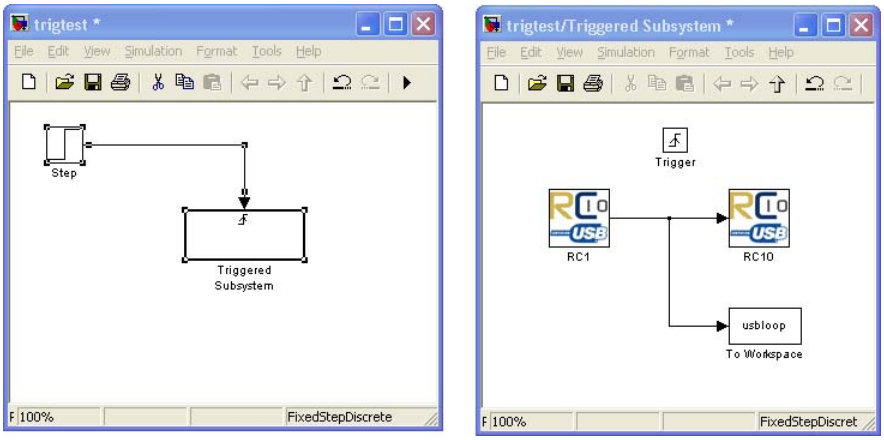
Stejný příklad jako *usb_cnt*, ale blok má pouze jeden vstup, který je zapisován dvakrát rychleji než je čten výstup. Simulace je startována od času = 1, aby se předešlo problému v čase 0, kdy je do bloku zapsáno a hned je z něj čteno (po prvním zápisu).




Obrázek 16: Model příkladu *multirate*.

## 4.7 Příklad TRIGTEST

Příklad jednorázově přečte 4000 bytu z karty a poté je jednorázově zapíše do karty. Příklad je určen pro kartu RC10 a nekonfiguruje FPGA před vlastní simulací, protože požívá základní vestavěný test karty RC10 'Board test', který musí být spuštěn. Kromě samotného poslaní dat zpět do desky jsou data pro kontrolu uložena i do proměnné Matlabu.




Obrázek 17: Model příkladu *trigtest*.

# 5. Objekt RCCOM

Objekt RCCOM slouží ke komunikaci s hardwarem přes rozhraní USB a sériovou linku a případně další rozhraní, které jsou podpořeny v RCCOM serveru. Seznam funkcí pro práci s objektem RCCOM je v tabulce 1.

| Funkce | Popis |
|---|---|
| rccom_getversion | zjištění verze komunikačního serveru |
| rccom_enumifaces | zjištění všech podporovaných komunikačních rozhraní |
| rccom_enumdevices | zjištění všech dostupných zařízení pro dané komunikační rozhraní |
| rccom_enumobjects | zjištění všech RCCOM objektů |
| rccom | vytvoření nového RCCOM objektu |
| fopen | otevření komunikace definované RCCOM objektem |
| fclose | uzavření komunikace |
| get | přečtení hodnoty nastavení RCCOM |
| set | změna hodnoty nastavení objektu RCCOM |
| fread | čtení dat ze zařízení |
| fwrite | zápis dat do zařízení |
| delete | zrušení objektu RCCOM |

Tabulka 1: Seznam funkcí pro práci s RCCOM objektem

Při zavolání jakékoliv z těchto funkcí je spuštěn komunikační server, pokud ještě neběží. Popis komunikačního serveru je v technické zprávě s názvem „RCCOM knihovna“. V následujících sekcích je popis jednotlivých příkazů. Součástí jsou vždy i popsané volání funkcí z příkazového řádku Matlabu.

## 5.1 rccom_getversion

Funkce slouží ke zjištění verze běžícího, nebo spouštěného, komunikačního serveru. Příkaz buď vypíše nebo vrátí jméno a verzi serveru. Vracenou identifikaci serveru lze použít ve skriptech pro zjištění podporovaných funkcí používané verze serveru.
Příkaz lze zavolat buď samostatně nebo jako přiřazení proměnné, do které se identifikace serveru uloží jako řetězec.

Vypsání identifikace serveru:

```
>> rccom_getversion
RC server version: 'RCCOM server v1.1.2'
```

Uložení identifikace (s vypsáním) do proměnné `ver`:

```
>> ver = rccom_getversion
ver =
RCCOM server v1.1.2
```

## 5.2 rccom_enumifaces

Funkce slouží ke zjištění komunikačních rozhraní, které jsou běžícím serverem podporované. Při založení RCCOM objektu je jedním z parametrů identifikace rozhraní, přes které je zařízení připojeno. Tuto identifikaci rozhraní lze získat z výstupu této funkce.
Protože je komunikační server připraven na dynamické přidávání podporovaných rozhraní, je při výčtu všech připojených zařízení třeba nejdříve zjistit tímto příkazem všechna podporovaná rozhraní.
Příkaz lze zavolat buď samostatně nebo jako přiřazení proměnné, do které se uloží sloupcový vektor buněk obsahujících řetězce s identifikátory rozhraní.

Vypsání všech podporovaných rozhraní:

```
>> rccom_enumifaces
RC server known interfaces: 'Celoxica USB,Serial'
```

Uložení výčtu rozhraní jako vektoru buněk do proměnné `ifc` (s vypsáním vektoru):

```
>> ifc = rccom_enumifaces
ifc =
    'Celoxica USB'
    'Serial'
```

## 5.3 rccom_enumdevices

Funkce provede výčet připojených a použitelných zařízení a nebo výčet použitelných portů pro sériové rozhraní. Funkce vypíše nebo vrátí identifikaci všech zařízení pro dané rozhraní. Součástí identifikátoru je i index zařízení pro odlišení více zařízení se stejným identifikátorem.
Funkce potřebuje jako vstupní parametr identifikátor rozhraní, pro které se má výčet provést. Tento identifikátor lze získat funkcí `rccom_enumifaces`.

Vypsání všech využitelných zařízení pro rozhraní `Serial`:

```
>> rccom_enumdevices('Serial')
Connected and usable devices: '1-COM1'
```

Uložení výčtu v danou chvíli připojených a využitelných zařízení jako vektoru buněk do proměnné `devs` (s výpisem obsahu vektoru na obrazovce):

```
>> devs = rccom_enumdevices('Celoxica USB')
devs =
    '0-Celoxica RC10(FW:0011)'
    '1-Celoxica RC10(FW:0011)'
```

## 5.4 rccom_enumobjects

Funkce vypíše všechny `rccom` objekty vytvořené a dosud nezrušené. Při smazání proměnné Matlabu funkcí `clear` nedochází ke zrušení objektu, ale pouze ke smazaní ukazatele (proměnné Matlabu) na objekt. Po smazání proměnné je `rccom` objekt stále aktivní. Pro fyzické zrušení objektu je třeba použít funkci `delete` (viz. dále v textu).
Díky této funkci je možné najít objekty se smazaným propojením k proměnné Matlabu.
Funkce nemá žádné vstupní parametry a vrací pole buněk s `rccom` objekty, které nebyly korektně smazány pomocí funkce `delete`.

Uložení seznamu existujících objektů do proměnné `objs (dva rccom objekty jsou vytvořené)`:

```
>> objs = rccom_enumobjects
objs =
    [1x1 rccom]
    [1x1 rccom]
```

## 5.5 rccom

Funkce pro vytvoření nového `rccom` objektu ze zadané požadované identifikace rozhraní a identifikace zařízení. Zavoláním této funkce se vždy vytvoří `rccom` objekt, i když není žádné rozhraní a zařízení vybrané. Při zavolání funkce s identifikátorem rozhraní se vytvoří objekt s přednastaveným rozhraním, ale bez vybraného konkrétního zařízení. Funkci lze též zavolat s identifikátorem rozhraní a zařízení, tím se vytvoří objekt s přiřazeným zařízením. Namísto identifikátoru zařízení lze také zadat pouze index zařízení.
Funkce zařízení neotevírá, pouze vytváří objekt propojení k zařízení. Pokud není při vytvoření objektu nastavena identifikace rozhraní nebo zařízení, lze je později nastavit pomocí příkazu `set`.
Vytvořený objekt lze zrušit pomocí funkce `delete`.
Všechny funkce uvedené dále v této části potřebují jako jeden z parametrů proměnnou `rccom` objekt.
Funkci je možné volat s následujícími parametry:
Bez parametrů – tím se vytvoří objekt bez určeného rozhraní i zařízení

```
>> o = rccom
o =
   Status    = Closed
   Interface hasn't been selected yet
   Device hasn't been selected yet
  --- object settings ---
   Endianing = little endian
   Verbosity = 1
   LastError = 0
```

Jeden parametr - identifikátor rozhraní

```
>> o = rccom('Celoxica USB')
o =
         Status    = Closed
   Interface = Celoxica USB
   Device hasn't been selected yet
  --- object settings ---
```

```
    Endianing = little endian
    Verbosity = 1
    LastError = 0
```

Dva parametry – identifikátor rozhraní a identifikátor zařízení

```
>> o = rccom('Celoxica USB','0-Celoxica RC10(FW:0011)')
o =
    Status    = Closed
    Interface = Celoxica USB
    Device    = 0-Celoxica RC10(FW:0011)
   --- object settings ---
    Endianing = little endian
    Verbosity = 1
    LastError = 0
```

Dva parametry – identifikátor rozhraní a index zařízení

```
>> o = rccom('Celoxica USB',0)

o =
    Status    = Closed
    Interface = Celoxica USB
    Device(idx) = 0
   --- object settings ---
    Endianing = little endian
    Verbosity = 1
    LastError = 0
```

Návratovou hodnotou funkce je vytvořený objekt. Objekt je vytvořen při každém volání i když není výstup funkce přiřazen. Takové objekty - stejně jako smazané, ale nezrušené, objekty – lze získat funkcí `rccom_enumobjects`.
Vytvořený objekt má různé vlastnosti podle jeho momentálního stavu propojení se zařízením – viz. popis příkazu `set` a `get`.

## 5.6 fopen

Funkce propojuje `rccom` objekt se zařízením, které je tímto objektem popsané. Pokud je komunikační kanál (zařízení) přístupný, kanál (zařízení) je otevřen a lze s ním komunikovat. Tím se vlastnost objektu `Status` změní na `Opened` a změní se dostupné vlastnosti objektu, podle otevřeného kanálu. Pro komunikaci se zařízením (pomocí funkcí `fread`, `fwrite`, `set` a `get`) je třeba komunikaci úspěšně touto funkcí otevřít.
Jediným parametrem funkce je RCCOM objekt vytvořený pomocí funkce `rccom`. Funkce vrací chybový kód.

```
>> rc = rccom('Celoxica USB',0);
>> err = fopen(rc)
err =
     0
```

## 5.7 fclose

Funkce uzavírá komunikaci se zařízením (popsané pomocí objektu `rccom)`, dříve otevřenou funkcí `fopen`. Status objektu se úspěšným uzavřením komunikace změní na `Closed`. Komunikaci lze znovu otevřít funkcí fopen.
Funkce má jako vstupní parametr rccom objekt.

```
>> rc = rccom('Celoxica USB',0);
>> err = fopen(rc);
>> fclose(rc)
```

## 5.8 get

Funkce `get` slouží k získání hodnot vlastností objektu stejně jako pro další objekty v prostředí Matlab. Prvním parametrem je vždy objekt `rccom` a dalšími parametry jsou jména požadovaných vlastností. Seznam dostupných vlastností je dynamický podle momentálního stavu objektu a zařízení popsané tímto objektem. Seznam možných vlastností lze rozdělit na vlastnosti objektu a vlastnosti zařízení. Jejich výčet a popis je v následující sekci 3.
Funkce vrací hodnotu nebo pole buněk s hodnotami požadovaných vlastností.
Pokud je funkce zavolaná pouze s 1.parametrem – objektem `rccom` - je vypsán nebo vrácen jako struktura seznam momentálně dostupných vlastností objektu `rccom` a jejich hodnoty.

Příklad přečtení hodnoty vlastnosti ‚Status‘:

```
>> rc = rccom('Celoxica USB',0);
>> val = get(rc,'Status')
val =
     0
```

## 5.9 set

Funkce `set` umožňuje nastavit hodnoty vlastností objektu. Syntaxe příkazu je stejná jako pro jiné objekty v prostředí Matlab. Seznam nastavitelných vlastností a jejich popis je v následující sekci 3, tento seznam je dynamický podle stavu objektu a popisovaného zařízení.
Funkce vrací hodnotu 1 pokud se nastavení hodnoty vlastnosti povedlo, jinak je vrácena hodnota 0.
Pokud je funkce zavolána pouze s 1.parametrem je vrácena struktura nebo vypsán seznam vlastností, které je možné momentálně nastavit.

Příklad nastavení pořadí bytů ve slově na `big endian`.

```
>> rc = rccom('Celoxica USB',0);
>> err = set(rc,'BigEndian',1)
err =
     0
```

## 5.10 fread

Funkce přečte data ze zařízení. Syntaxe příkazu je obdobná jako pro jiné objekty v prostředí Matlab. Tato funkce přečte maximálně požadovaný počet znaků požadovaného typu. Pokud je nastavené hlídání času a dojde k jeho naplnění je vrácen počet znaků přečtených do té doby. Pokud dojde

během čtení k chybě, je tato chyba vypsána (pokud je povolená vlastnost 'Verbosity') a chybový kód je uložen do 'LastError'. Funkce umí pracovat z následujícími typy dat:

`char, int8, uint8, int16, uint16, int32, uint32, int64, uint64, float32, float64, double.`

Typ vrácených dat je nastaven podle požadovaného typu. Funkce je blokující, tzn. pokud neskončí dokud není přečten požadovaný objem dat nebo nenastane timeout.

Příklad přečtení 20 bytů

```
>> rc = rccom('Celoxica USB',0);
>> data = fread(rc,20,'uint8')
data =
     '12345678901234567890'
```

## 5.11 fwrite

Funkce zapíše data do zařízení. Syntaxe příkazu je obdobná jako pro jiné objekty v prostředí Matlab. Funkcí lze zapsat číselnou nebo znakovou (řetězec) matici dat. Pokud je zadán další parametr `precision` jsou data braná jako zadaný typ (viz. typy platné pro funkci `fread`). Pokud je nastaven timeout a nestihne se odeslání všech dat do tohoto času, vrátí funkce chybu. Pokud vše proběhne v pořádku vrátí funkce 0 jako úspěch. Funkce je blokující a tedy neskončí dokud nezapíše všechny data nebo nenastane chyba (timeout).

Příklad odeslání 10 číslic typu double

```
>> rc = rccom('Celoxica USB',0);
>> err = fwrite(rc,[1,2,3,4,5,6,7,8,9,10])
err =
     0
```

## 5.12 delete

Funkce uvolňuje objekty `rccom` z paměti. Pouhé `clear rco` neuvolní paměť, ale pouze smaže proměnnou, která spojovala objekt s prostředím Matlab. Objekty nezrušené pomocí tohoto příkazu budou smazány při ukončení Matlabu. Pokud objekt označuje momentálně otevřené zařízení (pomocí příkazu `fopen`) je toto zařízení nejdříve automaticky uzavřeno (`fclose`).
Všechny nesmazané objekty lze zjistit příkazem `rccom_enumobjects`.

Příklad vytvoření a zrušení objektu RCCOM

```
>> rc = rccom('Celoxica USB',0);
>> rc
rc =
   Status    = Closed
   Interface = Celoxica USB
   Device    = 0-Celoxica RC10(FW:0011)
  --- object settings ---
   Endianing = little endian
   Verbosity = 1
   LastError = 0
```

```
>> delete(rc)
rc =
   Warning: Invalid object!
```

## 6. Vlastnosti objektu RCCOM

Výčet vlastností objektu RCCOM, které lze zístat pomocí funkce `get` a nastavit pomocí funkce `set`, je dynamicky měněn podle stavu objektu a zařízení, které popisuje.
Každý objekt (i označující zatím neotevřené zařízení) má následující vlastnosti

- pro `get`:
  - `Interface`
  - `DeviceName`
  - `DeviceIndex`
  - `Status`
  - `BigEndian`
  - `Verbosity`
  - `LastError`

- pro `set`:
  - `Interface`
  - `DeviceName`
  - `DeviceIndex`

  A pouze tyto při otevřeném zařízení
  - `BigEndian`
  - `Verbosity`

Vlastnosti jednotlivých zařízení (otevřených) je možné zadáním příkazu `get` a `set` pouze se jménem objektu (`get(rc)` nebo `set(rc)`).

## 7. Příklady použití objektu RCCOM

Základní příklady na použití jednotlivých příkazů jsou uvedeny přímo u daných příkazů.

## 8. Potřebné nástroje

Pro spuštění komunikačního serveru je třeba mít správně nainstalovaný ovladač RCUSB, který slouží pro komunikaci s kartami RC přes rozhraní USB. Tento ovladač je součástí programové podpory u každé karty RC nebo jako součást knihovny PDK vývojového nástroje Celoxica Handel-C. Komunikační server běží momentálně pouze v systému Windows a je otestován ve verzi Windows 2000 a Windows XP.
Propojovací blok v Simulinku byl testován v prostředí Matlab RC14 až Matlab 2007a. Přiložená verze knihovny je zkompilovaná proti 32bitovému prostředí Matlab2007a a vyzkoušena i na verzi Matlab2006b.

## 9. Postup instalace a použití knihovny

Knihovnu není třeba složitě instalovat. Je možné pouze zkopírovat adresář *rc_library* někam na disk a přidat v Matlabu cestu k této knihovně. Při zkopírování skriptu *install_rclibrary.m* spolu s knihovnou a jeho spuštěním bude cesta ke knihovně přidána do Matlabu automaticky.
Při používání se nejdříve Simulinkový blok pokusí otevřít komunikaci se serverem a pokud žádný server neběží, je server automaticky spuštěn. Při běhu simulace se v příkazovém okně Matlabu vypisují případně upozornění a chyby komunikace.
Při práci s RCCOM objektem se při otevření zařízení nebo pokusu o enumeraci existujících zařízení také spustí automaticky server, pokud zatím neběžel. Případné výpisy chyb lze zrušit nastavením vlastnosti objektu `verbosity` na 0.

## 10. Obsah přiloženého CD

Přiložené CD obsahuje knihovnu RCCOM a příklady použití (modely i konfigurace FPGA). Struktura adresářů je následující:

```
cdrom    - doc
         - rccom - rc_library - demos - rc_demos
                              - private
```

V adresáři `/doc` je tato technická zpráva v elektronické podobě. V adresáři `/rccom` je celá knihovna i s příklady (v adresáři `/rccom/rc_library/demos`), kterou je nutné zkopírovat. V adresáři `/rccom` je také soubor `install_rclibrary.m`, který může ulehčit přidání cesty ke knihovně po jejím zkopírování.

## 11. Poděkování

Tato práce byla podpořena Národním programem výzkumu v rámci projektu SESAp pod identifikačním číslem 1ET400750406.

## 12. Reference

[1] Celoxica, *RC10 Manual*, www.celoxica.com
[2] The Mathworks, MATLAB User's guide, www.mathworks.com
[3] The Mathworks, Writing S-Functions, www.mathworks.com
[4] Bartosinski, R.; Kadlec, J. Hardware co-simulation with communication server from MATLAB/Simulink. In Technical computing Prague 2006. 14th annual conference Technical computing Prague 2006