ADS Matlab a Simulink toolbox

ADS toolbox slúži na sieťovú komunikáciu so systémom TwinCAT. Implementuje prácu s premennými, ktoré môžu byť namapované na fyzické vstupy a výstupy.

1. Kompilácia kódu

Pre každú odlišnú verziu Matlabu a Simulinku je nutné funkcie toolboxu individuálne prekompilovať.

Na tento účel slúži zvláštny skript kompilujúci ako implementované MEX funkcie tak aj S funkcie. Kompiláciu všetkých funkcii teda realizujeme spustením *build_script.m*. Prirodzene je nutné mať všetky zdrojové súbory , hlavičkové súbory (.cpp .h) a knižnice (.lib,.dll) v pracovnom priečinku. Taktiež treba mať v Matlabe nainštalovaný a nastavený kompilátor (najlepšie MinGW). Kód je tak úspešne skompilovaný po vypísaní:

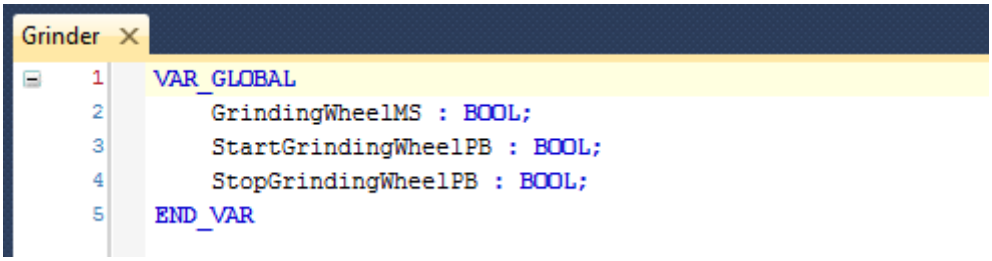
```
Building with 'MinGW64 Compiler (C++)'.
MEX completed successfully.
```

Kompilačný script vyzerá nasledovne.

```
mex ('ADS_write_mex.cpp','ADS_lib.cpp',strcat('-L',pwd()),'-lTcAdsDll')
mex ('ADS_read_mex.cpp','ADS_lib.cpp',strcat('-L',pwd()),'-lTcAdsDll')
mex ('ADS_open_mex.cpp','ADS_lib.cpp',strcat('-L',pwd()),'-lTcAdsDll')
mex ('ADS_close_mex.cpp','ADS_lib.cpp',strcat('-L',pwd()),'-lTcAdsDll')
mex ('ADS_setup_SFUN.cpp','ADS_lib.cpp',strcat('-L',pwd()),'-lTcAdsDll')
mex ('ADS_write_SFUN.cpp','ADS_lib.cpp',strcat('-L',pwd()),'-lTcAdsDll')
mex ('ADS_read_SFUN.cpp','ADS_lib.cpp',strcat('-L',pwd()),'-lTcAdsDll')
```

2. Dátové typy

Každá premenná v systéme TwinCat má svoj dátový typ. Ten sa dá zistiť v zozname globálnych premenných (global variable list - GVL) na strane servera.



V rámci tohto ADS toolboxu sú týmto dátovým typom priradené identifikačné čísla. Preto pri práci s väčšinou funkcii je nutné poznať dátový typ premennej s ktorou chceme pracovať a teda aj poznať číslo tohto dátového typu. Identifikačné číslo dátového typu vstupuje ako parameter do MEX funkcii a rovnako ako parameter v prípade masiek S-funkcii.

Zoznam dátových typov vyzerá nasledovne , je rovnako obsiahnutý v súbore *TC_data_types.m*

```
TC_BOOL_type=0
TC_BYTE_type=1
TC_WORD_type=2
TC_DWORD_type=3
TC_INT_type=4
TC_DINT_type=5
TC_LINT_type=6
TC_USINT_type=7
TC_UINT_type=8
TC_UDINT_type=9
TC_ULINT_type=10
TC_REAL_type=11
TC_LREAL_type=12
```

Je vhodné si tento súbor pred prácou spustiť a zadefinovať si tak prehľadnejšie mená pre dátové typy.

3. Matlab MEX funkcie

MEX funkcie sú funkcie, ktoré je možné volať štandardným rozhraním Matlabu, teda priamo v príkazovom riadku. Rovnako je možné ich používať v skriptoch a Matlabovských funkciách. Tieto funkcie sú však kompilovaným kódom a nie je možné do nich pri debugovaní vstupovať a ani ich vnútornú činnosť modifikovať.

Pre účely tohoto toolboxu sú implementované nasledovné:

`ADS_open_mex();`
Funkcia slúži na otvorenie ADS komunikačného portu. Nevstupujú do nej žiadne argumenty a žiadny výsledok nevracia. Túto funkciu je nutné zavolať na začiatku programu/skriptu.

`ADS_close_mex();`
Funkcia slúži na zatvorenie ADS komunikačného portu. Nevstupujú do nej žiadne argumenty a žiadny výsledok nevracia. Túto funkciu je nutné zavolať na konci programu/skriptu

`ADS_write_mex([net_id],'nazov_premennej',cislo_datoveho_typu ,vstupna_hodnota);`
Funkcia slúži na zápis hodnoty do premennej.

Vstupné parameter funkcie:

- net_id – adresa servera v sieti, 6 miestny riadkový vektor v štandarnnej syntaxi Matlabu napríklad `[10, 3, 1, 138, 3, 1]`
- názov premennej – reťazec názvu premennej v TwinCAT, štandardnný tvar: 'GVL.premenna'
- dátovy typ – klasický celočíselný skalár, identifikačné číslo dátového typu, napríklad 4 pre typ INT
- vstupná hodnota – hodnota, ktorú chceme zapisovať, reálnočíselný skalár. V prípade, že premenná je celočíselného typu, dôjde k zaokrúhleniu.

Funkcia nevracia žiadnu hodnotu

`y=ADS_read_mex([net_id],'nazov_premennej',cislo_datoveho_typu);`
Funkcia slúži na čítanie hodnoty premennej.

Vstupné parameter funkcie:

- net_id – adresa servera v sieti, 6 miestny riadkový vektor v štandarnnej syntaxi Matlabu napríklad `[10, 3, 1, 138, 3, 1]`
- názov premennej – reťazec názvu premennej v TwinCAT, štandardnný tvar: 'GVL.premenna'
- dátovy typ – klasický celočíselný skalár, identifikačné číslo dátového typu, napríklad 4 pre INT

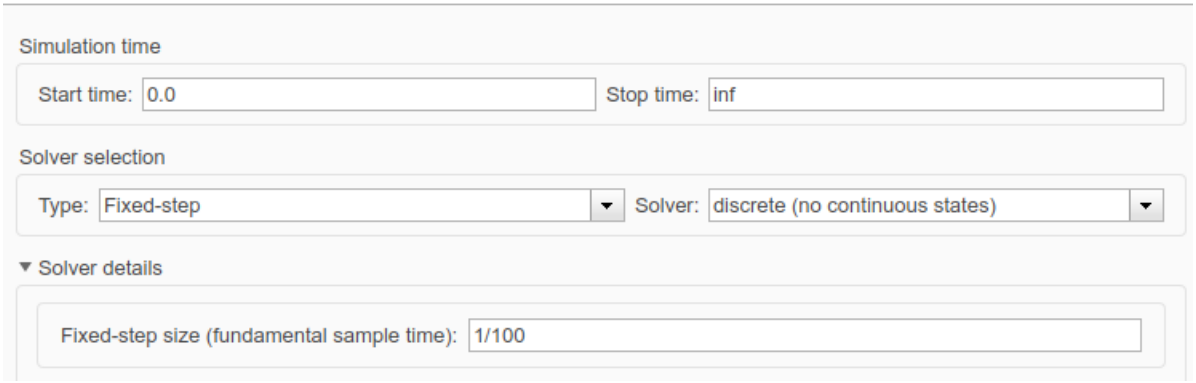
Funkcia vracia reálnočíselnú hodnotu premennej (aj v prípade, že je premenná celočíselného typu).

4. Simulink S-funkcie

S funkcie sú rozšírením štandardných knižníc Simulinku o vlastný kompilovaný kód so špecifickou funkcionalitou. S funkcie teda umožňujú používať štandardné rozhranie Simulinku (vstupy, výstupy) vo vlastnom kóde , napríklad v tomto prípade v jazyku C++. S-funkcie sú programované úplne nezávisle od MEX funkcii aj keď principiálne implementujú podobnú funkcionalitu.

Pre správnu funkčnosť ADS toolboxu je nutné v simulačnej schéme spraviť niektoré nastavenia.

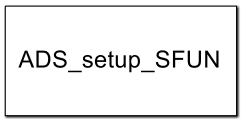
Volanie užitočného kódu – komunikácie so serverom sa uskutočňuje v diskrétnych krokoch. Preto je nutné nastaviť solver simulácie na diskrétny (bez spojitých stavov) s konštantnou periódou.



Aby simulácia mohla bežať v reálnom čase je nutné do schémy vložiť jeden blok Real-Time sync z knižnice toolboxu Simulink desktop real-time

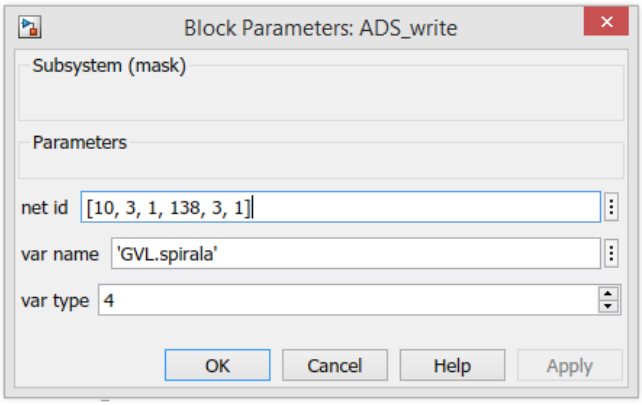
a. ADS setup

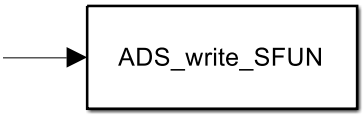
Tento blok sa stará o automatické otvorenie komunikačného portu pri štarte simulácie a jeho uzatvorenie na konci. S-funkcia nemá žiadne vstupy, výstupy ani parametre a je nutné mať iba jeden takýto blok v schéme.



b. ADS write

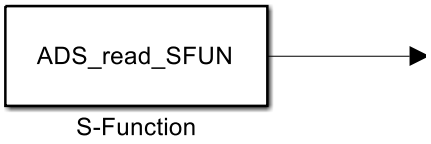
Blok na zápis do premennej. Vstupom do funkcie je signál s hodnotou, ktorú chceme zapísať. Týchto blokov môže byť v schéme ľubovoľný počet. Parametre S – funkcie sú reprezentované maskou a sú v princípe rovnaké ako tomu bolo v prípade ekvivalentnej MEX funkcie.





c. ADS read

Blok na čítanie premennej. Platí to isté ako v prípade bloku ADS write.



5. Demo skript a schéma

Pre účely demonštrovania funkcionality som napísal demo skript využívajúci všetky spomínané MEX- funkcie.

```
ADS_open_mex();

ventilator_name='GVL.ventilator';
spirala_name='GVL.spirala';
snimac_name='GVL.snimac';

for i=1:N

    t(i)=i*Tvz;
    u(i)=30000*spirala_in;
    ADS_write_mex([10, 3, 1, 138, 3, 1],ventilator_name,TC_INT_type,ventilator_in);
    ADS_write_mex([10, 3, 1, 138, 3, 1],spirala_name,TC_INT_type,u(i));
    y(i)=ADS_read_mex([10, 3, 1, 138, 3, 1],snimac_name,TC_INT_type);
    pause(Tvz);

end
ADS_write_mex([10, 3, 1, 138, 3, 1],ventilator_name,TC_INT_type,0);
ADS_write_mex([10, 3, 1, 138, 3, 1],spirala_name,TC_INT_type,0);
ADS_close_mex();
```

Podobne aj v prípade Simulinku a S-funkcii som vytvoril testovaciu schému obsahujúce spomínané bloky, ich využitie a správne nastavenie.