Solar Monitor s.r.o.

Knihovna SolarMonitorLib_2019

Historie změn

Datum	Vydání	Popis změn
Listopad 2017	1.0	První vydání, popis odpovídá SolarMonitorLib_2017
15.10.2019	1.1	Aktualizace podle knihovny V24, přidání příkladů, popisu funkčních bloků, dalších datových typů.
19.3.2020	1.2	Aktualizace podle knihovny V25, přidán popis nastavení komunikačního kanálu
8.6.2020	1.3	Aktualizace podle knihovny V26, doplněn popis chyb

Kontakt

https://www.solarmonitor.cz/cz/o-spolecnosti	Kontaktní údaje Solar Monitor s.r.o.
support@solarmonitor.cz	Emailová adresa zákaznické podpory

Obsah

1. Úvod

Knihovna *SolarMonitorLib_2019* obsahuje několik funkčních bloků pro práci s jednotkami SolarMonitor (SM). Tyto bloky komunikují s různými jednotkami pomocí komunikačního protokolu MODBUS. Každému funkčnímu bloku můžeme nastavit, zda bude komunikovat pomocí TCP nebo UDP protokolu.

Funkční blok (FB) fb_SMInit se vždy musí pustit jako první. FB zjistí z jednotky SM, jaká zařízení jsou k ní připojena, zjistí jaké funkčnosti tato zařízení podporují a tyto údaje pak poskytne dalším funkčním blokům. Úvodní zjišťování chvilku trvá, proto není vhodné ho zařazovat cyklicky, i když je to možné. Typicky se použije pouze jednou na začátku běhu programu.

Jedním z dalších funkčních bloků se načítají data ze zařízení (střídač, baterie, tracker, měřidla), která jsou nadetekována na jednotce SolarMonitor. Dalším blokem je možné individuálně řídit výkon na zařízeních. Třetím funkčním blokem je možné číst nebo zapisovat libovolný parametr, pokud tuto funkcionalitu zařízení podporuje. Funkční bloky Init a Fini slouží pro zahájení a ukončení komunikace se zařízení Solar Monitor.

Pokud chceme využít funkce z knihovny, je potřeba tuto knihovnu přidat do projektu, vytvořeného v programovacím prostředí Mosaic. Současně se s knihovnou do projektu přidá knihovna *MODBUSRTU_V36_20190430.MLB,* protože knihovna *SolarMonitorLib* využívá některé funkce z knihovny *MODBUSRTU*.

Ukázky použití jsou v jednotlivých příkladech v kapitole 6. Funkční bloky.

1.1 Přehled podporovaných zařízení

K 11 / 2019 byla k dispozici komunikace s následujícími výrobci. Jedná se o střídače, MPP trackery, bateriové systémy, měřidla a zařízení pro optimalizaci vlastní spotřeby. Knihovna umožňuje jak data číst, tak je i nastavovat, mj. tím lze realizovat řízení výkonu, optimalizaci ¼ hodinových maxim, nastavování nabíjecích proudů, povolit nabíjení baterie ze sítě, či naopak její vybíjení. Podle toho, co který konkrétní systém umožňuje.

1.2 Ke stažení

Knihovna i s příklady je ke stažení zde.

Tento popis je ke stažení jako SolarmonitorLib_2019.pdf.

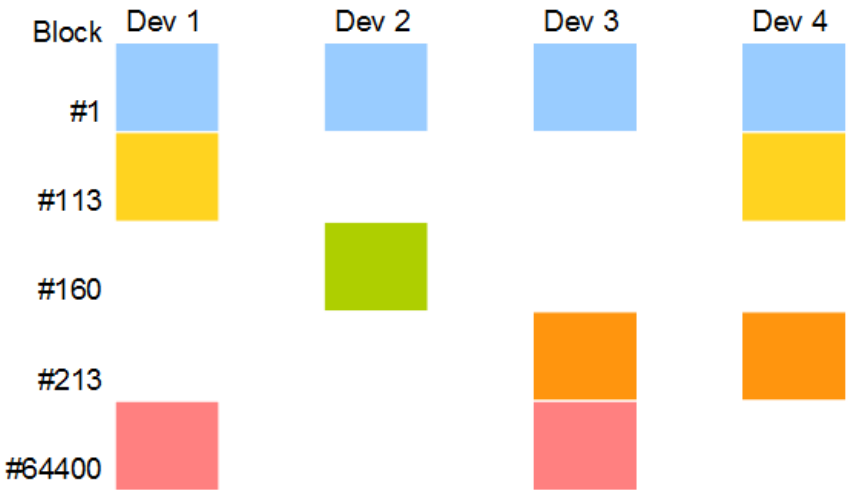
Prezentace s ukázkami je ke stažení jako Solar_Monitor – Automatizace_fotovoltaiky_v_praxi.ppt, od strany 6 je vysvětlen i přístup pomocí této knihovny.

Datový model je popsán zde, v tomto adresáři jsou i konkrétní popisy adesace Modbus registrů jednotlivých zařízení Studer-Innotec, včetně otestování utilitou Sunspec Dashboard.

1.3 Komunikace

Se zařízením Solar Monitor může knihovna komunikovat protokolem TCP či UDP. Protokol TCP je zabezpečený (proti ztrátě dat přenosem, nikoli hackery :-)), volíme ho tehdy, pokud je zařízení vzdáleno od PLC, např. přes VPN. Protokol UDP je rychlejší, spojení se zde nenavazuje, na 1 UDP port může současně komunikovat i více PLC či jiných M2M zařízení.

Hlavní předností nové knihovny je univerzalita komunikace pomocí Sunspec bloků, jednoduchost použití funkčními bloky v Mosaicu a malý overhead při komunikaci Modbusem oproti XML přes http.



Obr. 1: Každý barevný blok reprezentuje určitou vlastnost. Zařízení ji buď má nebo ne.

Každé zařízení, které je připojené k jednotce Solar Monitor (JSM) dostane přidělenu Modbus adresu postupně od adresy 1. Na začátku komunikace PLC s JSM funkční blok fb_SMInit „prohledá“ JSM na počet existujících zařízení a adresový prostor každého zařízení prohledá a zjistí jaké bloky zde existují a jakého jsou typu.

Při vyčítání dat pak z jednotlivých bloků načítá hodnoty proměnných. Při řízení výkonu, či nastavování individuálního parametru PLC program nejdříve najde blok, který tuto funkcionalitu má na starosti a pak komunikuje s proměnnými v rámci tohoto bloku.

1.3.1.1 Podpora systémů Studer-Innotec

Knihovna umožňuje i čtení / nastavování parametrů v systémech Studer-Innotec přímo až v konkrétním zařízení (např Xtender). Lze tak mj. nastavovat nabíjecí proudy, zda se může nabíjet ze sítě, či naopak dodávat do sítě, výši napětí, na kterou se baterie bude přednostně nabíjet.

Tuto komunikaci jsme vyvinuli ve spolupráci s firmou Studer-Innotec a je kompatibilní s rozhraním, poskytovaným protokolem SNMP.

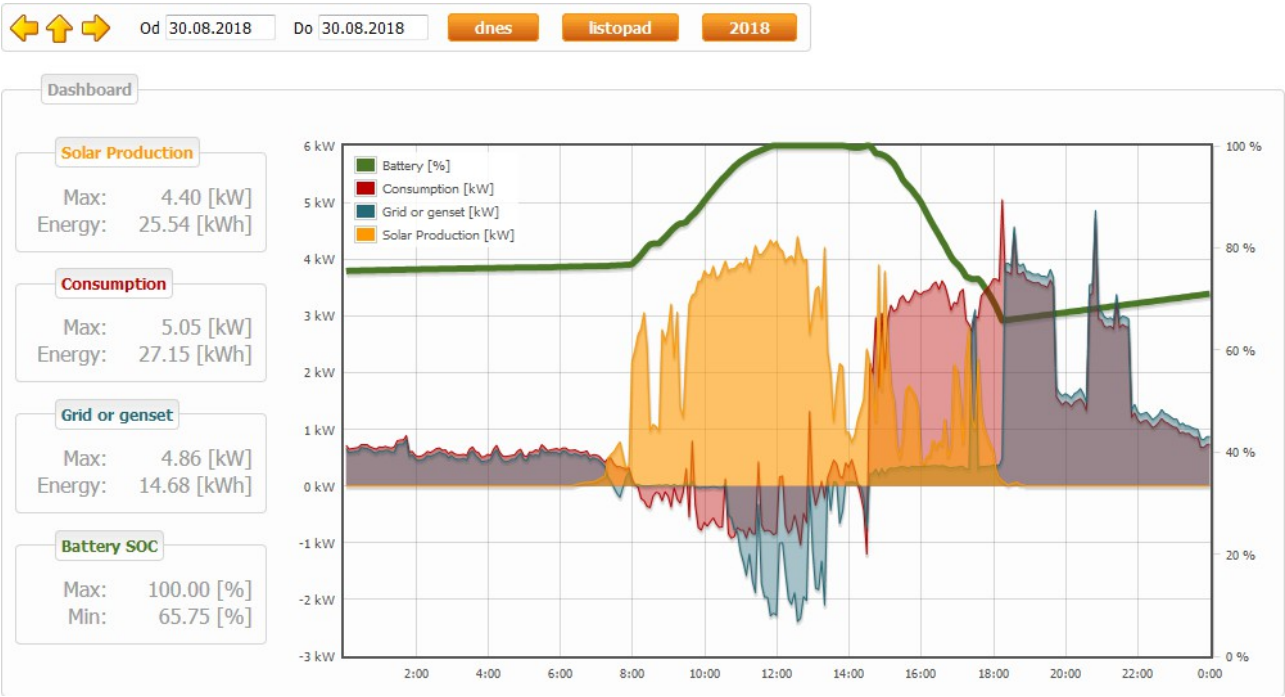
1.4 Bezpečnost

Protokol Modbus je nezabezpečený, toto je nutné vyřešit sekundárně. Buď používat komunikaci pouze v lokální síti nebo použít šifrování, které nabízí VPN, např. Open VPN či Ipsec.

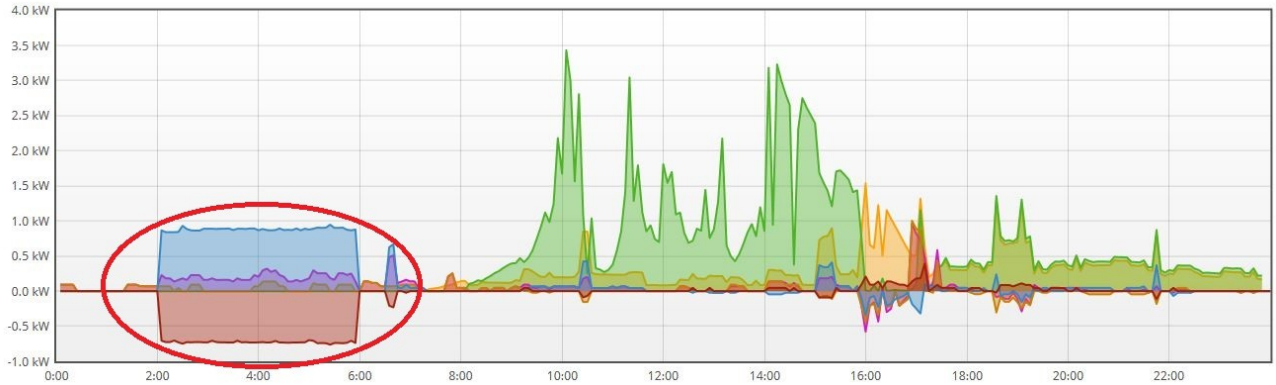
1.5 Ukládání dat do databáze, grafy

Zařízení Solar Monitor nejen data poskytuje, ale může je posílat pravidelně i do databáze na Internetu. Grafy jsou jednak předpřipravené a jednak je možné si tvořit i vlastní.

Ukázka několika grafů:

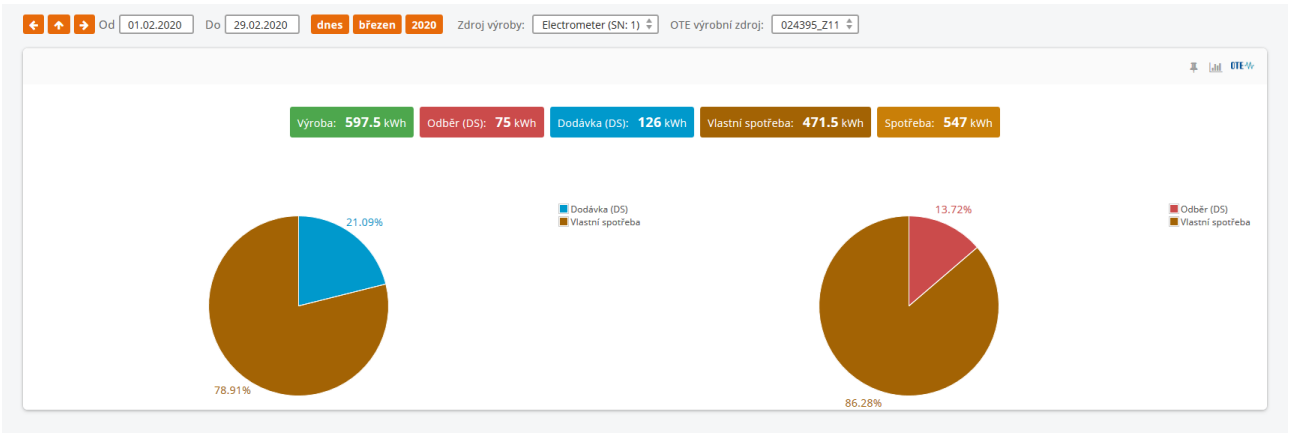


Obr. 2: Hybridní systém se stavem nabití baterie

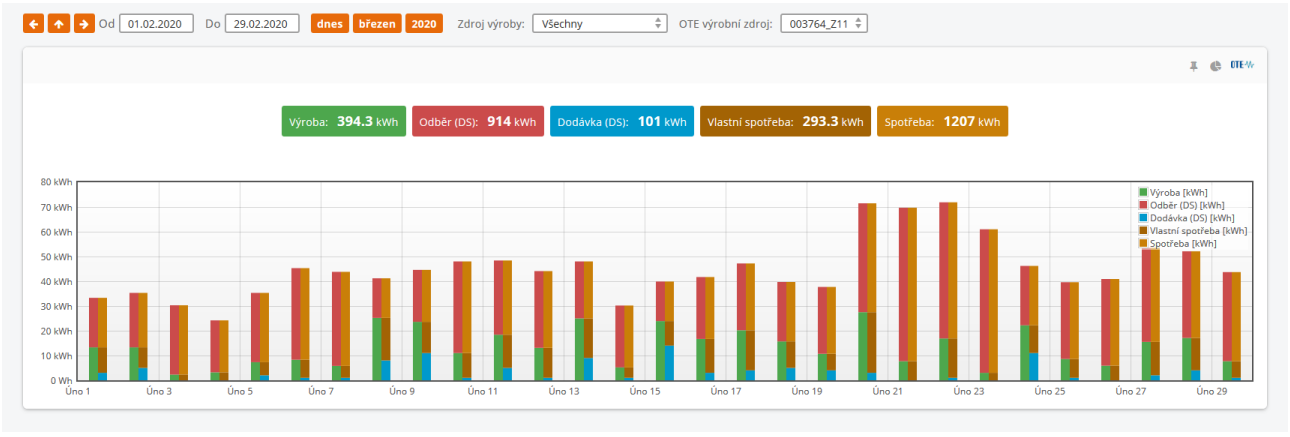


Obr. 3: Topení bojleru na noční proud, CT na 1 fázi obráceně zapojen

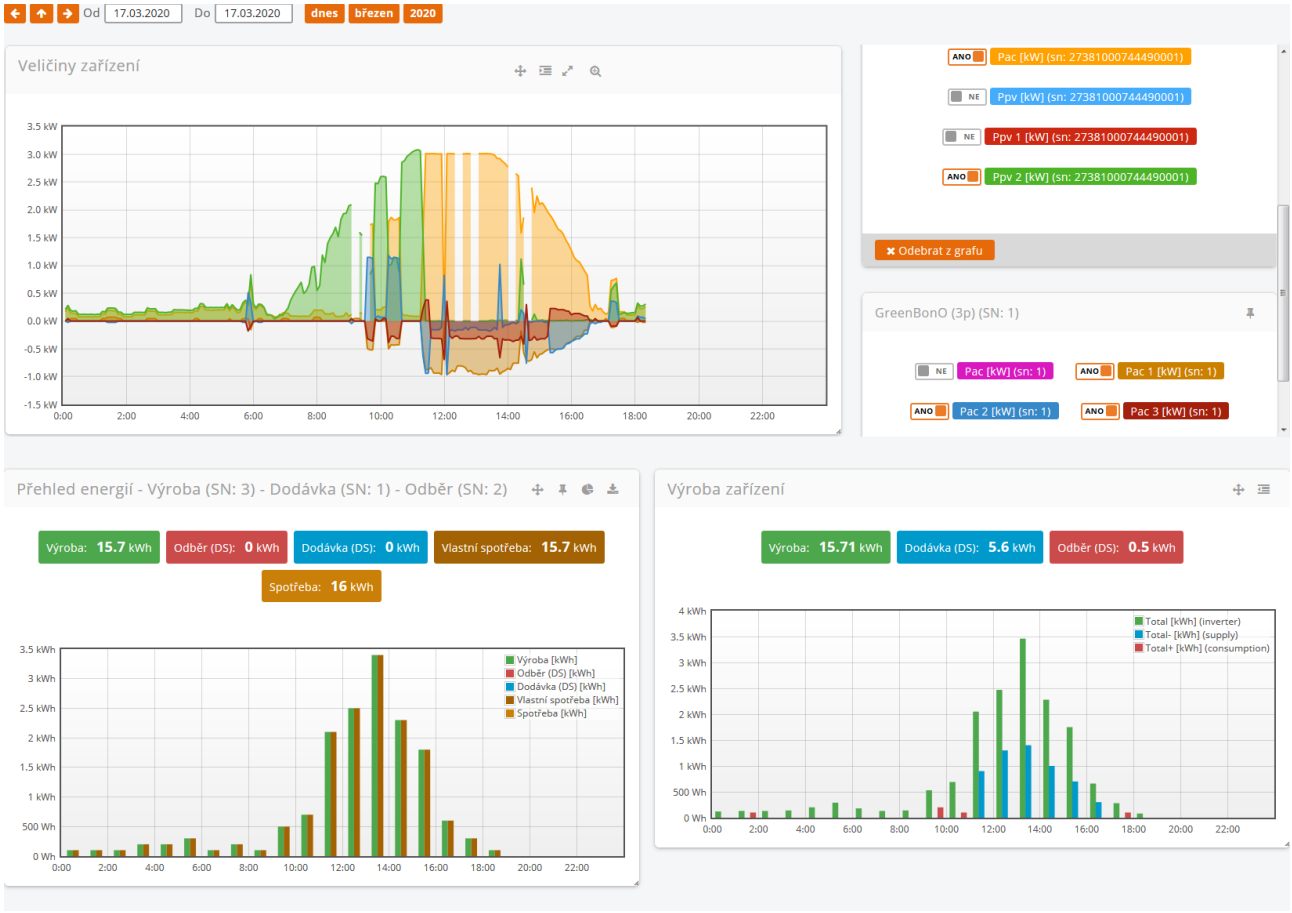
Obr. 4: Konfigurace grafu - střídač Fronius Symo Hybrid a GreenBonO současně



Obr. 5: Přehledný graf vlastní spotřeby a poměru odběru k vlastní spotřebě



Obr. 6: Kombinace výroby a spotřeby s odběrem a dodávkou, získanými ze serveru OTE



Obr. 7: Celková nástěnka s několika grafy, nahoře je vidět kdy se přestala nabíjet baterie a kdy se začalo dodávat do sítě

2. Nastavení

2.1 Komunikační brána SMx-MU

Protokol Modbus je nutné nejprve povolit v menu Nastavení / Modbus.

Detaily jsou vysvětleny na webu zde.

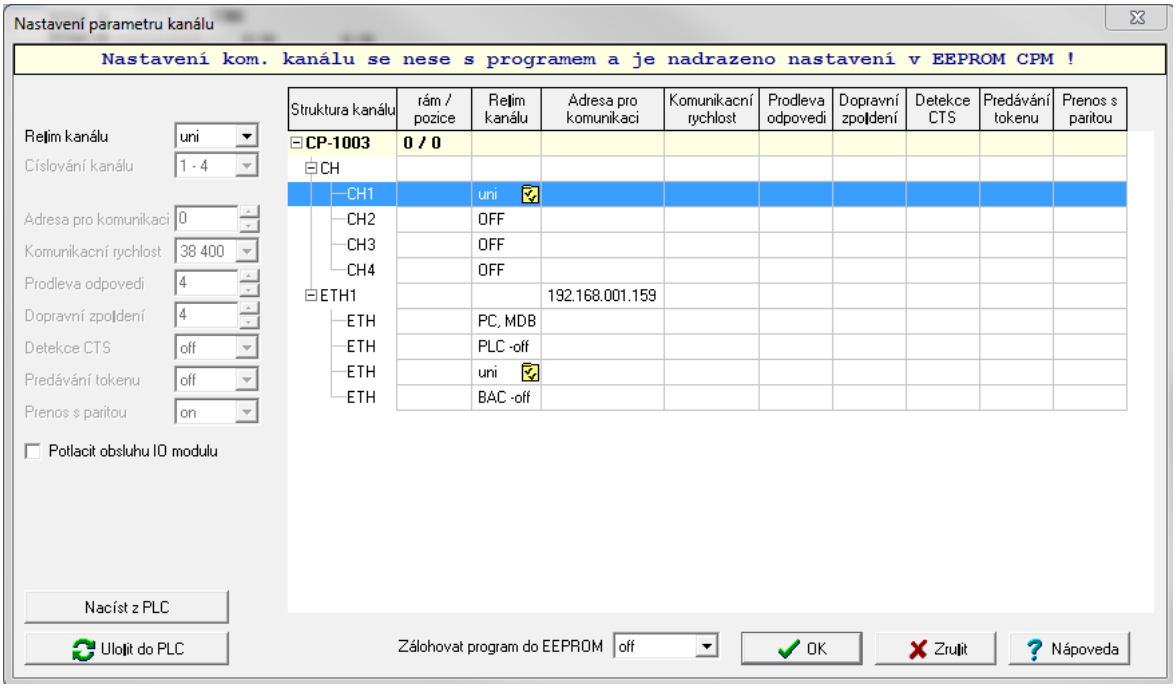
Konfigurace celého zařízení je buď na Wiki, video je možné zhlédnout na youtube zde.

2.2 PLC Foxtrot – nastavení komunikačního kanálu

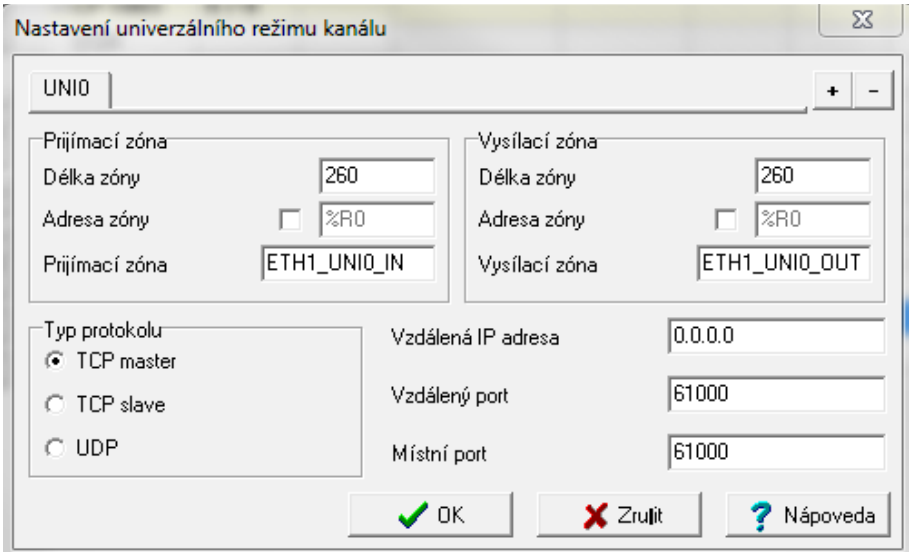
Dalším krokem je připojení PLC, do kterého budete Vámi vytvořený projekt nahrávat. Zde je důležité, abyste nastavili správnou konfiguraci pro Vaše PLC.

- *Projekt -> manažer projektu*
- *Typ pripojeni* – zde si uživatel vybere jak chce PLC připojit
  - např. máte PLC s adresou 192.168.1.159 a portem 61682, připojené přes lokální LAN (Ethernet)
- *HW -> výběr řady PLC* – zde zvolíte Vaše PLC (např. CP-1003)

- *HW -> konfigurace HW* – zde nastavíte několik parametrů (po kliknutí na obrázek složky vlevo od zeleného zaškrtávátka, řádek CPU)
  - nastavíte IP (ETH1) na adresu 192.168.1.159 a kliknete na tlačítko *Načíst z PLC*



  - měli byste vidět toto nastavení, jestliže ne, doplníte informace ručně
  - u záložky *ETH1 -> ETH -> uni* byste měli mít toto nastavení



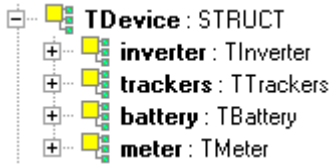
  - UNI0 je název komunikačního kanálu, který budete využívat
  - zde nastavíte, jestli bude PLC komunikovat přes Modbus přes TCP master nebo UDP
  - po dokončení uložíte konfiguraci do PLC a přitom se řídíte pokyny prostředí Mosaic

3. Datové typy

V PLC není možné použít dynamickou alokaci proměnných, proto byly vytvořeny datové typy s určitou redundancí. Pro optimalizaci použité paměti pracuje knihovna s ukazateli, co umožňuje v programu definovat proměnné typů TDevice a TDeviceMiB jako pole a jejich velikost dimenzovat podle potřeb konkrétní aplikace - viz příklady. To vyhoví většině projektů. Pro ještě větší optimalizaci by bylo možné, abychom knihovnu upravili.

Přehled datových struktur	
Název	**Popis**
TDevice	Základní struktura, která obsahuje hodnoty proměnných jednotlivých zařízení ve strukturách Tbattery, Tinverter, Ttrackers, Tmeter. V uživatelském programu se používá jako pole, které se dimenzuje podle potřeby konkrétní aplikace (projektu). Zařízení odpovídají zařízením, které byly před tím nadetekovány na jednotce Solar Monitor.
TDeviceTypeCount	Struktura, ve které jsou počty jednotlivých druhů nadetekovaných zařízení
TDeviceBlockInfo	Slouží pro informaci o tom, jaké objekty zařízení podporuje.
TPowerControl	Struktura pro informace o řízení výkonu
TInverter	Struktura s hodnotami vyčtenými ze střídačů
TTrackers	Struktura s jednotlivými trackery na daném zařízení (max. 3 trackery)
TTracker	Pomocná struktura s hodnotami vyčtenými z jednoho trackeru
TBattery	Struktura s hodnotami pro vyčítání baterií
TMeter	Struktura s hodnotami vyčtenými z měřidel
TParams	Struktura pro zápis/čtení speciálního bloku
TDeviceMiB	Rozšíření struktury TDevice o proměnné, poskytované snmp protokolem
TDeviceMiBTypeCount	Podobně jako TdeviceTypeCount, ale druhy jsou podle toho, jak jsou zařízení naimplementována pro snmp protokol
TXtenderMiB	Struktura pro zařízení Studer-Innotec Xtender
TVarioTrackerMiB	Struktura pro zařízení Studer-Innotec Variotrack
TVarioStringMiB	Struktura pro zařízení Studer-Innotec Variostring
TBatteryMiB	Struktura pro zařízení Studer-Innotec BSP

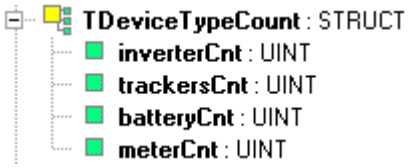
3.1 TDevice



Datová struktura, která obsahuje hodnoty vyčtené z jednotlivých zařízení. Obsahuje proměnné datových typů, které jsou popsány dále.

Pro každé zařízení, které je v Solar Monitoru nadetekováno, je použit jeden prvek pole typu TDevice. Podle typu zařízení pak tento prvek plní tu ze svých proměnných (struktur), která mu odpovídá. Ostatní struktury zůstanou nepoužité. Takto knihovna dokáže pracovat s různými typy zařízení.

3.2 TDeviceTypeCount



Struktura je po počátečním vyčtení (inicializaci) naplněna počtem jednotlivých druhů zařízení.

Název	Typ	Popis
inverterCnt	UINT	Počet nalezených zařízení typu střídač
trackersCnt	UINT	Počet nalezených zařízení typu sledovač (tracker)
batteryCnt	UINT	Počet nalezených zařízení typu baterie
meterCnt	UINT	Počet nalezených zařízení typu měřidlo

3.3 TDeviceBlockInfo

Interní struktura knihovny.

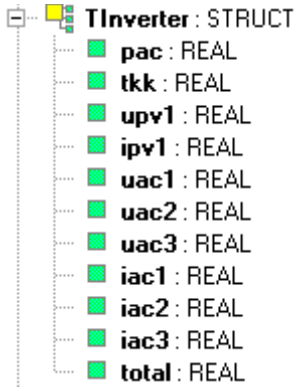
Datová oblast, kde si knihovna drží informace o nadetekovaných blocích pro jednotlivá zařízení. Tuto strukturu plní funkční blok 5.1 fb_SMInit. Všechny ostatní funkční bloky pak tuto strukturu používají pro komunikaci s jednotkou Solar Monitor.

Podle nalezených bloků jsou vyčítána data, při řízení výkonu je nejprve v této struktuře nalezen blok, který má tuto funkcionalitu na starosti, poté s ním knihovna komunikuje. Podobně pracuje i individuální nastavování parametrů.

Viz chyby SM_ERR_PWCTRL_BLOCK_MISSING a SM_ERR_RWPAR_BLOCK_MISSING.

3.4 TPowerControl

3.5 TInverter

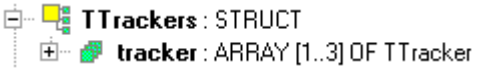


Datový typ *TInverter* je struktura, do které blok *fb_SMReadData* ukládá hodnoty získané ze střídačů. Jsou vybrány pouze základní proměnné, je možné tuto strukturu rozšířit podle potřeb vašeho projektu.

Pokud naopak dané zařízení proměnnou neobsahuje, bude její hodnota NaN (dle IEEE-754).

Název	Typ	Popis
iac1	REAL	Proud L1 [A]
iac2	REAL	Proud L2 [A]
iac3	REAL	Proud L3 [A]
uac1	REAL	Napětí L1 [V]
uac2	REAL	Napětí L2 [V]
uac3	REAL	Napětí L3 [V]
pac	REAL	Aktuální výkon [W]
fac	REAL	Frekvence sítě [Hz]
total	REAL	Celková výroba [kWh]
ipv	REAL	Vstupní proud [V]
upv	REAL	Vstupní napětí [V]
ppv	REAL	Aktuální výkon na vstupu [W]
tkk	REAL	Teplota střídače [°C]
pwr_ctl	TPowerControl	

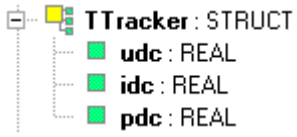
3.6 TTrackers



Podle počtu MPP trackerů je naplněno pole tracker.

Pozn.: Např. u střídače Fronius Symo Hybrid se tracker č. 2 používá pro práci s baterií.

3.7 TTracker



Datový typ *Ttracker* je struktura, do které blok *fb_SMReadData* ukládá hodnoty získané z MPP trackerů.

Pokud zařízení proměnnou neobsahuje, bude její hodnota NaN (dle IEEE-754).

Název	Typ	Popis
udc	REAL	Napětí trackeru [V]
idc	REAL	Proud trackeru [A]
pdc	REAL	Výkon trackeru [W]

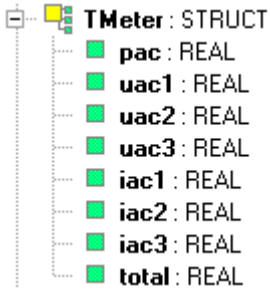
3.8 TBattery

Datový typ *TBattery* je struktura, do které blok *fb_SMReadData* ukládá hodnoty získané z baterií.

Pokud zařízení proměnnou neobsahuje, bude její hodnota NaN (dle IEEE-754).

Název	Typ	Popis
inBatV	REAL	Napětí baterie [V]
SOC	REAL	Stav baterie [%]

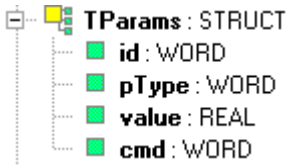
3.9 TMeter



Datový typ *TMeter* je struktura, do které blok *fb_SMReadData* ukládá hodnoty získané z měřidel.

Název	Typ	Popis
pac	REAL	Aktuální výkon [W]
uac1	REAL	Napětí L1 [V]
uac2	REAL	Napětí L2 [V]
uac3	REAL	Napětí L3 [V]
iac1	REAL	Proud L1 [A]
iac2	REAL	Proud L2 [A]
iac3	REAL	Proud L3 [A]
total	REAL	Celková výroba [kWh]

3.10 TParams



Datový typ *TParams* je struktura, do které uživatel zadává parametry pro čtení/zápis libovolného parametru z / do zařízení. Práci s touto strukturou zapouzdřuje funkční blok *fb_SMReadWriteParameter.*

Pro čtení je zadán identifikátor, typ parametru a cmd = READ. Následně je prvek cmd aktualizován a akce končí tehdy, až je přečten cmd = 0. Přečtená hodnota je v prvku value.

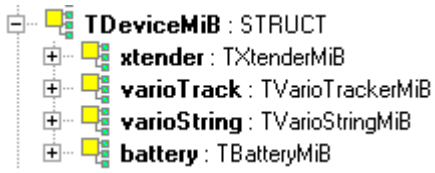
Podobně při zápisu je zadán identifikátor, typ parametru, zapisovaná hodnota a cmd = WRITE. Proměnnou cmd opakovaně čteme, v okamžiku kdy je cmd = 0, hodnota je zapsána do zařízení, přípojeného k jednotce Solar Monitor.

Více v popisu funkčního bloku a v popisu příkladu.

Název	Typ	Popis

id	WORD	Identifikátor parametru
pType	WORD	Typ parametru – viz kapitola 4 Konstanty
value	REAL	Přečtená/zapisovaná hodnota
cmd	WORD	Řídící příkaz (READ, WRITE)

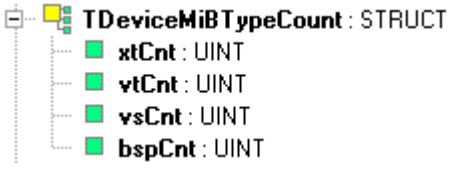
3.11 TDeviceMiB



Podobně jako TDevice, obsahuje data ze zařízení a podle jejich druhu je naplněna vždy jen 1 struktura.

Proměnné v jednotlivých strukturách odpovídají proměnným, definovaným v MIB tabulce snmp protokolu. Vlastností protokolu snmp je proměnný počet záznamů v tabulce jednotlivého druhu zařízení. V Modbusu jsme tuto vlastnost implementovali proměnným počtem zařízení, kterým jsou přidělovány Modbus adresy (zařízení) podle toho, jak jsou na sběrnicích nalezena.

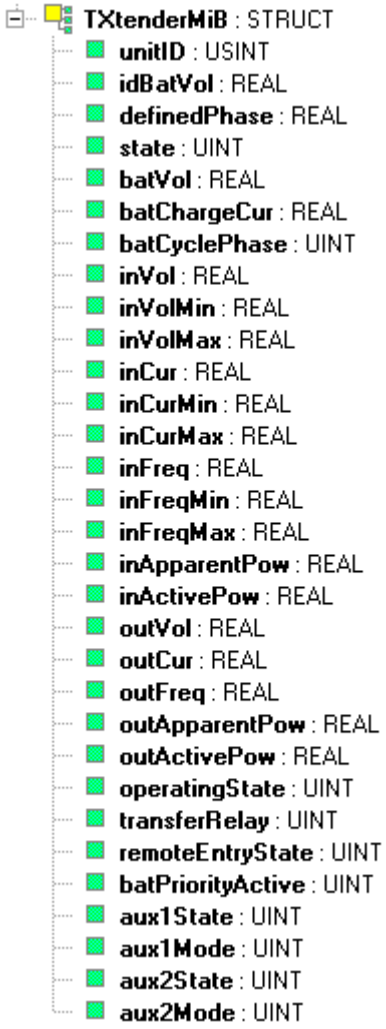
3.12 TDeviceMiBTypeCount



Podobně jako TdeviceMiBTypeCount, je i tato struktura po počátečním vyčtení (inicializaci) naplněna počtem jednotlivých druhů zařízení. Druhy jsou tentokrát podle toho, jak jsou implementovány v snmp protokolu.

Název	Typ	Popis
xtCnt	UINT	Počet Xtenderů
vtCnt	UINT	Počet Variotracků
vsCnt	UINT	Počet Variostringů
bspCnt	UINT	Počet BSP

3.13 TXtenderMiB



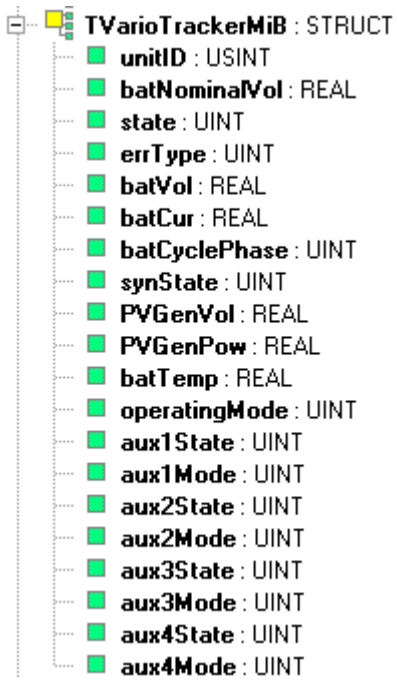
Datový typ *TXtenderMiB* je struktura, do které blok *fb_SMReadData* ukládá hodnoty získané ze střídačů.

Pokud zařízení proměnnou neobsahuje, bude její hodnota NaN (dle IEEE-754).

Název	Typ	Popis
unitID	USINT	XT FID
idBatVol	REAL	ID Battery Voltage
definedPhase	REAL	Defined Phase
state	UINT	State
batVol	REAL	Battery Voltage
batChargeCur	REAL	Battery Charge Current
batCyclePhase	UINT	Battery Cycle Current
inVol	REAL	Input Voltage
inVolMin	REAL	Input Voltage Min

Název	Typ	Popis
inVolMax	REAL	Input Voltage Max
inCur	REAL	Input Current
inCurMin	REAL	Input Current Min
inCurMax	REAL	Input Current Max
inFreq	REAL	Input Frequency
inFreqMin	REAL	Input Frequency Min
inFreqMax	REAL	Input Frequency Max
inApparentPow	REAL	Input Apparent Power
inActivePow	REAL	Input Active Power
outVol	REAL	Output Voltage
outCur	REAL	Output Current
outFreq	REAL	Output Frequency
outApparentPow	REAL	Output Apparent Power
outActivePow	REAL	Output Active Power
operatingState	UINT	Operating State
transferRelay	UINT	State of Transfer Relay
remoteEntryState	UINT	Remote Entry State
batPriorityActive	UINT	Battery Priority Active
aux1State	UINT	State of Auxillary Relay 1
aux1Mode	UINT	Relay AUX 1 Mode
aux2State	UINT	State of Auxillary Relay 2
aux2Mode	UINT	Relay AUX 2 Mode

3.14 TVarioTrackerMiB



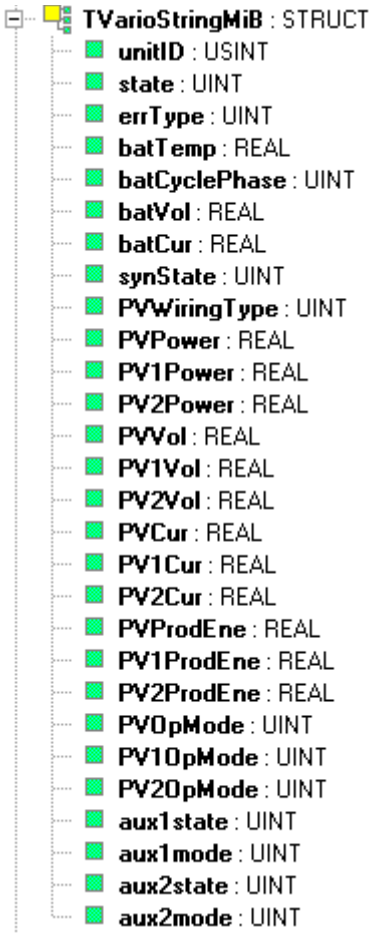
Datový typ *TVarioTrackerMiB* je struktura, do které blok *fb_SMReadData* ukládá hodnoty získané z Variotrack MPP trackerů.

Pokud zařízení proměnnou neobsahuje, bude její hodnota NaN (dle IEEE-754).

Název	Typ	Popis
unitID	USINT	VT FID
batNominalVol	REAL	Battery Nominal Voltage
state	UINT	State
errType	UINT	Error Type
batVol	REAL	Battery Voltage
batCur	REAL	Battery Current
BatCyclePhase	UINT	Battery Cycle Phase
synState	UINT	Synchronization State
PVGenVol	REAL	Voltage of the PV generator
PVGenPow	REAL	Power of the PV generator
batTemp	REAL	Battery Temperature
operatingMode	UINT	Operating Mode
aux1State	UINT	State of Auxillary Relay 1
aux1Mode	UINT	Relay AUX 1 Mode

Název	Typ	Popis
aux2State	UINT	State of Auxillary Relay 2
aux2Mode	UINT	Relay AUX 2 Mode
aux3State	UINT	State of Auxillary Relay 3
aux3Mode	UINT	Relay AUX 3 Mode
aux4State	UINT	State of Auxillary Relay 4
aux4Mode	UINT	Relay AUX 4 Mode

3.15 TvarioStringMiB

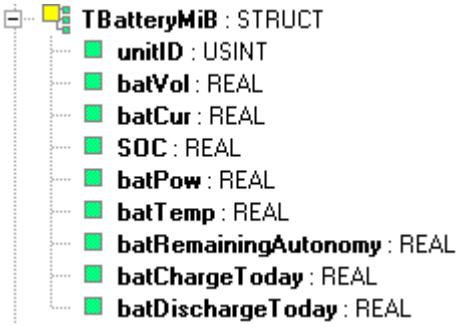


Datový typ *TVarioStringMiB* je struktura, do které blok *fb_SMReadData* ukládá hodnoty získané z Variostring MPP trackerů. Od Variotrack zařízení se liší především 2 nezávislými MPP tracker vstupy.

Pokud zařízení proměnnou neobsahuje, bude její hodnota NaN (dle IEEE-754).

Název	Typ	Popis
unitID	USINT	VS FID
state	UINT	Battery Nominal Voltage
errType	UINT	Error Type
batTemp	REAL	Battery Temperature
batCyclePhase	UINT	Battery Cycle Phase
batVol	REAL	Battery Voltage
batCur	REAL	Battery Current
synState	UINT	Synchronization State
PVWiringType	UINT	PV Type of Wiring
PVPower	REAL	PV Power
PV1Power	REAL	PV Power 1
PV2Power	REAL	PV Power 2
PVVol	REAL	PV Voltage
PV1Vol	REAL	PV Voltage 1
PV2Vol	REAL	PV Voltage 2
PVCur	REAL	PV Current
PV1Cur	REAL	PV Current 1
PV2Cur	REAL	PV Current 2
PVProdEne	REAL	PV Produced Energy
PV1ProdEne	REAL	PV Produced Energy 1
PV2ProdEne	REAL	PV Produced Energy 2
PV0pMode	UINT	PV Operating Mode
PV10pMode	UINT	PV 1 Operating Mode
PV20pMode	UINT	PV 2 Operating Mode
aux1state	UINT	Relay AUX 1 Mode
aux1mode	UINT	State of Auxillary Relay 1
aux2state	UINT	State of Auxillary Relay 2
aux2mode	UINT	Relay AUX 2 Mode

3.16 TBatteryMiB



Datový typ *TBatteryMiB* je struktura, do které blok *fb_SMReadData* ukládá hodnoty získané z BSP (Battery Status Processor).

Pokud zařízení proměnnou neobsahuje, bude její hodnota NaN (dle IEEE-754).

Název	Typ	Popis
unitID	USINT	BSP FID
batVol	REAL	Battery Voltage
batCur	REAL	Battery Current
SOC	REAL	State Of Charge
batPow	REAL	BSP Power
batTemp	REAL	Battery Temperature
batRemainingAutonomy	REAL	Remaining Autonomy
batChargeToday	REAL	Charged Today
batDischargeToday	REAL	Discharge Today

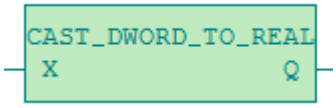
4. Konstanty

Název	Typ	Hodnota	Popis
Protokol MODBUS			
MODBUS_SLAVE_FAIL	USINT	140	Chyba při pokusu o navázání spojení se zařízením, které není připojené
MODBUS_START_ADR	UINT	40000	Počáteční MODBUS adresa
Blok PARAMETER			
PARAM_BLOCK	UINT	64400	ID parametrického bloku
PARAM_WRITE	WORD	16#10	Příkaz pro zápis params bloku
PARAM_READ	WORD	16#20	Přikaz pro čtení params bloku
PARAM_INT	WORD	0	Typ parametru INT
PARAM_ENUM	WORD	1	Typ parametru ENUM
PARAM_BOOL	WORD	2	Typ parametru BOOT
PARAM_FLOAT	WORD	3	Typ parametru FLOAT
PARAM_STRING	WORD	4	Typ parametru STRING
Blok řízení výkonu (POWER CONTROL)			
BLOCK_PC	UINT	123	Číslo bloku pro nastavení řízení výkonu (Power Control)
OSTATNÍ			
MAXIMUM_OF_BLOCKS	UINT	10	Maximální podporovaný počet bloků v zařízení *(konstanta knihovny)*
MAXIMUM_OF_TRACKERS	UINT	3	Maximální počet MPP trackerů na 1 zařízení *(konstanta knihovny)*
SM_ERR_PWCTRL_BLOCK_MISSING	USINT	SM_ERR-BASE +2	Implementace zařízení v jednotce Solar Monitor nepodporuje blok řízení výkonu
SM_ERR_RWPAR_BLOCK_MISSING	USINT	SM_ERR-BASE +1	Implementace zařízení v jednotce Solar Monitor nepodporuje blok individuálních parametrů
UNIT_ID_START	USINT	1	Identifikátor prvního zařízení

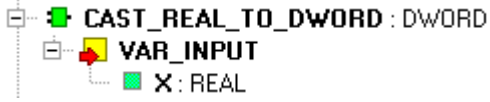
5. Funkce

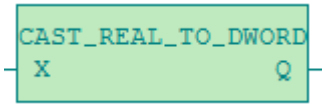
Základní programové jednotky		
Název	**Typ**	**Popis**
CAST_DWORD_TO_REAL	Funkce	Převod proměnné z formátu, ve kterém je přenášena po síti, do interního formátu PLC
CAST_REAL_TO DWORD	Funkce	Převod opačným směrem
GET_REAL	Funkce	Pomocná funkce používaná
ROL_DWORD	Funkce	Pomocná funkce pro tzv. Swapování 16 bitů (LSW a MSW)

5.1 CAST_DWORD_TO_REAL

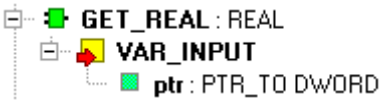


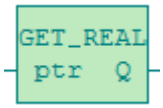
5.2 CAST_REAL_TO DWORD



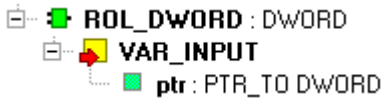


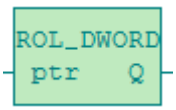
5.3 GET_REAL



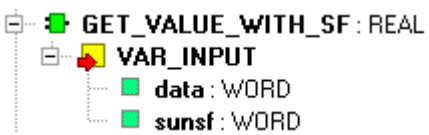


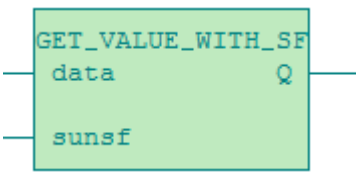
5.4 ROL_DWORD





5.5 GET_VALUE_WITH_SF





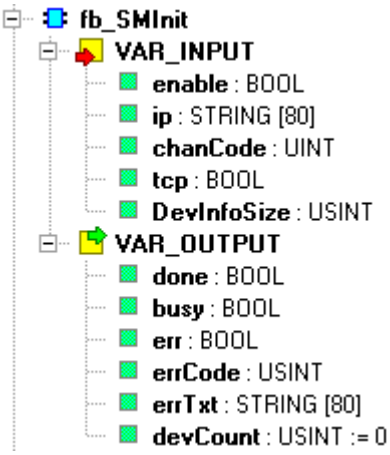
Funkce jsou používané knihovnou *SolarMonitorLib interně a nepředpokládá se jejich použití v aplikačním programu.*

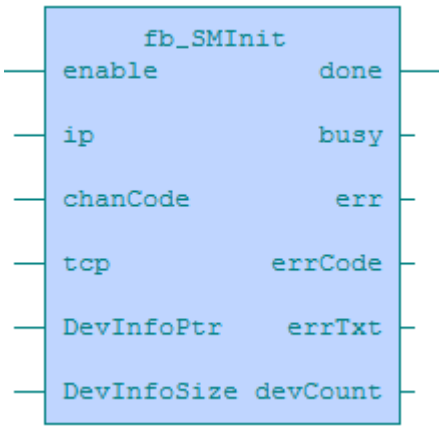
6. Funkční bloky

Nejsnadněji lze práci funkčních bloků knihovny porozumět na příkladech, které jsou ke stažení zde. Archiv se do prostředí Mosaic obnoví jako skupina projektů.

Základní programové jednotky		
Název	**Typ**	**Popis**
fb_SMInit	Funkční blok	Inicializace knihovny. Slouží pro zjištění počtu a druhu zařízení. Tento blok je nutné zavolat vždy na začátku komunikace PLC s EG. PLC si „osáhne“ zařízení, enumeruje Sunspec bloky v zařízení a naplní datové struktury pro pozdější použití.
fb_SMFini	Funkční blok	Ukončení spojení se zařízením Solar Monitor. S novější verzí knihovny *MODBUSRTU* by se již tento blok možná nemusel používat.
fb_SMPowerControl	Funkční blok	Je určen pro řízení výkonu střídačů. Např. v úloze řízení ¼ h maxim přetoku, v případě, že všechny tepelné spotřebiče jsou natopeny a bateriové úložiště je nabito (lze řídit např. blokem fb_SMReadWriteParameter). Když již nemám kam elektřinu posílat či ukládat, zbývá již jen jediná možnost - snížit okamžitý výkon střídačů. To Pak lze učinit pomocí tohoto bloku a pokud se změní podmínky, mohu pak výkon opět zvýšit na maximum.
fb_SMReadData	Funkční blok	Blok pro čtení dat do struktur TDevice.
fb_SMReadStuderMiBData	Funkční blok	Blok pro čtení dat do struktur TDeviceMiB.
fb_SMReadWriteParameter	Funkční blok	Slouží ke čtení / zápisu libovolného parametru. Funkční blok zapouzdřuje následující Modbus komunikaci. Nejprve je zapsána adresa proměnné a její typ, případně hodnota k zápisu. Poté se iniciuje vykonání příkazu a testuje se, zda se příkaz již vykonal. Zařízení Solar Monitor mezitím komunikuje (proxy gateway) s připojenou technologií. Funkční blok oznámí dokončení operace na svém výstupu done.

6.1 Funkční blok „fb_SMInit





Se zařízením Solar Monitor můžeme komunikovat přes TCP či UDP viz kapitola 1.3. Tuto volbu nastavujeme parametrem tcp. Dále nastavíme IP adresu číslo komunikačního kanálu a ukazatel na pole, do kterého funkční blok uloží informaci o komunikačních blocích jednotlivých zařízení.

Nastavením vstupu enable na true funkční blok začne provádět svou činnost, kterou můžeme kdykoli přerušit shozením vstupu enable do false. Typicky se blok používá s trvale nastaveným vstupem – důležitý je přechod z false na true (náběžná hrana), výstup done je pulzní. Tímto způsobem pracují všechny funkční bloky knihovny.

Pokud potřebujeme výstupním pulzem spustit další funkční blok, např. čtení dat, je nutné použít SR klopný obvod.

	Název	Typ	Popis
Vstupy	enable	BOOL	Povolení činnosti funkčního bloku
	ip	STRING	IP adresa jednotky, na kterou budeme provádět zápis/čtení
	chanCode	UINT	Číslo komunikačního kanálu
	tcp	BOOL	Typ protokolu (TRUE=TCP FALSE=UDP)
	DevInfoPtr	PTR_TO TDeviceBlockInfo	Ukazatel na pole struktur TDeviceBlockInfo
	DevInfoSize	USINT	Velikost tohoto pole
Výstupy	done	BOOL	Vyčítání ukončeno, pulzní výstup
	busy	BOOL	Probíhá vyčítání
	err	BOOL	Vyčítání skončilo chybou, pulzní výstup
	errCode	USINT	Chybový kód
	errTxt	STRING	Text chyby, který odpovídá errCode
	devCount	USINT	Počet nalezených zařízení. Odpovídá i počtu obsazených prvků pole, na který ukazuje ukazatel DevInfoPtr Pozn.: Hodnotu tohoto výstupu lze použít

			jako informaci o počtu zařízení, které jsou v jednotce SM2-MU nadetekovány

6.1.1 Příklad – FB: 11-jednoduchy_priklad_fb

Ukázka použití funkčního bloku je ve většině příkladů, jejichž název končí „_fb“.

Níže je screenshot, jak vypadá použití funkčního bloku v programové jednotce typu FB.

Informace o blocích se budou plnit do pole DeviceInfo, velikost tohoto pole určuje konstanta DEVICE_COUNT.

Funkční blok komunikuje s jednotkou Solar Monitor a provádí vyčítání hodnot jednotlivých zařízení, která jsou na jednotce nadetekována. Nejdříve se vyčtou z jednotky všechna zařízení a jejich jednotlivé bloky (pouze id bloku a velikost) - *fb_SMInit*. Poté se z daných bloků vyčtou potřebná data, která se uloží do struktury *TDevice* – *fb_SMReadData*.

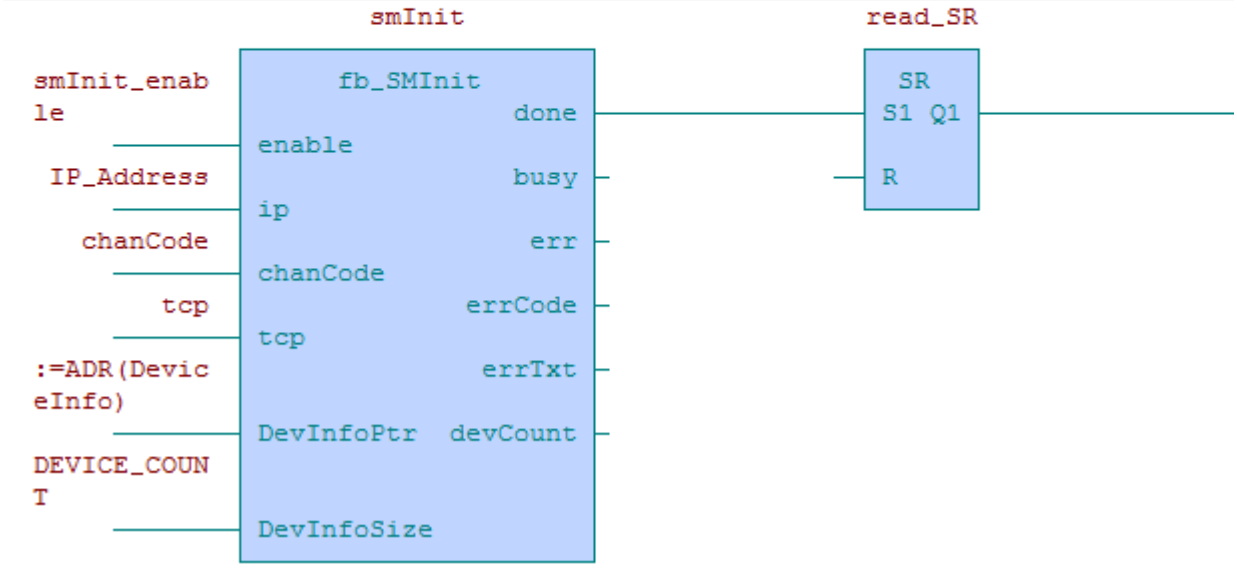
Po celou dobu, co *fb_SMInit* pracuje, je nastavena proměnná *busy* na TRUE. Když vyčítání dat skončí úspěšně, nastaví se hodnota proměnné *busy* na FALSE a hodnota proměnné *done* na TRUE. V případě, že vyčítání skončilo chybou, nastaví se proměnná *err* na hodnotu TRUE. V jednom okamžiku může být nastavena hodnota TRUE pouze na jedné z proměnných *busy*, *done* a *err*. Načtení dat z jednotky Solar Monitor trvá několik cyklů PLC.

Uživatel má dvě možnosti vyčtení dat:

1. Vyčítání dat se provede jednou
2. Vyčítání dat se provádí periodicky (uživatel si nastaví časovač, který daný blok periodicky spouští) – viz např. 6.4.1

Dále má uživatel možnost zvolit si jaký protokol (TCP / UDP) bude pro komunikaci využívat.

```
  Schema precte udaje ze zarizeni, ktera jsou pripojena k jednotce SM2-MU.

  Vysledky jsou v poli Device[], informace o datovych blocich
  v jednotlivych zarizenich v poli DeviceInfo[].
0001
  Pocatecni detekce pritomnych zarizeni.

  Oba bloky potrebuji nabeznou hranu a pak enable=1 po celou dobu sve prace.
  Klopny obvod je zde po podrzeni signalu done, ktery je pouze pulzni.
```



6.1.2 Příklad – ST: 1-jednoduchy_priklad

Stejný funkční blok je použit v programové jednotce, napsané v jazyce strukturovaného textu (ST).

```
//// Ukazka jednoducheho pouziti knihovny - inicializace

VAR_GLOBAL CONSTANT
  DEVICE_COUNT : USINT := 7;
END_VAR

PROGRAM prgMain
  VAR CONSTANT
    IP_Address : STRING := '192.168.1.117:502';
    chanCode : UINT := ETH1_uni0;
    tcp : BOOL := true;
  END_VAR

  VAR
    DeviceInfo : ARRAY [1 .. DEVICE_COUNT] OF TDeviceBlockInfo;

    smInit_enable : BOOL := true;

    smInit : fb_SMInit;
    smFini : fb_SMFini;
  END_VAR

  // pocatecni detekce pritomnych zarizeni
  smInit(enable := smInit_enable, ip := IP_Address, tcp := tcp, chanCode := chanCode,
DevInfoPtr := ADR(DeviceInfo), DevInfoSize := DEVICE_COUNT);

  IF smInit.done THEN
    smInit_enable := false;
    smFini(tcp := tcp);
  END_IF;

END_PROGRAM
```

6.2 Funkční blok „fb_SMFini“

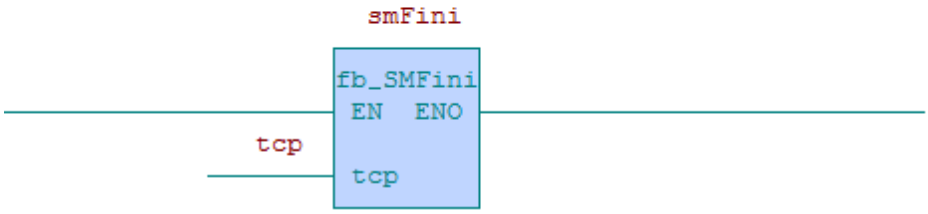
Ukončení spojení se zařízením Solar Monitor. S novější verzí knihovny MODBUSRTU by se již tento blok možná nemusel používat.

	Název	Typ	Popis
Vstupy	tcp	BOOL	Typ protokolu (TRUE=TCP FALSE=UDP)

6.2.1 Příklad – FB: 11-jednoduchy_priklad_fb

Ukázka použití funkčního bloku je ve většině příkladů, jejichž název končí „_fb“.

```
  Schema precte udaje ze zarizeni, ktera jsou pripojena k jednotce SM2-MU.

  Vysledky jsou v poli Device[], informace o datovych blocich
  v jednotlivych zarizenich v poli DeviceInfo[].
0001
Zavreni TCP spojeni.
```

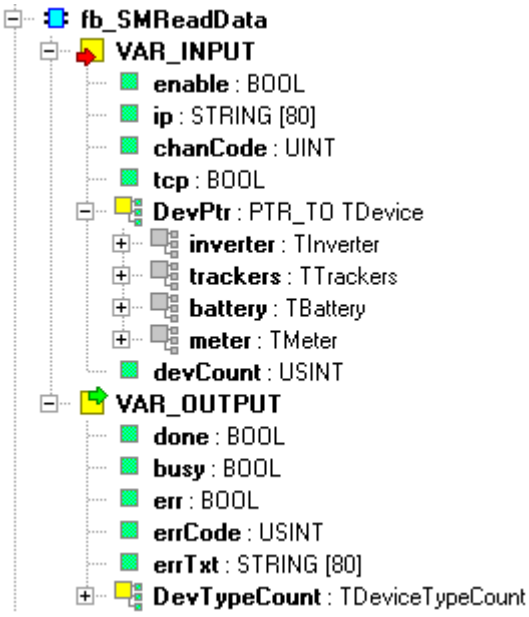


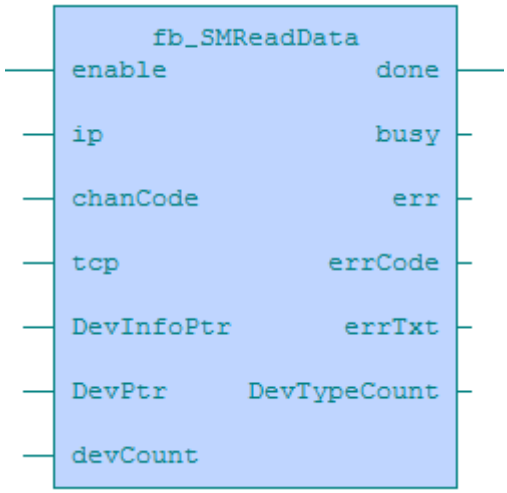
6.2.2 Příklad – ST: 1-jednoduchy_priklad

Podobně i ve strukturovaném textu.

```
//// Ukazka jednoducheho pouziti knihovny - inicializace

VAR_GLOBAL CONSTANT
  DEVICE_COUNT : USINT := 7;
END_VAR

PROGRAM prgMain
  VAR CONSTANT
    IP_Address : STRING := '192.168.1.117:502';
    chanCode : UINT := ETH1_uni0;
    tcp : BOOL := true;
  END_VAR

  VAR
    DeviceInfo : ARRAY [1 .. DEVICE_COUNT] OF TDeviceBlockInfo;

    smInit_enable : BOOL := true;

    smInit : fb_SMInit;
    smFini : fb_SMFini;
  END_VAR

  // pocatecni detekce pritomnych zarizeni
  smInit(enable := smInit_enable, ip := IP_Address, tcp := tcp, chanCode := chanCode,
DevInfoPtr := ADR(DeviceInfo), DevInfoSize := DEVICE_COUNT);

  IF smInit.done THEN
    smInit_enable := false;
    smFini(tcp := tcp);
  END_IF;

END_PROGRAM
```

6.3 Funkční blok „fb_SMReadData“





Většina parametrů je nám již známa z kapitoly 6.1, parametr DevPtr je ukazatel na pole struktur TDevice, kde budou po skončení tohoto bloku vyčtené hodnoty proměnných jednotlivých zařízení.

Nastavením vstupu enable na true funkční blok začne provádět svou činnost, kterou můžeme kdykoli přerušit shozením vstupu enable do false. Typicky se blok používá s trvale nastaveným vstupem – důležitý je přechod z false na true (náběžná hrana), výstup done je pulzní. Tímto způsobem pracují všechny funkční bloky knihovny.

Pokud potřebujeme výstupním pulzem spustit další funkční blok, např. čtení dat, je nutné použít SR klopný obvod.

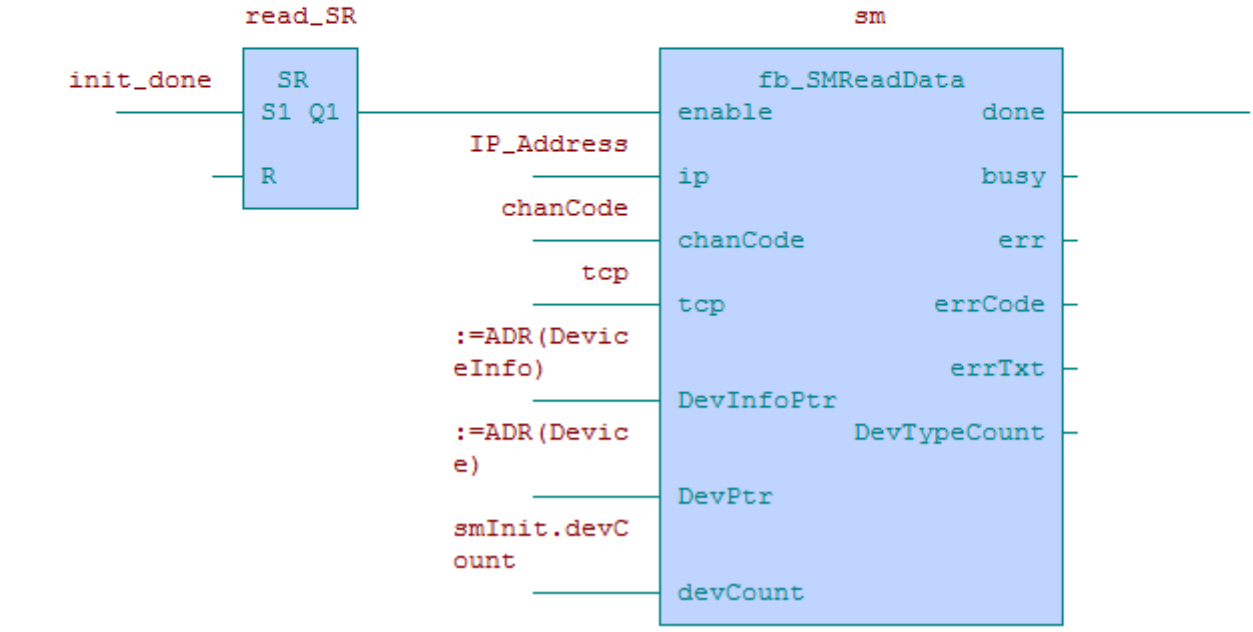
	Název	Typ	Popis
Vstup	enable	BOOL	Parametr pro spuštění bloku
	ip	STRING	IP adresa jednotky, které se bude nastavovat hodnota výkonu
	chanCode	UINT	Číslo komunikačního kanálu
	tcp	BOOL	Typ protokolu (TRUE=TCP FALSE=UDP)
	DevInfoPtr	PTR_TO TDeviceBlockInfo	Ukazatel na pole struktur TDeviceBlockInfo
	DevPtr	PTR_TO TDevice	Ukazatel na pole struktur TDevice
	devCount	USINT	Počet nadetekovaných zařízení (výstupní parametr devCount

			z fb_SMInit)
Výstup	done	BOOL	Vyčítání ukončeno, pulzní výstup
	busy	BOOL	Probíhá vyčítání
	err	BOOL	Vyčítání skončilo chybou, pulzní výstup
	errCode	USINT	Chybový kód
	errTxt	STRING	Text chyby, který odpovídá errCode
	DevTypeCount	UINT	Struktura, která je naplněna počty jednotlivých druhů zařízení – např. 3 střídače, 2 měřidla.

6.3.1 Příklad – FB: 12-cteni_dat_1x_fb

Funkční blok navazuje na fb_SMInit. Po jeho vykonání prohledá informační bloky v poli DeviceInfo a podle toho vyčte hodnoty přednastavených proměnných a naplní je do pole Device.

```
Schema precte udaje ze zarizeni, ktera jsou pripojena k jednotce SM2-MU.
Rozsirene vycitani zjisti pripadne jednotky Studer-Innotec
a vycte z nich detailnejsi info (do pole DeviceMiB).

Vysledky jsou v polich Device[] a DeviceMiB[],
informace o datovych blocich v jednotlivych zarizenich v poli DeviceInfo[].
0002
Jeden cteci cyklus - nejprve zakladni blok cteni.

Oba bloky potrebuji nabeznou hranu a pak enable=1 po celou dobu sve prace.
Klopne obvody jsou zde po podrzeni signalu done, ktery je pouze pulzni.
```



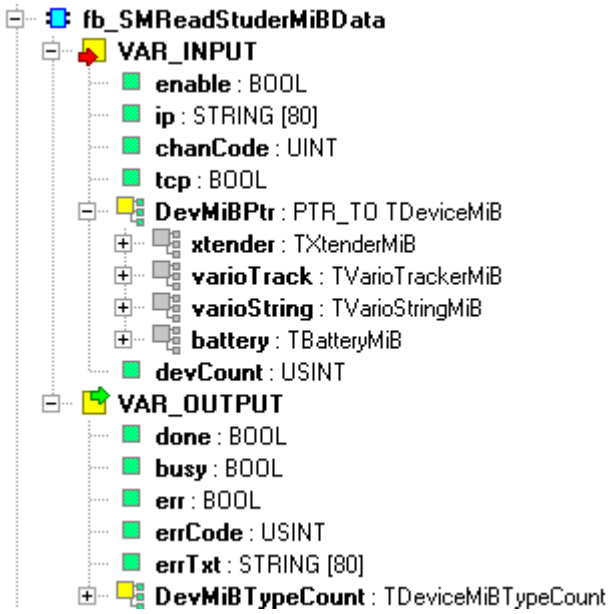
6.3.2 Příklad – ST: 2-cteni_dat_1x

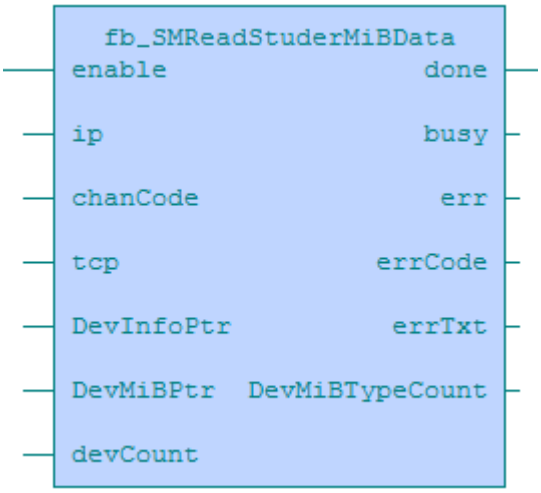
Ve strukturovaném textu to vypadá takto.

V ukázce níže jsou pouze relevantní řádky, plnou verzi si můžete stáhnout – viz Chyba: zdroj odkazu nenalezen.

```
//// Ukazka jednorazoveho nacteni dat - jak zakladni, tak pripadne i rozsirene promenne

PROGRAM prgMain
  VAR
    Device :     ARRAY [1 .. DEVICE_COUNT] OF TDevice;

    sm_enable : BOOL;

    sm : fb_SMReadData;
  END_VAR

  IF smInit.done THEN
    smInit_enable := false;
    sm_enable := true;
  END_IF;

  //nacteni zakladnich promennych
  sm(enable := sm_enable, ip := IP_Address, chanCode := chanCode, tcp := tcp,
DevInfoPtr := ADR(DeviceInfo), DevPtr := ADR(Device), devCount:= smInit.devCount);

  IF sm.done OR sm.err THEN
    sm_enable := false;
    smFini(tcp := tcp);
  END_IF;

END_PROGRAM
```

6.4 Funkční blok „fb_SMReadStuderMiBData“





Podobně jako fb_SMReadData tento funkční blok čte data podle struktury TDeviceInfo a plní jimi strukturu TdeviceMiB.

Nastavením vstupu enable na true funkční blok začne provádět svou činnost, kterou můžeme kdykoli přerušit shozením vstupu enable do false. Typicky se blok používá s trvale nastaveným vstupem – důležitý je přechod z false na true (náběžná hrana), výstup done je pulzní. Tímto způsobem pracují všechny funkční bloky knihovny.

Pokud potřebujeme výstupním pulzem spustit další funkční blok, např. čtení dat, je nutné použít SR klopný obvod.

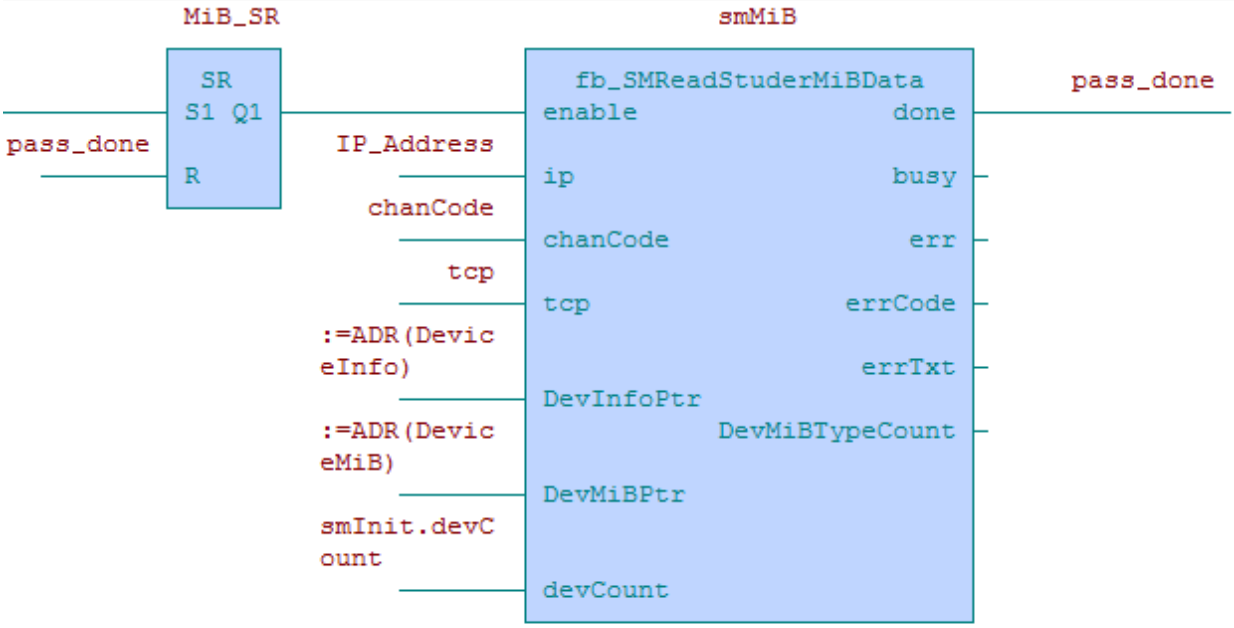
	Název	**Typ**	**Popis**
Vstup	enable	BOOL	Parametr pro spuštění bloku
	ip	STRING	IP adresa jednotky, které se bude nastavovat hodnota výkonu
	chanCode	UINT	Číslo komunikačního kanálu
	tcp	BOOL	Typ protokolu (TRUE=TCP FALSE=UDP)
	DevInfoPtr	PTR_TO TDeviceBlockInfo	Ukazatel na pole struktur TDeviceBlockInfo
	DevMiBPtr	PTR_TO TDeviceMiB	Ukazatel na pole struktur TDeviceMiB
	devCount	USINT	Velikost těchto polí
Výstup	done	BOOL	Vyčítání ukončeno
	busy	BOOL	Probíhá vyčítání
	err	BOOL	Vyčítání skončilo chybou
	errCode	USINT	Chybový kód
	errTxt	STRING	Text chyby, který odpovídá errCode
	DevMiBTypeCount	UINT	Struktura, která je naplněna počty jednotlivých druhů zařízení. Tyto druhy se liší od DevTypeCount v minulém funkčním bloku tím, že rozdělují zařízení Studer – např. 3 střídače Xtender, 2 MPP trackery Variotrack, 8 MPP trackerů Variostring

6.4.1 Příklad – ST: 13-cteni_dat_cyklicky

Podobně jako příklad 12 vyčte informace o blocích, uloží je do pole DeviceInfo a podle něj pak cyklicky vyčítá funkčním blokem b_SMReadStuderMiBData přednastavené informace a plní je do pole DeviceMiB.

Pokud zařízení nemá žádný Studer blok, tak se s tímto zařízením vůbec nebude komunikovat. Pokud žádné z připojených zařízení takový blok nebude mít, funkční blok fb_SMReadStuderMiBData nevyvolá žádnou komunikaci.

```
Schema cyklicky cte udaje ze zarizeni,
ktera jsou pripojena k jednotce SM2-MU.
Rozsirene vycitani zjisti pripadne jednotky Studer-Innotec
a vycte z nich detailnejsi info (do pole DeviceMiB).

Vysledky jsou v polich Device[] a DeviceMiB[],
informace o datovych blocich v jednotlivych zarizenich v poli DeviceInfo[].

Nejprve probehne smInit a na 1 cyklus nastavi init_done.
Ten je zachycen read_SR a drzi vykonavani sm.
Kdyz sm skonci s komunikaci, nastavi MiB_SR a komunikce pokracuje v smMiB.
Po jeho dokonceni je nastaven signal pass_done, ktery jednak vyresetuje
read_SR a MiB_SR a jednak spusti timer_SR a timer. Ti vygeneruji prodlevu
mezi cykly, po ktere se opet nahodi read_SR.
0004
Jeden cteci cyklus - rozsirene cteni.

Oba bloky potrebuji nabeznou hranu a pak enable=1 po celou dobu sve prace.
Klopne obvody jsou zde po podrzeni signalu done, ktery je pouze pulzni.
Pozn.: Pred spustenim dalsiho cyklu je zapotrebi vratit enable na 0.
```



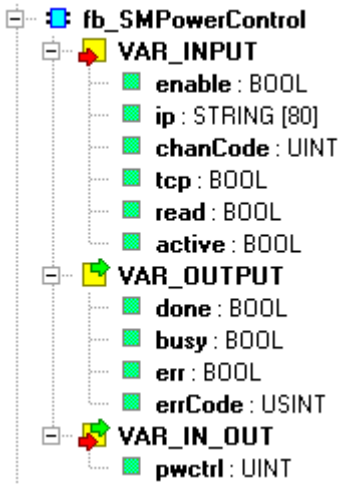
6.4.2 Příklad – ST: 3-cteni_dat_cyklicky

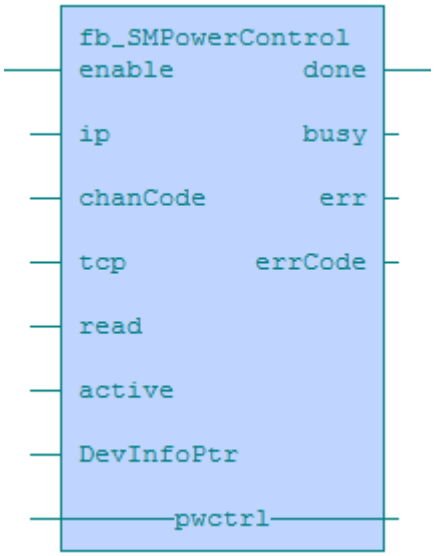
Ve strukturovaném textu to pak vypadá takto.

V ukázce níže jsou pouze relevantní řádky, plnou verzi si můžete stáhnout – viz 1.2.

```
//// Ukazka cyklickeho cteni dat - jako priklad 2, navic se opakuje s
periodou INTERVAL_TIME

VAR_GLOBAL CONSTANT
  INTERVAL_TIME: TIME := T#5s;
END_VAR

PROGRAM prgMain

  VAR
    DeviceMiB :  ARRAY [1 .. DEVICE_COUNT] OF TDeviceMiB;

    smMiB_enable : BOOL;

    Timer : TON;

    smMiB :fb_SMReadStuderMiBData;
  END_VAR

  IF smInit.done THEN
     smInit_enable := false;
     sm_enable := true;
   END_IF;

  //nacteni rozsirenych promennych (provede se pouze pro Studer-Innotec zarizeni)
  smMiB(enable := smMiB_enable, ip := ip, chanCode := chanCode, tcp := tcp,
DevInfoPtr := ADR(DeviceInfo), DevMiBPtr := ADR(DeviceMiB), devCount:= smInit.devCount);

  IF smMiB.done OR smMiB.err THEN
    smMiB_enable := false;
    Timer(IN := 1, PT := INTERVAL_TIME);
  END_IF;

  Timer();      // timer pro cyklicke vycitani

  IF Timer.Q THEN
    Timer(IN := 0);
    sm_enable := true;
  END_IF;

END_PROGRAM
```

6.5 Funkční blok „fb_PowerControl“





Od předchozích funkčních bloků se tento liší tím, že DevInfoPtr ukazuje pouze na 1 prvek pole. V něm je i adresa zařízení, podle které pak funkční blok komunikuje. Parametry read a pwrctrl jsou specifikem tohoto funkčního bloku.

Nastavením vstupu enable na true funkční blok začne provádět svou činnost, kterou můžeme kdykoli přerušit shozením vstupu enable do false. Typicky se blok používá s trvale nastaveným vstupem – důležitý je přechod z false na true (náběžná hrana), výstup done je pulzní. Tímto způsobem pracují všechny funkční bloky knihovny.

Pokud potřebujeme výstupním pulzem spustit další funkční blok, např. čtení dat, je nutné použít SR klopný obvod.

	Název	Typ	Popis
Vstup	enable	BOOL	Parametr pro spuštění bloku
	ip	STRING	IP adresa jednotky, které se bude nastavovat hodnota výkonu
	chanCode	UINT	Číslo komunikačního kanálu
	tcp	BOOL	Typ protokolu (TRUE=TCP FALSE=UDP)
	read	BOOL	TRUE – čtení, FALSE - zápis
	active	BOOL	TRUE=aktivní výkon, FALSE=jalový výkon
	DevInfoPtr	PTR_TO TDeviceBlockInfo	Ukazatel na strukturu TdeviceBlockInfo, typicky je to adresa prvku pole, naplněného funkčním blokem fb_SMInit :=ADR(DeviceInfo[slaveID])

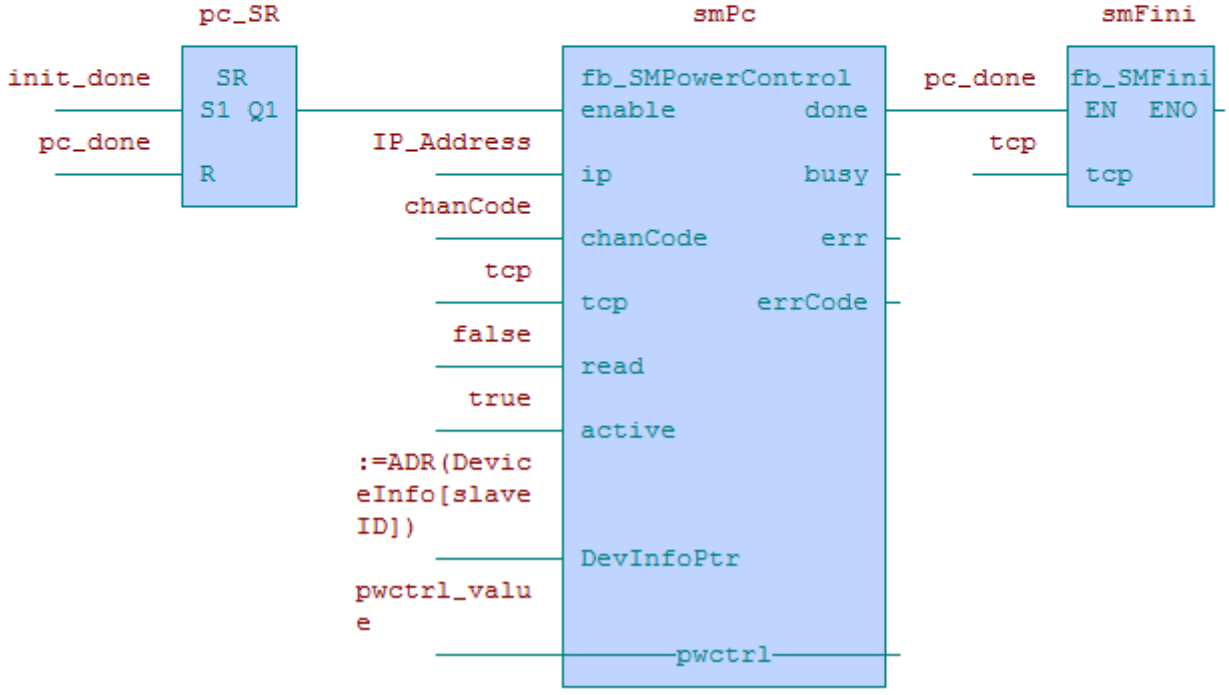
Výstup	done	BOOL	Vyčítání ukončeno
	busy	BOOL	Probíhá vyčítání
	err	BOOL	Vyčítání skončilo chybou
	errCode	USINT	Chybový kód
Vstup / Výstup	pwctrl	UINT	Přečtená hodnota / hodnota k zápisu

6.5.1 Příklad – FB: 14-rizeni_vykonu_fb

Příklad přečte aktuální nebo zapíše požadovanou hodnotu výkonu. Poté obvykle střídači chvilku trvá než skutečný výkon upraví.

Cyklické změny výkonu jsou ukázány v příkladu 17, který má sekvenční variantu i variantu se stavovým automatem.

```
Schema precte / nastavi cinny / jalovy vykon v zarizeni.

O ktere zarizeni se jedna urcuje konstanta slaveID,
ktera udava relativni poradi v poli DeviceInfo[].
Nastaveni false / true vstupu read a active urcuje,
co se precte / nastavi.
0002
Samotne cteni / nastaveni vykonu.
Vstup / vystup je v promenne pwrctrl_value.

Blok smPc potrebuje nabeznou hranu a pak enable=1 po celou dobu sve prace.
Klopny obvod je zde po podrzeni signalu init_done, ktery je pouze pulzni.
```



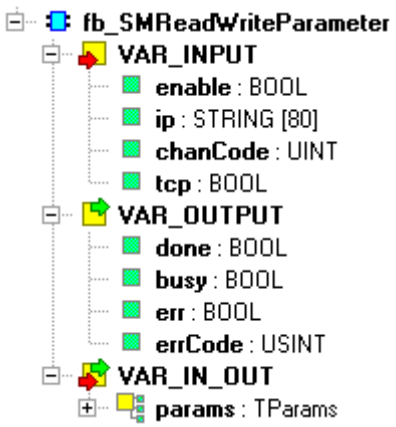
6.5.2 Příklad – ST: 4-rizeni_vykonu

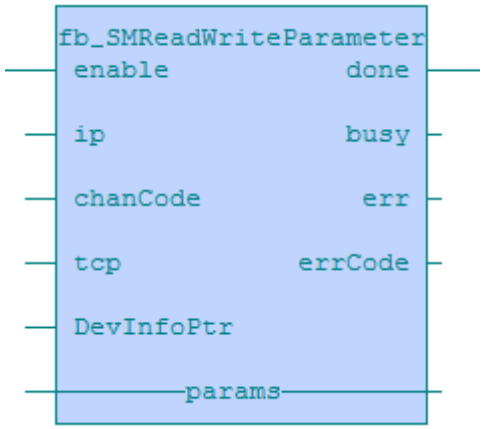
Ve strukturovaném textu to pak vypadá takto.

V ukázce níže jsou pouze relevantní řádky, plnou verzi si můžete stáhnout – viz 1.2.

```
//// Ukazka pouziti funkcniho bloku pro rizeni vykonu

PROGRAM prgMain
  VAR CONSTANT
    slaveID : USINT := 5;              // logicke id zarizeni v SM2-MU,
                                        kteremu nastavujeme vykon
    active : BOOL := true;             // TRUE - aktivni vykon, FALSE - jalovy vykon
    read : BOOL := false;              // TRUE - cteni, FALSE - zapis
  END_VAR

  VAR
    pwctrl_value : UINT := 60;         // nastavovaná hodnota

    smPC_enable : BOOL;

    smPC : fb_SMPowerControl;
  END_VAR

  IF smInit.done THEN
    smInit_enable := false;
    smPC_enable := true;
  END_IF;

  // jednorazove nastaveni vykonu
  smPC(enable := smPC_enable, ip := IP_Address, tcp := tcp, read := read,
chanCode := chanCode, pwctrl := pwctrl_value, active := active,
DevInfoPtr := ADR(DeviceInfo[slaveID]));

  IF smPC.done THEN
    smPC_enable := false;
    smFini(tcp := tcp);
  END_IF;

END_PROGRAM
```

6.6 Funkční blok „fb_SMReadWriteParameter“





Tento funkční blok slouží k nastavování individuálních parametrů v zařízení, která sou připojena k jednotce Solar Monitor (JSM). Lze to pouze tehdy, když to zařízení umožňuje a ovladač v jednotce Solar Monitor tuto funkcionalitu podporuje. Pokud ne, dojde k chybě SM_ERR_RWPAR_BLOCK_MISSING.

Funkční blok používá strukturu Tparams – viz 3.10. Pracuje v několika krocích – nejprve operaci spustí, tím se přenese požadavek (příkaz s parametry) do jednotky Solar Monitor, která v tu chvíli pracuje jako proxy brána, a ta požadavek začne vykonávat, tj. začne komunikovat se zařízením, které má připojeno přes sériovou sběrnici (RS485, RS422, RS232) či přes LAN. Funkční blok se mezitím neustále dotazuje, zda je požadavek již vykonán. Jednotka Solar Monitor to signalizuje vynulováním parametru cmd. Všechny tyto operace zapouzdřuje funkční blok, uživatel se o ně nemusí starat.

Pro čtení je zadán identifikátor, typ parametru a cmd=READ. Následně je prvek cmd aktualizován a akce končí tehdy, až je přečten cmd=0. Přečtená hodnota je v prvku value.

Podobně při zápisu je zadán identifikátor, typ parametru, zapisovaná hodnota a cmd=WRITE. Až je cmd=0, hodnota je zapsána do vzdáleného zařízení.

Nastavením vstupu enable na true funkční blok začne provádět svou činnost, kterou můžeme kdykoli přerušit shozením vstupu enable do false. Typicky se blok používá s trvale nastaveným vstupem – důležitý je přechod z false na true (náběžná hrana), výstup done je pulzní. Tímto způsobem pracují všechny funkční bloky knihovny.

Pokud potřebujeme výstupním pulzem spustit další funkční blok, např. čtení dat, je nutné použít SR klopný obvod.

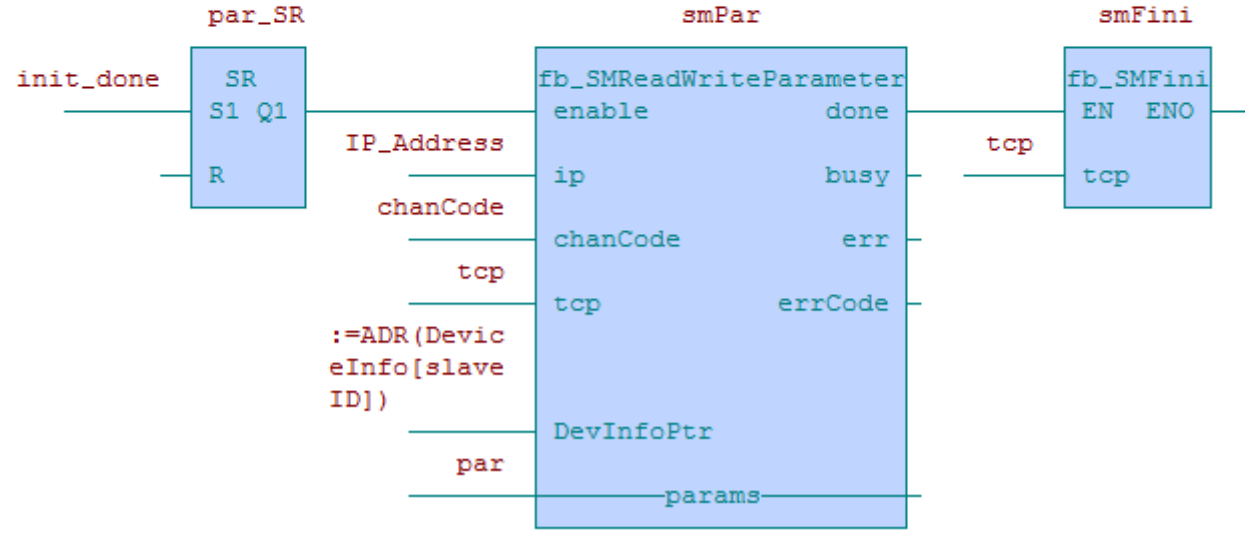
	Název	**Typ**	**Popis**
Vstup	enable	BOOL	Parametr pro spuštění bloku
	ip	STRING	IP adresa jednotky, které se bude nastavovat hodnota výkonu
	chanCode	UINT	Číslo komunikačního kanálu
	tcp	BOOL	Typ protokolu (TRUE=TCP FALSE=UDP)
	DevInfoPtr	PTR_TO	

		TDeviceBlockInfo	
Výstup	done	BOOL	Vyčítání ukončeno
	busy	BOOL	Probíhá vyčítání
	err	BOOL	Vyčítání skončilo chybou
	errCode	USINT	Chybový kód
Vstup / Výstup	params	TParams	Struktura pro práci s parametry

6.6.1 Příklad – FB: 15-jeden_parametr_fb

Příklad ukazuje nastavení jednoho parametru. Nastavení více parametrů je ukázáno v příkladu 16.

```
Schema cte / nastavuje parametr v zarizenich (napr. Studer).

O ktere zarizeni se jedna urcuje konstanta slaveID,
ktera udava relativni poradi v poli DeviceInfo[].
Nastaveni false / true vstupu read urcuje,
co se cte / nastavuje.
0002
Samotne cteni / zapis parametru.
Vystup je v promenne par.value.

Blok smPar potrebuje nabeznou hranu a pak enable=1 po celou dobu sve prace.
Klopny obvod je zde po podrzeni signalu init_done, ktery je pouze pulzni.
```



6.6.2 Příklad – ST: 5-jeden_parametr

Ve strukturovaném textu je snadné změnit číslo parametru.

Co který parametr znamená lze zjistit v technické specifikaci výrobce.

```
//// Ukazka nacteni jednoho parametru (detailnejsi popis komunikace viz dokumentace)

PROGRAM prgMain
  VAR CONSTANT
    slaveID : USINT := 1;              // logicke id zarizeni v SM2-MU (zde musi byt
                                        id Xtenderu, protoze pracujeme s parametrem 1297)
  END_VAR

  VAR
    smPar_enable : BOOL;

    smPar : fb_SMReadWriteParameter;;

    par : TParams;
    res : REAL;
  END_VAR

  IF smInit.done THEN
    smInit_enable := false;
    smPar_enable := true;

    par.id := 1297;
    par.pType := PARAM_FLOAT;
    par.cmd := PARAM_READ;
  END_IF;

  //nacteni promenne podle parametru
  smPar(enable := smPar_enable, ip := IP_Address, chanCode := chanCode, tcp := tcp,
params := par, DevInfoPtr := ADR(DeviceInfo[slaveID]));

  IF smPar.done THEN
    smPar_enable := false;
    res := par.value;
    smFini(tcp := tcp);
  END_IF;

END_PROGRAM
```

7. Kódy chybových hlášení

Číslo chyby	Popis chyby (anglicky)	Popis chyby (česky)
0	No error	Bez chyby
129	Response with other slave address	Odpověď s jinou podřízenou adresou
130	Response with other FNC	Odpověď s jinou FNC
131	Checksum error in reception	Chyba kontrolního součtu během příjmu
132	Unknow FNC in transmit command	Neznámý FNC ve vysílaném příkazu
133	Responce with Unknown FNC	Odpověď s neznámým FNC
135	Response Timeout Error	Uplynul čas čekání na odpověď
137	Exception:ILLEGAL FUNCTION	Vyjímka: Nepovolená funkce
138	Exception:ILLEGAL DATA ADDRESS	Vyjímka: Nepovolená adresa dat
139	Exception:ILLEGAL DATA VALUE	Vyjímka: Nepovolená hodnota dat
140	Exception: SLAVE DEVICE FAILURE	Vyjímka: Neschopné podřízené zařízení
141	Exception: ACKNOWLIDGE	Vyjímka: Potvrzení
142	Exception: SLAVE DEVICE BUSY	Vyjímka: Podřízené zařízení zaměstnáno
144	Exception: MEMORY PARTITY ERROR	Vyjímka: Chyba parity paměti
146	Exception: GATEWAY PATH UNAVAILABLE	Vyjímka: Nepoužitelná cesta bránou
147	GATEWAY TARGET DEVICE FAILED TO RESPOND	Vyjímka: Cílové zařízení nepřístupné touto bránou
148	Invalid parameter chanCode	Neplatný parametr chanCode
149	Can not establish a TPC connention	Nelze navázat TCP spojení
150	Invalid IP address od slave device	Neplatná IP adresa slave zařízení
151	Change of IP address failed	Změna adresy IP se nezdařila
200	SMERR_BASE	Offset pro chyby knihovny SolarMonitor
201	SMERR_NO_SUNSPEC_MARK	Modbus promenne nezačínají „SunS“
202	SMERR_MAX_DEVICE_NUM_EXCEEDED	Prekročení maximálního počtu zařízení (→ zvětšete DEVICE_COUNT)
203	SMERR_MAX_BLOCK_NUM_EXCEEDED	Prekročení maximálního počtu bloků
204	SMERR_RWPAR_BLOCK_MISSING	Zařízení nemá blok pro čtení/zápis individuálních parametrů
205	SMERR_PWCTRL_BLOCK_MISSING	Zařízení nemá blok pro řízení výkonu