PLC TECOMAT JAKO STANICE V SÍTI BACnet

PLC TECOMAT JAKO STANICE V SÍTI BACnet

1. vydání - srpen 2009

OBSAH

1. BACnet

1.1 CO JE BACnet ?

BACnet je komunikační protokol pro automatizační a operátorskou úroveň řízení technického zařízení budov (TZB). Základní myšlenkou protokolu BACnet je formulace univerzálního popisu všech možných objektů, jejich funkcí a parametrů používaných v TZB. K tomu slouží normou předepsané objekty, jejich funkce a parametry. Systém BACnet je celosvětovou normou, výkonným standardem automatizace budov. Používá se bez licenčních poplatků.

Popis komunikačního protokolu přesahuje rámec této dokumentace, více dokument ASHARE 135-2004, www.bacnet.org) .

Předpokládáme, že uživatel je seznámen se základy protokolu BACnet. K testování můžeme používat nástroj VTS3 volně dostupný na http://sourceforge.net/projects/vts.

2. KOMUNIKAČNÍ PROTOKOL BACnet V PLC TECOMAT

2.1. REALIZACE STANICE PLC TECOMAT v BACnet

PLC TECOMAT je možné připojit do sítě BACnet protokolem:

- BACnet IP
- MS/TP (zatím v přípravě)

jako slave stanici.

Požadované verze firmware:
CP-7004 sw 4.7 a vyšší.
FOXTROT sw 4.7 a vyšší.

Požadovaná verze vývojového prostředí MOSAIC verze 2.0.19.0 a vyšší.

Připojení protokolem BACnet IP je realizováno na ethernet rozhraní PLC TECOMAT. Konfigurace rozhraní a služby protokolu BACnet viz dále.

Po připojení PLC TECOMAT do sítě BACnet jsou zveřejněny definované objekty s jejich parametry ostatním stanicím sítě BACnet, které mohou parametry objektů číst a zapisovat.

3. OBJEKTY BACnet V PLC TECOMAT

3.1. OBJEKTY BACnet PODPOROVANÉ V PLC TECOMAT

Objekty BACnet a jejich vlastnosti implementované v PLC TECOMAT.

Tabulka 1: implementované objekty a jejich vlastnosti

	Zařízení	Program	Binární vstup	Binární výstup	Binární hodnota	Analogový vstup	Analogový výstup	Analogová hodnota
Object Identifier	☑	☑	☑	☑	☑	☑	☑	☑
Object Name	☑	☑	☑	☑	☑	☑	☑	☑
Object Type	☑	☑	☑	☑	☑	☑	☑	☑
Description	☑	☑	☑	☑	☑	☑	☑	☑
System Status	☑	☑						
Vendor Name	☑							
Vendor Identifier	☑							
Model Name	☑							
Firmware Revision	☑							
Protocol Version	☑							
Location	☑							
Services Supported	☑							
Object Types Supported	☑							
Object List	☑							
Present Value			☑	☑	☑	☑	☑	☑
Units						☑	☑	☑
Status Flags		☑	☑	☑	☑	☑	☑	☑
Event State			☑	☑	☑	☑	☑	☑
Out of service		☑	☑	☑		☑	☑	
Description of halt		☑						
Polarity			☑	☑				

3.2. ZAŘÍZENÍ

Objekt zařízení (device) slouží k identifikaci zařízení v síti BACnet. Objekt obsahuje vlastnosti popisující parametry zařízení.

Tabulka 2: objekt zařízení (device) a jeho vlastnosti

Device	Popis
Object Identifier	Identifikátor typu objektu: ID objektu viz dále *****
Object Name	Jméno objektu: TECONAT BACnet server
Object Type	Typ objektu: DEVICE = 8
Description	Popis objektu: PLC TECOMAT with BACnet server
System Status	Status objektu: STATUS OPERATIONAL
Vendor Name	Jméno výrobce: TECO a.s. Czech Republic
Vendor Identifier	Identifikační číslo výrobce: 344
Model Name	Jméno modelu PLC TECOMAT: TECOMAT FOXTROT / TECOMAT CP-7004
Firmware Revision	Revize firmware: verze firmware PLC TECOMAT
Protocol Version	Verze protokolu: 1
Location	Lokace: CZ
Services Supported	Seznam podporovaných služeb BACnet: viz. ASHARE 135-2004
Object Types Supported	Seznam podporovaných objektů
Object List	Seznam objektů implementovaných a použitých v PLC TECOMAT

***** vlastnost Object Identifier se nastavuje v konfiguračním dialogu vývojového prostředí MOSAIC

3.3. PROGRAM

Objekt program slouží k:

- k identifikaci aplikačního programu v PLC (jméno programu, verze překladače, typu PLC, použitého překladače, času a data překladu)
- stavu programu RUN / HALT
- zjištění příčiny stavu HALT – je dostupné chybové hlášení diagnostiky PLC

Tabulka 3: objekt program a jeho vlastnosti

Program	**Popis**
Object Identifier	Identifikátor typu objektu: 0 (pouze jedna instance programu PLC)
Object Name	Jméno objektu: jméno programu
Object Type	Typ objektu: PROGRAM = 16
Description	Popis objektu: hlavička programu (jméno programu, verze, typ PLC, použitý překladač, datum a čas překladu)
System Status	Systémový status objektu: RUN / HALT
Status Flags	Status běhu programu: bez chyb, varování, chyba
Out of service	TRUE pokud je program ve stavu HALT, FALSE pokud je program ve stavu RUN
Description of halt	Popis příčiny stavu HALT: např. E-80-04-0000 Invalid program at EEPROM

3.4. BINÁRNÍ VSTUP

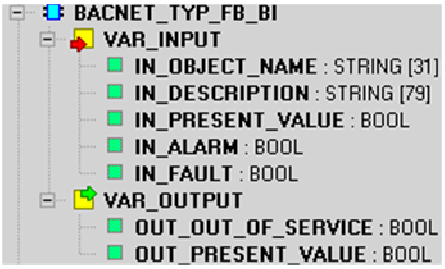
Objekt binární vstup slouží k:

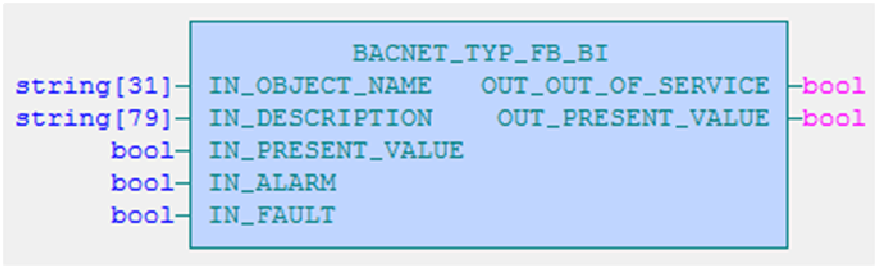
- zjištění stavu binárního vstupu PLC
- zjištění chyby obsluhy binárního vstupu (nastavuje uživatelský program)
- simulaci stavu binárního vstupu (ručnímu nastavení na hodnotu TRUE/FALSE, (1/0), pro testování)

Tabulka 4.1: objekt binární vstup a jeho vlastnosti

Binární vstup	Popis
Object Identifier	Identifikátor typu objektu: 0..maximální počet binárních vstupů PLC viditelných v BACnet (typicky 32) viz dále konfigurace v prostředí MOSAIC
Object Name	Jméno objektu: jméno objektu např. BINARY_INPUT_0
Object Type	Typ objektu: BINARY_INPUT = 3
Description	Popis objektu: popis objektu např. Description of BINARY_INPUT_0
Present Value	Aktuální hodnota binárního vstupu: TRUE / FALSE, (1/0)
Status Flags	Stav vstupu: OUT OF SERVICE=1 pokud je požadavek na OUT OF SERVICE
Event State	Vždy EVENT_STATE_NORMAL
Out of service	1 = OUT OF SERVICE, odpojení fyzické svorky vstupu, za platnou se bere hodnota z Preset Value, kterou lze přepisovat na hodnotu TRUE / FALSE
Polarity	Vždy POLARITY_NORMAL

Deklarace typu **BACNET_TYP_FB_BI** - BINÁRNÍ VSTUP





Tabulka 4.2: Význam položek datového typu BACNET_TYP_FB_BI

Binární vstup	Popis
IN_OBJECT_NAME	Jméno objektu: např. BINARY_INPUT_0, FIRE_007,…
IN_DESCRIPTION	Popis objektu: popis objektu např. Description of BINARY_INPUT_0, Požární hlásič kotelna K12, atd.
IN_PRESENT_VALUE	Aktuální hodnota binárního vstupu: TRUE / FALSE, (1/0), při volání funkčního bloku zde může být přiřazena hodnota konkrétního binárního vstupu, libovolná logická proměnná nebo konstanta TRUE / FALSE.
IN_ALARM	Přiřazení poruchového vstupu (alarmu) z libovolného zdroje, např. systémové proměnné, libovolné logické proměnné atd.
IN_FAULT	Přiřazení poruchového vstupu (poruchy) z libovolného zdroje, např. systémové proměnné, libovolné logické proměnné atd.
OUT_OUT_OF_SERVICE	OUT_OUT_OF_SERVICE = TRUE, odpojení vstupu funkčního bloku IN_PRESENT_VALUE, na výstupu funkčního bloku OUT_PRESENT_VALUE je hodnota z vnitřní proměnné *Preset_value*, kterou lze přepisovat na hodnotu TRUE / FALSE např. z důvodu testování aplikace atd.
OUT_PRESENT_VALUE	Pokud je OUT_OUT_OF_SERVICE = TRUE, pak OUT_PRESENT_VALUE = IN_PRESENT_VALUE. Pokud je OUT_OUT_OF_SERVICE = FALSE, pak OUT_PRESENT_VALUE = vnitřní proměnné *Preset_value*.

3.5. BINÁRNÍ VÝSTUP

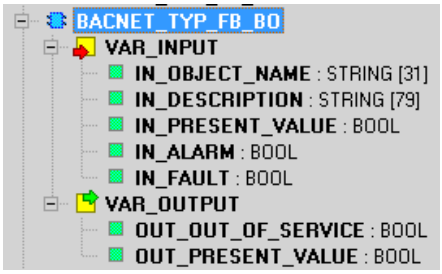
Objekt binární výstup slouží k:

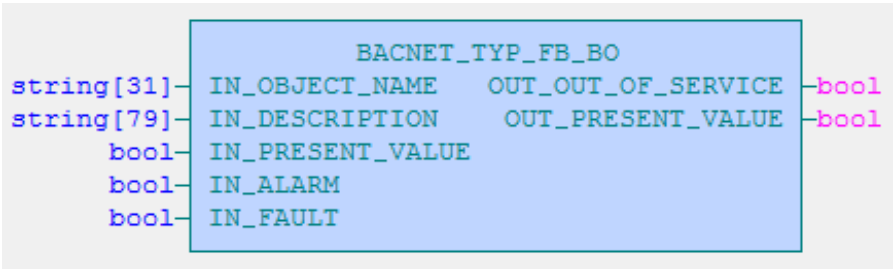
- zjištění stavu binárního výstupu PLC
- zjištění chyby obsluhy binárního výstupu (nastavuje uživatelský program)
- simulaci stavu binárního výstupu (ručnímu nastavení na hodnotu TRUE/FALSE, (1/0), pro testování)

Tabulka 5.1: objekt binární výstup a jeho vlastnosti

Binární výstup	Popis
Object Identifier	Identifikátor typu objektu: 0..maximální počet binárních výstupů PLC viditelných v BACnet (typicky 32) viz dále konfigurace v prostředí MOSAIC
Object Name	Jméno objektu: jméno objektu např. BINARY_OUTPUT_0
Object Type	Typ objektu: BINARY_OUTPUT = 4
Description	Popis objektu: popis objektu např. Description of BINARY_OUTPUT_0
Present Value	Aktuální hodnota binárního výstupu: TRUE / FALSE, (1/0)
Status Flags	Stav vstupu: OUT OF SERVICE=1 pokud je požadavek na OUT OF SERVICE
Event State	Vždy EVENT_STATE_NORMAL
Out of service	1 = OUT OF SERVICE, odpojení výstupu funkčního bloku do Preset Value, za platnou se bere hodnota z Preset Value, kterou lze přepisovat na hodnotu TRUE / FALSE
Polarity	Vždy POLARITY_NORMAL

Deklarace typu **BACNET_TYP_FB_BO** - BINÁRNÍ VÝSTUP





Tabulka 5.2: Význam položek datového typu BACNET_TYP_FB_BO

Binární výstup	Popis
IN_OBJECT_NAME	Jméno objektu: např. BINARY_OUTPUT_0, VENTIL_V387, …
IN_DESCRIPTION	Popis objektu: popis objektu např. Description of BINARY_OUTPUT_0, Ventil V378 přidej, Ventil V387 uber, atd.
IN_PRESENT_VALUE	Aktuální hodnota přiřazovaná funkčním blokem do binárního výstupu: TRUE / FALSE, (1/0), při volání funkčního bloku zde může být libovolná logická proměnná nebo konstanta TRUE / FALSE.
IN_ALARM	Přiřazení poruchového vstupu (alarmu) z libovolného zdroje, např. systémové proměnné, libovolné logické proměnné atd.
IN_FAULT	Přiřazení poruchového vstupu (poruchy) z libovolného zdroje, např. systémové proměnné, libovolné logické proměnné atd.
OUT_OUT_OF_SERVICE	OUT_OUT_OF_SERVICE = TRUE, odpojení vstupu funkčního bloku IN_PRESENT_VALUE, na výstupu funkčního bloku OUT_PRESENT_VALUE je hodnota z vnitřní proměnné *Preset_value*, kterou lze přepisovat na hodnotu TRUE / FALSE např. z důvodu testování aplikace atd.
OUT_PRESENT_VALUE	Pokud je OUT_OUT_OF_SERVICE = TRUE, pak OUT_PRESENT_VALUE = IN_PRESENT_VALUE. Pokud je OUT_OUT_OF_SERVICE = FALSE, pak OUT_PRESENT_VALUE = vnitřní proměnné *Preset_value.*

3.6. BINÁRNÍ HODNOTA

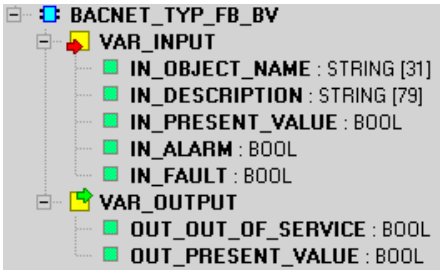
Objekt binární hodnota slouží k:

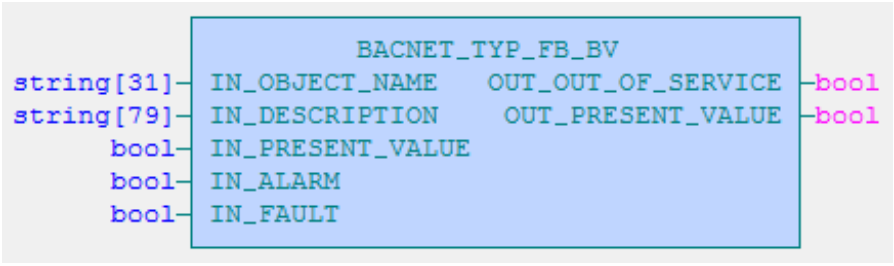
- zjištění nebo změně stavu binárního proměnné v PLC
- zjištění chyby obsluhy binární proměnné (nastavuje uživatelský program)
- simulaci stavu binární proměnné (ručnímu nastavení na hodnotu 1/0 pro testování)

Tabulka 6.1: objekt binární hodnota a její vlastnosti

Binární hodnota	Popis
Object Identifier	Identifikátor typu objektu: 0..maximální počet binárních proměnných PLC viditelných v BACnet (typicky 32) viz dále konfigurace v prostředí MOSAIC
Object Name	Jméno objektu: jméno objektu např. BINARY_VALUE_0
Object Type	Typ objektu: BINARY_VALUE = 5
Description	Popis objektu: popis objektu např. Description of BINARY_OUTPUT_0
Present Value	Aktuální hodnota binární proměnné: TRUE / FALSE, (1/0)
Status Flags	Stav vstupu: OUT OF SERVICE=1 pokud je požadavek na OUT OF SERVICE
Event State	Vždy EVENT_STATE_NORMAL
Out of service	1 = OUT OF SERVICE, odpojení výstupu funkčního bloku do Preset Value, za platnou se bere hodnota z Preset Value, kterou lze přepisovat na hodnotu TRUE / FALSE

Deklarace typu **BACNET_TYP_FB_BV** - BINÁRNÍ HODNOTA





Tabulka 6.2: Význam položek datového typu BACNET_TYP_FB_BV

Binární hodnota	Popis
IN_OBJECT_NAME	Jméno objektu: např. BINARY_VALUE_0, VENTIL_V387_MANUAL, …
IN_DESCRIPTION	Popis objektu: popis objektu např. Description of BINARY_VALUE_0, Ventil V378 ruční řízení, atd.
IN_PRESENT_VALUE	Aktuální hodnota přiřazovaná funkčním blokem do binárního výstupu: TRUE / FALSE, (1/0), při volání funkčního bloku zde může být libovolná logická proměnná nebo konstanta TRUE / FALSE.
IN_ALARM	Přiřazení poruchového vstupu (alarmu) z libovolného zdroje, např. systémové proměnné, libovolné logické proměnné atd.
IN_FAULT	Přiřazení poruchového vstupu (poruchy) z libovolného zdroje, např. systémové proměnné, libovolné logické proměnné atd.
OUT_OUT_OF_SERVICE	OUT_OUT_OF_SERVICE = TRUE, odpojení vstupu funkčního bloku IN_PRESENT_VALUE, na výstupu funkčního bloku OUT_PRESENT_VALUE je hodnota z vnitřní proměnné *Preset_value*, kterou lze přepisovat na hodnotu TRUE / FALSE např. z důvodu testování aplikace atd.
OUT_PRESENT_VALUE	Pokud je OUT_OUT_OF_SERVICE = TRUE, pak OUT_PRESENT_VALUE = IN_PRESENT_VALUE. Pokud je OUT_OUT_OF_SERVICE = FALSE, pak OUT_PRESENT_VALUE = vnitřní proměnné *Preset_value.*

3.7. ANALOGOVÝ VSTUP

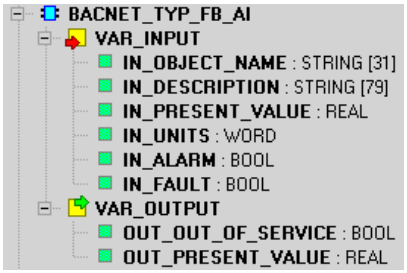
Objekt analogový vstup slouží k:

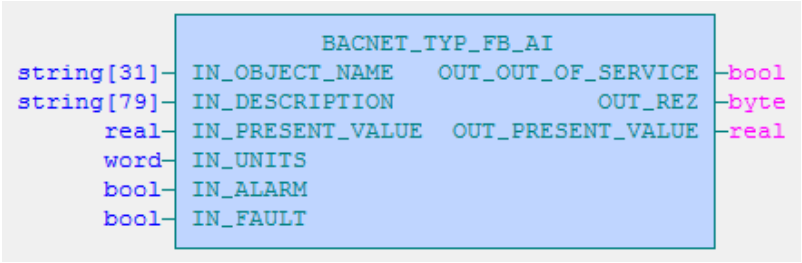
- zjištění stavu analogového vstupu PLC
- zjištění chyby obsluhy analogového vstupu (nastavuje uživatelský program)
- simulaci stavu analogového vstupu (ručnímu nastavení na hodnotu v rozsahu REAL pro testování)

Tabulka 7.1: analogový vstup a jeho vlastnosti

Analogový vstup	Popis
Object Identifier	Identifikátor typu objektu: 0..maximální počet analogových vstupů PLC viditelných v BACnet (typicky 32) viz dále konfigurace v prostředí MOSAIC
Object Name	Jméno objektu: jméno objektu např. ANALOG_INPUT_0
Object Type	Typ objektu: ANALOG_INPUT = 0
Description	Popis objektu: popis objektu např. Description of ANALOG_INPUT_0
Present Value	Aktuální hodnota analogového vstupu.
Units	Použité inženýrské jednotky: např.[mA]
Status Flags	Stav vstupu: OUT OF SERVICE=1 pokud je požadavek na OUT OF SERVICE
Event State	Vždy EVENT_STATE_NORMAL
Out of service	1 = OUT OF SERVICE, odpojení fyzické svorky vstupu, za platnou se bere hodnota z Preset Value, kterou lze přepisovat na hodnotu v rozsahu ???

Deklarace typu **BACNET_TYP_FB_AI** - ANALOGOVÝ VSTUP





Tabulka 7.2: Význam položek datového typu BACNET_TYP_FB_AI

Analogový vstup	Popis
IN_OBJECT_NAME	Jméno objektu: např. ANALOG_INPUT_0, VENTIL_V387_POSITION, …
IN_DESCRIPTION	Popis objektu: popis objektu např. Description of ANALOG_INPUT_0, Ventil V378 poloha skutečná, atd.
IN_PRESENT_VALUE	Aktuální hodnota přiřazovaná funkčním blokem z analogového vstupu: rozsah REAL, při volání funkčního bloku zde může být libovolná proměnná nebo konstanta typu REAL.
IN_ALARM	Přiřazení poruchového vstupu (alarmu) z libovolného zdroje, např. systémové proměnné, libovolné logické proměnné atd.
IN_FAULT	Přiřazení poruchového vstupu (poruchy) z libovolného zdroje, např. systémové proměnné, libovolné logické proměnné atd.
OUT_OUT_OF_SERVICE	OUT_OUT_OF_SERVICE = TRUE, odpojení vstupu funkčního bloku IN_PRESENT_VALUE, na výstupu funkčního bloku OUT_PRESENT_VALUE je hodnota z vnitřní proměnné *Preset_value*, kterou lze přepisovat na hodnotu typu REAL např. z důvodu testování aplikace atd.
OUT_PRESENT_VALUE	Pokud je OUT_OUT_OF_SERVICE = TRUE, pak OUT_PRESENT_VALUE = IN_PRESENT_VALUE. Pokud je OUT_OUT_OF_SERVICE = FALSE, pak OUT_PRESENT_VALUE = vnitřní proměnné *Preset_value.*

3.8. ANALOGOVÝ VÝSTUP

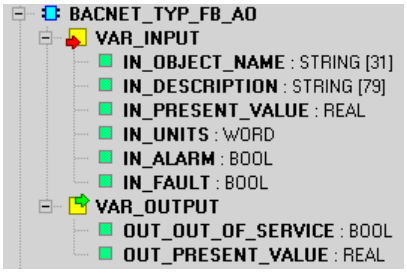
Objekt analogový výstup slouží k:

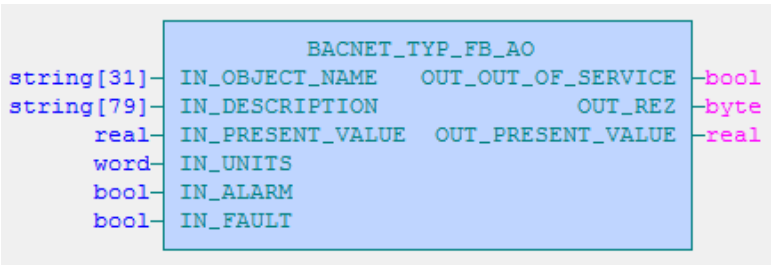
- zjištění stavu analogového výstupu PLC
- zjištění chyby obsluhy analogového výstupu (nastavuje uživatelský program)
- simulaci stavu analogového výstupu (ručnímu nastavení na hodnotu v rozsahu REAL pro testování)

Tabulka 8.1: objekt analogový výstup a jeho vlastnosti

Analogový výstup	Popis
Object Identifier	Identifikátor typu objektu: 0..maximální počet analogových výstupů PLC viditelných v BACnet (typicky 32) viz dále konfigurace v prostředí MOSAIC
Object Name	Jméno objektu: jméno objektu např. ANALOG_OUTPUT_0
Object Type	Typ objektu: ANALOG_OUTPUT = 1
Description	Popis objektu: popis objektu např. Description of ANALOG_OUTPUT_0
Present Value	Aktuální hodnota analogového výstupu
Units	Použité inženýrské jednotky: např.[℃]
Status Flags	Stav vstupu: OUT OF SERVICE=1 pokud je požadavek na OUT OF SERVICE
Event State	Vždy EVENT_STATE_NORMAL
Out of service	1 = OUT OF SERVICE, odpojení fyzické svorky výstupu, za platnou se bere hodnota z Preset Value, kterou lze přepisovat na hodnotu v rozsahu ???

Deklarace typu **BACNET_TYP_FB_AO** - ANALOGOVÝ VÝSTUP





Tabulka 8.2: Význam položek datového typu BACNET_TYP_FB_AO

Analogový výstup	Popis
IN_OBJECT_NAME	Jméno objektu: např. ANALOG_OUTPUT_0, VENTIL_V387_POSITION, …
IN_DESCRIPTION	Popis objektu: popis objektu např. Description of ANALOG_OUTPUT_0, Ventil V378 poloha žádaná, atd.
IN_PRESENT_VALUE	Aktuální hodnota přiřazovaná funkčním blokem do analogového výstupu: rozsah REAL, při volání funkčního bloku zde může být libovolná proměnná nebo konstanta typu REAL.
IN_ALARM	Přiřazení poruchového vstupu (alarmu) z libovolného zdroje, např. systémové proměnné, libovolné logické proměnné atd.
IN_FAULT	Přiřazení poruchového vstupu (poruchy) z libovolného zdroje, např. systémové proměnné, libovolné logické proměnné atd.
OUT_OUT_OF_SERVICE	OUT_OUT_OF_SERVICE = TRUE, odpojení vstupu funkčního bloku IN_PRESENT_VALUE, na výstupu funkčního bloku OUT_PRESENT_VALUE je hodnota z vnitřní proměnné *Preset_value*, kterou lze přepisovat na hodnotu typu REAL např. z důvodu testování aplikace atd.
OUT_PRESENT_VALUE	Pokud je OUT_OUT_OF_SERVICE = TRUE, pak OUT_PRESENT_VALUE = IN_PRESENT_VALUE. Pokud je OUT_OUT_OF_SERVICE = FALSE, pak OUT_PRESENT_VALUE = vnitřní proměnné *Preset_value*.

3.9. ANALOGOVÁ HODNOTA

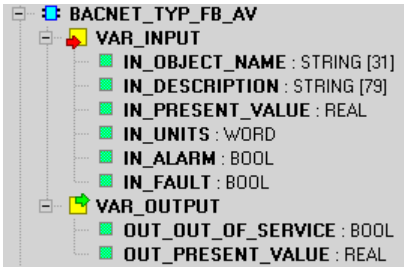
Objekt analogová hodnota slouží k:

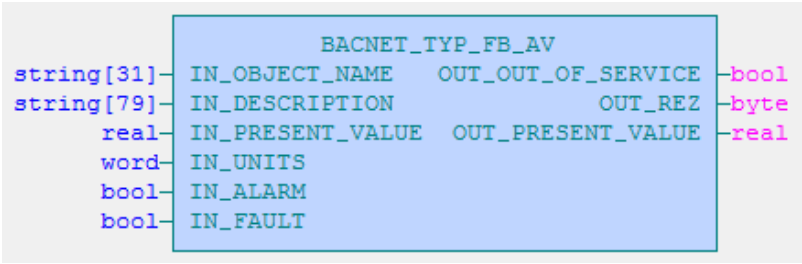
- zjištění nebo změně stavu analogové proměnné (typu single) PLC
- zjištění chyby obsluhy analogové proměnné (nastavuje uživatelský program)
- simulaci stavu analogové proměnné (ručnímu nastavení na hodnotu v rozsahu typu REAL pro testování)

Tabulka 8.1: objekt analogový objekt a jeho vlastnosti

Analogová hodnota	Popis
Object Identifier	Identifikátor typu objektu: 0..maximální počet analogových hodnot PLC viditelných v BACnet (typicky 32) viz dále konfigurace v prostředí MOSAIC
Object Name	Jméno objektu: jméno objektu např. ANALOG_VALUE_0
Object Type	Typ objektu: ANALOG_VALUE = 2
Description	Popis objektu: popis objektu např. Description of ANALOG_VALUE_0
Present Value	Aktuální hodnota analogové hodnoty
Units	Použité inženýrské jednotky: např.[%]
Status Flags	Stav vstupu: OUT OF SERVICE=1 pokud je požadavek na OUT OF SERVICE
Event State	Vždy EVENT_STATE_NORMAL
Out of service	1 = OUT OF SERVICE, odpojení výstupu funkčního bloku do Preset Value, za platnou se bere hodnota z Preset Value, kterou lze přepisovat na hodnotu v rozsahu ???

Deklarace typu **BACNET_TYP_FB_AV** - ANALOGOVÁ HODNOTA





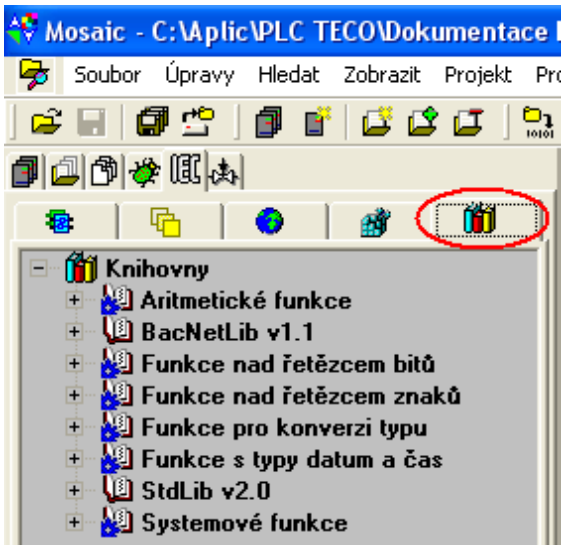
Tabulka 9.2: Význam položek datového typu BACNET_TYP_FB_AV

Analogová hodnota	Popis
IN_OBJECT_NAME	Jméno objektu: např. ANALOG_OBJECT_0, VENTIL_V387_MAX_POSITION, …
IN_DESCRIPTION	Popis objektu: popis objektu např. Description of ANALOG_VALUE_0, Ventil V378 poloha maximální, atd.
IN_PRESENT_VALUE	Aktuální hodnota přiřazovaná funkčním blokem do analogového objektu: rozsah REAL, při volání funkčního bloku zde může být libovolná proměnná nebo konstanta typu REAL.
IN_ALARM	Přiřazení poruchového vstupu (alarmu) z libovolného zdroje, např. systémové proměnné, libovolné logické proměnné atd.
IN_FAULT	Přiřazení poruchového vstupu (poruchy) z libovolného zdroje, např. systémové proměnné, libovolné logické proměnné atd.
OUT_OUT_OF_SERVICE	OUT_OUT_OF_SERVICE = TRUE, odpojení vstupu funkčního bloku IN_PRESENT_VALUE, na výstupu funkčního bloku OUT_PRESENT_VALUE je hodnota z vnitřní proměnné *Preset_value*, kterou lze přepisovat na hodnotu typu REAL např. z důvodu testování aplikace atd.
OUT_PRESENT_VALUE	Pokud je OUT_OUT_OF_SERVICE = TRUE, pak OUT_PRESENT_VALUE = IN_PRESENT_VALUE. Pokud je OUT_OUT_OF_SERVICE = FALSE, pak OUT_PRESENT_VALUE = vnitřní proměnné *Preset_value.*

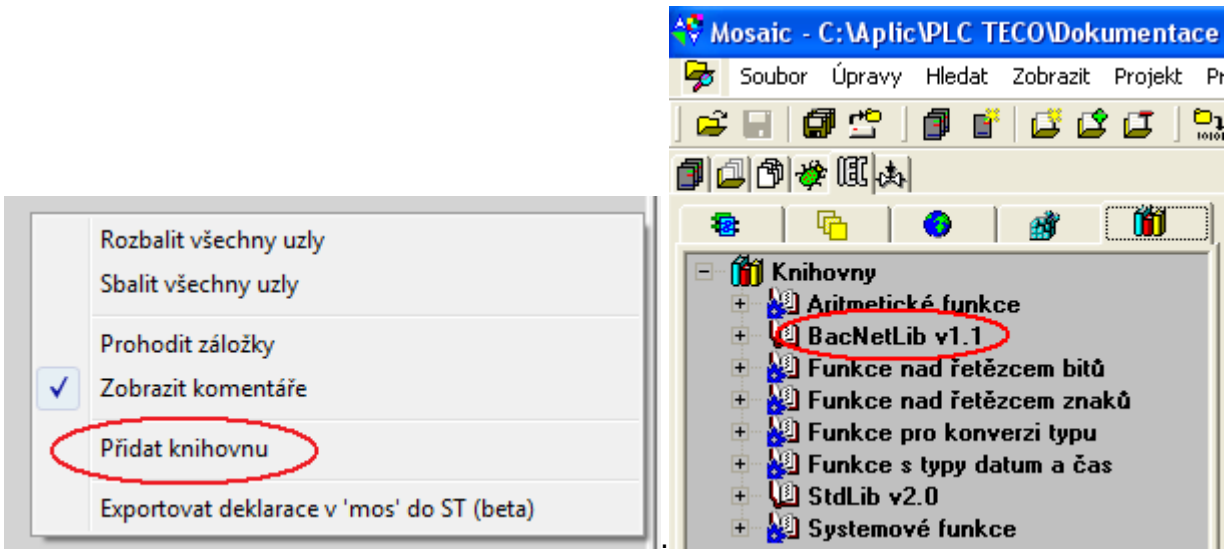
4. KONFIGURACE PROSTŘEDÍ MOSAIC A SLUŽBY BACnet (od verze 2.0.19.0)

4.1 KONFIGURACE PROSTŘEDÍ MOSAIC

Služby BACnet je možno používat po instalaci knihovny BacNetLib v1.1 a vyšší. Instalaci provedeme pomocí správce knihoven.



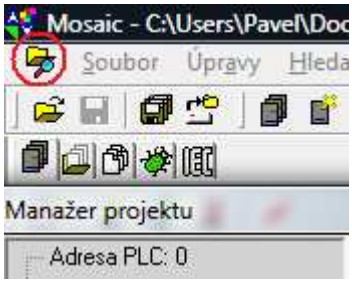
Po stisknutí pravého tlačítka myši se objeví dialog, zvolíme položku Přidat knihovnu a provedeme vlastní instalaci výběrem knihovny BacNetLib v příslušném složce. Pokud instalace proběhla v pořádku bude seznam knihoven obsahovat knihovnu BacNetLib odpovídající verze.

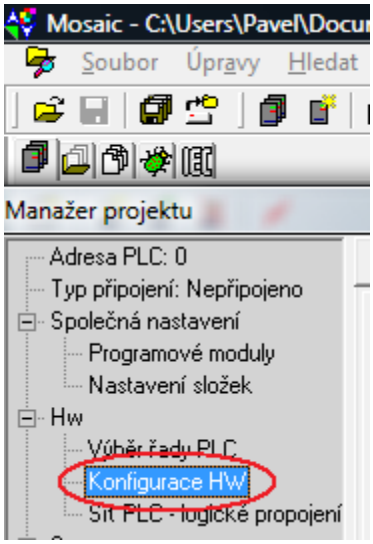


4.2 KONFIGURACE SLUŽBY BACnet V PLC TECOMAT

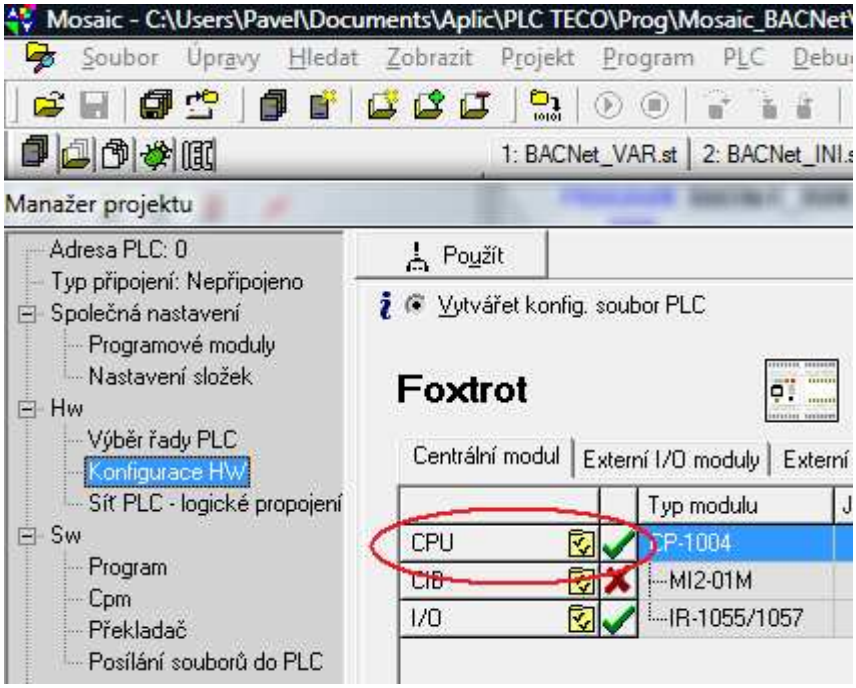
Službu BACnet je nutno před použitím nakonfigurovat. Konfigurace se provádí v prostředí MOSAIC (od verze 2.0.19.0). Služba BACnet „běží “ nad komunikačním kanálem CPU PLC.

4.2.1 Zvolte menu „Manažer projektů“ a následně položku Konfigurace HW

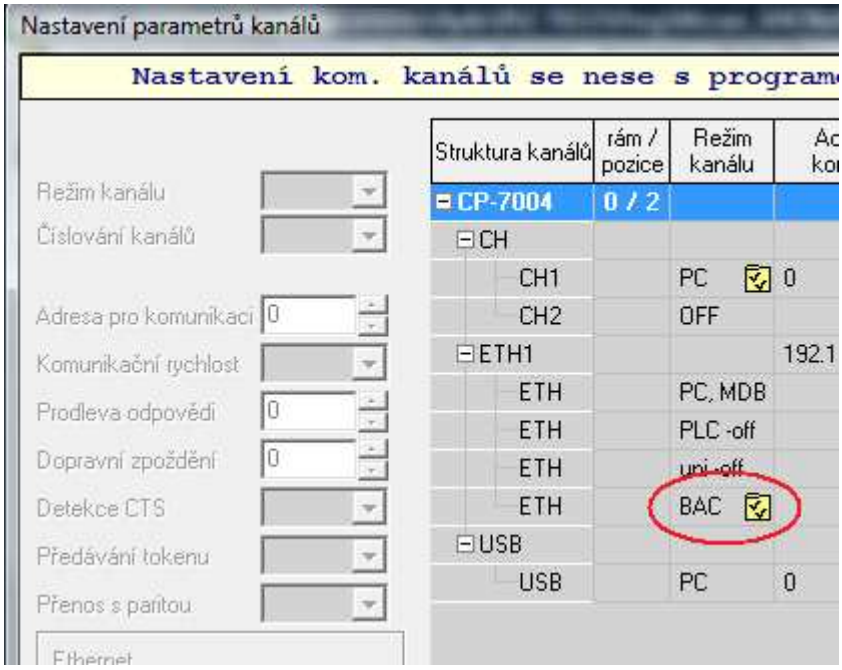




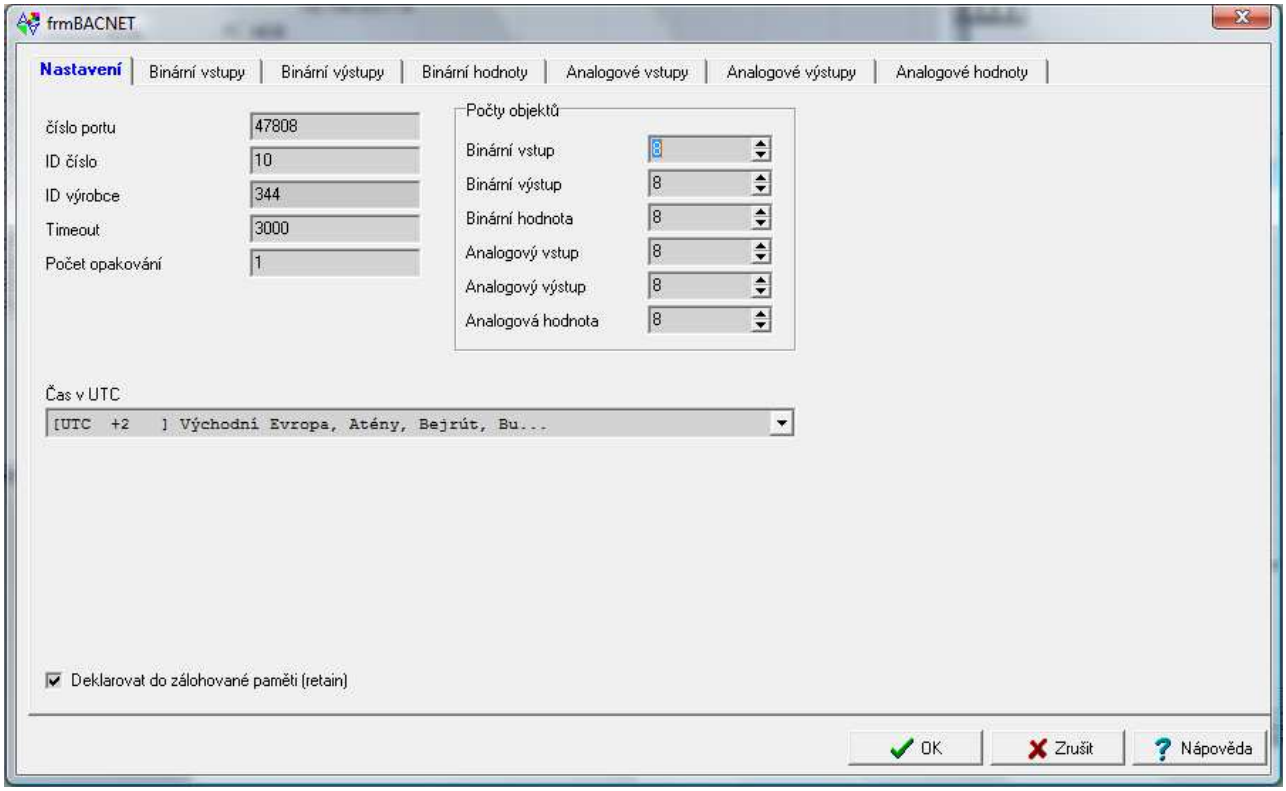
4.2.2 Zvolte položku CPU v záložce Centrální modul



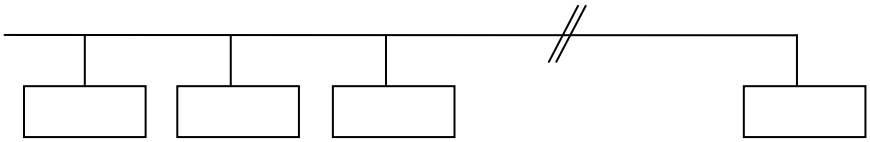
4.2.3 Zvolte položku režim kanálu BAC



4.2.4 Nyní se objeví dialogové okno pro konfiguraci služby BACnet

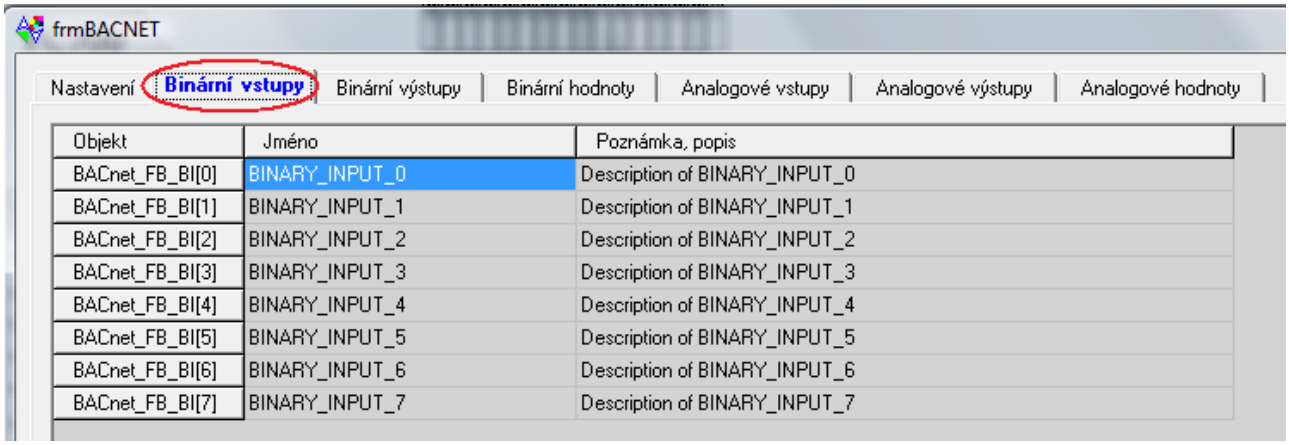


4.2.5 Provedeme nastavení parametrů služby BACnet.

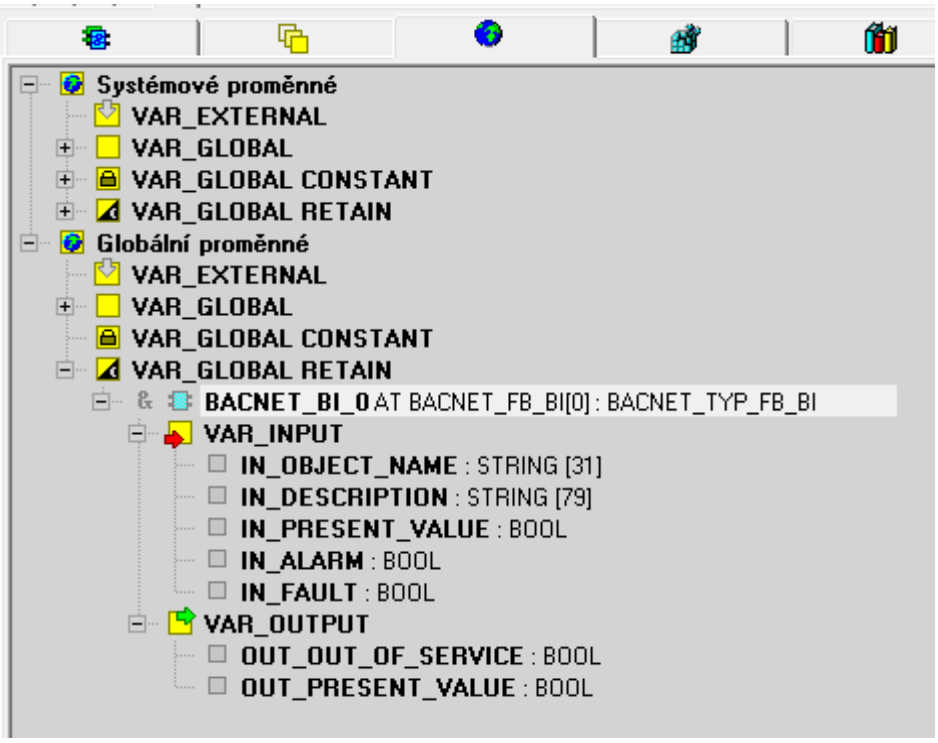
Parametr	Popis
Číslo portu	Typicky 47808 = 0xBAC0, základní číslo portu pro službu BACnet, v rámci vlastní lokální sítě lze použít i jiné číslo, ale musí být stejné u všech jednotek v síti !
ID číslo	Identifikační číslo jednotky v síti = adresa v rámci sítě BACnet. 
ID výrobce	344 – Teco a.s.
Timeout	Doba čekání na odezvu.
Počet opakování	Počet opakování komunikačních cyklů při chybě komunikace.
Čas v UTC	Časové pásmo vůči UTC.
Deklarovat do zálohované paměti (retain)	Deklarace do zálohovaná paměti zaručuje pamatování nastavených parametrů objektů BACnet i po vypnutí/zapnutí PLC. Může nastat situace kdy parametry objektů např. jejich popis je nastaven s operátorské konzole a chceme aby tato změna byla „zapamatována“ i po vypnutí/zapnutá PLC.
Počty objektů	Určuje počet objektů BACnet dostupných v uživatelském programu. Počet každého typu objektů je limitován dle verze firmware CPU PLC. Typicky je maximální počet pro každý typ objektů 32.

4.2.6 Provedeme nastavení parametrů jednotlivých typů objektů

U objektů Binární vstupy, Binární výstupy, Binární hodnoty, Analogové vstupy, Analogové výstupy, Analogové hodnoty můžeme nastavit jméno, popis a případně fyzikální jednotky pokud hodnota je fyzikální veličina. Každý typ objektů má svoji záložku.

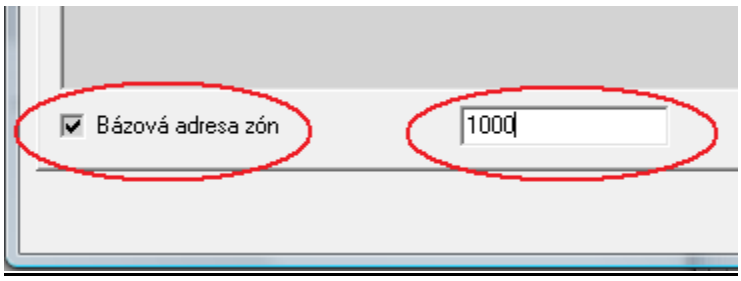


Položka	Popis
Jméno	Symbolické jméno objektu BACnet, který je reprezentován funkčním blokem v uživatelském programu, s tímto jménem se nadále pracuje v uživatelském programu jako s globální proměnnou. Proměnná je

	dostupná přes správce proměnných. 
Poznámka, popis	Podrobný popis objektu, dostupný v uživatelském programu i ostatním stanicím v síti BACnet
Jednotky	Fyzikální jednotky, výběr z dostupného seznamu

4.2.7 Nastavení bázové adresy zón

Při definici parametrů objektů Binární vstupy, Binární výstupy, Binární hodnoty, Analogové vstupy, Analogové výstupy, Analogové hodnoty můžeme nastavit jejich umístění v zápisníku. Navolíme volbu Bázová adresa zón a zadáme offset začátku zónu (tj. adresu prvního objektu v zápisníku).



4.2.8 Kontrola konfigurace objektů BACnet

Po ukončení konfigurace službu a objektů BACnet, provedeme překlad programu. Po bezchybném překladu můžeme v okně správce proměnných vidět a zkontrolovat námi deklarované objekty BACnet.

5. POUŽITÍ OBJEKTŮ BACnet V UŽIVATELSKÉM PROGRAMU

5.1 OBJEKT BACnet JAKO FUNKČNÍ BLOK

Po konfiguraci služby a objektů BACnet a úspěšné překladu programu můžeme tyto objekty používat v uživatelském programu PLC. Objekty jsou reprezentovány funkčními bloky. Deklaraci funkčních bloků je dostupná v knihovně.

Příklad deklarace funkčního bloku FUNCTION_BLOCK BACNET_TYP_FB_BI který reprezentuje objekt binární vstup v BACnet.

```
FUNCTION_BLOCK BACNET_TYP_FB_BI
  VAR_INPUT
    IN_OBJECT_NAME : string [31];
    IN_DESCRIPTION : string [79];
    IN_PRESENT_VALUE {ALIGNED} : bool;
    IN_ALARM {ALIGNED} : bool;
    IN_FAULT {ALIGNED} : bool;
  END_VAR
  VAR_OUTPUT
    OUT_OUT_OF_SERVICE {ALIGNED} : bool;
    OUT_PRESENT_VALUE {ALIGNED} : bool;
  END_VAR
  VAR
    BAC_OUT_OF_SERVICE {ALIGNED} : bool;
    BAC_NORMAL {ALIGNED} : bool;
    BAC_ALARM {ALIGNED} : bool;
    BAC_FAULT {ALIGNED} : bool;
    BAC_PRESENT_VALUE {ALIGNED} : bool;
    BAC_EVENT_STATE : byte;
  END_VAR
END_FUNCTION_BLOCK
```

5.2 AUTOMATICKÉ GENEROVÁNÍ OBJEKTŮ BACnet

Při úspěšném překladu prostředí MOSAIC automaticky generuje deklarace v uživatelském programu, viz následující ukázka.

```
VAR_GLOBAL CONSTANT
  BACNET_MAX_BINARY_INPUTS    : UINT := 0;
  BACNET_MAX_BINARY_OUTPUTS   : UINT := 0;
  BACNET_MAX_BINARY_VALUES    : UINT := 3;
  BACNET_MAX_ANALOG_INPUTS    : UINT := 0;
  BACNET_MAX_ANALOG_OUTPUTS   : UINT := 0;
  BACNET_MAX_ANALOG_VALUES    : UINT := 2;
END_VAR

VAR_GLOBAL
  BACNET_FB_BV : ARRAY[0..BACNET_MAX_BINARY_VALUES-1] OF T_BACNET_TYP_FB_BV := [
        ( IN_OBJECT_NAME := 'BACO_BV_CTUD1_Reset', IN_DESCRIPTION := 'Reset obousměrného čítače' ),
        ( IN_OBJECT_NAME := 'BACO_BV_CTUD1_Up', IN_DESCRIPTION := 'Výstup z čítače, předvolby
dosaženo, výstup = TRUE jinak FALSE' ),
        ( IN_OBJECT_NAME := 'BACO_BV_CTUD1_Down', IN_DESCRIPTION := 'Výstup z čítače, skutečná
hodnota = 0 => výstup TRUE, jinak FALSE' ) ];
```

```
  BACNET_FB_AV : ARRAY[0..BACNET_MAX_ANALOG_VALUES-1] OF T_BACNET_TYP_FB_AV := [
        ( IN_OBJECT_NAME := 'BACO_AV_CTUD1_PresentValue', IN_DESCRIPTION := 'Předvolba čítače',
IN_UNITS := BACNET_UNITS_NO_BACNET_UNITS ),
        ( IN_OBJECT_NAME := 'BACO_AV_CTUD1_CurrentValue', IN_DESCRIPTION := 'Skutečná hodnota
čítače', IN_UNITS := BACNET_UNITS_NO_BACNET_UNITS ) ];

  BACO_BV_CTUD1_Reset         AT BACNET_FB_BV[0] : BACNET_TYP_FB_BV;
  BACO_AV_CTUD1_PresentValue  AT BACNET_FB_AV[0] : BACNET_TYP_FB_AV;
  BACO_BV_CTUD1_Up            AT BACNET_FB_BV[1] : BACNET_TYP_FB_BV;
  BACO_AV_CTUD1_CurrentValue  AT BACNET_FB_AV[1] : BACNET_TYP_FB_AV;
  BACO_BV_CTUD1_Down          AT BACNET_FB_BV[2] : BACNET_TYP_FB_BV;
END_VAR
```

5.3 OBSLUHA OBJEKTŮ BACnet V UŽIVATELSKÉM PROGRAMU

Obsluhou objektů BACnet uživatelským programem se rozumí volání funkčních bloků, které tyto objekty reprezentují. Toto volání není generované vývojovým prostředím MOSAIC, je plně v režii programátora aplikace.

Příklad volání funkčního bloku BINARY_INPUT_0, který reprezentuje objekt binary inputs BACnet.

```
VAR_GLOBAL
  BUT_BI0: BOOL;            // Tlačítko
END_VAR

BINARY_INPUT_0( IN_PRESENT_VALUE := BUT_BI0);
atd.
BINARY_OUTPUT_0 ( IN_PRESENT_VALUE := BUT_BO0 );
BINARY_VALUE_0  ( IN_PRESENT_VALUE := BUT_BV0 );
ANALOG_INPUT_0  ( IN_PRESENT_VALUE := BUT_AI0 );
ANALOG_OUTPUT_0 ( IN_PRESENT_VALUE := BUT_AO0 );
ANALOG_VALUE_0  ( IN_PRESENT_VALUE := BUT_AV0 );
```

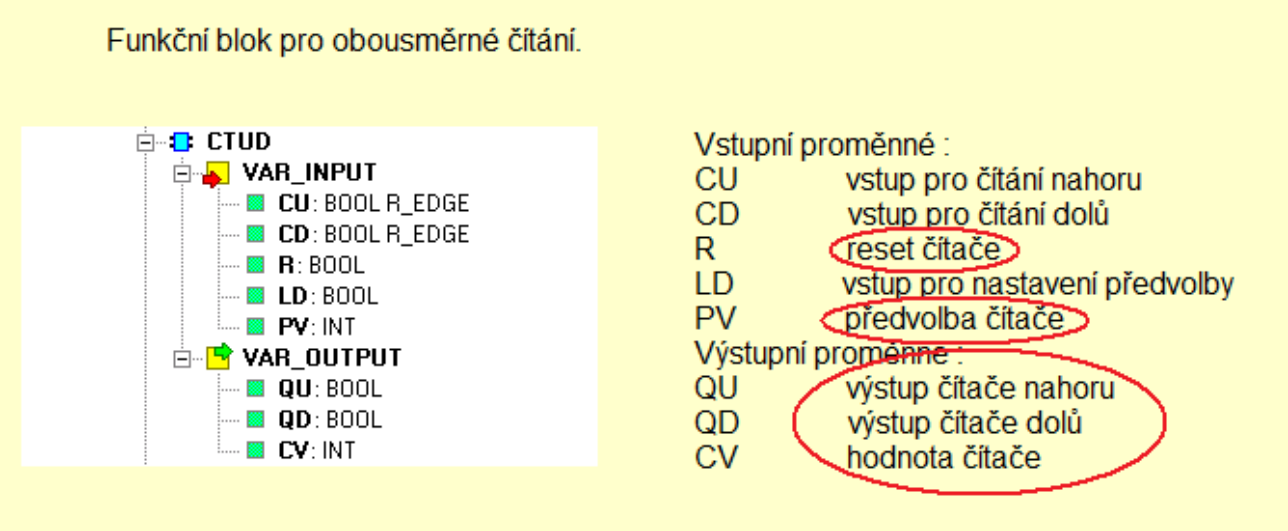
Deklarace funkčního bloku binary inputs a ukázka použití vstupního parametru **IN_PRESENT_VALUE.**

```
BINARY_INPUT_0( IN_PRESENT_VALUE := BUT_BI0);

FUNCTION_BLOCK BACNET_TYP_FB_BI
  VAR_INPUT
    IN_OBJECT_NAME : string [31];
    IN_DESCRIPTION : string [79];
    IN_PRESENT_VALUE {ALIGNED} : bool;
    IN_ALARM {ALIGNED} : bool;
    IN_FAULT {ALIGNED} : bool;
  END_VAR
  VAR_OUTPUT
    OUT_OUT_OF_SERVICE {ALIGNED} : bool;
    OUT_PRESENT_VALUE {ALIGNED} : bool;
  END_VAR
  VAR
    BAC_OUT_OF_SERVICE {ALIGNED} : bool;
    BAC_NORMAL {ALIGNED} : bool;
    BAC_ALARM {ALIGNED} : bool;
    BAC_FAULT {ALIGNED} : bool;
    BAC_PRESENT_VALUE {ALIGNED} : bool;
    BAC_EVENT_STATE : byte;
  END_VAR
 END_FUNCTION_BLOCK
```

5.4 PŘÍKLADY UŽIVATELSKÝCH PROGRAMŮ S OBJEKTY BACnet

5.4.1 Příklad ovládání parametrů čítače pomocí objektů BACnet

Předpokládejme úlohu ve které potřebujeme u obousměrného čítače zadávat předvolbu, resetovat čítač a sledovat skutečnou hodnotu a výstupy čítače. Požadujeme aby tyto parametry byly ovládané a viditelné v prostředí BACnet, tím myslíme, že mohou být ovládané a viditelné z operátorské konzole, nebo jiných řídících systémů s protokolem BACnet. Tuto situaci znázorňuje následující obrázek.



Ukázka programu s čítačem CTUD bez „ovládání“ přes objekty BACnet

```
VAR_GLOBAL
  pulseUP     : BOOL;
  pulseDOWN   : BOOL;
  resetCTUD   : BOOL;
  limitUP     : BOOL;
  limitDOWN   : BOOL;
  presentCTUD : INT;
  counterCTUD : CTUD;
END_VAR

PROGRAM Example_CTUD1

  counterCTUD(
    CU := pulseUP,
    CD := pulseDOWN,
    R  := resetCTUD,
    PV := presentCTUD,
    QU => limitUP,
    QD => limitDOWN);

END_PROGRAM
```

Nyní provedeme deklarace objektů BACnet.
Binární hodnoty:

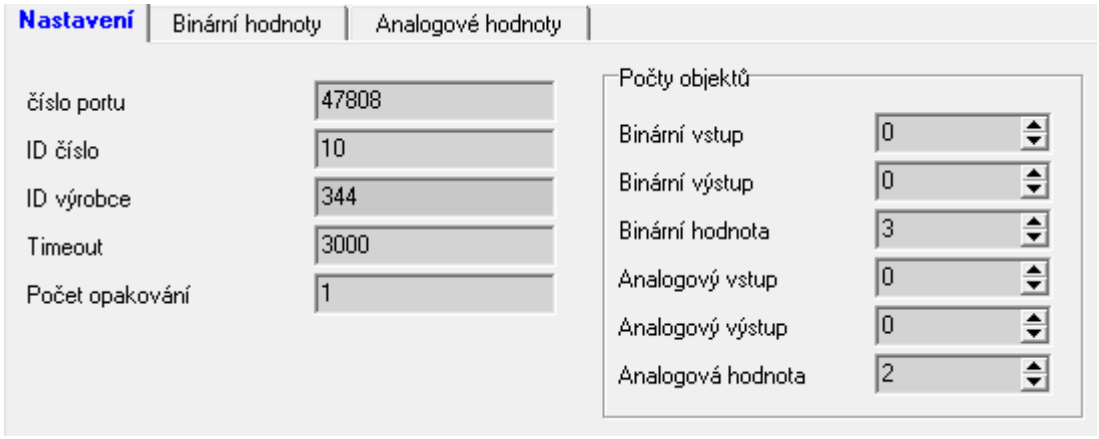
- reset čítače
- výstup čítače nahoru
- výstup čítače dolů

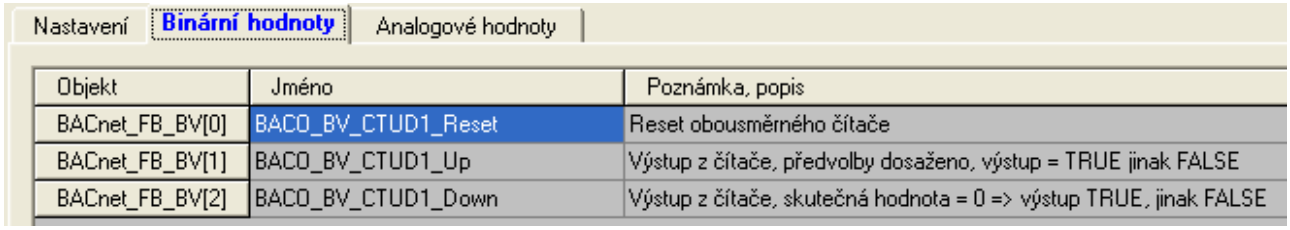
celkem 3

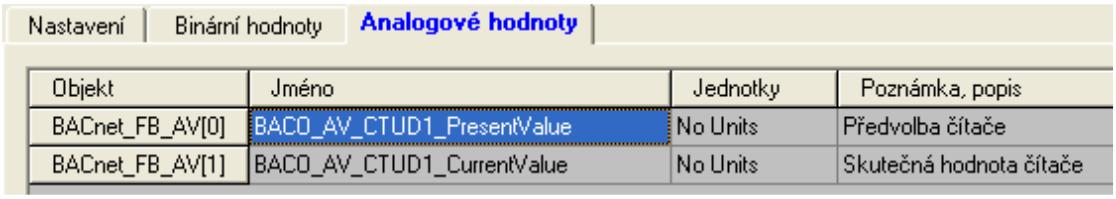
Analogové hodnoty:

- předvolba čítače
- hodnota čítače
  celkem 2

Deklaraci provedeme dle kroků uvedených v kapitole 4.2.





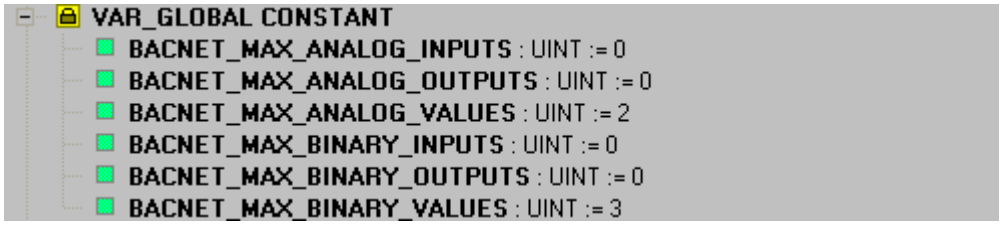


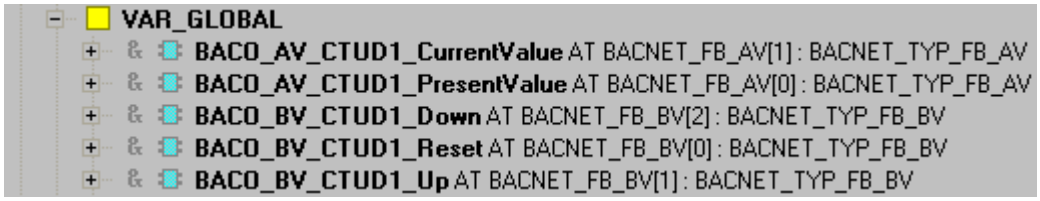
Poznámka:

Doporučujeme věnovat zvýšenou pozornost jménům objektů BACnet, volte jména tak, aby bylo zřejmé, že se jedná o objekty BACnet, ke kterému funkčnímu bloku vašeho programu se váží a co ovlivňují.

Např. BACO_BV_CTUD1_Reset

Po úspěšném překladu můžeme nyní vidět v inspektoru proměnných námi deklarované objekty BACnet.





Dále je automaticky vývojovým prostředím vytvořen soubor HWConfig.ST, který obsahuje potřebné deklarace viz ukázka dále. Zajímavá je poslední část deklarace která deklaruje námi definované objekty, které můžeme použít uživatelském programu.

```
VAR_GLOBAL CONSTANT
  BACNET_MAX_BINARY_INPUTS    : UINT := 0;
  BACNET_MAX_BINARY_OUTPUTS   : UINT := 0;
  BACNET_MAX_BINARY_VALUES    : UINT := 3;
  BACNET_MAX_ANALOG_INPUTS    : UINT := 0;
  BACNET_MAX_ANALOG_OUTPUTS   : UINT := 0;
  BACNET_MAX_ANALOG_VALUES    : UINT := 2;
END_VAR

VAR_GLOBAL
  BACNET_FB_BV : ARRAY[0..BACNET_MAX_BINARY_VALUES-1] OF T_BACNET_TYP_FB_BV := [
        ( IN_OBJECT_NAME := 'BACO_BV_CTUD1_Reset', IN_DESCRIPTION := 'Reset obousměrného čítače' ),
        ( IN_OBJECT_NAME := 'BACO_BV_CTUD1_Up', IN_DESCRIPTION := 'Výstup z čítače, předvolby
dosaženo, výstup = TRUE jinak FALSE' ),
        ( IN_OBJECT_NAME := 'BACO_BV_CTUD1_Down', IN_DESCRIPTION := 'Výstup z čítače, skutečná
hodnota = 0 => výstup TRUE, jinak FALSE' ) ];

  BACNET_FB_AV : ARRAY[0..BACNET_MAX_ANALOG_VALUES-1] OF T_BACNET_TYP_FB_AV := [
        ( IN_OBJECT_NAME := 'BACO_AV_CTUD1_PresentValue', IN_DESCRIPTION := 'Předvolba čítače',
IN_UNITS := BACNET_UNITS_NO_BACNET_UNITS ),
        ( IN_OBJECT_NAME := 'BACO_AV_CTUD1_CurrentValue', IN_DESCRIPTION := 'Skutečná hodnota
čítače', IN_UNITS := BACNET_UNITS_NO_BACNET_UNITS ) ];

BACO_BV_CTUD1_Reset             AT BACNET_FB_BV[0] : BACNET_TYP_FB_BV;
BACO_AV_CTUD1_PresentValue      AT BACNET_FB_AV[0] : BACNET_TYP_FB_AV;
BACO_BV_CTUD1_Up                AT BACNET_FB_BV[1] : BACNET_TYP_FB_BV;
BACO_AV_CTUD1_CurrentValue      AT BACNET_FB_AV[1] : BACNET_TYP_FB_AV;
BACO_BV_CTUD1_Down              AT BACNET_FB_BV[2] : BACNET_TYP_FB_BV;

END_VAR
```

Ukázka modifikace programu s čítačem CTUD s „ovládáním“ přes objekty BACnet

```
VAR_GLOBAL
  pulseUP     : BOOL;
  pulseDOWN   : BOOL;
  resetCTUD   : BOOL;
  limitUP     : BOOL;
  limitDOWN   : BOOL;
  presentCTUD : INT;

  counterCTUD : CTUD;
END_VAR

PROGRAM Example_CTUD1
  BACO_AV_CTUD1_CurrentValue();
  BACO_AV_CTUD1_PresentValue();
  BACO_BV_CTUD1_Down();
  BACO_BV_CTUD1_Reset();
  BACO_BV_CTUD1_Up();

  // naplnění parametrů z BACnet objektů
  resetCTUD    := BACO_BV_CTUD1_Reset.OUT_PRESENT_VALUE;
  presentCTUD  := REAL_TO_INT(BACO_AV_CTUD1_PresentValue.OUT_PRESENT_VALUE);
```

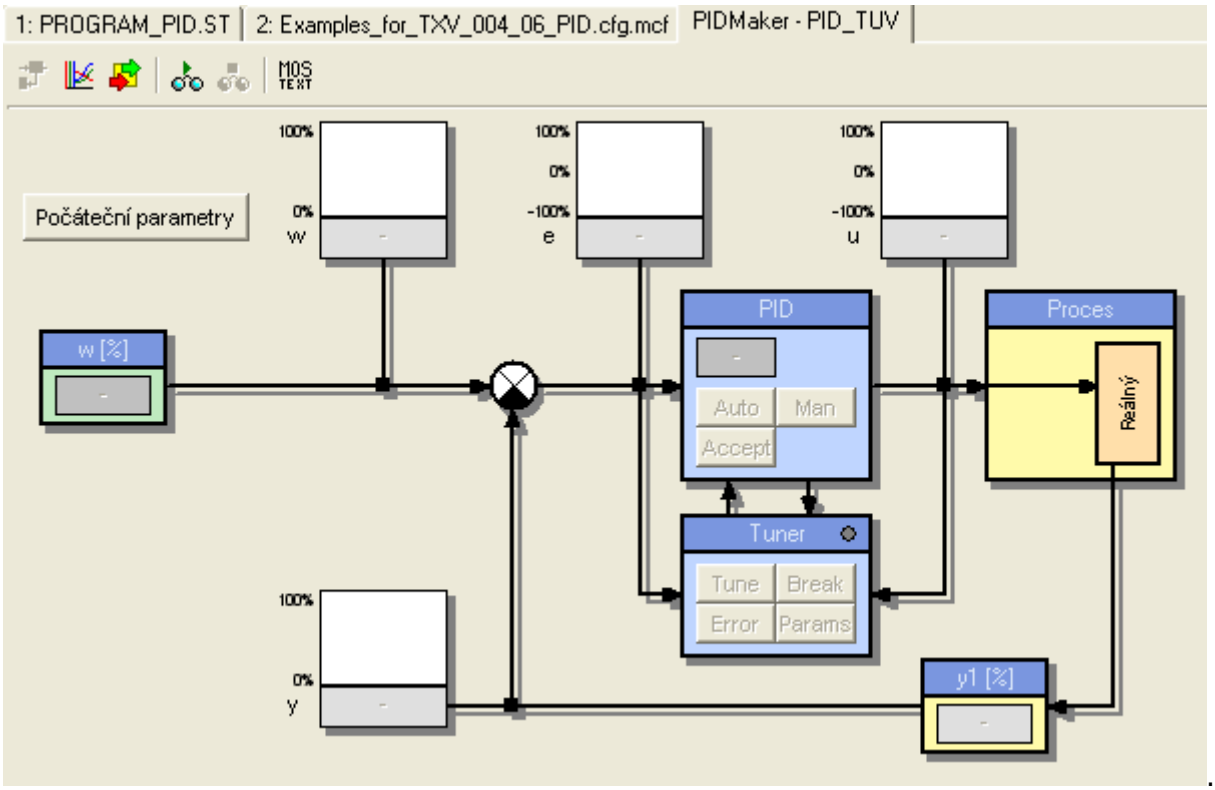
```
  counterCTUD(
    CU := pulseUP,
    CD := pulseDOWN,
    R  := resetCTUD,
    PV := presentCTUD,
    QU => limitUP,
    QD => limitDOWN);

  // převzetí výstupu čítače do BACnet objektů
  BACO_AV_CTUD1_CurrentValue.OUT_PRESENT_VALUE := INT_TO_REAL(counterCTUD.CV);
  BACO_BV_CTUD1_UP.OUT_PRESENT_VALUE           := limitUP;
  BACO_BV_CTUD1_DOWN.OUT_PRESENT_VALUE         := limitDOWN;

END_PROGRAM
```

Případně můžeme výše uvedený kód zjednodušit, tak že objekty BACnet použijeme přímo jako parametry funkčního bloku counterCTUD.

```
VAR_GLOBAL
  pulseUP     : BOOL;
  pulseDOWN   : BOOL;
  counterCTUD : CTUD;
END_VAR

PROGRAM Example_CTUD1
  BACO_AV_CTUD1_CurrentValue();
  BACO_AV_CTUD1_PresentValue();
  BACO_BV_CTUD1_Down();
  BACO_BV_CTUD1_Reset();
  BACO_BV_CTUD1_Up();

  // použití objektů BACnet přímo jako parametrů funkčního bloku

  counterCTUD(
    CU := pulseUP,
    CD := pulseDOWN,
    R  := BACO_BV_CTUD1_Reset.OUT_PRESENT_VALUE,
    PV := REAL_TO_INT(BACO_AV_CTUD1_PresentValue.OUT_PRESENT_VALUE),
    QU => BACO_BV_CTUD1_UP.OUT_PRESENT_VALUE,
    QD => BACO_BV_CTUD1_DOWN.OUT_PRESENT_VALUE);

  // převzetí výstupu čítače do BACnet objektu
  BACO_AV_CTUD1_CurrentValue.OUT_PRESENT_VALUE := INT_TO_REAL(counterCTUD.CV);

END_PROGRAM
```

5.4.2 Příklad ovládání parametrů regulátoru pomocí objektů BACnet

Předpokládejme úlohu ve které potřebujeme u regulátoru zadávat žádanou hodnotu, sledovat skutečnou hodnotu, přepínat mezi ručním a automatickým řízením a v ručním řízení ovládat výstup regulátoru. Požadujeme aby tyto parametry byly ovládané a viditelné v prostředí BACnet, tím myslíme, že mohou být ovládané a viditelné z operátorské konzole, nebo jiných řídících systémů. Tuto situaci znázorňuje následující obrázek (výstup z nástroje PIDMaker, který je součástí vývojového prostření MOSAIC)



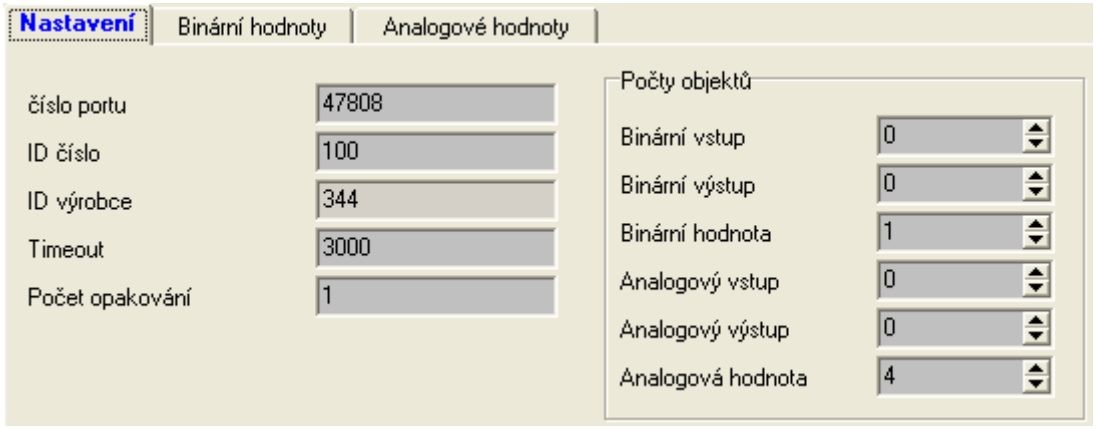
Nyní provedeme deklarace objektů BACnet.
Binární hodnoty:

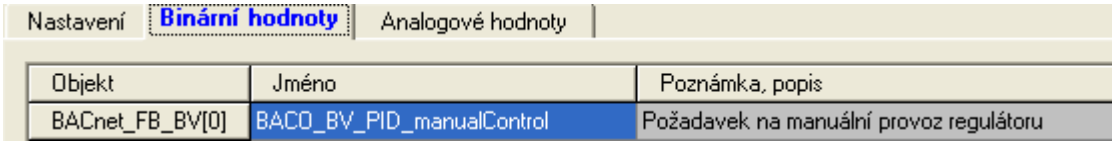
- požadavek na manuální provoz regulátoru (vstup do regulátoru)
  celkem 1

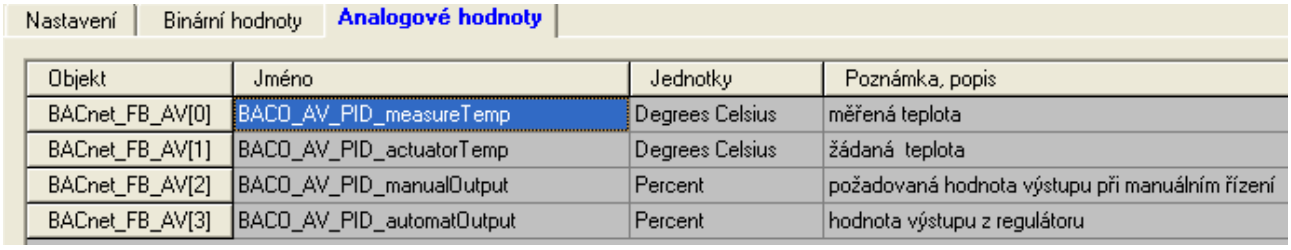
Analogové hodnoty:

- měřená teplota (vstup do regulátoru)
- žádaná teplota (vstup do regulátoru)
- akční zásah (výstup z regulátoru)
- požadovaný akční zásah v manuálním provozu regulátoru
  celkem 4

Deklaraci provedeme dle kroků uvedených v kapitole 4.2.





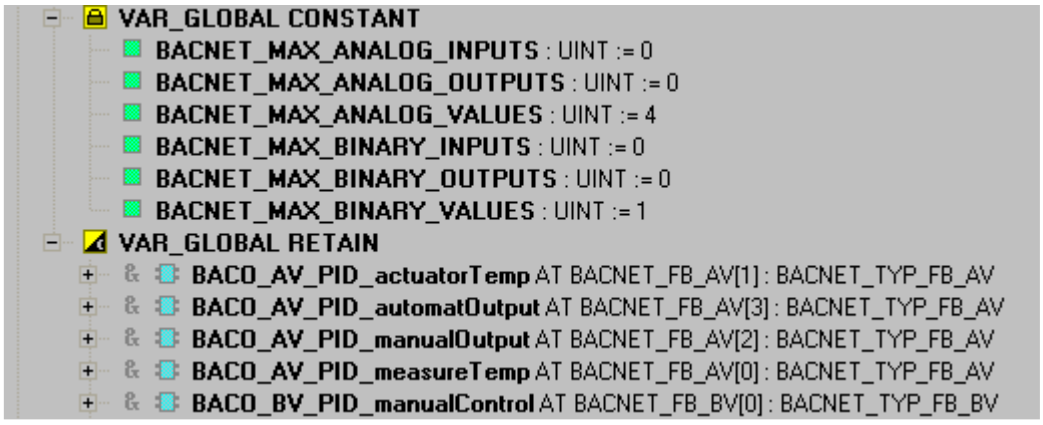


Poznámka:

Doporučujeme věnovat zvýšenou pozornost jménům objektů BACnet, volte jména tak, aby bylo zřejmé, že se jedná o objekty BACnet, ke kterému funkčnímu bloku vašeho programu se váží a co ovlivňují.

Např. BACO_BV_PID_manualControl

Po úspěšném překladu můžeme nyní vidět v inspektoru proměnných námi deklarované objekty BACnet.



Dále je automaticky vývojovým prostředím vytvořen soubor HWConfig.ST, který obsahuje potřebné deklarace viz ukázka dále. Zajímavá je poslední část deklarace která deklaruje námi definované objekty, které můžeme použít uživatelském programu.

```
VAR_GLOBAL CONSTANT
  BACNET_MAX_BINARY_INPUTS    : UINT := 0;
  BACNET_MAX_BINARY_OUTPUTS   : UINT := 0;
  BACNET_MAX_BINARY_VALUES    : UINT := 1;
  BACNET_MAX_ANALOG_INPUTS    : UINT := 0;
  BACNET_MAX_ANALOG_OUTPUTS   : UINT := 0;
  BACNET_MAX_ANALOG_VALUES    : UINT := 4;
END_VAR

VAR_GLOBAL RETAIN
  BACNET_FB_BV : ARRAY[0..BACNET_MAX_BINARY_VALUES-1] OF T_BACNET_TYP_FB_BV := [
  ( IN_OBJECT_NAME := 'BACO_BV_PID_manualControl', IN_DESCRIPTION := 'Požadavek na manuální provoz
regulátoru' ) ];

  BACNET_FB_AV : ARRAY[0..BACNET_MAX_ANALOG_VALUES-1] OF T_BACNET_TYP_FB_AV := [
  ( IN_OBJECT_NAME := 'BACO_AV_PID_measureTemp', IN_DESCRIPTION := 'měřená teplota', IN_UNITS :=
BACNET_UNITS_DEGREES_CELSIUS ),
  ( IN_OBJECT_NAME := 'BACO_AV_PID_actuatorTemp', IN_DESCRIPTION := 'žádaná  teplota', IN_UNITS :=
BACNET_UNITS_DEGREES_CELSIUS ),
  ( IN_OBJECT_NAME := 'BACO_AV_PID_manualOutput', IN_DESCRIPTION := 'požadovaná hodnota výstupu při
manuálním řízení', IN_UNITS := BACNET_UNITS_PERCENT ),
  ( IN_OBJECT_NAME := 'BACO_AV_PID_automatOutput', IN_DESCRIPTION := 'hodnota výstupu z regulátoru',
IN_UNITS := BACNET_UNITS_PERCENT ) ];

  BACO_BV_PID_manualControl   AT BACNET_FB_BV[0] : BACNET_TYP_FB_BV;
  BACO_AV_PID_measureTemp     AT BACNET_FB_AV[0] : BACNET_TYP_FB_AV;
  BACO_AV_PID_actuatorTemp    AT BACNET_FB_AV[1] : BACNET_TYP_FB_AV;
  BACO_AV_PID_manualOutput    AT BACNET_FB_AV[2] : BACNET_TYP_FB_AV;
  BACO_AV_PID_automatOutput   AT BACNET_FB_AV[3] : BACNET_TYP_FB_AV;
END_VAR
```

Ukázka modifikace programu s PID a „ovládáním“ přes objekty BACnet

```
PROGRAM PROGRAM_PID

  // volání objektů BACnet
  BACO_AV_PID_actuatorTemp();   // žádaná teplota
  BACO_AV_PID_automatOutput();  // výstup regulátoru
  BACO_AV_PID_manualOutput();   // požadovaný výstup regulátoru v ručním řízení
  BACO_AV_PID_measureTemp();    // měřená teplota
  BACO_BV_PID_manualControl();  // požadavek na ruční řízení

  // přiřazení objektů BACnet do parametrů regulátoru
  PID_TUV_MAN.0 := BACO_BV_PID_manualControl.OUT_PRESENT_VALUE;
  PID_TUV_sp    := BACO_AV_PID_actuatorTemp.OUT_PRESENT_VALUE;

  // přiřezení proměnných do objektů BACnet
  // měřená teplota
  BACO_AV_PID_measureTemp.IN_PRESENT_VALUE   := measureTemp;
  // výstup z regulátoru = akční zásah
  BACO_AV_PID_automatOutput.IN_PRESENT_VALUE := automatOutput;

  // pro ovládání výstupu regulátoru v ručním řízení
  PID_TUV_hv    := BACO_AV_PID_manualOutput.OUT_PRESENT_VALUE;

END_PROGRAM
```

Objednávky a informace:
Teco a. s. Havlíčkova 260, 280 58 Kolín 4, tel. 321 737 611, fax 321 737 633

Výrobce si vyhrazuje právo na změny dokumentace. Poslední aktuální vydání je k dispozici na internetu www.tecomat.cz