Knihovna iControlLib

TXV 003 59.01
deváté vydání
Březen 2023
změny vyhrazeny

Historie změn

Datum	Vydání	Popis změn
Prosinec 2014	1	První vydání, popis odpovídá iControlLib_v10
Únor 2017	2	Druhé vydání, popis odpovídá iControlLib_v20
Prosinec 2018	3	Třetí vydání, popis odpovídá iControlLib_v22
Září 2019	4	Čtvrté vydání, popis odpovídá iControlLib_v24
Říjen 2019	5	Páté vydání, doplněn popis pro HomeKit
Listopad 2020	6	Šesté vydaní, popis odpovídá iControlLib_v29
Červenec 2021	7	Sedmé vydání, popis odpovídá iControlLib_v30
Říjen 2022	8	Osmé vydání, doplněn popis bloků fb_iDisplay_TimeEdit, fb_iDisplay_DateEdit a fb_iDimmerWWCW, popis odpovídá iControlLib_v32
Březen 2023	9	Deváté vydání, doplněn popis typů pro definici hierarchie domácnosti, nové parametry pro uživatelské aplikace, přidána nová kapitola 10 Podpora dalších platforem, popis odpovídá iControlLib_v33

OBSAH

1 ÚVOD

Knihovna iControlLib obsahuje funkční bloky využitelné především v inteligentních domech. Tyto bloky jsou připraveny pro ovládání z webového rozhraní a pro snadnou integraci s uživatelskými aplikacemi (iFoxtrot, HomeKit, Google Home, ...).

Knihovna iControlLib je standardně dodávaná jako součást programovacího prostředí Mosaic. Pro knihovnu iControlLib_v20 a vyšší je potřebný Mosaic verze v2017.1, aplikace iFoxtrot musí být verze v2.4.1 nebo vyšší.

Knihovna iControlLib není podporovaná na systémech TC-650, u systému TC700 nelze knihovnu použít s procesorovými moduly CP-7002, CP-7003 a CP-7005.

Funkce z knihovny iControlLib v2.0 jsou podporovány v centrálních jednotkách řady K a L (TC700 CP-7004 a CP-7007, všechny varianty systému Foxtrot 1) od verze v9.8. Na centrálních jednotkách řady I (systémy Foxtrot 2) lze knihovnu použít ve všech verzích. Knihovna nevyžaduje aplikační profil.

Objednací číslo dokumentace ke knihovně iControlLib je TXV 003 59.01.

2 DATOVÉ TYPY

V knihovně iControlLib jsou definovány následující datové typy:

Typ	*Popis*
T_SET_RGB_LIGHT	Struktura obsahující hodnoty nastavení pro stmívané RGB světlo
TStructure	Deklarace názvu domácnosti
TRoom	Deklarace názvu místnosti

2.1 Typ T_SET_RGB_LIGHT

Knihovna : *iControlLib*

Struktura *T_SET_RGB_LIGHT* slouží k uchování parametrů pro stmívané barevné světlo. Význam jednotlivých položek struktury je následující:

- *level* úroveň jasu zapnutého světla <0,100> %
- *RGB.red* červená složka <0,255>
- *RGB.green* zelená složka <0,255>
- *RGB.blue* modrá složka <0,255>
- *RGB.opacity* průhlednost <0,255> (0 = neprůhledné, 255 = plně průhledné)

Struktura *T_SET_RGB_LIGHT* je používána blokem *fb_iDimmerRGB*.

2.2 Typy pro popis hierarchie domácnosti TStructure a TRoom

V knihovně iControlLib jsou od verze 3.3 definovány speciální datové typy, které umožňují pro uživatelskou aplikaci popsat hierarchii domácnosti, tedy název domácnosti a názvy místností, ve kterých jsou umístěny řídicí prvky.

Deklarací proměnných těchto typů se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje popisná proměnná, která slouží pro integraci s uživatelskou aplikací.

TStructure

	Proměnná	**Typ**	**Význam**
R	*GTSAP1_STRUCTURE_name*	STRING[48]	*Označení domácnosti*

TRoom

	Proměnná	**Typ**	**Význam**
R	*GTSAP1_ROOM_name*	STRING[48]	*Označení místnosti*

Pro zařazení prvku z iControlLib pod deklarovanou hierarchickou strukturu, tedy domácnost nebo místnost, je využíváno shody ve jmenných cestách publikovaných proměnných.

Použití si ukážeme na následujícím příkladu:

Vytvoříme základní program a deklarujeme globální proměnnou typu *TStructure*, ve které inicializujeme parametr *name* na hodnotu odpovídající názvu naší domácnosti.

```
VAR_GLOBAL
  Structure : TStructure := ( name := 'Home' );
END_VAR

PROGRAM prgMain
  VAR_INPUT
  END_VAR
  VAR_OUTPUT
  END_VAR
  VAR
```

```
  END_VAR
  VAR_TEMP
  END_VAR

END_PROGRAM
```

Vytvoříme další programovou instanci (např. *prgKitchen*). V programu deklarujeme proměnné typu *TRoom* a *fb_iLight*. Parametr *name* ve struktuře *TRoom* inicializujeme na hodnotu odpovídající názvu naší místnosti.

```
PROGRAM prgKitchen
  VAR_INPUT
  END_VAR
  VAR_OUTPUT
  END_VAR
  VAR
     Room : TRoom := ( name := 'Kitchen' );
     Light : fb_iLight;
  END_VAR
  VAR_TEMP
  END_VAR

  Light();

END_PROGRAM
```

V generovaném souboru „iFoxtrot.pub“ (public soubor) se pak budou nacházet tyto deklarace proměnných:

```
Structure.GTSAP1_STRUCTURE_name R B 226 STRING[48]
Kitchen.Room.GTSAP1_ROOM_name R B 276 STRING[48]
Kitchen.Light.GTSAP1_LIGHT_name R B 331 STRING[48]
Kitchen.Light.GTSAP1_LIGHT_enable R B 380 .1 BOOL
Kitchen.Light.GTSAP1_LIGHT_needAck R B 380 .2 BOOL
Kitchen.Light.GTSAP1_LIGHT_needPin R B 381 STRING[10]
Kitchen.Light.GTSAP1_LIGHT_type R B 392 .0 BOOL
Kitchen.Light.GTSAP1_LIGHT_onoff R B 392 .1 BOOL
```

Námi deklarovaná globální proměnná *Structure* není uvozena žádnou jmennou cestou. Je tedy na vrcholu hierarchie a všechny ostatní deklarace budou spadat pod tuto domácnost, jejíž označení je dáno parametrem *name*. Další deklarovanou proměnnou je definice místnosti *Room*. Proměnná *Room* je uvozena jmennou cestou *Kitchen*, neboť tato proměnná je deklarována ve stejnojmenné instanci programu. Vše, co bude mít stejný jmenný prefix bude tedy zařazenou do této místnosti. Její označení je dáno parametrem *name. V našem případě se jedná o světlo Light*, které je deklarováno ve stejné programové instanci jako *Room* a má tudíž stejný jmenný prefix.

Takto definovanou hierarchii pak mohou aplikace používající data z „iFoxtrot.pub“ souboru využít k automatickému roztřídění prvků z iControlLib.

3 KONSTANTY

Konstanty definované v knihovně iControlLib jsou určeny pro vlastní potřeby knihovny.

4 GLOBÁLNÍ PROMĚNNÉ

V knihovně iControlLib nejsou definovány žádné globální proměnné.

5 FUNKCE

V knihovně iControlLib nejsou definovány žádné funkce.

6 FUNKČNÍ BLOKY

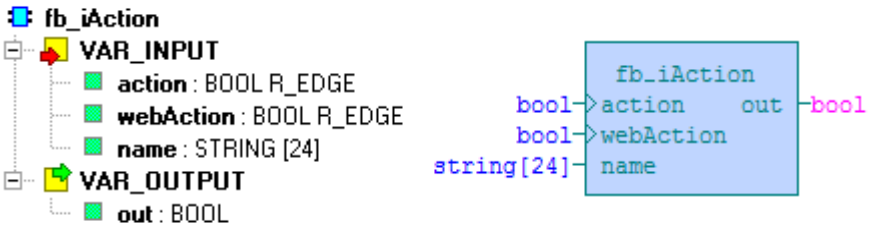
V knihovně iControlLib jsou definovány následující funkční bloky:

Funkční blok	Popis
fb_iAction	Zachycení požadavku na akci
fb_iButton	Zveřejnění tlačítka pro uživatelskou aplikaci
fb_iContact	Testování kontaktu (stavu BOOL proměnné)
fb_iDimmer	Řízení stmívaného světla (0..100%)
fb_iDimmerLED	Řízení stmívaného LED světla (0..100%)
fb_iDimmerRGB	Řízení stmívaného RGB světla (0..100%)
fb_iDimmerWWCW	Řízení stmívaného dvoukanálového bílého světla
fb_iDisplay_OnOff	Zobrazení BOOL hodnoty v uživatelské aplikaci
fb_iDisplay_Val	Zobrazení REAL hodnoty v uživatelské aplikaci
fb_iDisplay_Edit	Zobrazení a změna REAL hodnoty v uživatelské aplikaci
fb_iDisplay_TimeEdit	Zobrazení a změna TIME hodnoty v uživatelské aplikaci
fb_iDisplay_DateEdit	Zobrazení a změna DATE hodnoty v uživatelské aplikaci
fb_iFan	Ovládání ventilátoru či obecného výstupu (0..100%)
fb_iJalousie	Ovládání žaluzií
fb_iLight	Řízení spínaného světla
fb_iOpener	Řízení systémů pro otevírání dveří, bran, vrat nebo rolet

Funkční blok	Popis
fb_iPublicEncoding	Zveřejnění kódování textů pro uživatelskou aplikaci
fb_iRelay	Řízení relé (BOOL výstupu)
fb_iScene4	Nastavení jedné ze 4 scén
fb_iScene8	Nastavení jedné z 8 scén
fb_iSensorCO	Měření CO s vyhodnocením překročení zadané meze
fb_iSensorCO2	Měření CO2 s vyhodnocením překročení zadané meze
fb_iSensorLight	Měření intenzity osvětlení (lx)
fb_iSensorPIR	Detekce pohybu
fb_iSensorSmoke	Měření kouře (ppm) s vyhodnocením překročení zadané meze
fb_iSensorTemp	Měření teploty
fb_iSensorHumidity	Měření vlhkosti
fb_iSocket	Ovládání zásuvky
fb_iThermostat_cool	Termostat pro chlazení
fb_iThermostat_heat	Termostat pro topení
fb_iThermostat_heatCool	Termostat pro topení a chlazení
fb_iTherm	Měření teploty – nahrazeno blokem *fb_iSensorTemp*
fb_iTimeProgWeek	Ukládání a načítání časového programu
fb_iWebCamera	Zveřejnění web kamery pro uživatelskou aplikaci
fb_iWebConf	Zveřejnění web stránky pro uživatelskou aplikaci

6.1 Funkční blok fb_iAction

Knihovna : *iControlLib*



Funkční blok *fb_iAction* slouží k zachycení požadavku na nějakou akci. Vstup *action* slouží pro připojení požadavků vznikajících v PLC (například stisknutí tlačítka připojeného na binární vstup PLC nebo nastavení BOOL proměnné v PLC programu na hodnotu TRUE). Vstup *webAction* slouží k zachycení požadavků z web rozhraní (tuto proměnnou je třeba nastavit na TRUE v případě, že chceme vyvolat akci z web stránky). Uživatelská aplikace musí pro vyvolání akce nastavit hodnotu TRUE do proměnné *GTSAP1_ACTION_state*. Vstup *name* slouží k pojmenování akce.

Výstup *out* je nastaven na TRUE v případě, že se na kterémkoliv ze vstupů změní hodnota z FALSE na TRUE. Jinými slovy náběžná hrana na kterémkoliv vstupu nastaví výstup *out* na dobu jednoho cyklu PLC. Vstupy *webAction* a *GTSAP1_ACTION_state* jsou po zachycení náběžné hrany automaticky vynulovány.

Popis proměnných :

	Proměnná	***Typ***	***Význam***
VAR_INPUT			
	action	BOOL R_EDGE	Žádost o vyvolání akce z PLC Náběžná hrana této proměnné nastaví výstup *out*
	webAction	BOOL R_EDGE	Žádost o vyvolání akce z web stránky Náběžná hrana této proměnné nastaví výstup *out*
	name	STRING[48]	Pojmenování bloku pro uživatelskou aplikaci
VAR_OUTPUT			
	out	BOOL	TRUE pokud přišel požadavek na akci na některém ze vstupů, jinak FALSE

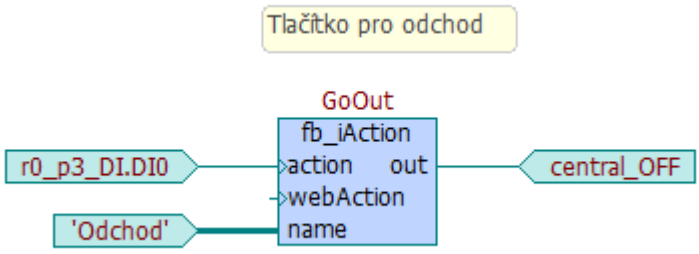
Příkladem použití funkčního bloku *fb_iAction* může být například centrální zhasnutí světel v domě. Tuto akci chceme vyvolávat jak pomocí odchodového tlačítka, tak z web stránky a také z uživatelské aplikace. V jazyce ST bude program vypadat následovně:

```
VAR_GLOBAL
  central_OFF : BOOL;
END_VAR
```

```
PROGRAM prgMain
  VAR
    GoOut : fb_iAction;
  END_VAR

  GoOut( action := r0_p3_DI.DI0, out => central_OFF);
END_PROGRAM
```

V programu je založena instance funkčního bloku *fb_iAction* s názvem *GoOut*. Program předpokládá, že odchodové tlačítko je připojeno na binární vstup r0_p3_DI.DI0. Výstup funkčního bloku *GoOut* je zapisován do globální proměnné *central_OFF*. Tato proměnná bude pak dále použita pro vstupy všech bloků s funkcí centrálního zhasnutí.

Stejnou funkci jako v předcházejícím příkladu lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iAction* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

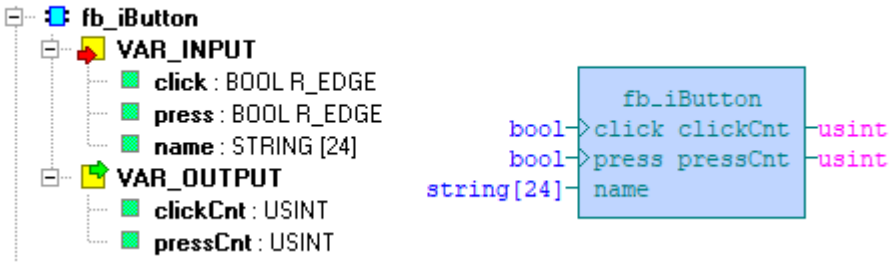
Každá instance *fb_iAction* přidá následující public proměnné:

	Proměnná	**Typ**	**Význam**
R	*GTSAP1_ACTION_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_ACTION_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_ACTION_needAck*	BOOL	Nastavením na TRUE bude pro změnu stavu bloku vyžadováno potvrzení. Změnit volbu lze pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
W	*GTSAP1_ACTION_exec*	BOOL	Žádost o vyvolání akce z uživatelské aplikace

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.2 Funkční blok fb_iButton

Knihovna : *iControlLib*



Funkční blok *fb_iButton* slouží pro zveřejnění krátkého a dlouhého stisku tlačítka do mobilní aplikace (např. HomeKit). Vstupy *click* a *press* slouží pro připojení informací o krátkém resp. dlouhém stisku (například z CIB jednotky C-WS-0400R apod.). Do mobilní aplikace jsou místo těchto vstupů předávány čítače krátkých resp. dlouhých stisků. Vstup *name* slouží k pojmenování tlačítka (pod tímto jménem bude viditelné v aplikaci).

Výstupy *clickCnt a pressCnt* počítají náběžné hrany na vstupech *click* a *press*. Každá změna těchto čítačů je aplikací vyhodnocena jako krátký resp. dlouhý stisk tlačítka.

Popis proměnných :

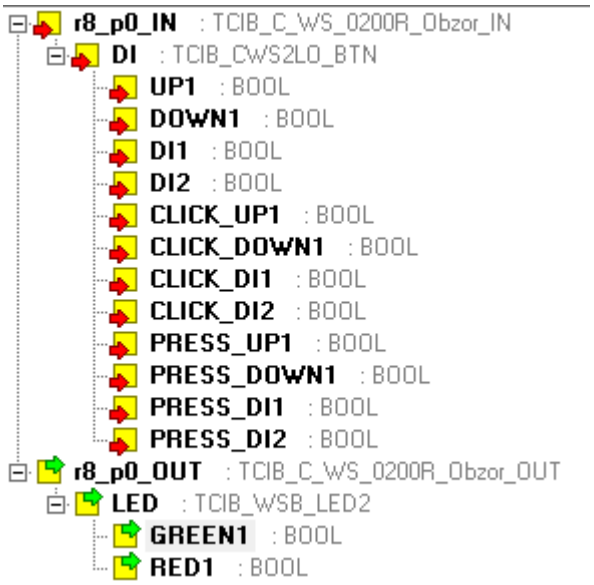
	Proměnná	Typ	Význam
VAR_INPUT			
	click	BOOL R_EDGE	Krátký stisk tlačítka Náběžná hrana této proměnné zvýší o 1 hodnotu výstupu *clickCnt*
	press	BOOL R_EDGE	Dlouhý stisk tlačítka Náběžná hrana této proměnné zvýší o 1 hodnotu výstupu *pressCnt*
	name	STRING[48]	Pojmenování bloku pro mobilní aplikaci
VAR_OUTPUT			
	clickCnt	BOOL	Čítač náběžných hran vstupu *click*
	pressCnt	BOOL	Čítač náběžných hran vstupu *press*

Příkladem použití funkčního bloku *fb_iButton* může být například zveřejnění tlačítek z CIB modulu C-WS-0200R-Obzor, kterými budeme v uživatelské aplikaci ovládat zařízení třetí strany. V jazyce ST bude program vypadat následovně:

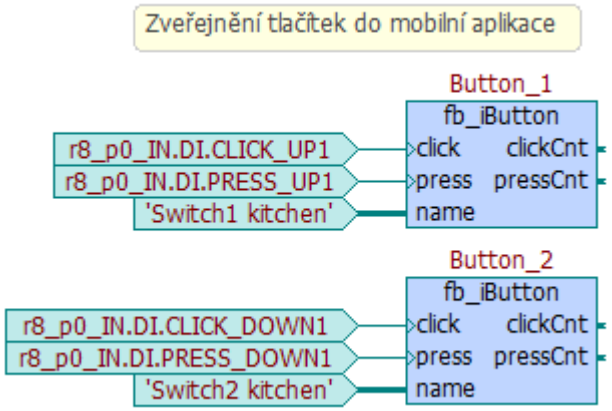
```
PROGRAM prgMain
  VAR
    Button_1 : fb_iButton := ( name := 'Switch1 kitchen');
    Button_2 : fb_iButton := ( name := 'Switch2 kitchen');
  END_VAR

  Button_1( click := r8_p0_IN.DI.CLICK_UP1,   press := r8_p0_IN.DI.PRESS_UP1);
  Button_2( click := r8_p0_IN.DI.CLICK_DOWN1, press := r8_p0_IN.DI.PRESS_DOWN1);
```

```
END_PROGRAM
```

V programu jsou založeny dvě instance funkčního bloku *fb_iButton* s názvem *Button_1 a Button_2*. Při volání těchto instancí jsou jako parametry předávány krátké a dlouhé stisky poskytované modulem C-WS-0200R-Obzor. Pro úplnost uveďme jak vypadá celá datová struktura poskytovaná modulem C-WS-0200R-Obzor:



Stejnou funkci jako v předcházejícím příkladu lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iButton* se do souboru „iFox-trot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

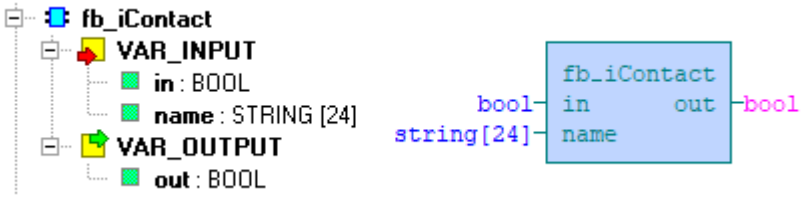
Každá instance *fb_iButton* přidá následující public proměnné:

	Proměnná	Typ	Význam
R	*GTSAP1_BUTTON_name*	STRING[48]	Název bloku (kopie vstupu *name*)
RW	*GTSAP1_BUTTON_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Fox-trot 2 zálohována.
R	*GTSAP1_BUTTON_clickCnt*	BOOL	Počet krátkých stisků
R	*GTSAP1_BUTTON_pressCnt*	BOOL	Počet dlouhých stisků

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.3 Funkční blok fb_iContact

Knihovna : *iControlLib*



Funkční blok *fb_iContact* slouží k signalizaci stavu kontaktu (proměnné typu BOOL) do mobilní aplikace. Kontakt se připojuje na vstup *in*. Vstup *name* slouží k pojmenování kontaktu.

Výstup *out* kopíruje stav vstupu *in*.

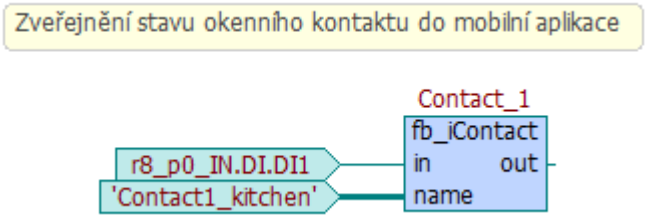
Popis proměnných :

	Proměnná	*Typ*	*Význam*
VAR_INPUT			
	in	BOOL	Stav kontaktu (FALSE rozepnuto, TRUE sepnuto)
	name	STRING[48]	Pojmenování bloku pro mobilní aplikaci
VAR_OUTPUT			
	out	BOOL	Kopie vstupu *in*

Předpokládejme, že chceme zveřejnit stav okenního kontaktu od okna v kuchyni v mobilní aplikaci. V jazyce ST bude program vypadat následovně:

```
PROGRAM prgMain
  VAR
    Contact_1 : fb_iContact := ( name := 'Contact1_kitchen');
  END_VAR

  Contact_1(in := r8_p0_IN.DI.DI1);
END_PROGRAM
```

nebo v jazyce CFC



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iContact* se do souboru „iFox-trot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

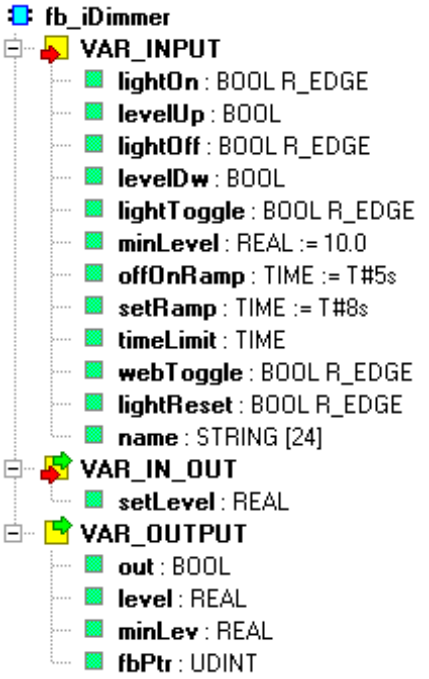
Každá instance *fb_iContact* přidá následující public proměnné:

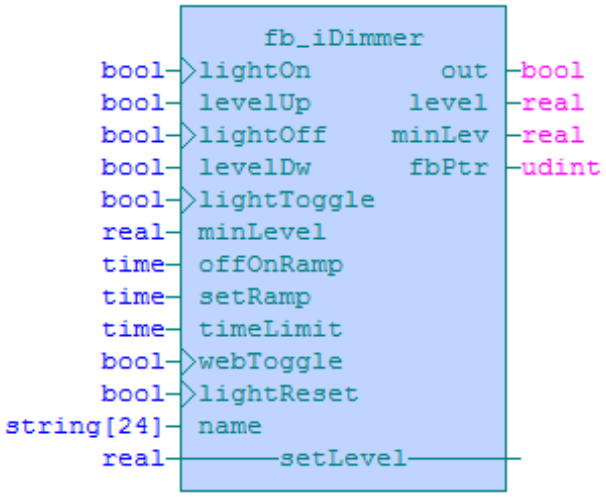
	Proměnná	**Typ**	**Význam**
R	*GTSAP1_CONTACT_name*	STRING[48]	Název bloku (kopie vstupu *name*)
RW	*GTSAP1_CONTACT_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Fox-trot 2 zálohována.
R	*GTSAP1_CONTACT_state*	BOOL	Stav kontaktu

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.4 Funkční blok fb_iDimmer

Knihovna : *iControlLib*





Funkční blok *fb_iDimmer* je určen k řízení stmívaného světla v rozsahu 0...100%. Výstup bloku je vhodný pro řízení světla např. převodníkem 0...10V.

Vstup *lightOn* slouží k rozsvícení světla na úroveň nastavenou parametrem *setLevel*. Světlo se rozsvítí plynule, rychlost rozsvícení určuje vstup *offOnRamp*. Ten udává celkový čas, za který se světlo rozsvítí z *minLevel* na 100%. Vstup *minLevel* určuje minimální úroveň, při které světlo ještě svítí (ta odpovídá např. zapalovacímu napětí zářivek). Vstup *lightOff* zhasne světlo. Pomocí vstupů *levelUp* a *levelDw* lze nastavovat úroveň, na kterou se světlo bude rozsvěcet. Při nastavování úrovně je rychlost rozsvícení resp. zhasínání určena vstupem *setRamp*, který opět udává čas, za který se světlo rozsvítí z *minLevel* na 100%. Vstup *lightToggle* přepne světlo – pokud svítí tak ho zhasne a pokud je zhasnuté tak ho rozsvítí. Vstup *webToggle* funguje stejně jako *lightToggle* a slouží k zapínání a vypínání světla z web rozhraní (tuto proměnnou je třeba nastavit na TRUE v případě, že chceme přepnout světlo z web stránky). Vstup *timeLimit* umožňuje omezit dobu svícení. Pokud má hodnotu T#0s tak doba svícení není omezena. A konečně vstup *lightReset* slouží jako centrální vypnutí všech světel.

Vstup *name* slouží pro pojmenování světla pro uživatelskou aplikaci.

Proměnná *setLevel* slouží k zapamatování poslední nastavené úrovně a měla by být založena v sekci VAR_GLOBAL RETAIN.

Výstup *out* signalizuje, že je světlo zapnuté. Výstup *level* udává úroveň, na jakou se světlo rozsvítí (0...100%). Výstup *fbPtr* je pointer na funkční blok.

Světlo lze ovládat dvoutlačítkově, pomocí vstupů *lightOn* a *lightOff*, nebo jednotlačítkově pomocí vstupu *lightToggle*.

Popis proměnných :

	Proměnná	Typ	Význam
VAR_INPUT			
	lightOn	BOOL R_EDGE	Náběžná hrana zapne světlo na úroveň *setLevel* Rychlost rozsvícení určuje vstup *ramp*
	levelUp	BOOL	Zvýšení úrovně, na kterou se světlo bude rozsvěcet (mění proměnnou *setLevel*) Rychlost změny určuje vstup *setRamp*
	lightOff	BOOL R_EDGE	Náběžná hrana vypne světlo Rychlost zhasnutí určuje vstup *ramp*
	levelDw	BOOL	Snížení úrovně, na kterou se světlo bude rozsvěcet (mění proměnnou *setLevel*) Rychlost změny určuje vstup *setRamp*
	lightToggle	BOOL R_EDGE	Náběžná hrana přepne světlo Rychlost rozsvícení/zhasnutí určuje vstup *ramp*
	minLevel	REAL	Minimální úroveň, ze které se zahájí rozsvícení světla [%]Přednastavená hodnota 10%
	offOnRamp	TIME	Doba, za kterou se světlo rozsvítí z minLevel na 100% pro zapnutí a vypnutí světla [sec] Přednastavená hodnota 5 sec
	setRamp	TIME	Doba, za kterou se světlo rozsvítí z minLevel na 100% pro změnu požadované úrovně [sec] Přednastavená hodnota 8 sec
	timeLimit	TIME	Omezení doby svícení Při T#0s není doba omezena
	webToggle	BOOL R_EDGE	ovládání světla z web stránky
	lightReset	BOOL R_EDGE	Vstup pro centrální vypnutí světel (odchodové tlačítko)
	name	STRING[48]	Pojmenování bloku pro uživatelskou aplikaci
VAR_OUTPUT			
	out	BOOL	TRUE signalizuje zapnuté světlo
	level	REAL	Aktuální hodnota pro řízení světla [0...100%]
	minLev	REAL	minimální úroveň
	fbPtr	UDINT	Pointer na funkční blok
VAR_IN_OUT			
	setLevel	REAL	Požadovaná úroveň, na kterou se má světlo rozsvítit [0...100%]

Předpokládejme, že potřebujeme řídit stmívané světlo převodníkem 0...10V připojeným na analogový výstup AO1. Světlo budeme ovládat nástěnným ovladačem C-WS-0200R-Logus, který je vybaven dvěma tlačítky. V jazyce ST bude program vypadat následovně:

```
VAR_GLOBAL RETAIN
  dimmer2level : REAL;
END_VAR

VAR_GLOBAL
  central_OFF : BOOL;
END_VAR

PROGRAM prgMain
  VAR
    GoOut       : fb_iAction;
    Dimmer2     : fb_iDimmer;
  END_VAR

  // akce centralniho zhasnuti
  GoOut( action := r0_p3_DI.DI0, out => central_OFF);

  // stmivac ovladany CIB modulem C-WS-0200R-Logus
  Dimmer2( lightOn    := MI_CIB1_IN.ID1_IN.DI.CLICK_UP1,
           levelUp    := MI_CIB1_IN.ID1_IN.DI.PRESS_UP1,
           lightOff   := MI_CIB1_IN.ID1_IN.DI.DOWN1,
           levelDw    := MI_CIB1_IN.ID1_IN.DI.PRESS_DOWN1,
           minLevel   := 20.0,
           offOnramp  := T#6s,
           setRamp    := T#10s,
           timeLimit  := T#12h,
           lightReset := central_OFF,
           setLevel   := dimmer2level,
           level      => r1_p0_AO1.ENG);
END_PROGRAM
```

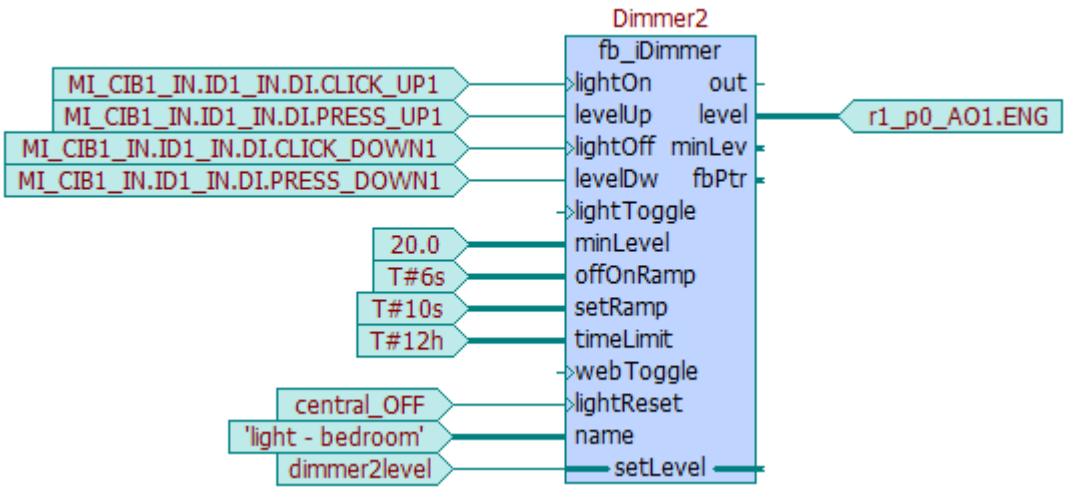
Světlo je zapínáno a vypínáno krátkými stisky tlačítek modulu C-WS-0200R-Logus. Dlouhé stisky těchto tlačítek nastavují požadovanou úroveň. Minimální úroveň pro rozsvícení světla je 20% a světlo se z této úrovně rozsvítí na 100% za 6 sec. Maximální doba, po kterou bude světlo nepřetržitě svítit je omezena na 12 hodin. Stejnou funkci lze naprogramovat v jazyce CFC například následovně:

Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iDimmer* se do souboru „iFox-trot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

Každá instance *fb_iDimmer* přidá následující public proměnné:

	Proměnná	**Typ**	**Význam**
R	*GTSAP1_LIGHT_name*	STRING[48]	Kopie vstupu *name*



	Proměnná	Typ	Význam
RW	*GTSAP1_LIGHT_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_ACTION_needAck*	BOOL	Nastavením na TRUE bude pro změnu stavu bloku vyžadováno potvrzení. Změnit volbu lze pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
R	*GTSAP1_LIGHT_type*	BOOL	Typ světla (1 = stmívané)
R	*GTSAP1_LIGHT_dimtype*	BOOL	Typ stmívače (0 = jedna barva)
RW	*GTSAP1_LIGHT_onoff*	BOOL	Vypínání a zapínání světla z uživatelské aplikace.
RW	*GTSAP1_LIGHT_dimlevel*	REAL	Úroveň stmívače (pouze při *dimtype* = 1)
RW	*GTSAP1_LIGHT_tgtlevel*	REAL	Cílová úroveň stmívače (alternativní řízení úrovně)

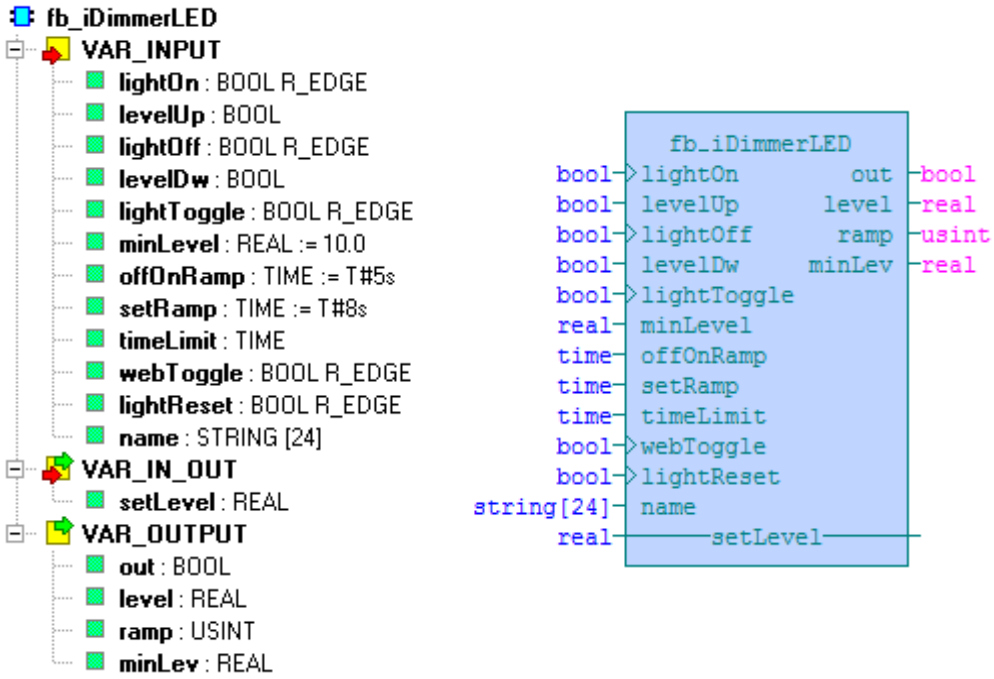
R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.5 Funkční blok fb_iDimmerLED

Knihovna : *iControlLib*

Funkční blok *fb_iDimmerLED* je určen k řízení stmívaného LED světla v rozsahu 0...100%.

Vstup *lightOn* slouží k rozsvícení světla na úroveň nastavenou parametrem *setLevel.* Světlo se rozsvítí plynule, rychlost rozsvícení určuje vstup *offOnRamp.* Ten určuje čas potřebný k rozsvícení z *minLevel* (zapalovací úroveň) na úroveň určenou pomo-



cí *setLevel.* Vstup *lightOff* se používá stejně ale k vypínání světla. Pomocí vstupů *levelUp* a *levelDw* lze nastavit aktuální hodnotu osvětlení. Tato hodnota se zároveň stává úrovní na na kterou se světlo bude rozsvěcet. Vstup *lightToggle* přepne světlo – pokud svítí tak ho zhasne a pokud je zhasnuté tak ho rozsvítí. Vstup *minLevel* určuje minimální úroveň, při které světlo ještě svítí. Proměnná *SetRamp* představuje čas nastavení úrovně z 0 na 100% a určuje rychlost plynulé regulace okamžité úrovně svitu. Vstup *timeLimit* umožňuje omezit dobu svícení. Vstup *webToggle* funguje stejně jako *lightToggle* a slouží k zapínání a vypínání světla z web rozhraní (tuto proměnnou je třeba nastavit na TRUE v případě, že chceme přepnout světlo z web stránky). Proměnná *lightReset* slouží k centrálnímu vypnutí osvětlení. Vstupní řetězec *name* slouží k pojmenování bloku pro uživatelskou aplikaci.

Proměnná *setLevel* slouží k zapamatování poslední nastavené úrovně a měla by být založena v sekci VAR_GLOBAL RETAIN.

Výstupy *level* (0..100%) a *ramp* jsou určeny pro jeden kanál CIB modulu typu ULED/ILED/RLC. Výstup *out* signalizuje, že je světlo zapnuté.

Světlo lze ovládat jak dvoutlačítkově pomocí vstupů *lightOn* a *lightOff*, tak jednotlačítkově pomocí vstupů *lightToggle* resp. *webToggle*.

Popis proměnných :

	Proměnná	***Typ***	***Význam***
VAR_INPUT			
	lightOn	BOOL R_EDGE	Náběžná hrana zapne světlo na úroveň *setLevel* Rychlost rozsvícení určuje vstup *cffOnRamp*
	levelUp	BOOL	Zvýšení úrovně, na kterou se světlo bude rozsvěcet (mění proměnnou *setLevel*) Rychlost změny určuje vstup *setRamp*
	lightOff	BOOL R_EDGE	Náběžná hrana vypne světlo Rychlost zhasnutí určuje vstup *cffOnRamp*

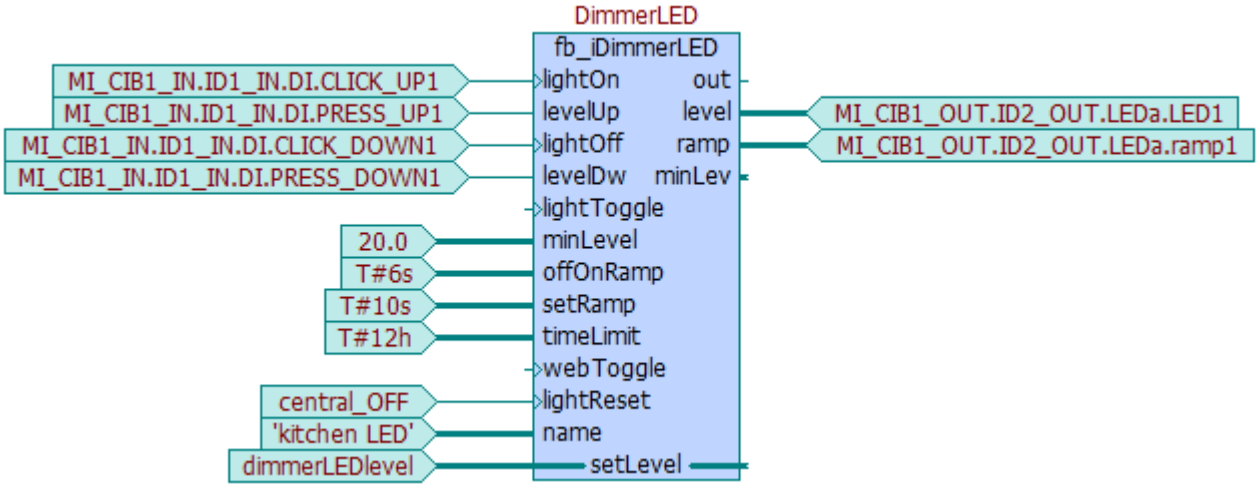
	levelDw	BOOL	Snížení úrovně, na kterou se světlo bude rozsvěcet (mění proměnnou *setLevel*) Rychlost změny určuje vstup *setRamp*
	lightToggle	BOOL R_EDGE	Náběžná hrana přepne světlo. Rychlost rozsvícení/ zhasnutí určuje vstup *cffOnRamp*
	minLevel	REAL	Minimální úroveň, ze které se zahájí rozsvícení světla [%] Přednastavená hodnota 10%
	cffOnRamp	TIME	Doba, za kterou se světlo rozsvítí z *minLevel* na 100% pro zapnutí a vypnutí světla [sec] Přednastavená hodnota 5 sec
	setRamp	TIME	Doba, za kterou se světlo rozsvítí z *minLevel* na 100% pro změnu požadované úrovně [sec] Přednastavená hodnota 8 sec
	timeLimit	TIME	Omezení doby svícení
	webToggle	BOOL R_EDGE	Omezení doby svícení Při T#0s není doba omezena
	lightReset	BOOL R_EDGE	Vstup pro centrální vypnutí světel (odchodové tlačítko)
	name	STRING[48]	Pojmenování bloku pro uživatelskou aplikaci
VAR_OUTPUT			
	out	BOOL	TRUE signalizuje zapnuté světlo
	level	REAL	Aktuální hodnota pro řízení světla [0...100%]
	Ramp	USINT	Aktuální pracovní rampa pro řízení světla [sec]
	minLev	REAL	minimální úroveň 0..90 [%]
VAR_IN_OUT			
	setLevel	REAL	Požadovaná úroveň, na kterou se má světlo rozsvítit [0...100%] umístit do VAR_GLOBAL RETAIN.

Předpokládejme, že potřebujeme řídit stmívané LED světlo připojené na výstup CIB modulu C-DM-0006M-ULED. Světlo budeme ovládat nástěnným ovladačem C-WS-0200R-Logus se dvěma tlačítky. V jazyce ST bude program vypadat následovně:

```
VAR_GLOBAL RETAIN
  dimmerLEDlevel : REAL;
END_VAR

VAR_GLOBAL
  central_OFF  : BOOL;
END_VAR
```

```
PROGRAM prgMain
  VAR
    DimmerLED   : fb_iDimmerLED;
    GoOut       : fb_iAction;
  END_VAR

  // akce centralniho zhasnuti
  GoOut( action := r0_p3_DI.DI0, out => central_OFF);

  // stmivac ovladany CIB modulem C-WS-0200R-Logus
  DimmerLED( lightOn     := MI_CIB1_IN.ID1_IN.DI.CLICK_UP1,
             levelUp     := MI_CIB1_IN.ID1_IN.DI.PRESS_UP1,
             lightOff    := MI_CIB1_IN.ID1_IN.DI.CLICK_DOWN1,
             levelDw     := MI_CIB1_IN.ID1_IN.DI.PRESS_DOWN1,
             minLevel    := 20.0,
             offOnRamp   := T#6s,
             setRamp     := T#10s,
             timeLimit   := T#12h,
             lightReset  := central_OFF,
             name        :='kitchen LED',
             setLevel    := dimmerLEDlevel,
             level       => MI_CIB1_OUT.ID2_OUT.LEDa.LED1,
             ramp        => MI_CIB1_OUT.ID2_OUT.LEDa.ramp1);
END_PROGRAM
```

Světlo je zapínáno a vypínáno krátkými stisky tlačítek modulu C-WS-0200R-Logus. Dlouhé stisky těchto tlačítek nastavují požadovanou úroveň. Minimální úroveň pro rozsvícení světla je 20% a světlo se z této úrovně rozsvítí na 100% za 6 sec. Maximální doba, po kterou bude světlo nepřetržitě svítit je omezena na 12 hodin. Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iDimmerLED* se do souboru „iFox-trot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

Každá instance *fb_iDimmerLED* přidá následující public proměnné:

	Proměnná	Typ	Význam
R	*GTSAP1_LIGHT_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_LIGHT_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_LIGHT_needAck*	BOOL	Nastavením na TRUE bude pro změnu stavu bloku vyžadováno potvrzení. Změnit volbu lze pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
R	*GTSAP1_LIGHT_type*	BOOL	Typ světla (1 = stmívané)
R	*GTSAP1_LIGHT_dimtype*	BOOL	Typ stmívače (0 = jedna barva)
RW	*GTSAP1_LIGHT_onoff*	BOOL	Vypínání a zapínání světla z uživatelské aplikace.
RW	*GTSAP1_LIGHT_dimlevel*	REAL	Úroveň stmívače (pouze při *dimtype* = 1)

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

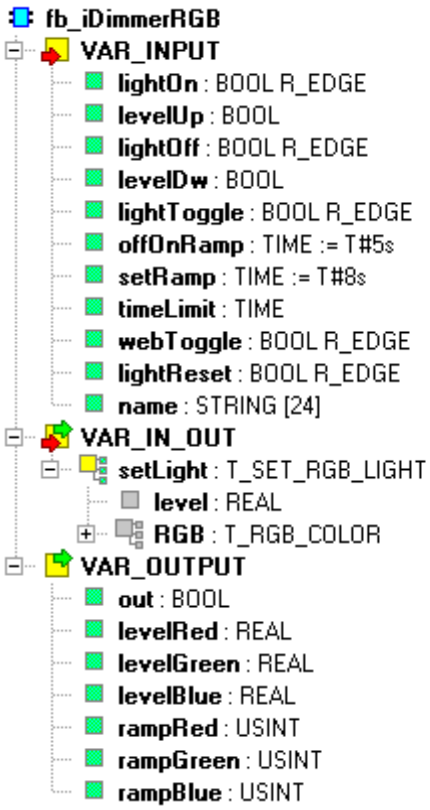
6.6 Funkční blok fb_iDimmerRGB

Knihovna : *iControlLib*

Funkční blok *fb_iDimmerRGB* je určen k řízení stmívaného RGB světla v rozsahu 0...100%. Výstup bloku je vhodný pro řízení světla CIB modulem ULED/ILED.

Vstup *lightOn* slouží k rozsvícení světla na úroveň nastavenou parametrem *setLight.level.* Světlo se rozsvítí plynule, rychlost rozsvícení určuje vstup *offOnRamp.* Ten určuje čas potřebný k rozsvícení světla na danou úroveň. Vstup *lightOff* se používá stejně ale k vypínání světla. Pomocí vstupů *levelUp* a *levelDw* lze nastavit aktuální hodnotu osvětlení. Tato hodnota se zároveň stává úrovní na na kterou se světlo bude rozsvěcet. Vstup *lightToggle* přepne světlo – pokud svítí tak ho zhasne a pokud je zhasnuté tak ho rozsvítí. Vstup *minLevel* určuje minimální úroveň, při které světlo ještě svítí.Vstup *setRamp* představuje čas nastavení úrovně z 0 na 100%, určuje rychlost plynulé regulace okamžité úrovně svitu. Vstup *timeLimit* umožňuje omezit dobu svícení. Vstup *webToggle* funguje stejně jako *lightToggle* a slouží k zapínání a vypínání světla z web rozhraní (tuto proměnnou je třeba nastavit na TRUE v případě, že chceme přepnout světlo z web stránky). Vstup *lightReset* slouží k centrálnímu vypnutí osvětlení. Vstupní řetězec *name* slouží k pojmenování bloku pro uživatelskou aplikaci.

Proměnná *setLight* slouží k zapamatování poslední nastavené úrovně a barvy světla a měla by být založena v sekci VAR_GLOBAL RETAIN.



Výstupy *levelRed(Geen,Blue)* 0..100% a *rampRed(Geen,Blue)* – slouží pro připojení k CIB modulu ULED/ILED. Výstup *out* signalizuje, že je světlo zapnuté. Světlo lze ovládat jak dvoutlačítkově pomocí vstupů *lightOn* a *lightOff*, tak jednotlačítkově pomocí vstupu *lightToggle*.

Popis proměnných :

	Proměnná	Typ	Význam
VAR_INPUT			
	lightOn	BOOL R_EDGE	Náběžná hrana zapne světlo na úroveň *setLevel* Rychlost rozsvícení určuje vstup *cffOnRamp*
	levelUp	BOOL	Zvýšení úrovně, na kterou se světlo bude rozsvěcet (mění proměnnou *setLight.level*) Rychlost změny určuje vstup *setRamp*
	lightOff	BOOL R_EDGE	Náběžná hrana vypne světlo Rychlost zhasnutí určuje vstup *cffOnRamp*
	levelDw	BOOL	Snížení úrovně, na kterou se světlo bude rozsvěcet (mění proměnnou *setLight.level*) Rychlost změny určuje vstup *setRamp*
	lightToggle	BOOL R_EDGE	Náběžná hrana přepne světlo. Rychlost rozsvícení/ zhasnutí určuje vstup *cffOnRamp*
	offOnRamp	TIME	Doba, za kterou se světlo rozsvítí z 0 na 100% pro zapnutí a vypnutí světla [sec] Přednastavená hodnota 5 sec

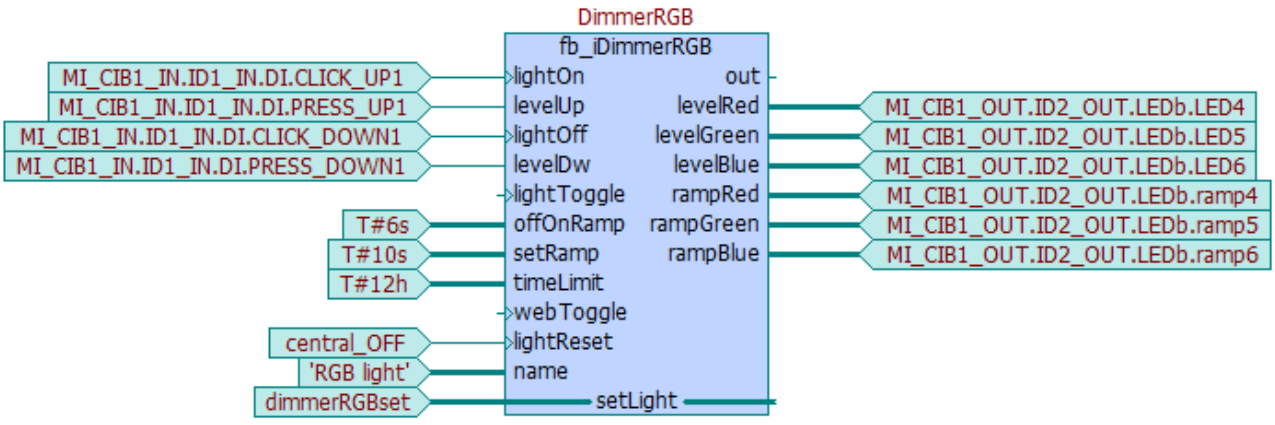
	setRamp	TIME	Doba, za kterou se světlo rozsvítí z *minLevel* na 100% pro změnu požadované úrovně [sec] Přednastavená hodnota 8 sec
	timeLimit	TIME	Omezení doby svícení
	webToggle	BOOL R_EDGE	Přepínání světla z webu
	lightReset	BOOL R_EDGE	Vstup pro centrální vypnutí světel (odchodové tlačítko)
	name	STRING[48]	Pojmenování bloku pro uživatelskou aplikaci
VAR_OUTPUT			
	out	BOOL	TRUE signalizuje zapnuté světlo
	levelRed	REAL	Aktuální hodnota pro řízení červené barvy [0...100%]
	levelGreen	REAL	Aktuální hodnota pro řízení zelené barvy [0...100%]
	levelBlue	REAL	Aktuální hodnota pro řízení modré barvy [0...100%]
	rampRed	USINT	Aktuální pracovní rampa pro řízení červené barvy
	rampGreen	USINT	Aktuální pracovní rampa pro řízení zelené barvy
	rampBlue	USINT	Aktuální pracovní rampa pro řízení modré barvy
VAR_IN_OUT			
	SetLight *.level* *.RGB*	T_SET_RGB_LIGHT	Parametry pro rozsvícení světla (umístit do VAR_GLOBAL RETAIN) Úroveň, na kterou se má světlo rozsvítit [0...100%] Struktura pro zadání barvy světla (*red, green, blue*)

Předpokládejme, že potřebujeme řídit stmívané RGB světlo CIB modulem C-DM-0006M-ULED. Světlo budeme ovládat nástěnným ovladačem C-WS-0200R-Logus, který je vybaven dvěma tlačítky. V jazyce ST bude program vypadat následovně:

```
VAR_GLOBAL RETAIN
  dimmerRGBset : T_SET_RGB_LIGHT;
END_VAR
VAR_GLOBAL
  central_OFF  : BOOL;
END_VAR

PROGRAM prgMain
  VAR
    DimmerRGB     : fb_iDimmerRGB;
```

```
    GoOut          : fb_iAction;
  END_VAR

  // nastaveni barvy R=255 G=255 B=255 bila; R=255,G=30, B=80 magenta; ...
  dimmerRGBset.RGB.red   := 255;
  dimmerRGBset.RGB.green := 255;
  dimmerRGBset.RGB.blue  := 255;

  // akce centralniho zhasnuti
  GoOut( action := r0_p3_DI.DI0, out => central_OFF);

  // stmivac ovladany CIB modulem C-WS-0200R-Logus
  DimmerRGB(lightOn     := MI_CIB1_IN.ID1_IN.DI.CLICK_UP1,
            levelUp     := MI_CIB1_IN.ID1_IN.DI.PRESS_UP1,
            lightOff    := MI_CIB1_IN.ID1_IN.DI.CLICK_DOWN1,
            levelDw     := MI_CIB1_IN.ID1_IN.DI.PRESS_DOWN1,
            offOnRamp   := T#6s,
            setRamp     := T#10s,
            timeLimit   := T#12h,
            lightReset  := central_OFF,
            name        := 'RGB light',
            setLight    := dimmerRGBset,
            levelRed    => MI_CIB1_OUT.ID2_OUT.LEDb.LED4,
            levelGreen  => MI_CIB1_OUT.ID2_OUT.LEDb.LED5,
            levelBlue   => MI_CIB1_OUT.ID2_OUT.LEDb.LED6,
            rampRed     => MI_CIB1_OUT.ID2_OUT.LEDb.ramp4,
            rampGreen   => MI_CIB1_OUT.ID2_OUT.LEDb.ramp5,
            rampBlue    => MI_CIB1_OUT.ID2_OUT.LEDb.ramp6);
END_PROGRAM
```

Světlo je zapínáno a vypínáno krátkými stisky tlačítek modulu C-WS-0200R-Logus. Dlouhé stisky těchto tlačítek nastavují požadovanou úroveň jasu. Z úrovně 0% se rozsvítí na 100% za 6 sec. Maximální doba, po kterou bude světlo nepřetržitě svítit je omezena na 12 hodin. Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iDimmerRGB* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

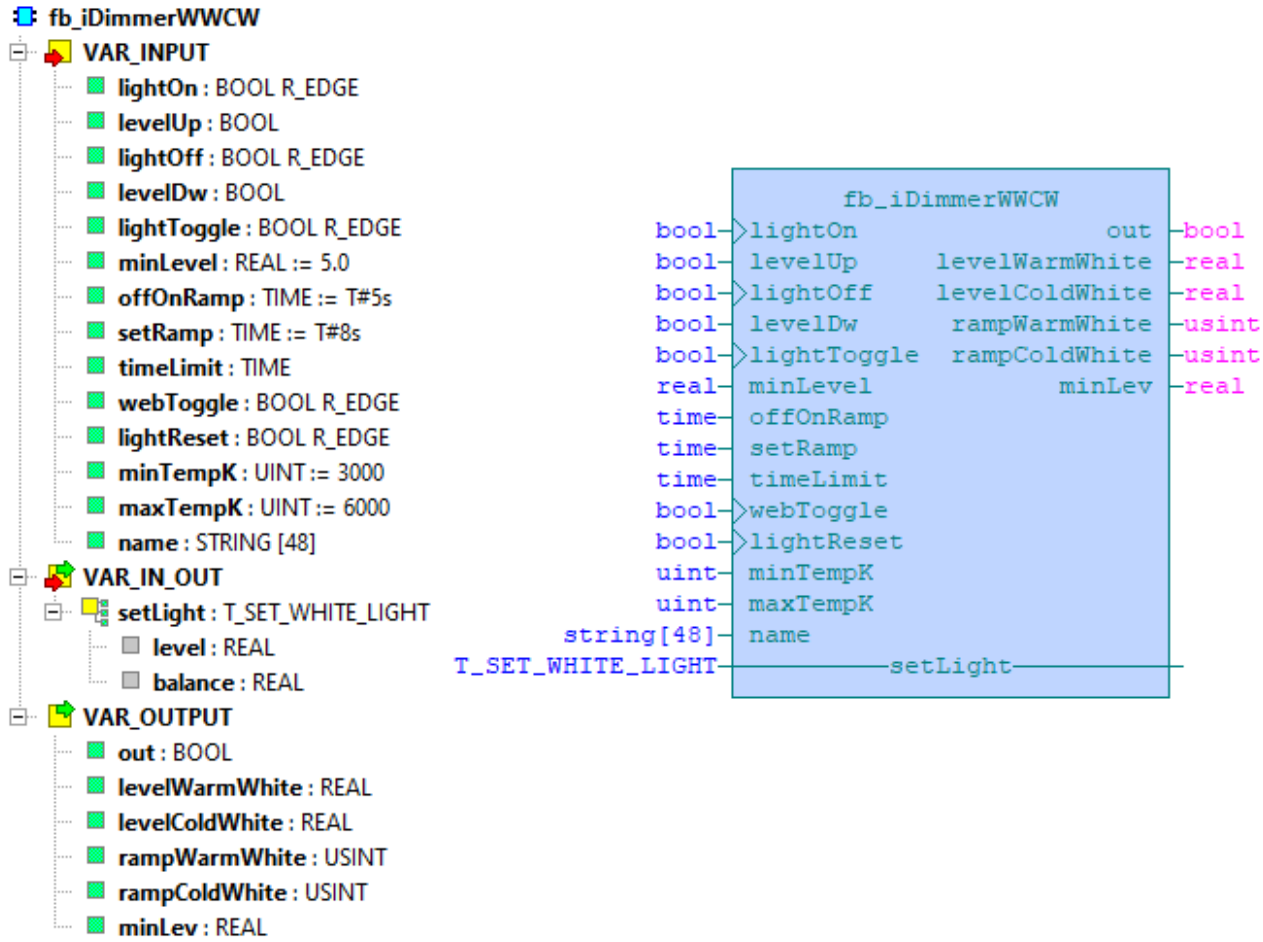
Každá instance *fb_iDimmerRGB* přidá následující public proměnné:

	Proměnná	Typ	Význam
R	*GTSAP1_LIGHT_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_LIGHT_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_ACTION_needAck*	BOOL	Nastavením na TRUE bude pro změnu stavu bloku vyžadováno potvrzení. Změnit volbu lze pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
R	*GTSAP1_LIGHT_type*	BOOL	Typ světla (1 = stmívané)
R	*GTSAP1_LIGHT_dimtype*	USINT	Typ stmívače (1 = RGB)
RW	*GTSAP1_LIGHT_onoff*	BOOL	Vypínání a zapínání světla z uživatelské aplikace
RW	*GTSAP1_LIGHT_dimlevel*	REAL	Úroveň stmívače [0...100%]
RW	*GTSAP1_LIGHT_rgb*	UDINT	Barva světla
RW	*GTSAP1_LIGHT_colortemp*	UDINT	Teplota světla [Kelvin]

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.7 Funkční blok fb_iDimmerWWCW

Knihovna : *iControlLib*



Funkční blok *fb_iDimmerWWCW* je určen k řízení stmívaného dvoukanálového bílého světla (teplá + studená bílá) v rozsahu 0...100%. Výstup bloku je vhodný pro řízení světla CIB modulem ULED/ILED (např. C-DM-0006M-ULED).

Vstup *lightOn* slouží k rozsvícení světla na úroveň nastavenou parametrem *setLight.level*, přičemž poměr teplé a studené bílé udává parametr *setLight.balance*. Světlo se rozsvítí plynule, rychlost rozsvícení určuje vstup *offOnRamp.* Ten určuje čas potřebný k rozsvícení světla na danou úroveň. Vstup *lightOff* se používá stejně ale k vypínání světla. Pomocí vstupů *levelUp* a *levelDw* lze nastavit aktuální hodnotu osvětlení. Tato hodnota se zároveň stává úrovní na na kterou se světlo bude rozsvěcet. Vstup *lightToggle* přepne světlo – pokud svítí tak ho zhasne a pokud je zhasnuté tak ho rozsvítí. Vstup *minLevel* určuje minimální úroveň, při které světlo ještě svítí.Vstup *setRamp* představuje čas nastavení úrovně z 0 na 100%, určuje rychlost plynulé regulace okamžité úrovně svitu. Vstup *timeLimit* umožňuje omezit dobu svícení. Vstup *webToggle* funguje stejně jako *lightToggle* a slouží k zapínání a vypínání světla z web rozhraní (tuto proměnnou je třeba nastavit na TRUE v případě, že chceme přepnout světlo z web stránky). Vstup *lightReset* slouží k centrálnímu vypnutí osvětlení. Vstupní řetězec *name* slouží k pojmenování bloku pro uživatelskou aplikaci.

Proměnná *setLight* slouží k zapamatování poslední nastavené úrovně jasu a poměru bílé a měla by být založena v sekci VAR_GLOBAL RETAIN nebo VAR_RETAIN.

Výstupy *levelWarmWhite a levelColdWhite* 0..100% a *rampWarmWhite a rampColdWhite* – slouží pro připojení k CIB modulu ULED/ILED. Výstup *out* signalizuje, že je světlo zapnuté. Světlo lze ovládat jak dvoutlačítkově pomocí vstupů *lightOn* a *lightOff*, tak jednotlačítkově pomocí vstupu *lightToggle*.

Popis proměnných :

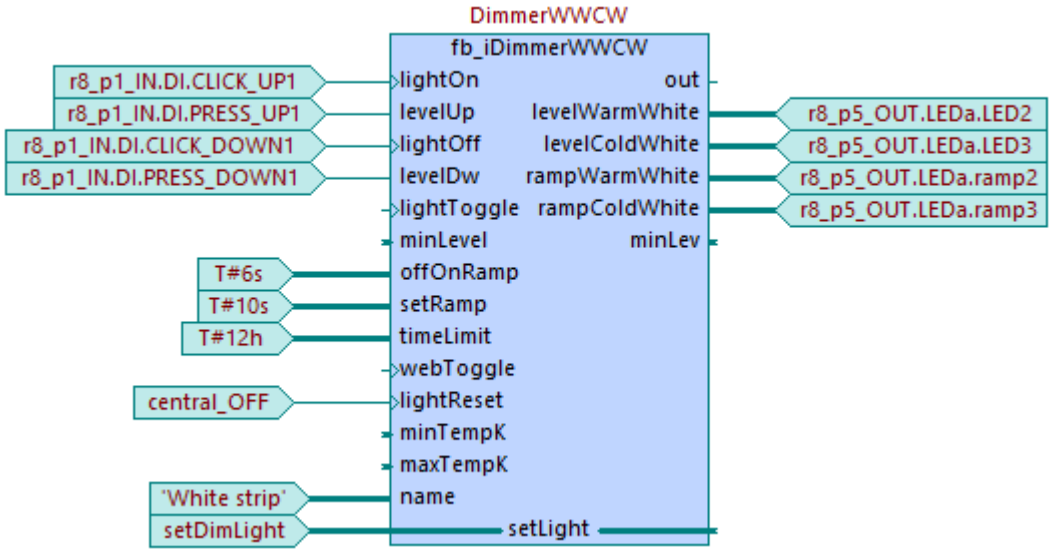
	Proměnná	Typ	Význam
VAR_INPUT			
	lightOn	BOOL R_EDGE	Náběžná hrana zapne světlo na úroveň *setLevel* Rychlost rozsvícení určuje vstup *cffOnRamp*
	levelUp	BOOL	Zvýšení úrovně, na kterou se světlo bude rozsvěcet (mění proměnnou *setLight.level*) Rychlost změny určuje vstup *setRamp*
	lightOff	BOOL R_EDGE	Náběžná hrana vypne světlo Rychlost zhasnutí určuje vstup *cffOnRamp*
	levelDw	BOOL	Snížení úrovně, na kterou se světlo bude rozsvěcet (mění proměnnou *setLight.level*) Rychlost změny určuje vstup *setRamp*
	lightToggle	BOOL R_EDGE	Náběžná hrana přepne světlo. Rychlost rozsvícení/ zhasnutí určuje vstup *cffOnRamp*
	offOnRamp	TIME	Doba, za kterou se světlo rozsvítí z 0 na 100% pro zapnutí a vypnutí světla [sec] Přednastavená hodnota 5 sec
	setRamp	TIME	Doba, za kterou se světlo rozsvítí z *minLevel* na 100% pro změnu požadované úrovně [sec] Přednastavená hodnota 8 sec
	timeLimit	TIME	Omezení doby svícení
	webToggle	BOOL R_EDGE	Přepínání světla z webu
	lightReset	BOOL R_EDGE	Vstup pro centrální vypnutí světel (odchodové tlačít-ko)
	minTempK	UINT	Výchozí hodnota teplé bílé (3000 °K)
	maxTempK	UINT	Výchozí hodnota studené bílé (6000 °K)
	name	STRING[48]	Pojmenování bloku pro uživatelskou aplikaci
VAR_OUTPUT			
	out	BOOL	TRUE signalizuje zapnuté světlo
	levelWarmWhite	REAL	Aktuální hodnota pro řízení teplé bílé [0...100%]
	levelColdWhite	REAL	Aktuální hodnota pro řízení studené bílé [0...100%]
	ramWarmWhite	USINT	Aktuální pracovní rampa pro řízení teplé bílé

	rampColdWhite	USINT	Aktuální pracovní rampa pro řízení studené bílé
VAR_IN_OUT			
	SetLight *.level* *.balance*	T_SET_WHITE_LIGHT	Parametry pro rozsvícení světla Úroveň, na kterou se má světlo rozsvítit [0...100%] Poměr teplé a studené bílé vyjádřený v % 0% … žádná studená bílá, tj. 100% teplá bílá 25% … 25% studená bílá a 75% teplá bílá 50% … 50% studená bílá a 50% teplá bílá 100% … 100% studená bílá

Předpokládejme, že potřebujeme řídit dvoukanálové bílé světlo CIB modulem C-DM-0006M-ULED. Světlo budeme ovládat nástěnným ovladačem C-WS-0200R-Logus, který je vybaven dvěma tlačítky. V jazyce ST bude program vypadat následovně:

```
VAR_GLOBAL RETAIN
  setDimLight : T_SET_WHITE_LIGHT := ( level := 50.0,  balance := 40.0);
END_VAR
VAR_GLOBAL
  central_OFF  : BOOL;
END_VAR

PROGRAM prgMain
  VAR
    DimmerWWCW : fb_iDimmerWWCW;
    GoOut      : fb_iAction;
  END_VAR

  // akce centralniho zhasnuti
  GoOut( action := r0_p5_DI.DI0, out => central_OFF);

  // stmivac ovladany CIB modulem C-WS-0200R-Logus
  DimmerWWCW( lightOn    := r8_p1_IN.DI.CLICK_UP1,
              levelUp    := r8_p1_IN.DI.PRESS_UP1,
              lightOff   := r8_p1_IN.DI.CLICK_DOWN1,
              levelDw    := r8_p1_IN.DI.PRESS_DOWN1,
              offOnRamp  := T#6s,
              setRamp    := T#10s,
              timeLimit  := T#12h,
              lightReset := central_OFF,
              name       := 'White strip',
              setLight   := setDimLight,
              levelWarmWhite => r8_p5_OUT.LEDa.LED2,
              levelColdWhite => r8_p5_OUT.LEDa.LED3,
              rampWarmWhite => r8_p5_OUT.LEDa.ramp2,
              rampColdWhite => r8_p5_OUT.LEDa.ramp3);
END_PROGRAM
```

Světlo je zapínáno a vypínáno krátkými stisky tlačítek modulu C-WS-0200R-Logus. Dlouhé stisky těchto tlačítek nastavují požadovanou úroveň jasu. Z úrovně 0% se rozsvítí na 100% za 6 sec. Maximální doba, po kterou bude světlo nepřetržitě svítit je omezena na 12 hodin. Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iDimmerWWCW* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

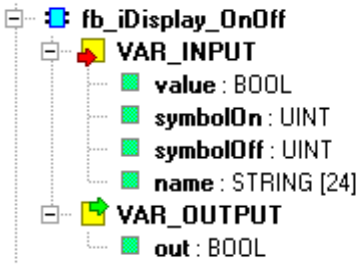
Každá instance *fb_iDimmerWWCW* přidá následující public proměnné:

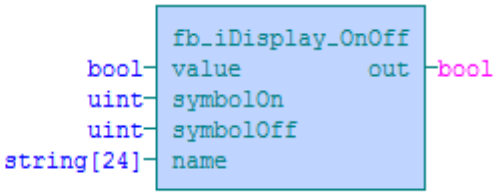
	Proměnná	Typ	Význam
R	*GTSAP1_LIGHT_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_LIGHT_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_LIGHT_needAck*	BOOL	Nastavením na TRUE bude pro změnu stavu bloku vyžadováno potvrzení. Změnit volbu lze pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
R	*GTSAP1_LIGHT_type*	BOOL	Typ světla (1 = stmívané)
R	*GTSAP1_LIGHT_dimtype*	USINT	Typ stmívače (2 = dual white)
R	*GTSAP1_LIGHT_minTempK*	UINT	Výchozí hodnota teplé bílé
R	*GTSAP1_LIGHT_maxTempK*	UINT	Výchozí hodnota studené bílé
RW	*GTSAP1_LIGHT_onoff*	BOOL	Vypínání a zapínání světla z uživatelské aplikace.
RW	*GTSAP1_LIGHT_dimlevel*	REAL	Úroveň stmívače [0...100%]
RW	*GTSAP1_LIGHT_balance*	REAL	Poměr teplé a studené bílé
RW	*GTSAP1_LIGHT_colortemp*	UDINT	Teplota světla [Kelvin]

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.8 Funkční blok fb_iDisplay_OnOff

Knihovna : *iControlLib*





Funkční blok *fb_iDisplay_OnOff* je určen k zobrazení BOOL hodnoty v uživatelské aplikaci. Zobrazovaná hodnota je daná vstupem *value*. Pokud má vstup *value* hodnotu TRUE, pak bude zobrazena ikona daná vstupem *symbolOn*. Pokud má vstup *value* hodnotu FALSE, pak bude zobrazena ikona daná vstupem *symbolOff*. Vstupní řetězec *name* slouží k pojmenování bloku pro uživatelskou aplikaci.

Výstup *out* kopíruje hodnotu *value*.

Popis proměnných :

	Proměnná	Typ	Význam
VAR_INPUT			
	value	BOOL	Zobrazovaná hodnota
	symbolOn	UINT	Ikona, která bude zobrazena při *value* = *TRUE*
	symbolOff	UINT	Ikona, která bude zobrazena při *value* = *FALSE*
	name	STRING[48]	Pojmenování bloku pro uživatelskou aplikaci
VAR_OUTPUT			
	out	BOOL	Kopíruje hodnotu na vstupu *value*

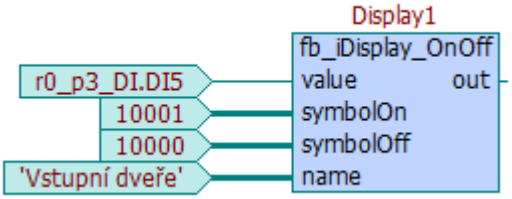
Definice ikon pro uživatelskou aplikaci

Uživatelsky definované ikony začínají hodnotou 10000. Pokud je například *symbolOn := 10000* tak aplikace při *value* = *TRUE* zobrazí ikonu danou obrázkem s názvem *10000.png*, který musí být uložen v adresáři WWW/iFOX/ na SD kartě v PLC systému. Pokud soubor *10000.png* uložíme v projektu Mosaicu do složky SendRoot\WWW\iFOX\ pak se soubor automaticky uloží na SD kartu v PLC při prvním odeslání přeloženého programu do PLC.

Předpokládejme, že chceme v uživatelské aplikaci signalizovat stav vstupních dveří (otevřené / zavřené). Dveřní kontakt bude připojen na digitální vstup PLC systému. V jazyce ST bude program vypadat následovně:

```
PROGRAM prgMain
  VAR
    Display1 : fb_iDisplay_OnOff;
  END_VAR
  VAR CONSTANT
    APP_ICON_DOOR_OPEN  : UINT := 10000;  // tato ikona se nacte z PLC
                                          // WWW/iFOX/10000.png
    APP_ICON_DOOR_CLOSE : UINT := 10001;  // tato ikona se nacte z PLC
                                          // WWW/iFOX/10001.png
  END_VAR

  Display1( value     := r0_p3_DI.DI5,
            symbolOn  := APP_ICON_DOOR_CLOSE,
            symbolOff := APP_ICON_DOOR_OPEN,
            name      := 'Vstupní dveře');
END_PROGRAM
```

Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iDisplay_OnOff* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

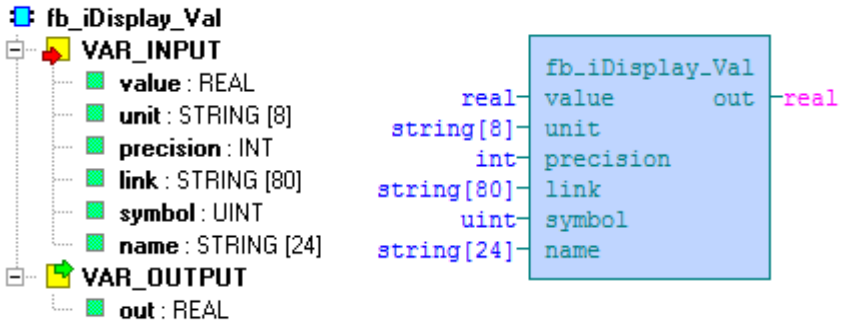
Každá instance *fb_iDisplay_OnOff* přidá následující public proměnné:

	Proměnná	Typ	Význam
R	*GTSAP1_DISPLAY_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_DISPLAY_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
R	*GTSAP1_DISPLAY_edit*	BOOL	Povolení editace (0 = editace zakázaná)
R	*GTSAP1_DISPLAY_type*	USINT	Typ zobrazení (0 = BOOL)
R	*GTSAP1_DISPLAY_symbol*	UINT	Aktuální kód symbolu
R	*GTSAP1_DISPLAY_onOff*	BOOL	Zobrazovaná BOOL hodnota

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.9 *Funkční blok fb_iDisplay_Val*

Knihovna : *iControlLib*



Funkční blok *fb_iDisplay_Val* je určen k zobrazení REAL hodnoty v uživatelské aplikaci. Zobrazovaná hodnota je daná vstupem *value*. Vstup *precision* udává kolik desetinných míst bude zobrazeno. Vstup *unit* umožňuje specifikovat fyzikální jednotku (např. °C, mA, atd.). Vstupem *link* lze zadat vazbu na konkrétní web stránku v PLC. Prvek displeje v uživatelské aplikaci pak nabídne možnost zobrazení této web stránky v aplikaci.

Vstup *symbol* udává ikonu, která bude zobrazena v aplikaci u zobrazené hodnoty. Pokud má hodnotu 0, pak bude zobrazeno výchozí ikona pro displej. Vstupní řetězec *name* slouží k pojmenování bloku pro uživatelskou aplikaci.

Výstup *out* kopíruje hodnotu *value*.

Popis proměnných :

	Proměnná	***Typ***	***Význam***
VAR_INPUT			
	value	REAL	Zobrazená hodnota
	unit	STRING[8]	Fyzikální jednotka zobrazené hodnoty
	precision	INT	Počet zobrazených desetinných míst
	link	STRING	Odkaz na web stránku v PLC s doplňujícími informacemi (nepovinný)
	symbol	UINT	Kód ikony zobrazené v uživatelské aplikaci
	name	STRING[48]	Pojmenování bloku pro uživatelskou aplikaci
VAR_OUTPUT			
	out	REAL	Kopíruje hodnotu na vstupu *value*

Definice ikon pro uživatelskou aplikaci

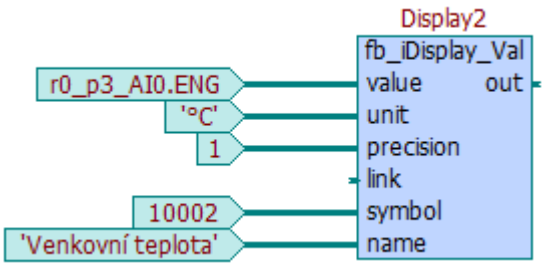
Uživatelsky definované ikony začínají hodnotou 10000. Pokud je například *symbolOn := 10000* tak aplikace při *value = TRUE* zobrazí ikonu danou obrázkem s názvem *10000.png*, který musí být uložen v adresáři WWW/iFOX/. Pokud soubor *10000.png* uložíme v projektu Mosaicu do složky SendRoot\WWW\iFOX\ pak se soubor automaticky uloží na SD kartu v PLC při prvním odeslání přeloženého programu do PLC.

Příklad

Předpokládejme, že chceme v uživatelské aplikaci zobrazit venkovní teplotu měřenou vstupem r0_p3_AI0.ENG. Teplota bude zobrazená na jedno desetinné místo, jednotky budou °C. V jazyce ST bude program vypadat následovně:

```
PROGRAM prgMain
  VAR
    Display2 : fb_iDisplay_Val;
  END_VAR
  VAR CONSTANT
    APP_ICON_OUTSIDE_TEMP : UINT := 10002;  // tato ikona se nacte z PLC
                                            // WWW/iFOX/10002.png
  END_VAR

  Display2( value     := r0_p3_AI0.ENG,
            unit      := '°C',
            precision := 1,
            symbol    := APP_ICON_OUTSIDE_TEMP,
            name      := 'Venkovní teplota');
END_PROGRAM
```

Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iDisplay_Val* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

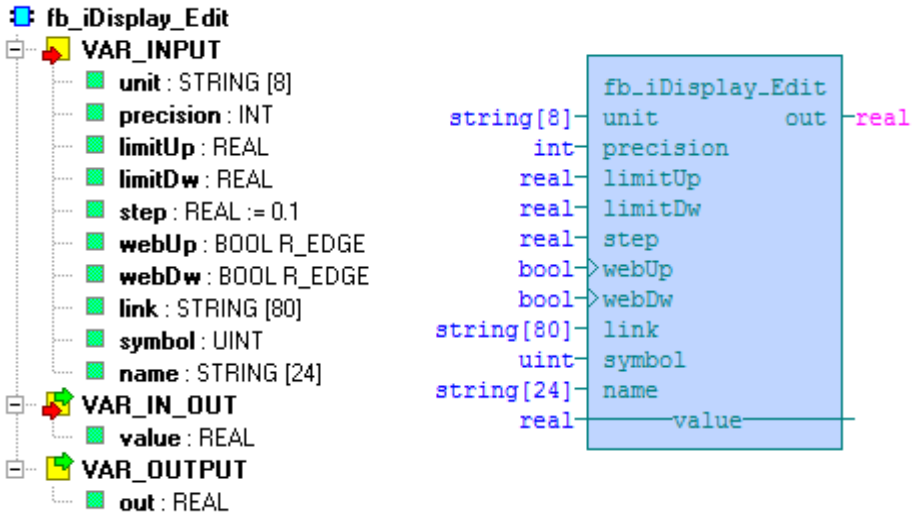
Každá instance *fb_iDisplay_Val* přidá následující public proměnné:

	Proměnná	Typ	Význam
R	*GTSAP1_DISPLAY_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_DISPLAY_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
R	*GTSAP1_DISPLAY_edit*	BOOL	Povolení editace (0 = editace zakázaná)
R	*GTSAP1_DISPLAY_type*	USINT	Typ zobrazení (1 = REAL)
R	*GTSAP1_DISPLAY_symbol*	UINT	Aktuální kód symbolu
R	*GTSAP1_DISPLAY_value*	REAL	Zobrazovaná REAL hodnota
R	*GTSAP1_DISPLAY_unit*	STRING[8]	Fyzikální jednotka zobrazené hodnoty
R	*GTSAP1_DISPLAY_precision*	INT	Počet zobrazených desetinných míst
R	*GTSAP1_DISPLAY_url*	STRING	Odkaz na web stránku

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.10 Funkční blok fb_iDisplay_Edit

Knihovna : *iControlLib*



Funkční blok *fb_iDisplay_Edit* je určen k zobrazení a editaci REAL hodnoty v uživatelské aplikaci. Zobrazovaná hodnota je daná vstupem *value*. Vstup *precision* udává kolik desetinných míst bude zobrazeno. Vstup *unit* umožňuje specifikovat fyzikální jednotku (např. °C, mA, atd.). Vstupy *limitUp* a *limitDw* určují horní a dolní mez, mezi kterými je

možné hodnotu editovat, vstup *step* udává přírůstek o který bude hodnota zvýšena resp. snížena v každém kroku. Vstupy webUp a webDw umožňují editovat zobrazenou hodnotu z web stránky v PLC. Zápis hodnoty TRUE do těchto proměnných zvýší resp. sníží zobrazenou hodnotu o přírůstek daný vstupem *step*.

Vstupem *link* lze zadat vazbu na konkrétní web stránku v PLC. Prvek displeje v uživatelské aplikaci pak nabídne možnost zobrazení této web stránky v aplikaci.

Vstup *symbol* udává ikonu, která bude zobrazena v aplikaci u zobrazené hodnoty. Pokud má hodnotu 0, pak bude zobrazena výchozí ikona pro displej. Vstupní řetězec *name* slouží k pojmenování bloku pro uživatelskou aplikaci.

Výstup *out* kopíruje hodnotu *value*.

Definice ikon pro uživatelskou aplikaci

Uživatelsky definované ikony začínají hodnotou 10000. Pokud je například *symbolOn := 10000* tak aplikace při *value = TRUE* zobrazí ikonu danou obrázkem s názvem *10000.png*, který musí být uložen v adresáři WWW/iFOX/. Pokud soubor *10000.png* uložíme v projektu Mosaicu do složky SendRoot\WWW\iFOX\ pak se soubor automaticky uloží do souborového systému v PLC při prvním odeslání přeloženého programu do PLC.

Popis proměnných :

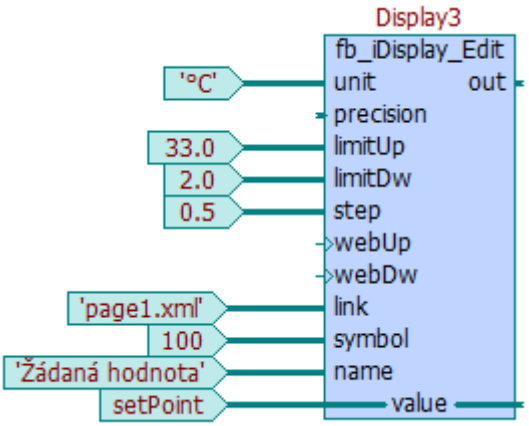
	Proměnná	***Typ***	***Význam***
VAR_INPUT			
	unit	STRING[8]	Fyzikální jednotka zobrazené hodnoty
	precision	INT	Počet zobrazených desetinných míst
	limitUp	REAL	Horní mez pro edici hodnoty
	limitDw	REAL	Spodní mez pro edici hodnoty
	step	REAL	Přírůstek pro edici hodnoty
	webUp	BOOL	Žádost o zvýšení hodnoty o přírůstek (určeno pro ovládání z web stránky)
	webDw	BOOL	Žádost o snížení hodnoty o přírůstek (určeno pro ovládání z web stránky)
	link	STRING	Odkaz na web stránku v PLC s doplňujícími informacemi (nepovinný)
	symbol	UINT	Kód ikony zobrazené v uživatelské aplikaci
	name	STRING[48]	Pojmenování bloku pro uživatelskou aplikaci
VAR_OUTPUT			

	out	REAL	Kopíruje hodnotu na vstupu *value*
VAR_IN_OUT			
	value	REAL	Editovaná hodnota

Předpokládejme, že chceme v uživatelské aplikaci umožnit nastavení žádané hodnoty teploty pro topení. Teplota bude zobrazená na jedno desetinné místo, jednotky budou °C. V jazyce ST bude program vypadat následovně:

```
VAR_GLOBAL RETAIN
  setPoint : REAL := 22.5;
END_VAR

PROGRAM prgMain15
  VAR
    Display3 : fb_iDisplay_Edit;
  END_VAR
  VAR CONSTANT
    APP_ICON_SET_POINT : UINT := 10003;  // tato ikona se nacte z PLC
                                         // WWW/iFOX/10003.png
  END_VAR

  Display3( unit      := '°C',
            precision := 1,
            limitUp   := 33.0,
            limitDw   := 2.0,
            step      := 0.5,
            symbol    := APP_ICON_SET_POINT,
            name      := 'Žádaná hodnota',
            value     := setPoint);
END_PROGRAM
```

Žádanou hodnotu bude možné editovat v mezích <2.0,33.0>. Přírůstek na jeden krok bude 0.5. Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iDisplay_Edit* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

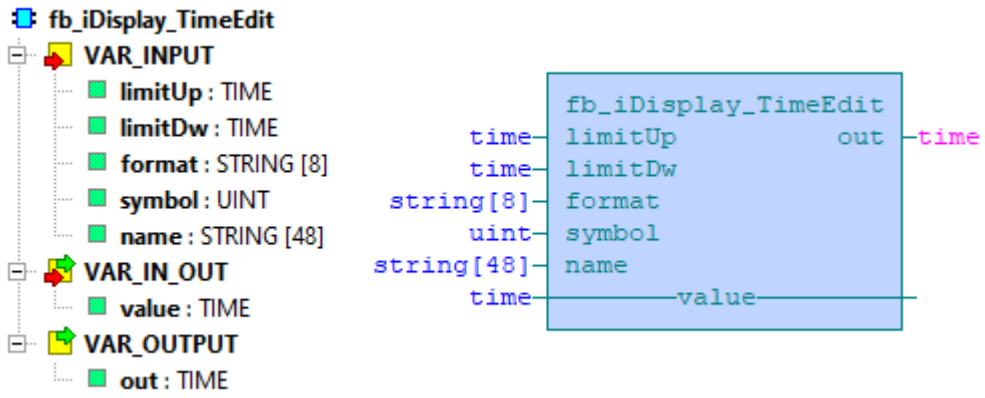
Každá instance *fb_iDisplay_Edit* přidá následující public proměnné:

	Proměnná	Typ	Význam
R	*GTSAP1_DISPLAY_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_DISPLAY_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
R	*GTSAP1_DISPLAY_edit*	BOOL	Povolení editace (0 = editace zakázaná)
R	*GTSAP1_DISPLAY_type*	USINT	Typ zobrazení (1 = REAL)
R	*GTSAP1_DISPLAY_symbol*	UINT	Aktuální kód symbolu
R	*GTSAP1_DISPLAY_value*	REAL	Zobrazovaná REAL hodnota
R	*GTSAP1_DISPLAY_unit*	STRING[8]	Fyzikální jednotka zobrazené hodnoty
R	*GTSAP1_DISPLAY_precision*	INT	Počet zobrazených desetinných míst
R	*GTSAP1_DISPLAY_url*	STRING	Odkaz na web stránku
W	*GTSAP1_DISPLAY_incValue*	BOOL	Zvýšení hodnoty o přírůstek
W	*GTSAP1_DISPLAY_decValue*	BOOL	Snížení hodnoty o přírůstek

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.11 Funkční blok fb_iDisplay_TimeEdit

Knihovna : *iControlLib*



Funkční blok *fb_iDisplay_TimeEdit* je určen k zobrazení a editaci hodnot typu TIME v uživatelské aplikaci. Zobrazovaná hodnota je daná vstupem *value*. Vstupy *limitUp* a *limitDw* určují horní a dolní mez, mezi kterými je možné hodnotu editovat a vstup *format* udává formát zobrazení času (ve tvaru ‘hh:mm:ss‘, ‘hh:mm‘, ‘mm:ss‘, apod).

Vstup *symbol* udává ikonu, která bude zobrazena v aplikaci u zobrazené hodnoty. Pokud má hodnotu 0, pak bude zobrazena výchozí ikona pro displej. Vstupní řetězec name slouží k pojmenování bloku pro uživatelskou aplikaci.

Výstup *out* kopíruje hodnotu *value*.

Popis proměnných :

	Proměnná	Typ	Význam
VAR_INPUT			
	limitUp	TIME	Horní mez pro edici hodnoty
	limitDw	TIME	Spodní mez pro edici hodnoty
	format	STRING[8]	Formát zobrazení času
	symbol	UINT	Kód ikony zobrazené v uživatelské aplikaci
	name	STRING[48]	Pojmenování bloku pro uživatelskou aplikaci
VAR_OUTPUT			
	out	TIME	Kopíruje hodnotu na vstupu *value*
VAR_IN_OUT			
	value	TIME	Editovaná hodnota

Definice ikon pro uživatelskou aplikaci

Uživatelsky definované ikony začínají hodnotou 10000. Pokud je například *symbolOn := 10000* tak aplikace při *value = TRUE* zobrazí ikonu danou obrázkem s názvem *10000.png*, který musí být uložen v adresáři WWW/iFOX/. Pokud soubor *10000.png* uložíme v projektu Mosaicu do složky SendRoot\WWW\iFOX\ pak se soubor automaticky uloží do souborového systému v PLC při prvním odeslání přeloženého programu do PLC.

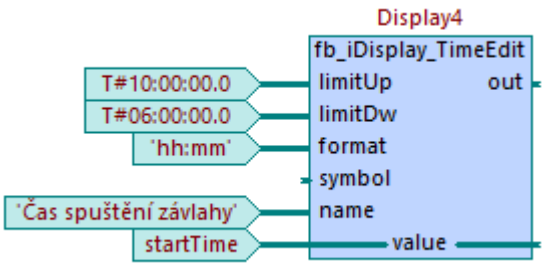
Příklad

Předpokládejme, že chceme v uživatelské aplikaci umožnit nastavení času kdy se má spustit závlahový systém. V jazyce ST bude program vypadat následovně:

```
VAR_GLOBAL RETAIN
  startTime    : TIME := T#08:00:00.0;
END_VAR

PROGRAM prgMain
  VAR
    Display4   : fb_iDisplay_TimeEdit := ( limitUp := T#10:00:00.0,
                                           limitDw := T#06:00:00.0,
                                           format  := 'hh:mm',
                                           name    := 'Čas spuštění závlahy');
  END_VAR
```

```
  Display4( value := startTime);
END_PROGRAM
```

Zadávanou hodnotu času bude možné editovat v mezích <06:00,10:00>. Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iDisplay_TimeEdit* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

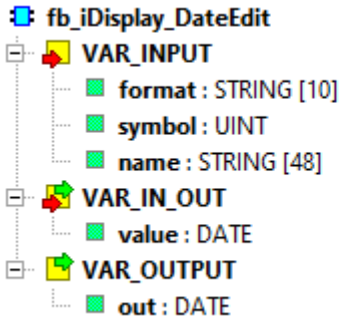
Každá instance *fb_iDisplay_TimeEdit* přidá následující public proměnné:

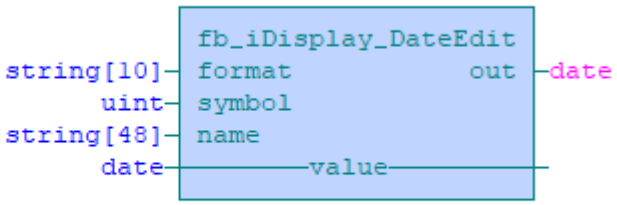
	Proměnná	**Typ**	**Význam**
R	*GTSAP1_DISPLAY_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_DISPLAY_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
R	*GTSAP1_DISPLAY_edit*	BOOL	Povolení editace (0 = editace zakázaná)
R	*GTSAP1_DISPLAY_type*	USINT	Typ zobrazení (2 = TIME)
R	*GTSAP1_DISPLAY_symbol*	UINT	Aktuální kód symbolu
RW	*GTSAP1_DISPLAY_value*	TIME	Editovaná TIME hodnota
R	*GTSAP1_DISPLAY_format*	STRING[8]	Formát zobrazení

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.12 Funkční blok fb_iDisplay_DateEdit

Knihovna : *iControlLib*





Funkční blok *fb_iDisplay_DateEdit* je určen k zobrazení a editaci hodnot typu DATE v uživatelské aplikaci. Zobrazovaná hodnota je daná vstupem *value*. Vstup *format* udává formát zobrazení datumu (ve tvaru ‘yyyy.mm.dd‘, ‘dd.mm.yyyy‘, apod).

Vstup *symbol* udává ikonu, která bude zobrazena v aplikaci u zobrazené hodnoty. Pokud má hodnotu 0, pak bude zobrazena výchozí ikona pro displej. Vstupní řetězec name slouží k pojmenování bloku pro uživatelskou aplikaci.

Výstup *out* kopíruje hodnotu *value*.

Popis proměnných :

	Proměnná	Typ	Význam
VAR_INPUT			
	format	STRING[10]	Formát zobrazení datumu
	symbol	UINT	Kód ikony zobrazené v uživatelské aplikaci
	name	STRING[48]	Pojmenování bloku pro uživatelskou aplikaci
VAR_OUTPUT			
	out	DATE	Kopíruje hodnotu na vstupu *value*
VAR_IN_OUT			
	value	DATE	Editovaná hodnota

Definice ikon pro uživatelskou aplikaci

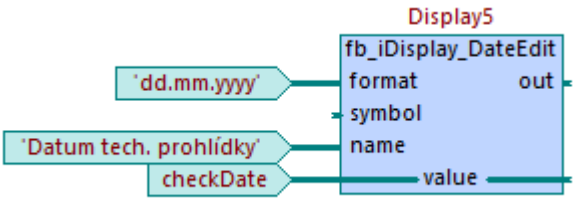
Uživatelsky definované ikony začínají hodnotou 10000. Pokud je například *symbolOn := 10000* tak aplikace při *value = TRUE* zobrazí ikonu danou obrázkem s názvem *10000.png*, který musí být uložen v adresáři WWW/iFOX/. Pokud soubor *10000.png* uložíme v projektu Mosaicu do složky SendRoot\WWW\iFOX\ pak se soubor automaticky uloží do souborového systému v PLC při prvním odeslání přeloženého programu do PLC.

Příklad

Předpokládejme, že chceme v uživatelské aplikaci umožnit nastavení datumu technické prohlídky. V jazyce ST bude program vypadat následovně:

```
VAR_GLOBAL RETAIN
  checkDate    : DATE;
END_VAR

PROGRAM prgMain
  VAR
    Display5   : fb_iDisplay_DateEdit := ( format := 'dd.mm.yyyy',
                                           name   := 'Datum tech. prohlídky');
  END_VAR

  Display5( value := checkDate);
END_PROGRAM
```

Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iDisplay_DateEdit* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

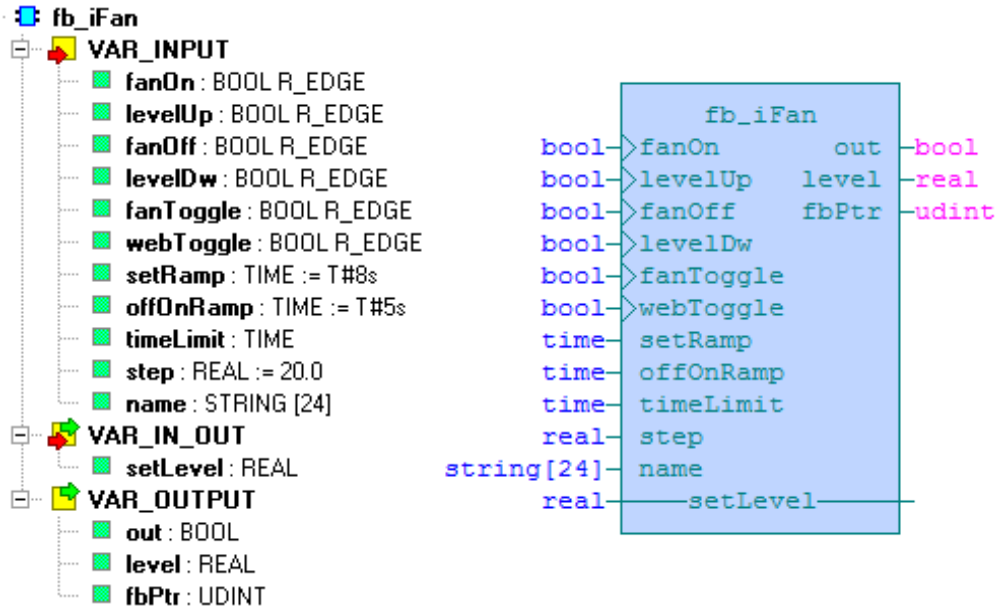
Každá instance *fb_iDisplay_DateEdit* přidá následující public proměnné:

	Proměnná	Typ	Význam
R	*GTSAP1_DISPLAY_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_DISPLAY_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
R	*GTSAP1_DISPLAY_edit*	BOOL	Povolení editace (0 = editace zakázaná)
R	*GTSAP1_DISPLAY_type*	USINT	Typ zobrazení (3 = DATE)
R	*GTSAP1_DISPLAY_symbol*	UINT	Aktuální kód symbolu
RW	*GTSAP1_DISPLAY_value*	DATE	Editovaná DATE hodnota
R	*GTSAP1_DISPLAY_format*	STRING[10]	Formát zobrazení

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.13 Funkční blok fb_iFan

Knihovna : *iControlLib*



Funkční blok *fb_iFan* je určen k řízení ventilátoru či obecného výstupu, který je třeba regulovat v rozsahu 0 – 100%.

Vstup fan*On* zapíná ventilátor, vstup fan*Off* ho vypíná. Vstup *fanToggle* přepíná ventilátor – pokud běží je vypnut a naopak. Vstup levelUp a levelDw slouží z zvýšení/snížení požadované úrovně výstupu o krok *step*. Ventilátor se rozjíždí plynule, rychlost určuje vstup *offOnRamp*. Ten udává celkový čas, za který se ventilátor rozjede z *0* na 100%. Při nastavování úrovně je rychlost zapínání resp. vypínání určena vstupem *setRamp*, který opět udává čas, za který se ventilátor přepne z *0* na 100%.

Vstup *webToggle* funguje stejně jako fan*Toggle* a slouží k ovládání ventilátou z web rozhraní. Vstup *timeLimit* umožňuje omezit dobu zapnutí. Pokud má hodnotu T#0s, doba není omezena. Pojmenování bloku pomocí proměnné *name* je určeno pro rozeznání bloku v uživatelské aplikaci.

Proměnná *setLevel* slouží k zapamatování poslední nastavené úrovně a měla by být založena v sekci VAR_GLOBAL RETAIN.

Výstup *out* signalizuje, že je ventilátor zapnutý na úroveň *level* (0...100%). Výstup *fbPtr* je pointer na funkční blok.

	Proměnná	Typ	Význam
VAR_INPUT			
	fanOn	BOOL R_EDGE	Náběžná hrana zapne ventilátor na úroveň *setLevel* Rychlost změny určuje vstup *ramp*
	levelUp	BOOL	Zvýšení úrovně, na kterou se ventilátor zapne (mění proměnnou *setLevel*) Rychlost změny určuje vstup *setRamp*
	fanOff	BOOL R_EDGE	Náběžná hrana vypne ventilátor Rychlost změny určuje vstup *ramp*

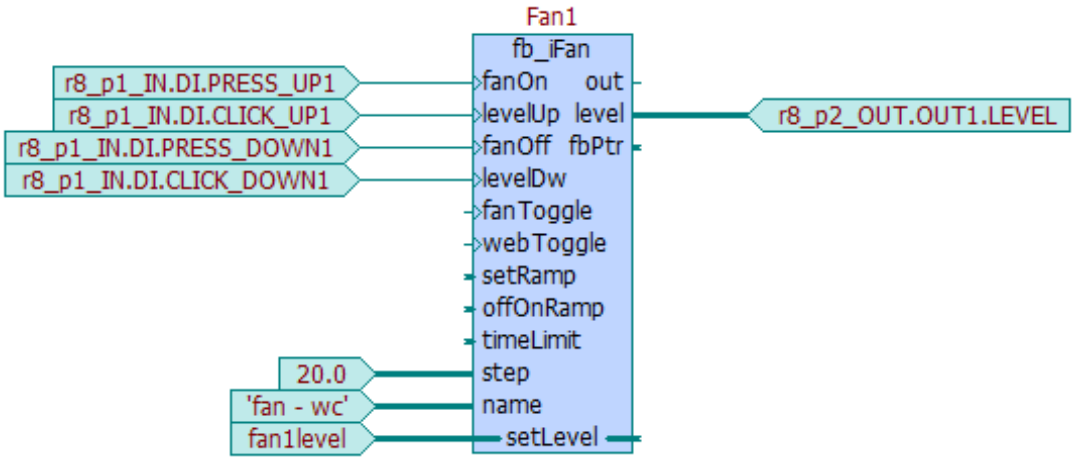
	Proměnná	Typ	Význam
	levelDw	BOOL	Snížení úrovně, na kterou se ventilátor zapne (mění proměnnou *setLevel*) Rychlost změny určuje vstup *setRamp*
	fanToggle	BOOL R_EDGE	Náběžná hrana přepne ventilátor Rychlost zapnutí/vypnutí určuje vstup *ramp*
	webToggle	BOOL R_EDGE	Náběžná hrana z web stránky přepne ventilátor
	setRamp	TIME	Doba, za kterou se ventilátor přepne z 0 na 100% pro změnu požadované úrovně [sec] Přednastavená hodnota 8 sec
	offOnRamp	TIME	Doba, za kterou se ventilátor přepne z 0 na 100% pro zapnutí a vypnutí světla [sec] Přednastavená hodnota 5 sec
	timeLimit	TIME	Omezení doby běhu ventilátoru Při T#0s není doba omezena
	step	REAL	Krok
	name	STRING[48]	Pojmenování bloku pro uživatelskou aplikaci
VAR_OUTPUT			
	out	BOOL	TRUE signalizuje zapnutý ventilátor
	level	REAL	Aktuální hodnota pro řízení ventilátoru [0...100%]
	fbPtr	UDINT	Pointer na funkční blok
VAR_IN_OUT			
	setLevel	REAL	Požadovaná úroveň, na kterou se má ventilátor zapnout [0...100%]

Předpokládejme, že potřebujeme řídit ventilátor stmívačem C-DM-0402M-RLC připojený na výstup OUT1. Ventilátor budeme ovládat nástěnným ovladačem C-WS-0200R-ABB. V jazyce ST bude program vypadat následovně:

```
VAR_GLOBAL RETAIN
  fan1level    : REAL;
END_VAR

PROGRAM prgMain
  VAR
    Fan1         : fb_iFan;
  END_VAR

  // stmivac ovladany CIB modulem C-WS-0200R-Logus
  Fan1( fanOn     := r8_p1_IN.DI.PRESS_UP1,
        levelUp   := r8_p1_IN.DI.CLICK_UP1,
```

```
        fanOff    := r8_p1_IN.DI.PRESS_DOWN1,
        levelDw   := r8_p1_IN.DI.CLICK_DOWN1,
        step      := 20.0,
        name      := 'fan - wc',
        setLevel  := fan1level,
        level       => r1_p0_AO1.ENG);
END_PROGRAM
```

Stisk tlačítka nahoru a dolu mění úroveň výstupu 0 – 100%. Dlouhý stisk ventilátor vypne/zapne. Stejnou funkci lze naprogramovat v jazyce CFC následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iFan* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

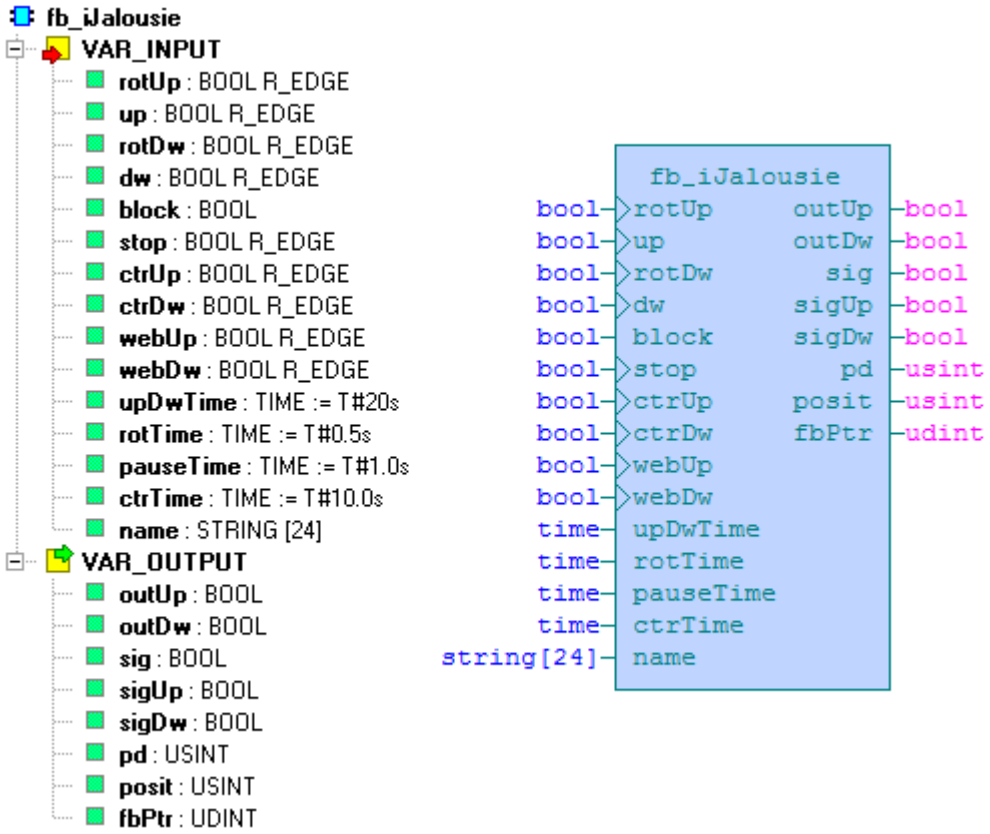
Každá instance *fb_iFan* přidá následující public proměnné:

	Proměnná	Typ	Význam
R	*GTSAP1_FAN_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_FAN_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_FAN_needAck*	BOOL	Nastavením na TRUE bude pro změnu stavu bloku vyžadováno potvrzení. Změnit volbu lze pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_FAN_onoff*	BOOL	Vypínání a zapínání ventilátoru z uživatelské aplikace.
RW	*GTSAP1_FAN_level*	REAL	Aktuální hodnota pro řízení ventilátoru

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.14 Funkční blok fb_iJalousie

Knihovna : *iControlLib*



Funkční blok *fb_iJalousie* slouží k ovládání žaluzií bez zpětné vazby aktuální pozice. Je schopen žaluzii posunout o přednastavený krok, sloužící k překlopení lamel, nebo aktivovat kompletní vytažení/zavření stínidla. Dále dokáže pracovat s prodlevou pro reverzaci směru pohybu použitého motoru.

Vstup *rotUp* slouží k posunu stínidla směrem vzhůru o jeden krok zatímco vstup *up* spouští kompletní vytažení žaluzie. Vstup *rotDw* aktivuje krok směrem dolů a *dw* kompletní zavření stínidla. Započatý kompletní pohyb lze přerušit aktivací jednoho z vstupů vyvolání pohybu (*rotUp*, *up*, *rotDw*, *dw)* stejně tak jako pomocí aktivace z web rozhraní, jedné z proměnných určené pro uživatelskou aplikaci nebo vstupu *stop*. Vstup *block = 1* zablokuje ovládací rozhraní daného bloku po dobu přítomnosti „1“. Vstupy *ctrUp* a *ctrDw* slouží pro připojení centrálního ovládání všech žaluzií. Z důvodu omezení proudového rázu je funkční blok vybaven vstupem *ctrTime* který slouží pro zadání zpoždění rozběhu pří aktivaci centrálního vytažení/spuštění. Zadáním různých hodnot se velikost proudového rázu omezí. Vstupy *webUp* a *webDw* fungují stejně jako *up* a *dw* a jsou určeny pro ovládání z web stránky. Pojmenování bloku pomocí proměnné *name* je určeno pro rozeznání bloku v uživatelské aplikaci.

Výstup *outUp* slouží k spínání relé pro směr vzhůru. Výstup *outDw* slouží k spínání relé pro směr dolů. Výstup *sig* funguje jako detekce pohybu stínidla. Výstup *sigUp* je nastaven na TRUE po kompletním pohybu vzhůru. Výstup *sigDw* je nastaven na TRUE po kompletním pohybu dolů. Výstup pd signalizuje dobu impulzu pro natočení žaluzií a *posit* pozici žaluzie. Výstup *fbPtr* je pointer na funkční blok.

	Proměnná	Typ	Význam
VAR_INPUT			
	rotUp	BOOL R_EDGE	Pootočit lamely o krok vzhůru
	up	BOOL R_EDGE	Kompletní pohyb vzhůru
	rotDw	BOOL R_EDGE	Pootočit lamely o krok dolů
	dw	BOOL R_EDGE	Kompletní pohyb dolů
	block	BOOL	Blokace vstupů bloku
	stop	BOOL R_EDGE	Zastavení pohybu
	ctrUp	BOOL R_EDGE	Centrální pohyb vzhůru
	ctrDw	BOOL R_EDGE	Centrální pohyb dolů
	webUp	BOOL R_EDGE	Aktivace "Kompletní pohyb vzhůru" z web rozhraní
	webDw	BOOL R_EDGE	Aktivace "Kompletní pohyb dolů" z web rozhraní
	upDwTime	TIME	Čas pro kompletní pohyb
	rotTime	TIME	Délka kroku [sec]
	pauseTime	TIME	Prodleva pro reverzaci pohybu[sec]
	ctrTime	TIME	Zpoždění pohybu centrální aktivace [sec]
	name	STRING[48]	Jméno žaluzie[sec]
VAR_OUTPUT			
	outUp	BOOL	Žaluzie nahoru
	outDw	BOOL	Žaluzie dolů
	sig	BOOL	Signalizace chodu
	sigUp	BOOL	Signalizace žaluzie nahoře
	sigDw	BOOL	Signalizace žaluzie dole

	Proměnná	Typ	Význam
	pd	USINT	Doba impulzu pro natočení žaluzií
	posit	USINT	Poloha žaluzie [%]
	fbPtr	UDINT	Pointer na funkční blok

Použití funkčního bloku pro žaluzie připojené k CIB modul reléových výstupů C-OR-0202B. Pro jejich ovládání použijeme tlačítkový CIB modul C-WS-0200R-Logus Výrobcem udaná prodleva pro reverzaci je 0.5 sec a naměřená doba kompletního pojezdu je 20 sec.

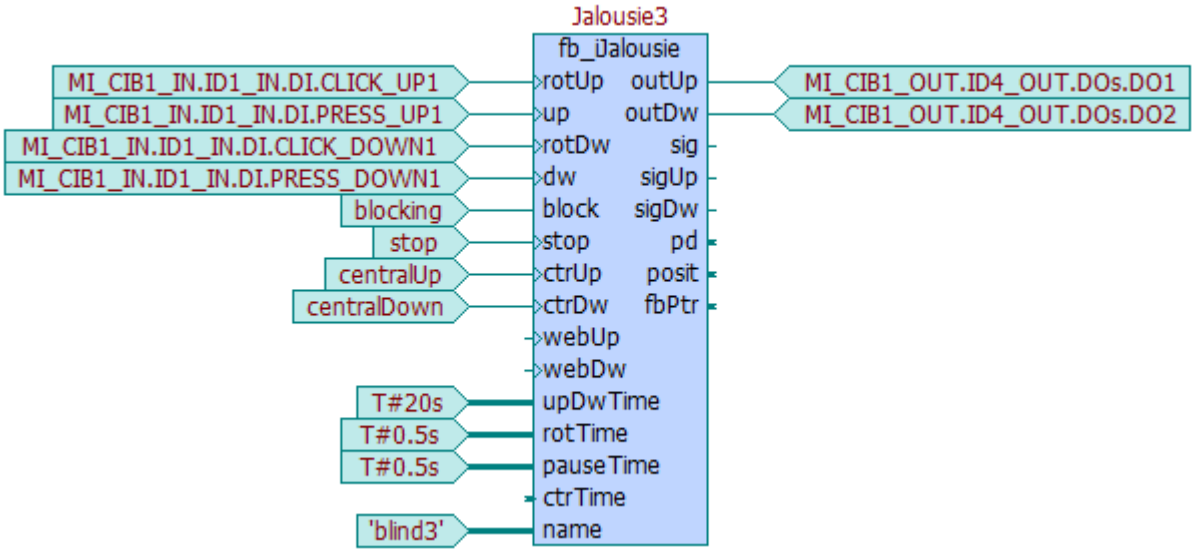
```
PROGRAM prgMain
VAR
  Jalousie3     : fb_iJalousie;
  blocking      : bool;
  stop          : bool;
  centralUp     : bool;
  centralDown   : bool;
END_VAR

  //jednoduche ovladani zaluzii
  Jalousie3(rotUp       := MI_CIB1_IN.ID1_IN.DI.CLICK_UP1,
            up          := MI_CIB1_IN.ID1_IN.DI.PRESS_UP1,
            rotDw       := MI_CIB1_IN.ID1_IN.DI.CLICK_DOWN1,
            dw          := MI_CIB1_IN.ID1_IN.DI.PRESS_DOWN1,
            block       := blocking,
            stop        := stop,
            ctrUp       := centralUp,
            ctrDw       := centralDown,
            upDwTime    := T#20s,
            rotTime     := T#0.5s,
            pauseTime   := T#0.5s,
            name        := 'zaluzie1',
            outUp       => MI_CIB1_OUT.ID4_OUT.DOs.DO1,
            outDw       => MI_CIB1_OUT.ID4_OUT.DOs.DO2);
END_PROGRAM
```

Krátký stisk tlačítka up aktivuje pootočení o krok směrem vzhůru. Dlouhý stisk tlačítka up aktivuje kompletní pohyb směrem vzhůru. Tlačítko down funguje obdobně ale místo aktivace pohybu vzhůru aktivuje pohyb dolů. Stejný program vytvořen v jazyce CFC vypadá následně:

Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iJalousie* se do souboru „iFox-trot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

Každá instance *fb_iJalousie* přidá následující public proměnné:

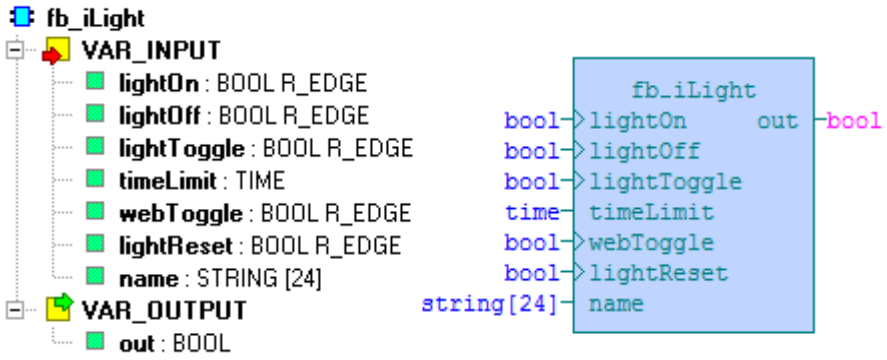


	Proměnná	Typ	Význam
R	*GTSAP1_SHUTTER_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_SHUTTER_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_SHUTTER_needAck*	BOOL	Nastavením na TRUE bude pro změnu stavu bloku vyžadováno potvrzení. Změnit volbu lze pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_SHUTTER_needPin*	STRING[10]	V případě neprázdného řetězce bude pro změnu stavu bloku vyžadován uvedený PIN. Změnu PINu lze provést pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
R	*GTSAP1_SHUTTER_up*	BOOL	Žaluzie se pohybuje nahoru
R	*GTSAP1_SHUTTER_down*	BOOL	Žaluzie se pohybuje dolů
R	*GTSAP1_SHUTTER_run*	BOOL	Žaluzie v pohybu
R	*GTSAP1_SHUTTER_uppos*	BOOL	Žaluzie nahoře
R	*GTSAP1_SHUTTER_downpos*	BOOL	Žaluzie dole
W	*GTSAP1_SHUTTER_up_control*	BOOL	Vyvolání pohybu nahoru
W	*GTSAP1_SHUTTER_down_control*	BOOL	Vyvolání pohybu dolů
W	*GTSAP1_SHUTTER_rotup_control*	BOOL	Vyvolání kroku nahoru
W	*GTSAP1_SHUTTER_rotdown_control*	BOOL	Vyvolání kroku dolů

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.15 *Funkční blok fb_iLight*

Knihovna : *iControlLib*



Funkční blok *fb_iLight* je určen k řízení světla pomocí tlačítek. Blok umožňuje jak 2-tlačítkové ovládání (pomocí vstupů *lightOn* a *lightOff*) tak 1-tlačítkové ovládání (pomocí vstupu *lightToggle*).

Vstup *lightOn* zapíná světlo, vstup *lightOff* světlo vypíná. Vstup *lightToggle* přepíná světlo – pokud svítí tak ho zhasne a pokud je zhasnuté tak ho rozsvítí. Vstup *webToggle* funguje stejně jako *lightToggle* a slouží k zapínání a vypínání světla z web rozhraní (tuto proměnnou je třeba nastavit na TRUE v případě, že chceme přepnout světlo z web stránky). Vstup *timeLimit* umožňuje omezit dobu svícení. Pokud má hodnotu T#0s tak doba svícení není omezena. A konečně vstup *lightReset* slouží jako centrální vypnutí všech světel. Všechny vstupy typu BOOL reagují na náběžnou hranu vstupního signálu.

Výstup *out* slouží k ovládání světla.

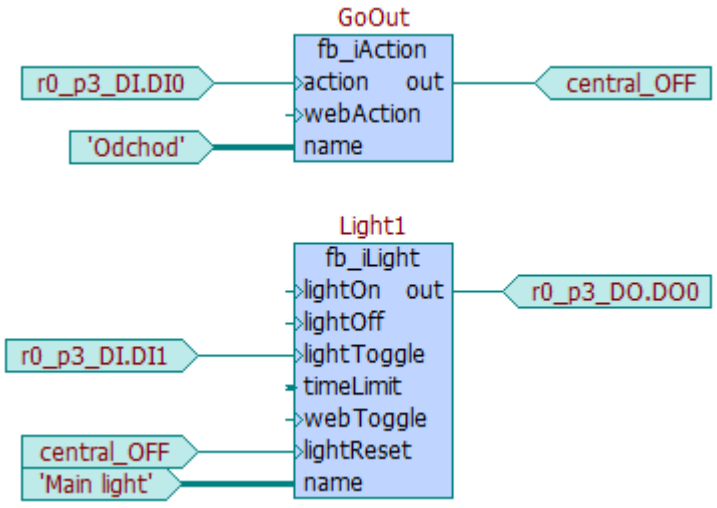
	Proměnná	***Typ***	***Význam***
VAR_INPUT			
	lightOn	BOOL R_EDGE	Náběžná hrana zapne světlo
	lightOff	BOOL R_EDGE	Náběžná hrana vypne světlo
	lightToggle	BOOL R_EDGE	Náběžná hrana přepne světlo
	timeLimit	TIME	Omezení doby svícení Při T#0s není doba omezena
	webToggle	BOOL R_EDGE	Ovládání světla z web stránky
	lightReset	BOOL R_EDGE	Vstup pro centrální vypnutí světel (odchodové tlačítko)
	name	STRING[48]	název světla
VAR_OUTPUT			

	out	BOOL	Ovládání světla

Předpokládejme, že potřebujeme ovládat světlo připojené v PLC systému na binární výstup DO0. Světlo budeme ovládat tlačítkem připojeným na binární vstup DI1. V jazyce ST bude program vypadat následovně:

```
VAR_GLOBAL
  central_OFF : BOOL;
END_VAR

PROGRAM prgMain
  VAR
    GoOut       : fb_iAction;
    Light1      : fb_iLight;
  END_VAR

  // akce centralniho zhasnuti
  GoOut( action := r0_p3_DI.DI0, out => central_OFF, name := 'Odchod');

  // 1-tlacitkove ovladani svetla
  Light1( lightToggle := r0_p3_DI.DI1,
          lightReset  := central_OFF,
          out         => r0_p3_DO.DO0);
END_PROGRAM
```

Každý stisk tlačítka změní stav světla. Pomocí proměnné *Main.Light1.webToggle* lze světlo ovládat z web stránky. Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Funkce centrálního zhasnutí je v tomto příkladu řešena blokem *fb_iAction*. Centrální zhasnutí lze vyvolat i web stránky nastavením proměnné *Main.GoOut.webAction* na TRUE. Globální proměnná *central_OFF* pak bude použita pro všechny bloky řídící světla jako vstup pro centrální vypnutí.

Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iLight* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

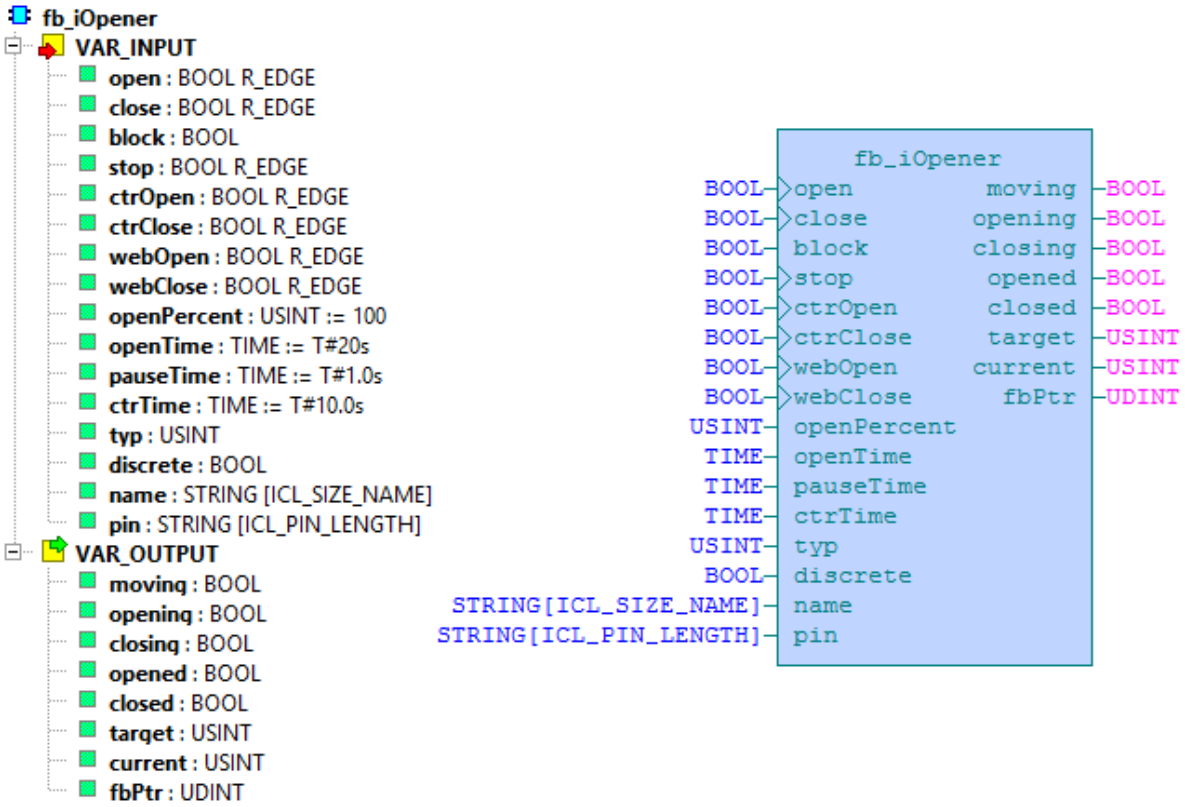
Každá instance *fb_iLight* přidá následující public proměnné:

	Proměnná	**Typ**	**Význam**
R	*GTSAP1_LIGHT_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_LIGHT_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_LIGHT_needAck*	BOOL	Nastavením na TRUE bude pro změnu stavu bloku vyžadováno potvrzení. Změnit volbu lze pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
R	*GTSAP1_LIGHT_type*	BOOL	Typ světla (0 = onOff)
RW	*GTSAP1_LIGHT_onoff*	BOOL	Vypínání a zapínání světla z uživatelské aplikace.

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.16 Funkční blok fb_iOpener

Knihovna : *iControlLib*



Funkční blok *fb_iOpener* je univerzálním funkčním blokem určeným pro systémy pro otevírání dveří, bran, garážových vrat, rolet apod., a to bez zpětné vazby aktuální pozice. Umožňuje systém zcela otevřít, zavřít nebo případně otevřít pouze v zadaném procentuálním rozsahu. Jelikož se jedná o funkční blok pro systémy bez zpětné vazby, je doba sepnutí aktoru nutná pro plné otevření/zavření dána vstupním parametrem bloku. Blok dále dokáže pracovat s prodlevou pro reverzaci směru pohybu použitého motoru.

Náběžná hrana na vstupu *open* spustí proces otevírání. Hrana na vstupu *close* spustí zavírání. Aktivní proces lze přerušit nastavením vstupu *stop*. Vstup *block* zablokuje v případě logické hodnoty *TRUE* ovládací rozhraní daného bloku. Vstupy *ctrOpen* a *ctrClose* slouží pro připojení centrálního ovládání. Z důvodu omezení proudového rázu je funkční blok vybaven vstupem *ctrTime,* který slouží pro zadání zpoždění rozběhu pří aktivaci centrálního otevření/zavření. Zadáním různých hodnot se velikost proudového rázu omezí. Vstupy *webOpen* a *webClose* fungují stejně jako *open* a *close* a jsou určeny pro ovládání bloku z web stránky. Vstupem *discrete* lze specifikovat, že systém podporuje pouze stav plného otevření nebo zavření. V případě, že není vstup *discrete* nastaven, lze míru otevření ovládat vstupem *openPercent*. Vstupní proměnná *typ* určuje, jaký systém daný funkční blok realizuje. Tato informace může být využívána uživatelskou aplikací pro výběr vhodné grafiky. Pojmenování bloku pomocí proměnné *name* je určeno pro rozeznání bloku v uživatelské aplikaci.

Výstupy *moving* je možné použít pro spínání hlavního aktoru. Výstupy *opening* a *closing* určují směr, tedy otvírání nebo zavírání. Výstupní proměnná *opened* je nastavena v případě, že je systém úplně nebo i částečně otevřen. Proměnná *closed* je nastavena pouze v případě, že je systém zcela uzavřen. Výstup *current* určuje na kolik procent je systém aktuálně otevřen. *Target* specifikuje cílový stav otevření v případě provádění pohybu. Výstup *fbPtr* je pointer na funkční blok.

Popis proměnných :

	Proměnná	Typ	Význam
VAR_INPUT			
	open	BOOL R_EDGE	požadavek na otevření
	close	BOOL R_EDGE	požadavek na zavření
	block	BOOL	blokace ovládacích vstupů
	stop	BOOL R_EDGE	zastavení pohybu
	ctrOpen	BOOL R_EDGE	centrální otevření
	ctrClose	BOOL R_EDGE	centrální zavření
	webOpen	BOOL R_EDGE	požadavek na otevření z web rozhraní
	webClose	BOOL R_EDGE	požadavek na zavření z web rozhraní
	openPercent	USINT	požadavek na procentuální otevření
	openTime	TIME	maximální doba běhu pohonu
	pauseTime	TIME	prodleva před otočením směru
	ctrTime	TIME	zpoždění centrálního pohybu
	typ	USINT	typ zařízení
	discrete	BOOL	diskrétní chování plného otevření/zavření
	name	STRING[48]	název ovladače otevírání
VAR_OUTPUT			
	moving	BOOL	signalizace chodu
	opening	BOOL	stav otevírání
	closing	BOOL	stav zavírání
	opened	BOOL	stav otevřeno
	closed	BOOL	stav zavřeno
	target	USINT	cílový stav [%]
	current	USINT	aktuální stav [%]
	fbPtr	UDINT	pointer na funkční blok

Definované konstanty pro typ zařízení:

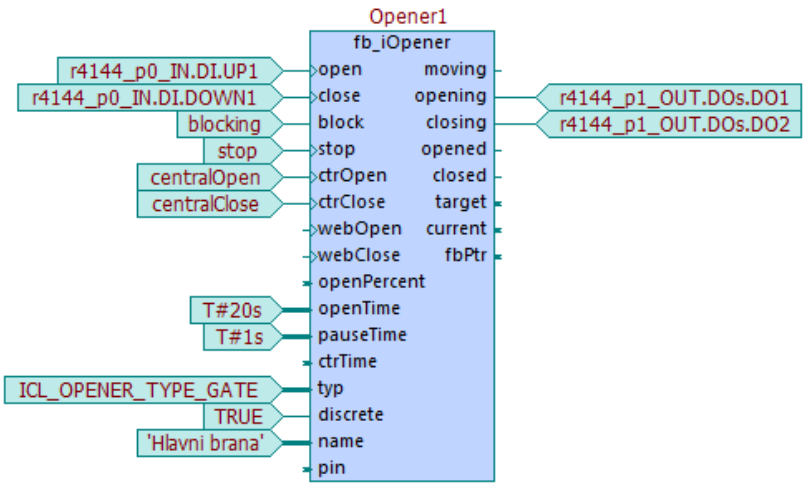
```
VAR_GLOBAL CONSTANT
  ICL_OPENER_TYPE_BLINDS    : USINT := 0; // typ zařízení - rolety
```

```
  ICL_OPENER_TYPE_DOOR     : USINT := 1; // typ zařízení - dveře
  ICL_OPENER_TYPE_GATE     : USINT := 2; // typ zařízení - brána
  ICL_OPENER_TYPE_GARAGE   : USINT := 3; // typ zařízení - garážová vrata
END_VAR
```

Použití funkčního bloku pro ovládání brány připojené k CIB modulu reléových výstupů C-OR-0202B. Pro jejich ovládání použijeme například tlačítkový CIB modul C-WS-0200R-Logus. Dobu pro otevření brány uvažujme 20 sec. Ovládání bude podporovat pouze diskrétní režim, tedy režim plného otevření / zavření.

```
PROGRAM prgMain
VAR
  Opener1      : fb_iOpener;
  blocking     : bool;
  stop         : bool;
  centralOpen  : bool;
  centralClose : bool;
END_VAR

  Opener1(open         := r4144_p0_IN.DI.UP1,
          close        := r4144_p0_IN.DI.DOWN1,
          block        := blocking,
          stop         := stop,
          ctrOpen      := centralOpen,
          ctrClose     := centralClose,
          openTime     := T#20s,
          pauseTime    := T#1s,
          typ          := ICL_OPENER_TYPE_GATE,
          discrete     := TRUE,
          name         := 'Hlavni brana',
          opening      => r4144_p1_OUT.DOs.DO1,
          closing      => r4144_p1_OUT.DOs.DO2);
END_PROGRAM
```

Stejný program vytvořen v jazyce CFC vypadá následně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iOpener* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

Každá instance *fb_iOpener* přidá následující public proměnné:

	Proměnná	Typ	Význam
R	*GTSAP1_OPENER_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_OPENER_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_ACTION_needAck*	BOOL	Nastavením na TRUE bude pro změnu stavu bloku vyžadováno potvrzení. Změnit volbu lze pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_ACTION_needPin*	STRING[10]	V případě neprázdného řetězce bude pro změnu stavu bloku vyžadován uvedený PIN. Změnu PINu lze provést pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
R	*GTSAP1_OPENER_moving*	BOOL	Signalizace aktivního pohybu
R	*GTSAP1_OPENER_block*	BOOL	Signalizace blokace ovládání
R	*GTSAP1_OPENER_discrete*	BOOL	Signalizace diskrétní režimu
W	*GTSAP1_OPENER_target*	USINT	Cílový stav otevření v procentech
R	*GTSAP1_OPENER_current*	USINT	Aktuální stav otevření v procentech
R	*GTSAP1_OPENER_type*	USINT	Typ zařízení

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.17 Funkční blok fb_iPublicEncoding

Knihovna : *iControlLib*

Funkční blok *fb_iPublicEncoding* zveřejňuje do mobilní aplikace jaké kódování proměnných typu STRING bylo použito v prostředí Mosaic. Vstup *encoding* udává použitý typ kódování. Přípustné hodnoty jsou:

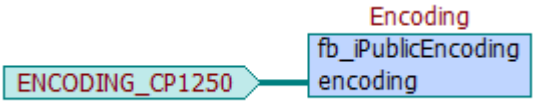
- *ENCODING_UTF8*
- *ENCODING_CP1250*

- *ENCODING_CP1251*
- *ENCODING_CP1252*
- *ENCODING_CP1253*
- *ENCODING_CP1254*
- *ENCODING_CP1255*
- *ENCODING_CP1256*
- *ENCODING_CP1257*
- *ENCODING_CP1258*

Výchozí hodnota je *ENCODING_CP1250*. V programu postačuje jedna instance bloku *fb_iPublicEncoding*.

	Proměnná	***Typ***	***Význam***
VAR_INPUT			
	*encoding*	UINT	Kódování použité v prostředí Mosaic

Příklad použití:



Integrace s uživatelskými aplikacemi

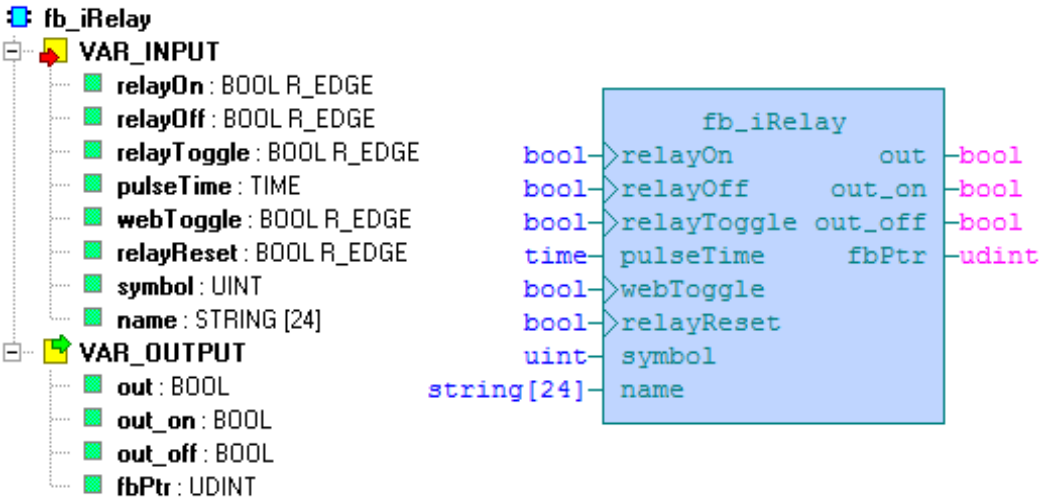
Instance *fb_iPublicEncoding* přidá následující public proměnné:

	Proměnná	**Typ**	**Význam**
R	*GTSAP1_ENCODING*	UINT	Typ kódování

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.18 Funkční blok fb_iRelay

Knihovna : *iControlLib*



Funkční blok *fb_iRelay* je určen k řízení releového výstupu.

Vstup *relayOn* spíná relé, vstup *relayOff* relé rozpíná. Vstup *relayToggle* přepíná stav relé. Vstup *webToggle* funguje stejně jako *relayToggle* a slouží k ovládání relé z web rozhraní (tuto proměnnou je třeba nastavit na TRUE v případě, že chceme přepnout relé z web stránky). Vstup *pulseTime* určuje jak dlouho bude relé sepnuté. Pokud má hodnotu T#0s tak doba sepnutí relé není omezena. A konečně vstup *relayReset* slouží jako centrální rozepnutí všech relé. Všechny vstupy typu BOOL reagují na náběžnou hranu vstupního signálu.

Výstup *out* slouží k ovládání relé. Výstup *fbPtr* je pointer na funkční blok.

Popis proměnných :

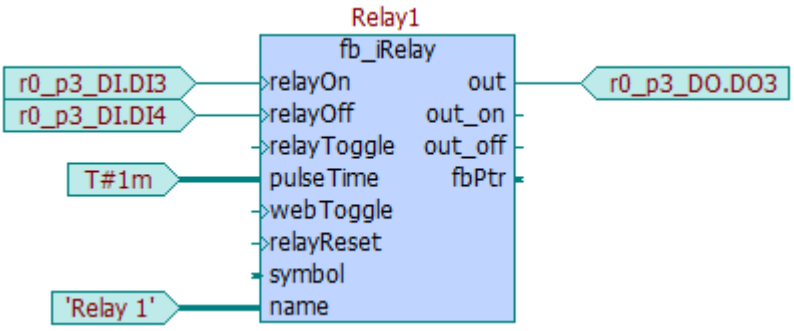
	Proměnná	Typ	Význam
VAR_INPUT			
	relayOn	BOOL R_EDGE	Náběžná hrana sepne relé
	relayOff	BOOL R_EDGE	Náběžná hrana rozepne relé
	relayToggle	BOOL R_EDGE	Náběžná hrana přepne relé
	pulseTime	TIME	Doba výstupního pulzu Při T#0s není doba sepnutí relé omezena
	webToggle	BOOL R_EDGE	Ovládání relé z web stránky
	relayReset	BOOL R_EDGE	Vstup pro centrální rozepnutí relé (odchodové tlačítko)
	symbol	UINT	Kód ikony zobrazené v uživatelské aplikaci

	Proměnná	Typ	Význam
	name	STRING[48]	název
VAR_OUTPUT			
	out	BOOL	Ovládání relé
	out_on	BOOL	Pulz při náběžné hraně out (při přechodu 0 → 1)
	out_off	BOOL	Pulz při sestupné hraně out (při přechodu 1 → 0)
	fbPtr	UDINT	Pointer na funkční blok

Předpokládejme, že potřebujeme ovládat reléový výstup DO3. Při každém sepnutí binárního vstupu DI1 chceme sepnout relé na dobu 1 minuta. Sepnutím vstupu DI2 relé rozepneme. V jazyce ST bude program vypadat následovně:

```
PROGRAM prgMain
  VAR
    Relay1      : fb_iRelay;
  END_VAR

  // ovladani rele
  Relay1( relayOn   := r0_p3_DI.DI1,
          relayOff  := r0_p3_DI.DI2,
          pulseTime := T#1m,
          name      := 'Relay 1',
          out       => r0_p3_DO.DO3);
END_PROGRAM
```

Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iRelay* se do souboru „iFox-trot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

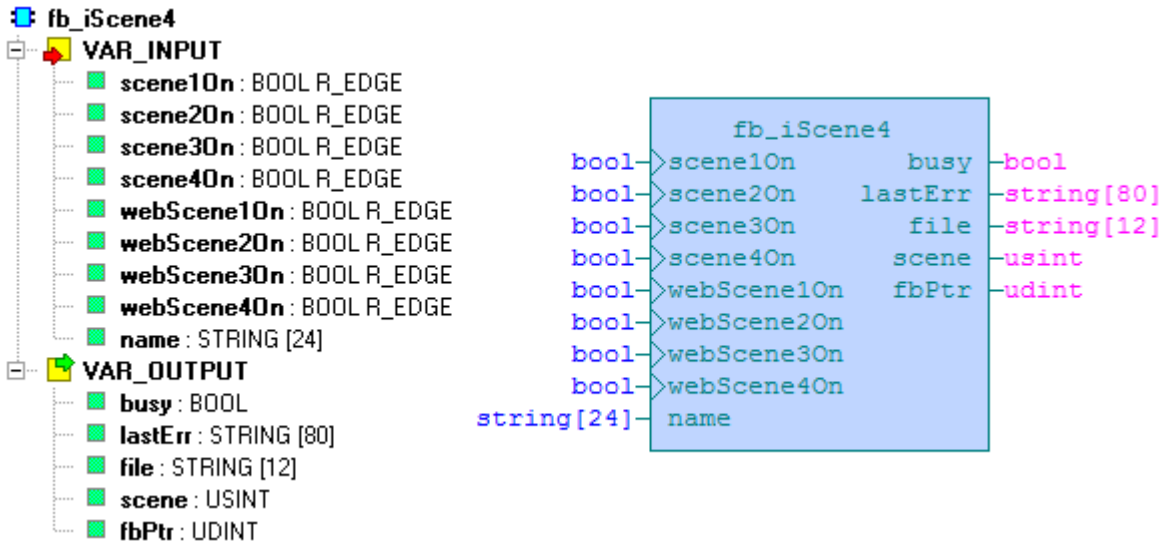
Každá instance *fb_iRelay* přidá následující public proměnné:

	Proměnná	Typ	Význam
R	*GTSAP1_RELAY_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_RELAY_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_RELAY_needAck*	BOOL	Nastavením na TRUE bude pro změnu stavu bloku vyžadováno potvrzení. Změnit volbu lze pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_RELAY_needPin*	STRING[10]	V případě neprázdného řetězce bude pro změnu stavu bloku vyžadován uvedený PIN. Změnu PINu lze provést pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
R	*GTSAP1_RELAY_type*	USINT	Typ zobrazeni (0 = BOOL)
R	*GTSAP1_RELAY_symbol*	UINT	Kód ikony
RW	*GTSAP1_RELAY_onoff*	BOOL	Vypínání a zapínání relé z uživatelské aplikace.

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.19 Funkční blok fb_iScene4, fb_iScene8

Knihovna : *iControlLib*



Funkční bloky *fb_iScene4* a *fb_iScene8* jsou určeny k nastavení scény. Blok *fb_iScene4* umožňuje nastavit max. 4 různé scény, blok *fb_iScene8* je určen pro 8 různých scén. Scénou se rozumí konkrétní nastavení pro světla, žaluzie, atd. Sestavení scén se provede v uživatelské aplikaci s danou podporou. Scéna může obsahovat nastavení pro

všechny prvky, které lze z aplikace ovládat. Popis v této kapitole odpovídá bloku *fb_iScene4*, blok *fb_iScene8* se chová analogicky.

Vstupy *scene1On, scene2On, scene3On a scene4On* nastavují scénu odpovídajícího čísla. Na tyto vstupy lze připojit tlačítka, pomocí kterých lze volit požadovanou scénu. Pro aktivaci scény z web stránky slouží vstupy *webScene1On* až *webScene4On*. Všechny vstupy typu BOOL reagují na náběžnou hranu vstupního signálu.

Výstup *busy* je nastaven na TRUE dokud je blok *fb_iScene4* zaneprázdněn nastavováním scény. Pokud dojde při nastavování scény k nějaké chybě, tak je ve výstupu *lastErr* popis chyby. Výstup *file* obsahuje společný základ jména všech souborů, ve kterých jsou popisy s nastavením scén. Pokud bude *file = SCN_1399.sc?* pak první scéna bude popsaná souborem SCN_1399.sc1, druhá scéna bude popsaná souborem SCN_1399.sc2, atd. Soubory s popisem scén jsou generovány uživatelskou aplikací s danou podporou, mají formát JSON a jsou uloženy na SD kartě PLC systému v adresáři WWW/iFOX/. Výstup *scene* definuje aktivní scénu. Výstup *fbPtr* je pointer na funkční blok.

Popis proměnných :

	Proměnná	Typ	Význam
VAR_INPUT			
	scene1On	BOOL R_EDGE	Náběžná hrana nastaví 1.scénu
	scene2On	BOOL R_EDGE	Náběžná hrana nastaví 2.scénu
	scene3On	BOOL R_EDGE	Náběžná hrana nastaví 3.scénu
	scene4On	BOOL R_EDGE	Náběžná hrana nastaví 4.scénu
	webScene1On	BOOL R_EDGE	Náběžná hrana nastaví 1.scénu (určeno pro ovladání z web stránky)
	webScene2On	BOOL R_EDGE	Náběžná hrana nastaví 2.scénu (určeno pro ovladání z web stránky)
	webScene3On	BOOL R_EDGE	Náběžná hrana nastaví 3.scénu (určeno pro ovladání z web stránky)
	webScene4On	BOOL R_EDGE	Náběžná hrana nastaví 4.scénu (určeno pro ovladání z web stránky)
	name	STRING[48]	název
VAR_OUTPUT			
	busy	BOOL	TRUE znamená, že blok je zaneprázdněn nastavováním scény
	lastErr	STRING	Popis případné chyby
	file	STRING[12]	Základ jména všech souborů s popisem scén

	scene	USINT	Aktivní scéna
	fbPtr	UDINT	Pointer na funkční blok

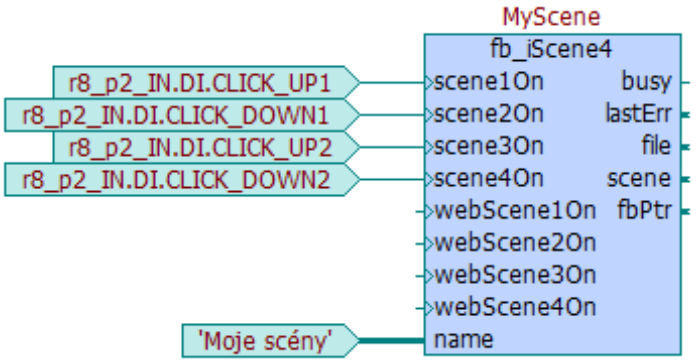
Pokud v programu použijeme jednu instanci funkčního bloku *fb_iScene4* pak se v uživatelské aplikaci s danou podporou objeví možnost sestavení čtyř různých scén. Do každé scény lze pak umístit libovolné množství prvků, které se v aplikaci ovládají a tyto prvky lze nastavit na konkrétní hodnoty. Například lze nastavit, že `svetlo1` bude ve scéně svítit, `zásuvka1` bude ve scéně vypnutá, stmívané `světlo2` bude svítit na 90% a barevné `svetloRGB` bude nastaveno na určitou barvu a bude svítit na 100%. Aplikace uloží nastavení každé scény do JSON souboru, který může vypadat např. následovně:

```
{
  "name": "Moje scény",
  "index": 3,
  "scene": {
    "main": {
      "svetlo1": {
        "GTSAP1_LIGHT_onoff": 1
      },
      "zasuvka1": {
        "GTSAP1_SOCKET_onoff": false
      },
      "svetlo2": {
        "GTSAP1_LIGHT_onoff": 1,
        "GTSAP1_LIGHT_dimlevel": 90.0
      },
      "svetloRGB": {
        "GTSAP1_LIGHT_onoff": 1,
        "GTSAP1_LIGHT_dimlevel": 100.0,
        "GTSAP1_LIGHT_rgb": 4227264
      }
    }
  }
}
```

Scény nastavené v uživatelské aplikaci bude možné aktivovat (spouštět) také pomocí tlačítek (např. modulem C-WS-0400R-Logus). V jazyce ST bude program vypadat následovně:

```
PROGRAM prgMain16
  VAR
    MyScene : fb_iScene4;
  END_VAR

  MyScene( scene1On := r8_p2_IN.DI.CLICK_UP1,
           scene2On := r8_p2_IN.DI.CLICK_DOWN1,
           scene3On := r8_p2_IN.DI.CLICK_UP2,
           scene4On := r8_p2_IN.DI.CLICK_DOWN2,
           name     := 'Moje scény');
END_PROGRAM
```

Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iScene4* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

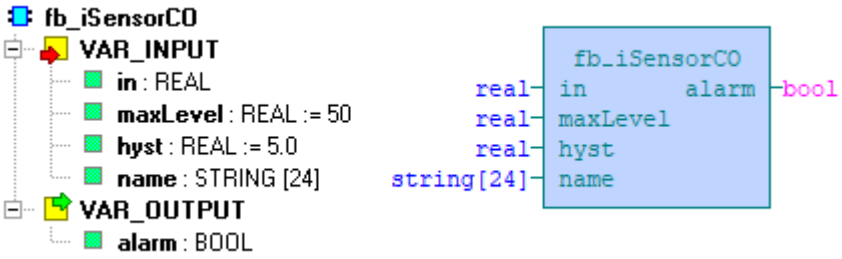
Každá instance *fb_iScene4* přidá následující public proměnné:

	Proměnná	Typ	Význam
R	*GTSAP1_SCENE_name*	STRING[48]	Kopie vstupu *name*
R	*GTSAP1_SCENE_file*	STRING[24]	Společný základ jména JSON souboru
RW	*GTSAP1_SCENE_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_SCENE_needAck*	BOOL	Nastavením na TRUE bude pro změnu stavu bloku vyžadováno potvrzení. Změnit volbu lze pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
R	*GTSAP1_SCENE_num*	USINT	Počet scén (4 nebo 8)
W	*GTSAP1_SCENE_set1*	BOOL	Aktivace 1.scény z uživatelské aplikace
W	*GTSAP1_SCENE_set2*	BOOL	Aktivace 2.scény z uživatelské aplikace
W	*GTSAP1_SCENE_set3*	BOOL	Aktivace 3.scény z uživatelské aplikace
W	*GTSAP1_SCENE_set4*	BOOL	Aktivace 4.scény z uživatelské aplikace

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.20 Funkční blok fb_iSensorCO

Knihovna : *iControlLib*



Funkční blok *fb_iSensorCO* je určen k vyhodnocení naměřených hodnot oxidu uhelnatého. Vstup *in* slouží k připojení měřené hodnoty CO (například z modulu C-AQ-005R), která se porovnává se vstupem *maxLevel*.

Pokud vstup *in* překročí hodnotu danou vstupem *maxLevel*, pak je nastaven výstup *alarm*. Hysterezi při nastavování výstupu *alarm* udává vstup *hyst*.

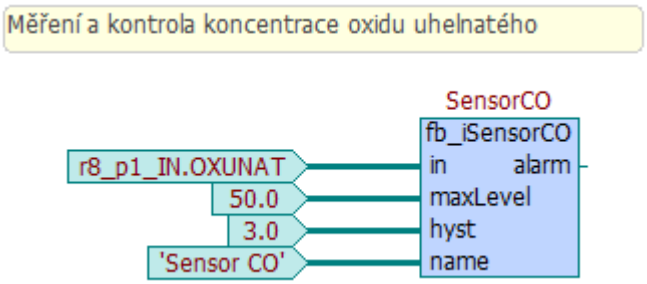
Popis proměnných :

	Proměnná	Typ	Význam
VAR_INPUT			
	in	REAL	Vstup pro připojení hodnoty koncentrace oxidu uhelnatého
	maxLevel	REAL	Prahová hodnota oxidu uhelnatého Přednastavená hodnota je 50 ppm
	hyst	REAL	Hystereze pro nastavení výstupu *alarm* Přednastavená hodnota je 3 ppm
	name	STRING[48]	název
VAR_OUTPUT			
	alarm	BOOL	TRUE pokud je *in* > *maxLevel* FALSE pokud *in* < *maxLevel - hyst*

Předpokládejme, že potřebujeme měřit koncentraci CO a vyhodnocovat překročení povolené koncentrace (včetně signalizace překročení do mobilní aplikace). Pro měření bude použit modul C-AQ-005R. V jazyce ST bude program vypadat následovně:

```
PROGRAM prgMain
  VAR
    SensorCO  : fb_iSensorCO := ( name := 'Sensor CO');
  END_VAR

  SensorCO(in := r8_p1_IN.OXUNAT);
END_PROGRAM
```

Limitní hodnotou pro koncentraci CO je 50 ppm. Pokud měřená hodnota překročí 50 ppm, pak bude nastaven výstup *alarm*. Ten zůstane nastaven až do doby, kdy koncentrace CO klesne pod 47 ppm. Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iSensorCO* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

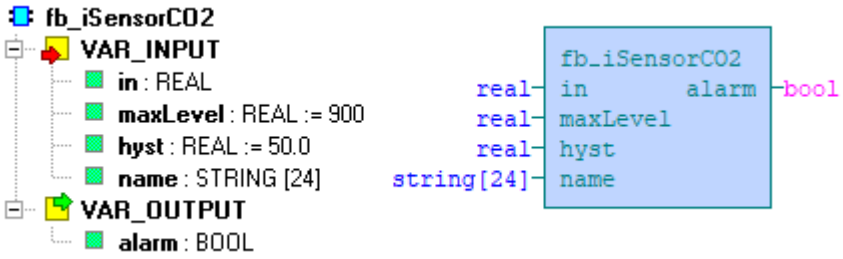
Každá instance *fb_iSensorCO* přidá následující public proměnné:

	Proměnná	**Typ**	**Význam**
R	*GTSAP1_DISPLAY_name*	STRING[48]	Název bloku (kopie vstupu *name)*
R W	*GTSAP1_DISPLAY_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
R	*GTSAP1_DISPLAY_edit*	BOOL	Povolení editace (0 = editace zakázaná)
R	*GTSAP1_DISPLAY_alarm*	BOOL	Value > maxValue
R	*GTSAP1_DISPLAY_type*	USINT	Typ zobrazení (1 = REAL)
R	*GTSAP1_DISPLAY_symbol*	UINT	Aktuální kód symbolu (105)
R	*GTSAP1_DISPLAY_value*	REAL	Zobrazovaná koncentrace CO
R	*GTSAP1_DISPLAY_unit*	STRING[8]	Fyzikální jednotka (ppm)
R	*GTSAP1_DISPLAY_precision*	INT	Počet zobrazených desetinných míst (0)
R	*GTSAP1_DISPLAY_maxValue*	REAL	Max. povolená hodnota

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.21 Funkční blok fb_iSensorCO2

Knihovna : *iControlLib*



Funkční blok *fb_iSensorCO2* je určen k vyhodnocení naměřených hodnot oxidu uhličitého. Vstup *in* slouží k připojení měřené hodnoty CO2 (například z modulu C-AQ-001R), která se porovnává se vstupem *maxLevel*.

Pokud vstup *in* překročí hodnotu danou vstupem *maxLevel*, pak je nastaven výstup *alarm*. Hysterezi při nastavování výstupu *alarm* udává vstup *hyst*.

Popis proměnných :

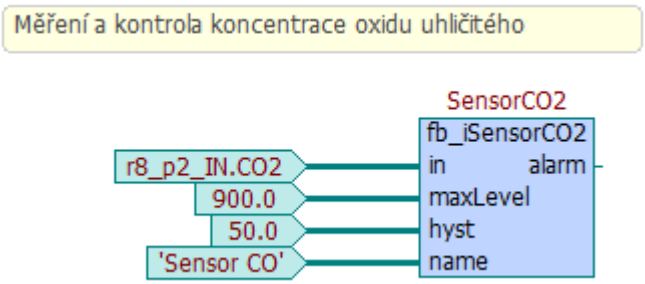
	Proměnná	Typ	Význam
VAR_INPUT			
	in	REAL	Vstup pro připojení hodnoty koncentrace oxidu uhličitého
	maxLevel	REAL	Prahová hodnota oxidu uhličitého Přednastavená hodnota je 900 ppm
	hyst	REAL	Hystereze pro nastavení výstupu *alarm* Přednastavená hodnota je 50 ppm
	name	STRING[48]	název
VAR_OUTPUT			
	alarm	BOOL	TRUE pokud je *in* > *maxLevel* FALSE pokud *in* < *maxLevel* - *hyst*

Předpokládejme, že potřebujeme měřit koncentraci CO2 a vyhodnocovat překročení povolené koncentrace (včetně signalizace překročení do mobilní aplikace). Pro měření bude použit modul C-AQ-001R. V jazyce ST bude program vypadat následovně:

```
PROGRAM prgMain
  VAR
    SensorCO2  : fb_iSensorCO2 := ( name := 'Sensor CO2');
  END_VAR

  SensorCO2(in := r8_p2_IN.CO2);
END_PROGRAM
```

Limitní hodnotou pro koncentraci CO2 je 900 ppm. Pokud měřená hodnota překročí 900 ppm, pak bude nastaven výstup *alarm*. Ten zůstane nastaven až do doby, kdy kon-

centrace CO2 klesne pod 850 ppm. Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iSensorCO2* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

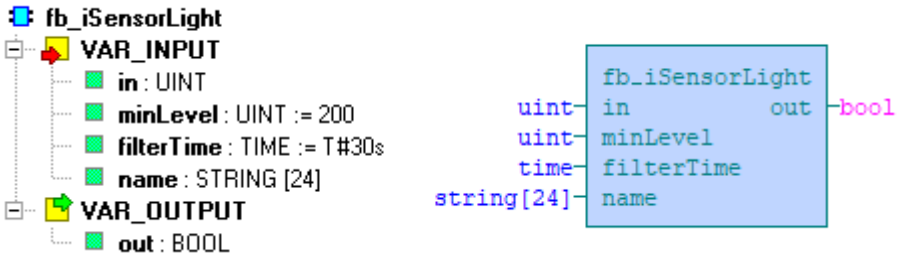
Každá instance *fb_iSensorCO2* přidá následující public proměnné:

	Proměnná	**Typ**	**Význam**
R	*GTSAP1_DISPLAY_name*	STRING[48]	Název bloku (kopie vstupu *name)*
R W	*GTSAP1_DISPLAY_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
R	*GTSAP1_DISPLAY_edit*	BOOL	Povolení editace (0 = editace zakázaná)
R	*GTSAP1_DISPLAY_alarm*	BOOL	Value > maxValue
R	*GTSAP1_DISPLAY_type*	USINT	Typ zobrazení (1 = REAL)
R	*GTSAP1_DISPLAY_symbol*	UINT	Aktuální kód symbolu (104)
R	*GTSAP1_DISPLAY_value*	REAL	Zobrazovaná koncentrace CO
R	*GTSAP1_DISPLAY_unit*	STRING[8]	Fyzikální jednotka (ppm)
R	*GTSAP1_DISPLAY_precision*	INT	Počet zobrazených desetinných míst (0)
R	*GTSAP1_DISPLAY_maxValue*	REAL	Max. povolená hodnota

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.22 Funkční blok fb_iSensorLight

Knihovna : *iControlLib*



Funkční blok *fb_iSensorLight* slouží k vyhodnocení intenzity osvětlení. Vstup *in* slouží k připojení měřené intenzity (například z modulu C-RI-0401S), která se nejprve filtruje ve filtru 1.řádu a poté se porovnává se vstupem *minLevel*. Vstup *filterTime* udává časovou konstantu filtru.

Pokud vstup *in* klesne pod hodnotu danou vstupem *minLevel*, pak je nastaven výstup *out.* Ten může být použit například pro ovládání venkovního osvětlení.

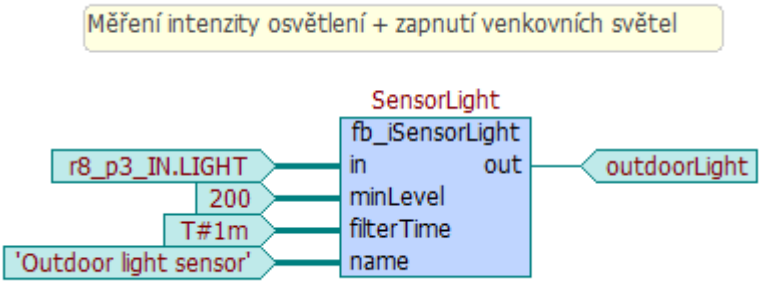
Popis proměnných :

	Proměnná	***Typ***	***Význam***
VAR_INPUT			
	in	UINT	Vstup pro měřenou hodnotu intenzity osvětení [lx]
	minLevel	UINT	Minimální úroveň osvětlení Přednastavená hodnota je 200 lx
	filterTime	REAL	Časová konstanta pro filtr 1.řádu, který je použit pro filtraci měřené hodnoty Přednastavená hodnota je 30 sec
	name	STRING[48]	název
VAR_OUTPUT			
	out	BOOL	TRUE pokud je *in* < *minLevel*, jinak FALSE

Předpokládejme, že budeme měřit intenzitu osvětlení a pokud bude menší než 200 lx tak zapneme venkovní osvětlení. Pro měření bude použit modul C-RI-0401S. V jazyce ST bude program vypadat následovně:

```
PROGRAM prgMain
  VAR
    SensorLight : fb_iSensorLight := ( name := 'Outdoor light sensor');
  END_VAR

  SensorLight(in := r8_p3_IN.LIGHT, out => outdoorLight);
END_PROGRAM
```

Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

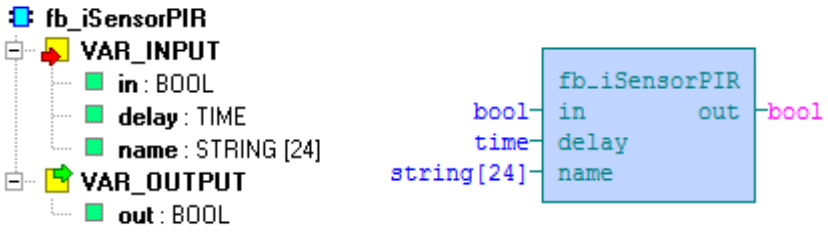
Pro každou použitou instanci funkčního bloku *fb_iSensorLight* se do souboru „iFox-trot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

Každá instance *fb_iSensorLight* přidá následující public proměnné:

	Proměnná	Typ	Význam
R	*GTSAP1_DISPLAY_name*	STRING[48]	Název bloku (kopie vstupu *name)*
R W	*GTSAP1_DISPLAY_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Fox-trot 2 zálohována.
R	*GTSAP1_DISPLAY_edit*	BOOL	Povolení editace (0 = editace zakázaná)
R	*GTSAP1_DISPLAY_type*	USINT	Typ zobrazení (1 = REAL)
R	*GTSAP1_DISPLAY_symbol*	UINT	Aktuální kód symbolu (107)
R	*GTSAP1_DISPLAY_value*	REAL	Zobrazovaná intenzita osvětlení
R	*GTSAP1_DISPLAY_unit*	STRING[8]	Fyzikální jednotka (lx)
R	*GTSAP1_DISPLAY_precision*	INT	Počet zobrazených desetinných míst (0)
R	*GTSAP1_DISPLAY_minValue*	REAL	Minimální hodnota

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.23 Funkční blok fb_iSensorPIR



Funkční blok *fb_iSensorPIR* slouží k připojení PIR čidla pro vyhodnocení přítomnosti osob, které se připojuje na vstup *in*. Vstup *delay* umožňuje nastavit minimální délku vstupního pulsu na kterou bude blok reagovat.

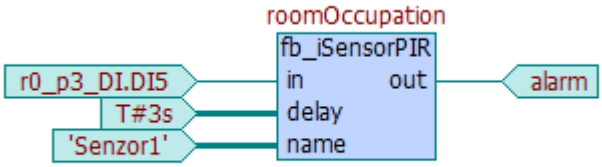
Popis proměnných :

	Proměnná	Typ	Význam
VAR_INPUT			
	in	BOOL	Vstup pro PIR čidlo (0 = detekuje osobu, 1 = bez detekce)
	delay	TIME	Necitlivost
	name	STRING[48]	Pojmenování bloku
VAR_OUTPUT			
	out	BOOL	Detekce pohybu

Jednoduchý příklad použití *fb_iSensorPIR* v jazyce ST.

```
PROGRAM prgMain
  VAR
    roomOccupation : fb_iSensorPIR;
    alarm          : BOOL;
  END_VAR

  roomOccupation (in     := r0_p3_DI.DI5,
                  delay  := T#3s  ,
                  name   := 'senzor1',
                  out    => alarm );
 END_PROGRAM
```

Pokud je čidlo aktivováno (0 na vstupu *in*) na déle jak 3 sekundy, tak funkční blok vyhodnotí přítomnost osoby a vyšle na výstup *out* 1. Stejný program lze realizovat v CFC následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iSensorPIR* se do souboru „iFox-trot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

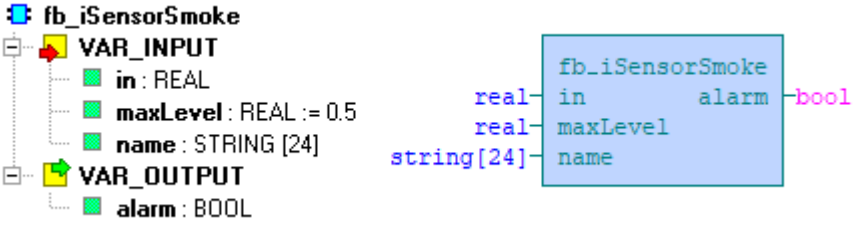
Každá instance *fb_iSensorPIR* přidá následující public proměnné:

	Proměnná	Typ	Význam
R	*GTSAP1_PIRSENSOR_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_PIRSENSOR_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
R	*GTSAP1_PIRSENSOR_value*	BOOL	Stav PIR čidla

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.24 Funkční blok fb_iSensorSmoke

Knihovna : *iControlLib*



Funkční blok *fb_iSensorSmoke* je určen k vyhodnocení naměřených hodnot kouře. Vstup *in* slouží k připojení měřené hodnoty kouře (například z modulu C-AQ-003R), která se porovnává se vstupem *maxLevel*.

Pokud vstup *in* překročí hodnotu danou vstupem *maxLevel*, pak je nastaven výstup *alarm*.

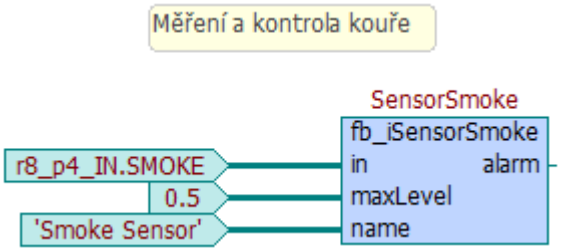
	Proměnná	*Typ*	*Význam*
VAR_INPUT			
	in	REAL	Vstup pro připojení naměřené hodnoty kouře
	maxLevel	REAL	Prahová hodnota kouře Přednastavená hodnota je 0,5 ppm
	name	STRING[48]	název
VAR_OUTPUT			
	alarm	BOOL	TRUE pokud je *in* > *maxLevel*, jinak FALSE

Předpokládejme, že potřebujeme měřit koncentraci kouře a vyhodnocovat překročení povolené koncentrace. Pro měření bude použit modul C-AQ-003R. V jazyce ST bude program vypadat následovně:

```
PROGRAM prgMain
  VAR
    SensorSmoke : fb_iSensorSmoke := ( name := 'Smoke sensor');
  END_VAR

  SensorSmoke(in := r8_p4_IN.SMOKE);
END_PROGRAM
```

Limitní hodnotou pro koncentraci kouře je 0.5 ppm. Pokud měřená hodnota překročí 0.5 ppm, pak bude nastaven výstup *alarm*.

Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iSensorSmoke* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

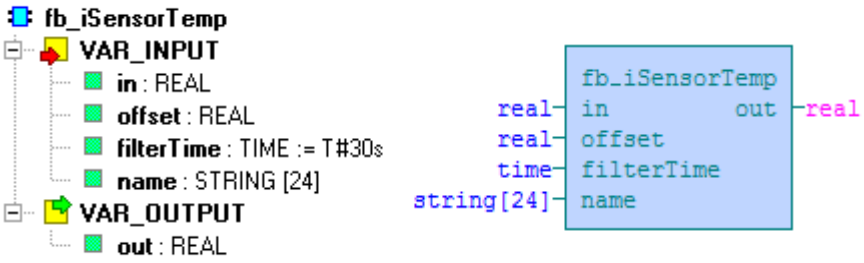
Každá instance *fb_iSensorSmoke* přidá následující public proměnné:

	Proměnná	**Typ**	**Význam**
R	*GTSAP1_DISPLAY_name*	STRING[48]	Název bloku (kopie vstupu *name)*
R W	*GTSAP1_DISPLAY_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
R	*GTSAP1_DISPLAY_edit*	BOOL	Povolení editace (0 = editace zakázaná)
R	*GTSAP1_DISPLAY_alarm*	BOOL	Value > maxValue
R	*GTSAP1_DISPLAY_type*	USINT	Typ zobrazení (1 = REAL)
R	*GTSAP1_DISPLAY_symbol*	UINT	Aktuální kód symbolu (106)
R	*GTSAP1_DISPLAY_value*	REAL	Zobrazovaná koncentrace kouře
R	*GTSAP1_DISPLAY_unit*	STRING[8]	Fyzikální jednotka (ppm)
R	*GTSAP1_DISPLAY_precision*	INT	Počet zobrazených desetinných míst (0)
R	*GTSAP1_DISPLAY_maxValue*	REAL	Max. povolená hodnota

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.25 Funkční blok fb_iSensorTemp

Knihovna : *iControlLib*



Funkční blok *fb_iSensorTemp* je určen k měření teploty. Měřenou teplotu lze korigovat o pevný offset a filtrovat filtrem 1.řádu. Tento blok nahrazuje blok *fb_iTherm*, který byl v předchozích verzích knihovny iControlLib.

Vstup *in* slouží k připojení teplotního čidla. Vstup *offset* umožňuje korigovat měřenou teplotu o zadanou hodnotu. Vstup *filterTime* udává časovou konstantu pro filtr 1.řádu, který filtruje měřenou teplotu. Pokud má vstup *filterTime* hodnotu T#0s tak je filtrace vypnutá. Na výstupu *out* je k dispozici teplota po filtraci korigovaná o zadaný offset.

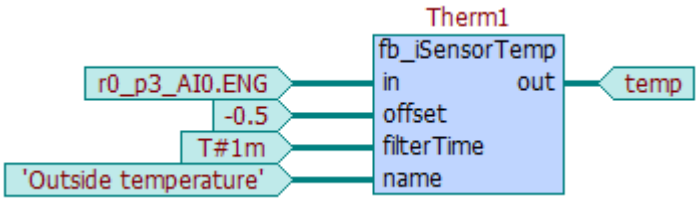
Popis proměnných :

	Proměnná	Typ	Význam
VAR_INPUT			
	in	REAL	Vstup pro připojení teplotního čidla
	offset	REAL	Korekce měřené teploty
	filterTime	TIME	Časová konstanta filtru pro měřenou teplotu Přednastavená hodnota je 30 sec
	name	STRING[48]	název
VAR_OUTPUT			
	out	REAL	Výsledná teplota

Předpokládejme, že potřebujeme měřit teplotu čidlem připojeným na analogový vstup AI0. V jazyce ST bude program vypadat následovně:

```
PROGRAM prgMain
  VAR
    Therm1     : fb_iSensorTemp;
    temp       : REAL;          // teplota
  END_VAR

  // mereni teploty
  Therm1( in := r0_p3_AI0.ENG, offset := -0.5, filterTime := T#1m, out => temp);
END_PROGRAM
```

Měřená teplota bude bude korigovaná o -0.5 °C a filtrovaná filtrem s časovou konstantou 1 min. Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iSensorTemp* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

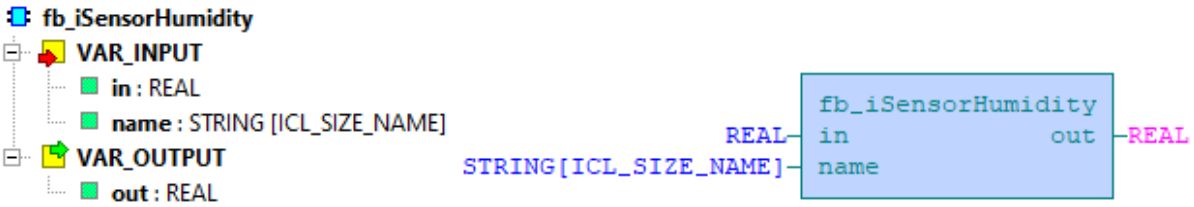
Každá instance *fb_iSensorTemp* přidá následující public proměnné:

	Proměnná	**Typ**	**Význam**
R	*GTSAP1_DISPLAY_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_DISPLAY_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
R	*GTSAP1_DISPLAY_edit*	BOOL	Povolení editace (0 = editace zakázaná)
R	*GTSAP1_DISPLAY_type*	USINT	Typ zobrazení (1 = REAL)
R	*GTSAP1_DISPLAY_symbol*	UINT	Aktuální kód symbolu (100)
R	*GTSAP1_DISPLAY_value*	REAL	Zobrazovaná teplota
R	*GTSAP1_DISPLAY_unit*	STRING[8]	Fyzikální jednotka (°C)
R	*GTSAP1_DISPLAY_precision*	INT	Počet zobrazených desetinných míst

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.26 Funkční blok fb_iSensorHumidity

Knihovna : *iControlLib*



Funkční blok *fb_iSensorHumidity* je určen k publikování měření vlhkosti do uživatelské aplikace.

Vstup *in* slouží k připojení čidla vlhkosti. Na výstupu *out* je k dispozici kopie vstupní měřené veličiny.

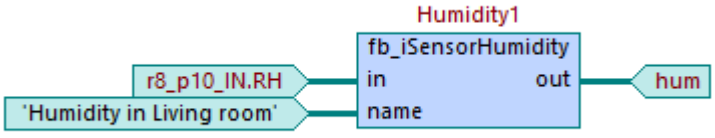
Popis proměnných :

	Proměnná	***Typ***	***Význam***
VAR_INPUT			
	in	REAL	Vstup pro připojení čidla vlhkosti
	name	STRING[48]	název
VAR_OUTPUT			
	out	REAL	Kopie vstupu *in*

Předpokládejme, že měříme relativní vlhkost modulem C-AQ-0004R. Na vstup funkčního bloku *in* připojíme měřenou hodnotu poskytovanou modulem přes vstup *RH.* V jazyce ST bude program vypadat následovně:

```
PROGRAM prgMain
  VAR
    Humidity1     : fb_iSensorHumidity;
    hum           : REAL; // vlhkost
  END_VAR

  // mereni vlhkosti
  Humidity1( in := r8_p10_IN.RH, out => hum);
END_PROGRAM
```

Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iSensorHumidity* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

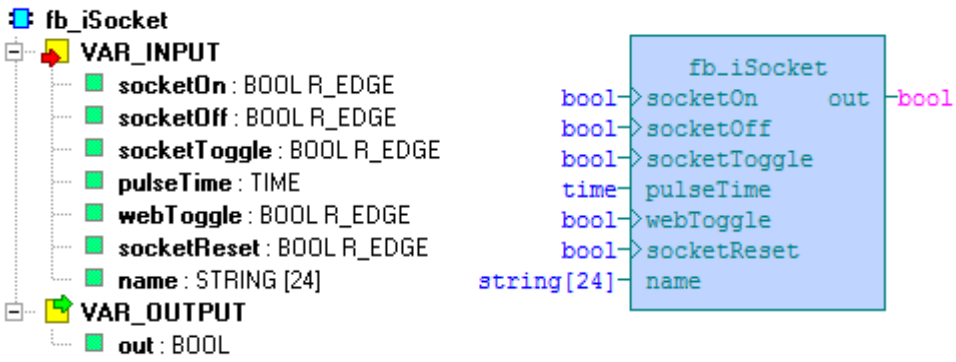
Každá instance *fb_ iSensorHumidity* přidá následující public proměnné:

	Proměnná	**Typ**	**Význam**
R	*GTSAP1_DISPLAY_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_DISPLAY_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
R	*GTSAP1_DISPLAY_edit*	BOOL	Povolení editace (0 = editace zakázaná)
R	*GTSAP1_DISPLAY_type*	USINT	Typ zobrazení (1 = REAL)
R	*GTSAP1_DISPLAY_symbol*	UINT	Aktuální kód symbolu (101)
R	*GTSAP1_DISPLAY_value*	REAL	Zobrazovaná vlhkost
R	*GTSAP1_DISPLAY_unit*	STRING[8]	Fyzikální jednotka (%)
R	*GTSAP1_DISPLAY_precision*	INT	Počet zobrazených desetinných míst

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.27 Funkční blok fb_iSocket

Knihovna : *iControlLib*



Funkční blok *fb_iSocket* je určen k řízení zásuvky.

Vstup *socketOn* spíná zásuvku, vstup *socketOff* zásuvku rozpíná. Vstup *socketToggle* přepíná stav zásuvky. Vstup *webToggle* funguje stejně jako *socketToggle* a slouží k ovládání zásuvky z web rozhraní (tuto proměnnou je třeba nastavit na TRUE v případě, že chceme přepnout zásuvku z web stránky). Vstup *pulseTime* určuje jak dlouho bude zásuvka sepnutá. Pokud má hodnotu T#0s tak doba sepnutí zásuvky není omezena. A konečně vstup *socketReset* slouží jako centrální rozepnutí všech zásuvek. Všechny vstupy typu BOOL reagují na náběžnou hranu vstupního signálu.

Výstup *out* slouží k ovládání zásuvky.

Popis proměnných :

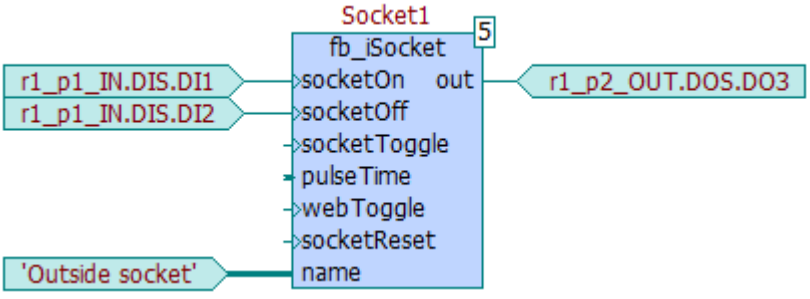
	Proměnná	***Typ***	***Význam***

VAR_INPUT			
	socketOn	BOOL R_EDGE	Náběžná hrana sepne zásuvku
	socketOff	BOOL R_EDGE	Náběžná hrana rozepne zásuvku
	socketToggle	BOOL R_EDGE	Náběžná hrana přepne zásuvku
	pulseTime	TIME	Doba výstupního pulzu Při T#0s není doba sepnutí zásuvky omezena
	webToggle	BOOL R_EDGE	Ovládání zásuvky z web stránky
	socketReset	BOOL R_EDGE	Vstup pro centrální rozepnutí zásuvek (odchodové tlačítko)
	name	STRING[48]	název
VAR_OUTPUT			
	out	BOOL	Ovládání zásuvky

Předpokládejme, že potřebujeme ovládat venkovní zásuvku připojenou na výstup DO3. Zásuvku zapneme sepnutím binárního vstupu DI1. K vypnutí zásuvky dojde při sepnutí vstupu DI2. V jazyce ST bude program vypadat následovně:

```
PROGRAM prgMain
  VAR
    Socket1 : fb_iSocket;
  END_VAR

  // ovladani zasuvky
  Socket1( socketOn  := r1_p1_IN.DIS.DI1,
           socketOff := r1_p1_IN.DIS.DI2,
           name      := 'Outside socket',
           out       => r1_p2_OUT.DOS.DO3);
END_PROGRAM
```

Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iSocket* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

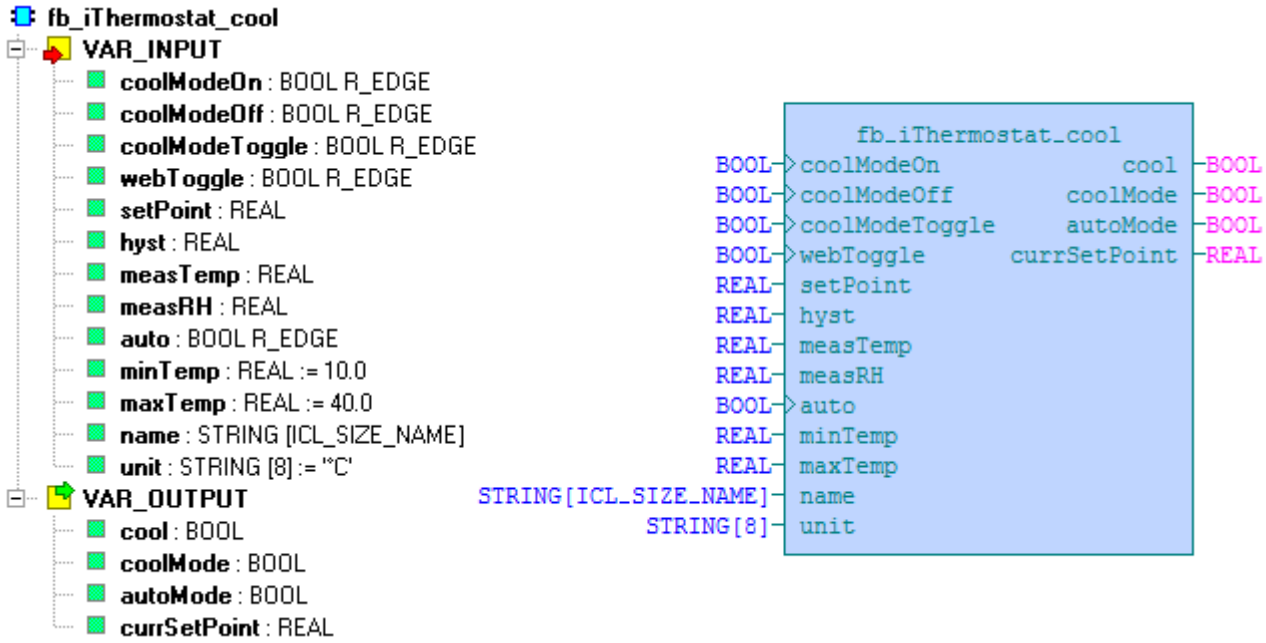
Každá instance *fb_iSocket* přidá následující public proměnné:

	Proměnná	Typ	Význam
R	*GTSAP1_SOCKET_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_SOCKET_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_SOCKET_needAck*	BOOL	Nastavením na TRUE bude pro změnu stavu bloku vyžadováno potvrzení. Změnit volbu lze pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_SOCKET_onoff*	BOOL	Vypínání a zapínání zásuvky z uživatelské aplikace.

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.28 Funkční blok fb_iThermostat_cool

Knihovna : *iControlLib*



Funkční blok *fb_iThermostat_cool* je určen k realizaci jednoduchého termostatu pro chlazení.

Vstup *coolModeOn* zapíná režim chlazení, vstup *coolModeOff* režim chlazení vypíná. Vstup *coolModeToggle* stav zapnutí režimu přepne. Vstup *webToggle* funguje stejně jako *coolModeToggle* a slouží k ovládání z web rozhraní. Proměnná *setPoint* udává cílovou teplotu pro chlazení, pokud termostat pracuje v *auto* režimu. V tomto režimu je pro řízení chlazení sledován vstup *setPoint*. Jakmile dojde k jakékoliv korekci žádané teploty pomocí uživatelské aplikace, přechází řízení do manuálního režimu a žádaná teplota pro chlazení je dána vnitřní proměnnou funkčního bloku. K opětovnému přechodu do *auto* režimu pak dochází až po nastavení vstupní proměnné *auto*. Vstup *hyst* definuje hysterézi spínání výstupu pro chlazení. Řízení je pak realizováno na základě měřené teploty dané vstupní proměnnou *measTemp*. Vstup *measRH* je určen pro připojení měřené relativní vlhkosti, která však nijak neovlivňuje funkcí řízení. Slouží pouze pro případ, kdy vizualizace termostatu v některé z externích aplikací umožňuje zobrazení této veličiny. Proměnné *minTemp* a *maxTemp* určují rozsah nastavitelných teplot v aplikaci. Pojmenování bloku pomocí proměnné *name* je určeno pro rozeznání bloku v uživatelské aplikaci. Vstup *unit* umožňuje specifikovat fyzikální jednotku měřené teploty.

Výstup *cool* je nastaven v případě požadavku na zapnutí chlazení. Proměnná *coolMode* signalizuje, že je chladicí režim aktivní. Proměnná *autoMode* pak značí, že funkční blok pracuje v *auto* režimu, tedy chlazení je řízeno dle teploty udávané vstupem *setPoint*. Výstupní proměnná *currSetPoint* reprezentuje aktuální cílovou teplotu pro chlazení. V případě aktivního *auto* režimu je hodnota této proměnné shodná se vstupem *setPoint*. V případě deaktivace *auto* režimu pak tato proměnná obsahuje cílovou teplotu po korekci.

Popis proměnných :

	Proměnná	***Typ***	***Význam***
VAR_INPUT			
	coolModeOn	BOOL	Náběžná hrana zapne chladicí režim

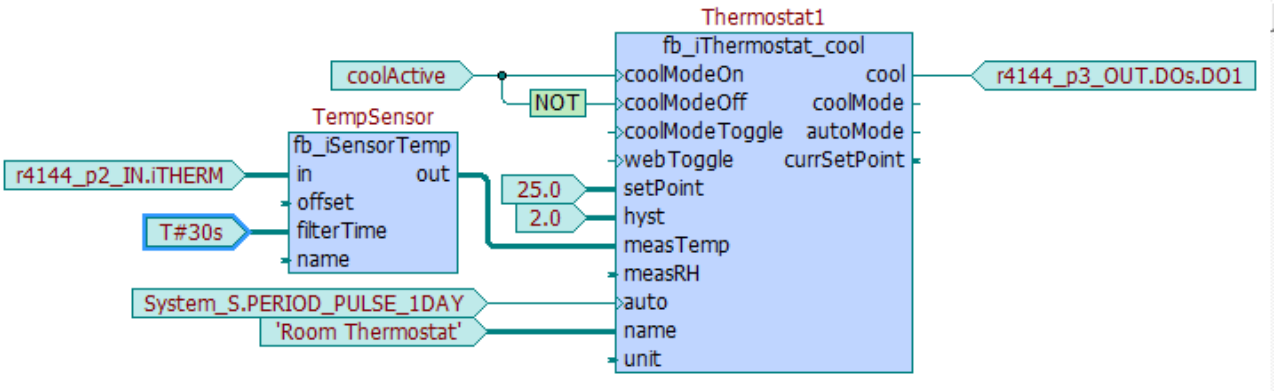
		R_EDGE	
	coolModeOff	BOOL R_EDGE	Náběžná hrana vypne chladicí režim
	coolModeToggle	BOOL R_EDGE	Náběžná hrana přepne chladicí režim
	webToggle	BOOL R_EDGE	Přepnutí chladícího režimu z web stránky
	setPoint	REAL	Požadovaná teplota pro auto režim
	hyst	REAL	Hystereze spínání výstupu *cool*
	measTemp	REAL	Měřená teplota
	measRH	REAL	Měřená vlhkost
	auto	BOOL R_EDGE	Náběžná hrana zapne auto režim
	minTemp	REAL	Minimální teplota k zobrazení
	maxTemp	REAL	Maximální teplota k zobrazení
	name	STRING[48]	název
	unit	STRING[8]	Jednotky pro zobrazení v aplikaci (výchozí °C)
VAR_OUTPUT			
	cool	BOOL	Požadavek na chlazení
	coolMode	BOOL	Signalizace zapnutého chladicího režimu
	autoMode	BOOL	Signalizace aktivního auto režimu
	currSetPoint	REAL	Aktuálně nastavená požadovaná teplota

Jednoduché použití funkčního bloku lze demonstrovat na následujícím příkladě. Předpokládejme, že chceme ovládat chlazení v místnosti pomocí aktoru připojeného přes reléový kontakt. Aktivace termostatu je řízena proměnnou *coolActive*. Žádaná teplota je pro daný příklad dána konstantou. V reálné situaci může být hodnota dána nějakou formou časového programu. V případě, že dojde ke korekci teploty pomocí uživatelské aplikace, je *auto* režim nastaven zpět vždy o půlnoci. Měřená teplota může být dána například hodnotou senzoru z jednotky *C-RC-0002R*. Před použitím je ale vhodné hodnotu filtrovat pomocí funkčního bloku *fb_iSensorTemp*. V jazyce ST bude program vypadat následovně:

```
PROGRAM prgMain
  VAR
    TempSensor    : fb_iSensorTemp;
    Thermostat1   : fb_iThermostat_cool;
    coolActive    : BOOL;
  END_VAR

  TempSensor(in := ROUND(r4144_p2_IN.iTHERM*10.0)/10.0,
             offset := 0.0,
             filterTime := T#30s);

  Thermostat1(coolModeOn := coolActive,
```

```
                coolModeOff := NOT coolActive,
                setPoint := 25.0,
                hyst := 2.0,
                measTemp := TempSensor.out,
                auto := NOT System_S.PERIOD_PULSE_1DAY,
                name := 'Room Thermostat',
                cool => r4144_p3_OUT.Dos.DO1);

END_PROGRAM
```

Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iThermostat_cool* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

Každá instance *fb_iThermostat_cool* přidá následující public proměnné:

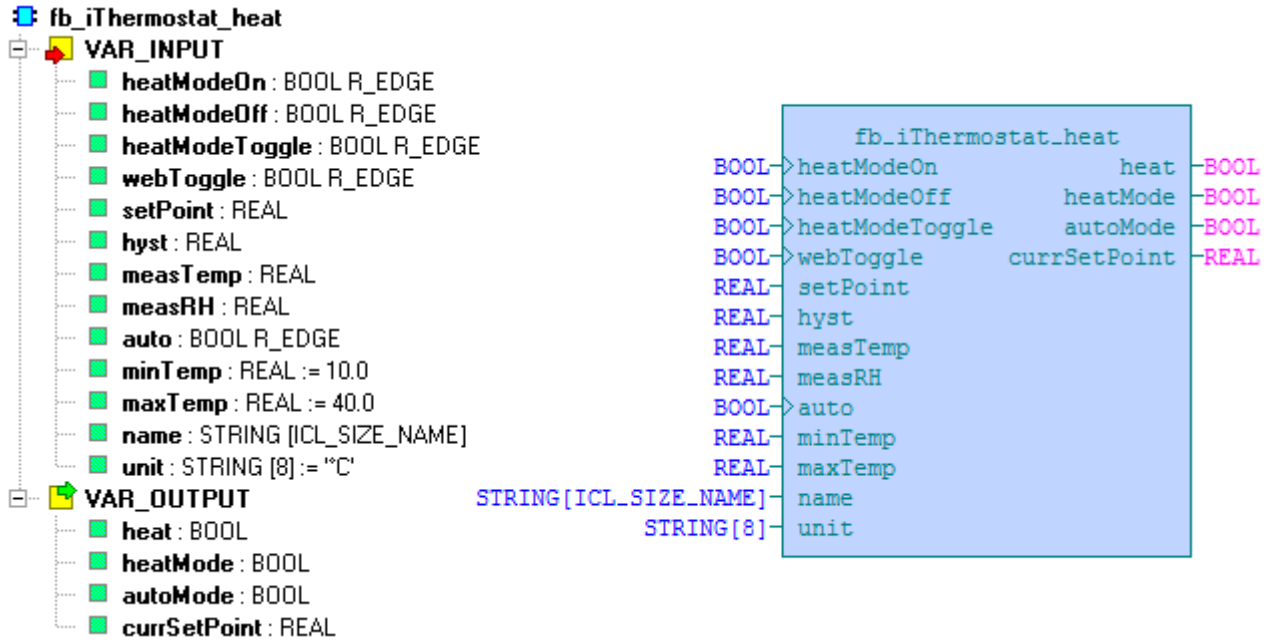
	Proměnná	Typ	Význam
R	*GTSAP1_THERMOSTAT_type*	USINT	Typ termostatu: **1 – termostat pro chlazení** 2 – termostat pro topení 3 – termostat pro chlazení/ topení
RW	*GTSAP1_THERMOSTAT_coolMode*	BOOL	ovládání a signalizace režimu chlazení
RW	*GTSAP1_THERMOSTAT_autoMode*	BOOL	ovládání a signalizace *auto* režimu
R	*GTSAP1_THERMOSTAT_cool*	BOOL	požadavek na chlazení
RW	*GTSAP1_THERMOSTAT_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_THERMOSTAT_needAck*	BOOL	Nastavením na TRUE bude pro změnu stavu bloku vyžadováno potvrzení. Změnit vol-

	Proměnná	Typ	Význam
			bu lze pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_THERMOSTAT_needPin*	STRING[10]	V případě neprázdného řetězce bude pro změnu stavu bloku vyžadován uvedený PIN. Změnu PINu lze provést pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_THERMOSTAT_setPoint*	REAL	nastavení cílové teploty pro chlazení
R	*GTSAP1_THERMOSTAT_hystCool*	REAL	nastavená hystereze spínání výstupu
R	*GTSAP1_THERMOSTAT_measTemp*	REAL	měřená teplota
R	*GTSAP1_THERMOSTAT_measRH*	REAL	měřená relativní vlhkost
R	*GTSAP1_THERMOSTAT_minTemp*	REAL	minimální teplota k zobrazení
R	*GTSAP1_THERMOSTAT_maxTemp*	REAL	maximální teplota k zobrazení
R	*GTSAP1_THERMOSTAT_name*	STRING[48]	Kopie vstupu *name*
R	*GTSAP1_THERMOSTAT_unit*	STRING[8]	fyzikální jednotky měřené teploty

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.29 Funkční blok fb_iThermostat_heat

Knihovna : *iControlLib*



Funkční blok *fb_iThermostat_heat* je určen k realizaci jednoduchého termostatu pro topení.

Vstup *heatModeOn* zapíná režim topení, vstup *heatModeOff* režim topení vypíná. Vstup *heatModeToggle* stav zapnutí režimu přepne. Vstup *webToggle* funguje stejně jako *heatModeToggle* a slouží k ovládání z web rozhraní. Proměnná *setPoint* udává cílovou teplotu pro topení, pokud termostat pracuje v *auto* režimu. V tomto režimu je pro řízení topení sledován vstup *setPoint*. Jakmile dojde k jakékoliv korekci žádané teploty pomocí uživatelské aplikace, přechází řízení do manuálního režimu a žádaná teplota pro topení je dána vnitřní proměnnou funkčního bloku. K opětovnému přechodu do *auto* režimu pak dochází až po nastavení vstupní proměnné *auto*. Vstup *hyst* definuje hysterézi spínání výstupu pro topení. Řízení je pak realizováno na základě měřené teploty dané vstupní proměnnou *measTemp*. Vstup *measRH* je určen pro připojení měřené relativní vlhkosti, která však nijak neovlivňuje funkcí řízení. Slouží pouze pro případ, kdy vizualizace termostatu v některé z externích aplikací umožňuje zobrazení této veličiny. Proměnné *minTemp* a *maxTemp* určují rozsah nastavitelných teplot v aplikaci. Pojmenování bloku pomocí proměnné *name* je určeno pro rozeznání bloku v uživatelské aplikaci. Vstup *unit* umožňuje specifikovat fyzikální jednotku měřené teploty.

Výstup *heat* je nastaven v případě požadavku na zapnutí topení. Proměnná *heatMode* signalizuje, že je režim topení aktivní. Proměnná *autoMode* pak značí, že funkční blok pracuje v *auto* režimu, tedy topení je řízeno dle teploty udávané vstupem *setPoint*. Výstupní proměnná *currSetPoint* reprezentuje aktuální cílovou teplotu pro topení. V případě aktivního *auto* režimu je hodnota této proměnné shodná se vstupem *setPoint*. V případě deaktivace *auto* režimu pak tato proměnná obsahuje cílovou teplotu po korekci.

Popis proměnných :

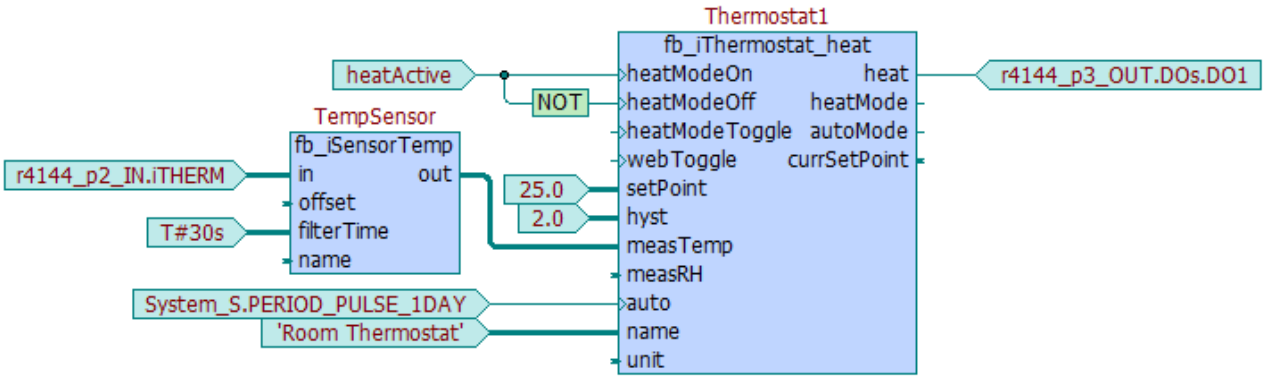
	Proměnná	Typ	Význam
VAR_INPUT			
	heatModeOn	BOOL R_EDGE	Náběžná hrana zapne topný režim
	heatModeOff	BOOL R_EDGE	Náběžná hrana vypne topný režim
	heatMode-Toggle	BOOL R_EDGE	Náběžná hrana přepne topný režim
	webToggle	BOOL R_EDGE	Přepnutí topného režimu z web stránky
	setPoint	REAL	Požadovaná teplota pro auto režim
	hyst	REAL	Hystereze spínání výstupu *heat*
	measTemp	REAL	Měřená teplota
	measRH	REAL	Měřená vlhkost
	auto	BOOL R_EDGE	Náběžná hrana zapne auto režim
	minTemp	REAL	Minimální teplota k zobrazení
	maxTemp	REAL	Maximální teplota k zobrazení
	name	STRING[48]	Název
	unit	STRING[8]	Jednotky pro zobrazení v aplikaci (výchozí °C)
VAR_OUTPUT			
	heat	BOOL	Požadavek na topení
	heatMode	BOOL	Signalizace zapnutého topného režimu
	autoMode	BOOL	Signalizace aktivního auto režimu
	currSetPoint	REAL	Aktuálně nastavená požadovaná teplota

Jednoduché použití funkčního bloku lze demonstrovat na následujícím příkladě. Předpokládejme, že chceme ovládat topení v místnosti pomocí aktoru připojeného přes reléový kontakt. Aktivace termostatu je řízena proměnnou *heatActive*. Žádaná teplota je pro daný příklad dána konstantou. V reálné situaci může být hodnota dána nějakou formou časového programu. V případě, že dojde ke korekci teploty pomocí uživatelské aplikace, je *auto* režim nastaven zpět vždy o půlnoci. Měřená teplota může být dána například hodnotou senzoru z jednotky *C-RC-0002R*. Před použitím je ale vhodné hodnotu filtrovat pomocí funkčního bloku *fb_iSensorTemp*. V jazyce ST bude program vypadat následovně:

```
PROGRAM prgMain
  VAR
    TempSensor    : fb_iSensorTemp;
    Thermostat1   : fb_iThermostat_heat;
    heatActive    : BOOL;
  END_VAR
```

```
  TempSensor(in := ROUND(r4144_p2_IN.iTHERM*10.0)/10.0,
             offset := 0.0,
             filterTime := T#30s);

  Thermostat1(heatModeOn := heatActive,
              heatModeOff := NOT heatActive,
              setPoint := 22.0,
              hyst := 1.0,
              measTemp := TempSensor.out,
              auto := NOT System_S.PERIOD_PULSE_1DAY,
              name := 'Room Thermostat',
              heat => r4144_p3_OUT.Dos.DO1);

END_PROGRAM
```

Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iThermostat_heat* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

Každá instance *fb_iThermostat_heat* přidá následující public proměnné:

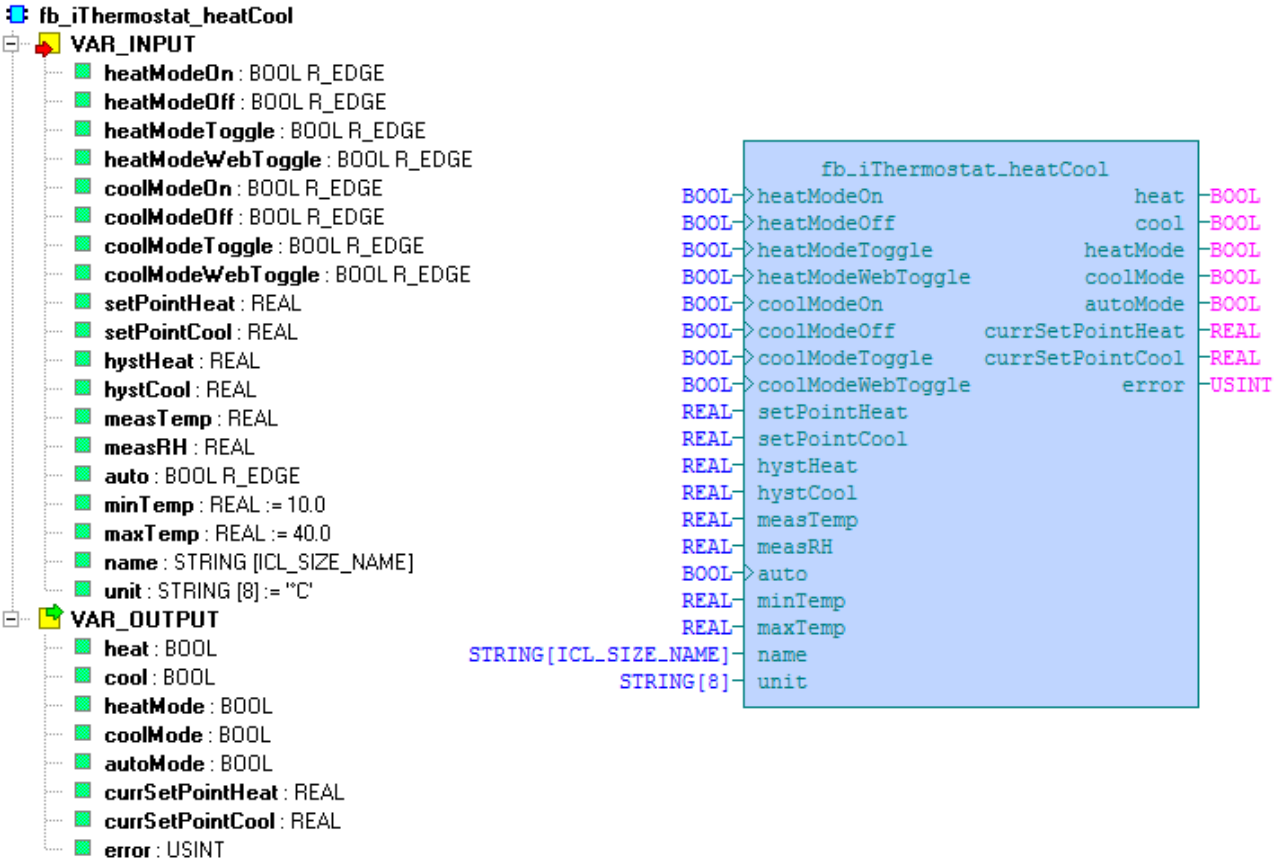
	Proměnná	Typ	Význam
R	*GTSAP1_THERMOSTAT_type*	USINT	Typ termostatu: 1 – termostat pro chlazení **2 – termostat pro topení** 3 – termostat pro chlazení/ topení
RW	*GTSAP1_THERMOSTAT_heatMode*	BOOL	ovládání a signalizace režimu topení
RW	*GTSAP1_THERMOSTAT_autoMode*	BOOL	ovládání a signalizace *auto* režimu
R	*GTSAP1_THERMOSTAT_heat*	BOOL	požadavek na topení
RW	*GTSAP1_THERMOSTAT_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v

			případě Foxtrot 2 zálohována.
RW	*GTSAP1_THERMOSTAT_needAck*	BOOL	Nastavením na TRUE bude pro změnu stavu bloku vyžadováno potvrzení. Změnit volbu lze pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_THERMOSTAT_needPin*	STRING[10]	V případě neprázdného řetězce bude pro změnu stavu bloku vyžadován uvedený PIN. Změnu PINu lze provést pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_THERMOSTAT_setPoint*	REAL	nastavení cílové teploty pro topení
R	*GTSAP1_THERMOSTAT_hystHeat*	REAL	nastavená hystereze spínání výstupu
R	*GTSAP1_THERMOSTAT_measTemp*	REAL	měřená teplota
R	*GTSAP1_THERMOSTAT_measRH*	REAL	měřená relativní vlhkost
R	*GTSAP1_THERMOSTAT_minTemp*	REAL	minimální teplota k zobrazení
R	*GTSAP1_THERMOSTAT_maxTemp*	REAL	maximální teplota k zobrazení
R	*GTSAP1_THERMOSTAT_name*	STRING[48]	Kopie vstupu *name*
R	*GTSAP1_THERMOSTAT_unit*	STRING[8]	fyzikální jednotky měřené teploty

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.30 Funkční blok fb_iThermostat_heatCool

Knihovna : *iControlLib*



Funkční blok *fb_iThermostat_heatCool* je určen k realizaci jednoduchého termostatu pro chlazení a topení. Jedná se kombinaci bloků *fb_iThermostat_heat* a *fb_iThermostat_cool*.

Vstup *coolModeOn/heatModeOn* zapíná režim chlazení/topení, vstup *coolModeOff/heatModeOff* režim chlazení/topení vypíná. Vstup *coolModeToggle/heatModeToggle* stav zapnutí režimu přepne. Vstup coolModeW*ebToggle/heatModeWebToggle* funguje stejně jako *coolModeToggle/heatModeToggle* a slouží k ovládání z web rozhraní. Proměnná *setPointCool/setPointHeat* udává cílovou teplotu pro chlazení/topení, pokud termostat pracuje v *auto* režimu. V tomto režimu je pro řízení chlazení/topení sledován vstup *setPointCool/setPointHeat*. Jakmile dojde k jakékoliv korekci žádané teploty pomocí uživatelské aplikace, přechází řízení do manuálního režimu a žádaná teplota pro chlazení/topení je dána vnitřní proměnnou funkčního bloku. K opětovnému přechodu do *auto* režimu pak dochází až po nastavení vstupní proměnné *auto*. Vstup *hystCool/hystHeat* definuje hysterézi spínání výstupu pro chlazení/topení. Řízení je pak realizováno na základě měřené teploty dané vstupní proměnnou *measTemp*. Vstup *measRH* je určen pro připojení měřené relativní vlhkosti, která však nijak neovlivňuje funkcí řízení. Slouží pouze pro případ, kdy vizualizace termostatu v některé z externích aplikací umožňuje zobrazení této veličiny. Proměnné *minTemp* a *maxTemp* určují rozsah nastavitelných teplot v aplikaci. Pojmenování bloku pomocí proměnné *name* je určeno pro rozeznání bloku v uživatelské aplikaci. Vstup *unit* umožňuje specifikovat fyzikální jednotku měřené teploty.

Výstup *cool/heat* je nastaven v případě požadavku na zapnutí chlazení/topení. Proměnná *coolMode/heatMode* signalizuje, že je režim chlazení/topení aktivní. Proměnná *autoMode* pak značí, že funkční blok pracuje v *auto* režimu, tedy chlazení/topení je řízeno

dle teploty udávané vstupem *setPointCool/setPointHeat*. Výstupní proměnná *currSetPointCool/currSetPointHeat* reprezentuje aktuální cílovou teplotu pro chlazení/topení. V případě aktivního *auto* režimu je hodnota této proměnné shodná se vstupem *setPointCool/setPointHeat*. V případě deaktivace *auto* režimu pak tato proměnná obsahuje cílovou teplotu po korekci. Výstup *error* slouží pro hlášení chyb nebo varování funkčního bloku:

Hodnota error	***Význam***
0	Žádná chyba
1	Chybné nastavení parametrů. Dané nastavení požadovaných teplot pro chlazení/topení a jejich hysterezí může způsobit současné sepnutí výstupů *heat* a *cool*

Popis proměnných :

	Proměnná	***Typ***	***Význam***
VAR_INPUT			
	heatModeOn	BOOL R_EDGE	Náběžná hrana zapne topný režim
	heatModeOff	BOOL R_EDGE	Náběžná hrana vypne topný režim
	heatModeToggle	BOOL R_EDGE	Náběžná hrana přepne topný režim
	heatModeWebToggle	BOOL R_EDGE	Přepnutí topného režimu z web stránky
	coolModeOn	BOOL R_EDGE	Náběžná hrana zapne chladicí režim
	coolModeOff	BOOL R_EDGE	Náběžná hrana vypne chladicí režim
	coolModeToggle	BOOL R_EDGE	Náběžná hrana přepne chladicí režim
	coolModeWebToggle	BOOL R_EDGE	Přepnutí chladicí režimu z web stránky
	setPointHeat	REAL	Požadovaná teplota pro topení v auto režimu
	setPointCool	REAL	Požadovaná teplota pro chlazení v auto režimu
	hystHeat	REAL	Hystereze spínání výstupu heat
	hystCool	REAL	Hystereze spínání výstupu cool
	measTemp	REAL	Měřená teplota
	measRH	REAL	Měřená vlhkost
	auto	BOOL R_EDGE	Náběžná hrana zapne auto režim
	minTemp	REAL	Minimální teplota k zobrazení
	maxTemp	REAL	Maximální teplota k zobrazení

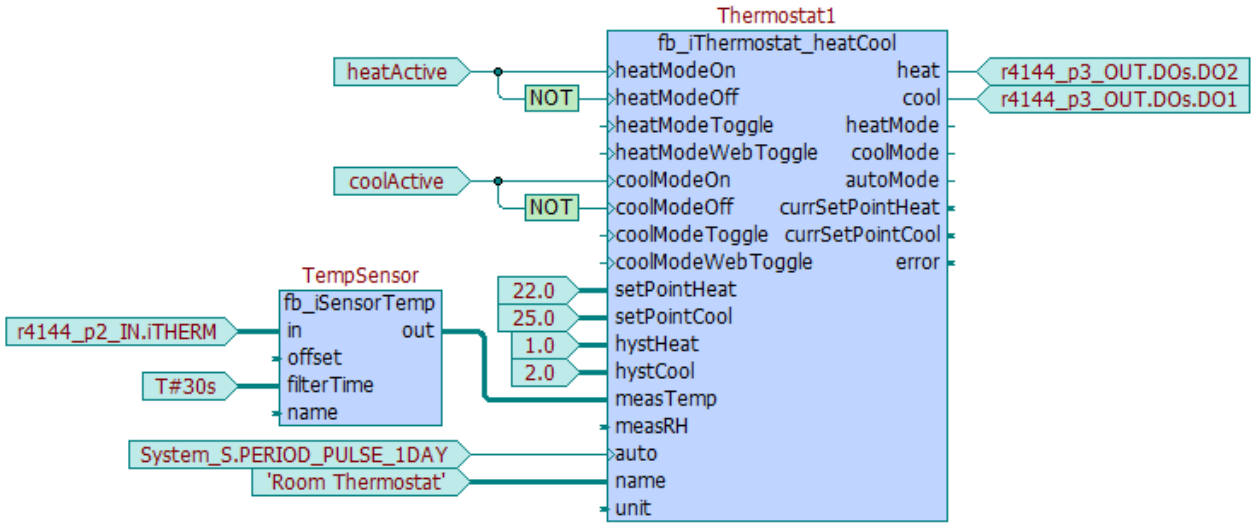
	Proměnná	Typ	Význam
	name	STRING[48]	Název
	unit	STRING[8]	Jednotky pro zobrazení v aplikaci
VAR_OUTPUT			
	heat	BOOL	Požadavek na topení
	cool	BOOL	Požadavek na chlazení
	heatMode	BOOL	Signalizace zapnutého topného režimu
	coolMode	BOOL	Signalizace zapnutého chladicího režimu
	autoMode	BOOL	Signalizace aktivního auto režimu
	currSetPointHeat	REAL	Aktuálně nastavená požadovaná teplota pro topení
	currSetPointCool	REAL	Aktuálně nastavená požadovaná teplota pro chlazení
	error	USINT	Hlášení chyby

Jednoduché použití funkčního bloku lze demonstrovat na následujícím příkladě. Předpokládejme, že chceme ovládat chlazení a topení v místnosti pomocí aktorů připojených přes reléové kontakty. Aktivace chladicího/topného režimu je řízena proměnnou *coolActive/heatActive*. Žádané teploty pro chlazení a topení jsou pro daný příklad dány konstantami. V reálné situaci můžou být hodnoty dány nějakou formou časového programu. V případě, že dojde ke korekci teplot pomocí uživatelské aplikace, je *auto* režim nastaven zpět vždy o půlnoci. Měřená teplota může být dána například hodnotou senzoru z jednotky *C-RC-0002R*. Před použitím je ale vhodné hodnotu filtrovat pomocí funkčního bloku *fb_iSensorTemp*. V jazyce ST bude program vypadat následovně:

```
PROGRAM prgMain
  VAR
    TempSensor    : fb_iSensorTemp;
    Thermostat1   : fb_iThermostat_heatCool;
    coolActive    : BOOL;
    heatActive    : BOOL;
  END_VAR

  TempSensor(in := ROUND(r4144_p2_IN.iTHERM*10.0)/10.0,
             offset := 0.0,
             filterTime := T#30s);

  Thermostat1(coolModeOn := coolActive,
              coolModeOff := NOT coolActive,
              heatModeOn := heatActive,
              heatModeOff := NOT heatActive,
              setPointCool := 25.0,
              setPointHeat := 22.0,
              hystCool := 2.0,
              hystHeat := 1.0,
              measTemp := TempSensor.out,
              auto := NOT System_S.PERIOD_PULSE_1DAY,
              name := 'Room Thermostat',
              cool => r4144_p3_OUT.Dos.DO1,
```

```
                heat => r4144_p3_OUT.Dos.DO2);

END_PROGRAM
```

Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iThermostat_heatCool* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

Každá instance *fb_iThermostat_heatCool* přidá následující public proměnné:

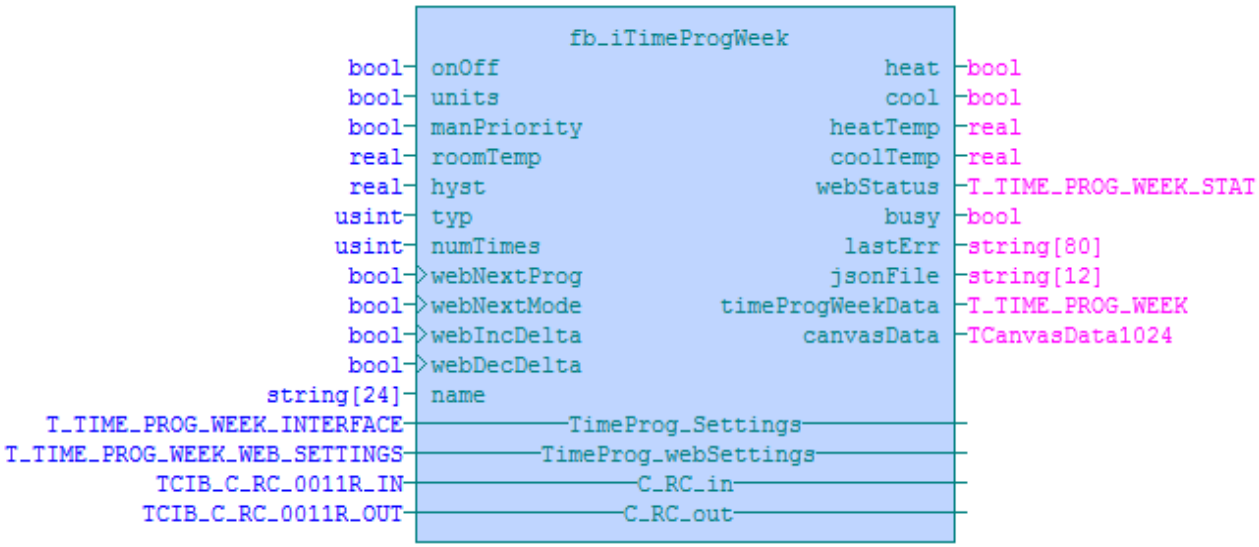
	Proměnná	**Typ**	**Význam**
R	*GTSAP1_THERMOSTAT_type*	USINT	Typ termostatu: 1 – termostat pro chlazení 2 – termostat pro topení **3 – termostat pro chlazení/ topení**
RW	*GTSAP1_THERMOSTAT_heatMode*	BOOL	ovládání a signalizace režimu topení
RW	*GTSAP1_THERMOSTAT_coolMode*	BOOL	ovládání a signalizace režimu chlazení
RW	*GTSAP1_THERMOSTAT_autoMode*	BOOL	ovládání a signalizace *auto* režimu
R	*GTSAP1_THERMOSTAT_heat*	BOOL	požadavek na topení
R	*GTSAP1_THERMOSTAT_cool*	BOOL	požadavek na chlazení
RW	*GTSAP1_THERMOSTAT_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot 2 zálohována.

	Proměnná	Typ	Význam
RW	*GTSAP1_THERMOSTAT_needAck*	BOOL	Nastavením na TRUE bude pro změnu stavu bloku vyžadováno potvrzení. Změnit volbu lze pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_THERMOSTAT_needPin*	STRING[10]	V případě neprázdného řetězce bude pro změnu stavu bloku vyžadován uvedený PIN. Změnu PINu lze provést pouze pomocí uživatelské aplikace. Proměnná je v případě Foxtrot 2 zálohována.
RW	*GTSAP1_THERMOSTAT_setPoint*	REAL	nastavení cílové teploty pro pouze jeden z aktivních režimů chlazení/topení
RW	*GTSAP1_THERMOSTAT_setPointHeat*	REAL	nastavení cílové teploty pro topení v případě obou aktivních režimů
RW	*GTSAP1_THERMOSTAT_setPointCool*	REAL	nastavení cílové teploty pro chlazení v případě obou aktivních režimů
R	*GTSAP1_THERMOSTAT_hystHeat*	REAL	nastavená hystereze spínání výstupu topení
R	*GTSAP1_THERMOSTAT_hystCool*	REAL	nastavená hystereze spínání výstupu chalzení
R	*GTSAP1_THERMOSTAT_measTemp*	REAL	měřená teplota
R	*GTSAP1_THERMOSTAT_measRH*	REAL	měřená relativní vlhkost
R	*GTSAP1_THERMOSTAT_minTemp*	REAL	minimální teplota k zobrazení
R	*GTSAP1_THERMOSTAT_maxTemp*	REAL	maximální teplota k zobrazení
R	*GTSAP1_THERMOSTAT_error*	USINT	hlášení chyby
R	*GTSAP1_THERMOSTAT_name*	STRING[48]	Kopie vstupu *name*
R	*GTSAP1_THERMOSTAT_unit*	STRING[8]	fyzikální jednotky měřené teploty

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.31 Funkční blok fb_iTimeProgWeek

Knihovna : *iControlLib*



Funkční blok *fb_iTimeProgWeek* je určen k řízení topení a chlazení podle týdenního časového programu. Základní funkci lze přirovnat k pokojovému termostatu.

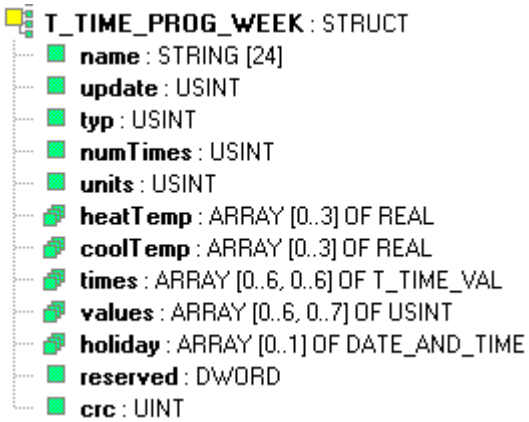
Blok zajišťuje následující činnosti:

- dlouhodobé řízení podle časového programu (auto)
- krátkodobé korekce požadovaného řízení (ručně)
- správu časového programu (ukládání a načítání časového programu na SD kartu)
- editaci časového programu z web rozhraní
- synchronizaci časového programu s modulem C-RC-0011R
- spolupráci s uživatelskou aplikací s danou podporou

Týdenní časový program

Týdenní časový program definuje nastavení vytápění resp. chlazení pro celý týden. Každý den může být libovolně rozdělen až do 8 intervalů (tj. 7 časových změn) v rastru 15 minut. Každému intervalu je přiřazen teplotní režim: komfort, útlum, ekonomy a ochranný. Požadovaná teplota v každém režimu je volitelná.

Týdenní časový program má následující strukturu:



Význam jednotlivých položek je následující:

- *name* název časového programu
- *update* požadavek na update časového programu
- *typ* určuje, jestli se podle programu bude topit nebo chladit nebo obojí

- *numTimes* počet časových značek za den (minimálně 2, maximálně 7)
- *units* jednotky teploty (°C, °F)
- *heatTemp[ ]* pole teplot topení pro program limit, ekonomy, útlum a komfort
- *coolTemp[ ]* pole teplot chlazení pro program limit, ekonomy, útlum a komfort
- *times[ ]* časové značky pro všechny dny v týdnu
- *values[ ]* požadovaný program pro každý zadaný interval
- *holiday[ ]* datum dovolené (od, do)
- *reserved* rezerva
- *crc* zabezpečení dat

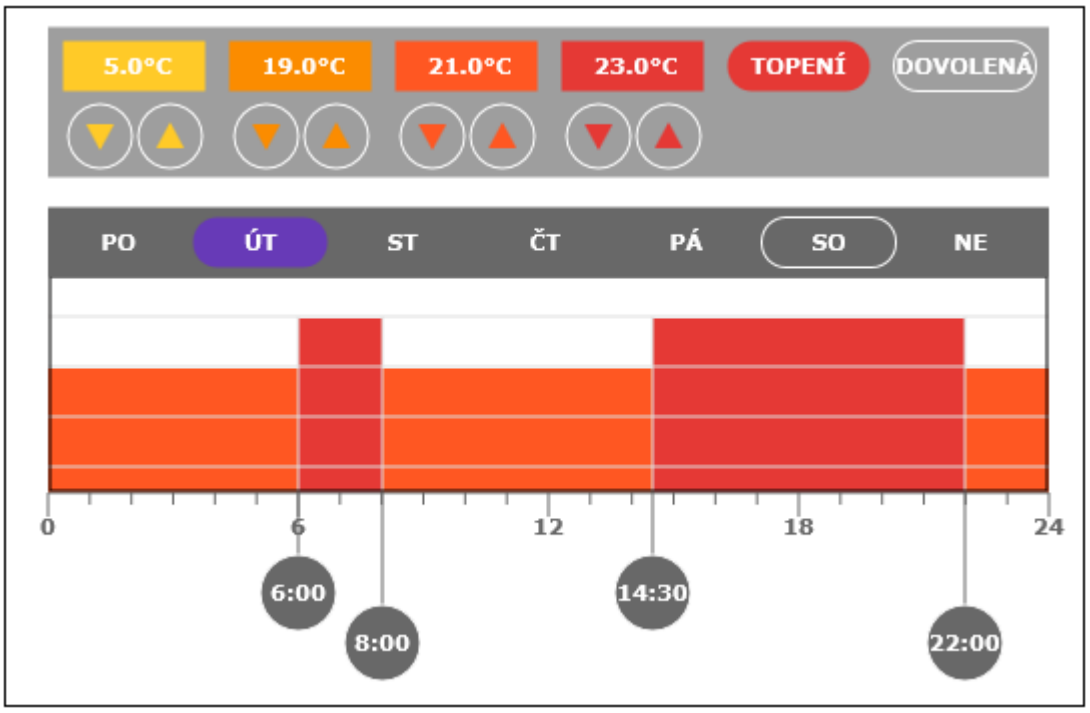
Časový program je při každé změně ukládán na SD kartu. Data v souboru jsou ve formátu JSON, název souboru je uveden na výstupu *jsonFile*. Soubor je uložen v adresáři WWW/iFOX/.

```
{
  "Name": "Program topeni",
  "timeProgWeekData":{
    "typ":3,
    "numTimes":6,
    "units":0,
    "heatTemp":[5.0,19.0,21.0,23.0],
    "coolTemp":[33.0,29.0,25.0,24.0],
    "times":[
      ["00:00:0","06:00:0","08:00:0","15:00:0","21:00:0","24:00:0","24:00:0"],
      ["01:00:0","06:15:0","08:00:0","15:00:0","21:00:0","24:00:0","24:00:0"],
      ["02:00:0","06:30:0","08:00:0","15:00:0","19:00:0","23:00:0","24:00:0"],
      ["03:00:0","06:45:0","09:00:0","13:45:0","21:00:0","24:00:0","24:00:0"],
      ["04:00:0","07:00:0","08:00:0","15:00:0","21:00:0","24:00:0","24:00:0"],
      ["05:00:0","07:15:0","22:00:0","24:00:0","24:00:0","24:00:0","24:00:0"],
      ["00:00:0","07:15:0","22:00:0","24:00:0","24:00:0","24:00:0","24:00:0"]
    ],
    "values":[
      [1,2,3,2,3,1,1,1],
      [1,2,3,2,3,1,1,1],
      [1,2,3,2,3,2,1,1],
      [1,2,3,2,3,1,1,1],
      [1,2,3,2,3,1,1,1],
      [1,2,3,2,1,1,1,1],
      [1,2,3,2,1,1,1,1]
    ],
    "holiday":["1970-01-01-00:00:0","1970-01-01-00:00:0"]
  }
}
```

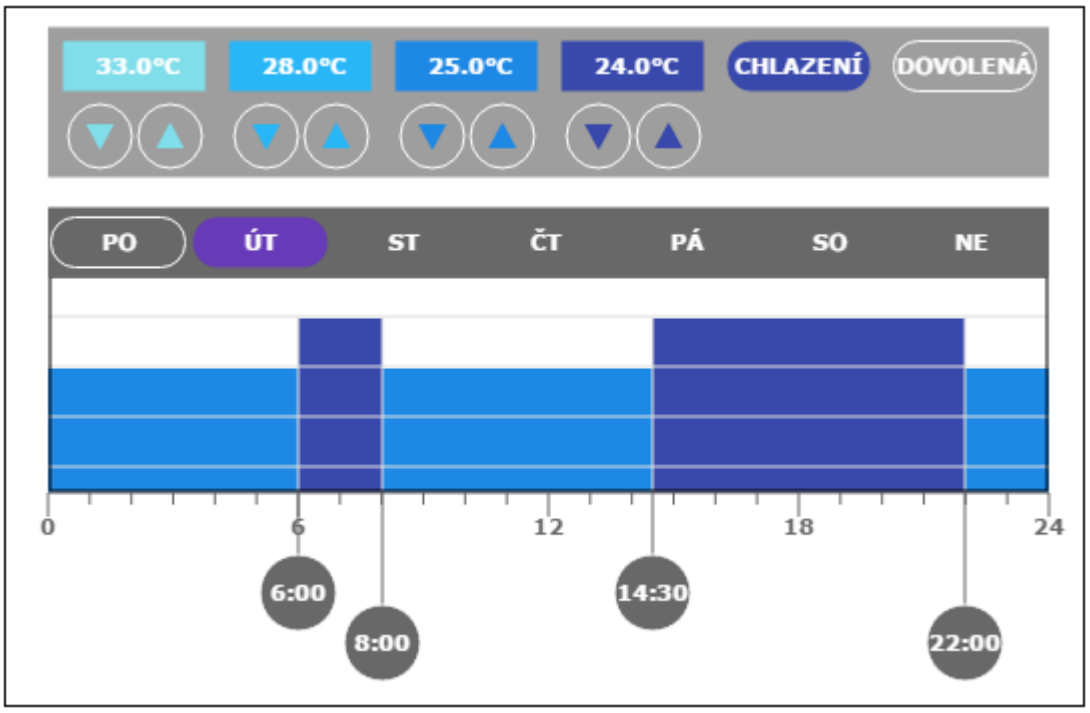
Editace časového programu

Editaci časového programu je možné provádět přes web rozhraní (z web stránky PLC vytvořené nástrojem WebMaker) nebo pomocí modulu C-RC-0011R nebo pomocí uživatelské aplikace.

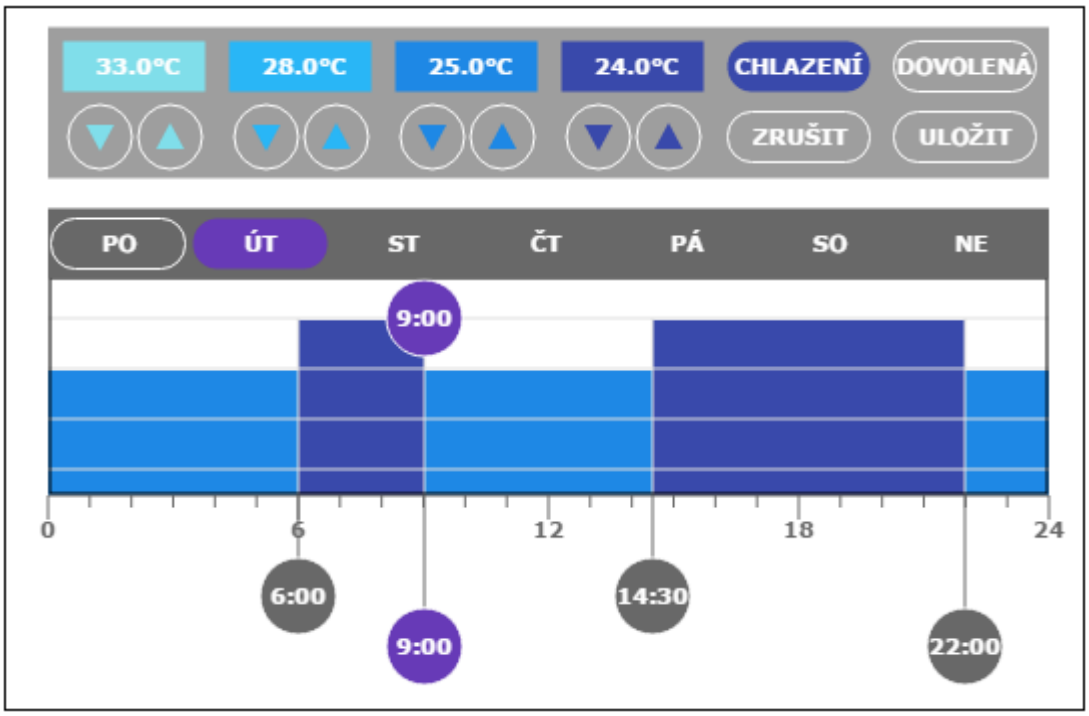
Pro editaci časového programu z web stránky nabízí funkční blok *fb_iTime-ProgWeek* data pro prvek „kreslící plátno“ (canvas). V nástroji WebMaker tedy stačí vložit do web stránky prvek „kreslící plátno“ a ve vlastnostech „kreslícího plátna“ nastavit výstup *canvasData* jako řídící strukturu. Dále je třeba zatrhnout pole „vracet pozici“. Výsledek bude ve web stránce vypadat např. následovně:



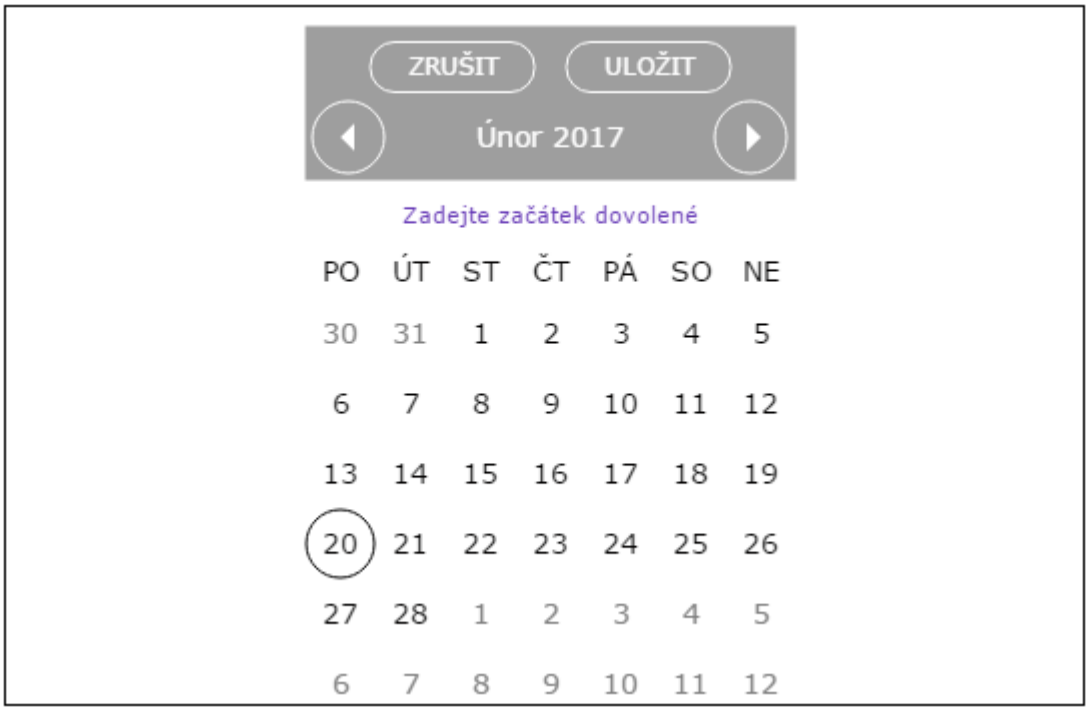
V horní části je možno měnit teploty přiřazené k jednotlivým režimům (zleva ochrana, ekonomy, útlum a komfort). Spodní část definuje žádanou teplotu pro topení na jeden den. Vybraný den je podbarven fialově, aktuální den v týdnu má bílé orámování. V uvedeném příkladu je zobrazen požadovaný průběh pro topení v úterý, kde je zvolen režim komfort (žádaná teplota 23°C) od 06:00 do 08:00 a od 14:30 do 22:00, v ostatních časech je zvolen režim útlum (žádaná teplota 21°C). Kliknutím na tlačítko „TOPENÍ“ lze přepnout zobrazení na „CHLAZENÍ“.



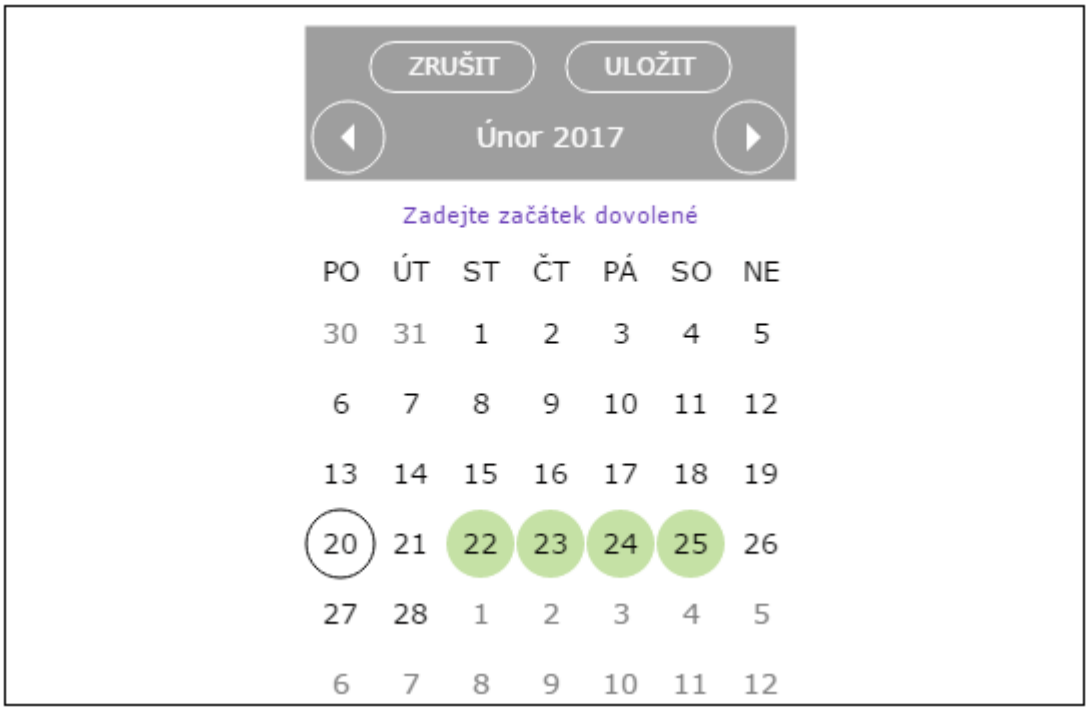
Časové značky ve spodní části lze přesunovat pomocí myši. Při přesouvání změní značka barvu a časový údaj se objeví na obou koncích čáry definující časový interval. Kliknutím do sloupce v intervalu lze měnit požadovaný režim. Pokud je provedena nějaká změna v časovém programu tak se v dialogu objeví tlačítka „ZRUŠIT“ a „ULOŽIT“. Kliknutím na tlačítko „ZRUŠIT“ se vrátíme k předešlému stavu (změna se zapomene). Kliknutím na tlačítko „ULOŽIT“ se změna akceptuje a uloží se do časového programu.



Počet zobrazených intervalů lze ovlivnit vstupem *numTimes*. Tlačítko „DOVOLENÁ“ zobrazí kalendář, ve kterém je možno definovat dovolenou.



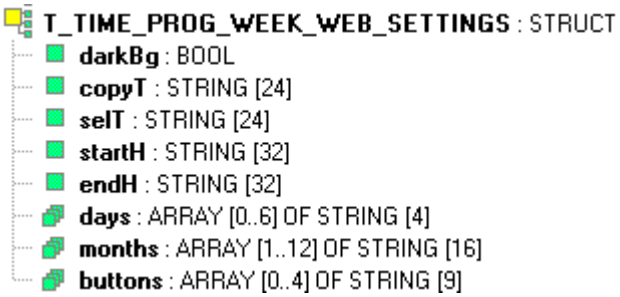
Prvním kliknutím do kalendáře lze zvolit začátek dovolené, druhým kliknutím se označí její konec. Vybrané dny dovolené jsou v kalendáři podbarvené zeleně. A stejně jako u změn časového programu je třeba nastavenou dovolenou uložit pomocí tlačítka „ULOŽIT“. Tlačítkem „ZRUŠIT“ se obnoví předešlý stav. Obě tlačítka zavřou dialog pro nastavení dovolené a zobrazí časový program.



Aktuální datum je v kalendáři označeno kroužkem. Pokud chceme vymazat zadanou dovolenou, pak stačí 3x kliknout na libovolný den v kalendáři.

Lokalizace textů pro web rozhraní

Všechny texty použité ve web rozhraní jsou dány vstupem *TimeProg_webSettings*. Ten vyžaduje proměnnou typu *T_TIME_PROG_WEEK_WEB_SETTINGS*, která musí obsahovat všechny texty použité ve web rozhraní a má následující strukturu:



V knihovně jsou od tohoto typu odvozeny dva další typy:

- *T_TIME_PROG_WEEK_WEB_SETTINGS_CZE* určený pro češtinu
- *T_TIME_PROG_WEEK_WEB_SETTINGS_ENG* určený pro angličtinu

Z definice *T_TIME_PROG_WEEK_WEB_SETTINGS_ENG* pro angličtinu lze snadnou odvodit typ pro libovolný další jazyk.

```
TYPE
  T_TIME_PROG_WEEK_WEB_SETTINGS_ENG  : T_TIME_PROG_WEEK_WEB_SETTINGS := (
    darkBg  := FALSE,
    copyT   := 'Copy ',
    selT    := 'Select dest. day',
    startH  := 'Set start of holiday',
    endH    := 'Set end of holiday',
    days    := ['MO','TU','WE','TH','FR','SA','SU'],
    months  := ['January','February','March','April','May','June',
                'July','August','September','October','November','December'],
    buttons := ['HEATING','COOLING','HOLIDAY','CANCEL','SAVE']);
END_TYPE
```

Takže například pro slovenštinu lze nadefinovat texty následovně:

```
TYPE
  T_TIME_PROG_WEEK_WEB_SETTINGS_SVK  : T_TIME_PROG_WEEK_WEB_SETTINGS := (
    darkBg  := true,
    copyT   := 'Kopírovať ',
    selT    := 'Zvoľte cieľový deň',
    startH  := 'Zadajte začiatok dovolenky',
    endH    := 'Zadajte koniec dovolenky',
    days    := ['PO','UT','ST','ŠT','PT','SO','NE'],
    months  := ['Január','Február','Marec','Apríl','Máj','Jún',
                'Júl','August','September','Október','November','December'],
    buttons := ['KÚRENIE','CHLADENIE','DOVOLENÁ','ZRUŠIŤ','ULOŽIŤ']);
END_TYPE
```

Položka *darkBg* definuje na jakém pozadí bude prvek „kreslící plátno“ zobrazen. Hodnota FALSE je pro světlá pozadí (časová osa bude mít popis tmavými písmeny, ...), hodnota TRUE naopak pro tmavá pozadí (časová osa bude mít popis světlými písmeny, atd.).

Řízení podle časového programu

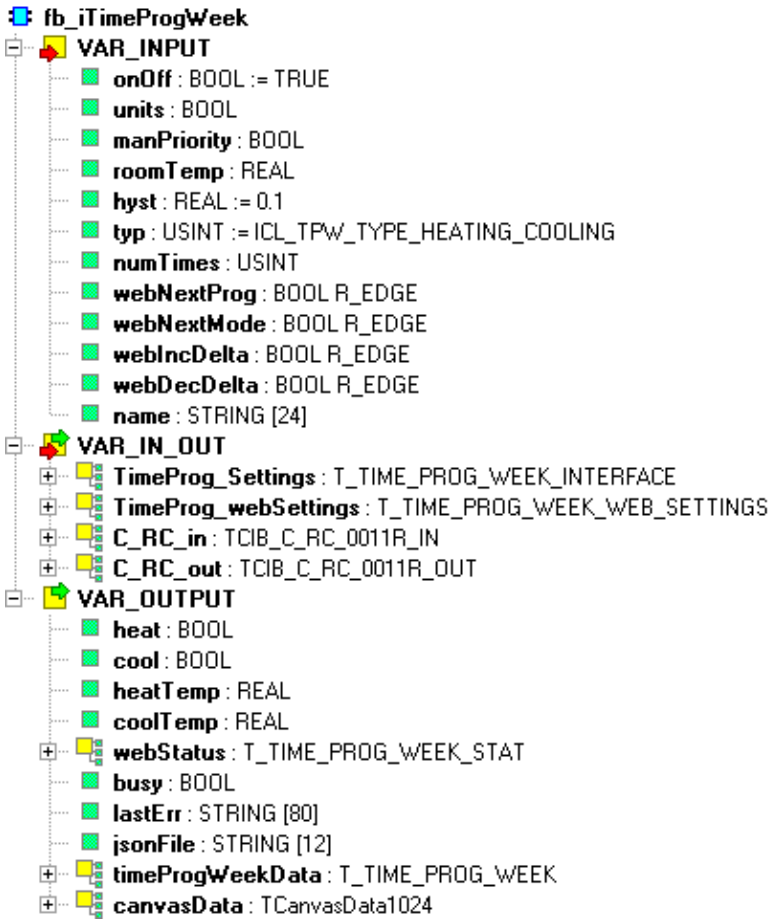
Řízení podle časového programu má několik stavů:

- **auto** funkční blok zvolí žádanou teplotu topení a chlazení podle časového programu
- **ručně** stav umožňující ruční korekci žádané teploty topení / chlazení nebo změnu požadovaného režimu (ekonomy<->útlum<->komfort)
- **dovolená** během dovolené se žádané hodnoty topení resp. chlazení nastaví na teploty pro režim ekonomy

Ve všech stavech funkční blok *fb_iTimeProgWeek* porovná žádané hodnoty pro topení a chlazení s měřenou hodnotou. Pokud je měřená teplota nižší než teplota požadovaná pro topení tak se nastaví výstup *heat*, pokud je měřená teplota vyšší než teplota požadovaná pro chlazení tak se nastaví výstup *cool*, v ostatních případech mají výstupy

heat a *cool* hodnotu FALSE. Chování výstupů *heat* a *cool* lze ovlivnit zadáním hystereze na vstupu *hyst*. Aktuálně nastavené žádané teploty jsou k dispozici na výstupech *heatTemp* (pro topení) a *coolTemp* (pro chlazení).

Proměnné bloku *fb_iTimeProgWeek*



Popis proměnných :

	Proměnná	Typ	Význam
VAR_INPUT			
	onOff	BOOL	TRUE znamená, že řízení topení / chlazení je zapnuté, FALSE vypíná řízení
	units	BOOL	Jednotky teploty FALSE = °C, TRUE = °F
	manPriority	BOOL	Priorita ručního ovládání 0 = ruční nastavení se ukončí při první změně časového programu 1 = ruční nastavení lze ukončit pouze přepnutím do stavu auto
	roomTemp	REAL	Měřená teplota (typicky teplota v místnosti)

	Proměnná	Typ	Význam
	hyst	REAL	Hystereze spínání výstupu *heat* resp. *cool* minimálně 0.0, maximálně 0.5 °C
	typ	USINT	Typ časového programu 1 = pouze topení (ICL_TPW_TYPE_HEATING) 2 = pouze chlazení (ICL_TPW_TYPE_COOLING) 3 = topení i chlazení (ICL_TPW_TYPE_HEATING_COOLING)
	numTimes	USINT	Počet časových značek za den minimálně 2, maximálně 7
	webNextProg	BOOL R_EDGE	Změna stavu auto<->ručně (z web rozhraní)
	webNextMode	BOOL R_EDGE	Změna režimu ekonomy<->útlum<->komfort (z web rozhraní) Způsobí přechod do stavu ručně
	webIncDelta	BOOL R_EDGE	Zvýšení požadované teploty (z web rozhraní) Způsobí přechod do stavu ručně
	webDecDelta	BOOL R_EDGE	Snížení požadované teploty (z web rozhraní) Způsobí přechod do stavu ručně
	name	STRING[48]	Název časového programu
VAR_IN_OUT			
	TimeProg_Settings	T_TIME_PROG_WEEK_IN-TERFACE	
	TimeProg_web-Settings	T_TIME_PROG_WEEK_WEB_SETTINGS	Texty pro lokalizaci web rozhraní
	C_RC_in	TCIB_C_RC_0011R_IN	Vstupní data z modulu C_RC_0011R
	C_RC_out	TCIB_C_RC_0011R_OUT	Vstupní data pro modul C_RC_0011R
VAR_OUTPUT			
	heat	BOOL	Požadavek na topení
	cool	BOOL	Požadavek na chlazení
	heatTemp	REAL	Požadovaná teplota topení
	coolTemp	REAL	Požadovaná teplota chlazení
	webStatus	T_TIME_PROG_WEEK_STAT	Stav řízení (pro web rozhraní)

	Proměnná	Typ	Význam
	busy	BOOL	Blok zpracovává JSON soubor
	lastErr	STRING	Popis chyby vzniklé při zpracování JSON souboru
	jsonFile	STRING[12]	Jméno JSON souboru s popisem časového programu
	timeProgWeekData	T_TIME_PROG_WEEK	Týdenní časový program
	canvasData	TCanvasData1024	Data pro web prvek „kreslící plátno“ (canvas)

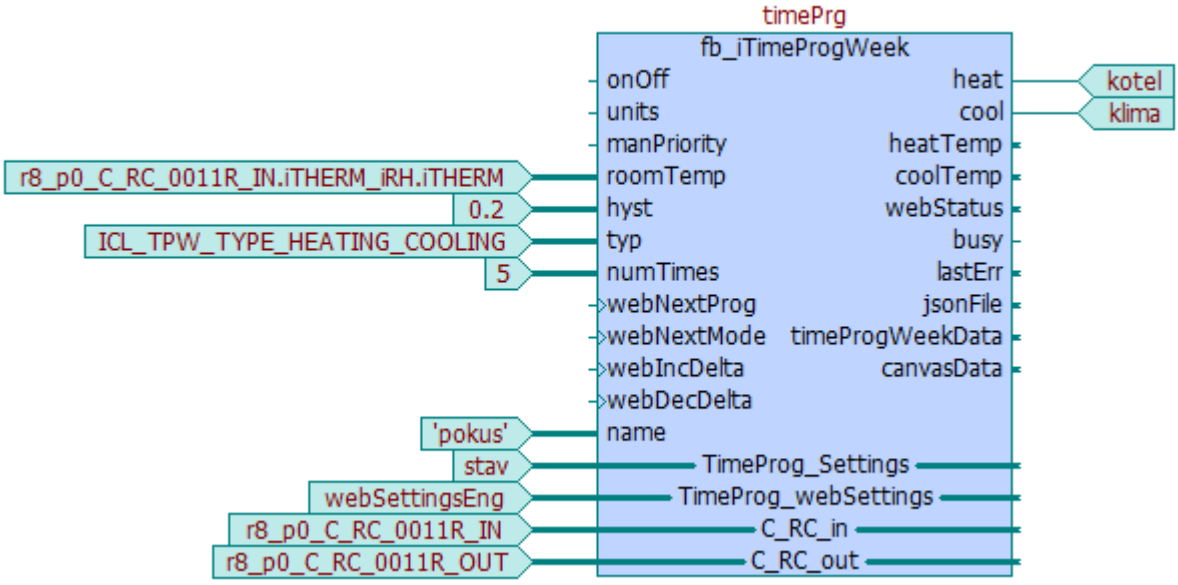
Předpokládejme, že chceme realizovat funkci termostatu s týdenním časovým programem. Lokální ovládání a nastavení termostatu bude zajišťovat modul C-RC-0011R. Termostat bude využit pro topení i chlazení. V jazyce ST bude program vypadat následovně:

```
VAR_GLOBAL RETAIN
  timePrgSettings     : T_TIME_PROG_WEEK_INTERFACE;
END_VAR

PROGRAM prgMain
  VAR
    webSettingsEng  : T_TIME_PROG_WEEK_WEB_SETTINGS_ENG := ( darkBg  := TRUE);
    TimePrg         : fb_iTimeProgWeek;
    kotel           : BOOL;
    klima           : BOOL;
  END_VAR

  TimePrg( roomTemp              := r8_p0_C_RC_0011R_IN.iTHERM_iRH.iTHERM,
           hyst                  := 0.2,
           typ                   := ICL_TPW_TYPE_HEATING_COOLING,
           numTimes              := 5,
           name                  := 'Termostat',
           TimeProg_Settings     := timePrgSettings,
           TimeProg_webSettings  := webSettingsEng,
           C_RC_in               := r8_p0_C_RC_0011R_IN,
           C_RC_out              := r8_p0_C_RC_0011R_OUT,
           heat                  => kotel,
           cool                  => klima);
END_PROGRAM
```

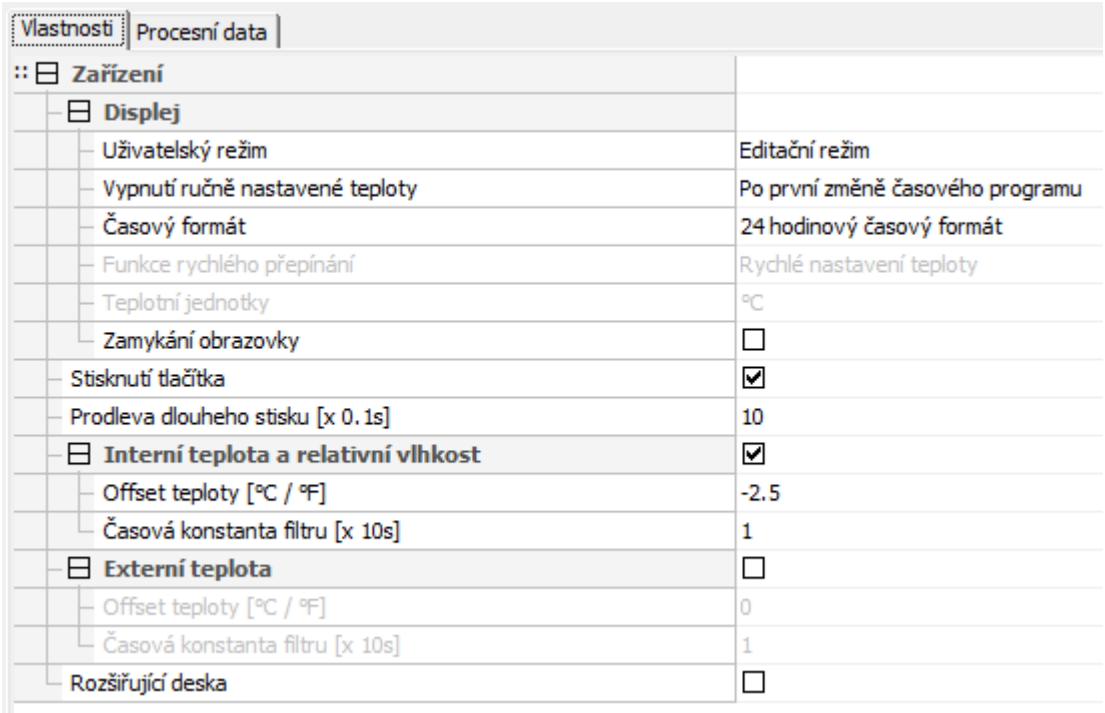
Z programu je vidět, že teplota v referenční místnosti je měřená interním čidlem teploty v modulu C-RC-0011R. Termostat bude pracovat s teplotami ve °C. Hystereze pro výstupy `heat` a `cool` bude 0.2°C. Každý den bude možno rozdělit do 6 intervalů (počet časových značek je `numTimes := 5`). Ruční režim bude automaticky ukončen při první nejbližší změně časového programu (`manPriority := 0`).

Web rozhraní pro nastavení týdenního programu bude mít popis v angličtině a bude mít tmavé pozadí (`webSettingsEng.darkBg := TRUE`). Požadavek na topení je zapisován do proměnné `kotel`. Požadavek na chlazení je zapisován do proměnné `klima`.

Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Nastavení modulu C-RC-0011R ukazuje následující dialog. Displej musí být v editačním režimu.



Ovládání termostatu z web stránky

Pro ovládání slouží vstupy *webNextProg*, *webNextMode*, *webIncDelta* a *webDecDelta*. Zápisem TRUE do těchto proměnných se provedou následující akce:

- *webNextProg* změna stavu auto<->ručně
- *webNextMode* změna režimu ekonomy<->útlum<->komfort
- *webIncDelta* zvýšení požadované teploty

- *webDecDelta* snížení požadované teploty

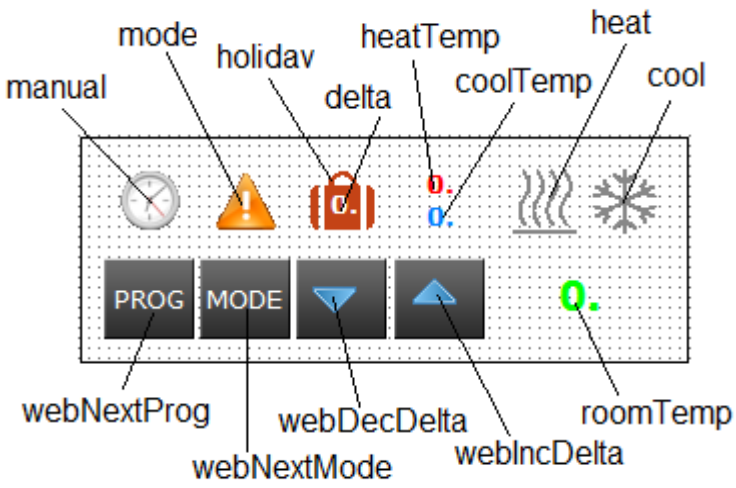
Pokud se řízení nachází ve stavu auto, zápis TRUE do *webNextMode*, *webIncDelta* a *webDecDelta* způsobí přechod do stavu ručně.

Pro zobrazení stavu řízení podle časového programu slouží výstup *webStatus*. Ten má následující strukturu:

Význam jednotlivých položek je následující:

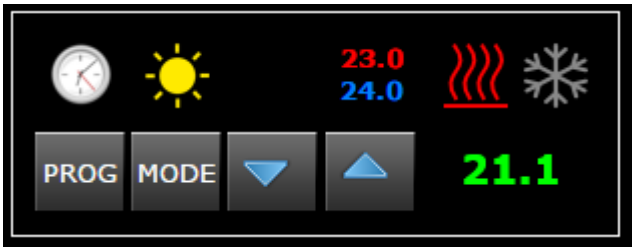
- *manual* 0 = stav auto, 1 = stav ručně
- *holiday* 1 = dovolená
- *heat* požadavek na topení
- *cool* požadavek na chlazení
- *units* jednotky teploty 0 = °C, 1 = °F
- *roomTemp* měřená teplota
- *heatTemp* požadovaná teplota pro topení
- *coolTemp* požadovaná teplota pro chlazení
- *mode* aktuální teplotní režim
  0 = ochrana, 1 = ekonomy, 2 = útlum, 3 = komfort
- *interval* číslo intervalu během dne (0..7)
- *delta* posun žádané teploty ve stavu ručně (-5...+5°C resp. -9...+9°F)
- *initial* pro interní použití

Ovládání ve web stránce může vypadat např. následovně:



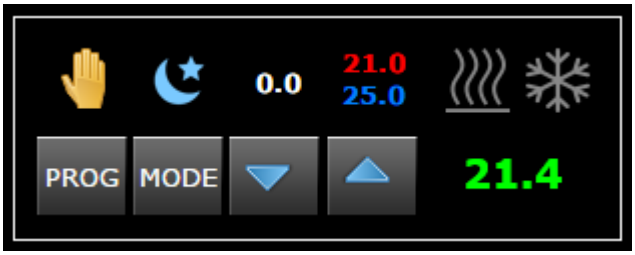
Tlačítka jsou realizovaná prvky pro nastavení hodnoty, teploty jsou realizovány jako zadávací pole (pouze ke čtení), zobrazení mode je vícestavový obrázek, ostatní prvky jsou dvoustavové obázky.

V našem příkladu to pak vypadá následovně:



Řízení je ve stavu auto (tj. probíhá řízení podle časového programu), teplotní režim je komfort, požadovaná teplota topení je 23.0°C, požadovaná teplota chlazení je 24.0°C, měřená teplota je 21.1°C. Požadavek na topení je sepnutý.

Stiskem libovolné klávesy ve stavu auto se přejde do stavu ručně. Při stisku tlačítka PROG se změní pouze stav z auto na ručně, teplotní režim zůstane komfort. Pokud ho chceme změnit stiskneme MODE. Pokud chceme pouze změnit žádanou hodnotu použijeme tlačítka se šipkami, která umožňují posunout žádanou teplotu o 5 stupňů.

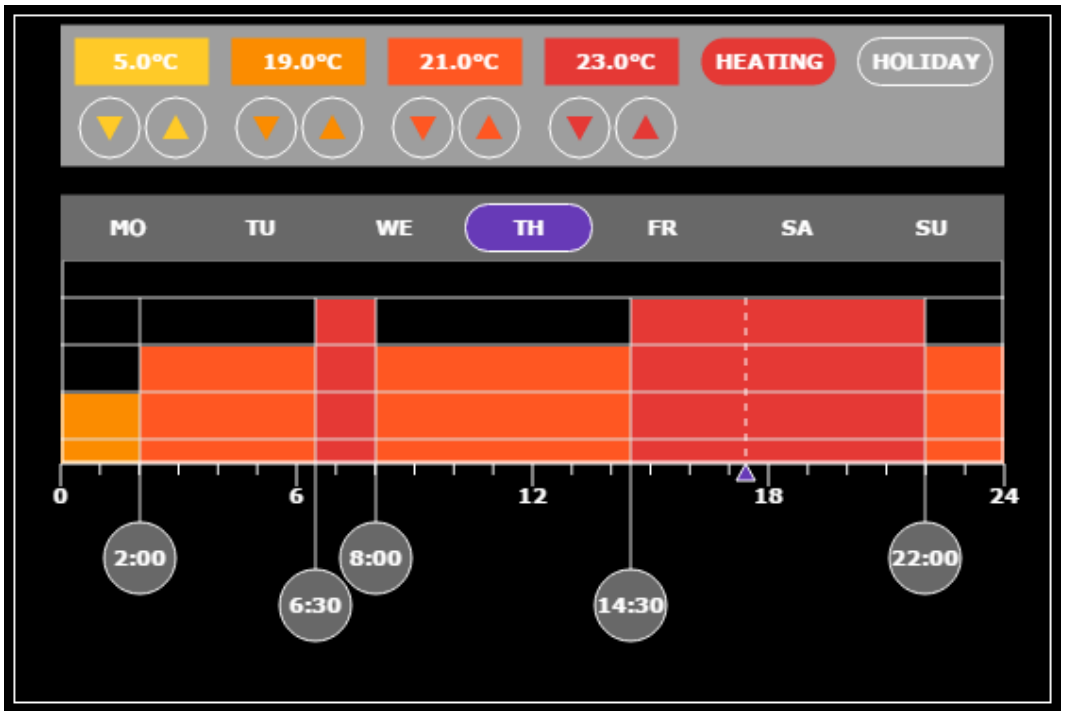


Ze stavu ručně se lze vrátit do stavu auto stiskem tlačítka PROG. Dále je stav ručně automaticky ukončen při přechodu nejbližší časové značky v časovém programu a to za předpokladu, že vstup *manPriority* má hodnotu 0 (například pokud přepneme do stavu ručně ve 14:00 a v časovém programu je v 15:00 změna teplotního režimu na komfort, pak se stav ručně ukončí v 15:00 přechodem do stavu auto).

Během dovolené se ignoruje časový program, žádaná teplota se řídí hodnotami nastavenými pro režim ekonomy. Prvek signalizující stav auto/ručně má viditelnost podmíněnou nulovou hodnotou proměnné *holiday* a prvek signalizující dovolenou má podmíněnou viditelnost nulovou hodnotou proměnné *manual*.

Také během dovolené lze stiskem libovolného tlačítka přejít do stavu ručně. Stav ručně se v tomto případě ukončí pouze stiskem tlačítka PROG nebo ukončením dovolené.

Stránka pro editaci časového programu bude vypadat následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iTimeProgWeek* se do souboru „iFoxtrot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

Každá instance *fb_iTimeProgWeek* přidá následující public proměnné:

	Proměnná	**Typ**	**Význam**
R	*GTSAP1_TPW_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_TPW_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Foxtrot

	Proměnná	Typ	Význam
			2 zálohována.
R	*GTSAP1_TPW_type*	USINT	Typ časového programu 1 = pouze topení 2 = pouze chlazení 3 = topení i chlazení
R	*GTSAP1_TPW_file*	STRING	Jméno JSON souboru s popisem časového programu
R	*GTSAP1_TPW_crc*	UINT	CRC signatura časového programu
RW	*GTSAP1_TPW_update*	BOOL	Změna časového programu z aplikace
R	*GTSAP1_TPW_manual*	BOOL	Příznak stavu ručně
R	*GTSAP1_TPW_holiday*	BOOL	Příznak dovolené
R	*GTSAP1_TPW_heat*	BOOL	Stav výstupu *heat*
R	*GTSAP1_TPW_cool*	BOOL	Stav výstupu *cool*
R	*GTSAP1_TPW_roomTemp*	REAL	Měřená teplota
R	*GTSAP1_TPW_heatTemp*	REAL	Požadovaná teplota pro topení
R	*GTSAP1_TPW_coolTemp*	REAL	Požadovaná teplota pro chlazení
R	*GTSAP1_TPW_mode*	SINT	Aktuální teplotní režim
R	*GTSAP1_TPW_delta*	REAL	Hodnota inkrementu/dekrementu teploty
RW	*GTSAP1_TPW_nextProg*	BOOL	Změna stavu auto <-> ručně
RW	*GTSAP1_TPW_nextMode*	BOOL	Změna teplotního režimu
RW	*GTSAP1_TPW_incDelta*	BOOL	Zvýšení požadované teploty
RW	*GTSAP1_TPW_decDelta*	BOOL	Snížení požadované teploty
RW	*GTSAP1_TPW_manualSet*	SINT	Nastavení stavu ručně
RW	*GTSAP1_TPW_modeSet*	SINT	Změna teplotního režimu

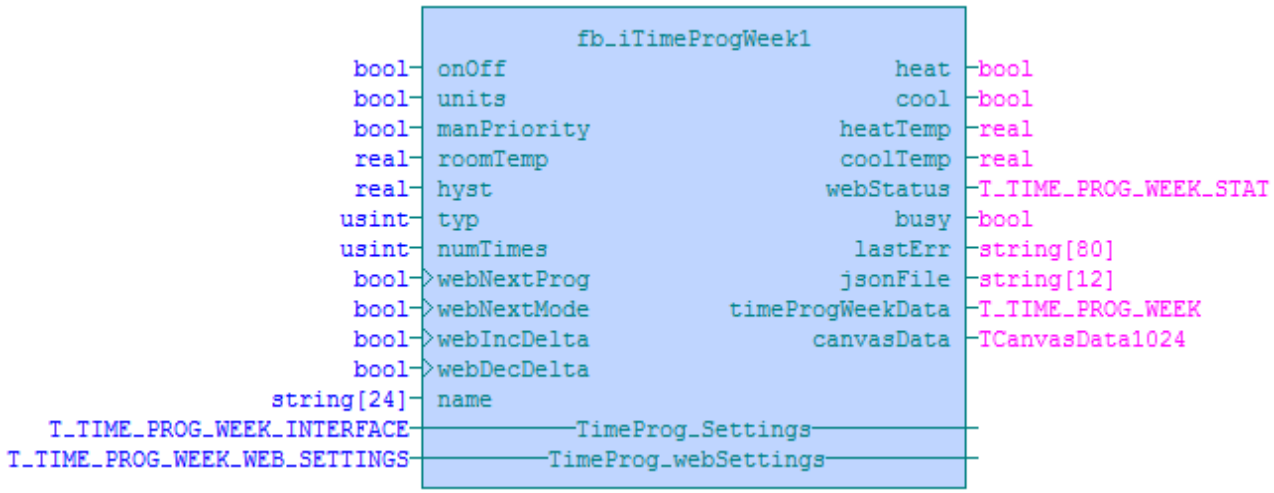
R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.32 Funkční blok fb_iTimeProgWeek1, ..., fb_iTimeProgWeek3

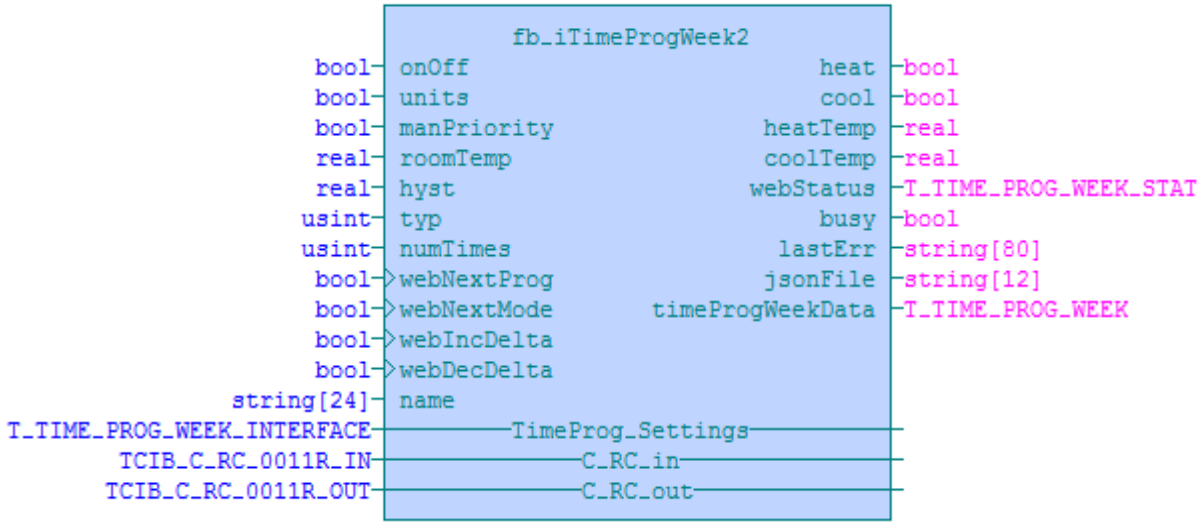
Knihovna : *iControlLib*

Funkční bloky *fb_iTimeProgWeek1*, *fb_iTimeProgWeek2* a *fb_iTimeProgWeek3* jsou pouze modifikací základního bloku *fb_iTimeProgWeek*. Popis viz předcházející kapitola. Motivací pro tyto bloky je úspora paměti a strojového času procesoru PLC.

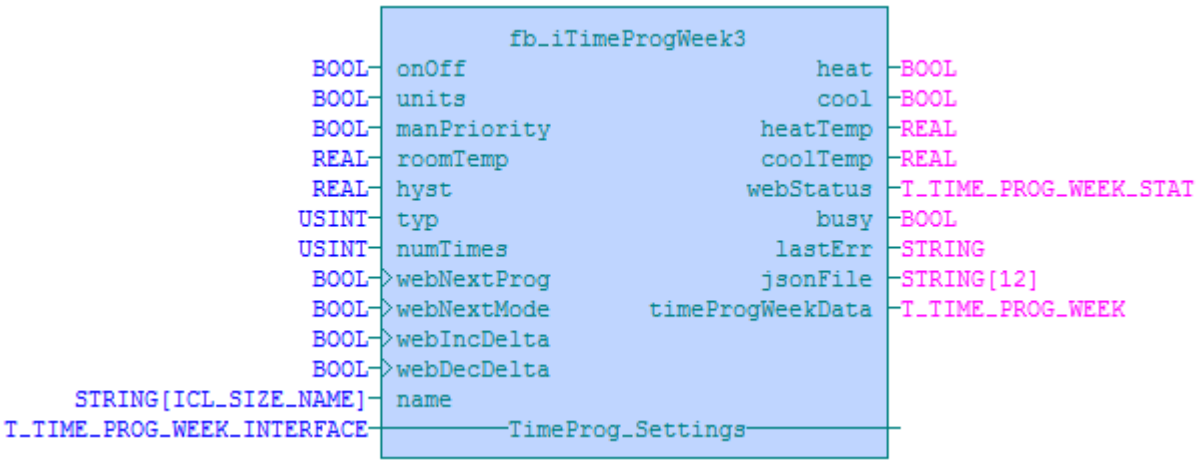
Funkční blok *fb_iTimeProgWeek1* neobsahuje spolupráci s modulem C-RC-0011R. Časový program lze nastavovat pouze z web rozhraní nebo z uživatelské aplikace.



Funkční blok *fb_iTimeProgWeek2* neobsahuje podporu pro nastavování časového programu z web rozhraní. Časový program lze nastavovat pouze z modulu C-RC-0011R nebo z uživatelské aplikace.

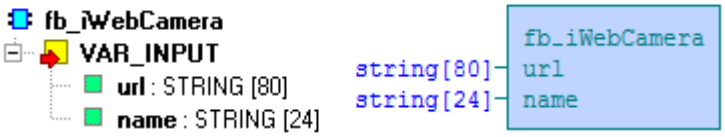


Funkční blok *fb_iTimeProgWeek3* neobsahuje spolupráci s modulem C-RC-0011R. Časový program lze nastavovat pouze z uživatelské aplikace. Ovládat topení/chlazení je možné z uživatelské aplikace nebo z web stránky.



6.33 Funkční blok fb_iWebCamera

Knihovna : *iControlLib*



Funkční blok *fb_iWebCamera* slouží k zahrnutí webové kamery do uživatelské aplikace.

Vstup *url* definuje url adresu webové kamery. Uživatelská aplikace získává obrázek přímo z kamery. Pokud má kamera veřejnou IP adresu, pak bude obrázek z kamery dostupný vždy, když bude mít aplikace přístup k internetu. Pokud bude kamera v lokální síti, pak aplikace zobrazí obrázek z kamery pouze tehdy, bude-li v téže síti.

Vstup *name* slouží k rozlišení kamer v uživatelské aplikaci.

Popis proměnných :

	Proměnná	***Typ***	***Význam***
VAR_INPUT			
	url	STRING[80]	Url adresa kamery
	name	STRING[48]	Název kamery

Předpokládejme, že chceme vidět obrázek z kamery sledující náměstí v Kolíně v uživatelské aplikaci. Kamera je veřejně dostupná na adrese http://posta.mukolin.cz/axis-cgi/jpg/image.cgi. V jazyce ST bude program vypadat následovně:

```
PROGRAM prgMain
  VAR
    cityCamera : fb_iWebCamera;
  END_VAR

  cityCamera( url  := 'http://posta.mukolin.cz/axis-cgi/jpg/image.cgi',
              name := 'kamera Kolín');
END_PROGRAM
```

Stejnou funkci lze naprogramovat v jazyce CFC například následovně:

Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iWebCamera* se do souboru „iFox-trot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

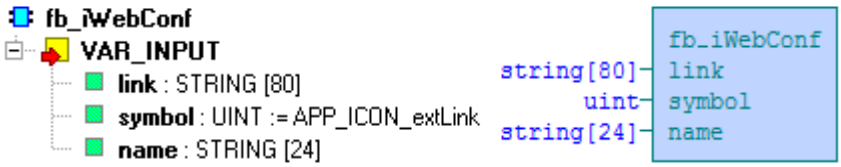
Každá instance *fb_iWebCamera* přidá následující public proměnné:

	Proměnná	Typ	Význam
R	*GTSAP1_CAMERA_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_CAMERA_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Fox-trot 2 zálohována.
R	*GTSAP1_CAMERA_url*	STRING[48]	Url adresa kamery

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

6.34 Funkční blok fb_iWebConf

Knihovna : *iControlLib*



Funkční blok *fb_iWebConf* umožňuje v uživatelské aplikaci s danou podporou zobrazit web stránku z PLC nebo spustit další aplikaci (seznam viz dále).

Vstup *link* definuje odkaz na web stránku v PLC nebo url aplikace (pozor ! rozlišují se velká a malá písmena). Vstup *symbol* umožňuje přiřadit ikonu v uživatelské aplikaci.

Vstup *name* udává výchozí název v uživatelské aplikaci.

Popis proměnných :

	Proměnná	Typ	Význam
VAR_INPUT			
	link	STRING[80]	Název web stránky v PLC nebo odkaz na aplikaci, kterou chceme spouštět
	symbol	UINT	Kód ikony v uživatelské aplikaci
	name	STRING[48]	Výchozí název v uživatelské aplikaci

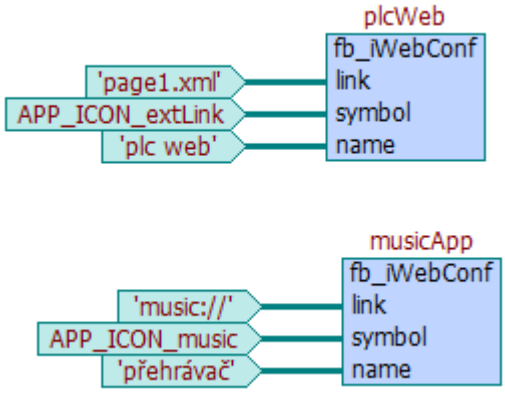
z uživatelské aplikace lze spouštět následující aplikace:

url	aplikace
SoundTouch://	Bose SoundTouch Controller https://itunes.apple.com/cz/app/soundtouch-controller/id708379313?mt=8
control4v2://	Control4 https://itunes.apple.com/us/app/control4/id734435367?mt=8
sonos://	Sonos Controller https://itunes.apple.com/cz/app/sonos-controller/id293523031?mt=8
vlc-x-callback://	VLC s parametry https://wiki.videolan.org/Documentation:IOS/#x-callback-url
vlc://	VLC bez moznosti dalsich parametru
xbmcremote://	Official Kodi Remote https://itunes.apple.com/cz/app/official-kodi-remote/id520480364?mt=8
music://	Systémová iOS Music aplikace

Předpokládejme, že chceme přidat do uživatelské aplikace odkaz na web stránku v PLC (aby bylo možné zobrazit tuto stránku přímo z uživatelské aplikace) a dále možnost spustit přehrávač hudby. V jazyce ST bude program vypadat následovně:

```
PROGRAM prgMain18
  VAR
    plcWeb   : fb_iWebConf;
    musicApp : fb_iWebConf := ( link   := 'music://',
                                symbol := APP_ICON_music,
                                name   := 'přehrávač');
  END_VAR

  plcWeb( link   := 'page1.xml',
          symbol := APP_ICON_extLink,
          name   := 'plc web');

  musicApp();  // pro spusteni IOs music aplikace z uživatelské aplikace
END_PROGRAM
```

Stejnou funkci lze naprogramovat v jazyce CFC například následovně:



Integrace s uživatelskými aplikacemi

Pro každou použitou instanci funkčního bloku *fb_iWebConf* se do souboru „iFox-trot.pub“ (public soubor) automaticky generuje sada proměnných, která slouží pro integraci s uživatelskými aplikacemi.

Každá instance *fb_iWebConf* přidá následující public proměnné:

	Proměnná	**Typ**	**Význam**
R	*GTSAP1_WEBCONF_name*	STRING[48]	Kopie vstupu *name*
RW	*GTSAP1_WEBCONF_enable*	BOOL	Povoluje viditelnost v uživatelské aplikaci. Proměnná je v případě Fox-trot 2 zálohována.
R	*GTSAP1_WEBCONF_url*	STRING	Url adresa web stránky v PLC nebo url adresa aplikace
R	*GTSAP1_WEBCONF_symbol*	UINT	Kód ikony

R = pouze čtení, **W** = pouze zápis, **RW** = čtení i zápis

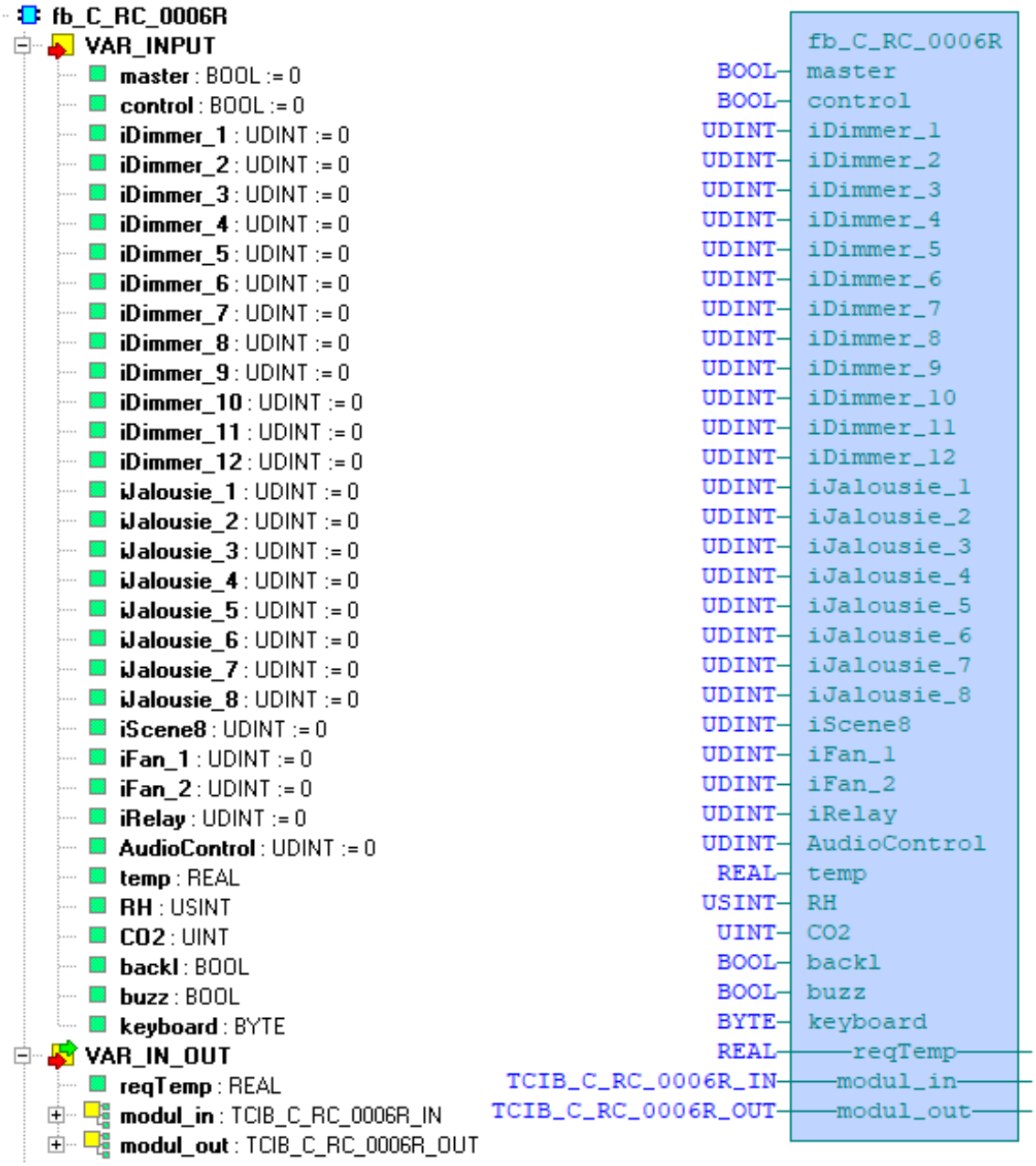
7 DOPLŇKOVÉ BLOKY

V knihovně iControlLib jsou definovány následující doplňkové bloky:

Doplňkové funkční bloky	**Popis**
fb_C_RC_0006R	Synchronizace dat s modulem C-RC-0006R
fb_Button1	Vyhodnocení krátkého a dlouhého stisku tlačítka
fb_RndPulse	Generátor náhodných pulzů
fb_JalAlarm	Vyhodnocení alarmů pro žaluzie
fb_TimeAction	Jednoduché časové ovládání

7.1 Funkční blok fb_C_RC_0006R

Knihovna : *iControlLib*

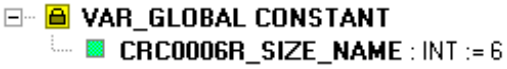


Funkční blok fb_C_RC_0006R je určen k synchronizaci dat a nastavení s pokojovým nástěnným ovladačem C-RC-0006R (dále jen modul).

Modul umožňuje zobrazit čas, teplotu, vlhkost, koncentraci CO2 a připojit funkční bloky nebo řídící struktury pro ovládání:

- 12 stmívatelných světel
- 8 žaluzií
- 8 scén
- 2 ventilátory
- výstup pro relé
- audio zařízení

Definována je následující konstanta:



Konstanta CRC0006R_SIZE_NAME definuje délku názvů zařízení zobrazených na displeji modulu.

Vstupy *iDimmer_1 - iDimmer_12, iJalousie_1 - iJalousie_8, iScene8, iFan_1* a *iFan_2* slouží k připojení funkčních bloků z iControl knihovny. K vstupu *AudioControl* se připojuje řídící struktura z *BoseAudioLib* knihovny. Vstupy *temp*, *RH* a *CO2* (teplota, relativní vlhkost a koncentrace CO2) jsou veličiny zobrazené na displeji modulu.

Pokud je vstup *master* nastaven na hodnotu *TRUE* , kromě samostatného ovládání světel modul umožňuje zhasnout všechna najednou.

Vstupně-výstupní parametry *modul_in a modul_out* jsou vstupní a výstupní zóny ovládaného modulu, *reqTemp* je požadovaná (regulovaná) teplota v místnosti.

Popis proměnných :

	Proměnná	***Typ***	***Význam***
VAR_INPUT			
	master	BOOL	Nastavení modulu jako master
	iDimmer_1 - 12	UDINT	Pointer na funkční blok stmívatelného světla fb_iDimmer nebo řídící strukturu TIDIMMER_UNIT
	iJalousie_1 - 8	UDINT	Pointer na funkční blok žaluzie fb_iJaloiusie
	iScene8	UDINT	Pointer na funkční blok scén fb_iScene8
	iFan_1 - 2	UDINT	Pointer na funkční blok ventilátoru fb_iFan
	iRelay	UDINT	Pointer na funkční blok reléového výstupu fb_iRelay
	AudioControl	UDINT	Pointer na řídící strukturu audio zařízení TCIB_C_RC_0006R_AudioSource_Bose nebo TCIB_C_RC_0006R_AudioSource_Denon
	temp	REAL	Teplota
	RH	USINT	Relativní vlhkost
	CO2	UINT	Koncentrace oxidu uhličitého
	backl	BOOL	Podsvícení displeje

	Proměnná	Typ	Význam
	buzz	BOOL	Zvuková signalizace
	keyboard	BYTE	Intenzita podsvícení klávesnice 0 – 10 (0 – 100%)
VAR_IN_OUT			
	reqTemp	REAL	Požadovaná (regulovaná) teplota
	modul_in	TCIB_C_RC_0006R_IN	Vstupní datová zóna modulu
	modul_out	TCIB_C_RC_0006R_OUT	Výstupní datová zóna modulu

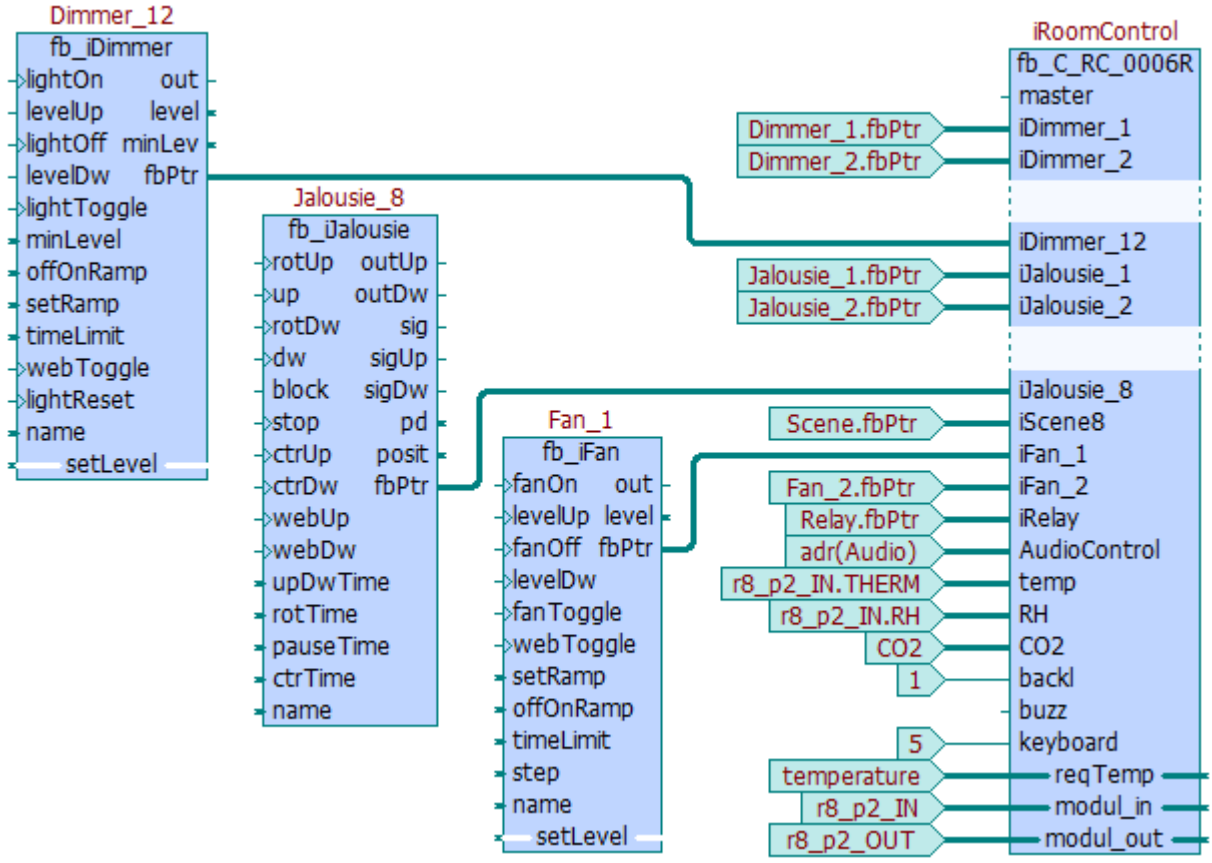
Následující příklad ilustruje, jak ovládat 12 světel, 8 žaluzií, 2 ventilátory, scény a audio v místnosti.

Funkční bloky pro ovládání světel *fb_iDimmer* je třeba na vstup připojovat kontinuálně, pokud budou např. připojena k *Dimmer_1*, *Dimmer_2* a následně až na *Dimmer_4*, nebudou světla na vstupu *Dimmer_4* a dále obsluhována. Totéž platí pro *fb_iJalousie*!!! K připojení bloků slouží výstup *fbPtr* (pointer na funkční blok).

```
PROGRAM prgMain
  VAR
    RoomControl  : fb_C_RC_0006R;

    Dimmer_1     : fb_iDimmer;            // svetlo 1
    Dimmer_2     : fb_iDimmer;            // svetlo 2
    ....
    Dimmer_12    : fb_iDimmer;            // svetlo 12
    Jalousie_1   : fb_iJalousie;          // zaluzie 1
    Jalousie_2   : fb_iJalousie;          // zaluzie 2
    .....
    Jalousie_8   : fb_iJalousie;          // zaluzie 8
    Scene        : fb_iScene8;            // ovladani scen
    Fan_1        : fb_iFan;               // ventilator wc
    Fan_2        : fb_iFan;               // ventilator rekuperace
    Relay        : fb_iRelay;             // ovladani primotopu
    Audio        : TCIB_C_RC_0006R_AudioSource_Bose;  // ovladani audio

    co2          : uint := 920;
    temperature  : real := 22.0;          // pozadovana teplota
  END_VAR

  RoomControl( iDimmer_1    := Dimmer_1.fbPtr,
               iDimmer_2    := Dimmer_2.fbPtr,
               .....
               iDimmer_12   := Dimmer_12.fbPtr,
               iJalousie_1  := Jalousie_1.fbPtr,
               iJalousie_2  := Jalousie_2.fbPtr,
               .....
               iJalousie_8  := Jalousie_8.fbPtr,
               iScene8      := Scene.fbPtr;
               iFan_1       := Fan_1.fbPtr;
               iFan_2       := Fan_2.fbPtr;
```

```
                iRelay        := Relay.fbPtr;
                AudioControl  := ptr_to_udint(adr(Audio));
                temp          := r8_p0_IN.THERM;
                RH            := r8_p0_IN.RH;
                CO2           := co2;
                backl         := 1;
                keyboard      := 5;
                reqTemp       := temperature;
                modul_in      := r8_p0_IN,
                modul_out     := r8_p0_OUT);
END_PROGRAM
```

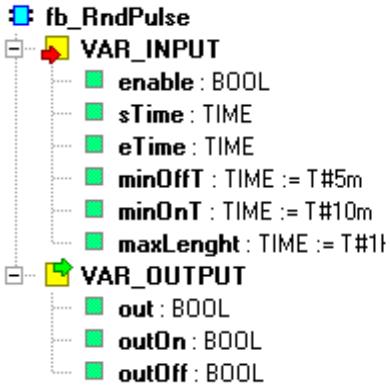
Stejnou funkci lze naprogramovat v jazyce CFC následovně:

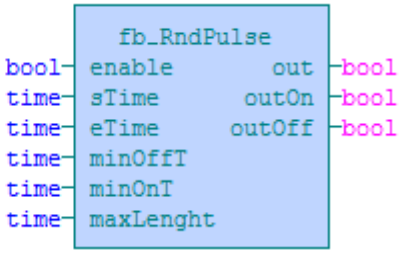


Každé ovládané zařízení je v modulu zobrazeno společně s 6-ti znakovým uživatelským názvem, který vychází ze vstupu *name* jednotlivých funkčních bloků.

7.2 Funkční blok fb_RndPulse

Knihovna : *iControlLib*





Funkční blok *fb_RndPulse* nastavuje svůj výstup na náhodně dlouhé intervaly „0“ a „1“. Umožňuje uživateli nastavit časový úsek pomocí *sTime* a *eTime,* ve kterém bude docházet k náhodnému nastavování výstupu *out*. Pokud denní čas překročí hodnotu *eTime* tak se dokončí započatý časový průběh aktivace výstupu a k další aktivaci již nedojde. Tento blok byl vytvořen primárně pro simulaci lidské aktivity v domech za pomoci spínání světel.

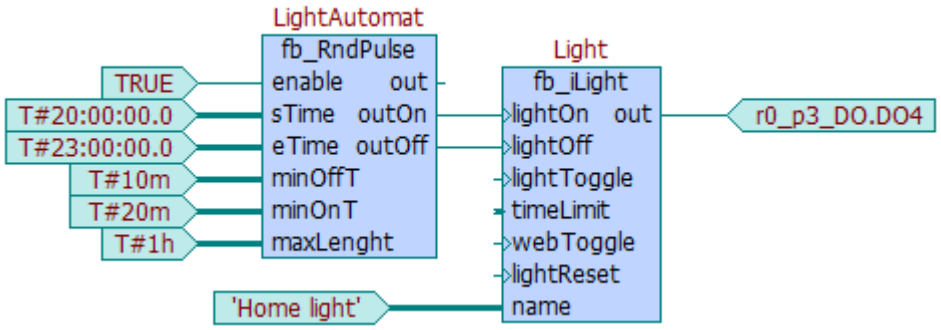
Výstup *outOn* se nastaví na TRUE v okamžiku, kdy se zapíná výstup *out*. Výstup *outOff* se nastaví na TRUE v okamžiku, kdy se vypíná výstup *out*. Oba tyto výstupy jsou sepnuty na dobu jednoho cyklu.

	Proměnná	Typ	Význam
VAR_INPUT			
	enable	BOOL	Povolení funkce bloku
	sTime	TIME	Čas začátku generování náhodných pulzů
	eTime	TIME	Konec generování náhodných pulzů
	minOffT	TIME	Minimální délka intervalu „0“
	minOnT	TIME	Minimální délka intervalu „1“
	maxLenght	TIME	Maximální délka pulzu jak „1" tak „0".
VAR_OUTPUT			
	out	BOOL	výstup náhodně generovaných a náhodně dlouhých pulzů
	outOn	BOOL	TRUE při náběžné hraně výstupu *out*
	outOff	BOOL	TRUE při sestupné hraně výstupu *out*

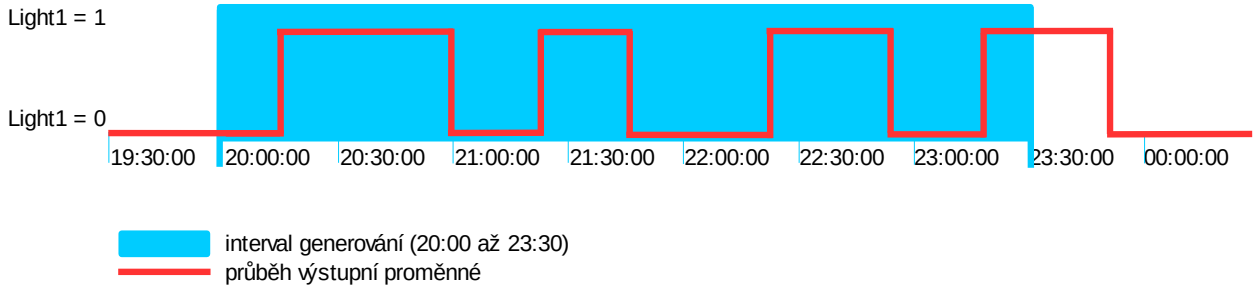
Jednoduchý příklad použití *fb_RndPulse* v jazyce ST:

```
PROGRAM prgMain9
  VAR
    LightAutomat   : fb_RndPulse;
    Light          : fb_iLight;
  END_VAR

  LightAutomat( enable    := TRUE,
                sTime     := T#20:00:00.0,      // start time
                eTime     := T#23:30:00.0,      // end time
                minOffT   := T#10m ,
                minOnT    := T#20m ,
                maxLenght := T#1h );
  Light       ( lightOn   := LightAutomat.outOn,
                lightOff  := LightAutomat.outOff,
                name      := 'Home light' ,
                out       => r0_p3_DO.DO4);
END_PROGRAM
```

Při aktivaci *enable* dochází mezi 20:00 a 23:30 k náhodné zapínání světla, přičemž minimální trvání vypnutého světla je 10 minut a minimální délka zapnutého 20 minut. Maximální délka vypnutého či zapnutého světla je 60 minut. Stejná funkce realizovaná za použití jazyku CFC:



Jeden z možných průběhů spínání světla za použití programu z příkladu:

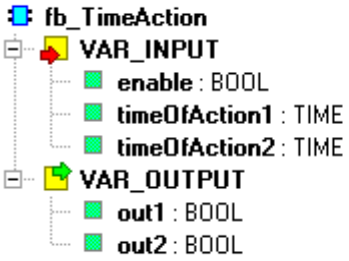


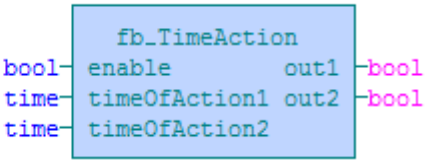
Na tomto průběhu je názorně vidět fungování programu z příkladu. Začátek aktivního intervalu neznamená okamžitě aktivovaný výstup a konec intervalu opět neznamená okamžitou deaktivaci výstupu. Maximální čas vypnutí výstupu po skončení intervalu generování je *eTime* + *maxlenght* v tomto případě tedy 00:30. Také může dojít k poslední deaktivaci výstupu už v *eTime* – *maxlenght*, v našem případě v 22:00.

V případě nastavení sTime a eTime na stejnou hodnotu blok žádné pulzy negeneruje. Stejně tak při nastavení *minOffT* nebo *minOnT* na větší hodnotu než je *maxLenght* nedojde ke generování pulzů.

7.3 Funkční blok fb_TimeAction

Knihovna : *iControlLib*





Funkční blok *fb_TimeAction* slouží k vytvoření impulsu v uživatelem zadaný čas.

Přivedením TRUE na vstup *enable* se povolí akce celého funkčního bloku. Jakmile vnitřní čas plc dosáhne hodnoty zadané na vstup *timeOfAction1* na výstup *out1* se zapíše 1 a setrvá na něm po dobu jedné otočky programu. Vstup *TimeOfAction2* stejným způsobem ovládá výstup *out2.* Čas 0:0 je nastaven jako mrtvá zóna, což znamená že vstupy *timeOfAction* nastavené na tento čas nezpůsobí aktivaci výstupu.

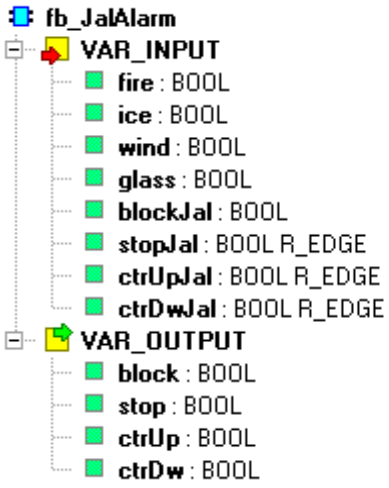
Popis proměnných:

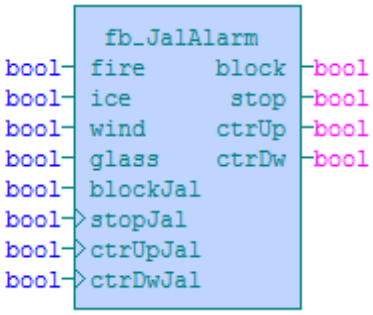
	Proměnná	Typ	Význam
VAR_INPUT			
	enable	BOOL	povolení aktivace výstupů
	timeOfAction1	TIME	čas aktivace výstupu *out1*
	timeOfAction2	TIME	čas aktivace výstupu *out2*
VAR_OUTPUT			
	out1	BOOL	Výstup 1
	out2	BOOL	Výstup 2

Příklad použití *fb_TimeAction* v jazyce ST pro jednoduché časové řízení žaluzií. Žaluzie jsou připojené k CIB modul reléových výstupů C-OR-0202B. Pro jejich ovládání použijeme tlačítkový CIB modul C-WS-0200R-Logus Výrobcem udaná prodleva pro reverzaci je 0.5 sec a naměřená doba kompletního pojezdu je 20 sec. Aktivace výstupů *fb_TimeAction* je nastavena na 23:00 a 7:30.

7.4 Funkční blok fb_JalAlarm

Knihovna : *iControlLib*





Funkční blok *fb_JalAlarm* slouží k vyhodnocení poplachů pro žaluzie. Používá se jako předřadný blok pro *fb_iJalousie*, což je funkční blok na ovládání žaluzií. Informace o poplachu se připojuje na některý z následujících vstupů:

- *fire* požární poplach (nejvyšší priorita)
- *ice* námraza
- *wind* velký vítr
- *glass* rozbité sklo (nejnižší priorita)

Pokud je některý z uvedených vstupů nastaven na hodnotu TRUE, funkční blok *fb_JalAlarm ovládá* svoje výstupy následovně: požární poplach (*fire*) vytáhne žaluzie a zablokuje vstupy pro ovládání žaluzie, poplach námrazy (*ice*) zastaví případný pohyb žaluzie a zablokuje vstupy pro ovládání žaluzie, poplach překročení rychlosti větru (*wind*) vytáhne žaluzie a zablokuje vstupy pro ovládání žaluzie a poplach rozbití skla (*glass*) zavře žaluzie. Poplach je aktivní po celou dobu, kdy je na příslušný vstup přivedena hodnota TRUE. Deaktivovat poplach lze pouze nastavením vstupu na FALSE nebo aktivací poplachu s vyšší prioritou.

Další vstupy slouží k centrálnímu ovládání a mají následující význam:

- *blockJal* blokuje ovládací vstupy připojené žaluzie
- *stopJal* zastaví pohyb připojené žaluzie
- *ctrUp* centrální otevření žaluzií
- *ctrDw* centrální zavření žaluzií

Vstupy centrálního ovládání mají nižší prioritu než poplachové vstupy.

Výstupy *fb_JalAlarm* se připojují na stejnojmenné vstupy bloku *fb_iJalousie.*

Popis proměnných :

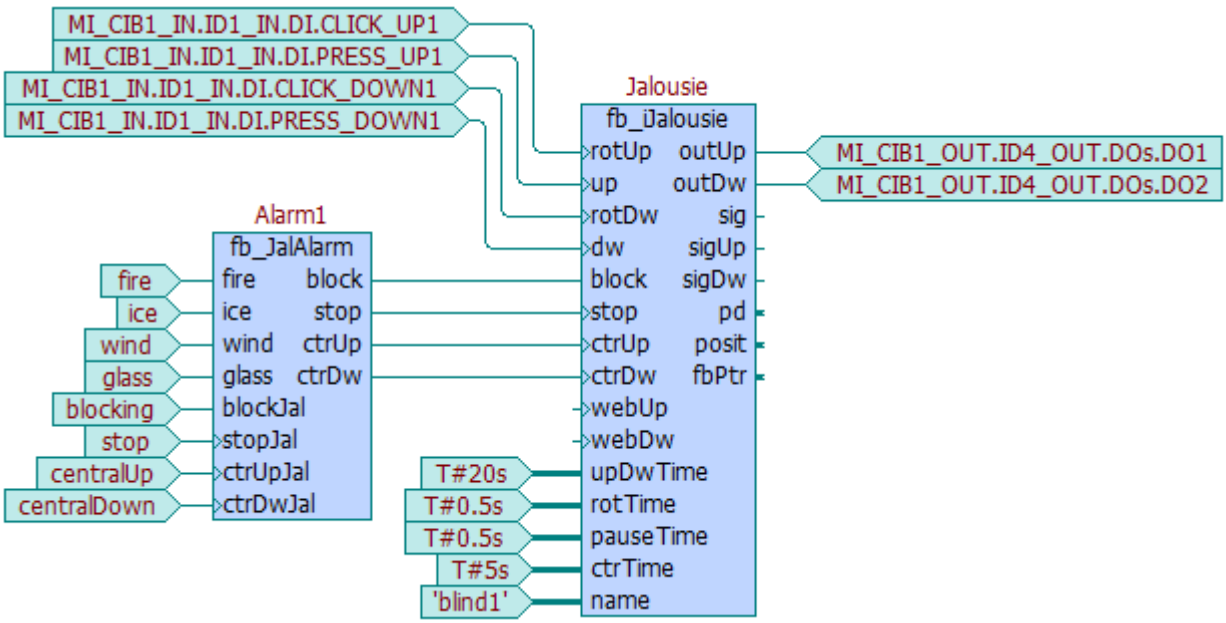
	Proměnná	Typ	Význam
VAR_INPUT			
	fire	BOOL	Poplach „oheň“: vytáhne žaluzie a blokuje vstupy
	ice	BOOL	Poplach „námraza“: zastaví pohyb a blokuje vstupy
	wind	BOOL	Poplach „vítr“: vytáhne žaluzie a blokuje vstupy
	glass	BOOL	Poplach „sklo“: zavře žaluzie
	blockJal	BOOL	Blokování ovládacích vstupů připojené žaluzie
	stopJal	BOOL R_EDGE	Zastavení pohybu žaluzie
	ctrUpJal	BOOL R_EDGE	Centrální otevření žaluzií
	ctrDwJal	BOOL R_EDGE	Centrální zavření žaluzií
VAR_OUTPUT			
	block	BOOL	Propojit se vstupem „*fb_iJalousie.block*“
	stop	BOOL	Propojit se vstupem „*fb_iJalousie.stop*“
	ctrUp	BOOL	Propojit se vstupem „*fb_iJalousie.ctrUp*“
	ctrDw	BOOL	Propojit se vstupem „*fb_iJalousie.ctrDw*“

Příklad propojení *fb_JalAlarm* s blokem *fb_iJalousie* vytvořený v jazyce *ST.* Pro ovládání *fb_Jalousie* je použit nástěnný ovladač C-WS-0200R-Logus. Pro spínání pohonu žaluzie je použit CIB modul reléových výstupů C-OR-0202B. Předpokládáme že samotné aktivační signály poplachů jsou vyřešeny v programu.

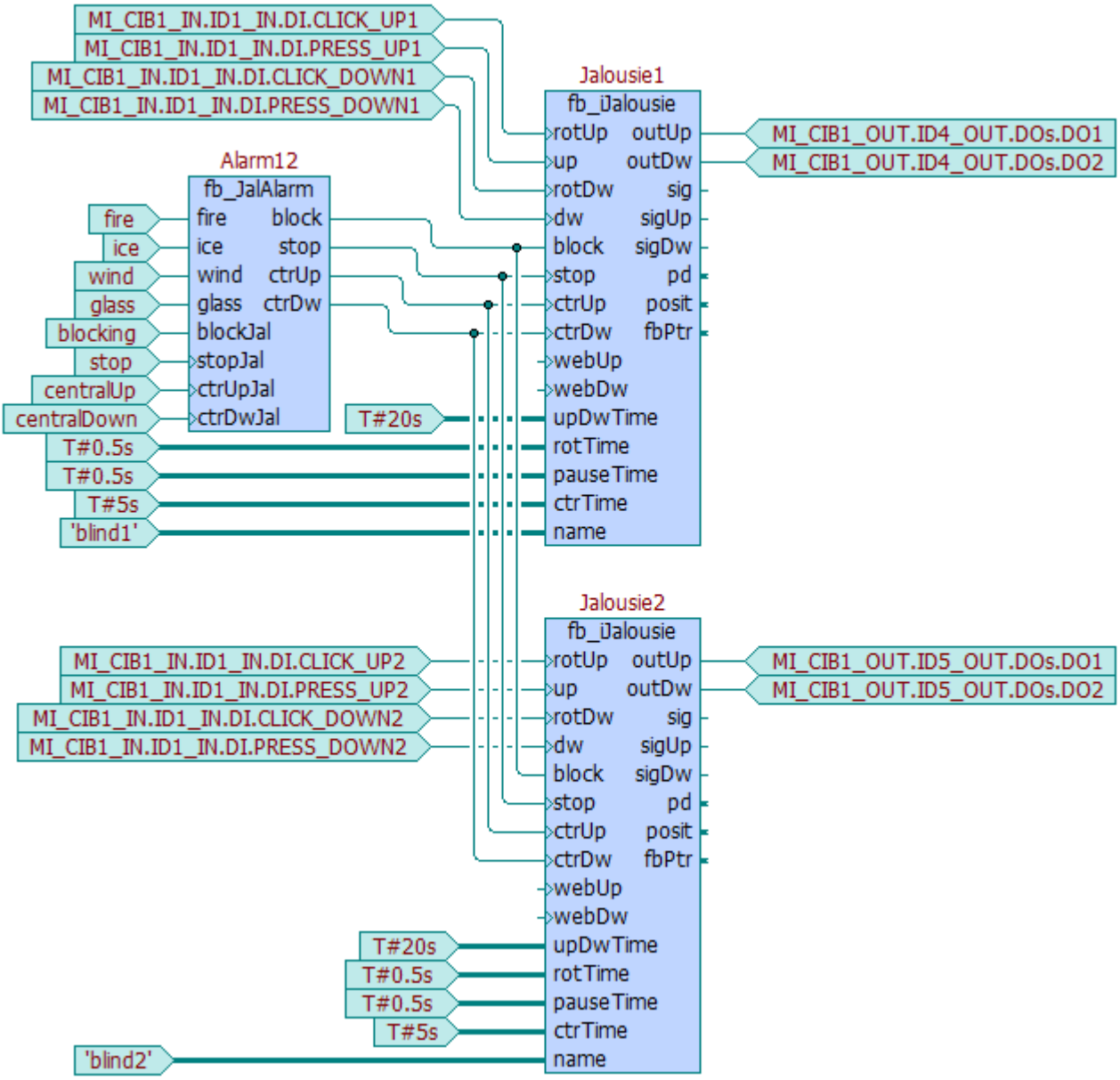
```
PROGRAM prgMain5
  VAR
    Jalousie1     : fb_iJalousie;
    Alarm1        : fb_JalAlarm;
    fire          : bool;
    ice           : bool;
    wind          : bool;
    glass         : bool;
    blocking      : bool;
    stop          : bool;
    centralUp     : bool;
    centralDown   : bool;
  END_VAR
```

```
  // vyhodnoceni alarmu
  Alarm1    (fire         := fire,
             ice          := ice,
             wind         := wind,
             glass        := glass,
             blockJal     := blocking,
             stopJal      := stop,
             ctrUpJal     := centralUp,
             ctrDwJal     := centralDown);

  // ovladani zaluzii
  Jalousie1   (rotUp        := MI_CIB1_IN.ID1_IN.DI.CLICK_UP1,
               up           := MI_CIB1_IN.ID1_IN.DI.PRESS_UP1,
               rotDw        := MI_CIB1_IN.ID1_IN.DI.CLICK_DOWN1,
               dw           := MI_CIB1_IN.ID1_IN.DI.PRESS_DOWN1,
               block        := Alarm1.block,
               stop         := Alarm1.stop,
               ctrUp        := Alarm1.ctrUp,
               ctrDw        := Alarm1.ctrDw,
               upDwTime     := T#20s,
               rotTime      := T#0.5s,
               pauseTime    := T#0.5s,
               ctrTime      := T#5s,
               name         := 'blind1',
               outUp        => MI_CIB1_OUT.ID4_OUT.DOs.DO1,
               outDw        => MI_CIB1_OUT.ID4_OUT.DOs.DO2);
END_PROGRAM
```

Žaluzie jsou ovládány dlouhými a krátkými stisky tlačítka a zároveň signály z bloku alarmů. Tlačítko UP1 ovládá pohyb vzhůru a tlačítko DOWN1 pohyb dolu. Krátký stisk tlačítka aktivuje pootočení o časový krok 500ms. Dlouhý stisk aktivuje kompletní vytažení nebo otevření žaluzie podle aktivovaného tlačítka. Stejnou funkci lze v jazyce CFC naprogramovat například následovně:



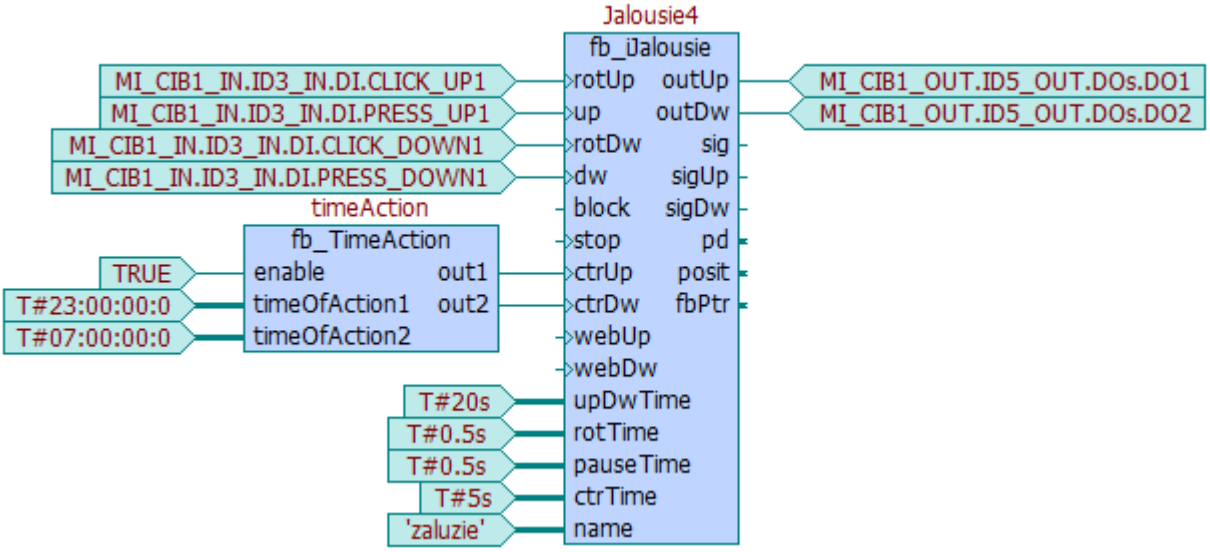
Stejný blok alarmů lze použít k ovládání většího počtu žaluzií. Je doporučeno jedním blokem *fb_iJalAlarm* ovládat více žaluzií pouze v rámci jedné místnosti. Příklad v CFC:



Žaluzie jsou ovládány CIB modulem tlačítek C-WS-0400R-Logus a každou žaluzii spíná vlastní CIB modul reléových výstupů C-OR-0202B. Funkce poplachu a vstupy *block*, *stop*, *ctrUp* a *ctrDw* jsou společné.

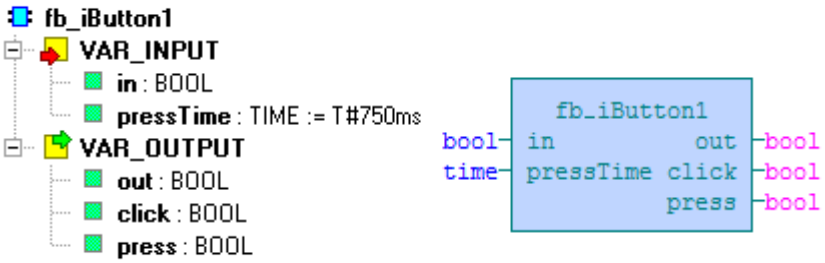
```
PROGRAM prgMain
VAR
    Timeaction    : fb_TimeAction;
    Jalousie4     : fb_iJalousie;
    centralUp     : BOOL;
    centralDown   : BOOL;
  END_VAR

  // Timeaction pripojene na centralUp a centralDown
  Timeaction( enable         := TRUE,
              timeOfAction1  := T#23:00:00.0,
              timeOfAction2  := T#07:30:00.0,
              out1           => centralDown,
              out2           => centralUp );

  // tlacitkove ovladani svetla
  Jalousie4(rotUp        := MI_CIB1_IN.ID1_IN.DI.CLICK_UP1,
            up           := MI_CIB1_IN.ID1_IN.DI.PRESS_UP1,
            rotDw        := MI_CIB1_IN.ID1_IN.DI.CLICK_DOWN1,
            dw           := MI_CIB1_IN.ID1_IN.DI.PRESS_DOWN1,
            ctrUp        := centralUp,
            ctrDw        := centralDown,
            upDwTime     := T#20s,
            rotTime      := T#0.5s,
            pauseTime    := T#0.5s,
            ctrTime      := T#5s,
            name         := 'zaluzie',
            outUp        => MI_CIB1_OUT.ID5_OUT.DOs.DO1,
            outDw        => MI_CIB1_OUT.ID5_OUT.DOs.DO2);
 END_PROGRAM
```

Každý den v 23:00 časová aktivace zavře žaluzie a druhý den v 7:30 je opět plně otevře. Stejná funkce v jazyce CFC.



7.5 Funkční blok fb_Button1

Knihovna : *iControlLib*



Funkční blok *fb_Button1* slouží k vyhodnocení krátkých a dlouhých stisků tlačítka, které je připojeno na běžný binární vstup PLC systému. Tento blok nahrazuje *fb_iButton1*, který byl v předchozích verzích knihovny iControlLib.

Vstup *in* slouží pro připojení tlačítka. Vstup *pressTime* definuje dobu stisknutí, která je potřebná pro vyhodnocení dlouhého stisku. Pokud bude na vstupu *in* hodnota TRUE po kratší dobu než udává *pressTime*, bude stisknutí považováno za krátký stisk.

Výstup *out* je kopií vstupu *in*. Výstup *click* je nastaven na dobu jednoho cyklu PLC na hodnotu TRUE v případě, že došlo ke krátkému stisku tlačítka připojeného na vstup *in*. Výstup *press* je nastaven na hodnotu TRUE pokud je tlačítko stisknuté delší dobu než udává vstup *pressTime*. Výstup *press* pak setrvá na hodnotě TRUE až do okamžiku uvolnění tlačítka (než bude mít vstup *in* hodnotu FALSE).

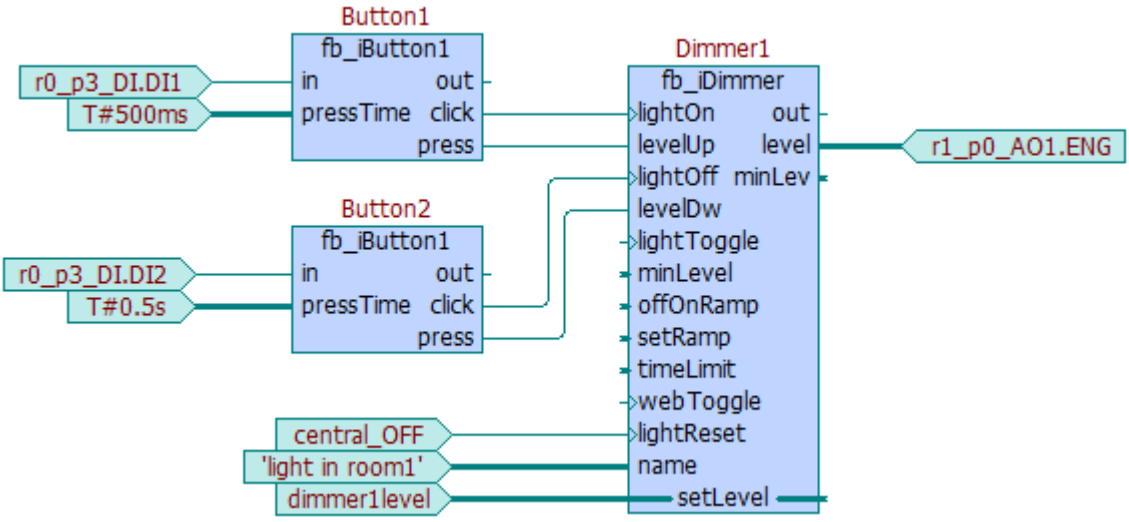
Popis proměnných :

	Proměnná	***Typ***	***Význam***
VAR_INPUT			
	in	BOOL	Vstup pro připojení tlačítka
	pressTime	TIME	Minimální doba pro vyhodnocení dlouhého stisku. Přednastavená hodnota je 750 ms
VAR_OUTPUT			
	out	BOOL	Kopie vstupu *in*
	click	BOOL	TRUE na dobu jednoho cyklu PLC, pokud došlo ke krátkému stisku tlačítka
	press	BOOL	TRUE pokud je tlačítko stisknuto déle než udává vstup *pressTime*

Následující příklad ukazuje použití funkčních bloků *fb_Button1* pro ovládání stmívacího bloku *fb_iDimmer*. Krátké stisky zapínají respektive vypínají světlo, dlouhé stisky mění úroveň jasu světla. V jazyce ST bude program vypadat následovně:

```
VAR_GLOBAL RETAIN
  dimmer1level : REAL;
END_VAR

VAR_GLOBAL
  central_OFF : BOOL;
END_VAR

PROGRAM prgMain
  VAR
    Button1   : fb_Button1;
    Button2   : fb_Button1;
    Dimmer1   : fb_iDimmer;
  END_VAR

  // akce centralniho zhasnuti
  GoOut( action := r0_p3_DI.DI0, out => central_OFF);

  // stmivac ovladany 2 tlacitky
  Button1( in := r0_p3_DI.DI1, pressTime := T#500ms);
  Button2( in := r0_p3_DI.DI2, pressTime := T#0.5s);
  Dimmer1( lightOn    := Button1.click,
           levelUp    := Button1.press,
           lightOff   := Button2.click,
           levelDw    := Button2.press,
           lightReset := central_OFF,
           setLevel   := dimmer1level,
           level      => r0_p3_AO1.ENG);
END_PROGRAM
```

Funkční bloky *Button1* a *Button2* zpracovávají signály od tlačítek připojených na binární vstupy DI1 a DI2. Čas pro vyhodnocení dlouhého stisku je nastaven na 0.5 sec. Světlo je připojeno na analogový výstup AO1. Stejnou funkci lze naprogramovat v jazyce CFC například následovně:

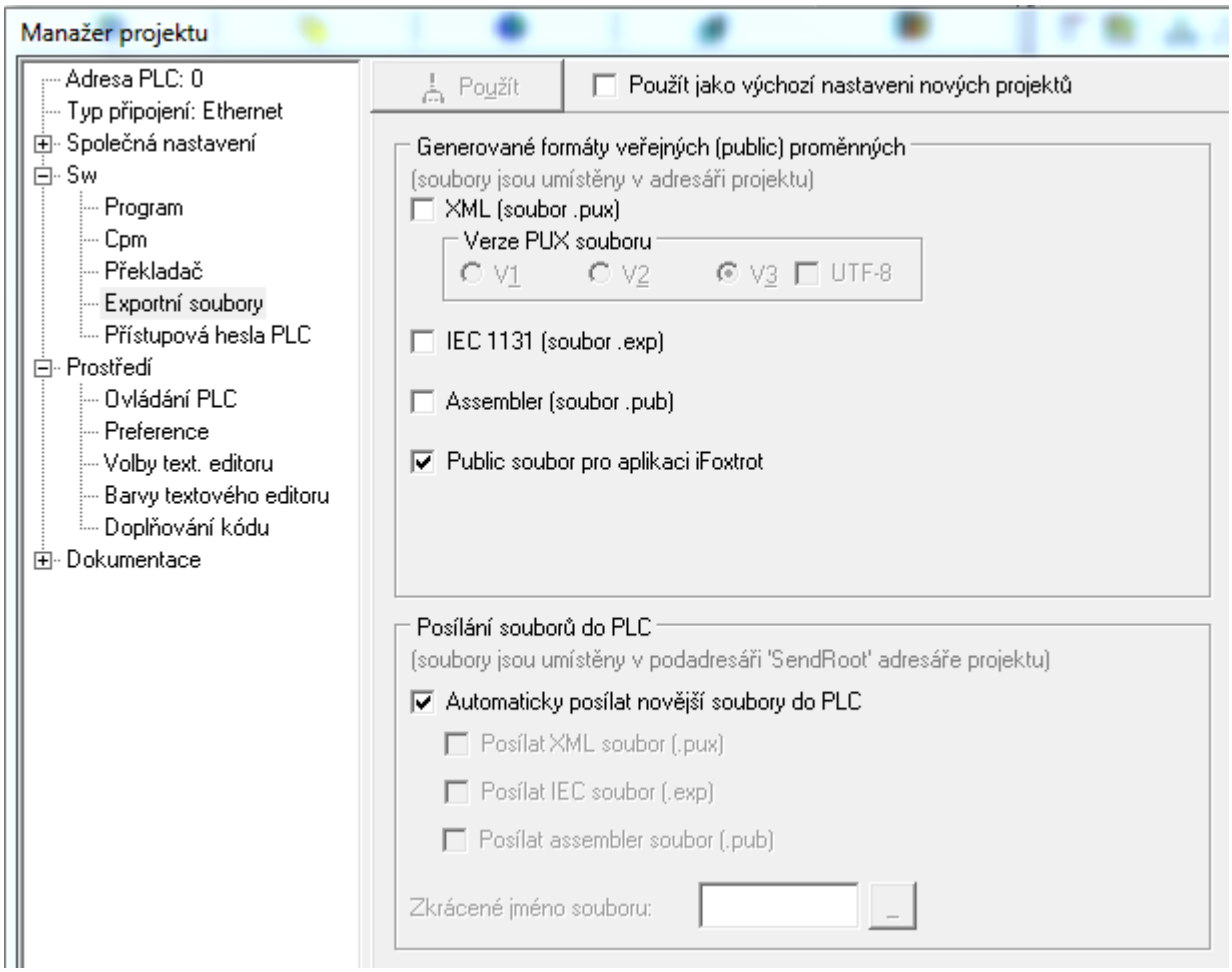


Poznámka.

Některé CIB moduly z produkce Teco mají integrovanou funkci vyhodnocení krátkého a dlouhého stisku. V těchto případech není funkční blok *fb_Button1* potřeba, protože informaci o krátkém resp. dlouhém stisku poskytuje přímo CIB modul. Doba pro vyhodnocení dlouhého stisku se pak zadává v HW konfiguraci modulu. Jedná se například o moduly C-IB-1800M, C-IR-0203M, C-IT-0908S, C-JC-0201B, C-LC-0202B, C-RQ-0400, C-RQ-0600, C-WS-0200R-LOGUS, C-WS-0400R-LOGUS, C-WS-0200R-OBZOR a C-WS-0400R-OBZOR.

8 GENEROVÁNÍ PUBLIC SOUBORU

Public soubor je potřebný pro automatickou integraci uživatelské aplikace s programem v PLC systému. Všechny potřebné informace jsou při překladu PLC programu v prostředí Mosaic automaticky generovány do souboru iFoxtrot.pub za předpokladu, že je v Manažeru projektu v uzlu SW | Exportní soubory zatržena volba „Public soubor pro aplikaci iFoxtrot“. Dále je třeba, aby volba „Automaticky posílat novější soubory do PLC“ byla rovněž zatržená. Správně nastavený dialog v Manažeru projektu ukazuje následující obrázek:



Novější verze prostředí Mosaic (od verze 2018.1) generují soubor iFoxtrot.pub při každém překladu automaticky, takže tuto vlastnost již není třeba zadávat v Manažeru projektu.

9 PODPORA APPLE HOMEKIT

Díky specifickým vlastnostem knihovny iControlLib bylo možné vybrané funkční bloky, které realizují prvky inteligentní domácnosti, integrovat do aplikačního rozhraní HomeKit společnosti Apple.

HomeKit je framework, který umožňuje pomocí zařízení se systémem iOS komunikovat a řídit prvky inteligentního domu, jako jsou světla, rolety či různé typy senzorů. Tato zařízení mohou být od různých výrobců, přičemž každé musí mít implementovánu softwarovou podporu tohoto frameworku. Výrobce zařízení navíc musí získat certifikaci od společnosti Apple.

Jednotlivé instance funkčních bloků z knihovny iControlLib lze považovat za samostatná zařízení domu a samotný PLC systém je pak komunikačním mostem (bridge) k těmto zařízením. PLC systémy Tecomat nemají implentován HAP (Home Accessory Protocol) protokol pro integraci do HomeKit. Lze však využít open-source platformy Homebridge (https://homebridge.io) s instalovaným doplňkem *homebridge-platform-tecomat*, který je oficiálním softwarovým produktem společnosti Teco a.s..

Homebridge je nástroj vytvořený pro běhové prostředí *Node.js*, který emuluje aplikační rozhraní HomeKit pro zařízení, která toto rozhraní standardně nepodporují. Pomocí jednoduchého systému umožňuje implementaci instalovatelných doplňků (plugin), které mohou být navázány na aplikační rozhraní zařízení různých výrobců. Takovým doplňkem je i *homebridge-platform-tecomat*, který zpřístupňuje služby systémů Tecomat s využitím serveru PLCComS (https://www.tecomat.cz/ke-stazeni/software/plccoms/). Díky běhovému prostředí *Node.js* je možné Homebridge spustit téměř na libovolné platformě.

9.1 Balíček teco-homebridge

Pro systémy Foxtrot řady CP-2xxx je k dispozici softwarový balíček *teco-homebridge*, který usnadňuje instalaci Homebridge, včetně všech potřebných doplňků, přímo na HW daného PLC. Balíček usnadňuje i následnou konfiguraci díky integrovanému webovému formuláři dostupnému přes konfigurační webové rozhraní PLC. Pro starší řadu Foxtrot CP-1xxx je možné využít externího HW s instalovaným PLCComS a Homebridge s pluginem *homebridge-platform-tecomat.* Konfigurace se pak provádí manuálně pomocí konfiguračního souboru Homebridge.

Balíček *teco-homebridge* není standardní součástí softwarové výbavy PLC Foxtrot 2. Je možné ho instalovat pomocí instalátoru balíčků, který je dostupný přes konfigurační

web stránky zařízení. Instalace se provede přes sekci *Packages/Management* výběrem daného balíčku a stiskem tlačítka *Install*.

9.2 Konfigurace balíčku teco-homebridge

Po úspěšné instalaci přibude v konfiguračním rozhraní sekce *Packages/teco-homebridge*. Zde je možné parametrizovat běh nástroje Homebridge a pluginu *homebridge-platform-tecomat*.

Konfigurační parametry Homebridge:

- **Adapter** - výběr síťového rozhraní, skrze které bude HomeKit služba dostupná
- **QR Code** - QR kód, který je možné použít pro registraci emulovaného zařízení v HomeKit aplikaci
- **Pin** - PIN kód, který je možné alternativně použít pro registraci emulovaného zařízení v HomeKit aplikaci místo QR kódu
- **Clear Accessories** - zaškrtnutím této volby je při restartu Homebridge serveru vymazán adresář s pamětí již registrovaných zařízení
- **Clear Persist** - zaškrtnutím této volby je při restartu Homebridge serveru vymazán adresář s perzistentními daty, které obsahují informace o spárování Homebridge s aplikací HomeKit. Vymazání adresáře lze chápat jako nastavení aplikace Homebridge do továrního nastavení a celé párování s HomeKit je nutné učinit znovu
- **Debug** - povolí rozšířený výpis ladících informací do log souboru

Konfigurační parametry pluginu Tecomat platform:

- **Host** - IP adresa PLCComS serveru s iFoxtrot public souborem

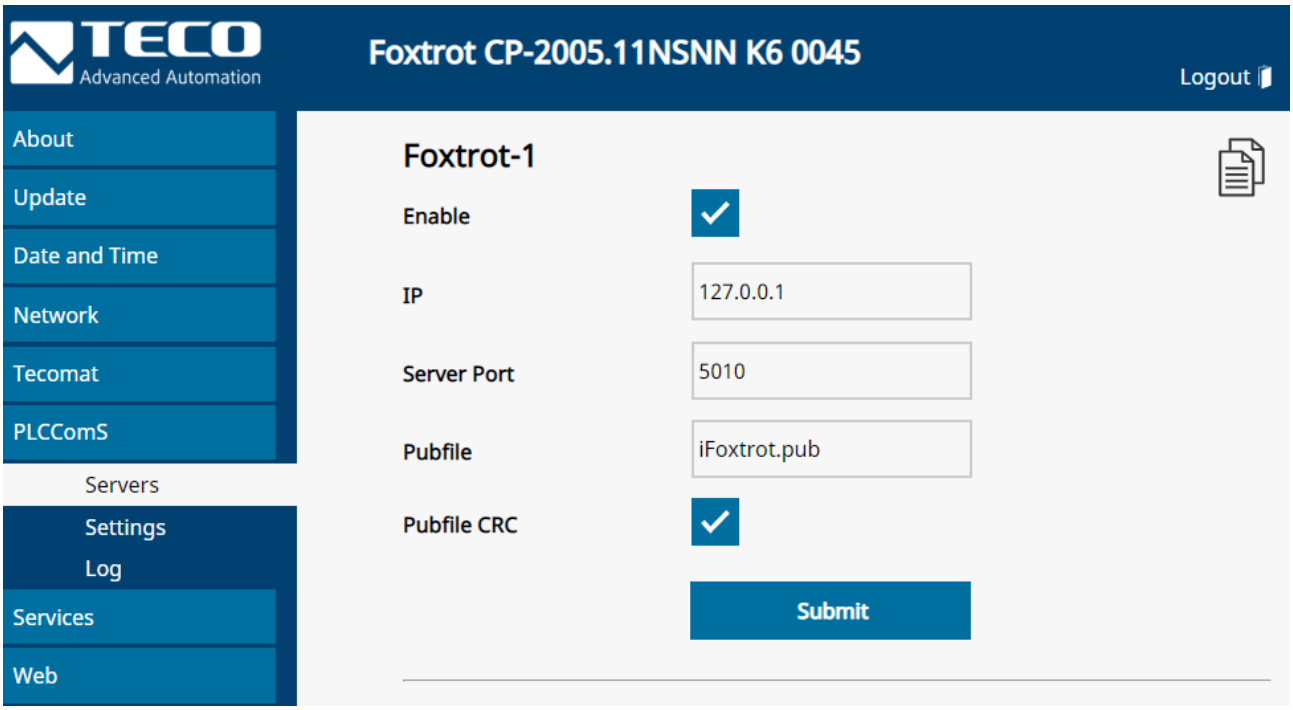
- **Port** - port PLCComS serveru
- **Timeout** - timeout pro připojení k PLCComS serveru
- **iFoxtrot Filter** - zaškrtnutím této volby bude na výčet iFoxtrot zařízení aplikován zvolený filtr
- **RegExp Pattern** - umožňuje definovat regularní výraz, který bude aplikován jako filtr názvu proměnných při načítání iFoxtrot zařízení
- **Log** - log soubor se záznamem aktivity Homebridge serveru a pluginu Tecomat Platform

Stikem tlačítka *Submit* se Homebridge restartuje s aktuálně nastavenými parametry. Celý software je zinicializován a připraven k použití, jakmile je zobrazen QR kód.

9.3 Konfigurace PLCComS serveru

Pro správnou funkci *teco-homebridge* je nutná existence alespoň jedné instance serveru PLCComS s nastaveným sledováním public souboru *iFoxtrot.pub*. Doplněk *homebridge-platform-tecomat* pro *Homebridge* pak musí být nakonfigurován pro čtení dat z tohoto serveru.

Pro systémy Foxtrot 2 je možné server spustit přímo na samotném zařízení. Konfigurace serveru je dostupná přes konfigurační web stránky PLC.



9.4 Sestavení PLC programu s prvky knihovny iControlLib

Vybrané funkční bloky z knihovny iControlLib mohou významem své funkce emulovat jednotlivé prvky inteligentní domácnosti a díky Homebridge mohou být ovládány přes rozhraní Apple HomeKit.

Podporované funkční bloky s vazbou na HomeKit:

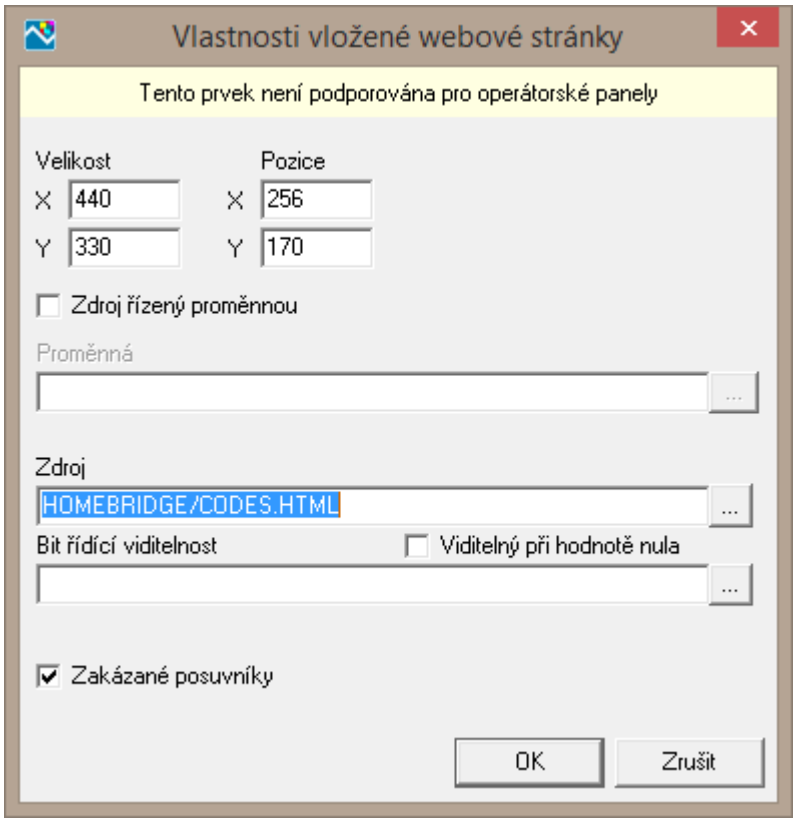
Funkční blok	HomeKit zařízení
fb_iButton	bezstavový programovatelný spínač (StatelessProgrammableSwitch)
fb_iContact	spínací/rozpínací kontakt (ContactSensor)
fb_iDimmer	světlo s funkcí zapnutí/vypnutí, stmívání, RGB (LightBulb)
fb_iDimmerLED	
fb_iDimmerRGB	
fb_iLight	
fb_iFan	ventilátor s funkcí zapnutí/vypnutí (Fan)
fb_iJalousie	rolety (WindowCovering)
fb_iOpener	univerzální ovladač otevírání (WindowCovering, Door, GarageDoorOpener)
fb_iRelay	univerzální vypínač (Switch)
fb_iSensorTemp	senzor teploty (TemperatureSensor) alternativně lze využít funkční block **fb_iDisplay*** s nastaveným vstupním parametrem *symbol* na hodnotu 100
fb_iSensorHumidity	senzor vlhkosti (HumiditySensor) alternativně lze využít funkční block **fb_iDisplay*** s nastaveným vstupním parametrem *symbol* na hodnotu 101
fb_iSensorCO2	detektor oxidu uhličitého (CarbonDioxideSensor) alternativně lze využít funkční block **fb_iDisplay*** s nastaveným vstupním parametrem *symbol* na hodnotu 104
fb_iSensorCO	detektor oxidu uhelnatého (CarbonMonoxideSensor) alternativně lze využít funkční block **fb_iDisplay*** s nastaveným vstupním parametrem *symbol* na hodnotu 105
fb_iSensorSmoke	detektor kouře (SmokeSensor) alternativně lze využít funkční block **fb_iDisplay*** s nastaveným vstupním parametrem *symbol* na hodnotu 106
fb_iSensorLight	senzor osvětlení (LightSensor) alternativně lze využít funkční block **fb_iDisplay*** s nastaveným vstupním parametrem *symbol* na hodnotu 107
fb_iSensorPIR	pohybový senzor (MotionSensor)
fb_iSocket	zásuvka (Outlet)
fb_iThermostat_cool	termostat s funkcí topení/chlazení (Thermostat)
fb_iThermostat_heat	
fb_iThermostat_heatCool	

Všechny instance těchto funkčních bloků v PLC programu jsou pomocí Homebridge pluginu *homebridge-platform-tecomat* automaticky detekovány skrze komuni-

kační server PLCComS a prezentovány v aplikaci HomeKit. V případě změny PLC programu není nutné Homebridge restartovat. Změny jsou doplňkem automaticky detekovány a v případě změn je výčet zařízení v HomeKit aktualizován.

Důležitou součástí systému Homebridge pro následné párování do HomeKit je vygenerovaný QR kód. Ten je automaticky zobrazen v konfigurační stránce balíčku *teco-homebridge*. Jelikož koncový uživatel nemusí disponovat právem přístupu do konfiguračních stránek PLC, nástroj *teco-homebridge* automaticky generuje QR kód ve formě SVG obrázku včetně předpřipravené HTML stránky, usnadňující jeho načtení, do umístění aplikačního webserveru PLC. Programátor pak může uživateli QR kód zpřístupnit pomocí nástroje WebMaker v prostředí Mosaic použitím prvku *Vložená webová stránka* (iframe). Zdroj vložené stránky pak stačí nastavit na cestu:

HOMEBRIDGE/CODES.HTML



Ve webové stránce, do které byl vložen tento prvek, bude po spuštění Homebridge zobrazen QR kód v této podobě:

9.5 Registrace do Apple HomeKit

Pro připojení zařízení generovaných PLC systémem Foxtrot do systému HomeKit je třeba použít zařízení s iOS a aplikací Domácnost. Je důležité se ujistit, že iOS zařízení je připojeno ke stejné síti jako ta, ke které je připojen síťový adaptér, na kterém byla dostupnost služby Homebridge nakonfigurována v PLC (viz. *Konfigurace balíčku teco-homebridge*).

Po spuštění aplikace Domácnost zvolíme nabídku *Přidat příslušenství*. Zobrazí se dialog s možností naskenovat QR kód. Naskenujeme fotoaparátem kód vygenerovaný nástrojem Homebridge. Pokud iOS zařízení není schopno naskenovat QR kód, lze zadat ručně PIN (*XXX-XX-XXX*), který je uváděn společně s QR kódem. Při úspěšné detekci zařízení v síti je v případě Homebridge zobrazeno upozornění, že se jedná o pokus o přidání necertifikovaného příslušenství. Potvrdíme volbou *Přesto přidat*. Následuje výpis všech detekovaných zařízení emulovaných funkčními bloky iControlLib, které lze následně roztřídit. Tím je celý proces registrace dokončen.

Názvy zařízení v HomeKit, respektive instancí funkčních bloků, jsou dány proměnnou *name*, která je standardní součástí vstupního rozhraní všech funkčních bloků z iControlLib. Pokud tato proměnná není u dané instance funkčního bloku v programu PLC nastavena, je pro zobrazení v HomeKit použit název proměnné z deklarace instance funkčního bloku. Názvy lze po načtení libovolně upravit v aplikaci Domácnost.

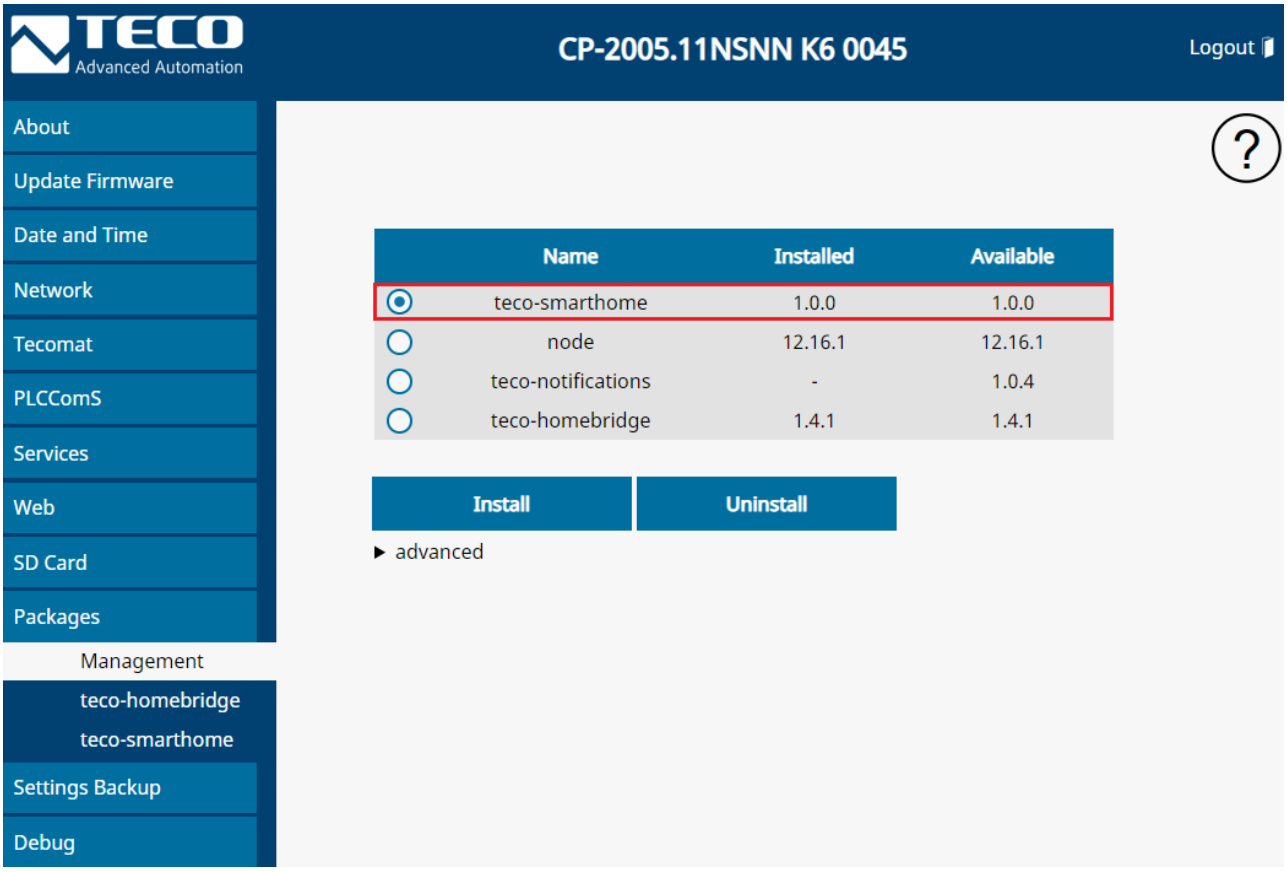
10 PODPORA DALŠÍCH PLATFOREM

Integrace PLC systému Tecomat jako prvku pro řízení inteligentní domácnosti do různých uživatelských platforem je primárně založena na použití komunikačního serveru PLCComS ((https://www.tecomat.cz/ke-stazeni/software/plccoms/). Tento server poskytuje TCP/IP spojení mezi klientem a PLC. Aplikační data, která jsou z PLC programu publikována (viz. kapitola 8), je pak možné komunikovat pomocí jednoduchého textově orientovaného protokolu.

I přesto, že je komunikace s PLCComS založena na jednoduchém protokolu, může někdy nevyhovovat současným potřebám a návykům při implementaci uživatelských aplikací. Systémy **Foxtrot 2** tak nabízejí volitelný sw doplněk (balíček) *teco-smarthome*, který umožňuje rozšíření možností komunikace pomocí websocketu s několika subprotokoly, založenými na JSON formátu.

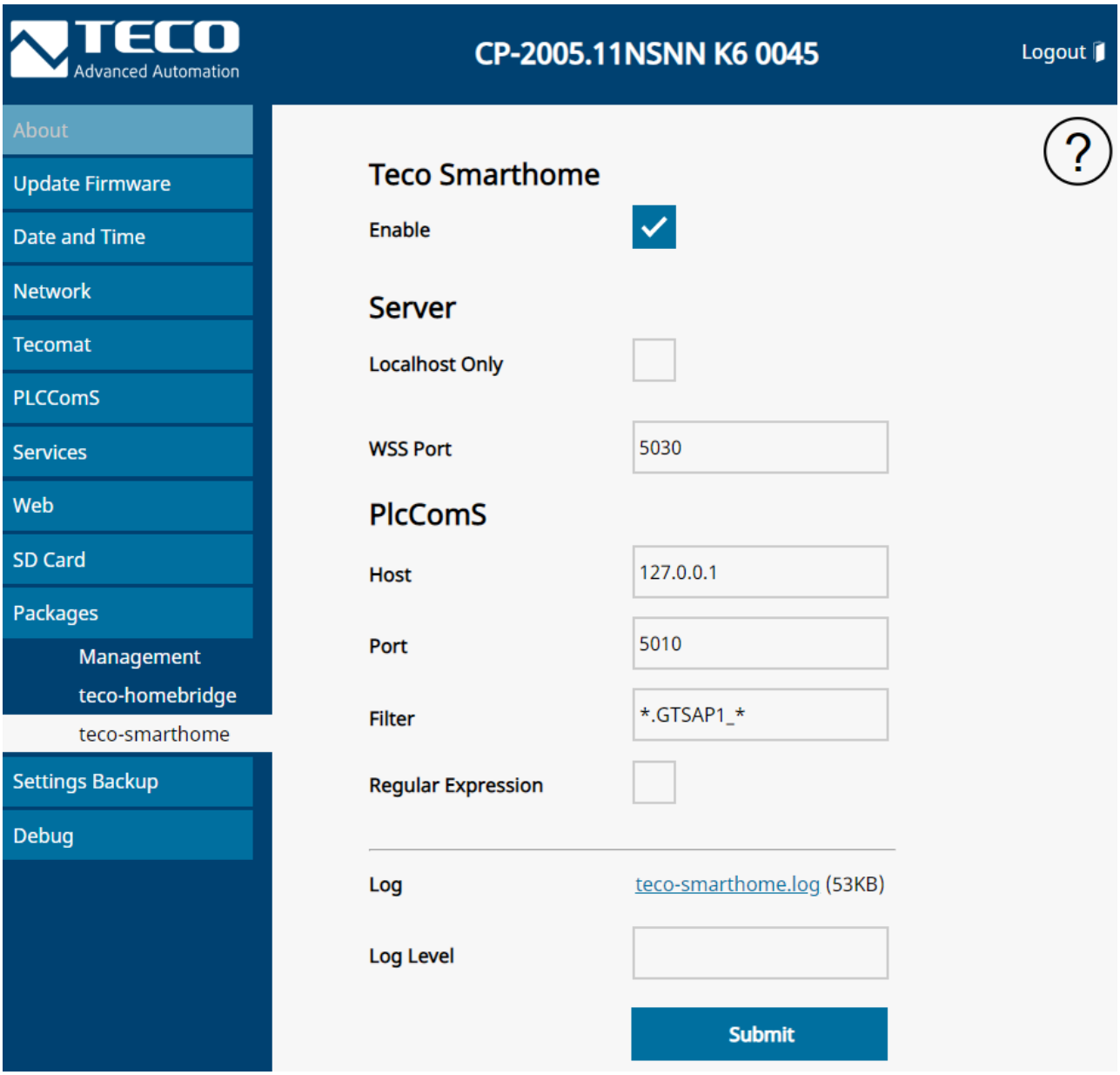
10.1 Balíček teco-smarthome

Balíček *teco-smarthome* není standardní součástí softwarové výbavy PLC Foxtrot 2. Je možné ho instalovat pomocí instalátoru balíčků, který je dostupný přes konfigurační web stránky zařízení. Instalace se provede přes sekci *Packages/Management* výběrem daného balíčku a stiskem tlačítka *Install*. Pro dokončení instalace je nutné provést restart PLC.



10.2 Konfigurace balíčku teco-smarthome

Po úspěšné instalaci balíčku a restartu PLC by se v konfigurační sekci *Packages* měla zobrazit nová položka *teco-smarthome.* Po výběru této položky se zobrazí web stránka s nastavením balíčku.



Ve výchozím stavu je sw deaktivován. Pokud ho chceme povolit, je nutné zatrhnout položku *Enable*. Tím dojde k rozbalení dalších konfiguračních parametrů.

Balíček *teco-smarthome* reprezentuje websocket server. Jeho služby mohou využívat další sw bežící na stejném PLC zařízení. Pokud chceme k serveru přistupovat i z lokální sítě, je nutné tuto možnost povolit. To je možné provést odškrtnutím zátržítka *Localhost Only*, které je ve výchozím stavu aktivní. Pak je možné nastavit veřejný port (*WSS Port*), na kterém bude šifrované websocket spojení dostupné.

Zdrojem dat pro websocket server je běžící instance PLCComS. Parametry připojení k PLCComS se definují ve stejnojmenné sekci konfigurace. Pro připojení je nutné zadat IP adresu (*Host*) a port (*Port*) serveru PLCComS. Výchozí parametry připojení jsou

nastaveny na instanci serveru běžícím na stejném PLC zařízení, tedy *localhost* (127.0.0.1) s portem 5010. Komunikaci je samozřejmě možné namířit i na server běžící mimo zařízení. Pro vyšší efektivitu je vhodné nastavit filtr (*Filter*) proměnných, se kterými se bude přes komunikační server PLCComS pracovat. Formát zápisu filtru odpovídá dokumentaci k PLCComS (https://www.tecomat.cz/ke-stazeni/software/plccoms/). Jelikož je *teco-smarthome* server určen především k účelům z oblasti chytrých domů a domácností (smarthome), je výchozí hodnotou filtru řetězec **.GTSAP1_**, který filtruje pouze proměnné, které jsou generovány knihovnou *iControlLib*. Filtr je možné zapsat i pomocí regulárního výrazu. Takovou skutečnost je nutné dát na vědomí zaškrtnutím volby *Regular Expression*.

Pro možnost ladění je zpřístupněn log soubor, do kterého jsou zapisovány informace dle nastavitelné logovací úrovně (*Log Level)*.

Všechny změny provedené v nastavení je nutné potvrdit tlačítkem *Submit*. Tím dojde k uložení všech parametrů a restartování websocket serveru.

10.3 Veřejné komunikační subprotokoly

S komunikačním serverem lze z klientské strany komunikovat několika textově orientovanými subprotokoly v JSON formátu. Volba komunikačního subprotokolu se provádí standardním způsobem dle websocket specifikace, tedy nastavením hlavičky v úvodním HTTP požadavku navazujícím websocket komunikaci:

```
GET / HTTP/1.1
Host: 192.168.134.178:5030
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Protocol: <protocol>
…
```

Při navazování spojení, může server odpovědět HTTP chybovým kódem *503 Service Unavailable*. V takovém případě server pravděpodobně připravuje spojení s PLCComS a inicializuje data. Klient by se měl pokusit o připojení později.

Po vytvoření websocket spojení může klient začít komunikovat se serverem dle zvoleného subprotokolu. Pokud server v průběhu komunikace vyhodnotí některou z přijatých zpráv jako neplatnou vzhledem ke zvolenému protokolu, ukončí s klientem spojení.

10.3.1 Subprotokol raw

Jedná se o nejjednodušší a nejuniverzálnější protokol, jehož cílem je se co nejblíže přiblížit výsledkům získávaných přímou komunikací s PLCComS, a to ve formě JSON formátu. Server drží automaticky aktualizovaný obraz proměnných získaný z PLCComS, který je případně filtrovaný nastaveným filtrem dle konfigurace.

Protokol definuje tyto typy zpráv:

Získání seznamu publikovaných proměnných:

Dotaz:

```
{
  "cmd": "list"
}
```

Příklad odpovědi:

```
{
  "list": {
    "MAIN.SOCKET.GTSAP1_SOCKET_ONOFF": {
      "dt": "BOOL",
      "val": false
    },
    "MAIN.SOCKET.GTSAP1_SOCKET_ENABLE": {
      "dt": "BOOL",
      "val": true
    },
    "MAIN.SOCKET.GTSAP1_SOCKET_NEEDACK": {
      "dt": "BOOL",
      "val": true
    },
    "MAIN.SOCKET.GTSAP1_SOCKET_NAME": {
      "dt": "STRING",
      "val": "Test Socket"
    }
  }
}
```

Na zaslaný dotaz ze strany klienta server odpoví objektem *list*. Klíče objektu reprezentují seznam publikovaných proměnných. Každá položka má pak podobu struktury s následujícími atributy:

- `"dt"` – datový typ proměnné
- `"val"` – aktuální hodnota proměnné

Získání vybraného setu proměnných:

Dotaz:

```
{
  "cmd": "get",
  "data": ["MAIN.SOCKET.GTSAP1_SOCKET_ONOFF"]
}
```

Doplňkovým parametrem *get* požadavku je atribut *data*, který je reprezentovaný polem řetězců s identifikátory proměnných, jejichž stav je dotazován.

Příklad odpovědi:

```
{
  "get": {
    "MAIN.SOCKET.GTSAP1_SOCKET_ONOFF": {
      "dt": "BOOL",
      "val": false
    }
  }
}
```

Na zaslaný dotaz ze strany klienta server odpoví objektem *get*. Klíče objektu reprezentují seznam dotazovaných proměnných. Každá položka má pak podobu struktury s následujícími atributy:

- "dt" – datový typ proměnné
- "val" – aktuální hodnota proměnné

Nastavení hodnoty proměnné:

Dotaz:

```
{
  "cmd": "set",
  "data": {
    "MAIN.SOCKET.GTSAP1_SOCKET_ONOFF": {
      "dt": "BOOL",
      "val": true
    }
  }
}
```

Doplňkovým parametrem *set* požadavku je atribut *data*, který je reprezentovaný objektem. Klíče objektu jsou seznamem identifikátorů proměnných, jejichž hodnoty chceme nastavit. Každá položka má pak podobu struktury s následujícími atributy:

- "dt" – datový typ proměnné
- "val" – hodnota proměnné k nastavení

Na požadavek *set* server nijak neodpovídá. Úspěšné nastavení se v případě změny hodnoty proměnné projeví zasláním zprávy *diff* ze strany serveru.

Reportování změny hodnoty proměnné

Pokud dojde ke změně hodnoty proměnné, například vlivem aplikačního programu PLC, je tato skutečnost reportována všem připojeným klientům pomocí zprávy *diff.* Tyto zprávy jsou klientovi zasílány automaticky po navázání spojení se serverem.

Příklad vysílané zprávy *diff*:

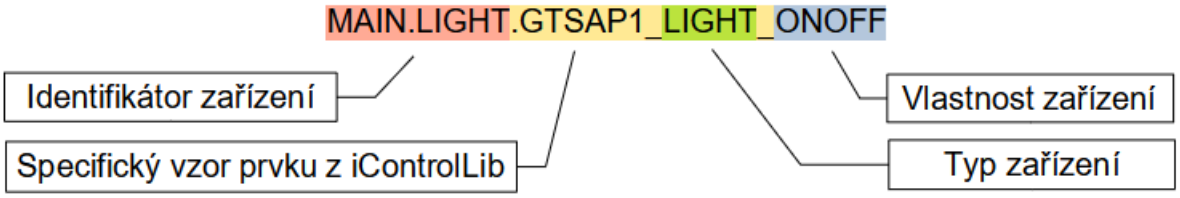
```
{
  "diff": {
    "MAIN.SOCKET.GTSAP1_SOCKET_ONOFF": {
      "dt": "BOOL",
      "val": false
    }
  }
}
```

Zpráva obsahuje objekt *diff*. Klíče objektu reprezentují seznam proměnných, jejichž hodnota se změnila. Každá položka má pak podobu struktury s následujícími atributy:

- "dt" – datový typ proměnné
- "val" – aktuální hodnota proměnné

10.3.2 Subprotokol devs

Tento protokol je více specializovaný, a to zejména na použití společně s knihovnou iControlLib v oblasti chytrých domů a domácností (smarthome). Server drží automaticky aktualizovaný obraz proměnných získaný z PLCComS, který je případně filtrovaný nastaveným filtrem dle konfigurace. Dále jsou zpracovávány jen ty proměnné, jejichž název splňuje specifický vzor:



Tento vzor splňuje většina prvků z iControlLib knihovny. Takto vytříděné proměnné jsou pak seskupeny do struktur se společným identifikátorem a typem zařízení. Vzniká tak obraz celého zařízení deklarovaného v aplikačním programu PLC.

Protokol definuje tyto typy zpráv:

Získání seznamu zařízení:

Dotaz:

```
{
  "intent": "list"
}
```

Příklad odpovědi:

```
{
  "list": [
    {
      "id": "MAIN.LIGHT",
      "type": "LIGHT",
      "props": {
        "NEEDACK": {
          "dt": "BOOL",
          "val": false
        },
        "NAME": {
          "dt": "STRING",
          "val": "Test Light"
        },
        "TYPE": {
          "dt": "BOOL",
          "val": false
        },
        "ENABLE": {
          "dt": "BOOL",
          "val": true
        },
        "ONOFF": {
          "dt": "BOOL",
          "val": false
        }
```

```
      }
    },
    {
      "id": "MAIN.SOCKET",
      "type": "SOCKET",
      "props": {
        "NEEDACK": {
          "dt": "BOOL",
          "val": true
        },
        "NAME": {
          "dt": "STRING",
          "val": "Test Socket"
        },
        "ONOFF": {
          "dt": "BOOL",
          "val": false
        },
        "ENABLE": {
          "dt": "BOOL",
          "val": true
        }
      }
    }
  ]
}
```

Na zaslaný dotaz ze strany klienta server odpoví polem *list*. Prvky pole tvoří objekty, které reprezentují jednotlivá zařízení. Každý objekt zařízení pak obsahuje tyto parametry:

- `"id"` – identifikátor zařízení
- `"type"` – typ zařízení
- `"props"` – seznam vlastností zařízení

Seznam vlastností zařízení je tvořen objektem, jehož klíče tvoří názvy vlastností. Každá položka vlastnosti má pak podobu struktury s následujícími atributy:

- `"dt"` – datový typ proměnné
- `"val"` – aktuální hodnota proměnné

Získání vybraného setu zařízení:

Dotaz:

```
{
  "intent": "get",
  "payload": ["MAIN.SOCKET"]
}
```

Doplňkovým parametrem *get* požadavku je atribut *payload*, který je reprezentovaný polem řetězců s identifikátory zařízení, jejichž stav je dotazován.

Příklad odpovědi:

```
{
  "get": [
```

```
    {
      "id": "MAIN.SOCKET",
      "type": "SOCKET",
      "props": {
        "NEEDACK": {
          "dt": "BOOL",
          "val": true
        },
        "ENABLE": {
          "dt": "BOOL",
          "val": true
        },
        "NAME": {
          "dt": "STRING",
          "val": "Test Socket"
        },
        "ONOFF": {
          "dt": "BOOL",
          "val": false
        }
      }
    }
  ]
}
```

Na zaslaný dotaz ze strany klienta server odpoví polem *get*. Prvky pole tvoří objekty, které reprezentují dotazovaná zařízení. Každý objekt zařízení pak obsahuje tyto parametry:

- `"id"` – identifikátor zařízení
- `"type"` – typ zařízení
- `"props"` – seznam vlastností zařízení

Seznam vlastností zařízení je tvořen objektem, jehož klíče tvoří názvy vlastností. Každá položka vlastnosti má pak podobu struktury s následujícími atributy:

- `"dt"` – datový typ proměnné
- `"val"` – aktuální hodnota proměnné

Nastavení hodnoty vlastnosti zařízení:

Dotaz:

```
{
  "intent": "set",
  "payload": [
    {
      "id": "MAIN.SOCKET",
      "props": {
        "ONOFF": {
                "dt": "BOOL",
          "val": true
        }
      }
    }
  ]
}
```

Doplňkovým parametrem *set* požadavku je atribut *payload*, který je reprezentovaný polem. Prvky pole tvoří objekty zařízení, jejichž vlastnosti chceme nastavit. Každý objekt zařízení pak obsahuje tyto parametry:

- `"id"` – identifikátor zařízení
- `"props"` – seznam vlastností zařízení k nastavení

Seznam vlastností zařízení je tvořen objektem, jehož klíče tvoří názvy vlastností, které chceme nastavit. Každá položka vlastnosti má pak podobu struktury s následujícími atributy:

- `"dt"` – datový typ proměnné
- `"val"` – hodnota proměnné k nastavení

Na požadavek *set* server nijak neodpovídá. Úspěšné nastavení se v případě změny hodnoty vlastnosti projeví zasláním zprávy *diff* ze strany serveru.

Reportování změny vlastností zařízení

Pokud dojde ke změně některé z vlastností zařízení, například vlivem aplikačního programu PLC, je tato skutečnost reportována všem připojeným klientům pomocí zprávy *diff.* Tyto zprávy jsou klientovi zasílány automaticky po navázání spojení se serverem.

Příklad vysílané zprávy *diff*:

```
{
  "diff": [
    {
      "id": "MAIN.SOCKET",
      "type": "SOCKET",
      "props": {
        "ENABLE": {
          "dt": "BOOL",
          "val": true
        },
        "ONOFF": {
          "dt": "BOOL",
          "val": true
        },
        "NAME": {
          "dt": "STRING",
          "val": "Test Socket"
        },
        "NEEDACK": {
          "dt": "BOOL",
          "val": true
        }
      },
      "changed": [
        "ONOFF"
      ]
    }
  ]
}
```

Zpráva obsahuje pole *diff*. Prvky pole tvoří objekty, které reprezentují zařízení, jejichž vlastnosti se změnily. Každý objekt zařízení pak obsahuje tyto parametry:

- `"id"` – identifikátor zařízení
- `"type"` – typ zařízení
- `"props"` – seznam vlastností zařízení

Seznam vlastností zařízení je tvořen objektem, jehož klíče tvoří názvy vlastností. Každá položka vlastnosti má pak podobu struktury s následujícími atributy:

- `"dt"` – datový typ proměnné
- `"val"` – aktuální hodnota proměnné

Ve zprávě *diff* se vždy posílá celý obraz zařízení, tedy se všemi vlastnostmi, i když se změnila třeba jen jedna hodnota z vlastností. Pro identifikaci změny struktura zařízení obsahuje navíc pole `"changed"`, které obsahuje jmenný výčet vlastností, u kterých došlo ke změně.

Zasílání *diff* zpráv je založeno na principu PLCComS, který posílá stejnojmenné zprávy klientovi (v tomto případě náš server) v případě, že dojde ke změně hodnoty některé sledované proměnné. Pokud v jednom okamžiku (respektive v jednom komunikačním cyklu PLCComS) dojde ke změně hodnot více proměnných, PLCComS odvysílá změnovou zprávu postupně pro každou z nich. Server *teco-smarthome* by pak každou takto přijatou zprávu interpretoval adekvátně k protokolu *devs*. Pokud by se vícenásobná změna hodnot proměnných týkala jednoho definovaného zařízení, tedy jeho vlastností, server by vygeneroval *diff* zprávu pro každou změněnou vlastnost zvlášť. Pole `"changed"` by tedy v každé zprávě obsahovalo pouze jednu položku. To může být v některých situacích nežádoucí, ať už vzhledem ke zvýšenému datovému provozu v komunikaci, nebo pro další zpracování, kdy je třeba znát ustálený obraz zařízení po dokončení všech změn. Server proto po přijetí změnové zprávy z PLCComS určitou dobu vyčká s vysláním *diff* zprávy pro identifikované zařízení, aby bylo možné případné další přijaté změny agregovat do jedné zprávy.