Uživatelská příručka SFC Editoru

Revize 1
Plzeň
22.11.2022

Obsah

1 Úvod

Editor **S**ekvenčních **F**unkčních **G**rafů (SFC Editor) je grafické vývojové prostředí, které usnadňuje návrh řídicích algoritmů pro sekvenční logické automaty. Tento nástroj implementuje jeden z pěti jazyků určených pro programování PLC, které jsou definovany v normě `IEC 61131-3` [1], a to grafický formalismus (jazyk) SFC. Editor nabízí uživatelsky velice přívětivé prostředí, které umožňuje uložení, otevření nebo tisk sestrojeného SFC schématu. Kromě těchto standardních funkcí má uživatel možnost využít plně vektorového chování konstruovaného schématu nebo export schématu do vektorového formátu *SVG*. Klíčovou funkčností, kterou SFC Editor nabízí je překlad SFC schématu do tabulky přechodů, která realizuje sestavený algoritmus sekvenčního řízení.

SFC Editor je vyvinut pro platformu Windows x86 (x64). Pro svůj běh vyžaduje .Net Framework verze 2.0 nebo vyšší.

Pro práci v editoru je nutná alespoň základní znalost jazyka sekvenčních funkčních grafů. Proto je v následující kapitole uveden stručný úvod do této problematiky.

Protože je editor primárně vyvinut jako nástroj pro podporu bloků `ATMT` a `EATMT`, je nutná znalost možností a omezení těchto bloků. Popis omezení a nároků bloků `ATMT` a `EATMT` ve vztahu k jazyku SFC je uveden v kapitole 3.

Prostředí editoru je popsáno v kapitole 4. Postup, jak vytvořit a editovat schéma je uveden v kapitolách 6 a 7. Způsob překladu sestaveného SFC schématu je naznačen v kapitole 8. Kapitola 10 se zabývá možnostmi vizualizace vývoje stavu sekvenčního automatu, který realizuje vygenerovanou tabulku přechodů. Kompletní příklad zkonstruování SFC schématu je v kapitole 11. Poslední kapitola je věnována způsobu začlenění SFC Editoru do nadřazeného systému.

2 Stručný úvod do SFC

Metoda SFC není nijak nová metoda návrhu řídících algoritmů, jedná se v podstatě pouze o přejmenovanou metodu GRAFCET.

GRAFCET byl vyvinut týmem francouzských vědců na univerzitě v Grenoblu už na začátku 70. let. Název je zkratkou GRaph a Association Francaise de Cybernétique Et Technique. Metoda vycházela ze znalosti Petriho sítí[1]. Metoda však nebyla v době vzniku přijata širší komunitou a na vědomí byla vzata až v 90. letech jako metoda sekvenčních funkčních grafů (SFC).

Následující podkapitoly se budou věnovat popisu grafického formalismu (jazyka) SFC.

[1] Petriho síť je matematická reprezentace diskrétních distribuovaných systémů v podobě orientovaného bipartitního grafu s ohodnocením.

2.1 Základní prvky jazyka SFC

SFC rozlišuje tyto základní prvky:

- Hrana
- Krok
- Přechod

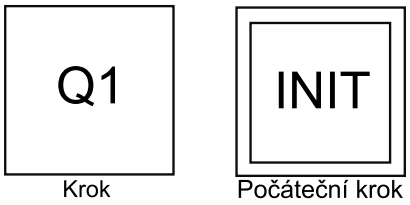
2.1.1 Hrana

Hrana je v SFC chápána jako spojnice kroku a přechodu nebo přechodu a kroku. Jedná se vždy o orientovanou hranu, která je označena šipkou jen v případě, jde-li zdola nahoru. Hrany se mohou lámat a to s respektováním určitých pravidel, které nedovolují jiné než pravoúhlé zlomy. Orientovanost hran umožňuje realizovat v diagramu cykly.

2.1.2 Kroky

SFC rozlišuje dva druhy kroku:

- Krok – graficky znázorněný jako čtverec (obr. 1).
- Počáteční (inicializační) krok - graficky znázorněný jako dvojitý čtverec (obr. 1).



Obrázek 1: Druhy kroků

Uvnitř čtverce, který reprezentuje krok, je vykresleno jeho jméno.

Každému kroku může být přiřazena jedna nebo více akcí. Akce jsou vykresleny jako samostatné obdélníky, které obsahující další informace a jsou ke kroku připojeny jednoduchou čárou.

Každý, krok je buď v aktivním nebo neaktivním stavu. Počáteční krok je krok, ve kterém algoritmus začíná.

2.1.3 Přechod

Přechod je graficky reprezentován jako silná vodorovná čára obr. 2. Přechod nemůže být přímo spojen s dalším přechodem. Každý přechod má svoji podmínku přechodu.

Podmínka přechodu je rozhodovací práh *R(i)*, kde *i* je přechod. Podmínka je funkcí

vstupních parametrů nebo vnitřních stavů SFC.

Stavová proměnná kroku je logická proměnná *Xi*, která nabývá hodnoty *true*, pokud je krok v aktivním stavu.

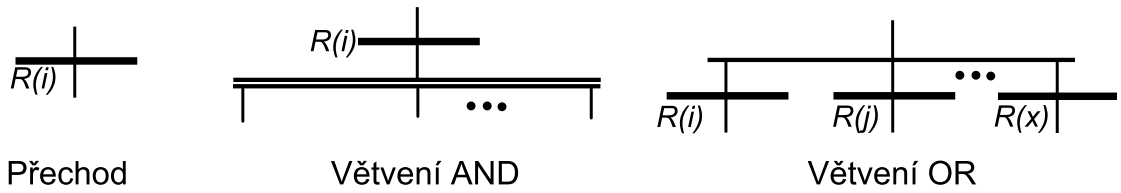
SFC dovoluje větvení diagramu a to dvojího typu AND a OR. Tato větvení se dají chápat jako specifický druh přechodu (z jednoho kroku do více kroků).

- Větvení (divergence) AND

  Graficky je tento typ větvení diagramu znázorňován jako dvojitá vodorovná čára, které bezprostředně předchází přechod obr. 2. Obecně má toto větvení neomezený počet větví. Větvení tohoto typu umožňuje jednou podmínkou (receptivou) aktivovat více kroků (resp. spustit více akcí), které jsou k sobě paralelně řazené. V případě aktivity bezprostředního předchůdce x přechodu a za předpokladu splnění podmínky přechodu *R(x)* se příznak aktivity kroku rozpadne na tolik aktivních kroků (analogie k tokenům v Petriho sítích), na kolik větví se diagram v příslušném přechodu větví. Bezprostřední následníci větvení se aktivují současně.

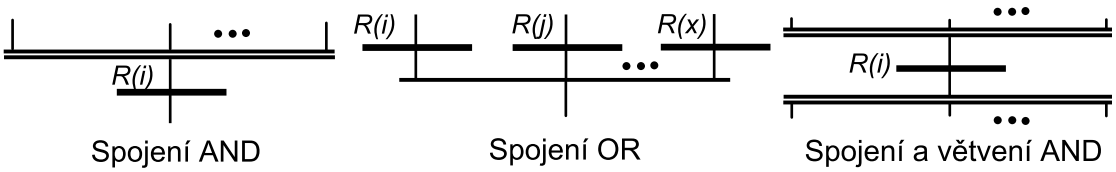
- Větvení (divergence) OR

  Gaficky se větvení znázorňuje jednoduchou vodorovnou čarou, za kterou následuje obecně libovolný počet přechodů obr. 2. Princip fungování by se dal přirovnat ke konstrukci IF R(i) THEN ... ELSEIF R(j) THEN ... ELSEIF R(x)... . Algoritmus pokračuje tím krokem, kterému předchází splněná podmínka příslušného přechodu.



Obrázek 2: Přechod a větvení diagramu

Větvení OR i AND mají své opaky obr. 3, tedy spojení (konvergenci) OR a AND. Jejich princip je patrný: z více větví diagramu slučují běh do jedné. U spojení typu AND, kde jsou paralelní větve resp. kroky v paralelních větvích aktivovány současně, musí i současně přejít do jedné větve. To v praxi vypadá tak, že přes spojení typu AND přejde algoritmus pouze tehdy, jsou-li všechny jeho bezprostřední předchůdci (kroky) v aktivním stavu a podmínka, která následuje za spojením AND je splněna. Z toho plyne i důvod použití konstrukce spojení AND a větvení AND obr. 3, takto sestrojená konstrukce funguje jako synchronizace obecně n kroků (akcí) a následnému aktivování dalších m kroků.



Obrázek 3: Spojení AND a OR

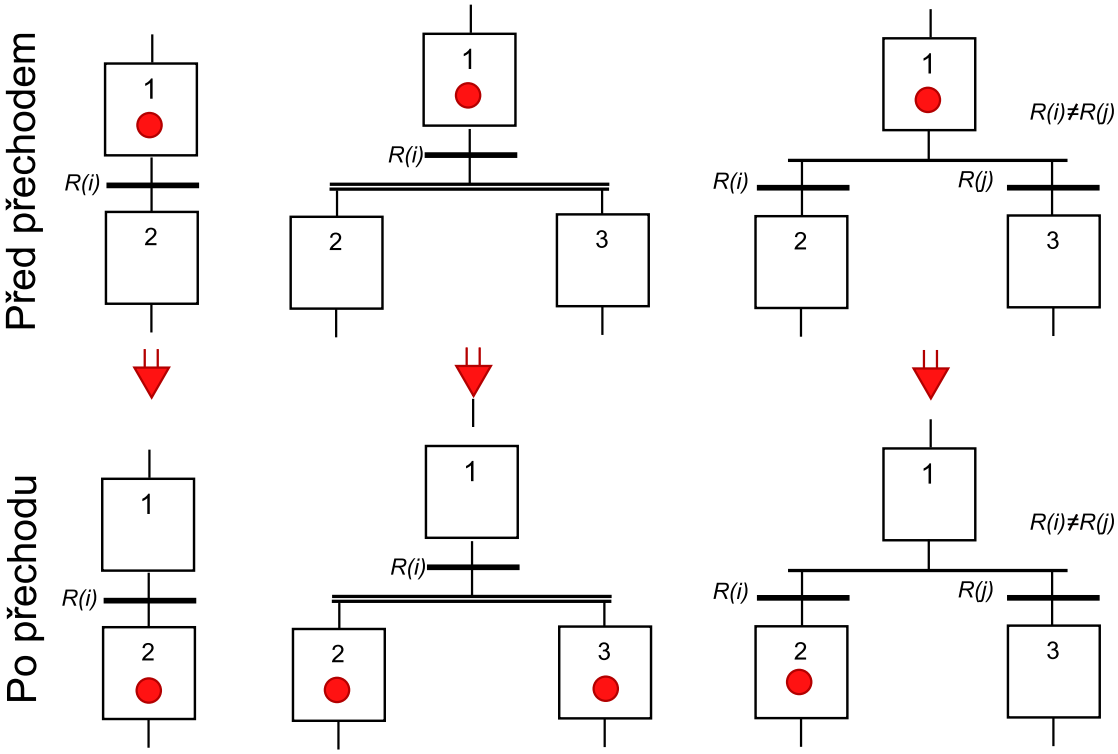
2.2 Vývoj stavu v SFC

Z obrázku 4 je patrný vývoj stavu v SFC schématu.

Pokud je krok 1 v aktivním stavu a následující podmínka $R(i)$ přechodu je splněna dochází k deaktivaci kroku 1 a k aktivaci následujícího kroku 2.

Na druhém příkladě je zobrazen vývoj stavu SFC schématu při použití rozvětvení (divergence) AND. Pokud je stejně jako v předchozím případě splněna podmínka přechodu $R(i)$ a krok 1 je v aktivním stavu, dojde k aktivaci následujících kroků 2 a 3 (dochází k rozvětvení algoritmu do dvou paralelně běžících větví).

Poslední uvedený příklad ukazuje příklad použití větvení typu OR. V případě aktivního stavu 1 a splnění podmínky $R(i)$ dojde k aktivaci kroku 2. Nutnost rozdílnosti podmínek přechodu *R(i)* a *R(j)* obr. 4 je uvedena z důvodu zamezení kolize přechodů.



Obrázek 4: Příklady pálení přechodu

Kolize přechodů

Ke kolizi přechodů může dojít pouze u větvení typu OR a to ve chvíli, kdy dvě nebo více podmínek bude mít logickou hodnotu *true*, to by mělo za následek nepředvídatelné chování algoritmu. Tato kolize přechodů se dá odstranit vzájemnou exkluzivitou podmínek. Metoda SFC neumí rozpoznat takovéto kolize sama, proto zůstává na tvůrci algoritmu dbát exkluzivity podmínek nebo problém řešit jiným způsobem.

3 Blok ATMT (EATMT) a SFC

Blok `ATMT`, stejně jako blok `EATMT`, realizuje konečný automat. Bloky se od sebe liší pouze počty stavů, kterých mohou nabývat a počty podmínek přechodu. Oba mají, ale několik společných omezení. Protože je SFC Editor navržen tak, aby podporoval tyto bloky jako cílové zařízení (bloky, kde se bude vygenerovaná tabulka přechodů realizovat) omezení a nároky bloků `ATMT` a `EATMT` ovlivňují i možnosti a funkce SFC Editoru.

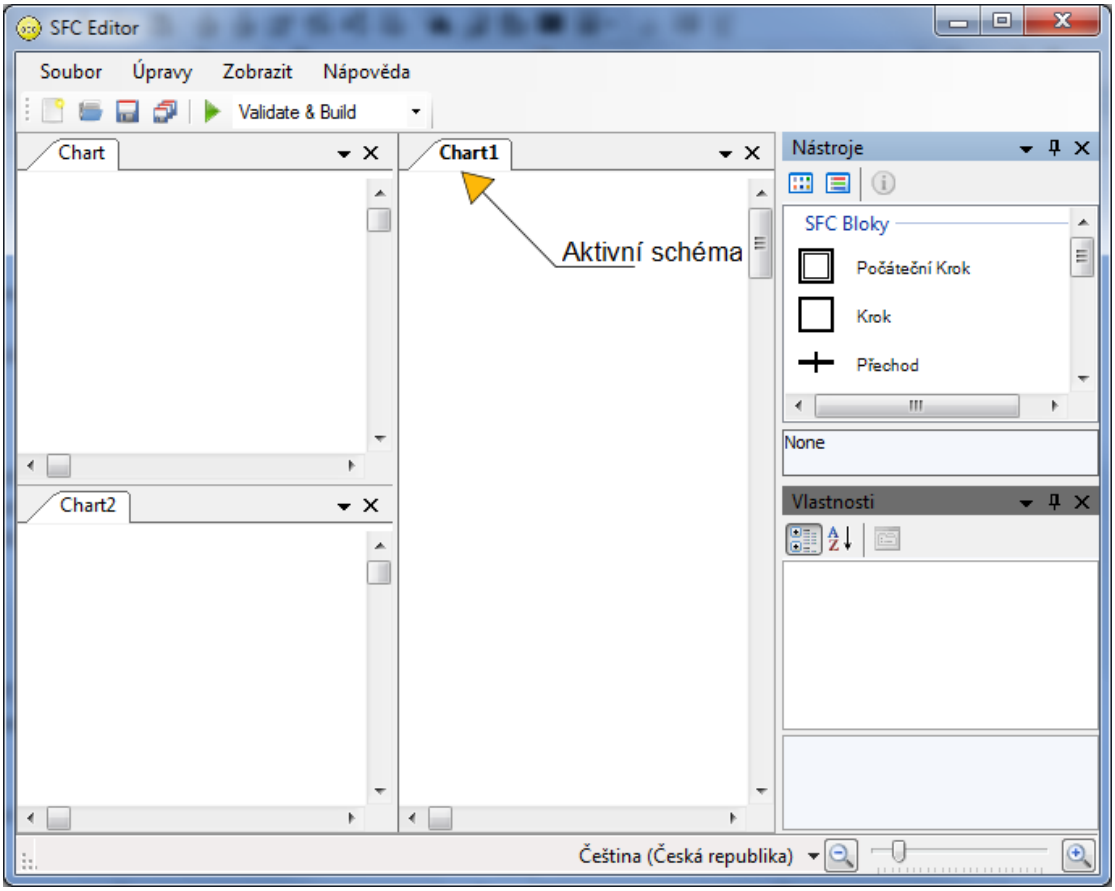
První omezení obou bloků (`ATMT`, `EATMT`) je skutečnost, že nedokáží realizovat paralelní běh algoritmu. To v SFC schématu odpovídá rozvětvení schématu v bloku AND divergence. Tento fakt se v editoru projeví tak, že pro tyto bloky nelze do schématu přidat AND divergenci a AND konvergenci.

Další omezení je, že vstupy bloků přímo korespondují s podmínkami přechodů v SFC schématu. To má za důsledek, že podmínky přechodu nelze ve schématu definovat jako složitější logický výraz. Blok `EATMT` umožňuje alespoň negaci logické hodnoty a to uvedením symbolu „!“ před Tag podmínky. Vstupy bloků `ATMT` a `EATMT` pak souvisí s bloky přechodu v SFC schématu následovně: u bloku `ATMT` odpovídá jméno vstupu Tagu přechodu v SFC schématu (je jich pouze 16). Blok `EATMT` má také „pouze“ 16 vstupů, ale nejsou logické, jako u `ATMT`, ale celočíselné a používá se spodních 16 bitů, kde každý bit představuje logickou hodnotu jedné podmínky přechodu ve schématu. Tato implementace nám dává 256 logických vstupů, které jsou mapovány na 16 vstupních pinů bloku `EATMT`. Ve schématu odpovídá druhý vstup bloku `EATMT` podmínce s Tagem c0.1 nebo c1. Poslední vstup bloku `EATMT` (256 - tý) pak odpovídá Tagu c15.15 nebo c255. Oba uvedené způsoby jsou dovoleny. V případě negace výrazu se před Tag uvede symbol „!“ (např. !c0.5).

Poslední omezení bloků `ATMT` a `EATMT` je obdobné jako předchozí omezení. Výstupy bloků jsou opět pouze logické a korespondují přímo s kroky (Step) v SFC schématu. Nelze tak ve schématu definovat akce, které se provedou po vstoupení algoritmu do stavu (kroku). Rozdíl mezi bloky `ATMT` a `EATMT` je opět pouze v počtu výstupů. `ATMT` jich má k dispozici 16, `EATMT` potom 256 (výstupy jsou celočíselné, kde se používá 16 spodních bitů). Toto omezení se v editoru projevuje tak, že není možné přidávat akce k blokům Step.

Rozšíření, které není součástí jazyka SFC popisovaného normou `IEC 61131-3` je možnost definovat u kroku (Step) *TimeOut*. Jedná se o reálné číslo, které udává po jaké době strávené v aktivním stavu bloku se nastaví výstup *tout* bloku `ATMT` nebo `EATMT` do logické jedničky. Výstup *tout* se nastaví do nuly po přechodu automatu do nového stavu.

Podrobnosti o vlastnostech bloků `ATMT` a `EATMT` naleznete v textu [2].

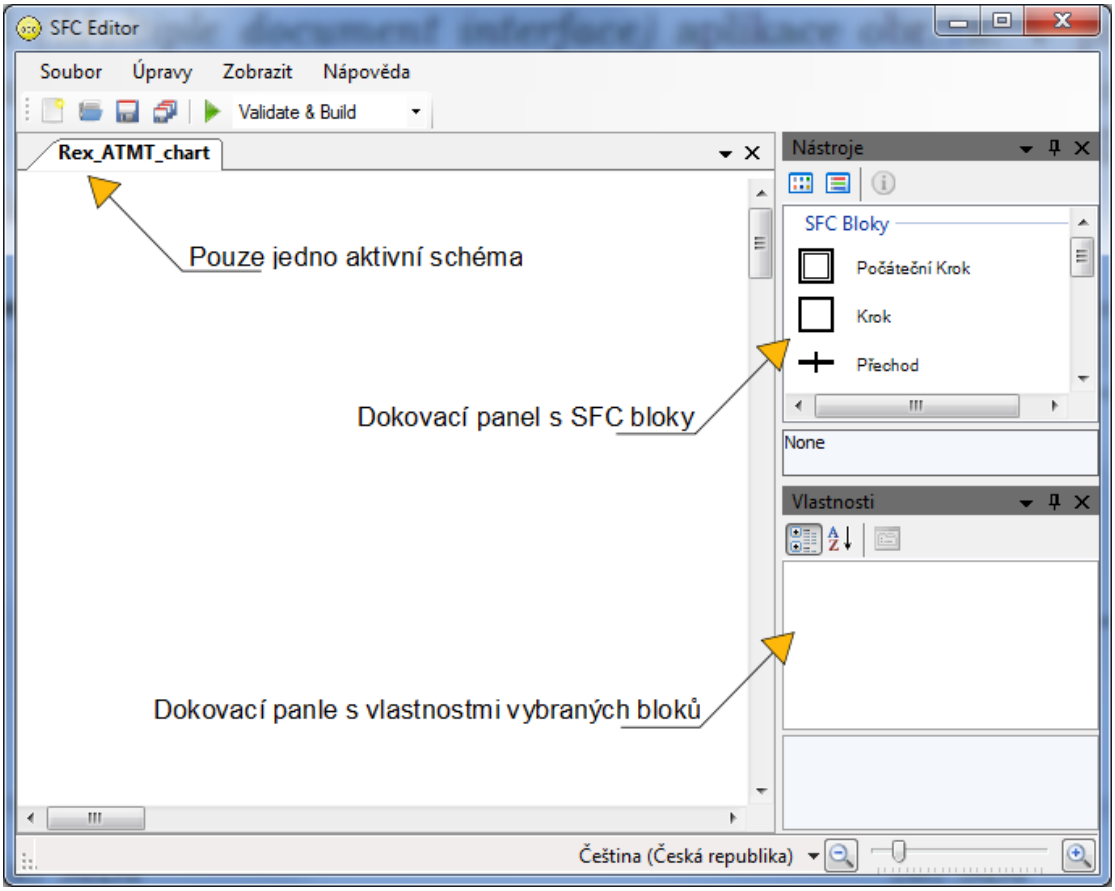


Obrázek 5: MDI prostředí SFC editoru

4 Prostředí editoru

Ve své nejnovější verzi 3.1 disponuje editor dvojím způsobem chování, které závisí na parametrech předaných editoru při spuštění. Pokud je editor spuštěn bez jakýchkoli parametrů, (prostým poklikáním na ikonu editoru), chová se jako MDI *(Multiple document interface)* aplikace obr. 5. V případě, že jsou editoru při spuštění předány správné parametry, (spuštěn z kontextu nadřazené aplikace), chová se jako SDI *(Single document interface)* aplikace obr. 6. Veškeré nabídky na vložení nebo editaci SFC bloků jsou umístěny v dokovacích panelech, které si může uživatel libovolně umístit obr. 7. Panely mohou být na pracovní ploše zobrazeny trvale, „připíchnuty“ nebo se mohou automaticky skrývat (dokovat). Jednotlivé panely se dají zavřít pomocí symbolu - „křížek“ a otevřít z nabídky menu `View`.

Pracovní plocha editoru umožňuje přiblížit resp. oddálit schéma (Zoom). Při určitém stupni přiblížení je na pracovní plochu vykreslena mřížka, která slouží ke snadnějšímu umístění (srovnání) bloků. Změna přiblížení se provádí rotací kolečka myši se stisknutou klávesou CTRL. Druhá možnost, jak změnit velikost přiblížení pracovní plochy, je



Obrázek 6: SDI prostředí SFC editoru

posuvník v pravém dolním rohu aplikace obr. 4.1.

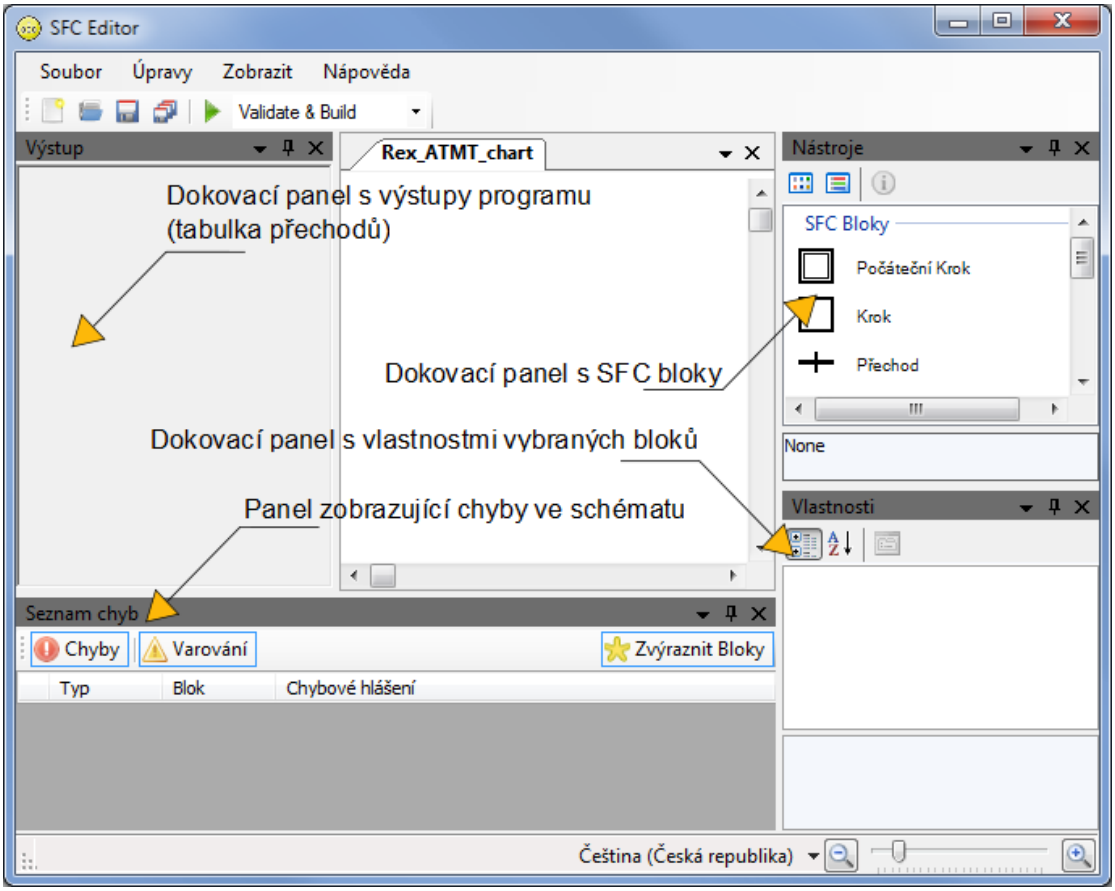
Posunutí pracovní plochy na požadovanou pozici je možné v rámci jejího rozsahu realizovat tažením za příslušnou postranní lištu okna nebo rotací kolečka myši - vertikální pohyb, resp. rotací kolečka se stisklou klávesou SHIFT - horizontální pohyb.

4.1 Lokalizace

Prostředí editoru podporuje jazykovou lokalizaci. V současné verzi jsou podpořeny tyto jazyky:

- Čeština (Česká Republika)
- Angličtina (Spojené státy)
- Němčina (Německo) - pouze uživatelské rozhraní

Změna jazyka se provádí sekvenčně po kliknutí na název aktuálního jazyka v pravém dolním rohu aplikace nebo libovolně po rozbalení nabídky jazyků (také pravý dolní roh



Obrázek 7: Všechny dokovací panely SFC editoru

aplikace) obr. 4.1. Změna jazykové lokalizace je okamžitá a nevyžaduje restart aplikace. [2]

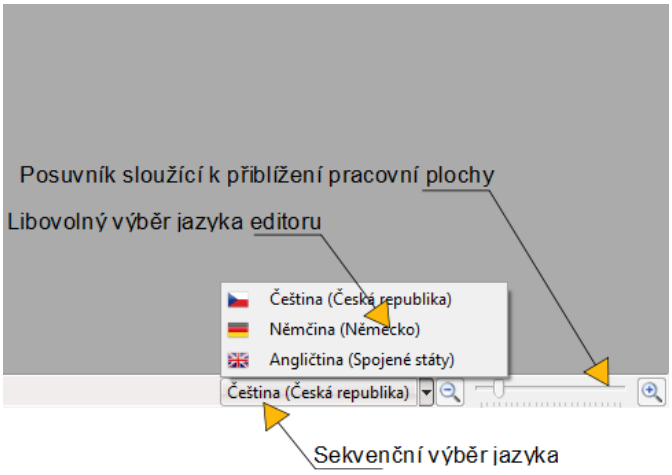
Načítání jazykových mutací aplikace se provádí při startu editoru a to tak, že jsou v adresáři, kde je nainstalován editor, vyhledány všechny složky, které obsahují jazykové zdroje pro SFC Editor. Pokud je nalezena pouze jediná jazyková mutace, jsou funkce spojené s možností změny jazyka zakázány.

Ukončit editor lze standardně z menu `Soubor-Konec` nebo pomocí křížku. Před ukončením se editor dotáže na uložení dosud neuložených schémat.

5 Spuštění editoru

Editor lze spustit bez jakýchkoli parametrů a to prostým poklikáním na ikonu *SFC Editoru*. Další způsob jakým lze SFC Editor spustit, je využít nadřazený systém, který

[2] V dokovacím panelu Error List se lokalizace projeví až po novém překladu schématu



Obrázek 8: Lokalizace a ovládání funkce přiblížení

integruje SFC Editor a nabízí možnost jeho spuštění jako dceřiné aplikace[3].

6 Vytvoření nového SFC diagramu

Po spuštění editoru není otevřen žádný SFC diagram (schéma), a proto je nutné vytvořit nový diagram nebo otevřít již vytvořený. To je možné v menu `Soubor-Nový...` nebo kliknutím na ikonu nového schématu v hlavní liště editoru. Zobrazí se nabídka s možnostmi výběru typu schématu obr. 9. Možnost *Prázdný* je určena pro navržení schématu bez jakýchkoli omezení, ale bez možnosti překladu schématu. Možné je pouze ověření správnosti sestavení schématu. Naproti tomu možnost `ATMT` je omezena určitými limity, viz kapitola 3. Ze schématu sestaveného pro `ATMT` lze generovat tabulku přechodů. Schéma typu `EATMT` je rozšířením bloku `ATMT` a nabízí obdobné možnosti. Překladu schématu do tabulky přechodů se blíže věnuje kapitola 8. V nabídce vytvoření nového schématu je ještě možné zadat název schématu.

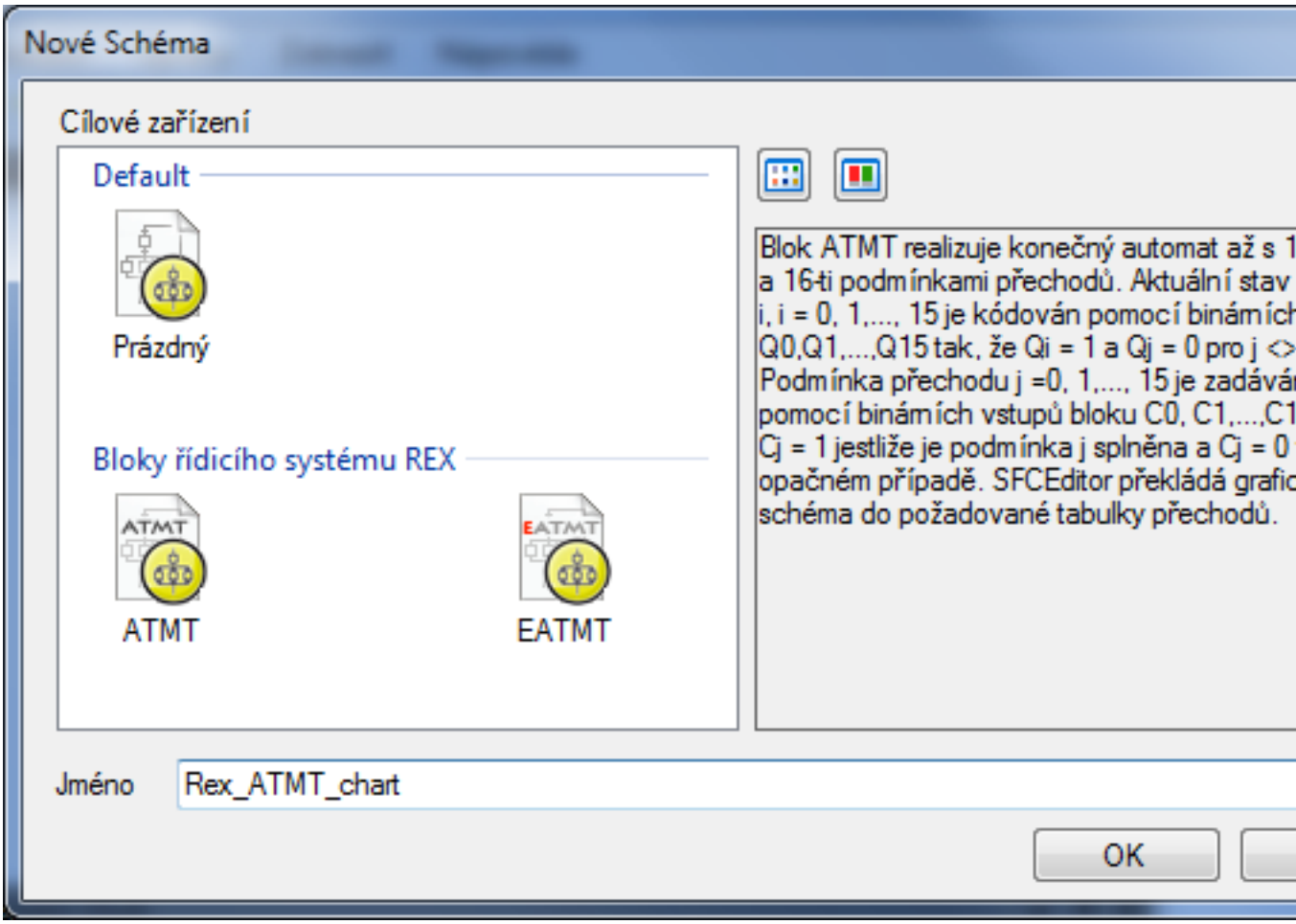
Další možností, jak vytvořit nový diagram, je přímo z nabídky `Soubor-Nový...-SFC pro blok ATMT` (`Soubor-Nový...-SFC pro blok EATMT`) nebo klávesovou zkratkou `CTRL + SHIFT + A(E)`. Tento způsob vytváření nových schémat je funkční pouze v MDI režimu editoru. Pokud je editor spuštěn z příkazové řádky chová se jako SDI aplikace a dovoluje otevřít (vytvořit) pouze jedno schéma.

7 Konstrukce schématu

7.1 Přidání bloků do schématu

Všechny bloky, které je možné přidat do schématu, nalezneme v panelu *Nástroje* obr. 10a. Bloky jsou rozděleny do několika skupin. Skupina *SFC Bloky* obsahuje pouze bloky

[3]Způsob začlenění SFC Editoru do nadřazené aplikace je popsán v kapitole 12.



Obrázek 9: Nabídka vytvoření nového schématu

z SFC formalismu. Ve skupině *Skupiny SFC Bloků* jsou nejčastěji používaná spojení SFC bloků. Skupina *Speciální Bloky* obsahuje pouze jeden blok, a to popisek (*Text*). Bloky do schématu přidáme prostým přetáhnutím z panelu Nástrojů na požadovanou pozici ve schématu.

Odstranit blok nebo skupinu bloků ze schématu, lze po jejich označení klávesou *Delete*.

7.2 Editace vlastností bloků

Téměř všechny vlastnosti bloků je možné editovat v panelu *Vlastnosti* obr. 10b. Aby bylo možné vlastnosti vybraného bloku editovat, je nutné blok označit. Blok označíme kliknutím. Pokud je panel vlastností *Vlastnosti* otevřený („připíchnutý“), zobrazí se v něm editovatelné vlastnosti. Pokud je panel *Properties* zavřený nebo minimalizovaný, lze stiskem klávesy *ENTER* nebo *dvojklikem na blok* panel *Vlastnosti* otevřít.

Následující vlastnosti: pozice bloků, velikost bloků, pozice přípojného bodu (pinu) nebo text popisku lze editovat přímo ve schématu pomocí myši nebo po označení šipkami na klávesnici.

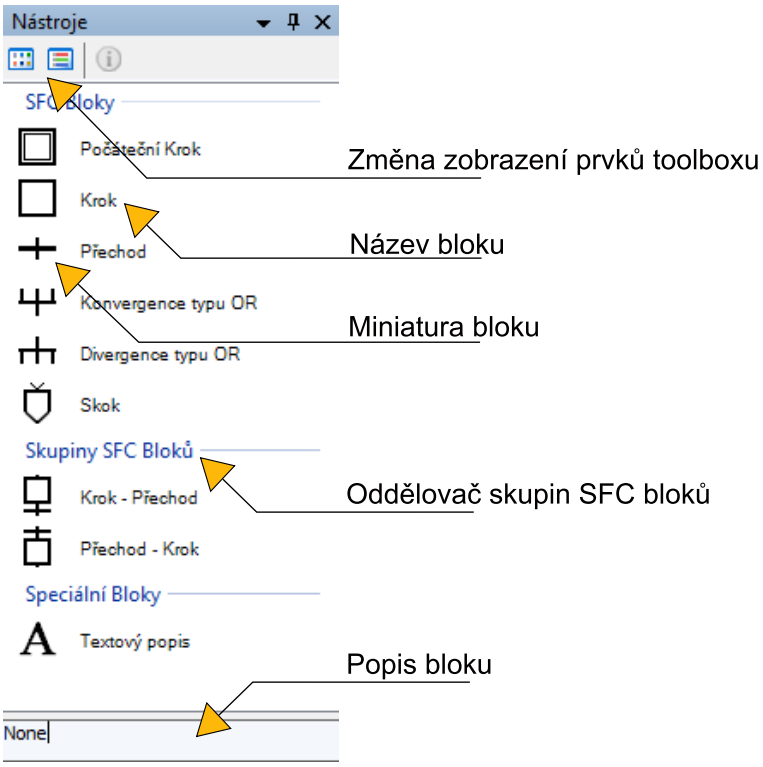
Pozici bloku můžeme měnit jeho uchopením a tažením na požadovanou pozici. Editor umožňuje přesouvat i více bloků najednou. Pozice, na kterou je blok nebo skupina bloků přesouvána, není nijak kontrolována, proto je zcela na uživateli, dbát na to, aby bloky nebyly umístěny mino hranice výkresu.

Výška resp. šířka bloku se dá editovat tažením za čtverce, které se po označení bloku zobrazí. Pokud při táhnutí za zvětšovací čtverce stiskneme klávesu SHIFT, bloky se budou symetricky zvětšovat (zmenšovat), viz obr. 11. Velikost, na kterou lze blok zvětšit (zmenšit), je limitována minimální a maximální dovolenou hodnotou. Tyto hodnoty se mění v závislosti na typu bloku. V závislosti na typu bloku se mění i směry, ve kterých je možné blok zvětšovat (zmenšovat).

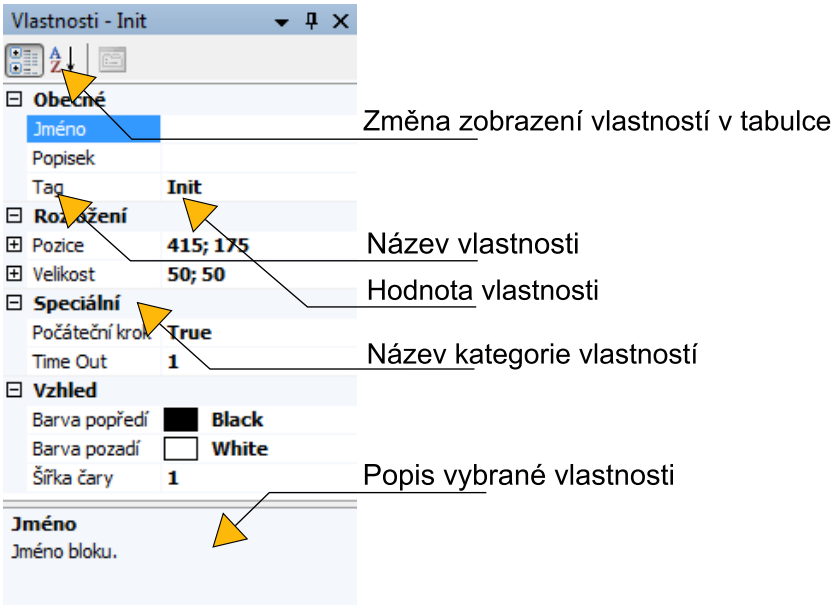
Při změně velikosti bloku dochází i k posunutí přípojných bodů bloku (pinů). V případě bloků s jedním vstupem nebo výstupem se piny snaží udržet ve středu bloku s respektováním mřížky, do které se všechny prvky ve schématu vykreslují. U bloků s více piny si nepřipojené piny udržují ekvidistantní vzdálenost (s respektováním mřížky). Piny, které jsou připojené, zůstávají i po změně velikosti bloku na svém místě, pokud je to možné. V případě, že se blok zmenší natolik, že by pin byl mimo blok, dojde k jeho posunutí.

Pozici pinu v rámci bloku je možné měnit pouze přetažením myši nebo šipkami na klávesnici. V případě, že je přesouvaný pin nepřipojený, je nutné dbát na to, aby byl pin tažen pouze ve vodorovném směru. V opačném případě se začne vytvářet spojnice mezi bloky (cesta) a pin se přestane posouvat. Pokud je pin, se kterým se pohybuje, připojený, nedochází k předchozí situaci (vytváření cesty) obr. 12.

Text bloku *Text* lze editovat také přímo ve schématu. Po dvojkliku je možné editovat text popisku. Ukončení editace je možné provést kliknutím mimo hranice bloku nebo stiskem klávesové kombinace `CTRL + ENTER`.

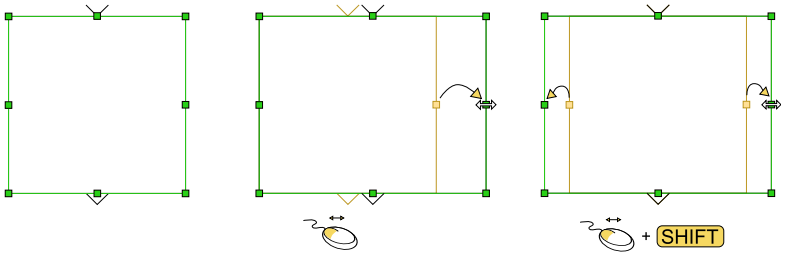


(a) Panel Nástrojů

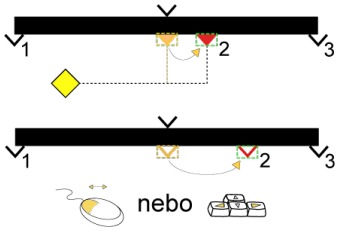


(b) Panel Vlastností

Obrázek 10: Dokovací panely



Obrázek 11: Změna velikosti bloku



Obrázek 12: Rozmístění pinu v rámci bloku

7.2.1 Popis vlastností

Vlastnosti společné pro většinu bloků

- **Popisek** – textová poznámka bloku, která se zobrazuje po najetí na blok u kurzoru (ToolTip text). Text popisu lze libovolně měnit.
- **Jméno** – krátký popisek bloku, vypisuje se před tagem bloku, oddělen dvojtečkou. Text popisku lze libovolně měnit (v případě dlouhých textů nebude popisek zobrazen celý).
- **Tag** – identifikátor bloku, v případě bloku *Přechod*, *Počáteční Krok* a *Krok* je tato vlastnost klíčová pro překlad schématu. U bloku *Přechod* udává číslo, které následuje za písmenem „C“, číslo vstupu bloku `ATMT` (`EATMT`), se kterým *Přechod* koresponduje. U bloků *Počáteční Krok* a *Krok* je tomu obdobně, pouze s výstupy bloků `ATMT` (`EATMT`).
- **Pozice** – pozice bloku ve schématu. Tuto vlastnost lze editovat myší.
- **Velikost** – velikost bloku. Tuto vlastnost lze editovat myší.
- **Barva pozadí** – barva pozadí bloku.
- **Barva popředí** –barva popředí bloku.

Speciální vlastnosti jednotlivých bloků V následující části budou blíže popsány speciální vlastnosti jednotlivých bloků. Výše uvedené vlastnosti společné pro všechny bloky již nebudou zmiňovány.

1. **Krok** (**Počáteční Krok**) Vlastnosti bloku *Počáteční Krok* jsou zcela shodné s vlastnostmi bloku *Krok*. Blok *Počáteční Krok* je pouze speciální případ bloku *Krok*, který se ve schématu může vyskytovat pouze jednou. Liší se hodnotou vlastnosti *Tag*, *Počáteční Krok* a grafickou podobou obr. 13a.

    - **Počáteční Krok** – tento atribut nabývá pouze logické hodnoty *true* nebo *false*. Určuje, zda se jedná o „normální“ *Krok* nebo o *Počáteční Krok*.
    - **TimeOut** - reálné číslo, které určuje dobu strávenou v aktivním stavu bloku, po které se nastaví příznak uplynutí timeoutu na logickou hodnotu *true*.
    - **Šířka čáry** – tloušťka čar, kterými je blok vykreslován.

2. **Přechod** Tento blok má fixní velikost, proto není vlastnost *Size* editovatelná obr. 13b.

    - **Skrýt popisek** – příznak určující viditelnost popisku bloku (tento atribut se vyskytuje u všech bloků s popiskem mimo tělo bloku).

3. **Divergence typu OR** Divergence typu OR (Divergence of sequencial selection) umožňuje editovat pouze svou šířku obr. 13c.

    - **Počet výstupů** – Počet výstupních přípojných míst (pinů). Počet výstupních pinů je omezen zdola hodnotou 2 a shora velikostí bloku.

4. **Konvergence typu OR** Konvergence typu OR (Convergence of sequencial selection) umožňuje stejně jako Divergence typu OR editovat pouze svou šířku. 13d.

    - **Počet vstupů** – Počet vstupních přípojných míst (pinů). Počet vstupních pinů je omezen zdola hodnotou 2 a shora velikostí bloku.

5. **Divergence typu AND** Divergence typu AND (Divergence of simultaneous sequences) umožňuje editovat pouze svou šířku. 13e.

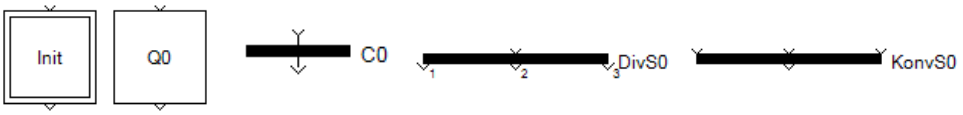
    - **Počet výstupů** – Počet výstupních přípojných míst (pinů). Počet výstupních pinů je omezen zdola hodnotou 2 a shora velikostí bloku.

6. **Konvergence typu AND** Konvergence typu AND (Convergence of simultaneous sequences) umožňuje editovat pouze svou šířku. 13f.

    - **Počet vstupů** – Počet vstupních přípojných míst (pinů). Počet vstupních pinů je omezen zdola hodnotou 2 a shora velikostí bloku.

7. **Skok** Grafická podoba bloku obr. 13g.

    - **Cíl** – specifikuje SFC blok (konkrétně blok typu *Krok*), na kterém bude běh schématem pokračovat.
    - **Šířka čáry** – tloušťka čar, kterými je blok vykreslován.



(a) Počáteční krok, Krok (b) Přechod (c) Divergence OR (d) Konvergence OR

(e) Divergence AND (f) Konvergence AND (g) Skok (h) Text

Obrázek 13: SFC bloky

8. **Text** *Text* není definován v SFC formalismu, ale je vhodný pro popis schématu, a proto je v editoru implementován ve formě bloku obr. 13h.
   - **Zarovnání** – zarovnání textu.
   - **Font** – font textu.
   - **Text** – samotný text popisku. Tato vlastnost lze editovat po dvojkliku na blok přímo na pracovní ploše.

7.3 Spojení bloků

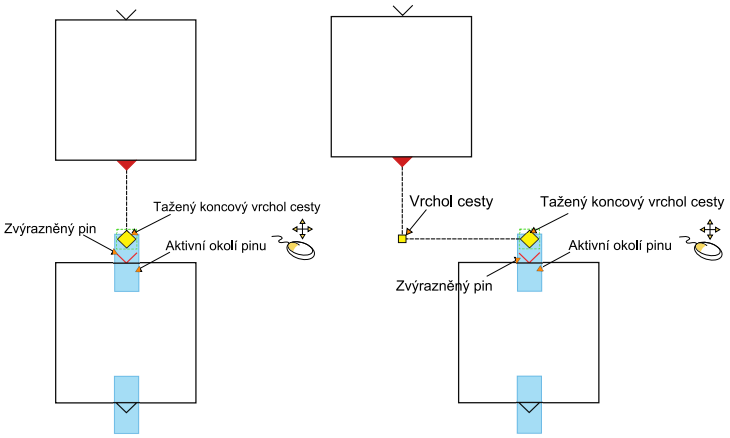
Bloky se propojují pomocí vstupních a výstupních pinů. Samotné spojení se provádí tahem myši. Ve chvíli, kdy se pin uchopí a je tažen mimo svou pozici, dojde k vytvoření cesty, která drží pravoúhlé zalomení a sleduje pozici kurzoru. Pokud se s takto vytvořenou cestou přiblížíme do okolí pinu, který je vhodný[4] pro připojení, pin se zvýrazní a po uvolnění tlačítka se cesta připojí k pinu obr. 14. Pokud je ke vstupnímu pinu bloku připojena cesta, je vstupní pin vykreslován jako šipka (důvodem je zvýšení přehlednosti schématu).

Cestu lze libovolně zalamovat (při stálém dodržení pravoúhlého zalomení). Zalomit cestu je možné tak, že nejprve vytvoříme první část cesty („L“ zalomení). Po uvolnění levého tlačítka myši znovu uchopíme koncový vrchol cesty a tažením vytváříme další část cesty (další „L“ zalomení) obr. 15a. Zalomení lze vytvořit i změnou pozice středového vrcholu cesty obr. 15b. Další způsob, jak změnit tvar čáry propojující bloky je tažení za jednotlivé úsečky zalomené čáry, viz obr. 16. Zalomení cesty se vytváří také při pohybu s bloky, a to z důvodu snahy editoru o dodržení pravoúhlého zalomení cest.

Cesta, která má oba konce připojeny, se vykresluje plnou čarou. Pokud cestu označíme[5], je možné v panelu *Vlastnosti* editovat šířku a barvu čáry, kterou bude cesta vykreslována obr. 17.

[4] **Vhodný pin** je takový, který není pinem stejného bloku jako pin počáteční (v případě, že existuje) a je opačného typu, než pin počáteční (opět za předpokladu existence počátečního pinu).

[5] **Označení** cesty se projeví zobrazením všech vrcholů cesty.



(a) Jednoduché propojení (b) Zalomené propojení

Obrázek 14: Propojení bloků

Cestu lze po označení smazat klávesou `DELETE`. Pokud byla cesta připojena na obou koncích, dojde po jejím smazání ke změně stavů obou pinu a je možné bloky dále propojovat.

8 Překlad schématu

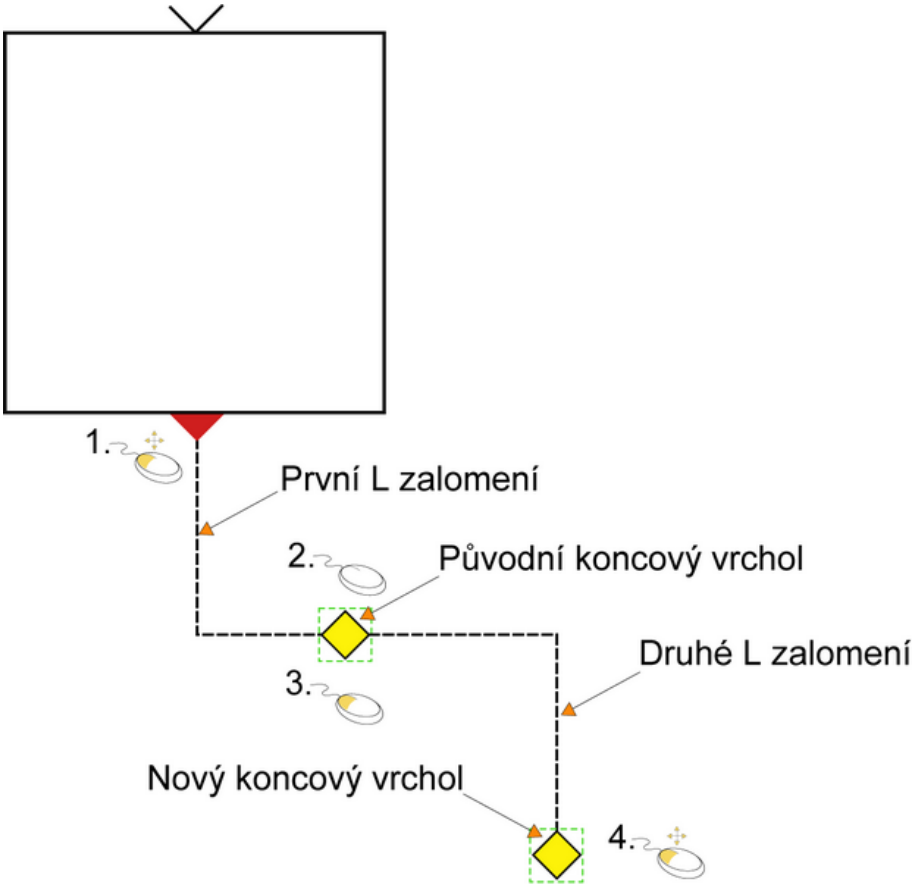
Překlad schématu je rozdělen do tří částí. V první části je prováděna validace schématu (ověření správnosti sestaveného schématu). Druhá část je vlastní generace tabulky přechodů a vektoru timeoutů. Ve třetí části je pak řešen způsob zobrazení a předání výsledků.

SFC Editor nabízí možnost samostatné validace schématu nebo validace schématu a následované generace tabulky přechodů. Tyto možnosti se nastavují v horní liště hlavního okna editoru, viz obr. 18. Samotný překlad se spustí ikonou na hlavní liště nebo z menu `Úpravy-Překlad`.

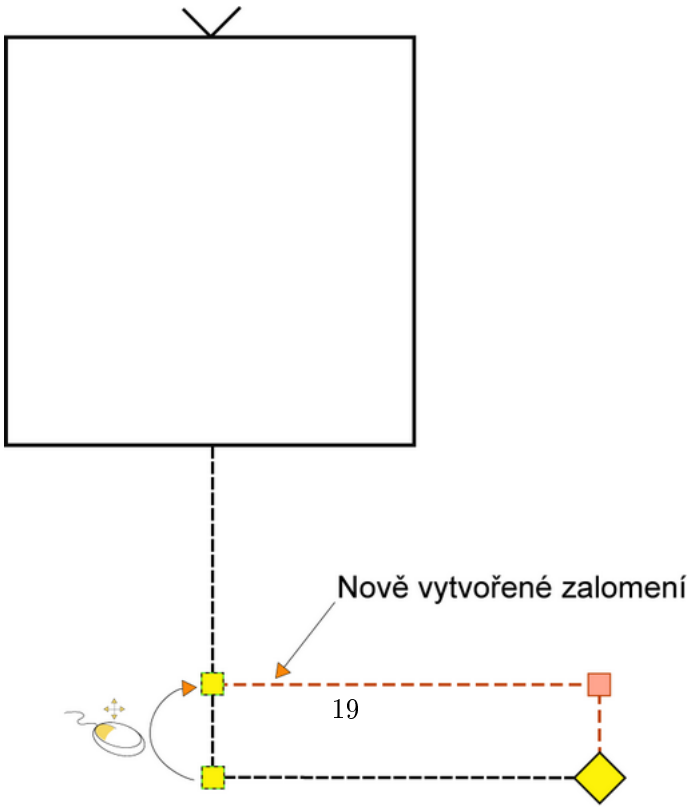
Úspěšný překlad schématu je doprovázen zobrazením nápisu *Succeeded* v levém horním rohu pracovní plochy. V případě chyby při překladu schématu je zobrazen nápis *Failed*, viz obr. 19.

8.1 Validace schématu

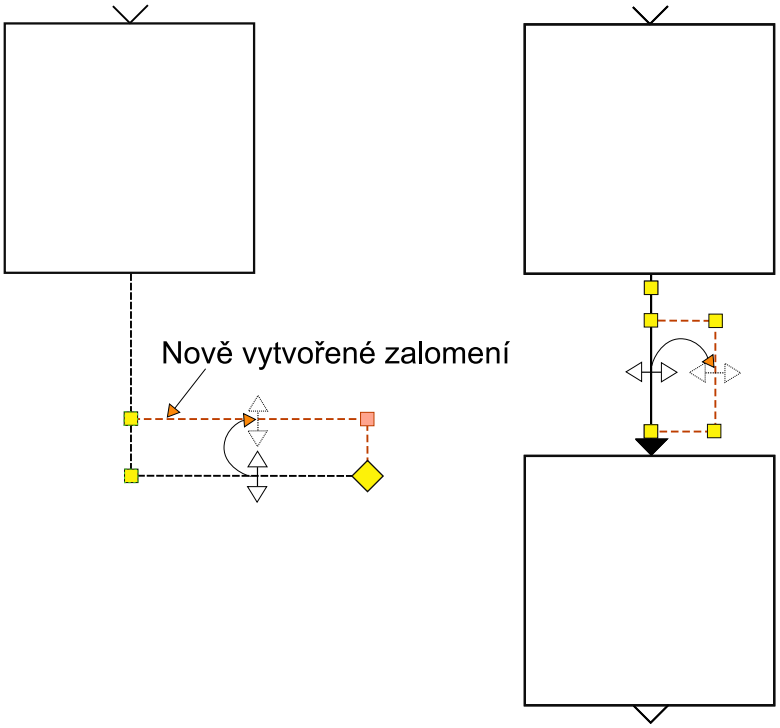
Validátor se liší v závislosti na cílovém zařízení, pro které je schéma generováno. Výsledkem validace jsou chybová a varovná hlášení. Tyto hlášení se zobrazují v panelu *Seznam Chyb* (obr. 20). Validace se považuje za úspěšnou, pokud je počet chybových hlášení nulový. Varovná hlášení jsou při generaci ignorována.



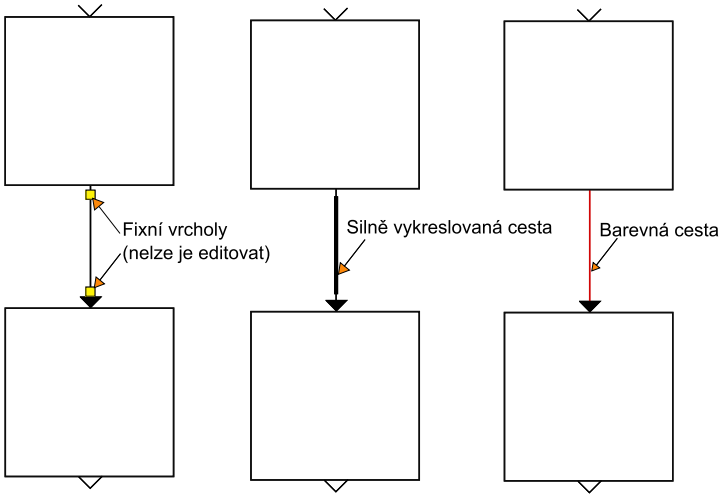
(a) Subcesta



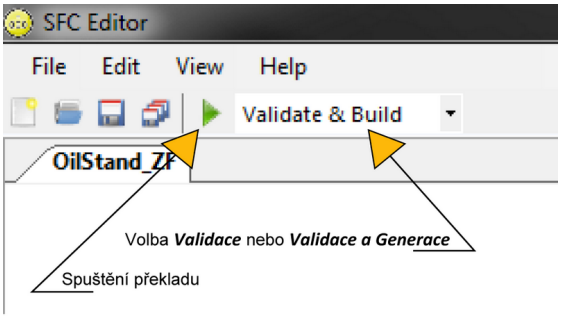
(b) Posunutí vrcholu



Obrázek 16: Další způsob zalomení cesty



Obrázek 17: Editovatelné vlastnosti cesty



Obrázek 18: Nastavení a spuštění typu překladu

(a) Neúspěšný překlad (b) Úspěšný překlad

Obrázek 19: Upozornění na výsledek překladu

8.2 Generace tabulky přechodů

Spuštění generace je podmíněno úspěšnou validací schématu. Generace tabulky přechodů a vektoru timeoutů se opět liší v závislosti na cílovém zařízení (např. výstup pro blok `ATMT` systému REXYGEN, viz obr. 21). Pro některá zařízení může mít výstup zcela jinou podobu (např. zdrojový kód).

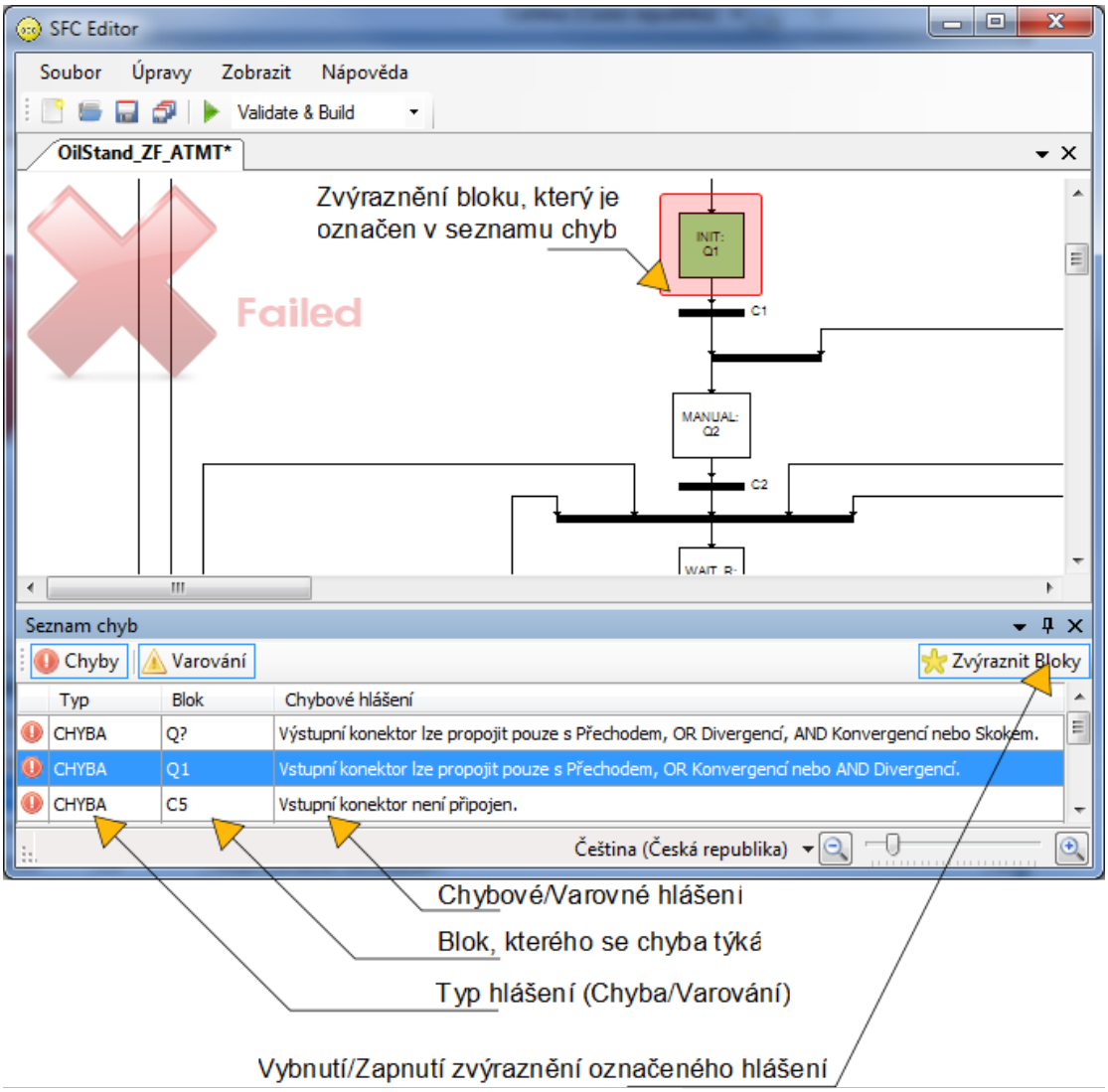
8.3 Předání výsledků

Komunikace editoru s okolím (jiným programem) není pro jeho správnou funkci nutná. Výstup editoru pro jednotlivá zařízení a spolupracující programy je snadno přizpůsobitelný. V současné verzi editoru je zcela implementován překladač pro blok `ATMT` a `EATMT`. Pro předání výsledných dat je použita technika paměťově mapovaných souborů, viz kapitola 12.

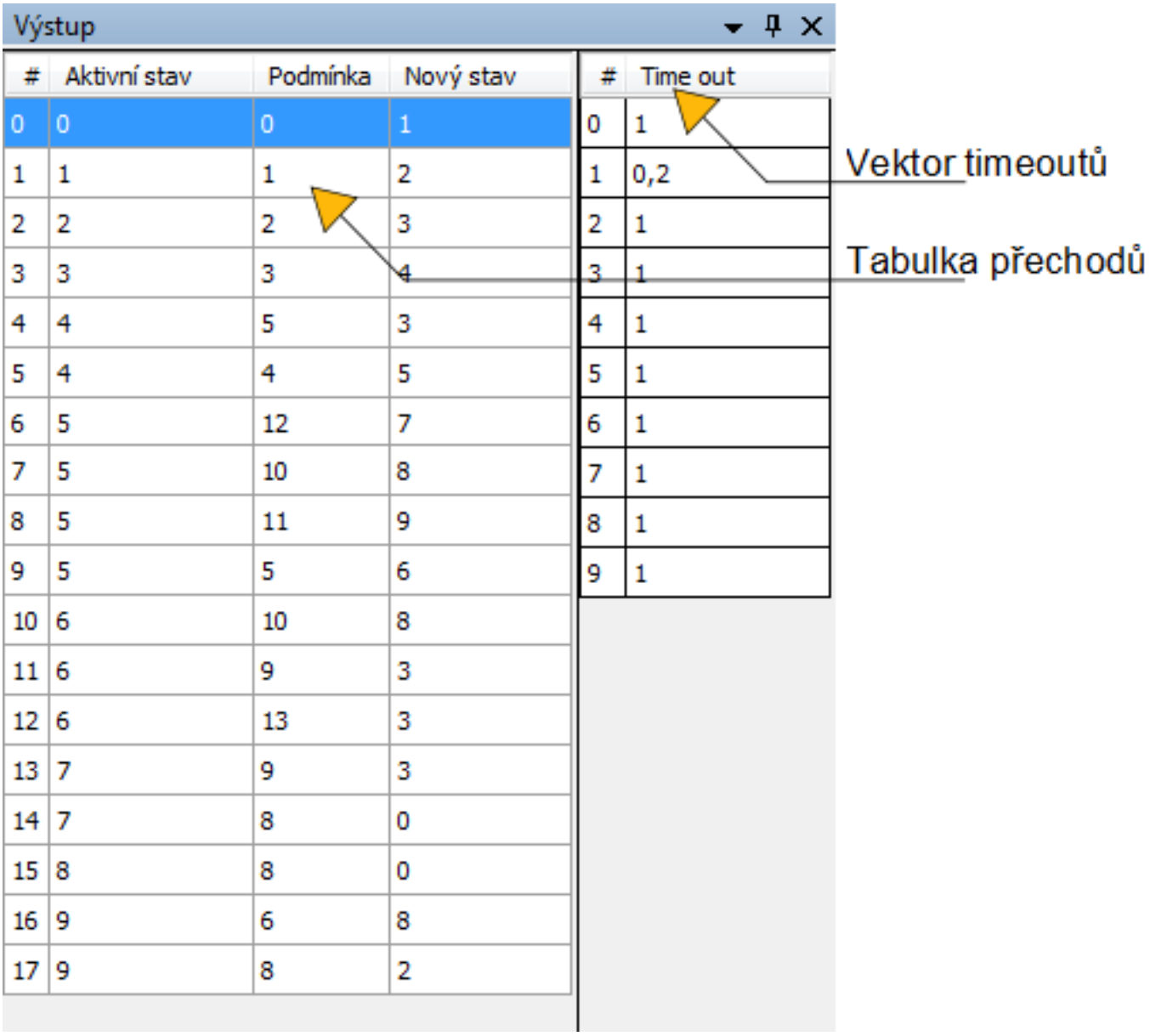
9 Tisk a export schématu

Naprostou samozřejmostí u jakéhokoli podobného grafického editoru jako je SFC Editor je možnost tisku. SFC Editor tisk schématu podporuje. Nabídku tisku lze vyvolat z nabídky `Soubor-Tisk...` .

Tiskový dialog nemá vzhled dobře známého dialogu z prostředí Windows, ale má rozšířené možnosti o velký náhled tisknutého schématu s možností stránkování a zvětšování náhledu. V tomto dialogu je samozřejmě možné provést veškeré klasické nastavení jako je počet tisknutých kopií, počet stránek, výběr tiskárny, atd. Mimo tyto klasické položky



Obrázek 20: Seznam chyb



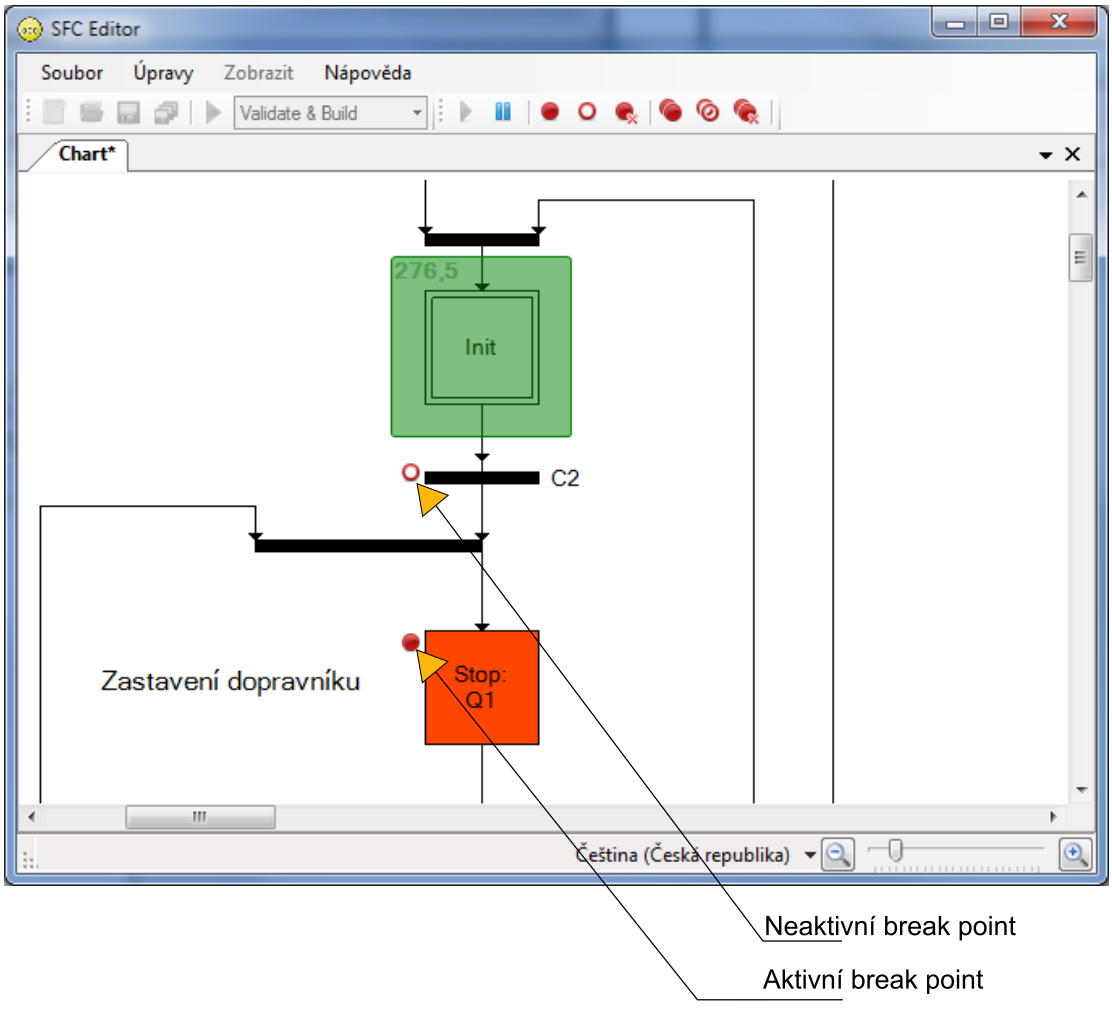
Obrázek 21: Výsledek generace pro zařízení (blok) `ATMT` systému REXYGEN

je zde možnost nastavení tisku legendy, ve které jsou uvedeny všechny bloky obsažené ve schématu se svými jmény a veškerými popisky.

Pro získání SFC schématu z editoru v grafické podobě, nabízí editor export schématu do vektorového formátu *SVG*. Export se spouští z nabídky `Soubor-Export do SVG...`.

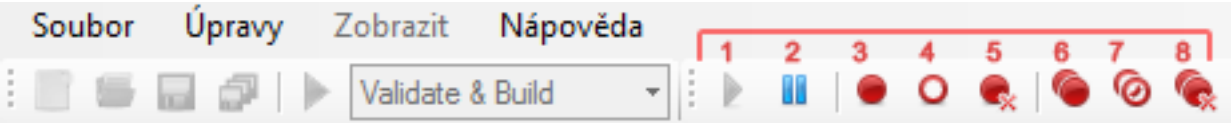
10 Monitorování vývoje stavu v SFC diagramu

Hlavní funkcí editoru je poskytovat nástroje k sestaveni, editaci a překladu SFC schémat. Kromě těchto funkcí však editor disponuje ještě možnosti zobrazit ve schématu aktuální stav konečného automatu, který realizuje příslušnou tabulku přechodů (schéma). Tato funkčnost je označována jako *monitorovací funkce* nebo *monitorovací mód* editoru.



Obrázek 22: Monitorovací režim

Logika implementující funkčnost monitorování využívá komunikační rozhraní popsané



Obrázek 23: Panel s nástroji pro řízení monitorování

v kapitole 12.3, pro načítání stavů vstupů a výstupů bloku realizujícího automat (např. `ATMT` nebo `EATMT`). Načtená data jsou zpracována a příslušné kroky resp. přechody jsou označeny jako aktivní. V případě přechodu (Transition) je za aktivní stav považována situace, kdy je splněna podmínka konkrétního přechodu. U kroku (Step) to odpovídá aktivnímu stavu. Ve schématu je aktivní blok orámovaný průhledným zeleným obdélníkem, který má v levém horním rohu uvedenu dobu strávenou v aktivním stavu (obr. 22)[6].

10.1 Spuštění editoru v monitorovacím módu

SFC Editor v monitorovacím módu nelze spustit bez implementace základní logiky nutné pro běh monitorování. Podrobnější popis jak spustit editor v monitorovacím režimu je uveden v kapitole 12.

Po spuštění editoru v monitorovacím módu je uživatel dotázán na umístění schématu, které odpovídá monitorovanému procesu. Pokud je dialog otevření schématu uzavřen a není vybráno SFC schéma je editor ukončen. V opačném případě se zobrazí vybrané schéma a je automaticky spuštěno monitorování. Volba odpovídajícího SFC schématu je ponechána uživateli.

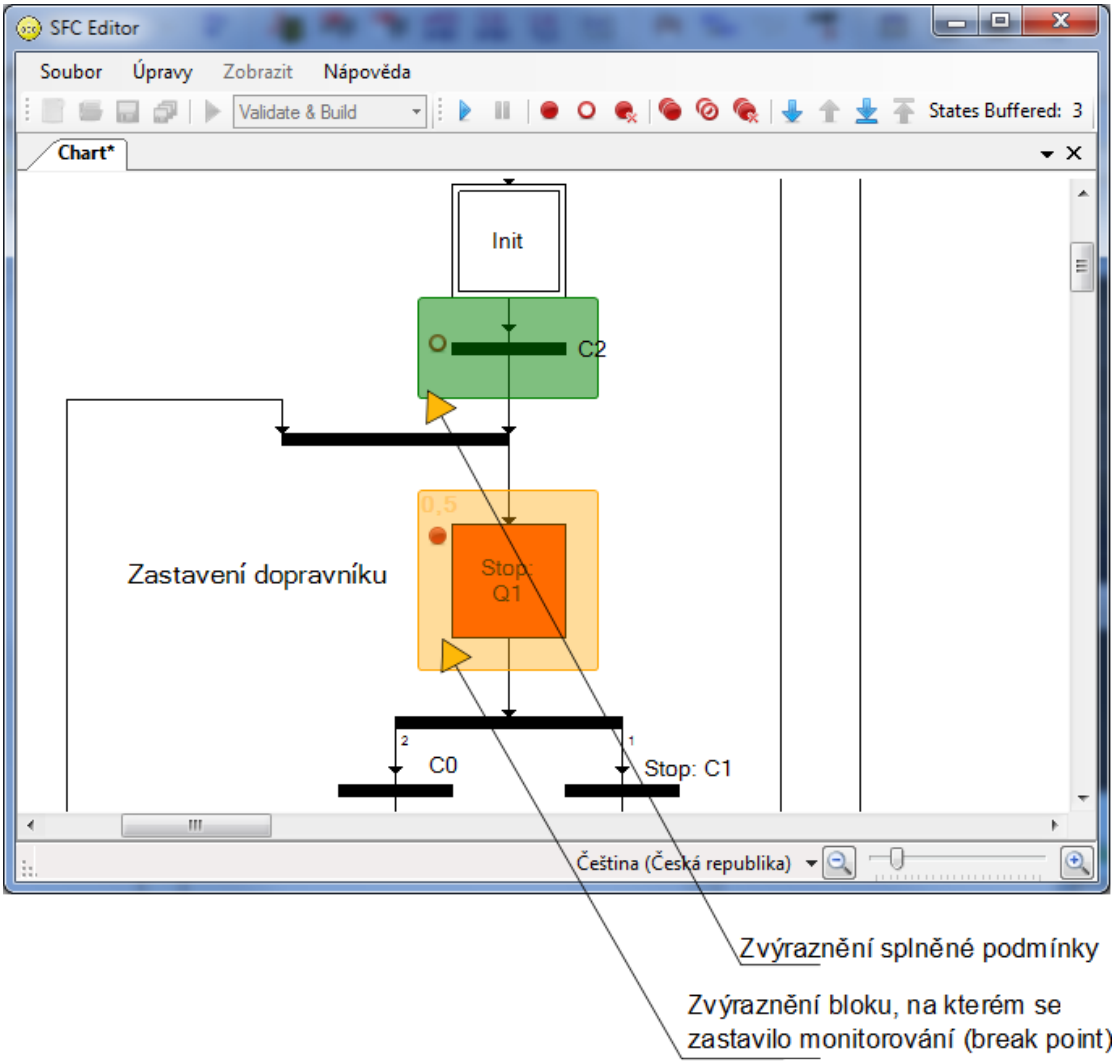
10.2 Řízení běhu monitorování

V monitorovacím režimu nabízí editor jednoduché nástroje „ladění“ schématu a jeho krokování. Všechny tyto nástroje jsou v panelu nástrojů umístěném v horní části aplikace, viz obr. 23.

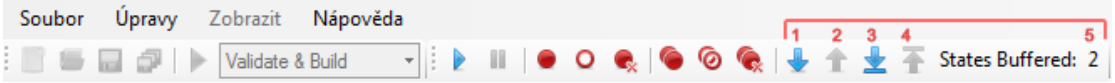
Tlačítko na obrázku 23 označené číslem 1 slouží ke spuštění monitorování. Tlačítko s číslem 2 pozastaví monitorování[7]. Další tlačítka slouží k umístění, smazání nebo dočasnému zakázání bodu ve schématu, na kterém se monitorování zastaví (podobné praci s breakpointy v prostředí Visual Studia). Tlačítko 3 označí vybraný blok (nebo skupinu) ve schématu jako blok, na kterém se zastaví monitorování (přidá bloku breakpoint). Po stisku tlačítka 4 je breakpoint označeného SFC bloku (pokud ho má) změněn na breakpoint, který se ignoruje. Smazání breakpointu se realizuje stiskem tlačítka 5. Zbylá tlačítka 6, 7, 8 fungují stejně, pouze s tím rozdílem, že ovlivňují všechny bloky označeny breakpointem. Pokud monitorovací logika narazí na aktivní blok, který je označen breakpointem zastaví se monitorování a blok je vyznačen oranžovým průhledným obdélníkem (obr. 24).

[6] Doba po kterou je blok v aktivním stavu se vypisuje pouze u kroku (Step).

[7] Spuštění nebo pozastavení monitorování nijak neovlivňuje skutečný běh algoritmu realizujícího automat. Dochází pouze ke spuštění/pozastavení **zobrazování** stavu automatu.



Obrázek 24: Monitorovací režim – zastavení na breakpointu.



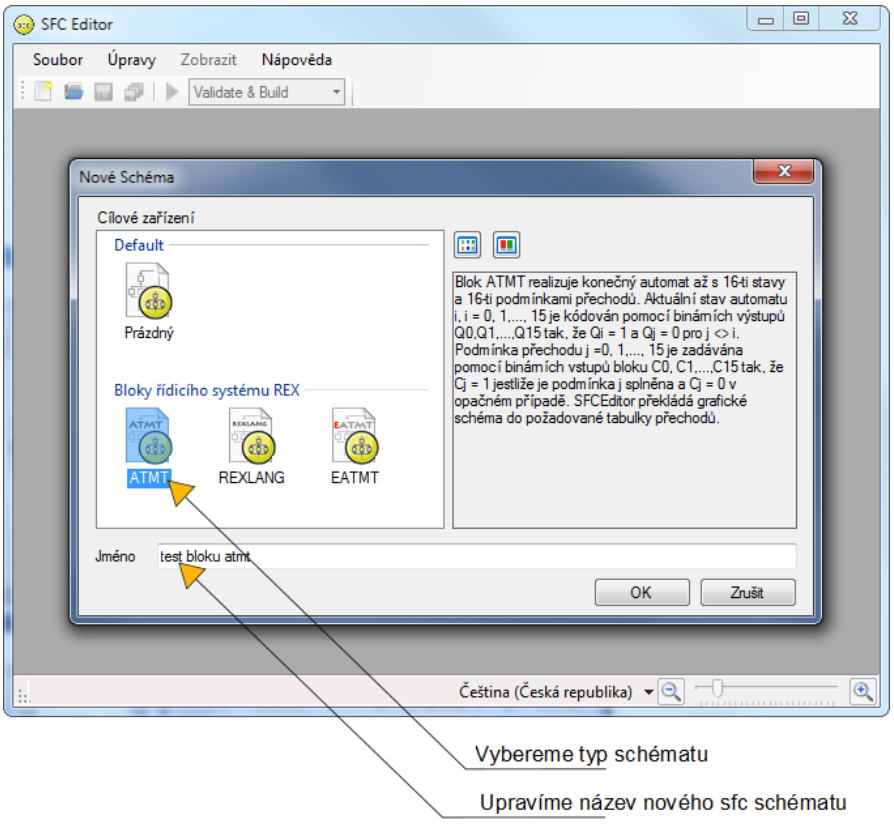
Obrázek 25: Panel s nástroji pro řízení krokování

10.3 Krokování ve schématu

Pokud se monitorování zastaví na breakpointu, zobrazí se v panelu nástrojů ovládání krokování, viz obr. 25. Zastavení monitorování, ale nijak neovlivní běh automatu, a proto je nutné stavy, které bude dále možné krokovat ukládat do paměti. Na obrázku 25 odpovídá hodnota označená číslem 5 počtu uložených stavů, do kterých je možné se krokováním přesunout. Tlačítka 1-4 pak realizují požadovaný pohyb mezi uloženými stavy. Maximální počet uložených stavů je 100.

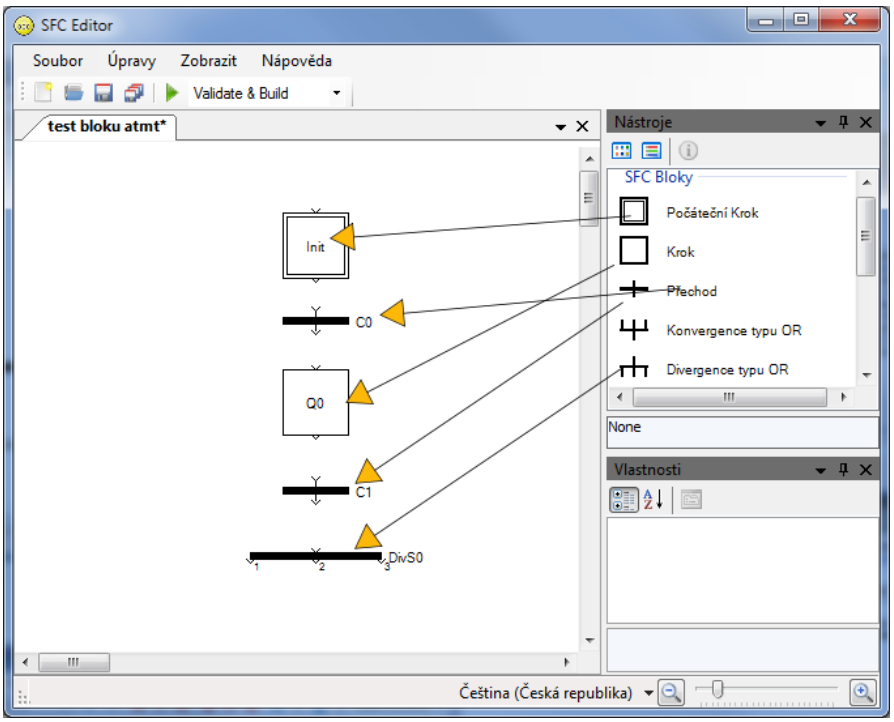
11 Příklad vytvoření diagramu

Spustíme SFC editor a vytvoříme nové schéma pro námi požadované cílové zařízení (`ATMT`, `EATMT`, BLANC) obr. 26. Bloky SFC formalismu přetáhnutím z nabídky *Nástroje* umístíme do schématu na požadovanou pozici (obr. 27), a propojíme je (obr. 28.a).

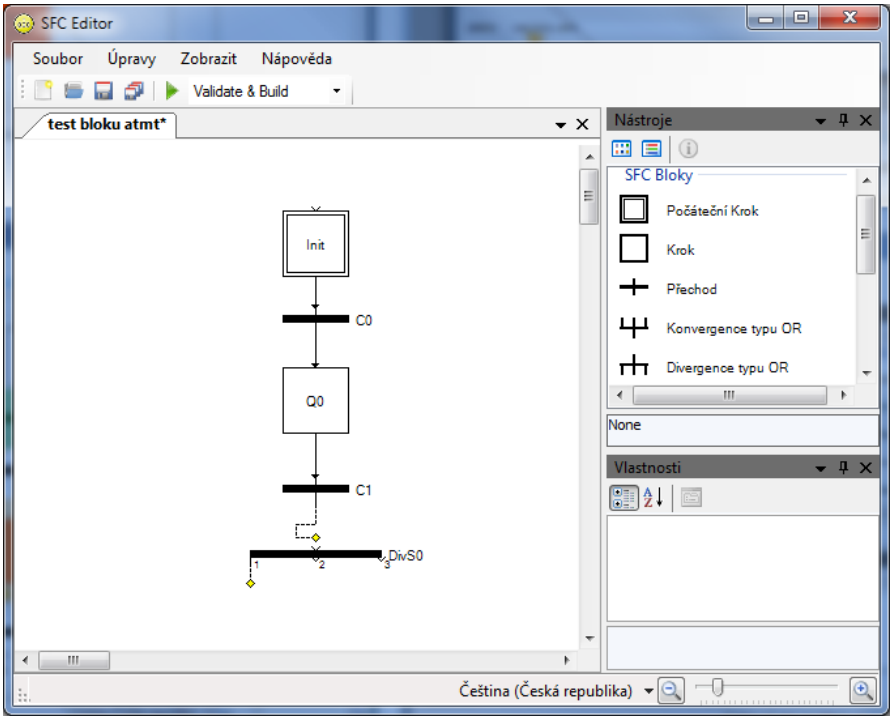


Obrázek 26: Postup vytvoření SFC diagramu - Vytvoření nového schématu

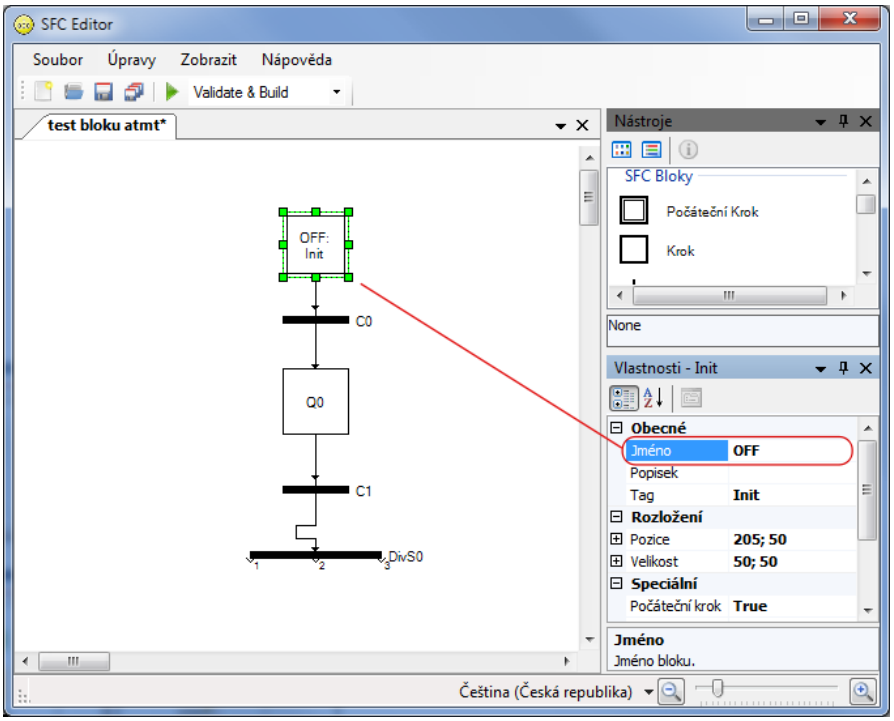
Dále upravíme vlastnosti bloků tak, aby bylo schéma validní (obr. 28.b). Z výsledného schématu (obr. 29.a) potom vygenerujeme tabulku přechodů (obr. 28.b).



Obrázek 27: Postup vytvoření SFC diagramu - Vložení bloků do schématu

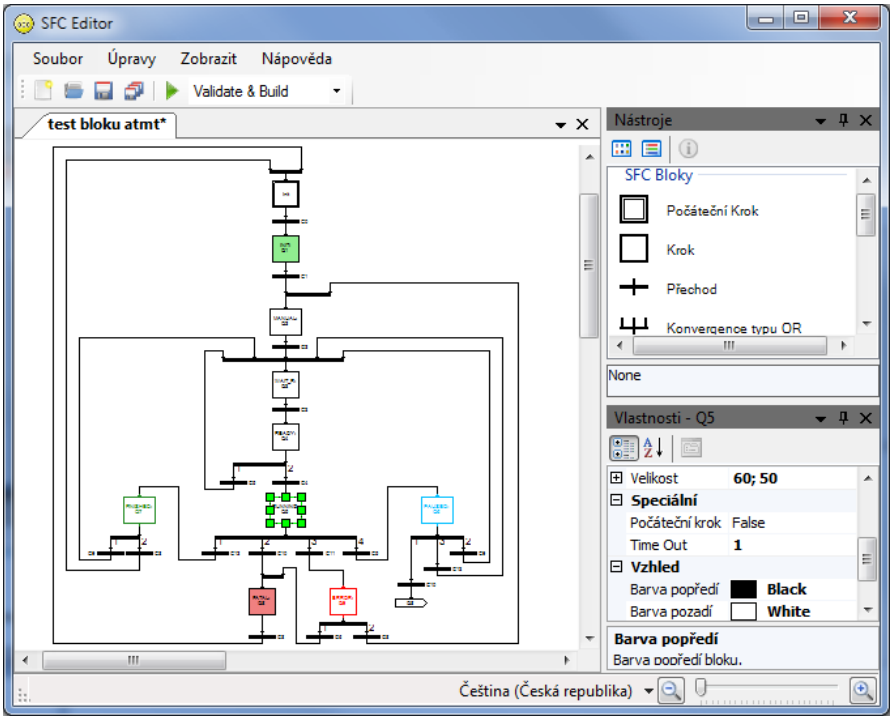


(a) Propojení bloků

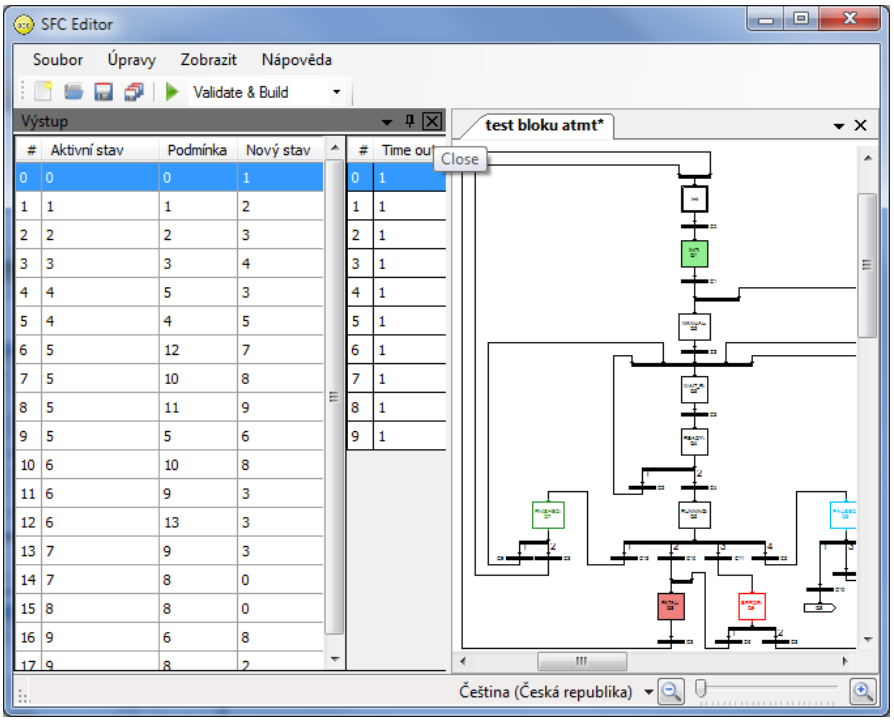


(b) Editace vlastností bloků

Obrázek 28: Postup vytvoření SFC diagramu



(a) Výsledné schéma



(b) Překlad sestaveného schématu

Obrázek 29: Postup vytvoření a překladu SFC diagramu

12 Začlenění SFC Editoru do prostředí nadřazeného řídicího systému

SFC Editor je navrhnut tak, aby jej bylo možno bez větších problému začlenit do různých nadřazených aplikací a softwarových balíků.

Aplikace, která chce editor využívat musí zvládat logiku spuštění editoru, přečtení výsledků a volitelně pak i komunikaci s editorem, sloužící pro monitorování vývoje stavu ve schématu.

V následující kapitole se budeme podrobněji zabývat implementačními detaily jednotlivých částí nutných pro integraci SFC Editoru.

12.1 Spuštění SCF Editoru

V kapitole 5 jsme se zmínili o možnosti spuštění editoru z prostředí Windows. Další ze způsobů jak spustit editor je uvést jeden z následujících parametrů:

```
/fb=<jmeno_bloku> /nstep=<max_pocet_kroku> /ntr=<max_pocet_pravidel>
<sfc_soubor> /mmf=<jmeno_mmf_souboru>
```

$< jmeno\_bloku >$ definuje typ cílového zařízení (v současné verzi editoru jsou podporovány tyto: `ATMT`, `EATMT`.),

$< max\_pocet\_kroku >$ udává kolik různých kroku (bloků Step) může být ve schématu (pro blok `ATMT` je to 16, u `EATMT` pak 256),

$< max\_pocet\_pravidel >$ limituje počet pravidel přechodu ve vygenerované tabulce přechodů (pro blok `ATMT` je to 64, u `EATMT` pak 1024),

$< sfc\_soubor >$ udává umístění schématu, který se má otevřít nebo vytvořit.

$< jmeno\_mmf\_souboru >$ jméno *Memory Mapped File* souboru, ve kterém se vrací vygenerovaná tabulka přechodů nebo probíhá komunikace s jinou aplikací.

Takto spuštěný editor se nejprve snaží otevřít SFC schéma na cestě předané v prametru $< sfc\_soubor >$. Pokud tento soubor neexistuje, nabídne možnost vytvoření nového schématu. Dále editor nabízí stejné možnosti editace schématu jako v případě spuštění v MDI režimu. Jediná odchylka je v nemožnosti otevření více schémat najednou (editor pracuje v SDI módu).

Dále SFC editor očekává existenci paměťově mapovaného souboru, do kterého se po překladu schématu uloží výsledky. Strukturu výsledků a způsob jejich načtení bude popsán v následující podkapitole.

```
/mon /fb=<jmeno_bloku> /mmf=<jmeno_mmf_souboru>
```

< *jmeno_mmf_souboru* > jméno *Memory Mapped File* souboru, ve kterém se vrací vygenerovaná tabulka přechodů nebo probíhá komunikace s jinou aplikací.

< *jmeno_bloku* > definuje typ cílového zařízení (v současné verzi editoru jsou podporovány tyto: `ATMT`, `EATMT`),

/mon tento parametr je interpretován jako spuštění editoru v monitorovacím módu, viz 10.

Pokud editor spustíme s těmito parametry, bude pracovat v monitorovacím módu. To znamená, že po otevření odpovídajícího SFC schématu bude editor očekávat data, definující v jakém stavu se automat, který je realizuje SFC schéma nachází. Bližší popis struktury dat provedeme v podkapitole 12.3.1.

Následující příklad naznačuje způsob spuštění editoru v monitorovacím módu. Příklad je implementován v jazyce C++.

```
TCHAR* sMmfName;
TCHAR sAppCmd[200];

PROCESS_INFORMATION m_pi;
STARTUPINFO si;

sMmfName = _T("\code{EATMT}_MMF_0");
_tcscpy(sAppCmd, _T("SFCEditor.exe /mon /fb=\code{ATMT} /mmf="));
_tcscat(sAppCmd, sMmfName);

/* Launch external monitor */
memset(&si, 0x00, sizeof(STARTUPINFO));
si.cb = sizeof(STARTUPINFO);
memset(&m_pi, 0x00, sizeof(PROCESS_INFORMATION));

BOOL bOK = CreateProcess(
    NULL,        // pointer to name of executable module
    sAppCmd,     // pointer to command line string
    NULL,        // process security attributes
    NULL,        // thread security attributes
    FALSE,       // handle inheritance flag
    0,           // creation flags
    NULL,        // pointer to new environment block
    NULL,    // pointer to current directory name
    &si,             // pointer to STARTUPINFO
    &m_pi        // pointer to PROCESS_INFORMATION
);
```

Listing 1: Spuštění SFCEditoru v monitorovacím módu z C++

12.2 Načtení výsledku překladu SFC schématu

Jak už bylo výše řečeno, je nutné vytvořit paměťově sdílený soubor s fixně definovaným jménem, do kterého se výsledná data (tabulka přechodů) uloží. Data se do souboru zapisují ve formě textového řetězce formátu ASCII a mají následující strukturu[8]:

[8] Textový řetězec neobsahuje žádné znaky konce řádky. Jediný netisknutelný znak je mezera mezi čísly v jednotlivých vektorech.

$$pocet\_pravidel \overbrace{[stary\_stav\ podminka\ novy\_stav; \ldots; \ldots]}^{Tabulka\ prechodu} \underbrace{[timeOut\_prvniho\_stavu\ \ldots]}_{Vektor\ timeOutu}$$

např.: 3[0 0 1;1 1 2;2 2 1][8.1 2 1]

Vektor timeoutů obsahuje desetinná čísla, ostatní hodnoty v řetězci jsou celočíselné datové typy.
Příklad vytvoření paměťově mapovaného souboru z jazyka C++ je uveden níže:

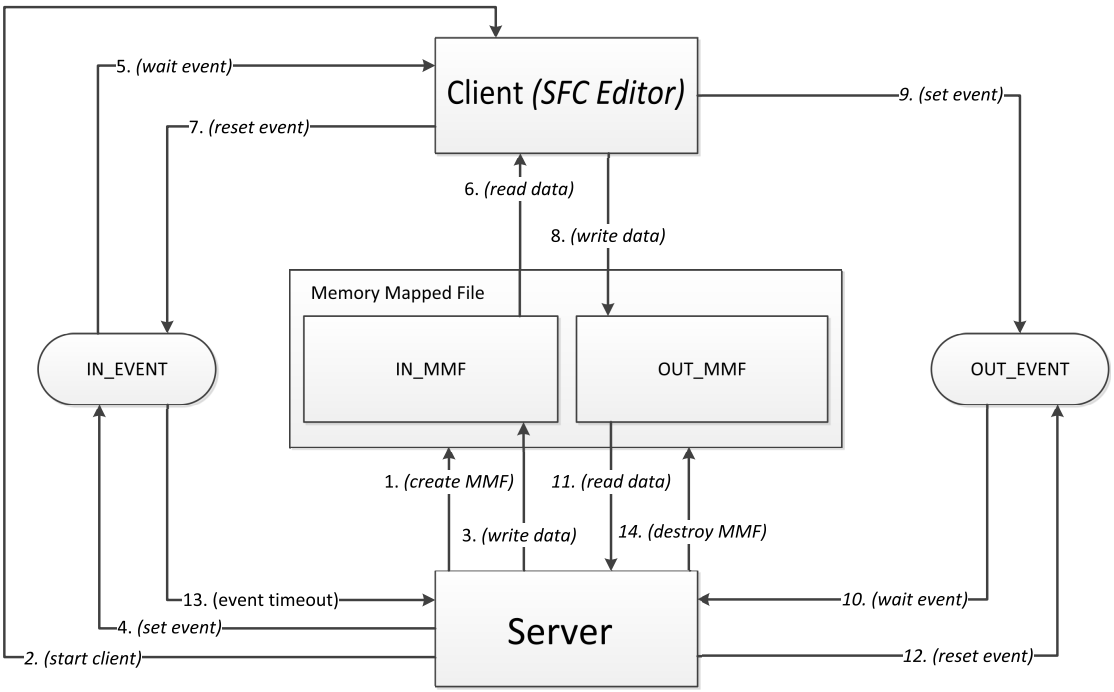
```
#define MMFH_FM_SIZE                4096*1024    /* memory mapped file size */
...
HANDLE  hMapFile = NULL;          /* handle to MMF */
TCHAR     sMmfName[80];           /* MMF name */

_tcscpy(sMmfName, _T("SFC_EDITOR_RESULTS"));

/* Creating of memory mapped file */
    hMapFile = CreateFileMapping(
                  INVALID_HANDLE_VALUE,     // use paging file
                  NULL,                     // default security
                  PAGE_READWRITE,           // read/write access
                  0,                      // maximum object size (high-order DWORD)
                  MMFH_FM_SIZE,            // maximum object size (low-order DWORD)
                  sMmfName);                // name of mapping object

    dwErr = GetLastError();
    if (hMapFile == NULL)
    {
        _tprintf(TEXT("Could not create file mapping object (%d).\n"),
            GetLastError());
    }
    else if(dwErr == ERROR_ALREADY_EXISTS)
    {
        _tprintf(TEXT("MMF object named (%s) already exist - (%d).\n"),
            sMmfName,
            GetLastError());
    }
    else
        _tprintf(TEXT("<MMF object was successfully created.>\n"));
```

Listing 2: Vytvoření paměťově mapovaného souboru z C++

Čtení výsledků překladu může být realizováno například tak, že po ukončení SFC Editoru, spuštěného jako dceřiný proces, se přečte obsah paměťově mapovaného souboru a poté se soubor uzavře.

12.3 Komunikační rozhraní

Na obrázku obr. 30 je naznačeno zjednodušené schéma komunikace SFC Editoru (klienta) a serveru (nadřazená aplikace). Tento způsob komunikace mezi aplikacemi se používá pouze v monitorovacím módu editoru. Nejprve server vytvoří paměťově mapovaný soubor (MMF soubor). Paměťově mapovaný soubor je rozdělen na dvě části, jedna je určena pro vstupní data a druhá pro data výstupní (orientace je udávána z pohledu klienta).



Obrázek 30: Schéma komunikačního protokolu.

Dále musí server spustit SFC Editor, kterému jako parametr předá jméno vytvořeného paměťově mapovaného souboru. V dalším kroku server zapíše data do MMF (do části pro vstupní data) a nastaví událost zapsání vstupních dat do aktivního stavu. Po detekci události zapsání dat do vstupní části MMF souboru, client data načte a událost resetuje. Reset události umožní serveru poslat další data (pokud je to nutné).

Komunikace ve směru od klienta k serveru funguje obdobně. Pokud je výstupní událost (událost zapsání dat do výstupní části MMF souboru) v resetovaném stavu, zapíší se data do MMF a výstupní událost se nastaví do aktivní stavu. Server tuto událost zdetekuje, načte data a událost opět resetuje.

Jména vstupní a výstupní události, které jsou využívány pro řízení komunikace, jsou složeny z prefixu „InEvent_“ resp. „OutEvent_“ a jména MMF souboru.

Komunikace je ukončena serverem v případě uplynutí časového intervalu definovaného jako maximální doba, do které musí klient resetovat vstupní událost (načíst data předaná serverem).

Toto komunikační rozhraní není využíváno na přenos dat v editačním módu. Předání výsledné vygenerované tabulky přechodů a timeoutů se realizuje pouze přímým zápisem dat do MMF souboru, který vytvoří server při startu klenta.

12.3.1 Formát předávaných dat

Data, která jsou v monitorovacím režimů nadřazenou aplikací předávána SFC Editoru, obsahují informace o stavu realizovaného sekvenčního automatu. Na základě těchto dat

jsou v SFC schématu zvýrazněny odpovídající bloky.

V současné verzi editoru jsou očekávána data ve tvaru textového řetězce, který má 52 řádek. Pátý řádek tohoto řetězce udává počet vstupů, šestý řádek počet výstupů, sedmý řádek počet parametrů a osmý řádek pak počet stavů bloku (např. `ATMT`). Od patnáctého řádků následují hodnoty charakterizující stavy přechodů[9]. Od řádky 11 + počet vstupů, následují stavy kroků. Formát dat, který je zde popsán odpovídá monitorování bloku `ATMT`. Monitorovací data pro blok `EATMT` se liší pouze v tom, že hodnoty přechodů aR stavů jsou celočíselné datové typy, kde se využívá 16 spodních bitů k identifikaci stavu přechodu (stavu).

Logika, která implementuje deködování tohoto řetězce se může lišit v závislosti na typu schématu resp. na typu bloku, pro který je schéma sestaveno (`ATMT`/`EATMT`). Pokud nadřazená aplikace preferuje jiný formát předávaných dat, je nutné upravit třídu *Monitor_ATMT* resp. *Monitor_EATMT*, která se stará o jejich zpracování.

Reference

[1] IEC 61131-3 Final draft - IEC 61131-3, 2nd Ed. Programmable controllers - Programming languages

[2] **Funkční bloky systému REXYGEN**, Referenční příručka. REX Controls, Plzeň, leden 2010

[9] 0 – podmínka přechodu není splněna, 1 – podmínka přechodu je splněna.