IMPLEMENTACE SLOŽITÝCH ŘÍDICÍCH ALGORITMŮ Z PROGRAMU MATLAB – SIMULINK DO PROSTŘEDÍ PROGRAMOVATELNÝCH AUTOMATŮ

Bohumil Hnilička, Petr Pivoňka
ÚAMT FEI VUT v Brně

Abstract: Programy pro simulaci dynamických systémů zpravidla obsahují nástroje pro podporu návrhu řídicího algoritmu, realizaci, simulaci a ověření řídicího algoritmu v simulačním prostředí nebo často i na reálném procesu. Problém však nastává, pokud chceme daný řídicí algoritmus přenést do řídicího systému, který je určen pro trvalý provoz a má pro řízení příslušného procesu certifikát. Proto je nezbytná modifikace řídicího algoritmu na tvar vyhovující danému řídicímu systému. Často je nutné provést zásadní změny ve struktuře řídicího algoritmu. Přitom návrh a realizace složitého řídicího algoritmu a jeho implementace na reálný proces je časově značně náročná a s nejistým výsledkem. V prezentovaném příspěvku bude představeno nové řešení uvedených problémů s použitím programu MATLAB – Simulink a doplňkového programového vybavení určeného pro práci s programovatelným automatem firmy B&R.

1. Popis prostředí B&R AUTOMATION STUDIO

Pro práci s automatem bylo použito vývojové prostředí firmy B&R **Automation studio v. 1.3**. Při vytváření aplikace (projektu) v automatu se používají tzv. objekty, do kterých se umísťují vlastní sekvence programu. V našem případě využíváme jeden systémový objekt a dva cyklické objekty (první pro řízení reálné soustavy a druhý pro komunikaci s PC). Systémový objekt obsahuje základní systémové programy, které nelze modifikovat a jsou nutnou součástí projektu. Naproti tomu každý cyklický objekt má své jméno, periodu opakování, paritu a typ programovacího jazyka (programovací jazyk C nebo diagram pro zápis posloupnosti řídicích příkazů). Pro danou úlohu použijeme programovací jazyk C, který se jen v malých detailech liší od standardního ANSI C. Z tohoto důvodu je nutné dodržet tyto zásady:

- Pokud požadujeme, aby byla určitá část programu cyklicky opakována, musí být vložena do funkce typu CYCLIC (void).
- Jestliže požadujeme, aby se určitá část programu spouštěla pouze při inicializaci systému, musíme ji vložit do funkce INIT (void).

Více informací o programovatelném automatu B&R najdeme v [1].

2. Možnosti programu MATLAB pro simulaci v reálném čase

Výhodou prostředí programu MATLAB je bezproblémové předávání dat mezi jednotlivými toolboxy. V této práci byly použity především tyto toolboxy:

- **Real-Time Toolbox** od firmy HUMUSOFT umožňuje využití Simulinku v reálném procesu. Přidává schopnosti získávání dat v reálném čase, jejich okamžité zpracování a předání výsledků do reálného procesu [2].
- **Real-Time Workshop** dovoluje přímo automaticky generovat kód v jazyce C z blokového diagramu vytvořeného v Simulinku. Takto vytvořený program může být aktivován v reálném čase [3], [4].

V programu MATLAB existují dva typy funkcí – interní a externí. Interní procedury jsou součástí systému a nemohou být měněny. Z hlediska časové náročnosti však jen málo zatěžují systém. Naproti tomu externí funkce jsou realizovány pomocí M-files. Jedná se o textové soubory, a proto musí interpret MATLABu při každém vyvolání této funkce provádět postupně jednotlivé příkazy. Jako u všech podobných systémů je nevýhodou malá rychlost. Existuje však řešení v podobě souborů typu MEX. MEX soubor je program napsaný v jazyce C a přeložený do speciálního tvaru, který může být aktivován programem MATLAB. Tyto soubory jsou dynamicky linkované do paměti v okamžiku jejich prvého použití MATLABem a při jejich dalším použití již nedochází k časové prodlevě.

S-functions (system-functions) poskytují výkonný mechanismus pro rozšíření schopností Simulinku, protože dovolují přidat uživatelský algoritmus (blok) do modelu. Forma S-funkce je velmi obecná a vyhovuje spojitým, diskrétním i hybridním systémům. Zdrojový kód S-funkce může být napsán v jazyce MATLABu (M-file) nebo v jazyce C. Pro použití S-funkce, napsané v jazyce C, v Simulinku je nutné ji nejprve zkompilovat na MEX-file pomocí utility **mex**. Soubory typu MEX většinou slouží jako akcelerátory nebo drivery pro některé zařízení. Proto je výhodné zapsání algoritmů regulátorů do tohoto typu souboru. Tato metoda přináší následující výhody:

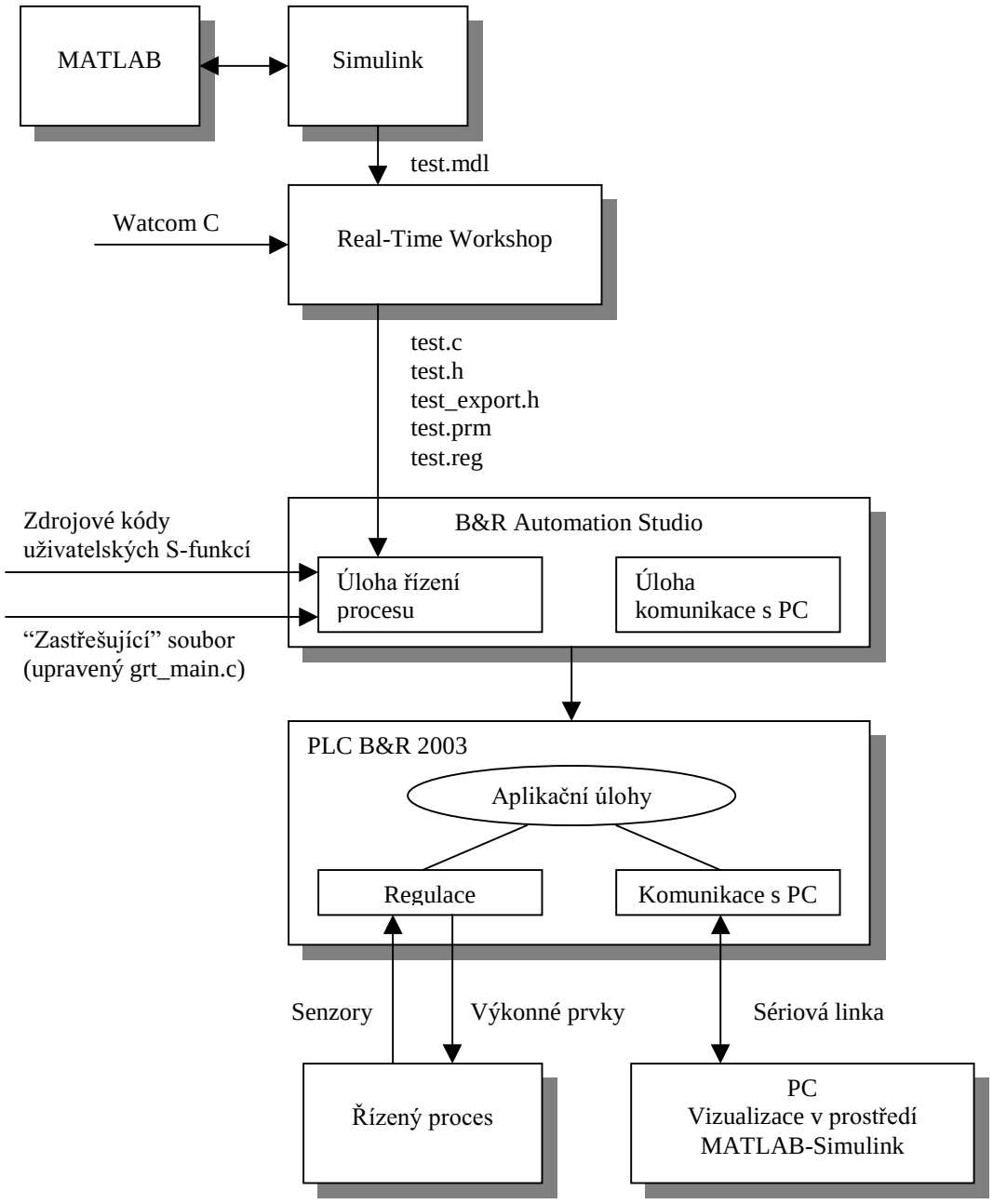
- program lze modifikovat a je rychlý
- zdrojový kód jediné S-funkce lze použít při realizaci regulátorů v programovatelném automatu
- vygenerovaný zdrojový kód celé aplikace v Simulinku lze s menšími úpravami použít při realizaci regulátorů v programovatelném automatu.

Tvorba MEX souborů a S-funkcí v jazyce C je však poměrně náročná a vyžaduje podrobné seznámení s firemní literaturou [5].

3. Metodika implementace složitých řídicích algoritmů z programu MATLAB – Simulink do prostředí programovatelných automatů

Implementaci lze rozčlenit na následující kroky:

- splnění předpokladů pro použití programu MATLAB a toolboxu Real-Time Workshop
- vytvoření regulační smyčky
- generování zdrojových kódů
- použití takto vytvořených zdrojových kódů v prostředí programovatelného automatu
- spojení reálného systému s globálními proměnnými v programovatelném automatu
- snímání regulovaných veličin z PC a vizualizace procesu na PC
- přeložení a aktivace projektu v programovatelném automatu.



Obr. 1 Blokové schéma implementace řídicích algoritmů do programovatelného automatu B&R 2003

4. Realizace přenosu zdrojových kódů do automatu B&R 2003

Pro použití Real-Time Workshopu při generování zdrojových kódů v jazyce C z blokového diagramu Simulinku je nutné splnění těchto předpokladů:

- Instalovaný překladač Watcom C v. 11.0.
- Blokové schéma se musí skládat z bloků ze standardních knihoven Simulinku nebo lze použít uživatelem vytvořené bloky, které ale musí být naprogramovány jako S-funkce v jazyce C.
- Některé bloky ze standardních knihoven nelze použít (např. Display, Scope, From Workspace, To Workspace atd.).
- Pokud chceme měnit některou z veličin nebo ji používat pro monitorování, je nutné ji v blokovém diagramu zapojit pomocí bloků In/Out.
- Nastavení volitelných parametrů Real-Time Workshopu, nabídka **RTW Options…,** která se nachází v roletovém menu Tools.

Při splnění těchto požadavků je možné vygenerovat zdrojové kódy pomocí příkazu **RTW Build**, který se nachází v roletovém menu Tools. Tímto překladač vygeneruje řadu souborů, z nichž se do cyklického objektu pro řízení reálné soustavy přidají všechny s příponami: .c, .h, .reg, .prm. Dále se do tohoto objektu musí přidat všechny zdrojové kódy uživatelských S-funkcí, které se v blokovém diagramu použily. Poslední soubor, který je nutné do objektu přidat, ”zastřešuje” všechny doposud přidané soubory a musí si jej uživatel vytvořit sám. V tomto souboru je hlavní funkce main(), kterou je ale nutné rozčlenit na dvě funkce INIT (void) a CYCLIC (void) z důvodu uvedeném v první kapitole tohoto článku. Tvorba tohoto souboru není zcela jednoduchá a vyžaduje pochopení hlavní funkce grt_main.c podle které pracuje celý Simulink.

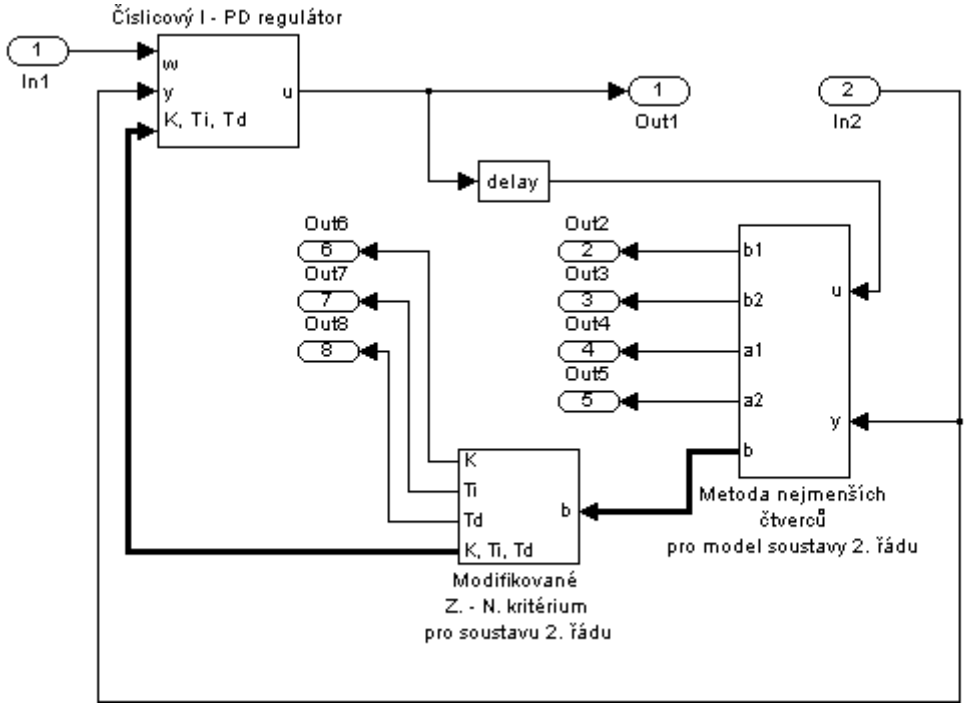
Poslední úpravou, kterou je nutné provést, je spojení bloků In/Out, kterými bylo vyřešeno napojení na regulovanou soustavu, zadávání veličin z PC a monitorování, s globálními proměnnými v automatu. Tyto globální proměnné je potřebné deklarovat pomocí funkce GLOBAL. Propojení lze provést úpravou souboru s příponou .reg, ve kterém najdeme odkazy na bloky In/Out. Dále nastavíme cesty k hlavičkovým souborům do MATLABu pro překladač v B&R Automation studio. Přeložené soubory zašleme po síti do programovatelného automatu.

Druhý objekt projektu slouží pro spojení programovatelného automatu s PC, které monitoruje, komunikuje a mění potřebné parametry regulátoru. Všechny hodnoty, které chceme sledovat či měnit musí být uloženy v globálních proměnných automatu. Spojení mezi automatem a PC bylo provedeno pomocí sériové linky. Pro monitorování veličin z automatu a zadávání potřebných parametrů do automatu bylo použito opět programu MATLAB – Simulink. Protože v Simulinku není nástroj na obsluhu sériového portu, bylo nutné tuto obsluhu realizovat. Byl proto vytvořen MEX soubor a jelikož v tomto souboru jsou použity API funkce, je činnost obsluhy velice rychlá.

5. Ukázka možného použití

Adaptivní regulátor je realizován podle obr. 2, kde je ukázáno schéma zapojení upravené do formy vhodné pro další generování zdrojových kódů. Průběžná metoda nejmenších čtverců pro model 2. řádu s exponenciálním zapomínáním je použita k identifikaci regulované soustavy realizované pomocí fyzikálního reálného modelu. V bloku ”Modifikovaná metoda Zieglera – Nicholse” je v každém kroku výpočtu určeno kritické zesílení a kritická perioda kmitů modelu procesu, ke kterému se použily výsledky identifikace. Zároveň jsou v tomto bloku vypočteny parametry K, T_I, T_D regulátoru I-PD za použití Takahashiho algoritmu podle [6]. Průběh žádané hodnoty je zadáván z PC, proto je potřeba místo bloku s nastavením žádané hodnoty použit blok In1. Připojení regulované soustavy je realizováno pomocí bloků In2, Out1. Ostatní veličiny pro monitorování v PC jsou vyvedeny bloky Out2 – Out8.

Pokud předpokládáme, že model v Simulinku je označen test.mdl, potom z generovaných souborů získaných programem Real-Time Workshop v automatu použijeme soubory: test.c, test.prm, test.reg, test.h, test_export.h. V souboru s příponou .reg jsou uložena data s nastavením jednotlivých bloků, které byly v simulaci použity. Chceme-li tedy nastavit jiné konstanty regulačního algoritmu při tvorbě projektu do programovatelného automatu, než které byly použity při generování zdrojových kódů, stačí modifikovat právě soubor s příponou .reg. Snadná změna tohoto souboru, ve kterém jsou hodnoty nastavení použitých bloků patřičně okomentovány, umožňuje urychlení celé práce a dovoluje širší použití již jednou vygenerovaných souborů, předpokládáme-li zachování struktury blokového zapojení v prostředí Simulink.



Obr. 2 Blokový diagram s adaptivním regulátorem

6. Závěr

V článku uvedený originální postup dovolí zásadním způsobem ovlivnit návrh, ověřování a přímou implementaci řídicích algoritmů na reálný proces. Při jeho použití se dá předpokládat nejen výrazné urychlení celého procesu implementace, ale i podstatné omezení chyb vznikajících při přepisech řídicích algoritmů. Postup byl s úspěchem použit nejen pro přenos klasických řídicích algoritmů z programu MATLAB – Simulink do programovatelného automatu B&R, ale i pro paměťově náročné soubory a řídicí algoritmus fuzzy regulátoru. Metoda je vhodná nejen pro pevně nastavené regulátory, ale i pro adaptivní regulátory a pro regulátory s prvky umělé inteligence. Podrobnější popis najdeme v [7].

7. Literatura

[1] SYSTÉM B&R 2000. Firemní literatura. B&R automatizace s.r.o., Brno, 1999.
[2] REAL TIME TOOLBOX for use with MATLAB User's manual. HUMUSOFT s.r.o., Praha, 1992 – 1998.
[3] The MathWorks Inc.: Real-Time Workshop User's Guide, The MathWorks Inc., 1999.
[4] The MathWorks Inc.: Real-Time Windows Target, The MathWorks Inc., 1999.
[5] The MathWorks Inc.: Writing S-Functions, The MathWorks Inc., 1998.
[6] BOBÁL, V. – BŐHM, J. – PROKOP, R. – FESSEL, J.: Praktické aspekty samočinně se nastavujících regulátorů: algoritmy a implementace, VUTIUM, Brno, 1999.
[7] HNILIČKA, B.: Adaptivní řízení tepelných procesů. Diplomová práce VUT FEI Brno, 2000.
[8] HNILIČKA, B.: Přenos řídicích algoritmů z programu MATLAB-Simulink do programovatelného automatu B&R 2003, Sborník prací studentů a doktorandů FEI VUT, VI. (2000), CERM, Brno, 2000, s. 20-22.

VUT - FEI
Ústav automatizace a měřicí techniky
Božetěchova 2
612 66 Brno

e-mail: hnilicka@dame.fee.vutbr.cz