

Funkční bloky pro pokročilé regulační algoritmy

Miloš SCHLEGEL, Pavel BALDA

Západočeská univerzita v Plzni, ZAT a.s. Příbram

V příspěvku jsou stručně popsány některé zajímavější funkční bloky z připravované knihovny funkčních bloků pro průmyslovou regulaci. Podstatné je, že tato knihovna může být použita jak v programovém systému Matlab-Simulink, tak i na libovolně otevřené platformě (jazyk C, C++) procesních stanic s operačním systémem reálného času (VxWorks, WinCE, OS9).

Klíčová slova: PID, autotuner, Matlab, Simulink, funkční blok.

Some interesting function blocks from a new industrial control block library are described in this paper. The substantial feature of the library is the fact that the library can be used in Matlab-Simulink program package as well as in an arbitrary open platform (C, C++ languages) process station equipped by some real time operating system (VxWorks, WinCE, OS9).

Keywords: PID, autotuner, Matlab, Simulink, function block.

1. Úvod

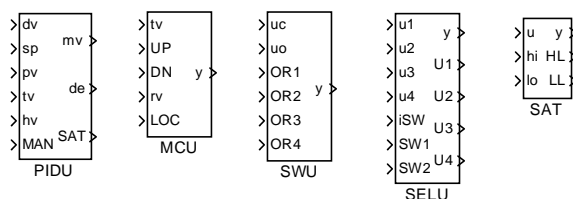
Pověstná propast mezi teorií a praxí v oboru automatického řízení technologických procesů rozděluje tvůrčí pracovníky v této oblasti na čisté teoretiky a přízemní praktiky. Prvé nalezneme nejčastěji na univerzitách, kde pro své výzkumy končící většinou simulačním ověřením vyvinutých algoritmů hojně využívají programový systém Matlab-Simulink. Ty druhé pak přímo v terénu při uvádění do provozu, kde svoji inženýrskou intuici převádějí do algoritmů pomocí nejrůznějších programovacích jazyků a konfiguratorů. Navzdory rutinní skutečnosti si mnozí praktikové uvědomují (zvláště ve složitějších případech), že by bylo efektivnější nejdříve celý problém modelovat a počítačovou simulací nalézt nejvhodnější strategii řízení. Bohužel tento postup není obvykle dostatečně podporován ve vývojových systémech výrobců programovatelných automatů a distribuovaných řídicích systémů. Použití populárního systému Matlab-Simulink pro výše zmíněný praktický problém má však též jednu velkou nevýhodu: řídicí algoritmus v podobě blokového schématu se standardními bloky Simulinku není vhodný pro průmyslové účely. Navíc jej nelze ani jednoduše a bez extrémních nákladů převést do programovacího jazyka ANSI C, který je většinou k dispozici v otevřených programovatelných automatech.

Cílem projektu, který je velmi stručně popsán v tomto příspěvku, je vytvořit technologii, která umožňuje efektivně využívat systém Matlab-Simulink pro vývoj pokročilých průmyslově orientovaných regulačních algoritmů. Ke splnění uvedeného záměru je zřejmě třeba vytvořit speciální knihovnu funkčních bloků pro Simulink a příslušné programové nástroje pro akceptování souborů Simulinku s příponou **mdl** jako zdrojového popisu řídicího algoritmu co možná největším počtem programovatelných automatů.

Příspěvek se omezuje na stručný popis několika zajímavých „regulačních“ bloků připravované univerzální knihovny určené pro konfiguraci řídicích algoritmů regulačního i logického typu.

2. Bloky pro průmyslovou regulaci

V této kapitole stručně popíšeme sadu funkčních bloků (viz obr. 1), pomocí které lze pohodlně a kvalitně realizovat všechny obvyklé průmyslové regulační struktury jako vlečná, poměrová, kaskádní a selektorová regulace. Bloky jsou navrženy tak, aby je bylo možno snadno pospojovat do požadované regulační funkce a aby mohlo být jednoduše a korektně realizováno bezrázové přepínání veškerých režimů činnosti i parametrů bloků za běhu řídicího systému.



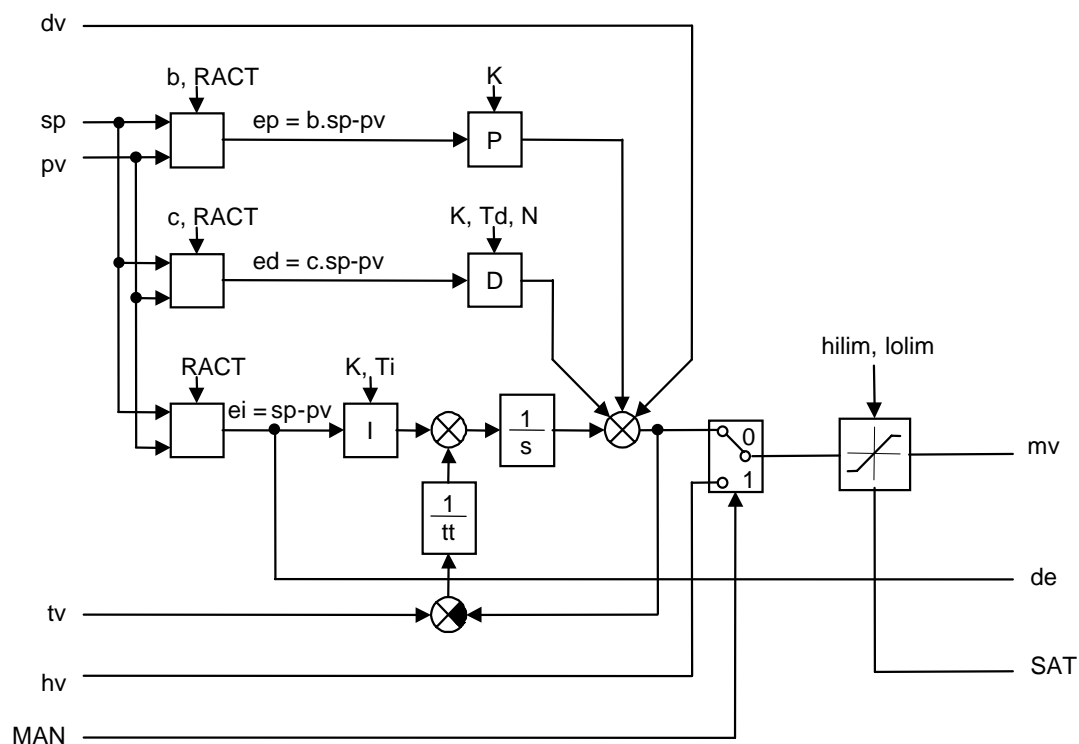
Obr. 1: Sada základních funkčních bloků pro průmyslovou regulaci

2.1. PIDU – Jednotka PID regulátoru

Blok **PIDU** je základní blok pro vytvoření úplného regulátoru PI(D) nebo regulátoru PI(D)S, tj. PI(D) regulátoru doplněného Smithovým prediktorem. Blok generuje akční veličinu **mv**, regulační odchylku **de** a příznak saturace **SAT** v automatickém (**MAN** = „0“) a manuálním režimu (**MAN** = „1“). Vhodným zapojením vstupu pro vysledování **tv** lze dosáhnout bezrázového přepínání režimů a odstranění tzv. unášení integrační složky (wind up effect) při saturaci regulátoru a to dokonce i ve složitých regulačních obvodech (např. v kaskádní regulaci, atd.). V automatickém režimu v lineární oblasti realizuje blok zákon řízení daný vztahem

$$U = \pm K \left[bW - Y + \frac{1}{T_i s} E + \frac{T_d s}{T_d s / N + 1} (cW - Y) \right] + Z$$

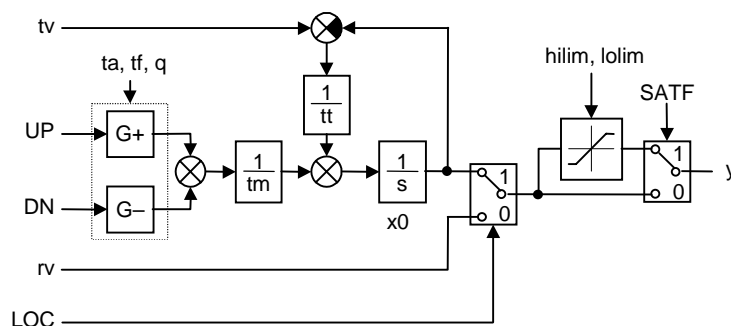
kde U je obraz akční veličiny **mv**, W je obraz požadované hodnoty **sp**, Y je obraz regulované veličiny **pv**, Z je obraz dopředné vazby od měřitelné poruchy **dv** a K , T_i , T_d , N , b , c odpovídají základním parametrům **k**, **ti**, **td**, **N**, **b** a **c** regulátoru. Znaménko pravé strany je určeno parametrem **RACT**, určujícím směr působení akční veličiny na regulovanou veličinu. Parametr **IRTYPE** určuje typ regulátoru, parametry **b**, **c** umožňují seřídit regulátor jak pro optimální odregulování poruchy, tak i pro optimální sledování požadované veličiny (regulátor se dvěma stupni volnosti). Rozsah výstupu **mv** je omezen saturačními mezemi **lolim** a **hilim**. V manuálním režimu je vstup **hv** (po případném omezení) kopírován na výstup **mv**. Signál připojený na vstup **tv** zajišťuje v tomto režimu (stejně jako ve všech případech, kdy je zpětnovazební smyčka rozpojena) příslušné vysledování vnitřního stavu regulátoru pro následné bezrázové přepnutí do automatického režimu. Funkce bloku je dobře patrná z obr. 2 se zjednodušeným vnitřním schématem regulátoru.



Obr. 2: Zjednodušené vnitřní schéma bloku **PIDU**

2.2. MCU – Jednotka pro ruční zadávání

V lokálním režimu ($LOC = „1“$) je blok **MCU** určen k ručnímu zadávání výstupu y pomocí tlačítek „více“ (vstup **UP**) a „méně“ (vstup **DN**). Strmost najíždění z počáteční hodnoty y_0 na žádanou hodnotu je určena integrační časovou konstantou tm a dobou stlačení ovládacích tlačítek. Po uplynutí každých ta sekund je strmost vždy násobena faktorem q , až do vypršení doby tf . Rozsah výstupu y může být omezen ($SATF = „1“$) saturačními mezemi $lolim$ a $hilim$. V případě, že žádné z tlačítek není stlačeno ($UP = „0“$ a $DN = „0“$), vysleduje výstup y vstupní hodnotu tv . Rychlost výsledování je dána integrační časovou konstantou tt . V případě $LOC = „0“$ je vstup rv s případnými omezeními ($SATF = „1“$) kopírován na výstup y . Podrobná funkce bloku je přímo patrná z obr. 3, zobrazujícího vnitřní schéma bloku.



Obr. 3: Vnitřní schéma bloku **MCU**

2.3. SWU – Přepínač vstupu pro výsledování

Blok **SWU** je určen pro přepínání vhodného signálu na vstup pro výsledování bloků **PIDU** a **MCU**. Jeho funkce je velmi jednoduchá. V případě, že všechny logické vstupy **OR1**, ..., **OR4** jsou shozeny, potom na výstup y je kopírována hodnota vstupu uc , v opačném případě hodnota vstupu uo .

2.4. SELU – Selektor aktivního regulátoru

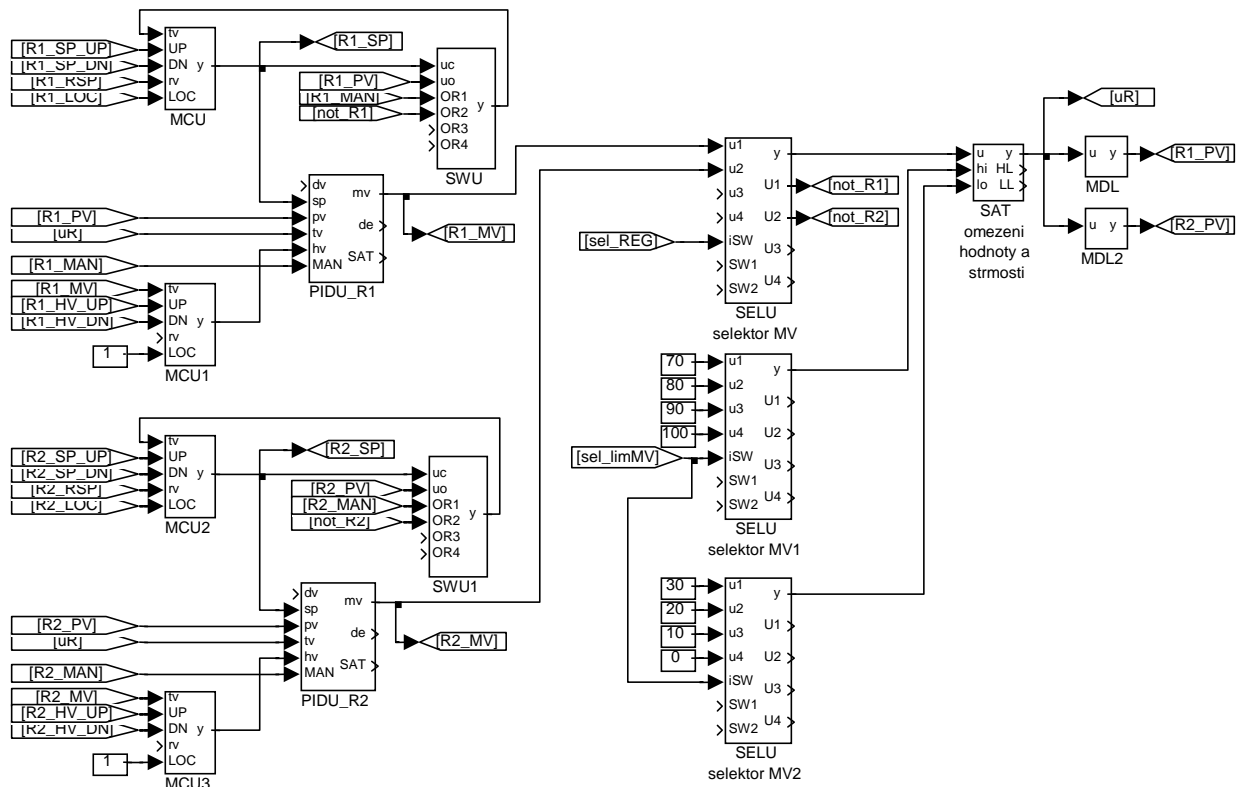
Blok **SELU** je určen pro přepínání aktivního regulátoru v případě selektorové regulace. Provádí výběr jednoho ze vstupních signálů $u1$, $u2$, $u3$, $u4$ a kopíruje ho na výstup y buď podle celočíselného vstupu iSW (je-li parametr bloku **BINF** = „0“) nebo podle binárních vstupů **SW1** a **SW2** (**BINF** = „1“) dle následující tabulky.

iSW	SW1	SW2	y	U1	U2	U3	U4
0	0	0	$u1$	0	1	1	1
1	0	1	$u2$	1	0	1	1
2	1	0	$u3$	1	1	0	1
3	1	1	$u4$	1	1	1	0

Z této tabulky je patrný též význam logických výstupů **U1**, **U2**, **U3**, **U4**, které se používají jako vstupy bloků **SWU** pro realizaci funkce výsledování neaktivních regulátorů v selektorové regulaci

2.5. SAT – Saturace výstupu s proměnnými mezemi

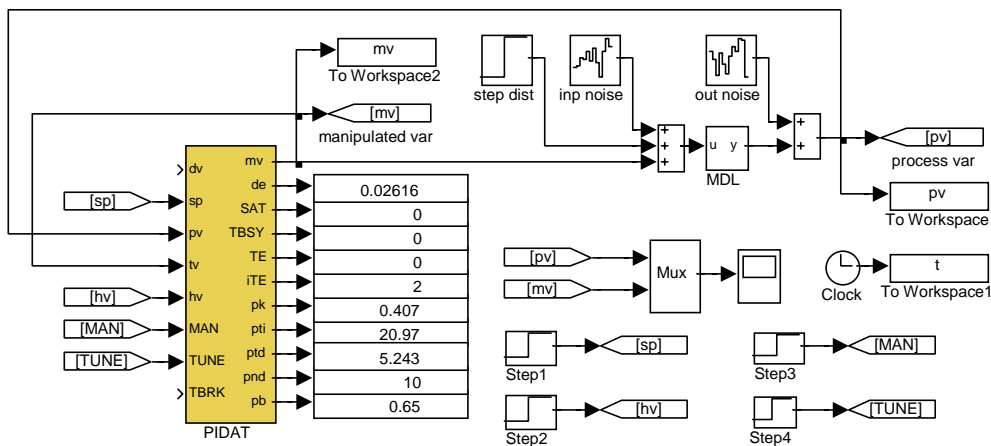
Blok **SAT** kopíruje vstup u do výstupu y , pokud je vstupní veličina v intervalu $\langle lolim, hilim \rangle$. Je-li $u < lolim$ ($u > hilim$) potom $y = lolim$ ($y = hilim$). Horní a dolní limit jsou buď pevné hodnoty dané po řadě parametry bloku $hilim0$ a $lolim0$ (případ **HLD** = „1“, **HLD** je další parametr bloku) nebo jsou řízeny vstupy hi a lo (**HLD** = „0“). Maximální rychlost změny aktivních mezí $hilim$ a $lolim$ je dána časovými konstantami tp a tn . Parametr tp určuje maximální kladnou strmost a tn maximální zápornou strmost změny $hilim$ a $lolim$. Výstupy **HL** a **LL** signalizují po řadě horní a dolní saturaci. Příklad užití bloků **PIDU**, **MCU**, **SWU**, **SELU** a **SAT** je uveden na obr. 4.



Obr. 4: Příklad použití bloků **PIDU**, **MCU**, **SELU** a **SAT** pro selektorovou regulaci

3. Inteligentní regulátor

Prudký rozvoj mikroelektroniky v nedávné době umožnil, aby základní řídicí algoritmus průmyslového regulátoru byl doplněn dalšími expertními funkcemi, které dříve vykonával pouze drahý živý expert. V průmyslu nejvíce žádanou takovou funkcí je automatické nastavování parametrů regulátoru. V posledních patnácti letech bylo uvedeno na trh mnoho kompaktních regulátorů se zabudovaným autotunerem, který na povel operátora určitým způsobem vybudí řízený systém (v otevřené nebo uzavřené smyčce) a ze změřené odezvy automaticky určí „optimální“ parametry regulátoru. Podobný vývoj probíhá v distribuovaných řídicích systémech. Téměř všechny v průmyslu používané autotunery jsou založené na skoro 60 let starých empirických metodách Zieglera a Nicholse [1], které potřebují k návrhu regulátoru pouze jistá dvě charakteristická čísla řízeného systému. To je důvod, proč jsou tyto metody tak populární. Bohužel jejich spolehlivost je velmi nízká [2], [3]. V [3] je provedena exaktní revize Zieglerovy-Nicholsovy frekvenční metody na základě řešení nově zformulované úlohy robustního návrhu regulátoru. Odtud vyplývá, že pro návrh vyhovujícího PI(D) regulátoru skutečně stačí znát pouze jeden chytré vybraný bod frekvenční charakteristiky řízeného systému. Na rozdíl od Zieglerovy-Nicholsovy metody však není fázové zpoždění tohoto bodu rovné 180° , ale je závislé na typu navrhovaného regulátoru a požadované bezpečnosti ve stabilitě. Konečným výsledkem výše zmíněné revize je blok **PIDAT** – regulátor s robustním autotunerem (viz obr. 5).



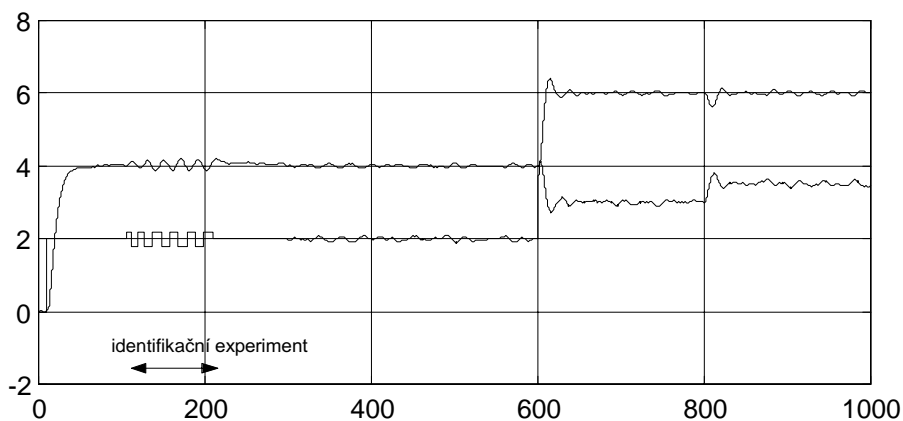
Obr. 5: Funkční schéma příkladu použití bloku **PIDAT**

Blok **PIDAT** má základní regulační funkci identickou s výše popsaným blokem **PIDU**. Jeho další vstupy a výstupy se týkají funkce autotuneru. Vstupem **TUNE** se spouští identifikační experiment, ve kterém se určí jeden vhodný bod frekvenční charakteristiky. V této fázi (**TBSY** = „1“) je činnost standardního PID regulátoru nahrazena pomocným reléovým regulátorem s hysteresí doplněným adaptivním fázovacím filtrem, který zařídí vybuzení řízeného systému na frekvenci odpovídající požadovanému fázovému zpoždění. Amplitudu vybuzení lze řídit parametrem **amp**. Činnost bloku **PIDAT** po ukončení experimentu závisí na tom, zda identifikace proběhla úspěšně (**TE** = „0“). V kladném případě jsou původní parametry regulátoru nahrazeny nově určenými (podle požadovaného typu regulátoru) a regulátor dále pracuje ve stejném režimu, v jakém pracoval před experimentem. V záporném případě se parametry nemění. Všechny parametry určené autotunerem jsou vyvedeny na výstupy bloku **PIDAT** (**pk**, **pti**, **ptd**, **pnd** a **pb** odpovídají po řadě parametrům **k**, **ti**, **td**, **nd** a **b** popsaným u bloku **PIDU**). Logickým vstupem **TBRK** lze předčasně ukončit identifikační experiment. V průběhu experimentu se na výstupu **iTE** aktualizuje odhad času do konce ladící fáze, což může být užitečné zvláště u velmi pomalých procesů. Po ukončení experimentu je na tomto výstupu nastaven chybový kód informující o přesnosti provedeného naladění regulátoru.

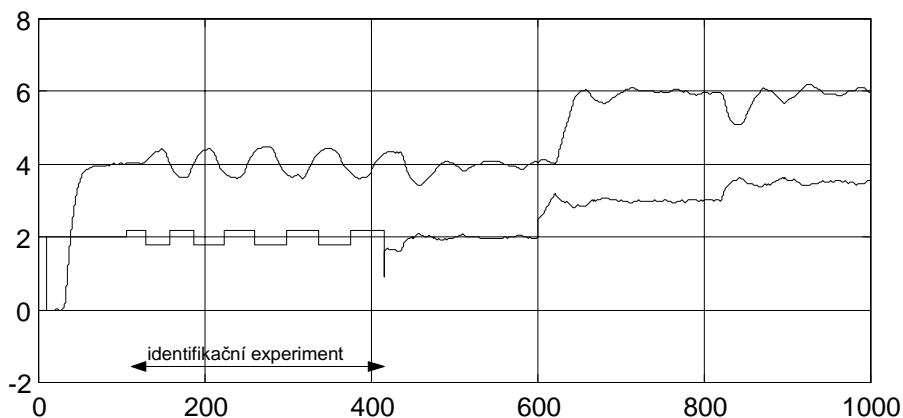
Z uživatelského hlediska je využití autotuneru velmi jednoduché. Od operátora se pouze vyžaduje, aby uvedl řízený systém při vhodných pracovních podmínkách do ustáleného stavu (v manuálním nebo automatickém režimu s předběžnými parametry regulátoru), zadal požadovaný typ regulátoru (**PI**, **PID**) a parametr **amp**, určující přípustnou mez změny akční veličiny a konečně spustil identifikační experiment vstupem **TUNE**.

Funkce autotuneru je ilustrována na obr. 6 a 7, kde je zobrazen průběh procesu automatického nastavování pro případ řízeného systému s přenosem

$$F(s) = \frac{2e^{-Ds}}{(5s + 1)^2}.$$



Obr. 6: Nastavování parametrů regulátoru **PIDAT** vestavěným autotunerem pro $D = 1$.



Obr. 7: Nastavování parametrů regulátoru **PIDAT** vestavěným autotunerem pro $D = 20$.

4. Závěr

Všechny bloky uvedené v tomto příspěvku byly vytvořeny výše popsanou „technologíí“. Jejich algoritmy byly nejprve napsány v jazyku ANSI C a odladěny v systému Matlab-Simulink s využitím diagnostických nástrojů tohoto systému. Po důkladném otestování všech funkcí byl již odladený zdrojový kód přenesen do cílového řídicího systému. Kromě výrazného zkrácení doby vývoje pokročilých bloků, bylo umožněno vyvinout tak složité funkční bloky jako např. blok PID regulátoru s vestavěným autotunerem (**PIDAT**).

Autoři tohoto příspěvku věří, že tato technologie vývoje regulačních systémů může sblížit teoretiky a praktiky v oboru průmyslové regulace a že v důsledku toho mohou vzniknout nové důmyslnější řídicí systémy.

5. Literatura

- [1] Ziegler, J. G. – Nichols, N. B.: Optimum settings for automatic controllers. Trans. ASME, 1942, 64, 759-768
- [2] Aström, K.J. – Hägglund, T.: PID Controllers: Theory, Design and Tuning. Instrument Society of America, Research Triangle Park, NC, second edition, 1995.
- [3] Schlegel, M.: Nový přístup k robustnímu návrhu průmyslových regulátorů. Habilitační práce. Západočeská univerzita v Plzni, 2000.