

# VYUŽITÍ KNIHOVNY SWING PROGRAMOVACÍHO JAZYKU JAVA PŘI TVORBĚ UŽIVATELSKÉHO ROZHRANÍ SYSTÉMU "HOST PC - TARGET PC" PRO ŘÍZENÍ POLOVODIČOVÝCH MĚNIČŮ

*Stanislav Flígl*

Katedra elektrických pohonů a trakce (K13114), FEL ČVUT v Praze

**Abstrakt.** Pro realizaci číslicového řídicího systému, obzvláště v laboratorních podmínkách, je v současné době k dispozici celá řada již hotových hardwarových řešení, z kterých si lze vybrat. Příspěvek popisuje netypický systém „Host PC - Target PC“ složený ze dvou osobních počítačů, z nichž jeden pracuje v reálném čase (Target) a druhý slouží k zadávání příkazů a vyhodnocování výsledků (Host). Target PC je pro dosažení maximální rychlosti programován nízkourovňově v programovacím jazyku C. V nadřazeném systému je pro ovládání k dispozici monitorovací program vytvořený s využitím knihoven jazyka Java integrovaných do prostředí Matlabu. Prezentovaný systém sice nemůže konkurovat v komfortu obsluhy komerčním systémům, ale v některých výkonostních parametrech je výrazně předstihuje.

## 1. Úvod

Hlavními kritérii hodnocení řídicích systémů jsou především výpočetní výkon, integrované periférie, možnost průmyslového nasazení a v neposlední řadě také cena. Požadavky na ně kladené v případě elektrických pohonů resp. polovodičových měničů jsou obecně vysoké. Vhodný řídicí systém musí zajistit:

- digitalizaci měřených veličin
- výpočet matematického modelu regulovaného zařízení
- časově naprosto přesné generování řídicích pulsů pro jednotlivé spínací prvky
- komunikaci s nadřazeným systémem

V literatuře se uvádí i konkrétní číselné hodnoty pro maximální délku vzorkovací periody. Jako příklad bývá dáván pohon robotů, kde se předpokládá dosažení jmenovitých otáček za dobu 30 až 50 ms. Pro požadovanou přesnost nastavení úhlu nesmí přesáhnout vzorkovací perioda 100 $\mu$ s.

Obdobná je i situace u jiných pohonů, kde se požaduje vysoká dynamika. Při řízení asynchronního motoru stoupá požadavek na výpočetní výkon vzhledem k nutnosti řešení jeho matematického modelu v reálném čase. Současným trendem jsou pohony bez snímačů otáček. Tím se pak komplikovanost matematických modelů ještě zvětšuje.

Spínací frekvence se u měničů v závislosti na výkonu pohybuje řádově od 500 Hz do 10 kHz. Pulsy musí být běžně generovány s přesností výrazně pod 1 $\mu$ s. To vyžaduje přítomnost speciálních obvodů, které to dokáží zajistit.

## 2. Přehled existujících řešení

Procesory běžně používané v aplikacích s elektrickými pohony je možné rozdělit do následujících kategorií (v závorce je uvedeno vždy několik příkladů i s orientačním výkonem v miliónech instrukcí za sekundu):

- 16bitové mikrokontrolery založené na von Neumannově architektuře (Intel 8xC196, Infineon SAB 8xC166 - 10 MIPS)
- 16bitové signálové procesory s pevnou řádovou čárkou (Motorola DSP56F80x - 40 MIPS, Analog Devices ADSP 2100a ADMC200/201 - 20 až 160 MIPS, Texas Instruments C24x - 20 až 40 MIPS)
- 32bitové signálové procesory s pevnou řádovou čárkou (Texas Instruments C28x - až 400 MIPS)
- signálové procesory s plovoucí řádovou čárkou (Analog Devices ADSP 21K - 40 až 100 MIPS, Texas Instruments TMS320C3x - 20 až 40 MIPS/MFLOPS)

Procesory se pak značně liší integrovanými perifériemi pro komunikaci s okolím, obvody pro generování spínacích pulsů, počtem digitálních vstupů/výstupů, rychlostí, počtem A/D a D/A převodníků a v neposlední řadě pamětí na čipu (typicky desítky KB). Méně výkonné zástupce je nutné programovat výlučně v assembleru, u rychlejších je to možné alespoň částečně v programovacím jazyce C.

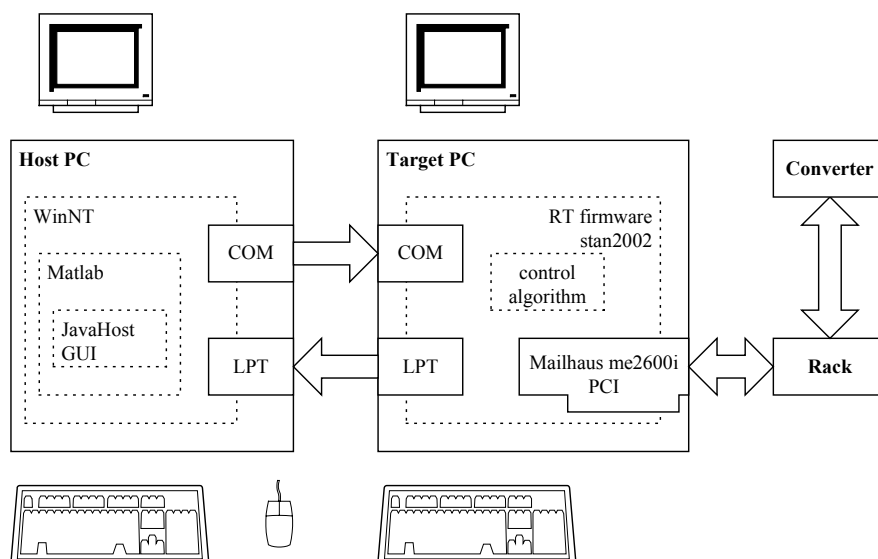
Vývoj v oblasti procesorů probíhá velmi rychle. Některé typy procesorů ve výše uvedeném stručném přehledu jsou sice již zastaralé, ale i přesto je patrné značné zaostávání signálových procesorů za procesory určenými pro osobní počítače (Intel Pentium 4 3 GHz - 9000 MIPS a 2600 MFLOPS). Pořizovací ceny průmyslových řídicích systémů nejsou malé a tak se nabízí otázka, zda by nešlo využít běžný osobní počítač k řízení měničů alespoň v laboratorním prostředí.

Běžné PC ale samo o sobě vyhovuje našim požadavkům na řídicí systém pro polovodičové měniče pouze ve výpočetním výkonu. Při praktickém nasazení osobních počítačů jakožto cílové platformy je nutné PC dovybavit vhodným hardwarem a softwarem.

- PC s obyčejnou I/O kartou a multiúlohovým operačním systémem: Jde o levné řešení s nesčetnými praktickými realizacemi, možností rychlého nasazení a na trhu je k dispozici řada specializovaných softwarových produktů, které je možné k použití k vývoji potřebného programového vybavení i pro různé operační systémy. Takovýmto způsobem je možné vytvořit řídicí aplikaci běžící v reálném čase s regulační periodou pouze v řádu ms. Tento systém je tedy pro řízení měničů nepoužitelný.
- PC s dSpace kartou a multiúlohovým operačním systémem: Karty firmy dSpace jsou typickým představitelem karet vybavenými kromě převodníků analogových veličin také podpůrnými signálovými procesory (20-400 MIPS). Vlastní řídicí algoritmus je vykonáván signálovým procesorem na přídavné kartě a osobní počítač slouží pouze k vytvoření kódu pro cílový procesor a následnému zpracování a zobrazení výsledků, které jsou většinou přístupné přes dvoubránovou paměť. Důsledně vzato nepatří takovýto systém do kategorie řídicích systémů na bázi PC.
- PC s obyčejnou I/O kartou a jednoúlohovým operačním systémem: Tato varianta je představována osobním počítačem, nad jehož chodem převezme plně kontrolu řídicí program měniče a jiné programy v době jeho činnosti není možné spouštět. Výhodou je již zmíněný vysoký výpočetní výkon daný frekvencí, na které pracují současné procesory osobních počítačů, tj. frekvence v řádu GHz. Pro toto řešení mluví i nízká pořizovací cena vhodné I/O karty. Navíc se dá rychlost systému jednoduše zvýšit koupí nového procesoru, resp. základní desky. Nevýhodou je, že se jedná o naprosto nestandardní řešení, které vyžaduje značné množství vývojové práce (vývoj přídavného hardwaru, nízkourovňové programování) ještě před realizací prvního řídicího algoritmu. Problémem je i dlouhá reakce na přerušení (od 4 resp. 8  $\mu$ s) a její jitter, tj. její rozptyl, se kterým je při obsluze přerušení nutné počítat.

### 3. Systém Host PC - Target PC a jeho výpočetní jádro pro funkci pro práci v reálném čase

Na základě výše uvedených úvah bylo přistoupeno k vývoji systému založenému na nízkourovňově naprogramovaném osobním počítači v konfiguraci tak, jak je uvedeno v posledním bodě výše uvedeného přehledu – dále označován jako Target PC (cílový počítač). Po doplnění o další počítač vybavený monitorovacím programem vznikne systém označovaný v dalším jako Host PC – Target PC.



Obr. 1. Uspořádání řídicího systému a měniče

Jednotlivé elementy celého systému jsou patrné z obr. 1. Softwarové komponenty jsou orámovány čárkovaně a hardwarové plnou čarou. Jak je z obrázku patrné, je systém doplněn ještě o skříň se zásuvnými kartami (rack), v které jsou rozvedeny analogové a digitální signály karty Meilhaus umístěné v počítači. Pro řízení měničů je nutné systém doplnit o kartu s vyrovnávací pamětí pro generování spínacích pulsů. V běžné aplikaci je nutné generovat 3 až 24 spínacích pulsů s přesností výrazně pod 1  $\mu$ s a to není osobním počítačem vybaveným běžnou vstupně výstupní kartou realizovatelné.

Pro snadnější práci s tímto systémem jsem vytvořil firmware pro cílový počítač (Target PC) a monitorovací program pro Host PC.

Obr.2. Vzhled servisní obrazovky Target PC

Firmware je sadou nízkoúrovňových knihoven napsaných v Assembleru a jazyku C. Obsahuje synchronizační rutiny pro práci v reálném čase, inicializaci a obsluhu převodníků a karet pro generování spínacích pulsů, komunikační rutiny pro přenos dat po sériovém a paralelním portu, obslužné rutiny klávesnice a obrazovky v textovém režimu a funkci osciloskop pro záznam rychlých dějů. Pro uživatele je připraveno jednoduché aplikační rozhraní. Ten musí napsat proceduru v jazyce C, která na základě naměřených hodnot a příkazů zaslaných z nadřazeného systému vypočítá odpovídající regulační zásah, potažmo tomu odpovídající spínací sekvence výkonových prvků. Po kompilaci vznikne 32bitová dosová aplikace. Na obr. 2 je zobrazen typický vzhled monitoru cílového počítače po spuštění této aplikace.

#### 4. Matlab jako výchozí platforma hostujícího počítače

Matlab je ideálním prostředím pro zpracování, analýzu a zobrazení průběhů měřených veličin. Protože jde zároveň o systém univerzálně programovatelný, rozhodl jsem se implementovat uživatelské rozhraní řídicího systému přímo v něm.

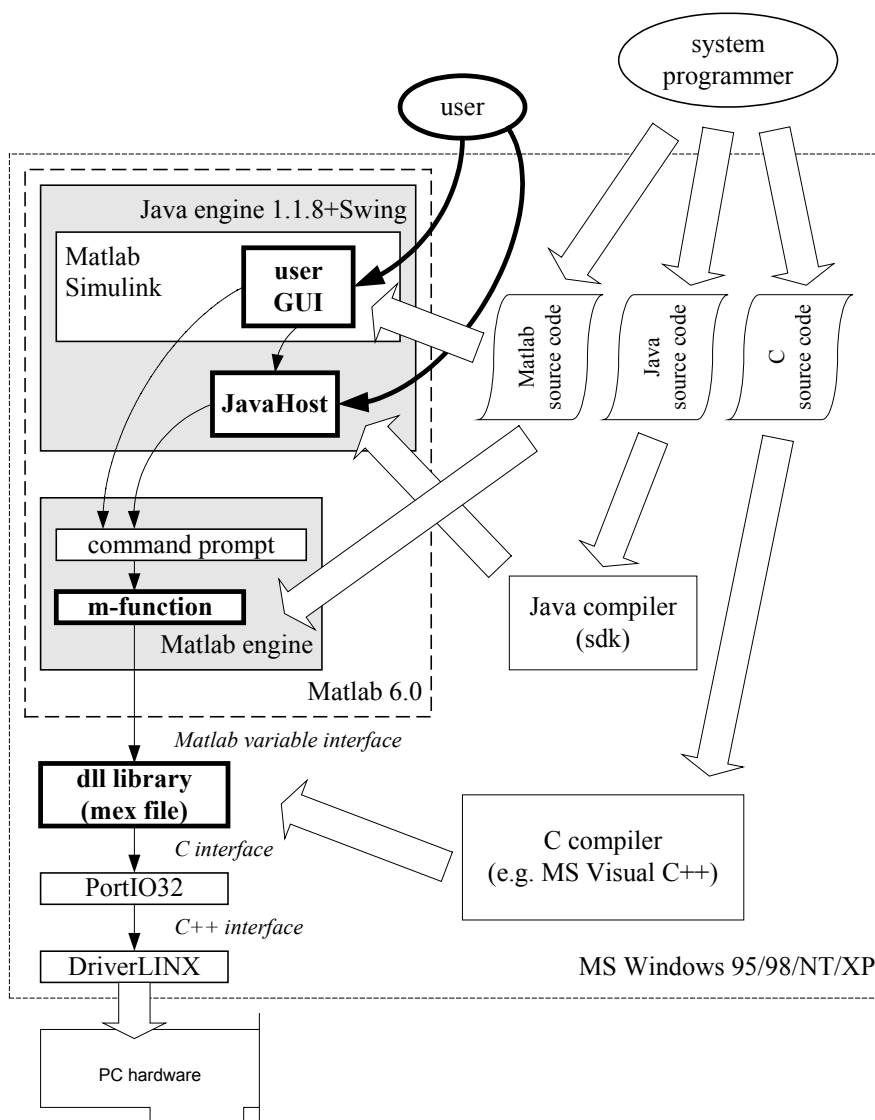
V několika posledních verzích Matlabu je integrován interpreter byte-kódu, který umožňuje spuštění zkompilovaný programů vytvořených v jazyce Java. Součástí Matlabu R12 je JDK verze 1.1.8. i s grafickou knihovnou Swing, která je běžně k dispozici až od verze 1.2. Matlab sám je z velké části v Javě implementován. Nic tedy nebrání tomu implementovat v Javě i uživatelské programy a využít tak všech výhod Javy včetně spuštění procesů běžících na pozadí, komunikace po Internetu, bitových operací apod.

V principu by nebylo nutné volat řadu funkcí Matlabu z příkazové řádky, ale i přímo jako javovskou funkci. Problémem je v tomto případě dokumentace k javovským ekvivalentům funkcí Matlabu. Když jsem před několika měsíci začínal s prvními experimenty, byl na toto téma na stránkách firmy Mathworks jen jeden příklad.

Dalším problémem je pak obsluha sériového a paralelního portu z prostředí Matlabu. Standartně jsou dodávány pouze funkce pro komunikaci po sériové lince. Paralelní rozhraní bylo vybráno pro přenos dat z cílového počítače kvůli jeho vyšší přenosové rychlosti.

## 5. Komunikace mezi Matlabem, uživatelským programem a hardwarem počítače

Celková struktura programu zvolená při realizaci monitorovacího softwaru je patrná z Obr. 3. O přímou komunikaci s hardwarem se starají systémové ovladače. Tím je umožněna funkce systému i pod novějšími operačními systémy. Matlab neumožňuje spouštět funkce z libovolné dynamické knihovny, proto bylo nutné vytvořit vlastní mex-soubory disponující vhodným rozhraním. Ty je pak možné volat jako běžnou m-funkci.



Obr. 3. Implementace monitorovacího programu v prostředí Matlabu

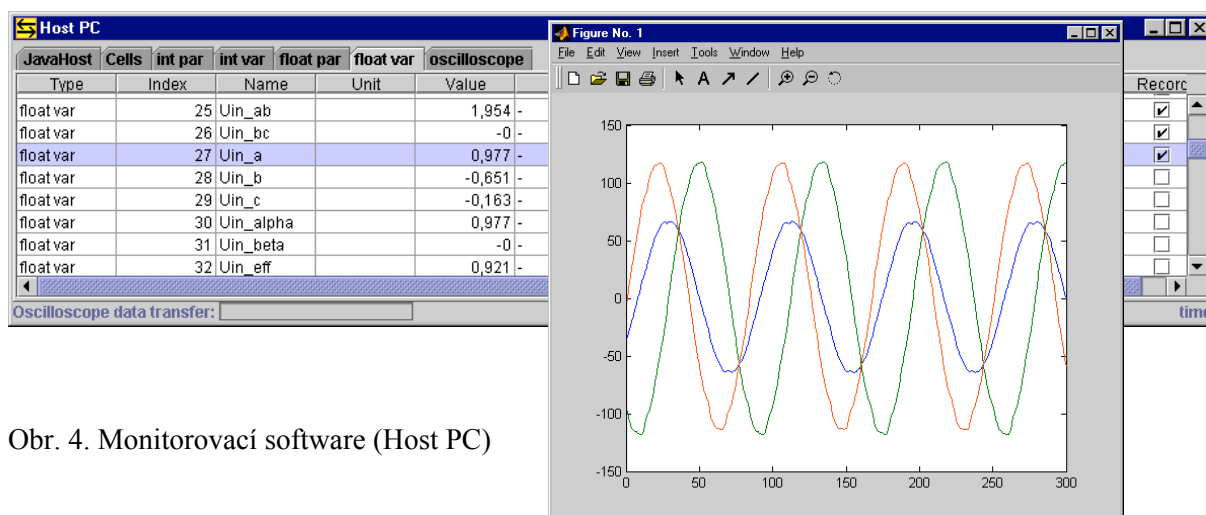
Grafické prostředí je vytvořené z větší části pomocí komponentů z grafické knihovny Swing. Separátně běžící procesy se starají o komunikaci s Target PC na základě požadavků uživatele. Matlabu je z každého javovského kódu možné předat příkazy k vykonání. Pro přenos hodnot proměnných opačným směrem jsou použity statické javovské proměnné.

Ač je monitorovací software v Host PC sadou vzájemně komunikujících programů napsaných v jazycích Java, C, jeví se z pohledu uživatele Matlabu jako jedna aplikace (podrobněji v následující kapitole).

## 6. Zrealizované uživatelské rozhraní

Implementovaný monitorovací software se spouští z příkazové řádky zadáním příkazu `hostPC`. Poté se již objeví hlavní okno aplikace. Jakmile jsou vybrány příslušné porty sériového a paralelního rozhraní pro komunikaci s cílovým počítačem, dojde ke spuštění podprogramů starajících se o přenos a vyhodnocení zpráv. Po jejich startu se nejprve provede identifikace cílového počítače. Pro datovou výměnu mezi Host PC a Target PC jsou podporovány 3 datové formáty: `integer` (32 bit), `float` (32 bit) a `ASCII string` (proměnlivá délka). Jakmile je z cílového počítače zjištěn počet proměnných a jejich jména dojde k inicializaci tabulek v monitorovacím programu podobně jak je tomu na obr. 4.

Po inicializaci jsou do hostícího počítače v pravidelných intervalech přenášeny již jen aktuální hodnoty proměnných. Ty mohou být z monitorovacího programu samozřejmě také měněny. Tento způsob by nedostačoval pro zpracování a záznam rychlých dějů. K tomu slouží funkce osciloskopu. Průběhy proměnných vybraných uživatelem jsou nejprve uloženy do operační paměti cílového počítače a teprve průběžně přenášeny do monitorovacího programu, který se postará o jejich uložení do prostředí Matlabu a jejich zobrazení. Díky této formě implementace mohou být data následně libovolným nástrojem Matlabu okamžitě analyzována a dále zpracována.



Obr. 4. Monitorovací software (Host PC)

## 7. Zhodnocení systému a podněty pro další vývoj

Popsaný systém prošel svou první fází vývoje. Byly implementovány základní funkce nutné pro jeho praktické používání. Systém byl testován na třech různých stanovištích, kde byla úspěšně ověřena jeho schopnost práce v reálném čase s periodou pod  $100\mu\text{s}$  bez přídavného signálového procesoru a správnost funkce monitorovacího softwaru. Slabinou je v současné verzi zvolený způsob komunikace mezi javovským jádrem programu a Matlabem, který znemožňuje dosáhnout maximální přenosovou rychlost danou fyzikálními vlastnostmi paralelního rozhraní v ECP módu. Přenosová rychlost je navíc také závislá na verzi JDK pod kterou Matlab běží. Presentované řešení nepředstavuje konečnou implementaci popisovaného systému. Ostatně jedním z hlavních cílů tohoto článku bylo přispět do diskuse na téma javovských programů a Matlabu.

### Kontakt

České vysoké učení technické v Praze  
Fakulta elektrotechnická  
Katedra elektrických pohonů a trakce (13114)  
Stanislav Flígl  
[xfligl@centrum.cz](mailto:xfligl@centrum.cz)