Simulační manažer pro OrCAD PSpice

Milan JAROŠ [1], Jaroslav KADLEC [2], Dalibor BIOLEK [2, 3]

Abstract. The paper describes a conception of so-called simulation manager, which considerably extends the application range of the simulation program OrCAD PSpice. It enables the operation of this program in the so-called sequential mode when the relatively independent tasks are run subsequently with a possibility of data exchange. A powerful programming language also enables the iterative runs within the conditional loops, which can be utilized e.g. for optimization.

1. Úvod

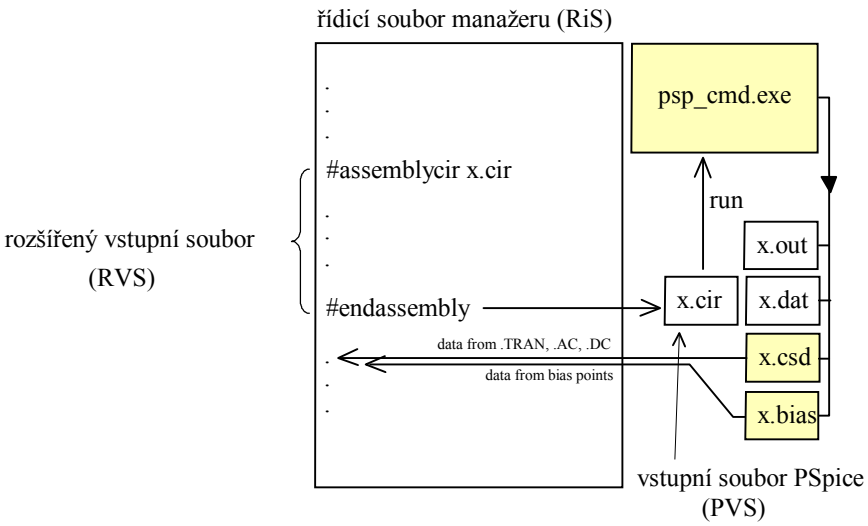
Programy třídy Spice, resp. PSpice jsou široce využívány k řešení nejrůznějších elektrotechnických problémů, a to jak na akademických pracovištích, tak i v průmyslu. Jedním z nejrozšířenějších programů tohoto typu je dnes OrCAD PSpice. Na rozdíl od programů WinSpice, ISSpice4 a dalších však nepodporuje využívání jazyka ICL (Interactive Command Language) pro řízení simulačních úloh. Tento jazyk přitom představuje účinný nástroj pro práci v tzv. sekvenčním módu, což je automatické spouštění posloupnosti simulačních úloh s možností ovlivňování charakteru následujících činností v závislosti na dosaženém stavu simulační úlohy. Uživatelé programu OrCAD PSpice tak nemohou principiálně řešit například úlohy tohoto typu: Postupné automatizované spouštění různých typů analýz, např. AC, Transient, DC bezprostředně po skončení předchozí analýzy s předáním výsledku dané analýzy jako vstupních dat, která by ovlivnila následující analýzu. Průběžné změny parametrů modelu obvodu v závislosti na výsledcích předchozích analýz. Opakovaná spouštění různých simulačních úloh v cyklech do té doby, než je dosaženo optimálního chování modelu obvodu.

V článku je popsán tzv. simulační manažer (SiM), který je navržen tak, aby řídil běh simulátoru OrCAD PSpice v souladu se záměry uživatele. Algoritmus řízení je definován tzv. řídicím souborem (RiS) simulačního manažeru. RiS je napsán v souladu se syntaktickými pravidly, definovanými speciálním programovacím jazykem manažeru. Tento jazyk obsahuje mj. instrukce pro vytváření rozšířeného vstupního souboru (RVS), který je zdrojovým textem pro generování klasického vstupního souboru PSpice (PVS), příkazy pro definici proměnných, pro definování základních analýz PSpice, které má program provést, pro řízení běhu PSpice a pro získávání výsledků simulací, jejich ukládání do proměnných a jejich matematické zpracování.

[1] Milan Jaroš, Mgr. Bc.,
iSEC - IT Services and Enterprise Communications s.r.o., Bidláky 20, 639 00 Brno, Czech Republic
email: milan.jaros@siemens.com
[2] Jaroslav Kadlec, Ing. Ph.D.,
UMEL FEKT VUT Brno, Údolní 53, 602 00 Brno, Czech Republic
tel.: +420 541146120, fax: +420 541146298, e-mail: kadlecja@feec.vutbr.cz
[3] Dalibor Biolek, Prof. Ing. CSc.
K217 UO Brno, Kounicova 65, 612 00 Brno, Czech Republic
tel.: +420 973442487, fax: +420 973443773, e-mail: dalibor.biolek@unob.cz

2. Koncepce simulačního manažeru

Koncepce spolupráce simulačního manažeru a výpočetního jádra PSpice je zjednodušeně znázorněna na obr. 1.



Obr. 1. Schéma komunikace mezi manažerem a simulačním programem prostřednictvím řídicího souboru (RiS) manažeru.

Příkazy v RiS jsou vykonávány postupně v pořadí, v jakém jsou v řídicím souboru zapsány. Dvojice příkazů #assemblycir a #endassembly ohraničuje rozšířený vstupní soubor (RVS), sloužící ke generování klasického vstupního souboru PSpice (PVS) pro spuštění relativně samostatné simulační úlohy. V tomto zdrojovém textu mohou být kombinovány jak klasické zápisy ze syntaxe PSpice, tak i rozšířené příkazy simulačního manažeru. V okamžiku zpracování příkazu #endassembly je automaticky vygenerován klasický PVS a následně je spuštěno výpočetní jádro psp_cmd.exe s vytvořeným PVS jako parametrem. SiM čeká na dokončení simulace. Na základě návratového kódu zjistí, zda simulace proběhla korektně. V případě chyby je činnost SiMu ukončena. Uživatel má možnost chybu identifikovat z vygenerovaného výstupního souboru.

V případě korektního ukončení simulační úlohy jsou k dispozici jak výstupní soubor, tak – v závislosti na charakteru simulační úlohy – další soubory s výsledky jednotlivých analýz typu Transient, AC nebo DC, jakož i vypočtené hodnoty stejnosměrných pracovních bodů. Tyto výsledky je možné načíst jako nové proměnné SiMu (např. definujeme proměnnou *VP*, do níž uložíme velikost napětí uzlu *p* v čase 10ms) a tyto proměnné použijeme při definování následně spouštěné simulační úlohy.

3. Charakteristika simulačního manažeru a jeho jazyka

Vstupem pro SiM je RiS, který obsahuje jednak modely simulovaných obvodů v jazyce PSpice, jednak speciální příkazy pro SiM, na jejichž základě budou řízeny simulační úlohy. Příkazy SiMu musí být jednoznačně odlišitelné od příkazů jazyka Spice. RiS musí mít jasnou a přehlednou strukturu.

RiS je tedy zápisem programu pro SiM. Bylo tedy nutné navrhnout nový, jednoduchý programovací jazyk, který by umožňovat výše naznačené řízení simulačních úloh. Požadavky na tento jazyk byly definovány takto:

• SiM bude číst RiS postupně od prvního řádku (jde tedy o sekvenční zpracování RiSu).

- Programovací jazyk SiMu by měl podporovat jednoduché matematické výpočty. Program tedy musí umět pracovat s proměnnými, reprezentujícími reálná čísla, a vyhodnocovat aritmetické výrazy. Ty mohou obsahovat proměnné, číselné konstanty, základní operátory (+, -, *, /, ^), závorky a některé matematické funkce. Musí tedy existovat příkazy, umožňující definování/deklaraci proměnných a vyhodnocování aritmetických výrazů.
- V RiSu se mohou vyskytovat RVS obvodů, které budou simulovány. Musí tedy existovat příkazy, které označují začátek a konec takového vstupního souboru. Příkaz, označující začátek RVS, musí mít parametr, udávající jméno souboru. Do tohoto souboru bude daný PVS generován. Příkaz, označující konec RVS, způsobí spuštění simulačního programu Spice, kde jako parametr tohoto programu bude předáno jméno souboru s právě vygenerovaným PVS. SiM v tomto okamžiku počká na dokončení simulace externím programem a poté bude pokračovat na dalším řádku RiSu.
- PVS může být simulačním manažerem před jeho vygenerováním modifikován. Jednou z možností je zápis hodnoty aritmetického výrazu na určité místo ve PVS. Musí existovat možnost zapsat do textu RVS příkaz pro vyhodnocení aritmetického výrazu. V okamžiku, kdy je daný PVS generován, je tento aritmetický výraz simulačním manažerem vyhodnocen a do výsledného souboru je zapsána jeho hodnota. Tímto způsobem je možné např. měnit parametr některé součástky za účelem jeho optimalizace.
- SiM musí být schopen zpracovat výsledky provedených simulací. Je tedy třeba, aby existovaly příkazy, které umožní získat tyto výsledky. Výsledky některých simulací je možné uložit programem PSpice do textových souborů. SiM proto musí být schopen tyto textové soubory zpracovat. Příkazy, uvedené v RiSu, pak umožní uložení takto získané hodnoty do proměnné, což umožní další práci s touto hodnotou.
- Po skončení činnosti SiMu musí být k dispozici všechny vygenerované soubory ze všech provedených simulačních úloh. Jde jednak o generované PVS a k nim příslušné výstupní .out soubory. Dále pak o soubory, obsahující výsledky provedených simulací, které vznikly díky použití příkazů .SAVEBIAS nebo .PROBE. Uchování těchto souborů může SiM zajistit tím, že je po provedení simulační úlohy bude zálohovat pod změněným jménem. Nová jména souborů budou odvozena od jmen původních s tím, že budou obsahovat číslo, odpovídající pořadí proběhnuvší simulace.
- SiM by měl obsahovat příkazy, umožňující větvení a cykly (příkazy *if* a *while*, známé z jiných programovacích jazyků) na základě vyhodnocení logických výrazů. Pomocí logických výrazů by mělo být možné porovnávat hodnoty aritmetických výrazů a samozřejmě by měla existovat možnost spojovat logické výrazy do složitějších podmínek – pomocí operátorů logického součtu, součinu a negace.

Příkazy větvení a cyklů budou moci být zapsány také v části, generující PVS. Tímto způsobem bude možné řídit, které části PVS mají být generovány, případně některé jeho části zopakovat.

K příkazům pro větvení a cykly lze zařadit také příkaz skoku (příkaz *goto*), který slouží k přenosu řízení na zadané návěští (definované příkazem *label*). Celá tato skupina příkazů se nazývá příkazy pro řízení běhu. Díky těmto příkazům bude možné algoritmizovat vyhodnocování výsledků předchozích simulací a řídit spouštění simulací dalších.

- Měl by existovat příkaz, umožňující vložení souboru (podobně jako PSpice příkaz *.INC*). Vkládaný soubor by mohl také obsahovat příkazy SiMu (a byl by tedy sám o sobě řídicím programem – proto nelze k tomuto účelu použít PSpice příkaz .INC). Tímto způsobem by bylo možné rozdělit řídicí program pro SiM do více souborů a zpřehlednit tak jeho zápis. Rovněž je tímto způsobem umožněno znovupoužití již vytvořených částí programu.
- Dalším přirozeným požadavkem na jazyk SiM je možnost zápisu komentářů. Zde se přímo nabízí ponechat komentářům stejný formát jako v jazyce PSpice, tj. pokud řádek začíná znakem *, jde o komentář. Jsou-li komentáře zapsány v RVS, který generuje PVS, budou také zapsány do generovaného PVS.

SiM pracuje jako interpret výše popsaného jazyka. Činnost SiMu lze rozdělit do dvou fází. Nejprve je provedena syntaktická analýza RiSu. V této fázi lze nalézt syntaktické chyby, tj. nesprávný zápis příkazů SiMu nebo jejich chybné umístění. Reakcí na nalezenou chybu je ukončení zpracování RiSu a zobrazení chybového hlášení.

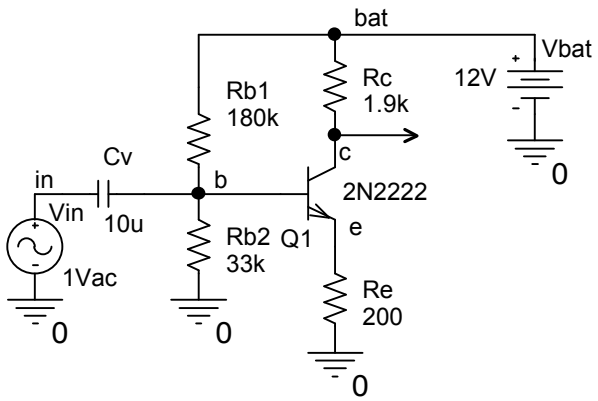
V průběhu syntaktické analýzy je text RiSu rozložen na jednotlivé syntaktické elementy (např. příkaz, komentář, řádek generovaného PVS atd.). Syntaktické elementy jsou reprezentovány datovou strukturou (třídou). V průběhu syntaktické analýzy se vytvoří seznam těchto elementů, resp. příkazů. Druhá fáze činnosti SiMu spočívá ve vykonání příkazů v tomto seznamu. Činnost SiMu je ukončena v případě, že je dosaženo konce seznamu, nebo je proveden příkaz, sloužící k zastavení SiMu, nebo dojde k chybě (např. dělení nulou apod.).

V průběhu zpracování RiSu není simulačním manažerem kontrolována syntaktická správnost generovaných vstupních souborů PSpice. Tuto kontrolu provede až samotný program PSpice v okamžiku, kdy mu bude předán vygenerovaný soubor. Zároveň s tím provede samotnou simulaci.

Výsledek syntaktické kontroly, případně výskyt chyby jiného druhu, lze zjistit z návratového kódu programu psp_cmd.exe. Je-li návratový kód nulový, simulace proběhla bez chyby. Je-li nenulový, vyskytla se chyba. Popis chyby je uveden ve výstupním .out souboru. V případě, že dojde k takovéto chybě při simulaci, je činnost SiMu ukončena.

4. Ukázka práce se simulačním manažerem

Na obr. 2 je schéma tranzistorového zesilovače se stabilizací pracovního bodu zápornou zpětnou vazbou přes emitorový odpor.



Obr. 2. Příklad simulovaného obvodu. Úkolem je navrhnout Rc tak, aby obvod vykazoval zesílení 10 na kmitočtu 1kHz.

Analýza programem PSpice ukazuje, že zesílení obvodu v pásmu středních kmitočtů je asi 9. Úkolem je navrhnout velikost kolektorového odporu Rc tak, aby na kmitočtu 1kHz bylo zesílení 10.

Kompletní obsah řídicího souboru pro SiM je uveden níže. Jednotlivé řádky jsou pro přehlednost očíslovány (netvoří součást souboru).

```
1:  *transistor amplifier
2:  #defsim AC .AC dec 1 1k 10k
3:  #set Rc 1.9k gain 1
4:  *
5:  #while (gain)<=10
6:  #assemblycir run.cir
7:  *
8:  Vbat bat 0 12V
9:  Q c b e Q2N2222
10: Rc bat c #$Rc$
11: Re e 0 200
12: Rb1 bat b 180k
13: Rb2 b 0 33k
14: Cv in b 10u
15: Vin in 0 AC 1
16: .lib
17: #runsim AC
18: .print AC v([c])
19: #endassembly
20: #setprobe AC V([c])
21: #getprobe gain AC V(c) 1k
22: #set Rc Rc+20
23: #endwhile
```

První řádek je klasické záhlaví vstupního souboru dle běžných konvencí PSpice. Na řádcích 2-6 jsou příkazy SiMu (začínají znakem #). Nejprve je na řádku 2 definován typ simulace jménem *AC*. Jde o základní PSpice analýzu AC, která je na řádku definována dle běžných syntaktických pravidel PSpice. Zde se konkrétně jedná o jednobodovou AC analýzu na kmitočtu 1kHz. Na řádku 3 jsou příkazem #set deklarovány proměnné *Rc* a *gain* a jsou jim přiřazeny číselné hodnoty. Jde o obdobu příkazu .param v PSpice. Na řádku 5 začíná cyklus *while* s ukončením na řádku 23. Jeho úkolem je testovat, zda proměnná *gain* je menší nebo rovna hodnotě 10. Pokud ano, provedou se instrukce v tělu cyklu. V opačném případě činnost SiMu končí, protože řádek 23 je posledním řádkem RiS.

Tělo cyklu začíná příkazem #assemblycir na řádku 6 s párovým příkazem #endassembly na řádku 19. V rámci těchto příkazů je definován RVS pro generování PVS. Na řádcích 8 až 15 je zapsán klasický PSpice netlist zesilovače z obr. 2 s výjimkou na řádku 10, kde je k definici odporu Rc použit vzorec #Rc. Symbol # znamená, že interpretaci výrazu provede SiM. Následují párové znaky $ $, mezi nimiž je vlastní vzorec, jehož číselný výsledek vloží SiM do generovaného PVS. V tomto případě vzorec obsahuje pouze proměnnou *Rc*.

Na řádku 17 je příkaz pro provedení analýzy, definované na řádku 2. Rozdíl oproti použití klasického PSpice příkazu .AC je v tom, že výsledky analýzy se automaticky

ukládají do datového souboru pro postprocesor PROBE, ovšem v textovém formátu CSDF, aby je byl SiM schopen číst. Detailní nastavení parametrů příkazu .PROBE lze nastavit příkazem #setprobe na řádku 20. Zde je konkrétně nastaveno, že do datového souboru budou ukládány pouze hodnoty napětí uzlu c. Na řádku 18 je ještě doplněn klasický PSpice příkaz .PRINT, kterým uložíme do výstupního souboru aktuální hodnotu střídavého napětí uzlu c, což je číselně rovno hodnotě střídavého zesílení.

Příkaz #getprobe na řádku 21 zabezpečí uložení velikosti napětí uzlu *c* na kmitočtu 1kHz do proměnné *gain*. Následně je příkazem #set v řádku 22 zvýšena hodnota *Rc* o 20 Ohmů. Generování PVS run.cir a jeho analýza se pak opakují v cyklu *while* tak dlouho, dokud není dosaženo zesílení většího než 10.

V tomto konkrétním případě proběhne analýza celkem 11 krát. V posledních dvou výstupních souborech nalezneme tyto dvojice hodnot proměnných *Rc* a *V(c)*:

2100 Ohmů, 9.997

2120 Ohmů, 10.09

Optimální hodnota *Rc* tedy leží v intervalu (2.1 - 2.12) kOhmů.

5. Závěry

Simulační manažer (SiM), popsaný v tomto článku, je samostatný spustitelný program, který umožňuje s využitím tzv. řídicího souboru (RiS) řídit činnost programu OrCAD PSpice. V současné době je dokončován vývoj konzolové aplikace SiMu, který umožňuje práci na úrovni textových souborů. Současně pracujeme na vývoji grafického prostředí, které umožní pohodlné programování sekvenčních operací i pro uživatele, kteří detailně neovládají skriptovací jazyk manažeru.

Poděkování

Výzkum, jehož výsledky jsou prezentovány v tomto článku, je podporován Grantovou agenturou ČR prostřednictvím grantu č. 102/08/0784, Výzkumnými záměry VUT Brno č. MSM0021630503 a MSM0021630513 a Výzkumným záměrem UO Brno č. MO FVT0000403.

Literatura

[1] VLADIMIRESCU, A. The SPICE book. John Willey&Sons, Inc., 1994.
[2] LÁNÍČEK, R. Simulační programy pro elektroniku. BEN – technická literatura, 2000.
[3] PSpice Reference Guide. Cadence Design Systems, Inc. Online on http://www.classes.usc.edu/engr/ee-ep/326/pspcref.pdf
[4] JAROŠ, M. Simulační manažer pro SPICE-kompatibilní programy. Bakalářská práce, UMEL FEKT VUT Brno, 2007.