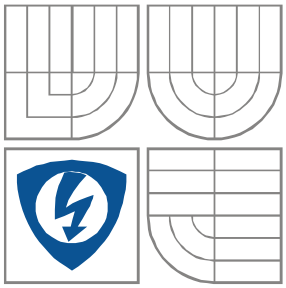
FAKULTA VOJENSKÝCH TECHNOLOGIÍ
Vojenská akademie v Brně



Navrhování elektronických obvodů počítačem

Počítačové modelování, analýza a simulace

Prof. Ing. Dalibor Biolek, CSc.

Brno **únor 2004**

OBSAH

1 Úvod

Tento učební text je věnován počítačové analýze a simulaci analogových elektronických obvodů. Zde se seznámíte s abecedou, jejíž zvládnutí vám otevře cestu k efektivní práci s jakýmkoliv současným simulačním programem z rodiny „SPICE“. Jestliže se Vám podaří následující texty dočíst do konce a čtení budete prokládat pokusy na svém počítači, ověřovat popisované experimenty na demonstračních souborech a přemýšlet o výsledcích, které vám budou poskytovat programy SNAP a MicroCap, pak učiníte mnoho pro to, aby vaše „cesta na vrchol“ byla co nejpoctivější. Nejen proto, že platí rovnice: MicroCap = SPICE + „něco navíc“. Do problematiky počítačové simulace budete zasvěcováni postupně, přes relativně jednodušší programy založené na symbolických algoritmech (SNAP) až po „numerické“ simulátory (MicroCap). Program SNAP jsme vytvořili právě pro studenty, kteří potřebují rychle zvládnout první krůčky v počítačové simulaci obvodů. K zvládání výše zmíněné „abecedy“ by vám měla hodně napomoci netradičně pojatá kapitola 5, v níž jsou koncentrovány 3 na sebe navazující lekce počítačové simulace v programu SNAP. K umocnění celkového „výukového“ efektu je tato část formálně upravena poněkud odlišně od ostatních kapitol (převaha grafické informace nad textovou, vynechání číslování obrázků atd.). Nechybí zde ani náměty na samostatné experimentování v SNAPu. Máte možnost pokračovat v řešení sady 123 elektronických příkladů, které si můžete stáhnout na váš počítač z [?]. Jejich popis je uveden v příloze P1. Pokrývají širokou škálu problémů a jsou setříděny do kategorií „Základy elektrotechniky“, „Zesilovače“, „Oscilátory“, „Syntetické prvky“, „Filtry“, „Vysokofrekvenční obvody“. Tuto úvodní část však doporučuji i zkušeným uživatelům komerčních simulátorů. SNAP totiž poskytuje – na rozdíl od simulátorů rodiny SPICE – výsledky v tzv. symbolické formě, což vám umožňuje získávat některá řešení, která prostě komerční simulátory neumí.

Navazující kapitoly o „numerických“ simulačních programech je sice možno studovat nezávisle na předchozích částech, efektivnější je ovšem nejprve projít předchozími „základy“. Uvedený „skok“ je možné doporučit snad jen zkušenějším uživatelům některého z komerčních simulátorů. Naučíte se „pohybovat“ v profesionálním schématickém editoru, seznámíte se základními pojmy a pravidly, jak pracovat s modely součástek, s knihovnami, jak správně používat příkazy typu .MODEL a .DEFINE atd. Naučíte se pracovat s nejrůznějšími typy analýz, v různých analyzačních módech a režimech. V závislosti na vašich potřebách můžete zvládnout tuto problematiku do hloubky, která vám bude vyhovovat, neboť kromě základních typů analýz, které mají v podstatě stejný charakter v různých simulačních programech, jsou zde popisovány do podrobností i další užitečné praktiky (například různé „interaktivní“ analýzy, citlivostní analýza, animace apod.). Budete mít možnost na konkrétních příkladech pochopit postupy, které jsou sice v „nabídkách“ různých simulačních programů, avšak pro jejich běžné uživatele, kteří nemají čas studovat rozsáhlé manuály, jsou většinou „obestřeny tajemstvím“. Řeč je o spektrální analýze časových průběhů, o zásadách používání algoritmů dopředné a zpětné rychlé Fourierovy transformace v simulačních programech, o nastavování počátečních podmínek v různých typech analýz, o povolování či zakazování výpočtu stejnosměrného pracovního bodu, o správném používání příkazů .IC a .NODESET, o šumové analýze, o zobecněné stejnosměrné analýze, jak se vyznat v „simulačních teplotách“, v různých metodách krokování parametrů, jak prakticky používat metodu Monte Carlo, jak správně pracovat v optimalizačním režimu, jak konkrétně překonávat různé problémy při počítačové simulaci včetně problémy s konvergencí vnitřních

algoritmů atd. Tyto „koncentrované zkušenosti“ by mohly knihu učinit zajímavou i pro již pokročilejší uživatele některého z profesionálních simulačních programů.

Jak studovat tyto texty, aby vám přinesly co největší užitek?“ Individuálně, v závislosti na tom, „jak daleko jste od vrcholu“, s vědomím, že „cesta není přímočará“. Text obsahuje řadu křížových odkazů jak na předchozí kapitoly, tak i na přílohovou část. Přílohy vám mohou posloužit jako stručné přehledové „manuály“, zejména co se týče konkrétních způsobů práce s modely součástek.

2 Programy pro simulaci analogových obvodů

Cíl kapitoly:

- Podat stručný přehled o současných počítačových programech pro analýzu a simulaci elektronických obvodů.

Simulační programy, využívající výkonnost soudobých počítačů, otevírají dříve nevídané možnosti pro analýzu a simulaci dějů v složitých elektronických obvodech. Výsledkem historického vývoje, který započal zhruba v padesátých letech minulého století, jsou programy určené pro simulaci analogových obvodů („analogové simulátory“) a pro simulaci digitálních obvodů na logické úrovni („digitální simulátory“). Celosvětově rozšířeným standardem analogové simulace je program SPICE [3.2, 3.3, 6.3] a jeho komerční verze, zatímco u digitální simulace existuje několik celosvětově používaných programů. Simulátory s přívlastkem „Mixed-Mode“ mají schopnost simulovat obvody na analogové i logické úrovni.

Programů pro analogovou, resp. smíšenou simulaci obvodů, kterých se týká tato kniha, je dnes celá řada. Kromě simulátorů z rodiny SPICE jsou velmi zajímavé programy MicroCap [6.1], Tina [6.4], Electronics Workbench, resp. Multisim [6.2], a další. U nás vznikl zajímavý program CIA [3.7], který využívá modelů SPICE, jeho numerické algoritmy jsou však odlišné. Z programů vytvořených pro pracovní stanice jmenujme SABER a ELDO. Některé z programů, určené pro klasické počítače PC, které jsou dostupné zdarma nebo za symbolický poplatek, jsou představeny v knize [3.1].

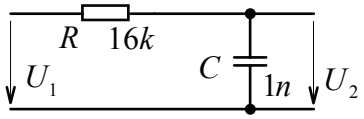
V poslední době se kromě výše zmíněných standardních programů, jejichž matematickým základem jsou numerické metody řešení rozsáhlých soustav rovnic, objevují programy, pracující na tzv. symbolickém, resp. semisymbolickém principu (podrobnější vysvětlení viz část 4.2). Tyto programy mají schopnost prezentovat výsledky analýzy ve formě matematických vzorců. Protože interní algoritmy si činí extrémní nároky na výkon hardware, nacházejí tyto programy uplatnění až v současné době. K rozšířeným programům tohoto typu patří Analog Insydes, nadstavba softwarového balíku Mathematica, nebo NODAL, nadstavba MAPLE. Symbolické výpočty umožňuje i program TINA, který jinak využívá numerické algoritmy. Programy CIA a Multisim poskytují jednu z forem semisymbolických výpočtů – nulové body a póly obvodových funkcí. Tyto údaje získávají na základě čistě numerických metod. Z programů „dostupných pro každého“, které jsou založeny na symbolických algoritmech, jmenujme programy SNAP [6.5], COCO, COFACTOR a LTP2 [6.6], vzniklé na VUT v Brně.

3 Symbolické, semisymbolické a numerické výsledky analýzy lineárních obvodů

Cíl kapitoly:

- Vysvětlit tvar, výhody a nevýhody různých forem výsledků, které poskytují počítačové programy pro analýzu a simulaci lineárních a linearizovaných obvodů.

Pojmy uvedené v nadpisu této kapitoly vysvětlíme na příkladu jednoduchého *RC* článku na **obr. 3.1**. Cílem analýzy bude vyšetření způsobu přenosu napětí U_1 ze vstupu článku na jeho výstup, přičemž výstupní sledovanou veličinou je napětí U_2.



Obr. 3.1: RC článek typu dolní propust 1. řádu.

Výpočet operátorové přenosové funkce vede na přenos napěťového děliče:

$$K_u = \frac{U_2}{U_1} = \frac{\dfrac{1}{pC}}{R + \dfrac{1}{pC}}.$$

Po úpravě získáme výsledek analýzy v jednom ze symbolických tvarů:

$$K_U = \frac{1}{1 + pRC}. \tag{3.1}$$

Termín symbolický označuje, že ve výsledku jsou přítomny *symboly parametrů prvků obvodu* spolu se *symbolem Laplaceova operátoru*, chcete-li komplexního kmitočtu *p*. Nevyskytují se zde skutečné, číselné hodnoty parametrů.

Níže uvádíme příklad textového výstupu programu SNAP při analýze výše uvedeného RC článku:

```
_____symbolic_____

1
---------------------------------------
1
+s*( R*C )
```

Čárkovaná linie představuje zlomkovou čáru. Ve jmenovateli je použit pro Laplaceův operátor symbol *s* namísto u nás zaužívaného *p*.

Symbolický výsledek je výhodný v tom, že přesně ukazuje, jaký je vliv každé součástky v obvodu na jeho přenosové vlastnosti. Dosazováním různých číselných hodnot za symbolické parametry pak můžeme snadno specifikovat chování obvodu při použití konkrétních součástek.

Výhody symbolického popisu se však rychle ztrácejí při analýze rozsáhlejších obvodů se složitými modely součástek. Například vzorec pro střídavé zesílení třístupňového tranzistorového zesilovače může obsahovat i několik stovek nebo i více členů podle složitosti

modelů tranzistoru. Některé programy proto nabízejí možnost generace zjednodušených symbolických vzorců. Zjednodušení je provedeno tak, že se zanedbávají členy, jejichž číselný příspěvek k celkové hodnotě výrazu je nevýznamný. Míra zjednodušení se obvykle dá regulovat zadáním přípustné chyby.

Aby bylo možné ze symbolického výrazu odvodit další informace o obvodu, které běžného uživatele zajímají nejvíce, například kmitočtovou charakteristiku nebo chování obvodu v čase, je třeba symboly ve vzorci nahradit příslušnými číselnými hodnotami. Učiníme-li tak pouze pro některé, ale ne všechny symboly, získáme tzv. semisymbolický výsledek: ve vzorci budou přítomna *jak čísla, tak i symboly*. Nejčastější forma semisymbolického výsledku vznikne dosazením číselných hodnot všude kromě komplexního kmitočtu *p*. Pro RC článek na **obr. 3.1** vychází

$$K_U = \frac{1}{1+16.10^{-6}\,p} = 62500\frac{1}{62500+p}\,. \tag{3.2}$$

Příklad výstupu programu SNAP je zde:

_____semisymbolic_____

Multip. Coefficient = 6.25000000000000E+0004

1.00000000000000E+0000

6.25000000000000E+0004
1.00000000000000E+0000 * s

Význam multiplikativního koeficientu je zřejmý z členu na pravé straně rovnice (3.2).

Takovýto semisymbolický výsledek je pak pro simulátor „vstupní branou“ do další analýzy, jejíž výsledky jsou v čistě numerické formě. V skriptech [1.23] je ukázáno, jak lze ze semisymbolického vzorce zjistit, zda je obvod stabilní, jak určit kmitočtovou charakteristiku nebo odezvu obvodu na jednoduché signály.

Numerickým výsledkem analýzy může být výpočet nulových bodů a pólů obvodové funkce, což jsou kořeny jejího čitatele a jmenovatele. Program SNAP poskytne toto řešení:

_____zeros_____

none

_____poles_____

-6.25000000000000E+0004

V čitateli přenosové funkce je pouze jednička, nikoliv polynom, z něhož by bylo možné vypočítat kořeny. Nulové body tedy neexistují. Ve jmenovateli je výraz, který – pokud se má rovnat nule – poskytne kořen -62500:

$$p + 62500 = 0\,.$$

Z teorie pak vyplývá, že analyzovaný obvod je stabilní, protože všechny jeho póly (zde je pouze jeden) mají záporné reálné části.

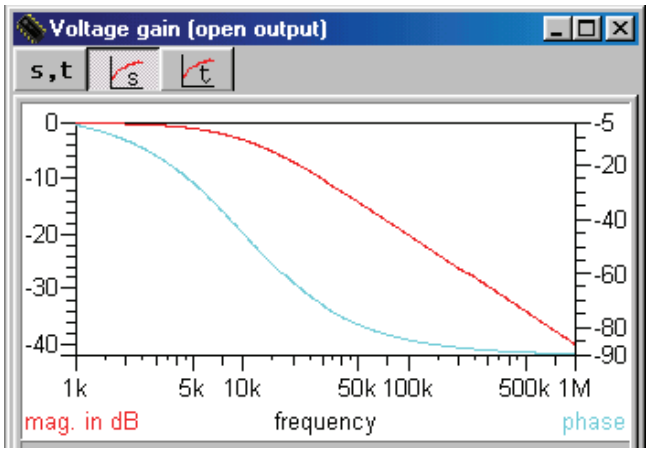
Požadujeme-li například zjistit, jaké jsou přenosové vlastnosti RC článku na kmitočtu 5 *k*Hz, dosadíme do semisymbolického vzorce pro přenosovou funkci komplexní kmitočet

$$p = j\omega = j.2.\pi.5000 = j10000\pi :$$

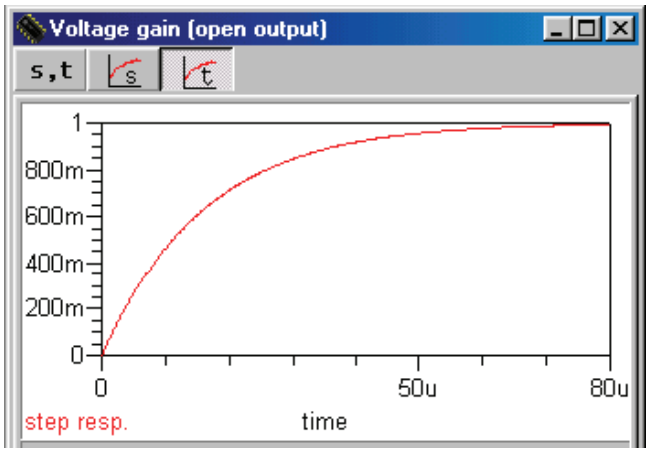
$$K_u(p = j10000\pi) = \frac{62500}{62500 + j10000\pi} \doteq 0{,}893\angle -26{,}7° .$$

Obdrželi jsme typický numerický výsledek analýzy, který říká, že harmonický signál o kmitočtu 5 *k*Hz bude v ustáleném stavu pronikat na výstup se zeslabením amplitudy na 89 % původní velikosti a bude na výstupu posunut - zpožděn o necelých 27 stupňů oproti vstupnímu signálu.

Donutíme-li program, aby tento numerický výpočet provedl pro mnoho různých kmitočtů ze zadaného intervalu, například od 1 *k*Hz do 1 *M*Hz, a vynesl zeslabení a fázový posuv do křivek, získáme kmitočtové charakteristiky (viz **obr. 3.2** a).



a)



b)

Obr. 3.2: a) amplitudová a fázová kmitočtová charakteristika, b) přechodová charakteristika *RC* článku z **obr. 3.1**. Výstupy programu SNAP.

V [1.23] je popsáno, jak je možné z přenosové funkce odvodit přechodovou charakteristiku, tj. odezvu na jednotkový skok. Program SNAP nejprve určí semisymbolický výsledek v tomto tvaru:

_____step response_____

1.00000000000000E+0000
-1.00000000000000E+0000*exp(-6.25000000000000E+0004*t)

který odpovídá vzorci

$$h(t) = 1 - 1.e^{-6,25.10^4 t}.$$

Potom program dosazuje konkrétní hodnoty času ze zadaného výpočetního intervalu a vykreslí získaný časový průběh (viz **obr. 3.2** b).

4 Symbolické, semisymbolické a numerické algoritmy analýzy lineárních obvodů

Cíl kapitoly:

- Vysvětlit používané způsoby analýzy elektronických obvodů v počítačových programech, jejich výhody, nevýhody, principiální omezení.

Z předchozího výkladu by se mohlo zdát, že numerické výsledky analýzy je možné získat jen prostřednictvím symbolických a semisymbolických výsledků. Naštěstí tomu tak není. Komerční simulační programy získávají numerické výsledky pomocí jiných algoritmů, které jsou založeny na numerickém řešení rozsáhlých soustav rovnic. Řešení takovýchto rovnic symbolickými algoritmy vysoce převyšuje možnosti současné nejvýkonnější výpočetní techniky.

Symbolické algoritmy jsou založeny na složitých matematických postupech. Z hlediska programátora symbolického programu to představuje manipulaci s proměnnými, jejichž počet roste exponenciálně se složitostí analyzovaného obvodu. Algoritmy jsou náročné na paměť a výpočty. Pokud jsou daným hardware vůbec realizovatelné, pak trvají mnohem déle než u numerických algoritmů. Na jednu stranu tedy symbolické algoritmy umožňují přístup ke všem formám výsledků, symbolickým, semisymbolickým i numerickým, avšak jejich použití je omezeno na třídu ne příliš rozsáhlých obvodů.

Oproti tomu numerické algoritmy poskytují pouze numerické výsledky, avšak bez tak drastického omezení na složitost analyzační úlohy.

Semisymbolické výsledky může simulátor získat dvěma různými způsoby. První způsob již známe – přímo ze symbolického výrazu dosazením číselných hodnot parametrů. Druhý způsob vede přes čistě numerické metody hledání koeficientů obvodových funkcí, nebo – což je častější – výpočtem nulových bodů a pólů obvodových funkcí. První způsob je sice jednoduchý, ale trpí všemi omezeními nadřazeného symbolického algoritmu. Druhý způsob je výhodnější zejména při analýze rozsáhlejších obvodů, neposkytuje však samozřejmě symbolické výsledky. Ty však mnohdy ani nejsou vyžadovány.

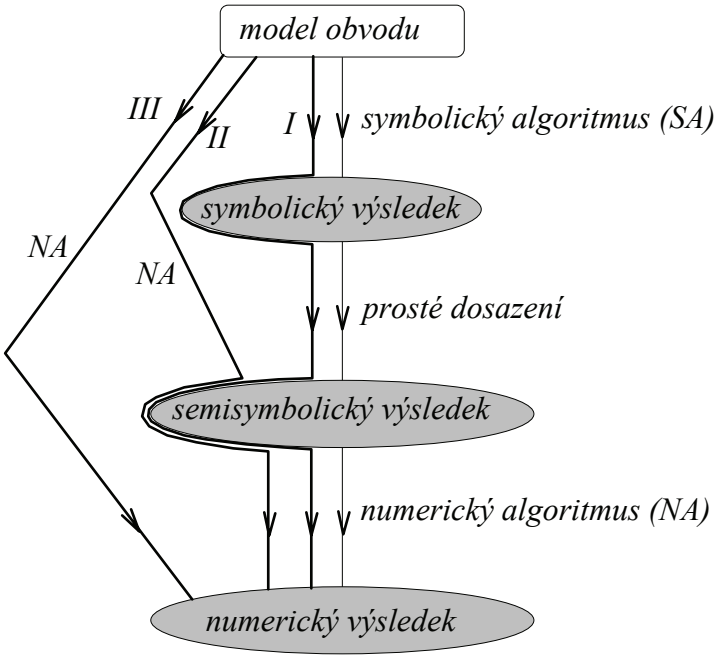
Právě vyložený vztah mezi formami výsledků analýzy a algoritmy analýzy je přehledně uveden na **obr. 4.1**.

Cesta s označením *I* je typická pro programy, jejichž jádro je založené na symbolických algoritmech. V této knize je bude zastupovat program SNAP.

Cesta označená symbolem *III* je charakteristická pro drtivou většinu komerčních simulátorů. Pro výklad typických činností spojených s používáním simulátorů tohoto typu jsme vybrali program MicroCap.

„Prostřední“ cesta *II* je využívána některými simulátory, které jinak pracují na numerickém principu, k realizaci výpočtů nulových bodů a pólů hledaných obvodových

funkcí. Jmenujme alespoň programy CIA a Multisim, z tuzemských programů SADYS (resp. DYNAST) . Těmito technikami se nebudeme blíže zabývat.



Obr. 4.1: Symbolické, semisymbolické a numerické výsledky analýzy a tři typické cesty k těmto výsledkům.

V následujících kapitolách se zaměříme na způsoby práce dvou skupin programů, jejichž činnost je charakterizována cestami *I* a *III* na **obr. 4.1**. Přes rozdílné pojetí analýzy mají programy obou skupin mnoho společného. Z metodického hlediska je výhodnější seznámit se nejen s těmito společnými bloky, ale vůbec s celou strukturou simulačního programu nejprve na jednodušších programech ze skupiny *I*.

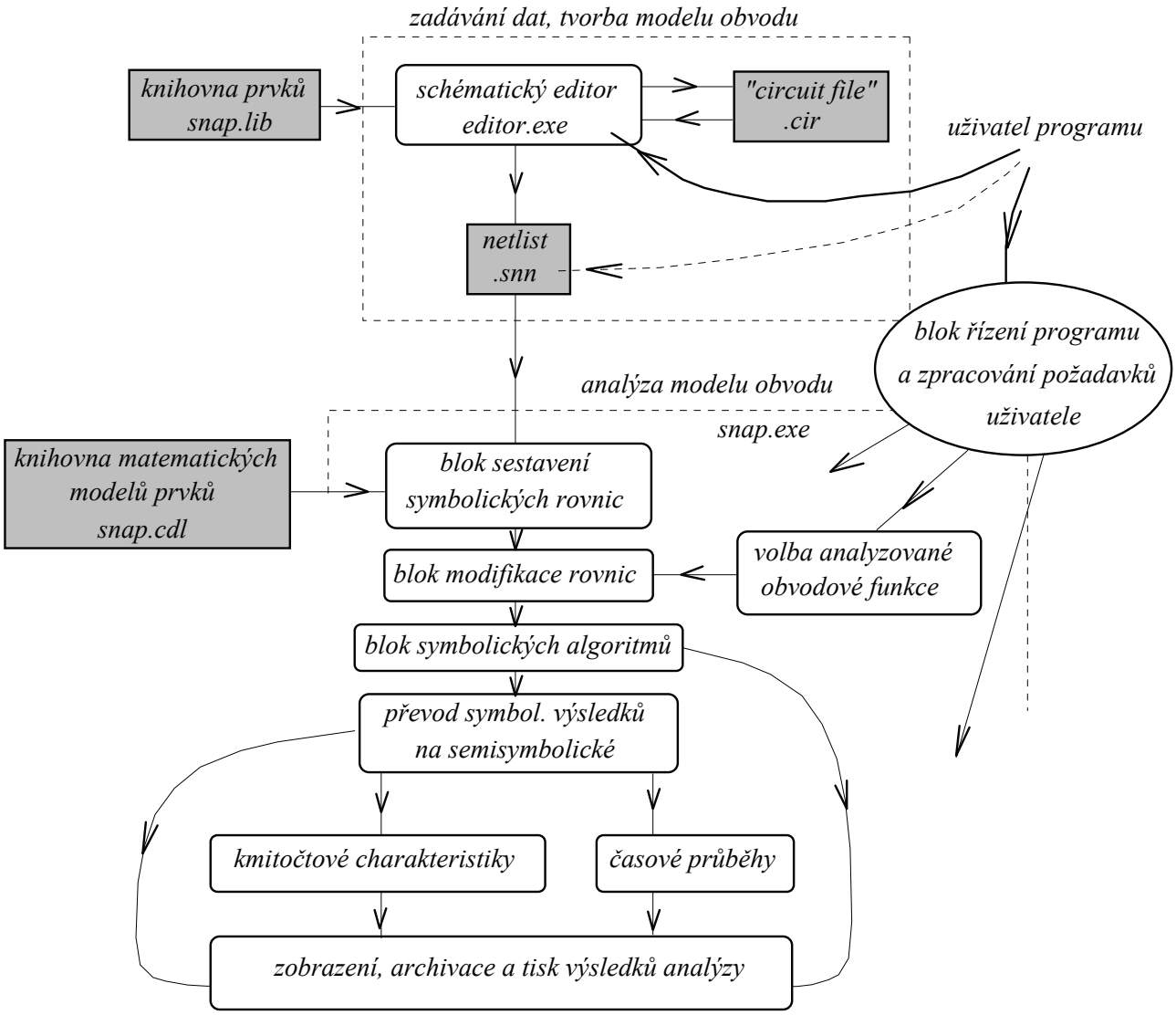
5 Struktura simulačního programu, založeného na symbolických algoritmech (SNAP)

Cíle kapitoly:

- Vysvětlit vnitřní strukturu programů typu SNAP. Pochopení této struktury čtenáři usnadní pozdější studium složitých komerčních simulačních programů.
- Objasnění pojmů *schématický editor*, *knihovna schématických značek*, *knihovma modelů*, *netlist*, *vstupní soubor*.

Na **obr. 5.1** je zjednodušená struktura programu SNAP v. 2.6, jehož výpočetní jádro je tvořeno symbolickými algoritmy. Vysvětlíme si principy fungování programů tohoto typu, i když daná struktura není pochopitelně jednotná pro všechny programy z dané kategorie a může se v jednotlivostech případ od případu lišit.

Schéma analyzovaného obvodu je nejprve vytvořeno v schématickém editoru a zde konvertováno do tzv. snap netlistu (soubor s příponou .snn). Netlist je textový soubor, obsahující všechny informace o typu součástek obvodu a způsobu jejich propojení. Poté je spuštěn vlastní program SNAP, který načte data z netlistu a provede příslušný druh analýzy.



Obr. 5.1: Struktura simulačního programu, založeného na symbolických výpočtech (SNAP).

K programu SNAP je standardně dodáván schématický editor **EDITOR.EXE**. Je však možné použít libovolný jiný editor, jehož výstupem je netlist ve standardním formátu SPICE. Program **EDITOR.EXE** k své činnosti využívá textový ascii soubor **SNAP.LIB**, v němž jsou speciálním jazykem definovány schématické značky elektrických prvků editoru a množina jejich zadávaných parametrů. Úpravou tohoto souboru lze rozšiřovat množinu používaných prvků a modifikovat již prvky nadefinované.

Vytvořené schéma je možné uložit do souboru se standardní příponou **.CIR**. Tento soubor se obvykle označuje anglickým termínem „circuit file“ neboli vstupní soubor. Obsahuje veškeré informace nutné k rekonstrukci elektrického schématu. Z tohoto souboru je tedy teoreticky možné jednoznačně odvodit netlist, opačný přechod však možný není. Na příkladu MicroCapu později ukážeme, že prakticky všechny celosvětově rozšířené simulátory umísťují do vstupního souboru další informace, zejména údaje o požadované analýze, o stavu programu v době jeho posledního používání apod.

Z **obr. 5.1** vyplývá, že k analýze obvodů teoreticky schématický editor nepotřebujeme, pokud ovšem dokážeme vytvořit textový soubor s netlistem v požadovaném formátu. Tento způsob zadávání obvodu se ale často nepoužívá.

Vlastní analyzační program **SNAP.EXE** využívá pomocného textového ascii souboru **SNAP.CDL**. V tomto souboru jsou informace o matematických modelech prvků. Filozofie modelování je založena na modifikované metodě uzlových napětí, konkrétně na metodě razítek. V souboru **SNAP.CDL** je pomocí jednoduchého jazyka ke každému prvku definována příslušná matice – razítko. Informace v souboru **SNAP.CDL** souvisejí se způsobem definice prvků v souboru **SNAP.LIB**. Uživatel, ovládající metodu razítek, tak má možnost neomezeně rozšiřovat analyzační možnosti programu pouhou editací dvou textových souborů (viz část 7.4).

Po načtení dat z netlistu dojde k sestavení symbolických obvodových rovnic. Příslušný blok programu, který má tuto činnost na starosti, k tomu využívá matice – razítka jednotlivých prvků obvodu, která čte z knihovny **SNAP.CDL**. Po zadání hledané obvodové funkce se v bloku modifikace rovnic „ujasní“, které řádky a sloupce pseudoadmitanční matice se budou vynechávat při výpočtu příslušných algebraických doplňků. Tyto doplňky se pak určí v bloku symbolických algoritmů.

Ve skutečnosti se v paměti počítače nesestavuje pseudoadmitanční matice v klasické formě, ale využívají se speciální techniky pro úsporné zaznamenávání jejích prvků. Pro urychlení výpočtů byl klasický algoritmus rozvoje determinantu modifikován s využitím teorie grafů. Implementovaná metoda má díky tomu nízké nároky na velikost paměti během rozvoje determinantu a symbolická analýza probíhá velmi rychle.

Semisymbolický tvar výsledku je získán ze symbolického dosazením numerických hodnot parametrů obvodových prvků, čímž získáme číselné hodnoty koeficientů polynomů v čitateli a jmenovateli obvodové funkce. Ze semisymbolického tvaru jsou dále získány nulové body a póly, vzorce pro impulsní a přechodovou odezvu (použitím algoritmu zpětné Laplaceovy transformace) a grafy těchto odezev, jakož i grafy kmitočtových závislostí obvodových funkcí.

V následující kapitole se pokusíme na konkrétních příkladech počítačové simulace pomocí programu SNAP ukázat jednotlivé praktické kroky se simulací související. Vysvětlíme si problémy, spojené se zadáváním modelu obvodu a s jeho postupnou analýzou. V závěru kapitoly shrneme některé základní pojmy z terminologie počítačové simulace.

Je vhodné mít na paměti, že každý konkrétní simulátor má svá specifika a že postupy zde objasněné na příkladu programu SNAP nelze beze zbytku přenášet na jiné programy. To se týká například zásad práce se schématickým editorem, kde se u jednotlivých programů vyskytují velké rozdíly. Nastudováním těchto úvodních lekcí však získáme dobrý základ pro práci se simulačními programy jako takovými, jejichž dokonalé zvládnutí se stejně neobejde bez občasného nahlédnutí do manuálu nebo nápovědy.

6 Základy práce s programy pro (nejen symbolickou) analýzu obvodů

Cíle kapitoly:

- Vymezit typy úloh, které je výhodné řešit programy pro symbolickou analýzu.
- Usnadnit začínajícímu uživateli simulačních programů první praktické kroky mj. absolvováním tří názorných lekcí.
- Objasnit princip tvorby modelů součástek v programu SNAP na základě modifikované metody uzlových napětí.

6.1 Úvod

Klasické numerické simulační programy poskytují pouze kvantitativní výsledky analýzy, většinou ve formě grafů. Tyto výsledky získává uživatel programu bezprostředně po zadání modelu obvodu a požadavků na analýzu a spuštění analýzy, tj. bez jakýchkoliv mezivýsledků, které by mu pomohly v orientaci, **proč** jsou výsledky právě takové jaké vyšly.

Programy pro symbolickou a semisymbolickou analýzu jsou většinou schopny poskytovat rovněž tyto kvantitativní grafické informace, kromě toho však generují i podstatné mezivýsledky, tj. analytické vzorce. Z vzorců jsou pak patrné důležité souvislosti mezi obvodem a jeho chováním, například:

- které součástky zesilovače a které parametry tranzistoru se podílejí na tvorbě střídavého zesílení stupně,
- co musí být splněno, aby se v oscilátoru udrženy ustálené kmity a které součástky mají vliv na velikost kmitočtu,
- jaké jsou podmínky rovnováhy konkrétního střídavého můstku,
- které parametry operačního zesilovače je třeba „hlídat“, aby aktivní filtr s tímto zesilovačem měl požadovanou kmitočtovou charakteristiku,
- jaká je optimální hodnota neutralizační kapacity ve vysokofrekvenčním zesilovači,

apod.

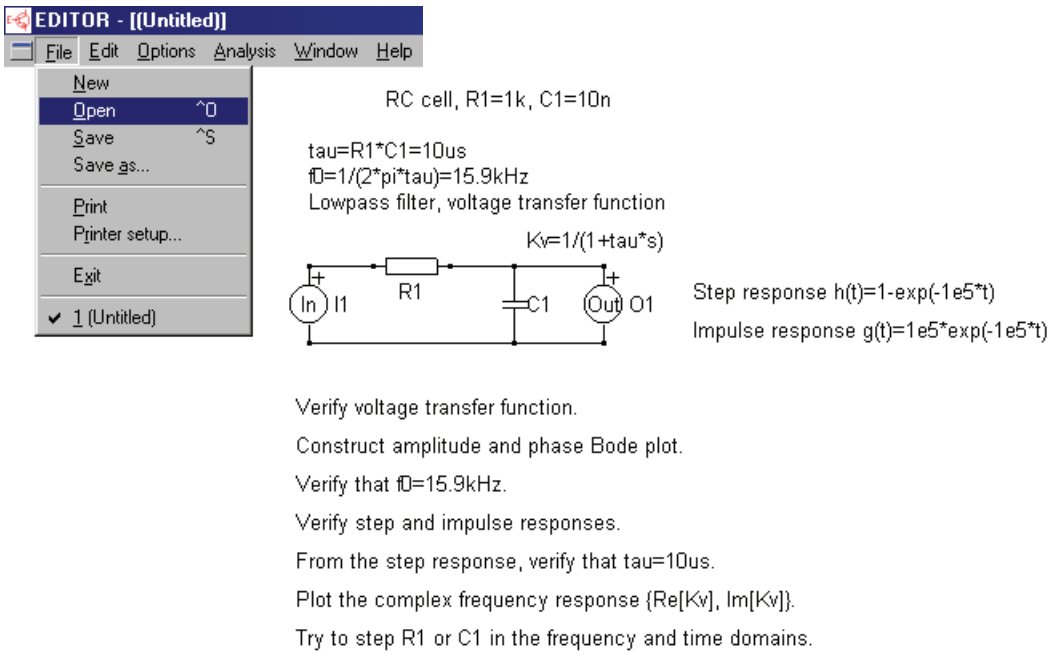
K těmto výsledkům se za pomocí klasických simulátorů nelze dopracovat buď vůbec, nebo v ojedinělých případech pracnou a neefektivní opakovanou analýzou metodou „pokusu a omylu“.

Z těchto a dalších důvodů vznikají programy pro symbolickou analýzu jako je SNAP (Symbolic Network Analysis Program). Počítačovou simulaci se nenaučíme pouhým čtením příruček. Proto bude studium této kapitoly nejefektivnější při současném experimentování s programem, jehož instalační soubory jsou dostupné na [ ]. Doporučujeme projít nejprve lekcemi č. 1, 2 a 3 a teprve pak přejít na příklady z kapitoly „Tvorba vlastního zadání“. Po zvládnutí těchto základů můžete přistoupit k řešení sady 123 příkladů z různých oblastí analogové techniky, které jsou součástí instalace. Přehled příkladů naleznete jednak v části P3 tohoto elektronického textu, jednak v souboru **EXAMPLES.XLS**, případně **EXAMPLES.PDF**. Výborným cvičením pro vás může být vaše vlastní analýza obvodů, které jsou řešeny „ručně“ v přednáškových skriptech [1.23], pomocí SNAPu. A narazíte-li na problém, využijte rozsáhlé nápovědy SNAPu, kterou můžete vyvolat po stlačení klávesy *F*1.

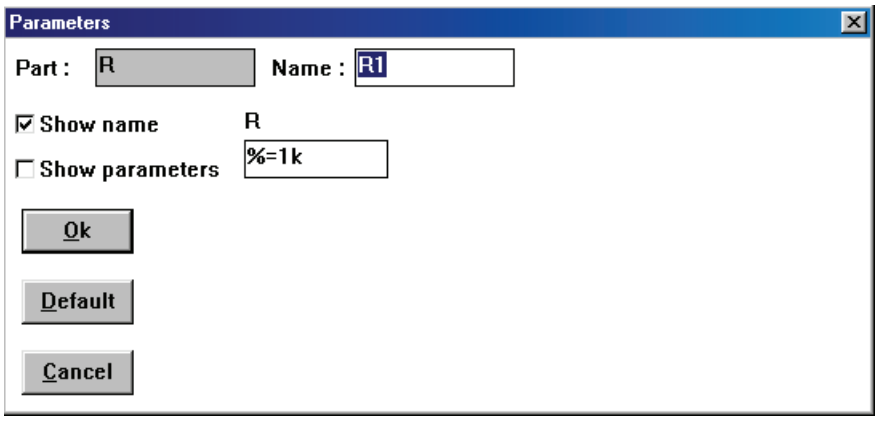
6.2 První praktické kroky k počítačové simulaci v programu SNAP

6.2.1 LEKCE 1 - Rychlé seznámení se základními možnostmi programu; soubor DEMRC.CIR

Spustíme program **EDITOR.EXE**. Zvolíme nabídku ***File/Open***. Otevře se okno, v němž se přemístíme do podadresáře *examples/basic* a vybereme soubor ***demrc.cir***. Na pracovní ploše editoru se objeví schéma s následujícím textem:



Součástky typu ***In/Out*** definují vstupní a výstupní bránu obvodu.
Dvakrát klikněme na tělo součástky R_1. Otevře se okno parametrů dané součástky.



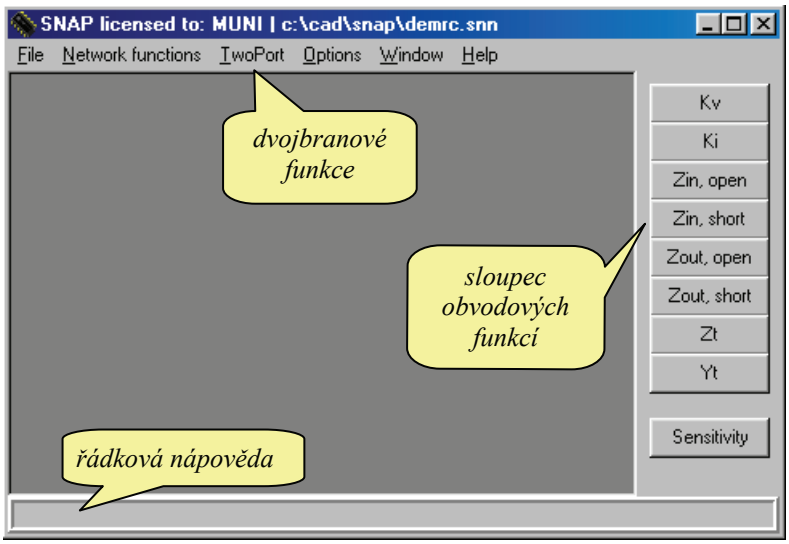
- V položce **Part** je uveden symbol *R*, tj. rezistor.
- V položce **Name** je parametr součástky v symbolickém tvaru. Zjednodušeně řečeno, tento symbol se pak objeví ve vzorci obvodové funkce jako výsledku symbolické analýzy obvodu.
- V položce **R** je definována numerická hodnota parametru R_1 na 1 $k\Omega$. Podrobnosti o možných formátech dat v této položce se dozvíme později v části 4.5.3 „Tvorba vlastního zadání“.

Okno parametrů zavřete klávesou *ESC*. Podobně si můžete prohlédnout okno parametrů kapacitoru C_1 a ověřit, že číselná hodnota kapacity je 10*n*F. Součástky typu *In* a *Out* jsou bez parametrů.

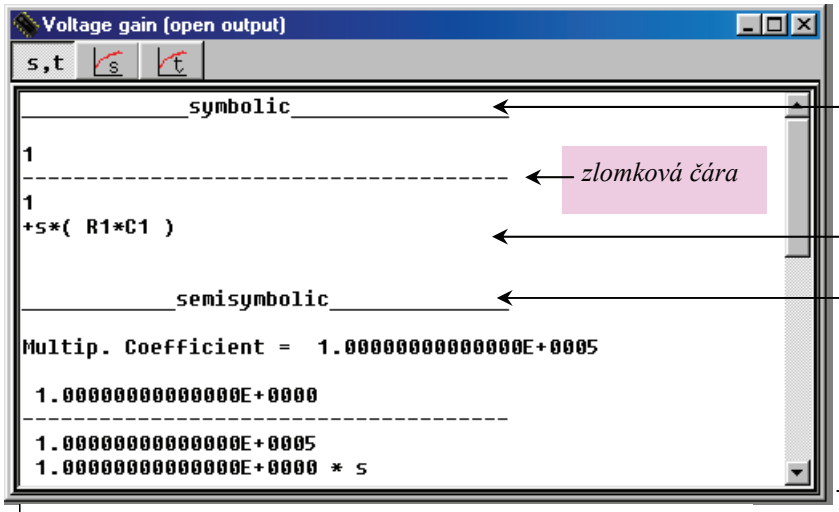
Analýza obvodu se aktivuje v nabídce ***Analysis/Snap***. Spustí se vlastní analyzační program SNAP.

Tip: detailní hypertextovou nápovědu SNAPu získáte v jakémkoliv režimu po stlačení klávesy *F*1.

Okno programu SNAP vypadá následovně:



V pravém sloupci obvodových funkcí zvolíme K_v (přenos napětí). Výsledky analýzy se objeví v okně ***Voltage gain (open output)***.

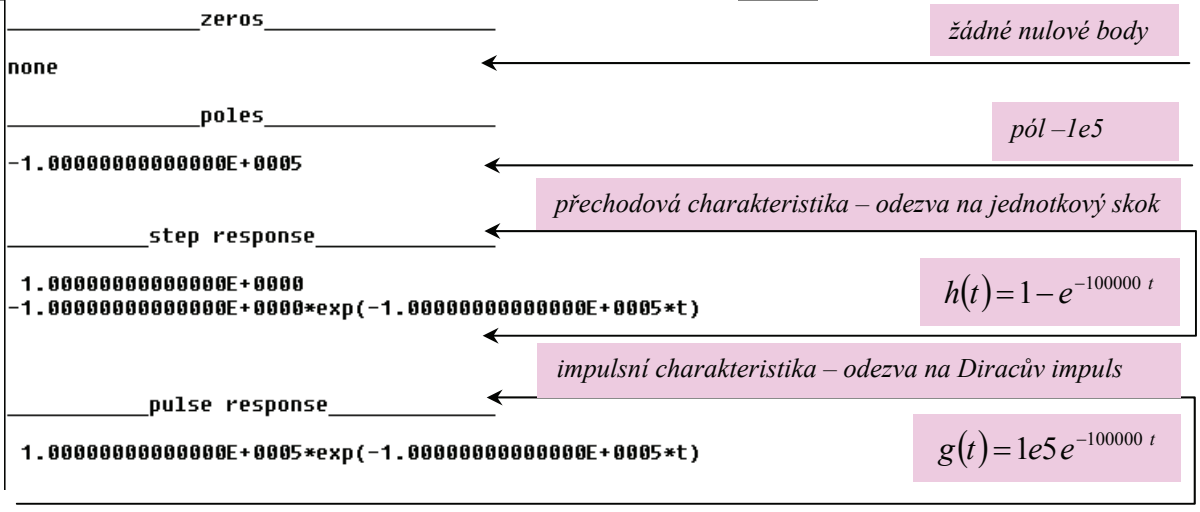


symbolická analýza:

$$K_V = \frac{1}{1 + sR_1C_1}$$

semisymbolická analýza:

$$K_V = 1e5\,\frac{1}{1e5 + s}$$



žádné nulové body

pól –1e5

přechodová charakteristika – odezva na jednotkový skok

$$h(t) = 1 - e^{-100000\ t}$$

impulsní charakteristika – odezva na Diracův impuls

$$g(t) = 1e5\,e^{-100000\ t}$$

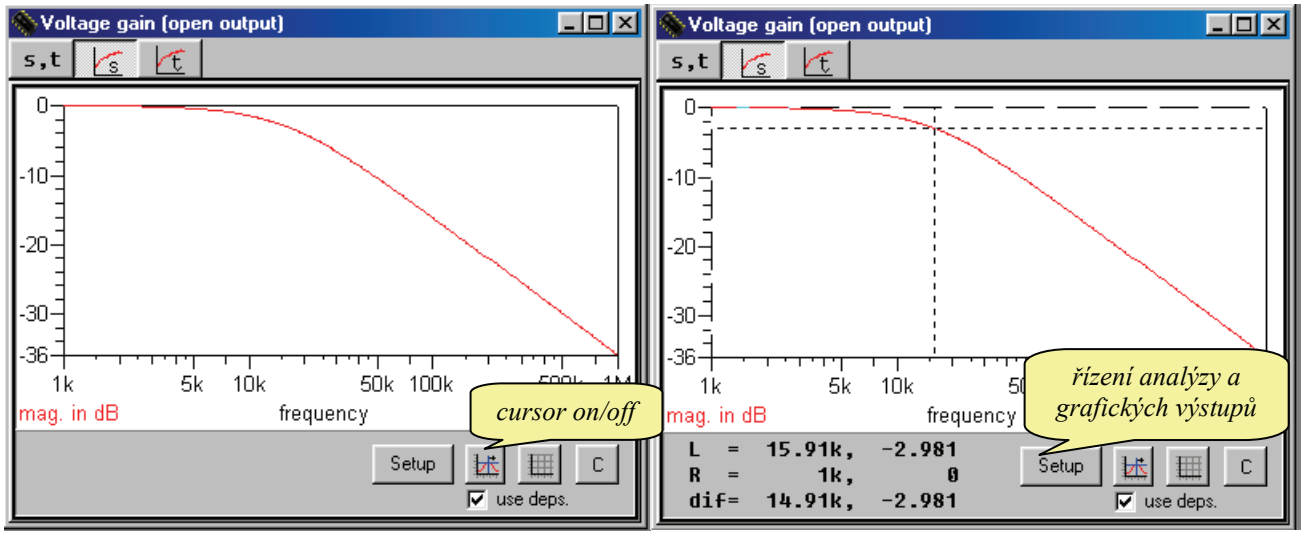
Všimněme si, že okno výsledků je uspořádáno do tří složek.

 - symbolická a semisymbolická analýza obvodové funkce

 - grafy kmitočtových charakteristik

 - grafy časových odezev.

Klikněme do složky kmitočtových charakteristik  . Objeví se graf amplitudové kmitočtové charakteristiky typu dolní propust.

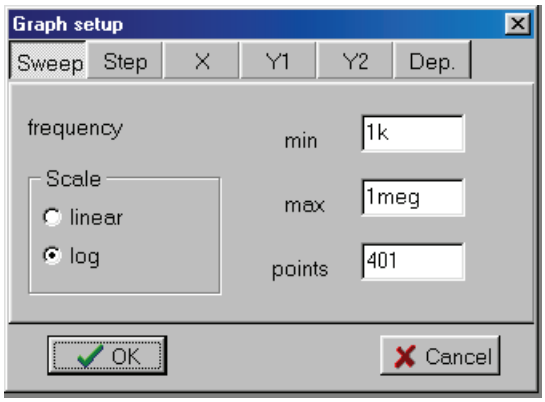


Klikneme do ikony ***cursor on/off***. Táhnutím myši za současného držení levého tlačítka můžeme pohybovat tzv. levým kurzorem (left cursor). Ve spodní části obrázku odečítáme v řádku *L* (left) souřadnice kurzoru. Zkusme změřit mezní kmitočet f_0, při němž poklesne přenos článku o 3 dB oproti referenční hodnotě stejnosměrného přenosu 0 dB: je to asi 15.9 *k*Hz (viz obrázek vpravo).

Další tip na samostatné experimenty:

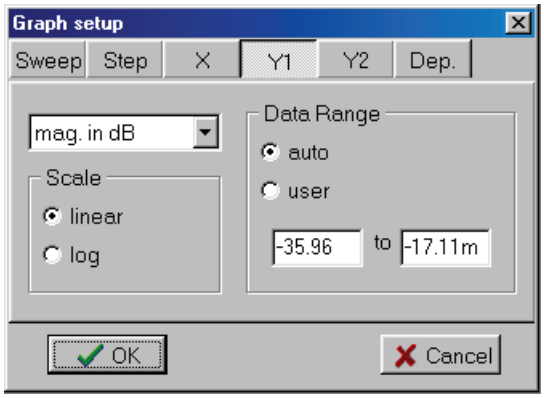
Na obrazovce je i tzv. pravý kurzor (right cursor), který lze posouvat pravým tlačítkem myši. Podrobnosti viz nápověda (*F*1).

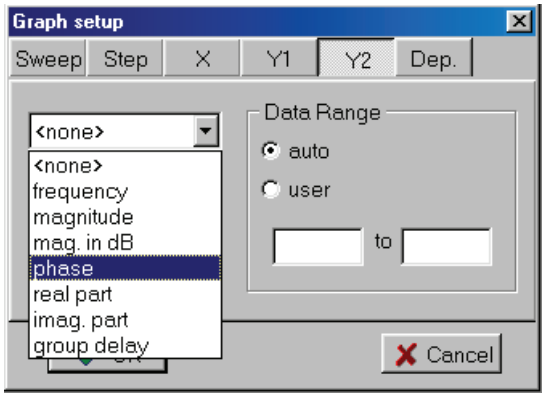
Nyní přidejme do grafu fázovou kmitočtovou charakteristiku. Klikneme na ikonu ***Setup***. Objeví se okno ***Graph setup***. Okno je uspořádáno do 6 složek (podrobnosti viz nápověda):

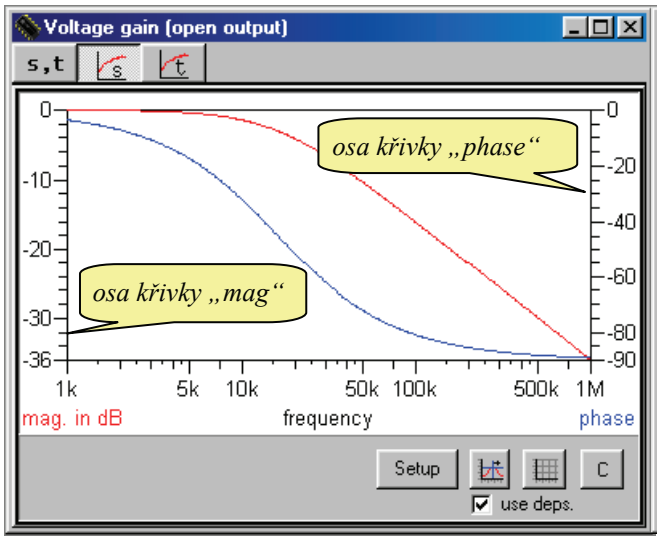


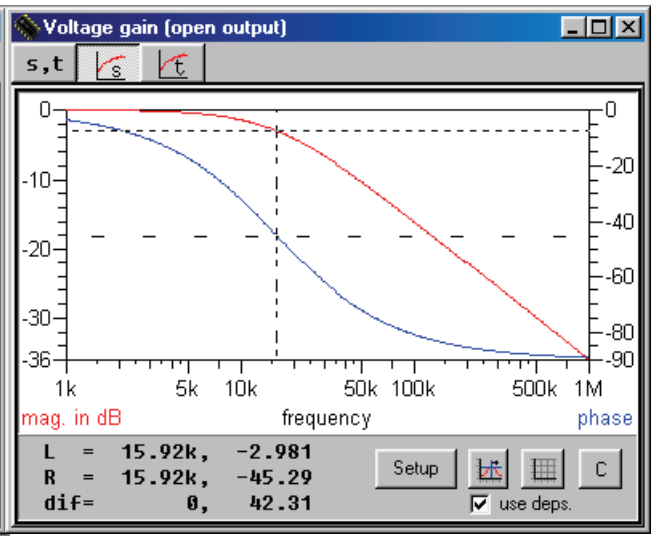
Sweep - parametry nezávisle proměnné (u kmitočtové analýzy frekvence, u časové analýzy čas): hraniční hodnoty a počet bodů výpočtu;
Step - krokování parametrů součástek pro vícenásobnou analýzu
X - definice proměnné zobrazované na vodorovné ose a parametry zobrazení
Y1 - definice proměnné č.1, které bude odpovídat křivka č.1, a parametry zobrazení
Y2 - definice proměnné č.2 ,které bude odpovídat křivka č.2, a parametry zobrazení
Dep. - editor závislostí.

Po kliknutí na složku ***Y1*** se přesvědčíme, že nastavenou proměnnou č.1 je „*mag. in dB*“. Proměnná č.2 standardně není definována (položka „*none*“ v složce ***Y2***). Změňme ji na *phase* (viz obr.). Po potvrzení *OK* získáme křivku fázové kmitočtové charakteristiky.





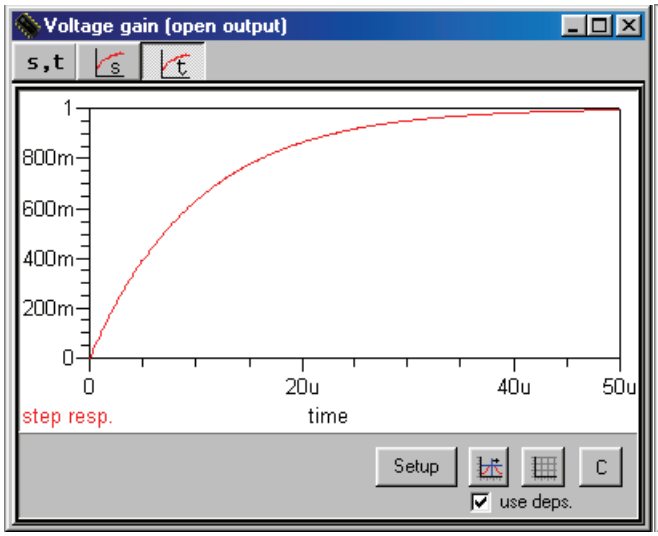


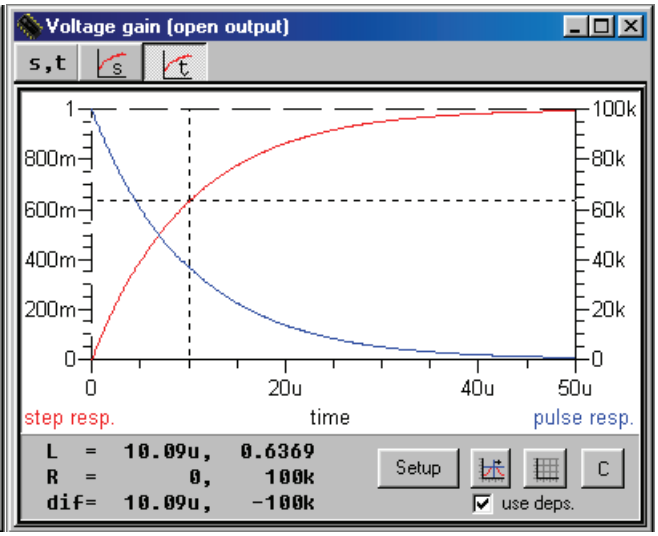


Tip:

pomocí nápovědy (*F*1) nastudujte, jak přepínat kurzory mezi křivkami. Pak ověřte, že na kmitočtu f_0 (kmitočet třídecibelového poklesu amplitudové kmitočtové charakteristiky) je fázový posuv mezi výstupním a vstupním napětím 45 stupňů.

Klikneme do složky časové analýzy  . Zobrazí se přechodová charakteristika obvodu jako jeho odezva na jednotkový skok.





Stejným postupem jako u kmitočtové analýzy přidejte křivku impulsní charakteristiky, tj. odezvy na jednotkový (Diracův) impuls (*Setup/Y2/pulse resp.*).

V režimu *Cursor on* si ověřte poučku, že kondenzátor se za časovou konstantu $\tau = R_1C_1 = 10\mu s$ nabije na 0,632 násobek konečného napětí v ustáleném stavu (viz obr. P1.8 v příloze P1 skript [1.23]).

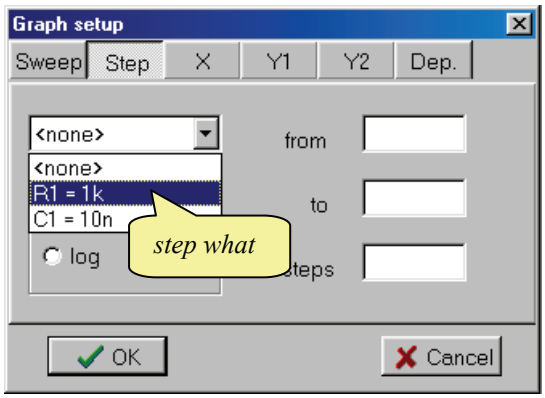
Tipy pro samostatné experimenty:

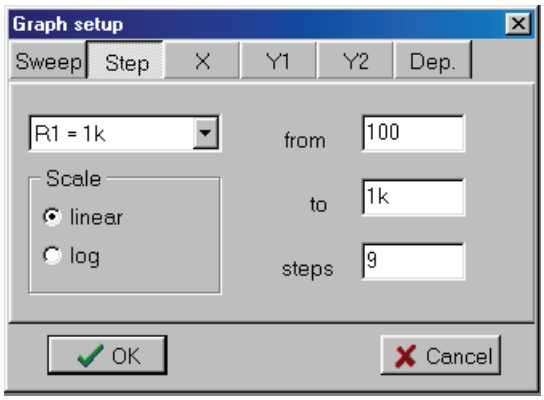
Pro přesnější odečítání souřadnic kurzorů zvětšete počet bodů výpočtu (*Setup/Sweep/points*). Detaily křivek lze získat buď úpravou parametrů *min* a *max* v složce *Sweep*, nebo v režimu *Cursor off* lze detail „vyříznout“ přímo pomocí levého tlačítka myši. Původní měřítka je možné obnovit klávesou F6.

Pokuste se v složce kmitočtové analýzy  zobrazit komplexní kmitočtovou charakteristiku {*Im*[K_V], *Re*[K_V]}: *Setup/X/real part, Y1/imag. part, Y2/none*. Prozkoumejte způsob prohlížení křivky kurzory. Zamyslete se nad souvislostmi mezi komplexní kmitočtovou charakteristikou a dílčími charakteristikami amplitudovou a fázovou.

Krokování: provedeme vícenásobnou analýzu obvodu při různých parametrech určité součástky, např. R_1.

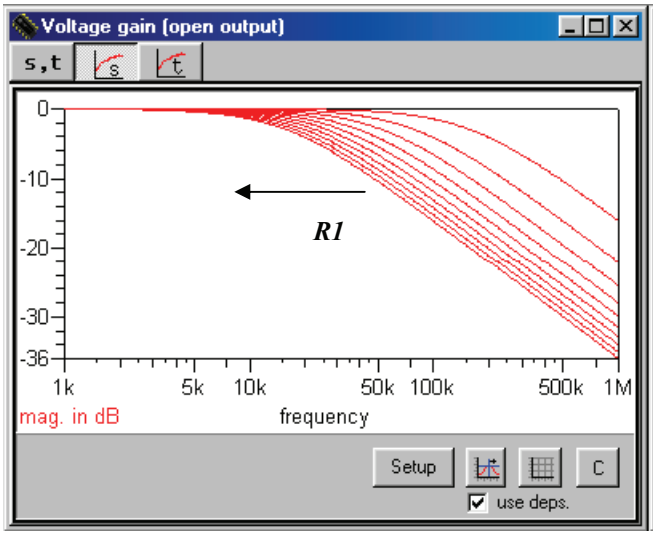
Přepneme se do složky kmitočtové analýzy  .Aktivujeme ikonu ***Setup*** a poté složku ***Step***.

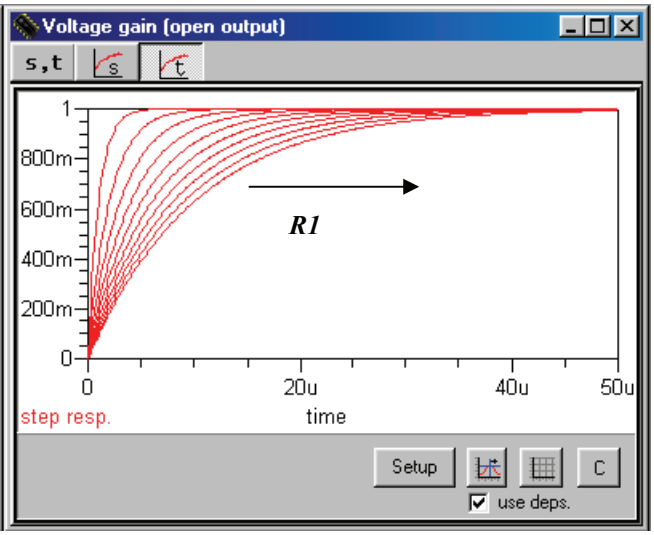




V okně *Step what* vybereme R_1=1*k* a položky *from*, *to* a *steps* vyplníme podle obrázku. Kliknutím na *OK* se vykreslí 10 kmitočtových charakteristik pro

R_1=(100,200,300,400,500,600,700,800,900,1000) Ω.





V režimu *Cursor on* můžeme kurzor přepínat po křivkách pomocí kurzorových kláves ↑↓ (levý kurzor), příp. Shift + ↑↓ (pravý kurzor). Krokovaný parametr R_1 aktuální křivky se objevuje na řádku souřadnic. Detaily viz nápověda (F1).

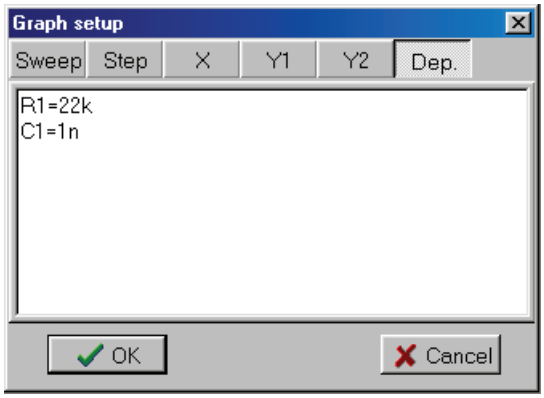
Přepněte do složky časové analýzy  a zobrazte síť přechodových charakteristik.

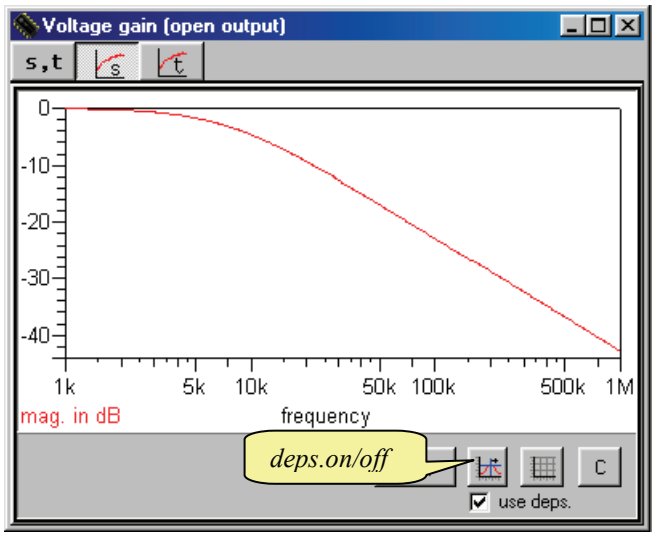
Ověřte, že při růstu R_1 klesá mezní kmitočet obvodu a roste jeho časová konstanta.

Krokování se vypíná volbou *Setup/Step/none*.

Práce s editorem závislostí.

Aby se projevily níže popsané efekty, je třeba vypnout krokování R_1 z předchozího příkladu.

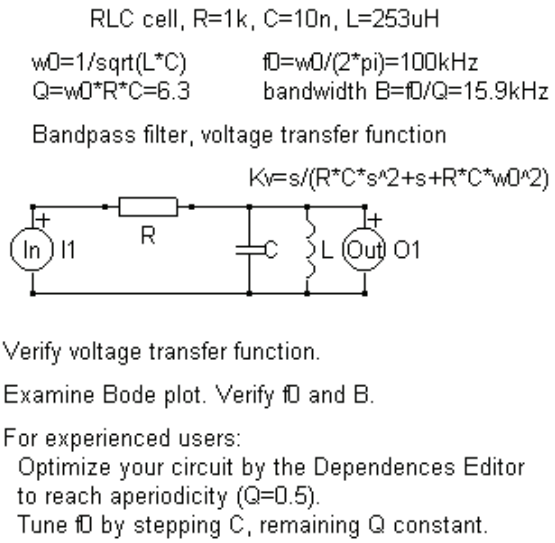


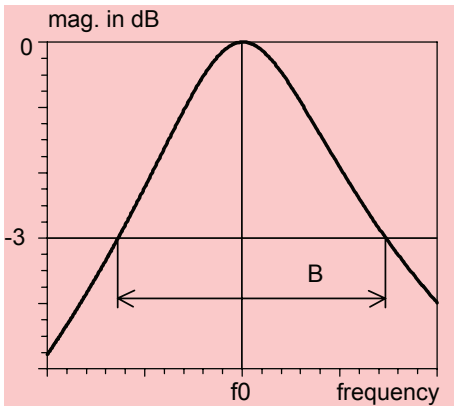


Okno editoru závislostí je poslední složkou v *Graph setup* (*Setup/Dep.*). Jde o užitečný nástroj pro dodatečnou modifikaci parametrů obvodu a zavádění vazebních podmínek mezi tyto parametry. Na úvod se seznámíme pouze s nejjednodušším použitím editoru: zápisem podle obrázku změníme parametry R_1 a C_1. Po kliknutí na „*OK*“ se provede analýza obvodu s těmito hodnotami. Vyzkoušejte si funkci přepínače *deps.on/off*, kterým můžete podmínky definované v editoru závislostí vyřadit nebo potvrdit.

Programování funkčních vztahů v editoru závislostí se naučíme v textu **Lekce2**.

6.2.2 LEKCE 2 - Rezonanční obvod RLC jako pásmová propust, soubor DEMRLC1.CIR





Obvod modelovaný v souboru demrlc1.cir se chová jako pásmová propust: Na nízkých kmitočtech induktor představuje zanedbatelnou impedanci a výstupní napětí je proto prakticky nulové. Na vysokých kmitočtech je zase výstup zkratováván zanedbatelnou impedancí kapacitoru. Největší napětí je při rezonančním kmitočtu paralelního okruhu LC: při paralelní rezonanci je impedance paralelní kombinace L a C nekonečná, výstup tedy není zatěžován a výstupní napětí se rovná vstupnímu. Přenos napětí je pak jednotkový, tj. 0 dB. Šířka pásma je definována pro třídecibelový pokles, jak je znázorněno na obrázku.

Analýza programem SNAP: Provedeme analýzu přenosu napětí (K_v – voltage gain V_{out}/V_{in}, $I_{out} = 0$). V složce **s,t** se objeví výsledky:

```
_____________symbolic_____________________
s*( L )
-------------------------------------
R
+s*( L )
+s^(2)*( R*C*L )
_____________semisymbolic_________________
Multip. Coefficient =  1.00000000000000E+0005
 1.00000000000000E+0000 * s
-------------------------------------
 3.95256916996047E+0011
 1.00000000000000E+0005 * s
 1.00000000000000E+0000 * s^(2)
_____________________zeros_____________________
 0.00000000000000E+0000
_____________________poles_____________________
-5.00000000000000E+0004 + j  6.26703212849629E+0005
-5.00000000000000E+0004 - j  6.26703212849629E+0005
_____________step response_____________
 1.59565162503793E-0001*exp(-5.00000000000000E+0004*t)
*sin( 6.26703212849629E+0005*t)
_____________pulse response_____________
 1.00000000000000E+0005*exp(-5.00000000000000E+0004*t)
*cos( 6.26703212849629E+0005*t)
-7.97825812518963E+0003*exp(-5.00000000000000E+0004*t)
*sin( 6.26703212849629E+0005*t)
```

Výsledky symbolické analýzy:

$$K_V = \frac{sL}{R + sL + s^2 RLC}$$

Výsledky semisymbolické analýzy (za R, L a C se dosadí numerické hodnoty):

$$K_V = 1e5 \frac{s}{3.952569e11 + 1e5s + s^2}$$

Nulové body (kořeny čitatele):

$$s = 0$$

Póly (kořeny jmenovatele):

$$s_{1,2} = -5e4 \pm j6.26703e5$$

Přechodová charakteristika (odezva na skok):

$$h(t) = 0.159565e^{-50000t} \sin(626703t)$$

Impulsní charakteristika (odezva na Diracův impuls):

$$g(t) = e^{-50000t}[100000\cos(626703t) - 797.8258\sin(626703t)]$$

$$K_V = \frac{sL}{R + sL + s^2 RLC} = \frac{s/RC}{\underbrace{1/LC}_{\omega_0^2} + s\underbrace{/RC}_{\omega_0/Q} + s^2}$$

$$\omega_0 = \frac{1}{\sqrt{LC}}$$

$$\frac{\omega_0}{Q} = \frac{1}{RC} \Rightarrow Q = \omega_0 RC = \frac{R}{\sqrt{L/C}}$$

Z výsledků symbolické analýzy je možné určit parametry ω_0 a Q obvodu (viz obsah rámečku).

Reálná složka pólů je záporná, což potvrzuje stabilitu obvodu.

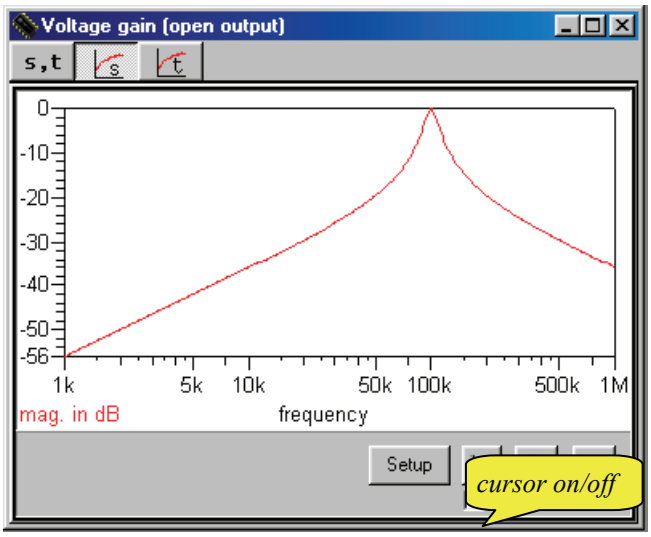
Póly jsou komplexní, takže odezva obvodu na skokové vstupní signály bude zakmitávat. Kmitočet zákmitů ω_K, resp. f_K je dán imaginární částí:

ω_K = 6.26703212849629e5 rad/s, $f_K = \omega_K/2\pi$ = 99.75kHz, odpovídající perioda zákmitů vychází $1/f_k \doteq 10.026\mu s$.

Časová konstanta τ tlumení těchto zákmitů je dána reciprokou hodnotou reálné složky pólů: $\tau = 1/50000 = 20\mu s$.

Tyto údaje jsou pak potvrzeny vzorci pro přechodovou a impulsní charakteristiku.

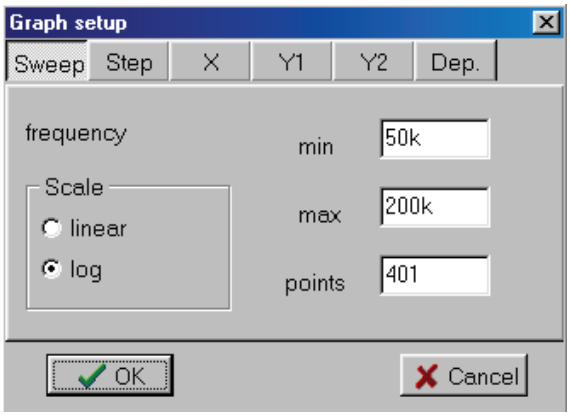
Kmitočtová analýza – složka  :

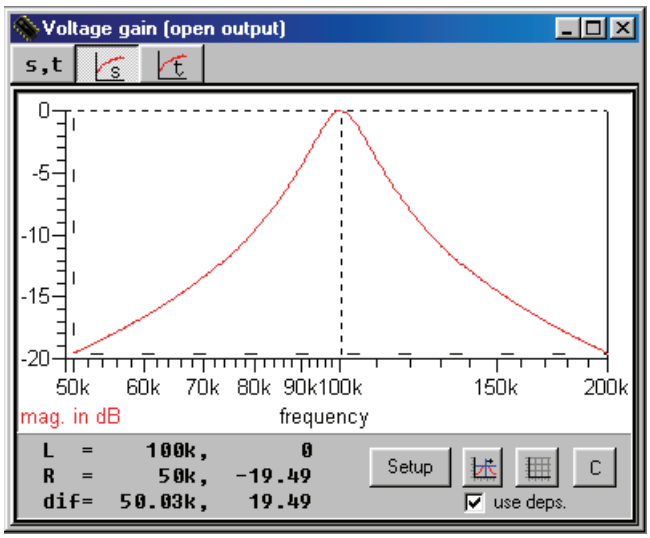


Zobrazíme detail amplitudové kmitočtové charakteristiky v okolí rezonančního kmitočtu 100 *k*Hz: klikneme na ikonu *Setup* a ve složce *Sweep* nastavíme rozmítání kmitočtu od 50 *k*Hz do 200 *k*Hz.

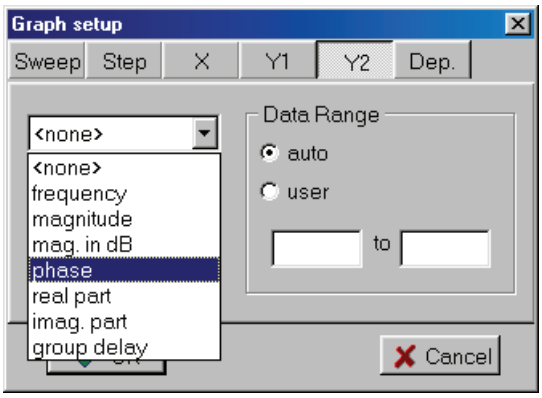
Po kliknutí na *OK* nebo *Enter* se charakteristika vykreslí v daném detailu.

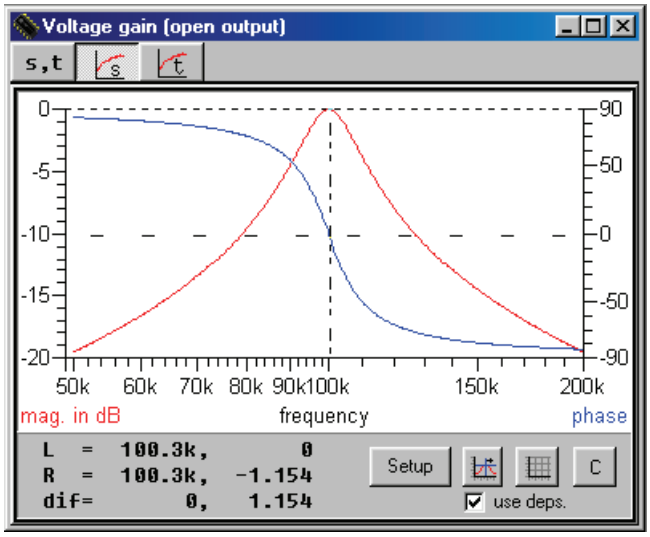
Klikněte do ikony *Cursor on/off*. Pomocí levého a pravého tlačítka myši nyní můžete ovládat pozici tzv. levého a pravého kurzoru a odečítat jejich souřadnice. Detailní popis je možno nalézt v nápovědě po stlačení klávesy *F*1. Pomocí kurzorů ověřte velikost rezonančního kmitočtu 100 *k*Hz a šířky pásma 15.9*k*Hz.





Volbou *Y2/phase* přidejte do grafu křivku fázové charakteristiky a přesvědčte se, že při rezonanci je fázový posuv mezi vstupním a výstupním napětím nulový.

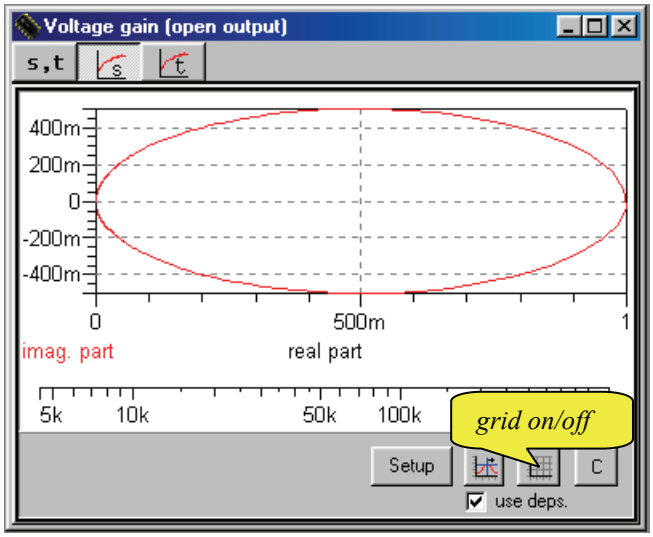




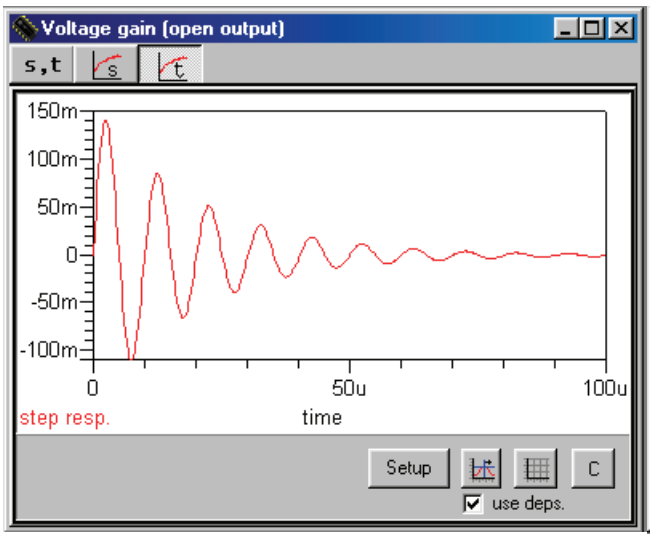
Nastavte rozmítání kmitočtu od 5 *k*Hz do 500 *k*Hz, 600 bodů výpočtu. Volbou *X/real part, Y1/imag.part.* a *Y2/none* zobrazte komplexní kmitočtovou charakteristiku podle obrázku. Přesvědčte se, že pod/na/nad rezonančním kmitočtem je imaginární složka přenosu kladná/nulová/záporná. Zamyslete se nad souvislostí mezi touto komplexní kmitočtovou charakteristikou a dílčími charakteristikami amplitudovou a fázovou.

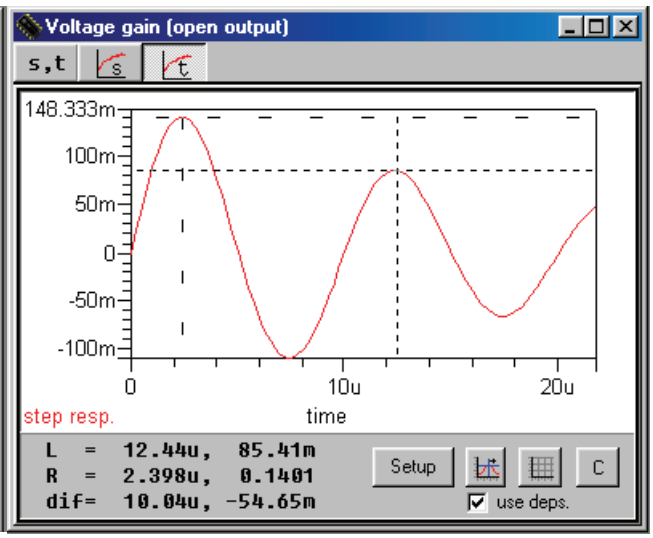
Vyzkoušejte si režim *grid on/off*.

V režimu *Cursor on* se naučte odečítat souřadnice kurzorů a parametrický kmitočet.



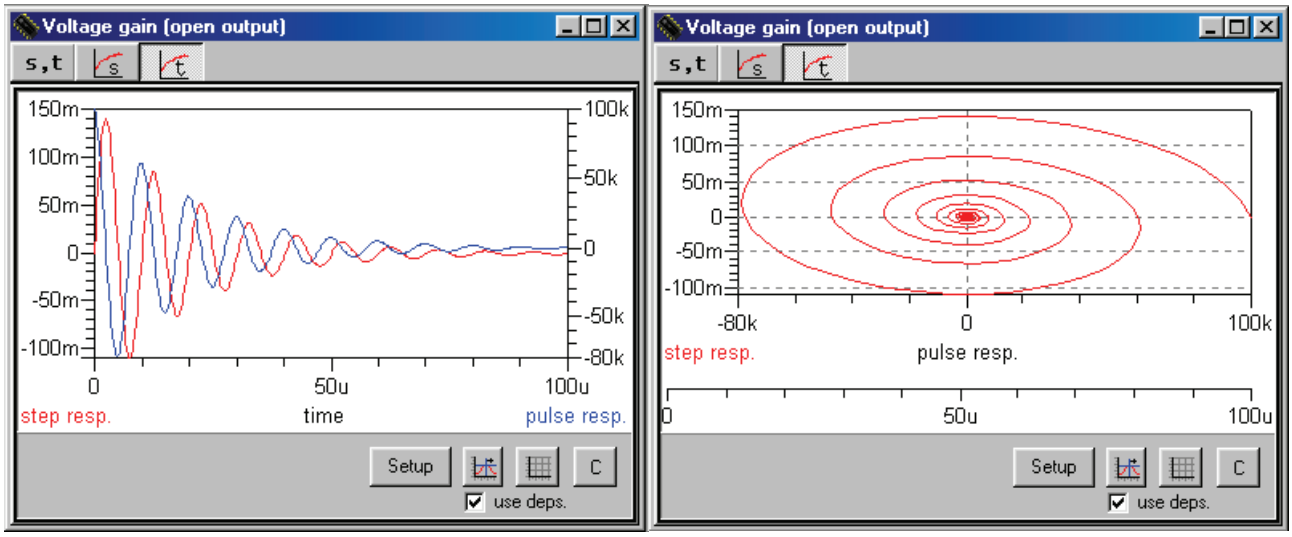
Časová analýza – složka  :





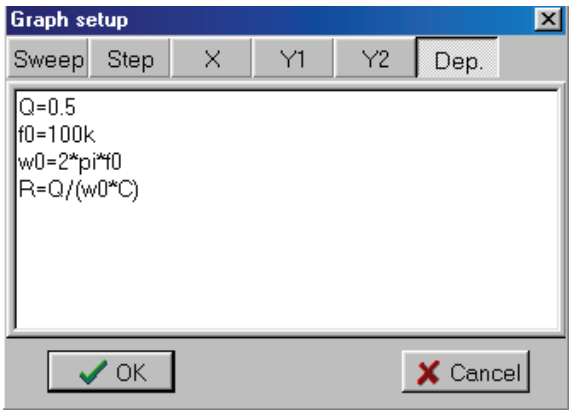
V režimu *Cursor on* změřte periodu zákmitů přechodné charakteristiky a porovnejte s teoretickou hodnotou $1/f_k \doteq 10.026\mu s$.

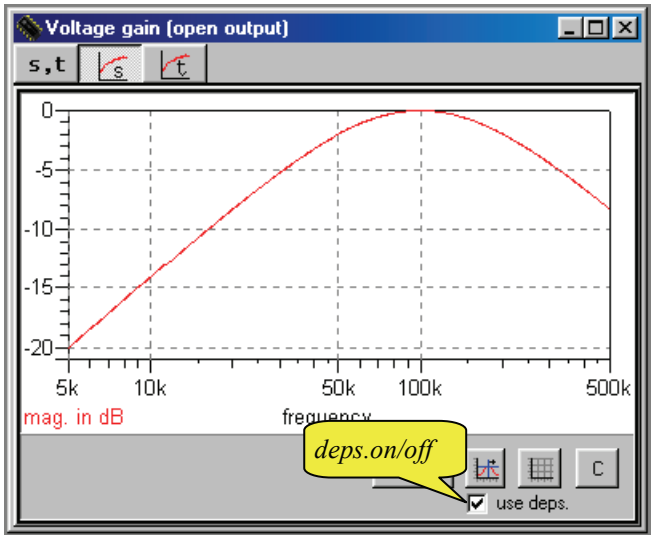
Volbou *Y2/pulse resp.* přidejte do grafu křivku impulsní charakteristiky. Porovnáním obou křivek ověřte poučku, že impulsní charakteristika je derivací přechodové charakteristiky.



Volbou *X/*pulse resp., Y1/step resp. a Y2/none zobrazte fázovou trajektorii podle obrázku. Protože se jedná o stabilní obvod, spirála konverguje do bodu, který představuje stejnosměrný ustálený stav.

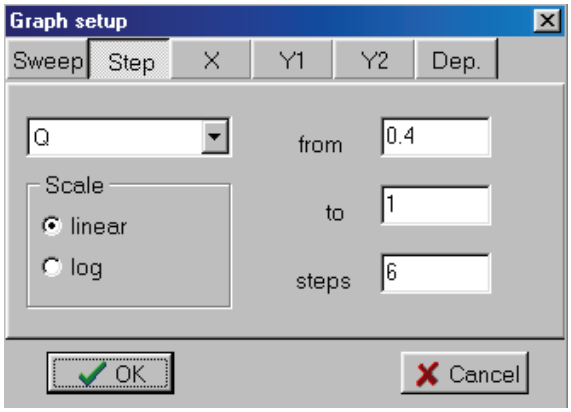
Optimalizace obvodu: nastavení činitele jakosti na hodnotu 0.5 (mez periodicity) pomocí *R*.





Nejprve nastavíme standardní režimy kmitočtové a časové analýzy (*mag. in dB, step response*). Poté vyplníme okno *Setup/Dep*. (okno Editoru závislostí) podle obrázku.

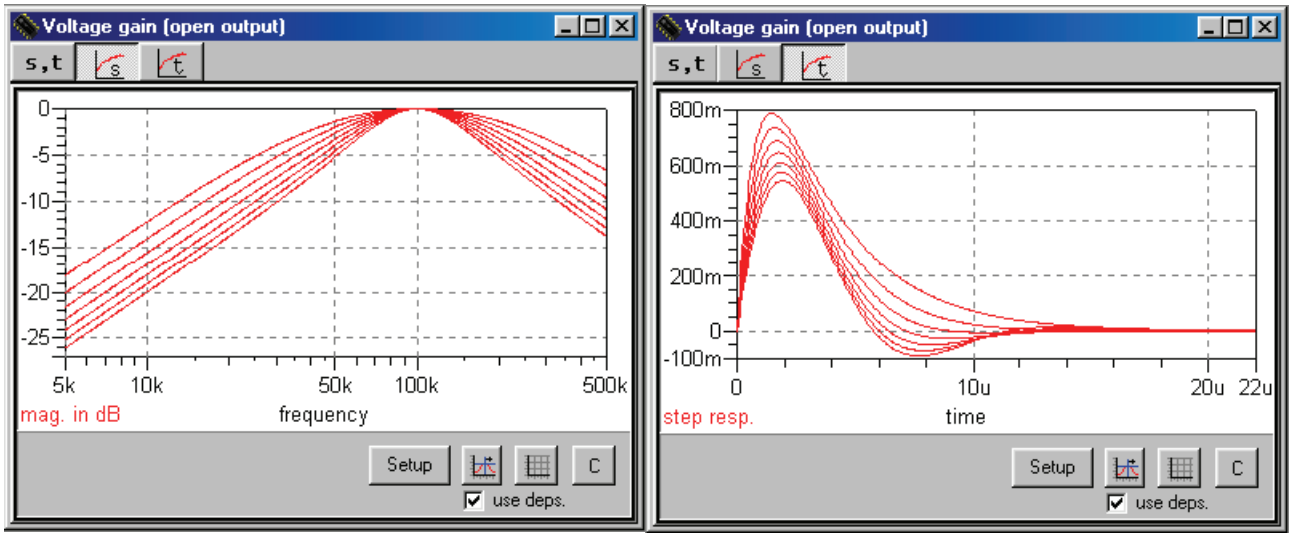
- V prvním řádku je definována proměnná *Q* a je jí přiřazena hodnota 0.5.
- V druhém řádku je definována proměnná f_0 a je jí přiřazena hodnota 100*k*.
- V třetím řádku je zavedena proměnná ω_0 jako 2π násobek f_0.
- V posledním řádku je z výše definovaných parametrů navržena hodnota *R*.



Po potvrzení „*OK*“ jsou k dispozici výsledky analýzy takto modifikovaného obvodu.

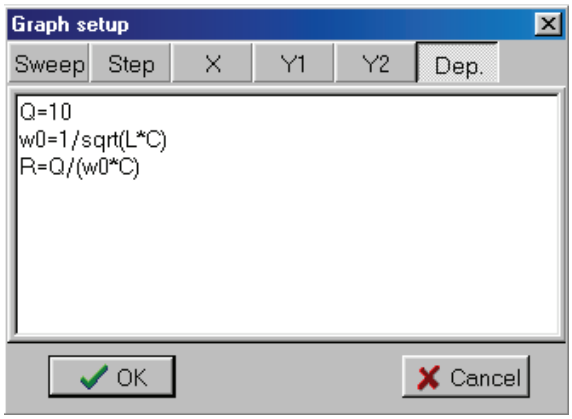
Podmínky definované v okně *Editoru závislostí* lze vypnout/zapnout pomocí položky *use deps.*

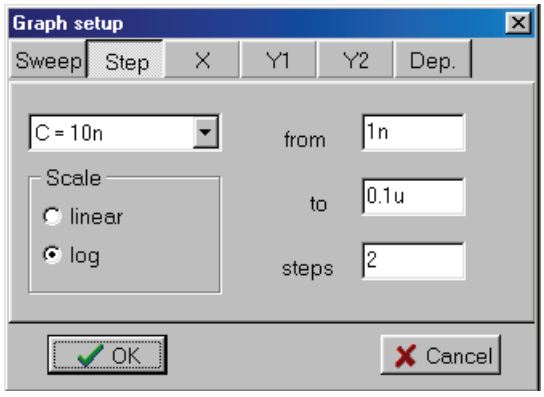
Zkusme nyní krokovat veličinu *Q* definovanou v *Editoru závislostí* v hodnotách 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 a 1. Složku *Setup/Step* vyplníme tak, jak je uvedeno na obrázku. Získáme tak 7 křivek kmitočtové charakteristiky ve složce  a 7 křivek přechodové charakteristiky ve složce  .



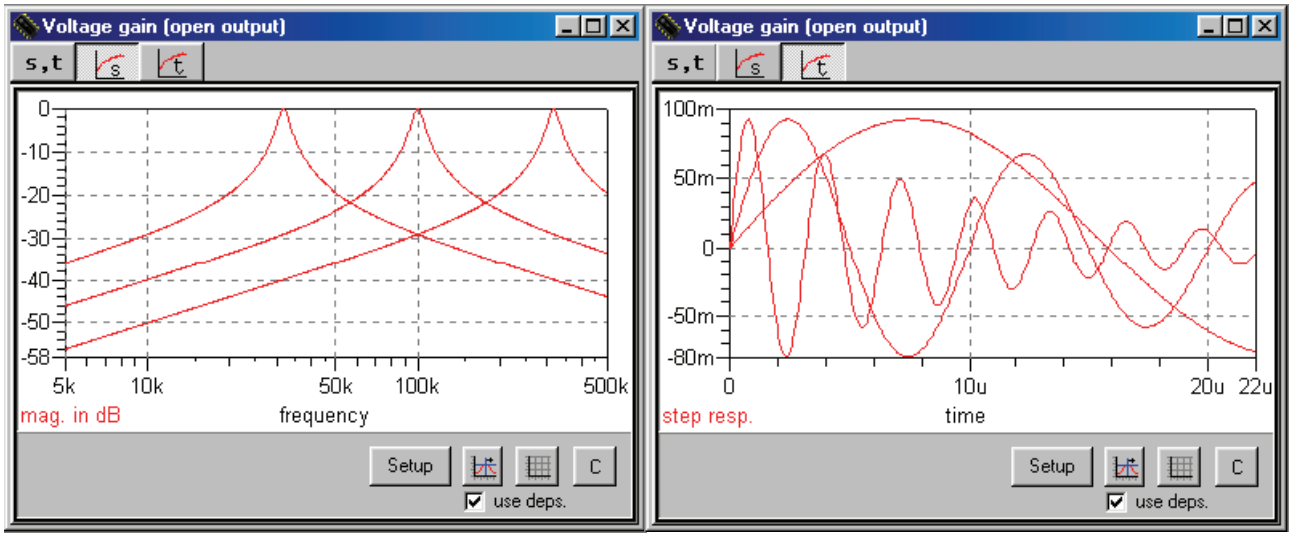
Ladění obvodu: změna rezonančního kmitočtu pomocí C při konstantním činiteli jakosti.

Ladíme-li filtr pomocí C, mění se činitel jakosti a tím i tvar amplitudové kmitočtové charakteristiky. Abychom při přelaďování udrželi Q na konstantní hodnotě, budeme změnu Q korigovat změnou R.





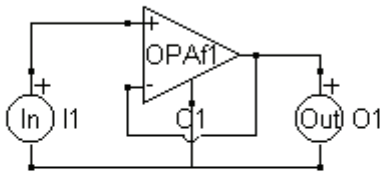
Na obrázku je ukázáno, jak toho docílit pomocí *Editoru závislostí*. Nejprve je nastaven činitel jakosti na hodnotu 10. Pak je vypočtena pomocná proměnná – rezonanční kmitočet ω_0, který pak poslouží k výpočtu odporu. Budeme-li nyní krokovat *C*, automaticky se bude přepočítávat i *R* tak, aby činitel jakosti zůstal konstantní. Na obrázcích jsou výsledky analýzy pro $C = 1nF$, $10nF$ a $100nF$ (logaritmická metoda krokování).

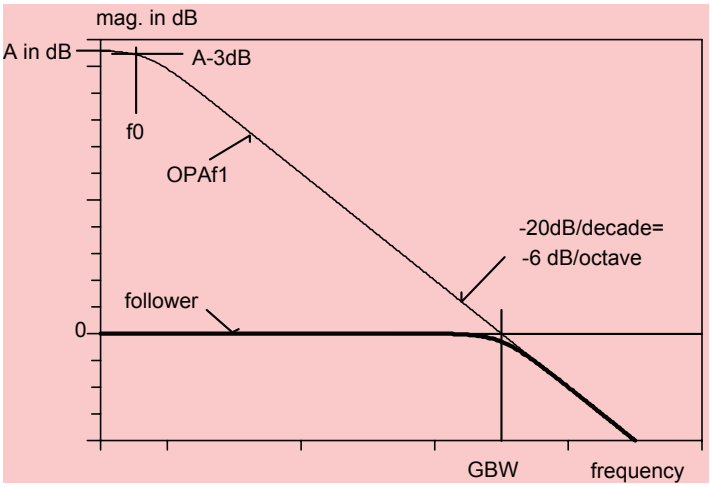


6.2.3 LEKCE 3 - Operační zesilovač zapojený jako sledovač napětí – jednopólový model; soubor DEMOPA1.CIR

OpAmp follower

One-pole model of 741: GBW=1MHz
Ao=200k
Rout=50 ohms





Verify voltage transfer function

Kv=2*pi*GBW/[s+2*pi*GBW*(1+1/A)]

Examine Bode plot. Verify GBW.

Examine step and impulse responses.
Verify that the transient processes are monotonic
with the time constant 1/(2*pi) microseconds.

Ideální operační zesilovač (v knihovně SNAPu zkratka OPA) má nekonečné a kmitočtově nezávislé napěťové zesílení, nekonečný vstupní a nulový výstupní odpor. Spojíme-li invertující vstup s výstupem, vznikne obvod se stoprocentní zápornou zpětnou vazbou. Pak v důsledku nekonečného zesílení vlastního operačního zesilovače se diferenční napětí, tj. napětí mezi vstupy + a -, ustálí na nule. Výstupní napětí pak bude přesně odpovídat vstupnímu napětí. Vstupní odpor celého obvodu bude nekonečný, výstupní odpor nulový a přenos napětí bude jednotkový. Hovoříme o ideálním oddělovacím zesilovači, o sledovači napětí, bufferu apod.

Skutečný operační zesilovač vykazuje konečné a kmitočtově závislé zesílení a lze jej popsat pomocí parametrů *A* a *GBW* (viz obrázek). *A* je tzv. stejnosměrné zesílení, které je sice velmi vysoké, avšak oproti ideálnímu operačnímu zesilovači konečné. S růstem kmitočtu

zesílení klesá: Na kmitočtu f_0, který je relativně nízký (u OPA 741 je to asi 5 Hz), je již zesílení menší o 3 dB. Při dalším růstu kmitočtu zesílení klesá s rychlostí 20 dB/dekádu (6 dB/oktávu), což znamená, že při vzrůstu kmitočtu desetinásobně (dvojnásobně) klesne zesílení vždy o 20 dB (6 dB). Na kmitočtu *GBW* (Gain Bandwidth Product) je již zesílení jen 0 dB, což znamená, že zesilovač již zcela ztratil svou zesilovací schopnost. Při ještě vyšších kmitočtech se prvek již chová jako zeslabovač.

Přesnější modelování chování OPA pro kmitočty nad *GBW* pak vede na tzv. **dvoupólový model** (viz soubor **demopa2.cir**). Od jistého kmitočtu f_2 totiž začíná zesílení klesat rychleji se strmostí 40 db/dekádu.

Z dalších neideálních vlastností, které mají často negativní vliv na chování obvodů, je nenulový výstupní odpor operačního zesilovače R_o.

Zapojíme-li reálný operační zesilovač jako sledovač, kmitočtová charakteristika se změní podle obrázku. Zesílení sice bude 0 dB, což odpovídá sledovači, ale jen zhruba do kmitočtu *GBW*. Na vyšších kmitočtech pak už dochází k zeslabování signálu. Dále se projevuje zpoždění signálu průchodem sledovače, což souvisí s jeho fázovou kmitočtovou charakteristikou.

Počítačová analýza:

V schématickém editoru otevřete soubor **demopa1.cir**. Poklepáním na značku operačního zesilovače se přesvědčete, že jsou zadány jeho parametry A=200k, GBW=1MEG, R_o=50Ω.

Provedeme analýzu přenosu napětí (K_v – voltage gain V_{out}/V_{in}, $I_{out}=0$). V složce **s,t** se objeví výsledky:

```
_____________symbolic_____________________
6.28319*A*GBW
-----------------------------------
6.28319*A*GBW +6.28319*GBW
+s*( A )
_____________semisymbolic_________________
Multip. Coefficient =  6.28318530000000E+0006
 1.00000000000000E+0000
-----------------------------------
 6.28321671592650E+0006
 1.00000000000000E+0000 * s
_________________zeros_____________________
none
_________________poles_____________________
-6.28321671592650E+0006
_____________step response_________________
 9.99995000025000E-0001
-9.99995000025000E-0001*exp(-
6.28321671592650E+0006*t)
_____________pulse response________________
 6.28318530000000E+0006*exp(-
6.28321671592650E+0006*t)
```

Výsledky symbolické analýzy:

$$K_V = \frac{2\pi\, AGBW}{2\pi\, AGBW + 2\pi\, GBW + sA}$$

Výsledky semisymbolické analýzy:

$$K_V = 2\pi.10^6 \frac{1}{2\pi.10^6 + s}$$

Nulové body neexistují.
Pól (kořen jmenovatele):

$$s = -2\pi.10^6$$

Přechodová charakteristika:

$$h(t) \doteq 1 - e^{-2\pi.10^6 t}$$

Impulsní charakteristika:

$$g(t) = 2\pi.10^6\, e^{-2\pi.10^6 t}$$

Z přenosové funkce je možno potvrdit, že:

- Stejnosměrné zesílení celého obvodu je

$$A_0 = K_V(s = j\omega = 0) = \frac{2\pi\, A\, GBW}{2\pi\, A\, GBW + 2\pi\, GBW} = \frac{1}{1+\frac{1}{A}} = \frac{1}{1+\frac{1}{200000}} \doteq 0{,}999995 \doteq 1 \hat{=} 0dB\,.$$

Pól přenosové funkce (kořen jmenovatele) závisí na *GBW* a *A* podle vzorce:

$$2\pi\, A\, GBW + 2\pi\, GBW + sA = 0 \Rightarrow s = -2\pi\, GBW\left(1+\frac{1}{A}\right) \doteq -2\pi\, GBW\,,$$

takže lomový kmitočet kmitočtové charakteristiky je prakticky roven *GBW*.

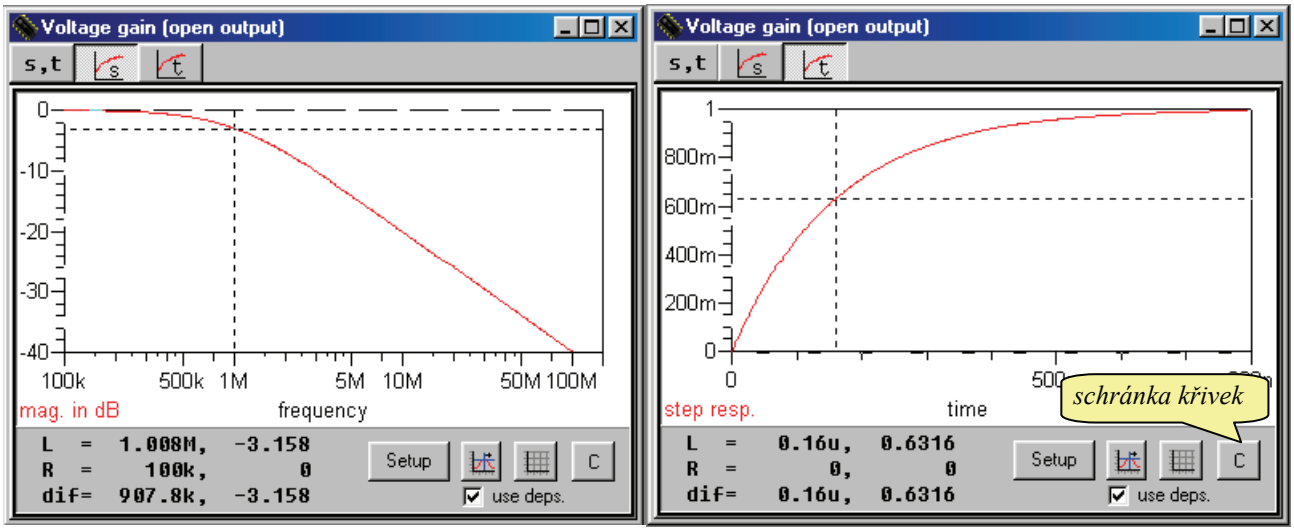
- Přechodné děje v obvodu jsou popsány exponenciálními funkcemi typu

$$e^{-t/\tau} = e^{-2\pi.10^6 t}\,,$$

takže odpovídající časová konstanta je

$$\tau = \frac{1}{2\pi e6} \doteq 159ns\,.$$

- V tomto zapojení se neuplatní vliv výstupního odporu OPA R_o (nefiguruje ve vzorcích). Pokuste se o vysvětlení!



Klikneme do složky kmitočtové analýzy  .

Pomocí kurzoru ověříme, že mezní kmitočet sledovače (pro pokles přenosu o 3 dB) je 1 MHz.

Klikneme do složky časové analýzy  . Pomocí kurzoru ověříme poučku, že za časovou konstantu τ=159μs výstupní napětí dospěje na 0,632 násobek konečného napětí v ustáleném stavu.

Pokusme se nyní ke kmitočtové charakteristice sledovače přikreslit charakteristiku samotného operačního zesilovače. Za tím účelem bude třeba:

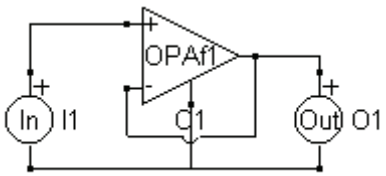
1. Uložit aktuální charakteristiku do tzv. **schránky křivek**;
2. V editoru upravit schéma – rozpojit zpětnou vazbu z výstupu na invertující vstup a tento vstup spojit se společným vodičem;
3. Zopakovat analýzu takto modifikovaného obvodu;
4. Přidat výslednou charakteristiku do schránky křivek a provést srovnání.

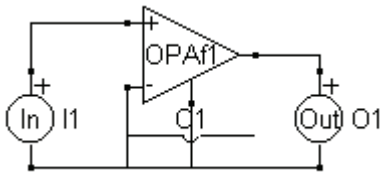
Klikněte do složky kmitočtové analýzy  . Pak klikněte do ikony schránky křivek  . Objeví se okno, žádající nás o zadání názvu křivky. Zadáme například 1. Vytvoří se okno „*Clipboard*“, do níž se překopíruje daná křivka.

Nyní na spodní liště najdeme zástupce „Editor“ a přepneme se do prostředí schématického editoru.

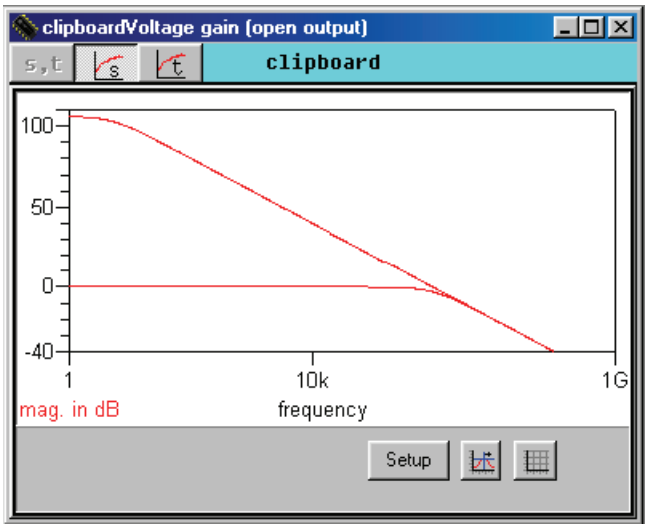
Poznámka: vyskytnou-li se při práci s editorem problémy, přečtěte si informace z kapitoly „Co je nutné vědět před zahájením práce s editorem“.

Ujistíme se, že se nacházíme v režimu *Select* (musí být aktivní položka *Select* v spodní části obrazovky). Pomocí levého tlačítka myši „uchopíme“ vertikální část spoje z výstupu OPA do invertujícího vstupu, přemístíme jej tak, abychom spojili invertující vstup se společným vodičem, a spoj umístíme uvolněním tlačítka. Poté klikneme do libovolného místa





na plochu mimo součástky. Výsledek by měl odpovídat obrázku vpravo. Zbylou část spoje nemusíme mazat, na výsledku analýzy se neprojeví. Analýzou takto upraveného obvodu nyní získáme přenosovou funkci samotného operačního zesilovače. Spustíme opět SNAP (*Analysis/Snap* nebo *F*11). Ve výsledkovém okně *Voltage gain* se nyní objeví charakteristika blízká přímce (efekt nevhodného měřítka). Tuto charakteristiku přidáme do schránky křivek ( , zadáme název křivky, např. 2). Okno schránky má podobné atributy (*Setup, kurzory*,..) jako obyčejné výsledkové okno. Do schránky se nepřenášejí vypočtené body křivky, ale celé vzorce, takže lze s křivkami dále plnohodnotně pracovat. Pomocí *Setup/Sweep* nastavte rozmítání kmitočtu od 1 Hz do 1 *G*Hz. Dostanete výsledek podle obrázku.



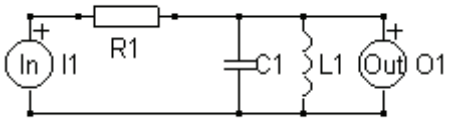
Na tomto místě si můžete vyzkoušet některé možnosti exportu vámi vytvořených dat, které jsou nabízeny v položce *Results*. Například položka *Copy Graph* zkopíruje obsah grafického okna do schránky Windows s možností dalšího zpracování grafiky v aplikacích Windows apod. Podrobnosti naleznete ve vestavěné nápovědě (*F*1).

Před ukončením práce s editorem schémat se můžete pokusit o uvedení schématu do původního stavu. Pokud se vám to nepodaří, raději modifikované schéma neukládejte, neboť byste si tím změnili originální soubor **demopa1.cir**.

6.3 Tvorba vlastního zadání

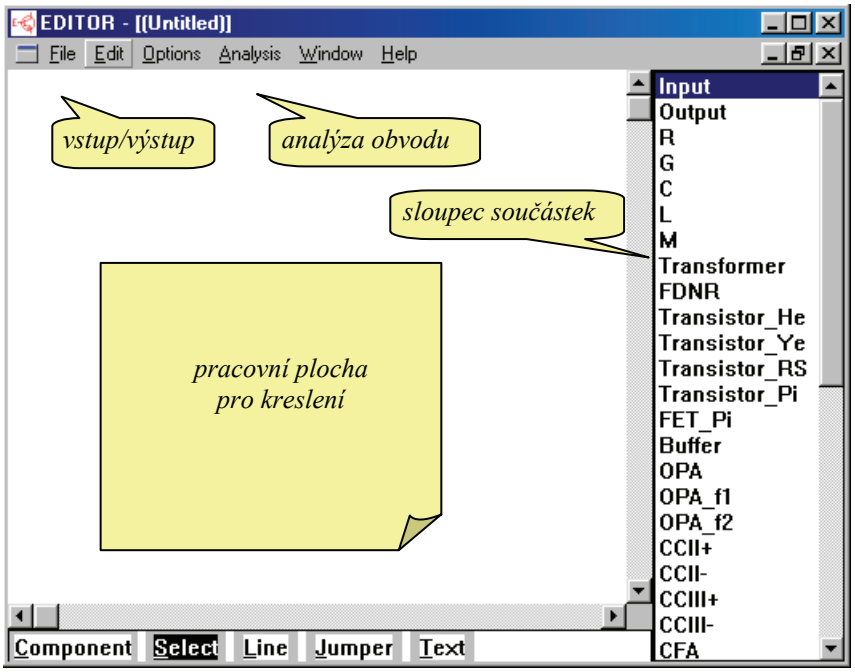
6.3.1 Můj první obvod v SNAPu

Nakreslete schéma *RLC* obvodu podle obrázku (viz též vzorový soubor **demrlc1.cir**). Součástkám přiřaďte parametry R_1=1 *k*Ω, C_1=10 *n*F, L_1=253 μH.



Vysvětlující poznámka: Součástky typu *In* a *Out* jsou vstupní a výstupní dvojpóly. Umístěním součástky *In*, resp. *Out* mezi 2 uzly definujeme vstupní, resp. výstupní svorky obvodu. Zda je vstupní, resp. výstupní veličinou napětí nebo proud, se dodatečně určí až při analýze obvodu v programu SNAP.

Po spuštění programu **EDITOR.EXE** se objeví jeho úvodní obrazovka:



Lišta režimů editoru:

Component: pokládání schématických značek na plochu pro kreslení.

Select: Editace již položených značek, vodičů a textů (mazání, rotace, přemísťování, modifikace).

Line: Kreslení vodičů.

Jumper: Umísťování „jumperů“ pro nevodivé křížení vodičů.

Text: Umísťování textů na plochu (nadpisy, nápisy, poznámky…).

Co je nutné vědět před zahájením práce s editorem (aneb nejčastěji se vyskytující chyby začátečníka):

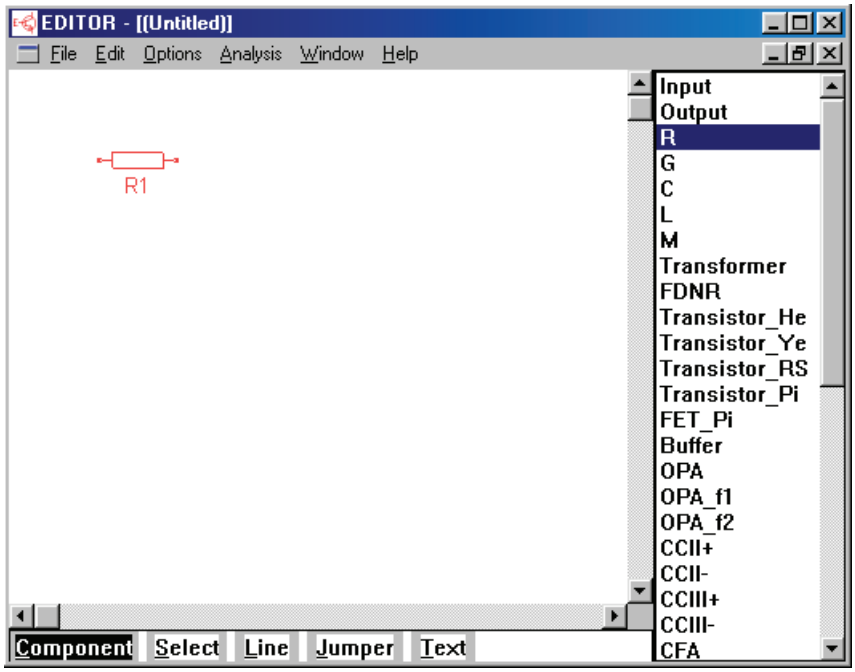
Standardně je editor nastaven v režimu *Select*. Jestliže však kliknete levým tlačítkem myši na libovolnou součástku v sloupci součástek, editor se nastaví do režimu *Component* a je připraven pro pokládání vyznačené součástky na pracovní plochu. K umístění součástky dojde kliknutím levého tlačítka myši kamkoliv na pracovní plochu, a to do místa kliknutí. Z toho plynou **důležité závěry**:

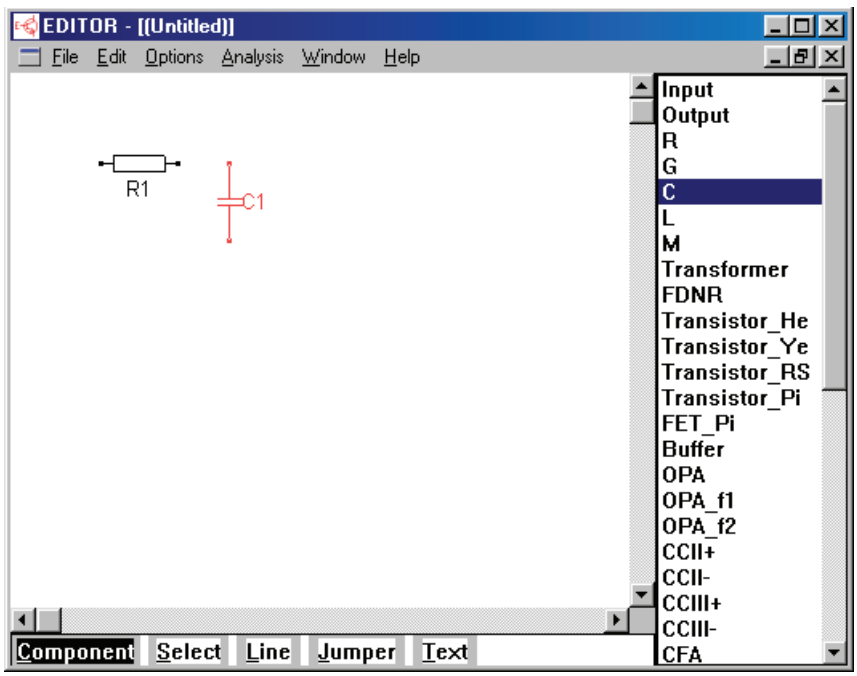
- Při umísťování součástky najeďte kurzorem kamkoliv na pracovní plochu, **zmáčkněte a držte** levé tlačítko myši. Umístěte součástku do místa, kam ji chcete položit, a až pak tlačítko uvolněte.

  Tip: pokud při držení levého tlačítka současně mačkáte pravé, dochází k rotaci schématické značky. Opakujte tak dlouho, dokud není značka v požadované pozici. Jestliže je značka složitější (např. tranzistor), dochází postupně i k jejímu zrcadlení.

- Pokud se vám nepodařilo součástku umístit do správné pozice a chcete ji dodatečně změnit, **neklikejte znovu** na pracovní plochu. Jste totiž v režimu *Component*, takže každé kliknutí znamená opakované umísťování téže vybrané součástky na plochu. Pokud se tak stane, musíme se nejprve přepnout do režimu *Select*. Klikneme-li na součástku v tomto režimu, dojde k jejímu prosvětlení. Nyní máme řadu možností editace:
  - smazání (Del)
  - přesun (držením levého tlačítka myši)
  - rotaci a zrcadlení (držením levého a mačkáním pravého tlačítka myši)
  - změny atributů součástky (dvojím kliknutím).

Zahájení práce s editorem





Klikněte levým tlačítkem myši do lišty součástek na položku R (rezistor). Položka *R* se zvýrazní a editor přejde do režimu *Component* (režim vkládání značek na pracovní plochu, zvýrazní se položka *Component* v spodní liště). Nyní přesuňte kurzor myši kamkoliv do prostoru pracovní plochy a **zmáčkněte a držte** levé tlačítko myši. Na místě kurzoru se objeví schématická značka rezistoru, kterou můžeme pohybovat po ploše. Po nalezení vhodné polohy (předloha viz obr.) uvolníme levé tlačítko, čímž se schématická značka umístí.

POZOR! Pokud nejste s pozicí značky spokojeni, postupujte podle výše uvedených pokynů z části „Co je nutné vědět před zahájením práce s editorem (aneb nejčastěji se vyskytující chyby začátečníka)“.

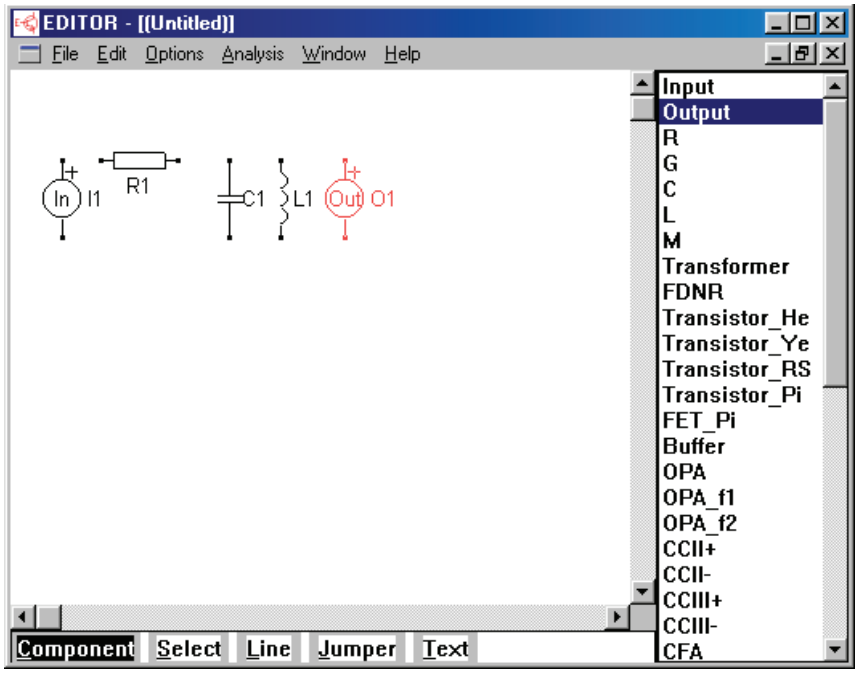
Poznámka: pokud se vám podařilo součástku správně umístit až na několikerý pokus a mezitím jste prováděli její mazání, budete mít pravděpodobně namísto *R*1 součástku označenu vyšším indexem. Vůbec se tím neznepokojujte, nápravu sjednáme na závěr tvorby schématu.

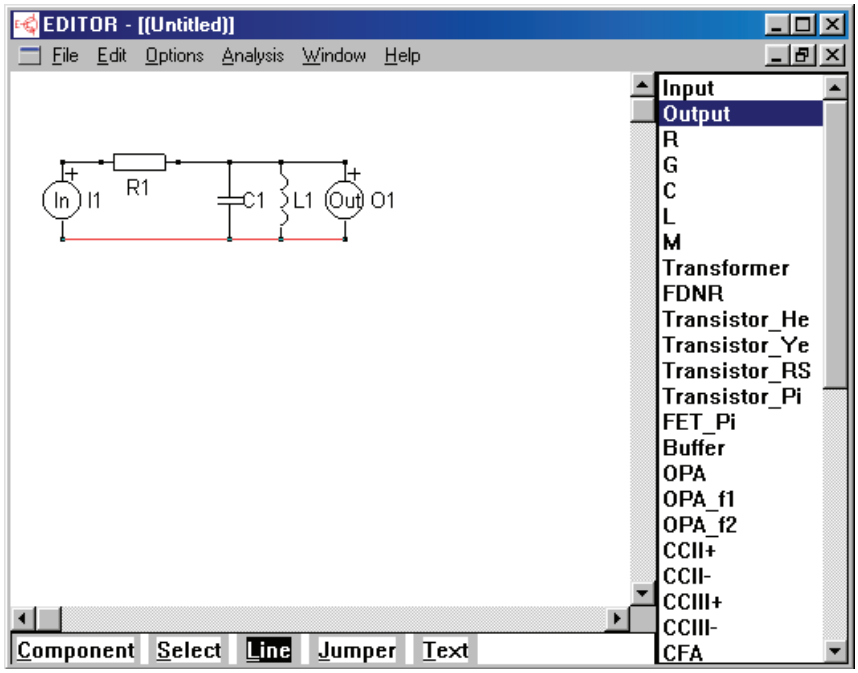
Nyní přidáme kapacitor podle následujícího obrázku. Klikneme na *C* v sloupci součástek, najedeme kurzorem na pracovní plochu, zmáčkneme a držíme levé tlačítko myši. Kapacitor je nutné překlopit do svislé polohy, což zajistíme kliknutím na pravé tlačítko myši. Pak teprve součástku umístíme.

Pokud jste to zvládli, umístěte obdobným způsobem induktor (*L*), vstupní (input) a výstupní (output) dvojpól. Snažte se docílit stavu podle obrázku (pozor na orientaci polarity vstupního a výstupního dvojpólu!).

Nyní můžeme doplnit schéma propojovacími vodiči.

Postup při kreslení vodiče z bodu A do bodu B:





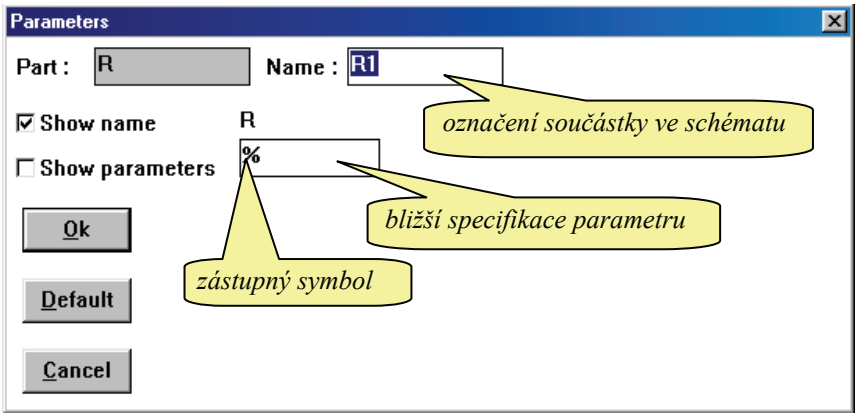
- Editor přepneme do režimu *Line*.
- Kurzor myši přemístíme do bodu *A*.
- Zmáčkneme a držíme levé tlačítko myši.
- Táhnutím přemístíme kurzor do bodu *B*.
- Uvolníme tlačítko myši.

Aplikujte na naše schéma. Výsledek by měl odpovídat následujícímu obrázku.

Tip: Nejsou-li body *A* a *B* na stejné horizontální úrovni, je možné měnit charakter zalomení čáry při jejím „tažení“ přepínat pravým tlačítkem myši.

Poznámka: Hodláte-li nyní editovat položené součástky nebo vodiče, je nutné přepnutí do režimu *Select*.
Všimněte si, že vždy poslední umístěný objekt je vykreslen červenou barvou (je označen). Toto odstraníme v režimu *Select* kliknutím myší kamkoliv na pracovní plochu mimo prostor, kde jsou umístěny objekty.

Parametry součástek



Součástky mají zatím zadány pouze své symbolické parametry *R*1, *L*1 a *C*1. Parametry je možné u každé součástky modifikovat v okně *Parameters*, které se objeví, jestliže v režimu *Select* dvakrát klikneme na tělo součástky. Kliknutím na rezistor *R*1 zpřístupníme okno na obrázku. Význam jednotlivých položek je následující:

Part: Označení typu součástky. *R* znamená rezistor. Tmavé pozadí okénka značí, že jeho obsah není možné editovat.

Name: Označení součástky ve schématu. Toto označení je možné uživatelem změnit.

R .. bližší specifikace parametru, v tomto případě odporu součástky *R*1. **Zástupný symbol** % má následující význam:

Zastupuje výraz v položce **Name**, tj. v našem případě *R*1. Tento výraz je pak použit ve vzorci symbolického výsledku. Chceme-li docílit, aby ve vzorci výsledku namísto *R*1 figurovalo například jen krátké *R*, máme 2 možnosti:

- namísto symbolu % zapíšeme přímo *R*
- zástupný symbol necháme nezměněn, ale přepíšeme obsah položky **Name** z *R*1 na *R*. Tím se ale ve schématu u rezistoru objeví označení *R* namísto *R*1.

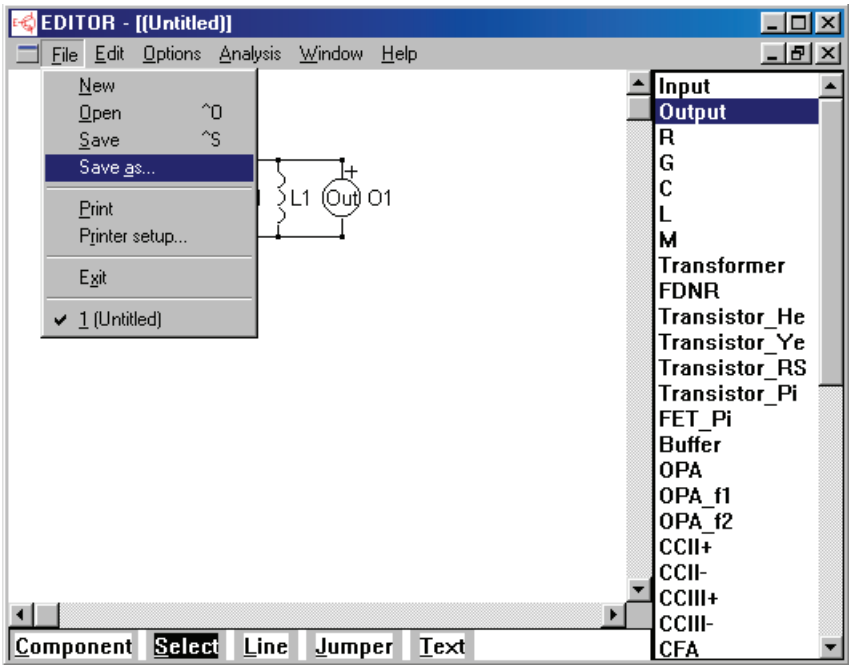
Chceme-li blíže specifikovat parametr součástky, např. zadat i jeho číselnou hodnotu *R*1=1kΩ, můžeme to udělat připsáním %=1k (podrobnosti v dalším textu).

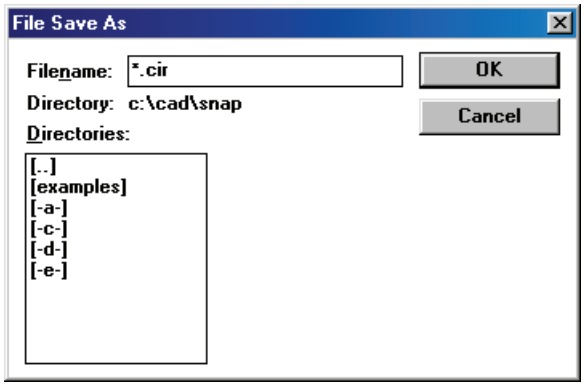
Poznámka: Pokud indexy vašich součástek jsou z jakéhokoliv důvodu jiné než jedničky, změňte je tak, aby vaše schéma odpovídalo hornímu obrázku.

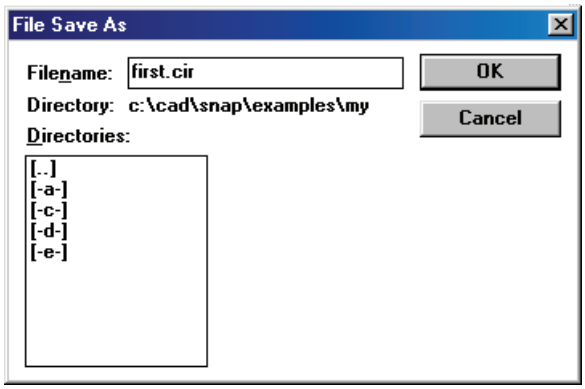
V první fázi nebudeme parametry součástek dále modifikovat.

Ukládání zadání do vstupního souboru

Před analýzou je vhodné uložit vytvořené schéma do souboru. Provedete tak pomocí nabídky *File/Save as*. Na obrázcích je ukázka ukládání do adresáře *examples/my*. Název souboru jsme zvolili *first.cir*.

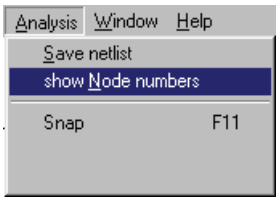






Číslování uzlů, netlist a analýza

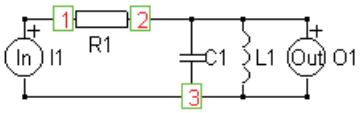
Během vytváření schématu si editor provádí interní číslování uzlů. Výsledek číslování závisí na tom, v jakém pořadí jsme jednotlivé součástky umísťovali na plochu. Čísla uzlů lze zobrazit pomocí nabídky *Analysis/show Node numbers* (viz obr.).



V rozbaleném menu *Analysis* jsme si mohli všimnout položky

Save netlist.

Netlist je výstup editoru ve formě textového souboru *.snn, v němž jsou uloženy informace o elektrických parametrech součástek a jejich vzájemném propojení. Netlist je vstupním souborem pro analyzátor SNAP.



Je-li schéma uloženo v souboru *.cir a spustíme-li analýzu přes menu *Analysis/Snap*, automaticky se vygeneruje netlist *.snn, spustí se Snap a převezme data z netlistu. V tom případě tedy není nutné provádět samostatné ukládání netlistu příkazem *Save netlist*.

Pomocí menu *Analysis/Snap* (resp. stlačením horké klávesy *F*11) tedy spustíme analyzátor Snap, čímž se otevře jeho okno.

Upozornění: Pokud se místo spuštění SNAPu objeví chybové hlášení „*Could not launch application*“, znamená to, že je narušen obsah inicializačního souboru **EDITOR.INI**. V tom případě vyvolejte posloupností *Options/Options* okno *Options* a vyplňte položku *command line* textem **snap.exe %f**. Podrobnosti viz dokumentace k editoru.

Nyní se přesvědčíme, že na disku je vygenerovaný netlist **first.snn**. Najdeme jej pomocí průzkumníka nebo libovolného diskového manažera v stejném adresáři, do něhož jsme uložili soubor **first.cir**. Prohlédneme si jeho obsah:

```
R_R1   1  2 R1
C_C1   2  3 C1
L_L1   2  3 L1
I_I1   1  3
O_O1   2  3
```

Každému řádku odpovídá jedna součástka ze schématu. První symbol na řádku koresponduje s položkou **Part** v okně *Parameters* součástky. Pak následuje spojka _ a za ní obsah položky **Name**. Čísla pak znamenají uzly, mezi nimiž je součástka zapojena. Řádek

končí interpretací položky bližší specifikace parametru, v níž jsme u všech součástek zatím ponechali zástupný symbol %. Znamená to tedy, že se zde objeví kopie položky **Name**.

Vrátíme se do okna Snapu a klikneme do ikony K_v (přenos napětí). V okně výsledků se objeví tento obsah:

```
_____________symbolic_____________
s*( L1 )
-------------------------------------
R1
+s*( L1 )
+s^(2)*( R1*C1*L1 )
```

Tento výsledek lze interpretovat jako vzorec pro přenosovou funkci

$$K_V = \frac{sL_1}{R_1 + sL_1 + s^2 R_1 C_1 L_1}$$

Další typy analýz (semisymbolická a numerická) nejsou k dispozici, protože jsme nezadali číselné hodnoty parametrů všech součástek.

Zadávání číselných hodnot parametrů součástek – zpřístupnění dalších možností analýzy

Následující práce budeme provádět ve schématickém editoru. Na spodní liště nalezneme jeho ikonu a přepneme se do něj. Naším cílem bude nyní zadat numerické hodnoty parametrů $R_1 = 1\ k\Omega$, $C_1 = 10\ n$F, $L_1 = 253\ \mu$H.

Číselný parametr součástky se definuje v položce bližší specifikace parametru v okně **Parameters**. V režimu *Select* dvakrát klikněte do schématické značky rezistoru *R*1. Zástupný symbol % doplňte takto:

%=1k

Pozor! Mezi jednotlivými znaky nesmí být mezery! Další častou chybou je psaní desetinné čárky namísto správné desetinné tečky, například: 1.75n (správně), 1,75n (špatně).

Tím jsme symbolu *R*1, který se skrývá za zástupným symbolem %, přiřadili hodnotu 1*k*. Stejně tak dobře je možné místo 1*k* zapsat například 1000, 1e3 apod.

Poznámka k inženýrské notaci: nerozlišují se velká a malá písmena, takže 10^{-3} je 1*m* stejně jako 1M. Proto 10^6 má speciální označení 1meg (nebo 1MEG). Další anomálií je 10^{-6} jako 1u (1U). Podrobnosti viz nápověda SNAPu a příloha P1).

Doplňte tedy postupně číselné hodnoty pro *R*1=1k (1k), *C*1=10nF (10n) a *L*1=253μH (253u). Pak opět spusťte analýzu (*F11* nebo *Analysis/Snap*). Pokud jste při zadávání udělali chybu, objeví se chybové hlášení „*Error in parameter definition*“ s odkazem na číslo řádku v netlistu, kde k chybě došlo. V tom případě nepostupujte dále, dokud chybu neodstraníte.

Zkuste si nyní prohlédnout netlist **first.snn**, který se mezitím změnil:

```
R_R1   1  2 R1=1k
C_C1   2  3 C1=10n
L_L1   2  3 L1=253u
I_I1   1  3
O_O1   2  3
```

Je doplněn o číselné parametry součástek.

Po aktivaci výpočtu K_V v Snapu dostaneme kromě symbolické analýzy i analýzu semisymbolickou a uvolní se další funkce včetně analýzy kmitočtových charakteristik a časových průběhů (viz Lekce 1 až Lekce 3).

Některé další možnosti bližší specifikace parametrů součástek

Vrátíme se do editoru. Změňme zadání symbolických parametrů tak, aby ve vzorci přenosové funkce namísto symbolů *R*1, *C*1 a *L*1 figurovaly pouze zkrácené symboly *R*, *L* a *C*. V režimu *Select* dvakrát klikněte na značku rezistoru a původní obsah položky

%=1k

přepišme na

R=1*k*

Původní zástupný symbol % představuje položku *R*1 z okénka **Name**. Nyní bude symbolický parametr rezistoru R1 přímo R.

Obdobně změňte zástupné symboly *C*1 a *L*1 na *C* a *L*. Přesvědčte se, že výsledek symbolické analýzy ve Snapu je nyní

```
________symbolic____________________
s*( L )
-------------------------------------
R
+s*( L )
+s^(2)*( R*C*L )
```

Další výsledky analýzy jsou pochopitelně nezměněny.

Vraťte se do editoru. Zkuste nyní modifikovat současný popis induktoru

L=253u

na

253u.

Prohlédněte si nyní výsledky symbolické analýzy

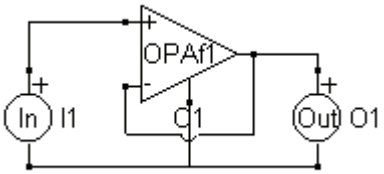
```
_____symbolic_____________________
s*( 0.000253 )
-------------------------------------
R
+s*( 0.000253 )
+s^(2)*( 0.000253*R*C )
```

a pokuste se porozumět principu. K čemu je možno tohoto postupu využít?

S dalšími možnostmi definování parametrů se seznámíme v příkladu 2. Detaily viz nápověda Snapu.

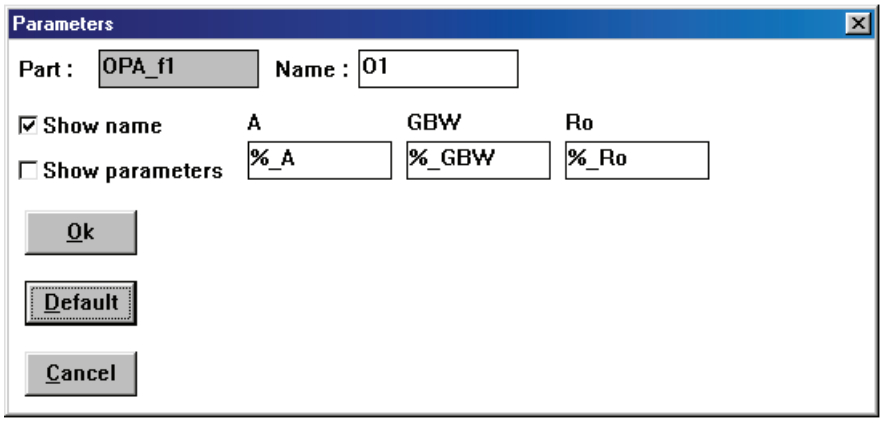
6.3.2 Obvod se součástkami, které jsou popsány několika parametry

Doporučujeme nejprve prostudovat úvod k části 6.2.3 Lekce 3. Předpokladem k práci na tomto příkladu je i průchod předchozím příkladem v části 6.3.1.



Nakreslete schéma napěťového sledovače podle obrázku (viz též ukázkový příklad ***demopa1.cir***). Aktivní součástkou „OPAf1“ je jednopólový model operačního zesilovače, popsaný stejnosměrným zesílením otevřené smyčky A, tranzitním kmitočtem GBW a výstupním odporem R_o. Těmto parametrům přiřaďte hodnoty typické pro operační zesilovač typu 741: A = 200000, GBW = 1 MHz, R_o = 50 Ω. Dále se pokuste získat přenos napětí pro ideální operační zesilovač, tj. pro $A = \infty$, $GBW = \infty$, $R_o = 0$.

Nejprve nakreslete schéma podle zásad, vysvětlených v příkladu 1. Poté v režimu *Select* dvakrát klikneme na značku operačního zesilovače a prohlédneme si okno parametrů.



Jméno operačního zesilovače je O1. Zápisy %_A, %_GBW a %_Ro znamenají označení parametrů A, GBW a R_o zesilovače O1. V případě více zesilovačů stejného typu v obvodu je tak zajištěna jednoznačná identifikace jejich parametrů.

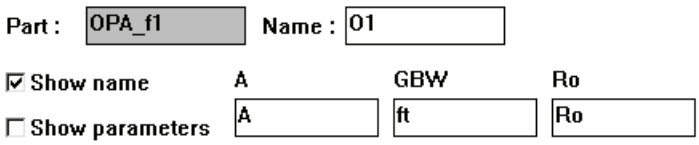
Analýza přenosu napětí Snapem dá pouze symbolický výsledek

```
_____________symbolic_____________________
6.28319*O1_A*O1_GBW
-------------------------------------
6.28319*O1_A*O1_GBW +6.28319*O1_GBW
+s*( O1_A )
```

který odpovídá vzorci

$$K_V = \frac{2\pi.A.GBW}{2\pi.A.GBW + 2\pi.GBW + sA}.$$

Dlouhé symboly O1_A, O1_GBW a O1_Ro mohou činit symbolické výsledky nepřehlednými. Navíc v případě jediného zesilovače ve schématu je takovýto zápis parametrů zbytečně složitý. Změňte označení parametrů například tak, jak je naznačeno na obrázku.



Namísto dlouhého výrazu GBW je použit kratší ft (tranzitní kmitočet). Pak spusťte analýzu. Symbolický výsledek nyní bude přehlednější:

```
_____________symbolic_____________________
6.28319*A*ft
-----------------------------------
6.28319*A*ft +6.28319*ft
+s*( A )
```

Nyní přiřadíme symbolickým parametrům číselné hodnoty:

☑ Show name	A	GBW	Ro
☐ Show parameters	A=200k	ft=1meg	Ro=50

Výsledky symbolické analýzy se pochopitelně nezmění, přibudou však výsledky semisymbolické analýzy, nulové body a póly a vzorce impulsní a přechodné charakteristiky.

```
_____________semisymbolic_________________
Multip. Coefficient =  6.28318530000000E+0006
 1.00000000000000E+0000
-----------------------------------
 6.28321671592650E+0006
 1.00000000000000E+0000 * s
_________________zeros_____________________
none
_________________poles_____________________
-6.28321671592650E+0006
____________step response_________________
 9.99995000025000E-0001
-9.99995000025000E-0001*exp(-6.28321671592650E+0006*t)
____________pulse response_________________
 6.28318530000000E+0006*exp(-6.28321671592650E+0006*t)
```

V složkách  a  je pak možné pracovat s grafy v kmitočtové a časové oblasti.

Pokusme se nyní idealizovat model operačního zesilovače volbou hraničních parametrů $A=\infty, f_t=\infty, R_o=0$. Máme na výběr 3 možnosti jak to udělat:

1. Symbolické parametry ponecháme beze změny a přiřadíme jim pouze numerické parametry. Pak symbolický výsledek bude beze změny, ke změně dojde počínaje semisymbolickou analýzou.
2. Některé symbolické parametry ponecháme beze změny, ostatní přepíšeme jejich numerickými hodnotami. Tím dosáhneme částečné modifikace (zjednodušení) symbolického výrazu.
3. Místo všech symbolických parametrů zapíšeme přímo numerické parametry. Pak se změny promítnou maximální měrou již do symbolických výsledků. Tím dosáhneme maximální úpravy symbolického vzorce.

První možnost: definování numerických hodnot k symbolickým parametrům:

☑ Show name	A	GBW	Ro
☐ Show parameters	A=inf	ft=inf	Ro=0

Všimněte si že program umožňuje pracovat i se symbolem nekonečna (inf). Po analýze se objeví následující výsledky:

```
_____________symbolic_____________
6.28319*A*ft
-------------------------------------
6.28319*A*ft +6.28319*ft
+s*( A )
_____________semisymbolic_____________
 1.00000000000000E+0000
-------------------------------------
 1.00000000000000E+0000
_____________zeros_____________
none
_____________poles_____________
none
_____________step response_____________
 1.00000000000000E+0000
_____________pulse response_____________
+ 1.00000000000000E+0000*Dirac(0)
```

Výsledky semisymbolické analýzy ukazují, že v případě ideálního operačního zesilovače se obvod chová jako ideální sledovač napětí s přenosem 1.

Druhá možnost: některé parametry jsou definovány jen numericky:

☑ Show name	A	GBW	Ro
☐ Show parameters	inf	ft=inf	0

Symbolický výraz bude nyní popisovat vliv f_t na přenos za předpokladu nekonečného zesílení A a nulového výstupního odporu R_o:

```
_____________symbolic_____________
6.28319*ft
-------------------------------------
6.28319*ft
+s*( 1 )
```

neboli

$$K_V = \frac{2\pi f_t}{2\pi f_t + s}.$$

Třetí možnost: všechny parametry jsou definovány jen numericky:

☑ Show name	A	GBW	Ro
☐ Show parameters	inf	inf	0

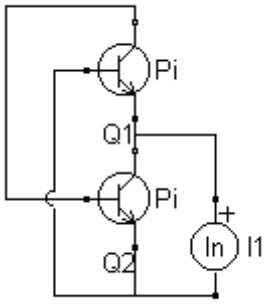
Pak symbolický výsledek dává přímo hodnotu 1.

6.3.3 Obvod s několika součástkami stejného typu

Nakreslete schéma obvodu podle obrázku (viz též vzorový příklad **demindtr.cir**). Podle článku

SUNG-Gi Yan et al.: Fully Symmetrical, Differential- Pair Type Floating Active Inductors. ISCAS'97 HongKong 1997, Vol.1, pp. 93-96

se obvod chová vzhledem k vstupním svorkám jako paralelní uspořádání rezistoru, kapacitoru a induktoru.



Tranzistory jsou typu „Transistor_Pi“. Jejich parametry zadejte následovně:

Name : Q1			Name : Q2		
rbe	gm	gce	rbe	gm	gce
inf	gm1=0.1	0	inf	gm2=0.1	0
Cbe	Cbc	Cce	Cbe	Cbc	Cce
Cbe1=5p	0	0	Cbe2=5p	0	0

Fyzikální význam jednotlivých parametrů (podrobnosti viz přílohy P1.4 a P4 skript [1.23]):

rbe .. střídavý odpor báze-emitor,
gm .. transkonduktance I_c/U_{be} (někdy nazývána strmostí),
gce .. střídavá vodivost kolektor-emitor,
Cbe .. kapacita báze-emitor,
Cbc .. kapacita báze-kolektor,
Cce .. kapacita kolektor-emitor.

Analýza tedy bude provedena za zjednodušujícího předpokladu, že oba tranzistory budou mít nekonečný odpor *rbe*, nulovou vodivost *gce* a nulové kapacity *Cbc* a *Cce*. Zbývající parametry *gm* a *Cbe* mají sice u obou tranzistorů stejné velikosti, v symbolech jsou však odlišeny indexy, abychom mohli v symbolickém výsledku identifikovat jejich vlivy.

Postup řešení: spočítáme vstupní impedanci a zjistíme, zda je ji možno interpretovat jako paralelní spojení součástek typu *R*, *C* a *L*.

Poznámka: Počítáme-li jen vstupní impedanci, pak do obvodu není nutné kreslit součástku „*Out*“ označující výstupní bránu. V SNAPu je však možné analyzovat dva druhy vstupní impedance: s výstupem naprázdno nebo nakrátko. Pokud ve schématu nemáme vyznačenu výstupní bránu, můžeme analyzovat pouze vstupní impedanci při výstupu naprázdno. Pokus o analýzu obvodových funkcí, které vyžadují definování výstupní brány, pak povede k chybovému hlášení

„*The function is not available because INPUT or OUTPUT was not defined*.“.

Provedeme analýzu vstupní impedance (*Zin, open*). Prohlédneme si první část výsledků:

```
_____________symbolic_____________________
s*( Cbe2 )
-------------------------------------
gm1*gm2
+s*( gm1*Cbe2 )
+s^(2)*( Cbe1*Cbe2 )
_____________semisymbolic_________________
Multip. Coefficient =  2.00000000000000E+0011
 1.00000000000000E+0000 * s
-------------------------------------
 4.00000000000000E+0020
 2.00000000000000E+0010 * s
 1.00000000000000E+0000 * s^(2)
```

kterou lze interpretovat vzorcem

$$Z_{in} = \frac{sC_{be2}}{g_{m1}g_{m2} + sg_{m1}C_{be2} + s^2 C_{be1}C_{be2}} = 2\text{x}10^{11}\,\frac{s}{4\text{x}10^{20} + 2\text{x}10^{10}s + s^2}\,.$$

Vstupní admitance tedy bude

$$Y_{in} = \frac{1}{Z_{in}} = \frac{g_{m1}g_{m2} + sg_{m1}C_{be2} + s^2 C_{be1}C_{be2}}{sC_{be2}} = \frac{g_{m1}g_{m2}}{sC_{be2}} + g_{m1} + sC_{be1} = \frac{1}{500\text{x}10^{-12}s} + 0.1 + 5\text{x}10^{-12}s$$

a lze ji interpretovat jako paralelní řazení induktoru o indukčnosti

$$L = \frac{C_{be2}}{g_{m1}g_{m2}} = 0.5nH$$

rezistoru o odporu

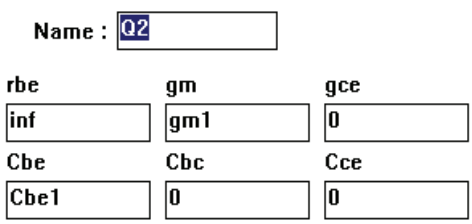
$$R = \frac{1}{g_{m1}} = 10\Omega$$

a kapacitoru o kapacitě

$$C = C_{be1} = 5pF\,.$$

Modifikujeme-li parametry tranzistoru *Q2* podle obrázku, dosáhneme shody odpovídajících symbolických parametrů. Symbolický výraz pro vstupní impedanci nyní bude

```
_____________symbolic_____________________
s*( Cbe1 )
-------------------------------------
gm1^(2)
+s*( gm1*Cbe1 )
+s^(2)*( Cbe1^(2) )
```



Výhodu takového zjednodušování oceníme například v režimu krokování: Krokováním parametru *gm1* současně krokujeme stejné parametry obou tranzistorů.

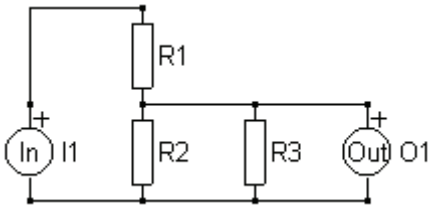
6.3.4 Vazby mezi parametry různých součástek

V předchozím příkladu jsme se seznámili s nejjednodušším typem takovýchto vazeb: parametry *gm* a *Cbe* tranzistoru *Q*2 byly stejné jako parametry *gm* a *Cbe* tranzistoru *Q*1. Obecně je možná vazba mezi parametry *y* a *x* součástek v tomto tvaru:

$$y = ax/c,$$

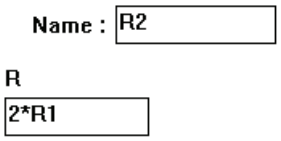
kde *a* a *c* jsou libovolné reálné konstanty.

Nakresleme schéma zatíženého děliče napětí podle obrázku. Standardně jsou parametry součástek pouze symbolické a jsou definovány zástupnými symboly %. Pak přenos napětí vyjde




```
_____________symbolic_____________________
R2*R3
-------------------------------------
R2*R3 +R1*R3 +R1*R2
```

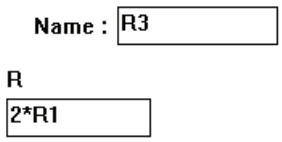
Víme-li, že ve skutečnosti *R*2 je například vždy dvakrát větší než *R*1, pak parametr *R*2 zapíšeme takto:



Výsledky symbolické analýzy se příslušně změní:

```
_____________symbolic_____________________
2*R3
-------------------------------------
3*R3 +2*R1
```

Položíme-li navíc i podmínku R3=2*R1, změní se přenos na konstantu 1/2:



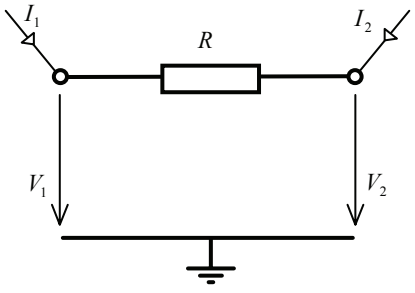

```
_____________symbolic_____________________
0.5
-------------------------------------
1
```

6.4 Princip tvorby modelů prvků SNAPu na základě modifikované metody uzlových napětí

Matematické modely obvodových prvků jsou v programu SNAP založeny na modifikované metodě uzlových napětí. Modely jsou uloženy v přídavném a snadno editovatelném textovém souboru **SNAP.CDL**.

Například model rezistoru je následující:

```
[R]   <-------  název modelu
2     <-------  počet uzlů
1 1   <-------  rozměr matice parametrů
0     <-------  počet přídavných obvodových veličin
MAT   <-------  začátek definice obvodové matice
1 1 2 2   1 1 1 0 -1  <-------  definice obvodové matice
```



$$\begin{bmatrix} I_1 \\ I_2 \end{bmatrix} = \begin{bmatrix} 1/R & -1/R \\ -1/R & 1/R \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \end{bmatrix}$$

Definice má tuto obecnou strukturu:

- [název modelu]
- počet uzlů, jimiž je součástka spojena s okolím,
- rozměr matice parametrů *r* součástky (počet řádků počet sloupců); u rezistoru je jediný parametr odpor,
- počet přídavných proměnných, nutných k modelování součástky modifikovanou metodou uzlových napětí (MMUN),
- MAT – klíčové slovo, za nímž následuje definice prvků matice modifikované metody uzlových napětí,
- definice obvodové matice.

Prvky matice jsou definovány pomocí speciální konvence – tzv. „*atom definice*“. Obecná struktura:

a b c d k i j n m

Poslední čtyři proměnné

k i j n m

definují *prvek matice* – tzv. *atom* podle vzorce

$$atom = (k*r[i,j]*s\hat{}(n))\hat{}m,$$

kde

r[*i*,*j*] .. parametr součástky v *i*-tém řádku a *j*-tém sloupci matice parametrů *r*,
s .. Laplaceův operátor,
n,*m* .. celá čísla.

Například pro zapsání susceptance induktoru 1/(*sL*) by bylo zapotřebí definovat

k i j n m ……. 1 1 1 1 –1

za předpokladu, že induktor má jediný parametr – indukčnost, uložený v jednoprvkové matici parametrů *r*(1x1).

První čtyři proměnné

a b c d

definují polohu „*atomu*“ v obvodové matici, a to podle následující konvence:

„Atom“ se objeví v průsečících

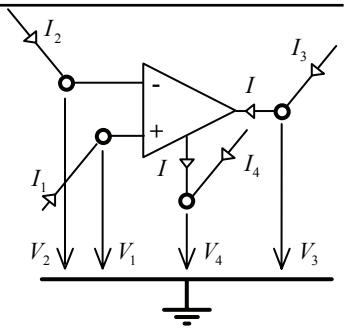
a – tého řádku a *b*-tého sloupce se znaménkem +
c – tého řádku a *d*-tého sloupce se znaménkem +
a – tého řádku a *d*-tého sloupce se znaménkem –
c – tého řádku a *b*-tého sloupce se znaménkem –

za předpokladu, že ani jedno z dvojice čísel řádku a sloupce není nula. V opačném případě se „*atom*“ na žádnou pozici nezapisuje.

Následující příklad ukazuje strukturu modelu ideálního operačního zesilovače. V modelu je kromě uzlových napětí použita 1 pomocná obvodová veličina – výstupní proud zesilovače *I*. Ideální operační zesilovač nevyžaduje žádný obvodový parametr.

```
[OPA]
4
0 0
1
MAT
-1 1 0 2   1 0 0 0 1
3 -1 4 0   1 0 0 0 1
```

$$\begin{array}{|c|} \hline I_1 \\ \hline I_2 \\ \hline I_3 \\ \hline I_4 \\ \hline 0 \\ \hline \end{array} = \begin{array}{|c|c|c|c|c|} \hline & & & & \\ \hline & & & & \\ \hline & & & & 1 \\ \hline & & & & -1 \\ \hline 1 & -1 & & & \\ \hline \end{array} \begin{array}{|c|} \hline V_1 \\ \hline V_2 \\ \hline V_3 \\ \hline V_4 \\ \hline I \\ \hline \end{array}$$



Znaménko – u čísel řádků (sloupců) matice v příkazu MAT značí symbol přídavného řádku (sloupce). Například –1 značí přidaný řádek, resp. sloupec č.1.

Doporučujeme nahlédnout do souboru **SNAP.CDL** a seznámit se se způsobem definice modelů nejrůznějších prvků. Ve všech případech jde o praktickou aplikaci metody „razítek" MMUN. Po zvládnutí této techniky budete moci psát modely vlastních součástek.

6.5 Shrnutí kapitoly 6

- Počítačová analýza elektrického obvodu je realizována ve dvou na sebe navazujících krocích. Nejprve se vytvoří model obvodu a pak proběhnou požadované výpočty nad tímto modelem.
- Model obvodu se dnes nejčastěji vytváří pomocí schématického editoru. Alternativou je tvorba textového souboru - netlistu, do něhož se smluveným způsobem vloží informace o jednotlivých součástkách obvodu a jejich vzájemném propojení. Netlist, resp. soubor, jehož je netlist součástí, je generován rovněž schématickými editory jako zdroj vstupních dat simulátoru. Vytvořený model obvodu – schéma – lze uložit na disk do tzv. *vstupního souboru*, angl. „*circuit file*".
- Schématický editor ke své funkci potřebuje knihovnu schématických značek jednotlivých součástek. Pro účely jejího dalšího rozšiřování a pohodlné modifikace uživatelem je potřebné, aby tato knihovna existovala ve formě samostatného textového souboru.
- Všechny tři soubory, tj. vstupní soubor, knihovna značek i netlist mají formát, který je specifický pro konkrétní schématický editor. Tyto soubory tudíž nejsou jednoduše přenositelné mezi různými programy. Existuje pouze snaha, aby netlisty vykazovaly jednotné rysy formátu, používaného v programech z rodiny SPICE.
- Schématický editor umožňuje práci v několika režimech. V této kapitole jsme se seznámili mj. se základními režimy „*Select*", „*Component*" a „*Wire*", které se objevují i u jiných typů editorů. V navazujících kapitolách 9 a 10 zjistíme, že schématické editory profesionálních simulátorů nabízejí řadu dalších režimů. Každý editor se bohužel liší i ve filozofii tvorby schématu, v ovládání, ve způsobu pokládání značek součástek na plochu a kreslení vodičů. Způsob zadávání parametrů součástek souvisí s podobou knihoven součástek, které jsou důležitou „výbavou" každého simulátoru. Zde jsou rovněž velké rozdíly mezi jednotlivými programy.
- Dále je třeba říci něco o vazbách mezi první a druhou etapou počítačové simulace, tj. etapou přípravy dat k simulaci a navazující etapou realizace vlastní simulace. Schématický editor může být samostatným programem, který předává data dalšímu programu – vlastnímu simulátoru – prostřednictvím netlistu. Tato tzv. modulární koncepce je uplatněna – kromě SNAPu –mj. i u profesionálních simulátorů rodiny

SPICE. Druhou možností je sloučit schématický editor s výkonnou výpočetní částí do jednoho celku. To s sebou přináší některé výhody, např. zpětné promítání výsledků simulací přímo do schématu (vizualizace uzlových napětí a další efekty). Toto řešení je uplatněno například u simulátorů MicroCap a TINA.

- Různé analogové simulátory jsou schopné realizovat řadu typů analýz. Zatím jsme se seznámili mj. s kmitočtovou analýzou, tj. s analýzou kmitočtových závislostí sledovaných parametrů obvodů (program simuluje funkci obvodového analyzátoru kmitočtových charakteristik), a s časovou analýzou, tj. analýzou časových průběhů (simulujeme činnost osciloskopu). V další kapitole toto spektrum analyzačních možností podstatně rozšíříme o možnosti výkonných simulátorů pracujících na numerickém principu.
- Dále jsme se naučili používat tzv. vícenásobnou analýzu neboli krokování parametrů. Tento analyzační režim bylo možné použít jak u kmitočtové, tak i u časové analýzy. Toto je možné zobecnit a konstatovat, že v rámci daného typu analýzy můžeme simulátor využívat v několika speciálních režimech. Z dalších možných režimů můžeme zmínit např. režim optimalizační.
- Kromě klasických typů analýz, k nimž – jak uvidíme dále – patří u výkonných numerických simulátorů analýzy Transient (časová), AC (střídavá, kmitočtová) a DC (stejnosměrná, tj. analýza stejnosměrných, např. ampérvoltových charakteristik), nabízejí jednotlivé programy široké možnosti specifických typů analýz. V kapitole 10.6 si ukážeme některé ze zajímavých typů, začleněné do MicroCapu VII.
- V této kapitole jsme uskutečnili první praktické krůčky k počítačové simulaci složitých obvodů. Simulátory využívající symbolické algoritmy jsou však určeny pouze k analýze lineárních a linearizovaných obvodů. Nyní nás čeká seznámení se simulátory, které jsou založeny na numerických algoritmech, a kde neexistuje principiální omezení na typ analogových součástek ani na režim, v němž se má analyzovaný obvod nacházet. Typů různých analýz i analyzačních režimů zde bude daleko více. Značná složitost modelů jednotlivých součástek si vynutila speciální způsoby jejich zadávání při tvorbě modelu obvodu, což se promítá do pro nás nových postupů již při práci se schématickým editorem.
- V neposlední řadě rostou při používání výkonných simulátorů požadavky na znalosti uživatele z oblasti teorie elektrických obvodů a součástek. Stává se, že obsluha simulátoru často nemá tušení, zdali má nebo nemá ponechat v příslušném menu například zatrženou položku „Operating Point“ (pracovní bod). Někdy prostě stačí málo, a drahý a výkonný program vygeneruje chybné výsledky i při analýze relativně jednoduchých obvodů. Asi nejhorší případ nastane, pokud si toho uživatel programu nevšimne. Proto je na místě vždy, ale u profesionálních simulátorů zvlášť, držet se osvědčeného „důvěřuj, ale prověřuj“. A zde je vhodné dodat: Prověřuj se znalostí věci.

7 Struktura simulačního programu, založeného na numerických algoritmech (MicroCap)

Cíle kapitoly:

- Na konkrétním příkladu programu MicroCap 7 objasnit strukturu simulačního programu, založeného na numerických algoritmech výpočtů.
- Vysvětlit význam základních bloků programu a způsoby předávání dat mezi bloky navzájem a mezi programem a uživatelem.
- Objasnit strukturu dat, uchovávajících informace o modelech součástek.
- Podat informaci o typech knihoven a souborů pro uchovávání a předávání dat.

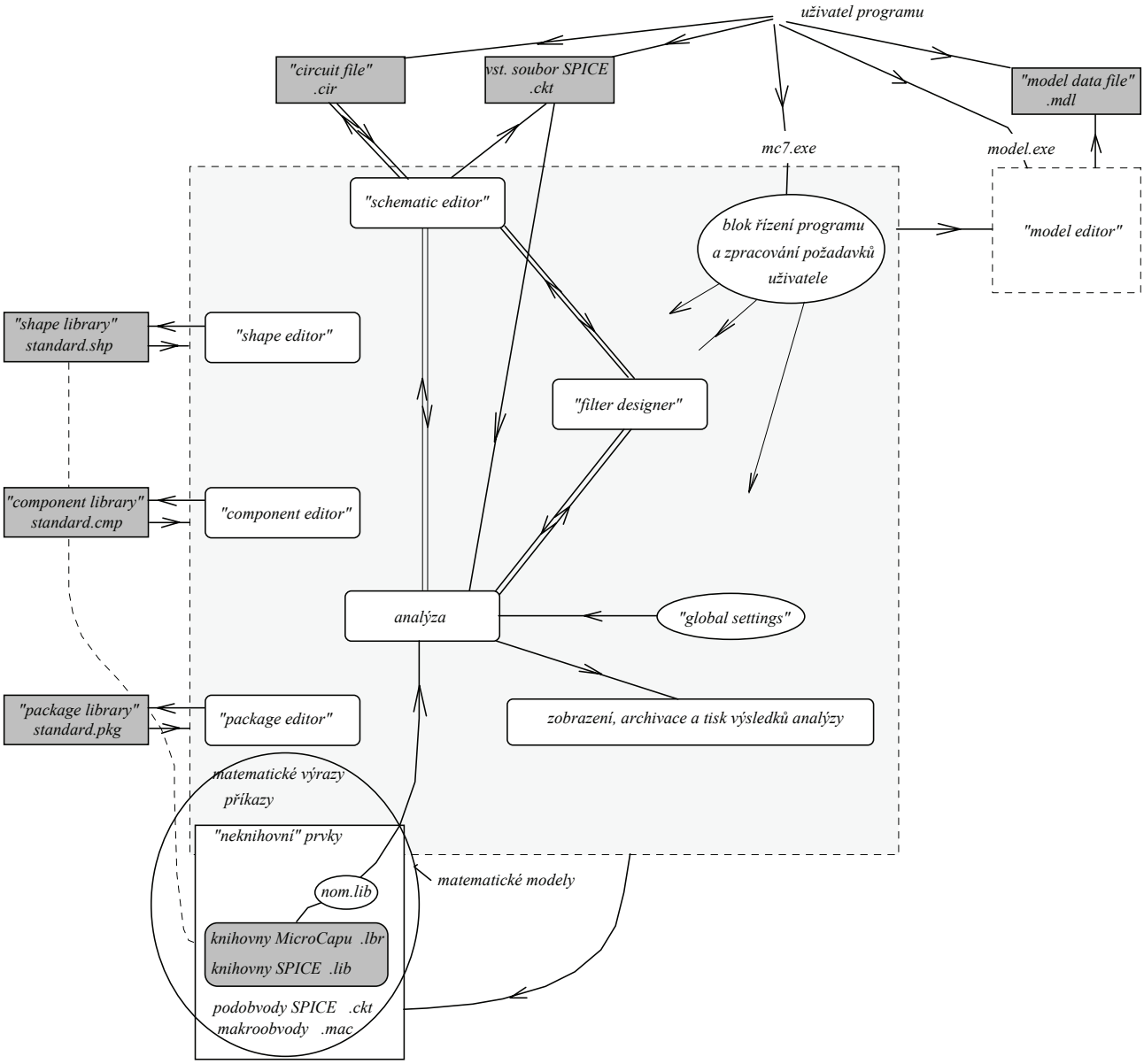
Některým snadno dostupným simulačním programům z této kategorie je věnována samostatná kniha [3.1]. Nebudeme zde proto opakovat fakta v ní podrobně vyložená. Pokusíme se navázat na předchozí části 5 a 6, v nichž jsme vyložili způsoby práce se „symbolickými“ programy, a principy zde osvojené použijeme jako základ pro zvládnutí práce s poněkud složitějšími simulátory „numerickými“. Styl výkladu rovněž pozměníme: Po praktickém zvládnutí SNAPu již bude snazší rychleji se „zapracovat“ do používání schématického editoru Microcapu, takže se více soustředíme na podstatné rozdíly a nové možnosti programu Microcap a na jeho analyzační schopnosti.

Na **obr. 7.1** je zjednodušené blokové schéma programu MicroCap 7 jako reprezentanta „numerických“ simulátorů. Všechny činnosti, známé ze SNAPu, a řadu dalších zde obstarává jediný spustitelný soubor **MC7.EXE**. Dalším programem je **MODEL.EXE**, který se dá spustit buď samostatně, nebo i z menu MicroCapu. Tento program slouží k vytváření knihoven – matematických modelů součástek na základě jejich nejrůznějších charakteristik, získaných reálným měřením.

Běžný uživatel využívá z MicroCapu bloky „*schematic editor*“ – schématický editor, analýza, zobrazení, archivace a tisk výsledků. Kromě toho je do programu zakomponováno několik dalších relativně samostatných programů. „*Filter designer*“ je nepříliš zdařilý program pro návrh kmitočtových filtrů. „*Shape editor*“ je program pro editaci stávajících a vytváření dalších schématických značek. „*Component editor*“ obhospodařuje databázi součástek a dává do souvislosti jejich schématické značky s jejich matematickými modely, které jsou začleněny buď přímo do simulátoru (tzv. „neknihovní“ prvky), nebo jsou uloženy v externích knihovnách. „*Package editor*“ je program, umožňující pracovat s knihovnou patic jednotlivých součástek a generovat netlisty simulovaných obvodů pro programy na návrh plošných spojů, konkrétně pro *Accel*, *OrCad*, *Protel* a *Pads*.

Jednotlivé části programu jsou natolik provázány, že je někdy obtížné vymezit jejich hranice. Například schématický editor je provázán s blokem analýzy, který zpět do editoru vysílá data. Toto interaktivní pojetí analýzy pak umožňuje například zobrazení rozložení napětí, proudů, výkonů a dalších atributů jednotlivých součástek a sledování jejich bezprostředních změn při změnách v zapojení.

Schématický editor umožňuje práci v řadě „nových“ režimů. Při kreslení schématu vybíráme jednotlivé součástky z hierarchicky uspořádané databáze součástek, která je uložena v „*component library*“ – knihovně součástek. Na obrazovce se pak objeví příslušná schématická značka z „*shape library*“ – knihovny značek. Příslušné knihovny jsou poměrně rozsáhlé soubory s názvy **standard.cmp** a **standard.shp**. Je možné je modifikovat nebo přidávat další.



Obr. 7.1: Zjednodušené blokové schéma programu MicroCap VII.

Aby nedošlo k nedorozumění, je třeba rozlišovat mezi právě zmíněnými knihovnami a knihovnami, v nichž jsou uložena data pro matematické modelování. Jde zejména o rozsáhlé knihovny polovodičových i jiných součástek s příponami **.lbr** a **.lib**. V binárních souborech ***.lbr** jsou uloženy „MicroCapovské“ modely součástek, jejichž struktura je popsána v dokumentaci k programu. Soubory ***.lib** jsou snadno editovatelné ASCII soubory se SPICE modely součástek. Knihovny MicroCapu lze čas od času aktualizovat z www stránky programu http://www.spectrum-soft.com/. Rovněž je lze modifikovat z prostředí MicroCapu. Knihovny SPICE jsou v tomto směru ještě výhodnější. Lze je neomezeně rozšiřovat nebo přidávat nové knihovny pomocí textových souborů, které jsou většinou volně k dispozici na Internetu na stránkách výrobců integrovaných obvodů.

Pro který z modelů se rozhodnout, pro „MicroCapovský“ nebo SPICE, mám-li k dispozici oba? Já osobně dávám přednost modelům SPICE, tedy modelům „od výrobce“. Jsou však výjimky. Občas se stane, že SPICE model se „nepovede“ a dává nekorektní výsledky při některém z typů analýz, např. časové (tranzientní) analýze, zatímco při použití

k analýze kmitočtových charakteristik je vynikající. Modely „MicroCapovské“ bývají jednodušší a vedou k rychlejší analýze. Toto však rovněž nelze zevšeobecnit. Univerzální odpověď na výše položenou otázku zřejmě neexistuje.

Knihovny součástek jsou značně rozsáhlé a jsou na disku uloženy v mnoha desítkách souborů. Mnohdy je zbytečné, abychom pracovali se všemi knihovnami. Proto existuje speciální textový soubor **nom.lib**. Je to jakýsi filtr mezi všemi knihovnami na disku a těmi knihovnami, které hodláme využívat. V souboru jsou uvedeny názvy těch knihoven, které budou po spuštění simulátoru k dispozici. Po vytvoření vlastní knihovny bychom tedy neměli opomenout její název přidat do souboru **nom.lib**. Během spuštění Microcapu se načítá obsah tohoto souboru a v případě zjištění změny se modifikuje tzv.“*index file*“, který slouží programu k rychlému přístupu do všech aktuálních knihoven.

V seznamu modelů, které lze v prostředí schématického editoru používat, jsou dále tzv. *podobvody SPICE* a *makroobvody*. V obou případech se jedná o modely relativně samostatných obvodů, které jsou uloženy na disku a jsou využívány pro tvorbu jiných, složitějších obvodů. Podobvody SPICE mají své modely uloženy v ASCII souborech napsaných v jazyku SPICE, zatímco makroobvody jsou vytvořeny pomocí schématického editoru a uloženy do vstupních souborů („*circuit files*“) s příponou **.mac**. Makroobvody mohou (ale nemusí) mít definovánu sadu parametrů, které se konkretizují při použití daného makra v obvodu. Podobvody SPICE mohou být z principu rovněž volány s parametry, této možnosti se však v rámci MicroCapu nevyužívá. Makroobvody a podobvody SPICE jsou důležitým nástrojem k neomezenému rozšiřování analyzačních možností programu o modely dalších elektrických prvků. Zatímco makroobvody jsou „lokální“ záležitostí MicroCapu, podobvody SPICE tvoří důležitý nástroj modelování v simulátorech rodiny SPICE. Knihovny SPICE se v podstatě skládají z modelů jednotlivých podobvodů. Model podobvodu začíná příkazem *.subckt* a končí řetězcem *.ends*. Většina světových výrobců polovodičových součástek poskytuje modely svých výrobků právě ve formě podobvodů.

Model obvodu můžeme – kromě klasického výběru součástek s jejich automatickým napojením na jejich matematické modely v knihovnách – dále obohacovat o *matematické výrazy* (například lze zapsat odpor rezistoru jako prakticky libovolnou funkci obvodových napětí nebo proudů) a *příkazy*, které se umísťují buď přímo na pracovní plochu schématického editoru nebo do speciálního textového pole.

Z **obr. 7.1** je zřejmé, že existují součástky, jejichž modely nejsou uloženy v žádných knihovnách. Jedná se například o pasivní součástky *R*, *L* a *C* a další součástky s jedním nebo jen několika parametry a tudíž s velmi jednoduchým modelováním. Později však poznáme, že i těmto součástkám lze přidělit modely, které pak umožní rozšířit simulační možnosti například o toleranční nebo teplotní analýzu.

Výsledek zadávání modelu obvodu je možné podobně jako u SNAPu uložit do *vstupního souboru*. Tento soubor je zde textový a obsahuje řadu informací o obvodu, jakož i o posledním stavu programu při jeho analýze. V zájmu dosažení lepší přenositelnosti souborů mezi uživateli, kteří mohou mít k dispozici různé knihovny součástek a jejich schématických značek, bylo dokonce přistoupeno k následujícímu opatření. Vstupní soubor obsahuje kopie schématických značek použitých součástek, jakož i příslušná data z „*Component library*“. Domníváme-li se, že příjemce souboru nebude mít k dispozici ani matematický model součástky, lze jej do vstupního souboru včlenit příkazem „*Refresh models*“. Když MC7 načítá vstupní soubor, obsahující součástku, která není uvedena v „*Component Library*“, přečte si schématickou značku a další data ze vstupního souboru a automaticky vytvoří přídavný soubor **IMPORT.CMP** jako rozšíření „*component library*“. Tím je obvod připraven

k okamžité analýze bez toho, abychom kolegovi spolu se souborem **.cir** byli nuceni posílat i své další knihovny, jak to bylo nutné u předchozích verzí.

Kromě načítání dat ze vstupních souborů ***.cir** je možné analyzovat obvod na základě *vstupních textových souborů*, napsaných v jazyku SPICE. Prostředí MicroCapu umožňuje jak vytváření těchto souborů ve vlastním textovém editoru, tak i jejich import. Mnohdy je výhodné použít i další možnost, totiž převod schématu z „*circuit file*“ do vstupního souboru SPICE, s dalším využitím souboru pro jiné účely. Vstupní soubor SPICE totiž obsahuje netlist obvodu spolu s definicí toho, jak má vypadat analýza. Jednoduchou úpravou tohoto souboru pak můžeme zasahovat jak do modelu obvodu, tak i řídit jeho následnou simulaci.

Pro úplnost poznamenejme, že simulátory z rodiny SPICE mají na rozdíl od MicroCapu modulární strukturu. Populární simulátory PSPICE jsou složeny ze tří základních bloků – schématického editoru, vlastního simulátoru PSPICE a programu pro grafické zpracování výsledků analýzy PROBE. Součástí základního „balíku“ jsou i moduly PARTS (obdoba souboru MODEL u MicroCapu) a Stimulus Editor (grafické vytváření zdrojů signálů s komplikovaným časovým průběhem). Nepovinně je možné připojit další moduly. Mezi jednotlivými moduly jsou data předávána soubory s unifikovanou strukturou.

8 Základy práce s programy pro numerickou analýzu obvodů

Cíle kapitoly:

- Ve čtyřech na sobě navazujících lekcích seznámit uživatele se základními možnostmi schématického editoru MicroCapu a se základními typy analýz „Transient“, „AC“ a DC“.
- Vést uživatele krok za krokem při tvorbě vlastního zadání.
- Objasnit způsoby práce s příkazem .MODEL.
- Podrobně vysvětlit způsoby kreslení vodičů a práci s „grid“ textem.
- Na příkladech vyložit způsoby práce se SPICE modely a s podobvody SPICE.
- Objasnit způsoby práce s příkazem .DEFINE, s „formula textem“ a se symbolickými proměnnými.

8.1 Úvod

Značná složitost profesionálních simulátorů vyžaduje jiné přístupy k jejich „zvládání“ ve srovnání s jednoduššími programy založenými na symbolických algoritmech. V následující kapitole nejprve pomocí vzorového příkladu absolvujeme velmi jednoduchý průchod programem. Zaregistrujeme řadu faktorů, jejichž význam nám sice bude zatím unikat, ale na druhou stranu nás upozorní na to, čemu bychom se měli věnovat v dalším studiu. Jde v prvé řadě o porozumění postupů při zadávání modelu řešeného obvodu. Poukážeme na některé z možností práce v schématickém editoru, na způsoby práce s příkazy *.define* a *.model* apod. Teprve potom se můžeme věnovat vlastní analýze. Jak bylo řečeno již na úvodních stránkách, nepůjde o kopírování rozsáhlých manuálů. Těch můžete – a budete muset - využít k soustavnému sebezdokonalování. Instalace MicroCapu obsahuje velké množství ukázkových příkladů, které Vám to usnadní, spolu s elektronickou verzí uživatelské a referenční příručky.

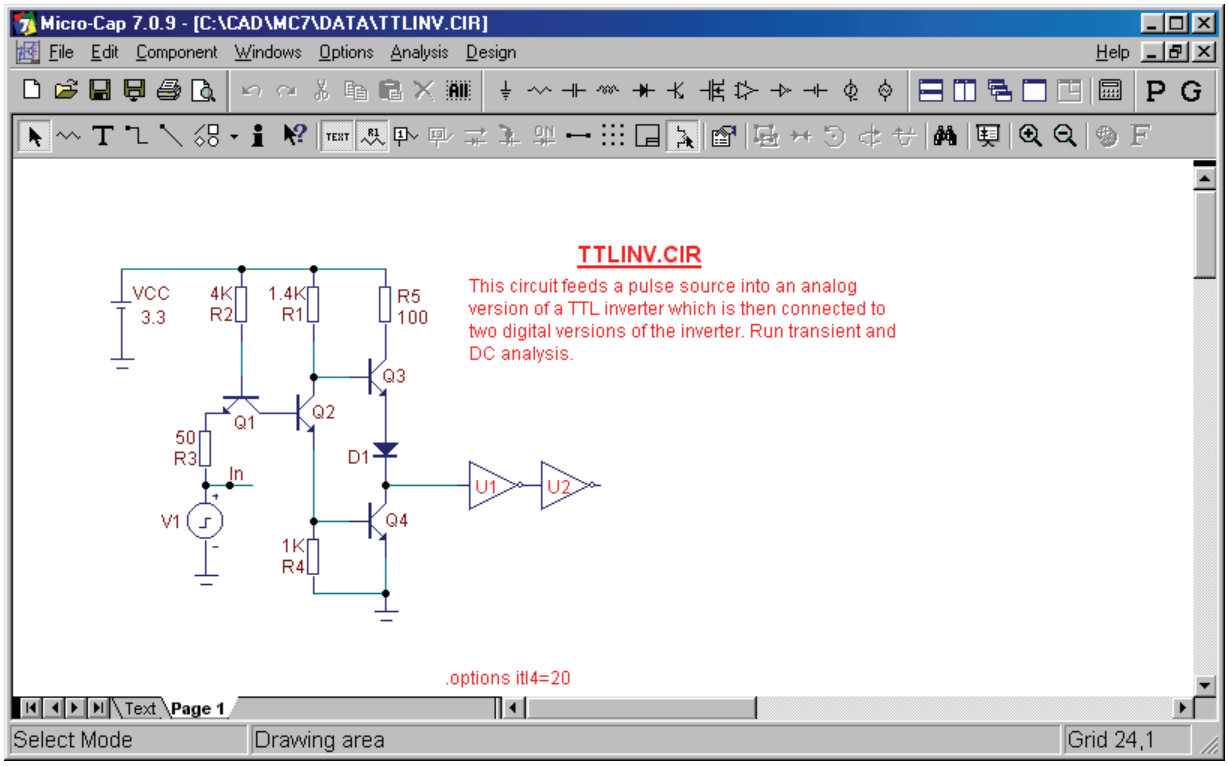
8.2 První praktické kroky v programu MicroCap 7

Na úvod jedno **důležité upozornění**. V následujících čtyřech lekcích budeme pracovat se vzorovými vstupními soubory („*circuit files*“), které jsou součástí instalace MC7. Během experimentů bude docházet k jejich modifikacím. Je proto vhodné před ukončením programu nebo vždy, když se program zeptá, zda si přejeme uložit změny, odpovědět NE (NO). Pokud chcete mít větší jistotu, že originální soubory zůstanou nezměněny, zálohujte si raději jejich kopie.

8.2.1 LEKCE 1 - Toulky schématickým editorem

V této lekci se seznámíme s prostředím schématického editoru, ovšem zatím bez toho, abychom se učili kreslit schémata. Nahlédneme do metod zadávání vstupních dat pro simulátor. Poprvé se seznámíme s příkazem *.model* a začneme tušit jeho význam. Vyvstane řada otázek. Na některé nalezneme odpověď hned, na zodpovězení dalších si budeme muset ještě chvíli počkat.

Spustíme program **MC7.EXE**. V nabídce *File* zvolíme *Open* a v okně *Otevřít* vybereme vstupní soubor **TTLINV.CIR**, který se nachází v adresáři DATA. Na monitoru se objeví schéma z **obr. 8.1**.



Obr. 8.1: Model hradla NAND na tranzistorové úrovni.

Jedná se o model hradla NAND na tranzistorové úrovni. Je doplněn dvěma invertory, které jsou modelovány jako logické obvody.

Na pracovní ploše jsou v podstatě tyto objekty:

Schématické značky součástek, popsané jejich identifikátory (označení součástek, resp. jejich parametrů, a číselné hodnoty), propojovací vodiče a text. Text v horní části obrázku evidentně obsahuje informativní poznámky, které nemají pro vlastní simulaci význam. Text

.options itl4=20 je příkaz (jak uvidíme později, příkazy začínají tečkou) a jako takový ovlivňuje činnost simulátoru. Další typ textu je představován textem *In*, který je umístěn u horního vývodu zdroje *V*1. Jedná se o jméno příslušného uzlu.

Všimněme si, že se nacházíme v složce, ne nepodobné složkám Excelu, jejíž jméno je *Page* 1. Druhá složka editoru se jmenuje *Text*. Klikneme-li do ní, objeví se její obsah:

```
.options digiolvl=1
.MODEL d d (is=10f tt=10n cjo=900f vj=0.7)
.MODEL qn npn (bf=75 is=1f cjc=5p cje=2p vaf=50 tf=.5n tr=5n var=100)
.MODEL v pul (vone=3.5 p1=1000p p2=2n p3=40n p4=41n p5=100n)
.MODEL DLY_TTL UGATE (TPLHTY=11NS TPLHMX=22NS TPHLTY=8NS
TPHLMX=15NS)

* STANDARD TTL DIGITAL INPUT and OUTPUT MODELS
.model DO74 doutput (
+ s0name="1" s0vlo=2 s0vhi=5.5
+ s1name="X" s1vlo=.8 s1vhi=2
+ s2name="0" s2vlo=-1.5 s2vhi=.8)
```

V této složce jsou z „estetických“ důvodů umístěny texty, které by působily rušivě, kdybychom je rozmístili přímo na kreslicí plochu do složky *Page* 1. Principiálně je to však možné. Zatím víme jen to, že texty typu *.options* a *.model* jsou příkazy pro simulátor. Text začínající znakem * je poznámka. Prozatím si řekněme, že zde první tři uvedené příkazy *.model* definují matematické modely diody *D*1, tranzistorů *Q*1, *Q*2 a *Q*3 (které mají stejný model s označením *qn*), a model zdroje signálu *V*1.

Přepněme se zpět do složky *Page* 1.

V tuto chvíli nám možná již vrtají hlavou **první nejasnosti**, například:

- *Nestačí nakreslit schéma jako v programu SNAP bez toho, abychom museli psát nějaké příkazy (kdo se je má učit..)?*

V této věci můžeme zůstat prozatím klidní: například příkazy .model se vytvářejí zcela automaticky po umístění značky konkrétní součástky na kreslicí plochu. Tyto příkazy pak můžeme, ale nemusíme upravovat k „obrazu svému“. Příkaz *.options .itl4=20* je vložen “ručně” uživatelem programu (již zkušenějším) ve snaze obejít určité problémy simulátoru při analýze tohoto konkrétního obvodu (viz dále).

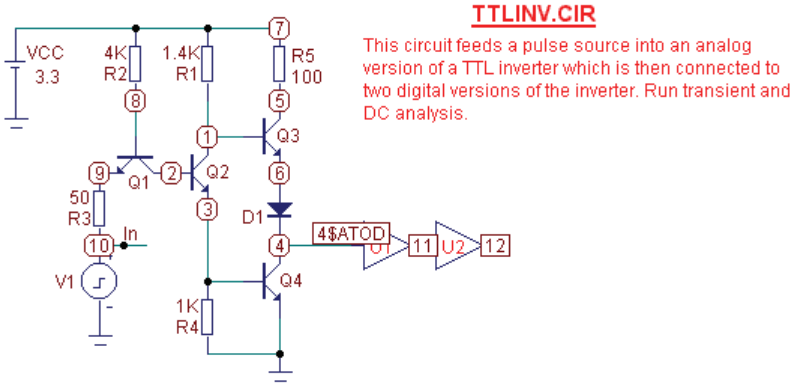
- *Když už musím něco psát do záložek, jak je to s dodržováním konvence VELKÁ/malá písmena (např. u příkazů .options jsou malá písmena a u .MODEL velká)?*

Většina simulátorů nerozlišuje mezi velkými a malými písmeny. Tak je tomu i u MicroCapu. To je například důvod, proč se u inženýrské notace musí odlišit *mili* (což lze zapsat buď jako *m* nebo *M*) od *mega* (zapisuje se jako *MEG*, případně *meg*, *mEg* apod.). Podrobnosti o formě zápisu jsou uvedeny v příloze P1.

- *Z jaké nabídky mohu vybírat součástky, jaká jsou pravidla pro ukládání značek na kreslicí plochu, jakým způsobem se kreslí vodiče, co to znamená, že některé uzly mohou mít své jméno, a jak se vůbec pracuje v prostředí schématického editoru?*

Nepředbíhejme, tyto věci se naučíme v části 9.3 „Tvorba vlastního zadání“.

Během kreslení schématu si program automaticky čísluje uzly. Čísla uzlů nejsou běžně viditelná. Zobrazit je můžeme buď kliknutím na ikonku  na dolní liště v horní části obrazovky, nebo aktivací nabídky *Options/View/Node Numbers*. Výsledek je uveden na **obr. 8.2**



Obr. 8.2: Zviditelnění čísel uzlů.

Uzel, k němuž je připojeno uzemnění, je automaticky opatřen číslem 0 a toto číslo se nezobrazuje. Z hlediska metody uzlových napětí, která bude použita k sestavování obvodových rovnic, se jedná o referenční uzel. Značka uzemnění nesmí chybět v modelu žádného obvodu, chceme-li jej analyzovat MicroCapem. Upozorňujeme na rozdíl oproti programu SNAP, kde se uzemnění nepoužívá. SNAP totiž pracuje s úplnou pseudoadmitanční maticí, z nichž nakonec sestavuje potřebné algebraické doplňky.

Způsob číslování uzlů závisí na tom, v jakém postupu jsme vytvářeli schéma. Teoreticky tedy mohou existovat dvě na první pohled stejná zapojení s různě očíslovanými uzly.

Texty 4*$ATOD*, 11 a 12 v obdélníčcích souvisí s pravidly simulace logických obvodů. V této knize se jimi nebudeme zabývat.

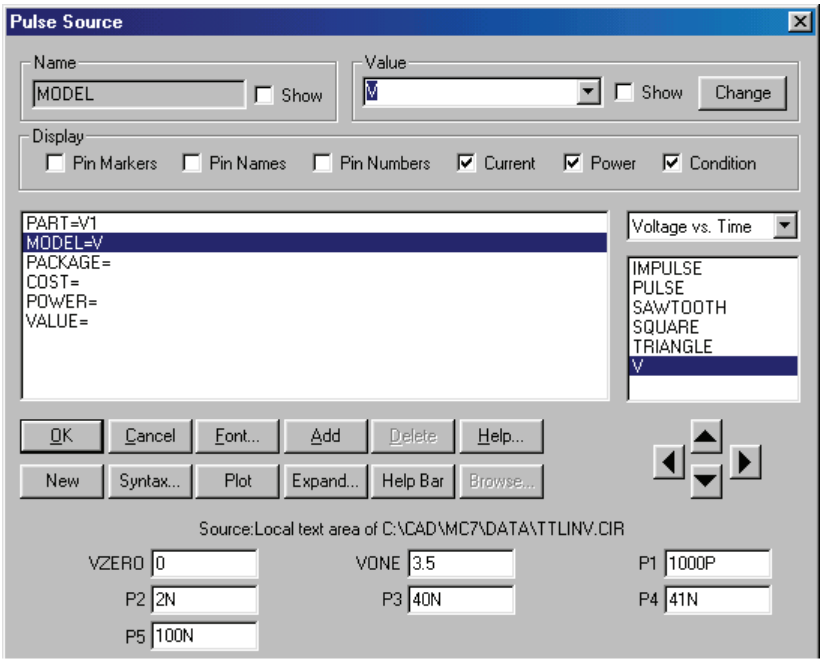
Zatím toho mnoho nevíme o signálovém zdroji *V*1, který je připojen k vstupu simulovaného hradla do uzlu *In* (číslo uzlu 10). Poklepáním na jeho značku levým tlačítkem myši se aktivuje okno *Pulse Source* – viz **obr. 8.3**.

Z okna lze i bez předběžných znalostí zjistit, že:

Součástka se nazývá "*Pulse Source*" – zdroj impulsů. Autor obvodu **TTLINV.CIR** nazval model tohoto zdroje jednoduše *V*. Pro tento model pak vyplnil 7 číselných parametrů v dolní části obrazovky. Význam těchto parametrů zdroje impulsů je zřejmý z přílohy P10.1.1: Jedná se o zdroj napětí o minimální hodnotě 0 V (*VZERO*) a maximální hodnotě 3,5 V (*VONE*). Signál je nulový až do časového okamžiku 1000 ps (= 1*n*s = *P*1). V čase *P*2 = *2n*s dosáhne hodnoty *VONE* = 3,5 V, nástupná hrana impulsu tedy bude trvat 1 *n*s. V čase *P*3=40 *n*s začíná sestupná hrana impulsu a končí zpátky na úrovni *VZERO*= 0 V v čase *P*4=41 *n*s (sestupná hrana tedy má délku 1 *n*s). Impulsy se opakují s periodou *P*5=100 *n*s.

Z textu "*Source:Local text area of C:\CAD\MC7\DATA\TTLINV.CIR*" je zřejmé, že parametry modelu jsou načteny ze vstupního souboru **TTLINV.CIR**. To je v zdánlivém rozporu s informací z kapitoly 7, kde se hovořilo o tom, že knihovny jsou uloženy v souborech typu **.lbr** a **.lib**. Tento problém objasníme v části 9.3.3.

Porovnejme nyní parametry zdroje s příslušným zápisem modelu tohoto zdroje v složce *Text*:



Obr. 8.3: Editační okno zdroje impulsního napětí.

.MODEL v pul (vone=3.5 p1=1000p p2=2n p3=40n p4=41n p5=100n)

Chybí parametr *VZERO*, jinak vše ostatní „sedí“. Proč tento parametr chybí, pochopíme v části 9.3.1.

Před vlastní analýzou obvodu shrňme **možné nejasnosti, nahromaděné v této fázi:**

- *Z obr. 7.1 vyplývá, že knihovny prvků mohou být dvojí, obsahující matematické modely „MicroCapovské“ nebo modely SPICE. Jak mohu ovlivnit volbu modelu? Například při zadávání zdroje impulsů jsem neměl na výběr, a ani nevím, jestli se jedná o model „MicroCapovský“ nebo model SPICE.*

Pro typ modelu se rozhodujeme v okamžiku, kdy vybíráme typ součástky. Například „*Pulse source*“ je součástka vlastní MicroCapu, nemá model SPICE. Typ modelu se pozná podle toho, jak vypadá editační okno součástky. Pokud je něm obsažena položka „*Model*“ a dole se rozbalí okénka na vyplňování parametrů modelu, jde o „MicroCapovský“ model. Podrobnosti budou vysvětleny v části 9.3.

- *Dobře, teď už vím, že modely zdrojů impulsů jsou asi uloženy na disku v nějaké knihovně, která má příponu .lbr (je „MicroCapovská“). Jak mohu zjistit, kde ta knihovna je a jaké modely obsahuje?*

Máme-li otevřené okno *Pulse Source*, vidíme v pravém sloupci názvy těchto modelů:

- *Impulse, Pulse, Sawtooth, Square, Triangle, V.*

Kliknutím do libovolného z prvních pěti modelů se změní text informující o zdroji dat k modelu takto:

- *“Source:Local text area of C:\CAD\MC7DEMO\LIBRARY\SMALL.LBR”*

“Originální” knihovna zdrojů impulsů, dodaná výrobcem programu, tedy obsahuje 5 modelů. Šestý model “*V*” je vytvořen uživatelem pouze pro účely výpočtů obvodu

TTLINV. Proto je automaticky uložen pouze v příslušném vstupním souboru. Pokud bychom model tohoto zdroje chtěli využívat i k analýze jiných obvodů, museli bychom jej včlenit do knihovny **SMALL.LBR**.

Do knihovny **SMALL.LBR**, tak jako do každé jiné, můžeme nahlédnout volbou *File/Open*, načež vybereme masku "*Model Library (*.LBR)*".

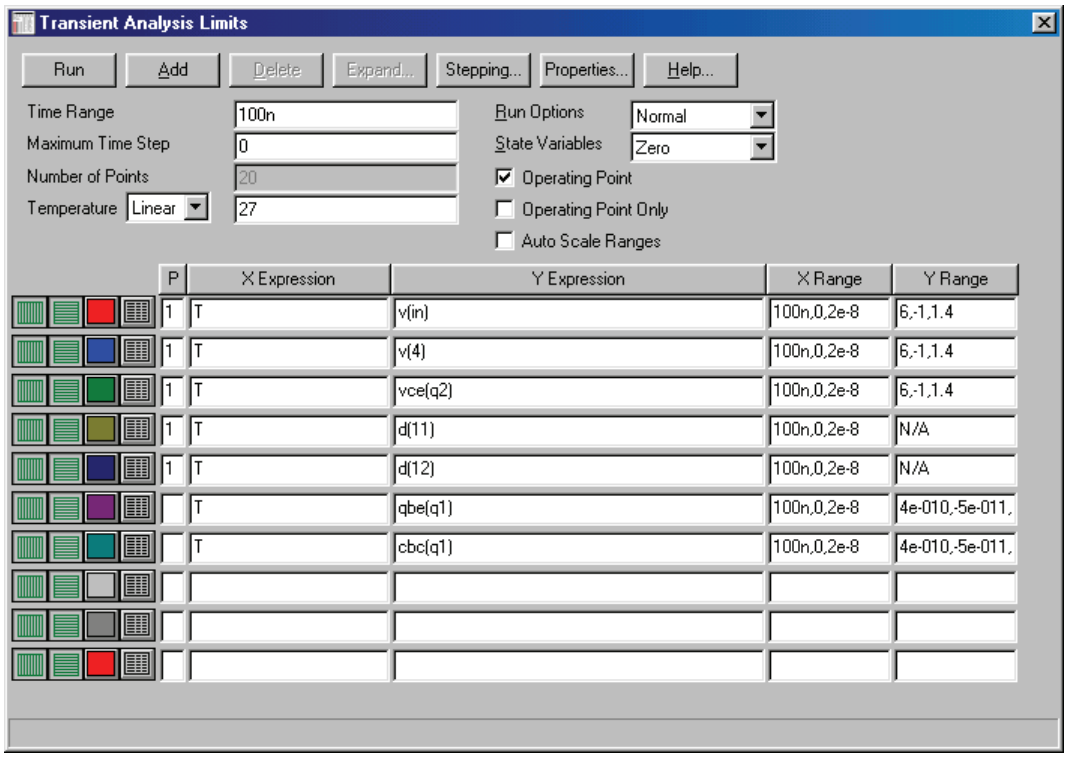
- *Znamená to tedy, že jestliže potřebuji pracovat se součástkou, jejíž model není v "pevných" knihovnách, musím si tento "vlastní" model vytvořit. Jak mám postupovat, když mám o práci s příkazem .model představu méně než mlhavou?*

Je to kupodivu poměrně snadné, podrobný postup je popsán v části 9.3.1.

8.2.2 LEKCE 2 - Analýza "*Transient*"

V této lekci přistoupíme k analýze obvodu **TTLINV.CIR**. Pokusíme se zjistit, jaké budou časové průběhy výstupního napětí hradla (uzel 4 oproti zemi) a napětí kolektor-emitor u tranzistoru *Q*2, bude-li na vstupu působit zdroj definovaných impulsů.

Rozbalíme položku *Analysis* na horní liště a zvolíme první z nabízených typů analýz – *Transient* (analýza časových průběhů). Objeví se okno *Transient Analysis Limits* (viz **obr. 8.4**).



Obr. 8.4: Přednastavené okno "Transient Analysis Limits" pro obvod **TTLINV.CIR**.

Podrobný popis jednotlivých položek je uveden v částech 10.2.4 a 10.3.5. Prozatím si řekneme, že:

Simulace časových průběhů proběhne pro časový interval 100 *n*s (*Time Range*). Výsledkem analýzy bude 5 časových průběhů v jediném obrázku č. 1 (jedničky ve sloupci *P*). Na vodorovné ose bude vždy čas (proměnná *T* v sloupcích *X expression*), na svislé ose budou vyneseny tyto veličiny: napětí uzlu *in* proti zemi, napětí uzlu č. 4, napětí mezi kolektorem a

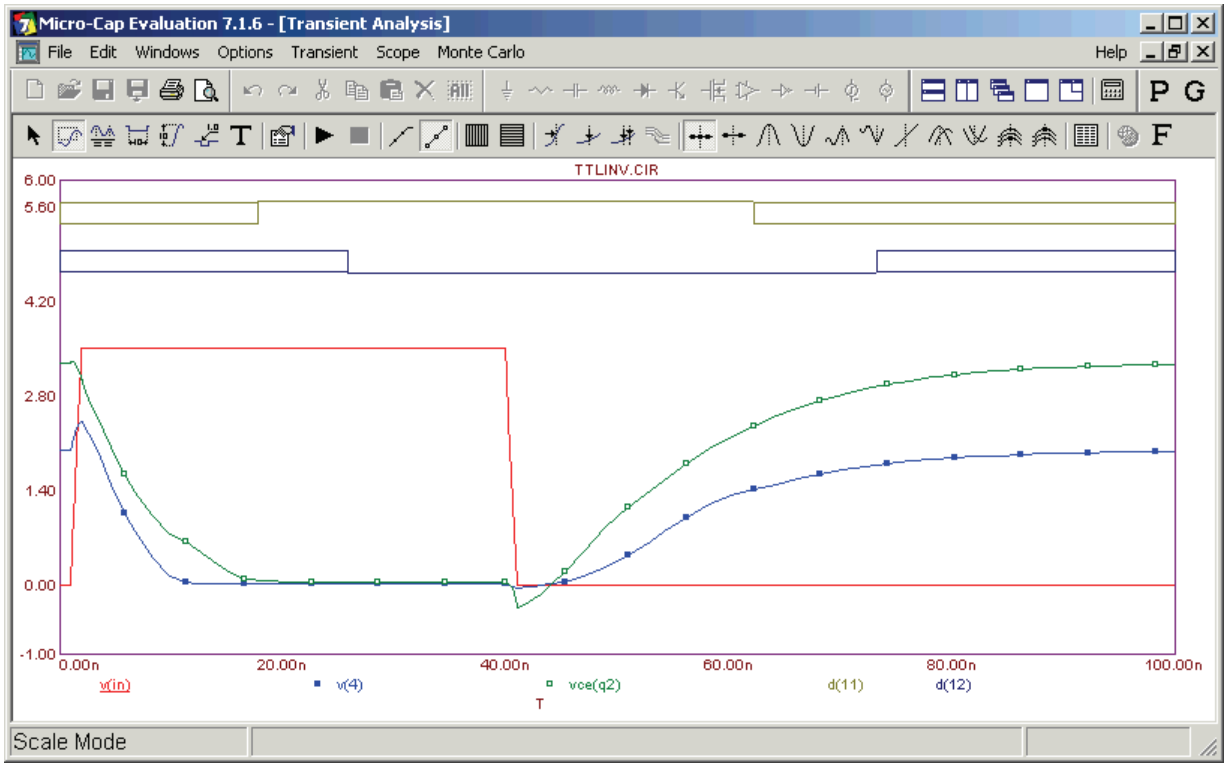
emitorem tranzistoru *q*2, a digitální stavy signálů na uzlech č. 11 a 12 (*v*(*in*), *v*(4), *vce*(*q*2), *d*(11), *d*(12) ve sloupci *Y expression*). Další dva časové průběhy, elektrický náboj mezi bází a emitorem tranzistoru *q*1 (*qbe*(*q*1)) a kapacita mezi bází a kolektorem téhož tranzistoru (*cbc*(*q*1)) se nebudou zobrazovat.

K simulaci dojde po aktivaci tlačítka *Run* v okně *Transient Analysis Limits* nebo po stisku „horké“ klávesy *F*2. Objeví se okno s výsledkem analýzy (viz **obr. 8.5**).

Z obrázku je zřejmá setrvačnost odezvy hradla (křivka *v*(4)) na vstupní signál (křivka *v*(*in*)).

Poznámka: Jestliže se vám „podaří“ během experimentů v tomto grafickém režimu („*Scale Mode*“, viz dále) například podstatně změnit měřítka grafu a potřebujete obrázek uvést do původního stavu, slouží k tomu „dvojhmat“ *Ctrl Home*.

Okno grafických výstupů se nyní nachází v tzv. „*Scale Mode*“ (je „zamáčklá“ ikona tohoto režimu). Požadujeme-li „měřit“ souřadnice jednotlivých křivek, hledat jejich maxima, minima a další význačné body, přepneme se do „*Cursor Mode*“ kliknutím na ikonu , případně na některou z ikon v pravé části lišty pro vyhledávání specifických bodů křivky. Podrobnosti naleznete v nápovědě programu nebo v manuálu.

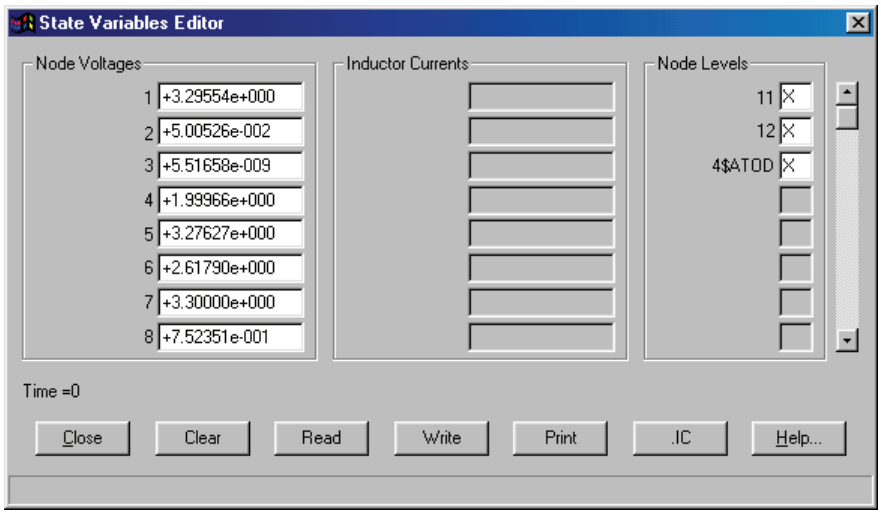


Obr. 8.5: Výsledky časové analýzy (Transient Analysis) obvodu **TTLINV.CIR**.

Prohlédneme-li si znovu **obr. 8.4**, zjistíme, že je zatržena položka *Operating Point*, tj. „Pracovní bod“. Znamená to, že program nejprve nalezne stejnosměrný pracovní bod obvodu „v čase 0“, tzn. při uvažování napětí vstupního impulsního zdroje na úrovni *VZERO* = 0 V, a teprve pak řeší – rozvíjením z tohoto bodu – odezvy na impulsní signál. V praxi to odpovídá situaci, kdy při *VZERO* = 0 V připojíme k obvodu napájecí zdroj, počkáme, až dozní přechodné procesy s tím spojené a obvod se ustálí do stejnosměrného pracovního bodu, a teprve pak simulujeme odezvu na vstup. Podrobnosti, týkající se nastavování této a dalších položek, budou vysvětleny v části 10.3.6.

Nyní zkusme zatrhnout položku „*Operating Point Only*“, tj. „Jen pracovní bod“. V tomto režimu simulátor nalezne stejnosměrný pracovní bod a ukončí analýzu, abychom si mohli prohlédnout výsledek. Po aktivaci *Run*, resp. *F*2, se objeví „prázdné“ okno výsledků analýzy, bez vykreslených odezev. Nacházíme se totiž v počátku časové osy. Nalezený pracovní bod si můžeme prohlédnout dvěma metodami: pomocí tzv. „*State Variables Editoru*“ nebo přímo ve schématu.

Do *State Variables Editoru*, neboli editoru stavových proměnných, se dostaneme přes volby *Transient/State Variables Editor*. Příslušné okno je na **obr. 8.6**.



Obr. 8.6: Okno editoru stavových proměnných.

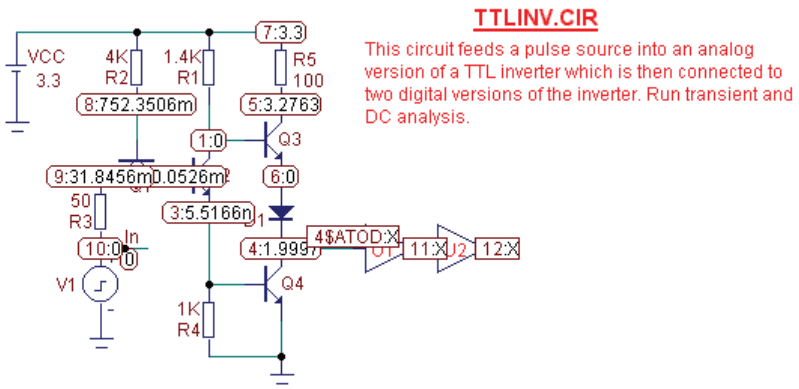
Podrobnosti budou popsány v části 10.3.7. Prozatím postačí konstatování, že za stavové proměnné považuje simulátor všechna uzlová napětí, proudy všemi induktory, pokud se v obvodu nacházejí, popř. logické stavy v uzlech, kam jsou připojeny logické obvody. V sloupci „*Node Voltages*“ si můžeme prohlédnout příslušná napětí v pracovním bodu.

Stavové veličiny si můžeme takto zobrazit vždy po ukončení každého analyzačního běhu, nikoliv pouze u stejnosměrného pracovního bodu. Důležité je uvědomit si, že se jedná o okamžité hodnoty v okamžiku ukončení analýzy. Je-li viditelné toto okno i během analýzy, můžeme dokonce sledovat vývoj stavových proměnných i při analyzačním běhu.

Všimněme si, že editor umožňuje jednak libovolně editovat jednotlivé proměnné, jednak stav obvodu ukládat („*Write*“) do souboru, načítat stav ze souboru („*Read*“) a další operace, o nichž bude podrobněji pojednáno na příslušném místě.

Nyní ukončíme analýzu v režimu „*Transient*“ a vrátíme se do schématického editoru. Můžeme tak učinit buď postupným uzavíráním otevřených oken, bebo výhodněji aktivací horké klávesy *F*3. V paměti je držena informace o stavu obvodu, ve kterém se nacházel v okamžiku ukončení analyzačního běhu, tj. informace o pracovním bodu. Kliknutím do ikony na liště zviditelníme rozložení uzlových napětí – viz **obr. 8.7**.

Podobně lze zobrazit proudy všemi prvky ( ), výkon rozptylovaný na každém prvku ( ) a stav aktivních prvků, tj. lineární režim, saturace apod. ( ). Pro přehlednost nedoporučujeme zobrazovat více druhů těchto veličin najednou.



Obr. 8.7: Druhá metoda zviditelnění uzlových napětí – přímo v schématu.

Na závěr této lekce provedeme opět **shrnutí a rekapitulaci problémů**, jejichž objasnění nás teprve čeká:

- *Rozbalením menu Analysis zjistíme, že analýza „Transient" je jen jednou z mnoha nabízených typů. Co se skrývá pod dalšími položkami?*

Bude podrobněji ukázáno v části 10.1.

- *Jaký je význam dalších položek v okně „Transient Analysis Limits", o kterých jsme se nezmínili?*

Viz část 10.3.5.

- *Jak nastavit optimálním způsobem položky „State Variables" a „Operating Point", abychom dosáhli správných výsledků analýzy konkrétních obvodů v konkrétních pracovních režimech (zesilovačů, filtrů, usměrňovačů, oscilátorů, klopných obvodů, setrvačných obvodů v přechodných stavech...)?*

Dozvíme se v části 10.3.6.

- *Jak vypočítat a zobrazit spektrální čáry analyzovaných průběhů, tj. jak provádět v režimu „Transient" Fourierovu analýzu?*

Bude popsáno v části 10.3.9.

- *Jaké „horké" klávesy je vhodné používat k urychlení práce se simulátorem?*

F2 … Spuštění analýzy, pokud jsou již nastavena její vstupní data v okně „*Analysis Limits*". Platí pro všechny tři základní typy analýz (*Transient*, *AC*, *DC*).

F3 … Ukončí daný analyzační režim, uzavře všechna okna s výjimkou základního okna se schématem obvodu. Výsledky analýzy jsou drženy v paměti.

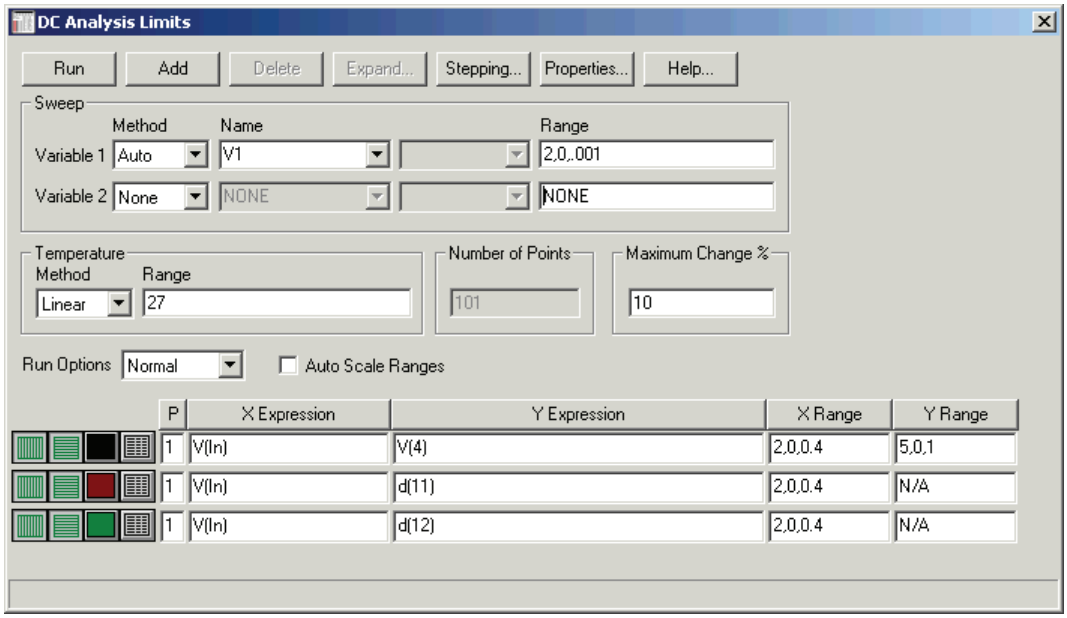
8.2.3 LEKCE 3 - Analýza "*DC*"

V této lekci nahlédneme do dalšího „zákoutí" programu, které nabízí tzv. *DC* neboli stejnosměrnou analýzu. Program v tomto režimu simuluje stejnosměrná měření v laboratoři metodou „bod po bodu". Typickými úlohami jsou například měření ampérvoltových charakteristik diod $I = f(U)$ nebo napěťových převodních charakteristik $U_2 = f(U_1)$. K demonstraci použijeme opět model hradla *NAND* ze souboru **TTLINV.CIR**.

Z nabídky „*Analysis*" nyní vybereme třetí položku „*DC*". Otevře se okno „*DC Analysis Limits*" (viz **obr. 8.8**).

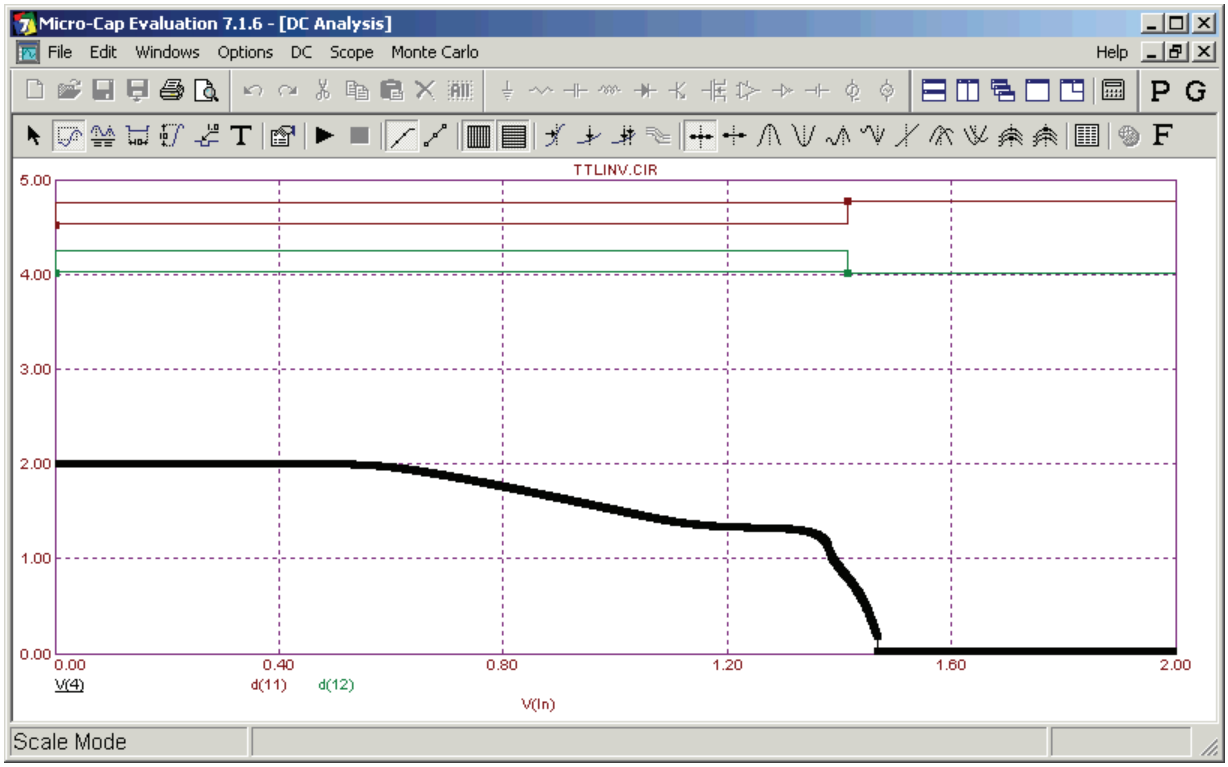
Bez podrobného objasňování jednotlivých položek shrňme, že:

Budeme simulovat postupné nastavování napětí zdroje V1 („Variable 1“, je na vstupu hradla) od 0 V do 2 V. Krok nastavování 0,001 V se neuplatní, protože se bude nastavovat automaticky (AUTO) tak, aby získaná křivka byla dostatečně hladká. Výsledkem analýzy budou 3 křivky v jediném obrázku. Na vodorovné ose bude vždy uzlové napětí příslušející uzlu In, tj. napětí zdroje V1. Na svislou osu se postupně vynáší napětí uzlu 4 a stavy logických signálů v uzlech č. 11 a 12. Simulace se provede pro teplotu 27 °C.



Obr. 8.8: Okno „DC Analysis Limits“ přednastavené pro obvod **TTLINV.CIR**.

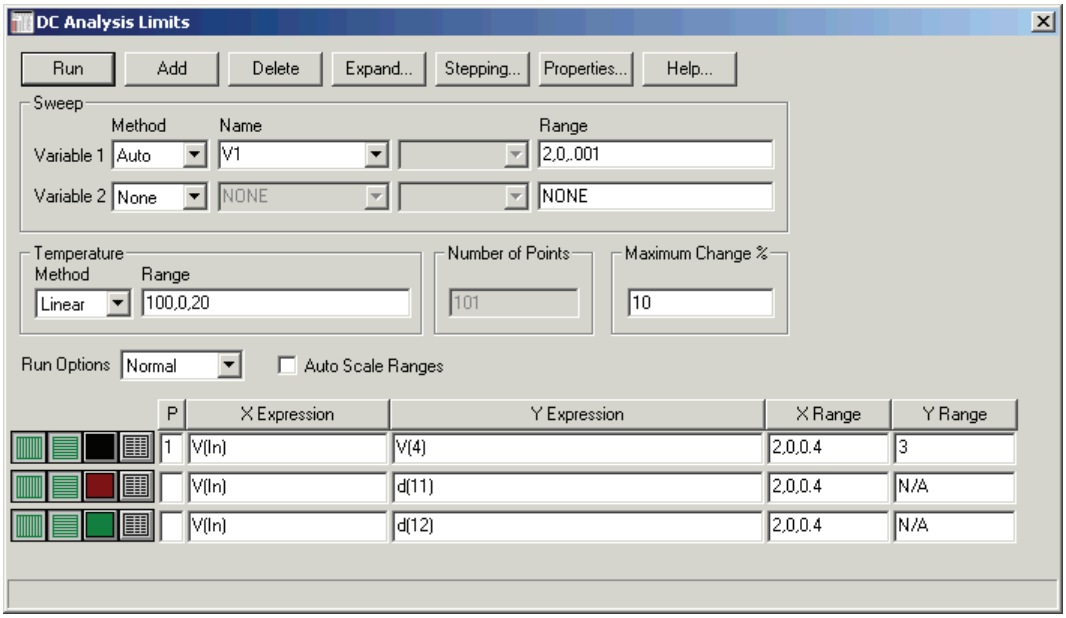
Po aktivaci „*Run*“ (*F*2) získáme výsledky simulace na **obr. 8.9**.



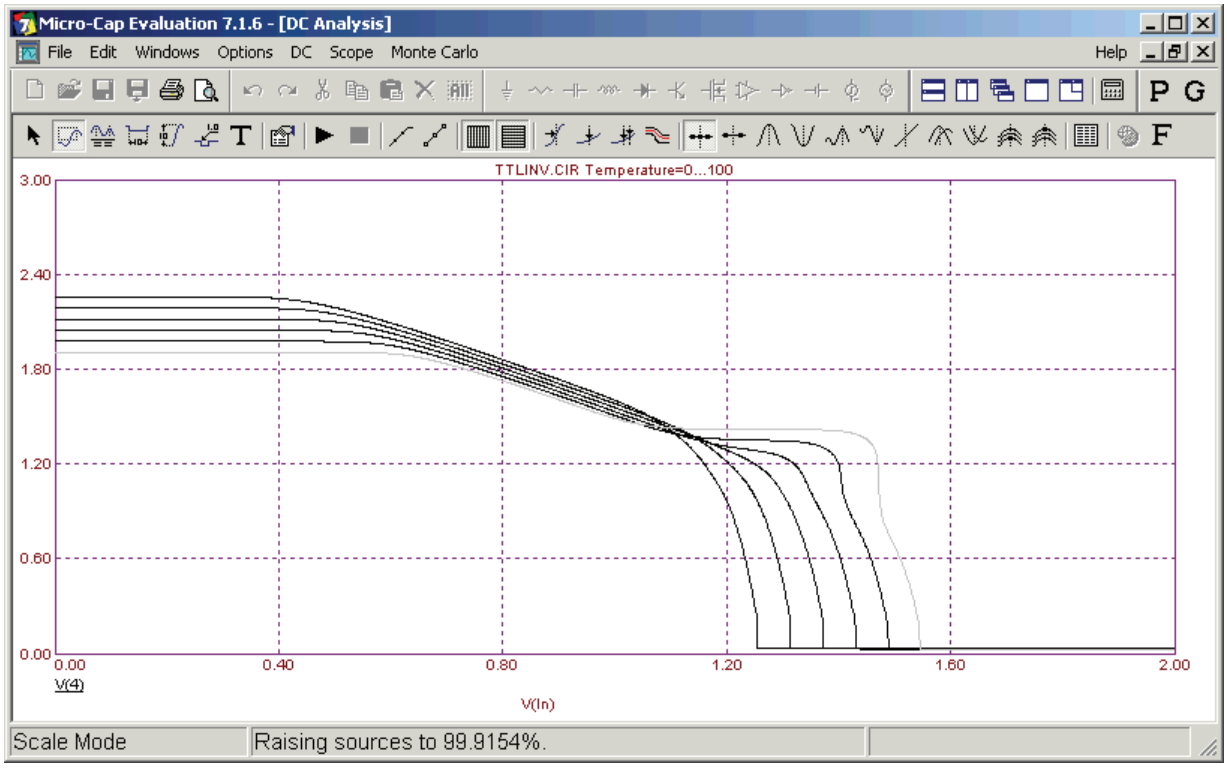
Obr. 8.9: Výsledek stejnosměrné (DC) analýzy obvodu **TTLINV.CIR**.

Převodní charakteristika hradla je vykreslena poněkud silně, protože je aktivována ikona  „*Data Points*“. To znamená, že na křivce jsou znázorněny všechny body, v nichž proběhl výpočet. Zkuste do ikony kliknout, funguje jako přepínač.

Vrátíme se do okna „*DC Analysis Limits*“. Přepíšeme položky tak, jak je to vidět na **obr. 8.10**. Nový zápis v položce „*Range*“ u „*Temperature*“ nyní znamená, že analýza proběhne pro teplotní rozsah od 0°C do 100°C v krocích po 20°C. Měli bychom tedy obdržet celkem 6 charakteristik pro teploty (0, 20, 40, 60, 80, 100)°C. V sloupci „*P*“ jsou v 2. a 3. řádku odstraněna čísla obrázku 1, takže se nebudou do obrázku vykreslovat logické stavy v uzlech 11 a 12. Měřítko na ose *Y* je pozměněno na 3V (tj. rozsah od 0 V po 3 V).



Obr. 8.10: Modifikace vstupních požadavků na DC analýzu zadáním simulačních teplot.



Obr. 8.11: Převodní charakteristika hradna NAND pro teploty (0, 20, 40, 60, 80, 100)°C.

Po proběhnutí analýzy obdržíme výsledek na **obr. 8.11**.

Pomocí *Helpu* nebo své šikovnosti zkuste (v grafickém režimu „*Cursor Mode*“) zjistit, jak jsou křivkám přiřazeny jednotlivé teploty. Z fyziky polovodičů je známo, že při růstu teploty se snižují prahová napětí tranzistorů. To se promítá do posuvů převodní charakteristiky hradla.

Na závěr lekce č. 3 provedeme opět **rekapitulaci**:

- *Co znamenají další položky v okně „DC Analysis Limits“, například „Variable 2“, „Temperature/Metod“, “Maximum Change”?*

  Dozvíme se v části 10.5.5.

- *Je možné v tomto režimu analyzovat součástky, jejíž stejnosměrné charakteristiky tvoří celý system křivek, například soustavu výstupních charakteristik tranzistoru $I_C=f(U_{CE})$ při parametru I_B?*

  Ano, zkombinujeme-li položky “*Variable* 1” a “*Variable* 2”. Podrobněji o tom pojednáme v části 10.5. Tento typ analýzy je ukázán např. v souboru **IVBJT.CIR**.

8.2.4 LEKCE 4 - Analýza "*AC*"

Seznámíme se s jednou z nejčastěji používaných analýz – “*AC*” neboli “střídavou” analýzou, nebo lépe kmitočtovou analýzou. V tomto režimu program simuluje činnost obvodového analyzátoru při snímání kmitočtových charakteristik.

Z příkladu analýzy kmitočtového korektoru mimo jiné vyplynou další poznatky o způsobech modelování součástek.

Otevřeme vstupní soubor **BAX.CIR**, který se nachází v adresáři *DATA*. Je to zapojení tzv. Baxandalova korektoru hloubek, tedy obvodu určeného pro zpracován hudebního signálu. Zajímat nás bude kmitočtový charakteristika, t.j. kmitočtová závislost zesílení signálu ze vstupu “*In*1” do výstupu “*Vout*”.

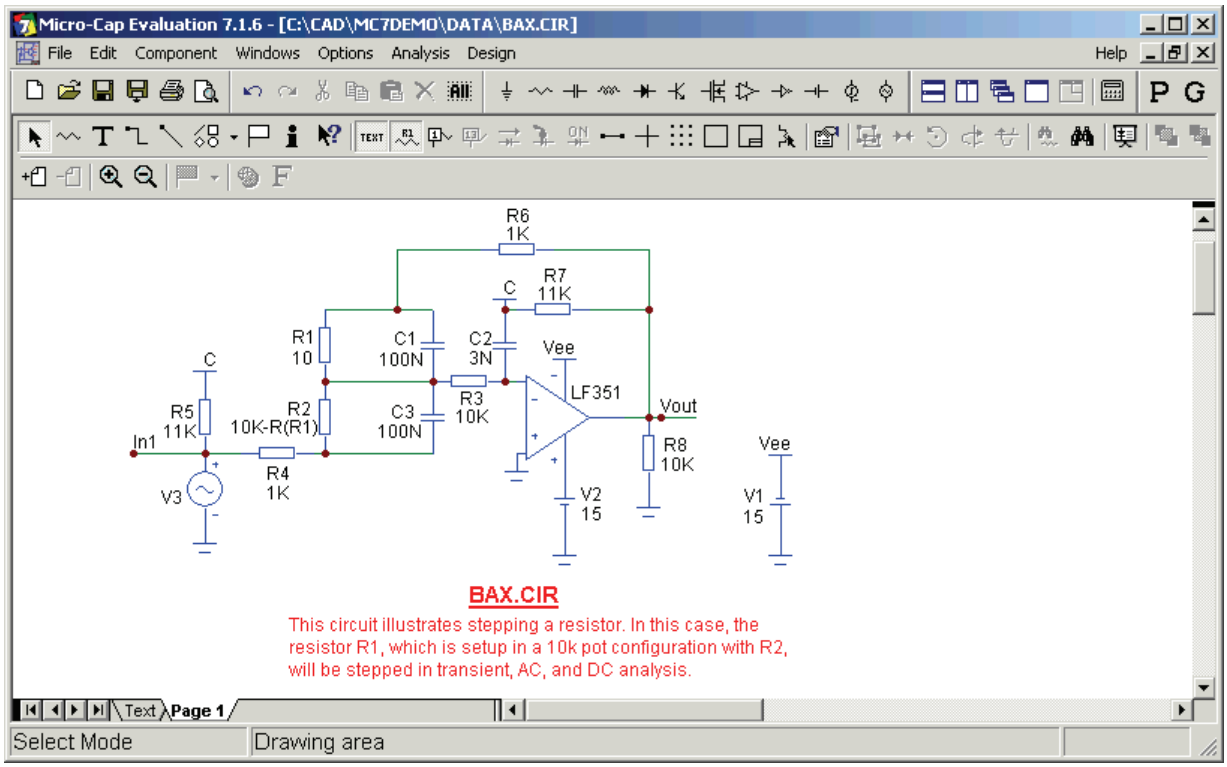
Před vlastní analýzou si povšimněme několika “nových věcí” ve schématu na **obr. 8.12**. Součástka ve tvaru T má originální anglický název “*Tie*”, vázanka. Umístíme-li do různých uzlů obvodu vázanky se stejným označením, např. “*C*” – viz obrázek, pak budou tyto ozly automaticky vodivě spojeny. Tímto způsobem můžeme zpřehlednit schéma, protože se mnohdy vyhneme komplikovanému vedení dlouhých vodičů. Poznamenejme, že v novějších verzích MicroCapu se již bez vázanek docela dobře obejdeme, protože postačí, nazveme-li dva uzly stejnými jmény, a tím dojde k jejich spojení stejně jako u vázanek. Vázanky však mohou působit přehledněji.

Dále si všimněme způsobu modelování rezistorů *R*1 a *R*2. Dohromady totiž modelují potenciometr s odporem 10 *k*Ω, jeho jezdec je vyveden na spojnici *R*1 a *R*2. Odpor *R*1 má nastavenou hodnotu na 10 Ω, odpor *R*2 je definován vzorcem

$$R2 = 10k - R(R1),$$

neboli je roven 10 *k*Ω mínus odpor rezistoru *R*1. Je to výhodnější zápis než

$$R2 = 9990\ \Omega.$$



Obr. 8.12: Model Baxandalova korektoru.

Nyní totiž máme možnost měnit (krokovat) odpor *R*1 a *R*2 se bude automaticky přepočítávat tak, že ve výsledném efektu budeme simulovat otáčení jezdce potenciometru.

Poklepáním na značku operačního zesilovače *LF*351 zjistíme, že se jedná o „MicroCapovský“ model o 24 parametrech. Některé z těchto parametrů jsou zapsány i ve složce „*Text*“. Proč ne všechny, dozvíme se později.

Kmitočtovou analýzu aktivujeme volbou „*Analysis/AC*“. Objeví se okno „*AC Analysis Limits*“ podle **obr. 8.13**.

Bez hlubšího vysvětlování konstatujme, že

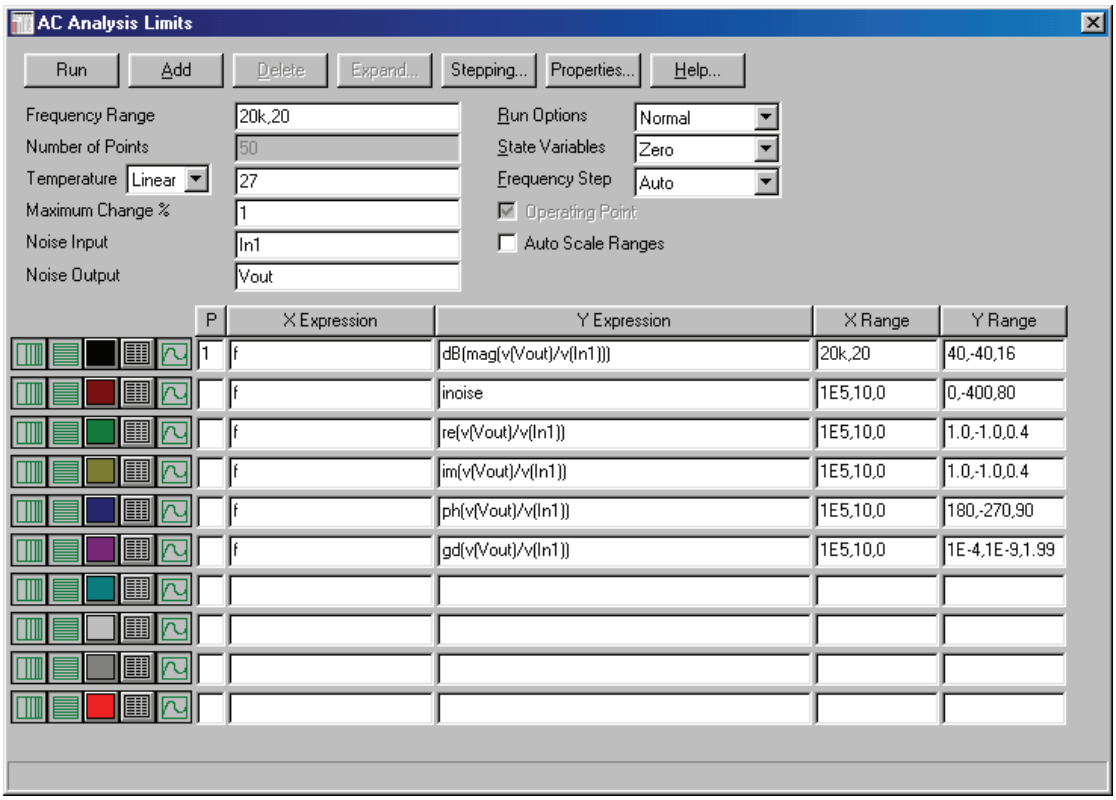
kmitočtová charakteristika bude analyzována v kmitočtovém rozsahu od 20 Hz do 20 kHz („Frequency Range“), v jediném obrázku bude jedna křivka (jen v prvním řádku definovaných veličin je v sloupci „P“ jednička), na vodorovnou osu se vynáší kmitočet (proměnná f v sloupci „X Expression“) a na svislou poměr amplitud výstupního a vstupního napětí v decibelech (vzorec v sloupci „Y Expression“).

Poznamenejme, že vzorec

$$dB(mag(v(Vout)/v(In1)))$$

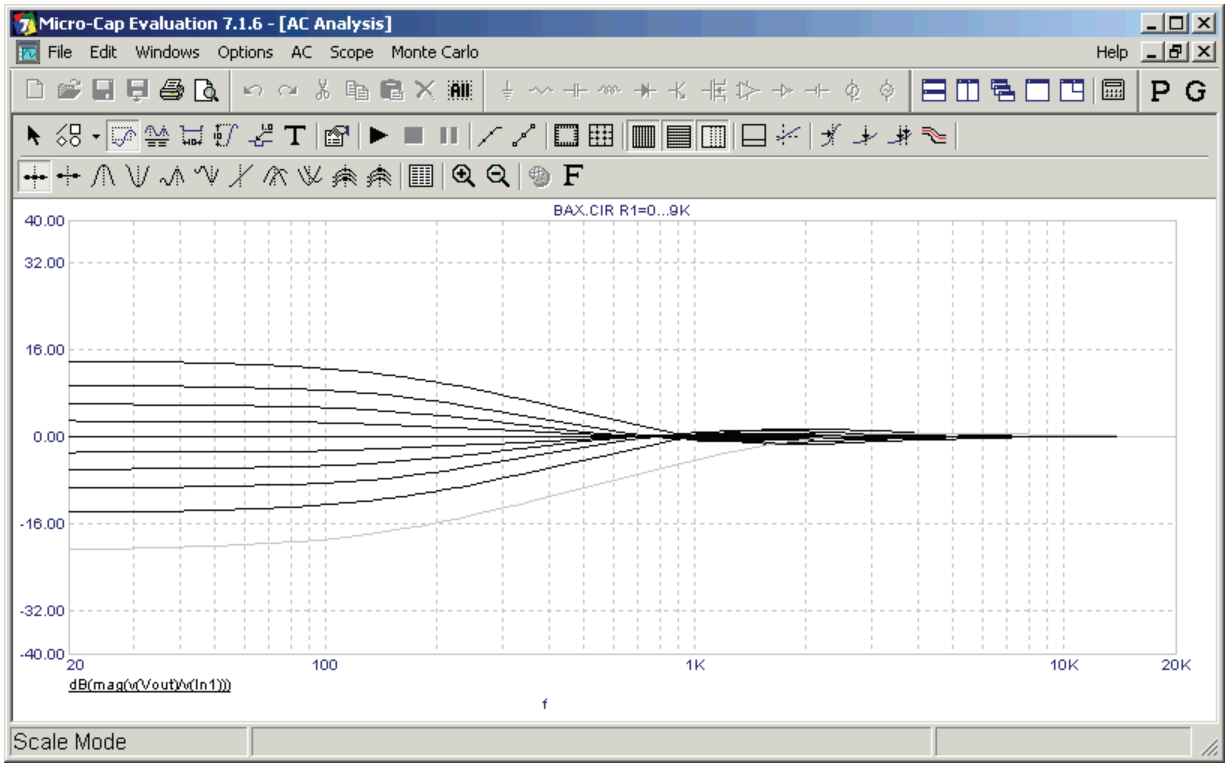
je možné zapsat podstatně úsporněji, jak se dozvíme později v části 10.4.6.

Ještě je vhodné poznamenat, že vodorovná, tj. kmitočtová osa, bude vynesena v logaritmickém měřítku, kdežto svislá, tj. decibelová, bude lineární. Pozná se to podle ikon  a  v 1. řádku, které se při poklepání myší chovají jako přepínače.



Obr. 8.13: Okno „AC analýzy“ přednastavené pro obvod **BAX.CIR**.

Po proběhnutí analýzy obdržíme výsledek na **obr. 8.14**.



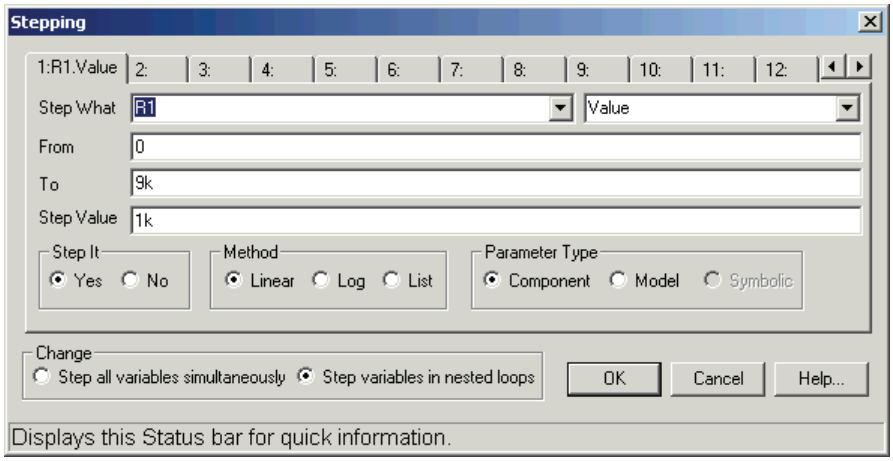
Obr. 8.14: Výsledek kmitočtové analýzy obvodu **BAX.CIR**.

Oproti našemu očekávání se však namísto jediné kmitočtové charakteristiky objevilo hned 10 křivek. Je tomu tak proto, že je aktivován tzv. režim „*Stepping*“ neboli krokování. Z nápisu nad grafem

*BAX.CIR R*1=0…9*K*

je možné vydedukovat, že je krokován odpor *R*1 v hodnotách (0,1,2,3,4,5,6,7,8,9) *k*Ω, čímž simulujeme otáčení jezdce potenciometru, modelovaného rezistory *R*1 a *R*2. Přepneme-li zobrazení do režimu „*Cursor Mode*“, pak pomocí kurzorových šipek ↑↓ snadno zjistíme, že „spodní“ charakteristice, která ukazuje na útlum *nf* kmitočtových složek do cca 1 *k*Hz, odpovídá nulový odpor *R*1, „horní“ charakteristice (zesílení nízkých tónů) odpor *R*1 = 9 *k*Ω, a k rovnoměrnému přenosu dochází při odporu 5 *k*Ω, je-li jezdec v polovině dráhy.

Režim „*Stepping*“ je možné použít ve všech třech typech analýz, „*Transient*“, „*AC*“ i „*DC*“. Příslušné ovládací okno se aktivuje pomocí nabídek na horní liště *AC*/*Stepping* nebo přímo tlačítkem „*Stepping*“ z okna „*AC Analysis Limits*“, případně horkou klávesou *F*11. Podrobnosti budou vysvětleny v části 10.7.1.



Obr. 8.15: Okno pro nastavování podmínek krokování.

Z okna „*Stepping*“ je zřejmé, že je nastaveno krokování hodnot („*Value*“) odporu *R*1 („*Step What*“) od („*From*“) 0 Ω do („*To*“) 9 *k*Ω po („*Step Value*“) 1 *k*Ω. Krokování je povoleno („*Step It – Yes*“).

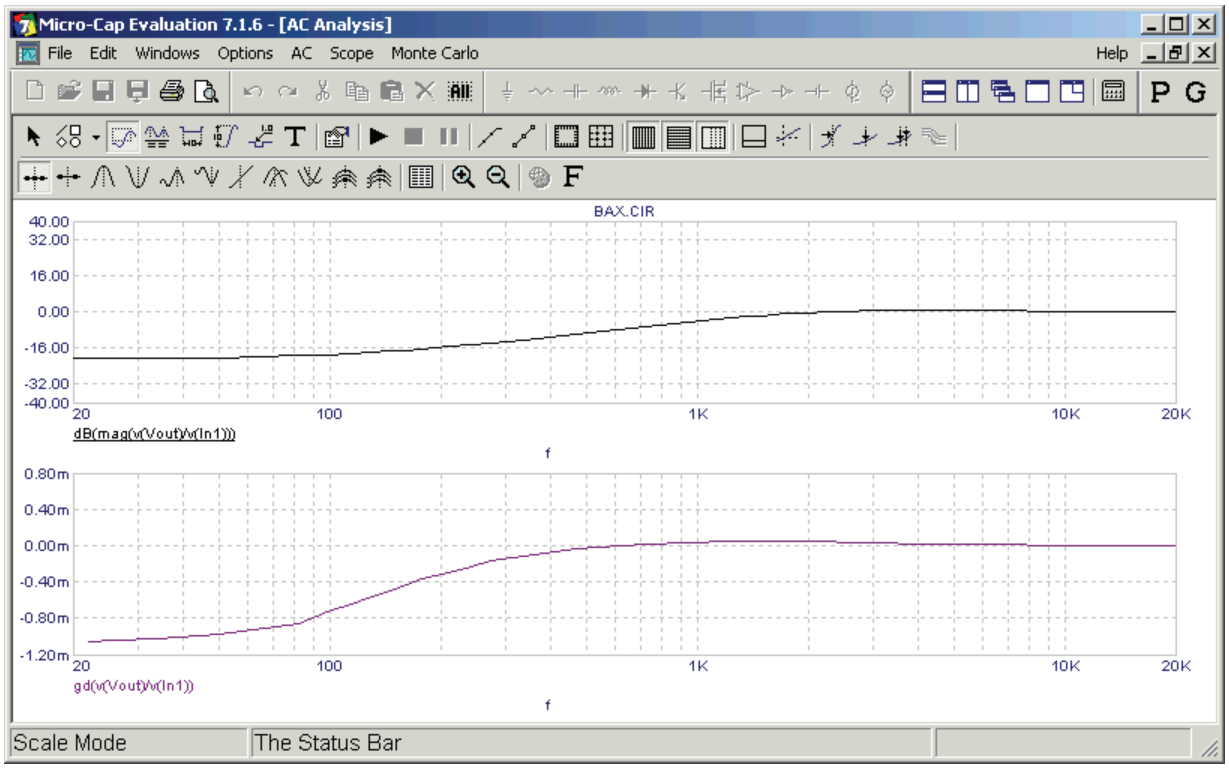
Vyblokujte krokování kliknutím na „*Step It – No*“. Pak zmáčkněte klávesu *F*2, čímž spustíte analýzu. Nyní obdržíme jedinou charakteristiku. Pro jakou hodnotu *R*1? Pro tu, která byla původně zadaná v schématickém editoru, tedy 10 Ω.

Na závěr lekce se pokusíme kromě amplitudové kmitočtové charakteristiky získat i křivku skupinového zpoždění (*GD = Group Delay*). Aktivujeme opět okno „*AC Analysis Limits*“ a upravíme poslední řádek, obsahující příkaz definující skupinové zpoždění tak, jak je vidět na **obr. 8.16**. Do sloupce „*P*“ zapíšeme dvojku, čímž dosáhneme toho, že křivka skupinového zpoždění se vykreslí do jiného obrázku než amplitudová charakteristika. Je to rozumné vzhledem k úplně odlišným jednotkám – a tím pádem i měřítkům – na svislých osách. Údaj v sloupci „*X Range*“ nejpohodlněji změníme tak, že umístíme kurzor do místa editace, aktivujeme pravé tlačítko myši a ze seznamu vybereme text „20*k*,20“ (měřítko na ose *X* bude od 20 Hz do 20 *k*Hz). Podobně budeme postupovat pro sloupec „*Y Range*“, kde vybereme slovo „*Auto*“. Měřítko svislé osy se upraví automaticky podle výsledků analýzy.

Po proběhnutí analýzy budou výsledky uspořádány ve dvou obrázcích tak, jak je to vidět na **obr. 8.17**.

P	X Expression	Y Expression	X Range	Y Range
1	f	dB(mag(v(Vout)/v(In1)))	20k,20	40,-40,16
	f	inoise	1E5,10,0	0,-400,80
	f	re(v(Vout)/v(In1))	1E5,10,0	1.0,-1.0,0.4
	f	im(v(Vout)/v(In1))	1E5,10,0	1.0,-1.0,0.4
	f	ph(v(Vout)/v(In1))	1E5,10,0	180,-270,90
2	f	gd(v(Vout)/v(In1))	20k,20	Auto

Obr. 8.16: Doplnění editačního okna o možnost analýzy skupinového zpoždění (*gd*).



Obr. 8.17: Analýza amplitudové kmitočtové charakteristiky a skupinového zpoždění.

Na závěr proveďme opět ***shrnutí poznatků z této lekce***:

- *V okně „AC Analysis Limits" jsou některé položky, jejichž význam jsme si neobjasnili, např. „Frequency Step", „Numer of Points", „Maximum Change". Je možné při simulaci standardních obvodů ponechat implicitní přednastavení těchto položek?*

V podstatě ano. Číslem v položce „*Maximum Change*" můžeme regulovat hladkost vykreslení křivek. Čím menší číslo, tím vyhlazenější průběh, ovšem za cenu většího počtu bodů výpočtu a tím pádem prodlužování doby analýzy. Detailnější popis je uveden v části 10.4.5.

- *V režimu „AC" analýzy je možné používat zajímavé funkce, např. „dB", „gd" a existuje jistě řada dalších. Kde je možné najít jejich úplný seznam?*

V manuálech a programové nápovědě. Některé funkce budou popsány v částech 10.4.6 a P5.2.

- *Kmitočtové charakteristiky obvodů se měří v harmonickém ustáleném stavu, což znamená, že vstupní signály by měly být harmonické a jejich amplitudy bychom měli volit natolik malé, abychom obvod nepřebuzovali do nelineárního režimu. Znamená*

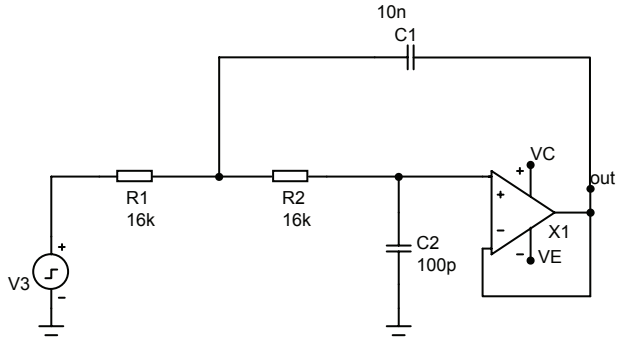
to, že při analýze „AC“ bychom měli tyto skutečnosti respektovat při výběru zdroje signálu?

Naštěstí ne, a to z praktických důvodů. Při kmitočtové analýze bychom museli zaměňovat původní zdroje signálu za jiné, které by vyhovovaly výše popsaným atributům, a při přechodu na jiné typy analýz, např. „*Transient*“ zase vše uvádět do původního stavu. Je to tedy zařízeno tak, že při analýze „*AC*“ je každý budicí zdroj považován za zdroj harmonického signálu o amplitudě, která je specifikována speciálním atributem v modelu zdroje. Obyčejně je tato amplituda 1 V. Nezávisle na tom, jak je amplituda velká, provede program linearizaci obvodu kolem stejnosměrného pracovního bodu, z tohoto modelu určí požadovaný přenos na výstup, a pak jej vynásobí velikostí vstupního signálu. Podrobnosti se dozvíme v části části 10.4.4.

8.3 Tvorba vlastního zadání

8.3.1 Můj první obvod v MicroCapu

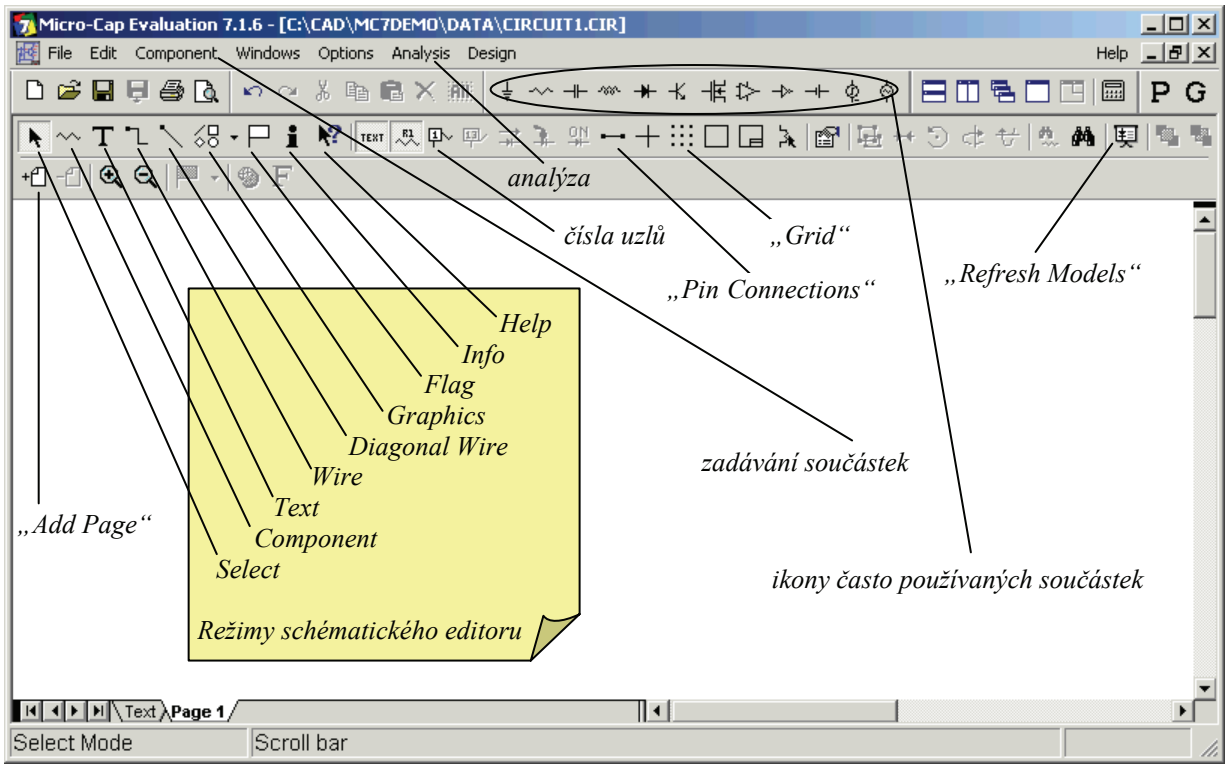
Pokusíme se vytvořit model aktivního filtru typu dolní propust „*Sallen-Key*“ podle **obr. 8.18**. Použijeme operační zesilovač typu *OP*27. Poznamenejme, že tento obvod si můžete pro zajímavost odsimulovat i v programu SNAP a výsledky porovnat. Příslušné soubory **STUSALI.CIR** (ideální operační zesilovač), případně **STUSALR.CIR** (zesilovač s 2 kmitočty lomu) jsou k dispozici po nainstalování SNAPu (viz též příloha P3). Musíte však změnit parametry operačního zesilovače, v souborech je modelován typ 741.



Obr. 8.18: Aktivní filtr „Sallen-Key“ typu dolní propust 2. řádu.

Spustíme program **MC7.EXE**. Objeví se úvodní obrazovka schématického editoru. Na **obr. 8.19** jsou popsány některé důležité položky a ikony.

Schéma se kreslí na tzv. stránky („*Pages*“). Nyní se nacházíme na stránce 1 („*Page* 1“) – viz popis příslušné záložky ve stylu „Excel“. Rozsáhlá schémata se vyplatí rozdělit a kreslit na více stránek. Existují jednoduché prostředky na jejich propojování. Další stránka se přidá pomocí ikony „*Add Page*“, ikona vpravo od ní slouží k odstraňování stránek.



Obr. 8.19: Úvodní obrazovka schématického editoru MicroCapu 7.

Editor umožňuje práci v 9 režimech („*Modes*“):

- **Select** – pro editaci (posuv, rotaci, zrcadlení, mazání, změnu atributů) objektů, již umístěných na stránku.
- **Component** – pro umísťování schématických značek součástek na stránku. Značky se vybírají buď kliknutím do příslušné ikony často používaných součástek, nebo z menu „*Component*“.
- **Text** – pro umísťování textové informace na stránku (poznámky, příkazy, názvy uzlů).
- **Wire** – pro kreslení ortogonálních vodičů.
- **Diagonal Wire** – pro kreslení vodičů vedenými pod obecnými úhly.
- **Graphics** – pro vkládání různých grafických objektů na stránku za účelem „vylepšení“ její grafické podoby (přímka, elipsa, obdélník, kosočtverec, kruhový oblouk, kruhová výseč, obrázky ve formátu .wmf, .emf, .bmp, .gif, .jpg).
- **Flag** – pro umísťování tzv. vlajek do libovolných míst stránky za účelem rychlé orientace v složitých zapojeních. Každé vlajce přiřadíme jméno. Při jeho zadání v režimu „*Go To Flag*“ (ikona s motivem tmavé vlajky vpravo od „*Zoom Out*“) kurzor přeskočí do daného místa.
- **Info** – slouží k zjišťování detailních informací o konkrétní součástce, na kterou klikneme v tomto režimu.
- **Help** – objeví se obecná nápověda k typu součástky, na kterou klikneme v tomto režimu.

Po spuštění programu se editor automaticky nastaví do režimu „*Component*“ a na pozici kurzoru se objeví schématická značka součástky, s kterou se pracovalo naposledy. Můžeme ji buď pokládat na stránku nebo vybrat jinou.

Objekty se na stránku neumísťují do libovolných pozic, ale přichycují se do mřížky („*Grid*“), kterou lze zviditelnit kliknutím na příslušnou ikonu. Objektů, které jsou normálně skryty, je více (čísla uzlů, tzv. *piny*, a řada dalších). Pro jejich zviditelňování slouží buď ikony, nebo položky v menu „*Options*/*View*“.

Za zmínku stojí ještě ikona „*Refresh Model*“, o jejímž významu bude hovořeno později.

Co je nutné vědět před zahájením práce s editorem

Takto se jmenovala obdobná kapitola v části 7.3.1 o editoru programu SNAP, s podtitulem „*Aneb nejčastěji se vyskytující chyby začátečníka*“. Předpokládejme, že jsme již seznámeni s editorem SNAPu, tudíž že nejsme začátečníci. Způsob práce v režimech „*Select*“ a „*Component*“ je u obou programů velmi podobný. Režim „*Line*“ u SNAPu odpovídá režimu „*Wire*“ u MicroCapu. Proto můžeme v podstatě převzít všechny hlavní zásady popsané v úvodu části 7.3.1.

Oproti SNAPu platí v MicroCapu tyto odlišnosti:

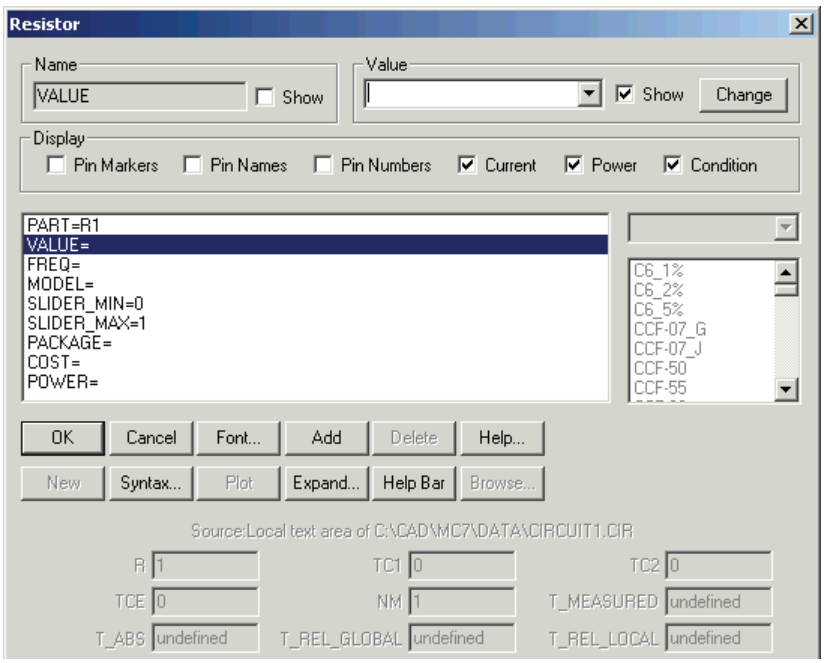
1. Standardně je nastaven režim „*Component*“.
2. Bezprostředně po položení značky součástky na stránku se otevře okno jejích atributů, které je nutné vyplnit. Položky jsou samozřejmě jiné než u SNAPu.
3. Při kreslení vodičů je třeba uvážit jinou filozofii spojování vodičů a jejich nevodivého křížení. Má-li se vodič elektricky spojit s daným bodem, je třeba jej dotáhnout do tohoto bodu a uvolnit levé tlačítko myši. Má-li v daném bodě dojít k nevodivému průchodu, přejedeme při tažení vodiče přes daný bod bez kliknutí. Na rozdíl od SNAPu je vodivé spojení vyznačeno plným kolečkem, tedy uzlem. Ze starších verzí MicroCapu zde zůstává možnost použít pro nevodivé křížení prvek typu „*Jumper*“, jak je tomu u SNAPu.
4. U MicroCapu existuje reálné nebezpečí, že v důsledku chybného umístění objektů na stránku nedojde k jejich elektrickému propojení. Blíže to objasníme na následujícím příkladu. Účinnou pevencí je zobrazování čísel uzlů.
5. V schématu obvodu nesmí chybět značka uzemnění. Microcap pracuje na principu modifikované metody uzlových napětí a sestavuje zkrácenou pseudoadmitanční matici. SNAP sestavuje úplnou admitanční matici, takže předpokládá referenční uzel vně analyzovaného obvodu. Pak z ní vytváří algebraické dopoňky podle umístění vstupní a výstupní brány a podle toho, kterou obvodovou funkci počítáme.
6. SNAP je určen pouze k řešení linearizovaných obvodů, stejnosměrné zdroje zde tedy nehrají žádnou úlohu. U MicroCapu to samozřejmě neplatí. Například ke každému operačnímu zesilovači je třeba zajistit napájecí napětí ze stejnosměrných zdrojů.
7. Oproti SNAPu zde bude hrát důležitou úlohu textová informace, umísťovaná buď přímo na kreslicí plochu, nebo do speciální složky „*Text*“. Kromě poznámek, známých ze SNAPu, budeme mít možnost přiřazovat uzlům jejich jména, definovat symbolické parametry součástek a jejich modely, jakož i ovlivňovat činnost simulátory pomocí příkazů.

Zahájení práce s editorem

Zahájíme vytváření schématu podle **obr. 8.18**. Začneme rezistorem *R*1. Ikonu pro rezistor nalezneme na liště hned vpravo od ikony pro uzemnění (viz též **obr. 8.19**), nebo v menu „*Components/Analog Primitives/Passive Components/Resistors*“. Po aktivaci se objeví značka rezistoru přichycená ke kurzoru myši (pokud je to značka podle americké normy a Vy upřednostňujete evropský způsob kreslení, je možné sjednat nápravu v

„*Component Editoru*“). Posuneme kurzor do vhodné pozice na ploše a značku umístíme levým tlačítkem myši. Objeví se okno „*Resistor*“ podle **obr. 8.20**.

Vidíme, že k součástce lze přiřadit řadu parametrů. Nyní jsme žádáni o zadání „*Value*“ – hodnoty. Do okénka „*Value*“ zapíšeme 16*k*.

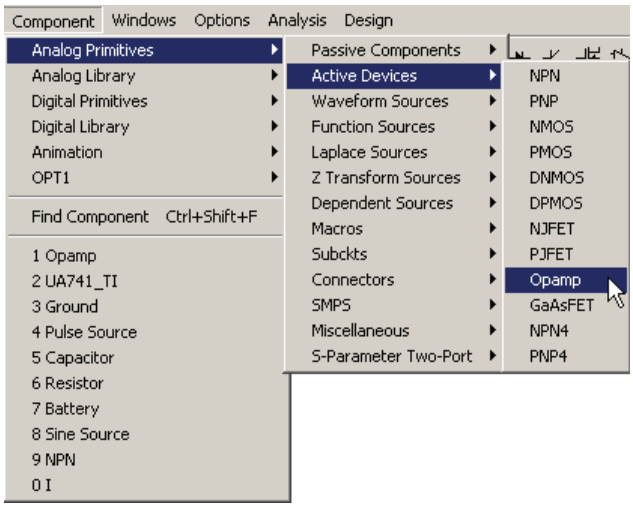


Obr. 8.20: Editační okno rezistoru.

Ostatní položky není třeba v běžných případech vyplňovat. Položka „*PART*“ je analogická k položce „*Name*“ ve SNAPu. Je to označení součástky ve schématu, které je programem generováno automaticky po umístění na plochu. Uživatel má možnost editace. Do položky „*FREQ*“ můžeme zapsat hodnotu nebo vzorec pro výpočet odporu, který má vykazovat rezistor v analýze „*AC*“. Pokud nic nevyplníme, převezme se údaj z pole „*VALUE*“. Pokud hodláme modelovat rezistor podrobněji, např. teplotní závislost odporu nebo toleranci jeho hodnoty pro statistickou analýzu, musíme mu přiřadit model pomocí položky „*MODEL*“ (viz P4.2.1). Význam ostatních položek si může zájemce nastudovat z nápovědy nebo z manuálů.

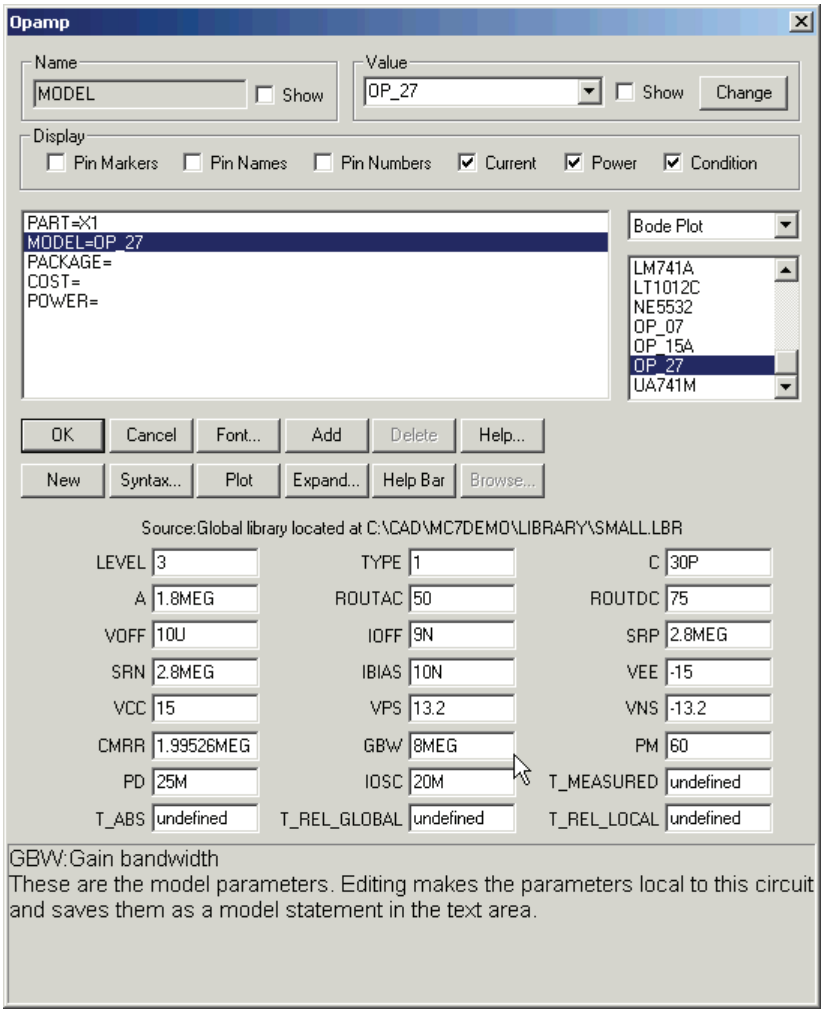
Zadání hodnoty potvrdíme („*OK*“). Okno se zavře a můžeme pokračovat v pokládání dalších součástek. Jejich případná editace je samozřejmě možná, avšak v režimu „*Select*“. Po rezistorech a kapacitorech přejdeme k operačnímu zesilovači. Signálový zdroj si necháme nakonec.

Nyní se zmíníme o výběru operačního zesilovače typu *OP*27. Nejprve zvolíme z nabídky „MicroCapovských“ modelů: „*Component/Analog Primitives/Active Devices/Opamp*“.



Obr. 8.21: Cesta k nabídce „MicroCapovských“ modelů operačních zesilovačů.

Na pozici kurzoru se objeví značka operačního zesilovače. Po umístění do požadovaného místa se objeví editační okno „*Opamp*“. V sloupci modelů na pravé straně vybereme typ *OP*27, tak jak je to znázorněno na **obr. 8.22**.



Obr. 8.22: Editační okno operačního zesilovače OP27.

„MicroCapovský“ model operačního zesilovače obsahuje celkem 24 parametrů. Text nad seznamem parametrů informuje, že model je uložen v knihovně **SMALL.LBR**. Každému

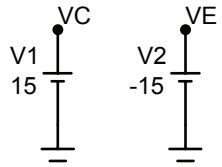
parametru odpovídá stručná řádková nápověda s upozorněním, že pokud se rozhodneme parametry editovat, uloží se ve formě příkazu *.model* do složky *text*. Zde je vhodné dodat, že potom se tento model uloží i do příslušného vstupního souboru.

Význam jednotlivých parametrů modelu je možné zjistit z nápovědy, případně z přílohy P4.3.3. Zde se spokojíme s konstatováním, že pro tento typ operačního zesilovače je použita nejvyšší úroveň modelování (*LEVEL* = 3), že stejnosměrné zesílení otevřené smyčky je 1,8 miliónů (*A*), že výstupní odpory jsou pro stejnosměrný signál 75 Ω (*ROUTDC*) a pro střídavý signál 50 Ω (*ROUTAC*), pozitivní a negativní rychlosti přeběhu jsou 2,8 MV/s neboli 2,8 V/μs (*SRP*, *SRN*), kladné a záporné napájecí napětí jsou +15 V a -15 V (*VCC*, *VEE*), kladné a záporné saturační napětí +13,2 V a -13,2 V (*VPS*, *VNS*), tranzitní kitočet je 8 MHz.

Zadávání ukončíme kliknutím na položku „*OK*“. Objeví se informační okno MicroCapu s hlášením

Opamp power supplies added.

Potvrdíme „*OK*“. Všimněme si, na ploše přibyla další stránka, označená jako „*Power Supplies*“, tj. napájecí zdroje. Ubezpečte se, že do ní program automaticky vložil dvě 15-ti voltové baterie určené k symetrickému napájení operačního zesilovače (viz **obr. 8.23**). Jejich napětí odpovídá hodnotám v položkách „*VCC*“ a „*VEE*“ z modelu. Vývody baterií jsou nazvány jako „*VC*“ a „*VE*“. Stejně jsou pojmenovány uzly pro napájení integrovaného obvodu na stránce 1 („*Page* 1“), čímž je zajištěno příslušné vodivé propojení.



Obr. 8.23: Napájecí zdroje, automaticky vložené na stránku 2 („Page 2“) po umístění operačního zesilovače na plochu.

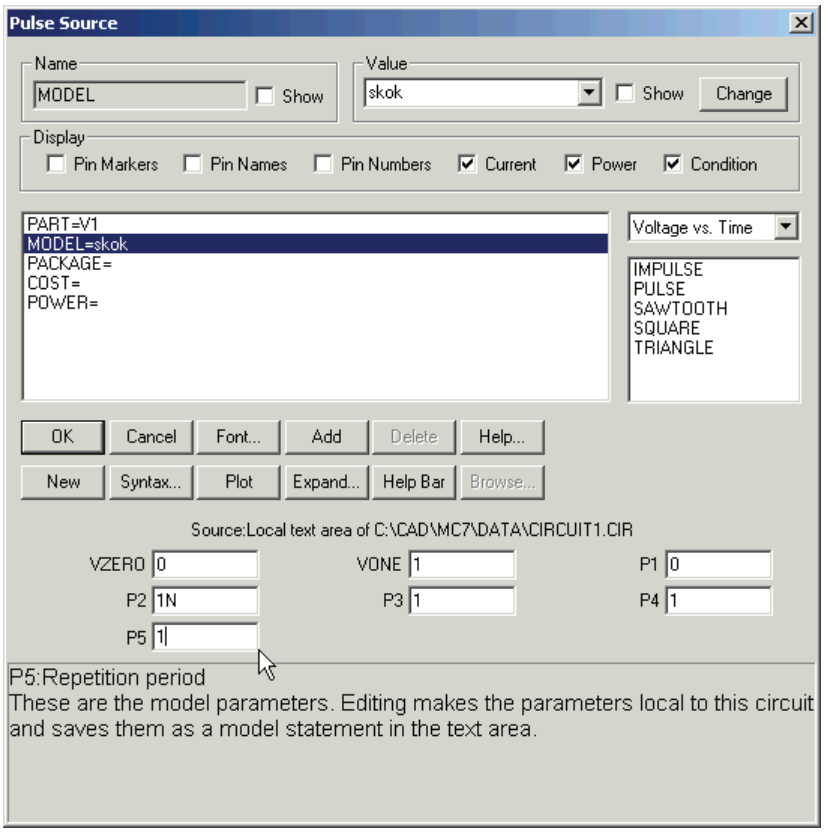
Přemístíme se opět na stránku „*Page* 1“. Na vstup filtru přidáme zdroj signálu, který bude simulovat jednotkový skok. To nám umožní v režimu analýzy „*Transient*“ určit přechodovou charakteristiku filtru. V této fázi se seznámíme trochu podrobněji se zásadami používání příkazu *.model*.

Zásady pro používání příkazu .MODEL

Jako zdroj signálu vybereme „*Pulse Source*“, zdroj impulsů. Jeho ikona se nachází jako předposlední zleva v řadě ikon často používaných součástek. Zdroj impulsů je rovněž dosažitelný v menu

Component/Analog Primitives/Waveform Sources/Pulse Source

Po umístění na plochu se objeví zadávací okno, jehož položky vyplníme tak, jak je to zřejmé z **obr. 8.24**.



Obr. 8.24: Editační okno zdroje impulsního napětí.

Nejprve do pole „*Value*“ zapíšeme text „*skok*“. Tím založíme nový model zdroje impulsů s názvem „*skok*“. Nevyužijeme tedy ani jednoho z 5 modelů, které jsou předdefinovány v knihovně. Podle přílohy P4.1.1 by jednotkovému skoku měly odpovídat následující parametry modelu:

$$VZERO = 0,\ VONE = 1,\ P1 = 0,\ P2 = 0,\ P3 = \infty,\ P4 = \infty,\ P5 = \infty$$

Nekonečno zadat nelze, ale můžeme zadat tak velká čísla, která přesahují hodnoty simulačních časů. Přechodné děje ve filtru se odehrávají v časech řádově jednotky až desítky milisekund. Jestliže časová analýza bude probíhat maximálně do desítek *m*s, pak stačí za parametry *P*3, *P*4 a *P*5 dosadit čísla přesahující maximální dobu simulace. V **obr. 8.24** jsou dosazeny jedničky.

Parametr *P*2 je vhodné zvolit větší než 0, která odpovídá teoretickému průběhu jednotkového skoku. Jsou k tomu dva praktické důvody. Při $P2 = 0$ vykazuje budicí signál v počátku nekonečně velkou derivaci, což může simulátoru způsobit velké problémy při časové analýze. Druhý důvod spočívá v tom, že MicroCap při volbě $P1{=}P2$ nekorektně modeluje celý zdroj impulsů. Můžete se o tom přesvědčit, když spustíte analýzu „*Transient*“ a necháte si vykreslit napětí na tomto zdroji.

V okně „*Pulse Source*“ (viz též **obr. 8.24**) je nad parametry modelu poznámka, že parametry modelu jsou lokálně zapsány (ve skutečnosti tam teprve zapsány budou) ve vstupním souboru **CIRCUIT1.CIR**. Tak se nazývá soubor, s nímž pracujeme. Vytvořili jsme totiž nový model, který není v knihovně. Všech 5 modelů je umístěno v knihovně **SMALL.LBR**, o čemž se můžete přesvědčit kliknutím na název konkrétního modelu.

Po umístění zdroje (kliknutí na „*OK*“ v okně „*Pulse Source*“) nahlédneme do složky „*Text*“. Nalezneme tam následující příkaz:

.MODEL SKOK PUL (VONE=1 P1=0 P2=1n P3=1 P4=1 P5=1)

Tato textová informace se uloží do vstupního souboru spolu s obsahem složky „*Page* 1“ a dalšími daty.

Vzniká otázka, proč nejsou v závorkách vypsány všechny parametry zdroje impulsů a jaká je vůbec struktura příkazu *.model*.

Zjednodušená struktura příkazu .model je následující:

*.MODEL jmeno_modelu typ_modelu (parametr*1*=hodnota parametr*2*=hodnota .....)*

I když se jedná o příkaz programu SPICE, tvůrci Microcapu jej rozšířili o modelování prvků, které nemají v SPICE ekvivalent. Základní SPICE zná 14 předdefinovaných typů modelu, MicroCap jich používá více. U typů sdílených se SPICE je dodržena kompatibilita. Dosud jsme se měli možnost seznámit s typy *PUL* (impulsní zdroj) a *OPA* (Operační zesilovač), které jsou „MicroCapovské“. V části 9.2.1 jsme v příkladu hradla **TTLINV.CIR** zaregistrovali typy *D* (dioda) a *NPN* (NPN tranzistor), které jsou „SPICEovské“.

Zatímco typ modelu je přesně definované klíčové slovo, jméno modelu si určuje uživatel.

Každý typ modelu je charakterizován množinou parametrů. Pro tyto parametry jsou určeny tzv. implicitní hodnoty. Například pro typ *PUL* – zdroj impulsů jsou implicitní hodnoty tyto:

VZERO=0*, VONE*=5*, P*1=100*n, P*2=110*n, P*3=500*n, P*4=510*n, P*5=1*u (mikro)*

Platí zásada, že jestliže v příkazu *.model* nezadáme některý parametr, pak platí automaticky jeho implicitní hodnota. To znamená, že například příkaz

.MODEL POKUS PUL ( )

představuje zdroj impulsů, jehož model se jmenuje *POKUS* a všechny parametry tohoto modelu jsou rovny implicitním hodnotám.

Jestliže v příkazu pro modelování jednotkového skoku

.MODEL SKOK PUL (VONE=1 *P*1=0 *P*2=1*n P*3=1 *P*4=1 *P*5=1*)*

chybí parametr *VZERO*, uvažuje se implicitní hodnota tohoto parametru, což je 0. Samozřejmě že je možné parametr *VZERO* do příkazu dodefinovat, ale je to zbytečné.

Tento princip je při praktickém modelování často využíván. Například model „typického“ operačního zesilovače, u něhož chceme přesně specifikovat jeho tranzitní kmitočet 50 MHz, by vypadal takto:

.MODEL MUJ_OZ OPA (GBW=1*meg)*

S příkazem .model lze kombinovat klíčové slovo *AKO* („*A Kind Of*“ – volně přeloženo „typ odvozený od ..“). Slouží k vytváření klonů existujících modelů součástek. Klonovaná součástka „zdědí“ všechny parametry původního modelu „rodiče“ s výjimkou parametrů definovaných slovy *LOT* a *DEV* (viz část 10.7.4) a s výjimkou parametrů, specifikovaných uvnitř následujících závorek.

Například příkaz

.MODEL 1*N*914*A AKO:*1*N*914 *D (RS*=10*)*

definuje „klon“ 1N914A diody 1N914, který má stejné parametry jako původní dioda s výjimkou parametru RS.

MicroCap nám tvorbu modelů hodně usnadňuje. Pokud si při zadávání součástky vybereme přímo model z knihovny, příkaz *.model* se v textovém poli vůbec nevygeneruje. Není k tomu důvod, simulátor si model načte z příslušné knihovny na disku. Jestliže si vybereme model z knihovny, ale některé jeho parametry modifikujeme, nebo si založíme model vlastní, vygeneruje se v textovém poli příkaz *.model* s parametry, které budou odlišné od implicitních. Tento příkaz se pak uloží do vstupního souboru, takže daný model bude platit pouze pro analyzovaný obvod.

Kreslení vodičů a práce s „GRID“ textem

Nyní editor nastavíme do režimu „*Wire*“ a dokreslíme vodiče. Způsob práce v tomto režimu je stejný jako u SNAPu. Nakonec přidáme značky uzemnění. Zapojení by mělo odpovídat **obr. 8.18**. Chybí vytvořit jméno výstupního uzlu „*out*“.

Zjednodušeně řečeno, pojmenování uzlu znamená umístění příslušného textu na uzel. Je však třeba vysvětlit, co znamená „na uzel“, případně co je vlastně uzel.

Veškerý text, umísťovaný na stránku „*Page*“ v režimu „*Text*“, se nazývá „*Grid Text*“ (umísťuje se do pomyslné mřížky na kreslicí plochu). Protikladem je „*Component Attribute Text*“, který je spojen vždy se součástkou, při jejím pohybu po ploše se pohybuje s ní. „*Grid text*“ můžeme přesouvat do složky „*Text*“ a zpět na stránku vyznačením textu a volbou *CTRL+B*.

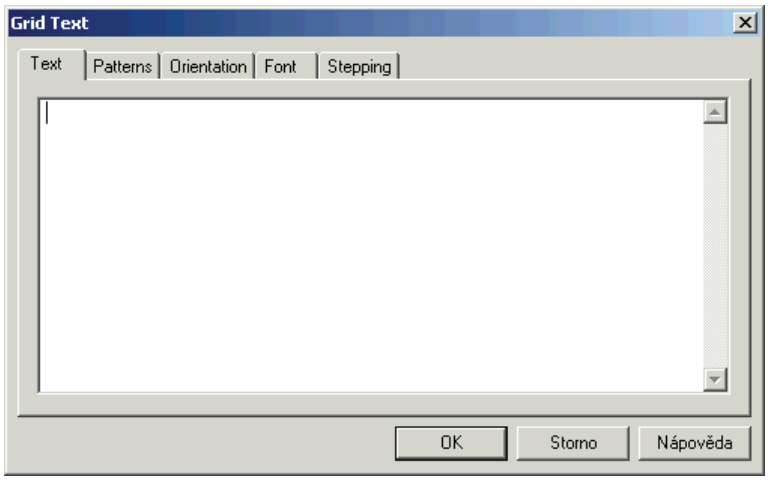
Existují čtyři základní druhy „*Grid Textu*“:

1. Poznámka.
2. Příkaz.
3. Jméno uzlu.
4. „Formula Text“ (viz část 9.3.4).

Jak simulátor „pozná“, o jaký druh textu jde? Příkaz začíná tečkou. Název uzlu „leží na uzlu“ (viz dále). „*Formula Text*“ začíná rovnítkem (=). Veškerý ostatní text jsou poznámky, kterých si simulátor nevšímá.

Postup při umísťování jakéhokoliv „*Grid Textu*“ na stránku:

Kliknutím na ikonu  aktivujeme režim „*Text*“. Přesuneme kurzor do pozice, kam chceme vložit text, a klikneme na plochu. Rozbalí se okno podle **obr. 8.25**.

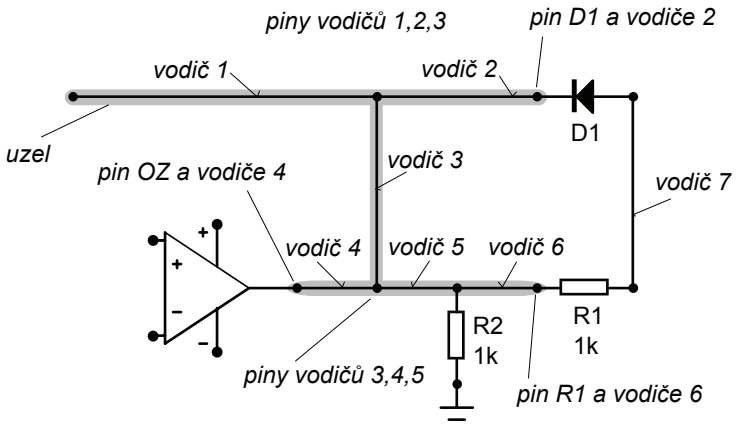


Obr. 8.25: Editační okno pro tvorbu „*Grid Textu*“.

Po zapsání textu a případné modifikaci jeho atributů v dalších složkách okna (fonty, barvy apod.) neukládáme text na plochu zmáčknutím *ENTER* (což je častá chyba), nýbrž kliknutím na „*OK*“.

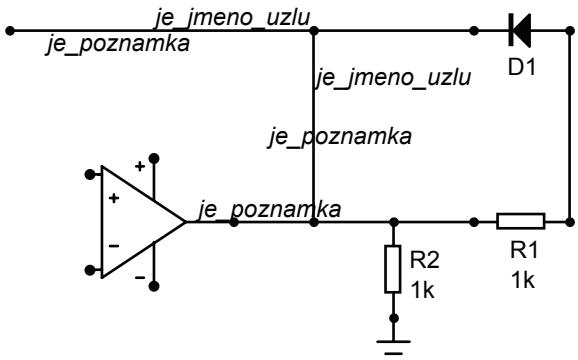
Nyní vysvětlíme, co je to uzel v prostředí schématického editoru a jak mu přiřadit jméno.

Schématická značka každého prvku sestává z těla a tzv. pinů, které se dají zviditelnit kliknutím na příslušnou ikonu na liště nebo způsobem, popsaným v části 9.3.1. Piny jsou „vodivé plošky“ na vývodech. Můžeme si představit, že piny jsou jediná místa, pomocí nichž se dá součástka vodivě propojit s okolím. Vše ostatní je „pokryto izolací“. Pin se může dotýkat s pinem jiné součástky nebo s ním být spojen vodičem. **Uzel je množina všech vodivě propojených pinů plus všechny vodiče, které tyto piny spojují a dotýkají se jich**. Tato skutečnost je ilustrována na **obr. 8.26**.



Obr. 8.26: K vysvětlení pojmu „uzel“.

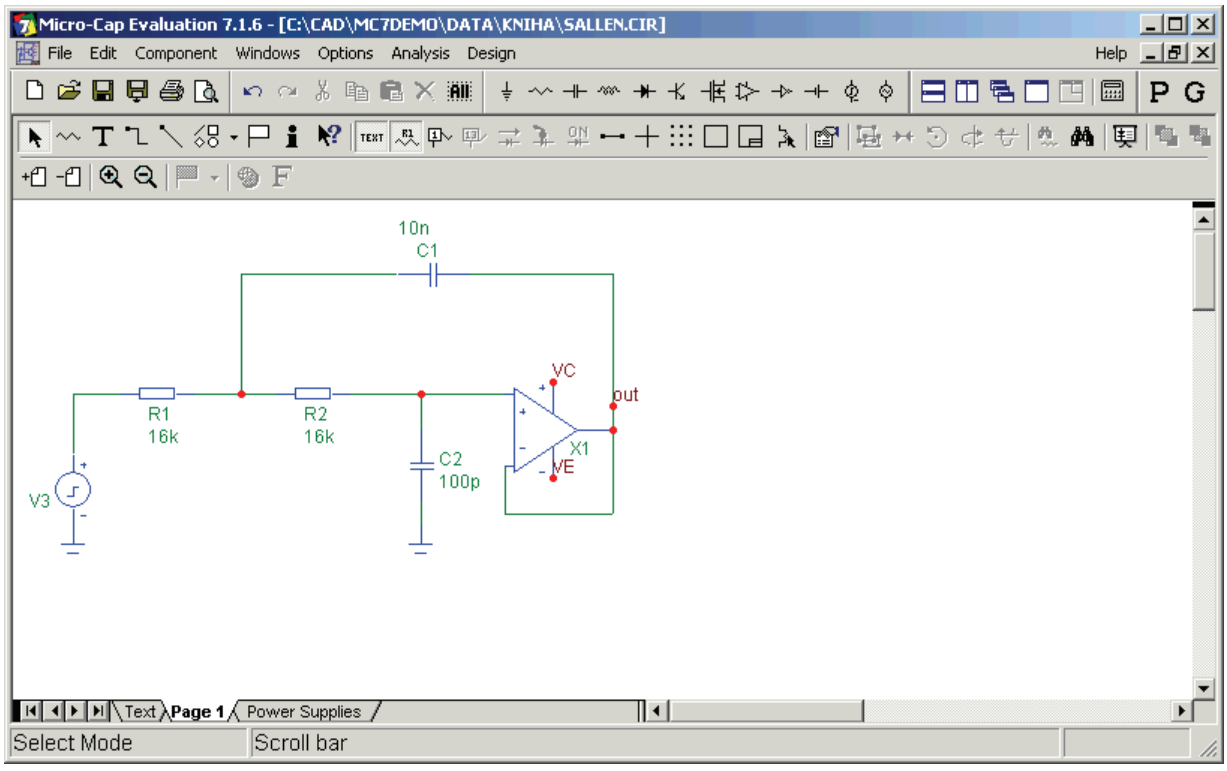
Jméno uzlu je „*grid text*“ umístěný kamkoliv na uzel. „*Grid text*“ je na uzlu, jestliže jeho tzv. stykový bod („*pin connection dot*“) leží na uzlu. **Tento bod se nachází v levém dolním rohu bloku, který ohraničuje text po jeho vyznačení** („vyselektování“). Na **obr. 8.27** je toto pravidlo opět ilustrováno.



Obr. 8.27: K vysvětlení správného umísťování jména uzlu do schématu.

V standardním nastavení programu je správné umísťování jména uzlu na uzel usnadněno zatržením položky „*Node Snap*“ v složce „*Common Options*“ v okně „*Preferences*“, které je dosažitelné volbou „*Options*/*Preferences*“. Pokud umístíme text do správné pozice v rámci chyby elementární rozteče mřížky, text je automaticky „přitažen“ do správné polohy.

Tvorbu obvodu podle **obr. 8.18** tedy zakončíme pojmenováním výstupního uzlu textem „*out*“. Dílo uložíme na disk do libovolného adresáře. Soubor nazveme například „**SALLEN.CIR**“. Výsledek by měl odpovídat **obr. 8.28**. Složka „*Text*“ je prázdná, v složce „*Power Supplies*“ jsou napájecí baterie (viz **obr. 8.23**).

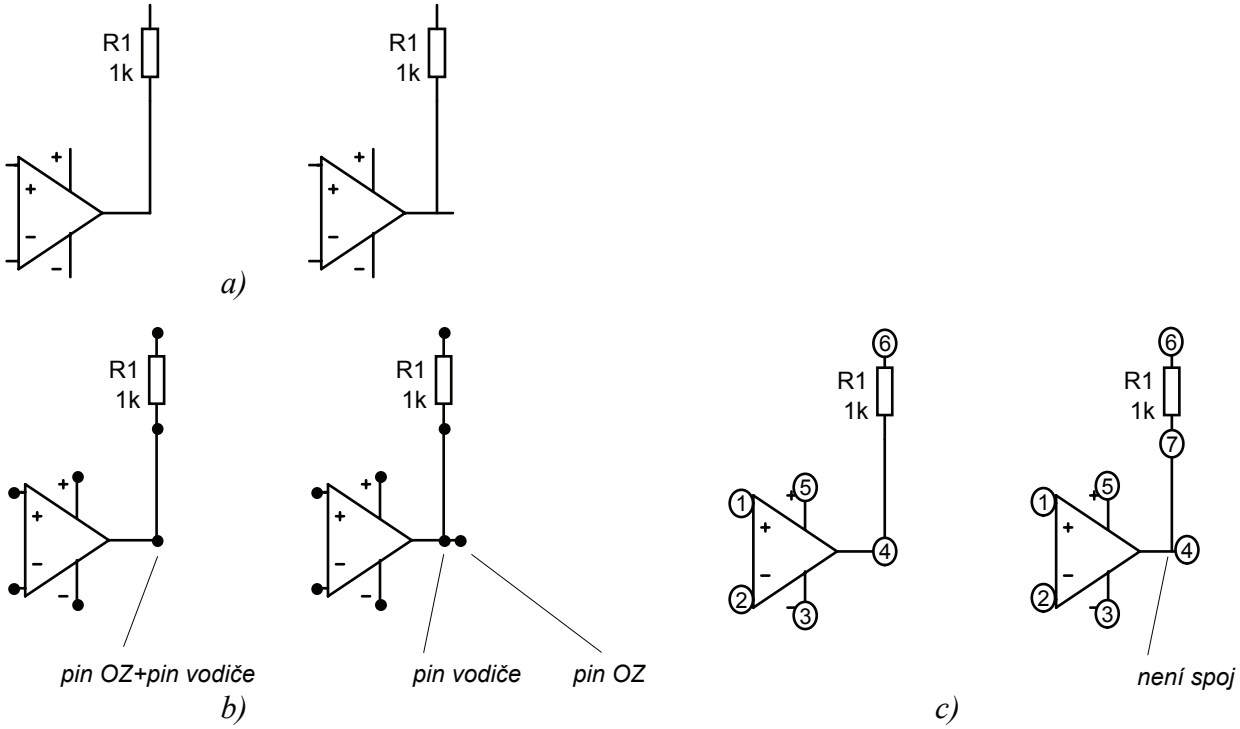


Obr. 8.28: Schéma filtru typu *Sallen-Key*.

Dříve než provedeme konkrétní analýzu obvodu a seznámíme se s dalšími možnostmi modelování obvodů, přečtěme si o časté chybě, které se můžeme snadno dopustit při kreslení schématu.

Problém nevodivých spojení

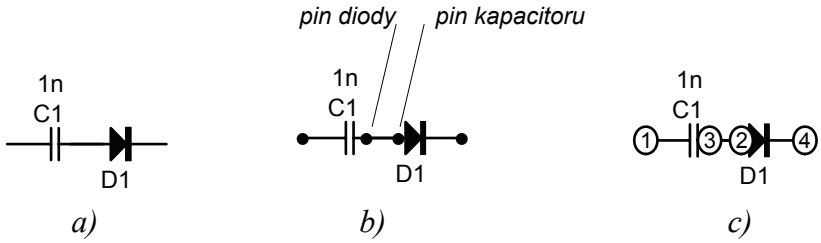
Na **obr. 8.29** a) je ukázka dvou zapojení, které vypadají z hlediska elektrického zapojení jako ekvivalentní. Na zapojení vpravo ale není rezistor *R*1 ve skutečnosti spojen s výstupem operačního zesilovače.



Obr. 8.29: K problému nevodivého spojení.

Problém je v tom, že editor MicroCapu vyznačuje kolečkem uzel až od spojení tří a více komponentů. Ze zapojení na **obr. 8.29** a) je tedy nesnadné na první pohled poznat, jedná-li se o vodivé spojení nebo ne.

Na **obr. 8.29** b) jsou tyto obvody zakresleny znovu při zviditelněných pinech. Je vidět, že u obvodu vlevo se pin operačního zesilovače kryje s pinem vodiče, na obrázku vpravo leží pin vodiče na „izolaci“, nikoliv na vývodu operačního zesilovače. V tomto místě tedy není spoj. Po zviditelnění čísel uzlů (obr. **obr. 8.29** c) se tato chyba projeví jako dvě různá čísla uzlů tam, kde očekáváme jediný uzel.



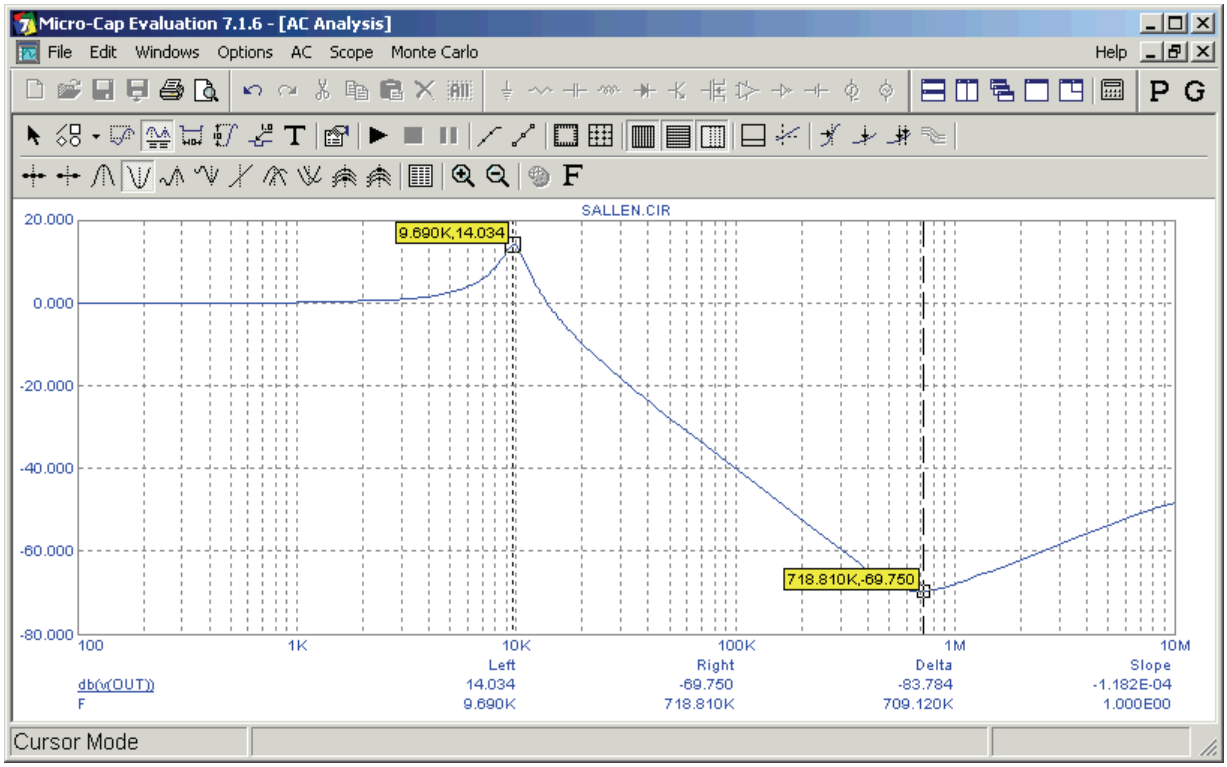
Obr. 8.30: Problém „překřížení pinů“.

Na **obr. 8.30** je ukázána další chyba obdobného typu. Danou situaci bychom mohli označit jako „překřížení pinů“. Na obr. a) se zdá, že je vše v pořádku. Po zviditelnění pinů však vidíme, že kapacitor s diodou vlastně nejsou vodivě spojeny: piny obou prvků leží na „izolaci“.

Pokud je aktivní režim „*Node Snap*“, pravděpodobnost vzniku těchto chyb se sice zmenšuje, ale nelze je při nepozorném kreslení vyloučit. Pokud nás nenapadne zviditelnit číslování uzlů, pak se chyba tohoto typu těžko hledá.

Kontrolní analýza obvodu SALLEN.CIR

Analýza sice není náplní této kapitoly, ale když se nám podařilo sestavit model filtru, byla by škoda nevyzkoušet jeho funkci. Pokuste se zjistit amplitudovou kmitočtovou charakteristiku filtru v kmitočtovém rozsahu od 100 Hz do 10 MHz. Postupujte podobně jako v části 9.2.4. LEKCE 4 - Analýza "*AC*". Měli byste obdržet výsledek na **obr. 8.31**.



Obr. 8.31: Kmitočtová charakteristika analyzovaného filtru *Sallen-Key* z **obr. 8.28**.

8.3.2 Práce s modely SPICE

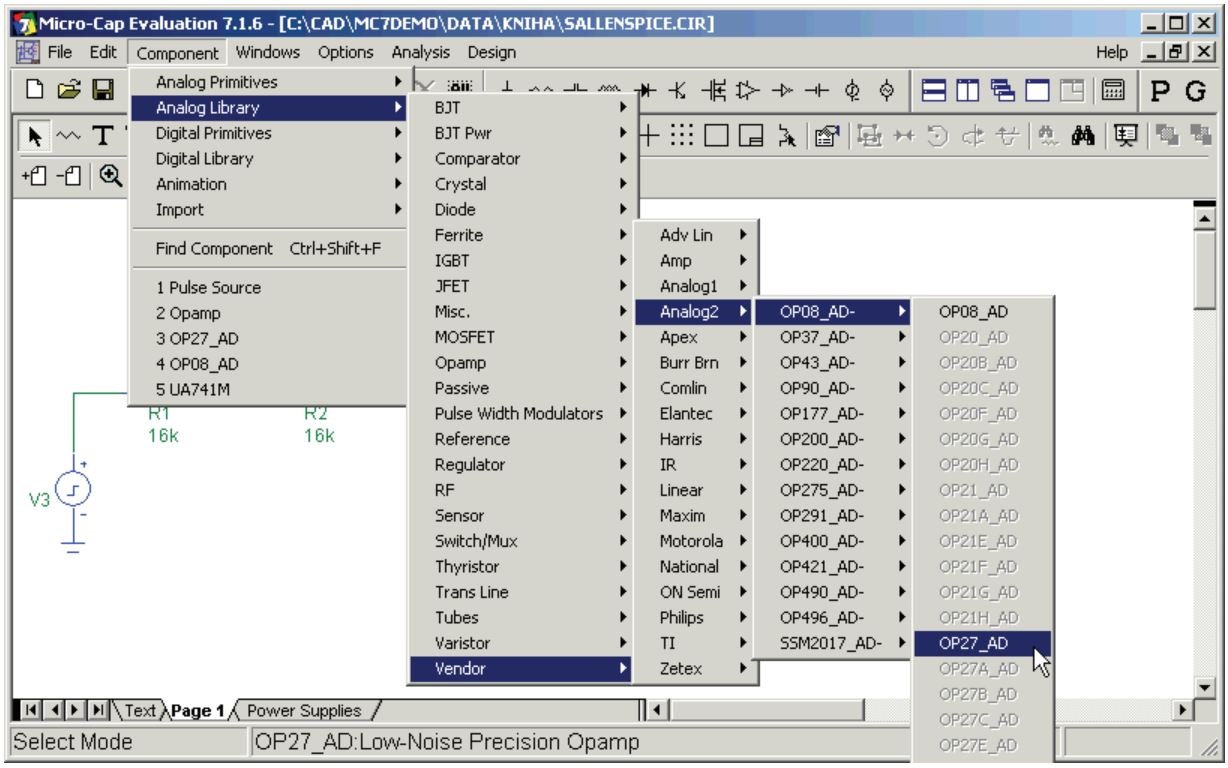
S modely SPICE můžeme v MicroCapu pracovat těmito základními způsoby:

1. Využíváním součástek, jejichž modely jsou v knihovnách nebo dalších textových souborech na disku ve formě podobvodů.
2. Konverzí dat ze schématického editoru do vstupního souboru SPICE. Ten je pak možno využít k simulaci v SPICE programech a také v MicroCapu.
3. Načtením, příp. tvorbou SPICE textového souboru **(*.ckt)** a následnou simulací obvodu.

V následujícím textu se seznámíme se všemi způsoby. Informace o způsobech 2 a 3 budou pouze orientační, protože v této fázi studia nepředpokládáme hlubší znalost jazyka SPICE.

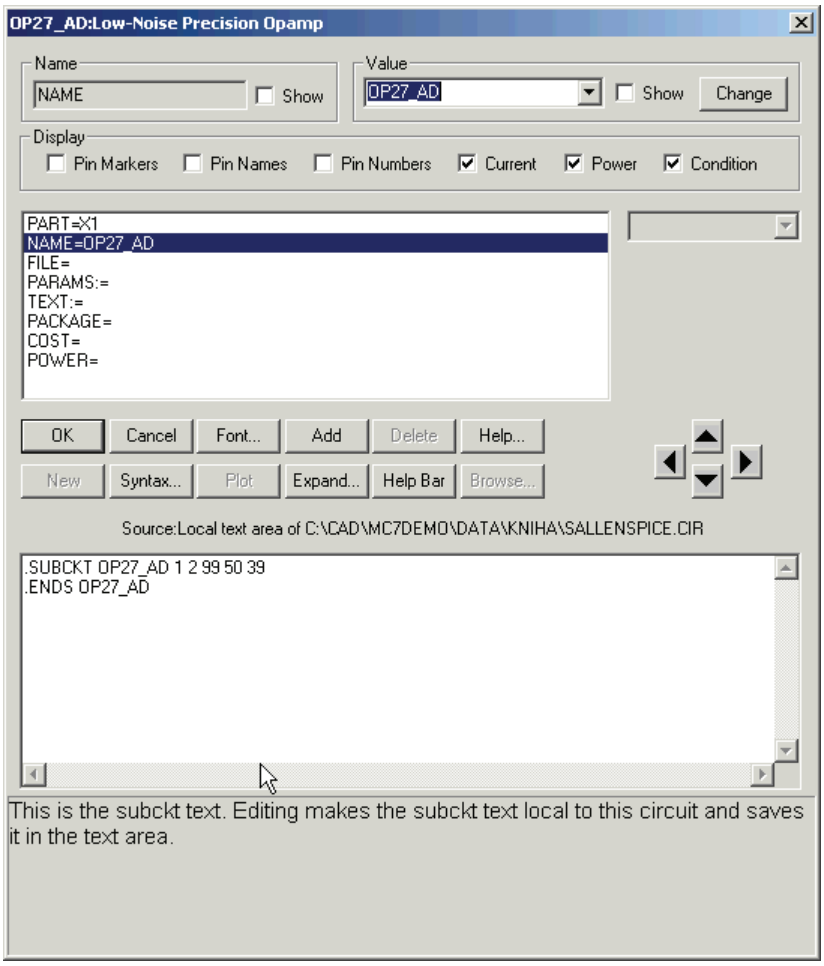
Využívání podobvodů SPICE

K následujícím experimentům využijeme soubor **SALLEN.CIR** z předchozí části 9.3.1. Po jeho načtení do editoru jej uložíme pod názvem **SALLENSPICE.CIR** („*File/Save As*“). V režimu „*Select*“ smažeme značku operačního zesilovače OP27. Místo ní umístíme součástku napojenou na model SPICE firmy Analog Devices *OP27_AD*. Nalezneme ji zanořenou do složky „*Vendor*“ (prodejce, dodavatel) podle **obr. 8.32**.



Obr. 8.32: Úplná cesta k SPICE modelu operačního zesilovače OP27 firmy Analog Devices.

Po umístění do správné pozice poklepeme na značku v režimu „*Select*“. Objeví se okno „*OP27_AD:Low-Noise Precision Opamp*“ – viz **obr. 8.33**. Z hlavičky textu v spodním podokně je ovšem zřejmé, že tady něco nefunguje. Je zde deklarace podobvodu SPICE (začíná příkazem *.SUBCKT*), ovšem následuje hned konec (*.ENDS*). Nad textovým polem je zobrazena informace, že model, který ovšem chybí, je umístěn lokálně ve vstupním souboru. Znamená to, že MicroCap nenalezl na disku knihovnu s modelem „*OP27_AD*“.



Obr. 8.33: Editační okno operačního zesilovače *OP27*.

Nejprve se podíváme, je-li na disku daná knihovna. Zvolíme

File/Open

a jako masku souborů vybereme

Spice library (.LIB)*

V adresáři "*Library*" nalezneme soubor **Op27.lib**. Otevřeme jej. Nalezneme hlavičku podobvodu:

```
* Node assignments
*                 non-inverting input
*                 |   inverting input
*                 |   |  positive supply
*                 |   |   |  negative supply
*                 |   |   |   |  output
*                 |   |   |   |  |
.SUBCKT OP27      1   2   99 50 39
```

Vidíme, že podobvod se jmenuje *OP*27, ale MicroCap hledal jiný název *OP*27*_AD*. Změníme tedy název *OP*27 na *OP*27*_AD*. Změnu uložíme kliknutím na ikonu  . Poté okno se souborem zavřeme kliknutím na  v druhé liště shora.

Nyní se ještě přesvědčíme, jestli je knihovna **OP27.LIB** zařazena do seznamu dostupných knihoven v souboru **NOM.LIB**. Zvolíme opět

File/Open

a s nastavenou maskou souborů

Spice library (.LIB)*

vybereme a otevřeme soubor **NOM.LIB**. Jeho obsah je standardně takovýto:

```
.lib "small.lbr"
.lib "digio.lib"
.lib "digdemo.lib"
.lib "smps_cb.lib"
.lib "tube.lib"
.lib "cqlib.lib"
.lib "japan.lbr"
```

Soubor **OP27.lib** zde není. Do posledního řádku tedy zapíšeme

.lib "OP27.lib"

uložíme a okno se souborem zavřeme.

Poklepeme-li nyní znovu na značku operačního zesilovače, zjistíme, že už došlo k napojení na knihovnu a že model podobvodu je překopírován do příslušného pole.

Prohlédněme si text, definující podobvod. Jeho část (s vynecháním mnoha řádků) je zde:

```
.SUBCKT OP27_AD     1 2 99 50 39
*
* INPUT STAGE   POLE AT 80 MHZ
*
R3   5  97    0.0619
….
Q1   5  2  7  QX
….
D1   2   1    DX
….
* MODELS USED
*
.MODEL QX NPN(BF=50E6)
.MODEL DX   D(IS=1E-15)
….
.ENDS
```

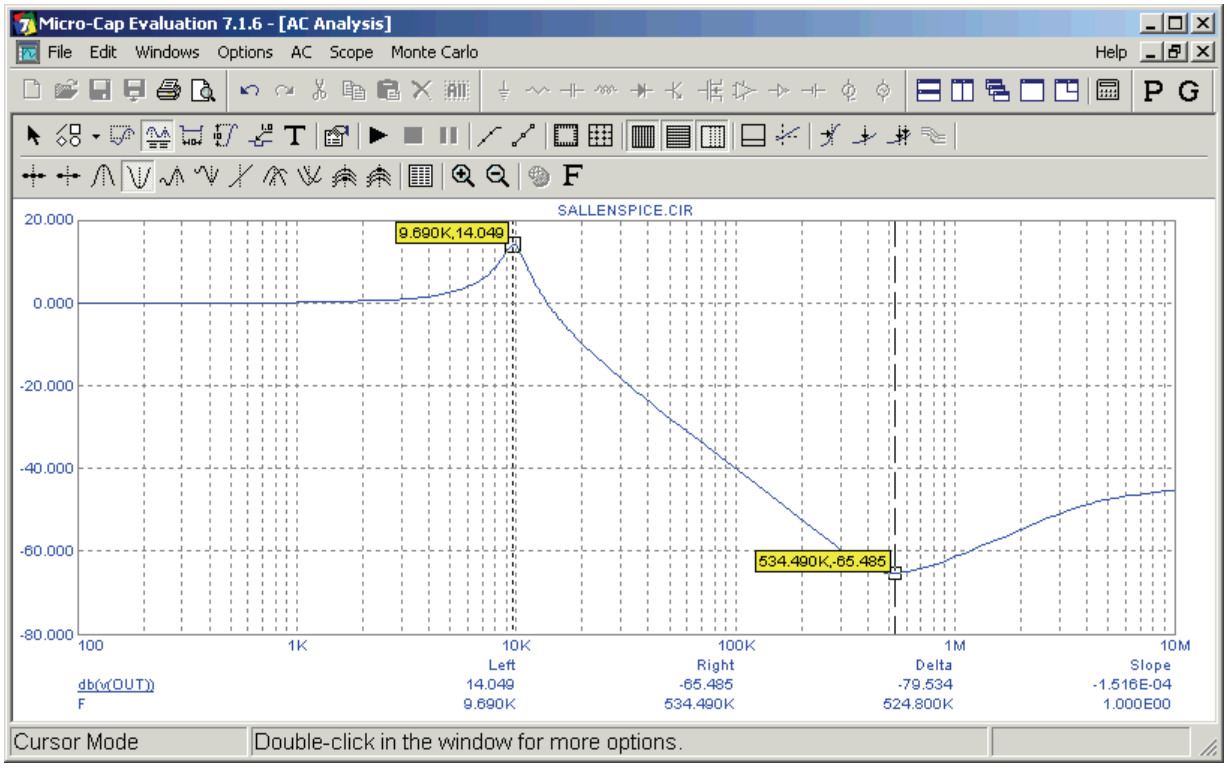
První řádek iniciuje podobvod s názvem *OP27_AD*, který je s okolím spojen pěti vývody. Jejich čísla 1 2 99 50 39 odpovídají číslům pěti vnitřních uzlů podobvodu. Jedná se o vývody operačního zesilovače v tomto pořadí: neinvertující vstup, invertující vstup, kladné napájecí napětí, záporné napájecí napětí, výstup (toto pořadí je definováno v „*Component Editoru*“).

Jednotlivé řádky mezi příkazy *.SUBCKT* a *.ENDS* definují součástky, zapojené k vnitřním uzlům modelu, a na posledních dvou vnitřních řádcích jsou příkazy *.MODEL* pro definici modelu *QX* bipolárního tranzistoru *NPN* a *DX* diody. Model diody *DX* se vyznačuje saturačním proudem $IS=10^{-15}$ A, ostatní parametry diody jsou rovny implicitním hodnotám. Podobně všechny tranzistory, které budou mít přiřazen model *QX*, budou mít parametr „*Beta Forward*“ (něco jako proudový zesilovací činitel h_{21e}) 50.10^6, ostatní parametry implicitní.

Na řádku, který následuje za poznámkami (řádky začínající hvězdičkou), je definován rezistor *R*3, který je zapojen mezi uzly 5 a 97 a jeho odpor je 0,0619 Ω. Následuje definice tranzistoru *Q*1, který je zapojen mezi uzly 5, 2 a 7 (kolektor, báze, emitor) a jeho model je *QX*, a diody *D*1, která je mezi uzly 2 a 1 (anoda a katoda) a její model je *DX*.

Na podobvod se můžeme dívat jako na podprogram ve vyšším programovacím jazyku. Hierarchie simulovaného obvodu může být i víceúrovňová, kdy se z podobvodu můžeme odvolávat na podobvody nižší úrovně. Můžeme si všimnout, že v některých případech vypadají modely „podivně“, mohou obsahovat součástky s nereálnými hodnotami. Jde však o matematické modelování s jediným efektem, totiž aby se modelovaný obvod jako celek choval „reálně“.

Kontrolní „*AC*“ analýzou obvodu zjistíme, že kmitočtová charakteristika filtru je nyní oproti simulaci s modelem z „MicroCapovské“ knihovny mírně odlišná, zejména v nepropustném pásmu filtru v oblasti vyšších kmitočtů. Odlišnosti jsou zřejmé po srovnání **obr. 8.31** a **obr. 8.34**. Je jasné, že modely nejsou ekvivalentní. Je nejasné, kterému „více věřit“.Určité srovnání provedeme v navazující části „*Konverze do vstupního souboru SPICE*“.

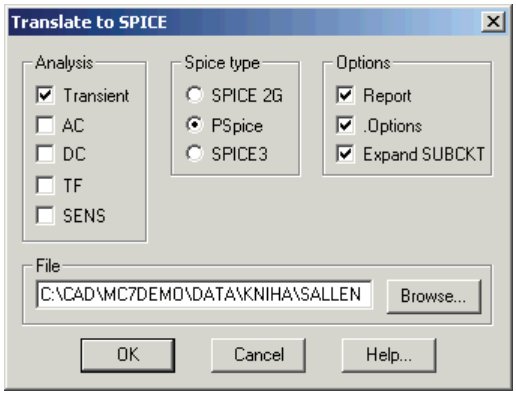


Obr. 8.34: Kmitočtová charakteristika filtru Sallen-Key z **obr. 8.28** se SPICE modelem operačního zesilovače. Srovnejte s charakteristikou na **obr. 8.31** („MicroCapovský“ model OZ).

Před dalšími experimenty je vhodné ještě jednou soubor **SALLENSPICE.CIR** uložit.

Konverze do vstupního souboru SPICE

Data ze vstupního souboru MicroCapu lze převést do vstupního souboru formátu SPICE. Provedeme-li převod ze souboru **SALLENSPICE.CIR**, pak se při konverzi využije beze změny podobvod *OP*27*_AD*, který je již modelován v jazyku SPICE. Převádíme-li data ze souboru **SALLEN.CIR**, kde operační zesilovač využívá „MicroCapovský“ model, pak MicroCap automaticky převede tento model na unifikovaný podobvod SPICE.



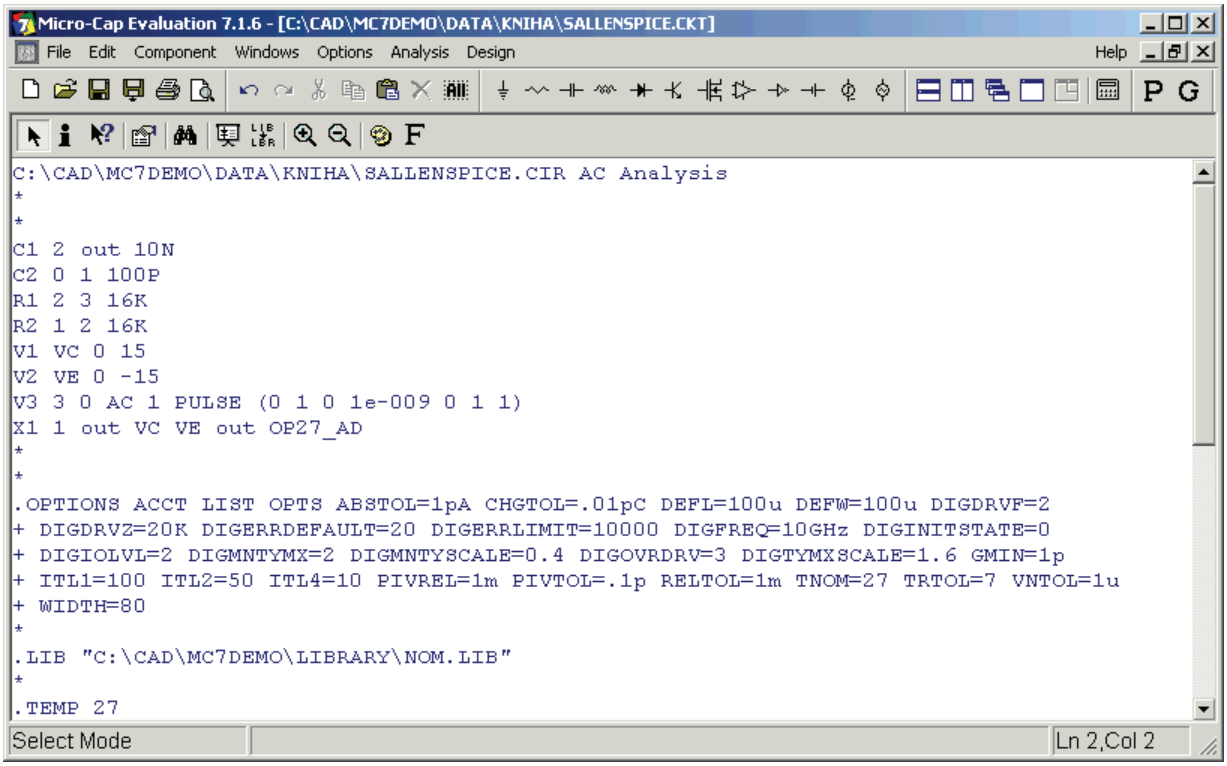
Nejprve provedeme převod souboru **SALLENSPICE.CIR**. Načteme jej do schématického editoru, pokud již není načten, a pak vybereme z roletového menu položky „*File/Translate/Schematic to SPICE Text File*“. Objeví se okno podle **obr. 8.35**.

Obr. 8.35: Okno převodu vstupního souboru do formátu SPICE.

Z okna je zřejmé, že do SPICE souboru lze převést nejen data o modelu celého obvodu (v jednom ze tří možných SPICE formátů), ale i příkazy pro typ analýzy („*Analysis*“) a podmínky pro běh simulátoru („*Options*“). Položky v „*Options*“ mají následující význam:

- **Report** – na konec souboru se připojí ve formě poznámek výčet prvků, obsažených v modelu obvodu.
- **.Options** – do souboru se umístí příkaz, který bude obsahovat všechny tzv. globální podmínky simulace, do nichž je program momentálně nastaven. Tím je zaručeno, že simulace bude probíhat za stejných podmínek.
- **Expand SUBCKT** – do souboru se překopíruje celý zdrojový text použitého podobvodu. Je to výhodné, jestliže soubor SPICE využijeme v prostředí, kde nebude daný podobvod k dispozici.

Deaktivujeme položku “*Transient*”, aktivujeme “*AC*”, deaktivujeme položku „*Expand SUBCKT*“ a potvrdíme („*OK*“). Na disku se vygeneruje SPICE soubor **SALLENSPICE.CKT** a jeho obsah se zobrazí na ploše:



Obr. 8.36: Zobrazení obsahu vygenerovaného souboru **SALLENSPICE.CKT**.

První řádek SPICE souboru musí povinně obsahovat tzv. nadpis, který má jen popisný význam. MicroCap zde vygeneroval název zdrojového vstupního souboru včetně úplné cesty k němu a typ analýzy, který jsme zvolili.

Následuje netlist obvodu. Součástka *V*3 je zdroj impulsů, avšak modelovaný příkazem SPICE, který má odlišný formát od „MicroCapovského“. Text *AC* 1 znamená, že v režimu „*AC*“ analýzy bude tento zdroj nahrazen zdrojem harmonického signálu o amplitudě 1 V. Součástka *X*1 je operační zesilovač, se zapojením vývodů: [*neinvertující vstup, invertující vstup, kladné napájení, záporné napájení, výstup*] v uzlech [1 *out VC VE out*]. Protože jsme nezatrhli položku „*Expand SUBCKT*“, vlastní podobvod zde není uveden, pouze zprostředkovaný odkaz na něj přes knihovnu **NOM.LIB** (již víme, že jde o textový soubor s výčtem všech knihoven, které má simulátor k dispozici, je tam tedy i knihovna **OP27.LIB** s podobvodem *OP*27*_AD*).

V příkaze *.OPTIONS* jsou uvedeny globální podmínky smulace. Prohlédneme-li si zbytek souboru, uvidíme toto:

```
.TEMP 27
.AC DEC 10 100 1e+007
.PRINT AC VDB([OUT])
.PLOT AC VDB([OUT]) -80,20
.PROBE
***  Parts Count
** Battery         2
** Resistor        20
** Inductor        3
** Capacitor       9
** Diode           18
** NPN             2
** ISpice          2
** Pulse source     1
** VSpice          10
** Subckt          1
** GIofV           13
** FIofI           2
** EVofV           6
.END
```

Za příkazem definujícím teplotu 27°C následuje příkaz pro provedení „*AC*“ analýzy. Řetězec „*DEC* 10“ znamená, že při analýze je kmitočet rozmítán logaritmicky po dekádách při 10 bodech na jednu dekádu. Analýza proběhne od kmitočtu 100 Hz do kmitočtu 10 MHz.

Následují příkazy *.PRINT* a *.PLOT* pro generování výsledků analýzy (napětí uzlu *OUT* v decibelech) do tabulky hodnot a do grafu. Příkazem *.PROBE* se spustí grafický postprocesor programu SPICE. Pokud bychom využívali vygenerovaného souboru k simulaci v MicroCapu, jsou příkazy *.PRINT* a *.PROBE* ignorovány. Data v příkazech *.AC* a *.PLOT* se nastaví do okna „*AC Analysis Limits*“.

Ve formě poznámek je uveden výčet jednotlivých součástek v obvodu.

Provedeme kontrolní analýzu „*AC*“. Zjistíme, že nastavené rozmítání kmitočtové osy je nedostatečně jemné a že výsledná kmitočtová charakteristika je vykreslena příliš hrubě. Můžeme to napravit v okně „*AC Analysis Limits*“ překonfigurováním některých položek

(„*Frequency Step*“ – auto, „*Maximum Change*“ – 1), čímž eliminujeme vliv nevhodně nastaveného příkazu *.AC* v SPICE souboru.

Horkou klávesou *F*3 se vrátíme na plochu se SPICE souborem a kliknutím na spodnější ikonu ✕ okno se souborem zavřeme.

Nyní do editoru načteme soubor **SALLEN.CIR** s „MicroCapovským“ modelem operačního zesilovače a provedeme za stejných podmínek jeho konverzi do SPICE souboru **SALLEN.CKT**. Prohlédneme si jeho obsah. I když jsme neaktivovali volbu „*Expand SUBCKT*“, text podobvodu se vygeneroval, protože SPICE nerozumí „MicroCapovským“ modelům. Kdybychom porovnali vygenerovaný podobvod s podobvodem *OP*27*_AD*, zjistili bychom, že způsoby modelování jsou odlišné.

Převodu souboru se schématem obvodu do souboru SPICE můžeme využít i k studiu struktury „MicroCapovských“ modelů. Doporučujeme následující experiment se souborem **SALLEN.CIR**: V „MicroCapovském“ modelu operačního zesilovače *OP*27 změníme obsah položky „LEVEL“ z 3 na 1. Z přílohy P4.3.3 vyplývá, že jsme nyní nastavili nejjednodušší úroveň modelování zesilovače – je to zdroj proudu řízený vstupním diferenčním napětím, pracující do odporové zátěže. Po vygenerování souboru SPICE zjistíme, že podobvod operačního zesilovače se podstatně zjednodušil:

```
* OPAMP
* PINS:  1=NC+ 2=NC- 3=VEE 4=VO 5=VCC
.SUBCKT OP_27 1 2 3 4 5
GOUT 0 4 1 2 14400
RIN 1 2 1G
ROUT 4 0 125
RSUPMIN  3 0 1
RSUPPLUS 5 0 1
.ENDS OP_27
```

Písmenem *G* se v jazyku SPICE označuje zdroj proudu řízený napětím. Zápis

GOUT 0 4 1 2 14400

znamená, že daný zdroj proudu se jmenuje *OUT*, že jeho svorky jsou zapojeny mezi uzly 0 a 4, jeho řídicí svorky jsou na uzlech 1 a 2 a přenosová vodivost, tj.

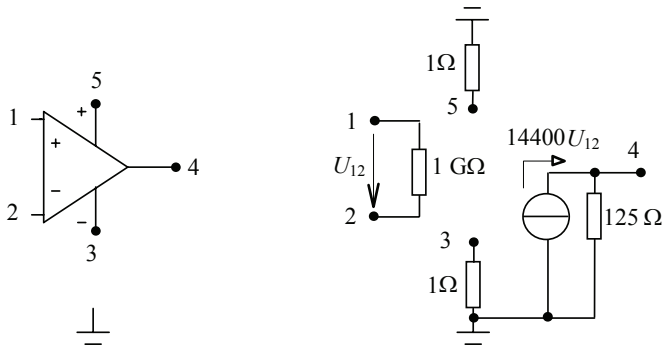
proud zdroje/řídicí napětí

je 14400 *S*.

Kromě toho jsou v modelu ještě 4 rezistory, modelující vstupní odpor (*RIN*) a výstupní odpor (*ROUT*) zesilovače, a pomocné rezistory *RSUPMIN* a *RSUPPLUS*. U tohoto jednoduchého lineárního modelu operačního zesilovače totiž nehrají žádnou úlohu napájecí napětí. Napájecí svorky není možné nechat nezapojeny, protože by vznikly izolované uzly a obvodová matice by obsahovala prázdné řádky, což by způsobilo kolaps výpočetních algoritmů. Proto jsou dané uzly uzemněny přes odpory 1 Ω. Náhradní schéma, odpovídající SPICE podobvodu, je na **obr. 8.37**. Stejnosměrné zesílení operačního zesilovače vychází podle Ohmova zákona

$$U_4/U_{12} = 14400.125=1800000,$$

což je katalogový údaj obvodu *OP*27.



Obr. 8.37: SPICE model odpovídající „MicroCapovskému“ modelu operačního zesilovače OP27 při použité úrovni modelování LEVEL=1.

Načtení vstupního souboru SPICE a následná simulace

V adresáři ..*MC7DEMO/Data* je SPICE soubor **TTLINV.CKT**. Otevřeme jej volbou „*File/Open*“, jestliže vybereme masku souborů „*SPICE (*.CKT; *.STM)*“.

```
ttlinv
*SPICE_NET
.MODEL D1 D RS=40 TT=0.1NS CJO=0.9PF
.MODEL QND NPN BF=50 RB=70 RC=40 CCS=2PF TF=0.1NS TR=10NS
+ CJE=0.9PF CJC=1.5PF PC=0.85 VA=50
Q2 3 2 7 QND
Q3 6 3 4 QND
Q5 10 13 5 QND
D1 4 5 D1
D2 10 9 D1
D3 9 0 D1
RB1 11 12 4K
RC3 11 6 100
RC2 11 3 1.4K
RE2 7 0 1K
VCC 11 0 5
VIN 8 0 PULSE 0 3.5 1NS 1NS 1NS 40NS
RS 8 1 50
Q4 5 7 0 QND
RB5 11 13 4K
Q1 2 12 1 QND
.TRAN 1e-009 100NS 0 0
.PLOT TRAN V(3) 6,0
.PLOT TRAN V(5) 4,0
.PRINT TRAN V(3) V(5)
.dc VIN 0 5 0.05
.TEMP 0
.PLOT DC V(3) 5.5,0.5
.PLOT DC V(5) 4,0
.PRINT DC V(3)
.PRINT DC V(5)
.END
```

Jedná se o model hradla *NAND* na tranzistorové úrovni, avšak ne zcela ekvivalentní tomu, s nímž jsme pracovali v části 9.2.1 (soubor **TTLINV.CIR**). Je zde několik odlišností. Model je rozšířen o další tranzistorový obvod, modelovaný součástkami *Q*5, *D*2, *D*3 a *RB*5. Napájecí napětí je *VCC*=5 V namísto původní velikosti 3,3 V. Modely diod a tranzistorů jsou odlišné.

Pokuste se v MicroCapu provést analýzu „*Transient*“ a zobrazit průběh napětí na uzlech č. 3 a 5, které odpovídají uzlům č. 1 a 4 u původního obvodu *TTLINV.CIR* na **obr. 8.2**. Dále můžete v SPICE souboru vyblokovat zapsáním znaku * na začátek příslušného řádku definice prvků *Q*5, *D*2, *D*3 a *RB*5, čímž se přiblížíme k zjednodušenému modelu ze souboru **TTLINV.CIR**, a analýzu provést znovu. Ve vlastním zájmu však změny do souboru **TTLINV.CKT** neukládejte.

8.3.3 Začleňování modelů prvků do vstupních souborů

Pošleme-li například kolegovi emailem vlastní vstupní soubor s modelem obvodu připraveným k analýze, nemusíme mít jistotu, že kolega pracuje se stejnými knihovnami prvků jako my. Může mu chybět zrovna model, který jsme použili. V části 7 bylo naznačeno, že v MicroCapu verze 7 je tato „*File Portability*“ velmi dobře řešena. Do vstupního souboru se automaticky kopírují schématické značky součástek, obsažených v obvodu. Kromě toho máme možnost uložit zde i „MicroCapovské“ a SPICE modely těchto součástek, čímž je zabezpečena „soběstačnost“ vstupního souboru a bezproblémová analýza bez ohledu na typ knihoven.

Vše objasníme konkrétním experimentem. V MicroCapu založíme nový soubor volbou „*File/New/Schematic*“. Na pracovní plochu umístíme značku operačního zesilovače *UA*741*M*, jehož „MicroCapovský“ model vybereme buď v nabídce

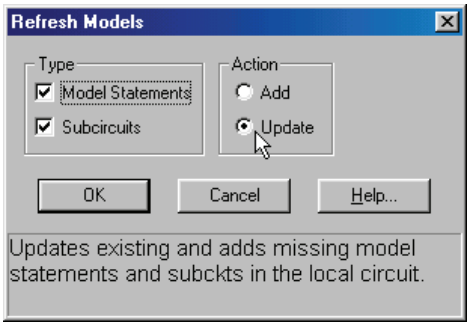
Component/Analog Primitives/Active Devices/Opamp

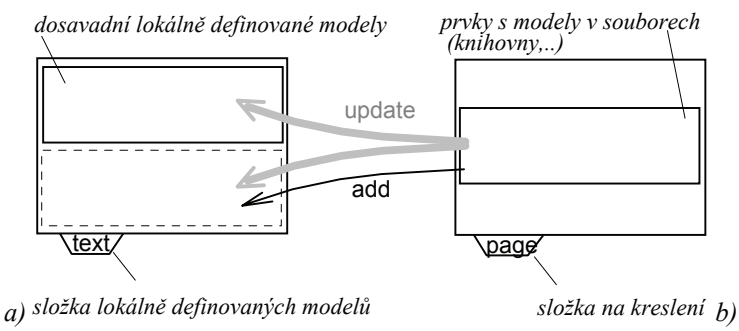
a v okně „*Opamp*“ zvolíme model *UA*741*M*, nebo můžeme jít přímo do „analogové knihovny“

*Component/Analog Library/Opamp/UA*0000*/UA*741*M.*

Po umístění na plochu se přesvědčíme, že do složky „*Text*“ se nepřekopírovala žádná informace o modelu součástky. Model je totiž pro simulátor k dispozici na disku v knihovně **SMALL.LBR**.

Chceme-li tento model začlenit do vstupního souboru, klikneme na ikonu  („*Refresh Model*“). Objeví se okno podle **obr. 8.38** a):





a) b)

Obr. 8.38: a) Okno „Refresh Models“, b) mechanismy akcí „*Add*“ a „*Update*“.

Položky v části „*Action*“ definují, co se bude dít s položkami v části „*Type*“. Vše je přehledně znázorněno na **obr. 8.38** b).

Vše, co se „dostane“ do složky „*Text*“, se stane součástí vstupního souboru. Je to tedy složka dat, které platí lokálně pro simulaci obvodu bez nutných vazeb na vnější knihovny. V našem případě je tato složka prázdná. Obecně jsou v této složce v každém okamžiku lokálně definované modely a v složce „*Page*“ prvky, jejichž modely mají napojení na knihovny nebo jiné soubory na disku.

- Akce „*Add*“ způsobí, že MicroCap překopíruje do složky „*Text*“ modely těch prvků ze složky „*Page*“, které dosud nebyly lokálně definovány. Dosavadní lokálně definované modely zůstanou nedotčeny.
- Akce „*Update*“ způsobí, že MicroCap překopíruje do složky „*Text*“ modely těch prvků ze složky „*Page*“, které dosud nebyly lokálně definovány. Dosavadní lokálně definované modely budou aktualizovány z knihoven nebo jiných souborů (pokud tyto budou nalezeny).
- V sekci „*Type*“ můžeme navolit, zda se daná akce má týkat příkazů *.MODEL*, podobvodů SPICE, nebo obojího.

Z uvedeného je zřejmé, že v naší situaci, kdy pole „*Text*“ je zatím prázdné, jsou akce „*Add*“ a „*Update*“ rovnocenné. Ujistíme se, že je zatrženo „*Model Statement*“ (na stavu položky „*Subcircuit*“ nezáleží) a potvrdíme „*OK*“. Zkontrolujeme, že v složce „*Text*“ se objevil následující text:

```
*** From file C:\CAD\MC7DEMO\LIBRARY\SMALL.LBR
*** General purpose operational amplifier
.MODEL UA741M OPA (LEVEL=3 ROUTAC=50 ROUTDC=75 IOFF=20N IBIAS=80N VPS=13.4
+ VNS=-13.4 CMRR=31.6228K PD=50M IOSC=25M)
```

Můžeme se přesvědčit, že v příkazu *.MODEL* jsou uvedeny jen ty parametry operačního zesilovače, které se liší od implicitních hodnot.

Nyní se vraťme do složky „*Page*1“ a na plochu přidejme zdroj impulsů („*Pulse Source*“), jehož model nazvěme „*skok*“ a přidělíme mu parametry

VZERO=0*, VONE*=1*, P*1=0*, P*2=1*n, P*3=*1, P*4=1*, P*5=1.

Podíváme-li se pak do složky „*Text*“, zjistíme, že tam přibyl automaticky, bez nutnosti aktivace režimu „*Refresh Model*“, příkaz

```
.MODEL SKOK PUL (VONE=1 P1=0 P2=1n P3=1 P4=1 P5=1)
```

Zdroj impulsů totiž není v žádné knihovně, je tedy automaticky definován lokálně.

Nyní zkusíme modifikovat model operačního zesilovače, a to přímo v složce „*Text*“. Změníme parametr „*LEVEL*“ z čísla 3 na 1. Pak se přesuneme do složky „*Page*1“ a v režimu „*Select*“ poklepeme na značku operačního zesilovače. Zjistíme, že změna parametru „*LEVEL*“ se promítla i do editačního okna. V tomto smyslu jsou parametry modelu v obou složkách spolu provázány, takže jsme změnu parametru mohli provést stejně dobře v editačním okně. Okno opět zavřeme.

Nyní na plochu umístíme ještě operační zesilovač *OP*27 volbou

*Component/Analog Library/Vendor/Analog*2*/OP*08*-AD/OP*27*_AD,*

který je, jak už víme, modelován podobvodem SPICE z knihovny **OP27.LIB**.

Shrneme nastalou situaci a zkonfrontujeme ji s **obr. 8.38** b). V složce „*Page*1“ jsou 3 součástky s různými modely. V složce „*Text*“ jsou lokálně definované modely dvou

součástek, operačního zesilovače a zdroje impulsů, z toho model zesilovače je upraven oproti jeho originálu z knihovny, model zdroje v knihovně není. Model *OP*27*_AD* lokálně definován není.

Nyní budeme chtít dodefinovat lokálně model *OP*27*_AD*, aby byl k dispozici přímo ve vstupním souboru pro jiné uživatele, kteří tento model jinak nemají k dispozici. Klikneme na ikonu  („*Refresh Model*“). Aktivujeme „*Subcircuit*“ (na volbě položky „*Model Statement*“ nezáleží) a jako „*Action*“ zvolíme „*Add*“. Výsledek bude takový, že do složky „*Text*“ se k dvěma stávajícím příkazům, které zůstanou nezměněny, přikopíruje model podobvodu *OP*27*_AD*. Přesvědčte se o tom.

Nyní zopakujeme akci „*Refresh Model*“ při aktivaci položek „*Model Statement*“, „*Subcircuit*“ a „*Update*“. Příkaz *.MODEL* pro operační zesilovač *UA*741*M* se aktualizoval z knihovny **SMALL.LBR**, takže parametr „*LEVEL*“ se opět změnil na 3.

8.3.4 Práce s příkazem *.DEFINE* a s „*Formula Textem*“

Příkazem *.DEFINE* můžeme nebývalým způsobem rozšířit možnosti klasické simulace obvodů. Ukážeme jedno z typických využití: Definici tzv. symbolických proměnných a způsob práce s nimi.

K výkladu využijeme obvodu ze souboru **SALLEN.CIR**.

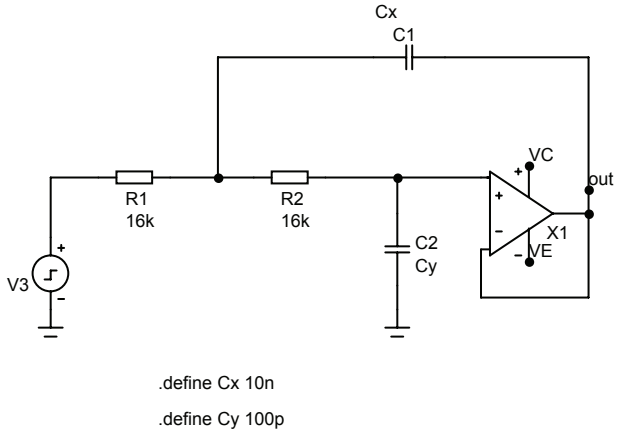
Symbolické proměnné a práce s nimi

Do editoru načteme soubor **SALLEN.CIR** a uložíme ho pod názvem **SALLENDEFINE.CIR**. V režimu „*Select*“ poklepeme na *C*1 a pak *C*2 a změníme položky „*Value*“ (původně 10*n* a 100*p*) na *Cx* a *Cy*. Poté v režimu „*Text*“ (klikneme na ikonu ) umístíme na plochu grid texty

.define Cx 10*n*
.define Cy 100*p*

Musíme přitom dodržet následující zásadu: Musí jít skutečně o dva objekty, nelze tedy po zapsání prvního textu přejít stisknutím *ENTER* na další řádek a zapsat druhý příkaz *.define*. Tvorba každého z příkazů tedy musí být ukončena potvrzením „*OK*“.

Výsledek je na **obr. 8.39**.



Obr. 8.39: Definice kapacit *C*1 a *C*2 symbolickými proměnnými *Cx* a *Cy*.

Velikosti kapacit *C*1 a *C*2 jsou nyní vyjádřeny symbolickými proměnnými *Cx* a *Cy*. Pomocí příkazů *.define* jsou těmto symbolickým proměnným přiřazeny číselné hodnoty. Výsledný efekt je tedy stejný, jako kdyby kapacitorům byly přiřazeny kapacity 10*nF* a 100*pF* přímo. Můžeme se o tom přesvědčit provedením analýzy „*AC*“ Dostaneme kmitočtovou charakteristiku, shodnou s výsledkem na **obr. 8.31**.

Na situaci se nic nezmění, když na plochu přidáme text

.define a 1

a definice *Cx* a *Cy* změníme takto:

.define Cx 10*n*a*
.define Cy 100*p*a*

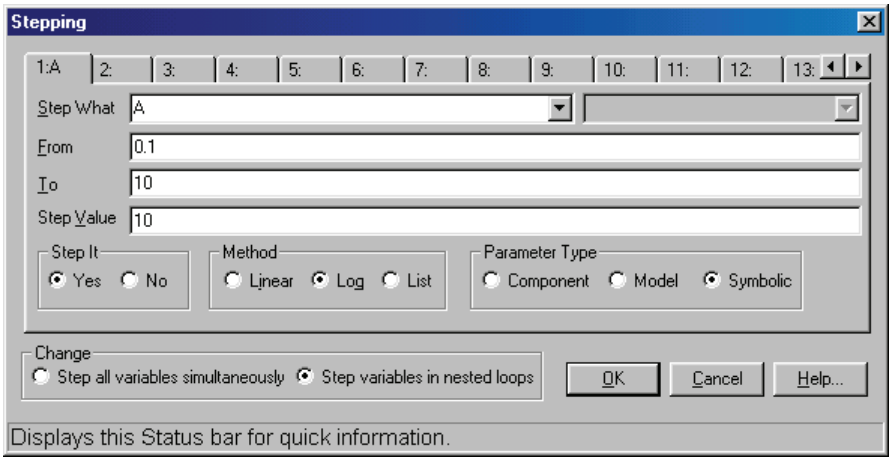
Zavedli jsme třetí symbolickou proměnnou *a*. Touto proměnnou násobíme obě kapacity. Zde je vhodné poznamenat, že umístění textů na plochu může být zcela libovolné, vyhodnocování příkazů se děje zcela nezávisle na jejich „poloze“.

Symbolickou proměnnou *a* nyní můžeme krokovat, čímž vlastně budeme současně krokovat kapacity *Cx* a *Cy*. Zkusme opět analyzovat kmitočtovou charakteristiku, ovšem pro sadu kapacit [*Cx Cy*] = [1*n* 10*p*], [10*n* 100*p*], [100*n* 1*n*]. Stačí tedy krokovat parametr *a* v hodnotách 0,1 1 10.

Zvolme

Analysis/AC

a v okně „*AC Analysis Limits*“ klikneme na „*Stepping*“. V části „*Parameter Type*“ navolíme „*Symbolic*“. Rozbalíme nabídku v řádku „*Step What*“ kliknutím na . Objeví se nabídka všech tří symbolických proměnných: *A*, *Rx*, *Ry*. Vybereme *A*. Vyplníme ostatní položky podle **obr. 8.40** a stlačíme *F*2 (horká klávesa pro zahájení simulace).



Obr. 8.40: Okno pro editaci podmínek krokování („*Stepping*“).

Získáme tři kmitočtové charakteristiky s rezonančními vrcholy na kmitočtech přibližně (1 10 100) *k*Hz podle **obr. 8.41**.

Z obrázku vyplývá, že souběžná změna kapacit vyvolává efekt ladění filtru beze změny tvaru kmitočtové charakteristiky, nebo jinými slovy, dochází k posuvu kmitočtu ω_0 při zachování činitele jakosti *Q*.

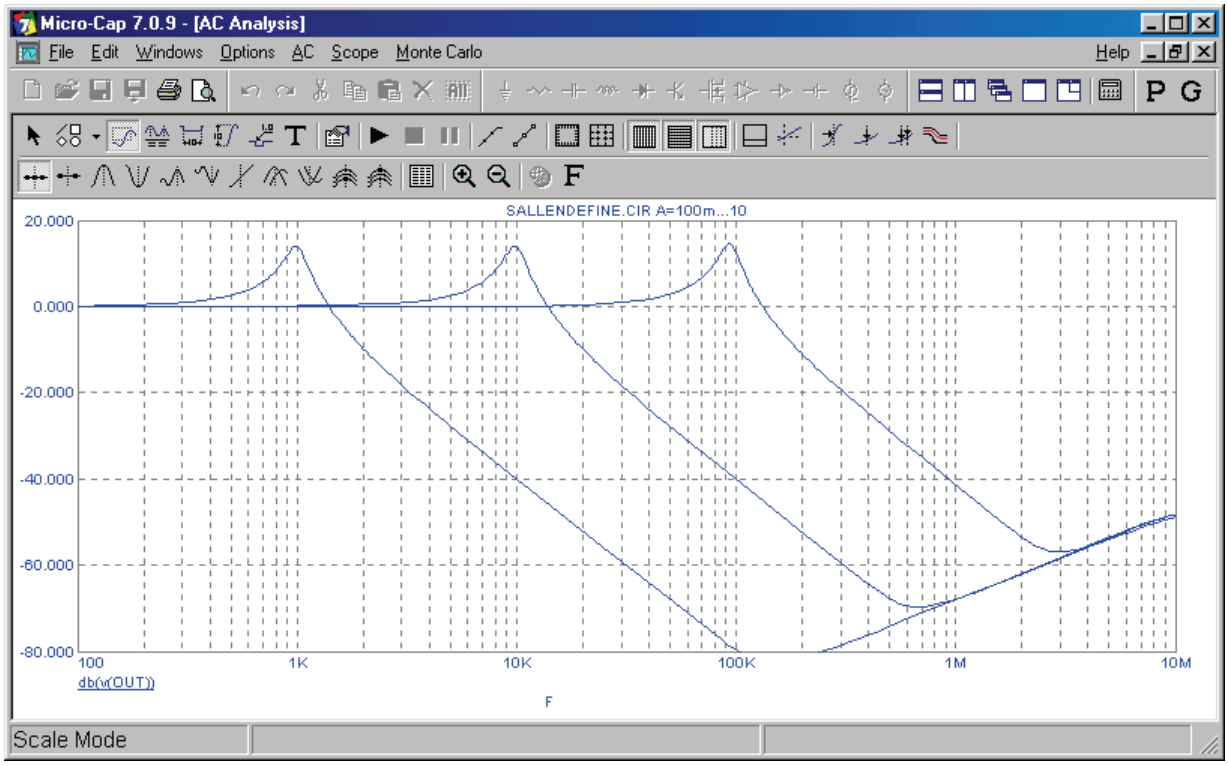
Stisknutím *F*3 se vrátíme do schématického editoru.

Pokud uvažujeme ideální operační zesilovač, pak filtr na **obr. 8.39** představuje obvod 2. řádu, jehož parametry ω_0, resp. $f_0 = \omega_0/(2\pi)$, a Q jsou dány těmito vzorci (viz například výstupy programu SNAP):

$$f_0 = \frac{1}{2\pi\sqrt{R_1 R_2 C_X C_Y}} = \left|R_1 = R_2 = R\right| = \frac{1}{2\pi R\sqrt{C_X C_Y}},$$

$$Q = \frac{\sqrt{R_1 R_2}}{R_1 + R_2}\sqrt{\frac{C_X}{C_Y}} = \left|R_1 = R_2\right| = \frac{1}{2}\sqrt{\frac{C_X}{C_Y}}.$$

Z vzorečků skutečně vyplývá, že pokud zvětšíme C_X i C_Y a-krát, pak se f_0 zmenší a-krát a Q zůstane nezměněno.



Obr. 8.41: Simulace přelaďování filtru *Sallen-Key* souběžnými změnami kapacit.

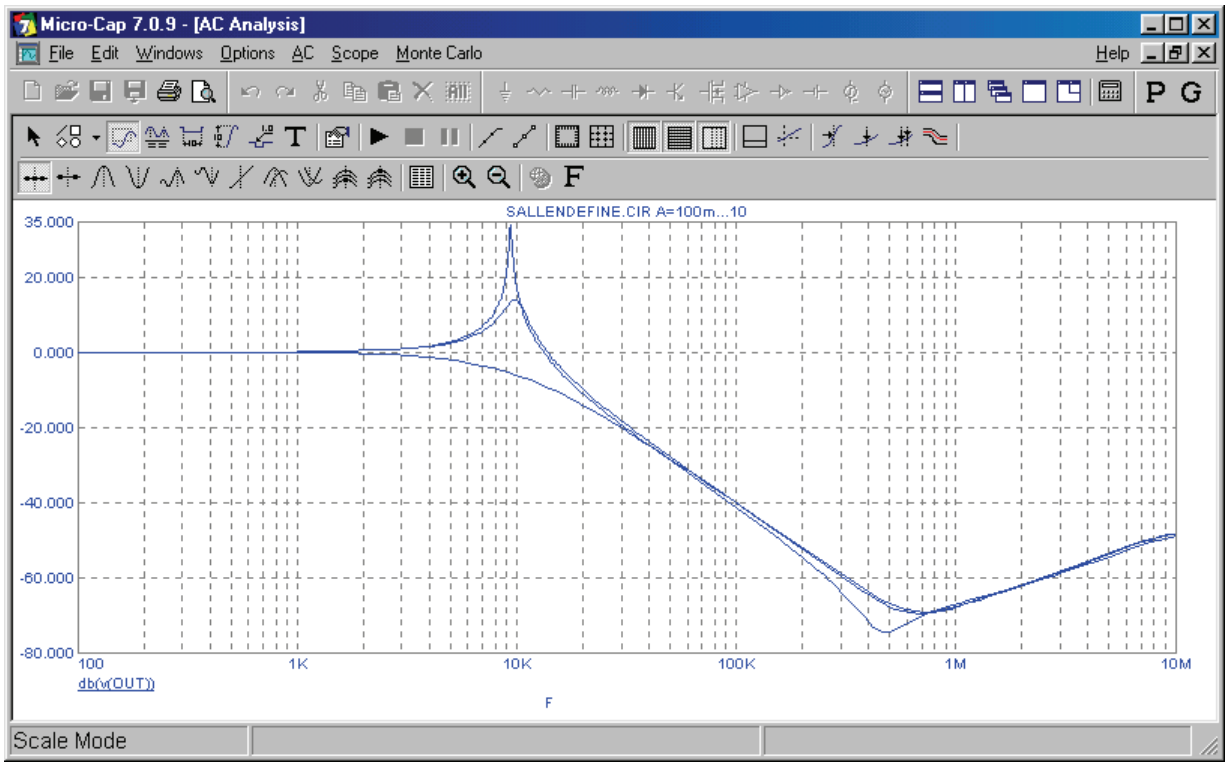
Obdobně lze vysledovat, že zvětšíme-li C_X a-krát při současném zmenšení C_Y a-krát, pak bude f_0 konstantní, ale a-krát vzroste činitel jakosti. Můžeme si to vyzkoušet. Modifikujeme příkaz *.define* pro C_Y:

.define Cy 100*p/a*

Po vykonání „*AC*“ analýzy a úpravě měřítka na decibelové stupnici získáme tři charakteristiky podle **obr. 8.42**.

Opět se vrátíme do schématického editoru stlačením *F*3.

Další možnosti práce se symbolickými proměnnými můžeme nastudovat z dokumentace programu. Je jich skutečně hodně… .



Obr. 8.42: Simulace změn činitele jakosti filtru *Sallen-Key*.

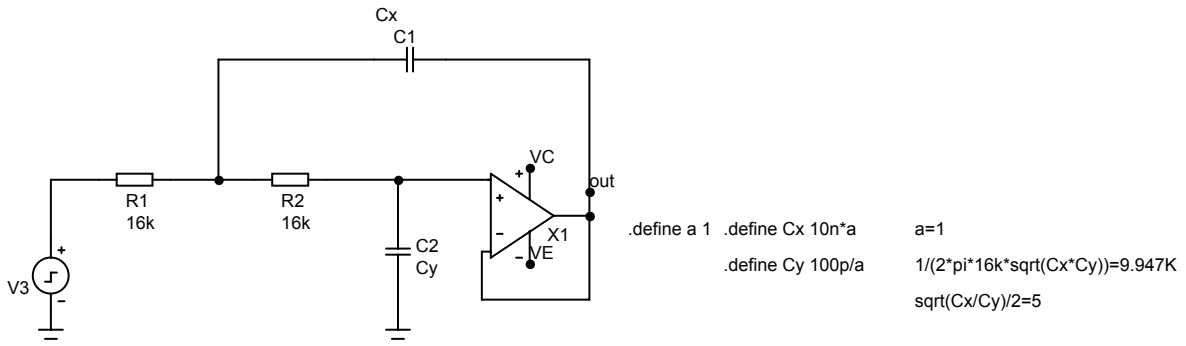
„Formula Text“

Práci se symbolickými proměnnými lze vhodně skloubit s tzv. „*Formula Textem*“. Tento text slouží k okamžitým vyčíslováním vzorců, v nichž figurují symbolické proměnné, přímo na pracovní ploše editoru.

Položme na kreslicí plochu tyto tři nezávislé grid texty:

=a
*=1/(2*pi*16k*sqrt(Cx*Cy))*
=sqrt(Cx/Cy)/2

První „*Formula Text*“ slouží k vizualizaci aktuální hodnoty proměnné *a*, druhý ukáže velikost kmitočtu f_0 a třetí činitel jakosti *Q*. „*Formula Text*“ začíná vždy rovnítkem. Ve skutečnosti se na ploše „projevuje“ zápisem **vzorec = číselná hodnota**, jak je zřejmé i z **obr. 8.43**.



Obr. 8.43: Způsob práce s „*Formula Textem*“.

Zkuste nyní znovu spustit „*AC*“ analýzu s krokováním parametru *a*. Pak se vraťte pomocí *F*3 zpět do schématu. Hodnoty „*Formula Textů*“ se nyní změnily, protože v paměti jsou platná data z posledního simulačního běhu:

```
.define a 1  .define Cx 10n*a        a=10
             .define Cy 100p/a       1/(2*pi*16k*sqrt(Cx*Cy))=9.947K
                                     sqrt(Cx/Cy)/2=5
```

Všimněte si, že činitel jakosti je však vypočítaný špatně! Správně má totiž vyjít při $a = 10$

$$Cx = 100\ n\text{F},\ Cy = 10\ p\text{F a } Q = sqrt(100n/10p)/2 = 50.$$

Kde je chyba?

Grid texty typu *.define* a „*Formula*“ pracují na principu „postupného dosazování“. V dokumentaci programu je uveden tento příklad dvou příkazů *.define*:

```
.define a 4+c
.define b a*x
```

Tyto příkazy byly napsány s cílem výpočtu veličiny

$$b = (4+c)*x.$$

Ve skutečnosti však program vypočte

$$b = 4+c*x.$$

Program vyhodnocuje tyto příkazy prostým „dosazováním textu“. Text '*a*' je nahrazen textem

'4+*c*'.

Abychom se vyvarovali podobných chyb, je vhodné používat závorky při definici výrazů, které se používají v dalších definicích. Konkrétně pro uvedený příklad:

```
.define a (4+c)
.define b a*x
```

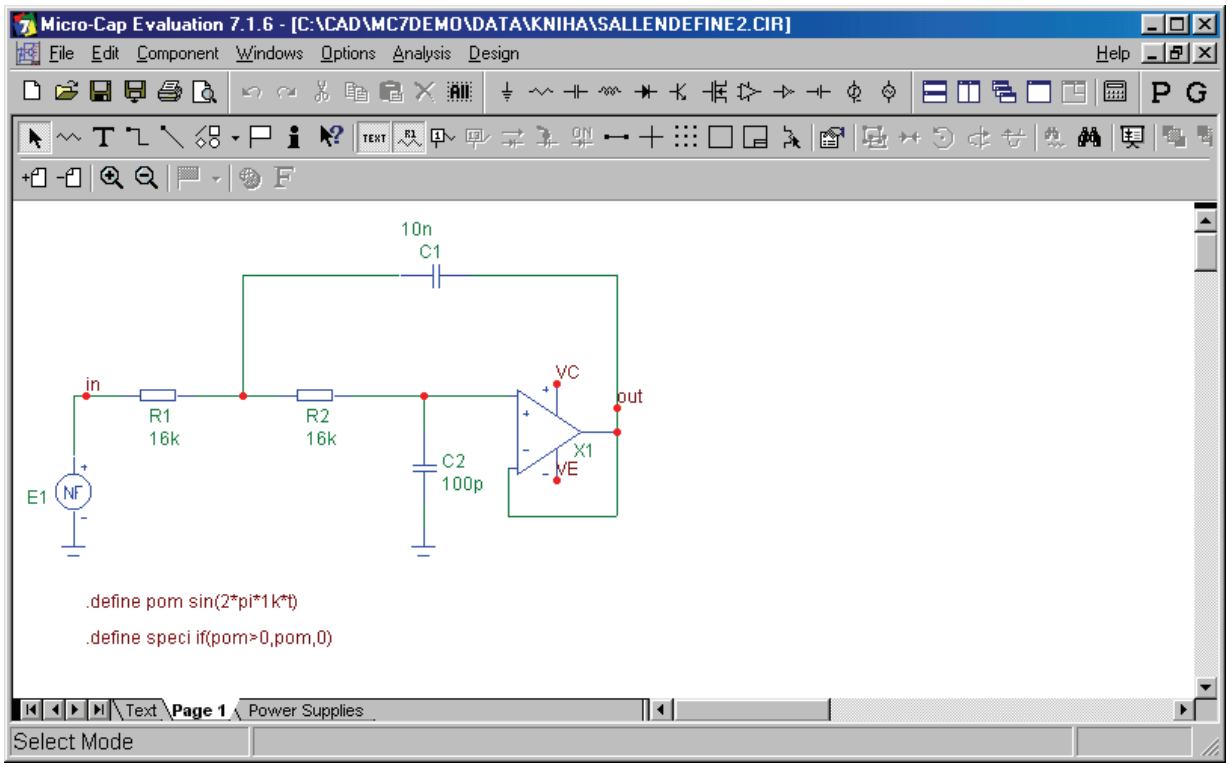
Nyní dostaneme správný výsledek (4+*c*)**x*.

V tomto smyslu opravíme naše definice *Cx* a *Cy* a po proběhnutí vícenásobné analýzy již dostaneme správné výsledky:

```
.define a 1  .define Cx (10n*a)      a=10
             .define Cy (100p/a)     1/(2*pi*16k*sqrt(Cx*Cy))=9.947K
                                     sqrt(Cx/Cy)/2=50
```

Vytváření uživatelských signálů pomocí symbolických proměnných

Do editoru načteme soubor **SALLEN.CIR** a přejmenujeme jej na **SALLENDEFINE2.CIR**. Pokusíme se nahradit vstupní signál filtru – jednotkový skok – jednocestně, případně dvoucestně usměrněným harmonickým signálem o kmitočtu 1 *k*Hz a amplitudě 1 V. Zdroje takovýchto signálu nejsou k dispozici v knihovnách. Ukážeme, jak si je jednoduše „vyrobit“.



Obr. 8.44: Způsob definice jednocestně usměrněného signálu generovaného funkčním zdrojem *NF*.

Smažeme značku zdroje impulsů a nahradíme ji značkou tzv. funkčního zdroje „*NFV*“ (zdroj napětí, velikost napětí lze definovat vzorcem). Tento zdroj nalezneme volbou

Component/Analog Primitives/Fubction Sources/NFV

Po umístění značky na plochu vyplníme do položky „*Value*“ libovolnou symbolickou proměnnou, např.

Speci

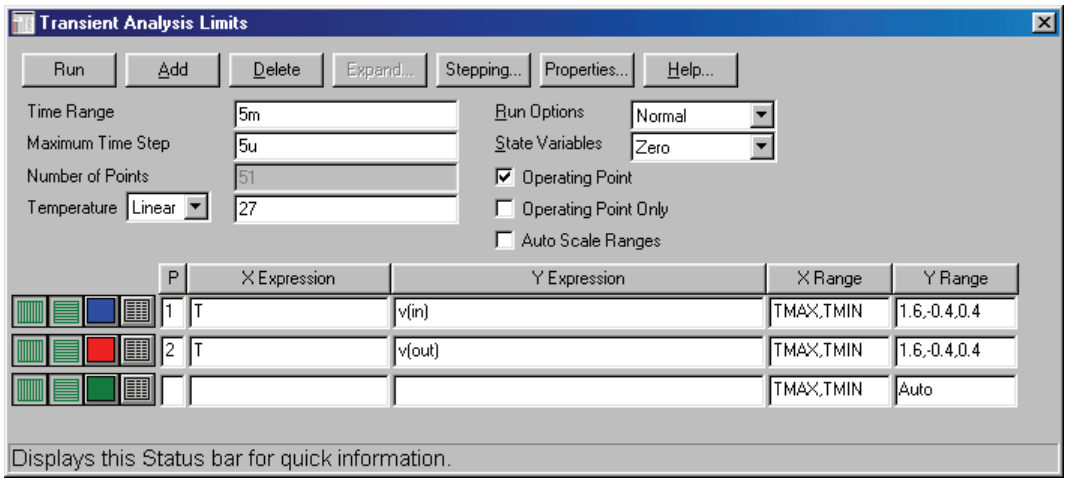
Pak zapíšeme grid text podle **obr. 8.44** a vstupní uzel opatříme jménem „*In*“.

První příkaz *.define* definuje signál „*pom*“ – harmonický signál o amplitudě 1 V a kmitočtu 1 *k*Hz.

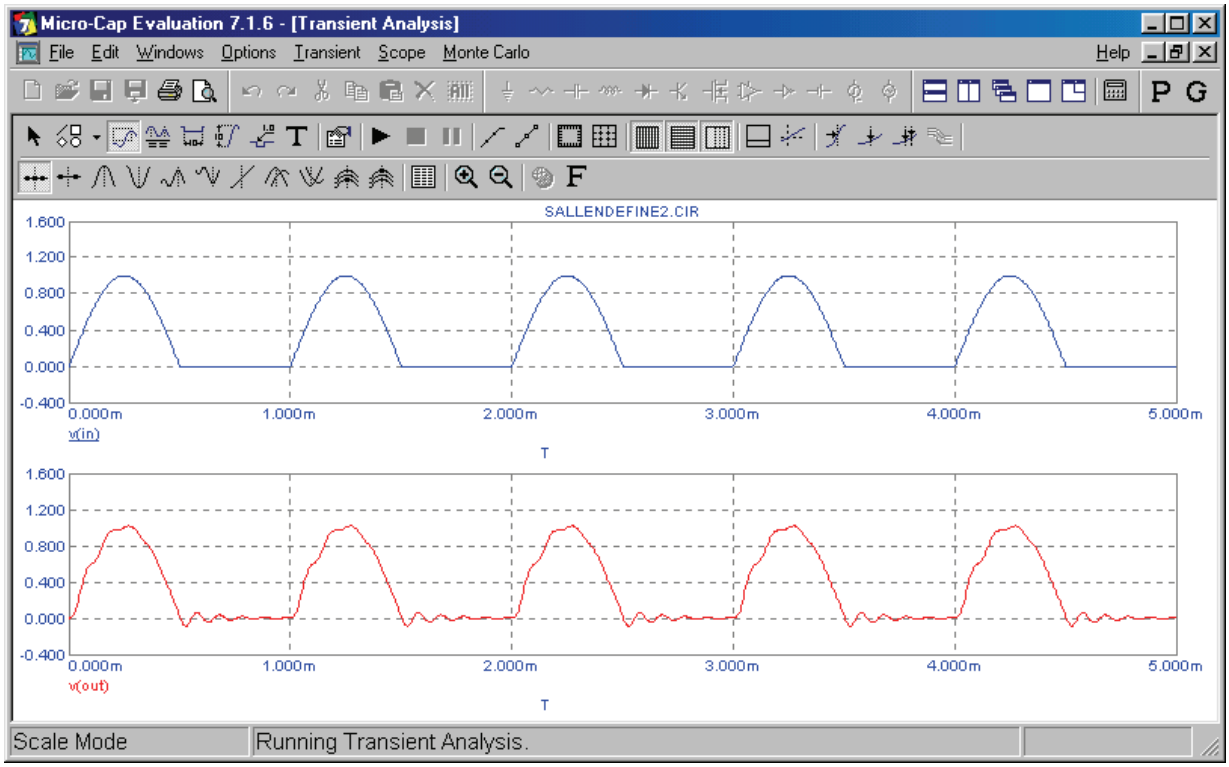
Druhý příkaz definuje signál funkčního zdroje „*speci*“. Využívá se zde funkce „*if*“: je-li pravdivá logická podmínka, pak má funkce hodnotu „*pom*“, jinak je nulová. Signál „*speci*“ je tedy jednocestně usměrněný harmonický signál.

Ověříme si funkčnost takového modelování. Spustíme analýzu „*Transient*“ („*Analysis/Transient*“), vyplníme okno „*Transient Analysis Limits*“ podle **obr. 8.45**, a spustíme analýzu. Výsledek je na **obr. 8.46**.

Protože obvod se chová jako filtr typu dolní propust s mezním kmitočtem okolo 10 *k*Hz (viz **obr. 8.34**), propouští všechny významné harmonické složky budicího signálu. Tomu odpovídají stejné tvary vstupního i výstupního signálu. Na kmitočtu 10 *k*Hz však kmitočtová charakteristika vykazuje značné rezonanční převýšení, což se projevuje na výstupu ve zvýrazněné 10. harmonické (viz zvlnění na **obr. 8.46**).



Obr. 8.45: Způsob vyplnění okna se vstupními podmínkami časové analýzy.



Obr. 8.46: Výsledky časové analýzy obvodu z **obr. 8.44**.

Modifikací příkazu *.define* pro signál „*speci*“ je možné dosáhnout různých časových průběhů. Například:

.define speci if(*pom*<0.8,*pom*,0.8) … sinusový s „oříznutými špičkami nad 0,8 V
.define speci if(*pom*>0,*pom*,-*pom*) … dvojcestně usměrněný signál

Pro dvojcestné usměrnění ovšem můžeme použít jednodušší zápis

.define speci abs(pom)

a pro jednocestné usměrnění

.define speci pom(pom*>0)*

(využívá se zde toho, že pokud je relační výraz pravdivý, jeho hodnota je 1, jinak 0).

Zkuste si výše uvedené ověřit v režimu analýzy „*Transient*“.

8.4 Shrnutí kapitoly 8

- Při práci se schématickým editorem je třeba dávat pozor na dodržování určitých pravidel. V opačném případě vznikají chyby již při tvorbě modelu obvodu. Na některé z nich nás upozorní samotný program při pokusu o analýzu, jiné jsou však na první pohled „skryté“ a mohou vést k nesprávným výsledkům (např. nesprávné používání symbolických proměnných v zřetězených příkazech .DEFINE).
- Velmi důležité je zvládnutí aktivního využívání příkazu .MODEL. Bude se mj. hodit při vašem pozdějším přechodu na simulační program SPICE.
- MicroCap umožňuje plnohodnotnou práci se SPICE modely. Doporučujeme podrobně nastudovat kapitolu 8.3.2.
- Možnost zápisu modelů prvků do vstupního souboru je důležitou vlastností programu MicroCap 7, která zajišťuje přenositelnost analyzovaných dat mezi uživateli, pracujícími s různými knihovnami prvků. Techniky popsané v části 8.3.3 na str. 88 vám umožní pracovat v rámci volně šířitelné verzi MicroCapu s modely prvků, které nejsou obsaženy v knihovnách této verze.

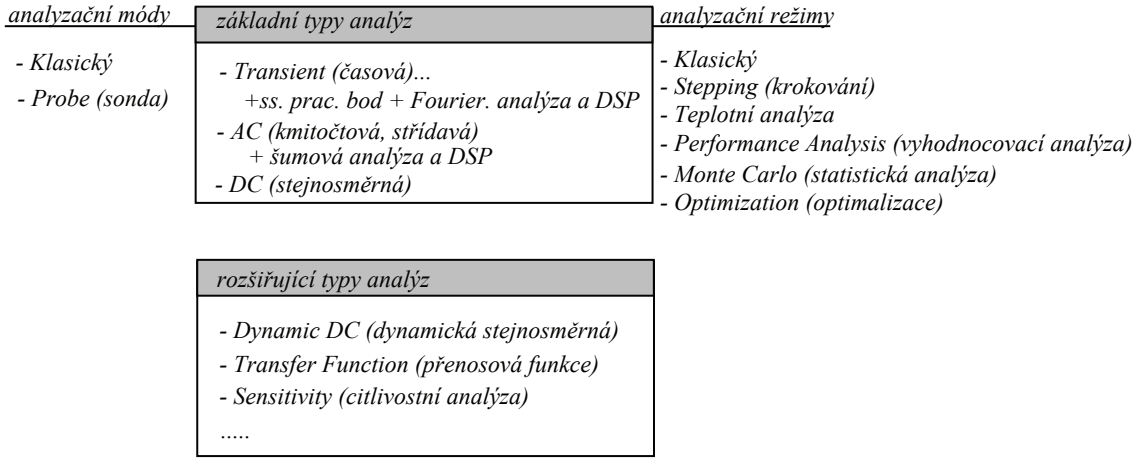
9 Analýza pomocí numerického simulátoru

9.1 Typy analýz, analyzační módy a režimy

Víme již, že **základními typy analýz** v numerických simulátorech jsou analýzy „*Transient*“, „*AC*“ a „*DC*“. Různé simulátory nabízejí nad rámec těchto základních typů **rozšiřující analýzy**. Na **obr. 9.1** jsou uvedeny typy, používané v MicroCapu.

Základní analýzy mohou být provozovány v tzv. **analyzačních módech**. Například PSpice nezná žádný jiný mód než klasický, což znamená posloupnost: tvorba modelu obvodu, analýza, zobrazení výsledků analýzy. MicroCap umožňuje práci v interaktivním módu „*Probe*“, kdy je obrazovka rozdělena na dvě části. V jedné z nich je schéma obvodu, v druhé části se zobrazují výsledky analýzy. Uživatel kliká do různých míst ve schématu (uzly, součástky..) a bezprostředně se mu objevují výsledky analýzy, například časové průběhy signálů v příslušných uzlech, kmitočtové charakteristiky apod. V tomto módu je tedy imitováno používání měřicích sond („*Probe*“) v laboratoři.

Dále je možno základní typy analýz provozovat v několika **analyzačních režimech**. Dosud jsme se měli možnost seznámit – kromě klasického režimu – s režimem „*Stepping*“ – krokování, neboli vícenásobnou analýzou. Speciálním případem vícenásobné analýzy je *teplotní analýza*. Mnoho simulátorů je kromě toho vybaveno dalšími režimy, zejména možností *statistické analýzy*. Na **obr. 9.1** jsou uvedeny režimy programu MicroCap.



Obr. 9.1: Typy analýz, analyzační módy a režimy MicroCapu.

První dva typy základních analýz – „*Transient*“ a „*AC*“ – většinou nabízejí i další možnosti analýzy. V menu *časové analýzy* bývá možnost vybrat vyhodnocení stejnosměrného pracovního bodu obvodu. Dále je možno časové průběhy, získané analýzou, podrobit *Fourierové transformaci* a nalézt spektrum signálů, popřípadě provádět další operace nad signálem, které se označují zkratkou *DSP* („*Digital Signal Processing*“ – číslicové zpracování signálů).

Kmitočtová analýza bývá obvykle doplněna tzv. *šumovou analýzou*, jejímž cílem je posouzení, do jaké míry proniká vlastní šum jednotlivých součástek obvodu na výstupy. V rámci kmitočtové analýzy bývá možnost aplikace operací „*DSP*“ na kmitočtové

charakteristiky například s cílem stanovení časových průběhů *zpětnou Fourierovou transformací*. Těchto možností analýzy se však využívá jen velmi okrajově.

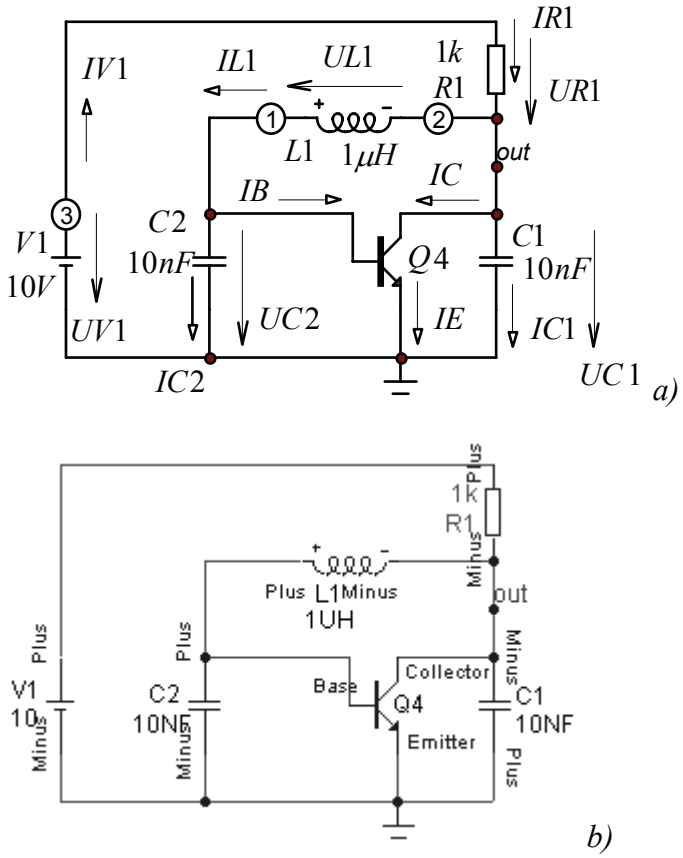
V této kapitole probereme podrobněji tři základní typy analýz v klasickém analyzačním módu. Práci v módu „*Probe*“ je možné snadno nastudovat z dokumentace MicroCapu. Pak stručně popíšeme některé z rozšiřujících typů analýz. V závěru se naučíme pracovat ve vybraných analyzačních režimech.

9.2 Co je dobré vědět před zahájením vlastní analýzy

Po zvolení typu analýzy budeme postaveni před problém „říci“ programu, jaký má být konkrétní cíl analýzy, například: zajímá mě napětí mezi uzly 3 a 8, nebo proud tekoucí do báze tranzistoru *Q*5, a podobně. Musíme to říci jazykem, kterému program rozumí. V následujícím textu se tedy seznámíme se způsoby zápisu zejména napětí a proudů. Z části 4.7.2 již víme, že tyto zápisy se provádějí do příslušných políček v oknech „*Analysis Limits*“.

9.2.1 Zápis napětí a proudů

Na **obr. 9.2** je schéma Colpittsova oscilátoru, převzaté ze souboru na disku **COLPITTS.CIR**. V obrázku jsou vyznačeny některá napětí a proudy, které by mohly být předmětem analýzy.



Obr. 9.2: a) Schéma Colpittsova oscilátoru s vyznačenými napětími a proudy, b) zviditelněná jména pinů.

Napětí se značí symbolem *V*, proudy *I*. Připomínáme, že simulátor nerozlišuje mezi velkými a malými písmeny.

Zápis napětí

V(*uzel*) …	Napětí mezi uzlem, jehož jméno je „uzel“, a referenčním uzlem (zemí). Jméno může být buď číslo uzlu nebo grid text.
V(*uzel*1,*uzel*2) …	Napětí měřené od *uzlu*1 do *uzlu*2. Jde o ekvivalenci zápisu *V*(*uzel*1)-*V*(*uzel*2).
V(*dvojpol*) …	Napětí měřené mezi svorkami dvojpólu od pinu „*Plus*“ k pinu „*Minus*“. Názvy pinů daného dvojpólu můžeme zobrazit poklepáním na součástku a zatržením položky „*Display*“, „*Pin Names*“ (viz **obr. 9.2** b).
VXY(*Npol*) …	Napětí měřené mezi svorkami *X* a *Y* *N*-pólu, například tranzistoru. Namísto *X* a *Y* je třeba volit symboly specifikované pro danou součástku v dokumentaci programu. Například *VCE*(*Q*4) je napětí kolektor-emitor tranzistoru *Q*4.

Tak například **napětí *UC*1** na **obr. 9.2** a) můžeme zapsat těmito způsoby:

V(2) nebo *V*(*out*) nebo –*V*(*C*1) nebo *VCE*(*Q*4) nebo –*VEC*(*Q*4).

Obdobně napětí na induktoru *UL*1:

V(2,1) nebo *V*(*out*,1) nebo *V*(2)-*V*(1) nebo *V*(*out*)-*V*(1) nebo –*V*(*L*1) nebo *VCB*(*Q*4),

nebo taky trochu komplikovaněji

V(*out*)-*V*(*C*2) nebo –*V*(*C*1)-*V*(*C*2) atd.

Je tedy zřejmé, že pokud chceme použít zápis *V*(*dvojpol*), měli bychom si v zájmu správného znaménka ve výsledku zviditelnit jména pinů. Existuje ještě druhá možnost. Rotujeme-li značkou součástky ještě před jejím umístěním na plochu, pak si všimněme, že k rotaci dochází vždy kolem jednoho z pinů, který se nazývá „*origin*“. U dvojpólů je *origin* totožný s pinem, označeným jako „*minus*“ (tak je to standardně definováno v „*Component Editoru*“). Pak je tedy napětí dvojpólu orientováno do *originu*.

Zápis proudu

I(*uzel*1,*uzel*2) …	Proud tekoucí dvojpólem, připojeným mezi *uzel*1 a *uzel*2, ve směru od *uzlu*1 do *uzlu*2. Mezi těmito uzly musí být bezprostředně připojen jen jeden dvojpól, jinak program zahlásí chybu.
I(*dvojpol*) …	Proud tekoucí dvojpólem od pinu „*Plus*“ k pinu „*Minus*“.
IX(*Npol*) …	Proud tekoucí do pinu *X* *N*-pólu. Například *IB*(*Q*4) je proud tekoucí do báze tranzistoru *Q*4.

! Z definice zápisu *I*(*dvojpol*) vyplývá, že program nerozlišuje mezi pasivním a aktivním dvojpólem, mezi spotřebičovou a zdrojovou orientací čítacích šipek napětí a proudů. To znamená, že proud zdroje napětí se uvažuje stejné orientace jako napětí.

Vrátíme-li se k **obr. 9.2**, pak například **proud *IC*1** můžeme zapsat takto:

$I(2,0)$ nebo $-I(0,2)$ nebo $-I(C1)$.

První zápis je platný, i když mezi uzly 2 a 0 je kromě kapacitoru *C*1 i přechod kolektor-emitor tranzistoru *Q*4. Není to totiž dvojpól. Kdybychom paralelně k *C*1 připojili například rezistor, pak by zápis *I*(2,0) vygeneroval chybu.

Proud induktorem *IL*1 lze zapsat takto:

$I(2,1)$ nebo $-I(1,2)$ nebo $-I(L1)$

Emitorový proud *IE* tranzistoru *Q*4:

$IE(Q4)$

Proud rezistorem *IR*1:

$I(3,2)$ nebo $-I(2,3)$ nebo $I(R1)$

Proud *IV*1 odebíraný ze zdroje *V*1:

$-I(V1)$ nebo $-I(3,0)$ nebo $I(0,3)$

Podrobnosti ohledně značení svorek dalších součástek (tranzistorů *MOSFE* a dalších), jakož i značení jiných obvodových veličin než je napětí a proud (kapacita, el. náboj, výkon rozptylovaný na součástce, energie nahromaděná v součástce apod.), jsou uvedeny v dokumentaci programu.

9.2.2 Význam symbolů *V* a *I* v různých typech analýz

Napětí a proudy mají různou fyzikální interpretaci při různých typech analýz:

- V analýze „*DC*“ se jedná o stejnosměrné hodnoty. Číselné údaje jsou reálná čísla.
- V analýze „*Transient*“ dostáváme okamžité hodnoty signálů ve výpočetních bodech na časové ose. Číselné údaje jsou opět reálná čísla.
- V analýze „*AC*“ se počítají signály linearizovaných modelů v harmonickém ustáleném stavu. Výpočty probíhají s využitím fázorů. Číselné údaje jsou komplexní čísla. Z nich jsou pak pomocí postupů, známých z teorie, získávány cíle analýzy (amplitudové a fázové kmitočtové charakteristiky apod.).

9.2.3 Zápis odvozených veličin pomocí vzorců

Simulační programy obvykle disponují rozsáhlým arzenálem matematických operátorů a funkcí. Jejich využíváním získáváme prakticky neomezené možnosti při vytyčování konečných cílů analýzy.

Řada operátorů a funkcí je použitelná univerzálně, jiné jsou navrženy pouze pro konkrétní typy analýz. Jejich úplný seznam naleznete v nápovědě nebo v programové dokumentaci. S některými z nich se seznámíme u konkrétních analýz a v příloze P11.2.

K nejjednodušším a často používaným aplikacím vzorců patří analýza přenosových vlastností obvodů (nejčastěji podíl dvou napětí nebo dvou proudů), výkonů (součin napětí a proudů) a impedancí (podíl napětí a proudů). Při analýze impedančních poměrů je v některých případech nutné zasáhnout do originálního zapojení obvodu: například při zjišťování výstupní impedance zesilovače je třeba přemístit zdroj budicího signálu na výstupní bránu, vstupní

bránu nahradit zkratem pro střídavý signál (nebo rozpojením, podle toho, jedná-li se o zdroj napětí nebo proudu), a výstupní impedanci vypočítat z napětí zdroje a proudu, který je z něho odebírán.

9.2.4 Co mají všechny základní analýzy společné

Srovnáme-li editační okna všech tří základních analýz na **obr. 9.3**, zjistíme, že některé položky se zde opakují. Zaměříme se na tyto:

„**Run Options**“ (podmínky běhu programu při analýze).

Vybrat si můžeme z těchto tří možností:

Normal: Základní simulační běh bez zápisu výsledků na disk.

Save: Výsledky analýzy jsou průběžně ukládány na disk do souboru s příponou **.TSA** (analýza „*Transient*“), **.ASA** (analýza „*AC*“) nebo **.DSA** (analýza „*DC*“) s cílem jejich využití v režimu „*Retrieve*“.

Retrieve: Analýza neprobíhá v klasickém smyslu, ale výsledky jsou čteny z datového souboru, vytvořeného v režimu „*Save*“.

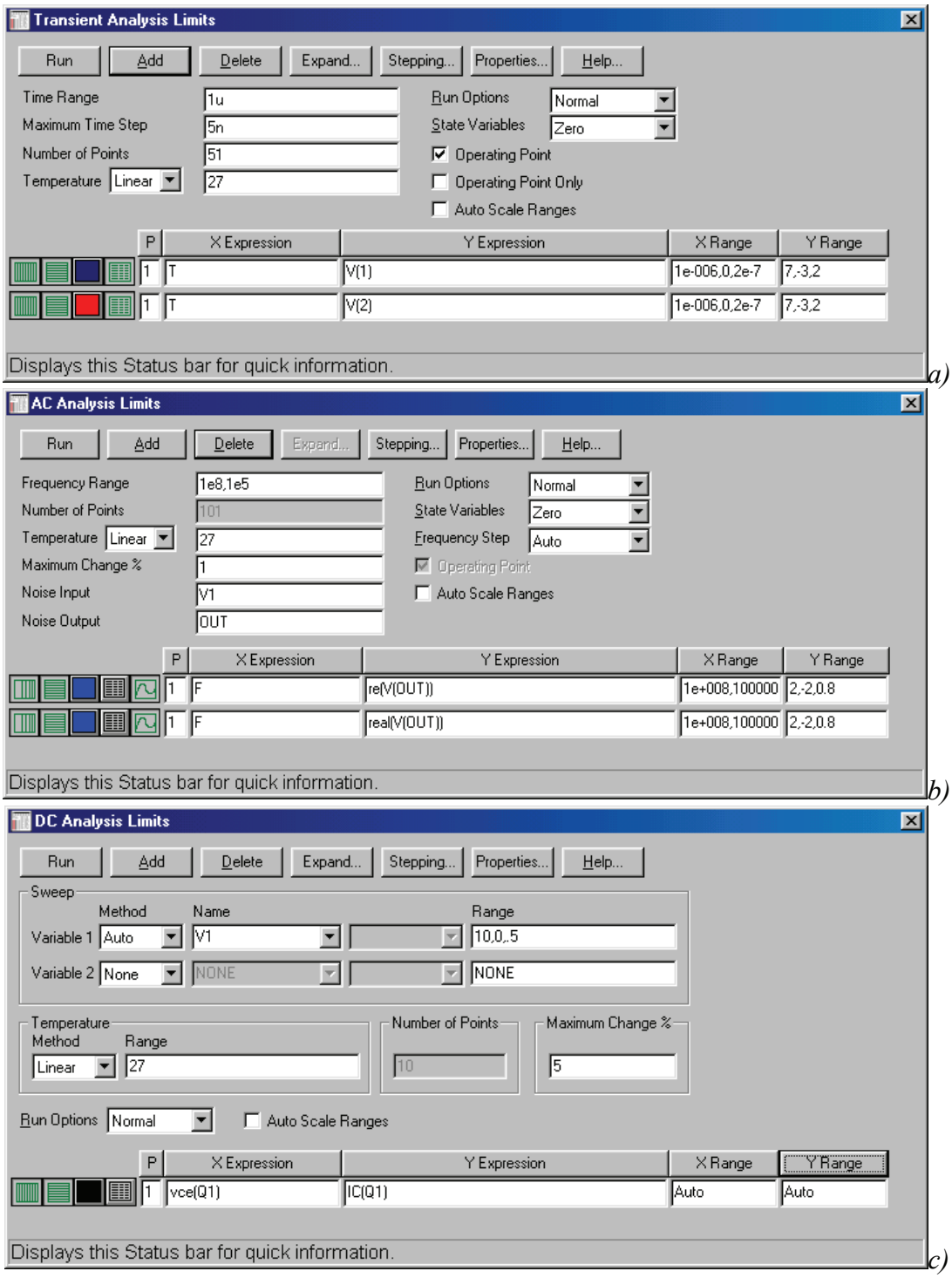
Podmínek „*Save*“ a „*Retrieve*“ můžeme využít k vytvoření dat pro simulaci složitého obvodu na výkonném počítači a jejich následném využití k další práci na méně výkonném hardware. Dalším vhodným použitím jsou prezentace a ukázky.

„**Number of Points**“ (počet bodů).

Tato položka bývá často zdrojem nedorozumění. V drtivé většině případů ji uživatel nemusí vůbec vyplňovat. Obecně tato položka obsahuje údaj o počtu bodů křivky, které se uloží do výstupního textového souboru, pokud se uživatel rozhodne tento soubor vytvořit. Nesouvisí to tedy se skutečným počtem bodů výpočtu. Body, o nichž je řeč, se určí ze skutečných bodů křivky interpolací. Standardní počet je nastaven na 51.

Ve všech základních typech analýz můžeme získat výsledky buď v standardní grafické formě (křivky), nebo navíc i ve formě tabulky hodnot zapsaných v textovém souboru (viz **obr. 9.4**). Druhou z možností musí uživatel povolit. Textový soubor může sloužit k přenosu grafických dat do jiných programů, např. Excelu, Easy Plotu a dalších.

Křivky se vykreslují pospojováním vypočtených bodů lomenými čarami. Hustotu rozložení bodů na ose „*X*“, neboli výpočetní krok, většinou program volí automaticky s ohledem na to, jak je křivka v různých místech strmá (automatická volba kroku je standardně přednastavena). Před zahájením analýzy proto není znám počet bodů výpočtu. Položka „*Number of Points*“ s tímto počtem nesouvisí.



Obr. 9.3: Okna „Analysis Limits“ pro všechny tři základní typy analýz.

Pokud uživatel zvolí fixní krok, pak teprve „*Number of Points*“ určuje počet bodů výpočtu, přičemž její původní význam, počet bodů ukládaných do případného textového souboru, je zachován.

Zvyšováním údaje „*Number of points*“ pak můžeme zvyšovat hladkost získaných křivek, ovšem za cenu prodlužujících se výpočtů.

V režimu automatické volby kroku je možné řídit hladkost křivek pomocí prostředků, které jsou popsány dále.

Prostředky pro řízení hladkosti křivek při automatické volbě výpočetního kroku

Upozorňujeme, že tyto prostředky umožňují řídit hladkost křivek volbou dostatečného množství bodů na těchto křivkách, a nemají vliv na **přesnost** výpočtu těchto bodů.

U časové analýzy se jedná o položku „*Maximum Time Step*“, u ostatních analýz „*Maximum Change*“.

„**Maximum Time Step**“ (maximální časový krok) – u analýzy „*Transient*“.

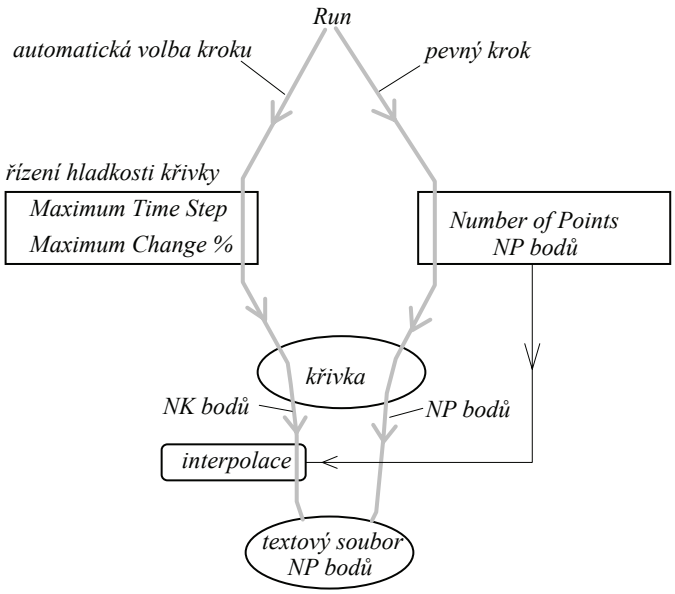
Je to maximální možná velikost kroku, kterou simulátor nemůže překročit při jeho automatickém nastavování.
Standardně je maximální krok nastaven na 1/50 rozsahu osy *X*. Snižováním této hodnoty docílíme jemnějšího vykreslení křivky, ovšem za cenu zvětšení objemu dat a prodloužení analýzy. Při standardním nastavení je v položce „*Maximum Time Step*“ zapsána nula, což může být matoucí.

„**Maximum Change%**“ (maximální změna v %) – u analýz „*AC*“ a „*DC*“.

Číslo, které se zapisuje do této položky, udává procentuální část z nastaveného rozsahu na ose *Y*. Výpočetní krok je řízen tak, aby se „*Y*“ souřadnice dvou po sobě jdoucích bodů křivky nelišily o více než je „*Maximum Change %*“.
Standardně je nastavena hodnota 5 (%). Pokud se křivka jeví jako nedostatečně hladká, zkusmo tuto hodnotu snížíme.

Upozornění: Tento mechanismus řídí délku kroku podle křivky, definované **v prvním řádku** prostoru pro definici křivek. Pokud je křivek definováno více a první z nich je relativně plochá, mechanismus nemůže správně fungovat a ostatní křivky budou vykresleny hrubě, i když položku „*Maximum Change %*“ hodně snížíme. V tomto případě pomůže přehození pořadí zápisu křivek: na první místo zapíšeme křivku, která vykazuje nejvýraznější změny.

Přehledné znázornění souvislostí mezi položkou „*Number of Points*“ a položkami pro řízení hladkosti křivky je na **obr. 9.4**. V případě pevného kroku obsahuje jak křivka, tak i textový soubor stejný počet *NP* vypočtených bodů, který je dán obsahem položky „*Number of Points*“. Při častěji využívané automatické volbě délky kroku je pro počet bodů křivky *NK* rozhodující nastavení položek pro řízení hladkosti křivky. Rozložení bodů je na ose *X* nerovnoměrné. Pokud je povolen zápis do textového souboru, program z nich lineární interpolací přepočítá *NP* bodů rovnoměrně rozložených na ose *X* a uloží je do textového souboru s příponami **.TNO**, **.ANO** a **.DNO** pro případ analýzy „*Transient*“, „*AC*“ a „*DC*“.



Obr. 9.4: Souvislosti mezi položkou „Number of points“ a položkami pro řízení hladkosti křivky.

Pro úplnost dodejme, že položka „*Number of Points*“ se uplatňuje ještě při Fourierové analýze signálů, což bude objasněno v části 9.3.9.

Další položkou, společnou pro všechny analýzy, je

„**Temperature**“ (teplota).

Specifikuje tzv. globální teplotu ve stupních Celsia. Standardní nastavení je na 27°C. Je platná pro všechny součástky, pokud ovšem některé nemají specifikovánu teplotu vlastní. Nerozlišuje se mezi teplotou okolí, pouzdra, čipu atd. Myslí se tím vždy „interní“ teplota součástky, která bezprostředně ovlivňuje její elektrické parametry.

Můžeme si vybrat mezi formáty „*Linear*“ a „*List*“.

Linear: *max_teplota*[,*min_teplota*[,*krok*]]
List: *teplota*1[,*teplota*2[,…..]]

Parametry v závorkách jsou nepovinné.
Příklady na režim “*Linear*”:

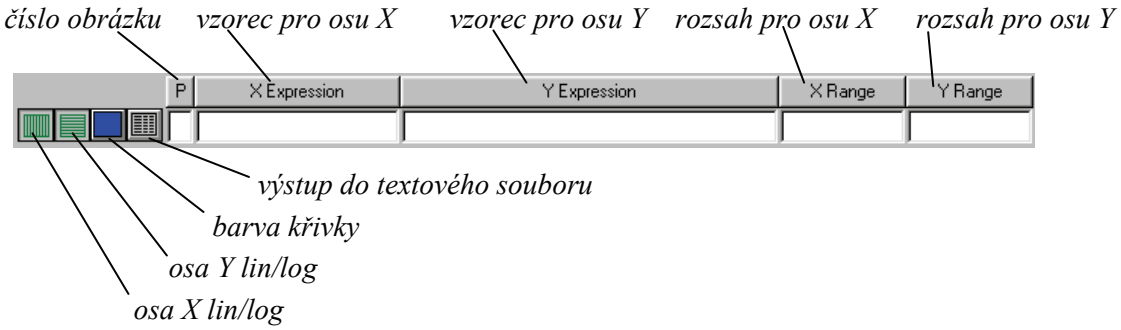
50	provede se jediný analyzační běh pro teplotu 50°C.
50,20	2 analýzy pro teploty 20°C a 50°C.
50,20,10	4 analýzy pro teploty (20, 30, 40, 50)°C.

Příklad na režim “*List*”:

0,27,37,50,59,65 6 analýz pro vyjmenované teploty.

Zadávání analyzovaných závislostí

Jeden z řádků, sloužící pro zadávání analyzované závislosti, je na **obr. 9.5**. Obrázek je platný pro všechny analýzy s jednou výjimkou u analýzy „*AC*“, kde je jedna ikona navíc vpravo od ikony „výstup do textového souboru“. Její význam vysvětlíme na jiném místě.



Obr. 9.5: Popis řádku z okna „Analysis Limits“ pro zadávání analyzovaných závislostí.

Při analýzách, kdy požadujeme najednou zobrazit velký počet křivek, je možné přidávat další řádky klikáním na prvek „*Add*“ (vpravo od „*Run*“). Podobně lze řádky, na něž umístíme kurzor, mazat prvkem „*Delete*“.

Všechny křivky mohou být umístěny do jediného obrázku, nebo je můžeme sdružovat po skupinách do více obrázku, které budou umístěny pod sebou. Tyto obrázky pak budou číslovány vzestupně shora dolů. Je výhodné sdružovat do jednoho obrázku křivky stejných fyzikálních rozměrů a odpovídajících si měřítek. Do různých obrázků umístíme například napětí a proudy apod.

Do sloupce „*P*“ („*Plot Group*“) zapisujeme číslo obrázku, v němž bude umístěna křivka, definovaná na daném řádku. Pokud políčko nevyplníme, křivka se nevykreslí vůbec.

Do políčka „*vzorec pro osu X*“ se u analýzy „*Transient*“ standardně zapisuje proměnná „*T*“ – čas a u analýzy „*AC*“ proměnná „*F*“ – kmitočet. Není to však pravidlo. Je zde možné zapsat libovolný vzorec, jehož číselné výsledky pak budou řídit souřadnice kreslícího bodu na ose *X*. Obdobně to platí pro políčko „*vzorec pro osu Y*“.

Údaje v položkách „*rozsah pro osu X*“ a „*rozsah pro osu Y*“ mají následující formát:

max [*,min*] [*,grid_spacing*] [*,bold_grid_spacing*]

Údaje v závorkách jsou opět nepovinné. Daná osa bude ohraničena hodnotami „*min*“ a „*max*“ a vnitřně bude rozdělena čarami mřížky („*grid*“) po úsecích „*grid_spacing*“. Údajem „*bold_grid_spacing*“ můžeme dosáhnout toho, že některé z čar mřížky budou zvýrazněny.

Poslední dva parametry „*grid*“ mají význam jen v případě lineární osy. U logaritmické osy se mřížka kreslí automaticky po dekádách.

Pokud některý z nepovinných parametrů neuvedeme, předpokládá se jeho implicitní hodnota 0.

Příklady použití:

50,0 nebo 50	Osa je ohraničena hranicemi 0 a 50 jednotek. Musí být lineární.
50,20,10	Osa je ohraničena hranicemi 20 a 50 jednotek. Tento prostor je rozdělen mřížkou po 10 jednotkách.

Umístíme-li kurzor dovnitř daného okénka, pak po kliknutí pravého tlačítka myši si zpřístupníme některá výhodná nastavení rozsahů, např. nastavení „*auto*“, kdy po proběhnutí analýzy dojde k automatickému nastavení měřítek podle rozsahu číselných údajů, které jsou v paměti. Kliknutím na položku „*X Range*“, resp. „*Y Range*“ (viz **obr. 9.5**) levým tlačítkem myši můžeme tato nastavení realizovat globálně pro všechny křivky.

Nastavení měřítek „*auto*“ lze doporučit jen jako výchozí pro analýzu, kdy nejsme schopni odhadnout, v jakých mezích se výsledky budou pohybovat. V tomto režimu totiž probíhá analýza s „hrubým krokem“, takže u členitějších křivek nedojde k správnému odhadu mezí. Meze pak musíme stejně dostavit „ručně“.

Význam čtyř ikon v levé části **obr. 9.5** vyplývá z jejich popisu a není nutné jej komentovat.

9.3 Analýza „*Transient*“ neboli časová analýza

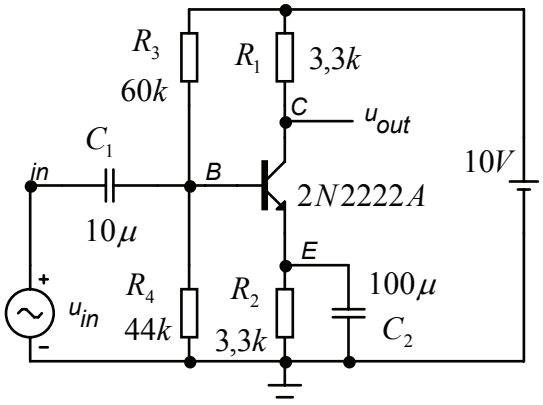
9.3.1 Cíle analýzy

Prvotním cílem časové analýzy je imitace činnosti „inteligentního osciloskopu“, tj. zjišťování časových průběhů signálů v obvodu, který se může nacházet v libovolném přechodovém nebo ustáleném stavu.

Druhotné cíle časové analýzy mohou spočívat v různých způsobech dalšího zpracování analyzovaných signálů. Zde se často používá Fourierova analýza s cílem získání spektrálních charakteristik signálů.

9.3.2 „Inteligentní osciloskop“

Uvažujme tranzistorový zesilovač na **obr. 9.6**. Pracovní bod tranzistoru je nastaven do třídy *A*. Na vstupu je harmonický signál o kmitočtu 10 *k*Hz a nastavitelné amplitudě. Úkolem obvodu je signál napěťově zesílit.

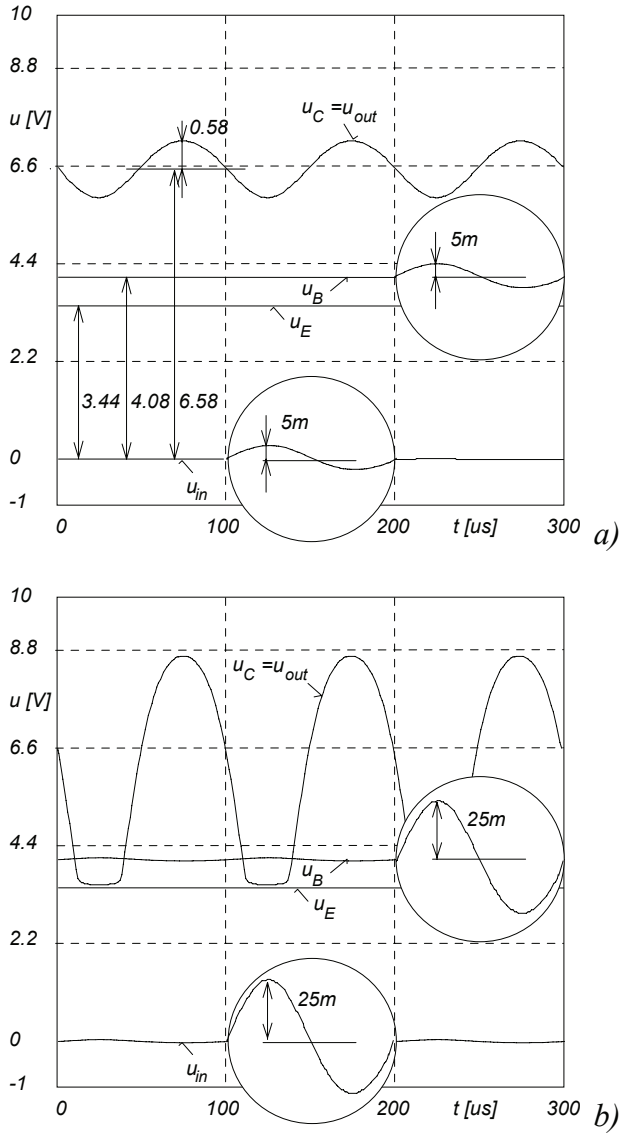


Obr. 9.6: Jednostupňový tranzistorový zesilovač.

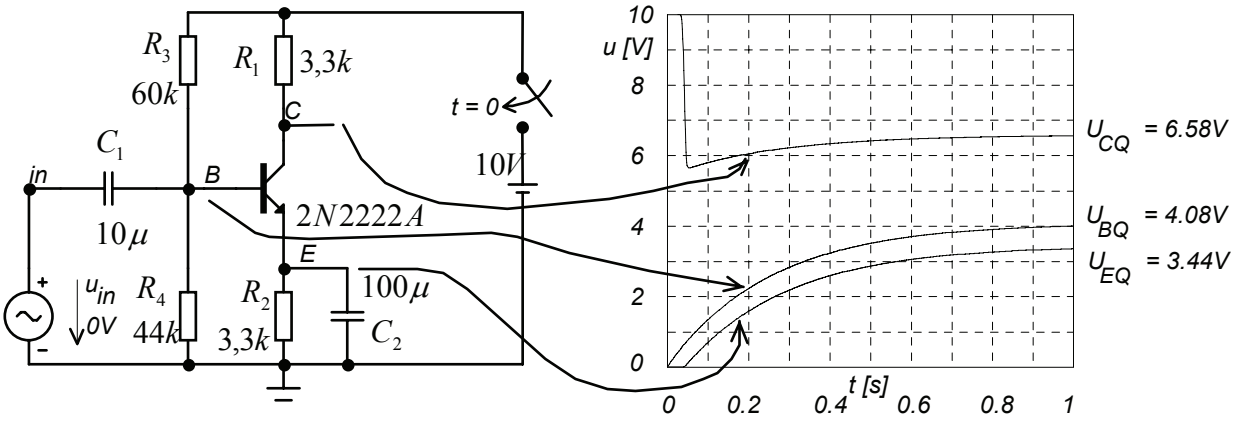
Pomocí běžného osciloskopu je možné sledovat časové průběhy napětí v různých uzlech obvodu v ustáleném stavu. Ukázky jsou na **obr. 9.7** a) a b) pro vstupní signál o amplitudě 5 *m*V a 25 *m*V.

Simulátor by při časové analýze měl umět mnohem více. Jednak by měl umožňovat zobrazení všech možných časových průběhů, které lze odvodit na základě napětí a proudů, například časový vývoj kolektorových ztrát tranzistoru, výkonovou účinnost zesilovače apod.

Dále by měl program simulovat činnost dokonalého paměťového osciloskopu k zaznamenávání jednorázových přechodných dějů. Na **obr. 9.8** je ukázka děje, ke kterému by došlo po připojení napájecí baterie k zesilovači, přičemž zesilovaný signál zatím na vstupu nepůsobí (lépe řečeno, jeho napětí je nulové). Vznikne přechodný děj, jehož výsledkem je „náběh“ obvodu do stejnosměrného ustáleného stavu. Ustálená napětí U_{BQ}, U_{CQ} a U_{EQ} na bázi, kolektoru a emitoru pak můžeme považovat za souřadnice stejnosměrného pracovního bodu tranzistoru. Vidíme, že tento „náběh“ obvodu do pracovního bodu trvá asi sekundu. Průběh přechodu bude jistě záviset na tom, zda v okamžiku připojení napájecího zdroje byly všechny vnitřní akumulační prvky, včetně modelovaných parazitních, vybity či nikoliv. Tyto vlivy by měl program rovněž dokázat modelovat.

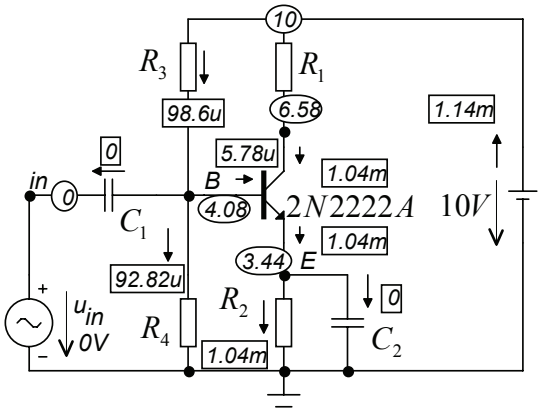


Obr. 9.7: Ustálená napětí a proudy v zesilovači při působení relativně a) slabého, b) silného vstupního signálu.

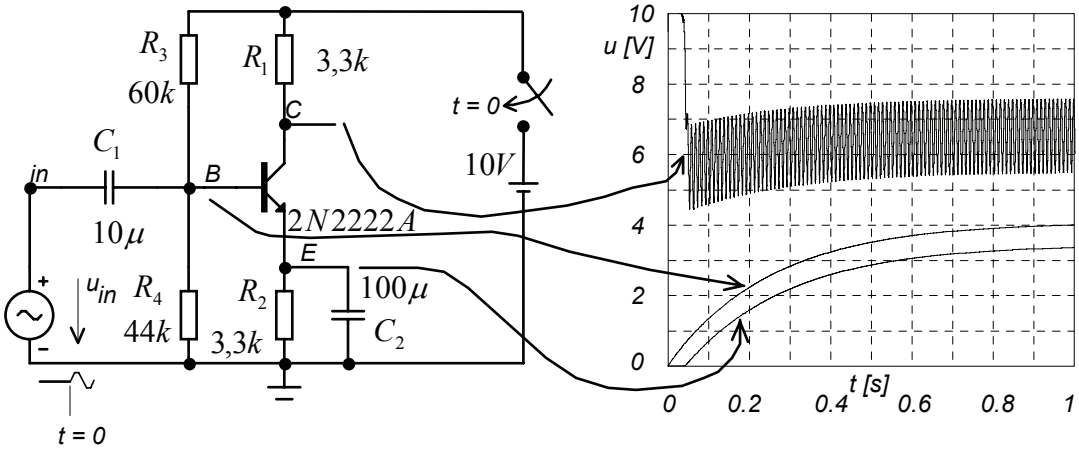


Obr. 9.8: Ukázka „náběhu“ obvodu do stejnosměrného ustáleného stavu po připojení napájecího zdroje.

V řadě případů nás nemusí zajímat přechod do pracovního bodu, nýbrž jen daný pracovní bod jako takový. Program by měl dokázat „přeskočit náběh“, přímo vypočítat souřadnice pracovního bodu a prezentovat je uživateli vhodnou formou, například promítnutím stejnosměrných napětí a proudů přímo do schématického editoru, jak je ukázáno na **obr. 9.9**.



Obr. 9.9: Zobrazení souřadnic stejnosměrného pracovního bodu přímo do schématu obvodu.



Obr. 9.10: Složený přechodný děj při současném připojení napájecího zdroje a budicího signálu.

Na **obr. 9.10** je ukázka složitého přechodného děje, kdy současně s připojením napájecí baterie začíná na vstup zesilovače působit střídavý signál. Výsledkem jsou signály, které znázorňují pomalý přechod obvodu do ustáleného stavu. V našem případě tento přechod trvá řádově 10000 period vstupního signálu. Takováto simulace s sebou přináší vysoké nároky na paměť počítače i rychlost programu, nehledě na problémy s rozlišením grafiky. Proto je výhodné, umí-li si program řešení časových průběhů „rozkouskovat" na posloupnost časově navazujících oken, kdy analýzu v daném okně naváže na výsledky analýzy na konci předchozího okna. MicroCap toto umožňuje v tzv. režimu *LEAVE*.

Existuje řada elektronických obvodů, kdy náběh do periodických ustálených stavů trvá relativně velmi dlouho. Typickým příkladem jsou krystalové oscilátory. Při simulaci těchto obvodů bývá velmi ceněnou vlastností simulátoru okamžité nalezení ustáleného stavu. Bohužel z dostupných simulátorů se touto schopností může pochlubit snad jen program CIA [3.7, 3.14]. V MicroCapu je k studiu ustálených stavů nutno použít režim *LEAVE*. V příslušné kapitole se seznámíme s dalším dobrým nástrojem pro řešení podobných problémů, s možností spuštění simulace za přesně definovaných tzv. počátečních podmínek. Tato možnost je využitelná ve všech standardních simulátorech.

V následující kapitole provedeme shrnutí a zobecnění uvedených poznatků s cílem objasnit, jak správně nastavit simulátor podle toho, co konkrétně požadujeme v režimu „*Transient*" analyzovat.

9.3.3 Stavové proměnné a počáteční podmínky pro časovou analýzu

Vraťme se na okamžik zpět do *Lekce 1* v části 6.2.1, kde jsme pomocí programu SNAP analyzovali *RC* článek typu dolní propust. Je zřejmé, že časový průběh výstupního napětí článku, sledovaný od počátečního času $t = 0$, bude záviset na dvou faktorech:

1. na časovém průběhu budicího signálu v čase $t \geq 0$,
2. na počátečním napětí, na něž byl nabit kapacitor v obvodu v čase $t = 0$.

Po zobecnění dostáváme známou poučku: napětí na všech kapacitorech a proudy všemi induktory jsou tzv. **stavové veličiny**. Definují (energetický) **stav obvodu** v daném okamžiku. Známe-li stav obvodu v čase $t = 0$ a průběh budicích signálů v časovém intervalu od 0 do T, pak můžeme jednoznačně určit stav, do něhož obvod dospěje v čase T (podrobnosti viz příloha P1).

V analogových simulátorech se z praktických důvodů považují za stavové veličiny, kterým se zde říká **stavové proměnné** („*State Variables*"), všechna uzlová napětí plus proudy všemi induktory. Uzlová napětí jsou přímo neznámé, které simulátor průběžně počítá, a jednoznačně určují i napětí na kapacitorech, které z nich lze odvodit lineárními kombinacemi. Hodnoty stavových proměnných v počátečním čase simulace $t = 0$ se nazývají **počáteční podmínky** simulace („*Initial Conditions*", *IC*). Aby mohlo dojít k analýze v režimu „*Transient*", potřebuje mít simulátor definované počáteční podmínky. Způsob jejich nastavení bude záviset na tom, z jakého výchozího stavu obvodu si bude uživatel přát daný obvod analyzovat. Tato problematika bude podrobně popsána v další kapitole.

9.3.4 Jak postupuje simulátor při časové analýze

Jak uvidíme dále, před vlastní analýzou se musí uživatel simulačního programu rozhodnout pro určité nastavení parametrů simulace, které program nabízí. Jsou to zejména položky, rozhodující o tom, zda počítat stejnosměrný pracovní bod a jaké zvolit počáteční

podmínky simulace. Uživatel neznalý věci raději ponechá implicitní nastavení těchto položek. Musí pak ovšem počítat s tím, že v některých situacích obdrží jiný výsledek simulace, než jaký požaduje. Ve výhodě jsou zde ti uživatelé, kteří správně porozuměli pojmům „*pracovní bod*“ a „*počáteční podmínka*“ (viz příloha P1) a zhruba vědí, jaký je postup simulátoru při řešení.

Důležité jsou teoretické poznatky, které na tomto místě pro přehlednost shrneme:

a) Simulátor sestavuje rovnice obvodu na základě modifikované metody uzlových napětí. Neznámými veličinami, které počítá, jsou napětí mezi jednotlivými uzly a tzv. referenčním uzlem, který je označen značkou uzemnění, a proudy induktory.

   Znamená to, že i když počítáme například jen jedno z uzlových napětí, které nás zajímá, program ve skutečnosti řeší celou soustavu rovnic jako takovou – k dispozici jsou všechny neznámé.

b) Reakce obvodu na daný budicí signál závisí nejen na tomto signálu, ale i na počátečních podmínkách, v nichž se obvod nacházel v okamžiku připojení budicího signálu. Například průběh nabíjení kapacitoru, který připojíme k baterii o daném vnitřním odporu, závisí na tom, na jaké napětí byl kapacitor nabit v okamžiku připojení. Simulátor rozumí pod pojmem počáteční podmínky množinu všech neznámých – tj. uzlových napětí a proudů induktory – v počátečním okamžiku simulace.

   Tyto počáteční podmínky bývají implicitně nastaveny na nulu. Implicitní nastavení nelze použít, hodláme-li např. zkoumat vybíjení kapacitoru, k němuž je paralelně připojen rezistor, ze zadaného počátečního napětí. Na druhou stranu je toto nastavení vhodné např. ke studiu nasazování kmitů v oscilátoru, nikoliv však k sledování ustálených kmitů.

Z výše uvedeného vyplývá jednoduchý závěr: kdybychom v libovolném okamžiku přerušili časovou analýzu, pak neznámé (vypočítané) veličiny tvoří vektor počátečních podmínek pro další simulaci. Z časového vývoje počátečních podmínek lze odvodit všechny existující signály v obvodu.

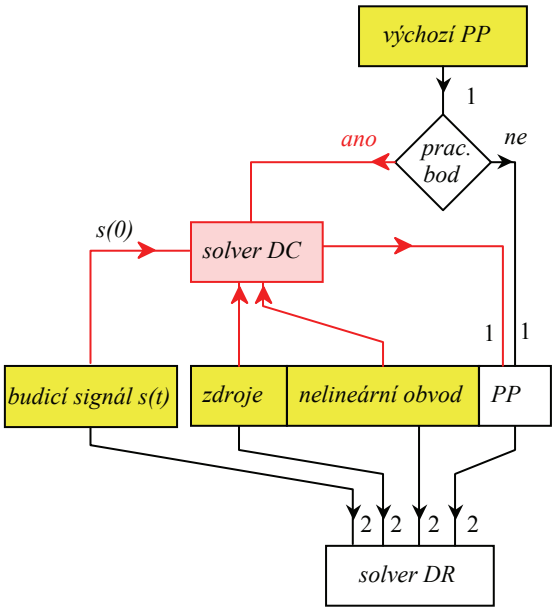
c) Připojíme-li obvod k stejnosměrným napájecím zdrojům, pak po odeznění přechodných dějů obvod „naběhne“ do stejnosměrného ustáleného stavu. Neznámé (vypočítané) veličiny simulátoru by pak odpovídaly stejnosměrnému pracovnímu bodu. Nelineární obvod může vykazovat několik pracovních bodů. Ke kterému z nich program „zkonverguje“, závisí na počátečních podmínkách simulace.

   Je zde tedy možnost počítat stejnosměrný pracovní bod prostřednictvím časové analýzy přechodných dějů po připojení obvodu ke zdrojům. Této možnosti se však využívá jen výjimečně, a to tehdy, selže-li interně zabudovaná metoda hledání pracovních bodů.

d) Stejnosměrný pracovní bod je hledán analýzou nelineárního nesetrvačného modelu, tj. řešením soustavy nelineárních algebraických rovnic. Jediné univerzální řešení představuje použití iteračních metod (konkrétně je to modifikovaná Newtonova-Raphsonova iterační metoda, viz [3.5]). Po proběhnutí každé iterace je srovnán výsledek řešení s výsledkem v předchozím kroku. Pokud se již výsledky neliší v mezích určité chyby, která je přednastavena v globálních podmínkách simulátoru, výpočet je ukončen a obsah proměnných je prohlášen za souřadnice pracovního bodu. Počáteční podmínky simulace zde fungují jako iterační „násada“ (první odhad řešení). Volbou „násady“ se můžeme pokusit ovlivnit, který z existujících pracovních bodů bude nalezen.

Na **obr. 9.11** je zjednodušené znázornění interních mechanismů, které působí v simulátoru při časové analýze.

V obrázku jsou šedou barvou vyznačeny bloky, reprezentující vstupy simulace: vlastní nelineární obvod spolu s napájecími a signálovými zdroji a výchozí počáteční podmínky (implicitně nastavené jako nulové). V první fázi simulace se naplní vektor počátečních podmínek (*PP*), v druhé fázi se vstupní data využijí k vlastnímu numerickému řešení časových průběhů v solveru diferenciálních rovnic. Způsob naplnění vektoru počátečních podmínek závisí na tom, je-li povolen výpočet pracovního bodu (implicitně je povolen). Jednoduché je to při nepovolení: pak se výchozí počáteční podmínky překopírují do vektoru *PP*. Je-li vyžadován výpočet pracovního bodu, použijí se výchozí počáteční podmínky jako „násada“ pro iterativní řešení v solveru *DC*. Solver si z modelu obvodu „vypůjčí“ jeho rezistivní část, budicí signály nahradí jejich počátečními hodnotami v čase $t = 0$, a provede řešení. Nalezený pracovní bod se pak překopíruje do vektoru počátečních podmínek a stává se počátečním bodem, od něhož se odvíjí řešení časových průběhů.



Obr. 9.11: Zjednodušené schéma mechanismu časové analýzy v simulačním programu. Význam zkratek a označení: PP jsou počáteční podmínky; solver DC je jednotka řešení stejnosměrného pracovního bodu iteračním řešením soustavy nelineárních algebraických rovnic; solver DR je jednotka výpočtu časových odezev numerickou integrací soustavy diferenciálních rovnic.

Časové průběhy jsou generovány tzv. solverem *DR* – viz **obr. 9.11**. Jeho činnost lze do jisté míry ovlivňovat pomocí parametrů v globálních podmínkách simulátoru. Blíže o tom pojednáme v části 8.6.

Výše popsanou strukturu řešení z **obr. 9.11** musíme respektovat při volbě strategie simulace konkrétních obvodů. Problematiku ukážeme na konkrétních příkladech v části 9.3.6.

9.3.5 Menu „Transient Analysis Limits“

Toto menu bylo ukázáno na **obr. 9.3** a). Z dosud neobjasněných položek vysvětlíme položky „*State Variables*“, „*Operating Point*“, „*Operating Point Only*“. Význam těchto položek má úzký vztah k vysvětlujícímu obrázku 9.11.

„**State Variables**“ (stavové proměnné).

Zde se definují počáteční podmínky analýzy. Je třeba vybrat z těchto možností:

Zero Nulové počáteční podmínky.
Read Počáteční podmínky se načtou ze souboru.
Leave První běh analýzy je proveden s nulovými počátečními podmínkami. Stav obvodu na konci analyzačního běhu se zapamatovává a stává se výchozími počátečními podmínkami pro další běh.

„**Operating Point**“ (pracovní bod).

Zatržením/nezatržením této položky realizujeme rozhodování, které je na **obr. 9.11** znázorněno rozhodovacím blokem „*prac. bod ano/ne*”.

„**Operating Point Only**“ (počítat **jen** pracovní bod).

Tento režim byl ukázán v kapitole 8.2.2. Pokud jej zatrhneme, nebude záležet na stavu položky „*Operating Point*“. Program vypočítá pouze pracovní bod a analýza se ukončí, aby měl uživatel možnost prohlédnout si výsledky. Následná časová analýza již neproběhne, protože bychom si tak „přepsali“ souřadnice nalezeného pracovního bodu.

9.3.6 Typická nastavení časové analýzy při řešení různých typů obvodů

V **Tab. 9.1** jsou shrnuty všechny nastavitelné kombinace stavů položek „*State Variables*“ a „*Operating Point*“. Ke každé kombinaci je uvedena jedna nebo více typických analyzačních úloh.

State Variables	Operating Point	
	ano	ne
Zero	*I.* - Obvod je již připojen k napájecím zdrojům, řešíme jeho reakci na vstupní signály (nejčastější typ úlohy).	*IV.* - Rozběh oscilátorů a generátorů kmitů. - Analýza pasivních obvodů s nulovými počátečními stavy. - Hledání ss. pracovních bodů u obvodů, kde klasický výpočet selhává.
Read	*II.* - Hledání pracovních bodů, které leží blízko zadaných počátečních podmínek.	*V.* - Ustálené kmity autonomních obvodů. - Start analýzy z přesně definovaných stavů obvodu. - Řešení přechodných dějů v obvodech s nenulovými počátečními podmínkami. - Hledání stejnosměrných pracovních bodů u obvodů, kde klasický výpočet selhává, v okolí počátečních podmínek.
Leave	*III.* ?	*VI.* - Postupný přechod obvodu do ustáleného stavu sledem opakovaných analyzačních běhů.

Tab. 9.1: Možné kombinace nastavení položek „*State Variables*“ a „*Operating Point*“ a typické analyzační úlohy k těmto kombinacím.

Obecně pracovní bod není nutné počítat v obvodech, kde je nulový (všechna stejnosměrná napětí a proudy jsou nulové), tj. v pasivních obvodech a ve všech obvodech bez stejnosměrných zdrojů. Údaje v tabulce jsou jen logickým vyústěním principu fungování simulátoru podle **obr. 9.11**. Ke kombinaci *III.* se praktická aplikace hledá opravdu těžko.

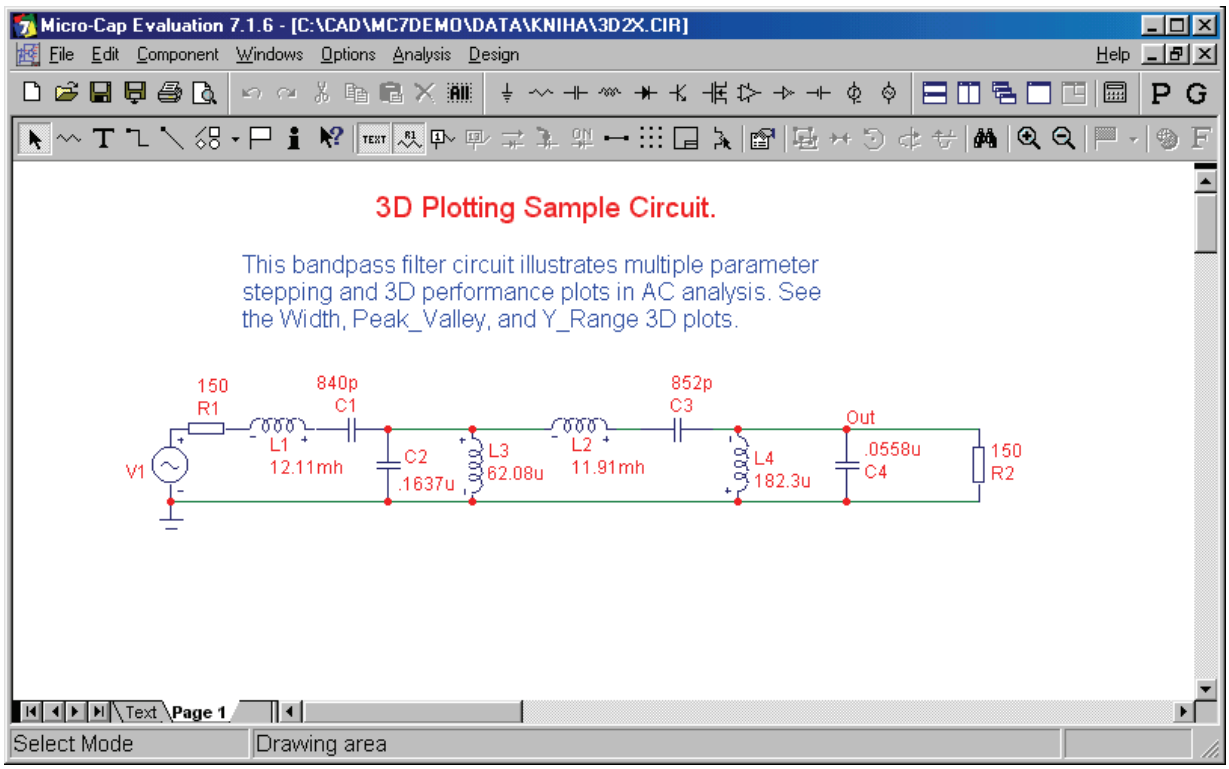
S kombinacemi *II* a *V* souvisí dva příkazy simulátorů z rodiny SPICE, které používá i MicroCap, a to *.IC* a *.NODESET*. Pomocí těchto příkazů můžeme dále modifikovat mechanismy fungování simulátoru při časové analýze podle **obr. 9.11** i při ostatních typech analýz. Pojednáme o nich v části 9.3.8.

9.3.7 Konkrétní příklady časové analýzy

Kombinace *I* z **Tab. 9.1** je typická pro řešení řady obvodů. Prakticky byl tento typ analýzy ukázán v části 8.2.2 na příkladu *TTL* hradla **TTLINV.CIR**. Kombinace *II* je naopak velmi speciální a běžný uživatel simulátoru ji zřejmě příliš často nevyužije. Jednoduchý příklad ukážeme v části 10.2.3. V následujících ukázkách se proto zaměříme – kromě kombinace *I* – zejména na kombinace *IV* až *VI*.

Příklad 1 – analýza pasivního příčkového filtru

Do editoru načteme ukázkový soubor **3D2.CIR**, který je uložen v adresáři *..MC7DEMO/DATA*. Protože jej budeme modifikovat, uložíme si jej volbou „*File/Save As*“ pod jiným názvem, například **3D2x.CIR**.



Obr. 9.12: Pasivní příčkový filtr typu pásmová propust.

Jedná se o příčkový filtr typu pásmová propust. Soubor je sice původně určen k demonstraci speciální funkce MicroCapu v analýze „*AC*“, ale dobře nám poslouží i k vysvětlení mechanismů analýzy „*Transient*“.

Poklepáním na značku zdroje signálu *V*1 zjistíme, že se jedná o zdroj sinusového signálu o kmitočtu 1 MHz a amplitudě 1 V. Protože pásmová propust je navržena na 50 *k*Hz, změníme v modelu zdroje obsah položky *F* na 50k. Zobrazením čísel uzlů se přesvědčíme, že vstupní uzel filtru má číslo 1.

Poté aktivujeme analýzu „*Transient*“. Simulační čas nastavíme na desetinásobek doby trvání opakovací periody budicího signálu, tedy na

$$10.\frac{1}{50k} = 200\ \mu s\ .$$

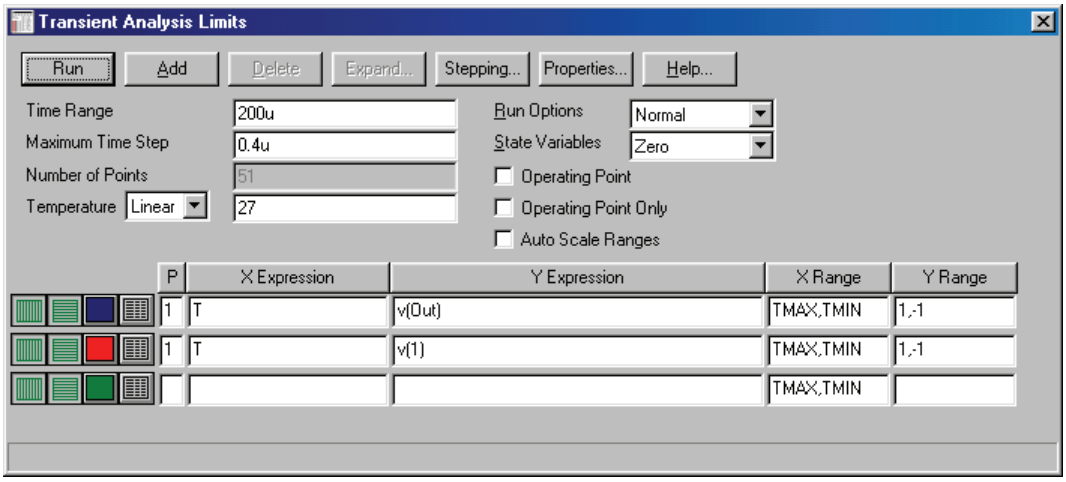
Původní obsah položky „*Time Range*“ tedy přepíšeme údajem 200U. Musíme změnit měřítko na ose *X*: Klikneme levým tlačítkem myši na položku „*X Range*“ a vybereme „*TMAX,TMIN*“.

V okénku „*Maximum Time Step*“ je nula, což ve skutečnosti znamená implicitní rozdělení časového rozsahu 200μs na 50 dílů. Protože v tomto rozsahu je 10 opakovacích period budicího signálu, připadalo by na vykreslení jedné periody 5 bodů, což je velmi málo. Kdybychom zvýšili počet bodů na periodu alespoň na 50, vycházel by maximální časový krok 200μs/500 = 0,4 μs.

Změníme i měřítka na ose *Y*. Přizpůsobíme je rozkmitu vstupního signálu (-1, +1) V.

Budeme řešit odezvu filtru na signál za předpokladu, že v okamžiku připojení zdroje signálu na vstup budou všechny vnitřní kapacity vybity a cívkami nepoteče proud. Tomu bude odpovídat nastavení „*State Variables – Zero*“. Pracovní bod není třeba počítat, protože ve filtru nejsou žádné stejnosměrné zdroje. Kdybychom položku „*Operating Point*“ přesto zatrhli, program by při jeho výpočtu v souladu s **obr. 9.11** uvažoval jediný stejnosměrný zdroj, a to na vstupu, o velikosti, rovné hodnotě vstupního signálu v čase 0. U sinusového signálu je to nula.

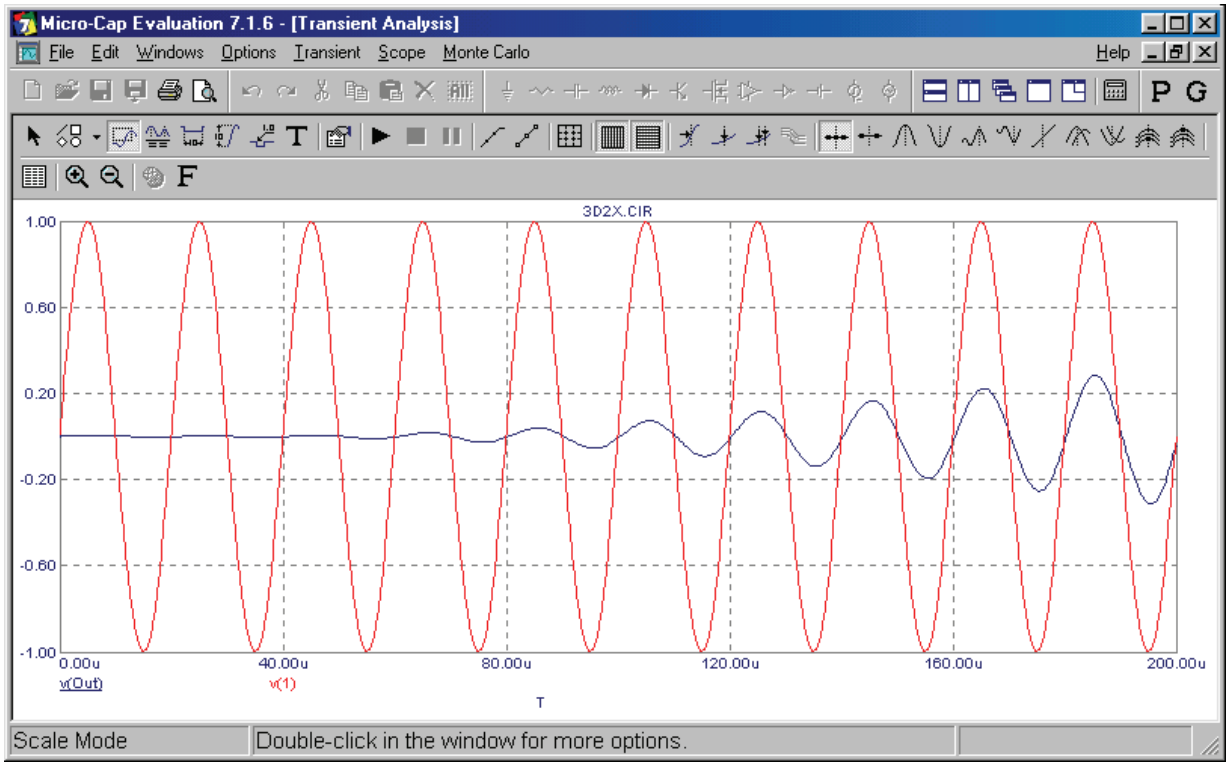
Na konci našich úprav by okno „*Transient Analysis Limits*“ mělo vypadat jako na **obr. 9.13**. V jediném obrázku se vykreslí časové průběhy výstupního a vstupního napětí filtru.



Obr. 9.13: Nastavení podmínek časové analýzy pro filtr z **obr. 9.12**.

Po proběhnutí analýzy dostaneme výsledek podle **obr. 9.14**. Je zřejmé, že po 10 opakovacích periodách se filtr ještě nedostal do ustáleného stavu, z něhož bychom byli schopni posoudit přenosové vlastnosti na kmitočtu 50 *k*Hz.

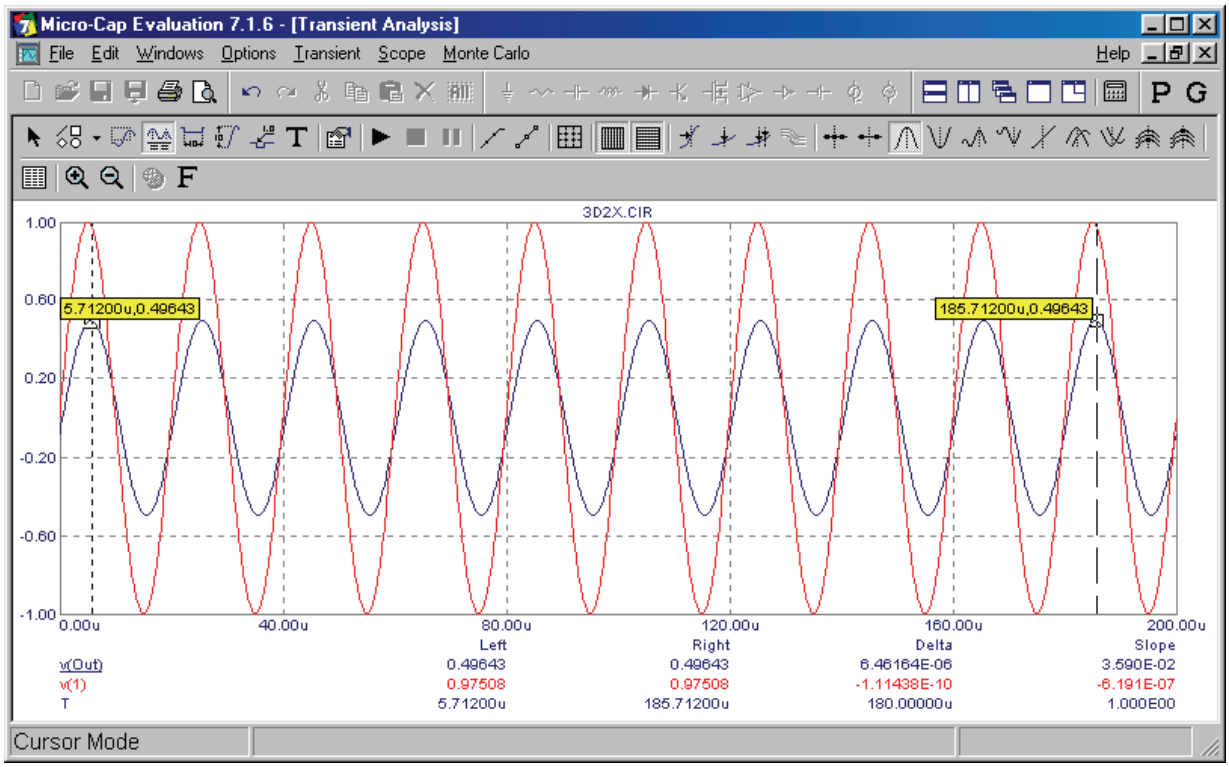
Vrátíme se do okna „*Transient Analysis Limits*“ (buď jeho zástupce nalezneme na spodní liště Windows, nebo se k němu dostaneme volbou „*Transient/Limits*“). Změníme nastavení položky „*State Variables*“ na „*Leave*“. Spustíme analýzu a po jejím ukončení ji spustíme ještě několikrát stiskem *F*2, dokud neuvidíme, že došlo k ustálení odezvy. Výsledný stav je zachycen na **obr. 9.15**. Z obrázku je zřejmé, že na kmitočtu 50 *k*Hz má filtr přenos asi 0,496. Je zde patrné i určité zpoždění výstupního signálu oproti vstupu.



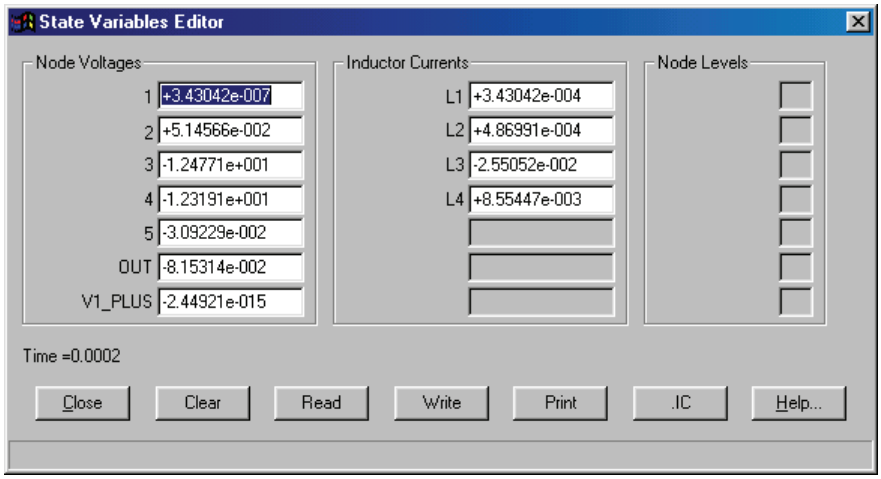
Obr. 9.14: „Náběh“ filtru do ustáleného stavu z nulových počátečních podmínek.

Nyní provedeme opatření, abychom mohli filtr analyzovat v ustáleném stavu bez mezivýpočtů přechodných dějů. Jestliže je nyní filtr v ustáleném stavu, uložíme si vektor jeho stavových veličin do souboru. Spustíme-li pak analýzu v režimu „*Read*“, vykreslí se přímo ustálený stav.

Volbou „*Transient/State Variables Editor*” se dostaneme do editoru stavových proměnných. Objeví se okno na **obr. 9.16**. Text “*Time* = 0.0002” připomíná, že se jedná o výpis stavových proměnných obvodu v čase 200 μs od zahájení posledního simulačního běhu. Klikneme do tlačítka “*Write*”. Na disk uložíme soubor **3D2X.TOP** se současným stavem obvodu. Pozor, soubor je třeba uložit do stejného adresáře, kde je vstupní soubor **3D2X.CIR**.



Obr. 9.15: Filtr v harmonickém ustáleném stavu, kterého bylo dosaženo opakovanou analýzou v režimu „*State Variables – Leave*“.



Obr. 9.16: Stavové proměnné, odpovídající počátečním podmínkám ustáleného stavu.

Nyní změníme v okně „*Transient Analysis Limits*“ položku „*State Variables*“ na „*Read*“. Po spuštění analýzy dostaneme přímo časové průběhy ustáleného stavu.

Z příkladu vyplývá, že k spouštění analýzy za přesně stanovených počátečních podmínek potřebujeme soubor s příponou **.TOP**, ve kterém jsou uloženy počáteční podmínky. Tento soubor je možno vyrobit v editoru stavových proměnných.

Někdy je výhodnější počáteční podmínky začlenit přímo do vstupního souboru. Pak k analýze nebude potřebný žádný další soubor. Následující postup využívá příkazu *.IC* („*Initial Conditions*“, počáteční podmínky).

Vyvoláme opět okno editoru stavových proměnných a klikneme na prvek „*IC*“. Objeví se informační okno s následujícím hlášením:

This command translates the voltages, currents, and states in this editor into .IC statements and places them in the text area of the current circuit.
Do you wish to continue?
Yes No

Tento příkaz převádí napětí, proudy a (logické) stavy z tohoto editoru do příkazů .IC a umísťuje je do složky „Text" aktuálního obvodu.
Přejete si pokračovat?
Ano Ne

Potvrdíme *Ano*. Horkou klávesou *F*3 uzavřeme všechna okna a vrátíme se do schématického editoru. V složce „*Text*" nalezneme následující příkazy:

```
.IC V(1)=3.43042e-007 V(2)=0.0514566 V(3)=-12.4771 V(4)=-12.3191 V(5)=-0.0309229
+  V(OUT)=-0.0815314 V(V1_PLUS)=-2.44921e-015
.IC I(L1)=0.000343042 I(L2)=0.000486991 I(L3)=-0.0255052 I(L4)=0.00855447
```

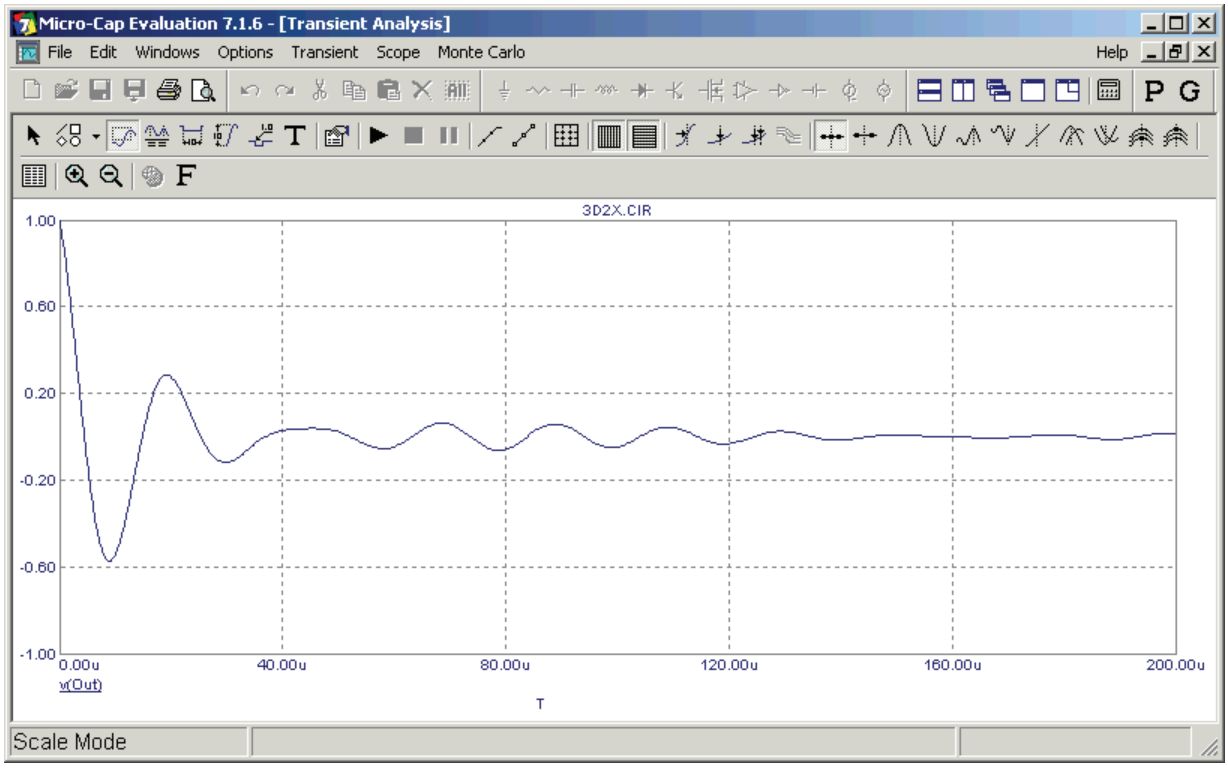
Význam příkazů *.IC* je intuitivně zřejmý. Na závěr příkladů provedeme shrnutí a zobecnění.

Zkusme spustit znovu časovou analýzu v režimu „*State Variables – Zero*". Program provede přímo analýzu ustáleného stavu.

Ukážeme užitečnost příkazu *.IC* ještě na dalším příkladu. U filtru na **obr. 9.12** řešme případ, kdy kapacitor C_4 na výstupu je nabit na počáteční napětí 1V. Jaký bude vývoj tohoto napětí při nepůsobení zdroje vstupního napětí?

Nejprve nahradíme zdroj vstupního signálu zkratem. Pak se přesuneme do složky „*Text*", vymažeme všechny příkazy *.IC* a místo nich umístíme tento text:

.IC V(out)=1



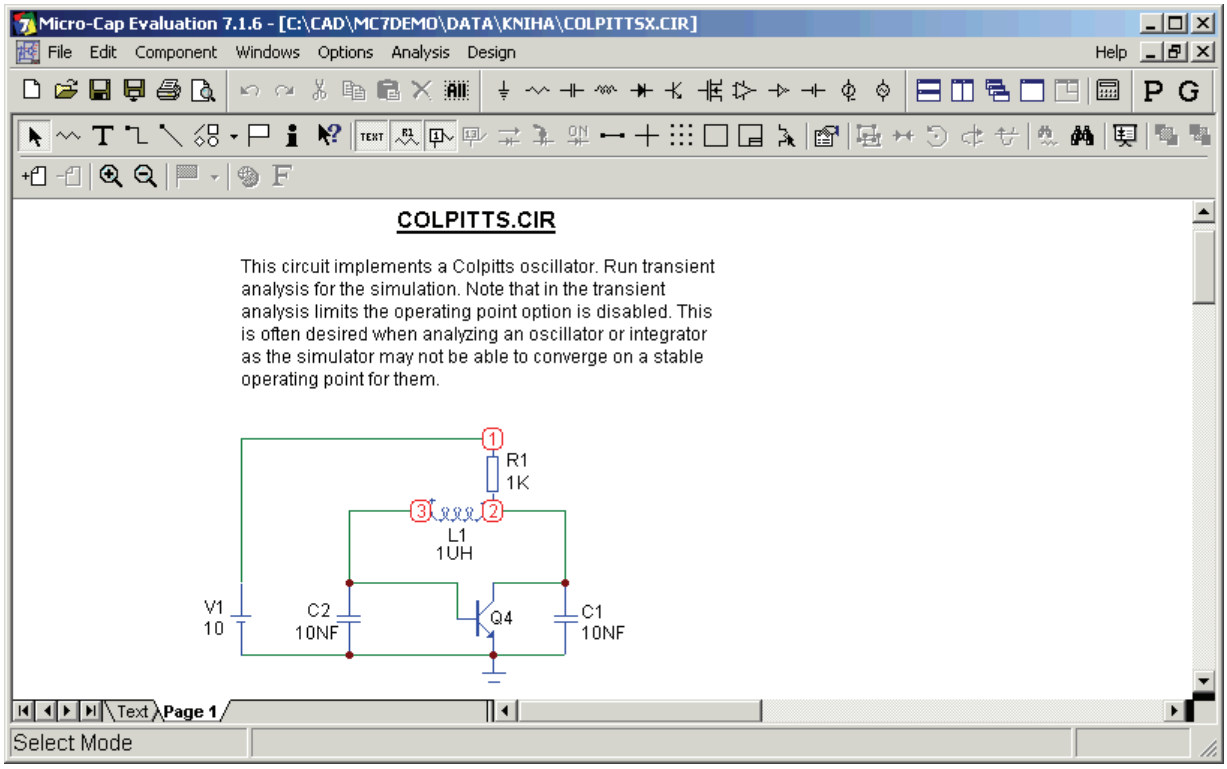
Obr. 9.17: Přirozená odezva filtru z **obr. 9.12** na počáteční napětí 1V na kapacitoru C_4.

Před analýzou je třeba vyblokovat kreslení křivky *V*(1) vymazáním jedničky ze sloupce „*P*“ (zkratováním vstupní brány stejně došlo k přečíslování uzlů). Analýza poskytne tentokrát výsledek podle **obr. 9.17**.

Dané nastavení počátečních podmínek je samozřejmě možné provést přímo pomocí editoru stavových proměnných.

Příklad 2 – analýza nasazování kmitů oscilátoru

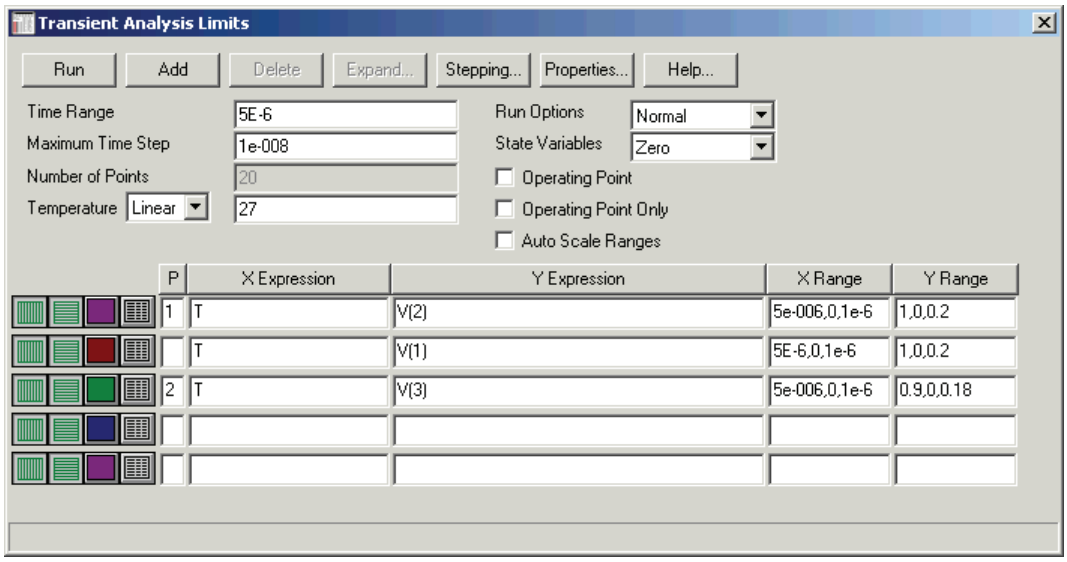
V adresáři *MC7DEMO/DATA/* vyhledáme soubor **COLPITTS.CIR** a načteme jej do editoru. Pak jej z bezpečnostních důvodů uložíme pod jiným jménem, například **COLLPITSx.CIR**. Zviditelníme čísla uzlů. Situaci znázorňuje **obr. 9.18**.



Obr. 9.18: Schéma Colpittsova oscilátoru.

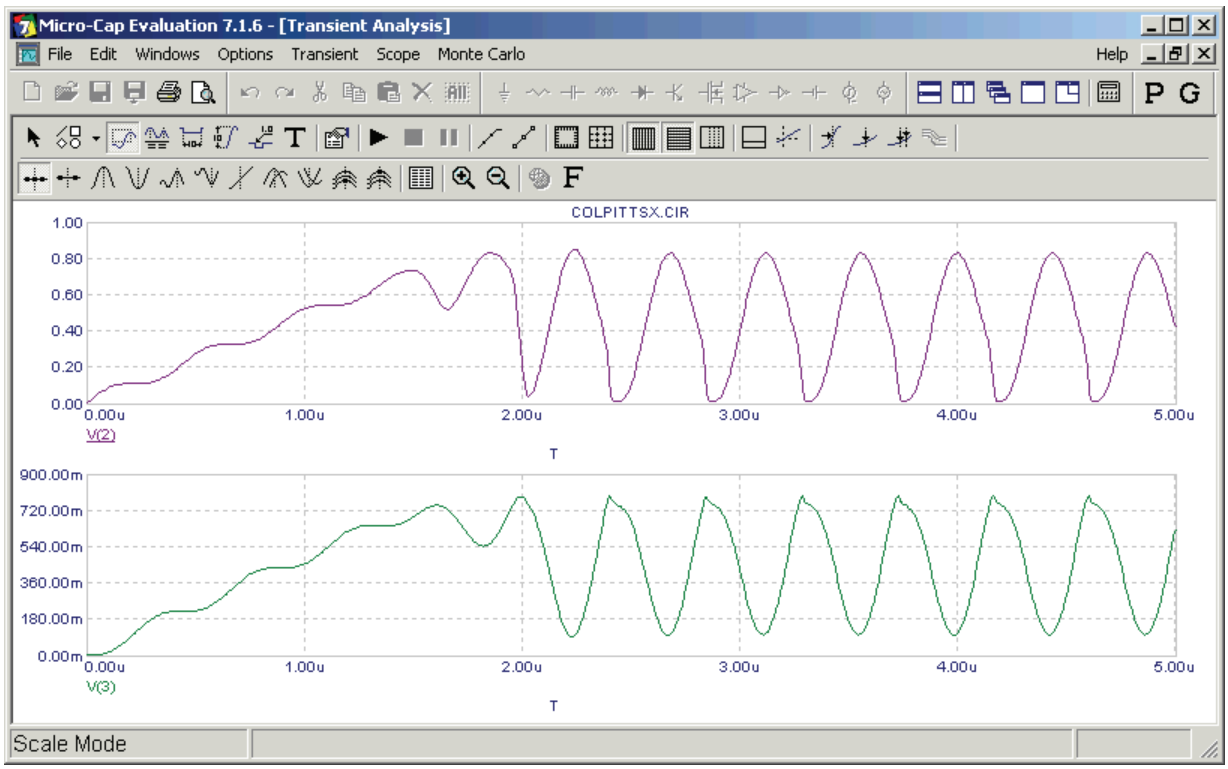
Jedná se o model Colpittsova oscilátoru. Naším úkolem bude prozkoumat, jak se bude obvod chovat po připojení baterie, konkrétně zda se z nulového počátečního stavu postupně dostane do kmitavého režimu. Jedná se o typickou úlohu č. *IV* z **Tab. 9.1** (nulové počáteční podmínky, vyblokování počítání stejnosměrného pracovního bodu).

Obsah okna „*Transient Analysis Limits*“ je přednastaven tak, jak vidíme na **obr. 9.19**. Do dvou obrázků se pod sebou vykreslí průběhy napětí *V*(2) a *V*(3), tj. napětí na kolektoru a bázi tranzistoru. Zkontrolujte, že počáteční podmínky jsou nulové a není požadován výpočet pracovního bodu.



Obr. 9.19: Nastavení podmínek časové analýzy k studiu rozběhu oscilátoru.

Výsledek analýzy je na **obr. 9.20**. Oscilátor naběhne do ustáleného režimu asi 2 μs po připojení napájecího zdroje.



Obr. 9.20: Náběh oscilátoru do kmitavého režimu.

Vyzkoušejte si přechod oscilátoru do ustálených kmitů v režimu „*Leave*“. Ověřte, že kmitočet je 2,3 MHz.

Příklad 3 – analýza ustálených kmitů krystalového oscilátoru

Analyzujte krystalový oscilátor z ukázkového příkladu ze souboru **XTAL1.CIR**. Krystal je definován makroobvodem. Nasazování kmitů trvá tak dlouho, že je nutná analýza v režimu „*State variables – Read*“.

9.3.8 Využívání příkazů .IC a .NODESET

V předchozích příkladech jsme měli možnost poznat výhody použití příkazu *.IC*. Tento příkaz má následující obecnou strukturu:

.IC V(*uzel*)=*hodnota_napeti*, nebo
.IC V(*uzel*1,*uzel*2)=*hodnota_napeti*, nebo
.IC I(*induktor*)=*hodnota_proudu*

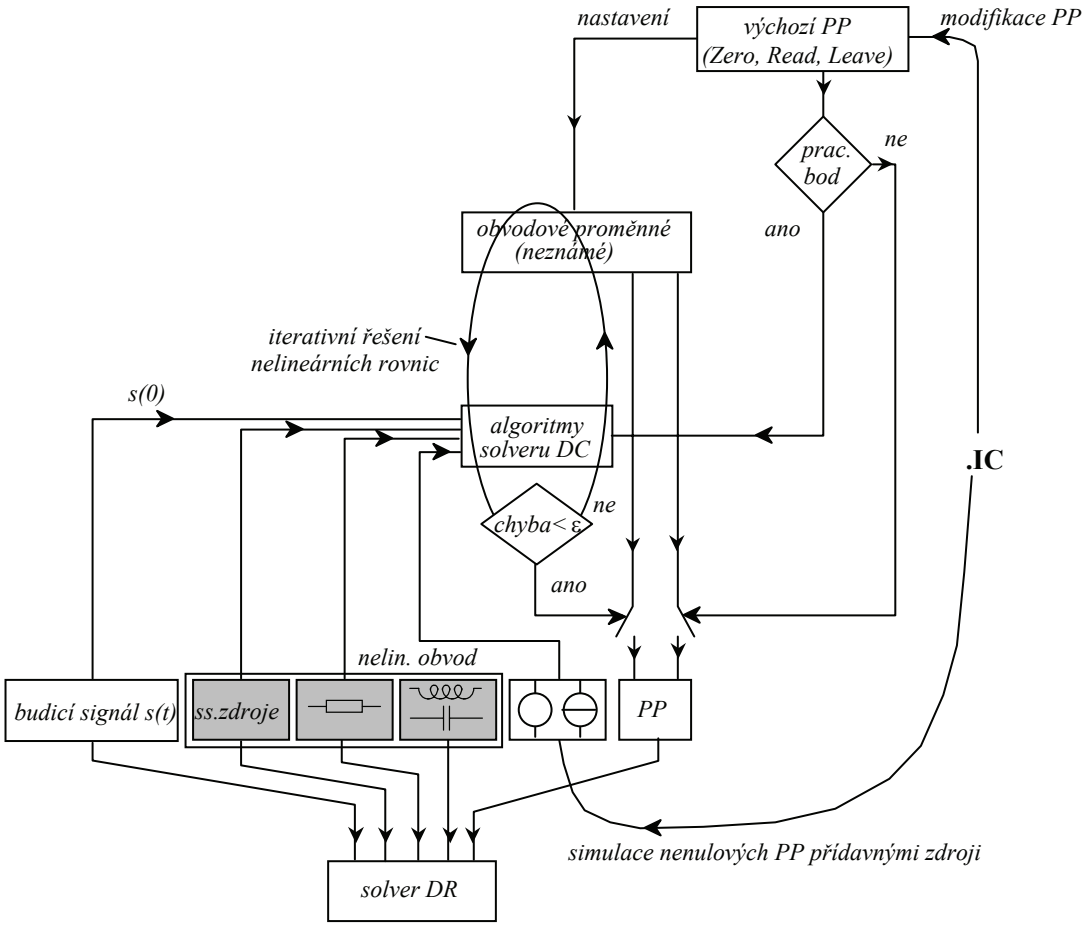
- První formou definujeme počáteční hodnotu konkrétního uzlového napětí.
- Druhou formou definujeme počáteční hodnotu napětí mezi dvěma uzly.
- Třetí formou definujeme počáteční hodnotu proudu konkrétním induktorem.

Jednotlivé formy lze sdružovat do jediného příkazu, např.

.IC V(out)=5 *I(L1)*=1*M*

Pokud simulátor počítá stejnosměrný pracovní bod, respektuje zadané počáteční podmínky tak, že mezi příslušné uzly umístí zdroje napětí a namísto induktorů zdroje proudu. Tyto zdroje jsou odstraněny až po nalezení pracovního bodu.

Pro podrobnější vysvětlení účinků příkazů *.IC* na průběh analýzy je původní **obr. 9.11**, objasňující mechanismus časové analýzy, rozkreslen do podoby na **obr. 9.21**.



Obr. 9.21: Účinky příkazů *.IC* na průběh časové analýzy.

Výchozí počáteční podmínky („*Zero*“, „*Read*“ nebo „*Leave*“) jsou příkazy *.IC* modifikovány. Například pokud je režim „*State variables – Zero*“ doplněn příkazem *.IC*

V(1)=1, pak uzlové napětí příslušející uzlu 1 je změněno z 0 V na 1 V. Toto nastavení se „překopíruje“ jako výchozí údaj do vektoru obvodových proměnných, které pro simulátor představují neznámé veličiny. Úkolem simulátoru je tyto veličiny vypočítat ze složité soustavy rovnic.

Pokud není povolen výpočet pracovního bodu, překopíruje se výchozí vektor obvodových proměnných do vektoru počátečních podmínek pro časovou analýzu. Uvažujeme-li například výše uvedené nastavení (stav „*State variables – Zero*“ je doplněn příkazem *.IC V*(1)=1), pak to znamená, že časové průběhy všech stavových proměnných budou vycházet z nuly kromě průběhu napětí uzlu 1, který bude mít počáteční hodnotu 1 V.

Je-li povolen výpočet pracovního bodu, doplní se obvod o zdroje napětí a proudu, které modelují vliv nenulových počátečních podmínek, zadaných příkazem *.IC*, vyloučí se vliv akumulačních prvků (kapacitory se rozpojí, induktory zkratují) a signálové zdroje se nahradí stejnosměrnými zdroji, jejichž napětí, resp. proudy se budou rovnat hodnotám těchto zdrojů v čase 0. Vznikne model odporového obvodu ve formě soustavy nelineárních algebraických rovnic, které jsou řešeny iterační metodou v solveru *DC*. Jestliže vektor neznámých splňuje ve dvou po sobě jdoucích iteracích zadané chybové kritérium, je obsah tohoto vektoru prohlášen za řešení a překopíruje se do vektoru počátečních podmínek pro časovou analýzu.

Časová analýza je pak provedena solverem *DR* (blokem pro řešení diferenciálních rovnic). V této fázi jsou již pomocné zdroje odpojeny a neovlivňují řešení.

Tímto mechanismem je zajištěno, že jednotlivé časové průběhy budou vždy vycházet z počátečních podmínek, které byly zadány.

Podobnou strukturu jako příkaz *.IC*, ale jiný význam, má příkaz *.NODESET*. Tento příkaz využije běžný uživatel simulátoru jen zřídka. Může nám pomoci v překonávání problémů s nalezením pracovního bodu. Sebedokonalejší program bude mít těžkosti při hledání stejnosměrného ustáleného stavu u obvodů typu bistabilní klopný obvod, kdy nepatrná změna v nastavení obvodových poměrů může vést k „přeskočení“ do jiného pracovního bodu (viz část 10.2.3). Protože program hledá řešení iterační metodou, můžeme příkazem *.NODESET* přednastavit vektor obvodových veličin tak „blízko“ k řešení, že tím usnadníme řešení nalézt. Lze říci, že tím předkládáme programu první odhad řešení.

Struktura příkazu .NODESET je následující:

.NODESET V(*uzel*)=*hodnota_napeti*, nebo
.IC I(*induktor*)=*hodnota_proudu*

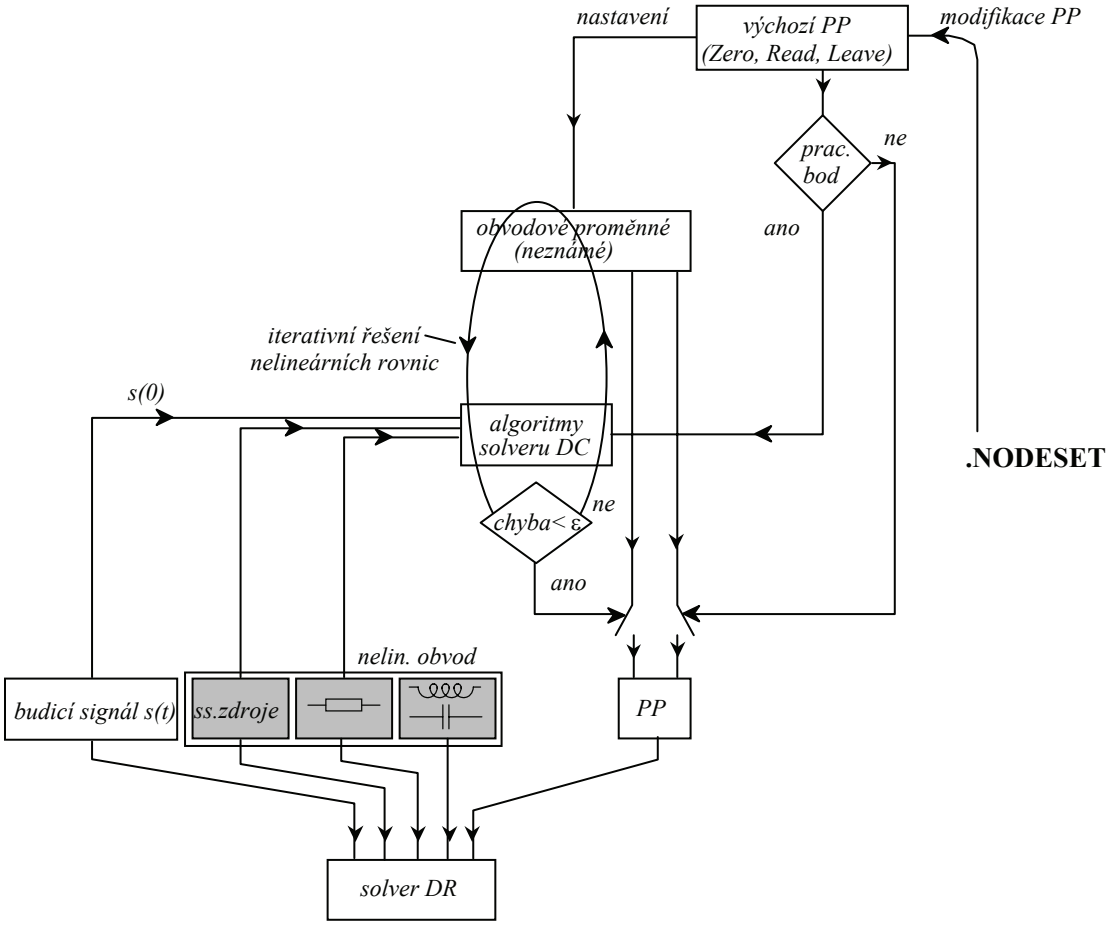
Obě formy lze opět jako u příkazu *.IC* sdružovat, například

.NODESET V(out)=5 *I(L*1)=1*M*

Rozdíl mezi příkazy *.NODESET* a *.IC* spočívá v tom, že zatímco příkaz *.IC* vnucuje počáteční podmínky simulátoru po celou dobu řešení pracovního bodu, příkaz *.NODESET* programu definuje pouze počáteční „násadu“ pro iterační řešení. Tento rozdíl vyplyne i po srovnání obrázků 9.21 a 9.22. Z **obr. 9.22** vyplývá, že příkaz *.NODESET* obecně ovlivňuje počáteční podmínky časové analýzy, není-li povolen výpočet pracovního bodu. Je však třeba upozornit, že příkaz *.NODESET* byl vyvinut právě pro překonávání problémů s konvergencí při hledání pracovních bodů, takže jeho použití v režimu zakázaného výpočtu pracovního bodu nemá praktický smysl.

Jestliže použijeme při simulaci současně příkazy *.IC* a *.NODESET*, pak prioritu mají příkazy *.IC*. Příkazy *.NODESET* jsou pak ignorovány.

Ještě je vhodné poznamenat, že nemá smysl používat příkazy *.IC* a *.NODESET* na uzly, kterým přísluší přesně definovaná napětí. Například příkaz *.IC V*(1,2)=10 se neprovede, pokud mezi uzly 1 a 2 je již zdroj napětí nebo třeba induktor, který je v počátečním stavu analýzy nahrazen zkratem.



Obr. 9.22: Účinky příkazů .NODESET na průběh časové analýzy.

9.3.9 Fourierova analýza

Simulátory obvodů většinou nabízejí v menu časové analýzy výpočet spektra analyzovaných signálů, případně další algoritmy jejich zpracování. Nejinak je tomu u MicroCapu.

Spektrum je počítáno pomocí algoritmu *FFT* („*Fast Fourier Transform*“, rychlá Fourierova transformace). Při používání této transformace je nutné dodržovat několik základních zásad, jinak hrozí nebezpečí, že spektrální složky signálu budou určeny nesprávně.

V této kapitole nejprve shrneme hlavní zásady spektrální analýzy. Pak si ukážeme, jak se tyto zásady promítají do prostředí simulačních programů. V závěru se seznámíme s některými vybranými speciálními funkcemi, určenými k dalšímu zpracování výsledků spektrální analýzy.

Algoritmus *FFT* a zásady jeho využívání v simulačních programech

Algoritmus *FFT*

je předpis, který přepočítává N vzorků signálu na N komplexních vzorků jeho kmitočtového spektra. Přitom N je třeba volit jako celočíselnou mocninu dvojky, tedy 2,4,8,16,32, … . Vzorky signálu musí být rovnoměrně rozloženy na časové ose. Pak vzorky spektra jsou rovnoměrně rozloženy na kmitočtové ose.

Z uvedeného plyne, že při výpočtu spektra nesmíme dovolit simulátoru použití automatické délky kroku. Krok musí být pevný a jeho velikost je třeba volit tak, aby počet vzorků signálu po dobu simulačního běhu byl roven celočíselné mocnině dvojky.

Algoritmus *FFT* „chápe“ navzorkovaný signál jako jednu periodu periodického signálu. Z délky trvání časové analýzy T („*Time Range*“) tedy automaticky vypočte opakovací kmitočet první harmonické $F_1 = 1/T$.

Je-li tedy předmětem analýzy skutečně periodický signál, je vhodné, aby se do analyzačního okna „vešla“ právě jedna opakovací perioda signálu. Přípustný je i celistvý násobek opakovacích period. Například při dvou periodách signálu T_S bude délka „simulačního okénka“ $T=2T_S$. Algoritmus FFT stanoví kmitočet první harmonické na

$$F_1 = \frac{1}{T} = \frac{1}{2T_S} = \frac{F_S}{2},$$

neboli na polovinu skutečné opakovací frekvence signálu. Výsledek bude takový, že 1., 3., 5., .. prostě každá druhá harmonická složka ve spektru vyjde nulová (analyzovaný signál na těchto kmitočtech nemá žádné spektrální složky). Ostatní spektrální čáry vyjdou správně. Přesnost výpočtu však obecně klesá, protože daný počet bodů N je zbytečně „vyčerpán“ na vykreslení několika period signálu a jediným výsledkem jsou „zbytečné“ nulové spektrální čáry ve výsledku. Proto se doporučuje analyzovat právě jednu periodu signálu.

Jakékoliv nedodržení zásady „celistvý počet period signálu v analyzačním okně“ má za následek nesprávný výpočet spektra a vznik „falešných“ spektrálních čar, které ve skutečnosti ve spektru nejsou obsaženy.

Pokud se obvod nachází v přechodném stavu, je třeba jej nejprve uvést do periodického ustáleného stavu buď v režimu „Leave“, nebo pomocí příkazu .IC. Jinak spektrální analýza povede na nesprávné výsledky.

Krok časové analýzy ΔT je vlastně vzdálenost sousedních vzorků signálu na časové ose, neboli tzv. *vzorkovací perioda*. Převrácená hodnota je *vzorkovací kmitočet* F_V, neboli počet vzorků za sekundu.

Spektrum je vypočteno přesně jen po dodržení tzv. vzorkovací poučky.

Vzorkovací poučka:

Spektrální složky signálu jsou vypočteny správně, zvolíme-li vzorkovací kmitočet větší, než je dvojnásobek hraničního kmitočtu, k němuž sahá spektrum analyzovaného signálu.

Většina technických signálů má hraniční kmitočet ve spektru neostrý. Pak je v zájmu přesnosti výpočtů vhodné volit vzorkovací kmitočet vyšší než činí dvojnásobek této „neostré hranice“. Čím vyšší, tím lépe. Avšak pak roste počet bodů N, takže je třeba volit kompromis mezi získanou přesností výsledků a nároky na hardware a výpočetní časy.

Například při analýze periodického signálu o opakovacím kmitočtu 1 kHz, který obsahuje 200 nezanedbatelných harmonických složek, sahá jeho spektrum do kmitočtu 200 kHz, takže je třeba volit vzorkovací kmitočet větší než 2x200k =400 kHz. Při analýze jedné opakovací periody signálu nastavíme na ose X simulační čas T=1 ms. Časový krok při vzorkovací frekvenci 400 kHz by vycházel

$$\Delta T = \frac{1}{400\ kHz} = 2{,}5\ \mu s\,.$$

Počet vzorků signálu, tj. počet bodů N, by vycházel

$$N = \frac{T}{\Delta T} = 400\,.$$

Zvolíme nejbližší vyšší celočíselnou mocninu dvojky, tedy N = 512.

Z matematické podstaty algoritmu *FFT* vyplývají dvě důležité vlastnosti výsledků *FFT*, jejich periodičnost a tzv. komplexně sdružená symetrie. Praktický důsledek je takový, že z celkového počtu N vzorků spektra je pro výpočet spektra použitelná jen první polovina, tj. pouze vzorky č. 0 až $N/2 - 1$. Druhá polovina vychází jako zrcadlový obraz první části a nesouvisí se skutečným průběhem spektra signálu.

Zadáme-li například simulátoru počítat 512-bodovou FFT signálu, neznamená to, že musíme chtít jako výsledek analýzy 512 harmonických složek. Počet harmonických zadáváme simulátoru zvlášť. Jsme však omezeni teoretickou hranicí N/2 = 256.

Výstupem algoritmu *FFT* je N komplexních čísel X_0, X_1, X_2, …, X_{N-1}. Bohužel to ještě nejsou matematické hodnoty spektrálních čar. Musíme provést přepočet:

$S_0 = \frac{1}{N} X_0$ … stejnosměrná složka signálu,

$S_k = \frac{2}{N} X_k,\ k = 1..N/2 - 1$ .. komplexní k-tá harmonická signálu. Modul S_k je amplituda, argument je počáteční fáze.

MicroCap má pro algoritmus FFT zavedenou funkci

FFT(signal)

Příklad jejího použití v režimu analýzy „Transient“:

FFT(v(out)) … vypočítá „X“ koeficienty napětí uzlu „out“.

Kromě toho je k dispozici další funkce HARM, která provádí přímo výše uvedený přepočet. Jejím výsledkem jsou „S“ koeficienty. Příklad použití:

HARM(v(out)) … vypočítá „S“ koeficienty napětí uzlu „out“, tj. přímo „komplexní“ spektrální čáry.

Pro manipulaci s komplexními čísly je pak zavedena řada dalších funkcí, z nichž jmenujme aspoň

REAL, IMAG, MAG, PHASE (reálná část, imaginární část, modul, argument komplexního čísla).

Dále platí užitečná vlastnost, že pokud se snažíme zobrazit do grafu na jednu z os komplexní číslo, automaticky se zobrazí jeho modul. Pak není třeba k zobrazení amplitudového spektra psát složitě

MAG(HARM(v(out))),

ale postačí zápis

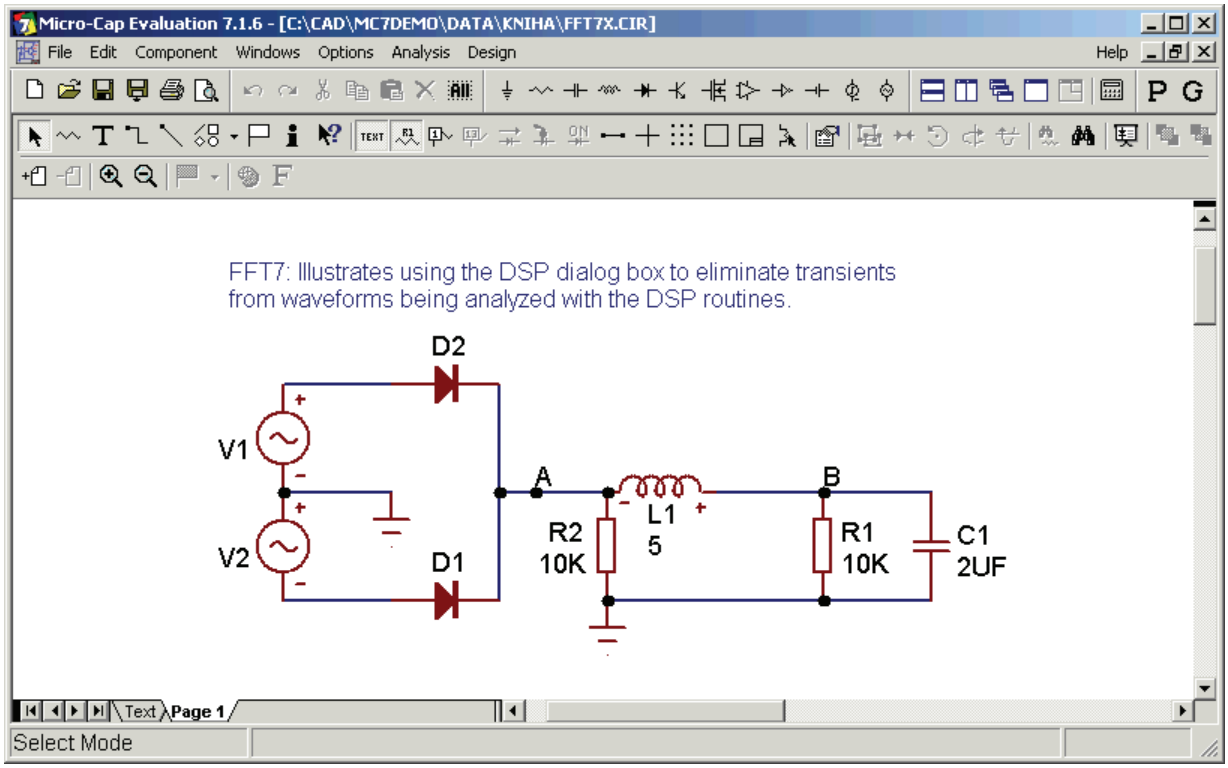
HARM(v(out)).

K zobrazení fázového spektra však již musíme zapsat

PHASE(HARM(v(out)))

Ilustrační příklad

Do editoru načteme soubor **FFT7.CIR**. Uložíme jeho kopii do souboru **FFT7x.CIR**. Soubor obsahuje model dvoucestného usměrňovače s vyhlazovacím *LC* filtrem. Nahlédnutím do složky „*Text*“ zjistíme, že zdroje napětí *V*1 a *V*2 jsou modelovány jako sinusové o amplitudě 100 V a kmitočtu 50 Hz.

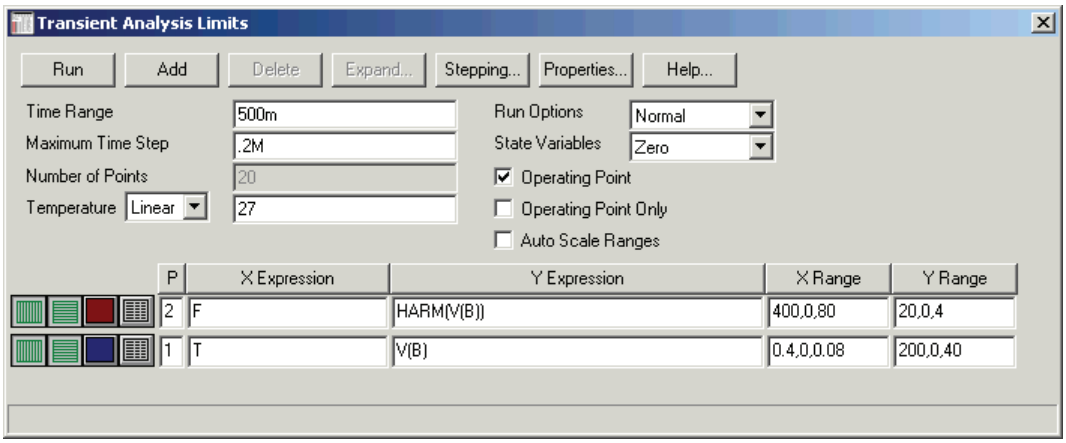


Obr. 9.23: Usměrňovač s *LC* filtrem.

Cílem časové analýzy bude prozkoumat časový průběh přechodného děje na výstupu po připojení zdrojů k usměrňovači, jakož i zvlnění výstupního napětí v ustáleném stavu. Ze spektrální analýzy tohoto zvlnění vyplynou některé závěry o vlastnostech vyhlazovacího filtru.

Vybereme časovou analýzu. Položky v okně „*Transient Analysis Limits*“ by měly být vyplněny podle **obr. 9.24**. Jsou nastaveny nulové počáteční podmínky, které jsou výchozí pro hledání pracovního bodu. Protože však oba zdroje v obvodu jsou sinusové, na počátku simulace jsou jejich napětí nulová. Proto vyjde pracovní bod o nulových souřadnicích a zatržení položky „*Operating Point*“ je vlastně zbytečné.

Časová analýza proběhne po dobu 500 *m*s. Protože po dvoucestném usměrnění bude mít zvlnění výstupního napětí kmitočet 100 Hz, „vejde“ se do analyzačního okénka 50 opakovacích period zvlněného napětí. Do obrázku č. 1 se zakreslí výstupní napětí $V(B)$, do druhého obrázku jeho spektrum. Zde se na chvíli zastavíme.



Obr. 9.24: Nastavení podmínek analýzy „Transient“ k zobrazení napětí $V(B)$ a jeho spektrálních čar.

Všimněte si, že v definičním řádku je v políčku „*X Expression*“ zapsán symbol *F*, tedy kmitočet, což je pro menu časové analýzy nezvyklé. V políčku „*Y Expression*“ je funkce pro výpočet harmonických složek *HARM*. Bylo již vysvětleno, že není třeba doplňovat vzorec o funkci *MAG* (modul, absolutní hodnota) a program vykreslí na svislou osu přímo amplitudové spektrální čáry. Údaj „*X Range*“ se týká měřítka na kmitočtové ose. Osa bude vynesena od kmitočtu 0 do kmitočtu 400 Hz s dělením po 80 Hz.

Zatím není zřejmé, jak zadat počet bodů *FFT*, tj. číslo *N*. Ve skutečnosti je *N* určováno položkami „*Time Range*“ a „*Maximum Time Step*“. Objeví-li se v okně „*Transient Analysis Limits*“ příkaz používající *FFT*, simulátor automaticky zvolí pevný časový krok. Jeho velikost určí tak, že vydělí dobu analýzy („*Time Range*“) maximálním časovým krokem („*Maximum Time Step*“) a výsledek převede na nejbližší vyšší celočíselnou mocninu dvojky. V našem případě vyjde

$$N > \frac{500}{0{,}2} = 1000 \Rightarrow N = 1024\,.$$

Skutečný časový krok bude

$$\Delta T = \frac{500}{1024} \doteq 0{,}488\ ms,$$

a vzorkovací kmitočet

$$F_V = \frac{1}{\Delta T} = 2{,}048 \ \ k\text{Hz}.$$

Spektrální analýza tedy povede na přesné výsledky, pokud analyzovaný signál bude mít spektrum omezeno do kmitočtu 1,024 *k*Hz.

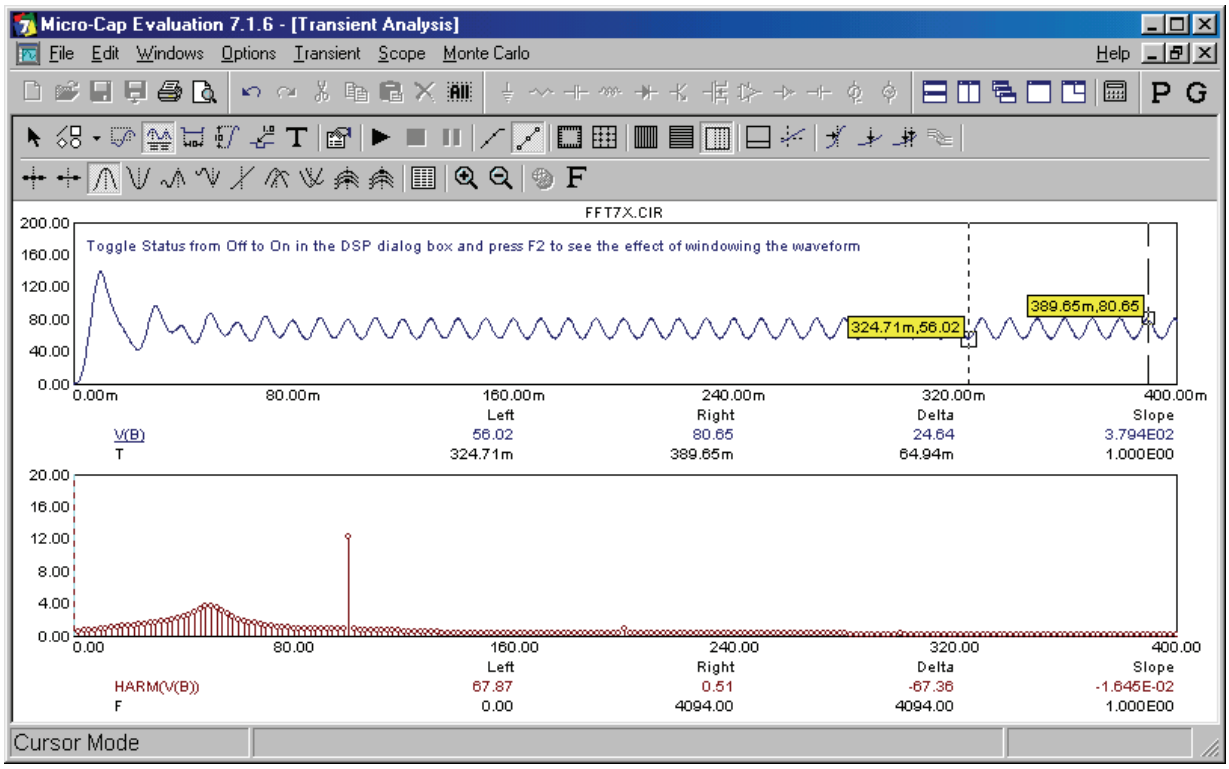
Po spuštění analýzy dostaneme výsledek na **obr. 9.25**.

Výstupní napětí usměrňovače vykazuje přechodovou složku, která odezní asi po 130 *m*s. Ustálená odezva je tvořena stejnosměrnou složkou o velikosti asi 68 V a střídavou složkou s rozkmitem asi 12,3 V na každou stranu kolem této hodnoty.

Níže vykreslené spektrální čáry přísluší periodickému signálu, jehož jedna perioda je vykreslena v horním obrázku č. 1. Tato „perioda“ je dlouhá 400 *m*s a odpovídá jí kmitočet první harmonické

$$1/400 \ m\text{s} = 2{,}5 \text{ Hz}.$$

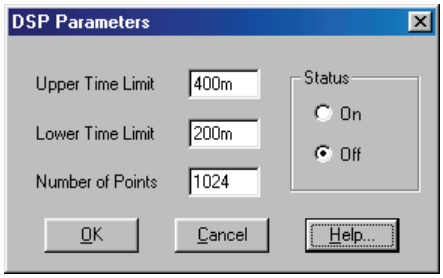
Tomu přísluší hustá síť spektrálních čar. Je zřejmé, že tento kmitočet je dán pouze volbou délky trvání simulace 400 *m*s a nemá žádnou souvislost s opakovací periodou střídavé složky signálu, která je 100 Hz. Ze spektra „vyčnívá“ významná spektrální čára na kmitočtu 100 Hz, která s přítomností této střídavé složky právě souvisí, a je patrná i čára na dvojnásobném kmitočtu 200 Hz, představující 2. harmonickou. Všechny ostatní spektrální čáry jsou vlastně parazitním důsledkem počátečního přechodného děje.



Obr. 9.25: Výsledek časové a spektrální analýzy. Výpočet spektrálních čar je zatížen principiální chybou, neboť obvod není v periodickém ustáleném stavu.

Pokusme se tedy vyeliminovat vliv počátečního přechodného děje tak, aby nebyl zahrnut do výpočtu spektra. Pak lze očekávat, že ve spektru bude kromě stejnosměrné složky pouze první harmonická na kmitočtu 100 Hz a její případné vyšší harmonické, pokud tvar střídavého zvlnění nebude přesně harmonický.

Text vepsaný v horním obrázku č. 1 (**obr. 9.25**) napovídá, jak na to. Klikneme na „*Transient*“ a poté vybereme „*DSP Parameters*“. Objeví se okno podle **obr. 9.26**.

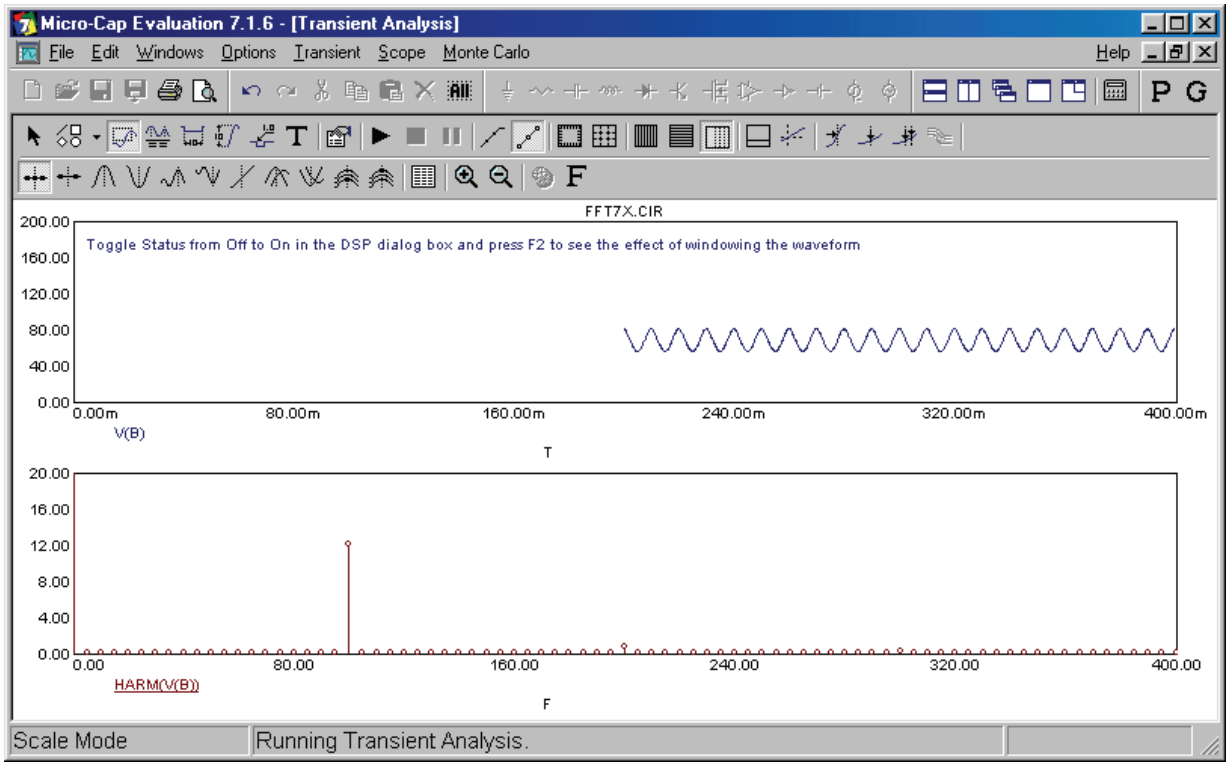


Obr. 9.26: Okno „DSP Parameters“ pro specifikaci podmínek analýzy *FFT*.

Pomocí tohoto okna vymezíme, jaká část časového průběhu (odkud – „*Lower Time Limit*“, kam – „*Upper Time Limit*“) bude použita k spektrální analýze. Položka „*Number of Points*“ udává, v kolika bodech bude signál v tomto novém okně převzorkován. Protože v našem případě sahá nové okno od 200 *m*s do 400 *m*s, zabírá polovinu původního okna, takže obsahuje 512 původně vypočtených bodů. Pokud je v okně „*DSP Parameters*“ uveden údaj 1024, program zřejmě provede převzorkování interpolovaného signálu.

Kliknutím nastavíme „*Status*“ na „*On*“ a potvrdíme „*OK*“. Program čeká na spuštění simulace, které provedeme horkou klávesou *F*2. Objeví se nový výsledek podle **obr. 9.27**.

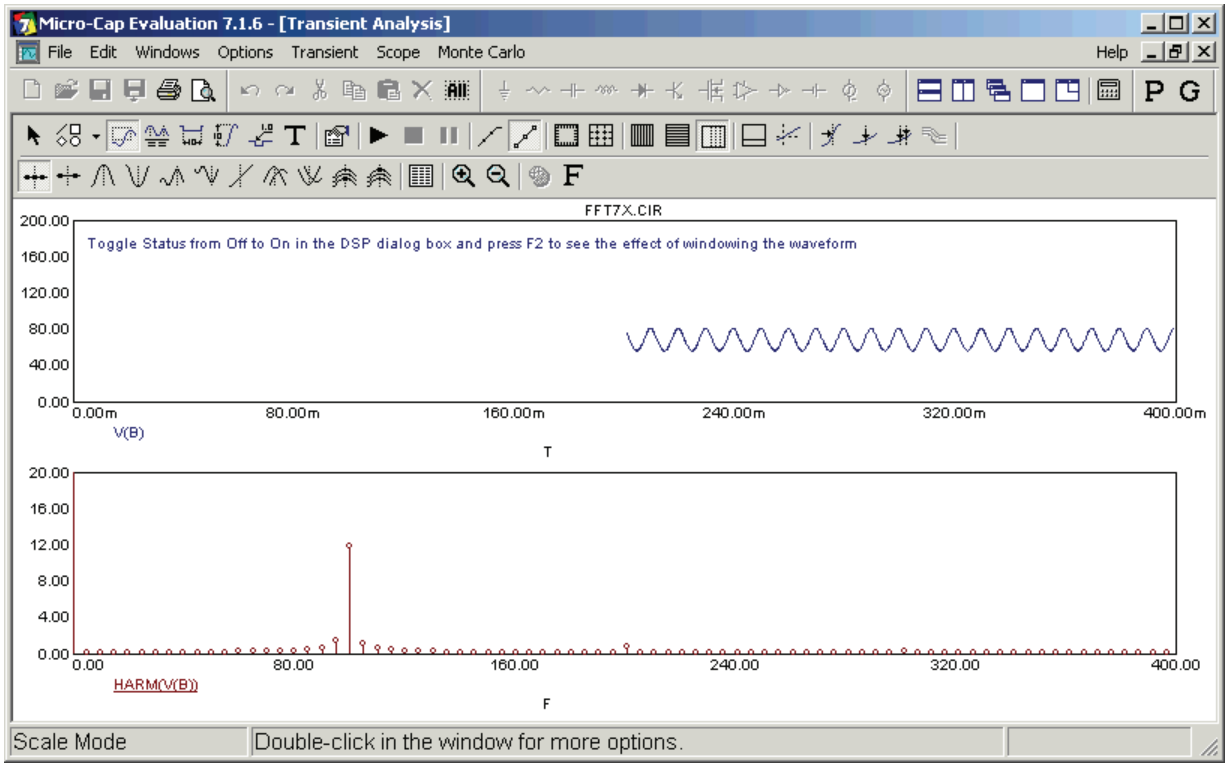
Do nového okna je nyní zahrnuto 20 opakovacích period ustálené odezvy. Ve spektru tomu odpovídá situace, kdy až 20. harmonická je nenulová a přísluší 1. harmonické ustálené odezvy. Čára je na kmitočtu 100 Hz. Na kmitočtu 200 Hz je mnohem menší 2 harmonická.



Obr. 9.27: Po „odříznutí“ přechodné složky signálu již spektrální analýza poskytuje správné výsledky.

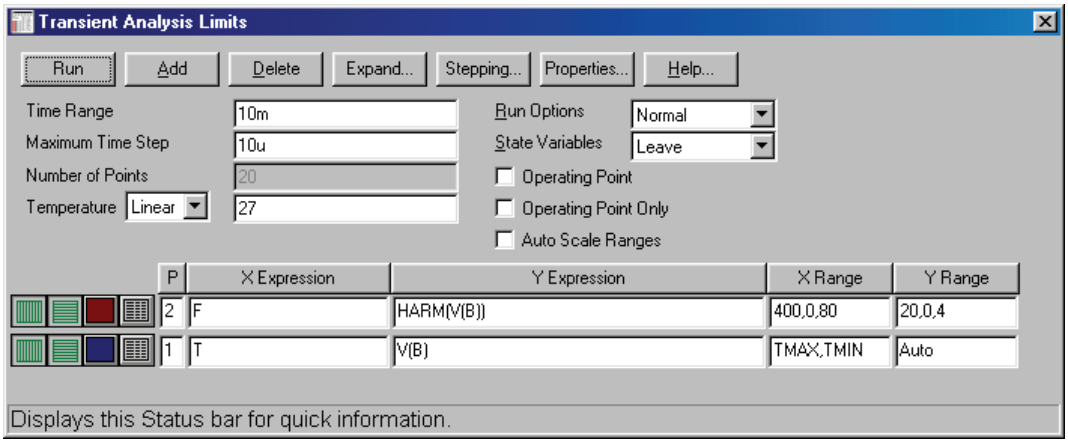
Vyzkoušejme si, co se stane, jestliže do výpočtu spektra nezahrneme celistvý počet opakovacích period. V okně „*DSP Parameters*“ změníme dolní hranici („*Lower Time Limit*“)

z 200 *m*s na 201 *m*s. Po provedení analýzy (*F*2) zjistíme, že se ve spektru objevily „falešné harmonické“ (viz **obr. 9.28**). Tento jev je známý z teorie číslicového zpracování signálů jako „*Leakage Phenomenon*“: je možné jej vysvětlit nespojitostí signálu v místech napojování sousedních priod. „Konec“ signálu v rámci opakovací periody se nekryje se „začátkem“ a vzniklý skok je příčinou obohacení spektra.



Obr. 9.28: Vznik „falešných harmonických“: signál neobsahuje celistvý počet opakovacích period střídavé složky.

Nyní provedeme spektrální analýzu „klasickým“ způsobem, bez použití okna „*DSP Parameters*“.

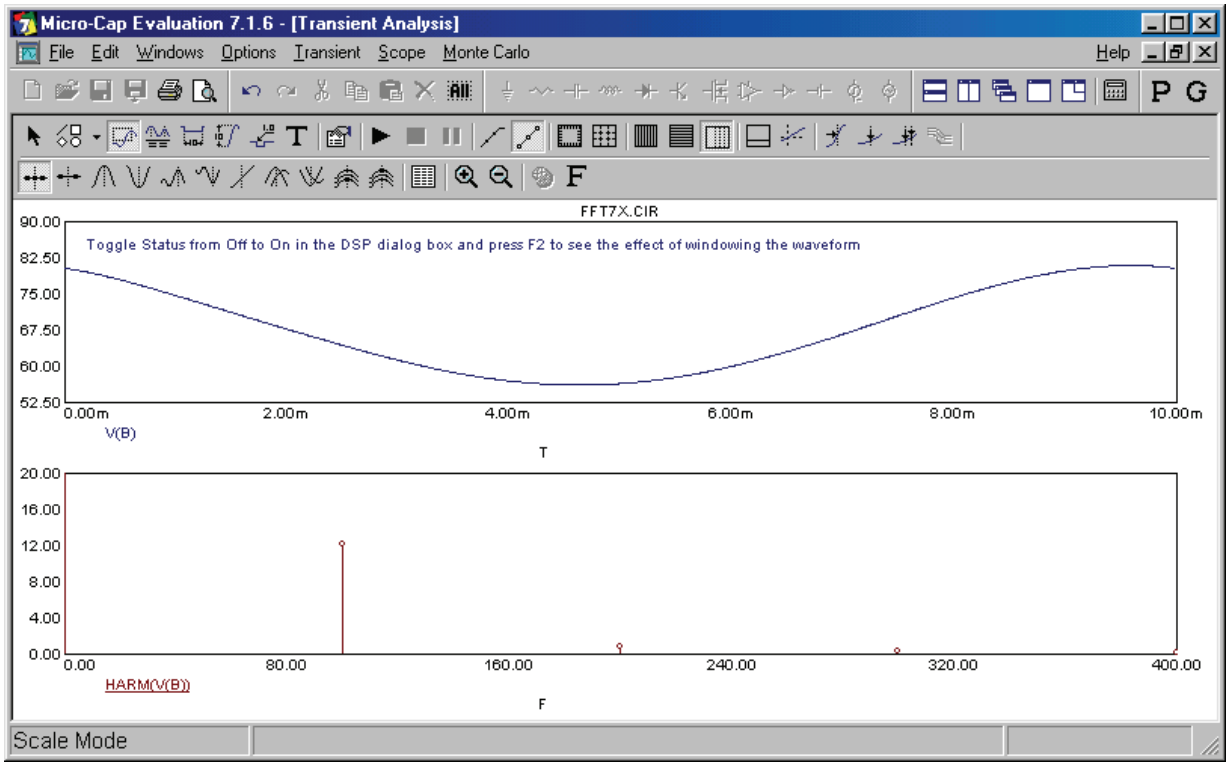


Obr. 9.29: Úprava podmínek časové analýzy pro dosažení správných výsledků spektrální analýzy (vyblokovaný výpočet pracovního bodu, stavové proměnné – *Leave*).

Nejprve v tomto okně nastavíme „*Status*“ zpět na „*Off*“. Pak upravíme okno „*Transient analysis Limits*“ podle **obr. 9.29**: Ustáleného stavu dosáhneme vyblokováním výpočtu pracovního bodu a nastavením počátečních podmínek do režimu „*Leave*“. Časový rozsah analýzy nastavíme tak, aby pokrýval jednu opakovací periodu signálu. Jestliže zvolíme

„*Maximum Time Step*“ 10 μs, nastavíme tím vlastně 1024 bodů *FFT*. Měřítko na svislé ose pro výstupní napětí nastavíme „*Auto*“.

Výsledek je na **obr. 9.30**. Nejsou zde již žádné „falešné“ nulové harmonické. Měřením v režimu „*Cursor*“ se můžete přesvědčit, že stejnosměrná složka výstupního napětí usměrňovače je 67,65 V, první harmonická má amplitudu 12,15 V, druhá 0,7 V, třetí 0,24 V. *RLC* filtr na výstupu usměrňovače propouští stejnosměrnou složku proudových impulsů z diod, a z harmonických složek propustí prakticky jen část první harmonické, která pak udává tvar zvlnění.



Obr. 9.30: Výsledek spektrální analýzy jedné opakovací periody signálu na výstupu usměrňovače.

Použití funkce THD

THD je zkratka anglického termínu „*Total Harmonic Distortion*“, česky „činitel harmonického zkreslení“. Je to číslo, udávající, nakolik se periodický signál tvarově liší od „čistého“ harmonického signálu. Ze spektrálního pohledu udává THD míru zastoupení první harmonické v celém spektru, neboť právě přítomnost vyšších harmonických složek ve spektru je indikátorem, že periodický signál se tvarově bude lišit od harmonické křivky.

Pokud program provede spektrální analýzu signálu, určí *THD* podle vzorce

$$THD = \frac{\sqrt{U_2^2 + U_3^2 + U_4^2 + \ldots}}{U_1}.100\,.$$

Výsledek je v procentech. Přitom symboly U_k, $k = 1, 2, 3, 4, \ldots$ udávají amplitudu k-té harmonické analyzovaného signálu.

Činitelem *THD* se často vyhodnocuje zkreslení signálů vyráběných oscilátory a *RC* generátory (pohybuje se řádově od 1% níže) a zkreslení, které vykazují zesilovače při zpracování harmonických signálů.

MicroCap vyhodnocuje *THD* pomocí funkce *THD*, která se musí aplikovat na vypočítané harmonické složky signálu. Základní syntaxe je následující:

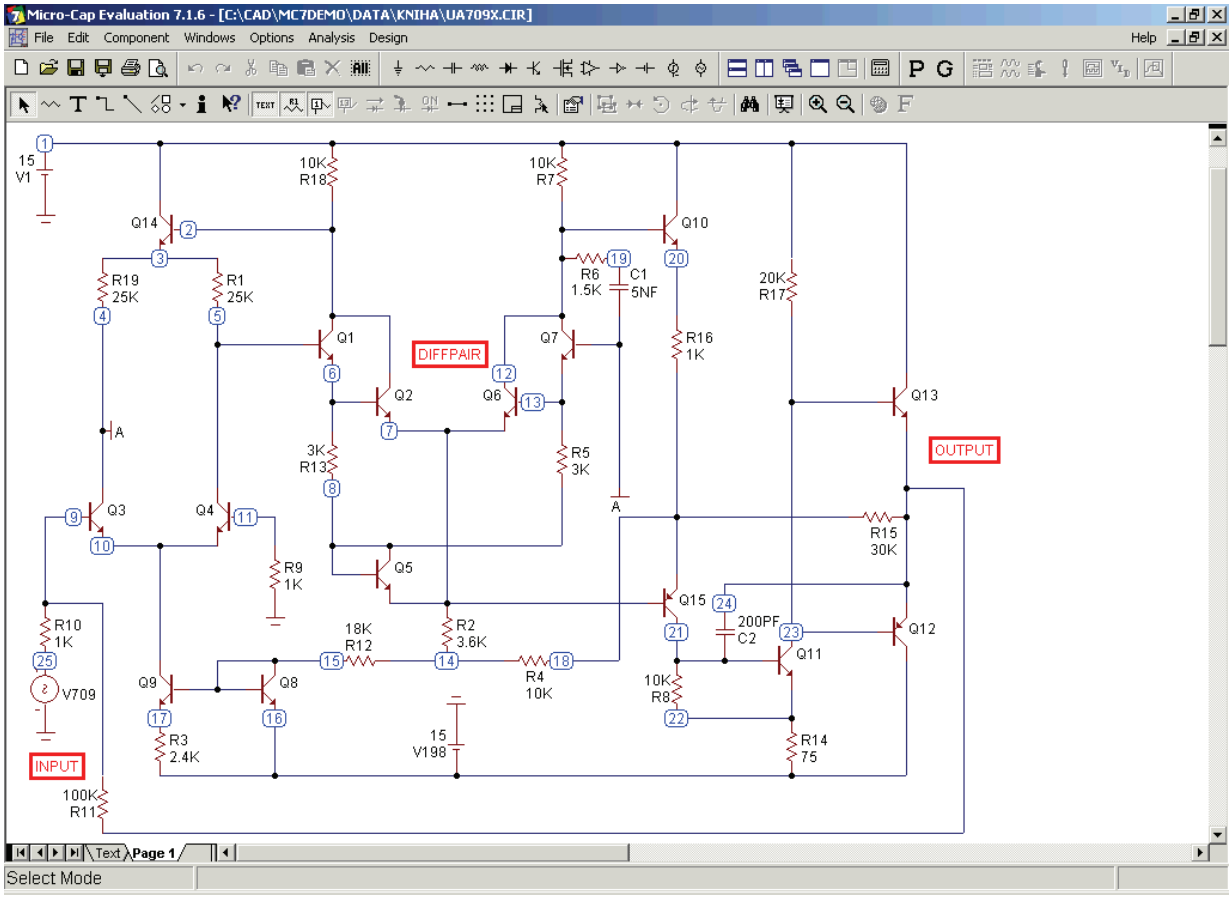
THD(HARM(signal))

Program považuje za první harmonickou tu složku, jejíž spektrální čára leží na kmitočtu *F*, kde *F* je převrácená hodnota doby trvání časové analýzy. Jinými slovy, výsledek výpočtu z výše uvedeného vzorce bude správně jen za předpokladu, že v okně časové analýzy bude právě jedna opakovací perioda signálu.

Je pamatováno i na případy více celistvých opakovacích period v časovém okně. Pak je třeba doplnit údaj o kmitočtu první harmonické F_1:

THD(HARM(signal),F1)

Použití funkce *THD* ukážeme na příkladu obvodu ze souboru **UA709.CIR** (viz **obr. 9.31**). Po jeho načtení do editoru si jej uložte do pomocného souboru **UA709x.CIR**.



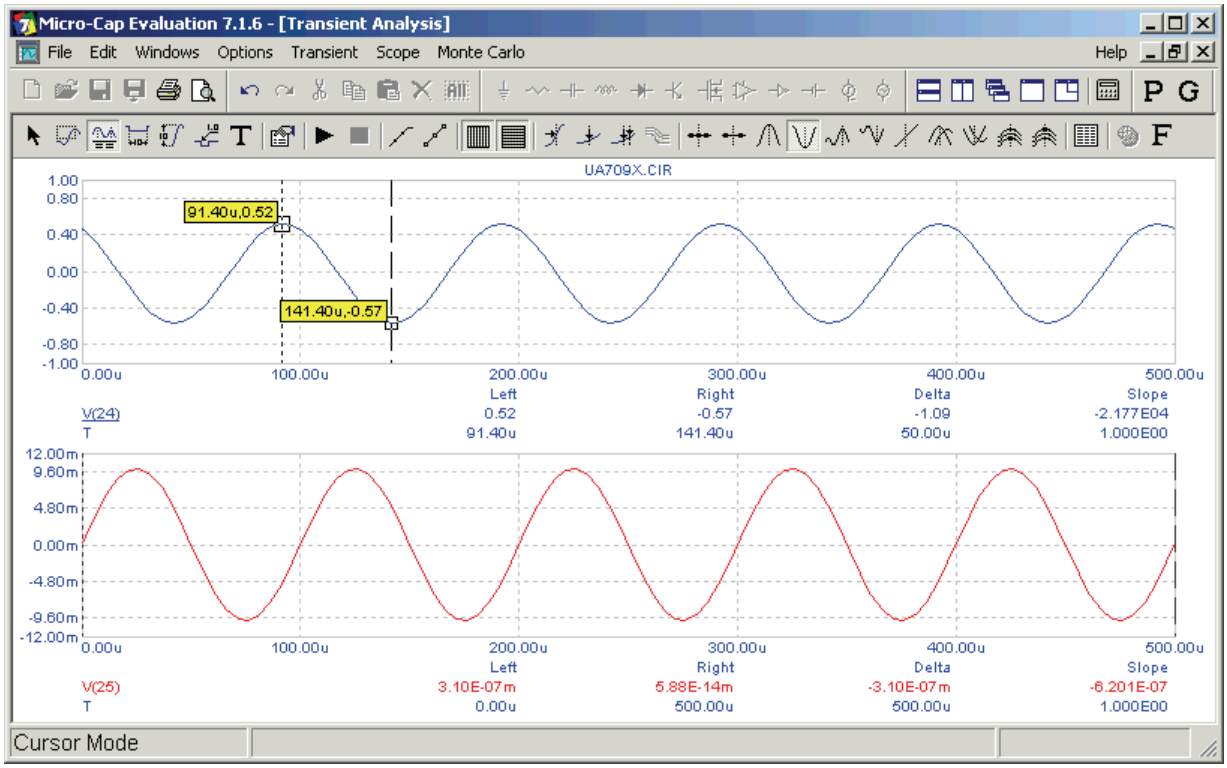
Obr. 9.31: Model operačního zesilovače µA709.

V souboru je model operačního zesilovače µA709 na tranzistorové úrovni. Tranzistory *Q*3 a *Q*4 tvoří vstupní rozdílový zesilovač. Neinvertující vstup operačního zesilovače je na uzlu č. 11, invertující na uzlu č. 9, výstup zesilovače je na uzlu č. 24. Vnější odpory *R*10 a *R*11 definují stejnosměrné zesílení

$$-R_{11}/R_{10} = -100\,.$$

Poklepáním na značku zdroje vstupního signálu *V*709 zjistíme, že signál je definován zdrojem typu „*V*“ (viz příloha P10.1.1) jako sinusový, s nulovou stejnosměrnou složkou, o amplitudě 10 *m*V a opakovacím kmitočtu 10 *k*Hz. Protože tranzitní kmitočet operačního zesilovače µA709 je pouhý 1 MHz, můžeme na kmitočtu 10 *k*Hz očekávat pokles reálného zesílení pod stejnosměrnou hodnotu 100.

Spustíme analýzu „*Transient*“. V okně „*Transient Analysis Limits*“ zakážeme výpočet pracovního bodu a počáteční podmínky nastavíme do režimu „*Leave*“. Několikrát po sobě spustíme analýzu, až dosáhneme ustáleného stavu. Výsledek by měl odpovídat **obr. 9.32**.



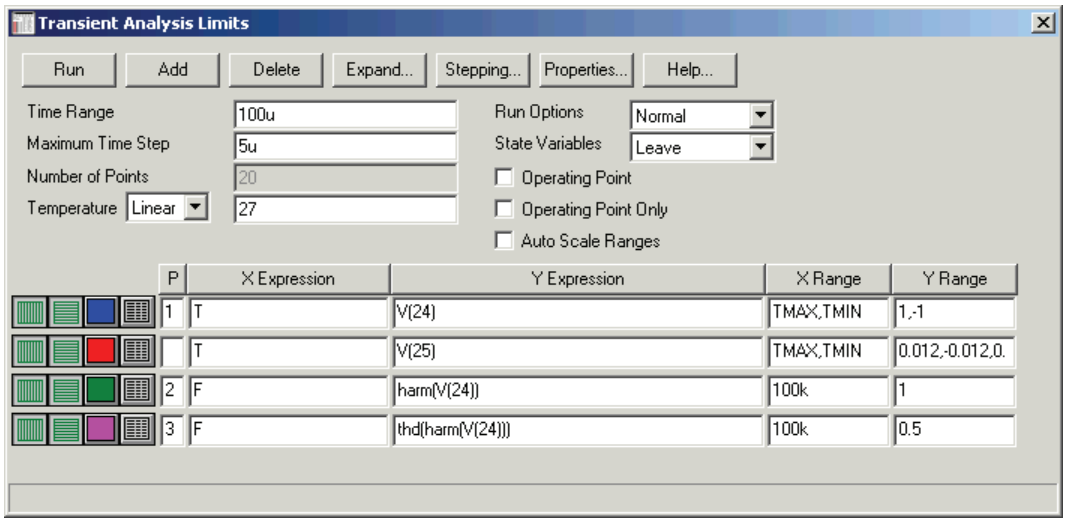
Obr. 9.32: Výsledek analýzy ustáleného stavu.

U výstupního napětí je v režimu „*Cursor*“ patrný mírný ofset. Napětí špička-špička je 1,09 V, čemuž odpovídá střídavé zesílení asi 50. Všimněme si, že fázový posuv mezi vstupním a výstupním signálem již není zdaleka 180 stupňů, jak by se „slušelo“ na invertující zesilovač.

Vrátíme se do okna „*Transient Analysis Limits*“ a upravíme některé položky podle **obr. 9.33**. „*Time Range*“ nastavíme na délku jedné opakovací periody. Zakážeme vykreslení vstupního signálu *V*(25). Přidáme příkazy pro výpočet harmonických složek výstupního signálu *V*(24) a koeficientu *THD*. Ponecháme-li obsah položky „*Maximum Time Step*“ na 5 µs, pak počet bodů *FFT* bude

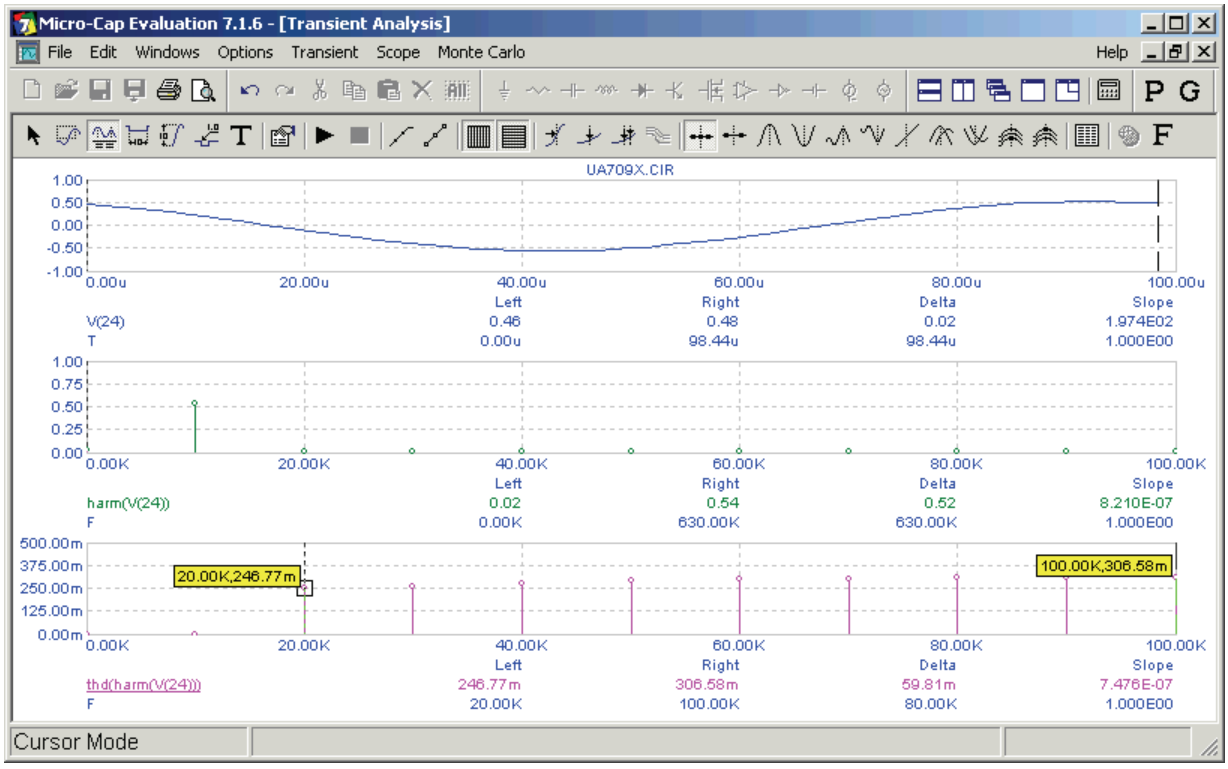
$$N > 100/5 = 20 \Rightarrow N = 32\,.$$

Tento počet bude dostatečný, protože analyzovaný signál nemá příliš bohaté spektrum. Z teoretického počtu 16 harmonických zobrazíme prvních 10.



Obr. 9.33: Úprava podmínek časové a spektrální analýzy.

Po proběhnutí analýzy by se měly objevit výsledky podle **obr. 9.34**.



Obr. 9.34: Výsledky spektrální analýzy zesilovače.

Interpretace spodního obrázku č. 3 je následující:

Jednotlivé „spektrální“ čáry souvisejí s postupným přidáváním dalších harmonických složek do čitatele definičního vztahu pro *THD*. Symboly v čitateli začínají až 2. harmonickou, takže v obrázku jsou pro kmitočty 0 Hz a 10 *k*Hz (stejnosměrná složka a 1. harmonická) nulové hodnoty *THD*. Na kmitočtu 2. harmonické, tedy 20 *k*Hz, do hry vstupuje první člen v čitateli, tj. U_2. Tomu odpovídá vypočtená hodnota 0,247 %. Na kmitočtu 30 *k*Hz přibude do čitatele člen U_3, atd. Protože hodnoty vyšších harmonických velmi rychle konvergují k nule, posloupnost takto vypočítávaných hodnot *THD* se ustaluje do výsledku, který odpovídá

definiční sumě. Z obrázku je zřejmé, že činitel harmonického zkreslení signálu na výstupu operačního zesilovače je asi 0,3 %.

Jako námět k samostatné práci doporučujeme zobrazit si spektrum signálu *V*(24) v logaritmické ose hodnot (je nutné změnit měřítko, dolní mez nesmí být nula), abyste si „zviditelnili“ vyšší harmonické, které se v lineárním měřítku zdají být prakticky nulové.

Dále si můžete určit *THD* u výstupního napětí krystalového oscilátoru ze souboru **XTAL1.CIR**. Pamatujte, že přesnost spektrální analýzy závisí na přesnosti vymezení opakovací periody signálu. Správný výsledek je cca 2%.

9.4 Analýza „*AC*“ neboli kmitočtová analýza

9.4.1 Cíle analýzy

Hlavním cílem kmitočtové analýzy je napodobování funkce „inteligentního obvodového analyzátoru“, tj. přístroje pro snímání kmitočtových charakteristik obvodů, pracujících v malosignálovém lineárním režimu. Přívlastkem „inteligentní“ vyjadřujeme řadu funkcí, které se zdají být z pohledu možností dnešních obvodových analyzátorů jako nadstandardní.

Další cíle analýzy: S režimem „*AC*“ analýzy bývá spojována šumová analýza „*Noise Analysis*“, která však využívá jiných matematických algoritmů, takže analýza šumových poměrů nemůže probíhat současně s analýzou kmitočtových charakteristik. Další cíle kmitočtové analýzy mohou být spojeny s různými operacemi nad získanými komplexními kmitočtovými charakteristikami, například ve spojení s algoritmy *DSP* („*Digital Signal Processing*“, tj. číslicové zpracování signálů).

9.4.2 „Inteligentní obvodový analyzátor“

Standardní obvodový analyzátor budí měřený objekt slabým kmitočtově rozmítaným harmonickým signálem, vyhodnocuje odezvu na buzení a vykresluje na obrazovce kmitočtovou závislost přenosových nebo impedančních funkcí. Při měření se obvod musí nacházet v lineárním režimu, tj. nesmí docházet k nelineárnímu zkreslení signálu vlivem přebuzení. To je obecně obtížný úkol, protože k přebuzení nesmí docházet v žádné části obvodu. Mnohdy však uživatel nemá dovnitř obvodu přístup nebo není v jeho moci „hlídat“ všechny důležité měřicí body. Nastavení extrémně slabého budicího signálu není vhodným řešením z hlediska šumových poměrů. Další známé problémy jsou spojeny s rychlostí rozmítání vstupu, zejména při proměřování v pásmu nízkých kmitočtů.

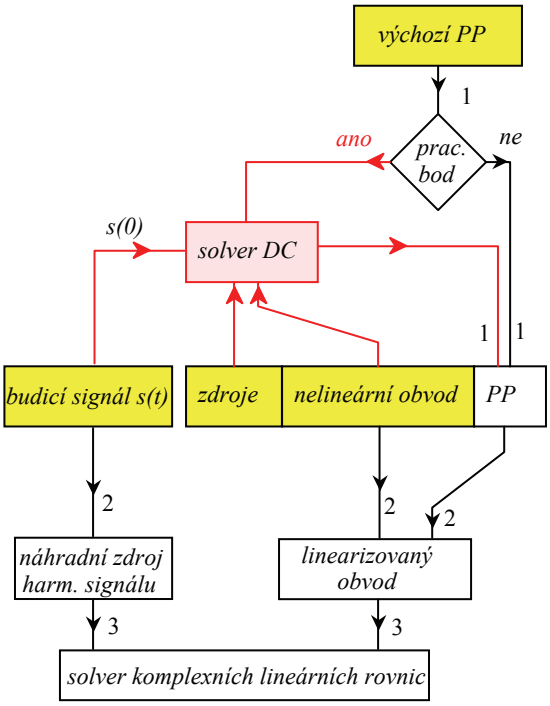
Z hlediska simulačního programu výše uvedené problémy v zásadě neexistují. Při analýze kmitočtových charakteristik se neberou v úvahu zdroje vnitřního ani vnějšího šumu. Problém přebuzení je zde vyloučen tím, že obvod se v okolí stejnosměrného pracovního bodu dopředu linearizuje matematickými algoritmy. Vlastní analýza probíhá již nad lineárním modelem, který prostě nelze „přebudit“. Výpočet bodů kmitočtové charakteristiky se uskutečňuje v cyklu, kdy kmitočet, obsažený v soustavě komplexních rovnic, je krokován od minimální po maximální hodnotu. Toto „kmitočtové rozmítání“ proto není zatíženo realizačními problémy, které jsou známy u obvodových analyzátorů.

Simulační program nabízí další možnosti při kmitočtové analýze, které by se ve světě reálných měření daly realizovat opravdu obtížně. Je zde například možné měnit polohu výchozího pracovního bodu a zkoumat, jaký to bude mít vliv na přenosové vlastnosti. Je možné zkoumat střídavé poměry ve všech uzlech a na všech individuálních součástkách „najednou“. Není nutné se omezovat jen na kmitočtové charakteristiky typu „přenos napětí“ nebo impedance, dosažitelné je vše, co lze popsat rovnicemi. Možnosti grafického vyjádření výsledků jsou rovněž značné (Nyquistovy komplexní kmitočtové charakteristiky, Smithův diagram atd.). A to nehovoříme o dalších nástrojích, které přímo nesouvisí s kmitočtovou analýzou, nicméně jich můžeme v rámci této analýzy využít (vyhodnocovací analýza, optimalizace, krokování parametrů a další).

Další potřebnou funkcí simulátoru je tzv. jednobodová analýza, konkrétně analýza střídavých poměrů v obvodu na jednom konkrétním kmitočtu. Nejvhodnějším zobrazením výsledků je rozložení napětí a proudů (amplitud a počátečních fází), případně komplexních výkonů přímo ve schématickém editoru, tak jak je to u stejnosměrné analýzy. Bohužel tuto funkci MicroCap (verze 7) nenabízí. Ze známých simulačních programů umožňuje práci v tomto režimu program TINA.

9.4.3 Jak postupuje simulátor při kmitočtové analýze

Na **obr. 9.35** je zjednodušené znázornění interních mechanismů, které působí v simulátoru při kmitočtové („*AC*“) analýze.



Obr. 9.35: Mechanismy kmitočtové analýzy v simulačním programu.

V první fázi simulace se stejně jako u časové analýzy naplní vektor počátečních podmínek (*PP*), v druhé fázi se sestaví linearizovaný model obvodu a budicí signál(y) se nahradí zdrojem (zdroji) harmonického signálu, v třetí fázi dojde k řešení linearizovaného obvodu pro zadaný rozsah kmitočtů.

Možnost nepovolení výpočtu pracovního bodu existuje jen při volbě počátečních podmínek v režimech „*Read*“ nebo „*Leave*“. Pokud je povolen výpočet pracovního bodu (implicitní nastavení), dojde k linearizaci obvodu kolem pracovního bodu, který je nalezen iterací z výchozích počátečních podmínek. V opačném případě se linearizace provede v okolí specifikované počáteční podmínky. Této možnosti využije běžný uživatel jen výjimečně. Je třeba upozornit na to, že přesnost linearizovaného modelu závisí na přesnosti souřadnic pracovního bodu, resp. počátečních podmínek. V dokumentaci MicroCapu je možné nalézt komentář v podobném smyslu: **jestliže zakážete výpočet pracovního bodu, pak lze jen doufat, že víte, co děláte.**

Pokud se jedná o příkazy *.NODESET* a *.IC*, pak pro jejich používání platí stejné zásady, které byly vyloženy u analýzy „*Transient*“ v části 9.3.8.

9.4.4 Atributy součástek při kmitočtové analýze

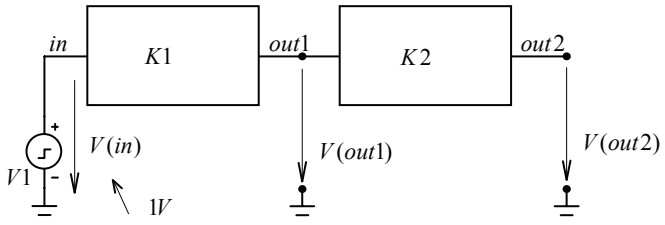
Při linearizaci se nelineární charakteristiky prvků nahrazují v okolí pracovního bodu jejich tečnami. Směrnice tečen se určí numerickou derivací. Parametry *R*, *L* a *C* lineárních rezistorů, induktorů a kapacitorů zůstávají zachovány.

U pasivních součástek *R*, *L* a *C* a u tzv. funkčních zdrojů napětí a proudu *NFV* a *NFI* (viz příloha P10.1.1) je možné zadat atribut „*FREQ*“ v editačním okně součástky. Vzorec, který zde zapíšeme, může obsahovat proměnnou *F*, tedy kmitočet. Tento vzorec definuje daný parametr výlučně pro analýzu „*AC*“. Pokud není položka „*FREQ*“ vyplněna, platí při analýze „*AC*“ klasické parametry definované v položce „*Value*“.

Na **obr. 9.35** je naznačeno, že zdroje signálu, ať jsou jakéhokoliv typu, jsou při analýze „*AC*“ nahrazeny zdroji harmonického signálu. Kmitočet je u všech zdrojů stejný, je generován centrálně při analýze linearizovaného modelu. Počáteční fáze a amplituda závisí na faktorech, které nyní popíšeme. Podrobnosti naleznete v příloze P10.1.3.

- U funkčních zdrojů *NFV* a *NFI* je amplituda určována pro každý kmitočet zvlášť vzorcem, který jsme zadali v položce „*FREQ*“. Počáteční fáze je 0 nebo 180 stupňů podle toho, zda vychází amplituda kladná nebo záporná. Pokud není položka „*FREQ*“ definována, tyto zdroje negenerují v analýze „*AC*“ žádné signály.
- U zdroje impulsů („*Pulse Source*“) a zdrojů harmonického signálu („*Sine Source*“) je amplituda automaticky 1 V.
- Zdroje „*V*“ a „*I*“ (původem z programu SPICE) mají své atributy „*AC magnitude*“ (amplituda ve voltech nebo ampérech) a „*AC Phase*“ (počáteční fáze ve stupních), které je třeba specifikovat.

Jestliže je v obvodu jediný zdroj signálu, připojený například mezi uzel 1 a uzel referenční, pak zápisu *V*(2) přísluší napětí, které se objeví mezi uzlem č. 2 a zemí, vyvolané působením zdroje signálu. Zápisem *V*(2)/*V*(1) definujeme přenos napětí z uzlu 1 do uzlu 2. Pokud je amplituda signálového zdroje jednotková a počáteční fáze 0, pak zápis *V*(2) je ekvivalentní zápisu *V*(2)/*V*(1). Toho se často využívá: jestliže obvod budíme jediným zdrojem typu „*Pulse Source*“ nebo „*Sine Source*“, pak k zadání požadavku na analýzu kmitočtové charakteristiky stačí zapsat *V*(*Y*), kde *Y* je jméno výstupního uzlu.



$$V(out1) = K1$$
$$V(out2) = K1.K2$$
$$V(out2)/V(out1) = K2$$

Obr. 9.36: Analyzovaný obvod rozdělený do kaskády bloků.

Obr. 9.36 dále ukazuje kaskádu dvou vzájemně se neovlivňujících bloků, které jsou popsány kmitočtovými charakteristikami *K*1 a *K*2. Požadujeme-li analyzovat charakteristiku bloku č.1, celkovou charakteristiku kaskády, nebo charakteristiku bloku č.2, musíme vložit požadavky na analýzu ve formě, uvedené v obrázku.

Jestliže ovšem v obvodu působí více signálových zdrojů, pak simulátor počítá napětí a další obvodové veličiny pomocí principu superpozice jako součet příspěvků jednotlivých

zdrojů. V tomto případě tedy nelze analyzovat kmitočtové charakteristiky přenosů z jednoho uzlu do druhého.

9.4.5 Menu "Frequency Analysis Limits"

V kapitole 9.2.4 bylo toto menu uvedeno na **obr. 9.3** b). Zde byl také vysvětlen význam těch položek, které jsou společné pro všechny základní analýzy. Nyní se zaměříme na ostatní položky, specifické pro analýzu „*AC*“. Jejich význam bude mít vztah k **obr. 9.35**, který ilustroval mechanismus kmitočtové analýzy.

„**Frequency Range**“ (kmitočtový rozsah).

Definice krajních hodnot kmitočtového rozsahu analýzy. Syntaxe je následující:

Fmax [,*Fmin*]

Pokud se uvede jen jeden údaj, proběhne výpočet jen na jednom kmitočtu. Pro hodnoty *Fmax* a *Fmin* existují tato omezení: $10^{30} > Fmax > Fmin > 0$.

Příklady:

10k	Proběhne výpočet v jediném bodě na kmitočtu 10 *k*Hz.
10k, 100	Kmitočet od 100 Hz do 10 *k*Hz.

Způsob vyplňování této položky závisí na tom, v jakém režimu je nastaven „*Frequency Step*“ (viz níže). Dosud uvedené přestává platit pro „*Frequency Step – List*“. Při tomto nastavení se do položky „*Frequency range*“ zapisují individuální kmitočty, oddělené čárkami, v nichž dojde k výpočtu. Přitom nezáleží na pořadí zápisu jednotlivých kmitočtů.

Příklad:

10*k*,100,5meg	Analýza proběhne v 3 bodech kmitočtové osy na kmitočtech 100 Hz, 10 *k*Hz a 5 MHz.

„**Frequency Step**“ (kmitočtový krok).

Program nabízí tyto možnosti:

Auto Výpočetní krok je řízen automaticky podle momentálního průběhu analyzované křivky. Hladkost křivky můžeme řídit obsahem položky „*Maximum Change %*“. Počet bodů výpočtu není dopředu znám, nelze tedy vyplnit položku „*Number of Points*“ (pokud není povolen výstup dat do textového souboru; je vysvětleno v části 9.2.4).
Tento režim je vhodný pro většinu řešených případů. Výjimky poznáme v části 9.4.7.

Linear Rozdíl kmitočtů, příslušejících každé dvojici sousedních bodů, je stejný a je roven hodnotě

$$\Delta F = \frac{F_{\max} - F_{\min}}{N - 1}.$$

N je celkový počet bodů výpočtu, který je zadán v položce „*Number of Points*“. Pak

$$F_i = F_{i-1} + \Delta F \text{ , } i = 1, 2, \ldots, N.$$

Tuto volbu kroku není výhodné používat při zobrazování *Bodeho* charakteristik, kdy volíme logaritmickou kmitočtovou osu. Lineární krok se používá výjimečně, například při zpětné Fourierově transformaci (viz část 9.4.9).

Log Podíl kmitočtů, příslušejících každé dvojici sousedních bodů, je konstantní a je roven

$$d = \left(\frac{F_{\max}}{F_{\min}} \right)^{\frac{1}{N-1}} \text{. Pak}$$

$$F_i = d.F_{i-1}, \; i = 1, 2, \ldots, N.$$

Tato volba se uplatní při zobrazování kmitočtových charakteristik na logaritmické kmitočtové ose. Většinou však dáme přednost režimu „*Auto*“.

List Seznam hodnot kmitočtů, oddělených čárkami (viz výše). Položka „*Number of Points*“ bude při analýze ignorována, uplatní se však při tvorbě případného textového datového souboru.

Noise Input (specifikace zdroje, na jehož svorky se přepočítává výstupní šum).

Noise Output (specifikace uzlů, mezi nimiž počítáme šumové poměry).

Význam těchto položek bude objasněn v části 9.4.8.

Na obr. 9.5 v části 4.8.2.4 je znázorněna řada 4 ikon, které jsou umístěny u jednotlivých řádků pro zadávání podmínek pro analýzy „*Transient*“ a „*DC*“. U analýzy „*AC*“ je zde ikona navíc, která slouží jako přepínač režimů zobrazení v těchto souřadnicích:

 pravoúhlé souřadnice (přednastaveno),

 polární souřadnice,

 Smithův diagram.

Při zadávání požadavků na analýzu v sloupcích „*X Expression*“ a „*Y Expression*“ (viz **obr. 9.3** b) je třeba dodržovat určitá pravidla. Některá z nich, která vyplývají ze způsobu práce s komplexními funkcemi, byla naznačena již v části 9.2.2 a 9.3.9. V následující části provedeme jejich celkové shrnutí a rozšíříme je o ukázku několika zabudovaných funkcí, které se při analýze „*AC*“ často používají.

9.4.6 Zásady pro práci s proměnnými u analýzy „*AC*“

V režimu analýzy „*AC*“ jsou napětí a proudy popsány fázory, tj. komplexními čísly. Požadujeme-li proto například analyzovat a vykreslit amplitudovou kmitočtovou charakteristiku, vyjadřující přenos napětí z uzlu 1 do uzlu 2, měli bychom do políčka „*X Expression*“ vepsat *F* (symbol rezervovaný pro kmitočet) a do políčka „*Y Expression*“ *MAG*(*V*(2)/*V*(1)), případně *MAG*(*V*(2)), je-li k vstupnímu uzlu připojen zdroj o jednotkové amplitudě. Zde *MAG* je funkce pro výpočet absolutní hodnoty (modulu) komplexního čísla, jak bude uvedeno dále.

V části 9.3.9 jsme poznali, že používání funkce *MAG* je nepovinné. Program nedokáže na osu *Y* vynášet „dvousložkové“ hodnoty komplexní funkce, vynese tedy automaticky modul.

Níže uvádíme funkce, zabudované v MicroCapu, které se často používají pro analýzu „*AC*“. Jejich argumentem je vždy komplexní číslo, které je označeno symbolem *z*. Podrobnější přehled funkcí naleznete v příloze P11.2.

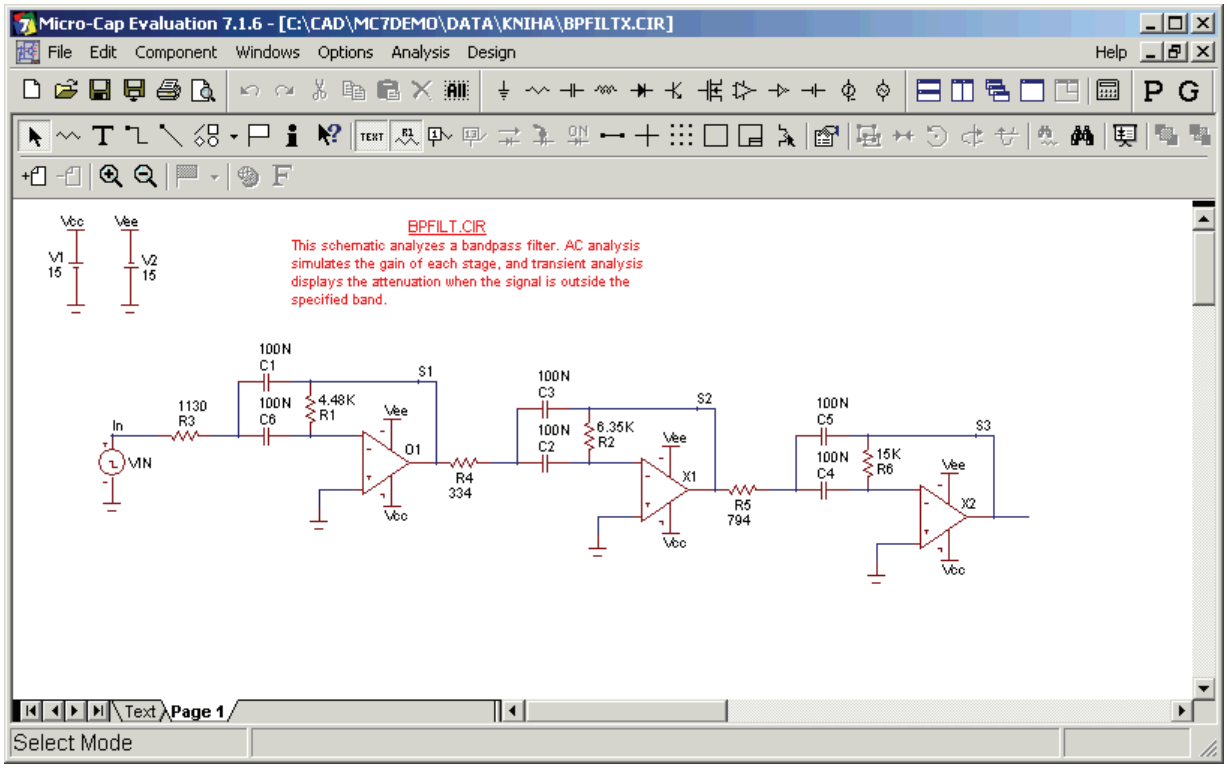
MAG(*z*)	modul
PH(*z*)	argument ve stupních
Re(*z*)	reálná část
Im(*z*)	imaginární část
DB(*z*)	modul v decibelech (20*log(*MAG*(*z*)))
GD(*z*)	skupinové zpoždění („*Group Delay*“, -*d*(*PH*(*z*))/*d*ω)

Jestliže například napíšeme do políčka „*X Expression*“ *Re*(*V*(2)) a do políčka „*Y Expression*“ *Im*(*V*(2)), získáme po proběhnutí analýzy obrazec komplexní kmitočtové charakteristiky.

9.4.7 Konkrétní příklady kmitočtové analýzy

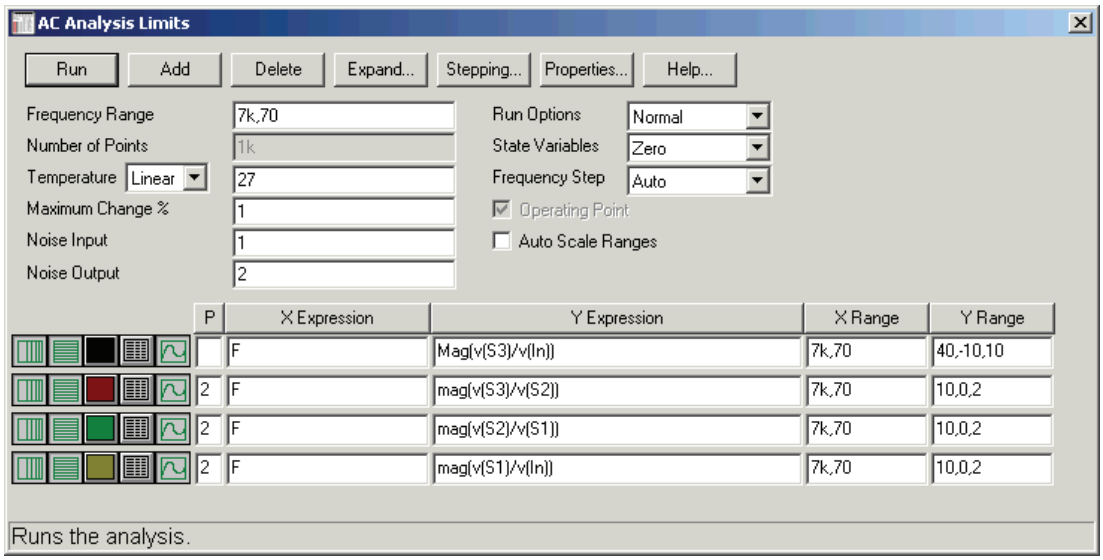
Příklad 1 - kaskádní aktivní filtr

V editoru otevřeme soubor **BPFILT.CIR** a uložíme si jej do zálohy jako **BPFILTx.CIR**. Jedná se o kaskádní filtr 6. řádu typu pásmová propust. Vstupní uzel má jméno „*in*“ a výstupní uzly jednotlivých bloků v kaskádě jsou „*S*1“, „*S*2“ a „*S*3“ (viz **obr. 9.37**).



Obr. 9.37: Kaskádní filtr typu pásmová propust.

Okno „*AC Analysis Limits*“ je přednastaveno tak, jak je ukázáno na **obr. 9.38**.

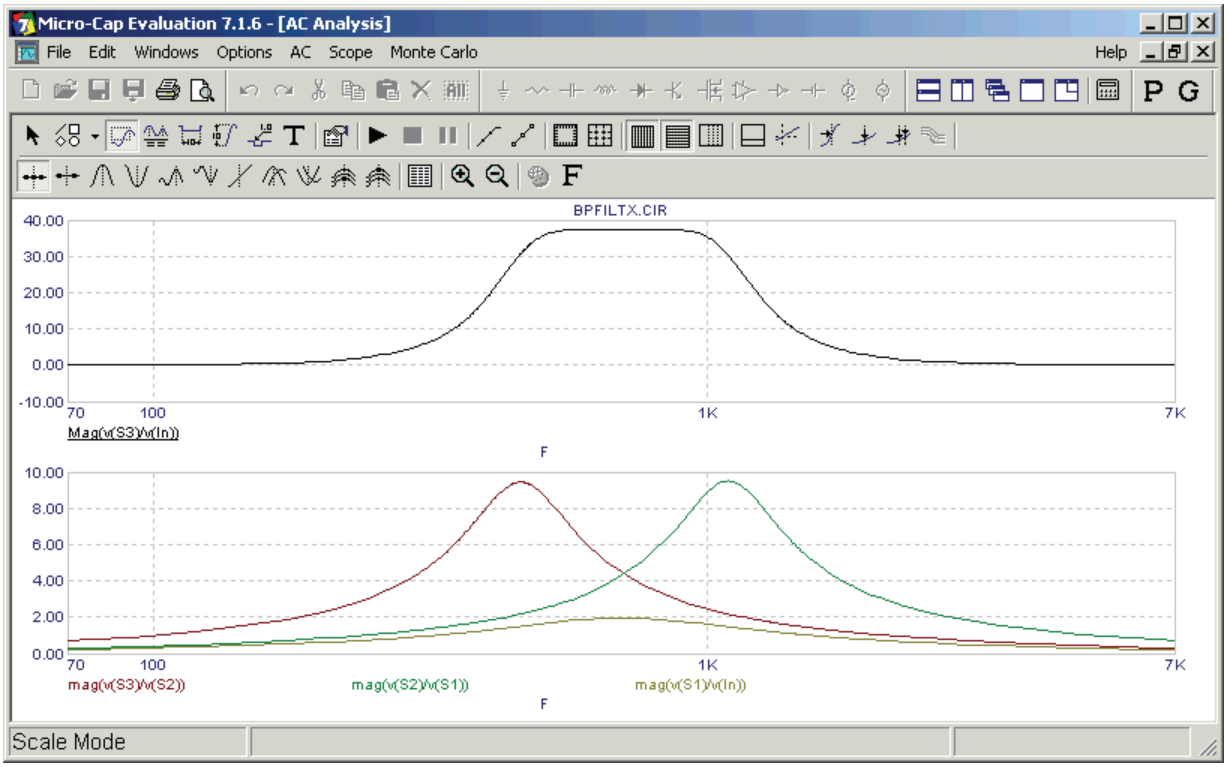


Obr. 9.38: Přednastavené položky pro analýzu „*AC*“.

Všimněme si, že v řádcích definujících analyzované závislosti je použito „předpisových“, nepříliš úsporných zápisů (lze vynechat funkci „*Mag*“ apod.).

V prvním řádku je definována amplitudová kmitočtová charakteristika celého filtru, nebude však vykreslena. V druhém, třetím a čtvrtém řádku jsou definovány zvlášť charakteristiky bloků č. 3, 2 a 1.

Zapíšeme do 1. řádku do sloupce „*P*“ jedničku (do obrázku č. 1 se vykreslí charakteristika celého filtru). Po spuštění analýzy dostaneme výsledky podle **obr. 9.39**.



Obr. 9.39: Výsledné kmitočtové charakteristiky.

Můžete se přesvědčit o správnosti zkrácených zápisů:

originální zápis	zkrácený zápis
Mag(*V*(*S*3)/*V*(*in*))	*V*(*S*3)
Mag(*V*(*S*3)/*V*(*S*2))	*V*(*S*3)/*V*(*S*2)
Mag(*V*(*S*2)/*V*(*S*1))	*V*(*S*2)/*V*(*S*1)
Mag(*V*(*S*1)/*V*(*in*))	*V*(*S*1)

Můžete si vyzkoušet i zobrazení *Bodeho* charakteristik s využitím funkce "*db*" a analýzu fázových kmitočtových charakteristik ("*ph*") a skupinového zpoždění ("*gd*"), případně komplexní kmitočtové charakteristiky.

Další doporučené příklady

3D2.CIR - Pasivní příčkový filtr. Všimněte si, že je nastaven pevný kmitočtový krok. Proč?
L1.CIR - Seznamte se se způsobem práce s *Laplaceovými zdroji* (viz též příloha P10.1.2).

9.4.8 Šumová analýza

Cíle šumové analýzy

Elektronické součástky, speciálně rezistory a polovodičové prvky, jsou zdroji vlastního šumu. Šumovou analýzou zjišťujeme, jak šumové příspěvky jednotlivých součástek pronikají na výstup obvodu. Výstupní šum pak lze přepočítat přes vstupně-výstupní přenos zpět na vstupní svorky obvodu. Porovnáním s úrovněmi užitečného signálu na výstupu či vstupu si vytvoříme představu o odstupu signálu od šumu.

Předmětem šumové analýzy jsou tedy výhradně vnitřní zdroje šumů a způsoby jejich šíření obvodem. Vstupní signály jsou uvažovány jako bezšumové.

MicroCap, stejně jako programy SPICE, modeluje tři druhy šumu: tepelný šum rezistorů, blikavý a výstřelový šum v polovodičích. Běžného uživatele obvykle nezajímá, jak jsou zdroje šumu modelovány, měl by však porozumět významu základních pojmů a obvodových veličin, které simulátor v souvislosti s šumovou analýzou počítá a jejichž grafy zobrazuje.

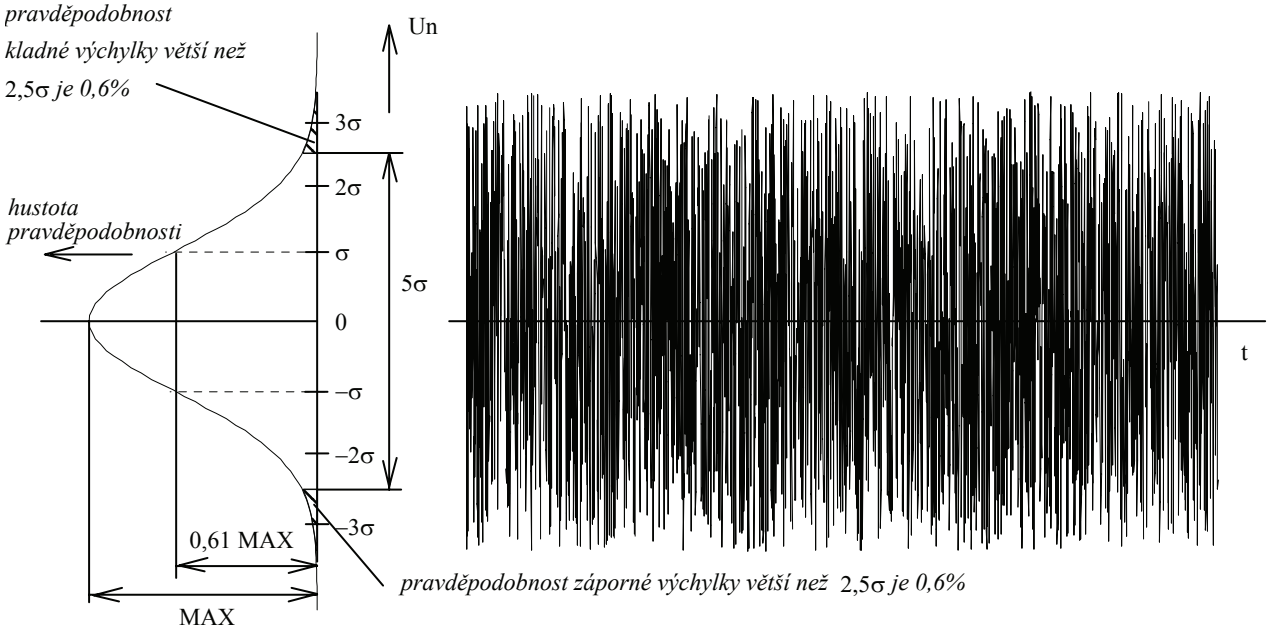
Jak lze charakterizovat šum

Šumová napětí a proudy, uvažované při počítačové šumové analýze, jsou signály, jejichž hodnoty se sice náhodně mění v čase, ale jejich statistické vlastnosti jsou stálé. Takové signály můžeme popsat buď jejich časovými průběhy $u_n(t)$ a $i_n(t)$, nebo tzv. spektrálními hustotami. Zatímco časový průběh je náhodný proces s dopředu nepředvídatelným průběhem, spektrální charakteristiky jsou v čase relativně stálé a dobře měřitelné. Lze z nich odvodit výkon šumu, soustředěný v určitém kmitočtovém pásmu, a z výkonu efektivní hodnotu a z ní pravděpodobnou mezivrcholovou hodnotu šumu. Simulátory proto pracují jen se spektrálními charakteristikami šumu, konkrétně s napěťovou spektrální hustotou (viz dále).

Typický záznam časového průběhu šumového signálu je na **obr. 9.40** [2.1]. V záznamu se nejčastěji vyskytují hodnoty v těsném okolí nuly. Čím větší hodnota, tím menší je pravděpodobnost jejího výskytu. Nejpravděpodobnější hodnota šumu je nula, což je současně průměrná – střední hodnota šumu, měřená v dostatečně dlouhém časovém úseku. Rozložení pravděpodobnosti výskytu jednotlivých hodnot šumu je poměrně věrně popsatelné tzv. hustotou pravděpodobnosti ve tvaru zvonovité Gaussovy křivky. Štíhlost křivky je řízena jejím parametrem σ, což je tzv. směrodatná odchylka šumu. Druhá mocnina směrodatné odchylky se nazývá disperze *D*. Z hlediska signálového je důležitější toto přiřazení:

Směrodatná odchylka a disperze šumu:

- Směrodatná odchylka σ je efektivní hodnota šumu.
- Disperze *D* je činný výkon šumu do jednotkového odporu.



Obr. 9.40: Typický záznam Gaussova šumu a jeho souvislost s hustotou pravděpodobnosti [2.1].

Význam Gaussovy křivky hustoty pravděpodobnosti je v tom, že ohraničuje plochu, která udává pravděpodobnost výskytu šumu v daném rozmezí hodnot. Plocha pod celou křivkou je jednotková, čemuž odpovídá stoprocentní pravděpodobnost, že mezivrcholová hodnota šumu se bude nacházet někde v intervalu šumových napětí (-∞, +∞). Z **obr. 9.40** vyplývá, že například pravděpodobnost, že šumový signál překročí hladiny ±2,5σ, neboli že jeho mezivrcholová hodnota nebude větší než 5σ, je jen asi 1,2%. Tato pravděpodobnost pak dále rychle klesá pro rostoucí mezivrcholové hodnoty. Pro 6σ je to například již jen 0,27% a pro 10σ vychází 6.10^{-5} %.

Představu o efektivní hodnotě šumu získáme ze spektrálních charakteristik, které vyhodnocuje simulační program (viz dále). Předchozí rozbor dává praktický odhad, jak získat představu o mezivrcholové hodnotě šumu z efektivní hodnoty:

Odhad mezivrcholové hodnoty šumu:

Mezivrcholová hodnota = (5 až 6) krát efektivní hodnota.

Tento odhad je platný s pravděpodobností kolem 98,8 % , resp. nad 99,7 % (pro pětku, resp. šestku).

Připomínáme, že pro harmonický signál platí násobitel $2.\sqrt{2} \doteq 2{,}8$ s pravděpodobností 100%.

Při šumové analýze jde především o analýzu výkonu šumu, který pak srovnáváme s výkonem užitečného signálu s cílem získání šumových charakteristik, jakými jsou například *SNR* („*Signal-to-Noise Ratio*“, poměr signál-šum), nebo šumové číslo [2.1]. Pro výpočet poměrů výkonů nepotřebujeme znát velikost zátěže, do níž signály pracují. Počítáme tedy s tzv. normovanými výkony do odporu 1 Ω. Normovaný výkon se pak počítá z druhé mocniny napětí nebo proudu. V obou případech vychází číselně stejně. Jeho jednotka je udávána buď $[A^2]$ nebo $[V^2]$.

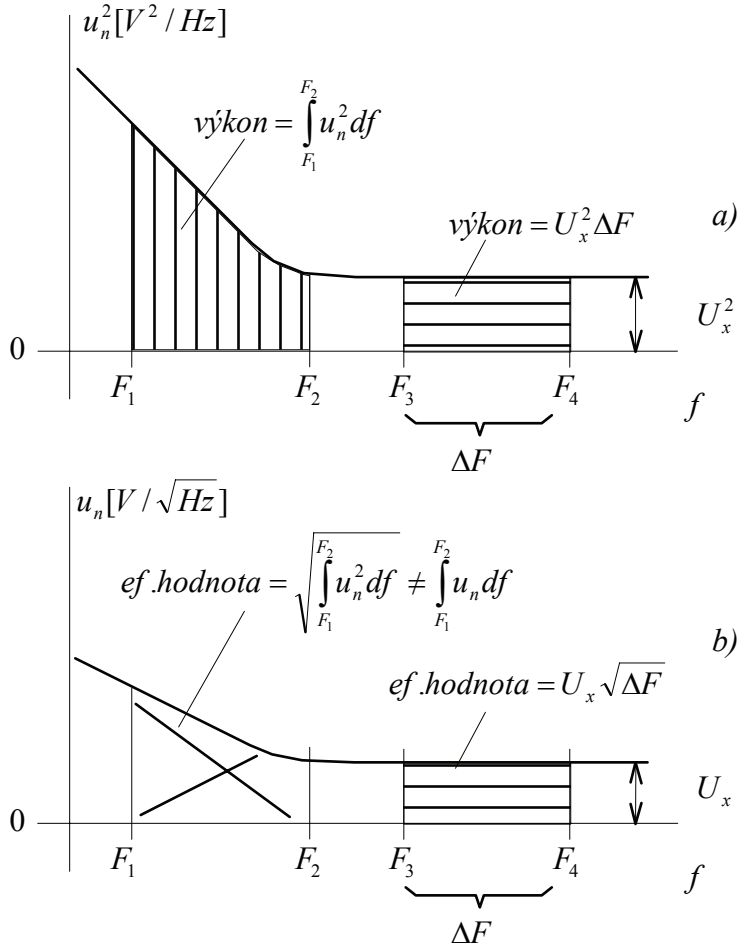
Při šumové analýze obvodů nás zajímá nejen celkový výkon, resp. efektivní hodnota šumu, ale i rozložení tohoto výkonu ve spektru. Ukázka typické křivky šumového výkonu polovodičového prvku je na **obr. 9.41** a). Na nízkých kmitočtech, do 1 Hz až asi 1 *k*Hz podle typu polovodičové součástky, se uplatňuje blikavý šum a další šumy charakteru „1/*F*“, jejichž výkon s rostoucím kmitočtem zaniká. Podstatnou úlohu zde hrají napěťové a proudové ofsety a drifty. V navazujícím kmitočtovém pásmu se uplatňuje zhruba konstantní úroveň šumu tepelného a výstřelového (tzv. bílý šum).

Výkon, soustředěný v daném pásmu kmitočtů, je dán plochou (integrálem), kterou nad tímto pásmem ohraničuje křivka šumového výkonu. Na svislou osu na **obr. 9.41** a) se tedy vynáší veličina u_n^2, jejíž rozměr je [V^2/Hz]. Nazývá se spektrální hustota výkonu (výkon šumu, nacházející se v elementárním kmitočtovém pásmu 1 Hz). V pásmu kmitočtů, kde je spektrální hustota konstantní, se výkon vypočítá jednoduše vynásobením spektrální hustoty šířkou pásma.

Na **obr. 9.41** b) je křivka tzv. spektrální hustoty napětí (efektivní hodnota šumu v elementárním kmitočtovém pásmu 1 Hz). Je to druhá odmocnina ze spektrální hustoty výkonu, což vede na zvláštní jednotku $[V/\sqrt{Hz}]$. Tato veličina se objevuje jako katalogový údaj řady polovodičových součástek. Z **obr. 9.41** b) vyplývá, že efektivní hodnotu šumového napětí vypočítáme vynásobením spektrální hustoty napětí druhou odmocninou z šířky pásma, v němž šum působí, ale jen za předpokladu konstantní spektrální hustoty. Při jiné kmitočtové závislosti šumového napětí, např. v pásmu nízkých kmitočtů, kde se projevuje šum 1/*F*, je nutné počítat efektivní hodnotu integrací.

Vykazuje-li např. operační zesilovač konstantní spektrální hustotu vstupního šumového napětí 6,5 *n*V/$\sqrt{Hz}$ od 10 Hz nahoru, pak šum „posbíraný“ v širokém pásmu od 10 Hz do 10 *k*Hz bude mít efektivní hodnotu asi 0,65 μV a mezivrcholovou hodnotu asi 3,25 μV (efektivní krát 5).

Hlavním výstupem počítačové simulace šumových poměrů v obvodu je křivka spektrální hustoty zadaného uzlového napětí. Z této křivky je pak možno integrací získat výkon a efektivní hodnotu šumu v různých kmitočtových pásmech.



Obr. 9.41: Typický průběh křivky spektrální hustoty a) výkonu, b) šumového napětí polovodičového prvku. Vodorovná i svislá osa jsou u obou obrázků logaritmické.

Jak se promítá šum součástek do výstupu obvodu

Šumová analýza je postavena na předpokladu, že šumové signály jsou natolik slabé, že jejich průchod obvodem nevyvolá v obvodu nelineární efekty. Pro analýzu, jakým způsobem jednotlivé „šumící“ součástky vytvářejí výsledný šumový signál na výstupních svorkách, se tedy používá linearizovaný model obvodu.

Představme si „šumicí“ součástku, například rezistor. Modelování šumu je realizováno paralelně připojeným zdrojem proudu, který vyvolává na součástce šumové napětí o spektrální hustotě napětí $u_n(f)$. Uzly této součástky A, B jsou s výstupními svorkami obvodu X, Y propojeny dalšími součástkami. Průchod šumového napětí na výstupní svorky je dán kmitočtovou charakteristikou $K(f)$, měřenou při průchodu signálu z brány A-B do brány X-Y. Z teorie vyplývá, že spektrální hustota napětí na výstupu $u_{n,out}(f)$ pak bude

$$u_{n,out}(f) = u_n(f).\left|K(f)\right|. \tag{9.1}$$

Jestliže je v obvodu více zdrojů šumu o spektrálních hustotách napětí $u_{n1}(f)$, $u_{n2}(f)$, …, jejich příspěvky se na výstupu sčítají. Sčítají se však spektrální hustoty výkonu, nikoliv napětí:

$$u_{n,out}^2(f) = u_{n1}^2\left|K_1(f)\right|^2 + u_{n2}^2\left|K_2(f)\right|^2 + \ldots,$$

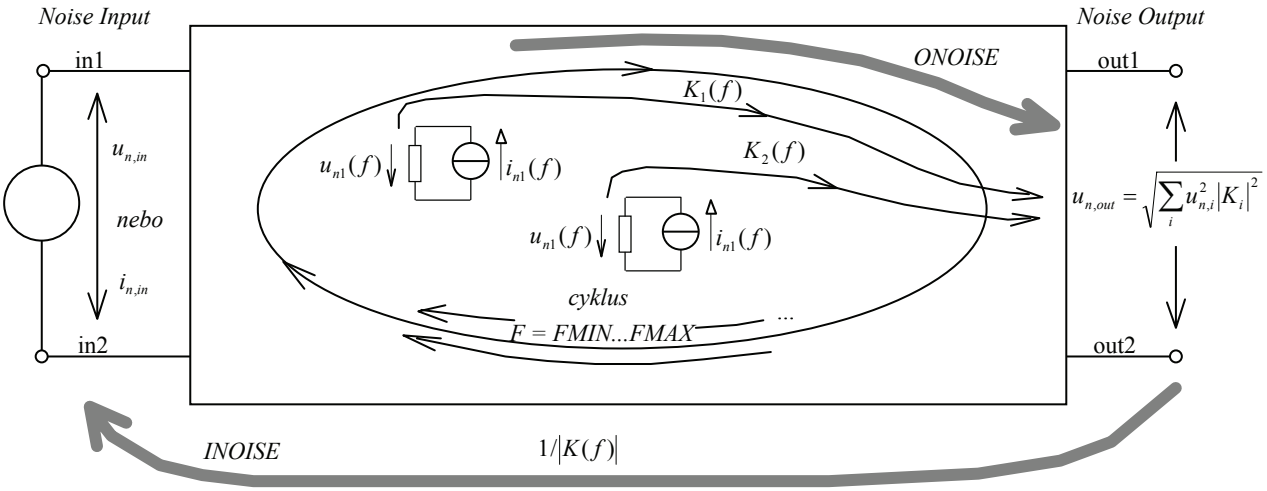
kde K_1, K_2, … jsou kmitočtové charakteristiky přenosů z dílčích šumových zdrojů na výstup. Vzorec platí za předpokladu, že signály z jednotlivých zdrojů šumu jsou nekorelované.

Výsledná spektrální hustota napětí se proto musí počítat ze zobecněné Pythagorovy věty:

$$u_{n,out}(f) = \sqrt{u_{n1}^2 \left|K_1(f)\right|^2 + u_{n2}^2 \left|K_2(f)\right|^2 + \ldots}\,. \tag{9.2}$$

Postup simulačního programu při šumové analýze

Mechanismus šumové analýzy je znázorněn na **obr. 9.42**.



Obr. 9.42: Mechanismus počítačové šumové analýzy a práce s funkcemi *ONOISE* a *INOISE*.

Při šumové analýze se do modelu obvodu začlení paralelně k daným součástkám zdroje jejich šumu ve formě proudových zdrojů, jejichž velikosti jsou v závislosti na typu součástky určovány vzorci pro spektrální hustotu výkonu tepelného, výstřelového a blikavého šumu. Teoreticky by bylo možné použít i ekvivalentní zdroje napětí, pak by však v obvodu přibyly uzly a došlo by k rozšíření soustavy rovnic.

Velikosti spektrálních hustot obecně závisí na kmitočtu díky modelování šumu 1/*F*. Počítají se tedy v analyzační smyčce pro každý kmitočtový krok znova.

Program počítá každý bod spektrální hustoty šumového napětí na výstupu pro daný kmitočet pomocí vzorce (9.2). K tomu potřebuje hodnoty jednotlivých kmitočtových charakteristik K_1, K_2, … . To je důvod, proč je šumová analýza začleněna do analýzy „*AC*“. Výpočet spektrální hustoty výstupního šumového napětí se aktivuje funkcí *ONOISE*.

Na požadavek uživatele programu lze přepočítat výstupní šum na vstupní svorky podle vzorce

$$u_{n,in}(f) = \frac{u_{n,out}(f)}{\left|K(f)\right|}, \tag{9.3}$$

kde $K(f)$ je kmitočtová charakteristika obvodu ze vstupu na výstup. Tento přepočet se aktivizuje funkcí *INOISE*.

Interpretace výsledku je následující: jedná se o spektrální hustotu napětí ekvivalentního zdroje šumu, který by po připojení na vstup vygeneroval stejné poměry na výstupu obvodu,

jehož vnitřní šum je nulový. Tento přepočet šumových poměrů na vstupní svorky umožňuje stanovit požadavky na spektrální rozložení výkonu užitečného signálu na vstupu obvodu.

Jak uvidíme v další části, uživatel definuje vstupní bránu zadáním jména zdroje napětí nebo proudu, který je k této bráně připojen. Půjde-li o zdroj proudu, pak program přepočte na vstupní bránu ekvivalentní spektrální hustotu proudu.

Z vzorců (9.2) a (9.3) vyplývá, že v režimu šumové analýzy jsou všechny proměnné reálná čísla, takže například nemá smysl analyzovat fázové poměry u spektrálních hustot. Taky je zřejmé, že při sčítání jednotlivých šumových příspěvků podle vzorce (9.2) nehrají roli znaménka, takže při modelování zdrojů šumu nezáleží na orientaci napětí a proudů.

Pomocí spektrální hustoty napětí můžeme určit výkon šumu nebo jeho efektivní hodnotu integrací v požadovaném kmitočtovém pásmu $f \in \langle F1, F2 \rangle$. V tomto pásmu zadáme „*Frequency Range*“ a na zvláštní řádek v okně „*AC Analysis Limits*“ definujeme funkci

*SD(onoise*onoise)*

pro výkon šumu a

*SQRT(SD(onoise*onoise))*

pro efektivní hodnotu. Požadovaná hodnota (výkon nebo efektivní hodnota) bude souřadnicí křivky na kmitočtu $F2$. Funkce „*SD*“ realizuje integraci argumentu v mezích „*Frequency Range*“, „*SQRT*“ je druhá odmocnina.

Simulátor zjišťuje poměry při globální teplotě, která je zadaná v položce „*Temperature*“ v okně „*AC Analysis Limits*“, pokud ovšem některá ze součástek nemá zadanou svoji vlastní teplotu (viz část 9.7.2).

Závěrem je třeba upozornit, že modelování šumu tak, aby výsledky odpovídaly realitě, je nesnadné. Modely v simulačních programech nejsou v tomto ohledu příliš dokonalé. Tomu odpovídají i výsledky simulací, které bývají mnohdy optimističtější než realita.

Nastavení okna "*AC Analysis Limits*" při šumové analýze

V okně „*AC Analysis Limits*“ (viz **obr. 9.3** b) je třeba vyplnit položky „*Noise Input*“ a „*Noise Output*“.

Noise Input (šumový vstup): Zde napíšeme jméno zdroje napětí nebo proudu ze schématu, na jehož svorky se má přepočítávat ekvivalentní vstupní šum.

Noise Output (šumový výstup): Zde napíšeme buď jméno uzlu (pak bude výstupem šumové napětí mezi tímto uzlem a zemí), nebo jména dvou uzlů, oddělená čárkou (pak bude výstupem napětí mezi těmito uzly).

Šumová analýza se aktivuje zapsáním funkce *ONOISE* nebo *INOISE* nebo obou do sloupce „*Y Expression*“ (případně netypicky do „*X Expression*“). Současně nesmíme požadovat výpočet veličin typu napětí nebo proud. Každý takovýto pokus by vedl na chybové hlášení

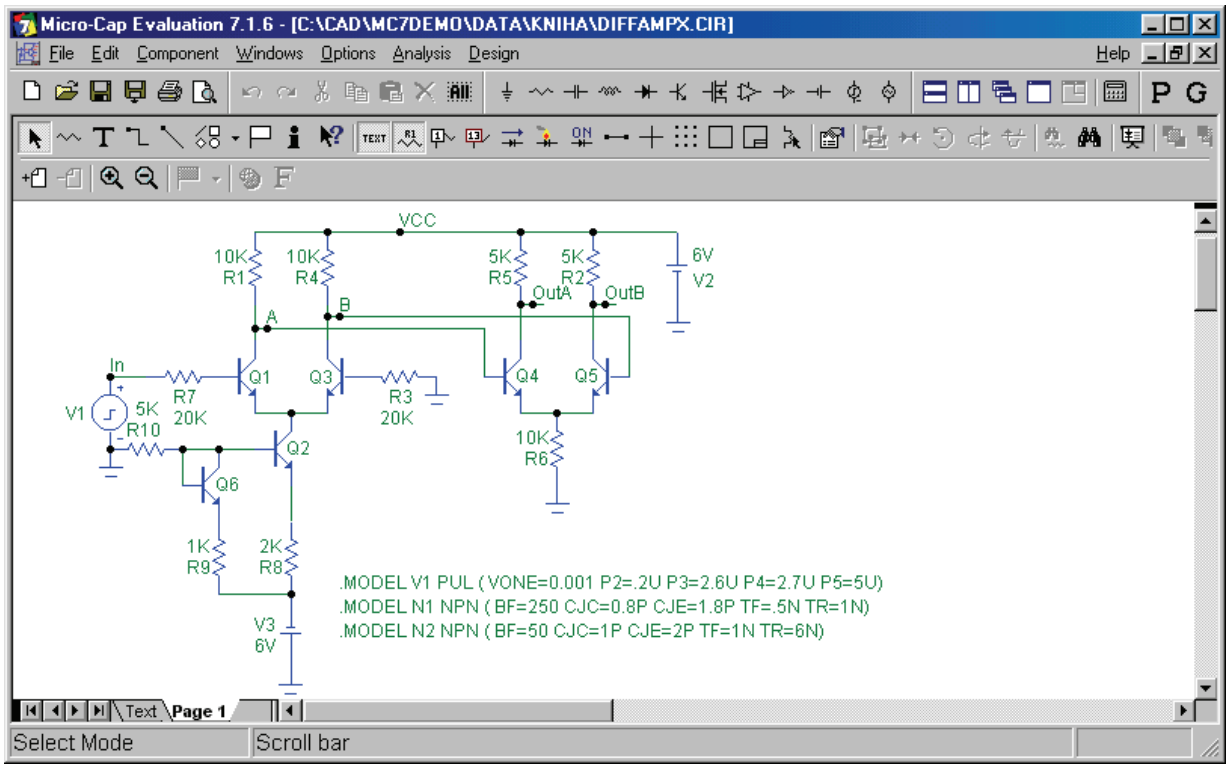
„Can’t plot noise with other expressions”

a analýza by neproběhla.

Příklad šumové analýzy

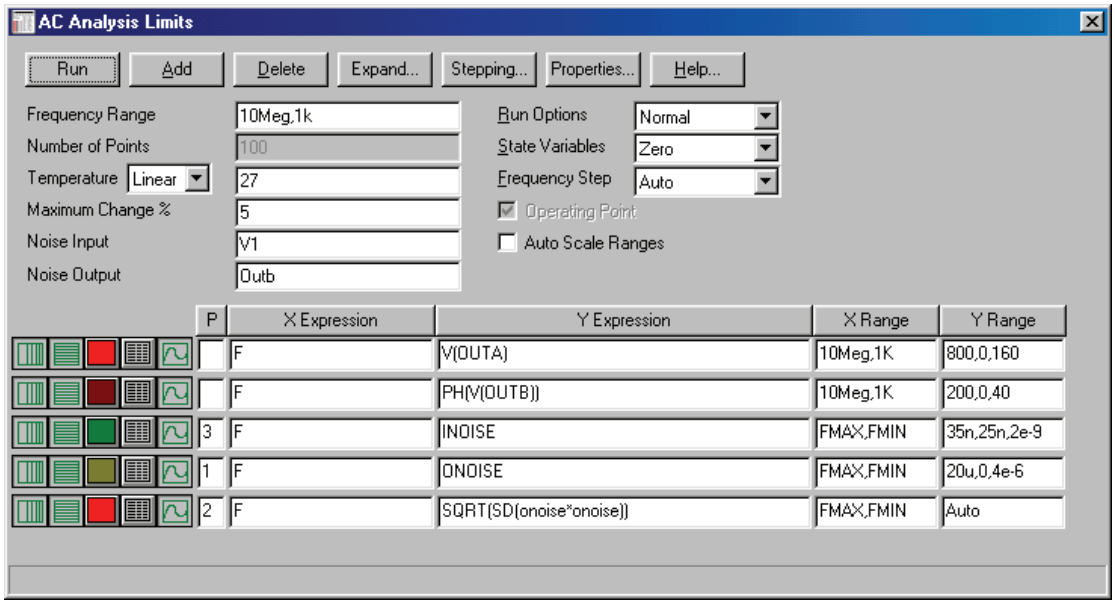
Do editoru načteme soubor **DIFFAMP.CIR** a uložíme jej do záložního souboru **DIFFAMPx.CIR**.

V souboru je model diferenčního zesilovače podle **obr. 9.43**. Určíme spektrální hustotu šumového napětí na výstupu *OutB*, efektivní hodnotu šumového napětí na výstupu, a ekvivalentní vstupní šum na svorkách zdroje *V*1.



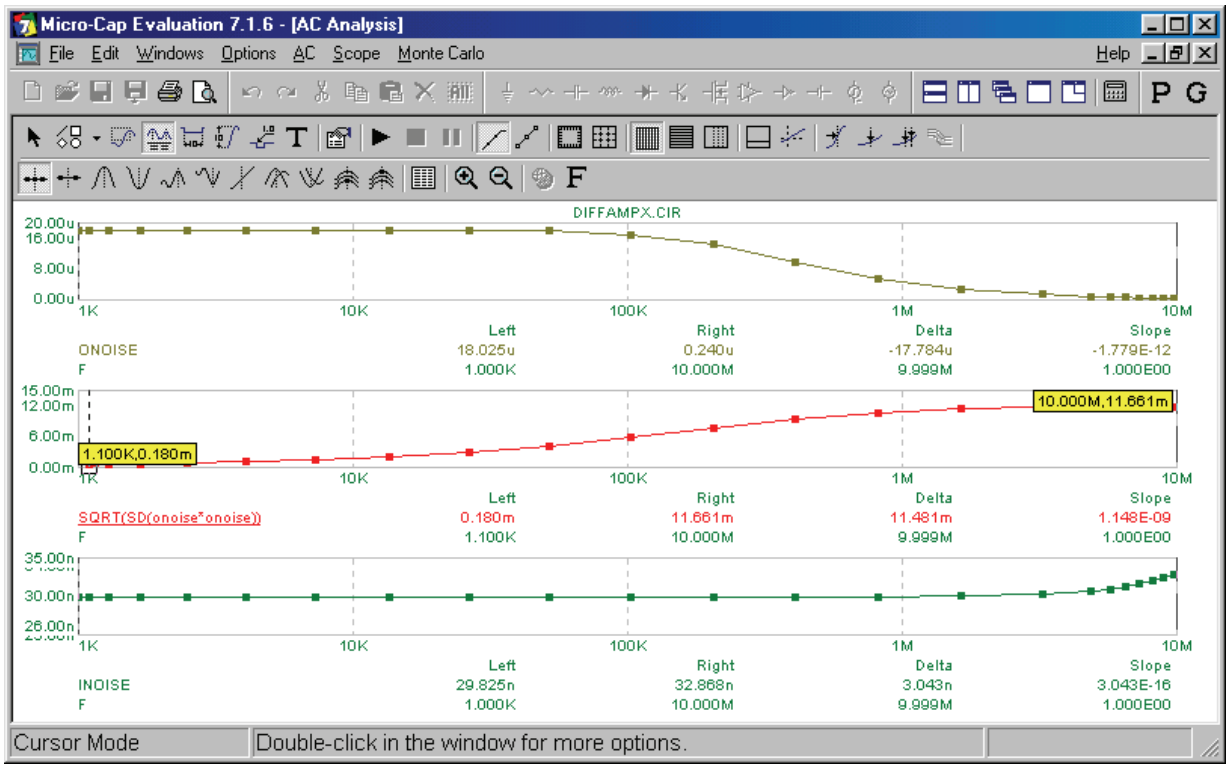
Obr. 9.43: Model diferenčního zesilovače.

Po aktivaci analýzy „*AC*“ se objeví okno „*AC Analysis Limits*“, které upravíme podle **obr. 9.44**.



Obr. 9.44: Úprava podmínek „*AC*“ analýzy pro účely šumové analýzy.

V obrázku č.1 bude křivka spektrální hustoty výstupního šumového napětí. Pod ním, v obrázku č.2, bude křivka druhé odmocniny z integrálu kvadrátu křivky výstupního šumu, která pro konečný kmitočet 10 MHz ukáže efektivní hodnotu výstupního šumového napětí. Ve spodním obrázku č. 3 bude křivka ekvivalentního vstupního šumu.



Obr. 9.45: Výsledky šumové analýzy.

Z obrázku lze vyčíst, že v širokém kmitočtovém pásmu od 1 *k*Hz do 10 MHz má výstupní šumové napětí efektivní hodnotu asi 11 *m*V, tudíž mezivrcholovou hodnotu asi 55 *m*V. Přitom maximální mezivrcholová hodnota užitečného výstupního napětí je limitována napájecím napětím 6 V.

Spektrální hustota ekvivalentního šumového napětí na vstupu vychází asi 29,8 $n\mathrm{V}/\sqrt{Hz}$. Je konstantní zhruba do kmitočtu 1 MHz.

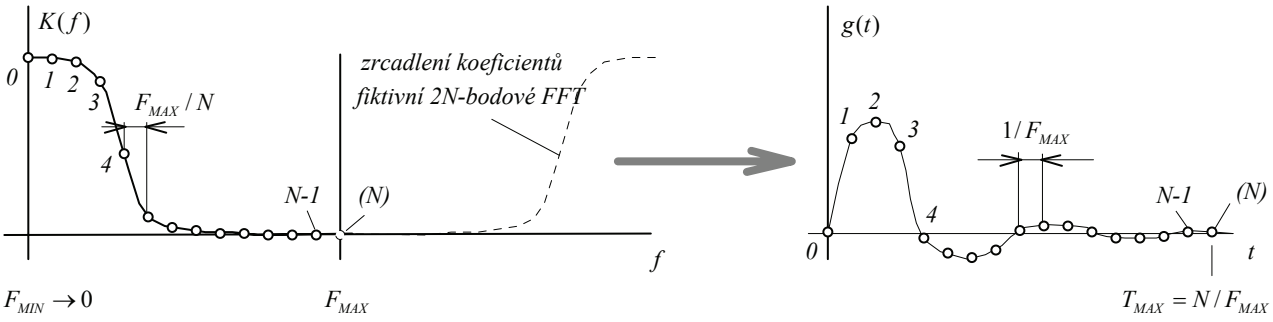
Analyzovaný zesilovač má stejnosměrné zesílení asi 55,6 dB a kmitočet třídecibelového poklesu asi 254 *k*Hz (ověřte si analýzou kmitočtové charakteristiky!). Efektivní hodnota ekvivalentního šumu na vstupu pro kmitočtové pásmo do 254 *k*Hz vychází asi $29{,}8\ n\mathrm{V} \cdot \sqrt{254000} \doteq 15\ \mu\mathrm{V}$. Užitečný signál z tohoto kmitočtového pásma by měl mít efektivní hodnotu adekvátně větší, například pro požadovaný odstup signál/šum 40 dB vychází 1,5 *m*V.

9.4.9 Inverzní Fourierova transformace

V této části je popsán způsob, jak je možné z vzorků kmitočtové charakteristiky, vypočítaných při analýze „*AC*“, určit vzorky impulsní charakteristiky obvodu. Tento postup analýzy nepatří k běžně používaným, uvádíme jej však pro úplnost. Čtenář, kterého příliš nezajímají techniky číslicového zpracování signálů, může tuto část s klidným svědomím přeskočit.

Jak určit impulsní charakteristiku v analýze „*AC*“

Algoritmus inverzní rychlé Fourierovy transformace (*IFFT* – „*Inverse Fast Fourier Transform*“) převádí *N* vzorků komplexních spektrálních koeficientů na *N* vzorků časového průběhu signálu. Zde *N* je podobně jako u transformace *FFT* celočíselná mocnina dvojky. Tento algoritmus má své místo v analýze „*AC*“, která mu poskytuje jako vstupní data vzorky komplexní kmitočtové charakteristiky. Využívá se zde poznatku z teorie, že kmitočtová charakteristika a impulsní charakteristika (odezva obvodu na jednotkový – Diracův impuls) tvoří transformační pár Fourierovy transformace. Jestliže impulsní charakteristika splňuje podmínku vzorkovací poučky (viz část 9.3.9), tvoří vzorkovaná impulsní charakteristika a vzorkovaná kmitočtová charakteristika také transformační pár rychlé Fourierovy transformace. Vzorkováním charakteristik se myslí způsob jejich uchovávání v paměti počítače ve formě vypočtených bodů, rovnoměrně od sebe vzdálených na ose času, resp. kmitočtu (viz **obr. 9.46**).



Obr. 9.46: Princip výpočtu impulsní charakteristiky obvodu z jeho kmitočtové charakteristiky pomocí inverzní rychlé Fourierovy transformace.

Problém je ale v tom, že do algoritmu *IFFT* by mělo vstupovat *N* komplexních koeficientů X_i, které vykazují tzv. komplexně sdruženou symetrii podle vzorce

$$X_i = X^*_{N-i}\,.$$

Znak * znamená komplexně sdružené číslo. To znamená, že koeficienty č. *N*/2 až *N*-1 jsou komplexně sdružené ke koeficieficientům č. 0 až *N*/2-1.

Program proto chápe *N* vzorků kmitočtové charakteristiky jako první polovinu komplexních koeficientů fiktivní 2*N*-bodové *FFT*. Interně používá *N*-bodovou *FFT*, ovšem výsledky musí vhodně interpretovat. Teoretické odvození vede k tomuto praktickému postupu:

1. Provedeme výpočet kmitočtové charakteristiky s konstantním kmitočtovým krokem v *N* bodech, od kmitočtu „co nejnižšího“ (MicroCap neumožňuje začít od kmitočtu 0 Hz) do kmitočtu *Fmax*. Získáme tak *N* komplexních vzorků X_i, vzdálených jeden od druhého na kmitočtové ose o kmitočtový krok *Fmax*/*N*.
2. Impulsní charakteristika, která odpovídá kmitočtové charakteristice, je dána vzorcem

$$g(kT_V) = 2.F_{\max}\,\mathrm{Re}\{IFT(K_i)\}\,, \tag{9.4}$$

kde *IFT* je vnitřní funkce MicroCapu, provádějící inverzní rychlou Fourierovu transformaci, *Re* je funkce vracející reálnou část komplexního argumentu, a

$$T_V = \frac{1}{F_{\max}}$$

je vzdálenost sousedních vzorků impulsní charakteristiky na ose času. Vypočtená impulsní charakteristika bude mít délku trvání $T_{MAX} = N.T_V = N/F_{max}$, což je převrácená hodnota vzdálenosti sousedních vzorků kmitočtové charakteristiky na kmitočtové ose.

K dosažení potřebné přesnosti výpočtu je obecně nutné zadat dostatečně velký počet bodů *N*, tak aby byla kmitočtová charakteristika dostatečně jemně navzorkována. Je třeba zvolit poměrně vysoký kmitočtový rozsah výpočtu charakteristiky, tak aby na kmitočtu *FMAX* byl přenos obvodu zanedbatelný. Pokud analyzovaný obvod tomuto požadavku nemůže z principu vyhovět, například jedná-li se o filtr typu horní propust, pak celá metoda selhává a nemá smysl ji použít.

Jak nastavit menu „AC Analysis Limits“

„Frequency range“: FMAX,FMIN

Maximální kmitočet volíme takový, aby přenos obvodu byl již dostatečně zanedbatelný pro všechny vyšší kmitočty. Oproti maximálnímu přenosu bychom měli volit útlum alespoň 40 dB.

Minimální kmitočet volíme tak, aby byl zanedbatelný oproti *FMAX* a aby přenos na tomto kmitočtu byl co nejvíce roven stejnosměrnému přenosu.

„Number of Points“:

Před vyplněním této položky nastavíme „*Frequency Step*“ na „*Linear*“.

Číslo (označíme symbolem *NP*) v této položce má dva významy:

1. Program zvolí za počet bodů *FFT*, tj. *N*, nejbližší vyšší celočíselnou mocninu dvojky. Kmitočtová charakteristika bude vypočtena v rozsahu kmitočtů od *FMIN* do *FMAX* v $N \geq NP$ bodech.
2. Číslo *NP* bude rovno počtu bodů zobrazené impulsní charakteristiky.

V zájmu přesnosti výpočtu je třeba číslo *NP* zvolit dostatečně vysoké. Můžeme začít s hodnotou 100 a případně ji v další analýze zvyšovat. Jsou dvě kritéria ověření přesnosti: **1.** Dostatečné vykreslení detailů kmitočtové charakteristiky (hrubé kritérium). **2.** Impulsní charakteristika musí zanikat, na jejím „konci“ tedy musíme dostat prakticky nulové hodnoty.

„State Variables“: Zero

„Frequency Step“: Linear

Zadání požadavků na analyzovanou křivku:

„X Expression“: *t*

„Y Expression“: 2*Fmax*re(*ift*(*V*(*Out*)))

Namísto symbolu *Fmax* je možné vyplnit konkrétní číselnou hodnotu. Zápis *V*(*Out*) podle potřeby nahradíme konkrétní definicí kmitočtové charakteristiky.

Příklad analýzy

Otevřeme soubor **FFT4.CIR** a uložíme jej pod jménem **FFT4x.CIR**.

Je zde modelován jednoduchý *RLC* filtr. V souboru jsou přednastaveny dvě metody analýzy impulsní charakteristiky filtru, tj. jeho odezvy na Diracův impuls z nulových počátečních podmínek. Diracův impuls je limitním případem krátkého obdélníkového impulsu o délce trvání ΔT a výšce $1/\Delta T$ pro $\Delta T \rightarrow 0$.

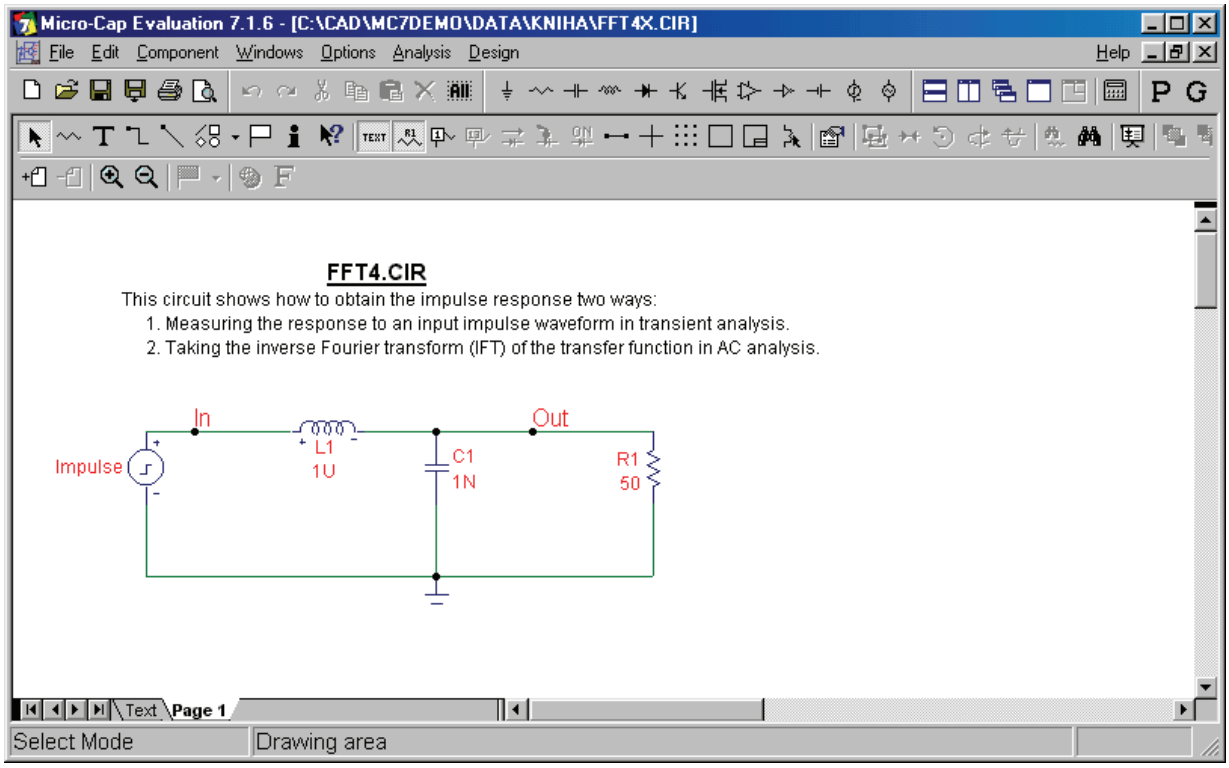
První metoda se odehrává v analýze „*Transient*“, kdy počítáme odezvu obvodu na „krátký a vysoký“ impuls. Tento impuls modelujeme pomocí součástky „*Pulse Source*“.

Druhá metoda vychází z výpočtu kmitočtové charakteristiky filtru v analýze „*AC*“ a následném použití algoritmu zpětné Fourierovy transformace.

V složce „*Text*“ nalezneme následující definici zdroje impulsů:

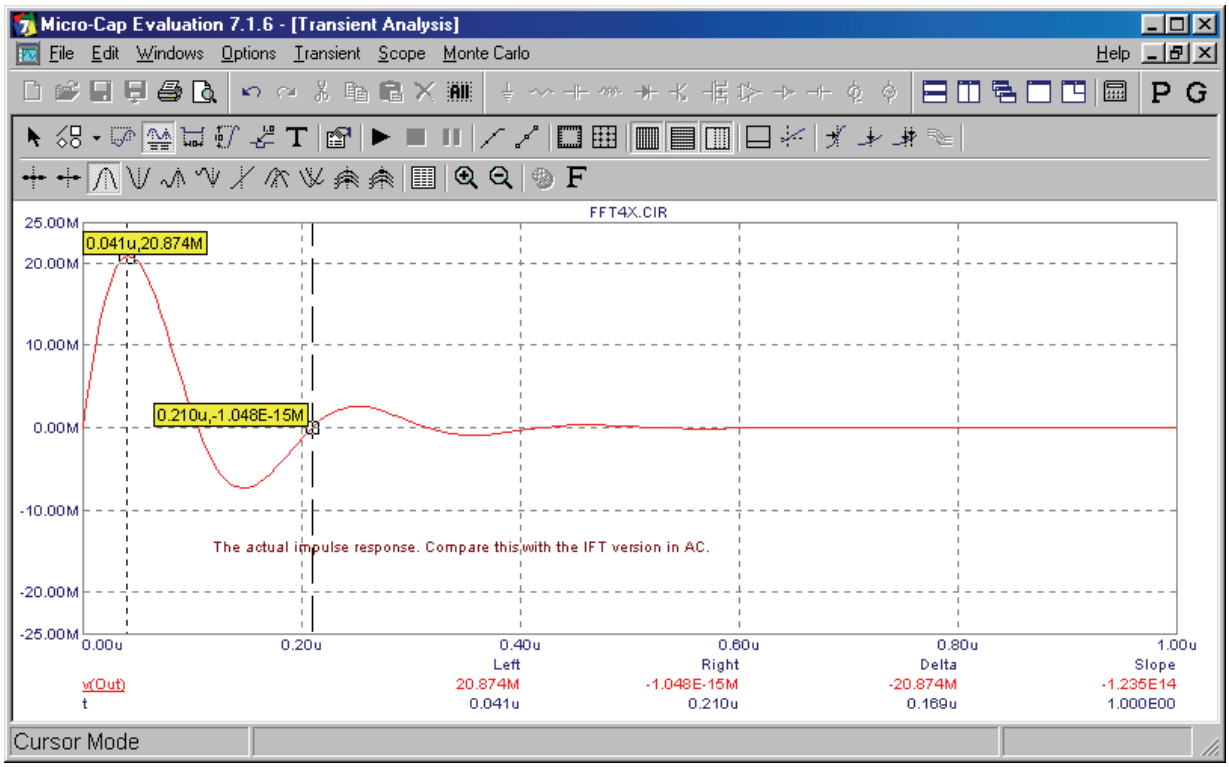
.model impulse pul (vzero=0.0 *vone*=1*e*9 *p*1=0 *p*2=1*p p*3=1*n p*4=1.001*n p*5=1*U)*

V podstatě se jedná o impulsy o výšce 10^9 voltů a šířce 1 *n*s, tedy 10^{-9} s. Jsou-li časové konstanty analyzovaného obvodu řádově větší než 1 *n*s, pak obvod „nepozná“, zdali je na vstupu tento signál nebo ideální Diracův impuls (podrobnosti viz příloha P1.5). Opakovací perioda impulsů je 1 μs. Je tedy předpoklad, že impulsní charakteristika bude kratší než 1 μs.



Obr. 9.47: Obvod k analýze impulsní charakteristiky.

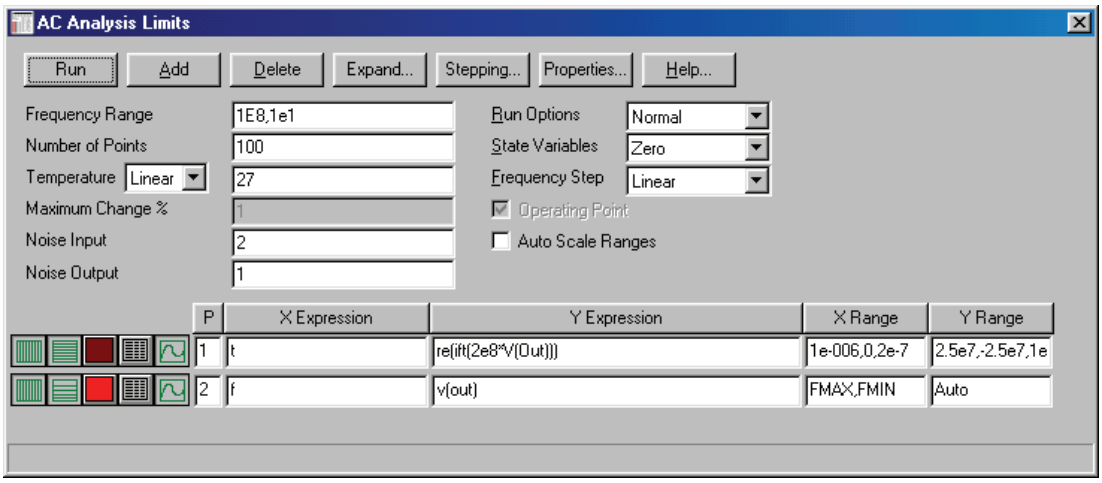
Spustíme nejprve analýzu „*Transient*“ s přednastavenými parametry okna „*Transient Analysis Limits*“. Výsledek vidíme na **obr. 9.48**.



Obr. 9.48: Impulsní charakteristika obvodu z **obr. 9.47** získaná analýzou jeho odezvy na krátký impuls.

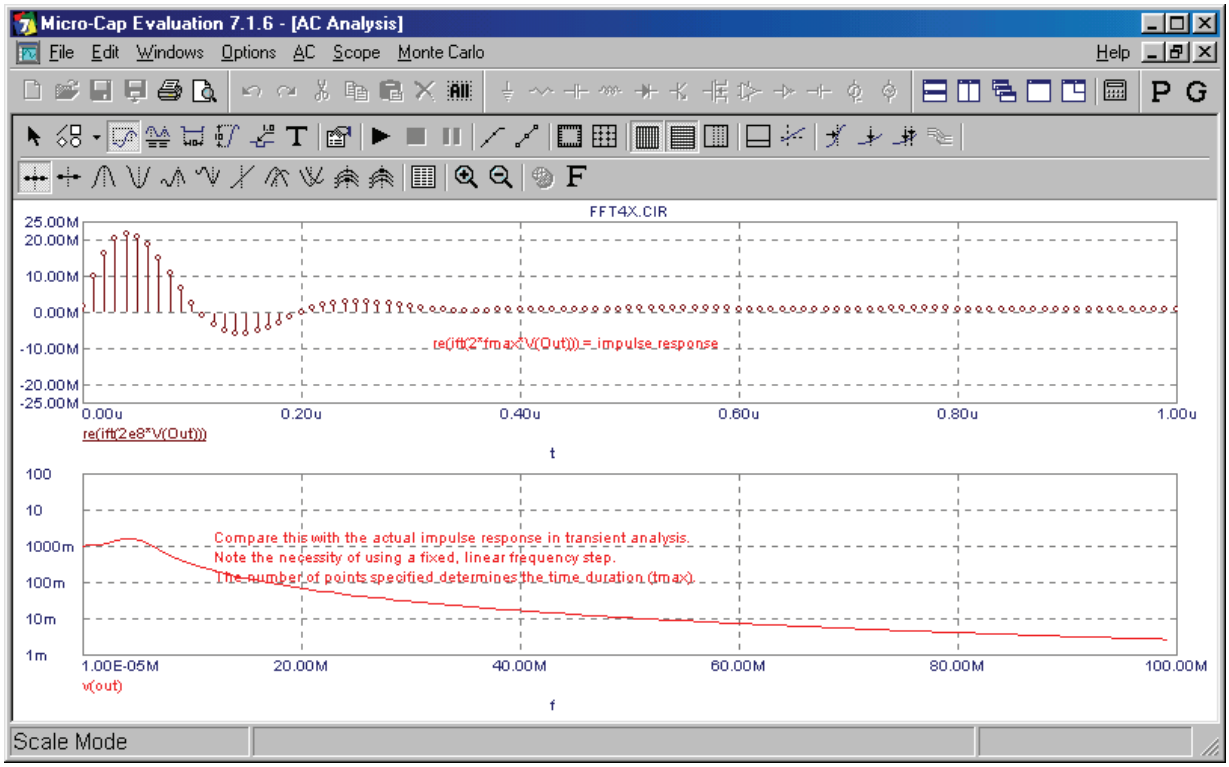
V obrázku jsou vyznačeny souřadnice dvou význačných bodů impulsní charakteristiky kvůli srovnání s výsledky, které získáme metodou inverzní Fourierovy transformace. Asi po 0,6 µs se charakteristika výrazně ustaluje na nulovou úroveň.

Analýzu ukončíme stlačením *F*3. Přejdeme do analýzy „*AC*“. Okno „*AC Analysis Limits*“ doplníme o vykreslování kmitočtové charakteristiky, tak jak je ukázáno na **obr. 9.49**. Všimněte si ikon  : osa *X* je u obou křivek nastavena jako lineární. Obsah položky „*Number of Points*“ napovídá, že k výpočtu impulsní charakteristiky bude použita 128-bodová *IFFT* a že bude zobrazeno prvních 100 bodů této charakteristiky.



Obr. 9.49: Nastavení podmínek „*AC*“ analýzy k výpočtu impulsní charakteristiky obvodu metodou *IFFT*.

Spustíme analýzu. Po jejím proběhnutí vyblokujme pro přehlednost vykreslování pomocných mřížek na logaritmické stupnici kliknutím na ikonu („*Minor Log Grids*“). Výsledek by měl odpovídat **obr. 9.50**.



Obr. 9.50: Impulsní charakteristika obvodu z **obr. 9.47** získaná z kmitočtové charakteristiky algoritmem zpětné *FFT*.

Kliknutím na ikonu („*Data Points*“) zviditelněte vypočtené body kmitočtové charakteristiky. Přesvědčte se, že jsou rozloženy na kmitočtové ose rovnoměrně. Pro další práci je však vhodné zobrazení těchto bodů opět potlačit.

Po srovnání vzorků impulsní charakteristiky s výsledkem na **obr. 9.48** zjistíme některé nepřesnosti, které jsou zřejmé zejména v příliš velkých hodnotách vzorků na konci charakteristiky. Vraťte se do okna „*AC Analysis Limits*“ a zvětšete obsah položky „*Number of Points*“ na 1000. Po opětovném proběhnutí analýzy již zjistíte lepší shodu s přesným průběhem.

9.5 Analýza „*DC*“ neboli stejnosměrná analýza

9.5.1 Cíle analýzy

Základním cílem stejnosměrné analýzy je napodobování funkce „inteligentního charakterografu“, tj. přístroje pro snímání stejnosměrných charakteristik nelineárních obvodů. K těmto charakteristikám patří například ampérvoltové charakteristiky dvojpólů a vůbec všechny charakteristiky, měřené „bod po bodu“ - ať už ručně nebo automatizovaně – postupným nastavováním napětí a proudů ze stejnosměrných zdrojů. Přívlastkem „inteligentní“ zde opět pojmenováváme řadu možností, které jsou softwarově zpřístupněny a které by klasickými charakterografy byly těžko realizovatelné.

Dalším cílem stejnosměrné analýzy může být sledování, jak nejrůznější parametry součástek obvodu (například proudový zesilovací činitel tranzistoru) mohou ovlivňovat stejnosměrné poměry. Tato funkce simulátoru nabízí nevídané možnosti v zkoumání stejnosměrných vlastností obvodů.

9.5.2 „Inteligentní charakterograf“

Ke klasickému charakterografu připojujeme pomocí speciálních přípravků definovaným způsobem buď dvojpóly (diody, nelineární rezistory) nebo vybrané typy vícepólů (zejména tranzistory). Charakterograf pak může pracovat ve dvou různých režimech:

1. **Základní „dvojpólový“ režim**. Výsledkem je jedna ampérvoltová charakteristika.
2. **Parametrický režim**. Výsledkem je síť charakteristik, např. výstupní charakteristiky tranzistoru.

Simulační program umožňuje při stejnosměrné analýze práci v obou těchto režimech, ovšem v mnohem obecnější rovině. Především neexistuje omezení na typ analyzovaného obvodu. Můžeme snímat ampérvoltovou charakteristiku diody stejně jako třeba napěťovou převodní charakteristiku celého integrovaného zesilovače. Nejsou kladena žádná omezení na typ obvodových veličin, které mohou být sledovány současně a vyhodnocovány tak jejich souvislosti. Samozřejmostí je krokování teploty, tak jako u ostatních analýz.

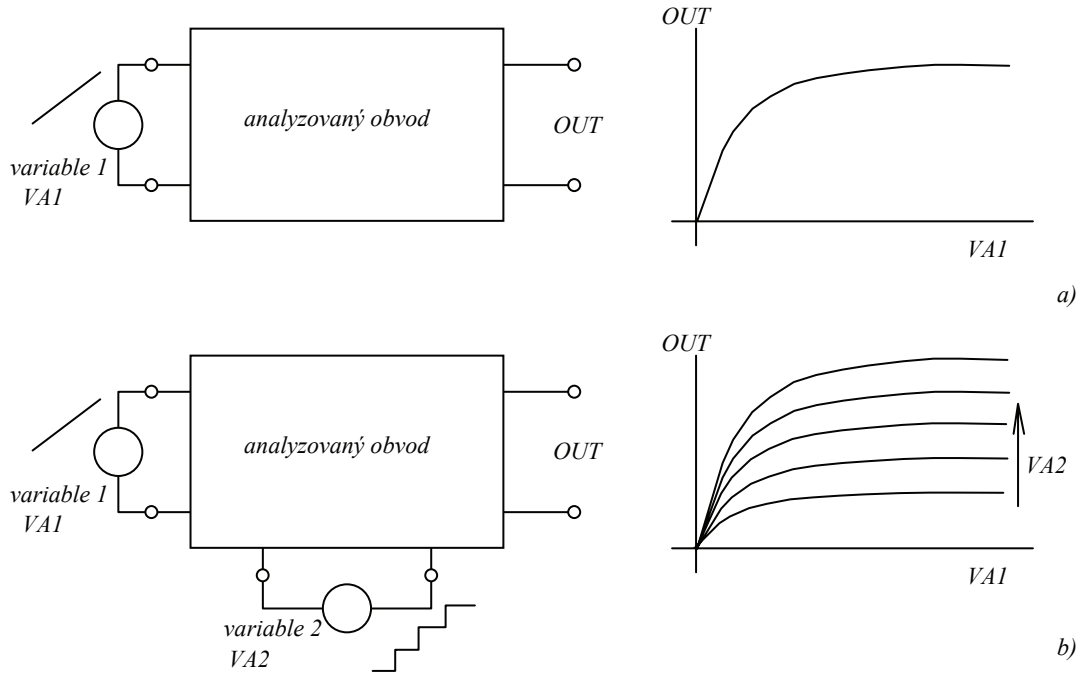
Další pozoruhodnou funkcí simulačního programu je již výše zmíněná možnost zkoumání, co by se stalo se sledovanou stejnosměrnou veličinou, například proudem kolektoru tranzistoru, kdybychom měnili v daných mezích některý parametr, například proudový zesilovací činitel nebo jakýkoliv jiný parametr modelu, symbolickou proměnnou apod. Tak je možné simulovat vlivy čehokoliv a získané souvislosti znázorňovat příslušnými křivkami. Toto již nemá mnoho společného s původní stejnosměrnou analýzou. Dané postupy budeme v dalším textu označovat termínem *zobecněná stejnosměrná analýza*.

Můžeme tedy shrnout, že simulátor realizuje buď **klasickou** nebo **zobecněnou** stejnosměrnou analýzu. V obou případech pracuje buď v základním režimu nebo v parametrickém režimu.

Základní režim je charakteristický postupnou změnou jedné veličiny v obvodu po dostatečně jemných krocích, například stejnosměrného napětí na diodě, vyhodnocováním jiné stejnosměrné veličiny, například proudu diodou, a vykreslováním požadované závislosti, například ampérvoltové charakteristiky. Krokovaná veličina se v MicroCapu nazývá „**Variable 1**“ (*proměnná č.*1).

V **parametrickém režimu** se kromě proměnné č.1, např. napětí kolektor-emitor tranzistoru, krokuje další *proměnná č.* 2 („**Variable 2**“), například proud báze tranzistoru. Krokování však zpravidla bývá hrubší v porovnání s krokováním proměnné č.1. Výsledkem je síť charakteristik, například síť výstupních charakteristik tranzistoru $I_C = f(U_{CE})$, I_B = konst. Parametrem sítě křivek je *proměnná č.* 2. Obojí krokování probíhá programově ve dvou smyčkách, ve vnější smyčce se krokuje *proměnná č.* 2 v tolika krocích, kolik je křivek v síti, ve vnitřní smyčce *proměnná č.* 1. Na „jemnosti“ krokování *proměnné č.* 1 závisí hladkost výsledných křivek.

Na obr. 9.51 je naznačeno, že *proměnná č.* 1 se obvykle vynáší na vodorovnou osu. Není to samozřejmě nutné, způsob vykreslování závislostí je věcí uživatele a nesouvisí se způsobem výpočtu. Analyzujeme-li například ampérvoltovou charakteristiku, můžeme z ní „udělat“ záměnou os voltampérovou charakteristiku.



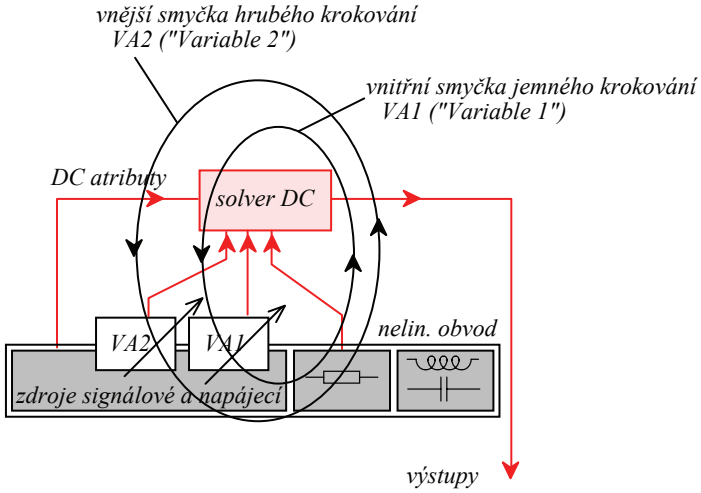
Obr. 9.51: a) Základní, b) parametrický režim *DC* analýzy.

9.5.3 Jak postupuje simulátor při stejnosměrné analýze

Na **obr. 9.52** je zjednodušené znázornění interních mechanismů, které působí v simulátoru při stejnosměrné („*DC*“) analýze.

Dalo by se říci, že simulátor počítá při analýze „*DC*“ opakovaně a mnohonásobně stejnosměrný pracovní bod. Proto je tato analýza označována za nejnáročnější z hlediska fungování numerických algoritmů.

Z nelineárního obvodu vstupuje do analýzy jen jeho rezistivní část. Ze signálových a napájecích zdrojů, pokud jejich signály nejsou vybrány jako *proměnné č.* 1 a 2 („*variable* 1/2“), jsou vyseparovány jejich tzv. „*DC*“ atributy (viz část 9.5.4, resp. příloha P10.1.3), tj. jsou nahrazeny stejnosměrnými zdroji.



Obr. 9.52: Mechanismus „DC“ analýzy v obvodovém simulátoru.

Při **klasické** stejnosměrné analýze jsou za *proměnné č.* 1, příp. 2 vybrány signály z existujících zdrojů v obvodu.

U **zobecněné** analýzy mohou být za *proměnné č.* 1 a 2 vybrány tyto struktury: zdroje napětí, zdroje proudu, teplota, libovolný parametr z modelů součástek v obvodu, libovolná symbolická proměnná.

9.5.4 Atributy součástek při stejnosměrné analýze

Při stejnosměrné analýze jsou kapacitory vyňaty z modelu obvodu a induktory jsou nahrazeny zkraty. Vzorce kmitočtových závislostí v modelech jsou redukovány s uvažováním kmitočtu 0 Hz.

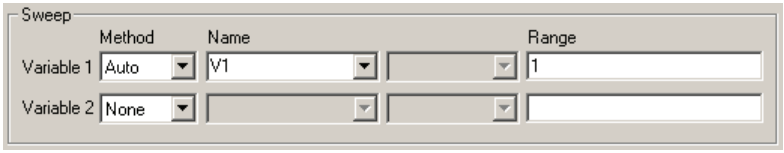
Všechny zdroje v obvodu, pokud nebyly vybrány k reprezentaci *proměnných č.* 1 a 2, vstupují do analýzy takto (uvádíme jen nejpoužívanější typy zdrojů, viz příloha P10.1.3):

- Baterie zůstává zachována.
- Zdroj impulsů („*Pulse Source*“) je nahrazen stejnosměrným zdrojem o napětí, které se rovná napětí zdroje v čase 0, tedy parametru *VZERO* v modelu.
- Zdroj harmonického signálu („*Sine Source*“) je nahrazen stejnosměrným zdrojem o napětí, které se rovná hodnotě signálu v čase 0.
- Zdroj proudu („*I Source*“) zůstává zachován.
- Zdroje *V* a *I* (SPICE formát) jsou nahrazeny stejnosměrnými zdroji o hodnotách, které se rovnají parametru DC v modelu.

Je zajímavé, že poslední z uvedených pravidel neplatí pro dynamickou stejnosměrnou analýzu („*Dynamic DC*“), jak uvidíme v části 9.6.1 (viz též příloha P10.1.3).

9.5.5 Menu "DC Analysis Limits"

Toto menu bylo ukázáno v kapitole 9.2.4 na **obr. 9.3** c) spolu s vysvětlením položek, které jsou společné pro všechny tři základní analýzy. Níže je vyložen význam zbývajících položek z části „*Sweep*“, specifických pro analýzu „*DC*“.



Obr. 9.53: Položky skupiny „*Sweep*“ okna „*DC Analysis Limits*“.

Variable 1:

Method Auto
Linear
Log
List

Auto: Krok *proměnné č.* 1 je řízen automaticky v závislosti na nastaveném parametru „*Maximum Change*“ (viz řízení kroku v analýze „*AC*“, část 9.2.4).
Položka „*Range*“ – rozsah, má v tomto režimu formát: *Max* [,*Min*]. Nezadá-li se „*Min*“, platí implicitní hodnota 0. Za položkou „*Min*“ může být ještě „*krok*“ tak jako u dalších metod „*Linear*“ a „*Log*“, ale v tomto režimu bude ignorován.

Linear: Lineární krok (viz část 9.4.5), položka „*Range*“ má formát: *Max* [,*Min* [,*krok*]]. Položky v závorkách jsou nepovinné, pokud se nezadají, platí implicitní hodnoty: *Min* = 0, *krok* = (*Max*-*Min*)/50. Parametr „*Maximum Change*“ se neuplatní.

Log: Logaritmický krok (viz část 9.4.5), položka „*Range*“ má formát: *Max* [,*Min* [,*násobek*]]. Položky v závorkách jsou nepovinné, pokud se nezadají, platí implicitní hodnoty: *Min*=*Max*/10, násobek=$(Max/Min)^{1/10}$. Parametr „*Maximum Change*“ se neuplatní.

List: Výčet hodnot, položka „*Range*“ má formát: *hodnota*1 [,*hodnota*2 [,*hodnota*3 …]].

Name (jméno): Vybere se jméno *proměnné č.* 1 z nabídky. Pokud vybereme model prvku, v políčku vpravo se objeví nabídka prvků modelu.

Variable 2:

Method None
Linear
Log
List

None: *proměnná č.* 2 není vybrána, stejnosměrná analýza bude probíhat v základním, neparametrickém režimu.

Ostatní položky mají stejný význam jako u „*Variable* 1“.

9.5.6 Příklady stejnosměrné analýzy

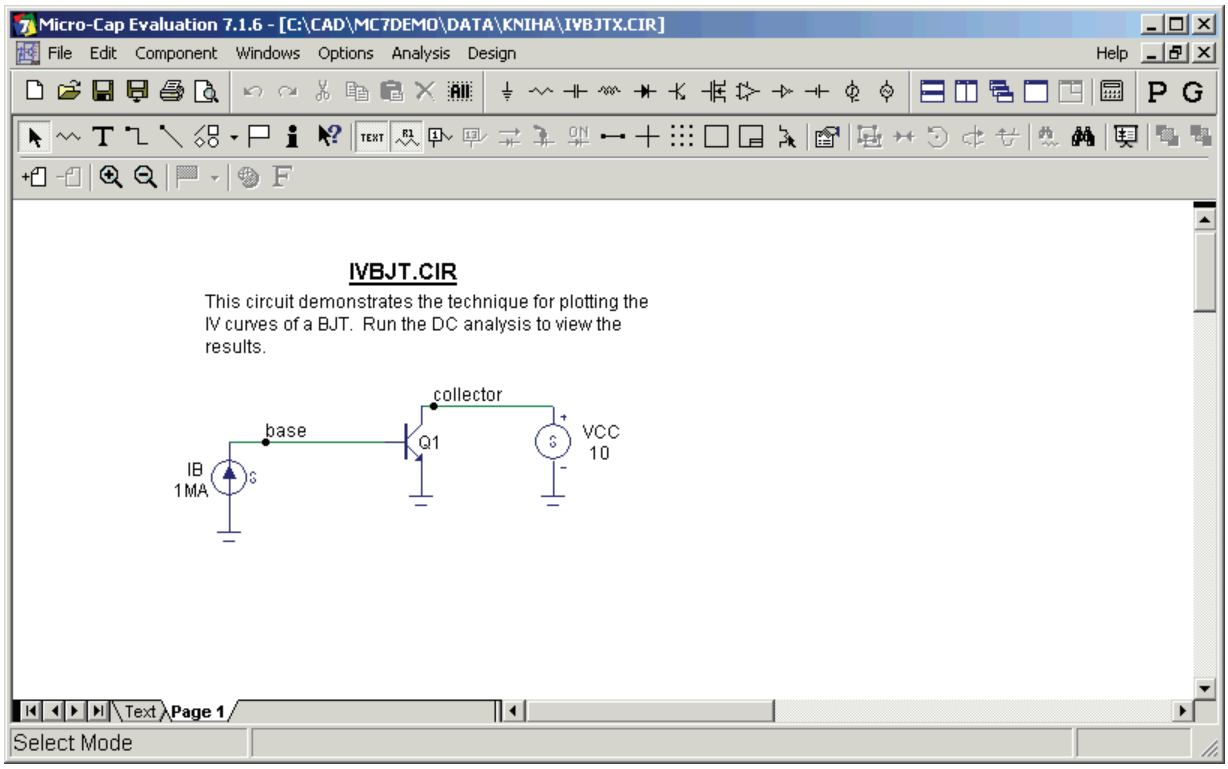
Příklad 1: **TTLINV.CIR** - klasická stejnosměrná analýza v základním režimu

V části 8.2.3 je ukázka práce se souborem **TTLINV.CIR**, konkrétně v režimu stejnosměrné analýzy modelu hradla *NAND* na tranzistorové úrovni. Výsledkem byla nelineární napěťová převodní charakteristika. Projděte si tento příklad znova a pokuste se na

něj dívat již se znalostí významu jednotlivých položek přednastavených v okně „*DC Analysis Limits*“.

Příklad 2: **IVBJT.CIR** – klasická stejnosměrná analýza v parametrickém režimu

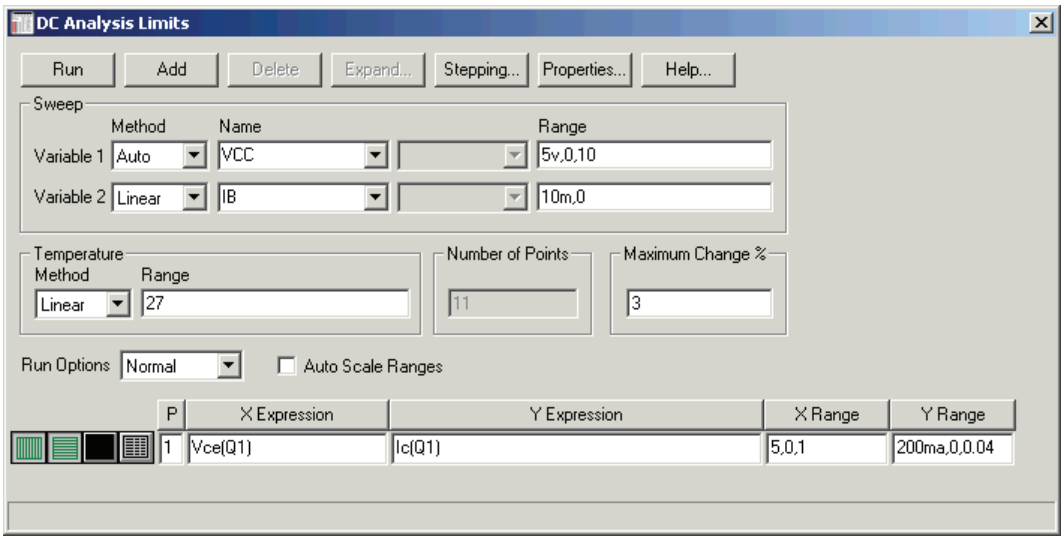
Otevřeme soubor **IVBJT.CIR** a uložíme jeho obsah do záložního souboru **IVBJTx.CIR**. Soubor obsahuje model bipolárního tranzistoru, zapojeného s cílem analýzy jeho statických výstupních charakteristik, tj. charakteristik typu $I_C = f(U_{CE})$, I_B = konst.



Obr. 9.54: Obvod k analýze stejnosměrných výstupních charakteristik bipolárního tranzistoru.

Pro tyto účely jsou do obvodu zařazeny dva zdroje. Napětí kolektor-emitor bude „jemně“ krokováno ze zdroje napětí, který je ve schématu označen jako *VCC*. Jde o „SPICE“ zdroj typu „*V*“. Parametr křivek výstupních charakteristik, proud báze, bude nastavován v „hrubých“ krocích proudovým zdrojem „SPICE“ typu „*I*“, označeným jako *IB*.

Okno „*DC Analysis Limits*“ má standardní přednastavení podle **obr. 9.55**:

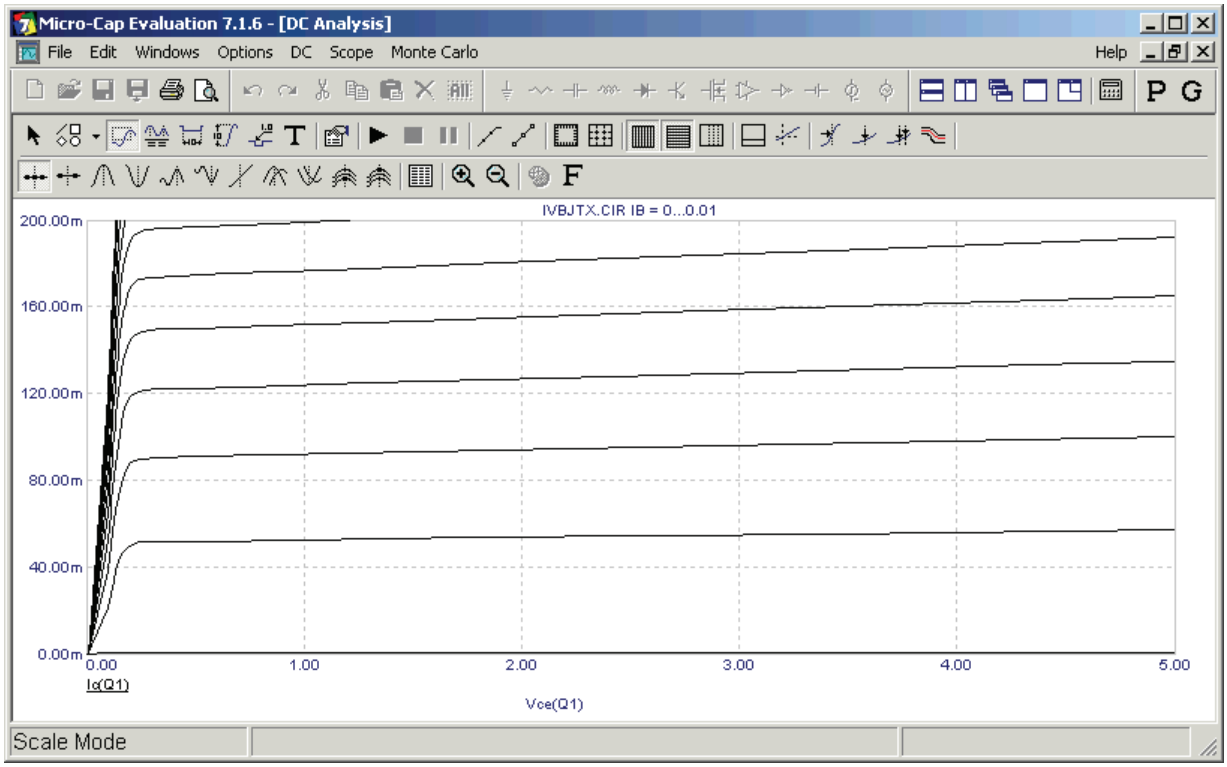


Obr. 9.55: Přednastavení parametrů „*DC*“ anylýzy obvodu z **obr. 9.54**.

Jako *proměnná č.* 1 je nastaveno napětí zdroje *VCC*. Bude nastavováno od 0 V do 5 V metodou “*Auto*”, což znamená, že délka kroku bude řízena automaticky s využitím položky „*Maximum Change*“. Krok „10“ jako poslední údaj v položce „*Range*“ bude ignorován.

Proměnná č. 2 je proud zdroje *IB*, krokovaný lineární metodou od 0 *m*A do 10 *m*A. Velikost kroku není udána, platí tedy implicitní hodnota (*Max-Min*)/50 = 0,2 *m*A. Na vodorovnou osu se bude vynášet napětí kolektor-emitor tranzistoru *Q*1, na svislou osu proud kolektoru *Q*1.

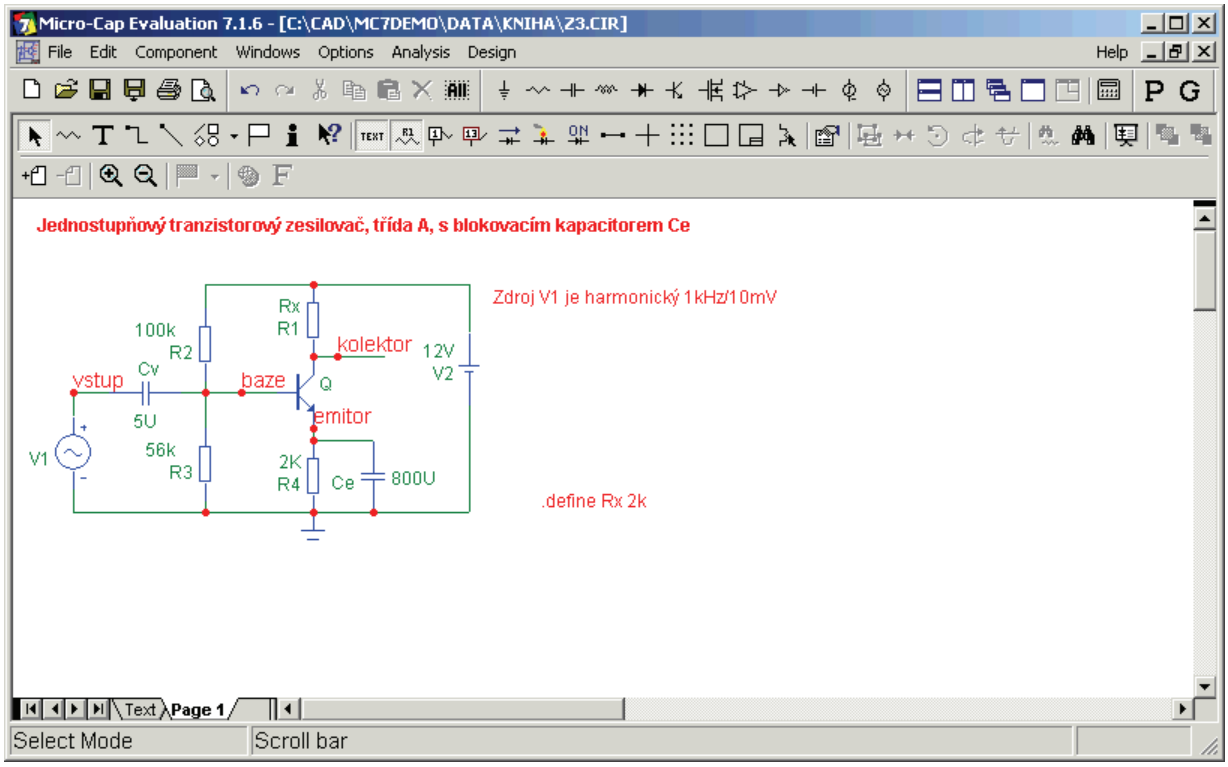
Výsledek analýzy je na **obr. 9.56**.



Obr. 9.56: Síť výstupních charakteristik tranzistoru jako výsledek parametrické „*DC*“ analýzy.

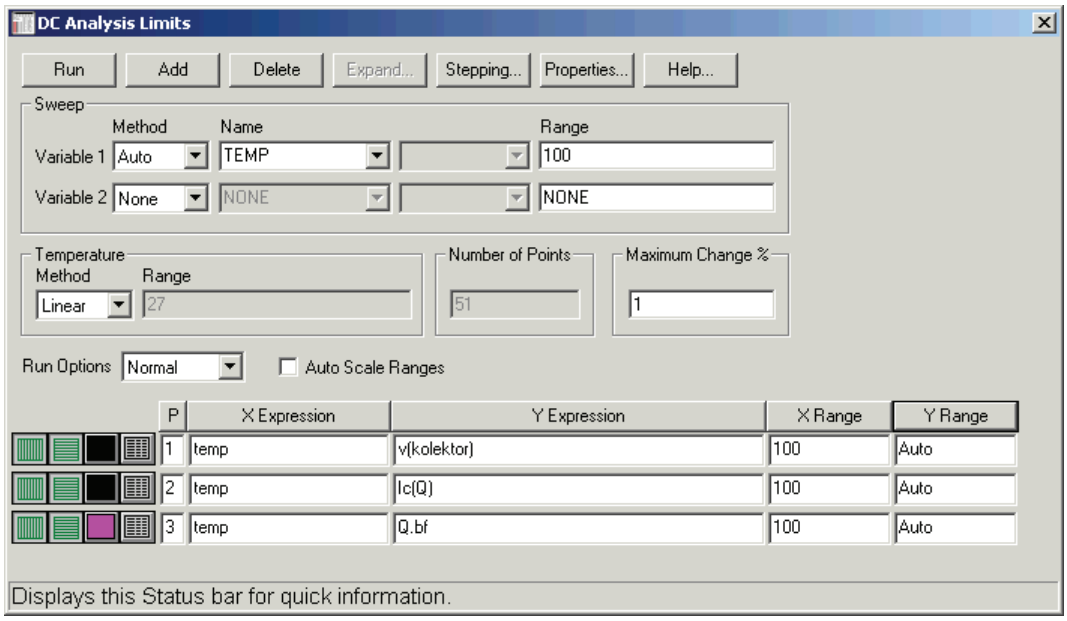
Příklad 3: **Z3.CIR** – zobecněná stejnosměrná analýza

Následující příklad není součástí originální instalace programu MC7. Můžeme jej nalézt v dodatkovém souboru **Z3.CIR**.



Obr. 9.57: Tranzistorový zesilovač pro demonstraci zobecněné „*DC*“ analýzy.

Jedná se o jednostupňový tranzistorový zesilovač se stejnosměrným pracovním bodem stabilizovaným do oblasti třídy *A*. Tranzistor *Q* je typu *BC*107*A*. Všimněme si, že kolektorový odpor je definován pomocí symbolické proměnné *Rx*. Bude nás zajímat „ujíždění“ pracovního bodu s teplotou a další efekty s tím spojené.

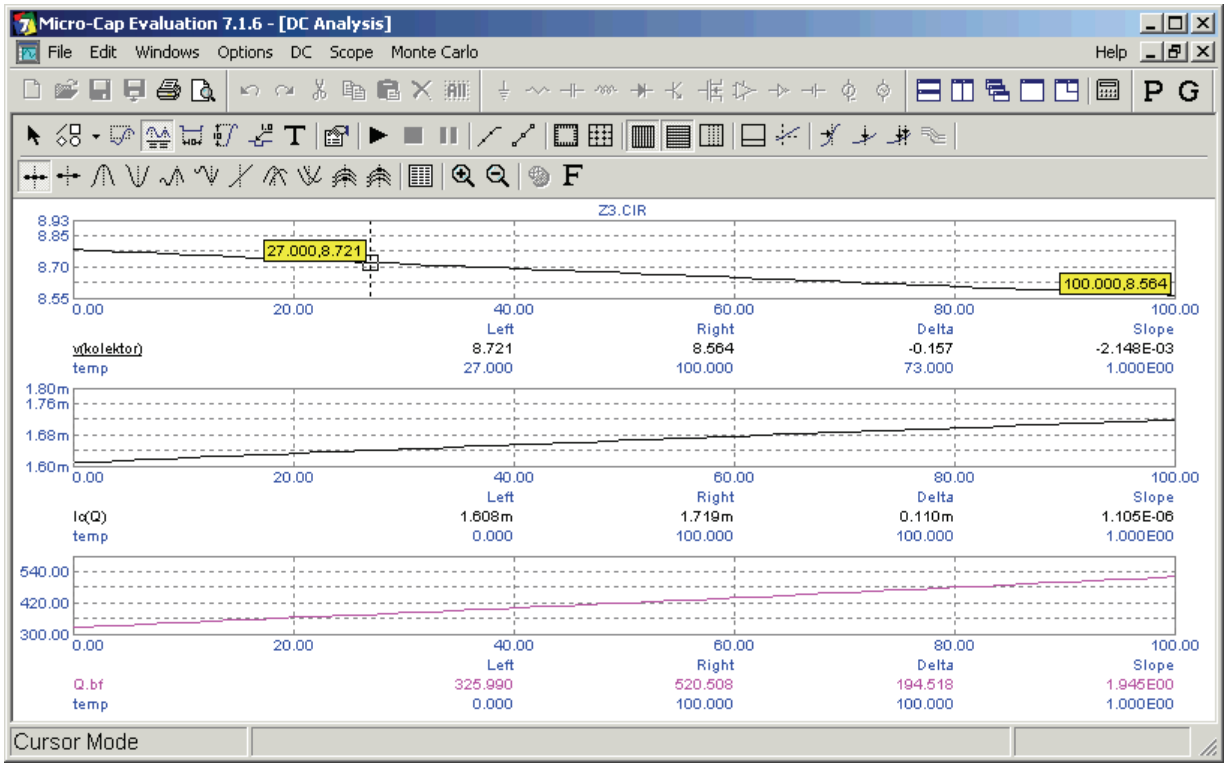


Obr. 9.58: Parametry pro zobecněnou „*DC*“ analýzu v okně „*DC Analysis Limits*“.

Zkontrolujte, že okno „*DC Analysis Limits*“ je nastaveno tak, jak vidíte na **obr. 9.58**.

Proměnná č. 1 je teplota („*TEMP*“), variovaná od nuly do 100 stupňů Celsia. Dole jsou definovány tři křivky, na vodorovné ose je vždy teplota. První křivka udává teplotní závislost kolektorového napětí, druhá proudu kolektoru, třetí proudového zesilovacího činitele *BF* tranzistoru. Všimněme si formy posledního zápisu: název prvku, tečka, název parametru modelu prvku.

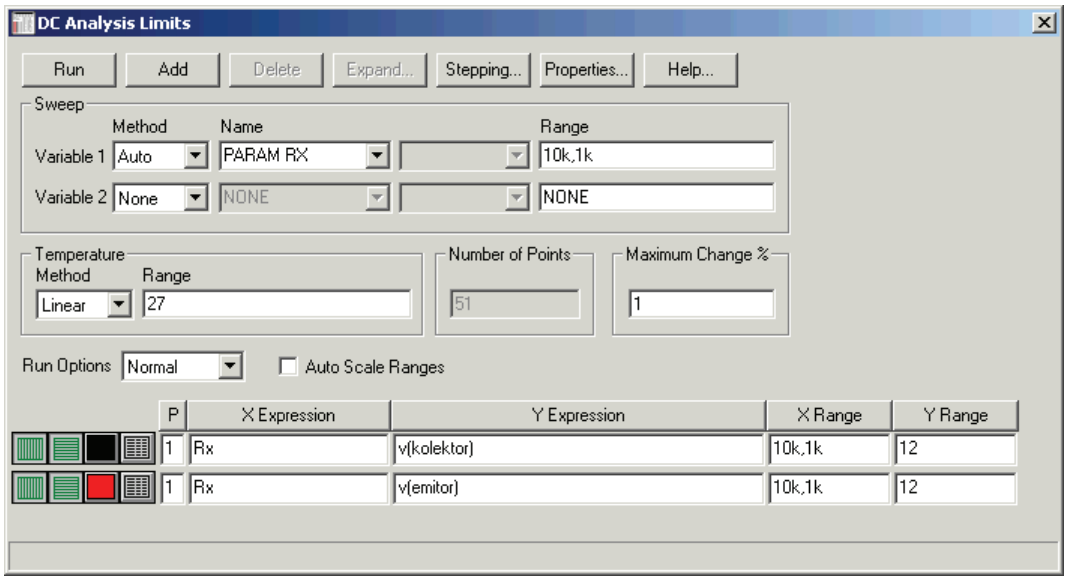
Výsledky analýzy jsou na **obr. 9.59**. S růstem teploty se tranzistor více otevírá, klesá kolektorové napětí, roste klidový proud kolektoru i proudový zesilovací činitel. Při 27°C je kolektorové napětí asi 8,7 V, při 100°C by pokleslo na 8,56 V. Stabilizační součinitel $\Delta UC/\Delta TEMP$ je roven souřadnici „*Slope*“ a vychází asi -2,1 *m*V/°C.



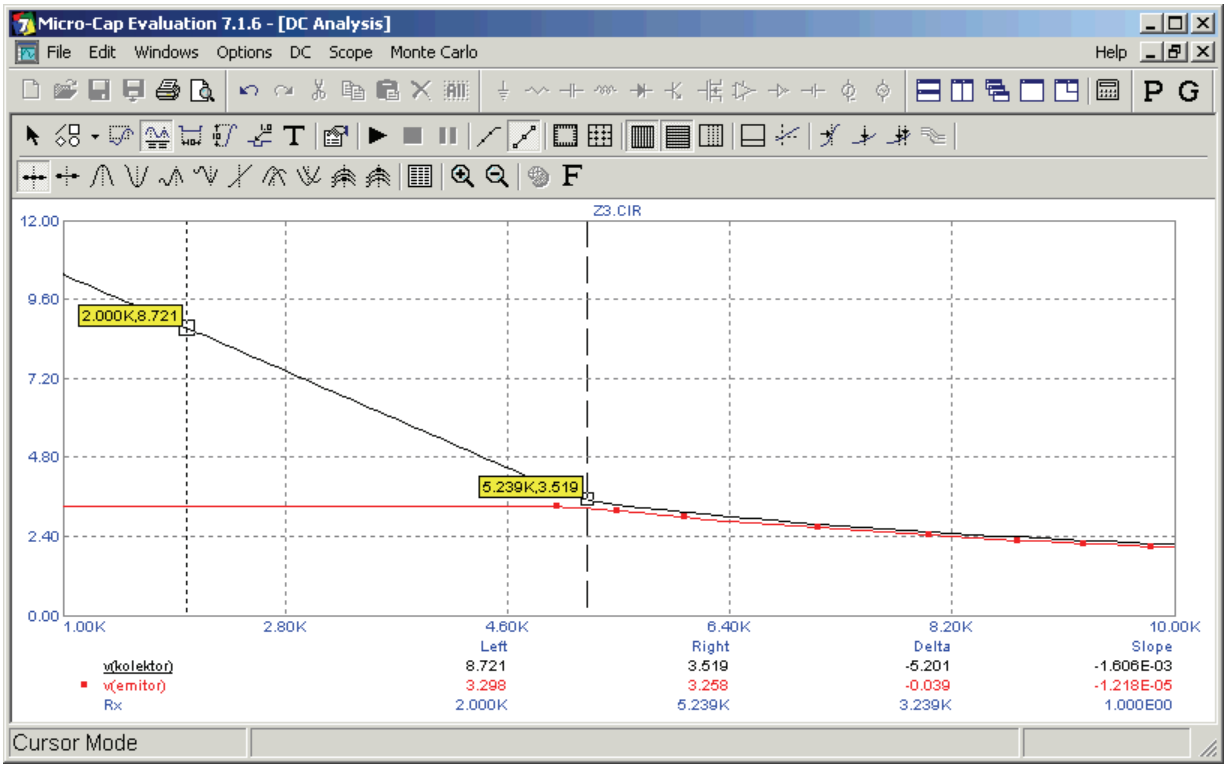
Obr. 9.59: Analyzované teplotní závislosti kolektorového napětí a proudu a proudového zesilovacího činitele tranzistoru.

Nyní zjistíme, jak se budou měnit kolektorové a emitorové napětí při změnách kolektorového odporu *Rx*. Provedeme změny v nastavení okna „*DC Analysis Limits*“ podle **obr. 9.60**.

Z nabídky „*Name*“ v řádku „*Variable* 1“ nyní vybereme „*PARAM RX*“ – symbolickou proměnnou, označující kolektorový odpor. Rozsah změn odporu nastavíme od 1 *k*Ω do 10 *k*Ω. Definujeme dvě křivky do jediného obrázku – závislosti kolektorového a emitorového napětí na odporu *Rx*.



Obr. 9.60: Nastavení požadavků na analýzu závislostí kolektorového a emitorového napětí na *Rx*.



Obr. 9.61: Závislost kolektorového a emitorového napětí na odporu *Rx*.

Výsledky analýzy jsou na **obr. 9.61**. S růstem kolektorového odporu roste taky úbytek napětí na tomto odporu, takže klesá napětí mezi kolektorem a zemí. Při jmenovitém odporu 2 *k*Ω vychází napětí na kolektoru asi 8,7 V a na emitoru asi 3,3 V. Vzroste-li odpor kolektoru nad hodnotu asi 5,2 *k*Ω, napětí na kolektoru poklesne prakticky na velikost napětí na emitoru. Jejich rozdíl, napětí kolektor-emitor, je zanedbatelné. Tranzistor se již nenachází v aktivním zesilovacím režimu.

9.6 Rozšiřující typy analýz

9.6.1 Dynamická stejnosměrná analýza („*Dynamic DC*“)

V tomto zajímavém interaktivním režimu uživatel provádí modifikaci obvodu v schématickém editoru, program okamžitě přepočítává stejnosměrný pracovní bod a zobrazuje výsledky.

Analýza se aktivuje v menu *Analysis/Dynamic DC* (*Alt* + 4). Automaticky je aktivováno zobrazování uzlových napětí (je aktivní ikona  „*Node Voltages*“). Za účelem zobrazení proudů, výkonů a stavů aktivních součástek, resp. spínačů, je třeba aktivovat příslušné další ikony ( ,  , ).

Obvod můžeme modifikovat velmi obecně, od změny konkrétního parametru součástky až po mazání stávajících a přidávání dalších součástek. Kromě toho můžeme zvláštním způsobem krokovat parametry následujících součástek:

- Napětí baterie
- Napětí zdroje „*V*“
- Proud zdroje „*I*“
- Stejnosměrný odpor rezistoru („*Value*“).

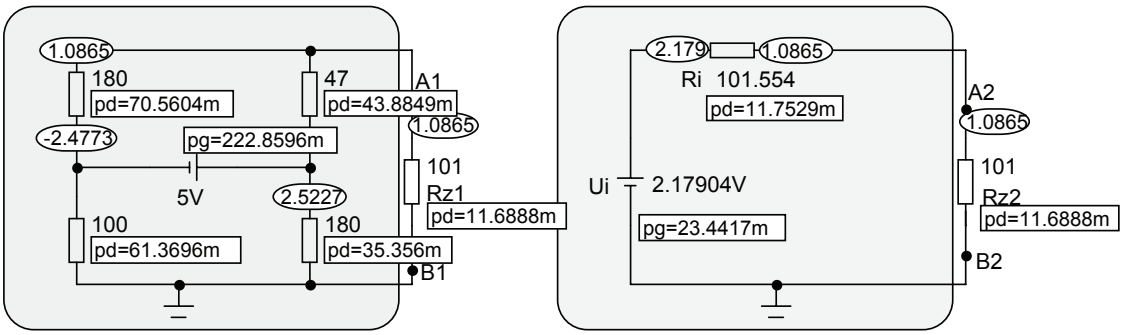
Krokování lze provést jedním ze dvou způsobů:

1. V režimu „*Select*“ klikneme do dané součástky (prosvětlíme její značku). Kurzorovými klávesami ↓↑ měníme parametr součástky po krocích, které jsou dány hodnotou 1% z položky „*SLIDER_MAX*“ (o prvku „*Slider*“ viz dále) v okně atributů součástky. Možné změny jsou vymezeny parametry „*SLIDER_MIN*“ a „*SLIDER_MAX*“. Standardně je „*SLIDER_MIN*“ nastaveno na nulu a „*SLIDER_MAX*“ přebírá hodnotu z položky „*Value*“. Vyznačíme-li více součástek najednou, např. postupným klikáním na jejich značky za současného držení klávesy „*Shift*“, pak dochází k současnému krokování jejich parametrů.
2. Každá z výše vyjmenovaných součástek má tzv. “*Slider*”, t.j. symbol tahového potenciometru, který se standardně u součástky nevykresluje. Vykreslení povolíme zatržením položky “*Show Slider*” v okně *Options/Preferences*. Pomocí myši pak máme možnost tahový potenciometr ovládat.

Co není uvedeno v dokumentaci programu: Za *stejnosměrné atributy zdrojů* se považují u „dynamické DC“ analýzy obecně jiné parametry než u analýzy „*DC*“. Přehledná tabulka je uvedena v příloze P10.1.3. U Laplaceových zdrojů se uvažují hodnoty pro Laplaceův operátor $s = 0$.

Režim dynamické stejnosměrné analýzy se deaktivuje buď opětnou volbou v menu *Analysis/Dynamic DC*, nebo aktivací jiné analýzy.

Ukázka výsledků dynamické „*DC*“ analýzy je na **obr. 9.62**. Program analyzuje obvod, který se ve skutečnosti skládá ze dvou samostatných podobvodů. V levé části je můstkové zapojení, mezi jehož svorky *A*1 a *B*1 je zapojena zátěž *Rz*1. V pravé části je Théveninův model tohoto obvodu se stejným zatěžovacím odporem *Rz*2 = *Rz*1.



Obr. 9.62: Ukázka interaktivního režimu dynamické stejnosměrné analýzy.

Obvody jsou ekvivalentní, takže napětí na obou zátěžích, jejich proudy i výkony by se měly shodovat bez ohledu na velikosti *Rz*1 = *Rz*2.

Tuto ekvivalenci můžeme v režimu dynamické stejnosměrné analýzy ověřit tak, že vyznačíme obě zátěže a kurzorovými klávesami ↓↑ krokujeme jejich velikosti. Na **obr. 9.62** jsou aktivována zobrazení uzlových napětí a výkonů. U výkonů značí symboly „*pg*“ a „*pd*“ tzv. „*power generated*“ – generovaný výkon (vytvářený zdrojem), a „*power dissipated*“ – ztrátový výkon na rezistorech. Sledováním výkonu na zátěži si můžeme navíc ověřit poučku o výkonovém přizpůsobení: k maximu výkonu dojde při rovnosti odporu zátěže a vnitřního odporu Théveninova modelu.

Počet desetinných míst zobrazovaných údajů můžeme změnit v menu

Options/Preferences/Format/Schematic Voltages/Current/Power.

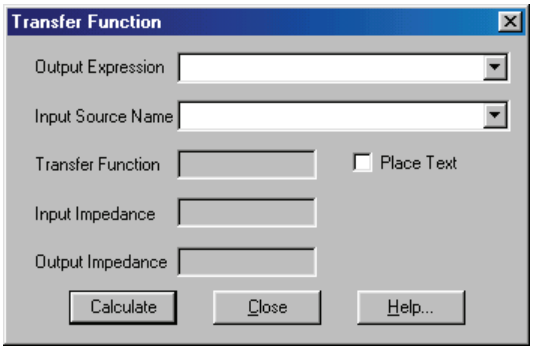
9.6.2 Přenosová funkce („*Transfer Function*“)

V tomto režimu program počítá malosignálové stejnosměrné obvodové funkce. Akumulační prvky jsou při této analýze ignorovány. Znamená to, že vlastně počítáme nízkofrekvenční linearizované parametry obvodu jako jsou střídavá zesílení, impedance apod.

Analýza se aktivuje volbou

Analysis/Transfer Function (Alt + F5).

Objeví se okno podle **obr. 9.63**.



Obr. 9.63: Okno pro zadávání požadavků na obvodovou funkci.

První dvě položky je třeba vyplnit, další tři jsou výstupy analýzy.

- **Output Expression** (vzorec výstupní veličiny): jakýkoliv povolený vzorec. Nejčastěji to však bývají vzorce pro napětí nebo proudy, např. *V*(5), *V*(*out*1,*out*2), *I*(*R*5) apod. Veličinu, kterou vzorec definuje, označme symbolem *Y*.
- **Input Source Name** (jméno vstupního zdroje): Jméno jednoho ze zdrojů ve schématu. Je třeba vybrat z nabídky. Vždy se jedná buď o zdroj napětí nebo proudu. Napětí zdroje napětí, resp. proud zdroje proudu, označme symbolem *X*.
- **Transfer Function** (přenosová funkce).
- **Input Impedance** (vstupní impedance).
- **Output Impedance** (výstupní impedance).

Program provede velmi malou změnu *DC* hodnoty vstupního zdroje a vyhodnocuje vyvolanou změnu výstupní veličiny, která je definována vzorcem. Podíl změny výstupní a vstupní veličiny je přenosová funkce.

Vstupní impedanci program zjišťuje tak, že provede velmi malou změnu *DC* hodnoty vstupního zdroje a vyhodnotí vyvolanou změnu proudu, resp. napětí zdroje (závisí na tom, jde-li o zdroj napětí nebo proudu). Vstupní impedance je poměrem obou hodnot.

Pro výpočet výstupní impedance program nejprve zapojí pomocný zdroj napětí mezi svorky, které jsou definovány vzorcem pro výstupní veličinu. Je-li například vzorec ve tvaru *V*(5,2), zdroj se připojí mezi uzly 5 a 2. V druhé fázi je provedena velmi malá změna *DC* parametru zdroje a program vyhodnotí změnu proudu tímto zdrojem. Výstupní impedance je určena jako poměr napěťové a proudové změny.

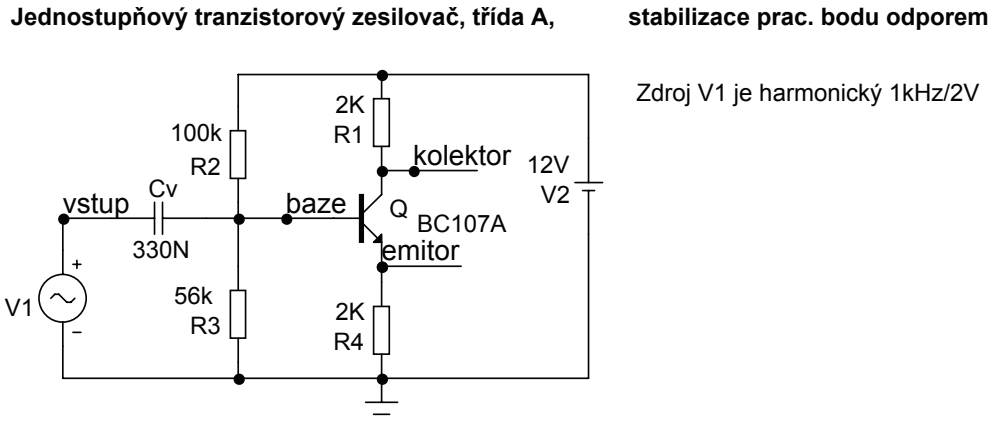
Jestliže vzorec pro výstupní veličinu specifikuje cokoliv jiného než napětí, výpočet výstupní impedance se neprovede a v příslušném políčku se objeví údaj „*N*/*A*" (*Not Available* – není k dispozici).

Po zatržení položky

Place Text (umísti text)

jsou výsledky analýzy umístěny na pracovní plochu jako text.

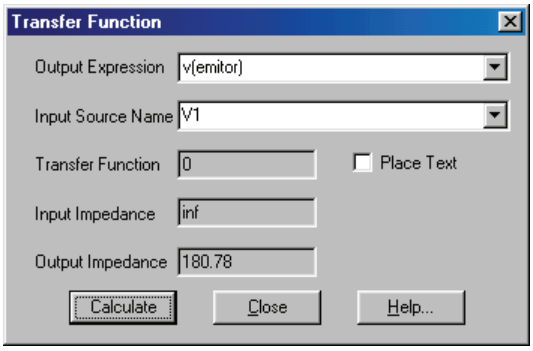
Uvažujme tranzistorový zesilovač na **obr. 9.64**. Jedná se o obvod z **obr. 9.57** bez blokovacího emitorového kapacitoru. Jeho zadání naleznete v dodatkovém souboru **Z2.CIR**. Pokusme se určit výstupní odpor zesilovače, je-li výstupní napětí bráno z emitoru tranzistoru.



Obr. 9.64: Příklad tranzistorového zesilovače. Analyzujeme střídavý výstupní odpor mezi svorkami „*emitor*" – zem.

V nabídce „*Analysis*" vybereme „*Transfer Function*". Položky „*Output Expression*" a „*Input Source Name*" vyplníme tak, jak je vidět na **obr. 9.65**. Po aktivaci „*Calculate*"

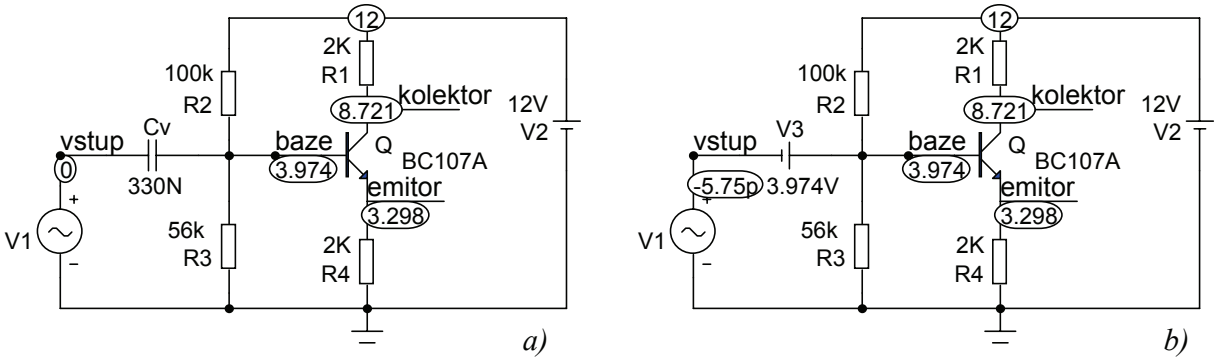
zjistíme, že výstupní impedance („*Output Impedance*“) je 180,78 Ω, přenosová funkce je nulová a vstupní impedance je nekonečná („*inf*“).



Obr. 9.65: Způsob vyplnění okna „*Transfer Function*“ pro výpočet výstupní impedance.

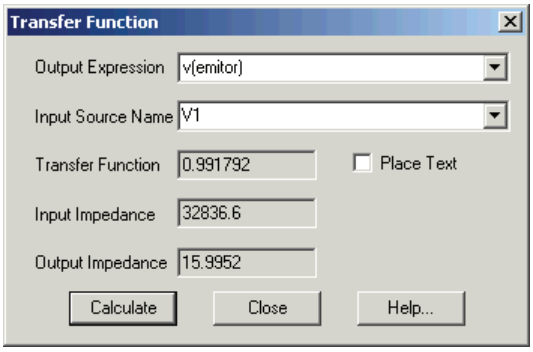
Za dva poslední údaje „může“ vazební kondenzátor C_V, který představuje pro stejnosměrný signál nekonečnou impedanci. Abychom však přesto mohli určit nízkofrekvenční zesílení stupně a jeho vstupní odpor, provedeme jednoduché opatření.

Nejprve určíme dynamickou *DC* analýzou stejnosměrný pracovní bod. Výsledek vidíme na **obr. 9.66** a). Je zřejmé, že na kapacitoru C_V je stejnosměrné napětí 3,974 V. Nahradíme jej tedy baterií o tomto napětí. Na **obr. 9.66** b) je vidět, že po náhradě se stejnosměrné poměry nezměnily (až na napětí na vstupu -5,75 *p*V, což je rozdíl mezi přesným napětím na C_V a napětím, které jsme vložili zaokrouhlené na 3 desetinná místa).



Obr. 9.66: Výsledky dynamické „*DC*“ analýzy a) původního zesilovače, b) upraveného obvodu, kde vazební kapacitor je nahrazen zdrojem napětí.

Analýza upraveného obvodu nyní vede na výsledky podle **obr. 9.67**. Střídavé zesílení je těsně menší než 1 (funkce emitorového sledovače) a vstupní impedance je asi 32,8 *k*Ω, což je o něco méně než činí paralelní kombinace R_2 a R_3 (35,9 *k*Ω).



Obr. 9.67: Výsledky analýzy upraveného obvodu z **obr. 9.66** b).

Oproti původnímu zapojení se však změnila výstupní impedance (srovnejte s údajem na **obr. 9.65**). Je tomu tak proto, že původní hodnota 180,78 Ω byla určena za přítomnosti vazebního kapacitoru, tedy při vstupu (stejnosměrně) naprázdno. Tuto impedanci proto naměříme při relativně nízkých kmitočtech, kdy kapacitor představuje rozpojení. Nová hodnota, necelých 16 Ω, platí pro pracovní kmitočty, kdy se kapacitor chová jako střídavý zkrat, tj. když je na něm pouze stejnosměrné napětí.

9.6.3 Citlivostní analýza („*Sensitivity*“)

Je počítána stejnosměrná citlivost jedné nebo více veličin, vyjádřené vzorcem nebo vzorci, na jednu nebo více vstupních proměnných.

Citlivost je počítána takto:

Změna ve vzorci/malá změna vstupní proměnné.

Malou změnou se myslí jedna milióntina hodnoty vstupní proměnné v pracovním bodu, nebo číslo 10^{-6}, pokud je tato hodnota nulová.

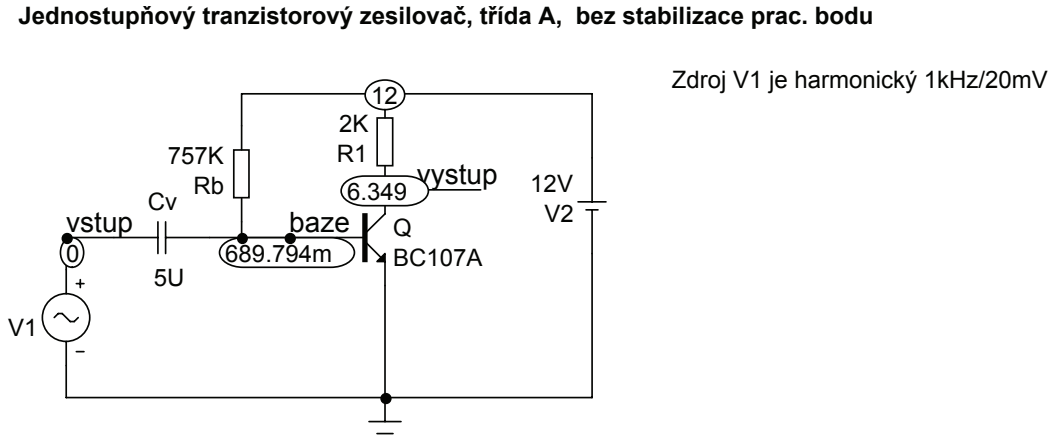
Citlivost je tedy derivací veličiny, vyjádřené vzorcem, podle vstupní proměnné, v pracovním bodě. Princip je proto stejný jako u výpočtu přenosové funkce. Rozdíl je v tom, že nyní můžeme počítat derivace podle všech parametrů obvodu a jeho součástek, které ovlivňují stejnosměrné poměry. U přenosové funkce se jednalo jen o výpočty derivací podle *DC* hodnot zdrojů.

Citlivostní analýza je důležitá k zjišťování, do jaké míry mají tolerance parametrů součástek kolem jmenovitých hodnot vliv na sledovaný parametr celého obvodu. Typickým příkladem je otázka, o kolik procent se změní klidový kolektorový proud tranzistoru, změní-li se proudový zesilovací činitel o 1 %.

Program bohužel neumí počítat jinou citlivost než stejnosměrnou. Jinými slovy, nelze například počítat citlivost kmitočtové charakteristiky v daném frekvenčním pásmu.

Jak uvidíme dále, program umožňuje současně navolit výpočet citlivostí na více parametrů součástek a jejich modelů. Při volbě velkého počtu parametrů mohou výpočty trvat značně dlouhou dobu, znamená-li to mnohonásobně opakovaný výpočet stejnosměrného pracovního bodu.

Konkrétní postupy při citlivostní analýze ukážeme na příkladu tranzistorového zesilovače podle **obr. 9.68**. Obvod je možné nalézt v dodatkovém souboru **Z1.CIR**.



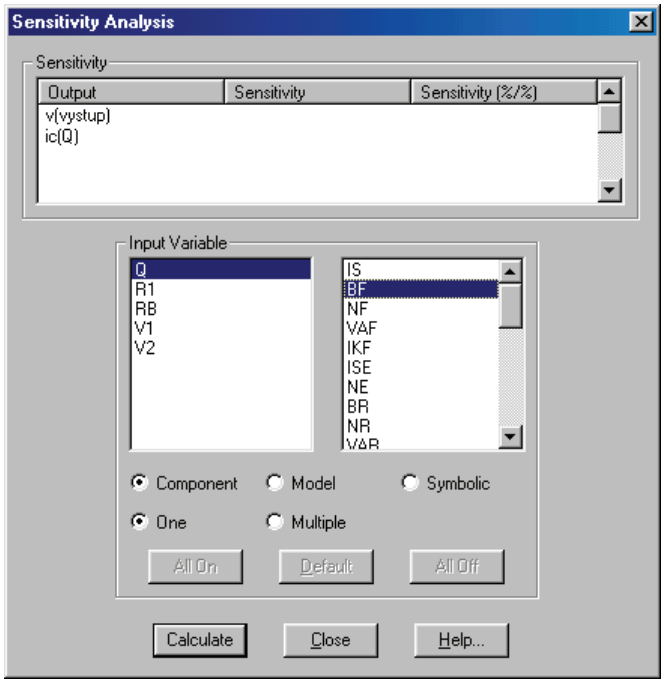
Obr. 9.68: Tranzistorový zesilovač bez stabilizace stejnosměrného pracovního bodu.

Obrázek zachycuje zobrazení uzlových napětí stejnosměrného pracovního bodu, vypočtených dynamickou *DC* analýzou. Bude nás zajímat citlivost výstupního napětí a proudu kolektoru na proudovém zesilovacím činiteli tranzistoru. Ten se skrývá pod parametrem „*BF*“ v modelu tranzistoru a jeho hodnota pro tranzistor *BC*107*A* je 375,5.

Citlivostní analýza se aktivuje pomocí menu

„Analysis/Sensitivity” (Alt + 6)

Objeví se okno „*Sensitivity Analysis*“, které vyplníme podle **obr. 9.69**.



Obr. 9.69: Okno pro zadávání podmínek a vyhodnocování citlivostní analýzy.

Do sloupce „*Output*“ ve skupině „*Sensitivity*“ se zapisují vzorce, jejichž citlivosti určujeme. V sloupci „*Sensitivity*“ se po výpočtu objevují příslušné citlivosti, počítané podle již uvedeného vzorce

Změna ve vzorci/malá změna vstupní proměnné.

V sloupci „*Sensitivity* %/%“ se objeví citlivosti, vyjádřené jako procentuální změna ve vzorci dělená procentuální změnou vstupní proměnné.

V okně „*Input Variable*“ – vstupní proměnná se objeví seznam proměnných v závislosti na tom, která z níže uvedených položek je aktivní:

- **Component** - součástka
- **Model** - model součástky
- **Symbolic** - symbolická proměnná.

V sloupci napravo se objeví parametr, resp. parametry příslušející vybrané proměnné, které mají vliv na stejnosměrné poměry v obvodu. **Obr. 9.69** ukazuje situaci, kdy je vybrán parametr „*BF*“ tranzistoru *Q*.

Dále se musíme rozhodnout pro jeden z režimů „*One*“ (jedna proměnná) nebo „*Multiple*“ (více proměnných). V režimu „*One*“ jsou počítány citlivosti vzorců na jedinou proměnnou. V režimu „*Multiple*“ se počítají odděleně citlivosti na celou řadu proměnných,

které vyznačíme níže uvedeným způsobem. V tom případě se výsledky citlivostní analýzy zapíší do externího souboru s příponou **.SEN** a obsah souboru se automaticky otevře ve zvláštním okně.

V režimu „*Multiple*“ se uvolní nabídky „*All On*“, „*Default*“, „*All Off*“.

„**All On**“: Vyznačí se všechny parametry všech součástek. Citlivostní analýza se vykoná pro všechny tyto parametry. To může vést k časově náročným výpočtům, obsahuje-li například obvod několik součástek, jejichž modely obsahují desítky parametrů.

„**Default**“: Vyznačí se jen některé typické přednastavené parametry.

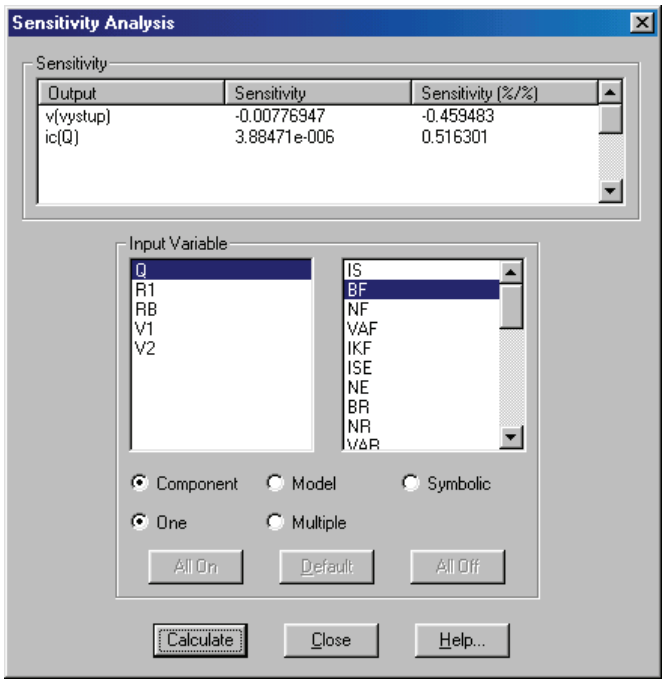
„**All Off**“: Zruší se aktuální nastavení parametrů. Je na uživateli, aby vybral parametry ručně.

Ruční vybrání parametrů: Klikneme na jméno součástky v poli „*Input Variable*“ a poté na vybraný parametr vpravo. Chceme-li vybrat více parametrů součástky, vybíráme při současném držení klávesy „*Shift*“. Po vybrání parametrů klikneme na jméno další součástky a postup opakujeme.

Citlivosti vypočteme kliknutím na položku „*Calculate*“.

Máme-li tedy nastaveny položky v okně „*Sensitivity Analysis*“ tak, jak je ukázáno na **obr. 9.69**, proběhne výpočet citlivosti kolektorového napětí a proudu kolektoru na parametr „*BF*“ (proudový zesilovací činitel) tranzistoru. Výpočtem můžeme simulovat situaci, co se stane se stejnosměrným pracovním bodem po změně zesilovacího činitele, způsobené buď změnou teploty, nebo výměnou vadného tranzistoru za jiný kus.

Výsledky citlivostní analýzy jsou na **obr. 9.70**. Je možné je interpretovat takto:



Obr. 9.70: Výsledné citlivosti stejnosměrného výstupního napětí a kolektorového proudu zesilovače z **obr. 9.68** na proudový zesilovací činitel tranzistoru.

- Při vzrůstu proudového zesilovacího činitele tranzistoru o jedničku klesne kolektorové napětí asi o 7,8 *m*V a kolektorový proud vzroste asi o 3,9 μA.
- Při vzrůstu proudového zesilovacího činitele tranzistoru o 1 % klesne kolektorové napětí asi o 0,46 % a kolektorový proud vzroste asi o 0,52 %.

Pohledem na schéma analyzovaného zesilovače na **obr. 9.68** zjistíme, že v obvodu nepůsobí mechanismy stabilizace pracovního bodu, takže získané citlivosti jsou poměrně velké. Zkuste si výpočet citlivostí pro zesilovač z **obr. 9.64**, kde je provedeno opatření pro stabilizaci. Zadání je uloženo v dodatkovém souboru **Z2.CIR**. Přesvědčte se o tom, že při vzrůstu zesilovacího činitele o 1 % nyní klesne kolektorové napětí o pouhých 0,016 % a kolektorový proud se zvětší jen o 0,04 %.

9.7 Analyzační režimy

V části 9.1 byly uvedeny tyto analyzační režimy: Krokování, teplotní analýza, vyhodnocovací analýza, statistická analýza a optimalizace.

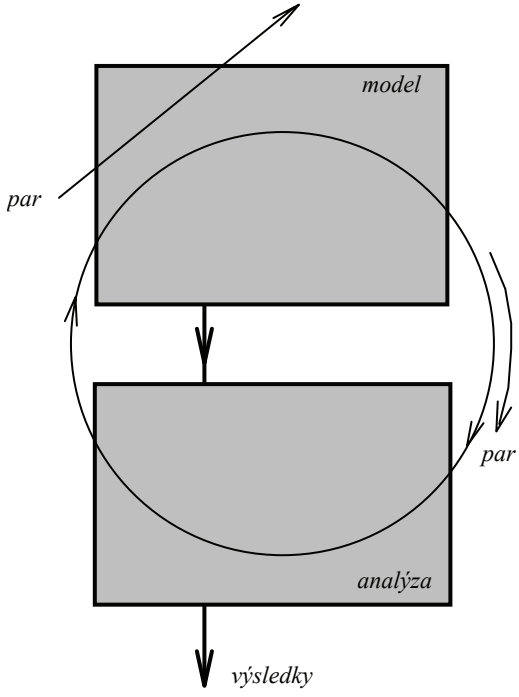
Jak uvidíme, v režimech krokování, statistické analýzy a optimalizace dochází k variaci parametrů součástek v analyzačních smyčkách. Způsob těchto variací můžeme dále ovlivnit volbou tzv. veřejných nebo privátních knihoven modelů součástek. O této problematice se zmíníme v závěrečné části 9.7.6.

9.7.1 Krokování („*Stepping*“)

Krokování je užitečný režim, který můžeme využít ve všech třech základních typech analýz. Použití tohoto režimu je vyloučeno při současném využívání režimu statistické analýzy (viz část 9.7.4).

Na režim krokování jsme již narazili v části 6.2.1 u SNAPu. U MicroCapu je možno krokovat buď jeden nebo více parametrů.

Princip krokování jednoho parametru je ukázán na **obr. 9.71**. Změna parametru součástky má za následek změnu modelu celého obvodu. Vždy po provedení změny proběhne analýza. Vše se opakuje ve výpočetní smyčce tak dlouho, dokud nejsou vyčerpány všechny hodnoty parametru, definované pro krokování.

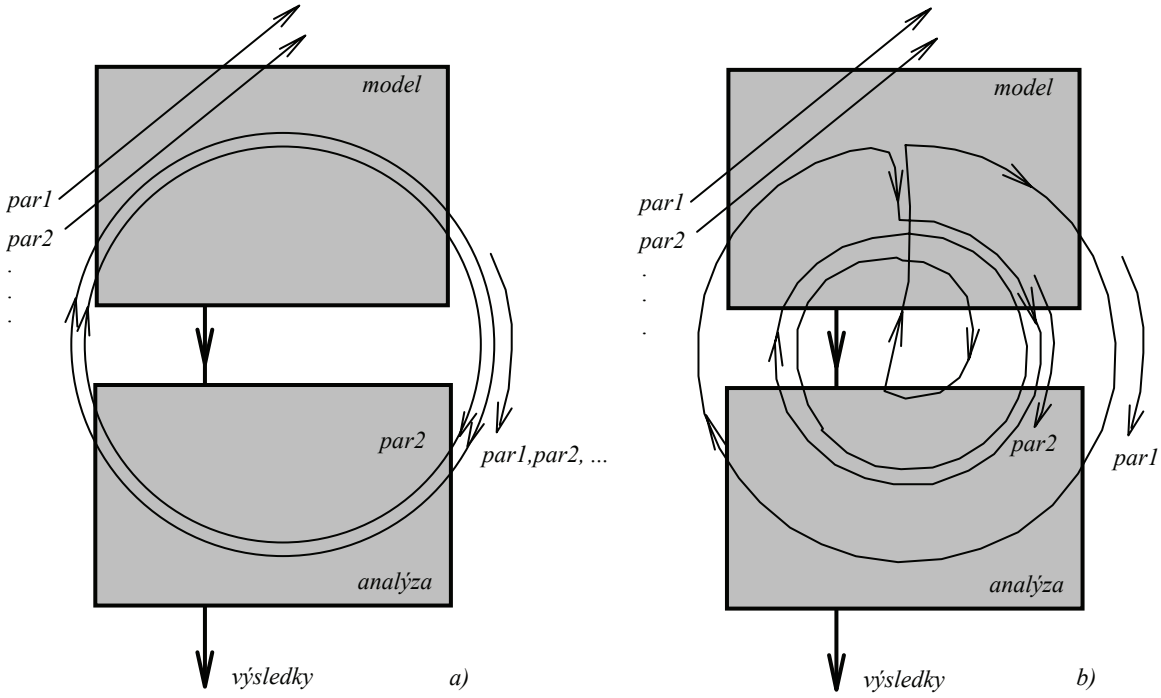


Obr. 9.71: Schéma jednoduchého krokování.

Při krokování více parametrů lze zvolit dvě metody:

- „**Simultaneous**“ – simultánní, současné krokování.
- „**Nested Loops**“ – krokování „každý s každým“, ve vnořených smyčkách.

Vysvětlení je na **obr. 9.72**. Při *simultánním krokování* (obr. a) je třeba, aby všechny parametry byly krokovány ve stejném počtu kroků. V každém dalším kroku je nastavena další definovaná hodnota každého parametru.



Obr. 9.72: Schéma krokování a) simultánního, b) „každý s každým“.

Krokujeme-li například odpor R_1 v hodnotách

$$(1\ 2\ 5\ 10)\ k\Omega$$

a kapacitu $C1$ v hodnotách

$$(1\ 10\ 100\ 1000)\ n\mathrm{F},$$

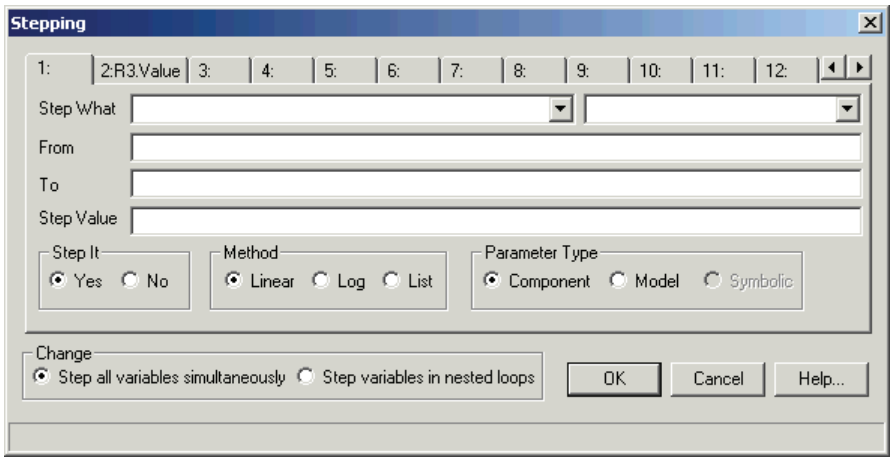
pak při simultánním krokování proběhne analýza ve 4 krocích, kdy dvojice [R_1 C_1] bude postupně nabývat těchto hodnot:

[1 *k*Ω 1 *n*F], [2 *k*Ω 10 *n*F], [5 *k*Ω 100 *n*F], [10 *k*Ω 1000 *n*F].

Při krokování „každý s každým“ (**obr. 9.72** b) proběhne celkem 16 analýz s těmito dvojicemi:

[1 *k*Ω 1 *n*F], [1 *k*Ω 10 *n*F], [1 *k*Ω 100 *n*F], [1 *k*Ω 1000 *n*F],
[2 *k*Ω 1 *n*F], [2 *k*Ω 10 *n*F], [2 *k*Ω 100 *n*F], [2 *k*Ω 1000 *n*F],
[5 *k*Ω 1 *n*F], [5 *k*Ω 10 *n*F], [5 *k*Ω 100 *n*F], [5 *k*Ω 1000 *n*F],
[10 *k*Ω 1 *n*F], [10 *k*Ω 10 *n*F], [10 *k*Ω 100 *n*F], [10 *k*Ω 1000 *n*F].

Režim krokování se aktivuje z okna příslušné analýzy „*Analysis Limits*“ kliknutím do položky „*Stepping*“, případně při aktivované analýze horkou klávesou *F*11. Objeví se okno „*Stepping*“ podle **obr. 9.73**.



Obr. 9.73: Okno „*Stepping*“ pro zadávání podmínek krokování.

V okně je naskládáno celkem 20 karet pro nastavení krokovaných parametrů. Význam jednotlivých položek:

„**Step What**“ – co se má krokovat. Je třeba vybrat ze seznamu. Nabídka závisí na zvolené položce v sekci „*Parameter Type*“:

„Component“ – součástka
„Model“ – model součástky
„Symbolic“ – symbolická proměnná, definovaná příkazem *.DEFINE*.

Po výběru konkrétní položky v okně „*Step What*“ dojde k naplnění obsahu vedlejšího okna, kde provedeme konkrétnější výběr (např. u pasivních součástek to bude pouze „*Value*“ – hodnota, u tranzistoru pak parametry jeho modelu, apod.).

„**From**“ – počáteční hodnota krokovaného parametru.

„**To**“ – konečná hodnota krokovaného parametru.

„**Step Value**“ – hodnota kroku. Význam závisí na metodě krokování:

„**Metod**“ – Linear, Log, List

Linear: hodnota kroku je postupně přičítána k počáteční hodnotě tak dlouho, až výsledek dosáhne, resp. překročí konečnou hodnotu.
Log: počáteční hodnota je postupně násobena hodnotou kroku tak dlouho, až výsledek dosáhne, resp. překročí konečnou hodnotu.
List: hodnoty krokovaného parametru oddělené čárkami.

„**Change**“ – volba strategie krokování:

Step all variables simultaneously – simultánní krokování.
Step variables in nested loops – krokování „každý s každým“.

Krokování každého z parametrů se povoluje volbou

„Step It“ – Yes.

Krokovat lze téměř cokoliv. Výjimky jsou uvedeny v dokumentaci programu.

Křivky, vzniklé analýzou v režimu krokování, se vykreslují do společného obrázku. V režimu „*Cursor*“ lze mezi nimi přepínat pomocí kurzorových kláves ↑↓.

9.7.2 Teplotní analýza

Druhy „simulačních teplot“

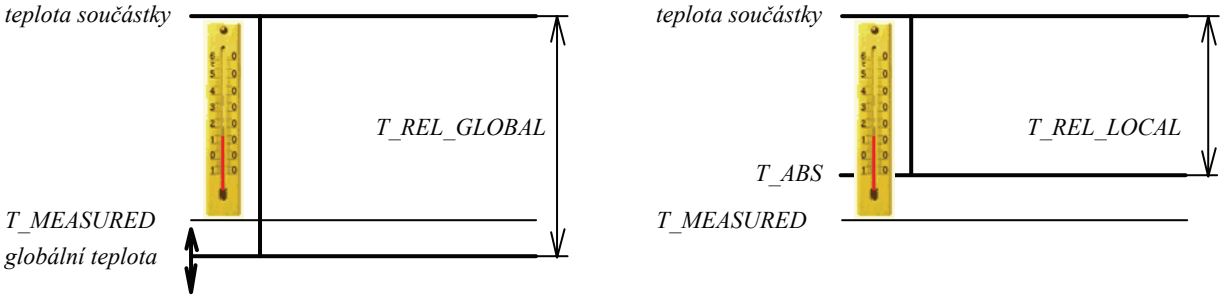
V modelech rezistorů, kapacitorů, induktorů a zejména polovodičových prvků figurují teplotní závislosti nejrůznějších parametrů. V příslušných vzorcích se obvykle vyskytují dvě teploty:

Teplota součástky – Pracovní teplota součástky, která bezprostředně ovlivňuje její elektrické parametry. U integrovaných obvodů se nerozlišuje mezi teplotou čipu, pouzdra, chladiče a okolí.

Nominální teplota – Vztažná teplota, při níž byly změřeny konkrétní parametry součástky uváděné v knihovně.

V základním režimu se teplota součástky zadává v položce „*Temperature*“ v okně „*Analysis Limits*“. Nominální teplota *TNOM* je definována v globálních podmínkách simulace („*Global Settings*“) a standardně je nastavena na 27°C.

V některých případech je však třeba ošetřit situace, kdy různé součástky mají různé teploty, kdy teploty různých součástek jsou navzájem posunuty o definovaný rozdíl, nebo kdy při krokování teploty se teplota vybraných součástek nemění (jsou uzavřeny v prostoru termostatu apod.). Proto vznikl nad jednoduchým systémem “teplota součástky – nominální teplota” složitější systém teplotní simulace, který je schématicky znázorněn na **obr. 9.74**.



Obr. 9.74: Vztahy mezi teplotou součástky, globální teplotou a teplotami *T_MEASURED, T_ABS, T_REL_GLOBAL* a *T_REL_LOCAL.*

Teplotní analýza se odvíjí od tzv. globální teploty. U MicroCapu se jedná o údaj, který je zapsán v položce „*Temperature*“ v okně „*Analysis Limits*“. V modelech teplotně závislých prvků je možné zadat parametry

T_MEASURED, T_ABS, T_REL_GLOBAL a T_REL_LOCAL.

Pokud nejsou uživatelem definovány, platí jejich implicitní hodnoty:

T_MEASURED = TNOM,
T_ABS .. není aktivní,
T_REL_GLOBAL = 0,
T_REL_LOCAL .. není aktivní.

Připomínáme, že *TNOM* je nominální teplota, definovaná v globálních podmínkách simulátoru.

Pomocí uvedených parametrů můžeme modelovat teplotní režim každé součástky zvlášť.

Parametrem *T_MEASURED* lze definovat teplotu, pro niž jsou platné hodnoty parametrů modelu součástky. Nastavování této teploty připadá v úvahu zejména při tvorbě vlastních modelů. Modely dodané výrobci součástek jsou obvykle platné pro standardní nastavení

$$T\_MEASURED = TNOM = 27\,°C.$$

Následující příklad ukazuje příkaz model pro tranzistor *Q* typu *NMOS*,

.Model Q NMOS (T_MEASURED=32)

Všechny parametry tranzistoru mají implicitní hodnoty, které ale platí pro teplotu 32°C.

Na obr. **obr. 9.74** a) je naznačeno, že teplota součástky je odvozena od globální teploty, tj. od položky „*Temperature*“ v okně „*Analysis Limits*“, přes parametr *T_REL_GLOBAL* (relativní globální teplotu). Protože tento parametr je implicitně nulový, pak teplota součástky je přímo rovna globální teplotě. Parametrem *T_REL_GLOBAL* můžeme teplotu součástky zvětšovat, resp. zmenšovat o libovolnou hodnotu oproti globální teplotě.

Zapíšeme-li například do políčka „*Temperature*“ teplotu 20°C, pak teplota všech tranzistorů, jejichž model je *N*1,

.MODEL N1 NPN(BF=50 T_REL_GLOBAL=40)

bude uvažována 60°C.

Na obr. **obr. 9.74** b) je ukázáno, že parametrem *T_ABS* lze nastavit fixní teplotu, nezávislou na globální teplotě. Teplota součástky může být oproti této teplotě „posunuta“ o parametr *T_REL_LOCAL*. Tento teplotní režim však vyžaduje vysvětlení formou příkladu:

V políčku „*Temperature*“ je zapsána teplota 27°C. Uvažujme dva příkazy, které definují dva modely:

.MODEL N1 NPN(BF=50 T_ABS=30)
.MODEL N2 AKO:N1 NPN(T_REL_LOCAL=25)

První příkaz definuje model tranzistoru s implicitními hodnotami parametrů vyjma proudového přenosu, který má hodnotu 50. Tranzistor má fixní teplotu 30°C nezávisle na tom, že globální teplota je 27°C.

Model *N*2 je odvozen z modelu *N*1 konvencí *AKO* („*A Kind Of*“ – viz část 8.3.1). Přebírá všechny vlastnosti modelu *N*1 a rozšiřuje je o relativní lokální teplotu 25°C. Všechny tranzistory, jejichž model je *N*2, tedy budou mít teplotu 55°C nezávisle na globální teplotě.

Použití relativní lokální teploty je tedy vázáno na příkaz *AKO*, neboť finální model musí být odvozen z modelu, kde je definována absolutní teplota *T_ABS*. Tento příkaz je třeba zadat „ručně“ jako „*grid text*“, nevygeneruje se automaticky.

Teplotní závislosti pasivních součástek

Požadujeme-li modelovat teplotní závislosti odporů, kapacit nebo indukčností, musíme nejprve příslušné součástce přiřadit teplotní součinitele: lineární (*TC*1) a v případě potřeby kvadratický (*TC*2) pro součástky *R*, *L* a *C*, případně exponenciální (*TCE*) pro součástky typu *R*. Hodnota parametru (odpor, indukčnost, kapacita) je pak násobena teplotním faktorem *TF*, který je dán vzorci

$$TF = 1 + TC1 * (T - T\_MEASURED) + TC2 * (T - T\_MEASURED)^2$$

při uvažování lineárního a kvadratického součinitele a

$$TF = 1{,}01^{TCE*(T-T\_MEASURED)}$$

pro exponenciální součinitel.

Připomeňme, že nespecifikujeme-li teplotu *T_MEASURED*, její implicitní hodnota rovna nominální hodnotě *TNOM* z globálních nastavení simulátoru.

Implicitní hodnoty všech teplotních součinitelů jsou nulové.

Exponenciální součinitel se využije k modelování teplotních závislostí jen výjimečně. Nejběžnější je použití lineárního součinitele, který bývá udáván v katalogových listech pasivních součástkách v jednotkách *ppm*/°C. Zde *ppm* je zkratka „*Parts Per Million*“ = části z miliónu. Údaj v těchto jednotkách je třeba dělit miliónem. Tak například typické hodnoty *TC*1 pro odpory, kapacity a indukčnosti, které jsou 3000 *ppm*/°C, 1500 *ppm*/°C a 200 *ppm*/°C, je třeba zadat v hodnotách 3*m*, 1,5*m* a 0,2*m*.

Teplotní součinitele *TC*1 a *TC*2 je možné zadat buď přímo do položky “*Value*”, nebo prostřednictvím příkazu *.MODEL*. Exponenciální součinitel odporu lze zadat jen prostřednictvím modelu.

Zadávání do položky „*Value*“

Syntaxe položky „*Value*“ pro prvky *R*, *L* a *C* je obecně takováto:

hodnota [*TC*=*hodnota_TC*1[,*hodnota_TC*2]]

V hranatých závorkách jsou nepovinné údaje. Pokud nejsou zadány, při analýze jsou uvažovány jejich implicitní (zde nulové) hodnoty.

Příklady zadávání odporů:

1*k*	odpor 1 *k*Ω, *TC*1 = 0, *TC*2 = 0
1*k TC*=2*m*	odpor 1 *k*Ω, *TC*1 = 2*m*, *TC*2 = 0
1*k TC*=2*m*,1u	odpor 1 *k*Ω, *TC*1 = 2*m*, *TC*2 = 1μ

Položku „*Value*“ můžeme zadat rovněž pomocí příkazu *.DEFINE* (viz část 9.4.3.4).

Zadávání pomocí modelu

Součástkám typu *R*, *L* a *C* lze přiřadit modely. Přiřazování modelů se provede v editačním okně součástky. Pak dojde k automatickému vygenerování příkazu *.MODEL*. Druhou možností je přímý zápis příkazu *.MODEL* jako „*grid textu“*. V parametrech modelu jsou i teplotní koeficienty *TC*1 a *TC*2 a v případě rezistoru i *TCE*. Definujeme-li v modelu číselnou hodnotu parametru *TCE*, pak jsou parametry *TC*1 a *TC*2 ignorovány a teplotní simulace probíhá podle exponenciálního zákona. V parametrech modelu jsou i všechny výše uvedené teplotní parametry

T_MEASURED, T_ABS, T_REL_GLOBAL a T_REL_LOCAL,

takže tepelný režim takového pasivního prvku pak lze nastavit individuálně.

Poznámka: Definujeme-li teplotní koeficienty nadvakrát, tj. příkazem *.MODEL* i v položce „*Value*“, pak analýza proběhne podle koeficientů v položce „*Value*“.

Teplotní závislosti polovodičových součástek

Jsou simulovány složitými vnitřními modely. V modelech tranzistorů a diod je kromě parametrů

T_MEASURED, T_ABS, T_REL_GLOBAL a T_REL_LOCAL

vždy několik teplotních koeficientů, které ovlivňují teplotní vlastnosti součástky.

Teplotní závislosti se uplatňují i u operačních zesilovačů při úrovni modelování „*Level*“ 3. Uživateli jsou však mechanismy těchto závislostí skryty. Jedná se o model, obsahující po převodu do formátu SPICE (viz část 8.3.2) tranzistorové a diodové struktury, tedy součástky s teplotními závislostmi.

9.7.3 Vyhodnocovací analýza („*Performance Analysis*“)

Vyhodnocovací analýza a její režimy

Na výsledky analýzy je možné pohlížet jako na značné množství numerických dat. K jejich zpracování jsou určeny tzv. vyhodnocovací funkce („*Performance Functions*“). Tyto funkce slouží k hledání „jednobodových“ charakteristik celých křivek, jako jsou například lokální či globální maxima a minima křivek, doby náběhu impulsů, šířky impulsů, opakovací kmitočet či perioda signálu a řada dalších.

Je třeba říci, že vyhodnocovací analýza je v programech SPICE propracována podstatně lépe než v MicroCapu. V SPICE může uživatel psát speciální rutiny, v kterých je možné využívat dostupné vyhodnocovací funkce k řešení složitých úloh. V MicroCapu jsme omezeni na aplikaci vybrané funkce a případně na její kombinování s dalšími funkcemi. Nezbývá než doufat, že v dalších verzích MicroCapu už bude vyhodnocovací analýza na vyšší úrovni.

Vyhodnocovací funkce mohou být použity ve dvou různých režimech:

Okamžitý režim („*Immediate Mode*“):

Pracujeme-li v okně grafu, klikneme na liště do ikony  „*Go To Performance*“. Otevře se stejnojmenné okno, v němž vybereme příslušnou vyhodnocovací funkci a nastavíme její parametry. V dialogovém poli se objeví číselný výsledek a do příslušných bodů křivky se umístí kurzory.

Vyhodnocovací grafy („*Performance plots*“):

V tomto režimu se zpracovávají výsledky vícenásobné analýzy, vzniklé krokováním parametrů. Výsledkem jsou grafy závislostí vyhodnocovacích funkcí na krokovaných parametrech, například závislost doby náběhu impulsu na časové konstantě *RC* článku, závislost činitele jakosti a rezonančního kmitočtu filtru na pracovních odporech apod. MicroCap umožňuje tvorbu dvoj – i třírozměrných grafů (demoverze je omezena na dvojrozměrné grafy). K zvláštnímu způsobu vícenásobné analýzy dochází při statistické analýze (viz část 9.7.4). Vyhodnocovací grafy pak mají podobu tzv. histogramů.

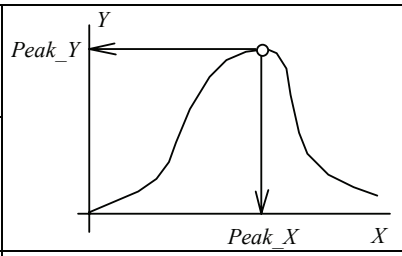
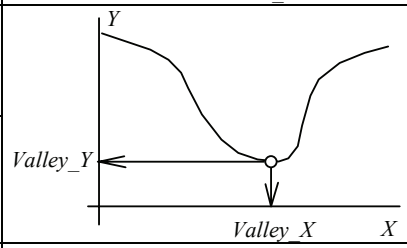
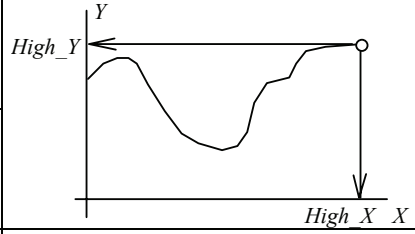
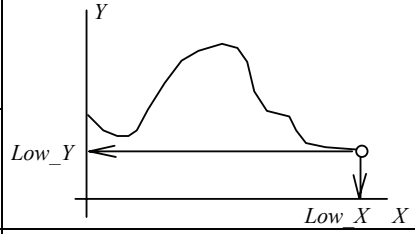
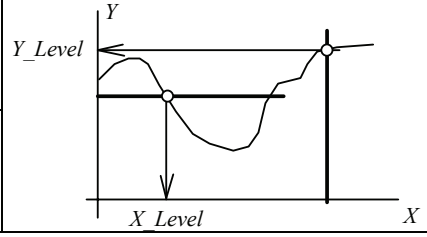
Vyhodnocovací funkce

V tabulkách **Tab. 9.2** a **Tab. 9.3** jsou uvedeny vyhodnocovací funkce MicroCapu.

Pro úplnost dodejme, že v okamžitém režimu jsou kromě uvedených vyhodnocovacích funkcí k dispozici i další vyhledávací funkce, jako je například vyhledávání inflexních bodů apod.

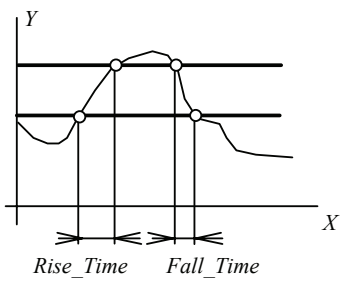
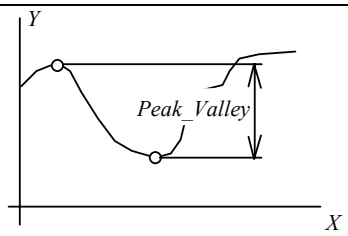
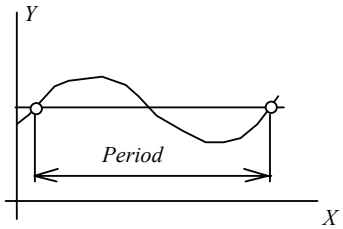
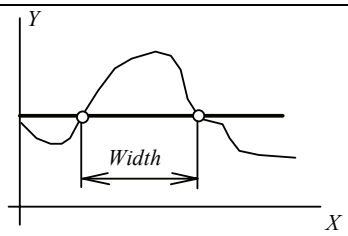
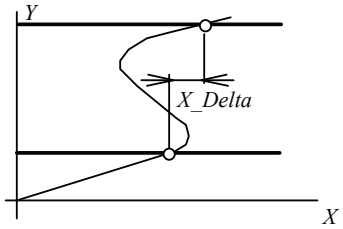
Demoverze MicroCapu umožňuje pracovat pouze s funkcí „*Rise_Time*“.

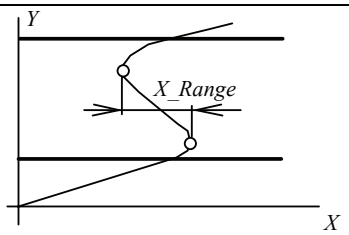
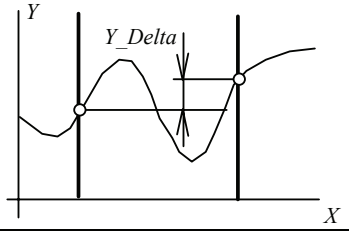
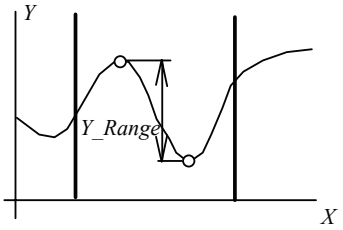
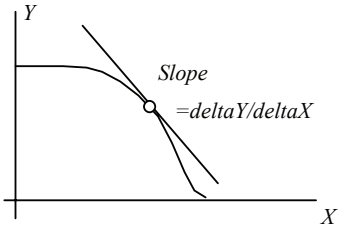
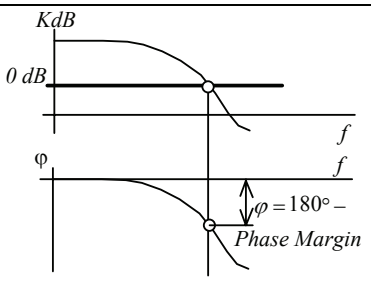
Funkce typu „Jeden měřicí bod“:

Funkce	Popis		Použití
Peak_X	Vyhledá lokální maximum křivky. Výsledkem je *X*-ová souřadnice tohoto maxima.		Vyhledávání rezonančních a přechodových překmitů, maxim výkonů, maximálních útlumů apod.
Peak_Y	Vyhledá lokální maximum křivky. Výsledkem je *Y*-ová souřadnice tohoto maxima.		
Valley_X	Vyhledá lokální minimum křivky. Výsledkem je *X*-ová souřadnice tohoto minima.		
Valley_Y	Vyhledá lokální minimum křivky. Výsledkem je *Y*-ová souřadnice tohoto minima.		
High_X	Vyhledá absolutní maximum křivky. Výsledkem je *X*-ová souřadnice tohoto maxima.		Vyhledávání maximálních a minimálních hodnot souřadnic analyzovaných závislostí.
High_Y	Vyhledá absolutní maximum křivky. Výsledkem je *Y*-ová souřadnice tohoto maxima.		
Low_X	Vyhledá absolutní minimum křivky. Výsledkem je *X*-ová souřadnice tohoto minima.		
Low_Y	Vyhledá absolutní minimum křivky. Výsledkem je *Y*-ová souřadnice tohoto minima.		
X_Level	Vyhledá bod o specifikované *Y*-souřadnici. Výsledkem je *X*-ová souřadnice tohoto bodu.		Přesné nastavování jedné ze souřadnic křivky.
Y_Level	Vyhledá bod o specifikované *X*-souřadnici. Výsledkem je *Y*-ová souřadnice tohoto bodu.		

Tab. 9.2: Vyhodnocovací funkce („*Performance Functions*“) typu „*jeden měřicí bod*“.

Funkce typu „Dva měřicí body“:

Funkce	Popis		Použití
Rise_Time	Vyhledá dva průsečíky rostoucího úseku křivky se specifikovanými *Y*-úrovněmi „*Low*“ a „*High*“. Výsledkem je rozdíl *X*-ových souřadnic těchto bodů.		Měření doby trvání náběžných a sestupných hran impulsů.
Fall_Time	Vyhledá dva průsečíky klesajícího úseku křivky se specifikovanými *Y*-úrovněmi „*Low*“ a „*High*“. Výsledkem je rozdíl *X*-ových souřadnic těchto bodů.		
Peak_Valley	Vyhledá sousední lokální maximum a minimum křivky. Výsledkem je rozdíl *Y*-ových souřadnic těchto bodů.		Měření mezivrcholových hodnot signálů.
Period	Vypočte střední hodnotu průběhu. Pak hledá sousední průsečíky rostoucí křivky s úrovní střední hodnoty. Výsledkem je rozdíl *X*-ových souřadnic těchto průsečíků, neboli opakovací perioda.		Měření opakovacích period a kmitočtů.
Frequency	Pracuje na stejném principu jako funkce „*Period*“. Výsledkem je převrácená hodnota opakovací periody, neboli opakovací kmitočet.		
Width	Vyhledá sousední průsečíky křivky se specifikovanou *Y*-ovou úrovní. Výsledkem je rozdíl mezi *X*-ovými souřadnicemi těchto průsečíků (např. šířka impulsu).		Měření šířek impulsů.
X_Delta	Vyhledá sousední průsečíky křivky se specifikovanými *Y*-ovými úrovněmi „*Low*“ a „*High*“. Výsledkem je rozdíl mezi *X*-ovými souřadnicemi těchto průsečíků.		Měření strmosti decibelových charakteristik v analýze „*AC*“.

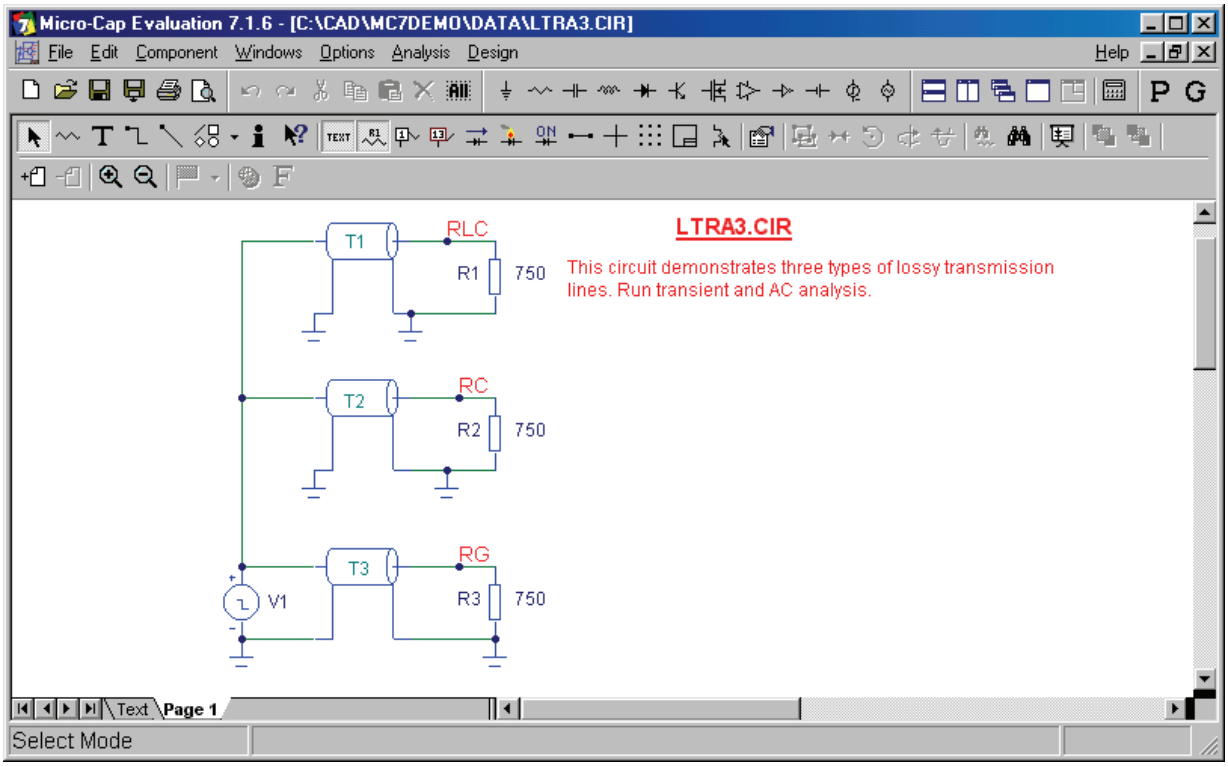
X_Range	Vyhledá sousední průsečíky křivky se specifikovanými *Y*-ovými úrovněmi „*Low*“ a „*High*“. V úseku křivky mezi těmito body najde maximální a minimální *X*-ovou souřadnici. Výsledkem je jejich rozdíl.		Měření vlastností charakteristik typu „*S*“.
Y_Delta	Vyhledá sousední průsečíky křivky se specifikovanými *X*-ovými úrovněmi „*Low*“ a „*High*“. Výsledkem je rozdíl mezi *Y*-ovými souřadnicemi těchto průsečíků.		Měření rozdílu hodnot charakteristik na začátku a konci měřicího intervalu.
Y_Range	Vyhledá sousední průsečíky křivky se specifikovanými *X*-ovými úrovněmi „*Low*“ a „*High*“. V úseku křivky mezi těmito body najde maximální a minimální *Y*-ovou souřadnici. Výsledkem je jejich rozdíl.		Měření zvlnění kmitočtových charakteristik filtrů.
Slope	Vyhledá průsečík křivky se specifikovanou *X*-ovou úrovní. Najde na křivce dva nejbližší body kolem průsečíku, a proloží jimi přímku. Výsledkem je strmost této přímky.		Měření dif. parametrů v analýze „*DC*“.
Phase Margin	Výsledkem je fázová bezpečnost, odvozená z křivek *dB*( ) a *PHASE* ( ). K dispozici pouze u „*AC“* analýzy.		Měření fázové bezpečnosti v analýze „*AC*“. Musí být k dispozici obě křivky a bod „0 *dB*“ na křivce zesílení.

Tab. 9.3: Vyhodnocovací funkce („*Performance Functions*“) typu „*dva měřicí body*“.

Ukázky okamžitého režimu vyhodnocovací analýzy

Ztrátová přenosová vedení – ukázka funkce *Rise_Time*.

Otevřeme soubor **LTRA3.CIR**, který obsahuje modely tří typů přenosových vedení, buzených společně ze zdroje impulsů podle **obr. 9.75**.



Obr. 9.75: Tři modely přenosových vedení.

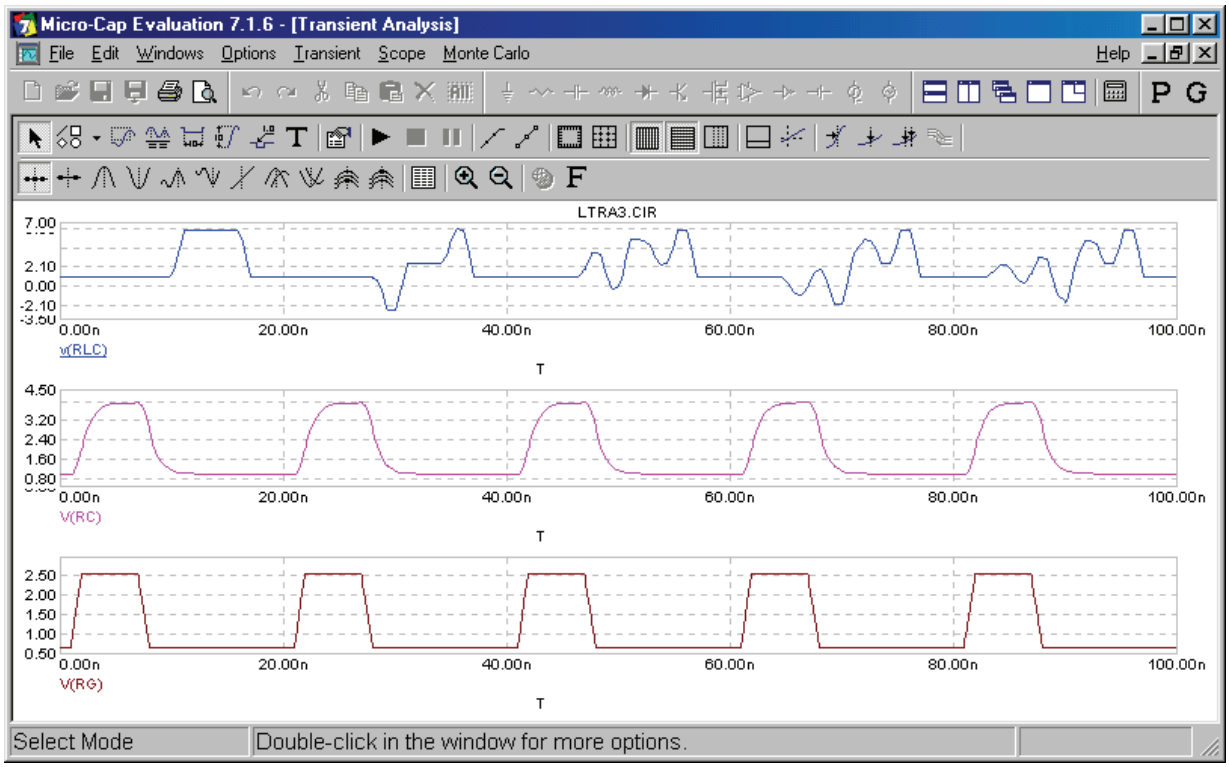
V složce „*Text*“ nalezneme m.j. model zdroje impulsů, z něhož vyplyne, že se jedná o impulsy o úrovních 1 V a 4 V o délkách nástupných a sestupných hran 1 *n*s, šířka impulsů je 5 *n*s a opakovací perioda 20 *n*s:

.MODEL SQUARE PUL (vzero=1 *vone*=4 *P1*=1*n P2*=2*n p3*=7*n P4*=8*N p5*=20*n)*

Spustíme časovou analýzu. Po vykreslení časových průběhů bychom měli dostat výsledky na **obr. 9.76**.

Naším úkolem bude změřit dobu trvání nástupní hrany impulsů *V*(*RC*), tedy výstupního napětí vedení *T*2 na **obr. 9.75**.

Nejprve změříme maximální a minimální velikost impulsů. I když v demoverzi nejsou příslušné vyhodnocovací funkce k dispozici, pomůžeme si vyhledávacími funkcemi, které jsou skryty pod ikonami  (hledání lokálního maxima) a  (hledání lokálního minima). Klikneme na text *V*(*RC*) pod obrázkem č. 2. Tím se křivka *V*(*RC*) stane aktivní pro vyhledávací režim. Pak klikáním na ikonu  přemísťujeme kurzor na jednotlivá lokální maxima impulsů. Podobně klikáním na  vyhledáváme lokální minima. Přesvědčíme se o tom, že velikosti těchto minim a maxim jsou přibližně $U_{min} = 0{,}987$ V a $U_{max} = 3{,}94$ V. Mezivrcholová hodnota je tedy přibližně $U_{mvh} = 2{,}953$ V.

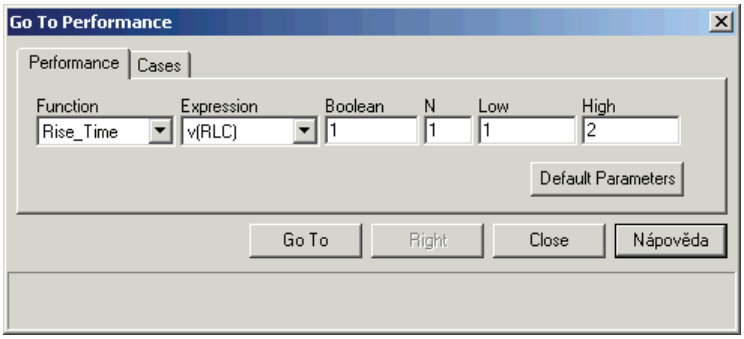


Obr. 9.76: Výsledky časové analýzy obvodu z **obr. 9.75**.

Délku náběžné hrany vyhodnotíme tak, že budeme hledat průsečíky této hrany s úrovněmi

$$U_{min} + 0{,}1\ U_{mvh} = 1{,}282 \text{ V a } U_{min} + 0{,}9\ U_{mvh} = 3{,}645 \text{ V}.$$

Klikneme na ikonu  „*Go To Performance*“. Objeví se okno „*Go To Performance*“ podle **obr. 9.77**.



Obr. 9.77: Okno „*Go To Performance*“ - vstupní brána k vyhodnocovacím funkcím.

Okno obsahuje záložky „*Performance*“ a „*Cases*“. V záložce „*Performance*“ jsou tyto položky:

Function - Nabídka vyhodnocovacích funkcí.

Expression - Nabídka křivek, na jednu z nichž se má aplikovat vyhodnocovací funkce.

Boolean - Logický výraz, který může omezovat rozsah působnosti vyhodnocovací funkce. Standardně je 1 – bez omezení. Typickou omezovací funkcí je například $T > 20n$ (vyhodnocovací funkce se použije pouze pro úsek signálu pro časy delší než 20 *n*s).

N - Pořadí výskytu hledaného stavu, který by vyhovoval zadaným parametrům vyhodnocování (v našem případě hledání vzestupné hrany signálu, která by protínala *Y*-ové úrovně „*Low*“ a „*High*“).

Low, High - Parametry vyhodnocovací funkce. Pro funkci „*Rise_Time“* jsou to *Y*-ové úrovně, jejichž průsečíky s vzestupnou hranou vymezují délku hrany.

Default Parameters:

Po aktivaci tohoto tlačítka program dosadí přednastavené parametry vyhodnocovací funkce, které však většinou nejsou v souladu s našimi záměry analýzy.

V záložce „**Cases**“ je možné vybrat pro vyhodnocování jednu z křivek, vzniklých krokováním parametrů obvodu nebo teploty. Jestliže ke krokování nedochází, je tato složka prázdná.

V nabídce „**Expression**“ vybereme *V*(*RC*) a položky „*Low*“ a „*High*“ vyplníme čísly 1.282 a 3.645. Klikneme na „*GoTo*“. Kurzory se uchytí na první nalezenou náběžnou hranu a vyznačí se jejich souřadnice. V okně „*Go To Performance*“ se objeví číselná hodnota délky náběžné hrany 2,04282 *n*s. V okénku „*N*“ se zvětšil údaj na dvojku – program je připraven hledat 2. náběžnou hranu. Po kliknutí na „*Go To*“ ji nalezne. Vyzkoušejte si, že po nalezení poslední náběžné hrany se hledání opakuje od začátku.

Další doporučené příklady

Jestliže máte možnost pracovat s profesionální verzí MicroCapu, která umožňuje použití všech vyhodnocovacích funkcí, doporučujeme tyto příklady:

COLPITTS.CIR, časová analýza:

Vyzkoušejte měření maxim (*Peak_X*, *Peak_Y*), minim (*Valley_X*, *Valley_Y*), mezivrcholové hodnoty (*Peak_Valley*), opakovacího kmitočtu (*Frequency*) výstupního napětí.

UA709.CIR, kmitočtová analýza:

Změřte kmitočet třídecibelového poklesu zesílení (*X_Level*), ověřte, že strmost poklesu zesílení je asi 20 dB/dekádu, resp. 6 dB/oktávu (*X_Delta*), změřte fázovou bezpečnost (*Phase_margin*, je třeba zobrazit i fázovou kmitočtovou charakteristiku).

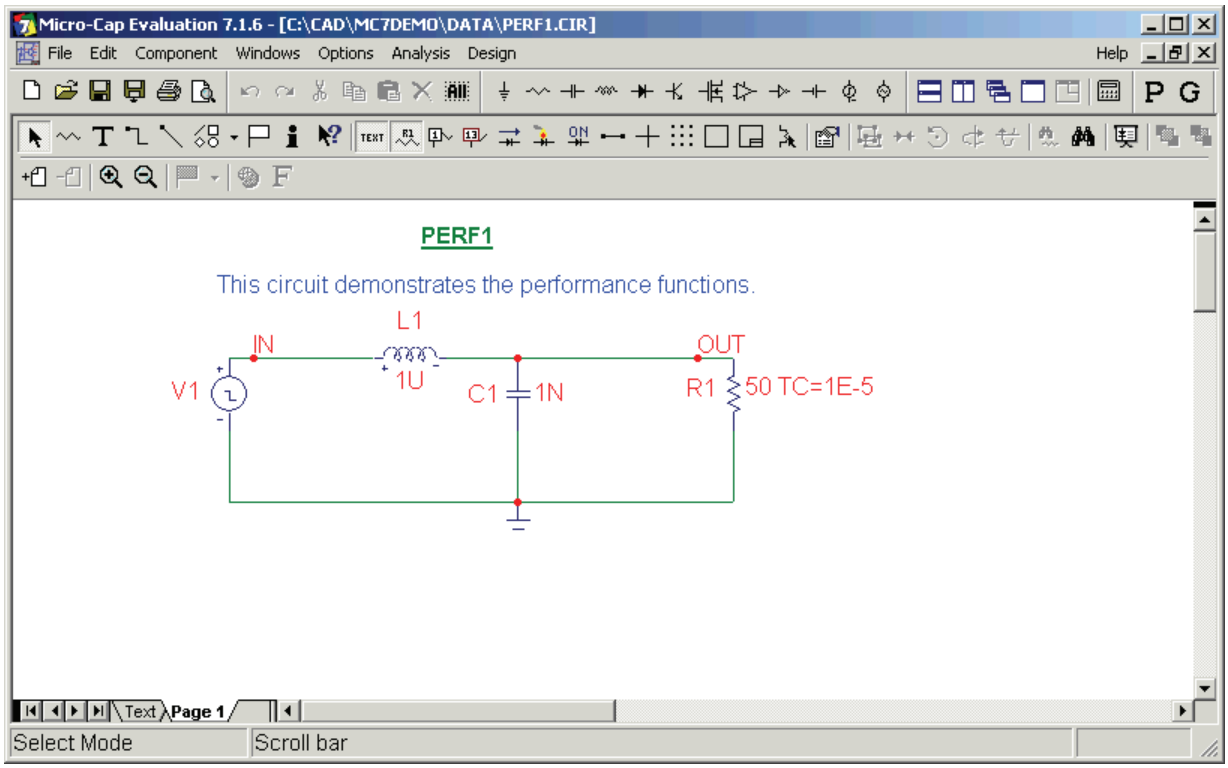
Vyhodnocovací grafy

Vyhodnocovací grafy objasníme na příkladu článku *RLC*, jehož model je v souboru **PERF1.CIR** (viz **obr. 9.78**).

V složce „*Text*“ je mimo jiné tento model zdroje impulsů:

.MODEL PULSE PUL (VZERO=0 *VONE*=5 *P*1=100*N P2*=100*N P*3=500*N P*4=500*N P5*=1*U)*

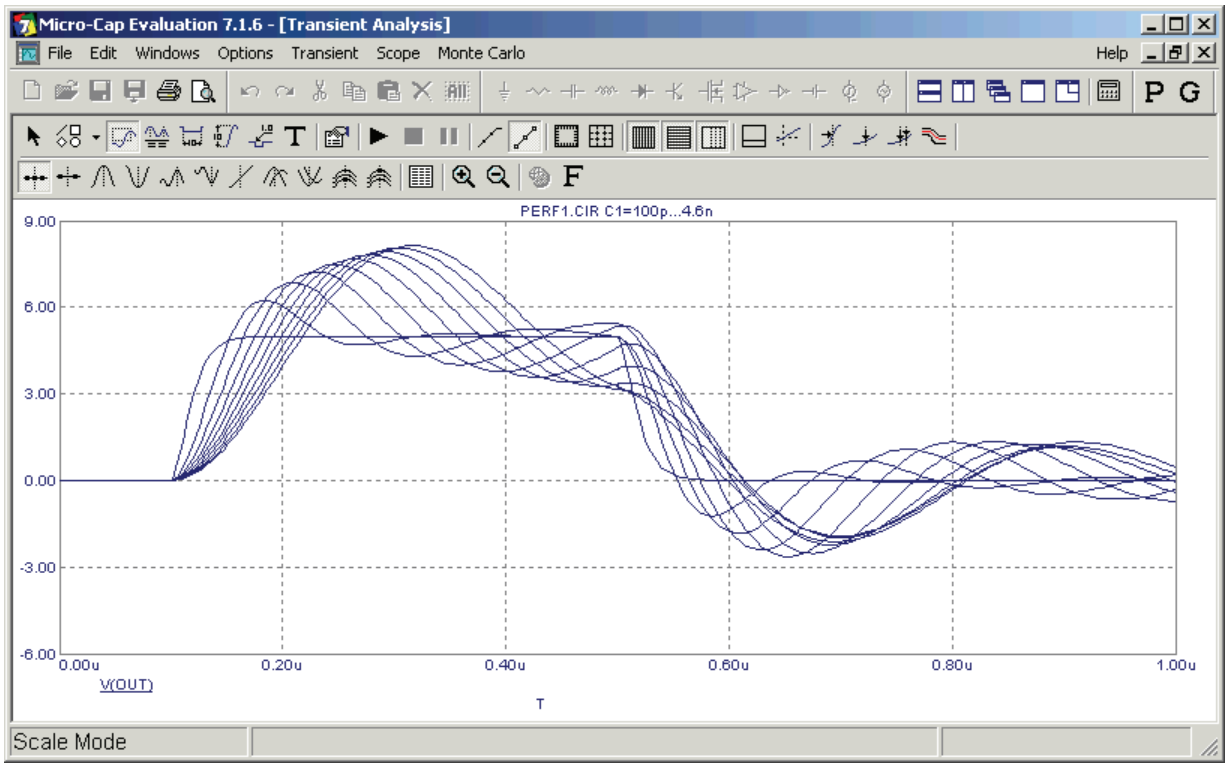
Vstupní signál je tedy tvořen obdélníkovými impulsy o úrovních 0 V a 5 V o šířce 400 *n*s a opakovací periodě 1 μs. Jedná se o filtr typu dolní propust 2. řádu. Úkolem simulace bude získání závislosti délky náběžné hrany výstupního impulsu na kapacitě *C*1.



Obr. 9.78: Pasivní obvod *RLC* k demonstraci vyhodnocovacích grafů.

Vyvoláme analýzu „*Transient*“. V okně „*Transient Analysis Limits*“ se přesvědčíme, že analýza proběhne v časovém intervalu od 0 do 1 µs. Analyzovanou funkcí bude *v*(*out*). Klikneme do tlačítka „*Stepping*“ a přesvědčíme se, že při analýze bude krokována kapacita *C*1 od 0,1 *n*F do 4,6 *n*F po 0,5 *n*F. Rovněž jsou zadány údaje pro krokování *R*1, ale toto krokování není povoleno.

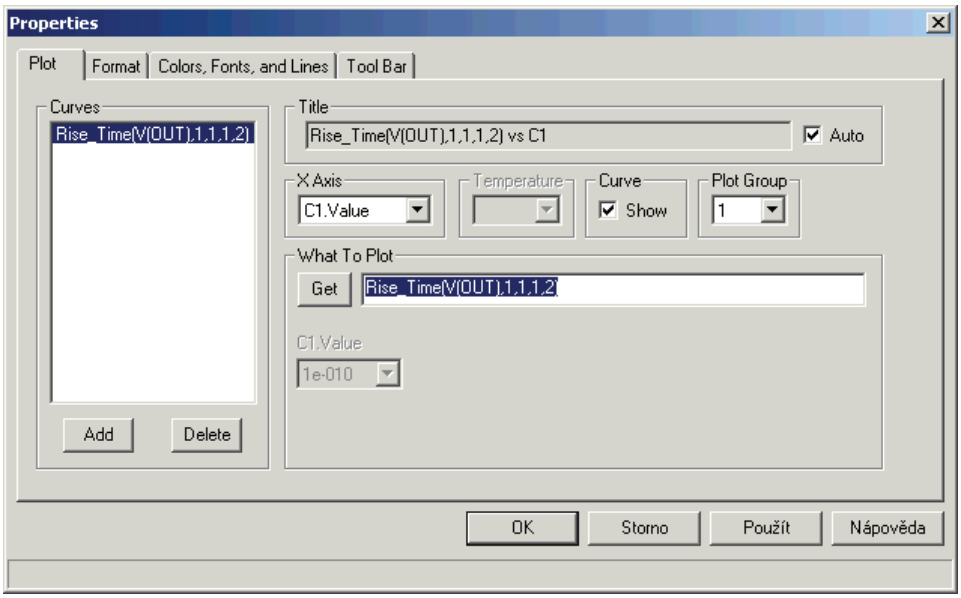
Horkou klávesou *F*2 spustíme analýzu. Výsledek je znázorněn na **obr. 9.79**.



Obr. 9.79: Výsledky časové analýzy obvodu z **obr. 9.78** při krokování kapacity *C*1.

V paměti jsou uloženy vypočtené body všech křivek pro krokovaný parametr *C*1. Nyní bychom mohli „ruční“ vyhodnocovací analýzou určovat doby náběhu pro jednotlivé křivky a vynášet do grafu jejich závislosti na kapacitě *C*1. Program to provede za nás.

Zvolíme „*Transient/Performance Window/Add Performance Window*”. Objeví se okno “*Properties*” – viz **obr. 9.80**.



Obr. 9.80: Okno “*Properties*” k specifikaci vyhodnocovacích grafů.

V okénku “*X Axis*” je přednastavena proměnná, která se bude vynášet na vodorovnou osu, tedy hodnota kapacity *C*1.

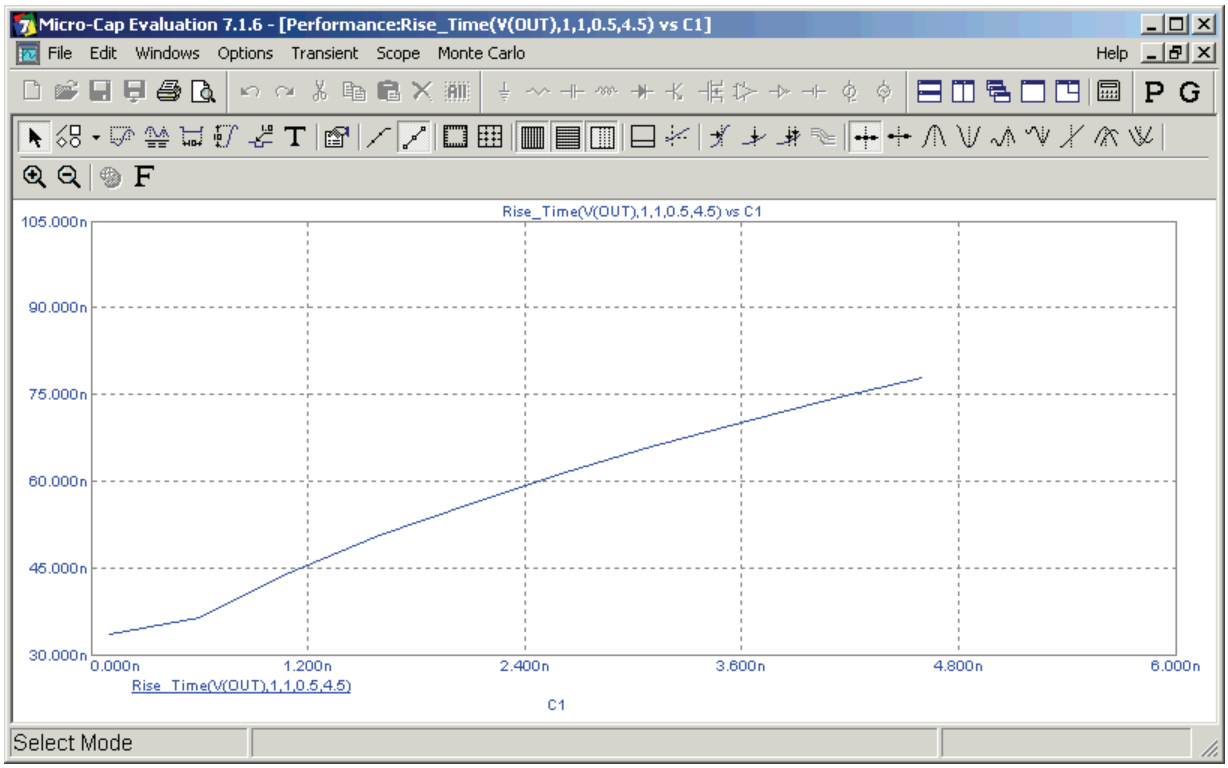
V okénku “*What To Plot*” je přednastavena vyhodnocovací funkce “*Rise_Time”* s parametry:

- Křivka, která se bude vyhodnocovat: *V*(*OUT*)
- *Boolean*: 1
- *N*: 1
- *Low*: 1
- *High*: 2.

Vzhledem k tomu, že ustálené úrovně impulsů by měly být 0 V a 5 V, změníme parametry “*Low*” a “*High*” na 0,5 a 4,5. Změnu provedeme kliknutím na položku “*Get*”. Rozbalí se okno “*Get Performance Function*”, což je jen jinak nazvané okno “*Go To Performance*” z **obr. 9.77**. Položky “*Low*” a “*High*” přepíšeme na 0.5 a 4.5 a potvrdíme “*OK*”. Automaticky se vrátíme do okna “*Properties*”. Zkontrolujeme, že údaj v položce “*What To Plot*” se změnil. Potvrdíme “*OK*”. Obdržíme graf na **obr. 9.81**.

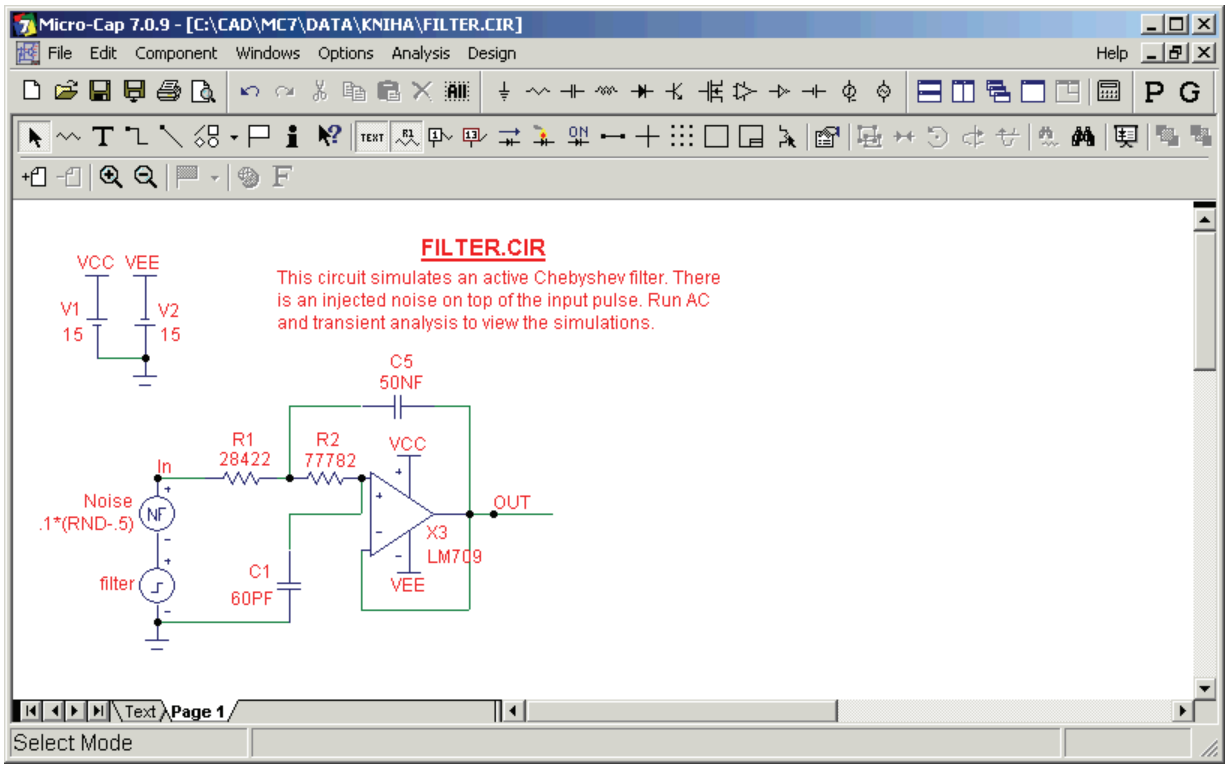
Uvědomme si, že kapacita *C*1 byla krokována v 10 hodnotách, čemuž nyní odpovídá počet bodů křivky. Pokud bychom požadovali její jemnější vykreslení, museli bychom celou analýzu zopakovat s jemnějším krokem.

Vyhodnocovací funkce můžeme různým způsobem kombinovat za účelem dosažení potřebných efektů. Následuje ukázka, k jejímuž ověření je potřebná profesionální verze MicroCapu.



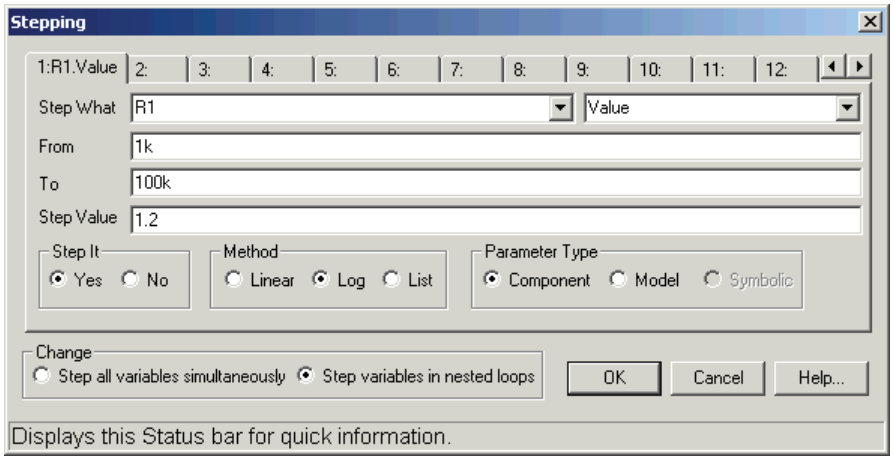
Obr. 9.81: Výsledný vyhodnocovací graf – závislost délky náběhu signálu na kapacitě *C*1.

V souboru **FILTER.CIR**, který je součástí instalace, je model filtru "*Sallen-Key*" typu dolní propust (viz **obr. 9.82**). Bude nás zajímat, jak závisí amplitudová kmitočtová charakteristika filtru na odporu *R*1, konkrétně závislost kmitočtu maxima charakteristiky, velikosti tohoto maxima a třídecibelové šířky pásma.



Obr. 9.82: Aktivní filtr 2. řádu typu "*Sallen-Key*".

Spustíme kmitočtovou analýzu. Ponecháme přednastavení okna "*AC Analysis Limits*" s jednou výjimkou – smažeme číslo obrázku 2 v řádku, kde je definován výpočet fázové kmitočtové charakteristiky. Poté klikneme na "*Stepping*" a nastavíme krokování odporu *R*1 podle **obr. 9.83**. Potvrdíme "*OK*" a spustíme analýzu horkou klávesou *F*2. Objeví se výsledky podle **obr. 9.84** – soustava kmitočtových charakteristik.



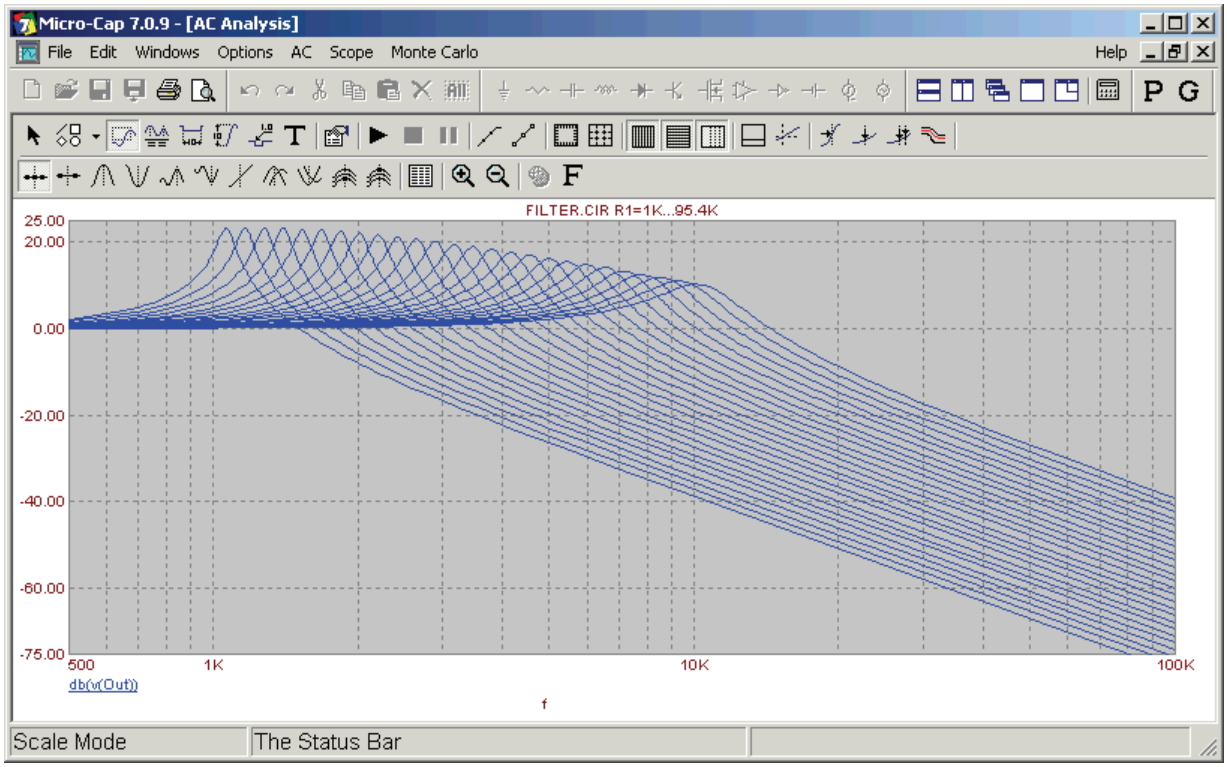
Obr. 9.83: Způsob nastavení podmínek krokování.

Vyhodnocovací analýzou lze nyní zkoumat závislosti souřadnic maxim kmitočtových charakteristik na odporu *R*1, a tím možnosti přelaďování filtru odporem *R*1. Aktivujeme vyhodnocovací analýzu ("*AC/Performance Windows/Add Performance Window*"). Pro zjišťování závislosti kmitočtu maxima bychom vybrali funkci

$$Peak\_X(db(V(out)),1,1)$$

a pro velikost maxima

$$Peak\_Y(db(V(out)),1,1)$$

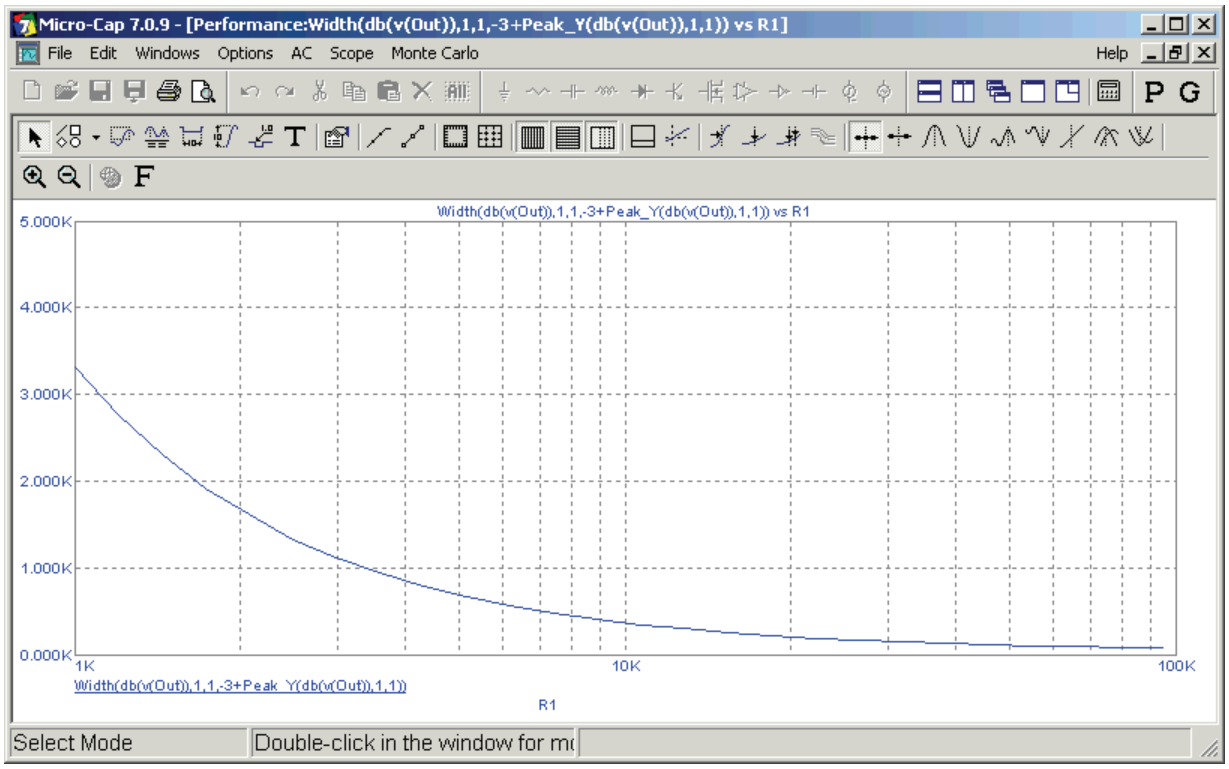


Obr. 9.84: Síť kmitočtových charakteristik filtru z **obr. 9.82** pro různé hodnoty odporu *R*1.

Trochu složitější bude vyhodnocování třídecibelové šířky pásma. Mohli bychom použít funkci *WIDTH* (viz **Tab. 9.3**), ale situaci komplikuje fakt, že velikost maxima a tudíž i parametr "*Y-Level*" nejsou při krokování konstantní. Řešením je složená vyhodnocovací funkce:

*Width(db(v(Out)),*1,1,-3+*Peak_Y(db(v(Out)),*1,1*))*

Posledním členem na pravé straně je právě *Y*-úroveň, jejíž průsečíky s křivkou definují šířku pásma. Tato úroveň je počítána jako maximální hodnota křivky (funkcí "*Peak"*) mínus 3 decibely. Výsledný graf pro šířku pásma je na **obr. 9.85**.



Obr. 9.85: Výsledný vyhodnocovací graf – závislost třídecibelové šířky pásma na odporu *R*1.

S dalším typem vyhodnocovacích grafů – tzv. histogramy, se seznámíme v následující části, která se bude týkat statistické analýzy obvodů.

9.7.4 Statistická analýza ("*Monte Carlo*")

Cíle statistické analýzy

Hlavním cílem statistické analýzy je zjistit, nakolik výrobní rozptyly parametrů jednotlivých součástek ovlivňují vlastnosti obvodu. Jinými slovy, jak může nedodržení jmenovitých hodnot jednotlivých parametrů odklonit výsledné charakteristiky obvodu od charakteristik požadovaných.

Praktická aplikace statistické analýzy znamená (viz **Obr. 9.86**):

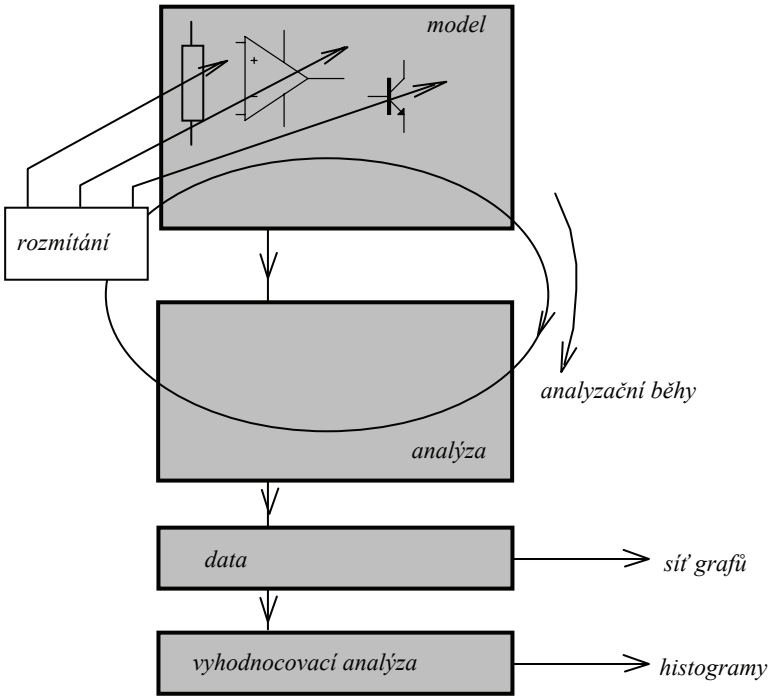
1. Výběr parametrů součástek, které budou vykazovat tolerance. Definice velikostí těchto tolerancí a statistického zákona jejich výskytu.

2. Provedení mnohonásobné analýzy obvodu. Při každém analyzačním běhu jsou za hodnoty parametrů, které vykazují tolerance, dosazena náhodná čísla, která závisí na jmenovitých hodnotách parametrů a na statistickém rozložení příslušných tolerancí kolem těchto jmenovitých hodnot. Výsledkem není jedna, ale celá síť analyzovaných závislostí.
3. Vyhodnocení výsledků mnohonásobné analýzy. Vyhodnocení může být buď vizuální (zda síť charakteristik není příliš "široká", nebo statistické (statistická analýza velkého množství dat s využitím vyhodnocovacích funkcí; výsledkem jsou histogramy a číselné statistické charakteristiky jako střední hodnota, rozptyl apod.).

Jak již bylo jednou zmíněno, statistickou analýzu lze uskutečnit nad analýzami "*Transient*", "*AC*" i "*DC*". Není však současně možné používat režim krokování ("*Stepping*").

Statistické zákonitosti rozložení parametrů součástek

V této kapitole ukážeme, jak různě je možno chápat například konstatování, že odpor *R* má jmenovitou hodnotu 1 *k*Ω a toleranci ±10%. Závisí na výrobních postupech, s jakou četností se budou v dané sérii vyskytovat jednotlivé hodnoty v tolerančním pásu kolem jmenovité hodnoty, neboli jaké je statistické rozložení těchto hodnot.

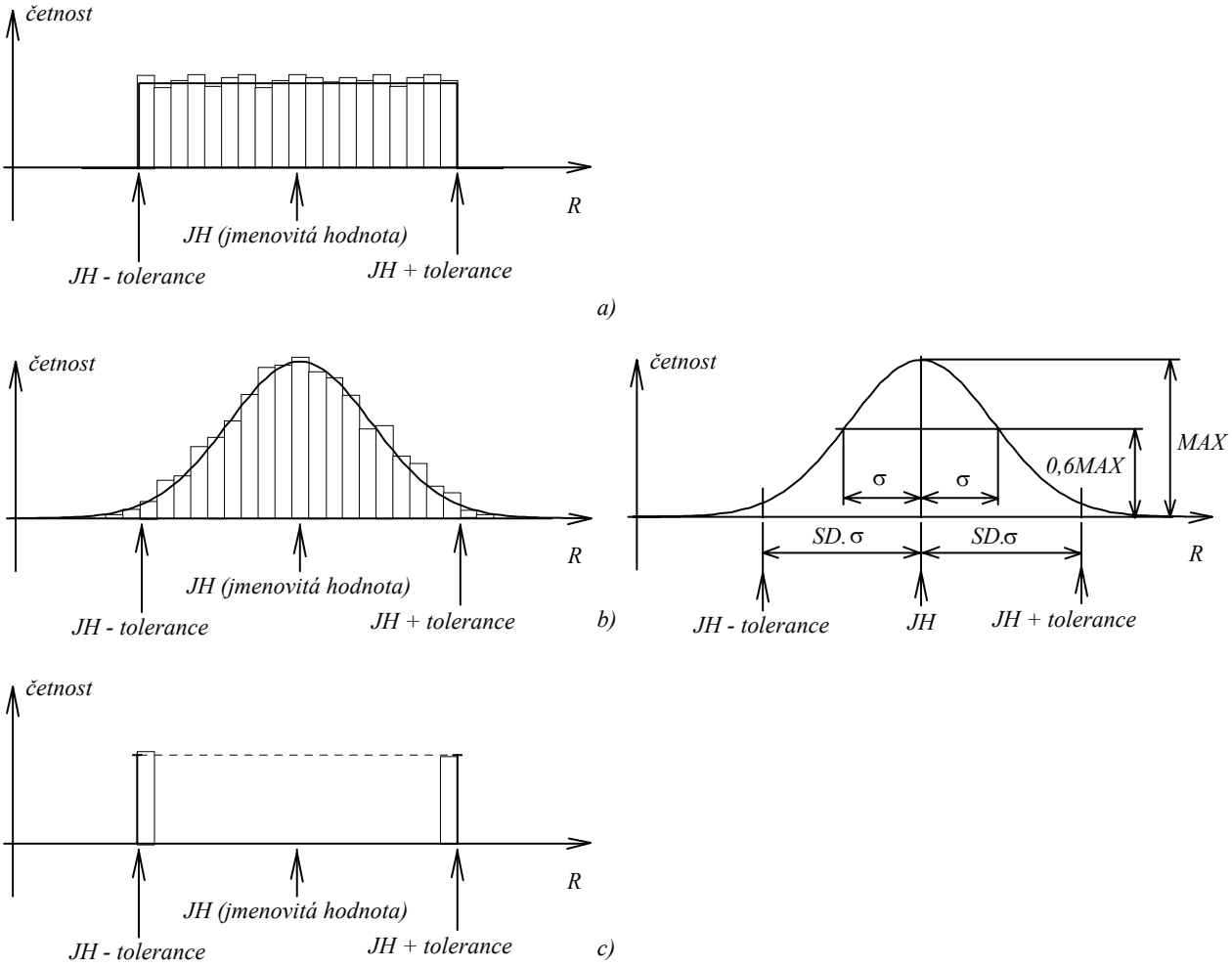


Obr. 9.86: Mechanismus statistické analýzy.

Toto statistické rozložení lze experimentálně zjistit následovně. Uvažujme opět rezistor o jmenovité hodnotě 1 *k*Ω a toleranci ±10%. Předpokládejme, že budeme mít k dispozici značné množství vyrobených rezistorů. Jejich měřením získáme množství hodnot, které budou vykazovat rozptyl kolem jmenovité hodnoty. Tyto údaje zobrazíme ve formě tzv. histogramu: vodorovnou osu, na kterou budeme vynášet velikosti odporů, rovnoměrně rozdělíme v okolí jmenovité hodnoty na dostatečný počet úseků, například po 10 ohmech. Nad každým úsekem nakreslíme obdélník, jehož výška bude odpovídat počtu rezistorů, jejichž odpor spadá do daného intervalu. Dělení vodorovné osy bychom měli provádět dostatečně jemně, ale na

druhou stranu tak, aby do každého intervalu odporů spadal statisticky významný počet změřených hodnot. Pak při růstu celkového počtu měřených rezistorů a příslušném zkracování měřicích intervalů se začne projevovat statistická zákonitost v rozložení měřeného parametru a obálka histogramu se začne blížit k určité křivce statistického rozložení.

Tři různé typické výsledky jsou na **obr. 9.87**.



Obr. 9.87: Pravděpodobnostní rozložení náhodných veličin: a) rovnoměrné, b) normální (Gaussovo), c) nejhorší případ ("Worst Case").

Na obr. a) je rovnoměrné rozdělení, které modeluje situaci, kdy v specifikovaném tolerančním pásu se všechny hodnoty náhodného parametru vyskytují se stejnou četností. Současně je nulová pravděpodobnost výskytu jakýchkoliv hodnot mimo toleranční pás.

Rovnoměrné rozdělení většinou modeluje skutečné rozdělení hodnot parametrů jen přibližně. Nerespektuje tyto skutečnosti: 1. Hodnoty v těsném okolí jmenovité hodnoty mívají četnost výskytu větší než hodnoty vzdálenější. 2. Nelze zcela vyloučit výskyt hodnot mimo toleranční pole.

V řadě případů tedy skutečnosti více odpovídá tzv. normální neboli Gaussovo rozložení na obr. b). Z matematického vyjádření Gaussovy křivky vyplývá geometrická konstrukce, naznačená na obrázku, z které vyplývá důležitá statistická veličina – rozptyl (nebo též směrodatná odchylka) σ kolem jmenovité hodnoty. Čím menší bude rozptyl, tím "štíhlejší"

bude Gaussova křivka a tím menší bude procento rezistorů, vyskytujících se mimo toleranční pás. Na obr. b) je naznačen vztah mezi rozptylem a tolerancí:

$$tolerance = SD.\sigma .$$

Parametr *SD* tedy představuje počet směrodatných odchylek v specifikované toleranci. Tento parametr je standardně nastaven na hodnotu 2,58 v globálních podmínkách simulace. Uživatel jej může měnit a tím pozměňovat procento výskytu všech odporů v tolerančním pásmu. Zvětšováním *SD* zvětšujeme jejich počet až k maximu 100% podle následující tabulky:

SD	*procento výskytů v tolerančním pásu*
1	68
1,96	95
2	95,5
2,58	99
3	99,7
3,29	99,9

Jak je vidět, přednastavená hodnota 2,58 znamená, že v průměru 1% součástek z velké série nebude vyhovovat specifikované toleranci, což je v praxi obvyklý případ. Pokud výrobce zaručuje, že v toleranci bude jen 95% součástek, nastavíme parametr *SD* na hodnotu 1,96.

Jestliže rovnoměrné rozdělení je příliš jednoduchý a Gaussovo rozdělení více realistický model skutečného rozdělení, pak rozložení na obr. c), označované anglickým termínem „*Worst Case*“ – nejhorší případ, lze charakterizovat jako model, používaný „opatrným pesimistou“. U tohoto rozdělení se předpokládají pouze hodnoty parametrů na krajích tolerančního pásu s rovnoměrnou četností výskytu na obě strany.

Způsob rozmítání parametrů součástek

Při statistické analýze dochází při opakované analýze obvodu k rozmítání těch parametrů, u nichž je definována tolerance. Při prvním analyzačním běhu jsou za parametry dosazeny jejich jmenovité hodnoty. V dalších bězích je prováděno rozmítání parametrů, které obstarávají generátory náhodných čísel, jejichž běh je řízen v závislosti na definovaném statistickém rozložení. To například u Gaussova rozložení znamená, že v průměru nejčastěji se budou objevovat hodnoty v okolí jmenovité hodnoty, v souladu s tvarem Gaussovy křivky.

Způsob rozmítání parametrů součástek metodou Worst Case vede při analýze k specifickým výsledkům. Uvažujme například *RC* článek typu dolní propust o hodnotách odporu a kapacity

$$R = 10\ k\Omega,\ C = 1\ n\mathrm{F}.$$

U obou parametrů jsou zadány tolerance 10%. Znamená to, že při statistické analýze budou za *R* a *C* dosazovány hodnoty

$$R_{min} = 9\ k\Omega,\ R_{max} = 11\ k\Omega,$$
$$C_{min} = 900\ p\mathrm{F},\ C_{max} = 1{,}1\ n\mathrm{F}.$$

Běžná je stochastická metoda *Worst Case*, kdy jsou parametry přepínány mezi hodnotami „*min*“ a „*max*“ náhodně, ale v průměru se stejnou četností. Některé simulátory,

např. TINA, umožňují analytickou metodu *Worst Case*, kdy se vystřídají všechny existující kombinace minimálních a maximálních hodnot parametrů. Při celkovém počtu N parametrů se zadanými tolerancemi je těchto kombinací 2^N. Při relativně malém N je výhodnější analytická metoda. Avšak například již při 10 parametrech existuje celkem 1024 kombinací a tudíž 1024 analyzačních běhů analytické metody. Pak je výhodnější použít stochastickou metodu s menším počtem analyzačních běhů. Některé kombinace sice při analýze nenastanou, ale to většinou není na závadu. Uvážíme-li například, že součin parametrů R a C tvoří časovou konstantu, na níž závisí dejme tomu průběh kmitočtové charakteristiky obvodu, pak kombinace (R_{min}, C_{max}) a (R_{max}, C_{min}) poskytnou prakticky stejné výsledky analýzy. Výsledkem statistické analýzy budou „hraniční" křivky, odpovídající kombinacím (R_{min}, C_{min}) a (R_{max}, C_{max}). Budou tvořit jakýsi „obal" všech možných křivek, které bychom získali statistickou analýzou při uvažování jiných zákonů rozdělení.

Program MicroCap umožňuje pouze stochastickou metodu *Worst Case*.

Způsob rozmítání parametrů můžeme dále ovlivnit využíváním rozšířené syntaxe příkazů *LOT* a *DEV*. O tom bude zmínka v závěru kapitoly.

Zadávání velikostí a statistického charakteru tolerancí

Tolerance lze zadávat dvojím způsobem: buď jako tolerance parametrů modelů v příkazu *.MODEL*, nebo jako tolerance symbolických proměnných v příkazu *.DEFINE*.

Statistický charakter tolerancí lze zvolit buď „centrálně" a jednotně pro všechny parametry v menu statistické analýzy, nebo u každého parametru zvlášť v příkazech *.MODEL* nebo *.DEFINE*. Druhá metoda má prioritu před první.

V dalším ukážeme, že základní podmínky statistické analýzy lze velmi jednoduše definovat příkazem *LOT*. Simulační program však kromě toho poskytuje řadu nástrojů, kterými můžeme běh statistické analýzy modifikovat, například s přihlédnutím k statistickým závislostem mezi různými parametry jedné součástky nebo mezi parametry různých součástek. K těmto nástrojům patří příkaz *DEV* a rozšířená syntaxe příkazů *LOT* a *DEV*.

Zjednodušené používání příkazu *LOT* jako součást příkazu *.MODEL*

Jako příklad uvažujme operační zesilovač typu *LF*155. Chceme definovat desetiprocentní toleranci jeho tranzitního kmitočtu, který je v modelu označován symbolem *GBW* („*Gain BandWidth"*).

Po umístění schématické značky na plochu se otevře okno atributů „*OpAmp*". Vybereme model *LF*155 a přesvědčíme se o tom, že hodnota *GBW* je 2,5 MHz: v poli „*GBW*" je údaj 2.5MEG. Obsah pole doplníme příkazem *LOT* takto:

2.5*MEG LOT*=10*%*

a zadávání potvrdíme (*OK*). Podíváme-li se nyní do složky "*Text*", objeví se tam příkaz *.MODEL*:

```
.MODEL LF155 OPA (LEVEL=3 TYPE=3 ROUTAC=50 ROUTDC=75 VOFF=2M
+ IOFF=3P SRP=7MEG  SRN=7MEG IBIAS=30P VPS=12.4 VNS=-12.4 GBW=2.5MEG
+ LOT=10%)
```

Můžeme zobecnit:

Parametru modelu přiřadíme toleranci tak, že za jeho definici umístíme text

LOT=x%

kde *x* je číselná hodnota požadované tolerance v procentech.

Poznámka: toleranci nemusíme zadávat v procentech. Například text

LOT=250k

bude ve výše uvedeném případě ekvivalentní textu

LOT=10%

Tolerance pasivních součástek typu *R*, *L* a *C* definujeme tak, že jim nejprve přidělíme model a v něm pak zavedeme tolerance násobícího součinitele *R*, *L*, resp. *C* (viz příloha P10.2). Tak například model rezistoru o toleranci odporu 5% by mohl vypadat takto:

.MODEL ODPOR RES (R=1 *LOT*=5*%)*

Každý rezistor, kterému přiřadíme model „*ODPOR*“, bude mít definovanou toleranci jmenovité hodnoty 5%. Upozorňujeme, že velikost jmenovité hodnoty není v modelu definována a že symbol *R* značí násobící součinitel, který je standardně jednotkový a kterým se jmenovitá hodnota násobí. Tímto způsobem se tedy tolerance jmenovité hodnoty definuje nepřímo přes toleranci násobícího součinitele.

Dále je vhodné zdůraznit, že v daném analyzačním běhu budou mít rezistory se společným modelem „*ODPOR*“ různé hodnoty odporů, jinými slovy, že rozmítání těchto odporů bude řízeno z nezávislých generátorů náhodných čísel. Vše je schématicky ilustrováno na **obr. 9.88** a). Předpokládejme pro jednoduchost, že jmenovité hodnoty všech tří odporů *R*1, *R*2 a *R*3 jsou stejné, např. 1 *k*Ω, a že statistický zákon rozdělení tolerancí je „*Worst Case“*. Pak v každém analyzačním běhu budou mít odpory přiřazeny náhodné hodnoty z množiny 950 Ω a 1050 Ω.

Zjednodušené používání příkazu *LOT* jako součást příkazu *.DEFINE*

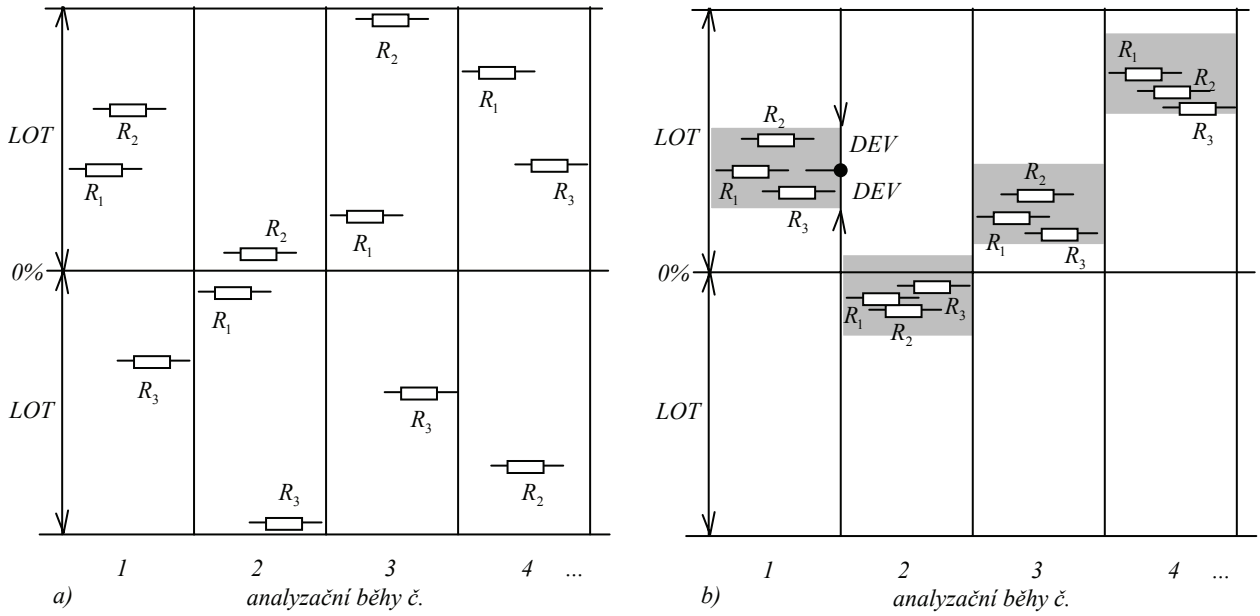
Při definování symbolické proměnné je možné uvést její toleranci příkazem *LOT* v složených závorkách **před** touto proměnnou. Například označíme odpor rezistoru symbolicky jako *Rx* a zapíšeme tento „*grid text*“:

.DEFINE {LOT=10*%} Rx* 1*k*

Rozdíl oproti definování tolerance příkazem *.MODEL* je v tom, že jestliže označíme symbolickou proměnnou *Rx* více odporů, budou mít všechny tyto odpory v daném analyzačním běhu stejnou hodnotu.

Zjednodušené používání příkazu *DEV* jako součást příkazu *.MODEL*

Příkazem *DEV* se definuje tzv. relativní tolerance. Vysvětlení je naznačeno na **obr. 9.88** b).



Obr. 9.88: Porovnání mechanismů fungování příkazů *LOT* (a) a *DEV* (b).

Uvažujme příklad následujícího modelu rezistoru:

.MODEL ODPOR RES (R=1 *LOT*=5% *DEV*=1*%)*

Představme si, že tohoto modelu využívají tři rezistory *R*1, *R*2 a *R*3. Při každém simulačním běhu se nyní nastavují hodnoty *R*1, *R*2 a *R*3 nadvakrát: V prvním kroku se všechny nastaví na hodnotu, odkloněnou od jmenovité hodnoty o **stejnou** toleranci, náhodně vybranou příkazem *LOT*. V druhém kroku se tato tolerance u každého odporu doplní zvlášť o přídavnou náhodnou odchylku vybranou příkazem *DEV*.

Předpokládejme pro jednoduchost, že jmenovité hodnoty všech odporů *R*1, *R*2 a *R*3 jsou stejné, např. 1 *k*Ω, a že statistický zákon rozdělení tolerancí je „*Worst Case*“. Pak v prvním kroku se všechny odpory nastaví na stejnou náhodnou hodnotu z množiny 950 Ω a 1050 Ω (podstatný rozdíl oproti použití samotného příkazu *LOT* bez příkazu *DEV*). Dejme tomu, že „padne“ hodnota 1050 Ω. V druhém kroku se tato hodnota modifikuje u každého odporu zvlášť o náhodnou odchylku 1 % z jmenovité hodnoty, tj. o +10 Ω nebo -10 Ω (jde o rozložení „*Worst Case*“). Předpokládejme, že u *R*1 a *R*2 tato odchylka bude kladná a u *R*3 záporná. Výsledkem bude tato sada hodnot v daném analyzačním běhu:

$$R1 = 1060\ \Omega,\ R2 = 1060\ \Omega,\ R3 = 1040\ \Omega.$$

Kombinací příkazů *LOT* a *DEV* tedy můžeme modelovat situaci, kdy sada některých parametrů součástek vykazuje určité tolerance, ale parametry navzájem se liší maximálně o definovanou toleranci. Tak je tomu například u kapacit realizovaných *CMOS* technologií: hodnoty dílčích kapacit mohou vykazovat relativně velké tolerance 5-10% (*LOT*), kdežto jejich vzájemné poměry se daří udržovat v toleranci zlomků procent (*DEV*).

Poznámka 1: Neuvedeme-li příkaz *LOT*, je to to samé jako *LOT* = 0. Pak funkci tohoto příkazu přebírá vlastně příkaz *DEV*.

Poznámka 2: Uvedeme-li současně s příkazem *LOT* i příkaz *DEV* = 0, budou mít všechny odpory z předchozího výkladu v daném analyzačním běhu stejnou (i když náhodnou) hodnotu (stoprocentní souběh).

Poznámka 3: Příkaz *DEV* by neměl v kombinaci s příkazem *.DEFINE* smysl, proto je jeho použití omezeno na příkaz *.MODEL*.

Zobecněné použití příkazů *LOT* a *DEV*

Příkaz *LOT* má obecně tuto strukturu:

LOT[/*číslo_generátoru*[/*rozdělení*]]=*tolerance*[%]

Údaje v hranatých závorkách jsou nepovinné. Stejnou strukturu má i příkaz *DEV*.

Simulátor má pro zvláštní použití funkce *LOT* k dispozici 10 nezávislých generátorů náhodných čísel, označovaných čísly 0 až 9. Dalších 10 je určeno pro funkci *DEV*. Přejeme-li si například, aby v daném analyzačním běhu byly náhodné hodnoty různých parametrů měněny synchronně, budeme tyto parametry rozmítat ze stejného generátoru. Například synchronní změny odporů *RE* a *RC* v modelu tranzistoru zajistíme příkazem:

.MODEL 2*N*1711 *NPN (RE*=2 *LOT/*1=10*% RC*=2 *LOT/1*=10*%)*

Pokud bychom v příkazu nespecifikovali čísla generátorů nebo bychom zvolili čísla různá, odpory *RC* a *RE* by v každém analyzačním běhu nabývaly různě velkých statisticky nezávislých hodnot.

Dále je možné klíčem „*/rozdělení*“ specifikovat pravděpodobnostní rozložení individuálně pro každý rozmítaný parametr, a dokonce zvlášť pro tolerance *LOT* a *DEV*. Takováto specifikace pak má prioritu před centrálně nastaveným typem rozdělení. Pro tyto účely existují tři klíčová slova:

GAUSS	Gaussovo rozdělení
UNIFORM	rovnoměrné rozdělení
WCASE	Worst Case

Příkazem

.MODEL ODPOR RES (R=1 *LOT/WCASE*=5*% DEV/GAUSS*=1*%)*

definujeme odpor s absolutní tolerancí 5 % s rozložením „*Worst Case*“ a s relativní tolerancí 1 % s Gaussovým rozložením, nezávisle na „centrálním“ nastavení typu statistického rozložení.

Zobecněný příkaz *LOT* můžeme použít i při definování tolerancí symbolických proměnných. Například grid text

*.DEFINE {LOT/*3*/WCASE=*2*%} Rx* 10*k*

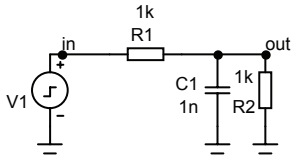
definuje symbolickou proměnnou *Rx* o jmenovité hodnotě 10*k*, toleranci 2%, rozmítání odporu je řízeno generátorem č. 3, statistické rozložení „*Worst Case“*.

Ilustrační příklad statistické analýzy

Jednotlivé fáze statistické analýzy ukážeme na příkladu, jehož zadání nalezneme v dodatkovém souboru **RANDOM.CIR**. Jedná se o jednoduchý *RC* filtr na **obr. 9.89**. Rezistory *R*1 a *R*2 mají přiřazený model s názvem *ODPOR*1. Příslušný příkaz *.MODEL* nalezneme v složce „*Text*“:

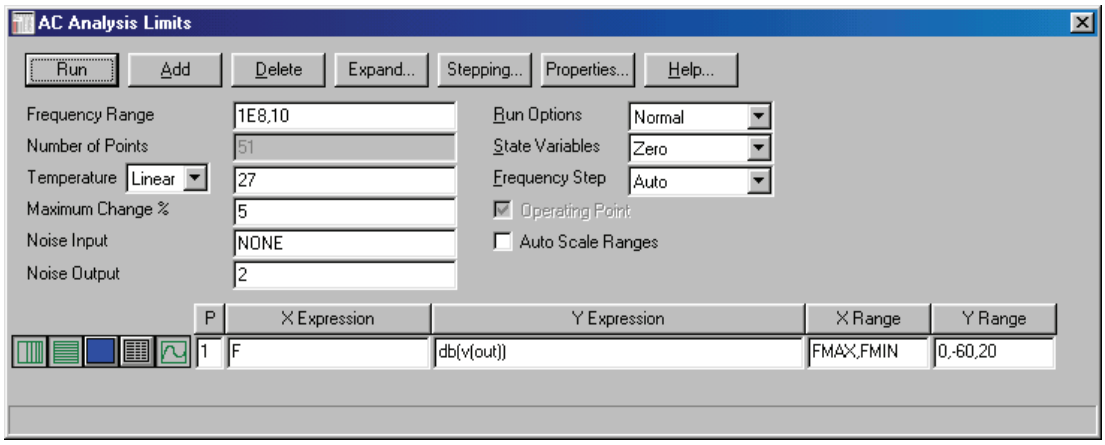
*.MODEL ODPOR*1 *RES* (*R*=1 *LOT*=10%)

Tento model přiřazuje oběma odporům toleranci 10%. Při variaci *R*1 a *R*2 můžeme očekávat změny stejnosměrného přenosu obvodu a změny ve velikosti časové konstanty a lomového kmitočtu filtru.



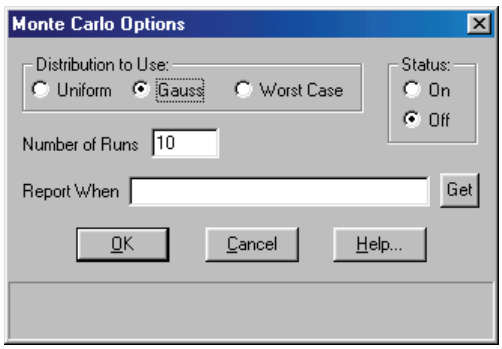
Obr. 9.89: Jednoduchý *RC* článek pro demonstraci statistické analýzy.

Aktivujeme „*AC*“ analýzu. Okno „*AC Analysis Limits*“ by mělo být vyplněno tak, jak vidíme na **obr. 9.90**.



Obr. 9.90: Nastavení podmínek „*AC*“ analýzy obvodu z **obr. 9.89**.

Na pozadí tohoto okna je hlavní okno MicroCapu. V tomto okně vyhledáme na liště položku „*Monte Carlo*“ a klikneme na ni. Rozbalí se nabídka se zatím jedinou aktivní položkou „*Options*“. Aktivujeme ji. Objeví se okno „*Monte Carlo Options*“ podle **obr. 9.91**.



Obr. 9.91: Okno pro zadávání podmínek statistické analýzy.

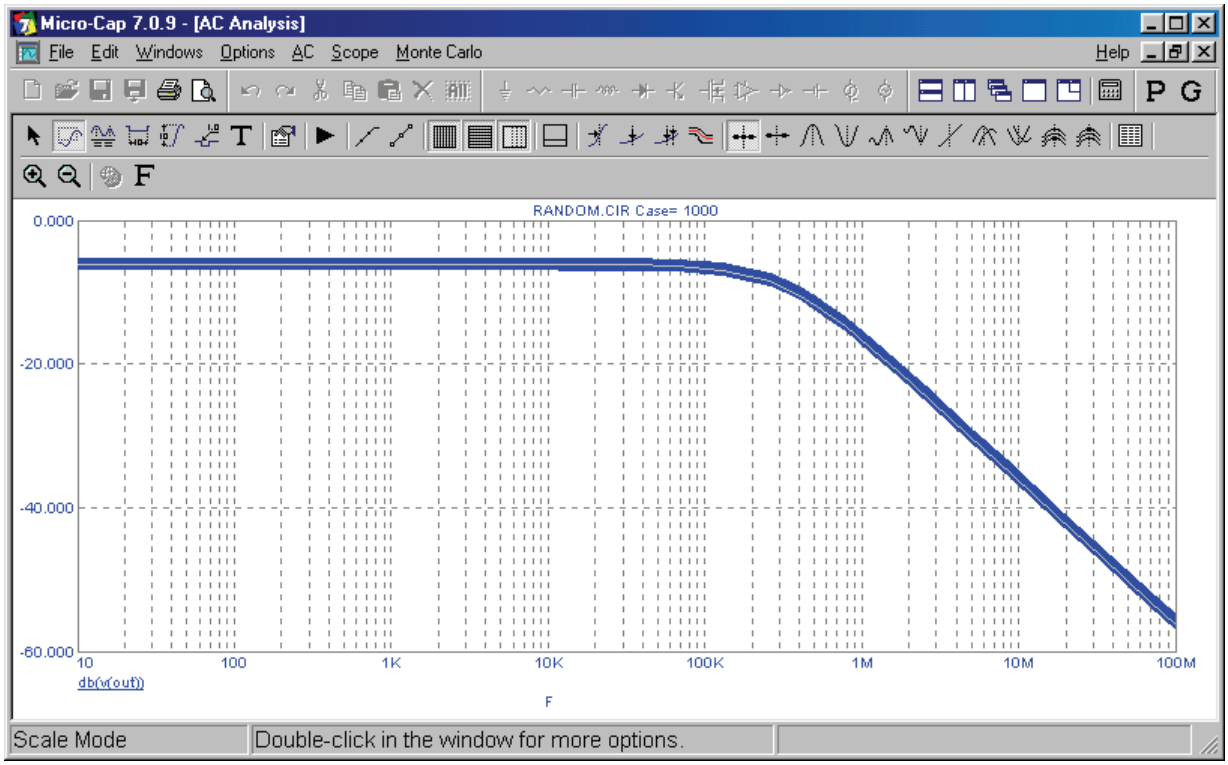
Zde se „centrálně“ volí typ statistického rozdělení („*Uniform*“, „*Gauss*“, „*Worst Case*“ v sekci „*Distribution to Use*“), počet analyzačních běhů („*Number of Runs*“) a povoluje, resp. zakazuje se celá statistická analýza („*Status On/Off*“). Kolonka „*Report When*“ slouží k zápisu případné podmínky, jejíž splnění během simulace pak vede k výpisu hlášení o výskytu dané události do textového souboru „*Numeric Output*“. Po kliknutí na tlačítko „*Get*“ se objeví nabídka dostupných vyhodnocovacích funkcí („*Performance functions*, viz část 9.7.3). Po jejím výběru ji v kolonce „*Report When*“ doplníme požadovaným relačním operátorem. Například podmínka

X_Level(db(v(out)),1,1,-20)>1meg

vyvolá hlášení vždy, když při daném analyzačním běhu protne kmitočtová charakteristika úroveň –20 dB při kmitočtu větším než 1 MHz.

Vyplňování kolonky „*Report When*“ je nepovinné a většinou se této funkce příliš nevyužívá.

V okně „*Monte Carlo Options*“ zvýšíme počet analyzačních běhů na 1000, ponecháme Gaussovské rozložení a aktivujeme status na „*On*“. Potvrdíme „*OK*“. Poté spustíme analýzu horkou klávesou *F*2. Proběhne 1000 analyzačních běhů s výsledkem na **obr. 9.92**.

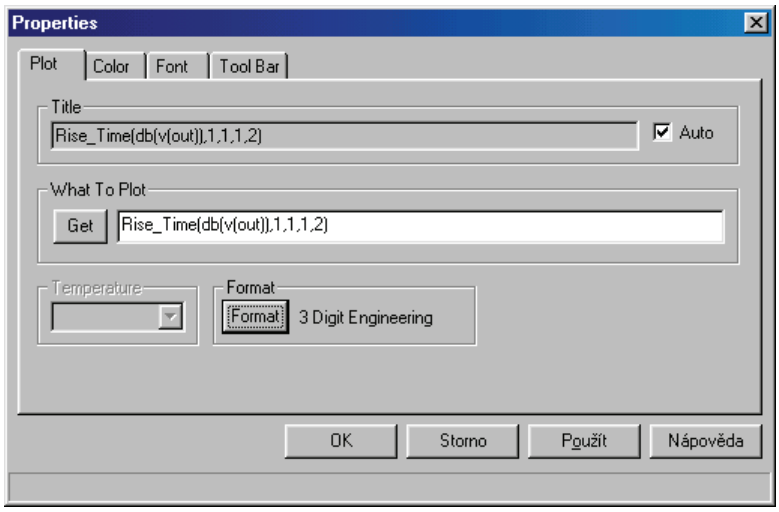


Obr. 9.92: Výsledek statistické analýzy kmitočtové charakteristiky obvodu z **obr. 9.89**.

K vyhodnocení vlivu tolerancí odporů na charakteristiku můžeme využít následujících postupů.

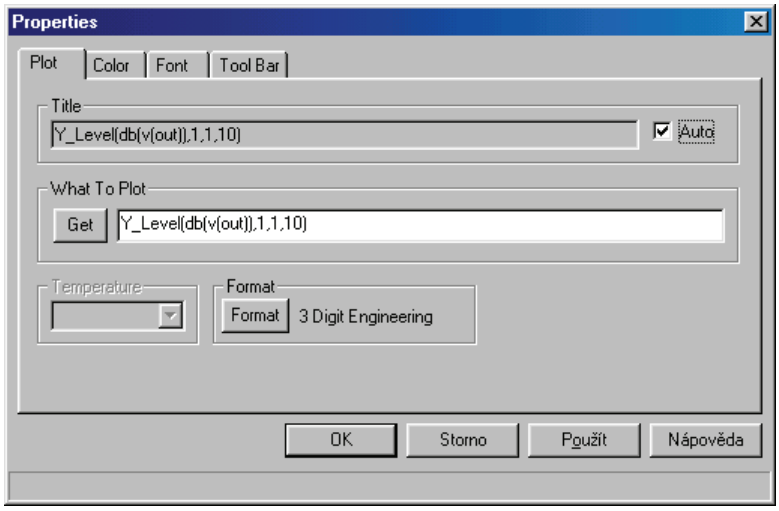
Nyní jsou v paměti data z tisícovky analyzačních běhů. Můžeme je podrobit statistické analýze s využitím vyhodnocovacích funkcí. Zkusme vyhodnotit histogramy stejnosměrného přenosu a lomového kmitočtu třídecibelového poklesu. K tomu však bude zapotřebí použít profesionální verzi MicroCapu, v níž jsou zpřístupněny potřebné vyhodnocovací funkce.

V menu „*Monte Carlo*“ je nyní zpřístupněna volba „*Histograms/Add Histogram*“. Objeví se okno „*Properties*“ podle **obr. 9.93** s přednastavenou vyhodnocovací funkcí „*Rise_Time*“.



Obr. 9.93: Okno k zadávání požadavků na definovaný histogram.

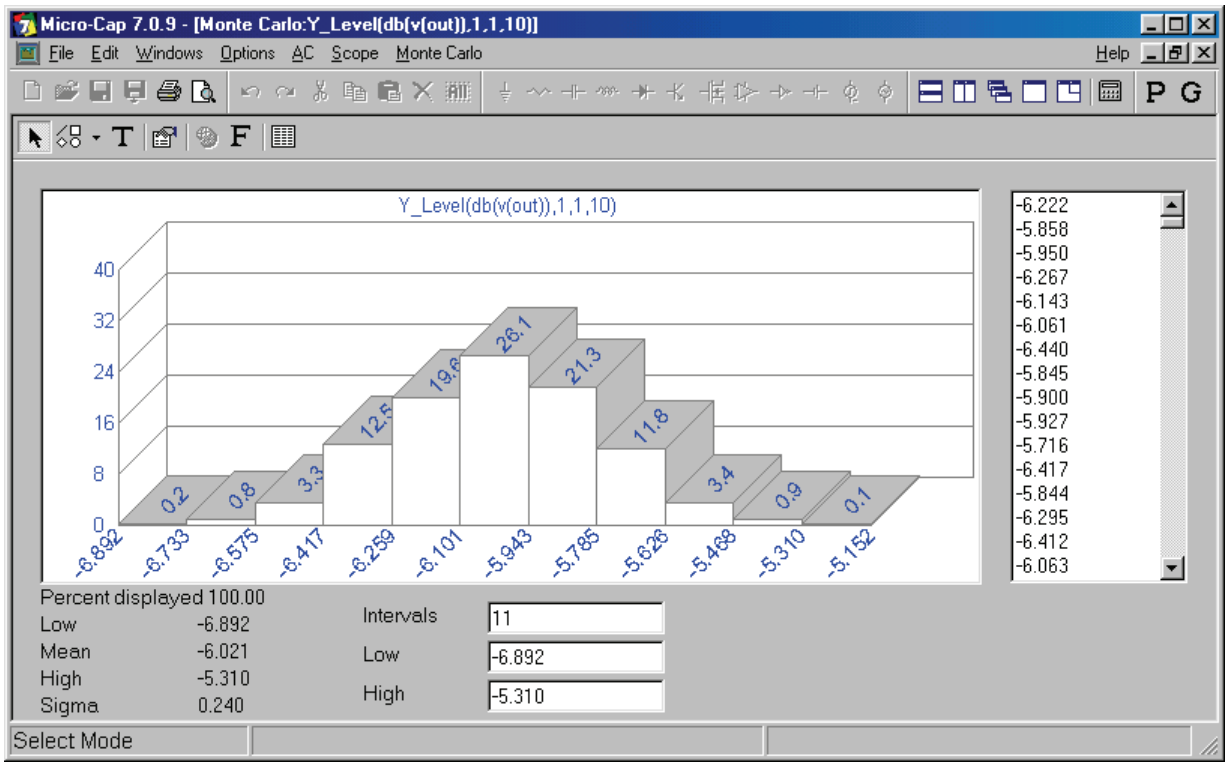
Nejprve stanovíme histogram přenosu filtru na nízkých kmitočtech. Při přesně stejných odporech *R*1 a *R*2 (viz schéma na **obr. 9.89**) vychází přenos –6 dB. Můžeme tedy očekávat největší frekvenci výskytu tohoto přenosu právě v okolí této hodnoty. Přenos budeme měřit na nejnižším analyzovaném kmitočtu 10 Hz vyhodnocovací funkcí „*Y_Level*“. Vymažeme obsah políčka „*What To Plot*“. Klikneme do tlačítka „*Get*“ a v okně „*Get Performance Function*“ vybereme funkci „*Y_Level*“. Hodnotu „*X_Level*“ nastavíme na 10 (=10 Hz). Volbu potvrdíme („*OK*“). Výsledek je na **obr. 9.94**.



Obr. 9.94: Okno „*Properties*“ vyplněné k definici histogramu přenosu filtru na nízkých kmitočtech.

Po kliknutí na „*OK*“ v okně „*Properties*“ se objeví výsledný histogram (viz **obr. 9.95**). Připomínáme, že při analýze jsou hodnoty odporů nastavovány náhodně, neboli po každém spuštění statistické analýzy jinak. Výsledné histogramy tedy mohou pokaždé vypadat jinak, i když při počtu analyzačních běhů 1000 se již začínají projevovat statistické zákonitosti.

Na jednotlivých sloupcích histogramu jsou uvedeny počty výskytů daných událostí v procentech k celkovému počtu analyzačních běhů. Standardní rozdělení vodorovné osy na 11 dílů se dá změnit v okénku „*Intervals*“, jakožto i krajní hodnoty („*Low*“, „*High*“). V pravém sloupci je uvedeno všech 1000 hodnot vyhodnocovací funkce, v našem případě přenosu filtru na kmitočtu 10 Hz, jak postupně „padaly“ v jednotlivých analyzačních běžích.



Obr. 9.95: Histogram přenosu obvodu z **obr. 9.89** na nízkých kmitočtech.

Z histogramu je zřejmé, že k největšímu počtu výskytů (ve 26 procentech) došlo v intervalu od –6,101 dB do –5,943 dB. V levé části obrázku jsou vypočteny tyto hodnoty analyzovaného přenosu: minimální („*Low*“), střední („*Mean*“), maximální („*High*“) a směrodatná odchylka – rozptyl kolem střední hodnoty („*Sigma*“). Střední hodnota odpovídá teoretickému výsledku. Rozptyl činí méně než 4 % ze střední hodnoty přenosu, při toleranci obou odporů 10 %.

Z histogramu se „rýsuje“ Gaussova křivka statistického rozložení, která by byla výraznější při rozdělení vodorovné osy na větší počet dílů a při větším počtu analyzačních běhů.

Nyní stanovíme histogram lomového kmitočtu. Poslouží nám k tomu vyhodnocovací funkce „*X_Level*“ (viz **Tab. 9.2**). Protože se však stejnosměrný přenos mění v každém analyzačním běhu, je třeba tuto funkci zkombinovat s funkcí „*Y_Level*“, pomocí níž „změříme“ vztažný nízkofrekvenční přenos:

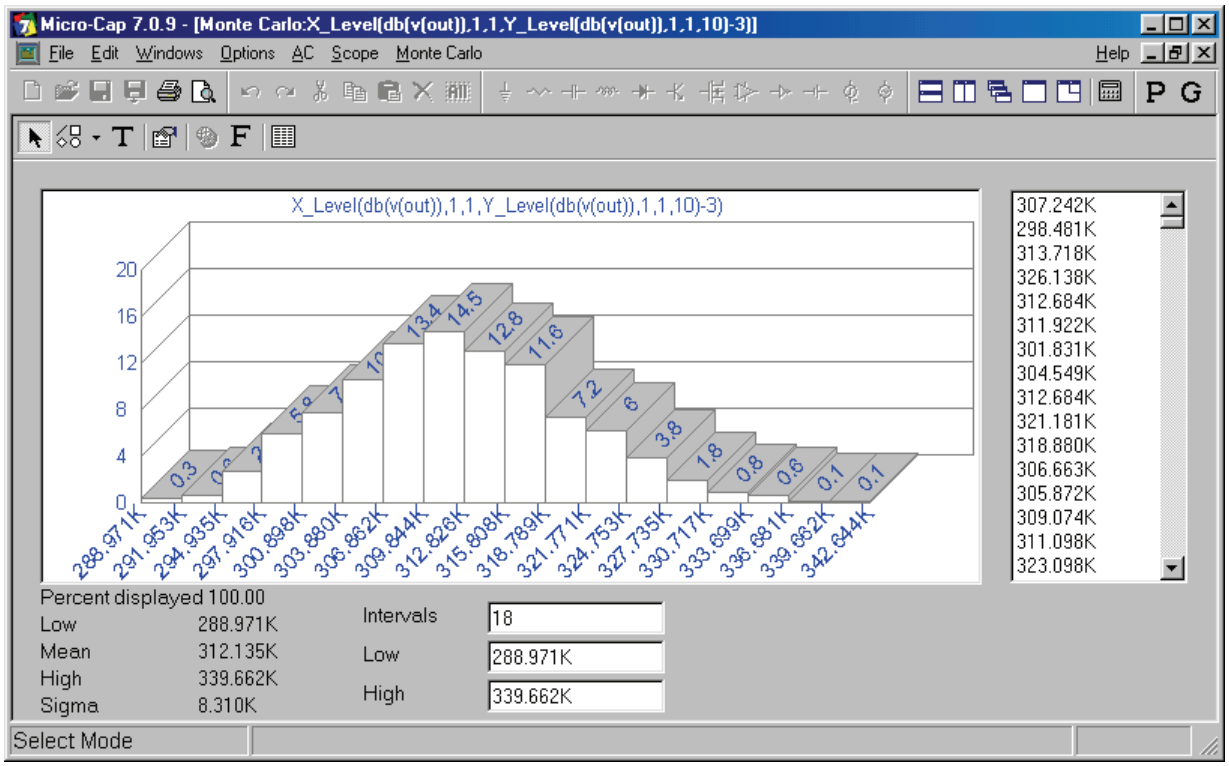
$$X\_Level(db(v(out)),1,1,Y\_Level(db(v(out)),1,1,10)-3)$$

Příklad výsledného histogramu vidíme na **obr. 9.96**, opět pro 1000 analyzačních běhů. Vodorovnou osu jsme rozdělili na 18 dílů, což se ukazuje jako dobrý kompromis mezi jemností ve vykreslení křivky rozložení a statistickou významností počtu událostí v jednotlivých intervalech.

Střední hodnota lomového kmitočtu je asi 312 *k*Hz. Teoretická hodnota se dá odvodit ze schématu na **obr. 9.89** jako

$$\frac{1}{2\pi\dfrac{R_1R_2}{R_1+R_2}C} \doteq 318\ kHz\ .$$

Rozptyl kolem střední hodnoty je menší než 2,7 % této střední hodnoty.



Obr. 9.96: Histogram kmitočtu třídecibelového poklesu přenosu obvodu z **obr. 9.89**.

Doporučujeme vyzkoušet si další ukázkové příklady na statistickou analýzu, které jsou obsaženy v souborech **CARLO.CIR**, **CARLO2.CIR** a **CARLO4.CIR**.

9.7.5 Optimalizace („*Optimization*“)

Co je to optimalizační režim

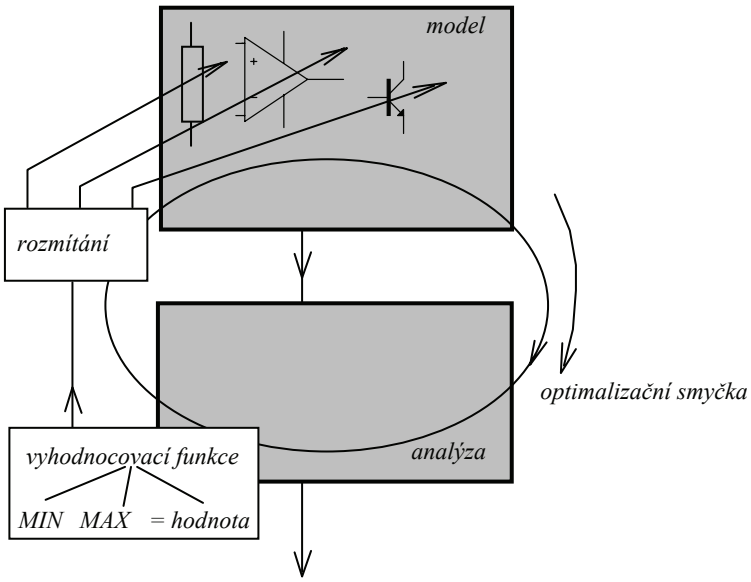
Optimalizace je užitečný režim, který je přístupný pro všechny tři základní typy analýz („*Transient*“, „*AC*“, „*DC*“).

Podle toho, jakých cílů chceme při optimalizaci dosáhnout, je třeba zadat kritéria optimalizace (například maximalizace činného výkonu na rezistoru *R*8). Kritérií může být více a jejich působení se tak může sdružovat (viz dále).

Dále je nutné zadat parametry součástek obvodu, které bude simulátor měnit a hledat jejich optimální velikosti tak, aby bylo naplněno optimalizační kritérium. Je třeba zadat povolený interval změn těchto parametrů, případně i tzv. omezení při optimalizaci („*Constraints*“) ve formě booleovských výrazů (například *VCE*(*Q*1)<=200*m*, tj. kolektorová ztráta tranzistoru *Q*1 nesmí přesáhnout hodnotu 200 *m*W).

Zjednodušené znázornění fungování optimalizačního režimu je na obr. 9.97. Z výsledku analýzy se vypočte zadaná vyhodnocovací funkce. Optimalizační algoritmus vyhodnotí, do jaké míry je splněno kritérium optimalizace, a na základě toho rozhodne o způsobu změny parametrů. Toto vše probíhá v cyklu tak dlouho, dokud není dosaženo cíle optimalizace, nebo – v tom horším případě – až je vyčerpán povolený rozsah změn parametrů bez dosažení cíle.

K optimalizaci využívá MicroCap tzv. *Powellovu* metodu, která má dvě různé formy – standardní („*Standard*“) a vícekrokovou („*Stepping*“). Přednastavená je standardní metoda. Vícekroková metoda nejprve rozdělí interval možných změn parametrů na řadu dílčích podintervalů. Nad každým z nich pak hledá „lokální optimum“ podle zadaného kritéria. Nakonec se rozhodne pro „nejlepší“ z dílčích výsledků. Tato metoda často vede na velmi rozsáhlé výpočty. Měli bychom rozhodně v první fázi zvolit standardní metodu a pouze v případě problémů přejít na metodu vícekrokovou.



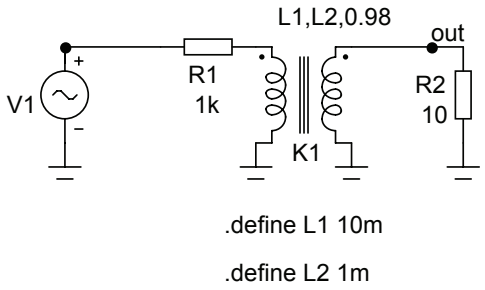
Obr. 9.97: Schéma mechanismu optimalizace.

Protože demoverze MicroCapu má zpřístupněnu jen jednu vyhodnocovací funkci („*Rise_Time*“), jsou zde velmi omezeny možnosti zadávání kritérií optimalizace. Vzorové příklady **OPT1.CIR** až **OPT4.CIR** proto v demoverzi nelze vyzkoušet. Doporučujeme tedy alespoň spustit si demonstraci optimalizačního režimu v nabídce „*Help/Optimizer Demo*“.

V následující části popíšeme na konkrétním příkladu možný způsob práce v optimalizačním režimu. Příklad využívá vyhodnocovací funkce, která je k dispozici pouze v profesionální verzi programu.

Příklad optimalizace

Na **obr. 9.98** je ukázka výkonového přizpůsobení zdroje a spotřebiče pomocí transformátoru. Model obvodu naleznete v dodatkovém souboru **OPTIMA.CIR**.



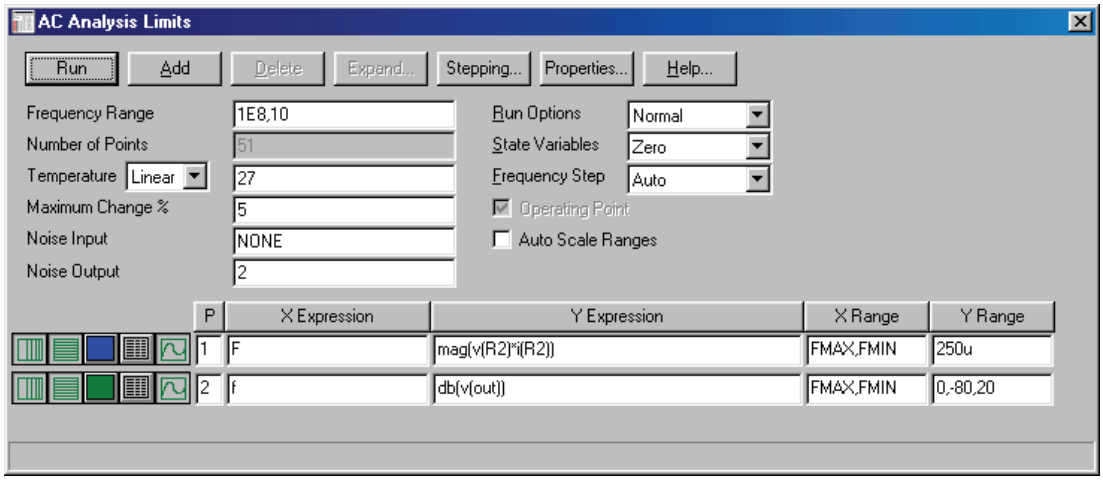
Obr. 9.98: Obvod k demonstraci optimalizace činného výkonu do zátěže *R*2.

Zdroj harmonického signálu má kmitočet 100 *k*Hz, ostatní jeho parametry jsou implicitní. Transformátor má definovány indukčnosti primárního (*L*1) a sekundárního (*L*2)

vinutí a činitel vazby 0,98. Indukčnosti jsou definovány přes symbolické proměnné, aby je bylo možno variovat při optimalizaci. Úkolem bude optimalizovat $L1$ a $L2$ tak, aby na kmitočtu 100 *k*Hz byl maximální přenos činného výkonu do odporové zátěže $R2$.

Z teorie vyplývá, že v případě ideálního transformátoru by poměr čtverců počtu závitů primární a sekundární cívky měl být roven poměru odporů $R1$ a $R2$, tedy 100. Tomuto poměru by měl zhruba odpovídat poměr indukčností $L1$ a $L2$. Ve skutečnosti je tento poměr 10. Kromě toho je však třeba uvážit kmitočtové závislosti přenosu, které nejsou v těchto jednoduchých výpočtech zohledněny.

Kdybychom uvažovali efektivní hodnotu vstupního harmonického napětí 1 V, pak by podle teorie byl maximální možný přenesený výkon do $R2$ roven výkonu na $R1$, tj $0{,}5^2/R1=250$ μW. Při výkonovém přizpůsobení je totiž napětí na $R1$ rovno polovině napětí zdroje.



Obr. 9.99: Nastavení podmínek „*AC*“ analýzy obvodu z **obr. 9.98**.

Aktivujeme analýzu „*AC*“. Okno „*AC Analysis Limits*“ by mělo být vyplněno tak, jak je znázorněno na **obr. 9.99**. Výraz

$$Mag(v(R2)*i(R2))$$

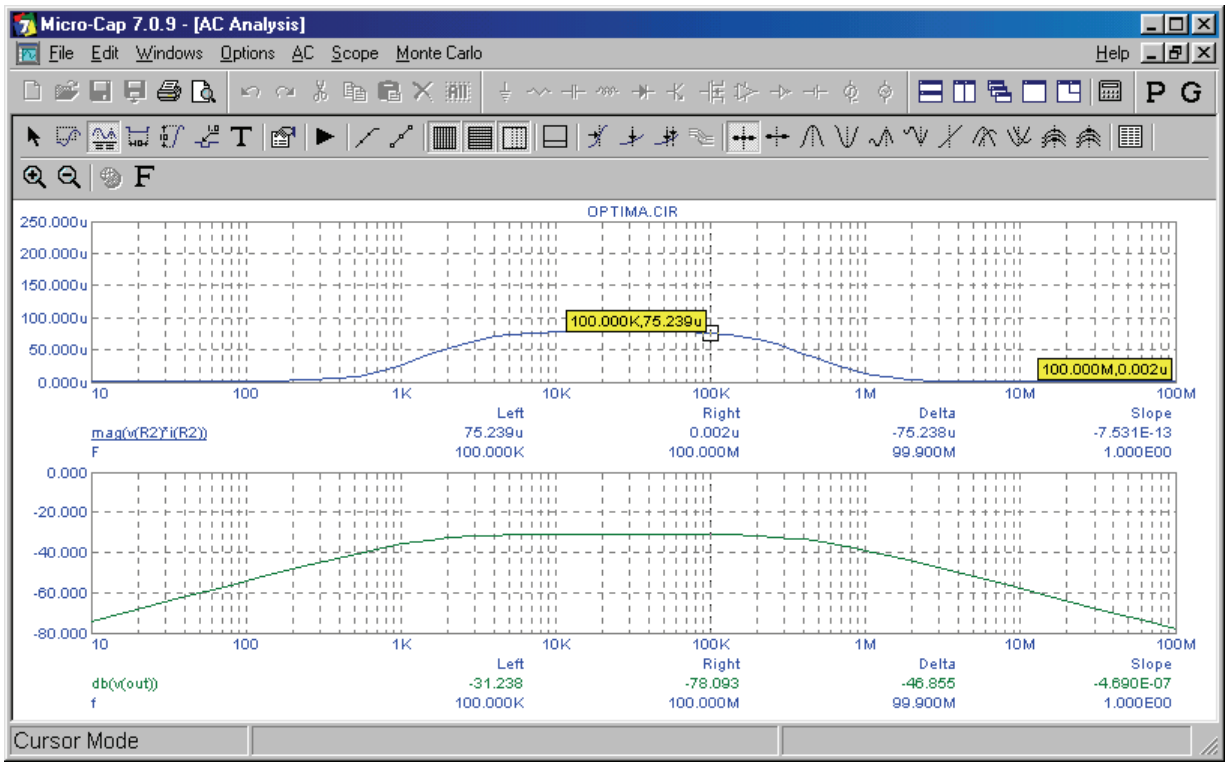
definuje modul součinu fázorů napětí a proudu na odporu $R2$, tedy činný výkon (za předpokladu, že velikosti fázorů chápeme jako efektivní hodnoty).

Po provedení analýzy obdržíme výsledky na **obr. 9.100**.

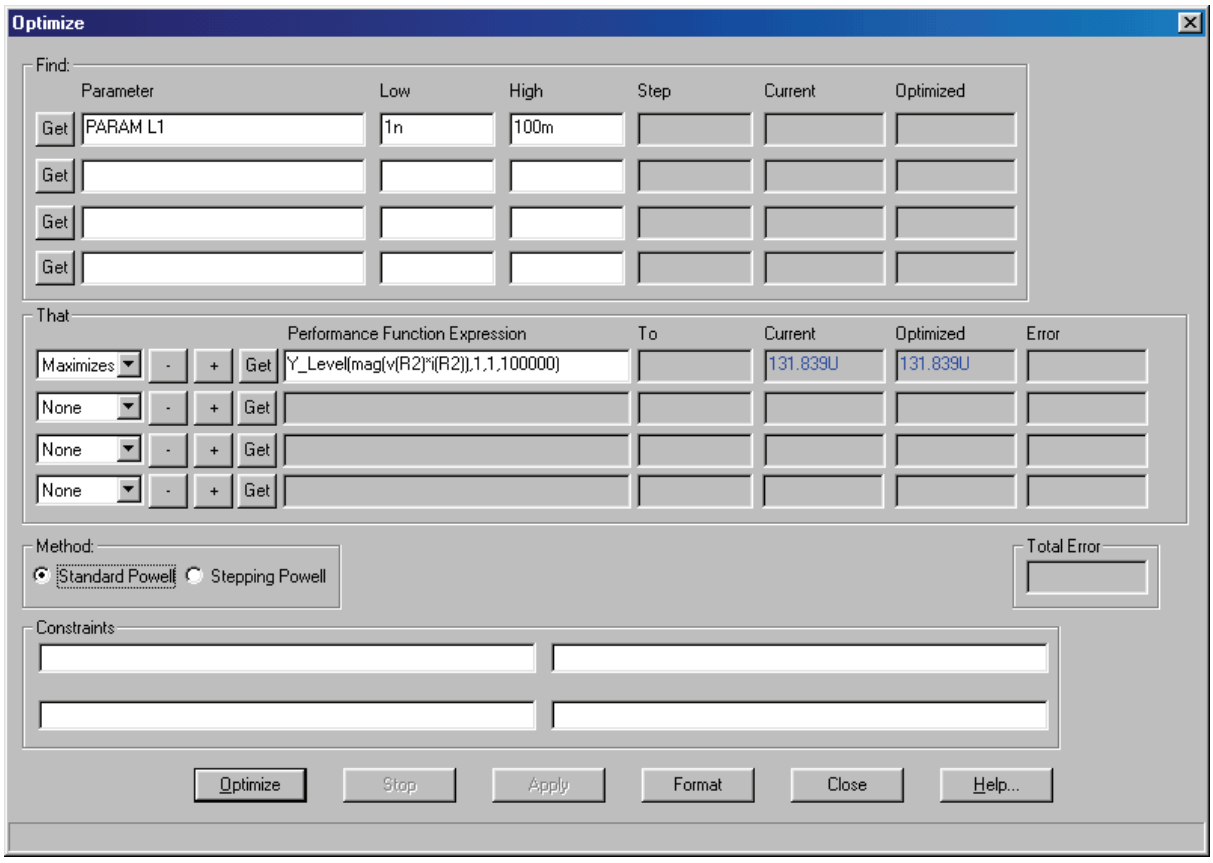
Z obrázků vyplývá, že obvod se chová z hlediska vstupně-výstupního přenosu jako pásmová propust. Kmitočet 100 *k*Hz, na němž máme maximalizovat přenášený výkon, leží poblíž hranice propustného pásma. Velikost výkonu na zátěži je však relativně malá, asi 75 μW.

Pokusme se nejprve zvýšit tento výkon optimalizací indukčnosti $L1$ a potom současnou optimalizací $L1$ a $L2$.

Na horní liště klikneme do položky „*AC*“ a zvolíme „*Optimize*“. Vyvoláme tím okno „*Optimize*“ podle **obr. 9.101**.



Obr. 9.100: Kmitočtová závislost činného výkonu na zátěži a napěťového přenosu před optimalizací obvodu.



Obr. 9.101: Okno pro zadání podmínek a vyhodnocení výsledků optimalizace. Program variuje $L1$ a hledá maximum výkonu na $R2$.

V skupině „*Find*“ jsou čtyři řádky pro zadávání parametrů, jejichž variací bude program optimalizovat cílovou funkci. V prvním řádku je zadán jediný parametr *L*1. Zadávání se děje nepřímo přes tlačítko „*Get*“, které zpřístupní nabídku možných parametrů. Během optimalizace program může „zkoušet“ měnit tento parametr v mezích „*Low*“ (od 1 *n*H) do „*High*“ (do 100 *m*H).

V skupině „*That*“ definujeme cíle optimalizace, na každém řádku jedno kritérium optimalizace.

„Rozbalovací“ okno obsahuje tyto položky:

Minimizes - Hledá se minimum vyhodnocovací funkce.
Maximizes - Hledá se maximum vyhodnocovací funkce.
Equates - Vyhodnocovací funkce se bude rovnat zadané hodnotě.
None - Není vybráno žádné kritérium.

Pomocí ikonek – a + můžeme podle potřeby ubírat nebo přidávat řádky pro další cíle optimalizace. Jak je vidět, je možné kombinovat několik optimalizačních kritérií. Platí jediné omezení: funkce „*minimizes*“ a „*maximizes*“ není možné kombinovat s funkcemi „*equates*“.

Ikonou „*Get*“ vybíráme požadovanou vyhodnocovací funkci. Argumentem funkce může být pouze některá z proměnných, definovaná v okně „*Analysis Limits*“ v poli „*Y Expressions*“. V našem případě je zvolena funkce

*Y_Level(mag(v(R2)*i(R2)),1,1,100000)*

která definuje velikost činného výkonu na kmitočtu 100 *k*Hz.

Položky v sloupci „*To*“ jsou aktivní pouze při výběru kritéria „*Equates*“: pak se zde zapíše hodnota vyhodnocovací funkce, které chceme při optimalizaci dosáhnout (například 0, chceme-li optimalizací vyvážit můstek apod.).

V položkách „*Current*“ a „*Optimized*“ se průběžně objevují hodnoty, odpovídající současné a optimalizované hodnotě vyhodnocovací funkce. Jejich rozdíl se zobrazuje v položce „*Error*“. Hodnoty, které vidíme na **obr. 9.101**, jsou údaji z předchozích optimalizačních běhů.

V skupině „*Method*“ může uživatel vybrat standardní nebo vícekrokovou Powellovu metodu.

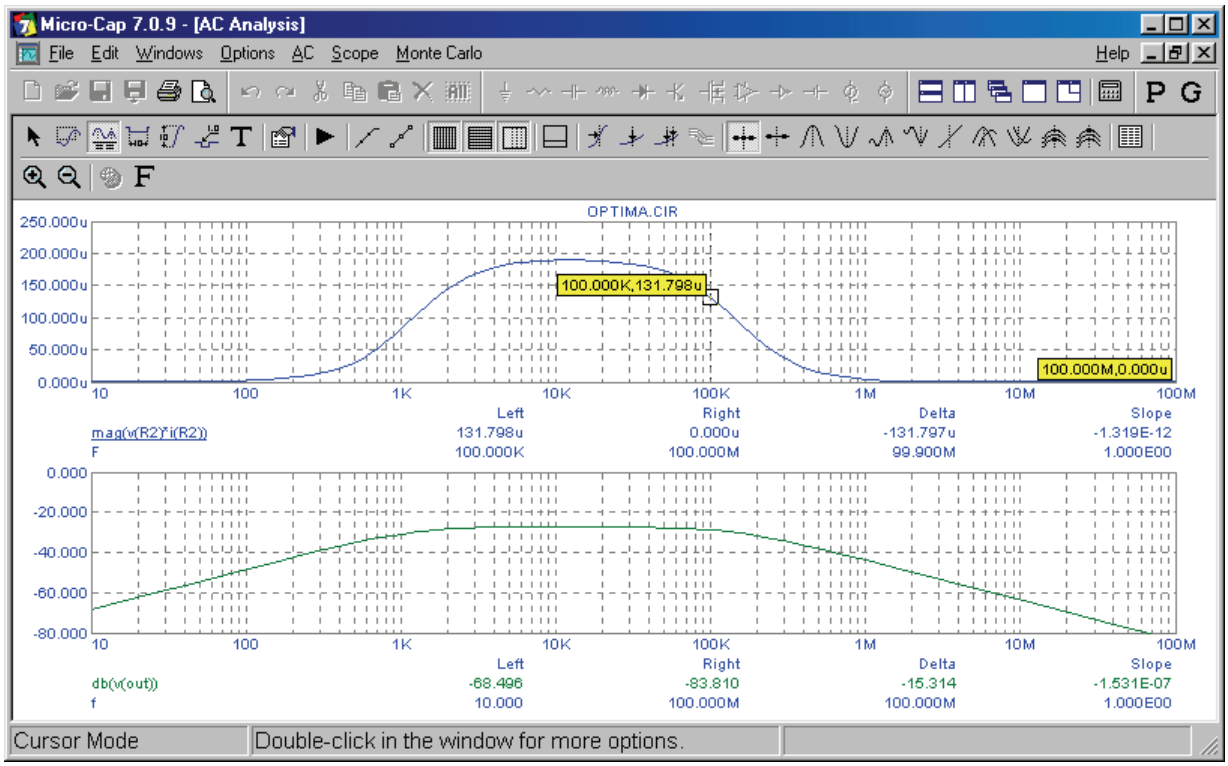
V skupině „*Constraints*“ je možné zadat až čtyři omezující podmínky optimalizace.

Optimalizační běh spustíme kliknutím na ikonu „*Optimize*“. Výsledkem budou tyto údaje v sloupcích „*Optimized*“:

Optimální velikost *L*1 je 37,296 *m*H.

Tomu odpovídající velikost výkonu na kmitočtu 100 *k*Hz je 131,839 μW.

V spodní řadě ikon vyhledáme „*Apply*“ a klikneme na ni. Nalezená optimální velikost *L*1 nyní nahradí ve schématu původní hodnotu 10 *m*H. Analýza bude nyní probíhat s novou hodnotou *L*1. Zavřeme okno „*Optimize*“ a spustíme analýzu horkou klávesou *F*2. Výsledky jsou na **obr. 9.102**.



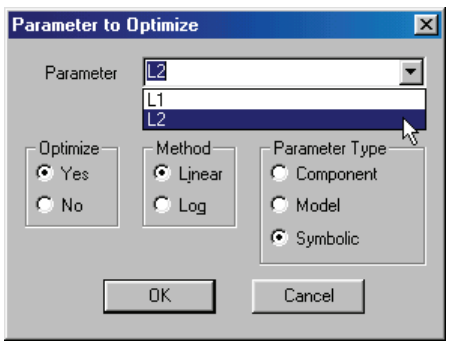
Obr. 9.102: Kmitočtová závislost činného výkonu na zátěži a napěťového přenosu po optimalizací obvodu podle parametru *L*1.

Po porovnání s **obr. 9.100** můžeme konstatovat, že výkon na kmitočtu 100 *k*Hz sice vzrostl, ale že se nacházíme na boku kmitočtové charakteristiky. Přesun energie výstupního signálu do okolí kmitočtu 100 *k*Hz bude možný až při současné optimalizaci indukčností *L*1 a *L*2.

Vrátíme se do okna „*Optimize*“ („*AC*/*Optimize*“, resp. *Ctrl*+*F*11). Do dalšího řádku pod definici „*PARAM L*1“ dodefinujeme parametr *L*2: Klikneme do ikonky „*Get*“ v daném řádku. Objeví se okno „*Parameter To Optimize*“, které vyplníme podle **obr. 9.103**. Potvrdíme „*OK*“. Rozmezí změn *L*2 („*Low*“, „*High*“) zvolíme stejné jako u *L*1. Spustíme optimalizační proces („*Optimize*“). Po jeho proběhnutí obdržíme tyto výsledky:

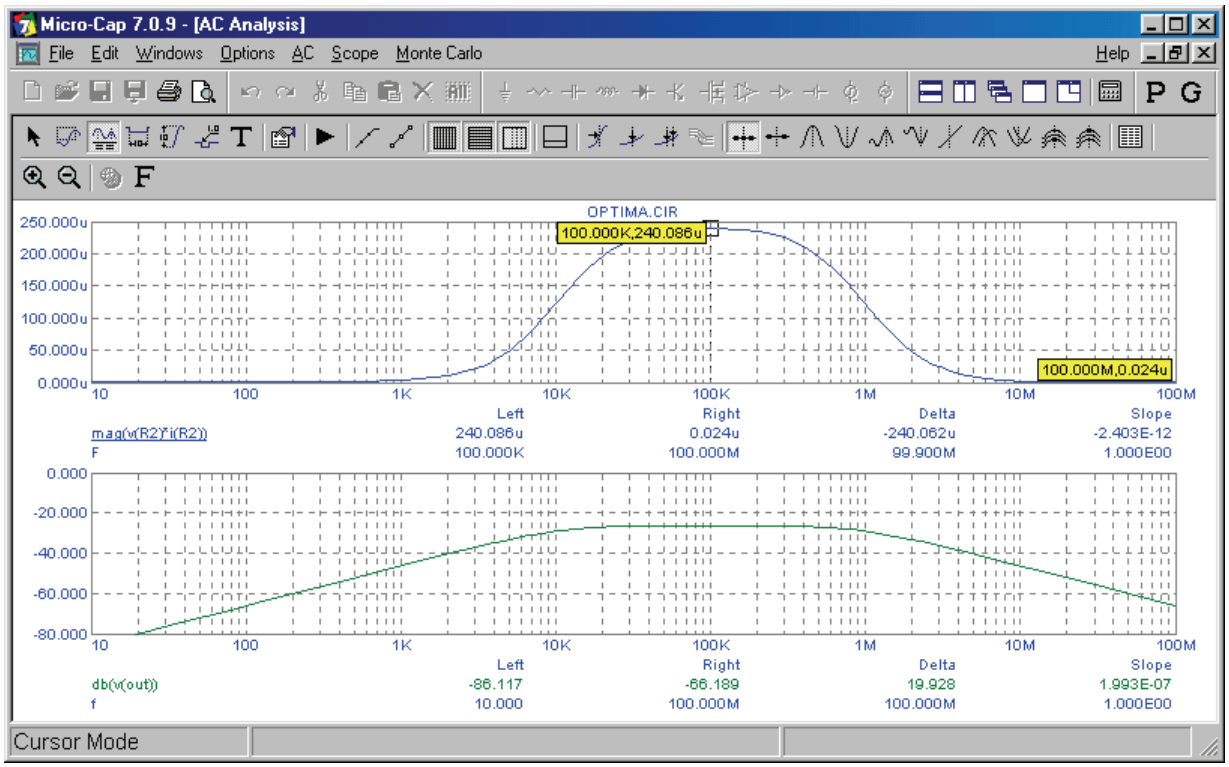
Optimální velikost *L*1 je 8,032 *m*H.
Optimální velikost *L*2 je 80,295 μH.

Tomu odpovídající velikost výkonu na kmitočtu 100 *k*Hz je 240,1 μW.



Obr. 9.103: Okno pro přidávání parametrů k optimalizaci.

Klikneme na „*Apply*“, zavřeme okno „*Optimize*“ a spustíme analýzu (*F*2). Konečný výsledek je na **obr. 9.104**.



Obr. 9.104: Kmitočtová závislost činného výkonu na zátěži a napěťového přenosu po konečné optimalizací obvodu podle parametrů *L*1 a *L*2.

Vidíme, že do teoretického maxima výkonu 250 μW chybí asi 10 μW. Je to způsobeno činitelem magnetické vazby, který je menší než 1 (0,98, viz **obr. 9.98**).

9.7.6 Veřejné a privátní knihovny modelů a jejich úloha v analyzačních režimech

Knihovny modelů prvků mohou mít obecně přiřazen jeden ze dvou tzv. statutů: status privátních („*Private*“) nebo veřejných („*Public*“) knihoven. Standardně je status nastaven na „*Private*“ v globálních podmínkách simulátoru („*Options/Global Settings*“) pomocí zatržených položek *PRIVATEANALOG* a *PRIVATEDIGITAL* (status analogových a digitálních knihoven). Zrušením těchto aktivací budou mít příslušné knihovny status „*Public*“.

Status knihoven může ovlivnit mechanismy rozmítání parametrů součástek v knihovnách v režimech krokování, statistické analýzy a optimalizace.

Krokování:

Z **obr. 9.73** je zřejmé, že při krokování jsou parametry vybírány ze skupin označených jako „*Component*“, „*Model*“ a „*Symbolic*“. V prvních dvou skupinách mohou být parametry z knihoven prvků.

Uvažujme příklad, kdy obvod obsahuje dva tranzistory *Q*1 a *Q*2. Oběma tranzistorům je přiřazem tentýž model 2*N*2221. Chceme krokovat parametr „*BF*“ tranzistoru *Q*1, případně obou tranzistorů současně.

Vybereme-li při krokování skupinu parametrů „*Component*“, dovedou nás jednotlivé nabídky okna „*Stepping*“ k obsahu položky „*Step What*“

*Q*1.*BF*

Mají-li knihovny status „*Private*“, pak bude krokován parametr *BF* jen u tranzistoru *Q*1. Při statusu „*Public*“ však budou současně krokovány parametry *BF* obou tranzistorů. Tranzistory totiž sdílejí stejný model, který je nyný „veřejný“. Změna jakéhokoliv parametru modelu se tudíž promítne do chování všech součástek, které tento model sdílejí.

Vybereme-li při krokování skupinu parametrů „*Model*“, změní se obsah položky „*Step What*“ takto:

2*N*2221.*BF*

Nyní se při krokování budou měnit parametry obou tranzistorů bez ohledu na to, jaký je status knihoven. Je tomu tak proto, že krokujeme parametr modelu, a ne parametr konkrétního tranzistoru.

Z uvedeného vyplývá, že chceme-li krokovat individuálně parametr jediné součástky, i když její model sdílejí další součástky, je to možné jen při krokování typu „*Component*“ a privátní knihovně.

Statistická analýza:

Při statistické analýze dochází rovněž ke krokování parametrů, které je však řízeno generátory náhodných čísel.

Uvažujme opět obvod obsahující dva tranzistory *Q*1 a *Q*2, kterým je přiřazem tentýž model 2*N*2221. V tomto modelu provedeme zápis

BF=100 *LOT*=10%

V případě statusu *PRIVATEANALOG = OFF*, tj. veřejných knihoven, by oba tranzistory *Q*1 a *Q*2 měly nastavenu v daném analyzačním běhu stejnou (i když náhodnou) hodnotu *BF* (viz **obr. 9.105** a). Při standardním nastavení privátních knihoven však bude každý tranzistor vykazovat různé hodnoty *BF* (**obr. 9.105** b). Srovnejte s **obr. 9.88** a).

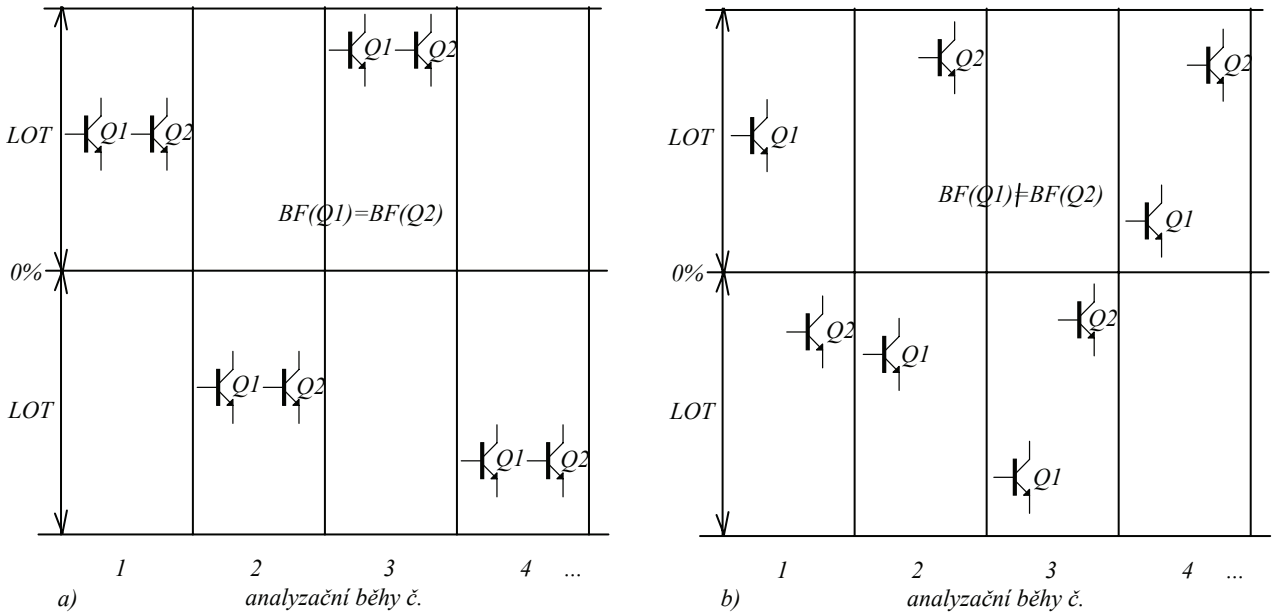
Jinak je tomu při použití relativní tolerance *DEV*. Uvažujme například zápis

BF=100 *LOT*=10% *DEV*=1%

Objeví-li se v zápisu modelu prvku slovo *DEV*, stává se tento model privátním bez ohledu na to, jak je nastaven status *PRIVATEANALOG* v globálních podmínkách. Rozmítání nyní funguje tak, jak je znázorněno na **obr. 9.88** b): v daném analyzačním běhu vykazuje každá součástka jiné hodnoty parametrů.

Optimalizace:

Při optimalizaci vybíráme „volné“ parametry, které bude optimalizační algoritmus variovat, ze stejných skupin jako při klasickém krokování, tedy „*Component*“, „*Model*“ a „*Symbolic*“ (viz **obr. 9.103**). Pro působení privátních a veřejných knihoven tedy platí stejné zákonitosti jako v režimu krokování.



Obr. 9.105: Charakter variace parametrů tranzistorů se stejným modelem během statistické analýzy v případě a) veřejných, b) privátních knihoven.

9.7.7 Další speciální funkce simulátoru

9.7.7.1 Možnosti detailního sledování proměnných a trasování během analýzy

V některých případech uživatel simulátoru ocení možnost prohlížení hodnot obvodových proměnných nebo různých funkcí během analyzačních běhů. Další výhodnou vlastností může být rozfázování analýzy do jednotlivých po sobě jdoucích kroků s možností prohlížení výsledků v každém kroku, případně s možností ručního krokování nebo automatického zastavení analýzy při splnění konkrétních podmínek.

MicroCap v tomto směru nabízí tyto základní možnosti:

- „*P*-Key“ (klávesa *P*),
- „Animation Mode“ (animační mód),
- „;Watch Grid Text” (text ;watch),
- „Watch Window“ (sledovací okno),
- „Breakpoints“ (*Stop* body).

„P-Key“ – klávesa „P“

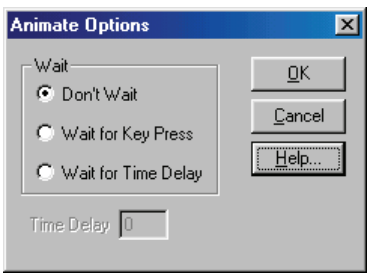
Klepneme-li během analýzy „*Transient*“, „*AC*“ nebo „*DC*“ do klávesy „*P*“, analýza se zpomalí a v grafech vedle jmen zobrazovaných proměnných se objeví údaje o jejich aktuálních hodnotách. Další klepnutí na tlačítko „*P*“ způsobí ukončení snímání hodnot. Číselné údaje se přestanou aktualizovat a znamenají tedy okamžité hodnoty v okamžiku poslední aktivace tlačítka „*P*“. Při dalším klepnutí na klávesu se vše opakuje.

Tento způsob snímání aktuálních hodnot analyzovaných veličin je velmi primitivní a příliš se nepoužívá: na pomalých počítačích představuje další citelné zpomalení analýzy a v případě výkonných počítačů naopak uživatel nestačí rychle se měnící data sledovat.

„Animation mode“ – animační mód

Animace znamená rozfázování analyzačního běhu na posloupnost jednotlivých výpočetních bodů, s možností přecházet od jednoho bodu k druhému buď manuálním krokováním, nebo přinutit program, aby mezi jednotlivé kroky vkládal definované časové prodlevy. Tím jsou vytvořeny podmínky pro přehledné sledování počítaných veličin a závislostí.

Po zadání dat do okna „*Analysis Limits*“ je možné aktivovat animační mód volbou „*Scope/Animate Options*“, resp. kliknutím na ikonu  na liště hlavního okna MicroCapu. Objeví se okno “*Animate Options*” podle **obr. 9.106**.



Obr. 9.106: Okno pro volbu animačních režimů.

Okno umožňuje vybrat jednu ze tří možností v skupině "*Wait*":

„**Don't Wait**" Animace je neaktivní.

"**Wait for Key Press**" Program čeká na dvojstisk *Ctrl-Space*, po něm provede výpočet dalšího bodu a opět čeká.

"**Wait for Time Delay**" Probíhá výpočet bod po bodu, ale mezi sousedními výpočty program čeká určitou dobu, která je specifikovaná v položce „*Time Delay*" (v sekundách). Je možné zadat i čas 0. I pak bude program v animačním módu, na rozdíl od položky "*Don't Wait*".

Spustíme-li analýzu v animačním módu, máme možnost sledovat obvodové veličiny několika metodami: přímo ve schématu, pomocí *grid textu* „*;watch*" nebo v speciálním okně „*Watch Window*".

Při použití první možnosti je výhodné umístit na obrazovku vedle sebe okno schématického editoru spolu s oknem výsledků analýzy (volbou „*Windows/Tile Vertical*") a pomocí ikon  na liště editoru aktivovat zobrazení uzlových napětí, proudů, výkonů nebo stavů aktivních součástek. Tyto údaje se při animaci budou aktuálně obnovovat.

Animační mód byl původně vyvinut pro účely časové analýzy. Při animaci se ve schématu objevují aktuální velikosti pozorovaných signálů. Taktéž je tomu při analýze „*DC*", kdy můžeme ve schématu pozorovat proces rozmítání stejnosměrných poměrů v obvodu. Výjimku však tvoří analýza „*AC*". Protože ve schématu není možné zobrazit signály v harmonickém ustáleném stavu, zobrazují se souřadnice stejnosměrného pracovního bodu. K změnám tedy může docházet pouze při krokování parametrů součástek nebo při statistické analýze. Chceme-li přesto sledovat vývoj střídavých veličin, musíme použít jednu z dále popsaných metod.

„;Watch Grid Text" (text ;watch)

Máme-li například v úmyslu sledovat hodnoty napětí na kapacitoru *C*1, umístíme v editoru do složky "*Text*" následující *grid text*:

;watch(v(C1))

Máme-li pak v animačním módu možnost sledovat okno s touto složkou, objeví se na místě tohoto textu výsledek

;v(C1)=aktuální hodnota

Takovýmto způsobem můžeme současně sledovat širokou škálu proměnných.

„Watch Window" (sledovací okno)

Je-li aktivována určitá analýza, například „*Transient*", pak sledovací okno přidáme na plochu volbou „*Transient/Watch*". Na **obr. 9.107** je ukázka, jak se automaticky rozmístí na plochu okno editoru, analyzační okno a okno „*Watch*". Do sloupce „*Expr*" sledovacího okna zapíšeme vzorce sledovaných závislostí. V sloupci „*Value*" se pak během animačního módu objevují jejich číselné hodnoty.

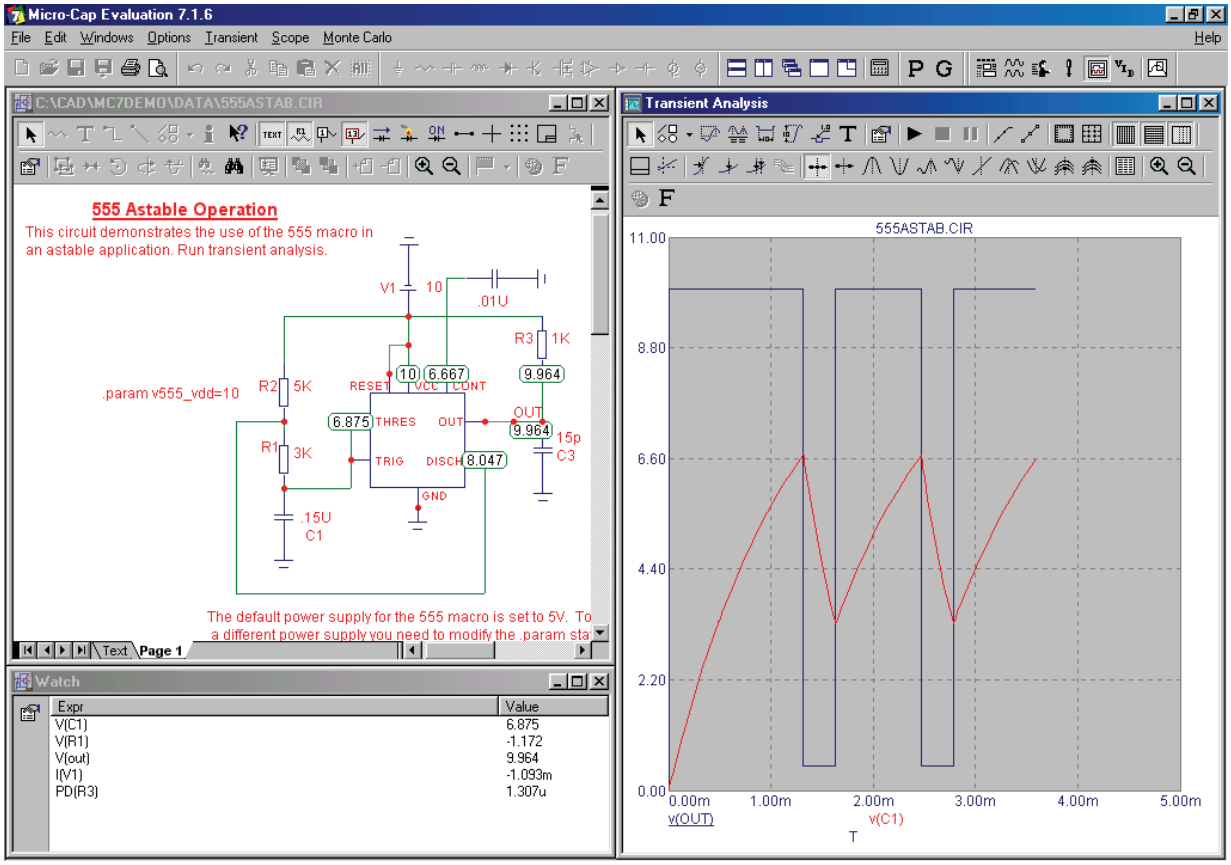
"Breakpoints" (*Stop* body)

Často je potřeba zastavit analýzu, pokud dojde k naplnění přesně stanovených podmínek, například jestliže napětí mezi dvěma uzly překročí konkrétní hodnotu. Pomocí okna "*Watch*" si pak můžeme prohlédnout hodnoty proměnných nebo funkcí, které nás v daném časovém bodě zajímají.

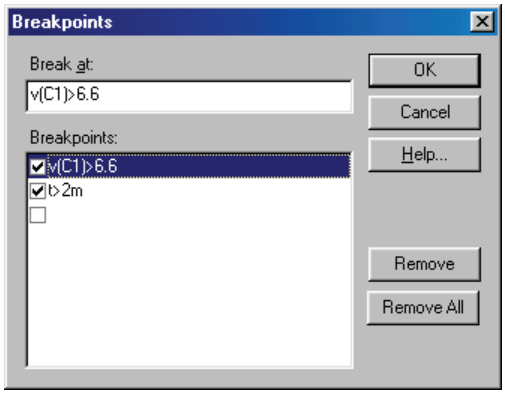
Stop body se po aktivaci určité analýzy, např. analýzy „*Transient*“, definují volbou

Transient/Breakpoints

Objeví se okno „*Breakpoints*“. Způsob definice a aktivace jednotlivých bodů je zřejmý z **obr. 9.108**.



Obr. 9.107: Ukázka sledovacího okna „*Watch*“ v animačním módu.



Obr. 9.108: Definice *Stop* bodů – „*Breakpoints*“.

Stop body jsou aktivní v klasickém, nikoliv v animačním módu. Jakmile se analýza zastaví v daném bodu, je možné pokračovat dvojstiskem *Ctrl+Space*.

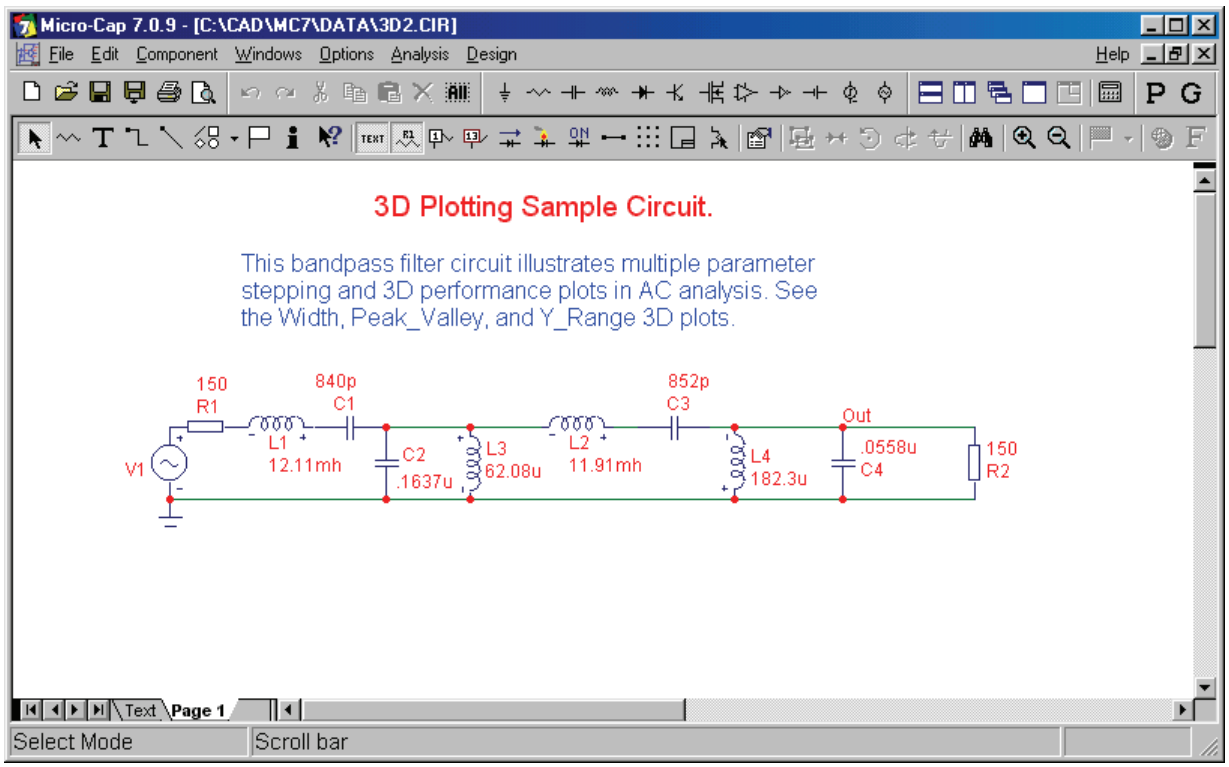
Poznámka: Existuje další způsob, jak sledovat vývoj veličin (konkrétně uzlových napětí a proudů induktory) během simulace v analýzách „*Transient*“ a „*DC*“ – pomocí okna „*State Variables Editor*“, které umístíme na plochu tak, aby nebylo překryto oknem s výsledky analýzy.

9.7.7.2 Další užitečné funkce

Kromě základních a rozšiřujících typů analýz obvykle poskytují různé simulační programy další užitečné funkce. U MicroCapu zmíníme režim "3*D Windows*". Je však přístupný jen u profesionální verze. Zájemcům, kteří profesionální verzi nevlastní, doporučujeme alespoň shlédnutí demonstračního běhu v nabídce „*Help/3D Plots Demo*".

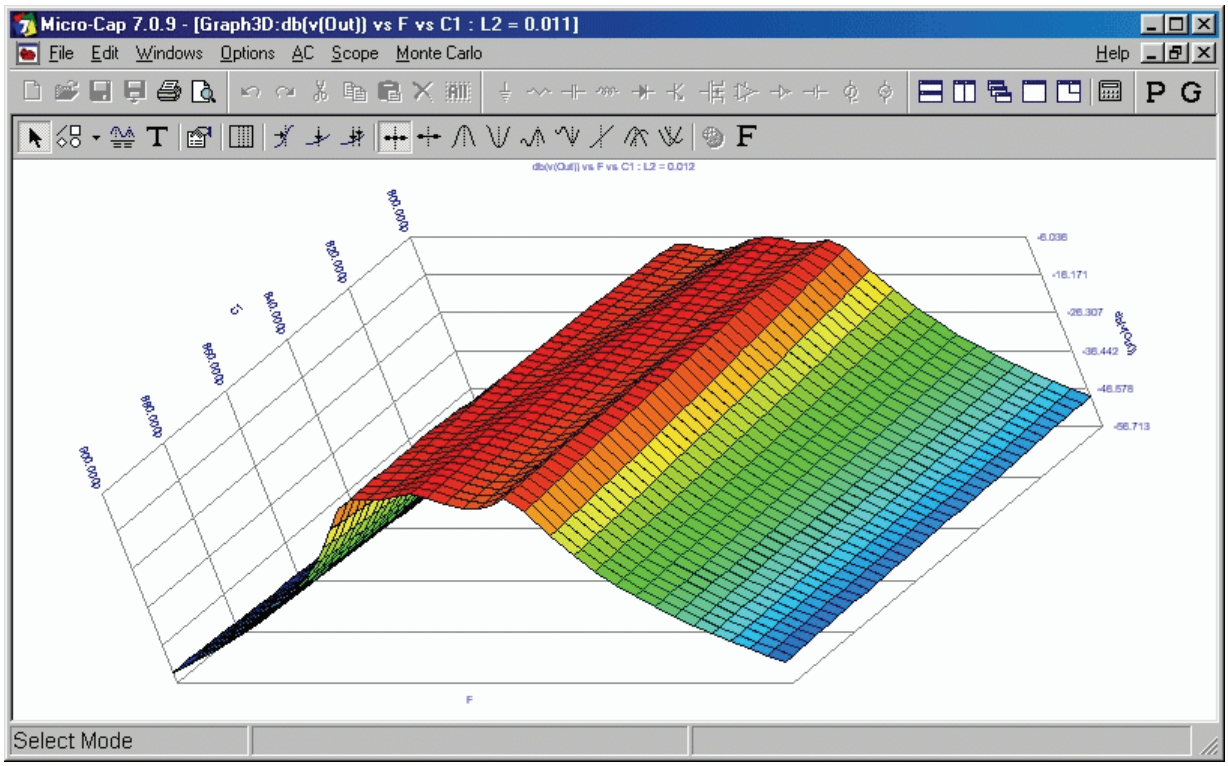
V režimu "3*D Windows*" se analyzované závislosti zobrazují "trojrozměrně" pomocí ortogonálních os *X*, *Y* a *Z*. Nutnou podmínkou je krokování alespoň jednoho parametru.

Jeden z demonstračních příkladů nalezneme v souboru **3D2.CIR**. Jedná se o pasivní příčkový filtr typu pásmová propust podle **obr. 9.109**.



Obr. 9.109: Obvod k demonstraci funkce třírozměrných grafů.

Po provedení analýzy „*AC*" zjistíme, že je aktivní režim krokování součástek *L*2 a *C*1. Režim "3*D Windows*" aktivujeme volbou „*AC/3D Windows/Add 3D Windows*". Na **obr. 9.110** je ukázka závislosti amplitudové kmitočtové charakteristiky filtru na kapacitě *C*1.

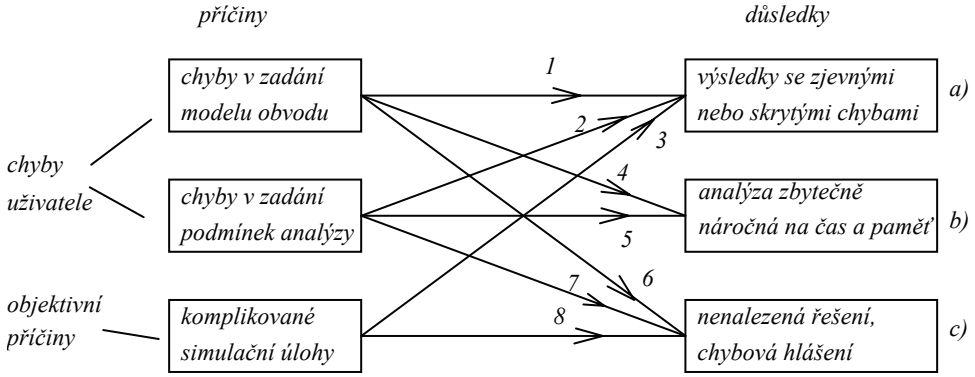


Obr. 9.110: Závislost kmitočtové charakteristiky obvodu z **obr. 9.109** na kapacitě *C*1.

9.8 Problémy při počítačové simulaci a jak se s nimi vypořádat

9.8.1 Vybrané problémy při simulaci a jejich příčiny

Na problémy při počítačové simulaci narazí jak začátečník, tak čas od času i zkušený profesionál. Tyto problémy lze zhruba rozčlenit do tří kategorií (viz **obr. 9.111**).



Obr. 9.111: Příčiny a projevy problémů při počítačové simulaci.

a) Výsledky jsou zatíženy chybami, takže neodpovídají skutečnosti.

V lepším případě si toho uživatel všimne ještě předtím, než z těchto výsledků vyvodí nesprávné závěry, které jsou pak zdrojem omylů a dalších chyb. Může se však stát, že v důsledku nedostatečných znalostí a praktických zkušeností uživatele programu jsou výsledky simulace považovány za akceptovatelné.

Často ani nejde o chyby jako takové, jako o nesprávný způsob interpretace výsledků. Opíráme-li se například při simulaci aktivního filtru nebo zesilovače jen o jeho kmitočtovou charakteristiku, můžeme obvod považovat za správně navržený a realizovatelný, i když je evidentně nestabilní. Analýzu „*AC*“ bychom měli doplnit testem na stabilitu, například analýzou časové odezvy obvodu na impuls nebo skok.

b) Simulační program získává výsledky “namáhavě”, analýza trvá relativně dlouho a je náročná na paměť. Přitom se může jednat o relativně jednoduché obvody.

Interní algoritmy programu mají “problémy s konvergencí”. Zde je většinou možné zjednat nápravu, protože na vině bývá uživatel, který se dopustil buď hrubých chyb nebo použil nevhodné postupy při modelování obvodu. Častou příčinou takového chování simulátoru jsou i neoptimální nebo chybná nastavení podmínek simulace.

c) Program nedospěje k řešení. Jeho běh je přerušen chybovým hlášením, pomocí něhož je většinou možné identifikovat interní příčinu takovéhoto chování.

Příčinou mohou být syntaktické nebo další chyby v zadávání obvodu nebo požadovaných cílů analýzy. Program je může odhalit již na úrovni schématického editoru nebo okna “*Analysis Limits*”, tedy ještě před vlastní analýzou. Nejzávažnější chyby však vznikají během analýzy, kdy program není schopen zvládnout vstupní data svými interními matematickými algoritmy. Tyto jevy lze souhrnně označit stejně jako u skupiny b) termínem “problémy s konvergencí”.

Na **obr. 9.111** je dále znázorněno, že všechny uvedené nežádoucí formy chování programu mívají příčiny trojího druhu. Za chyby v zadávání modelu obvodu a podmínek jeho analýzy může uživatel. Závažnou příčinou problémů při analýze však může být samotný model obvodu, pokud po matematické stránce představuje špatně podmíněnou numerickou úlohu. I v těchto případech však existují některé nástroje a postupy, které většinou vedou k cíli.

Na obrázku je naznačeno celkem 8 hlavních vazeb mezi příčinami a důsledky nežádoucího chování simulátoru. Vyjmenujme nejtypičtější "reprezentanty" těchto vazeb. U některých z nich bude přímo popsán způsob překonávání daných problémů. Další metody budou popsány následně.

1. Chyby ve schématu

   - Vodivé spojení místo nevodivého křížení vodičů a naopak (viz část 9.4.3.1, obr. 4.34).
   - Překřížené piny součástek (viz část 9.4.3.1, obr. 4.35).
   - Špatně zadané hodnoty parametrů (*m* místo *meg*, desetinná čárka místo desetinné tečky, mezi číselnou hodnotou parametru a zkratkou inženýrské notace je mezera, např. 1 *k*).
   - Chyby vzniklé nerespektováním řetězení příkazů *.DEFINE* postupným dosazováním (viz část 9.4.3.4).

2. Nastavení podmínek typu „*Operating Point*" a „*State Variables*" neodpovídá charakteru simulační úlohy (viz část 9.5.3.6).

   - Špatně nastavená chybová kritéria v globálních podmínkách simulace („*Global Settings*", viz dále).

3. Selhání optimalizačního algoritmu.

   - Program nalezl nevhodný pracovní bod a provedl AC analýzu v okolí tohoto bodu.
     Tento problém lze překonat použitím příkazu *.NODESET* (viz část 9.5.3.8).
   - Výsledky jsou v důsledku špatně podmíněného matematického modelu zatíženy nepřípustně velkými chybami.

4. Nevhodné modelování, vedoucí zejména u časové analýzy k přílišné redukci výpočetního kroku.

   - Nespojitosti v časových průbězích kapacit nebo vodivostí.
     Příslušné vzorce je nutno upravit tak, aby ke změnám docházelo „pozvolněji".
   - Nulové hodnoty kapacit.
     Je vhodné zajistit, aby se kapacity nikdy nezmenšily na nulové hodnoty.
   - Sériová kombinace diody nebo spínače s induktorem.
     Při snaze o rozpojení obvodu, kterým teče proud induktoru, vzniká v místě rozpojení velké napětí, protože proud induktorem nelze změnit skokem. Program reaguje na prudkou změnu napětí redukcí výpočetního kroku. Diodu, resp. spínač je proto vhodné přemostit odporem, kterým v případě rozpojení bude veden proud z induktoru. Odpor musí být dostatečně velký, aby podstatně neovlivňoval vlastnosti obvodu. Nesmí však být ani příliš velký, aby na něm proud nevyvolal extrémně vysoké napětí. Praktickou hodnotu je třeba vyzkoušet, je možné začít od hodnoty 10 *k*Ω. Tato technika může podstatně urychlit analýzu.
   - Nevhodné hodnoty sériových a paralelních odporů v modelech diodových a tranzistorových struktur.

Souvisí částečně s předchozím bodem. V modelu diody jsou parametry *RL* a *RS* (paralelní a sériový odpor). Pokud je v modelu parametr *RL* roven nule, znamená to ve skutečnosti nekonečnou velikost (zápis typu „*inf*“ není povolen, možné jsou pouze číselné hodnoty). Při problémech s konvergencí je třeba dosadit jinou hodnotu. Můžeme začít od 1 GΩ a případně tuto hodnotu snižovat.
Nulový nebo relativně nízký sériový odpor může být rovněž příčinou zpomalení analýzy, protože tento odpor omezuje exponenciální růst proudu diodou při růstu napětí v propustném směru. Totéž platí o parametrech *RB*, *RE*, *RC* a *RBM* v modelu tranzistoru (viz dokumentace k programu).

5. Neoptimálně nastavené globální podmínky simulátoru (viz dále).
   - V okně „*Analysis Limits*“ jsou nevhodně nastaveny parametry pro řízení výpočetního kroku (viz část 9.5.2.4).

6. Příčiny z bodu 4.
   - Špatná polarita napájecích zdrojů u operačních zesilovačů, případně chybějící tyto zdroje.
   - Text pro symbolické označení uzlu neleží na uzlu a je tedy považován za poznámku (viz část 9.4.3.1, obr. 4.32).
   - Některé uzly nemají stejnosměrnou cestu na zem.
     Pro výpočet pracovního bodu je nutné, aby pro každý uzel existovala stejnosměrně vodivá cesta na zem. Pokud se například v obvodu objeví dva kondenzátory v sérii, je vhodné jejich spojovací uzel propojit na zem rezistorem o dostatečně velkém odporu, kterým neovlivníme parametry obvodu, např. 1 GΩ.
     V menu „*Options/Preferences*“ jsou v složce „*Common Options*“ položky „*DC Path To Ground Check*“ (testování existence stejnosměrných cest na zem) a „*Add DC Path To Ground*“ (přidej stejnosměrnou cestu na zem). Standardně je aktivována jen první z nich. Pokud aktivujeme i druhou, pak se uzly bez této cesty automaticky propojí se zemí odpory o velikostech 1/*GMIN*, kde *GMIN* je parametr z globálních podmínek simulace o standardní hodnotě 10^{-12} *S* pro klasické obvody a 10^{-9} *S* pro výkonové obvody (viz dále).
   - Ve schématu chybí značka uzemnění.
   - Zdroje proudu v sérii a zdroje napětí paralelně.
     Tvoří logický rozpor, „popření“ Kirchhoffových zákonů. Je třeba alespoň jeden zdroj proudu (napětí) doplnit pomocným paralelním (sériovým) odporem.
   - Smyčky, obsahující pouze ideální zdroje napětí a kapacitory.
     Tvoří logický rozpor, „popření“ spojitosti změn elektrického náboje na kapacitoru. Je třeba do smyčky doplnit pomocný odpor.
   - Uzly, k nimž jsou připojeny pouze větve s ideálními zdroji proudu a induktory.
     Tvoří logický rozpor, „popření“ spojitosti změn magnetického indukčního toku induktoru. Je třeba do uzlu doplnit větev s pomocným odporem.
   - Plovoucí uzly (k uzlu je připojena jen jedna součástka).
     Mohou být zdrojem konvergenčních problémů. V menu „*Options/Preferences*“ je vhodné mít v složce „*Common Options*“ zatrženou položku „*Floating Nodes Check*“. U některých součástek, jako jsou například napětím řízené zdroje, program automaticky připojuje jejich „vysokoimpedanční“ piny k zemi přes odpory 1/*GMIN*.

7. Nevhodně nastavené globální podmínky simulace („*Global Settings*“, viz dále).

8. Různé problémy s konvergencí (viz dále).

9.8.2 Problémy s konvergencí vnitřních algoritmů a cesty k jejich řešení

9.8.2.1 „*Global Settings*“ (globální nastavení simulátoru)

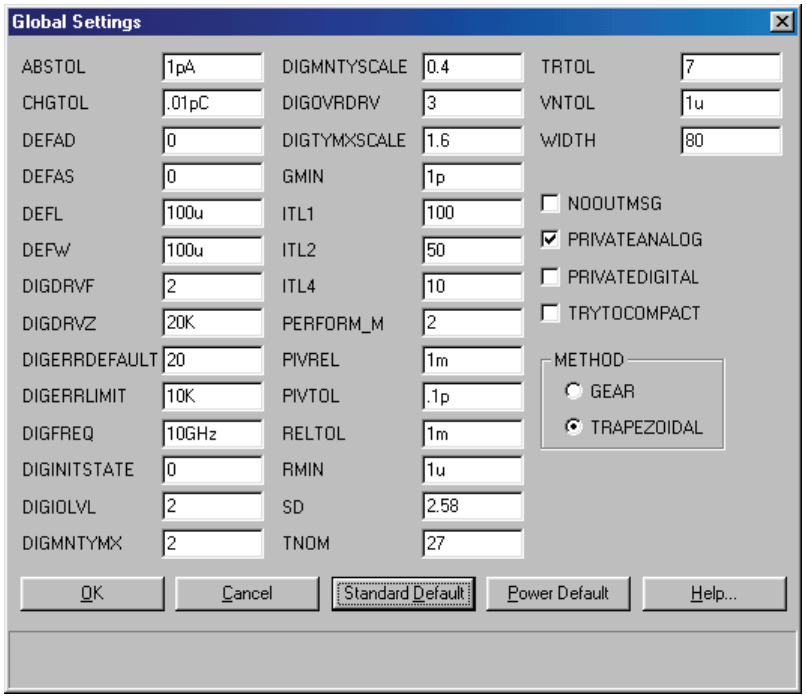
V závislosti na typu analýzy musí být simulační program schopen řešit velmi rozsáhlé soustavy různých typů rovnic. Důležitou součástí řady analýz je výpočet stejnosměrného pracovního bodu, který z hlediska matematického představuje řešení soustav nelineárních rovnic iterační metodou. Iterační řešení se neobejde bez definování řady chybových a dalších kritérií, jejichž úkolem je hledat cestu od počátečního odhadu řešení až k skutečnému řešení soustavy metodou postupného přibližování, a ukončit toto přibližování, jestliže se po sobě jdoucí řešení již liší méně než kolik činí předem zadaná akceptovatelná chyba. Protože je snaha, aby řešení bylo nalezeno vždy a pokud možno co nejrychleji, musela být vyvinuta soustava poměrně složitých konvergenčních kritérií s řadou „ladicích konstant“, které jsou uživateli k dispozici k případné modifikaci v menu „Options/Global Settings“ (globální nastavení) – viz **obr. 9.112**.

Podrobnější popis jednotlivých položek je uveden v dokumentaci programu. Jejich případná modifikace může způsobit podstatnou změnu vlastností simulátoru. V tomto směru stojí za pozornost dvě ikony v spodní části okna:

„**Standard Default**“– nastaví typické hodnoty, optimalizované pro výkon běžných simulačních úloh z oblasti slaboproudé elektrotechniky.

„**Power Default**“ – typické nastavení pro bezproblémové řešení výkonových obvodů, kde napětí a proudy dosahují řádově vyšších hodnot.

Jakékoliv jiné zásahy do globálního nastavení bychom měli provádět jen s vědomím, že víme, co děláme. Následující řádky by v tomto směru mohly být alespoň stručným návodem. Pro vážné zájemce o hlubší znalosti doporučujeme literaturu [3.3].



Obr. 9.112: Okno pro modifikaci globálních podmínek simulace.

Některé položky v globálním nastavení jsou hodnotové (tj. vyjádřené čísly), některé příznakové (platí nebo neplatí). V dosavadním textu jsme se prozatím měli možnost seznámit

s třemi hodnotovými položkami (*GMIN*, *SD*, *TNOM*) a s dvěma příznakovými (*PRIVATEANALOG*, *PRIVATEDIGITAL*).

Přesnost výpočtů je řízena zejména těmito třemi kritérii:

- *ABSTOL* – Absolutní chyba ve výpočtu proudů.
- *VNTOL* – Absolutní chyba ve výpočtu napětí.
- *RELTOL* – Relativní chyba ve výpočtu napětí a proudů.

Hodnoty (*ABSTOL*, *VNTOL*, *RELTOL*) pro nastavení „*Standard Default*“ jsou na obr. 4.163: (1*p*A, 1μV, 1*m*). Pro „*Power Default*“ jsou používány poněkud vyšší hodnoty: (1μA, 1*m*V, 10*m*).

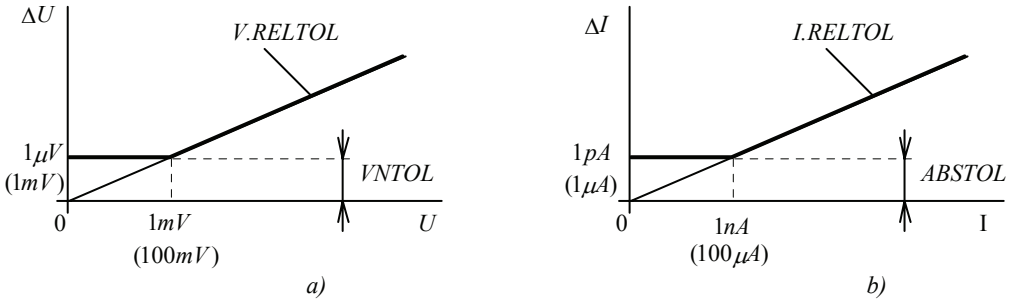
Při iteračním řešení rovnic se uvedená čísla využívají k rozhodování, zda řešení ukončit, následujícím způsobem:

1. V dané iteraci se modifikovanou metodou uzlových napětí vypočtou všechna uzlová napětí plus případné další pomocné veličiny, například proudy singulárními prvky (viz část 2.3.2). Z těchto hodnot se aplikací Ohmova zákona dopočítají proudy všemi větvemi.
2. Pro každé vypočtené napětí U a pro každý vypočtený proud I se určí čísla ΔU a ΔI podle těchto vzorců:

$$\Delta U = MAX(VNTOL, V.RELTOL)\,,\ \Delta I = MAX(ABSTOL, I.RELTOL)\,.$$

Grafické znázornění závislostí veličin ΔU a ΔI na napětí U a proudu I je na **obr. 9.113**. Číselné hodnoty jsou uvedeny pro nastavení „*Standard Default*“, hodnoty v závorkách pro „*Power Default*“.

3. Porovnává se změna každého napětí U a každého proudu I oproti jejich velikostem v předchozím iteračním kroku. Výpočet se zastaví (tj. sada hodnot U a I je prohlášena za řešení), pokud jsou změny všech napětí menší než ΔU a současně změny všech proudů menší než ΔI.



Obr. 9.113: Grafické znázornění významu chybových kritérií *VNTOL*, *RELTOL* a *ABSTOL*.

Jestliže se programu nepodaří nalézt řešení, objeví se příslušné chybové hlášení, například „*Failed to converge in specified number of iterations at time* = 0“ nebo „*Internal time step too small*“.

Z **obr. 9.113** vyplývá, že pokud je při standardním nastavení testované napětí větší než 1 *m*V a testovaný proud větší než 1 *n*A, pak se iterace ukončí, když rozdíl po sobě jdoucích hodnot nepřekročí jednu tisícinu („*RELTOL*“) aktuálního napětí, resp. proudu. Při relativně nízkých hodnotách napětí nebo proudů by však tato chyba vycházela velmi malá (při nulových hodnotách dokonce nulová). Pokud by byla porovnatelná s numerickými chybami

výpočtů, pak bychom se konvergence nemuseli dočkat vůbec. Z těchto důvodů jsou zavedeny absolutní chyby „*VNTOL*“ a „*ABSTOL*“, které při malých hodnotách *U* a *I* reprezentují povolené hranice odchylek Δ*U* a Δ*I* namísto parametru „*RELTOL*“.

Podmínka ukončení iteračního hledání řešení je i tak poměrně přísná. Proto se při řešení komplikovaných obvodů může stát, že v povoleném počtu iterací nedojde k jejímu naplnění. Maximální počet iterací je určen parametry *ITL*1, *ITL*2 a *ITL*4:

*ITL*1 (Standard Default = 100, Power Default = 200):
Určuje maximální počet iterací při počítání stejnosměrného pracovního bodu (inicializace časové analýzy, předstupeň kmitočtové analýzy).

*ITL*2 (Default = 50):
Určuje maximální počet iterací při počítání stejnosměrného pracovního bodu v každém kroku analýzy „*DC*“.

*ITL*4 (Default = 10):
Určuje maximální počet iterací v časové analýze při počítání v každém bodu na časové ose. Pokud program nenalezne řešení, dvakrát se zkrátí časový krok a začíná nanovo iterační hledání v tomto novém bodě. Tento postup se případně opakuje, dokud není nalezeno řešení.

Z uvedeného plyne, že pokud nedojde k nalezení řešení v daném počtu iterací, je možné vyzkoušet buď zvětšení parametrů *RELTOL*, *ABSTOL* a *VNTOL* (zlepšení konvergence na úkor přesnosti výsledků), nebo zvětšit povolený počet iterací (pokusit se o konvergenci zvětšením počtu iterací, tj. na úkor rychlosti výpočtů).

O parametru *GMIN* již byla zmínka v části 3.9.1. Jsou-li v modelech polovodičových diod použity nulové hodnoty paralelních vodivostí, pak parametr *GMIN* tyto nulové hodnoty automaticky nahradí. Zvětšování *GMIN* tedy může rovněž napomoci při problémech s konvergencí obvodů s polovodičovými strukturami.

9.8.2.2 Možné přístupy k řešení problémů s konvergencí

Problémy s konvergencí mohou mít nejrůznější příčiny. Každý takovýto problém je většinou značně „individuální záležitostí“ pro daný konkrétní obvod. Neexistuje tedy žádný univerzální návod, jak donutit program, aby nalezl řešení. Překonávání problémů s konvergencí proto často vykazuje ve větší či menší míře prvky metody pokusu a omylu.

Přesto je možné zformulovat některé obecně platné zásady, kterých bychom se měli držet. Z **obr. 9.111** vyplývá, že problémům s konvergencí, soustředěným v důsledcích b) a zejména c), lze částečně předejít dodržováním zásad, vyjmenovaných v skupinách označených čísly 4 až 6 (jedná se o eliminaci chyb uživatele), 7 (nastavení globálních podmínek simulátoru) a 8 (nejsložitější problém, kdy matematický model analyzovaného obvodu je z principu nesnadno řešitelný numerickými metodami). Zaměříme se tedy na skupiny č. 7 a 8. Mezi typické představitele obvodů, kdy při hledání stejnosměrného pracovního bodu může být simulátor „bezradný“, patří tranzistorové klopné obvody.

Možný postup při odstraňování problémů s konvergencí:

První fáze:

Ověřit, zda jsou dodrženy zásady popsané výše v částech označených čísly 1 *až* 6.

Druhá fáze:

Zkontrolovat, je-li aktivní volba „Convergence Assist“ v menu „Options/Preference“ v složce „Common Options“, skupině „Analysis“.

Pak se program snaží v případě problémů nalézt sám optimální řešení hledáním několika kombinací parametrů v „*Global Settings*“. V případě úspěchu definuje tyto parametry příkazem .*OPTIONS*, který umístí na pracovní plochu editoru. Nastavení těchto parametrů příkazem .*OPTIONS* má pak prioritu před nastavením v „*Global Settings*“.

Třetí fáze:

„Ruční“ zásahy do „Global Settings“:

- Zvětšení parametru *GMIN* o 1 až dva řády.
- Zvětšení *RELTOL* o 1 řád.
- Zvětšení *ABSTOL* nebo *VNTOL* v případě, že v analyzovaném obvodu působí napětí řádově větší než desítky voltů nebo proudy řádově větší než desítky *m*A.
- Zvětšení *ITL*1 postupně na hodnoty 200 a v případě neúspěchu na 300. Pokud to nepomůže, vrátit se na hodnotu 100, případně 200.
- Zvětšení *ITL*2, pokud řešení nekonverguje při analýze „*DC“*, postupně z původní hodnoty 50 až na 200-300.
- Zvětšení *ITL*4, jestliže jsou problémy s konvergencí během časové analýzy, z původních 10 na 30-50.

Čtvrtá fáze aneb „každá rada drahá, neboť osvědčené postupy selhaly“:

Podíváme se do výstupního textového souboru („*Numeric Output File*“), jehož prohlížení aktivujeme horkou klávesou *F*5. Pokusíme se v něm najít informaci o součástce, která je zdrojem poruchy konvergence.

Pokud je touto součástkou polovodičový aktivní prvek (dioda, tranzistor), aplikujeme **metodu klíčového slova *OFF***: toto slovo napíšeme do políčka „*Value*“ v okně atributů součástky (okno vyvoláme poklepáním na schématickou značku v režimu „*Select*“). Tímto způsobem zajistíme, že při první iteraci při hledání pracovního bodu není součástka v obvodu uvažována. Jedná se o velmi účinnou techniku zlepšení konvergence. Její účinek ještě vzroste, doplníme-li slovo *OFF* o definici tzv. počátečních podmínek aktivního prvku. U tranzistoru jsou to napětí báze-emitor a kolektor-emitor, u diody napětí anoda-katoda. Pak se v prvním iteračním kroku uvažují mezi příslušnými uzly daná napětí. Je ovšem třeba dobře odhadnout jejich velikosti v počítaném pracovním bodu. Uvažujeme-li například odhady *UBE*=0,7 V a *UCE*=0,2 V, pak do pole „*Value*“ tranzistoru zapíšeme *OFF IC* 0.7 0.2.

Speciální techniky, jak nalézt stejnosměrný pracovní bod na začátku analýzy „*Transient*“ nebo „*DC*“:

(*Hledání tohoto bodu může být u některých obvodů problém: obecně existuje větší počet pracovních bodů a simulátor nemusí najít zrovna ten, který nás zajímá. V některých případech nemusí být nalezen žádný pracovní bod*).

Základní postup při analýze „problematických“ obvodů:

- Budicí signály nahradíme konstantními signály, „prodloužením“ jejich počátečních hodnot (analýza „*Transient*“), nebo jejich stejnosměrnými složkami (analýza „*AC*“).
- V menu „*Transient Analysis Limits*“ zakážeme výpočet pracovního bodu („*Operating Point*“). Poté spustíme časovou analýzu z nulových počátečních podmínek („*State Variables*“ – *Zero*).
- Necháme proběhnout analýzu až do stejnosměrného ustáleného stavu. Pokud je tohoto stavu dosaženo, pak uložíme souřadnice stavových proměnných do souboru jako souřadnice nalezeného pracovního bodu (v „*State Variable Editoru*“ –viz část 9.5.3.7).
- Poté již můžeme studovat odezvy na původní signály v režimu „*State Variables*“ – *Read* při vyblokovaném výpočtu pracovního bodu.

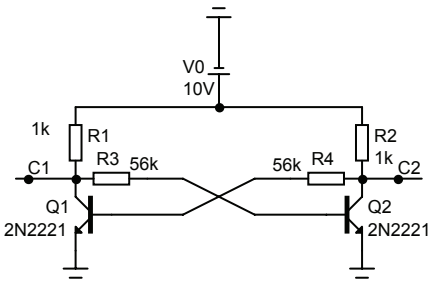
Poznamenejme, že uložený pracovní bod můžeme využít i v analýze „*AC*“, pokud ji aktivujeme v režimu „*State Variables*“ – *Read* a zakážeme výpočet pracovního bodu.

V některých ojedinělých případech obvod dospěje do jiného pracovního bodu, než o který máme zájem. V tom případě si můžeme pomoci příkazem *.NODESET* (viz část 9.5.3.8), pomocí něhož dodáme iterační proceduře první odhad souřadnic hledaného pracovního bodu.

U zvlášť problematických obvodů program nenalezne pracovní bod ani po použití uvedených postupů. Pak zkusíme následující: všechny stejnosměrné zdroje nahradíme signály ze zdrojů impulsů. Modelujeme postupný lineární nárůst z nulových počátečních hodnot až do nominálních hodnot. Opět je třeba deaktivovat výpočet pracovního bodu. Program by měl nalézt stejnosměrný ustálený stav „snadněji“ než při původním modelování skokových změn stejnosměrných zdrojů v počátku simulace.

9.8.2.3 Příklad analýzy tranzistorového bistabilního klopného obvodu

Analyzujme bistabilní klopný obvod na **obr. 9.114**, jehož zadání nalezneme v dodatkovém souboru **BISTAB.CIR**.



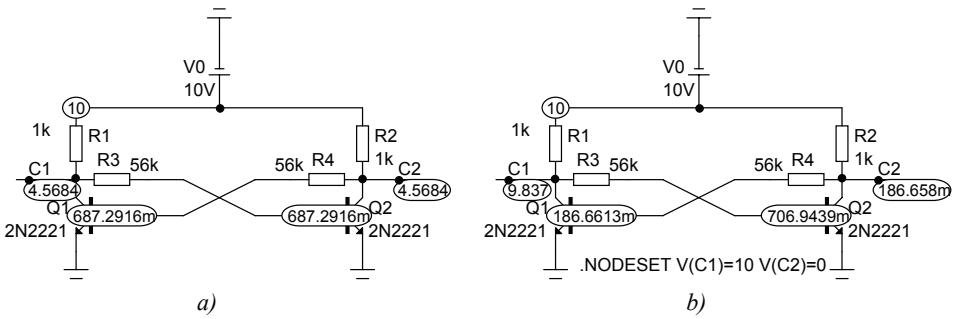
Obr. 9.114: Tranzistorový bistabilní klopný obvod.

Tento obvod vykazuje stoprocentní symetrii vzhledem k myšlené ose, procházející svisle baterií *V*0. Je otázka, jaký stejnosměrný pracovní bod program vyhledá. Podle teorie existují v obvodu tři různé pracovní body:

1. Při *Q*1 otevřeném a *Q*2 uzavřeném.
2. Při *Q*1 uzavřeném a *Q*2 otevřeném.
3. Při *Q*1 a *Q*2 zčásti otevřenými.

První dva pracovní body popisují stabilní stavy v bistabilním klopném obvodu. Jeden z nich vždy nastane po připojení obvodu k napájecímu zdroji. Třetí pracovní bod je v praxi nestabilní. Příslušný rovnovážný stav by se mohl v obvodu udržet jen teoreticky při nepůsobení poruch.

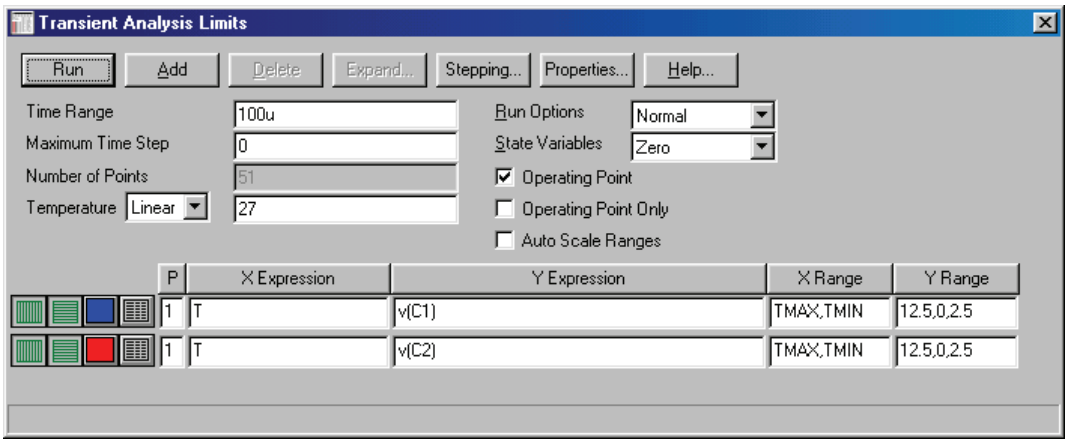
Po provedení dynamické „*DC*“ analýzy obdržíme výsledem na obr. 4.166 a). Je zřejmé, že byl nalezen právě nestabilní pracovní bod. Pokud chceme nalézt jeden ze stabilních stejnosměrných stavů, například při *Q*1 uzavřeném a *Q*2 otevřeném, doplníme do schématu *grid text* podle **obr. 9.115** b). Příkazem *.NODESET* nastavíme podmínky první iterace hledání pracovního bodu tak, jako kdyby tranzistor *Q*1 byl úplně uzavřen (*V*(*C*1)=10V) a tranzistor *Q*2 úplně otevřen (*V*(*C*2)=0).



Obr. 9.115: Výsledky dynamické „*DC*“ analýzy: a) nalezen nestabilní pracovní bod, b) díky příkazu *.NODESET* nalezen stabilní pracovní bod.

Obdobného výsledku bychom dosáhli, kdybychom tranzistoru *Q*1 definovali atribut „*Value*“ na „*OFF*“. Tím by program před první iterací neuvažoval přítomnost tranzistoru *Q*1, jinými slovy, zavedli bychom do obvodu takovou počáteční nesymetrii, která by způsobila konvergenci do stavu, kdy tranzistor *Q*2 je otevřen a *Q*1 zavřen.

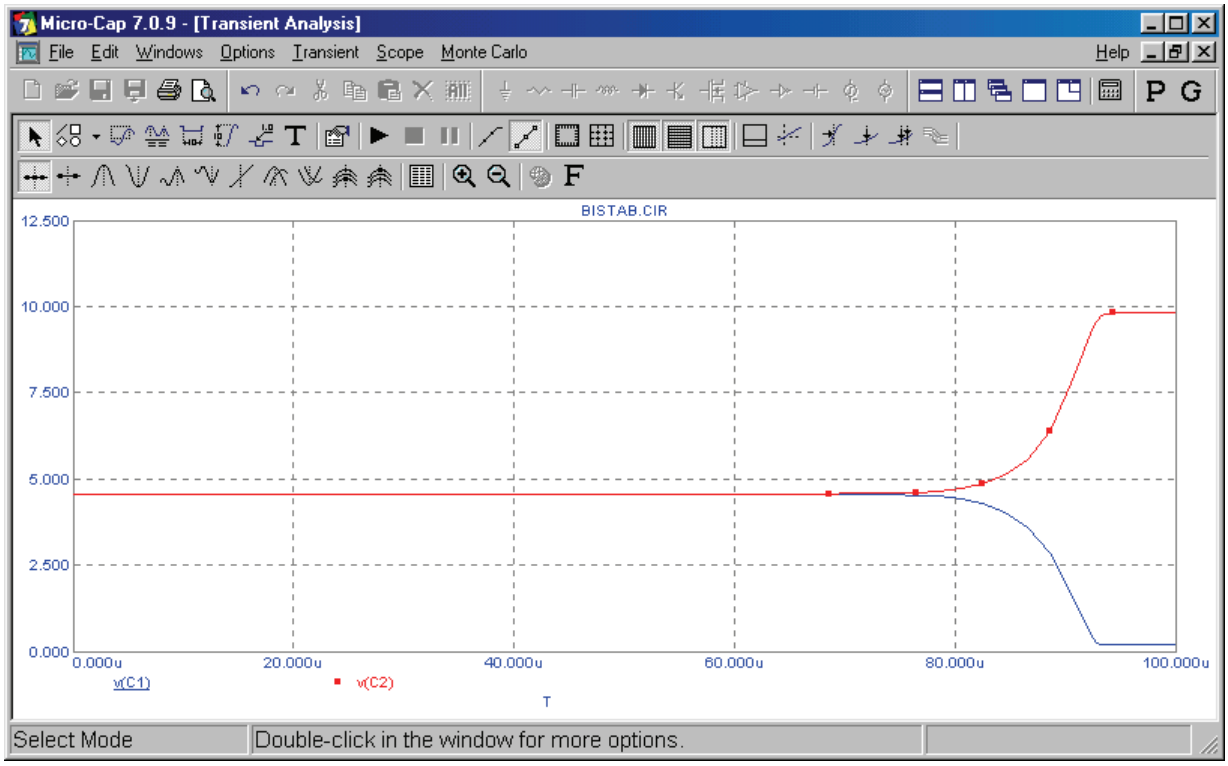
Odstraňme příkaz *.NODESET*. Nyní se podívejme, jak se bude simulátor chovat při analýze „*Transient*“. Okno „*Transient Analysis Limits*“ vyplníme tak, jak je ukázáno na **obr. 9.116**.



Obr. 9.116: Nastavení podmínek časové analýzy pro klopný obvod z **obr. 9.115**.

Analýzou zjistíme průběhy napětí na kolektorech obou tranzistorů podle **obr. 9.117**. Je zřejmé, že program nejprve nalezl nestabilní stejnosměrný pracovní bod (máme zatrženou položku „*Operating Point*“), a po poměrně dlouhé době časovou analýzou dospěl k jednomu ze stabilních pracovních bodů. Stačí k tomu sebemenší nesymetrie v obvodu, která je zde zastoupena působením numerických chyb. Můžete si vyzkoušet čtyři zajímavé věci:

1. Vyblokujte výpočet pracovního bodu a spusťte analýzu znovu při nulových počátečních podmínkách. Řešení přechodného děje vede nejprve k nestabilnímu pracovnímu bodu, a posléze dojde opět k přechodu do stabilního stavu.
2. Snižte velikost kolektorového odporu $R1$ z 1 $k\Omega$ na 999 Ω. Tím zavedete do obvodu mírnou nesymetrii. Náběh do stabilního pracovního bodu bude nyní o něco rychlejší.
3. Vyzkoušejte časovou analýzu při povoleném výpočtu pracovního bodu a při současném působení příkazu *.NODESET* podle **obr. 9.115** b).
4. Vyzkoušejte časovou analýzu při nepovoleném výpočtu pracovního bodu a při současném působení příkazu *.IC* $V(C1)=10$ $V(C2)=0$. Snažte se porozumět rozdílnému chování simulátoru oproti bodu 3 (viz rozdíly mezi příkazy *.NODESET* a *.IC*, část 9.5.3.8).

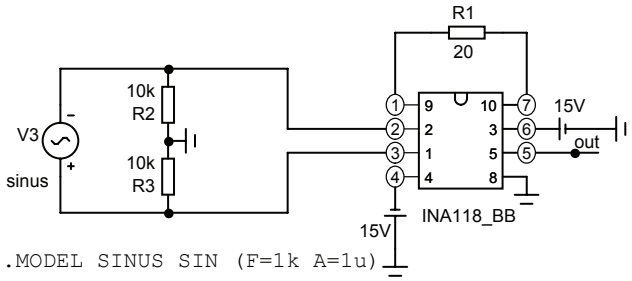


Obr. 9.117: Přechod klopného obvodu z nestabilního do stabilního pracovního bodu.

9.8.2.4 Příklad analýzy obvodu s nesnadno zjistitelným pracovním bodem

V předchozím příkladu byl ukázán obvod, u něhož program neměl problém nalézt pracovní bod, ovšem museli jsme učinit pomocná opatření, aby našel stabilní a tudíž v reálném obvodu pozorovatelné stavy. Nyní ukážeme, že existují obvody, u nichž se simulačnímu programu za normálních okolností nemusí podařit nalézt pracovní bod žádný.

Na **obr. 9.118** je zapojení zesilovače, převzaté z [3.15]. Zadání naleznete v dodatkovém souboru **INA118.CIR**.



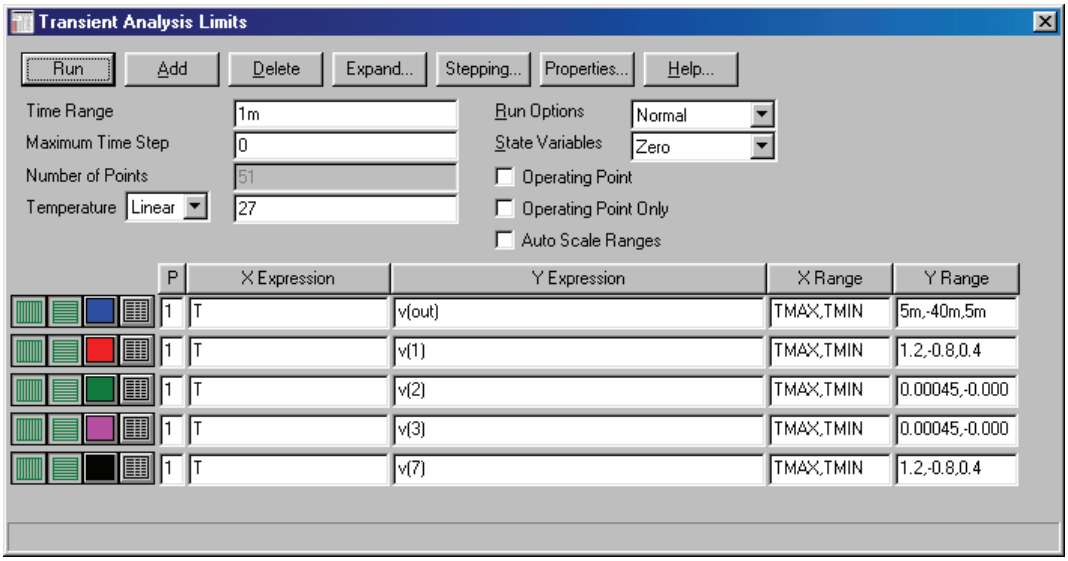
Obr. 9.118: Zesilovač, u něhož je zjištění stejnosměrného pracovního bodu klasickým postupem prakticky nemožné.

*INA*118 je přesný, nízkopříkonový přístrojový operační zesilovač firmy *Burr-Brown*. Napěťové zesílení obvodu na **obr. 9.118** je dáno vzorcem $1+50k/R1=2501$ [3.15].

Pokusy o dynamickou „*DC*“ analýzu, analýzu „*Transient*“ i „*AC*“ vedou na chybová hlášení

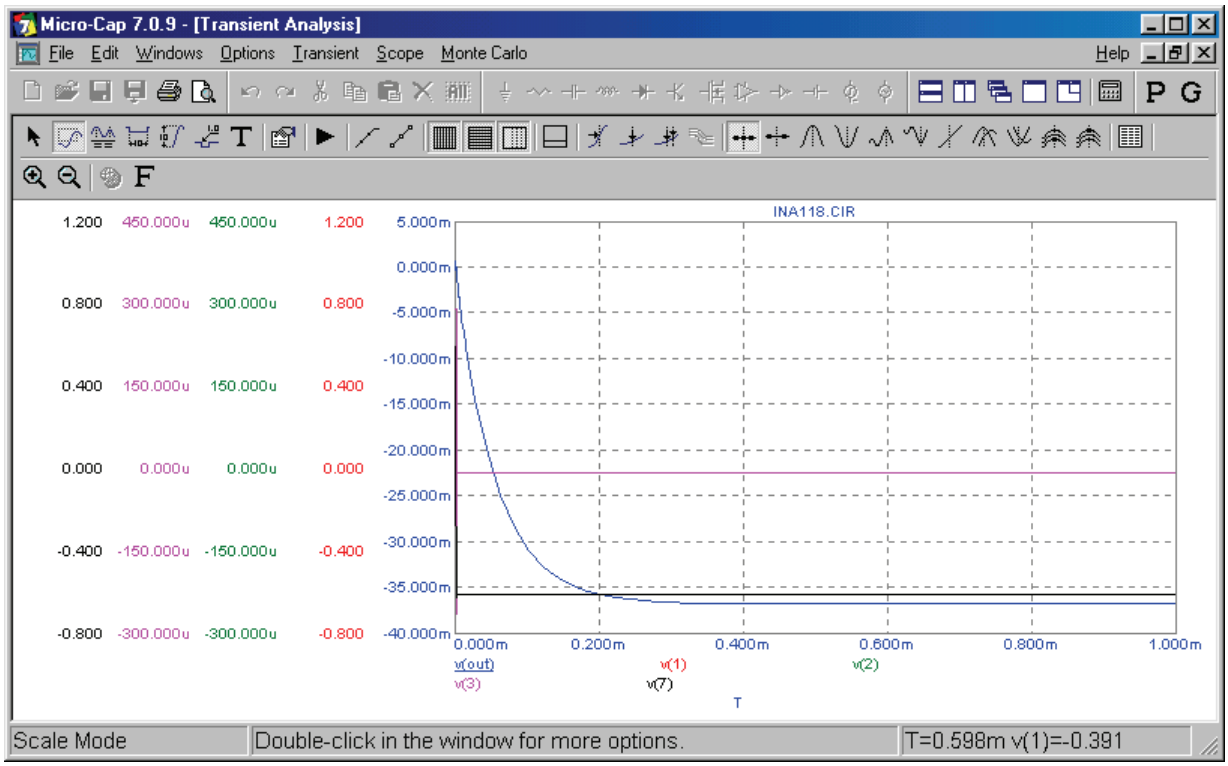
„Failed to converge in specified number of iterations at time = 0“.

Pokusíme se tedy nejprve určit stejnosměrný pracovní bod řešením přechodného děje po napojení obvodu na napájecí zdroje při nepůsobícím signálovém zdroji. Parametry modelu zdroje *V*3 upravíme tak, aby poskytoval nulové napětí (*F*=0 nebo *A*=0 nebo obojí). Pak spustíme analýzu „*Transient*“ s parametry podle **obr. 9.119**. Výsledek je na **obr. 9.120**.

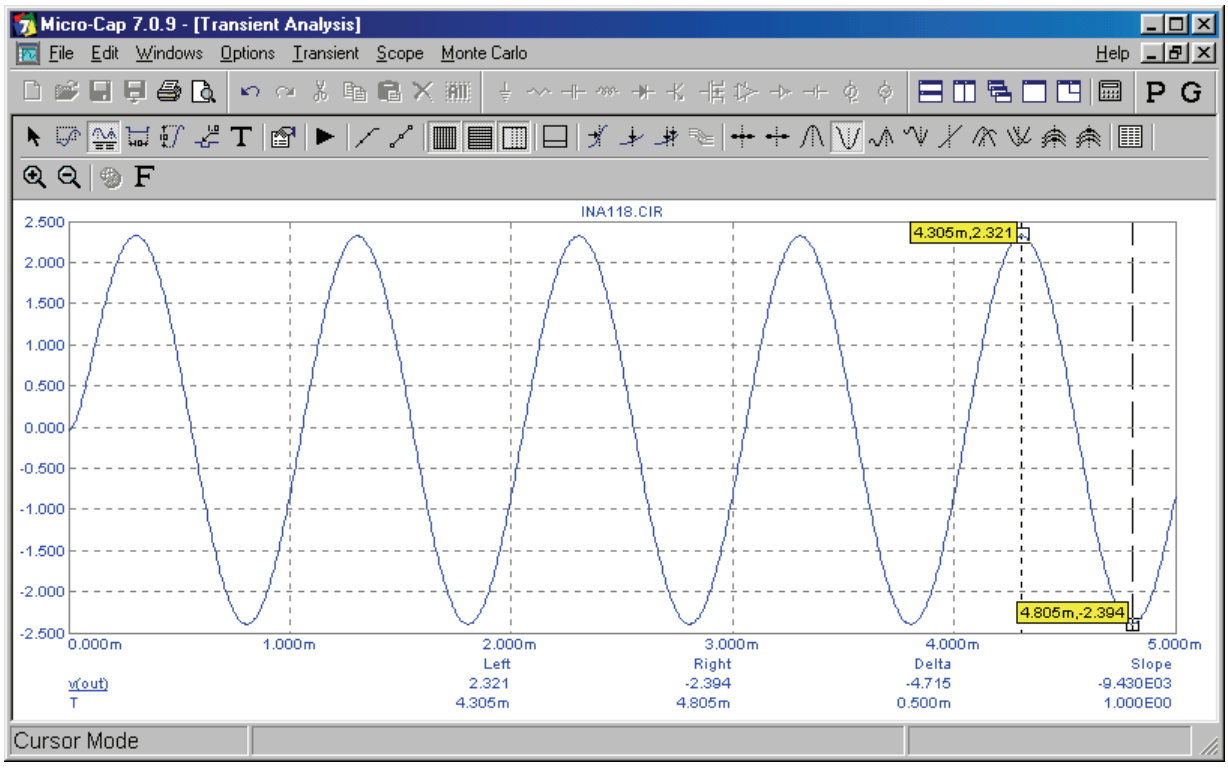


Obr. 9.119: Podmínky pro analýzu „*Transient*“: vyblokovaný výpočet stejnosměrného pracovního bodu, nulové počáteční podmínky.

Grafy jsou vykresleny při neaktivní položce „*Scope/Same Y Scales*”, takže každá křivka v obrázku má svou vlastní *Y*-osu. Z obrázku je zřejmé, že za dobu 1 *m*s již došlo k ustálení vnitřních přechodných dějů a že výsledný stav obvodu je možné považovat za stejnosměrný pracovní bod. Stavové proměnné tedy uložíme do souboru volbou „*Transient/State Variables Editor/Write*”. Soubor **INA118.TOP** uložíme do adresáře, v němž je výchozí soubor **INA118.CIR**. Nyní spustíme analýzu „*Transient*“ znovu, ale v režimu „*State Variables-Read*“ a při neaktivním výpočtu pracovního bodu.



Obr. 9.120: Výsledek časové analýzy obvodu z **obr. 9.118** – jeho přechod do stejnosměrného ustáleného stavu – stejnosměrného pracovního bodu.

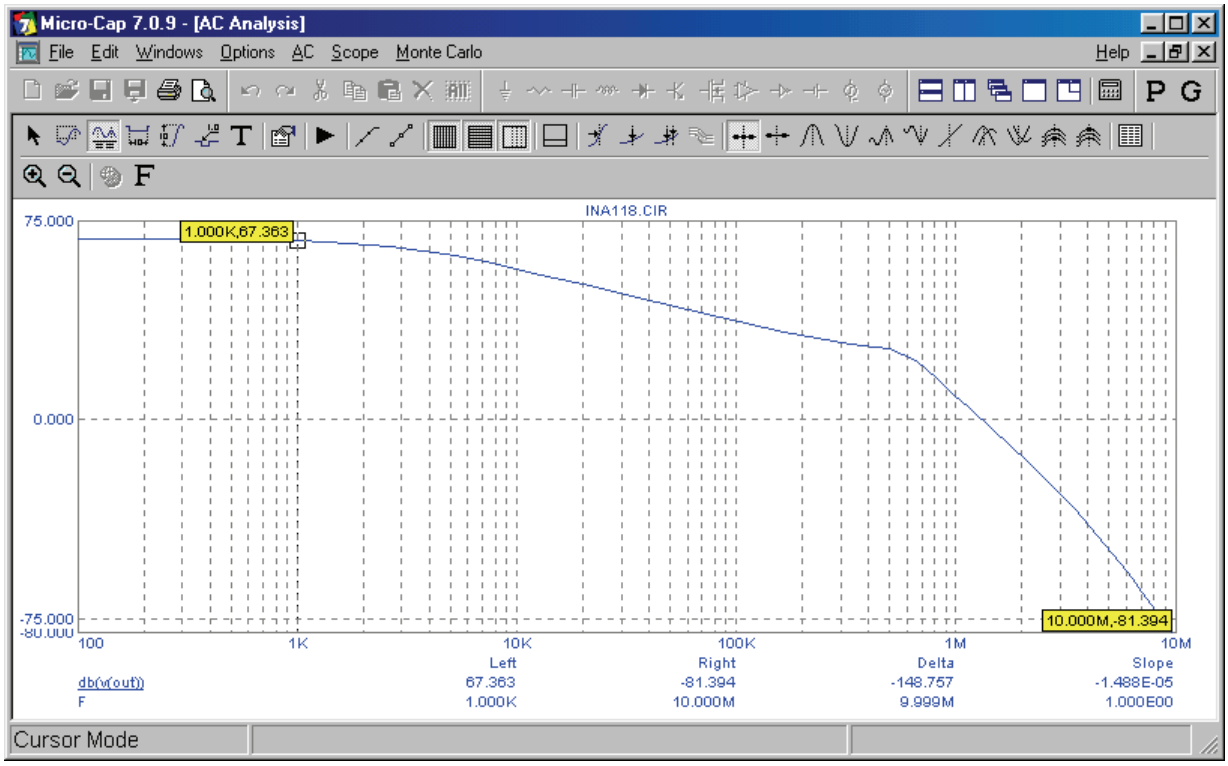


Obr. 9.121: Výstupní napětí zesilovače získané na základě „importovaného“ pracovního bodu.

Ukázka výsledku – křivky výstupního napětí po dobu prvních pěti milisekund – je na **obr. 9.121**. Po tuto dobu již počáteční přechodný děj prakticky odezněl a výstupní napětí má mezivrcholovou hodnotu asi 2,321+2,394 = 4,715 V, čemuž odpovídá amplituda 2,3575 V.

Při amplitudě vstupního napětí 1 *m*V to představuje zesílení 2357,5, tj. asi 67,4 dB na kmitočtu 1 *k*Hz.

Nakonec provedeme analýzu amplitudové kmitočtové charakteristiky. Analýzu „*AC*“ je třeba provést za podmínek „*State Variables-Read*“ a neaktivní položce „*Operating Point*“. Výsledek je na **obr. 9.122**. Na kmitočtu 1 *k*Hz je možné odečíst zesílení 67,363 dB.



Obr. 9.122: Kmitočtová charakteristika zesilovače z **obr. 9.118**, získaná na základě „importovaného“ pracovního bodu.

Na závěr si všimněte zajímavé věci. Nejprve se přesvědčete o tom, zda v okně „*Transient Analysis Limits*“ je nastaveno „*State Variables-Read*“. Pak aktivujte dynamickou „*DC*“ analýzu. Nalezení stejnosměrného pracovního bodu nyní proběhne bez problémů. Dynamická „*DC*“ analýza je tedy napojena na nastavení analýzy „*Transient*“: Je-li zde nastaveno „*State Variables-Read*“, přečtou se souřadnice pracovního bodu ze souboru a promítnou se do schématu. Při změnách parametrů součástek slouží tyto souřadnice jako výchozí pro další iterace. Tato vlastnost programu není uvedena v dokumentaci MicroCapu.

10 Místo závěru

Metody analýzy elektrických obvodů, ať už jsou vykonávány „ručně“ nebo pomocí počítače, byly vyvinuty zejména pro ty z nás, kterým přináší radost a uspokojení objevovat dosud nepoznané, v našem případě hledat vlastnosti a předvídat chování elektrických systémů na základě znalosti jejich modelů. Věrohodnost získaných výsledků silně závisí na věrohodnosti výchozích modelů a na korektnosti postupů od modelu ke konečným výsledkům. Na různých místech knihy je ukazováno, že oba uvedené faktory je třeba mít neustále pod kontrolou, a to jak při ručních výpočetních postupech, tak zejména při použití simulačních programů. Pokud nerozumíme detailně funkci analyzovaného obvodu, měli bychom se alespoň snažit jeho chování lépe porozumět právě prostřednictvím kritického vyhodnocování výsledků analýzy. Nekritické přebírání a interpretace výsledků sebedokonalejšího simulačního programu může vést k fatálním důsledkům. Místo závěru tedy doporučení:

Mysleme hravým, tvůrčím a hledajícím způsobem. Při analýze obvodů existují jediná „dogmata“: základní zákony a principy teoretické elektrotechniky. Kromě nich nejsme svázáni ničím jiným, nežli nedostatečností dvojího: naší zkušenosti + naší schopnosti analytického myšlení. Výsledky analýzy nepřebírejme automaticky, ale konfrontujme je s naším úsudkem. Chyby i úspěchy budou formovat naši zkušenost a poučí nás o míře respektu k fyzikální podstatě problému, který by měl být v každém okamžiku nadřazen našemu respektu k sebedokonalejšímu nástroji analýzy. Složitý simulační program se může stát v rukou neznalého uživatele bezcennou hračkou. Potřebujete-li simulační program ke své práci, udělejte všechno pro to, abyste se nestali jeho neznalým uživatelem. Tato kniha vám v tomto směru podává pomocnou ruku. Ten větší kus cesty už ale budete muset dojít sami.

11 Přílohy

11.1 Vyjadřování čísel v programech SNAP a MicroCap

V zásadě se nerozlišují malá a velká písmena.

Vyjádření čísel v pevné řádové čárce: 1.287, -27.1, 8

Vyjádření čísel v pohyblivé řádové čárce: 1e-12, -5.8E8, 32.11e19 (nerozlišují se malé a velké *e*)

Inženýrská notace: 5K, 8m, 3.3MEG,... (nerozlišují se malá a velká písmena)

zkratka	*označení*	*číselná hodnota*
F	Femto	1E-15
P	Pico	1E-12
N	Nano	1E-9
U	Mikro	1E-6
M	Mili	1E-3
K	Kilo	1E3
MEG	Mega	1E6
G	Giga	1E9
T	Tera	1E12

Tab. 11.1: Tabulka inženýrské notace.

11.2 Vybrané prvky programu MicroCap

11.2.1 Napájecí a signálové zdroje, zdroje pro transformaci signálů

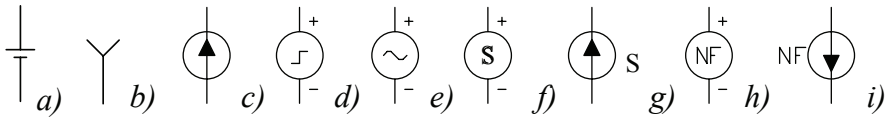
Zdroje MicroCapu je možné rozdělit do dvou kategorií:

a) Generátory signálů (baterie, „*Fixed Analog*“, zdroj proudu, zdroj impulsů, zdroj harmonického signálu, univerzální zdroje napětí („*V*“) a proudu („*I*“), uživatelské zdroje, funkční zdroje napětí *NFV* a proudu *NFI*.

b) Obvody pro transformaci signálů, tj. obvody, na jejichž vstupy působí signály a obvod je převádí na výstupní signály podle definované závislosti (funkční zdroje typu *NT*, Laplaceovy zdroje typu *LF* a *LT*, závislé zdroje klasické a zdroje typu *E*, *F*, *G* a *H*).

Níže uvedeme popis nejčastěji používaných zdrojů. Ostatní zdroje je možné nastudovat z dokumentace programu. V části P10.1.3 je přehledně uvedeno, jak se uplatňují konkrétní atributy zdrojů při různých typech analýz.

11.2.1.1 Generátory signálů



Obr. 11.1: Schématické značky generátorů signálů: a) baterie, b) „*Fixed Analog*“, c) zdroj stejnosměrného proudu, d) zdroj napěťových impulsů, e) zdroj harmonického napětí, f) univerzální zdroj napětí *V* (*SPICE*), g) univerzální zdroj proudu *I* (*SPICE*), h) funkční zdroj napětí, i) funkční zdroj proudu.

Baterie
(„*Components/Analog Primitives/Waveform Sources/Battery*“), obr. a)

je definována pouze hodnotou svého napětí ve voltech v poli „*Value*“. Nejedná se o model SPICE.

„**Fixed Analog**“
(„*Components/Analog Primitives/Waveform Sources/Fixed Analog*“), obr. b)

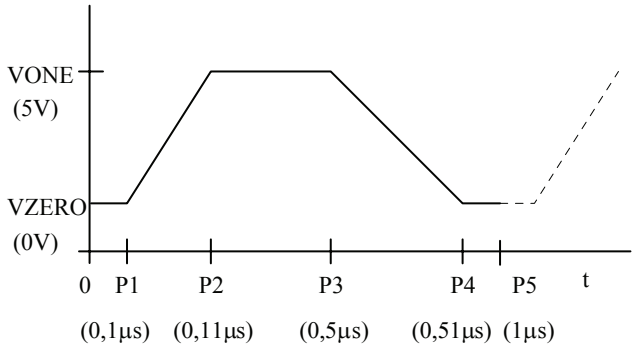
nahrazuje baterii s uzemněním. Tato součástka má jediný pin a jediný parametr v poli „*Value*“, napětí mezi tímto pinem a referenčním uzlem. Nejedná se o model SPICE.

Zdroj proudu
(„*Components/Analog Primitives/Waveform Sources/I Source*“), obr. c)

je definován jedinou hodnotou – stejnosměrným proudem v ampérech v poli „*Value*“. Nejedná se o model SPICE.

Zdroj impulsů
(„*Components/Analog Primitives/Waveform Sources/Pulse Source*“), obr. d)

je zdroj napětí o nulovém vnitřním odporu. K definici je použit příkaz *.MODEL* o parametrech *VZERO*, *VONE*, *P*1, *P*2, *P*3, *P*4 a *P*5, jejichž význam je zřejmý z **obr. 11.2**. Nejde o model SPICE.



Obr. 11.2: Definice parametrů modelu zdroje impulsů („*Pulse Source*“).

Zdroj harmonického signálu
(„*Components/Analog Primitives/Waveform Sources/Sine Source*“), obr. e).

Je to zdroj napětí. K definici je použit příkaz *.MODEL*. Zdroj je definován níže uvedenými parametry. Nejedná se o model SPICE.

Název	*Parametr*	*Jednotka*	*Přednastavená hodnota*
F	kmitočet	Hz	1e6
A	amplituda	V	1
DC	stejnosměrné posunutí	V	0
PH	počáteční fáze	radiány	0
RS	vnitřní odpor zdroje	Ω	.001
RP	opak. perioda exponenciálního tlumení	s	0
TAU	časová konstanta exponenciálního tlumení	s	0

Časový průběh napětí zdroje je popsán těmito rovnicemi:

Když TAU = 0:

V=A*sin(2*PI*F*TIME+PH)+DC.

Když TAU ≠ 0:

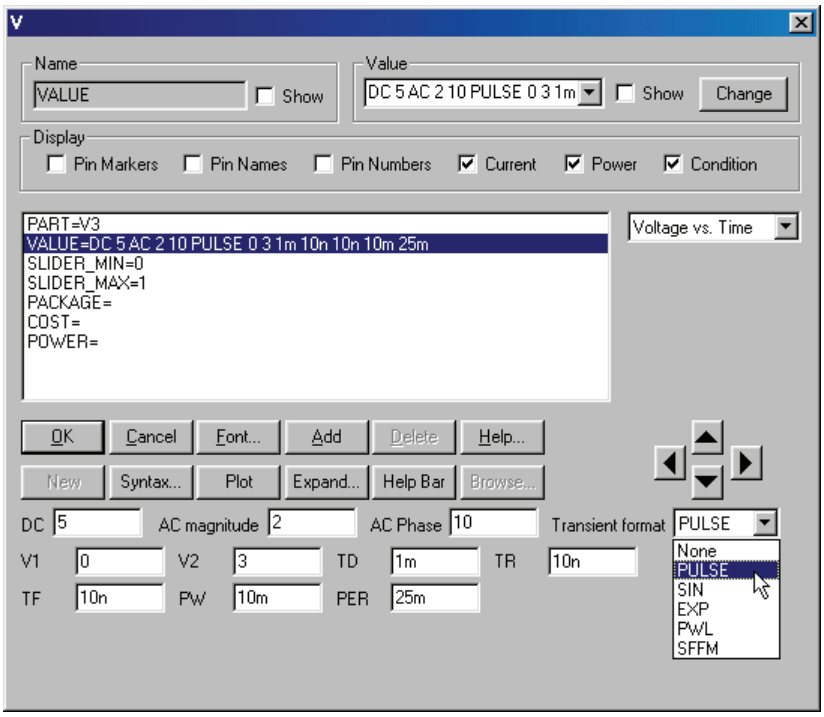
V=A*exp(-T/TAU)*sin(2*PI*F*TIME+PH)+DC,

kde T = TIME modulo RP, jinými slovy: když simulační čas TIME je menší než RP, pak T = TIME. Pokud překročí hodnotu RP, pak T = TIME-RP, atd.

Časová konstanta TAU se uplatní při modelování jedině tehdy, definujeme-li současně nenulovou (kladnou) hodnotu RP.

Univerzální zdroje napětí a proudu
(„*Components/Analog Primitives/Waveform Sources/V*“, „*Components/Analog Primitives/Waveform Sources/I*“), obr. f) a g).

Modely těchto zdrojů jsou převzaty ze SPICE. Při jejich definování se nevyužívá příkaz *.MODEL*.



Obr. 11.3: Editační okno univerzálního zdroje napětí „*V*“.

Na **obr. 11.3** je ukázka okna atributů zdroje „*V*“. Prakticky stejné okno má zdroj „*I*“.

Atributy „DC“, „AC magnitude“, „AC Phase“

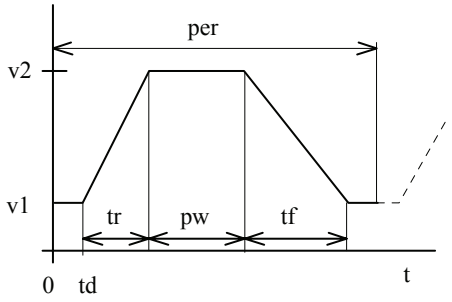
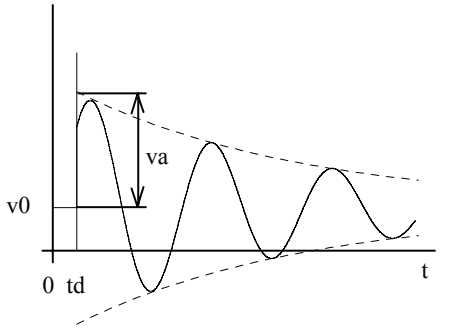
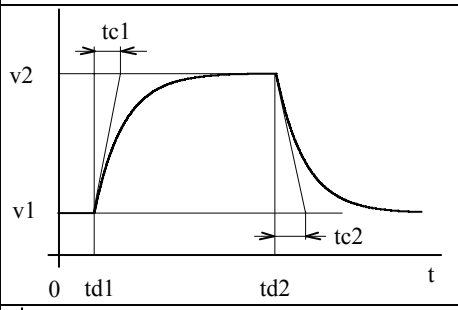
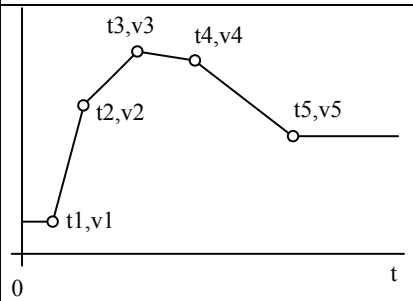
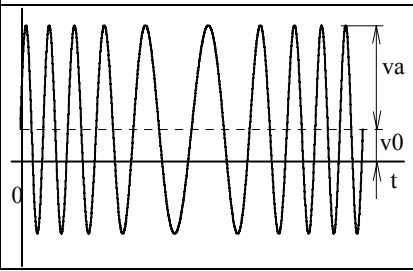
- Atribut „*DC*“ se může uplatnit při výpočtu pracovního bodu v analýzách „*Dynamic DC*“, „Transient, „*AC*“ a „*DC*“.
- Atributy „*AC magnitude*“ (amplituda) a „*AC Phase*“ (počáteční fáze) bude zdroj vykazovat při malosignálové analýze „*AC*“, kdy je nahrazován zdrojem harmonického signálu. Detailně je význam těchto atributů popsán v příloze 11.2.1.3.

Atributy časových průběhů

Závisí na typu průběhu („*Transient format*“). Ten může být následující:

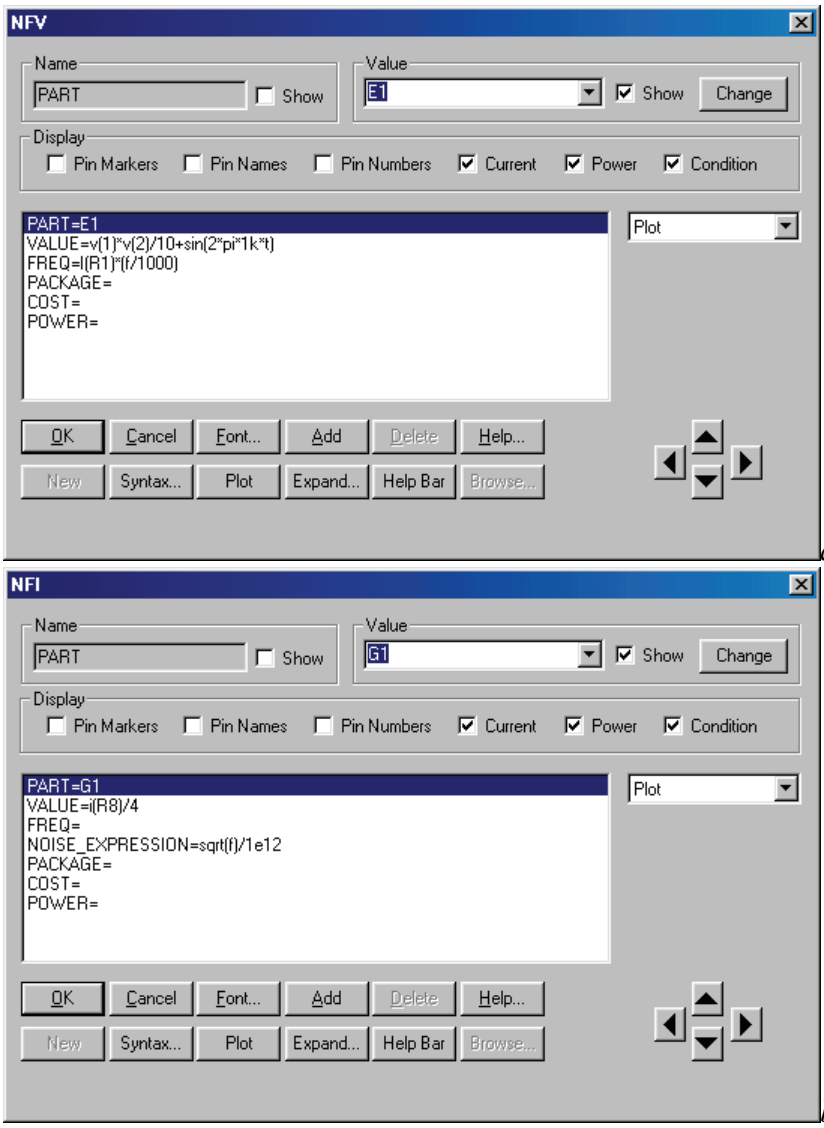
None	Průběh není vybrán. Je uvažován konstantní signál o velikosti *DC*.
PULSE	Impulsní signál.
SIN	Zobecněný harmonický signál (s možností zpoždění a exponenciálního tlumení).
EXP	„Impulsní“ signál s exponenciálně tvarovanými „hranami“.
PWL	Signál složený z po částech lineárních úseků („*Piece-Wise Linear*“).
SFFM	Harmonický signál, kmitočtově modulovaný jiným harmonickým signálem („Single-Frequency *FM*“).

Význam atributů jednotlivých formátů je zřejmý z tabulky 11.2.

formát	atributy				časový průběh
PULSE	název	popis	jednotka	přednast.	
	v1 (i1)	poč. hodnota	V (A)	-	
	v2 (i2)	výška impulsu	V (A)	-	
	td	zpoždění	s	Ts	
	tr	doba náběhu	s	Ts	
	tf	doba doběhu	s	0	
	pw	šířka impulsu	s	0,96 MAX	
	per	perioda	s	TMAX	
	Ts = 1/50 TMAX				
SIN	název	popis	jednotka	přednast.	
	v0 (i0)	posunutí	V (A)	-	
	va (ia)	amplituda	V (A)	-	
	freq	kmitočet	Hz	1/TMAX	
	td	zpoždění	s	0	
	df	tlumicí faktor	1/s	0	
	phase	poč. fáze	stupně	0	
	df = 1/časová konstanta tlumení nulová poč. fáze znamená sinusový signál.				
EXP	název	popis	jednotka	přednast.	
	v1 (i1)	poč. hodnota	V (A)	-	
	v2 (i2)	špičk. hodnota	V (A)	-	
	td1	začátek náběhu	Hz	0	
	tc1	τ náběhu	s	Ts	
	td2	konec náběhu	s	td1+Ts	
	tc2	τ doběhu	s	Ts	
	Ts = 1/50 TMAX				
PWL	Časový průběh signálu je tvořen lomenou čarou, která spojuje body o zadaných souřadnicích. Signál „vlevo“ od prvního bodu a „vpravo“ od posledního bodu na časové ose je prodloužením vertikálních souřadnic krajních bodů.				
SFFM	název	popis	jednotka	přednast.	
	v0 (i0)	posunutí	V (A)	-	
	va (ia)	amplituda	V (A)	-	
	freq	kmitočet nosné	Hz	1/TMAX	
	mi	index FM=ΔF/fm	-	0	
	fm	modulační kmit.	Hz	1/TMAX	
	ΔF je kmitočtový zdvih.				

Tab. 11.2: Význam atributů časových průběhů univerzálních zdrojů napětí a proudu. *TMAX* je položka v okně „*Time Range*“ okna „*Transient Analysis Limits*“.

Funkční zdroje napětí a proudu typu „NF“
(„*Components/Analog Primitives/Function Sources/NFV*“,
„*Components/Analog Primitives/Function Sources/NFI*“), obr. h) a i).

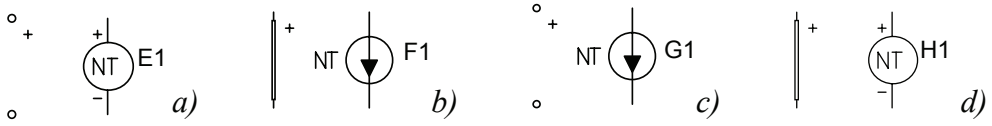


Obr. 11.4: Editační okno funkčního zdroje a) napětí, b) proudu.

Okna atributů těchto zdrojů jsou na **obr. 11.4**. Hodnota napětí, resp. proudu může být definována v položce „*Value*“ libovolnou platnou rovnicí (viz ukázky na **obr. 11.4**). V položce „*Freq*“ můžeme definovat hodnoty platné pouze při malosignálové „*AC*“ analýze. U zdroje proudu *NFI* lze definovat v položce „*Noise_Expression*“ proud uvažovaný při šumové analýze.

11.2.1.2 Obvody pro transformaci signálů

Nelineární funkční zdroje typu „*NT*“
(„*Components/Analog Primitives/Function Sources/NFVofV*“ a další), **obr. 11.5**).



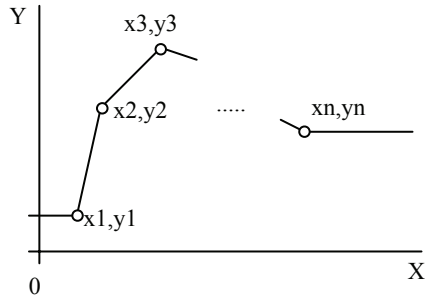
Obr. 11.5: Schématické značky zdrojů a) NTVofV, b) NTVofI, c) NTIofV, d) NTIofI.

NTVofV	zdroj napětí řízený napětím,
NTVofI	zdroj napětí řízený proudem,
NTIofV	zdroj proudu řízený napětím,
NTIofI	zdroj proudu řízený proudem.

Obecně se jedná o zdroj *NTYofX*, kde nelineární závislost veličiny *Y* na veličině *X* je modelována po částech lineární funkcí. Zápisu v poli „*Table*“

$$(x1,y1)\ (x2,y2)\ \ldots\ (xn,yn)$$

odpovídá výsledek podle **obr. 11.6**.



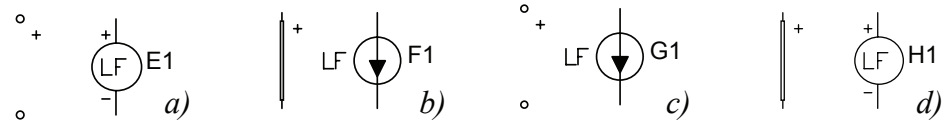
Obr. 11.6: K definici nelineárního zdroje typu *NTYofX*.

Souřadnice bodů lomu je možné zadávat i pomocí symbolických proměnných. Podrobnosti jsou uvedeny v dokumentaci programu.

Příklad – *NTVofV*:

(-0.01 –10) (0.01 10) … napěťový zesilovač o zesílení 1000 a saturačních úrovních –10V a +10V.

Laplaceovy zdroje typu „LF“
(„*Components/Analog Primitives/Laplace Sources/LFVofV*“ a další), **obr. 11.7**).



Obr. 11.7: Laplaceovy zdroje typu a) *LFVofV*, b) *LFVofI*, c) *LFIofV*, d) *LFIofI*.

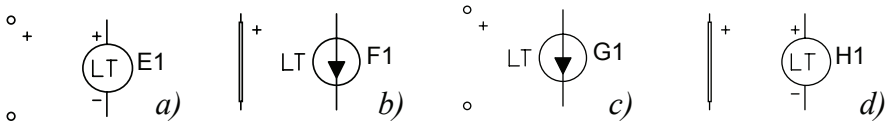
LFVofV	zdroj napětí řízený napětím,
LFVofI	zdroj napětí řízený proudem,
LFIofV	zdroj proudu řízený napětím,
LFIofI	zdroj proudu řízený proudem.

Obecně se jedná o zdroj *LFYofX*, kde setrvačná závislost veličiny *Y* na veličině *X* je modelována operátorovou přenosovou funkcí. Tato funkce Laplaceova operátoru *s* se zapisuje do pole „*Laplace*“ okna atributů zdroje. Často používanou alternativou je zápis názvu zdroje do tohoto pole a dodatečná definice příkazem *.define*.

Příklad – *LFVofV*:

100**s*/(*s***s*+100**s*+10000) … pásmová propust 2. řádu o přenosu napětí $\frac{100s}{s^2+100s+10000}$, nekonečném vstupním a nulovém výstupním odporu.

Laplaceovy zdroje typu „LT“
(„*Components/Analog Primitives/Laplace Sources/LTVofV*“ a další), **obr. 11.8**).



Obr. 11.8: Laplaceovy zdroje typu a) *LTVofV*, b) *LTVofI*, c) *LTIofV*, d) *LTIofI*.

Obecně se jedná o zdroj *LTYofX*, kde kmitočtová závislost veličiny *Y* na veličině *X* je modelována následujícím zápisem v poli „*FREQ*“:

(*F*1,*MAG*1,*PH*1) (*F*2,*MAG*2,*PH*2)…(*Fn*,*MAGn*,*PHn*),

například

(0,0,-5) (10,-1,-15) (100,-3,-45) (200,-9,-54) (500,-15,-60) (1000,-23,-72)

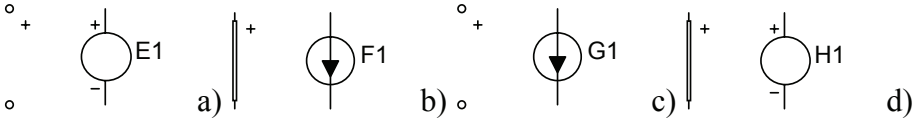
Čísla v každé trojici mají následující význam:

Fi	kmitočet v Hz,
MAGi	modul přenosu v dB,
PHi	argument přenosu ve stupních.

Amplitudová i fázová kmitočtová charakteristika jsou proloženy definovanými body lomenou čarou. Pro kmitočty menší než *F*1 a větší než *Fn* platí hodnoty *MAG*1, *PH*1, resp. *MAGn*, *PHn*.

Podrobnosti viz demonstrační soubor **P1.CIR**.

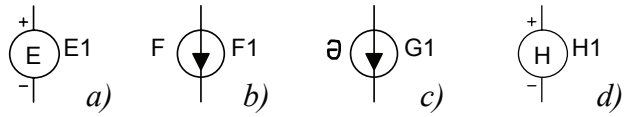
Závislé (řízené) zdroje typu „YofX“
(„*Components/Analog Primitives/Dependent Sources/VofV*“ a další), **obr. 11.9**).



Obr. 11.9: Závislé zdroje typu a) *VofV*, b) *IofI*, c) *IofV*, d) *VofI*

Jedná se o zdroj výstupní veličiny *Y* (napětí *V* nebo proud *I*), která je přímo úměrná vstupní veličině *X* (napětí *V* nebo proud *I*). Konstanta úměrnosti se zapisuje do pole „*Value*“ atributového okna součástky.

Závislé (řízené) zdroje typu E, F, G, H
(„*Components/Analog Primitives/Dependent Sources/EVOFV, FIOFI, GIOFV, HVOFI*“, **obr. 11.10**).



Obr. 11.10: Řízené zdroje SPICE: a) napětí řízený napětím, b) proudu řízený proudem, c) proudu řízený napětím, d) napětí řízený proudem.

Jsou to velmi univerzálně použitelné zdroje z rodiny programů SPICE. Zadáme-li je v MicroCapu jako součástky typu SPICE, tedy v textovém poli, pak můžeme plně využít této univerzálnosti. V schématickém módu nejsou některé funkce uvolněny, neboť jsou zabezpečeny jinými zdroji MicroCapu – konkrétně funkčními zdroji a Laplaceovými zdroji. Níže jsou uvedeny funkce v schématickém módu. Všechny funkce jsou detailně uvedeny v dokumentaci programů MicroCap a SPICE.

Symboly *E*, *F*, *G* a *H* označují následující typy zdrojů:

E .. zdroj napětí řízený napětím,
F .. zdroj proudu řízený proudem,
G ..zdroj proudu řízený napětím,
H ..zdroj napětí řízený proudem.

Princip bude vysvětlen na zdrojích napětí řízených napětím (*EVOFV*).

Dva základní tvary v položce „*Value*“:

1)

*n*1*p n*1*m* ...[*nkp nkm*] [*p*0...*pk*] [*IC*=*c*1...[,*ck*]]

Význam jednotlivých položek:

*n*1*p*	jméno uzlu první řídicí brány se znaménkem „*plus*“
*n*1*m*	jméno uzlu první řídicí brány se znaménkem „*mínus*“
…	
nkp	jméno uzlu *k*-té řídicí brány se znaménkem „*plus*“
nkm	jméno uzlu *k*-té řídicí brány se znaménkem „*mínus*“
*p*0...*pk*	koeficienty polynomu
*c*1...[,*ck*]	počáteční podmínky, tj. počáteční hodnoty napětí *v*1 … *vk* (viz dále) pro časovou analýzu.

Výstupní napětí *Vout* zdroje je pak dáno vzorcem

$$Vout=p0+p1*v1+p2*(v2\text{^}2)+p3*(v3\text{^}3)+...pk*(vk\text{^}k) = p0+\sum_{i=1}^{k} pi\ vi\text{^}i\,,$$

kde *vi* je napětí mezi uzly *nip* a *nim* *i*-té řídicí brány (viz **obr. 11.11**).

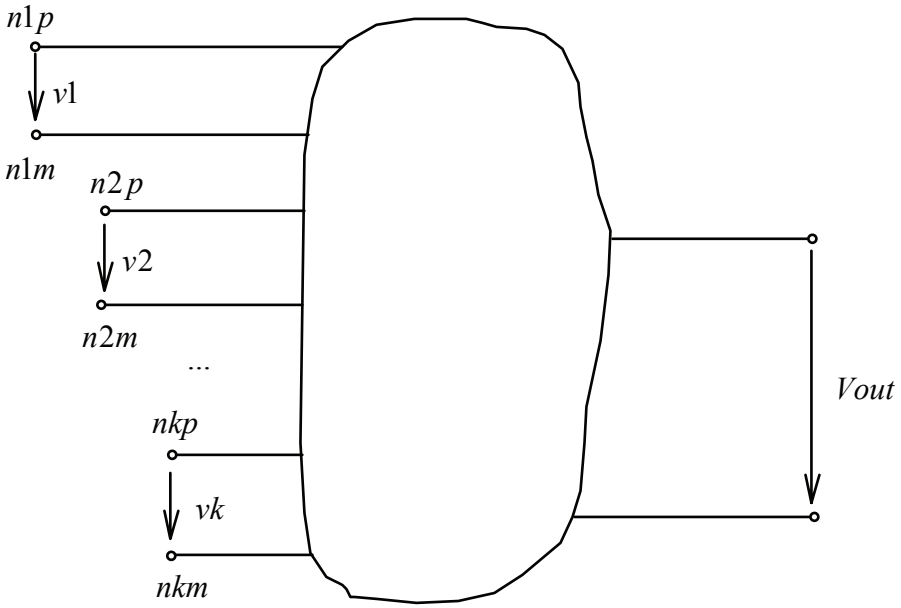
Tímto způsobem je možné modelovat polynomiální závislosti výstupního napětí zdroje na jiných napětích v obvodu.

2)

POLY(<*k*>) *n*1*p n*1*m* ...[*nkp nkm*] [*p*0...*pk*] [*IC*=*c*1...[,*ck*]]

V tomto případě je výstupní napětí zdroje dáno složitým váhovaným součtem součinů různých mocnin řídicích napětí v obvodu. Podrobnosti lze nalézt v dokumentaci programu.

Poznámka: V režimu schématického editoru nefungují popisované zdroje tak jak by měly. Nelze zadávat příkaz *IC*. Týká se verze programu MicroCap 7.0.9.



Obr. 11.11: K objasnění položek ve vzorcích pro výstupní napětí polynomiálního zdroje *EVOFV*.

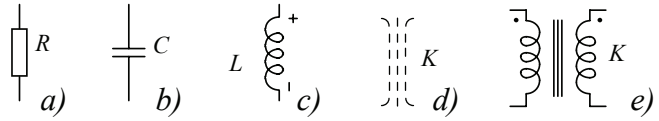
11.2.1.3 Význam atributů zdrojů v různých typech analýz

Tabulka 11.3 ukazuje, jaké atributy zdrojů se použijí při výpočtu stejnosměrného pracovního bodu obvodu v analýzách „*Dynamic DC*“, „*Transient*“ a „*DC*“ a jaké amplitudy a počáteční fáze se jim přiřadí v analýze „*AC*“. Všimněte si zejména rozdílů v interpretaci parametrů zdrojů typu „*V*“ a „*I*“ při výpočtu stejnosměrného pracovního bodu v různých analýzách.

zdroj	DC pracovní bod, počítaný v analýzách:				amplituda / poč. fáze
	„Dynam.DC“	„*Transient*“	„*AC*“	„*DC*“	„*AC*“
Battery	VALUE	VALUE	VALUE	VALUE	0 / 0º
Fixed Analog	VALUE	VALUE	VALUE	VALUE	0 / 0º
I source	VALUE	VALUE	VALUE	VALUE	0 / 0º
NFV, NFI	VALUE	VALUE	VALUE	VALUE	FREQ
Pulse Source	VZERO	VZERO	VZERO	VZERO	1V / 0º
Sine Source	V(0)	V(0)	V(0)	V(0)	1V / 0º
V,I – transient format:					AC mag./AC ph.
None	DC	DC	DC	DC	
Pulse	V1	V1	DC	DC	
Sin	V0	V0	DC	DC	
Exp	V1	V1	DC	DC	
PWL	V1	V1	V1	-	
SFFM	V0	V0	DC	DC	

Tab. 11.3: Uplatnění parametrů zdrojů signálů v různých typech analýz.

11.2.2 Pasivní prvky typu R, C a L, obvody s magnetickými vazbami a transformátory



Obr. 11.12: Schématické značky a) rezistoru, b) kapacitoru, c) induktoru, d) feritového jádra nebo magnetické vazby mezi induktory, e) lineárního transformátoru.

11.2.2.1 Rezistory

Vysvětlivky k editačnímu oknu rezistoru na **obr. 11.13**:

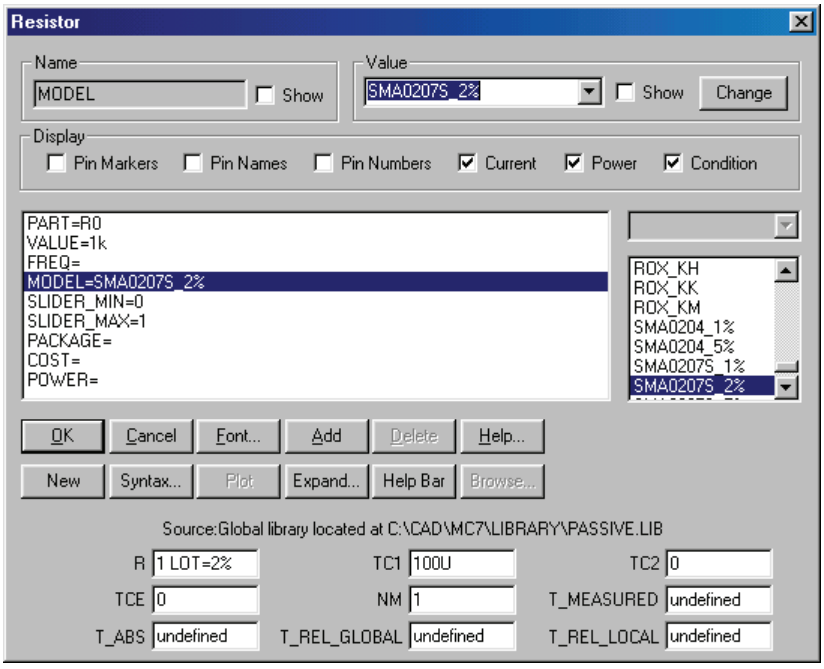
VALUE

V nejjednodušším případě definujeme rezistor pouze číselnou hodnotou jeho odporu v poli „*VALUE*“. Tato položka může obecně obsahovat vzorec pro výpočet odporu z obvodových veličin, např.:

$5k*(1+v(8)/4)$
$1k*(2+10*(t>10m))$

První vzorec definuje závislost odporu na napětí uzlu č. 8, druhý pak závislost odporu na simulačním čase v analýze „*Transient*“ (pro časy do 10 *m*s bude odpor 2 *k*Ω, pro větší časy pak 12 *k*Ω).

Hodnota může být doplněna teplotními koeficienty. Podrobnosti viz část 9.5.7.2. Položku „*VALUE*“ lze rovněž definovat jako symbolickou proměnnou (viz část 9.4.3.4).



Obr. 11.13: Editační okno rezistoru.

FREQ

Položka „*FREQ*“ je standardně nepředvyplněna. To znamená, že při analýze „*AC*“ bude pro odpor rezistoru použita hodnota z položky „*VALUE*“. Zápisem hodnoty nebo vzorce (s

možností využití proměnné *F* – kmitočtu) do pole „*FREQ*“ definujeme odpor speciálně pro analýzu „*AC*“.

MODEL

Rezistoru lze nepovinně přiřadit model o 9 položkách (viz **obr. 11.13**). Hlavní důvody, které by nás mohly vést k přiřazení modelu, jsou: zavedení tolerancí, teplotních součinitelů nebo definování teplotních režimů. S výjimkou položky „*NM*“ byl význam všech ostatních postupně vysvětlen v kapitolách o teplotní a statistické analýze. „*NM*“ značí „*Noise Multiplier*“ – násobící koeficient šumu. Tímto číslem se násobí tepelný šum generovaný rezistorem. Přednastavená hodnota je 1. Dosazením nuly získáme model „nešumícího“ rezistoru.

Význam ostatních položek *SLIDER*, *PACKAGE*, *COST*, *POWER* lze vyčíst z dokumentace programu.

11.2.2.2 Kapacitory

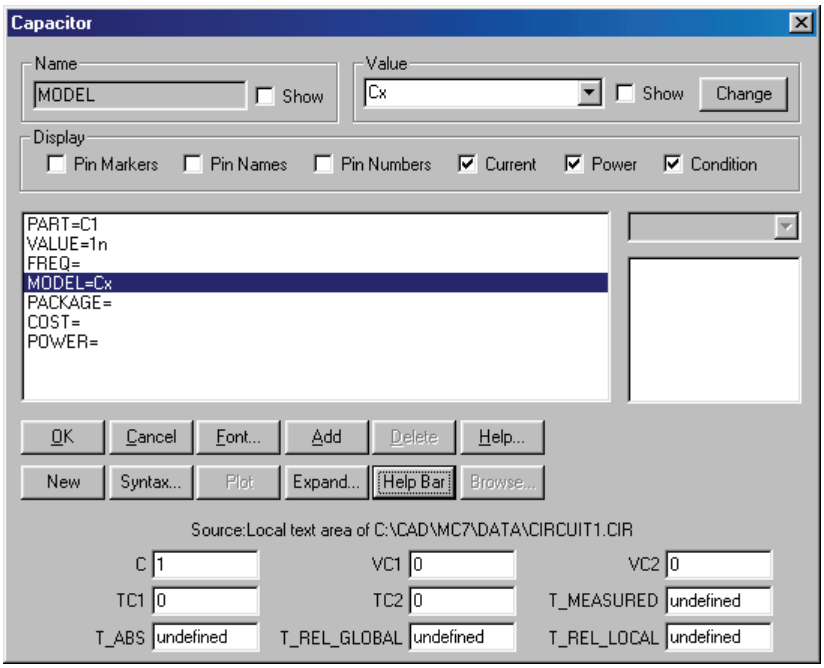
Vysvětlivky k editačnímu oknu kapacitoru na **obr. 11.14**:

VALUE

V nejjednodušším případě definujeme kapacitor pouze číselnou hodnotou jeho kapacity v poli „*VALUE*“. Tato položka může obecně obsahovat vzorec pro výpočet kapacity z obvodových veličin. Hodnota může být doplněna počáteční podmínkou, tj počátečním napětím na kapacitoru, pomocí klíčového slova *IC*.

Příklady:

5u
2.5p/SQRT(1-V(2)/0.7))
2.5P*(1+2u*T)
5u IC=5V



Obr. 11.14: Editační okno kapacitoru.

Položku „*VALUE*“ lze rovněž definovat jako symbolickou proměnnou (viz část 9.4.3.4).

FREQ

Podobně jako u rezistoru, zápisem hodnoty nebo vzorce (s možností využití proměnné *F* – kmitočtu) do pole „*FREQ*“ definujeme kapacitu speciálně pro analýzu „*AC*“.

MODEL

Kapacitoru lze nepovinně přiřadit model o 9 položkách (viz **obr. 11.14**). Položka „*C*“ je násobící koeficient, který má stejný význam jako násobící koeficient „*R*“ u rezistoru.

Význam položek *VC*1 a *VC*2: hodnota „*VALUE*“ je vynásobena tzv. faktorem *QF*, který je roven

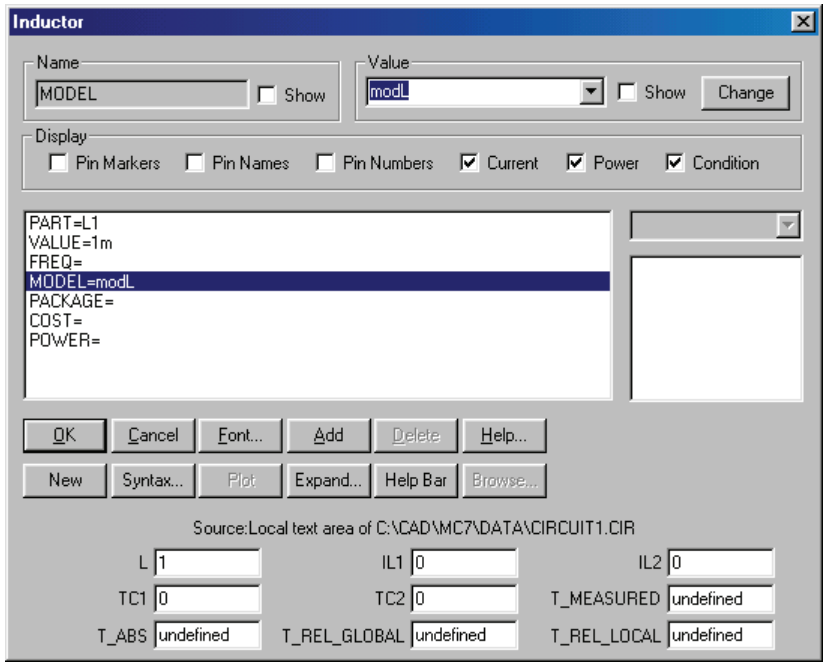
$$QF = 1 + VC1.V + VC2.V^2,$$

kde V je napětí na kapacitoru.

Význam ostatních položek *PACKAGE*, *COST*, *POWER* lze vyčíst z dokumentace programu.

11.2.2.3 Induktory

Vysvětlivky k editačnímu oknu induktoru na **obr. 11.15**:



Obr. 11.15: Editační okno induktoru.

VALUE

V nejjednodušším případě definujeme induktor pouze číselnou hodnotou jeho indukčnosti v poli „*VALUE*“. Tato položka může obecně obsahovat vzorec pro výpočet indukčnosti z obvodových veličin. Hodnota může být doplněna počáteční podmínkou, tj počátečním proudem induktorem, pomocí klíčového slova *IC*.

Příklady:

10m
2.5u*(1+2u*T)
5m IC=2mA

Položku „*VALUE*“ lze rovněž definovat jako symbolickou proměnnou (viz část 9.4.3.4).

FREQ

Zápisem hodnoty nebo vzorce (s možností využití proměnné *F* – kmitočtu) do pole „*FREQ*“ definujeme indukčnost speciálně pro analýzu „*AC*“.

MODEL

Induktoru lze nepovinně přiřadit model o 9 položkách (viz **obr. 11.15**). Položka „*L*“ je násobící koeficient, který má stejný význam jako násobící koeficient „*R*“ u rezistoru, resp. „*C*“ u kapacitoru.

Význam položek *IL*1 a *IL*2: hodnota „*VALUE*“ je vynásobena tzv. faktorem *QF*, který je roven

$$QF = 1 + IL1.I + IL2.I^2,$$

kde *I* je proud induktorem.

Význam ostatních položek *PACKAGE*, *COST*, *POWER* lze vyčíst z dokumentace programu.

Modelování lineární magnetické vazby mezi induktory

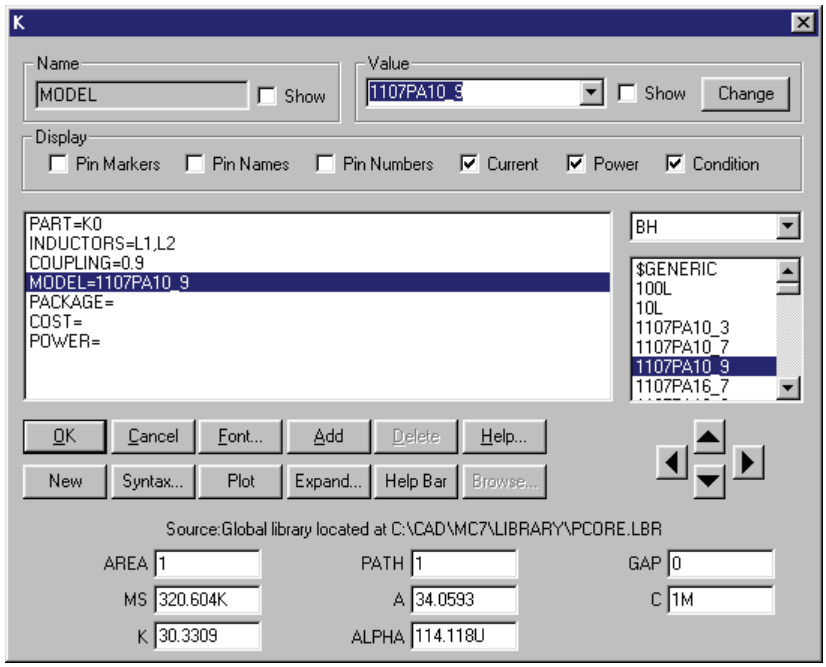
Je zabezpečováno součástkou typu

„*K Device*“ („*Component/Analog Primitives/Passive Components/K*“), **obr. 11.12** d).

Na **obr. 11.16** je editační okno této součástky.

Chceme-li modelovat magnetickou vazbu mezi induktory, umístíme kamkoliv na plochu značku „*K device*“ z **obr. 11.12** d), do položky „*Inductors*“ v editačním okně „*K*“ zapíšeme jména induktorů s magnetickou vazbou, oddělených čárkami, a do položky „*Coupling*“ zapíšeme stupeň magnetické vazby (číslo od 0 do 1).

Důležité je, že při modelování lineární magnetické vazby nesmíme součástce „*K Device*“ přiřazovat model.



Obr. 11.16: Editační okno součástky „*K Device*“ pro definování magnetické vazby mezi induktory.

Modelování cívek na nelineárním magnetickém jádru

Je opět zabezpečováno součástkou typu

„*K device*“ („*Component/Analog Primitives/Passive Components/K*“), **obr. 11.12** d).

V editačním okně „*K*“ podle **obr. 11.16** je třeba definovat model jádra. MicroCap modeluje nelineární jádro na základě tzv. *Jiles-Athertonova* modelu. Význam jednotlivých položek modelu nemusí být pro běžného uživatele důležitý, většinou si vybírá z existujících modelů z knihovny.

V případě, že součástce „*K Device*“ přiřadíme konkrétní model, změní svůj význam položky „*Value*“ u jednotlivých induktorů v tom smyslu, že se bude jednat o počty závitů, nikoliv o indukčnosti. Do položek „*Value*“ všech induktorů s nelineárním magnetickým jádrem tedy musíme zapisovat celá kladná čísla.

Pro úplnost dodejme, že namísto součástky „*K Device*“ je možno k definici jádra použít taky modely konkrétních feritových jader, které nalezneme v složce „*Component/Analog Library/Ferrite/*“.

Příklad 1: Modelování cívek *L*1 a *L*2 na nelineárním magnetickém jádru.

1. Na plochu umístíme induktory *L*1 a *L*2. Do položky „*Value*“ editačního okna induktoru *L*1 zapíšeme počet závitů cívky *L*1, do položky „*Value*“ editačního okna induktoru *L*2 počet závitů cívky *L*2.
2. Na plochu umístíme „*K Device*“. Do položky „*Inductors*“ editačního okna „*K*“ zapíšeme *L*1,*L*2, do položky „*Coupling*“ stupeň magnetické vazby. Vybereme, případně editujeme model jádra.

Příklad 2: Modelování jediné cívky na nelineárním magnetickém jádru.

1. Na plochu umístíme induktor *L*1. Do položky „*Value*“ editačního okna induktoru *L*1 zapíšeme počet závitů cívky *L*1.
2. Na plochu umístíme „*K Device*“. Do položky „*Inductors*“ editačního okna „*K*“ zapíšeme *L*1, do položky „*Coupling*“ stupeň magnetické vazby závitů a jádra. Vybereme, případně editujeme model jádra.

K snadnějšímu pochopení způsobů modelování induktorů s magnetickým jádrem doporučujeme analýzu demonstračních příkladů **CORE.CIR** (cívka na magnetickém jádru z materiálu 3C8, ukázka hystereze) a **CORE3.CIR** (vazba tří induktorů nelineárním jádrem).

11.2.2.4 Transformátory

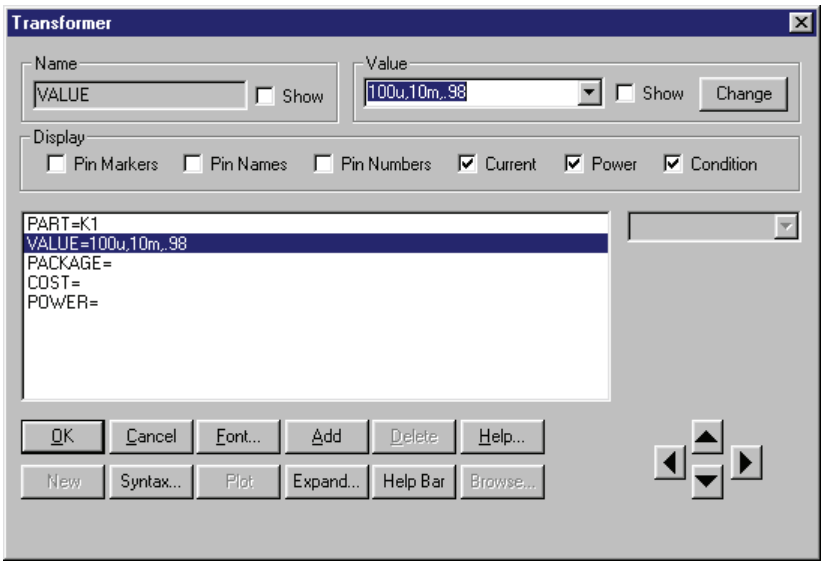
Demonstrační příklad **TRANS.CIR** ukazuje tři základní způsoby modelování transformátorů:

a) Součástka „*Transformer*“

(„*Component/Analog Primitives/Passive Components/Transformer*“)

Jde o lineární model dvou cívek s magnetickou vazbou. Schématická značka je na **obr. 11.12** e). Editační okno je na **obr. 11.17**. V položce „*Value*“ se zadávají 3 hodnoty, oddělené čárkami:

Indukčnost primárního vinutí, indukčnost sekundárního vinutí, stupeň magnetické vazby.



Obr. 11.17: Editační okno transformátoru.

b) Induktory s magnetickou vazbou

Transformátor je možné modelovat pomocí klasických induktorů a zařízení „*K Device*“ (viz část 11.2.2.3). Výhodou je možnost modelování transformátorů o více vinutích a uvažování nelineárního jádra.

c) Makroobvod

Demonstrační soubor **TRANS.CIR** používá zabudované makro **CENTAP**, což je model lineárního transformátoru o jednom primárním a dvou sekundárních vinutích.

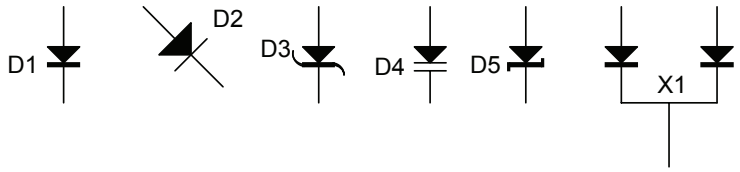
11.2.3 Polovodičové a aktivní prvky

11.2.3.1 Diody

Modely polovodičových diod jsou přístupné v těchto nabídkách MicroCapu:

a) „*Component/Analog Primitives/Passive Components/Diode/“, resp. D*45/

Z této nabídky je možné vybírat modely všech diod z aktivních knihoven typu **.LIB** a **.LBR**. Nabídka *D*45 přiřadí diodě schématickou značku otočenou o úhel 45 stupňů oproti ortogonálním souřadnicím.



Obr. 11.18: Obecná dioda (D1), dioda „D45“ (D2), Schottkyho dioda (D3), varaktor – kapacitní dioda (D4), Zenerova dioda (D5), dvojitý usměrňovač (X1).

b) „*Component/Analog Library/Diode/*“

V této nabídce jsou modely diod z knihoven členěny do podnabídek (Schottkyho diody, Zenerovy diody, varikapy apod.).

Editační okno je ukázáno na **obr. 11.19**.

VALUE

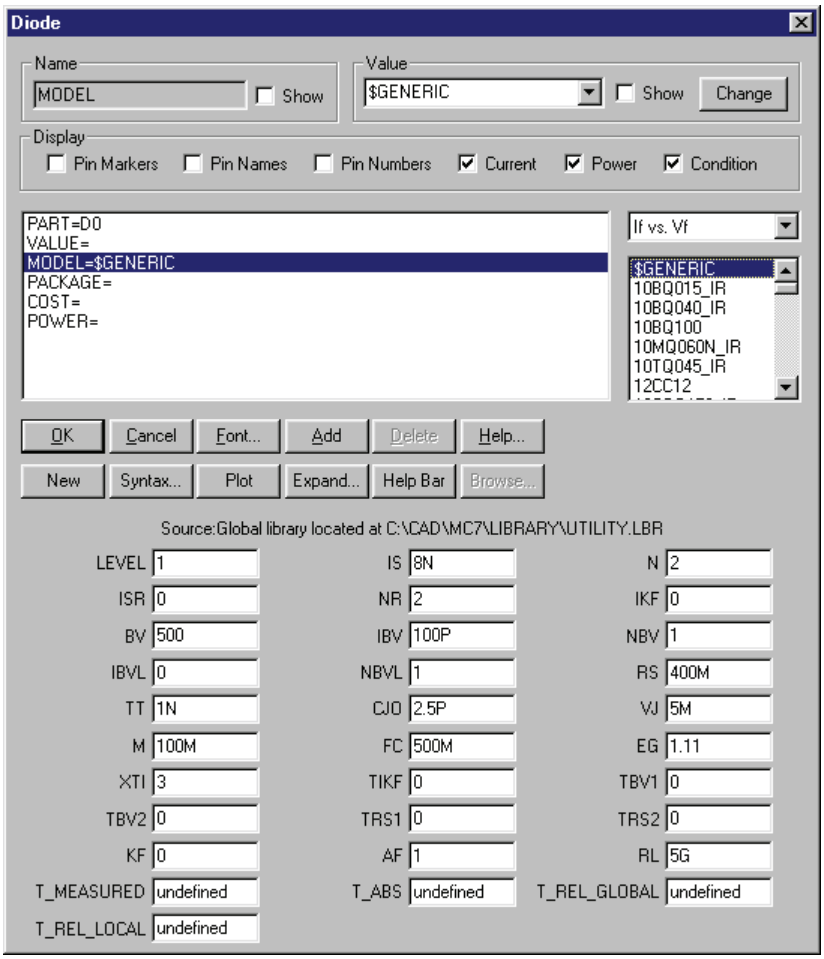
Tato položka je standardně nepředvyplněná. Mohou se zde objevit nepovinně až 3 údaje:

[*area*] [*OFF*] [*IC=napětí na diodě*]

Příklady:

1.0 OFF IC=1mV
2.1 IC=.7mV

- Area: číslo, související s plochou čipu. Toto číslo slouží jako násobící (*), resp. dělicí (/) faktor některých parametrů modelu (viz symboly * a / v sloupci „*area*“ níže uvedené tabulky).
- OFF: přítomnost tohoto klíčového slova způsobí, že dioda nebude uvažována v modelu obvodu během první iterace při hledání stejnosměrného pracovního bodu. Jde o jednu z metod překonávání problémů s konvergencí, viz část 4.9.2.
- IC: tímto příkazem definujeme počáteční napětí na diodě. Příkaz se uplatní jen při analýze „*Transient*“, je-li vyblokován výpočet stejnosměrného pracovního bodu.



Obr. 11.19: Editační okno polovodičové diody.

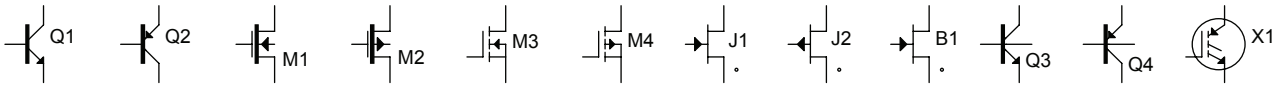
Parametry modelu:

Dioda je modelována 27 parametry, k nimž se přičítají 4 parametry pro teplotní modelování (*T_MEASURED* až *T_REL_LOCAL*, viz část 9.5.7.2). Základní popis parametrů je uveden v Tab. P10.3. V tabulce jsou uvedeny originální anglické názvy parametrů. Objasňování jejich významu by šlo mimo rámec této knihy. Zájemce odkazujeme na literaturu [3.7]. Jakékoliv editování těchto parametrů způsobí překopírování příslušného příkazu *.MODEL* do složky „*Text*“. Model pak bude mít lokální platnost pro analyzovaný obvod.

Název	Parametr	Jednotka	Default	Area	Level
LEVEL	1=Spice2G 2=PSpice		1		
IS	Saturation current	A	1E-14	*	1,2
N	Emission coefficient		1.0		1,2
ISR	Recombination current parameter	A	0	*	2
NR	Emission coefficient for ISR		2		2
IKF	High-injection "knee" current	A	INF	*	2
BV	Reverse breakdown "knee" voltage	V	INF		1,2
IBV	Reverse breakdown "knee" current	A	1E-10	*	1,2
NBV	Reverse breakdown ideality factor		1		2
IBVL	Low-level rev. breakdown "knee" current	A	0		2
NBVL	Low-level rev. breakdown ideality factor		1		2
RS	Series resistance	Ω	0	/	1,2
TT	Transit time	s	0.0		1,2
CJO	Zero-bias junction capacitance	F	0.0	*	1,2
VJ	Junction potential	V	1.0		1,2
M	Junction grading coefficient		0.5		1,2
FC	Forward bias depletion coefficient		0.5		1,2
EG	Energy gap	eV	1.11		1,2
XTI	Temperature exponent for IS		3.0		1,2
TIKF	IKF temperature coefficient (linear)	1/°C	0		2
TBV1	BV temperature coefficient (linear)	1/°C	0		2
TBV2	BV temperature coefficient (quadratic)	$1/(°C)^2$	0		2
TRS1	RS temperature coefficient (linear)	1/°C	0		2
TRS2	RS temperature coefficient (quadratic)	$1/(°C)^2$	0		2
KF	Flicker noise coefficient		0.0		1,2
AF	Flicker noise exponent		1.0		1,2
RL	Junction Leakage Resistance	Ω	INF		1
T_MEASURED	Measured temperature	°C	nedef.		1,2
T_ABS	Absolute temperature	°C	nedef.		1,2
T_REL_GLOBAL	Relative to current temperature	°C	nedef.		1,2
T_REL_LOCAL	Relative to AKO model temperature	°C	nedef.		1,2

Tab. 11.4: Parametry modelu polovodičové diody.

11.2.3.2 Tranzistory



Obr. 11.20: Tranzistory: bipolární NPN (Q1) a NPN (Q2), NMOS (M1), PMOS (M2), DNMOS (M3), DPMOS (M4), NJFET (J1), PJFET (J2), GaAsFET (B1), NPN4 (Q3), PNP4 (Q4), IGBT (X1).

Modely tranzistorů jsou dostupné v těchto položkách:

a) „*Component/Analog Primitives/Active Devices/*"

Tranzistory jsou zde rozděleny do následujících kategorií:

NPN, PNP, NMOS, PMOS, DNMOS, DPMOS, NJFET, PJFET, GaAsFET, NPN4, PNP4.

b) "*Component/Analog Library/*"

Zde je možné nalézt následující kategorie:

BJT (bipolární), BJT Pwr (bipolární výkonové), IGBT, JFET, MOSFET

c) "*Component/Analog Library/Vendor/*"

Modely tranzistorů je možné nalézt v knihovnách některých výrobců integrovaných obvodů:

IR (MOSFET), MOTOROLA (TMOS), ON Semi (BJT), Philips (BJT), Zetex (BJT).

Modely některých speciálních tranzistorů, např. tranzistorů IBGT, jsou napsány jako speciální podobvody SPICE.

Editační okno bipolárního tranzistoru obsahuje podobně jako u diody položky "*PART*", "*VALUE*" a "*MODEL*".

Položka "*VALUE*" je standardně nevyplněná, může obsahovat až 4 údaje ve formátu:

[*area*] [*OFF*] [*IC=napětí báze-emitor* [*napětí kolektor-emitor*]]

Vysvětlení způsobu použití položky „*VALUE*" je obdobné jako u položky „*VALUE*" v editačním okně diody.

Vyplnění položky „*VALUE*" je nepovinné i u ostatních typů tranzistorů. Například u tranzistorů MOSFET je syntaxe poněkud komplikovaná. Zájemce proto odkazujeme na odbornou literaturu [3.7].

Parametry modelů:

V tabulkách jsou pro úplnost uvedeny parametry modelů tranzistorů BJT (bipolárních), JFET a Ga AsFET. Význam parametrů je možné nastudovat z odborné literatury [3.7]. Jsou vynechány teplotní parametry (*T_MEASURED* až *T_REL_LOCAL*), které jsou stejné jako u polovodičové diody.

Příslušné tabulky nejsou uvedeny pro tranzistory MOSFET. Modelování těchto tranzistorů je velmi komplikované a příslušné modely jsou v neustálém vývoji. Zájemce odkazujeme na aktuální odbornou literaturu, zejména na Internetové zdroje.

Název	Parametr	Jednotka	Default	Area
IS	Saturation Current	A	1E-16	*
BF	Forward Beta		100	
NF	Forward emission coefficient		1	
VAF	Forward Early voltage	V	INF	
IKF	BF high-current-roll-off corner	A	INF	*
ISE	BE leakage saturation current	A	0	*
NE	BE leakage emission coefficient		1.5	
BR	Ideal maximum reverse beta		1	
NR	Reverse current emission coefficient		1	
VAR	Reverse Early voltage	V	INF	
IKR	BR high-current roll-off corner	A	INF	*
ISC	BC leakage saturation current	A	0	*
NC	BC leakage emission coefficient		2	
NK	High current rolloff coefficient		.5	
ISS	Substrate pn saturation current	A	0	
NS	Substrate pn emission coefficient		1	
RE	Emitter resistance	Ω	0	/
RB	Zero-bias base resistance	Ω	0	/
RBM	Minimum RB at high currents	Ω	RB	/
IRB	Current where RB falls by half	A	INF	*
RC	Collector resistance	Ω	0	/
CJE	BE zero-bias depletion capacitance	F	0	*
VJE	BE junction built-in potential	V	.75	
MJE	BE junction grading coefficient		.33	
CJC	BC zero-bias depletion capacitance	F	0	*
VJC	BC junction built-in potential	V	.75	
MJC	BC junction grading coefficient		.33	
XCJC	Fraction of BC dep. cap. to internal base		1	
CJS	CS zero-bias depletion capacitance	F	0	*
VJS	CS junction built-in potential	V	.75	
MJS	CS junction grading coefficient		0	
FC	Forward-bias depletion coefficient		.5	
TF	Ideal forward transit time	s	0	
XTF	TF bias dependence coefficient		0	
VTF	Transit time dependence on VBC	V	INF	
ITF	Transit time dependence on IC	A	0	*
PTF	Excess phase at 1/(2*PI*TF)Hz	stupeň	0	
TR	Ideal reverse transit time	s	0	
EG	Energy gap	eV	1.11	
XTB	Temperature coefficient for betas		0	
XTI	Temperature exponent for IS		3	
TRE1	RE temp. coefficient (linear)	1/°C	3	

Tab. 11.5: Parametry modelu tranzistoru BJT (část 1)

Název	Parametr	Jednotka	Default	Area
TRE2	RE temp. coefficient (quadratic)	$1/(°C)^2$	3	
TRB1	RB temp. coefficient (linear)	1/°C	3	
TRB2	RB temp. coefficient (quadratic)	$1/(°C)^2$	3	
TRM1	RBM temp. coefficient (linear)	1/°C	3	
TRM2	RBM temp. coefficient (quadratic)	$1/(°C)^2$	3	
TRC1	RC temp. coefficient (linear)	1/°C	3	
TRC2	RC temp. coefficient (quadratic)	$1/(°C)^2$	3	
KF	Flicker-noise coefficient		0	
AF	Flicker-noise exponent		1	

Tab. 11.5: Parametry modelu tranzistoru BJT (dokončení)

Název	Parametr	Jednotka	Default	Area
VTO	Threshold voltage	V	-2.0	
BETA	Transconductance parameter	A/V^2	1E-4	*
LAMBDA	Channel-Length modulation	1/V	0.0	
IS	Gate p-n saturation current	A	1E-14	*
RD	Drain ohmic resistance	Ω	0.0	/
RS	Source ohmic resistance	Ω	0.0	/
CGD	GD zero-bias junction cap.	F	0.0	*
CGS	GS zero-bias junction cap.	F	0.0	*
M	Gate p-n grading coefficient		0.5	
PB	Gate junction potential	V	1.0	
FC	Forward-bias depletion coeff.		0.5	
VTOTC	VTO temperature coefficient	V/°C	0	
BETATCE	Forward bias depletion coeff.	%/°C	0	
XTI	IS temperature coefficient		3	
KF	Flicker noise coefficient		0.0	
AF	Flicker noise exponent		1.0	

Tab. 11.6: Parametry modelu tranzistoru JFET.

Název	Parametr	Jednotka	Default	Area	Level
LEVEL	Model level(1,2,3)		1		All
VTO	Pinch-off voltage	V	-2.5		All
ALPHA	Saturation voltage parameter	1/V	2.0		All
BETA	Transconductance coefficient	A/V^2	0.1	*	All
B	Doping tail extending parameter	1/V	0.3		2
LAMBDA	Channel length modulation	1/V	0.0		All
GAMMA	Static feedback parameter		0.0		3
DELTA	Output feedback parameter	1/(A*V)	0.0		3
Q	Power-law parameter		2.0		3
RG	Gate ohmic resistance		0.0	/	All

Tab. 11.7: Parametry modelu tranzistoru GaAsFET (část 1)

Název	Parametr	Jednotka	Default	Area	Level
RD	Drain ohmic resistance		0.0	/	All
RS	Source ohmic resistance		0.0	/	All
IS	Gate pn saturation current	A	1E-14		All
N	Gate pn emission coefficient		1.0		All
M	Gate pn grading coefficient		0.5		All
VBI	Gate pn built-in potential	V	1.0		All
CGD	Zero-bias GD capacitance	F	0.0	*	All
CGS	Zero-bias GS capacitance	F	0.0	*	All
CDS	Fixed DS capacitance	F	0.0	*	All
FC	Depletion capacitance coefficient		0.5		All
VDELTA	Capacitance transition voltage	V	0.2		2,3
VMAX	Capacitance limiting voltage	V	0.5		2,3
EG	Bandgap voltage (barrier height)	eV	1.11		All
XTI	IS temperature exponent		0.0		All
VTOTC	VTO temperature coefficient	V/°C	0.0		All
BETATCE	BETA exponential temp. coeff.	%/°C	0.0		All
TRG1	RG temp. coefficient (linear)	1/°C	0.0		All
TRD1	RD temp. coefficient (linear)	1/°C	0.0		All
TRS1	RS temp. coefficient (linear)	1/°C	0.0		All
KF	Flicker noise coefficient		0.0		All
AF	Flicker noise exponent		1.0		All

Tab. 11.7: Parametry modelu tranzistoru GaAsFET (dokončení)

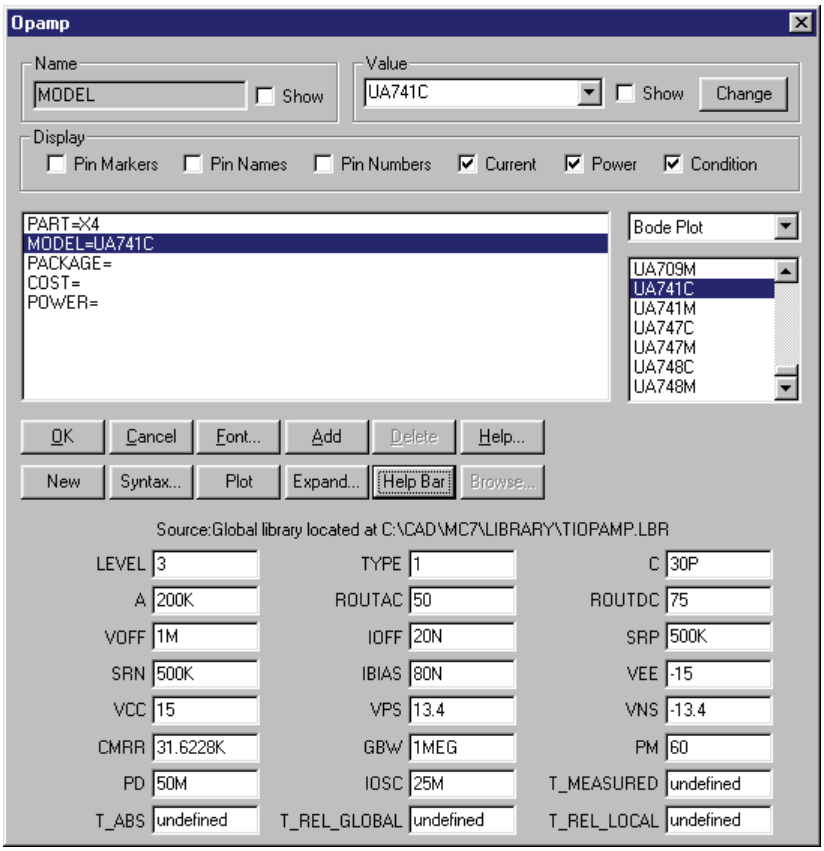
11.2.3.3 Operační zesilovače

Modely operačních zesilovačů jsou dostupné v těchto nabídkách:

a) „*Component/Analog Primitives/Active Devices/OpAmp/*"

Jde o "MicroCapovské" modely z knihoven ***.lbr**. Příslušné editační okno je na **obr. 11.21**. Model operačního zesilovače je charakterizován dvaceti elektrickými parametry plus čtyřmi „teplotními" parametry podobně jako u diody nebo dalších součástek. Pojmenování a implicitní hodnoty jednotlivých parametrů jsou zřejmé z Tab. 11.8.

„MicroCapovský" model operačního zesilovače může být využíván ve třech úrovních („*LEVELS*"): od úrovně 1 (velmi jednoduchý) až po úroveň 3 (komplikovaný model). V sloupci „*LEVEL*" v Tab. 11.8 je poznamenáno, v kterých úrovních se uplatňuje příslušný parametr modelu.



Obr. 11.21: Editační okno operačního zesilovače.

Název	Parametr	Jednotka	Default	Level
LEVEL	Model Level		1	
TYPE	1=NPN,2=PNP,3=JFET		1	3
C	Compensation Capacitor	F	30e-12	3
A	DC open loop gain		2e5	all
ROUTAC	AC output resistance	Ω	75	all
ROUTDC	DC output resistance	Ω	125	all
VOFF	Input offset voltage	V	.001	3
IOFF	Input offset current	A	1E-9	3
SRP	Maximum positive slew rate	V/s	5e5	2,3
SRN	Maximum negative slew rate	V/s	5e5	2,3
IBIAS	Input bias current	A	1e-7	3
VCC	Positive power supply	V	15	3
VEE	Negative power supply	V	-15	3
VPS	Maximum positive voltage swing	V	13	3
VNS	Maximum negative voltage swing	V	-13	3
CMRR	Common-mode rejection ratio		1e5	3
GBW	Unity gain bandwidth	Hz	1e6	2,3
PM	Phase margin	°	60	2,3
PD	Quiescent power dissipation	W	.025	3
IOSC	Output saturation current	A	.02	3

Tab. 11.8: Parametry modelu operačního zesilovače (model MicroCapu).

Model úrovně 1 (LEVEL 1):

Na této úrovni je operační zesilovač modelován jako jednoduchý zdroj proudu řízený diferenčním vstupním napětím. Zdroj proudu pracuje do odporové zátěže, která definuje výstupní odpor operačního zesilovače. Tento odpor spolu s transkonduktancí zdroje definuje celkové napěťové zesílení (*A* – „*DC open loop gain*“). V modelu se rozlišuje mezi stejnosměrným („*DC*“) a střídavým („*AC*“) výstupním odporem. Z dokumentace MicroCapu vyplývá, že celkový výstupní odpor je dán jejich součtem.

Z uvedeného je zřejmé, že daný prvek má nekonečný vstupní odpor a kmitočtově nezávislé napěťové zesílení. Model je možné použít pro jednoduché simulační úlohy, kde zjišťujeme zejména stejnosměrné poměry v obvodu.

Model úrovně 2 (LEVEL 2):

Model z úrovně 1 je zde rozšířen na modelování kmitočtově závislého přenosu se dvěma lomovými kmitočty. Tyto kmitočty souvisí se dvěma parametry: „*GBW*“ („*Unity gain bandwidth*“ – tranzitní kmitočet) a „*PM*“ („*Phase margin*“ – fázová bezpečnost). Z Tab. 11.8, převzaté z dokumentace MicroCapu, vyplývá, že od úrovně 2 je do modelu implementována nelineární část, umožňující modelování mezní rychlosti přeběhu („*SRP/SRN*“ – „*Slew rate positive/negative*“). Ve skutečnosti je však toto modelování aktivní až v úrovni 3.

Model úrovně 3 (LEVEL 3):

Tento model představuje rozšíření *Boyleova modelu*, používaného v programech typu SPICE ve formě podobvodů. Operační zesilovač je zde modelován zapojením se dvěma vstupními tranzistory, 5 diodami a řadou pasivních součástek a zdrojů. Typ tranzistorů (bipolární *NPN* nebo *PNP* nebo tranzistory *JFET*) určuje zejména parametry vstupní části operačního zesilovače. Lze jej zvolit v položce *TYPE*. Simulační teplota má pouze vliv na parametry interních tranzistorů a diod. Šum je generován rovněž pouze polovodičovými součástkami, bohužel však zde není spojitost se skutečnými šumovými vlastnostmi konkrétního typu operačního zesilovače!!!

b) „*Component/Analog Library/OpAmp/*“

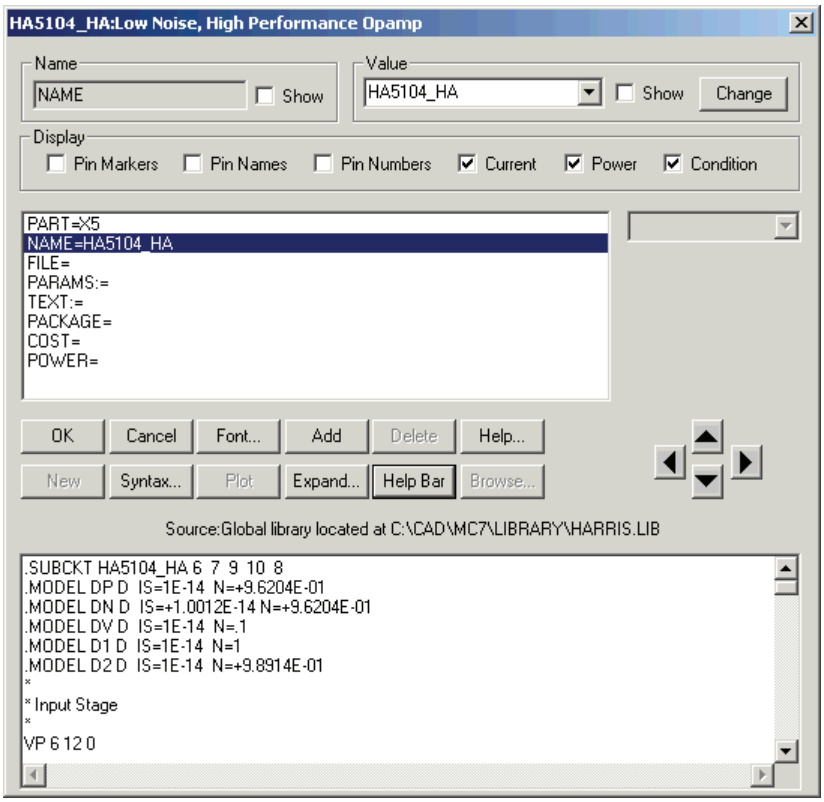
V této sekci jsou modely operačních zesilovačů roztříděny abecedně do několika kategorií podle označení typu. Převážně se jedná o “MicroCapovské” modely, jsou však zde i modely ve formě podobvodů SPICE.

c) „*Component/Analog Library/Vendor/*“

Tato sekce nabízí velký výběr operačních zesilovačů, roztříděných podle výrobců. Jedná se vesměs o podobvody SPICE.

Editační okno operačního zesilovače, modelovaného podobvodem SPICE, se liší od okna „MicroCapovského“ operačního zesilovače – srovnejte **obr. 11.21** s **obr. 11.22**:

V spodním podokně se objevuje zdrojový text podobvodu SPICE z dané knihovny. Jeho případná editace by vedla k automatickému kopírování zdrojového textu do složky „*Text*“. Příslušný model by pak měl lokální platnost právě pro analyzovaný obvod.



Obr. 11.22: Editační okno SPICE modelu operačního zesilovače.

Obsah položky „*Name*“ musí být totožný s názvem modelu v hlavičce „*.subckt*“. Vyplňování položek „*FILE*“, „*PARAMS*“ a „*TEXT*“ je nepovinné. Pokročilý uživatel může jejich význam nastudovat z dokumentace programu.

11.2.4 Jiné prvky

MicroCap nabízí modely řady dalších součástek – interně zabudované, ve formě makroobvodů nebo podobvodů SPICE. Jmenujme alespoň tyto:

- Přenosová vedení („*Transmission Lines*“) –viz ukázkové soubory **TL1.CIR**, **TL2.CIR**, **TL3.CIR**.
- Elektronické spínače („*Switches*“) – viz ukázkový soubor **SWITCH.CIR**.
- Elektronky („*Tubes*“) – viz ukázkové soubory **TUBE_AMP.CIR**, **TUBE6l6.CIR**.
- Krystaly („*Crystals*“) - viz ukázkový soubor **XTAL1.CIR**
- Tyristory („*Thyristors*“) - viz ukázkové soubory **THY1.CIR**, **THY2.CIR**
- Zdroje diskrétních signálů, popsané *z*-obrazy („*Z Transform Sources*“) - viz ukázkový soubor **ZDOMAIN.CIR**.

11.3 Některé konstanty, proměnné a funkce programu MicroCap

11.3.1 Některé konstanty a systémové proměnné MicroCapu

PI	3.141592653589793
E	EXP(1) = 2.718281828459045
GMIN	minimální vodivost specifikovaná v *Global Settings*
TEMP	teplota v stupních Celsia
VT	teplotní napětí ve voltech počítané ze vzorce

$$VT = \frac{1.3806226.10^{-23}\left(273.15 + TEMP\right)}{1.60219118.10^{-19}}.$$

Při 27°C vychází přibližně 25.86mV.

T	čas v sekundách
F	kmitočet v hertzích
J	imaginární jednotka
S	komplexní frekvence J.2.PI.F
INOISE	vstupní šum u analýzy „*AC*“
ONOISE	výstupní šum u analýzy „*AC*“
PGT	„Power Generated Total“, celkový výkon vytvářený v obvodu
PST	„Power Stored Total“, celkový výkon uchovávaný (akumulovaný) v obvodu
PDT	„Power Dissipated Total“, celkový výkon přeměňovaný v obvodu v teplo
EGT	„Energy Generated Total“, celková energie vytvářená v obvodu
EST	„Energy Stored Total“, celková energie uchovávaná (akumulovaná) v obvodu
EDT	„Energy Dissipated Total“, celková energie přeměňovaná v obvodu v teplo

Úplný soubor systémových konstant a proměnných je uveden v dokumentaci programu. Při označování vlastních konstant a proměnných uživatelem nesmí dojít ke kolizi s „rezervovanými“ symboly.

11.3.2 Některé funkce MicroCapu

Komplexní funkce - výběr:

MicroCap provádí většinu výpočtů v komplexní aritmetice. V analýzách „*DC*“ a „*Transient*“ jsou imaginární složky proměnných implicitně nulové.

RE(X) nebo REAL(X)	Reálná část *X*.
IM(X) nebo IMAG(X)	Imaginární část *X*.
MAG(X)	Modul *X*.
	MAG je nepovinné, pokud je z kontextu jasné, že se jedná o modul. Např. příkaz pro vykreslení modulové kmitočtové charakteristiky může být dvojí:

	MAG(V(1)) nebo V(1): vykreslí se modul V(1).
PH(X) nebo PHASE(X)	argument komplexního čísla *X* ve stupních.

Funkce pro zpracování signálů - výběr:

HARM(x)	Harmonické složky signálu x. RE(HARM(x))..Kosinové členy harmonických složek IM(HARM(x))..Sinové členy harmonických složek MAG(HARM(x))..Amplitudy harmonických složek PH(HARM(x))..Fáze harmonických složek
FFT(x)	Rychlá Fourierova transformace posloupnosti $x(n)$: $X(k)=\sum_{n=0}^{N-1} x(n)e^{-j2\pi kn/N}, k=0,1,2,..,N-1$ *N*.. počet členů posloupnosti $x(n)$. Rozdíly oproti HARM(x): - Stejnosměrná složka vychází *N* krát větší, - Amplitudy harmonických vycházejí *N*/2 krát větší.
IFFT(X)	Zpětná rychlá Fourierova transformace: $x(n)=\frac{1}{N}\sum_{k=0}^{N-1} X(k)e^{j2\pi kn/N}, n=0,1,2,..,N-1.$
THD(X)	posloupnost kumulativních hodnot celkového harmonického zkreslení spektra X způsobeného druhou, třetí, .., N/2- tou harmonickou, vyjádřená v procentech amplitudy základní harmonické. Př.: THD(HARM(v(1))).. zkreslení průběhu v(1): body č.0 a 1: nula bod č.2: procentní zkreslení 2. harmonickou bod č.3: procentní zkreslení 2. a 3. harmonickou ... bod č. N/2: zkreslení harmonickými do indexu N/2.
RMS(x)	posloupnost kumulativních hodnot RMS spektra (Root-Mean-Square, odmocnina ze střední hodnoty kvadrátu harm. složek = efektivní hodnota).

Další funkce - výběr:

ABS(y)	Absolutní hodnota, $\lvert y \rvert$.
SQRT(y)	Druhá odmocnina, $\sqrt{y}$.

SGN(y)	Znaménková funkce: SGN(y) = 1 pro y > 0, -1 pro y < 0 0 pro y = 0.
db(y)	Decibelové vyjádření: db(y) = 20∗LOG(y).
^	Exponenciální operátor: 10^2 = 100.
POW(y,x)	Mocninná funkce, y^x.
SUM(y,x)	Integrál, $\int_0^x y dx$, *x* je simulační čas u analýzy „*Transient*“, kmitočet u analýzy „*AC*“, DCINPUT1 u analýzy „*DC*“.
RMS(y)	Průběhová efektivní hodnota, $\sqrt{\frac{1}{x}\int_0^x y^2 dx}$.
AVG(y)	Průběhová střední hodnota, $\frac{1}{x}\int_0^x y dx$.
D(y)	Rozdíl mezi veličinou *y* v tomto a předešlém bodu řešení (diference). Př.: D(y)/D(t).. odpovídá časové derivaci signálu *y*(*t*).

Příklady často simulovaných elektrických veličin:

db(V(1))	Napětí V(1) v decibelech. Je-li v obvodu připojen jen jeden generátor signálu, pak se při kmitočtové analýze automaticky nastaví jeho úroveň na 0 db (podrobnosti viz část 4.8.4.4). Uvedený příkaz pak postačí k vykreslení amplitudové kmitočtové charakteristiky obvodu s výstupem na uzlu 1.
GD(V(1))	Skupinové zpoždění obvodu s výstupem na uzlu 1, viz předchozí text.
V(imp)/I(imp)	modul vstupní impedance obvodu, je-li k vstupním svorkám připojen zdroj s označením *imp*.
PH(V(imp)/I(imp))	fáze (argument) výše uvedené vstupní impedance.
V(Vcc)∗I(Vcc)	Okamžitý výkon odebíraný ze zdroje se jménem Vcc.
SUM(V(Vcc)∗I(Vcc),T)	Energie odebíraná ze zdroje Vcc.
ES(C1)	Energie nahromaděná v kapacitoru C1 („Energy Stored“).
PD(R1)	Ztrátový výkon na rezistoru R1 („Power Dissipated“).

Další příklady viz *Sample expressions* v interaktivní nápovědě.

Seznam použité literatury

1. Knihy, skripta, učební texty a články o elektrických obvodech a metodách jejich analýzy

[1.1] DESOER, CH.A.-KUH, E.S.: Basic Circuit Theory. McGraw-Hill, New York, 1969.

[1.2] KUO, B.C.: Linear Networks and Systems. McGraw-Hill, New York, 1967.

[1.3] SIEBERT, W.McC.: Circuits, Signals, and Systems. The MIT Press, McGraw-Hill Book Company, 1986. 2 díly.
Dostupný též ruský překlad "SIBERT,U.M.: Cepi, signaly, sistemy, Moskva, Mir, 1988".

[1.4] VLACH,J.: Basic Network Theory with Computer Applications. Van Nostrand Reinhold, New York, 1992.

[1.5] HOROWITZ,P.-HILL,W.: The Art of Electronics. Cambridge University Press, Second edition, 1989.

[1.6] LÁNÍČEK,R.: Elektronika – obvody, součástky, děje. BEN, Praha 1998.

[1.7] MAYER,D.: Úvod do teorie elektrických obvodů. SNTL/ALFA, Praha 1978.

[1.8] MAYER,D.: Analýza elektrických obvodů maticovým počtem. ACADEMIA Praha, 1966.

[1.9] MORRIS,N.M.: Mastering Electronic and Electrical Calculations. MacMillan, Press Ltd., 1996.

[1.10] ČAJKA,J.-KVASIL,J.: Teorie lineárních obvodů. TKI, SNTL/ALFA 1979.

[1.11] LEVINŠTEJN, M.L.: Operátorový počet v elektrotechnice. SNTL , Praha 1977.

[1.12] PUNČOCHÁŘ,J,: Operační zesilovače – historie a současnost. BEN, 2002.

[1.13] BIOLEK,D.: Elektrické systémy. Skriptum VA Brno, S-1589, 1995, 83s.

[1.14] BIOLEK,D.: Elektrické signály a systémy. S-2584, VA Brno, 1998.

[1.15] MASON, S.J.: Feedback Theory: Some Properties of Signal-flow Graphs. Proc. IRE, Vol. 41, pp. 1144-1156, September 1953.

[1.16] MASON, S.J.: Feedback Theory: Further Properties of Signal-flow Graphs. Proc. IRE, Vol. 44, No. 7, pp. 920-926, July 1956.

[1.17] COATES, C.L.: Flow-Graph Solutions of Linear Algebraic Equations. IRE Trans. on CT, Vol. 6, No. 2, pp. 170-187, June 1959.

[1.18] BIOLEK,D.: Analýza elektronických obvodů (nejen) na počítači. Slaboproudý obzor, Vol. 58, č. 4, prosinec 2001, s.25-31.

[1.19] BIOLKOVÁ,V.-BIOLEK,D.: Stamp-Based M-C Graphs of Current Conveyors. Internet Journal ElectronicsLetters.com, November 2002.
K dispozici na http://www.ElectronicsLetters.com/papers/2002/0018/paper.asp

[1.20] BIOLEK,D.-BIOLKOVÁ,V.: Signal Flow Graphs Suitable For Teaching Circuit Analysis (Grafy signálových toků vhodné pro výuku analýzy obvodů). Radioelektronika2001, Brno, s. 310-313.
Česká verze je k dispozici na
http://www.vabo.cz/Stranky/biolek/veda/articles/RADIO01_C.pdf

[1.21] BIOLEK,D.-BIOLKOVÁ,V.: Analysis of Circuits Containing Active Elements by Using Modified T - Graphs (Analýza obvodů s aktivními prvky pomocí modifikovaných T- grafů). Konference CSCC01, Kréta, s. 4431-4435.

Česká verze je k dispozici na
http://www.vabo.cz/Stranky/biolek/veda/articles/CSCC01_C.pdf
[1.22] BIOLEK,D.-BIOLKOVÁ,V.: Modelling and Optimization of Active Filters by Hybrid "VIV-MMC"-graphs (Modelování a optimalizace aktivních filtrů hybridními "VIV-MMC" grafy). CSCC'00 Vouliagmeni, Athens, 2000, pp. 1321-1326.
K dispozici na http://www.vabo.cz/Stranky/biolek/veda/articles/CSCC00_1.pdf
[1.23] BIOLEK,D.: Modelování a počítačová simulace-P. Elektronický učební text FEKT VUT Brno.

2. Knihy a další literatura o moderních elektronických součástkách a jejich aplikacích v analogovém zpracování signálů

[2.1] DOSTÁL, J.: Operační zesilovače. SNTL Praha, 1981.
[2.2] SCHAUMANN, R.-GHAUSI, M.S.-LAKER, K.R.: Design of Analog Filters. Prentice Hall, New Jersey, 1990. ISBN 0-13-200288-4.
[2.3] UNBEHAUEN, R.-CICHOCKI, A.: MOS Switched-Capacitor and Continuous-Time Integrated Circuits and Systems. Springer-Verlag, Berlin, 1989. ISBN 3-540-50599-7.
[2.4] TOUMAZOU, C. at all: Circuits& Systems. Tutorials. IEEE, ISCAS'94. LTP Electronics Ltd., Oxford, 1994.
[2.5] HÁJEK, K.-SEDLÁČEK, J.: Kmitočtové filtry. BEN, 2002.
[2.6] PUNČOCHÁŘ,J.: Operační zesilovače v elektronice (páté vydání). BEN, Praha 2002.
[2.7] SEIDELMANN,L.: Nové zapojení operačního usměrňovače. Sdělovací technika, 12/98, s. 12-13.

3. Knihy, skripta, učební texty a články o řešení obvodů pomocí počítače

[3.1] LÁNÍČEK, R.: Simulační programy pro elektrotechniku. BEN, Praha 2000.
[3.2] VLADIMIRESCU,A.: The SPICE Book. John Willey & Sons, Inc., 1994.
[3.3] KIELKOWSKI,R.: Inside SPICE. McGraw-Hill, 1998.
[3.4] MANN,H.: Využití počítače v elektrotechnických návrzích. SNTL Praha, 1984.
[3.5] VLACH,J.-SINGHAL,K.: Computer Methods for Circuit Analysis and Design. Van Nostrand Reinhold Company, New York, 1982. K dispozici je též ruský překlad Mašinnyje metody analiza i projektirovanija elektronnych schem, Moskva, Radio i Svjaz, 1988.
[3.6] KUNEŠ,J. a kol.: Základy modelování. TKI, SNTL Praha 1989.
[3.7] DOBEŠ,J.: Návrh vysokofrekvenčních a mikrovlnných obvodů počítačem. Skriptum FEL ČVUT Praha, 2003.
[3.8] KEJHAR,M. a kol.: Program SPICE v příkladech. Skriptum ČVUT Praha, 1995.
[3.9] MANN,H.: Podpora projektování dynamických soustav program DYNAST: SADYS. ČSVTS při ČVUT Praha, 1990.
[3.10] BIOLEK,D. a kol.: TERO. Využití počítačových programů v elektrotechnice. Skriptum VA Brno, S-1738, 1992, 124s.
[3.11] BIOLEK,D.: Počítačová cvičení v teorii obvodů. Skriptum VA Brno, S-803, 1995, 89s.
[3.12] BIOLEK,Z.-BIOLEK,D.: MICROCAP IV. Program pro analýzu elektrických obvodů. STUDENT-L. Lupress s.r.o., učební text SPŠE Rožnov p.R., 1996.
[3.13] BIOLEK,D.: "Behaviorální" modelování v programu MicroCap VI. ELEKTROREVUE, červen 2000.
K dispozici na http://www.elektrorevue.cz/clanky/00021/index.htm

[3.14] DOBEŠ,J.: Analýza nelineárních statických a dynamických elektronických obvodů. 1. seminář "Spolupráce vysokých a středních škol", Pardubice , 13. říjen 1999, s. 13-18.
[3.15] AC Analysis Convergence Technique. Spectrum News, Summer 1998, s. 15-17. K dispozici nahttp://www.spectrum-soft.com.

4. Obecné otázky využití počítačů ve výuce

[4.1] BIOLEK,D.: Výuka obecných metod analýzy lineárních obvodů. STO-5, VA Brno, 1994, s.1-4.
K dispozici na http://www.vabo.cz/Stranky/biolek/veda/articles/STO5_2.pdf
[4.2] BIOLEK,D.: Respektování didaktických principů při využívání počítačových programů ve výuce elektrických obvodů. ELEKTROREVUE, prosinec 1999.
K dispozici na http://www.elektrorevue.cz/clanky/99008/index.htm
[4.3] BIOLEK,D.: Využití programů pro symbolickou a semisymbolickou analýzu elektrických obvodů ve výuce i výzkumu. ELEKTROREVUE, prosinec 1999.
K dispozici na http://www.elektrorevue.cz/clanky/99012/index.htm
[4.4] BIOLEK,D.: Program SNAP v. 2.6: Nové možnosti pro výuku i výzkum. STO-7, VA Brno, 1999, s. 66-69. ISBN 80-214-1392-1.
K dispozici na http://www.vabo.cz/Stranky/biolek/veda/articles/STO7_3.pdf
[4.5] BIOLEK,D.-KOLKA,Z.: Využití programu SNAP 2.6 ve vybraných elektrotechnických předmětech. STO-7, VA Brno, 1999, s. 70-75. ISBN 80-214-1392-1.
K dispozici na http://www.vabo.cz/Stranky/biolek/veda/articles/STO7_2.pdf
[4.6] BIOLEK,D.: Využití programů pro symbolickou a semisymbolickou analýzu ve výuce elektrických obvodů. 1. seminář "Spolupráce vysokých a středních škol", Pardubice , 13. říjen 1999, s. 19-24. ISBN 80-902417-5-1.
K dispozici na http://www.vabo.cz/Stranky/biolek/veda/articles/PARDUBICE99.pdf
[4.7] BOREŠ,P.-MARTINEK,P.: Výuka a počítač. Seminář STO-6 Moderní směry výuky elektrotechniky a elektroniky, VA Brno, 1997, s. 125-128.
[4.8] BIOLEK,Z.: Počítačové experimenty ve výuce elektrotechnických předmětů. Seminář STO-6 Moderní směry výuky elektrotechniky a elektroniky, VA Brno, 1997, s. 129-131.
[4.9] DOSTÁL,T,-PODROUŽEK,V.: Zkušenosti s výukou a použitím výpočetní techniky v kursu Navrhování elektronických obvodů. Seminář STO-6 Moderní směry výuky elektrotechniky a elektroniky, VA Brno, 1997, s. 136-139.
[4.10] BIOLEK,Z.: Podpora výuky základů elektrotechniky pomocí programu MC5. Seminář STO-7 Moderní směry výuky elektrotechniky a elektroniky, VA Brno, 1999, s. 86-88.
[4.11] BIOLEK,D.-BIOLKOVÁ,V.-VRBA,K.: Teaching Linear Circuits analysis effectively. 4th UICEE Annual Conference on Engineering Education, Bangkok, Thailand, 7-10 February 2001, pp. 277-280.
K dispozici na http://www.vabo.cz/Stranky/biolek/veda/articles/UICEE01_1.pdf
[4.12] BIOLEK,D.-KOLKA,Z.-SVIEZENY,B.: Teaching of Electrical Circuits using Symbolic and Semisymbolic Programs. EAEEIE Ulm, Germany, April 2000, pp. 26-30.
K dispozici na http://www.vabo.cz/Stranky/biolek/veda/articles/EAEEIE00.pdf

5. Knihy o numerických metodách

[5.1] RALSTON,A.: Základy numerické matematiky. ACADEMIA, Praha 1973.

[5.2] PRESS,W.H. et al.: Numerical Recipes in Pascal. The Art of Scientific Computing. Cambridge University Press, 1994.
Dostupné na http://www.nr.com
[5.3] ACTON, F.S.: Numerical Methods that Work. The Mathematical Association of America, Washington D.C., 1990.

6. Domovské stránky některých simulačních a analyzačních programů

[6.1] MicroCap – http://www.spectrum-soft.com
[6.2] Electronics Workbench a Multisim – http://www.cadware.cz/cad204.htm, http://www.electronicsworkbench.com/
[6.3] PSpice – http://pcb.cadence.com/company/move.asp, http://www.ee.mtu.edu/faculty/rzulinsk/pspice.htm
[6.4] TINA – http://www.designsoftware.com/TINA.HTM
[6.5] SNAP – http://www.fee.vutbr.cz/UREL/snap
[6.6] Programy Prof. Valsy – http://www.feec.vutbr.cz/UTEE/OBVODY/index.html
[6.7] Dynast - http://virtual.cvut.cz/cacsd/msa/dynast.html
[6.8] Programy využívané na FEL ČVUT v Praze – http://hippo.feld.cvut.cz/~bores/prog/uvod.htm