

# **HSPICE® Reference Manual: Commands and Control Options**

---

Version E-2010.12, December 2010

**SYNOPTSYS®**

# Copyright Notice and Proprietary Information

Copyright © 2010 Synopsys, Inc. All rights reserved. This software and documentation contain confidential and proprietary information that is the property of Synopsys, Inc. The software and documentation are furnished under a license agreement and may be used or copied only in accordance with the terms of the license agreement. No part of the software and documentation may be reproduced, transmitted, or translated, in any form or by any means, electronic, mechanical, manual, optical, or otherwise, without prior written permission of Synopsys, Inc., or as expressly provided by the license agreement.

## Right to Copy Documentation

The license agreement with Synopsys permits licensee to make copies of the documentation for its internal use only. Each copy shall include all copyrights, trademarks, service marks, and proprietary rights notices, if any. Licensee must assign sequential numbers to all copies. These copies shall contain the following legend on the cover page:

“This document is duplicated with the permission of Synopsys, Inc., for the exclusive use of \_\_\_\_\_ and its employees. This is copy number \_\_\_\_\_.”

## Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

## Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

## Registered Trademarks (®)

Synopsys, AMPS, Astro, Behavior Extracting Synthesis Technology, Cadabra, CATS, Certify, CHIPit, CoMET, Design Compiler, DesignWare, Formality, Galaxy Custom Designer, HAPS, HapsTrak, HDL Analyst, HSIM, HSPICE, Identify, Leda, MAST, METeor, ModelTools, NanoSim, OpenVera, PathMill, Physical Compiler, PrimeTime, SCOPE, Simply Better Results, SiVL, SNUG, SolvNet, Syndicated, Synplicity, the Synplicity logo, Synplify, Synplify Pro, Synthesis Constraints Optimization Environment, TetraMAX, UMRBus, VCS, Vera, and YIELDirector are registered trademarks of Synopsys, Inc.

## Trademarks (™)

AFGen, Apollo, Astro-Rail, Astro-Xtalk, Aurora, AvanWaves, BEST, Columbia, Columbia-CE, Confirma, Cosmos, CosmosLE, CosmosScope, CRITIC, CustomExplorer, CustomSim, DC Expert, DC Professional, DC Ultra, Design Analyzer, Design Vision, DesignerHDL, DesignPower, DFTMAX, Direct Silicon Access, Discovery, Eclipse, Encore, EPIC, Galaxy, HANEX, HDL Compiler, Hercules, Hierarchical Optimization Technology, High-performance ASIC Prototyping System, HSIM<sup>plus</sup>, i-Virtual Stepper, IICE, in-Sync, iN-Tandem, Jupiter, Jupiter-DP, JupiterXT, JupiterXT-ASIC, Liberty, Libra-Passport, Library Compiler, Magellan, Mars, Mars-Rail, Mars-Xtalk, Milkyway, ModelSource, Module Compiler, MultiPoint, Physical Analyst, Planet, Planet-PL, Polaris, Power Compiler, Raphael, Saturn, Scirocco, Scirocco-i, Star-RCXT, Star-SimXT, StarRC, System Compiler, System Designer, Taurus, TotalRecall, TSUPREM-4, VCS Express, VCSi, VHDL Compiler, VirSim, and VMC are trademarks of Synopsys, Inc.

## Service Marks (sm)

MAP-in, SVP Café, and TAP-in are service marks of Synopsys, Inc.

SystemC is a trademark of the Open SystemC Initiative and is used under license.

ARM and AMBA are registered trademarks of ARM Limited.

Saber is a registered trademark of SabreMark Limited Partnership and is used under license.

All other product or company names may be trademarks of their respective owners.

# Contents

---

---

<b>1. HSPICE and HSPICE RF Application Commands</b> .....	1
hspice .....	2
Examples of Starting HSPICE .....	9
Using HSPICE for Calculating New Measurements .....	10
hspicrf .....	12

---

<b>2. HSPICE and HSPICE RF Netlist Commands</b> .....	13
HSPICE and RF Commands Overview .....	14
Analysis .....	14
Alter Block .....	15
Conditional Block .....	15
Digital Vector .....	16
Encryption .....	16
Field Solver .....	16
Files .....	16
Input/Output Buffer Information Specification (IBIS) .....	17
Library Management .....	17
Model and Variation Definition .....	17
Node Naming .....	17
Output Porting .....	18
Setup .....	18
Simulation Runs .....	18
Subcircuits .....	18
Verilog-A .....	19
Alphabetical Listing of Commands .....	19
.AC .....	25
.ACMATCH .....	28
.ACPHASENOISE .....	30
.ALIAS .....	31
.ALTER .....	33
.APPENDMODEL .....	35
.BA_ACHECK .....	36

## Contents

.BIASCHK .....	38
.CFL_PROTOTYPE .....	44
.CHECK EDGE .....	48
.CHECK FALL .....	50
.CHECK GLOBAL_LEVEL .....	51
.CHECK HOLD .....	52
.CHECK IRDROP .....	54
.CHECK RISE .....	56
.CHECK SETUP .....	58
.CHECK SLEW .....	59
.CONNECT .....	61
.DATA .....	64
.DC .....	71
.DCMATCH .....	76
.DCSENS .....	78
.DCVOLT .....	80
.DEL LIB .....	81
.DESIGN_EXPLORATION .....	84
.DISTO .....	86
.DOUT .....	88
.EBD .....	90
.ELSE .....	92
.ELSEIF .....	93
.END .....	94
.ENDDATA .....	95
.ENDIF .....	96
.ENDL (or) .ENDL TT .....	97
.ENDS .....	98
.ENV .....	99
.ENVFFT .....	100
.ENVOSC .....	101
.EOM .....	102
.FFT .....	103
.FLAT .....	108
.FOUR .....	110
.FSOPTIONS .....	111
.GLOBAL .....	114
.HB .....	115
.HBAC .....	118

## Contents

.HBLIN .....	119
.HBLSP .....	121
.HBNOISE .....	123
.HBOSC .....	126
.HBXF .....	131
.HDL .....	132
.IBIS .....	134
.IC .....	138
.ICM .....	140
.IF .....	142
.INCLUDE (or) .INC (or) .INCL .....	144
.IVTH .....	145
.JITTER .....	147
.LAYERSTACK .....	148
.LIB .....	150
.LIN .....	154
.LOAD .....	158
.LPRINT .....	160
.LSTB .....	161
.MACRO .....	164
.MALIAS .....	167
.MATERIAL .....	169
.MEASURE (or) .MEAS .....	171
.MEASURE (Rise, Fall, Delay, and Power Measurements) .....	173
.MEASURE (FIND and WHEN) .....	178
.MEASURE (Continuous Results) .....	182
.MEASURE (Equation Evaluation/Arithmetic Expression) .....	185
.MEASURE (AVG, EM_AVG, INTEG, MIN, MAX, PP, and RMS) .....	187
.MEASURE (Integral Function) .....	191
.MEASURE (Derivative Function) .....	193
.MEASURE (Error Function) .....	196
.MEASURE PHASENOISE .....	198
.MEASURE PTDNOISE .....	202
.MEASURE (Pushout Bisection) .....	203
.MEASURE (ACMATCH) .....	205
.MEASURE (DCMATCH) .....	206
.MEASURE FFT .....	208
.MEASURE LSTB .....	211
.MODEL .....	213

## Contents

.MODEL_INFO . . . . .	221
.MOSRA . . . . .	222
.MOSRAPRINT . . . . .	226
.MOSRA_SUBCKT_PIN_VOLT . . . . .	227
.NODESET . . . . .	228
.NOISE . . . . .	230
.OP . . . . .	233
.OPTION (or) .OPTIONS . . . . .	235
.PARAM (or) .PARAMETER (or) .PARAMETERS . . . . .	237
.PAT . . . . .	241
.PHASENOISE . . . . .	244
.PKG . . . . .	247
.POWER . . . . .	249
.POWERDC . . . . .	251
.PRINT . . . . .	252
.PROBE . . . . .	255
.PROTECT or .PROT . . . . .	258
.PTDNOISE . . . . .	259
.PZ . . . . .	263
.SAMPLE . . . . .	265
.SAVE . . . . .	266
.SENS . . . . .	268
.SHAPE . . . . .	270
.SHAPE (Defining Rectangles) . . . . .	271
.SHAPE (Defining Circles) . . . . .	272
.SHAPE (Defining Polygons) . . . . .	273
.SHAPE (Defining Strip Polygons) . . . . .	275
.SHAPE (Defining Trapezoids) . . . . .	276
.SN . . . . .	277
.SNAC . . . . .	279
.SNFT . . . . .	280
.SNNOISE . . . . .	283
.SNOSC . . . . .	285
.SNXF . . . . .	288
.STATEYE . . . . .	290
.STIM . . . . .	294
.STORE . . . . .	298
.SUBCKT . . . . .	301
.SURGE . . . . .	306

.SWEEPBLOCK .....	307
.TEMP (or) .TEMPERATURE .....	309
.TF .....	312
.TITLE .....	313
.TRAN .....	314
.TRANNOISE .....	320
.UNPROTECT or .UNPROT .....	324
.VARIATION .....	325
.VEC .....	327

---

<b>3. HSPICE and RF Netlist Simulation Control Options .....</b>	<b>329</b>
HSPICE Control Options Grouped By Function .....	330
General Control Options .....	330
Input/Output Controls .....	331
Model Analysis .....	332
HSPICE Analysis Options .....	334
Transient and AC Small Signal Analysis Options .....	334
HSPICE RF Analysis Options .....	336
HB Options .....	336
Phase Noise Analysis .....	336
Power Analysis .....	337
RC Network Reduction .....	337
Simulation Output .....	337
Shooting Newton Options .....	337
DSPF Options .....	338
SPEF Options .....	338
Transient Accuracy Options .....	338
Alphabetical Listing of HSPICE Control Options .....	339
.OPTION ABSH .....	352
.OPTION ABSI .....	353
.OPTION ABSIN .....	354
.OPTION ABSMOS .....	355
.OPTION ABSTOL .....	356
.OPTION ABSV .....	357
.OPTION ABSVAR .....	358
.OPTION ABSVDC .....	359
.OPTION ACCURATE .....	360
.OPTION ACOUT .....	361

## Contents

.OPTION ALTCC .....	362
.OPTION ALTCHK .....	363
.OPTION APPENDALL .....	364
.OPTION ARTIST .....	366
.OPTION ASPEC .....	367
.OPTION AUTOSTOP (or) .OPTION AUTOST .....	368
.OPTION BA_ACTIVE .....	370
.OPTION BA_ACTIVEHIER .....	371
.OPTION BA_ADDPARAM .....	372
.OPTION BA_COUPLING .....	373
.OPTION BA_DPFPFX .....	374
.OPTION BA_ERROR .....	375
.OPTION BA_FILE .....	376
.....	
.OPTION BA_FINGERDELIM .....	377
.OPTION BA_GEOSHRINK .....	378
.....	
.OPTION BA_HIERDELIM .....	379
.....	
.OPTION BA_IDEALPFX .....	380
.OPTION BA_MERGEPORT .....	381
.OPTION BA_NETFMT .....	382
.OPTION BA_PRINT .....	383
.OPTION BA_SCALE .....	384
.OPTION BA_TERMINAL .....	385
.OPTION BADCHR .....	387
.OPTION BDFATOL .....	388
.OPTION BDFRTOL .....	390
.OPTION BEEP .....	392
.OPTION BIASFILE .....	393
.OPTION BIASINTERVAL .....	394
.OPTION BIASNODE .....	395
.OPTION BIASPARALLEL .....	396
.OPTION BIAWARN .....	397
.OPTION BINPRNT .....	398
.OPTION BPNMATCHTOL .....	399
.OPTION BRIEF .....	400
.OPTION BSIM4PDS .....	401
.OPTION BYPASS .....	402



## Contents

.OPTION BYTOL .....	403
.OPTION CAPTAB .....	404
.OPTION CFLFLAG .....	405
.OPTION CHGTOL .....	406
.OPTION CMIMCFLAG .....	407
.OPTION CMIFLAG .....	408
.OPTION CMIPATH .....	409
.OPTION CMIUSRFLAG .....	410
.OPTION CONVERGE .....	412
.OPTION CPTIME .....	413
.OPTION CSCAL .....	414
.OPTION CSDF .....	415
.OPTION CSHDC .....	416
.OPTION CSHUNT .....	417
.OPTION CUSTCMI .....	418
.OPTION CVTOL .....	419
.OPTION D_IBIS .....	420
.OPTION DCAP .....	421
.OPTION DCCAP .....	422
.OPTION DCFOR .....	423
.OPTION DCHOLD .....	424
.OPTION DCIC .....	425
.OPTION DCON .....	426
.OPTION DCSTEP .....	427
.OPTION DCTRAN .....	428
.OPTION DEFAD .....	429
.OPTION DEFAS .....	430
.OPTION DEFL .....	431
.OPTION DEFNRD .....	432
.OPTION DEFNRS .....	433
.OPTION DEFPPD .....	434
.OPTION DEFPS .....	435
.OPTION DEFSA .....	436
.OPTION DEFSB .....	437
.OPTION DEFSD .....	438
.OPTION DEFW .....	439
.OPTION DEGF .....	440
.OPTION DEGFN .....	441
.OPTION DEGFP .....	442

## Contents

.OPTION DELMAX	443
.OPTION DI	444
.OPTION DIAGNOSTIC (or) .OPTION DIAGNO	445
.OPTION DLENCSDF	446
.OPTION DV	447
.OPTION DVDT	448
.OPTION DVTR	449
.OPTION DYNACC	450
.OPTION EM_RECOVERY	451
.OPTION EPSMIN	452
.OPTION EXPLI	453
.OPTION EXPMAX	454
.OPTION FAST	455
.OPTION FFT_ACCURATE	456
.OPTION FFTOUT	457
.OPTION FMAX	458
.OPTION FS	459
.OPTION FSCAL	460
.OPTION FT	461
.OPTION GDCPATH	462
.OPTION GENK	463
.OPTION GEOSHRINK	464
.OPTION GMAX	465
.OPTION GMIN	466
.OPTION GMINDC	467
.OPTION GRAMP	468
.OPTION GSCAL	469
.OPTION GSHDC	470
.OPTION GSHUNT	471
.OPTION HBACKRYLOVDIM	472
.OPTION HBACKRYLOVITER (or) HBAC_KRYLOV_ITER	473
.OPTION HBACTOL	474
.OPTION HBCONTINUE	475
.OPTION HBFREQABSTOL	476
.OPTION HBFREQRELTOL	477
.OPTION HB_GIBBS	478
.OPTION HBJREUSE	479
.OPTION HBJREUSETOL	480
.OPTION HBKRYLOVDIM	481

.OPTION HBKRYLOVTOL .....	482
.OPTION HBKRYLOVMAXITER (or) HB_KRYLOV_MAXITER .....	483
.OPTION HBLINESEARCHFAC .....	484
.OPTION HBMAXITER (or) HB_MAXITER .....	485
.OPTION HBOSCMAXITER (or) HBOSC_MAXITER .....	486
.OPTION HBPROBETOL .....	487
.OPTION HBSOLVER .....	488
.OPTION HBTOL .....	489
.OPTION HBTRANFREQSEARCH .....	490
.OPTION HBTRANINIT .....	491
.OPTION HBTRANPTS .....	492
.OPTION HBTRANSTEP .....	493
.OPTION HIER_DELIM .....	494
.OPTION HIER_SCALE .....	495
.OPTION IC_ACCURATE .....	497
.OPTION ICSWEEP .....	498
.OPTION IMAX .....	499
.OPTION IMIN .....	500
.OPTION INGOLD .....	501
.OPTION INTERP .....	503
.OPTION IPROP .....	504
.OPTION ITL1 .....	505
.OPTION ITL2 .....	506
.OPTION ITL3 .....	507
.OPTION ITL4 .....	508
.OPTION ITL5 .....	509
.OPTION ITLPTRAN .....	510
.OPTION ITLPZ .....	511
.OPTION ITRPRT .....	512
.OPTION IVTH .....	513
.OPTION KCLTEST .....	514
.OPTION KLIM .....	515
.OPTION LA_FREQ .....	516
.OPTION LA_MAXR .....	517
.OPTION LA_MINC .....	518
.OPTION LA_TIME .....	519
.OPTION LA_TOL .....	520
.OPTION LENNAM .....	521
.OPTION LIMPTS .....	522

## Contents

.OPTION LIMTIM . . . . .	523
.OPTION LISLVL . . . . .	524
.OPTION LIS_NEW . . . . .	525
.OPTION LIST . . . . .	526
.OPTION LOADHB . . . . .	527
.OPTION LOADSNINIT . . . . .	528
.OPTION LSCAL . . . . .	529
.OPTION LVLTIM . . . . .	531
.OPTION MACMOD . . . . .	532
.OPTION MAXAMP . . . . .	534
.OPTION MAXORD . . . . .	535
.OPTION MAXWARNS . . . . .	536
.OPTION MBYPASS . . . . .	537
.OPTION MCBRIEF . . . . .	538
.OPTION MEASDGT . . . . .	539
.OPTION MEASFAIL . . . . .	540
.OPTION MEASFILE . . . . .	541
.OPTION MEASFORM. . . . .	542
.OPTION MEASOUT . . . . .	544
.OPTION MESSAGE_LIMIT . . . . .	545
.OPTION METHOD . . . . .	546
.OPTION MODMONTE . . . . .	549
.OPTION MODPRT . . . . .	551
.OPTION MONTECON . . . . .	553
.OPTION MOSRALIFE. . . . .	554
.OPTION MOSRASORT . . . . .	555
.OPTION MRAAPI . . . . .	556
.OPTION MRAEXT . . . . .	557
.OPTION MRAPAGED . . . . .	558
.OPTION MRA00PATH, MRA01PATH, MRA02PATH, MRA03PATH . . . .	559
.OPTION MTTRESH . . . . .	560
.OPTION MU . . . . .	561
.OPTION NCFILTER . . . . .	562
.OPTION NCWARN . . . . .	563
.OPTION NEWTOL . . . . .	564
.OPTION NODE. . . . .	565
.OPTION NOELCK. . . . .	566
.OPTION NOISEMINFREQ . . . . .	567
.OPTION NOMOD . . . . .	568

## Contents

.OPTION NOPIV .....	569
.OPTION NOTOP .....	570
.OPTION NOWARN .....	571
.OPTION NUMDGT .....	572
.OPTION NUMERICAL_DERIVATIVES .....	573
.OPTION NXX .....	574
.OPTION OFF .....	575
.OPTION OPFILE .....	576
.OPTION OPTCON .....	577
.OPTION OPTLST .....	579
.OPTION OPTPARHIER .....	580
.OPTION OPTS .....	581
.OPTION PARHIER (or) .OPTION PARHIE .....	582
.OPTION PATHNUM .....	583
.OPTION PCB_SCALE_FORMAT .....	584
.OPTION PHASENOISEKRYLOVDIM .....	586
.OPTION PHASENOISEKRYLOVITER (or) PHASENOISE_KRYLOV_ITER .....	587
.OPTION PHASENOISETOL .....	588
.OPTION PHASETOLI .....	589
.OPTION PHASETOLV .....	590
.OPTION PHD .....	591
.OPTION PHNOISELORENTZ .....	592
.OPTION PHNOISEAMPM .....	593
.OPTION PIVOT .....	594
.OPTION PIVTOL .....	595
.OPTION POST .....	596
.OPTION POSTLVL .....	598
.OPTION POST_VERSION .....	599
.OPTION POSTTOP .....	601
.OPTION PROBE .....	602
.OPTION PSF .....	603
.OPTION PURETP .....	604
.OPTION PUTMEAS .....	605
.OPTION PZABS .....	606
.OPTION PZTOL .....	607
.OPTION RADEGFILE .....	608
.OPTION RADEGOUTPUT .....	609
.OPTION RANDGEN .....	610
.OPTION REDEFSUB .....	611

## Contents

.OPTION RELH	612
.OPTION RELI	613
.OPTION RELIN	614
.OPTION RELMOS	615
.OPTION RELQ	616
.OPTION RELTOL	617
.OPTION RELV	618
.OPTION RELVAR	619
.OPTION RELVDC	620
.OPTION REPLICATES	621
.OPTION RES_BITS	622
.OPTION RESMIN	623
.OPTION RISETIME (or) .OPTION RISETI	624
.OPTION RITOL	626
.OPTION RMAX	627
.OPTION RMIN	628
.OPTION RUNLVL	629
.OPTION SAMPLING_METHOD	633
.OPTION SAVEHB	635
.OPTION SAVESNINIT	636
.OPTION SCALE	637
.OPTION SCALM	638
.OPTION SEARCH	639
.OPTION SEED	640
.OPTION SHRINK	641
.OPTION SIM_ACCURACY	642
.OPTION SIM_DELTAI	643
.OPTION SIM_DELTAV	644
.OPTION SIM_DSPF	645
.OPTION SIM_DSPF_ACTIVE	647
.OPTION SIM_DSPF_INSERTOR	648
.OPTION SIM_DSPF_LUMPCAPS	649
.OPTION SIM_DSPF_MAX_ITER	650
.OPTION SIM_DSPF_RAIL	651
.OPTION SIM_DSPF_SCALEC	652
.OPTION SIM_DSPF_SCALER	653
.OPTION SIM_DSPF_VTOL	654
.OPTION SIM_LA	656
.OPTION SIM_LA_FREQ	657

.OPTION SIM_LA_MAXR .....	658
.OPTION SIM_LA_MINC .....	659
.OPTION SIM_LA_TIME .....	660
.OPTION SIM_LA_TOL .....	661
.OPTION SIM_ORDER .....	662
.OPTION SIM_OSC_DETECT_TOL .....	663
.OPTION SIM_POSTAT .....	664
.OPTION SIM_POSTDOWN .....	665
.OPTION SIM_POSTSCOPE .....	666
.OPTION SIM_POSTSKIP .....	667
.OPTION SIM_POSTTOP .....	668
.OPTION SIM_POWER_ANALYSIS .....	669
.OPTION SIM_POWER_TOP .....	670
.OPTION SIM_POWERDC_ACCURACY .....	671
.OPTION SIM_POWERDC_HSPICE .....	672
.OPTION SIM_POWERPOST .....	673
.OPTION SIM_POWERSTART .....	674
.OPTION SIM_POWERSTOP .....	675
.OPTION SIM_SPEF .....	676
.OPTION SIM_SPEF_ACTIVE .....	677
.OPTION SIM_SPEF_INSERTOR .....	678
.OPTION SIM_SPEF_LUMPCAPS .....	679
.OPTION SIM_SPEF_MAX_ITER .....	680
.OPTION SIM_SPEF_PARVALUE .....	681
.OPTION SIM_SPEF_RAIL .....	682
.OPTION SIM_SPEF_SCALEC .....	683
.OPTION SIM_SPEF_SCALER .....	684
.OPTION SIM_SPEF_VTOL .....	685
.OPTION SIM_TG_THETA .....	686
.OPTION SIM_TRAP .....	687
.OPTION SI_SCALE_SYMBOLS .....	688
.OPTION SLOPETOL .....	689
.OPTION SNACCURACY .....	690
.OPTION SNCONTINUE .....	691
.OPTION SNMAXITER (or) SN_MAXITER .....	692
.OPTION SOIQ0 .....	693
.OPTION SPLIT_DP .....	694
.OPTION SPMODEL .....	696
.OPTION STATFL .....	697

## Contents

.OPTION STRICT_CHECK .....	698
.OPTION SX_FACTOR .....	699
.OPTION SYMB .....	700
.OPTION TIMERES .....	701
.OPTION TMIFLAG .....	702
.OPTION TMIPATH .....	703
.OPTION TMIVERSION .....	704
.OPTION TNOM .....	705
.OPTION TRANFORHB .....	706
.OPTION TRCON .....	707
.OPTION TRTOL .....	708
.OPTION UNWRAP .....	709
.OPTION VAMODEL .....	710
.OPTION VERIFY .....	711
.OPTION VFLOOR .....	712
.OPTION VNTOL .....	713
.OPTION WACC .....	714
.OPTION WARN .....	715
.OPTION WARN_SEP .....	716
.OPTION WARNLIMIT (or) .OPTION WARNLIM .....	717
.OPTION WAVE_POP .....	718
.OPTION WDELAYOPT .....	719
.OPTION WDF .....	720
.OPTION WINCLUDEGDIMAG .....	722
.OPTION WL .....	723
.OPTION WNFLAG .....	724
.OPTION XDTEMP .....	725
.OPTION (X0R,X0I) .....	727
.OPTION (X1R,X1I) .....	728
.OPTION (X2R,X2I) .....	729
.VARIATION Block Control Options .....	730
.DESIGN_EXPLORATION Block Control Options .....	732

---

<b>4. Digital Vector File Commands .....</b>	<b>735</b>
ENABLE .....	736
IDELAY .....	737
IO .....	739



ODELAY.....	740
OUT or OUTZ .....	742
PERIOD.....	743
RADIX .....	744
SLOPE.....	745
TDELAY.....	747
TFALL .....	749
TRISE .....	751
TRIZ.....	753
TSKIP.....	754
TUNIT .....	755
VIH.....	757
VIL .....	758
VNAME .....	759
VOH.....	761
VOL .....	763
VREF .....	765
VTH .....	766

---

<b>A. Obsolete Commands and Options.....</b>	<b>769</b>
.ACDCFACTOR .....	770
.GRAPH .....	771
.MODEL Command for .GRAPH .....	773
.NET.....	775
.PLOT .....	777
.WIDTH .....	779
.OPTION ACCT .....	780
.OPTION ALT999 or ALT9999 .....	781
.OPTION BKPSIZ .....	782
.OPTION CDS .....	783
.OPTION CO .....	784

## Contents

.DEGINFO .....	785
.OPTION H9007 .....	786
.OPTION MEASSORT .....	787
.OPTION MENTOR .....	788
.OPTION MODSRH .....	789
.OPTION PIVREF .....	791
.OPTION NOPAGE .....	792
.OPTION PIVREL .....	793
.OPTION PLIM .....	794
.OPTION SDA .....	795
.OPTION SPICE .....	796
.OPTION SIM_LA_MINMODE .....	797
.OPTION ZUKEN .....	798

---

<b>B. How Options Affect other Options .....</b>	<b>799</b>
GEAR Method .....	800
ACCURATE .....	800
FAST .....	800
GEAR Method, ACCURATE .....	801
ACCURATE, GEAR Method .....	802
ACCURATE, FAST .....	802
GEAR Method, FAST .....	803
GEAR Method, ACCURATE, FAST .....	804
RUNLVL=N .....	804
RUNLVL, ACCURATE, FAST, GEAR method .....	805
DVDT=1,2,3 .....	805
LVLTIM=0,2,3 .....	805
KCLTEST .....	806
BRIEF .....	806
Option Notes .....	806
RUNLVL Option Notes .....	807

## Contents

Finding the Golden Reference for Options. . . . .	808
---	-----

---

<b>Index</b> . . . . .	<b>809</b>
------------------------	------------

## Contents

# About This Manual

---

This manual describes the individual HSPICE commands you can use to simulate and analyze your circuit designs.

---

## Inside This Manual

This manual contains the chapters described below. For descriptions of the other manuals in the HSPICE documentation set, see the next section, [The HSPICE Documentation Set](#).

Chapter	Description
<a href="#">Chapter 1, HSPICE and HSPICE RF Application Commands</a>	Describes the commands you use to start HSPICE or HSPICE RF, including syntax, arguments, and examples.
<a href="#">Chapter 2, HSPICE and HSPICE RF Netlist Commands</a>	Describes the commands you can use in HSPICE and HSPICE RF netlists.
<a href="#">Chapter 3, HSPICE and RF Netlist Simulation Control Options</a>	Describes the HSPICE and HSPICE RF simulation control options you can set using various forms of the .OPTION command.
<a href="#">Chapter 4, Digital Vector File Commands</a>	Contains an alphabetical listing of the HSPICE commands you can use in a digital vector file.
<a href="#">Appendix A, Obsolete Commands and Options</a>	Describes commands and options no longer commonly used in HSPICE.
<a href="#">Appendix B, How Options Affect other Options</a>	Describes the effects of specifying control options on other options in the netlist.

---

## The HSPICE Documentation Set

This manual is a part of the HSPICE documentation set, which includes the following manuals:

Manual	Description
<a href="#">HSPICE User Guide: Simulation and Analysis</a>	Describes how to use HSPICE to simulate and analyze your circuit designs, and includes simulation applications. This is the main HSPICE user guide.
<a href="#">HSPICE User Guide: Signal Integrity</a>	Describes how to use HSPICE to maintain signal integrity in your chip design.
<a href="#">HSPICE User Guide: RF Analysis</a>	Describes how to use special set of analysis and design capabilities added to HSPICE to support RF and high-speed circuit design.
<a href="#">HSPICE Reference Manual: Elements and Device Models</a>	Describes standard models you can use when simulating your circuit designs in HSPICE, including passive devices, diodes, JFET and MESFET devices, and BJT devices.
<a href="#">HSPICE Reference Manual: MOSFET Models</a>	Describes available MOSFET models you can use when simulating your circuit designs in HSPICE.
<a href="#">HSPICE Integration to Cadence® Virtuoso® Analog Design Environment User Guide</a>	Describes use of the HSPICE simulator integration to the Cadence tool.
<a href="#">AMS Discovery Simulation Interface Guide for HSPICE</a>	Describes use of the Simulation Interface with other EDA tools for HSPICE.
<a href="#">AvanWaves User Guide</a>	Describes the AvanWaves tool, which you can use to display waveforms generated during HSPICE circuit design simulation.

---

---

## Searching Across the HSPICE Documentation Set

You can access the PDF format documentation from your install directory for the current release by entering `-docs` on the terminal command line when the HSPICE tool is open.

Synopsys includes an index with your HSPICE documentation that lets you search the entire HSPICE documentation set for a particular topic or keyword. In a single operation, you can instantly generate a list of hits that are hyper-linked to the occurrences of your search term. For information on how to perform searches across multiple PDF documents, see the HSPICE release notes.

**Note:** To use this feature, the HSPICE documentation files, the Index directory, and the `index.pdx` file must reside in the same directory. (This is the default installation for Synopsys documentation.) Also, Adobe Acrobat must be invoked as a standalone application rather than as a plug-in to your web browser.

You can also invoke HSPICE and RF documentation in a browser-based help system by entering `-help` on your terminal command line when the HSPICE tool is open. This provides access to all the HSPICE manuals with the exception of the *AvanWaves User Guide* which is available in PDF format only.

---

## Known Limitations and Resolved STARs

You can find information about known problems and limitations and resolved Synopsys Technical Action Requests (STARs) in the *HSPICE Release Notes* shipped with this release. For updates, go to SolvNet.

To access the *HSPICE Release Notes*:

1. Go to <https://solvnet.synopsys.com/ReleaseNotes>. (If prompted, enter your user name and password. If you do not have a Synopsys user name and password, follow the instructions to register with SolvNet.)
2. Select Download Center> HSPICE> version number> Release Notes.

---

## Conventions

The following conventions are used in Synopsys HSPICE documentation.

*Table 1* *Typographical conventions*

Convention	Description
<code>Courier</code>	Indicates command syntax.
<i>Italic</i>	Indicates a user-defined value, such as <i>object_name</i> .
<b>Bold</b>	Indicates user input — text you type verbatim — in syntax and examples. <b>Bold</b> indicates a GUI element.
[ ]	Denotes optional parameters, such as: <code>write_file [-f filename]</code>
...	Indicates that parameters can be repeated as many times as necessary: <code>pin1 pin2 ... pinN</code>
	Indicates a choice among alternatives, such as <code>low   medium   high</code>
+	Indicates a continuation of a command line.
/	Indicates levels of directory structure.
Edit > Copy	Indicates a path to a menu command, such as opening the Edit menu and choosing Copy.
Control-c	Indicates a keyboard combination, such as holding down the Control key and pressing c.

---

## Customer Support

Customer support is available through SolvNet online customer support and through contacting the Synopsys Technical Support Center.



---

## Accessing SolvNet

SolvNet includes an electronic knowledge base of technical articles and answers to frequently asked questions about Synopsys tools. SolvNet also gives you access to a wide range of Synopsys online services, which include downloading software, viewing Documentation on the Web, and entering a call to the Support Center.

To access SolvNet:

1. Go to the SolvNet Web page at <http://solvnet.synopsys.com>.
2. If prompted, enter your user name and password. (If you do not have a Synopsys user name and password, follow the instructions to register with SolvNet.)

If you need help using SolvNet, click Help on the SolvNet menu bar.

The link to any recorded training is

<https://solvnet.synopsys.com/trainingcenter/view.faces>

Access recent release update training by going to

[https://solvnet.synopsys.com/search/advanced\\_search.faces](https://solvnet.synopsys.com/search/advanced_search.faces)

---

## Contacting the Synopsys Technical Support Center

If you have problems, questions, or suggestions, you can contact the Synopsys Technical Support Center in the following ways:

- Open a call to your local support center from the Web by going to <http://solvnet.synopsys.com/EnterACall> (Synopsys user name and password required).
- Send an e-mail message to your local support center.
  - E-mail [support\\_center@synopsys.com](mailto:support_center@synopsys.com) from within North America.
  - Find other local support center e-mail addresses at [http://www.synopsys.com/support/support\\_ctr](http://www.synopsys.com/support/support_ctr).
- Telephone your local support center.
  - Call (800) 245-8005 from within the continental United States.
  - Call (650) 584-4200 from Canada.
  - Find other local support center telephone numbers at [http://www.synopsys.com/support/support\\_ctr](http://www.synopsys.com/support/support_ctr).

Customer Support

# HSPICE and HSPICE RF Application Commands

---

*Describes the commands you use to start HSPICE or HSPICE RF, including syntax, arguments, and examples.*

This chapter provides the syntax and arguments for the `hspice` or `hspicerf` application commands. You can enter these commands at the command-line prompt to start HSPICE or HSPICE RF from the unified HSPICE binary (`hspice`) on all primary platforms. (You can run RF features with one HSPICE license and one HSPICE RF license or with two HSPICE licenses.)

The following sections show you how to invoke:

- [hspice](#)
- [hspicerf](#)

---

## hspice

Invokes HSPICE or HSPICE RF.

### Syntax

```
hspice
[-i path/input_file | -i longpath_exceed256/input_file]
[-o path/output_file | -o longpath_exceed256/output_file]
[-o path [-n number] [-html path/html_file] [-d]
[-C path/input_file] [-CC path/input_file] [-I] [-K]
[-L command_file] [-S] [-case 0|1]
[-dp dp# [-dpconfig dpconfig_file] [-merge]
[-mp process_count] [-mt thread_count] [-hpp]
[-meas measure_file] [-top subcktname]
[-restore checkpoint_file.store.gz]
[-hdl file_name] [-hdlpath pathname]
[-vamodel name] [-vamodel name2...]
[-help] [-doc] [-h] [-v]
```

---

Argument	Description
<code>-i path/input_file</code>	<p>Input netlist file name for which an extension <code>*.ext</code> is optional. If you do not specify an input file name extension in the command, HSPICE searches</p> <ul style="list-style-type: none"> <li>▪ for a <code>*.sp#</code> file, or</li> <li>▪ for a <code>*.tr#</code>, <code>*.ac#</code>, or <code>*.sw#</code> file (PSF files are not supported).</li> </ul> <p>HSPICE uses the input file name as the root for the output files. To exceed 256 character use the <code>-i longpath_exceed256/ filename</code> command. HSPICE also checks for an initial conditions file (<code>.ic</code>) that has the input file root name. The following is an example of an input file name: <code>/usr/sim/work/rb_design.sp</code> In this file name:</p> <ul style="list-style-type: none"> <li>▪ <code>/usr/sim/work/</code> is the directory path to the design</li> <li>▪ <code>rb_design</code> is the design root name</li> <li>▪ <code>.sp</code> is the file name suffix</li> </ul>

## Chapter 1: HSPICE and HSPICE RF Application Commands

hspice

Argument	Description
<code>-o path/output_file</code>	<p>Name of the output file. Here, <code>output_file</code> is the root name of the output file. HSPICE appends the <code>.lis</code> extension to all output files. For example:</p> <ul style="list-style-type: none"><li>▪ For the output log file: <code>output_file.lis</code></li><li>▪ For transient waveform: <code>output_file.tr0</code></li><li>▪ For transient measurement: <code>output_file.mt0</code></li></ul> <p>Everything up to the last period is the root file name and everything after the last period is the file name extension.</p> <ul style="list-style-type: none"><li>▪ If you either do not use this option or you use it without specifying a file name, HSPICE uses the output root file name specified in the <code>-html</code> option. To turn off the html popup, use the <code>-o</code> following the input file name.</li><li>▪ If you include the <code>.lis</code> extension in the file name that you enter using this option, then HSPICE does not append another <code>.lis</code> extension to the root file name of the output file.</li><li>▪ If you do not specify an output file name, HSPICE directs output to stdout.</li></ul> <p>For the <code>.meas</code> option, some case results differ from the measure result HSPICE produces. To exceed 256 character use the <code>-o longpath_exceed256/filename</code> command.</p>
<code>-n number</code>	<p>Starting number for numbering output data file revisions (<code>output_file.tr#</code>, <code>output_file.ac#</code>, <code>output_file.sw#</code>, where # is between 0 and 9999.).</p>
<code>-html path/html_file</code>	<p>HTML output file.</p> <ul style="list-style-type: none"><li>▪ If a path is unspecified, HSPICE saves the HTML output file in the same directory that you specified in the <code>-o</code> option.</li><li>▪ If you do not specify an <code>-o</code> option, HSPICE saves the HTML output in the working directory.</li><li>▪ If you do not specify an output file name in either the <code>-o</code> or <code>-html</code> option, then HSPICE uses the input root file name as the output file root file name.</li></ul> <p>If you add <code>.option itrprt = 1</code> to your netlist to print output variables at their internal time points, and you use the <code>-html</code> option when invoking HSPICE, then HSPICE prints the values to a separate file (<code>*.printtr0</code>).</p>
<code>-d</code>	<p>(UNIX) Displays the content of <code>.st0</code> files on screen while running HSPICE. For example, to show the status during simulation.</p>

## Chapter 1: HSPICE and HSPICE RF Application Commands

hspice

Argument	Description
<code>-C path/input_file</code>	<p>Client/Server (C/S) mode.</p> <ul style="list-style-type: none"><li>▪ Entering hspice -C checks out an HSPICE license and starts client/server mode.</li><li>▪ Entering hspice -C <i>path/input_file</i> simulates your netlist.</li><li>▪ Entering hspice -C -K releases the HSPICE license and exits.</li></ul> <p>For additional information, see <a href="#">Using HSPICE in Client-Server Mode</a> in the <i>HSPICE User Guide: Simulation and Analysis</i>.</p>
<code>-CC path/input_file</code>	<p>Advanced Client/Server mode.</p> <ul style="list-style-type: none"><li>▪ Entering hspice -CC checks out an HSPICE license and starts the advanced client/server mode.</li><li>▪ Entering hspice -CC <i>path/input_file</i> simulates your netlist.</li><li>▪ Adding -mp [<i>process_count</i>] enables multiprocessing in the file contains .Alter, Tran sweeps, or Monte Carlo trials.</li><li>▪ Entering hspice -CC -share common.sp -o output redirects share file to avoid issues with *.lis file for the shared model file while running the multiple servers on a farm or multi-CPU machine.</li><li>▪ Entering hspice -CC -K releases the HSPICE license and exits.</li></ul> <p>For additional information, see <a href="#">Launching the Advanced Client-Server Mode (-CC)</a> in the <i>HSPICE User Guide: Simulation and Analysis</i></p>
<code>-I</code>	<p>Interactive mode.</p> <ul style="list-style-type: none"><li>▪ Entering hspice -I invokes interactive mode.</li><li>▪ Entering help at the HSPICE prompt lists supported commands.</li><li>▪ Entering hspice -I -L <i>file_name</i> runs a command file.</li><li>▪ Entering quit at the HSPICE exits interactive mode.</li></ul> <p>For additional information, see “<a href="#">Using Interactive Mode</a>” in the <i>HSPICE User Guide: Simulation and Analysis</i>.</p>
<code>-K</code>	<p>Used with -C option to terminate client/server mode and exit.</p>
<code>-L file_name</code>	<p>Used with -I option to run commands contained in a command file.</p>
<code>-S</code>	<p>Performs as a server. Accepts data from SPEED2000, simulates the circuit, and returns results to SPEED2000.</p> <ul style="list-style-type: none"><li>▪ On UNIX and Linux, HSPICE waits for successive simulations after invocation.</li><li>▪ On Windows you must re-invoke for each successive simulation.</li></ul>

## Chapter 1: HSPICE and HSPICE RF Application Commands

hspice

Argument	Description
<code>-case 0 1</code>	<ul style="list-style-type: none"><li>▪ 0: (default) case sensitivity disabled</li><li>▪ 1: case sensitivity enabled</li></ul> Enables case sensitivity only for the following items (HSPICE commands and control options continue to be case-insensitive): <ul style="list-style-type: none"><li>▪ Parameter Names</li><li>▪ Node Names</li><li>▪ Instance Names</li><li>▪ Model Names</li><li>▪ Subcircuit Names</li><li>▪ Data Names</li><li>▪ Measure Names</li><li>▪ File Names and Paths (case sensitive by default)</li><li>▪ Library Entry Names</li></ul>
<code>-dp dp#</code> <code>[-dpconfig</code> <code>dpconfig_file]</code> <code>[-merge]</code>	<ul style="list-style-type: none"><li>▪ Invokes distributed processing and specifies number of processes.</li><li>▪ Specifies the configuration file for DP. HSPICE runs DP on a single local machine if this option is not specified.</li><li>▪ Merges the output files from HSPICE. DP only merges the output files if this option is specified.</li></ul> For details see <a href="#">Running Distributed Processing (DP) on a Network Grid</a> in the <i>HSPICE User Guide: Simulation and Analysis</i> .
<code>-hpp</code>	Enables HSPICE Performance Precision. The multicore algorithm can be applied without multithreading ( <code>-mt</code> ), but for best performance, use <code>-hpp -mt N</code> , together, where <i>N</i> is number of threads. For details, see the <i>HSPICE User Guide: Simulation and Analysis</i> , Chapter 3, Setup and Simulation, section <a href="#">HSPICE Precision Parallel (-hpp)</a> .
<code>-mp [process_count]</code>	Activates multiprocessing while running ALTER cases, transient sweeps, and Monte Carlo analyses on one machine with multiple processors/cores. If you specify the number of CPUs you can limit the number of CPUs to avoid overtaxing performance scalability. If a CPU number is not specified, HSPICE auto-determines the child processes by the number of available CPUs. For details see <a href="#">Multiprocessing (MP) DC Monte Carlo</a> in the <i>HSPICE User Guide: Simulation and Analysis</i> .

## Chapter 1: HSPICE and HSPICE RF Application Commands

hspice

Argument	Description
<code>-mt <i>thread_count</i></code>	<p>Invokes multithreading and specifies the number of processors for a multithreaded simulation. If <code>thread_count</code> is not entered, HSPICE issues an error.</p> <p>For additional information, see <a href="#">Running Multithread/Multiprocess HSPICE Simulations</a> in the <i>HSPICE User Guide: Simulation and Analysis</i>. See also <a href="#">.OPTION MTTHRESH</a> in this manual.</p>
<code>-hpp</code>	<p>Enables HSPICE Performance Precision. The multicore algorithm can be applied without multithreading (<code>-mt</code>), but for best performance, use <code>-hpp -mt N</code>, together, where <i>N</i> is number of threads. For details, see the <i>HSPICE User Guide: Simulation and Analysis</i>, Chapter 3, Setup and Simulation, section <a href="#">HSPICE Precision Parallel (-hpp)</a>.</p>
<code>-meas <i>measure_file</i></code>	<p>Re-invokes the measure file to calculate new measurements from a previous simulation. The format of <code>measure_file</code> is similar to the HSPICE netlist format. The first line is a comment line and the last line is an <code>.END</code> command. The following netlist commands are supported.</p> <ul style="list-style-type: none"><li>▪ <code>.MEASURE</code></li><li>▪ <code>.PARAM</code></li><li>▪ <code>.TEMP</code></li><li>▪ <code>.OPTION</code></li><li>▪ <code>.DATA</code></li><li>▪ <code>.ENDDATA</code></li><li>▪ <code>.FFT</code></li><li>▪ <code>.MEASURE FFT</code></li><li>▪ <code>.END</code></li></ul> <p>Note: The <code>.DATA</code> command in the measure file must be consistent with the <code>.DATA</code> command in the wavefile.</p> <p>The following types of <code>.OPTION</code> commands are supported:</p> <ul style="list-style-type: none"><li>▪ <code>MEASFAIL</code></li><li>▪ <code>NUMDGT</code></li><li>▪ <code>INGOLD</code></li><li>▪ <code>MEASDGT</code></li><li>▪ <code>EM_RECOVERY</code></li></ul> <p>Warnings are issued if other options or commands are used. Wave files formatted as PSF, CSDF, and WDF are not supported.</p> <p>Syntax to perform spectrum analysis measurements from previous simulation results:</p> <pre>hspice -i *.tr0 -meas <i>measure_file</i></pre>



Argument	Description
<code>-top <i>subcktname</i></code>	<p>Top level subcircuit name. Effectively eliminates the need for defining “.subckt <i>subcktname</i>” and corresponding “.ends” statements. Users do not need to instantiate top-level SUBCKT using “X” syntax of HSPICE.</p>
<code>-restore checkpoint_file. store.gz</code>	<p>The <code>checkpoint_file</code> specifies from which simulation the checkpoint data is to be restored.</p> <p>The <code>restore</code> operation should be submitted on a machine that has the same kernel version as the machine used to store, otherwise, a failure may occur.</p> <p>Note: Do not use <code>-o <i>outputfile</i></code> to specify output files; the output files will be the same as were used in the previous simulation.</p> <p>Any output files generated by the previous simulation should not be removed. After the restore simulation is done, the output files will be updated. For example:  <code>hspice -restore test_1000.store.gz.</code></p> <p>The simulation starts from the time point that data was stored at in the previously interrupted simulation. The data files are named <code>test_1000.store.gz</code>, and <code>test_1000.tar</code>.  See <a href="#">Storing and Restoring Checkpoint Files (HSPICE)</a> for full details.</p>
<code>-hdl <i>file_name</i></code>	<p>Verilog-A module. The Verilog-A file is assumed to have a *.va extension when only a prefix is provided. One <code>-hdl</code> option can include one Verilog-A file, use multiple <code>-hdl</code> options if multiple Verilog-A files are needed. This example loads the <code>amp.va</code> Verilog-A source file:  <code>hspice amp.sp -hdl amp.va</code></p> <p>When a module to be loaded has the same name as a previously-loaded module or the names differ in case only, the latter one is ignored and the simulator issues a warning message.</p> <p>If a Verilog-A module file is not found or the Compiled Model Library file has an incompatible version, the simulation exits and an error message is issued.</p>

## Chapter 1: HSPICE and HSPICE RF Application Commands

hspice

---

Argument	Description
<code>-hdlpath <i>pathname</i></code>	<p>Search path for a Verilog-A file if HSPICE cannot find it in the current working directory. The search order for Verilog-A files is:</p> <ol style="list-style-type: none"><li>1. Current working directory</li><li>2. Path defined by command-line argument <code>-hdlpath</code></li><li>3. Path defined by environment variable <code>HSP_HDL_PATH</code></li></ol> <p>The path defined by either <code>-hdlpath</code> or <code>HSP_HDL_PATH</code> can consist a set of directory names. The path separator must follow HSPICE conventions or platform conventions (“;” on UNIX). Path entries that do not exist are ignored and no error/warning messages are issued.</p> <p>This example first searches the current working directory and when a *.va file is not found, the relative location <code>./my_modules</code> directory is searched: <code>hspice amp.sp -hdlpath ./my_modules</code></p>
<code>-vamodel <i>name</i></code> <code>-vamodel <i>name2...</i></code>	<p>Cell names for Verilog-A definitions. <i>name</i> is the cell name that uses a Verilog-A definition rather than a subcircuit definition when both exist. Each <code>-vamodel</code> option can take no more than one name. Repeat this option if multiple Verilog-A modules are defined. If no name is supplied after <code>-vamodel</code>, then the Verilog-A definition will be used whenever it is available.</p>
<code>-help</code>	<p>Searchable browser-based help system for HSPICE/HSPICFE RF. An html browser must be installed on your machine to access this help system.</p>
<code>-doc</code>	<p>PDF documentation set user manuals for HSPICE and RF flow. Requires Adobe Acrobat Reader to be installed on your system. You can do full text searches of the documentation set. See the Release Notes for instructions.</p>
<code>-h</code>	<p>Displays a help message and exits.</p>
<code>-v</code>	<p>Outputs version information and exits.</p>

---

---

## Examples of Starting HSPICE

The following are more examples of commands to start running HSPICE.

- `hspice demo.sp -n 7 > demo.out`

This command redirects output to a file instead of stdout. `demo.sp` is the input netlist file. The `.sp` extension is optional. The `-n 7` starts the output data file revision numbers at 7; for example: `demo.tr7`, `demo.ac7`, `demo.sw7`, and so forth. The `>` redirects the program output listing to file `demo.out`.
- `hspice -i demo.sp`

`demo` is the root input file name. Without the `-o` argument and without redirection, HSPICE does not generate an output listing file.
- `hspice -i demo.sp -o demo`

`demo` is the output file root name (designated with the `-o` option). Output files are named `demo.lis`, `demo.tr0`, `demo.st0`, and `demo.ic0`.
- `hspice -i rmdir/demo.sp`

`demo` is the input root file name. HSPICE writes the `demo.lis`, `demo.tr0`, and `demo.st0` output files into the directory where you executed the HSPICE command. It also writes the `demo.ic0` output file into the same directory as the input source—that is, `rmdir`.
- `hspice -i a.b.sp`

`a.b` is the root name. The output files are `./a.b.lis`, `./a.b.tr0`, `./a.b.st0`, and `./a.b.ic0`.
- `hspice -i a.b -o d.e`

`a.b` is the root name for the input file. `d.e` is the root output file name, except for the `.ic` file to which HSPICE assigns the `a.b` input file root name. The output files are `d.e.lis`, `d.e.tr0`, `d.e.st0`, and `a.b.ic0`.
- `hspice -i a.b.sp -o outdir/d.e`

HSPICE writes the output files as: `outdir/d.e.lis`, `outdir/d.e.tr0`, `outdir/d.e.st0`, and `outdir/d.e.ic0`.
- `hspice -i indir/a.b.sp -o outdir/d.e.lis`

`a.b` is the root for the `.ic` file. HSPICE writes the `.ic0` file into a file named `indir/a.b.ic0`. `d.e` is the root for the output files.
- `hspice test.sp -o test.lis -html test.html`

This command creates output file in both .lis and .html format after simulating the test.sp input netlist.

- `hspice test.sp -html test.html`

This command creates only a .html output file after simulating the test.sp input netlist.

- `hspice test.sp -o test.lis`

This command creates only a .lis output file after simulating the test.sp input netlist.

- `hspice -i test.sp -o -html outdir/a.html`

This command creates output files in both .lis and .html format. Both files are in the outdir directory and their root file name is a.

- `hspice -i test.sp -o out1/a.lis -html out2/b.html`

This command creates output files in both .lis and .html format. The .lis file is in the out1 directory and its root file name is a. The .html file is in the out2 directory and its root file name is b.

- `hspice -i test.sp -o test -x`

This command launches a full parasitic back-annotation for the file named test.sp.

---

## Using HSPICE for Calculating New Measurements

When you want to calculate new measurements from previous simulation results produced by HSPICE you can use the following mode to rerun HSPICE without having to do another simulation:

```
hspice -meas measurefile -i wavefile -o outputfile
```

See the following table for arguments and descriptions.

Argument	Description
-meas <i>measurefile</i>	<p>This format is similar to the HSPICE netlist format. The first line is a comment line and the last line is an .END command. Seven commands are supported:</p> <ul style="list-style-type: none"> <li>▪ .MEASURE</li> <li>▪ .PARAM</li> <li>▪ .TEMP</li> <li>▪ .OPTION</li> <li>▪ .DATA</li> <li>▪ .ENDDATA</li> <li>▪ .END</li> </ul> <p>Note: The .DATA command in the measure file must be consistent with the .DATA command in the wavefile.</p> <p>The .OPTION command support four types:</p> <ul style="list-style-type: none"> <li>▪ MEASFAIL</li> <li>▪ NUMDGT</li> <li>▪ INGOLD</li> <li>▪ MEASDGT</li> </ul> <p>Warnings are issued if other options or commands are used. Wave files formatted as PSF and CSDF are not supported.</p>
-i <i>wavefile</i>	<p>*.tr#, *.ac#, and *.sw# files produced by HSPICE. Waveform files formatted as PSF are not supported.</p> <p>If a plot fails to open, it is due to one of the following reasons:</p> <p>Waveform file format is not supported.</p> <p>File format is not understood</p> <p>File is not found</p> <p>File larger than max size of x</p> <p>Note: “x” depends on any file size limitation of your application. For example, a 2GB file size limitation exists on 32-bit HSPICE Linux, SuSe versions when reading the waveforms. Solaris has no such limitation</p>
-o <i>outputfile</i>	<p>Same output files as HSPICE. Some case results are different from the measure result HSPICE produces due to an accuracy problem.</p>
-h	<p>Displays a help message and exits.</p>

Argument	Description
-v	Outputs version information and exits.

## hspicerf

Invokes HSPICE RF in standalone mode. HSPICE RF can be launched with either the integrated executable (**hspice**) or in standalone mode (**hspicerf**).

### Syntax

```
hspicerf [-a] input_file [output_file] [-n] [-h] [-v]
```

Argument	Description
-a	Generates output to stdout in ASCII format. For example, <pre>% hspicerf -a ckt.in</pre> You can redirect the ASCII output to another file. For example, <pre>% hspicerf -a ckt.in &gt; xt</pre> Output from a .PRINT command goes to an ASCII file with a .print# or .printac# file extension.
<i>input_file</i>	Name of the input netlist.
<i>output_file</i>	Name of the output listing file. If specified, the simulation output is written to this file and given a .lis file extension. For example, <pre>% hspicerf ckt.in xt</pre> automatically sets -a and generates output to xt.lis.
-n <i>number</i>	Starting number for numbering output data file revisions (output_file.tr#, output_file.ac#, output_file.sw#, where # is between 0 and 9999.).
-h	Returns a help message.
-v	Returns version information.

## HSPICE and HSPICE RF Netlist Commands

---

*Describes the commands you can use in HSPICE/HSPICE RF netlists.*

This chapter provides a list of the HSPICE and HSPICE RF netlist commands, arranged by task, followed by detailed descriptions of the individual commands.

The netlist commands described in this chapter fall into the following categories:

- [Alter Block](#)
- [Analysis](#)
- [Conditional Block](#)
- [Digital Vector](#)
- [Encryption](#)
- [Field Solver](#)
- [Files](#)
- [Input/Output Buffer Information Specification \(IBIS\)](#)
- [Library Management](#)
- [Model and Variation Definition](#)
- [Node Naming](#)
- [Output Porting](#)
- [Setup](#)
- [Simulation Runs](#)
- [Subcircuits](#)
- [Verilog-A](#)

### Use of Example Syntax

To copy and paste proven syntax use the demonstration files shipped with your installation of HSPICE (see [Listing of Demonstration Input Files](#)). Attempting to copy and paste from the book or help documentation may present unexpected results, as text used in formatting may include hidden characters, white space, etc. for visual clarity.

---

## HSPICE and RF Commands Overview

---

### Analysis

Use these commands in your netlist to start different types of HSPICE analysis to save the simulation results into a file and to load the results of a previous simulation into a new simulation.

#### HSPICE

---

<a href="#">.AC</a>	<a href="#">.DCSENS</a>	<a href="#">.LIN</a>	<a href="#">.PAT</a>	<a href="#">.STATEYE</a>
<a href="#">.ACMATCH</a>	<a href="#">.DISTO</a>	<a href="#">.LSTB</a>	<a href="#">.PZ</a>	<a href="#">.TEMP (or) .TEMPERATURE</a>
<a href="#">.DC</a>	<a href="#">.FFT</a>	<a href="#">.NOISE</a>	<a href="#">.SAMPLE</a>	<a href="#">.TF</a>
<a href="#">.DCMATCH</a>	<a href="#">.FOUR</a>	<a href="#">.OP</a>	<a href="#">.SENS</a>	<a href="#">.TRAN</a>

---

### HSPICE RF Analysis

Use these commands in your RF netlist to run different types of HSPICE RF analyses, save the simulation results into a file, and to load the results of a previous simulation into a new simulation.

---

<a href="#">.AC</a>	<a href="#">.ENVFFT</a>	<a href="#">.LIN</a>	<a href="#">.SN</a>
<a href="#">.ACPHASENOISE</a>	<a href="#">.ENVOSC</a>	<a href="#">.LPRINT</a>	<a href="#">.SNAC</a>
<a href="#">.CHECK EDGE</a>	<a href="#">.FFT</a>	<a href="#">.MEASURE PHASENOISE</a>	<a href="#">.SNFT</a>

---



---

<code>.CHECK FALL</code>	<code>.FOUR</code>	<code>.MEASURE PTDNOISE</code>	<code>.SNNOISE</code>
<code>.CHECK GLOBAL_LEVEL</code>	<code>.HB</code>	<code>.NOISE</code>	<code>.SNOSC</code>
<code>.CHECK HOLD</code>	<code>.HBAC</code>	<code>.OP</code>	<code>.SNXF</code>
<code>.CHECK IRDROP</code>	<code>.HBLIN</code>	<code>.PHASENOISE</code>	<code>.SURGE</code>
<code>.CHECK RISE</code>	<code>.HBLSP</code>	<code>.POWER</code>	<code>.SWEEPBLOCK</code>
<code>.CHECK SETUP</code>	<code>.HBNOISE</code>	<code>.POWERDC</code>	<code>.TEMP (or) .TEMPERATURE</code>
<code>.CHECK SLEW</code>	<code>.HBOSC</code>	<code>.PTDNOISE</code>	<code>.TF</code>
<code>.DC</code>	<code>.HBXF</code>	<code>..PZ</code>	<code>.TRAN</code>
<code>.ENV</code>			.

---



---

## Alter Block

Use these commands in your netlist to run alternative simulations of your netlist by using different data.

---

<code>.ALTER</code>	<code>.DEL LIB</code>	<code>.TEMP (or) .TEMPERATURE</code>
---------------------	-----------------------	--------------------------------------

---



---

## Conditional Block

Use these commands in your HSPICE netlist to setup a conditional block. HSPICE does not execute the commands in the conditional block unless the specified conditions are true.

---

<code>.ELSE</code>	<code>.ELSEIF</code>	<code>.ENDIF</code>	<code>.IF</code>
--------------------	----------------------	---------------------	------------------

---

---

## Digital Vector

Use these commands in your digital vector (VEC) file.

---

ENABLE	PERIOD	TRISE	VIL	VTH
IDELAY	RADIX	TRIZ	VNAME	
IO	SLOPE	TSKIP	VOH	
ODELAY	TDELAY	TUNIT	VOL	
OUT or OUTZ	TFALL	VIH	VREF	

---

---

## Encryption

Use these commands in your netlist to mark the start and end of a traditionally (Freelib) encrypted section of a netlist.

`.PROTECT` or `.PROT`      `.UNPROTECT` or `.UNPROT`

---

## Field Solver

Use these commands in your netlist to define a field solver.

`.FSOPTIONS`      `.LAYERSTACK`      `.MATERIAL`      `.SHAPE`

---

## Files

Use this command in your netlist to call other files that are not part of the netlist.

`.VEC`



---

## Output Porting

Use these commands in your netlist to specify the output of a simulation to a printer or graph. You can also define the parameters to measure and to report in the simulation output.

`.BIASCHK`      `.MEASURE`      `.PROBE`      `.DOUT`      `.PRINT`      `.STIM`  
(or) `.MEAS`

---

## Setup

Use these commands in your netlist to set up your netlist for simulation.

`.DATA`      `.ENDDATA`      `.IC`      `.NODESET`      `.PARAM`      `.TITLE`  
(or) `.PARAMETER`  
(or) `.PARAMETERS`

`.DCVOLT`      `.GLOBAL`      `.LOAD`      `.OPTION`      `.SAVE`  
(or) `.OPTIONS`

---

## Simulation Runs

Use these commands in your netlist to mark the start and end of individual simulation runs and conditions that apply throughout an individual simulation run.

`.END`      `.TEMP`      `.TITLE`  
(or) `.TEMPERATURE`

---

## Subcircuits

Use these commands in your netlist to define subcircuits and to add instances of subcircuits to your netlist.

`.ENDS`      `.INCLUDE` (or) `.INC`      `.MODEL`  
(or) `.INCL`

[.EOM](#)[.MACRO](#)[.SUBCKT](#)

---

## Verilog-A

Use the following command in your netlist to declare the Verilog-A source name and path within the netlist.

[.HDL](#)

---

## Alphabetical Listing of Commands

The following is the alphabetical list of links to the HSPICE/RF command set. For simulation controls see [Chapter 3, HSPICE and RF Netlist Simulation Control Options](#).

- [.AC](#)
- [.ACMATCH](#)
- [.ACPHASENOISE](#)
- [.ALIAS](#)
- [.ALTER](#)
- [.APPENDMODEL](#)
- [.BA\\_ACHECK](#)
- [.BIASCHK](#)
- [.CFL\\_PROTOTYPE](#)
- [.CHECK EDGE](#)
- [.CHECK FALL](#)
- [.CHECK GLOBAL\\_LEVEL](#)
- [.CHECK HOLD](#)
- [.CHECK IRDROP](#)
- [.CHECK RISE](#)
- [.CHECK SETUP](#)

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

Verilog-A

- .CHECK SLEW
- .CONNECT
- .DATA
- .DC
- .DCMATCH
- .DCSENS
- .DCVOLT
- .DEL LIB
- .DESIGN\_EXPLORATION
- .DISTO
- .DOUT
- .EBD
- .ELSE
- .ELSEIF
- .END
- .ENDDATA
- .ENDIF
- .ENDL (or) .ENDL TT
- .ENDS
- .ENV
- .ENVFFT
- .ENVOSC
- .EOM
- .FFT
- .FLAT
- .FOUR
- .FSOPTIONS
- .GLOBAL
- .HB

- .HBAC
- .HBLIN
- .HBLSP
- .HBNOISE
- .HBOSC
- .HBXF
- .HDL
- .IBIS
- .IC
- .ICM
- .IF
- .INCLUDE (or) .INC (or) .INCL
- .IVTH
- .JITTER
- .LAYERSTACK
- .LIB
- .LIN
- .LOAD
- .LPRINT
- .LSTB
- .MACRO
- .MALIAS
- .MATERIAL
- .MEASURE (or) .MEAS
- .MEASURE (Rise, Fall, Delay, and Power Measurements)
- .MEASURE (FIND and WHEN)
- .MEASURE (Continuous Results)
- .MEASURE (Equation Evaluation/Arithmetic Expression)
- .MEASURE (AVG, EM\_AVG, INTEG, MIN, MAX, PP, and RMS)

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

Verilog-A

- .MEASURE (Integral Function)
- .MEASURE (Derivative Function)
- .MEASURE (Error Function)
- .MEASURE PHASENOISE
- .MEASURE PTDNOISE
- .MEASURE (Pushout Bisection)
- .MEASURE (ACMATCH)
- .MEASURE (DCMATCH)
- .MEASURE FFT
- .MEASURE LSTB
- .MODEL
- .MODEL\_INFO
- .MOSRA
- .MOSRAPRINT
- .MOSRA\_SUBCKT\_PIN\_VOLT
- .NODESET
- .NOISE
- .OP
- .OPTION (or) .OPTIONS
- .PARAM (or) .PARAMETER (or) .PARAMETERS
- .PAT
- .PHASENOISE
- .PKG
- .POWER
- .POWERDC
- .PRINT
- .PROBE
- .PROTECT or .PROT
- .PTDNOISE



- .PZ
- .SAMPLE
- .SAVE
- .SENS
- .SHAPE
- .SHAPE (Defining Rectangles)
- .SHAPE (Defining Circles)
- .SHAPE (Defining Polygons)
- .SHAPE (Defining Strip Polygons)
- .SHAPE (Defining Trapezoids)
- .SN
- .SNAC
- .SNFT
- .SNNOISE
- .SNOSC
- .SNXF
- .STATEYE
- .STIM
- .STORE
- .SUBCKT
- .SURGE
- .SWEEPBLOCK
- .TEMP (or) .TEMPERATURE
- .TF
- .TITLE
- .TRAN
- .TRANNOISE
- .UNPROTECT or .UNPROT

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

Verilog-A

- `.VARIATION`
- `.VEC`

---

## .AC

Performs several types of AC analyses.

### Syntax

#### Single or Double Sweep

```
.AC type np fstart fstop
.AC type np fstart fstop [SWEEP var [START=]start
+ [STOP=]stop [STEP=]incr]
.AC type np fstart fstop [SWEEP var type np start stop]
.AC type np fstart fstop
+ [SWEEP var START="param_expr1"
+ STOP="param_expr2" STEP="param_expr3"]
.AC type np fstart fstop [SWEEP var start_expr
+ stop_expr step_expr]
```

#### Sweep Using Parameters

```
.AC type np fstart fstop [SWEEP DATA=datanm]
.AC DATA=datanm
.AC DATA=datanm [SWEEP var [START=]start [STOP=]stop
+ [STEP=]incr]
.AC DATA=datanm [SWEEP var type np start stop]
.AC DATA=datanm [SWEEP var START="param_expr"
+ STOP="param_expr2" STEP="param_expr3"]
.AC DATA=datanm [SWEEP var start_expr stop_expr
+ step_expr]
```

#### Optimization

```
.AC DATA=datanm OPTIMIZE=opt_par_fun
+ RESULTS=measnames MODEL=optmod
```

#### Monte Carlo

```
.AC type np fstart fstop [SWEEP MONTE=MCcommand]
```

---

Argument	Description
DATA=datanm	Data name, referenced in the .AC command.
incr	Increment value of the voltage, current, element, or model parameter. If you use type variation, specify the np (number of points) instead of incr.
fstart	Starting frequency. If you use POI (list of points) type variation, use a list of frequency values, not fstart fstop.

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

.AC

---

Argument	Description
fstop	Final frequency.
MONTE= MCcommand	Where MCcommand can be any of the following: <ul style="list-style-type: none"><li>▪ <i>val</i> Specifies the number of random samples to produce.</li><li>▪ <i>val firstrun=num</i> Specifies the sample number on which the simulation starts.</li><li>▪ <i>list num</i> Specifies the sample number to execute.</li><li>▪ <i>list(num1:num2 num3 num4:num5)</i> Samples from <i>num1</i> to <i>num2</i>, sample <i>num3</i>, and samples from <i>num4</i> to <i>num5</i> are executed (parentheses are optional).</li></ul>
np	Number of points or points per decade or octave, depending on which keyword precedes it.
start	Starting voltage or current or any parameter value for an element or model.
stop	Final voltage or current or any parameter value for an element or a model.
SWEEP	Second sweep.
TEMP	Temperature sweep
type	Any of the following keywords: <ul style="list-style-type: none"><li>▪ DEC – decade variation.</li><li>▪ OCT – octave variation.</li><li>▪ LIN – linear variation.</li><li>▪ POI – list of points.</li></ul>
var	Name of an independent voltage or current source, element or model parameter or the TEMP (temperature sweep) keyword. HSPICE supports source value sweep, referring to the source name (SPICE style). If you select a parameter sweep, a .DATA command and a temperature sweep, then you must choose a parameter name for the source value. You must also later refer to it in the .AC command. The parameter name cannot start with V or I.
firstrun	The <i>val</i> value specifies the number of Monte Carlo iterations to perform. The <i>firstrun</i> value specifies the desired number of iterations. HSPICE runs from <i>num1</i> to <i>num1+val-1</i> .
list	Iterations at which HSPICE performs a Monte Carlo analysis. You can write more than one number after a list. The colon represents “from ... to ...”. Specifying only one number causes to HSPICE run at only the specified point.

---

## Description

The `.AC` command is usable in several different formats, depending on the application as shown in the examples. You can also use the `.AC` command to perform data-driven analysis in HSPICE.

If the input file includes an `.AC` command, HSPICE runs AC analysis for the circuit over a selected frequency range for each parameter in the second sweep.

For AC analysis, the data file must include at least one independent AC source element command (for example, `VI INPUT GND AC 1V`). HSPICE checks for this condition and reports a fatal error if you did not specify such AC sources.

## Examples

```
.AC DEC 10 1K 100MEG
```

This example performs a frequency sweep by 10 points per decade from 1kHz to 100MHz.

## See Also

[.DC](#)

[.DISTO](#)

[.LSTB](#)

[.NOISE](#)

[.TRAN](#)

[Using the .AC Statement](#)

[BJT and Diode Examples](#) for the paths to the demo files *mextram\_ac.sp* and *vbic99\_ac.sp*, which use the `.AC` command.

[Device Optimization Examples](#) for paths to the demo netlists *bjtopt.sp* and *bjtopt2.sp* which use `.AC sweep` keywords.

[MOSFET Device Examples](#) for paths to the demo netlists *calcap.sp* and *cascode.sp* for use of the `.AC` command.

[Applications of General Interest Examples](#) for the paths to the demo files *alm124.sp* and *quickAC.sp*, for `.AC` command usage.

[Transmission \(W-element\) Line Examples](#) for the paths to the demo files *ex1.sp*, *ex2.sp*, *ex3.sp*, *rlgc.sp*, and *umodel.sp* for `.AC` command usage.

---

## .ACMATCH

Calculates the effects of variations in device characteristics and parasitic capacitance sensitivities on a circuit's AC response.

### Syntax

```
.ACMATCH OUTVAR [THRESHOLD=T] [FILE=string] [INTERVAL=Int]
+ [Virtual_Sensitivity=Yes|No] [Sens_threshold=x]
+ [Sens_node=(nodei_name,nodej_name), ...,
+ (nodem_name,noden_name)]
```

---

Argument	Description
OutVar	OutputVariable can be one or several output voltages, difference voltages, or branch current through an independent voltage source. The voltage or current specifier is followed by an identifier of the AC quantity of interest: M: magnitude P: phase R: real part I: imaginary part
Threshold	Only devices with variation contributions above Threshold are reported in the table. Results for all devices are displayed if Threshold=0 is set. The maximum value for Threshold is 1.0, but at least 10 devices (or all) are displayed. Default is 0.01.
File	Valid file name for the output tables. Default is <i>basename.am#</i> , where # is the regular HSPICE sequence number.
Interval	This option applies to the frequency sweep definition in the .AC command. A table is printed at the first sweep point, then for each subsequent increment of SweepValue, and at the final sweep point.
Virtual_Sensitivity	Invokes ACmatch computation and output of virtual sensitivity; sensitivity table is printed even if variation block does not exist in netlist. Default: Yes
Sens_Threshold= <i>x</i>	Only nodes with sensitivity above <i>x</i> are reported. At least 10 sensitivities (or all) are displayed. This avoids generation of null output if you specify too large a value for <i>x</i> . Default: 1e-6
Sens_Node	Output all sensitivities associated with the requested nodes. The node name should appear in pairs. (See examples below.)

---

## Description

Use to calculate the effects of variations in device characteristics on a circuit's AC response. ACMatch allows for calculation of parasitic capacitor sensitivities whose nominal values are “zero” in the original design. Such analysis is useful for high precision (differential) analog circuits and switched capacitor filters. If more than one ACMatch analysis is specified per simulation, only the last command is executed. dB syntax is supported in .ACMatch for Vdb and Idb, for local, global, and element variation.

**Note:** ACMatch does not support Spatial Variations.

## Examples

```
.ACMATCH VM(out) VP(out) IM(x1.r1) IP(x1.r1) IM(c1) IP(c1)  
.AC dec 10 1k 10Meg interval=10
```

When using the virtual capacitance sensitivity option *Sens\_Node* multiple name pairs are supported with one comma between node names, but commas are optional between node name pairs. Either of the following specifications is valid in HSPICE:

```
.ACmatch v(out) virtual_sens=yes  
+ sens_node= (out, xi82.net044),  
+ (0,out), (xi82.net044,xi82.net031) sens_threshold=1e-6
```

OR

```
.ACmatch v(out) virtual_sens=yes  
+ sens_node= (out, xi82.net044)  
+ (0,out) (xi82.net044,xi82.net031) sens_threshold=1e-6
```

## See Also

- [.AC](#)
- [.MEASURE \(or\) .MEAS](#)
- [.MEASURE \(ACMATCH\)](#)
- [.OPTION POST](#)
- [ACMatch Analysis](#)

## **.ACPHASENOISE**

Helps you interpret signal and noise quantities as phase variables for accumulated jitter for closed-loop PLL analysis.

### **Syntax**

```
.ACPHASENOISE output input [interval] carrier=freq  
+ [listfreq=(frequencies|none|all)]  
+ [listcount=val] [listfloor=val]  
+ [listsources=(1|0)]
```

### **Description**

The `.ACPHASENOISE` command aids in the ability to compute “Accumulated Jitter” or “Timing Jitter” for the closed loop PLL. The accumulated jitter response is essentially an integral transformation of the closed-loop PLL response. The `.ACPHASENOISE` analysis outputs raw data to `*.pn0` and `*.printpn0` files. The `PHNOISE` data is given in units of dBc/Hz, i.e., dB relative to the carrier, per Hz, across the output nodes specified by the `.ACPHASENOISE` command. The data plot is a function of offset frequency. If the “JITTER” keyword is present, `.ACPHASENOISE` also outputs the accumulated TIE jitter data to `*.jt0` and `*.printjt0` data files. These data are plotted as a function of time in units of seconds. The Timing Jitter data itself has units of seconds. The timing jitter calculations make use of the parameters given in the `.ACPHASENOISE` syntax, such as “freq” and “interval”.

For details, see [Small-Signal Phase-Domain Noise Analysis \(.ACPHASENOISE\)](#) in the *HSPICE User Guide: RF Analysis*.



---

## .ALIAS

Renames a model or library containing a model; deletes an entire library of models.

### Syntax

```
.ALIAS model_name1 model_name2
```

### Description

Use in instances when you have used `.ALTER` commands to rename a model, to rename a library containing a model, or to delete an entire library of models in HSPICE. If your netlist references the old model name, then after you use one of these types of `.ALTER` commands, HSPICE no longer finds this model.

For example, if you use `.DEL LIB` in the `.ALTER` block to delete a library, the `.ALTER` command deletes all models in this library. If your netlist references one or more models in the deleted library, then HSPICE no longer finds the models.

To resolve this issue, HSPICE provides an `.ALIAS` command to let you keep the old model name that HSPICE can find in the existing model libraries.

### Examples

*Example 1* For a scenario in which you delete a library named `poweramp` that contains a model named `pa1`, while another library contains an equivalent model named `par`: You can then convert the `pa1` model name to the `par1` model name.

```
.ALIAS pa1 par1
```

*Example 2* During simulation when HSPICE encounters a model named `pa1` in your netlist, it initially cannot find this model because you used an `.ALTER` command to delete the library that contained the model. However, the `.ALIAS` command indicates to use the `par1` model in place of the old `pa1` model and HSPICE does find this new model in another library so simulation continues. You must specify an old model name and a new model name to use in its place. You cannot use `.ALIAS` without any model names:

```
.ALIAS
```

or with only *one* model name:

```
.ALIAS pa1
```

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .ALIAS

*Example 3* You also cannot alias a model name to more than one model name because the simulator cannot determine which of these new models to use in place of the deleted or renamed model. For the same reason, you cannot substitute a model name to a second model name and then substitute the second model name to a third model name.

```
.ALIAS pa1 par1 par2
```

*Example 4* If your netlist does not contain an .ALTER command and if the .ALIAS does not report a usage error, then the .ALIAS does not affect the simulation results.

```
.ALIAS pa1 par1  
.ALIAS par1 par2
```

Your netlist might contain the command:

```
.ALIAS myfet nfet
```

Without an .ALTER command, HSPICE does not use `nfet` to replace `myfet` during simulation.

If your netlist contains one or more .ALTER commands, the first simulation uses the original `myfet` model. After the first simulation if the netlist references `myfet` from a deleted library, .ALIAS substitutes `nfet` in place of the missing model.

- If HSPICE finds model definitions for both `myfet` and `nfet`, it reports an error and aborts.
- If HSPICE finds a model definition for `myfet`, but not for `nfet`, it reports a warning and simulation continues by using the original `myfet` model.
- If HSPICE finds a model definition for `nfet`, but not for `myfet`, it reports a “replacement successful” message.

### See Also

[.ALTER](#)  
[.MALIAS](#)

---

## .ALTER

Reruns an HSPICE/HSPICE RF simulation using different parameters and data.

### Syntax

```
.ALTER title_string
```

---

Argument	Description
<code>title_string</code>	Any string up to 80 characters. HSPICE prints the appropriate title string for each <code>.ALTER</code> run in each section heading of the output listing and in the graphical data ( <code>.tr#</code> ) files.

---

### Description

Use this command to rerun an HSPICE simulation using different parameters and data. Use parameter (variable) values for `.PRINT` commands before you alter them. The `.ALTER` block cannot include `.PRINT`, or any other input/output commands. You can include analysis commands (`.DC`, `.AC`, `.TRAN`, `.FOUR`, `.DISTO`, `.PZ`, and so on) in a `.ALTER` block in an input netlist file.

However, if you change only the analysis type and you do not change the circuit itself, then the simulation runs faster if you specify all analysis types in one block, instead of using separate `.ALTER` blocks for each analysis type.

To activate multiprocessing while running `.ALTER` cases, enter **hspice -mp** on the command line. While running in parallel mode, HSPICE checks if the input case has `.ALTER` commands. If it has, HSPICE splits the input case into several subcases, then fork HSPICE processes to run each subcase at the same time. After all HSPICE processes finish running the subcases, HSPICE merges all the output files of the subcases.

The `.ALTER` sequence or block can contain:

- Element commands (except E, F, G, H, I, and V source elements)
- [.AC](#) commands
- [.ALIAS](#) commands
- [.DATA](#) commands
- [.DC](#) commands
- [.DEL LIB](#) commands

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .ALTER

- .HDL commands
- .IC (initial condition) commands
- .INCLUDE (or) .INC (or) .INCL commands
- .LIB commands
- .MODEL commands
- .NODESET commands
- .OP commands
- .OPTION (or) .OPTIONS commands
- .PARAM (or) .PARAMETER (or) .PARAMETERS commands
- .TEMP (or) .TEMPERATURE commands
- .TF commands
- .TRAN commands
- .VARIATION commands

**Caution:** When using an .INCLUDE command within an .ALTER statement, the purpose of this feature is to enable you to *slightly* modify the original netlist; perhaps adding some elements/nodes without changing or deleting any elements/nodes that were already defined in the original .INC. This feature is *not* intended or able to significantly modify elements/nodes to the previously existing circuit topology. Using .INC statements within an .ALTER that disregard this limitation will yield simulation results that are unlikely to reflect the reality of the intended netlist.

**Note:** Beginning with the B-2008.09-SP1 release, HSPICE reports the elapsed time for the top level simulation and each .ALTER block separately.

### Examples

```
.ALTER simulation_run2
```

### See Also

[.OPTION ALTCC](#)  
[.OPTION MEASFILE](#)  
[.OPTION OPTCON](#)  
[Multiprocessing \(MP\) DC Monte Carlo](#)

---

## .APPENDMODEL

Appends the .MOSRA (model reliability) parameters to a model card.

### Syntax

```
.APPENDMODEL SrcModel ModelKeyword1 DestModel ModelKeyword2
```

Argument	Description
SrcModel	Source model name, e.g., the name of the MOSRA model.
ModelKeyword	Model type for SrcModel. For example, the keyword <code>mosra</code> .
DestModel	Destination model name, e.g, the original model in the model library.
ModelKeyword2	Model type for DestModel. For example, 'nmos'.

### Description

Appends the parameter values from the source model card (SrcModel) to the destination model card (DestModel). All arguments are required. Wildcards are supported for the .APPENDMODEL command. In addition, the .OPTION APPENDALL enables the top hierarchical level to use the .APPENDMODEL command even if the MOSFET model is embedded in a subcircuit.

### Examples

*Example 1* Appending the content of the model card `hci_1` to the `b3_nch BSIM3` model card.

```
.appendmodel hci_1 mosra b3_nch nmos
```

*Example 2* Model `p1_ra` is appended to all of the `pmos` models. Quotation marks are required if the model name is defined only by a wildcard.

```
.appendmodel p1_ra mosra "*" pmos
```

*Example 3* The model `p1_ra` is appended to all of the `pmos` models that are named `pch*` (`pch1`, `pch2`, `pch_tt`, etc.).

```
.appendmodel p1_ra mosra pch* pmos
```

### See Also

[.MODEL](#)  
[.MOSRA](#)  
[.OPTION APPENDALL](#)

---

## .BA\_ACHECK

Specifies the rule for detecting node activity in back-annotation.

### Syntax

```
.BA_ACHECK [include=node_pattern]  
+ [exclude=node_pattern]  
+ [level=val] [level=val2 0/1/n]  
+ [dv=val] [start=start_time] [stop=stop_time]
```

---

Argument	Description
include= node_pattern	Defines the signal node name(s) which can be the node name of a single node or a node name containing wildcard character '*' representing a group of node names. The node name with wildcard character must be quoted by single quotation marks as ' <i>node_name</i> ', because in HSPICE syntax, all characters after unquoted '*' are treated as comments and are ignored.
exclude= node_pattern	Defines the signal node name(s) which are excluded from the list of nodes that need to be checked. Wildcard characters can be used and need to be quoted such as: 'a*'.  
level=val	Defines the threshold of voltage variation. A node is considered active when the voltage change, compared to the initial value of the node, is larger than <i>val</i> . DEFAULT of <i>val</i> is 0.1 volt.
level=val2	The level value <i>val2</i> specifies the number of hierarchical depth levels when checking node activity. <ul style="list-style-type: none"><li>▪ When <i>val2</i> is set to 1, only nodes in the root circuit are considered for node activity.</li><li>▪ When <i>val2</i> is set to <i>n</i>, nodes in the range from the root circuit to <i>n</i>th level subckt are considered for node activity.</li><li>▪ When <i>val2</i> is set to 0, all subckt levels are considered for node activity.</li><li>▪ The default value of <i>val2</i> is 0.</li></ul>
dv=val	Defines the threshold of voltage variation. A node is considered active when the voltage change, compared to the initial value of the node, is larger than <i>val</i> . DEFAULT of <i>val</i> is 0.1 volt.
start= start_time, stop= stop_time	Specifies the <i>start_time</i> and <i>stop_time</i> in the time window. The activity is checked at the time within the specified time. If no time window is specified, the check is performed from the time 0 ns to the end of simulation.

---

### Description

Use this option to specify the rule for detecting node activity. A node is considered active if its voltage change exceeds the specified threshold during the simulation time.

**Note:** The `.BA_ACHECK` command is similar to the `HSIM` command:  
`acheck.`

### See Also

[Post-Layout Back-Annotation](#)

## **.BIASCHK**

Monitors the voltage bias, current, device size, expression, and region.

### **Syntax**

As an expression monitor

```
.BIASCHK 'expression' [limit=lim] [noise=ns]  
+ [max=max] [min=min]  
+ [simulation=op|dc|tr|all] [monitor=v|i|w|l]  
+ [tstart=time1] [tstop=time2] [autostop]  
+ [interval=time]
```

As an element and model monitor

```
.BIASCHK type terminal1=t1 [terminal2=t2]  
+ [limit=lim] [noise=ns] [max=max] [min=min]  
+ [simulation=op|dc|tr|all] [monitor=v|i]  
+ [name=name1,name2,...] ]  
+ [mname=modname_1,modname_2,...] ]  
+ [tstart=time1] [tstop=time2] [autostop]  
+ [except=name_1,name_2,...] ]  
+ [interval=time] [sname=subckt_name1,subckt_name2,...] ]
```

As a region monitor

```
.BIASCHK MOS [region=cutoff|linear|saturation]  
+ [simulation=op|dc|tr|all]  
+ [name=name1,name2,...] ]  
+ [mname=modname_1,modname_2,...] ]  
+ [tstart=time1] [tstop=time2] [autostop]  
+ [except=name1,name2,...] ]  
+ [interval=time] [sname=subckt_name1,subckt_name2,...] ]
```

As a length and width monitor

```
.BIASCHK type monitor=w|l  
+ [limit=lim] [noise=ns] [max=max] [min=min]  
+ [simulation=op|dc|tr|all]  
+ [name=devname_1,devname_2,...] ]  
+ [name=devname_n,devname_n+1,...] ]  
+ [mname=modelname_1,modelname_2,...] ]  
+ [tstart=time1] [tstop=time2] [autostop]  
+ [interval=time] [sname=subckt_name1,subckt_name2,...] ]
```



Argument	Description
type	Element type to check. MOS (C, BJT, ...) For a monitor, <i>type</i> can be DIODE, BIPOLAR, BJT, JFET, MOS, NMOS, PMOS, C, or SUBCKT. When used with REGION, <i>type</i> can be MOS only.
terminal 1, 2	Terminals between which HSPICE checks (that is, checks between <i>terminal1</i> and <i>terminal2</i> ): <ul style="list-style-type: none"> <li>▪ For MOS level 57: nd, ng, ns, ne, np, n6</li> <li>▪ For MOS level 58: nd, ngf, ns, ngb</li> <li>▪ For MOS level 59: nd, ng, ns, ne, np</li> <li>▪ For other MOS level: nd, ng, ns, nb</li> <li>▪ For capacitor: n1, n2</li> <li>▪ For diode: np, nn</li> <li>▪ For bipolar: nc, nb, ne, ns</li> <li>▪ For JFET: nd, ng, ns, nb</li> </ul> For type=subckt, the terminal names are those pins defined by the subcircuit definition of mname.
limit	Bias check limit that you define. Reports an error if the bias voltage (between appointed terminals of appointed elements and models) is larger than the limit.
noise	Bias check noise that you define. The default is 0.1v. Noise-filter some of the results (the local maximum bias voltage that is larger than the limit). The next local max replaces the local max if all of the following conditions are satisfied: <i>local_max-local_min noise</i> . <i>next local_max-local_min noise</i> . This local max is smaller than the next local max. For a parasitic diode, HSPICE ignores the smaller local max biased voltage and does not output this voltage. To disable this feature, set the noise detection level to 0.
max	Maximum value.
min	Minimum value.

Argument	Description
name	<p>Element name to check. If name and mname are not both set for the element type, the elements of this type are all checked. You can define more than one element name in keyword name with a comma (,) delimiter. If doing bias checking for subcircuits:</p> <ul style="list-style-type: none"> <li>▪ When both mname and name are defined while multiple name definitions are allowed if a name is also an instance of mname, then only those names are checked, others will be ignored.</li> <li>▪ This command is ignored if no name is an instance of mname.</li> <li>▪ For name definitions which are not of the type defined in mname will be ignored.</li> <li>▪ If a mname is not defined, the subcircuit type is determined by the first name definition.</li> </ul>
mname	<p>Model name. If you are doing bias checking for a subcircuit, it is the subcircuit definition name. HSPICE checks elements of the model for bias. If you define mname, then HSPICE checks all devices of this model. You can define more than one model name in the keyword mname with the comma (,) delimiter. If <i>mname</i> and <i>name</i> are not both set for the element <i>type</i>, the elements of this type are all checked. If doing bias checking for subcircuits:</p> <ul style="list-style-type: none"> <li>▪ Once there is one and only one mname defined, the terminal names for this command are those pins defined by the subckt definition of mname.</li> <li>▪ Multiple mname definitions are not allowed.</li> <li>▪ Wildcards are supported for mname.</li> <li>▪ If only mname is specified in a subckt bias check, then all subcircuits will be checked.</li> </ul> <p>See also sname below.</p>
region	<p>Values can be cutoff, linear, or saturation. HSPICE monitors when the MOS device, defined in the .BIASCHK command, transitions to and from the specified region (such as cutoff).</p>
simulation	<p>Simulation type you want to monitor. You can specify op, dc, tr (transient), and all (op, dc, and tr). The tr option is the default simulation type.</p>
monitor	<p>Type of value you want to monitor. You can specify v (voltage), i (current), w, and l (device size) for the element type. This parameter is not used for an expression-type monitor.</p>
tstart	<p>Bias check start time during transient analysis. The default is 0.</p>

Argument	Description
tstop	Bias check end time during transient analysis. The analysis ends on its own by default if you do not set this parameter.
autostop	When set, HSPICE supports an autostop for a biaschk card so that it can report error messages and stop the simulation immediately.
except	Specify the element or instance that you do not want to bias check.
interval	Active when .OPTION BIASINTERVAL is set to a nonzero value. This argument prevents reporting intervals that are less than or equal to the time specified.
sname	Name of the subcircuit definition that the <i>type</i> of element of lies in. HSPICE checks all elements in this subcircuit for bias. You can define more than one subcircuit name in the keyword <i>sname</i> with a comma (,) delimiter. If you are doing bias checking for a subcircuit, sname = the X-element name.

### Description

Use this command to monitor the voltage bias, current, device size, expression, and region during analysis. The output reports:

- Element (instance) name
- Time
- Terminals
- Bias that exceeds the limit
- Number of times the bias exceeds the limit for an element

HSPICE saves the information as both a warning and a bias check summary in the *\*.lis* file or a file you define in the BIASFILE option. You can use this command only for active elements, capacitors, and subcircuits.

More than one simulation type or all simulation types can be set in a single .BIASCHK command. Also, more than one region can be set in a single .BIASCHK command.

After a simulation that uses the .BIASCHK command runs, HSPICE outputs a results summary including the element name, time, terminals, model name, and the number of times the bias exceeded the limit for a specified element.

The keywords *name*, *mname*, and *sname* act as OR'd filters for element selection. Also, if *type* is *subckt* in a .BIASCHK command that tries to check

the ports of a subcircuit, the keyword *sname* then behaves identically to the *name* keyword.

Element and model names can contain wildcards, either “?” (stands for one character) or “\*” (stands for 0 or more characters).

If a model name that is referenced in an active element command contains a period (.), then .BIASCHK reports an error. This occurs because it is unclear whether a reference such as `x.123` is a model name or a subcircuit name (123 model in “x” subcircuit).

If you do not specify an element and model name, HSPICE checks all elements of this type for bias voltage (you must include `type` in the BIASCHK card). However, if `type` is `subckt` at least one element or model name must be specified in the .BIASCHK command; otherwise, a warning message is issued and this command is ignored.

**Note:** To perform a complete bias check and print all results in the Outputs Biaschk Report, do not use `.protect/.unprotect` in the netlist for the part that is used in `.biaschk`. For example: If a model definition such as `model nch` is contained within `.prot/.unprot` commands, in the `*.lis` you'll see a warning message as follows: `**warning** : model nch defined in .biaschk cannot be found in netlist--ignored`

## Examples

*Example 1 Monitoring an expression:*

```
.biaschk 'v(1)' min='v(2)*2' simulation= op
```

*Example 2 Monitoring element m1 and model types between two specified terminals.*

```
.biaschk nmos terminal1=ng terminal2=ns simulation=tr name=m1
```

*Example 3 Monitoring MOSFET model m1 whose bias voltage exceeds 2.5 V and interval exceeds 5 ns.*

```
.biaschk nmos terminal1=nb terminal2=ng limit=2.5  
+ mname=m1 interval=5n
```

*Example 4 The following two examples use .BIASCHK commands that do not require terminal specifications. Example 4 monitors the MOS transistor region of operation*

```
.biaschk mos region=saturation name=x1.m1 mname=nch name=m2
```

*Example 5 Monitors MOS transistor length and width.*

```
.biaschk mos monitor=l mname=m* p* min=1u minu=op
```

*Example 6 Defines the differences between using .BIASCHK with a MOSFET instance and macro models. If the MOSFET in your netlist is written as follows:*

```
mp0 g d s sub dgxnfet w=1.0u l=0.18u
```

...then the .BIASCHK statement can be written as:

```
.biaschk pmos terminal1=ns terminal2=nd mname=m*.dgnfet*  
+ limit=0.9v
```

*Example 7 If a macro model is used, then the MOSFET is defined inside a subcircuit:*

```
.subckt test g d  
mp0 g d s sub dgxnfet w=1.0u l=0.18u  
.model dgxnfet nmos level=54  
.ends  
X1 g d test
```

For this case

```
.biaschk pmos terminal1=ns terminal2=nd mname=x*.dgnfet*  
+ limit=0.9v
```

For a full example netlist go to: *\$installdir/demo/hspice/apps/biaschk.sp*

## See Also

- [.OPTION BIASFILE](#)
- [.OPTION BIASINTERVAL](#)
- [.OPTION BIASNODE](#)
- [.OPTION BIASPARALLEL](#)
- [.OPTION BIAWARN](#)
- [.PROTECT or .PROT](#)
- [.UNPROTECT or .UNPROT](#)

---

## .CFL\_PROTOTYPE

Specifies function protocol type for the Compiled Function Library capability.

### Syntax

```
.CFL_PROTOTYPE function_name(arg1_type, arg2_type, ..., argn_type)
```

---

Argument	Description
function_name	CFL function name
arg_type	input argument type; it can be <ul style="list-style-type: none"><li>▪ <i>double</i>: (default) keyword argument passed to C library as C language's "double" type for C code used in C-based file (not HSPICE netlist); used to generate the *.so file (CFL library file)</li><li>▪ <i>param</i>: parameter storage reference as output only</li></ul>

---

### Description

This command specifies a function type.

Function types include:

- a predefined parameter value
- a mathematical expression of multiple predefined parameter values
- a built-in mathematical function in the standard library
- an output of another evaluated CFL function

The CFL function can re-assign local and global parameter values. Only local functions with the parameters in the argument list are updated with the new local values. The global functions are updated with the new global values.

The following rules apply:

- CFL functions cannot be used in .PRINT, .PROBE and .MEASURE statements.
- Only a single compiled CFL \*.so file is allowed in the simulation input netlist.
- Parameter definition must be order-dependent when using multiple return values for the CFL functions (unlike the current HSPICE order-independent parameter definition requirement).
- If the CFL function name is same as the user-defined function (UDF), the UDF is used and CFL is not called.

**Note:** In the C code, the protocol type of C library must be `func(argc, argv)`, where `argc` is argument number, and `argv` is the argument array. See Examples 2 and 3 for sample CFL function syntax.

The CFL feature requires setting an environment variable, `CFL_COMPILED_LIB CFL_library_file_name`, (\*.so file) and use of the `.OPTION CFLFLAG` to enable it in a netlist. For other descriptive information on the Compiled Library Function see [Features](#) in the *HSPICE User Guide: Simulation and Analysis*.

## Examples

*Example 1* Note the use of the “&” notation that signifies the bidirectional nature of an argument which means the value of the argument is updated upon returning from the function. The example presents multiple return values for the CFL functions:

```
.CFL_PROTOTYPE xyz_eval (double, param)
.param p1=5
.param p2=9
.param p3= xyz_eval(p1, &p2)
```

*Example 2* In this Sample CFL function, the content of the “return” parameters does not affect the calculated return values from the function with the same arguments. Parameters passed into the functions as references are used only as return values and do not contribute to the calculation inside the CFL function in any form.

```
double func1 (int argc, long **argv)
{
    double a1 = *(double*)argv[0];
    double *a2 = (double*)argv[1];
    return eval1_func(a1, a2);
}
double eval1_func (double arg1, double *arg2)
{
    double val = 0;
    *arg2 = arg1 + 2;
    val = (*arg2) * (arg1 + 4)
    return val
}
```

In Example 2, CFL C-code Function Implementation, CFL functions are an arbitrary number of function arguments with any combination of the argument

base type as either “double” or parameter address. Example 3 is the function prototype:

*Example 3 Netlist Showing Redefinition of User Functions*

```
static double
eval1_func(double a1, double *a2)
{
  *a2 = a1 + 10;
  return a1 - 4;}
double xyz_eval_1(int argc, long **argv)
{
  double a1 = *(double *) (argv[0]);
  double *a2 = (double*)argv[1];
  return eval1_func(a1, a2);
}
static double
eval2_func(double *a1, double a2, double *a3)
{
  *a1 = a2 + 2;
  *a3 = a2 - 3;
  return a2 - 1;
}
double xyz_eval_2(int argc, long **argv)
{
  double *a1 = (double*) (argv[0]);
  double a2 = *(double *)argv[1];
  double *a3 = (double *)argv[2];
  return eval2_func(a1, a2, a3);
}
```

*Example 4 Redefinition following evaluations*

```
.param p2 = 10
.param p1 = 2
.param p3 = func1 (p1, &p2)
```

After returning from evaluating func1 (), p3=24, and user function are redefined to p2=4.

```
.subckt INV
.param p1 = 3
.param p2 = 3
.param p3 = func1(p1, &p2)
```



After returning from evaluating `func1()`, the user function `p1=3` is redefined to `p2=5` while `p3=35`.

```
.param p3 = 0
.param p4 = func1(p2, &p3)
```

After the evaluation of `func1()`, `p2=5` and `p3=7` & `p4=63`.

```
.ends INV
```

*Example 5 Sample Netlist*

```
*
.cfl_prototype xyz_eval_1(double, param)
.cfl_prototype xyz_eval_2(param, double, param)
*
.param p1 = 5
.param p2 = 8
.param p3 = 9
.param p4 = 7
.param p5 = 10
*
.param p7 = xyz_eval_1(p1, &p2) * p2 = 15 ; p7 = 1
.param p8 = xyz_eval_2(&p3, p4, &p5) * p3 = 9; p5 = 4; p8 = 6
.param p9 = p8 + p5 - p3 + p2
*
R1 n1 A r="p2" * R = 15
R2 n3 B r="p7" * R = 1
M1 n1 n2 n3 NMOS w=5u l=6u bqi="p9" * bqi = 16
```

**See Also**

[.OPTION CFLFLAG](#)

## .CHECK EDGE

Verifies that a triggering event provokes an appropriate RISE or FALL action in HSPICE RF.

### Syntax

```
.CHECK EDGE (ref RISE | FALL minmax RISE | FALL)
+ node1 [node2 ...] (hi lo hi_th low_th)
```

Argument	Description
ref	Name of the reference signal.
min	Minimum time.
max	Maximum time.
node1 node2 ...	List of nodes to which you apply the edge condition.
hi lo hi_th lo_th	Logic levels for the timing check.

### Description

Use a `.CHECK EDGE` command to verify that a triggering event provokes an appropriate RISE or FALL action within the specified time window.

### Examples

This example sets the condition that the rising action of the clock (clk) triggers the falling edge of VOUTA within 1 to 3 ns, as shown in [Figure 1](#):

```
.CHECK EDGE (clk RISE 1ns 3ns FALL) VOUTA
```

Values for hi, lo, and the thresholds were defined in a `.CHECK GLOBAL_LEVEL` command placed earlier in the netlist.

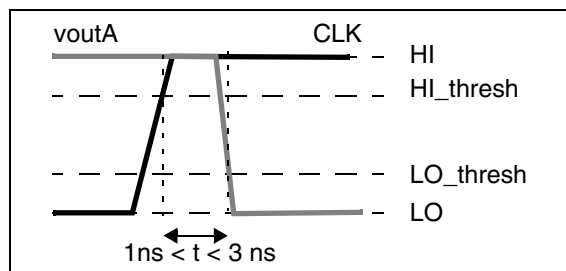


Figure 1 EDGE Example

**See Also**

[.CHECK HOLD](#)  
[.CHECK GLOBAL\\_LEVEL](#)  
[.CHECK SETUP](#)

---

## .CHECK FALL

Verifies that a fall time occurs within a specified time window in HSPICE RF.

### Syntax

```
.CHECK FALL (minmax) node1 [node2 ...]
(hi lo hi_th lo_th)
```

---

Argument	Description
min	Lower boundary for the time window.
max	Upper limit for the time window.
node1 node2 ...	List of all nodes to check.
hi lo hi_th lo_th	Logic levels for the timing check.

---

### Description

Use a `.CHECK FALL` command verifies that a fall time occurs within the specified window of time.

### See Also

- [.CHECK GLOBAL\\_LEVEL](#)
- [.CHECK RISE](#)
- [.CHECK SLEW](#)

---

## .CHECK GLOBAL\_LEVEL

Globally sets specified high and low definitions for all CHECK commands in HSPICE RF.

### Syntax

```
.CHECK GLOBAL_LEVEL (hi lo hi_th lo_th)
```

---

Argument	Description
hi	Value for logic high.
lo	Value for logic low.
hi_th	Is the minimum value considered high.
lo_th	Is the maximum value considered low.

---

### Description

Use this command to globally set the desired high and low definitions for all CHECK commands. The high and low definitions can be either numbers or expressions, and *hi\_th* and *lo\_th* can be either absolute values or percentages if punctuated with the % symbol. You can also locally set different logic levels for individual timing checks.

### Examples

*Example 1* Defines a logic high as 5 volts and a logic low as 0 volts. A voltage value as small as 4 V is considered high, while a value up to 1 V is low.

```
.CHECK GLOBAL_LEVEL (5 0 4 1)
```

*Example 2* Illustrates an alternative definition for the first example.

```
.CHECK GLOBAL_LEVEL (5 0 80% 20%)
```

### See Also

- [.CHECK EDGE](#)
- [.CHECK FALL](#)
- [.CHECK HOLD](#)
- [.CHECK IRDROP](#)
- [.CHECK RISE](#)
- [.CHECK SLEW](#)

## .CHECK HOLD

Ensures that specified signals do not switch for a specified period of time in HSPICE RF.

### Syntax

```
.CHECK HOLD (ref RISE | FALL duration RISE | FALL)
+ node1 [node2 ...] (hi lo hi_th lo_th)
```

Argument	Description
ref	Reference or trigger signal.
duration	Minimum time required after the triggering event before the specified nodes can rise or fall.
node1 node2 ...	List of nodes for which the HOLD condition applies.
hi lo hi_th lo_th	Logic levels for the timing check.

### Description

Use this command to ensure that the specified signals do not switch for a specific period of time.

### Examples

This example specifies that  $vin^*$  (such as  $vin_1$ ,  $vin_2$ , and so on), must not switch for 2ns after every falling edge of nodeA (see [Figure 2](#)).

```
.CHECK HOLD (nodeA FALL 2ns RISE) vin*
```

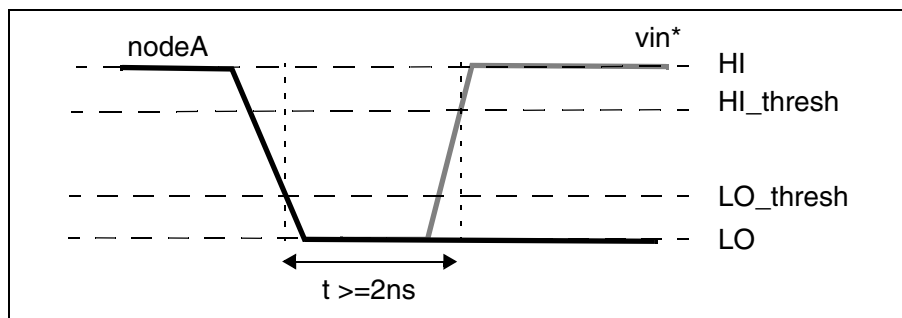


Figure 2 HOLD Example

**See Also**

[.CHECK EDGE](#)  
[.CHECK GLOBAL\\_LEVEL](#)  
[.CHECK SETUP](#)

---

## .CHECK IRDROP

Verifies that IR drop does not fall below or exceed a specified value in HSPICE RF.

### Syntax

```
.CHECK IRDROP (volt_val tduration) node1 [node2 ...]  
+ (hi lo hi_th lo_th)
```

---

Argument	Description
volt_val	Limiting voltage value. <ul style="list-style-type: none"><li>▪ A positive <i>volt_val</i> (voltage value) indicates ground bounce checking.</li><li>▪ A negative <i>volt_val</i> denotes VDD drop.</li></ul>
duration	Maximum allowable time. If you set duration to 0, then HSPICE RF reports every glitch that strays beyond the specified <i>volt_val</i> .
node1 [node2 ...]	List of nodes for which the IR drop checking applies.
hi lo hi_th lo_th	Logic levels for the timing check.

---

### Description

Use this command to verify that the IR drop does not fall below or exceed a specified value for a specified duration.

### Examples

This example specifies that v1 must not fall below -2 volts for any duration exceeding 1ns (see [Figure 3](#)).

```
.CHECK IRDROP (-2 1ns) v1
```

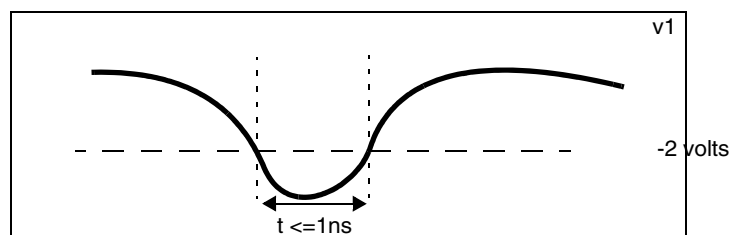


Figure 3 IR Drop Example



**See Also**

.CHECK EDGE  
.CHECK GLOBAL\_LEVEL  
.CHECK SETUP

## .CHECK RISE

Verifies that a rise time occurs within a specified time window in HSPICE RF.

### Syntax

```
.CHECK RISE (minmax) node1 [node2 ...] (hi lo hi_th lo_th)
```

Argument	Description
min	Lower boundary for the time window.
max	Upper limit for the time window.
node1 node2 ...	List of all nodes to check.
hi lo hi_th lo_th	Logic levels for the timing check.

### Description

Use this command to verify that a rise time occurs within the specified window of time.

### Examples

This example defines a window between 1.5ns and 2.2ns wide, in which the va and vb signals must complete their rise transition (see [Figure 4](#)). Values for the HI, LO, and the thresholds were defined in a `.CHECK GLOBAL_LEVEL` command placed earlier in the netlist.

```
.CHECK RISE (1.5ns 2.2ns) va vb
```

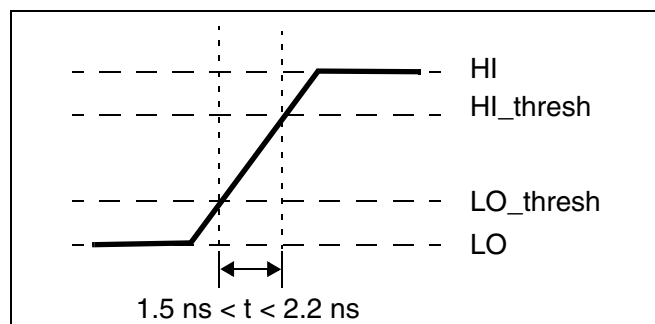


Figure 4 RISE Time Example

**See Also**

[.CHECK GLOBAL\\_LEVEL](#)  
[.CHECK FALL](#)  
[.CHECK SLEW](#)

## .CHECK SETUP

(RF) Verifies that specified signals do not switch for a specified time-period.

### Syntax

```
.CHECK SETUP (ref RISE | FALL duration RISE | FALL)
+ node1 [node2 ...] (hi lo hi_th lo_th)
```

Argument	Description
ref	Reference or trigger signal.
duration	Minimum time before the triggering event during which the specified nodes cannot rise or fall
node1 [node2 ...]	List of nodes for which the HOLD condition applies.
hi lo hi_th lo_th	Logic levels for the timing check.

### Description

Use to verify that specified signals do not switch for a specified period of time.

### Examples

This example specifies that v1 and v2 must not switch for 2 ns before every rising edge of nodeA (see [Figure 5](#)).

```
.CHECK SETUP (nodeA RISE 2ns FALL) v1 v2
```

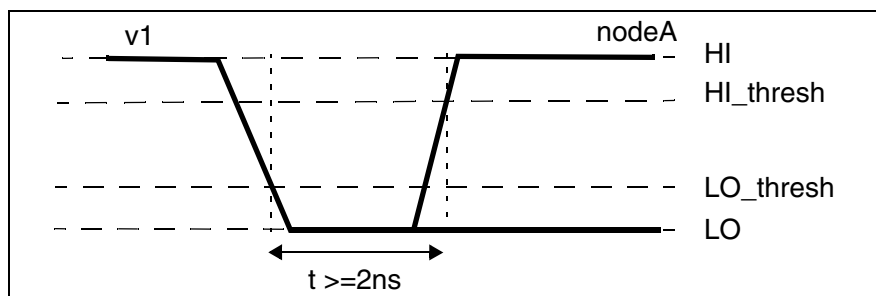


Figure 5 SETUP Example

### See Also

- [.CHECK EDGE](#)
- [.CHECK GLOBAL\\_LEVEL](#)
- [.CHECK HOLD](#)

## .CHECK SLEW

Verifies that a slew rate occurs within a specified time window in HSPICE RF.

### Syntax

```
.CHECK SLEW (minmax) node1 [node2 ...] (hi lo hi_th lo_th)
```

Argument	Description
min	Lower boundary for the time window.
max	Upper limit for the time window.
node1 node2 ...	List of all nodes to check.
hi lo hi_th lo_th	Logic levels for the timing check.

### Description

Use this command to verify that a slew rate occurs within specified time range.

### Examples

This example sets the condition that nodes starting with a\* nodes must have a slew rate between  $(HI\_thresh - LO\_thresh)/3ns$  and  $(HI\_thresh - LO\_thresh)/1ns$ . If either node has a slew rate greater than that defined in the .CHECK SLEW command, HSPICE RF reports the violation in the .err file.

```
.CHECK SLEW (1ns 3ns) a* (3.3 0 2.6 0.7)
```

The slew rate check in [Figure 6](#) defines its own *hi*, *lo*, and corresponding threshold values, as indicated by the four values after the node names.

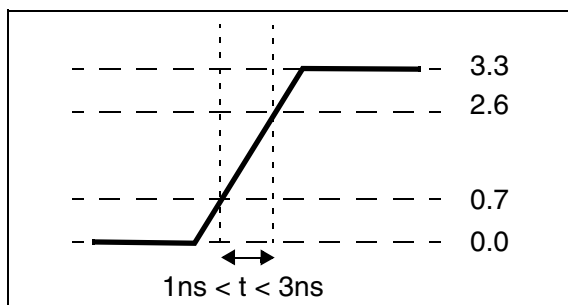


Figure 6 SLEW Example

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

.CHECK SLEW

### See Also

.CHECK FALL  
.CHECK GLOBAL\_LEVEL  
.CHECK RISE

---

## .CONNECT

Connects two nodes together; the first node replaces the second node in the simulation.

### Syntax

```
.CONNECT node1node2
```

---

Argument	Description
node1	Name of the first of two nodes to connect together
node2	Name of the second of two nodes to connect together. This node is replaced by Node1, which is the first node, in the simulation.

---

### Description

Use this command to connect two nodes together in your netlist. This causes the simulation to evaluate the two nodes as if they were only one node that uses the name of the first node. The name of the second node is not recognized in the simulation. Both nodes must be at the same level in the circuit design that you are simulating: you cannot connect nodes that belong to different subcircuits.

A `.CONNECT` statement in the higher level does not work on the `.SUBCKT` node.

### Examples

*Example 1* `A` is the name of the first of two nodes to connect together and `VSS` is the name of the second of two nodes to connect together. `A`, the first node, replaces the second node, `VSS`, in the simulation.

```
.CONNECT A VSS

...
.subckt eye_diagram node1 node2 ...
.connect node1 node2
...
.ends
```

*Example 2* Example 1 now is the same as the following:

```
...
.subckt eye_diagram node1 node1 ...
...
.ends
...
```

HSPICE reports the following error message:

```
**error**: subcircuit definition duplicates node node1
```

To apply any HSPICE command to *node2*, apply it to *node1*, instead. Then, to change the netlist construction to recognize *node2*, use an `.ALTER` command.

HSPICE reports the following error message:

```
**error**: subcircuit definition duplicates node node1
```

To apply any HSPICE command to *node2*, apply it to *node1*, instead. Then, to change the netlist construction to recognize *node2*, use an `.ALTER` command.

In the following variation of the example, *node1* *node2* are not be the same in this situation and the duplicated node error message will not be issued.

```
.connect node1 node2
.subckt eye_diagram node1 node2 ...
...
.ends
```

*Example 3* The first `.TRAN` simulation includes two resistors. Later simulations have only one resistor because *r2* is short-circuited by connecting *cc* with 1. *v(1)* does not print out, but *v(cc)* prints out instead.

```
*example for .connect
vcc 0 cc 5v
r1 0 1 5k
r2 1 cc 5k
.tran 1n 10n
.print i(vcc) v(1)
.alter
.connect cc 1
.end
```

*Example 4* Shows how to use multiple `.CONNECT` commands to connect several nodes together. This example connects both *node2* and *node3* to *node1*. All connected nodes must be in the same subcircuit or all in the main



*circuit. The first HSPICE simulation evaluates only node1; node2 and node3 are the same node as node1. Use .ALTER commands to simulate node2 and node3.*

```
.CONNECT node1 node2
.CONNECT node2 node3
```

If you set .OPTION NODE, then HSPICE prints out a node connection table.

```
vcc cc 0 5v
r1 cc net1 5k
r2 net1 net2 5k
c1 net2 0 1n
.tran 1n 10n
.connect net2 0
.print i(vcc) v(net2)
.end
```

This causes the circuit elements to be connected as shown in Example 5:

#### *Example 5*

```
vcc cc net2 5v
r1 cc net1 5k
r2 net1 net2 5k
c1 net2 net2 1n
.tran 1n 10n
.connect net2 0
.print i(vcc) v(net2)
.end
```

For Example 5, HSPICE reports the following error message for the elements vcc r1 and r2, since there is now no ground node in the netlist.

```
**error** no dc path to ground from node
```

*Example 6 The correct way to connect net2 to ground is to specify the .CONNECT command as follows:*

```
.connect 0 net2
```

#### **See Also**

[.ALTER](#)  
[.OPTION NODE](#)

---

## .DATA

Concatenates or column-laminates data sets to optimize measured I-V, C-V, transient, or S-parameter data.

### Syntax

Inline command

```
.DATA datanm pnam1 [pnam2 pnam3 ... pnamxxx]
+ pval1 [pval2 pval3 ... pvalxxx]
+ pval1' [pval2' pval3' ... pvalxxx']
.ENDDDATA
```

External File command for concatenated data files

```
.DATA datanm MER
+ FILE='filename1' pname1=col_num [pname2=col_num ...]
+ [FILE='filename2' pname1=col_num
+ [pname2=col_num ...] ... [OUT='fileout']
.ENDDDATA
```

Column Laminated command (not available for HSPICE RF)

```
.DATA datanm LAM
+ FILE='filename1' pname1=col_num
+ [pname2=col_num ...]
+ [FILE='filename2' pname1=col_num
+ [pname2=col_num ...] ... [OUT='fileout']
.ENDDDATA
```

---

Argument	Description
col_num	Column number in the data file for the parameter value. The column does not need to be the same between files.
datanm	Data name—referenced in the .TRAN, .DC, or .AC command.
filename <i>i</i>	Data file to read. HSPICE concatenates files in the order they appear in the .DATA command. You can specify up to 10 files.
fileout <i>i</i>	Data file name, where simulation writes concatenated data. This file contains the full syntax for an inline .DATA command and can replace the .DATA command that created it in the netlist. You can output the file and use it to generate one data file from many.
LAM	Column-laminated (parallel merging) data files to use.

Argument	Description
MER	Concatenated (series merging) data files to use.
pnami	Parameter names—used for source value, element value, device size, model parameter value, and so on. You must declare these names in a <code>.PARAM</code> command.
pvali	Parameter value.

### Description

Use the `.DATA` command to concatenate or column-laminate data sets to optimize measured I-V, C-V, transient, or S-parameter data.

You can also use the `.DATA` command for a first or second sweep variable when you characterize cells and test worst-case corners. Simulation reads data measured in a lab, such as transistor I-V data, one transistor at a time in an outer analysis loop. Within the outer loop, the analysis reads data for each transistor (IDS curve, GDS curve, and so on), one curve at a time in an inner analysis loop.

Data-driven analysis syntax requires a `.DATA` command and an analysis command that contains a `DATA=dataname` keyword.

The `.DATA` command specifies parameters that change values, and the sets of values to assign during each simulation. The required simulations run as an internal loop. This bypasses reading-in the netlist and setting-up the simulation, which saves computing time. In internal loop simulation you can also plot simulation results against each other and print them in a single output.

You can enter any number of parameters in a `.DATA` block. The `.AC`, `.DC`, and `.TRAN` commands can use external and inline data provided in `.DATA` commands. For example, to specify the circuit temperature for an HSPICE simulation you can use the `.TEMP` command, the `TEMP` parameter in the `.DC`, `.AC`, and `.TRAN` commands, or the `TEMP/TEMPER` parameter in the first column of the `.DATA` command. The number of data values per line does not need to correspond to the number of parameters. For example, you do not need to enter 20 values on each line in the `.DATA` block if each simulation pass requires 20 parameters: the program reads 20 values on each pass, however the values are formatted.

Each `.DATA` command can contain up to 50 parameters. If you need more than 50 parameters in a single `.DATA` command, place 50 or less parameters in the `.DATA` command, and use `.ALTER` commands for the other parameters.

HSPICE refers to `.DATA` commands by their data names so each data name must be unique. HSPICE supports three `.DATA` command formats:

- Inline data, which is parameter data, listed in a `.DATA` command block. The `datanm` parameter in a `.DC`, `.AC`, or `.TRAN` analysis command, calls this command. The number of parameters that HSPICE reads determines the number of columns of data. The physical number of data numbers per line does not need to correspond to the number of parameters. For example, if the simulation needs 20 parameters you do not need 20 numbers per line.
- Data that is concatenated from external files. Concatenated data files are files with the same number of columns, placed one after another.
- Data that is column-laminated from external files. Column laminated data are columns of files with the same number of rows, arranged side-by-side.

To use external files with the `.DATA` format:

- Use the `MER` and `LAM` keywords to prepare HSPICE for external file data, rather than inline data.
- Use the `FILE` keyword to specify the external filename.
- Use simple file names, such as `out.dat` without single or double quotation marks ( ' ' or " " ), but use quotation marks when file names start with numbers, such as "1234.dat".
- Use the proper case, since file names are case sensitive on UNIX systems.

For data-driven analysis, specify the start time (time 0) in the analysis command so that the analysis correctly calculates the stop time.

The following shows how different types of analyses use `.DATA` commands: `.DC DATA=dataname`

Operating point: `.DC vin 1 5 .25 SWEEP DATA=dataname`

DC sweep: `.DC vin 1 5 .25 SWEEP DATA=dataname`

AC sweep: `AC dec 10 100 10meg SWEEP DATA=dataname`

TRAN sweep: `.TRAN 1n 10n SWEEP DATA=dataname`

### Examples

*Example 1* HSPICE performs these analyses for each set of parameter values defined in the `.DATA` command. For example, the program first uses the `width=50u`, `length=30u`, `thresh=1.2v`, and `cap=1.2pf` parameters to

*perform .TRAN, .AC, and .DC analyses. HSPICE then repeats the analyses for width=25u, length=15u, thresh=1.0v, and cap=0.8pf, and again for the values on each subsequent line in the .DATA block.*

```
* Inline .DATA statement
.TRAN 1n 100n SWEEP DATA=devinf
.AC DEC 10 1hz 10khz SWEEP DATA=devinf
.DC TEMP -55 125 10 SWEEP DATA=devinf
.DATA devinf width length thresh cap
+ 50u 30u 1.2v 1.2pf
+ 25u 15u 1.0v 0.8pf
+ 5u 2u 0.7v 0.6pf
.ENDDATA
```

**Example 2** HSPICE performs a DC sweep analysis for each set of *VBS*, *VDS*, and *L* parameters in the *.DATAvdot* block. That is, HSPICE runs eight DC analyses one for each line of parameter values in the *.DATA* block.

```
* .DATA as the inner sweep
M1 1 2 3 0 N W=50u L=LN
VGS 2 0 0.0v
VBS 3 0 VBS
VDS 1 0 VDS
.PARAM VDS=0 VBS=0 L=1.0u
.DC DATA=vdot
.DATA vdot
VBS VDS L
0 0.1 1.5u
0 0.1 1.0u
0 0.1 0.8u
-1 0.1 1.0u
-2 0.1 1.0u
-3 0.1 1.0u
0 1.0 1.0u
0 5.0 1.0u
.ENDDATA
```

**Example 3** *These values result in transient analyses at every time value from 0 to 100 ns in steps of 1 ns by using the first set of parameter values in the .DATA d1 block. Then HSPICE reads the next set of parameter values and does another 100 transient analyses. It sweeps time from 0 to 100*

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .DATA

*ns in 1 ns steps. The outer sweep is time and the inner sweep varies the parameter values. HSPICE performs 200 analyses: 100 time increments, times 2 sets of parameter values.*

```
* .DATA as the outer sweep
.PARAM W1=50u W2=50u L=1u CAP=0
.TRAN 1n 100n SWEEP DATA=d1
.DATA d1
      W1      W2      L      CAP
      50u     40u     1.0u   1.2pf
      25u     20u     0.8u   0.9pf
.ENDDATA
```

**Example 4** This example shows the external file .DATA for concatenated data files.

```
* External File .DATA for concatenated data files
.DATA datanm MER
+ FILE=filename1 pname1 = colnum
+ pname2=colnum ...
+ FILE=filename2 pname1=colnum
+ pname2=colnum ...
+ ...
+ OUT=fileout
.ENDDATA
```

If you concatenate the three files (*file1*, *file2*, and *file3*).

```
file1  file2  file3
a a a  b b b  c c c
a a a  b b b  c c c
a a a
```

The data appears as follows:

**Example 5**

```
a a a
a a a
a a a
b b b
b b b
c c c
c c c
```

The number of lines (rows) of data in each file does not need to be the same. The simulator assumes that the associated parameter of each column of the A

file is the same as each column of the other files. The `.DATA` command for this example is:

```
* External File .DATA statement
.DATA inputdata MER
  FILE='file1' p1=1 p2=3 p3=4
  FILE='file2' p1=1
  FILE='file3'
.ENDDATA
```

This example listing concatenates *file1*, *file2*, and *file3* to form the `inputdata` data set. The data in *file1* is at the top of the file, followed by the data in *file2*, and *file3*. The `inputdata` in the `.DATA` command references the data name specified in either the `.DC`, `.AC`, or `.TRAN` analysis commands. The parameter fields specify the column that contains the parameters (you must already have defined the parameter names in `.PARAM` commands). For example, the values for the `p1` parameter are in column 1 of *file1* and *file2*. The values for the `p2` parameter are in column 3 of *file1*. For data files with fewer columns than others, HSPICE assigns values of zero to the missing parameters.

(HSPICE only) In Example 5 three files (D, E, and F) contain the following columns of data:

#### Example 6

File D	File E	File F
d1 d2 d3	e4 e5	f6
d1 d2 d3	e4 e5	f6
d1 d2 d3	e4 e5	f6

The laminated data appears as follows:

d1 d2 d3	e4 e5	f6
d1 d2 d3	e4 e5	f6
d1 d2 d3	e4 e5	f6

The number of columns of data does not need to be the same in the three files.

The number of lines (rows) of data in each file does not need to be the same. HSPICE interprets missing data points as zero.

The `.DATA` command for this example is:

```
* Column-Laminated .DATA statement
.DATA dataname LAM
  FILE='file1' p1=1 p2=2 p3=3
  FILE='file2' p4=1 p5=2
  OUT='fileout'
.ENDDATA
```

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .DATA

This listing laminates columns from *file1* and *file2* into the *fileout* output file. Columns one, two, and three of *file1* and columns one and two of *file2* are designated as the columns to place in the output file. You can specify up to 10 files per .DATA command.

If you run HSPICE on a different machine than the one on which the input data files reside (such as when you work over a network), use full path names instead of aliases. Aliases might have different definitions on different machines.

#### See Also

[.AC](#)

[.DC](#)

[.ENDDATA](#)

[.PARAM \(or\) .PARAMETER \(or\) .PARAMETERS](#)

[.TRAN](#)



## .DC

Performs several types of sweeps during DC analysis.

### Syntax

Sweep or Parameterized Sweep:

```
.DC var1 START=start1 STOP=stop1 STEP=incr1
.DC var1 START=[param_expr1]
+ STOP=[param_expr2] STEP=[param_expr3]
.DC var1 start1 stop1 incr1
+ [SWEEP var2 type np start2 stop2]
.DC var1 start1 stop1 incr1 [var2 start2 stop2 incr2]
```

Data-Driven Sweep:

```
.DC var1 type np start1 stop1 [SWEEP DATA=datanm]
.DC DATA=datanm [SWEEP var2 start2 stop2 incr2]
.DC DATA=datanm
```

Monte Carlo:

```
.DC var1 type np start1 stop1 [SWEEP MONTE=MCcommand]
.DC MONTE=MCcommand
```

Optimization:

```
.DC DATA=datanm OPTIMIZE=opt_par_fun
+ RESULTS=measnames MODEL=optmod
.DC var1 start1 stop1 SWEEP OPTIMIZE=OPTxxx
+ RESULTS=measname MODEL=optmod
```

Argument	Description
DATA=datanm	Datanm is the reference name of a .DATA command.
incr1...	Voltage, current, element, or model parameters; or temperature increments.
MODEL	Optimization reference name. The .MODEL OPT command uses this name in an optimization analysis

Argument	Description
MONTE= MCcommand	Where MCcommand can be any of the following: <ul style="list-style-type: none"> <li>▪ <i>val</i> Specifies the number of random samples to produce.</li> <li>▪ <i>val</i> <i>firstrun=num</i> Specifies the sample number on which the simulation starts.</li> <li>▪ <i>list num</i> Specifies the sample number to execute.</li> <li>▪ <i>list(num1:num2 num3 num4:num5)</i> Samples from <i>num1</i> to <i>num2</i>, sample <i>num3</i>, and samples from <i>num4</i> to <i>num5</i> are executed (parentheses are optional).</li> </ul>
np	Number of points per decade or per octave or just number of points, based on which keyword precedes it.
OPTIMIZE	Specifies the parameter reference name, used for optimization in the <code>.PARAM</code> command
RESULTS	Measure name used for optimization in the <code>.MEASURE</code> command
start1 ...	Starting voltage, current, element, or model parameters; or temperature values. If you use the POI (list of points) variation type, specify a list of parameter values, instead of <i>start stop</i> . HSPICE supports the <i>start</i> and <i>stop</i> syntax; HSPICE RF does not.
stop1 ...	Final voltage, current, any element, model parameter, or temperature values.
SWEEP	Second sweep has a different type of variation (DEC, OCT, LIN, POI, or DATA command; or MONTE= <i>val</i> ).
TEMP	Temperature sweep.
type	Can be any of the following keywords: <ul style="list-style-type: none"> <li>▪ DEC — decade variation</li> <li>▪ OCT — octave variation</li> <li>▪ LIN — linear variation</li> <li>▪ POI — list of points</li> </ul>

Argument	Description
<i>var1</i> ...	<ul style="list-style-type: none"> <li>▪ Name of an independent voltage or current source, or</li> <li>▪ Name of any element or model parameter, or</li> <li>▪ TEMP keyword (indicating a temperature sweep).</li> </ul> <p>HSPICE supports a source value sweep, which refers to the source name (SPICE style). However, if you select a parameter sweep, a <code>.DATA</code> command, and a temperature sweep, then you must select a parameter name for the source value. A later <code>.DC</code> command must refer to this name. The parameter must not start with the TEMP keyword. The <i>var1</i> parameter should be defined in advance using the <code>.PARAM</code> command.</p>
<i>firstrun</i>	The <i>val</i> value specifies the number of Monte Carlo iterations to perform. The <i>firstrun</i> value specifies the desired number of iterations. HSPICE runs from <i>num1</i> to <i>num1+val-1</i> .
<i>list</i>	The iterations at which HSPICE performs a Monte Carlo analysis. You can write more than one number after <i>list</i> . The colon represents "from ... to ...". Specifying only one number makes HSPICE run at only the specified point.

### Description

You can use the `.DC` command in DC analysis to:

- Sweep any parameter value.
- Sweep any source value.
- Sweep temperature range.
- Perform a DC Monte Carlo (random sweep) analysis.
- Perform a data-driven sweep.
- Perform a DC circuit optimization for a data-driven sweep.
- Perform a DC circuit optimization by using start and stop.
- Perform a DC model characterization.

The format for the `.DC` command depends on the application that uses it. The DC sweep functionality is enhanced by use of the GSHUNT algorithm.

#### Examples

*Example 1 Sweeps the value of the VIN voltage source from 0.25 volts to 5.0 volts in increments of 0.25 volts.*

```
.DC VIN 0.25 5.0 0.25
```

*Example 2 Sweeps the drain-to-source voltage from 0 to 10 v in 0.5 v increments at VGS values of 0, 1, 2, 3, 4, and 5 v.*

```
.DC VDS 0 10 0.5 VGS 0 5 1
```

*Example 3 Starts a DC analysis of the circuit from -55°C to 125°C in 10°C increments.*

```
.DC TEMP -55 125 10
```

*Example 4 Script runs a DC analysis at five temperatures: 0, 30, 50, 100, and 125 °C.*

```
.DC XVAL 1K 10K .5K SWEEP TEMP LIN 5 25 125
```

*Example 5 Runs a DC analysis on the circuit at each temperature value. The temperatures result from a linear temperature sweep from 25°C to 125°C (five points), which sweeps a resistor value named xval from 1 k to 10 k in 0.5 k increments.*

```
.DC XVAL 1K 10K .5K SWEEP TEMP LIN 5 25 125
```

*Example 6 Specifies a sweep of the par1 value from 1 k to 100 k in increments of 10 points per decade.*

```
.DC DATA=DATANM SWEEP PAR1 DEC 10 1K 100K
```

*Example 7 Requests a DC analysis at specified parameters in the .DATA DATANM command. It also sweeps the par1 parameter from 1k to 100k in increments of 10 points per decade.*

```
.DC PAR1 DEC 10 1K 100K SWEEP DATA=DATANM
```

*Example 8 Invokes a DC sweep of the par1 parameter from 1k to 100k by 10 points per decade by using 30 randomly generated (Monte Carlo) values.*

```
.DC PAR1 DEC 10 1K 100K SWEEP MONTE=30
```

**Example 9** *Schmitt Trigger script.*

```

*file: bjtschmt.sp    bipolar schmitt trigger
.OPTION post=2
vcc 6 0 dc 12
vin 1 0 dc 0 pwl(0,0 2.5u,12 5u,0)
cb1 2 4 .1pf
rc1 6 2 1k
rc2 6 5 1k
rb1 2 4 5.6k
rb2 4 0 4.7k
re 3 0 .47k
diode 0 1 dmod
q1 2 1 3 bmod 1 ic=0,8
q2 5 4 3 bmod 1 ic=.5,0.2
.dc vin 0,12,.1
.model dmod d is=1e-15 rs=10
.model bmod npn is=1e-15 bf=80 tf=1n
+ cjc=2pf cje=1pf rc=50 rb=100 vaf=200
.probe v(1) v(5)
.print
.end

```

**Example 10** *Invokes a DC sweep of the par1 parameter from 1k to 100k by 10 points per decade and uses 10 Monte Carlo values from 11th to 20th trials.*

```
.DC par1 DEC 10 1k 100k SWEEP MONTE=list(10 20:30 35:40 50)
```

**Example 11** *Invokes a DC sweep of the par1 parameter from 1k to 100k by 10 points per decade and a Monte Carlo analysis at the 10th trial, then from the 20th to the 30th trials, followed by the 35th to 40th trials and finally at the 50th trial.*

```
.DC par1 DEC 10 1k 100k SWEEP MONTE=list(10 20:30 35:40 50)
```

**See Also**[.MODEL](#)[.OPTION DCIC](#)[.PARAM \(or\) .PARAMETER \(or\) .PARAMETERS](#)[Behavioral Application Examples](#) for the path to the demo file[inv\\_vin\\_vout.sp](#)

---

## .DCMATCH

Calculates the effects of variations on a circuit's DC characteristics.

### Syntax

```
.DCMATCH OUTVAR [THRESHOLD=T] [FILE=string] [INTERVAL=Int]
```

---

Argument	Description
OUTVAR	One or more node voltages, voltage differences for a node pair, or currents through an independent voltage source or currents through a resistor, a capacitor, or an inductor.
THRESHOLD	Report devices with a relative contribution above Threshold in the summary table. <ul style="list-style-type: none"><li>▪ T=0: reports results for all devices</li><li>▪ T&lt;0: suppresses table output; however, individual results are still available through .PROBE or .MEASURE commands.</li></ul> The upper limit for T is 1, but at least 10 devices are reported or all if there are less than 10. Default value is 0.01.
FILE	Valid file name for the output tables. Default is <code>basename.dm#</code> where “#” is the usual sequence number for HSPICE output files.
INTERVAL	Applies only if a DC sweep is specified. <code>Int</code> is a positive integer. A summary is printed at the first sweep point, then for each subsequent increment of <code>Int</code> and then if not already printed at the final sweep point. Only single sweeps are supported.

---

### Description

Use this command to calculate the effects of variations in device characteristics on the DC solution of a circuit.

You can perform only one DCMATCH analysis per simulation. Only the last .DCMATCH command is used in case more than one is present. The others are discarded with warnings.

### Examples

*Example 1* HSPICE reports DCmatch variations on the voltage of node 9, the voltage difference between nodes 4 and 2, and on the current through the source VCC and on the current through resistor x1.r1.

```
.DCMatch V(9) V(4,2) I(VCC) I(x1.r1)
```

*Example 2* The variable XVal is being swept in the .DC command. It takes nine values in sequence from 1k to 9k in increments of 1k. Tabular output for the .DCMATCH command is only generated for the set XVal={1k, 4k, 7k, 9k}.

```
.DC XVal Start=1K Stop=9K Step=1K  
.DCMATCH V(vcc) interval=3
```

**See Also**

[.DC](#)  
[.MEASURE \(DCMATCH\)](#)  
[.PROBE](#)  
[DCMatch Analysis](#)

---

## .DCSENS

Invokes DC sensitivity analysis using variation definitions as specified in the Variation Block.

### Syntax

```
.DCSENS Output_Variable [File=string] [Perturbation=x]
+ [Interval=SweepValue] [Threshold=x] [GroupByDevice=0|1]
```

---

Argument	Description
Output_Variable	Response with regard to the parameters designated in Sensitivity Block. Similar to the .DCMATCH command, the Output_Variable can be node voltage or branch current in a circuit.
File=string	Valid file name for the output tables. Default= <i>basename.ds#</i> where “#” is a number in the style of ds0, ds1, etc. If multiple dcsweep commands are specified in the netlist, then sensitivity analysis table results for each dcsweep are listed in *.ds# files. If .OPTION OPFILE specified, sensitivity result tables on operating points are listed in *.dp# files, otherwise, these tables are listed in the *.lis file.
Perturbation=x	Perturbations of x standard deviation are used in computing the finite difference approximations to device derivatives. The valid range for the parameter is 0.0001 to 1.0 with a default value of 0.05.
Interval=SweepValue	This option only applies to one dimensional sweeps. The SweepValue fields are positive integers. A summary is printed at the first sweep point, then for each subsequent increment of SweepValue, and then, if not already printed, at the final sweep point. The Interval key is ignored with a warning if a sweep is not being carried out.  The option only controls the printed summary table. The analysis may be carried out at additional sweep values if required by other forms of output such as Probe and Measure statements.
Threshold=x	Only devices with absolute sensitivity value above x are reported. Results for all devices are displayed if Threshold=0 is set. Default=10u.
GroupByDevice = 0 1	Alternate mode of generating sensitivity result tables; Default=0

---



## Description

Use this command to calculate the parameter sensitivity in the following instances:

- Global variation
- Local variation
- Local element variation with model type and model parameters that are permitted for DCmatch, including subckt variation

The methodology is based on using a finite difference approximation algorithm. DC sensitivity analysis combines the device derivatives, the DC solution, and the adjoint variables to get the sensitivity. DC sensitivity analysis enables you to compute sensitivity of any model parameter and many more models than traditional HSPICE sensitivity analysis. In addition, the analysis supports sensitivity for Probe and Measure output statements and for DC sweeps.

**Note:** .DCSENS does not support spatial variation and global element variation.

## Examples

In the following example, the variable XVal is being swept in the DC command. It takes nine values in sequence from 1K to 9K in increments of 1K. Tabular output for the sensitivity command is only generated for the set XVal={1K, 4K, 7K, 9K}.

```
.DC XVal Start=1K Stop=9K Step=1K  
.DCsens V(vcc) Interval=3
```

## See Also

[.OPTION OPFILE](#)  
[DC Sensitivity Analysis and Variation Block](#)

---

## .DCVOLT

Sets initial conditions in HSPICE.

### Syntax

```
.DCVOLT V(node1)=val1 V(node2)=val2 ...
.DCVOLT V node1val1 [node2val2 ...]
```

---

Argument	Description
<i>val1</i> ...	Voltages. The significance of these voltages depends on whether you specify the UIC parameter in the .TRAN command.
<i>node1</i> ...	Node numbers or names can include full paths or circuit numbers.

---

### Description

Use the .IC command or the .DCVOLT command to set transient initial conditions in HSPICE. How it initializes depends on whether the .TRAN analysis command includes the UIC parameter.

If you specify the UIC parameter in the .TRAN command, HSPICE does not calculate the initial DC operating point but directly enters transient analysis. Transient analysis uses the .IC initialization values as part of the solution for timepoint zero (calculating the zero timepoint applies a fixed equivalent voltage source). The .IC command is equivalent to specifying the IC parameter on each element command but is more convenient. You can still specify the IC parameter, but it does not take precedence over values set in the .IC command.

If you do *not* specify the UIC parameter in the .TRAN command, HSPICE computes the DC operating point solution before the transient analysis. The node voltages that you specify in the .IC command are fixed to determine the DC operating point. Transient analysis releases the initialized nodes to calculate the second and later time points.

### Examples

```
.DCVOLT 11 5 4 -5 2 2.2
```

### See Also

[.IC](#)  
[.TRAN](#)

---

## .DEL LIB

Removes library data from memory for HSPICE/HSPICE RF.

### Syntax

```
.DEL LIB '[file_path]file_name' entry_name
.DEL LIB libnumberentryname
```

---

Argument	Description
entry_name	Name of entry used in the library call command to delete.
file_name	Name of a file to delete from the data file; the file path, plus the file name, can be up to 256 characters long. You can use any file name that is valid for the operating system that you use. Enclose the file path and file name in single or double quotation marks.
file_path	Path name of a file if the operating system supports tree-structured directories.
libnumber	Library number, used in the library call command to delete.

---

### Description

Use this command to remove library data from memory. The next time you run a simulation, the `.DEL LIB` command removes the `.LIB` call command with the same library number and entry name from memory. You can then use a `.LIB` command to replace the deleted library. In this way, `.DEL LIB` helps you avoid name conflicts.

You can use the `.DEL LIB` command with the `.ALTER` command.

### Examples

#### Example 1

Example 1 calculates a DC transfer function for a CMOS inverter using these steps:

1. HSPICE simulates the device by using the `NORMAL` inverter model from the `MOS.LIB` library.
2. Using the `.ALTER` block and the `.LIB` command, HSPICE substitutes a faster CMOS inverter, `FAST` for `NORMAL`.
3. HSPICE then resimulates the circuit.

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

.DEL LIB

4. Using the second .ALTER block, HSPICE executes DC transfer analysis simulations at three different temperatures and with an n-channel width of 100 mm, instead of 15 mm.
5. HSPICE also runs a transient analysis in the second .ALTER block and uses a .MEASURE command to measure the rise time of the inverter.

```
FILE1: ALTER1 TEST CMOS INVERTER
.OPTION ACCT LIST
.TEMP 125
.PARAM WVAL=15U VDD=5
*
.OP
.DC VIN 0 5 0.1
.PRINT DC V(3) V(2)
*
VDD 1 0 VDD
VIN 2 0
*
M1 3 2 1 1 P 6U 15U
M2 3 2 0 0 N 6U W=WVAL
*
.LIB 'MOS.LIB' NORMAL
.ALTER
  .DEL LIB 'MOS.LIB' NORMAL $removes LIB from memory
  .DEL LIB 'MOS.LIB' NORMAL $removes normal library from memory
  .OPTION BRIEF=1 $suppress printing of details
  .LIB 'MOS.LIB' FAST $get fast model library
  .OPTION BRIEF=0 $resume normal printing
.ALTER
  .OPTION NOMOD OPTS $suppress printing model
  $parameters and print the
  $option summary
  .TEMP -50 0 50 $run with different temperatures
  .PARAM WVAL=100U VDD=5.5 $change the parameters using
  VDD 1 0 5.5 $VDD 1 0 5.5 to change the power
  $supply VDD value doesn't work
  VIN 2 0 PWL 0NS 0 2NS 5 4NS 0 5NS 5
  $change the input source
  .OP VOL $node voltage table of
  $operating points
  .TRAN 1NS 5NS $run with transient also
  M2 3 2 0 0 N 6U WVAL $change channel width
  .MEAS SW2 TRIG V(3) VAL=2.5 RISE=1 TARG V(3)
  + VAL=VDD CROSS=2 $measure output
  *
.END
```

*Example 2* The `.ALTER` block adds a resistor and capacitor network to the circuit. The network connects to the output of the inverter and HSPICE simulates a DC small-signal transfer function.

```
FILE2: ALTER2.SP CMOS INVERTER USING SUBCIRCUIT
.OPTION LIST ACCT
.MACRO INV 1 2 3
M1 3 2 1 1 P 6U 15U
M2 3 2 0 0 N 6U 8U
.LIB 'MOS.LIB' NORMAL
.EOM INV
XINV 1 2 3 INV
VDD 1 0 5
VIN 2 0
.DC VIN 0 5 0. 1
.PRINT V(3) V(2)
.ALTER
.DEL LIB 'MOS.LIB' NORMAL
.TF V(3) VIN          $DC small-signal transfer
    $function
*
.MACRO INV 1 2 3          $change data within
    $subcircuit def
M1 4 2 1 1 P 100U 100U    $change channel length,width,also
                          $topology
M2 4 2 0 0 N 6U 8U      $change topology
R4 4 3 100              $add the new element
C3 3 0 10P              $add the new element
.LIB 'MOS.LIB' SLOW      $set slow model library
$.INC 'MOS2.DAT'        $not allowed to be used
                          $inside subcircuit, allowed
                          $outside subcircuit

.EOM INV
.END
```

**See Also**

[.ALTER](#)  
[.LIB](#)

---

## .DESIGN\_EXPLORATION

Creates an Exploration Block to extract the parameters suitable for exploration from a netlist.

### Syntax

```
.Design_Exploration
Options
    Parameter Parameter_Name = value
    Parameter Parameter_Name = expression
.Data BlockName
    Index Name Name, ...
...
.EndData
.End_Design_Exploration
```

---

Argument Option	Description
Option Explore_only Subckts= SubcktList	This command is executed hierarchically—the specified subcircuits and all instantiated subcircuits and elements underneath are affected. Thus, if an inverter with name INV1 is placed in a digital control block called DIGITAL and in an analog block ANALOG, and Option Explore_only Subckts = ANALOG, then the perturbations only affect the INV1 in the analog block. You must create a new inverter INV1analog, with the new device sizes.
Option Do_not_explore Subckts= SubcktList	Excludes listed subcircuits.
Option Export=yes	Exports extraction data and runs one simulation with the original netlist
Option Export=no	(Default) Runs a simulation with Exploration data
Option Exploration_method= external Block_name= Block_name	The Block_name is the same as the name specified in the .DATA block; HSPICE will sweep the row content with the EXCommand (see <a href="#">Executing Exploration in HSPICE</a> ).
Option Ignore_exploration= yes no	(Default=no) HSPICE ignores the content in the design_exploration block, when Ignore_exploration=yes.

Argument Option	Description
Option Secondary_param= yes no	(Default = no) If Secondary_param= yes, HSPICE exports the MOSFET secondary instance parameters to a *.mex file (created when option export=yes), and also permits the secondary parameters to be imported as a column header in the .DATA block (option export=no).

### Description

Use the command to create an exploration block to extract prearrangers from a netlist to explore in the early stages of designing integrated circuits in CMOS technology.

Exploration is currently supported for:

- Independent sources: DC value
- MOS devices: W, L, M, dtemp
- Resistors: R or W, L, M, dtemp
- Capacitors: C or W, L, M, dtemp

When designing circuits, the multiplicity factor *M* is always a positive integer, but the Exploration tool can request arbitrary positive values.

To preserve relationships which have been previously defined through expressions, exploration can only be applied to parameters which are defined with numerical values.

The Export and non-export modes of exploration are distinguished by setting `Export` either `yes` or `no`.

The perturbation types are selected by setting any of the last three option listed in the Argument section.

For a detailed description of the Exploration Block usage, see [Exploration Block](#) in the *HSPICE User Guide: Simulation and Analysis*.

---

## .DISTO

Computes the distortion characteristics of the circuit in an AC analysis.

### Syntax

```
.DISTO Rload [inter [skw2 [refpwr spwf]]]
```

---

Argument	Description
Rload	Resistor element name of the output load resistor into which the output power feeds.
inter	Interval at which HSPICE prints a distortion-measure summary. Specifies a number of frequency points in the AC sweep (see the <i>np</i> parameter in the <code>.AC</code> command). <ul style="list-style-type: none"><li>▪ If you omit <i>inter</i> or set it to zero, HSPICE does not print a summary. To print or plot the distortion measures, use the <code>.PRINT</code> command.</li><li>▪ If you set <i>inter</i> to 1 or higher, HSPICE prints a summary of the first frequency and of each subsequent inter-frequency increment. To obtain a summary printout for only the first and last frequencies, set <i>inter</i> equal to the total number of increments needed to reach <i>fstop</i> in the <code>.AC</code> command. For a summary printout of only the first frequency, set <i>inter</i> to greater than the total number of increments required to reach <i>fstop</i>.</li></ul> HSPICE prints an extensive summary from the distortion analysis for each frequency listed. Use the <i>inter</i> parameter in the <code>.DISTO</code> command to limit the amount of output generated.
skw2	Ratio of the second frequency (F2) to the nominal analysis frequency (F1) in the range $1e-3 < skw2 < 0.999$ . If you omit <i>skw2</i> , the default value is 0.9.
refpwr	Reference power level—used to compute the distortion products. If you omit <i>refpwr</i> , the default value is 1mW—measured in decibels magnitude (dbM). The value must be $\geq 1e-10$ .
spwf	Amplitude of the second frequency (F2). The value must be $\geq 1e-3$ . The default is 1.0.

---

### Description

Use the `.DISTO` command to calculate the distortion characteristics of the circuit in an AC small-signal, sinusoidal, steady-state analysis. The program



computes and reports five distortion measures at the specified load resistor. The analysis assumes that the input uses one or two signal frequencies.

- HSPICE uses the first frequency (F1, the nominal analysis frequency) to calculate harmonic distortion. The `.AC` command frequency-sweep sets it.
- HSPICE uses the optional second input frequency (F2) to calculate intermodulation distortion. To set it implicitly, specify the `skw2` parameter, which is the F2/F1 ratio

HSPICE performs only one distortion analysis per simulation. If your design contains more than one `.DISTO` command, HSPICE runs only the last command. The `.DISTO` command calculates distortions for diodes, BJTs (levels 1, 2, 3, and 4), and MOSFETs (Level49 and Level53, Version 3.22). You can use the `.DISTO` command only with the `.AC` command.

### Examples

```
.DISTO RL 2 0.95 1.0E-3 0.75
```

### See Also

[.AC](#)

---

## .DOUT

Specifies the expected final state of an output signal.

### Syntax

```
.DOUT nd VTH (time state [time state])
.DOUT nd VLO VHI (time state [timestate])
```

---

Argument	Description
nd	Node name
time	Absolute time point (maximum 60)
state	Expected condition of the nd node at the specified <i>time</i> : <ul style="list-style-type: none"> <li>▪ 0: Expect ZERO,LOW.</li> <li>▪ 1: Expect ONE,HIGH.</li> <li>▪ Else: Do not care.</li> </ul>
VTH	Single voltage threshold
VLO	Voltage of the logic-low state
VHI	Voltage of the logic-high state

---

### Description

Use `.DOUT` to specify the expected final state of an output signal. During simulation, HSPICE compares simulation results with the expected output. If the states are different, an error report results.

For both syntax cases, the *time*, *state* pair describes the expected output. During simulation, the simulated results are compared against the expected output vector.

`.DOUT` State values are 0, 1, X, x, U, u, Z, z. Legal values for *state* are:

- 0: Expect zero
- 1: Expect one
- X, x: Do not care
- U, u: Do not care
- Z, z: Expect high impedance (do not care)

In addition, HSPICE supports multiple nodes in the .DOUT statement. This enables you to verify signals at the same time point in a single .DOUT statement.

### Examples

*Example 1* The .PARAM command in this example sets the VTH variable value to 3. The .DOUT command, operating on the node1 node, uses VTH as its threshold voltage.

*When node1 is above 3V, it is a logic 1; otherwise, it is a logic 0.*

*At 0ns, the expected state of node1 is logic-low.*

*At 2ns, 3ns, and 4ns, the expected state is “do not care.”*

*At 5ns, the expected state is again logic low.*

```
.PARAM VTH=3.0
.DOUT node1 VTH(0.0n 0 1.0n 1
+ 2.0n X 3.0n U 4.0n Z 5.0n 0)
```

*Example 2* Multiple nodes: verifying signals at the same time point

```
.DOUT B C D (0n 1 1 0 5n 0 0 0)
```

### See Also

[.MEASURE \(or\) .MEAS](#)

[.PARAM \(or\) .PARAMETER \(or\) .PARAMETERS](#)

[.PRINT](#)

[.PROBE](#)

[.STIM](#)

## .EBD

Invokes IBIS Electronic Board Description (EBD) functionality.

### Syntax

```
.EBD ebdname
+ file = 'filename'
+ component = 'compname:reference_designator'
+ {component = 'compname:reference_designator' ...}
+ {usemap = package_value}
```

Argument	Description
compname	Name after the .IBIS command that describes a component.
reference_designator	Reference designator that maps the component.
package_value	Value=0,1, 2, or 3 sets the package value (the same as option 'package' of .IBIS) of all components in [Reference Designator Map]. Default=0.

### Description

Enter the .EBD command to use the IBIS EBD feature. HSPICE uses the EBD file when simulating the line connected with the reference\_designator. When the keyword 'usemap' is added to the .EDB command, new components are added into the circuit according to the [Reference Designator Map]. The new component names are: 'Comp'+referenceName+'\_'+ebdName

In [Figure 7](#), CompU22\_ebd and CompU23\_ebd are added if U22 and U23 occur in [Reference Designator Map].

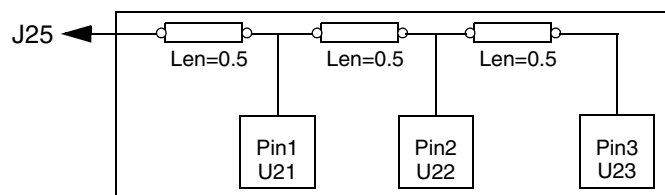


Figure 7 Circuit Connection for EBD Example

If a component is associated with both the keywords `component` and `usemap`, then the mapping relation defined by `component` only is used. The format of the node name on the EBD side is `ebdName_pinName`. For example, the name **J25** is `ebd_J25`

**Note:** If a component pin is not found and it is not a terminal node in the EBD path, then the name is used to designate the related node. For example, in [Figure 7 on page 90](#), if U22\_2 (here, 2 is the pin name) does not exist, then the node name will be ebd\_U22\_2.

If the component pin is a terminal node in the EBD path and is not found, then the node and the associated section will not be added into circuit. For example, in [Figure 7](#), if U23\_3 does not exist, then the section between Pin2 and Pin3 will be ignored and U22\_2 is the terminal node.

### Examples

```
.ebd ebd
+ file = 'test.ebd'
+ model = '16Meg X 8 SIMM Module'
+ component = 'cmpnt:u21'
* + usemap = 0
.ibis cmpnt
+ file = 'ebd.ibs'
+ component = 'SIMM'
+ hsp_ver=2003.09 nowarn
```

This example corresponds to the following *.ebd* file:

```
.....
[Begin Board Description] 16Meg X 8 SIMM Module
.....
[Pin List] signal_name
J25 POWER5
[Path Description] CAS_2
Pin J25
Len=0.5 L=8.35n C=3.34p R=0.01 /
Node u21.1
Len=0.5 L=8.35n C=3.34p R=0.01 /
Node u22.2
Len=0.5 L=8.35n C=3.34p R=0.01 /
Node u23.3
```

### See Also

[.IBIS](#)

[.PKG](#)

[IBIS Examples](#) and see `.EBD` command use in *ebd.sp* and *pinmap.sp*

---

## **.ELSE**

Precedes commands to be executed in a conditional block when preceding `.IF` and `.ELSEIF` conditions are false.

### **Syntax**

```
.ELSE
```

### **Description**

Use this command to precede one or more commands in a conditional block after the last `.ELSEIF` command, but before the `.ENDIF` command.

HSPICE/HSPICE RF executes these commands by default if the conditions are all false in the preceding `.IF` command and in all of the preceding `.ELSEIF` commands in the same conditional block.

For the syntax and a description of how to use the `.ELSE` command within the context of a conditional block, see the `.IF` command.

For information on use of conditional blocks with the Exploration Block, see, [Specifying Constraints](#) in the *HSPICE User Guide: Simulation and Analysis*.

### **See Also**

[.ELSEIF](#)

[.ENDIF](#)

[.IF](#)

## **.ELSEIF**

Specifies conditions that determine whether HSPICE/HSPICE RF executes subsequent commands in a conditional block.

### **Syntax**

```
.ELSEIF (condition)
```

### **Description**

HSPICE executes the commands that follow the first .ELSEIF command only if *condition1* in the preceding .IF command is false and *condition2* in the first .ELSEIF command is true.

If *condition1* in the .IF command and *condition2* in the first .ELSEIF command are both false, then HSPICE moves on to the next .ELSEIF command if there is one.

If this second .ELSEIF condition is true, HSPICE executes the commands that follow the second .ELSEIF command, instead of the commands after the first .ELSEIF command.

HSPICE ignores the commands in all false .IF and .ELSEIF commands, until it reaches the first .ELSEIF condition that is true. If no .IF or .ELSEIF condition is true, HSPICE continues to the .ELSE command.

For the syntax and a description of how to use the .ELSEIF command within the context of a conditional block, see the .IF command.

For information on use of conditional blocks with the Exploration Block, see, [Specifying Constraints](#) in the *HSPICE User Guide: Simulation and Analysis*.

### **See Also**

- [.ELSE](#)
- [.ENDIF](#)
- [.IF](#)

---

## .END

Ends a simulation run in an input netlist file.

### Syntax

```
.END [comment]
```

---

Argument	Description
comment	Can be any comment. Typically, the comment is the name of the netlist file or of the simulation run that this command terminates.

---

### Description

An `.END` command must be the last command in the input netlist file. The period preceding `END` is required. Text that follows the `.END` command is regarded as a comment only. An input file that contains more than one simulation run must include an `.END` command for each simulation run. You can concatenate several simulations into a single file.

### Examples

```
MOS OUTPUT
.OPTION NODE NOPAGE
VDS 3 0
VGS 2 0
M1 1 2 0 0 MOD1 L=4U W=6U AD=10P AS=10P
.MODEL MOD1 NMOS VTO=-2 NSUB=1.0E15 TOX=1000
+ UO=550
VIDS 3 1
.DC VDS 0 10 0.5 VGS 0 5 1
.PRINT DC I(M1) V(2)
.END MOS OUTPUT
MOS CAPS
.OPTION SCALE=1U SCALM=1U WL ACCT
.OP
.TRAN .1 6
V1 1 0 PWL 0 -1.5V 6 4.5V
V2 2 0 1.5VOLTS
MODN1 2 1 0 0 M 10 3
.MODEL M NMOS VTO=1 NSUB=1E15 TOX=1000
+ UO=800 LEVEL=1 CAPOP=2
.PRINT TRAN V(1) (0,5) LX18(M1) LX19(M1) LX20(M1)
+ (0,6E-13)
.END MOS CAPS
```



## **.ENDDATA**

Ends a `.DATA` block in an HSPICE input netlist file.

### **Syntax**

`.ENDDATA`

### **Description**

Use this command to terminate a `.DATA` block in an HSPICE input netlist.

### **See Also**

[.DATA](#)

---

## **.ENDIF**

Ends a conditional block of commands in an HSPICE input netlist file.

### **Syntax**

`.ENDIF`

### **Description**

Use this command to terminate a conditional block of commands that begins with an `.IF` command.

For the syntax and a description of how to use the `.ENDIF` command within the context of a conditional block, see the `.IF` command.

### **See Also**

[.ELSE](#)  
[.ELSEIF](#)  
[.IF](#)

## **.ENDL (or) .ENDL TT**

Ends a `.LIB` command in an HSPICE/HSPICE RF input netlist file.

### **Syntax**

```
.ENDL  
.ENDL TT
```

### **Description**

Use this command to terminate a `.LIB` command in an HSPICE input netlist.

### **Examples**

Either the `.ENDL` or `.ENDL TT` command is valid for ending a `.LIB` statement.

```
.lib tt  
.param vth=0.1  
.include 'model_tt.sp'  
.endl tt
```

or

```
.lib tt  
.param vth=0.1  
.include 'model_tt.sp'  
.endl
```

### **See Also**

[.LIB](#)

---

## .ENDS

Ends a subcircuit definition (.SUBCKT) in an HSPICE input netlist file.

### Syntax

```
.ENDS subckt_name
```

---

Argument	Description
subckt_name	Subcircuit name definition to end a command that begins with a .SUBCKT command.

---

### Description

Use this command to terminate a .SUBCKT command. This command must be the last for any subcircuit definition that starts with a .SUBCKT command. You can nest subcircuit references (calls) within subcircuits in HSPICE.

**Note:** Using `-top subckt_name` on the command line effectively eliminates the need for the `.subckt subckt_name` and `.ends subckt_name`

### Examples

*Example 1* Terminates a subcircuit named mos\_circuit.

```
.ENDS mos_circuit
```

*Example 2* Terminates all subcircuit definitions that begin with a .SUBCKT command.

```
.ENDS
```

### See Also

[.SUBCKT](#)

---

## .ENV

Performs standard envelope simulation in HSPICE RF.

### Syntax

```
.ENV TONES=f1 [f2...fn] NHARMS=h1 [h2...hn]  
+ ENV_STEP=tstep ENV_STOP=tstop
```

---

Argument	Description
TONES	Carrier frequencies, in hertz
NHARMS	Number of harmonics
ENV_STEP	Envelope step size, in seconds
ENV_STOP	Envelope stop time, in seconds

---

### Description

Use this command to perform standard envelope simulation.

The simulation proceeds just as it does in standard transient simulation, starting at `time=0` and continuing until `time=env_stop`. An HB analysis is performed at each step in time. You can use Backward-Euler (BE), trapezoidal (TRAP), or level-2 Gear (GEAR) integration.

- For BE integration, set `.OPTION SIM_ORDER=1`.
- For TRAP, set `.OPTION SIM_ORDER=2 (default) METHOD=TRAP (default)`.
- For GEAR, set `.OPTION SIM_ORDER=2 (default) METHOD=GEAR`.

### See Also

[.ENVOOSC](#)  
[.HB](#)  
[.PRINT](#)  
[.PROBE](#)

---

## .ENVFFT

Performs Fast Fourier Transform (FFT) on envelope output in HSPICE RF.

### Syntax

```
.ENVFFT output_var NP=value FORMAT=keyword  
+ WINDOW=keyword ALFA=value
```

---

Argument	Description
<i>output_var</i>	Any valid output variable.
NP	Number of points to use in the FFT analysis. <i>NP</i> must be a power of 2. If not a power of 2, then it is automatically adjusted to the closest higher number that is a power of 2. The default is 1024.
FORMAT	Output format: NORM= normalized magnitude UNORM=unnormalized magnitude (default)
WINDOW	Window type to use: <ul style="list-style-type: none"><li>▪ RECT=simple rectangular truncation window (default)</li><li>▪ BART=Bartlett (triangular) window</li><li>▪ HANN=Hanning window</li><li>▪ HAMM=Hamming window</li><li>▪ BLACK=Blackman window</li><li>▪ HARRIS=Blackman-Harris window</li><li>▪ GAUSS=Gaussian window</li><li>▪ KAISER=Kaiser-Bessel window</li></ul>
ALFA	Controls the highest side-lobe level and bandwidth for GAUSS and KAISER windows. The default is 3.0.

---

### Description

Use this command to perform Fast Fourier Transform (FFT) on envelope output. This command is similar to the `.FFT` command. In HSPICE RF the data being transformed is complex. You usually want to do this for a specific harmonic of a voltage, current, or power signal.

### See Also

[.ENV](#)  
[.ENVOOSC](#)  
[.FFT](#)

---

## .ENVOOSC

Performs envelope simulation for oscillator startup or shutdown in HSPICE RF.

### Syntax

```
.ENVOOSC TONE=f1 NHARMS=h1 ENV_STEP=tstep ENV_STOP=tstop
+ PROBENODE=n1,n2,vosc [FSPTS=num, min, max]
```

---

Argument	Description
TONES	Carrier frequencies, in hertz.
NHARMS	Number of harmonics.
ENV_STEP	Envelope step size, in seconds.
ENV_STOP	Envelope stop time, in seconds.
PROBENODE	Defines the nodes used for oscillator conditions and the initial probe voltage value.
FSPTS	Specifies the frequency search points used in the initial small-signal frequency search. Usage depends on oscillator type.

---

### Description

Use `.ENVOOSC` to perform envelope simulation for oscillator startup or shutdown. Oscillator startup or shutdown analysis must be helped along by converting a bias source from a DC description to a PWL description that either:

- Starts at a low value that supports oscillation and ramps up to a final value (startup simulation)
- Starts at the DC value and ramps down to zero (shutdown simulation).

In addition to computing the state variables at each envelope time point, the `.ENVOOSC` command also computes the frequency. This command is applied to high-Q oscillators that take a long time to reach steady-state. For these circuits, standard transient analysis is too costly. Low-Q oscillators, such as typical ring oscillators are more efficiently simulated with standard transient analysis.

### See Also

[.ENV](#)  
[.ENVFFT](#)

---

## .EOM

Ends a .MACRO command.

### Syntax

```
.EOM subckt_name
```

---

Argument	Description
subckt_name	Subcircuit name definition to end a macro that begins with a .SUBCKT command.

---

### Description

Use this command to terminate a .MACRO command. .EOM must be the last for any subcircuit definition that starts with a .MACRO command. You can nest subcircuit references (calls) within subcircuits.

### Examples

*Example 1* Terminates a subcircuit named diode\_circuit.

```
.EOM diode_circuit
```

*Example 2* If you omit the subcircuit name as in this second example, this command terminates all subcircuit definitions that begin with a .MACRO command.

```
.EOM
```

### See Also

[.MACRO](#)



## .FFT

Calculates the Discrete Fourier Transform (DFT) value used for spectrum analysis. Numerical parameters (excluding string parameters) can be passed to the .FFT command.

### Syntax

#### Syntax # 1 Alphanumeric input

```
.FFT output_var [START=value] [STOP=value]
+ NP=value [FORMAT=keyword]
+ [WINDOW=keyword] [ALFA=value]
+ [FREQ=value] [FMIN=value] [FMAX=value]
```

#### Syntax #2 Numerics and expressions

```
.FFT [output_var] [START=param_expr1] [STOP=param_expr2]
+ [NP=param_expr3] [FORMAT=keyword]
+ [WINDOW=keyword] [ALFA=param_expr4]
+ [FREQ=param_expr5] [FMIN=param_expr6] [FMAX=param_expr7]
```

#### Syntax # Verilog-A Blocks

```
.FFT VAblock:SigName StartIdx=n1 StartIdx=n2
+ SamplePeriod=val
+ ...
```

Argument	Description
output_var	Any valid output variable, such as voltage, current, or power.
START	Start of the output variable waveform to analyze. Defaults to the START value in the .TRAN command (tstart), which defaults to 0.
FROM	An alias for START in .FFT commands.
STOP	End of the output variable waveform to analyze. Defaults to the TSTOP value in the .TRAN command.
TO	An alias for STOP, in .FFT commands.
NP	Number of points to use in the FFT analysis. NP must be a power of 2. If NP is not a power of 2, HSPICE automatically adjusts it to the closest higher number that is a power of 2. The default is 1024.

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

.FFT

Argument	Description
FORMAT	Output format: <ul style="list-style-type: none"><li>▪ NORM= normalized magnitude (default)</li><li>▪ UNORM=unnormalized magnitude</li></ul>
WINDOW	Window can be one of the following types: <ul style="list-style-type: none"><li>▪ RECT=simple rectangular truncation window (default).</li><li>▪ BART=Bartlett (triangular) window.</li><li>▪ HANN=Hanning window.</li><li>▪ HAMM=Hamming window.</li><li>▪ BLACK=Blackman window.</li><li>▪ HARRIS=Blackman-Harris window.</li><li>▪ GAUSS=Gaussian window.</li><li>▪ KAISER=Kaiser-Bessel window.</li></ul>
ALFA	Parameter to use in GAUSS and KAISER windows to control the highest side-lobe level, bandwidth, and so on. $1.0 \leq \text{ALFA} \leq 20.0$ The default is 3.0
FREQ	Frequency to analyze. If FREQ is non-zero, the output lists only the harmonics of this frequency, based on FMIN and FMAX. HSPICE also prints the THD for these harmonics. The default is 1.0 (STOP-START) (Hz).
FMIN	Minimum frequency for which HSPICE prints FFT output into the listing file. THD calculations also use this frequency. $T = (\text{STOP} - \text{START})$ The default is $1.0/T$ (Hz).
FMAX	Maximum frequency for which HSPICE prints FFT output into the listing file. THD calculations also use this frequency. The default is $0.5 * \text{NP} * \text{FM IN}$ (Hz).
VAblock	Name of the Verilog-A block.
SigName	Parameter name of the series output from Verilog-A. It should have the following type definition in Verilog-A block: <pre>(* desc="SigName" *) real SigName [n1:n2];</pre>
StartIdx	Start index of the series for FFT.
StopIdx	End index of the series for FFT; it must be greater than StartIdx; otherwise, HSPICE uses the whole series for the FFT process.

Argument	Description
SamplePeriod	Time interval between two samples inside the series. It must be a positive value, the default value is 1 second.

### Description

Use this command to calculate the Discrete Fourier Transform (DFT) values for spectrum analysis. `.FFT` uses internal time point values to calculate these values. A DFT uses sequences of time values to determine the frequency content of analog signals in circuit simulation. You can pass numerical parameters/expressions (but no string parameters) to the `.FFT` command. Output variables for `.FFT` can be voltage, current, or power, followed by a parenthesis containing the instance name. If it is power, for example, you need to write the signal's name in the format `p(instance_name)`.

You can specify only one output variable in an `.FFT` command. The following is an *incorrect* use of the command because it contains two variables in one `.FFT` command:

For an `.FFT` analysis using a Verilog A-block, the FFT time window is:  
`TimeWindow = SamplePeriod*(stopidx-startidx)`

A FFT process requires sampling the waveform with equally spaced time points, and the total point number must be  $2^N$  (N: integer). Therefore, the start/stop time points, fundamental frequency, sampling rate, and total point number are not independent of each other. They need to satisfy the following relationship:

$$\frac{\text{point\_number}}{t_{\text{stop}} - t_{\text{start}}} = \text{sample\_rate}, \text{ where } \text{point\_number} = 2^N$$

$$F_{\text{fund}} = \frac{M}{t_{\text{stop}} - t_{\text{start}}}, \text{ where } M \text{ is an integer number}$$

If that relationship is compromised, conflicts between parameters may arise. To avoid such conflicts, HSPICE conducts an error check process according to the following:

Parameter	Check if input...	Adjust if input...	Set if not input...
START	Error if < tstart (start point in .TRAN)	N/A	=tstart (start point in .TRAN)

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .FFT

Parameter	Check if input...	Adjust if input...	Set if not input...
STOP	Error if > tstop (stop point in .TRAN)	N/A	=tstop (stop point in .TRAN)
NP	Error if NP < 4 Error if NP > 227	Is not a power of 2; adjust to nearest power of 2, issue warning and final value	Default value (1024)
FREQ	Error if $< \frac{1}{STOP - START}$	If not integer multiple of $1/(STOP-START)$ , adjust to nearest multiple of $1/(STOP-START)$ , issue warning and final value	$\frac{1}{STOP - START}$
SamplePeriod	Error if non-positive	Use default value: 1s	1 second
StartIdx	Error if $\geq$ StopIdx	N/A	Start index of the VA array
StopIdx	Error if $\leq$ StartIdx	N/A	Stop index of the VA array

An embedded `.FFT` command in a `measure_file` can be called to perform FFT measurements from previous simulation results as follows:

```
HSPICE -i *.tr0 -meas measure_file
```

### Examples

```
.FFT v(1)
.FFT v(1,2) np=1024 start=0.3m stop=0.5m freq=5.0k
+ window=kaiser alfa=2.5
.FFT I(rload) start=0m to=2.0m fmin=100k fmax=120k
+ format=unorm
.FFT par('v(1) + v(2)') from=0.2u stop=1.2u
+ window=harris
```

The example below generates an `.ft0` file for the FFT of `v(1)` and an `.ft1` file for the FFT of `v(2)`.

```
.FFT v(1) np=1024
.FFT v(2) np=1024
```

## See Also

[.TRAN](#)

[.MEASURE FFT](#)

[Spectrum Analysis](#)

[Fourier Analysis Examples](#) for demo files on window weighting including

- *gauss.sp*
- *hamm.sp*
- *hann.sp*
- *harris.sp*
- *kaiser.sp*
- *rect.sp*

[Fourier Analysis Examples](#), netlists demonstrating use of the `.FFT` command:

- *fft5.sp* (data-driven with transient analysis)
- *fft6.sp* and *sine.sp* (sine source)
- *intermod.sp* (intermodulation distortion)
- *mod.sp* (modulated pulse)
- *pulse.sp* (pulse source)
- *pwl.sp* (PWL source)
- *sffm.sp* (single-frequency FM source)
- *swcap5.sp* (fifth-order elliptic, switched-capacitor filter)

---

## .FLAT

Provides subcircuit OP back annotation when a device is modeled as a subckt.

### Syntax

If defined in subckt definition block:

```
.FLAT element_name
```

If defined in main circuit:

```
.FLAT subckt_nameelement_name
```

### Description

This command enables subcircuit OP back annotation when a device is modeled as a subckt.

**Note:** *subckt\_name* is the name appearing in a `.subckt` definition statement; *element\_name* is a simple element name which is defined in the same subckt definition block.

When a device is modeled as a subcircuit rather than as `.MODEL`, using the `.FLAT` command within a subcircuit allows the writing of a results file with proper values for the device. Back-annotation is done by retrieving results from the *input.op0* (for DC) and *input.op1* (for transient) results files. The `.FLAT` command works for *\*.wdf* format, *\*.psf*, and *\*.tr0* files.

---

This subckt file	...is equal to
<pre>.subckt nmos_sub d g s b m0 d g s b nmos 10u 10u r0 int_d d 100 .flat m0 .model nmos nmos level=49 .ends nmos_sub</pre>	<pre>.flat nmos_sub m0 .subckt nmos_sub d g s b m0 d g s b nmos 10u 10u r0 int_d d 100 .model nmos nmos level=49 .ends nmos_sub</pre>

---

If the `.FLAT` command is in both the subckt definition block and main circuit, the subckt block `.FLAT` takes priority. If more than one `.FLAT` is defined for the same subckt, the last one takes priority.

### Examples

```
.subckt nmossub D G S B l=l w=w  
M1 D_int G_int S_int B nch l=l w=w  
M2 D_int G_int S_int B nch l=l w=w  
RD D D_int 100  
RG G G_int 10  
RS S S_int 400  
.flat M1  
.ends nmossub  
  
X1 1 2 0 0 nmossub
```

---

## .FOUR

Performs a Fourier analysis as part of the transient analysis.

### Syntax

```
.FOUR freq ov1 [ov2 ov3 ...]
```

---

Argument	Description
freq	Fundamental frequency
ov1 ...	Output variables to analyze

---

### Description

Use this command to perform a Fourier analysis as part of the transient analysis. You can use this command in HSPICE to perform the Fourier analysis over the interval (tstop-fperiod, tstop), where:

- tstop is the final time, specified for the transient analysis.
- fperiod is a fundamental frequency period (freq parameter).

HSPICE performs Fourier analysis on 501 points of transient analysis data on the last 1/f time period, where f is the fundamental Fourier frequency. HSPICE interpolates transient data to fit on 501 points, running from (tstop-1/f) to tstop.

To calculate the phase, the normalized component and the Fourier component, HSPICE uses 10 frequency bins. The Fourier analysis determines the DC component and the first nine AC components. For improved accuracy, the .FOUR command can use non-linear, instead of linear interpolation.

You can use a .FOUR command only with a .TRAN command.

### Examples

```
.FOUR 100K V(5)
```

### See Also

[.TRAN](#)  
[.FFT](#)



## .FSOPTIONS

Sets various options for the HSPICE Field Solver.

### Syntax

```
.FSOPTIONS name [ACCURACY=LOW|MEDIUM|HIGH]
+ [GRIDFACTOR=val] [PRINTDATA=YES|NO]
+ [COMPUTE_GO=YES|NO] [COMPUTE_GD=YES|NO]
+ [COMPUTE_RO=YES|NO] [COMPUTE_RS=YES|NO]
+ [COMPUTE_RS=YES|NO|DIRECT|ITER]
+ [COMPUTE_TABLE=FREQUENCY_SWEEP]
+ [PRINTDATA=YES|NO|APPEND]
```

Argument	Description
<i>name</i>	Option name.
ACCURACY	Solver accuracy is one of the following: <ul style="list-style-type: none"> <li>▪ LOW</li> <li>▪ MEDIUM</li> <li>▪ HIGH</li> </ul>
GRIDFACTOR	Multiplication factor (integer) to determine the final number of segments used to define the shape. If you set COMPUTE_RS=yes, the field solver does not use this parameter to compute R <sub>o</sub> and R <sub>s</sub> values.
PRINTDATA	Prints output matrixes to a file.
COMPUTE_GO	Computes the static conductance matrix.
COMPUTE_GD	Computes the dielectric loss matrix.
COMPUTE_RO	Computes the DC resistance matrix.
COMPUTE_RS	Activates and chooses filament solver to compute R <sub>o</sub> and R <sub>s</sub> . The solver computes the skin-effect resistance matrix. <ul style="list-style-type: none"> <li>▪ YES: activate filament solver with direct matrix solver</li> <li>▪ NO: (Default) Does not perform filament solver</li> <li>▪ DIRECT: Activate filament solver with direct matrix solver (same as "YES")</li> <li>▪ ITER: Activates filament solver with iterative matrix solver</li> </ul>

Argument	Description
COMPUTE_TABLE	Specifies a type of frequency sweep for extracting RLGC Tabular Model. You can specify either LIN, DEC, OCT, POI. Specify the nsteps, start, and stop values using the following syntax for each type of sweep: <ul style="list-style-type: none"> <li>▪ LIN <i>nsteps start stop</i></li> <li>▪ DEC <i>nsteps start stop</i></li> <li>▪ OCT <i>nsteps start stop</i></li> <li>▪ POI <i>nsteps freq_values</i></li> </ul>
PRINTDATA	When PRINTDATA=APPEND, RLGC model output is appended to the specified output file.

### Description

Use the .FSOPTIONS command to set various options for the field solver. The following rules apply to the field solver when specifying options with the .FSOPTIONS command:

- The field solver always computes the L and C matrixes.
- If COMPUTE\_RS=YES, the field solver starts and calculates  $L_o$ ,  $R_o$ , and  $R_s$ .
- For each accuracy mode, the field solver uses either the predefined number of segments or the number of segments that you specified. It then multiplies this number times the GRIDFACTOR to obtain the final number of segments.

Because a wide range of applications are available, the predefined accuracy level might not be accurate enough for some applications. If you need a higher accuracy than the value that the HIGH option sets, then increase either the GRIDFACTOR value or the N, NH, or NW values to increase the mesh density. NW and NH quantities are used for rectangles and N is used for circles, polygons and strips. See the .SHAPE commands in this chapter for the complete syntax for each shape.

**Note:** The forms of the following arguments are interchangeable:

```
COMPUTE_GO : COMPUTEGO
COMPUTE_GD : COMPUTEGD
COMPUTE_RO : COMPUTERO
COMPUTE_RS : COMPUTERS
COMPUTE_TABLE : COMPUTETABLE
```

See the *HSPICE User Guide: Signal Integrity* for more information on [Extracting Transmission Line Parameters \(Field Solver\)](#).

### Examples

```
// LU solver
*.fsoptions printem printdata=yes compute_rs=direct
compute_gd=yes
// GMRES solver
.fsoptions printem printdata=yes compute_rs=iter compute_gd=yes
```

### See Also

[.LAYERSTACK](#)

[.MATERIAL](#)

[.SHAPE](#)

[Transmission \(W-element\) Line Examples](#)

---

## **.GLOBAL**

Globally assigns a node name.

### **Syntax**

```
.GLOBAL node1 node2 node3 ...
```

---

<b>Argument</b>	<b>Description</b>
node1, node2...	Name of a global nodes, such as supply and clock names; overrides local subcircuit definitions.

---

### **Description**

Use this command to globally assign a node name in HSPICE. This means that all references to a global node name, used at any level of the hierarchy in the circuit, connect to the same node.

The most common use of a `.GLOBAL` command is if your netlist file includes subcircuits. This command assigns a common node name to subcircuit nodes. Another common use of `.GLOBAL` commands is to assign power supply connections of all subcircuits. For example, `.GLOBALVCC` connects all subcircuits with the internal node name `VCC`.

Typically, in a subcircuit, the node name consists of the circuit number concatenated to the node name. When you use a `.GLOBAL` command, HSPICE does not concatenate the node name with the circuit number and assigns only the global name. You can then exclude the power node name in the subcircuit or macro call.

### **Examples**

This example shows global definitions for `VDD` and `input_sig` nodes.

```
.GLOBAL VDD input_sig
```

---

## .HB

Invokes the single and multitone harmonic balance algorithm for periodic steady state analysis.

### Syntax

#### *Syntax # 1 without SS\_TONE*

```
.HB TONES=F1 [F2 ... FN] [SUBHARMS=SH]  
+ [NHARMS=H1, H2 ... HN] [INTMODMAX=n]  
+ [SWEEP parameter_sweep]
```

#### *Syntax#2 with SS\_TONE*

```
.HB TONES=F1 [F2 ... FN] [SUBHARMS=SH]  
+ [NHARMS=H1, H2 ... HN] [INTMODMAX=n]  
+ [SS_TONE=n] [SWEEP parameter_sweep]
```

---

Argument	Description
TONES	Fundamental frequencies.
SUBHARMS	Subharmonics in the analysis spectrum. The minimum non-DC frequency in the analysis spectrum is f/subharms, where f is the frequency of oscillation.
NHARMS	Number of harmonics to use for each tone. Must have the same number of entries as TONES. You must specify NHARMS, INTMODMAX or both.
INTMODMAX	Maximum intermodulation product order that you can specify in the analysis spectrum. You must specify NHARMS, INTMODMAX or both.
SWEEP	Type of sweep. You can sweep up to three variables. You can specify either LIN, DEC, OCT, POI, SWEEPBLOCK, DATA, OPTIMIZE or MONTE. Specify the nsteps, start, and stop times using the following syntax for each type of sweep: <ul style="list-style-type: none"><li>▪ LIN <i>nsteps start stop</i></li><li>▪ DEC <i>nsteps start stop</i></li><li>▪ OCT <i>nsteps start stop</i></li><li>▪ POI <i>nsteps freq_values</i></li><li>▪ SWEEPBLOCK <i>nsteps freq1 freq2 ... freqn</i></li><li>▪ DATA=<i>dataname</i></li><li>▪ OPTIMIZE=OPT<i>xxx</i></li><li>▪ MONTE=<i>val</i></li></ul>
SS_TONE	Small-signal tone number for HBLIN analysis. The value must be an integer number. The default value is 0, indicating that no small signal tone is specified.

---

### Description

Use this command to invoke the single and multitone harmonic balance algorithm for periodic steady state analysis.

The NHARMS and INTMODMAX input parameters define the spectrum.

- If INTMODMAX=N, the spectrum consists of all  $f = a*f_1 + b*f_2 + \dots + n*f_n$  frequencies so that  $f \geq 0$  and  $|a| + |b| + \dots + |n| \leq N$ . The a,b,...,n coefficients are integers with absolute value  $\leq N$ .
- If INTMODMAX is not specified, HSPICE RF defaults it to the largest value in the NHARMS list.
- If entries in the NHARMS list are  $> INTMODMAX$ , HSPICE RF adds the  $m*f_k$  frequencies to the spectrum, where  $f_k$  is the corresponding tone, and m is a value  $\leq$  the NHARMS entry.

For detailed discussion of HBLIN analysis, see [Frequency Translation S-Parameter \(HBLIN\) Extraction](#) in the *HSPICE User Guide: RF Analysis*.

### Examples

In Example 1, the resulting HB analysis spectrum={dc,  $f_1$ ,  $f_2$ }.

#### Example 1

```
.hb tones=f1, f2 intmodmax=1
```

In Example 2, the HB analysis spectrum={dc,  $f_1$ ,  $f_2$ ,  $f_1+f_2$ ,  $f_1-f_2$ ,  $2*f_1$ ,  $2*f_2$ }.

#### Example 2

```
.hb tones=f1, f2 intmodmax=2
```

In Example 3, the resulting HB analysis spectrum={dc,  $f_1$ ,  $f_2$ ,  $f_1+f_2$ ,  $f_1-f_2$ ,  $2*f_1$ ,  $2*f_2$ ,  $2*f_1+f_2$ ,  $2*f_1-f_2$ ,  $2*f_2+f_1$ ,  $2*f_2-f_1$ ,  $3*f_1$ ,  $3*f_2$ }.

#### Example 3

```
.hb tones=f1, f2 intmodmax=3
```

In Example 4, the resulting HB analysis spectrum={dc,  $f_1$ ,  $f_2$ ,  $f_1+f_2$ ,  $f_1-f_2$ ,  $2*f_1$ ,  $2*f_2$ }.

#### Example 4

```
.hb tones=f1, f2 nharms=2,2
```

In Example 5, the resulting HB analysis spectrum={dc,  $f_1$ ,  $f_2$ ,  $f_1+f_2$ ,  $f_1-f_2$ ,  $2*f_1$ ,  $2*f_2$ ,  $2*f_1-f_2$ ,  $2*f_1+f_2$ ,  $2*f_2-f_1$ ,  $2*f_2+f_1$ }.

*Example 5*

```
hb tones=f1, f2 nharms=2,2 intmodmax=3
```

The resulting HB analysis spectrum={dc,  $f_1$ ,  $f_2$ ,  $f_1+f_2$ ,  $f_1-f_2$ ,  $2*f_1$ ,  $2*f_2$ ,  $2*f_1-f_2$ ,  $2*f_1+f_2$ ,  $2*f_2-f_1$ ,  $2*f_2+f_1$ ,  $3*f_1$ ,  $3*f_2$ ,  $4*f_1$ ,  $4*f_2$ ,  $5*f_1$ ,  $5*f_2$ }.

*Example 6*

```
.hb tones=f1, f2 nharms=5,5 intmodmax=3
```

**See Also**

- .ENV
- .HBAC
- .HBLIN
- .HBNOISE
- .HBOSC
- .OPTION HBCONTINUE
- .OPTION HBJREUSE
- .OPTION HBJREUSETOL
- .OPTION HBACKRYLOVDIM
- .OPTION HBKRYLOVTOL
- .OPTION HBLINESEARCHFAC
- .OPTION HBMAXITER (or) HB\_MAXITER
- .OPTION HBSOLVER
- .OPTION HBTOL
- .OPTION LOADHB
- .OPTION SAVEHB
- .OPTION TRANFORHB
- .PRINT
- .PROBE

---

## .HBAC

Performs harmonic-balance–based periodic AC analysis on circuits operating in a large-signal periodic steady state.

### Syntax

`.HBAC frequency_sweep`

---

Argument	Description
<code>frequency_sweep</code>	<p>Frequency sweep range for the input signal (also refer to as the input frequency band (IFB) or <i>fin</i>). You can specify LIN, DEC, OCT, POI, SWEEPBLOCK, or DATA. Specify the <i>nsteps</i>, start and stop times using the following syntax for each type of sweep:</p> <ul style="list-style-type: none"> <li>▪ LIN <i>nsteps start stop</i></li> <li>▪ DEC <i>nsteps start stop</i></li> <li>▪ OCT <i>nsteps start stop</i></li> <li>▪ POI <i>nsteps freq_values</i></li> <li>▪ SWEEPBLOCK <i>nsteps freq1 freq2 ... freqn</i></li> <li>▪ DATA=<i>dataname</i></li> </ul>

---

### Description

Use this command to invoke a harmonic balance-based periodic AC analysis to analyze small-signal perturbations on circuits operating in a large-signal periodic steady state.

### See Also

- [.HB](#)
- [.HBNOISE](#)
- [.HBOSC](#)
- [.OPTION HBACTOL](#)
- [.OPTION HBACKRYLOVDIM](#)
- [.PRINT](#)
- [.PROBE](#)



---

## .HBLIN

Extracts frequency translation S-parameters and noise figures.

### Syntax

Without `SS_TONE`

```
.HBLIN frequency_sweep
+ [NOISECALC=1|0|yes|no] [FILENAME=file_name]
+ [DATAFORMAT=ri|ma|db]
+ [MIXEDMODE2PORT=dd|cc|cd|dc|sd|sc|cs|ds]
```

With `SS_TONE`

```
.HBLIN [NOISECALC=1|0|yes|no] [FILENAME=file_name]
+ [DATAFORMAT=ri|ma|db]
+ [MIXEDMODE2PORT=dd|cc|cd|dc|sd|sc|cs|ds]
```

Argument	Description
<i>frequency_sweep</i>	Frequency sweep range for the input signal (also referred to as the input frequency band (IFB) or <i>fin</i> ). You can specify LIN, DEC, OCT, POI, SWEEPBLOCK, or DATA. Specify the <i>nsteps</i> , <i>start</i> , and <i>stop</i> times using the following syntax for each type of sweep: <ul style="list-style-type: none"> <li>▪ LIN <i>nsteps start stop</i></li> <li>▪ DEC <i>nsteps start stop</i></li> <li>▪ OCT <i>nsteps start stop</i></li> <li>▪ POI <i>nsteps freq_values</i></li> <li>▪ SWEEPBLOCK <i>nsteps freq1 freq2 ... freqn</i></li> <li>▪ DATA=<i>dataname</i></li> </ul>
NOISECALC	Enables calculating the noise figure. The default is no (0).
FILENAME	Output file name for the extracted S-parameters or the object name after the <code>-o</code> command-line option. The default is the netlist file name.
DATAFORMAT	Format of the output data file. <ul style="list-style-type: none"> <li>▪ <code>dataformat=RI</code>, real-imaginary. This is the default for <code>.sc#/citi</code> file.</li> <li>▪ <code>dataformat=MA</code>, magnitude-phase. This is the default format for Touchstone file.</li> <li>▪ <code>dataformat=DB</code>, DB(magnitude)-phase.</li> </ul>

Argument	Description
MIXEDMODE2PORT	<p>Mixed-mode data map of output mixed mode S-parameter matrix. The availability and default value for this keyword depends on the first two port (P element) configuration as follows:</p> <ul style="list-style-type: none"> <li>▪ case 1: p1=p2=single-ended (standard-mode P element) available: ss default: ss</li> <li>▪ case 2: p1=p2=balanced (mixed-mode P element) available: dd, cd, dc, cc default: dd</li> <li>▪ case 3: p1=balanced p2=single-ended available: ds, cs default: ds</li> <li>▪ case 4: p1=single p2=balanced available: sd, sc default: sd</li> </ul>

---

### Description

Use this command in HSPICE RF to extract frequency translation S-parameters and noise figures.

### See Also

[.HB](#)  
[.HBAC](#)  
[.PRINT](#)  
[.PROBE](#)

---

## .HBLSP

Performs periodically driven nonlinear circuit analyses for power-dependent S parameters.

### Syntax

```
.HBLSP NHARMS=nh [POWERUNIT=dbm|watt]
+ [SSPCALC=1|0|YES|NO] [NOISECALC=1|0|YES|NO]
+ [FILENAME=file_name] [DATAFORMAT=ri|ma|db]
+ FREQSWEEP freq_sweep POWERSWEEP power_sweep
```

---

Argument	Description
NHARMS	Number of harmonics in the HB analysis triggered by the .HBLSP command.
POWERUNIT	Power unit. Default is watt.
SSPCALC	Extract small-signal S-parameters. Default is 0 (NO).
NOISECALC	Perform small-signal 2-port noise analysis. Default is 0 (NO).
FILENAME	Output data <i>.p2d#</i> filename. Default is the netlist name or the object name after the -o command-line option.
DATAFORMAT	Format of the output data file. Default is ma (magnitude, angle).
FREQSWEEP	Frequency sweep specification. A sweep of type LIN, DEC, OCT, POI, or SWEEPBLOCK. Specify the <i>nsteps</i> , <i>start</i> , and <i>stop</i> times using the following syntax for each type of sweep: <ul style="list-style-type: none"> <li>▪ LIN <i>nsteps start stop</i></li> <li>▪ DEC <i>nsteps start stop</i></li> <li>▪ OCT <i>nsteps start stop</i></li> <li>▪ POI <i>nsteps freq_values</i></li> <li>▪ SWEEPBLOCK=<i>blockname</i></li> </ul> This keyword must appear before the POWERSWEEP keyword.

---

Argument	Description
POWERSWEEP	<p>Power sweep specification. A sweep of type LIN, DEC, OCT, POI, or SWEEPBLOCK. Specify the nsteps, start, and stop times using the following syntax for each type of sweep:</p> <ul style="list-style-type: none"><li>▪ LIN <i>nsteps start stop</i></li><li>▪ DEC <i>nsteps start stop</i></li><li>▪ OCT <i>nsteps start stop</i></li><li>▪ POI <i>nsteps power_values</i></li><li>▪ SWEEPBLOCK=<i>blockname</i></li></ul> <p>This keyword must follow the FREQSWEEP keyword.</p>

---

### Description

Use this command in HSPICE RF to invoke periodically driven nonlinear circuit analyses for power-dependent S-parameters.

For details, see the *HSPICE User Guide: RF Analysis*, [Large-Signal S-parameter \(HBLSP\) Analysis](#).

### See Also

[.HB](#)  
[.PRINT](#)  
[.PROBE](#)

## .HBNOISE

Performs cyclo-stationary noise analysis on circuits operating in a large-signal periodic steady state.

### Syntax

```
.HBNOISE [output] [insrc] [parameter_sweep]
+ [n1, n2, ..., nk,+/-1]
+ [listfreq=(frequencies|none|all)] [listcount=val]
+ [listfloor=val] [listsources=on|off]
```

Argument	Description
output	Output node, pair of nodes, or 2-terminal element. HSPICE RF references equivalent noise output to this node (or pair of nodes). Specify a pair of nodes as V(n+,n-). If you specify only one node, V(n+), then HSPICE RF assumes that the second node is ground. You can also specify a 2-terminal element name that refers to an existing element in the netlist.
insrc	Input source. If this is a resistor, HSPICE RF uses it as a reference noise source to determine the noise figure. If the resistance value is 0, the result is an infinite noise figure.
parameter_sweep	Frequency sweep range for the input signal. Also referred to as the input frequency band (IFB) or <i>fin</i> . You can specify LIN, DEC, OCT, POI, SWEEPBLOCK, DATA, MONTE, or OPTIMIZE sweeps. Specify the nsteps, start, and stop frequencies using the following syntax for each type of sweep: <ul style="list-style-type: none"> <li>▪ LIN <i>nsteps start stop</i></li> <li>▪ DEC <i>nsteps start stop</i></li> <li>▪ OCT <i>nsteps start stop</i></li> <li>▪ POI <i>nsteps freq_values</i></li> <li>▪ SWEEPBLOCK <i>nsteps freq1 freq2 ... freqn</i></li> <li>▪ DATA <i>dataname</i></li> <li>▪ MONTE <i>niterations</i></li> <li>▪ OPTIMIZE <i>optxxx</i></li> </ul>

Argument	Description
n1,n2,...,nk, +/-1	<p>Index term defining the output frequency band (OFB or <i>fout</i>) at which the noise is evaluated. Generally, <math>f_{out} = \text{ABS}(n1*f1 + n2*f2 + \dots + nk*fk +/- fin)</math> Where:</p> <ul style="list-style-type: none"> <li>▪ f1,f2,...,fk are the first through k-th steady-state tones determined from the harmonic balance solution</li> <li>▪ n1,n2,...,nk are the associated harmonic multipliers</li> <li>▪ fin is the IFB defined by <i>parameter_sweep</i>.</li> </ul> <p>The default index term is [1,1,...1,-1]. For a single tone analysis, the default mode is consistent with simulating a low-side, down conversion mixer where the RF signal is specified by the IFB and the noise is measured at a down-converted frequency that the OFB specifies. In general, you can use the [n1,n2,...,nk,+/-1] index term to specify an arbitrary offset. The noise figure measurement is also dependent on this index term.</p>
listfreq	<p>Prints the element noise value to the .lis file. You can specify at which frequencies the element noise value is printed. The frequencies must match the sweep_frequency values defined in the <i>parameter_sweep</i>, otherwise they are ignored. In the element noise output, the elements that contribute the largest noise are printed first. The frequency values can be specified with the NONE or ALL keyword, which either prints no frequencies or every frequency defined in <i>parameter_sweep</i>. Frequency values must be enclosed in parentheses. For example: <code>listfreq=(none)</code>  <code>listfreq=(all)</code> <code>listfreq=(1.0G)</code> <code>listfreq=(1.0G, 2.0G)</code> The default value is NONE.</p>
listcount	<p>Prints the element noise value to the .lis file, which is sorted from the largest to smallest value. You do not need to print every noise element; instead, you can define <code>listcount</code> to print the number of element noise frequencies. For example, <code>listcount=5</code> means that only the top 5 noise contributors are printed. The default value is 1.</p>
listfloor	<p>Prints the element noise value to the .lis file and defines a minimum meaningful noise value (in <math>V/\text{Hz}^{1/2}</math> units). Only those elements with noise values larger than <code>listfloor</code> are printed. The default value is <math>1.0\text{e-}14 V/\text{Hz}^{1/2}</math>.</p>
listsources	<p>Prints the element noise value to the .lis file when the element has multiple noise sources, such as a FET, which contains the thermal, shot, and 1/f noise sources. You can specify either ON or OFF: ON Prints the contribution from each noise source and OFF does not. The default value is OFF.</p>

**Description**

Use this command to invoke cyclo-stationary noise analysis on circuits operating in a large-signal periodic steady state.

**See Also**

[.HB](#)  
[.HBAC](#)  
[.HBOSC](#)  
[.PRINT](#)  
[.PROBE](#)

---

## .HBOSC

Performs oscillator analysis on autonomous (oscillator) circuits. The input syntax for HBOSC analysis supports two different formats, depending on whether the PROBENODE location is specified using a circuit element (current source) or using the HBOSC PROBENODE parameters:

### Syntax

#### Syntax #1

```
.HBOSC TONE=F1 NHARMS=H1
+ PROBENODE=N1,N2,VP
+[FSPTS=NUM, MIN, MAX] [STABILITY=(-2|-1|0|1|2)]
+[SWEEP PARAMETER_SWEEP] [SUBHARMS=I]
```

#### Syntax #2 (Uses current source to set PROBENODE)

```
ISRC N1N2 HBOSCVPROBE=VP
.HBOSC TONE=F1 NHARMS=H1
+[FSPTS=NUM, MIN, MAX] [STABILITY=(-2|-1|0|1|2)]
+[SWEEP PARAMETER_SWEEP] [SUBHARMS=I]
```

---

Argument	Description
TONE	Approximate value for oscillation frequency (Hz). The search for an exact oscillation frequency begins from this value unless you specify an FSPTS range or transient initialization.
NHARMS	Number of harmonics to use for oscillator HB analysis.
PROBENODE	Circuit nodes that are probed for oscillation conditions. <ul style="list-style-type: none"> <li>▪ N1 and N2 are the positive and negative nodes for a voltage probe inserted in the circuit to search for oscillation conditions.</li> <li>▪ VP is the initial probe voltage value (suggested: 1/2 the supply voltage). The phase of the probe voltage is forced to zero; all other phases are relative to the probe phase. HSPICE RF uses this probe to calculate small-signal admittance for the initial frequency estimates. It should be connected near the “heart” of the oscillator (near resonators, inside the ring of a ring oscillator, and so on). Note: The PROBENODE pins and approximate voltage value can also be set by using a zero amp current source that uses the HBOSCVPROBE keyword.</li> </ul>
HBOSCVPROBE = VP	Sets PROBENODE with a current source. If a current source with HBOSCVPROBE is used, the PROBENODE syntax is not necessary.



---

Argument	Description
FSPTS	<p>Frequency search points that HSPICE RF uses in its initial small-signal frequency search to find an oscillation frequency. Optional, but recommended for high-Q and most LC oscillators.</p> <ul style="list-style-type: none"><li>▪ NUM is an integer.</li><li>▪ MIN and MAX are frequency values in units of Hz.</li></ul> <p>If the FSPTS analysis finds an approximate oscillation frequency, the TONE parameter is ignored. An option for FSPTS</p>

---

Argument	Description
STABILITY	<p>When used with FSPTS, activates the additional oscillator stability analyses depending on the following values:</p> <ul style="list-style-type: none"><li>▪ 0: A single point oscillator frequency-search stability analysis is performed. The FSPTS search is executed, and the first successful linear oscillation frequency value found is used as the starting point for the two-tier Newton nonlinear oscillator analysis. The probenode vp value specified is used as the starting amplitude for the Newton solver.</li><li>▪ 1: (default) A single point oscillator frequency-search stability analysis, plus an estimate of oscillator amplitude, is performed. The FSPTS search is executed, and the first successful linear oscillation frequency value found is used as the starting point for the two-tier Newton nonlinear oscillator analysis. An additional analysis for automatically estimating the probenode amplitude is also performed, and this value is used as the starting amplitude for the two-tier Newton solver.</li><li>▪ -1: A single point oscillator frequency-search stability analysis, plus an estimate of oscillator amplitude, is performed. The FSPTS search is executed, and the first successful linear oscillation frequency value found is accurately computed and reported. An additional analysis for automatically estimating the probenode amplitude is also performed, and this value is also reported. The analysis aborts without attempting the two-tier Newton nonlinear oscillator analysis. By using STABILITY=-1, a check can be made if any linear oscillation conditions are found, before attempting the nonlinear oscillator analysis.</li><li>▪ 2: A multipoint frequency-search stability analysis is performed. The FSPTS search is executed, and all successful linear oscillation frequency values found over the entire FSPTS search range are reported. For each potential oscillation frequency found, an additional analysis for estimating the probenode amplitude is also performed. All frequency and amplitude values are reported. The frequency value that has the largest predicted amplitude is used as the starting point for the two-tier Newton nonlinear oscillator analysis.</li><li>▪ -2: A multipoint frequency-search stability analysis is performed. The FSPTS search is executed, and all successful linear oscillation frequency values found over the entire FSPTS search range are reported. For each potential oscillation frequency found, an additional analysis for estimating the probenode amplitude is also performed. All frequency and amplitude values are reported. The analysis aborts without attempting the two-tier Newton nonlinear oscillator analysis. By using STABILITY=-2, a check can be made if any linear oscillation conditions are found, before attempting the nonlinear oscillator analysis.</li></ul>

Argument	Description
SWEEP	<p>Type of sweep. You can sweep up to three variables. You can specify either LIN, DEC, OCT, POI, SWEEPBLOCK, DATA, OPTIMIZE, or MONTE. Specify the nsteps, start, and stop frequencies using the following syntax for each type of sweep:</p> <ul style="list-style-type: none"> <li>▪ LIN <i>nsteps start stop</i></li> <li>▪ DEC <i>nsteps start stop</i></li> <li>▪ OCT <i>nsteps start stop</i></li> <li>▪ POI <i>nsteps freq_values</i></li> <li>▪ SWEEPBLOCK <i>nsteps freq1 freq2 ... freqn</i></li> <li>▪ DATA=<i>dataname</i></li> <li>▪ OPTIMIZE=OPT<i>xxx</i></li> <li>▪ MONTE=<i>val</i></li> </ul>
SUBHARMS	<p>Subharmonics in the analysis spectrum. The minimum non-DC frequency in the analysis spectrum is <math>f/\text{subharms}</math>, where <math>f</math> is the frequency of oscillation. Use this option if your oscillator circuit includes a divider or prescaler that result in frequency terms that are subharmonics of the fundamental oscillation frequency</p>

### Description

Use this command to invoke oscillator analysis on autonomous (oscillator) circuits.

### Examples

*Example 1* Performs an oscillator analysis searching for frequencies in the vicinity of 900 MHz. This example uses nine harmonics with the probe inserted between the gate and gnd nodes. The probe voltage estimate is 0.65 V.

```
.HBOSC tone=900MEG nharms=9 probenode=gate,gnd,0.65
```

*Example 2* Performs an oscillator analysis searching for frequencies in the vicinity of 2.4 GHz. This example uses 11 harmonics with the probe inserted between the drainP and drainN nodes. The probe voltage estimate is 1.0 V.

```
.HBOSC tone=2400MEG nharms=11
+ probenode=drainP,drainN,1.0 fspts=20,2100MEG,2700MEG
```

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .HBOSC

Another means to define the probenode information is through a zero-current source. Examples 3 and 4 shows two methods define an equivalent .HBOSC command.

#### *Example 3 Method 1*

```
.HBOSC tone = 2.4G nharms = 10
+ probenode = drainP, drainN, 1.0
+ fspts = 20, 2.1G, 2.7G
```

In Method 2, the PROBENODE information is defined by a current source in the circuit. Only one such current source is needed, and its current must be 0.0 with the HBOSC PROBENODE voltage defined through its HBOSCVPROBE property.

#### *Example 4 Method 2*

```
ISRC drainP drainN 0 HBOSCVPROBE = 1.0
.HBOSC tone = 2.4G nharms = 10
+ fspts = 20, 2.1G, 2.7G
```

### See Also

- [.HB](#)
- [.OPTION HBFREQABSTOL](#)
- [.OPTION HBFREQRELTOL](#)
- [.OPTION HBOSCMAXITER \(or\) HBOSC\\_MAXITER](#)
- [.OPTION HBPROBETOL](#)
- [.OPTION HBTRANFREQSEARCH](#)
- [.OPTION HBTRANINIT](#)
- [.OPTION HBTRANPTS](#)
- [.OPTION HBTRANSTEP](#)
- [.PRINT](#)
- [.PROBE](#)

---

## .HBXF

Calculates transfer from the given source in the circuit to the designated output.

### Syntax

```
.HBXF out_varfreq_sweep
```

---

Argument	Description
out_var	Specify $i(2\_port\_elem)$ or $v(n1<, n2>)$
freq_sweep	A sweep of type LIN, DEC, OCT, POI, or SWEEPBLOCK. Specify nsteps, start/stop times the syntax below for each type of sweep: <ul style="list-style-type: none"> <li>▪ LIN <i>nsteps start stop</i></li> <li>▪ DEC <i>nsteps start stop</i></li> <li>▪ OCT <i>nsteps start stop</i></li> <li>▪ POI <i>nsteps freq_values</i></li> <li>▪ SWEEPBLOCK = <i>BlockName</i></li> </ul> Specify the frequency sweep range for the output signal. HSPICE RF determines the offset frequency in the input sidebands; for example, $f_1 = \text{abs}(f_{out} - k \cdot f_0)$ s.t. $f_1 \leq f_0/2$ The $f_0$ is the steady-state fundamental tone and $f_1$ is the input frequency.

---

### Description

Calculates the transfer function from the given source in the circuit to the designated output.

### Examples

Here, trans-impedance from *isrc* to *v(1)* is calculated based on HB analysis.

```
.hb tones=1e9 nharms=4
.hbxf v(1) lin 10 1e8 1.2e8
.print hbxf tfv(isrc) tfi(n3)
```

### See Also

[.HB](#)  
[.HBAC](#)  
[.HBNOISE](#)  
[.HBOSC](#)  
[.SNXF](#)

---

## .HDL

Specifies the Verilog-A source name and path.

### Syntax

```
.HDL "file_name" [module_name] [module_alias]
```

---

Argument	Description
file_name	Verilog-A or CML file.
module_name	Optional module name. If a module is specified, then only that module is loaded from the specified Verilog-A or CML file. If the module is not found or if the module specification is not uniquely case-insensitive inside, then an error is generated. (HSPICE only).
module_alias	If specified (in addition to a module name), then that module is loaded into the system using the alias in place of the module name defined in the Verilog-A source file. Thereafter, any reference to the module is made using its alias. The system behaves as if the module had the alias as its module name. A module might be loaded with any number of aliases in addition to being loaded without an alias. This argument is useful when loading modules of the same name from different files. See Example 4 below. (HSPICE only)

---

### Description

Use `.HDL` commands to specify the Verilog-A or compiled model library (CML) source name and path within a netlist. The Verilog-A file is assumed to have a `*.va` extension only when a prefix is provided. You can also use `.HDL` commands in `.ALTER` blocks to vary simulation behavior. For example, to compare multiple variations of Verilog-A modules.

In `.MODEL` commands you must add the Verilog-A type of model cards. Every Verilog-A module can have one or more associated model cards. The type of model cards should be the same as the Verilog-A module name. Verilog-A module names cannot conflict with HSPICE built-in device keywords. If a conflict occurs, HSPICE issues a warning message and the Verilog-A module definition is ignored.

The `module_name` and `module_alias` arguments can be specified without quotes or with single or double quotes. Any tokens after the module alias are ignored.

The same Verilog-A case insensitivity rules used for module and parameter names apply to both the `module_name` and `module_alias` arguments, and the same module override logic applies.

### Examples

Example 1 loads the `res.va` Verilog-A model file from the directory `/myhome/Verilog_A_lib`.

#### *Example 1*

```
.HDL "/myhome/Verilog_A_lib/res.va"
```

Example 2 loads the `va_models.va` Verilog-A model file (not `va_model` file) from the current working directory.

#### *Example 2*

```
.HDL "va_models"
```

Example 3 loads the module called `va_amp` from the `amp_one.va` file for the first simulation run. For the second run, HSPICE loads the `va_amp` module from the `amp_two.va` file.

#### *Example 3*

```
* simple .alter test
.hdl amp_one.va
v1 1 0 10
x1 1 0 va_amp
.tran 10n 100n
.alter alter1
.hdl amp_two.va
.end
```

### See Also

[.ALTER](#)  
[.MODEL](#)

---

## .IBIS

Provides IBIS functionality by specifying an IBIS file and component and optional keywords.

### Syntax

```
.IBIS 'ibis_name'
+ file='ibis_file_name'
+ component='component_name' [time_control=0|1]
+ [mod_sel='sel1=mod1,sel2=mod2,...']
+ [package=0|1|2|3] [pkgfile='pkg_file_name']
+ [typ={typ|min|max}]
+ [nowarn]
+ ...
```

---

### Argument (Keyword) Description

---

ibis_name	Instance name of this ibis command.
file	Name of ibis ( <i>*ibs</i> ) file.
component or cname	Component name.
time_control	Invokes an HSPICE time-control algorithm to achieve greater accuracy for high speed digital signal buffers: <ul style="list-style-type: none"> <li>▪ 0 (default): Time step algorithm will not take effect.</li> <li>▪ 1: Launches the time-step algorithm.</li> </ul>
mod_sel	Assigns special model for model selector, here model selector can be used for series model. If model selector is used for a pin of a component, but <code>mod_sel</code> is not set in the <code>.ibis</code> command, then the first model under the corresponding [Model Selector] will be selected as default.



Argument (Keyword)	Description
package	<p>When package equals:</p> <p>0, then the package is not added into the component.</p> <p>1, then RLC of [Package] (in the <i>.ibs</i> file) is added.</p> <p>2, then RLC of [Pin] (in the <i>.ibs</i> file) is added.</p> <p>3 (default), and if [Package Model] is defined, set package with a package model.</p> <p>If the [Package Model] is not defined, set the package with [Pin].</p> <p>If the package information is not set in [Pin], set the package with [Package] as a default.</p> <p>You can define the [Package Model] in an IBIS file specified by the <i>file</i> keyword or a PKG file specified by the <b>pkgfile</b> keyword.</p> <p>The pkgfile keyword is useful only when package =3</p>
typ	<p>The value of the typ signifies a column in the <i>IBIS</i> file from which the current simulation extracts data. The default is typ=typ. If min or max data are not available, typ data are used instead.</p>
nowarn	<p>The nowarn keyword suppresses warning messages from the IBIS parser.</p>

### Description

The general syntax above shows the `.IBIS` command when used with a component. The optional keywords are in square brackets.

### Examples

```
.ibis cmpnt
+ file = 'ebd.ibs'
+ component = 'SIMM'
+ hsp_ver=2002.4 nowarn package=2
```

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

.IBIS

This example corresponds to the following *ebd.ibs* file:

```
[Component]      SIMM
[Manufacturer]   TEST
[Package]
R_pkg           200m      NA      NA
L_pkg           7.0nH     NA      NA
C_pkg           1.5pF     NA      NA
|
[Pin]           signal_name  model_name  R_pin    L_pin    C_pin
|
1      ND1      ECL      40.0m    2n      0.4p
2      ND2      NMOS     50.0m    3n      0.5p
.....
```

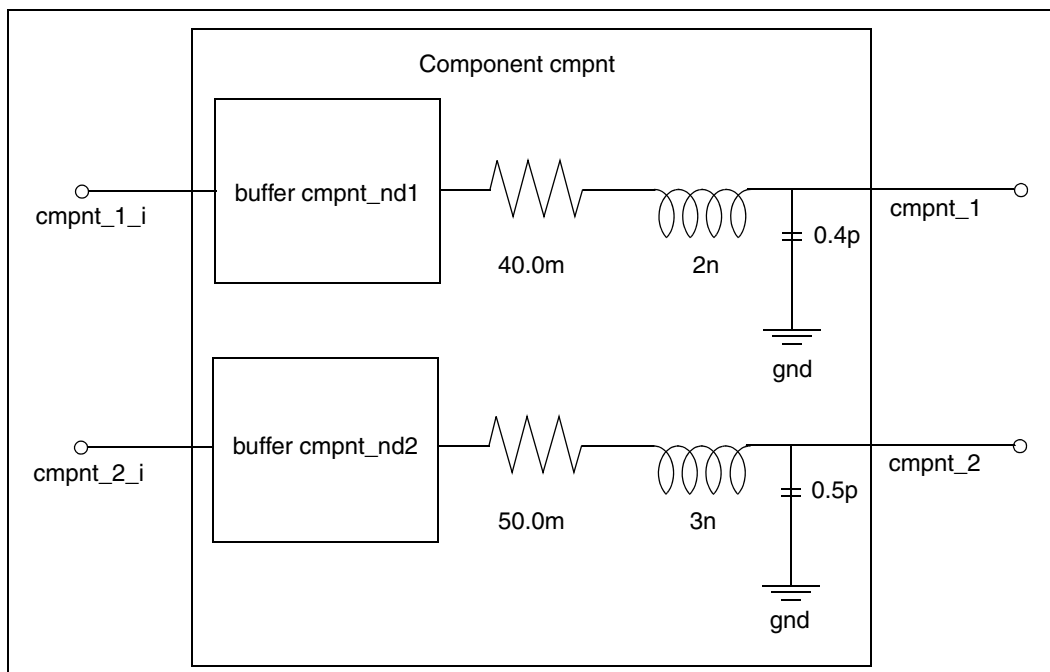


Figure 8 Equivalent Circuit for IBIS Component Example

```
.IBIS cmpt1
+ file='example.ibs'
+ component='EXAMPLE'
+ mod_sel = 'DQ = DQ_FULL'
```

In the following example, the model `DQ_FULLL` will be used for all pins that use the model name `DQ`. The corresponding IBIS file, *example.ibs*, contains the following [Model Selector] section:

```
*****MODEL_SELECTOR*****  
|  
| Model Selector] DQ  
|  
DQ_FULLL           Full-Strength IO Driver  
DQ_HALF           54% Reduced Drive Strength IO Driver  
*
```

**See Also**

[.EBD](#)

[.PKG](#)

[IBIS Examples \(/iob\\_ex1.sp\)](#) for demonstration files and see `.IBIS` command use in *ebd.sp* and *pinmap.sp*

---

## .IC

Sets transient initial conditions in HSPICE.

### Syntax

```
.IC V(node1)=val1 V(node2)=val2 ... [subckt=sub_name]
```

---

Argument	Description
<i>val1</i> ...	Specifies voltages. The significance of these voltages depends on whether you specify the UIC parameter in the .TRAN command.
<i>node1</i> ...	Node numbers or names can include full paths or circuit numbers.
<i>subckt</i> = <i>sub_name</i>	Initial condition is set to the specified node name(s) within all instances of the specified subcircuit name. This subckt setting is equivalent to placing the .IC statement within the subcircuit definition.

---

### Description

Use the .IC command or the .DCVOLT command to set transient initial conditions in HSPICE. How it initializes depends on whether the .TRAN analysis command includes the UIC parameter. This command is less preferred compared to using the .NODESET command in many cases.

When using the .IC command, forcing circuits are connected to the .IC nodes for the duration of DC convergence. After DC convergence is obtained, the forcing circuits are removed for all further analysis. The DC operating point for each .IC'd node should be very close to the voltage specified in the .IC command. If a node is not, then that node has a DC conductance to ground comparable to GMAX. This is almost certainly an error condition. In the rare case that it is not, GMAX can be increased to prevent appreciable current division. Example: .OPTION GMAX=1000

**Note:** In nearly all applications, .NODESET should be used to ensure a true DC operating point is obtained. Intentionally floating (or very high impedance) nodes should be set to a known good voltage using .IC.

If you do not specify the UIC parameter in the .TRAN command, HSPICE computes the DC operating point solution before the transient analysis. The node voltages that you specify in the .IC command are fixed to determine the DC operating point. They are used only in the first iteration to set an initial guess for the DC operating point analysis. The .IC command is equivalent to

specifying the IC parameter on each element command, but is more convenient.

If you specify the UIC parameter in the `.TRAN` command, HSPICE does not calculate the initial DC operating point, but directly enters transient analysis. When you use `.TRAN UIC`, the `.TRAN` node values (at time zero) are determined by searching for the first value found in this order: from `.IC` value, then IC parameter on an element command, then `.NODESET` value, otherwise use a voltage of zero.

Note that forcing a node value of the dc operating point may not satisfy KVL and KCL. In this event you may likely see activity during the initial part of the simulation. This may happen if UIC is used and some node values left unspecified, when too many (conflicting) `.IC` values are specified, or when node values are forced and the topology changes. In this event you may likely see activity during the initial part of the simulation. Forcing a node voltage applies a fixed equivalent voltage source during DC analysis and transient analysis removes the voltage sources to calculate the second and later time points.

Therefore, to correct DC convergence problems use `.NODESETs` (without `.TRAN UIC`) liberally (when a good guess can be provided) and use `.ICs` sparingly (when the exact node voltage is known).

In addition, you can use wildcards in the `.IC` command. See [Using Wildcards on Node Names](#) in the *HSPICE User Guide: Simulation and Analysis*.

## Examples

### Example 1

```
.IC V(11)=5 V(4)=-5 V(2)=2.2
```

### Example 2 *All settings in this statement are applied to subckt my\_ff.*

```
.IC V(in)=0.9 subckt=my_ff
```

## See Also

- [.DCVOLT](#)
- [.TRAN](#)
- [.OPTION DCIC](#)
- [.NODESET](#)
- [.OPTION GMAX](#)

---

## .ICM

Automatically creates port names that reference the pin name of an ICM model and generate a series of element nodes on the pin.

### Syntax

```
.ICM icmname
+ file='icmfilename'
+ model='icmmodelname'
```

---

Argument	Description
<i>icmname</i>	.ICM command card name.
<i>icmfilename</i>	Name of an *. <i>icm</i> file that contains an ICM model.
<i>icmmodelname</i>	Working model in an . <i>icm</i> file.
<i>nodemapname</i>	Name of the [ICM node map] keyword in an . <i>icm</i> file.
<i>pinmapname</i>	Name of the [ICM pin map] keyword in an . <i>icm</i> file.
<i>pinname</i>	Name of the first column of entries of the [ICM node map] or [ICM pin map].
<i>sidename</i>	Name of the side subparameter

---

### Description

Use this command to automatically create port names that reference the pin name of an ICM model and generate a series of element (W/S/RLGCK) nodes on the pin when one of the following conditions occur:

- If the model is described using [Nodal Path Description] *'icmname'\_'nodemapname'\_'sidename'\_'pinname'*
- If the model is described using [Tree Path Description] *'icmname'\_'pinmapname'\_'sidename'\_'pinname'*

**Note:** If a side subparameter is not used in an ICM file, then *'sidename'\_'* (above) should be removed.

## Examples

```
.ICM icm1  
+ file='test1.icm'  
+ model='FourLineModel11'
```

The following example shows how to reference a pin of the ICM model in a HSPICE netlist.

```
icm1_NodeMap1_SideName1_pin1, icm1_NodeMap2_SideName2_pin1,  
icm1_NodeMap2_SideName2_pin2, ...
```

## See Also

[IBIS Examples](#) for .ICM command usage (RLGC approach—*/icm/nodepath\_rlgc/bga\_1.sp*), (S-element approach—*/icm/nodepath\_sele test1.sp*), (treepath—*test1.sp*), and treepath swath matrix expansion (*/icm/treepath\_swath/complex.sp*)

---

## .IF

Specifies conditions that determine whether HSPICE executes subsequent commands in conditional block.

### Syntax

```
.IF (condition1)...
+ [.ELSEIF (condition2)...]
+ [.ELSE ...]
.ENDIF
```

---

Argument	Description
<i>condition1</i>	Condition that must be true before HSPICE executes the commands that follow the <code>.IF</code> command.
<i>condition2</i>	Condition that must be true before HSPICE executes the commands that follow the <code>.ELSEIF</code> command. HSPICE executes the commands that follow <i>condition2</i> only if <i>condition1</i> is false and <i>condition2</i> is true.

---

### Description

HSPICE executes the commands that follow the first `.ELSEIF` command only if *condition1* in the preceding `.IF` command is false and *condition2* in the first `.ELSEIF` command is true.

If *condition1* in the `.IF` command and *condition2* in the first `.ELSEIF` command are both false, then HSPICE moves on to the next `.ELSEIF` command if there is one. If this second `.ELSEIF` condition is true, HSPICE executes the commands that follow the second `.ELSEIF` command, instead of the commands after the first `.ELSEIF` command.

HSPICE ignores the commands in all false `.IF` and `.ELSEIF` commands, until it reaches the first `.ELSEIF` condition that is true. If no `.IF` or `.ELSEIF` condition is true, HSPICE continues to the `.ELSE` command.

`.ELSE` precedes one or more commands in a conditional block after the last `.ELSEIF` command, but before the `.ENDIF` command. HSPICE executes these commands by default if the conditions in the preceding `.IF` command and in all of the preceding `.ELSEIF` commands in the same conditional block all false.

The `.ENDIF` command ends a conditional block of commands that begins with an `.IF` command.



For information on use of conditional blocks with the Exploration Block, see, [Specifying Constraints](#) in the *HSPICE User Guide: Simulation and Analysis*.

### Examples

```
.IF (a==b)
.INCLUDE /myhome/subcircuits/diode_circuit1
...
.ELSEIF (a==c)
.INCLUDE /myhome/subcircuits/diode_circuit2
...
.ELSE
.INCLUDE /myhome/subcircuits/diode_circuit3
...
.ENDIF
```

### See Also

- [.ELSE](#)
- [.ELSEIF](#)
- [.ENDIF](#)

---

## .INCLUDE (or) .INC (or) .INCL

Includes another netlist as a subcircuit of the current netlist.

### Syntax

```
.INCLUDE `file_pathfile_name`
```

---

Argument	Description
file_path	Path name of a file for computer operating systems that support tree-structured directories.  An include file can contain nested <code>.INCLUDE</code> calls to itself or to another include file. If you use a relative path in a nested <code>.INCLUDE</code> call, the path starts from the directory of the parent <code>.INCLUDE</code> file, not from the current working directory. If the path starts from the current working directory, HSPICE can also find the <code>.INCLUDE</code> file, but prints a warning.
file_name	Name of a file to include in the data file. The file path, plus the file name, can be up to 16 characters long. You can use any valid file name for the computer's operating system.

---

### Description

Use this command to include another netlist in the current netlist. You can include a netlist as a subcircuit in one or more other netlists. You must enclose the file path and file name in single or double quotation marks. Otherwise, an error message is generated. Any file name following an `.INC` command is case sensitive beginning with the 2009.09 release. This command can be used as part of a compressed (`.gzip`) netlist file.

### Examples

```
.INCLUDE `/myhome/subcircuits/diode_circuit`
```

---

## .IVTH

invokes the constant-current based threshold voltage characterization. The threshold voltage reported by `ivth` characterization is defined as the MOSFET's gate-to-source voltage at which the drain terminal current reaches the user-defined constant current value. The drain and body biases of the device are set to their corresponding bias conditions in the circuit. For example, in DCOP, the drain and body bias of the device is set to its operating point condition. in DC Sweep or transient analysis, drain and body bias of the device is set to its solution at each sweep or time point.

### Syntax

```
.IVTH model_name Ivth0=val DW=val DL=val VDSMIN=val
[.OPTION SX_factor=x]
```

---

Argument	Description
<i>model_name</i>	Model name that <code>ivth</code> characterization applies to.
<i>Ivth0=val</i>	Constant drain terminal current density.
<i>DW=val</i>	Width offset for <code>ivth</code> current calculation.
<i>DL=val</i>	Length offset for <code>ivth</code> current calculation.
<i>VDSMIN=val</i>	User-defined minimum <code>vds</code> value.
<code>.OPTION SX_factor</code>	A special option, <code>.OPTION SX_factor</code> , is provided to scale the width and length specifically for <code>ivth</code> characterization.

---

### Description

Use this command to enable constant current-based threshold voltage characterization. The constant current for each MOSFET is given as follows:

$$Ivth = Ivth0 * (Wdrawn * SX\_factor + DW) / (Ldrawn * SX\_factor + DL)$$

`VDSMIN` provides a user-defined minimum `Vds` value and invokes a special characterization method for small `Vds` bias to ensure continuation and meaningful characterization result.

If `VDSMIN` is not given, the same `ivth` characterization methodology is applied for all `vds` bias regions. If `VDSMIN` is given, and `Vds` is smaller than `VDSMIN`, then:

1. Simulate `Vth_op(Vdsmin)` and `Vth_ivth(Vdsmin)`

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .IVTH

where:  $V_{th\_op}()$  is the threshold voltage acquired from model formulation, and  $v_{th\_ivth}()$  is the threshold voltage acquired from  $iv_{th}$  method.

2. Calculate  $\Delta V_{th} = V_{th\_op}(V_{dsmin}) - v_{th\_ivth}(V_{dsmin})$

3. Simulate  $V_{th\_op}(V_{ds})$

4. Calculate  $v_{th\_ivth}(V_{ds}) = V_{th\_op}(V_{ds}) - \Delta V_{th}$

Multiple  $iv_{th}$  commands can be added in a netlist to invoke characterization of different models.

### Examples

```
.ivth nch Ivth0=1.5e-7 DW=2e-8 DL=1e-8 VDSMIN=0.06  
.ivth pch Ivth0=1e-7 DW=2e-8 DL=1e-8 VDSMIN=0.06
```

In OP analysis, a constant current based  $v_{th}$  is reported in the OP output. In addition, the element region operation check and  $V_{od}$  output are based on the new  $v_{th}$ .

During transient or DC analysis, template output of  $LX142(m^*)$  or  $iv_{th}(m^*)$  could be used for the new  $v_{th}$  output.

---

## .JITTER

Computes jitter time interval error measurements from transient noise or non-random noise errors from transient analyses.

### Syntax

*Computation of TIE (time interval error)*

```
.JITTER TRANNOISE TRIG OutputVar VAL=num|expression
+ TD=num|expression
```

*Computation of deterministic jitter without random noise*

```
.JITTER TRAN TRIG OutputVar VAL=num|expression
+ TD=num|expression
```

---

Argument	Description
TRIG	Repetitive trigger specification.
<i>OutputVar</i> VAL=num expression	The output variable is expected to be an ideal clock reference. The value can be a number or an expression.
TD	Amount of simulation time that must elapse before the measurement is enabled. The time delay can be a number or an expression.

---

### Description

Use this command following a TRANNOISE analysis which used the autocorrelation=1 keyword. The autocorrelation function is calculated at the specified output. TIE measures the time-shift behavior relative to a reference signal. The autocorrelation function is used for tracking the relative time-shift behavior of the signal.

---

## .LAYERSTACK

Defines a stack of dielectric or metal layers.

### Syntax

```
.LAYERSTACK sname [BACKGROUND=mname]  
+ [LAYER=(mname,thickness) ...]
```

---

Argument	Description
sname	Layer stack name.
mname	Material name.
BACKGROUND	Background dielectric material name. By default, the field solver assumes AIR for the background.
thickness	Layer thickness.

---

### Description

Use this command to define a stack of dielectric or metal layers. You must associate each transmission line system with *only* one layer stack. However, you can associate a single-layer stack with many transmission line systems.

In the layer stack:

- Layers are listed from bottom to top.
- Metal layers (ground planes) can be located only at the bottom, only at the top, or both at the top and bottom.
- Layers are stacked in the y-direction; the bottom of a layer stack is at  $y=0$ .
- All conductors must be located above  $y=0$ .
- Background material must be dielectric.

The following limiting cases apply to the .LAYERSTACK command:

- Free space without ground:  
.LAYERSTACK mystack
- Free space with a (bottom) ground plane where a predefined metal name = perfect electrical conductor (PEC):  
.LAYERSTACK halfSpace PEC 0.1mm

**See Also**

[.FSOPTIONS](#)

[.MATERIAL](#)

[.SHAPE](#)

[Transmission \(W-element\) Line Examples](#)

---

## .LIB

Creates and reads from libraries of commonly used commands, device models, subcircuit analyses, and commands.

### Syntax

Use the following syntax for library calls:

```
.LIB `[file_path] file_name' entry_name
```

Use the following syntax to define library files:

```
.LIB entry_name1
. $ ANY VALID SET OF HSPICE STATEMENTS
.ENDL entry_name1
.LIB entry_name2
.
. $ ANY VALID SET OF HSPICE STATEMENTS
.ENDL entry_name2
.LIB entry_name3
.
. $ ANY VALID SET OF HSPICE STATEMENTS
.ENDL entry_name3
```

---

Argument	Description
<code>file_path</code>	Path to a file. Used where a computer supports tree-structured directories. When the LIB file (or alias) is in the same directory where you run HSPICE RF you do not need to specify a directory path; the netlist runs on any machine. Use “. . /” syntax in the <code>file_path</code> to designate the parent directory of the current directory.
<code>entry_name</code>	Entry name for the section of the library file to include. The first character of an <code>entry_name</code> cannot be an integer. If more than one entry with the same name is encountered in a file, only the first one is loaded.
<code>file_name</code>	Name of a file to include in the data file. The combination of <code>filepath</code> plus <code>file_name</code> can be up to 256 characters long, structured as any filename that is valid for the computer’s operating system. Enclose the file path and file name in single or double quotation marks. Use “. . /” syntax in the filename to designate the parent directory of the current directory.

---

### Description

Use the `.LIB` call command to read from libraries of commonly used commands, device models, subcircuit analyses, and commands (library calls) in library files. Note that as HSPICE RF encounters each `.LIB` call name in the



main data file, it reads the corresponding entry from the designated library file, until it finds an `.ENDL` command.

You can also place a `.LIB` call command in an `.ALTER` block.

To build libraries (library file definition), use the `.LIB` command in a library file. For each macro in a library, use a library definition command (`.LIB entry_name`) and an `.ENDL` command. The `.LIB` command begins the library macro and the `.ENDL` command ends the library macro. The text after a library file entry name must consist of HSPICE RF commands. Library calls can call other libraries (nested library calls) if they are different files. You can nest library calls to any depth. Use nesting with the `.ALTER` command to create a sequence of model runs. Each run can consist of similar components by using different model parameters without duplicating the entire input file.

The simulator uses the `.LIB` command and the `.INCLUDE` command to access the models and skew parameters. The library contains parameters that modify `.MODEL` commands.

You must enclose the file path and file name in single or double quotation marks. Otherwise, an error message is generated. Any file name following a `.LIB` command is case sensitive beginning with the 2009.09 release. To terminate the `.LIB` command use `.ENDL` or `.ENDL TT`.

This command can be used as part of a compressed (*.gzip*) netlist file.

### Examples

Example 1 is a simple library call.

#### Example 1

```
* Library call
.LIB 'MODELS' cmos1
```

Example 2 shows the syntax of using any valid set of RF commands.

#### Example 2

```
.LIB MOS7
$ Any valid set of HSPICE RF commands
.
.
.
.ENDL MOS7
```

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

.LIB

Example 3 is an example of *illegal* nested .LIB commands for the file3 library.

### Example 3

```
.LIB MOS7
...
.LIB 'file3' MOS7 $ This call is illegal in MOS7 library
...
.ENDL
```

Example 4 is a .LIB call command of model skew parameters and features both worst-case and statistical distribution data. The statistical distribution median value is the default for all non-Monte Carlo analyses. The model is in the /usr/meta/lib/cmos1\_mod.dat include file.

### Example 4

```
.LIB TT
$TYPICAL P-CHANNEL AND N-CHANNEL CMOS LIBRARY
$ PROCESS: 1.0U CMOS, FAB7
$ following distributions are 3 sigma ABSOLUTE GAUSSIAN
.PARAM TOX=AGAUSS(200,20,3) $ 200 angstrom +/- 20a
+ XL=AGAUSS(0.1u,0.13u,3) $ polysilicon CD
+ DELVTON=AGAUSS(0.0,.2V,3) $ n-ch threshold change
+ DELVTOP=AGAUSS(0.0,.15V,3)
  $ p-ch threshold change
.INC '/usr/meta/lib/cmos1_mod.dat'
  $ model include file
.ENDL TT
.LIB FF
$HIGH GAIN P-CH AND N-CH CMOS LIBRARY 3SIGMA VALUES
.PARAM TOX=220 XL=-0.03 DELVTON=-.2V
+ DELVTOP=-0.15V
.INC '/usr/meta/lib/cmos1_mod.dat'
  $ model include file
.ENDL FF
```

In example 5, the .MODEL keyword (left side) equates to the skew parameter (right side). A .MODEL keyword can be the same as a skew parameter.

### Example 5

```
.MODEL NCH NMOS LEVEL=2 XL=XL TOX=TOX
+ DELVTO=DELVTON .....
.MODEL PCH PMOS LEVEL=2 XL=XL TOX=TOX
+ DELVTO=DELVTOP .....
```

**See Also**

.ALTER  
.ENDL (or) .ENDL TT  
.INCLUDE (or) .INC (or) .INCL

---

## .LIN

Extracts noise and linear transfer parameters for a general multiport network.

### Syntax

#### *Multiport Syntax*

```
.LIN [sparcalc=[1|0] [modelname = modelname]]
+ [filename = filename]
+ [format=selem|citi|touchstone|touchstone2]
+ [noisecalc=[2|1|0] [gdcalc=[1|0]]]
+ [mixedmode2port=dd|dc|ds|cd|cc|cs|sd|sc|ss] >
+ [dataformat=ri|ma|db]
```

#### *Two-Port Syntax*

```
.LIN [sparcalc=1|0 [modelname = modelname]]
+ [filename = filename]
+ [format=selem|citi|touchstone|touchstone2]
+ [noisecalc=1|0] [gdcalc=1|0]
+ [mixedmode2port=dd|dc|ds|cd|cc|cs|sd|sc|ss]
+ [dataformat=ri|ma|db] [FREQDIGIT=x] [SPARDIGIT=x]
+ [listfreq=(frequencies|none|all|freq1 freq2...)]
+ [listcount=num] [listfloor=val] [listsources=1|0|yes|no]
```

Argument	Description
sparcalc	If 1, extract S parameters (default).
modelname	Model name to be listed in the .MODEL command in the .sc# model output file.
filename	Output file name (The default is netlist name).
format	Output file format: <ul style="list-style-type: none"> <li>▪ selem is for S-element .sc# format, which you can include in the netlist.</li> <li>▪ citi is CITIfile format.</li> <li>▪ touchstone and touchstone2: TOUCHSTONE v1.0 and v2.0 format, respectively.</li> </ul>
noisecalc	Specifies level of N-port noise wave correlation matrix extraction. If 2, extract N noise parameters (perform multiport noise analysis). If 1, extract noise parameters (perform 2-port noise analysis). Default is 0.

Argument	Description
gdcalc	If 1, extract group delay (perform group delay analysis). Default is 0.
mixedmode2port	<p>The mixedmode2port keyword describes the mixed-mode data map of output mixed mode S-parameter matrix. The availability and default value for this keyword depends on the first two port (P-element) configuration as follows:</p> <ul style="list-style-type: none"> <li>▪ case 1: p1=p2=single (standard mode P element) available: ss default: ss</li> <li>▪ case 2: p1=p2=balanced (mixed mode P element) available: dd, cd, dc, cc default: dd</li> <li>▪ case 3: p1=balanced p2=single available: ds, cs default: ds</li> <li>▪ case 4: p1=single p2=balanced available: sd, sc default: sd</li> </ul>
dataformat	<p>The dataformat keyword describes the data format output to the <i>.sc#/touchstone/citi</i> file.</p> <ul style="list-style-type: none"> <li>▪ dataformat=RI, real-imaginary. This is the default for the <i>.sc#/citi</i> file.</li> <li>▪ dataformat=MA, magnitude-phase. This is the default format for touchstone file.</li> <li>▪ dataformat=DB, DB(magnitude)-phase.</li> </ul> <p>HSPICE uses six digits for both frequency and S-parameters in HSPICE generated data files (<i>.sc#/touchstone/citifile</i>). The number of digits for noise parameters are five in <i>.sc#</i> and Touchstone files and six in CITIfiles.</p> <p><b>Note:</b> The lower limit of DB output is -300.</p>
FREQDIGIT	Sets the numerical precision (number of digits) for frequency output in Touchstone, Citi, or <i>sc#</i> files. The default is 6.
SPARDIGIT	Sets the numerical precision (number of digits) for S parameter output in Touchstone, Citi, or <i>sc#</i> files. The default is 6.

Argument	Description
listfreq= (none all freq1req2....)	<p>Dumps the element noise figure contribution to the total NF in the <i>*.lis</i> file. You can specify at which frequencies HSPICE dumps the element noise figure contribution. The elements that contribute the largest noise figure are dumped first. The frequency values can be specified by the NONE or ALL keyword, which either dumps no frequencies or every frequency defined in the AC sweep.</p> <ul style="list-style-type: none"> <li>▪ ALL: Output all of the frequency points (default, if LIST* is required).</li> <li>▪ NONE - Do not output any of the frequency points.</li> <li>▪ freq1 freq2...: Output the information on the specified frequency points.</li> </ul> <p>For example:</p> <pre>listfreq=none listfreq=all listfreq=1.0G listfreq=1.0G 2.0G</pre>
listcount= <i>num</i>	Outputs the first few noise elements that make the biggest contribution to NF. The number is specified by <i>num</i> . The default is to output all of the noise element contribution to NF. The NF contribution is calculated with the source impedance equal to the Zo of the first port.
listfloor= <i>val</i>	Lists elements whose noise contribution to NF (in dB) are higher than value specified in dB to <i>.lis</i> file. Default is 0.
listsources=[1 0 yes no]	Defines whether or not to output the contribution of each noise source of each noise element. Default is no/0.

### Description

Use this command to extract noise and linear transfer parameters for a general multiport network.

When used with P- (port) element(s) and .AC commands, .LIN makes available a broad set of linear port-wise measurements:

- standard and mixed-mode multiport S- (scattering) parameters
- standard and mixed-mode multiport Y/Z parameters
- standard mode multiport H-parameter
- standard mode two-port noise parameters
- standard and mixed-mode group delays

- standard mode stability factors
- standard mode gain factors
- standard mode matching coefficients

The `.LIN` command computes the S-(scattering), Y-(admittance), Z-(impedance) parameters directly, and H-(hybrid) parameters directly based on the location of the port (P) elements in your circuit, and the specified values for their reference impedances. The `.LIN` command also supports mixed-mode transfer parameters calculation and group delay analysis when used together with mixed-mode P elements.

To calculate the insertion and return loss for the high speed differential signal on my PCB board you can use the `.LIN` command with a port (P) element at input and output, where Port=1 defines the input and Port=2 defines the output. The return loss in dB is  $|S_{111}(DB)|$  and the insertion loss in dB is  $|S_{21}(DB)|$ .

By default, the `.LIN` command creates a `.sc#` file with the same base name as your netlist. This file contains S-parameter, noise parameter, and group delay data as a function of the frequency. You can use this file as model data for the S-element. Noise contributor tables are generated for every frequency point and every circuit device. The last four arguments allow users to better control the output information. If the `LIST*` arguments are not set, `.LIN` 2port noise analysis will output to `./s` file with the older format. If any of the `LIST*` arguments is set, the output information follows the syntax noted in the arguments section.

### Examples

This example extracts linear transfer parameters for a general multiport network, performs a 2-port noise analysis and a group-delay analysis for a model named `my_custom_model`. The output is in the `mydesign` Touchstone format output file. The data format in the Touchstone file is real-imaginary.

```
.LIN sparcalc=1 modelname=my_custom_model
+ filename=mydesign format=touchstone noisecalc=1
+ gdcalc=1 dataformat=ri
```

### See Also

[Filters Examples](#), *fbpnet.sp*, for a bandpass LCR filter demo using the `.LIN` command

---

## .LOAD

Uses the operating point information of a file previously created with a `.SAVE` command.

### Syntax

```
.LOAD [FILE=load_file] [RUN=PREVIOUS|CURRENT]
```

---

Argument	Description
<i>load_file</i>	Name of the file in which <code>.SAVE</code> saved an operating point for the circuit under simulation. The format of the file name is <i>design.ic#</i> . Default is <i>design.ic0</i> , where <i>design</i> is the root name of the design.
<i>RUN</i>	The format of file name is <i>design.ic#</i> . Used only outside of <code>.ALTER</code> commands in a netlist that contains <code>.ALTER</code> commands. <ul style="list-style-type: none"> <li>▪ PREVIOUS: Each <code>.ALTER</code> uses the saved operating point from the previous <code>.ALTER</code> run in the current simulation run.</li> <li>▪ CURRENT: Each <code>.ALTER</code> uses the saved operating point from the corresponding <code>.ALTER</code> run in the previous simulation run.</li> </ul>

---

### Description

Use this command to input the contents of a file that you stored using the `.SAVE` command. Files stored with the `.SAVE` command contain operating point information for the point in the analysis at which you executed `.SAVE`.

Do not use the `.LOAD` command for concatenated netlist files.

This command can be used as part of a compressed (*.gzip*) netlist file.

### Examples

Example 1 loads a file name *design.ic0*, which you previously saved using a `.SAVE` command.

#### Example 1

```
.SAVE FILE=design.ic0
.LOAD FILE=design.ic0
        $load--design.ic0 save--design.ic0
.alter
...      $load--none    save--design.ic1
.alter
...      $load--none    save--design.ic2
.end
```



Example 2 runs a previously saved and loaded design.

*Example 2*

```
.SAVE FILE=design.ic
.LOAD FILE=design.ic RUN=PREVIOUS
      $load--none    save--design.ic0
.alter
...      $load--design.ic0 save--design.ic1
.alter
...      $load--design.ic1 save--design.ic2
.end
```

Example 3 runs the current design.

*Example 3*

```
.SAVE FILE=design.ic
.LOAD FILE=design.ic RUN=CURRENT
      $load--design.ic0 save--design.ic0
.alter
...      $load--design.ic1 save--design.ic1
.alter
...      $load--design.ic2 save--design.ic2
.end
```

**See Also**

[.ALTER](#)  
[.SAVE](#)

---

## .LPRINT

Produces output in VCD file format from transient analysis in HSPICE RF.  
(Valid only for HSPICE RF.)

### Syntax

```
.LPRINT (v1,v2) output_variable_list
```

---

Argument	Description
v1, v2	Threshold values for digital output. Values less than v1 are output as digital 0. Values greater than 1 are output as digital 1.
output_variable_list	Output variables to .PRINT. These are variables from a DC, AC, TRAN, or NOISE analysis).

---

### Description

Use this command to produce output in VCD file format from transient analysis.

### Examples

In this example, the .LPRINT command sets threshold values to 0.5 and 4.5, and the voltage level at voltage source VIN.

```
.OPTION VCD
.LPRINT (0.5,4.5) v(VIN)
```

### See Also

[.PRINT](#)

---

## .LSTB

Invokes linear loop stability analysis.

### Syntax

```
Vxxx drv fbk 0
.LSTB mode=[single|diff|comm]
+ vsource=[vlstb|vlstbp,vlstbn]
.PRINT|PROBE AC
+ LSTB|LSTB(DB)|LSTB(M)|LSTB(P)|LSTB(R)|LSTB(I)
```

Argument	Description
Vxxx	The 0V voltage source(s) indicating the insertion point of test circuit. Note that the direction of Vxxx is of significance in diff/comm mode analysis. <ul style="list-style-type: none"> <li>drv: Driving node (i.e., input of amplifier)</li> <li>fbk: Feedback node (i.e., output of amplifier)</li> </ul>
mode	<ul style="list-style-type: none"> <li>Single: (default) single-ended test.</li> <li>Diff: differential mode test.</li> <li>Comm: common mode test.</li> </ul>
Vsource	<ul style="list-style-type: none"> <li>Vlstb: The only one vsource for single-ended mode test.</li> <li>Vlstbp: One of the two vsources for differential or common mode test.</li> <li>Vlstpn: The other one of the two vsources for differential or common mode test.</li> </ul>
LSTB	Output all results: dB, magnitude, phase, real and imaginary part of loop gain.
LSTB(x)	<ul style="list-style-type: none"> <li>x=DB: Output the dB values of loop gain.</li> <li>X=M: Output magnitude of loop gain.</li> <li>X=P: Output phase of loop gain.</li> <li>X=R: Output real part of loop gain.</li> <li>X=I: Output imaginary part of loop gain.</li> </ul>

---

### Description

The .LSTB command measures the loop gain by successive injection (Middlebrook's Technique). A 0V voltage source is placed in series in the loop: one pin of the voltage loop must be connected to the loop input, the other pin to the loop output. The orientation of inserted voltage sources in differential and common-mode testing is significant. It is required that the positive terminal of both voltage sources go to the input of amplifier or go to the output of amplifier. The first 3 characters of the mode type are effective (*sin*, *dif* or *com*). For single-

ended (default mode) test: place one 0V DC voltage source in series and specify its name in the loop of interest, then add the `.LSTB` statement and specify `single` as mode. For differential and common-mode loop analysis, set `diff` or `comm` as the mode and specify the names of two 0V DC voltage sources.

`.MEASURE` statements are supported similar to any ac output variables. The feature can be used with `.ALTER` to generate multiple loop analyses. (See examples below.)

The outputs for loop stability analysis are as follows:

- The gain margin (GM), phase margin (PM), unity gain frequency (FU) and gain at minimum frequency (ADC) are reported in the `*.lis` file.
- The Loop Gain is reported to the `*.cx#` file, which is always produced for `.LSTB` analysis. The `*.cx#` file is a general file for all the complex number outputs. It contains the data for waveforms as complex vectors.
- If you specify `.probe ac lstb(db) lstb(mag) lstb(real) lstb(imag) lstb(phase)`, the specific format of loop gain goes to the `*.ac#` file for viewing.
- If an `*.ac#` file is produced with `.probe ac lstb`, then both `*.ac#` and `*.cx#` file could be used to view magnitude, phase, real, and imaginary versus frequency as complex vectors.

Considerations regarding loop stability analysis include the following:

- `.LSTB` analysis is based on a linearized circuit at a given DC operating point. It does not guarantee a stable condition for large signal condition. As a final stability check, designers should perform transient analysis; i.e., inject a slow sinusoid superimposed with a series of fast pulses into the loop; the amount of ringing indicates the degree of stability for the circuit.
- All other independent AC voltage sources are disabled automatically when `.LSTB` is enabled.
- `.OPTION UNWRAP` is set to 1 and if phase wrapping is found, the phase is corrected by 180 degrees.
- If phase/gain margin is not found in the given ac analysis frequency range, a warning message is issued.

## Examples

**Example 1** *This example shows a sample portion of a netlist where the first two lines are the 0 voltage sources indicating the insertion point of circuit under test; line 3 sets the .LSTB analysis using the differential mode and specifies the two vsources; line 4 sets the .AC analysis, and the last three lines specify post-processing (printing, plotting, and measurements).*

```
V1 n1 n2 0
V2 n3 n4 0
.LSTB mode=diff vsource=v1,v2
.AC DEC 10 1K 1MEG
.PRINT AC LSTB(DB) LSTB(M) LSTB(P) LSTB(R) LSTB(I)
.PROBE AC LSTB(DB) LSTB(M) LSTB(P) LSTB(R) LSTB(I)
.MEASURE AC phase_margin FIND LSTB(P) when LSTB(DB)=0
```

**Example 2** *The .MEASURE statement is supported such as any common ac output variable.*

```
.measure ac phase_margin FIND lstb(P) when lstb(db)=0
.measure ac integ1 INTEGRAL lstb(P) FROM=1k TO=100k
```

**Example 3** *In this example, the lstb scalars are measured in which lstb is the type name, out1-out4 are names for output, followed by scalar variable keywords:*

```
.measure lstb out1 gain_margin
.measure lstb out2 phase_margin
.measure lstb out3 unity_gain_freq
.measure lstb out4 loop_gain_minifreq
```

**Example 4** *A series of loop stability analyses are supported by the .alter command.*

```
V3 n3 n4 0
.lstb mode=single vsource=V3
.alter
V4 n5 n6 0
V5 n7 n8 0
.lstb mode=common vsource=v4, v5
```

## See Also

- [.AC](#)
- [.ALTER](#)
- [.MEASURE LSTB](#)
- [.PRINT](#)
- [.PROBE](#)
- [.OPTION UNWRAP](#)

[Using .LSTB for Loop Stability Analysis](#)

---

## .MACRO

Defines a subcircuit in your netlist.

### Syntax

```
.MACRO subckt_namen1 [n2n3...] [parnam=val]  
.EOM
```

---

Argument	Description
subckt_nam	reference name for the subcircuit model call.
n1 ...	Node numbers for external reference; cannot be the ground node (zero). Any element nodes that are in the subcircuit, but are not in this list strictly local with three exceptions: <ul style="list-style-type: none"><li>▪ Ground node (zero).</li><li>▪ Nodes assigned using BULK=node in MOSFET or BJT models.</li><li>▪ Nodes assigned using the .GLOBAL command.</li></ul>
parnam	Parameter name set to a value. Use only in the subcircuit. To override this value, assign it in the subcircuit call or set a value in a .PARAM command.
SubDefaultsList	<i>SubParam1=Expression</i> <i>[SubParam2=Expression...]</i>

---

### Description

Use this command to define a subcircuit in your netlist (effectively the same as the .SUBCKT command). You can create a subcircuit description for a commonly used circuit and include one or more references to the subcircuit in your netlist. Use the .EOM command to terminate a .MACRO command.

### Examples

Example 1 defines two subcircuits: SUB1 and SUB2. These are resistor divider networks, whose resistance values are parameters (variables). The X1, X2,

and X3 commands call these subcircuits. Because the resistor values are different in each call, these three calls produce different subcircuits.

### Example 1

```
*FILE SUB2.SP TEST OF SUBCIRCUITS
.OPTION LIST ACCT
  V1 1 0 1
.PARAM P5=5 P2=10
.SUBCKT SUB1 1 2 P4=4
  R1 1 0 P4
  R2 2 0 P5
  X1 1 2 SUB2 P6=7
  X2 1 2 SUB2
.ENDS
*
.MACRO SUB2 1 2 P6=11
  R1 1 2 P6
  R2 2 0 P2
.EOM
  X1 1 2 SUB1 P4=6
  X2 3 4 SUB1 P6=15
  X3 3 4 SUB2
*
.MODEL DA D CJA=CAJA CJP=CAJP VRB=-20 IS=7.62E-18
+ PHI=.5 EXA=.5 EXP=.33
.PARAM CAJA=2.535E-16 CAJP=2.53E-16
.END
```

Example 2 implements an inverter that uses a *Strength* parameter. By default, the inverter can drive three devices. Enter a new value for the *Strength* parameter in the element line to select larger or smaller inverters for the application.

### Example 2

```
.SUBCKT Inv a y Strength=3
  Mp1 <MosPinList> pMosMod L=1.2u W='Strength * 2u'
  Mn1 <MosPinList> nMosMod L=1.2u W='Strength * 1u'
.ENDS
...
xInv0 a y0 Inv          $ Default devices: p device=6u,
  $ n device=3u
xInv1 a y1 Inv Strength=5      $ p device=10u, n device=5u
xInv2 a y2 Inv Strength=1      $ p device= 2u, n device=1u
...
```

**See Also**

[.ENDS](#)  
[.EOM](#)  
[.SUBCKT](#)



---

## .MALIAS

Assigns an alias to a diode, BJT, JFET, or MOSFET model that you defined in a .MODEL command.

### Syntax

```
.MALIAS model_name=alias_name1 [alias_name2 ...]
```

---

Argument	Description
model_name	Model name defined in the .MODEL card
alias_name1...	Alias that an instance (element) of the model references

---

### Description

Use this command to assign an alias (another name) to a diode, BJT, JFET, or MOSFET model that you defined in a .MODEL command.

.MALIAS differs from .ALIAS in two ways:

- A model can define the alias in an .ALIAS command, but not the alias in a .MALIAS command. The .MALIAS command applies to an element (an instance of the model), not to the model itself.
- The .ALIAS command works only if you include .ALTER in the netlist. You can use .MALIAS without .ALTER.

You can use .MALIAS to alias to a model name that you defined in a .MODEL command or to alias to a subcircuit name that you defined in a .SUBCKT command. The syntax for .MALIAS is the same in either usage.

**Note:** The .MALIAS command is supported for diode, BJT, JFET, and MOSFET models in .Global\_Variation and .Local\_Variation blocks.

#### Examples

- zendef is a diode model
- zen and zend are its aliases.
- The zendef model points to both the zen and zend aliases.

```
*file: test malias statement
.OPTION acct tnom=50 list gmin=1e-14 post
.temp 0.0 25
.tran .1 2
vdd 2 0 pw1 0 -1 1 1
d1 2 1 zend dtemp=25
d2 1 0 zen dtemp=25
* malias statements
.malias zendef=zen zend
* model definition
.model zendef d (vj=.8 is=1e-16 ibv=1e-9 bv=6.0 rs=10
+ tt=0.11n n=1.0 eg=1.11 m=.5 cjo=1pf tref=50)
.end
```

#### See Also

[.ALIAS](#)  
[.MODEL](#)

## .MATERIAL

Specifies material to be used with the HSPICE field solver.

### Syntax

```
.MATERIAL mname METAL|DIELECTRIC [ER=val]  
+ [UR=val] [CONDUCTIVITY=val] [LOSSTANGENT=val]  
+ ROUGHNESS=val
```

Argument	Description
<i>mname</i>	Material name.
METAL DIELECTRIC	Material type: METAL or DIELECTRIC.
ER	Dielectric constant (relative permittivity).
UR	Relative permeability.
CONDUCTIVITY	Static field conductivity of conductor or lossy dielectric (S/m).
LOSSTANGENT	Alternating field loss tangent of dielectric ( $\tan \delta$ ).
ROUGHNESS	RMS surface roughness height, used when scaling the field solver.

### Description

The field solver assigns the following default values for metal:  
 CONDUCTIVITY=-1 (perfect conductor), ER=1, UR=1.

PEC (perfect electrical conductor) is a predefined metal name. You cannot redefine its default values. The field solver assigns default values for dielectrics:

- CONDUCTIVITY=0 (lossless dielectric)
- LOSSTANGENT=0 (lossless dielectric)
- ER=1
- UR=1

AIR is a predefined dielectric name. You cannot redefine its default values. Because the field solver does not currently support magnetic materials, it ignores UR values.

**Chapter 2: HSPICE and HSPICE RF Netlist Commands**  
**.MATERIAL**

**See Also**

[.LAYERSTACK](#)

[.FSOPTIONS](#)

[Transmission \(W-element\) Line Examples](#)

---

## **.MEASURE (or) .MEAS**

Modifies information to define the results of successive simulations.

### **Syntax**

See the links below for the various syntaxes.

### **Description**

Use this command to modify information and to define the results of successive HSPICE simulations. The `.MEASURE` command prints user-defined electrical specifications of a circuit. Optimization uses `.MEASURE` commands extensively. You can shorten the command name to `.MEAS`. The specifications include:

- Propagation
- Delay
- Rise time
- Fall time
- Peak-to-peak voltage
- Minimum and maximum voltage over a specified period
- Other user-defined variables

You can also use `.MEASURE` with either the error function (`ERRfun`) or `GOAL` parameter to optimize circuit component values, and to curve-fit measured data to model parameters.

The `.MEASURE` command can use several different formats, depending on the application. You can use it for DC sweep, and AC or transient analyses.

**Note:** If a `.measure` command uses the result of previous `.meas` command, then the calculation starts when the previous result is found. Until the previous result is found, it outputs zero.

### **Examples**

To measure the difference between two different nodes in a dc analysis:

```
.MEAS dc V1 MAX V(1)
.MEAS dc V2 MAX V(2)
.MEAS VARG PARAM="(V2 - V1)"
```

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

.MEASURE (or) .MEAS

### See Also

.MEASURE (Rise, Fall, Delay, and Power Measurements)  
.MEASURE (FIND and WHEN)  
.MEASURE (Equation Evaluation/Arithmetic Expression)  
.MEASURE (AVG, EM\_AVG, INTEG, MIN, MAX, PP, and RMS)  
.MEASURE (Integral Function)  
.MEASURE (Derivative Function)  
.MEASURE (Error Function)  
.MEASURE (Pushout Bisection)  
.MEASURE (ACMATCH)  
.MEASURE (DCMATCH)  
.MEASURE FFT  
.MEASURE LSTB  
.AC  
.DC  
.DCMATCH  
.DOUT  
  
.OPTION NCWARN  
.OPTION MEASFAIL  
.OPTION MEASFILE  
.OPTION MEASOUT  
.PRINT  
.PROBE  
.STIM  
.TRAN  
Measuring the Value of MOSFET Model Card Parameters

## .MEASURE (Rise, Fall, Delay, and Power Measurements)

Measures independent-variable differentials such as rise time, fall time, and slew rate.

### Syntax

#### Trigger and Target Subcommands

The following are parameters for the TRIG and TARG subcommands.

```
.MEASURE [DC|AC|TRAN] ResultName TRIG TrigSpec TARG TargSpec
+ [GOAL=val] [MINVAL=val] [WEIGHT=val] [PRINT 0|1]
```

The input syntax for delay, rise time, and fall time in HSPICE RF is:

```
.MEASURE [TRAN] varnameTRIG_ SPECTARG_ SPEC
```

In this syntax, *varname* is the user-defined variable name for the measurement (the time difference between TRIG and TARG events). The input syntax for *TRIG\_SPEC* and *TARG\_SPEC* is:

```
TRIG var VAL=val [TD=time] [CROSS=c|LAST]
+ [RISE=r|LAST] [FALL=f|LAST] [TRIG AT=time]
TARG var VAL=val [TD=time-delay] [CROSS=c|LAST]
+ [RISE=r|LAST] [FALL=f|LAST] [REVERSE] [TARG AT=time]
```

Argument	Description
DC   AC   TRAN	Analysis type of the measurement. If you omit this parameter, HSPICE uses the last analysis mode that you requested.
result	Name associated with the measured value in the HSPICE output, can be up to 16 characters long. This example measures the independent variable, beginning at the trigger and ending at the target: <ul style="list-style-type: none"> <li>▪ Transient analysis measures time.</li> <li>▪ AC analysis measures frequency.</li> <li>▪ DC analysis measures the DC sweep variable.</li> </ul> If simulation reaches the target before the trigger activates, the resulting value is negative. Do not use DC, TRAN, or AC as the <i>result</i> name.
TRIG	Beginning of trigger specifications.

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .MEASURE (Rise, Fall, Delay, and Power Measurements)

Argument	Description
TARG	Beginning of the target specification.
TrigSpec	<i>OutputVar</i> VAL={ <i>Numeric</i>   <i>Expression</i> } [TD={ <i>Numeric</i>   <i>Expression</i> }] where: <i>NumericExpression</i> = { <i>FloatingPointNumber</i>   <i>AlgebraicExpression</i> } See <a href="#">Using Algebraic Expressions</a> for information on algebra in output statements.
TargSpec	<i>OutputVar</i> VAL={ <i>Numeric</i>   <i>Expression</i> } [TD={ <i>Number</i>   <i>Expression</i> }] where: <i>NumericExpression</i> := { <i>FloatingPointNumber</i>   <i>AlgebraicExpression</i> } See <a href="#">Using Algebraic Expressions</a> for information on algebra in output statements. If a time-delay is not specified for the Target, the TD is inherited from the Trigger value.
GOAL	Desired measure value in ERR calculation for optimization. To calculate the error, the simulation uses the equation: $ERR_{fun} = (GOAL - result) / GOAL.$
MINVAL	If the absolute value of GOAL is less than MINVAL, the MINVAL replaces the GOAL value in the denominator of the ERRfun expression. Used only in ERR calculation for optimization. The default is 1.0e-12.
WEIGHT	Multiplies the calculated error by the weight value. Used only in ERR calculation for optimization. The default is 1.0.
PRINT	<ul style="list-style-type: none"><li>▪ print=0 prevents the printing a measure result into the measure output file</li><li>▪ print=1 (Default) prints the measure result into the output file</li></ul>
trig_var	Value of <i>trig_var</i> , which increments the counter by one for crossings, rises, or falls. See <a href="#">Using Algebraic Expressions</a> for information on algebra in output statements.
trig_var	Specifies the name of the output variable that determines the logical beginning of a measurement. If HSPICE reaches the target before the trigger activates, .MEASURE reports a negative value. See <a href="#">Using Algebraic Expressions</a> for information on algebra in output statements.
REVERSE	Allows trigger time to come before target time when added to the command.



Argument	Description
TD	Amount of simulation time that must elapse before HSPICE enables the measurement. Simulation counts the number of crossings, rises, or falls only after the <i>time_delay</i> value. Default trigger delay is zero. If a time-delay is not specified for the Target, the TD is inherited from the Trigger value.
AT=val	Special case for trigger specification. <i>val</i> is: <ul style="list-style-type: none"> <li>▪ Time for TRAN analysis.</li> <li>▪ Frequency for AC analysis.</li> <li>▪ Parameter for DC analysis.</li> <li>▪ SweepValue from .DC mismatch analysis.</li> </ul> The trigger determines where measurement takes place.

### Description

Use the Rise, Fall, and Delay form of the .MEASURE command to measure independent-variable (time, frequency, or any parameter or temperature) differentials such as rise time, fall time, slew rate, or any measurement that requires determining independent variable values. This format specifies TRIG and TARG subcommands. These two commands specify the beginning and end of a voltage or current amplitude measurement.

### Examples

*Example 1* HSPICE automatically measures  $T_{prop}$  using the .MEASURE command. This reference file contains .MEAS commands for rising edge and falling edge measurements. The time delay is measured and saved during simulation in an \*.mt0 file. Note that if a falling edge simulation is run, the rising edge measurements are invalid. Similarly, if a rising edge simulation is run, the falling edge measurements are invalid. (Remember this when referring to the \*.mt0 file after a simulation.) In this sample file, .MEASURE statements are provided to measure  $T_{prop}$  from the ref\_50pf waveform to each of ten loads. Since each load is measured, the worst-case  $T_{prop}$  for a given configuration can be quickly determined by finding the largest value. The .MEASURE commands work by “triggering” on the ref\_50pf signal as it crosses 1.5 volts, and ending the measurement when the “target” waveform, crosses the specified voltage

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .MEASURE (Rise, Fall, Delay, and Power Measurements)

*for the last time. For rising edge measurements, this value is 2.0 Volts. For falling edge measurements, the value is 0.8 Volt. Examples from a sample file are listed here.*

```
*****
*           Rising edge T_prop measurements           *
*****
.MEAS tran tr1_val TRIG V(ref_50pf) val=1.5v td='per/2' cross=1
+ TARG V(load1) val=2.0v rise=last
.MEAS tran tr2_val TRIG B(ref_50pf) val=1.5v td='per/2' cross=1
+TARG V(load2) val=2.0v rise=last
.
.
.
.MEAS tran tr10_val TRIG V(ref_50pf) val=1.5v td='per/2' cross=1
+ TARG V(load10) val=2.0v rise=last
*****
*           Falling edge T_prop measurements           *
*****
.MEAS tran tf1_val TRIG V(ref_50pf) val=1.5v td='per/2' cross=1
+TARG V(load1) val=0.8v fall=last
.
.
.
.MEAS tran tf10_val TRIG V(ref_50pf) vbal=1.5v td='per/2' cross=1
+ TARG V(load10) val=0.8v fall=last
```

**Example 2** Measures the propagation delay between nodes 1 and 2 for a transient analysis. HSPICE measures the delay from the second rising edge of the voltage at node 1 to the second falling edge of node 2. Measurement begins when the second rising voltage at node 1 is 2.5 V and ends when the second falling voltage at node 2 is 2.5 V. The TD=10n parameter counts the crossings after 10 ns have elapsed. HSPICE prints results as tdlay=value.

```
* Example of rise/fall/delay measurement
.MEASURE TRAN tdlay TRIG V(1) VAL=2.5 TD=10n
+ RISE=2 TARG V(2) VAL=2.5 FALL=2
```

In Example 3, `TRIG AT=10n` starts measuring time at  $t=10$  ns in the transient analysis. The `TARG` parameters terminate time measurement when  $V(IN) = 2.5$  V on the third crossing. `pwidth` is the printed output variable.

If you use the `.TRAN` analysis command with a `.MEAS` command, do not use a non-zero start time in the `.TRAN` command to avoid incorrect `.MEAS` results.

#### Example 3

```
.MEASURE TRAN riset TRIG I(Q1) VAL=0.5m RISE=3
+ TARG I(Q1) VAL=4.5m RISE=3
* Rise/fall/delay measure with TRIG and TARG specs
.MEASURE pwidth TRIG AT=10n TARG V(IN) VAL=2.5
+ CROSS=3
```

Example 4 shows a target delayed until the trigger time before the target counts the edges.

#### Example 4

```
.MEAS TRAN TDEL12 TRIG V(signal1) VAL='VDD/2'
+ RISE=10 TARG V(signal2) VAL='VDD/2' RISE=1 TD=TRIG
```

By using the `cross` keyword, Example 4 calculates the final settled value when you don't know how many times the signal crosses the final value.

#### Example 5

```
.meas tran tim2 when v(out)='final_value' cross=last
```

In Example 6, `print=0` prevents the printing of  $V_{max}$  to the `*.mt0` and `*.lis` files. Delay will be output into `*.mt#` file and `*.lis` file.

#### Example 6

```
.meas tran Vmax max v(out) print=0
.meas tran delay trig V(in) val='vmax/2' rise=1 targ v(out)
+ val='vmax/2' rise=1
```

#### See Also

[.OPTION AUTOSTOP](#) (or) [.OPTION AUTOST](#)  
[Filters Examples](#), `fbp_1.sp`, for a bandpass LCR filter measurement demo netlist

---

## .MEASURE (FIND and WHEN)

Measures independent and dependent variables (as well as derivatives of dependent variables if a specific event occurs).

### Syntax

```
.MEASURE [DC|AC|TRAN] result WHEN out_var=val [TD=val]
+ [FROM=val] [TO=val]
+ [RISE=r|LAST] [FALL=f|LAST] [CROSS=c|LAST] [REVERSE]
+ [GOAL=val] [MINVAL=val] [WEIGHT=val] [PRINT 0|1]
```

```
.MEASURE [DC|AC|TRAN] result
+ WHEN out_var1=out_var2 [TD=val] [RISE=r|LAST]
+ [FALL=f|LAST] [CROSS=c|LAST] [GOAL=val] [MINVAL=val]
+ [WEIGHT=val] [PRINT 0|1]
```

```
.MEASURE [DC|AC|TRAN] result FIND out_var1
+ WHEN out_var2=val [TD=val] [FROM=val] [TO=val]
+ [RISE=r|LAST] [FALL=f|LAST] [CROSS=c|LAST] [REVERSE]
+ [GOAL=val] [MINVAL=val] [WEIGHT=val] [PRINT 0|1]
```

```
.MEASURE [DC|AC|TRAN] result FIND out_var1
+ WHEN out_var2=out_var3 [TD=val]
+ [RISE=r|LAST] [FALL=f|LAST] [REVERSE] [CROSS=c|LAST]
+ [GOAL=val] [MINVAL=val] [WEIGHT=val] [PRINT 0|1]
```

```
.MEASURE [DC|AC|TRAN] result FIND out_var1
+ AT=val [FROM=val] [TO=val] [GOAL=val] [MINVAL=val]
+ [WEIGHT=val] [PRINT 0|1]
```

---

Argument	Description
DC   AC   TRAN	Analysis type of the measurement. If you omit this parameter, HSPICE uses the last analysis mode that you requested.
result	Name of a measured value in the HSPICE output.
WHEN	WHEN function.
out_var(1,2,3)	Variables that establish conditions to start a measurement.
TD	Time at which measurement starts.

Argument	Description
FROM... TO...	Allows adding multiple trigger conditions to some WHEN measurements.
CROSS=c RISE=r FALL=f	<p>Numbers indicate which CROSS, FALL, or RISE event to measure. For example:</p> <pre>.meas tran tdlay trig v(1) val=1.5 td=10n + rise=2 targ v(2) val=1.5 fall=2</pre> <p>In this example, rise=2 specifies the measure of the v(1) voltage only on the first two rising edges of the waveform. The value of these first two rising edges is 1. However, trig v(1) val=1.5 indicates to trigger when the voltage on the rising edge voltage is 1.5, which never occurs on these first two rising edges. So the v(1) voltage measurement never finds a trigger.</p> <p>RISE=r, the WHEN condition is met and measurement occurs after the designated signal has risen <i>r</i> rise times.</p> <p>FALL =f, measurement occurs when the designated signal has fallen <i>f</i> fall times.</p> <p>A crossing is either a rise or a fall so for CROSS=c, measurement occurs when the designated signal has achieved a total of <i>c</i> crossing times as a result of either rising or falling.</p> <p>For TARG, the LAST keyword specifies the last event.</p>
LAST	<p>HSPICE measures when the last CROSS, FALL, or RISE event occurs.</p> <ul style="list-style-type: none"> <li>▪ CROSS=LAST, measurement occurs the last time the WHEN condition is true for a rising or falling signal.</li> <li>▪ FALL=LAST, measurement occurs the last time the WHEN condition is true for a falling signal.</li> <li>▪ RISE=LAST, measurement occurs the last time the WHEN condition is true for a rising signal.</li> </ul> <p>LAST is a reserved word; you cannot use it as a parameter name in the above .MEASURE commands.</p>
REVERSE	Allows FALL to precede RISE in specified .MEAS commands, when declared.

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .MEASURE (FIND and WHEN)

---

Argument	Description
GOAL	<p>Desired .MEASURE value. Optimization uses this value in ERR calculation. The following equation calculates the error:</p> $ERR_{fun} = (GOAL - result) / GOAL$ <p>In HSPICE RF output you cannot apply .MEASURE to waveforms generated from another .MEASURE command in a parameter sweep.</p>
MINVAL	<p>If the absolute value of GOAL is less than MINVAL, then MINVAL replaces the GOAL value in the denominator of the ERRfun expression. Used only in ERR calculation for optimization. The default is 1.0e-12.</p>
WEIGHT	<p>Calculated error multiplied by the weight value. Used only in ERR calculation for optimization. The default is 1.0.</p>
PRINT	<ul style="list-style-type: none"><li>▪ print=0 prevents the printing a measure result into the measure output file</li><li>▪ print=1 (Default) prints the measure result into the output file</li></ul>
FIND	<p>FIND function.</p>
AT=val	<p>Special case for trigger specification. <i>val</i> is:</p> <ul style="list-style-type: none"><li>▪ Time for TRAN analysis.</li><li>▪ Frequency for AC analysis.</li><li>▪ Parameter for DC analysis.</li><li>▪ SweepValue from .DC mismatch analysis.</li></ul> <p>The trigger determines where measurement takes place.</p>

---

#### Description

The FIND and WHEN functions of the .MEASURE command measure:

- Any independent variables (time, frequency, parameter).
- Any dependent variables (voltage or current for example).
- A derivative of a dependent variable if a specific event occurs.

## Examples

### Calculating Voltage

In Example 1, the first measurement, TRT, calculates the difference between V(3) and V(4) when V(1) is half the voltage of V(2) at the last rise event.

The second measurement, STIME, finds the time when V(4) is 2.5V at the third rise-fall event. A CROSS event is a rising or falling edge.

#### Example 1

```
* MEASURE statement using FIND/WHEN
.MEAS TRAN TRT FIND PAR('V(3)-V(4)')
+ WHEN V(1)=PAR('V(2)/2') RISE=LAST
.MEAS STIME WHEN V(4)=2.5 CROSS=3
```

Using a DC Sweep Variable: By adding `par()` to the sweep variable it can be used in a `.MEASURE` command.

#### Example 2

```
* sweep measure
v0 1 0 3
r0 1 0 x
.dc x 1 5 1
.meas res find par(x) when i(r0)=2
.end
```

Example 3 calculates capacitance from node to node:

#### Example 3

```
.meas tran pct_5 when v(out)='vddr*0.05' rise=1
.meas tran pct_95 when v(out)='vddr*0.95' rise=1
.meas tran avg_rout_n avg par('v(out)/i(xinv.mn)')
+ from=pct_5 to=pct_95
```

## See Also

[.OPTION AUTOSTOP](#) (or) [.OPTION AUTOST](#)

---

## .MEASURE (Continuous Results)

Measures continuous results for TRIG-TARG and Find-When functions.

### Syntax

```
.MEASURE [DC_CONT|AC_CONT|TRAN_CONT] result TRIG ... TARG ...
+ [GOAL=val] [MINVAL=val] [WEIGHT=val] [PRINT 0|1]
```

```
.MEASURE [DC_CONT|AC_CONT|TRAN_CONT] result
+ WHEN out_var=val [TD=val]
+ [RISE=r | LAST] [FALL=f | LAST] [CROSS=c | LAST]
+ [GOAL=val] [MINVAL=val] [WEIGHT=val] [PRINT 0|1]
```

```
.MEASURE [DC_CONT|AC_CONT|TRAN_CONT] result
+ WHEN out_var1=out_var2 [TD=val]
+ [RISE=r | LAST] [FALL=f | LAST] [CROSS=c | LAST]
+ [GOAL=val] [MINVAL=val] [WEIGHT=val]
```

```
.MEASURE [DC_CONT|AC_CONT|TRAN_CONT] result FIND out_var1
+ WHEN out_var2=val [TD=val] [RISE=r | LAST]
+ [FALL=f | LAST] [CROSS=c | LAST] [GOAL=val]
+ [MINVAL=val] [WEIGHT=val] [PRINT 0|1]
```

```
.MEASURE [DC_CONT|AC_CONT|TRAN_CONT] result FIND out_var1
+ WHEN out_var2=out_var3 [TD=val] [RISE=r | LAST]
+ [FALL=f | LAST] [CROSS=c | LAST] [GOAL=val]
+ [MINVAL=val] [WEIGHT=val] [PRINT 0|1]
```

---

Argument	Description
DC_CONT AC_CONT TRAN_CONT	Analysis type of the continuous measurement.
result	Name of a measured value in the HSPICE output.
TRIG...	Beginning of trigger specifications.
TARG...	Beginning of the target specification.



Argument	Description
GOAL	<p>Desired .MEASURE value. Optimization uses this value in ERR calculation. The following equation calculates the error:</p> $ERRfun = (GOAL - result) / GOAL$ <p>In HSPICE RF output you cannot apply .MEASURE to waveforms generated from another .MEASURE command in a parameter sweep.</p>
MINVAL	<p>If the absolute value of GOAL is less than MINVAL, then MINVAL replaces the GOAL value in the denominator of the ERRfun expression. Used only in ERR calculation for optimization. The default is 1.0e-12.</p>
WEIGHT	<p>Calculated error multiplied by the weight value. Used only in ERR calculation for optimization. The default is 1.0.</p>
PRINT	<ul style="list-style-type: none"> <li>▪ print=0 prevents the printing a measure result into the measure output file</li> <li>▪ print=1 (Default) prints the measure result into the output file</li> </ul>
WHEN	<p>WHEN function.</p>
out_var(1,2,3)	<p>Variables that establish conditions to start a measurement.</p>
TD	<p>Time at which measurement starts.</p>
CROSS=c RISE=r FALL=f	<p>Numbers indicate which CROSS, FALL, or RISE event to measure. For example: <code>.meas tran tdlay trig v(1) val=1.5 td=10n + rise=2 targ v(2) val=1.5 fall=2</code> In this example, rise=2 specifies the measure of the v(1) voltage only on the first two rising edges of the waveform. The value of these first two rising edges is 1. However, trig v(1) val=1.5 indicates to trigger when the voltage on the rising edge voltage is 1.5, which never occurs on these first two rising edges. So the v(1) voltage measurement never finds a trigger. RISE=r, the WHEN condition is met and measurement occurs after the designated signal has risen <i>r</i> rise times. FALL=f, measurement occurs when the designated signal has fallen <i>f</i> fall times. A crossing is either a rise or a fall so for CROSS=c, measurement occurs when the designated signal has achieved a total of <i>c</i> crossing times as a result of either rising or falling. For TARG, the LAST keyword specifies the last event.</p>

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .MEASURE (Continuous Results)

Argument	Description
LAST	<p>HSPICE measures when the last CROSS, FALL, or RISE event occurs.</p> <ul style="list-style-type: none"><li>▪ CROSS=LAST, measurement occurs the last time the WHEN condition is true for a rising or falling signal.</li><li>▪ FALL=LAST, measurement occurs the last time the WHEN condition is true for a falling signal.</li><li>▪ RISE=LAST, measurement occurs the last time the WHEN condition is true for a rising signal.</li></ul> <p>LAST is a reserved word; you cannot use it as a parameter name in the above .MEASURE commands.</p>
FIND	FIND function.

#### Description

Enables HSPICE to give multiple results during the measurement of DC, AC, and transient analysis data. For example, it gives all the time points at which two signals cross each other. Similar to HSIM, this command uses the same syntax. The standalone measure utility also supports this feature. The continuous measurement feature only applies to TRIG-TARG and Find-When functions.

#### Examples

In Example 1, the `.measure` statement will continuously find the voltage `out1` when the voltage value of node `a1` reaches 2.5 starting from the first falling edge.

##### *Example 1*

```
.measure tran_cont vout1 find v(out1) when v(a1)=2.5 fall=1
```

In Example 2, the `.measure` statement will continuously report the time when the voltage value of node `a1` reaches 2.5V, starting from the second falling edge.

##### *Example 2*

```
.measure tran_cont cont_vout1 when v(a1)=2.5 fall=2
```

#### See Also

[.OPTION AUTOSTOP \(or\) .OPTION AUTOST](#)

## .MEASURE (Equation Evaluation/Arithmetic Expression)

Evaluates an equation that is a function of the results of previous .MEASURE commands.

### Syntax

```
.MEASURE [DC|TRAN|AC] result PARAM='equation'  
+ [GOAL=val] [MINVAL=val] [PRINT 0|1]  
.MEASURE TRAN varname PARAM="expression"
```

Argument	Description
DC   AC   TRAN	Analysis type of the measurement. If you omit this parameter, HSPICE uses the last analysis mode that you requested.
result	Name of a measured value in the HSPICE output.
PARAM='equation'	Equation wrapped in single quotes, a function of the results of previous .MEASURE commands.
GOAL	Desired .MEASURE value. In HSPICE RF output you cannot apply .MEASURE to waveforms generated from another .MEASURE command in a parameter sweep.
MINVAL	If the absolute value of GOAL is less than MINVAL, then MINVAL replaces the GOAL value in the denominator of the ERRfun expression. Used only in ERR calculation for optimization. The default is 1.0e-12.
TRAN	Transient analysis results.
varname	Name of variable to be used in evaluation.
PARAM="expression"	Arithmetic expression that uses results from other prior .MEASURE commands.

### Description

Use the Equation Evaluation form of the .MEASURE command to evaluate an equation that is a function of the results of previous .MEASURE commands. The equation must not be a function of node voltages or branch currents.

The *expression* option is an arithmetic expression that uses results from other prior .MEASURE commands.

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .MEASURE (Equation Evaluation/Arithmetic Expression)

Expressions used in arithmetic expression must not be a function of node voltages or branch currents. Expressions used in all other .MEASURE commands can contain either node voltages or branch currents, but must not use results from other .MEASURE commands.

When using formulas in a .MEAS command, use the `PAR ( )` keyword to designate the formula.

#### Examples

In Example 1, the first two measurements, `V3MAX` and `V2MIN`, set up the variables for the third .MEASURE command.

- `V3MAX` is the maximum voltage of `V(3)` between 0ns and 100ns of the simulation.
- `V2MIN` is the minimum voltage of `V(2)` during that same interval.
- `VARG` is the mathematical average of the `V3MAX` and `V2MIN` measurements.

#### Example 1

```
.MEAS TRAN V3MAX MAX V(3) FROM 0NS TO 100NS  
.MEAS TRAN V2MIN MIN V(2) FROM 0NS TO 100NS  
.MEAS VARG PARAM='(V2MIN + V3MAX)/2'
```

Example 2 illustrates use of the `par ( )` keyword to measure the integral of a formula.

#### Example 2

```
.meas i1 integ par('v(a)+v(b)')
```

## .MEASURE (AVG, EM\_AVG, INTEG, MIN, MAX, PP, and RMS)

Reports statistical functions of the output variable (voltage, current, or power).

### Syntax

```
.MEASURE [DC|AC|TRAN] result func out_var
+ [FROM=val] [TO=val] [GOAL=val] MINVAL=val]
+ [WEIGHT=val] [PRINT 0|1]
```

Argument	Description
DC AC TRAN	Analysis type for the measurement. If you omit this parameter, HSPICE defaults to the last analysis mode that you requested.
result	Name of the measured value in the output, can be up to 16 characters long. The value is a function of the variable ( <i>out_var</i> ) and <i>func</i> .
func	Indicates one of the following measure function types: <ul style="list-style-type: none"> <li>▪ AVG (average): Calculates the area under the <i>out_var</i>, divided by the periods of interest.</li> <li>▪ INTEG (Integral function): Reports the integral of an output variable over a specified period.</li> <li>▪ MIN (minimum): Reports the minimum value of the <i>out_var</i> over the specified interval.</li> <li>▪ MAX (maximum): Reports the maximum value of the <i>out_var</i> over the specified interval.</li> <li>▪ PP (peak-to-peak): Reports the maximum value, minus the minimum value of the <i>out_var</i> over the specified interval.</li> <li>▪ RMS (root mean squared): Calculates the square root of the area under the <i>out_var</i><sup>2</sup> curve, divided by the period of interest.</li> <li>▪ EM_AVG: Calculates the average electromigration current. For a symmetric bipolar waveform, the current is: <math>I_{avg}(0, T/2) - R \cdot I_{avg}(T/2, T)</math>, where R is the recovery factor specified using <code>.option em_recovery</code>. Wildcards are also supported during this measurement.</li> </ul>
out_var	Name of any output variable whose function ( <i>func</i> ) the simulation measures (voltage, current, or power). An output variable can be any dependent variable (voltage, current, or power).
FROM	Initial value for the INTEG calculation.

**Chapter 2: HSPICE and HSPICE RF Netlist Commands**  
**.MEASURE (AVG, EM\_AVG, INTEG, MIN, MAX, PP, and RMS)**

Argument	Description
TO	End of the INTEG calculation.
GOAL	.MEASURE value. Optimization uses this value for ERR calculation. This equation calculates the error:  $ERRfun = (GOAL - result) / GOAL$ In HSPICE RF simulation output you cannot apply .MEASURE to waveforms generated from another .MEASURE command in a parameter sweep.
WEIGHT	Calculated error multiplied by the weight value. Used only in ERR calculation for optimization. The default is 1.0.
PRINT	<ul style="list-style-type: none"> <li>▪ print=0 prevents the printing a measure result into the measure output file</li> <li>▪ print=1 (Default) prints the measure result into the output file</li> </ul>

**Description**

Average (AVG), EM\_AVG, RMS, MIN, MAX, and peak-to-peak (PP) measurement modes report statistical functions of the output variable, rather than analysis values. Output variables are voltage, current, or power. Wildcards are supported for the From-To functions for AVG, EM\_AVG, RMS, MIN, MAX and PP measurement (unlike other measurement functions).

AVG, RMS, and INTEG have no meaning in a DC data sweep so if you use them, HSPICE issues a warning message.

**Examples**

*Example 1*    *Calculates the average nodal voltage value for node 10 during the transient sweep from the time 10ns to 55ns. It prints out the result as avgval*

```
.MEAS TRAN avgval AVG V(10) FROM=10ns TO=55ns
```

*Example 2*    *Finds the maximum voltage difference between nodes 1 and 2 for the time period from 15 ns to 100 ns.*

```
.MEAS TRAN MAXVAL MAX V(1,2) FROM=15ns TO=100ns
```

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

.MEASURE (AVG, EM\_AVG, INTEG, MIN, MAX, PP, and RMS)

**Example 3** *The first command finds the minimum voltage difference between nodes 1 and 2 over the time period 15 ns to 100 ns. The second command measures the peak to peak current through transistor M1 from 10ns to 100ns.*

```
.MEAS TRAN MINVAL MIN V(1,2) FROM=15ns TO=100ns
.MEAS TRAN P2PVAL PP I(M1) FROM=10ns TO=100ns
```

**Example 4** *The coefficient value is set by .option em\_recovery=val. The electromagnetic migration average is measured from 5 ns to 10.2 ns.*

```
.option em_recovery=0.2
.measure tran vout_1 EM_AVG v(5) from=5ns to=10.2ns
```

**Example 5** *These commands measure result parameter currents over specified ranges.*

```
.measure tran em1 em_avg i(rload) from=1n to=3.5n
.measure tran em2 em_avg i(rload) from=4n to=9n
```

**Example 6** *Finds the average of all the positive currents (lpos\_avg) from 5ns to 50ns.*

```
.MEASURE TRAN EM_AVG I(OUT) FROM=5N TO=50N
```

**Example 7** *The .MEASURE command calculates the RMS voltage of the OUT node from 0ns to 10ns. It then labels the result RMSVAL.*

```
.MEAS TRAN RMSVAL RMS V(OUT) FROM=0NS TO=10NS
```

**Example 8** *The .MEASURE command finds the maximum current of the VDD voltage supply between 10ns and 200ns. The result is called MAXCUR.*

```
.MEAS MAXCUR MAX I(VDD) FROM=10NS TO=200NS
```

**Example 9** *The .MEASURE command uses the ratio of V(OUT) and V(IN) to find the peak-to-peak value in the interval of 0ns to 200ns.*

```
.MEAS P2P PP PAR('V(OUT)/V(IN)') FROM=0NS TO=200NS
```

**Example 10** *Power measurement supplied by source vdd*

```
.MEAS P(VDD)
```

**Example 11** *Three commands measuring power*

```
.MEAS TRAN avg_cur avg par('-I(vh)')
.MEAS TRAN total_cur integ par('-I(vh)') from=0n to=3n
.MEAS TRAN total_pwr PARAM='total_cur*V(vdda)'
```

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

.MEASURE (AVG, EM\_AVG, INTEG, MIN, MAX, PP, and RMS)

### See Also

[.OPTION AUTOSTOP](#) (or) [.OPTION AUTOST](#)  
[.OPTION EM\\_RECOVERY](#)



## .MEASURE (Integral Function)

Reports the real time integration (instantaneous time integral) of an output variable over a specified period.

### Syntax

```
.MEASURE [DC|AC|TRAN] result INTEG[RAL] out_var
+ [FROM=val] [TO=val] [GOAL=val]
+ [MINVAL=val] [WEIGHT=val] [PRINT 0|1]
```

Argument	Description
DC   AC   TRAN	Analysis type of the measurement. If you omit this parameter, HSPICE uses the last analysis mode that you requested.
result	Name of a measured value in the HSPICE output.
INTEG	Integral function to find an output variable over a specified period.
outvar	Name of any output variable whose function the simulation measures.
FROM	Initial value for the <i>func</i> calculation. For transient analysis, this value is in units of time.
TO	End of the <i>func</i> calculation.
GOAL	Desired .MEASURE value. In HSPICE RF output you cannot apply .MEASURE to waveforms generated from another .MEASURE command in a parameter sweep.
MINVAL	If the absolute value of GOAL is less than MINVAL, then MINVAL replaces the GOAL value in the denominator of the ERRfun expression. Used only in ERR calculation for optimization. The default is 1.0e-12.

### Description

The INTEGRAL function reports the integral of an output variable over a specified period. The INTEGRAL function uses the same syntax as the AVG (average), RMS, MIN, MAX and peak-to-peak (PP) measurement modes.

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .MEASURE (Integral Function)

#### Examples

This example calculates the integral of  $I(\text{cload})$  from 10ns to 100ns.

```
.MEAS TRAN charge INTEG I(cload) FROM=10ns TO=100ns
```

The following .MEASURE command calculates the integral of  $I(R1)$  from 50ns to 200ns.

```
.MEASURE TRAN integ_i INTEGRAL I(r1) FROM=50ns TO=200ns
```

#### See Also

[.OPTION AUTOSTOP](#) (or) [.OPTION AUTOST](#)

## .MEASURE (Derivative Function)

Provides the derivative of an output or sweep variable.

### Syntax

```
.MEASURE [DC|AC|TRAN result DERIV[ATIVE] out_var
+ [FROM=val] [TO=val] AT=val [GOAL=val] [MINVAL=val]
+ [WEIGHT=val] [PRINT 0|1]
```

```
.MEASURE [DC|AC|TRAN] result DERIV[ATIVE] out_var
+ [FROM=val TO=val] WHEN var2=val [RISE=r|LAST]
+ [FALL=f|LAST] [CROSS=c|LAST] [TD=tdval] [GOAL=goalval]
+ [MINVAL=minval] [WEIGHT=val] [PRINT 0|1]
```

```
.MEASURE [DC|AC|TRAN] result DERIV[ATIVE] out_var
+ [FROM=val] [TO=val] WHEN var2=var3 [RISE=r|LAST]
+ [FALL=f|LAST] [CROSS=c|LAST] [TD=tdval]
+ [GOAL=val] [MINVAL=val] [WEIGHT=val] [PRINT 0|1]
```

Argument	Description
DC   AC   TRAN	Analysis type of the measurement. If you omit this parameter, HSPICE uses the last analysis mode that you requested.
result	Name of the measured value in the output.
DERIVATIVE	Derivative function.
out_var	Variable for which HSPICE finds the derivative.
FROM=val TO=val	Specifies a range to measure, such as time window.
var(2,3)	Variables establish that conditions to start a measurement.
AT=val	Value of <i>out_var</i> at which the derivative is found.
GOAL	Specifies the desired .MEASURE value. Optimization uses this value for ERR calculation. This equation calculates the error: $ERR_{fun} = (GOAL - result) / GOAL$ In HSPICE RF output you cannot apply .MEASURE to waveforms generated from another .MEASURE command in a parameter sweep.

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .MEASURE (Derivative Function)

---

Argument	Description
MINVAL	If the absolute value of GOAL is less than MINVAL, MINVAL replaces the GOAL value in the denominator of the ERRfun expression. Used only in ERR calculation for optimization. The default is 1.0e-12.
WEIGHT	Calculates the error between result and GOAL by multiplied by the weight value. Used only in ERR calculation for optimization. The default is 1.0.
WHEN	WHEN function.
RISE=r FALL=f CROSS=c	Numbers indicate which occurrence of a CROSS, FALL, or RISE event starts a measurement. <ul style="list-style-type: none"><li>▪ For RISE=r when the designated signal has risen r rise times, the WHEN condition is met and measurement starts.</li><li>▪ For FALL=f, measurement starts when the designated signal has fallen f fall times.</li><li>▪ A crossing is either a rise or a fall so for CROSS=c, measurement starts when the designated signal has achieved a total of c crossing times as a result of either rising or falling.</li></ul>
LAST	Last CROSS, FALL, or RISE event. <ul style="list-style-type: none"><li>▪ CROSS=LAST, measures the last time the WHEN condition is true for a rising or falling signal.</li><li>▪ FALL=LAST, measures the last time WHEN is true for a falling signal.</li><li>▪ RISE=LAST, measures the last time WHEN is true for a rising signal.</li></ul> LAST is a reserved word; do not use it as a parameter name in the above .MEASURE commands.
TD	Time when measurement starts.

---

#### Description

The DERIV function provides the derivative of:

- An output variable at a specified time or frequency.
- Any sweep variable, depending on the type of analysis.
- A specified output variable when some specific event occurs.

#### Examples

*Example 1* Calculates the derivative of V(out) at 25 ns.

```
.MEAS TRAN slew rate DERIV V(out) AT=25ns
```

*Example 2* Calculates the derivative of  $VP(output)/360.0$  when the frequency is 10 kHz.

```
.MEAS AC delay DERIV 'VP(output)/360.0' AT=10khz
```

*Example 3* Measures the derivative of a nodal waveform.

```
.MEAS DC result find v(in) when deriv v(out)= ...
```

*Example 4* If you plot result from the command you will get the  $dV(out)/dT$  vs Temperature plot.

```
.MEAS DC result deriv v(out) ...
```

*Example 5* Measures and finds when the maximum derivative of a signal occurs. The example shows (1) a probe of the derivative of the signal, (2) the maximum value of the derivative, and (3) when the maximum value of the derivative occurred.

```
.probe dt=deriv("v(out)")  
.meas m0 max par(dt)  
.meas m1 when par(dt)=m0
```

**See Also**

[.OPTION AUTOSTOP](#) (or) [.OPTION AUTOST](#)

---

## .MEASURE (Error Function)

Reports the relative difference between two output variables.

### Syntax

```
.MEASURE [DC|AC|TRAN] result
+ ERRfun meas_varcalc_var
+ [MINVAL=val] [IGNOR|YMIN=val]
+ [YMAX=val] [WEIGHT=val] [FROM=val] [TO=val] [PRINT 0|1]
```

---

Argument	Description
DC AC TRAN	Analysis type for the measurement. If you omit this parameter, HSPICE defaults to the last analysis mode requested.
result	Name of the measured result in the output.
ERRfun	Error function to use: ERR, ERR1, ERR2, or ERR3.
meas_var	Name of any output variable or parameter in the data command. <i>M</i> denotes the <i>meas_var</i> in the error equation.
calc_var	Name of the simulated output variable or parameter in the .MEASURE command to compare with <i>meas_var</i> . <i>C</i> is the <i>calc_var</i> in the error equation.
MINVAL	If the absolute value of <i>meas_var</i> is less than <i>MINVAL</i> , <i>MINVAL</i> replaces the <i>meas_var</i> value in the denominator of the <i>ERRfun</i> expression. Used only in ERR calculation for optimization. Default: 1.0e-12.
IGNOR YMIN	If the absolute value of <i>meas_var</i> is less than the <i>IGNOR</i> value, then the <i>ERRfun</i> calculation does not consider this point. Default: 1.0e-15.
YMAX	If the absolute value of <i>meas_var</i> is greater than the <i>YMAX</i> value, then the <i>ERRfun</i> calculation does not consider this point. Default: 1.0e+15.
WEIGHT	Calculates error multiplied weight value. Used only in ERR calculation for optimization. The default is 1.0.
FROM	Specifies the beginning of the <i>ERRfun</i> calculation. For transient analysis, the <i>FROM</i> value is in units of time. Defaults to the first value of the sweep variable.

---

Argument	Description
TO	End of the <i>ERRfun</i> calculation. Default is last value of the sweep variable.

---

### Description

The relative error function reports the relative difference between two output variables. You can use this format in optimization and curve-fitting of measured data. The relative error format specifies the variable to measure and calculate from the `.PARAM` variable. To calculate the relative error between the two, HSPICE uses the `ERR`, `ERR1`, `ERR2`, or `ERR3` functions. With this format you can specify a group of parameters to vary to match the calculated value and the measured data.

### Examples

```
.measure ac comp1 err1 par(s11m) s11(m)  
.measure tran rel err1 par(out2) v(out) from=1u to=2u
```

### See Also

[.OPTION AUTOSTOP \(or\) .OPTION AUTOST](#)

---

## .MEASURE PHASENOISE

Enables measurement of phase noise at various frequency points in HSPICE RF.

### Syntax

#### *Find-When ... Phase Noise*

```
.MEASURE PHASENOISE result FIND phnoise At = IFB_value  
+ [PRINT 0|1]  
.MEASURE PHASENOISE result WHEN phnoise=value [PRINT 0|1]
```

#### *RMS, average, min, max, and peak-to-peak Phase Noise*

```
.MEASURE PHASENOISE result funcphnoise + [FROM = IFB1] [TO  
= IFB2] [PRINT 0|1]
```

#### *Integral Evaluation of Phase Noise*

```
.MEASURE PHASENOISE result INTEGRAL phnoise + [FROM = IFB1]  
[TO = IFB2] [PRINT 0|1]
```

#### *Derivative Evaluation of Phase noise*

```
.MEASURE PHASENOISE result DERIV[ACTIVE] phnoise AT = IFB1  
+ [PRINT 0|1]
```

#### *Amplitude modulation noise*

```
.MEASURE phasenoise result AM[NOISE] phnoise  
+ [FROM = IFB1] [TO = IFB2] [PRINT 0|1]
```

#### *Phase modulation noise*

```
.MEASURE phasenoise result PM[NOISE] phnoise  
+ [FROM = IFB1] [TO = IFB2] [PRINT 0|1]
```

---

Argument	Description
FIND	Selects the FIND function
result	Name of the measured result in the output.
<i>phnoise</i>	.MEASURE PHASENOISE value for phase noise
WHEN	Selects the WHEN function
<i>IFB_value</i>	Input frequency band point value

---



Argument	Description
func	Indicates one of the measure command types: <ul style="list-style-type: none"> <li>▪ AVG (average): Calculates the phase noise over the frequency range.</li> <li>▪ MAX (maximum): Reports the maximum value of the phase noise over the specified frequency range.</li> <li>▪ MIN (minimum): Reports the minimum value of the phase noise over the specified frequency range.</li> <li>▪ PP (peak-to-peak): Reports the maximum value, minus the minimum value of the phase noise over the specified frequency range.</li> <li>▪ RMS (root mean squared): Calculates the square root of the phase noise over the specified frequency range.</li> </ul>
FROM...TO	Optional range for input frequency bands (IFB)
INTEGRAL	Integrates the phase noise value from the first to the second IFB frequency points
DERIVATIVE	Finds the derivative of the phase noise at the first IFB frequency point
PM[NOISE]	Measures the phase modulation noise from the specified first to the second IFB frequency points (when <code>.OPTION PHASENOISEAMP=1</code> )
AM[NOISE]	Measures the amplitude modulation noise from the specified first to the second IFB frequency points (when <code>.OPTION PHASENOISEAMP=1</code> )

### Description

This command enables measurement of phase noise at various frequency points in HSPICE RF.

The `.MEASUREPHASENOISE` syntax supports yielding the following phase noise instances in dbc/Hz:

- Yields the phase noise using `FIND` or `WHEN` functions: at a specified input frequency band (`FIND`), or phase noise found at a specified input frequency point (`WHEN`).
- Yields the average, RMS, minimum, maximum, or peak-to-peak value of the phase noise from frequency `IFB1` to frequency `IFB2`, where the value of `func` can be `RMS`, `AVG`, `MIN`, `MAX` or `PP`. If `FROM` and `TO` are not specified, the value will be calculated over the frequency range specified in the `.PHASENOISE` command.

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .MEASURE PHASENOISE

- Integrates the phase noise value from the IFB1 frequency to the IFB2 frequency.
- Finds the derivative of phase noise at the IFB1 frequency point.

**Note:** The .MEASURE PHASENOISE command cannot contain an expression that uses a phase noise variable as an argument. You also cannot use .MEASUREPHASENOISE for error measurement and expression evaluation of PHASENOISE.

The HSPICE RF optimization flow can read the measured data from a .MEASURE PHASENOISE analysis. This flow can be combined in the HSPICE RF optimization routine with a .MEASUREHBTTR analysis.

### Examples

*Example 1* The **FIND** keyword yields the result of a variable value at a specific input frequency band (IFB) point.

```
.MEASURE PHASENOISE np1 find PHNOISE at=100K
```

*Example 2* The **WHEN** keyword yields the input frequency point at a specific phase noise value.

```
.MEASURE PHASENOISE fcorn1 when PHNOISE=-120
```

*Example 3* The following sample command find functions such as the RMS, AVG, MIN, MAX, or PP over the frequency range.

```
.measure PHASENOISE rn1 RMS phnoise  
.measure PHASENOISE agn1 AVG phnoise from=100k to=10meg  
.measure PHASENOISE nmin MIN phnoise
```

*Example 4* The **INTEGRAL** command integrates the phase noise across the two specified Input frequency band points.

```
.measure PHASENOISE inns1 INTEGRAL phnoise  
.measure PHASENOISE rns1 INTEGRAL phnoise from=50k to 500k
```

*Example 5* These **DERIV** sample commands find the derivative of the phase noise at one input frequency band point.

```
.measure PHASENOISE dnf1 DERIVATIVE phnoise at=100k  
.measure PHASENOISE fdn1 DERIVATIVE phnoise at=10meg
```

*Example 6* These **AM/PM** sample commands find the amplitude modulation (AM) and phase modulation (PM) noise across the specified input frequency range.

```
.measure PHASENOISE amp1 AM phnoise from=100k to 400k  
.measure PHASENOISE pmp1 PM phnoise from=10meg to=30meg
```

**See Also**

.PHASENOISE  
.MEASURE PTDNOISE  
.MEASURE (FIND and WHEN)  
.MEASURE (AVG, EM\_AVG, INTEG, MIN, MAX, PP, and RMS)  
.MEASURE (Integral Function)  
.MEASURE (Derivative Function)  
Measuring Phase Noise with .MEASURE PHASENOISE  
.HB  
.OPTION PHNOISEAMPM  
.OPTION AUTOSTOP (or) .OPTION AUTOST

---

## .MEASURE PTDNOISE

Allows for the measurement of integrated phase noise, time-point, tdelta-value, slewrate, and strobed jitter parameters in HSPICE RF.

### Syntax

```
.MEASURE PTDNOISE meas_name STROBEJITTER onoisefreq_sweep  
+ [PRINT 0 | 1]
```

---

Argument	Description
strobed jitter	Calculated from the noise voltage (integrated over the frequency range specified by <i>frequency_range</i> ), divided by the slewrate at the same node(s), at the time point specified by <i>time_value</i> . While only STROBEJITTER can be specified, all of the parameters listed below are also output to the *.msnptrn# file. Unit: sec
integptdnoise	Unit: V
timepoint	Unit: sec
tdelta-value	Unit: sec
slewrate	Unit: V/sec

---

### Description

Use to obtain strobed jitter parameters in large signal periodic time-dependent noise analysis. For more information, see the *HSPICE User Guide: RF Analysis* section on [Periodic Time-Dependent Noise Analysis \(.PTDNOISE\)](#).

### See Also

[.PTDNOISE](#)  
[.MEASURE Syntax and File Format](#)

## .MEASURE (Pushout Bisection)

Specifies a maximum allowed pushout time to control the distance from failure in bisection analysis.

### Syntax

```
.MEASURE TRAN result MeasureClause
+ pushout=time [lower|upper] [POSITIVE|NEGATIVE]
+ [PRINT 0|1]
-or-
.MEASURE TRAN result MeasureClause
+ pushout_per=percentage [lower|upper] [POSITIVE|NEGATIVE]
+ [PRINT 0|1]
```

Argument	Description
result	Name associated with the measured value in the HSPICE output, can be up to 16 characters long.
pushout=time	The absolute time to obtain the pushout result.
pushout_per=percentage	Relative error. If you specify a 0.1 relative error, the T_lower or T_upper and T_pushout have more than a 10% difference in value. This occurrence causes the iteration to stop and output the optimized parameter.
lower upper	Parameter boundary values for pushout comparison. These arguments are optional. If the parameter is defined as <code>.PARAM ParamName= OPTxxx(Initial, min, max)</code> The “lower” means the lower bound “min”, and the “upper” means the upper bound “max”. The default is lower.
POSITIVE	Pushout constraints only take effect when the measuring results are larger than the golden measure.
NEGATIVE	Pushout constraints only take effect when the measuring results are smaller than the golden measure.

### Description

Pushout is used only in bisection analysis. In Pushout Bisection, instead of finding the last point just before failure, you specify a maximum allowed pushout time to control the distance from failure.

To limit the range you can add both absolute and relative pushout together.

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .MEASURE (Pushout Bisection)

For example:

```
.Measure Tran pushout When v(D_Output)='vih/2'  
+ rise=1 pushout=20p,50p pushout_per=0.1
```

The final measure result for the preceding example should be in the range of:

```
| measresult-goldmeas | < Min (pushout_max,  
pushout_per*goldmeas)
```

...or the final measure result should satisfy,

```
Max(pushout_per*goldmeas, pushout_min)
```

### Examples

*Example 1 The parameter to be optimized is Delaytime and the evaluation goal is setup\_prop. The Pushout=1.5n lower means that the setup\_prop of the final solution is not 1.5n far from the setup\_prop of the lower bound of the parameter (0.0n).*

```
.Param DelayTime=Opt1 ( 0.0n, 0.0n , 5.0n )  
.Tran 1n 8n Sweep Optimize=Opt1 Result=setup_prop  
+ Model=OptMod  
.Measure Tran setup_prop Trig v(data)  
+ Val='v(Vdd) 2' fall=1 Targ v(D_Output)  
+ Val='v(Vdd)' rise=1 pushout=1.5n lower
```

*Example 2 The differences between the setup\_prop of the final solution and that of the lower bound of the parameter (0.0n) is not more than 10%.*

```
.Measure Tran setup_prop Trig v(data)  
+ Val='v(Vdd)/2' fall=1 Targ v(D_Output)  
+ Val='v(Vdd)' rise=1 pushout_per=0.1 lower
```

*Example 3 Pushout constraints only take effect when the measuring results are larger than the golden measure.*

```
.MEASURE TRAN result MeasureClause pushout=time  
+ pushout_perpercentage POSITIVE
```

*Example 4 Pushout constraints only take effect when the measuring results are smaller than the golden measure.*

```
.MEASURE TRAN result MeasureClause pushout=time  
+ pushout_perpercentage NEGATIVE
```

### See Also

[Pushout Bisection Methodology](#)

## .MEASURE (ACMATCH)

Introduces special keywords to access results for ACMatch analysis in HSPICE.

### Syntax

```
.MEASURE AC result [MAX] [ACM_Total|ACM_Global|+
    ACM_Global(par)|ACM_Local|ACM_Local(dev)] [PRINT 0|1]
```

Argument	Description
results	Name associated with the measured values in the HSPICE output, can be up to 16 characters long.
MAX	Sample function; Instead of "MAX" other functions can be used which select one out of multiple results.
ACM_Total	Output sigma due to global, local, and spatial variations.
ACM_Global	Output sigma due to global variations.
ACM_Global(par)	Contribution of parameter ( <i>par</i> ) to output sigma due to global variations.
ACM_Local	Output sigma due to local variations.
ACM_Local(dev)	Contribution of device ( <i>dev</i> ) to output sigma due to local variations.

### Description

ACMatch analysis saves results using `.MEASURE` commands, with AC type (M,P,R,I) for an output variable, as specified on the `.ACMatch` command. If you specify multiple output variables the command issues a result for the last one only. You must specify an AC sweep to produce these kinds of outputs; a single point sweep is sufficient. ACMatch uses the special keywords shown above to access the results from the different variation types. For usable keywords with the `.PROBE` command, see [Output from .PROBE and .MEASURE Commands for ACMatch](#) in the *HSPICE User Guide: Simulation and Analysis*.

### See Also

[.AC](#)  
[.MEASURE \(ACMATCH\)](#)  
[.PROBE](#)

---

## .MEASURE (DCMATCH)

Introduces special keywords to access the different types of results for DCMATCH analysis in HSPICE.

### Syntax

```
.MEASURE DC result [MAX] [DCM_Total | DCM_global |
+ DCM_Global(par) | DCM_Local | DCM_Local(dev) |
+ DCM_Spatial | DCM_Spatial(par)] [PRINT 0|1]
```

---

Argument	Description
result	Name associated with the measured values in the HSPICE output, can be up to 16 characters long.
MAX	Sample function. Instead of "MAX," other functions can be used which select one out of multiple results.
DCM_Total	Output sigma due to global, local, and spatial variations.
DCM_Global	Output sigma due to global variations.
DCM_Global( <i>par</i> )	Contribution of parameter ( <i>par</i> ) to output sigma due to global variations.
DCM_Local	Output sigma due to local variations.
DCM_Local( <i>dev</i> )	Contribution of device ( <i>dev</i> ) to output sigma due to local variations.
DCM_Spatial	Output sigma due to local variations.
DCM_Spatial( <i>par</i> )	Contribution of parameter ( <i>par</i> ) to output sigma due to spatial variations.

---

### Description

DCMATCH analysis uses special keywords to access the different types of results. You can save the different results produced by a DCMATCH analysis using the .MEASURE command for the output variable specified on the .DCMATCH command. For keywords to be used with the .PROBE command, see [Syntax for .PROBE Command for DCMATCH](#) in the *HSPICE User Guide: Simulation and Analysis*. If you specify multiple output variables, the command produces a result for the last one only. You must specify a DC sweep to produce these kinds of outputs; a single point sweep is sufficient.



### Examples

In this example, the result `systoffset` reports the systematic offset of the amplifier; the result `matchoffset` reports the variation due to mismatch; and the result `maxoffset` reports the maximum (3-sigma) offset of the amplifier.

```
.MEAS DC systoffset avg V(inp,inn)
.MEAS DC matchoffset avg DCm_local
.MEAS DC maxoffset
param='abs(systoffset)+3.0*matchoffset'
```

### See Also

[.DC](#)

[.PROBE](#)

---

## .MEASURE FFT

Specifies measurement of FFT results.

### Syntax

#### Syntax #1

```
.MEASURE FFT result
+ Find [vm|vp|vr|vi|vdb|im|ip|ir|ii|idb] (signal) AT=freq
+ [PRINT 0|1]
```

#### Syntax #2

```
.MEASURE FFT result THD signal_name [nbharm=num]
[PRINT 0|1]
```

#### Syntax #3

```
.MEASURE FFT result [SNR|SNDR|ENOB] signal_name
+ [nbharm=num|maxfreq=val] [BINSIZ=num] [PRINT 0|1]
```

#### Syntax #4

```
.MEASURE FFT result SFDR signal_name
+ [minfreq=val] [maxfreq=val] [PRINT 0|1]
```

---

Argument	Description
result	Name associated with the measured values in the FFT output, can be up to 16 characters long.
Find	FIND function.
At	Value of the <i>frequency</i> at which the component frequency and signal are found
frequency component/signal	Can be any of the following: vm vp vr vi vdb im ip ir ii idb
freq	Specified frequency
THD	Total harmonic distortion
signal_name	User-supplied name of signal
nbharm	Harmonic up to which to carry out the measurement. Default: highest harmonic in FFT result.

Argument	Description
maxfreq	Higher limit of frequency range to carrying out the measurement. Default: maximum frequency in an FFT result.
minfreq	Lower limit of frequency range to calculate SFDR.
SNR	Signal to noise ratio.
SNDR	Signal to noise-plus-distortion ratio.
ENOB	Effective number of bits.
BINSIZ	Filters out noise component within the bin; the noise component is calculated from the index of "fundamental_freq_idx+BINSIZ+1". Default=0.
SFDR	Spurious free dynamic range.

### Description

Four syntaxes are provided for finding measurements of several types for FFT results.

See examples below for sample usage.

- Syntax #1: Measures a frequency component at certain frequency.
- Syntax #2: Measures THD of a signal spectrum up to a specified harmonic; Default: nbharm=maximum harmonic in FFT result
- Syntax # 3: Measures SNR/SNDR/ENOB of a signal up to a specified frequency; Defaults: nbharm=maximum harmonic in FFT result; maxfreq=maximum frequency in FFT result; BINSIZ=0.
- Syntax # 4: Measures SFDR of a signal from minfreq to maxfreq; searches the frequency component with maximum magnitude from minfreq to maxfreq.

An embedded .MEASURE FFT command in a measure file can be called to perform FFT measurements from previous simulation results as follows:

```
HSPICE -i *.tr0 -meas measure_file
```

### Examples

*Example 1 Measures frequency component at certain frequency.*

```
.meas FFT v12 Find vm(1,2)at=20k
```

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .MEASURE FFT

*Example 2 Measures THD of a signal spectrum up to a specified harmonic.*

```
.meas FFT thd56 THD V(node5, node6) nbharm=10
```

*Example 3 Measures SNR/SNDR/ENOB of a signal up to a specified frequency.*

```
.meas FFT snr12 SNDR V(node1, node2) maxfreq=1G
```

*Example 4 Measures SFDR of a signal from minfreq to maxfreq and searching the frequency component with maximum magnitude from minfreq to maxfreq.*

```
.meas FFT sfdr9 SFDR V(node9)
```

*Example 5 Filters out the noise component within the bin.*

```
.meas fft snrsrc SNR v(out) BINSIZ=10
```

### See Also

[.FFT](#)

[Spectrum Analysis](#)

---

## .MEASURE LSTB

Enables the measurement of `lstb` output variables similar to any other common `ac` variable.

### Syntax

```
.MEASURE AC Result FIND LSTB(x) WHEN LSTB(x)=val
.MEASURE AC Result LSTB(x) FROM=val TO=val
```

---

Argument	Description
AC	AC analysis result
Result	Result name of an .AC .LSTB analysis
LSTB	Output loop gain as complex numbers
LSTB( <i>x</i> )	<ul style="list-style-type: none"> <li>▪ <code>x=DB</code>: Output the dB values of loop gain</li> <li>▪ <code>x=M</code>: Output magnitude of loop gain</li> <li>▪ <code>x=P</code>: Output phase of loop gain</li> <li>▪ <code>x=R</code>: Output real part of loop gain</li> <li>▪ <code>x=I</code>: Output imaginary part of loop gain</li> </ul>
FIND...WHEN	Selects the Find and When functions
FROM...TO	Selects the range of measurement

---

### Description

Use the `.MEASURE LSTB` statement to measure linear loop stability outputs in a similar manner to any common `ac` output variable. Measure phase margin, gain margin, unity gain frequency, dc gain, etc... from the return ratio waveform that is generated in the `.LSTB` analysis. The `lstb` scalars can be measured as follows:

```
.measure lstb out1 gain_margin
.measure lstb out2 phase_margin
.measure lstb out3 unity_gain_freq
.measure lstb out4 loop_gain_minifreq
```

in which `out1 - out4` are names for measured output, `lstb` is the type name, and `gain_margin`, `phase_margin`, `unity_gain_freq`, and `loop_gain_minifreq` are keywords of scalar variables.

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .MEASURE LSTB

#### Examples

*Example 1* Finds the measurement for the output phase of loop gain of phase margin when the decibel output is 0.

```
.MEASURE AC PHASE_MARGIN FIND LSTB(P) WHEN LSTB(DB)=0
```

*Example 2* Measures the first output phase of loop gain Integral across a range of 1 to 100k.

```
.MEASURE AC INTEG1 INTEGRAL LSTB(P) FROM=1k TO=100k
```

#### See Also

[.LSTB](#)

---

## .MODEL

Includes an instance of a predefined HSPICE model in an input netlist.

### Syntax

#### *Passive and active device model syntax*

```
.MODEL mname type [level=num]  
+ [pname1=val1pname2=val2 ...]
```

See specific element type for supported model parameter information.

#### *Optimization model syntax*

```
.MODEL mname OPT [METHOD=BISECTION|PASSFAIL] [close=num]  
+ [max] [cut=val] [difsiz=val] [grad=val] [parmin=val]  
+ [relin=val] [relout=val] [absout=val]  
+ [itrop=val] [absin=val]  
+ [DYNACC=0|1] [cendif=num]
```

*The following syntax is used for a Monte Carlo analysis:*

```
.MODEL mname ModelType ([level=val]  
+ [keyword1=val1] [keyword2=val2]  
+ [keyword3=val3] [lot/n/distributionvalue]  
+ [DEV/n /distribution value]...)
```

*The following syntax is used for model reliability analysis*

```
.model mname mosra  
+ level|mosralevel value  
+ [relmodelparam]
```

---

Argument	Description
<i>mname</i>	Model name reference. Elements must use this name to refer to the model. If model names contain periods (.), the automatic model selector might fail. When used with .MOSRA it is the user-defined MOSFET reliability model name.

Argument	Description
type	<p>Model type. Must be one of the following.:</p> <ul style="list-style-type: none"> <li>▪ AMP—operational amplifier model</li> <li>▪ C—capacitor model</li> <li>▪ CORE—magnetic core model</li> <li>▪ D—diode model</li> <li>▪ L—inductor model or magnetic core mutual inductor model</li> <li>▪ NJF—n-channel JFET model</li> <li>▪ NMOS—n-channel MOSFET model</li> <li>▪ NPN—npn BJT model</li> <li>▪ OPT—optimization model</li> <li>▪ PJF—p-channel JFET model</li> <li>▪ PLOTQ—plot model for the .GRAPH command (obsolete)</li> <li>▪ PMOS—p-channel MOSFET model</li> <li>▪ PNP—pnp BJT model</li> <li>▪ R—resistor model</li> <li>▪ U—lossy transmission line model (lumped)</li> <li>▪ W—lossy transmission line model</li> <li>▪ S—S-parameter</li> </ul>
level	<p>Model level.</p> <ul style="list-style-type: none"> <li>▪ For optimization model, LEVEL=1 specifies the Modified Levenberg-Marquardt method. Use this setting with multiple optimization parameters and goals.</li> <li>▪ Only Level=1 is available in HSPICE. Additional levels are available in HSPICE RF. See below This argument is ignored when METHOD has been specified.</li> <li>▪ LEVEL=2 (HSPICE RF) specifies the BISECTION method in HSPICE RF. You would use this setting with one optimization parameter.</li> <li>▪ LEVEL=3 (HSPICE RF) specifies the PASSFAIL method. You would use this setting with two optimization parameters.</li> <li>▪ For transistors, diodes, and some passive element models, see the <a href="#">HSPICE Reference Manual, Elements and Device Models</a>.</li> <li>▪ For MOSFET Models, see the <a href="#">HSPICE Reference Manual: MOSFET Models</a>.</li> <li>▪ To use custom MOSRA models and for discussion of LEVEL values, refer to the HSPICE Application Note: Unified Custom Reliability Modeling API (MOSRA API). Contact your Synopsys technical support team for more information.</li> </ul>



Argument	Description
pname1 ...	Parameter name. Assign a model parameter name ( <i>pname1</i> ) from the parameter names for the appropriate model type. Each model section provides default values. For legibility, enclose the parameter assignment list in parentheses and use either blanks or commas to separate each assignment. Use a plus sign (+) to start a continuation line.
OPT	Keyword to indicate the definition model is for optimization analysis.
METHOD	Specifies an optimization method. <ul style="list-style-type: none"> <li>▪ METHOD=BISECTION specifies the Bisection method. When the difference between the two latest test input values is within the error tolerance and the latest measured value exceeds the goal, bisection has succeeded and then ends. This process reports the optimized parameter that corresponded to the test value that satisfies this error tolerance and this goal (passes).</li> <li>▪ METHOD=PASSFAIL specifies the passfail method. When the difference between the last two optimization parameter test values is &lt; the error tolerance and the associated goal measurement fails for one of the values and passes for the other, bisection has succeeded and then ends. The process reports the optimization parameter test value associated with the last passing measurement. “Pass” is defined as a condition in which the associated goal measurement can produce a valid result. “Fail” is defined as a condition in which the associated goal measurement is unable to produce a valid result.</li> </ul> <p>This argument supersedes LEVEL when present.</p>
close	(Optimization) Initial estimate of how close parameter initial value estimates are to the solution. The close argument multiplies changes in new parameter estimates. If you use a large close value, the optimizer takes large steps toward the solution. For a small value, the optimizer takes smaller steps toward the solution. You can use a smaller value for close parameter estimates and a larger value for rough initial guesses. The default is 1.0. <ul style="list-style-type: none"> <li>▪ If close is greater than 100, the steepest descent in the Levenburg-Marquardt algorithm dominates.</li> <li>▪ If close is less than 1, the Gauss-Newton method dominates.</li> </ul> <p>For more details, see L. Spruiell, “Optimization Error Surfaces,” <i>Meta-Software Journal</i>, Volume 1, Number 4, December 1994.</p>
max	(Optimization) Upper limit on close. Use values > 100. The default is 6.0e+5.

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

.MODEL

Argument	Description
cut	<p>(Optimization) Modifies close, depending on how successful iterations are toward the solution. If the last iteration succeeds, descent toward the close solution decreases by the cut value. That is, <math>close=close / cut</math></p> <p>If the last iteration was not a successful descent to the solution, close increases by cut squared. That is, <math>close=close * cut * cut</math>.</p> <p>Cut drives close up or down, depending on the relative success in finding the solution. The cut value must be <math>&gt; 1</math>. The default is 2.0.</p>
difsiz	<p>(Optimization) Increment change in a parameter value for gradient calculations (<math>\Delta x=DIFSIZ \cdot MAX(x, 0.1)</math>). If you specify delta in a .PARAM command, then <math>\Delta x=delta</math>. The default is 1e-3.</p>
grad	<p>(Optimization) Represents possible convergence if the gradient of the RESULTS function is less than GRAD. Most applications use values of 1e-6 to 1e-5. Too large a value can stop the optimizer before finding the best solution. Too small a value requires more iterations. The default is 1.0e-6.</p>
parmin	<p>(Optimization) Allows better control of incremental parameter changes during error calculations. The default is 0.1. This produces more control over the trade-off between simulation time and optimization result accuracy. To calculate parameter increments, HSPICE uses the relationship: <math>\Delta par\_val=\Delta IFSIZ \cdot MAX(par\_val, PARMIN)</math></p>
relin	<p>(Optimization) Relative input parameter (<math>delta\_par\_val / MAX(par\_val, 1e-6)</math>) for convergence. If all optimizing input parameters vary by no more than RELIN between iterations, the solution converges. RELIN is a relative variance test so a value of 0.001 implies that optimizing parameters vary by less than 0.1% from one iteration to the next. The default is 0.001.</p>
relout	<p>(Optimization) Relative tolerance to finish optimization. For relout=0.001, if the relative difference in the RESULTS functions from one iteration to the next, is less than 0.001, then optimization is finished. The default is 0.001.</p>
absout	<p>(Bisection) Absolute tolerance to finish bisection. For absout=0.001, if the absolute difference in the RESULTS functions from one iteration to the next, is less than 0.001, then bisection is completed. The default is 0.0, which means inactive, and use relout.</p>
itropt	<p>(Optimization) Maximum number of iterations. Typically, you need no more than 20-40 iterations to find a solution. Too many iterations can imply that the relin, grad, or relout values are too small. The default is 20.</p>

Argument	Description
absin	(Optimization) Overrides relin parameter and ignores relout and itropt; there is no default value
DYNACC	(Optimization) Dynamic accuracy tolerance setting to accelerate bisection simulation. The default is 0.  When DYNACC=1, if HSPICE is in accuracy mode, it uses reduced accuracy simulations to narrow the bisection window, then switches to the original accuracy algorithm to refine the solution. This method reduces simulation time by doing the majority of simulations at lower accuracy, which run faster by taking fewer time steps.
ceendif	(Optimization) Point at which more accurate derivatives are desired.
keyword	(Monte Carlo) Model parameter keyword.
distribution	(Monte Carlo) The distribution function name, which must be specified as GAUSS, AGAUSS, LIMIT, UNIF, or AUNIF. If you do not set the distribution function, the default distribution function is used. The default distribution function is uniform distribution.
DEV	(Monte Carlo) DEV tolerance, which is independent (each device varies independently).
LOT	(Monte Carlo) The LOT tolerance, which requires all devices that refer to the same model use the same adjustments to the model parameter.
LOT/n DEV/n	(Monte Carlo) Sets which of ten random number generators numbered 0 through 9 is used to calculate parameter value deviations. This correlates deviations between parameters in the same model as well as between models. The generators for DEV and LOT tolerances are distinct: Ten generators exist for both DEV tracking and LOT tracking. N must be an integer 0 to 9.
mosra	Keyword to indicate the definition model is for MOSRA analysis.

Argument	Description
level (alias: mosrlevel)	<p>To use the Synopsys MOSRA model, set LEVEL=1. For compatibility with HSPICE, in the .MODEL statement, 'LEVEL=' can be replaced with 'MOSRALEVEL='. HSPICE will consider them equivalent. Example: The following two lines will be interpreted the same by HSPICE.</p> <pre>.MODEL my_mod MOSRA LEVEL=1</pre> <pre>.MODEL my_mod MOSRA MOSRALEVEL=1</pre> <p>To use custom MOSRA models and for discussion of LEVEL values, refer to the HSPICE Application Note: Unified Custom Reliability Modeling API (MOSRA API). Contact your Synopsys technical support team for more information.</p>
RelMode	<p>HSPICE reliability mode level; selects whether a simulation accounts for both HCI and NBTI effects or either one of them. If the RelMode in the .MOSRA command is defined as 1 or 2, it takes higher priority and applies to all MOSRA models. If RelMode in the .MOSRA command is not set or set to 0, then the RelMode inside individual MOSRA models take precedence for that MOSRA model only; the rest of the MOSRA models take the RelMode value from the .MOSRA command. If any other value is set, except 0, 1, or 2, a warning is issued, and RelMode is automatically set to the default value 0.</p> <ul style="list-style-type: none"> <li>▪ 0: both HCI and NBTI, Default.</li> <li>▪ 1: HCI only</li> <li>▪ 2: NBTI only</li> </ul>
relmodelparam	<p>Model parameter for HCI or BTI, when doing a reliability MOSFET device analysis. See <a href="#">Level 1 MOSRA BTI and HCI Model Parameters in the HSPICE User Guide: Simulation and Analysis</a> for listing of HCI and NBTI parameters. Contact Synopsys Technical Support for access to the MOSRA API.</p>

### Description

Use this command to include an instance (element) of a predefined HSPICE model in your input netlist.

For each optimization within a data file, specify a .MODEL command. HSPICE can then execute more than one optimization per simulation run. The .MODEL optimization command defines:

- **Convergence criteria:** (Bisection accuracy is controlled by the parameters: `relin`, `relout`, `absin`, and `itropt`. The `relin` parameter (default 1e-3) times the bisection window size is the goal accuracy. This goal accuracy is also influenced by `relout` (default 1e-3, the difference ratio between last two iterations) and `itropt` (default 20 iterations). To keep the same bisection accuracy, these three parameters need to change accordingly with

a changing bisection window size. You can override the `relin` value by using the `absin` option which has no default (see [.OPTION ABSIN.](#)) The `absin` parameter also ignores `itropt` and `relout`.

- Number of iterations
- Derivative methods

### Examples

*Example 1 A standard .model statement.*

```
.MODEL MOD1 NPN BF=50 IS=1E-13 VBF=50 AREA=2 PJ=3 N=1.05
```

*Example 2 Shows the addition of the DYNACC=1 option in an optimization model card to invoke bisection speedup.*

```
.MODEL optmod OPT METHOD=BISECTION ITROPT=20 dynacc=1 relout=1e20
```

*Example 3 Model command used for a Monte Carlo analysis.*

```
.model m1 nmos level=6 bulk=2 vt=0.7 dev/2 0.1
+ tox=520 lot/gauss 0.3 a1=.5 a2=1.5 cdb=10e-16
+ csb=10e-16 tcv=.0024
```

*Example 4 Transistors M1 through M3 have the same random vto model parameter for each of the five Monte Carlo runs through the use of the LOT construct.*

```
...
.model mname nmos level=53 vto=0.4 LOT/agauss 0.1 version=3.22
M1 11 21 31 41 mname W=20u L=0.3u
M2 12 22 32 42 mname W=20u L=0.3u
M3 13 23 33 43 mname W=20u L=0.3u
...
.dc v1 0 vdd 0.1 sweep monte=5
.end
```

*Example 5 Transistors M1 through M3 have different values of the vto model parameter for each of the Monte Carlo runs through the use of the DEV construct.*

```
...
.model mname nmos level=54 vto=0.4 DEV/agauss 0.1
M1 11 21 31 41 mname W=20u L=0.3u
M2 12 22 32 42 mname W=20u L=0.3u
M3 13 23 33 43 mname W=20u L=0.3u
...
.dc v1 0 vdd 0.1 sweep monte=5
.end
```

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .MODEL

*Example 6 Establishes a MOS reliability model card.*

```
.model NCH_RA mosra
+ level=1
+ a_hci=1e-2
+ n_hci=1
```

### See Also

[Cell Characterization Examples](#) for demo files of .MODEL opt

Method=bisection or passfail

[BJT and Diode Examples](#) for all listed \*.sp files in the demo group which use the .MODEL command for npn transistors.

---

## .MODEL\_INFO

Enables printout of all or specified MOSFET model parameters for each simulation.

### Syntax

```
.MODEL_INFO ALL | instance_name1, instance_name2, ...,
```

---

Argument	Description
ALL	Prints all MOSFET instances.
instance_name1... instance_name2	Specific MOSFET instance. If the MOSFET instance is in a .SUBCKT command, it must be written in the full hierarchical path, e.g.: x1.x2.main.

---

### Description

This command generates a text format file with the suffix *\*.model\_info#*.

**Note:** If the arguments ALL and instance\_name are specified together, ALL will take higher priority.

### Examples

*Example 1* Prints all MOSFET instances' model parameters.

```
.MODEL_INFO ALL
```

*Example 2* Prints the model parameters for devices.

```
.model_info main x1.m1 x2.m2
```

*Example 3* Prints the model parameters for devices nch.27 and pch.19.

```
.model_info x_m_11.nch.27 x_m_2.nch.27 x_m_3.pch.19
```

*Example 4* Prints all MOSFET instances' model parameters. ("ALL" takes higher priority over instances.)

```
.MODEL_INFO ALL x1.m1
```

### See Also

[Using .MODEL\\_INFO to Print Model Parameters](#)

---

## .MOSRA

Starts HSPICE HCI and/or BTI reliability analysis for HSPICE.

### Syntax

```
.MOSRA RelTotalTime=time_value
+ [RelStartTime=time_value] [DEC=value] [LIN=value]
+ [RelStep=time_value] [RelMode=0|1|2] SimMode=[0|1|2|3]
+ [AgingStart=time_value] [AgingStop=time_value]
+ [AgingPeriod=time_value] [AgingWidth=time_value]
+ [AgingInst="inst_name"]
+ [HciThreshold=value] [NbtiThreshold=value]
+ [Integmod=0|1|2] [Xpolatemod=0|1|2]
+ [Tsample1=value] [Tsample2=value]
+ [Agethreshold=value]
+ [MosraLlife=degradation_type_keyword] [DegF=value]
+ [DegFN=value] [DegFP=value]
```

---

Argument	Description
RelTotalTime	Final reliability test time to use in post-stress simulation phase. Required argument.
RelStartTime	Time point of the first post-stress simulation. Default is 0.
DEC	Specifies number of post-stress time points simulated per decade.
LIN	Linear post-stress time points from RelStartTime to RelTotalTime.
RelStep	Post-stress simulation phase on time= RelStep, 2* RelStep, 3* RelStep, ... until it achieves the RelTotalTime; the default is equal to RelTotalTime. Value is ignored if DEC or LIN value is set.



Argument	Description
RelMode	<p>HSPICE reliability model mode selects whether a simulation accounts for both HCI and BTI effects or either one of them. If the RelMode in the .MOSRA command is defined as 1 or 2, it takes higher priority and applies to all MOSRA models. If RelMode in the .MOSRA command is not set or set to 0, then the RelMode inside individual MOSRA models take precedence for that MOSRA model only; the rest of the MOSRA models take the RelMode value from the .MOSRA command. If any other value is set, except 0, 1, or 2, a warning is issued, and RelMode is automatically set to the default value 0.</p> <ul style="list-style-type: none"> <li>▪ 0: both HCI and BTI, Default.</li> <li>▪ 1: HCI only</li> <li>▪ 2: BTI only</li> </ul>
SimMode	<ul style="list-style-type: none"> <li>▪ 0: Select pre-stress simulation only</li> <li>▪ 1: Select post-stress simulation only</li> <li>▪ 2: Select both pre- and post-stress simulation</li> <li>▪ 3: Select continual degradation integration through alters</li> </ul> <p>When SimMode=1</p> <ul style="list-style-type: none"> <li>▪ HSPICE reads in the *.radeg0 file and uses it to update the device model for reliability analysis; new transient output is generated in a *.tr1 waveform file.</li> <li>▪ The *.radeg file and input netlist must be in the same directory.</li> <li>▪ The netlist stimuli could be different from the SimMode=0 netlist that generated the *.radeg file.</li> </ul> <p>When SimMode=3</p> <ul style="list-style-type: none"> <li>▪ If no .option radegfile is specified in the top level netlist, simulation will start from fresh device.</li> <li>▪ If .option radegfile is specified in the top level netlist, HSPICE reads in the last suite degradation in the radeg file, and continues the degradation integration and extrapolation from the corresponding circuit time in the radeg file.</li> <li>▪ In consecutive alters, the radeg generated from the previous alter run is read in.</li> </ul>
AgingStart	Optionally defines time when HSPICE starts stress effect calculation during transient simulation. Default is 0.0.
AgingStop	Optionally defines time when HSPICE stops stress effect calculation during transient simulation. Default is tstop in .TRAN command.
AgingPeriod	Stress period. Scales the total degradation over time.

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .MOSRA

Argument	Description
AgingWidth	The AgingWidth (circuit time “on”) argument works with the AgingPeriod argument. For example: if you specify AgingPeriod=1.0s and AgingWidth=0.5s, then the circuit is turned on for 0.5s, and turned off for 0.5s. (The period is 1.0s.)
AgingInst	Selects MOSFET devices to which HSPICE applies HCI and/or BTI analysis. The default is all MOSFET devices with reliability model appended. The name must be surrounded by quotes. Multiple names allowed/wildcards supported.
HciThreshold	Optionally, used in post-stress simulation. HCI effect is accounted for in a particular transistor, based on the specified HCI threshold value. Default is 0.0
NbtiThreshold	Optionally, used in post-stress simulation. BTI effect is accounted for in a particular transistor, based on this threshold value. Default is 0.0
Integmod	The flag is used to select the integration method and function. <ul style="list-style-type: none"><li>▪ 0 (default): User-defined integration function in MOSRA API</li><li>▪ 1: True derivation and integration method</li><li>▪ 2: Linearized integration method (support non-constant n coefficient)</li></ul>
Xpolatmod	The flag is used to select the extrapolation method and function. <ul style="list-style-type: none"><li>▪ 0 (default): User-defined extrapolation function in MOSRA API</li><li>▪ 1: Linearization extrapolation method (support non-constant n coefficient)</li><li>▪ 2: Two-point fitting extraction and extrapolation method</li></ul>
Tsample1	First simulation time point of stress_total sampling for Xpolatmod=2
Tsample2	Second simulation time point of stress_total sampling for Xpolatmod=2
Agethreshold	Only when the degradation value $\geq$ Agethreshold, the MOSFET information will be printed in the MOSRA output file. Default is 0.
MosraLife	Argument to compute device lifetime calculation for the degradation type specified. This argument has the same function as <code>.option mosralife</code> .
DegF	Sets the MOSFET’s failure criteria for lifetime computation. This argument has the same function as <code>.option degf</code> . If <code>.option degf</code> is specified, it takes precedence over <code>.mosra degf</code> .

Argument	Description
DegFN	Sets the NMOS's failure criteria for lifetime computation. This argument has the same function as <code>.option degfn</code> . If <code>.option degfn</code> is specified, it takes precedence over <code>.mosra degfn</code> .
DegFP	Sets the PMOS's failure criteria for lifetime computation. This argument has the same function as <code>.option degfp</code> . If <code>.option degfp</code> is specified, it takes precedence over <code>.mosra degfp</code> .

### Description

Use the `.MOSRA` command to initiate HCI and BTI analysis for the following models: Level 49, Level 53, Level 54, Level 57, Level 66, Level 69, Level 70, Level 71, Level 73, and external CMI MOSFET models. This is a two-phase simulation, the fresh simulation phase and the post stress simulation phase. During the fresh simulation phase, HSPICE computes the electron age/stress of selected MOS transistors in the circuit based on circuit behavior and the HSPICE built-in stress model including HCI and/or BTI effect. During the post stress simulation phase, HSPICE simulates the degradation effect on circuit performance, based on the stress information produced during the fresh simulation phase. If you specify either DEC or LIN, the RelStep value is ignored.

For a full description refer to the *HSPICE User Guide: Simulation and Analysis: MOSFET Model Reliability Analysis (MOSRA)*.

### Examples

```
.mosra reltotaltime=6.3e+8 relstep=6.3e+7
+ agingstart=5n agingstop=100n
+ hcithreshold=0 nbtithreshold=0
+ aginginst="x1.*"
```

### See Also

[.APPENDMODEL](#)  
[.MODEL](#)

---

## .MOSRAPRINT

Provides .PRINT/.PROBE capability to access the electrical degradation of the specified element.

### Syntax

```
.MOSRAPRINT output_nameoutput_type(element_name, vds=exp1,  
vgs=exp2, vbs=exp3)
```

---

Argument	Description
output_name	User-defined output variable; this output_name@element_name is used as the as output variable name in the output file.
output_type	One of the following output variable types: vth, gm, gds, ids, dids or dvth
element_name	The element name that the .MOSRAPRINT command applies.

---

### Description

The .MOSRAPRINT command supports the following models: B3SOI, B4SOI, PSP, BSIM3, BSIM4, HVMOS, HiSIM-HV, and Custom CMI MOSFETS.

This command provides access to device degradation information. The vds, vgs and vbs are user-specified bias conditions used to characterize the device electrical property as specified by the output type. There is no order requirement for vds, vgs, and vbs. Wildcards '?' and '\*' are supported in element\_name. The output variable dids reports the change of ids between post-stress simulation and fresh-simulation. dvth reports the percentage change of vth between post-stress simulation and fresh-simulation. The output file format is the same as the measurement file format with file extension \*.ra.

.MOSRAPRINT does approximate calculation to get the ids.

### Examples

The following syntax prints the ids value of the MOSFET m1, when vds = 5vgs=5, vbs=0, at each reltime point.

```
.MOSRA reltotaltime=5e+7 relstep=1e+7  
.MOSRAPRINT ids(m1, vds=5, vgs=5, vbs=0)
```

### See Also

[.MOSRA](#)

---

## .MOSRA\_SUBCKT\_PIN\_VOLT

When a MOSFET is wrapped by a subckt-based macro model, this command specifies the subckt terminal voltages used by MOSRA model evaluation.

### Syntax

```
.MOSRA_SUBCKT_PIN_VOLT subckt=subckt_name1,
+ subckt_name2,...
```

### Description

Use this command to specify subckt-based macro terminal voltages HSPICE will use for MOSRA model evaluation.

*subckt*: The subcircuit name whose terminal voltages to be used for MOSRA model evaluation.

**Note:** There is a limitation to this capability. The subckt-based macro model can contain only one MOSFET, and the number and definition of subckt terminals must be consistent with HSPICE MOSFET terminal number and definition.

### Examples

In this example, HSPICE will use subckt sub1's terminal voltages  $v(d)/v(g)/v(s)/v(b)$ , instead of the MOSFET M1's terminal voltages,  $v(d1)/v(g1)/v(s1)/v(b1)$ ,  $v(d1)/v(g1)/v(s1)/v(b)$  for MOSRA model evaluation.

```
.subckt sub1 d g s b ...
M1 d1 g1 s1 b ...
Rd d d1 1k
Rs s s1 1k
Rg g g1 1k
.model ...
.ends
.mosra_subckt_pin_volt subckt=sub1
...
.end
```

---

## .NODESET

Initializes specified nodal voltages for DC operating point analysis and corrects convergence problems in DC analysis.

### Syntax

```
.NODESET V(node1)=val1 V(node2)=val2 ... [subckt=sub_name]  
-or-  
.NODESET node1val1node2val2 [subckt=sub_name]
```

---

Argument	Description
<i>node1</i> ...	Node numbers or names can include full paths or circuit numbers.
<i>val1</i>	Voltages.
<i>subckt</i> = <i>sub_name</i>	Initial condition is set to the specified node name(s) within all instances of the specified subcircuit name. This <i>subckt</i> setting is equivalent to placing the <code>.NODESET</code> command within the subcircuit definition.

---

### Description

Use the `.NODESET` command to set a seed value for the iterative DC convergence algorithm for all specified nodal voltages. Use this to correct convergence problems in DC analysis. How it behaves depends on whether the `.TRAN` analysis command includes the UIC parameter.

Forcing circuits are connected to the `.NODESET` nodes for the first iteration of DC convergence. To increase the number of held iterations, see [.OPTION DCHOLD](#). The forcing circuits are then removed and Newton Raphson iterations continued until DC convergence is obtained. The `.NODESET` nodes can move to their true DC operating points. For this reason, `.NODESET` should be used to provide initial guesses to either speed up convergence, aid non-convergence, or to set the preferred DC state of multi-stable nodes. If the DC operating voltage of a `.NODESET` node is appreciably different than the voltage in the `.NODESET` command you should investigate the circuit to determine why. It is a likely error condition.

**Note:** In nearly all applications you should use `.NODESET` to ensure a true DC operating point. Set intentionally floating (or very high impedance) nodes to a known good voltage using `.IC`.

If you do not specify the UIC parameter in the `.TRAN` command then use `.NODESET` to set seed values for an initial guess for DC operating point

analysis. If the node value is close to the DC solution then you will enhance convergence of the simulation.

If you specify the UIC parameter in the `.TRAN` command, HSPICE does not calculate the initial DC operating point, but directly enters transient analysis. When you use `.TRAN UIC`, the `.TRAN` node values (at time zero) are determined by searching for the first value found in this order: from `.IC` value, then `IC` parameter on an element command, then `.NODESET` value, otherwise use a voltage of zero.

Note that forcing a node value of the DC operating point might not satisfy KVL and KCL. In this event you might see activity during the initial part of the simulation. This might happen if you use UIC and do not specify some node values, when you specify too many conflicting `.IC` values, or when you force node values and topology changes. Forcing a node voltage applies a fixed equivalent voltage source during DC analysis and transient analysis removes the voltage sources to calculate the second and later time points. Therefore to correct DC convergence problems use `.NODESETs` (without `.TRAN UIC`) liberally (when a good guess can be provided) and use `.ICs` sparingly (when the exact node voltage is known).

In addition, you can use wildcards in the `.NODESET` command. See [Using Wildcards on Node Names](#) in the *HSPICE User Guide: Simulation and Analysis*.

## Examples

### Example 1

```
.NODESET V(5:SETX)=3.5V V(X1.X2.VINT)=1V  
.NODESET V(12)=4.5 V(4)=2.23  
.NODESET 12 4.5 4 2.23 1 1
```

*Example 2* All settings in this statement are applied to subckt my\_ff.

```
.NODESET V(in)=0.9 subckt=my_ff
```

## See Also

- [.DC](#)
- [.IC](#)
- [.OPTION DCHOLD](#)
- [.TRAN](#)

---

## .NOISE

Controls the noise analysis of the circuit.

### Syntax

```
.NOISE v(out) vin [interval|inter=x]  
+ [listckt=[1|0]]  
+ [listfreq=frequencies|none|all]  
+ [listcount=num] [listfloor=val]  
+ [listsources=1|0|yes|no]
```

---

Argument	Description
v(out)	Nodal voltage or branch current output variable. Defines the node or branch at which HSPICE sums the noise.
vin	Independent voltage source to use as the noise input reference
interval   inter	Interval at which HSPICE prints a noise analysis summary. <i>inter</i> specifies how many frequency points to summarize in the AC sweep. If you omit <i>inter</i> or set it to zero, HSPICE or HSPICE RF does not print a summary. If <i>inter</i> is equal to or greater than one, HSPICE prints summary for the first frequency, and once for each subsequent increment of the interval frequency. The noise report is sorted according to the contribution of each node to the overall noise level. If any of the LIST* arguments below are specified, the output information will follow the format required by LIST*, and <i>interval</i> does not influence the output information for later sweeps.
listckt= [1 0]	<ul style="list-style-type: none"><li>1: The contribution of each subcircuit is listed in the <i>.lis</i> file and you can view the subcircuit noise contribution curve in WaveView.</li><li>0: (default) HSPICE does not list the noise contribution of any subcircuits.</li></ul>



Argument	Description
listfreq= (none all freq1 freq2....)	<p>Dumps the element noise figure value to the <i>.lis</i> file. You can specify which frequencies the element phase noise value dumps. The frequencies must match the sweep_frequency values defined in the parameter_sweep, otherwise they are ignored. In the element phase noise output, the elements that contribute the largest phase noise are dumped first. The frequency values can be specified with the NONE or ALL keyword, which either dumps no frequencies or every frequency defined in the parameter_sweep.</p> <ul style="list-style-type: none"> <li>▪ ALL: output all of the frequency points (default, if LIST* is required.)</li> <li>▪ NONE: do not output any of the frequency points</li> <li>▪ freq1 freq2...: output the information on the specified frequency points</li> </ul> <p>Frequency values must be enclosed in parentheses. For example:  listfreq=(none) listfreq=(all) listfreq=(1.0G) listfreq=(1.0G, 2.0G)</p>
listcount= <i>num</i>	<p>Outputs the first few noise elements that make the biggest contribution to NF. The number is specified by <i>num</i>. The default is to output all of the noise element contribution to NF. The NF contribution is calculated with the source impedance equal to the <math>Z_o</math> of the first port.</p>
listfloor= <i>val</i>	<p>Contribution to the output noise power greater than the value specified by LISTFLOOR. Default is to output all the noise elements. The unit of LISTFLOOR is <math>V^2/hz</math></p>
listsources= [1 0 yes no]	<p>Defines whether or not to output the contribution of each noise source of each noise element. Default is no/0.</p>

### Description

Use this command and *.AC* commands to control the noise analysis of the circuit. You can use this command only with an *.AC* command. Noise contributor tables are generated for every frequency point and every circuit device. The last four arguments allow users to better control the output information.

### Examples

*Example 1* This example sums the output noise voltage at the node 5 by using the voltage source VIN as the noise input reference and prints a noise analysis summary every 10 frequency points.

```
.NOISE V(5) VIN 10
```

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .NOISE

*Example 2* Sums the output noise current at the r2 branch by using the voltage source VIN as the noise input reference and prints a noise analysis summary every 5 frequency points.

```
.NOISE I(r2) VIN 5
```

*Example 3* Shows the list subcircuit option turned on and sample results:

```
*****  
subcircuit squared noise voltages (sq v/hz)  
x1 total 1.90546e-20  
x7 total 7.14403e-19  
x1.x3 total 1.90546e-20  
*****
```

### See Also

[.AC](#)

---

## .OP

Calculates the DC operating point of the circuit; saves circuit voltages at multiple timesteps.

### Syntax

```
.OP format time format time... [interpolation]  
...  
.op voltage time1 time2...
```

---

Argument	Description
format	<p>Any of the following keywords. Only the first letter is required. The default is ALL</p> <ul style="list-style-type: none"><li>▪ ALL: Full operating point, including voltage, currents, conductances, and capacitances. This parameter outputs voltage/current for the specified time.</li><li>▪ BRIEF: One-line summary of each element's voltage, current, and power. Current is stated in milliamperes and power in milliwatts.</li><li>▪ CURRENT: Voltage table with a brief summary of element currents and power.</li><li>▪ DEBUG: Usually invoked only if a simulation does not converge. Debug prints the non-convergent nodes with the new voltage, old voltage, and the tolerance (degree of non-convergence). It also prints the non-convergent elements with their tolerance values.</li><li>▪ NONE: Inhibits node and element printouts, but performs additional analysis that you specify.</li><li>▪ VOLTAGE: Voltage table only.</li></ul> <p>The preceding keywords are mutually-exclusive; use only one at a time.</p>
time	<p>Time at which HSPICE prints the report. HSPICE RF returns node voltages only if time (t) is 0.</p>
interpolation	<p>Interpolation method for .OP time points during transient analysis or no interpolation. Only the first character is required; that is, typing <b>i</b> has the same effect as typing <b>interpolation</b>. Default is not active. If you specify <i>interpolation</i>, all of the time points in the .OP command (except time=0) use the interpolation method to calculate the OP value during the transient analysis. If you use this keyword, it must be at the end of the .OP command. HSPICE ignores any word after this keyword.</p>

---

### Description

Use this command to calculate the DC operating point of the circuit. You can also use the .OP command to produce an operating point during a transient analysis. You can include only one .OP command in a simulation.

If an analysis requires calculating an operating point you do not need to specify the `.OP` command; HSPICE calculates an operating point. If you use a `.OP` command and if you include the `UIC` parameter in a `.TRAN` analysis command, then simulation omits the `time=0` operating point analysis and issues a warning in the output listing.

Use `.OP` to output circuit node voltages at different timesteps to `*.ic0` files. You can replace use of the `.SAVE` command to save node voltages. The `*.ic0` files are identical to those created by the `.SAVE` command. (Remove `.SAVE` commands to avoid conflict with the `.OP` command used to save node voltages.)

If you want to generate `*.dp#` files for your transient simulations, use `.OPTION OPFILE` in your netlist.

**Note:** Without `.OP` in the netlist, HSPICE does not create an `*.op0` file. Operating point information is printed in `*.lis`, `*op0` and `psf` format files. HICUM level 0 information is also supported.

### Examples

Example 1 calculates:

- Operating point at .5ns.
- Currents at 10 ns for the transient analysis.
- Voltages at 17.5 ns, 20 ns and 25 ns for the transient analysis.

#### Example 1

```
.OP .5NS CUR 10NS VOL 17.5NS 20NS 25NS
```

Example 2 calculates a complete DC operating point solution.

#### Example 2

```
.OP
```

### See Also

[.TRAN](#)  
[.OPTION OPFILE](#)

---

## .OPTION (or) .OPTIONS

Modifies various aspects of an HSPICE simulation; individual options for HSPICE and HSPICE RF commands are described in [Chapter 3, HSPICE and RF Netlist Simulation Control Options](#).

### Syntax

```
.OPTION opt1 [opt2opt3 ...]
```

---

Argument	Description
----------	-------------

---

<i>opt1</i> ...	Input control options. Many options are in the form <i>opt=x</i> , where <i>opt</i> is the option name and <i>x</i> is the value assigned to that option.
-----------------	---

---

### Description

Use this command to modify various aspects of an HSPICE simulation, including:

- output types
- accuracy
- speed
- convergence

You can set any number of options in one `.OPTION` command, and you can include any number of `.OPTION` commands in an input netlist file. Most options default to 0 (OFF) when you do not assign a value by using either `.OPTIONopt=val` or the option with no assignment: `.OPTIONopt`.

To reset options, set them to 0 (`.OPTIONopt=0`). To redefine an option, enter a new `.OPTION` command; HSPICE uses the last definition.

You can use the following types of options with this command. For detailed information on individual options, see [Chapter 3, HSPICE and RF Netlist Simulation Control Options](#).

- [General Control Options](#)
- [Input/Output Controls](#)
- [Model and Variation Definition](#)
- [Analysis](#)

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

.OPTION (or) .OPTIONS

For instructions on how to use options that are relevant to a specific simulation type, see the appropriate analysis chapters in the *HSPICE User Guide: Simulation and Analysis* for:

- [Initializing DC-Operating Point Analysis](#)
- [Pole-Zero Analysis](#)
- [Spectrum Analysis](#)
- [Transient Analysis](#)
- [AC Small-Signal and Noise Analysis](#)
- [Linear Network Parameter Analysis](#)
- [Timing Analysis Using Bisection](#)
- [Monte Carlo - Traditional Flow and Statistical Analysis](#)
- [Analyzing Variability and Using the Variation Block](#)
- [Monte Carlo Analysis Variation Block Flow](#)
- [Mismatch Analyses](#)
- [Optimization](#)
- [RC Reduction and Post-Layout Simulation](#)
- [MOSFET Model Reliability Analysis \(MOSRA\)](#)

### Examples

```
.OPTION BRIEF $ Sets BRIEF to 1 (turns it on)
* Netlist, models,
...
.OPTION BRIEF=0 $ Turns BRIEF off
```

This example sets the BRIEF option to 1 to suppress a printout. It then resets BRIEF to 0 later in the input file to resume the printout.

---

## .PARAM (or) .PARAMETER (or) .PARAMETERS

Defines parameters in HSPICE and HSPICE RF.

### Syntax

Simple parameter assignment:

```
.PARAM ParamName=RealNumber
```

Algebraic parameter assignments:

```
.PARAM ParamName='AlgebraicExpression'
```

```
.PARAM ParamName1=ParamName2
```

User-defined functions:

```
.PARAM ParamName(pv1 [pv2])='Expression'
```

Redefined analysis functions—Variability definitions (see [.PARAM Distribution Function](#)):

```
.PARAM ParamName=DistributionFunction(Arguments)
```

Optimization parameter assignment:

```
.PARAM ParamName=OPTxxx (initial_guess, low_limit,  
+ upper_limit)
```

```
.PARAM ParamName=OPTxxx (initial_guess, low_limit,  
+ upper_limit, delta)
```

String parameter assignment:

```
.PARAM ParamName=str('string')
```

---

Argument	Description
parameter	Parameter to vary. <ul style="list-style-type: none"><li>▪ Initial value estimate.</li><li>▪ Lower limit.</li><li>▪ Upper limit.</li></ul> If the optimizer does not find the best solution within these constraints, it attempts to find the best solution without constraints.
OPTxxx	Optimization parameter reference name. The associated optimization analysis references this name. Must agree with the <i>OPTxxx</i> name in the analysis command associated with an OPTIMIZE keyname.

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

.PARAM (or) .PARAMETER (or) .PARAMETERS

---

Argument	Description
delta	The final parameter value is the initial guess $\pm (n \cdot \text{delta})$ . If you do not specify <i>delta</i> , the final parameter value is between <i>low_limit</i> and <i>upper_limit</i> . For example, you can use this parameter to optimize transistor drawn widths and lengths, which must be quantized.

---

### Description

Use this command to define parameters. Parameters in HSPICE are names that have associated numeric values.

**Note:** A .PARAM statement with no definition is illegal.

A parameter definition always uses the last value found in the input netlist (subject to global parameter rules).

Use any of the following methods to define parameters:

- A simple parameter assignment is a constant real number. The parameter keeps this value unless a later definition changes its value or an algebraic expression assigns a new value during simulation. HSPICE does not warn you if it reassigns a parameter.
- An algebraic parameter (equation) is an algebraic expression of real values, a predefined or user-defined function or circuit or model values. Enclose a complex expression in single quotes to invoke the algebraic processor, *unless* the expression begins with an alphabetic character and contains no spaces. A simple expression consists of a single parameter name. To use an algebraic expression as an output variable in a .PRINT, or .PROBE command, use the PARAM keyword.
- A user-defined function assignment is similar to an algebraic parameter. HSPICE extends the algebraic parameter definition to include function parameters, used in the algebraic that defines the function. You can nest user-defined functions up to three levels deep.
- A predefined analysis function. HSPICE provides several specialized analysis types, which require a way to control the analysis:
  - Temperature functions (fn)
  - Optimization guess/range

HSPICE also supports the following predefined parameter types:

- Frequency



- Time
- Monte Carlo functions

**Note:** To print the final evaluated values of all .PARAM commands in the netlist, use .OPTION LIST. This helps you avoid seeing the same value for every time point if you run a transient analysis.

## Examples

*Example 1 Examples 1-3 illustrate predefined analysis function*

```
.PARAM mcVar=Agauss(1.0,0.1,1)
```

*Example 2 In this example, uox and vtx are the variable model parameters, which optimize a model for a selected set of electrical specifications. The estimated initial value for the vtx parameter is 0.7 volts. You can vary this value within the limits of 0.3 and 1.0 volts for the optimization procedure. The optimization parameter reference name (OPT1) references the associated optimization analysis command (not shown).*

```
PARAM vtx=OPT1(.7,.3,1.0) uox=OPT1(650,400,900)
```

*Example 3*

```
.PARAM fltmod=str('bpfmodel')  
s1 n1 n2 n3 n_ref fqmodel=fltmod zo=50 fbase=25e6 fmax=1e9
```

*Example 4 Simple parameter assignment*

```
.PARAM power_cycles=256
```

*Example 5 Numerical parameter assignment*

```
.PARAM TermValue=1g  
rTerm Bit0 0 TermValue  
rTerm Bit1 0 TermValue  
...
```

*Example 6 Parameter assignment using expressions*

```
.PARAM Pi          ='355/113'  
.PARAM Pi2         ='2*Pi'  
.PARAM npRatio     =2.1  
.PARAM nWidth      =3u  
.PARAM pWidth      ='nWidth * npRatio'  
Mp1 ... <pModelName> W=pWidth  
Mn1 ... <nModelName> W=nWidth  
...
```

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

.PARAM (or) .PARAMETER (or) .PARAMETERS

### *Example 7 Algebraic parameter*

```
.param x=cos(2)+sin(2)
```

### *Example 8 Algebraic expression as an output variable*

```
.PRINT DC v(3) gain=PAR('v(3)/v(2)')  
+ PAR('V(4)/V(2)')
```

### *Example 9 User-defined functions*

```
.PARAM MyFunc( x, y )='Sqrt((x*x)+(y*y))'  
.PARAM CentToFar (c) = '(((c*9)/5)+32)'  
.PARAM F(p1,p2) = 'Log(Cos(p1)*Sin(p2))'  
.PARAM SqrProd (a,b) = '(a*a)*(b*b)'
```

### *Example 10 Undefined .PARAM statement results in a warning requesting parameter variables with their respective values or expressions.*

```
.PARAM $ Illegal as a standalone netlist command.
```

## See Also

[.OPTION LIST](#)

---

## .PAT

Specifies predefined pattern names to be used in a pattern source; also defines new patnames.

### Syntax

```
.PAT PatName=data [RB=val] [R=int]
```

```
.PAT patName=[component 1... component n] [RB=val]  
+ [R=repeat]
```

[or]

```
.PAT PatName=data [RB=param_expr1] [R=param_expr2]
```

```
.PAT patName=[component 1 ... component n] [RB=param_expr1]  
+ [R=param_expr2]
```

---

Argument	Description
data	String of 1, 0, M, or Z that represents a pattern source. The first letter must be B to represent it as a binary bit stream. This series is called b-string. A 1 represents the high voltage or current value, and a 0 is the low voltage or current value. An M represents the value that is equal to $0.5 \cdot (v_{hi} + v_{lo})$ , and a Z represents the high impedance state (only for voltage source).
PatName	Pattern name that has an associated b-string or nested structure.
component	Elements that make up a nested structure. Components can be b-strings or a patname defined in other .PAT commands.
RB=val	Starting component of a repetition. The repeat data starts from the component or bit indicated by RB. RB must be an integer. If RB is larger than the length of the NS or b-string, an error is issued. If it is less than 1, it is automatically set to 1.
R=repeat	Specifies how many times the repeating operation is executed. With no argument, the source repeats from the beginning of the nested structure or b-string. If R=-1, the repeating operation continues infinitely. The R must be an integer. If it is less than -1, it is automatically set to 0.

---

## Description

When the `.PAT` command is used in an input file, some *patnames* are predefined and can be used in a pattern source. Patnames can associate a b-string or nested structure, two different types of pattern sources. In this case, a b-string is a series of 1, 0, m, and z states. The nested structure is a combination of a b-string and another netlisted structure defined in the `.PAT` command. The `.PAT` command can also be used to define a new patname, which can be a b-string or nested structure.

Avoid using a predefined patname to define another patname to lessen the occurrence of a circular definition for which HSPICE issues an error report.

Nested structures must use brackets “[ ]”, but HSPICE does not support using multiple brackets in one command. If you need to use another nested structure as a component, define it in a new `.PAT` command.

## Examples

*Example 1 Shows eight instances of the .PAT command used for a b-string.*

```
.PAT a1=b1010 r=1 rb=1
.PAT a1=b10101010
.PAT a1=b1010 b0011 r=1 rb=2
.PAT a1=b1010 b0011011
.PAT a1=b1010 r=1 rb=1 b0011 r=1 rb=2
.PAT a1=b10101010 b0011011
.PAT a1=b1010 b0011 r=2 rb=2
.PAT a1=b1010 b0011011011
```

*Example 2 Shows four instances of how an existing patname is used to define a new patname:*

```
.PAT a1=b1010 r=1 rb=1
.PAT a2=a1
.PAT a1=b1010 r=1 rb=1
.PAT a2=b1010 r=1 rb=1
```

*Example 3 Shows a nested structure:*

```
.PAT a1=[b1010 r=1 rb=2 b1100]
```

**Example 4** Shows several instances of how a predefined nested structure is used as a component in a new nested structure:

```
.PAT a1=[b1010 r=1 rb=2 b1100] r=1 rb=1  
.PAT a2=[a1 b0m0m] r=2 rb=1  
.PAT a1=[b1010 r=1 rb=2 b1100] r=1 rb=1  
.PAT a2=a1 b0m0m a1 b0m0m a1 b0m0m  
.PAT a1=b1010 r=1 rb=2 b1100 b1010 r=1 rb=2 b1100  
.PAT a2=b1010 r=1 rb=2 b1100 b1010 r=1 rb=2 b1100 b0m0m  
+ b1010 r=1 rb=2 b1100 b1010 r=1 rb=2 b1100 b0m0m  
+ b1010 r=1 rb=2 b1100 b1010 r=1 rb=2 b1100 b0m0m
```

**Example 5** Shows several instances of how a predefined nested structure is used as a component in a new nested structure:

```
.PAT a1=[b1010 r=1 rb=2 b1100] r=1 rb=2  
.PAT a2=[a1 b0m0m] r=2 rb=2  
.PAT a1=[b1010 r=1 rb=2 b1100] r=1 rb=2  
.PAT a2=a1 b0m0m b0m0m b0m0m  
.PAT a1=b1010 r=1 rb=2 b1100 b1100  
.PAT a2=b1010 r=1 rb=2 b1100 b1100 b0m0m b0m0m b0m0m
```

---

## .PHASENOISE

Performs phase noise analysis on autonomous (oscillator) circuits in HSPICE RF.

### Syntax

```
.PHASENOISE outputfrequency_sweep [method=0|1|2]
+ [carrierindex=int] [listfreq=(frequencies|none|all)]
+ [listcount=val] [listfloor=val] [listsources=on|off]
+ [spurious=0|1]
```

---

Argument	Description
output	Output node, pair of nodes, or 2-terminal element. HSPICE RF references phase noise calculations to this node (or pair of nodes). Specify a pair of nodes as V(n+,n-). If you specify only one node, V(n+), then HSPICE RF assumes that the second node is ground. You can also specify a 2-terminal element.
frequency_sweep	Sweep of type LIN, OCT, DEC, POI, or SWEEPBLOCK. Specify the type, nsteps, and start and stop time for each sweep type, where: <ul style="list-style-type: none"> <li>▪ type = Frequency sweep type, such as OCT, DEC, or LIN.</li> <li>▪ nsteps = Number of steps per decade or total number of steps.</li> <li>▪ start = Starting frequency.</li> <li>▪ stop = Ending frequency.</li> </ul> The four parameters determine the offset frequency sweep about the carrier used for the phase noise analysis. <ul style="list-style-type: none"> <li>▪ LIN <i>type nsteps start stop</i></li> <li>▪ OCT <i>type nsteps start stop</i></li> <li>▪ DEC <i>type nsteps start stop</i></li> <li>▪ POI <i>type nsteps start stop</i></li> <li>▪ SWEEPBLOCK <i>freq1 freq2 ... freqn</i></li> </ul>
METHOD	<ul style="list-style-type: none"> <li>▪ METHOD=0 (default) selects the Nonlinear Perturbation (NLP) algorithm, which is used for low-offset frequencies.</li> <li>▪ METHOD=1 selects the Periodic AC (PAC) algorithm, which is used for high-offset frequencies.</li> <li>▪ METHOD=2 selects the Broadband Phase Noise (BPN) algorithm, which you can use to span low and high offset frequencies.</li> </ul> You can use <i>METHOD</i> to specify any single method.

Argument	Description
carrierindex	Harmonic index of the carrier at which HSPICE RF computes the phase noise (optional). The phase noise output is normalized to this carrier harmonic. The default is 1.
listfreq	Element phase noise value written to the <i>.lis</i> file. You can specify which frequencies the element phase noise value dumps. The frequencies must match the sweep_frequency values defined in the parameter_sweep, otherwise they are ignored. In the element phase noise output, the elements that contribute the largest phase noise are dumped first. The frequency values can be specified with the NONE or ALL keyword, which either dumps no frequencies or every frequency defined in the parameter_sweep. Frequency values must be enclosed in parentheses. For example: listfreq=(none) listfreq=(all) listfreq=(1.0G) listfreq=(1.0G, 2.0G) The default value is the first frequency value.
listcount	Dumps the element phase noise value to the <i>.lis</i> file, which is sorted from the largest to smallest value. You do not need to dump every noise element; instead, you can define listcount to dump the number of element phase-noise frequencies. For example, listcount=5 means that only the top 5 noise contributors are dumped. The default value is 20.
listfloor	Dumps the element phase noise value to the <i>.lis</i> file and defines a minimum meaningful noise value (in dBc/Hz units). Only those elements with phase-noise values larger than the <i>listfloor</i> value are dumped. For example, listfloor=-200 means that all noise values below -200 (dbc/Hz) are not dumped. The default value is -300 dbc/Hz.
listsources	Writes the element phase-noise value to the <i>.lis</i> file. When the element has multiple noise sources, such as a level 54 MOSFET, which contains the thermal, shot, and 1/f noise sources. When dumping the element phase-noise value you can decide if you need to dump the contribution from each noise source. You can specify either ON or OFF: ON dumps the contribution from each noise source and OFF does not. The default value is OFF.

---

Argument	Description
spurious	Additional .HBAC analysis that predicts the spurious contributions to the phase noise. Spurs result from deterministic signals present within the circuit. In most cases, the spurs are very small signals and do not interfere with the steady-state operation of the oscillator but do add energy to the output spectrum of the oscillator. The energy that the spurs adds might need to be included in jitter measurements. 0 - No spurious analysis (default) 1 - Initiates a spurious noise analysis

---

### Description

Use this command to invoke phase noise analysis on autonomous (oscillator) circuits.

### See Also

[.HB](#)

[.HBAC](#)

[.HBOSC](#)

[.SN](#)

[.SNAC](#)

[.SNOSC](#)

[.PRINT](#)

[.PROBE](#)

[Identifying Phase Noise Spurious Signals](#)



---

## .PKG

Provides the IBIS Package Model feature; automatically creates a series of W-elements or discrete R, L and C components.

### Syntax

```
.PKG pkgname  
+ file = 'pkgfilename'  
+ model = 'pkgmodelname'
```

---

Argument	Description
pkgname	Package card name
pkgfilename	Name of a <i>.pkg</i> or <i>.ibs</i> file that contains package models.
pkgmodelname	Working model in the <i>.pkg</i> file

---

### Description

The `.PKG` command provides the IBIS Package Model feature. It supports both sections and matrixes.

The `.PKG` command automatically creates a series of W-elements or discrete R, L and C components. The following nodes are referenced in the netlist:

- Nodes on the die side:  
`'pkgname'_'pinname'_dia`
- Nodes on the pin side:  
`'pkgname'_'pinname'`

See Example 2 for how pin1 is referenced.

- If package = 0 in the *.IBIS* card, then no package information is added.
- If package = 1 or 2, then the package information in the *.ibs* file is added.
- If package = 3, then the package information in the *.pkg* file is added.

### Examples

*Example 1* Illustrates a typical *.PKG* statement.

```
.pkg p_test  
+ file='processor_clk_ff.ibs'  
+ model='FCPGA_FF_PKG'
```

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

.PKG

*Example 2 Shows how pin1 is referenced.*

p\_test\_pin1\_dia and p\_test\_pin1

*Example 3 The element name becomes:*

w\_p\_test\_pin1\_? ? or r\_p\_test\_pin1\_? ? ...

### See Also

[.EBD](#)

[.IBIS](#)

---

## .POWER

Prints a table containing the AVG, RMS, MAX, and MIN measurements for specified signals in HSPICE RF.

### Syntax

```
.POWER signal [REF=vname FROM=start_time TO=end_time]
```

---

Argument	Description
<i>signal</i>	Signal name.
<i>vname</i>	Reference name.
<i>start_time</i>	Start time of power analysis period. You can also use parameters to define time.
<i>end_time</i>	End time of power analysis period. You can also use parameters to define time.

---

### Description

Use this command to print a table containing the AVG, RMS, MAX, and MIN measurements for every signal specified.

By default, the scope of these measurements are set from 0 to the maximum timepoint specified in the `.TRAN` command.

For additional information, see [POWER Analysis](#) in the *HSPICE User Guide: RF Analysis*.

### Examples

*Example 1* No simulation start and stop time is specified for the `x1.in` signal, so the simulation scope for this signal runs from the start (0ps) to the last `.tran` time (100ps).

```
.power x1.in
.tran 4ps 100ps
```

Example 2 shows how you can use the `FROM` and `TO` times to specify a separate measurement start and stop time for each signal. In this example.

- The scope for simulating the `x2.in` signal is from 20ps to 80ps.
- The scope for simulating the `x0.in` signal is from 30ps to 70ps.

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .POWER

#### *Example 2*

```
.param myendtime=80ps  
.power x2.in REF=a123 from=20ps to=80ps  
.power x0.in REF=abc from=30ps to='myendtime - 10ps'
```

#### **See Also**

- [.TRAN](#)
- [.OPTION SIM\\_POWER\\_ANALYSIS](#)
- [.OPTION SIM\\_POWER\\_TOP](#)
- [.OPTION SIM\\_POWERPOST](#)
- [.OPTION SIM\\_POWERSTART](#)
- [.OPTION SIM\\_POWERSTOP](#)

---

## .POWERDC

Calculates the DC leakage current in the design hierarchy.

### Syntax

```
.POWERDC keywordsubckt_name1...
```

---

Argument	Description
keyword	One of these keywords: <ul style="list-style-type: none"> <li>▪ TOP – prints the power for top-level instances</li> <li>▪ ALL (default) – prints the power for all instances</li> </ul>
subckt_name#	Prints the power of all instances in this subcircuit definition

---

### Description

Use this command to calculate the DC leakage current in the design hierarchy.

This option prints a table containing the measurements for AVG, MAX, and MIN values for the current of every instance in the subcircuit. This table also lists the sum of the power of each port in the subcircuit.

For additional information, see [POWER Analysis](#) in the *HSPICE User Guide: RF Analysis*.

You can use the `SIM_POWERDC_HSPICE` and `SIM_POWERDC_ACCURACY` options to increase the accuracy of the `.POWERDC` command.

### See Also

[.OPTION SIM\\_POWERDC\\_ACCURACY](#)  
[.OPTION SIM\\_POWERDC\\_HSPICE](#)

---

## .PRINT

Prints the values of specified output variables.

### Syntax

```
.PRINT antype ov1 [ov2 ... ]
```

---

Argument	Description
----------	-------------

---

antype	Type of analysis for outputs. Can be one of the following types: DC, AC, TRAN, NOISE, or DISTO.
ov1 ...	Output variables to print. These are voltage, current, or element template variables from a DC, AC, TRAN, NOISE, or DISTO analysis.

---

### Description

Use this command to print the values of specified output variables. You can include wildcards in .PRINT commands. You can also use the `iall` keyword in a .PRINT command to print all branch currents of all diode, BJT, JFET, or MOSFET elements in your circuit design. By default, the .PRINT command prints out simulation data at a time interval of `tstep` of .TRAN command, so the number of points for this output data reported in the \*.lis are the “# points” shown at the end of \*.lis file.

### Examples

*Example 1 Three cases of invoking the print function:  
In Case 1, if you replace the .PRINT command with: .print TRAN v(din)i(mnx), then all three cases have identical .sw0 and .tr0 files.  
If you replace the .printcommand with: .print DC v(din) i(mnx), then the .sw0 and .tr0 files are different.*

```
* CASE 1
.print v(din) i(mxn18)
.dc vdin 0 5.0 0.05
.tran 1ns 60ns
* CASE 2
.dc vdin 0 5.0 0.05
.tran 1ns 60ns
.print v(din) i(mxn18)
* CASE 3
.dc vdin 0 5.0 0.05
.print v(din) i(mxn18)
.tran 1ns 60ns
```

**Example 2** *Example 2 prints the results of a transient analysis for the nodal voltage named 4. It also prints the current through the voltage source named VIN. It also prints the ratio of the nodal voltage at the OUT and IN nodes.*

```
.PRINT TRAN V (4) I(VIN) PAR(`V(OUT)/V(IN)')
```

**Example 3** *In Example 3:  
Depending on the value of the ACOUT option, VM(4,2) prints the AC magnitude of the voltage difference, or the difference of the voltage magnitudes between nodes 4 and 2.  
VR(7) prints the real part of the AC voltage between node 7 and ground.  
Depending on the ACOUT value, VP(8,3) prints the phase of the voltage difference between nodes 8 and 3, or the difference of the phase of voltage at node 8 and voltage at node 3.  
II(R1) prints the imaginary part of the current through R1.*

```
.PRINT AC VM(4,2) VR(7) VP(8,3) II(R1)
```

**Example 4** *This example prints:  
The magnitude of the input impedance.  
The phase of the output admittance.  
Several S and Z parameters.*

```
.PRINT AC ZIN YOUT(P) S11(DB) S12(M) Z11(R)
```

**Example 5** *This example prints the DC analysis results for several different nodal voltages and currents through:  
The resistor named R1.  
The voltage source named VSRC.  
The drain-to-source current of the MOSFET named M1.*

```
.PRINT DC V(2) I(VSRC) V(23,17) I1(R1) I1(M1)
```

**Example 6** *Prints the equivalent input noise.*

```
.PRINT NOISE INOISE
```

**Example 7** *Prints the magnitude of third-order harmonic distortion, and the dB value of the intermodulation distortion sum through the load resistor that you specify in the .DISTO command.*

```
.PRINT DISTO HD3 SIM2(DB)
```

**Example 8** *The command in Example 8 includes NOISE, DISTO, and AC output variables in the same .PRINT statement.*

```
.PRINT AC INOISE ONOISE VM(OUT) HD3
```

**Example 9** *Prints the value of pj1 with the specified function. (HSPICE ignores .PRINT command references to nonexistent netlist part names, and prints those names in a warning.)*

```
.PRINT pj1=par(`p(rd) +p(rs)`)
```

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .PRINT

*Example 10* The commands in Example 10 illustrate print statements for a derivative function and an integrative function. The parameter can be a node voltage or a reasonable expression.

```
.PRINT der=deriv('v(NodeX)')  
.PRINT int=integ('v(NodeX)')
```

*Example 11* Shows how you can use p1 and p2 as parameters in netlist. The p1 value is 3; the p2 value is 15. You can use p1 and p2 as parameters in netlist.

```
.param p1=3  
.print par('p1')  
.print p2=par("p1*5")
```

*Example 12* Shows the syntax for outputting the length and width of a polygon in template format for the following models: BSIM3, BSIM4, BSIM3SOI, BSIM4SOI, and PSP.

```
.print ac wpoly() lpoly()
```

### See Also

- [.AC](#)
- [.DC](#)
- [.DCMATCH](#)
- [.DISTO](#)
- [.DOUT](#)
- [.MEASURE \(or\) .MEAS](#)
- [.NOISE](#)
- [.PROBE](#)
- [.STIM](#)
- [.TRAN](#)

[Measuring the Value of MOSFET Model Card Parameters](#)



---

## .PROBE

Saves output variables to interface and graph data files.

### Syntax

```
.PROBE analysis_type ov1 [ov2 ...]
.PROBE analysis_type v(inst_name.subckt_port_name)
```

---

Argument	Description
analysis_type	Type of analysis for the specified plots. Analysis types are: DC, AC, TRAN, NOISE, or DISTO for HSPICE; ENV, HB, HBAC, HBLSP, HBNOISE, HBTR, HBTRAN, HBXF, NOISE, or PHASENOISE for HSPICE RF.
ov1...	Output variables to plot: voltage, current, or element template (HSPICE-only variables from a DC, DCMATCH, AC, ACMATCH, TRAN, NOISE, or DISTO analysis. .PROBE can include more than one output variable. HSPICE RF analyses include: ENV, HB, HBAC, HBLSP, HBNOISE, HBTR, HBTRAN, HBXF, NOISE, or PHASENOISE analysis.
inst_name	Specifies instance name.
subckt_port_name	Specifies subcircuit port name.

---

### Description

Use this command to save output variables and print to interface and graph data files. Parameters can be node voltages, currents, elements, reasonable expressions, and node probe instances and ports. You can include wildcards in .PROBE commands. In order to save instance port nodes, you need to set .OPTION PROBE. The .PROBE command outputs the signals to waveform files no matter how .OPTION PROBE and .OPTION PUTMEAS are set.

**Note:** For AC analysis in HSPICE, only the magnitude is saved to the waveform file unless a complex quantity is explicitly specified.

### Examples

*Example 1* Saves several node voltages and an expression.

```
.PROBE DC V(4) V(5) V(1) beta=PAR(`I1(Q1)/I2(Q1)')
```

*Example 2* This syntax probes the voltage of the net connected with the Gate of XINST1.MN0.

```
PROBE TRAN V2(XINST1.MN0)
```

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .PROBE

*Example 3* Illustrates saving derivative and integrative functions.

```
* Derivative function
.PROBE der=deriv('v(NodeX)')
* Integrate function
.PROBE int=integ('v(NodeX)')
```

*Example 4* Last section of a netlist to generate a NAND circuit, illustrating printing of subcircuit node instances and ports. Adding `.OPTION POST PROBE` limits the output to the `*.lis` file.

```
...
.subckt nand0 data clk out vdd
mna n_mid data 0 0 n w=2u l=1u
mnb out clk n_mid 0 n w=2u l=1u
mpa out clk vdd vdd p w=2u l=1u
mpb out data vdd vdd p w=2u l=1u
.ends
xa data clk out vdd nand5
v1 vdd 0 3
vdata data 0 pw1 0 0 5n 0 5.01n 3
vclk clk 0 pw1 0 0 12n 0 12.01n 3
.tran 1p 200n
.probe tran v(xa.x5x4.x4x3.clk)
.probe tran v(xa.x5x1.x4x1.clk)
.probe tran v(xa.x5x1.x4x3.data)
.opt post probe lis_new
.end
```

### See Also

- [.AC](#)
- [.ACMATCH](#)
- [.DC](#)
- [.DCMATCH](#)
- [.DISTO](#)
- [.DOUT](#)
- [.ENV](#)
- [.HB](#)
- [.HBAC](#)
- [.HBLSP](#)
- [.HBNOISE](#)
- [.HBOSC](#)
- [.HBXF](#)
- [.MEASURE \(or\) .MEAS](#)
- [.NOISE](#)
- [.PHASENOISE](#)

.PRINT

.STIM

.TRAN

.OPTION PROBE

.OPTION PUTMEAS

Measuring the Value of MOSFET Model Card Parameters

---

## .PROTECT or .PROT

Keeps models and cell libraries private as part of the encryption process in HSPICE.

### Syntax

```
.PROTECT
```

### Description

Use this command to designate the start of the file section to be encrypted when using Metaencrypt.

- Use `.UNPROTECT` to end the file section that will be encrypted.
- Any elements and models located between a `.PROTECT` and an `.UNPROTECT` command inhibit the element and model listing from the `LIST` option.
- The `.OPTION NODE` nodal cross-reference and the `.OP` operating point printout do not list any nodes that are contained between the `.PROTECT` and `.UNPROTECT` commands.

**Caution:** If you use `.prot/.unprot` in a library or file that is not encrypted you will get warnings that the file is encrypted and the file or library is treated as a “black box.”

**Note:** To perform a complete bias check and print all results in the Outputs Biaschk Report, do not use `.protect/.unprotect` in the netlist for the part that is used in `.biaschk`. For example: If a model definition such as `model nch` is contained within `.prot/.unprot` commands, in the `*.lis` you'll see a warning message as follows: `**warning** : model nch defined in .biaschk cannot be found in netlist--ignored`

Usage Note: The `.prot/.unprot` feature is meant for the encryption process and *not* netlist echo suppression. Netlist and model echo suppression is on by default since HSPICE C-2009.03. For a compact and better formatted output (`*.lis`) file, use `.OPTION LIS_NEW`

### See Also

[.UNPROTECT or .UNPROT](#)  
[.OPTION BRIEF](#)  
[.OPTION LIS\\_NEW](#)

## .PTDNOISE

Calculates the noise spectrum and total noise at a point in time for HSPICE RF.

### Syntax

```
.PTDNOISE output time_value [time_delta]
+ frequency_sweep
+ [listfreq=(frequencies|none|all)] [listcount=val]
+ [listfloor=val] [listsources=on|off]
```

Argument	Description
output	An output node, pair of nodes, or 2-terminal elements. HSPICE RF references the equivalent noise output to this node (or pair of nodes). Specify a pair of nodes as V(n+,n-); only one node as V(n+, n-). If you specify only one node, V(n+), then HSPICE RF assumes the second node is ground. You can also specify a 2-terminal element name that refers to an existing element in the netlist.
time_value	Time point at which time domain noise is evaluated. Specify either a time point explicitly, such as: TIME=value, where value is either numerical or a parameter name or a .MEASURE name associated with a time domain .MEASURE command located in the netlist. PTDNOISE uses the time point generated from the .MEASURE command to evaluate the noise characteristics. This is useful if you want to evaluate noise or jitter when a signal reaches some threshold value.
time_delta	A time value used to determine the slew rate of the time-domain output signal. Specified as TDELTA=value. The signal slew rate is then determined by the output signal at TIME +/- TDELTA and dividing this difference by 2 x TDELTA. This slew rate is then used in the calculation of the strobed jitter. If this term is omitted a default value of 0.01 x the .SN period is assumed.

Argument	Description
frequency_sweep	<p>Frequency sweep range for the output noise spectrum. The upper and lower limits also specify the integral range in calculating the integrated noise value. Specify LIN, DEC, OCT, POI, SWEEPBLOCK, DATA sweeps. Specify the nsteps, start, and stop frequencies using the following syntax for each type of sweep:</p> <ul style="list-style-type: none"> <li>▪ LIN <i>nsteps start stop</i></li> <li>▪ DEC <i>nsteps start stop</i></li> <li>▪ OCT <i>nsteps start stop POI nsteps freq_values</i></li> <li>▪ SWEEPBLOCK <i>nsteps freq1 freq2 ... freqn</i></li> <li>▪ DATA <i>data_name</i></li> </ul>
listfreq	<p>Prints the element noise value to the <i>.lis</i> file. This information is only printed if a noise spectrum is requested in a PRINT or PROBE statement. (See <a href="#">PTDNOISE Output Syntax and File Format</a>.) You can specify which frequencies the element noise is printed. The frequencies must match the sweep_frequency values defined in the <i>frequency_sweep</i>, otherwise they are ignored.</p> <p>In the element noise output, the elements that contribute the largest noise are printed first. The frequency values can be specified with the NONE or ALL keyword, which either prints no frequencies or every frequency defined in <i>frequency_sweep</i>. Frequency values must be enclosed in parentheses. For example:</p> <ul style="list-style-type: none"> <li>▪ listfreq=(none)</li> <li>▪ listfreq=(all)</li> <li>▪ listfreq=(1.0)</li> <li>▪ listfreq=(1.0G, 2.0G)</li> </ul> <p>The default value is NONE.</p>
listcount	<p>Prints the element noise value to the <i>.lis</i> file, which is sorted from the largest to smallest value. You do not need to print every noise element; instead, you can define <i>listcount</i> to print the number of element noise frequencies. For example, <i>listcount=5</i> means that only the top 5 noise contributors are printed. The default value is 1.</p>
listfloor	<p>Prints the element noise value to the <i>.lis</i> file and defines a minimum meaningful noise value (in <math>V/Hz^{1/2}</math> units). Only those elements with noise values larger than <i>listfloor</i> are printed. The default value is <math>1.0e-14 V/Hz^{1/2}</math>.</p>

Argument	Description
listsources	Prints the element noise value to the <i>.lis</i> file when the element has multiple noise sources, such as a MOSFET, which contains the thermal, shot, and 1/f noise sources. You can specify either ON or OFF: ON prints the contribution from each noise source and OFF does not. The default value is OFF.

### Description

Periodic Time-Dependent noise analysis (*PTDNOISE*) calculates the noise spectrum and the total noise at a point in time. Jitter in a digital threshold circuit can then be determined from the total noise and the digital signal slew rate.

*.MEASURE PTDNOISE* allows for the measurement of these parameters: integnoise, time-point, tdelta-value, slewrate, and strobed jitter. See [Periodic Time-Dependent Noise Analysis \(.PTDNOISE\)](#) in the *HSPICE User Guide: RF Analysis* for details.

### Examples

The following example does a time point sweep.

```
.param f0 = 5.0e8
.sn tones=f0 nharms=4 trinit=10n
.PTDNOISE v(out1) TIME=lin 3 0 2n TDELTA=.1n dec 5 1e5 1e10
+ listfreq=(1e6,1e8)
+ listcount=1
+ listsources=ON
...
```

### See Also

[.HBNOISE](#)  
[.SNNOISE](#)

---

## .PZ

Performs pole/zero analysis.

### Syntax

```
.PZ output input
.PZ ovsrcname
```

---

Argument	Description
input	Input source; the name of any independent voltage or current source.
output	Output variables, which can be: <ul style="list-style-type: none"><li>▪ Any node voltage, <math>V(n)</math>.</li><li>▪ Any branch current, <math>I(branch\_name)</math>.</li></ul>
ov	Output variable: <ul style="list-style-type: none"><li>▪ a node voltage <math>V(n)</math>, or a branch current <math>I(element)</math></li></ul>
srcnam	Input source: <ul style="list-style-type: none"><li>▪ an independent voltage or a current source name</li></ul>

---

### Description

Use to perform Pole/Zero analysis. You do not need to specify `.OP` because the simulator automatically invokes an operating point calculation. See [Pole/Zero Analysis](#) in the *HSPICE User Guide: Simulation and Analysis* for complete information about pole/zero analysis.

### Examples

```
.PZ V(10) VIN
.PZ I(RL) ISORC
```

- In the first pole/zero analysis, the output is the voltage for node 10 and the input is the `VIN` independent voltage source.
- In the second pole/zero analysis, the output is the branch current for the `RL` branch and the input is the `ISORC` independent current source.

### See Also

#### [.DC](#)

[Filters Examples](#), for full demo netlists using the `.PZ` command, including:

- *fbp\_2.sp* (bandpass LCR filter, pole/zero)
- *ninth.sp* (active low pass filter using Laplace elements)



- *fhp4th.sp* (high-pass LCR, fourth-order Butterworth filter, pole-zero analysis)
- *fkerwin.sp* (pole/zero analysis of Kerwin's circuit)
- *flp5th.sp* (low-pass, fifth-order filter, pole-zero analysis)
- *flp9th.sp* (low-pass, ninth-order FNDR, with ideal op-amps, pole-zero analysis)

---

## .SAMPLE

Analyzes data sampling noise.

### Syntax

```
.SAMPLE FS=freq [TOL=val] [NUMF=val]  
+ [MAXFLD=val] [BETA=0|1]
```

---

Argument	Description
FS= <i>freq</i>	Sample frequency in hertz.
TOL	Sampling-error tolerance: the ratio of the noise power (in the highest folding interval) to the noise power (in baseband). The default is 1.0e-3.
NUMF	Maximum number of frequencies that you can specify. The algorithm requires about ten times this number of internally-generated frequencies so keep this value small. The default is 100.
MAXFLD	Maximum number of folding intervals (The default is 10.0). The highest frequency (in hertz) that you can specify is: FMAX=MAXFLD · FS
BETA	Optional noise integrator (duty cycle) at the sampling node: <ul style="list-style-type: none"><li>▪ BETA=0 no integrator</li><li>▪ BETA=1 simple integrator (default)</li></ul> If you clock the integrator (integrates during a fraction of the 1/FS sampling interval), then set BETA to the duty cycle of the integrator.

---

### Description

Use this command to acquire data from analog signals. It is used with the .NOISE and .AC commands to analyze data sampling noise in HSPICE. The SAMPLE analysis performs a noise-folding analysis at the output node.

### See Also

[.AC](#)  
[.NOISE](#)

---

## .SAVE

Stores the operating point of a circuit in a file that you specify in HSPICE only.

### Syntax

```
.SAVE [TYPE=type_keyword] [FILE=save_file]  
+ [LEVEL=level_keyword] [TIME=save_time]
```

---

Argument	Description
TYPE= type_keyword	Storage method for saving the operating point. The type can be one of the following. Default is NODESET. <ul style="list-style-type: none"><li>▪ NODESET: Stores the operating point as a NODESET command. Later simulations initialize all node voltages to these values if you use the .LOAD command. If circuit conditions change incrementally, DC converges within a few iterations.</li><li>▪ IC: Stores the operating point as an IC command. Later simulations initialize node voltages to these values if the netlist includes the .LOAD commands.</li></ul>
save_file	Name of the file that stores DC operating point data. The file name format is <i>save_file.ic#</i> . Default is <i>design.ic0</i> .
level_keyword	Circuit level at which you save the operating point. The level can be one of the following. <ul style="list-style-type: none"><li>▪ ALL (default): Saves all nodes from the top to the lowest circuit level. This option offers the greatest improvement in simulation time.</li><li>▪ TOP: Saves only nodes in the top-level design. Does not save subcircuit nodes.</li><li>▪ SELECT: Enables you to select nodes that you would like to be reported using .PRINT or .PROBE statements.</li><li>▪ NONE: Does not save the operating point.</li></ul>
save_time	Time during transient analysis when HSPICE saves the operating point. HSPICE requires a valid transient analysis command to save a DC operating point. The default is 0.

---

### Description

Use this command to store the operating point of a circuit in a file that you specify. For quick DC convergence in subsequent simulations, use the .LOAD command to input the contents of this file. HSPICE saves the operating point by default, even if the HSPICE input file does not contain a .SAVE command. To

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .SAVE

not save the operating point, specify `.SAVELEVEL=NONE`. You can save the operating point data as either an `.IC` or a `.NODESET` command. A parameter or temperature sweep saves only the first operating point.

The `.SAVE` command only saves one bias point to a file.

**Note:** To save multiple node voltages at different timesteps, it is preferable to use the `.OP` command.

MP and DP do not support saving and loading `*.ic` files. If a netlist contains `.save` or `.load` commands, then `-mp` and `-dp` are disabled.

### Examples

*Example 1* This example saves the operating point corresponding to `.TEMP -25` to a file named `my_design.ic0`.

```
.TEMP -25 0 25
.SAVE TYPE=NODESET FILE=my_design.ic0 LEVEL=ALL
+ TIME=0
```

*Example 2* In this example statement, only the four specified signals are printed in the `test.ic0` file.

```
.SAVE LEVEL=SELECT FILE='test.ic0'
.probe v(in) v(x1.clk) v(x1.xpll.4gout) v(out_n)
```

*Example 3* This example appears in a file where there are eight `.end`'s where there are `.SAVE` lines in every other `.end` (four total). The `save_file` flag is `'6230_lrmf.ic'`. The resultant files are:

```
6230_lrmf.ic0
6230_lrmf.ic1
6230_lrmf.ic2
6230_lrmf.ic3
```

```
.SAVE TYPE=IC TIME=1.72323e-09 FILE='6230_lrmf.ic'
```

### See Also

- [.IC](#)
- [.LOAD](#)
- [.NODESET](#)
- [.OP](#)
- [.PRINT](#)
- [.PROBE](#)

---

## .SENS

Determines DC small-signal sensitivities of output variables for circuit parameters.

### Syntax

```
.SENS ov1 [ov2 ...]
```

---

Argument	Description
ov1 ov2 ...	Branch currents or nodal voltage for DC component-sensitivity analysis

---

### Description

Use this command to determine DC small-signal sensitivities of output variables for circuit parameters.

If the input file includes a `.SENS` command, HSPICE determines DC small-signal sensitivities for each specified output variable relative to every circuit parameter. The sensitivity measurement is the partial derivative of each output variable for a specified circuit element measured at the operating point and normalized to the total change in output magnitude. Therefore, the sum of the sensitivities of all elements is 100%. DC small-signal sensitivities are calculated for:

- resistors
- voltage sources
- current sources
- diodes
- BJTs (including Level 4, the VBIC95 model)
- MOSFETs (Level49 and Level53, Version=3.22).

**Note:** The only BSIM3 model version supported in sensitivity analysis is the BSIM3V3.22 model. BSIMV3.2, BSIM3V3.24, and BSIM3V3.3 models are not supported.

You can perform only one `.SENS` analysis per simulation. Only the last `.SENS` command is used in case more than one is present. The others are discarded with warnings. The amount of output generated from a `.SENS` analysis is dependent on the size of the circuit.

## Examples

In Example 1, the `.SENS v(2)` command is used to find out how sensitive the voltage at node 2 is to change at any element value.

For sensitivity analysis only one element is changed at a time while all other element values are retained at their original value. The output of the `.SENS v(2)` command appears in the list file as follows:

### Example 1

```
v1 1 0 1
r1 1 2 1k
r2 2 0 1k
.SENS v(2)
.end
```

In Example 2, the element sensitivity column lists the absolute change in  $V(2)$  when the element value is changed by unity. As shown, an element sensitivity of  $-250.0000\mu$  for element `r1` indicates that  $v(2)$  decreases by  $250\mu\text{V}$  when `R1` is increased from  $1000\ \Omega$  to  $1001\ \Omega$ . Similarly, an element sensitivity of  $500.0000\text{m}$  for element `v1` indicates that  $v(2)$  increases by  $500\text{mV}$  when `v1` increases by  $1\text{V}$ .

The normalized sensitivity column lists the absolute change in  $v(2)$  when the element value is increased by  $1\%$ . As shown for element `r1`, the normalized sensitivity of  $-2.5000\text{m}$  indicates that  $v(2)$  decreases by  $2.5\text{mV}$  when the value of `r1` is increased by  $1\%$ .

### Example 2

```
dc sensitivities of output v(2)

element element element normalized
name value sensitivity sensitivity
(volts/unit) (volts/percent)

0:r1 1.0000k -250.0000u -2.5000m
0:r2 1.0000k 250.0000u 2.5000m
0:v1 1.0000 500.0000m 5.0000m
```

**Note:** In both columns, a negative sign indicates a decrease and a positive sign indicates an increase in the output variable (in this case,  $v(2)$ ).

## See Also

[.DC](#)

---

## .SHAPE

Defines a shape to be used by the HSPICE field solver.

### Syntax

`.SHAPE snameShape_Descriptor`

---

Argument	Description
<code>sname</code>	Shape name.
<code>Shape_Descriptor</code>	One of the following: <ul style="list-style-type: none"> <li>▪ Rectangle</li> <li>▪ Circle</li> <li>▪ Strip</li> <li>▪ Polygon</li> <li>▪ Trapezoid</li> </ul>

---

### Description

Use this command to define a shape. The field solver uses the shape to describe a cross-section of the conductor.

### See Also

- [.SHAPE \(Defining Rectangles\)](#)
- [.SHAPE \(Defining Circles\)](#)
- [.SHAPE \(Defining Polygons\)](#)
- [.SHAPE \(Defining Strip Polygons\)](#)
- [.SHAPE \(Defining Trapezoids\)](#)
- [.FSOPTIONS](#)
- [.LAYERSTACK](#)
- [.MATERIAL](#)
- [Transmission \(W-element\) Line Examples](#)

---

## .SHAPE (Defining Rectangles)

Defines a rectangle to be used by the HSPICE field solver.

### Syntax

```
.SHAPE RECTANGLE WIDTH=val HEIGHT=val [NW=val] [NH=val]
```

---

Argument	Description
WIDTH	Width of the rectangle (size in the x-direction).
HEIGHT	Height of the rectangle (size in the y-direction).
NW	Number of horizontal (x) segments in a rectangle with a specified width.
NH	Number of vertical (y) segments in a rectangle with a specified height.

---

### Description

Use this keyword to define a rectangle. Normally, you do not need to specify the NW and NH values because the field solver automatically sets these values, depending on the accuracy mode. You can specify both values or only one of these values and let the solver determine the other.

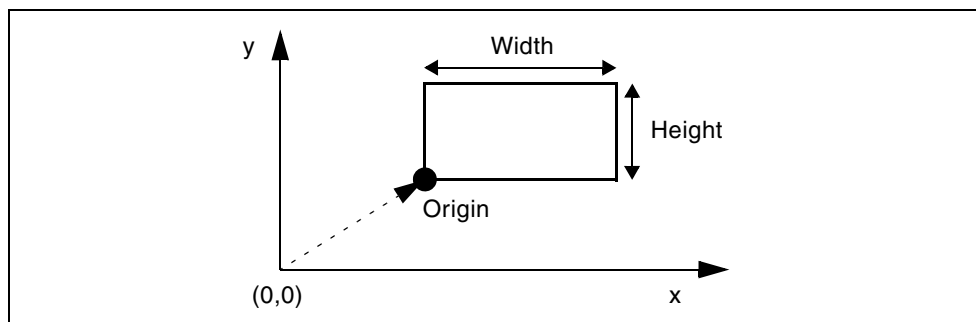


Figure 9 Coordinates of a Rectangle



---

## .SHAPE (Defining Circles)

Defines a circle to be used by the HSPICE field solver.

### Syntax

```
.SHAPE CIRCLE RADIUS=val [N=val]
```

---

Argument	Description
RADIUS	Radius of the circle.
N	Number of segments to approximate a circle with a specified radius.

---

### Description

Use this keyword to define a circle in the field solver. The field solver approximates a circle as an inscribed regular polygon with  $N$  edges. The more edges, the more accurate the circle approximation is.

Do not use the `CIRCLE` descriptor to model actual polygons; instead use the `POLYGON` descriptor.

Normally, you do not need to specify the  $N$  value because the field solver automatically sets this value, depending on the accuracy mode. But you can specify this value if you need to

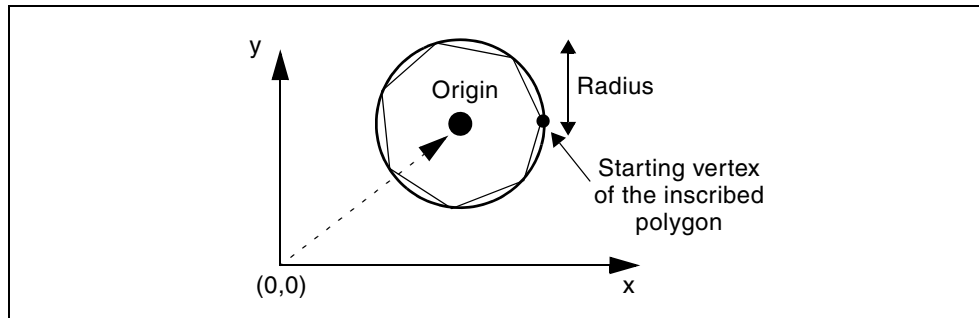


Figure 10 Coordinates of a Circle

---

## .SHAPE (Defining Polygons)

Defines a polygon to be used by the HSPICE field solver.

### Syntax

```
.SHAPE POLYGON VERTEX=(x1y1x2y2 ...)  
+ [N=(n1,n2,...)]
```

---

Argument	Description
VERTEX	(x, y) coordinates of vertices. Listed either in clockwise or counter-clockwise direction.
N	Number of segments that define the polygon with the specified x and y coordinates. You can specify a different N value for each edge. If you specify only one N value, then the field solver uses this value for <i>all</i> edges. For example, the first value of N, n1, corresponds to the number of segments for the edge from (x1 y1) to (x2 y2).

---

### Description

Use this command to define a polygon in a field solver. The specified coordinates are within the local coordinate with respect to the origin of a conductor.

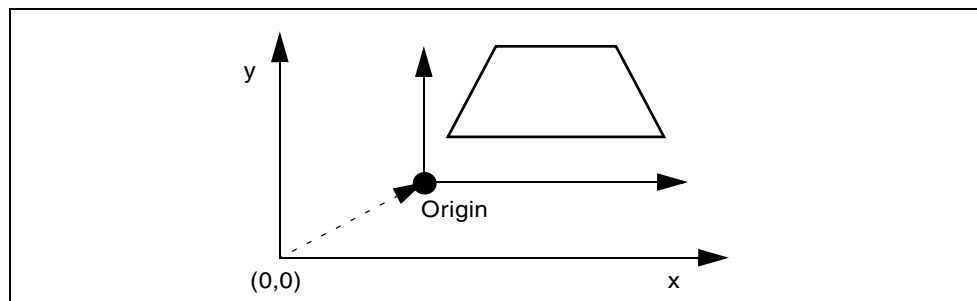


Figure 11 Coordinates of a Polygon

### Examples

Example 1 demonstrates a rectangular polygon using the default number of segments.

#### Example 1

```
.SHAPE POLYGON VERTEX=(1 10 1 11 5 11 5 10)
```

The rectangular polygon specified in Example 2 uses five segments for each edge.

*Example 2*

```
.SHAPE POLYGON VERTEX=(1 10 1 11 5 11 5 10)  
+ N=5
```

Example 3 shows how rectangular polygon uses different number of segments for each edge.

*Example 3*

```
.SHAPE POLYGON VERTEX=(1 10 1 11 5 11 5 10)  
+ N=(5 3 5 3)
```

---

## .SHAPE (Defining Strip Polygons)

Defines a strip polygon to be used by the HSPICE field solver.

### Syntax

```
.SHAPE STRIP WIDTH=val [N=val]
```

---

Argument	Description
WIDTH	Width of the strip (size in the x-direction).
N	Number of segments that define the strip shape with the specified width.

---

### Description

Use this command to define a strip polygon in a field solver. Normally, you do not need to specify the N value because the field solver automatically sets this value, depending on the accuracy mode. But you can specify this value if you need to.

The field solver (filament method) does not support this shape.

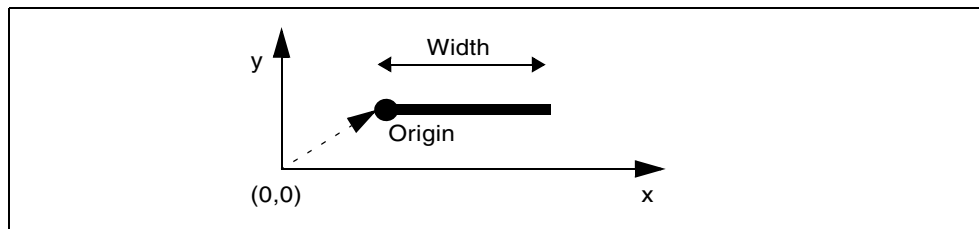


Figure 12 Coordinates of a Strip Polygon

## .SHAPE (Defining Trapezoids)

Defines a trapezoid to be used by the HSPICE field solver.

### Syntax

```
.SHAPE TRAPEZOID TOP=val BOTTOM=val HEIGHT=val
+ [NW=val] [NH=val]
```

Argument	Description
TOP	Top edge length of the trapezoid (size in the x-direction).
BOTTOM	Bottom edge length of the trapezoid (size in the x-direction).
HEIGHT	Height of the trapezoid (size in the y-direction).
NW	Number of horizontal (x) segments in a trapezoid with a specified top and bottom.
NH	Number of vertical (y) segments in a trapezoid with a specified height.

### Description

Use this keyword to define a trapezoid. Normally, you do not need to specify the NW and NH values because the field solver automatically sets these values, depending on the accuracy mode. You can specify both values or only one of these values to let the solver determine the other.

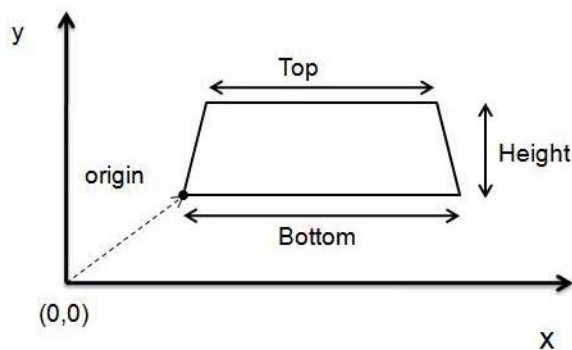


Figure 13 Coordinates of a Trapezoid

---

## .SN

In HSPICE RF, Shooting Newton provides two syntaxes. Syntax #1 is recommended when you are using/making Time Domain sources and measurements (for example, going from .TRAN to .SN). Syntax #2 effectively supports Frequency Domain sources and measurements (and should be used, for example, when going from .HB to .SN).

### Syntax

#### Syntax #1

```
.SN TRES=Tr PERIOD=T [TRINIT=Ti] [SWEEP parameter_sweep]
+ [MAXTRINITCYCLES=integer] [NUMPEROUT=val]
```

#### Syntax #2

```
.SN TONE=F1 NHARMS=N [TRINIT=Ti] [SWEEP parameter_sweep]
+ [MAXTRINITCYCLES=integer] [NUMPEROUT=val]
```

Argument	Description
TRES	Time resolution to be computed for the steady-state waveforms (in seconds).
PERIOD	Expected period T (seconds) of the steady-state waveforms. Enter an approximate value when using for oscillator analysis. The period of the steady-state waveform may be entered either as PERIOD or its reciprocal, TONE.
TRINIT	Transient initialization time. If not specified, the transient initialization time will be equal to the period (for Syntax 1) or the reciprocal of the tone (for Syntax 2).
SWEEP	Parameter sweep. As in all main analyses in HSPICE RF such as .TRAN, .HB, etc., you can specify LIN, DEC, OCT, POI, SWEEPBLOCK, DATA, MONTE, or OPTIMIZE.
MAXTRINITCYCLES	SN stabilization simulation and frequency detection is stopped when the simulator detects that maxtrinitcycles have been reached in the oscnode signal, or when time=trinit, whichever comes first. Minimum cycles is 1.
TONE	Fundamental frequency (in Hz).

Argument	Description
NHARMS	Specifies the number of high-frequency harmonic components to include in the analysis. NHARMS defaults to PERIOD/TRES rounded to nearest integer. NHARMS is required to run subsequent SNAC, SNNOISE, SNXF, and PHASENOISE analyses. When using Syntax #1, NHARMS is computed automatically as NHARMS=Round(PERIOD/TRES).
NUMPEROUT	Allows you to dump more than one period of output to ease waveform viewing. (Your eye doesn't have to struggle in the viewer to connect the end of a waveform period to its beginning.) By default, SN analysis only dumps one period of output.

### Description

Shooting-Newton adds analysis capabilities for PLL components, digital circuits/logic, such as ring oscillators, frequency dividers, phase/frequency detectors (PFDs), and for other digital logic circuits and RF components that require steady-state analysis, but operate with waveforms that are more square wave than sinusoidal. Refer to the *HSPICE User Guide: RF Analysis*, [Steady-State Shooting Newton Analysis](#).

In addition to all .TRAN options, .SN analysis supports the following options:

- .OPTION LOADSNINIT
- .OPTION SAVESNINIT
- .OPTION SNACCURACY
- .OPTION SNMAXITER

### See Also

[.SNAC](#)  
[.SNFT](#)  
[.SNNOISE](#)  
[.SNOSC](#)  
[.SNXF](#)  
[.OPTION LOADSNINIT](#)  
[.OPTION SAVESNINIT](#)  
[.OPTION SNACCURACY](#)  
[.OPTION SNMAXITER \(or\) SN\\_MAXITER](#)

---

## .SNAC

Runs a frequency sweep across a range for the input signal based on a Shooting Newton algorithm.

### Syntax

```
.SNAC frequency_sweep
```

---

Argument	Description
<i>frequency_sweep</i>	<p>Frequency sweep range for the input signal (also referred to as the input frequency band (IFB) or fin). You can specify LIN, DEC, OCT, POI, or SWEEPBLOCK. Specify the nsteps, start, and stop times using the following syntax for each type of sweep:</p> <ul style="list-style-type: none"> <li>▪ LIN <i>nsteps start stop</i></li> <li>▪ DEC <i>nsteps start stop</i></li> <li>▪ OCT <i>nsteps start stop</i></li> <li>▪ POI <i>nsteps freq_values</i></li> <li>▪ SWEEPBLOCK <i>nsteps freq1 freq2 ... freqn</i></li> </ul>

---

### Description

The *frequency\_sweep* runs across a range for the input signal based on a Shooting Newton algorithm. For more information, see [Shooting Newton AC Analysis \(.SNAC\)](#) in the *HSPICE User Guide: RF Analysis*.

### Examples

```
VSRC node1 node2 0 SNAC 1 45
.SNAC DEC 10 1k 10K
```

### See Also

[.HBAC](#)

[.SN](#)

[.SNNOISE](#)



---

## .SNFT

Calculates the Discrete Fourier Transform (DFT) value used for Shooting Newton analysis. Numerical parameters (excluding string parameters) can be passed to the .SNFT command.

### Syntax

Syntax # 1 Alphanumeric input

```
.SNFT output_var [START=value] [STOP=value]
+ [NP=value] [FORMAT=keyword]
+ [WINDOW=keyword] [ALFA=value]
+ [FREQ=value] [FMIN=value] [FMAX=value]
```

Syntax #2 Numerics and expressions

```
.SNFT output_var [START=param_expr1] [STOP=param_expr2]
+ [NP=param_expr3] [FORMAT=keyword]
+ [WINDOW=keyword] [ALFA=param_expr4]
+ [FREQ=param_expr5] [FMIN=param_expr6] [FMAX=param_expr7]
```

---

Argument	Description
output_var	Any valid output variable, such as voltage, current, or power.
START	Start of the output variable waveform to analyze. Defaults to the START value in the .SN command, which defaults to 0.
FROM	Alias for START in .SNFT commands.
STOP	End of the output variable waveform to analyze. Defaults to the TSTOP value in the .SN command.
TO	Alias for STOP, in .SNFT commands.
NP	Number of points to use in the SNFT analysis. NP must be a power of 2. If NP is not a power of 2, HSPICE automatically adjusts it to the closest higher number that is a power of 2. The default is 1024.
FORMAT	Output format: <ul style="list-style-type: none"> <li>▪ NORM= normalized magnitude (default)</li> <li>▪ UNORM=unnormalized magnitude</li> </ul>

Argument	Description
WINDOW	Window type to use: <ul style="list-style-type: none"> <li>▪ RECT=simple rectangular truncation window (default).</li> <li>▪ BART=Bartlett (triangular) window.</li> <li>▪ HANN=Hanning window.</li> <li>▪ HAMM=Hamming window.</li> <li>▪ BLACK=Blackman window.</li> <li>▪ HARRIS=Blackman-Harris window.</li> <li>▪ GAUSS=Gaussian window.</li> <li>▪ KAISER=Kaiser-Bessel window.</li> </ul>
ALFA	Parameter to use in GAUSS and KAISER windows to control the highest side-lobe level, bandwidth, and so on. $1.0 \leq ALFA \leq 20.0$ The default is 3.0
FREQ	Frequency to analyze. If FREQ is non-zero, the output lists only the harmonics of this frequency, based on FMIN and FMAX. HSPICE also prints the THD for these harmonics. The default is 0.0 (Hz).
FMIN	Minimum frequency for which HSPICE prints SNFT output into the listing file. THD calculations also use this frequency. $T=(STOP-START)$ The default is $1.0/T$ (Hz).
FMAX	Maximum frequency for which HSPICE prints SNFT output into the listing file. THD calculations also use this frequency. The default is $0.5*NP*FM IN$ (Hz).

### Description

Use this command to calculate the Discrete Fourier Transform (DFT) spectrum analysis values for Shooting Newton analysis. It uses internal time point values to calculate these values. A DFT uses sequences of time values to determine the frequency content of analog signals in circuit simulation. You can pass numerical parameters/expressions (but no string parameters) to the .SNFT command. The output goes to a file with extension *.snft#*.

You can specify only one output variable in an .SNFT command. The following is an incorrect use of the command because it contains two variables in one .SNFT command:

## Examples

*Example 1* Correctly designates the variables per .SNFT command.

```
.SNFT v(1)
.SNFT v(1,2) np=1024 start=0.3m stop=0.5m freq=5.0k
+ window=kaiser alfa=2.5
.SNFT I(rload) start=0m to=2.0m fmin=100k fmax=120k
+ format=unorm
.SNFT par('v(1) + v(2)') from=0.2u stop=1.2u
+ window=harris
```

*Example 2* Generates a .snft0 file for the SNFT of v(1) and a .snft1 file for the SNFT of v(2).

```
.SNFT v(1) np=1024
.SNFT v(2) np=1024
```

## See Also

[.SN](#)

---

## .SNNOISE

Runs a periodic, time-varying AC noise analysis based on a Shooting Newton algorithm.

### Syntax

```
.SNNOISE [output] [insrc] [frequency_sweep]  
+ [n1, +/-1]  
+[listfreq=(frequencies|none|all)> [listcount=val]]  
+[listfloor=val] [listsources=on|off]
```

---

Argument	Description
output	Output node, pair of nodes, or 2-terminal element that the equivalent noise output references.
insrc	Input source.
frequency_sweep	Frequency sweep range for the input signal. You can specify LIN, DEC, OCT, POI, SWEEPBLOCK, DATA, MONTE, or OPTIMIZE sweeps.
n1, +/-	Index term defining the output frequency band at which the noise is evaluated. The output frequency is computed according to $f_{out} =  n1 * f1 +/- f_{in} $ , where $f1$ is the fundamental tone (inverse of fundamental period) and $f_{in}$ is from the frequency sweep.
listfreq	Prints the element noise value to the <i>.lis</i> file; the default is none.
listcount	Prints the element noise value to the <i>.lis</i> file, sorted from the largest to smallest value.
listfloor	Prints the element noise value to the <i>.lis</i> file and defines a minimum meaningful noise value. Only those elements with noise values larger than listfloor are printed. The default value is 1.0e-14 V/sqrt(Hz).
listsources	Prints the element noise value to the <i>.lis</i> file when the element has multiple noise sources. The default is off.

---

### Description

The functionality for the .SNNOISE command is similar to the Harmonic Balance (HBNOISE command) for periodic, time-varying AC noise analysis, but

the Shooting Newton-based algorithm completes the analysis in a much faster run time with the same result.

### Examples

```
.SNNOISE V(n1,n2) RIN DEC 10 1k 10k 0 -1
```

### See Also

[.HBNOISE](#)

[.SN](#)

[.SNAC](#)

---

## .SNOSC

Performs oscillator analysis on autonomous (oscillator) circuits. As with regular Shooting Newton analysis, input might be specified in terms of time or frequency values.

### Syntax

#### Syntax #1

```
.SNOSC TONE=F1 NHARMS=H1 [TRINIT=Ti] OSCNODE=N1
+ [MAXTRINITCYCLES=N] [SWEEP PARAMETER_SWEEP]
```

#### Syntax #2

```
.SNOSC TRES=Tr PERIOD=Tp [TRINIT=Tr] OSCNODE=N1
+ [MAXTRINITCYCLES=I] SWEEP PARAMETER_SWEEP
```

---

Argument	Description
TONE	Approximate value for oscillation frequency (Hz). The search for an exact oscillation frequency begins from this value.
NHARMS	Number of harmonics to be used for oscillator SN analysis.
OSCNODE	Node used to probe for oscillation conditions. This node is automatically analyzed to search for periodic behavior near the TONE or PERIOD value specified.
TRINIT	Transient initialization time. If not specified, the transient initialization time is equal to the period (for Syntax 1) or the reciprocal of the tone (for Syntax 2). For oscillators we recommend specifying a transient initialization time since the default initialization time is usually too short to effectively stabilize the circuit.
MAXTRINITCYCLES	SN stabilization simulation and frequency detection is stopped when the simulator detects that MAXTRINITCYCLES have been reached in the <code>oscnode</code> signal, or when <code>time=trinit</code> , whichever comes first. Minimum cycles is 1.
TRES	Time resolution to be computed for the steady-state waveforms (in seconds). The period of the steady-state waveform may be entered either as <code>PERIOD</code> or its reciprocal, <code>TONE</code> .

Argument	Description
PERIOD	Expected period T (seconds) of the steady-state waveforms. Enter an approximate value when using for oscillator analysis.
SWEEP	Type of sweep. You can sweep up to three variables. You can specify either LIN, DEC, OCT, POI, SWEEPBLOCK, DATA, OPTIMIZE, or MONTE. Specify the nsteps, start, and stop frequencies using the following syntax for each type of sweep: <ul style="list-style-type: none"> <li>▪ LIN <i>nsteps start stop</i></li> <li>▪ DEC <i>nsteps start stop</i></li> <li>▪ OCT <i>nsteps start stop</i></li> <li>▪ POI <i>nsteps freq_values</i></li> <li>▪ SWEEPBLOCK <i>nsteps freq1 freq2 ... freqn</i></li> <li>▪ DATA=<i>dataname</i></li> <li>▪ OPTIMIZE=OPT<i>xxx</i></li> <li>▪ MONTE=<i>val</i></li> </ul>

### Description

Use this command to invoke oscillator analysis on autonomous (oscillator) circuits. The SNOSC command is very effective for ring oscillator circuits, and oscillators that operate with piecewise linear waveforms (HBOSC is superior for sinusoidal waveforms). As with the Harmonic Balance approach, the goal is to solve for the additional unknown oscillation frequency. This is accomplished in Shooting Newton by considering the period of the waveform as an additional unknown, and solving the boundary conditions at the waveform endpoints that coincide with steady-state operation. As with regular Shooting Newton analysis, input might be specified in terms of time or frequency values. See the examples, below.

### Examples

*Example 1* Performs an oscillator analysis searching for periodic behavior after an initial transient analysis of 10 ns. This example uses nine harmonics while searching for a oscillation at the gate node.

```
.SNOSC tone=900Meg nharms=9 trinit=10n oscnode=gate
```

*Example 2* Performs an oscillator analysis searching for frequencies in the vicinity of 2.4 Ghz. This example uses 11 harmonics and a search at the drainP.

```
.SNOSC tone=2400MEG nharms=11 trinit=20n oscnode=drainP
```

*Example 3* Presents another equivalent method to define the OSCNODE information through a zero-current source. Example 3 is identical to Example 2, except that the OSCNODE information is defined by a current

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .SNOSC

*source in the circuit. Only one such current source is needed and its current source must be 0.0 with the SNOSC OSCNODE identified by the SNOSCVPROBE keyword.*

```
ISRC drainP 0 SNOSCVPROBE  
.SNOSC tone = 2.4 G nharms = 1 trinit=20n
```

### See Also

- [.HB](#)
- [.OPTION HBFREQABSTOL](#)
- [.OPTION HBFREQRELTOL](#)
- [.OPTION HBOSCMAXITER \(or\) HBOSC\\_MAXITER](#)
- [.OPTION HBPROBETOL](#)
- [.OPTION HBTRANFREQSEARCH](#)
- [.OPTION HBTRANINIT](#)
- [.OPTION HBTRANPTS](#)
- [.OPTION HBTRANSTEP](#)
- [.PRINT](#)
- [.PROBE](#)



---

## .SNXF

Calculates the transfer function from the given source in the circuit to the designated output.

### Syntax

```
.SNXF out_varfreq_sweep
```

Argument	Description
out_var	I(2_port_elem) or V(n1<, n2>)
freq_sweep	Sweep of type LIN, DEC, OCT, POI, or SWEEPBLOCK. Specify the nsteps, start, and stop times using the following syntax for each type of sweep: <ul style="list-style-type: none"> <li>▪ LIN nsteps start stop</li> <li>▪ DEC nsteps start stop</li> <li>▪ OCT nsteps start stop</li> <li>▪ POI nsteps freq_values</li> <li>▪ SWEEPBLOCK = BlockName</li> </ul> Specify the frequency sweep range for the output signal. HSPICE RF determines the offset frequency in the input sidebands; for example, $f_1 = \text{abs}(f_{out} - k \cdot f_0)$ s.t. $f_1 \leq f_0/2$ . The $f_0$ is the steady-state fundamental tone and $f_1$ is the input frequency.

---

### Description

Use this command in HSPICE RF to calculate the transfer function from the given source in the circuit to the designated output. The functionality for the .SNXF command is similar to the Harmonic Balance (.HBXF) command for periodic, time-varying AC noise analysis, but the Shooting Newton based algorithm completes the analysis in a much faster run time with the same result.

### Examples

In this example, the trans-impedance from `isrc` to `v(1)` is calculated based on the HB analysis.

```
.hb tones=1e9 nharms=4
.snxf v(1) lin 10 1e8 1.2e8
.print snxf tfv(isrc) tfi(n3)
```

### See Also

[.HB](#)

**Chapter 2: HSPICE and HSPICE RF Netlist Commands**  
**.SNXF**

**.HBAC**  
**.HBNOISE**  
**.HBOSC**  
**.PRINT**  
**.PROBE**

## .STATEYE

Enables use of statistical eye diagram analysis.

### Syntax

```
.STATEYE T=time_interval Trf=rise_fall_time
+ [Tr=rise_time] [Tf=fall_time]
+ Incident_port=idx1, [idx2, ... idxN]
+ Probe_port=idx1, [idx2, ... idxN]
+ [Tran_init=n_periods]
+ [V_low=val] [V_high=val]
+ [T_resolution=n] [V_resolution=n]
+ [TD_In=val] [TD_Probe=val]
+ [VD_range=val] [TD_NUI=n] [Edge=1|2]
+ [Pattern_max=n] [Pattern_repeat=n]
+ [SAVE_TR=ascii] [LOAD_TR=ascii] [SAVE_DIR=string]
+ [Ignore_Bits=n] [AMIGW_Nbit=n]
```

Argument	Description
T	Time (in seconds) of single bit width of the incident signal, normally referred as Unit Interval (UI)
Trf	Single value (in seconds) to set both the rise and fall times of the incident pulse
Tr	Rise time (in seconds) of the incident port
Tf	Fall time (in seconds) of the incident port
Incident_port	An array of the index numbers of the incident port elements
Probe_port	An array of the index numbers of the probing port elements
V_low	Low voltage level of the incident pulse. The value is used when the voltage level is not specified in the incident port(s).
V_high	High voltage level of the incident pulse. The value is used when the voltage level is not specified in the incident port(s).
Tran_init	An integer number that specifies the numbers of unit intervals (T) that is used by the initial transient analysis to determine the response of the system. Default value is 60.

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

.STATEYE

Argument	Description
T_resolution	An integer number used to specify the probability density function (PDF) image resolution of the time axis. Default value is 200.
TD_In	Applies specified time delay to the incident pulse/step in the initial transient analysis. Default value is 0 (no delay).
TD_Probe	When a positive time value is specified, StatEye only uses initial transient analysis waveforms after the specified time for the eye diagram generation. Default value is 0.
V_resolution	An integer number used to specify the probability density function (PDF) image resolution of the voltage axis. Default value is 200.
VD_range	Specifies voltage display (output data) range. By default, the .StatEye analysis engine automatically determines the optimum voltage display range. Specifying VD_range can enlarge the display range.
TD_NUI	An integer number to specify time display range relative to the unit interval. Default value is 2 to display a single eye. TD_NUI value may be from 1 to 5. The higher value requires a larger data size.
EDGE	Number of edges to be used. <ul style="list-style-type: none"><li>▪ 1: Conventional statistical eye generation using the single pulse response (default).</li><li>▪ 2: Double-edge mode. The rising and falling edges are evaluated separately.</li></ul>
PATTERN_MAX	Limits the number of bits to be examined for a custom bit pattern specified with LFSR/PAT in incident Port-element(s). For example, if PATTERN_MAX=100 is specified, StatEye examines only the first 100 bits in the LFSR/PAT sources. This keyword is especially effective when LFSR is used with very high (over 20-bit) feedback tap(s) since it generates an extremely long bit stream. The default value of the PATTERN_MAX is $2^{21}$ to provide sufficient accuracy for most cases. When PATTERN_MAX is not specified or PATTERN_MAX=0 is specified, StatEye examines all the given pattern(s).
PATTERN_REPEAT	When a positive number, $n$ , is specified, .StatEye repeats pattern examination $n$ times until the number of bits hits the Pattern_max value. Default=0 (no repeats).
SAVE_TR=ascii	Saves initial transient data in text files.

Argument	Description
LOAD_TR=ascii	Loads initial transient data from text files when available.
SAVE_DIR= <i>string</i>	Specifies a target directory other than the one specified by the <code>-o</code> command option for the SAVE_TR and LOAD_TR transient files.
Ignore_Bits= <i>n</i>	When a positive number is specified, StatEye's pattern-specific eye diagram generation process ignores the first <i>n</i> bits. The value is overridden by one specified in the AMI parameter file when one exists for each probe port.
AMIGW_Nbit= <i>n</i>	Specifies the number of bits per each AMI_GetWave call (see <a href="#">amicheck Output</a> ). For example, when a given pattern has 10,000 bits and AMIGW_Nbit=1000, StatEye sequentially calls AMI_GetWave 10 times with stream waveform of 1000 bits.  The default value is 1000. For more details about AMI_GetWave, see chapter 10 of the IBIS version 5.0 specification.

### Description

Use this command to perform statistical eye analysis to evaluate high-speed serial interfaces.

**Note:** The `.STATEYE` command is invoked using the `hspicerf` executable on the command line for this release, not `hspice`. However, running a statistical eye analysis only requires an HSPICE license token.

The statistical eye diagram is a fundamental performance metric for high-speed serial interfaces in the bit error rate (BER). When setting up a Statistical Eye Analysis, the Port element is used to designate the incident (input) and probe (output) ports for the system to be analyzed. Ports can be specified as single-ended or mixed mode. Random jitter can be applied to each incident and probe point in the system. Each incident port acts as random bit pattern source with specified voltage magnitude. If an incident port element does not have a time domain voltage magnitude specification, the default values, `V_high=1.0`, `V_low=-1.0` are used. Probe ports are used as observation points where `.PRINT`, `.PROBE`, and `.MEASURE` commands can be defined.

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .STATEYE

#### Examples

```
.STATEYE T=400p Trf=20p
+ incident_port= 1, 2
+ probe_port= 3, 4
+ Rj = 5p, 5p, 2p, 2p tran_init= 50
+ T_resolution= 300 V_resolution= 300
```

#### See Also

[.MEASURE \(or\) .MEAS](#)

[.PRINT](#)

[.PROBE](#)

[Statistical Eye Analysis](#)

---

## .STIM

Uses the results (output) of one simulation as input stimuli in a new simulation in HSPICE.

### Syntax

General Syntax:

```
.STIM [tran|ac|dc] PWL|DATA|VEC
+ [filename=output_filename ...]
```

### PWL Source Syntax (Transient Analysis Only)

```
.STIM [tran] PWL [filename=output_filename]
+ [name1=] ovar1 [node1=n+] [node2=n-]
+ [[name2=] ovar2 [node1=n+] [node2=n-] ...]
+ [from=val] [to=val] [npoints=val]
.STIM [tran] PWL [filename=output_filename]
+ [name1=] ovar1 [node1=n+] [node2=n-]
+ [[name2=] ovar2 [node1=n+] [node2=n-] ...]
+ indepvar=[(] t1 [t2 ... ()]]
```

### Data Card Syntax

```
.STIM [tran|ac|dc] DATA [filename=output_filename]
+ dataname [name1=] ovar1
+ [[name2=] ovar2 ...] [from=val] [to=val]
+ [npoints=val] [indepout=val]
.STIM [tran | ac | dc] DATA [filename=output_filename]
+ dataname [name1=] ovar1
+ [[name2=] ovar2 ...] indepvar=[(] t1 [t2 ... ()]]
+ [indepout=val]
```

### Digital Vector File Syntax (Transient Analysis Only)

```
.STIM [tran] VEC [filename=output_filename]
+ vth=val vtl=val [voh=val] [vol=val]
+ [name1=] ovar1 [[name2=] ovar2 ...]
+ [from=val] [to=val] [npoints=val]

.STIM [tran] VEC [filename=output_filename]
+ vth=val vtl=val [voh=val] [vol=val]
+ [name1=] ovar1 [[name2=] ovar2 ...]
+ indepvar=[(] t1 [t2 ... ()]]
```

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

.STIM

---

Argument	Description
tran   ac   dc	Simulation type: transient, AC, or DC.
filename	Output file name. If you do not specify a file, HSPICE uses the input filename.
name1	PWL Source Name that you specify. The name must start with V (for a voltage source) or I (for a current source). Or—Name of a parameter of the data card to generate.
ovar1	Output variable that you specify. <i>ovar</i> can be: <ul style="list-style-type: none"><li>▪ Node voltage.</li><li>▪ Element current.</li><li>▪ Parameter string. If using a parameter string you must specify <i>name1</i>. For example: v(1), i(r1), v(2,1), par('v(1)+v(2)')</li></ul>
dataname	Name of the data card to generate.
node1	Positive terminal node name.
node2	Negative terminal node name.
from	Time to start output of simulation results. For transient analysis, it uses the time units that you specified. Cannot use with indepvar.
npoints	Number of output time points or independent-variable points.
to	Time to terminate output of simulation results. For transient analysis, it uses the time units that you specified. The <i>from</i> value can be greater than the <i>to</i> value. Cannot use with indepvar.
indepvar	Dispersed (independent-variable) time points. Specify dispersed time points in increasing order. Replaces the “from” and “to” construct.
indepout	Indicates whether to generate the independent variable column. <ul style="list-style-type: none"><li>▪ indepout, indepout=1, or on, produces the independent variable column. You can specify the independent-variables in any order.</li><li>▪ indepout= 0 or off (default) does not create an independent variable column. You can place the indepout field anywhere after the ovar1 field.</li></ul>
vth	High voltage threshold.
vtl	Low voltage threshold.



Argument	Description
voh	Logic-high voltage for each output signal.
vol	Logic-low voltage for each output signal.

### Description

Use this command to reuse the results (output) of one simulation as input stimuli in a new simulation.

The `.STIM` command specifies:

- Expected stimulus (PWL Source, DATA CARD, or VEC FILE).
- Signals to transform.
- Independent variables.

One `.STIM` command produces one corresponding output file.

For additional information, see “[Reusing Simulation Output as Input Stimuli](#)” in the *HSPICE User Guide: Simulation and Analysis*.

### Examples

In Example 1, the `.STIM` command creates a file “test.pwl0\_tr0”, having a voltage source “v0” applied between nodes neg and 0 (ground). It has a PWL source function based on the voltage of node n0 during the time 0.0ns to 5.0ns with 10 points.

#### Example 1

```
.stim tran pwl filename=test v0=v(n0) node1=neg
+ node2=0 from=0.0ns to=5ns npoints=10
```

Example 2: In this example the “from and to” construct is used:

```
.stim tran data filename=new PWL v(2) from=start to=end
```

Example 3: In this example, the `indepvar` construct replaces “from and to”; using both constructs results in an error.

```
.stim tran pwl filename=new v(2) indepvar=(2n 3n 4n)
```

### See Also

[.DOUT](#)  
[.MEASURE \(or\) .MEAS](#)

**Chapter 2: HSPICE and HSPICE RF Netlist Commands**  
**.STIM**

**.PRINT**  
**.PROBE**

## .STORE

Starts a store operation to create checkpoint files describing a running process during transient analysis (HSPICE).

### Syntax

```
.STORE [file=checkpoint_file] [time=time1]  
+ [repeat=checkpoint_interval]
```

Argument	Description
file= <i>checkpoint_file</i>	Checkpoint files are saved as <i>checkpoint_file.store.gz</i> and <i>checkpoint_file.tar</i> , if not specified, the default checkpoint file name prefix is same as the HSPICE output file.
time= <i>time1</i>	Collects checkpoint data beginning at <i>time1</i> after the start of transient analysis. It then updates the checkpoint data every 21,600 wall-clock seconds if no checkpoint interval is specified.
repeat= checkpoint_interval	If you specify a nonzero <i>checkpoint_interval</i> , new checkpoint data is collected at every <i>checkpoint_interval</i> , starting at transient time=0 and overwriting previous interval checkpoint data. If a nonzero <i>time1</i> is specified, checkpoint data is collected at $time1 + checkpoint\_interval * n$ , where <i>n</i> is an integer. <i>Checkpoint_interval</i> is always calculated based on <i>time1</i> . If repeat=0, the store operation is disabled. If you set both time=0 and repeat=0, checkpoint data is saved at transient time=0 only.
trantime=flag	<ul style="list-style-type: none"> <li>▪ If flag is not equal to 1 (!=1), or trantime is not specified, <i>time1</i> and <i>checkpoint_interval</i> are taken as wall-clock time.</li> <li>▪ If flag=1, <i>time1</i> and <i>checkpoint_interval</i> will be taken as simulation time; checkpoint will start at simulation time <math>time1 + checkpoint\_interval * n</math>.</li> </ul> To avoid too-frequent checkpoints, <i>checkpoint_interval</i> adjusts automatically if the wall-clock time interval of the store operation is less than 7200 seconds.

### Description

Use this command in a netlist to trigger a restore operation by creating checkpoint files describing a running process during transient analysis; the operating system can later reconstruct the process from the contents of this file. This feature is not supported in HSPICE-RF.

The shortest period for a `checkpoint_interval` is 7,200 seconds, anything shorter than that defaults to 7,200 seconds automatically.

If the netlist contains more than one `.store` statement, only the last statement takes effect.

The restore operation is done on the command-line with the `restore` keyword. See [Chapter 1, HSPICE and HSPICE RF Application Commands](#) for more information. For usage requirements and additional information, see [Storing and Restoring Checkpoint Files \(HSPICE\)](#) in the *HSPICE User Guide: Simulation and Analysis*.

### Examples

Example 1: Checkpoint data is saved at 50 wall-clock seconds of transient analysis in `chk_50.store.gz` and `chk_50.tar` files. Checkpoint data is also updated every 21,600 wall-clock seconds.

```
.store file=chk_50 time=50
```

Example 2: After transient analysis starts, the store operation occurs at 7,200 wall-clock seconds, 14,400 wall-clock seconds, 21,600 wall-clock seconds, and so on. Each new time interval overwrites previous interval checkpoint files.

```
.store repeat=7200
```

Example 3: After transient analysis starts, the store operation occurs at 300 wall-clock seconds, 7,500 wall-clock seconds, 14,700 wall-clock seconds, and so on. Each new time interval overwrites previous interval check-point files.

```
.store time=300 repeat=7200
```

Example 4: This statement turns off the store operation.

```
.store repeat=0
```

Example 5: The store operation saves the data at the time transient analysis begins in `outputfile.store.gz` and `outputfile.tar` files.

```
.store time=0 repeat=0
```

Example 6: The first store operation starts at simulation time 1ns, then repeats every 10ns. The repeat value is adjusted if the wall-clock time interval of the store operation less than 7,200 seconds.

```
store trantime=1 time=1n repeat=10n
```

Example 7: The first store operation starts at simulation time 10ns, then repeat every 10ns. The repeat value is adjusted if the wall-clock time interval of the store operation is less than 7,200 seconds.

```
.store trantime=1 repeat=10n
```

Example 8: The store operation starts at simulation time 0.00001s, but does not repeat.

```
.store trantime=1 time=0.00001
```

Example 9: HSPICE does not do a store operation.

```
.store trantime=1
```

### See Also

[.TRAN](#)

---

## .SUBCKT

Defines a subcircuit in a netlist.

### Syntax

#### Nodes and Parameters

```
.SUBCKT subnam n1 n2 n3 ... [param=val]
.ENDS
.SUBCKT SubNamePinList [SubDefaultsList]
.ENDS
```

#### Parameter String

```
.SUBCKT subnam n1 n2 n3 ... [param=str('string')]
.ENDS
```

#### Isomorphic Analyses

```
.SUBCKT analyses_sb [start=p1 stop=p2 steps=p3]
.DC ...
.AC ...
.TRAN ...
.ENDS analyses_sb
```

...followed by

```
x1 analyses_sb [start=a1] [stop=a2] [steps=a3]
x2 analyses_sb [start=b1] [stop=b2] [steps=b3]
```

---

Argument	Description
subnam	Reference name for the subcircuit model call.
n1...	Node numbers for external reference; cannot be the ground node (0, gnd, ground, gnd!). Any element nodes that are in the subcircuit, but are not in this list are strictly local with three exceptions: <ul style="list-style-type: none"> <li>▪ Ground node (0, gnd, ground, gnd!).</li> <li>▪ Nodes assigned using BULK=node in MOSFET or BJT models.</li> <li>▪ Nodes assigned using the .GLOBAL command.</li> </ul>
parnam	Parameter name set to a value. Use only in the subcircuit. To override this value, assign it in the subcircuit call or set a value in a .PARAM command.
SubDefaultsList	<i>SubParam1=Expression [SubParam2=Expression...]</i>

Argument	Description
<code>analysis_sb</code>	Reference name for the isomorphic analyses that can be run in a subckt block.
<code>p1...p2...p3</code>	Parameters specified for the start, stop, and number of steps.

### Description

Use this command to define a subcircuit in your netlist. You can create a subcircuit description for a commonly used circuit and include one or more references to the subcircuit in your netlist.

When you use hierarchical subcircuits, you can pick default values for circuit elements in a `.SUBCKT` command. You can use this feature in cell definitions to simulate the circuit with typical values.

The isomorphic analyses feature enables you to run unrelated analyses (`.DC`, `.AC`, and `.TRAN`) many times during a simulation by grouping the set of analyses into a subcircuit, which performs multiple analyses in one simulation with calls to the subcircuit. The usage model is: Specify the analyses commands within the subckt definition block and then instantiate the subckt to perform the analyses. Each call of the subcircuit is treated as an individual analysis with its own set of parameters.

Use the `.ENDS` command to terminate a `.SUBCKT` command.

**Note:** Using `-top subck_name` on the command line effectively eliminates the need for the `.subckt subckt_name` and `.ends subckt_name`.

## Examples

*Example 1* Defining two subcircuits: SUB1 and SUB2. These are resistor-divider networks, whose resistance values are parameters (variables). The X1, X2, and X3 commands call these subcircuits. Because the resistor values are different in each call, these three calls produce different subcircuits.

```
*FILE SUB2.SP TEST OF SUBCIRCUITS
.OPTION LIST ACCT
  V1 1 0 1
.PARAM P5=5 P2=10
.SUBCKT SUB1 1 2 P4=4
  R1 1 0 P4
  R2 2 0 P5
  X1 1 2 SUB2 P6=7
  X2 1 2 SUB2
.ENDS
*
.MACRO SUB2 1 2 P6=11
  R1 1 2 P6
  R2 2 0 P2
.EOM
  X1 1 2 SUB1 P4=6
  X2 3 4 SUB1 P6=15
  X3 3 4 SUB2
*
.MODEL DA D CJA=CAJA CJP=CAJP VRB=-20
  IS=7.62E-18
+ PHI=.5 EXA=.5 EXP=.33
.PARAM CAJA=2.535E-16 CAJP=2.53E-16
.END
```



**Example 2** *Implementing an inverter that uses a Strength parameter. By default, the inverter can drive three devices. Enter a new value for the Strength parameter in the element line to select larger or smaller inverters for the application.*

```
.SUBCKT Inv a y Strength=3
  Mp1 <MosPinList> pMosMod L=1.2u
  W='Strength * 2u'
  Mn1 <MosPinList> nMosMod L=1.2u
  W='Strength * 1u'
.ENDS
...
xInv0 a y0 Inv $ Default devices: p device=6u,
             $ n device=3u
xInv1 a y1 Inv Strength=5 $ p device=10u,
             n device=5u
xInv2 a y2 Inv Strength=1 $ p device= 2u,
             n device=1u
...
```

**Example 3** *Implementing an IBIS model (in HSPICE only) that uses string parameters to specify the IBIS file name and IBIS model name.*

```
* Using string parameters
.subckt IBIS vccq vss out in
+ IBIS_FILE=str('file.ibs')
+ IBIS_MODEL=str('ibis_model')
ven en 0 vcc
B1 vccq vss out in en v0dq0 vccq vss
+ file= str(IBIS_FILE) model=str(IBIS_MODEL)
.ends
```

**Example 4** *Specifying Isomorphic Analyses*

```
.subckt analyses_sb start_dc=-25 stop_dc=25 steps_dc=5
+ steps_tran=1n stop_tran=10n
.DC TEMP start_dc stop_dc steps_dc
.TRAN steps_tran stop_tran
.ends analyses_sb
...
x1 analyses_sb start_dc=25 stop_dc=75 steps_dc=10
x2 analyses_sb steps_tran=2n
x3 analyses_sb
```

Example 4 specifies both .DC and .TRAN analyses within the subckt. To invoke these analyses you can call the subckts.

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .SUBCKT

- Each subckt call will perform DC and Transient analysis.
- Parameters defined in the subcircuit calls will override the default values specified in the subcircuit definition.
- If parameters are not defined in the subckt calls they will take the default values given in the subcircuit.

#### See Also

[.ENDS](#)

[.EOM](#)

[.MACRO](#)

[.MODEL](#)

[.OPTION LIST](#)

[.PARAM \(or\) .PARAMETER \(or\) .PARAMETERS](#)

[Isomorphic Analyses in Subckt Blocks](#)

---

## .SURGE

Automatically detects and reports a current surge that exceeds the specified surge tolerance in HSPICE RF.

### Syntax

```
.SURGE surge_thresholdsurge_widthnode1 [node2 ...noden]
```

---

Argument	Description
<i>surge_threshold</i>	Minimum absolute surge current.
<i>surge_width</i>	Defines the minimum duration of a surge.
<i>noden</i>	Any valid node name at current or lower subcircuit level.

---

### Description

Use this command to automatically detect and report a current surge that exceeds the specified surge tolerance. The command reports any current surge that is greater than *surge\_threshold* for a duration of more than *surge\_width*.

*Surge current* is defined as the current flowing into or out of a node to the lower subcircuit hierarchy.

### Examples

In this example, the .SURGE command detects any current surge that has an absolute amplitude of more than 1mA, and that exceeds 100ns, x(xm.x1.a), x(xm.x2.c), and x(xn.y).

```
.SUBCKT sa a b
...
.ENDS
.SUBCKT sb c d
...
.ENDS
.SUBCKT sx x y
x1 x y sa
x2 x a sb
.ENDS
xm 1 2 sx
xn 2 a sx
.SURGE 1mA 100ns xm.x1.a xm.x2.c xn.y
```

---

## .SWEEPBLOCK

Creates a sweep whose set of values is the union of a set of linear, logarithmic, and point sweeps in HSPICE RF.

### Syntax

```
.SWEEPBLOCK swblocknamesweepspec [sweepspec  
+ [sweepspec [...]]]
```

---

Argument	Description
<i>swblockname</i>	Assigns a name to SWEEPBLOCK.
<i>sweepspec</i>	You can specify an unlimited number of <i>sweepspec</i> parameters. Each <i>sweepspec</i> can specify a linear, logarithmic, or point sweep by using one of the following forms: <i>start stop increment lin npoints start stop dec npoints start stop oct npoints start stop poi npoints p1 p2 ...</i>

---

### Description

Use this command to create a sweep whose set of values is the union of a set of linear, logarithmic, and point sweeps.

You can use this command to specify DC sweeps, parameter sweeps, AC, and HBAC frequency sweeps, or wherever HSPICE accepts sweeps.

For additional information, see “[SWEEPBLOCK in Sweep Analyses](#)” in the *HSPICE User Guide: RF Analysis*.

### Examples

The following example specifies a logarithmic sweep from 1 to 1e9 with more resolution from 1e6 to 1e7:

```
.sweepblock freqsweep dec 10 1 1g dec 1000 1meg 10meg
```

### See Also

- [.AC](#)
- [.DC](#)
- [.ENV](#)
- [.HB](#)
- [.HBAC](#)
- [.HBLSP](#)
- [.HBNOISE](#)
- [.HBOSC](#)

.HBXF  
.PHASENOISE  
.TRAN

---

## .TEMP (or) .TEMPERATURE

Specifies the circuit temperature for an HSPICE/HSPICE RF simulation.

### Syntax

```
.TEMP t1 [t2t3 ...]
```

---

Argument	Description
t1 t2	Temperatures in xC at when HSPICE/HSPICE RF simulates the circuit.

---

### Description

Use this command to specify the circuit temperature for an HSPICE simulation. You can use either the `.TEMP` command or the `TEMP` parameter in the `.DC`, `.AC`, and `.TRAN` commands. HSPICE compares the circuit simulation temperature against the reference temperature in the `TNOM` option. HSPICE uses the difference between the circuit simulation temperature and the `TNOM` reference temperature to define derating factors for component values.

HSPICE RF supports only one `.TEMP` command in a netlist. If you use multiple `.TEMP` commands, only the last one will be used.

**Note:** HSPICE allows multiple `.TEMP` commands in a netlist and performs multiple DC, AC or TRAN analyses for each temperature. Do not set the temperature to the same value multiple times.

When you use multiple temperature values in a `.TEMP` command, HSPICE RF perform multiple HB, SN, PHASENOISE, etc. analyses for each temperature. The simulation results for the different temperature values saved using a file naming convention consistent with `.ALTER` commands.

### Examples

In Example 1, the `.TEMP` command sets the circuit temperatures for the entire circuit simulation. To simulate the circuit by using individual elements or model temperatures, HSPICE/HSPICE RF uses:

- Temperature as set in the `.TEMP` command.
- `.OPTION TNOM` setting (or the `TREF` model parameter).
- `DTEMP` element temperature.

*Example 1*

```
.TEMP -55.0 25.0 125.0
```

In Example 2:

- The .TEMP command sets the circuit simulation temperature to 100° C.
- You do not specify .OPTION TNOM so it defaults to 25° C.
- The temperature of the diode is 30° C above the circuit temperature as set in the DTEMP parameter.

That is:

- $D1temp = 100^{\circ} C + 30^{\circ} C = 130^{\circ}$  .
- HSPICE/HSPICE RF simulates the D2 diode at 100° C.
- R1 simulates at 70° C.

Because the diode model command specifies TREF at 60° C, HSPICE/HSPICE RF derates the specified model parameters by:

- 70° C (130° C - 60° C) for the D1 diode.
- 40° C (100° C - 60° C) for the D2 diode.
- 45° C (70° C - TNOM) for the R1 resistor.

*Example 2*

```
.TEMP 100
D1 N1 N2 DMOD DTEMP=30
D2 NA NC DMOD
R1 NP NN 100 TC1=1 DTEMP=-30
.MODEL DMOD D IS=1E-15 VJ=0.6 CJA=1.2E-13
+ CJP=1.3E-14 TREF=60.0
```

In Example 3, parameterized .TEMP is also supported.

*Example 3*

```
.param mytemp =0
.temp '105 + 3*mytemp'
```

**See Also**

[.AC](#)  
[.DC](#)

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

.TEMP (or) .TEMPERATURE

.OPTION TNOM  
.TRAN



---

## .TF

Calculates DC small-signal values for transfer functions.

### Syntax

```
.TF ov srcnam
```

Argument	Description
ov	Small-signal output variable.
srcnam	Small-signal input source.

### Description

Use this command to calculate DC small-signal values for transfer functions (ratio of output variable to input source). You do not need to specify `.OP`.

The `.TF` command defines small-signal output and input for DC small-signal analysis. When you use this command, HSPICE computes:

- DC small-signal value of the transfer function (output/input)
- Input resistance
- Output resistance

### Examples

```
.TF V(5,3) VIN
.TF I(VLOAD) VIN
```

For the first example, HSPICE computes the ratio of V(5,3) to VIN. This is the ratio of small-signal input resistance at VIN to the small-signal output resistance (measured across nodes 5 and 3). If you specify more than one `.TF` command in a single simulation, HSPICE runs only the last `.TF` command.

### See Also

[.DC](#)

---

## .TITLE

Sets the simulation title.

### Syntax

`.TITLE string_of_up_to_73_characters`

Or, if `.TITLE` is not used

`string_of_up_to_80_characters`

---

Argument	Description
----------	-------------

---

string	Any character string up to 73 (or 80 if <code>.TITLE</code> is omitted) characters long.
--------	--

---

### Description

Use this command to set the simulation title in the first line of the input file. This line is read and used as the title of the simulation, regardless of the line's contents. The simulation prints the title verbatim in each section heading of the output listing file.

To set the title you can place a `.TITLE` command on the first line of the netlist. However, the `.TITLE` syntax is not required.

In the second form of the syntax, the string is the first line of the input file. The first line of the input file is always the implicit title. If any command appears as the first line in a file, simulation interprets it as a title and does not execute it.

An `.ALTER` command does not support using the `.TITLE` command. To change a title for a `.ALTER` command, place the title content in the `.ALTER` command itself.

### Examples

```
.TITLE my-design_netlist
```

### See Also

[.ALTER](#)

---

## .TRAN

Starts a transient analysis that simulates a circuit at a specific time. In HSPICE RF you can run a parameter sweep around a single analysis, but the parameter sweep cannot change an `.OPTION` value. In addition, HSPICE RF does not support the `.TRAN DATA` command and only supports the data-driven syntax for parameter sweeps (for example, `.TRAN AB sweepdata=name`).

### Syntax

Syntax for Single-Point Analysis:

```
.TRAN tstep1 tstop1 [START=val] [UIC]
```

Syntax for Double-Point Analysis:

```
.TRAN tstep1 tstop1 [tstep2 tstop2]
+ [START=val] [UIC] [SWEEP var type np pstart pstop]
.TRAN tstep1 tstop1 [tstep2 tstop2]
+ [START=val] [UIC] [SWEEP var START="param_expr1"
+ STOP="param_expr2" STEP="param_expr3"]
.TRAN tstep1 tstop1 [tstep2 tstop2] [START=val] [UIC]
+ [SWEEP var start_expr stop_expr step_expr]
```

Syntax for Multipoint Analysis:

```
.TRAN tstep1 tstop1 [tstep2 tstop2 ...tstepN tstopN]
+ [START=val] [UIC] [SWEEP var type np pstart pstop]
.TRAN tstep1 tstop1 [tstep2 tstop2 ...tstepN tstopN]
+ [START=val] [UIC] [SWEEP var START="param_expr1"
+ STOP="param_expr2" STEP="param_expr3"]
.TRAN tstep1 tstop1 [tstep2 tstop2 ...tstepN tstopN]
+ [START=val] [UIC]
+ [SWEEP var start_expr stop_expr step_expr]
```

Syntax for Data-Driven Sweep:

```
.TRAN DATA=datanm
.TRAN tstep1 tstop1 [tstep2 tstop2 ...tstepN tstopN]
+ [START=val] [UIC] [SWEEP DATA=datanm]
.TRAN DATA=datanm [SWEEP var type np pstart pstop]
.TRAN DATA=datanm [SWEEP var START="param_expr1"
+ STOP="param_expr2" STEP="param_expr3"]
.TRAN DATA=datanm
+ [SWEEP var start_expr stop_expr step_expr]
```

Syntax for Monte Carlo Analysis:

```
.TRAN tstep1 tstop1 [tstep2 tstop2 ...tstepN tstopN]
```

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .TRAN

```
+ [START=val] [UIC] [SWEEP MONTE=MCcommand]
```

#### Syntax for Optimization:

```
.TRAN DATA=datanm OPTIMIZE=opt_par_fun  
+ RESULTS=measnames MODEL=optmod  
.TRAN [DATA=filename] SWEEP OPTIMIZE=OPTxxx  
+ RESULTS=ierr1 ... ierrn MODEL=optmod
```

---

Argument	Description
DATA= <i>datanm</i>	Data name, referenced in the .TRAN command.
MONTE= MCcommand	Where MCcommand can be any of the following: <ul style="list-style-type: none"><li>▪ <i>val</i> Specifies the number of random samples to produce.</li><li>▪ <i>val</i> <i>firstrun=num</i> Specifies the sample number on which the simulation starts.</li><li>▪ <i>list num</i> Specifies the sample number to execute.</li><li>▪ <i>list(num1:num2 num3 num4:num5)</i> Samples from <i>num1</i> to <i>num2</i>, sample <i>num3</i>, and samples from <i>num4</i> to <i>num5</i> are executed (parentheses are optional).</li></ul>
np	Number of points or number of points per decade or octave, depending on what keyword precedes it.
param_expr...	Expressions you specify: <i>param_expr1...param_exprN</i> .
pincr	Voltage, current, element, or model parameter; or any temperature increment value. If you set the <i>type</i> variation, use <i>np</i> (number of points), not <i>pincr</i> .
pstart	Starting voltage, current, or temperature; or any element or model parameter value. If you set the <i>type</i> variation to POI (list of points), use a list of parameter values, instead of <i>pstart pstop</i> .
pstop	Final value: voltage, current, temperature; element or model param.
START	Time when printing or plotting begins. Caution: If you use .TRAN with a .MEASURE command, a non-zero START time can cause incorrect .MEASURE results. Do not use non-zero START times in .TRAN commands when you also use .MEASURE.
SWEEP	Indicates that .TRAN specifies a second sweep.

Argument	Description
tstep1...	Printing or plotting increment for printer output and the suggested computing increment for post-processing. This argument is always a positive value.
tstop1...	Time when a transient analysis stops incrementing by the first specified time increment ( <i>tstep1</i> ). If another tstep-tstop pair follows, analysis continues with a new increment. This argument is always a positive value.
UIC	<p>If you specify the UIC parameter in the .TRAN command, HSPICE does not calculate the initial DC operating point, but directly enters transient analysis. When you use .TRAN UIC, the .TRAN node values (at time zero) are determined by searching for the first value found in this order: from .IC value, then IC parameter on an element command, then .NODESET value, otherwise use a voltage of zero.</p> <p>Note that forcing a node value of the DC operating point might not satisfy KVL and KCL. In this event you might see activity during the initial part of the simulation. This might happen if you use UIC and do not specify some node values, when you specify too many (conflicting) .IC values are specified, or when you force node values and the topology changes. Forcing a node voltage applies a fixed equivalent voltage source during DC analysis and transient analysis removes the voltage sources to calculate the second and later time points.</p> <p>Therefore, to correct DC convergence problems use .NODESETS (without .TRAN UIC) liberally (when a good guess can be provided) and use .ICs sparingly (when the exact node voltage is known).</p>
type	<p>Any of the following keywords:</p> <ul style="list-style-type: none"> <li>▪ DEC – decade variation.</li> <li>▪ OCT – octave variation (the value of the designated variable is eight times its previous value).</li> <li>▪ LIN – linear variation.</li> <li>▪ POI – list of points.</li> </ul>
var	<p>Name of an independent voltage or current source, any element or model parameter, or the TEMP keyword (indicating a temperature sweep). You can use a source value sweep, referring to the source name (SPICE style). However, if you specify a parameter sweep, a .DATA command, or a temperature sweep you must choose a parameter name for the source value and subsequently refer to it in the .TRAN command. The parameter must not start with TEMP and should be defined in advance using the .PARAM command.</p>

Argument	Description
firstrun	MONTE= <i>val</i> value specifies the number of Monte Carlo iterations to perform. This argument specifies the desired number of iterations. HSPICE runs from num1 to num1+val-1.
list	Iterations at which HSPICE performs a Monte Carlo analysis. You can write more than one number after <i>list</i> . The colon represents “from... to...”. Specifying only one number makes HSPICE run at only the specified point.
OPTIMIZE	When used with .TRAN and SWEEP, this argument is either opt_par_fun or OPTxxx for a bisection/Monte Carlo analysis in HSPICE.

### Description

Use to start a transient analysis that simulates a circuit at a specific time.

For single-point analysis, the values of the *tstep*, *tstop1*, and *START* arguments should obey the following rules:

For double-point analysis, the values of the *tstep1*, *tstop1*, *tstep2*, *tstop2*, and *START* arguments should obey the following rules:

In double-point analysis, if  $tstep2 < tstop1$ ,  $tstop2 < tstop1$ , and *START* is not explicitly set, the command is interpreted as:

There can be three different “DELMAX” values involved in a .TRAN command:

- .OPTION DELMAX (value specified with this .OPTION)
- *delmax* (value that can be specified with the .TRAN command)
- “auto” DELMAX (value that is computed automatically)

When column 4 is interpreted as *delmax*, this command has a higher priority than the DELMAX option. The maximum internal timestep taken by HSPICE during transient analysis is referred to as  $\Delta t_{max}$ . Its value is normally computed automatically based on several timestep control settings. If you wish to override the automatically computed value, and force the maximum step size to be a specific value, you can do so with .OPTION DELMAX, or by specifying a *delmax* value with the .TRAN command. If not specified, HSPICE automatically computes a DELMAX “auto” value, based on timestep control factors such as *FS* and *RMAX*. (For a complete list of timestep control factors, see [Transient Control Options](#) in the *HSPICE User Guide: Simulation and Analysis*.)

For multipoint analysis, the values of the *tstep1*, *tstop1*, ..., *tstepN*, *tstopN*, and *START* arguments should obey the following rules:

The following limitation applies for HSPICE: The ratio between  $t_{stop1}$  and  $t_{step}$  must be  $t_{stop1} \leq 1e9 \cdot t_{step}$ . For example, `.TRAN 8n 8` is permissible, but `.TRAN 0.1n 8` is not.

You can initiate a store/restore operation that creates checkpoint files describing a running process during transient analysis; the operating system can later reconstruct the process from the contents of this file. This function is available in HSPICE only on Redhat Linux/SuSE Linux platforms for the current release.

### Examples

*Example 1* Performs and prints the transient analysis every 1 ns for 100 ns.

```
.TRAN 1NS 100NS
```

*Example 2* Performs the calculation every 0.1 ns for the first 25 ns; and then every 1 ns until 40 ns. Printing and plotting begin at 10 ns.

```
.TRAN .1NS 25NS 1NS 40NS START=10NS
```

*Example 3* Does the calculation every 0.1 ns for 25 ns; and then every 1 ns for 40 ns; and then every 2 ns until 100 ns. Printing and plotting begin at 10 ns.

```
.TRAN .1NS 25NS 1NS 40NS 2NS 100NS START = 10NS
```

*Example 4* Example 4 performs the calculation every 10 ns for 1  $\mu$ s. This example bypasses the initial DC operating point calculation. It uses the nodal voltages specified in the `.IC` command (or by `IC` parameters in element commands) to calculate the initial conditions.

```
.TRAN 10NS 1US UIC
```

*Example 5* This example increases the temperature by 10 degreesC through the range -55 degreesC to 75 degreesC. It also performs transient analysis for each temperature.

```
.TRAN 10NS 1US UIC SWEEP TEMP -55 75 10
```

*Example 6* Analyzes each load parameter value at 1 pF, 5 pF, and 10 pF.

```
.TRAN 10NS 1US SWEEP load POI 3 1pf 5pf 10pf
```

*Example 7* This example is a data-driven time sweep. It uses a data file as the sweep input. If the parameters in the data command are controlling sources, then a piecewise linear specification must reference them.

```
.TRAN data=dataname
```

## Chapter 2: HSPICE and HSPICE RF Netlist Commands

### .TRAN

*Example 8* Performs the calculation every 10ns for 1us from the 11th to 20th Monte Carlo trials.

```
.TRAN 10NS 1US SWEEP MONTE=10 firstrun=11
```

*Example 9* Performs the calculation every 10ns for 1us at the 10th trial, then from the 20th to the 30th trial, followed by the 35th to the 40th trial and finally at the 50th Monte Carlo trial.

```
.TRAN 10NS 1US SWEEP MONTE=list(10 20:30 35:40 50)
```

### See Also

[.IC](#)

[.NODESET](#)

[.STORE](#)

[.OPTION DELMAX](#)

[Timing Analysis Using Bisection](#)

[Transient Analysis](#)

[Signal Integrity Examples](#) for netlists using .TRAN including *iotran.sp*, *qa8.sp*, and *qabounce.sp*. See also *ipopt.sp* for an optimization example using .TRAN.

[Behavioral Application Examples](#) for the path to the demo file *invb\_op.sp* demonstrating use of .TRAN with OPTIMIZE to optimize a CMOS macromodel inverter.



---

## .TRANNOISE

RF analysis activates transient noise analysis to compute the additional noise variables over a standard .TRAN analysis. **Important:** FMAX has a dramatic effect on TRANNOISE, since it controls the amount of energy each noise source is allowed to emit. Therefore, huge values of FMAX (like 100G) can result in huge instantaneous noise levels. FMIN sets the low frequency limit for flicker noise, and therefore controls the energy in flicker noise sources. You can expect some significant differences with FMAX and FMIN changes: Noise power will increase linearly with FMAX; Flicker noise power can scale as 1/FMIN.

### Syntax

#### *Input Syntax—Monte Carlo Single Sample Approach*

```
.TRANNOISE output [METHOD=MC] [SEED=val]
+ [AUTOCORRELATION=0|1|off|on]
+ [FMIN=val] [FMAX=val] [SCALE=val]
```

#### *Input Syntax—Monte Carlo Multi-Sample Approach*

```
.TRANNOISE output
+ [METHOD=MC] [SEED=val] [SAMPLES=val]
+ [AUTOCORRELATION=0|1|off|on]
+ [FMIN=val] [FMAX=val] [SCALE=val]
```

#### *Input Syntax—SDE Approach*

```
.TRANNOISE output METHOD=SDE
+ [AUTOCORRELATION=0|1|off|on]
+ [TIME=(all|val)]
+ [FMIN=val] [FMAX=val] [SCALE=val]
```

---

Argument	Description
<i>output</i>	(Required) Output node, pair of nodes, or 2-terminal element. Noise calculations are referenced to this node (or node pair). Specify a pair of nodes as V(n+,n-). If you specify only one node, V(n+), then HSPICE RF reads the second node as ground. If you specify a 2-terminal element, the noise voltage across this element is treated as the output.

Argument	Description
METHOD= MC   SDE	Specifies Monte Carlo or SDE transient noise analysis method. The default, or, if METHOD is not specified, is the single-sample Monte Carlo method. Specifying METHOD=SDE is required to select the transient noise analysis SDE method.  METHOD=MC   SDE is position independent.
AUTOCORRELATION	(Optional) Used to enable the autocorrelation function calculation at the specified output (for jitter measurements, for example). <ul style="list-style-type: none"> <li>▪ AUTOCORRELATION=0 (OFF) - (default) Autocorrelation function is not calculated.</li> <li>▪ AUTOCORRELATION=1 (ON) - Autocorrelation function is calculated at the specified output.</li> </ul>
TIME	(Optional) Used to specify additional time points (breakpoints) where time-domain noise should be evaluated in addition to those time points that will be evaluated as part of the normal time-stepping algorithm. Use this parameter to force noise evaluation at important time points of interest (such as rising/falling edges). TIME=all: (default) causes time-domain noise ONOISE values to be computed and available for output at all time points selected by the .TRAN command time-step algorithm. TIME=val: Specifies a single additional time point at which time domain noise is measured. The value can be numeric or a parameter. A .TRAN analysis at this time point will be forced. Note that time-domain noise calculations require an accompanying .TRAN analysis at each time point. The TIME parameter may therefore add transient analysis time-points (breakpoints) as needed while values given outside the range of the .TRAN command constraints are ignored.
FMIN	(Optional) Base frequency used for modeling frequency dependent noise sources. Sets bandwidth for contributing noise sources. (Default: 1/ TSTOP) See Note below.
FMAX	(Optional) Maximum frequency used for modeling frequency dependent noise sources. Sets bandwidth for contributing noise sources. Default: 1/ TSTEP; See Note below.
SCALE	Scale factor that can be applied to uniformly amplify the intensity of all device noise sources to exaggerate their contributions. Default: 1.0

### Description

Use to analyze time-variant noise in HSPICE RF for circuits driven with non-periodic waveforms. The transient noise analysis requires an accompanying

.TRAN analysis which determines the time-sampling, matrix solutions, and deterministic output waveforms. The .TRANNOISE command is used to activate transient noise and to compute the additional noise variables. Note that this is consistent with how .NOISE computes additional noise outputs when added to an .AC analysis.

The Monte Carlo approach can capture very nonlinear noise behaviors. This is useful, for example, when the responses of circuits with noise are known to have non-Gaussian variations about their noise-less simulations.

For an in-depth discussion, see [Transient Noise Analysis](#) in the *HSPICE User Guide: RF Analysis*.

### Examples

*Example 1* Generates 30 Monte Carlo noise simulations beginning with a noiseless (index=1) simulation.

```
.TRANNOISE v(out) SWEEP MONTE=30
```

*Example 2* Generates 20 Monte Carlo noise simulations starting with the seed value (i.e., index) of 31 for the first simulation.

```
.TRANNOISE v(out) SWEEP MONTE=20 FIRSTRUN=31
```

*Example 3* Generates a single noise simulation, with seed value of 50, with all noise sources amplified by a factor of 10.

```
.TRANNOISE v(out) SWEEP MONTE=1 FIRSTRUN=50 SCALE=10.0
```

*Example 4* Activates SDE noise analysis, and dumps the ONOISE output to the \*.tr0 file:

```
.TRANNOISE v(out)
.PROBE TRANNOISE ONOISE
```

*Example 5* Activates SDE noise analysis, placing a lower bound on flicker noise to be 10kHz, and an upper bound on all noise power at 100MHz:

```
.TRANNOISE v(out) FMIN=10k FMAX=100MEG
```

### See Also

- [.TRAN](#)
- [.NOISE](#)
- [.PTDNOISE](#)
- [.OPTION MCBRIEF](#)
- [.OPTION MACMOD](#)
- [.OPTION MODMONTE](#)
- [.OPTION MONTECON](#)

**Chapter 2: HSPICE and HSPICE RF Netlist Commands**  
**.TRANNOISE**

**.OPTION RANDGEN**  
**.OPTION SEED**

---

## .UNPROTECT or .UNPROT

Restores normal output functions previously restricted by a `.PROTECT` command as part of the encryption process in HSPICE.

### Syntax

```
.UNPROTECT
```

### Description

Use this command to restore normal output functions previously restricted by a `.PROTECT` command.

- Any elements and models located between `.PROTECT` and `.UNPROTECT` commands, inhibit the element and model listing from the `LIST` option.
- Neither the `.OPTION NODE` cross-reference, nor the `.OP` operating point printout list any nodes within the `.PROTECT` and `.UNPROTECT` commands.
- The `.UNPROTECT` command is encrypted during the encryption process.

**Caution:** If you use `.prot/.unprot` in a library or file that is not encrypted warnings are issued that the file is encrypted and the file or library is treated as a “black box.”

**Note:** To perform a complete bias check and print all results in the Outputs Biaschk Report, do not use `.protect/.unprotect` in the netlist for the part that is used in `.biaschk`. For example: If a model definition such as `model nch` is contained within `.prot/.unprot` commands, in the `*.lis` you'll see a warning message as follows: `**warning** : model nch defined in .biaschk cannot be found in netlist--ignored`

Usage Note: The `.prot/.unprot` feature is meant for the encryption process and *not* netlist echo suppression. Netlist and model echo suppression is on by default since HSPICE C-2009.03. For a compact and better formatted output (`*.lis`) file, use `.OPTION LIS_NEW`

### See Also

[.PROTECT or .PROT](#)  
[.OPTION BRIEF](#)  
[.BIASCHK](#)  
[.OPTION LIS\\_NEW](#)

---

## .VARIATION

Specifies global and local variations on model parameters in HSPICE.

### Syntax

```
.Variation
  Define options
  Define common parameters that apply to all subblocks
  .Global_Variation
    Define the univariate independent random variables
    Define additional random variables through
    transformation
    Define variations of model parameters
  .End_Global_Variation
  .Local_Variation
    Define the univariate independent random variables
    Define additional random variables through
    transformation
    Define variations of model parameters
    .Element_Variation
      Define variations of element parameters
    .End_Element_Variation
  .End_Local_Variation
  .Spatial_Variation
    Define the univariate independent random variables
    Define additional random variables through
    transformation
    Define variations of model parameters
  .End_Spatial_Variation
.End_Variation
```

### Description

Use this command to specify global, local, and spatial variations on model parameters, resulting from variations in materials and manufacturing. If a Variation Block is read as part of .ALTER processing, then the contents are treated as additive. If the same parameters are redefined, HSPICE considers this an error.

The following are parameters and options available to the Variation Block:

- **Constant parameter**—definition which can be referenced anywhere within the Variation Block:

```
parameter PARAM=val
```

- **Univariate Independent Random Variable** normal, uniform, and cumulative distributions below, respectively:

```
parameter IVarName=N()
parameter IVarName=U()
parameter IVarName=CDF(xn, yn)
```

- **Transformed Random Variable**

```
parameter TVarName=expression(IVarName IVarName)
```

- **Variation Definition for Model Parameter**

```
modelType modelName paramName=Expression_For_Sigma
```

- **Variation Definition for Element Parameter**

```
modelType paramName=Expression_For_Sigma
modelType(condition) paramName=Expression_For_Sigma
```

- **Expression\_For\_Sigma**

---

#### Referencing a previously defined Random Variable

---

perturb('expression(IVarName TVarName IVarName TVarName)')	absolute
perturb('expression(IVarName TVarName IVarName TVarName)') %	relative

---



---

#### Referencing a previously defined Random Variable

---

perturb('expression(IVarName TVarName IVarName TVarName)')	absolute
perturb('expression(IVarName TVarName IVarName TVarName)') %	relative

---

- **Access Function**

---

#### For element parameter (for example w, l, x, y):

---

```
get_E(elementParameter)
```

---



---

#### For netlist parameter (for example .param vdd, temper):

---

```
get_P(Parameter)
```

---

- **Options:** For a detailed description of the Variation Block and usage examples, see [Analyzing Variability and Using the Variation Block](#) in the *HSPICE User Guide: Simulation and Analysis* and for Variation Block options, see [Control Options and Syntax](#).

## .VEC

Calls a digital vector file from an HSPICE/HSPICE RF netlist.

### Syntax

```
.VEC `digital_vector_file'
```

### Description

Use this command to call a digital vector file from an HSPICE netlist. A digital vector file consists of three sections:

- Vector Pattern Definition
- Waveform Characteristics
- Tabular Data

The `.VEC` file must be a text file. If you transfer the file between UNIX/Linux and Windows, use text mode. See [Chapter 4, Digital Vector File Commands](#) for more information.

### Examples

This is a fragment from a netlist with a call to a digital vector file.

```
*file: mos2bit_v.sp - adder - 2 bit all-nand-gate binary adder
*uses digital vector input
.options post nomod
.option opts fast
*
.tran .5ns 60ns
*
.vec 'digstim.vec'
...
```



## HSPICE and RF Netlist Simulation Control Options

---

*Describes the HSPICE and HSPICE RF simulation control options you can set using various forms of the .OPTION command.*

You can set HSPICE and HSPICE RF simulation control options using the .OPTION command. This chapter provides a list of the options grouped by usage, followed by detailed descriptions of the individual options in an alphabetical list. Note that in many cases an option is only usable in either the HSPICE or HSPICE RF mode of operation. In a few instances, an option has different functionality, depending on which mode (HSPICE or HSPICE RF) has been invoked. The description of the command notes the differences.

The control options described in this chapter fall into the following broad categories:

- [General Control Options](#)
- [Input/Output Controls](#)
- [Model Analysis](#)
- [HSPICE Analysis Options](#)
- [HSPICE RF Analysis Options](#)
- [Transient and AC Small Signal Analysis Options](#)
- [Transient Control \(Integration\) Method Options](#)
- [.VARIATION Block Control Options](#)
- [.DESIGN\\_EXPLORATION Block Control Options](#)

### Notes on Default Values

The typical behavior for options is:

- Option not specified: value is default value, typically “OFF” or 0.
- Option specified but without value: typically turns the option “ON” or to a value of 1.

If an option has more than two values allowed, specifying it without a value sets it to 1, if appropriate. In most cases, options without values are allowed only for flags that can be on or off, and specifying the option without a value turns it on. There are a few options (such as POST), where there are more than two values allowed, but you can still specify it without a value. Usually, you should expect it to be 1.

### Use of Example Syntax

To copy and paste proven syntax use the demonstration files shipped with your installation of HSPICE (see [Listing of Demonstration Input Files](#)). Attempting to copy and paste from the book or help documentation may present unexpected results, as text used in formatting may include hidden characters, white space, etc. for visual clarity.

---

## HSPICE Control Options Grouped By Function

---

### General Control Options

#### Netlist Parser Control

---

<code>.OPTION ALTCC</code>	<code>.OPTION DIAGNOSTIC</code> (or) <code>.OPTION DIAGNO</code>	<code>.OPTION NOTOP</code>	<code>.OPTION SEARCH</code>
<code>.OPTION ALTCHK</code>	<code>.OPTION NOELCK</code>	<code>.OPTION NOWARN</code>	<code>.OPTION WARNLIMIT</code> (or) <code>.OPTION WARNLIM</code>
<code>.OPTION BADCHR</code>	<code>.OPTION NOMOD</code>	<code>.OPTION PARHIER</code> (or) <code>.OPTION PARHIE</code>	

---

## Output Listing Control

.OPTION BRIEF    .OPTION LENNAM    .OPTION NOISEMINFREQ    .OPTION PATHNUM  
.OPTION CAPTAB    .OPTION LIST        .OPTION NUMDGT        .OPTION STATFL  
.OPTION CO        .OPTION MCBRIEF    .OPTION OPTLST        .OPTION VFLOOR  
.OPTION INGOLD    .OPTION NODE        .OPTION OPTS

## .MEAS Options

.OPTION MEASFAIL    .OPTION MEASFILE    .OPTION MEASOUT    .OPTION PUTMEAS  
.OPTION EM\_RECOVERY

## .BIASCHK Options

.OPTION BIASFILE    .OPTION BIASNODE    .OPTION BIASPARALLEL    .OPTION BIAWARN  
.OPTION BIASINTERVAL

## Multithreading Option

.OPTION MTTHRESH

---

## Input/Output Controls

### I/O Control Options

.OPTION D\_IBIS        .OPTION MONTECON    .OPTION POST        .OPTION POSTLVL  
.OPTION INTERP        .OPTION OPFILE        .OPTION POSTLVL    .OPTION POSTTOP  
.OPTION ITRPRT        .OPTION PROBE

## Interface Control Options

.OPTION ARTIST    .OPTION CSDF    .OPTION DLENCSDf    .OPTION PSF

---

## Model Analysis

.OPTION APPENDALL    .OPTION DEFAS    .OPTION DEFPS    .OPTION NCWARN  
.OPTION ASPEC    .OPTION DEFL    .OPTION DEFSA    .OPTION SOIQ0  
.OPTION BSIM4PDS    .OPTION DEFNRD    .OPTION DEFSB    .OPTION WL  
.OPTION DCAP    .OPTION DEFNRS    .OPTION DEFSD    .OPTION WNFLAG  
.OPTION DEFAD    .OPTION DEFPD    .OPTION DEFW

## Custom Models

.OPTION CMIFLAG    .OPTION CMIUSRFLAG    .OPTION CUSTCMI

## Model Control

.OPTION HIER\_SCALE    .OPTION MACMOD    .OPTION MODMONTE    .OPTION SEED

## Scaling

.OPTION SCALE    .OPTION SCALM

## Temperature

.OPTION TNOM    .OPTION XDTEMP

## Resistance

.OPTION RESMIN

## Verilog-A

`.OPTION SPMODEL`      `.OPTION VAMODEL`

## Diode and BJT

`.OPTION DCAP`      `.OPTION EXPLI`

## Inductor and Mutual Inductors

`.OPTION GENK`      `.OPTION KLIM`

## Back Annotation Post-Layout Options

<code>.OPTION BA_ACTIVE</code>	<code>.OPTION BA_DPFPIX</code>	<code>.OPTION BA_GEOSHRINK</code>	<code>.OPTION BA_NETFMT</code>
<code>.OPTION BA_ACTIVEHIER</code>	<code>.OPTION BA_ERROR</code>	<code>.OPTION BA_HIERDELIM</code>	<code>.OPTION BA_PRINT</code>
<code>.OPTION BA_ADDPARAM</code>	<code>.OPTION BA_FILE</code>	<code>.OPTION BA_IDEALPIX</code>	<code>.OPTION BA_SCALE</code>
<code>.OPTION BA_COUPLING</code>	<code>.OPTION BA_FINGERDELIM</code>	<code>.OPTION BA_MERGEPORT</code>	<code>.OPTION BA_TERMINAL</code>

## RC Reduction

`.OPTION LA_FREQ`      `.OPTION LA_MINC`      `.OPTION LA_TOL`      `.OPTION SIM_LA`  
`.OPTION LA_MAXR`      `.OPTION LA_TIME`

---

## HSPICE Analysis Options

---

### Transient and AC Small Signal Analysis Options

#### Transient Control (Integration) Method Options

.OPTION LVLTIM .OPTION MAXORD .OPTION METHOD .OPTION MU .OPTION PURETP

#### Transient Control Limit Options

.OPTION AUTOSTOP (or) .OPTION AUTOST .OPTION GMIN .OPTION ITL4 .OPTION RMAX

#### Error Tolerance

.OPTION ABSH .OPTION CHGTOL .OPTION RELH .OPTION RELVDC  
.OPTION ABSV .OPTION EM\_RECOVERY .OPTION RELQ .OPTION TRTOL  
.OPTION ABSVAR .OPTION KCLTEST .OPTION RELV .OPTION VNTOL  
.OPTION ABSVDC .OPTION MAXAMP

#### Speed and Accuracy

.OPTION BDFATOL .OPTION DVDT .OPTION IMAX .OPTION RISETIME  
(or) .OPTION RISETI  
.OPTION BDFRTOL .OPTION DVTR .OPTION IMIN .OPTION RMIN  
.OPTION CSHUNT .OPTION FAST .OPTION ITL3 .OPTION RUNLVL  
.OPTION CVTOL .OPTION FS .OPTION ITL4 .OPTION SLOPETOL  
.OPTION DELMAX .OPTION FT .OPTION ITL5 .OPTION TIMERES  
.OPTION DI .OPTION GMIN .OPTION NEWTOL .OPTION TRCON

## Bypass

`.OPTION BYPASS`    `.OPTION BYTOL`    `.OPTION MBYPASS`

## AC/Noise

`.OPTION NOISEMINFREQ`

## Spectral Analysis Controls

`.OPTION FFT_ACCURATE`    `.OPTION FFTOUT`

## Transmission Lines

`.OPTION RISETIME` (or) `.OPTION RISETI`    `.OPTION WACC`

---

## HSPICE RF Analysis Options

---

### HB Options

<code>.OPTION HBACKRYLOVDIM</code>	<code>.OPTION HBKRYLOVDIM</code>	<code>.OPTION HBTOL</code>
<code>.OPTION HBACKRYLOVITER (or) HBAC_KRYLOV_ITER</code>	<code>.OPTION HBKRYLOVMAXITER (or) HB_KRYLOV_MAXITER</code>	<code>.OPTION HBTRANFREQSEARCH</code>
<code>.OPTION HBACTOL</code>	<code>.OPTION HBKRYLOVTOL</code>	<code>.OPTION HBTRANINIT</code>
<code>.OPTION HBCONTINUE</code>	<code>.OPTION HBLINESEARCHFAC</code>	<code>.OPTION HBTRANPTS</code>
<code>.OPTION HBFREQABSTOL</code>	<code>.OPTION HBMAXITER (or) HB_MAXITER</code>	<code>.OPTION HBTRANSTEP</code>
<code>.OPTION HBFREQRELTOL</code>	<code>.OPTION HBOSCMAXITER (or) HBOSC_MAXITER</code>	<code>.OPTION LOADHB</code>
<code>.OPTION HB_GIBBS</code>	<code>.OPTION HBPROBETOL</code>	<code>.OPTION SAVEHB</code>
<code>.OPTION HBJREUSE</code>	<code>.OPTION HBSOLVER</code>	<code>.OPTION TRANFORHB</code>
<code>.OPTION HBJREUSETOL</code>		

---

### Phase Noise Analysis

<code>.OPTION BPNMATCHTOL</code>	<code>.OPTION PHASENOISEKRYLOVITER (or) PHASENOISE_KRYLOV_ITER</code>	<code>.OPTION PHNOISELORENTZ</code>
<code>.OPTION PHASENOISEKRYLOVDIM</code>	<code>.OPTION PHASENOISETOL</code>	<code>.OPTION PHNOISEAMPM</code>



## Power Analysis

.OPTION SIM\_POWER\_ANALYSIS      .OPTION SIM\_POWERDC\_HSPICE      .OPTION SIM\_POWERSTOP  
.OPTION SIM\_POWER\_TOP      .OPTION SIM\_POWERPOST  
.OPTION SIM\_POWERDC\_ACCURACY      .OPTION SIM\_POWERSTART

---

## RC Network Reduction

.OPTION SIM\_LA      .OPTION SIM\_LA\_MAXR      .OPTION SIM\_LA\_TOL  
.OPTION SIM\_LA\_FREQ      .OPTION SIM\_LA\_MINC      .OPTION SIM\_LA\_TIME

---

## Simulation Output

.OPTION SIM\_POSTAT      .OPTION SIM\_POSTSCOPE      .OPTION SIM\_POSTTOP  
.OPTION SIM\_POSTDOWN      .OPTION SIM\_POSTSKIP

---

## Shooting Newton Options

.OPTION LOADSNINIT      .OPTION SAVESNINIT      .OPTION SNACCURACY      .OPTION SNMAXITER (or)  
SN\_MAXITER

---

## DSPF Options

.OPTION SIM_DELTAI	.OPTION SIM_DSPF_INSError	.OPTION SIM_DSPF_SCALEC
.OPTION SIM_DELTAV	.OPTION SIM_DSPF_LUMPCAPS	.OPTION SIM_DSPF_SCALER
.OPTION SIM_DSPF	.OPTION SIM_DSPF_MAX_ITER	.OPTION SIM_DSPF_VTOL
.OPTION SIM_DSPF_ACTIVE	.OPTION SIM_DSPF_RAIL	

---

## SPEF Options

.OPTION SIM_SPEF	.OPTION SIM_SPEF_MAX_ITER	.OPTION SIM_SPEF_SCALER
.OPTION SIM_SPEF_ACTIVE	.OPTION SIM_SPEF_PARVALUE	.OPTION SIM_SPEF_VTOL
.OPTION SIM_SPEF_INSError	.OPTION SIM_SPEF_RAIL	
.OPTION SIM_SPEF_LUMPCAPS	.OPTION SIM_SPEF_SCALEC	

---

## Transient Accuracy Options

.OPTION FFT_ACCURATE	.OPTION SIM_ORDER	.OPTION SIM_TRAP
.OPTION SIM_ACCURACY	.OPTION SIM_TG_THETA	.OPTION SIM_OSC_DETECT_TOL

## Alphabetical Listing of HSPICE Control Options

The following is the alphabetical list of links to the HSPICE/RF control option set. For commands see [HSPICE and HSPICE RF Netlist Commands](#).

- [.OPTION ABSH](#)
- [.OPTION ABSI](#)
- [.OPTION ABSIN](#)
- [.OPTION ABSMOS](#)
- [.OPTION ABSTOL](#)
- [.OPTION ABSV](#)
- [.OPTION ABSVAR](#)
- [.OPTION ABSVDC](#)
- [.OPTION ACCURATE](#)
- [.OPTION ACOUT](#)
- [.OPTION ALTCC](#)
- [.OPTION ALTCHK](#)
- [.OPTION APPENDALL](#)
- [.OPTION ARTIST](#)
- [.OPTION ASPEC](#)
- [.OPTION AUTOSTOP](#) (or) [.OPTION AUTOST](#)
- [.OPTION BA\\_ACTIVE](#)
- [.OPTION BA\\_ACTIVEHIER](#)
- [.OPTION BA\\_ADDPARAM](#)
- [.OPTION BA\\_COUPLING](#)
- [.OPTION BA\\_DPFPFX](#)
- [.OPTION BA\\_ERROR](#)
- [.OPTION BA\\_FILE](#)
- [.OPTION BA\\_FINGERDELIM](#)
- [.OPTION BA\\_GEOSHRINK](#)

### Chapter 3: HSPICE and RF Netlist Simulation Control Options

Transient Accuracy Options

- .OPTION BA\_HIERDELIM
- .OPTION BA\_IDEALPFX
- .OPTION BA\_MERGEPORT
- .OPTION BA\_NETFMT
- .OPTION BA\_PRINT
- .OPTION BA\_SCALE
- .OPTION BA\_TERMINAL
- .OPTION BADCHR
- .OPTION BDFATOL
- .OPTION BDFRTOL
- .OPTION BEEP
- .OPTION BIASFILE
- .OPTION BIASINTERVAL
- .OPTION BIASNODE
- .OPTION BIASPARALLEL
- .OPTION BIAWARN
- .OPTION BINPRNT
- .OPTION BPNMATCHTOL
- .OPTION BRIEF
- .OPTION BSIM4PDS
- .OPTION BYPASS
- .OPTION BYTOL
- .OPTION CAPTAB
- .OPTION CFLFLAG
- .OPTION CHGTOL
- .OPTION CMIMCFLAG
- .OPTION CMIFLAG
- .OPTION CMIPATH
- .OPTION CMIUSRFLAG

- .OPTION CONVERGE
- .OPTION CPTIME
- .OPTION CSCAL
- .OPTION CSDF
- .OPTION CSHDC
- .OPTION CSHUNT
- .OPTION CUSTCMI
- .OPTION CVTOL
- .OPTION D\_IBIS
- .OPTION DCAP
- .OPTION DCCAP
- .OPTION DCFOR
- .OPTION DCHOLD
- .OPTION DCIC
- .OPTION DCON
- .OPTION DCSTEP
- .OPTION DCTRAN
- .OPTION DEFAD
- .OPTION DEFAS
- .OPTION DEFL
- .OPTION DEFNRD
- .OPTION DEFNRS
- .OPTION DEFPPD
- .OPTION DEFPS
- .OPTION DEFSA
- .OPTION DEFSB
- .OPTION DEFSD
- .OPTION DEFW
- .OPTION DEGF

### Chapter 3: HSPICE and RF Netlist Simulation Control Options

#### Transient Accuracy Options

- .OPTION DEGFN
- .OPTION DEGFP
- .OPTION DELMAX
- .OPTION DI
- .OPTION DIAGNOSTIC (or) .OPTION DIAGNO
- .OPTION DLENCSDF
- .OPTION DV
- .OPTION DVDT
- .OPTION DVTR
- .OPTION DYNACC
- .OPTION EM\_RECOVERY
- .OPTION EPSMIN
- .OPTION EXPLI
- .OPTION EXPMAX
- .OPTION FAST
- .OPTION FFT\_ACCURATE
- .OPTION FFTOUT
- .OPTION FMAX
- .OPTION FS
- .OPTION FSCAL
- .OPTION FT
- .OPTION GDCPATH
- .OPTION GENK
- .OPTION GEOSHRINK
- .OPTION GMAX
- .OPTION GMIN
- .OPTION GMINDC
- .OPTION GRAMP
- .OPTION GSCAL

- .OPTION GSHDC
- .OPTION GSHUNT
- .OPTION HBACKRYLOVDIM
- .OPTION HBACKRYLOVITER (or) HBAC\_KRYLOV\_ITER
- .OPTION HBACTOL
- .OPTION HBCONTINUE
- .OPTION HBFREQABSTOL
- .OPTION HBFREQRELTOL
- .OPTION HB\_GIBBS
- .OPTION HBJREUSE
- .OPTION HBJREUSETOL
- .OPTION HBKRYLOVDIM
- .OPTION HBKRYLOVTOL
- .OPTION HBKRYLOVMAXITER (or) HB\_KRYLOV\_MAXITER
- .OPTION HBLINESEARCHFAC
- .OPTION HBMAXITER (or) HB\_MAXITER
- .OPTION HBOSCMAXITER (or) HBOSC\_MAXITER
- .OPTION HBPROBETOL
- .OPTION HBSOLVER
- .OPTION HBTOL
- .OPTION HBTRANFREQSEARCH
- .OPTION HBTRANINIT
- .OPTION HBTRANPTS
- .OPTION HBTRANSTEP
- .OPTION HIER\_DELIM
- .OPTION HIER\_SCALE
- .OPTION IC\_ACCURATE
- .OPTION ICSWEEP
- .OPTION IMAX

### Chapter 3: HSPICE and RF Netlist Simulation Control Options

#### Transient Accuracy Options

- .OPTION IMIN
- .OPTION INGOLD
- .OPTION INTERP
- .OPTION IPROP
- .OPTION ITL1
- .OPTION ITL2
- .OPTION ITL3
- .OPTION ITL4
- .OPTION ITL5
- .OPTION ITLPTRAN
- .OPTION ITLPZ
- .OPTION ITRPRT
- .OPTION IVTH
- .OPTION KCLTEST
- .OPTION KLIM
- .OPTION LA\_FREQ
- .OPTION LA\_MAXR
- .OPTION LA\_MINC
- .OPTION LA\_TIME
- .OPTION LA\_TOL
- .OPTION LENNAM
- .OPTION LIMPTS
- .OPTION LIMTIM
- .OPTION LISLVL
- .OPTION LIS\_NEW
- .OPTION LIST
- .OPTION LOADHB
- .OPTION LOADSNINIT
- .OPTION LSCAL



- .OPTION LVLTIM
- .OPTION MACMOD
- .OPTION MAXAMP
- .OPTION MAXORD
- .OPTION MAXWARNS
- .OPTION MBYPASS
- .OPTION MCBRIEF
- .OPTION MEASDGT
- .OPTION MEASFAIL
- .OPTION MEASFILE
- .OPTION MEASFORM
- .OPTION MEASOUT
- .OPTION MESSAGE\_LIMIT
- .OPTION METHOD
- .OPTION MODMONTE
- .OPTION MODPRT
- .OPTION MONTECON
- .OPTION MOSRALIFE
- .OPTION MOSRASORT
- .OPTION MRAAPI
- .OPTION MRAEXT
- .OPTION MRAPAGED
- .OPTION MRA00PATH, MRA01PATH, MRA02PATH, MRA03PATH
- .OPTION MTTHRESH
- .OPTION MU
- .OPTION NCFILTER
- .OPTION NCWARN
- .OPTION NEWTOL
- .OPTION NODE

### Chapter 3: HSPICE and RF Netlist Simulation Control Options

#### Transient Accuracy Options

- .OPTION NOELCK
- .OPTION NOISEMINFREQ
- .OPTION NOMOD
- .OPTION NOPIV
- .OPTION NOTOP
- .OPTION NOWARN
- .OPTION NUMDGT
- .OPTION NUMERICAL\_DERIVATIVES
- .OPTION NXX
- .OPTION OFF
- .OPTION OPFILE
- .OPTION OPTCON
- .OPTION OPTLST
- .OPTION OPTPARHIER
- .OPTION OPTS
- .OPTION PARHIER (or) .OPTION PARHIE
- .OPTION PATHNUM
- .OPTION PCB\_SCALE\_FORMAT
- .OPTION PHASENOISEKRYLOVDIM
- .OPTION PHASENOISEKRYLOVITER (or) PHASENOISE\_KRYLOV\_ITER
- .OPTION PHASENOISETOL
- .OPTION PHASETOLI
- .OPTION PHASETOLV
- .OPTION PHD
- .OPTION PHNOISELORENTZ
- .OPTION PHNOISEAMPM
- .OPTION PIVOT
- .OPTION PIVTOL
- .OPTION POST

- .OPTION POSTLVL
- .OPTION POST\_VERSION
- .OPTION POSTTOP
- .OPTION PROBE
- .OPTION PSF
- .OPTION PURETP
- .OPTION PUTMEAS
- .OPTION PZABS
- .OPTION PZTOL
- .OPTION RADEGFILE
- .OPTION RADEGOUTPUT
- .OPTION RANDGEN
- .OPTION REDEFSUB
- .OPTION RELH
- .OPTION RELI
- .OPTION RELIN
- .OPTION RELMOS
- .OPTION RELQ
- .OPTION RELTOL
- .OPTION RELV
- .OPTION RELVAR
- .OPTION RELVDC
- .OPTION REPLICATES
- .OPTION RES\_BITS
- .OPTION RESMIN
- .OPTION RISETIME (or) .OPTION RISETI
- .OPTION RITOL
- .OPTION RMAX
- .OPTION RMIN

### Chapter 3: HSPICE and RF Netlist Simulation Control Options

#### Transient Accuracy Options

- .OPTION RUNLVL
- .OPTION SAMPLING\_METHOD
- .OPTION SAVEHB
- .OPTION SAVESNINIT
- .OPTION SCALE
- .OPTION SCALM
- .OPTION SEARCH
- .OPTION SEED
- .OPTION SHRINK
- .OPTION SIM\_ACCURACY
- .OPTION SIM\_DELTAI
- .OPTION SIM\_DELTAV
- .OPTION SIM\_DSPF
- .OPTION SIM\_DSPF\_ACTIVE
- .OPTION SIM\_DSPF\_INSERTOR
- .OPTION SIM\_DSPF\_LUMPCAPS
- .OPTION SIM\_DSPF\_MAX\_ITER
- .OPTION SIM\_DSPF\_RAIL
- .OPTION SIM\_DSPF\_SCALEC
- .OPTION SIM\_DSPF\_SCALER
- .OPTION SIM\_DSPF\_VTOL
- .OPTION SIM\_LA
- .OPTION SIM\_LA\_FREQ
- .OPTION SIM\_LA\_MAXR
- .OPTION SIM\_LA\_MINC
- .OPTION SIM\_LA\_TIME
- .OPTION SIM\_LA\_TOL
- .OPTION SIM\_ORDER
- .OPTION SIM\_OSC\_DETECT\_TOL

- .OPTION SIM\_POSTAT
- .OPTION SIM\_POSTDOWN
- .OPTION SIM\_POSTSCOPE
- .OPTION SIM\_POSTSKIP
- .OPTION SIM\_POSTTOP
- .OPTION SIM\_POWER\_ANALYSIS
- .OPTION SIM\_POWER\_TOP
- .OPTION SIM\_POWERDC\_ACCURACY
- .OPTION SIM\_POWERDC\_HSPICE
- .OPTION SIM\_POWERPOST
- .OPTION SIM\_POWERSTART
- .OPTION SIM\_POWERSTOP
- .OPTION SIM\_SPEF
- .OPTION SIM\_SPEF\_ACTIVE
- .OPTION SIM\_SPEF\_INSERROR
- .OPTION SIM\_SPEF\_LUMPCAPS
- .OPTION SIM\_SPEF\_MAX\_ITER
- .OPTION SIM\_SPEF\_PARVALUE
- .OPTION SIM\_SPEF\_RAIL
- .OPTION SIM\_SPEF\_SCALEC
- .OPTION SIM\_SPEF\_SCALER
- .OPTION SIM\_SPEF\_VTOL
- .OPTION SIM\_TG\_THETA
- .OPTION SIM\_TRAP
- .OPTION SI\_SCALE\_SYMBOLS
- .OPTION SLOPETOL
- .OPTION SNACCURACY
- .OPTION SNCONTINUE
- .OPTION SNMAXITER (or) SN\_MAXITER

### Chapter 3: HSPICE and RF Netlist Simulation Control Options

#### Transient Accuracy Options

- .OPTION SOIQ0
- .OPTION SPLIT\_DP
- .OPTION SPMODEL
- .OPTION STATFL
- .OPTION STRICT\_CHECK
- .OPTION SX\_FACTOR
- .OPTION SYMB
- .OPTION TIMERES
- .OPTION TMIFLAG
- .OPTION TMIPATH
- .OPTION TMIVERSION
- .OPTION TNOM
- .OPTION TRANFORHB
- .OPTION TRCON
- .OPTION TRTOL
- .OPTION UNWRAP
- .OPTION VAMODEL
- .OPTION VERIFY
- .OPTION VFLOOR
- .OPTION VNTOL
- .OPTION WACC
- .OPTION WARN
- .OPTION WARN\_SEP
- .OPTION WARNLIMIT (or) .OPTION WARNLIM
- .OPTION WAVE\_POP
- .OPTION WDELAYOPT
- .OPTION WDF
- .OPTION WINCLUDEGDIMAG
- .OPTION WL

- .OPTION WNFLAG
- .OPTION XDTEMP
- .OPTION (X0R,X0I)
- .OPTION (X1R,X1I)
- .OPTION (X2R,X2I)
- .VARIATION Block Control Options
- .DESIGN\_EXPLORATION Block Control Options

---

## **.OPTION ABSH**

Sets the absolute current change through voltage-defined branches.

### **Syntax**

```
.OPTION ABSH=x
```

**Default**    0.0

### **Description**

Use this option to set the absolute current change through voltage-defined branches (voltage sources and inductors). Use this option with options `DI` and `RELH` to check for current convergence.

### **See Also**

[.OPTION DI](#)

[.OPTION RELH](#)



## **.OPTION ABSI**

Sets the absolute error tolerance for branch currents in diodes, BJTs, and JFETs during DC and transient analysis.

### **Syntax**

```
.OPTION ABSI=x
```

**Default**     $1e-9$  when `KCLTEST=0` or  $1e-6$  when `KCLTEST=1`.

### **Description**

Use this option to set the absolute error tolerance for branch currents in diodes, BJTs, and JFETs during DC and transient analysis. Decrease `ABSI` if accuracy is more important than convergence time.

To analyze currents less than 1 nanoamp, change `ABSI` to a value at least two orders of magnitude smaller than the minimum expected current. Min value:  $1e-25$ ; Max value: 10.

### **See Also**

- [.AC](#)
- [.OPTION ABSMOS](#)
- [.OPTION KCLTEST](#)
- [.TRAN](#)

---

## .OPTION ABSIN

Convergence criteria for bisection/passfail optimization.

### Syntax

```
.OPTION ABSIN=val
```

**Default** None

### Description

This option invokes the absolute input parameter value and takes effect only for bisection methods `bisection` or `passfail`. When set as `.OPTION ABSIN`, it overrides all optimization model card accuracy settings and ignores the `relout` and `itropt` parameters; when set in the model card, `absin` takes effect only for the specified model. In cases where both `absin` and `relin` are set, `absin` takes higher priority and dominates the simulation.

### Examples

```
.OPTION ABSIN=5.0e-11
```

For use in a model card:

```
.MODEL optmod opt absin=5.0e-11
```

### See Also

[.MODEL](#)

## .OPTION ABSMOS

Specifies the current error tolerance for MOSFET devices in DC or transient analysis.

### Syntax

```
.OPTION ABSMOS=x
```

**Default** 1uA

### Description

Use this option to specify the current error tolerance for MOSFET devices in DC or transient analysis. The `ABSMOS` setting determines whether the drain-to-source current solution has converged. The drain-to-source current converged if:

- The difference between the drain-to-source current in the last iteration and the current iteration is less than `ABSMOS`, or
- This difference is greater than `ABSMOS`, but the percent change is less than `RELMOS`.

Min value: 1e-15; Max value 10.

If other accuracy tolerances also indicate convergence, HSPICE solves the circuit at that timepoint and calculates the next timepoint solution. For low-power circuits, optimization, and single transistor simulations, set `ABSMOS=1e-12`.

### See Also

[.DC](#)  
[.OPTION RELMOS](#)  
[.TRAN](#)

---

## **.OPTION ABSTOL**

Sets the absolute error tolerance for branch currents in DC and transient analysis.

### **Syntax**

```
.OPTION ABSTOL=x
```

**Default**    1e-9

### **Description**

Use this option to set the absolute error tolerance for branch currents in DC and transient analysis. Decrease `ABSTOL` if accuracy is more important than convergence time. `ABSTOL` is the same as `ABSI`. Min value: 1e-25; Max value: 10.

### **See Also**

- [.DC](#)
- [.OPTION ABSI](#)
- [.OPTION ABSMOS](#)
- [.TRAN](#)

## **.OPTION ABSV**

Sets the absolute minimum voltage for DC and transient analysis.

### **Syntax**

```
.OPTION ABSV=x
```

**Default** 50 uV

### **Description**

Use this option to set the absolute minimum voltage for DC and transient analysis. `ABSV` is the same as `VNTOL`.

- If accuracy is more critical than convergence, decrease `ABSV`.
- If you need voltages less than 50 uV, reduce `ABSV` to two orders of magnitude less than the smallest desired voltage. This ensures at least two significant digits.

Typically, you do not need to change `ABSV`, except to simulate a high-voltage circuit. A reasonable value for 1000-volt circuits is 5 to 50 uV. Default value: 5e-05; Min value: 0; Max value: 10.

### **See Also**

[.DC](#)  
[.OPTION VNTOL](#)  
[.TRAN](#)

## **.OPTION ABSVAR**

Sets the absolute limit for maximum voltage change between time points.

### **Syntax**

```
.OPTION ABSVAR=volts
```

**Default** 0.5 (volts)

### **Description**

Use this option to set the absolute limit for the maximum voltage change from one time point to the next. Use this option with `.OPTION DVDT`. If the simulator produces a convergent solution that is greater than `ABSVAR`, HSPICE discards the solution, sets the timestep to a smaller value and recalculates the solution. This is called a timestep reversal.

For additional information, see “[DVDT Dynamic Timestep](#)” in the *HSPICE User Guide: Simulation and Analysis*.

### **See Also**

[.OPTION DVDT](#)

## **.OPTION ABSVDC**

Sets the minimum voltage for DC and transient analysis.

### **Syntax**

```
.OPTION ABSVDC=volts
```

**Default** 50uV.

### **Description**

Use this option to set the minimum voltage for DC and transient analysis. If accuracy is more critical than convergence, decrease `ABSVDC`. If you need voltages less than 50 uV, reduce `ABSVDC` to two orders of magnitude less than the smallest voltage. This ensures at least two digits of significance. Typically, you do not need to change `ABSVDC` unless you simulate a high-voltage circuit. For 1000-volt circuits, a reasonable value is 5 to 50 uV.

### **See Also**

- [.DC](#)
- [.OPTION VNTOL](#)
- [.TRAN](#)

---

## .OPTION ACCURATE

Selects a time algorithm for circuits such as high-gain comparators.

### Syntax

```
.OPTION ACCURATE= [0 | 1]
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### Description

Use this option to select a time algorithm that uses `LVLTIM=3` and `DVDT=2` for circuits such as high-gain comparators. Use this option with circuits that combine high gain and large dynamic range to guarantee accurate solutions in HSPICE. When set to 1, this option sets these control options:

The default does not set the above control options.

In HSPICE RF, this option turns on `.OPTION FFT_ACCURATE` and is subordinate to `.OPTION SIM_ACCURACY`.

To see how use of the ACCURATE option impacts the value settings when used with `.METHOD=GEAR`, and other options, see [Appendix B, How Options Affect other Options](#).

### See Also

- [.OPTION ABSVAR](#)
- [.OPTION DVDT](#)
- [.OPTION FFT\\_ACCURATE](#)
- [.OPTION FT](#)
- [.OPTION LVLTIM](#)
- [.OPTION METHOD](#)
- [.OPTION RELMOS](#)
- [.OPTION RELVAR](#)
- [.OPTION SIM\\_ACCURACY](#)



## .OPTION ACOUT

Specifies the method for calculating differences in AC output values.

### Syntax

.OPTION ACOUT=0 | 1

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### Description

Use this option to specify a method for calculating the differences in AC output values for magnitude, phase, and decibels for print output and plots.

- ACOUT=0: selects the SPICE method which calculates the *magnitude* of the differences real and imaginary in HSPICE. (Rarely used, but available for backward compatability.)
- ACOUT=1: (rarely used by analog and HSPICE RF designers) selects the HSPICE method which calculates the *difference* of the magnitudes of the values real and imaginary.

### Examples

#### ACOUT=0

```
VR(N1,N2) = REAL [V(N1,0) - V(N2,0)]
VI(N1,N2) = IMAG [V(N1,0) - V(N2,0)]
Magnitude
VM(N1,N2) = [VR(N1,N2)^2+VI(N1,N2)^2]0.5 Phase
VP(N1,N2) = ARCTAN[VI(N1,N2)/VR(N1,N2)]
Decibel
VDB(N1,N2) = 20 * LOG10 [VM(N1,N2)]
```

#### ACOUT=1

```
VR(N1,N2) = REAL [V(N1,0)] - REAL [V(N2,0)]
VI(N1,N2) = IMAG [V(N1,0)] - IMAG [V(N2,0)] Magnitude
VM(N1,0) = [VR(N1,0)^2 + VI(N1,0)^2]0.5
VM(N2,0) = [VR(N2,0)^2 + VI(N2,0)^2]0.5
VM(N1,N2) = VM(N1,0) - VM(N2,0)
Phase
VP(N1,0) = ARCTAN[VI(N1,0)/VR(N1,0)]
VP(N2,0) = ARCTAN[VI(N2,0)/VR(N2,0)] VP(N1,N2) = VP(N1,0) -
VP(N2,0)
Decibel
VDB(N1,N2) = 20 * LOG10 (VM(N1,0)/VM(N2,0))
```

---

## .OPTION ALTCC

Sets onetime reading of the input netlist for multiple .ALTER commands.

### Syntax

```
.OPTION ALTCC= [-1 | 0 | 1]
```

**Default** 0

### Description

Use this option to enable HSPICE to read the input netlist only once for multiple .ALTER commands.

- ALTCC=1 reads input netlist only once for multiple .ALTER commands.
- ALTCC=0 or -1 disables this option. HSPICE does not output a warning message during transient analysis. Results are output following analysis.

.OPTION ALTCC or .OPTION ALTCC=1 ignores parsing of an input netlist before an .ALTER command during standard cell library characterization only when an .ALTER command changes parameters, source stimulus, analysis, or passive elements. Otherwise, this option is ignored.

### See Also

[.ALTER](#)

[.LIB](#)

## **.OPTION ALTCHK**

Disables (or re-enables) topology checking in redefined elements (in altered netlists).

### **Syntax**

```
.OPTION ALTCHK=0 | 1
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### **Description**

By default, HSPICE automatically reports topology errors in the latest elements in your top-level netlist. It does not report errors in elements that you redefine by using the `.ALTER` command (altered netlist).

To enable topology checking redefined elements in the `.ALTER` block, set:  
`.OPTION ALTCHK=1` or `.OPTION ALTCHK`

To disable topology checking in redefined elements (that is, to check topology only in the top-level netlist, not in the altered netlist), set:

```
.OPTION ALTCHK=0
```

### **See Also**

[.ALTER](#)

---

## .OPTION APPENDALL

Allows the top hierarchical level to use the `.APPENDMODEL` command even if the MOSFET model is embedded in a subcircuit.

### Syntax

```
.OPTION APPENDALL
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### Description

Use this option when, for example, MOSFET model cards from fabs might be embedded in subcircuit definitions. The option ends the need to edit fab model files to include `.APPENDMODEL` commands in subcircuit definitions. When this option is declared above the `.APPENDMODEL` command, then the main (uppermost) circuit level hierarchy can be used, even if the MOSFET model is embedded in a subcircuit. With this option, if the `.APPENDMODEL` command appears both in the main circuit and in a subcircuit, the `.APPENDMODEL` in the subcircuit takes priority. Without this option, the rules of `.APPENDMODEL` remain unchanged.

### Examples

In this example, the `.APPENDMODEL` in the main circuit is used.

```
.option appendall
.appendmodel n_ra mosra nch nmos
.SUBCKT mosra_test 1 2 3 4
M1 1 2 3 4 nch L=PL W=PW
.model nch nmos level= ...
.ENDS
```

In this example, the `.APPENDMODEL` in the subcircuit is used.

```
.option appendall
.appendmodel n_ra mosra nch nmos
.SUBCKT mosra_test 1 2 3 4
M1 1 2 3 4 nch L=PL W=PW
.model nch nmos level= ...
.appendmodel n_ra1 mosra nch nmos
.ENDS
```

### See Also

[.APPENDMODEL](#)

.MODEL  
.MOSRA

---

## .OPTION ARTIST

Enables the Cadence™ Virtuoso® Analog Design Environment interface.

### Syntax

```
.OPTION ARTIST=[0|1|2]
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 2

### Description

Enables the Virtuoso® Analog Design Environment if `ARTIST=2`. This option requires a specific license. For HSPICE RF, this option allows you to include HSPICE RF analyses such as Harmonic Balance, Shooting Newton, and their associated small-signal analyses and use their native waveform viewer.

This option is generally used together with `.OPTION PSF`. If you use `.OPTION PSF=1` or `2` with `ARTIST=1` or `2` then the output format is always binary (Parameter Storage Format) and you need to use the Cadence ADE converter utility to change the binary format to ASCII format.

**Note:** The PSF format is only supported on Sun/SPARC, Red Hat/SUSE Linux and IBM AIX platforms as well as the 64-bit versions.

The syntax is:

```
ADE_install_dir/platform/tools/dfII/bin/psf -i input_file  
-o output_file
```

### See Also

[.OPTION PSF](#)

## **.OPTION ASPEC**

Sets HSPICE or HSPICE RF to ASPEC-compatibility mode.

### **Syntax**

```
.OPTION ASPEC=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to set the application to ASPEC-compatibility mode. When you set this option to 1, the simulator reads ASPEC models and netlists, and the results are compatible.

If you set *ASPEC*, the following model parameters default to *ASPEC* values:

- ACM=1: Changes the default values for CJ, IS, NSUB, TOX, U0, and UTRA.
- Diode Model: TLEV=1 affects temperature compensation for PB.
- MOSFET Model: TLEV=1 affects PB, PHB, VTO, and PHI.
- SCALM, SCALE: Sets the model scale factor to microns for length dimensions.
- WL: Reverses implicit order for stating width and length in a MOSFET command. The default (WL=0) assigns the length first, then the width.

### **See Also**

[.OPTION SCALE](#)  
[.OPTION SCALM](#)  
[.OPTION WL](#)

---

## .OPTION AUTOSTOP (or) .OPTION AUTOST

Stops a transient analysis in HSPICE or HSPICE RF after calculating all TRIG-TARG, FIND-WHEN, and FROM-TO measure functions.

### Syntax

```
.OPTION AUTOSTOP
-or-
.OPTION AUTOSTOP='expression'
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### Description

Use this option to terminate a transient analysis in HSPICE after calculating all TRIG-TARG, FIND-WHEN, and FROM-TO measure functions. This option can substantially reduce CPU time. You can use the AUTOSTOP option with any measure type. You can also use the result of the preceding measurement as the next measured parameter.

When using .OPTION AUTOSTOP='expression', the 'expression' can only involve measure results, a logical AND (&&) or a logical OR(||). Using these types of expressions ends the simulation if any one of a set of .MEASURE commands succeeds, even if the others are not completed.

Also terminates the simulation after completing all .MEASURE commands. This is of special interest when testing corners.

### Examples

```
.option autostop='m1&&2||m4'
.meas tran m1 trig v(bd_a0) val='ddv/2' fall=1 targ v(re_bd)
+ val='ddv/2' rise=1
.meas tran m2 trig v(bd_a0) val='ddv/2' fall=2 targ v(re_bd)
+ val='ddv/2' rise=2
.meas tran m3 trig v(bd_a0) val='ddv/2' rise=2 targ v(re_bd)
+ val='ddv/2' rise=3
.meas tran m4 trig v(bd_a0) val='ddv/2' fall=3 targ v(re_bd)
+ val='ddv/2' rise=4
.meas tran m5 trig v(bd_a0) val='ddv/2' rise=3 targ v(re_bd)
+ val='ddv/2' rise=5
```

In this example, when either m1 and m2 are obtained or just m4 is obtained, the transient analysis ends.

### See Also

[.MEASURE \(Rise, Fall, Delay, and Power Measurements\)](#)



.MEASURE (FIND and WHEN)  
.MEASURE (Continuous Results)  
.MEASURE (AVG, EM\_AVG, INTEG, MIN, MAX, PP, and RMS)  
.MEASURE (Integral Function)  
.MEASURE (Derivative Function)  
.MEASURE (Error Function)  
.MEASURE PHASENOISE

---

## .OPTION BA\_ACTIVE

Specifies the active net file name(s) selective net back-annotation.

### Syntax

```
.OPTION BA_ACTIVE = "FILENAME [;FILENAME2; FILENAME3...]"
```

### Description

Conducts selective back-annotation. The active net file name contains the selected nets in the format defined by Star-RC or Star-RCXT. If no file is supplied, all nets (nodes) are selected for annotation. Multiple active net files can be specified, with each other being delimited by semicolon.

You must use this option with BA\_FILE, or it has no effect. To view examples of active net files used in a format for Star-RC/Star-RCXT, see [Selective Net Back-Annotation](#) in the *HSPICE User Guide: Simulation and Analysis*.

### Examples

```
.option ba_active = "./hspice/NETLIST/DSPF/active.rcxt"
```

### See Also

[.OPTION BA\\_ACTIVEHIER](#)

[.OPTION BA\\_FILE](#)

[Back-Annotation Demo Cases](#)

## **.OPTION BA\_ACTIVEHIER**

Annotate full hierarchical net names that are specified for BA\_ACTIVE files.

### **Syntax**

```
.OPTION BA_ACTIVEHIER = 0|1
```

**Default** 0

### **Description**

Setting this option to 1 annotates the full hierarchical net names that are specified in BA\_ACTIVE files, instead of the name starting from last period (.). For example, in an active net file, if the net name is *xi1.xi2.net\_name*, by default, HSPICE truncates this name from the last period and identifies the net name as '*net\_name*'. If you set *ba\_activehier=1*, HSPICE use the full net name.

### **See Also**

- [.OPTION BA\\_ACTIVE](#)
- [Post-Layout Back-Annotation](#)
- [Back-Annotation Demo Cases](#)

---

## .OPTION BA\_ADDPARAM

Specifies extra parameters to be scaled by .OPTIONS BA\_SCALE/  
BA\_GEOSHRINK.

### Syntax

```
.OPTION BA_ADDPARAM = "LINEAR: PARAM [PARAM2 ...] ;  
+ QUAD: PARAM [PARAM2 ...] "
```

---

Argument	Description
LINEAR/QUAD	Keywords to indicate how the following parameters to be scaled for instances in the DPF file, i.e., to be scaled linearly/quadratically.
PARAM	Parameter to be scaled by .OPTIONS BA_SCALE/ BA_GEOSHRINK. Multiple parameters can be specified, with each other being delimited by blank space. The parameter groups (LINEAR/QUAD) are delimited by semicolon.

---

### Description

.OPTION BA\_SCALE/BA\_GEOSHRINK is usually applied only to common elements (M/D/R/C/J) and common parameters needed for scaling (L/W/AD/AS/PD/PS/AREA ...). At times, extra, unusual parameters need to be scaled by BA\_SCALE/BA\_GEOSHRINK as well, such as the variation of common parameters from a subckt wrapping a type of element. For example, see a subckt wrapping a MOSFET with parameters *w<sub>r</sub>*/*l<sub>r</sub>*, which stands for width/length of the wrapped MOSFET in the following example.

### Examples

```
.OPTION BA_ADDPARAM = "LINEAR: WR LR; QUAD: ASR AREAR"
```

### See Also

[.OPTION BA\\_SCALE](#)  
[.OPTION BA\\_GEOSHRINK](#)

## **.OPTION BA\_COUPLING**

Controls how to treat cutoff coupling capacitors when invoking selective net back-annotation.

### **Syntax**

```
.OPTION BA_COUPLING = 0|1|2
```

**Default** 0

### **Description**

Coupling capacitors across two nets are very common in parasitic netlists. For example, assume one coupling capacitor CC with terminals connected to two nodes belonging to nets A and B, respectively. When selective net back-annotation is launched and net A is active while net B is inactive, then CC is cut off from the node under net B and the terminal becomes a dangling node.

.OPTION BA\_COUPLING allows three methods to deal with the cutoff coupling capacitor, with BA\_COUPLING assigned a value listed below:

- 0: Just discards this coupling capacitor (a warning is issued).
- 1: Let the cutoff terminal connect to the node defined by \*|GROUND\_NET.
- 2: Let the cutoff terminal connect to the unexpanded inactive node (node B in the example above).

### **Examples**

```
.OPTION BA_COUPLING = 2
```

### **See Also**

[Post-Layout Back-Annotation](#)  
[Back-Annotation Demo Cases](#)

---

## **.OPTION BA\_DPFPIX**

Prepends an extra prefix when searching the ideal netlist for instances referenced by the parasitic file (DSPF/SPEF/DPF).

### **Syntax**

```
.OPTION BA_DPFPIX= "string"
```

**Default** m\_ prefix string

### **Description**

BA\_DPFPIX causes HSPICE to prepend an extra prefix when searching the ideal netlist for instances referenced by the parasitic file (DSPF/SPEF/DPF). This option serves a different purpose from .OPTION BA\_DPFPIX in that BA\_DPFPIX is used to indicate the prefix that has been prepended to instances in the parasitic file while BA\_IDEALPIX is used to specify the extra prefix to be prepended. This option aids in precisely specifying backward-annotation. HSPICE automatically checks for a reverse hierarchy in device name. But if the name is prepended with a prefix string, you need to specify that prefix string using .OPTION BA\_DPFPIX so HSPICE can correctly extract the hierarchical name.

### **See Also**

[.OPTION BA\\_IDEALPIX](#)

## **.OPTION BA\_ERROR**

Mode for handling error on nets.

### **Syntax**

```
.OPTION BA_Error=0|1|2
```

**Default** 1 (LUMPCAP)

### **Description**

Specifies means to handle an error on nets, where:

- 0: EXIT — Terminates the simulation with an error message
- 1: LUMPCAP — Adds only the total lumped net capacitance
- 2: YES — Expands whatever can be expanded

### **Examples**

```
.OPTION BA_ERROR = 2
```

### **See Also**

[Post-Layout Back-Annotation](#)

---

## .OPTION BA\_FILE

Launches DPF parasitic back-annotation.

### Syntax

```
.OPTION BA_FILE = "FILENAME [;FILENAME2; FILENAME3 ...]"
```

### Description

This option enables you to specify the DPF file and invoke DPF back-annotation. This option expands usage so that a DSP file does not have to be embedded in a DSPF file as the "Instance Section."

FILENAME is the name of the file that contains parasitic information in SPEF or DSPF format. Multiple parasitic netlists can be specified, with each other being delimited by semicolon. These parasitic netlists must be independent but cannot cross-reference each other. The advantage of DPF back-annotation is that the pre-layout hierarchy is maintained for simulation.

For MOSFET devices, the supported DPF parameters are: L, W, AD, AS, PD, PS, NRD, NRS, SA, SB, SD, NF, DELVTO, MULU0, RGEOMOD, RDC, RSC, SCA, SCB, SCC, SA1, SA2, SA3, SA4, SA5, SA6, SA7, SA8, SA9, SA10, SB1, SB2, SB3, SB4, SB5, SB6, SB7, SB8, SB9, SB10, SW1, SW2, SW3, SW4, SW5, SW6, SW7, SW8, SW9, SW10.

Use .OPTION BA\_ACTIVE with .OPTION BA\_FILE to launch selective parasitic expansion. To view examples of the SPEF and DSPF file structures, see [DSPF and SPEF File Structures](#) in the *HSPICE User Guide: Simulation and Analysis*.

### Examples

#### Example 1 Single Parasitic Netlist

```
.OPTION BA_FILE = "./hspice/NETLIST/DSPF/add4.spf"
```

#### Example 2 Multiple Parasitic Netlists

```
.OPTION BA_FILE = "./ba_file1.spf; ba_file2.spf; ba_file3.spef"
```

### See Also

[Full Back-Annotation](#)

[Back-Annotation Demo Cases](#)



## **.OPTION BA\_FINGERDELIM**

Explicitly specifies the delimiter character used for finger devices.

### **Syntax**

```
.OPTION BA_FINGERDELIM=character
```

**Default** @

### **Description**

Use this option to specify delimiter characters used on fingered devices.

---

## **.OPTION BA\_GEOSHRINK**

Element scaling factor used with .OPTION BA\_SCALE.

### **Syntax**

```
.OPTION BA_GEOSHRINK=X
```

**Default** Same as .OPTION GEOSHRINK (or its default value)

### **Description**

In addition to .OPTION BA\_SCALE, use this option to further scale geometric parameters of element instances in the DPF file separately, whose default units are meters. By default the instances in the DPF file are scaled by .OPTION GEOSHRINK (and SCALE), no difference with instances in the ideal netlist.

When .OPTION BA\_GEOSHRINK is specified, the .OPTION GEOSHRINK is then disabled for instances in the DPF file and BA\_GEOSHRINK is applied to them separately.

The final instance geometric parameters are then calculated as:

```
final_dimension = original_dimension * BA_SCALE *  
BA_GEOSHRINK
```

The effective scaling factor is the product of the two parameters; HSPICE uses `ba_scale*ba_geoshrink` to scale the parameters/dimensions in the DPF file.

### **See Also**

- [.OPTION BA\\_SCALE](#)
- [.OPTION SCALE](#)
- [.OPTION GEOSHRINK](#)

## **.OPTION BA\_HIERDELIM**

Specifies the hierarchical separator in the DPF file.

### **Syntax**

```
.OPTION BA_HIERDELIM=character
```

### **Description**

If the hierarchical separator used in a DPF file is different from BA\_HIERDELIM, the hierarchical separator must be specified with BA\_HIERDELIM.

### **Examples**

```
.OPTION BA_HIERDELIM=/'
```

---

## **.OPTION BA\_IDEALPFX**

Instructs the simulator to add a second prepended prefix when doing a search of the ideal netlist.

### **Syntax**

```
.OPTION BA_IDEALPFX = "m_"
```

### **Description**

BA\_IDEALPFX instructs HSPICE to prepend an extra prefix when searching the ideal netlist for instances referenced by the parasitic file (DSPF/SPEF/DPF). Note that a different purpose is served here than when using .OPTION BA\_DPFPFX in that BA\_DPFPFX is used to indicate the prefix that has been prepended to instances in the parasitic file, while BA\_IDEALPFX is used to specify the extra prefix to be prepended.

### **See Also**

[.OPTION BA\\_DPFPFX](#)

## **.OPTION BA\_MERGEPORT**

Controls whether to merge net ports into one node.

### **Syntax**

```
.OPTION BA_MERGEPORT = 0 | 1
```

**Default** 1

### **Description**

Merging net ports into one node may introduce some small inaccuracy. To separate the net ports, set `BA_MERGEPORT = 0`.

### **Examples**

```
.OPTION BA_MERGEPORT = 0
```

### **See Also**

[Post-Layout Back-Annotation](#)  
[Back-Annotation Demo Cases](#)

---

## **.OPTION BA\_NETFMT**

Specifies the format of Active Net File.

### **Syntax**

`.OPTION BA_NETFMT= [1 | 2]`

**Default** 1

---

<b>Argument</b>	<b>Description</b>
1	Reports active nets in StarRC ( <i>*.rcxt</i> ) format for the Selective Net Extraction Flow.
2	Reports active nets in HSPICE Back-Annotation ( <i>*.hsimba</i> ) format for the Selective Net Back-Annotation Flow.

---

### **Description**

Enables HSPICE to output active nodes in STAR-RCXT format or HSPICE format.

## **.OPTION BA\_PRINT**

Controls whether to output nodes and resistors/capacitors introduced by back-annotation.

### **Syntax**

```
.OPTION BA_PRINT = IDEAL|ALL
```

**Default** IDEAL

### **Description**

Specify this option to control the output of nodes and resistors/capacitors added by back-annotation.

After back-annotation many nodes and resistors/capacitors are introduced in the output files, which can distract from the effective and useful information. By setting BA\_PRINT=IDEAL, the newly-added nodes and resistors/capacitors by back-annotation are filtered from the \*.lis, \*.ic# and \*.tr#. To switch on the output of these nodes and RCs, set BA\_PRINT=ALL.

### **Examples**

```
.OPTION BA_PRINT=IDEAL
```

### **See Also**

[Post-Layout Back-Annotation](#)  
[Back-Annotation Demo Cases](#)

---

## **.OPTION BA\_SCALE**

Sets the element scaling factor for instances in the DPF file separately.

### **Syntax**

```
.OPTION BA_SCALE=X
```

**Default** Same as .OPTION SCALE (or its default value)

### **Description**

Use this option to scale geometric parameters of element instances in the DPF file separately, whose default unit is meters. By default the instances in the DPF file are scaled by .OPTION SCALE, no difference with instances in the ideal netlist. When .OPTION BA\_SCALE is specified, the .OPTION SCALE is then disabled for instances in the DPF file and BA\_SCALE is applied to them separately.

You can also use this option with .OPTION BA\_GEOSHRINK to scale an element even more finely. The effective scaling factor is the product of the two parameters; HSPICE uses  $ba\_scale * ba\_geoshrink$  to scale the parameters/dimensions in the DPF file.

### **See Also**

[.OPTION BA\\_GEOSHRINK](#)

[.OPTION SCALE](#)

[.OPTION GEOSHRINK](#)



## .OPTION BA\_TERMINAL

Specifies mapping characters for back annotation terminal name.

### Syntax

```
.OPTION BA_TERMINAL = "TERMINAL= ALIAS [ ; TERMINAL2= ALIAS2 ;  
+ TERMINAL3=ALIAS3 ...] "
```

Argument	Description
TERMINAL	Terminal name used in the parasitic netlist.
ALIAS	Common terminal name recognized by the simulator, or user-defined/ tool-specific terminal name used in the ideal netlist.

### Description

Specifies the terminal name mapping between the parasitic netlist and the terminal names recognized by the simulator. Multiple TERMINAL=ALIAS pairs can be specified, with each other being delimited by semicolon. Generally, terminal names used in the parasitic netlist and ideal netlist are same. These terminals are widely accepted by various simulators, as listed in the following table.

Table 1 Default rules for element terminal names

Term. Index	M (MOS)	Q (BJT)	R,C,D (Resistor, Capacitor, Diode)
1	D [R] [A] [I] [N]	C [O] [L] [L] [E] [C] [T] [O] [R]	A [N] [O] [D] [E], P [L] [U] [S], P [O] [S] [I] [T] [I] [V] [E]
2	G [A] [T] [E]	B [A] [S] [E]	B, C [A] [T] [H] [O] [D] [E], M [I] [N] [U] [S], N [E] [G] [A] [T] [I] [V] [E]
3	S [O] [U] [R] [C] [E]	E [M] [I] [T] [T] [E] [R]	S [U] [B] [S] [T] [R] [A] [T] [E]
4	B [U] [L] [K]	S [U] [B] [S] [T] [R] [A] [T] [E]	N/A

## Chapter 3: HSPICE and RF Netlist Simulation Control Options

### .OPTION BA\_TERMINAL

HSPICE uses the first character and optional subsequent characters listed above to determine which terminal is referred to.

However, sometimes terminal names referenced in the parasitic netlist are user-defined/tool-specific and different from above default terminal characters. Another case is the terminal names employed in the parasitic netlist follow the default rules, but are different from the ones used in ideal netlist, which are user-defined/tool-specific. This is especially common for elements of subckt type. That's what BA\_TERMINAL is intended for.

.OPTION BA\_TERMINAL is used to set up the terminal name mapping between the parasitic netlist and ideal netlist. Of the TERMINAL ALIAS pair, the first entry is the terminal name used in the parasitic netlist, and the second entry is the corresponding terminal name used in the ideal netlist.

**Note:** Consider the following limitation for BA\_TERMINAL. All terminal mapping pairs specified are of global scope, not only applied for specific elements/blocks, but applicable for all un-found terminal names. Besides, if multiple mapping pairs have the same first entry (key), e.g., .OPTION BA\_TERMINAL = "UDRN=N1; UDRN=N2", then the latter pair will hide the previous one and take effect.

### Examples

*Example 1* This example maps user-defined terminals (UDRN, UGATE) in the parasitic netlist to default terminal characters (D, G).

```
.OPTION BA_TERMINAL="UDRN=D; UGATE=G"
```

*Example 2* This example maps widely accepted terminal characters (D, G, S) in the parasitic netlist to subckt-defined node list (SUBCKT\_N1, SUBCKT\_N2, SUBCKT\_N3) in the ideal netlist.

```
.OPTION BA_TERMINAL="D SUBCKT=N1; G=SUBCKT_N2; S=SUBCKT_N3"
```

### See Also

[Post-Layout Back-Annotation](#)  
[Back-Annotation Demo Cases](#)

## **.OPTION BADCHR**

Generates a warning on finding a nonprintable character in an input file.

### **Syntax**

```
.OPTION BADCHR= [0 | 1]
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to generate a warning on finding a nonprintable character in an input file by setting to 1.

---

## .OPTION BDFATOL

Sets the absolute tolerance for the global accuracy control of the Backward Differentiation Formulae integration method.

### Syntax

```
.OPTION BDFATOL=val
```

**Default**    1e-3

### Description

Use this option to set the absolute tolerance of the circuit convergence integration method BDF (a higher order integration algorithm than Backward-Euler, Gear, or Trapezoidal).

The option operates independent of `.OPTIONS RUNLVL` and `ACCURATE` settings with the following exception:

If either `.OPTION RUNLVL` or `ACCURATE` follows an `.OPTION BDFATOL` or `BDFRTOL` value, the `RUNLVL` or `ACCURATE` setting overrides the tolerance of the BDF algorithm. If `ACCURATE` is set with or without `RUNLVL`, the default for `BDFATOL` will always set to 1.e-5.

---

RUNLVL	BDFATOL
0	1e-4
1	1e-2
2	1e-2
3	1e-3
4	1e-4
5	1e-4
6	1e-5

---

The option appears in the `.lis` file.

### Examples

```
.OPTION METHOD=BDF
+.OPTIONS BDFATOL=1e-4 BDFRTOL=1e-4
```

**See Also**

[.OPTION METHOD](#)  
[.OPTION BDFRTOL](#)

---

## .OPTION BDFRTOL

Sets the relative tolerance for the global accuracy control of the Backward Differentiation Formulae integration method.

### Syntax

```
.OPTION BDFRTOL=val
```

**Default**    1e-3

### Description

Use this option to set the relative tolerance of the circuit convergence integration method BDF (a higher order integration algorithm than Backward-Euler, Gear, or Trapezoidal).

The option operates independent of .OPTIONS RUNLVL and ACCURATE settings with the following exception:

If .OPTION RUNLVL or ACCURATE follows an .OPTION BDFATOL or BDFRTOL value, the RUNLVL or ACCURATE setting overrides the tolerance of the BDF algorithm. If ACCURATE is set with or without RUNLVL, the default for BDFRTOL will always reset to 1.e-5.

---

RUNLVL	BDFRTOL
0	1e-4
1	1e-2
2	1e-2
3	1e-3
4	1e-4
5	1e-4
6	1e-5

---

The value of the option appears in the *.lis* file.

### Examples

```
.OPTION METHOD=BDF
+.OPTIONS BDFRTOL=1e-4 BDFATOL=1e-4
```

**See Also**

[.OPTION METHOD](#)  
[.OPTION BDFATOL](#)

## **.OPTION BEEP**

Enables or disables audible alert tone when simulation returns a message.

### **Syntax**

```
.OPTION BEEP=[0|1]
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to enable or disable the audible alert tone when simulation returns a message.

- BEEP=1 Turns on an audible tone when simulation returns a message (such as HSPICE job completed).
- BEEP=0 Turns off the audible tone.



## **.OPTION BIASFILE**

Sends .BIASCHK command results to a specified file.

### **Syntax**

```
.OPTION BIASFILE=file_name
```

**Default** \*.lis

### **Description**

Use this option to output the results of all .BIASCHK commands to a file that you specify. If you do not set this option, HSPICE outputs the .BIASCHK results to the \*.lis file.

### **Examples**

```
.OPTION BIASFILE='biaschk/mos.bias'
```

### **See Also**

[.BIASCHK](#)

## **.OPTION BIASINTERVAL**

Controls the level of information output during transient analysis.

### **Syntax**

```
.OPTION BIASINTERVAL= [0 | 1 | 2 | 3]
```

### **Description**

Use this option with the `.BIASCHKinterval` argument to control the level of information output during transient analysis.

- `BIASINTERVAL=0`: Ignores the `interval` argument.
- `BIASINTERVAL=1`: Output the total number of suppressed violation regions for those elements being monitored. Violation warning messages that are generated in these suppressed regions are removed from the output.
- `BIASINTERVAL=2`: Output detailed information regarding suppressed violation regions. This includes element information, start time, stop time, and peak values. Also, violation warning messages that are generated in these suppressed regions are removed from the output.
- `BIASINTERVAL=3`: Output detailed information about all violation regions. Also, violation warning messages that are generated in these regions are removed from the output.

### **Examples**

```
.OPTION BIASINTERVAL=1
```

### **See Also**

[.BIASCHK](#)

## **.OPTION BIASNODE**

Specifies whether to use node names or port names in element commands.

### **Syntax**

```
.OPTION BIASNODE= [0 | 1]
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to specify whether to use node names or port names in element commands in `.BIASCHK` warning messages.

- `BIASNODE=1`: use node names instead of port names
- `BIASNODE=0`: use port names (for example, ng of MOS element)

### **Examples**

```
.OPTION BIASNODE=1
```

### **See Also**

[.BIASCHK](#)

## **.OPTION BIASPARALLEL**

Controls whether `.BIASCHK` sweeps the parallel elements being monitored.

### **Syntax**

```
.OPTION BIASPARALLEL=[0 | 1]
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option with the `.BIASCHKmname` argument to control whether `.BIASCHK` sweeps the parallel elements being monitored.

- `BIASPARALLEL=1`: sweep parallel elements. If node voltage is also being monitored, only the first element is used to generate warning messages.
- `BIASPARALLEL=0`: do not sweep parallel elements.

### **Examples**

```
.OPTION BIASPARALLEL=1
```

### **See Also**

[.BIASCHK](#)

## **.OPTION BIAWARN**

Controls whether HSPICE outputs warning messages when local max bias voltage exceeds limit during transient analysis.

### **Syntax**

```
.OPTION BIAWARN= [ 0 | 1 ]
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to control whether HSPICE outputs warning messages when a local max bias voltage exceeds the limit during transient analysis.

- BIAWARN=1: Output warning messages. When transient analysis is completed, the results are output as filtered by noise.
- BIAWARN=0: Do not output a warning message. When the transient analysis is completed, output the results.

### **Examples**

```
.OPTION BIAWARN=1
```

### **See Also**

[.TRAN](#)

---

## **.OPTION BINPRNT**

Outputs the binning parameters of the CMI MOSFET model.

### **Syntax**

```
.OPTION BINPRNT= [ 0 | 1 ]
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to output the binning parameters of the CMI MOSFET model. Currently available only for Level 57.

## **.OPTION BPNMATCHTOL**

Determines the minimum required match between the NLP and PAC phase noise algorithms in HSPICE RF.

### **Syntax**

```
.OPTION BPNMATCHTOL=val
```

**Default** 0.5dB

### **Description**

Use this option to determines the minimum required match between the NLP and PAC phase noise algorithms. An acceptable range is 0.05dB to 5dB.

### **See Also**

- [.OPTION PHASENOISEKRYLOVDIM](#)
- [.OPTION PHASENOISEKRYLOVITER \(or\) PHASENOISE\\_KRYLOV\\_ITER](#)
- [.OPTION PHASENOISETOL](#)
- [.OPTION PHNOISELORENTZ](#)

---

## .OPTION BRIEF

Stops echoing (printback) of data file to stdout until HSPICE reaches an `.OPTION BRIEF=0` or `.END` command.

### Syntax

```
.OPTION BRIEF=[0|1]
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### Description

Use this option to terminate echoing (printback) of the data file to *stdout* until HSPICE finds an `.OPTION BRIEF=0` or the `.END` command. It also resets the `LIST`, `NODE` and `OPTS` options, and sets `NOMOD.BRIEF=0` enables printback. The `NXX` option is the same as `BRIEF`. `BRIEF=1` disables printback. `.OPTION BRIEF=1` and `.OPTION BRIEF=0` act similar to the commands `.PROTECT` and `.UNPROTECT`, respectively.

For information on how `BRIEF` impacts other options, see [Appendix B, How Options Affect other Options](#).

### Examples

This example shows how, if you include in your netlist `.option brief=0`, the printback of the library file would be prevented but the `list` option would work.

```
.option brief
.lib 'mymodels.lib' tt
.option brief=0
```

### See Also

- [.END](#)
- [.OPTION LIST](#)
- [.OPTION NODE](#)
- [.OPTION NXX](#)
- [.OPTION OPTS](#)
- [.PROTECT](#) or [.PROT](#)
- [.UNPROTECT](#) or [.UNPROT](#)



## **.OPTION BSIM4PDS**

Flag to control the BSIM4  $Ps_{\text{eff}}$  (effective source perimeter) and  $Pd_{\text{eff}}$  (effective drain perimeter) model equation calculation.

### **Syntax**

```
.OPTION BSIM4PDS=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Setting `BSIM4PDS=1` enhances the  $ps_{\text{eff}}$  and  $pd_{\text{eff}}$  calculation, so that when the calculated  $ps_{\text{eff}}$  and  $pd_{\text{eff}}$  is negative, HSPICE uses the `PAeffGeo` function to recalculate it. (This option solves the issue of negative  $ps_{\text{eff}}$  and  $pd_{\text{eff}}$  causing potential non-convergence issues.) When `BSIM4PDS=0`, HSPICE strictly follows the UCB code, and results in no recalculation if negative  $ps_{\text{eff}}$  or  $pd_{\text{eff}}$  occurs.

**Note:** This option is only available for BSIM4 (Level 54).

## **.OPTION BYPASS**

Bypasses model evaluations if the terminal voltages stay constant.

### **Syntax**

```
.OPTION BYPASS= [0 | 1 | 2]
```

**Default** 1 for MESFETs, JFETs, or BJTs; 2 for MOSFETs and diodes

### **Description**

Use this option to bypass model evaluations if the terminal voltages do not change. Values can be 0 (off), 1 (on), or 2 (advanced algorithm, applies to BSIM3v3, BSIM4, BSIM3SOI (LEVEL=57), BSIM4SOI (LEVEL 70), HVMOS (LEVEL 66), and PSP (LEVEL=69) MOSFETs in special cases).

To speed up simulation, `BYPASS=1` does not update the status of latent devices. `BYPASS=2` uses linear prediction to update the devices and balance speed and accuracy.

(Assuming `BYPASS` is not explicitly set otherwise): When the `BYPASS` option is not given in the netlist, its value is determined by the value of `RUNLVL` and `ACCURATE`. When `RUNLVL=0` then `BYPASS=1`; when `RUNLVL=0 + ACCURATE=1` then `BYPASS=0`; when `RUNLVL=1` through 6, then `BYPASS=2`.

### **See Also**

[.OPTION ACCURATE](#)  
[.OPTION RUNLVL](#)

## **.OPTION BYTOL**

Sets a voltage tolerance at which a MOSFET, MESFET, JFET, BJT, or diode becomes latent.

### **Syntax**

```
.OPTION BYTOL=x
```

**Default** 100.00u

### **Description**

Use this option to specify a voltage tolerance at which a MOSFET, MESFET, JFET, BJT, or diode becomes latent. HSPICE does not update status of latent devices. The default=MBYPASS x VNTOL.

### **See Also**

[.OPTION MBYPASS](#)  
[.OPTION VNTOL](#)

## **.OPTION CAPTAB**

Adds up all the capacitances attached to a node and prints a table of single-plate node capacitances.

### **Syntax**

```
.OPTION CAPTAB=[0|1]
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to print a compiled table of single-plate node capacitances for diodes, BJTs, MOSFETs, JFETs, and passive capacitors at each operating point.

**Note:** When `.OPTION CAPTAB` is used to estimate the equivalent capacitance of the circuit nodes, HSPICE can give a zero capacitance values for some nodes when a resistance is connected to that node. The reason for getting 0 is that the capacitance is a dynamic, frequency-dependent capacitance and not a static capacitance. You need to run an AC analysis to see a non-zero node capacitance.

## **.OPTION CFLFLAG**

Activates the Compiled Function Library (CFL) feature in HSPICE.

### **Syntax**

```
.OPTION CFLFLAG= [0 | 1]
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to turn on the CFL capability and pass arguments (mathematical or user-defined functions written in C that can be dynamically linked to HSPICE during run time). See the [Features](#) section of the *HSPICE User GUIDE: Simulation and Analysis* for more information.

### **Examples**

In the following example, `mysqrt(x)` and `func(arg1, arg2)` are coded as a CFL function. The functions `mysqrt` and `func` are called in the netlist as follows:

```
.option CFLflag  
.param area = 4u*u  
.param p1 = mysqrt(area)  
.param p2 = mysqrt(area/2)  
.param p3 = func(p1, p2)
```

### **See Also**

[.CFL\\_PROTOTYPE](#)  
[.PARAM \(or\) .PARAMETER \(or\) .PARAMETERS](#)

## **.OPTION CHGTOL**

Sets a charge error tolerance.

### **Syntax**

```
.OPTION CHGTOL=x
```

**Default** 1.00f

### **Description**

Use this option to set a charge error tolerance if you set `LVLTIM=2`. Use `CHGTOL` with `RELQ` to set the absolute and relative charge tolerance for all HSPICE capacitances. The default is  $1e-15$  (coulomb). Min value:  $1e-20$ ; Max value: 10.

### **See Also**

- [.OPTION CHGTOL](#)
- [.OPTION LVLTIM](#)
- [.OPTION RELQ](#)

## **.OPTION CMIMCFLAG**

Restricted: for specified users only. Enables model memory allocation for each element.

### **Syntax**

```
.OPTION CMIMCFLAG=0 | 1
```

**Default** 0

### **Description**

For restricted use: Setting this option to 1, changes the method for storing instance-specific local variation information. With use of this flag, during the instance reset process model memory is allocated for each element. Each instance will have its own model structure to store the local variation information.

**Note:** Users employing conventional public domain models where many model parameters exist should not use this option to avoid memory issues.

---

## .OPTION CMIFLAG

Loads and links the dynamically linked Common Model Interface (CMI) library.

### Syntax

```
.OPTION CMIFLAG=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### Description

Use this option to load and link the compiled CMI object `.so` file to HSPICE/HSPICE RF during simulation runs. If this option parameter is set with no value or to 1, then the CMI `.so` file is loaded as a dynamically-linked object file. If this option parameter does not exist (deemed as default) in the netlist, or is explicitly set to 0, no loading or linking takes place.

If `CMIFLAG` is set, model parameter `CMIMODEL` can be used to enable “hybrid” model usage, i.e., you can determine if a built-in model or model from custom CMI library is to be used in the simulation.

```
CMIMODEL=0 | 1 | 2 | undefined
```

Model parameter `CMIMODEL` values are as follows:

- 0: HSPICE searches for the model from built-in models. If not found, an error message is issued and HSPICE aborts.
- 1: HSPICE searches for the model from the Custom CMI. If not found, a warning message is issued and HSPICE then searches for the model from built-in models.
- 2: Invokes the CMI2 mode.
- undefined: HSPICE proceeds as if `CMIMODEL=1`.

If `.OPTION CMIFLAG` is not set, model parameter `CMIMODEL` is ignored.

### See Also

[.OPTION CUSTCMI](#)



## **.OPTION CMIPATH**

Enables automatic selection of correct Custom CMI .so library platform. For information on the HSPICE CMI, contact your Synopsys technical support team.

### **Syntax**

```
.OPTION CMIPATH= 'LIB_DIRECTORY'
```

### **Description**

This option allows you to automatically select the correct custom CMI .so library platform, even though you might not have the right information about the platform HSPICE is running on. This functionality eliminates the need to manually search for the correct platform and allows for efficient CMI .so library distribution and customer applications. The solution to this issue keeps the environment variable `hspice_lib_models` backward compatible in its usage model, but users can add the control option `.OPTION CMIPATH= 'LIB_DIRECTORY'` to the model file.

For the UNIX OS, HSPICE provides two scripts, `hspice` and `hspice64` to invoke the right HSPICE executable for the platform on which HSPICE is being invoked to run. These scripts are enhanced to recognize the correct machine and platform for automatic CMI .so library selection. For the Windows OS, no HSPICE script is required, since all Windows platforms share the same single CMI .so library: `LIB_DIRECTORYWIN` for all Windows platforms.

---

## .OPTION CMIUSRFLAG

Flag to control .OPTION SCALE parsing into the External Common Model Interface (CMI).

### Syntax

```
.OPTION CMIUSRFLAG=0 | 1 | 2 | 3
```

**Default** 0

### Description

Controls the CMI element instance parameter value (unit) scaling. This option is only available for custom CMI MOS Level 101. It permits users and/or foundry model development teams to choose desired scaling for the instance parameters of the MOSFET devices that call a foundry's CMI model libraries.

The CMIUSRFLAG values are as follows:

- 0: Turns off other functions of the CMIUSRFLAG option.
- 1: Passes  $scale * geoshrink$  value to custom CMI through artificial instance parameter "scale". If set with no value or to 1, the products of option parameters SCALE and GEOSHRINK are passed and made available to scale the CMI model instance parameter values.
- 2: Turns on dynamic model bin selection for custom CMI and turns off other functions.
- 3 HSPICE will pass options SHRINK, SCALE and M into Custom CMI (both MOSFET model with BSIM4-like topology and DIODE model), using string names "optshrink", "optscale", and "mult", respectively. In addition, final constant capacitance value (for capacitors) will be scaled by .OPTION SHRINK.

If the CMIUSRFLAG option parameter does not exist in the netlist (default), or is explicitly set to 0, then the option parameters SCALE and GEOSHRINK are not accessible in the CMI; and the element instance parameter scaling is not activated for the foundry CMI models and libraries.

### Examples

In this example, the value  $scale * geoshrink = 0.9e-6$  is parsed to the external CMI.

```
.option cmiflag=1  
.option scale=1e-6 geoshrink=0.9 cmiusrflag=1  
...  
.model nch nmos level=101 ...
```

**See Also**

[.OPTION SCALE](#)  
[.OPTION GEOSHRINK](#)  
[.OPTION SHRINK](#)

## .OPTION CONVERGE

Invokes various methods for solving nonconvergence problems.

### Syntax

```
.OPTION CONVERGE=[-1|0|1|2|3|4|5|100]
```

### Description

Use this option to run different methods for solving nonconvergence issues. This option is part of the autoconverge flow.

**Note:** In HSPICE RF, this option is ignored because it is replaced by automated algorithms.

- CONVERGE=-1: Use with DCON=-1 to disable autoconvergence.
- CONVERGE=0: Autoconvergence.
- CONVERGE=1: Use the Damped Pseudo Transient algorithm. If simulation does not converge within the set CPU time (in the CPTIME control option), then simulation halts.
- CONVERGE=2: Use a combination of DCSTEP and GMINDC ramping. Not used in the autoconvergence flow.
- CONVERGE=3: Invoke the source-stepping method. Not used in the autoconvergence flow.
- CONVERGE=4: Use the gmath ramping method.
- CONVERGE=5: Use the gshunt ramping method. Even you did not set it in an .OPTION command, the CONVERGE option activates if a matrix floating-point overflows or if HSPICE reports a “timestep too small” error. The default is 0. If a matrix floating-point overflows, then CONVERGE=1.
- CONVERGE=100 Adaptive option control for autoconvergence; this value requires less dependence on convergence option settings, such as DV, ITL1, GRAMP, SYMB, and DCON.

### See Also

[.OPTION DCON](#)  
[.OPTION GMINDC](#)  
[.OPTION DV](#)  
[.OPTION GRAMP](#)  
[.OPTION ITL1](#)  
[.OPTION SYMB](#)

## **.OPTION CPTIME**

Sets the maximum CPU time allotted for a simulation.

### **Syntax**

```
.OPTION CPTIME=x
```

**Default** 10.00x

### **Description**

Use this option to set the maximum CPU time, in seconds, allotted for this simulation job. When the time allowed for the job exceeds `CPTIME`, HSPICE prints or plots the results up to that point and concludes the job. Use this option if you are uncertain how long the simulation takes, especially when you debug new data files. The default is `1e7` (400 days).

---

## **.OPTION CSCAL**

Sets the capacitance scale for Pole/Zero analysis.

### **Syntax**

```
.OPTION CSCAL=x
```

**Default** 1.0e+12

### **Description**

Use this option to set the capacitance scale for Pole/Zero analysis. HSPICE multiplies capacitances by CSCAL.

### **See Also**

- [.OPTION FMAX](#)
- [.OPTION GSCAL](#)
- [.OPTION ITLPZ](#)
- [.OPTION LSCAL](#)
- [.OPTION PZABS](#)

## **.OPTION CSDF**

Selects the Common Simulation Data Format (Viewlogic-compatible graph data file format).

### **Syntax**

```
.OPTION CSDF=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to specify whether HSPICE/ HSPICE RF outputs CSDF data when you run a HSPICE simulation.

- If CSDF=0, CSDF output is disabled. If CSDF=1, HSPICE produces CSDF output.

### **See Also**

[.OPTION POST](#)

---

## **.OPTION CSHDC**

Adds capacitance from each node to ground; used only with the `CONVERGE` option.

### **Syntax**

```
.OPTION CSHDC=x
```

### **Description**

Use this option to add capacitance from each node to ground. This is the same option as `CSHUNT`; use `CSHDC` only with the `CONVERGE` option. When defined, `.OPTION CSHDC` is the same as `.OPTION CSHUNT`, except that `CSHDC` becomes invalid after DC OP analysis, while `CSHUNT` stays in both DC OP and transient analysis.

### **See Also**

[.OPTION CONVERGE](#)  
[.OPTION CSHUNT](#)



## **.OPTION CSHUNT**

Adds capacitance from each node to ground.

### **Syntax**

```
.OPTION CSHUNT=x
```

**Default**    0

### **Description**

Use this option to add capacitance from each node to ground. Add a small CSHUNT to each node to solve internal “timestep too small” timestep problems caused by high frequency oscillations or numerical noise. When defined, .OPTION CSHUNT is the same as .OPTION CSHDC, except that CSHDC becomes invalid after DC OP analysis, while CSHUNT stays in both DC OP and transient analysis.

### **Examples**

```
.option gshunt=1e-13 cshunt=1e-17  
.option gshunt=1e-12 cshunt=1e-16  
.option gshunt=1e-11 cshunt=5e-15  
.option gshunt=1e-10 cshunt=1e-15  
.option gshunt=1e-9  cshunt=1e-14
```

### **See Also**

[.OPTION CSHDC](#)  
[.OPTION GSHUNT](#)

## **.OPTION CUSTCMI**

Turns on gate direct tunneling current modeling and additional instance parameter support.

### **Syntax**

```
.OPTION CUSTCMI= 0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to turn on gate direct tunneling current modeling and instance parameter support. Set `.OPTION CUSTCMI=1` jointly with `.OPTION CMIFLAG` to turn on gate direct tunneling current modeling and instance parameters.

`.OPTION CUSTCMI=0` turns off the feature.

The existing HSPICE BSIM4-like instance parameters include: geomod, acnqsmod, delk1, delnfct, deltox, min, mulu0, nf, rbdb, rbodymod, rbpb, rbpd, rbps, rbsb, rgatemod, sa, sa1, sa10,sa2, sa3, sa4, sa5, sa6, sa7,sa8, sa9, sb, sb1, sb10, sb2,sb3, sb4, sb5, sb6, sb7, sb8,sb9, sd, stimod, sw1, sw10, sw2, sw3, sw4, sw5, sw6, sw7, sw8, sw9, and trnqsmod.

`.OPTION CUSTCMI=1` also supports the six integer instance model flags: `insflg1`, `insflg2`, `insflg3`, `insflg4`, `insflg5`, and `insflg6` and the ten double precision instance parameters supported for customer CMI: `insprm1`, `insprm2`, `insprm3`, `insprm4`,...,`insprm10`.

### **See Also**

[.OPTION CMIFLAG](#)

## **.OPTION CVTOL**

Changes the number of numerical integration steps when calculating the gate capacitor charge for a MOSFET.

### **Syntax**

```
.OPTION CVTOL=x
```

### **Description**

Use this option to change the number of numerical integration steps when calculating the gate capacitor charge for a MOSFET by using `CAPOP=3`. See the discussion of `CAPOP=3` in the “[Overview of MOSFET Models](#)” chapter of the *HSPICE Reference Manual: MOSFET Models* for explicit equations and discussion.

## **.OPTION D\_IBIS**

Specifies the directory containing the IBIS files.

### **Syntax**

```
.OPTION D_IBIS='ibis_files_directory'
```

### **Description**

Use this option to specify the directory containing the IBIS files. If you specify several directories, the simulation looks for IBIS files in the local directory (the directory from which you run the simulation). It then checks the directories specified through `.OPTION D_IBIS` in the order that `.OPTION` cards appear in the netlist. You can use the `D_IBIS` option to specify up to 40 directories.

### **Examples**

```
.OPTION d_ibis='/home/user/ibis/models'
```

## **.OPTION DCAP**

Specifies equations used to calculate depletion capacitance for Level 1 and 3 diodes and BJTs.

### **Syntax**

```
.OPTION DCAP
```

### **Description**

Use this option to specify equations for HSPICE to use when calculating depletion capacitance for Level 1 and 3 diodes and BJTs. The *HSPICE Reference Manual: Elements and Device Models* describes these equations in the section [Using Diode Capacitance Equations](#).

## **.OPTION DCCAP**

Generates C-V plots.

### **Syntax**

```
.OPTION DCCAP=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to generate C-V plots. Prints capacitance values of a circuit (both model and element) during a DC analysis. You can use a DC sweep of the capacitor to generate C-V plots. If not set, MOS device or voltage-variable capacitance values are not evaluated and the printed value is zero. When doing C-V curves for devices, make sure you set `.OPTION DCCAP` so that the capacitance values can be output. Depending on the MOS model level you are using, make sure that you use the appropriate model templates for the models.

### **See Also**

[.DC](#)

[MOSFET Device Examples](#), for paths to demo files *gatecap.sp* and *mosivcv.sp*.

## **.OPTION DCFOR**

Sets the number of iterations to calculate after a circuit converges in the steady state.

### **Syntax**

```
.OPTION DCFOR=x
```

**Default**    0

### **Description**

Use this option to set the number of iterations to calculate after a circuit converges in the steady state. The number of iterations after convergence is usually zero, so `DCFOR` adds iterations (and computation time) to the DC circuit solution. `DCFOR` ensures that a circuit actually, not falsely, converges.

Use this option with `.OPTIONDCHOLD` and the `.NODESET` command to enhance DC convergence.

### **See Also**

- [.DC](#)
- [.NODESET](#)
- [.OPTION DCHOLD](#)

## .OPTION DCHOLD

Specifies how many iterations to hold a node at the .NODESET voltage values.

### Syntax

```
.OPTION DCHOLD=n
```

**Default** 1

### Description

Use this option to specify how many iterations to hold a node at the .NODESET voltage values.

**Note:** In HSPICE RF, this option is ignored; it is replaced by automated algorithms.

Use DCFOR and DCHOLD together to initialize DC analysis. DCFOR and DCHOLD enhance the convergence properties of a DC simulation. DCFOR and DCHOLD work with the .NODESET command. The effects of DCHOLD on convergence differ, according to the DCHOLD value and the number of iterations before DC convergence.

If a circuit converges in the steady state in fewer than DCHOLD iterations, the DC solution includes the values set in .NODESET.

If a circuit requires more than DCHOLD iterations to converge, HSPICE ignores the values set in the .NODESET command, and calculates the DC solution by setting the .NODESET fixed-source voltages as open circuited.

### See Also

- [.DC](#)
- [.NODESET](#)
- [.OPTION DCFOR](#)



## **.OPTION DCIC**

Specifies whether to use or ignore `.IC` commands in the netlist.

### **Syntax**

```
.OPTION DCIC=0 | 1
```

### **Description**

Use this option to specify whether to use or ignore `.IC` commands in the netlist.

- `DCIC=1` (default): Each point in a DC sweep analysis acts like an operating point and all `.IC` commands in the netlist are used.
- `DCIC=0`: `.IC` commands in the netlist are ignored for DC sweep analysis.

### **See Also**

[.IC](#)  
[.DC](#)

---

## .OPTION DCON

Aids in the auto-convergence routines; can also disable autoconverge routines when set to =-1.

### Syntax

.OPTION DCON=x

**Default** Value if option is not specified in the netlist: 0

Value if option name is specified without a corresponding value: 1

### Description

This option aids in the autoconvergence routines.

When DCON equals

- -1: Disables convergence routines, Steps 2 and 3 of the HSPICE autoconverge process (when DCON=-1 and .OPTION CONVERGE=-1).
- 0: Enables autoconvergence routines as designed
- 1: If a circuit cannot converge using Newton-Raphson, HSPICE automatically sets DCON=1 and calculates the following:

$$DV = \max\left(0.1, \frac{V_{max}}{50}\right), \text{ if } DV = 1000$$

$$GRAMP = \max\left(6, \log_{10}\left(\frac{I_{max}}{GMINDC}\right)\right) \quad ITL1 = ITL1 + 20 \cdot GRAMP$$

- 2: If the circuit still cannot converge, HSPICE sets DCON=2, which sets DV=1e6.

### See Also

[.OPTION CONVERGE](#)

[.OPTION DV](#)

[Autoconverge Process](#)

## **.OPTION DCSTEP**

Converts DC model and element capacitors to a conductance.

### **Syntax**

```
.OPTION DCSTEP=n
```

**Default** 0 (seconds)

### **Description**

Use this option to convert DC model and element capacitors to a conductance to enhance DC convergence properties. HSPICE divides the value of the element capacitors by `DCSTEP` to model DC conductance.

### **See Also**

[.DC](#)

## **.OPTION DCTRAN**

Invokes different methods to solve nonconvergence problems.

### **Syntax**

```
.OPTION DCTRAN=x
```

### **Description**

Use this option to run different methods to solve nonconvergence problems.  
DCTRAN is an alias for CONVERGE.

### **See Also**

[.OPTION CONVERGE](#)

## **.OPTION DEFAD**

Sets the default MOSFET drain diode area.

### **Syntax**

```
.OPTION DEFAD=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to set the default MOSFET drain diode area.

## **.OPTION DEFAS**

Sets the default MOSFET source diode area.

### **Syntax**

```
.OPTION DEFAS=x
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to set the default MOSFET source diode area.

## **.OPTION DEFL**

Sets the default MOSFET channel length.

### **Syntax**

```
.OPTION DEFL=x
```

**Default** 100.00u

### **Description**

Use this option to set the default MOSFET channel length.

---

## **.OPTION DEFNRD**

Sets the default number of squares for the drain resistor on a MOSFET.

### **Syntax**

```
.OPTION DEFNRD=n
```

**Default**    0

### **Description**

Use this option to set the default number of squares for the drain resistor on a MOSFET.



## **.OPTION DEFNRS**

Sets the default number of squares for the source resistor on a MOSFET.

### **Syntax**

```
.OPTION DEFNRS= n
```

**Default**    0

### **Description**

Use this option to set the default number of squares for the source resistor on a MOSFET.

## **.OPTION DEFDPD**

Sets the default MOSFET drain diode perimeter.

### **Syntax**

```
.OPTION DEFDPD=n
```

**Default**    0

### **Description**

Use this option to set the default MOSFET drain diode perimeter.

## **.OPTION DEFPS**

Sets the default MOSFET source diode perimeter.

### **Syntax**

```
.OPTION DEFPS=x
```

**Default**    0

### **Description**

Use this option to set the default MOSFET source diode perimeter.

---

## **.OPTION DEFSA**

Sets the default BSIM4 MOSFET SA parameter in HSPICE.

### **Syntax**

```
.OPTION DEFSA=x
```

**Default**    0.0

### **Description**

Use this option to set the default distance between the S/D diffusion edge to the poly gate edge from one side in the BSIM STI/LOD model.

## **.OPTION DEFSB**

Sets the default BSIM4 MOSFET SB parameter.

### **Syntax**

```
.OPTION DEFSB=x
```

**Default** 0.0

### **Description**

Use this option to set the default distance between the S/D diffusion edge to the poly gate edge from side opposite the SA side in the BSIM STI/LOD model.

## **.OPTION DEFSD**

Sets default for BSIM4 MOSFET SD parameter.

### **Syntax**

```
.OPTION DEFSD=x
```

**Default** 0.0

### **Description**

Use this option to set the default for the distance between neighboring fingers (SD parameter) in a BSIM STI/LOD model.

## **.OPTION DEFW**

Sets the default MOSFET channel width.

### **Syntax**

```
.OPTION DEFW=x
```

**Default** 100.00u

### **Description**

Use this option to set the default MOSFET channel width. The default is  $1e-4m$ .

---

## **.OPTION DEGF**

Sets the device's failure criteria for lifetime computation when using the MOSRA API if no values are set for `.OPTIONS DEGFN` or `DEGFP`.

### **Syntax**

```
.OPTION DEGF=val
```

### **Description**

This option is used in conjunction with `.OPTION MOSRALIFE`. For NMOS, `DEGFN` is used. If `DEGFN` is not defined, `DEGF` is used instead.

For PMOS, `DEGFP` is used. If `DEGFP` is not defined, `DEGF` is used instead. This option sets the device's degradation value at lifetime. The options apply to all MOSFETs. The lifetime values are printed in the RADEG file.

### **See Also**

- [.OPTION DEGFN](#)
- [.OPTION DEGFP](#)
- [.OPTION MOSRALIFE](#)



## **.OPTION DEGFN**

Sets the NMOS's failure criteria for lifetime computation when using the MOSRA API.

### **Syntax**

```
.option DEGFN=val
```

### **Description**

This option is used in conjunction with `.OPTION MOSRALIFE`. This option sets the PMOS's degradation value at lifetime. If the option is not specified or the keyword can not be identified by the `MRAlifetimeDeg` function, HSPICE substitutes `.OPTION DEGF` for lifetime computation. The options apply to all MOSFETs. The lifetime values are printed in the RADEG file.

### **See Also**

- [.OPTION DEGF](#)
- [.OPTION DEGFP](#)
- [.OPTION MOSRALIFE](#)

---

## **.OPTION DEGFP**

Sets the PMOS's failure criteria for lifetime computation when using the MOSRA API.

### **Syntax**

```
.option DEGFP= val
```

### **Description**

This option is used in conjunction with `.OPTION MOSRALIFE`. This option sets the PMOS's degradation value at lifetime. If the option is not specified or the keyword can not be identified by the `MRAlifetimeDeg` function, HSPICE substitutes `.OPTION DEGF` for lifetime computation. The options apply to all MOSFETs. The lifetime values are printed in the RADEG file.

### **See Also**

- [.OPTION DEGF](#)
- [.OPTION DEGFN](#)
- [.OPTION MOSRALIFE](#)

---

## .OPTION DELMAX

Sets the maximum allowable step size of the timesteps taken during transient analysis in HSPICE/HSPICE RF.

### Syntax

```
.OPTION DELMAX=x
```

**Default** (Computed automatically)

### Description

Use this option to set the maximum allowable step size of the internal timestep. The maximum internal timestep taken by HSPICE during transient analysis is referred to as  $\Delta t_{max}$ . Its value is normally computed automatically based on several timestep control settings. If you wish to override the automatically computed value, and force the maximum step size to be a specific value, you can do so with `.OPTION DELMAX`, or by specifying a *delmax* value with the `.TRAN` command. If not specified, HSPICE automatically computes a DELMAX “auto” value, based on timestep control factors such as FS and RMAX.

The initial calculated DELMAX “auto” value, shown in the output listing, is generally not the value used for simulation. The calculated DELMAX value is automatically adjusted by the timestep control methods, DVDT, RUNLVL and LVLTIM.

If DELMAX is defined in an `.OPTION` command, its priority is higher than the value given with a `.TRAN` command and it overrides the DELMAX “auto” value calculations. Min value: -1e10; Max value 1e10.

### See Also

- [.TRAN](#)
- [.OPTION DVDT](#)
- [.OPTION RUNLVL](#)
- [.OPTION LVLTIM](#)
- [.OPTION FS](#)
- [.OPTION RMAX](#)

[Appendix B, How Options Affect other Options](#)

---

## **.OPTION DI**

Sets the maximum iteration to iteration current change in HSPICE.

### **Syntax**

```
.OPTION DI=n
```

**Default**    100.00

### **Description**

Use this option to set the maximum iteration to iteration current change through voltage-defined branches (voltage sources and inductors). Use this option only if the value of the `ABSH` control option is greater than 0.

### **See Also**

[.OPTION ABSH](#)

## **.OPTION DIAGNOSTIC (or) .OPTION DIAGNO**

Logs the occurrence of negative model conductances.

### **Syntax**

```
.OPTION DIAGNOSTIC
```

### **Description**

Use this option to log the occurrence of negative model conductances.

---

## **.OPTION DLENCSDF**

Specifies how many digits to include in scientific notation (exponents) or to the right of the decimal point when using Common Simulation Data Format.

### **Syntax**

```
.OPTION DLENCSDF=x
```

**Default**    5

### **Description**

If you use the Common Simulation Data Format (Viewlogic graph data file format) as the output format, this digit length option specifies how many digits to include in scientific notation (exponents) or to the right of the decimal point. Valid values are any integer from 1 to 10.

If you assign a floating decimal point or if you specify less than 1 or more than 10 digits, HSPICE uses the default. For example, it places 5 digits to the right of a decimal point.

## **.OPTION DV**

Specifies maximum iteration to iteration voltage change for all circuit nodes in both DC and transient analyses.

### **Syntax**

```
.OPTION DV=x
```

**Default**    1.00k

### **Description**

Use this option to specify maximum iteration to iteration voltage change for all circuit nodes in both DC and transient analysis. High-gain bipolar amplifiers can require values of 0.5 to 5.0 to achieve a stable DC operating point. Large CMOS digital circuits frequently require about 1 V. The default is 1000 (or 1e6 if DCON=2).

### **See Also**

- [.DC](#)
- [.OPTION DCON](#)
- [.TRAN](#)

## .OPTION DVDT

Adjusts the timestep based on rates of change for node voltage.

### Syntax

```
.OPTION DVDT=0 | 1 | 2 | 3 | 4
```

**Default** 4 (regardless of `runlvl` setting)

### Description

Use this option to adjust the timestep based on rates of change for node voltage.

- 0: Original algorithm
- 1: Fast
- 2: Accurate
- 3, 4: Balance speed and accuracy
- The `ACCURATE` option also increases the accuracy of the results.

For additional information, see “[DVDT Dynamic Timestep](#)” in the *HSPICE User Guide: Simulation and Analysis*.

For information on how DVDT values impact other options, see [Appendix B, How Options Affect other Options](#).

### See Also

[.OPTION ACCURATE](#)  
[.OPTION DELMAX](#)



## **.OPTION DVTR**

Limits the voltage in transient analysis.

### **Syntax**

```
.OPTION DVTR=x
```

**Default** 1.00k

### **Description**

Use this option to limit the voltage in transient analysis. The default is 1000.

## **.OPTION DYNACC**

(Optimization) Dynamic accuracy tolerance setting to accelerate bisection simulation.

### **Syntax**

```
.OPTION DYNACC = 0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

When `DYNACC=1`, if HSPICE is in accuracy mode, it uses reduced accuracy simulations to narrow the bisection window, then switches to the original accuracy algorithm to refine the solution. This method reduces simulation time by doing the majority of simulations at lower accuracy, which run faster by taking fewer time steps. If `DYNACC` is set using the `.OPTION` command, the setting of `DYNACC` in `.model` card is overridden.

### **See Also**

[.MODEL](#)

## **.OPTION EM\_RECOVERY**

Provides a coefficient value for measuring “recovered” average current such as electromigration for bipolar currents.

### **Syntax**

```
.OPTION EM_RECOVERY=value
```

**Default**    1

### **Description**

This option is used in a transient analysis with the `.MEAS` keyword `em_avg` (electromigration average) using the From-To function. `.OPTION EM_RECOVERY` assists in measuring “recovered” average current from an electromigration perspective. The option can have a coefficient value between 0.0 and 1.0. Recovered average current is especially meaningful for bipolar currents (for example output of the inverter), as the mathematical average for such a waveform is zero.

### **Examples**

```
.option em_recovery=0.9
```

### **See Also**

[.MEASURE \(AVG, EM\\_AVG, INTEG, MIN, MAX, PP, and RMS\)](#)

## **.OPTION EPSMIN**

Specifies the smallest number a computer can add or subtract.

### **Syntax**

```
.OPTION EPSMIN=x
```

### **Description**

Use this option to specify the smallest number that a computer can add or subtract, a constant value. This options helps avoid zero denominator issues.

## **.OPTION EXPLI**

Enables the current-explosion model parameter.

### **Syntax**

```
.OPTION EXPLI=x
```

**Default** 0 (amp/area effective)

### **Description**

Use this option to enable the current-explosion model parameter. PN junction characteristics, above the explosion current are linear. HSPICE/HSPICE RF determines the slope at the explosion point. This improves simulation speed and convergence.

### **See Also**

[BJT and Diode Examples](#) for the path to the demo file *bjtgmsp*, which uses `.OPTION EXPLI=10`.

---

## **.OPTION EXPMAX**

Specifies the largest exponent that you can use for an exponential before overflow occurs.

### **Syntax**

```
.OPTION EXPMAX=x
```

**Default**    80.00

### **Description**

Use this option to specify the largest exponent that you can use for an exponential before overflow occurs. Typical value for an IBM platform is 350.

## .OPTION FAST

Disables status updates for latent devices; this speeds up simulation.

### Syntax

```
.OPTION FAST=[0|1]
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### Description

Use this option to set additional options, which increase simulation speed with minimal loss of accuracy.

To speed up simulation, this option disables status updates for latent devices. Use this option for MOSFETs, MESFETs, JFETs, BJTs, and diodes.

A device is latent if its node voltage variation (from one iteration to the next) is less than the value of either the `BYTOL` control option or the `BYPASSTOL` element parameter. (If `FAST` is on, HSPICE sets `BYTOL` to different values for different types of device models.)

Besides the `FAST` option, you can also use the `NOTOP` and `NOELCK` options to reduce input preprocessing time. Increasing the value of the `MBYPASS` or `BYTOL` option, also helps simulations to run faster, but can reduce accuracy. To see how use of `FAST` impacts the value settings of other options, see [Appendix B, How Options Affect other Options](#).

### See Also

- [.OPTION BYTOL](#)
- [.OPTION MBYPASS](#)
- [.OPTION NOELCK](#)
- [.OPTION NOTOP](#)

---

## **.OPTION FFT\_ACCURATE**

Dynamically adjusts the time step so that each FFT point is a real simulation point in HSPICE/HSPICE RF.

### **Syntax**

```
.OPTION FFT_ACCURATE= [0 | 1]
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to dynamically adjust the time step so that each FFT point is a real simulation point. This eliminates interpolation error and provides the highest FFT accuracy with minimal overhead in simulation time.

### **See Also**

[.OPTION ACCURATE](#)

[.OPTION SIM\\_ACCURACY \(RF\)](#)



## **.OPTION FFTOUT**

Prints 30 harmonic fundamentals.

### **Syntax**

```
.OPTION FFTOUT=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to print 30 harmonic fundamentals sorted by size, THD, SNR, and SFDR, but only if you specify a `FFTOUT` option and a `.FFTfreq=xxx` command.

### **See Also**

[.FFT](#)

## **.OPTION FMAX**

Sets the maximum frequency value of angular velocity, for poles and zeros.

### **Syntax**

```
.OPTION FMAX=x
```

**Default**    1.0e+12

### **Description**

Use this option to set the maximum frequency value of angular velocity for Pole/Zero analysis. The units of value are in rad/sec.

### **See Also**

- [.OPTION CSCAL](#)
- [.OPTION FSCAL](#)
- [.OPTION GSCAL](#)
- [.OPTION ITLPZ](#)
- [.OPTION LSCAL](#)
- [.OPTION PZABS](#)
- [.OPTION PZTOL](#)
- [.OPTION RITOL](#)
- [.PZ](#)

---

## .OPTION FS

Decreases FS value to help circuits that have timestep convergence difficulties.

### Syntax

```
.OPTION FS=x
```

### Description

Use this option to decrease delta (internal timestep) by the specified fraction of a timestep (TSTEP) for the first time point of a transient. Decreases the FS value to help circuits that have timestep convergence difficulties. DVDT=3 uses FS to control the timestep.  $Delta = FS \cdot [MIN(TSTEP, DELMAX, BKPT)]$

- You specify DELMAX.
- BKPT is related to the breakpoint of the source.
- The .TRAN command sets TSTEP.

### See Also

[.OPTION DELMAX](#)

[.OPTION DVDT](#)

[.TRAN](#)

## **.OPTION FSCAL**

Sets the frequency scale for Pole/Zero analysis.

### **Syntax**

```
.OPTION FSCAL=x
```

**Default**    1e-9

### **Description**

Use this option to set the frequency scale for Pole/Zero analysis. HSPICE multiplies capacitances by `FSCAL`.

### **See Also**

- [.OPTION CSCAL](#)
- [.OPTION FMAX](#)
- [.OPTION GSCAL](#)
- [.OPTION ITLPZ](#)
- [.OPTION LSCAL](#)
- [.OPTION PZABS](#)
- [.OPTION PZTOL](#)
- [.OPTION RITOL](#)
- [.PZ](#)

## **.OPTION FT**

Decreases delta by a specified fraction of a timestep for iteration set that does not converge.

### **Syntax**

```
.OPTION FT=x
```

### **Description**

Use this option to decrease delta (the internal timestep) by a specified fraction of a timestep (TSTEP) for an iteration set that does not converge. If DVDT=2 or DVDT=4, FT controls the timestep.

### **See Also**

[.OPTION DVDT](#)  
[.TRAN](#)

## **.OPTION GDCPATH**

Adds conductance to nodes having no DC path to ground.

### **Syntax**

```
.OPTION GDCPATH [=x]
```

**Default** 1e-12

### **Description**

Use this option to add conductance to nodes having no DC path to ground.

## **.OPTION GENK**

Automatically computes second-order mutual inductance for several coupled inductors.

### **Syntax**

```
.OPTION GENK= 0 | 1
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to automatically calculate second-order mutual inductance for several coupled inductors. The default (1) enables the calculation.

---

## .OPTION GEOSHRINK

Element scaling factor used with `.OPTION SCALE`.

### Syntax

```
.OPTION GEOSHRINK=x
```

### Description

Use this option as a global model to apply to all elements. In addition to `.OPTION SCALE`, use this option (usually through a technology file) on top of the existing `scale` option to further scale geometric element instance parameters whose default units are meters. The final instance geometric parameters are then be calculated as:  $final\_dimension = original\_dimension * SCALE * GEOSHRINK$

The effective scaling factor is the product of the two parameters.

The default value for both `SCALE` and `GEOSHRINK` is 1.

If a model library contains devices other than MOSFET, such as R, L, C, diode, bjt... etc., and/or the netlist is a post-layout design with RCs, the shrink factor is applied to all elements.

### Examples

Example 1: If there is more than one `geoshrink` option set, only the last `geoshrink` is used.

```
.option geoshrink=0.8
.option geoshrink=0.9
```

Then the  $final\_dimension = original\_dimension * SCALE * 0.9$

Example 2: If there is more than one `geoshrink` and `scale` in the model card, only the last `scale` and the last `geoshrink` are used.

```
.option scale=2u
.option scale=1u
.option geoshrink=0.8
.option geoshrink=0.9
```

Then the  $final\_dimension = original\_dimension * 1u * 0.9$

### See Also

[.OPTION SCALE](#)  
[.OPTION CMIUSRFLAG](#)



## **.OPTION GMAX**

Specifies the maximum conductance in parallel with a current source for `.IC` and `.NODESET` initialization circuitry.

### **Syntax**

```
.OPTION GMAX=x
```

**Default** 100.00 (mho)

### **Description**

Use this option to specify the maximum conductance in parallel with a current source for `.IC` and `.NODESET` initialization circuitry. Some large bipolar circuits require you to set `GMAX=1` for convergence.

### **See Also**

[.IC](#)

[.NODESET](#)

---

## **.OPTION GMIN**

Specifies the minimum conductance added to all PN junctions for a time sweep in transient analysis for HSPICE/HSPICE RF.

### **Syntax**

```
.OPTION GMIN=x
```

**Default** 1e-12

### **Description**

Use this option to specify the minimum conductance added to all PN junctions for a time sweep in transient analysis. Min value: 1e-30; Max value: 100.

### **See Also**

[.OPTION GMINDC](#)

## **.OPTION GMINDC**

Specifies conductance in parallel for PN junctions and MOSFET nodes in DC analysis.

### **Syntax**

```
.OPTION GMINDC=x
```

### **Description**

Use this option to specify conductance in parallel for all PN junctions and MOSFET nodes except gates in DC analysis. `GMINDC` helps overcome DC convergence problems caused by low values of off-conductance for pn junctions and MOSFETs. You can use `GRAMP` to reduce `GMINDC` by one order of magnitude for each step. Set `GMINDC` between  $1e-4$  and the `PIVTOL` value. Min value:  $1e-30$ ; Max value: 100.

Large values of `GMINDC` can cause unreasonable circuit response. If your circuit requires large values to converge, suspect a bad model or circuit. If a matrix floating-point overflows and if `GMINDC` is  $1.0e-12$  or less, HSPICE sets it to  $1.0e-11$ . HSPICE manipulates `GMINDC` in auto-converge mode.

### **See Also**

- [.DC](#)
- [.OPTION GRAMP](#)
- [.OPTION PIVTOL](#)

---

## .OPTION GRAMP

Specifies a conductance range over which DC operating point analysis sweeps GMINDC.

### Syntax

```
.OPTION GRAMP=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### Description

Use this option to specify a conductance range over which the DC operating point analysis sweeps GMINDC. HSPICE sets this value during auto-convergence. Use GRAMP with the GMINDC option to find the smallest GMINDC value that results in DC convergence.

GRAMP specifies a conductance range over which the DC operating point analysis sweeps GMINDC. HSPICE replaces GMINDC values over this range, simulates each value, and uses the lowest GMINDC value where the circuit converges in a steady state.

If you sweep GMINDC between  $1e-12$  mhos (default) and  $1e-6$  mhos, GRAMP is 6 (value of the exponent difference between the default and the maximum conductance limit). In this example:

- HSPICE first sets GMINDC to  $1e-6$  mhos and simulates the circuit.
- If circuit simulation converges, HSPICE sets GMINDC to  $1e-7$  mhos and simulates the circuit.
- The sweep continues until HSPICE simulates all values of the GRAMP ramp.

If the combined GMINDC and GRAMP conductance is greater than  $1e-3$  mho, false convergence can occur.

Min value: 0; Max value: 1000.

### See Also

[.DC](#)  
[.OPTION GMINDC](#)

## **.OPTION GSCAL**

Sets the conductance scale for Pole/Zero analysis.

### **Syntax**

```
.OPTION GSCAL=x
```

**Default** 1e+3

### **Description**

Use this option to set the conductance scale for Pole/Zero analysis. HSPICE multiplies the conductance and divides the resistance by `GSCAL`.

### **See Also**

- [.OPTION CSCAL](#)
- [.OPTION FMAX](#)
- [.OPTION FMAX](#)
- [.OPTION FSCAL](#)
- [.OPTION GSCAL](#)
- [.OPTION LSCAL](#)
- [.OPTION PZABS](#)
- [.OPTION PZTOL](#)
- [.OPTION RITOL](#)
- [.PZ](#)

---

## **.OPTION GSHDC**

Adds conductance from each node to ground when calculating the DC operating point of the circuit.

### **Syntax**

```
.OPTION GSHDC=[0 | 1]
```

**Default** 0

### **Description**

Use this option to add conductance from each node to ground when calculating the DC operating point of the circuit (.OP).

### **See Also**

[.OPTION GSHUNT](#)

## **.OPTION GSHUNT**

Adds conductance from each node to ground.

### **Syntax**

```
.OPTION GSHUNT=x
```

**Default** 0

### **Description**

Use this option to add conductance from each node to ground. Add a small GSHUNT to each node to help solve “timestep too small” problems caused by either high-frequency oscillations or numerical noise.

### **Examples**

```
.option gshunt=1e-13 cshunt=1e-17  
.option gshunt=1e-12 cshunt=1e-16  
.option gshunt=1e-11 cshunt=5e-15  
.option gshunt=1e-10 cshunt=1e-15  
.option gshunt=1e-9 cshunt=1e-14
```

### **See Also**

[.OPTION CSHUNT](#)  
[.OPTION GSHDC](#)

## **.OPTION HBACKRYLOVDIM**

Specifies the dimension of the Krylov subspace used by the Krylov solver.

### **Syntax**

```
.OPTION HBACKRYLOVDIM=value
```

**Default**    300

### **Description**

Use this option to specify the dimension of the Krylov subspace that the Krylov solver uses.

The *value* parameter must specify an integer greater than zero. The range is 1 to infinity.

This option overrides the corresponding PAC option if specified in the netlist.

When this option is not specified in the netlist if HBACKRYLOVDIM < HBKRYLOVDIM, then HBACKRYLOVDIM = HBKRYLOVDIM.

### **See Also**

[.HB](#)

[.OPTION HBKRYLOVDIM](#)



## **.OPTION HBACKRYLOVITER (or) HBAC\_KRYLOV\_ITER**

Specifies the number of GMRES solver iterations performed by the HB engine.

### **Syntax**

```
.OPTION HBACKRYLOVITER | HBAC_KRYLOV_ITER = value
```

### **Description**

Use this option to specify the number of Generalized Minimum Residual (GMRES) solver iterations that the HB engine performs.

The *value* parameter must specify an integer greater than zero. The range is 1 to infinity.

This option overrides the corresponding PAC option if specified in the netlist.

### **See Also**

[.HBAC](#)

[.OPTION HBKRYLOVDIM](#)

## **.OPTION HBACTOL**

Specifies the absolute error tolerance for determining convergence.

### **Syntax**

`.OPTION HBACTOL=value`

**Default**    `1.e-8`

### **Description**

Use this option to specify the absolute error tolerance for determining convergence. The *value* parameter must specify a real number greater than zero. The range is 1.e-14 to infinity.

This option overrides the corresponding PAC option if specified in the netlist.

When this option is not specified in the netlist if `HBACTOL > HBTOL`, then `HBACTOL = HBTOL`.

### **See Also**

[.HB](#)

[.OPTION HBTOL](#)

## **.OPTION HBCONTINUE**

Specifies whether to use the sweep solution from the previous simulation as the initial guess for the present simulation.

### **Syntax**

```
.OPTION HBCONTINUE= 0 | 1
```

**Default**    1

### **Description**

Use this option to specify whether to use the sweep solution from the previous simulation as the initial guess for the present simulation.

- HBCONTINUE=1 Use solution from previous simulation as the initial guess.
- HBCONTINUE=0: Start each simulation in a sweep from the DC solution.

### **See Also**

[.HB](#)

## **.OPTION HBFREQABSTOL**

Specifies the maximum absolute change in frequency between solver iterations for convergence.

### **Syntax**

```
.OPTION HBFREQABSTOL=value
```

**Default** 1Hz

### **Description**

Use this option to specify the maximum absolute change in frequency between solver iterations for convergence.

This option is an additional convergence criterion for oscillator analysis.

### **See Also**

[.HBOSC](#)

## **.OPTION HBFREQRELTOL**

Specifies the maximum relative change in frequency between solver iterations for convergence.

### **Syntax**

```
.OPTION HBFREQRELTOL=value
```

### **Description**

Use this option to specify the maximum relative change in frequency between solver iterations for convergence.

This option is an additional convergence criterion for oscillator analysis.

### **See Also**

[.HBOSC](#)

---

## .OPTION HB\_GIBBS

Option for HBTRAN output to minimize Gibbs' phenonema.

### Syntax

```
.OPTION HB_GIBBS=n
```

**Default**    0

### Description

Minimize any Gibbs' phenomenon that may occur in transforming a square-wave signal from the frequency domain to the time domain.  $n \geq 0$  (defaults to zero, which is equivalent to not using it at all). The result is that the HBTRAN waveforms are filtered by a  $(\text{sinc}(x))^N$  function before being transformed to the time domain via FFT. This option applies only to single-tone output.

### Examples

```
.option hb_gibbs = 2  
...  
.print hbtran v(2)
```

### See Also

The *HSPICE User Guide: RF Analysis*, [Minimizing Gibbs Phenomenon](#)

## **.OPTION HBJREUSE**

Controls when to recalculate the Jacobson matrix.

### **Syntax**

```
.OPTION HBJREUSE=0 | 1
```

**Default** Conditional, see below

### **Description**

Use this option to control when to recalculate the Jacobson matrix.

- **HBJREUSE=0** : Recalculates the Jacobian matrix at each iteration. This is the default if **HBSOLVER=1**.
- **HBJREUSE=1** : Reuses the Jacobian matrix for several iterations if the error is sufficiently reduced. This is the default if **HBSOLVER=0**.

### **See Also**

[.HB](#)

[.OPTION HBSOLVER](#)

---

## **.OPTION HBJREUSETOL**

Determines when to recalculate Jacobian matrix if HBJREUSE=1 . 0.

### **Syntax**

`.OPTION HBJREUSETOL=value`

### **Description**

Determines when to recalculate Jacobian matrix (if HBJREUSE=1 . 0).

This is the percentage by which HSPICE RF must reduce the error from the last iteration so you can use the Jacobian matrix for the next iteration. The *value* parameter must specify a real number between 0 and 1.

### **See Also**

[.HB](#)

[.OPTION HBJREUSE](#)



## **.OPTION HBKRYLOVDIM**

Specifies the dimension of the subspace used by the Krylov solver.

### **Syntax**

```
.OPTION HBKRYLOVDIM=value
```

### **Description**

Use this option to specify the dimension of the Krylov subspace that the Krylov solver uses.

The *value* parameter must specify an integer greater than zero.

### **See Also**

[.HB](#)

## **.OPTION HBKRYLOVTOL**

Specifies the error tolerance for the Krylov solver.

### **Syntax**

`.OPTION HBKRYLOVTOL=value`

**Default**    0.01

### **Description**

Use this option to specify the error tolerance for the Krylov solver.

The *value* parameter must specify a real number greater than zero.

### **See Also**

[.HB](#)

## **.OPTION HBKRYLOVMAXITER (or) HB\_KRYLOV\_MAXITER**

Specifies the maximum number of GMRES solver iterations performed by the HB engine.

### **Syntax**

```
.OPTION HBKRYLOVMAXITER | HB_KRYLOV_MAXITER =value
```

**Default** 500

### **Description**

Use this option to specify the maximum number of Generalized Minimum Residual (GMRES) solver iterations that the HB engine performs.

Analysis stops when the number of iterations reaches this value.

### **See Also**

[.HB](#)

## **.OPTION HBLINESEARCHFAC**

Specifies the line search factor.

### **Syntax**

```
.OPTION HBLINESEARCHFAC=value
```

**Default**    0.35

### **Description**

Use this option to specify the line search factor.

If Newton iteration produces a new vector of HB unknowns with a higher error than the last iteration, then scale the update step by this value and try again. The *value* parameter must specify a real number between 0 and 1.

### **See Also**

[.HB](#)

## **.OPTION HBMAXITER (or) HB\_MAXITER**

Specifies the maximum number of Newton-Raphson iterations performed by the HB engine.

### **Syntax**

```
.OPTION HBMAXITER | HB_MAXITER=value
```

**Default**    10000

### **Description**

Use this option to specify the maximum number of Newton-Raphson iterations that the HB engine performs.

Analysis stops when the number of iterations reaches this value.

### **See Also**

[.HB](#)

---

## **.OPTION HBOSCMAXITER (or) HBOSC\_MAXITER**

Specifies the maximum number of outer-loop iterations for oscillator analysis.

### **Syntax**

```
.OPTION HBOSCMAXITER | HBOSC_MAXITER=value
```

**Default** 10000

### **Description**

Use this option to specify the maximum number of outer-loop iterations for oscillator analysis.

### **See Also**

[.HBOSC](#)

## **.OPTION HBPROBETOL**

Searches for a probe voltage at which the probe current is less than the specified value.

### **Syntax**

```
.OPTION HBPROBETOL=value
```

**Default**    1.e-9

### **Description**

Use this option to cause oscillator analysis to try to find a probe voltage at which the probe current is less than the specified value.

This option defaults to the value of the `HBTOL` option, which defaults to 1.e-9.

### **See Also**

[.HBOSC](#)  
[.OPTION HBTOL](#)

## **.OPTION HBSOLVER**

Specifies a preconditioner for solving nonlinear circuits.

### **Syntax**

```
.OPTION HBSOLVER=0 | 1 | 2
```

**Default** 1

### **Description**

Use this option to specify a preconditioner for solving nonlinear circuits.

- HBSOLVER=0: Invokes the direct solver.
- HBSOLVER=1 Invokes the matrix-free Krylov solver.
- HBSOLVER=2: Invokes the two-level hybrid time-frequency domain solver.

### **See Also**

[.HBOSC](#)

[.OPTION HBJREUSE](#)



## **.OPTION HBTOL**

Specifies the absolute error tolerance for determining convergence.

### **Syntax**

```
.OPTION HBTOL=value
```

### **Description**

Use this option to specify the absolute error tolerance for determining convergence.

The *value* parameter must specify a real number greater than zero.

### **See Also**

[.HB](#)

## **.OPTION HBTRANFREQSEARCH**

Specifies the frequency source for the HB analysis of a ring oscillator.

### **Syntax**

```
.OPTION HBTRANFREQSEARCH= [1 | 0]
```

**Default** 1

### **Description**

Use this option to specify the frequency source for the HB analysis of a ring oscillator.

- `HBTRANFREQSEARCH=1`: HB analysis calculates the oscillation frequency from the transient analysis
- `HBTRANFREQSEARCH=0`: HB analysis assumes that the period is  $1/f$ , where  $f$  is the frequency specified in the tones description.

### **See Also**

[.HB](#)  
[.HBOSC](#)  
[.OPTION HBTOL](#)

## **.OPTION HBTRANINIT**

Selects transient analysis for initializing all state variables for HB analysis of a ring oscillator.

### **Syntax**

```
.OPTION HBTRANINIT=time
```

### **Description**

Use this option to cause HB to use transient analysis to initialize all state variables for HB analysis of a ring oscillator.

The *time* parameter is defined by when the circuit has reached (or is near) steady-state. The default is 0.

### **See Also**

[.HB](#)  
[.HBOSC](#)

---

## .OPTION HBTRANPTS

Specifies the number of points per period for converting time-domain data results into the frequency domain for HB analysis of a ring oscillator.

### Syntax

```
.OPTION HBTRANPTS=npts
```

**Default**     $4 * nh$

### Description

Use this option to specify the number of points per period for converting the time-domain data results from transient analysis into the frequency domain for HB analysis of a ring oscillator.

The *npts* parameter must be set to an integer greater than 0. The units are in nharms (nh).

This option is relevant only if you set .OPTION HBTRANINIT. You can specify either .OPTION HBTRANPTS or .OPTION HBTRANSTEP, but not both.

### See Also

- [.HB](#)
- [.HBOSC](#)
- [.OPTION HBTRANINIT](#)
- [.OPTION HBTRANSTEP](#)

## .OPTION HBTRANSTEP

Specifies transient analysis step size for the HB analysis of a ring oscillator.

### Syntax

```
.OPTION HBTRANSTEP=stepsize
```

### Description

Use this option to specify transient analysis step size for the HB analysis of a ring oscillator.

The *stepsize* parameter must be set to a real number. The default is  $1/(4*nh*f0)$ , where *nh* is the nharms value and *f0* is the oscillation frequency.

This option is relevant only if you set `.OPTION HBTRANINIT`.

**Note:** You can specify either `.OPTION HBTRANPTS` or `.OPTION HBTRANSTEP`, but not both.

### See Also

- [.HB](#)
- [.HBOSC](#)
- [.OPTION HBTRANINIT](#)
- [.OPTION HBTRANPTS](#)

---

## **.OPTION HIER\_DELIM**

Replaces the caret delimiter with a period (for output control only) when used for HSPICE/ADE only.

### **Syntax**

```
.OPTION HIER_DELIM= 0 | 1
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### **Description**

Use `.OPTION HIER_DELIM` to change the hierarchy delimiter from a caret (^) to a period (.) only with for the HSPICE integration to Cadence® Virtuoso® ADE. When `.OPTION HIER_DELIM=1`, a caret (^) is changed to a period(.). This option works with `.OPTION PSF` and `.OPTION ARTIST`.

- 0: Maintains the caret.
- 1: Replaces the caret with a period.

### **See Also**

[.OPTION ARTIST](#)  
[.OPTION PSF](#)

---

## .OPTION HIER\_SCALE

Uses the parameter S to scale subcircuits.

### Syntax

```
.OPTION HIER_SCALE=[0|1]
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### Description

Use this option so you can use the parameter S to scale subcircuits.

- 0 Interprets S as a user-defined parameter.
- 1 Interprets S as a scale parameter.

This option enables you to selectively scale the required instance. See the example below.

### Examples

Assume you have an encrypted subcircuit from an IP vendor A which has `.option SCALE=1e-6` defined. You have another encrypted subcircuit (from another IP vendor B), which has the units defined as microns and does not need to be scaled. When you simulate the circuit, HSPICE applies the `SCALE` option globally and the subcircuit from IP vendor B is scaled again. You can selectively apply the `SCALE` option so that this does not happen, as follows:

```
* Top level netlist
.option hier_scale=1
.include "subckt_a.inc" $ subcircuit from IP vendor A
.include "subckt_b.inc" $ subcircuit from IP vendor B
vin in 0 5
x1 in 2 subckt_a $ uses .option scale=1e-6 defined in subckt_a.inc
file
x2 2 0 subckt_b S=1e6 $ scale option is not required
.tran 100p 10n
.end
```

The `subckt_a.inc` file has `.option scale=1u` defined and this is applied globally. When `.option hier_scale=1` is used and the subcircuit instance, X2 contains `S=1e6`, the global scaling is offset. If `W=10u` is used in subcircuit instance X2 and `hier_scale` is used, then:

```
W="10u*SCALE*S"="10u*1u*1e6"=10u
```

### Chapter 3: HSPICE and RF Netlist Simulation Control Options

#### .OPTION HIER\_SCALE

If  $W=10$  is used in subcircuit instance X1 and “S” is not used, then only the global `.option SCALE=1e-6` is applied and the value of  $W$  is  $10\mu$ .



## **.OPTION IC\_ACCURATE**

Improves the accuracy of the `.IC` command.

### **Syntax**

```
.OPTION IC_ACCURATE=0 | 1
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### **Description**

When `.OPTION IC_ACCURATE=1` the `.IC` command accuracy is increased for cases requiring tighter precision (for example, when the `GMAX` value is too large) than is used to set the maximum conductance in parallel with a current source for `.IC` and `.NODESET` initialization circuitry. The option overrides the approximating method used by the `.IC` command with only slight performance cost. If the option is not set or it equals 0, then the default `.IC` method is used.

### **See Also**

[.IC](#)  
[.OPTION GMAX](#)

## **.OPTION ICSWEEP**

Saves the current analysis result of a parameter or temperature sweep as the starting point in the next analysis.

### **Syntax**

```
.OPTION ICSWEEP=0 | 1
```

**Default** 1

### **Description**

Use this option to save the current analysis result of a parameter or temperature sweep as the starting point in the next analysis in the sweep.

- If ICSWEEP=1, the next analysis uses the current results.
- If ICSWEEP=0, the next analysis does not use the results of the current analysis.

## **.OPTION IMAX**

Specifies the maximum timestep in timestep algorithms for transient analysis.

### **Syntax**

```
.OPTION IMAX=x
```

### **Description**

Use this option to specify the maximum timestep in algorithms for transient analysis. `IMAX` sets the maximum iterations to obtain a convergent solution at a timepoint. If the number of iterations needed is greater than `IMAX`, the internal timestep (delta) decreases by a factor equal to the `FT` transient control option. The new timestep calculates a new solution. `IMAX` also works with the `IMIN` transient control option. `IMAX` is the same as `ITL4`.

### **See Also**

- [.OPTION FT](#)
- [.OPTION IMIN](#)
- [.OPTION ITL4](#)

---

## **.OPTION IMIN**

Specifies the minimum timestep in timestep algorithms for transient analysis.

### **Syntax**

```
.OPTION IMIN=x
```

### **Description**

Use this option to specify the minimum number of iterations required to obtain convergence for transient analysis. If the number of iterations is less than `IMIN`, the internal timestep (`delta`) doubles.

Use this option to decrease simulation times in circuits where the nodes are stable most of the time (such as digital circuits). If the number of iterations is greater than `IMIN`, the timestep stays the same unless the timestep exceeds the `IMAX` option. `IMIN` is the same as `ITL3`.

### **See Also**

[.OPTION IMAX](#)

[.OPTION ITL3](#)

## .OPTION INGOLD

Controls whether HSPICE prints *\*/lis* file output in exponential form or engineering notation in HSPICE/HSPICE RF.

### Syntax

```
.OPTION INGOLD= [0 | 1 | 2]
```

**Default** Value if option is not specified in the netlist: 0  
 Value if option name is specified without a corresponding value: 1

Argument	Description
INGOLD=0	Engineering Format; defaults 1.234K, 123M
INGOLD=1	G Format (fixed and exponential); defaults 1.234e+03, .123
INGOLD=2	E Format (exponential SPICE); defaults 1.234e+03, .123e-1

### Description

Use this option to control if HSPICE prints output in exponential form (scientific notation) or engineering notation. Engineering notation provides two to three extra significant digits and aligns columns to facilitate comparison, as:

```
F=1e-15    M=1e-3
P=1e-12    K=1e3
N=1e-9     X=1e6
U=1e-6     G=1e9
```

HSPICE RF prints variable values in engineering notation by default. To use the exponential form, specify `.OPTION INGOLD=1` or `2`. To print variable values in exponential form, specify `.OPTION INGOLD=1` or `2`.

`.OPTION INGOLD` does not control the number format in measure files (*\*.mt#/\*.\*ms#/\*.\*ma#*). If you specify a measure output file using `.OPTION MEASFORM`, HSPICE automatically resets an `INGOLD=0` setting to `INGOLD=1`, which allows the measure file to be imported to Excel when `.OPTION MEASFORM=1`.

### Examples

```
.OPTION INGOLD=2
```

### Chapter 3: HSPICE and RF Netlist Simulation Control Options

.OPTION INGOLD

#### See Also

[.OPTION MEASDGT](#)

[.OPTION MEASFORM](#)

---

## .OPTION INTERP

Limits output to only the .TRAN timestep intervals for post-analysis tools.

### Syntax

```
.OPTION INTERP=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 (engineering notation)  
Value if option name is specified without a corresponding value: 1

### Description

Use to limit output for post-analysis tools to only the .TRAN timestep intervals for some post-analysis tools. This option can be used to reduce the size of the post-processing output. By default, HSPICE outputs data at internal timepoints. In some cases, INTERP produces a much larger design .tr# file, especially for smaller timesteps, and it also leads to longer runtime.

**Note:** Since HSPICE uses the post-processing output to compute the .MEASURE command results, interpolation errors result if you use the INTERP option and your netlist also contains .MEASURE commands. Using the INTERP option with .MEASURE commands is not recommended.

When you run data-driven transient analysis (.TRAN DATA) in an optimization routine, HSPICE forces INTERP=1. All measurement results are at the time points specified in the data-driven sweep. To measure only at converged internal timesteps (for example, to calculate the AVG or RMS), set ITRPRT=1.

### See Also

[.OPTION ITRPRT](#)  
[.TRAN](#)

## **.OPTION IPROP**

Controls whether to treat all of the circuit information as IP protected.

### **Syntax**

```
.OPTION IPROP 0|1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use to control whether to treat all of the circuit information as IP protected and not output this information during simulation.

- 0= Output information (IP not protected)
- 1=Do not output information (IP protected)



## **.OPTION ITL1**

Specifies the maximum DC iteration limit.

### **Syntax**

```
.OPTION ITL1=n
```

### **Description**

Use this option to specify the maximum DC iteration limit. Increasing this value rarely improves convergence in small circuits. Values as high as 400 have resulted in convergence for some large circuits with feedback (such as operational amplifiers and sense amplifiers). However, most models do not require more than 100 iterations to converge.

### **See Also**

[.DC](#)

## **.OPTION ITL2**

Specifies the iteration limit for the DC transfer curve.

### **Syntax**

```
.OPTION ITL2=n
```

### **Description**

Use this option to specify the iteration limit for the DC transfer curve. Increasing this limit improves convergence only for very large circuits.

### **See Also**

[.DC](#)

## **.OPTION ITL3**

Specifies minimum timestep in timestep algorithms for transient analysis.

### **Syntax**

```
.OPTION ITL3=x
```

### **Description**

Use this option to specify the minimum timestep in timestep algorithms for transient analysis. `ITL3` is the minimum number of iterations required to obtain convergence. If the number of iterations is less than `ITL3`, the internal timestep (delta) doubles.

Use this option to decrease simulation times in circuits where the nodes are stable most of the time (such as digital circuits). If the number of iterations is greater than `IMIN`, the timestep stays the same unless the timestep exceeds the `IMAX` option. `ITL3` is the same as `IMIN`.

### **See Also**

[.OPTION IMAX](#)  
[.OPTION IMIN](#)

---

## **.OPTION ITL4**

Specifies maximum timestep in timestep algorithms for transient analysis in HSPICE/HSPICE RF.

### **Syntax**

```
.OPTION ITL4=x
```

**Default**    8

### **Description**

Use this option to specify the maximum timestep in timestep algorithms for transient analysis. `ITL4` sets the maximum iterations to obtain a convergent solution at a timepoint. If the number of iterations needed is greater than `ITL4`, the internal timestep (delta) decreases by a factor equal to the `FT` transient control option. HSPICE uses the new timestep to calculate a new solution. `ITL4` also works with the `IMIN` transient control option. For HSPICE, `ITL4` is the same as `IMAX`.

### **See Also**

- [.OPTION FT](#)
- [.OPTION IMAX](#)
- [.OPTION IMIN](#)

## **.OPTION ITL5**

Sets an iteration limit for transient analysis.

### **Syntax**

```
.OPTION ITL5=x
```

**Default** 0 (infinite number of iterations)

### **Description**

Use this option to set an iteration limit for a transient analysis. If a circuit uses more than `ITL5` iterations, the program prints all results up to that point.

## **.OPTION ITLPTRAN**

Controls iteration limit used in the final try of the pseudo-transient method.

### **Syntax**

```
.OPTION ITLPTRAN=x
```

**Default**    30

### **Description**

Use this option to control the iteration limit used in the final try of the pseudo-transient method in OP or DC analysis. If a simulation fails in the final try of the pseudo-transient method, provide a higher value.

### **See Also**

[.DC](#)

[.OP](#)

## **.OPTION ITLPZ**

Sets the iteration limit for pole/zero analysis.

### **Syntax**

```
.OPTION ITLPZ=x
```

**Default** 100

### **Description**

Use this option to set the iteration limit for pole/zero analysis.

### **See Also**

- [.OPTION CSCAL](#)
- [.OPTION GSCAL](#)
- [.PZ](#)
- [.OPTION FMAX](#)

---

## .OPTION ITRPRT

Enables printing of output variables at their internal time points.

### Syntax

```
.OPTION ITRPRT 0|1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### Description

Use this option to enable printing of output variables at their internal time points.

When set to 1, HSPICE prints output variables at their internal transient simulation time points. In addition, if you use the `-html` option when invoking HSPICE, then HSPICE prints the values to a separate file (*\*.printtr0*).



## .OPTION IVTH

Invokes a constant-current threshold voltage probing and characterization function for BSIM4 models.

### Syntax

```
.OPTION IVTH=val | IVTHN=val | IVTHP=val
```

### Description

Specifies the *ivth* constant drain terminal current density, to be multiplied by the ratio of transistor width (W) and length (L). The value must be greater than zero to enable the function; the *IVTH* option should always be set to a positive value for both PMOS and NMOS.

*.OPTION IVTH* has been enhanced to support HSPICE BSIM4 (level 54), BSIMSOI4.x (level 70) and PSP (level 69). *.OPTION IVTHN* and *IVTHP* support NMOS and PMOS, respectively.

**Note:** The *val* should be a constant.

In OP analysis, a constant current based *vth* is reported in the OP output. In addition, the element region operation check and *Vod* output are based on the new *vth*. During transient or DC analysis, a template output of LX142 accesses the new *vth* value. LX142(m\*) or *ivth(m\*)* could be used for the new *vth* output. This methodology is based on the monotony *Id/Vgs* curve.

## .OPTION KCLTEST

Activates the KCL (Kirchhoff's Current Law) test.

### Syntax

```
.OPTION KCLTEST=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### Description

Use this option to activate the KCL test. This increases simulation time, especially for large circuits, but checks the solution with a high degree of accuracy.

If you set this value to 1, HSPICE sets these options:

- Sets RELMOS and ABSMOS options to 0 (off).
- Sets ABSI to  $1e-6$  A.
- Sets RELI to  $1e-6$ .

To satisfy the KCL test, each node must satisfy this condition:

$$|\sum i_b| < RELI \cdot \sum |i_b| + ABSI$$

In this equation, the  $i_b$ s are the node currents.

### See Also

[.OPTION ABSI](#)  
[.OPTION ABSMOS](#)  
[.OPTION RELI](#)  
[.OPTION RELMOS](#)

## **.OPTION KLIM**

Sets the minimum mutual inductance.

### **Syntax**

```
.OPTION KLIM=x
```

### **Description**

Use this option to set the minimum mutual inductance below which automatic second-order mutual inductance calculation no longer proceeds. `KLIM` is unitless (analogous to coupling strength, specified in the K-element). Typical `KLIM` values are between .5 and 0.0.

---

## .OPTION LA\_FREQ

Specifies the upper frequency for which accuracy must be preserved.

### Syntax

```
.OPTION LA_FREQ=value
```

**Default** 1GHz

### Description

Use this option to specify the upper frequency for which accuracy must be preserved.

The *value* parameter specifies the upper frequency for which the PACT algorithm must preserve accuracy. If *value* is 0, the algorithm drops all capacitors because only DC is of interest.

The maximum frequency required for accurate reduction depends on both the technology of the circuit and the time scale of interest. In general, the faster the circuit, the higher the maximum frequency.

For additional information, see “[Linear Acceleration](#)” in the *HSPICE User Guide: Simulation and Analysis*.

### See Also

[.OPTION SIM\\_LA](#)

[.OPTION LA\\_TIME](#)

## **.OPTION LA\_MAXR**

Specifies the maximum resistance for linear matrix reduction.

### **Syntax**

```
.OPTION LA_MAXR=value
```

**Default**    1e15 ohms

### **Description**

Use this option to specify the maximum resistance for linear matrix reduction.

The *value* parameter specifies the maximum resistance preserved in the reduction. The linear matrix reduction process assumes that any resistor greater than *value* has an infinite resistance and drops the resistor after reduction is completed.

For additional information, see “[Linear Acceleration](#)” in the *HSPICE User Guide: Simulation and Analysis*.

### **See Also**

[.OPTION SIM\\_LA](#)

---

## **.OPTION LA\_MINC**

Specifies the minimum capacitance for linear matrix reduction.

### **Syntax**

```
.OPTION LA_MINC=val
```

**Default** 1e-16 farads

### **Description**

Removes any capacitor in the original netlist less than the value of LA\_MINC prior to reduction. For additional information, see “[Linear Acceleration](#)” in the *HSPICE User Guide: Simulation and Analysis*.

### **See Also**

- [.OPTION SIM\\_LA](#)
- [.OPTION LA\\_FREQ](#)
- [.OPTION LA\\_MAXR](#)
- [.OPTION LA\\_TIME](#)
- [.OPTION LA\\_TOL](#)

---

## .OPTION LA\_TIME

Specifies the minimum time for which accuracy must be preserved.

### Syntax

```
.OPTION LA_TIME=value
```

### Description

Use this option to specify the minimum time for which accuracy must be preserved. The *value* parameter specifies the minimum switching time for which the PACT algorithm preserves accuracy.

Waveforms that occur more rapidly than the minimum switching time are not accurately represented.

This option is simply an alternative to .OPTION LA\_FREQ. The default is equivalent to setting LA\_FREQ=1GHz.

**Note:** Higher frequencies (smaller times) increase accuracy, but only up to the minimum time step used in HSPICE.

For additional information, see “[Linear Acceleration](#)” in the *HSPICE User Guide: Simulation and Analysis*.

### Examples

For a circuit having a typical rise time of 1ns, either set the maximum frequency to 1 GHz, or set the minimum switching time to 1ns:

```
.OPTION LA_FREQ=1GHz  
-or-  
.OPTION LA_TIME=1ns
```

However, if spikes occur in 0.1ns, HSPICE does not accurately simulate them. To capture the behavior of the spikes, use:

```
.OPTION LA_FREQ=10GHz  
-or-  
.OPTION LA_TIME=0.1ns
```

### See Also

[.OPTION SIM\\_LA](#)  
[.OPTION LA\\_FREQ](#)

---

## **.OPTION LA\_TOL**

Specifies the error tolerance for the PACT algorithm.

### **Syntax**

```
.OPTION LA_TOL=value
```

**Default** 0.05

### **Description**

Use this option to specify the error tolerance for the PACT algorithm.

The *value* parameter must specify a real number between 0.0 and 1.0.

For additional information, see “[Linear Acceleration](#)” in the *HSPICE User Guide: Simulation and Analysis*.

### **See Also**

[.OPTION SIM\\_LA](#)



## **.OPTION LENNAM**

Specifies maximum name length for printing operating point analysis results.

### **Syntax**

```
.OPTION LENNAM=x
```

**Default** 16 (characters)

### **Description**

Use this option to specify the maximum length of names in the printout of operating point analysis results. The maximum value is 1024. .OPTION LENNAME prints the full related name of the transistor in the noise tables and OP tables.

### **Examples**

```
...  
.OPTIONS POST=1 LENNAM=40  
...
```

## **.OPTION LIMPTS**

Specifies the number of points to print in AC analysis.

### **Syntax**

```
.OPTION LIMPTS=x
```

**Default**    2001

### **Description**

Use this option to specify the number of points to print or plot in AC analysis. You do not need to set `LIMPTS` for a DC or transient analysis. HSPICE spools the output file to disk.

### **See Also**

- [.AC](#)
- [.DC](#)
- [.TRAN](#)

## **.OPTION LIMTIM**

Specifies the amount of CPU time reserved to generate prints.

### **Syntax**

```
.OPTION LIMTIM=x
```

**Default** 2 (seconds)

### **Description**

Use this option to specify the amount of CPU time reserved to generate prints and plots if a CPU time limit (`CPTIME=x`) terminates simulation. Default is normally sufficient for short printouts.

### **See Also**

[.OPTION CPTIME](#)

---

## .OPTION LISLVL

Controls whether or not HSPICE suppresses the circuit number to circuit hierarchy information in the listing file.

### Syntax

LISLVL=0 | 1

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### Description

LISLVL=0 prints the circuit name directory information in the *.lis* file. If the value is 1, the circuit number and circuit hierarchy information is not output to the *.lis* file.

---

## .OPTION LIS\_NEW

Enables streamlining improvements to the *\*.lis* file.

### Syntax

```
.OPTION LIS_NEW=0 | 1
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### Description

Use `.OPTION LIS_NEW` to activate several streamlining improvements to the *\*.lis* file as noted below. A value of 0 disables the following functions. A value of 1 enables the following:

- Moves `.PRINT` data and `.NOISE` analysis data to separate files,
- Suppresses operating point node voltage table that exists in the *\*.ic#* file.
- Prints loading information for input files.
- Invokes console printing of simulation progress percentage.
- Adds a convergence status update to *\*.lis*.
- Increments every 10% of analysis update to *\*.lis*.
- Reports analysis output file with analysis-specific format.
- Prints Improved format of circuit statistics information.
- Operating point analysis information is separated to file if `.OP` is used in netlist (`LIS_NEW=1` automatically sets `.OPTION OPFILE=1`).
- Model related information is suppressed (`lis_new=1` automatically sets `.OPTION NOMOD=1`). Circuit hierarchy to number mapping information is not printed to *\*.lis* file and `.OPTION LISLVL` becomes nullified with usage of `LIS_NEW=1`.

### See Also

[.OPTION LIST](#)  
[.LIB](#)  
[.NOISE](#)  
[.OPTION LISLVL](#)  
[.OPTION NOMOD.OPTION OPFILE](#)  
[.OP](#)

---

## .OPTION LIST

Prints a list of netlist elements, node connections, and values for components, voltage and current sources, parameters, and more.

### Syntax

```
.OPTION LIST=0 | 1 | 2 | 3
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### Description

Use this option as follows:

Value	Alias	Description
0	NONE	None of below supported
1	ALL	Print circuit element summary table and parameter definitions
2	ELEMENT	Print circuit element summary table only
3	PARAMETER	Print circuit parameter definitions only

The `LIST` option also prints effective sizes of elements and key values.

**Note:** This option is suppressed by the `BRIEF` option.

### See Also

- [.OPTION BRIEF](#)
- [.OPTION LIS\\_NEW](#)
- [.OPTION UNWRAP](#)
- [.OPTION VFLOOR](#)

## **.OPTION LOADHB**

Loads state variable information from a specified file.

### **Syntax**

```
.OPTION LOADHB='filename'
```

### **Description**

Use this option to load the state variable information contained in the specified file. These values are used to initialize the HB simulation.

### **See Also**

[.HB](#)

[.OPTION SAVEHB](#)

## **.OPTION LOADSNINIT**

Loads the operating point saved at the end of Shooting Newton analysis initialization.

### **Syntax**

```
.OPTION LOADSNINIT="filename"
```

### **Description**

Use this option to load the operating point file saved at the end of SN initialization, which is used as initial conditions for the Shooting-Newton method.



---

## .OPTION LSCAL

Sets the inductance scale for Pole/Zero analysis.

### Syntax

.OPTION LSCAL=x

**Default** 1e+6

### Description

Use this option to set the inductance scale for Pole/Zero analysis. HSPICE multiplies inductance by LSCAL.

**Note:** Scale factors must satisfy the following relations:  $GSCAL = CSCAL \cdot FSCAL$

$$GSCAL = \frac{1}{LSCAL \cdot FSCAL}$$

If you change scale factors, you might need to modify the initial Muller points, (X0R, X0I), (X1R, X1I) and (X2R, X2I), even though HSPICE internally multiplies the initial values by (1.0e-9/GSCAL).

The three complex starting-trial points, in the Muller (x1R,X1I) algorithm for pole/zero analysis are listed below with their defaults. HSPICE multiplies these initial points, and FMAX, by FSCAL.

Starting-Trial Points	Defaults	
.OPTION (X0R,X0I)	X0R=-1.23456e6	X0I=0.0
.OPTION (X1R,X1I)	X1R=1.23456e5	X1I=0.0
.OPTION (X2R,X2I)	X2R=+1.23456e6	X2I=0.0

### See Also

[.OPTION CSCAL](#)  
[.OPTION FMAX](#)  
[.OPTION FSCAL](#)  
[.OPTION GSCAL](#)  
[.OPTION ITLPZ](#)  
[.OPTION PZABS](#)

### Chapter 3: HSPICE and RF Netlist Simulation Control Options

.OPTION LSCAL

.OPTION PZTOL

.OPTION RITOL

.OPTION (X0R,X0I)

.OPTION (X1R,X1I)

.OPTION (X2R,X2I)

.PZ

---

## .OPTION LVLTIM

Selects the timestep algorithm for transient analysis.

### Syntax

```
.OPTION LVLTIM= [1 | 2 | 3] | 4]
```

**Default** 1

### Description

Use this option, (levels 1-3, only) to select the timestep algorithm for transient analysis.

- LVLTIM=1 (default) uses the DVDT timestep control algorithm.
- LVLTIM=2 uses the local truncation error (LTE) timestep control method. You can apply LVLTIM=2 to the TRAP method.
- LVLTIM=3 uses the DVDT timestep control method with timestep reversal.
- LVLTIM=4 is invalid if set by user; it is invoked by the RUNLVL option only to enhance the LTE time step control method used by the latest RUNLVL algorithm.

The local truncation algorithm LVLTIM=2 (LTE) provides a higher degree of accuracy than LVLTIM=1 or 3 (DVDT). If you use this option, errors do not propagate from time point to time point, which can result in an unstable solution.

Selecting the GEAR method changes the value of LVLTIM to 2 automatically. For information on how LVLTIM values impact other options, see [Appendix B, How Options Affect other Options](#).

### See Also

[.OPTION CHGTOL](#)  
[.OPTION DVDT](#)  
[.OPTION FS](#)  
[.OPTION FT](#)  
[.OPTION RELQ](#)

---

## .OPTION MACMOD

Enables HSPICE MOSFET to access the subcircuit definition when there is no matching model reference or enables an HSPICE X-element to access the model reference when there is no matching subcircuit definition.

### Syntax

```
.OPTION MACMOD= [1 | 2 | 3 | 0]
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### Description

The following describes .OPTION MACMOD characteristics:

- When `macmod=1`, HSPICE seeks a subckt definition for the M\*\*\* element if no model reference exists. The desired subckt name must match (case insensitive) the `mname` field in the M\*\*\* instance command. In addition, the number of terminals of the subckt must match with the M\*\*\* element referencing it; otherwise HSPICE exits the simulation based on no definition for the M\*\*\* element. In addition, the M instance can call Verilog-A models when `macmod=1`.
- When `macmod=2`, HSPICE seeks a MOSFET model definition when it cannot find a matching subckt or Verilog-A definition for an X-element. The targeted MOSFET MODEL card could be either an HSPICE built-in MOSFET model or CMI MOSFET model. If the model card that matched the X-element reference name is not a type of MOSFET model, the simulator exits and displays an error message indicating that the reference is not found.
- When `macmod=3`, HSPICE enables the same features as when `macmod=1`. HSPICE seeks a `.subckt` definition for an M-element if there is no matching model reference; HSPICE seeks a `.model` MOSFET definition for an X-element if there is no matching `.subckt` or Verilog-A definition. Usage considerations and limitations remain the same for both features, respectively.

If `.OPTION TMIFLAG=1`, `.OPTION MACMOD` automatically equals 3.

**Note:** When `MACMOD=2` or 3, for the X-element that maps to an M-element, if it has an instance parameter named 'Multi' (case insensitive), then 'Multi' is used as an alias for the 'M' factor (the M multiply parameter).

When `macmod=0`: if there is no `.option MACMOD` in the input files or `MACMOD=0`, then neither of the features is enabled. HSPICE ignores the option `MACMOD` when any value other than `1 | 2 | 3 | 0` is set. The `MACMOD` option is a global option; if there are multiple `MACMOD` options in one simulation, HSPICE uses the value of the last `MACMOD` option.

For examples and detailed discussion, see [MOSFET Element Support Using .OPTION MACMOD](#) in the *HSPICE User Guide: Simulation and Analysis*.

**See Also**

[.OPTION TMIFLAG](#)

## **.OPTION MAXAMP**

Sets the maximum current through voltage-defined branches.

### **Syntax**

```
.OPTION MAXAMP=x
```

### **Description**

Use this option to set the maximum current through voltage-defined branches (voltage sources and inductors). If the current exceeds the `MAXAMP` value, HSPICE reports an error.

---

## .OPTION MAXORD

Specifies the maximum order of integration for the GEAR method.

### Syntax

```
.OPTION MAXORD= [1 | 2 | 3]
```

### Description

Use this option to specify the maximum order of integration for the GEAR method. When the GEAR method is used, based on the circuit type, HSPICE/HSPICE RF automatically switches the GEAR order on the fly. If this option is not specifically set, HSPICE automatically selects the BDF or GEAR integration method based on circuit type when METHOD=GEAR.

The value of the parameter can be either 1, 2, or 3:

- MAXORD=1 selects the first-order GEAR (Backward-Euler) integration (and prohibits GEAR from switching to BDF).
- MAXORD=2 selects the second-order GEAR (Gear-2), which is more stable and accurate than MAXORD=1.
- MAXORD=3 selects the third-order or high GEAR (Gear-3), which is most accurate, since it uses 3 previous time points to estimate the next time point.

### Examples

This example selects the Backward-Euler integration method.

```
.OPTION MAXORD=1 METHOD=GEAR
```

### See Also

[.OPTION METHOD](#)  
[.OPTION RUNLVL \(HSPICE\)](#)

## **.OPTION MAXWARNS**

Specifies maximum number of safe operating area (SOA) warning messages.

### **Syntax**

`.OPTION MAXWARNS=n`

**Default** 5

### **Description**

Use this option to specify the maximum number of SOA warning messages when terminal voltages of a device (MOSFET, BJT, Diode, Resistor, Capacitor etc...) exceed a safe operating area. This option is used with `.OPTION WARN`.

### **See Also**

[.OPTION WARN](#)

[Safe Operating Area \(SOA\) Warnings](#)



## **.OPTION MBYPASS**

Computes the default value of the `BYTOL` control option.

### **Syntax**

```
.OPTION MBYPASS=x
```

### **Description**

Use this option to calculate the default value of the `BYTOL` control option:

Also multiplies the `RELV` voltage tolerance. Set `MBYPASS` to about 0.1 for precision analog circuits.

- Default is 1 for `DVDT=0, 1, 2, or 3`.
- Default is 2 for `DVDT=4`.

### **See Also**

[.OPTION BYTOL](#)  
[.OPTION DVDT](#)  
[.OPTION RELV](#)

## .OPTION MCBRIEF

Controls how HSPICE outputs Monte Carlo parameters.

### Syntax

```
.OPTION MCBRIEF=0 | 1 | 2 | 3
```

**Default** 0

### Description

Use this option to control how HSPICE outputs Monte Carlo parameters:

- MCBRIEF=0: Outputs all Monte Carlo parameters
- MCBRIEF=1: Suppresses Monte Carlo parameters in *\*.mt#* and *\*.lis* files; also suppresses generation of *\*.mc?#*, *\*.mpp#*, *\*.annotate*, and *\*.corner* files.
- MCBRIEF=2: Outputs the Monte Carlo parameters into a *.lis* file only.
- MCBRIEF=3: Outputs the Monte Carlo parameters into the measure files only.

Note that this option only works for parameters defined in a netlist, and *not* for measurement results.

---

## .OPTION MEASDGT

Formats the .MEASURE command output of significant digits in both the listing file and the .MEASURE output files.

### Syntax

```
.OPTION MEASDGT=x
```

**Default** 4

### Description

Use this option to format the .MEASURE command output's significant digits in both the listing file and the .MEASURE output files (*.ma0*, *.mt0*, *.ms0*, and so on).

The value of *x* is typically between 1 and 7 significant digits, although you can set it as high as 10.

Use MEASDGT with .OPTION INGOLD=*x* to control the output data format.

### Examples

For example, if MEASDGT=5, then .MEASURE displays numbers as:

- Five decimal digits for numbers in scientific notation.
- Five digits to the right of the decimal for numbers between 0.1 and 999.

In the listing (*.lis*) file, all .MEASURE output values are in scientific notation so .OPTION MEASDGT=5 results in five decimal digits.

### See Also

[.OPTION INGOLD](#)  
[.MEASURE \(or\) .MEAS](#)

## **.OPTION MEASFAIL**

Specifies where to print the failed measurement output.

### **Syntax**

```
.OPTION MEASFAIL=0 | 1
```

**Default** 1

### **Description**

Use this option to specify where to print the failed measurement output. You can assign this option the following values:

- MEASFAIL=0, outputs “0” into the *.mt#*, *.ms#*, or *.ma#* file, and prints “failed” in the *.lis* file.
- MEASFAIL=1, prints “failed” in the *.mt#*, *.ms#*, or *.ma#* file, and in the *.lis* file.

### **See Also**

[.MEASURE \(or\) .MEAS](#)

## **.OPTION MEASFILE**

Controls whether measure information outputs to single or multiple files when an `.ALTER` command is present in the netlist.

### **Syntax**

```
.OPTION MEASFILE=0 | 1
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to control whether the measure information outputs to a single or multiple files when an `.ALTER` command is present in the netlist. You can assign this option the following values:

- `MEASFILE=0`, outputs measure information to several files.
- `MEASFILE=1`, outputs measure information to a single file.

### **See Also**

[.ALTER](#)  
[.MEASURE \(or\) .MEAS](#)

---

## .OPTION MEASFORM

Enables writing of measurement output files to Excel or HSIM formats, as well as the traditional HSPICE *\*.mt#* format.

### Syntax

```
.OPTION MEASFORM=0 | 1 | 2 | 3
```

**Default** 0

### Description

This option allows specification of file formats other than the traditional HSPICE *\*.mt#*, *\*.ms#*, and *\*.ma#* measure output files to include Excel or HSIM file formats.

- 0: Writes measure file in traditional default HSPICE output style. (Example 1)
- 1: Writes space-separated style which can be imported as data into Excel and Microsoft products (requires manual steps in Excel). (Example 2)
- 2: Writes the HSIM style in *name=value* format. Easy to read, but difficult to import into standard post-processing tools. (Does not work for *\*.mc?#* files [see value 3 below] and defaults to HSPICE default output style). (Example 3)
- 3: Writes the comma separated style with suffix *\*.csv* and this format includes *\*.m?#* and *\*.mc?#* files. This style and suffix is understood by Windows to be an Excel file and can be opened directly in Excel by double-clicking the file name. (Example 4)

### Examples

Results Example 1: Default (Traditional) Measure Format (`.option measform=0`)

```
.TITLE '***inverter circuit***'
delayf      delayr      delay      temper      alter#
9.187e-10    5.487e-10    7.337e-10 -25.0000    1.0000
```

Results Example 2: Excel Format (`.option measform=1`)

```
.TITLE '***inverter circuit***'
delayf      delayr      delay      temper      alter#
9.187e-10    5.487e-10    7.337e-10 -25.0000    1.0000
```

Results Example 3: HSIM Format (.option measform=2)

```
.TITLE '***inverter circuit***'
delayf = 9.187e-10
delayr = 5.487e-10
delay = 7.337e-10
temper = -25.0000
alter# = 1.0000
```

Results Example 4: CSV-Excel Format (.option measform=3)

\*\*\* File name opampmc.ms0\_D.csv\*\*\*

\$DATA1 SOURCE='HSPICE' VERSION='D-2010.12 32-BIT'							
.TITLE '***inverter circuit***'							
index	systoffset1	systoffset2	systoffset3	systoffset4	leakpwr	temper	alter#
1	1.40E-03	1.09E-03	9.05E-04	7.71E-04	2.44E-05	25	1
2	1.25E-03	1.01E-03	8.55E-04	7.32E-04	2.53E-05	25	1
3	7.50E-04	7.56E-04	6.72E-04	5.87E-04	2.62E-05	25	1
4	4.48E-03	2.60E-03	1.90E-03	1.52E-03	2.43E-05	25	1
5	2.58E-03	1.68E-03	1.31E-03	1.08E-03	2.56E-05	25	1

See Also

[.OPTION INGOLD](#)  
[.MEASURE \(or\) .MEAS](#)

---

## .OPTION MEASOUT

Outputs `.MEASURE` command values and sweep parameters into an ASCII file.

### Syntax

```
.OPTION MEASOUT=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### Description

Use this option to output `.MEASURE` command values and sweep parameters into an ASCII file. Post-analysis processing (WaveView or other analysis tools) uses this `<design>.mt#` file, where `#` increments for each `.TEMP` or `.ALTER` block.

For example, for a parameter sweep of an output load, which measures the delay, the `.mt#` file contains data for a delay-versus-fanout plot. You can set this option to 0 (off) in the `hspice.ini` file.

### See Also

- [.ALTER](#)
- [.MEASURE \(or\) .MEAS](#)
- [.TEMP \(or\) .TEMPERATURE](#)



---

## .OPTION MESSAGE\_LIMIT

Limits how many times a certain type warning can appear in the output listing based on the message index.

### Syntax

```
.OPTION MESSAGE_LIMIT 'message_index:number'
```

---

Argument	Description
message_index	Specifies the message index linked below
number	Specifies the limiting number of displays of the message

---

### Description

Use this option to set the number of display times for a certain warning type based on its message index number.

- The `message_index` parameter specifies the message index listed in the [Warning Message Index \[10001-10076\]](#) or [Error Message Index \[20001-20024\]](#), located in the *HSPICE User Guide: Simulation and Analysis*, Chapter 34, [Warning/Error Messages](#).
- The `number` parameter specifies the display times.

`.OPTION MESSAGE_LIMIT` has a higher priority than `OPTION WARNLIMIT` and increases the coverage of types messages to be limited.

### See Also

[.OPTION WARNLIMIT](#) (or) [.OPTION WARNLIM](#)  
[.OPTION STRICT\\_CHECK](#)

---

## .OPTION METHOD

Sets the numerical integration method for a transient analysis for HSPICE/  
HSPICE RF.

### Syntax

```
.OPTION METHOD=GEAR | TRAP [PURETP] | BDF
```

**Default** TRAP

### Description

Use this option to set the numerical integration method for a transient analysis.

- TRAP selects trapezoidal rule integration. This method inserts occasional Backward-Euler timesteps to avoid numerical oscillations. You can use the PURETP option to turn this oscillation damping feature off.
- TRAP PURETP selects pure trapezoidal rule integration. This method is recommended for high-Q LC oscillators and crystal oscillators.
- GEAR selects BDF integration or GEAR integration based on circuit type.
- GEAR MAXORD=2 | 3 selects GEAR integration.
- GEAR MAXORD=1 prohibits GEAR from selecting BDF.
- GEAR MU=0 selects Backward-Euler integration.
- BDF selects the high order integration method based on the backward differentiation formulation.

**Note:** To change LVLTIM from 2 to 1 or 3, set LVLTIM=1 or 3 after the METHOD=GEAR option. This overrides METHOD=GEAR, which sets LVLTIM=2.

TRAP (trapezoidal) integration usually reduces program execution time with more accurate results. However, this method can introduce an apparent oscillation on printed or plotted nodes, which might not result from circuit behavior. To test this, run a transient analysis by using a small timestep. If oscillation disappears, the cause is the trapezoidal method.

The GEAR method is a filter, removing oscillations that occur in the trapezoidal method. Highly non-linear circuits (such as operational amplifiers) can require very long execution times when you use the GEAR method. Circuits that do not converge in trapezoidal integration, often converge if you use GEAR.

The BDF method is a high order integration method based on the backward differentiation formulae. Two tolerance options are available to the user for the

BDF method: `.OPTIONS BDFRTOL` (relative) and `BDFATOL` (absolute); each has a default of 1e-3. BDF can provide a speed enhancement to mixed-signal circuit simulation, especially for circuits with a large number of devices. The BDF method currently provides no advantage for use with small circuits in standard cell characterization. The BDF supported models/devices/elements and limitations are listed. `METHOD=BDF` supports the following:

- Bulk MOSFET, levels 1-54
- SOI MOSFET, levels 57, 70
- BJT, levels 1, 2, 3
- Diodes, all
- Resistors, all
- Capacitors (excludes DC block)
- Independent sources: V and I
- Dependent sources: E/F/G/H
- L (excludes AC choke)
- K (excludes magnetic core, ideal transformer)
- Signal integrity elements: B (IBIS buffer)/S/ W/ T

**Note:** BDF issues a warning in the `.lis` file if it encounters an unsupported model. The message is similar to: `WARNING!!!, netlist contains 'unsupported models', HSP-BDF is disabled.`

When `RUNLVL` is turned off (`=0`), `method=GEAR` sets `bypass=0`; the user can reset `bypass` value by using `.option bypass=value`. Also, when `RUNLVL` is turned off, there is an order dependency with `GEAR` and `ACCURATE` options; if `method=GEAR` is set after the `ACCURATE` option, then the `ACCURATE` option does not take effect; if `method=GEAR` is set before the `ACCURATE` option, then both `GEAR` and `ACCURATE` take effect.

If `GEAR` is used with `RUNLVL`, then `GEAR` only determines the numeric integration method; anything else is controlled by `RUNLVL`; there is no order dependency with `RUNLVL` and `GEAR`. Since there is no order dependency with `RUNLVL` and `GEAR`, or `RUNLVL` and `ACCURATE`, then:

is equivalent to

To see how use of the `GEAR` method impacts the value settings of `ACCURATE` and other options, see [Appendix B, How Options Affect other Options](#).

#### Examples

Example 1 sets pure trapezoidal method integration. No Gear-2 or Backward-Euler is mixed in. Use this setting when you simulate harmonic oscillators.

##### *Example 1*

```
.option method=trap puretp
```

Example 2 sets pure Backward-Euler integration.

##### *Example 2*

```
.option method=gear maxord=1
```

Example 3 sets pure Gear-2 integration.

##### *Example 3*

```
.option method=gear
```

Example 4 sets the higher order backward differentiation formulation integration for supported models.

##### *Example 4*

```
.option method=bdf
```

#### See Also

- [.OPTION ACCURATE](#)
- [.OPTION LVLTIM](#)
- [.OPTION MAXORD](#)
- [.OPTION MTTRESH](#)
- [.OPTION PURETP](#)
- [.OPTION MU](#)
- [.OPTION RUNLVL](#)
- [.OPTION BDFATOL](#)
- [.OPTION BDFRTOL](#)

## .OPTION MODMONTE

Controls how random values are assigned to parameters with Monte Carlo definitions.

### Syntax

```
.OPTION MODMONTE=0 | 1
```

**Default** Value if option is not specified in the netlist: 0  
 Value if option name is specified without a corresponding value: 1

### Description

Ordinarily, the assignment of a random value is only done once, then used several times. The exception to this rule is for model parameters. Since a model definition is only done once, the behavior described above would assign the same parameter value to all devices referencing that model. To overcome this, `.OPTION MODMONTE` lets you decide if all instances of a device should get the same or unique model parameters. Use this option to control how random values are assigned to parameters with Monte Carlo definitions.

- If `MODMONTE=1`, then within a single simulation run, each device that shares the same model card and is in the same Monte Carlo index receives a different random value for parameters that have a Monte Carlo definition.
- If `MODMONTE=0`, then within a single simulation run, each device that shares the same model card and is in the same Monte Carlo index receives the same random value for its parameters that have a Monte Carlo definition.

### Examples

In the following example, transistors M1 through M3 have the same random `vto` model parameter for each of the five Monte Carlo runs through the use of the `MODMONTE` option.

```
...
.option MODMONTE=0 $$ MODMONTE defaults to 0;OK to omit this line.
.param vto_par=agauss(0.4, 0.1, 3)
.model mname nmos level=53 vto=vto_par version=3.22
M1 11 21 31 41 mname W=20u L=0.3u
M2 12 22 32 42 mname W=20u L=0.3u
M3 13 23 33 43 mname W=20u L=0.3u
...
.dc v1 0 vdd 0.1 sweep monte=5
.end
```

### Chapter 3: HSPICE and RF Netlist Simulation Control Options

#### .OPTION MODMONTE

In Example 2, transistors M1 through M3 have different values of the `vto` model parameter for each of the Monte Carlo runs by the means of setting `.option MODMONTE=1`.

#### *Example 1*

```
...  
.option MODMONTE=1  
.param vto_par=agauss(0.4, 0.1, 3)  
.model mname nmos level=54 vto=vto_par  
M1 11 21 31 41 mname W=20u L=0.3u  
M2 12 22 32 42 mname W=20u L=0.3u  
M3 13 23 33 43 mname W=20u L=0.3u  
...  
.dc v1 0 vdd 0.1 sweep monte=5  
.end
```

#### **See Also**

[.MODEL](#)

---

## .OPTION MODPRT

Invokes model-preprocessing and parameter flattening.

### Syntax

```
.OPTION MODPRT= [0 | 1]
```

**Default** 0 (Off)

### Description

When `.OPTION MODPRT=1` the following takes place:

- Model information is written to a file called *reduced.models*, readable by standard HSPICE.
- Information for each model occupies a single physical record to facilitate further processing necessary to link the new models to the cell library.
- Comments can appear between the model records (to help tracing).
- Values are printed to 17 digits to simplify validation.
- Fields that are missing on the original model record are not be printed (to avoid warnings and potential errors in HSPICE).
- Fields for which the values equal the default values for the particular level/version are not printed, thus reducing the size of the file; all other fields appearing on the original model card are printed.
- Since each MOSFET has its own model in the *reduced.models* file, the bin designator is replaced by the index of the MOSFET. For example, a model name is `pch_27` because the device belongs to bin 27.
- Since the models are non-binned, the extra fields: `lmin`, `lmax`, `wmin`, `wmax` (and similar fields, if any), are deleted.
- A *reduced.instances* file is generated in cases where additional information is required for the instance. For example, for the "main n1 ..." call inside the subckt, the parameters on the MOSFET need not be the same as what were specified on the subckt invocation. The *reduced.instances* file reports the MOSFET records with the resolved values for the parameters.

See the examples below for additional information.

### Examples

*Example 1* Original Netlist: In the case below, it is assumed that:  
(1) `X_1` and `X_2` use the same bin model card `pch.26`, while there are some different parameters values in model cards (because instance

## Chapter 3: HSPICE and RF Netlist Simulation Control Options

### .OPTION MODPRT

*parameters will affect the model parameters values);  
(2) X\_3 and X\_4 could share the model card with X\_1;  
(3) X\_5 could not share model card with other instance, and it uses the  
pch\_4 model card;*

X_1	1 2 0 0	pch_mac	W=... L=...
X_2	1 2 0 0	nch_mac	W=... L=...
X_3	1 2 0 0	pch_mac	W=... L=...
X_4	1 2 0 0	nch_mac	W=... L=...
X_5	1 2 0 0	pch_mac	W=... L=...

*Example 2 reduced.models output file for Example 1: This file prints unique model cards and adds instance name information on model card name.*

```
.model X_1_pch_26 level = 54 .....  
.model X_2_pch_26 level = 54 .....  
.model X_5_pch_4 level = 54 ... .
```

*Example 3 reduced.instance file: this file connects the model information with instance information as shown below.*

X_1	X_1_pch_26	W = ..... L = .....
X_2	X_2_pch_26	W = ..... L = .....
X_3	X_1_pch_26	W = ..... L = .....
X_4	X_1_pch_26	W = ..... L = .....
X_5	X_5_pch_4	W = ..... L = .....

1. In the *reduced.instance* file, the "." characters are replaced by "\_" in the model names; a model card name X\_1\_pch\_26 includes two parts:

- Instance name (X\_1)
- Bin model name (pch\_26)

the first part is the instance name (X\_1) and the second part is the bin

2. The *reduced.instance* file does not print the d/g/s/b connecting information; the format is:

instance name	model name	solved parameter values
X_1	X_1_pch_26	W = ..... L = .....

3. The *reduced.instance* file contains all the fields as they become resolved inside the macro, not just the ones on the original "X" record.

4. For each model, the information is printed to be a single physical record in *reduced.models* (not continued across multiple records with "+" continuation).



## **.OPTION MONTECON**

Continues a Monte Carlo analysis in HSPICE by retrieving the next random value, even if non-convergence occurs.

### **Syntax**

```
.OPTION MONTECON=0 | 1
```

**Default**    1

### **Description**

Use this option to retrieve the next random value, even if non-convergence occurs. A random value can be too large or too small to cause convergence to fail. Other types of analyses can use this Monte Carlo random value.

---

## **.OPTION MOSRALIFE**

Invokes the MOSRA “lifetime” computation.

### **Syntax**

`.OPTION MOSRALIFE=degradation_type_keyword`

### **Description**

Use this option to compute device lifetime calculation for the degradation type specified.

The option is used in conjunction with `.OPTION DegF=val` when no values are set for either `.OPTION DegFN=val` or `.OPTION DegFP=val`, the designated NMOS's or PMOS's failure criteria for lifetime computation, respectively. The options apply to all MOSFETs. The lifetime value is printed in the RADEG file.

### **See Also**

- [.OPTION DEGF](#)
- [.OPTION DEGFN](#)
- [.OPTION DEGFP](#)
- [.OPTION RADEGFILE](#)
- [.OPTION RADEGOUTPUT](#)

## **.OPTION MOSRASORT**

Enables the descending sort for reliability degradation (RADEG) output.

### **Syntax**

```
.OPTION MOSRASORT=degradation_type_keyword
```

**Default** `delvth0`

### **Description**

Use this option `mosrasort` to enable the descending sort for reliability degradation (RADEG) output.

If the `mosrasort` option is not specified, or the degradation type keyword is not recognized, HSPICE does not do the sorting. (Degradation type keywords are listed in the *HSPICE Application Note: Unified Custom Reliability Modeling API (MOSRA API)*, available by contacting the HSPICE technical support team.)

If you only specify the option `mosrasort`, and do not specify the degradation type keyword, HSPICE sorts RADEG by the `delvth0` keyword.

HSPICE sorts the output in two separate lists, one for NMOS devices, another for PMOS device. HSPICE prints the NMOS device list first, and then the PMOS device list.

### **Examples**

In the following usage, the option does a descending sort for RADEG output on `delvth0`'s value.

```
.option mosrasort=delvth0
```

### **See Also**

[.MOSRA](#)  
[MOSFET Model Reliability Analysis \(MOSRA\)](#)

## **.OPTION MRAAPI**

Loads and links the dynamically linked MOSRA API library.

### **Syntax**

```
.OPTION MRAAPI=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to load and link the compiled MOSRA API *.so* library file to HSPICE during simulation. If this option parameter is set with no value or to 1, then the MOSRA API *.so* library file is loaded as a dynamically-linked object file.

If this option parameter does not exist in the netlist, or is explicitly set to 0, the MOSRA API *.so* library will not be used.

## **.OPTION MRAEXT**

Enables access to MOSRA API extension functions.

### **Syntax**

```
.OPTION MRAEXT 0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to control the access to the MOSRAAPI extension functions. When `MRAEXT=1`, HSPICE can access the extension functions. Details are in the *HSPICE User Guide: Implementing the MOSRA API*. Contact HSPICE Technical Support for more information.

## **.OPTION MRAPAGED**

Enables the MOSRA API to enable two modes of model parameter degradation.

### **Syntax**

```
.OPTION MRAPAGED=0 | 1
```

**Default** 0

### **Description**

If this option parameter does not exist (deemed as default) in the netlist, or is explicitly set to 0, degradation from the MOSRA API model is the parameter value shift with regard to the fresh model,  $\Delta P$ . If this option parameter is set to 1, then the degradation from the MOSRA API model is the degraded model parameter,  $P + \Delta P$ .

- 0:  $\Delta P$  mode
- 1: Degraded model parameter

## **.OPTION MRA00PATH, MRA01PATH, MRA02PATH, MRA03PATH**

These options support file path access in MOSRA API functions.

### **Syntax**

```
.OPTION MRA00PATH = 'file_path1'  
.option MRA01PATH = 'file_path2'  
.option MRA02PATH = 'file_path3'  
.option MRA03PATH = 'file_path4'
```

**Default** NULL for each path

### **Description**

Use these options to enable global string type variables such as user-defined paths. This option is for API model developers to access the MOSRA API functions.

---

## .OPTION MTTHRESH

Reduces the default active device limit for multithreading.

### Syntax

```
.OPTION MTTHRESH=N
```

**Default**    64

### Description

Use `.OPTION MTTHRESH` only for model evaluation threading. For multithreading to be effective in model evaluation, the number of active devices or elements should meet certain requirements.

The condition for model evaluation to be multithreaded is ONE of the following:

- MOSFET  $\geq 64$
- BJT  $\geq 128$
- Diode  $\geq 128$
- G-element  $\geq 128$
- E-element  $\geq 128$
- F-element  $\geq 128$
- H-element  $\geq 128$
- or parameter expressions  $\geq 64$

If the circuit lacks the required number of active devices, HSPICE automatically uses a single thread. You can manually enforce multithreading on model evaluation by using `.OPTION MTTHRESH`. The default `MTTHRESH` value is 64. You can set it to any positive integer number equal to or greater than 2. This option has no effect on matrix solving. `MTTHRESH` must = 2 or more. Otherwise, HSPICE MT defaults to 64.

### Examples

If `MTTHRESH=50`, model evaluation of MOSFETs would be threaded if the number of MOSFETs is greater than 50. Similarly, a diode model evaluation would receive benefit from multithreading if the circuit contains more than 100 (50 x 2) diodes.



## **.OPTION MU**

Defines the integration method coefficient.

### **Syntax**

```
.OPTION MU=x
```

**Default**    0.5

### **Description**

Use this option to define the integration method coefficient. The value range is 0.0 to 0.5. The default integration method is trapezoidal which corresponds to the default coefficient value of 0.5. If the value is set to 0, then the integration method becomes backward-Euler. A value between 0 and 0.5 is a blend of the trapezoidal and backward-Euler integration methods.

### **See Also**

[.OPTION METHOD](#)

---

## **.OPTION NCFILTER**

Filters negative conductance warning messages according to the setting value.

### **Syntax**

```
.OPTION NCFILTER=val
```

**Default** -1e-12

### **Description**

When `.option ncwarn` is set, use this option to filter the negative conductance warning messages according to the setting value. If `gds`, `gm`, `gmbs` < *value*, a warning message is reported. When `ncwarn` is set, this filter is automatically enabled. The legal range of *val* is  $-1e20$  to 0.

### **See Also**

[.OPTION NCWARN](#)

## **.OPTION NCWARN**

Allows turning on a switch to report a warning message for negative conductance on MOSFETs.

### **Syntax**

```
.OPTION NCWARN=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use the option to turn on (`.option NCWARN=1`), printing out of the first occurrence of MOSFET related “negative conductance” in the listing file; if you want to check the entire negative conductance on MOSFETs, use `.option DIAGNOSTIC` to print all these warning messages. `NCWARN=0` (default) turns off all warning messages on negative conductance.

### **See Also**

[.OPTION DIAGNOSTIC](#) (or) [.OPTION DIAGNO](#)  
[.OPTION NCFILTER](#)

---

## **.OPTION NEWTOL**

Calculates one or more iterations past convergence for every calculated DC solution and timepoint circuit solution.

### **Syntax**

```
.OPTION NEWTOL=x
```

### **Description**

Use this option to calculate one or more iterations past convergence for every calculated DC solution and timepoint circuit solution. If you do not set `NEWTOL` after HSPICE determines convergence the convergence routine ends and the next program step begins.

## **.OPTION NODE**

Prints a node cross-reference table.

### **Syntax**

```
.OPTION NODE=x
```

### **Description**

Use this option to print a node cross-reference table. The `BRIEF` option suppresses `NODE`. The table lists each node and all elements connected to it. A code indicates the terminal of each element. A colon (:) separates the code from the element name.

The codes are:

- + — Diode anode
- — Diode cathode
- B — BJT base
- B — MOSFET or JFET bulk
- C — BJT collector
- D — MOSFET or JFET drain
- E — BJT emitter
- G — MOSFET or JFET gate
- S — BJT substrate
- S — MOSFET or JFET source

### **Examples**

This sample part of a cross-reference line indicates that the bulk of M1, the anode of D2 and the base of Q4, all connect to node 1.

### **See Also**

[.OPTION BRIEF](#)

---

## .OPTION NOELCK

Bypasses element checking to reduce preprocessing time for very large files.

### Syntax

```
.OPTION NOELCK 0|1
```

**Default** Value if option is not specified in the netlist: 0

Value if option name is specified without a corresponding value: 1

### Description

Use this option to bypass element checking to reduce preprocessing time for very large files. HSPICE typically checks for duplicate element definitions. If `.option NOELCK` is set (1), HSPICE skips the element checking and the simulation runs even if there is a duplicate element definition. For the duplicate elements, HSPICE uses the last definition it finds.

When `NOELCHK` is not turned on, if HSPICE finds a duplicate element definition, it issues an error and aborts the simulation.

**Note:** Subcircuit redefinition is not supported by this option.

### Examples

In the following netlist:

```
R1 1 2 1k  
R2 2 0 1k  
C1 2 end 1p  
C1 2 0 1n
```

...unless `.option NOELCHK` is set to 1, HSPICE aborts the simulation and issue an error message.

```
**error** attempts to redefine c1 at line xx and line yy
```

## **.OPTION NOISEMINFREQ**

Specifies the minimum frequency of noise analysis in HSPICE/HSPICE RF.

### **Syntax**

```
.OPTION NOISEMINFREQ=x
```

### **Description**

Use this option to specify the minimum frequency of noise analysis. If the frequency of noise analysis is smaller than the minimum frequency, then HSPICE automatically sets the frequency for `NOISEMINFREQ`.

## **.OPTION NOMOD**

Suppresses the printout of model parameters.

### **Syntax**

`.OPTION NOMOD= [0 | 1]`

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to suppress the printout of model parameters.



## **.OPTION NOPIV**

Controls whether HSPICE automatically switches to pivoting matrix factors.

### **Syntax**

```
.OPTION NOPIV=0 | 1
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to prevent HSPICE from automatically switching to pivoting matrix factors if a nodal conductance is less than PIVTOL. NOPIV=1 inhibits pivoting.

### **See Also**

[.OPTION PIVTOL](#)

## **.OPTION NOTOP**

Suppresses topology checks to increase preprocessing speed.

### **Syntax**

```
.OPTION NOTOP=0 | 1
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to suppress topology checks to increase the speed for preprocessing very large files. HSPICE normally checks the netlist topology and reports a warning or error message. The different topologies that HSPICE checks includes inductor/voltage loops, dangling nodes, stacked current sources and current sources in a closed capacitor loop. If you set the `NOTOP` option to 1, these checks will not be performed and there will be no warning or error messages issued for these topologies.

### **Examples**

If you run the following netlist:

```
R1 1 2 1k  
R2 2 0 1k  
C1 2 end 1p
```

...the dangling node check function causes HSPICE to issue a warning in the `.lis` file.

```
only 1 connection at node 0:end ...
```

If `.option NOTOP` is set, the topology check is skipped and you will not get the warning.

## **.OPTION NOWARN**

Suppresses parameter conflict warning messages.

### **Syntax**

```
.OPTION NOWARN=0 | 1
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to suppress all conflicting parameter warning messages, except those generated from commands in `.ALTER` blocks.

`.OPTION WARNLIMIT` can be used to limit the number of a same warning message.

**Note:** This option only suppresses warnings about conflicting parameters, not model-related or other warnings.

### **See Also**

[.ALTER](#)

[.OPTION WARNLIMIT \(or\) .OPTION WARNLIM](#)

---

## **.OPTION NUMDGT**

Controls the listing printout accuracy.

### **Syntax**

```
.OPTION NUMDGT=x
```

### **Description**

Use this option to control the listing printout (*.lis*) accuracy. The value of *x* is typically between 1 and 7, although you can set it as high as 10. This option does not affect the accuracy of the simulation. This option does, however, affect the results files (ASCII and binary) if you use the

`.OPTIONPOST_VERSION=2001` setting. The default setting is 5 digits for results for printout accuracy when using `POST_VERSION=2001`.

### **See Also**

[.OPTION POST\\_VERSION](#)

## **.OPTION NUMERICAL\_DERIVATIVES**

Diagnostic-only option for checking a problem with the device models.

### **Syntax**

```
.OPTION NUMERICAL_DERIVATIVES=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

This option can be used to help diagnose convergence problems or suspected inaccuracies in small-signal analyses such as HBAC, HBNOISE, or PHASENOISE. If a convergence or accuracy problem stems from an inaccuracy in the current or charge derivatives returned by a transistor or diode model, setting this option to 1 will resolve the problem, although with a performance decrease.

If `NUMERICAL_DERIVATIVES=1` resolves the problem, please contact Synopsys support so that the underlying transistor model issue can be resolved.

If you are confident that the models are providing accurate derivatives, do *not* use this option.

---

## **.OPTION NXX**

Stops echoing (printback) of the data file to stdout.

### **Syntax**

```
.OPTION NXX= [0 | 1]
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to terminate echoing (printback) of the data file to stdout until HSPICE finds an `.OPTION BRIEF=0` or the `.END` command. It also resets the `LIST`, `NODE` and `OPTS` options and sets `NOMOD`. When `BRIEF=0`, it enables printback. `NXX` is the same as `BRIEF`.

### **See Also**

- [.OPTION BRIEF](#)
- [.OPTION LIST](#)
- [.OPTION NODE](#)
- [.OPTION OPTS](#)

## **.OPTION OFF**

Initializes terminal voltages to zero for active devices not initialized to other values.

### **Syntax**

```
.OPTION OFF= [0 | 1]
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to initialize terminal voltages to zero if you did not initialize them to other values for all active devices. For example, if you did not initialize both drain and source nodes of a transistor (using `.NODESET`, `.IC` commands, or connecting them to sources), then `OFF` initializes all nodes of the transistor to 0.

HSPICE checks the `OFF` option before element `IC` parameters. If you assigned an element `IC` parameter to a node, simulation initializes the node to the element `IC` parameter value, even if the `OFF` option previously set it to 0.

You can use the `OFF` element parameter to initialize terminal voltages to 0 for specific active devices. Use the `OFF` option to help find exact DC operating-point solutions for large circuits.

### **See Also**

- [.DC](#)
- [.IC](#)
- [.NODESET](#)

---

## .OPTION OPFILE

Outputs the operating point information to a file.

### Syntax

```
.OPTION OPFILE=[0 | 1 | 2]
```

**Default** Value if option is not specified in the netlist: 0

Value if option name is specified without a corresponding value: 1

### Description

Use this option to output the operating point information to a file. When back annotating the operating point information for the Synopsys Galaxy Custom Designer product, use this option =1 in conjunction with `.OPTION SPLIT_DP=1`.

- 0: The operating point information outputs to stdout.
- 1: The operating point information is output to a file named *design.dp#*.
- 2: The operating point information is output to multiple separate *\*.dp#* files.

When used with multiprocessing (`-mp`) `.OPTION OPFILE=1` or `2` is valid (`3` is ignored).

### See Also

[.OP](#)

[.OPTION SPLIT\\_DP](#)



---

## .OPTION OPTCON

Continues running a bisection analysis (with multiple .ALTER commands) even if optimization failed.

### Syntax

```
.OPTION OPTCON=0|1
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### Description

Use this option to override how HSPICE treats bisection measure failure. With this option turned on, instead of issuing an error and exiting the simulation, HSPICE treats a bisection search failure like a measurement failure and completes the simulation, or continues if .ALTER commands are specified.

### Examples

```
.option optcon=1
r1 1 0 2000
v1 1 0 3
.param target=0.5
.param x=opt1(0, 0, 1)
.model opt_model opt method=bisection relout=1e6
relin=0.0005
.meas tran y param = x goal = target
.tran 1.0e-10 1.0e-9 sweep optimize=opt1 results=y
model=opt_model
.alter target=1.5
.param target=1.5
.alter target=0.75
.param target=0.75
.end
```

## Chapter 3: HSPICE and RF Netlist Simulation Control Options

### .OPTION OPTCON

If a bisection search fails because of endpoints having the same sign, for example, screen output might appear as follows:

```
>info: ***** hspice job concluded
  the maximum number of iterations ( 14)was
  exceeded. however, results might be accurate.
x = 3.556e-09
y = 1.7103E+00
>info: ***** hspice job concluded
  **Warning** endpoints have same sign in bisection
x = failed
y = failed
>info: ***** hspice job concluded
Output stored in file => test.lis
```

### See Also

[.ALTER](#)  
[.OPTION MEASFAIL](#)

## **.OPTION OPTLST**

Outputs additional optimization information.

### **Syntax**

```
.OPTION OPTLST=0 | 1 | 2 | 3
```

**Default** 0

### **Description**

Use this option to output additional optimization information:

- OPTLST=0: No information (default).
- OPTLST=1: Prints parameter, Broyden update and bisection results information.
- OPTLST=2: Prints gradient, error, Hessian, and iteration information.
- OPTLST=3: Prints all of the above and Jacobian.

Since the results of each iteration during an optimization do not meet the defined electrical specifications, HSPICE does not allow you to probe the results at each optimization iteration. However, you can use `.OPTION OPTLST=3` to get the useful information about each iteration.

---

## **.OPTION OPTPARHIER**

Specifies scoping rules to options.

### **Syntax**

```
.OPTION OPTPARHIER=[GLOBAL|LOCAL]
```

### **Description**

Use this option to specify scoping rules to options to support local GEOSHRINK and SCALE options within .SUBCKT commands. As shown in the example below, when OPTPARHIER=GLOBAL, SCALE=2u GEOSHRINK=0.8 will be valid in subcircuits.

When OPTPARHIER=LOCAL, SCALE=1e-6 GEOSHRINK=0.9 is valid in subcircuits.

### **Examples**

This example explicitly shows the difference between local and global scoping for using options in subcircuits.

```
.OPTION OPTPARHIER=[global | local]
.OPTION SCALE=2u GEOSHRINK=0.8
.PARAM DefPwid=1u
.SUBCKT Inv a y DefPwid=2u DefNwid=1u
.OPTIONS SCALE=1e-6 GEOSHRINK=0.9
Mp1 MosPinList pMosMod L=1.2u W=DefPwid
Mn1 MosPinList nMosMod L=1.2u W=DefNwid
.ENDS
```

### **See Also**

- [.OPTION GEOSHRINK](#)
- [.OPTION SCALE](#)
- [.SUBCKT](#)

## **.OPTION OPTS**

Prints current settings for all control options.

### **Syntax**

```
.OPTION OPTS
```

### **Description**

Use this option to print the current settings for all control options. If you change any of the default values of the options, the `OPTS` option prints the values that the simulation actually uses. The `BRIEF` option suppresses `OPTS`.

**Note:** All `SIM_LA*` printed settings are shown as `LA_*`.

### **See Also**

[.OPTION BRIEF](#)

---

## **.OPTION PARHIER (or) .OPTION PARHIE**

Specifies scoping rules.

### **Syntax**

```
.OPTION PARHIER= [GLOBAL | LOCAL]
```

### **Description**

Use this option to specify scoping rules. The example explicitly shows the difference between local and global scoping for using parameters in subcircuits.

### **Examples**

```
.OPTION parhier=global | local  
.PARAM DefPwid=1u  
.SUBCKT Inv a y DefPwid=2u DefNwid=1u  
    Mp1 <MosPinList> pMosMod L=1.2u W=DefPwid  
    Mn1 <MosPinList> nMosMod L=1.2u W=DefNwid  
.ENDS
```

## .OPTION PATHNUM

Prints subcircuit path numbers instead of path names; overrides 8-character model name limitation.

### Syntax

.OPTION PATHNUM= [0 | 1 | 2]

**Default** Value if option is not specified in the netlist: 0  
 Value if option name is specified without a corresponding value: 1

### Description

When set to 1, this option prints subcircuit path numbers instead of path names. When set to 2, the complete model name (no truncation) is printed to the \*.lis file; without this setting, model names are limited to eight characters. In addition, a full nodal hierarchy table is printed. For example, the following captab nodal hierarchy appears as:

```
< +xv2i.dac_x[1]= 3.331e-15      xv2i.dac_x[2]= 3.014e-15      xv2i.dac_x[3]= 3.014e-15
< +xv2i.dset   = 1.260e-13      xv2i.ed[0]   = 3.218e-15      xv2i.ed[1]   = 3.751e-15
< +xv2i.ed[2]  = 6.964e-15      xv2i.ed[3]   = 1.066e-14      xv2i.ev0    = 3.801e-15
< +xv2i.ev1    = 5.351e-15      xv2i.ev2    = 5.186e-15      xv2i.k_x[0] = 4.277e-15
< +xv2i.k_x[1] = 5.786e-15      xv2i.kvco[0] = 3.447e-15      xv2i.kvco[1] = 2.197e-15
< +xv2i.n$10444 = 1.858e-15      xv2i.n$10445 = 1.874e-15      xv2i.n2ab   = 9.120e-15
```

---

## .OPTION PCB\_SCALE\_FORMAT

Extends support for using a scaling factor in place of the decimal point for PCB part number formats during case-sensitive simulation.

### Syntax

```
.OPTION PCB_SCALE_FORMAT= [0 | 1]
```

**Default** Value if option is not specified in the netlist: 0

Value if option name is specified without a corresponding value: 1

### Description

Allows both uppercase and lowercase number formats to be supported when case sensitive simulations are run for parts placed on a PCB. This option maintains backward compatibility for those who use a number format that is common for parts placed on a PCB, where the scaling factor is used instead of the decimal point (e.g., 3k3 -> 3.3k).

Setting `.OPTION PCB_SCALE_FORMAT= 1` when case-sensitivity is turned on allows the 3k3 number format to be usable in an expression.

This option adds a new scaling factor, “r” or “R,” which is the multiplying factor 1e0 (often used for resistors).

### Examples

#### *Example 1 Decimal converted to Expression*

```
.option pcb_scale_format=1
.param a='3k3 +2k/2'
```

As seen in the examples below, backward compatibility with the features of HSPICE numbers is maintained (such as optional trailing units and scaling symbols). The examples below show how the values 0 or 1 affect the output.

#### *Example 2 .OPTION PCB\_SCALE\_FORMAT=0:*

```
1) 0u1 / 0U1 / 0u1farads / 0.1u / 0u1farads / 0.1ufarads -> 0.1u
2) 5R6 / 5r6 / 5600m0 / 5600M0 / 5600m -> 5.6
3) 5MEG35 / 5meg35 / 5.35Meg -> 5.35e6
```

#### *Example 3 .OPTION PCB\_SCALE\_FORMAT=1:*

```
a) 0u1 / 0u1farads / 0.1u -> 0.1u
b) 5R6 / 5r6 / 5600m0 / 0M0000056-> 5.6
c) 5MEG35 / 5meg35 -> 5.35e6
```



**See Also**

[.OPTION SI\\_SCALE\\_SYMBOLS](#)

---

## **.OPTION PHASENOISEKRYLOVDIM**

Specifies the dimension of the Krylov subspace that the Krylov solver uses.

### **Syntax**

```
.OPTION PHASENOISEKRYLOVDIM
```

**Default**    500

### **Description**

Specifies the dimension of the Krylov subspace that the Krylov solver uses. This must be an integer greater than zero.

### **See Also**

- [.OPTION BPNMATCHTOL](#)
- [.OPTION PHASENOISEKRYLOVITER \(or\) PHASENOISE\\_KRYLOV\\_ITER](#)
- [.OPTION PHASENOISETOL](#)
- [.OPTION PHNOISELORENTZ](#)

## **.OPTION PHASENOISEKRYLOVITER (or) PHASENOISE\_KRYLOV\_ITER**

Specifies the maximum number of Krylov iterations that the phase noise Krylov solver takes.

### **Syntax**

```
.OPTION PHASENOISEKRYLOVITER | PHASENOISE_KRYLOV_ITER
```

**Default** 1000

### **Description**

Specifies the maximum number of Krylov iterations that the phase noise Krylov solver takes. Analysis stops when the number of iterations reaches this value.

### **See Also**

- [.OPTION BPNMATCHTOL](#)
- [.OPTION PHASENOISEKRYLOVDIM](#)
- [.OPTION PHASENOISETOL](#)
- [.OPTION PHNOISELORENTZ](#)

---

## **.OPTION PHASENOISETOL**

Specifies the error tolerance for the phase noise solver.

### **Syntax**

```
.OPTION PHASENOISETOL
```

**Default**    1e-8

### **Description**

Specifies the error tolerance for the phase noise solver. This must be a real number greater than zero.

### **See Also**

[.OPTION BPNMATCHTOL](#)

[.OPTION PHASENOISEKRYLOVDIM](#)

[.OPTION PHASENOISEKRYLOVITER](#) (or) [PHASENOISE\\_KRYLOV\\_ITER](#)

[.OPTION PHNOISELORENTZ](#)

## **.OPTION PHASETOLI**

For HB output, aids in reporting when magnitude of phase current is very small.

### **Syntax**

`.OPTION PHASETOLI=val`

**Default** 1.e-15

### **Description**

Use this option in a harmonic balance analysis to report the output of the magnitude of a current phasor as zero. If the current phasor is less than the PHASETOLI value, then zero phase is reported. (If the magnitude of a current value is very tiny, the phase does not matter at all.)

### **See Also**

- [.OPTION PHASETOLV](#)
- [.HB](#)
- [.HBAC](#)
- [.HBLIN](#)
- [.HBLSP](#)
- [.HBNOISE](#)
- [.HBOSC](#)
- [.HBXF](#)

## **.OPTION PHASETOLV**

For HB output, aids in reporting when magnitude of phase voltage is very small.

### **Syntax**

`.OPTION PHASETOLV=val`

**Default** 1.e-15

### **Description**

Use this option in a harmonic balance analysis to report the output of the magnitude of a voltage phasor as zero. If the voltage phasor is less than the PHASETOLV value, the phase of that phasor is output as zero. (If the magnitude of a voltage value is very tiny, the phase does not matter at all.)

### **See Also**

- [.OPTION PHASETOLI](#)
- [.HB](#)
- [.HBAC](#)
- [.HBLIN](#)
- [.HBLSP](#)
- [.HBNOISE](#)
- [.HBOSC](#)
- [.HBXF](#)

## **.OPTION PHD**

Facilitates fast OP convergence for BSIM4 testcases.

### **Syntax**

```
.OPTION PHD= [0 | 1]
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

When PHD is set to 1 (ON), this option facilitates fast OP convergence for BSIM4 testcases. The PHD flow may show performance improvement in simulations that require large DC OP convergence iterations. When PHD is on but fails to converge, the simulation exits.

## **.OPTION PHNOISELORENTZ**

Turns on a Lorentzian model for the phase noise analysis.

### **Syntax**

```
.OPTION PHNOISELORENTZ= 0 | 1 | 2
```

**Default** 1

### **Description**

Turns on a Lorentzian model for the phase noise analysis.

- 0: Uses a linear approximation to a lorentzian model and avoids phasenoise values >0dB for low offsets
- 1 (default): Applies a lorentzian model to all noise sources
- 2: Applies a lorentzian model to all non-frequency dependent noise sources

### **See Also**

[.OPTION BPNMATCHTOL](#)

[.OPTION PHASENOISEKRYLOVDIM](#)

[.OPTION PHASENOISEKRYLOVITER \(or\) PHASENOISE\\_KRYLOV\\_ITER](#)

[.OPTION PHASENOISETOL](#)



---

## .OPTION PHNOISEAMP

Allows you to separate amplitude modulation and phase modulation components in a phase noise simulation.

### Syntax

```
.OPTION PHNOISEAMP=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### Description

Use this option to enable HSPICE RF to calculate separate amplitude (am) and phase modulation (pm) components using the output and measure syntax of a .PHASENOISE simulation. A value of 0 sets the Periodic AC (PAC) phase noise amplitude modulation (AM) component to zero and the results will be identical to earlier releases. A value of 1 calculates separate AM and phase noise components. When .OPTION PHNOISEAMP=1, then

```
.MEASURE PHASENOISE extends output variables to the set:<am[noise] >  
<pm[noise] >
```

### Examples

The following explicitly sets the calculation for separate am and pm calculation.

```
.opt phnoiseamp=1
```

### See Also

[.PHASENOISE](#)  
[Amplitude Modulation/Phase Modulation Separation](#)

---

## .OPTION PIVOT

Selects a pivot algorithm.

### Syntax

```
.OPTION PIVOT=0 | 1
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### Description

Set this option to 1 to select a pivot algorithm to achieve convergence in circuits that produce hard-to-solve matrix equations. `PIVOT` selects the numerical pivoting algorithm that is used to manipulate the matrices. Pivoting affects both DC and transient analysis. Usually the reason for choosing a pivot method other than the default 0 is that the circuit contains both very large and very small conductances.

If `PIVOT=0`, HSPICE automatically changes from a non-pivoting to pivot strategy if it detects any diagonal-matrix entry less than `PIVTOL`. This strategy provides the time and memory advantages of non-pivoting inversion and avoids unstable simulations and incorrect results. Use `.OPTION NOPIV` to prevent HSPICE from pivoting.

The `SPARSE` option is the same as `PIVOT`.

### See Also

[.OPTION NOPIV](#)  
[.OPTION PIVTOL](#)

---

## .OPTION PIVTOL

Sets the absolute minimum value for which HSPICE accepts a matrix entry as a pivot.

### Syntax

```
.OPTION PIVTOL=x
```

### Description

Use this option to set the absolute minimum value for which HSPICE accepts a matrix entry as a pivot. `PIVTOL` is used to prevent numeric overflow conditions like divide by 0. If the conductance is less than the value of `PIVTOL`, HSPICE rebuilds the matrix and chooses the `PIVOT` algorithm. If the conductance is greater than the value of `PIVTOL`, the `PIVTOL` value replaces the conductance in the matrix. When a non-pivot algorithm is selected by setting `PIVOT=0`, then `pivtol` is the minimum conductance in the matrix and not a pivot.

The default value of `PIVTOL` is `1e-15` and the range of `PIVTOL` is Min:`1e-35`, Max:`1`, excluding 0. The value of `PIVTOL` must be less than `GMIN` or `GMINDC`. Values that approach 1 increase the pivot. The example below shows how you can correct a “maximum conductance on node error.”

**Note:** If `PIVTOL` is set too small, you run the risk of creating an overflow condition and a convergence problem. If you set the value to 0, an out-of-bounds error is reported.

### Examples

If you get an error message such as:

```
**error** maximum conductance on node 1:v75 } =( 9.2414D-23) is  
less than pivtol in transient analysis.  
Check hookup for this node, set smaller option pivtol and rerun.
```

—the error message informs that the node conductance value is less than the value of `PIVTOL`. Decrease the `PIVTOL` value so that it is less than the value in the error message. The valid range of `pivtol` values is between `1e-35` to 1, excluding 0. For this case a setting `PIVTOL` to `1e-25` resolves the error.

### See Also

- [.OPTION GMIN](#)
- [.OPTION GMINDC](#)
- [.OPTION PIVOT](#)

---

## .OPTION POST

Saves simulation results for viewing by an interactive waveform viewer.

### Syntax

#### *HSPICE Syntax*

```
.OPTION POST=[0 | 1 | 2 | 3 | ASCII | BINARY | CSDF]
```

#### *HSPICE RF Syntax*

```
.OPTION  
  POST=[0 | 1 | 2 | 3 | ASCII | BINARY | CSDF | NW | P | TW | UT | VCD | WDBA]
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### Description

Use this option to save simulation results for viewing by an interactive waveform viewer and to provide output without specifying other parameters.

**Note:** The behavior for .OPTION POST in HSPICE RF is different from the same option used in HSPICE.

The defaults for the POST option supply usable data to most parameters.

- POST=0: Does not output simulation results.
- POST=1, BINARY: (Default if POST is declared without a value) Output format is binary.
- POST=2, ASCII: Output format is ASCII.
- POST=3: Output format is New Wave binary (which enables you to generate .tr0 files that are larger than 2 gigabytes on Linux platforms).
- POST=CSDF: Output format is Common Simulation Data Format (Viewlogic-compatible graph data file format).

Options available to HSPICE RF only:

- POST=NW: Output format is XP/AvanWaves.
- POST=TW: Output format is TurboWave.
- POST=UT: Output format is Veritools Undertow.
- POST=VCD: Output format is value change dump. Use with an .LPRINT command.

- POST=WDBA: Output format is XP/CosmosScope.
- POST=XP: Output format is XP/AvanWaves/CosmosScope.

By default, HSPICE outputs single precision for both time and signal data. If you want to get double precision data, in the netlist set:

```
.OPTION POST POST_VERSION=2001
```

**Note:** .OPTION POST in HSPICE is *not* a global option to dump output in general and then use other options to specify another format. Other options such as PSF, CSDF, SDA, ZUKEN override POST if they are specified after POST, and vice versa. This is unlike HSPICE RF which allows values beyond [0 | 1 | 2 | 3 | ASCII | BINARY | CSDF].

HSPICE uses the last output control option if multiple output control options are specified in the netlist.

### Examples

In this example the option post overrides the options artist/PSF.

```
.option artist=2 psf=2
.option post
```

In this example, the options artist/PSF override the option post.

```
.option post
.option artist=2 psf=2
```

### See Also

[.OPTION POST\\_VERSION](#)

---

## .OPTION POSTLVL

Limits the data written to your waveform file to a specified level of nodes.

### Syntax

```
.OPTION POSTLVL=n
```

### Description

Limits the data written to your waveform file to the level of nodes specified by the *n* parameter. This option differs from `POSTSTOP` in that it specifies the signals of one given level at any level.

**Note:** `.OPTIONS POSTLVL` and `POSTTOP` are ignored if `.OPTION PROBE` is also in the netlist. `.OPTION PROBE` overrides `POSTLVL` and `POSTTOP`.

### Examples

```
.OPTION POSTLVL=2
```

This example limits the data written to the waveform file to only the second-level nodes (voltage and current).

### See Also

[.OPTION POSTTOP](#)

[.OPTION PROBE](#)

---

## **.OPTION POST\_VERSION**

Specifies the post-processing output version for HSPICE/HSPICE RF.

### **Syntax**

```
.OPTION POST_VERSION=x
```

**Default** 9601

### **Description**

Use this option to set the post-processing output version:

- `x=9007` truncates the node name in the post-processor output file to a maximum of 16 characters.
- `x=9601` sets the node name length for the output file consistent with input restrictions (1024 characters) and limits the number of output variables to 9999.
- `x=2001` uses an output file header that displays the correct number of output variables when the number exceeds 9999. This option also changes the digit-number precision in results files to match the value of `.OPTIONNUMDGT` (when `< 5`).

By default, HSPICE outputs single precision for both time and signal data. If you want to get double precision data, in the netlist set:

```
.OPTION POST POST_VERSION=2001
```

If you set `.OPTION POST_VERSION=2001POST=2` in the netlist, HSPICE returns more accurate ASCII results.

To use binary values (with double precision) in the output file, include the following in the input file:

For more accurate simulation results, comment this format.

### **Examples**

If you need to probe more than 9999 signals, set the `POST_VERSION` option to 2001; for example,

```
.OPTION POST_VERSION=2001
```

HSPICE now outputs all the signals into a waveform file and the correct number of output signals is shown rather than \*\*\*\* when the number of signals exceeds 9999. You can load this waveform file in WaveView to view the signals.

### Chapter 3: HSPICE and RF Netlist Simulation Control Options

.OPTION POST\_VERSION

#### See Also

[.OPTION NUMDGT](#)

[.OPTION POST](#)



## **.OPTION POSTTOP**

Limits the data written to the waveform file to data from only the top  $n$  level nodes.

### **Syntax**

```
.OPTION POSTTOP= $n$ 
```

### **Description**

Use this option to limit the data written to your waveform file to data from only the top  $n$  level nodes. This option outputs instances up to  $n$  levels deep. If you do not specify either the `PROBE` or the `POSTTOP` options, HSPICE/HSPICE RF outputs all levels. To enable the waveform display interface, you also need to specify the `.OPTIONPOST` option. This option differs from `.OPTION POSTLVL` in that it specifies the signals of one or multiple levels from the top level down.

**Note:** `.OPTIONS POSTTOP` and `POSTLVL` are ignored if `.OPTION PROBE` is also in the netlist. `.OPTION PROBE` overrides `POSTLVL` and `POSTTOP`.

### **Examples**

```
POSTTOP=3
```

This example limits the data written to the waveform file to only the three top-level nodes (voltage and current).

### **See Also**

- [.OPTION POST](#)
- [.OPTION PROBE](#)
- [.OPTION POSTLVL](#)

---

## .OPTION PROBE

Limits post-analysis output to only variables specified in `.PROBE` and `.PRINT` commands for HSPICE/HSPICE RF.

### Syntax

```
.OPTION PROBE=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### Description

When turned on (1), allows you to set post-analysis output to only variables specified in `.PROBE`, and `PRINT` commands. 0=off. By default, HSPICE outputs all voltages and power supply currents in addition to variables listed in `.PROBE`, and `.PRINT` commands. Using this option can significantly decrease the sizes of simulation output files.

If `.OPTION PROBE` is not set:

- All node voltage and source currents are output to `*.tr#`, `*.ac#`, `*.sw#` files.
- If measured, the resistor or MOSFET current is also output to `*.tr#`, `*.ac#`, or `*.sw#` files.
- If the resistor or MOSFET current are determined by measurement variables, and `.OPTION PUTMEAS` is reset (set to 0), these measurement variables are not output to waveform files.

### See Also

[.PRINT](#)  
[.PROBE](#)  
[.OPTION PROBE](#)  
[.OPTION PUTMEAS](#)

---

## .OPTION PSF

In a standalone HSPICE simulation, specifies whether the output is binary (Parameter Storage Format) or ASCII. When used with HSPICE RF, specifies whether binary or ASCII data is output when you run an HSPICE simulation from the Cadence™ Virtuoso® Analog Design Environment.

### Syntax

```
.OPTION PSF=0|1|2
```

**Default** Value if option is not specified in the netlist: 0

Value if option name is specified without a corresponding value: 1

### Description

Use this option to specify whether HSPICE RF outputs binary (Parameter Storage Format) or ASCII data when you run an HSPICE RF simulation through the Cadence Virtuoso Analog Design Environment.

If you use `.OPTION PSF` only (without `.OPTION ARTIST`), the value of `x` can be 1 or 2.

- If `.OPTION PSF=1`, HSPICE produces binary output.
- If `.OPTION PSF=2`, HSPICE produces ASCII output.

When the netlist contains `.option psf=2` and a `.tran` analysis statement (with no `.op` statement in the netlist file), HSPICE creates the following output files:

- `.op0` — dc node voltage and dc operating points
- `.op1` — transient voltage and transient operating points for the transient end time.

Ordinarily, PSF output is directed to a directory named `./psf` to accommodate the Analog Design Environment. However, HSPICE and Custom Designer users can redirect PSF output by setting the HSPICE command line option `-o` to a directory other than `./psf` (for example: `-o ./results/input`).

**Note:** The PSF format is only supported on Sun/SPARC, Red Hat/SUSE Linux, and IBM AIX platforms, as well as 64-bit versions.

### See Also

[.OPTION ARTIST](#)

---

## .OPTION PURETP

Specifies the integration method to use for reversal time point in HSPICE/  
HSPICE RF.

### Syntax

```
.OPTION PURETP=[0|1]
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### Description

Use this option to specify the integration method to use for reversal time point.

If you set `PURETP=1` and HSPICE finds non-convergence, it uses `TRAP`  
(instead of `Bbackward-Euler`) for the reversed time point.

Use this option with an `.OPTION METHOD=TRAP` command to help some  
oscillating circuits to oscillate if the default simulation process cannot satisfy the  
result.

### See Also

[.OPTION METHOD](#)

## **.OPTION PUTMEAS**

Controls the output variables listed in the `.MEASURE` command.

### **Syntax**

```
.OPTION PUTMEAS=0 | 1
```

**Default** 1

### **Description**

Use this option to control the output variables listed in the `.MEASURE` command.

- 0: Does not save variable values listed in the `.MEASURE` command into the corresponding output file (such as `.tr#`, `.ac#` or `.sw#`). This option decreases the size of the output file.
- 1: Default. Saves variable values listed in the `.MEASURE` command to the corresponding output file (such as `.tr#`, `.ac#` or `.sw#`). This option is similar to the output of HSPICE 2000.4.

### **See Also**

[.MEASURE \(or\) .MEAS](#)

## **.OPTION PZABS**

Sets absolute tolerances for poles and zeros.

### **Syntax**

`.OPTION PZABS=x`

**Default** 1.0e-2

### **Description**

Use this option to set absolute tolerances for poles and zeros in Pole/Zero analysis. Use this option as follows: If  $(X_{\text{real}} + X_{\text{real}} < PZABS)$ , then

$X_{\text{real}}$  and  $X_{\text{imag}} = 0$ . You can also use this option for convergence tests.

### **See Also**

- [.OPTION CSCAL](#)
- [.OPTION FMAX](#)
- [.OPTION FSCAL](#)
- [.OPTION GSCAL](#)
- [.OPTION LSCAL](#)
- [.OPTION PZTOL](#)
- [.OPTION RITOL](#)

## **.OPTION PZTOL**

Sets the relative tolerance for poles and zeros.

### **Syntax**

```
.OPTION PZTOL=x
```

**Default** 1.0e-6

### **Description**

Use this option to set relative tolerances for poles and zeros in Pole/Zero analysis.

### **See Also**

- [.OPTION CSCAL](#)
- [.OPTION FMAX](#)
- [.OPTION FSCAL](#)
- [.OPTION GSCAL](#)
- [.OPTION LSCAL](#)
- [.OPTION PZABS](#)
- [.OPTION RITOL](#)

## **.OPTION RADEGFILE**

Use to specify a MOSRA degradation file name to be used with `SIMMODE=1`.

### **Syntax**

```
.OPTION RADEGFILE=file_name
```

### **Description**

Use this option to specify a MOSRA degradation file name to be used with `SIMMODE=1`. HSPICE will read in the degradation information in the specified file and do a MOSRA post-stress simulation.

### **Examples**

```
.mosra reltotaltime='10*365*24*60*60' lin=11 simmode=1  
.option radegfile = '1.radeg0'
```

### **See Also**

[.MOSRA](#)  
[.OPTION RADEGOUTPUT](#)



## **.OPTION RADEGOUTPUT**

Outputs the MOSRA degradation information to the Word Excel CSV format.

### **Syntax**

```
.OPTION RADEGOUTPUT=CSV
```

### **Description**

Use this option to output the MOSRA degradation information to the Microsoft Excel CSV format. If the `CSV` value is not specified no CSV file is generated.

### **See Also**

[.OPTION RADEGFILE](#)

## **.OPTION RANDGEN**

Specifies the random number generator used in traditional Monte Carlo analysis.

### **Syntax**

```
.OPTION RANDGEN= [0 | 'moa' | 1]
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to specify the random number generator used in HSPICE traditional Monte Carlo analysis. If `RANDGEN= 'moa'` or `1`, then a multiply-with-carry type random number generator with longer cycle is used. If `RANDGEN=0`, then the traditional random number generator is used.

**Note:** The `.OPTION SEED` command is also valid for the new random number generator without usage change.

### **See Also**

[.OPTION RUNLVL](#)  
[.OPTION SEED](#)

## **.OPTION REDEFSUB**

Allows redefinition of a subckt in a netlist.

### **Syntax**

```
.OPTION REDEFSUB = [0 | 1 | 2]
```

**Default** 0

### **Description**

Enables the redefinition of a subcircuit in a netlist.

- 0: Issues an error message for multiple definitions
- 1: Uses the last declared definition
- 2: Uses the first definition

`.option redefsub` without a value equals `.option redefsub=1`.

---

## **.OPTION RELH**

Sets the relative current tolerance from iteration to iteration through voltage-defined branches.

### **Syntax**

```
.OPTION RELH=x
```

### **Description**

Use this option to set the relative current tolerance through voltage-defined branches (voltage sources and inductors) from iteration to iteration.

This option can also be used to check current convergence, but only if the value of the `ABSH` option is greater than zero.

### **See Also**

[.OPTION ABSH](#)

## **.OPTION RELI**

Sets the relative error/tolerance change from iteration to iteration.

### **Syntax**

```
.OPTION RELI=x
```

### **Description**

Use this option to set the relative error/tolerance change from iteration to iteration.

This option determines convergence for all currents in diode, BJT, and JFET devices. (RELMOS sets tolerance for MOSFETs). This value is the change in current from the value calculated at the previous timepoint.

- Default=0.01 for .OPTION KCLTEST=0.
- Default=1e-6 for .OPTION KCLTEST=1.

### **See Also**

[.OPTION RELMOS](#)  
[.OPTION KCLTEST](#)

---

## .OPTION RELIN

(Optimization) Relative input parameter ( $\text{delta\_par\_val} / \text{MAX}(\text{par\_val}, 1\text{e-}6)$ ) for convergence.

### Syntax

```
.OPTION RELIN=value
```

**Default** 0.001

### Description

(Optimization) Relative input parameter ( $\text{delta\_par\_val} / \text{MAX}(\text{par\_val}, 1\text{e-}6)$ ) for convergence. If all optimizing input parameters vary by no more than RELIN between iterations, the solution converges. RELIN is a relative variance test so a value of 0.001 implies that optimizing parameters vary by less than 0.1% from one iteration to the next. If RELIN is set in .OPTION, the setting of RELIN in the .model card will be overridden.

### Examples

```
.option RELIN=1e-6 DYNACC
```

### See Also

[.MODEL](#)

## **.OPTION RELMOS**

Sets the relative error tolerance for drain-to-source current from iteration to iteration.

### **Syntax**

```
.OPTION RELMOS=x
```

### **Description**

Use this option to set the relative error tolerance for drain-to-source current from iteration to iteration.

This option determines convergence for currents in MOSFET devices while `.OPTION RELI` sets the tolerance for other active devices.

This option also sets the change in current from the value calculated at the previous timepoint. HSPICE uses the `.OPTION RELMOS` value only if the current is greater than the `.OPTION ABSMOS` floor value.

Min value: 1e-07; Max value 10.

### **See Also**

- [.OPTION ABSMOS](#)
- [.OPTION RELI](#)
- [.OPTION RELMOS](#)

---

## **.OPTION RELQ**

Sets the timestep size from iteration to iteration.

### **Syntax**

```
.OPTION RELQ=x
```

### **Description**

Use this option in the timestep algorithm for local truncation error (LVLTIM=2). If the capacitor charge calculation in the present iteration exceeds that of the past iteration by a percentage greater than the RELQ value, then HSPICE reduces the internal timestep (delta). The default is 0.01.

### **See Also**

[.OPTION LVLTIM](#)



## **.OPTION RELTOL**

Sets the relative error tolerance for voltages from iteration to iteration.

### **Syntax**

```
.OPTION RELTOL=x
```

**Default**    1e-3

### **Description**

Use this option to set the relative error tolerance for voltages from iteration to iteration. Min value: 1e-20; Max value: 10.

Use this option with the `ABSV` option to determine voltage convergence. Increasing `x` increases the relative error. This option is the same as the `RELV` option. The `RELI` and `RELVDC` options default to the `RELTOL` value.

### **See Also**

- [.OPTION ABSV](#)
- [.OPTION RELI](#)
- [.OPTION RELV](#)
- [.OPTION RELVDC](#)

---

## **.OPTION RELV**

Sets the relative error tolerance for voltages from iteration to iteration.

### **Syntax**

```
.OPTION RELV=x
```

**Default** 1e-3

### **Description**

Use this option to set the relative error tolerance for voltages from iteration to iteration.

If voltage or current exceeds the absolute tolerances, a RELV test determines convergence. Increasing *x* increases the relative error. You should generally maintain this option at its default value. It conserves simulator charge. For voltages, this option is the same as the RELTOL option. Min value: 1e-20; Max value: 10.

### **See Also**

[.OPTION RELTOL](#)

## **.OPTION RELVAR**

Sets the relative voltage change for LVLTIM=1 or 3 from iteration to iteration.

### **Syntax**

```
.OPTION RELVAR=x
```

### **Description**

Use this option to set the relative voltage change for LVLTIM=1 or 3 from iteration to iteration.

Use this option with the ABSVAR and DVDT timestep algorithm. If the node voltage at the current timepoint exceeds the node voltage at the previous timepoint by RELVAR, then HSPICE reduces the timestep and calculates a new solution at a new timepoint. The default is 0.30, or 30 percent.

For additional information, see “[DVDT Dynamic Timestep](#)” in the *HSPICE User Guide: Simulation and Analysis*.

### **See Also**

- [.OPTION ABSVAR](#)
- [.OPTION DVDT](#)
- [.OPTION LVLTIM](#)

---

## **.OPTION RELVDC**

Sets the relative error tolerance for voltages from iteration to iteration.

### **Syntax**

```
.OPTION RELVDC=x
```

### **Description**

Use this option to set the relative error tolerance for voltages from iteration to iteration.

If voltages or currents exceed their absolute tolerances, the `RELVDC` test determines convergence. Increasing the `x` parameter value increases the relative error. You should generally maintain `RELVDC` at its default value to conserve simulator charge.

### **See Also**

[.OPTION RELTOL](#)

## **.OPTION REPLICATES**

Runs replicates of the Latin Hypercube samples.

### **Syntax**

```
.OPTION REPLICATES=number
```

### **Description**

When the advanced sampling method Latin Hypercube is used with traditional Monte Carlo simulation, you can add this option following

`.OPTION SAMPLING_METHOD=LHS`. This option runs replicates of the Latin Hypercube samples. The sample with nominal conditions is simulated once. HSPICE repeats the LHS run the number of times specified by `number`. For example, if, in a regular run, you have 10+1 (including nominal value) iterations, if you set `.OPTION REPLICATES=2`, you generate 21 (or  $2 * \text{Value} + 1$ ) Latin Hypercube samples.

### **Examples**

```
.OPTION SAMPLING_METHOD=LHS  
.OPTION REPLICATES=2
```

### **See Also**

[.OPTION SAMPLING\\_METHOD](#)

---

## **.OPTION RES\_BITS**

Tightens tolerances when using HPP (High Performance Parallel) in transient simulations.

### **Syntax**

```
.OPTION RES_BITS=n
```

**Default** 0

### **Description**

When running a multithread operation in a transient simulation using HPP (only) this option can be used to tighten convergence tolerances. Tightening convergence tolerances enable resolving the least significant bit in an n-bit converter.

**Note:** Setting this option may result in increased number of iterations and, sometimes, slightly increased number of time steps.

### **Examples**

The following example, for a 14-bit A-to-D converter, is set as:

```
.option res_bits=14
```

## **.OPTION RESMIN**

Specifies the minimum resistance for all resistors.

### **Syntax**

```
.OPTION RESMIN=x
```

### **Description**

Use this option to specify the minimum resistance for all resistors, including parasitic and inductive resistances. The range is  $1e-15$  to 10 ohms.

---

## .OPTION RISETIME (or) .OPTION RISETI

Specifies the smallest signal risetime to be supported in elements and analyses that are sensitive to frequency bandwidth and time scale constraints.

### Syntax

.OPTION RISETIME=*x*

**Default** Calculated automatically (see below)

### Description

Use this option to specify the smallest signal risetime to be anticipated when analyzing certain elements that have frequency dependencies. Several HSPICE elements require some knowledge regarding either their maximum frequency of operation, or the minimum signal rise time to be expected. This is particularly true of elements that are described in the frequency domain, yet require time-domain simulation. The `RISETIME` option is used to establish time scale and frequency scale information needed for inverse Fourier transform and convolution calculations.

In the `W`-element (transmission line) model, `RISETIME` is used to determine the maximum signal frequency to be taken into account for frequency dependencies such as skin effect, and dielectric loss (non-zero `Rs` or `Gd`).

In the `S`-element (scattering-parameter) based model, the reciprocal of `RISETIME` sets the maximum signal frequency (`FMAX`) value used for the `S`-parameter analysis.

In the `U`-element (lumped transmission line) model, `RISETIME` is used to set the number of lumps according to the equation:

$$\#lumps = MIN \left[ 20, 1 + 20 \cdot \left( \frac{TDeff}{RISETIME} \right) \right]$$

where, `TDeff` is the end-to-end delay in a transmission line.

When needed, HSPICE automatically calculates a default value for `RISETIME` as follows:

- 25% of the `tstep` value specified with the `.TRAN` command.
- The time corresponding to a 90-degree phase shift for the highest frequency specified in `SIN`, `SFFM`, and `AM` sources.
- The smallest delay time, rise time, fall time, or time increment used in `PULSE`, `EXP`, and `PWL` sources.



**See Also**

[.MODEL](#)  
[.OPTION WACC](#)  
[.OPTION WDELAYOPT](#)

---

## .OPTION RITOL

Sets the minimum ratio value for the (real/imaginary) or (imaginary/real) parts of the poles or zeros.

### Syntax

```
.OPTION RITOL=x
```

**Default** 1.0e-2

### Description

Use this option to set the minimum ratio value for the (real/imaginary) or (imaginary/real) parts of the poles or zeros. Use the RITOL option as follows.

if:  $|X_{\text{imag}}| \leq RITOL \cdot |X_{\text{real}}|$ , then  $X_{\text{imag}} = 0$ . If  $|X_{\text{real}}| \leq RITOL \cdot |X_{\text{imag}}|$ , then  $X_{\text{real}} = 0$ .

### See Also

- [.OPTION CSCAL](#)
- [.OPTION FMAX](#)
- [.OPTION FSCAL](#)
- [.OPTION GSCAL](#)
- [.OPTION LSCAL](#)
- [.OPTION PZABS](#)
- [.OPTION PZTOL](#)
- [.PZ](#)

## **.OPTION RMAX**

Sets the `TSTEP` multiplier, which controls the maximum value for the internal timestep delta for HSPICE/HSPICE RF.

### **Syntax**

```
.OPTION RMAX=x
```

### **Description**

Use this option to set the `TSTEP` multiplier, which controls the maximum value (`DELMAX`) for the delta of the internal timestep:

```
DELMAX=TSTEP x RMAX
```

- The default is 5 if `DVDT` is 4 and `LVLTIM` is 1.
- Otherwise, the default is 2.

Min value:  $1e-9$ ; Max value:  $1e+9$ . The `RMAX` value cannot be smaller than `RMIN`.

### **See Also**

[.OPTION DELMAX](#)  
[.OPTION DVDT](#)  
[.OPTION LVLTIM](#)

---

## **.OPTION RMIN**

Sets the minimum value of delta (internal timestep).

### **Syntax**

```
.OPTION RMIN=x
```

### **Description**

Use this option to set the minimum value of delta (internal timestep). An internal timestep smaller than  $RMIN \times TSTEP$ , terminates the transient analysis, and reports an internal “timestep too small” error. If the circuit does not converge in  $IMAX$  iterations, delta decreases by the amount you set in the  $FT$  option. The default is  $1.0e-9$ . Min value:  $1e-15$ .

### **See Also**

[.OPTION FT](#)

[.OPTION IMAX](#)

---

## .OPTION RUNLVL

Controls runtime speed and simulation accuracy.

### Syntax

```
.OPTION RUNLVL= 1|2|3|4|5|6
```

### Description

Higher values of RUNLVL result in higher accuracy and longer simulation runtimes; lower values result in lower accuracy and faster simulation runtimes.

### For HSPICE:

The RUNLVL option setting controls the scaling of all simulator tolerances simultaneously, affecting timestep control, transient analysis convergence, and model bypass tolerances all at once. Higher values of RUNLVL result in smaller timestep sizes and could result in more Newton-Raphson iterations to meet stricter error tolerances. RUNLVL settings affect transient analysis only.

RUNLVL can be set to 0 (to disable) 1, 2, 3, 4, 5, or 6:

- 1: Lowest simulation runtime
- 2: More accurate than RUNLVL=1 and faster than RUNLVL=3
- 3: Default value, similar to HSPICE's original default mode
- 4: More accurate than RUNLVL=3 and faster than RUNLVL=5
- 5 or 6: Corresponds to HSPICE's standard accurate mode for most circuits:
  - 5 is similar to the standard accurate mode in HSPICE
  - 6 has the highest accuracy

If RUNLVL is specified in the netlist without a value, the value is the default, 3.

If .OPTION ACCURATE is specified in the netlist together with RUNLVL, the value of RUNLVL is limited to 5 or 6; specifying a specifying a RUNLVL value of 1, 2, 3, or 4 defaults to 5.

If .OPTION RUNLVL is not turned off, there is no dependency with GEAR and ACCURATE options, and

```
.OPTION ACCURATE method=GEAR RUNLVL
```

is equivalent to

```
.OPTION method=GEAR ACCURATE RUNLVL
```

## Chapter 3: HSPICE and RF Netlist Simulation Control Options

### .OPTION RUNLVL

The RUNLVL option interacts with other options as follows:

- Regardless of its position in the netlist, RUNLVL ignores the following step control-related options (which are replaced by automated algorithms):

```
LVLTIM DVDT FT FAST TRTOL ABSVARRELVAR RELQ CHGTOL DVTR  
IMIN ITL3
```

- See the notes to the table below for discussion of options ACCURATE and BYPASS in relation to RUNLVL if it is specified in the netlist.
- The `tstep` value specified with the `.TRAN` command affects timestep control when a RUNLVL option is used. Timestep values larger than `tstep*RMAX` use a tighter timestep control tolerance.

For information on how RUNLVL values affect other options, see the following section, and also see [RUNLVL=N](#) and [RUNLVL, ACCURATE, FAST, GEAR method](#) in Appendix B of this manual.

#### For HSPICE RF:

While HSPICE RF supports `.OPTION RUNLV`, this option is most compatible with HSPICE. For HSPICE RF, the `SIM_ACCURACY` option gives you a more continuous range of settings. You can use `.OPTION RUNLVL` to control runtime speed and simulation accuracy. As in HSPICE, higher values of RUNLVL result in higher accuracy and longer simulations; lower values result in lower accuracy and faster simulation.

`.OPTION RUNLVL` maps to `.OPTION SIM_ACCURACY` as follows:

- RUNLVL=1: SIM\_ACCURACY=0.5
- RUNLVL=2: SIM\_ACCURACY=0.75
- RUNLVL=3: SIM\_ACCURACY=1
- RUNLVL=4: SIM\_ACCURACY=5
- RUNLVL=5: SIM\_ACCURACY=10
- RUNLVL=6: SIM\_ACCURACY=100

#### Interactions Between .OPTION RUNLVL and Other Options

Since the latest algorithm invoked by RUNLVL sets the timestep and error tolerance internally, many transient error tolerance and timestep control options are no longer valid; furthermore, to assure the most efficiency of the new RUNLVL algorithm, you should let the new engine manage everything itself. Options that are recommended not to tune are listed in the table, as well.

**Note:** Once RUNLVL is set, it does not = 0.

Option	Default value without RUNLVL	Default value with RUNLVL=3	User definition ignored	Recommend not to tune
ABSV/VNTOL	50u	50u	-	x
ABSVAR	500m	500m	x	-
ACCURATE <sup>1</sup>	0	0	-	-
BYPASS <sup>a</sup>	2	2 for RUNLVL=1-6	-	-
CHGTOL	1.0f	1.0f	x	-
DI	100	100	-	x
DVDT	4	4	x	-
DVTR	1.0k	1.0k	x	-
FAST <sup>2</sup>	0	0	x	-
FS	250m	250m	-	x
FT	250m	250m	x	-
IMIN/ITL3	3	3	x	-
LVLTIM	1	4	x	-
METHOD <sup>3</sup>	TRAP	TRAP	-	-
RELQ	10m	10m	x	-
RELTOL	1.0m	1.0m	-	x
RELV	1.0m	1.0m	-	x
RELVAR	300.0m	300.0m	x	-
RMAX	5	5	x	-
RMIN	1.0n	1.0n	-	x
TRTOL	7	7	x	-

## Chapter 3: HSPICE and RF Netlist Simulation Control Options

### .OPTION RUNLVL

1. *ACCURATE and BYPASS notes:*

1. If *.option ACCURATE* is set, then the *RUNLVL* value is limited to 5 or 6. Specifying a *RUNLVL* less than 5 results in a simulation at *RUNLVL=5*. When both *ACCURATE* and *RUNLVL* are set, the *RUNLVL* algorithm will be used.

2. When *RUNLVL* is set, *BYPASS* is set to 2. Users can redefine the *BYPASS* value by setting *.option BYPASS=value*; this behavior is independent of the order of *RUNLVL* and *BYPASS*;

2. The *FAST* option is disabled by the *RUNLVL* option; setting the *RUNLVL* value to 1 is comparable to setting the *FAST* option.

3. *RUNLVL* can work with *METHOD=GEAR*; in cases where *GEAR* only determines the numeric integration method during transient analysis, the other options that were previously set by *GEAR* (when there is no *RUNLVL*) now are determined by the *RUNLVL* mode. This behavior is independent of the order of *RUNLVL* and *METHOD*. See below.

### See Also

- [.OPTION ACCURATE](#)
- [.OPTION BYPASS](#)
- [.OPTION DVDT](#)
- [.OPTION LVLTIM](#)
- [.OPTION METHOD](#)
- [.OPTION RELTOL](#)
- [.TRAN](#)
- [.OPTION SIM\\_ACCURACY \(RF\)](#)



---

## .OPTION SAMPLING\_METHOD

Enables use of advanced sampling methods with traditional Gaussian Monte Carlo trials.

### Syntax

```
.OPTION SAMPLING_METHOD=SRS | LHS | Factorial | OFAT | Sobol |
+ Neiderreiter
```

**Default** SRS

---

Argument	Description
SRS	Simple random sampling performed in traditional HSPICE Monte Carlo method
LHS	Latin Hypercube sampling; efficient for large number of variable parameters (used with .OPTION REPLICATES)
Factorial	Factorial sampling; <ul style="list-style-type: none"> <li>▪ Evaluates the circuit response at the extremes of variable ranges to get an idea of the worst and best case behavior.</li> <li>▪ Creates polynomial response surface approximations.</li> </ul>
OFAT	One-Factor-At-a-Time sampling; useful for sensitivity studies and for constructing low order response surface approximations.
Sobol	Sobol sampling uses low discrepancy sequences (LDS); LDS sample points are more frequently distributed compared to LHS and the sampling error is lower. Sobol is used with a sampling dimension of 40 or less.
Neiderreiter	LDS sampling sequence useful as a sampling method for cases of a sampling dimension up to 318. If that number is exceeded, HSPICE switches to the default SRS sampling method.

---

### Description

This option enables use of sampling methods other than Gaussian techniques available in traditional HSPICE Monte Carlo simulation. For a full discussion about advanced sampling methods see [Comparison of Sampling Methods](#) in the *HSPICE User Guide: Simulation and Analysis*. These methods are also available in the HSPICE Variation Block functionality.

### Chapter 3: HSPICE and RF Netlist Simulation Control Options

.OPTION SAMPLING\_METHOD

#### See Also

[.OPTION REPLICATES](#)

## **.OPTION SAVEHB**

Saves the final-state variable values from an HB simulation.

### **Syntax**

```
.OPTION SAVEHB='filename'
```

### **Description**

Use this option to save the final state (that is, the no-sweep point or the steady state of the first sweep point) variable values from an HB simulation to the specified file.

This file can be loaded as the starting point for another simulation by using a `LOADHB` option.

### **See Also**

[.HB](#)

[.OPTION LOADHB](#)

---

## **.OPTION SAVESNINIT**

Saves the operating point at the end of Shooting Newton initialization (sninit).

### **Syntax**

```
.OPTION SAVESNINIT="filename"
```

### **Description**

Use this option to save an operating point file at the end of a SN initialization for use as initial conditions for another Shooting Newton analysis. For more information, see [SN Steady-State Time Domain Analysis](#) in the *HSPICE User Guide: RF Analysis*.

### **See Also**

- [.SN](#)
- [.OPTION LOADSNINIT](#)
- [.OPTION SAVESNINIT](#)
- [.OPTION SNACCURACY](#)
- [.OPTION SNMAXITER \(or\) SN\\_MAXITER](#)

---

## .OPTION SCALE

Sets the element scaling factor for HSPICE/HSPICE RF.

### Syntax

```
.OPTION SCALE=x
```

### Description

Use this option to scale geometric element instance parameters whose default unit is meters. You can also use this option with `.OPTION GEOSHRINK` to scale an element even more finely (usually through a technology file). The effective scaling factor is the product of the two parameters; HSPICE will use `scale*geoshrink` to scale the parameters/dimensions.

In HSPICE, the possible geometrical instance parameters include width, length, or area for both passive and active devices, in addition to the commonly known MOSFET parameters such as AS, AD, PS, PD, and so on.

- For active elements, the geometric parameters are:  
Diode — W, L, Area JFET/MESFET — W, L, Area MOS — W, L, AS, AD, PS, PD
- For passive elements having values calculated as a function geometry, the geometric parameters are:  
Resistor — W, L Capacitor — W, L

In cases where you want to selectively scale a required instance, such as in an encrypted file, you can use `.OPTION HIER_SCALE`.

### See Also

[.OPTION GEOSHRINK](#)  
[.OPTION BA\\_SCALE](#)  
[.OPTION CMIUSRFLAG](#)  
[.OPTION HIER\\_SCALE](#)

## **.OPTION SCALM**

Sets the model scaling factor.

### **Syntax**

```
.OPTION SCALM=x
```

### **Description**

Use this option to set the scaling factor defined in a `.MODEL` command for an element. See the [HSPICE Elements and Device Models Manual](#) for parameters that this option scales. For MOSFET devices, this option is ignored in Level 49 and higher model levels. See the [HSPICE Reference Manual: MOSFET Models](#) for levels available to the SCALM option.

### **See Also**

[.MODEL](#)

## .OPTION SEARCH

Automatically accesses a library or individual vendor files.

### Syntax

```
.OPTION SEARCH=`directory_path' [path_name]
```

### Description

Use this option to auto-access a library, or, using `path_name`, to search for library (`*.lib`) files. Typically, vendors supply part files containing a single subcircuit. The name of the file is the same as the subcircuit with the file extension `*.inc`. The commands `.LIB`, `.INC`, and `.LOAD` search for the file. In addition, HSPICE supports `.OPTION SEARCH` for `.VEC` commands. The path can be given as `'/remote/home1/aa'` or as `'../'`

### Examples

```
.OPTION SEARCH=`$installdir/parts/vendor'
```

This example searches for models in the `vendor` subdirectory, under the `$installdir/parts` installation directory (see [Figure 14](#)). The `parts` directory contains the DDL subdirectories.

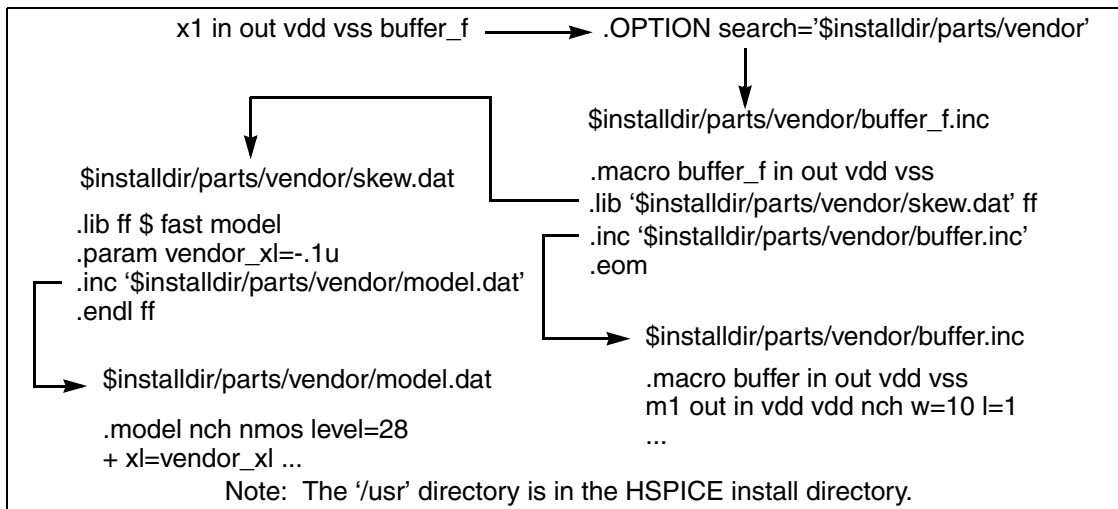


Figure 14 Vendor Library Usage

### See Also

[Signal Integrity Examples](#) for netlists using `.OPTION SEARCH` including `iotran.sp`, `qa8.sp`, and `qabounce.sp`.

---

## **.OPTION SEED**

Specifies the starting seed for the random-number generator in Monte Carlo analysis.

### **Syntax**

```
.OPTION SEED=x | 'random'
```

### **Description**

Use this option to specify the starting seed for the random-number generator in HSPICE Monte Carlo analysis. The minimum value is 1; the maximum value of is 259200 of SEED. If `SEED= 'random'`, HSPICE assigns a random number between 1 and 259200 according to the system clock and prints it in the `.lis` file for you to debug. `.OPTION SEED` is supported In HSPICE and the RF flow.

### **See Also**

[.OPTION RANDGEN](#)



## **.OPTION SHRINK**

Scales the final constant capacitance value (only works with `.OPTION CMIUSRFLAG=3`).

### **Syntax**

```
.OPTION SHRINK= val
```

### **Description**

Use this option to scale the final constant capacitance value. The default setting overrides `.OPTION SHRINK` before applying `shrink*shrink` scaling to constant capacitance value. The following is the usage of `.OPTION SHRINK` and instance parameter `shrink`:

- 1: If both `.OPTION SHRINK` and the `shrink` instance are not set in the netlist, do nothing.
- 2: If only `.OPTION SHRINK` is set in the netlist, use it to scale the final constant capacitance value.
- 3: If the instance parameter `shrink` is set in the netlist, use the instance `shrink` to scale the final constant capacitance value.

### **See Also**

[.OPTION CMIUSRFLAG](#)

---

## .OPTION SIM\_ACCURACY

Sets and modifies the size of timesteps.

### Syntax

```
.OPTION SIM_ACCURACY=value
```

**Default** Conditional, see below

### Description

Use this option to set and modify the size of timesteps. This option applies to all modes and tightens all tolerances, such as: Newton-Raphson tolerance, local truncation error, and other errors.

The *value* must be a positive number. The default is 1. If you specify `.OPTION ACCURATE`, the default value is 10; you can use `.option sim_accuracy=10` instead of `.option accurate`. They are interchangeable. You can set `.option sim_accuracy=10` if you have not set previous `sim_accuracy` settings that are 10 or greater or have previously set `.option accurate`.

To set global accuracy, use `.OPTION SIM_ACCURACY=n`, where *n* is a number greater than 0.

You can apply different accuracy settings to different blocks or time intervals. The syntax to set accuracy on a block, instance, or time interval is similar to the settings used for a power supply.

### Examples

This example sets accuracy to 3 for the XNAND1 and XNAND2 instances and 4 for all instances of the INV subcircuit. Globally, the accuracy is 2. If accuracy settings conflict, then HSPICE RF uses the higher accuracy value. At 12.0ns before the end of the simulation, the global accuracy level is 5. Because this is higher than 2, 3, or 4, it overrides all previous settings.

```
.OPTION SIM_ACCURACY=2
.OPTION SIM_ACCURACY=3 | XNAND1 XNAND2
.OPTION SIM_ACCURACY=4 | @INV
.OPTION SIM_ACCURACY=5 | 12.0n
.OPTION SIM_ACCURACY=5 | 20n
.OPTION SIM_ACCURACY=3 | 40ns
.OPTION SIM_ACCURACY=5 | 20ns 3 | 35ns 7 | 50ns
```

### See Also

[.OPTION FFT\\_ACCURATE](#)  
[.OPTION ACCURATE](#)

## **.OPTION SIM\_DELTAI**

Sets the selection criteria for current waveforms in WDB and NW format.

### **Syntax**

```
.OPTION SIM_DELTAI=value
```

**Default**    0 amps

### **Description**

Use this option to set the selection criteria for RF current waveforms in WDB and NW format (see “[Eliminating Current Datapoints](#)” in the *HSPICE User Guide: RF Analysis*). The *value* parameter specifies the amount of change.

### **Examples**

In this example, at the *n* timestep, HSPICE RF saves only data points that change by more than 0 amps from previous values at the *n-1* timestep.

```
.OPTION SIM_DELTAI = 0amps
```

### **See Also**

[.OPTION SIM\\_DELTAV](#)

---

## **.OPTION SIM\_DELTAV**

Sets the selection criteria for current waveforms in WDB and NW format.

### **Syntax**

```
.OPTION SIM_DELTAV=value
```

**Default**    1mv

### **Description**

Sets the selection criteria for RF current waveforms in WDB and NW format (see “[Eliminating Voltage Datapoints](#)” in the *HSPICE User Guide: RF Analysis*).

The *value* parameter specifies the amount of change.

### **Examples**

In this example, at the *n* timestep, HSPICE RF saves only data points that change by more than 1 mV from their previous values at the *n-1* timestep.

```
.OPTION SIM_DELTAV = 1mv
```

### **See Also**

[.OPTION SIM\\_DELTAI](#)

---

## .OPTION SIM\_DSPF

Runs simulation with standard DSPF expansion of all nets from one or more DSPF files.

### Syntax

```
.OPTION SIM_DSPF="[scope] dspf_filename"
```

### Description

Use this option to run simulation with standard DSPF expansion of all nets from one or more DSPF files.

`scope` can be a subcircuit definition or an instance. If you do not specify `scope`, it defaults to the top-level definition.

You can repeat this option to include more DSPF files.

This option can accelerate simulation by more than 100%. You can further reduce total CPU time by including the `.OPTION SIM_LA` in the netlist.

For designs of 5K transistors or more, including `.OPTION SIM_DSPF_ACTIVE` in your netlist to expand only active nodes also provides a performance gain.

**Note:** HSPICE RF requires both a DSPF file and an ideal extracted netlist. Only flat DSPF files are supported; hierarchy commands, such as `.SUBCKT` and `.x1` are ignored.

For additional information, see "[Post-Layout Back-Annotation](#)" in the *HSPICE User Guide: RF Analysis*.

### Examples

In Example 1, the parasitics in the DSPF file are mapped into the hierarchical ideal netlist.

#### Example 1

```
$ models
.MODEL p pmos
.MODEL n nmos

.INCLUDE add4.dspf
.OPTION SIM_DSPF="add4.dspf"
.VEC "dspf_adder.vec"
.TRAN 1n 5u
vdd vdd 0 3.3
.OPTION POST
.END
```

### Chapter 3: HSPICE and RF Netlist Simulation Control Options

#### .OPTION SIM\_DSPF

In Example 2, the `SIM_DSPF` option accelerates the simulation by more than 100%. By using the `SIM_LA` option at the same time, you can further reduce the total CPU time:

#### *Example 2*

```
$ models
.MODEL p pmos
.MODEL n nmos
.INCLUDE add4.dspf
.OPTION SIM_DSPF="add4.dspf"
.OPTION SIM_LA=PACT
.VEC "dspf_adder.vec"
.TRAN 1n 5u
vdd vdd 0 3.3
.OPTION POST
.END
```

Example 3, the `x1.spf` DSPF file is back-annotated to the `x1` top-level instance. It also back-annotates the `inv.spf` DSPF file to the `inv` subcircuit.

#### *Example 3*

```
.OPTION SIM_DSPF = "x1 x1.spf"
.OPTION SIM_DSPF = "inv inv.spf"
```

#### **See Also**

- [.OPTION SIM\\_LA](#)
- [.OPTION SIM\\_DSPF\\_ACTIVE](#)
- [.OPTION SIM\\_DSPF\\_SCALEC](#)
- [.OPTION SIM\\_DSPF\\_SCALER](#)
- [.OPTION SIM\\_SPEF](#)

---

## .OPTION SIM\_DSPF\_ACTIVE

Runs simulation with selective DSPF expansion of active nets from one or more DSPF files.

### Syntax

```
.OPTION SIM_DSPF_ACTIVE="active_node"
```

### Description

Use this option to run simulation with selective DSPF expansion of active nets from one or more DSPF files. HSPICE RF performs a preliminary verification run to determine the activity of the nodes and generates two ASCII files: *active\_node.rc* and *active\_node.rcxt*. These files save all active node information in both Star-RC and Star-RCXT formats. If an *active\_node* file is not generated from the preliminary run, no nets are expanded. Active nets are added to the file as they are identified in the subsequent transient simulation. A second simulation run using the same file and option causes only the nets listed in the *active\_node* file to be expanded. It is possible that activity changes are due to timing changes caused by expansion of the active nets. In this case, additional nets are listed in the *active\_node* file and a warning is issued.

HSPICE RF uses the *active\_node* file and the DSPF file with the ideal netlist to expand only the active portions of the circuit. If a net is latent, then HSPICE RF does not expand it, which saves memory and CPU time.

For additional information, see “[Selective Post-Layout Flow](#)” in the *HSPICE User Guide: RF Analysis*.

### Examples

In the following example, an active net in which the tolerance of the voltage change is greater than 0.5V is saved to both the *active.rc* and *active.rcxt* files. Based on these files, HSPICE RF back-annotates only the active parasitics from *x1.spf* and *s2.spf* to the *x1* and *x2* top-level instances.

```
.OPTION SIM_DSPF = "x1 x1.spf"  
.OPTION SIM_DSPF = "x2 x2.spf"  
.OPTION SIM_DSPF_ACTIVE = "active"  
.OPTION SIM_DSPF_VTOL = 0.5V
```

### See Also

- [.OPTION SIM\\_DSPF](#)
- [.OPTION SIM\\_DSPF\\_MAX\\_ITER](#)
- [.OPTION SIM\\_DSPF\\_VTOL](#)
- [.OPTION SIM\\_SPEF\\_ACTIVE](#)

## **.OPTION SIM\_DSPF\_INSEERROR**

Skips unmatched instances.

### **Syntax**

```
.OPTION SIM_DSPF_INSEERROR=ON | OFF
```

**Default**    OFF

### **Description**

Use this option to skip unmatched instances.

- ON: Skips unmatched instances
- OFF: Does not skip unmatched instances.

For additional information, see [“Additional Post-Layout Options”](#) in the *HSPICE User Guide: RF Analysis*.



## **.OPTION SIM\_DSPF\_LUMPCAPS**

Connects a lumped capacitor with a value equal to the net capacitance for instances missing in the hierarchical netlist.

### **Syntax**

```
.OPTION SIM_DSPF_LUMPCAPS=ON | OFF
```

**Default**    ON

### **Description**

Use this option to connect a lumped capacitor with a value equal to the net capacitance for instances missing in the hierarchical netlist.

- ON (default): Adds lumped capacitance while ignoring other net contents
- OFF: Uses net contents

For additional information, see “[Additional Post-Layout Options](#)” in the *HSPICE User Guide: RF Analysis*.

---

## **.OPTION SIM\_DSPF\_MAX\_ITER**

Specifies the maximum number of simulation runs for the second selective DSPF expansion pass.

### **Syntax**

```
.OPTION SIM_DSPF_MAX_ITER=value
```

**Default** 1

### **Description**

Use this option to specify the maximum number of simulation runs for the second selective DSPF expansion pass.

The *value* parameter specifies the number of iterations for the second simulation run.

Some of the latent nets might turn active after the first iteration of the second simulation run. In this case:

- Resimulate the netlist to ensure the accuracy of the post-layout simulation.
- Use this option to set the maximum number of iterations for the second run. If the *active\_node* remains the same after the second simulation run, HSPICE RF ignores these options.

For details, see “[Selective Post-Layout Flow](#)” *HSPICE User Guide: RF Analysis*.

### **See Also**

[.OPTION SIM\\_DSPF\\_ACTIVE](#)  
[.OPTION SIM\\_DSPF\\_VTOL](#)

## **.OPTION SIM\_DSPF\_RAIL**

Controls whether power-net parasitics are back-annotated

### **Syntax**

```
.OPTION SIM_DSPF_RAIL=ON | OFF
```

**Default**    OFF

### **Description**

Use this option to control whether power-net parasitics are back-annotated.

- OFF: Do not back-annotate nets in a power rail
- ON: Back-annotate nets in a power rail

For additional information, see “[Additional Post-Layout Options](#)” in the *HSPICE User Guide: RF Analysis*.

---

## **.OPTION SIM\_DSPF\_SCALEC**

Scales the capacitance values in a DSPF file for a standard DSPF expansion flow.

### **Syntax**

```
.OPTION SIM_DSPF_SCALEC=scaleC
```

### **Description**

Use this option to scale the capacitance values in a DSPF file for a standard DSPF expansion flow.

The *scaleC* parameter specifies the scale factor.

For additional information, see “[Additional Post-Layout Options](#)” in the *HSPICE User Guide: RF Analysis*.

### **See Also**

[.OPTION SIM\\_LA](#)

[.OPTION SIM\\_DSPF\\_ACTIVE](#)

## **.OPTION SIM\_DSPF\_SCALER**

Scales the resistance values in a DSPF file for a standard DSPF expansion flow.

### **Syntax**

```
.OPTION SIM_DSPF_SCALER=scaleR
```

### **Description**

Use this option to scale the resistance values in a DSPF file for a standard DSPF expansion flow.

The *scaleR* specifies the scale factor.

For additional information, see “[Additional Post-Layout Options](#)” in the *HSPICE User Guide: RF Analysis*.

### **See Also**

[.OPTION SIM\\_LA](#)  
[.OPTION SIM\\_DSPF\\_ACTIVE](#)

---

## .OPTION SIM\_DSPF\_VTOL

Specifies multiple DSPF active thresholds.

### Syntax

```
.OPTION SIM_DSPF_VTOL=value | scope1 scope2 ...
+ scopen"
```

**Default** 0.1V

### Description

Use this option to specify multiple DSPF active thresholds.

- The *value* parameter specifies the tolerance of voltage change. This value should be relatively small compared to the operating range of the circuit or smaller than the supply voltage.
- *scopen* can be a subcircuit definition that uses a prefix of “@” or a subcircuit instance.

HSPICE RF performs a second simulation run by using the *active\_node* file, the DSPF, and the hierarchical LVS ideal netlist to back-annotate only active portions of the circuit. If a net is latent, HSPICE RF does not expand the net. This saves simulation runtime and memory.

By default, HSPICE RF performs only one iteration of the second simulation run. Use the *SIM\_DSPF\_MAX\_ITER* option to change this setting.

For additional information, see “[Selective Post-Layout Flow](#)” in the *HSPICE User Guide: RF Analysis*.

### Examples

In Example 1, the first line sets the sensitivity voltage to 0.01V. Subcircuit definition *snsamp* and the subcircuit instance *xvco* have full parasitics if their nodes move more than 0.01V during active nodes generation. In the second line, *xand* and *xff* are less sensitive than the default, indicating that they are not sensitive to parasitics.

#### Example 1

```
.OPTION SIM_DSPF_VTOL="0.01 | @snsamp xvco"
.OPTION SIM_DSPF_VTOL="0.25 | xand xff"
```

In this example, the sense amp circuit uses full parasitics if their nodes move more than 0.01V during active-node generation. The inv subcircuit definition is less sensitive than the default so the nodes are less sensitive to the parasitics.

#### *Example 2*

```
.OPTION SIM_DSPF = "inv inv.spf"  
.OPTION SIM_DSPF = "senseamp senseamp.spf"  
.OPTION SIM_DSPF_ACTIVE = "activenet"  
.OPTION SIM_DSPF_VTOL = "0.15 | @inv"  
.OPTION SIM_DSPF_VTOL = "0.01 | @senseamp"
```

#### **See Also**

[.OPTION SIM\\_DSPF\\_ACTIVE](#)  
[.OPTION SIM\\_DSPF\\_MAX\\_ITER](#)

---

## .OPTION SIM\_LA

Activates linear matrix (RC) reduction for HSPICE/HSPICE RF.

### Syntax

```
.OPTION SIM_LA=[PACT | PI | 0 | 1 | 2 ]
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### Description

Use this option to activate linear matrix reduction. `SIM_LA` does not reduce a node used by any analysis command, such as `.PROBE`, `.MEASURE`, and so on

This option accelerates the simulation of circuits that include large linear RC networks by reducing all matrixes that represent RC networks.

- 0 turns off `SIM_LA`
- 1 is the equivalent of `PACT`, which selects the Pole Analysis via Congruence Transforms (PACT) algorithm to reduce RC networks in a well-conditioned manner, while preserving network stability.
- 2 invokes the `PI` algorithm to create PI models of the RC networks.
- If `SIM_LA` is not specified in the input file, the `lis` file returns `SIM_LA=0`.
- If `SIM_LA` is specified with no value or `SIM_LA=PACT`, the `lis` file returns `SIM_LA=1`.
- If `SIM_LA=PI`, the `lis` file returns `SIM_LA=2`.

For additional information, see “[Linear Acceleration](#)” in the *HSPICE User Guide: Simulation and Analysis* or “[Linear Acceleration](#)” in the *HSPICE User Guide: RF Analysis*.

### See Also

[.OPTION SIM\\_DSPF](#)  
[.OPTION LA\\_FREQ](#)  
[.OPTION LA\\_MAXR](#)  
[.OPTION LA\\_MINC](#)  
[.OPTION LA\\_TIME](#)  
[.OPTION LA\\_TOL](#)



## **.OPTION SIM\_LA\_FREQ**

Specifies the upper frequency for which accuracy must be preserved.

### **Syntax**

```
.OPTION SIM_LA_FREQ=value
```

**Default** 1GHz

### **Description**

Use this option to specify the upper frequency for which accuracy must be preserved. The *value* parameter specifies the upper frequency for which the PACT algorithm must preserve accuracy. If *value* is 0, the algorithm drops all capacitors because only DC is of interest.

The maximum frequency required for accurate reduction depends on both the technology of the circuit and the time scale of interest. In general, the faster the circuit, the higher the maximum frequency. For additional information, see “[Linear Acceleration](#)” in the *HSPICE User Guide: RF Analysis*.

### **See Also**

[.OPTION SIM\\_LA](#)  
[.OPTION SIM\\_LA\\_TIME](#)

---

## **.OPTION SIM\_LA\_MAXR**

Specifies the maximum resistance for linear matrix reduction.

### **Syntax**

```
.OPTION SIM_LA_MAXR=value
```

**Default**    1e15 ohms

### **Description**

Use this option to specify the maximum resistance for linear matrix reduction. The *value* parameter specifies the maximum resistance preserved in the reduction. The linear matrix reduction process assumes that any resistor greater than *value* has an infinite resistance and drops the resistor after reduction completes. For additional information, see “[Linear Acceleration](#)” in the *HSPICE User Guide: RF Analysis*.

### **See Also**

[.OPTION SIM\\_LA](#)

## **.OPTION SIM\_LA\_MINC**

Specifies the minimum capacitance for linear matrix reduction.

### **Syntax**

```
.OPTION SIM_LA_MINC=value
```

**Default** 1e-16 farads

### **Description**

Use this option to specify the minimum capacitance for linear matrix reduction.

The *value* parameter specifies the minimum capacitance preserved in the reduction.

The linear matrix reduction process lumps any capacitor smaller than *value* to ground after the reduction completes.

For additional information, see “[Linear Acceleration](#)” in the *HSPICE User Guide: RF Analysis*.

### **See Also**

[.OPTION SIM\\_LA](#)

---

## .OPTION SIM\_LA\_TIME

Specifies the minimum time for which accuracy must be preserved.

### Syntax

```
.OPTION SIM_LA_TIME=value
```

**Default** 1 ns.

### Description

Use this option to specify the minimum time for which accuracy must be preserved.

The *value* parameter specifies the minimum switching time for which the PACT algorithm preserves accuracy.

Waveforms that occur more rapidly than the minimum switching time are not accurately represented.

This option is simply an alternative to .OPTION SIM\_LA\_FREQ. The default is equivalent to setting SIM\_LA\_FREQ=1GHz.

**Note:** Higher frequencies (smaller times) increase accuracy, but only up to the minimum time step used in HSPICE RF.

For additional information, see “[Linear Acceleration](#)” in the *HSPICE User Guide: RF Analysis*.

### Examples

For a circuit having a typical rise time of 1ns, either set the maximum frequency to 1 GHz, or set the minimum switching time to 1ns:

```
.OPTION SIM_LA_FREQ=1GHz  
-or-  
.OPTION SIM_LA_TIME=1ns
```

However, if spikes occur in 0.1ns, HSPICE RF does not accurately simulate them. To capture the behavior of the spikes, use:

```
.OPTION SIM_LA_FREQ=10GHz  
-or-  
.OPTION SIM_LA_TIME=0.1ns
```

### See Also

[.OPTION SIM\\_LA](#)  
[.OPTION SIM\\_LA\\_FREQ](#)

## **.OPTION SIM\_LA\_TOL**

Specifies the error tolerance for the PACT algorithm.

### **Syntax**

```
.OPTION SIM_LA_TOL=value
```

**Default** 0.05ns.

### **Description**

Use this option to specify the error tolerance for the PACT algorithm.

The *value* parameter must specify a real number between 0.0 and 1.0.

For additional information, see “[Linear Acceleration](#)” in the *HSPICE User Guide: RF Analysis*.

### **See Also**

[.OPTION SIM\\_LA](#)

---

## .OPTION SIM\_ORDER

Controls the amount of Backward-Euler (BE) method to mix with the Trapezoidal (TRAP) method for hybrid integration.

### Syntax

```
.OPTION SIM_ORDER=x
```

**Default** 1.9

### Description

Use this option to control the amount of Backward-Euler (BE) method to mix with the Trapezoidal (TRAP) method for hybrid integration.

The *x* parameter must specify a real number between 1.0 and 2.0.

- SIM\_ORDER=1.0 selects BE
- SIM\_ORDER=2.0 selects TRAP.

**Note:** .OPTION SIM\_ORDER has precedence over .OPTION SIM\_TRAP.

A higher order is more accurate, especially with inductors (such as crystal oscillators), which need SIM\_ORDER=2.0. A lower order has more damping.

This option affects time stepping when you set .OPTION METHOD to TRAP or TRAPGEAR.

### Examples

This example causes a mixture of 10% Gear-2 and 90% BE-trapezoidal hybrid integration. The BE-trapezoidal part is 10% BE.

```
.option sim_order=1.9
```

### See Also

[.OPTION METHOD](#)  
[.OPTION SIM\\_TRAP](#)

## **.OPTION SIM\_OSC\_DETECT\_TOL**

Specifies the tolerance for detecting numerical oscillations.

### **Syntax**

```
.OPTION SIM_OSC_DETECT_TOL=value
```

**Default** 10<sup>8</sup>

### **Description**

Use this option to specify the tolerance for detecting numerical oscillations. If HSPICE RF detects numerical oscillations, it inserts Backward-Euler (BE) steps. Smaller values of this tolerance result in fewer BE steps.

### **See Also**

[.OPTION METHOD](#)

---

## .OPTION SIM\_POSTAT

Limits waveform output to nodes in the specified subcircuit instance.

### Syntax

```
.OPTION SIM_POSTAT=instance
```

### Description

Use this option to limit waveform output to nodes in the specified subcircuit instance only. SIM\_POSTAT is available for both HSPICE and HSPICE RF. Wildcards are supported.

### Examples

The following example outputs X1.X4; see [Figure 15](#).

```
.OPTION SIM_POSTAT=X1.X4
```

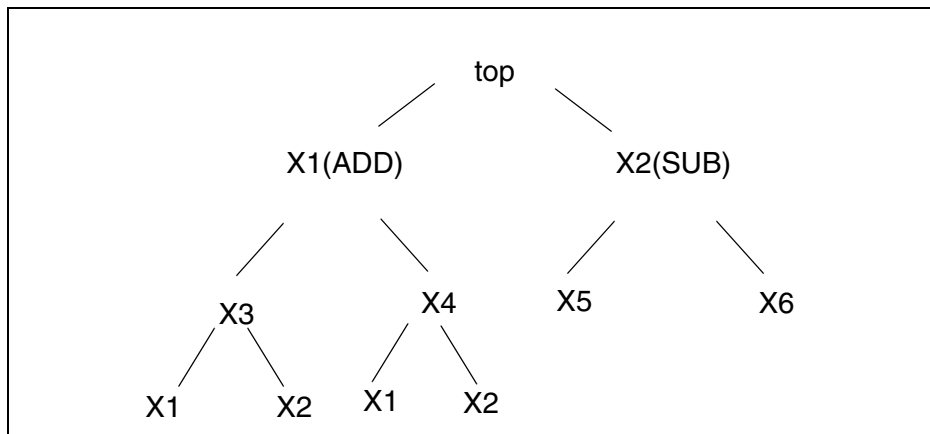


Figure 15 Node Hierarchy

### See Also

[.OPTION SIM\\_POSTSKIP](#)  
[.OPTION SIM\\_POSTTOP](#)



## **.OPTION SIM\_POSTDOWN**

Limits waveform output to nodes in the specified subcircuit instance and their children.

### **Syntax**

```
.OPTION SIM_POSTDOWN=instance
```

### **Description**

Use this option with `.OPTION SIM_POSTTOP` and it takes precedence over `.OPTION SIM_POSTSKIP`.

Wildcards are supported.

### **Examples**

The following example outputs top, X1, X1.X4, X1.X4.X1, X1.X4.X2, and X2. (See [Figure 15 on page 664](#).)

```
.OPTION SIM_POSTTOP=2  
.OPTION SIM_POSTDOWN=X1.X4
```

### **See Also**

- [.OPTION SIM\\_POSTAT](#)
- [.OPTION SIM\\_POSTSKIP](#)
- [.OPTION SIM\\_POSTTOP](#)

---

## **.OPTION SIM\_POSTSCOPE**

Specifies the signal types to probe from within a scope.

### **Syntax**

```
.OPTION SIM_POSTSCOPE= net | port | all
```

### **Description**

Use this option to specify the signal types to probe from within a scope.

- `net`: Outputs only nets in the scope
- `port`: Outputs both nets and ports
- `all`: Outputs nets, ports, and global variables.

### **See Also**

[.OPTION POST](#)

[.OPTION SIM\\_POSTSKIP](#)

[.OPTION SIM\\_POSTTOP](#)

## **.OPTION SIM\_POSTSKIP**

Causes the SIM\_POSTTOP option to skip *subckt\_definition* instances.

### **Syntax**

```
.OPTION SIM_POSTSKIP=subckt_definition
```

### **Description**

Use this option to cause the SIM\_POSTTOP option to skip any instances and their children that are defined by the *subckt\_definition* parameter. To specify more than one subcircuit definition, issue this option once for each definition you want to skip. SIM\_POSTSKIP is available for both HSPICE and HSPICE RF. Wildcards are supported.

### **Examples**

The following example outputs top, and skips X2. X1 because they are instances of the ADD subcircuit. (See [Figure 15 on page 664](#).)

```
.OPTION SIM_POSTTOP=2  
.OPTION SIM_POSTSKIP=ADD
```

### **See Also**

[.OPTION SIM\\_POSTTOP](#)

---

## .OPTION SIM\_POSTTOP

Limits data written to your waveform file to data from only the top  $n$  level nodes.

### Syntax

```
.OPTION SIM_POSTTOP= $n$ 
```

### Description

Limits the data written to your waveform file to data from only the top  $n$  level nodes. SIM\_POSTAT is available for both HSPICE and HSPICE RF.

This option outputs instances from up to  $n$  levels deep.

- SIM\_POSTTOP=3: Outputs instances from 3 levels deep
- SIM\_POSTTOP=1: Outputs instances from only the top-level signals.

If you specify the PROBE option without specifying a SIM\_POSTTOP option HSPICE RF sets the SIM\_POSTTOP=0. HSPICE RF outputs all levels if you do not specify either the PROBE option or a SIM\_POSTTOP option.

Wildcards are not currently supported.

**Note:** You must specify the POST option to enable the waveform display interface.

SIM\_POSTTOP is equivalent POSTTOP used in HSPICE.

### Examples

Example 1 outputs top, X1, and X2. (See [Figure 15 on page 664.](#))

#### Example 1

```
.OPTION SIM_POSTTOP=2
```

The following example outputs top, X1, X2, and X4, X1 and X2. (See [Figure 15 on page 664.](#))

#### Example 2

```
.OPTION SIM_POSTTOP=2  
.OPTION SIM_POSTDOWN=X1.X4
```

### See Also

[.OPTION POST](#)  
[.OPTION PROBE](#)  
[.OPTION SIM\\_POSTSKIP](#)

---

## .OPTION SIM\_POWER\_ANALYSIS

Prints a list of signals matching the tolerance setting at a specified point in time.

### Syntax

```
.OPTION SIM_POWER_ANALYSIS="time_pointtol"  
.OPTION SIM_POWER_ANALYSIS="bottom time_pointtol"
```

---

Argument	Description
time_point	Time when HSPICE RF detects signals where the port current is larger than the tolerance value.
tol	Tolerance value for the signal defined in the .POWER command.
bottom	Signal at the lowest hierarchy level, also called a <i>leaf</i> subcircuit.

---

### Description

Use this option to print a list of signals matching the tolerance (*tol*) setting at a specified point in time.

The first syntax produces a list of signals that consume more current than *tol* at *time point*, in this format:

The second syntax produces the list of lowest-level signals, known as leaf subcircuits that consume more than *tol* at *time point*. The output is similar to this:

For additional information, see “[Power Analysis Output Format](#)” in the *HSPICE User Guide: RF Analysis*.

### Examples

In this example, print the names of leaf subcircuits that use more than 100uA at 100ns into the simulation are printed.

```
.OPTION SIM_POWER_ANALYSIS="bottom 100ns 100ua"  
.POWER VDD
```

### See Also

[.POWER](#)

---

## **.OPTION SIM\_POWER\_TOP**

Controls the number of hierarchy levels on which power analysis is performed.

### **Syntax**

```
.OPTION SIM_POWER_TOP=value
```

### **Description**

Use this option to control the number of hierarchy levels on which power analysis is performed.

By default, power analysis is performed on the top levels of hierarchy.

### **Examples**

In the following example, HSPICE RF produces `.POWER` command results for top-level and first-level subcircuits (the subcircuit children of the top-level subcircuits).

```
.OPTION SIM_POWER_TOP=2
```

### **See Also**

[.POWER](#)

## **.OPTION SIM\_POWERDC\_ACCURACY**

Increases the accuracy of operating point calculations for POWERDC analysis.

### **Syntax**

```
.OPTION SIM_POWERDC_ACCURACY=value
```

### **Description**

Use this option to increase the accuracy of operating point calculations for POWERDC analysis.

A higher *value* results in greater accuracy, but more time to complete the calculation.

### **See Also**

[.POWERDC](#)

[.OPTION SIM\\_POWERDC\\_HSPICE](#)

---

## **.OPTION SIM\_POWERDC\_HSPICE**

Increases the accuracy of operating point calculations for POWERDC analysis.

### **Syntax**

```
.OPTION SIM_POWERDC_HSPICE
```

### **Description**

Use this option to increase the accuracy of operating point calculations for POWERDC analysis.

### **See Also**

[.POWERDC](#)

[.OPTION SIM\\_POWERDC\\_ACCURACY](#)



## **.OPTION SIM\_POWERPOST**

Controls power analysis waveform dumping.

### **Syntax**

```
.OPTION SIM_POWERPOST=ON|OFF
```

### **Description**

Use this option to enable or disable power analysis waveform dumping.

### **See Also**

[.POWER](#)

---

## **.OPTION SIM\_POWERSTART**

Specifies a default start time for measuring signals during simulation.

### **Syntax**

```
.OPTION SIM_POWERSTART=time
```

### **Description**

Use this option with a `.POWER` command to specify a default start time for measuring signals during simulation. This default time applies to all signals that do not have their own `FROM` measurement time. This option together with the `.OPTION SIM_POWERSTOP` control the power measurement scope for an entire simulation.

### **Examples**

In this example, the scope for simulating the `x1.in` signal is from 10 ps to 90 ps.

```
.OPTION SIM_POWERSTART=10ps  
.OPTION SIM_POWERSTOP=90ps  
.power x1.in
```

### **See Also**

[.OPTION SIM\\_POWERSTOP](#)  
[.OPTION SIM\\_POWERSTART](#)

## **.OPTION SIM\_POWERSTOP**

Specifies a default stop time for measuring signals during simulation.

### **Syntax**

```
.OPTION SIM_POWERSTOP=time
```

### **Description**

Use this option with a `.POWER` command to specify a default stop time for measuring signals during simulation. This default time applies to all signals that do not have their own TO measurement time. This option together with the `.OPTION SIM_POWERSTART` control the power measurement scope for an entire simulation.

### **See Also**

[.OPTION SIM\\_POWERSTART](#)  
[.POWER](#)

---

## .OPTION SIM\_SPEF

Runs simulation with SPEF expansion of all nets from one or more SPEF files.

### Syntax

```
.OPTION SIM_SPEF="spec_filename"
```

### Description

Use this option to run simulation with SPEF expansion of all nets from one or more SPEF files.

You can repeat this option to include more SPEF files.

For additional information, see "[Post-Layout Back-Annotation](#)" in the *HSPICE User Guide: RF Analysis*.

### Examples

In this example, the *senseamp.spf* SPEF file is back-annotated to the sense amp circuit.

```
.OPTION SIM_SPEF = "senseamp.spf"
```

### See Also

[.OPTION SIM\\_SPEF\\_ACTIVE](#)  
[.OPTION SIM\\_SPEF\\_SCALEC](#)  
[.OPTION SIM\\_SPEF\\_SCALER](#)

---

## .OPTION SIM\_SPEF\_ACTIVE

Runs simulation with selective SPEF expansion of active nets from one or more DSPF files.

### Syntax

```
.OPTION SIM_SPEF_ACTIVE="active_node"
```

### Description

Use this option to run simulation with selective SPEF expansion of active nets from one or more DSPF files.

HSPICE RF performs a preliminary verification run to determine the activity of the nodes and generates two ASCII files: *active\_node.rc* and *active\_node.rcxt*. These files save all active node information in both Star-RC and Star-RCXT formats.

If an *active\_node* file is not generated from the preliminary run, no nets are expanded. Active nets are added to the file as they are identified in the subsequent transient simulation. A second simulation run using the same file and option causes only the nets listed in the *active\_node* file to be expanded. It is possible that activity changes are due to timing changes caused by expansion of the active nets. In this case, additional nets are listed in the *active\_node* file and a warning is issued.

By default, a node is considered active if the voltage varies by more than 0.1 V. You can use the `SIM_SPEF_VTOL` option to change this value.

HSPICE RF uses the *active\_node* file and the DSPF file with the ideal netlist to expand only the active portions of the circuit. If a net is latent, then HSPICE RF does not expand it, which saves memory and CPU time.

For additional information, see "[Selective Post-Layout Flow](#)" in the *HSPICE User Guide: RF Analysis*.

### See Also

[.OPTION SIM\\_SPEF\\_VTOL](#)

## **.OPTION SIM\_SPEF\_INSERTERROR**

Skips unmatched instances.

### **Syntax**

```
.OPTION SIM_SPEF_INSERTERROR=ON | OFF
```

### **Description**

Use this option to skip unmatched instances.

- ON: Skips unmatched instances.
- OFF: Does not skip unmatched instances.

For more information, see [“Additional Post-Layout Options”](#) in the *HSPICE User Guide: RF Analysis*.

## **.OPTION SIM\_SPEF\_LUMPCAPS**

Connects a lumped capacitor with a value equal to the net capacitance for instances missing in the hierarchical netlist.

### **Syntax**

```
.OPTION SIM_SPEF_LUMPCAPS=ON | OFF
```

### **Description**

Use this option to connect a lumped capacitor with a value equal to the net capacitance for instances missing in the hierarchical netlist.

- **ON**: Adds lumped capacitance while ignoring other net contents.
- **OFF**: Uses net contents.

For additional information, see “[Additional Post-Layout Options](#)” in the *HSPICE User Guide: RF Analysis*.

---

## **.OPTION SIM\_SPEF\_MAX\_ITER**

Specifies the maximum number of simulation runs for the second selective SPEF expansion pass.

### **Syntax**

```
.OPTION SIM_SPEF_MAX_ITER=value
```

### **Description**

Use this option to specify the maximum number of simulation runs for the second selective SPEF expansion pass.

The *value* parameter specifies the number of iterations for the second simulation run.

Some of the latent nets might turn active after the first iteration of the second simulation run. In this case:

- Re simulate the netlist to ensure the accuracy of the post-layout simulation.
- Use this option to set the maximum number of iterations for the second run. If the *active\_node* remains the same after the second simulation run, HSPICE RF ignores these options.

For additional information, see “[Selective Post-Layout Flow](#)” in the *HSPICE User Guide: RF Analysis*.

### **See Also**

[.OPTION SIM\\_SPEF\\_ACTIVE](#)

[.OPTION SIM\\_SPEF\\_VTOL](#)



## **.OPTION SIM\_SPEF\_PARVALUE**

Interprets triplet format *float:float:float* values in SPEF files as best:average:worst.

### **Syntax**

```
.OPTION SIM_SPEF_PARVALUE=1 | 2 | 3
```

### **Description**

Use this option to interpret triplet format *float:float:float* values in SPEF files as best:average:worst.

- SIM\_SPEF\_PARVALUE = 1: Use best.
- SIM\_SPEF\_PARVALUE = 2: Use average.
- SIM\_SPEF\_PARVALUE = 3: Use worst.

For further information, see “[Additional Post-Layout Options](#)” in the *HSPICE User Guide: RF Analysis*.

---

## **.OPTION SIM\_SPEF\_RAIL**

Controls whether power-net parasitics are back-annotated.

### **Syntax**

```
.OPTION SIM_SPEF_RAIL=ON | OFF
```

### **Description**

Use this option to control whether power-net parasitics are back-annotated.

- OFF: Do not back-annotate nets in a power rail.
- ON: Back-annotate nets in a power rail.

For further information, see “[Additional Post-Layout Options](#)” in the *HSPICE User Guide: RF Analysis*.

## **.OPTION SIM\_SPEF\_SCALEC**

Scales the capacitance values in a SPEF file for a standard SPEF expansion flow.

### **Syntax**

```
.OPTION SIM_SPEF_SCALEC=scaleC
```

### **Description**

Use this option to scale the capacitance values in a SPEF file for a standard SPEF expansion flow.

The *scaleC* parameter specifies the scale factor.

See “[Additional Post-Layout Options](#)” in the *HSPICE User Guide: RF Analysis*.

### **See Also**

[.OPTION SIM\\_SPEF\\_ACTIVE](#)

---

## .OPTION SIM\_SPEF\_SCALER

Scales the resistance values in a SPEF file for a standard SPEF expansion flow.

### Syntax

```
.OPTION SIM_SPEF_SCALER=scaleR
```

### Description

Use this option to scale the resistance values in a SPEF file for a standard SPEF expansion flow.

The *scaleR* parameter specifies the scale factor.

For more information, see [“Additional Post-Layout Options”](#) in the *HSPICE User Guide: RF Analysis*.

### See Also

[.OPTION SIM\\_SPEF\\_ACTIVE](#)

---

## .OPTION SIM\_SPEF\_VTOL

Specifies multiple SPEF active thresholds.

### Syntax

```
.OPTION SIM_SPEF_VTOL=value | scope1 scope2...  
+ scopen"
```

### Description

Use this option to specify multiple SPEF active thresholds.

- The *value* parameter specifies the tolerance of voltage change. This value should be relatively small compared to the operating range of the circuit, or smaller than the supply voltage.
- The *scopen* parameter can be a subcircuit definition that uses a prefix of "@" or a subcircuit instance.

HSPICE RF performs a second simulation run by using the *active\_node* file, the SPEF, and the hierarchical LVS ideal netlist to back-annotate only active portions of the circuit. If a net is latent, then HSPICE RF does not expand the net. This saves simulation runtime and memory.

By default, HSPICE RF performs only one iteration of the second simulation run. Use the *SIM\_SPEF\_MAX\_ITER* option to change it.

For additional information, see "[Selective Post-Layout Flow](#)" in the *HSPICE User Guide: RF Analysis*.

### See Also

[.OPTION SIM\\_SPEF\\_ACTIVE](#)  
[.OPTION SIM\\_SPEF\\_MAX\\_ITER](#)

---

## .OPTION SIM\_TG\_THETA

Controls the amount of second-order Gear method to mix with Trapezoidal integration for the hybrid TRAPGEAR method.

### Syntax

```
.OPTION SIM_TG_THETA= [0 | 1]
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### Description

Use this option to control the amount of second-order Gear (Gear-2) method to mix with Trapezoidal (TRAP) integration for the hybrid TRAPGEAR method.

The *value* parameter must specify a value between 0.0 and 1.0. The default is 0.1.

- SIM\_TG\_THETA=0 selects TRAP without Gear-2.
- SIM\_TG\_THETA=1 selects pure Gear-2.

### See Also

[.OPTION METHOD](#)

## **.OPTION SIM\_TRAP**

Changes the default `SIM_TG_THETA=0` so that `METHOD=TRAPGEAR` acts like `METHOD=TRAP`.

### **Syntax**

```
.OPTION SIM_TRAP=x
```

### **Description**

Use this option to change the default `SIM_TG_THETA=0` so that `METHOD=TRAPGEAR` acts like `METHOD=TRAP`.

The *x* parameter must specify a value between 0.0 and 1.0.

### **See Also**

[.OPTION METHOD](#)

[.OPTION SIM\\_TG\\_THETA](#)

## .OPTION SI\_SCALE\_SYMBOLS

Controls whether the scale factors are HSPICE attributes or International System of Units (SI) when case sensitivity is invoked.

### Syntax

SI\_SCALE\_SYMBOLS=0 | 1

**Default** Value if option is not specified in the netlist: 0

Value if option name is specified without a corresponding value: 1

### Description

SI\_SCALE\_SYMBOLS=1 changes the scaling factors from the HSPICE standard (default) to the International System of Units (SI) to enable you to use case sensitive scaling symbols. (Using the (=1) setting assures consistency with spice scale factors for downstream tools.)

**Note:** This option is enabled when case-sensitivity is on (-case 1).

Multiplying Factors	Description	.OPTION SI_SCALE_SYMBOLS=S			
		S=0, C=0 (default)	S=0, C=1 (Same as S=0, C=0)	S=1, C=1	S=1, C=0 Same as S=0, C=0
1e12	Tera	T, t		T, t	
1e9	Giga	G, g		G, g	
1e6	Mega	MEG, meg, X, x		M, MEG, meg, X, X	
1e3	Kilo	K, k		K, k	
1e-3	Milli	M or m		m	
25.4e-6	1,000(s) of an inch	MIL, mil		MIL, mil	
1e-6	Mico	U, u		U, u	
1e-9	Nano	N, n		N, n	
1e-12	Pico	P, p		P, p	
1e-15	Femto	F, f		F, f	
1e-18	Atto	A, a		A, a	

### See Also

[.OPTION PCB\\_SCALE\\_FORMAT](#)



## **.OPTION SLOPETOL**

Specifies the minimum value for breakpoint table entries in a piecewise linear (PWL) analysis.

### **Syntax**

```
.OPTION SLOPETOL=x
```

### **Description**

Use this option to specify the minimum value for breakpoint table entries in a piecewise linear (PWL) analysis. If the difference in the slopes of two consecutive PWL segments is less than the `SLOPETOL` value, HSPICE RF ignores the breakpoint for the point between the segments. Min value: 0; Max value: 2.

## **.OPTION SNACCURACY**

Sets and modifies the size of timesteps.

### **Syntax**

```
.OPTION SNACCURACY=integer
```

**Default**    10

### **Description**

Use this option to set and modify the size of timesteps. Larger values of `snaccuracy` result in a more accurate solution but might require more time points. Because Shooting-Newton must store derivative information at every time point, the memory requirements might be significant if the number of time points is very large. The maximum integer value is 50.

For additional information, see [SN Steady-State Time Domain Analysis](#) in the *HSPICE User Guide: RF Analysis*.

### **See Also**

[.OPTION SIM\\_ACCURACY](#)  
[.OPTION SNMAXITER \(or\) SN\\_MAXITER](#)

## **.OPTION SNCONTINUE**

Specifies whether to use the sweep solution from the previous simulation as the initial guess for the present simulation.

### **Syntax**

```
.OPTION SNCONTINUE= 0 | 1
```

**Default** 1

### **Description**

Use this option to specify whether to use the sweep solution from the previous simulation as the initial guess for the present simulation.

- SNCONTINUE=1: Use solution from previous simulation as the initial guess.
- SNCONTINUE=0: Start each simulation in a sweep from the DC solution.

### **See Also**

[.SN](#)

---

## **.OPTION SNMAXITER (or) SN\_MAXITER**

Sets the maximum number of iterations for a Shooting Newton analysis.

### **Syntax**

```
.OPTION SNMAXITER | SN_MAXITER=integer
```

### **Description**

Use this option to limit the number of SN iterations. For more information, see [Steady-State Shooting Newton Analysis](#) in the *HSPICE User Guide: RF Analysis*.

### **See Also**

[.SN](#)

## **.OPTION SOIQ0**

Invokes the body charge initialization (BQI) algorithm.

### **Syntax**

```
.OPTION SOIQ0=[0 | 1]
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to invoke the BQI algorithm for floating body SOI transistors. This option is to be used in conjunction with instance parameter `soiq0`.

The BQI algorithm allows users to specify a SOI device initial state for simulation to start with the initial state. The initial body charge can be provided by CFL function calls.

The BQI algorithm is applied to SOI models (Levels: 57, 60, and 70). For additional information, see [MOSFET Models \(BSIM\): Levels 47 through 72](#) in the *HSPICE Reference Manual: MOSFET Models*.

### **See Also**

- [.DC](#)
- [.OP](#)
- [.TRAN](#)

---

## .OPTION SPLIT\_DP

Enables the writing of multiple operating points in separate files.

### Syntax

```
.OPTION SPLIT_DP=0|1|2
```

**Default** Value if option is not specified in the netlist: 0

Value if option name is specified without a corresponding value: 1

### Description

Use this option in conjunction with `.OPTION OPFILE` when back annotating the operating point information for the Synopsys Custom Designer product.

- If `.OPTION SPLIT_DP=0`, HSPICE ignores the `SPLIT_DP` option.
- If `OPFILE=1` and `SPLIT_DP=0`, HSPICE writes operating point information for all time points specified in the `.OP` statement in a single `.dp0` file. If `OPFILE=1` and `SPLIT_DP=1`, HSPICE writes operating point information in a separate file for each time point specified in `.OP` statement.
- If `OPFILE=2.op0` and `.dp0` files are created together except that with sweep variables, `split_dp=2` permits you to create `*.op0` files corresponding to `*.dp0` files with sweep.

**Note:** Because the `split_dp=2` option was developed for the Custom Designer Environment, HSPICE users should avoid its use lest you create more `*.dp0` and `*.op0` files than you desire.

### Examples

The following command, these files below are returned:

```
.option opfile=1 split_dp=2
*.op0
*.dp0
*.op@timepoint@sweep_index
*.dp@timepoint@sweep_index
```

With `.op timepoint1 timepoint2...` in the netlist,

```
.tran '1n' '2n' start='0' sweep monte=10 firstrun=1
.option opfile=1 split_dp=1
```

The resulting files are generated:

```
*.op0  
*.dp0  
*.dp@timepoint@sweep_index
```

**See Also**

[.OPTION OPFILE](#)  
[.OPTION WDF](#)  
[.OP](#)

---

## **.OPTION SPMODEL**

Disables the previous `.OPTION VAMODEL`.

### **Syntax**

```
.OPTION SPMODEL [= name]
```

### **Description**

Use this option to disable a previously issued `VAMODEL` option. In this option, the name is the cell name that uses a SPICE definition. Each `SPMODEL` option can take no more than one name. Multiple names need multiple `SPMODEL` options.

### **Examples**

Example 1 disables the previous `.OPTIONVAMODEL` but has no effect on the other `VAMODEL` options if they are specified for the individual cells. For example, if `.OPTIONVAMODEL=vco` has been set, the `vco` cell uses the Verilog-A definition whenever it is available until `.OPTIONSPMODEL=vco` disables it.

```
.OPTION SPMODEL
```

This example disables the previous `.OPTIONVAMODEL=chargepump`, which causes all instantiations of `chargepump` to now use the subcircuit definition again.

```
.option spmodel=chargepump
```

### **See Also**

[.OPTION VAMODEL](#)



## **.OPTION STATFL**

Controls whether HSPICE creates a *.st0* file.

### **Syntax**

```
.OPTION STATFL=0 | 1
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to control whether HSPICE creates a *.st0* file.

- `STATFL=0` Outputs a *.st0* file.
- `STATFL=1` Suppresses the *.st0* file.

---

## .OPTION STRICT\_CHECK

Turns a subset of HSPICE netlist syntax warnings into terminal (abortive) syntax errors.

### Syntax

```
.OPTION STRICT_CHECK 0|1
```

**Default** Value if option is not specified in the netlist: 0  
Value if option name is specified without a corresponding value: 1

### Description

When enabled (set to 1), netlist conditions listed below will abort HSPICE with an error message. When disabled (set to 0), HSPICE will make assumptions and continue to run with only a warning message.

The following is a list of the messages controlled by `STRICT_CHECK`:

10001, 10002, 10003, 10004, 10008, 10011, 10012, 10013, 10018, 10019, 10020, 10021, 10047, 10048. For more information refer to [Warning Message Index \[10001-10076\]](#) located in the *HSPICE User Guide: Simulation and Analysis*, Chapter 34, [Warning/Error Messages](#).

### See Also

[.OPTION MESSAGE\\_LIMIT](#)

## **.OPTION SX\_FACTOR**

External shrink factor, only used for Ivthx calculation with the .IVTH command.

### **Syntax**

```
.IVTH model_name Ivth0=x DW=x DL=x  
.OPTION SX_factor=x
```

### **Description**

This option is only used with the IVTH command as shown in the Syntax section. It is restricted to use for ivthx calculation only.

### **See Also**

[.IVTH](#)

---

## **.OPTION SYMB**

Uses a symbolic operating point algorithm to get initial guesses before calculating operating points.

### **Syntax**

```
.OPTION SYMB=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to calculate the operating point. When `SYMB` is set to 1, HSPICE operates with a symbolic operating point algorithm to get initial guesses before calculating operating points. `SYMB` assumes the circuit is digital and assigns a low/high state to all nodes that set a reasonable initial voltage guess. This option improves DC convergence for oscillators, logic, and mixed-signal circuits.

`.OPTION SYMB` does not have any effect on the transient analysis if you set `UIC` in the `.TRAN` command.

## **.OPTION TIMERES**

Sets the minimum separation between breakpoint values for the breakpoint table.

### **Syntax**

```
.OPTION TIMERES=x
```

### **Description**

Use this option to set the minimum separation between breakpoint values for the breakpoint table. If two breakpoints are closer together in time than the `TIMERES` value, HSPICE enters only one of them in the breakpoint table.

---

## .OPTION TMIFLAG

Invokes the TSMC Model Interface (TMI) flow.

### Syntax

```
.OPTION TMIFLAG=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### Description

Use this option to invoke the TMI flow using proprietary TSMC model files and compiled libraries. The `TMIFLAG` option must equal 1 to enable a TMI flow.

Both the technology and API are jointly developed by Synopsys and TSMC. The TMI is a compact model with additional instance parameters and equations to support TSMC's extension of the standard BSIM4 model. Modeling API code is written in C and available in a compiled format for HSPICE and HSIM to link to during the simulation. TMI-required settings to invoke the flow and the location of a `.so` file are set by TSMC using their `TMIMODEL` parameter with the HSPICE TMI. The API also performs automatic platform selection on the `.so` file. Both HSPICE and HSIM provide the tool binaries and support the same `.so` file. To point to a TMI `.so` file location use the `.OPTION TMIPATH` command.

If `.OPTION TMIFLAG=1`, `.OPTION MACMOD` automatically equals 3 to enable the mapping of an instance name starting with "x" to "m" thus allowing TMI libraries to be shared for all vendor's simulator(s).

Use the existing HSPICE and HSIM commands to run the simulation. (Contact Synopsys Technical Support for further information.)

### See Also

[.OPTION TMIPATH](#)  
[.OPTION MACMOD](#)

## **.OPTION TMIPATH**

Points to a TMI \*.so (compiled library) file location.

### **Syntax**

```
.OPTION TMIPATH='tmifilename_dir'
```

### **Description**

Use this option to point to a TSMC Model Interface (TMI) \*.so file location. The path must be enclosed in single quotation marks. This option supports both relative and absolute paths.

### **Examples**

```
.option tmipath='tmi_v0d03_dir'
```

### **See Also**

[.OPTION TMIFLAG](#)

## **.OPTION TMIVERSION**

Specifies TMI version.

### **Syntax**

```
.OPTION TMIVERSION=1.0|2.0
```

**Default** 1.0

### **Description**

Use this option to select the TMI version:

- 1.0: Compatible with version TMI 1. HSPICE passes the model level and model type id (e.g., TMI\_MOS\_MODEL in tmiDef.h) to TMI for TMI model selection
- 2.0: Compatible with TMI 2 and CMC TMI. HSPICE passes model name id (e.g., TMI\_MOS\_BSIM4, TMI\_MOS\_PSP... defined in tmiDef.h) to TMI for TMI model selection.



## **.OPTION TNOM**

Sets the reference temperature for the simulation.

### **Syntax**

```
.OPTION TNOM=x
```

**Default** 25°C

### **Description**

Use this option to set the reference temperature for the HSPICE RF simulation. At this temperature, component derating is zero.

**Note:** The reference temperature defaults to the analysis temperature if you do not explicitly specify a reference temperature.

### **See Also**

[.TEMP](#) (or) [.TEMPERATURE](#)

## **.OPTION TRANFORHB**

Forces HB analysis to recognize or ignore specific V/I sources.

### **Syntax**

```
.OPTION TRANFORHB= [0 | 1]
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

This option forces HB analysis to recognize or ignore specific V/I sources.

- `TRANFORHB=1` : Forces HB analysis to recognize V/I sources that include SIN, PULSE, VMRF, and PWL transient descriptions, and to use them in analysis. However, if the source also has an HB description, analysis uses the HB description instead.
- `TRANFORHB=0` : Forces HB to ignore transient descriptions of V/I sources and to use only HB descriptions.

To override this option, specify `TRANFORHB` in the source description.

### **See Also**

[.HB](#)

## .OPTION TRCON

Controls the automatic convergence process of transient simulation.

### Syntax

```
.OPTION TRCON=[0 | 1 | 2]
```

**Default** 1

### Description

Use this option to control autoconvergence of transient simulation. If the circuit fails to converge using the default trapezoidal (TRAP) numerical integration method (for example because of trapezoidal oscillation), HSPICE sets the GEAR method to run the transient simulation again from `time=0`. This process is autoconvergence. If HSPICE fails to converge, an “internal timestep too small” error is issued.

- TRCON=0: Disables autoconvergence.
- TRCON=1: Enables autoconvergence for transient simulation only when the accumulated CPU time of the current simulation is less than 1 hour.
- TRCON=2: Enables autoconvergence with no restriction; in addition, a simulation enters into transient analysis without a converged operating point.

---

## **.OPTION TRTOL**

Estimates the amount of error introduced when the timestep algorithm truncates the Taylor series expansion.

### **Syntax**

```
.OPTION TRTOL=x
```

### **Description**

Use this option timestep algorithm for local truncation error (LVLTIM=2).HSPICE multiplies TRTOL by the internal timestep, which is generated by the timestep algorithm for the local truncation error. TRTOL reduces simulation time and maintains accuracy. It estimates the amount of error introduced when the algorithm truncates the Taylor series expansion. This error reflects the minimum timestep to reduce simulation time and maintain accuracy.

The range of TRTOL is 0.01 to 100; typical values are 1 to 10. If you set TRTOL to 1 (the minimum value), HSPICE uses a very small timestep. As you increase the TRTOL setting, the timestep size increases.

### **See Also**

[.OPTION LVLTIM](#)

## .OPTION UNWRAP

Displays phase results for AC analysis in unwrapped form.

### Syntax

```
.OPTION UNWRAP=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### Description

Use this option to display phase results for AC analysis in unwrapped form (with a continuous phase plot). HSPICE uses these results to accurately calculate group delay. HSPICE also uses unwrapped phase results to compute group delay, even if you do not set UNWRAP. By default, HSPICE calculates the unwrapped phase first and then converts it to wrapped phase. The convention is to normalize the phase output from -180 degrees to +180 degrees. A phase of -181 degrees is the same as a phase of +179 degrees. Below is an example to illustrate how HSPICE wraps the phase.

### Examples

#### Default Method (Without)

```
Freq Phase
3.16228k --> -167.7243
3.98107k --> 178.7844
```

#### If you use .OPTION UNWRAP = 1

```
3.16228k --> -167.7243
3.98107k --> -181.2156
```

If the phase value goes beyond -180, then it wraps to a positive value. At the frequency 3.98107kHz the actual value is -181.2156, but by default, it is wrapped to +178.7844.

HSPICE does the following calculation to wrap the phase:

```
-181.2156
+180.0000
-----
-1.2156
+180.0000
-1.2156
-----
178.7844
```

## .OPTION VAMODEL

Specifies that *name* is the cell name that uses a Verilog-A definition rather than the subcircuit definition when both exist (for use in HSPICE with Verilog-A).

### Syntax

```
.OPTION VAMODEL [=name]
```

### Description

Use this option to specify that *name* is the cell name that uses a Verilog-A definition rather than the subcircuit definition when both exist. Each `VAMODEL` option can take no more than one name. Multiple names need multiple `VAMODEL` options.

If a name is not provided for the `VAMODEL` option, HSPICE uses the Verilog-A definition whenever it is available. The `VAMODEL` option works on cell-based instances only. Instance-based overriding is not allowed.

### Examples

The following example specifies a Verilog-A definition for all instantiations of the cell `vco`.

#### Example 1

```
.option vamodel=vco
```

Example 2 specifies a Verilog-A definition for all instantiations of the `vco` and `chargepump` cells.

#### Example 2

```
.option vamodel=vco vamodel=chargepump
```

The following example instructs HSPICE to always use the Verilog-A definition whenever it is available.

#### Example 3

```
.option vamodel
```

## **.OPTION VERIFY**

Duplicates the LIST option.

### **Syntax**

```
.OPTION VERIFY= [0 | 1]
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option as an alias for the LIST option.

### **See Also**

[.OPTION LIST](#)

---

## **.OPTION VFLOOR**

Sets the minimum voltage to print in the output listing.

### **Syntax**

```
.OPTION VFLOOR=x
```

### **Description**

Use this option to set the minimum voltage to print in the output listing. All voltages lower than VFLOOR print as 0. Affects only the output listing; VNTOL (ABSV) sets the minimum voltage to use in a simulation.

### **See Also**

[.OPTION ABSV](#)  
[.OPTION VNTOL](#)



## **.OPTION VNTOL**

Duplicates the ABSV option.

### **Syntax**

```
.OPTION VNTOL=x
```

**Default** 5e-05

### **Description**

Use this option as an alias for the ABSV option. Min value: 0; Max value: 10.

### **See Also**

[.OPTION ABSV](#)

---

## .OPTION WACC

Activates the dynamic step control algorithm for a W-element transient analysis.

### Syntax

```
.OPTION WACC=x
```

**Default** -1 (variable, see below)

### Description

Use this option to activate the dynamic step control algorithm for a W-element transient analysis. `WACC` is a non-negative real value that can be set between 0.0 and 10.0. The `WACC` value influences a series of tolerances for W-element simulation. The default value of `WACC` is determined by HSPICE, according to the transmission line properties, such as loss and delay. Therefore, for different transmission line, the default `WACC` value is different. It is suggested that you not give a `WACC` value in the `.option` line, because it will give a constant value to all the transmission lines in the netlist. HSPICE assigns `WACC -1` if you do not set a `WACC` option, or if you set `.OPTION WACC`. When a value of 1 is specified, HSPICE assigns `WACC` a positive value. If a non-negative value is set in the `.option` line (`.OPTION WACC=XXX`), HSPICE uses the specified `WACC` value for all the W-elements. When `WACC=0`, HSPICE uses static breakpoint with the interval between each two as the transmission line system delay. Otherwise, when a positive value is set, W element uses dynamic time step control, which may improve the performance, especially for short delay cases. A large `WACC` value results in loose tolerance and bigger time steps, while small values result in tight tolerances and smaller time steps.

The following refers to HSPICE only: For cases containing IBIS, PKG, EBD, or ICM blocks, HSPICE turns `WACC` off automatically. If you want to use the dynamic time step control algorithm for IBIS-related cases, you must set it explicitly in the netlist. For example:

```
.option WACC $ Make HSPICE use automatically generated
WACC value for each W element
```

or

```
.option WACC=value $ Use this value for all the W
elements
```

### See Also

[Using Dynamic Time-Step Control](#) in the *HSPICE User Guide: Signal Integrity*.

---

## .OPTION WARN

Enables or turns off SOA voltage warning message.

### Syntax

`.OPTION WARN=1 | 0`

**Default** 1 or unspecified

---

Argument	Description
1 or unspecified	Turns <i>on</i> the warning message
0	Turns <i>off</i> the warning message

---

### Description

Use this option to enable or disable HSPICE warning messages when terminal voltages of a device (MOSFET, BJT, Diode, Resistor, Capacitor, etc...) exceed safe operating area (SOA).

The warning message is as follows:

```
**warning** (filename:line number): node_voltage_name =val
has exceeded node_voltage_name max =val
```

Control the number of warnings issued by using `.OPTION MAXWARNS=n`

### See Also

[.OPTION MAXWARNS](#)  
[Safe Operating Area \(SOA\) Warnings](#)

---

## **.OPTION WARN\_SEP**

Separates out warnings to a file, while suppressing them in the *\*.lis* file.

### **Syntax**

```
.OPTION WARN_SEP [0 | 1]
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Setting a value of 1 for this option separates error and warning messages from the *\*.lis* file into a separate file (*.warnlog*). This file reports error and warning message subheadings, contents, and summaries. This option also prints message types to the terminal.

### **See Also**

[.OPTION WARNLIMIT \(or\) .OPTION WARNLIM](#)  
[.OPTION LIS\\_NEW](#)

## **.OPTION WARNLIMIT (or) .OPTION WARNLIM**

Limits how many times certain warnings appear in the output listing.

### **Syntax**

```
.OPTION WARNLIMIT=n
```

### **Description**

Use this option to limit how many times the same warning appears in the output listing. This reduces the output listing file size. The *n* parameter specifies the maximum number of warnings for each warning type.

This limit applies to the following warning messages:

- MOSFET has negative conductance.
- Node conductance is zero.
- Saturation current is too small.
- Inductance or capacitance is too large.

### **See Also**

[.OPTION NOWARN](#)

[.OPTION MESSAGE\\_LIMIT](#)

## **.OPTION WAVE\_POP**

Enables setting of buffer flush interval for *.tr0* and *.wdf* files (HSPICE only).

### **Syntax**

```
.OPTION WAVE_POP=val
```

**Default** 0.1 (10%)

### **Description**

Sets waveform buffer flush interval as a percentage of the total simulation time. The value can be set from 0.001 to 1, where 0.001 is 1% and 1 is 100% of the total transient run time. `.OPTION WAVE_POP` values can also work when `.OPTION PSF` is set. If the option is not set, then the waveform buffer will be flushed at every 10% of the total simulation time.

### **Examples**

In this example, the waveform buffer is flushed at every 5% of the total simulation time.

```
.OPTION WAVE_POP=0.05
```

---

## .OPTION WDELAYOPT

Globally applies the DELAYOPT keyword to a W-element transient analysis.

### Syntax

```
.OPTION WDELAYOPT=[0 | 1 | 2 | 3]
```

**Default** 0

### Description

Use this option as a global option which applies to all W-elements in a netlist.

.OPTION WDELAYOPT can be overridden by the DELAYOPT keyword for a specified W-element.

- In cases where WDELAYOPT is set in the .OPTION and the DELAYOPT keyword is not specially set for Wxxx, the WDELAYOPT keyword is auto-set for Wxxx.
- In cases where the DELAYOPT keyword is already set for Wxxx, .OPTION WDELAYOPT is overridden for the Wxxx.
- In cases where neither .OPTION WDELAYOPT nor the DELAYOPT keyword is set, the DELAYOPT keyword defaults to 0.

.OPTION WDELAYOPT helps construct a W-element transient (recursive convolution) model with a higher level of accuracy. By specifying this option, you can add the DELAYOPT keyword to the W-element instance line.

You can use DELAYOPT=0 | 1 | 2 to deactivate, activate, and automatically determine, respectively.

Use DELAYOPT=3 to achieve a level of accuracy up to a tens of GHz operation and involve harmonics up to THz order. With this option, line length limits are removed, which frees the simulation from segmenting and allows independence in the behavior of the RISETIME option setting. A setting of WDELAYOPT=3 automatically detects whether or not frequency-dependent phenomena need to be recorded, which makes it identical to the DELAYOPT=0 setting if it produces a high enough accuracy.

See [Use DELAYOPT Keyword for Higher Frequency Ranges](#) in the *HSPICE User Guide: Signal Integrity*

### See Also

[.OPTION WINCLUDEGDIMAG](#)  
[.OPTION RISETIME](#) (or) [.OPTION RISETI](#)

---

## .OPTION WDF

Enables HSPICE to produce waveform files in WDF format.

### Syntax

```
.OPTION WDF=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### Description

Use this option to enable HSPICE to produce waveform files in WDF format. The WDF (Waveform Data File) format is a proprietary waveform storage format. The WDF format compresses analog and logic waveform data, and facilitates fast waveform access for large data files. The compression scheme can be lossy or lossless (default). Use this option with the `.PRINT` or `.PROBE` command.

- `.option WDF=0`—Disables this option
- `.option WDF` or `.option WDF=1`—Enables HSPICE to produce the waveform file in WDF format

For the WDF waveform file, HSPICE automatically appends `_wdf` into the output file root name to specify that it is in WDF format. The file names appear as: `*_wdf.tr#`, `*_wdf.sw#`, or `*_wdf.ac#`.

For example, the WDF waveform output file will be named: `design_wdf.tr0`.

The WDF format is available to HSPICE RF for `.AC`, `.DC`, and `.TRAN` analyses.

When the netlist contains `.option wdf=1` and a `.tran` analysis statement (with no `.op` statement in the netlist file), HSPICE creates the following output files. See examples below.

- `.op0` — dc node voltage and dc operating points.
- `.op1` — transient voltage and transient operating points for the transient end time.



## Examples

*Example 1* In this example, HSPICE creates these output files:  
*input\_wdf.op0*  
*input.dp0@timepoint@spweep\_index*

```
.option WDF=1 opfile=1 split_dp=1  
.tran '1n' '2n' start='0' sweep monte=10 firstrun=1  
.op All 0.5n 1n 1.5n
```

*Example 2* In this example, HSPICE outputs:  
*input\_wdf.op0@timepoint@sweep\_index*  
*input.dp0@timepoint@spweep\_index*

```
.option WDF=1 opfile=1 split_dp=2
```

## See Also

- [.PRINT](#)
- [.PROBE](#)
- [.OPTION OPFILE](#)
- [.OPTION SPLIT\\_DP](#)

---

## .OPTION WINCLUDEGDIMAG

Globally activates the complex dielectric loss model in *W*-element analysis.

### Syntax

```
.OPTION WINCLUDEGDIMAG= [YES | NO]
```

**Default** NO

### Description

Use this option as a global option to activate the complex dielectric loss model for all *W*-elements a netlist by introducing an imaginary term of the skin effect to be considered. If `WINCLUDEGDIMAG=YES` and there is no `wp` input, the *W*-element regards the `Gd` matrix as the conventional model and then automatically extracts constants for the complex dielectric model.

The `.OPTION WINCLUDEGDIMAG` operates with the `.OPTION WDELAYOPT` option.

- In cases where `WINCLUDEGDIMAG` is set in the `.OPTION` and the `INCLUDEGDIMAG` keyword is not specially set for `Wxxx`, the `INCLUDEGDIMAG` is auto-set for `Wxxx`.
- In cases where the `INCLUDEGDIMAG` keyword is already set for `Wxxx`, `.OPTION WINCLUDEGDIMAG` is overridden for the `Wxxx`.
- In cases where neither `.OPTION WINCLUDEGDIMAG` nor the `INCLUDEGDIMAG` keyword is set, the `INCLUDEGDIMAG` keyword defaults to `NO`.

For details about the `INCLUDEGDIMAG` keyword, see [Fitting Procedure Triggered by INCLUDEGDIMAG Keyword](#) in the *HSPICE USER GUIDE: Signal Integrity*.

### See Also

[.OPTION WDELAYOPT](#)

[.OPTION RISETIME](#) (or) [.OPTION RISETI](#)

## **.OPTION WL**

Reverses the order of the `VSIZE` MOS element.

### **Syntax**

```
.OPTION WL=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 Value if option name is specified without a corresponding value: 1

### **Description**

Use this option to reverse the order of the MOS element `VSIZE`. The default order is length-width; this option changes the order to width-length.

## **.OPTION WNFLAG**

Controls whether bin is selected based on w or w/nf.

### **Syntax**

```
.OPTION WNFLAG=[0|1]
```

### **Description**

Use this option to control whether HSPICE selects the bin based on the total device width (`WNFLAG=0`) or based on the width of one finger of a multi fingered device (`WNFLAG=1`).

For devices which are using a BSIM4 model, an element parameter `wnflag=[0|1]` can be set, with the same effect as the option, and this element parameter overrides then the option setting on an element basis.

### **Examples**

For All Levels:

```
.option wnflag  
M1 out in vdd vdd pmos w=10u l=1u nf=5
```

For BSIM4 models only:

```
M1 out in vdd vdd pmos w=10u l=1u nf=5 wnflag=1
```

---

## .OPTION XDTEMP

Defines how HSPICE interprets the DTEMP parameter.

### Syntax

```
.OPTION XDTEMP=0 | 1
```

**Default** Value if option is not specified in the netlist: 0 (user-defined parameter) Value if option name is specified without a corresponding value: 1

### Description

Use this option to define how HSPICE interprets the DTEMP parameter, where *value* is either:

- 0: Indicates a user-defined parameter
- 1: Indicates a temperature difference parameter

If you set .OPTION XDTEMP to 1, HSPICE adds the DTEMP value in the subcircuit call command to all elements within the subcircuit that use the DTEMP keyword syntax. The DTEMP parameter is cumulative throughout the design hierarchy.

### Examples

```
.OPTION XDTEMP
X1 2 0 SUB1 DTEMP=2
.SUBCKT SUB1 A B
R1 A B 1K DTEMP=3
C1 A B 1P
X2 A B sub2 DTEMP=4
.ENDS
.SUBCKT SUB2 A B
R2 A B 1K
.ENDS
```

In this example:

- X1 sets a temperature difference (2 degrees Celsius) between the elements within the subcircuit SUB1.
- X2 (a subcircuit instance of X1) sets a temperature difference by the DTEMP value of both X1 and X2 (2+4=6 degrees Celsius) between the elements within the SUB2 subcircuit. The DTEMP value of each element in this example is:

**Chapter 3: HSPICE and RF Netlist Simulation Control Options**  
**.OPTION XDTEMP**

```
Elements DTEMP Value (Celsius)
X1 2
X1.R1 2+3 =5
X1.C1 2
X2 2+4=6
X2.R2 6
```

## **.OPTION (X0R,X0I)**

The first of three complex starting-trial points in the Muller algorithm used in Pole/Zero analysis.

### **Syntax**

`.OPTION (X0R,X0I) = x,x`

**Default**    `X0R=-1.23456e6`    `X0I=0.0`

### **Description**

Use this option in Pole/Zero analysis if you need to change scale factors and modify the initial Muller points, (X0R, X0I), (X1R, X1I) and (X2R, X2I). HSPICE multiplies these initial points, and FMAX, by FSCAL.

Scale factors must satisfy the following relations:  $GSCAL = CSCAL \cdot FSCAL$

$$GSCAL = \frac{1}{LSCAL \cdot FSCAL}$$

### **See Also**

- [.OPTION CSCAL](#)
- [.OPTION FMAX](#)
- [.OPTION FSCAL](#)
- [.OPTION GSCAL](#)
- [.OPTION ITLPZ](#)
- [.OPTION LSCAL](#)
- [.OPTION PZABS](#)
- [.OPTION PZTOL](#)
- [.PZ](#)

---

## **.OPTION (X1R,X1I)**

The second of three complex starting-trial points in the Muller algorithm used in Pole/Zero analysis.

### **Syntax**

```
.OPTION (X1R,X1I) = x,x
```

**Default** X1R=1.23456e5 X1I=0.0

### **Description**

Use this option in Pole/Zero analysis if you need to change scale factors and modify the initial Muller points, (X0R, X0I), (X1R, X1I) and (X2R, X2I). HSPICE multiplies these initial points, and FMAX, by FSCAL.

Scale factors must satisfy the following relations:

$$GSCAL = CSCAL \cdot FSCAL$$

$$GSCAL = \frac{1}{LSCAL \cdot FSCAL}$$

### **See Also**

- [.OPTION CSCAL](#)
- [.OPTION FMAX](#)
- [.OPTION FSCAL](#)
- [.OPTION GSCAL](#)
- [.OPTION ITLPZ](#)
- [.OPTION LSCAL](#)
- [.OPTION PZABS](#)
- [.OPTION PZTOL](#)
- [.PZ](#)



## **.OPTION (X2R,X2I)**

The third of three complex starting-trial points in the Muller algorithm used in Pole/Zero analysis.

### **Syntax**

`.OPTION (X2R,X2I) = x,x`

**Default**    `X2R=+1.23456e6 X2I=0.0`

### **Description**

Use this option in Pole/Zero analysis if you need to change scale factors and modify the initial Muller points, (X0R, X0I), (X1R, X1I) and (X2R, X2I). HSPICE multiplies these initial points, and FMAX, by FSCAL.

Scale factors must satisfy the following relations:  $GSCAL = CSCAL \cdot FSCAL$

$$GSCAL = \frac{1}{LSCAL \cdot FSCAL}$$

### **See Also**

- [.OPTION CSCAL](#)
- [.OPTION FMAX](#)
- [.OPTION FSCAL](#)
- [.OPTION GSCAL](#)
- [.OPTION ITLPZ](#)
- [.OPTION LSCAL](#)
- [.OPTION PZABS](#)
- [.OPTION PZTOL](#)
- [.PZ](#)

---

## .VARIATION Block Control Options

The following options can be applied when doing .VARIATION analysis. Note that no leading period is allowed with Variation Block control options.

### Syntax

```
Option Normal_Limit=[1|2|3|4]
Option Ignore_Variation_Block=Yes
Option Ignore_Local_Variation=Yes
Option Ignore_Global_Variation=Yes
Option Ignore_Spatial_Variation=Yes
Option Ignore_Interconnect_Variation=Yes
Option Output_Sigma_Value=Value
Option Vary_Only Subckts=SubcktList
Option Do_Not_Vary Subckts=SubcktList
Option Add_Variation=Yes
```

### Monte Carlo-Specific Options Using the Variation Block

```
Option Random_Generator = [Default | MSG]
Option Stream =[x | Random | Default]
Option Use_Agauss_Format = Yes|No (Default: Yes)
Option Normal_Limit=Value
Option Output_Sigma_Value=Value
Option Print_Only Subckts=SubcktList
Option Do_Not_Print Subckts=SubcktList
```

### Description

The following describes the available options:

- Option Use\_Agauss\_Format=Yes Allows use of Gaussian sampling methods as well as advanced sampling formats in a Variation Block.
- Option Normal\_Limit=[1|2|3|4] Limits the range of the Normal distributions. This option allows a foundry to limit the perturbations to parameter ranges where a model is still valid.

Where:

- 1: Infinity
- 2: Specifies no sampling value > 5 within 1000000 index
- 3: If a negative value is set it is automatically reset to 4
- 4: (Default) Numbers in the range +/- 4  $\sigma$  are generated. The range allowed is 0.1 to 20.

- Option `Ignore_Variation_Block=Yes` Ignores the Variation Block and executes earlier style variations (traditional Monte Carlo analysis). By default, the contents of the variation block are executed and other definitions (AGAUSS, GAUSS, AUNIF, UNIF, LOT, and DEV) are ignored. Previous methods of specifying variations on parameters and models are not compatible with the Variation Block. By default, the contents of the Variation Block are used and all other specifications are ignored. Thus no changes are required in existing netlists other than adding the Variation Block.
- Option `Ignore_Local_Variation=Yes` Excludes effects of local variations in simulation. Default is No.
- Option `Ignore_Global_Variation=Yes` Excludes effects of global variations in simulation. Default is No.
- Option `Ignore_Spatial_Variation=Yes` Excludes effects of spatial variations in simulation. Default is No.
- Option `Ignore_Interconnect_Variation=Yes` Excludes effects of interconnect variations in simulation. Default is No. (See [Interconnect Variation in Star-RC with the HSPICE Flow.](#))
- Option `Output_Sigma_Value=Value` Use to specify the sigma value of the results of Monte Carlo, DCMATCH, and ACMATCH analyses. Default is 1, range is 1 to 10. Note that this option only changes the output listings and that the input sigma is not affected.
- Option `Vary_Only Subckts=SubcktList` Use either this option to limit variation to the specified subcircuits or the one below, but not both. Actual subcircuit names are specified here (not the hierarchical names).
- Option `Do_Not_Vary Subckts=SubcktList` Excludes variation on the specified subcircuits. Use either this option to limit variation to the specified subcircuits or the one above, *but not both*. Actual subcircuit names are specified here (not the hierarchical names).
- Option `Screening_Method = Pearson|Spearman` HSPICE calculates the variables screened by importance using the Pearson or Spearman algorithm. Default: `Pearson`.

### See Also

[Analyzing Variability and Using the Variation Block](#)  
[Monte Carlo Analysis Using the Variation Block Flow](#)

---

## .DESIGN\_EXPLORATION Block Control Options

The following options can be applied when doing `.DESIGN_EXPLORATION` analysis. Note that no leading period is allowed with variation control options:

### Syntax

```
Option Explore_only Subckts= SubcktList
Option Do_not_explore Subckts= SubcktList
Option Export=yes|no
Option Exploration_method=external Block_name=Block_name
Option Ignore_exploration= yes|no
Option Secondary_param= yes|no
```

### Description

The Design Exploration control options are described below:

- `Option Explore_only Subckts= SubcktList` This command is executed hierarchically—the specified subcircuits and all instantiated subcircuits and elements underneath are affected. Thus, if an inverter with name `INV1` is placed in a digital control block called `DIGITAL` and in an analog block `ANALOG`, and `Option Explore_only Subckts = ANALOG`, then the perturbations only affect the `INV1` in the analog block. You must create a new inverter, `INV1analog`, with the new device sizes.
- `Option Do_not_explore Subckts= SubcktList` Excludes listed subcircuits.
- `Option Export=yes|no` If `yes`, exports extraction data and runs one simulation with the original netlist. If `no` (default), runs a simulation with Exploration data.
- `Option Exploration_method=external Block_name=Block_name` The `Block_name` is the same as the name specified in the `.DATA` block; HSPICE will sweep the row content with the `EXCommandexplore`.
- `Option Ignore_exploration= yes|no` (Default=`no`) HSPICE ignores the content in the `design_exploration` block, when `Ignore_exploration=yes`.
- `Option Secondary_param= yes|no` (Default=`no`) If `Secondary_param=yes`, HSPICE exports the MOSFET secondary instance parameters to a `*.mex` file (created when `option export=yes`), and also permits the secondary parameters to be imported as a column header in the `.DATA` block (`option export=no`).

**See Also**

[.DESIGN\\_EXPLORATION](#)  
Exploration Block

**Chapter 3: HSPICE and RF Netlist Simulation Control Options**  
.DESIGN\_EXPLORATION Block Control Options

## Digital Vector File Commands

---

*Contains an alphabetical listing of the commands you can use in a digital vector file.*

You can use the following HSPICE/HSPICE RF commands in a digital vector file.

ENABLE	SLOPE	VIH
IDELAY	TDELAY	VIL
IO	TFALL	VNAME
ODELAY	TRISE	VOH
OUT or OUTZ	TRIZ	VOL
PERIOD	TSKIP	VREF
RADIX	TUNIT	VTH

For additional information on vector file usage, see [Specifying a Digital Vector File and Mixed Mode Stimuli](#) in the *HSPICE User Guide: Simulation and Analysis*.

---

## ENABLE

Specifies the controlling signal(s) for bidirectional signals.

### Syntax

`ENABLE controlling_signalname mask`

---

ArgumentArgument	DescriptionDescription
<code>controlling_signalname</code>	Controlling signal for bidirectional signals. Must be an input signal with a radix of 1. The bidirectional signals become output when the controlling signal is at state 1 (or high). To reverse this default control logic, start the control signal name with a tilde (~).
<code>mask</code>	Defines the bidirectional signals to which ENABLE applies.

---

### Description

Use this command to specify the controlling signal(s) for bidirectional signals. All bidirectional signals require an `ENABLE` command. If you specify more than one `ENABLE` command, the last command overrides the previous command and HSPICE issues a warning message:

```
[Warning]:[line 6] resetting enable signal to WENB for  
bit 'XYZ'
```

### Examples

```
radix 144  
io ibb  
vname a x[[3:0]] y[[3:0]]  
enable a 0 F 0  
enable ~a 0 0 F
```

In this example, the `x` and `y` signals are bidirectional as defined by the `b` in the `io` line.

- The first enable command indicates that `x` (as defined by the position of `F`) becomes output when the `a` signal is 1.
- The second enable specifies that the `y` bidirectional bus becomes output when the `a` signal is 0.



## IDELAY

Defines an input delay time for bidirectional signals.

### Syntax

```
IDELAY delay_value [mask]
```

Argument	Description
<code>delay_value</code>	Time delay to apply to the signals.
<code>mask</code>	Signals to which the delay applies. If you do not provide a <i>mask</i> value, the delay value applies to all signals.

### Description

Use this command to define an input delay time for bidirectional signals relative to the absolute time of each row in the Tabular Data section. HSPICE ignores IDELAY settings on output signals and issues a warning message.

You can specify more than one TDELAY, IDELAY, or ODELAY command.

- If you apply more than one TDELAY (IDELAY, ODELAY) command to a signal, the last command overrides the previous commands and HSPICE or HSPICE RF issues a warning.
- If you do not specify the signal delays in a TDELAY, IDELAY, or ODELAY command, HSPICE or HSPICE RF defaults to zero.

### Examples

```
RADIX 1 1 4 1234 11111111
IO i i o iiib iiiiiiiii
VNAME V1 V2 VX[[3:0]] V4 V5[[1:0]] V6[[0:2]] V7[[0:3]]
+ V8 V9 V10 V11 V12 V13 V14 V15
TDELAY 1.0
TDELAY -1.2 0 1 F 0000 00000000
TDELAY 1.5 0 0 0 1370 00000000
IDELAY 2.0 0 0 0 000F 00000000
ODELAY 3.0 0 0 0 000F 00000000
```

This example does not specify the TUNIT command so HSPICE or HSPICE RF uses the default, ns, as the time unit for this example. The first TDELAY command indicates that all signals have the same delay time of 1.0ns. Subsequent TDELAY, IDELAY, or ODELAY commands overrule the delay time of some signals.

## Chapter 4: Digital Vector File Commands

### IDELAY

- The delay time for the V2 and Vx signals is -1.2.
- The delay time for the V4, V5[0:1], and V6[0:2] signals is 1.5.
- The input delay time for the V7[0:3] signals is 2.0, and the output delay time is 3.0.

### See Also

[ODELAY](#)

[TDELAY](#)

[TUNIT](#)

---

# IO

Defines the type for each vector: input, bidirectional, output, or unused.

## Syntax

```
IO I | O | B | U      [I | O | B | U ...]
```

---

Argument	Description
i	Input that HSPICE uses to stimulate the circuit.
o	Expected output that HSPICE compares with the simulated outputs.
b	Bidirectional vector.
u	Unused vector that HSPICE ignores.

---

## Description

Use this command to define the type for each vector. The line starts with the `IO` keyword followed by a string of `i`, `b`, `o`, or `u` definitions. These definitions indicate whether each corresponding vector is an input (`i`), bidirectional (`b`), output (`o`), or unused (`u`) vector.

- If you do not specify the `IO` command, HSPICE or HSPICE RF assumes that all signals are input signals.
- If you define more than one `IO` command, the last command overrides previous commands.

## Examples

```
io i i i bbbb iiiioouu
```

---

## ODELAY

Defines an output delay time for bidirectional signals.

### Syntax

```
ODELAY delay_value [mask]
```

---

Argument	Description
<i>delay_value</i>	Time delay to apply to the signals.
<i>mask</i>	Signals to which the delay applies. If you do not provide a <i>mask</i> value, the delay value applies to all signals.

---

### Description

Use this command to define an output delay time for bidirectional signals relative to the absolute time of each row in the Tabular Data section.

HSPICE ignores ODELAY settings on input signals and issues a warning message.

You can specify more than one TDELAY, IDELAY, or ODELAY command.

- If you apply more than one TDELAY (IDELAY, ODELAY) command to a signal, the last command overrides the previous commands and HSPICE issues a warning.
- If you do not specify the signal delays in a TDELAY, IDELAY, or ODELAY command, HSPICE defaults to zero.

### Examples

```
RADIX 1 1 4 1234 11111111  
IO i i o iiib iiiiiiiii  
VNAME V1 V2 VX[[3:0]] V4 V5[[1:0]] V6[[0:2]] V7[[0:3]]  
+ V8 V9 V10 V11 V12 V13 V14 V15  
TDELAY 1.0  
TDELAY -1.2 0 1 F 0000 00000000  
TDELAY 1.5 0 0 0 1370 00000000  
IDELAY 2.0 0 0 0 000F 00000000  
ODELAY 3.0 0 0 0 000F 00000000
```

This example does not specify the TUNIT command so HSPICE or HSPICE RF uses the default, ns, as the time unit for this example. The first TDELAY command indicates that all signals have the same delay time of 1.0ns.

Subsequent TDELAY, IDELAY, or ODELAY commands overrule the delay time of some signals.

- The delay time for the V2 and Vx signals is -1.2.
- The delay time for the V4, V5[0:1], and V6[0:2] signals is 1.5.
- The input delay time for the V7[0:3] signals is 2.0 and the output delay time is 3.0.

**See Also**

[IDELAY](#)  
[TDELAY](#)  
[TUNIT](#)

---

## OUT or OUTZ

Specifies output resistance for each signal for which the mask applies. `OUT` and `OUTZ` are equivalent.

### Syntax

```
OUT output_resistance [mask]
```

---

Argument	Description
<code>output_resistance</code>	Output resistance for an input signal. The default is 0.
<code>mask</code>	Signals to which the output resistance applies. If you do not provide a <i>mask</i> value, the output resistance value applies to all input signals.

---

### Description

The `OUT` and `OUTZ` keywords are equivalent: use these commands to specify output resistance for each signal (for which the mask applies). `OUT` or `OUTZ` applies to input signals only.

- If you do not specify the output resistance of a signal in an `OUT` (or `OUTZ`) command, HSPICE uses the default (zero).
- If you specify more than one `OUT` (or `OUTZ`) command for a signal, the last command overrides the previous commands and HSPICE issues a warning message.

The `OUT` (or `OUTZ`) commands have no effect on the expected output signals.

### Examples

```
OUT 15.1  
OUT 150 1 1 1 0000 00000000  
OUTZ 50.5 0 0 0 137F 00000000
```

The first `OUT` command in this example creates a 15.1 ohm resistor to place in series with all vector inputs. The next `OUT` command sets the resistance to 150 ohms for vectors 1 to 3. The `OUTZ` command changes the resistance to 50.5 ohms for vectors 4 through 7.

---

## PERIOD

Defines the time interval for the Tabular Data section.

### Syntax

```
PERIOD time_interval
```

---

Argument	Description
<code>time_interval</code>	Time interval for the Tabular Data.

---

### Description

Use this command to define the time interval for the Tabular Data section. You do not need to specify the absolute time at every time point. If you use a `PERIOD` command without the `TSKIP` command, the Tabular Data section contains only signal values, not absolute times. The `TUNIT` command defines the time unit of the `PERIOD`.

### Examples

```
radix 1111 1111
period 10
1000 1000
1100 1100
1010 1001
```

- The first row of the tabular data (1000 1000) is at time 0ns.
- The second row (1100 1100) is at 10ns.
- The third row (1010 1001) is at 20ns.

### See Also

[TSKIP](#)  
[TUNIT](#)

## RADIX

Specifies the number of bits associated with each vector.

### Syntax

`RADIX number_of_bits [number_of_bits...]`

Argument	Description
number_of_bits	Specifies the number of bits in one vector in the digital vector file. You must include a separate <i>number_of_bits</i> argument in the RADIX command for each vector listed in the file.

### Description

Use this command to specify the number of bits associated with each vector. Valid values for the number of bits range from 1 to 4.

A digital vector file must contain only one RADIX command and it must be the first non-comment line in the file.

Table 2 Valid Values for the RADIX command

# bits	Radix	Number System	Valid Digits
1	2	Binary	0, 1
2	4	–	0 – 3
3	8	Octal	0 – 7
4	16	Hexadecimal	0 – F

### Examples

```
; start of Vector Pattern Definition section
RADIX 1 1 4 1234 1111 1111
VNAME A B C[[3:0]] I9 I[[8:7]] I[[6:4]] I[[3:0]] 07 06 05 04
+ 03 02 01 00
IO I I I IIII OOOO OOOO
```

This example illustrates two 1-bit signals followed by a 4-bit signal, followed by one each 1-bit, 2-bit, 3-bit, and 4-bit signals, and finally eight 1-bit signals.



---

## SLOPE

Specifies the rise/fall time for the input signal.

### Syntax

```
SLOPE [input_rise_time | input_fall_time] [mask]
```

---

Argument	Description
<code>input_rise_time</code>	Rise time of the input signal.
<code>input_fall_time</code>	Fall time of the input signal.
<code>mask</code>	Name of a signal to which the SLOPE command applies. If you do not specify a <i>mask</i> value, the SLOPE command applies to all signals.

---

### Description

Use this command to specify the rise/fall time for the input signal. Use the TUNIT command to define the time unit for this command.

- If you do not specify the SLOPE command, the default slope value is 0.1 ns.
- If you specify more than one SLOPE command, the last command overrides the previous commands and HSPICE or HSPICE RF issues a warning message.

The SLOPE command has no effect on the expected output signals. You can specify the optional TRISE and TFALL commands to overrule the rise time and fall time of a signal.

### Examples

In the following example, the rising and falling times of all signals are 1.2 ns.

#### *Example 1*

```
SLOPE 1.2
```

In the following example, the rising/falling time is 1.1 ns for the first, second, sixth, and seventh signals.

#### *Example 2*

```
SLOPE 1.1 1100 0110
```

**Chapter 4: Digital Vector File Commands**  
SLOPE

**See Also**

TFALL  
TRISE  
TUNIT

## TDELAY

Defines the delay time for both input and output signals in the Tabular Data section.

### Syntax

TDELAY *delay\_value* [*mask*]

Argument	Description
<i>delay_value</i>	Time delay to apply to the signals.
<i>mask</i>	Signals to which the delay applies. If you do not provide a <i>mask</i> value, the delay value applies to all signals.

### Description

Use this command to define the delay time of both input and output signals relative to the absolute time of each row in the Tabular Data section.

You can specify more than one TDELAY, IDELAY, or ODELAY command.

- If you apply more than one TDELAY (IDELAY, ODELAY) command to a signal, the last command overrides the previous commands and HSPICE or HSPICE RF issues a warning.
- If you do not specify the signal delays in a TDELAY, IDELAY, or ODELAY command, HSPICE or HSPICE RF defaults to zero.

### Examples

```
RADIX 1 1 4 1234 11111111
IO i i o iiib iiiiiiiii
VNAME V1 V2 VX[[3:0]] V4 V5[[1:0]] V6[[0:2]] V7[[0:3]]
+ V8 V9 V10 V11 V12 V13 V14 V15
TDELAY 1.0
TDELAY -1.2 0 1 F 0000 00000000
TDELAY 1.5 0 0 0 1370 00000000
IDELAY 2.0 0 0 0 000F 00000000
ODELAY 3.0 0 0 0 000F 00000000
```

This example does not specify the TUNIT command so HSPICE or HSPICE RF uses the default, ns, as the time unit for this example. The first TDELAY command indicates that all signals have the same delay time of 1.0ns. Subsequent TDELAY, IDELAY, or ODELAY commands overrule the delay time of some signals.

## Chapter 4: Digital Vector File Commands

### TDELAY

- The delay time for the V2 and Vx signals is -1.2.
- The delay time for the V4, V5[0:1], and V6[0:2] signals is 1.5.
- The input delay time for the V7[0:3] signals is 2.0, and the output delay time is 3.0.

### See Also

[IDELAY](#)  
[ODELAY](#)  
[TUNIT](#)

---

## TFALL

Specifies the fall time of each input signal for which the mask applies.

### Syntax

```
TFALL input_fall_time [mask]
```

---

Argument	Description
<code>input_fall_time</code>	Fall time of the input signal.
<code>mask</code>	Name of a signal to which the TFALL command applies. If you do not specify a <i>mask</i> value, the TFALL command applies to all input signals.

---

### Description

Use this command to specify the fall time of each input signal for which the mask applies. The TUNIT command defines the time unit of TFALL.

- If you do not use any TFALL command to specify the fall time of the signals, HSPICE or HSPICE RF uses the value defined in the *slope* command.
- If you apply more than one TFALL command to a signal, the last command overrides the previous commands and HSPICE or HSPICE RF issues a warning message.

TFALL commands have no effect on the expected output signals.

### Examples

In Example1, the TFALL command assigns a fall time of 0.5 time units to all vectors.

#### Example 1

```
TFALL 0.5
```

In the following example, the TFALL command assigns a fall time of 0.3 time units overriding the older setting of 0.5 to vectors 2, 3, and 4 to 7.

#### Example 2

```
TFALL 0.3 0 1 1 137F 00000000
```

## Chapter 4: Digital Vector File Commands

### TFALL

In the following example, the `TFALL` command assigns a fall time of 0.9 time units to vectors 8 through 11.

```
TFALL 0.9 0 0 0 0000 11110000
```

#### See Also

[TRISE](#)  
[TUNIT](#)

---

## TRISE

Specifies the rise time of each input signal for which the mask applies.

### Syntax

```
TRISE input_rise_time [mask]
```

Argument	Description
<code>input_rise_time</code>	Rise time of the input signal.
<code>mask</code>	Name of a signal to which the TRISE command applies. If you do not specify a <i>mask</i> value, the TRISE command applies to all input signals.

### Description

Use this command to specify the rise time of each input signal for which the mask applies. The `TUNIT` command defines the time unit of `TRISE`.

- If you do not use any `TRISE` command to specify the rising time of the signals, HSPICE or HSPICE RF uses the value defined in the *slope* command.
- If you apply more than one `TRISE` command to a signal, the last command overrides the previous commands and HSPICE or HSPICE RF issues a warning message.

`TRISE` commands have no effect on the expected output signals.

### Examples

In this example, the `TRISE` command assigns a rise time of 0.3 time units to all vectors.

#### Example 1

```
TRISE 0.3
```

In this example, the `TRISE` command assigns a rise time of 0.5 time units overriding the older setting of 0.3 in at least some of the bits in vectors 2, 3, and 4 through 7.

#### Example 2

```
TRISE 0.5 0 1 1 137F 00000000
```

## Chapter 4: Digital Vector File Commands

### TRISE

In Example 3, the `TRISE` command assigns a rise time of 0.8 time units to vectors 8 through 11.

#### *Example 3*

```
TRISE 0.8 0 0 0 0000 11110000
```

#### **See Also**

[TFALL](#)  
[TUNIT](#)



---

## TRIZ

Specifies the output impedance when the signal for which the mask applies is in tristate.

### Syntax

```
TRIZ output_impedance [mask]
```

Argument	Description
<code>output_impedance</code>	Output impedance of the input signal.
<code>mask</code>	Name of a signal to which the TRIZ command applies. If you do not specify a <i>mask</i> value, the TRIZ command applies to all input signals.

### Description

Use this command to specify the output impedance when the signal (for which the mask applies) is in *tristate*; TRIZ applies only to the input signals.

- If you do not specify the tristate impedance of a signal, in a TRIZ command, HSPICE or HSPICE RF assumes 1000M.
- If you apply more than one TRIZ command to a signal, the last command overrides the previous commands and HSPICE or HSPICE RF issues a warning.

TRIZ commands have no effect on the expected output signals.

### Examples

```
TRIZ 15.1Meg
TRIZ 150Meg 1 1 1 0000 00000000
TRIZ 50.5Meg 0 0 0 137F 00000000
```

- The first TRIZ command sets the high impedance resistance globally at 15.1 Mohms.
- The second TRIZ command increases the value to 150 Mohms for vectors 1 to 3.
- The last TRIZ command increases the value to 50.5 Mohms for vectors 4 through 7.

---

## TSKIP

Causes HSPICE to ignore the absolute time field in the tabular data.

### Syntax

```
TSKIP absolute_time tabular_data ...
```

---

Argument	Description
<i>absolute_time</i>	Absolute time.
<i>tabular_data</i>	Data captured at <i>absolute_time</i> .

---

### Description

Use this command to cause HSPICE to ignore the absolute time field in the tabular data. You can then keep, but ignore, the absolute time field for each row in the tabular data when you use the `.PERIOD` command.

You might do this, for example, if for testing reasons the absolute times are not perfectly periodic. Another reason might be that a path in the circuit does not meet timing, but you might still use it as part of a test bench. Initially, HSPICE writes to the vector file using absolute time. After you fix the circuit, you might want to use periodic data.

### Examples

```
radix 1111 1111
period 10
tskip
11.0 1000 1000
20.0 1100 1100
33.0 1010 1001
```

HSPICE or HSPICE RF ignores the absolute times 11.0, 20.0 and 33.0, but HSPICE does process the tabular data on the same lines as those absolute times.

### See Also

[PERIOD](#)

---

## TUNIT

Defines the time unit for PERIOD, TDELAY, IDELAY, ODELAY, SLOPE, TRISE, TFALL, and absolute time.

### Syntax

```
TUNIT [fs|ps|ns|us|ms]
```

---

Argument	Description
fs	femtosecond
ps	picosecond
ns	nanosecond (default)
us	microsecond
ms	millisecond

---

### Description

Use this command to define the time unit in the digital vector file for PERIOD, TDELAY, IDELAY, ODELAY, SLOPE, TRISE, TFALL, and absolute time.

- If you do not specify the TUNIT command, the default time unit value is ns.
- If you define more than one TUNIT command, the last command overrides the previous command.

### Examples

The TUNIT command in this example specifies that the absolute times in the Tabular Data section are 11.0ns, 20.0ns, and 33.0ns.

```
TUNIT ns
11.0 1000 1000
20.0 1100 1100
33.0 1010 1001
```

The following are legal ways to write the time values.

```
tunit 999ns
tunit .99ps
tunit .99e+6ps
tunit 999 ns
tunit .99 ps
```

## Chapter 4: Digital Vector File Commands

### TUNIT

The following are examples of wrong syntax which will result in an error message:

```
tunit .99eps  
tunit .99 e+6ps  
tunit .99 eps
```

#### See Also

- IDELAY
- ODELAY
- PERIOD
- SLOPE
- TDELAY
- TFALL
- TRISE

---

## VIH

Specifies the logic-high voltage for each input signal to which the mask applies.

### Syntax

```
VIH logic-high_voltage [mask]
```

---

Argument	Description
<code>logic-high_voltage</code>	Logic-high voltage for an input signal. The default is 3.3.
<code>mask</code>	Name of a signal to which the VIH command applies. If you do not specify a <i>mask</i> value, the VIH command applies to all input signals.

---

### Description

Use this command to specify the logic-high voltage for each input signal to which the mask applies.

- If you do not specify the logic high voltage of the signals in a VIH command, HSPICE assumes 3.3.
- If you use more than one VIH command for a signal, the last command overrides previous commands and HSPICE issues a warning.

VIH commands have no effect on the expected output signals.

### Examples

```
VIH 5.0  
VIH 3.5 0 0 0 0000 11111111
```

- The first VIH command sets all input vectors to 5V when they are high.
- The last VIH command changes the logic-high voltage from 5V to 3.5V for the last eight vectors.

### See Also

[VIL](#)  
[VOH](#)  
[VOL](#)  
[VTH](#)

---

## VIL

Specifies the logic-low voltage for each input signal to which the mask applies.

### Syntax

```
VIL logic-low_voltage [mask]
```

---

Argument	Description
<code>logic-low_voltage</code>	Logic-low voltage for an input signal. The default is 0.0.
<code>mask</code>	Name of a signal to which the VIL command applies. If you do not specify a <i>mask</i> value, the VIL command applies to all input signals.

---

### Description

Use this command to specify the logic-low voltage for each input signal to which the mask applies.

- If you do not specify the logic-low voltage of the signals in a `VIL` command, HSPICE or HSPICE RF assumes 0.0.
- If you use more than one `VIL` command for a signal, the last command overrides previous commands and HSPICE issues a warning.

`VIL` commands have no effect on the expected output signals.

### Examples

```
VIL 0.0
VIL 0.5 0 0 0 0000 11111111
```

- The first `VIL` command sets the logic-low voltage to 0V for all vectors.
- The second `VIL` command changes the logic-low voltage to 0.5V for the last eight vectors.

### See Also

[VIH](#)  
[VOH](#)  
[VOL](#)  
[VTH](#)

## VNAME

Defines the name of each vector.

### Syntax

```
VNAME vector_name [[starting_index:ending_index]]
```

Argument	Description
<code>vector_name</code>	Name of the vector, or range of vectors.
<code>starting_index</code>	First bit in a range of vector names.
<code>ending_index</code>	Last bit in a range of vector names. You can associate a single name with multiple bits (such as bus notation).  The opening and closing brackets and the colon are required; they indicate that this is a range. The vector name must correlate with the number of bits available.  You can nest the bus definition inside other grouping symbols, such as {}, (), [], and so on. The bus indices expand in the specified order

### Description

Use this command to define the name of each vector. If you do not specify VNAME, HSPICE or HSPICE RF assigns a default name to each signal: V1, V2, V3, and so on. If you define more than one VNAME command, the last command overrides the previous command.

### Examples

Auto-defined names for each signal.

#### Example 1

```
RADIX 1 1 1 1 1 1 1 1 1 1 1 1
VNAME V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12
```

Example 2 represents a0, a1, a2, and a3, in that order. HSPICE or HSPICE RF does not reverse the order to make a3 the first bit. The bit order is MSB:LSB, which means most significant bit to least significant bit. For example, you can represent a 5-bit bus such as: {a4 a3 a2 a1 a0}, using this notation: a[[4:0]].

## Chapter 4: Digital Vector File Commands

### VNAME

The high bit is  $a_4$ , which represents  $2^4$ . It is the largest value and therefore is the MSB.

#### *Example 2*

```
VNAME a[[0:3]]
```

HSPICE or HSPICE RF generates voltage sources with the following names:

```
VA0 VA1 VB4 VB3 VB2 VB1
```

- *VA0* and *VB4* are the MSBs.
- *VA1* and *VB1* are the LSBs.

#### *Example 3*

```
RADIX 2 4  
VNAME VA[[0:1]] VB[[4:1]]
```

For Example 4, HSPICE or HSPICE RF generates voltage sources with the following names: *VA[0]* *VA[1]* *VB<4>* *VB<3>* *VB<2>* *VB<1>*

#### *Example 4*

```
VNAME VA[[0:1]] VB<[4:1]>
```

Example 5 specifies a single bit of a bus. This range creates a voltage source named *VA [2]*.

#### *Example 5*

```
VNAME VA[[2:2]]
```

Example 6 generates signals named *A0*, *A1*, *A2*, ... *A23*.

#### *Example 6*

```
RADIX 444444  
VNAME A[[0:23]]
```



## VOH

Specifies the logic-high threshold voltage for each output signal to which the mask applies.

### Syntax

```
VOH logic-high_threshold_voltage [mask]
```

Argument	Description
logic-high_threshold_voltage	Logic-high threshold voltage for an output vector. The default is 2.66.
mask	Name of a signal to which the VOH command applies. If you do not specify a <i>mask</i> value, the VOH command applies to all output signals.

### Description

Use this command to specify the logic-high threshold voltage for each output signal to which the mask applies.

- If you do not specify the logic-high threshold voltage in a VOH command, HSPICE assumes 2.64.
- If you apply more than one VOH command to a signal, the last command overrides the previous commands and HSPICE issues a warning.

VOH commands have no effect on input signals.

### Examples

```
VOH 4.75
VOH 4.5 1 1 1 137F 00000000
VOH 3.5 0 0 0 0000 11111111
```

- The first line tries to set a logic-high threshold output voltage of 4.75V, but it is redundant.
- The second line changes the voltage level to 4.5V for the first seven vectors.
- The last line changes the last eight vectors to a 3.5V logic-high threshold output.

These second and third lines completely override the first VOH command.

If you do not define either VOH or VOL, HSPICE or HSPICE RF uses VTH (default or defined).

**See Also**

[VIH](#)  
[VIL](#)  
[VOL](#)  
[VTH](#)

## VOL

Specifies the logic-low threshold voltage for each output signal to which the mask applies.

### Syntax

```
VOL logic-low_threshold_voltage [mask]
```

Argument	Description
logic-low_voltage	Logic-low threshold voltage for an output vector. The default is 0.64.
mask	Name of a signal to which the VOL command applies. If you do not specify a <i>mask</i> value, the VOL command applies to all output signals.

### Description

Use this command to specify the logic-low threshold voltage for each output signal to which the mask applies.

- If you do not specify the logic-low threshold voltage in a VOL command, HSPICE assumes 0.66.
- If you apply more than one VOL command to a signal, the last command overrides the previous commands and HSPICE issues a warning.

### Examples

```
VOL 0.0
VOL 0.2 0 0 0 137F 00000000
VOL 0.5 1 1 1 0000 00000000
```

- The first VOL command sets the logic-low threshold output to 0V.
- The second VOL command sets the output voltage to 0.2V for the fourth through seventh vectors.
- The last command increases the voltage further to 0.5V for the first three vectors.

These second and third lines completely override the first VOL command.

If you do not define either VOH or VOL, HSPICE or HSPICE RF uses VTH (default or defined).

**See Also**

[VIH](#)  
[VIL](#)  
[VOH](#)  
[VTH](#)

---

## VREF

Specifies the name of the reference voltage for each input vector to which the mask applies.

### Syntax

VREF *reference\_voltage*

Argument	Description
reference_voltage	Reference voltage for each input vector. The default is 0.

### Description

Use this command to specify the name of the reference voltage for each input vector to which the mask applies. Similar to the TDELAY command, the VREF command applies only to input signals.

- If you do not specify the reference voltage name of the signals in a VREF command, HSPICE assumes 0.
- If you apply more than one VREF command, the last command overrides the previous commands and HSPICE issues a warning.

VREF commands have no effect on the output signals.

### Examples

```
VNAME v1 v2 v3 v4 v5[[1:0]] v6[[2:0]] v7[[0:3]] v8 v9 v10
VREF 0
VREF 0 111 137F 000
VREF vss 0 0 0 0000 111
```

When HSPICE or HSPICE RF implements these commands into the netlist, the voltage source realizes *v1*:

```
v1 V1 0 pw1(.....)
```

as well as *v2*, *v3*, *v4*, *v5*, *v6*, and *v7*.

However, *v8* is realized by

```
v8 V8 vss pw1(.....)
```

*v9* and *v10* use a syntax similar to *v8*.

### See Also

[TDELAY](#)

---

## VTH

Specifies the logic threshold voltage for each output signal to which the mask applies.

### Syntax

*VTH logic-threshold\_voltage*

---

Argumentt	Description
logic-threshold_voltage	Logic-threshold voltage for an output vector. The default is 1.65.

---

### Description

Use this command to specify the logic threshold voltage for each output signal to which the mask applies. It is similar to the `TDELAY` command. The threshold voltage determines the logic state of output signals for comparison with the expected output signals.

- If you do not specify the threshold voltage of the signals in a `VTH` command, HSPICE assumes 1.65.
- If you apply more than one `VTH` command to a signal, the last command overrides the previous commands and HSPICE or HSPICE RF issues a warning.

`VTH` commands have no effect on the input signals.

### Examples

```
VTH 1.75
VTH 2.5 1 1 1 137F 00000000
VTH 1.75 0 0 0 0000 11111111
```

- The first `VTH` command sets the logic threshold voltage at 1.75V.
- The next line changes that threshold to 2.5V for the first 7 vectors.
- The last line changes that threshold to 1.75V for the last 8 vectors.

All of these examples apply the same vector pattern and both output and input control commands, so the vectors are all bidirectional.

### See Also

[TDELAY](#)  
[VIH](#)  
[VIL](#)

VOH  
VOL

**Chapter 4: Digital Vector File Commands**  
VTH



## Obsolete Commands and Options

---

*Describes the obsolete or rarely used HSPICE commands.*

The following commands and options are included for completeness only. More efficient functionality and commands are available.

- `.ACDCFACTOR`
- `.GRAPH`
- `.MODEL` Command for `.GRAPH`
- `.NET`
- `.PLOT`
- `.WIDTH`
- `.OPTION ALT999` or `ALT9999`
- `.OPTION BKPSIZ`
- `.OPTION CDS`
- `.OPTION CO`
- `.DEGINFO`
- `.OPTION H9007`
- `.OPTION MEASSORT`
- `.OPTION MENTOR`
- `.OPTION MODSRH`
- `.OPTION NOPAGE`
- `.OPTION PIVREF`
- `.OPTION PIVREL`

## Appendix A: Obsolete Commands and Options

### .ACDCFACTOR

- .OPTION PLIM
- .OPTION SDA
- .OPTION SIM\_LA\_MINMODE
- .OPTION SPICE
- .OPTION ZUKEN

---

## .ACDCFACTOR

OBSOLETE with the D-2010.03 release. Invokes degradation characterization under DC/AC stress, respectively.

### Syntax

```
.ACDCFACTOR vds=value vgs=value [vbs=value] Time=value
```

---

Argument	Description
vds	vds value for the dc calculation
vgs	vgs value for the dc calculation
vbs	vbs value for the dc calculation
Time	Time to calculate the acdcfactor

---

### Description

This command invokes degradation characterization under DC/AC stress, respectively, based on the customized algorithm(s) implemented in the MOSRA API model. A new output file *\*.acdcfactor* is generated.

### Example

```
.ACDCFACTOR Vds=1.0 Vgs=1.2 Vbs=0 time = 4n
```

## .GRAPH

OBSOLETE. Provides high-resolution plots of HSPICE simulation results. This is an obsolete command. You can gain the same functionality by using the .PROBE command.

```
.GRAPH antype [MODEL=mname] unam1= ov1,
+ [unam2=ov2] ... [unamn=ovn] (plo,phi)
```

Argument	Description
<i>antype</i>	Type of analysis for the specified plots (outputs). Analysis types are: DC, AC, TRAN, NOISE, or DISTO.
<i>mname</i>	Plot model name, referenced in the .GRAPH command. Use .GRAPH and its plot name to create high-resolution plots directly from HSPICE.
<i>unam1</i> ...	You can define output names, which correspond to the <i>ov1 ov2 ...</i> output variables ( <i>unam1 unam2 ...</i> ), and use them as labels, instead of output variables for a high resolution graphic output.
<i>ov1 ...</i>	Output variables to print. Can be voltage, current, or element template variables from a different type of analysis. You can also use algebraic expressions as output variables, but you must define them inside the PAR( ) command.
<i>plo</i> , <i>phi</i>	Lower and upper plot limits. Set the plot limits only at the end of the .GRAPH command.

### Description

Use this command when you need high-resolution plots of HSPICE simulation results.

Each .GRAPH command creates a new *.gr#* file, where # ranges first from 0 to 9 and then from a to z. You can create up to 10000 graph files.

You can include wildcards in .GRAPH commands.

You cannot use .GRAPH commands in the Windows version of HSPICE or in HSPICE RF.

## Appendix A: Obsolete Commands and Options

.GRAPH

### Example

```
.GRAPH DC cgb=1x18(m1) cgd=1x19(m1)
+ cgs=1x20(m1)
.GRAPH DC MODEL=plotbjt
+ model_ib=i2(q1) meas_ib=par(ib)
+ model_ic=i1(q1) meas_ic=par(ic)
+ model_beta=par('i1(q1)/i2(q1)')
+ meas_beta=par('par(ic)/par(ib)')(1e-10,1e-1)
.MODEL plotbjt PLOT MONO=1 YSCAL=2 XSCAL=2
+ XMIN=1e-8 XMAX=1e-1
```

## .MODEL Command for .GRAPH

OBSOLETE. For a description of how to use the .MODEL command with .GRAPH, see [.MODEL](#).

### Syntax

N/A

Argument	Description
MONO	Monotonic option. MONO=1 automatically resets the x-axis, if any change occurs in the x direction. Default 0.0
TIC	Shows tick marks. Default 0.0
FREQ	Plots symbol frequency. Default 0.0 <ul style="list-style-type: none"> <li>▪ A value of 0 does not generate plot symbols.</li> <li>▪ A value of <math>n</math> generates a plot symbol every <math>n</math> points. This is not the same as the FREQ keyword in element commands.</li> </ul>
XGRID, YGRID	Set these values to 1.0, to turn on the axis grid lines. Default 0.0
XMIN, XMAX	<ul style="list-style-type: none"> <li>▪ If XMIN is not equal to XMAX, then XMIN and XMAX determine the x-axis plot limits.</li> <li>▪ If XMIN equals XMAX, or if you do not set XMIN and XMAX, then HSPICE automatically sets the plot limits. These limits apply to the actual x-axis variable value, regardless of the XSCAL type.</li> </ul> Default 0.0
XSCAL	Scale for the x-axis. Two common axis scales are: Linear(LIN) (XSCAL=1) Logarithm(LOG) (XSCAL=2) Default 1.0
YMIN, YMAX	<ul style="list-style-type: none"> <li>▪ If YMIN is not equal to YMAX, then YMIN and YMAX determine the y-axis plot limits. The y-axis limits in the .GRAPH command overrides YMIN and YMAX in the model.</li> <li>▪ If you do not specify plot limits, HSPICE sets the plot limits. These limits apply to the actual y-axis variable value, regardless of the YSCAL type.</li> </ul> Default 0.0

## Appendix A: Obsolete Commands and Options

.MODEL Command for .GRAPH

Argument	Description
YSCAL	Scale for the y-axis. Two common axis scales are: Linear(LIN) (XSCAL=1) Logarithm(LOG) (XSCAL=2) Default 1.0

---

## .NET

OBSOLETE. Computes parameters for impedance, admittance, hybrid, and scattering matrixes. This functionality is replaced by the .LIN command.

### Syntax

#### One-Port Network

```
.NET input [RIN=val]
```

```
.NET input val
```

#### Two-Port Network

```
.NET output input [ROUT=val] [RIN=val]
```

---

Argument	Description
input	Name of the voltage or current source for AC input.
output	Output port. It can be: <ul style="list-style-type: none"> <li>▪ An output voltage, <math>V(n1[,n2])</math>.</li> <li>▪ An output current, I (source), or I (element).</li> </ul>
RIN	Input or source resistance. RIN calculates output impedance, output admittance, and scattering parameters. The default RIN value is 1 ohm.
ROUT	Output or load resistance. ROUT calculates input impedance, admittance, and scattering parameters. The default is 1 ohm.

---

### Description

You can use the .NET command to compute parameters for:

- Z impedance matrix
- Y admittance matrix
- H hybrid matrix
- S scattering matrix

You can use the .NET command only in conjunction with the .AC command.

HSPICE also computes:

## Appendix A: Obsolete Commands and Options

### .NET

- Input impedance
- Output impedance
- Admittance

This analysis is part of AC small-signal analysis. To run network analysis, specify the frequency sweep for the `.AC` command.

### Examples

#### One-Port Network

```
.NET VINAC RIN=50
.NET IIN RIN=50
```

#### Two-Port Network

```
.NET V(10,30) VINAC ROUT=75 RIN=50
.NET I(RX) VINAC ROUT=75 RIN=50
```



---

## .PLOT

OBSOLETE. Plots the output values of one or more variables in a selected HSPICE analysis as a low-resolution (ASCII) plot in the output listing file. This is an obsolete command. You get the same functionality using the .PRINT command.

### Syntax

```
.PLOT antype ov1 [(plo1,phi1)] [ov2] [(plo2,phi2)] ...]
```

---

Argument	Description
<i>antype</i>	Type of analysis for the specified plots. Analysis types are: DC, AC, TRAN, NOISE, or DISTO.
<i>ov1</i> ...	Output variables to plot: voltage, current, or element template variables (HSPICE only; HSPICE RF does not support element template output or .PLOT commands) from a DC, AC, TRAN, NOISE, or DISTO analysis. See the next sections for syntax.
<i>plo1, phi1</i> ...	Lower and upper plot limits. The plot for each output variable uses the first set of plot limits after the output variable name. Set a new plot limit for each output variable after the first plot limit. For example to plot all output variables that use the same scale, specify one set of plot limits at the end of the .PLOT command. If you set the plot limits to (0,0) HSPICE automatically sets the plot limits.

---

### Description

Use this command to plot the output values of one or more variables in a selected HSPICE analysis. Each .PLOT command defines the contents of one plot, which can contain more than one output variable.

If more than one output variable appears on the same plot, HSPICE prints *and* plots the first variable specified. To print out more than one variable, include another .PLOT command. You can include wildcards in .PLOT commands.

**Examples**

In Example 1,

- In the first line, `PAR` plots the ratio of the collector current and the base current for the Q1 transistor.
- In the second line, the `VDB` output variable plots the AC analysis results (in decibels) for node 5.
- In the third line, the AC plot can include `NOISE` results and other variables that you specify.

*Example 1*

```
.PLOT DC V(4) V(5) V(1) PAR(`I1(Q1)/I2(Q1)')  
.PLOT TRAN V(17,5) (2,5) I(VIN) V(17) (1,9)  
.PLOT AC VM(5) VM(31,24) VDB(5) VP(5) INOISE
```

In the last line of Example 2, `HSPICE` sets the plot limits for `V(1)` and `V(2)`, but you specify 0 and 5 volts as the plot limits for `V(3)` and `V(4)`.

*Example 2*

```
.PLOT AC ZIN YOUT(P) S11(DB) S12(M) Z11(R)  
.PLOT DISTO HD2 HD3(R) SIM2  
.PLOT TRAN V(5,3) V(4) (0,5) V(7) (0,10)  
.PLOT DC V(1) V(2) (0,0) V(3) V(4) (0,5)
```

---

## .WIDTH

OBSOLETE. Specifies the width of the low resolution (ASCII) plot in the listing file.

### Syntax

```
.WIDTH OUT={ 80 | 132 }
```

---

Argument	Description
OUT	Output print width.

---

### Description

Use this command to specify the width of the low resolution (ASCII) plot. Permissible values for OUT are 80 and 132. You can also use .OPTION CO to set the OUT value.

### Examples

```
.WIDTH OUT=132 $ SPICE compatible style
.OPTION CO=132 $ preferred style
```

---

## .OPTION ACCT

OBSOLETE. Generates a detailed accounting report.

### Syntax

```
.OPTION ACCT
.OPTION ACCT= [1 | 2]
```

**Default**    **Default** 1

Argument	Description
.OPTION ACCT	Enables reporting.
.OPTION ACCT=1 (default)	Is the same as ACCT without arguments.
.OPTION ACCT=2	Enables reporting and matrix statistic reporting.

### Description

Obsolete with 2009.03 release (MT starts will be printed by default.) Use this option to generate a detailed accounting report.

### Examples

```
.OPTION ACCT=2
```

The ratio of TOT. ITER to CONV. ITER is the best measure of simulator efficiency. The theoretical ratio is 2:1. In this example the ratio is 2.57:1. SPICE generally has a ratio from 3:1 to 7:1.

In transient analysis, the ratio of CONV. ITER to # POINTS is the measure of the number of points evaluated to the number of points printed. If this ratio is greater than about 4:1, the convergence and time step control tolerances might be too tight for the simulation.

## **.OPTION ALT999 or ALT9999**

Allows the `.GRAPH` command to create more output files when you run `.ALTER` simulations.

### **Syntax**

```
.OPTION ALT999  
.OPTION ALT9999
```

### **Description**

Use this option to allow the `.GRAPH` command to create more output files when you run `.ALTER` simulations.

This option is now obsolete. HSPICE can now generate up to 10,000 unique files without using this option.

## **.OPTION BKPSIZ**

OBSOLETE. Sets the size of the breakpoint table.

### **Syntax**

```
.OPTION BKPSIZ=x
```

**Default** **Default** 5000

### **Description**

Use this option to set the size of the breakpoint table. This is an obsolete option, provided only for backward-compatibility.

## **.OPTION CDS**

OBSOLETE. Produces a Cadence WSF (ASCII format) post-analysis file for Opus<sup>▲</sup>.

### **Syntax**

```
.OPTION CDS=x
```

### **Description**

Use this option to produce a Cadence WSF (ASCII format) post-analysis file for Opus<sup>▲</sup> when CDS=2. This option requires a specific license. The CDS option is the same as the SDA option.

---

## .OPTION CO

OBSOLETE. Sets column width for printouts.

### Syntax

```
.OPTION CO=column_width
```

---

Argument	Description
<i>column_width</i>	The number of characters in a single line of output.

---

### Description

(Obsolete) Use this option to set the column width for printouts. The number of output variables that print on a single line of output is a function of the number of columns.

You can set up to 5 output variables per 80-column output, and up to 8 output variables per 132-column output with 12 characters per column. HSPICE automatically creates additional print commands and tables for all output variables beyond the number that the CO option specifies. The default is 78.

### Examples

```
* Narrow print-out (default)
.OPTION CO=80
* Wide print-out
.OPTION CO=132
```



## .DEGINFO

OBSOLETE with the D-2010.03 release. Generates the degradation information during the transient simulation for user-specified MOSFET element when used with the MOSRA API.

### Syntax

```
.DEGINFO [YES|NO] MOSFET_instance_name1
+ MOSFET_instance_name2...
```

**Default** YES

Argument	Description
YES NO	YES: the MOSFETs in the instance name list will be recorded. NO: the MOSFETs in the instance name list will NOT be recorded.

### Description

Use this command to generate the degradation information during the transient simulation for user-specified MOSFET element. A separate output file, *\*.deginfo*, is generated.

### Examples

The command in Example 1 records the information of MN1 and MP2 in the *.deginfo* file.

#### Example 1

```
.DEGINFO YES MN1 MP2
```

The command in Example 2 records the information of all of the MOSFETs except MN3 and MP5.

#### Example 2

```
DEGINFO NO MN3 MP5
```

## **.OPTION H9007**

OBSOLETE. Sets default values for general-control options to correspond to values for HSPICE H9007D.

### **Syntax**

```
.OPTION H9007
```

**Default** 0

### **Description**

Use this option to set default values for general-control options to correspond to values for HSPICE H9007D. If you set this option, HSPICE does not use the `EXPLI` model parameter.

## .OPTION MEASSORT

OBSOLETE. Automatically sorts large numbers of .MEASURE commands.  
(This option is obsolete.)

### Syntax

```
.OPTION MEASSORT=x
```

**Default** 0

### Description

Starting in version 2003.09, this option is obsolete. Measure performance is now order-independent and HSPICE ignores this option.

In versions of HSPICE before 2003.09, to automatically sort large numbers of .MEASURE commands, you could use the .OPTION MEASSORT command.

- .OPTION MEASSORT=0 (default; did not sort .MEASURE commands).
- .OPTION MEASSORT=1 (internally sorted .MEASURE commands).

You needed to set this option to 1 only if you used a large number of .MEASURE commands, where you needed to list similar variables together (to reduce simulation time). For a small number of .MEASURE commands, turning on internal sorting sometimes slowed-down simulation while sorting, compared to not sorting first.

## .OPTION MENTOR

OBSOLETE. Enables the Mentor MSPICE-compatible (ASCII) interface.

### Syntax

```
.OPTION MENTOR=0 | 1 | 2
```

**Default**    **Default** 0

### Description

Use this option to enable the Mentor MSPICE-compatible (ASCII) interface. `MENTOR=2` enables that interface. This option requires a specific license.

---

## .OPTION MODSRH

OBSOLETE. Made obsolete beginning in the 2008.03 release as it increases runtime and costs more memory. Controls whether HSPICE loads or references a model described in a `.MODEL` command, but not used in the netlist.

### Syntax

```
.OPTION MODSRH=0 | 1
```

### Description

Use this option to control whether HSPICE loads or references a model described in a `.MODEL` command, but is not used in the netlist.

This option parameter determines if HSPICE reads and loads every model card or all model bins that are present in netlists and model libraries during a simulation run. When this parameter is set to 0, all the model cards in the model libraries are read into HSPICE even if there are certain models or bins that are not referenced by any elements of the netlists. If this option parameter is not assigned a numerical value or is set to 1, or it is not specified at all, then only those model cards or model bins that are referenced are read into the HSPICE executable for simulation.

**Note:** The `.OPTION MODSRH` control must appear before the `.MODEL` definition.

- `MODSRH=0`: all models expanded even if the model described in a `.MODEL` command is not referenced. This was the default prior to Y-2006.03 and restored in A-2008.03.
- `MODSRH=1`: only referenced models are expanded. This option shortens simulation runtime when the netlist references many models, but no element in the netlist calls those models. This option increased read-in time. This became the default after 2008.03.

### Examples

In this example, the input file automatically searches `t6.inc` for the `nch` model, but it is not loaded.

## Appendix A: Obsolete Commands and Options

### .OPTION MODSRH

```
example.sp:  
.option post modsrh=1  
x11 net8 b c t6  
xi0 a b net8 t6  
v1 a 0 pulse 3.3 0.0 10E-6 1E-9 1E-9  
+ 25E-6 50E-6  
v2 b 0 2  
v3 c 0 3  
.model nch nmos level=49 version=3.2  
.end
```

### See Also

[.MODEL](#)

## **.OPTION PIVREF**

OBSOLETE. Made obsolete with 2009.03 release. Sets a pivot reference.

### **Syntax**

```
.OPTION PIVREF=x
```

### **Description**

Use this option to set a pivot reference. Use PIVREF in PIVOT=11, 12, or 13 to limit the size of the matrix. The default is 1e+8.

## **.OPTION NOPAGE**

OBSOLETE Suppresses page ejects for title headings.

### **Syntax**

```
.OPTION NOPAGE= [0 | 1]
```

### **Description**

Use this option to suppress page ejects for title headings.



## .OPTION PIVREL

OBSOLETE (2009.03) release. Sets maximum and minimum ratio of a row or matrix.

### Syntax

```
.OPTION PIVREL=x
```

### Description

Use this option to set the maximum and minimum ratio of a row or matrix. Use only if `PIVOT=1`. Large values for `PIVREL` can result in very long matrix pivot times; however, if the value is too small, no pivoting occurs. Start with small values of `PIVREL` by using an adequate but not excessive value for convergence and accuracy. The default is  $1e-4$ .

---

## .OPTION PLIM

OBSOLETE. Made obsolete with 2009.03 release. Specifies plot size limits for current and voltage plots.

### Syntax

```
.OPTION PLIM
```

**Default**   **Default** 0

### Description

Use this option to specify plot size limits for current and voltage plots:

- Finds a common plot limit and plots all variables on one graph at the same scale.
- Enables SPICE-type plots, which create a separate scale and axis for each plot variable.

This option does not affect postprocessing of graph data.

## .OPTION SDA

OBSOLETE. Produces a Cadence WSF (ASCII format) post-analysis file for Opus<sup>▲</sup>.

### Syntax

```
.OPTION SDA=x
```

**Default**    **Default** 0

### Description

Use this option to produce a Cadence WSF (ASCII format) post-analysis file for Opus<sup>▲</sup>. Set `SDA=2` to produce this file. This option requires a specific license. The `SDA` is the same as the `CDS` option.

## **.OPTION SPICE**

Makes HSPICE compatible with Berkeley SPICE.

### **Syntax**

```
.OPTION SPICE
```

### **Description**

When the option SPICE is set, the following options and model parameters are used:

General parameters used with .OPTION SPICE:

```
TNOM=27 DEFNRD=1 DEFNRS=1 INGOLD=2 ACOUT=0 DC  
PIVOT PIVTOL=1E-13 PIVREL=1E-3 RELTOL=1E-3 ITL1=100  
ABSMOS=1E-6 RELMOS=1E-3 ABSTOL=1E-12 VNTOL=1E-6  
ABSVDC=1E-6 RELVDC=1E-3 RELI=1E-3
```

Transient parameters used with .OPTION SPICE:

```
DCAP=1 RELQ=1E-3 CHGTOL=1E-14 ITL3=4 ITL4=10 ITL5=5000  
FS=0.125 FT=0.125
```

Model parameters used with .OPTION SPICE:

For BJT: MJS=0

For MOSFET, CAPOP=0

LD=0 if not user-specified

UTRA=0 not used by SPICE for level=2

NSUB must be specified

NLEV=0 for SPICE noise equation

## .OPTION SIM\_LA\_MINMODE

Obsolete as of 2009.03. Reduces the number of nodes instead of the number of elements.

### Syntax

```
.OPTION SIM_LA_MINMODE=ON | OFF
```

**Default**    **Default**    OFF

### Description

Use this option to reduce the number of nodes instead of the number of elements.

- ON: reduces the number of nodes
- OFF: does not reduce the number of nodes.

## **.OPTION ZUKEN**

OBSOLETE. Enables or disables the Zuken interface.

### **Syntax**

```
.OPTION ZUKEN=x
```

### **Description**

Use this option to enable or disable the Zuken interface.

- If x is 2, the interface is enabled.
- If x is 1 (default), the interface is disabled.

## How Options Affect other Options

---

*Describes the effects of specifying control options on other options in the netlist.*

The following options either impact or are impacted by the specifying of other .OPTION parameters:

- GEAR Method
- ACCURATE
- FAST
- GEAR Method, ACCURATE
- ACCURATE, GEAR Method
- ACCURATE, FAST
- GEAR Method, FAST
- GEAR Method, ACCURATE, FAST
- RUNLVL=N
- RUNLVL, ACCURATE, FAST, GEAR method
- DVDT=1,2,3
- LVLTIM=0,2,3
- KCLTEST
- BRIEF
- Option Notes
- Finding the Golden Reference for Options

## GEAR Method

Specifying .OPTION METHOD=GEAR sets the values of other options as follows:

- BYPASS = 0
- BYTOL = 50u
- DVDT = 3
- LVLTIM = 2
- MBYPASS = 1.0
- METHOD = 2
- RMAX = 2.0
- SLOPETOL = 500m

---

## ACCURATE

Specifying the ACCURATE option sets the values of other options as follows:

- ABSVAR = 0.2
- ACCURATE = 1
- BYPASS = 2
- DVDT = 2
- FFT\_ACCU = 1
- FT = 0.2
- LVLTIM = 3
- RELMOS = 0.01
- RELVAR = 0.2

---

## FAST

Specifying the FAST option sets the values of other options as follows:



- BYTOL = 50u
- DVDT = 3
- BYPASS = 0
- DVDT = 2
- FAST = 1
- MBYPASS = 1.0
- RMAX = 2.0
- SLOPETOL = 500m

---

## GEAR Method, ACCURATE

Specifying .OPTION METHOD=GEAR first with the ACCURATE option sets the values of other options as follows:

- ABSVAR = 0.2
- ACCURATE = 1
- BYPASS = 2
- BYTOL = 50u
- DVDT = 2
- FFT\_ACCU = 1
- FT = 0.2
- LVLTIM = 3
- MBYPASS = 1.0
- METHOD = 2
- RELMOS = 0.01
- RELVAR = 0.2
- RMAX = 2
- SLOPETOL = 500m

**Note:** When GEAR is specified first, DVDT=2 and LVLTIM=3.

## ACCURATE, GEAR Method

Specifying the ACCURATE option first in with .OPTION METHOD=GEAR sets the values of other options as follows:

- ABSVAR = 0.2
- ACCURATE = 1
- BYPASS = 2
- BYTOL = 50u
- DVDT = 3
- FFT\_ACCU = 1
- FT = 0.2
- LVLTIM = 2
- MBYPASS = 1.0
- METHOD = 2
- RELMOS = 0.01
- RELVAR = 0.2
- RMAX = 2
- SLOPETOL = 500m

**Note:** When ACCURATE is specified before the GEAR method, then DVDT=2, LVLTIM=3.

---

## ACCURATE, FAST

Specifying the ACCURATE option with the FAST option sets the values of other options as follows:

- ABSVAR = 0.2
- ACCURATE = 1
- BYPASS = 2
- BYTOL = 50u

- DVDT = 2
- FAST = 1
- FFT\_ACCU = 1
- FT = 0.2
- LVLTIM = 3
- MBYPASS = 1.0
- RELMOS = 0.01
- RELVAR = 0.2
- RMAX = 2
- SLOPETOL = 500m

**Note:** The ACCURATE and FAST options are order-independent.

---

## GEAR Method, FAST

Specifying .OPTION METHOD=GEAR in combination with the FAST option sets the values of other options as follows:

- BYTOL = 50u
- DVDT = 3
- FAST = 1
- LVLTIM = 2
- MBYPASS = 2
- METHOD = 0.01
- RMAX = 2
- SLOPETOL = 500m

**Note:** The METHOD=GEAR and FAST options are order-independent.

## GEAR Method, ACCURATE, FAST

Specifying .OPTION METHOD=GEAR first in combination with the ACCURATE and FAST options sets the values of other options as follows:

- ABSVAR = 0.2
- ACCURATE = 1
- BYPASS = 2
- BYTOL = 50u
- DVDT = 2
- FAST = 1
- FFT\_ACCU = 1
- FT = 0.2
- LVLTIM = 3
- METHOD = 2
- MBYPASS = 1.0
- RELMOS = 0.01
- RELVAR = 0.2
- RMAX = 2
- SLOPETOL = 500m

**Note:** If GEAR is specified first, then DVDT=2 LVLTIM=3. Otherwise, the METHOD=GEAR, ACCURATE, and FAST options are order-independent.

---

## RUNLVL=N

Specifying the RUNLVL option with any legal numeric value sets the following options:

- BYPASS = 2 (If METHOD=GEAR with RUNLVL=0, then BYPASS=0)
- DVDT = 3
- LVLTIM = 4

- RUNLVL = N
- SLOPETOL = 500m

---

## RUNLVL, ACCURATE, FAST, GEAR method

Specifying the options RUNLVL, ACCURATE, and FAST with METHOD=GEAR is order-independent:

- RUNLVL option (LVLTIM = 4) is always on
- GEAR method is always selected
- RUNLVL = 5 is always selected
- FAST has no effect on RUNLVL

---

## DVDT=1,2,3

Specifying the DVDT option= 1,2,3 sets the following options:

- BYPASS = 0
- BYTOL = 50u
- MBYPASS = 1.0
- RMAX = 2
- SLOPETOL = 500m

---

## LVLTIM=0,2,3

Specifying the LVLTIM option= 1,2,3 sets the following options:

- BYPASS = 0
- BYTOL = 50u
- MBYPASS = 1.0
- RMAX = 2
- SLOPETOL = 500m

These options are order-independent.

**Note:** The DVDT value is ignored if LVLTIM = 2

---

## KCLTEST

Specifying the KCLTEST option sets the following options:

- ABSTOL = 1u
- RELI = 1u

KCLTEST is order-dependent with ABSTOL and RELI.

---

## BRIEF

Specifying the BRIEF option resets the following options to their defaults:

- NODE
- LIST
- OPTS

and sets the NOMOD option.

The BRIEF option is order-dependent with the affected options. If option BRIEF is specified after NODE, LIST, OPTS, and NOMOD, then it resets them. If option BRIEF is specified before NODE, LIST, OPTS, and NOMOD, then those options overwrite whatever values option BRIEF may have set.

---

## Option Notes

- ABSTOL aliases ABSI
- VNTOL aliases ABSV
- If ABSVDC is not set, VNTOL sets it
- DCTRAN aliases CONVERGE
- GMIN does not overwrite GMINDC, nor does GMINDC overwrite GMIN

- RELH only takes effect when ABSH is non-zero
- RELTOL aliases RELV
- RELVDC defaults to RELTOL
- If RELTOL < BYTOL, BYTOL = RELTOL
- RELVAR applies to LVLTIM = 1 or 3 only
- CHGTOL, RELQ & TRTOL are the only error tolerance options for LVLTIM = 2 (LTE)
- The DVDT algorithm works with LVLTIM = 1 and 3

---

## RUNLVL Option Notes

If RUNLVL is invoked, you can disable it by:

- Adding .OPTION RUNLVL=0 to your current simulation job.
- Copying \$installdir/hspice.ini to your HOME directory and customize it by adding .OPTION RUNLVL=0, which disables it for all of your simulation jobs.
- Re-invoking the \$installdir/bin/config program and deselecting the option runlvl setting in box 'hspice.ini' which disables it for the whole group of simulation jobs.

If RUNLVL is invoked, some options are ignored or automatically set:

Options below are automatically set (user setting will overwrite them):

- If runlvl=6, then .option bypass=0
- If runlvl=1|2|3|4|5, then .option bypass=2
- The following options are ignored; they are replaced by automated algorithms: lvltim, dvdt, ft, fast, trtol, absvar, relvar, relq, chgtol, dvtr, imin, itl3, rmax

If RUNLVL is invoked, actual values of options used by HSPICE are:

- runlvl= 3
- bypass= 2
- mbypass= 2.00
- bytol= 100.00u

## Appendix B: How Options Affect other Options

### Finding the Golden Reference for Options

- `bdfatol=1e-3`
- `bdfrtol=1e-3`

---

## Finding the Golden Reference for Options

When trying to determine the acceptable trade-off between HSPICE accuracy and transient analysis simulation performance, it is important to first establish a reference value for the measurements you are using to evaluate the performance (speed and accuracy) of a given HSPICE configuration. There are multiple ways to configure HSPICE for higher accuracy. The following is a good starting point that you might want to modify for your specific application:

```
.OPTION RUNLVL=6 ACCURATE KCLTEST DELMAX=a_small_value
```

The options are described as follows:

---

Options	Description
RUNLVL	Invokes the RUNLVL algorithm and sets tolerances to their tightest values. Refer to <a href="#">.OPTION RUNLVL</a> for more details:
ACCURATE	Sets even more HSPICE OPTIONS to tighter tolerances. (See <a href="#">.OPTION ACCURATE</a> for details.)
KCLTEST	Activates Kirchhoff's Current Law testing for every circuit node. (See <a href="#">.OPTION KCLTEST</a> for details.)
DELMAX	Sets the largest timestep that HSPICE is allowed to take. It should be set to the smallest value (1ps, for example) that still allows the simulation to finish in a reasonable amount of time. Typically, it should be set approximately <i>1 / (20 * highest-frequency-activity-in-the-circuit)</i> Warning: This option can create very large <i>tr0</i> files. Be careful to only probe the needed nodes (use <code>.OPTION PROBE</code> combined with <code>.PROBE</code> ). See <a href="#">.OPTION DELMAX</a> for details.

---



## Symbols

(X0R, X0I) option 727  
(X1R, X1I) option 728  
(X2R, X2I) option 729

## A

ABSH option 352  
ABSI option 353, 514  
ABSMOS option 355, 514  
ABSOUT optimization bisection parameter 216  
ABSTOL option 356  
ABSV option 357  
ABSVAR option 358  
ABSVDC option 359  
AC analysis  
  magnitude 361  
  optimization 25  
  output 361  
  phase 361  
.AC command 25  
  external data 65  
ACCT option 780  
ACCURATE option 360  
  combined with FAST option 802  
  combined with FAST option and GEAR method 804  
  combined with GEAR option 801, 802  
  plus FAST and RUNLVL options and METHOD=GEAR 805  
.ACMATCH command 28  
ACOUT option 361  
algorithms  
  DVDT 358, 531  
  local truncation error 531, 616, 708  
  pivoting 594  
  timestep control 448  
  transient analysis timestep 531  
  trapezoidal integration 546  
.ALIAS command 31  
ALL keyword 233, 266

ALT9999 option 781  
ALTCC option 362  
ALTCHK option 363  
alter block commands 15  
ALTER cases, multiprocessing 5  
.ALTER command 33, 81  
Analog Artist interface 603  
  *See also* Artist  
Analysis commands 14  
analysis, network 776  
APPENDALL option 364  
.APPENDMODEL 35  
arguments, command-line  
  hspice 2  
  hspicerf 12  
arithmetic expression 185  
ARTIST option 366  
ASCII output 12  
ASCII output data 544, 783, 795  
ASPEC option 367  
AT keyword 175, 180  
AUTOSTOP option 368  
average nodal voltage, with .MEASURE 187  
average value, measuring 188  
AVG keyword 187, 199

## B

BADCHR option 387, 392  
BETA keyword 265  
.BIASCHK command 38  
BIASFILE option 393  
BIASINTERVAL option 394  
BIASNODE option 395  
BIASPARALLEL option 396  
BIAWARN option 397  
BINPRNT option 398  
bisection  
  pushout 203  
BKPSIZ option 782

## Index

### C

BPNMATCHTOL option 399  
branch current error 353  
breakpoint table, size 782  
BRIEF option 233, 236, 400, 526, 565, 574, 581  
  effect on other options 806  
BSIM4PDS option 401  
bus notation 759  
BYPASS option 402  
BYTOL option 403

### C

#### Cadence

  Opus 783, 795  
  WSF format 783, 795

#### capacitance

  charge tolerance, setting 406  
  CSHUNT node-to-ground 417  
  table of values 404

#### capacitor, models 214

#### CAPTAB option 404

#### CDS option 783

#### .CFL\_PROTOTYPE 44

#### C-function library 44

#### characterization of models 73

#### charge tolerance, setting 406

#### .CHECK EDGE command 48

#### .CHECK FALL command 50

#### .CHECK GLOBAL\_LEVEL command 51

#### .CHECK HOLD command 52

#### .CHECK IRDROP command 54

#### .CHECK RISE command 56

#### .CHECK SETUP command 58

#### .CHECK SLEW command 59

#### CHGTOL option 406

#### CLOSE optimization parameter 215

#### CMIFLAG option 408

#### CMIPATH option 409

#### CMIUSRFLAG option 410

#### CO option 314, 317, 779, 784

#### column laminated data 69

#### command-line arguments

  hspice 2

  hspicerf 12

#### commands

  .AC 25

  .ACMATCH 28

.ALIAS 31

.ALTER 33, 81

alter block 15

analysis 14

.APPENDMODEL 35

.BIASCHK 38

.CFL\_PROTOTYPE 44

.CHECK EDGE 48

.CHECK FALL 50

.CHECK GLOBAL\_LEVEL 51

.CHECK HOLD 52

.CHECK IRDROP 54

.CHECK RISE 56

.CHECK SETUP 58

.CHECK SLEW 59

.CONNECT 61

.DATA 64

.DC 71

.DCMATCH 76

.DCVOLT 80

.DEL LIB 81

.DISTO 86

.DOUT 88

.EBD 90

.ELSE 92

.ELSEIF 93

.END 94

.ENDDATA 95

.ENDIF 96

.ENDL 97

.ENDS 98

.ENV 99

.ENVFFT 100

.ENVOSC 101

.EOM 102

.FFT 103

.FLAT 108

.FOUR 110

.FSOPTIONS 111

.GLOBAL 114

.GRAPH 771

.HB 115

.HBAC 118

.HBLIN 119

.HBLSP 121

.HBNOISE 123

.HBOSC 126

.HBXF 131

- .HDL 132
- .IBIS 134
- .IC 138
- .ICM 140
- .IF 142
- .INCLUDE 144
- .LAYERSTACK 148
- .LIB 150
- .LIN 154
- .LOAD 158
- .LPRINT 160
- .MACRO 164
- .MALIAS 167
- .MATERIAL 169
- .MEASURE 171
- .MEASURE PHASENOISE 198, 199
- .MEASURE PTDNOISE 202
- .MEASURE(ACMATCH) 205
- .MEASURE(DCMATCH) 206
- .MODEL 213
- .MOSRA 222
- .MOSRAPRINT 226
- .NET 775
- .NODESET 228
- .NOISE 230
- .OP 233
- .OPTION 235
- .PARAM 237
- .PAT 241
- .PHASENOISE 244
- .PKG 247
- .PLOT 777
- .POWER 249
- .POWERDC 251
- .PRINT 252
- .PROBE 255
- .PROTECT 258
- .PTDNOISE 259
- .PZ 263
- .SAVE 266
- .SENS 268
- .SHAPE 270
- .SNFT 280
- .SNOSC 285
- .SNXF 288
- .STATEYE 290
- .STIM 294
- subcircuit 18
- .SUBCKT 301
- .SURGE 306
- .SWEEPBLOCK 307
- .TEMP (or) .TEMPERATURE 309
- .TF 312
- .TITLE 313
- .TRAN 314
- .UNPROTECT 324
- .VARIATION 325
- .VEC 327
- Verilog-A 19
- .WIDTH 779
- Common Simulation Data Format 446
- concatenated data files 68
- Conditional Block 15
- conductance
  - current source, initialization 465
  - minimum, setting 466
  - models 427
  - MOSFETs 467
  - negative, logging 445
  - node-to-ground 471
  - sweeping 468
- .CONNECT command 61
- control options
  - printing 581
  - setting 236
  - transient analysis
    - limit 712
- CONVERGE option 412, 428
- convergence
  - for optimization 216
  - problems
    - changing integration algorithm 546
    - CONVERGE option 412, 428
    - DCON setting 426
    - decreasing the timestep 459
    - operating point Debug mode 233
    - steady state 468
- CPTIME option 413
- CPU time, reducing 566
- CROSS keyword 179, 183
- CSCAL option 414, 529
- CSDF option 415
- CSHDC option 416
- CSHUNT option 417
- current
  - ABSMOS floor value for convergence 615

## Index

### D

- branch 353
  - operating point table 233
- CURRENT keyword 233
- current threshold option 513
- CUSTCMI option 418
- CUT optimization parameter 216
- CVTOL option 419

### D

- d argument 3
- D\_IBIS option 420
- .DATA command 64
  - datanames 66
  - external file 64
  - for sweep data 65
  - inline data 66
- data files, disabling printout 400, 574
- DATA keyword 25, 65, 71, 315
- datanames 66, 295
- DC
  - analysis
    - decade variation 72
    - initialization 424
    - iteration limit 505
    - linear variation 72
    - list of points 72
    - octave variation 72
  - optimization 71
- .DC command 71, 73
  - external data with .DATA 65
- DCAP option 421
- DCCAP option 422
- DCFOR option 423
- DCHOLD option 424
- DCIC option 425
- .DCMATCH command 76
- DCON option 426
- DCSTEP option 427
- DCTRAN option 428
- .DCVOLT command 80, 138
- DEBUG keyword 233
- DEC keyword 26, 72, 316
- DEFAD option 429
- DEFAS option 430
- DEFL option 431
- DEFNRD option 432

- DEFNRS option 433
- DEFPD option 434
- DEFPS option 435
- DEFSA option 436
- DEFSEB option 437
- DEFSD option 438
- DEFW option 439
- .DEL LIB command 81
  - with .ALTER 81
  - with .LIB 81
- delays
  - group 709
- DELMAX option 443, 627
- DELTA internal timestep
  - See also* timestep
- demo files
  - MOSFETs 27
  - transmission (W-element) lines 113, 149, 170, 270
- derivative function 193
- DERIVATIVE keyword 193
- derivatives, measuring 180
- DI option 444
- DIAGNOSTIC option 445
- DIFSIZ optimization parameters 216
- digits, significant 539
- diode models 214
- .DISTO command 86
- distributed processing 5
- DLENCSDF option 446
- .DOUT command 88
- DP option 5
- DV option 426, 447
- DVDT
  - algorithm 358
  - option 448, 531
- DVDT option 448
- DVDToption
  - value e1,2,3 effect on other options
  - .OPTION DVDT
    - value 1,2,3 effect on other options
- 805
- DVTR option 449

### E

- .EBD command 90

element  
   checking, suppression of 566  
   OFF parameter 575  
 .ELSE command 92  
 .ELSEIF command 93  
 EM\_RECOVERY option 451  
 ENABLE command 736  
 Encryption 16  
 .END command 94  
   for multiple HSPICE runs 94  
   location 94  
 .ENDDATA command 95  
 ENDDATA keyword 64, 68  
 .ENDIF command 96  
 .ENDL command 97, 151  
 .ENDS command 98  
 .ENV command 99  
 envelope simulation 99  
   FFT on output 100  
   oscillator startup, shutdown 101  
 .ENVFFT command 100  
 .ENVOSC command 101  
 .EOM command 102  
 EPSMIN option 452  
 equation 185  
 ERR function 196, 197  
 ERR1 function 196  
 ERR2 function 196  
 ERR3 function 196  
 error function 196  
 errors  
   branch current 353  
   function 197  
   internal timestep too small 417, 628  
   optimization goal 174  
   tolerances  
     ABSMOS 355  
     branch current 353  
     RELMOS 355  
 EXPLI option 453  
 EXPMAX option 454  
 expression, arithmetic 185  
 external data files 66

## F

FALL keyword 179, 183

fall time  
   verification 50  
 FAST option 455  
   effect on other options 800  
 FASToption  
   combined with ACCURATE option 802  
   combined with ACCURATE option and GEAR  
   method 804  
   combined with GEAR method 803  
   plus ACCURATE and RUNLVL options and  
   METHOD=GEAR 805  
 .FFT command 103  
 FFT\_ACCURATE option 456  
 FFTOUT option 457  
 FIL keyword 66  
 files  
   column lamination 69  
   concatenated data files 68  
   filenames 66  
   hspice.ini 544  
   include files 144, 152  
   input 2  
   multiple simulation runs 94  
   output  
     version number 3, 12  
 FIND keyword 180  
 FIND, using with .MEASURE 178  
 .FLAT command 108  
 floating point overflow  
   CONVERGE setting 412  
   setting GMINDC 467  
 FMAX option 458  
 .FOUR command 110  
 FREQ  
   model parameter 773  
 frequency  
   ratio 86  
   sweep 27  
 FROM parameter 196  
 FS option 265, 459  
 FSCAL option 460  
 .FSOPTIONS command 111  
 FT option 461  
 functions  
   ERR 197  
   ERR1 196  
   ERR2 196  
   ERR3 196

## Index

### G

error 196

### G

GDCPATH option 462

GEAR method

combined with FAST option 803

effect on options 800

GEAR option

combined with ACCURATE option 801, 802

effect on other options 800

GENK option 463

GEOSHRINK option 464

.GLOBAL command 114

global node names 114

GMAX option 465

GMIN option 466, 467

GMINDC option 467

GOAL keyword 188

GRAD optimization parameter 216

GRAMP

calculation 426

option 468

.GRAPH command 771

graph data file (Viewlogic format) 446

ground bounce checking 54

group delay, calculating 709

GSCAL option 469

GSHDC option 470

GSHUNT option 471

### H

-h argument

usage information 12

H9007 option 786

harmonic balance analysis 116

harmonic balance noise analysis 125

harmonic balance transfer analysis 131, 288

harmonic balance-based periodic AC analysis 118

.HB command 115

HB\_GIBBS option 478

.HBAC command 118

HBACKRYLOVDIM option 472

HBACKRYLOVITR option 473

HBACTOL option 474

HBCONTINUE option 475

HBFREQABSTOL option 476

HBFREQRELTOL option 477

HBJREUSE option 479

HBJREUSETOL option 480

HBKRYLOVDIM option 481

HBKRYLOVMAXITER option 483

HBKRYLOVTOL option 482

.HBLIN command 119

HBLINESEARCHFAC option 484

.HBLSP command 121

HBMAXITER option 485

.HBNOISE command 123

.HBOSC command 126

HBOSCMAXITER option 486

HBPROBETOL option 487

HBSOLVER option 488

HTOL option 489

HBTRANFREQSEARCH option 490

HBTRANINIT option 491

HBTRANPTS option 492

HBTRANSTEP option 493

.HBXF command 131

HCI and NBTI analysis 225

.HDL command 132

HIER\_DELIM option 494

HIER\_SCALE option 495

HSPICE

job statistics report 780

version

H9007 compatibility 786

hspice

arguments 2

command 2

hspice.ini file 544

hspicrf

arguments 12

command 12

-html argument 3

### I

-I argument 4

-i argument 2

.IBIS command 134

IBIS commands 17

.IC command 80, 138

- from .SAVE 267
- IC keyword 266
- IC parameter 80
- .ICM command 140
- ICSWEEP option 498
- IDELAY command 737
- .IF command 142
- IGNOR keyword 196
- IMAX option 499, 508
- IMIN option 500, 507
- .INCLUDE command 144
- include files 144, 152
- indepout 295
- indepvar 295
- inductors, mutual model 214
- INGOLD option 501, 539
- initial conditions
  - saving and reusing 498
- initialization 575
- inline data 66
- input
  - data
    - adding library data 81
    - column laminated 69
    - concatenated data files 68
    - deleting library data 81
    - external, with .DATA command 65
    - filenames on networks 70
    - formats 66, 69
    - include files 144
    - printing 526
    - suppressing printout 526
  - file names 2
  - netlist file 94
- INTEG keyword 187, 191, 199
  - used with .MEASURE 187
- integral function 191
- integration
  - backward Euler method 535
- interfaces
  - Analog Artist 603
  - Mentor 788
  - MSPICE 788
  - ZUKEN 798
- INTERP option 503
- IO command 739
- iterations

- limit 505
  - maximum number of 509
- ITL1 option 505
- ITL2 option 506
- ITL3 option 507
- ITL4 option 508
- ITL5 option 509
- ITLPTRAN option 510
- ITLPZ option 511
- ITROPT optimization parameter 216
- ITRPRT option 512
- IVTH option 513

## J

- Jacobian data, printing 579

## K

- KCLTEST option 514
- KCLTESToption
  - effect on other options
    - .OPTION KCLTEST
      - effect on other options 806
- keywords
  - .AC command parameter 25
  - ALL 233, 266
  - AT 175, 180
  - AVG 187, 199
  - BETA 265
  - CROSS 179, 183
  - CURRENT 233
  - DATA 25, 65, 71, 315
    - .DATA command parameter 65
    - .DC command parameter 71
  - DEBUG 233
  - DEC 26, 72, 316
  - DERIVATIVE 193
  - ENDDATA 64, 68
  - FALL 179, 183
  - FIL 66
  - FIND 180
  - FS 265
  - IGNOR 196
  - INTEG 187, 191, 199
  - LAM 66, 69
  - LAST 179, 184
  - LIN 26, 72, 316

## Index

### L

MAXFLD 265  
.MEASUREMENT command parameter 187, 199  
MER 66, 69  
MINVAL 196  
MODEL 71  
MONTE 26, 72, 315  
NONE 233, 266  
NUMF 265  
OCT 26, 72, 316  
OPTIMIZE 72  
POI 26, 72, 316  
PP 187, 199  
RESULTS 72  
RIN 775  
RISE 179, 183  
START 315  
SWEEP 26, 72, 315  
target syntax 175, 180  
TO 188, 191, 197  
TOL 265  
TOP 266  
.TRAN command parameter 315  
TRIG 173  
VOLTAGE 233  
WEIGHT 188, 196  
weight 188  
WHEN 180  
Kirchhoff's Current Law (KCL) test 514  
KLIM option 515

### L

LA\_FREQ option 516  
LA\_MAXR option 517  
LA\_MINC option 518  
LA\_TIME option 519  
LA\_TOL option 520  
LAM keyword 66, 69  
laminated data 69  
LAST keyword 179, 184  
latent devices  
excluding 455  
.LAYERSTACK command 148  
LENNAM option 521  
.LIB command 150  
call command 150  
in .ALTER blocks 151

nesting 151  
with .DEL LIB 81  
libraries  
adding with .LIB 81  
building 151  
deleting 81  
private 258  
protecting 258  
Library Management 17  
LIMPTS option 522  
LIMTIM option 523  
.LIN command 154  
LIN keyword 26, 72, 316  
LIST option 526  
listing, suppressing 258  
.LOAD command 158  
LOADHB option 527  
LOADSNINIT option 528  
local truncation error algorithm 531, 616, 708  
.LPRINT command 160  
LVLTIM option 531, 708  
value 0,2,3 effect on other options 805

## M

MACMOD option 532  
.MACRO command 164  
macros 81  
magnetic core models 214  
.MALIAS command 167  
.MATERIAL command 169  
Material Properties 16  
matrix  
minimum pivot values 595  
parameters 775  
row/matrix ratio 793  
size limitation 791  
MAX 187  
MAX parameter 187, 199, 215  
MAXAMP option 534  
MAXFLD keyword 265  
maximum value, measuring 188  
MAXORD option 535  
MBYPASS option 537  
MCBRIEF option 538  
MEASDGT option 539



MEASFAIL option 540  
 MEASFILE option 541  
 MEASOUT option 544  
 MEASSORT option 787  
 .MEASURE command 171, 539, 544  
   average nodal voltage 187  
   expression 185  
   propagation delay 173  
 .MEASURE PHASENOISE 198, 199  
 .MEASURE(ACMATCH) command 205  
 .MEASURE(DCMATCH) command 206  
 measuring average values 188  
 measuring derivatives 180  
 Mentor interface 788  
 MENTOR option 788  
 MER keyword 66, 69  
 messages  
   *See also* errors, warnings  
 METHOD option 546  
 MIN 187  
 MIN parameter 187, 199  
 minimum value, measuring 188  
 MINVAL keyword 196  
 .MODEL command 213  
   ABSOUTT 216  
   CLOSE 215  
   CUT 216  
   DEV 217  
   DIFSIZ 216  
   distribution 217  
   GRAD 216  
   ITROPT 216  
   keyword 217  
   LOT 217  
   MAX 215  
   model name 213  
   PARMIN 216  
   RELIN 216  
   RELOUT 216  
   type 214  
 .MODEL command for .GRAPH 773  
 MODEL keyword 71  
 model parameters  
   .GRAPH command parameters 773  
   MONO 773  
   output 773  
   suppressing printout of 568

TEMP 309  
 TIC 773  
 models  
   BJTs 214  
   capacitors 214  
   characterization 73  
   diode 214  
   JFETs 214  
   magnetic core 214  
   MOSFETs 214  
   mutual inductors 214  
   names 213  
   npn BJT 214  
   op-amps 214  
   optimization 214  
   plot 214  
   private 258  
   protecting 258  
   simulator access 151  
   types 214  
 models, diode 214  
 MODMONTE option 549  
 MODSRH option 789  
 MONO model parameter 773  
 Monte Carlo  
   AC analysis 25  
   DC analysis 71  
   .MODEL parameters 217  
   time analysis 314  
 MONTE keyword 26, 72, 315  
 MONTECON option 553  
 .MOSRA command 222  
 MOSRALIFE option 554  
 .MOSRAPRINT command 226  
 MOSRASORT option 555  
 MSPICE simulator interface 788  
 -mt argument 5, 6  
 MTTHRESH option 560  
 MU option 561  
 multiprocessing, ALTER cases 5  
 multithreading, lowering device number threshold  
   560

## N

-n argument 3, 12  
 namei 295

## Index

### O

- NBTI and HCI analysis 225
  - NCFILTER option 562
  - n-channel, MOSFET's models 214
  - NCWARN option 563
  - negative conductance, logging 445
  - nested library calls 151
  - .NET comamnd 775
  - network
    - analysis 776
    - filenames 70
  - network analysis 776
  - NEWTOL option 564
  - Node Naming 17
  - NODE option 565
  - nodes
    - cross-reference table 565
    - global versus local 114
    - printing 565
  - .NODESET command 228
    - from .SAVE 267
  - NODESET keyword 266
  - NOELCK option 566
  - noise
    - folding 265
    - numerical 417
    - sampling 265
  - .NOISE command 230
  - NOISEMINFREQ option 567
  - NOMOD option 568
  - NONE keyword 233, 266
  - NOPAGE option 792
  - NOPIV option 569
  - NOTOP option 570
  - NOWARN option 571
  - npn BJT models 214
  - npoints 295
  - NUMDGT option 572
  - numerical integration algorithms 546
  - numerical noise 417, 471
  - NUMERICAL\_DERIVATIVES option 573
  - NUMF keyword 265
  - NXX option 574
- O**
- ODELAY command 740
  - OFF option 575
  - .OP command 233
  - op-amps model, names 214
  - operating point
    - capacitance 404
    - .IC command initialization 80
    - restoring 158
    - solution 575
    - voltage table 233
  - OPFILE option 576
  - OPTCON option 577
  - optimization
    - AC analysis 25
    - DC analysis 71
    - error function 174
    - iterations 216
    - models 214
    - time
      - analysis 315
  - optimization parameter, DIFSIZ 216
  - OPTIMIZE keyword 72
  - .OPTION (X0R, X0I) 727
  - .OPTION (X1R, X1I) 728
  - .OPTION (X2R, X2I) 729
  - .OPTION ABSH 352
  - .OPTION ABSI 353
  - .OPTION ABSMOS 355
  - .OPTION ABSTOL 356
  - .OPTION ABSV 357
  - .OPTION ABSVAR 358
  - .OPTION ABSVDC 359
  - .OPTION ACCT 780
  - .OPTION ACCURATE 360
    - combined with FAST option 802
    - combined with FAST option and GEAR method 804
    - combined with GEAR option 801, 802 plus FAST and RUNLVL options, METHOD=GEAR 805
  - .OPTION ACOUT 361
  - .OPTION ALT9999 781
  - .OPTION ALTCC 362
  - .OPTION ALTCHK 363
  - .OPTION APPENDALL 364
  - .OPTION ARTIST 366
  - .OPTION ASPEC 367

- .OPTION AUTOSTOP 368
- .OPTION BADCHR 387, 392
- .OPTION BDFATOL 388, 390
- .OPTION BIASFILE 393
- .OPTION BIASINTERVAL 394
- .OPTION BIASNODE 395
- .OPTION BIASPARALLEL 396
- .OPTION BIAWARN 397
- .OPTION BINPRNT 398
- .OPTION BISAAC 354
- .OPTION BKPSIZ 782
- .OPTION BPNMATCHTOL 399
- .OPTION BRIEF 233, 236, 400, 526, 565, 574, 581
  - effect on other options 806
- .OPTION BSIM4PDS 401
- .OPTION BYPASS 402
- .OPTION BYTOL 403
- .OPTION CAPTAB 404
- .OPTION CDS 783
- .OPTION CHGTOL 406
- .OPTION CMIFLAG 408
- .OPTION CMIPATH 409
- .OPTION CMIUSRFLAG 410
- .OPTION CO 314, 317, 779, 784
- .OPTION command 235
- .OPTION CONVERGE 412
- .OPTION CPTIME 413
- .OPTION CSCAL 414, 529
- .OPTION CSDF 415
- .OPTION CSHDC 416
- .OPTION CSHUNT 417
- .OPTION CUSTCMI 418
- .OPTION CVTOL 419
- .OPTION D\_IBIS 420
- .OPTION DCAP 421
- .OPTION DCCAP 422
- .OPTION DCFOR 423
- .OPTION DCHOLD 424
- .OPTION DCIC 425
- .OPTION DCSTEP 427
- .OPTION DCTRAN 428
- .OPTION DEFAD 429
- .OPTION DEFAS 430
- .OPTION DEFL 431
- .OPTION DEFNRD 432
- .OPTION DEFNRS 433
- .OPTION DEFPD 434
- .OPTION DEFPS 435
- .OPTION DEFSA 436
- .OPTION DEFSB 437
- .OPTION DEFSD 438
- .OPTION DEFW 439
- .OPTION DELMAX 443
- .OPTION DI 444
- .OPTION DIAGNOSTIC 445
- .OPTION DLENCSDF 446
- .OPTION DV 447
- .OPTION DVDT 448
- .OPTION DVTR 449
- .OPTION EM\_RECOVERY 451
- .OPTION EPSMIN 452
- .OPTION EXPLI 453
- .OPTION EXPMAX 454
- .OPTION FAST 455
  - combined with ACCURATE option 802
  - combined with ACCURATE option and GEAR method 804
  - combined with GEAR method 803
  - effect on other options 800
  - plus ACCURATE and RUNLVL options and METHOD=GEAR 805
- .OPTION FFT\_ACCURATE 456
- .OPTION FFTOUT 457
- .OPTION FMAX 458
- .OPTION FS 459
- .OPTION FSCAL 460
- .OPTION FT 461
- .OPTION GDCPATH 462
- .OPTION GEAR
  - combined with ACCURATE option 801, 802
  - effects on other options 800
- .OPTION GENK 463
- .OPTION GEOSHRINK 464
- .OPTION GMAX 465
- .OPTION GMIN 466
- .OPTION GMINDC 467
- .OPTION GRAMP 468
- .OPTION GSCAL 469
- .OPTION GSHDC 470
- .OPTION GSHUNT 471

## Index

### O

- .OPTION H9007 786
- .OPTION HB\_GIBBS 478
- .OPTION HBACKRYLOVDIM 472
- .OPTION HBACKRYLOVITR 473
- .OPTION HBACTOL 474
- .OPTION HBCONTINUE 475
- .OPTION HBFREQABSTOL 476
- .OPTION HBFREQRELTOL 477
- .OPTION HBJREUSE 479
- .OPTION HBJREUSETOL 480
- .OPTION HBKRYLOVDIM 481
- .OPTION HBKRYLOVMAXITER 483
- .OPTION HBKRYLOVTOL 482
- .OPTION HBLINESEARCHFAC 484
- .OPTION HBMAXITER 485
- .OPTION HBMAXOSCITER 486
- .OPTION HBPROBETOL 487
- .OPTION HBSOLVER 488
- .OPTION HBTOL 489
- .OPTION HBTRANFREQSEARCH 490
- .OPTION HBTRANINIT 491
- .OPTION HBTRANPTS 492
- .OPTION HBTRANSTEP 493
- .OPTION HIER\_DELIM 494
- .OPTION HIER\_SCALE 495
- .OPTION ICSWEEP 498
- .OPTION IMAX 499
- .OPTION IMIN 500
- .OPTION INGOLD 501
- .OPTION INTERP 503
- .OPTION ITL1 505
- .OPTION ITL2 506
- .OPTION ITL3 507
- .OPTION ITL4 508
- .OPTION ITL5 509
- .OPTION ITLPTRAN 510
- .OPTION ITLPZ 511
- .OPTION ITRPRT 512
- .OPTION IVTH 513
- .OPTION KCLTEST 514
- .OPTION KLIM 515
- .OPTION LA\_FREQ 516
- .OPTION LA\_MAXR 517
- .OPTION LA\_MINC 518
- .OPTION LA\_TIME 519
- .OPTION LA\_TOL 520
- .OPTION LENNAM 521
- .OPTION LIMPTS 522
- .OPTION LIMITIM 523
- .OPTION LIST 526
- .OPTION LOADHB 527
- .OPTION LOADSNINIT 528
- .OPTION LVLTIM 531
  - value 0,2,3 effect on other options 805
- .OPTION MACMOD 532
- .OPTION MAXAMP 534
- .OPTION MAXORD 535
- .OPTION MBYPASS 537
- .OPTION MCBRIEF 538
- .OPTION MEASDGT 539
- .OPTION MEASFAIL 540
- .OPTION MEASFILE 541
- .OPTION MEASOUT 544
- .OPTION MEASSORT 787
- .OPTION MENTOR 788
- .OPTION METHOD 546
- .OPTION METHOD=GEAR
  - combined with FAST option 803
  - effects on other options 800
- .OPTION MODMONTE 549
- .OPTION MODSRH 789
- .OPTION MONTECON 553
- .OPTION MTTHRESH 560
- .OPTION MU 561
- .OPTION NCFILTER 562
- .OPTION NCWARN 563
- .OPTION NEWTOL 564
- .OPTION NODE 565
- .OPTION NOELCK 566
- .OPTION NOISEMINFREQ 567
- .OPTION NOMOD 568
- .OPTION NOPAGE 792
- .OPTION NOPIV 569
- .OPTION NOTOP 570
- .OPTION NOWARN 571
- .OPTION NUMDGT 572
- .OPTION NUMERICAL\_DERIVATIVES 573
- .OPTION NXX 574
- .OPTION OFF 575
- .OPTION OPFILE 576

.OPTION OPTCON 577  
.OPTION OPTLST 579  
.OPTION OPTS 581  
.OPTION PARHIER 582  
.OPTION PATHNUM 583, 588  
.OPTION PHASENOISEAMPM 593  
.OPTION PHASENOISEKRYLOVDIM 586  
.OPTION PHASENOISEKRYLOVITER 587  
.OPTION PHD 591  
.OPTION PHNOISELORENTZ 592  
.OPTION PIVOT 594  
.OPTION PIVREF 791  
.OPTION PIVREL 793  
.OPTION PIVTOL 595  
.OPTION PLIM 794  
.OPTION POST 596  
.OPTION POST\_VERSION 599  
.OPTION POSTLVL 598  
.OPTION POSTTOP 601  
.OPTION PROBE 602  
.OPTION PSF 603  
.OPTION PURETP 604  
.OPTION PUTMEAS 605  
.OPTION PZABS 606  
.OPTION PZTOL 607  
.OPTION RANDGEN 610  
.OPTION RELH 612  
.OPTION RELI 613  
.OPTION RELMOS 615  
.OPTION RELQ 616  
.OPTION RELTOL 617  
.OPTION RELV 618  
.OPTION RELVAR 619  
.OPTION RELVDC 620  
.OPTION RESMIN 623  
.OPTION RISETIME 624  
.OPTION RITOL 626  
.OPTION RMAX 627  
.OPTION RMIN 628  
.OPTION RUNLVL 629  
    N value effect on other options 804  
.OPTION SAVEHB 635  
.OPTION SAVESNINIT 636  
.OPTION SCALE 637  
.OPTION SCALM 638  
.OPTION SDA 795  
.OPTION SEARCH 639  
.OPTION SEED 640  
.OPTION SIM\_ACCURACY 642  
.OPTION SIM\_DELTAI 643  
.OPTION SIM\_DELTAV 644  
.OPTION SIM\_DSPF 645  
.OPTION SIM\_DSPF\_ACTIVE 647  
.OPTION SIM\_DSPF\_INSERTERROR 648  
.OPTION SIM\_DSPF\_LUMPCAPS 649  
.OPTION SIM\_DSPF\_MAX\_ITER 650  
.OPTION SIM\_DSPF\_RAIL 651  
.OPTION SIM\_DSPF\_SCALEC 652  
.OPTION SIM\_DSPF\_SCALER 653  
.OPTION SIM\_DSPF\_VTOL 654  
.OPTION SIM\_LA 656  
.OPTION SIM\_LA\_FREQ 657  
.OPTION SIM\_LA\_MAXR 658  
.OPTION SIM\_LA\_MINC 659  
.OPTION SIM\_LA\_MINMODE 797  
.OPTION SIM\_LA\_TIME 660  
.OPTION SIM\_LA\_TOL 661  
.OPTION SIM\_ORDER 662  
.OPTION SIM\_OSC\_DETECT\_TOL 663  
.OPTION SIM\_POSTAT 664  
.OPTION SIM\_POSTDOWN 665  
.OPTION SIM\_POSTSCOPE 666  
.OPTION SIM\_POSTSKIP 667  
.OPTION SIM\_POSTTOP 668  
.OPTION SIM\_POWER\_ANALYSIS 669  
.OPTION SIM\_POWER\_TOP 670  
.OPTION SIM\_POWERDC\_ACCURACY 671  
.OPTION SIM\_POWERDC\_HSPICE 672  
.OPTION SIM\_POWERPOST 673  
.OPTION SIM\_POWERSTART 674  
.OPTION SIM\_POWERSTOP 675  
.OPTION SIM\_SPEF 676  
.OPTION SIM\_SPEF\_ACTIVE 677  
.OPTION SIM\_SPEF\_INSERTERROR 678  
.OPTION SIM\_SPEF\_LUMPCAPS 679  
.OPTION SIM\_SPEF\_MAX\_ITER 680  
.OPTION SIM\_SPEF\_PARVALUE 681  
.OPTION SIM\_SPEF\_RAIL 682  
.OPTION SIM\_SPEF\_SCALEC 683  
.OPTION SIM\_SPEF\_SCALER 684

## Index

### P

.OPTION SIM\_SPEF\_VTOL 685  
.OPTION SIM\_TG\_THETA 686  
.OPTION SIM\_TRAP 687  
.OPTION SLOPETOL 689  
.OPTION SNACCURACY 690  
.OPTION SNCONTINUE 691  
.OPTION SNMAXITER 692  
.OPTION SPMODEL 696  
.OPTION STATFL 697  
.OPTION SYMB 700  
.OPTION TIMERES 701  
.OPTION TMIFLAG 702  
.OPTION TNOM 705  
.OPTION TRANFORHB 706  
.OPTION TRCON 707  
.OPTION TRTOL 708  
.OPTION UNWRAP 709  
.OPTION VAMODEL 710  
.OPTION VERIFY 711  
.OPTION VFLOOR 712  
.OPTION VNTOL 713  
.OPTION WACC 714  
.OPTION WARNLIMIT 717  
.OPTION WDELAYOPT 719  
.OPTION WDF 720  
.OPTION WINCLUDEEGDIMAG 722  
.OPTION WL 723  
.OPTION WNFLAG 724  
.OPTION XDTEMP 725  
.OPTION ZUKEN 798  
options  
  BDFATOL 388, 390  
  BISACC 354  
  CAPTAB 404  
  IVTH 513  
  MOSRALIFE 554  
  MOSRASORT 555  
  NCFILTERr 562  
  PHD 591  
  TMIFLAG 702  
  WDF 720  
OPTLST option 579  
OPTS option 581  
Opus 783, 795  
oscillation, eliminating 546

oscillator analysis 129, 246  
OUT, OUTZ command 742  
Output 18  
output  
  ASCII 12  
  data  
    format 539, 603  
    limiting 503  
    significant digits specification 572  
    specifying 522  
    storing 544  
  data, redirecting 9  
  files  
    reducing size of 717  
    version number, specifying 3, 12  
  .MEASURE results 171  
  plotting 777–778  
  printing 253–??  
  printout format 501  
  redirecting 9, 12  
  variables  
    printing 512  
    probing 255  
    specifying significant digits for 572  
ovari 295

### P

.PARAM command 237  
parameters  
  ABSOUT optimization bisection 216  
  AC sweep 25  
  DC sweep 71  
  defaults 582  
  FROM 196  
  IC 80  
  inheritance 582  
  ITROPT optimization 216  
  matrix 775  
  names  
    .MODEL command parameter name 215  
  simulator access 151  
  skew, assigning 152  
  UIC 80, 138  
PARHIER option 582  
PARMIN optimization parameter 216  
.PAT command 241  
path names 583  
path numbers, printing 583

- PATHNUM option 583, 588
  - p-channel
    - JFETs models 214
    - MOSFET's models 214
  - peak-to-peak value
    - measuring 187
  - PERIOD command 743
  - PERIOD statement 743
  - periodic pime-dependent noise analysis 261
  - .PHASENOISE command 244
  - PHASENOISEAMPM option 593
  - PHASENOISEKRYLOVDIM option 586
  - PHASENOISEKRYLOVITER option 587
  - PHNOISELORENTZ option 592
  - pivot
    - algorithm, selecting 594
    - reference 791
  - PIVOT option 594
  - pivot option 594
  - PIVREF option 791
  - PIVREL option 793
  - PIVTOL option 595
  - .PKG command 247
  - PLIM option 794
  - plot
    - models 214
    - value calculation method 361
  - .PLOT command 777
    - in .ALTER block 33
  - pnP BJT models 214
  - POI keyword 26, 72, 316
  - pole-zero
    - (X0R, X0I) option 727
    - (X1R, X1I) option 728
    - (X2R, X2I) option 729
    - CSCAL option 414, 529
    - FSCAL option 460
    - GSCAL option 469
    - PZABS option 606
    - PZTOL option 607
    - RITOL option 626
  - pole-zero analysis
    - FMAX option 458
    - maximum iterations 511
  - polygon, defining 275
  - POST option 596
  - POST\_VERSION option 599
  - POSTLVL option 598
  - POSTTOP option 601
  - .POWER command 249
  - power operating point table 233
  - .POWERDC command 251
  - power-dependent S parameter extraction 122
  - PP 187, 191
  - PP keyword 187, 199
  - .PRINT command 252
    - in .ALTER 33
  - printing
    - Jacobian data 579
  - printout
    - disabling 400, 574
    - suppressing 258
    - value calculation method 361
  - .PROBE command 255
  - PROBE option 602
  - propogation delays
    - measuring 175
    - with .MEASURE 173
  - .PROTECT command 258
  - protecting data 258
  - PSF option 603
  - PTDNOISE
    - overview 261
  - .PTDNOISE command 259
  - PTDNOISE with .MEASURE command 202
  - PURETP option 604
  - pushout bisection 203
  - PUTMEAS option 605
  - .PZ command 263
  - PZABS option 606
  - PZTOL option 607
- ## R
- RADIX scommand 744
  - RANDGEN option 610
  - reference temperature 309
  - RELH option 612
  - RELI option 514, 613
  - RELIN optimization parameter 216
  - RELMOS option 355, 514, 615
  - RELOUT optimization parameter 216
  - RELQ option 616

## Index

### S

RELTOL option 406  
RELTOLoption 617  
RELV option 455, 537, 618  
RELVAR option 619  
RELVDC option 620  
resistance 623  
RESMIN option 623  
RESULTS keyword 72  
RF  
  .MEASURE PTDNOISE 202  
RF commands  
  .SNNOISE 279, 283  
RIN keyword 775  
Rise 173  
rise and fall times 175  
RISE keyword 179, 183  
rise time  
  example 56  
  specify 749, 751  
  verify 56  
RISETIME option 624  
RITOL option 626  
RMAX option 627  
RMIN option 628  
RMS keyword 187, 199  
ROUT keyword 775  
row/matrix ratio 793  
RUNLVL option 629  
  N value effect on other options 804

### S

.SAMPLE 265  
.SAMPLE command 265  
sampling noise 265  
.SAVE command 266  
SAVEHB option 635  
SAVESNINIT option 636  
SCALE option 637  
SCALM option 638  
SDA option 795  
SEARCH option 639  
SEED option 640  
.SENS command 268  
Setup 18  
.SHAPE command 270

  Defining Circles 272  
  Defining Polygons 273  
  Defining Rectangles 271  
  Defining Strip Polygons 275  
Shooting Newton syntaxes 277  
significant digits 539  
SIM\_ACCURACY option 642  
SIM\_DSPF option 645  
SIM\_DSPF\_ACTIVE option 647  
SIM\_DSPF\_DELTAI option 643  
SIM\_DSPF\_DELTAV option 644  
SIM\_DSPF\_INSERTERROR option 648  
SIM\_DSPF\_LUMPCAPS option 649  
SIM\_DSPF\_MAX\_ITER option 650  
SIM\_DSPF\_RAIL option 651  
SIM\_DSPF\_SCALEC option 652  
SIM\_DSPF\_SCALER option 653  
SIM\_DSPF\_VTOL option 654  
SIM\_LA option 656  
SIM\_LA\_FREQ option 657  
SIM\_LA\_MAXR option 658  
SIM\_LA\_MINC option 659  
SIM\_LA\_MINMODE option 797  
SIM\_LA\_TIME option 660  
SIM\_LA\_TOL option 661  
SIM\_ORDER option 662  
SIM\_OSC\_DETECT\_TOL option 663  
SIM\_POSTAT option 664  
SIM\_POSTDOWN option 665  
SIM\_POSTSCOPE option 666  
SIM\_POSTSKIP option 667  
SIM\_POSTTOP option 668  
SIM\_POWER\_ANALYSIS option 669  
SIM\_POWER\_TOP option 670  
SIM\_POWERDC\_ACCURACY option 671  
SIM\_POWERDC\_HSPICE option 672  
SIM\_POWERPOST option 673  
SIM\_POWERSTART option 674  
SIM\_POWERSTOP option 675  
SIM\_SPEF option 676  
SIM\_SPEF\_ACTIVE option 677  
SIM\_SPEF\_INSERTERROR option 678  
SIM\_SPEF\_LUMPCAPS option 679  
SIM\_SPEF\_MAX\_ITER option 680  
SIM\_SPEF\_PARVALUE option 681



- SIM\_SPEF\_RAIL option 682
- SIM\_SPEF\_SCALEC option 683
- SIM\_SPEF\_SCALER option 684
- SIM\_SPEF\_VTOL option 685
- SIM\_TG\_THETA option 686
- SIM\_TG\_TRAP option 687
- simulation
  - accuracy 360, 531
  - accuracy improvement 448
  - multiple analyses, .ALTER command 33
  - multiple runs 94
  - reducing time 65, 368, 448, 500, 507, 689, 708
  - results
    - plotting 777–778
    - printing 253
    - specifying 171
  - title 313
- Simulation Runs 18
- skew, parameters 152
- slew rate
  - verification 59
- SLEW, .CHECK command 59
- SLOPE command 745
- SLOPETOL option 689
- small-signal, DC sensitivity 268
- .SN command 277
- SNACCURACY option 690
- SNCONTINUE option 691
- .SNFT command 280
- SNMAXITER option 692
- .SNNOISE command 279, 283
- .SNOSC command 285
- .SNXF command 288
- source
  - AC sweep 25
  - DC sweep 71
- S-parameter, model type 214
- SPICE
  - compatibility
    - AC output 361
    - plot 794
- SPMODEL option 696
- START keyword 315
- statements
  - .AC 25
  - .ACMATCH 28
  - .ALIAS 31
  - .ALTER 33, 81
  - alter block 15
  - .BIASCHK 38
  - .CHECK EDGE 48
  - .CHECK FALL 50
  - .CHECK GLOBAL\_LEVEL 51
  - .CHECK HOLD 52
  - .CHECK IRDROP 54
  - .CHECK RISE 56
  - .CHECK SETUP 58
  - .CHECK SLEW 59
  - .CONNECT 61
  - .DATA 64
    - external file 64
    - inline 64
  - .DC 71, 73
  - .DCMATCH 76
  - .DCVOLT 80, 138
  - .DEL LIB 81
  - .DISTO 86, 87
  - .DOUT 88
  - .EBD 90
  - .ELSE 92
  - .ELSEIF 93
  - .END 94
  - .ENDDATA 95
  - .ENDIF 96
  - .ENDL 97, 151
  - .ENDS 98, 102
  - .ENV 99
  - .ENVFFT 100
  - .ENVOSC 101
  - .EOM 102
  - .FFT 103
  - .FOUR 110
  - .FSOPTIONS 111
  - .GLOBAL 114
  - .GRAPH 771
  - .HB 115
  - .HBAC 118
  - .HBLIN 119
  - .HBLSP 121
  - .HBNOISE 123
  - .HBOSC 126
  - .HBXF 131
  - .HDL 132
  - .IBIS 134
  - .IC 80, 138

## Index

### T

.ICM 140  
.IF 142  
.INCLUDE 92, 94, 142, 144, 267  
.LAYERSTACK 148  
.LIB 150, 151  
    nesting 151  
.LIN 154  
.LOAD 158  
.LPRINT 160  
.MACRO 164  
.MALIAS 167  
.MATERIAL 169  
.MEASURE 171, 539, 544  
.MODEL 213  
.MOSRA 222  
.MOSRAPRINT 226  
.NET 775  
.NODESET 228  
.NOISE 230  
.OP 233, 234  
.PARAM 237  
.PAT 241  
.PERIOD 743  
.PHASENOISE 244  
.PKG 247  
.PLOT 777  
.POWER 249  
.POWERDC 251  
.PRINT 252  
.PROBE 255  
.PROTECT 258  
.PZ 263  
.SAMPLE 265  
.SAVE 266  
.SENS 268  
.SHAPE 270  
.SNFT 280  
.SNOSC 285  
.SNXF 288  
.STIM 294  
.SUBCKT 301  
.SURGE 306  
.SWEEPBLOCK 307  
.TEMP 309  
.TF 312  
.TITLE 313  
.TRAN 314  
.UNPROTECT 324  
.VARIATION 325  
.VEC 327  
.WIDTH 779  
.STATEYE command 290  
STATFL option 697  
statistical eye diagram analysis 290  
statistics, listing 780  
.STIM command 294  
subcircuit commands 18  
subcircuits  
    calling 165, 302  
    global versus local nodes 114  
    names 164, 301  
    node numbers 164, 301  
    parameter 98, 102, 164, 165, 301, 302  
    printing path numbers 583  
.SUBCKT command 301  
.SURGE command 306  
sweep  
    data 544  
    frequency 27  
SWEEP keyword 26, 72, 315  
.SWEEPBLOCK command 307  
SYMB option 700

### T

Tabular Data section  
    time interval 743  
TARG\_SPEC 173  
target specification 173, 182  
TDELAY command 747  
TEMP  
    keyword 26, 72  
    model parameter 309  
.TEMP (or) .TEMPERATURE command 309  
temperature  
    AC sweep 25  
    DC sweep 71  
    derating 309, 310  
    reference 309  
.TF command 312  
TFALL command 749  
TIC model parameter 773  
time 233  
    *See also* CPU time  
TIMERES option 701  
timestep

- algorithms 448
  - calculation for DVDT=3 459
  - changing size 616
  - control 459, 619, 708
  - maximum 499, 508, 627
  - minimum 500, 507, 628
  - reversal 358
  - transient analysis algorithm 531
  - .TITLE command 313
  - title for simulation 313
  - TMI flow 702
  - TNOM option 309, 705
  - TO keyword 188, 191, 197
  - TOL keyword 265
  - TOP keyword 266
  - .TRAN command 314
  - TRANFORHB option 706
  - transient analysis
    - Fourier analysis 110
    - initial conditions 80, 138
    - number of iterations 509
  - TRAP algorithm
    - See trapezoidal integration
  - TRCON option 707
  - TRIG keyword 173
  - TRIG\_SPEC 173
  - trigger specification 173, 182
  - TRISE command 749, 751
  - TRIZ command 753
  - TRTOL option 708
  - TSKIP command 754
  - TSTEP
    - multiplier 627, 628
    - option 627, 628
  - TUNIT command 755
- U**
- UIC
    - parameter 80, 138
  - U-lement, transmission line model 214
  - .UNPROTECT command 324
  - UNWRAP option 709
- V**
- v argument
    - version information 12
  - VAMODEL option 710
  - .VARIATION command 325
  - .VEC command 327
  - VEC commands
    - ENABLE 736
    - IDELAY 737
    - IO 739
    - ODELAY 740
    - OUT, OUTZ 742
    - PERIOD 743
    - RADIX 744
    - SLOPE 745
    - TDELAY 747
    - TFALL 749
    - TRISE 751
    - TRIZ 753
    - TSKIP 754
    - TUNIT 755
    - VIH 757
    - VIL 758
    - VNAME 759
    - VOH 761
    - VOL 763
    - VREF 765
    - VTH 766
  - VERIFY option 711
  - Verilog-A commands 19
  - version
    - determining 12
    - H9007 compatibility 786
  - VFLOOR option 712
  - Viewlogic graph data file 446
  - VIH command 757
  - VIL command 758
  - VNAME command 759
  - VNTOL option 455, 713
  - VOH command 761, 763
  - VOL command 763
  - voltage
    - initial conditions 80, 138
    - iteration-to-iteration change 447
    - logic high 757, 761
    - logic low 758
    - logic low threshold 763
    - maximum change 358
    - minimum
      - DC analysis 359
      - listing 712

## Index

### W

- transient analysis 357
- operating point table 233
- tolerance
  - MBYPASS multiplier 537
  - value for BYPASS 403
- VOLTAGE keyword 233
- VREF command 765
- VREF statement 765
- VTH command 766

### W

- WACC option 714
- warnings
  - limiting repetitions 717
  - suppressing 571
- WARNLIMIT option 717
- WDELAYOPT option 719
- WEIGHT keyword 188, 196
- W-elements transmission line model 214
- WHEN keyword 180
- WHEN, using with .MEASURE 178
- .WIDTH command 779
- WINCLUDEGDIMAG option 722

- WL option 723
- WNFLAG option 724
- WSF output data 783, 795

### X

- XDTEMP option 725
- XGRID model parameter 773
- XMAX model parameter 773
- XMIN model parameter 773
- XSCAL model parameter 773

### Y

- YGRID model parameter 773
- YMAX parameter 196, 773
- YMIN parameter 196, 773
- YSCAL model parameter 774

### Z

- ZUKEN option 798