ZSIM

NONLINEAR Z DOMAIN SIMULATOR
TARGETD AT THE REDUCED ORDER MODELING OF DELTA SIMA
MODULATORS

MANUAL

Version ZSIM2.0

May 5, 2003

Michael B. Steer
Gregory T. Brauns
Sasan H. Ardalan

m.b.steer@ieee.org

North Carolina State University

Availability: http://www.freeda.org

1. INTRODUCTION

ZSIM is a nonlinear Z domain simulator targeted specifically at simulating delta sigma modulators. It has been used successfully to simulate the system performance of third order delta sigma circuits. A macromodel must first be generated by running spice multiple times on subcircuits of a switched capacitor delta sigma modulator. This macromodel captures the nonlinear Z domain response of the subcircuits. The original version of ZSIM (ZSIM0a1) was developed in 1987. This new release has many bug fixes and better help features. For the theory behind ZSIM see

1. R. J. Bishop, J. J. Paulos, M. B. Steer and S. H. Ardalan, "Table-based simulation of delta-sigma modulators," *IEEE Trans. on Circuits and Systems*, March 1990, pp. 447-451.
2. G. T. Brauns, R. J. Bishop, M. B. Steer, S. H. Ardalan and J. J. Paulos, "Table-based modeling of delta-sigma modulators using ZSIM," *IEEE Trans. Computer Aided Design*, Feb. 1990, pp. 142-150.

ZSIM2 is intended as a prototype for the implementation of a nonlinear Z domain analysis engine in fREEDA http://www.freeda.org .

2. VERSION NOTES

ZSIM is a nonlinear Z domain simulator tailored to the simulation of Delta Sigma Modulators (DSM) but should be applicable to most switched capacitor circuits. The key feature is the use of a nonlinear z domain macromodel which is developed using a circuit simulator. ZSIM has a user-friendly input format but lacks a totally stand alone topology specification. ZSIM is a prototype for a full nonlinear z-domain, event-driven implementation in fREEDA. Currently up to third order DSM can be simulated using difference equations and first and second order DSM can be simulated using table methods.

3. Standard Distribution

The standard distribution has several subdirectories

src:
Contains the source file and a makefile. To create the program ``zsim" simply type `make’ in this directory. Ignore the warning if you get one) in polate.f regarding using a variable before it has been defined. It is actually defined through a common statement.

example:
Contains an introductory example derived from zsimpap but simplified.

zsimpap
Contains the ZSIM simulations and plots for the paper: Table-based Modeling of Delta-Sigma Modulators using ZSIM.

deltapap:
Contains the ZSIM simulations and plots for the paper: Table-based Simulation of Delta-Sigma Modulators.

tripap:
Contains the ZSIM simulations and plots for the paper:
Improved Signal-To-Noise Ratio Using Tri-Level Delta-Sigma Modulators.

delta-sig:
Premilinary ZSIM runs and table analyses. This directory and its subdirectories are not documented. All of the final simulations (included in papers) are indentified in REAMDE files in the *pap directories.

4. KNOWN BUGS

1. When "decwin=para" to match "window=para" the program has an arithimetic overflow and bombs. With "decwin=ignore" it works fine.
2. Being FORTRAN there could be small compilation bugs to resolve on your systems.

5. INSTALLATION GUIDE

ZSIM is written in FORTAN 77 and operation has been verified on Sun OS 5.6. There are minor compilation incompatibilities with Linux.

** individual module files **

The files below should be compiled and linked together.

agen.f	aquant.f	ascint.f	calfbw.f	calsdr.f
casfil.f	compar.f	decimate.f	desim.f	dffil.f
envirn.f	fft.f	gauss.f	gendec.f	gentor.f
init.f	intgtr.f	nlfil.f	nodept.f	nodset.f
noiset.f	polate.f	ranf.f	sdrsub.f	tablrd.f
zana.f	zbdec.f	zcalsdr.f	zcmd.f	zdec.f
zdump.f	zequat.f	zhelp.f	zinput.f	zprint.f
zsdr.f	zset.f	zsim.f		

The program calls a routine fdate using call fdate(date) where date is dimensioned as
character*24 date

On UNIX systems this returns the date. On nonUNIX machines the above routine can be provided or the dummy file fdate.f (included in this distribution) can be included in the above list.

6. I/O CONVENTIONS

All input to ZSIM is converted to uppercase at input. On some systems, e.g. unix based, user's need to be aware that files are always accessed in uppercase by the program.

There is a high level input capability so that input can be taken from the keyboard and/or from a disk file - see user's guide for more information.

Comments can be included by preceding an input line by ' * '.

7. PROGRAM ORGANIZATION

The program is organized by sets of modules with a common set of high-level input routines contained in the file zinput.f.

The program is block oriented. A high level view of the program is as follows.

```
Level of program

top     next

ZSIM
        INPUT           ! Input parameters
        SIMULATION      ! Perform simulation of sampled data system
        DECIMATE        ! Perform decimation
        SDR             ! Perform SDR calculations
```

Description of a circuit must be in terms of nodes however in ZSIM arbitrary topology has not been completely implemented. ZSIM needs to be informed as to the interconnection of blocks in the sampled data system. At present only first and second delta sigma modulators can be modeled (indicated by the parameter CIRCUIT) using table modeling techniques and first, second and third order delta sigma modulators (again CIRCUIT) using difference equation techniques.

The distinction between a table based simulation and a difference equation simulation is indicated by the command to do the simulation (the command SIMULATE performs a table based simulation and DESIMULATE does a difference equation simulation).

8. USER'S GUIDE

This guide is organized by commands. ZSIM is largely self documenting and help on each command is available by typing

"COMMAND" HELP.

Commands Available:

COMMAND	KEYWORD	DESCRIPTION
SIMULATE	SIM	Perform z-domain simulation (tables)
DESIMULATE	DES	Perform z-domain simulation (diff.eq.)
GEN	GEN	Define generator input
SCINT	SCINT	Define integrator input
QUANT	QUANT	Define quantizer input
TABLE	TABLE	Define table description
CIRCUIT	CIRC	Define circuit type
ENVIRONMENT	ENV	Define simulation features
CLEAR	CLEAR	Erase all parameters to default
DUMP	DUMP	Dump contents of stored node values
STOP	STOP	End session
INIT	INIT	Initialize quantities
DECIMATE	DEC	Perform a decimation
SDR	SDR	Signal-to-Distortion calculation
EQUATION	EQ	Specify difference equation parameters

The following input/output commands are available:

read filename	: the file filename is opened for input
read	: the previously opened file is now used for input
write filename	: the file filename is opened for output
eof	: the input data file is closed
end	: the input data file is temporarily closed
title	: write line to output file
prompt	: write line to terminal
echo on	: the flag prt is set
echo off	: the flag prt is cleared
	If prt is set all lines input are echoed.

```
###################### CIRCUIT ##########################
        Usage:

          CIRCUIT HELP
            : This Message

          CIRCUIT = n
            : Set order of modulator to "n"



######################## CLEAR ##########################
        Usage:

          CLEAR HELP
            : This Message

          CLEAR
            : Set everything to default value
```

```
###################### DECIMATE ########################
     Usage:

       DEC HELP
         : This Message

       DECIMATE n1 n2 TYPE=ZBDEC
          This performs a decimation on the values of node n1
          and outputs the result (usually the baseband) at n2
          TYPE indicates the type of canned decimation.

       DECIMATE n1 n2 TYPE=GENDEC NUMBER=nn WINDOW=sss TAPS=nn
             INTDEC=nn BBANDDEC=bb FILE=filename FILTER=sss
          This performs a decimation on the values of node n1
          and outputs the result (usually the baseband) at n2
          NUMBER can be to base 2, e.g.:  NUMBER=nn BASE 2
          WINDOWs available: UNIFORM TRIANgle PARABolic
          TAPS  : length of the decimation filter
          INTDEC: FIR decimation factor
          BBANDDEC : baseband (low-pass) decimation factor
          FILE specifies file with filter coefficients
          FILTERs available: DIRECT_form NORMalized_lattice
            CASCade_form


        Included in this distribution of ZSIM is the file VB.CAS
        which specifies a cascade filter designed for decimation
        to voiceband.  VB.CAS has a 3.4 kHz rolloff for an 8kHz
        sampling frequency.  Coefficients for other filters must
        be generated by the user.

########### DIFFERENCE EQUATION SIMULATION ##############
     Usage:

       DES HELP
         : This Message

       DES
         : Simulate a circuit described by
         : difference equations
```

###################### DUMP ########################

Usage:

DUMP HELP
: This Message

DUMP file n no
: Dump first "no" values at node "n" to "file"

###################### EQUATION ########################

Usage:

EQ HELP
: This Message

EQ GAIN#=xxx1 DELTA#=xxx2 FGBW#=xxx3
: OFINT#=xxx4 SAT#=xxx5
: # = 1, 2, or 3 for respective integrator but may be
: left out to describe all integrators.
: This inputs ideal circuit parameters:
: GAIN# : gain of integrator
: DELTA# : reference voltage of quantizer
: FGBW# : finite gain bandwidth of integrator
: OFINT# : integrator offset referred to the output
: SAT# : integrator saturation

NOTE:: FGBW is currently not fully implemented.

Default Values are:

a1	1.00000	a2	1.00000	a3	1.00000
fgbw1	0.	fgbw2	0.	fgbw3	0.
delta1	1.00000	delta2	1.00000	delta3	1.00000
ofint1	0.	ofint2	0.	ofint3	0.
sat1	5.00000	sat2	5.00000	sat3	5.00000

```
###################### GENERATOR #########################
    Usage:

      GEN HELP
        : This Message

      GEN no node TYPE = DC AMP=x2
        : generator number "no" at node "node"
        : DC generator of amplitude "x1" and

      GEN no node TYPE = RAMP THR[ESHOLD]=x1 AMP=x2
        : generator number "no" at node "node"
        : ramp generator of amplitude "x2" and
        : threshold of "x1"

      GEN no node TYPE = SINE FREQ=freq OFFSET=x1 AMP=x2
        : generator number "no" at node "node"
        : sinewave generator of frequency "freq",
        : amplitude "x2", offset "x1", and zero phase

      GEN no node TYPE = SINE FREQ=freq AMP=x1 PHASE = x2 DEG
        : generator number "no" at node "node"
        : sinewave generator of frequency "freq",
        : amplitude "x1", and phase of "x2" degrees

      GEN no node TYPE = SINE FREQ=freq AMP=x1 DB WRT x3
          PHASE = x4 RAD
        : generator number "no" at node "node"
        : sinewave generator of frequency "freq",
        : amplitude "x1" With Respect To "x3" V and
        : phase of "x4" radians

      GEN no node TYPE=STEP INIT=x1 AMP=x2 DELAY=x3
        : step generator with initial value "x1", amplitude
        : "x2" and delay of "x3" seconds
##################### INITIALIZE ########################
    Usage:

      INIT HELP
        : This Message

      INIT NODE n x
          INitializes NODE n to x.  This is required as several
          blocks use previous node values in calculations
```

###################### NOISE #########################

Usage:

NOISE HELP
: This Message

NOISE n x1 x2
: Add Gaussian white noise to node "n" with mean "x1"
: and standard deviation "x2"

** NOT TESTED OR FULLY IMPLEMENTED **

###################### QUANTIZER ########################

Usage:

QUANT HELP
: This Message

QUANT n n1 n2 THRES=xxx1 TYPE=sss1 BAND=xxx2 SAME=nnn1
: Quantizer "n" at input node "n1" and output node "n2"
: has THRESHOLD of "xxx1" and TYPE = NONHysteresis or
: HYSTeresis.
: options 1.previously defined quantizer number "nnn1"
: uses the same data table
: 2."xxx2" is a deadzone region where the
: quantizer is undecisive (output=0).
: Integrator table must include x3=0 (see
: section 5 on generating tables.
:
: NOTE:: option 1 is presently disabled since the
: quantizer is ideal and not characterized by
: a table as of now.
: TYPE must be specified NONHysteresis
: until a hysteresis table is programmed

###################### SDR ########################

Usage:

```
SDR HELP
  : This Message

SDR n1 n2 TYPE=CALSDR NPOINTS=nnn1 NFFT=nnn2 NSKIP=nnn3
  : FFTWINDOW=sss1  DECWINDOW=sss2  FB=xxx1 NFAST=nnn4
  : This performs a Signal-to-Distortion calculation on
  : the values of node n1.  The calculated spectrum is
  : stored as node n2. Options available:
     : NPOINTS = total number of input bins
     : NFFT = number of bins to use for fft
     : NSKIP = number of initial bins to ignore
     :        (NPOINTS = NFFT + NSKIP)
  : FFTWINDOW = UNIForm HAMMing blackman-HARRis
  : DECWINDOW = IGNORE UNIForm TRIAngular PARAbolic
  : (DECWINDOW should be the same used in decimation.)
     :   The following parameters can be ignored if
     :   decimation not used (DECWINDOW=IGNORE)
  : FB : baseband sampling frequency
  : NFAST : number of taps in decimation filter
```

Warning: The parameters in SDR must agree with those in the decimate command.

We suggest that FFTWINDOW = UNIFORM be used as only this has be debugged and tested. This amounts to not using a window and is satisfactory provided the input signal coincides with a bin.

The SDR calculation determines the signal level SIG, the total distortion level DIST, the noise level NOIS, and the harmonic level HARM. DC is excluded from the calculation.

DIST = NOIS + HARM

Since very few bins (of the frequency spectrum) are typically available, the noise in the location of the harmonics is estimated by averaging the noise on either side of the harmonics and subtracting it from the total signal level at the harmonic bins.
The FFTWINDOW used determines the signal spread and the number of bins over which the signal levels should be determined.

################## TABLE SIMULATION #####################

Usage:

SIM HELP
: This Message

SIM
: Simulate a circuit described by tables

###################### INTEGRATOR #######################

Usage:

SCINT HELP
: This Message

SCINT n TYPE=SWITCH n1 n2 n3 SAME=nnn1
: switched capacitor integrator number "n"
: with positive input node "n1"
: and negative input node "n2"
: and output node "n3".
: option-> previously defined integrator number "nnn1"
:   uses the same data table

SCINT n TYPE=ANALOG DIM=nnn1 n1 n2 SAME=nnn2
: analog integrator number "n" with table DIMENSION
: "nnn1" (excluding input) and input node "n1"
: and output node "n2".
: option-> previously defined integrator number "nnn2"
:   uses the same data table

WARNING:: Since arbitrary circuit topology has not been
completely implemented, the ANALOG integrator
should not be used.

######################## STOP ###########################

Usage:

STOP HELP
: This Message

STOP
: End ZSIM session

######################## TABLE ############################

Usage:

```
TABLE HELP
  : This Message

TABLE SCINT n file
  : "file" is the input file which contains the TABLE
  : that describes the switched capacitor integrator
  : number "n".

TABLE QUANT n file
  : "file" is the input file which contains the TABLE
  : that describes the quantizer number "n".

NOTE: quantizer table not yet operational!
```

9. GENERATING YOUR OWN TABLE

Currently tables can only be used to describe the characteristics of switched capacitor integrators.

Integrator

```
                              __________
                             |          |
analog input (x1) -->-----|          |
   <node n1>                 |          |--->--  output of integrator (x2)
                             |          |        <node n3>
fedback bit (x3) --->-----|          |
   <node n2>                 |__________|
```

x1 and x2 can be any real value
x3 must be an integer

This is described by the difference equation

$$x2(n) = x2(n-1) + G*(x1(n) - x3(n-1))$$

where n and n-1 refer to cycle numbers and G is gain.

```
format of table:                    comments:

* comment lines
x1= values of x1 in monotonic order     ! this is the input to
                                        ! the integrator at the current
                                        ! cycle.  let there be I x1
                                        ! values
x2= values of x2 in monotonic order     ! this is the value of the
                                        ! integrator output at the
                                        ! previous cycle.
                                        ! Let there be J x2 values.
x3= values of x3 in monotonic order     ! this is the value of the
                                        ! binary fed back bit for the
                                        ! current cycle.
                                        ! Let there be K x3 values.

begin                                   ! instruction starting readin of
                                        ! table

                                        ! The tables are arranged so
```

```
                                              ! that x3 varys slowest, then x2
                                              ! and x1 varys the fastest
                                              ! each line of the table
                                              ! corresponds to constant x2 and
                                              ! and x3 and the numbers on a
                                              ! line are for the output x2  at
                                              ! the current cycle for each
                                              ! value of x1.

x2[x1(1)] ... x2[x1(i)]  ... x2[x1(I)]    ! line for x2(1) x3(1)
.             .          .
x2[x1(1)] ... x2[x1(i)]  ... x2[x1(I)]    ! line for x2(j) x3(1)
.             .          .
x2[x1(1)] ... x2[x1(i)]  ... x2[x1(I)]    ! line for x2(J) x3(1)
.             .          .
.             .          .
x2[x1(1)] ... x2[x1(i)]  ... x2[x1(I)]    ! line for x2(1) x3(k)
.             .          .
x2[x1(1)] ... x2[x1(i)]  ... x2[x1(I)]    ! line for x2(j) x3(k)
.             .          .
x2[x1(1)] ... x2[x1(i)]  ... x2[x1(I)]    ! line for x2(J) x3(k)
.             .          .
.             .          .
x2[x1(1)] ... x2[x1(i)]  ... x2[x1(I)]    ! line for x2(1) x3(K)
.             .          .
x2[x1(1)] ... x2[x1(i)]  ... x2[x1(I)]    ! line for x2(j) x3(K)
.             .          .
x2[x1(1)] ... x2[x1(i)]  ... x2[x1(I)]    ! line for x2(J) x3(K)

done                              ! indicates that table
                                  ! is finished
```