USERS' MANUAL FOR SWITCAP2 VERSION 1.1[1]

K. Suyama S. C. Fang[2]
Department of Electrical Engineering
and Center for Telecommunications Research
Columbia University
New York, NY 10027

February 1 1992

[1]Copyright ©1990, 1992 by Columbia University. All rights reserved.

[2]S. C. Fang is the author of the original SWITCAP manual which was written in 1982 at Department of Electrical Engineering, Columbia University.

Contents

List of Figures

ACKNOWLEDGEMENT

Professor Yannis Tsividis has contributed to the original concept of SWITCAP and SWITCAP2. The authors would like to thank Ilya Yusim for his help with the programming of the input parser for SWITCAP2 and with the integration of figures and text in this manual.

Chapter 1

INTRODUCTION

SWITCAP2 is a general simulation program for analyzing linear switched-capacitor networks (SCN's) and mixed switched-capacitor/digital (SC/D) networks. The major features of the program are as follows:

1. SWITCHING INTERVALS

   An arbitrary number of switching intervals per switching period is allowed. The durations of the switching intervals may be unequal and arbitrary.

2. NETWORK ELEMENTS

   The program can simulate linear SCN's containing the following types of ideal analog components: ON-OFF switches, linear capacitors, linear voltage-controlled voltage sources (VCVS's), and independent voltage sources. The waveforms of the independent sources may be continuous or piecewise-constant. The switches in linear SCN's are controlled by periodic clock waveforms only.

   In addition to the above elements mentioned, a mixed SC/D network may contain comparators and logic gates such as AND, OR, NOT, NAND, NOR, XOR, and XNOR. The ON-OFF switches in a mixed SC/D network may be controlled not only by periodic waveforms but also by nonperiodic waveforms from the output of comparators and logic gates.

3. TIME-DOMAIN ANALYSES OF LINEAR SCN'S AND MIXED SC/D NETWORKS

   The program provides two kinds of time-domain analyses. One is only applicable for linear SCN's and it computes the transient response of linear SCN's to any prescribed input waveform for $t \geq 0$ after computing the steady-state values for a set of dc inputs for $t < 0$.

The other time-domain analysis applies to mixed SC/D networks as well as linear SCN's. It computes the transient response of networks *without* computing the steady-state values as initial conditions. A set of the initial condition of analog and digital nodes at $t = 0^-$ may be specified by user.

4. VARIOUS WAVEFORMS FOR TIME DOMAIN ANALYSES

   Various time-domain waveforms such as pulse, pulse train, cosine, exponential, exponential cosine, piecewise linear, and dc sources are available as the input for transient analysis.

5. FREQUENCY DOMAIN ANALYSES OF LINEAR SCN'S

   Since SCN's are time-varying networks, a single-frequency sinusoidal input can produce a steady-state output containing many frequency components. The program can determine all of these output frequency components for both continuous and piecewise-constant input waveforms. Z-domain quantities can also be computed.

   Frequency-domain group delay and sensitivity analyses are also provided.

6. BUILT-IN SAMPLING FUNCTIONS

   Both the input and output waveforms may be sampled and held at arbitrary instants to produce the desired waveforms for time- and frequency-domain analyses of linear SCN's except for sensitivity analysis. The output waveforms may also be sampled with a train of impulse functions for z-domain analyses.

7. SUBCIRCUIT FACILITY

   Subcircuits, including analog and/or digital elements, may be defined with symbolic values for capacitances, VCVS gains, clocks, and other parameters. The symbolic values may then be replaced with actual values during each reference to a subcircuit. This facility provides a convenient means for building a library of often-used building-block circuits from which more complicated circuits may be pieced together. It also facilitates a hierarchical design approach.

8. FINITE RESISTANCES, OPERATIONAL AMPLIFIER POLES, AND SWITCH PARASITICS

   Finite resistances (say associated with nonideal switches or anti-aliasing filters) may be simulated by equivalent SC networks. To obtain good accuracy, the switches associated with the finite-resistor equivalent SC networks may be switched at a frequency much higher than both the input signal frequency and the switching frequency of the actual switches of the circuit. The time-constants introduced by the resistors may be longer than the length of an actual switching interval (i.e., the network need not be equilibrated).

Operational amplifiers with limited frequency response may also be simulated. The frequency response may be modeled with an RC network and then simulated with the R's replaced by equivalent SC networks.

Clock feedthrough from switch parasitics may be simulated in both the time and the frequency domains. The frequency-domain simulation of this effect is particularly efficient using the dc steady-state analysis capability of the program which does not require the direct computation of the Fourier components of any clock waveform.

To incorporate these higher order effects in a simulation, the subcircuit facility of the program may be used to advantage. A circuit can be simulated with, say, ideal elements at a high level of abstraction, or complex subcircuits including parasitics and nonidealities can be substituted for ideal elements to carry the simulation to a lower level. However, it must be warned that the simulation of these nonideal effects is computationally time consuming due to the need for higher switching rate to model the nonideal effects.

9. INPUT FILE COMPATIBILITY WITH SWITCAP

The input parser in SWITCAP is completely rewritten to allow the flexibility necessary to add new features. However, the new parser in SWITCAP2 is designed such that any old SWITCAP input files can be used in SWITCAP2 without any modifications. Should any incompatibilities come up, please contact SWITCAP Distribution Center immediately.

Internally, SWITCAP2 consists of several different algorithms depending on the type of desired analysis. For time- and frequency-domain analyses (excluding group delay and sensitivity analyses) of linear SCN's, SWITCAP2 uses an algorithm based on the formulation published in [1, 2, 3, 4, 5, 6, 7, 8, 9]. This formulation, which was incorporated in the original SWITCAP [6, 7], uses a set of charge variables as the network variables for computation. The number of charge variables is typically very small. For example, an *nth* order filter implemented in a leap-frog configuration wherein the number of integrators is the same as the order would have n charge variables per switching interval. As a result of the small number of network variables, the iterative steps in the computation are very efficient. The most memory-intensive portion of the computation is in the preprocessing stage during which the network matrices to be used in the subsequent iterations are computed and stored. Because of this memory requirement, a dynamic memory allocation scheme has been implemented. With this allocation scheme, the maximum complexity (number of elements times the number of unique switching intervals) of an SCN that can be simulated is limited solely by the user's computer memory size.

The analytical expressions that allow exact analyses of group delay and sensitivity for many-phase linear SCN's are derived from the above mentioned state charge variable formulation [10, 11]. The analytical expressions not only lead to efficient algorithms for computing the group delay and sensitivity but also retain efficiencies similar to those that were realized for simple time- and frequency-domain analyses using the state charge formulation.

The transient analysis of mixed SC/D networks is formulated in such a way that separate algorithms are used for analog, digital, and interface subgroups to take full advantage of their own properties. Detailed technical discussions of the algorithms and their applications can be found in [11, 12, 13].

This document is the user's guide for SWITCAP2. Chapter 2 of this guide gives an overview of the features of SWITCAP2 in a tutorial style. After reading this chapter, the user will be able to simulate most SCN's he or she is likely to encounter in practice. More advanced features or more subtle points of SWITCAP2 are described in the later chapters. The last chapter contains the SWITCAP2 Language Reference which is intended as a reference for users who are already familiar with SWITCAP2. The first-time user is encouraged to read through Chapter 2 first.

Columbia University welcomes all comments about the program and this manual. Comments can be addressed to:

SWITCAP Distribution Center
Columbia University
411 Low Memorial Library
New York, N.Y. 10027
U.S.A

Bibliography

[1] Y. P. Tsividis, “Analysis of switched capacitive networks,” *IEEE Trans. Circuits Syst.*, vol. CAS-26, no. 11, pp. 935-947, Nov. 1979.

[2] S. C. Fang and Y. P. Tsividis, “Modified Nodal Analysis with Improved Numerical Methods for Switched Capacitive Networks,” in *Proc. IEEE Int. Symp. Circuits Syst.* (Houston, TX), pp. 977-980, Apr. 1980.

[3] S. C. Fang, Y. P. Tsividis, and O. Wing, “Analysis of SCN’s with Nonlinear and Time-Varying Elements Using Time-Invariant Charge Variables,” in *Proc. IEEE Int. Symp. Circuits Syst.* (Rome, Italy), pp. 1-4, May 1982.

[4] S. C. Fang, Y. P. Tsividis, and O. Wing, “Efficient algorithms for analyzing many-phase switched-capacitor networks,” in *Proc. IEEE Int. Symp. on Circuits and Systems*, (Newport Beach, CA), pp. 60-63, May 1983.

[5] S. C. Fang, *Analysis, Computer Simulation, and Properties of Switched-Capacitor Networks*, Ph.D. dissertation, Columbia University, New York, 1983.

[6] S. C. Fang, Y. P. Tsividis, and O. Wing, “SWITCAP: A switched-capacitor network analysis program. Part I: basic features,” *IEEE Circuits Systems Magazine*, vol. 5, no. 3, pp. 4-10, Sept. 1983.

[7] S. C. Fang, Y. P. Tsividis, and O. Wing, “SWITCAP: A switched-capacitor network analysis program. Part II: advanced features,” *IEEE Circuits Systems Magazine*, vol. 5, no. 4, pp. 41-46, Dec. 1983.

[8] S. C. Fang, Y. P. Tsividis, and O. Wing, “State charge formulation of switched-capacitor networks containing nonlinear and time-varying elements,” *IEEE Trans. Circuits Syst.,* vol. CAS-31, no. 11, pp. 968-974, Nov. 1984.

[9] S. C. Fang, Y. P. Tsividis, and O. Wing, “Time- and frequency-domain analysis of linear switched-capacitor networks using state charge variables,” *IEEE Trans. on Computer-Aided Design*, vol. CAD-4, no. 4, pp. 651-661, Oct. 1985.

[10] K. Suyama and S. C. Fang, "Efficient state charge variable analyses of group delay and sensitivity for switched capacitor networks," *Proc. 29th Midwest Symp. on CAS*, Lincoln, NE, pp. 789-792, Aug. 1986.

[11] K. Suyama, *Analysis, Simulation, and Application of Linear and Nonlinear Switched-Capacitor and Mixed Switched-Capacitor/Digital Networks*, Ph.D dissertation, Columbia University, New York, 1989.

[12] K. Suyama, S. C. Fang, and Y. P. Tsividis, "Simulation of mixed switched-capacitor/digital networks with signal-driven switches," *IEEE J. Solid-State Circuits*, vol. SC-25, no. 6, pp. 1403-1413, Dec. 1990.

[13] K. Suyama, S. C. Fang, and Y. P. Tsividis, "Simulation methods for mixed switched-capacitor/digital networks," To be published.

Chapter 2

FUNDAMENTALS

This chapter covers the basics of SWITCAP2. A user should be able to get started with running the program by reading just this chapter. It is recommended that this chapter be read first since the later chapters in this user's guide will assume a familiarity with the contents here. It is also recommended that this chapter be read sequentially for a similar reason.

2.1 Relation of SWITCAP2 to Laboratory Setup

The input format for SWITCAP2 has been designed to correspond as much as possible to actual laboratory setups which one might use for making measurements on SCN's and mixed SC/D networks. Figure 2.1 shows schematically such a possible setup. In this setup, the signal generators may be, for example, a sinewave oscillator or a square wave generator. The measurement instruments may be a spectrum analyzer or an oscilloscope. The clocks are control signals for turning on and off a set of switches inside the network. To describe this setup to SWITCAP2, the input file is divided into three sections: First, there is a TIMING SECTION in which all of the clock and timing data are specified. Next, there is a NETWORK SECTION in which the network is specified (topology and element values). Finally, there is an ANALYSIS SECTION in which the input signals and output measurements are specified.

In the following, two complete examples will be worked through from circuit diagram and timing waveform to the final SWITCAP2 input description and output printout. The first example is a simple SC integrator and it is used to describe various analysis facilities for linear SCN's. The second example explains simulation capabilities for mixed SC/D networks using a second-order sigma-delta modulator.

The overall architecture of the program is shown in Fig. 2.2. The input parser translates

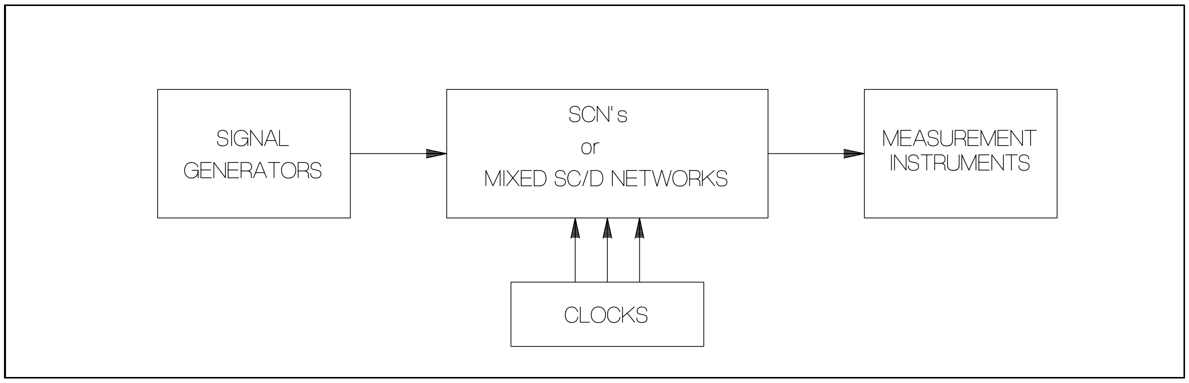

Figure 2.1: General laboratory setup.

the user input file into several intermediate disk files. The files are then used to construct necessary matrices and event tables for each analysis.

2.2 Linear SCN Example: A SC Integrator

Consider the SCN and its switch timing diagram shown in Fig. 2.3. This circuit contains three capacitors, C1, C2 and C3, two switches, S1 and S2, and an operational amplifier E1. Switch S1 is controlled by the the clock CLK. The switch is closed when the clock is HIGH and is open when the clock is LOW. Switch S2 is controlled by the complement value of CLK (this is indicated by the "#" sign before the clock name), so it is open when CLK is HIGH and closed when CLK is LOW. The clock CLK has a period of 8 us. The node numbers are enclosed in circles.

2.2.1 Timing Section

The first thing we must describe to SWITCAP2 is the switch timing information. We shall call each instant at which a clock makes a transition a "clock time". Each clock time corresponds to a switching instant. In Fig. 2.3 the clock times are at 0 us, 3 us, 8 us, and so on. SWITCAP2 uses a scheme of specifying all clock time values **normalized to the period of a fictitious master clock**. In addition, the master clock period chosen must be such that the resulting normalized time can be expressed as a **rational number**. We define the duration of one Master Clock Period as one MCP unit of time. Looking at the timing diagram in Fig. 2.3, we see that if we had a master clock period of 8 us, then all clock times could be expressed in rational numbers of units of mcp. For example, 3 us would be equal to 3/8 mcp, 8 us would be 1mcp, and so forth. This timing can be specified to

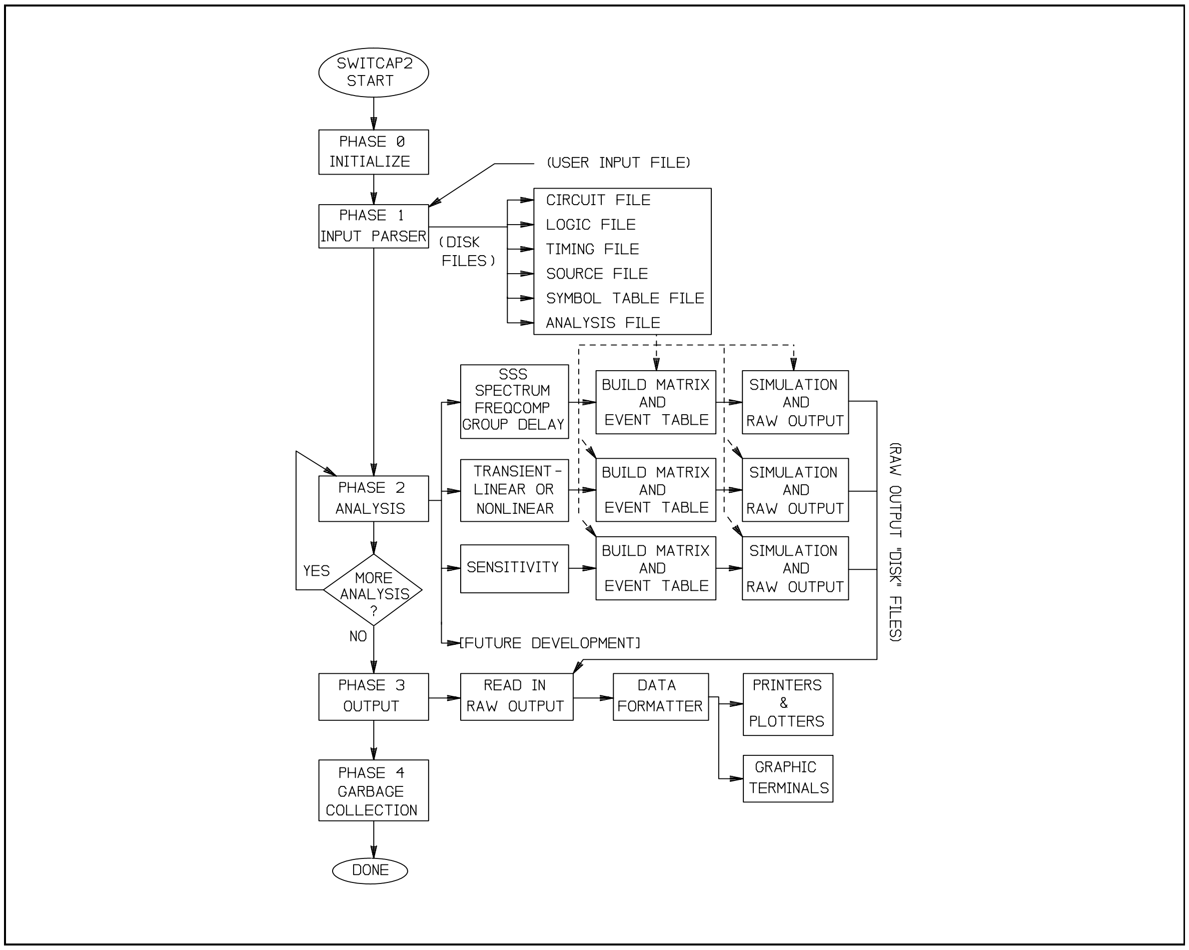

Figure 2.2: SWITCAP2 overall structure.

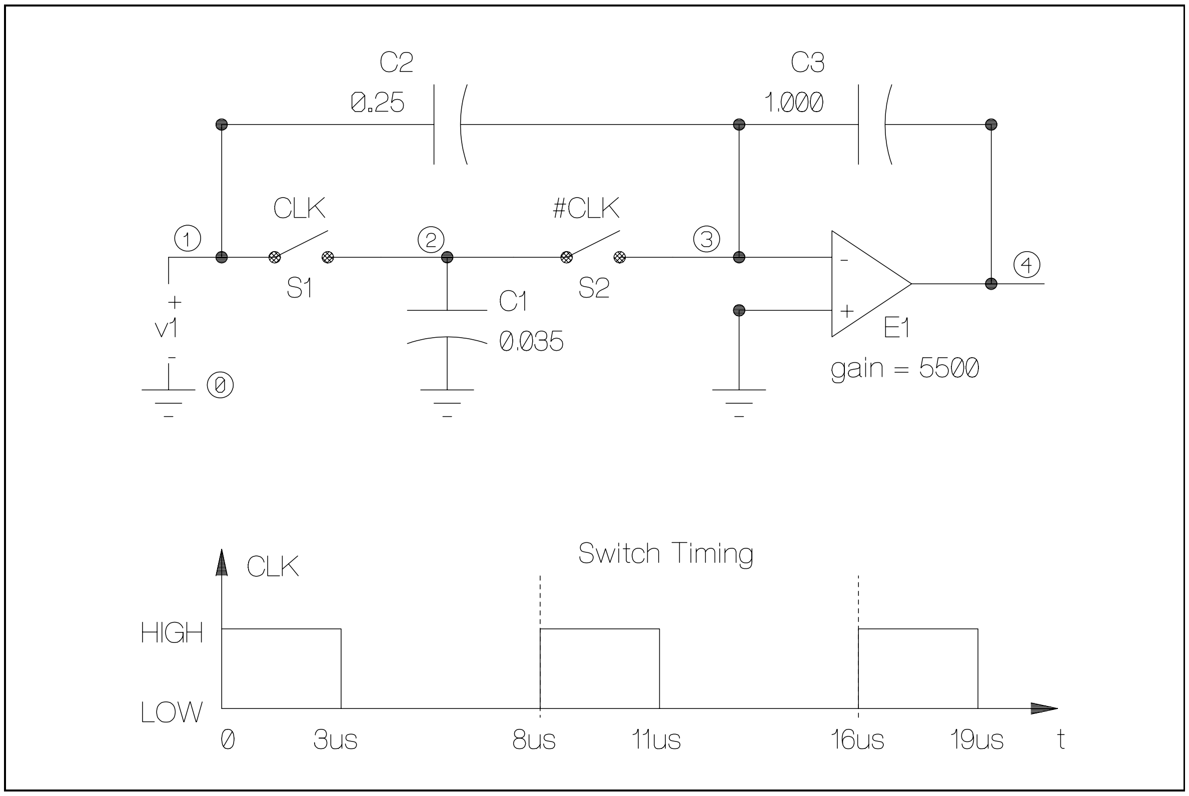

Figure 2.3: A simple switched-capacitor network.

SWITCAP2 by the statements:

```
1              TIMING;
2                   PERIOD 8e-6;
3                   CLOCK clk 1 (0 3/8);
4                   END;
```

There are four SWITCAP2 statements in (1-4). The indentation of each statement is arbitrary. The statement labels "1", "2", etc. are for reference only and are not part of the SWITCAP2 statements. The keywords "TIMING" and "END" define a TIMING block. (All keywords recognizable to SWITCAP2 will in this manual be written in CAPITAL letters, though this is not necessary for SWITCAP2 input file.) The middle two statements have the following meaning:

PERIOD 8e-6;

This statement defines the master clock period as being 8 us. It establishes the correspondence between 1mcp and an absolute unit of time.

CLOCK clk 1 (0 3/8);

This statement defines a switch-controlling clock. "clk" is the name of the clock. This name will later be referenced in the switch specification statements in the network section to indicate which switches are being controlled by which clocks. The "1" means that the period of this clock is 1mcp. The starting and ending times of the HIGH interval of the clock are at 0mcp and 3/8mcp, respectively. (Note that rational numbers are written as a ratio of two integers and not as a decimal.) Clock names must be a string of 78 or fewer alphanumeric characters, the first of which must be a letter (a-z).

*** IMPORTANT NOTE ***

It is important to keep in mind that all time values expressed here are in units of the master clock period defined in the PERIOD statement and *not* in units of the period of the individual switch-controlling clocks. For example, if another clock had a period of 16 us with HIGH intervals (0 us, 3 us), (16 us, 19 us), and so forth, it would be defined by the statement: CLOCK clk2 2 (0 3/8);
In each input file the period defined by the PERIOD statement is the only time base by which all clock times are reckoned.

In general, an SCN may have several switching-controlling clocks of different periods. Each clock may also have more than one HIGH intervals. (See Section 3.1 for more details.)

2.2.2 Network Section

To describe the network topology and the element values in the circuit in Fig. 2.3, we can use the statements:

```
5           CIRCUIT;
6                   S1 (1 2) clk;
7                   S2 (2 3) #clk;
8                   C1 (2 0) 0.035;
9                   C2 (1 3) 0.250;
10                  C3 (3 4) 1.000;
11                  E1 (4 0 0 3) 5500;
12                  V1 (1 0);
13                  END;
```

The keywords "CIRCUIT" and "END" define a CIRCUIT block. Statements (6-12) are the actual statements that describe the network elements. These statements follow the general format:

$$< r >< elementname > (< nodelist >) < elementvalue >;$$

where

$< r >$ is a key character indicating the type of element being defined. The key character designations are:

- **S** for switches
- **C** for capacitors
- **E** for voltage-controlled voltage sources (VCVS)
- **V** for independent voltage sources

$< elementname >$ is a string of alphanumeric characters less than 78 characters long which serves as a label for the element. Element names among elements of the same type should be unique.

< *nodelist* > is a list of the nodes to which the element is connected. This list *must be enclosed in parentheses*. Node names may be any string of 78 or fewer alphanumeric characters. The ground node is always 0 (the digit zero).

< *elementvalue* > gives the "value" of the element that is appropriate for the element.

The more specific features of statements (6-12) are described below. The portions of a statement being highlighted by the discussion are underlined.

S1 (1 2) clk;

S2 (2 3) #clk;

These are the clocks controlling the switches. The negate sign (#) indicates that the switch S2 is controlled by the complementary value of the clock "clk". A clock (without the # sign) must match one of the clock names specified in the TIMING block. The sequence of switch actions at each clock time is such that all switches to be opened are opened before any switch to be closed is closed, regardless of whether the switch is controlled by a clock or a complemented clock.

C1 (2 0) 0.035;

C2 (1 3) 0.250;

C3 (3 4) 1.000;

These are capacitance values. Since in SCN's the node voltages can depend only on the ratios of the capacitances, these capacitance values may be in arbitrary units.

E1 (4 0 0 3) 5500;

The branch relation of a VCVS can be written as

$$v(n+) - v(n-) = k \times [v(nc+) - v(nc-)]$$

where $(n + n-)$ are the nodes of the controlled branch, $(nc + nc-)$ are the nodes of the controlling branch, and k is the gain constant. In statement 11 the nodes (4 0 0 3) are in the order of $(n+ \ \ n- \ \ nc+ \ \ nc-)$ and 5500 is the gain k.

V1 (1 0);

Note that no specific value is defined for the independent voltage source. In this network section we only define the nodes to which the source is connected. Since the actual value of the source will depend on the type of simulation we want to perform, we defer the source value specification to the analysis section. The name V1 will be referenced later in the analysis section when the value associated with this source is defined.

An unlimited number of each type of element (including independent voltage sources) may be specified in a CIRCUIT block. The network topology and the action of the switches must be such that within each switching interval:

1. the network remains connected (an open switch does not count as a connection); and
2. there are no loops of only independent voltage sources, controlled branches of VCVS's, and closed switches.

2.2.3 Sinusoidal Steady-State Analysis

Having now specified the system timing and the network, we are ready to specify the analysis to be performed. For the moment let us suppose that we are interested in the sinusoidal steady-state (S.S.S.) response of this circuit from node (1) to node (4). By "the S.S.S. response" of this circuit at (any) frequency f_0 we mean the following (we will define it in such a way as to correspond to an actual laboratory measurement using a spectrum analyzer): We will attach a sinusoidal oscillator to the circuit between node (1) and ground and set it to the frequency f_0. Let the input waveform be of the form

$$v_1(t) = A\cos(2\pi f_0 t + \phi), \text{ for all } t \tag{2.1}$$

where A is magnitude and ϕ is the phase angle of the cosine waveform. The output waveform $v_4(t)$ on node (4) will contain a frequency component at frequency f_0 and, in general, since the circuit is a time-varying network, will also contain frequency components at frequencies other than f_0. Now, to correspond to the operation of a spectrum analyzer, we will attach a very narrow bandpass filter (BPF) to the output at node (4), as shown in Fig. 2.4, and tune its center frequency to that of the input. The BPF is assumed to have unity gain and zero phase shift at its center frequency. Now the only frequency component present at the output of the BPF is the one whose frequency is the same as the input frequency. In other words, the SCN and the BPF together, viewed as a whole, now have a sinusoidal input and a steady-state sinusoidal output (with the same frequency as the input). *We define the waveform at the output of the BPF to be the* S.S.S. *response of the SCN*. We emphasize that the S.S.S. response is defined only with respect to the SCN/BPF system as a whole wherein the BPF has been tuned to the input frequency, and not with respect to the SCN by itself. We also emphasize that this definition is good for *all* frequencies and not just for those less than the Nyquist frequency. If the output of the BPF $v_4'(t)$ is expressed in the form

$$v_4'(t) = B\cos(2\pi f_0 t + \theta), \text{ for all } t \tag{2.2}$$

where B is magnitude and θ is the phase angle of the output waveform, then when A, f_0, and ϕ are specified in S.S.S. analysis, the quantities B and θ will be computed. Let us

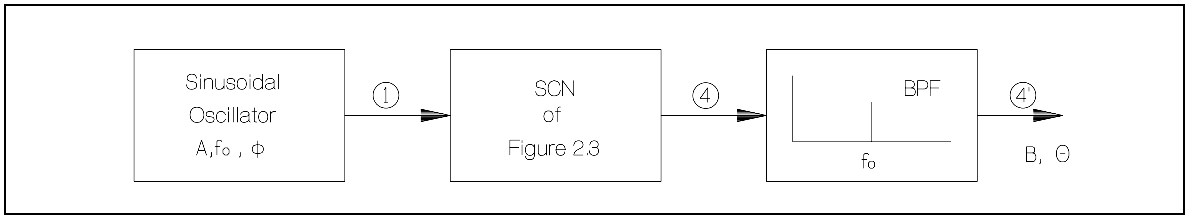

Figure 2.4: Measuring Sinusoidal Steady-state Response.

say that we are interested in the s.s.s. response of the circuit from 0.01 Hz to 100 KHz at 15 frequency points (including the two end points) equally spaced on a logarithmic scale between these two frequency limits; and that we want a printout of the real and imaginary parts and a plot of the magnitude (in decibels) and phase of this response. We can specify these conditions to SWITCAP2 with the statements:

```
14          ANALYZE SSS;
15              INFREQ 0.01 100000 LOG 15;
16              SET V1 AC 1.0 0.0;
17              PRINT VR(4) VI(4);
18              PLOT VDB(4) VP(4);
19              END;
```

The keywords "ANALYZE" and "END" define an ANALYZE block. The keyword "SSS" indicates that we want s.s.s. analysis. The other statements specify the input and output conditions for the simulation:

INFREQ 0.01 100000 LOG 15;
This statement defines the input frequency values (the f_0 in Eqn. 2.1). The keyword LOG causes the input frequency to be swept logarithmically. The other choice is to sweep linearly, which is specified by using LIN instead of LOG.

SET V1 AC 1.0 0.0;
This statement assigns a sinusoidal waveform to the independent voltage source V1. The independent source has been previously defined in the network section. The amplitude and phase (in degrees) of the input cosine waveform are 1.0 and 0.0 respectively. Thus the input waveform is given by

$$v_1(t) = 1.0\cos(2\pi(f_0 t + 0.0/360))$$

If more than one independent sources are defined in the CIRCUIT block, then each source must be assigned with a separate SET statement.

PRINT VR(4) VI(4);

PLOT VDB(4) VP(4);

These statements specify the output variables to be printed and plotted. The variable VR(4) means the real part of the voltage on node (4), VI(4) means the imaginary part, and so forth. The key characters indicating the various parts of the output quantities are:

R real part

I imaginary part

M magnitude

P phase in degrees

DB magnitude in decibels

GD group delay (see page 31 for more details)

(omitted) real part by default

Each node listed in a PRINT or PLOT statement will be called an "output node". In general, output nodes in PRINT and PLOT statement may be different. (See Chapter 4 for other output options.)

To complete the input consisting of the statements (1-4), (5-13), and (14-19), we need to add only one more END statement as the last statement to indicate the end of the input file. Thus the following set of statements constitute a complete and acceptable input file to SWITCAP2.

```
TIMING;
PERIOD 8e-6;
CLOCK clk 1 (0 3/8);
END;

CIRCUIT;
S1 (1 2)  clk;
S2 (2 3) \#clk;
C1 (2 0)  0.035;
C2 (1 3)  0.250;
C3 (3 4)  1.000;
```

```
E1 (4 0 0 3) 5500;
V1 (1 0);
END;

ANALYZE SSS;
INFREQ 0.01 100000 LOG 15;
SET V1 AC 1.0 0.0;
PRINT VR(4) VI(4);
PLOT VDB(4) VP(4);
END;
END;
```

This input file gives the output:

```
           frequency           vr(4)       vi(4)

        1.00000000D-02      -5.4659D+03  4.3182D+02
        3.16227766D-02      -5.1769D+03  1.2933D+03
        1.00000000D-01      -3.3864D+03  2.6753D+03
        3.16227766D-01      -7.5967D+02  1.8973D+03
        1.00000000D+00      -8.6968D+01  6.8513D+02
        3.16227766D+00      -9.0466D+00  2.1978D+02
        1.00000000D+01      -1.1309D+00  6.9600D+01
        3.16227766D+01      -3.3805D-01  2.2013D+01
        1.00000000D+02      -2.5875D-01  6.9611D+00
        3.16227766D+02      -2.5082D-01  2.2013D+00
        1.00000000D+03      -2.5003D-01  6.9611D-01
        3.16227766D+03      -2.4995D-01  2.2013D-01
        1.00000000D+04      -2.4994D-01  6.9611D-02
        3.16227766D+04      -2.4994D-01  2.2013D-02
        1.00000000D+05      -2.4994D-01  6.9611D-03


                                     (X): vp(4)
              90.0    105.0     120.0     135.0     150.0     165.0   180.0
              |---------+---------+---------+---------+---------+---------|

                                     (*): vdb(4)
        freq  -20.00    0.00        20.00       40.00       60.00     80.00
```

```
1.00-02 |----------+----------+----------+----------+-------=--|
3.16-02 |                                              X     * |
1.00-01 |                              X                    *  |
3.16-01 |             X                                 *      |
1.00+00 |    X                                     *           |
3.16+00 | X                                  *                 |
1.00+01 |X                              *                      |
3.16+01 |X                        *                            |
1.00+02 |X                   *                                 |
3.16+02 |   X          *                                       |
1.00+03 |         * X                                          |
3.16+03 |     *                      X                         |
1.00+04 |    *                                        X        |
3.16+04 |    *                                              X  |
1.00+05 |----*-----+----------+----------+----------+---------X|
  freq -20.00    0.00       20.00      40.00      60.00    80.00
                              (*): vdb(4)
```

A letter "d" may sometimes appear next to a frequency, with the response at the frequency usually differing drastically from those at nearby frequencies. This indicates a frequency-overlap condition due to aliasing and can be seen whenever the input frequency is an integral multiple of half the sampling frequency. (See Section 3.2 for details.) Plot axes may sometimes be labeled with numbers prefixed by dollar ($) signs, e.g., $6.000. These are only the least significant digits of a number and are intended to show as much precision as possible in a limited space. (See Section 4.3 for details.)

2.2.4 Input Format

Before we go further in describing the other types of analysis in SWITCAP2, let us state some general rules and features of the input format.

1. The three sections–TIMING, CIRCUIT, and ANALYZE–*must be written in the order given.* Statements within each block may, however, be written in any order.

2. The input may be free-formatted with an arbitrary number of spaces between words. More than one statements may be placed on the same line separated by semicolons (;).

3. A TITLE statement may be optionally added to the input file. A TITLE statement has the form:

TITLE:<anything>;

Everything between the colon and semicolon will be printed as a title in the output headings of the printout. The TITLE statement *can be at most one line long.* The statement can be anywhere[1] in the input file. Example:

TITLE:Switched-Capacitor Integrator (Fig. 2.2);

4. Comments may be inserted wherever a space is allowed. A comment is defined to be a character string that starts with the sequence /*, followed by an arbitrary sequence of characters, and ends with the sequence */. Comment strings are allowed to cross line boundaries[2]. Comments do not affect the input processing and are treated simply as a string of blanks. Examples:

```
/* This is a valid comment */       (valid)

/* This comment
   crosses a line boundary */       (also valid)
```

5. In the original SWITCAP, it was recommended that each statement be terminated with a semicolon (;). In SWITCAP2, the semicolon is not required for proper operation any more.

6. The input can be either in capital letters or in lower case letters.

7. All numerical values must be written in full as integers, rational numbers, or real numbers, whichever appropriate. Real numbers may be written in floating point notation, e.g. 1.2e3. Any scale factors such as MEG, P, etc. are *ignored.*

2.2.5 Piecewise-Constant Waveforms, Sampling, and Discrete-Time Frequency Response

So far we have considered sinusoidal steady-state analysis with a continuous sinusoidal input waveform for the circuit in Fig. 2.3. SWITCAP2 also has a set of built-in sampling functions for creating piecewise-constant waveforms and impulse-train waveforms for analysis. Using these functions, specifying input or output sampling amounts to adding just one statement in an ANALYZE block; no sampling circuitry need be added to the SCN. Moreover, sampling functions such as impulses, difficult to implement with real circuits but conceptually useful, may be simulated with ease.

[1]The TITLE statement was required to be the first statement in the input file in the original SWITCAP.
[2]This was not true in the original SWITCAP

Input Sample-and-Hold

Suppose we have a laboratory setup as shown in Fig. 2.5 wherein a continuous sinusoidal waveform is first sampled-and-held by a sample-and-hold (S/H) circuit and then applied to the SCN in Fig. 2.3 at node (1). The waveforms v1(t) at node (1) and v4(t) at node (4) are also shown in Fig. 2.5. The input sampling instants are at 3 us, 11 us, and so forth with a period of 8 us. The waveform on node (4) changes only once per period because the input sampling instants have been chosen to coincide with the instants when switch S2 closes. In general, for other input sampling instants, this would not be the case. To produce the input waveform v1(t) in SWITCAP2, we can use the two statements:

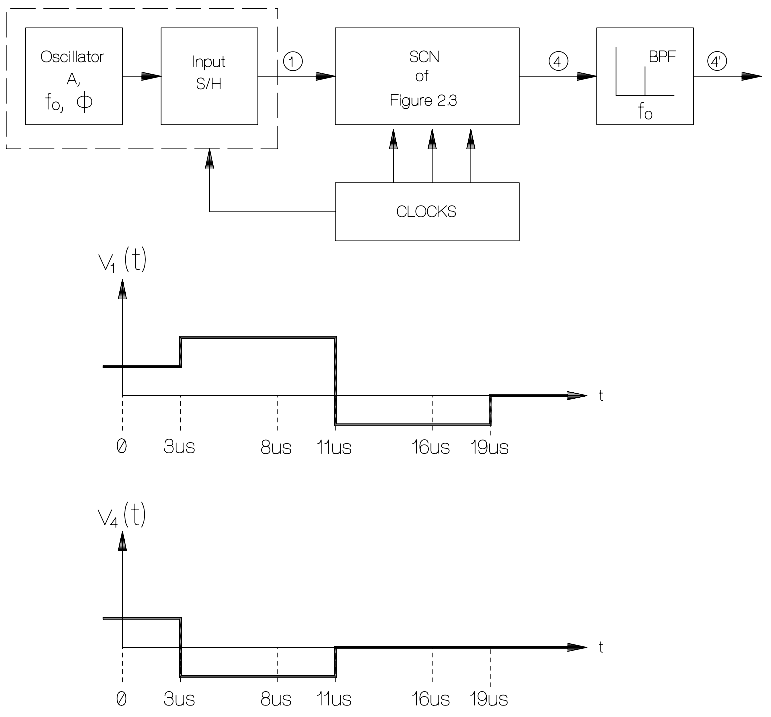

Figure 2.5: Input Sample-and-Hold.

```
20                 SET V1 AC 1.0 0.0;

21                 SAMPLE INPUT HOLD 1 3/8+;
```

Statement 20 defines a continuous sinusoidal waveform as before. Statement 21 effectively places an S/H circuit between this continuous-waveform source and node (1) as shown in Fig. 2.5. The "1" means that the period of the S/H is 1mcp or 8 us (mcp is defined by the PERIOD statement in Section 2.2.1). The "3/8+" means that a sampling action occurs at the instant just *after* the time 3/8mcp (3 us).

Our definition of sinusoidal steady-state response is still valid even with the input S/H circuit in place. Of course, the definition now applies to the total system consisting of the S/H, the SCN, and the BPF. To determine the s.s.s. response in Fig. 2.5 we can use the statements:

```
22          ANALYZE SSS;
23                 INFREQ 0.01 100000 LOG 15;
24                 SET V1 AC 1.0 0.0;
25                 SAMPLE INPUT HOLD 1 3/8+;
26                 PLOT VDB(4) VP(4);
27                 END;
```

Output Sample-and-Hold

The waveform on node (4) may also be sampled with another sample-and-hold circuit before it is analyzed as shown in Fig. 2.6. Suppose v4(t) is now sampled-and-held with a period of 8 us at instants just before 3 us, 11 us, and so forth to produce the waveform v4"(t) shown in Fig. 2.6. Note that this is just a delayed version of the waveform v4(t) in Fig. 2.5. To simulate this sampling function, we can use the statement

```
28                 SAMPLE OUTPUT HOLD 1 3/8-;
```

This statement effectively places an S/H circuit between the output nodes of the SCN specified by the PRINT and PLOT statements and the BPF as shown in Fig. 2.6. The minus (-) sign after "3/8" means that the sampling action occurs at the instant just *before* the time 3/8mcp. Hence the values picked up by the output S/H at each of these sampling instants are the *old* values of the waveform on node (4).

To determine the sinusoidal steady-state response in Fig. 2.6 (s.s.s. response is now defined with respect to the total system consisting of the input S/H, the SCN, the output S/H, and the BPF) we can use the statements:

```
29          ANALYZE SSS;
```

```
INFREQ 0.01 100000 LOG 15;
SET V1 AC 1.0 0.0;
SAMPLE INPUT HOLD 1 3/8+;
SAMPLE OUTPUT HOLD 1 3/8-;
PLOT VDB(4) VP(4);
END;
```

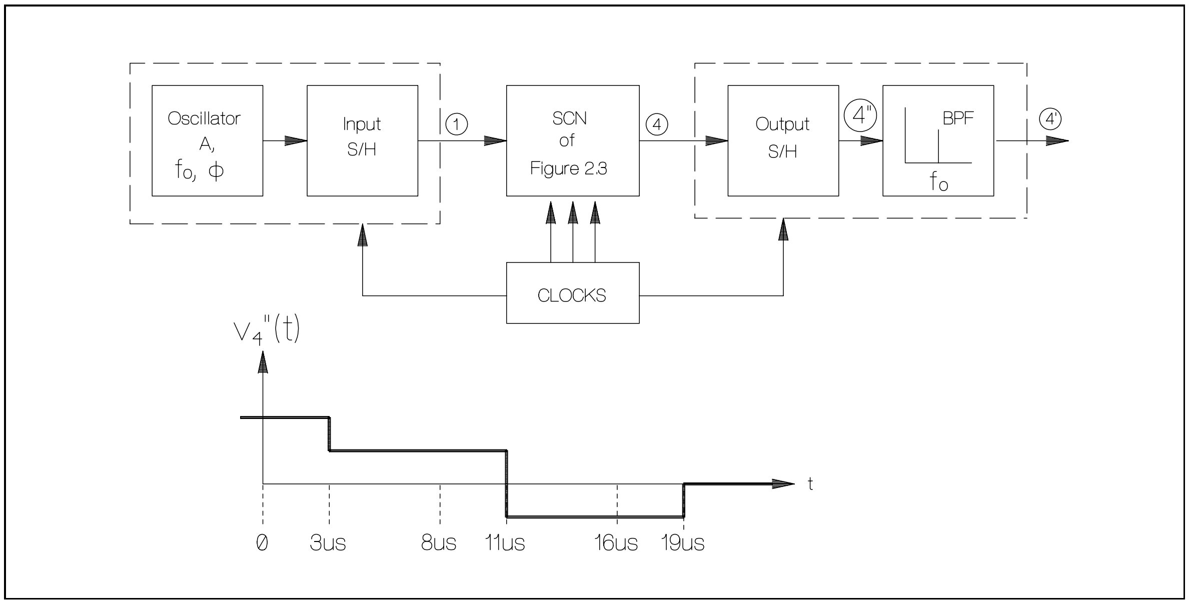

Figure 2.6: Output Sample-and-Hold.

Input sampling and output sampling may be used independently of each other. Moreover, when used, the input and output sampling periods and sampling instants may be independent of each other and independent of the switch-controlling clock periods and clock times of the SCN. (More than one sampling instants per sampling period may be specified. See Section 3.1).

*** IMPORTANT NOTE ***

In each ANALYZE block, a SAMPLE INPUT statement applies to all SET statements and a SAMPLE OUTPUT statement applies to all PRINT and PLOT statements. Each ANALYZE block can have at most one SAMPLE INPUT statement and at most one SAMPLE OUTPUT statement.

Output Impulse Sampling

If we replace the output S/H in Fig. 2.6 with an impulse sampler (I/S), then the output of the BPF can, for a given timing relationship among the input S/H, the SCN, and the output I/S, be related to a discrete-time transfer function (DTTF) associated with the SCN. For example, suppose we use the statement

```
36                    SAMPLE OUTPUT IMPULSE 1 3/8+;
```

in place of statement 33. Then the output S/H in Fig. 2.6 becomes an I/S whose sampling instants are at the instants just *after* 3 us, 11 us, and so forth. *The weight of each impulse is equal to the interval between successive pulses.* Since the source oscillator's continuous sinusoidal waveform in Fig. 2.6 has unit amplitude and 0 phase and since both the input S/H defined by statement 31 and the output I/S defined by statement 36 have uniform sampling intervals and the same sampling instants, we have the following result: The magnitude and phase of the output v4'(t) of the BPF in Fig. 2.6 (with an output I/S in place of the output S/H), written in the form of Eq. 2.2, is simply the magnitude and phase of the DTTF

$$H(z) = \frac{V_4(z)}{V_1(z)} \tag{2.3}$$

where $z = \exp(2\pi f_0 T)$ and where $V_1(z)$ and $V_4(z)$ are the z-transforms of the sequences $v_1(t_0 + nT^+)$ and $v_4(t_0 + nT^+)$, with $t_0 = 3$ us and $T = 8$ us. The DTTF in Eqn. 2.3 is only one of many DTTF's that can be implemented with the SCN in Fig. 2.3. We caution the reader that in general the particular DTTF implemented by an SCN is affected strongly by how the input sampling and output sampling are done in relation to the switch activities within the SCN, and the differences can be as drastic as having a lowpass response for one way of sampling and a highpass response for another. One should *be precise* in telling SWITCAP2 what to do.

In specifying sampling instants, the same rules apply to an output I/S as to an output S/H: An output I/S may be used without an input S/H, more than one sampling instants per sampling period may used, sampling intervals between successive impulses may unequal, and so forth. Of course, for these less restricted types of sampling, the output of the BPF we use for S.S.S. analysis may not correspond to any DTTF in the sense of Eq. 2.3. Nevertheless, the output of the BPF is well defined and may be computed.

Spectrum Analysis and Output Frequency Components

It has been noted that in sinusoidal steady-state analysis there is a bandpass filter (BPF) whose center frequency is matched to that of the input frequency, as shown in Fig. 2.4. It is quite possible to vary the center frequency of this BPF to pick up the other frequency components in the output waveform. Of course, when this is done, the frequency of the sinusoid at the output of the BPF will not be the same as that of the input sinusoid, and we do not consider the output of the BPF a sinusoidal steady-state response in the strict sense of the definition we have given before. Nevertheless, the amplitude and phase of the sinusoid at the output of the BPF are well defined and may be computed. In SWITCAP2, to allow movement of the center frequency of the BPF, we must first specify

```
ANALYZE SPECTRUM;
```

This statement defines an ANALYZE block in a manner analogous to the ANALYZE SSS statement. All of the statements that are used in an ANALYZE SSS block can also be used in an ANALYZE SPECTRUM block, including SAMPLE INPUT and SAMPLE OUTPUT statements. In addition, we need one more statement to specify the output frequency components that we want to look at. This is done with the OUTFREQ statement. Let us say that we are interested in the spectral content of $v_4(t)$ in the frequency range of 0 Hz to 300 KHz for two input frequencies, f_0 = 100 Hz and f_0 = 1000 Hz, for the circuit in Fig. 2.3. These frequency components can be computed with the statements

```
37          ANALYZE SPECTRUM;
38                INFREQ 100 1000 LOG 2;
39                OUTFREQ 0 300e3 SCANNED;
40                SET V1 AC 1.0 0.0;
41                PLOT VDB(4);
42                END;
```

Statement 39 is the only new statement here. It specifies that the range of the output frequency components is 0 to 300 KHz. The keyword "SCANNED" indicates that the output frequency is to be scanned in a fashion similar to the operation of a spectrum analyzer. The statements (37-42) result in the output:

```
----------------------------------------------------------------------
Output frequency component(s)  at  output frequencies <outfreq> related
to fixed input frequency f0 by <outfreq> = <n>*fs + <s>*f0.
----------------------------------------------------------------------
f0 = 1.00000000D+02                                 fs = 1.25000000D+05
----------------------------------------------------------------------
```

```
                                         (*): vdb(4)
 n  s  outfreq  -60.00      -40.00         -20.00          0.00         20.00
 0  1 1.0000+02 |=========================================================*-|
 1 -1 1.2490+05 |==========*                                                |
 1  1 1.2510+05 |==========*                                                |
 2 -1 2.4990+05 |======*                                                    |
 2  1 2.5010+05 |======*-------+--------------+--------------+--------------|
 n  s  outfreq  -60.00      -40.00         -20.00          0.00         20.00
                                         (*): vdb(4)
```

```
-----------------------------------------------------------------------
Output frequency component(s)  at  output frequencies <outfreq> related
to fixed input frequency f0 by <outfreq> = <n>*fs + <s>*f0.
-----------------------------------------------------------------------
f0 = 1.00000000D+03                                 fs = 1.25000000D+05
-----------------------------------------------------------------------
```

```
                                         (*): vdb(4)
 n  s  outfreq  -60.00 -50.00    -40.00    -30.00    -20.00    -10.00    0.00
 0  1 1.0000+03 |========================================================*--|
 1 -1 1.2400+05 |==============*                                            |
 1  1 1.2600+05 |==============*                                            |
 2 -1 2.4900+05 |========*                                                  |
 2  1 2.5100+05 |========*+---------+---------+---------+---------+---------|
 n  s  outfreq  -60.00 -50.00    -40.00    -30.00    -20.00    -10.00    0.00
                                         (*): vdb(4)
```

The magnitude and phase quantities computed in SPECTRUM analysis are those of the SINUSOIDAL WAVEFORM at the output of the BPF in Fig. 2.4. These quantities are related to the waveform by Eq. 2.2 (with f_0 replaced by the actual output frequency). Note that apart from the uniform spacing of the frequency components, the SWITCAP2 output above corresponds to what one would observe on the screen of a laboratory spectrum analyzer.

Another way to analyze the output frequency components is to use the ANALYZE FREQCOMP statement. ANALYZE FREQCOMP is very similar to ANALYZE SPECTRUM although the point of view is slightly different. In both analyses, we consider the output frequency components at frequencies $nf_s + f_0$ and $nf_s - f_0$, where f_0 is the input frequency, f_s is the switching frequency, and n is an integer. In SPECTRUM analysis, we view the input frequency f_0 as being fixed while the index n is varied. In FREQCOMP analysis the index n is considered fixed while the input frequency f_0 is varying. FREQCOMP analysis corresponds more closely with the viewpoint of using an SCN as a frequency modulator. For example, suppose we are interested in the output frequency components in the frequency range from 250 kHz to 300 kHz for inputs in the frequency range from 10 kHz to 50 kHz for the circuit in Fig. 2.3. This analysis can be obtained by the statements.

```
ANALYZE FREQCOMP;
      INFREQ 10e3 50e3 LIN 9;
      OUTFREQ 250e3 300e3 SCANNED;
      SET V1 AC 1.0 0.0;
      PLOT VDB(4) VP(4);
      END;
```

The resulting output is

```
-----------------------------------------------------------------------
Output frequency component(s)  at  output  frequencies related to input
frequencies by:
<outfreq>   =    2 * fs + <infreq>                  fs = 1.25000000D+05
-----------------------------------------------------------------------


                                         (X): vp(4)
                -200.0      -100.0           0.0           100.0        200.0
                |--------------+--------------+--------------+--------------|
 infreq outfreq                          (*): vdb(4)
10**+03 10**+03 -52.75 -52.50    -52.25    -52.00    -51.75    -51.50  -51.25
 10.000  260.00 |--X------+---------+---------+---------+---------+-*-------|
 15.000  265.00 |  X                                          *             |
 20.000  270.00 |  X                                   *                    |
 25.000  275.00 |                                *                       X  |
 30.000  280.00 |                         *                              X  |
 35.000  285.00 |                   *                                    X  |
 40.000  290.00 |             *                                          X  |
 45.000  295.00 |        *                                                 X   |
 50.000  300.00 |-*-------+--------+--------+--------+--------+------X--|
10**+03 10**+03 -52.75 -52.50    -52.25    -52.00    -51.75    -51.50  -51.25
 infreq outfreq                          (*): vdb(4)
```

The discontinuity in the phase function in this plot is an artifact of the computation. Actually the phase is around 180 deg, which is the same as -180 deg.

Group Delay Analysis

SWITCAP2 can compute group delay of linear SCN's exactly without using any numerical method to compute derivatives. It is assumed in SWITCAP2 that group delay is defined only at the fundamental frequency that is the same as the input frequency. Let us first define group delay. Let a complex function be of the form $F(\omega) = |F(\omega)|e^{j\phi(\omega)}$. Then, the group delay of the function $F(\omega)$ is defined as:

$$\tau(\omega) = -\frac{\partial\phi(\omega)}{\partial\omega} = -\mathrm{Im}\left[\frac{1}{f(\omega)}\frac{\partial F(\omega)}{\partial\omega}\right] \tag{2.4}$$

where Im[$\cdot$] denotes the operation "imaginary part of".

Since SWITCAP2 only computes group delay at the fundamental frequency, it is clear that SWITCAP2 computes the group delay of the output of the BPF in Fig. 2.4 ($v_4'(t)$ in Eqn. 2.2), if node (4) is taken as an output. Therefore, the input format to request group delay computation is simply to add **VGD** in PRINT and PLOT statements in S.S.S. analysis. For example, if we want to compute the group delay of node (4) from 1 Hz to 10 Hz, we can specify the following statements instead of statements (14 - 19):

```
49          ANALYZE SSS;
50                INFREQ 1 10 LIN 15;
51                SET V1 AC 1.0 0.0;
52                PRINT VDB(4) VP(4) VGD(4);
53                PLOT VDB(4) VP(4) VGD(4);
54                END;
```

The above input file gives the following output:

```
    frequency           vdb(4)      vp(4)       vgd(4)

 1.00000000D+00       5.6785D+01  9.7234D+01  1.9770D-02
 1.64285714D+00       5.2516D+01  9.4439D+01  7.3625D-03
 2.28571429D+00       4.9660D+01  9.3217D+01  3.7868D-03
 2.92857143D+00       4.7513D+01  9.2535D+01  2.2873D-03
 3.57142857D+00       4.5792D+01  9.2103D+01  1.5202D-03
 4.21428571D+00       4.4356D+01  9.1807D+01  1.0761D-03
 4.85714286D+00       4.3123D+01  9.1593D+01  7.9615D-04
 5.50000000D+00       4.2044D+01  9.1431D+01  6.0844D-04
 6.14285714D+00       4.1085D+01  9.1307D+01  4.7648D-04
 6.78571429D+00       4.0221D+01  9.1208D+01  3.8019D-04
 7.42857143D+00       3.9435D+01  9.1129D+01  3.0780D-04
 8.07142857D+00       3.8714D+01  9.1064D+01  2.5199D-04
 8.71428571D+00       3.8048D+01  9.1011D+01  2.0807D-04
 9.35714286D+00       3.7430D+01  9.0967D+01  1.7289D-04
 1.00000000D+01       3.6853D+01  9.0931D+01  1.4427D-04


                               (%): vgd(4)
                                 10**-03
      -5.00     0.00        5.00        10.00       15.00     20.00
      |-----------+-----------+-----------+-----------+-----------|

                               (X): vp(4)
      90.00        92.00          94.00          96.00        98.00
      |--------------+--------------+--------------+--------------|

                               (*): vdb(4)
freq  35.00     40.00       45.00       50.00       55.00     60.00
1.000 |-----------+-----------+-----------+-----------+---*-X----%|
1.643 |           .           .     %  X  .     *     .           |
2.286 |           .        %  X          *.           .           |
2.929 |           .    % X    .     *     .           .           |
3.571 |           .   =       . *         .           .           |
4.214 |           . X%      * .           .           .           |
4.857 | . . . . . X % . .*. . . . . . . . . . . . . . . . . . . . |
5.500 |          X.%   *      .           .           .           |
6.143 |         X .% *        .           .           .           |
 6.786 |       X  .=         .         .         .        |
 7.429 |      X  *.%         .         .         .        |
 8.071 |      X*  .%         .         .         .        |
 8.714 | . . .*X . % . . . . . . . . . . . . . . . . . . . |
 9.357 |    *X    %          .         .         .        |
10.000 |---*--X----%----------+----------+----------+----------|
freq  35.00    40.00      45.00      50.00      55.00    60.00
                               (*): vdb(4)
```

It should be emphasized that a frequency overlap condition may occur whenever the input frequency is an integral multiple of half the sampling frequency. When this condition happens, group delay at the frequency becomes meaningless. See Section 3.2 for more details.

SAMPLE INPUT HOLD, SAMPLE OUTPUT HOLD, and SAMPLE OUTPUT IMPULSE can also be used in group delay analysis.

Transient Analysis

SWITCAP2 can also analyze the transient response of an SCN given a prescribed input waveform. Suppose now that the input to the circuit in Fig. 2.3 has a pulse waveform shown in Fig. 2.7. The transient response of the circuit to such an input can be analyzed with the statements

```
ANALYZE TRAN;
     TIME 0+ 40 20/8;
     SET V1 PULSE 0 5 0 50e-6 50e-6 100e-6 500e-6;
     PLOT V(4) V(1);
     END;
```

Here, the ANALYZE block starts with ANALYZE TRAN for transient analysis. Two new statements are used.

TIME 0+ 40 20/8;

This statement specifies the range of time values for transient analysis. It is the time domain counterpart to the INFREQ statement for frequency domain analyses discussed previously. The time values are expressed in mcp (master clock period defined by the PERIOD statement in Sec. 2.2.1). This statement says: compute the response for time values from 0 mcp to 40 mcp at intervals of every 20/8 mcp. The

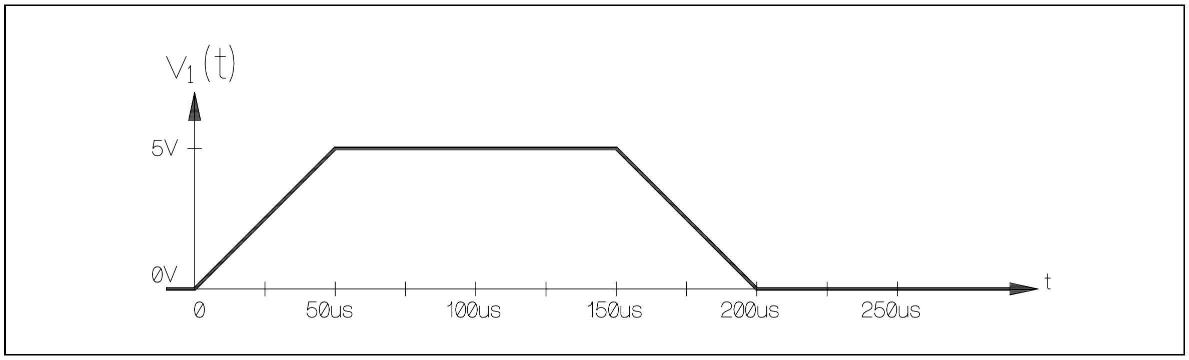

Figure 2.7: Input Pulse Waveform.

plus (+) sign appended to the "0" tells the program to output the values at the instants just *after* each output instant in case there are discontinuities in the output waveform. A minus (-) sign can also be used and it would mean "before each output instant". *The time values for output must all be after the time 0-.*

SET V1 PULSE 0 5 0 50e-6 50e-6 100e-6 500e-6;

This statement assigns the pulse waveform shown in Fig. 2.7 to the independent voltage source V1. The sequence of numbers after the keyword PULSE are

0 initial voltage of the pulse

5 peak voltage of the pulse

0 delay time to the start of the pulse

50e-6 rise time

50e-6 fall time

100e-6 pulse width

500e-6 period for repeated pulses.

When specifying time-domain waveforms in a SET statement, *all time values are expressed in absolute time units (not in mcp's).*

SAMPLE INPUT HOLD and SAMPLE OUTPUT HOLD statements may also be used in an ANALYZE TRAN block. SAMPLE OUTPUT IMPULSE statements are not allowed. Other types of waveforms besides PULSE may also be defined (see Sec. 3.3).

The statements (55-59) give the result

```
(X): v(1)
```

```
        -2.000       0.000          2.000          4.000        6.000
        |--------------+--------------+--------------+--------------|


  time                            (*): v(4)
10**-06 -4.000   -3.000      -2.000      -1.000       0.000     1.000
   0.00+|-----------+--X--------+-----------+-----------*-----------|
  20.00+|                             X          *                  |
  40.00+|                               *            X              |
  60.00+|                      *                             X      |
  80.00+|                  *                                 X      |
 100.00+|            *                                       X      |
 120.00+|       *                                            X      |
 140.00+| *                                                  X      |
 160.00+|*                                           X              |
 180.00+|  *                          X                             |
 200.00+|        *     X                                            |
 220.00+|        *     X                                            |
 240.00+|        *     X                                            |
 260.00+|        *     X                                            |
 280.00+|        *     X                                            |
 300.00+|        *     X                                            |
 320.00+|--------*--+--X--------+-----------+-----------+-----------|
10**-06 -4.000   -3.000      -2.000      -1.000       0.000     1.000
  time                            (*): v(4)
```

Subcircuits

The subcircuit facility in SWITCAP2 allows large circuits to be built up from smaller subcircuits in a hierarchical manner. To see how this facility can be used, suppose now that we wish to cascade two circuits, each like the circuit in Fig. 2.3, to form one circuit as shown in Fig. 2.8. Let Subcircuit A have identical values as the circuit in Fig. 2.3. In Subcircuit B, let C1 and C2 have different values and let the switches be operated in the opposite phase from that of Subcircuit A, as shown in Fig. 2.8.

To simulate the circuit in Fig. 2.8, we can first define a subcircuit with the statements

```
60        SUBCKT (1 4) lowpass (K:phi P:cap1 P:cap2);
61             S1 (1 2) phi;
62             S2 (2 3) #phi;
```

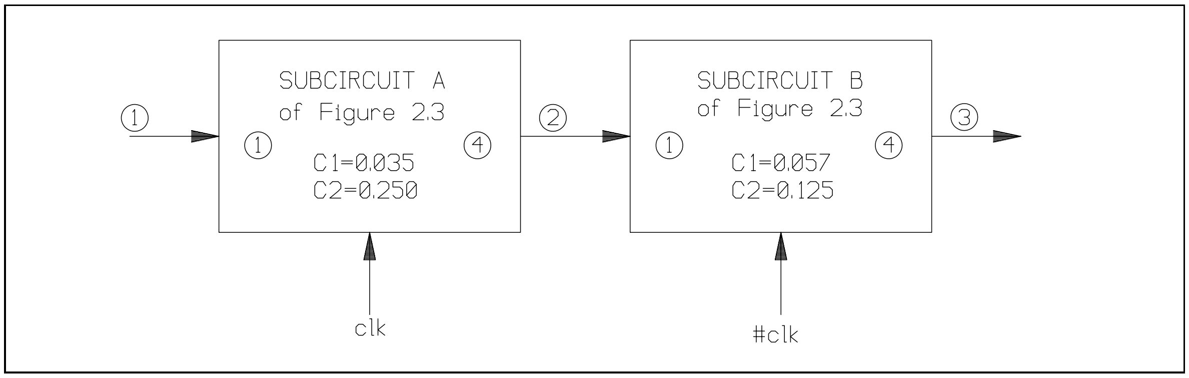

Figure 2.8: Cascade of Two Circuits.

```
63                    C1 (2 0) cap1;
64                    C2 (1 3) cap2;
65                    C3 (3 4) 1.000;
66                    E1 (4 0 0 3) 5500;
67                    END;
```

The keywords “SUBCKT” and “END” define a SUBCKT block. Some of the features of statements (60-66) are as follows.

SUBCKT (1 4) lowpass (K:phi P:cap1 P:cap2);

Nodes (1) and (4) are the “terminal” nodes through which the subcircuit may be connected to the outside world (besides the ground node). All nodes used within the subcircuit other than the ones declared here and other than the ground node are considered internal to the subcircuit and cannot be seen by rest of the world. For example, nodes (2) and (3) are internal nodes in this subcircuit. The names of the nodes (both the terminal nodes and the internal nodes) follow the same rules as node names used for elements (see Sec. 2.2.2) and may be arbitrarily chosen. One exception to the rules is the length of internal node names within a subcircuit block. This will be explained in page **??**. Node names in different SUBCKT blocks are completely independent. The ground node, however, is always the node (0) and is always the same node in a subcircuit as the ground node in the main circuit. There is a restriction on the length of the name of nodes and elements. This will be explained in page 38.

“lowpass” is the name of the subcircuit. It is the name by which this subcircuit will be referred when it is later used.

The sequence "K:phi" says that phi is a *symbolic clock.* [3] The symbolic clock name, phi in this instance, may be arbitrarily chosen. This means that phi can be used anywhere in the subcircuit where a clock name would normally go in identifying the clock controlling a switch.

The sequences "P:cap1" and "P:cap2" say that cap1 and cap2 are *symbolic values.* [4] This means that cap1 and cap2 can be used anywhere in the subcircuit where a real number would normally go, e.g., a capacitance value.

Symbolic names such as phi and cap1 must be a string of 78 or fewer alphanumeric characters, the first of which must be a letter (a-z).

IMPORTANT NOTE

In the original SWITCAP, the first parentheses contain analog terminal nodes and the second parentheses contain parameters. However, in SWITCAP2, this format may be relaxed. The only restriction in SUBCKT definition is to keep two sets of parenthesis around the subcircuit name. For example, any of the following produces the same result.

```
SUBCKT (1 4) lowpass (K:phi P:cap1 P:cap2);
SUBCKT (1 4 K:phi P:cap1 P:cap2) lowpass ();
SUBCKT () lowpass (K:phi P:cap1 P:cap2 1 4);
```

S1 (1 2) <u>phi</u>;

S2 (2 3) <u>#phi</u>;
These switches are controlled by symbolic clocks. Note that it is ok to complement a symbolic clock.

C1 (2 0) <u>cap1</u>;

C2 (1 3) <u>cap2</u>;

C3 (3 4) <u>1.000</u>;
Symbolic values and constant values can both be used within a subcircuit.

Now to cascade the two circuits together, we can use the statements

```
68                    CIRCUIT;
```

[3] In the original SWITCAP, the letter K was needed to tell SWITCAP that what comes after the colon (:) is a symbolic clock. However, "K:" is not required in SWITCAP2 any more.

[4] Again, "P:" can be omitted.

```
69                    Xa (1 2) lowpass ( clk 0.035 0.250);
70                    Xb (2 3) lowpass (#clk 0.057 0.125);
71                    V1 (1 0);
72                    END;
```

In statements 69 and 70, the key character "X" causes the subcircuit defined previously to be "pulled" into the circuit. We call this action a "subcircuit call." The terminal nodes of the subcircuits are now assigned to actual nodes and the symbolic parameters in the subcircuits are replaced by the actual values to be used. Note that it is ok to pass a complemented clock name to a subcircuit. The names "a" and "b" in statements 69 and 70 given to the subcircuit calls follow the same rules as element names (see Sec. 2.2.2) and may be arbitrarily chosen.

Subcircuit definitions may also contain subcircuit calls. In these definitions, symbolic parameters may also be passed to the subcircuit call. **Independent voltage sources are not allowed in SUBCKT blocks.**

IMPORTANT NOTE

> Internally, element and node names within a subcircuit are resolved by concatenating names. For example, when the above subcircuit definition "lowpass" is called in CIRCUIT through statement 69, then the name for C1 in the subcircuit definition becomes C1.Xa and the internal node 2 becomes 2.Xa. Therefore, the length of the capacitor and node name is 5 and 4, respectively. SWITCAP2 currently allows the length of this *internal* name to be 78 characters. Thus, a caution must be taken when many layers of subcircuit calls are used.

SUBCKT blocks must be placed in an input file in the order

TIMING block
SUBCKT block(s)
CIRCUIT block
ANALYZE block(s)

SUBCKT blocks may be written in any order among themselves so long as a subcircuit is first defined before is called.

2.2.6 Sensitivity Analysis

As in the case with group delay analysis, sensitivity is obtained only at the fundamental frequency that is the same as the input frequency. First, the definition of sensitivity is given.

Let a complex function be of the form $F(\omega) = |F(\omega)|e^{j\phi(\omega)}$. Then, the first order relative sensitivity of the function $F(\omega)$ with respect to a network parameter α_i is defined as:

$$S_{\alpha_i}^{F(\omega)} = \frac{\alpha_i}{F(\omega)}\frac{\partial F(\omega)}{\partial \alpha_i} = \frac{\partial ln\, F(\omega)}{\partial ln\, \alpha_i} = \frac{\alpha_i}{\mid F(\omega) \mid}\frac{\partial \mid F(\omega) \mid}{\partial \alpha_i} + j\, \alpha_i\, \frac{\partial \phi(\omega)}{\partial \alpha_i}. \tag{2.5}$$

The more meaningful forms of the above equation are the magnitude $S_{\alpha_i}^{|F(\omega)|}$ and phase $S_{\alpha_i}^{\phi(\omega)}$ sensitivities with respect to a network parameter α_i. They are defined respectively as follows:

$$S_{\alpha_i}^{|F(\omega)|} = \frac{\alpha_i}{|F(\omega)|}\frac{\partial |F(\omega)|}{\partial \alpha_i} = \text{Re}\left[S_{\alpha_i}^{F(\omega)}\right] \tag{2.6}$$

and

$$S_{\alpha_i}^{\phi(\omega)} = \alpha_i \frac{\partial \phi(\omega)}{\partial \alpha_i} = \text{Im}\left[S_{\alpha_i}^{F(\omega)}\right] \tag{2.7}$$

where Re[·] and Im[·] denote the operations "real part of" and "imaginary part of", respectively. Note that if $|F(\omega)|$ is zero then the magnitude sensitivity is defined as follows:[5]

$$S_{\alpha_i}^{|F(\omega)|} = \alpha_i \frac{\partial |F(\omega)|}{\partial \alpha_i} \tag{2.8}$$

IMPORTANT NOTE

SWITCAP2 computes $S_{\alpha_i}^{F(\omega)}$, therefore, when selecting the form of the output the real part must be requested if magnitude sensitivity is desired and the imaginary part must be requested if the phase sensitivity is desired. (This corresponds to "VR" and "VI" in SWITCAP2)

In SWITCAP2, the sensitivity analysis is invoked by specifying ANALYZE SENS block. Since the sensitivities are only computed at the fundamental frequency as explained before, the specification of ANALYZE SENS is almost the same as ANALYZE SSS block. For example, if we want to compute the magnitude and phase sensitivities of the circuit in Fig. 2.3 with respect to C1, C2, and E1 from 1 Hz to 10 Hz, we can specify the following statements:

```
73          ANALYZE SENS;
74               INFREQ 1 10 LIN 15;
75               SET V1 AC 1.0 0.0;
```

[5]For more details on definition of sensitivity see: Jiri Vlach and Kishore Singhal: *Computer Methods for Circuit Analysis and Design.* Van Nostrand Reinhold, New York, 1983

```
76                        PERTURB C1 C2 E1;
77                        PRINT VR(4) VI(4);
78                        PLOT VR(4) VI(4);
79                        END;
```

The descriptions of INFREQ, SET, PRINT, and PLOT are the same as in Sec. 2.2.3. More detailed descriptions are also available in Chap. 3. The only new statement is PERTURB.

PERTURB C1 C2 E1;

This line specifies the element names whose sensitivities are of interest. SWITCAP2 allows capacitor values and gain of VCVS's to be "perturbed". Despite the command name, PERTURB, numerical methods such as finite difference method are *not* used to compute derivatives. Rather, they are computed through analytical expressions.

An element within a subcircuit can be specified in PERTURB statement by concatenating names as explained in page 38. For example, if C1 in SUBCKT "lowpass" through a subcircuit call Xa (see page 38) is of interest, then type:

```
PERTURB C1.Xa
```

There must be only *one* PERTURB statement within ANALYZE SENS block. Multiple lines can be used to specify one PERTURB statement.

PRINT VR(4) VI(4);

PLOT VR(4) VI(4);

The magnitude and phase sensitivities are requested at node 4.

NOTE: "VR" and "VI" were used to obtain magnitude and phase sensitivities respectively. If instead, "VM" and "VP" were used the output would be the magnitude and phase of the sensitivity ($S_{\alpha_i}^{F(\omega)}$) itself. (see the above discussion on the definition of senstitivity)

IMPORTANT NOTE

Any sampling functions such as SAMPLE INPUT HOLD, SAMPLE OUTPOUT HOLD, and SAMPLE OUTPUT IMPULSE are currently *not* supported in ANALYZE SENS block.

Statements 73-79 produce the following output:

```
     frequency         vr(4)/E1    vr(4)/C2    vr(4)/C1

 1.00000000D+00        1.5994D-02 -4.4599D-05  9.8423D-01
 1.64285714D+00        6.1297D-03 -4.4828D-05  9.9410D-01
 2.28571429D+00        3.2867D-03 -4.4632D-05  9.9694D-01
 2.92857143D+00        2.0944D-03 -4.4254D-05  9.9813D-01
 3.57142857D+00        1.4845D-03 -4.3743D-05  9.9874D-01
 4.21428571D+00        1.1314D-03 -4.3114D-05  9.9909D-01
 4.85714286D+00        9.0885D-04 -4.2372D-05  9.9932D-01
 5.50000000D+00        7.5960D-04 -4.1521D-05  9.9946D-01
 6.14285714D+00        6.5467D-04 -4.0560D-05  9.9957D-01
 6.78571429D+00        5.7812D-04 -3.9493D-05  9.9964D-01
 7.42857143D+00        5.2056D-04 -3.8317D-05  9.9970D-01
 8.07142857D+00        4.7619D-04 -3.7035D-05  9.9974D-01
 8.71428571D+00        4.4127D-04 -3.5645D-05  9.9978D-01
 9.35714286D+00        4.1330D-04 -3.4149D-05  9.9980D-01
 1.00000000D+01        3.9054D-04 -3.2546D-05  9.9982D-01


-----------------------------------------------------------------
     frequency         vi(4)/E1    vi(4)/C2    vi(4)/C1

 1.00000000D+00       -1.2455D-01  3.5338D-04  1.2422D-01
 1.64285714D+00       -7.6571D-02  5.8637D-04  7.5999D-02
 2.28571429D+00       -5.5193D-02  8.1815D-04  5.4385D-02
 2.92857143D+00       -4.3129D-02  1.0495D-03  4.2087D-02
 3.57142857D+00       -3.5387D-02  1.2807D-03  3.4113D-02
 4.21428571D+00       -3.0000D-02  1.5117D-03  2.8494D-02
 4.85714286D+00       -2.6035D-02  1.7427D-03  2.4297D-02
 5.50000000D+00       -2.2995D-02  1.9737D-03  2.1026D-02
 6.14285714D+00       -2.0591D-02  2.2046D-03  1.8390D-02
 6.78571429D+00       -1.8642D-02  2.4355D-03  1.6210D-02
 7.42857143D+00       -1.7030D-02  2.6664D-03  1.4366D-02
 8.07142857D+00       -1.5674D-02  2.8972D-03  1.2780D-02
 8.71428571D+00       -1.4518D-02  3.1281D-03  1.1393D-02
 9.35714286D+00       -1.3521D-02  3.3589D-03  1.0165D-02
 1.00000000D+01       -1.2652D-02  3.5898D-03  9.0647D-03
```

```
                               (%): vr(4)/C1
       9xx.xx                     10**-03                    10xx.xx
       $80.00      $85.00         $90.00         $95.00       $00.00
       |--------------+--------------+--------------+--------------|

                               (X): vr(4)/C2
                                  10**-06
       -45.00 -42.50    -40.00    -37.50    -35.00    -32.50  -30.00
       |---------+---------+---------+---------+---------+---------|

                               (*): vr(4)/E1
                                  10**-03
 freq  0.00         5.00           10.00          15.00        20.00
 1.000 |-X----------%-+--------------+--------------+--*-----------|
 1.643 |X             .  *           .           %  .              |
 2.286 |X        *    .              .              .     %        |
 2.929 |  X  *        .              .              .        %     |
 3.571 |   *X         .              .              .          %   |
 4.214 |  *    X      .              .              .           %  |
 4.857 | .*. . . .X. ... . . . . . . . . . . . . . ... . . . . . % |
 5.500 | *           X.              .              .            % |
 6.143 | *            .  X           .              .             %|
 6.786 | *            .      X       .              .             %|
 7.429 | *            .           X  .              .             %|
 8.071 |*             .              . X            .             %|
 8.714 |*. . . . . . ... . . . . . . . . . .X. . . ... . . . . . .%|
 9.357 |*             .              .            X .             %|
10.000 |*-------------+--------------+--------------+----X--------%|
 freq  0.00         5.00           10.00          15.00        20.00
                                  10**-03
                               (*): vr(4)/E1


                               (%): vi(4)/C1
                                  10**-03
       0.0     25.0      50.0      75.0      100.0     125.0   150.0
       |---------+---------+---------+---------+---------+---------|
                               (X): vi(4)/C2
                                  10**-03
       0.000        1.000          2.000          3.000        4.000
       |--------------+--------------+--------------+--------------|

                               (*): vi(4)/E1
                                  10**-03
 freq  -150.0 -125.0    -100.0     -75.0     -50.0     -25.0     0.0
 1.000 |----X----*---------+---------+---------+---------%---------|
 1.643 |        X.         .        *%         .         .         |
 2.286 |         . X       . %       .       * .         .         |
 2.929 |         .     X%  .         .         .  *      .         |
 3.571 |         .   %    X.         .         .     *   .         |
 4.214 |         .%        .  X      .         .       * .         |
 4.857 | . . . . % . . . . . . . X . . . . . . . . . . . * . . . . |
 5.500 |       % .         .         X         .         .*        |
 6.143 |      %  .         .         .  X      .         . *       |
 6.786 |     %   .         .         .      X  .         .  *      |
 7.429 |     %   .         .         .         X         .  *      |
 8.071 |    %    .         .         .         .  X      .   *     |
 8.714 | . .%. . . . . . . . . . . . . . . . . . . . .X. . . * . . |
 9.357 |   %     .         .         .         .         X    *    |
10.000 |---%-----+---------+---------+---------+---------+---X*----|
 freq  -150.0 -125.0    -100.0     -75.0     -50.0     -25.0     0.0
                                  10**-03
```

2.3 Time-Domain analysis of Mixed SC/Digital Networks

This section considers the time-domain simulation of mixed switched-capacitor/digital (SC/D) networks containing capacitors, independent and linear dependent voltage sources, switches, comparators, and logic gates. The switches, in particular, may be controlled by not only periodic waveforms but also non-periodic waveforms from the outputs of comparators and logic gates. Since this section explains considerably different types of SCN's from the previous example, we first define mixed SC/D networks, and then point out the syntax for new commands through several examples.

2.3.1 Definition of mixed SC/D networks

The structure of a mixed SC/D network defined in SWITCAP2[6] is shown in Fig. 2.9. The network is divided into four groups: SCN, THRESHOLD, TIMING, and LOGIC, which are interconnected as shown in the figure. The signal loop from the SCN to itself through the THRESHOLD and LOGIC allows signal-dependent modification of the SCN topology.

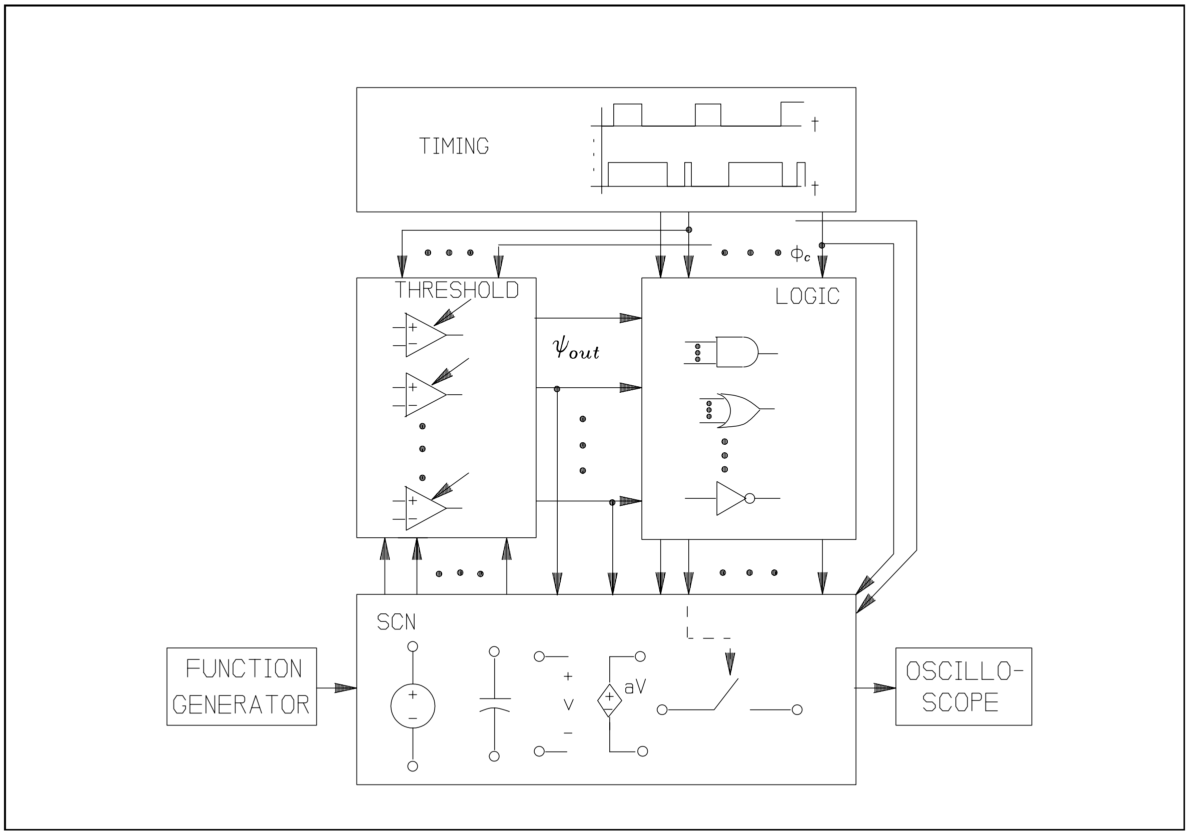

Figure 2.9: Structure of mixed SC/D network defined in SWITCAP2.

SCN

The SCN may be composed of linear capacitors, independent voltage sources, linear voltage-controlled voltage sources, and ideal switches which are controlled by periodic or

[6]More detailed discussions for the structure is published in [11, 12] in Chap. 1.

nonperiodic switch control signals. The switch control signals may be periodic or nonperiodic binary signals and they are supplied by the TIMING, THRESHOLD and LOGIC. The output variables of the SCN are node voltages; a subset of these voltages is monitored by the THRESHOLD as shown in Fig. 2.9. The topology of the network within the SCN can be set arbitrarily except for obvious pathological cases such as loops of voltage sources or disjoint networks.

THRESHOLD

The THRESHOLD contains *latched* comparators which act as the interface from the SCN to the LOGIC. A latched comparator (Fig. 2.10) is a comparator with non-inverting and inverting input terminals (signals at the terminals are denoted as $v_P(t)$ and $v_N(t)$, respectively as shown in Fig. 2.10), an enable terminal for a binary input $\psi_{enb}(t)$, and a binary output $\psi_{out}(t)$. The latched comparator *samples* its input signals $v_P(t)$ and $v_N(t)$ at the moment immediately before a switching instant t_k and, if warranted, changes its output value accordingly only when the enable signal $\psi_{enb}(t)$ changes from 0 to 1 at t_k. Then, it latches the output value until the next rising edge is observed at the enable terminal. For example, in Fig. 2.10, the output $\psi_{out}(t)$ of the latched comparator can change only at switching instants $t = t_{k-2}, t_k$, and t_{k+2}. Although the inputs $v_P(t)$ and $v_N(t)$ in Fig. 2.10 intersect at time instants within the intervals Δ_{k-2} and Δ_{k+1}, the output $\psi_{out}(t)$ does *not* change at these instants. The enable signal *must be periodic* and is generated by the TIMING. It should be emphasized that the latched comparator can be modeled by a composite circuit which consists of an ideal comparator and an edge-triggered D-type flip-flop as shown in Fig. 2.11.

TIMING

This block generates periodic clock waveforms which may be used as enable signals for latched comparators in the THRESHOLD, inputs to the LOGIC, or switch control signals directly. Detail description of the SWITCAP2 syntax for clocks can be found in Sec. 2.2.1.

LOGIC

The LOGIC realizes boolean logic functions *without* internal clocks. Clocks should be provided by the TIMING. It consists of basic logic gates such as AND, OR, NOT, NAND, NOR, XOR, and XNOR. Multiple inputs are allowed for applicable gates. Arbitrary combinational and sequential logic circuits (with external clock signals supplied by the TIMING for the latter) can be designed using the available logic gates. All gates are assumed to have *zero gate delay* and, therefore, the outputs from the LOGIC are determined immediately at the instants that the inputs change their states. The inputs to the LOGIC may consist of

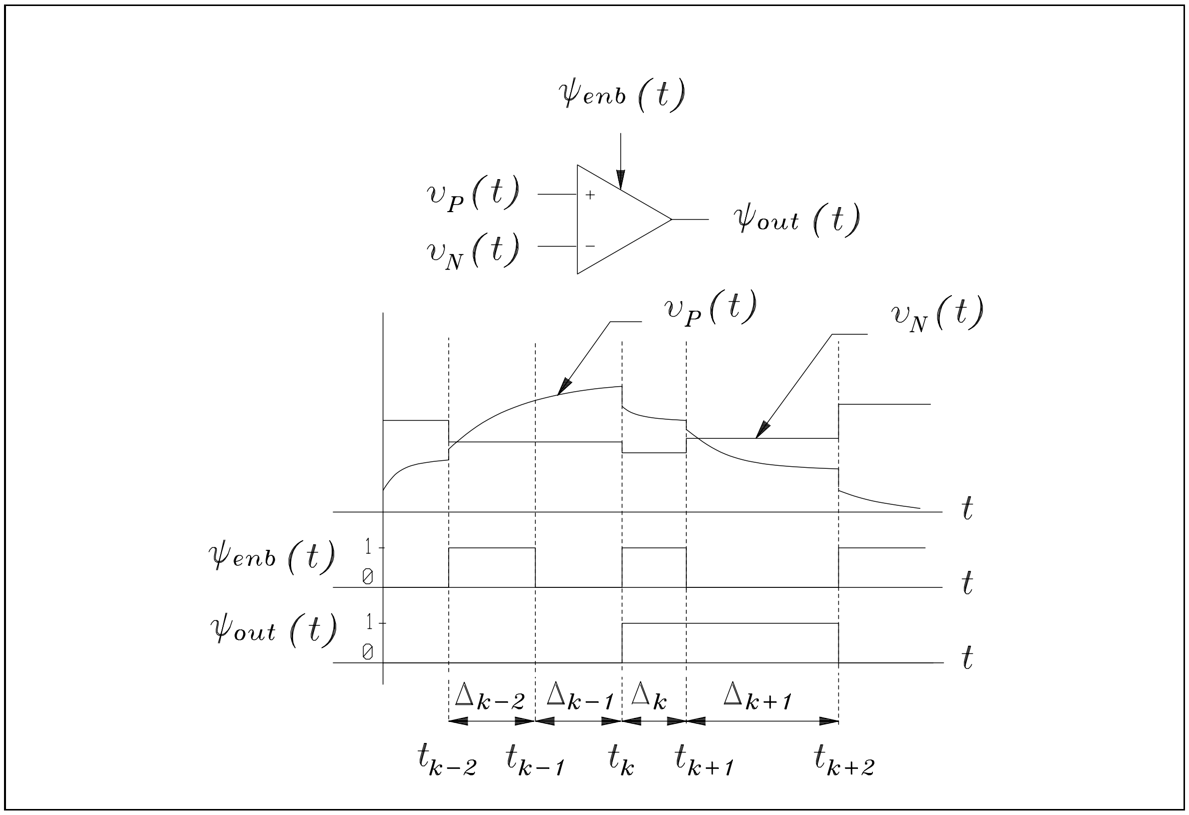

Figure 2.10: Latched comparator and its operation.

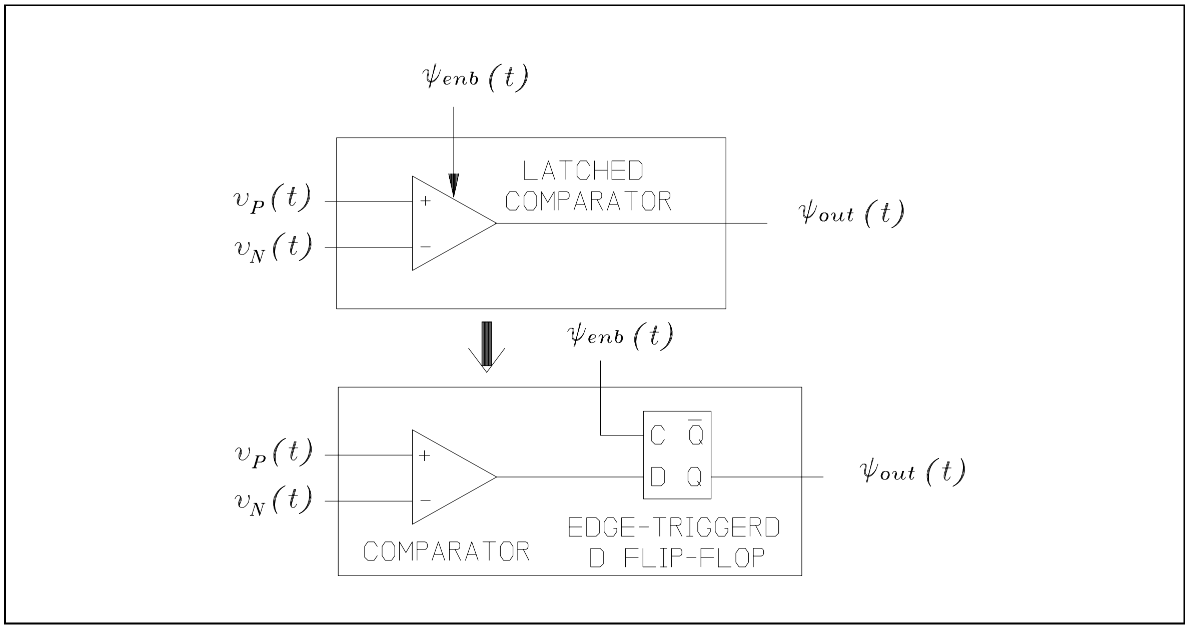

Figure 2.11: An equivalent model of a latched comparator.

the outputs of latched comparators in the THRESHOLD and of periodic clock waveforms generated by the TIMING as shown in Fig. 2.9.

2.3.2 SC Sigma-Delta Modulator

A SCN implementation of a second-order sigma-delta modulator used in high-precision A/D conversion[7] is shown in Fig. 2.12. The clock waveform ϕ_1 in the timing diagram is used for the enable signal of the latched comparator. Therefore, the comparison of the integrator outputs is done at the rising edge of ϕ_1.

Input file for the modulator is shown below:

```
TITLE: BOSER AND WOOLEY SIGMA DELTA MODULATOR
TIMING;
  PERIOD 5.9604644E-7
  CLOCK PHI1 1 (0 1/2);
  CLOCK PHI2 1 (1/2 1);
END;
```

[7]B. E. Boser and B. A. Wooley, "The design of sigma-delta modulation analog-to-digital converters," *IEEE J. Solid-State Circuits,* vol. SC-23, pp. 1298-1308, Dec. 1988.

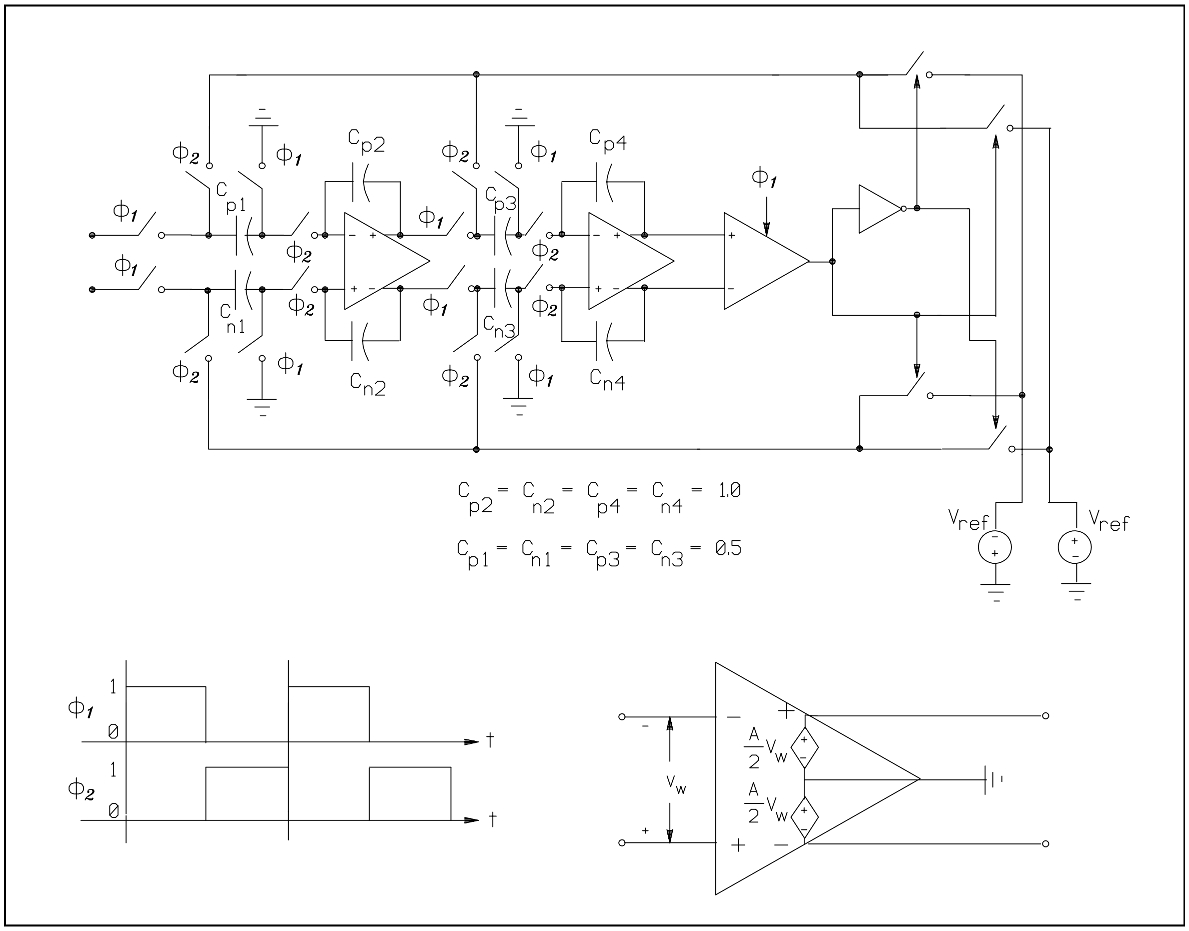

Figure 2.12: (a)SCN implementation of a second-order sigma-delta modulator, (b) timing waveforms, and (c) a balanced op amp model.

```
SUBCKT (N1 N2 N3 N4) DIFAMP (P1);
  E1 (N1 0 N3 N4) P1;
  E2 (0 N2 N3 N4) P1;
END;

CIRCUIT;
  CP1 (3 5) 0.5;
  CP2 (7 9) 1.0;
  CP3 (11 15) 0.5;
  CP4 (17 19) 1.0;
  CN1 (4 6) 0.5;
  CN2 (8 10) 1.0;
  CN3 (12 16) 0.5;
  CN4 (18 20) 1.0;
  SP1 (1 3) PHI1;
  SP2 (3 13) PHI2;
  SP3 (5 0) PHI1;
  SP4 (5 7) PHI2;
  SP5 (9 11) PHI1;
  SP6 (11 13) PHI2;
  SP7 (15 0) PHI1;
  SP8 (15 17) PHI2;
  SN1 (2 4) PHI1;
  SN2 (4 14) PHI2;
  SN3 (6 0) PHI1;
  SN4 (6 8) PHI2;
  SN5 (10 12) PHI1;
  SN6 (12 14) PHI2;
  SN7 (16 0) PHI1;
  SN8 (16 18) PHI2;
  SREFP1 (13 21) CMPNEG;
  SREFN1 (22 13) CMPPOS;
  SREFP2 (21 14) CMPPOS;
  SREFN2 (22 14) CMPNEG;
  YCMP1 CMPLAT (19 20 PHI1 CMPPOS);
  YG1 NOT (CMPPOS CMPNEG);
  X1 (9 10 8 7) DIFAMP (1E6);
  X2 (19 20 18 17) DIFAMP (1E6);
```

```
    VREFP (21 0);
    VREFN (22 0);
    VINP (1 0);
    VINN (2 0);
  END;

  ANALYZE NTRAN;
    TIME 65535/2- 98306 1;
    SET VREFP DC -2.5;
    SET VREFN DC +2.5;
    SET VINP COSINE 0.0 1.0 1024 0 0 -90;
    SET VINN COSINE 0.0 -1.0 1024 0 0 -90;
    PRINT V(1) V(13);
  END;
END;
```

2.3.3 Timing Section

Input format for TIMING section is the same as before. Refer to Sec. 2.2.1 for its syntax and Sec. 3.1 for multiple clocks.

2.3.4 Network Section

As it is described in Sec. 2.2.2, the keywords "CIRCUIT" and "END" define a CIRCUIT block. The input format for capacitors and voltage sources are the same as in Sec. 2.2.2. This section describes new elements (such as latched-comparators and digital gates) and new features associated with switches. Let's extract the essential part from the above input description.

SREFP1 (13 21) <u>CMPNEG</u>;

SREFN1 (22 13) <u>CMPPOS</u>;

SREFP2 (21 14) <u>CMPPOS</u>;

SREFN2 (22 14) <u>CMPNEG</u>;

The difference between these switches and the switches that are explained in Sec. 2.2.2 is that the controlling signals for the above switches are provided from latched-comparators and digital gates. CMPPOS is the output of a latched-comparator and CMPNEG is the output of an inverter.

YCMP1 CMPLAT (19 20 PHI1 CMPPOS);

A latched comparator must be specified with the following format:

$$\mathbf{Y}< elementname > \ CMPLAT\ (n+ \quad n- \quad enable \quad output);$$

where

Y The key character “Y” must be used to specify latched-comparators and logic gates.

$<$ *elementname* $>$ This is a string of alphanumeric characters less than 78 characters long which serves as a label for the comparator.

CMPLAT This is the key character to specify that this element is a latched comparator.

$n+$ This is an analog node in which the non-inverting input of the comparator is connected to.

$n-$ This is an analog node in which the inverting input of the comparator is connected to.

enable This is a clock signal which controls the enable terminal of the comparator (see Fig. 2.10 and Fig. 2.11). The clock signal *must* be generated in TIMING block and fed *directly* to the enable terminal.

output Note that the output generates *logic states,* “1” or “0”. Therefore, it can only be used as a switch control signal and/or an input to logic gates. If each logic state must be converted to an analog voltage, a circuit scheme shown in Fig. 2.12 (other schemes are possible, of course) may be used.

YG1 NOT (CMPPOS CMPNEG);

A logic gate has the following input format:

$$\mathbf{Y}< elementname > \ < elementtype > (in1 \quad in2 \quad in3 \ldots inN \quad out);$$

The key character “Y” is needed to define logic gates. The *elementname* should be a string of alphanumeric characters with a length less that 78 characters long. The $<$ *elementtype* $>$ can be any of the following:

AND
OR
NAND
NOR
XOR
XNOR

NOT

Except NOT, multiple inputs are allowed. SUBCKT can also be used for logic gates and latched-comparators (see Sec. 2.3.7).

It should be noted that output of logic gates generates *logic* values. Therefore, it can only be connected to inputs of other logic gates or to switch control terminal.

2.3.5 Nonlinear Transient Analysis:NTRAN

It is important to note that latched-comparators and logic gates in SWITCAP2 are valid *only* in transient analysis, NTRAN. The transient response of the above circuit can be obtained with the following statements:

```
ANALYZE NTRAN;
        TIME 65535/2- 98306 1;
        SET VREFP DC -2.5;
        SET VREFN DC +2.5;
        SET VINP COSINE 0.0 1.0 1024 0 0 -90;
        SET VINN COSINE 0.0 -1.0 1024 0 0 -90;
        PRINT V(1) V(13);
END;
```

Here, the statement ANALYZE NTRAN must be used to invoke the transient analysis of mixed SC/D networks. Latched-comparators and logic gates are *not* allowed in TRAN. The syntax for the other statements shown above is the same as that in TRAN block (see Sec. 2.2.5).

IMPORTANT NOTE

Currently, PRINT and PLOT statements are applicable only for analog nodes. A logic value must be converted into an analog value through switches and voltage sources. This will be fixed in subsequent releases.

IMPORTANT NOTE

Any sampling functions such as SAMPLE INPUT HOLD, SAMPLE OUTPOUT HOLD, and SAMPLE OUTPUT IMPULSE are *not* supported in NTRAN block.

2.3.6 Initialization

In ANALYZE NTRAN block, it is possible to initialize analog and logic nodes at $t = 0^-$. Initialization is particularly important when a sequential logic circuit is present in the network. If the unknown state "X" occurs, then a warning is given and the state is replaced by "0".

Let's assume that we want to initialize node (19) and (20) to +0.5 and -0.5 volts and CMPPOS and CMPNEG as "1" and "0". Its syntax is as follows:

```
INIT V(19)=0.5   V(20)=-0.5   L(CMPPOS)=1   L(CMPNEG)=0
```

The key character "V" and "L" must be used to initialize analog and logic nodes, respectively. There must be only *one* INIT statement in an NTRAN block. If you have many nodes to be initialized, you can have multiple lines of initialization after the INIT statement. The program keeps reading the initialization lines until it sees another statement such as TIME, SET, PRINT, PLOT, or END.

2.3.7 Subcircuit Revisited: Initialization Within Subcircuit

The subcircuit facility (explained in Sec. 2.2.5) may also include latched-comparators and logic gates. It is possible to define a subcircuit containing analog components or logic gates, or the mixture of both analog and logic components. The procedure for subcircuit definition and calling sequence is the same as in Sec. 2.2.5. In this section, the initialization within SUBCKT block and logic gates within SUBCKT are explained in detail.

Let us assume that an edge-triggered D-type flip-flop is defined in a SUBCKT block called DFF. Then, the SUBCKT block is referenced in another SUBCKT block, STEP. In CIRCUIT block, both subcircuit blocks are called. The SWITCAP2 input file would look like the following (only relevant lines are shown):

```
SUBCKT () DFF (CLK DATA Q QB);
                    /* D TYPE FLIP FLOP */
                    YG1 NAND (L4 L2 L1);
                    YG2 NAND (L1 CLK L2);
                    YG31 AND (L2 CLK L5);
                    YG32 NAND (L5 L4 L3);
                    YG4 NAND (L3 DATA L4);
                    YG5 NAND (L2 QB Q);
                    YG6 NAND (Q L3 QB);
END;

SUBCKT () STEP (PHI1 PHI2 SIGN OUT1 OUT2);
```

```
                    .
                    .
                    .
                    XGD1 () DFF (L5 L2 L3 L6);
                    .
                    .
END;

CIRCUIT;
                    .
                    .
                    .
                    YCMP1 CMPLAT (1 0 PHILAT CMPOUT);
                    XDFF1 () DFF (PHISGN CMPOUT PHINRF PHIPRF);
                    XSTEP1 () STEP (PHIY4 PHIY5 LS5 CNTY11 CNTY12);
                    XSTEP2 () STEP (PHIY3 PHIY4 LS5 CNTY21 CNTY22);
                    .
                    .
END;

ANALYZE NTRAN;
                    .
                    .
                    .
                    INIT L(CMPOUT)=0 L(CNTY11)=1 L(CNTY12)=0 L(CNTY21)=1
                    L(CNTY22)=0 L(PHINRF)=0 L(PHIPRF)=1 L(L3.XSTEP1)=0
                    L(L3.XSTEP2)=0 L(L6.XSTEP1)=1 L(L6.XSTEP2)=1
.
.
END;
END;
```

As it is explained in Sec. 2.2.5, there are no restriction on the format of node and parameter field of SUBCKT definition (such as what goes in the first and second parentheses).

In order to initialize an internal node in a SUBCKT definition, it is necessary to concatenate its reference name to the internal node when INIT statement is specified. In the above example, when internal nodes L3 and L6 in SUBCKT "STEP" are called in XSTEP1 and XSTEP2, their full names become L3.XSTEP1 and L6.XSTEP1, and L3.XSTEP2 and

L6.XSTEP2, respectively. These names must be used in INIT statement.

Chapter 3

ADVANCED FEATURES

This chapter describes a number of advanced features of SWITCAP2 related to multiple clocks, resistance simulations, operational amplifier poles, transient analysis waveforms, and dc steady-state analyses for simulating clock feedthrough. Some subtle points regarding NTRAN are also explained.

3.1 Multiple Clocks and Multiple Samples

SWITCAP2 allows an unlimited number of clocks to be defined for each network and each clock may have an unlimited number of HIGH intervals. For example, suppose an SCN has a set of clock waveforms shown in Fig. 3.1. These clocks may be described by the statements:

```
TIMING;
    PERIOD 8e-6;
    CLOCK clk1 3/4 (0 1/4) (1/2 5/8);
    CLOCK clk2 2 (0 1/4) (2/4 4/4) (3/2 2);
    END;
```

The numbers in this example have been chosen to show most of the possible number expressions acceptable to SWITCAP2 in these statements. If a master clock period of 1 us had been chosen, the statements would have appeared a little neater. Note that clk2 has effectively only two HIGH intervals per period but requires three HIGH intervals to describe. The rule is that all clock times corresponding to the edges of the HIGH intervals must be

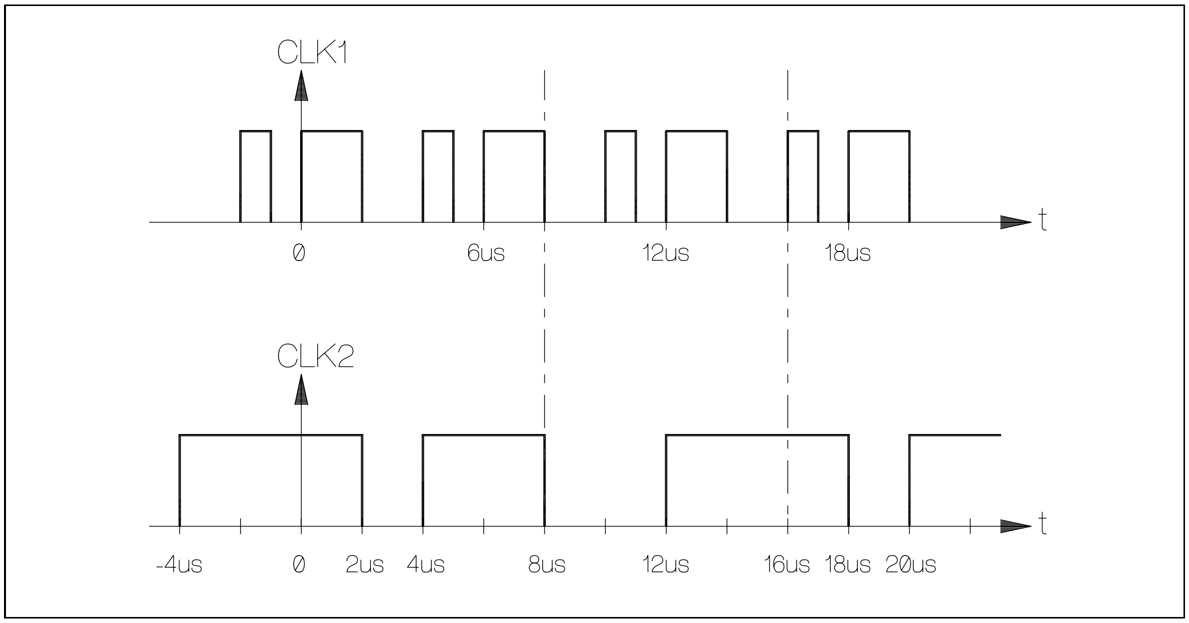

Figure 3.1: Multiple Clock Waveforms.

specified within the interval $0 \leq t \leq T$, where T is the period of the individual clock. Note also that the periods of the clocks need not be the same. SWITCAP2 will figure out what the effective switching period is (which is the least common multiple of all of the clock periods and is equal to 24 us in this example). Hence each clock need only be specified for as many HIGH intervals as there are before they start repeating.

Input and output sampling functions may have an unlimited number of sampling instants per sampling period. For example, suppose an input sample-and-hold circuit has sampling instants at the times just before $\ldots, 3\text{us}, 19\text{us}, \ldots$ and just after $\ldots, 8\text{us}, 24\text{us}, \ldots$ as shown in Fig. 3.2. Using a master clock period of 8 us, this sampling function can be described with the statement

6 SAMPLE INPUT HOLD 2 3/8- 1+;

For SAMPLE OUTPUT HOLD and SAMPLE OUTPUT IMPULSE statements, multiple sampling instants may be similarly specified. All sampling instants must be within the interval $0^- \leq t < T^-$, where T is the sampling period ($t = T^-$ not allowed).

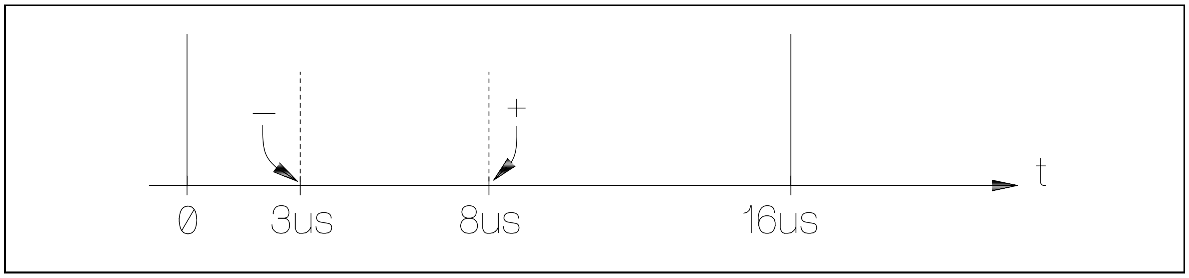

Figure 3.2: Multiple sampling instants per sampling period.

3.2 Frequency Overlap

In SSS, SPECTRUM, FREQCOMP, and SENS analyses, the output computed by SWITCAP2 is always the output of a bandpass filter as shown in Fig. 2.4. This has certain ramifications when the input frequency is an integral multiple of the Nyquist frequency of the system. At these frequencies, two separate output frequency components overlap at exactly the same output frequency. SWITCAP2 recognizes this condition and computes the vector sum of these two components. It indicates this condition by placing a letter “d” next to the frequency in the printout. Because of the summation, the responses at these frequencies will usually differ drastically from those at the nearby frequencies. This causes no problem in printing, but in plotting it often causes an extremely wide plot window to be selected as the plot routine tries to fit all points within a window. To avoid inappropriately selected windows, one can either adjust the input frequencies (via the INFREQ statement) so that the overlap frequencies are avoided, or specify a plot window explicitly in the PLOT statement (see Sec. 4.2.2).

It must be noted that the derivatives of the two frequency components are useless at these frequencies for computing group delay and sensitivity. Therefore, the values in group delay and sensitivity analysis at these frequencies (“d” is placed next to the frequency) must be ignored.

3.3 Time-Domain Waveforms

We have already shown the use of a PULSE waveform used in an ANALYZE TRAN block in Sec. 2.2.5. Here we give all of the possible time-domain waveforms in transient analysis. We repeat the PULSE waveform for convenient reference.

Function	Step, Pulse, or Pulse Train
Format	PULSE $< v1 >< v2 >< td >< tr >< tf >< pw >< per >$
Conditions	$0 \leq td, tr, tf, pw$ $0 < td + tr + tf + pw \leq per$

INTERVAL	**VALUE**
$t < t1$	v1 $t1 = td$
$t1 \leq t < t2$ $t2 = t1 + tr$	linearly interpolated between $(t1, v1)$ and $(t2, v2)$
$t2 \leq t < t3$ $t3 = t2 + pw$	$v2$
$t3 \leq t < t4$ $t4 = t3 + tf$	linearly interpolated between $(t3, v2)$ and $(t4, v1)$
$t4 \leq t < per$	$v1$
$per \leq t$	repeats the above waveform with period equal to $< per >$

Function	DC
Format	DC $< vdc >$

INTERVAL	**VALUE**
all t	vdc

Function	Cosine, Exponential, Exponential Cosine
Format	COSINE $< vo >< va >< f0 >< td >< h >< Q >$
Conditions	$0 \leq td$

INTERVAL	**VALUE**
$t < td$	vo
$td \leq t$	$vo + va \cdot exp(-(t - td)h)cos(2\pi((t - td) * f0 + Q/360))$

Function	Piece-Wise Linear
Format	PWL $< t[1] >< v[1] > ... < t[k] >< v[k] >$
Conditions	$0 \leq t[1] < t[2] < ... < t[k]$, $2 \leq k$
INTERVAL	**VALUE**
$t < t[1]$	$v[1]$
$t[j] <= t < t[j+1]$, $j = 1, ..., k-1$	linearly interpolated between $(t[j], v[j])$ and $(t[j+1], v[j+1])$
$t[k] \leq t$	$v[k]$

Examples:

```
SET Vin PULSE 2 -3 10e-6 20e-6 10e-6 50e-6 200e-6
SET Vin COSINE 0.5 10 3e3 10e-6 0 90;
SET Vin DC 5.5;
SET Vin PWL 0 0 1e-3 2 3e-3 2 5e-3 -1;
```

We remind the reader that in specifying time-domain voltage-source waveforms, all time values are expressed in *absolute* time units and not in mcp units.

3.4 Initial Values in TRAN for Linear SCN's

In TRAN, the specifications for time domain sources are designed so that the source values are well defined for *all* t, including $t < 0$. For $t < 0$, each source has a constant value. If these constant values for all sources are 0 for $t < 0$, then zero initial values are used for transient analysis. If any source has a non-zero value for $t < 0$, then the dc steady-state response corresponding to the non-zero input is first computed for $t < 0$ and this response at $t = 0^-$ is then used as the initial values for analysis. (The dc steady-state response is not necessarily a dc waveform since the network is time-varying.) If the dc steady-state response does not exist for an SCN, then all sources must have zero value for $t < 0$ in order for the program to be able to start the analysis.

3.5 Initial Values in NTRAN

In NTRAN, a set of the initial condition of analog and digital nodes at $t = 0^-$ may be specified through INIT statement. Unspecified nodes are set to 0 volt for analog nodes and logic '0' for digital nodes. SWITCAP2 makes an attempt to propagate the given initial condition to the unspecified nodes in the rest of the network in order to avoid inconsistent states. If the inconsistency persists, a warning is given.

Note that in TRAN, SWITCAP2 automatically determines the initial values for transient analysis by computing the dc steady-state response. However, in NTRAN, user defined initial values are used and the dc steady-state response is *not* computed.

3.6 Finite Resistance Simulation

Here we will show how finite resistors may be simulated with SWITCAP2. As an example, let us suppose that each of the switches in the circuit in Fig. 2.3 has a finite ON-resistance of 4 kΩ and an infinite OFF resistance. To simulate such a nonideal switch, we need only insert a 4 kΩ resistor in series with an ideal switch. It is known that a resistor in any active RC network can be replaced by an "equivalent" switched-capacitor as shown in Fig. 3.3. The exact sense of this equivalence is discussed elsewhere. [1] (Other equivalent SC configurations are also possible, but we will not discuss them here.) This particular equivalent SC network gives the approximate relation $R = T/(4C_{eq})$, where T is the period of the "resistor" switches and C_{eq} is the equivalent capacitance value. The approximation holds so long as the resistor switching frequency $f_r = 1/T$ is much higher than both the signal frequency and the switching frequency of the actual switches in the SCN (i.e., the frequency associated with CLK in Fig. 2.3). Let us use T = 8 ns, which makes the "resistor" switching frequency 1000 times higher than the "switch" switching frequency. For R = 4 kΩ, this gives us C_{eq} = 0.5 pF. (It is not always necessary to use such a high resistor switching frequency relative to the switch switching frequency. We are using a ratio of 1000 to 1 here just to illustrate the capability of the program. Sometimes, a ratio of even 10 to 1 gives adequate accuracy, depending on the circuit and the signal frequencies of interest.)

Now that we have finite resistances in the circuit, the capacitor values in Fig. 2.3 must be expressed in some definite units consistent with the units of resistance. Let us assume that the smallest capacitance value in Fig. 2.3 represents 1 pF. Then, with renormalization, the absolute capacitances are:

$$C1 = 1.00 \text{ pF}, C2 = 7.14 \text{ pF}, C3 = 28.57 \text{ pF}.$$

The circuit in Fig. 2.3 with finite switch ON-resistances can now be described with the statements

```
7           TIMING;
8                   PERIOD 8e-6;
```

[1] G. C. Temes, H. J. Orchard, and M. Jahanbegloo, "Switched-Capacitor Filter Design Using the Bilinear z-Transform," *IEEE Trans. Circuits and Systems,* CAS-25, pp. 1039-1044, Dec. 1978.

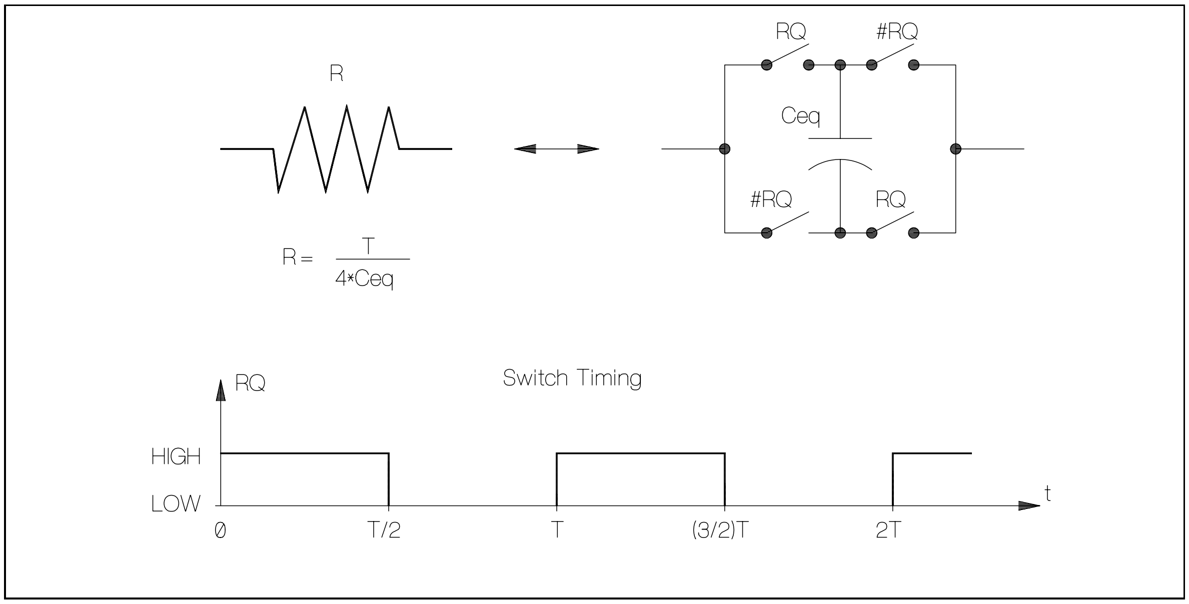

Figure 3.3: Switched-capacitor equivalent resistor.

```
        CLOCK clk 1 (0 3/8);
        CLOCK rq 1/1000 (0 1/2000);
        END;
SUBCKT (1 2) sw (K:ctl);
        Sma (5 2) ctl;
        S1a (1 3) rq;
        S1b (5 3) #rq;
        S2a (5 4) rq;
        S2b (1 4) #rq;
        Ceq (3 4) 0.5;
        END;
CIRCUIT;
```

```
21        Xs1 (1 2) sw ( clk);
22        Xs2 (2 3) sw (#clk);
23        C1 (2 0) 1.00;
24        C2 (1 3) 7.14;
25        C3 (3 4) 28.57;
26        E1 (4 0 0 3) 5500;
27        V1 (1 0);
28        END;
```

Here, with statements 13-18, we have placed an ideal switch in series with a 4 kΩ resistor in the subcircuit named sw. This subcircuit is then used as a nonideal switch in the main circuit given by statements 21-22. In subcircuit sw, the switch Sma controlled by the symbolic clock ctl represents the ideal switch of the switch/resistor combination. The switches S1a through S2b are part of the resistor-equivalent SC network and are controlled directly by the clocks rq and #rq. Note that it is not always necessary to use symbolic clocks in a subcircuit. If a switch in a subcircuit is always controlled by a certain clock, as it is the case here with all of the "resistor" switches, it may be more convenient to use actual clocks rather than symbolic clocks. This implies that clock names are always global parameter. Note also that we have dropped the pF units on all capacitance values. This is ok so long as we normalize for *all* capacitors, including the ones associated with resistors, by the same factor.

When more switching intervals (associated with either actual or "resistor" switches) are used, SWITCAP2 computation time will increase. The computation time, however, is also related to the number of intervals in which the input and output waveforms actually change value. When there are many switching intervals, significant reduction in computation can be obtained if the input and output waveforms are sampled only a few times per period. (Here we mean the complete period, which is the common multiple of all switch-controlling clock periods and sampling periods.) To sample-and-hold and to sample with impulse functions are equally effective. Minimum computation occurs when both input and output samplers are used, with both sampling once per period at 0+.

3.7 Operational Amplifier Poles

The frequency response of an operational amplifier (OA) can first be modeled with a combination of VCVS's and an RC network. The finite resistances in the RC network can then be simulated using the technique described in the previous section. Consider now an OA with a low frequency gain of 5500 and a dominant pole frequency of 200 Hz. This OA can be modeled by the circuit in Fig. 3.4. The single pole of the OA is effected by an RC network with R = 795.8 kΩ and C = Cp = 1 nF. To convert this circuit into an SC equivalent network, let us again use the resistor-equivalent SC network in Fig. 3.3 in place of R. Let T = 8 ns. For R = 795.8 kΩ, this gives Ceq = 2.513 fF. Hence the SC equivalent circuit of Fig. 3.4 can be described by the statements

```
29          SUBCKT (1 2 3 4) opamp (P:a0);
30              E1 (5 0 3 4) 1;
31              E2 (1 2 8 0) a0;
32              S1a (5 6) rq;
33              S1b (8 6) #rq;
34              S2a (8 7) rq;
35              S2b (5 7) #rq;
36              Ceq (6 7) 2.513e-3;
37              Cp (8 0) 1000;
38              END;
```

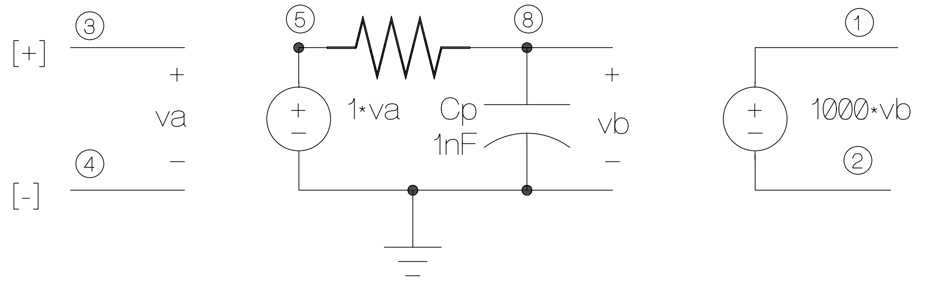

Figure 3.4: Finite-gain single-pole operational amplifier.

Here the clock rq is assumed to be the same as that defined by statements 7 in this chapter. This subcircuit can be used in place of ordinary VCVS's when the finite-bandwidth of the

VCVS must be taken into account. The nodes (1 2 3 4) are in the same order as the nodes of an VCVS, and the parameter a0 is the low-frequency gain of the OA. The transfer function of this OA implemented by statements 29-38 is given in Fig. 3.5. As the transfer function shows, this implementation is good for up to 10 MHz.

3.8 Clock Feedthrough

In real switches, parasitic capacitances often allow clock signals to feedthrough to the rest of network. If we were to simulate the clock waveforms using voltage sources with actual pulse waveforms, then in all frequency-domain analyses we (or SWITCAP2) would have to do a Fourier transform of either the input pulse waveforms or the output pulse waveforms. This Fourier transform computation can be avoided by using the dc steady-state analysis capability of SWITCAP2. As an example, consider an MOS transistor switch represented by the circuit shown in Fig. 3.6. Here we assume that the gate voltage of the MOS transistor varies between 0 V and 5 V. The gate voltage can be coupled into the rest of the network through the two parasitic capacitances Cgs and Cgd. To simulate the gate voltage waveform, a set of switches are used to switch the gate node alternately between ground and a dc voltage of 5 V. The branch Smain represents the conductance of the MOS transistor. This conductance will be modeled by the nonideal-switch subcircuit discussed in Sec 3.6. The circuit in Fig. 3.6, treated as a subcircuit, can be described by the statements

```
39        SUBCKT (s g d) mos (K:ctl);
40            Xsmain (s d) sw (ctl);
41            Ssg (g sg) ctl;
42            Ss0 (0 sg) #ctl;
43            Sdg (g dg) ctl;
44            Sd0 (0 dg) #ctl;
45            Cgs (s sg) 0.015;
46            Cgd (d dg) 0.015;
47            Csb (s 0) 0.2;
48            Cdb (d 0) 0.2;
49            END;
```

```
                                 (X): vp(4)
        80.0      100.0       120.0       140.0       160.0     180.0
        |-----------+-----------+-----------+-----------+-----------|

                                 (*): vdb(4)
  freq  -40.00 -20.00     0.00      20.00     40.00     60.00     80.00
1.00+01 |---------+---------+---------+---------+---------+------*X-|
2.15+01 |                                                       X*  |
4.64+01 |                                                   X    *  |
1.00+02 |                                           X            *  |
2.15+02 |                               X                       *   |
4.64+02 |                   X                                *      |
1.00+03 |            X                                    *         |
2.15+03 |        X                                     *            |
4.64+03 |      X                                    *               |
1.00+04 |      X                                *                   |
2.15+04 |     X                              *                      |
4.64+04 |     X                           *                         |
1.00+05 |     X                       *                             |
2.15+05 |     X                    *                                |
4.64+05 |     X                 *                                   |
1.00+06 |     X             *                                       |
2.15+06 |      X         *                                          |
4.64+06 |       X     *                                             |
1.00+07 |         =                                                 |
2.15+07 |      *       X                                            |
4.64+07 |   *                   X                                   |
1.00+08 |-*-------+---------+---------+------X--+---------+---------|
  freq  -40.00 -20.00     0.00      20.00     40.00     60.00     80.00
                                 (*): vdb(4)
```

Figure 3.5: Transfer function of finite-gain single-pole operational amplifier using equivalent SC network for finite resistance.

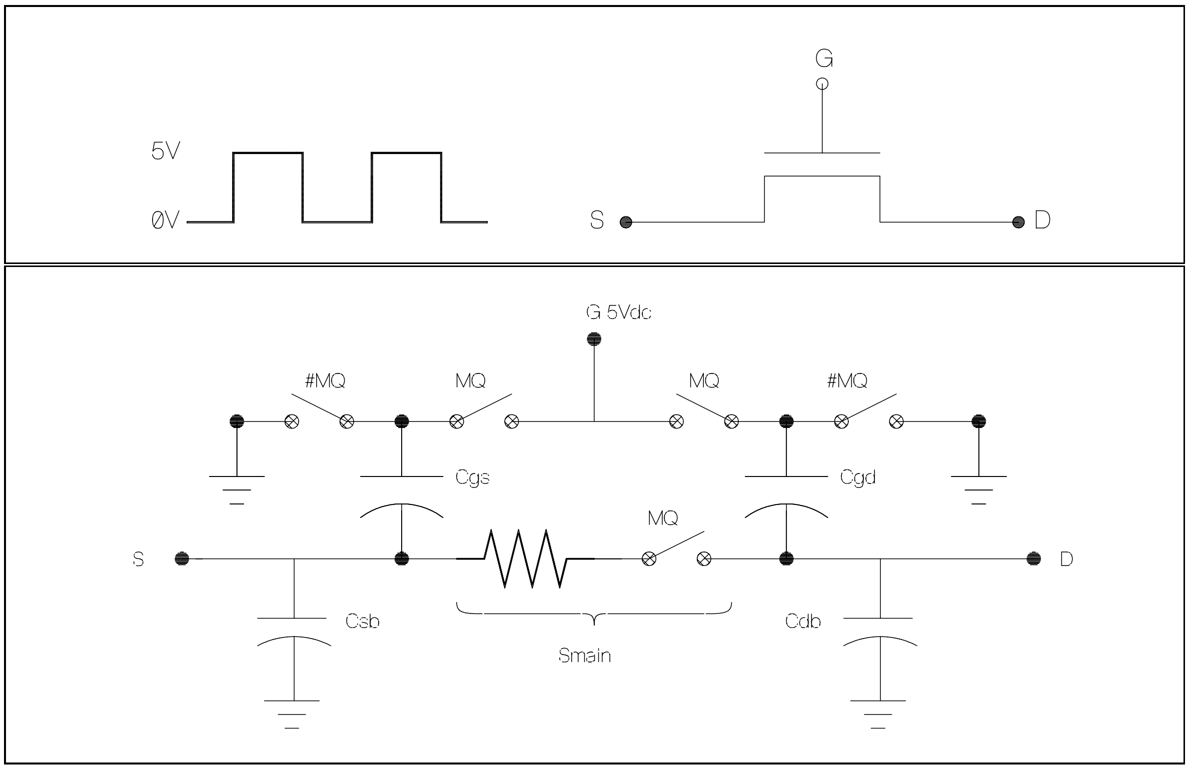

Figure 3.6: MOS Transistor Switch Model.

Here the subcircuit sw is assumed be to that given by statements 13-18. The capacitance values are assumed to be in units of pF (to be consistent with the capacitance values in subcircuit sw).

We will now analyze the spectrum of the clock feedthrough signal in the circuit in Fig. 2.3 incorporating the MOS transistor switch model in Fig. 3.6, and the finite-gain single-pole operational amplifier model in Fig. 3.4. We use the statements

```
50          CIRCUIT;
51                  Xs1 (1 g 2) mos ( clk);
52                  Xs2 (2 g 3) mos (#clk);
53                  C1 (2 0) 1.00;
54                  C2 (1 3) 7.14;
55                  C3 (3 4) 28.57;
56                  Xe1 (4 0 0 3) opamp (5500);
57                  V1 (1 0);
58                  Vg (g 0);
59                  END;
60          ANALYZE SPECTRUM;
61                  INFREQ 0 0 LIN 1;
62                  OUTFREQ 0 5e6 SCANNED;
63                  SET V1 AC 0 0;
64                  SET Vg AC 5 0;
65                  PLOT VDB(4);
66                  END;
```

The clock timing for this circuit is assumed to be defined by the TIMING block in statements 7-11 in this chapter. In the above statements 53-54, the subcircuit mos is assumed to be that given by statements 39-49 in page 65, the subcircuit opamp is assumed to be that given by statements 29-38 in page 64, and capacitance values are in pF. In statements 61-64, the input frequency is set to 0 to effect dc voltages, the source V1 is set to 0 V amplitude because we are not interested in the signal contributions from the input, and the source Vg is set to 5 V amplitude to create the top parts the pulse waveform of the MOS transistor gate voltage. Note that even though we are simulating dc input voltages, we must still use the AC type in the SET statements because this is required by SPECTRUM analysis. The dc nature of the waveform is controlled by the INFREQ statement.

To analyze the circuit in the time domain, we can use the statements:

```
ANALYZE TRAN;
        TIME 0+ 1 1/32;
        SET V1 DC 0;
        SET Vg DC 5;
        PLOT V(4);
        END;
```

As a final note, we observe that the technique we have used to create a square wave for the MOS transistor gate voltage can be used to create any piecewise-constant waveform simply by adding more dc voltage sources and more switching intervals. This technique can be used to advantage whenever we want to compute frequency-domain quantities for these types of waveforms.

3.9 Continuous Comparator

A continuous comparator, wherein its input terminals are monitored continuously and the output is allowed to change at the instant that the inputs cross over, is not available in SWITCAP2. However, if a continuous comparator is needed for some applications, it is possible to drive the enable terminal with a high-frequency clock as shown in Fig. 3.7, so that the latched-comparator behaves practically like a continuous one. This incurs a sacrifice in simulation speed. Remember that latched comparators are available *only* for NTRAN block.

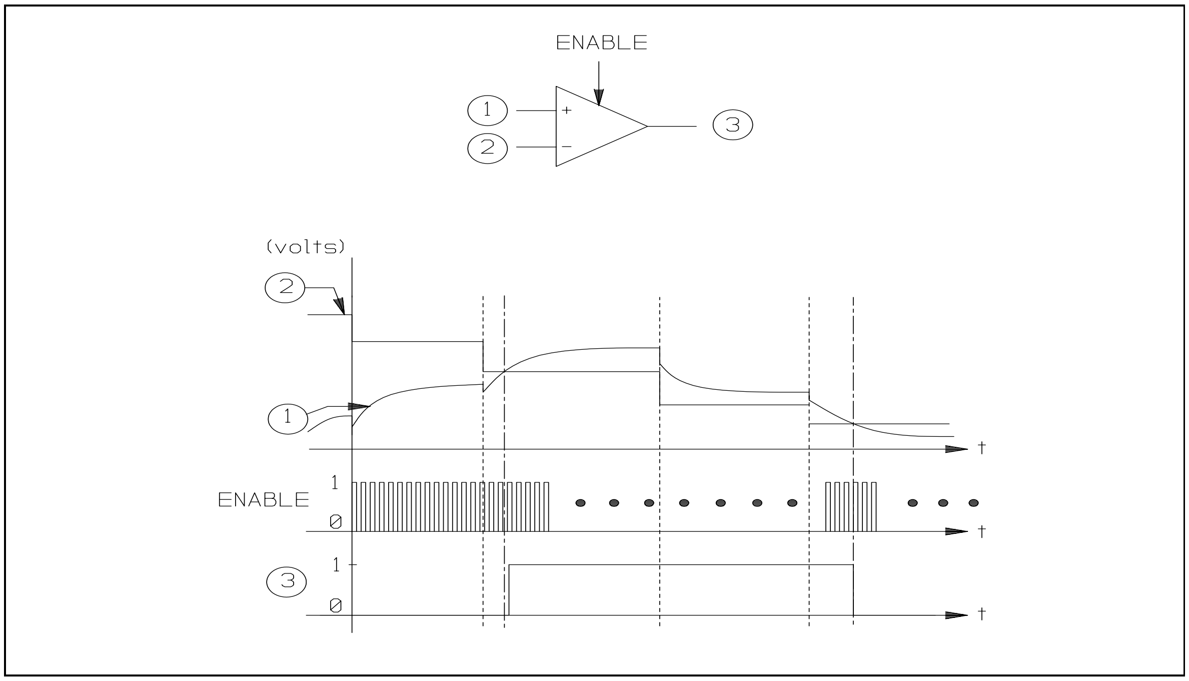

Figure 3.7: MOS Transistor Switch Model.

Chapter 4

OPTIONS AND OUTPUTS

4.1 Option Statements

In the input file to SWITCAP2 one can specify several options to control the format of the printout. For example, to suppress the echoing of the input file and to get a report of the execution times of the program, one can use the statements:

OPTIONS; NOLIST; REPORT; END;

The keywords "OPTIONS" and "END" define an OPTIONS block. The OPTIONS block must be written BEFORE the TIMING block in an input file. (It may be either before or after the TITLE statement.) The option keywords are listed below.

WIDTH132 Use 132 characters per line for output. Default is 80 (WIDTH80).

NOLIST Suppress printing of a listing of the input file.

REPORT Print a report of run times and other job statistics.

NOOVRLAY Default to NOOVRLAY for plotting when the OVRLAY/NOOVRLAY parameter in a PLOT statement is not specified. This default can be overridden by an explicit OVRLAY specification in each individual PLOT statement. Normal default is to OVRLAY. (See below for a description of the OVRLAY parameter in PLOT statements.)

GRID Default to GRID for superimposing grids on plots when the GRID/NOGRID parameter in a PLOT statement is not specified. This default can be overridden by an explicit NOGRID specification in each individual PLOT statement. Normal default is to NOGRID. (See below for a description of the GRID parameter in PLOT statement.)

NUMBER Default to NUMBER for plot annotation when the NUMBER/NONUMBER parameter in a PLOT statement is not specified. This default can be overridden by an explicit NONUMBER specification in each individual PLOT statement. Normal default is to NONUMBER. (See below for a description of the NUMBER parameter in a PLOT statement.)

SEN_ELEM Default to SEN_ELEM for print/plot annotation when the SEN_NODE/SEN_ELEM parameter in a PRINT/PLOT statement is not specified. This default can be overridden by an explicit SEN_NODE specification in each individual PRINT/PLOT statement. Normal default is SEN_NODE. (See Sec. 4.2.5 for a description of the SEN_ELEM parameter in a PRINT/PLOT statement.)

CHKCLK Check the clock timing for network switching, input sampling (if any), and output sampling (if any), and print out the actual clock periods used internally by the program for computation.

Each keyword is a separate statement and therefore should be terminated with a semicolon (;). The keywords may be written in any order.

4.2 PRINT and PLOT Statements

We have already covered the basics of PRINT and PLOT statements in Chapter 2. Here we will describe the other features associated with these commands.

4.2.1 Voltage Across Two Nodes

We have shown how the node voltage with respect to ground may be specified as an output variable. It is also possible to specify the voltage across two nodes as an output variable. For example, the statements

```
PRINT VM(4,1);

PLOT VR(a,0) VI(0,b);
```

would cause the magnitude part of the value v(4)-v(1) to be printed and cause VR(a) and -VI(b) to be plotted.

4.2.2 Plot Window

Normally, SWITCAP2 chooses the horizontal axis for a plot such that all computed values are contained in the plot. It is possible to override SWITCAP2's selection by an explicit request. For example, if it is desired that only the values of vdb(4) between 18 and 22 be plotted with 5 grid points, the following statement may be used:

PLOT VDB(4) (18 22 5);

The resulting axis would be:

```
                         (*): vdb(4)
18.00        19.00          20.00          21.00        22.00
|--------------+--------------+--------------+--------------|
```

When several output variables are specified in a PLOT statement, each variable may be individually windowed. Example:

PLOT VDB(4) (18 22 5) VP(4) (180 90 5);

4.2.3 Plot Numbering

SWITCAP2 can also print out a numerical value beside the point being plotted. This feature is enabled by inserting the keyword NUMBER after the output variable specification. For example, if the PLOT statement in statements 18 were

PLOT VDB(4) NUMBER;

the resulting plot would be:

```
                                 (*): vdb(4)
  freq  -20.00    0.00        20.00       40.00       60.00     80.00
1.00-02 |-----------+-----------+-----------+-----------+-(74.78)*--|
3.16-02 |                                                (74.54)*   |
1.00-01 |                                                (72.70)*   |
3.16-01 |                                            (66.21)*       |
1.00+00 |                                      (56.78)*             |
3.16+00 |                                (46.85)*                   |
1.00+01 |                          (36.85)*                         |
3.16+01 |                    (26.85)*                               |
1.00+02 |              (16.86)*                                     |
3.16+02 |         (6.91)*                                           |
 1.00+03 |  (-2.62)*                                                      |
 3.16+03 |     *(-9.55)                                                   |
 1.00+04 |    *(-11.72)                                                   |
 3.16+04 |    *(-12.01)                                                   |
 1.00+05 |----*(-12.04)---------+----------+----------+----------|
   freq  -20.00    0.00        20.00      40.00      60.00     80.00
                                        (*): vdb(4)
```

When several output variables are specified in a PLOT statement, each variable may be individually numbered in a manner similar to windowing. When there is not enough room in the plot to accommodate all numbers, priority is given in the order in which the variables are listed in the PLOT statement. When the NUMBER option and the plot window option are both specified, the two options may be specified in either order.

When the NUMBER option is not explicitly specified in a PLOT statement, the default is usually not to number. This default can be changed, however, by specifying the NUMBER option in an OPTIONS block. When this option is set by the sequence:

OPTIONS; NUMBER; END;

then all plots *will* be labeled with numbers by default unless otherwise specified. This default can then be overridden in a PLOT statement for each output variable by inserting the keyword NONUMBER in the place where NUMBER normally goes.

4.2.4 Plot Overlaying

Normally the output variables specified in a single PLOT statement are plotted together on the same graph. This is called overlaying. When plots are overlayed, a different symbol is used for each plot, starting with *, then X, and so on. There is no limit on how many plots may be overlayed on one graph, but there are only 8 distinct plot symbols. After 8 plots, the plot symbols simply repeat. When two or more points coincide at one location, an equal sign (=) is used as the plot symbol to indicate this condition.

At times one may wish to restrict plot overlaying. For example, one might want to ensure the appearance of all number labels by allotting only one plot per graph. One way to restrict overlaying of course is to use a separate PLOT statement for each graph. Since each PLOT statement forces a complete simulation, *it may be computationally inefficient to use many PLOT statements*. A better method is to use the NOOVRLAY option. For example, the statement

PLOT NOOVRLAY VDB(4) VP(4);

will force VDB(4) and and VP(4) to be plotted on separate graphs.

The relation of the NOOVRLAY option in a PLOT statement to the NOOVRLAY option in the OPTIONS block is similar to that of the NUMBER option. When the NOOVRLAY option is not explicitly specified in a PLOT statement, then normally there would be overlaying. But if the NOOVRLAY option is specified in the OPTIONS block, then the action is reversed: there would be NO overlaying unless an OVRLAY is explicitly declared in a PLOT statement.

For sensitivity analysis the actions of OVRLAY and NOOVRLAY options remain the same, but the default produces a completetly different result. The default is to group the data (according to nodes specified in the PLOT statement or the elements specified in the PERTURB statement) and then overlay the data in each group into one plot. (see Sec. 4.2.5)

NOTE: In order to obtain the default output, OVRLAY and NOOVRLAY must *not* appear in the PLOT statement or in the OPTIONS block.

For SPECTRUM analysis the plots cannot be overlayed.

4.2.5 Print/Plot options for Sensitivity

For the sensitivity analysis, there is a special print/plot option which controls the order in which the output is printed or plotted. For example, if the sensitivity of VR(4) and VI(4) with respect to C1, C2 and E1 is requested, there will be six sets of output data. (VR(4)/C1, VR(4)/C2, ...). The following options are used to group these sets.

SEN_NODE

This option is used to group the sets of output data according to the nodes specified in the PRINT or PLOT statement. For example, the statements

PERTURB C1, C2, E1;

PRINT SEN_NODE VR(4) VI(4);

will result in the following grouping of the output.

```
      frequency        vr(4)/E1    vr(4)/C2    vr(4)/C1
          .                .           .           .
          .                .           .           .
----------------------------------------------------------------------
      frequency        vi(4)/E1    vi(4)/C2    vi(4)/C1
          .                .           .           .
          .                .           .           .
```

SEN_ELEM

This option is used to group the sets of output data according to the elements specified in the PERTURB statement. For example, the statements

PERTURB C1, C2, E1;

PRINT SEN_ELEM VR(4) VI(4);

will result in the following grouping of the output.

```
        frequency        vr(4)/E1    vi(4)/E1
              .              .            .
              .              .            .
-----------------------------------------------------------------------
        frequency        vr(4)/C2    vi(4)/C2
              .              .            .
              .              .            .
-----------------------------------------------------------------------
        frequency        vr(4)/C1    vi(4)/C1
              .              .            .
              .              .            .
```

When using the options SEN_NODE and SEN_ELEM within the PLOT statement, the output data is grouped in the same manner except that the output data is plotted. SEN_NODE option creates a separate plot for each entry in the PLOT statement and in each plot it overlays the data for all of the elements specified in the PERTURB statement. SEN_ELEM does the opposite.

The default for both PRINT and PLOT statements is SEN_NODE. This default may be chaged by specifying SEN_ELEM in the OPTIONS block.

NOTE: When using the PLOT statement, in order to completely separate all of the output data into separate plots, NOOVRLAY option *must* be specified either in the PLOT statement or in the OPTIONS block. Similarly, the OVRLAY option must be specified in order to completely overlay all of the output data.

4.2.6 Grid Printing

To superimpose a set of grid lines within a plot, the GRID option can be used. For example, if the PLOT statement in 18 were

PLOT GRID VDB(4) VP(4);

the resulting plot would be:

```
                                        (X): vp(4)
             90.0     105.0      120.0      135.0      150.0      165.0   180.0
```

```
        |---------+---------+---------+---------+---------+---------|


                                 (*): vdb(4)
  freq  -20.00    0.00        20.00       40.00       60.00     80.00
1.00-02 |-----------+-----------+-----------+-----------+--------=--|
3.16-02 |           .           .           .           .  X     *  |
1.00-01 |           .           .         X .           .       *   |
3.16-01 |           .  X        .           .           .   *       |
1.00+00 |    X      .           .           .         * .           |
3.16+00 | X         .           .           .   *       .           |
1.00+01 |X. . . . . . . . . . . . . . . . * . . . . . . . . . . . . |
3.16+01 |X          .           .   *       .           .           |
1.00+02 |X          .         * .           .           .           |
3.16+02 |   X       .   *       .           .           .           |
1.00+03 |         * .X          .           .           .           |
3.16+03 |     *     .           .       X   .           .           |
1.00+04 | . .*. . . . . . . . . . . . . . . . . . . . . . X . . . . |
3.16+04 |    *      .           .           .           .        X  |
1.00+05 |----*------+-----------+-----------+-----------+----------X|
  freq  -20.00    0.00        20.00       40.00       60.00     80.00
                                 (*): vdb(4)
```

The GRID option works in a similar manner as the OVRLAY option. If the GRID option is used along with the NOOVRLAY option, then all plots generated by the NOOVRLAY option will have grids. The default to use when the GRID option is omitted from a PLOT statement can be set by the GRID option in the OPTIONS block. An explicit NOGRID specification in a PLOT statement overrides the default.

4.2.7 Summary of Plot Options

The general format of a plot statement is:

PLOT {OVRLAY} {NOOVRLAY} {GRID} {NOGRID} {SEN_NODE} {SEN_ELEM }

Vxx(<n1+> {,<n1->}) {NUMBER} {NONUMBER}

{(<ylim1> <ylim2> <ngrids>)}

{Vxx(<n2+> {,<n2->}) {NUMBER} {NONUMBER}

{(<ylim1> <ylim2> <ngrids>)} ...}

where items in curly brackets are optional.

4.3 Plot Axis Labeling

When the x- and y-axes of a plot are to be labeled with numbers whose most significant digits are the same, SWITCAP2 uses an abbreviated notation to convey information in a limited amount of space. Example:

```
7269x.xxx                  10**-03                  7270x.xxx
$6.000   $8.000      $0.000      $2.000      $4.000    $6.000
|-----------+-----------+-----------+-----------+-----------|
```

The actual value at the left is

72696.000*10**-3 = 72.696

The actual value at the right is

72706.000*10**-3 = 72.706

Other values on the axis can be interpreted in a similar manner. Abbreviations of numbers on the x-axis (vertical axis) follows a similar format.

4.4 Graphic Interface to NUTMEG

The documentation for this section will be available soon.

Chapter 5

REFERENCE MANUAL

This chapter summerizes the commands and keywords used in SWITCAP2 briefly and points out the sections describing more details for each command. This should be used by the one who is already familiar with SWITCAP2. For first-time users, it is recommended to read Chapter 2.

OPTIONS Its general syntax is:

OPTIONS; $<$ *keyword* $>$; ...; END;

where $<$ *keyword* $>$ can be any of the following:

WIDTH132

WIDTH80

NOLIST

LIST

REPORT

NOOVRLAY

OVRLAY

GRID

NOGRID

NUMBER

NONUMBER

SEN_NODE

SEN_ELEM

These keywords are separated by ";". For more detailed explanation, see Chap. 4.

TITLE Its general syntax is:

> TITLE: < *any alphanumeric characters* >

See Sec. 2.2.4 for more details (page 22).

TIMING Its general syntax is:

```
TIMING;
.
.
END;
```

This block should only contain one PERIOD statement and one or more CLOCK statements. TIMING block must be defined before SUBCKT, CIRCUIT, and ANALYZE blocks.

PERIOD Its syntax is:

> PERIOD *value*

where *value* is in second. (See page 15)

CLOCK Its general syntax is:

> CLOCK *clockname period* (ON1 OFF1) (ON2 OFF2) $\cdots$ (ONi OFFi) $\cdots$;

where *clockname* is a string of alphanumeric characters (at most 78 characters long), *period* is an integer or a rational number which specifies its period in MCP (master clock period), and ONi and OFFi specify ith on-interval. See page 15 for CLOCK statement and Sec. 3.1 for multiple number of HIGH intervals.

SUBCKT Its syntax is:

> SUBCKT ($\cdots$) *subcircuitname* ($\cdots$);

where *subcircuitname* is a string of alphanumeric characters (at most 78 characters long). The first and second parentheses should contain analog and digital "terminal" nodes, symbolic clocks, and symbolic values of capacitors and voltage controlled voltage sources. Although it is necessary to keep the two parentheses around *subcircuitname*, there is no restriction on the order or kind of their contents. See Sec. 2.2.5 and 2.3.7. Several examples can also be found in sections 3.6, 3.7, and 3.8.

CIRCUIT Its syntax:

> CIRCUIT;
> ...
> ...
> ...
> END;

This block specifies network topology. It may contain capacitors, switches, VCVS's, independent voltage sources, latched comparators, logic gates, and subcircuit calls. Latched comparators and logic gates are applicable only for ANALYZE NTRAN block.

CAPACITORS Its syntax is:

> C*name* ($n1$ $n2$) *value*

where *name* is a string of alphanumeric characters (at most 78 characters long), $n1$ and $n2$ are the analog nodes to which the capacitor is connected, and *value* is the capacitance in Farads. See Sec. 2.2.2 for more details.

SWITCHES Its syntax is:

> S*name* ($n1$ $n2$) *control*

where *name* is a string of alphanumeric characters (at most 78 characters long), $n1$ and $n2$ are the analog nodes to which the switch is connected, and *control* is the switch control signal that is generated by a clock, an output of a latched-comparator, or an output of a logic gate. See Sec. 2.2.2 and Sec. 2.3.4 for more details.

VOLTAGE-CONTROLLED VOLTAGE SOURCES Its syntax is:

> E*name* ($n+$ $n-$ $nc+$ $nc-$) *gain*

where *name* is a string of alphanumeric characters (at most 78 characters long), $n+$ and $n-$ are the positive and negative analog nodes, respectively, to which the VCVS is connected, $nc+$ and $nc-$ are the positive and negative controlling nodes, respectively, and *gain* is the voltage gain. See Sec. 2.2.2 for more details.

INDEPENDENT SOURCES Its syntax is:

> V*name* ($n+$ $n-$)

where *name* is a string of alphanumeric characters (at most 78 characters long), and $n+$ and $n-$ are the positive and negative analog nodes, respectively, to which the source is connected. Its source type and value are subsequently defined in ANALYZE block. See Sec. 2.2.2 for more details.

LATCHED-COMPARATORS Its syntax is:

> Y*name* CMPLAT ($in+$ $in-$ *enable out*)

where *name* is a string of alphanumeric characters (at most 78 characters long), $n+$ and $n-$ are the positive and negative analog nodes, respectively, to which the latched-comparator is connected, *enable* is the latch enable terminal (a periodic clock waveform directly from TIMING block must be used to drive this terminal) and *out* is the logic output. See Sec. 2.3.4 for more details.

LOGIC GATES Its syntax is:

> Y*name* $<$ *gatetype* $>$ (*in*1 *in*2 $\cdots$ *inN out*)

where *name* is a string of alphanumeric characters (at most 78 characters long) and *gatetype* may be AND, OR, NAND, NOR, XOR, XNOR, or NOT. *in*1, *in*2, $\cdots$, and *inN* are input nodes. NOT can have only one input. Input and output of a logic gate should have logic values ("0" or "1"), not analog quantities. See Sec. 2.3.3 for more details.

SUBCIRCUIT CALLS Its syntax is:

> X*name* ($\cdots$) *subcktname* ($\cdots$)

where *name* is a string of alphanumeric characters (at most 78 characters long) and *subcktname* is the referenced subcircuit name (this name is defined in SUBCKT definition block). Analog and digital terminal nodes and parameters must match the order that was specified in SUBCKT definition. See Sec. 2.2.5 in page 35 for more details.

ANALYZE Its syntax is:

> ANALYZE *nnn*
> ...
> ...
> ...
> END;

where *nnn* may be any of the following:

SSS See page 18 in Sec. 2.2.3.

SPECTRUM See page 27 in Sec. 2.2.5.

FREQCOMP See page 30 in Sec. 2.2.5.

TRAN See page 33 in Sec. 2.2.5.

SENS See page 38 in Sec. 2.2.6.

NTRAN See page 52 in Sec. 2.3.5.

SET Its syntax is:

SET V*sourcename sourcetype* $<$ *parameters* $>$

where $source_name$ is the name of the independent source that is being specified within an ANALYZE block, *sourcetype* is the type of source (AC, DC, PULSE, COSINE, or PWL). See sections 2.2.3, 2.2.5, 2.2.5, and 3.3 for more details.

TIME TIME $tstart< +/- >$ *tend tstep*;

where *tstart* in the initial time for printing, *tend* is the final time for printing, and *tstep* is the print time step. The plus (+/-) sign appended to the *tstart* indicates the program to print out the values at the time instants just (after/before) the time specified in *tstep*. Note that transient analysis starts at $t = 0^-$ regardless of the above setting. See Sec. 2.2.5 for more details.

INFREQ INFREQ *fstart fstop* LIN/LOG N;

where *fstart* is the starting frequency, *fstop* is the final frequency, LIN/LOG is an option for linear and logarithmic sweep, respectively, and N is the number of points. See sections 2.2.3 and 2.2.5 for more details.

OUTFREQ OUTFREQ *fstart fstop* SCANNED;

where *fstart* is the starting frequency and *fstop* is the final frequency. See sections 2.2.3 and 2.2.5 for more details.

INIT INIT V(*analognode*) $\cdots$ L(*logicnode*) $\cdots$

INIT statement is only allowed in NTRAN analysis. Only one INIT statement is allowed in an ANALYZE NTRAN block. However, multiple lines can be used if there are many nodes to be initialized.

PRINT See Chap. 4 for more details.

PLOT See Chap. 4 for more details.

NPLOT See Chap. 4 for more details.

XPLOT Not yet supported in this version.