A TOP-DOWN SYNTHESIS METHODOLOGY FOR BEHAVIORAL MIXED-SIGNAL SYSTEMS SPECIFIED IN VHDL-AMS

Adrián Núñez-Aldana, Alexa Doboli and Ranga Vemuri
Digital Design Environments Laboratory
Department of ECECS, University of Cincinnati
Cincinnati, OH 45221–0030

Abstract

This paper presents a top-down design methodology to synthesize mixed-signal systems described at behavioral level. Our methodology is intended to automate the synthesis process of analog CMOS integrated circuits. Also, the method enables the circuit description at high level of abstraction where complex systems are easier to specify. The behavioral models are represented by a set of simultaneous differential and algebraic equations (DAEs), which are written in VHDL-AMS using simple simultaneous and procedural statements. We present an algorithm to perform the topology selection process. As an example, a Dual-Tone Multiple-Frequency decoder is synthesized from behavioral model to analog CMOS transistor level.

1 Introduction

The emerging analog hardware description languages will speed up the analog verification process and enable the analog designer to describe circuits at higher levels of abstraction. Analog integrated circuits have been typically designed at circuit, component and transistor levels, where the low level circuit intrinsic characteristics are always taken into account. Working at low levels of abstraction usually leads to long design process, which may take, even for small analog cells, several months of man-work. With the use of behavioral models, instead of physical characterization, a much faster evaluation can be achieved at early stages of the design process. Besides, reducing the design complexity by using higher levels of abstractions, analog designers can describe applications at the system level. VHDL-AMS [1] is a hardware description languages which can specify analog and mixed-signal systems at different levels of abstraction.

In the research area, there are many CAD tools to synthesize analog circuits. However, most of the synthesis work has been limited to design individual components such as operational amplifiers [2]. Some of the tools have been focus in the cell block selection process, and others in the circuit sizing process. OASYS [3] is a hierarchically structured synthesis tool, which can generate a component level circuit topology from a structural description by making use of analog circuit design knowledge. ASTRX/OBLX [4] can size a given circuit topology to meet a set of given specifications. Some effort has been done to synthesize specific designs from the system level description [5]. Recently, some methods have been proposed to synthesize generic mixed-signal systems described at behavioral level. VASE [6] is a system environment proposed at the University of Cincinnati to perform high level synthesis of mixed-signal systems. Similarly, KANDIS [7] is a tool, which supports system-level design of mixed-signal systems.

In this paper, we present a design methodology to improve the mixed-signal design process. We propose a technique to automate the synthesis process of analog CMOS integrated circuits which are described at behavioral level. Using VHDL-AMS as a system specification languages, the analog designer can work at higher levels of abstraction and explore more complex designs. The organization of this paper is as follows: Section 2 presents an overview of VHDL-AMS. Section 3 presents the synthesis methodology of analog systems. In section 4, a design example, a Dual Tone Multiple Frequency (DTMF) decoder is specified in VHDL-AMS and synthesized. Finally, the future work and conclusions are presented in section 5.

2 VHDL-AMS

VHDL-AMS [1] is a hardware description language for the specification and simulation of analog and

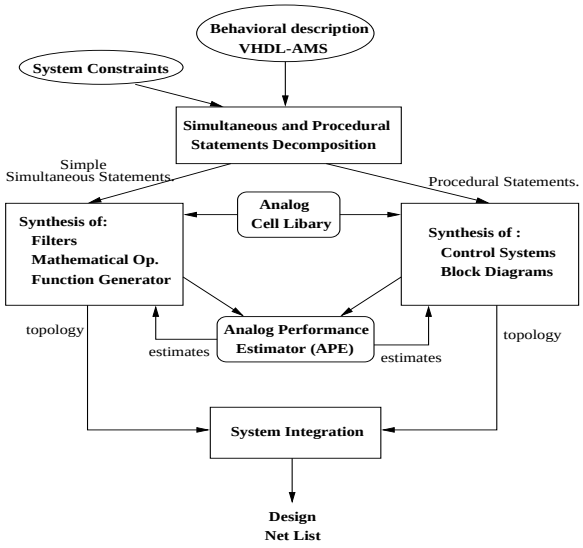

Figure 1: Design Methodology

mixed-signal systems. It is an extension of VHDL, a HDL for digital systems. VHDL-AMS describes continuous systems using a set of simultaneous differential and algebraic equations (DAEs), where time is the independent variable and quantities are the variables to solve. The simulation process uses an analog solver, which determines the value of all quantities (unknowns) over time such that the DAEs are satisfied. In electrical analog systems, the quantities can represent interface elements, voltages, or currents. VHDL-AMS permits two ways of describing simultaneous DAEs: simple simultaneous statements and simultaneous procedural statements. Simple simultaneous statements directly implement the mathematical equations, whereas, simultaneous procedural statements use algorithmic representations to formulate them. Simple simultaneous statements, simultaneous procedural statements, or a combination of both can specify analog systems.

3 Analog synthesis methodology

In general, an analog circuit synthesis process consists of two steps: topology selection and circuit sizing. Topology selection is the process of choosing an architecture that performs the desired behavior. The circuit-sizing phase determines the physical dimensions, bias points, and element values to meet user constraints. In this section, we present an algorithm to perform the topology selection process, which is leaded by an analog performance estimator such that the sizing process can meet the system requirements. The following paragraphs describe the design our system methodology, the VHDL-AMS subset used, and the synthesis algorithm.

Design Methodology: Figure 1 shows the block diagram of the synthesis process presented in this paper. The input system specification is a VHDL-AMS behavioral model. At the first stage, the simple simultaneous and procedural statements are decomposed. The simple simultaneous statements are mapped to filter, or function generator implementations; whereas, the simultaneous procedural statements are mapped to interconnected blocks topology. The analog performance estimator, APE [9], is used to size and evaluate each analog component selected from the library according to the fabrication process parameters and the given constraints. The estimates lead the component selection process. Finally, the topologies are integrated to produce a net-list.

VHDL-AMS subset: Several constraints involved in the synthesis of the behavioral descriptions lead us to restrict the specification to a subset of VHDL-AMS. A behavioral model is a mathematical representation of a system; however, it could portray more than one physical system. There is no one-to-one mapping between behavioral models and system implementations. Therefore, our synthesis process requires, from the designer, extra annotations, which lead to the desired implementation. Although the DAEs can be described with either simple simultaneous or procedural statements, in general, it is easier to describe transfer function implementations using simple simultaneous statements, and to describe signal flow using procedural statements. Therefore, we restrict the description of circuits such as filters, function generators and mathematical operations to simple simultaneous statements, and circuits such as control systems and block interconnections to procedural statements. Another problem created by using behavioral models is the implicit input/output declaration. An analog electrical model doesn't necessarily have to define when a signal is used as an input or as an output. However, a synthesis process requires explicitly to know a signal flow. So, we restrict the description to an explicit input/output declaration and a signal flow. In the procedural statements, we allow simple assignments and *if*-statements where the condition is the comparison of two quantities.

Analog synthesis algorithm: Figure 2 presents our mixed-signal synthesis algorithm (Missa). The procedure starts by decomposing the simultaneous and the procedural statements. Each set of statements is synthesized separately. For the set of simultaneous equations, *SIM_ST*, each equation is synthesized separately. First, the order of the differential algebraic equation is detected. Then, the required

number of integrators and differentiators are allocated. Note that if a n+1th order differentiator or integrator is required; the algorithm allocates a first order differentiator or integrator and makes use of the previous nth order component. Once all integrators and differentiators for the given equations are allocated, an adder/subtracter with the required number of inputs is generated, and the equation topology is produced. An array of equation topologies, *eq_top*, is generated after processing all equations. On the other hand, for the set of procedural statements, *PROC_ST*, a comparator is allocated for each *if*-statement, and a library component is allocated for each assignment. The topology for the procedural statement, *proc_top*, is generated using the comparator and block sets. The differentiator, integrator, adder/subtracter, and comparator sets are collapsed such that two components performing the same function and working with the same input signals become one. Finally, the design net-list is generated.

Analog component allocation procedure: Figure 3 presents the algorithm to perform component allocation. First, a set containing all components, which perform the specified function is generated from the analog cell library [8]. A selection process is done by repeatedly picking components and evaluating the performance, using APE, until the selected component meets the given constraints.

4 Synthesis of a DTMF decoder

A Dual-Tone Multiple-Frequency decoder (DTMF) [11, 12] is used as an example to demonstrate the design methodology. The DTMF is modeled by simple simultaneous and procedural statements, and it can be interfaced to digital systems, making it suitable as a mixed-signal system. The following paragraphs delineate the system specifications, how the system is modeled using simple simultaneous and procedural statements, the VHDL-AMS code and the simulation results.

Specifications: Figure 4 shows the block diagram of a DTMF presented in [10]. The first block is a pre-filter, which removes the unwanted frequency components; these include 60Hz line interface and dial tone. The band separators split the frequency tones into a low group 697hz-941hz, and high group 1209hz-1633hz. Then, a bank of band-pass filters is used to detect the presence of a tone. The amplitude detectors recognize the presence of a tone. And, the final stage converts the signal detection to a digital output.

```
procedure Missa()
begin
  spec ← read(VHDL-AMS);
  SIM_ST ← get_simultaneous_statements(spec);
  PROC_ST ← get_procedural_statements(spec);
  j ← 1;
  for each eq ∈ SIM_ST do
    der_order ← highest_diff_order_of(eq);
    int_order ← highest_integral_order_of(eq);
    op_DER[j] ← {}; /* differentiator set */
    op_INT[j] ← {}; /* integrator set*/
    for i:=1 to i≤der_order do
      op_DER[j] ← op_DER[j] ∪ allocate(diff);
    end for;
    for i:=1 to i≤int_order do
      op_INT[j] ← op_INT[j] ∪ allocate(integrator);
    end for;
    op_ADD[j] ← allocate(adder/subtracter);
    eq_top[j] ← do_eq(op_DER[j],op_INT[j],op_ADD[j]);
    j++;
  end for;
  op_COMP ← {}; /* comparator set */
  op_BLOCK ← {}; /* block component set */
  while head(PROC_ST) != {} do
    st ← head(PROC_ST);
    PROC_ST ← tail(PROC_ST);
    if (st == if-statement)
      op_COMP ← op_COMP ∪ allocate(comparator);
    else
      op_BLOCK ← op_BLOCK ∪ allocate(st.block);
    end if-else;
  end while;
  proc_top ← do_proc(op_COMP,op_BLOCK);
  collapse(op_DER[]);
  collapse(op_INT[]);
  collapse(op_ADD[]);
  collapse(op_COMP[]);
  netlist ← do_netlist(eq_top[],proc_top);
  performance ← APE(netlist);
  print performance;
  return netlist;
end;
```

Figure 2: Missa: Mixed-signal synthesis algorithm

```
procedure allocate(function)
begin
  c_set = { x ∈ library | x.function==function };
  repeat
    if c_set == { } then
      print "Design can not meet user constraints";
      exit();
    end if;
    pick G ∈ c_set;
    c_set ← c_set - { G };
    topology ← G.topology;
    performance ← APE(topology);
  until performance meets user_constraints;
  return topology;
end;
```

Figure 3: Analog component allocation procedure

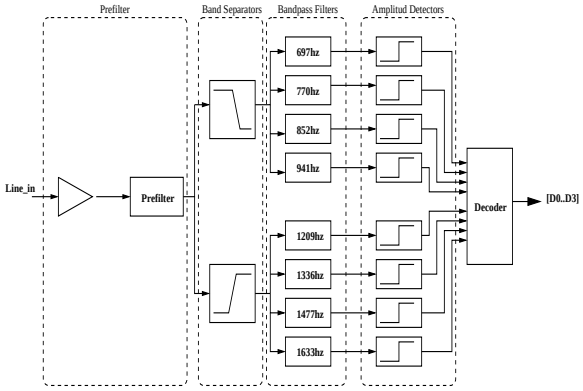

Figure 4: Dual Tone Multiple Frequency (DTMF)

Modeling: We can identify two types of circuit descriptions in this example, the filters and the signal flow. The filters are easily described by transfer functions, which can be specified by DAEs in VHDL-AMS. The signal flow is better represented as block-interconnection, where procedural statements of VHDL-AMS can be used. For example, the band pass filters required to detect the tones have the following DAE:

$$Q_{Fout} == a_1 \frac{d'}{dt} Q_{Fin} - a_2 \frac{d''}{dt} Q_{Fout} - a_3 \frac{d'}{dt} Q_{Fout}$$

Which maps directly to a DAE in the simple simultaneous statements format. The frequency tone detection is described using a procedural statement with *if*-statements and a threshold voltage. The output is a boolean signal indicating the presence of a tone, which can be decoded and used in a digital system.

VHDL-AMS code: The following is the VHDL-AMS code, which detects the low-band frequency tones, 697hz, 770hz, 852hz, and 941hz (for simplicity, the decodification of the tones is not shown):

```
ENTITY DTMF IS  PORT (
  TERMINAL Line: electrical;
  SIGNAL D_output: OUT bit_vector(3 downto 0);
  SIGNAL clk : IN bit);
END ENTITY DTMF;

ARCHITECTURE behavior OF DTMF IS
QUANTITY V_line ACROSS Line TO ground; -- Input Line
QUANTITY Q_prefilter : real;    -- Prefilter output
QUANTITY Q_BPLow,Q_BPHigh : real;
QUANTITY Q_F697,Q_F770,Q_F770,Q_F852,Q_F941 : real;
BEGIN
-- Prefilter: Bandpass filter: Butterworth 4-order
Q_pref == 3.55e-7*(V_line'dot'dot)
     - 6.416e-16*Q_pref'dot'dot'dot'dot
     - 2.14e-11*Q_pref'dot'dot'dot
     - 4.077e-7*Q_pref'dot'dot
     - 8.451e-4*Q_pref'dot;
-- Band Separator Low: Chebyshev 6-order
Q_BPLow == Q_pref
     - 5.19e-22*Q_BPLow'dot'dot'dot'dot'dot'dot
     - 2.42e-18*Q_BPLow'dot'dot'dot'dot'dot
     - 3.22e-14*Q_BPLow'dot'dot'dot'dot
     - 1.1e-10*Q__BPLow'dot'dot'dot
     - 4.97e-7*Q_BPLow'dot'dot
     - 9.97e-4*Q_BPLow'dot;
-- Band Separator High: Chebyshev 6-order
Q_BPHigh == 2.4e-25*Q_pref'dot'dot'dot'dot'dot'dot
      - 2.4e-25*Q_BPHigh'dot'dot'dot'dot'dot'dot
      - 1.055e-20*Q_BPHigh'dot'dot'dot'dot'dot
      - 2.361e-16*Q_BPHigh'dot'dot'dot'dot
      - 2.348e-12*Q_BPHigh'dot'dot'dot
      - 3.084e-8*Q_BPHigh'dot'dot
      - 1.037e-4*Q_BPHigh'dot;
-- Bandpass Filter: 697hz, Chebyshev 2-order
Q_F697 == 1.167e-5*Q_BPLow'dot
      - 5.214e-8*Q_F697'dot'dot
      - 1.167e-5*Q_F697'dot;
-- Bandpass Filter: 770hz, Chebyshev 2-order
Q_F770 == 2.067e-4*Q_BPLow'dot
      - 4.272e-8*Q_F770'dot'dot
      - 7.995e-6*Q_F770'dot;
-- Bandpass Filter: 852hz, Chebyshev 2-order
Q_F852 == 5.671e-6*Q_BPLow'dot
      - 3.489e-8*Q_F852'dot'dot
      - 5.671e-6*Q_F852'dot;
-- Bandpass Filter: 941hz, Chebyshev 2-order
Q_F941 == 5.008e-6*Q_BPLow'dot
      - 2.861e-8*Q_F941'dot'dot
      - 5.008e-6*Q_F941'dot;

PROCEDURAL IS
CONSTANT Vth : real := 0.5;
VARIABLE v697,V770,V852,V941 : boolean := false;
BEGIN
   IF (max(Q_F697) > Vth) THEN
     v697 := true;
   END IF;
   IF (max(Q_F770) > Vth) THEN
     v770 := true;
   END IF;
   IF (max(Q_F852) > Vth) THEN
     v852 := true;
   END IF;
   IF (max(Q_F941) > Vth) THEN
     v941 := true;
   END IF;
END PROCEDURAL;
END ARCHITECTURE behavior;
```

Results: First, we simulated the VHDL-AMS code. Since there aren't any VHDL-AMS simulators at present, we used the behavioral modeling option of HSPICE [13] (HSPICE permits to describe transfer functions using the Laplace transformation's). We manually transformed the VHDL-AMS code into behavioral HSPICE and simulated each filter. Table 1 shows the HSPICE simulation results for each filter.

A circuit topology was generated using the design methodology presented in the previous section. Figure 5 shows the topology generated for the band-pass

circuit	gain	passband (-3dB)	stopband (-10dB)
Pref	1	(251hz, 3.98Khz)	(0,150hz),(3.98Khz,∞)
Bplow	1	(0,966hz)	(1.01Khz,∞)
Bphigh	1	(1.18Khz,∞)	(0,1.12Khz)
F697	1	(679hz,716hz)	(0,645hz),(754hz,∞)
F770	1	(750hz,791hz)	(0,719hz),(824hz,∞)
F852	1	(838hz,863hz)	(0,813hz),(893hz,∞)
F941	1	(927hz,954hz)	(0,899hz),(984hz,∞)

Table 1: Behavioral simulation results

circuit	gain	passband (-3dB)	stopband (-10dB)
Pref	0.99	(359hz,3.75Khz)	(0,135hz),(7.1Khz,∞)
Bplow	0.93	(0,977hz)	(1.17Khz,∞)
Bphigh	0.95	(1.31Khz,∞)	(0,998hz)
F697	0.93	(667hz,747hz)	(0,588hz),(848hz,∞)
F770	1.02	(735hz,819hz)	(0,687hz),(853hz,∞)
F852	0.90	(801hz,912hz)	(0,775hz),(942hz,∞)
F941	0.91	(900hz,970hz)	(0,871hz),(990hz,∞)

Table 2: Simulation after synthesis

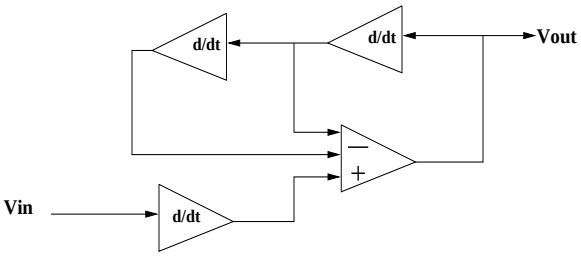

Figure 5: Bandpass Filter Topology

filters. In this case, three differential amplifiers and one adder/subtracter were allocated.

The system was sized using APE for MOSIS HP0.8μ process parameters. The SPICE simulation results for the sized topologies are shown in Table 2, and Figure 6 shows the ac analysis for the 697hz bandpass filter. All filters are working in the specification ranges.

5 Conclusions

A synthesis design methodology for behavioral mixed-signal systems was presented, where a subset of the VHDL-AMS was used as the input specification. The methodology was applied successfully to design a DTMF decoder. In future work, the design methodology will be integrated with a design automation tool such that mixed-signal systems can be automatically synthesized from behavioral specifications through physical layout.

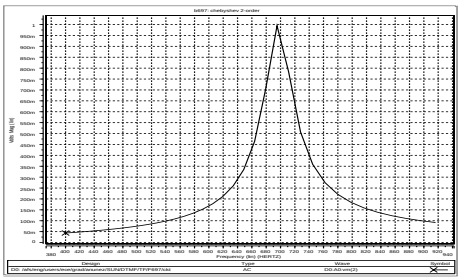

Figure 6: Bandpass Filter Simulation

Acknowledgment. This work was sponsored by the USAF, Wright Laboratories, Wright Patterson Air Force Base under contract number F33615-96-C-1911. The authors also thank Nazanin Mansouri and Satish Ganesan for their contributions.

References

[1] A. Vachoux, "VHDL Analog and Mixed-Signal Extensions Through Examples", ASP-DAC'97, Chiba, Japan 1997.

[2] L. Carley, G. Gielen, R. Rutenbar, W. Sansen, "Synthesis Tools for Mixed-Signal ICs: Progress on Frontend and Backend Strategies", DAC Proceedings, pp298-303, June 1996.

[3] R. Harjani, R. Rutenbar, L. Carley, "OASYS: a framework for analog circuit synthesis", IEEE Trans CAD, Vol8. No 12, pp 1247-1266, December 1989.

[4] E. S. Ochotta, R. A. Rutenbar, L. R. Carley, "Synthesis of high-performance analog circuits in ASTRX/OBLX", IEEE Trans. CAD, Vol. 15, pp. 273-294, Mar. 1996.

[5] J. Vandenbussche, S. Donnay, F. Leyn, G. Gielen, W. Sansen, "Hierarchical top-down design of analog sensor interfaces: from system-level specifications down to silicon", DATE Proceedings 1998, pp. 716-720, February 1998.

[6] R. Vemuri, N. Dhanwada, A. Nunez, P. Campisi, "VASE: VHDL-AMS Synthesis environment Tools for Mixed-Signal Systems", Proc. EETimes Conference on Analog And Mixed-Signal Applications, pp. 1C:77-1C:84, July 1997.

[7] C. Grimm, P. Oehler, I. Kanakis, K. Waldschmidt, "KANDIS - A Tool for System-Level Specification and Design of Mixed-Signal Systems", BMAS'97 Proceedings, Washington D.C., October 1997.

[8] P. Campisi, "A CMOS Analog Cell Library for Analog Synthesis Systems", Master of Science Thesis, University of Cincinnati, 1998.

[9] A. Nunez-Aldana, R. Vemuri, "Performance Estimation for CMOS Analog Circuit Synthesis", DDEL technical report, University of Cincinnati, 1998.

[10] M. Ismail, T. Fiez, "Analog VLSI Signal and Information Processing", McGraw-Hill, pp. 284-296, 1994

[11] R.N. Battista, C.G. Morrison, D.H. Nash, "Signaling System and Receiver for Touch-Tone Calling", IEEE Transactions on Communications Electronics, pp. 9-17, March 1963.

[12] B.J. White, G.M. Jacobs, G.F. Landsburg, "A Monolithic Dual Tone Multi-frequency Receiver", IEEE Journal of Solid State Circuits, vol. SC-14, no. 6, pp. 991-997, December 1979.

[13] HSPICE User's Manual, Meta-Software, Inc. 1996.