Behavioural VHDL-AMS Model for Monolithic Programmable Gain Amplifiers

Dimo S. Martev[1], Ivan D. Panayotov[2] and Ivailo M. Pandiev[3]

Abstract – **This paper presents behavioural VHDL-AMS model for monolithic programmable gain amplifiers (PGAs). The proposed model is independent from actual technical realizations and is based upon compromises regarding the representation of exact circuit structures in the model. For creating the PGA model, techniques known from modelling operational amplifiers have been adapted. The model accurately reflects input resistance, transfer functions (amplifier gain in binary steps versus controlling digital code), small-signal frequency responses, output resistance and other basic dc and ac electrical parameters of the real device. Model parameters are extracted for the integrated PGA AD526 from Analog Devices as an example. To confirm the validity, the simulation results are compared with the manufacturer's data, where a good agreement is found between simulations and performance of the actual device.**

Keywords – **Analog circuits, Programmable gain amplifiers, Operational amplifiers, Mixed-signal simulation, VHDL-AMS.**

I. Introduction

The monolithic programmable gain amplifiers (PGAs) are an important link between analog and digital signal processing circuits. They have an analog signal input, an analog signal output and a digital control of the voltage amplification. Typical PGAs may be configured either for selectable *binary gains* such as 1, 2, 4, 8 etc, or they might also be configured for *decade gains* such as 10, 100, 1000 etc. In addition, the most of the PGAs contain input and output voltage buffer [1-3].

Testing the workability of electronic circuits with PGAs is done usually using SPICE (Simulation Program with Integrated Circuit Emphasis). A variety of SPICE macromodels for the PGAs, are available in the literature [4-7]. The majority of published macromodels contain extensive number of active and passive elements. Furthermore, testing complete mixed-signal systems with large number of elements is an extremely difficult process and can often become infeasible due to the limitation of simulation capacity.

One method to decrease simulation time and improve the convergence, without a significant loss of information, is by using behavioural modelling technique. Nowadays, one of the most effective techniques for behavioural modelling of mixed-signal electronic circuits is by using VHDL-AMS. To the authors' knowledge, behavioural models of monolithic PGAs have not yet been reported in the literature. It is, therefore, the purpose of this paper to present a behavioural VHDL-AMS model for the most common monolithic PGAs.

II. Monolithic PGAs

The monolithic PGAs are controllable amplifiers in which the voltage gain can be conveniently changed via digital code or dc controlling voltage (current). In fact, the electronic circuits including PGAs are very suitable in a user's view point, because the electrical parameters can be changed electronically and do not contain mechanical potentiometers or analog switches. The PGAs are widely used in mixed-signal applications, such as programmable scaling amplifiers, controllable active filters and oscillators, sonar systems, amplitude-stabilized oscillators, programmable power supplies, etc. The monolithic PGA AD526 from Analog Devices [8] is of particular interest here. The IC AD526 is a digitally programmable gain amplifier with single-ended input and output. Fig. 1 summarizes the external view of AD526. The negative feedback of the amplifier is closed by connecting pin *Sense* with pin *Force*. The controllable digital code, applied to pins

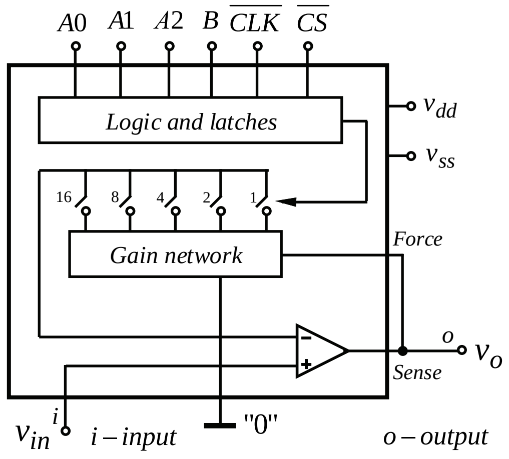
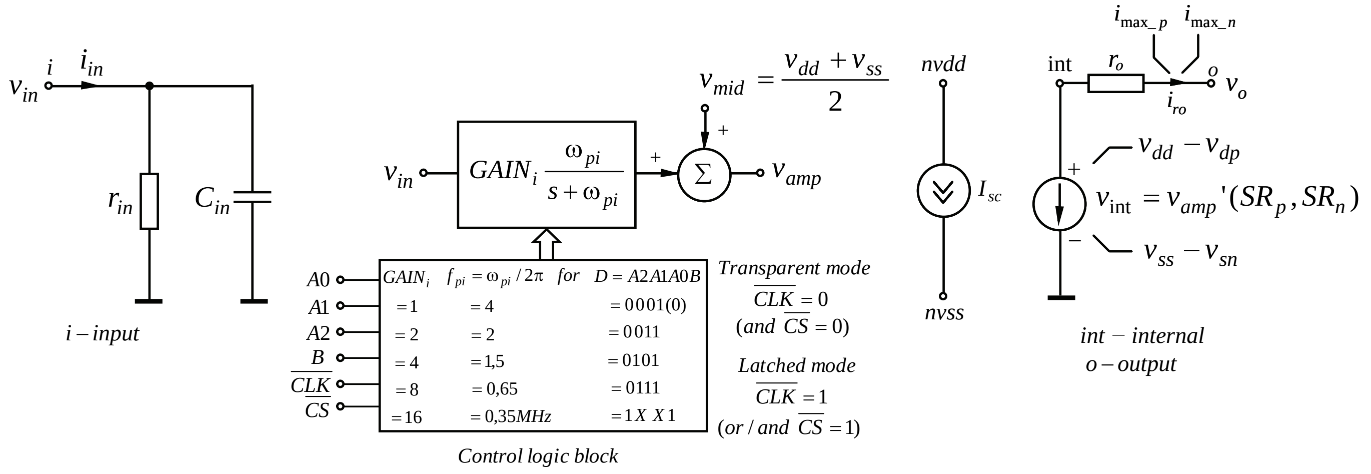


Fig. 1. PGA AD256A external view.

$A0$, $A1$, $A2$, B, $\overline{CLK}$ and $\overline{CS}$, provide voltage gains 1, 2, 4, 8 and 16. The internal structure of this amplifier generally consists of a non-inverting amplifier with a resistor network (or *gain network*), ideal voltage-controlled switches and a control logic block (or *logic and latches*). The control logic block must ensure the operation in either *latched* or *transparent mode*. In *latched mode*, the amplifier AD526 is capable of storing the gain code ($A0$, $A1$, $A2$, B) under the direction of control inputs $\overline{CLK}$ and $\overline{CS}$, which are func-

[1]Dimo S. Martev is with the Faculty of Electronics, TU-Sofia, Kl. Ohridski 8, 1000 Sofia, Bulgaria, E-mail: dsmartev@yahoo.com
[2]Ivan D. Panayotov is with the Faculty of Electronics, TU-Sofia, Kl. Ohridski 8, 1000 Sofia, Bulgaria, E-mail: idp@ecad.tu-sofia.bg
[3]Ivailo M. Pandiev is with the Faculty of Electronics, TU-Sofia, Kl. Ohridski 8, 1000 Sofia, Bulgaria, E-mail: ipandiev@tu-sofia.bg



v_{in} i i_{in}
r_{in} C_{in}
$i - input$
$v_{mid} = \frac{v_{dd} + v_{ss}}{2}$
v_{in} $GAIN_i \frac{\omega_{pi}}{s + \omega_{pi}}$ + Σ v_{amp}
A0 A1 A2 $\overline{B}$ $\overline{CLK}$ $\overline{CS}$
$GAIN_i$ $f_{pi} = \omega_{pi}/2\pi$ for $D = A2A1A0B$
$=1$ $=4$ $=0001(0)$
$=2$ $=2$ $=0011$
$=4$ $=1,5$ $=0101$
$=8$ $=0,65$ $=0111$
$=16$ $=0,35MHz$ $=1X\,X1$
Control logic block
Transparent mode $\overline{CLK} = 0$ (and $\overline{CS} = 0$)
Latched mode $\overline{CLK} = 1$ (or / and $\overline{CS} = 1$)
nvdd I_{sc} nvss
$i_{\max\_p}$ $i_{\max\_n}$
int r_o o v_o i_{ro}
$v_{dd} - v_{dp}$
$v_{int} = v_{amp}'(SR_p, SR_n)$
$v_{ss} - v_{sn}$
$int - internal$
$o - output$

Fig. 2. Circuit diagram of the Programmable Gain Amplifier (PGA) behavioural model.

tionally and electrically equivalent. Alternatively, in *transparent mode* it can respond directly to gain code changes if the control inputs are tied low. In fact, the AD526 is a typical representative of the commercially available PGAs, which are widely used for design and analysis of mixed-signal applications.

The electrical performance of the PGA AD526 is analysed performing electrical simulations with OrCAD PSpice simulator, using PGA macromodel given in [7]. Furthermore, the obtained simulation results are compared with the manufacturer's data. The analyses of the electrical characteristic curves lead to the definition of a behavioural VHDL-AMS model for PGA.

III. Behavioural modelling with VHDL-AMS

"The small simulation model is beautiful".
Stewart Robinson

The technical requirement for effective models is generally agreed when the simplest model possible is developed. Simple models have a number of advantages. They can be developed faster, are more flexible, require less data, run faster and it is easier to interpret the results, since the structure of the model is better understood. As the complexity increases, these advantages are lost [9]. The proposed behavioural model of the PGA is developed following the design method based on a Top-Down analysis approach and applying simplification and build-up technique. The process of model building and testing can be broken down into three basic steps: structure the model, build the model and validate the model [10].

A. A behavioural language: VHDL-AMS

VHDL-AMS is a comparatively new standard 1076.1 of VHDL that supports hierarchical description and simulation of analog, digital and mixed-signal applications with conservative and non-conservative equations [11, 12]. On the mixed-signal side, a variety of abstraction levels is supported. The VHDL-AMS modelling is not restricted to mixed-signal applications but also support thermal, mechatronic, optical and other systems.

B. A PGA behavioural VHDL-AMS model

The behavioural model of the PGA is built using the results obtained by analyses of the PGA macromodel from [7]. The circuit diagram of the PGA model is shown in Fig. 2, where the different stages are presented with controlled sources and passive components. The model includes the following elements and parameters with numerical values: $r_{in} = 10M\Omega$ and $C_{in} = 10pF$ – input resistance and capacitance; $I_{sc} = 10mA$ – dc supply current; $GAIN_i$ = 1, 2, 4, 8 and 16, where $i = \overline{1,4}$; $f_{pi} = \omega_{pi}/2\pi = 4$, 2, 1,5, 0,65 and 0,35$MHz$ the $-3dB$ frequencies for the dominant poles at gains 1, 2, 4, 8 and 16, respectively; $SR_p = -SR_n = 6V/\mu s$ – positive and negative slew rates at gains 1, 2 and 4; $SR_{pl} = -SR_{nl} = 24V/\mu s$ – positive and negative slew rates at gains 8 and 16; v_{int} – output voltage-controlled voltage source; $v_{dp} = v_{dn} = 3V$ – positive and negative voltage drops for the output voltage limitation; $i_{\max\_p} = -i_{\max\_n} = 30mA$ – maximum output currents; $r_o = 0{,}1\Omega$ – output resistance.

The proposed model includes small- and large-signal effects such as (1) accurate input impedance, (2) amplifier gain in binary steps versus controlling digital code, (3) transparent and latched mode of operation, (4) ac small-signal frequency responses, (5) slew rates, (6) dc supply current, (7) voltage and current limitations, and (8) output resistance.

The mathematical equations that describe the model can be given as

$$i_{in} = v_{in}\left(\frac{1}{r_{in}} + sC_{in}\right) \tag{1}$$

$$v_{amp} = GAIN_i \frac{\omega_{pi}}{s + \omega_{pi}} v_{in} + v_{mid} \tag{2}$$

$$I_{sc} = I_s \tag{3}$$

```
library ieee;
use   ieee.math_real.all;
use   ieee.std_logic_1164.all;
use   ieee.electrical_systems.all;

entity ad526 is
      generic(
   --generic constants here);
   port (terminal input,output,nvdd,nvss:ELECTRICAL;
   signal b,cs_n,clk_n : in STD_LOGIC;
   signal a0,a1,a2: in STD_LOGIC);
end entity ad526;

architecture behavioral of ad526 is
   --constant declarations here
   terminal internal:ELECTRICAL;
   quantity vin across iin,icin through input;
   quantity vdd across nvdd;
   quantity vss across nvss;
   quantity isc through nvdd to nvss;
   quantity vint across iintern through internal;
   quantity vro across iro through internal to
       output;
   quantity vout across output;
   quantity vamp:VOLTAGE;
   quantity iro_h:CURRENT;
   quantity vmid:VOLTAGE;
   signal gain:real:=1.0;
   signal a0_b,a1_b,a2_b:BIT;
   signal b_b,cs_n_b,clk_n_b:BIT;
   signal control_in:BIT_VECTOR(0 to 2);
begin
   isc==supply_current;
   iin==vin/rin;
   icin==cin*vin'dot;
   iro_h==vro/ro;
   vmid==(vdd+vss)/2.0;
   if  gain=1.0 use
    vamp==vin'ltf((0=>wp1*1.0),(0=>wp1,1=>1.0))+vmid;
   elsif gain=2.0 use
    vamp==vin'ltf((0=>wp2*2.0),(0=>wp2,1=>1.0))+vmid;
   elsif gain=4.0 use
    vamp==vin'ltf((0=>wp4*4.0),(0=>wp4,1=>1.0))+vmid;
   elsif gain=8.0 use
    vamp==vin'ltf((0=>wp8*8.0),(0=>wp8,1=>1.0))+vmid;
   else
    vamp==vin'ltf((0=>wp16*16.0),(0=>wp16,1=>1.0))+vmid;
   end use;
   if vamp'above(vdd-vdp) use
       vint==vdd-vdp;
   elsif not vamp'above(vss+vsn) use
       vint==vss+vsn;

   else
    if gain=8.0 or gain=16.0 use
        vint==vamp'slew(SRpl,SRnl);
       else
        vint==vamp'slew(SRp,SRn);
       end use;
      end use;
   if iro_h'above(imax_p) use
       iro==imax_p;
      elsif not iro_h'above(imax_n) use
       iro==imax_n;
      else
       iro==iro_h;
      end use;
      break on gain,
                vamp'above(vdd-vdp),
                vamp'above(vss+vsn);
   b_b<=to_bit(b);
   cs_n_b<=to_bit(cs_n);
   clk_n_b<=to_bit(clk_n);
   control_in<=(to_bit(a2),
      to_bit(a1), to_bit(a0));
   case_change:process is
        procedure gain_change
        (signal_in:in BIT_VECTOR;
            signal bu:in BIT;
            signal s:out REAL) is
            begin
             if bu='1' then
                  case signal_in(0 to 2) is
              when b"000" => s <= 1.0;
              when b"001" => s <= 2.0;
              when b"010" => s <= 4.0;
              when b"011" => s <= 8.0;
              when others => s <= 16.0;
                  end case;
             else s <= 1.0;
                   end if;
            end procedure gain_change;
   begin
      if clk_n_b='0' and cs_n_b='0' then
            gain_change(control_in,b_b,gain);
            else
         wait until clk_n_b='0' and cs_n_b='0';
                  gain_change(control_in,b_b,gain);
            end if;
      wait on clk_n_b,cs_n_b,control_in,b_b,gain;
end process case_change;
end architecture behavioral;
```

Fig. 3. A PGA behavioural VHDL-AMS model.

$$v_{mid} = (v_{dd} + v_{ss})/2 \tag{4}$$

$$SR = \left.\frac{dv_{amp}}{dt}\right|_{t=0} \tag{5}$$

$$v_{\text{int}} = v_{amp} \tag{6}$$

$$v_o = \begin{cases} v_{dd} - v_{dp} - r_o i_{ro}, & v_{amp} \ge v_{dd} - v_{dp} \\ v_{\text{int}} - r_o i_{ro}, & v_{dd} - v_{dp} < v_{amp} < v_{ss} + v_{sn} \\ v_{ss} + v_{sn} - r_o i_{ro}, & v_{amp} \le v_{ss} + v_{sn} \end{cases} \tag{7}$$

where I_s is the DC supply current, $GAIN_i = 1, 2, 4, 8$ and 16 is the voltage gains ($i = \overline{1,4}$) and ω_{pi} is the $-3dB$ radian frequencies.

Short-circuit current limiting is simulated with model parameters $i_{\max\_p}$ and $i_{\max\_n}$. In short-circuit mode the output current i_{ro} is limited to one of the values $i_{\max\_p}$ or $i_{\max\_n}$.

Fig. 3 shows the behavioural VHDL-AMS model of PGA. The library clause and the use clause make all declarations in the packages `electrical_systems`, `math_real` and `std_logic_1164` visible in the model. This is necessary, because the model uses nature `electrical` from package `electrical_system` and constant `math_2_pi` for the value of π from package `math_real`. The package `std_logic_1164` declares the signals `a0`, `a1`, `a2`, `B`, `CS_N` and `CLK_N` by `std_logic` type. The proposed PGA model is composed by an *entity* and an *architecture*, where bold text indicates reserved words and upper-case text indicates predefined concepts. The *entity* declares the generic model parameters, as well as specifies interface terminals of nature `electrical` and electrical ports of `std_logic` type. The generic parameters and constants, used in the simultaneous statements, are not given with their concrete numerical values in the model description. The proposed PGA model includes the following electrical terminals: input port – `input`, output port – `output`, port for the positive supply voltage – `nvdd` and port for the negative supply voltage – `nvss`. The model has one inner terminal `internal`. It's used to specify the controlled source `vint`.

The *architecture* contains the implementation of the model. It is coded by combining structural and behavioural elements.

IV. MODEL PERFORMANCE

The verification of the proposed behavioural PGA model is performed by comparing simulation results with the manufacturer's data for the AD526A, biased with ±15V. The simulations of the model are performed within System Vision 5.5 program (from Mentor Graphics). The test circuits are created following the test conditions, given in the semiconductor data book of the corresponding PGA.

To calculate the frequency characteristics at different voltage gain ($GAIN = 1, 2, 4, 8$ and 16), *ac analyses* are performed within the frequency range from $10Hz$ to $10MHz$. Simulated ac voltage gains as a function of the frequency are shown in

Fig. 4. The comparison gives a very good correspondence between the behavioural of the proposed PGA model and the real amplifier, with a resulting error not higher than 2%.

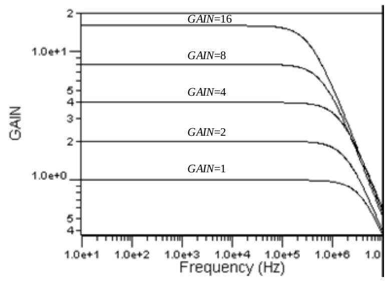


Fig. 4. The simulated frequency responses for the proposed PGA model.

To validate the proposed PGA model, simulation of 12+4-bit Floating-point analog-to-digital converter (FP-ADC) shown in Fig. 5 was carried out for the aforementioned model. This FP-ADC consists of a binary weighting circuit, realized with window comparator, a 12-bit ADC and a five-range PGA with AD526. The simulation results are shown in Fig. 6. As a result, the voltage gain of the PGA AD526 is modulating by the level of the input voltage v_{in}. The AD526, in conjunction with the comparator circuit, scales the input voltage v_o for the 12-bit ADC to a range between half scale and full scale for the maximum usable resolution.

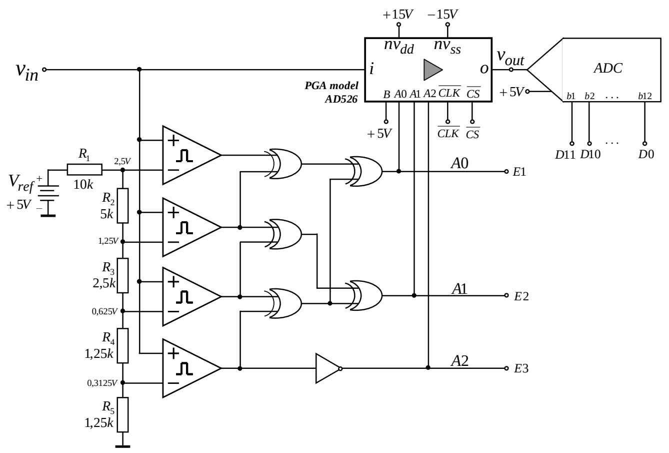


Fig. 5. 12+4-bit Floating-point analog-to-digital converter used in testing the proposed behavioural PGA model.

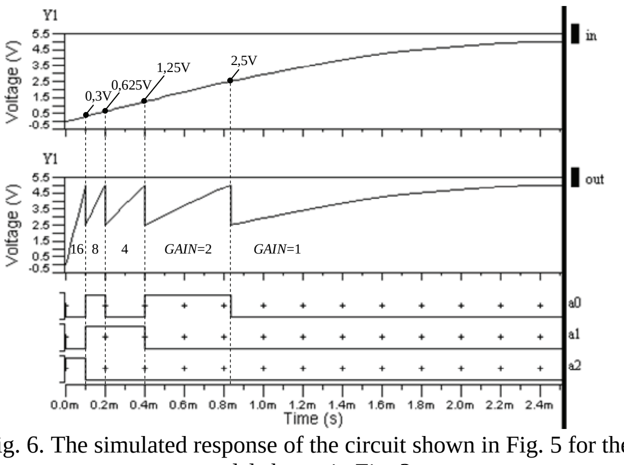


Fig. 6. The simulated response of the circuit shown in Fig. 5 for the model shown in Fig. 3.

The output data of this FP-ADC is presented as a 16-bit word, the lower 12 bits ($D_0 \dots D_{11}$) from the ADC form the *mantissa* and the upper 4 bits ($E_1 \dots E_3$, B) from the digital signal used to set the gain of the PGA form the *exponent*. The smallest value of the FP-ADC is $U_{LSB} = (5V/2^{12})/16 \approx 76\mu V \;\; = 76\mu V$ and the dynamic range is $D_R = 20\lg(5V/76\mu V) \; = 96{,}3dB$.

V. Conclusion

In this paper a generalized behavioural VHDL-AMS model of monolithic PGA, based on the data sheet characteristics, has been presented. The proposed model accurately describes the behaviour of a most common monolithic PGA with *binary voltage gains*, including electrical parameters, such as input impedance, amplifier gain versus controlling digital code, bandwidth, voltage and current limitations, slew rate, output resistance etc. The workability of the model was proved by comparison of simulation results and data sheet parameters of the monolithic PGA AD526A from Analog Devices. To achieve simplicity of the mathematical equations describing the model, it neglects several second-order effects found in the PGAs, such as the noise, the temperature effects, board parasitics, distortion (harmonic, intermodulation), etc.

One of the aims of the further work is to explore the possibility of modelling second-order effects of the typical PGA, which are taken into consideration for designing mixed-signal applications.

Acknowledgement

This paper is a part of a project which is sponsored by the research program of the Technical University of Sofia, Bulgaria.

References

[1] M. Seifart, *Analoge Schaltungen. 6 Auflage*. Verlag Technik Berlin, 2003 (in German).
[2] W. Jung, W. *Op amp applications*. Analog Devices, MA, 2002.
[3] J. Siegl, *Schaltungstechnik – Analog und gemischt analog/digital. 2. Auflage.* Springer-Verlag, 2005 (in German).
[4] *Spice models*, Analog Devices, 2000.
[5] *Spice models*, Texas Instruments, 2008.
[6] *Spice macromodels*, National Semiconductor, 2010.
[7] I. Pandiev, P. Yakimov, T. Todorov, "Macromodeling of programmable gain amplifiers". E+E, No 7-8, pp. 69-76, 2009.
[8] *AD526 software PGA – data sheet*, Analog Devices, 1999.
[9] S. Robinson, "Conceptual modeling for simulation: Issues and research requirements", Proceedings of the 2006 Winter Simulation Conference, pp. 792-799, 2006.
[10] S. Robinson, *Successful Simulation. A Practical approach to Simulation Projects*. McGraw-Hill, 1998.
[11] E. Christen, K. Bakalar, "VHDL-AMS – A hardware description language for analog and mixed-signal applications, IEEE Trans. on cir. and syst. – II, vol. 46 (10), pp. 1263-1272, 1999.
[12] *Definition of analog and mixed signal extensions to IEEE standard VHDL*, IEEE Standard 1076.1, 1999.