A New Approach for an AHDL Based on System Semantics

Youcef Bourai

Micro-electronics laboratory
CDTA
128, Chemin Mohammed Gacem, El madania
Algiers, Algeria
Tel: 213 02 67 73 25
Fax:213 02 66 26 89
email:lme@ist.cerist.dz

N. Izeboudjen, Y. Bouhabel, A. Tafat

Micro-electronics laboratory
CDTA
128, Chemin Mohammed Gacem, El madania
Algiers, Algeria
Tel: 213 02 67 73 25
Fax:213 02 66 26 89
email:lme@ist.cerist.dz

Abstract:
A new approach for Analog Hardware Design Language (AHDL) is presented. This is based on system semantics principle. This principle allows to define a language that provides a unified to define a language that provides a unified syntax to describe the different aspects of a Op_Amp. This is applicable by considering that the basic components of an Op_Amp are adirectional systems. These components are described by combinators. A set of semantic functions are applied on these combinators to give them a meaning.

I. Introduction

Due to communication image processing or automotive applications growth, ASIC nowadays integrate both analog and digital functions on one chip. While the digital part can be efficiently and rapidly designed by mature CAD tools, analog part is more time consuming because of the lack of analog design tools. Several CAD tools have emerged from both industry and academia to overcome this lack [1-7]. They all share the following disadvantage: the designer is kept out of important design decision taken in the background. He can only play with specifications and watch the results. Designing analog circuits with these tools is therefore a static process. This is a serious drawback since rapid process evolution and increasing demand for high-performance circuit challenges a steady evolution in circuit topologies.

Thus a tool that allows the extension of a system with new topologies will be certainly very helpful. One mean to extent a system with new topologies is to provide an HDL. Such mean can ease the entry of new descriptions into the synthesis tool. This solution works quite good in digital part using VHDL. Several attempts have been made to define an AHDL [8-11]. However these languages are mainly used for the behavioral description and simulation. Furthermore, they do not provide a unified description that can capture all the aspects of analog circuits.

In this paper we present a new approach that does not only provide to define an AHDL but also adds an important feature lacking in the existing AHDLs. In deed the language yields the description of all the aspects of analog circuits in a unified formalism. This is possible by applying the system semantics principle.

The paper is organized as follows. In section II a theoretical background is recalled. Section III recalls briefly the notion of σ-calculus. Section IV shows that the σ-calculus is inappropriate to describe the behavioral aspects of complex circuits. Section V gives an alternative to σ-calculus. Section VI treats an example in details. Section VII gives some practical examples. Section VIII ends the paper with conclusions and points out some future works.

II. System Semantics

Most existing Systems Description Languages (SDLs) are based on syntactic and semantic principles derived from imperative programming languages. The syntactic and semantic principles of these programming languages are suitable for expressing algorithms. In other words, these principles are suitable for describing computational systems. However, most natural and technical systems are governed essentially by the laws of the nature which are better modeled by mathematical functions than by algorithms. Secondly, the structure of systems, being static in nature, is even less well-described by algorithmic description. Since electronic systems belong to the class of systems whose behavior cannot be properly characterized as computational process, another formalism should be addressed. The system semantic based on functional paradigm has been suggested in [12] and [13], where it has been shown that this formalism is preferable over the imperative semantics.

The principle of system semantics is to distinguish between the various interpretations given to a formal description of the system. This can be done explicitly by means of semantic functions. This makes it possible to use a single formal language rather than a collection of different languages to express various system aspects.

In the area of analog circuits the variety of issues to be considered in one part, and the inappropriate semantic basis of current SDLs in another one has hampered the development of efficient HDL as in the digital case. The application of system semantics together with a different choice of language concepts will certainly fill this gap.

Before we define the principle of system semantics, let us define the concept of semantics:

A. Semantics [13]

Is introduced explicitly by mean of a model consisting of a meaning function m mapping element of S (The set of sentences of language) into element of a domain of interpretation D which is a set of possible meanings: $m \in S \rightarrow D$.

B. System Semantics [12]

Is the description of physical systems and their properties by mean of semantic functions. Given a formal language S, we define for every property of interest (structure, behavior, performance) a model M with meaning function $m{:}S \rightarrow D$, where D is the set of values the described property can assume.

An Analog circuit have different aspects. So, many properties may be given to the circuit. In other words many models can coexist. Thus a relationship between models must be set. We say that a model provides at least as much details as a model M' written $M \geq M'$ if two sentences s and s' are equivalent w.r.t the model M are also equivalent w.r.t model M'. An initial system model is defined as a model M such that $M \geq M'$ for all M' i.e its the most detailed of the useful models.

The system semantic influences an SDL in two important issues:

(a) The level of detail provided by the language which is that of initial system model. In circuit design, language syntax is chosen such that the initial system model is the structural one.

(b) The syntax is also chosen in such a way to ensure compositionality, that is, the meaning of a system must be expressible in terms of the meaning of the constituents. In other hand the compositionality is determined by the flow information inside the system. Without this property, the semantic definitions exhibit too many complex interdependies to be useful as rules (or "axioms") for transformational reasoning, restricting their usefulness to simulation only.

The application of this concept to digital and analog circuits has led to following conclusions:

1. Description levels to which information flow can be attributed: these are called unidirectional systems such as: logic gates, amplifiers.

2. Description levels at which the concept of information flow is not meaningful: these are called adirectional systems such as transistors, resistors, capacitance's.

For unidirectional systems the λ-calculus constitutes an excellent match in view of formal description [12]. For adirectional systems another type of calculus is required which is the subject of the next section.

III. Sigma Term for Adirectional Systems

The σ-term is the generalization of the λ-term over arbitrary structures. The principle is described in [14], [15]. The general form in (abstract syntax) for a σ-term is:

σ.varlist.appset

where *appset* is the following form: *{system varlist, ..., system varlist}.*

Example: *Def T∈ B^3*

with T<A, B, C> =

σ<x,y,z>.{A<x,c>,B<y,c>,C<z,c>}

In the structural interpretation of a σ-term for electrical circuits, the abstraction *σ.varlist* describes the externally accessible connections and the *appset* the internal connections. Every variable name stands for an interconnection net. For instance, the structural interpretation of the T circuit is depicted by Fig.1 [19]:

In the behavioral interpretation of a σ-term for electrical circuits, the most general model describes the circuit equations.

IV. Application of Sigma Term to Analog Circuits

Since the σ-term can describe arbitrary systems where the directionality of the flow information is not meaningful, this would be suitably applied for describing analog circuits whose basic components such as resistors, capacitance's, transistors or basic elements such as: current mirror, differential pair, high impedance...etc. are adirectional objects.

For this purpose we define a simple formal language based on σ-term to describe a reduced Operational Amplifier which is an OTA. The OTA is composed of : current source, differential pair and current mirror.

The abstract description of a transistor, a differential pair and a current mirror applying σ-term are depicted by Fig.2, Fig.3 and Fig.4 respectively.

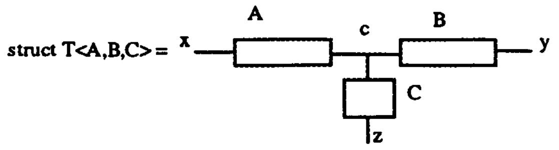

Fig. 1: The T circuit

Def tr = σ<a,b,c>.{tr<a,b,c>}

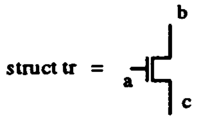

Fig.2. Abstract description and the structure of a transistor

***Def diffpair** = σ<a,b,c,d,e>.{tr<c,a,e>,tr<d,b,e>}*

struct diff pair =

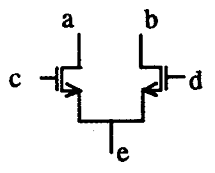

Fig.3. The abstract description and the structure of a differential pair

***Def** cur_mir = σ-<a,b,c>.{tr<a,d,c>,*<a,d>, tr<a,b,c>}*
/*The operator * specifies that noeuds *a* and *d* are the same*/.

struct *cur_mir* =

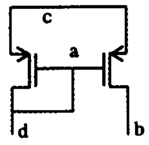

Fig.4. The abstract description and the structure of a current-mirror.

Then the concrete description of the OTA of Fig.5 is:
***Def** simple OTA = σ < a, b, c, d, e, f > .*
*{(σ < a, b, c > . { tr< a, d, c >, *< a, d >,tr< a, b, c >}) < j, b, a >,*
/ current mirror description applied to the actual parameters j,b,a*/*
(σ < a, b, c, d, e > .{ tr< c, a, e >, tr< d, b, e >) <j, b, c, d, k >,
/differential pair description applied to the actual parameters j, b, c, d, k/*
σ < a, b, c >.{tr< a, b, c >} < e, k, f> / current source description applied to the actual parameters e,k,f*/*
}

A. The behavioral interpretation of a simple OTA
Let us consider that all the transistors of the OTA shown in Fig.5 are the same. Given the node numbering of Fig.6, the behavioral interpretation of a transistor in a small signal mode is defined as follows:
beh tr = λ<<v_0,,i_0 ,> , <v_1 ,i_1 >, <v_2 , i_2 >>.($i_0 = 0 \wedge i_1 = (v_1 - v_2)/r + g_m * (v_0 - v_2)$)
Using the labeling convention in [15] that distinguishes between different occurrences of a same variable (in the same context), the interpretation of σ-abstractor above mentioned is:
beh[σ-varlist.appset] = λarglist.∃intlist.expression.

struct OTA =

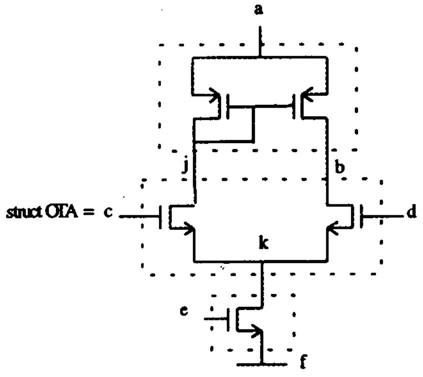

Fig.5. Description and the structure of a simple OTA.

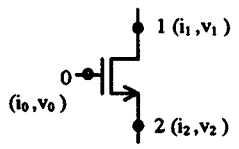

Fig.6 . Node numbering of a transistor

/* the function *beh[]* is the behavior interpretation of its argument*/
where in our case (the OTA of Fig. 5):
arglist = <<v_{a0} , i_{a0} >, <v_{b0} , i_{b0} >, <v_{c0} , i_{c0} >,<v_{d0} , i_{d0} >,<v_{e0} , i_{e0} >,<v_{f0} , i_{f0} >>.
Inlist = <<v_{j0} , i_{j0} >, <v_{b1} , i_{b1} >, <v_{a1} , i_{a1} >,<v_{j1} , i_{j1} >,<v_{b2} , i_{b2} >,<v_{c1} , i_{c1} >,<v_{d1} , i_{d1} >, <v_{k0} , i_{k0} >,<v_{e1} , i_{e1} >,<v_{k1} , i_{k1} >,<v_{f1} , i_{f1} >>.
Thus far the application is straightforward. But the pattern of the expression is more complex and too long:

*expression = beh({(σ < a, b, c > . { tr< a, d, c >, *< a, d >,tr< a, b, c >}) < j, b, a >∧beh (σ < a, b, c, d, e > . { tr< c, a, e >, tr< d, b, e >) <j, b, c, d, k >∧beh(tr < e, k, f>∧...*

$$52 equations \begin{cases} v_{a_0} = v_{b_0}; \\ \vdots \\ i_{a_0} = i_{a_1}; \\ \vdots \\ i_k = \left(v_{k_1} - v_{k_2}\right)/r + gm*\left(v_{k_0} - v_{k_2}\right) \end{cases}$$

This approach is not practical since for a simple OTA 52 equations are necessary to describe the behavioral interpretation. Therefore, another approach should be investigated to express the behavior more elegantly.

V. Analog Circuits and Combinators

Since it has been proved that the behavioral interpretation based on σ-calculus is inappropriate to describe the behavioral model of complex circuits, the use of combinators to solve the problem of analog circuits remains the adequate solution [16]. Although this solution has not been considered satisfactory because every interconnection pattern requires different combinators, this is not the case of Operational Amplifiers where a small set of combinators can cover a wide spectrum of circuit structures. Thereby, a little modification of abstract syntax of the description language is necessary. The basic cells: capacitance, resistors, transistors are considered as *first-order* constants and particular connections such as: current mirror, differential pair, inverter, ... are considered as *second-order* constants as is summarized by TABLE I. For this purpose, the following combinators have been defined see Fig.7.
SYS_SC<sc, A>: allows a connection between a system A and a current-source sc.
MIR_DIF1<pairdif, cur_mir>: allows a connection between a differential pair and a current-mirror.

TABLE I
SYNTACTIC CATEGORY CLASSIFICATION.

Syntactic category	structural interpretation	Examples
first-order	basic cells	current mirror, differential pair, inverter
second-order	combinators	**SYS_SC, MIR_DIF1, CASCADE**

MIR_DIF2<pairdif,tr,invrt,cur_mir> : allows a connection of current-mirror to a differential pair through an inverter and a transistor.
CASCADE<A,B>: allows a cascade connection between two stages A and B.

A. *Behavioral Interpretation of the Combinators*

Inspired by system semantics, we have defined a set of behavioral models that are:
IOUT = (R, I_{out})
where: R is the real values that the semantic function I_{out} may take.
The semantic function I_{out} is defined as follows:
I_{out} SYS_SC<sc,A> = I_{out}sc;
I_{out} MIR_DIF1< pairdif,cur_mir> = I_{out} pairdif;
I_{out} MIR_DIF2 <pairdif, tr, invrt, cur_mir> = I_{out} pairdif;
I_{out} CASCADE <A,B> = + (I_{out} A)(I_{out} B).
ROUT = (R,R_{out})
where: R is the real values that the semantic function R_{out} may take.
The semantic function R_{out} is defined as follows:
R_{out} SYS_SC <sc,A> = R_{out} A;
R_{out} MIR_DIF1< pairdif,cur_mir> = #(R_{out} pairdif)(R_{out} cur_mir) /* #ab = (1/a) + (1/b) */
R_{out} MIR_DIF2 <pairdif, tr, invrt, cur_mir> =#(*(R_{out} pairdif)(* gm_1 r_1))((R_{out} cur_mir)(*gm_2 r_2));
I_{out} CASCADE <A,B> = * (R_{out} A)(R_{out} B).

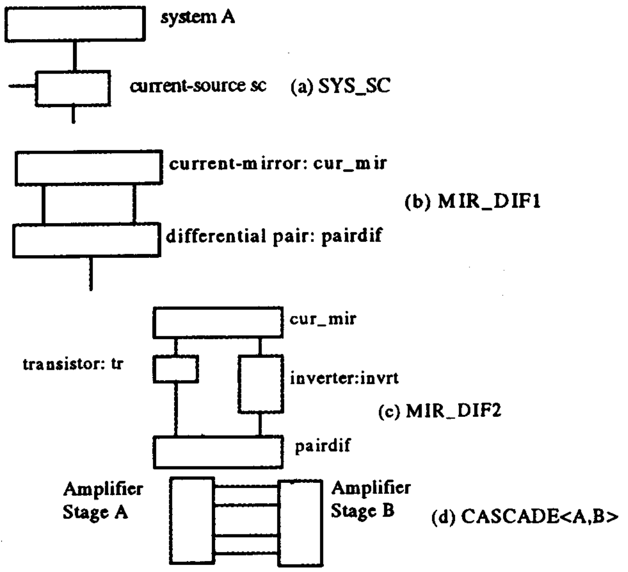

Fig.7. The different combinators

GAIN = <R, GAIN>
The semantic function GAIN is defined as follows:

GAIN SYS_SC <sc,A> = GAIN A;
GAIN MIR_DIF1< pairdif,cur_mir> = * (R_{out} MIR_DIF1< pairdif,cur_mir>)(gm pairdif);
GAIN MIR_DIF2 <pairdif, tr, invrt, cur_mir> = * (R_{out} MIR_DIF2 <pairdif, tr, invrt, cur_mir>)(gm pairdif);
GAIN CASCADE <A,B> = * (GAIN A)(GAIN B).

The equations are written in the polish format for compiling matter. In the same manner we have defined the models [17]: CMR+, CMR-, Slew Rate, ...etc.
Once the equation that models the specification is reached any known optimization algorithm can be used. In our case the equations are translated into the MATLAB format. The algorithms included in MATLAB packages are then applied [18].

VI. EXAMPLE

Hereafter is an example of an OTA on which our approach is applied.
The textual description of the OTA of Fig.8 is:

Def OTA = SYS_SC<sc,< MIR_DIF1<pa irdif,cur_mir> >
where pairdif<M1,M2>
cur_mir<M3,M4>
sc<M5>

After writing the textual description the user introduces interactively the desired specifications (see column 2 in TABLE II). Then the tool applies the semantic functions to the sentences describing the circuit to get the analytical models of the specifications (the behavioral interpretation) as detailed below:
The output current :
I_{out} SYS_SC<sc, MIR_DIF1< pairdif,cur_mir>> = I_{out} sc;
The Slew rate:
SR SYS_SC<sc, MIR_DIF1< pairdif,cur_mir>>
=/(I_{out}SYS_SC<sc,MIR_DIF1<pairdif,cur_mir>>)cl
= /(I_{out} sc) cl /* which is the slew rate of an OTA, Sr = I_5/C_{load}*/

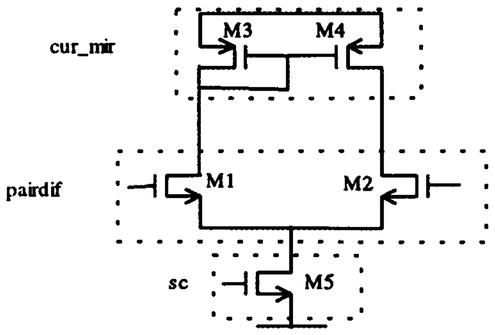

Fig.8. Simple OTA.

The power dissipation:

PDISS SYS_SC<sc, MIR_DIF1< pairdif,cur_mir>> =
= * (+ V_{DD} V_{SS})(I_{out} SYS_SC<sc, MIR_DIF1< pairdif,cur_mir>>)
= * (+ V_{DD} V_{SS})(I_{out} sc)
/* which is the power dissipation of an OTA that = (V_{DD} + V_{SS}) * I_5 */

The gain:

GAIN SYS_SC<sc, MIR_DIF1< pairdif,cur_mir>>
= GAIN MIR_DIF1< pairdif,cur_mir>
= * (R_{out} MIR_DIF1< pairdif,cur_mir>)(gm pairdif);

The CMR+:

CMR+ SYS_SC<sc, MIR_DIF1< pairdif,cur_mir>>
=CMR+MIR_DIF1<pairdif,cur_mir>
= - V_{DD} (-V_{DS3} (+V_{t1} V_{t3}))
/* which is the CMR+ of OTA that = V_{DD} -V_{DS3} - V_{t1} + V_{t3} */

The CMR-:

CMR- SYS_SC<sc, MIR_DIF1< pairdif,cur_mir>>
= +(CMR+ MIR_DIF1< pairdif,cur_mir>) V_{DS5}
= + (+V_{SS} V_{GS1}) V_{DS5}
/* which is the CMR- of OTA that = V_{SS} + V_{GS1} + V_{DS5}*/

The result of the optimization phase is a SPICE net list of the dimensioned OTA. SPICE net list is generated to verify the predicted performance. A comparison between the results of column 5 and 6 in TABLE II indicates that the results agree reasonably.

VII. Experimental Results

We are giving two examples: a Basic Two Stages Amplifier (BTS) and a cascode amplifier, each for which a textual description and the structural schema are given. The deviation between the simulated and the predicted values of some specifications are somewhat important because for the current version we have not focused our efforts on the accurate models of the MOS transistor. The current semantic functions are limited to:

- analytical equations of the specifications that are written in linear mode (fixed length).
- analytical equations such that the different transistors are considered in the saturation region and where we have neglected the second order effects.

A. BTS

The textual description of the BTS amplifier shown in Fig.9 is:

TABLE II
Simulated and Predicted Performance of Simple OTA

Attribute	Unit	Spec.	Weight	our Tool	SPICE
CMR+	V	–	–	1.883	1.90
CMR-	V	–	–	-1.253	-1.8
Voswing	V	< 5	0.5	3.934	3.64
Power dissip.	µW	= 350	0.4	323.2	450
Slew rate	V/µs	= 7	0.8	6.464	9
Gain	db	> 30	1	33.29	30.09
GB. Width	MHZ	–	–	4.17	6.8

Def BTS = CASCADE < SYS_SC < MIR_DIF1 < sc ,pairdif,cur_mir > >, invrt >
Where pairdif < M1, M2 >
cur_mir < M3, M4 >
sc < M5 >
invrt < M6, M7 >

With the desired specifications of the column 1 in TABLE III the result of the optimization is given by column 5.

B. Cascode Amplifier

The textual description of the cascode amplifier shown in Fig.10 is:

Def CASCODE = SYS_SC < sc, MIR_DIF2 <pairdif, tr, invrt, cur_mir>>
Where pairdif < M1, M2 >
cur_mir < M3, M4 >
sc < M5 >
invrt < MC1, MC2 >
tr<MC3>

With the desired specifications of the column 1 in TABLE IV the result of the optimization is given by column 5.

VIII. Conclusions and Future Works

A language for Operational Amplifier has been presented. The language is based on the system semantic formalism. This formalism has allowed to describe the different aspects of the circuit from a unique textual source program. The source program describes the schema of the circuit. The description is done by a set of combinators defined for this purpose. The combinators have been adopted after we have proved that the σ-calculus cannot relate easily the behavioral aspect of an Op_Amp. By generalizing the model concept, combinators are assigned a meaning by a semantic function. Thus, applying several semantic functions on the same

TABLE III
Simulated and Predicted Performance of a BTS

Attribute	Unit	Spec.	weight	our Tool	SPICE
CMR+	V	–	–	2.034	2.04
CMR-	V	–	–	-1.315	-1.8
$Vout_{swing}$	V	= 4	0.5	4.245	3.84
Power dissip.	µW	< 1000	0.5	318.5	459
Slew rate	V/µs	> 2	1	15.44	8.16
Gain	db	> 60	1	63.78	60.5
GB. Width	MHZ	> 1	0.8	13.31	10

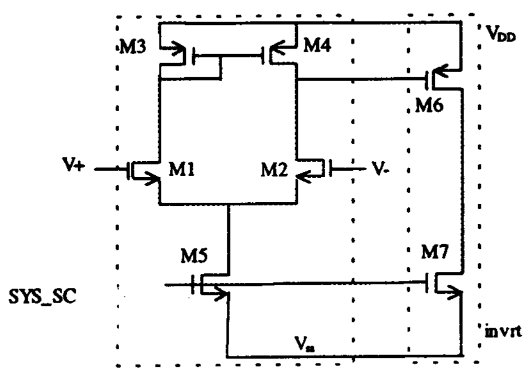

Fig.9. Basic Two Stage (BTS) Amplifier

combinator it gets several meanings. In order to complete the set of meanings given to the textual description, we plan to define another model which is the geometrical one. This model will be defined for each combinator. The meaning will be given by a semantic function that maps the textual description into layout [19]. Layout generation will be performed hierarchically. The key word ***where*** borrowed from the functional paradigm will play a crucial role in the hierarchy.

References

[1] R. Harjani, R. A. Rutenbar, L. R. Carley " A prototype framework for knowledge-based analog circuit synthesis", Proceedigns of Design Automation Conference'87, pp.42-49, 1987.

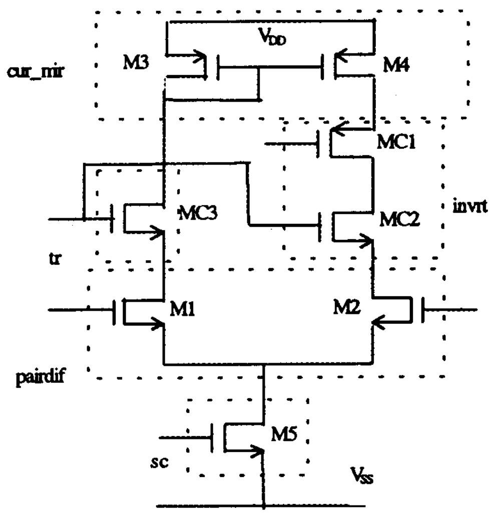

Fig.10. Cascode Amplifier

TABLE IV

SIMULATED AND PREDICTED PERFORMANCE OF CASCODE AMPLIFIER

Attribute	Unit	Spec.	weight	our Tool	SPICE
CMR+	V	--	--	1.42	1.51
CMR-	V	--	--	-1.245	-1.24
$Vout_{swing}$	V	--	--	2.67	3
Power issip.	μW	= 400	0.6	329.29	911
Slew rate	V/μs	> 10	0.6	13.16	18.2
Gain	db	> 65	1	70.40	47.29
GB. Width	MHZ	> 10	0.7	8.28	9.26

[2] H. Y. Koh, C. H. Séquin, P. R. Gray "Automatic synthesis of Operational Amplifiers based on analytical circuit models", Proceedings of ICCAD'87, pp. 502-505, 1987.

[3] Fatehy El-Turcky, Elizabeth E. Perry "BLADES: an artificial intelligence approach to analog circuit design", IEEE Transactions on CAD, Vol. 8, N0. 6, pp. 680-692, June 1989.

[4] R, J, Bowman, and D. J. Lane, "A knowledge-based System for Analog Integrated Circuit Design" Preceedings of IEEE ICCAD, 1985.

[5] E. Berckan, M. d'Abreu, W. Laughton "Analog compilation based on successive decompositoins" Proceedings of Design Automation Conference'88, pp.369-375, 1988.

[6] P. E. Allen, E. R. Macaluso, S. F. Bily and A. P. Nedungadi "AIDE2: An automatic analog IC design system", Proceedings of 1985 IEEE CICC, pp. 498-501.

[7] M. Degrawe "IDAC: an ainteractive design tool for analog integrated circuits", IEEE Journal of Solid-State Circuits, Vol. SC-22, No. 6, pp.1106-1116. December 1987.

[8] E. Berckan "From analog design description to layout: a new approach to analog silicon compilation", Proceedings of IEEE CICC, pp. 4.4.1-4.4.4, 1989.

[9] K. Swings, S. Donnay, W. Sansen "HECTOR: a hierarchical topology-construction program for analog circuits based on a declarative approach to circuit modeling", Proceedings of IEEE CICC, pp. 5.3.1-5.3.4, 1991.

[10] J. P. Harvey, M. I. Elmasry and B. Leung "STAIC: an interactive framework for synthesizing CMOS and BICMOS analog circuits", IEEE trans. on CAD, Vol. 11, no. 11, pp. 1402-1417, November 1992.

[11] R. A. Saleh "Analog Hardware Description Languages", Proceedings of IEEE CICC, pp. 15.1.1-15.1.8, 1994.

[12] R. T. Boute "System Semantics and Formal circuit description", IEEE trans. on Circuits and Systems, Vol. 33, no. 12, pp. 1219-1231, 1986.

[13] R. T. Boute "System Semantics: Principles, Applications and Implmentation", ACM trans. on Programming Languages and Systems, Vol. 10, no. 1, pp. 118-155, 1988.

[14] R. T. Boute "The Sigma Calculus: scoping and substitution in formal descriptions of systems that are not unidirectional", Report No. 92, Department of informatics, University of Nijmegen, January 1987.

[15] R. T. Boute "On the formal description of non-computational objects", in G. David, R. T. Boute, B. Shriver ed., Declarative systems, North-Holand, 1990.

[16] R. T. Boute "Fundamentals of hardware description languages and declarative languages", University of Nijmegen, Holland, 1993.

[17] Y. Bourai , N. Izeboudjen, Y. Bouhabel and A. Tafat "ASTL: Analog Synthesis Tool with Lanaguage", Technical Report, Micro-electronics Lab. CDTA, 1995.

[18] J. E. Dennis and D. J. Woods "New computing environments", Micro-Computer in Large Scale Computing, A. Woods eds. SIAM, pp.116-128, 1987.

[19] M. Seutter "Glass: a system description language and its environment introduction and user manuals", ESPRIT project: E881 FORFUN, University of Nijmegen, June 1991.