conversions. Output conversions for more than eight moduli can be handled with more than one chip and external binary adders. The same technique may be applied to handle output conversion when a larger output resolution is desired. Two (or more) chips can be used to form partial sums, which are then added off-chip.

The architecture can be slightly modified if one wishes to do QRNS processing, although the details of the modification are beyond the scope of the present discussion. The modifications are minimal (and are necessary only to input conversion) but they do increase the I/O burden on the chip. The converter without this modification can be used with QRNS if one is willing to provide a modulo multiply-and-accumulate for each RNS channel, and if one is willing to accommodate the necessary input connections on the RNS processors performing the RNS-to-QRNS conversion.

IV. Conclusion

A simple systolic architecture has been developed which permits both input and output conversion to be performed using the same hardware. The architecture is extremely powerful, as both the number of moduli and their values can be chosen arbitrarily (up to the capacity of the hardware design) and programmed into the converter. Another advantage of the design is that it is not a bottleneck for the RNS system, since its throughput is the same as the individual RNS channels. A chip designed in 1.25μm CMOS allows up to eight 6-bit moduli in the RNS. This design should provide an off-the-shelf solution for most RNS conversion requirements.

References

[1] M. A. Soderstrand, C. Vernia, and J.-H. Chang, "An improved residue number system digital-to-analog converter," *IEEE Trans. Circuits Syst.* vol. CAS-30, pp. 903–907, 1983.

[2] T. Van Vu, "Efficient implementations of the chinese remainder theorem for sign detection and residue decoding," *IEEE Trans. Comput.* vol. C-34, pp. 646–651, 1985.

[3] I. Niven and H. Zuckerman, "An introduction to the theory of numbers," New York: John Wiley & Sons Inc., 1972.

Application of Interval Analysis for Circuit Design

D. M. W. LEENAERTS

***Abstract* —In top-down circuit design, a principle task is to partition and map design constraints on a normal operating range of a collection of sub-blocks. This problem is propagated through each hierarchical level until the solutions for all levels are found. This top-down parameter assignment and instantiation of sub-blocks may eventually break down at some level due to an unrealizable circuit. Then the process has to be restarted a number of times before a realizable partition can be produced. In this paper, an application of interval analysis in the design environment is presented to assure in advance that this process will always yield a solution. The presented methodology and corresponding algorithm can be used in such hierarchical design strategies. At each hierarchical level, the solution space, if nonempty, is valid for all lower levels and is in agreement with decisions taken earlier in the hierarchy.**

Manuscript received April 29, 1989; revised November 1, 1989. This paper was recommended by Associate Editor T. T. Vu.

The author is with the Department of Electrical Engineering, Eindhoven University of Technology, Netherlands.

IEEE Log Number 9034912.

I. Introduction

In complex engineering design tasks, such as analog IC design, the circuit can be decomposed into smaller sub-blocks, which are considered to behave almost independently of each other. Each sub-block can be described by a set of underdetermined equations in some of the circuit parameters and circuit voltages and currents. Due to design constraints for normal circuit operation, some or all of these unknowns may only be chosen within specific domains. The problem is then to solve the equations of each sub-block in such a way that a solution exists when the sub-blocks are interconnected, leaving a maximum amount of freedom for the set of design parameters. A methodology to find such a solution is to use interval arithmetic [1]–[6].

For each parameter of a sub-block, a default and an actual domain—i.e., an interval—is defined. The default domain is based on designer's experience or design constraints. One can, for instance, think of minimum and maximum dimension of transistors in a certain technology or minimum and maximum allowable power dissipation of a sub-block, etc. A valid parameter is a value within the range of the actual domain. Synthesis is performed by a partitioning of the design constraints together with a partitioning of the circuit into smaller blocks at each hierarchical level.

Now, at each level the description of the domains of the parameters must be used to find the solution space for matching the design constraints with the chosen sub-blocks. The solution space actually yields new parameter domains, i.e., the new actual domains for each parameter. The advantage of this technique is that the solution space at each level does not conflict with the solution space at higher levels. Decisions made earlier at higher levels remain valid. The design process goes hierarchical downwards until the solution space for the lowest level, the basic building blocks, is found.

Then, bottom-up, the solution space at higher levels is assembled using the description of the solution at the lowest levels. At the top design level, actual values can be chosen in the produced solution space, automatically resulting in valid parameter values for the lower blocks in agreement with the design constraints at all levels of the hierarchy.

A design problem for which interval techniques can be used is the design of an op-amp. To simplify the problem consider only the gain of a three state op-amp as a design constraint. At the highest design level the gain could be partitioned over these stages in several ways. One level lower, there is a possibility that the decision made earlier is unrealizable. For instance, to design the first stage for the given gain may be impossible, whereas a higher gain in the design of the last stage could be easy. So, at the top level a wrong partition of the design constraint has taken place. To circumvent this problem, consider the interval in which the gain may exist. Now, at the top level, try to find the solution space of partitioning the gain over the three stages, given the domain of the amplification used by each stage. If a solution exists, one level lower a maximum of freedom for the design parameters can be used to design the stages, in agreement with the top level. Above, the interval methodology is used to find the solution space using a decomposition (top-down) strategy with the information for partitioning collected by a bottom-up construction.

At each hierarchical level, using the description of the solution space of the level above and the model description of this level, a set of equations, with domains for the variables, can be formulated. So, all levels can use the same strategy to solve this set of equations, described in an algorithm.

Here a methodology will be presented to find the parametric solution of a set of linear equations with bounded variables. In Section II, an explanation of the solution strategy will be given. In Section III, some examples to demonstrate the application of the algorithm in circuit design are given. Finally, some conclusions are drawn in Section IV.

II. Theory

To use interval techniques, model descriptions for the blocks have to be defined at each hierarchical level, in which the circuit can be partitioned.

In the case of linear sub-blocks, the problem is to find all the solutions x of n equations with m variables $(m \geqslant n)$,

$$Ax = b \qquad \text{with } x \in S_x \tag{1}$$

where S_x represents the domains for all elements of x. The solution for this problem can be derived from an algorithm from Tschernikow [7], who presented an algorithm to find the solution space of a set of k linear equations of p variables,

$$Cu = 0 \qquad \text{with } \forall_i \; u_i \geqslant 0. \tag{2}$$

The solution space describes all non-negative solutions of (2). Because the presented interval algorithm is based on the Tschernikow technique, a detailed explanation to solve (2) is outlined below. The algorithm starts to define a start tableau for (2)

$$T^1 = (T_1^1 | T_2^1) = \left[\begin{array}{ccc|ccc} 1 & & 0 & c_{11} & \cdots & c_{k1} \\ & \ddots & & \vdots & & \vdots \\ 0 & & 1 & c_{1p} & \cdots & c_{kp} \end{array}\right] \tag{3}$$

where T_1^1 is a unit matrix and T_2^1 is composed by placing a row of (2) as a column in (3). For every row i $(i = 1, \cdots, p)$, define $S(i)$ as the collection of columns of T_1^1 with zeros in row i. Define $S(i_1, i_2)$ for every combination (i_1, i_2) $(i_1, i_2 = 1, \cdots, p)$ as the collection of columns of T_1^1 with zeros in both i_1 and i_2. A column of T_2^1, say column j, is chosen at random with at least one nonzero element. From the collection $S(i_1, i_2)$, only the subset with opposite sign in column j, called $\bar{S}(i_1, i_2)$, is important. A new table, T^2, can now be composed by first placing the rows from T^1 with a zero in column j into T^2. Next, search for the pairs (i_1, i_2), for which $S(i) \not\supset \bar{S}(i_1, i_2)$ $(i \neq i_1, i \neq i_2)$, and place any linear combination of row i_1 and i_2 such that a zero in column j is created in T^2.

In the same way, a new table T^i can be composed from table T^{i-1}.

This process ends when there are one or more strict positive or strict negative columns in T_2^i (in which case there is no solution) or there are only zeros in T_2^i. In the latter case the table is given by

$$T^{\text{end}} = (T_1^{\text{end}} | T_2^{\text{end}}) = \left[\begin{array}{ccc|c} b_{11} & \cdots & b_{1p} & \\ \vdots & & \vdots & 0 \\ b_{c1} & \cdots & b_{cp} & \end{array}\right] \tag{4}$$

with the non-negative solution

$$u = \sum_{i=1}^{c} p_i b_i \tag{5}$$

where $b_i = (b_{i1}, \cdots, b_{ip})$ $(i = 1, \cdots, c)$, and p_i is a non-negative parameter. The set of $(b_1, \cdots, b_p)^T$ describes the corners of the convex solution space.

The procedure, written in pseudopascal, is given below

```
procedure Tschernikow;
begin stop := 0; r := 1;
        create start tableau;
        while (r <= n) and (stop = 0) do
begin choose column j from $T_2$
        if no j then stop := 1
                else calculate $S(k)$;
                        calculate $S(k_1, k_2)$;
                        create new tableau;
                        r := r + 1
        end;
end;
```

To use the algorithm of Tschernikow, (1) must be transformed into an equivalent representation of (2). In (1), only the parameter x_i, which is not in agreement with $x_i \geqslant 0$, according to $u_i \geqslant 0$ in (2), must be redefined. When x_i is negative, only the substitution $u_i = -x_i$ is necessary.

However, if x_i is bounded, then use a substitution variable, say u_i, such that u_i is bounded between zero and a positive value, $\bar{u}_i \in [0, \bar{U}_i]$. To get the restriction $u_i \geqslant 0$, use a new variable to describe the bound of u_i in an extra equation,

$$u_i + u_j - \bar{U}_i = 0 \qquad \text{with } u_i, u_j \geqslant 0. \tag{6}$$

Define a slack-variable, say u_c $(u_c = 1)$, and multiply it with the constant $\bar{U}_i$ in (6). This gives the linear equation

$$u_i + u_j - \bar{U}_i u_c = 0 \qquad \text{with } u_i, u_j, u_c \geqslant 0. \tag{7}$$

The total algorithm to solve problems according to (1) can now be defined. First transform the problem into a problem like (2), using the substitution in (6) and (7). Then solve the set of equations using Tschernikow and redescribe the solution space in the variables in which the problem was defined.

To demonstrate the technique to solve a given problem like (1), define

$$\begin{gathered} x_1 - x_2 + 2x_3 + x_4 = -4 \\ -2x_1 - x_2 + x_3 - x_4 = -10 \\ x_1 \in [0,2],\; x_2 \in [1,4],\; x_3 \in [-3,-1],\; x_4 \in [-8,5]. \end{gathered} \tag{8}$$

To reformulate (8) to a form equal to (2), first define

$$\begin{aligned} u_1 &= x_1 \\ u_2 &= x_2 - 1 \\ u_3 &= -x_3 - 1 \\ u_4 &= x_4 + 8 \end{aligned} \tag{9}$$

to get lower bounds for u_1, u_2, u_3, and u_4 equal to zero,

$$\begin{gathered} u_1 - u_2 - 2u_3 + u_4 - 7 = 0 \\ -2u_1 - u_2 - u_3 - u_4 + 16 = 0 \\ u_1 \in [0,2],\; u_2 \in [0,3],\; u_3 \in [0,2],\; u_4 \in [0,13]. \end{gathered} \tag{10}$$

Now, to get the restriction $\forall_i \; u_i \geqslant 0$, use new variables to describe the upper bounds of u_1, u_2, u_3, and u_4 in some extra equations. For u_1 this equation is

$$u_1 + u_5 - 2 = 0 \qquad \text{with } u_1 \geqslant 0, u_5 \geqslant 0. \tag{11}$$

Define a slack-variable, say u_9 with $u_9=1$, and multiply it with all the constants in (10) and (11). This gives the set of equations

$$\begin{aligned}
u_1-u_2-2u_3+u_4 \qquad\qquad & -7u_9=0\\
-2u_1-u_2-u_3-u_4 \qquad\qquad & +16u_9=0\\
u_1 \qquad +u_5 \qquad & -2u_9=0\\
u_2 \qquad +u_6 \qquad & -3u_9=0\\
u_3 \qquad +u_7 \qquad & -2u_9=0\\
u_4 \qquad & +u_8-13u_9=0
\end{aligned}\tag{12}$$

with $\forall_i\ u_i\geqslant 0$. This set of equations is in agreement with the Tschernikow set and can be solved.

Tableau T^1, according to (3), is now described by

$$T^1=\left(T_1^1|T_2^1\right)=\left[\begin{array}{ccccccccc|cccccc}
1&0&0&0&0&0&0&0&0&1&-2&1&0&0&0\\
0&1&0&0&0&0&0&0&0&-1&-1&0&1&0&0\\
0&0&1&0&0&0&0&0&0&-2&-1&0&0&1&0\\
0&0&0&1&0&0&0&0&0&1&-1&0&0&0&1\\
0&0&0&0&1&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&1&0&0&0&0&0&0&1&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&1&0&0&0&0&0&0&1\\
0&0&0&0&0&0&0&0&1&-7&16&-2&-3&-2&-13
\end{array}\right].\tag{13}$$

and u_4. According to (5), the solution of (12) is then given as

$$\begin{bmatrix}u_1\\u_2\\u_3\\u_4\\u_9\end{bmatrix}=p_1\begin{bmatrix}0\\3\\1\\12\\1\end{bmatrix}=p_2\begin{bmatrix}0\\3/2\\2\\25/2\\1\end{bmatrix}+p_3\begin{bmatrix}2\\3\\1/3\\26/3\\1\end{bmatrix}+p_4\begin{bmatrix}2\\1/2\\2\\19/2\\1\end{bmatrix},\tag{16}$$

with $\forall_i\ p_i\geqslant 0$.

The solution space of the slack-variable u_9 from (16) and the fact that $u_9=1$ gives

$$u_9=p_1+p_2+p_3+p_4=1.\tag{17}$$

Relation (17) can be used to reformulate (16) into the solution of (12) as

$$\begin{bmatrix}u_1\\u_2\\u_3\\u_4\end{bmatrix}=p_1\begin{bmatrix}0\\3\\1\\12\end{bmatrix}+p_2\begin{bmatrix}0\\3/2\\2\\25/2\end{bmatrix}+p_3\begin{bmatrix}2\\3\\1/3\\26/3\end{bmatrix}+p_4\begin{bmatrix}2\\1/2\\2\\19/2\end{bmatrix}$$

$$p_1+p_2+p_3+p_4=1,\quad \forall_i\ p_i\geqslant 0.\tag{18}$$

Taking the first column of T_2^1, the collection $S(i)$ is described as

$$S(i)=\{j=1,\cdots,9,\ j\neq i|(\text{column } j \text{ of } T_1^1)\},\quad \forall_{i=1,\cdots,9}.$$

For instance $S(3)=\{1,2,4,\cdots,9\}$. The sets (1,2), (1,3), (1,9), (4,2), (4,3), and (4,9) have opposite signs in the chosen column. It is easy to verify that for these sets, $S(i)\not\supset \bar{S}(i_1,i_2)\ (i\neq i_1, i\neq i_2)$ is valid, so that T^2 is given by

$$T^2=\left(T_1^2|T_2^2\right)=\left[\begin{array}{ccccccccc|cccccc}
0&0&0&0&1&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&1&0&0&0&0&0&0&1&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&1&0&0&0&0&0&0&1\\
1&1&0&0&0&0&0&0&0&0&-3&1&1&0&0\\
2&0&1&0&0&0&0&0&0&0&-5&2&0&1&0\\
7&0&0&0&0&0&0&0&1&0&2&5&-3&-2&-13\\
0&1&0&1&0&0&0&0&0&0&-2&0&1&0&1\\
0&0&1&2&0&0&0&0&0&0&-3&0&0&1&2\\
0&0&0&7&0&0&0&0&1&0&9&-2&-3&-2&6
\end{array}\right].\tag{14}$$

After six loops, the algorithm stops with an end tableau like (4),

$$T^{\text{end}}=\left[\begin{array}{cccccccc|cc}
0&3&1&12&2&0&1&1&1&0\\
0&3/2&2&25/2&2&3/2&0&1/2&1&0\\
2&3&1/3&26/3&0&0&5/3&13/3&1&0\\
2&1/2&2&19/2&0&5/2&0&7/2&1&0
\end{array}\right].\tag{15}$$

Due to the fact that the variables u_5, u_6, u_7, and u_8 were only used to describe the bounds of u_1, u_2, u_3, and u_4, they are not important by consideration of the solution space for u_1, u_2, u_3,

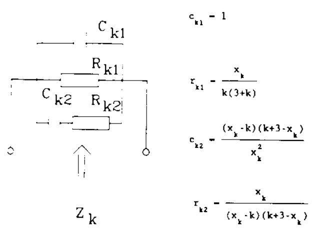
Using the substitution relation (9), the solution of (8) is found as

$$\begin{bmatrix}x_1\\x_2\\x_3\\x_4\end{bmatrix}=p_1\begin{bmatrix}0\\4\\-2\\4\end{bmatrix}+p_2\begin{bmatrix}0\\5/2\\-3\\9/2\end{bmatrix}+p_3\begin{bmatrix}2\\4\\-4/3\\2/3\end{bmatrix}+p_4\begin{bmatrix}2\\3/2\\-3\\3/2\end{bmatrix}$$

$$p_1+p_2+p_3+p_4=1,\quad \forall_i\ p_i\geqslant 0.\tag{19}$$

Equation (19) describes the full solution space of (8). To verify the correctness of (19), the vectors (19)—i.e., the corners of the solution space—can be substituted in (8).

The algorithm, outlined above, will be demonstrated for some applications in the following section.


C k1
R k1
C k2
R k2
Z k
c k1 = 1
r k1 = x k / k(3+k)
c k2 = (x k - k)(k+3-x k) / x k 2
r k2 = x k / (x k - k)(k+3-x k)
Fig. 1. Realization of impedance Z_k.

III. Applications of the Interval Algorithm

To demonstrate that the presented technique can be used to solve different design problems, two applications are given.

In the first example, an impedance $Z = Z_1 + Z_2 + Z_3$ is considered, representing a series connection of three impedances

$$Z_k = \frac{x_k + s}{(k+s)(3+k+s)}, \qquad k \in \{1,2,3\}. \tag{20}$$

Each of the impedances Z_k is composed of resistors and capacitors only (see Fig. 1).From circuit theory it is known that for non-negative C_{k2} and R_{k2}, it is required that $k \leqslant x_k \leqslant 3+k$.

The goal is to find solutions for $\underline{x}$, such that

$$\arg\{Z(s)\}|_{s=2j} = -45^\circ \quad \text{and} \quad \arg\{Z(s)\}|_{s=5j} = -60^\circ. \tag{21}$$

This results in the following set of equations:

$$\begin{aligned} 0.2018x_1 + 0.0696x_2 + 0.0155x_3 &= 1 \\ 0.1330x_1 + 0.0971x_2 + 0.0636x_3 &= 1 \\ \text{with } 1 \leqslant x_1 &\leqslant 4 \\ 2 \leqslant x_2 &\leqslant 5 \\ 3 \leqslant x_3 &\leqslant 6. \end{aligned} \tag{22}$$

Calculation of the solution space, using the above outlined procedure, yields

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = p_1 \begin{bmatrix} 3.058 \\ 3.528 \\ 3 \end{bmatrix} + p_2 \begin{bmatrix} 3.922 \\ 2 \\ 4.468 \end{bmatrix} \tag{23}$$

with $p_1 + p_2 = 1$ and $\forall_i \ p_i \geqslant 0$.

According to (23), several configurations are possible. Choosing $p_1 = p_2 = 0.5$ gives the configuration of Fig. 2(a), and $p_1 = 1$, $p_2 = 0$ leads to the realization of Fig. 2(b).

The second application is the design of a two-stage op-amp for dc biasing. To calculate the dc behavior linear equations appear to be sufficiently accurate.

Using a bottom-up strategy, the design process is to decompose the op-amp in a first and a second stage, to design them independently, and to connect the stages together to find an overall solution for the dc operation point of the op-amp. Using the interval algorithm to solve this problem, the designer is free to make a parameter choice at the highest design level, in agreement with the design of the two stages. The parameters for each stage are the bias current, the dc input/output voltage, and the dissipation. A possible linear description of the first

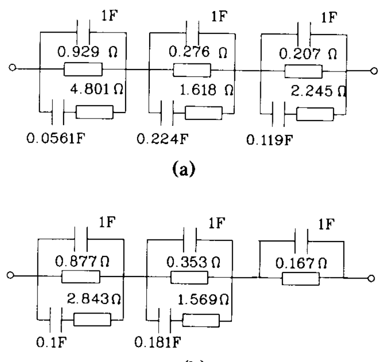


Fig. 2. Realization of (20) with (a) $p_1 = p_2 = 0.5$ and with (b) $p_1 = 1, p_2 = 0$.

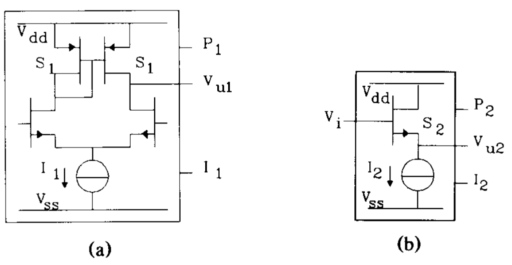


Fig. 3. Design of a two stage op-amp ($V_{dd} = -V_{ss} = 5$ V, $V_{tn} = -V_{tp} = 1$ V, $S_1 = (W/L)_1, S_2 = (W/L)_2$). (a) First stage. (b) Second stage.

stage, according to Fig. 3(a), could be given by

$$\begin{aligned} &V_{u1} = V_{dd} - |V_{tp}| - R_i I_1 && V_{u1}\,(\text{V}) \in [2.5, 3.8] \\ &R_i = \left(\sqrt{\beta_p S_1 I_1}\right)^{-1} = 100\ \text{k}\Omega && I_1\,(\mu\text{A}) \in [2, 20] \\ &P_1 = (V_{dd} + |V_{ss}|) I_1 && P_1\,(\text{mW}) \in [0, 0.1] \end{aligned} \tag{24}$$

and for the second stage, a simple source-follower,

$$\begin{aligned} &V_{u2} = V_i - V_{tn} - R_u I_2 && V_{u2}\,(\text{V}) \in [-1, 1] \\ &R_u = \left(\sqrt{2/\beta_n S_2 I_2}\right) = 20\ \text{k}\Omega && I_2\,(\mu\text{A}) \in [10, 100] \\ &P_2 = (V_{dd} + |V_{ss}|) I_2 && P_2\,(\text{mW}) \in [0, 2] \\ & && V_i\,(\text{V}) \in [2, 4] \end{aligned} \tag{25}$$

according to Fig. 3(b).

From the highest design level, the domains for the parameters are given. The solution space for the unknowns in the first stage as produced by the interval algorithm is given by

$$\begin{bmatrix} V_{u1}\,(\text{V}) \\ I_1\,(\mu\text{A}) \\ P_1\,(\mu\text{W}) \end{bmatrix} = k_1 \begin{bmatrix} 3.8 \\ 2 \\ 20 \end{bmatrix} + k_2 \begin{bmatrix} 3 \\ 10 \\ 100 \end{bmatrix}$$

$$\text{with } \sum_{i=1}^{2} k_i = 1, \quad \forall_i \ k_i \geqslant 0. \tag{26}$$

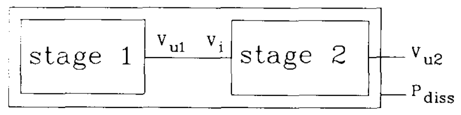


Fig. 4. Interconnection of two stages (P_{diss} (mW) $\in [0,1.8]$).

In the same way, the second stage yields

$$\begin{bmatrix} V_{u2}\,(\text{V}) \\ V_i\,(\text{V}) \\ I_2\,(\mu\text{A}) \\ P_2\,(\mu\text{W}) \end{bmatrix} = q_1\begin{bmatrix}1\\4\\100\\1000\end{bmatrix} + q_2\begin{bmatrix}-1\\2\\100\\1000\end{bmatrix} + q_3\begin{bmatrix}0.8\\2\\10\\100\end{bmatrix} + q_4\begin{bmatrix}1\\2.2\\10\\100\end{bmatrix}$$

$$\text{with } \sum_{j=1}^{4} q_j = 1 \quad \text{and } \forall_j\ q_j \geqslant 0. \qquad (27)$$

At the lowest level, the solution spaces for the two stages, which are independent of each other, are now found. The design process goes one hierarchical level upwards. At this level the stages are interconnected, which means $V_{u1} = V_i$ (see Fig. 4). Also at this level, a new design constraint, the total dissipation P_{diss}, is added to the system. P_{diss} is valid within a domain, P_{diss} (mW) $\in [0,1.8]$.

Using the interconnection relations $V_{u1} = V_i$ and $P_{\text{diss}} = P_1 + P_2$, as well the restrictions $k_1 + k_2 = q_1 + q_2 + q_3 + q_4 = 1$, produces a new set:

$$\begin{aligned} &3.8k_1 + 3k_2 - 4q_1 - 2q_2 - 2q_3 - 2.2q_4 = 0\ (V_{u1} = V_i) \\ &-20k_1 - 100k_2 - 1000q_1 - 1000q_2 - 100q_3 - 100q_4 + P_{\text{diss}} = 0\ (P_{\text{diss}} = P_1 + P_2) \\ &k_1 + k_2 = 1 \\ &q_1 + q_2 + q_3 + q_4 = 1 \\ &P_{\text{diss}}\,(\mu W) \in [0,1800] \end{aligned} \qquad (28)$$

according to (1).

Using the outlined procedure in Section II, the solution space of this set is given by

$$(k_1,k_2,q_1,q_2,q_3,q_4,P_{\text{diss}}) = \sum_{i=1}^{6} r_i \boldsymbol{a}_i$$

$$\begin{aligned} \text{with} \quad & r_1 + r_2 + r_3 + r_4 + r_5 + r_6 = 1, \quad \forall_i\ r_i \geqslant 0 \\ \text{and } & \boldsymbol{a}_1 = (1,0,0.9,0.1,0,0,1020) \\ & \boldsymbol{a}_2 = (0,1,0.5,0.5,0,0,1100) \\ & \boldsymbol{a}_3 = (1,0,0.9,0,0.1,0,930) \\ & \boldsymbol{a}_4 = (0,1,0.5,0,0.5,0,650) \\ & \boldsymbol{a}_5 = (1,0,0.89,0,0,0.12,920) \\ & \boldsymbol{a}_6 = (0,1,0.44,0,0,0.56,600). \end{aligned} \qquad (29)$$

When V_{u2} and P_{diss} are defined as the new output variables for the sub-block on this level, they follow from

$$\begin{bmatrix} V_{u2}\,(\text{V}) \\ P_{\text{diss}}\,(\text{mW}) \end{bmatrix} = r_1\begin{bmatrix}0.8\\1.02\end{bmatrix} + r_2\begin{bmatrix}0\\1.1\end{bmatrix} + r_3\begin{bmatrix}0.98\\0.93\end{bmatrix} + r_4\begin{bmatrix}0.9\\0.65\end{bmatrix} + r_5\begin{bmatrix}1\\0.92\end{bmatrix} + r_6\begin{bmatrix}1\\0.6\end{bmatrix}$$

with

$$r_1 + r_2 + r_3 + r_4 + r_5 + r_6 = 1, \qquad \forall_i\ r_i \geqslant 0. \qquad (30)$$

At this stage the full solution space is described. From these values using optimization criteria for V_u and P_{diss}, the sizes of the transistors could be determined. For instance, if the designer wants $V_u = 0$ V, r_2 must be equal to 1. The sub-blocks are defined using (26), (27), (29) and (30). The first stage is designed with $V_{udc} = 3$ V, $I_1 = 10$ μA, and $P_1 = 0.1$ mW, and the second stage with $V_i = 3$ V, $I_2 = 100$ μA and $P_2 = 1$ mW. All the parameters take values within their design constraints. Using (24) and (25), the values for S_1 and S_2 are found. Simulations with SPICE prove the usefulness of linear equations for such a design problem.

Normally the equations describing the dc behavior of an op-amp are nonlinear. To deal with the nonlinear problem solution techniques for interval arithmetic in nonlinear equations are under development.

IV. Conclusion

The presented interval technique can be used in the design environment to partition and map design constraints on a normal operating range of a collection of sub-blocks. This technique is valid only for linear equations.

Using a top-down strategy, the first advantage is that the solution space, if it exists, does not conflict with the solution space at higher levels and so decisions made earlier remain valid. At each level there is a maximum amount of freedom for the set of design parameters. The second advantage of this technique is the implementation in an algorithm. At each hierarchical design level the same problem exists, which means that the same algorithm can be used at each level. This makes the implementation of the interval technique within the design process easier.

Acknowledgment

The author wishes to thank W. M. G. van Bokhoven and J. A. Hegt for the valuable discussions and suggestions.

References

[1] R. E. Moore, "Methods and applications of interval analysis," Studies in Applied Mathematics, 1979.

[2] E. P. Oppenheimer and A. N. Michel, "Application of interval analysis techniques to linear systems: Part I—Fundamental results," *IEEE Trans. Circuits Syst.*, vol. CAS-35, pp 1129–1138, Sept. 1988.

[3] ——, "Application of interval analysis techniques to linear systems: Part II—The interval matrix exponential function," *IEEE Trans. Circuits Syst.*, vol. CAS-35, pp. 1230–1242, Oct. 1988.

[4] ——, "Application of interval analysis techniques to linear systems: Part III—Initial value problems," *IEEE Trans. Circuits Syst.*, vol. CAS-35, pp. 1243–1256, Oct. 1988.

[5] U. W. Kulisch and W. L. Miranker, "Arithmetic operations in interval spaces," *Computing, Suppl.* 2, pp. 51–67, Springer-Verlag, 1980.

[6] S. M. Markov, "Some applications of extended interval arithmetic to interval iterations," *Computing, Suppl.* 2, pp. 69–84, Springer-Verlag, 1980.

[7] S. N. Tschernikow, *Lineare Ungleichungen* Berlin: VEB Deutscher Verlag der Wissenschaften, 1971.