Buletinul Ştiinţific al Universităţii "Politehnica" din Timişoara

Seria ELECTRONICĂ şi TELECOMUNICAŢII
TRANSACTIONS on ELECTRONICS and COMMUNICATIONS

Tom 49(63), Fascicola 1, 2004

Mathematical counter/converter for mixed analog and digital simulations.

Valentin Maranescu[1], Lucian Jurca[2]

Abstract - This paper proposes a new method for simulations in mixed analog and digital circuits that offer one easy way to run all spice analysis from one or more different digital code combinations. With this method it is possible to generate or sweep a digital code without using counters, LFSR register, that work only in transient simulations. It is an easy way to set a digital stimulus with binary output and decimal format input.
Keywords: mixed circuits, digital stimulus, digital code sweep.

I. INTRODUCTION

Many times we need to stop a counter or a sequential circuit in one or in every state, to run analysis like .OP, .AC or .DC. This is always impossible, because the counters output a known digital code just after the load signal or many clock periods that occur only in transient simulations. A similar problem we can have with shift registers with or without feedback or even with the most simple memory element: the latch.

The solution in this case is to use a digital stimulus instead of the output of counter or shift registers. Editing this stimulus every time we want to change the code makes things annoying and very slow, and we need also a lot of manual conversion from digital to hex or binary because of the format of this digital stimulus [1].

This paper presents a very efficient method for situations like this.

The method is very useful when we want to focus on the functionality of the circuit, not on the digital part responsible with code generation that can be very complex sometimes and that makes impossible to "pause" the circuit in one known ore unknown state for other simulations than transient.

Using this method we open also new frontiers in mixed analog/digital simulations.

II. EIGHT BIT MATH COUNTER/CONVERTER STRUCTURE WITH ANALOG CARRY AND BORROW

The present work proposes a very simple math counter/converter implemented with basic functions from ABM.lib (Analog Behavioral Modeling) [2][3]. The counter has analog borrow (BO) input and carry (CO) output, so this kind of structure can be concatenated for a greater number of bits. The BO input for next stage is the input value for the first structure divided by 2^n, where n is the total number of bits from previous structures. An eight bit structure is presented in fig.1. The first structure in chain needs at input a structure presented in fig.2. This allows a typical converter characteristic if the offset input is set to 0.5, but it can be used also with other value depending how much we want to translate the characteristic. The analog bits are obtained with function (1):

$$Vi = LIMIT\left[10^{16}\sin\left(\left(\frac{BO}{2^i}+1\right)\pi\right),0V,+5V\right] \quad (1)$$

Thus we obtain an analog bus V[7..0]. To see these bits like a digital bus or like digital codes (LO and HI states) we have to add a buffer after each bit Vi with analog level, and we obtain an eight bit digital bus A[7..0] where:

$$Ai = digitalvalue(Vi) \quad (2)$$

The bits reflect the sign of different sine waves with periods multiple with power of two, but for avoiding high range of values used by simulator, the term $1/2^i$ is not calculated for each bit, and the previous argument is multiplied by 1/2. In this way number of

[1]Facultatea de Electronică şi Telecomunicaţii, Departamentul Electronică Aplicată, Bd. V. Pârvan Nr. 2, 300223 Timişoara, e-mail: valentin.maranescu@etc.utt.ro.
[2]Facultatea de Electronică şi Telecomunicaţii, Departamentul Electronică Aplicată, Bd. V. Pârvan Nr. 2, 300223 Timişoara, e-mail: lucian.jurca@etc.utt.ro

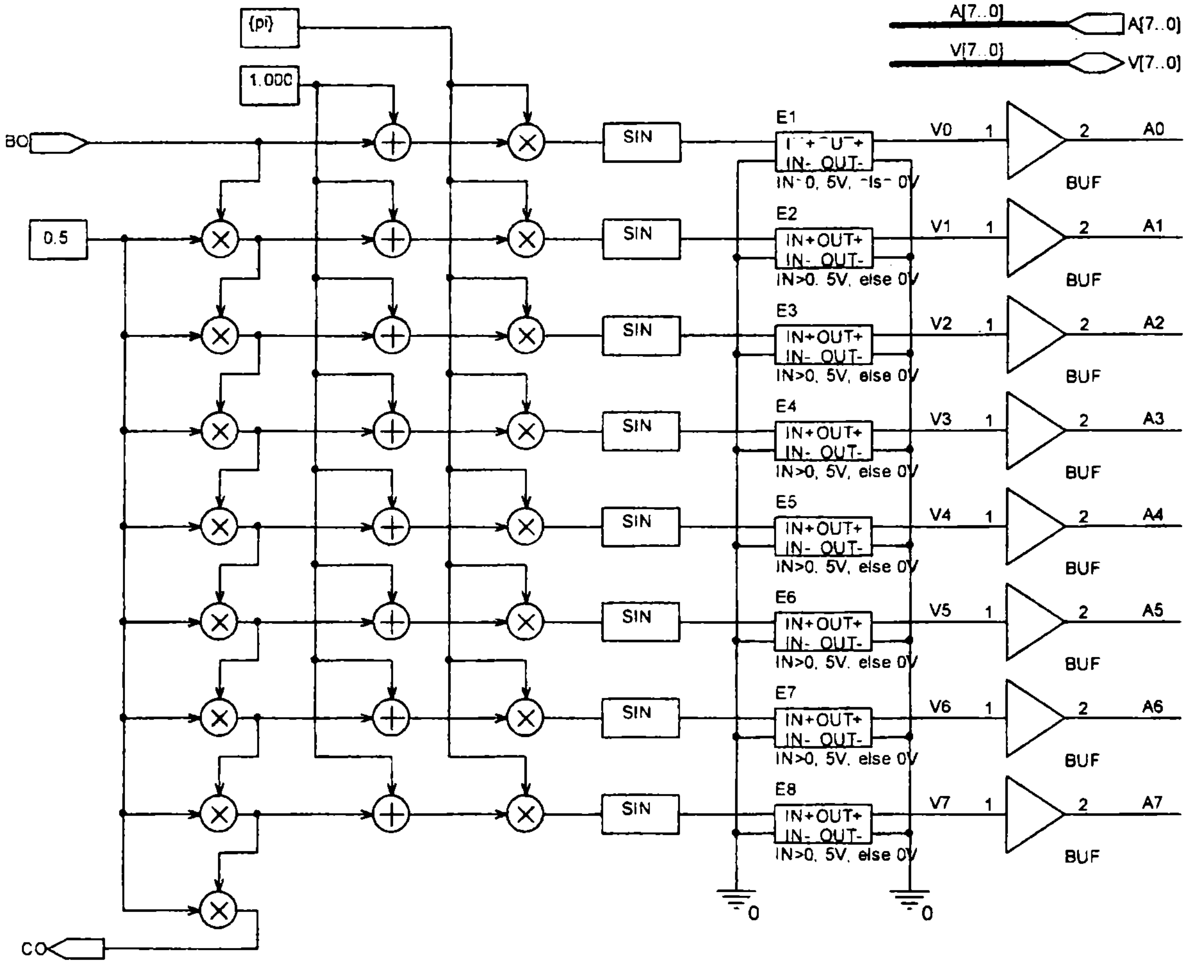

Fig 1. Eight bit math counter/converter.

bits can be extended up to 16. Because the counter works with analog values it is important to use numbers that can be represented in the simulator or operating systems.

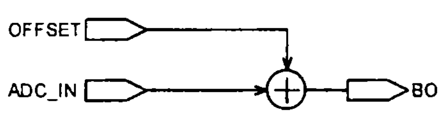

Fig 2. Input circuit for first structure.

To convert the sign of the sin waves in digital levels. we use a high gain amplifier (10^{16}), and a limiter to 0V for negative values or +5V for positive values. This functions act like a rectifier. The period for bit A_i is 2^{i-1}, and the period for one digital code is 1. In fig.3 bit A0 and A3 are illustrated with the corresponding sine waves.

III. RUNNING THE COUNTER IN TRANSIENT ANALYSIS

If we want to run the counter in transient analysis, with no clock signal, we have to use a PWL source at de ADC_IN port. The "increment" signal for counter occurs every time the PWL source value plus OFFSET value passed an integer value. It is easy to see that we can have an Up or Down count with different speed depending on the slope value and sign for PWL source (fig.4). The circuit is a math analog to digital converter but I call it "counter" because it acts like a counter of analog integer values passed by the input voltage source or parameter, when time elapsed.

Because we can define different slope this mean that the speed of the "counter" can be slowed or increased for some code intervals, and we can simulate very accurate and focus on details for a particular code intervals. It is also possible to skip codes by simply stepping the analog value from one value to other. In fig. 4 is shown an example how to set variable clock speed for counter, up or down count, or pause for some codes.

It is important that the maximum time step in transient analysis to be smaller than the period corresponding to one digital code. If the time step is bigger than the period of one code, the counter "skips" some codes, until the time step is smaller than the period of LSB. This means that this counter can work well even after it "fails" to some codes. For

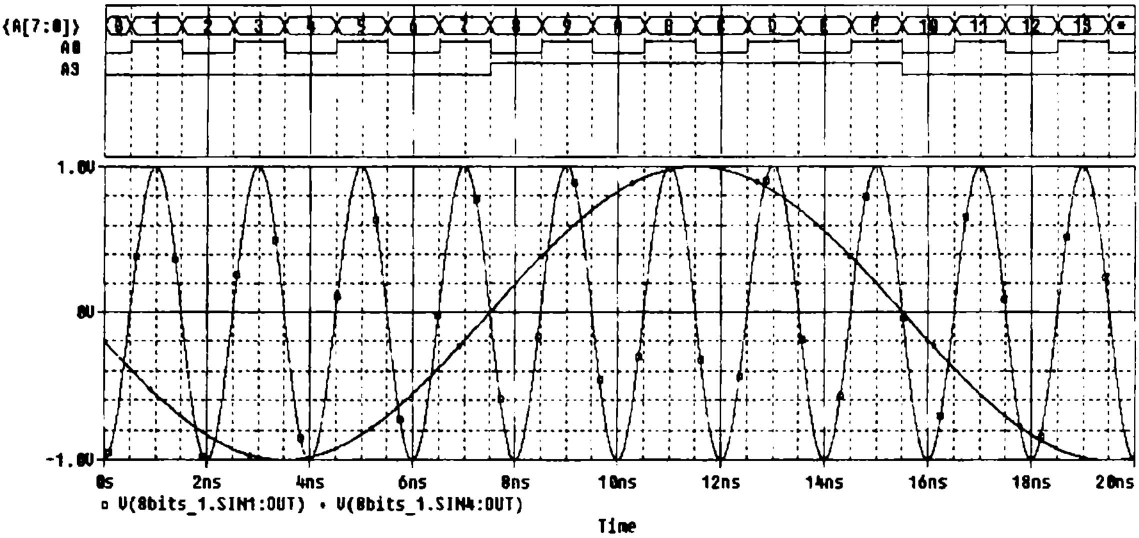

Fig 3. Transforming sine waves in digital bits.

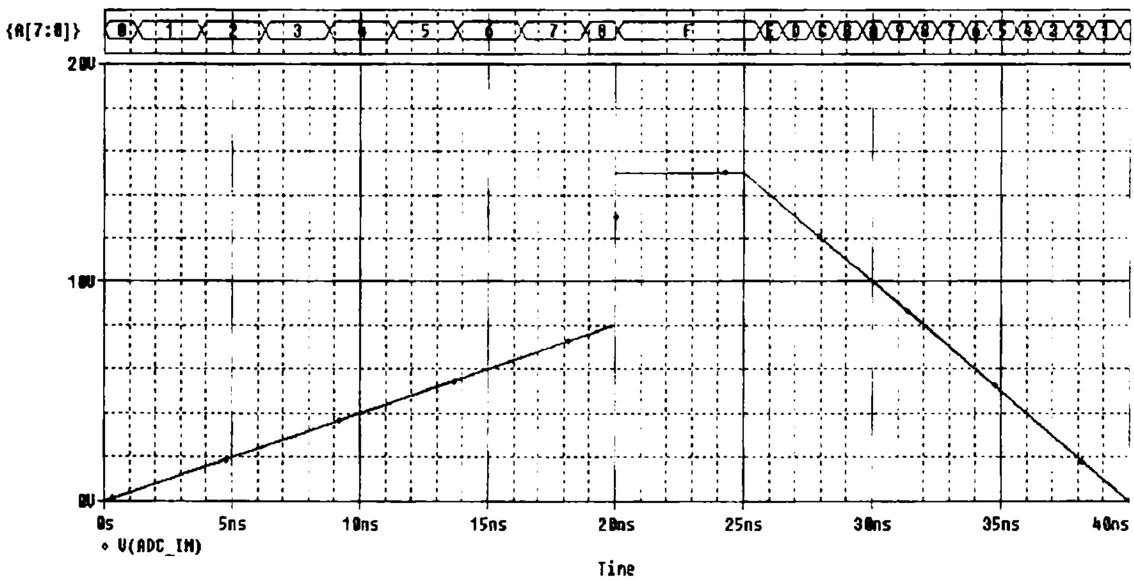

Fig 4. Running the counter with PWL source.

classic digital counters with latches if we apply clock signals that have very high frequency, the counter gives wrong codes, or undefined state. The proposed counter just skips some codes if we didn't drive correctly but the codes from the output can have a correct value after the time step is smaller than the period corresponding to a state of the counter.

IV. RUNNING THE CODES IN DC OR PARAMETRIC SWEEP ANALYSIS.

If we need to know the behavior of the mixed circuit just for some codes, at the ADC_IN we have put a parameter N that can be sweep in DC analysis. If we use linear sweep, then the increment used in this analysis is the increment of the counter. It is very easy to set the start value for the counter and the increment. Also is possible to sweep the codes to a logarithmic scale, or sweep just some value from a list.

If we run a parametric sweep we obtain in different charts the behavior of the circuit for different codes Let suppose that we have a circuit depending from codes (fig.5). Sweeping the digital code for the interest values we can run a transient analysis with any final time for each code fig.6. This is an important feature because it is very easy to pick the codes where we want to run another spice analysis than transient. The bias point can be saved for one ore more particular codes for future use with Load bias point command. If we sweep at the input of this math converter a string with random values in a range, we can obtain a set of random codes in a range. In this way we obtain a pseudorandom spice code generator very useful in the simulation of complex mixed analog digital circuits

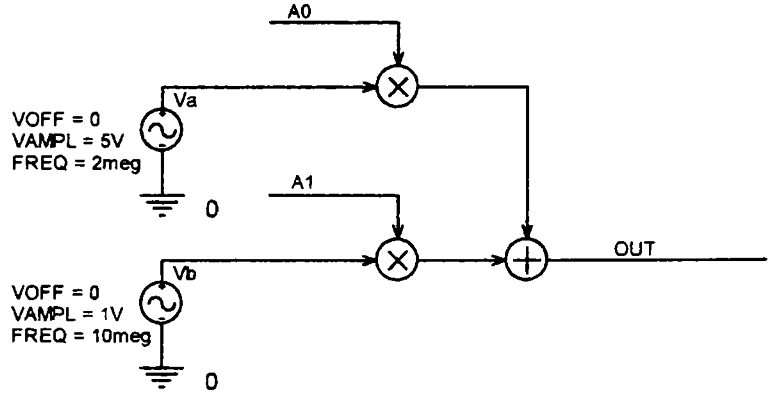

Fig 5. Test mixed analog digital circuit.

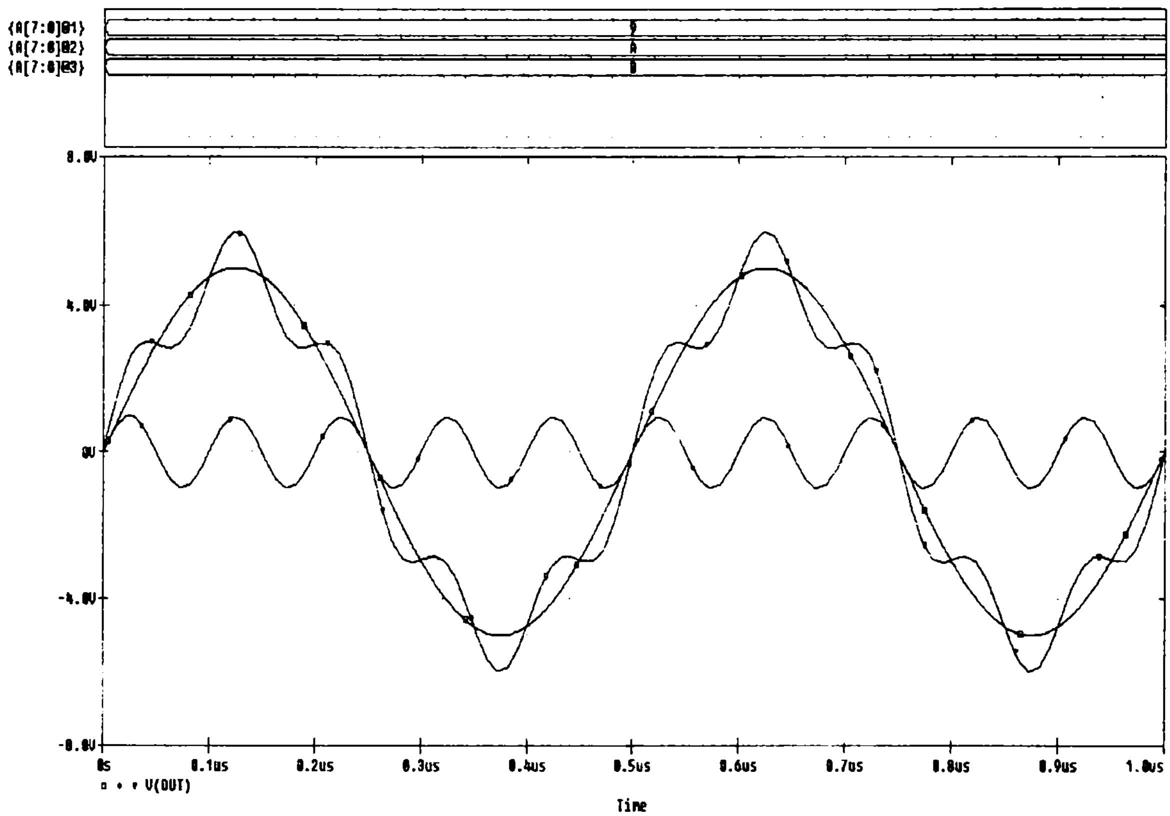

Fig 6. Transient analog analysis for different codes

V. CONCLUSIONS

The advantages of this mathematical circuit in spice simulations open new frontiers in mixed analog and digital simulations, because it is very easy to generate any digital code without transient simulations, and for this code we can run almost any spice analysis. It is very useful also because of the small time spent for simulations. Another purpose of this circuit is to help in modeling complex analog digital circuits [4]. The circuit is very useful for schematics or subcircuits where the relation between digital codes and analog values is not monotonic.

REFERENCES

[1] A. Vladimirescu, Spice, Editura Tehnică, Bucureşti 1999.
[2] http://www.orcad.com/
[3] http://www.ecircuitcenter.com/Circuits/ABM1/ABM1.htm
[4] http//www.sigda.org/Archives/ProceedingArchives/Dac/Dac96/papers/1996/dac96/pdffiles/16_4.pdf