Behavioural Library Development and Documentation

N. Milet-Lewis, S. Snaidero**, Y. Hervé**, G. Monnerie*, D. Geoffroy*, A. Fakhfakh*, H. Levi**

(*) Laboratoire IXL - Université Bordeaux 1
milet@ixl.u-bordeaux.fr

(**) ENSPS - ERM/PHASE - Strasbourg
herve@erm1.u-strasbg.fr

ABSTRACT

This paper deals with the development of standard behavioural libraries and focuses on the documentation. The final objective is that the libraries can be effectively included in CAD environment, easily and naturally used by circuits and systems designers. The complete model development procedure is explained and is performed to generate a VHDL-AMS behavioural library dedicated to frequency synthesis applications.

I. INTRODUCTION

Today, the design flow for analogue and mixed integrated systems doesn't achieve the same automation degree as the digital design flow. In 1999, VHDL-AMS advent [1] seemed to be very promising but in 2002 too less analogue designers have really integrated behavioural modelling and its advantages in a hierarchical design approach. In fact, the time and the experience needed to develop a behavioural model are not negligible and this task can't often be supported by the designer himself. Especially during the Top-Down design phase, libraries of standard behavioural models must be a part of the EDA environment. As the designer is not the "modeller", these models have to be precisely documented and qualified for an application or a class of application.

This paper proposes a systematic procedure to develop and document a behavioural model library. This procedure has been performed for developing a VHDL-AMS library of models dedicated to frequency synthesis applications [2].

II. LIBRARY DESCRIPTION

The content of the VHDL-AMS library is detailed in Table 1.

All fundamental blocks for frequency synthesis are present ; various structures from classical Phase-Locked-Loop (PLL) to fractional frequency synthesiser can be simulated. The second column of Table 1 indicates the model description level which is either purely behavioural or structural. In the structural description, behavioural models are instantiated and connected together.

Model name	Description level	Comment
Phase-Frequency Detector (PFD)	Behavioural	A purely digital circuit that delivers two signals Up and Down which vehicle the phase difference between the two input waves.
Charge Pump (CP)	Behavioural	Combines the signal Up and Down to delivers a unique current source to the loop filter.
Loop Filter (LF)	Behavioural	Computes the average of the charge pump current, giving thus a voltage level proportional to the phase difference between the two PFD inputs.
Voltage-Controlled Oscillator (VCO)	Behavioural	Delivers a voltage square wave which frequency varies around a carrier frequency, proportionally to the input level.
Frequency Divider by N or N+I (FD_N or FD_N_N+I)	Behavioural	A purely digital circuit that divides the input frequency by a fixed integer N in the case of FD_N, or by either N or N+I (dual modulus pre-scaler) in the case of FD_N_N+I.
ACCumulator (ACC)	Behavioural	Associated to a dual modulus pre-scaler (FD_N_N+I), it controls the division factor.
Phase-Locked-Loop (PLL)	Structural	(PFD + CP + LP + VCO + FD_N)
Fractional Phase-Locked-Loop (FPLL)	Structural	(PFD + CP + LP + VCO + ACC + FD_N_N+I)

Table 1 : Library content

III. LIBRARY DEVELOPMENT

This part describes step by step the overall model development procedure.

I.1. Modelling task

III.1.1. Introduction

Two modelling approaches are possible, depending on which design phase we are in. In the Top-Down design phase, the transistor implementation is unknown and the designer handles functional blocks, in order to choose the optimal architecture. The behavioural models used in this phase are generic models, that are developed by studying the functionality of a class of circuits : that is what we call *the functional approach*. In the Bottom-Up design phase, the previous behavioural models may be used with extracted parameter values or may have a “refined” description, based on the transistor-level implementation : that is what we call *the schematic approach* [3]. The present work focuses on the first situation, developing generic models used in the Top-Down phase.

III.1.2. General procedure

Here, the background of the model development is the circuit functionality. In our procedure, a model architecture is systematically decomposed in 3 blocks, corresponding to 3 steps in the behavioural description. This is illustrated by Figure 1. The model communicates with its environment with input/output ports and generic parameters allowing to adjust the model characteristics.

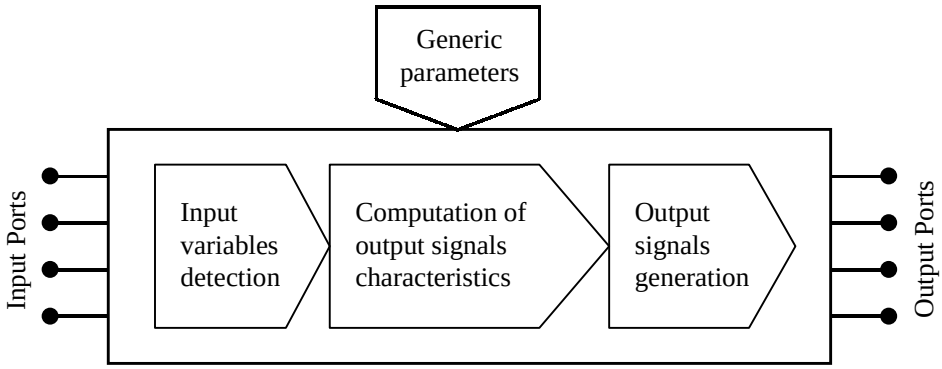


Figure 1 : Fundamental structure of a behavioural model

Concerning the behavioural description, three steps have to be considered.

Step 1 : Input variables detection

It consists in capturing information from the input ports and used to compute the characteristics of the output signals. In our application domain, we often need to capture the following input variables : voltage or current at one port; rising / falling edge, pulse duration, frequency.

Step 2 : Computation of output signals characteristics

This is generally the main part of the model, in which the output signal characteristics are calculated from the input variables and the generic parameters. These characteristics may be : level or amplitude, frequency, delay time, rise/fall time or whatever describes the output wave.

Step 3 : Output signals generation

Two cases can be considered.

The output signal can directly be related to the input signal, like for an amplifier or a filter. In this case, the circuit functionality and the output signal generation are described in the same time and the Step 2 is bypassed :

$V_{out} = AV_{in}$ for an ideal amplifier, $\tau \frac{dV_{out}}{dt} + V_{out} = AV_{in}$ for a first order filter …

In the other case, the output signal depends on characteristics of the input signal and not directly on the input signal. Independent sources also enter in this category. In our application domain, it consists in generating a periodical wave, sinus or square wave.

III.1.3. Illustration

Instead of giving VHDL-AMS code for each model of Table 1, we can explain the modelling task of each behavioural model, following the previous 3-steps procedure. This is summarised in Table 2.

Model entity	**Step 1**	**Step 2**	**Step 3**
in1, in2, PFD, up, down	Detection of an rising edge on *In1* / *In2* ports	Discrimination between the 3 cases : - *In1* / *In2* synchronised - *In1* delayed versus *In2* - *In2* delayed versus *In1*	*Up* and *Down* levels assignment (*high* or *low*), in the 3 cases of Step 2.

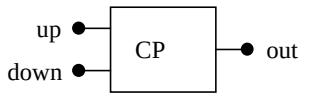
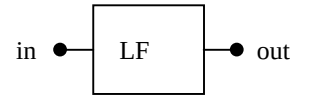
	Detection of *Up* and *Down* signals.	Transposition of Up and Down to analogue currents ; computation of the difference.	Generation of I_{out}.
	Detection of V_{in} and I_{in}.	Direct computation of V_{out} which is the solution of a linear differential equations set.	
	Detection of V_{in}	Computation of *Out* frequency, hence *Out* period.	Generation of a square wave with the corresponding period.
	Detection of *In* period (T_{in}).	Computation of *Out* period from T_{in} and N.	Generation of a square wave with the corresponding period.
	Capture of *Clk* state.	Count of K periods in a T-periods cycle. Reset of the counter.	Assignement of *Carry* sate, following to the counter state.

Table 2 : Decomposed modelling procedure for the library components

I.2. Documentation task

This task is considered as important as the previous one ; the question here is the “re-use” of the model. In fact, it has to be noted that, like in program development, the model can certainly be used and re-used, eventually adapted by the original model developer but may not be by other persons. As a consequence, if the designer is not the “modeller”, the model must have a reliable documentation to be effectively used in the design flow.

Each model developed in this library is associated to several documents, arranged as follows, on Figure 3. The *model_name* directory contains one sub-directory (*model*) dedicated to the model description and many others sub-directories (*model*, *test1, test2, …, testn*) describing as fully as possible the test-benches needed to characterise the model behaviour. The *testn* directory contains the following files :

- testn.txt is a text file that descibes the testbench (what are the sources, loads, stimuli and analysis) and its objective
- testn.vhd is the source file of the model (here in VHDL-AMS)
- testn.cmd is the file containing the ananlysis definition (for Spice-like simulators)
- test.ps is a postcript file that shows the simulation curves for this testbench.

One important file to be generated during this task is the *data-sheet.pdf* file which format imitates the data-sheets forms of commercial integrated circuits. As an example, the accumulator model data-sheet is given in Annexe. Models performances and limitations are clearly exposed, also simulator compatibility, and, in general, all information making the model “easy to use”.

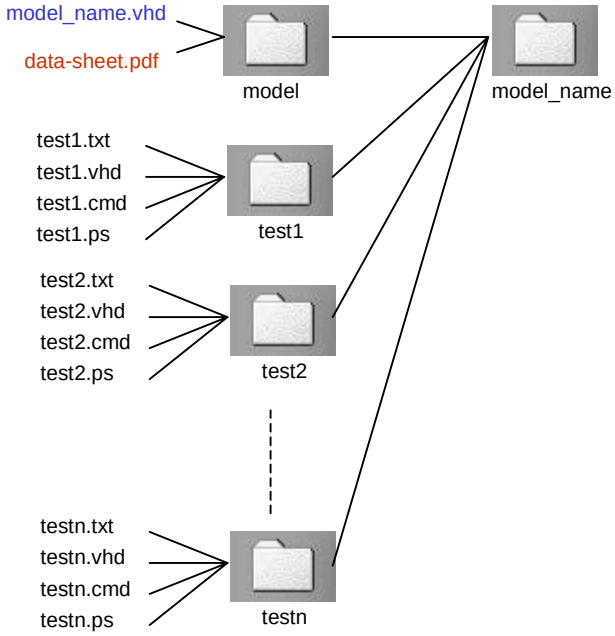


Figure 3 : Structure of the model documentation directory

I.3. Simulation results

Concerning the performances of the developed models, a frequency synthesiser has been simulated which corresponds to the UMTS2000 specifications : a frequency band from 1,92 GHz to 1,98 GHz with 12 channels to synthesise. This frequency synthesiser is composed of a fractional PLL (FPLL) which drives a synchronous oscillator (SO). The SO multiplies the PLL output frequency by 6. The PLL structure is depicted on Figure 2 ; each block comes from the VHDL-AMS library.

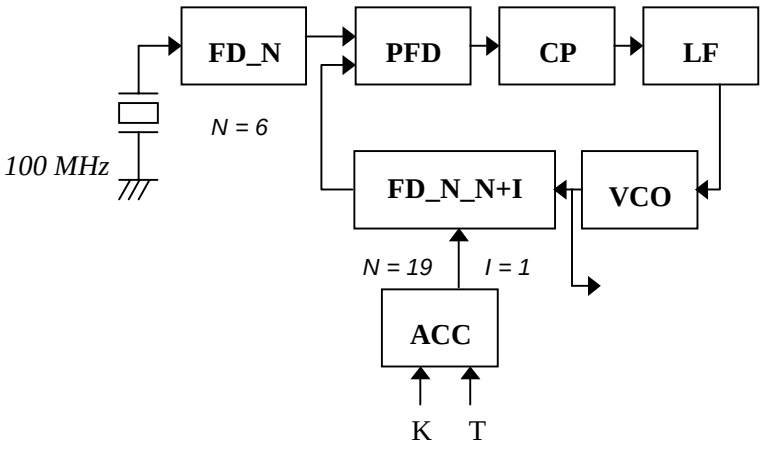


Figure 2 : Fractional PLL structure

Figure 3 shows some simulation results. The K parameter permits to select the PLL output frequency ; this parameter is varied as a stimulus. The first curve shows the variation of the mean ratio of fractional division, which linearly depends on K. The second curve represents the filter output reaction and the last curve the output period of the PLL. This simulation was performed over 60 µs which corresponds to about 120000 output periods.

The CPU time, on a E220R sun server, is only 12 s for the behavioural description, while it takes many hours for the transistor-level similar simulation.

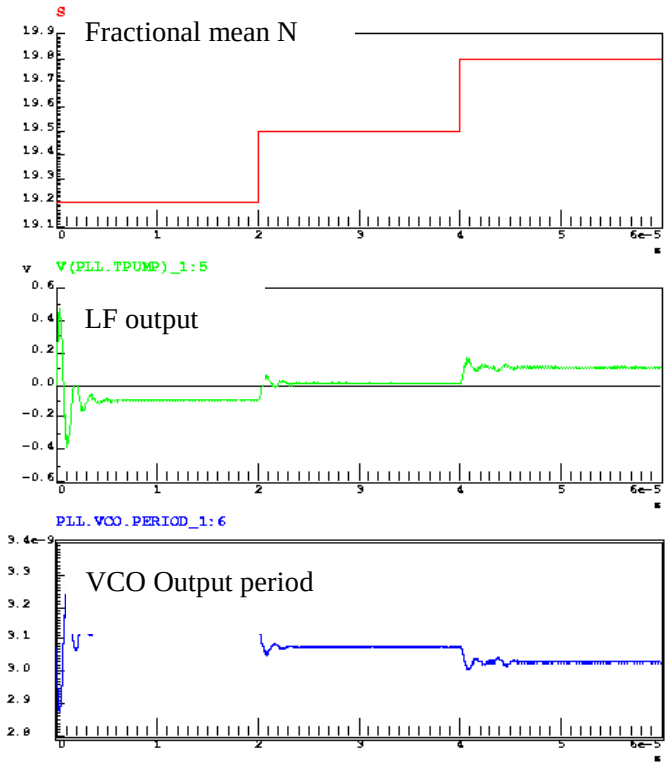


Figure 3 : PLL simulation results

IV. CONCLUSION

The work presented in this paper tries to response to the insufficient effective integration of behavioural modelling in the design flow of analogue and mixed integrated systems. Available model libraries are often incomplete or insufficiently documented, and the circuit designer has often no time and/or no sufficient experience to develop or adapted some models. That is why we propose libraries of standard behavioural models to be included in the CAD environment. The described library is composed of all blocks used in the frequency synthesis domain. A comparison between behavioural and transistor-level simulations proves the efficiency of the developed models. The complete model development procedure is described and a special effort has been made on the model documentation, in order to make a really "easy-to-use" library.

This modelling, documentation and qualification procedure is now well defined and will be adopted for the development of others libraries, for example for optronics applications. These models are part from the "BEAMS Open Library" [3], soon available and free on internet.

V. REFERENCES

[1] IEEE Standard VHDL Analog and Mixed-Signal Extensions, IEEE Std 1076.1 – 1999.

[2] N. Milet-Lewis, G. Monnerie, A. Fakhfakh, D. Geoffroy, Y. Hervé, H. Lévi, J-J. Charlot, *A VHDL-AMS library of RF blocks models*, BMAS'01, IEEE International Workshop on Behavioural Modelling and Simulation, Oct. 11-12, 2001, Santa-Rosa, USA.

[3] A. Fakhfakh, Contribution à la modélisation comportementale des circuits radio-fréquence, PHD Thesis, Bordeaux 1 University, France, Jan. 2002.

[4] BEAMS, Behavioural modelling of Analogue and Mixed Systems, a non-lucrative association for the promotion of behavioural modelling and simulation with HDL languages, http://www.beams.asso.fr

VI. ANNEXE : ACCUMULATOR DATA-SHEET (PARTS)

Model's name : DIGITAL ACCUMULATOR

Short description : This device generates a periodic output signal of 'bit' type. The period of the output signal is a generic number of input periods. The duty-cycle is defined by a generic number of input periods that determine the low level duration time of the output signal.

Archive's name : digital_accumulator.zip

….

Author's name (s) : G. MONNERIE

Validated by : S. SNAIDERO

Repository date : March 2002

Bibliographic reference(s) : N/A

Name of the project : PLL Modeling

Computer type : SUN WORK STATION

Operating system : SOLARIS

Version : 8

Simulation tool : ADVanceMS 1.1_1.1

	Name	**Version**
Entity name :	digital_accumulator	1.0
Architecture :	behavior	

Function description :

This model implements an accumulator which provides a square digital signal. The shape of the signal is managed by the *T* parameter that indicates the whole number of the input signal periods corresponding to the output signal period and its duty-cycle (time when the signal is at '1' divided by the signal period) by the *K* parameter that represents the whole number of input signal periods when the output signal is '0'.

The digital construction of this model implies that it can only be use for an overview of the component behavior. This model can't be use for small oscillation or slew rate modeling. As a counterpart, it is far more faster than an equivalent analog model.

Validity domain :

$0 \leq K \leq T \leq \text{Integer'high}$ (This rule is not checked by the model).

The model is synchronizing on the rising edges of the input signal.

Model interface :

Generics	*Name*	*Type*	*Description*
	T	Integer	Number of in-periods corresponding to an output cycle
	K	Integer	Number of in-periods when the output signal is at ‘0’

Ports	*Name*	*Type*	*Class*	*Mode*	*Description*
	clock	Bit	Signal	In	Input signal
	carry	Bit	Signal	Out	Output signal

Model structure :

Bloc-diagram :

N/A

Hierarchy :

Model name	*Version*	*File names*
N/A		

Package description :

No required packages.

Format of associated files :

*.vhd (test files), *.cmd and *.dou

Absolute maximum ratings :

N/A

Recommended operating conditions :

No intrinsic limitations.

Electrical characteristics :

The input and output of the model are two digital signals of ‘bit’ type.

Switching characteristics (time/frequence)

Switching time : 1 delta-time (tplh = tphl = 0 ns).

Max clock frequency : 500 THz (T_{min} = 2 fs) with a 50% duty-cycle clock signal.

Parameter to identify :

No parameter to identify.

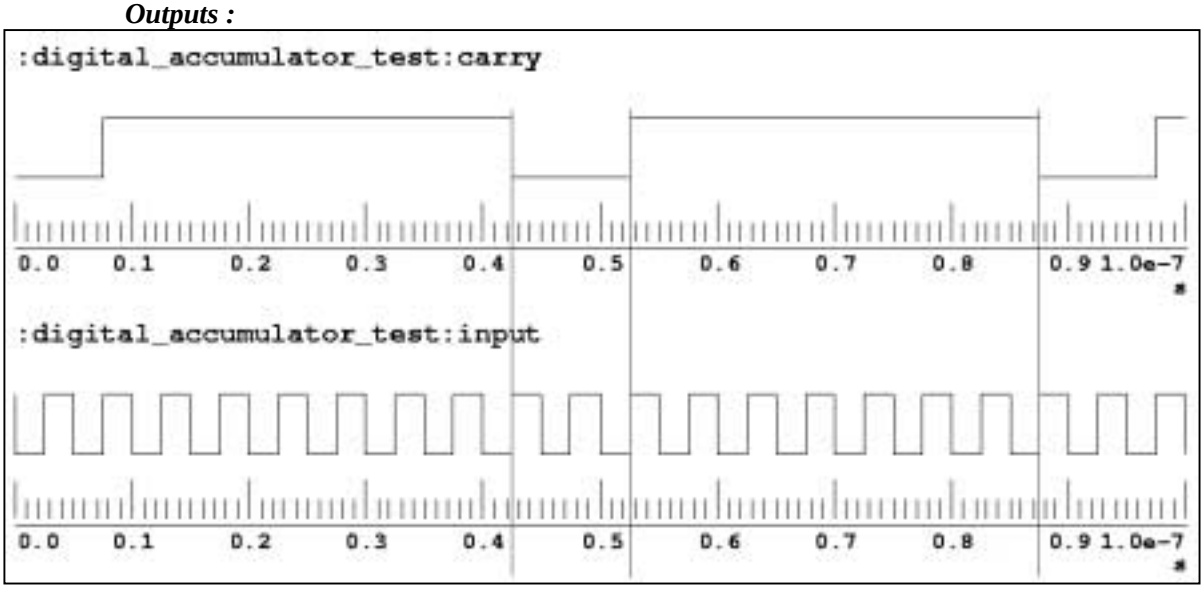
Tests description :

Test 1 : Standard work conditions

Input : Square signal with a period of 5 ns and a duty-cycle of ½.

Settings : T = 9, K = 2.

Outputs :



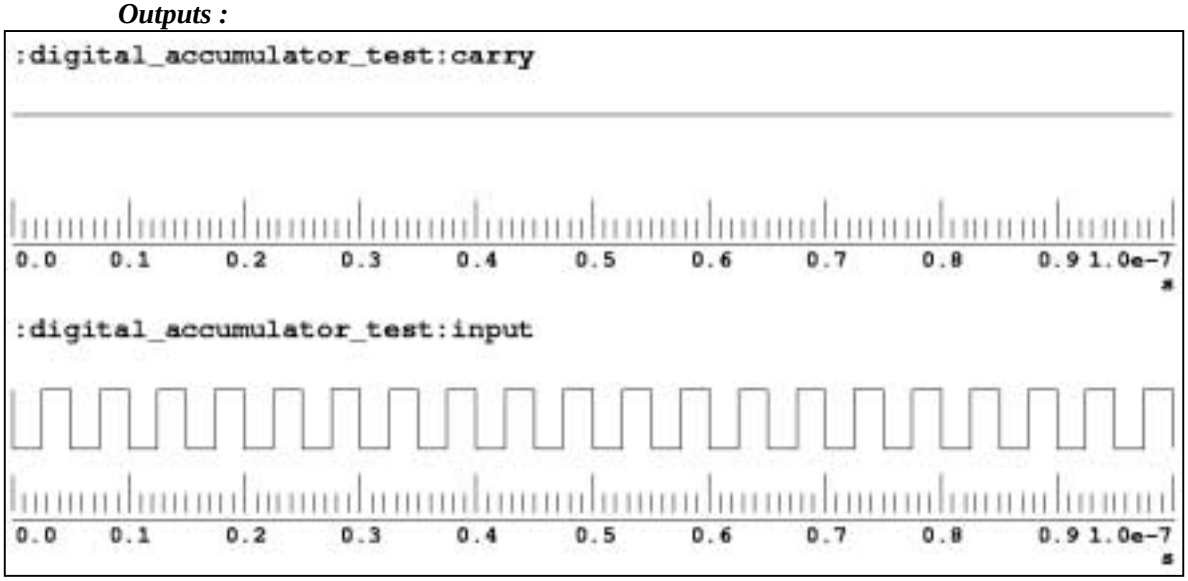
The period of the output signal is 9 times the input signal one and there are 2 consecutive in-periods among the 9 where the signal value is ‘0’.

Test 2 : Extreme work conditions

Input : Square signal with a period of 5 ns and a duty-cycle of ½.

Settings : T = 9, K = 0.

Outputs :



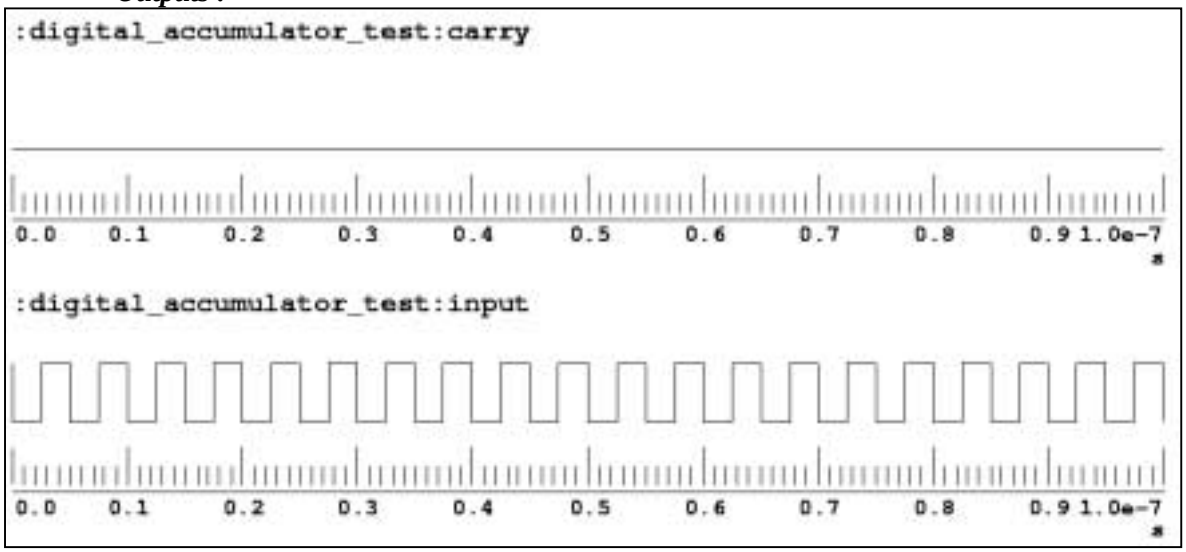
The signal is constantly at ‘1’.

Test 3 : Extreme work conditions

Input : Square signal with a period of 5 ns and a duty-cycle of ½.

Settings : T = 9, K = 9.

Outputs :



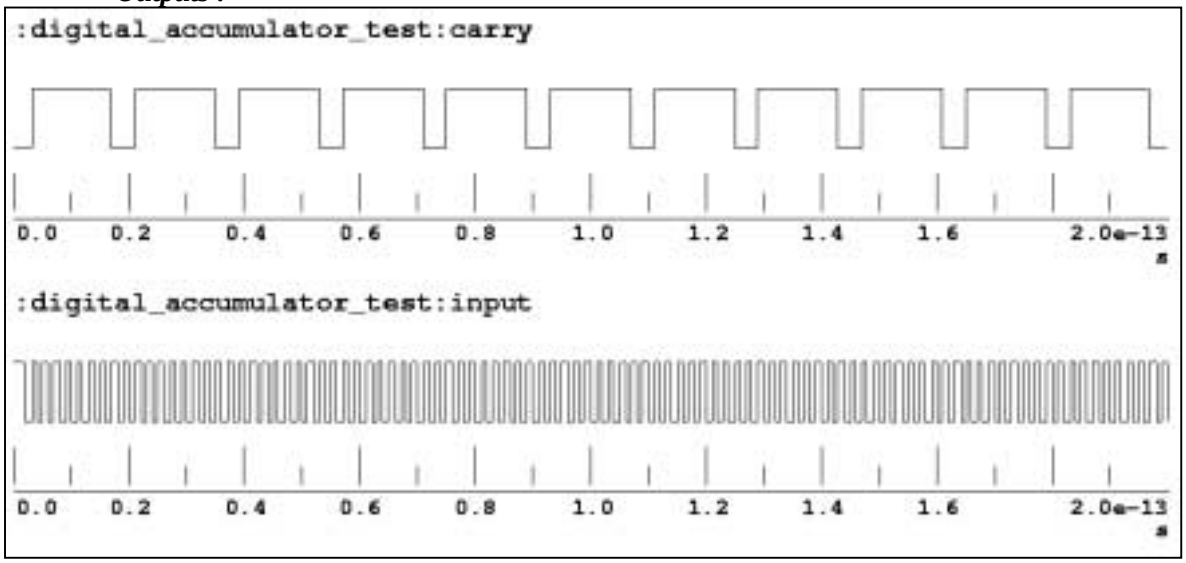
The signal is constantly at ‘0’.

Test 3 : Extreme work conditions

Input : Square signal with a period of 2 fs and a duty-cycle of ½.

Settings : T = 9, K = 2.

Outputs :



The period of the output signal is 9 times the input signal one and there are 2 consecutive in-periods among the 9 where the signal value is ‘0’.

Application notes :

No specific remarks.

Known bugs and limitations :

- No bug listed
- No intrinsic limitations

History :

Old Version	New Version	Modified by	Change(s)
0.9_1	0.9_2	S. SNAIDERO	Code improvement

Archive content :

digital_accumulator

+ Model

| - digital_accumulator.vhd

| - data-sheet.doc

+ Test1

| - digital_accumulator_test1.vhd

| - digital_accumulator_test1.txt

| - digital_accumulator_test1.cmd

| - digital_accumulator_test1.ps

+ Test2

| - digital_accumulator_test2.vhd

| - digital_accumulator_test2.txt

| - digital_accumulator_test2.cmd

| - digital_accumulator_test2.ps

+ Test3

| - digital_accumulator_test3.vhd

| - digital_accumulator_test3.txt

| - digital_accumulator_test3.cmd

| - digital_accumulator_test3.ps

+ Test4

- digital_accumulator_test4.vhd
- digital_accumulator_test4.txt
- digital_accumulator_test4.cmd
- digital_accumulator_test4.ps