Investigating opportunities for hardware realization of transfer functions

Krasimira Filipova, Vladimir Yankov, Filip Filipov, Yordan Kralev, Tzvetomir Dimov

Technical university of Sofia, Sofia, Bulgaria, kfilipova@abv.bg

***Abstract**: In the paper we use a model of real-world process, represented with well-posed transfer function (TF) - W(p). It is used to build a Simulink model of the process. It was necessary the TF to be discretized with appropriately chosen sample time T0 and discretization method. As a result we get a discrete transfer function W(z). These two models, described with respective TFs, are simulated in Simulink, and after that we estimate their functional nearness. For generation of hardware description in HDL for the model it's used Simulink HDL coder toolbox. It contains a library of Simulink blocks, which have corresponding HDL representation. To use that library, we represent discretized TF in canonical form (a graph of sum and gain operations and unit-delays). From resulting Simulink scheme, automatically is generated a VHDL and testbench code. To verify the correspondence between generated code and the functionality described with Simulink model, the testbench code is simulated in integrated environment Xilinx ISE Design Suite. There the values of each signal with time can be visualized. The next stage is implementation of generated code in Xilinx environment.*

***Keywords:** MATLAB; Simulink HDL coder; VHDL;HDL code generation; testbench; representation of linear systems; discretization; top – down synthesis; Xilinx ISE Design Suit.*

1. INTRODUCTION

Evolution of technologies in electronics and the necessity of designing devices for different spheres of human activity, leads to creation of integrated circuits and systems with complicated functions, high degree of integration and high operating frequencies. Also there is increasing requirements for precision and reliability of these devices. Utilization of high-level language for description of certain types of hardware (VHDL) and the environment of Xilinx ISE, provide a degree of automation of design process. Software product MATLAB together with Simulink HDL coder toolbox give an opportunity for automatic generation of HDL and testbench code. In the paper we will present a cooperation of these environments. It will be demonstrated with an example to the level of technological scheme.

2. BUILDING A SIMULINK MODEL

Languages originate because of necessity of equivalent representation of real structures in certain modelling environment, being a mediator for analysis or synthesis of these real structures. Simulink and VHDL can be thought like two separate formal languages (L_s and L_h), defined over alphabets (A_s and A_h). A formal language ($L_a = (L \subset W, \gamma : L \to 2^W)$) is a subset of $W = \bigcup A^k, k = 1..\infty$ (set of words over A), and representation γ showing parts of every word. Every Simulink model is: a set of blocks B, sets of input and output ports ($B_I, B_O, c_i : B \to B_I, c_o : B \to B_O$), set of connections between them ($C \subset B_O \times B_I$). Every block is an element of parameterised set of blocks ($\rho : B \to P \subset \bigcup R^k, k = 1..\infty$). This structure is defined in mdl file.

Functions of Simulink model are defined by $\sigma : B \times P \to S$. S – set of dynamical systems, P – set of parameters, B – set of blocks.

$$s \in S \Rightarrow s = (X, U, Y, f(x,u,k), g(x,u,k)), x(k+1) = f(x(k), u(k), k), y(k) = g(x(k), u(k), k)$$

This shows that for every block, with fixed parameters there is associated dynamical system. It can be defined over number of time lines, so part of it to be continuous, part discrete, eventually with number of different sample times. Dynamical systems, for blocks of standard Simulink library are described with a lower-level language (MATLAB, C, Ada). Therefore exists a map $\sigma_c : S \to L_c$ between the set of systems S and set words of L_c (Lang C, for ex.). This description is used for simulation of the model or for lower-level language code generation. Also there exist correspondence between set of blocks and set of equivalence classes in $\sigma(\{\sigma(b, \bullet)\})$ when equivalence relation is $(b_1, p_1, \sigma(b_1, p_1)) \sim (b_2, p_2, \sigma(b_2, p_2)) \Leftrightarrow b_1 = b_2$. Then every block is identified with some parametric set of dynamical systems.

VHDL is a language for hardware description, suitable for representation of processes and systems in discrete modelling environments. To be possible generation of VHDL code, for a Simulink model has to be known the map $\sigma_{hdl} : S \to L_h$. It associates a dynamical system with a word of L_h . Not for every block of standard Simulink library there exists such a correspondence. For this reason when build a model aimed for VHDL code generation, you have to use blocks of special library reachable by hdllib command.

3. APPROXIMATION OF CONTINUOUS DYNAMICAL SYSTEM WITH DISCRETE DYNAMICAL SYSTEM

A method for description of linear time-invariant dynamical systems is by transfer function (TF) – continuous or discrete. To be able to use Simulink HDL coder toolbox we have to discretize continuous TF and transform it into direct canonical form. It is a graph of sum and gain operations and integrators (unit-delays in discrete case).

3.1. Direct canonical form of continuous TF

Below follows deduction of that canonical form for example transfer function:

$$W(s) = \frac{4s^2 + 38,4s + 51,2}{s^4 + 12s^3 + 37s^2 + 36s + 4} = \frac{Y(s)}{U(s)}$$

$$\left(s^4+12s^3+37s^2+36s+4\right)Y(s) = \left(4s^2+38,4s+51,2\right)U(s)$$

$$Y(s)s^4 = -12s^3Y(s) - 37s^2Y(s) - 36sY(s) - 4Y(s) + 4s^2U(s) + 38,4sU(s) + 51,2U(s)$$

$$Y(s) = -\frac{12}{s}Y(s) - \frac{37}{s^2}Y(s) - \frac{36}{s^3}Y(s) - \frac{4}{s^4}Y(s) + \frac{4}{s^2}U(s) + \frac{38,4}{s^3}U(s) + \frac{51,2}{s^4}U(s) \quad (1)$$

On figure 1 and 2 are shown two possible realization of expression (1). They are put into subsystem blocks (Subsystem 1 and Subsystem 2 on figure 3). In Subsystem 2 is the original transfer function, too. Simulations are done and the figure 4 shows that their step responses are identical. Therefore the linear systems are equivalent.

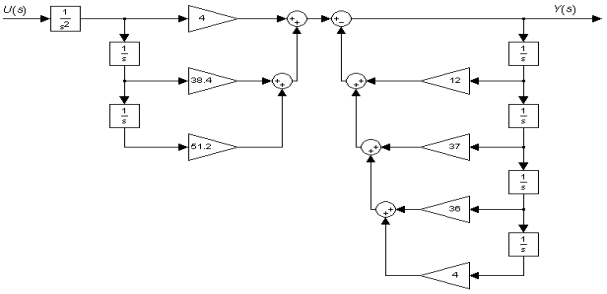

fig. 1 Realization of (1)

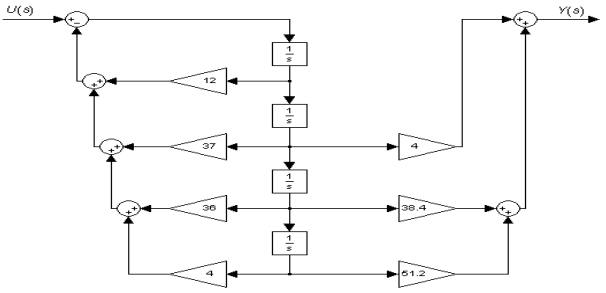

fig. 2 Minimal realization of (1)

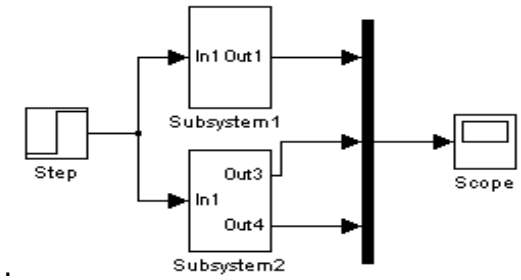

fig.3 Scheme for comparing step responses

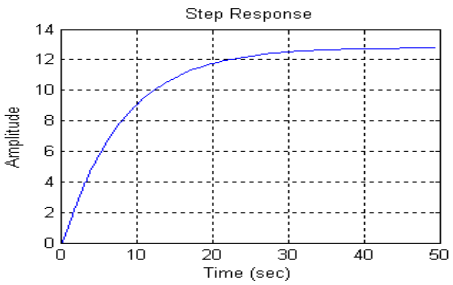

fig. 4 Step responses

3.2. Discretization of continuous system

Discretization is a mapping of set of continuous signals into a set of discrete signals in the way that values of continuous and discrete signals are identical in sample times. This map inducts a map between sets of continuous systems and discrete systems. The choice for sample time T0 is made in order with theorem of Kotelnikov-Shannon. So, for the example system we choose $T_0 = 0.8$ and then discrete TF is:

$$W(z) = \frac{0{,}71776z^3 + 0.17223z^2 - 0{,}1038z + 0{,}00013576}{z^4 - 1{,}309z^3 + 0{,}40794z^2 - 0.03757z + 6{,}7729*10^{-5}} = \frac{B(z)}{A(z)} = \frac{Y(z)}{U(z)}$$

$$b_0 = 0{,}71776, b_1 = 0.17223, b_2 = -0{,}1038, b_3 = 0{,}00013576.$$

$$a_1 = -1{,}309, a_2 = 0{,}40794, a_3 = -0.03757, a_4 = 6{,}7729*10^{-5}.$$

That discrete system is represented in direct canonical form.

$$\left(z^4 + a_1 z^3 + a_2 z^2 + a_3 z + a_4\right) Y(z) = \left(b_0 z^3 + b_1 z^2 + b_2 z + b_3\right) U(z)$$

$$Y(z)z^4 = -a_1 z^3 Y(z) - a_2 z^2 Y(z) - a_3 z Y(z) - a_4 Y(z) + b_0 z^3 U(z) + b_1 z^2 U(z) + b_2 z U(z) + b_3 U(z)$$

$$Y(z) = -\frac{a_1}{z} Y(z) - \frac{a_2}{z^2} Y(z) - \frac{a_3}{z^3} Y(z) - \frac{a_4}{z^4} Y(z) + \frac{b_0}{z} U(z) + \frac{b_1}{z^2} U(z) + \frac{b_2}{z^3} U(z) + \frac{b_3}{z^4} U(z) \quad (2)$$

On figure 5 is shown minimal realization of (2). Simulations show that step responses of continuous and discrete TF are identical in sample times (figure 6). Also the bundle of step response of discrete system is identical with step response of continuous system. This means that sample time T0 is chosen appropriately.

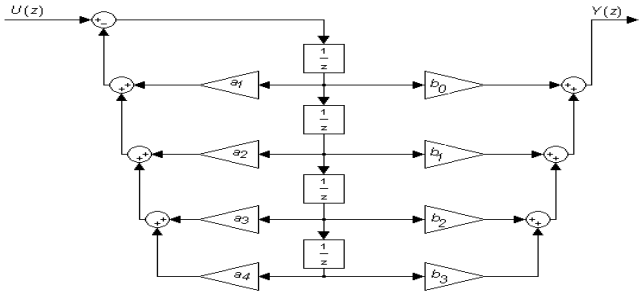

fig. 5 Minimal realization of (2)

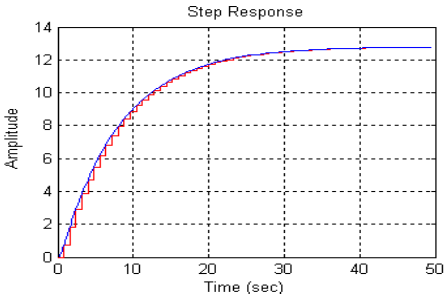

fig. 6 Step responses of continuous and discrete system

4. STEPS FOR EXPORTING MODEL FROM SIMULINK TO XILINX ISE 10 PROJECT NAVIGATOR

1) Representation of discrete transfer functions in direct canonical form and estimation of nearness with originals (by getting the step response for example).

2) Data types of signals have to be chosen correctly. Changing the data types of signals and variables in VHDL code leads to considerable differences in hardware realization. On target platform, for every defined variable or signal, corresponds a set of physical connection lines (wires). With their levels is coded the value of a variable or signal for every instant of time. The coding depends on data type and is specified in standards. Therefore the choice of data type also determines hardware elements used for operations with data. In Simulink the data types used by every block are specified like a block parameter. There can be used standard types for integer or floating-point numbers and used-defined types.

For our concrete TF, values of signals in time are real numbers. We use fixed-point numbers as a data type. For representation of real N with fixed-point number it is used scale factor M and then $N_{FIX} = [N * M], N_{FIX} \in Z$. We use 32 bits to store the integer and scale factor 2^10=1024. That representation is specified in block parameters dialog box with expression fixdt(1,32,10) (fixdt(sign, number of bits, scale factor like power of 2)). With changing the data-type, the properties of modelled system also change. Therefore the correspondence between former discrete system and discrete system with fixed-point numbers has to be checked. The comparison between step responses is shown on figure below.

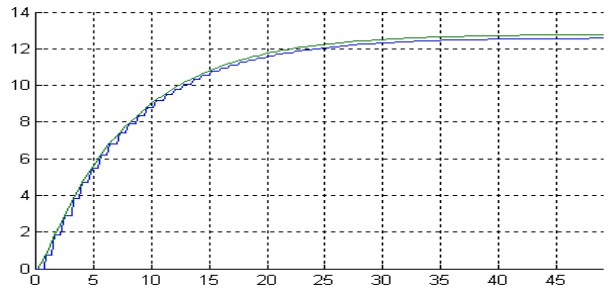

fig. 7 Step responses for scale factor 1024

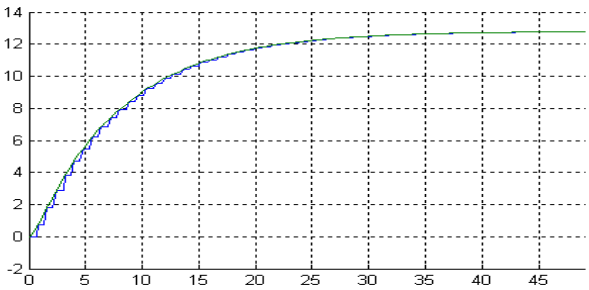

fig. 8 Step responses for scale factor 65536

3) Selection of subsystem for which VHDL code is generated (Menu Simulation/Config. Parameters/HDL Coder page/Generate HDL). Testbench code also can be generated. It is used for validation of generated HDL code.

For our example we have generated HDL and testbench code. In Xilinx Poject Navigator is created new project and generated code is added to it. Below are presented re-

sults of simulation in Xilinx environment (figure 9) and synthesed technological scheme (figure 10).

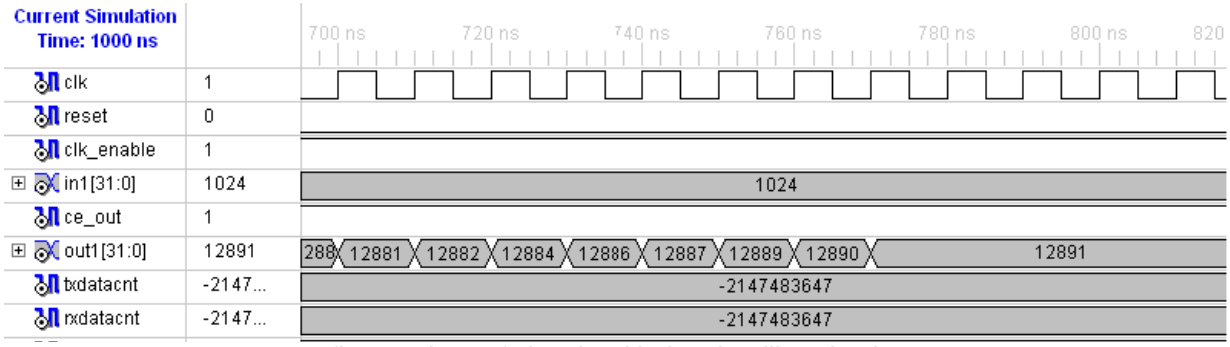

fig. 9 Values of signals with time in Xilinx simulator

From figure 9 we see that final value of output when input is step function is $N_{FIX} = 12891 \Rightarrow N = N_{FIX} / M = 12891/1024 = 12.5889$. From the step response of discrete system (figure 7) we see that final value is 12.5889. Other tests on signals also agree. This means that generated code possesses the functionality of Simulink model.

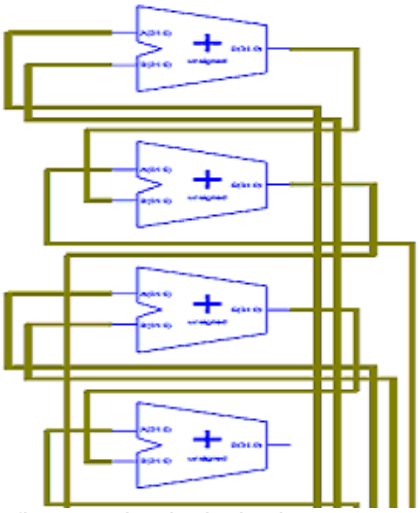

fig. 9 Technological scheme part 1

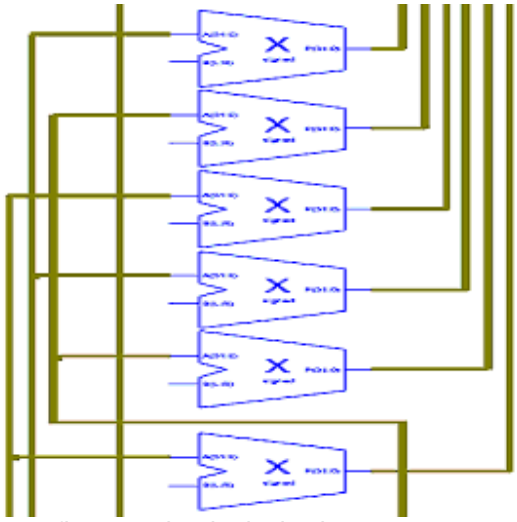

fig. 9 Technological scheme part 2

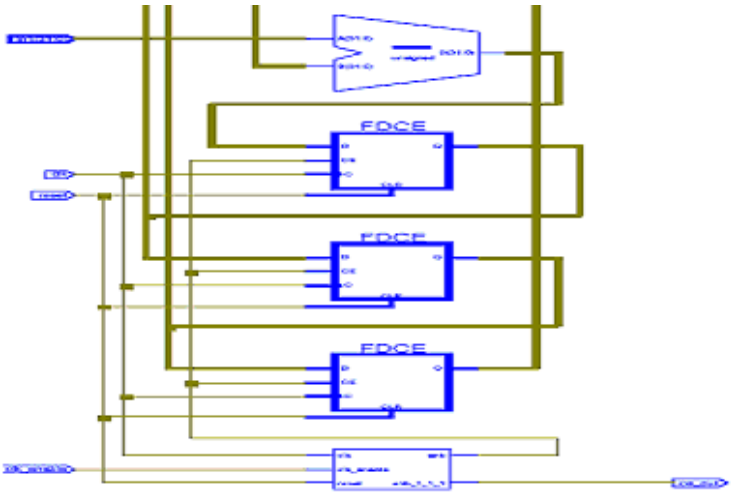

fig. 9 Technological scheme part 3

5. CONCLUSION

It was illustrated that integration of the two development environments is possible, eases and accelerates hardware synthesis. For automatic generation of code is necessary knowledge on MATLAB/Simulink but you don't have to be familiar with VHDL or Verilog in details. A big advantage of usage of Xilinx ISE – Project Navigator for synthesis is automatic selection of appropriate set of elements, connections between them, hierarchy structure for elements, such that to be maximized precision and minimized LUT usage in process of designing a device.

The research results presented in this paper are funded by Technical University – Sofia research project No. 112пд055-8.

6. REFERENCES

[1] Chu, Pong P. (2006), "RTL Hardware Design using VHDL", John Wiley, ISBN 978-0-471-72092-8

[2] Samilagic, Zoran S.(2000) "Digital systems design and prototyping using field programmable logic", Springer, ISBN 978-0792379201

[3] Roth, Charles H. (2007), "Digital systems design using VHDL", CL-Engineering, ISBN 978-0534384623

[4] Brown, Stephen (2008) "Fundamentals of Digital Logic with VHDL Design", McGraw-Hill, ISBN 978-0077221430

[5] Lee, S. (2005), "Advanced Digital Logic Design Using VHDL, State Machines, and Synthesis for FPGA's", CL-Engineering, ISBN 978-0534466022

[6] Kr. Filipova, S. Petrakieva, F. Filipov, I.Costov Hardware Realization of the Control Algorithm in Hydrosystems with FPGA , Proc. International Conference "Computer Science'08", Kawala , pp115-120.

[7] Mladenov, V., Kr. Filipova, S. Petrakieva, B. Dimov, F. Uhlmann, Analysis of Signal Competition in Asynchronous Ultra High-Speed Digital Circuits, Przeglad Elektrotechniczny (Electrical Review**)**, Issue 11, 2007, ISSN 0033-2097, Poland, pp. 197-200.

[8] T. Stoyadinova, Il. Buzov, V. Mladenov , Kr. Filipova , Thomas Ortlepp , I. Panayotov , Development of VHDL-models for transient simulation of complex asynchronous RSFQ circuits Paper ID: 209 , IWK 51 , Ilmenau, 2009.

[9] Kr. Filipova, I. Costov (2004), Design of electronic hardware using VHDL and Petri networks described with the example of Baseband and LMP layers in Bluetooth, IEEE, Catalog N 04EX830, 27th ISSE, Annual School. Lectures, ISBN 0-7803-8422-9, 13-16 may, Sofia.

[10] К. Филипова, Ф. Филипов (2008), ИДЕИ ЗА ИЗПОЛЗВАНЕ НА VHDL ЗА СИНТЕЗ , Intern. Conf. Automatics and Informatics' 08, Sofia, Bulgaria