A Novel Method to Design Analog Circuits Using Simulink

B K Mishra
Principal
Thakur college of Engg. & Tech.
Kandivali (E), Mumbai-101

Sandhya Save
Asst.Professor
Thakur college of Engg. & Tech.
Kandivali (E), Mumbai-101

Rupali Mane
Asst.Professor
Thakur college of Engg. & Tech.
Kandivali (E), Mumbai-101

Abstract

This paper applies embedding knowledge into model based design method, an efficient and cost-effective way to develop embedded systems. The approach is based on a single design description in the graphical Matlab/Simulink environment that is used for FPGA emulation, ASIC design, verification and chip testing. This simulation-based approach gives a better understanding of design alternatives and trade-offs than traditional prototype-based design methodologies. It enables to optimize design to meet predefined performance criteria. The methodology is tested for designing CMOS inverter and the simulation results confirm the efficiency of presented methodology.

Keywords-

Algorithms, ASIC, CAD, Design Aids, Design, Languages, EDA tools, Simulink , SoC, VCO,CMOS inverter.

1. INTRODUCTION

Traditional approaches to implement embedded system design on FPGA and ASIC involves gathering requirements using visual representations or word processors, creating design using various tools like MATLAB ®[8] , C, C++.and generating HDL code, verifying and testing the design . The translation phase is manual and is prone to introduction of human errors. Each translation step, whether manual or automated, requires additional verification to confirm that the original design has been preserved [2]. It is proposed to use Simulink to model, simulate, and analyze dynamic systems because it is familiar to system designers and provides graphical user interface (GUI) for building models as block diagrams. Chip design verification in the manufacturing process has to be conducted in the early stages to eliminate logic errors and thereby reducing costly respins[9]. Logic verification using simulation software may not be the right choice as this process may be extremely slow. In comparison to that Hardware emulation is multiple times faster than software based emulation.

FPGAs - Field Programmable Gate Arrays – are hardware emulation boards that allow good level of customization of the hardware at affordable prices. Using FPGA results in reducing costs and also time to market ratio. A time consuming and expensive redesign of a board can often be avoided through with the use of FPGA [1].

In today's world VCO is one of the major building bock of the many communication systems such as Mobile communication PLL-Phase Lock Loop, Clock generator, etc. The designing a VCO with reduced area meting the all the target the specification is a major challenge . The VCO can be broadly categories as Ring inverter switches from low (Vin=0) to high (V in = VDD), the capacitor discharges through the NMOS transistor. When the input of an inverter switches from high to low, the capacitor charges through the PMOS transistor..[10] The propagation delay occurs when output voltage switches from high to low and low to high and the $\mathbf{t_p}$ corresponding delay and the threshold voltage $\mathbf{V_I}$ is calculated by the following equation listed below[4] [5]. The propagation delay time can be reduced by reducing the value of C_L or increasing the value of $\mathbf{k_p}$ and $\mathbf{k_n}$

$$t_{pLH} = \frac{C_L.V_{DD}}{k_p(V_{DD} - |V_{TP}|)^2} \approx \frac{C_L}{k_p.V_{DD}} \quad (1)$$

$$t_{pHL} = \frac{C_L.V_{DD}}{k_n(V_{DD} - |V_{TN}|)^2} \approx \frac{C_L}{k_n.V_{DD}} \quad (2)$$

$$t_p = \frac{t_{pIH} + t_{pHL}}{2} = \frac{C_L}{2V_{DD}}.\frac{1}{k_p} + \frac{1}{k_n} \quad (3)$$

$$V_I = \frac{V_{DD} - |V_{TP}| + \left(\frac{\sqrt{\beta n}}{\beta p}\right) V_{TN}}{1 + \left(\sqrt{\frac{\beta n}{\beta p}}\right)} \quad (4)$$

Where,

t_{pLH} = propagation delay time for transition from Low to High

t_{pHL} =propagation delay time for transition from High to Low C_L = capacitive load

V_{TP} =Threshold voltage for PMOS

V_{TN} = Threshold voltage for NMOS

$k_p k_n$ =Transconductance for PMOS ,NMOs respectively

$\lambda = \beta_n / \beta_p$ = Aspect ratio

2. MODEL BASED DESIGN APPROACH USING SIMULINK HDL CODER

The flowchart of the design methodology for automation process using Simulink HDL Coder [3] from modeling to FPGA and ASIC implementation is as shown in figure 3.

The model can be designed using blocks and components Simulink, Simelectronics library and after running a simulation observed results can be easily compared with the analytical results. The coder generates VHDL or . Verilog code that implements the design embodied in the model. Usually, a corresponding test bench also can be generated. The test bench with HDL simulation tools can be used to drive the generated HDL code and evaluate its behavior. The coder generates scripts that automate the process of compiling and simulating code.. EDA Simulator Link™ MQ, EDA Simulator Link IN or EDA Simulator Link DS software can be used from the MathWorks™ to cosimulate generated HDL entities within a Simulink model.

The test bench feature increases confidence in the correctness of the generated code and saves time spent on test bench implementation. The design and test process is fully iterative. At any point, the designer can return to the original model, make modifications, and regenerate code.

When the design and test phases of the project have been completed, easily the generated HDL code can be exported to synthesis and layout tools for hardware realization. The coder generates synthesis scripts for the Synplify® family of synthesis tools. The procedure followed to obtain VHDL netlists that can be downloaded to FPGA boards. The HDL Workflow Advisor is an alternative method to generate HDL code

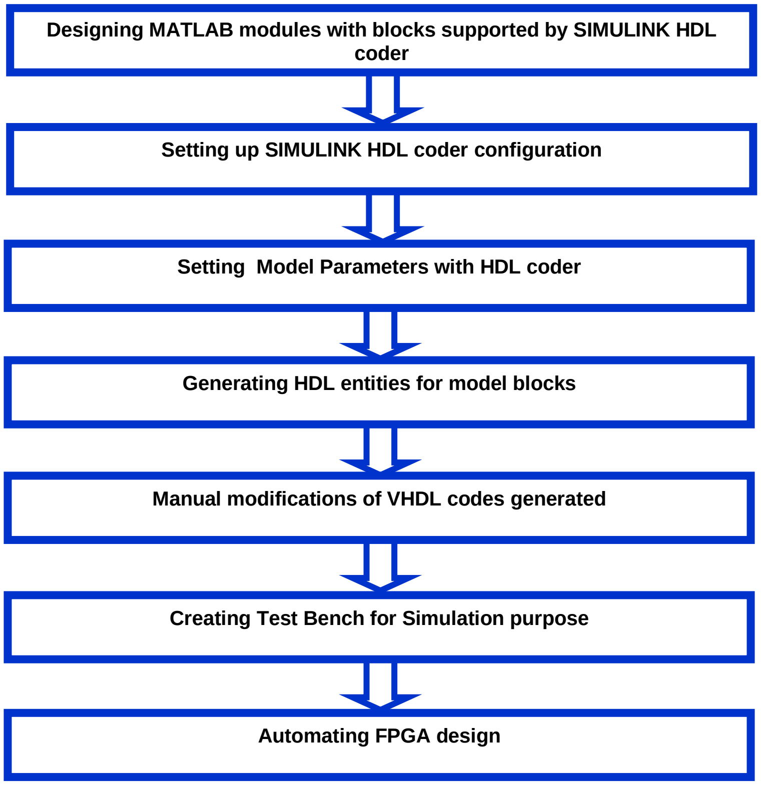

Designing MATLAB modules with blocks supported by SIMULINK HDL coder
Setting up SIMULINK HDL coder configuration
Setting Model Parameters with HDL coder
Generating HDL entities for model blocks
Manual modifications of VHDL codes generated
Creating Test Bench for Simulation purpose
Automating FPGA design

Figure 3. The flowchart of the design methodology using Simlink HDL coder

3. DESIGNING CMOS INVERTER MODEL USING MATLAB/SIMULINK

The proposed idea is implemented using a MATLAB-Simulink toolbox for the simulation and synthesis of the CMOS inverter. The Graphical User Interface (GUI) included in the toolbox allows to navigate easily through all steps of the simulation, synthesis and post-processing of results. The LEVEL 3 spice compitable MOS models are used for Inverter design. Figure 4a and 4b shows CMOS Inverter model using Simulink and Spice compatible components from Simelectronics.

By using this GUI, the user can either open an existing model create a new one in the SIMULINK platform. After running simulation, different Voltage transfer characteristics are obtained by changing the value of W and L of PMOS and NMOS transistors. The input and output waveforms can be seen on the scope component.

4. HDL Code Generation

HDL code can be generated by two options by using the command line interface or by using the GUI.

4.1 Creating environment to generate HDL CODE

Generating HDL code is very important step before implementing design on FPGA or ASIC. It can be generated by linking .mdl file with HDL coder.

Basic environment to generate HDL code can be set up by creating library of all blocks that supports HDL code generation. The utility commands are used for generating such library.

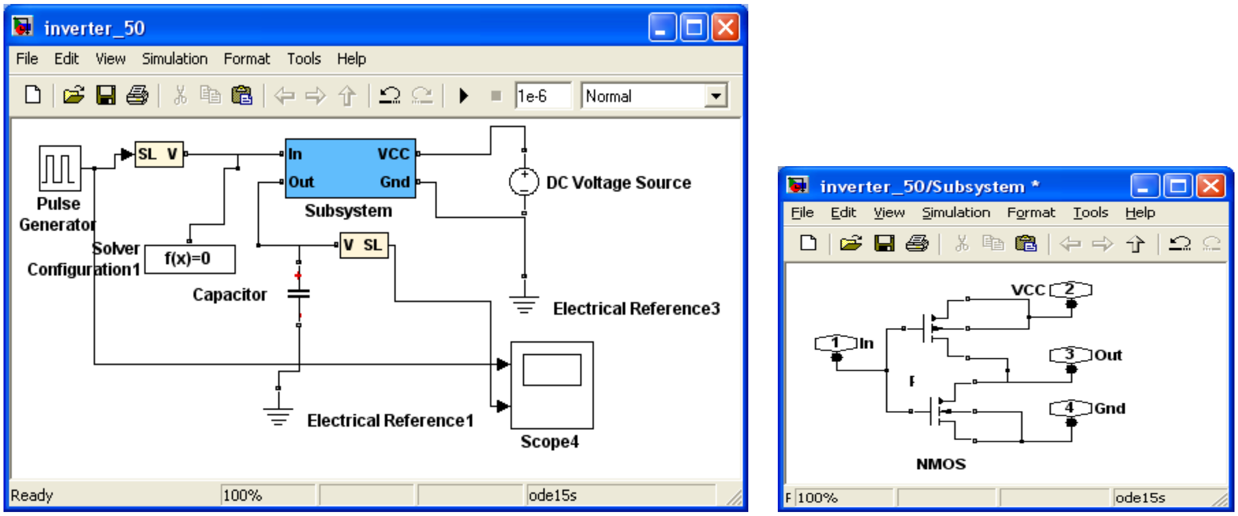


Figure 4 (a.) CMOS Inverter model implemented using Simulink (b.) CMOS Inverter model implemented using Simulink

4.1.1 Creating a Control File using the command line interface

Matlab Simulink .mdl file can be linked to support HDL code by creating a control file which is a Code Generation file. The control file is basically an M-file that needs to be attaches to a model.

4.1.2 Initializing Model Parameters with hdlsetup

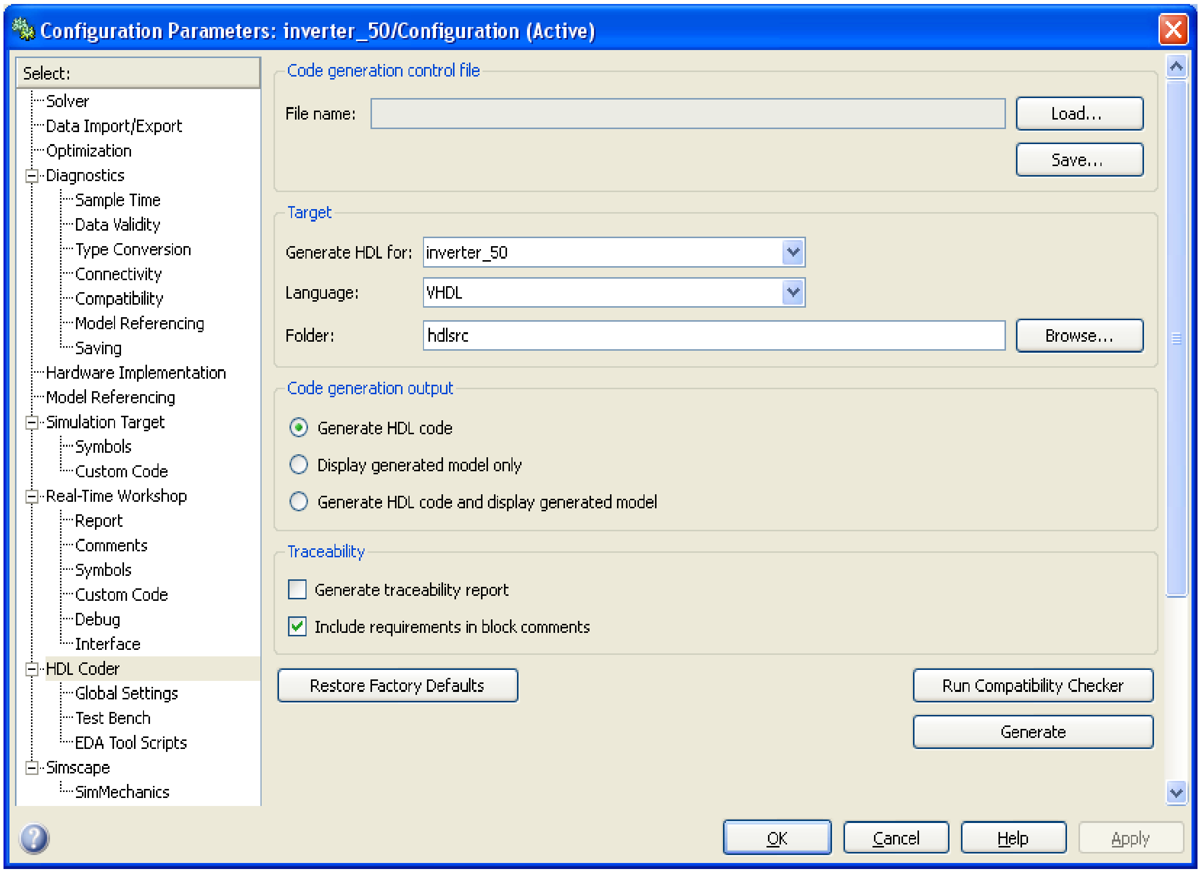
Before generating code, some parameters of the model need to be set up to ensure proper HDL code generation. This parameter initialization is completely model dependent. For example, if continuous input source to be used as a part of model the solver settings need to be changed to continuous mode.

4.1.3 Selecting and Checking a Subsystem for HDL Compatibility

The HDL compatibility checker as shown in figure 5 is used to examine the selected subsystem is fully HDL-compatible

4.2 Generating HDL CODE

HDL code is generated using makehdl command form the command line interface. makehdl also generates script files for third-party HDL simulation and synthesis tools. makehdl can specify numerous properties that control various features of the generated code. In this work, by using defaults makehdl properties the following files are generated in the hdlsrc folder which will be automatically created as subfolder in current working folder.



Configuration Parameters: inverter_50/Configuration (Active)

Code generation control file

File name:

Target

Generate HDL for: inverter_50

Language: VHDL

Folder: hdlsrc

Code generation output

● Generate HDL code

○ Display generated model only

○ Generate HDL code and display generated model

Traceability

□ Generate traceability report

☑ Include requirements in block comments

Figure 5 configuration parameters dialog box

- move_vco.vhd: VHDL code. This file contains an entity definition and RTL architecture implementing the move_vco.

- move_vco.do: Mentor Graphics ModelSim compilation script (vcom command) to compile the generated VHDL code.

- move_vco.tcl: Synplify synthesis script.

- move_vco.txt: Mapping file. This report file maps generated entities (or modules) to the subsystems that generated them

4.3 Manual Modification of the Generated VHDL Codes

The generated codes should be studied carefully. It is possible to be changed according to what the designer need. However, this step can be passed by designing an efficient MATLAB-Simulink model. In this work, few codes only have been slightly modified.

4.4 Generation of Test Benches for Simulation

The test bench is designed to drive and verify the operation of system entity that was generated by HDL coder. It can be generated using test bench generation function. The generated test bench includes:

a. Stimulus data generated by signal sources connected to the entity under test.

b. Output data generated by the entity under test. During a test bench run, this data is compared to the outputs of the VHDL model, for verification purposes.

c. Clock, reset, and clock enable inputs to drive the entity under test.

d. A component instantiation of the entity under test.

e. Code to drive the entity under test and compare its outputs to the expected data.

f. The generated test bench and script files reflects as .tb_vhd,tb_compile.do,tb_sim.doare respectively.

tb.vhd: VHDL test bench code and generated test and output data. tb_compile.do: Mentor Graphics ModelSim compilation script (vcom commands). This script compiles and loads both the entity to be tested and the test bench code (tb.vhd).

tb_sim.do: Mentor Graphics ModelSim script to initialize the simulator, set up wave window signal displays, and run a simulation.

The HDL Workflow Advisor available within Matlab 2010 is an alternative method to generate HDL code.

4.5 Automating FPGA design

Simulink HDL Coder enables to quickly implement Simulink model in Xilinx® and Altera® FPGAs. The HDL Workflow Advisor is a GUI tool that supports and integrates all stages of the FPGA design process, including the following: Checking the Simulink model for HDL code generation compatibility, Generating RTL code, an RTL test bench, and a cosimulation model forming synthesis and timing analysis through integration with Xilinx ISE® and Altera II Providing a resource estimation report and guidance on modifying the model to achieve design constraints. The advantage of this is back annotation of the Simulink model with critical path and other information obtained during synthesis process.

5. RESULT AND DISCUSSION

The one of the results listed is shown in the figure6.The transfer characteristics transfer curve for different values of λ is shown in the figure7. The switching speed due to change of the load is shown in the figure 8.

From the obtained results it can seen that, If there λ is decreased the transition region shifts from left to right however, output voltage transition remains sharp. The switching speed of CMOS transistor is limited by time taken to charge and discharge the load capacitance C_L the output waveform is shown in the Figure 9a and 9b .The effect of change in load on propagation delay is plotted in Figure 10 with λ =1and the relative comparative results are listed in table1.

The relative error observed in the comparative results is can be neglected since the formulae used for calculation are always approximate and the variation of empirical constants parameter changes as per the model whereas in Simulink for this work LEVEL3 MOS models are used.

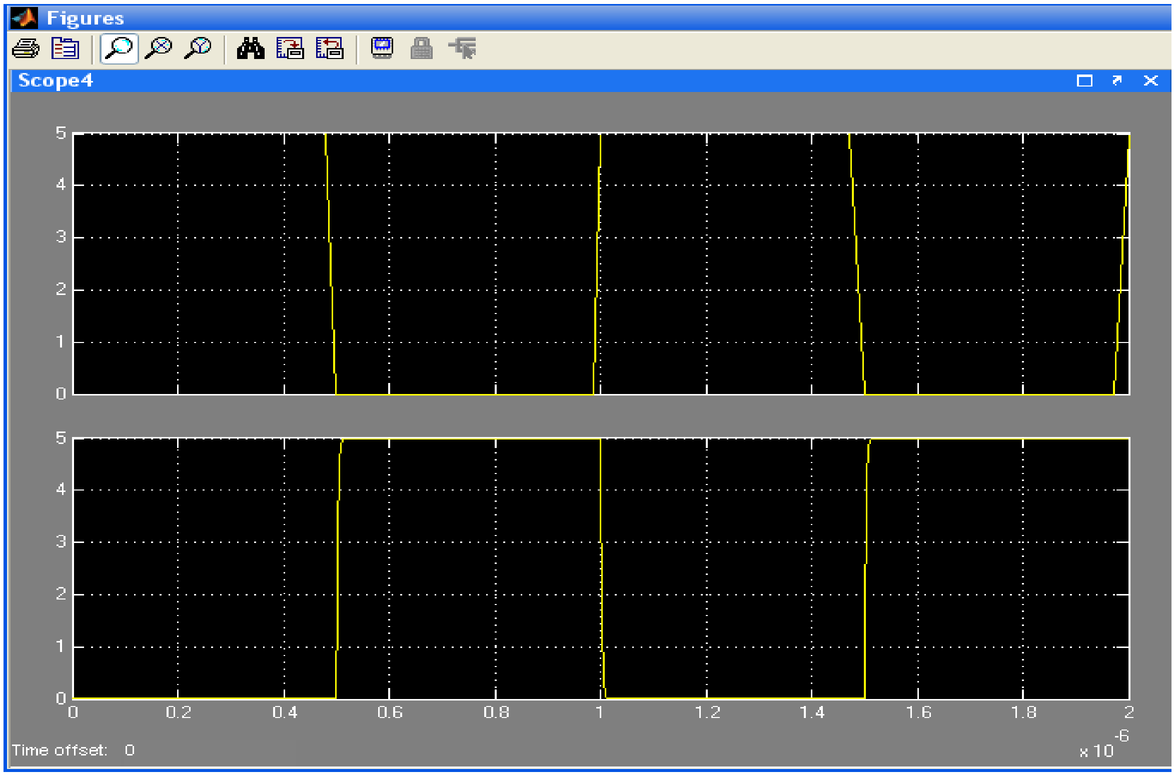


Figure 6. The input and output waveform: Simulink scope

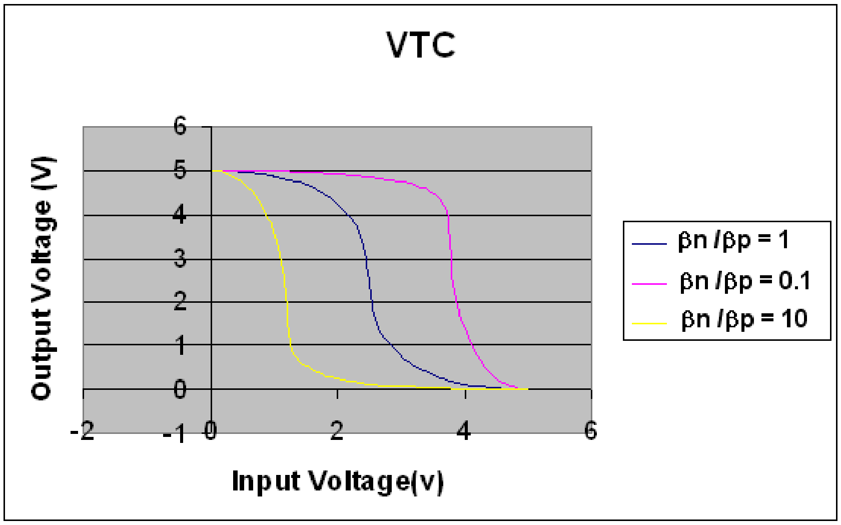


Figure 7. The transfer characteristics curve for different values of λ with load CL=50pf

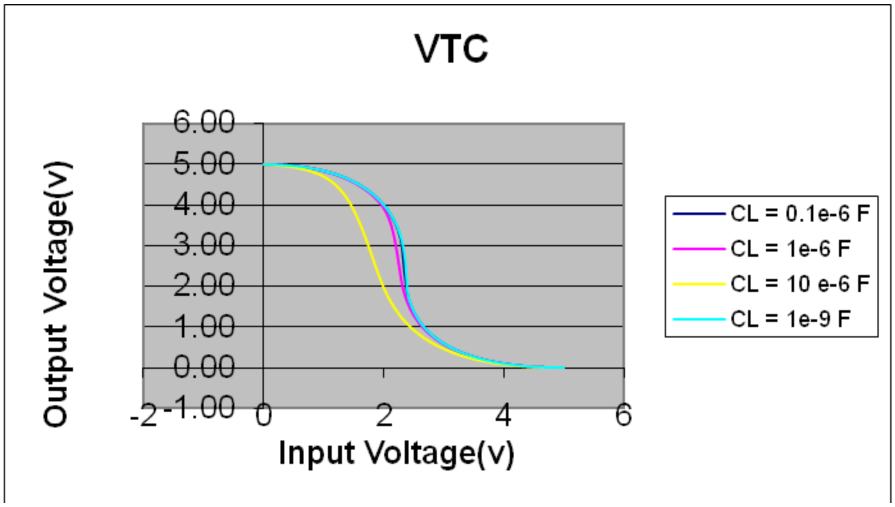


Figure8.The transfer characteristics curve for different values of load with λ =1

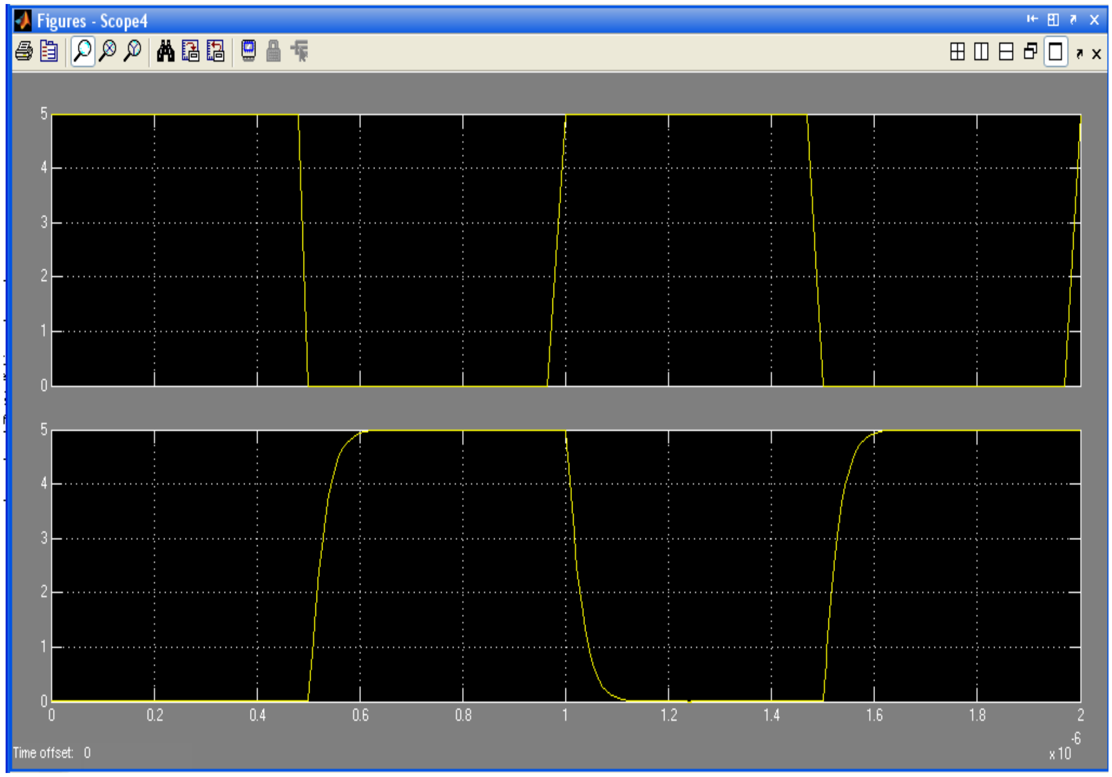


Figure 9a. The effect of change in load on propagation delay with λ =1

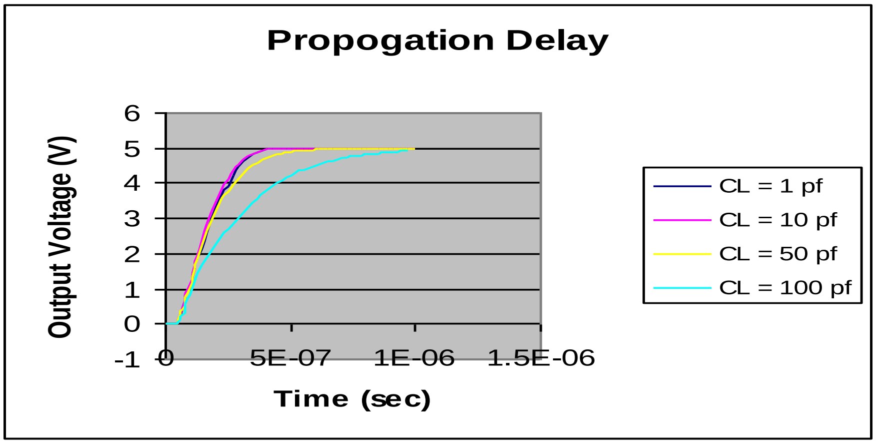


Figure 9b. The effect of change in load on propagation delay with λ =1

6. CONCULSION

A practical CAD Model based design (MBD) methodology for CMOS inverters presented in this paper. The obtained experimental results shows that Simulink can be used as a platform for MBD .The proposed algorithm can be used for implementing analog circuit design on FPGA using HDL coder. Using this algorithm, the HDL codes can be automatically generated from Simulink models for analog circuit design. Continuous verification can be achieved via processor-in the- loop, hardware-in-the-loop or co-simulation. Therefore by using the suggested methodology all development phases can be simulated and verified and implemented under one roof rather than actually doing them separately which saves time and costs effectively.

In future we propose to design VCO using CMOS inverter and implementing same on FPGA/ ASIC using HDL coder.

7. REFERENCES

[1] Boray S , Deepaksubramanyan, Priyank Parakh Zhenhua Chen, Hussam Diab, Duane Marcy and Fred H. Schlereth, Department of Electrical Engineering and Computer Science, *An FPGA-Based MOS Circuit Simulator* ,Syracuse University, Syracuse, NY 13244, USA, IEEE,2005.

[2] Dejan Markovic, Chen Chang, Brian Richards, Hayden So, Borivoje Nikolic, Robert W. Brodersen, Berkeley ,*ASIC Design and Verification in an FPGA Environment*. , Custom Integrated Circuits Conference (CICC) 21-7-4 IEEE 2007

[3] Dr. Hikmat N. Abdullah B.Sc. Hussein A.Hadi *Design And Implementation Of FPGA Based software Defined Radio Using Simulink HDL Coder* ,CUAS_Journal,2010.

[4] L.Bisdounis, 0. Koufopavlou, S. Nikolaidis L. ,*Modelling output waveform and propagation delay of CMOS inverter `in the submicron range,* IEEE 2007.

[5] Ms. Ujwala A. Belorkar 1 and Dr. S.A.Ladhake, *High performance voltage controlled oscillator using 65NM* ,VLSI Technology International Journal of Computer Networks & Communications (IJCNC), Vol.2, No.4,July2010

[6] Neil H. E. Weste, Kamram Eshcraghian, *Principles of CMOS VLSI design*.

[7] Razavi, *Monolithic Phase-locked loops and clock Recovery circuit,* IEEE press 1997.

[8] www.mathwokrs.com

[9] Thomas A. Henzinger, EPFL Joseph Sifakis, Verimag, *The Discipline of Embedded Systems Design,* IEEE computer Society 2007

[10] Zhangcai Huang, Hong Yu, Atsushi Kurokawa and Yasuaki Inoue, Modeling the Overshooting Effect for CMOS Inverter inNanometer Technologies, IEEE transactions on computer-aided design of integrated circuits and systems, vol. 29, no. 2, February 2010.