World Academy of Science, Engineering and Technology
International Journal of Aerospace and Mechanical Engineering
Vol:4, No:10, 2010

FPGA Based Longitudinal and Lateral Controller Implementation for a Small UAV

Hafiz ul Azad, Dragan V.Lazic, Waqar Shahid

Abstract—This paper presents implementation of attitude controller for a small UAV using field programmable gate array (FPGA). Due to the small size constrain a miniature more compact and computationally extensive; autopilot platform is needed for such systems. More over UAV autopilot has to deal with extremely adverse situations in the shortest possible time, while accomplishing its mission. FPGAs in the recent past have rendered themselves as fast, parallel, real time, processing devices in a compact size. This work utilizes this fact and implements different attitude controllers for a small UAV in FPGA, using its parallel processing capabilities. Attitude controller is designed in MATLAB/Simulink environment. The discrete version of this controller is implemented using pipelining followed by retiming, to reduce the critical path and thereby clock period of the controller datapath. Pipelined, retimed, parallel PID controller implementation is done using rapid-prototyping and testing efficient development tool of "system generator", which has been developed by Xilinx for FPGA implementation. The improved timing performance enables the controller to react abruptly to any changes made to the attitudes of UAV.

Keywords—Field Programmable gate array (FPGA), Hardware descriptive Language (HDL), PID, Pipelining, Retiming, Xilinx System Generator.

I. Introduction

UNMANNED Air vehicles (UAV) have been using for various applications in the recent past. These include military as well as civil applications. Due to the absence of pilot onboard, UAV flight control system (FCS) has to deal with all situations of taxing,take-off, landing, obstacle avoidance, path following etc. UAV flight control system (FCS) needs a processing unit which is capable of carrying out, all these tasks quickly and efficiently and at the same time occupying less volume and weight as these are the main constrains for such small size systems. Over the past decades both the advantage of parallel processing and the increased number of gates have lead to a rapid increase in the popularity of Field programmable Gate Array (FPGA) implementation. FPGA also offers the ability of reconfigurability both, compilation time and run-time as compare to ASIC which is configured only once. Furthermore FPGAs such as virtex-II pro, have embedded microprocessor units (MPU) , making it possible to build a whole system on a single chip. It is due to these reasons that currently FPGA are replacing DSP and microcontroller based embedded system as these implement different algorithms in software which is executed sequentially. For a rapid prototyping of FPGA based autopilot design a simulation software is needed that can represent exactly the hardware being used. Xilinx system generator tool box for simulink has made it possible to simulate the hardware within the graphical environment of simulink, and also generate Hardware Description Language (HDL) code needed for the implementation in FPGA.

FPGA based designs have been widely applied in digital system applications by many researchers. Shashikala Narasimha Murthy et al. presented a methodology for implementation of RC-truck control on FPGA using system generator [1]. W.alvis et al. Proposed FPGA based flexible autopilot platform for unmanned systems [2]. Amol A.kalage et al. presented modelling and simulation of FPGA based direct torque control of induction motor drive using Xilinx system generator tool box [3]. Angkul Kongmunvattana and Prabhas Chongstitvatana reported a FPGA based behavioural control system for a mobile robot [4]. Wei Zhao et al. implemented a digital PID controller based system in FPGA [5].Parallel and serial implementation schemes both for single channel and multichannel control, have been presented, taking in to consideration the tradeoffs of speed, area and power consumption. Though parallel PID controller is faster than the serial one, but at the cost of increased critical path ,thereby reducing the throughput of the controller.

This work implements a pipelined parallel PID attitude controller for a small UAV in FPGA. Matlab/Simulink having aerosim toolbox for aircraft modeling implements UAV dynamic model [9]. After identifying the potential problems in controlling this dynamic model, controller is designed in longitudinal and lateral planes separately. These controllers are then discretized so as to implement them in FPGA. Critical paths of these controllers are broken down, using cutest pipelining and retiming techniques to improve throughput of the controller [6]. Finally these controllers are augmented with the dynamic model of UAV using Xilinx system generator tool box.

H.Asad has done his Masters in Aerospace engineering from University of Belgrade Serbia (e-mail:lucid.vivi@ gmail.com).
Dragan V.Lazic is with Department of Automatic Control University of Belgrade Serbia (e-mail: dragan.lazic@gmail.com).
Waqar Shahid is with Department of Electrical Engineering, Air University Islamabad Pakistan (e-mail: waqar.shahid@mail.au.edu.pk).

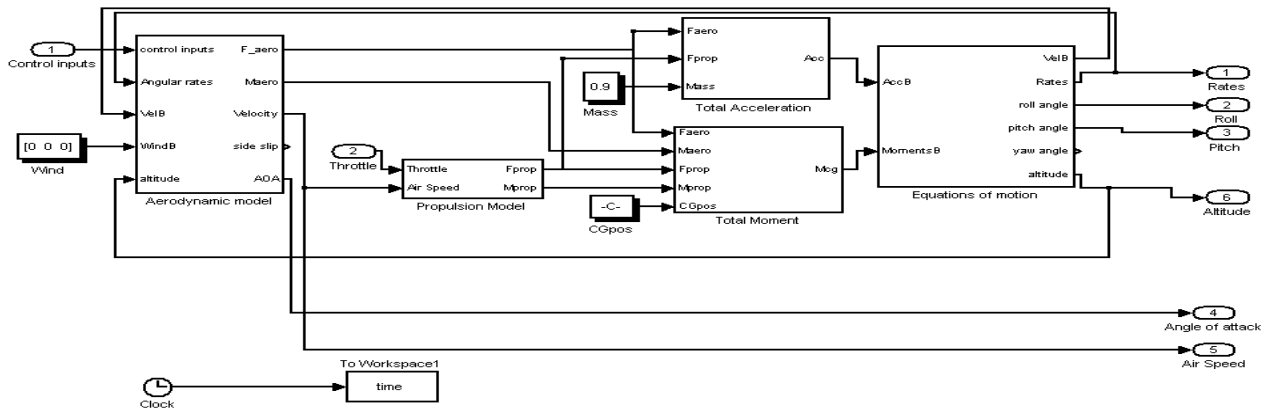

Fig. 1 Simulink UAV Model

II. UAV Model

UAV airframe used in this work is smartone, developed by FOI-Swedish Defence Research Agency [7].UAV is modeled as a standard 6-DOF non-linear rigid body aircraft model. Standard body axes centered at the aircraft center of gravity is considered as the modelling reference axes, where x points forward through the aircraft nose, y is directed to the starboard (right);and z is directed downward. The standard body axis equations of motion that are implemented in simulink, are given below using the notation of Stevens and Lewis [8]. These (1-4) are called force, moment, kinematic and navigation equations respectively. In these equations (U,V,W) are the body axis linear velocities; (P,Q,R) are the body axis angular rates; (Φ,θ,ψ) are the Euler angle representation of the vehicle attitudes; and (p_N , p_E , h) are the North, East and height positions.

$$\begin{bmatrix} \dot{U} \\ \dot{V} \\ \dot{W} \end{bmatrix} = \begin{bmatrix} +RV & -QW & +g_x & +\frac{F_x}{m} \\ -RU & +PW & +g_y & +\frac{F_y}{m} \\ +QU & -PV & +g_z & +\frac{F_z}{m} \end{bmatrix} \quad (1)$$

$$\begin{bmatrix} \dot{P} \\ \dot{Q} \\ \dot{R} \end{bmatrix} = J^{-1} \begin{bmatrix} L \\ M \\ N \end{bmatrix} - \begin{bmatrix} 0 & -R & Q \\ R & 0 & -P \\ -Q & P & 0 \end{bmatrix} J \begin{bmatrix} P \\ Q \\ R \end{bmatrix} \quad (2)$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \begin{bmatrix} P \\ Q \\ R \end{bmatrix} \quad (3)$$

$$\begin{bmatrix} \dot{p}_N \\ \dot{p}_E \\ -\dot{h} \end{bmatrix} = B^{-1} \begin{bmatrix} U \\ V \\ W \end{bmatrix} \quad (4)$$

A. UAV Model Implementation

Aerosim blockset [10] is used to implement UAV model in Matlab/Simulink. Aerosim blockset provides a complete set of tools for developing non-linear 6-degree-of-freedom(DOF) aircraft dynamic model. These Simulink blocks include the non-linear equations of motion, linear aerodynamics based on component build up, earth models, propulsion models and other important tools for the development of aircraft model. Simulink non-linear model of UAV is shown in figure.1.

III. MATLAB/SIMULINK CLOSE LOOP CONTROL

Figure.2 shows the methodology for the design and implementation of a close loop controller [1],[10],[11].

A. Continuous time Controller Design

After identifying undesired behaviour of non-linear model by simulation in simulink, the model is trimmed, and then linearised about different equilibrium points. This Linear model is then divided in to two independent models, i.e. longitudinal and lateral models. Longitudinal model consist of pitch(θ),angular rate(Q) ,forward velocity (U) and altitude (h) state variables. Lateral model consist of roll(φ), angular rates (P,R) and yaw(ψ) state variables. To improve damping of the system, stability augmentation is carried out for both the models by introducing gains in the pitch and roll rates feedback loops [10],[11]. PID is mainly used as a control algorithm, and its gains are tuned for these models. The performances of these controllers are finally verified, with non-linear model for various flight conditions. Figure.3 shows close loop control system along with non-linear model of UAV.

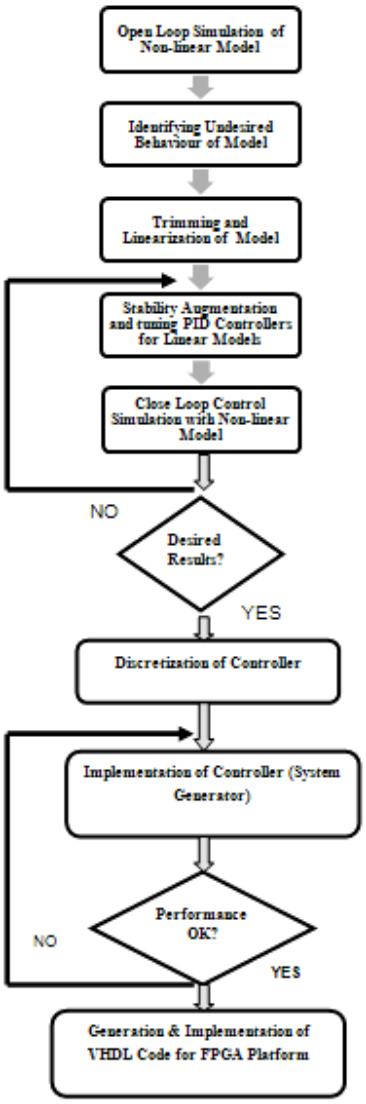

Fig. 2 Design Flow

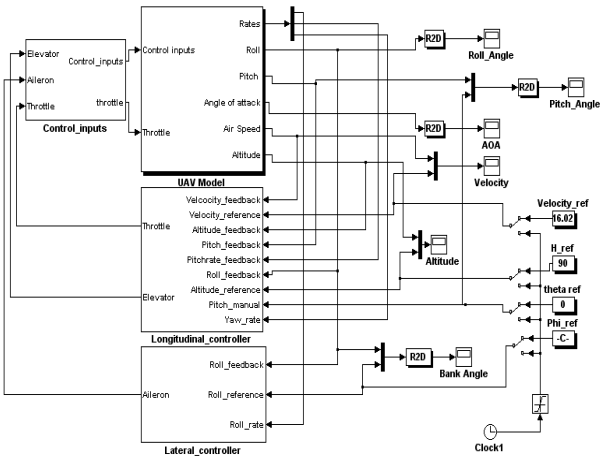

Fig. 3 Controller with Non-linear UAV Model

B. *Discritization of Controller*

PID is the most commonly used control law and is described in a differential equation as:

$$U(t)= K_P e(t) + K_I \int e(t)dt + K_D \frac{de(t)}{dt} \quad (5)$$

Where U(t) is the output of controller and e(t) is the input to the controller. K_p ,K_I and K_D are the three gains of the controller. Three types of approximations can be used to discretize this control law [12]. These are ,forward approximation ,backward approximation and trapezoidal approximation. Backward approximation is used here for discretization of PID control law, and the following difference equation is obtained.

$$U(t) = U(t-1) + K_p[e(t) - e(t-1)] + K_I.T.e(t) + \frac{K_D}{T}[e(t) - 2e(t-1) + e(t-2)] \quad (6)$$

This equation can also be written as [4]:

$$U(t) = U(t-1) + K_0 e(t) + K_1 e(t-1) + K_2 e(t-2) \quad (7)$$

Where

$$K_0 = K_p + (K_I T) + \frac{K_D}{T} \quad (8)$$

$$K_1 = -K_p - 2\frac{K_D}{T} \quad (9)$$

$$K_2 = \frac{K_D}{T} \quad (10)$$

IV. Parallel Design of Controller

Equation (7) can be implemented using either parallel or serial design techniques [5]. For a parallel design, each basic operation has its own arithmetic unit-either an adder or multiplier. In serial design technique only one arithmetic unit is used for each kind of arithmetic operation. This work emphasizes on speed and uses the parallel computational capability of FPGA .Therefore a parallel design technique is adopted to implement the controller. The data path of a parallel controller design is obtained by decomposing equation (7) in to its basic arithmetic operations [5].

A. *Pipelined Parallel Controller Architecture*

The shortest cycle time of the clock of a synchronous sequential data path is a measure of its performance, and it is bounded by the critical path [13]. Critical path in a data path is that path which produces the longest propagation delay. Parallel controller design introduces a long critical path as shown in figure 4 thus resulting in a long clock period.By inserting pipeline registers in the critical path, clock period can be decreased. To shorten this long critical path of the parallel controller design, famous techniques of cutset pipelining and cutset retiming are used [6],[13]. Figure.4 shows a parallel controller design of equation (7) highlighting different cutests.

Feed forward cutest is the minimum set of edges, if removed from the data flow graph (DFG), partitions it into two connected subgraphs such that there is no path between an

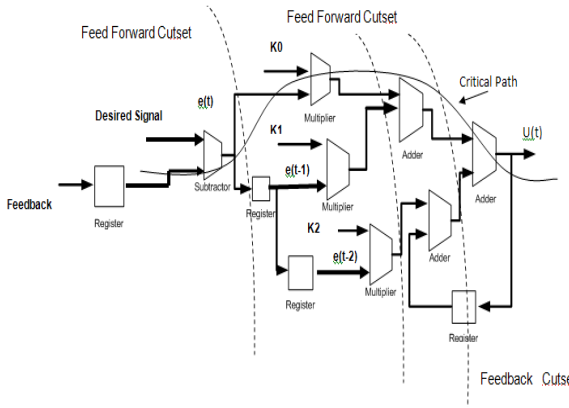

Fig. 4 Cutsets of Controller data path

input node and output node [13] . Feedback cutest cannot be used for introducing pipeline registers but we can transfer the same amount of delays from edges of the same direction across a feedback cutest of a data flow graph (DFG) to all edges of opposing edges. This will not change the output but its timing. This is called "Retiming". Figure.4 shows the critical path consist of two adders, one subtractor and one multiplier. Dashed lines representing the feed forward cutsets specify locuses for pipeline registers. Introducing pipeline registers in these feed forward cutsets shortens the critical path to just two adders. Applying then retiming to the feedback cutest, shifting the delay element in the feedback path to the other two edges of the feedback cutest. This reduces the critical path to a single functional unit. The pipelined, retimed, data path for the parallel controller design is shown in figure.5.

B. Controller Implementation in Xilinx System Generator

Implementation of different algorithms in FPGAs requires a comprehensive knowledge of both digital system design and Hardware descriptive Language (HDL). As algorithm becomes more complex, writing HDL code becomes more cumbersome. Xilinx system generator block sets developed by Xilinx provides high level abstraction tool for FPGA users [14]. It consists of a library of FPGA blocks, required for building hardware models. It also consists a VHDL code

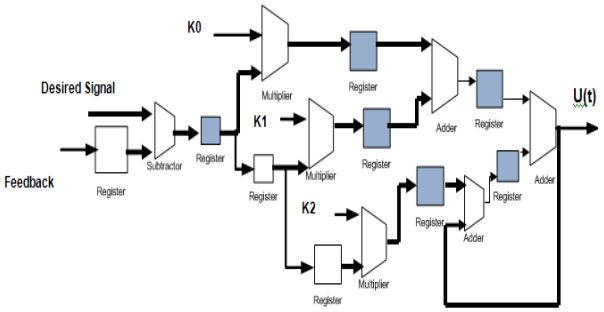

Fig. 5 Pipelined ,Retimed Parallel Controller

generator which automatically generates VHDL code from the created models [14], thereby saving the designer from the complex details of HDL code and reducing the Implementation time.

System generator/simulink tool box is used to implement the pipelined and retimed parallel controller design of figure 5. This is shown in figure 6. Signals from simulink non-system generator blocks are in continuous floating point form, therefore they must be digitized before giving to the system generator blocks. GatewayIn and GatewayOut blocks are the interface points between system generator blocks and non-system generator simulink environment. Controller is implemented using system generator blocks of adders, subtractors, multipliers and registers.

V. Simulation Results of Close Loop Controller with Non-Linear Model

The complete close loop controller consists of various controllers. This includes Pitch Attitude Hold(PAH) controller Altitude Hold (ALH) controller, Velocity Acquire and Hold (VAH) controller ,and Roll Attitude Hold(RAH) controller [10],[11]. Some of these controllers are PID, and some of them are PI controllers. The design of PI controller is similar to PID, as a pipelined, retimed parallel controller and implemented using implementation of figure 6.Results are shown in figures (7-9) below. These results are obtained by giving step inputs to different channels of the close loop control system.

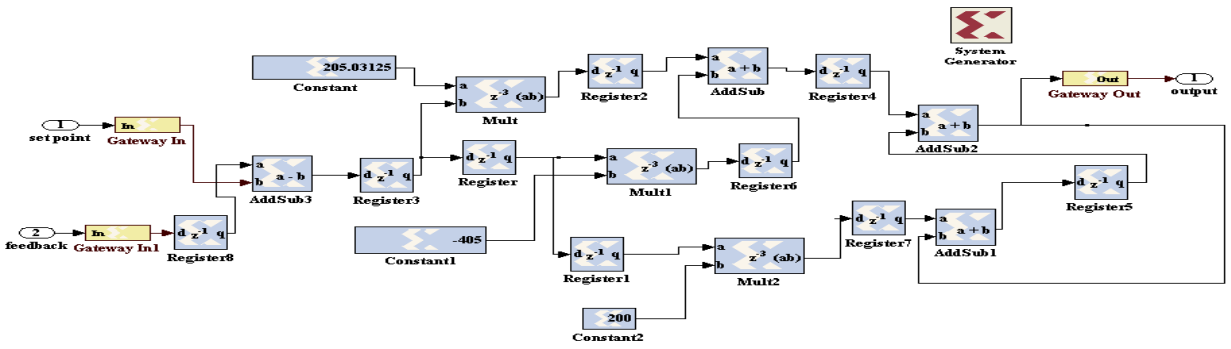

Fig. 6 System Generator Implementation of

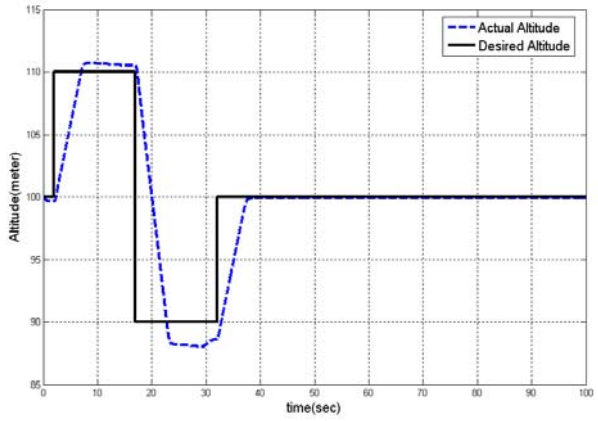

Fig. 7 ALH Controller Response

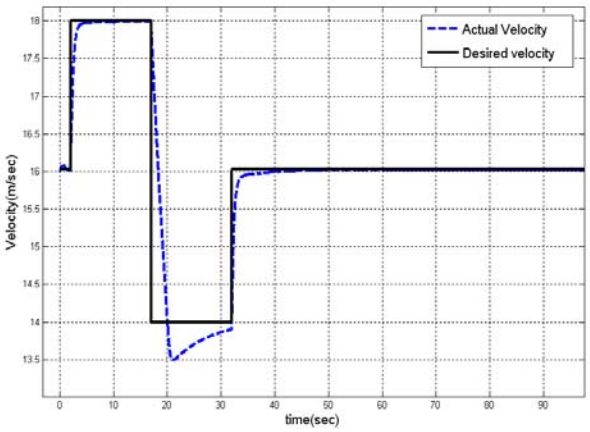

Fig. 8 VAH Controller Response

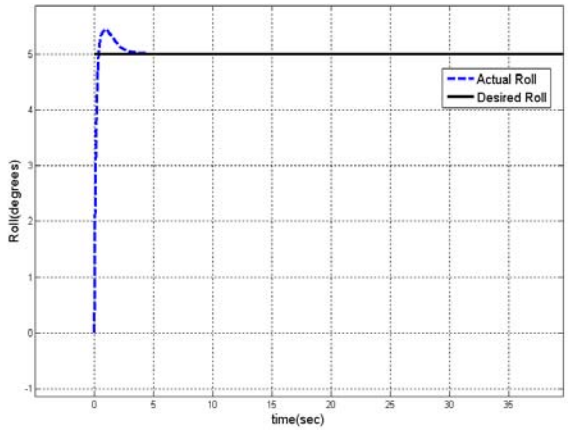

Fig. 9 RAH Controller Response

VI. Conclusion

FPGA based attitude controller implementation for a small UAV, has been proposed in this paper. FPGA being a fast and parallel device greatly reduces the computational time of different algorithms. A new approach to the implementation of PID controller has been adopted in this paper. Previously designed parallel architecture though improves the speed as compare to serial design, but results in a long critical path. This puts a lower limit on the clock period which is a bottleneck in case we need a faster clock. Pipelined and retimed parallel PID controller implementation reduces clock period to the time of a single functional unit, contrary to the previous parallel implementation which has a long critical path and a smaller throughput. The improved timing performance allows the controller to react abruptly to the minimal changes made in attitudes of UAV, which is extremely desirable in adverse situations that this type of vehicle usually undergoes through. Xilinx system generator has been used to implement the controller in simulink. This greatly reduces the design and testing time of the controller.

References

[1] Shashikala Narasimha Murthy, Wendy Alvis, Rakesh Shirodkar, Kimon Valavanis and Wilfrido Moreno, "Methodology for Implementation of Unmanned Vehicle Control on FPGA Using System Generator" in Proceedings of the 7th International Conference on Devices,Circuits and Systems,Mexico,Apr 28-30,2008

[2] W.Alvis, S.Murthy, K. Valavanis, W. Moreno, S. Katkoori , "FPGA Based Flexible Autopilot Platform for Unmanned Systems" in Proceedings of the 15th Mediterranean Conference on Control & Automation , july 27-29 , Athens -Greece.

[3] Amol A.Kalage, Vasant M.Jape, Sarika V. Tade, Manik M.Hapse, Ravindra G. Dabhade, "Modeling & Simulation of FPGA based direct Torque Control of Induction Motor Drive" ,International Journal of recent trends in Engineering, Vol 1, No.4,May 2009.

[4] Angkul Kongmunvattana and Prabhas Chongstitvatana, "A FPGA based behavioral Control System for a Mobile Robot" ,IEEE Asia Pacific Conference on Circuits and Systems, Chiangmai, Thailand, 1998

[5] Wei Zhao, Byung Hwa Kim, Amy C.Larson , Richard M. Voyles, "FPGA Implementation of Close loop Control System for Small Scale Robot" Department of Electrical Engineering University of Minnesota.

[6] K.K parhi "VLSI Digital signal processing Systems, Design & Implementation".

[7] Herik Grankvist "Autopilot Design and Path Planning for UAV" FOI Swedish Defense & Security, System &technology Stockholm. W.-K. Chen, *Linear Networks and Systems* (Book style). Belmont, CA: Wadsworth, 1993, pp. 123–135.

[8] B.L. Stevens and F.L. Lewis. Aircraft Control and Simulation. John Wiley and Sons New York, 1992.

[9] www.u-dynamics.com

[10] Jhon H.Blake Lock, Automatic control of Aircraft and Missiles. John Wiley & Sons publishing company, New York.

[11] Rober C.Nelson. Flight Stability and Automatic Control. MacGraw Hill Book Company New York.

[12] Chi-Tsong Chen, "Analog and Digital Control System Design Transfer Function, State-Space, and algebraic Methods." Saunders College Publishing

[13] *Michael D.Ciletti, "Advanced Digital Design with the Verilog HDL" Prentice Hall of India New Delhi 2005*

[14] Xilinx, "User Guide for System Generator",version 8.1.