Design of FPGA-based Sliding Mode Controller for Robot Manipulator

Farzin Piltan *SSP.ROBOTIC@yahoo.com*
Department of Electrical and Electronic Engineering,
Faculty of Engineering, Universiti Putra Malaysia 43400
Serdang, Selangor, Malaysia

N. Sulaiman *nasri@eng.upm.edu.my*
Department of Electrical and Electronic Engineering,
Faculty of Engineering, Universiti Putra Malaysia 43400
Serdang, Selangor, Malaysia

M. H. Marhaban *hamiruce@eng.upm.edu.my*
Department of Electrical and Electronic Engineering,
Faculty of Engineering, Universiti Putra Malaysia 43400
Serdang, Selangor, Malaysia

Adel Nowzary *adnowzary@yahoo.com*
Industrial Electrical and Electronic Engineering
SanatkadeheSabze Pasargad. CO (S.S.P. Co),
NO:16 , PO.Code 71347-66773, Fourth floor
Dena Apr , Seven Tir Ave , Shiraz , Iran

Mostafa Tohidian *mostafa.tohidian@gmail.com*
Industrial Electrical and Electronic Engineering
SanatkadeheSabze Pasargad. CO (S.S.P. Co),
NO:16 , PO.Code 71347-66773, Fourth floor
Dena Apr , Seven Tir Ave , Shiraz , Iran

Abstract

One of the most active research areas in the field of robotics is robot manipulators control, because these systems are multi-input multi-output (MIMO), nonlinear, and uncertainty. At present, robot manipulators is used in unknown and unstructured situation and caused to provide complicated systems, consequently strong mathematical tools are used in new control methodologies to design nonlinear robust controller with satisfactory performance (e.g., minimum error, good trajectory, disturbance rejection). Robotic systems controlling is vital due to the wide range of application. Obviously stability and robustness are the most minimum requirements in control systems; even though the proof of stability and robustness is more important especially in the case of nonlinear systems. One of the best nonlinear robust controllers which can be used in uncertainty nonlinear systems is sliding mode controller (SMC). Chattering phenomenon is the most important challenge in this controller. Most of nonlinear controllers need real time mobility operation; one of the most important devices which can be used to solve this challenge is Field Programmable Gate Array (FPGA). FPGA can be used to design a controller in a single chip Integrated Circuit (IC). In this research the SMC is designed using VHDL language for implementation on FPGA device (XA3S1600E-Spartan-3E), with minimum chattering and high processing speed (63.29 MHz).

Keywords: Robot Manipulator, Sliding Mode Controller, Chattering Phenomenon, FPGA, VHDL language.

1. INTRODUCTION

A robot is a machine which can be programmed as a reality of tasks which it has divided into three main categories i.e. robot manipulators, mobile robots and hybrid robots. PUMA 560 robot manipulator is an articulated 6 DOF serial robot manipulator. This robot is widely used in industrial and academic area and also dynamic parameters have been identified and documented in the literature. From the control point of view, robot manipulator divides into two main sections i.e. kinematics and dynamic parts. Estimate dynamic parameters are considerably important to control, mechanical design and simulation[1].

Sliding mode controller (SMC) is one of the influential nonlinear controllers in certain and uncertain systems which are used to present a methodical solution for two main important controllers' challenges, which named: stability and robustness. Conversely, this controller is used in different applications; sliding mode controller has subsequent drawbacks i.e. chattering phenomenon, and nonlinear equivalent dynamic formulation in uncertain systems[1-2].

In order to solve the chattering in the systems output, boundary layer method should be applied so beginning able to recommended model in the main motivation. Conversely boundary layer method is constructive to reduce or eliminate the chattering; the error response quality may not always be so high. Besides using boundary layer method in the main controller of a control loop, it can be used to adjust the sliding surface slope to have the best performance (reduce the chattering and error performance)[3].

Commonly, most of nonlinear controllers in robotic applications need a mobility real time operation. FPGA-based controller has been used in this application because it is small device in size, high speed, low cost, and short time to market. Therefore FPGA-based controller can have a short execution time because it has parallel architecture [4-7].

This paper is organized as follows:
In section 2, main subject of modelling PUMA-560 robot manipulator formulation are presented. Detail of classical sliding mode controller is presented in section 3. In section 4, the main subject of FPGA-based sliding mode controller is presented. In section 5, the simulation result is presented and finally in section 6, the conclusion is presented.

2. DYNAMIC FORMULATION OF ROBOT

It is well known that the equation of an *n-DOF* robot manipulator governed by the following equation [1-2]:

$$\boldsymbol{M(q)\ddot{q} + N(q,\dot{q}) = \tau} \tag{1}$$

Where $\boldsymbol{\tau}$ is actuation torque, $\boldsymbol{M(q)}$ is a symmetric and positive define inertia matrix, $N(q,\dot{q})$ is the vector of nonlinearity term. This robot manipulator dynamic equation can also be written in a following form:

$$\boldsymbol{\tau = M(q)\ddot{q} + B(q)[\dot{q}\,\dot{q}] + C(q)[\dot{q}]^2 + G(q)} \tag{2}$$

Where the matrix of coriolios torque is $\boldsymbol{B(q)}$, $\boldsymbol{C(q)}$ is the matrix of centrifugal torques, and $\boldsymbol{G(q)}$ is the vector of gravity force. The dynamic terms in equation (2) are only manipulator position. This is a decoupled system with simple second order linear differential dynamics. In other words, the component $\boldsymbol{\ddot{q}}$ influences, with a double integrator relationship, only the joint variableq_i, independently of the motion of the other joints. Therefore, the angular acceleration is found as to be[2]:

$$\boldsymbol{\ddot{q} = M^{-1}(q).\{\tau - N(q,\dot{q})\}} \tag{3}$$

)

This technique is very attractive from a control point of view. This paper is focused on the design FPGA-based controller for PUMA-560 robot manipulator.

2.1 PUMA 560 Dynamic Formulation

Position control of PUMA-560 robot manipulator is analyzed in this paper; as a result the last three joints are blocked. The dynamic equation of PUMA-560 robot manipulator is given as

$$\boldsymbol{M(\ddot{\theta})}\begin{bmatrix}\ddot{\theta}1\\ \ddot{\theta}_2\\ \ddot{\theta}_3\end{bmatrix}+\mathbf{B}(\boldsymbol{\theta})\begin{bmatrix}\dot{\theta}_1\dot{\theta}_2\\ \dot{\theta}_1\dot{\theta}_3\\ \dot{\theta}_2\dot{\theta}_3\end{bmatrix}+\boldsymbol{C}(\boldsymbol{\theta})\begin{bmatrix}\dot{\theta}_1^2\\ \dot{\theta}_2^2\\ \dot{\theta}_3^2\end{bmatrix}+\boldsymbol{G}(\boldsymbol{\theta})=\begin{bmatrix}\tau_1\\ \tau_2\\ \tau_3\end{bmatrix} \tag{4}$$

Where

$$\boldsymbol{M}(\boldsymbol{q})=\begin{bmatrix}M_{11} & M_{12} & M_{13} & 0 & 0 & 0\\ M_{21} & M_{22} & M_{23} & 0 & 0 & 0\\ M_{31} & M_{32} & M_{33} & 0 & M_{35} & 0\\ 0 & 0 & 0 & M_{44} & 0 & 0\\ 0 & 0 & 0 & 0 & M_{55} & 0\\ 0 & 0 & 0 & 0 & 0 & M_{66}\end{bmatrix} \tag{5}$$

$$\boldsymbol{B}(\boldsymbol{q})=$$
$$\begin{bmatrix}b_{112} & b_{113} & 0 & b_{115} & 0 & b_{123} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & b_{214} & 0 & 0 & b_{223} & 0 & b_{225} & 0 & 0 & b_{235} & 0 & 0 & 0 & 0\\ 0 & 0 & b_{314} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ b_{412} & b_{412} & 0 & b_{415} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & b_{514} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\end{bmatrix} \tag{6}$$

$$\boldsymbol{C}(\boldsymbol{q})=\begin{bmatrix}0 & C_{12} & C_{13} & 0 & 0 & 0\\ C_{21} & 0 & C_{23} & 0 & 0 & 0\\ C_{31} & C_{32} & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0\\ C_{51} & C_{52} & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0\end{bmatrix} \tag{7}$$

$$\boldsymbol{G}(\boldsymbol{q})=\begin{bmatrix}0\\ g_2\\ g_3\\ 0\\ g_5\\ 0\end{bmatrix} \tag{8}$$

Suppose $\ddot{q}$ is written as follows

$$\ddot{\boldsymbol{q}}=\boldsymbol{M}^{-1}(\boldsymbol{q}).\{\boldsymbol{\tau}-[\boldsymbol{B}(\boldsymbol{q})\dot{\boldsymbol{q}}\dot{\boldsymbol{q}}+\boldsymbol{C}(\boldsymbol{q})\dot{\boldsymbol{q}}^2+\boldsymbol{g}(\boldsymbol{q})]\} \tag{9}$$

and I is introduced as

$$\boldsymbol{I}=\{\boldsymbol{\tau}-[\boldsymbol{B}(\boldsymbol{q})\dot{\boldsymbol{q}}\dot{\boldsymbol{q}}+\boldsymbol{C}(\boldsymbol{q})\dot{\boldsymbol{q}}^2+\boldsymbol{g}(\boldsymbol{q})]\} \quad (1$$

0)

$\ddot{q}$ can be written as

$$\ddot{q} = M^{-1}(q).I \tag{11}$$

Therefore I for PUMA-560 robot manipulator can be calculated by the following equation

$$I_1 = \tau_1 - [\, b_{112}\dot{q}_1\dot{q}_2 + b_{113}\dot{q}_1\dot{q}_3 + 0 + b_{123}\dot{q}_2\dot{q}_3] - [\, C_{12}\dot{q}_2{}^2 + C_{13}\dot{q}_3{}^2] - g_1 \tag{12}$$

$$I_2 = \tau_2 - [\, b_{223}\dot{q}_2\dot{q}_3] - [\, C_{21}\dot{q}_1{}^2 + C_{23}\dot{q}_3{}^2] - g_2 \tag{13}$$

$$I_3 = \tau_3 - \left[C_{31}\dot{q}_1{}^2 + C_{32}\dot{q}_2{}^2\right] - g_3 \tag{14}$$

$$I_4 = \tau_4 - [\, b_{412}\dot{q}_1\dot{q}_2 + b_{413}\dot{q}_1\dot{q}_3] - g_4 \tag{15}$$

$$I_5 = \tau_5 - [\, C_{51}\dot{q}_1{}^2 + C_{52}\dot{q}_2{}^2] - g_5 \tag{16}$$

$$I_6 = \tau_6 \tag{17}$$

3. CLASSICAL SLIDING MODE CONTROL

Sliding mode controller (SMC) is a powerful nonlinear controller which has been analyzed by many researchers especially in recent years. This theory was first proposed in the early 1950 by Emelyanov and several co-workers and has been extensively developed since then with the invention of high speed control devices[1-2].

A time-varying sliding surface $s(x,t)$ is given by the following equation:

$$s(x,t) = (\frac{d}{dt} + \lambda)^{n-1}\, \widetilde{x} = 0 \tag{18}$$

where λ is the constant and it is positive. To further penalize tracking error integral part can be used in sliding surface part as follows:

$$s(x,t) = (\frac{d}{dt} + \lambda)^{n-1} \left(\int_0^t \tilde{x}\, dt \right) = 0 \tag{19}$$

The main target in this methodology is keep $s(x,t)$ near to the zero when tracking is outside of $s(x,t)$. Therefore, one of the common strategies is to find input U outside of $s(x,t)$.

$$\frac{1}{2}\frac{d}{dt} s^2(x,t) \le -\zeta |s(x,t)| \tag{20}$$

where ζ is positive constant.

$$\text{If } \mathbf{S(0)>0} \rightarrow \frac{d}{dt} S(t) \le -\zeta \tag{21}$$

To eliminate the derivative term, we used an integral term from t=0 to t=t_{reach}

$$\int_{t=0}^{t=t_{reach}} \frac{d}{dt} S(t) \le -\int_{t=0}^{t=t_{reach}} \eta \rightarrow S\,(t_{reach}) - S(0) \le -\zeta(t_{reach} - 0) \tag{22}$$

Where t_{reach} is the time that trajectories reach to the sliding surface so, if we assume that S(t_{reach} = 0) then:

$$0 - S(0) \le -\eta(t_{reach}) \rightarrow t_{reach} \le \frac{S(0)}{\zeta} \tag{23}$$

and

$$if\ S(0) < 0 \rightarrow 0 - S(0) \le -\eta(t_{reach}) \rightarrow S(0) \le -\zeta(t_{reach}) \rightarrow t_{reach} \le \frac{|S(0)|}{\eta} \tag{24}$$

Equation (24) guarantees time to reach the sliding surface is smaller than $\frac{|S(0)|}{\zeta}$ if trajectories are outside of **S(t).**

$$if\ S_{t_{reach}} = S(0) \rightarrow error(x - x_d) = 0 \tag{25}$$

suppose **S** defined as

$$s(x,t) = (\frac{d}{dt} + \lambda) \quad \tilde{x} = (\dot{x} - \dot{x}_d) + \lambda(x - x_d) \tag{26}$$

The derivation of S, namely, $\dot{S}$ can be calculated as the following formulation:

$$\dot{S} = (\ddot{x} - \ddot{x}_d) + \lambda(\dot{x} - \dot{x}_d) \tag{27}$$

suppose define the second order system as,

$$\ddot{x} = f + u \; \rightarrow \dot{S} = f + U - \ddot{x}_d + \lambda(\dot{x} - \dot{x}_d) \tag{28}$$

Where f is the dynamic uncertain, and also if $S = 0 \; and \; \dot{S} = 0$, to have the best approximation ,$\hat{U}$ defined by,

$$\hat{U} = -\hat{f} + \ddot{x}_d - \lambda(\dot{x} - \dot{x}_d) \tag{29}$$

A simple solution to get the sliding condition when the dynamic parameters have uncertainty is the switching control law:

$$U_{dis} = \hat{U} - K(\vec{x}, t).sgn(s) \tag{30}$$

Where the function of $sgn(S)$ defined as;

$$sgn(s) = \begin{cases} 1 & s > 0 \\ -1 & s < 0 \\ 0 & s = 0 \end{cases} \tag{31}$$

and the $K(\vec{x}, t)$ is the positive constant. Suppose to rewrite the equation (20) by the following equation,

$$\begin{aligned} \frac{1}{2}\frac{d}{dt}s^2(x,t) &= \dot{S}.S = \left[f - \hat{f} - Ksgn(s)\right].S \\ &= \left(f - \hat{f}\right).S - K|S| \end{aligned} \tag{32}$$

Another method is using equation (23) instead of (24) to get sliding surface

$$\begin{aligned} s(x,t) &= (\frac{d}{dt} + \lambda)^2 \left(\int_0^t \tilde{x} \, dt\right) \\ &= (\dot{x} - \dot{x}_d) + 2\lambda(\dot{x} - \dot{x}_d) - \lambda^2(x - x_d) \end{aligned} \tag{33}$$

in this method the approximation of U can be calculated as

$$\hat{U} = -\hat{f} + \ddot{x}_d - 2\lambda(\dot{x} - \dot{x}_d) + \lambda^2(x - x_d) \tag{34}$$

To reduce or eliminate the chattering it is used the boundary layer method; in boundary layer method the basic idea is replace the discontinuous method by saturation (linear) method with small neighborhood of the switching surface. This replace is caused to increase the error performance.

$$B(t) = \{x, |S(t)| \le \emptyset\}; \emptyset > 0 \tag{35}$$

Where ∅ is the boundary layer thickness. Therefore, to have a smote control law, the saturation function $Sat(S/_{\emptyset})$ added to the control law:

$$U = K(\vec{x},t).Sat\left(S/_{\emptyset}\right) \tag{36}$$

Where $Sat\left(S/_{\emptyset}\right)$ can be defined as

$$sat\left(S/_{\emptyset}\right) = \begin{cases} 1 & (S/_{\emptyset} > 1) \\ -1 & \left(S/_{\emptyset} < 1\right) \\ S/_{\emptyset} & (-1 < S/_{\emptyset} < 1) \end{cases} \tag{37}$$

Based on above discussion, the control law for a multi degrees of freedom robot manipulator is written as:

$$\hat{\tau} = \hat{\tau}_{eq} + \hat{\tau}_{sat} \tag{38}$$

Where, the model-based component $\hat{\tau}_{eq}$ is compensated the nominal dynamics of systems. Therefore $\hat{\tau}_{eq}$ can calculate as follows:

$$\hat{\tau}_{eq} = \left[M^{-1}(B + C + G) + \dot{S}\right]M \tag{39}$$

Where

$$\hat{\tau}_{eq} = \begin{bmatrix} \widehat{\tau_{eq1}} \\ \widehat{\tau_{eq2}} \\ \widehat{\tau_{eq3}} \\ \widehat{\tau_{eq4}} \\ \widehat{\tau_{eq5}} \\ \widehat{\tau_{eq6}} \end{bmatrix}, M^{-1} = \begin{bmatrix} M_{11} & M_{12} & M_{13} & 0 & 0 & 0 \\ M_{21} & M_{22} & M_{23} & 0 & 0 & 0 \\ M_{31} & M_{32} & M_{33} & 0 & M_{35} & 0 \\ 0 & 0 & 0 & M_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & M_{55} & 0 \\ 0 & 0 & 0 & 0 & 0 & M_{66} \end{bmatrix}^{-1}$$

$$B + C + G = \begin{bmatrix} b_{112}\dot{q}_1\dot{q}_2 + b_{113}\dot{q}_1\dot{q}_3 + 0 + b_{123}\dot{q}_2\dot{q}_3 \\ 0 + b_{223}\dot{q}_2\dot{q}_3 + 0 + 0 \\ 0 \\ b_{412}\dot{q}_1\dot{q}_2 + b_{413}\dot{q}_1\dot{q}_3 + 0 + 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} C_{12}\dot{q}_2^{\;2} + C_{13}\dot{q}_3^{\;2} \\ C_{21}\dot{q}_1^{\;2} + C_{23}\dot{q}_3^{\;2} \\ C_{31}\dot{q}_1^{\;2} + C_{32}\dot{q}_2^{\;2} \\ 0 \\ C_{51}\dot{q}_1^{\;2} + C_{52}\dot{q}_2^{\;2} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ g_2 \\ g_3 \\ 0 \\ g_5 \\ 0 \end{bmatrix}$$

$$\dot{S} = \begin{bmatrix} \dot{S}_1 \\ \dot{S}_2 \\ \dot{S}_3 \\ \dot{S}_4 \\ \dot{S}_5 \\ \dot{S}_6 \end{bmatrix} \text{ and } M = \begin{bmatrix} M_{11} & M_{12} & M_{13} & 0 & 0 & 0 \\ M_{21} & M_{22} & M_{23} & 0 & 0 & 0 \\ M_{31} & M_{32} & M_{33} & 0 & M_{35} & 0 \\ 0 & 0 & 0 & M_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & M_{55} & 0 \\ 0 & 0 & 0 & 0 & 0 & M_{66} \end{bmatrix}$$

Suppose that τ_{sat} is computed as

$$\hat{\tau}_{sat} = K.sat\left(S/_{\emptyset}\right) \quad (40)$$

where

$$\hat{\tau}_{sat} = \begin{bmatrix} \widehat{\tau_{dis1}} \\ \widehat{\tau_{dis2}} \\ \widehat{\tau_{dis3}} \\ \widehat{\tau_{dis4}} \\ \widehat{\tau_{dis5}} \\ \widehat{\tau_{dis6}} \end{bmatrix}, K = \begin{bmatrix} K_1 \\ K_2 \\ K_3 \\ K_4 \\ K_5 \\ K_6 \end{bmatrix}, \left(S/_{\emptyset}\right) = \begin{bmatrix} \frac{S_1}{\emptyset_1} \\ \frac{S_2}{\emptyset_2} \\ \frac{S_3}{\emptyset_3} \\ \frac{S_4}{\emptyset_4} \\ \frac{S_5}{\emptyset_5} \\ \frac{S_6}{\emptyset_6} \end{bmatrix} \; and \;\; S = \lambda e + \dot{e}$$

Moreover by replace the formulation (40) in (38) the control output is written as ;

$$\hat{\tau} = \hat{\tau}_{eq} + K.sat\left(S/_{\emptyset}\right) = \begin{cases} \tau_{eq} + K.sgn(S) & , |S| \geq \emptyset \\ \tau_{eq} + K.S/_{\emptyset} & , |S| < \emptyset \end{cases} \quad (41)$$

Figure 1 shows the position classical sliding mode control for PUMA-560 robot manipulator. By (41) and (39) the sliding mode control of PUMA 560 robot manipulator is calculated as;

$$\hat{\tau} = \left[M^{-1}(B + C + G) + \dot{S}\right]M + K.sat\left(S/_{\emptyset}\right) \quad (42)$$

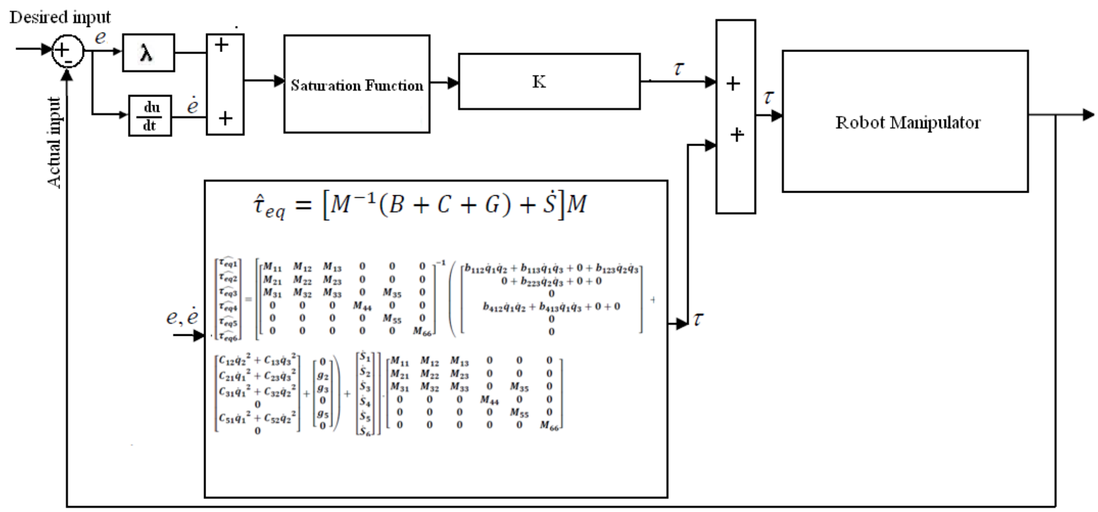

FIGURE 1: Block diagram of classical sliding mode controller

4. FPGA-BASED SLIDING MODE CONTROLLER

Research on FPGA-based control of systems is considerably growing as their applications such as industrial automation, robotic surgery, and space station's robot arm demand more accuracy, reliability, high performance. For instance, the FPGA-based controls of robot manipulator have been reported in [5-6, 8-13]. Shao and Sun [8, 10]have proposed an adaptive control algorithm based on FPGA for control of SCARA robot manipulator. They are designed this controller into two micro base controller, the linear part controller is implemented in the FPGA and the nonlinear estimation controller is implemented in DSP. Moreover this controller is implemented in a Xilinx-FPGA XC3S400 with the 20 KHz position loop frequency. The FPGA based servo control and inverse kinematics for Mitsubishi RV-M1 micro robot is presented in[9, 11-12] which to reduce the limitation of FPGA capacitance they are used 42 steps finite state machine (FSM) in 840 n second. Meshram and Harkare [5-6] have presented a multipurpose FPGA-based 5 DOF robot manipulator using VHDL coding in Xilinx ISE 11.1. This controller has two most important advantages: easy to implement and flexible. Zeyad Assi Obaid et al. [13] have proposed a digital PID fuzzy logic controller using FPGA for tracking tasks that yields semi-global stability of all closed-loop signals.

The basic information about FPGA has been reported in [4-5, 12-15]. A review of design and implementation of FPGA-based systems has been presented in [4]. The FPGA-based sliding mode control of systems has been reported in [7, 16-18]. Lin et al. [7] have presented low cost and high performance FPGA-based fuzzy sliding mode controller for linear induction motor with 80% of flip flops. The fuzzy inference system has 2 inputs (S & $\dot{S}$) and one output K_f with nine rules. Ramos et al. [16] have reported FPGA-based fixed frequency quasi sliding mode control algorithm to control of power inverter. Their proposed controller is implemented in XC4010E-3-PC84 FPGA from XILINX with acceptable experimental and theoretical performance. FPGA-based robust adaptive backstepping sliding mode control for verification of induction motor is reported in [17].

The introduction of language and architecture of Xilinx FPGA such as VHDL or Verilog in sliding mode control of robot manipulator will be investigated in this section. The Xilinx Spartan 3E FPGAs has 5 major blocks: Configurable Logic Blocks (CLBs), standard and high speed Input/output Blocks (IOBs), Block RAM's (BRAMs), Multipliers Blocks, and Digital Clock Managers (DCMs). CLBs is include flexible look up tables (LUTs) to implement memory (storage element) and logic gates. There are 4 slices per CLB each slice has two LUT's. IOB does control the rate of data between input/output pins and the internal logic gates or elements.

It supports bidirectional data with three state operation and multiplicity of signal standards. BRAMs require the data storage including 18-Kbit dual-port blocks. Product two 18-bit binary numbers is done by multiplier blocks. Self-calibrating, digital distributing solution, delaying, multiplying, dividing and phase-shift clock signal are done by DCM [15].

As shown in Figure 1, FPGA based sliding mode controller divided into two main parts: saturation part and equivalent part. To design FPGA based SMC controller using VHDL code, inputs and outputs is played important role. The block diagram of the FPGA-based sliding mode control systems for a robot manipulator is shown in Figure 2. Based on Figure 2 this block (controller) has 9 inputs and 3 outputs. Actual and desired displacements (inputs) are defined as 30 bits and the outputs (teta_dis) are defines as 35 bits in size. The desired inputs are generated from the operator and send to controllers for calculate the error and applied to sliding mode controller.

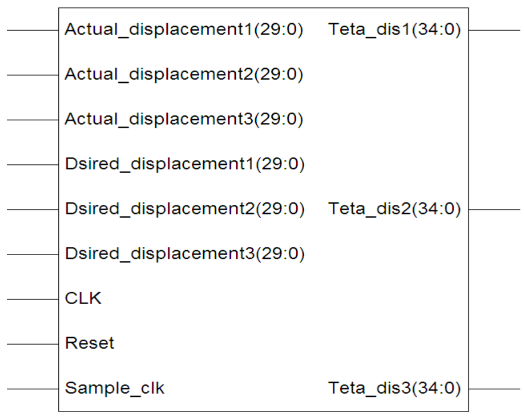

FIGURE 2 : RTL FPGA-based controller schematic in XILINX-ISE

To convert float input data to the integer it should be multiply input value by 1000000 and then save these new values in the input files. After the completing simulation, output response should be divided over 1000000 integers to real convert values. But due to simulator (XILINX ISE 9.1) limitations and restrictions on integer data length (32 bits) and it results are 33 bit's words so at the first, controller results is divided over 2 and convert them to the integer part. Therefore the result should be divided over 500000 instead of 1000000. To robot manipulator's FPGA based position sliding mode control, controller is divided into three main sub blocks; Figure 3 shows the VHDL code and RTL schematic in Xilinx ISE software.

The table in Figure 4 indicates the Summary of XA Spartan-3E FPGA Attributes. As mentioned in above, the most significant resources are the LUT's (610 out of 29504), CLB (77 out of 3688), Slice (305 out of 14752), Multipliers (27 out of 36), registers (397), and Block RAM memory (648 K) which there are 4 slices per CLB, each slice has two LUT's. So, Number of 4 input LUTs=610, $\frac{610}{2} = 305$ slices, $\frac{305}{4} \cong 77$ CLB's, 610 registers and as a Map report Peak memory usage is 175 MB and registers in the XA3S1600E FPGA.
Moreover the table in Figure 5 illustrates the utilization summary of XA3S1600E-spartan.

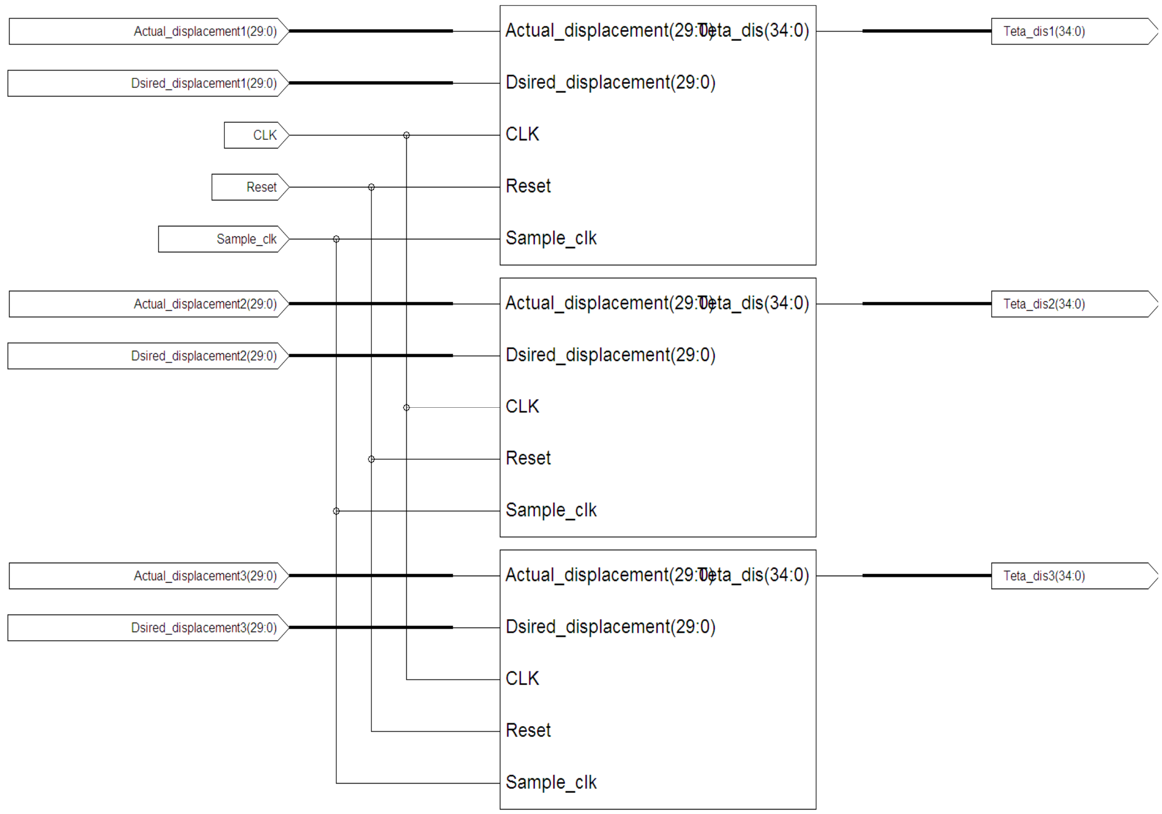

FIGURE 3: Design RTL FPGA-based SMC using XILINX-ISE

Device	System Gates	Equivalent Logic Cells	CLB Array (One CLB = Four Slices) Rows	Columns	Total CLBs	Total Slices	Distributed RAM bits[1]	Block RAM bits[1]	Dedicated Multipliers	DCMs	Maximum User I/O	Maximum Differential I/O Pairs
XA3S100E	100K	2,160	22	16	240	960	15K	72K	4	2	108	40
XA3S250E	250K	5,508	34	26	612	2,448	38K	216K	12	4	172	68
XA3S500E	500K	10,476	46	34	1,164	4,656	73K	360K	20	4	190	77
XA3S1200E	1200K	19,512	60	46	2,168	8,672	136K	504K	28	8	304	124
XA3S1600E	1600K	33,192	76	58	3,688	14,752	231K	648K	36	8	376	156

Notes:

1. By convention, one Kb is equivalent to 1,024 bits.

FIGURE 4: Summary of XA Spartan-3E FPGA attributes

5. RESULTS

PD Matlab-based sliding mode controller (PD-SMC) and PID Matlab-based sliding mode control ler (PID-SMC) and FPGA-based sliding mode controller were tested to Step response trajectory. In this simulation the first, second, and third joints are moved from home to final position without and with external disturbance. The simulation was implemented in Matlab/Simulink and Xilinx-ISE 9.1 environments. Trajectory performance, torque performance, disturbance rejection, steady state error and RMS error are compared in these controllers. It is noted that, these systems are tested by band limited white noise with a predefined 40% of relative to the input signal amplitude which the sample time is equal to 0.1. This type of noise is used to external disturbance in continuous and hybrid systems.

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	216	29,504	1%	
Number of 4 input LUTs	610	29,504	2%	
Logic Distribution				
Number of occupied Slices	342	14,752	2%	
Number of Slices containing only related logic	342	342	100%	
Number of Slices containing unrelated logic	0	342	0%	
Total Number of 4 input LUTs	622	29,504	2%	
Number used as logic	610			
Number used as a route-thru	12			
Number of bonded IOBs	288	376	76%	
IOB Flip Flops	181			
Number of GCLKs	2	24	8%	
Number of MULT18X18SIOs	27	36	75%	
Total equivalent gate count for design	10,334			
Additional JTAG gate count for IOBs	13,824			

Figure 5 XA3S1600E device utilization summaries

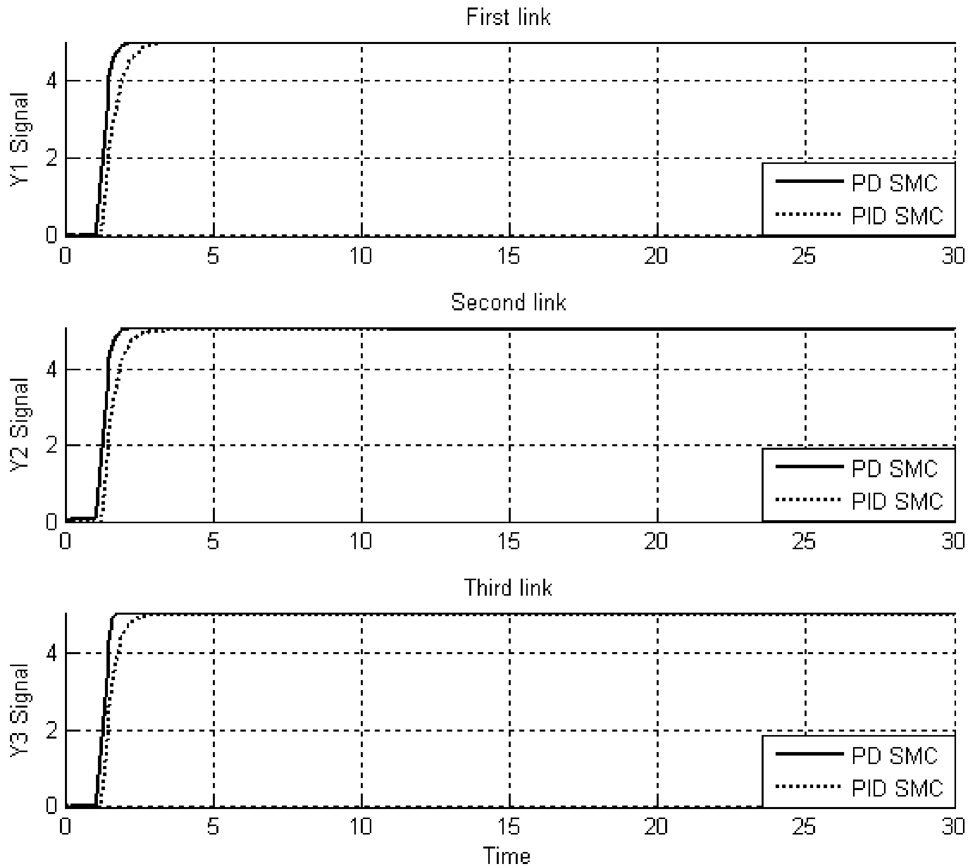

FIGURE 6 : Step PD-SMC and PID-SMC for first, second and third link trajectory without any disturbance.

By comparing step response, Figure 6, in PD and PID-SMC, conversely the PID's overshoot **(0%)** is lower than PD's **(1%)**, the PD's rise time **(0.483 Sec)** is dramatically lower than PID's **(0.9 Sec)**; in addition the Settling time in PD **(Settling time=0.65 Sec)** is fairly lower than PID **(Settling time=1.4 Sec)**.

Disturbance rejection: Figure 7 is indicated the power disturbance removal in PD and PID-SMC. As mentioned before, SMC is one of the most important robust nonlinear controllers. Besides a band limited white noise with predefined of 40% the power of input signal is applied to the step PD and PID-SMC; it found slight oscillations in trajectory responses.

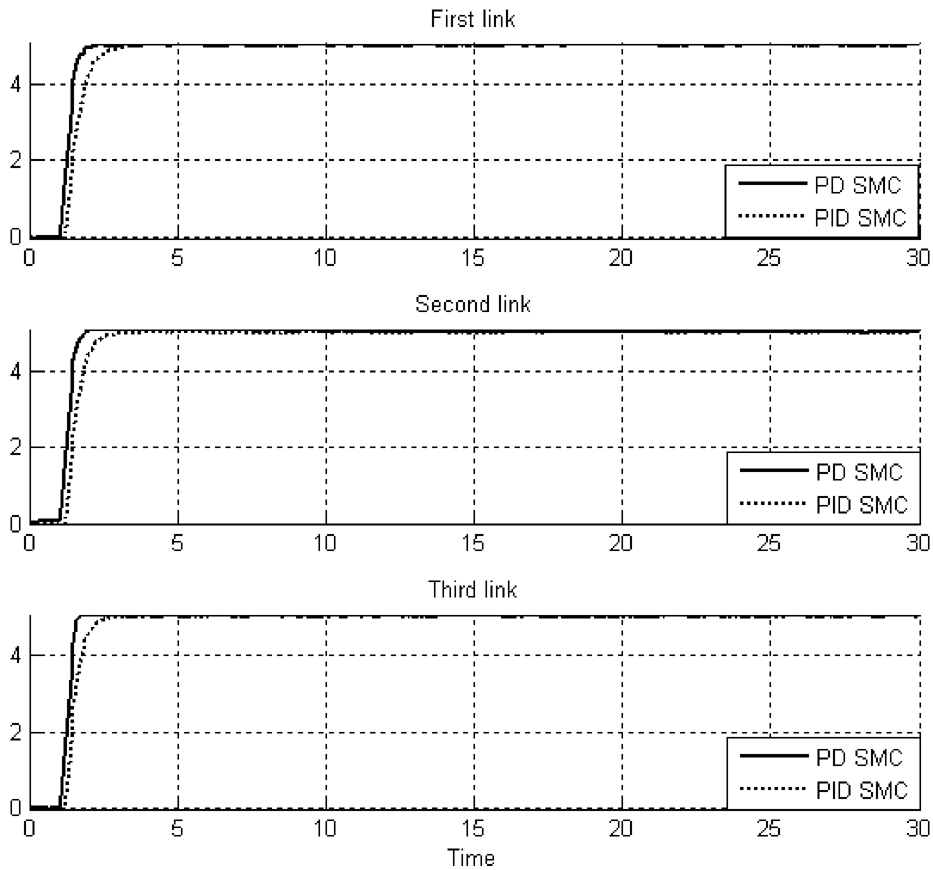

FIGURE 7: Step PD SMC and PID SMC for first, second and third link trajectory with external disturbance.

Among above graph, relating to step trajectory following with external disturbance, PID and PD SMC have slightly fluctuations. By comparing overshoot, rise time, and settling time; PID's overshoot **(0.9%)** is lower than PD's **(1.1%)**, PD's rise time **(0.48 sec)** is considerably lower than PID's **(0.9 sec)** and finally the Settling time in PD **(Settling time=0.65 Sec)** is quite lower than PID **(Settling time=1.5 Sec)**.

Chattering phenomenon: As mentioned in previous section, chattering is one of the most important challenges in sliding mode controller which one of the major objectives in this research is reduce or remove the chattering in system's output. Figure 8 has shown the power of boundary layer (saturation) method to reduce the chattering in PD-SMC.

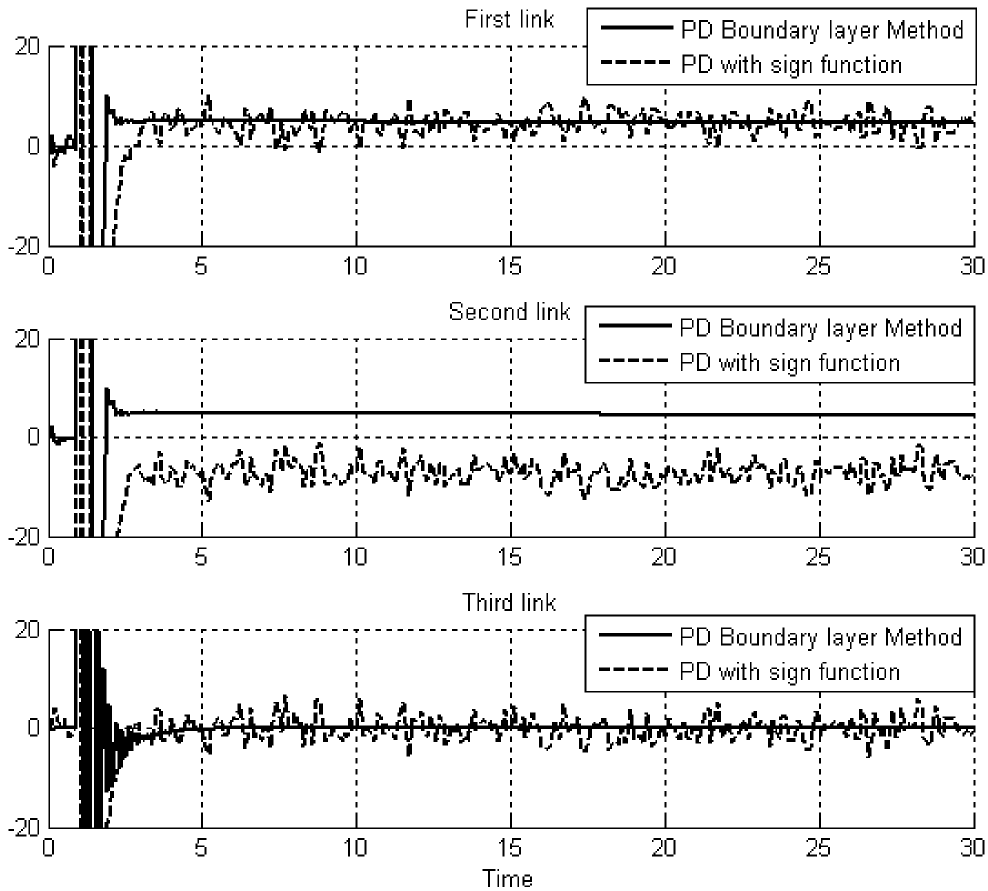

FIGURE 8 : PD-SMC boundary layer methods Vs. PD-SMC with discontinuous (Sign) function

Figure 9 has indicated the power of chattering rejection in PD and PID-SMC, with and without disturbance. As mentioned before, chattering can caused to the hitting in driver and mechanical parts so reduce the chattering is more important. Furthermore band limited white noise with predefined of 40% the power of input signal is applied the step PD and PID-SMC, it seen that the slight oscillations in third joint trajectory responses. Overall in this research with regard to the step response, PD-SMC has the steady chattering compared to the PID-SMC.

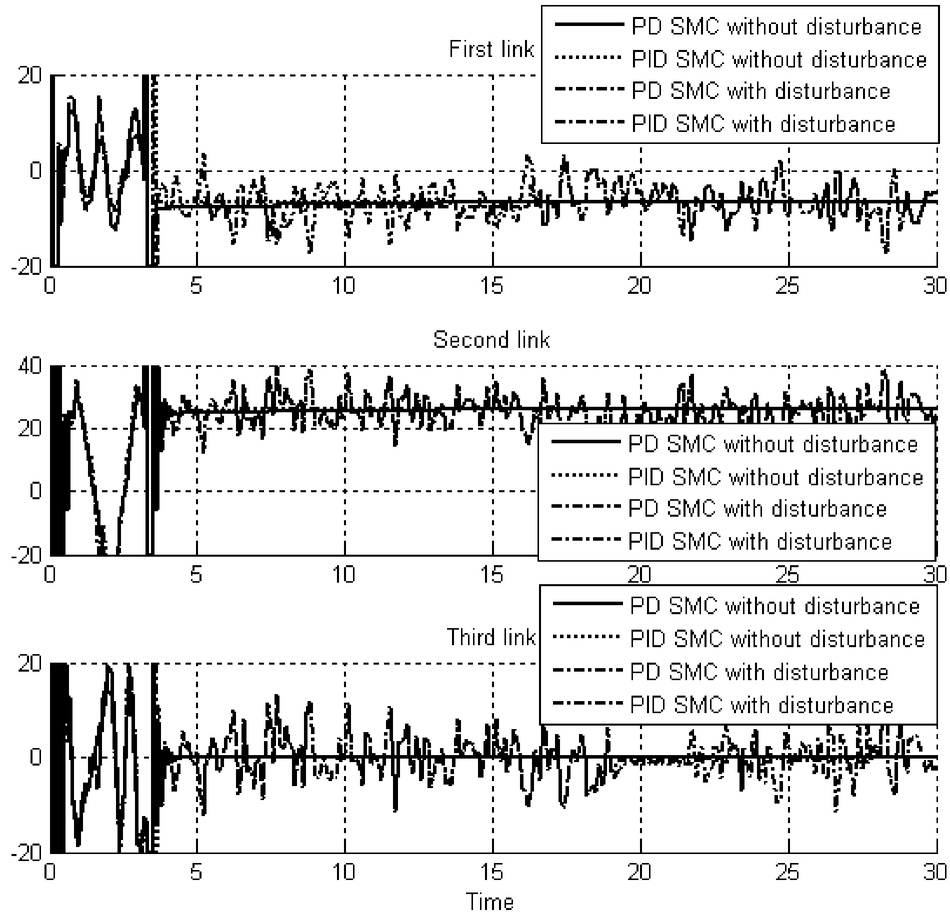

FIGURE 9 : Step PID SMC and PD SMC for first, second and third link chattering without and with disturbance.

Errors in the model: Figure 10 has shown the error disturbance in PD and PID SMC. The controllers with no external disturbances have the same error response, but PID SMC has the better steady state error. By comparing steady and RMS error in a system with no disturbance it found that the PID's errors **(Steady State error = 0 and RMS error=1e-8)** are approximately less than PD's **(Steady State error $\cong 1e-6$ and RMS error=$1.2e-6$)**.

Figure 10 shows that in first seconds; PID SMC and PD SMC are increasing very fast. By comparing the steady state error and RMS error it found that the PID's errors **(Steady State error = -0.0007 and RMS error=0.0008)** are fairly less than PD's **(Steady State error $\cong 0.0012$ and RMS error=0.0018)**, When disturbance is applied to PD and PID SMC the errors are about 13% growth.

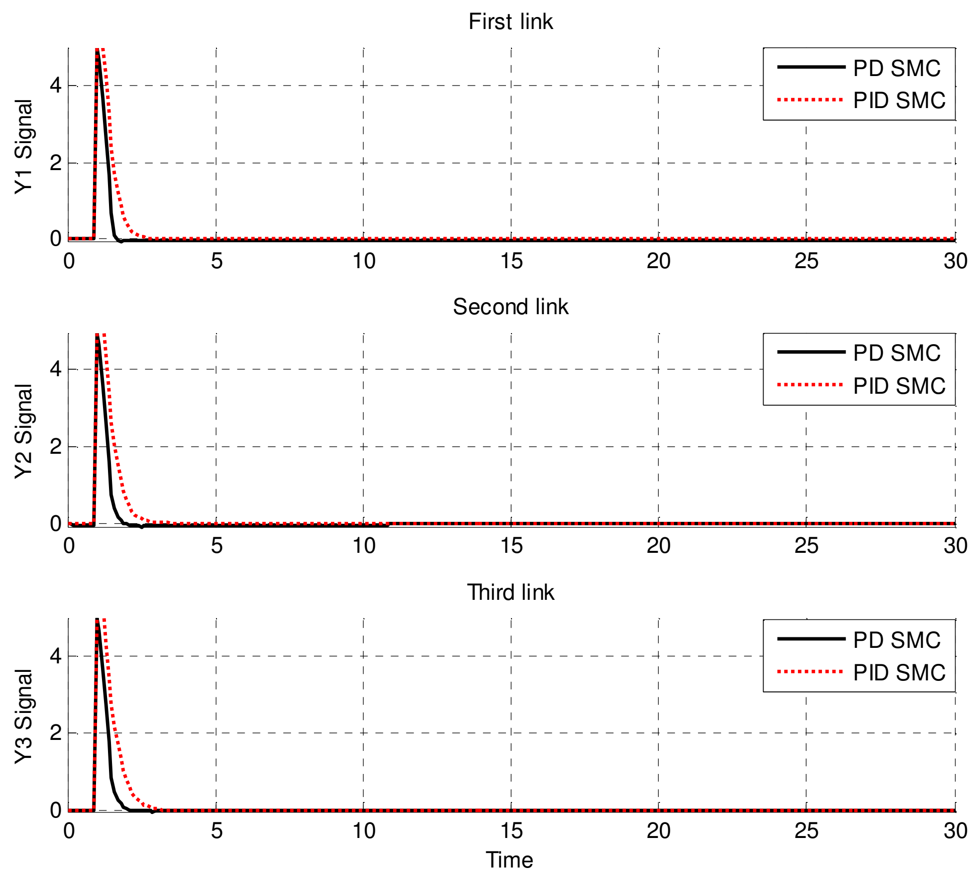

FIGURE 10 : Step PID SMC and PD SMC for first, second and third link steady state error performance.

5.2 FPGA-Based Sliding Mode Controller

Timing Detail: As a simulation result in XILINX-ISE 9.1, it found that this controller is able to make as a fast response at 15.716 ns with 63.29 MHz of a maximum frequency. From investigation and synthesis summary, this design has 15.716 ns delay to each controller for 46 logic elements and also the offset before CLOCK is 55.773 ns for 132 logic gates. Figures 11 to 13 have indicated the displacement, error performance, teta discontinuous (torque performance) at different time.

As shown in Figure 11 the controller gives action at 6 μs as a result before this time all signals and error equal to zeros.

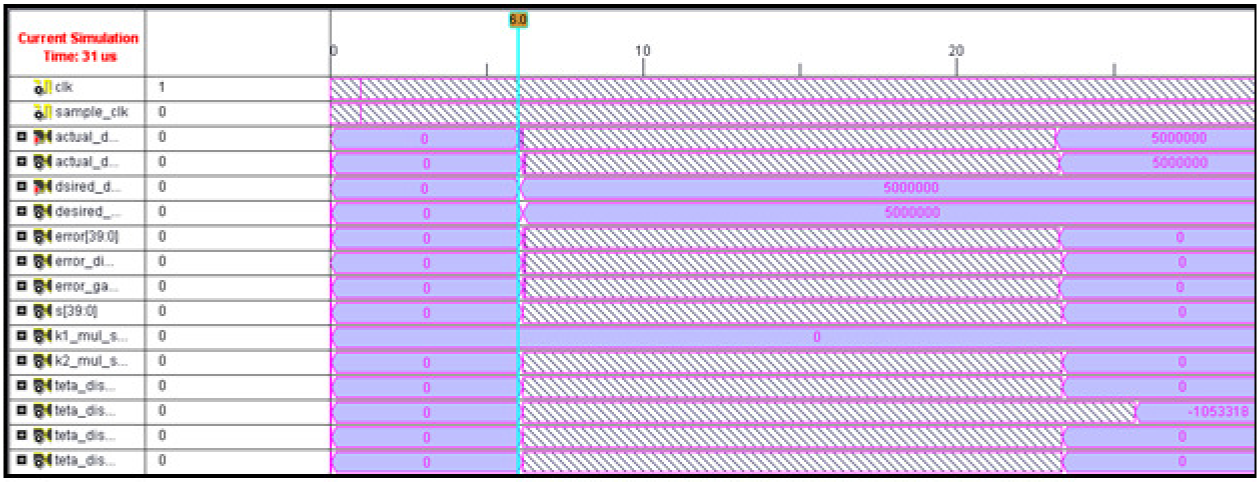

FIGURE 11 : Timing diagram using Xilinx ISE 9.1 of the FPGA-based SMC before running

In Figure 12 at 6.5 μs (transient response) the response has a large steady state error, 3.92, the desired displacement is 5, the actual displacement is 1.6 and the torque performance is 256.9 N.m.

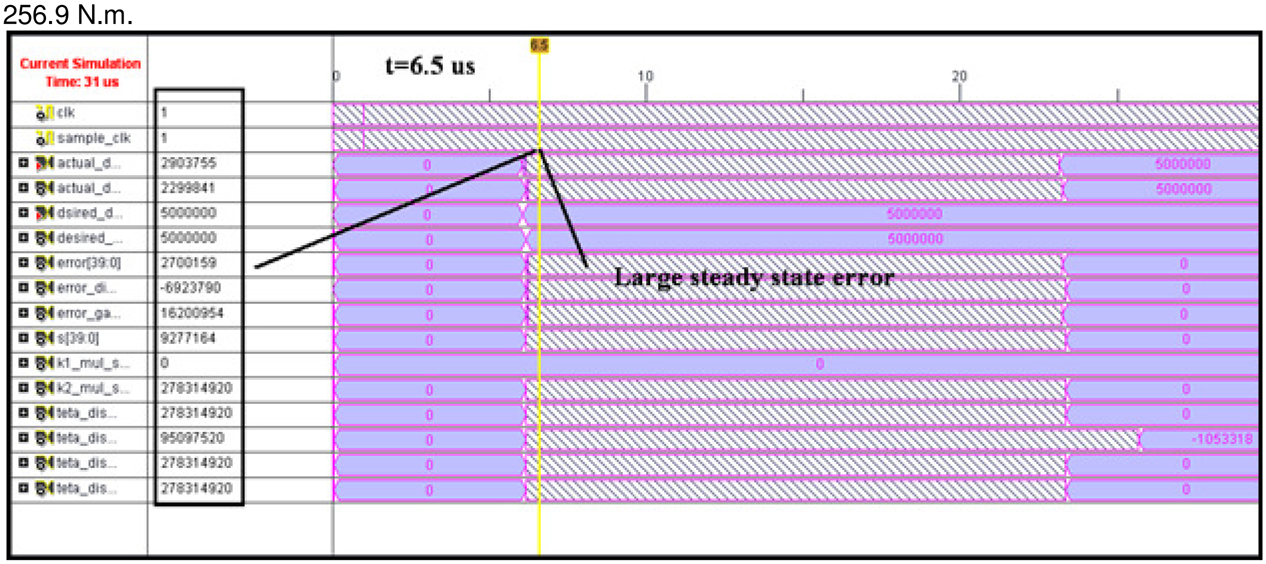

FIGURE 12: Step PD SMC for first, second and third link for desired and actual inputs, error performance, and torque performance at 6.5 μs

Figure 13 has shown the PD-SMC at t=100$\mu\, s$ (steady state response), at this time the steady State error is equal to zero **,** the desired displacement is 5, the actual displacement is 5 and the torque performance is 1.005 Nm.

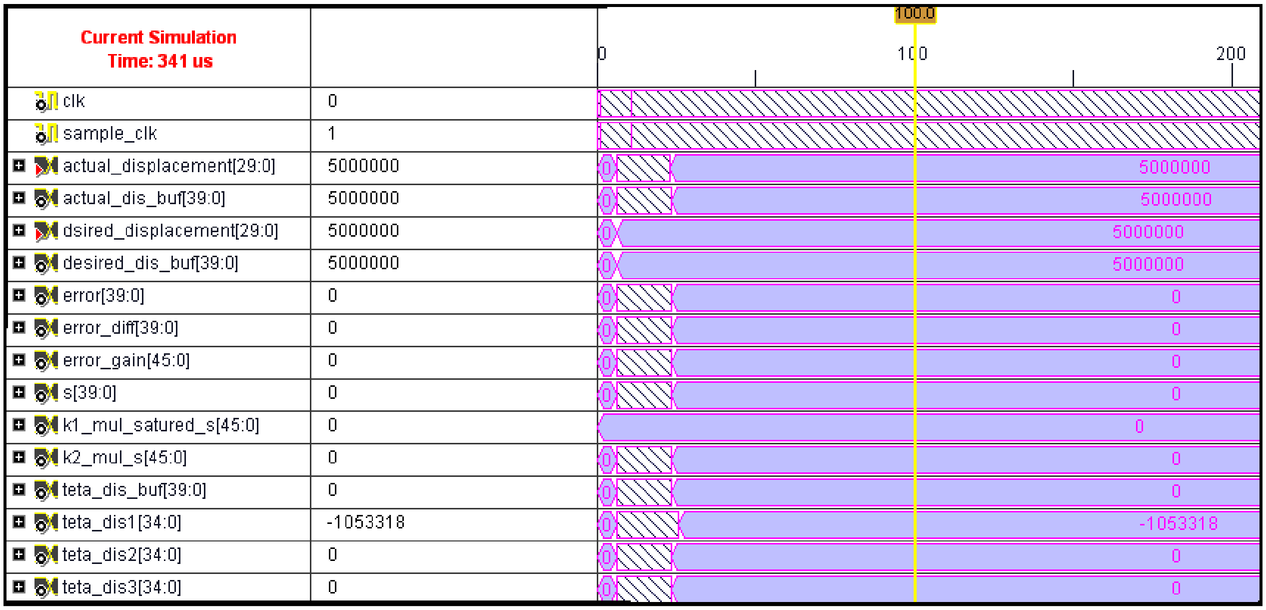

FIGURE 13 : Step PD SMC for first, second and third link for desired and actual inputs, error performance, and torque performance in 100μs.

Figure 14 shows the delay with the robot manipulator affects the beginning of the response. Consequently the delay for this system is equal to 0.1μ s.

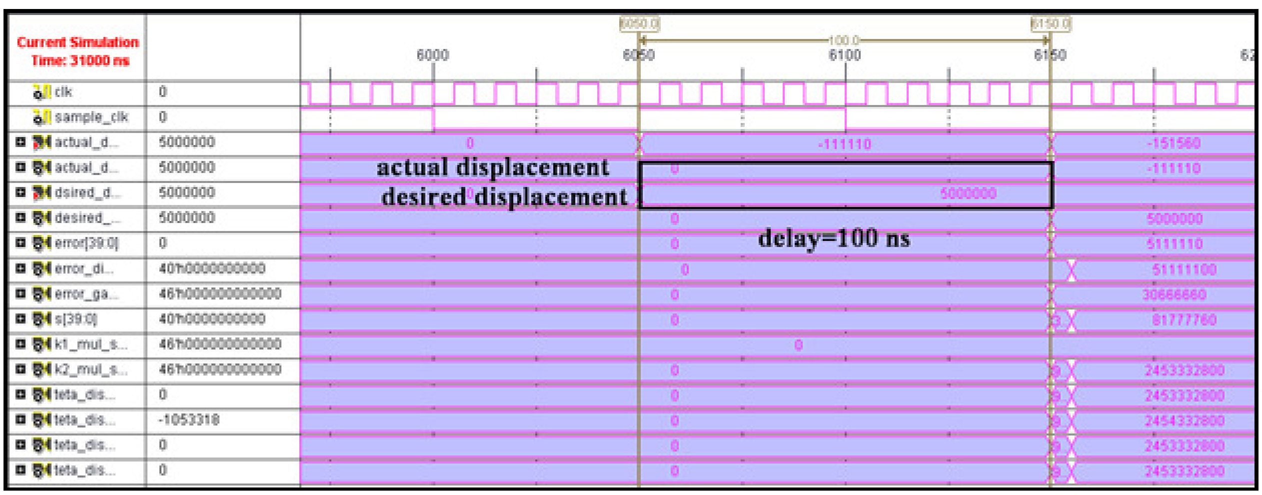

FIGURE 14 : The delay time in PD-SMC between desired displacement and actual displacement

Figures 15 and 16 show the chattering in FPGA-based SMC. In Figure 15, the chattering analysis from 6.2 μs to 7μs. It can be seen that the chattering is eliminated in this design.

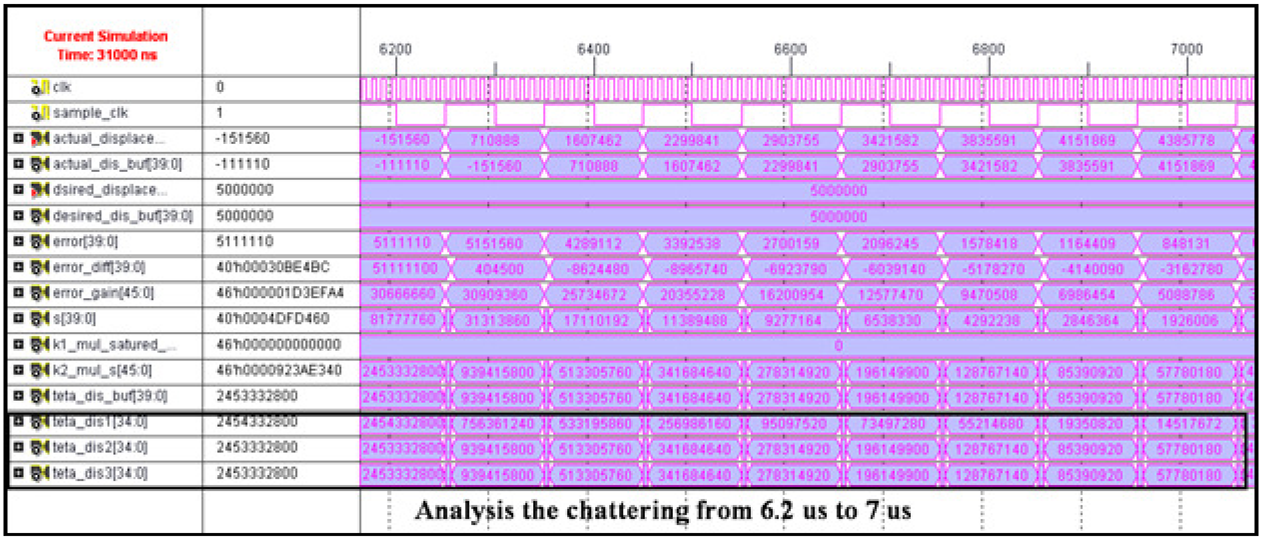

FIGURE 15: Chattering rejections in FPGA-based SMC (from 6.2 μs to 7μs)

Figure 16 shows the power of chattering rejections in FPGA-based SMC, it found that this design is eliminated the chattering in certain situation as well as Matlab-based PD SMC.

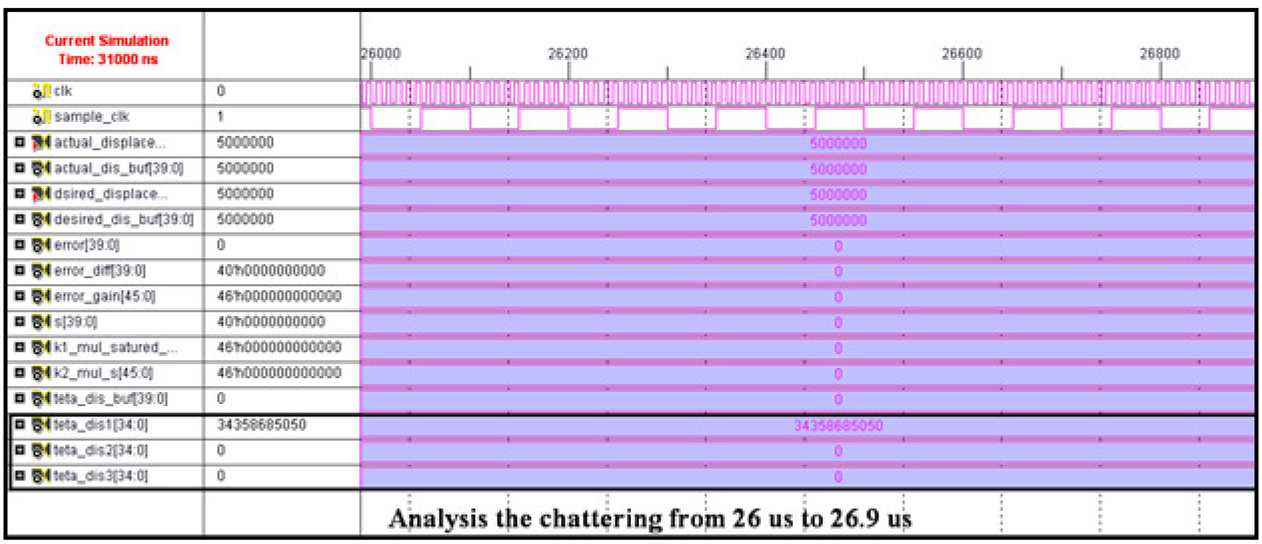

FIGURE 16: Chattering rejections in FPGA-based SMC (from 26 μs to 26.9μs)

The best possible coefficients in Step FPGA-based PD-SMC are; $K_p = 000001 = 1, K_v = 011110 = 30,\ \emptyset_1 = \emptyset_2 = \emptyset_3 = 000001 = 1, and\ \lambda_1 = \lambda_2 = \lambda_3 = 000110 = 6$. By comparing some control parameters such as overshoot, rise time, settling time and steady state error in Matlab–based PD-SMC, FPGA-based PD-SMC; overshoot (**PD-SMC=1% and FPGA-SMC=0.005%**), rise time (**PD-SMC=0.4 sec and FPGA-SMC8.2 μ s),** settling time (**PD-SMC=0.4 sec and FPGA-SMC=10 μ s)** and steady state error **(PD-SMC $\cong 0.0003$ and FPGA-SMC=0)** consequently it found that in fast response, the FPGA based-SMC's parameter has the high-quality performance.

6. CONCLUSION

Refer to the research, a position FPGA-based sliding mode control design and application to robot manipulator has proposed in order to design high performance nonlinear controller in the presence of certainties. Regarding to the positive points in sliding mode controller and FPGA the output has improved. Sliding mode controller by adding to the FPGA single chip IC has covered negative points. Obviously PUMA 560 robot manipulator is nonlinear so this paper focuses on comparison between Matlab-based sliding mode controller and FPGA-based sliding mode controller, to opt for mobility control method for the industrial manipulator.

Higher implementation speed and small chip size versus an acceptable performance is reached by designing FPGA-based sliding mode controller. This implementation considerably reduces the chattering phenomenon and error in the presence of certainties. The controller works with a maximum clock frequency of 63.29 MHz and the computation time (delay in activation) of this controller is $0.1\mu s$. As a result, this controller will be able to control a wide range of robot manipulators with a high sampling rates because it's small size versus high speed markets.

REFERENCES:

[1] T. R. Kurfess, *Robotics and automation handbook*: CRC, 2005.

[2] B. Siciliano and O. Khatib, *Springer handbook of robotics*: Springer-Verlag New York Inc, 2008.

[3] O. Kaynak, "Guest editorial special section on computationally intelligent methodologies and sliding-mode control," *IEEE Transactions on Industrial Electronics,* vol. 48, pp. 2-3, 2001.

[4] N. Sulaiman*, et al.*, "Design and Implementation of FPGA-Based Systems-A Review," *Australian Journal of Basic and Applied Sciences,* vol. 3, pp. 3575-3596, 2009.

[5] U. Meshram*, et al.*, "Robot arm controller using FPGA," 2009, pp. 8-11.

[6] U. D. Meshram and R. Harkare, "FPGA Based Five Axis Robot Arm Controller."

[7] F. J. Lin*, et al.*, "FPGA-based fuzzy sliding-mode control for a linear induction motor drive," 2005, pp. 1137-1148.

[8] X. Shao and D. Sun, "Development of an FPGA-based motion control ASIC for robotic manipulators," 2006, pp. 8221-8225.

[9] Y. S. Kung*, et al.*, "FPGA-implementation of inverse kinematics and servo controller for robot manipulator," *Proc. IEEE Int. on Robotics and Biomimetics,* pp. 1163–1168, 2006.

[10] X. Shao*, et al.*, "A new motion control hardware architecture with FPGA-based IC design for robotic manipulators," 2006, pp. 3520-3525.

[11] Y. S. Kung*, et al.*, "Design and Implementation of a Servo System for Robotic Manipulator," ed: CACS, 2005.

[12] Y. S. Kung and G. S. Shu, "Development of a FPGA-based motion control IC for robot arm," 2006, pp. 1397-1402.

[13] Z. A. Obaid*, et al.*, "Developed Method of FPGA-based Fuzzy Logic Controller Design with the Aid of Conventional PID Algorithm," *Australian Journal of Basic and Applied Sciences,* vol. 3, pp. 2724-2740, 2009.

[14] S. T. Karris, *Digital circuit analysis and design with Simulink modeling and introduction to CPLDs and FPGAs*: Orchard Pubns, 2007.

[15] K. D. Rogers, "ACCELERATION AND IMPLEMENTION OF A DSP PHASE-BASED FREQUENCY ESTIMATION ALGORITHM: MATLAB/SIMULINK TO FPGA VIA XILINX SYSTEM GENERATOR," Citeseer, 2004.

[16] R. R. Ramos*, et al.*, "A fixed-frequency quasi-sliding control algorithm: application to power inverters design by means of FPGA implementation," *Power Electronics, IEEE Transactions on,* vol. 18, pp. 344-355, 2003.

[17] F. J. Lin*, et al.*, "FPGA-based adaptive backstepping sliding-mode control for linear induction motor drive," *Power Electronics, IEEE Transactions on,* vol. 22, pp. 1222-1231, 2007.

[18] S. Lentijo*, et al.*, "FPGA based sliding mode control for high frequency power converters," 2004, pp. 3588-3592.