International Journal of Hybrid Information Technology
Vol. 9, No. 5 (2016), pp. 71-98
http://dx.doi.org/10.14257/ijhit.2016.9.5.07

Design Single Chip Micro-Based Controller for First Order Delays System

Saman Namvarrechi, Samira Soltani, Iman Nazari, Ali Roshanzamir and Arman Jahed

Intelligent Systems and Robotics Lab, Iranian Institute of Advanced Science and Technology (IRAN SSP), Shiraz/Iran
info@iranssp.org, WWW.IRANSSP.ORG

Abstract

Design a delay free FPGA-based Proportional-Integral-Derivative (PID) controller to control of first order delay system is the main objective in this research. In order to provide high performance micro-based controller, FPGA-based PID controller based on design Derivative and Integral algorithm is selected. Conventional PID controller is a stable linear type model-free controller that reduces the delay time in first order delay system. This controller has acceptable performance in presence of uncertainty (e.g., overshoot=0%, rise time=0.8 second, steady state error = 1e-9 and RMS error=1.8e-12). In this research, linear controller need real time mobility operation, and 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 HDL based derivative algorithm the minimum input arrival time before clock is 16.466 ns and the maximum frequency is 60.73 MHz, but in the best design action, the maximum frequency to design this single chip algorithm should be 63.629 MHz. In HDL integral algorithm the minimum input arrival time before clock is 15.599 ns and the maximum frequency is 64.1 MHz, but in the best design action, the maximum frequency to design this single chip algorithm should be 178.190 MHz.

Keywords: *First order delays system, HDL derivative algorithm, HDL integral algorithm, FPGA-based PID algorithm, minimum delay time*

1. Introduction and Background

A Field Programmable Gate Array (FPGAs) is a small Field Programmable Device (FPD) that supports thousands of gates. FPGAs are divided in two categories: SRAM-based FPGA, and Antifuse-based FPGA which SRAM-based FPGA used many semiconductor and consists of an array of logic element, routing paths, FPGA I/O pins, on chip memory, and the other resource and Antifuse-Based FPGAs used two metal layers to sandwich the layere of non-conductive silicon. A typical SRAM- Based FPGAs and Antifuse-Based FPGAs have shown in Figures 1 and Figure 2.

ISSN: 1738-9968 IJHIT

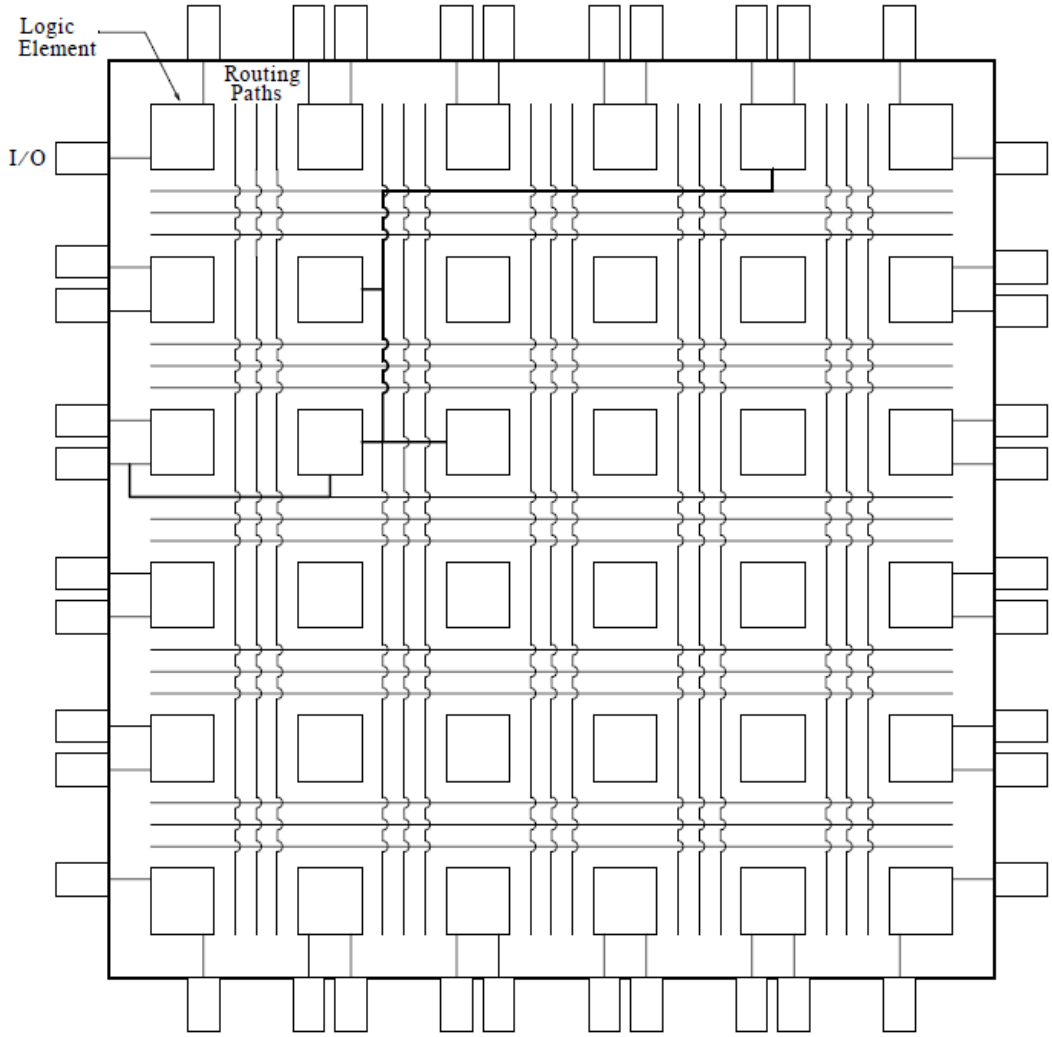


Figure 1. SRAM-Based FPGA

Several semiconductor vendors provides a wide range of FPGAs such as Xilinx, Altera, Atmel and Lattice that each one has own unique architecture. In most of FPGAs, logic elements consist of one or more RAM-based n input lookup tables, and one or more Flip-Flops. FPGAs can be used in wide range area such as, Fast Fourier Transforms (FFT), Discrete Cosine Transforms (DCT), Convolution, and Finite Impulse Response (FIR) filters. A FPGA chip can be programmed by Hardware Description Language (HDL) and HDL contains two type of language, Very High Description Language (VHDL) and Verilog which VHDL is one of the powerful programming language that can be used to describe the hardware design. VHDL was developed by the Institute of Electrical and Electronics Engineers (IEEE) in 1987 and Verilog was developed by Gateway Design Automation in 1984. This language became an IEEE standard in 1995 and was updated in 2001 [1-6].

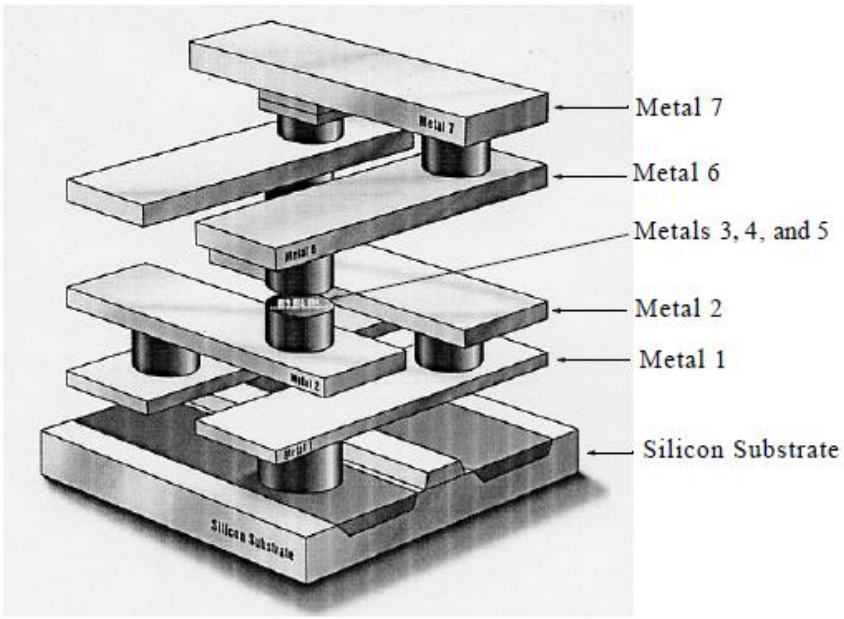


Figure 2. Antifuse-Based Fpgas

In order to provide information about implementing SMC using HDL on Xilinx FPGAs, this part present introducing of the Xilinx architecture. The Xilinx FPGAs has 6 major blocks namely; Configuration Logic Blocks (CLB'S), Block RAM's(B RAMS); multipliers; Digital Clock Managers (DCM'S); standard, and high speed I/O (IOB's), Figure 3, that can be connected to each other by fully buffered SRAM programmable switching matrix. The switching matrix is programmed and controlled by data of the configuration on loaded in to SRAM. The CLB take up over 75% of area resourse, so all of the other blocks related to the CLB array size. The BRAM's and multipliers are in a narrow space between the CLB's. The DCM's are blocks at the bottom and the top of each part of BRAM's and multipliers. The IOB's are parallel to serial embedded transceiver that can be used for high speed interfaces between multiple FPGA's.

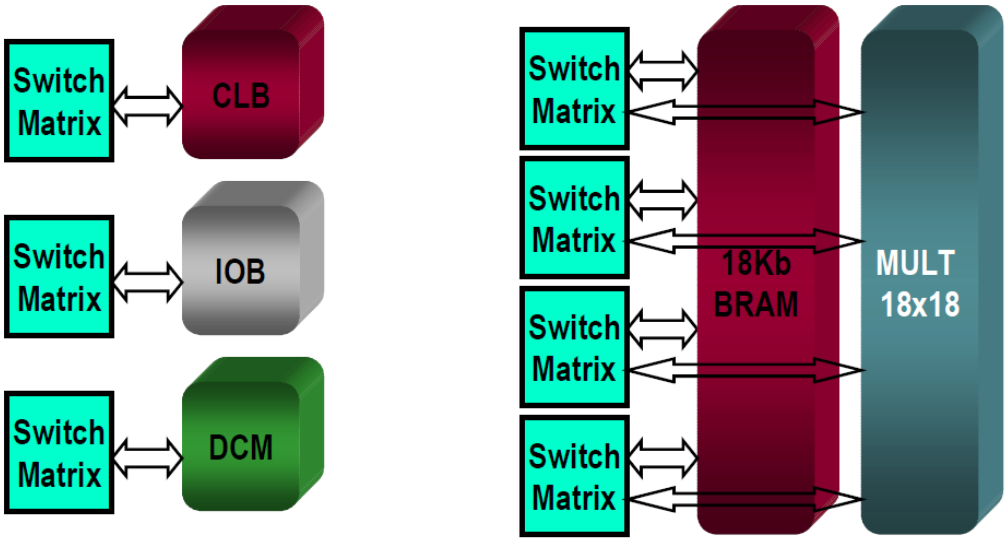


Figure 3. Block Diagram of Xilinx FPGA Architecture

System or plant is a set of components which work together to follow a certain objective. In this research, first order delay is system. First order delay (FOD) system is a type of nonlinear and time variant system [7-10].

A controller (control system) is a device, which cans sense information from system to improvement the dynamic behavior of first order delay system based on actuation and computation. From a scientific perspectives control theory is divided into two parts; linear control theory and nonlinear control theory. Linear control theory is divided into following groups:

- Proportional-Derivative (PD) control algorithm
- Proportional-Integral (PI) control algorithm
- Proportional-Integral-Derivative (PID) control algorithm

Nonlinear control theory is also has two main divisions;

- Conventional control theory
- Soft computing control theory

Linear control theory is used in linear and nonlinear systems. This type of theory is used in industries, because design of this type of controller is simple than nonlinear controller. Proportional algorithm is used to responds immediately to difference of control input variables by immediately changing its influences variables, but this type of control is unable to eliminate the control input difference. Figure 4 shows the block diagram of proportional controller [11-15].

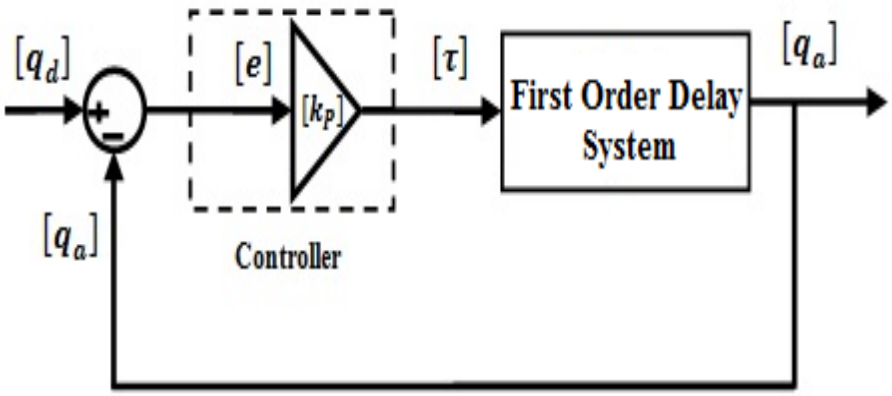


Figure 4. Block diagram of Proportional Algorithm

Derivative category: derivatives the input signal deviation over a period of time. This part of controller is used to system speed (rate of input signal) in a short time. Figure 5 shows the block diagram of derivative (D) controller with application to FOD system. In mathematically, the formulation of derivative part calculated as follows;

$$\boldsymbol{D} = \frac{\boldsymbol{d}}{\boldsymbol{dt}}(\boldsymbol{e}) = \dot{\boldsymbol{e}} \quad (1)$$

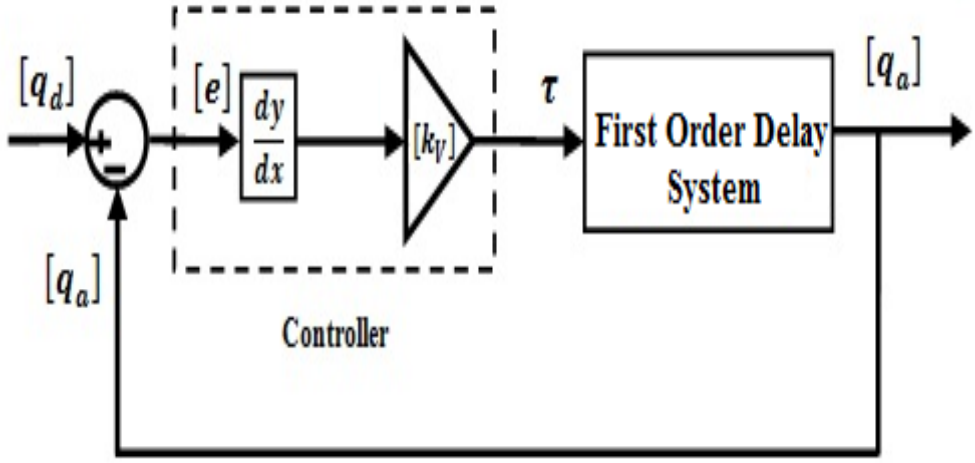


Figure 5. Block Diagram of Derivative Algorithm

Integral Algorithm: This category, integrate the input signal deviation over a period of time. This part of controller is used to system stability after a long period of time. Figure 6 shows the block diagram of Integral (I) controller with application to first order delay system. In contrast of Proportional type of controller, this type of controller used to eliminate the deviation. In mathematically, the formulation of integral part calculated as follows;

$$\boldsymbol{I = \frac{1}{T}\int e.dt = \sum e} \quad (2)$$

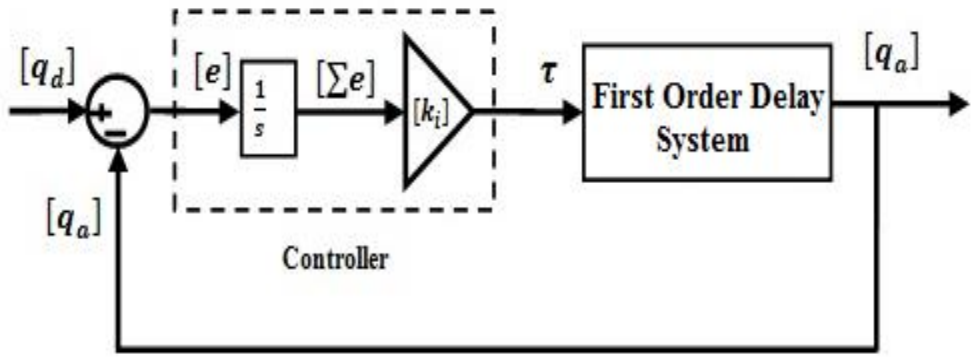


Figure 6. Block Diagram of Integral Control of FOD System

Figure 7 shows the step response of Integral Controller.

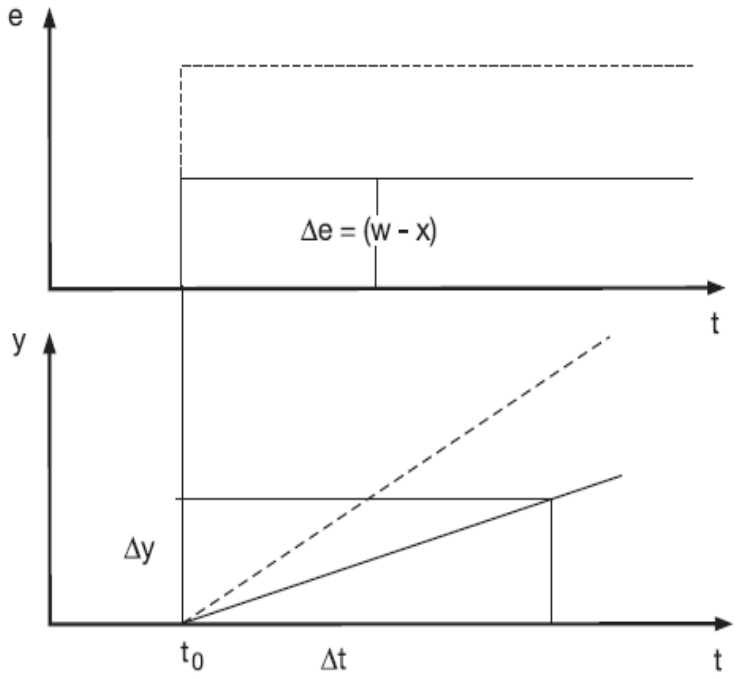


Figure 7. Step Response of an Integral (I) Controller

In this paper, FPGA based PID control algorithm is investigated. To design FPGA-based PID controller, SPARTAN 3E-XA3S1600E is used. The information of this device is introduced as the following Table (Table 1).

Table 1. Summary of XA Spartan-3E FPGA Attributes

Device	System Gates	Equivalent Logic Cells	CLB Array (One CLB = Four Slices)				Distributed RAM bits(1)	Block RAM bits(1)	Dedicated Multipliers	DCMs	Maximum User I/O	Maximum Differential I/O Pairs
			Rows	Columns	Total CLBs	Total Slices						
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.

This paper is organized as follows; second part focuses on the system modeling dynamic formulation and Design PD, PI controller. Third part is focused on the methodology. Simulation result and discussion is illustrated in forth part. The last part focuses on the conclusion and compare between this method and the other ones.

2. Theory

Many industrial processes can be represented by a first order model; equation (3) shows the mathematical plant model (in *s-plane*). Discrete transfer function of this model has obtained using ZOH method, and the selected sampling period (T) is 0.1, equation (4) shows the discrete transfer functions, (in *z-plane*).

$$CS_1(s) = \frac{1}{S+1} \tag{3}$$

and;

$$CS_1(z) = \frac{0.09516}{Z-0.9048}\ , T = 0.1 \tag{4}$$

The time delay occurs when a sensor or an actuator are used with a physical separation. Equation (5) shows the mathematical plant model (in *s-plane*). Discrete transfer functions of this model has been obtained using ZOH method, and the selected sampling period (T) is 0.1, equation (6 and 7) show the discrete transfer functions, (in *z-plane*).

$$\boldsymbol{CS_2(s) = \frac{1}{S^2 \times (S+1)}} \tag{5}$$

$$\boldsymbol{CS_2(z) = Z^{-2} \times CS_1(z)} \tag{6}$$

$$\boldsymbol{CS_2(z) = Z^{-2} \times \frac{0.09516}{Z-0.9048}\ , T = 0.1} \tag{7}$$

Proportional plus Derivative (PD) control: This type of linear controller is widely used in control process where the results are sensitive to exceeded of set point. This controller, like Proportional controller, has permanent variation in presence of self-limitation control. In mathematically, the formulation of Proportional-Derivative part calculated as follows;

$$\boldsymbol{U_{PD} = K_p \times e + K_v(\frac{de}{dt}) = K_p \times e + K_v \dot{e}} \tag{8}$$

The Derivative component in this type of methodology is used to cancel outs the change process variables change in presence of quick change in controllers input. Figure 8 shows the block diagram of Proportional-Derivative (PD) controller.

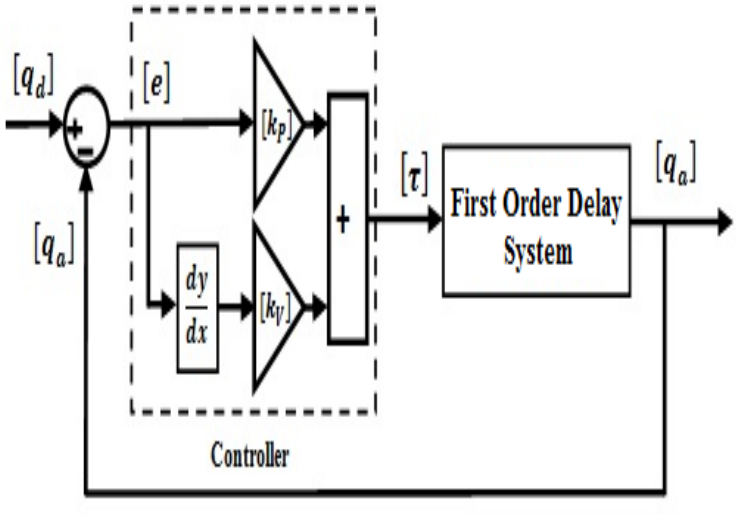


Figure 8. Block Diagram of PD Controller

Figure 9 shows the ramp response of PD controller.

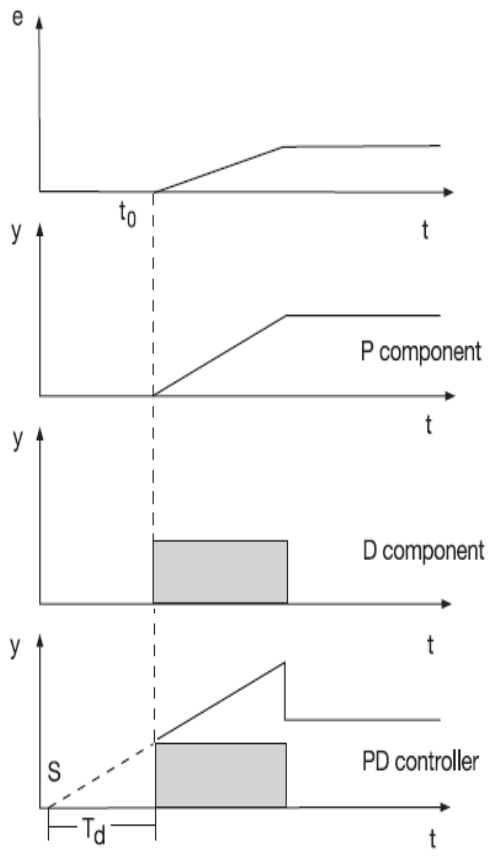


Figure 9. Ramp Response of a PD Controller [13]

Proportional plus Integral (PI) control: According to integral type of controller, it takes relatively long time. The proportional type controller used to immediately response to the input variations. The proportional-integral (PI) controller has the advantages of both proportional and integral controller; it is rapid response to the input deviation as well as the

exact control at the desired input. Figure 10 shows the block diagram of PI control of FOD system.

$$\boldsymbol{U_{PI} = K_p \times e + K_i(\frac{1}{T}\int e.dt) = K_p \times e + K_i \sum e} \quad (9)$$

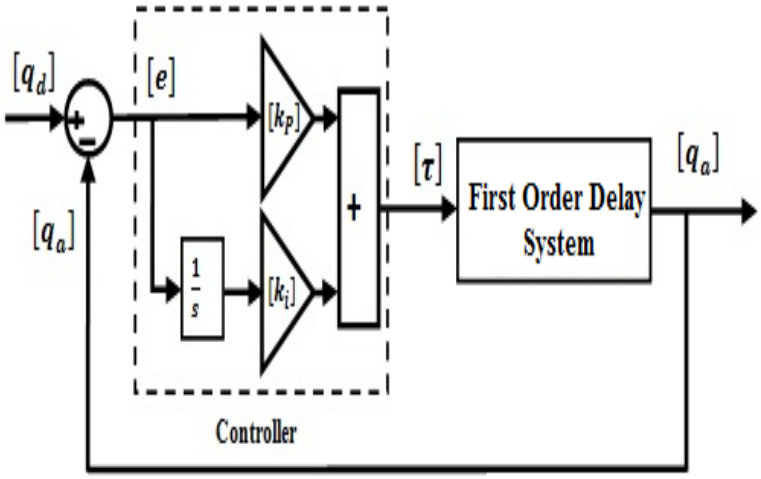


Figure 10. Block Diagram of PI Controller for First Order Delay System

Figure 11 shows the step response of PI controller.

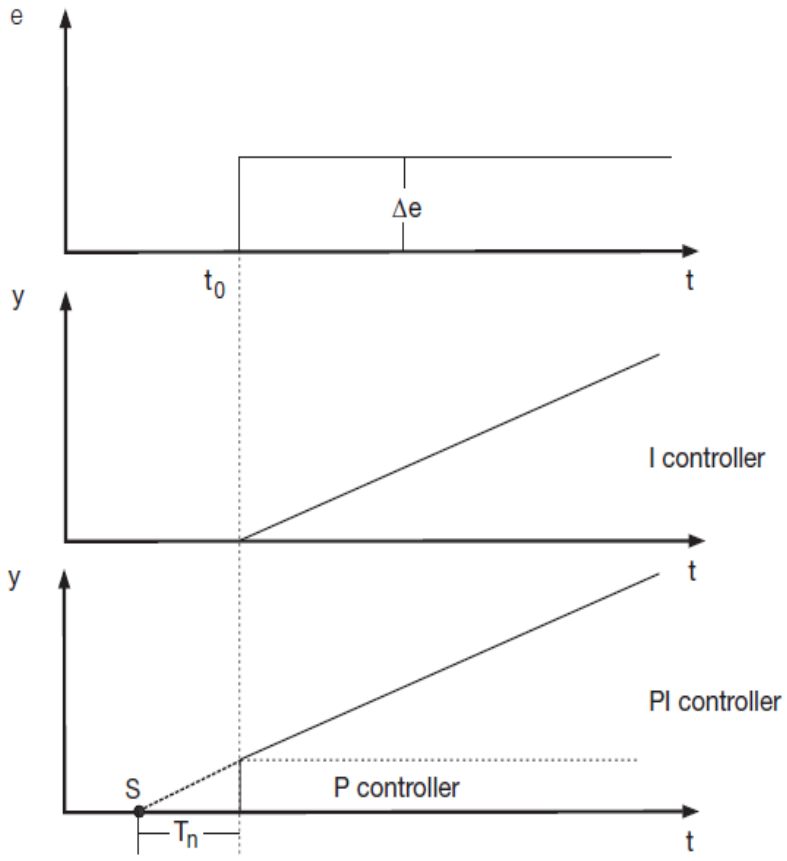


Figure 11. Step Response of A Proportional-Integral (PI) Controller [13]

3. Methodology

Design PID Control Algorithm: The combination of proportional (P) component, integral (I) component with a derivative (D) controller offered advantages in each case. This type of controller has rapid response to the input deviation, the exact control at the desired input as well as fast response to the disturbances. The PID controller takes the error between the desired variables and the actual variables to control the FOD systems. A proportional-derivative integral control system can easily be implemented. This method does not provide sufficient control for systems with time-varying parameters or highly nonlinear systems. Figure 12 shows the block diagram of PID control of FOD system. The formulation of PID controller calculated as follows;

$$\boldsymbol{U_{PID} = K_p \times e + K_i(\frac{1}{T}\int e.dt) + K_v(\frac{de}{dt}) = K_p \times e + K_i \sum e + K_v \dot{e}} \quad (10)$$

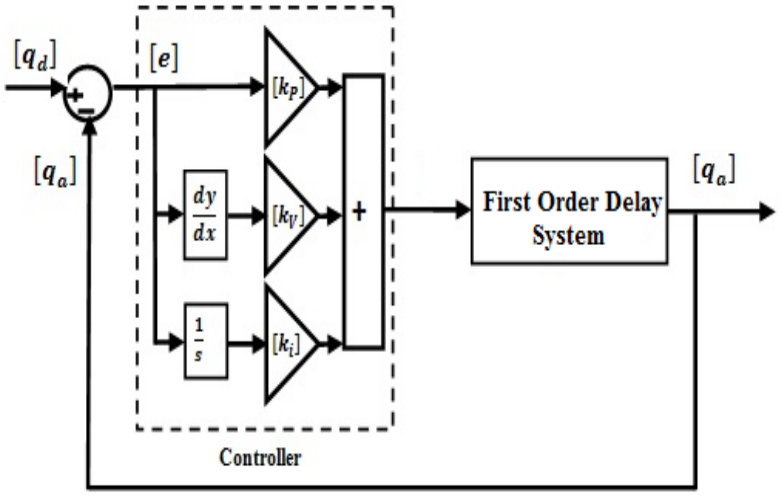


Figure 12. Block Diagram of PID Control of FOD System

Figure 13 shows the step response of PID controller.

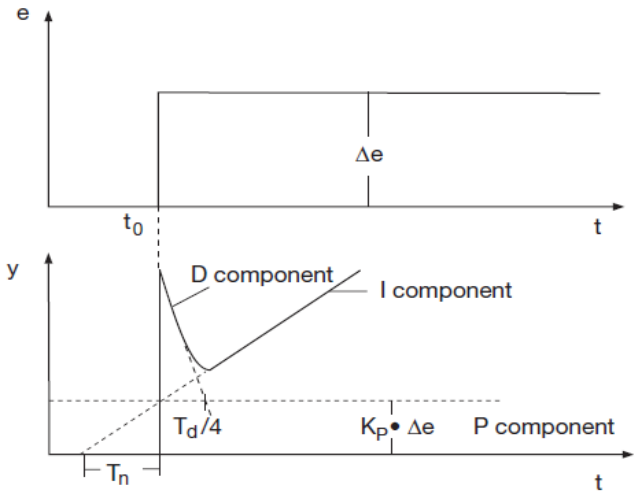


Figure 13. Step Response of a Proportional-Integral-Derivative (Pid) Controller

Design FPGA-Based Derivative Algorithm: The following formulation shows the derivative algorithm:

$$d(e) = \frac{Din(t) - Din(t-1)}{\Delta t} = \big(Din(k+1) - Din(k)\big) \times sample\ time \quad (11)$$

$$Din = q_d - q_a \quad (12)$$

However $\boldsymbol{q_d}$ **and** $\boldsymbol{q_a}$ are 30 bits but Din is 40 bits. In derivative algorithm, delay time is the main challenge. In this research the value of sample time is"01010".

To design $\boldsymbol{Din(k+1)}$, design a register has the main role. The vast majority of modern commercial systems are built with registers using positive edge-triggered D flip-flops. A group of cascaded flip flops used to store related bits of information is known as a register. Figure 14 shows D flip-flop.

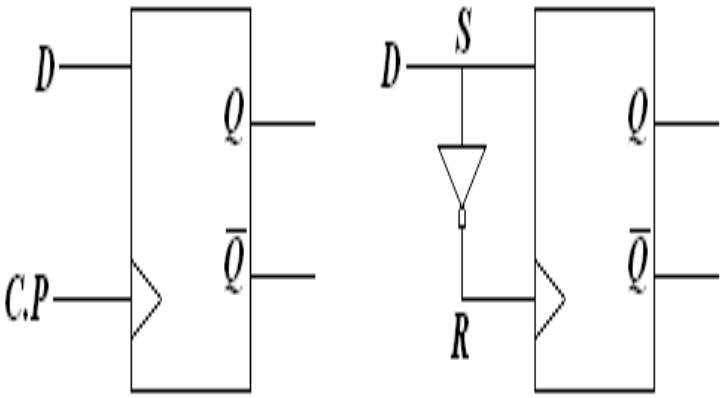


Figure 14. D Flip Flop

Figure 15 shows the RTL schematic of FPGA-based register.

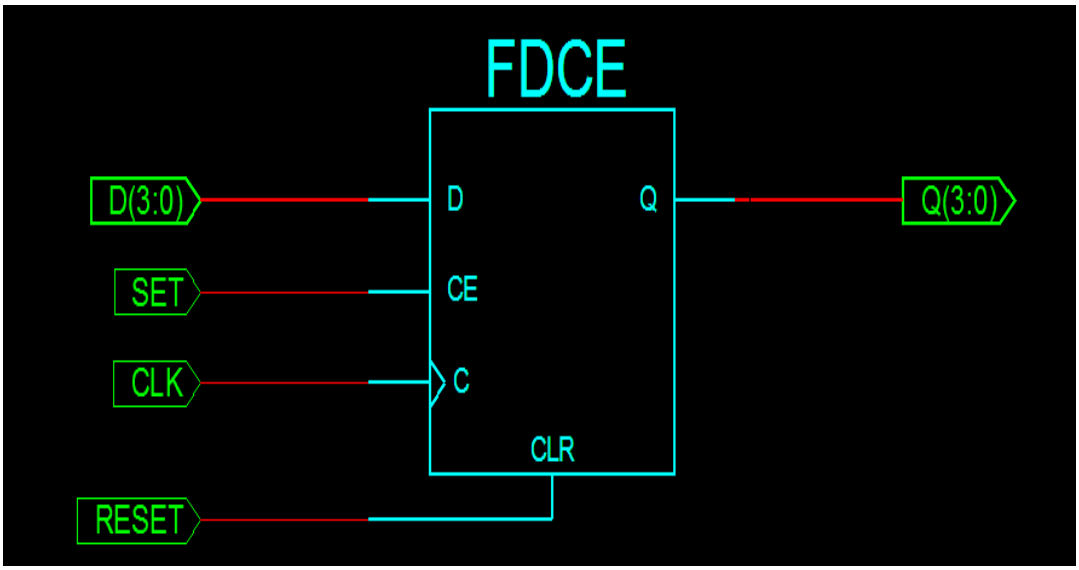


Figure 15. Outline Register Algorithm in HDL Using Spartan 3E

Figure 16 shows the outline of FPGA-based derivative algorithm with 40 bits input and 40 bits output. Regarding to this algorithm, the derivation of input signal is calculated in output.

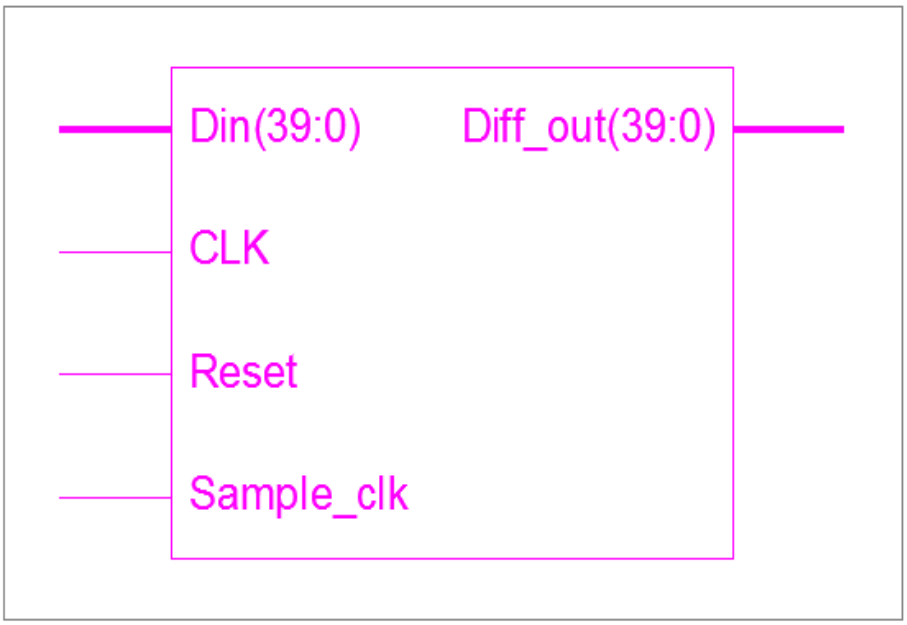


Figure 16. Outline FPGA-Based Derivative Algorithm

Figure 17 shows the interior view FPGA-based derivative algorithm using VHDL program. Regarding to this algorithm, CLK is used to synchronize three types register which used in this algorithm and SAMPLE-CLK is used to system synchronization.

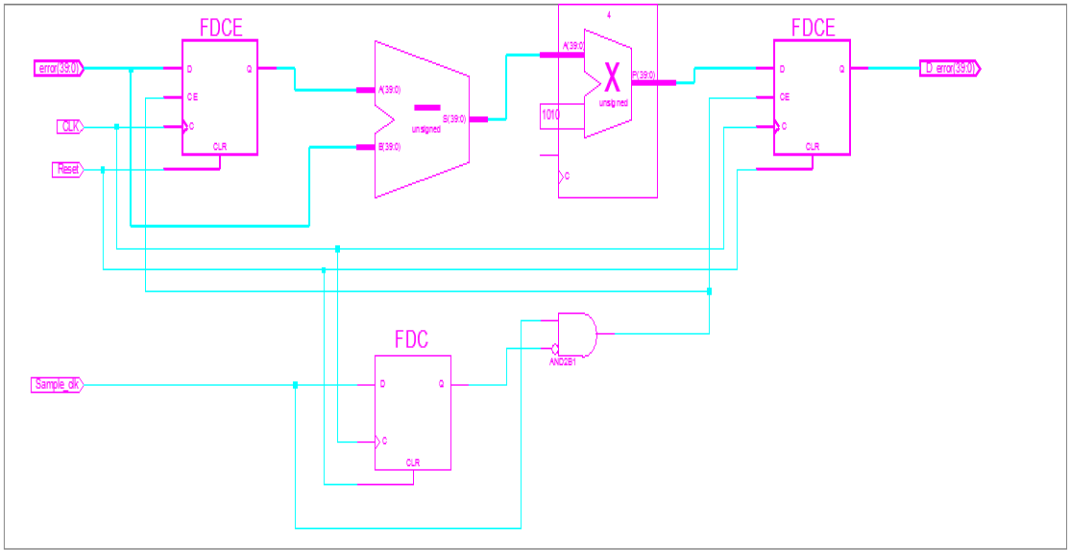


Figure 17. Interior View: FPGA-Based Derivative Algorithm

The VHDL code which applied to automotive Spartan 3E-XA3S1600E shows in Figure 18.

```
entity Derivative_control is
   Port(error : in  STD_LOGIC_VECTOR (39 downto 0);
        D_error : out  STD_LOGIC_VECTOR (39 downto 0);
        Reset : in  STD_LOGIC;
        CLK : in  STD_LOGIC;
        Sample_clk : in  STD_LOGIC);
end Derivative_control;

architecture Behavioral of Derivative_control is
----------------------------------------------------------------------------------------------------
constant sample_rate : std_logic_vector(4 downto 0) := "01010";
signal last_error : std_logic_vector(39 downto 0);
signal data_sample_error : std_logic_vector(39 downto 0);
signal diff_data : std_logic_vector(44 downto 0);
signal last_sample_clk : std_logic;
signal sample_clk_edge : std_logic;
----------------------------------------------------------------------------------------------------
begin
----------------------------------------------------------------------------------------------------
sample_clk_edge <= (not last_sample_clk) and Sample_clk;
process(CLK, Reset)
begin
   if(Reset = '1')then
      last_sample_clk <= '0';
   elsif(rising_edge(CLK))then
      last_sample_clk <= Sample_clk;
   end if;
end process;
----------------------------------------------------------------------------------------------------
process(CLK)
begin
   if(Reset = '1')then
      last_error <= (others => '0');
      D_error <= (others => '0');
   elsif(rising_edge(CLK) and sample_clk_edge = '1')then
```

Figure 18. VHDL Code: FPGA-Based Derivative Algorithm

Design FPGA-Based Integral Algorithm: The following formulation shows the derivative algorithm:

$$\boldsymbol{I(out)_{(K)} = I(out)_{(k-1)} + (Ki \times e(k) \times sample\ time)} \quad (13)$$

Figure 19 shows the outline of FPGA-based integral algorithm with 40 bits input and 40 bits output.

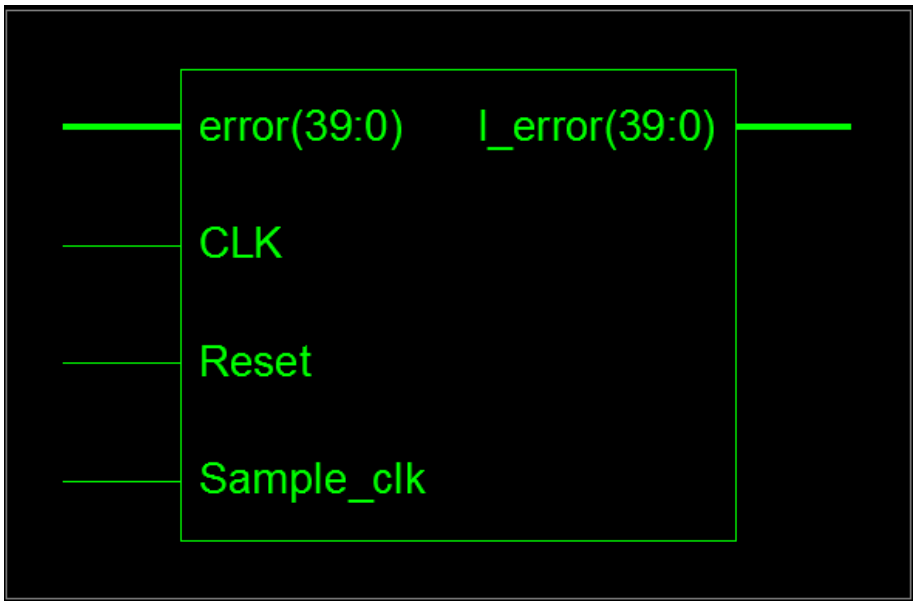


Figure 19. Outline FPGA-based Integral Algorithm

Figure 20 shows the interior view FPGA-based derivative algorithm using VHDL program. Regarding to this algorithm, CLK is used to synchronize three types register which used in this algorithm and SAMPLE-CLK is used to system synchronization.

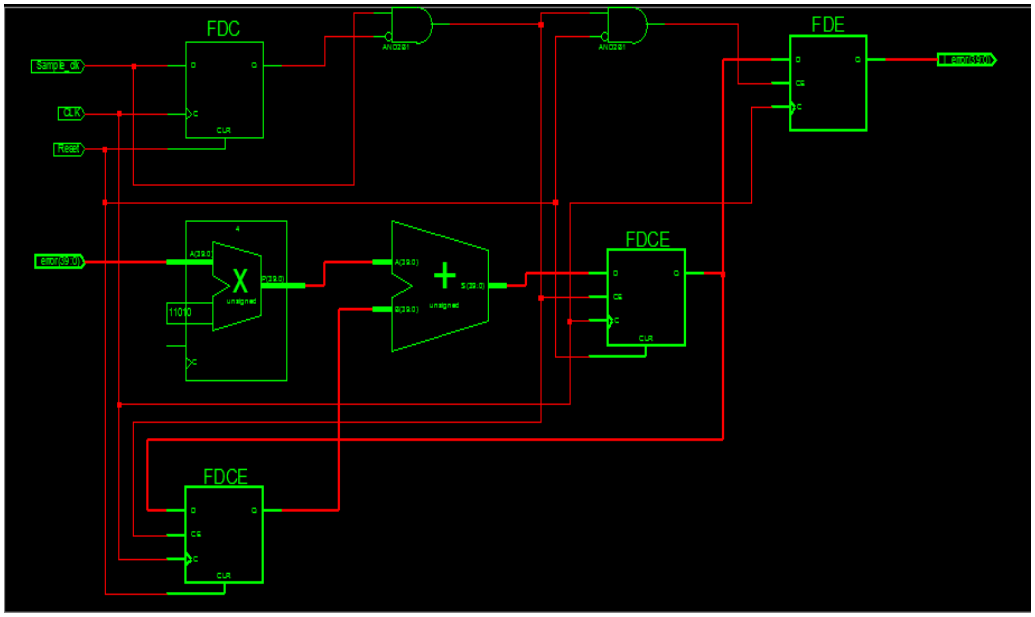


Figure 20. Interior View: FPGA-Based Integral Algorithm

The integral VHDL code which applied to automotive Spartan 3E-XA3S1600E shows in Figure 21.

```
entity I_control is
   Generic (Ki : std_logic_vector(4 downto 0) := B"11010");
   Port(error : in  STD_LOGIC_VECTOR (39 downto 0);
        I_error : out  STD_LOGIC_VECTOR (39 downto 0);
        Reset : in  STD_LOGIC;
        CLK : in  STD_LOGIC;
        Sample_clk : in  STD_LOGIC);
end I_control;

architecture Behavioral of I_control is
----------------------------------------------------------------------
--constant sample_rate : std_logic_vector(4 downto 0) := "01010";
signal last_I : std_logic_vector(39 downto 0);
signal data_sample_sub : std_logic_vector(39 downto 0);
signal I_data : std_logic_vector(44 downto 0);
signal I_error_sig : std_logic_vector(39 downto 0);
signal last_sample_clk : std_logic;
signal sample_clk_edge : std_logic;
----------------------------------------------------------------------
begin
----------------------------------------------------------------------
sample_clk_edge <= (not last_sample_clk) and Sample_clk;
process(CLK, Reset)
begin
   if(Reset = '1')then
      last_sample_clk <= '0';
   elsif(rising_edge(CLK))then
      last_sample_clk <= Sample_clk;
   end if;
```

Figure 21. VHDL Code: FPGA-Based Integral Algorithm

The Z formulation of PID controller is design as follows:

$$\boldsymbol{U[k] = U[k-1] + K_1 \times e[k] + K_2 \times e[k-1] + K_3 \times e[k-2]} \quad \textbf{(14)}$$

In this algorithm, we have five inputs (actual input, desired input, CLK, reset and SAMPLE-CLK) and an output (PID-Control). Actual and desired inputs are 30 bits and PID control output is 35 bits. In this design, CLK is used to activate the PID sub-parts (registers), reseat is used to re-start of registers and SAMPLE-CLK is used to synchronization proportional, derivate and integral parts. Figure 22 shows the outline of FPGA-based PID control algorithm.

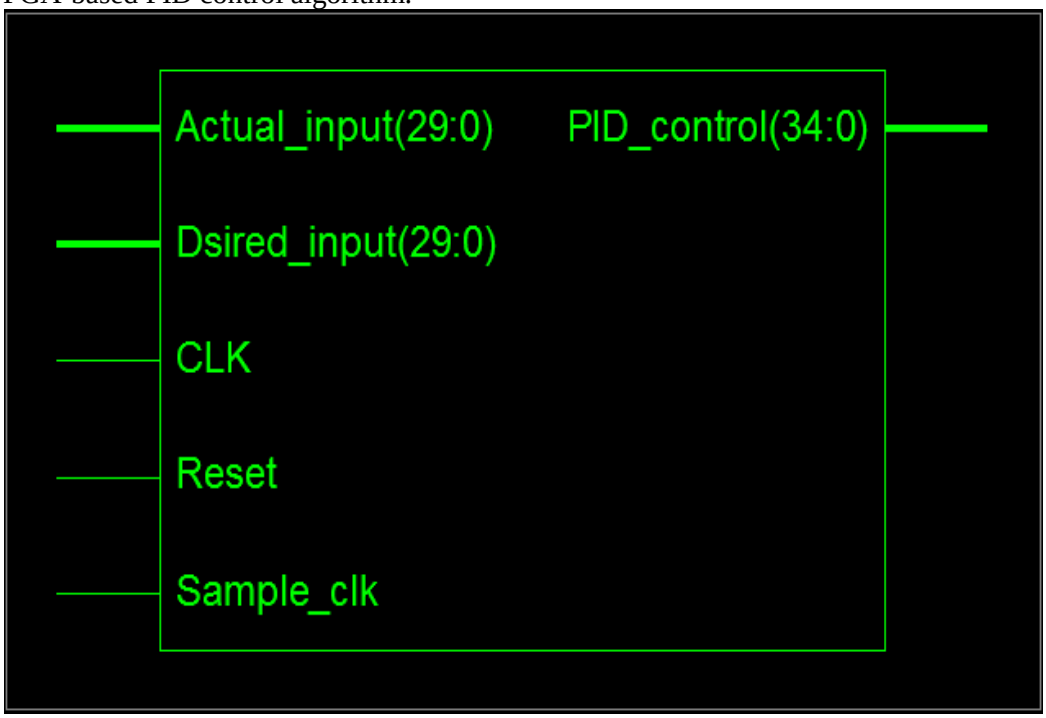


Figure 22. Outline: FPGA-Based PID Algorithm

The interior view of PD algorithm shows in Figure 23.

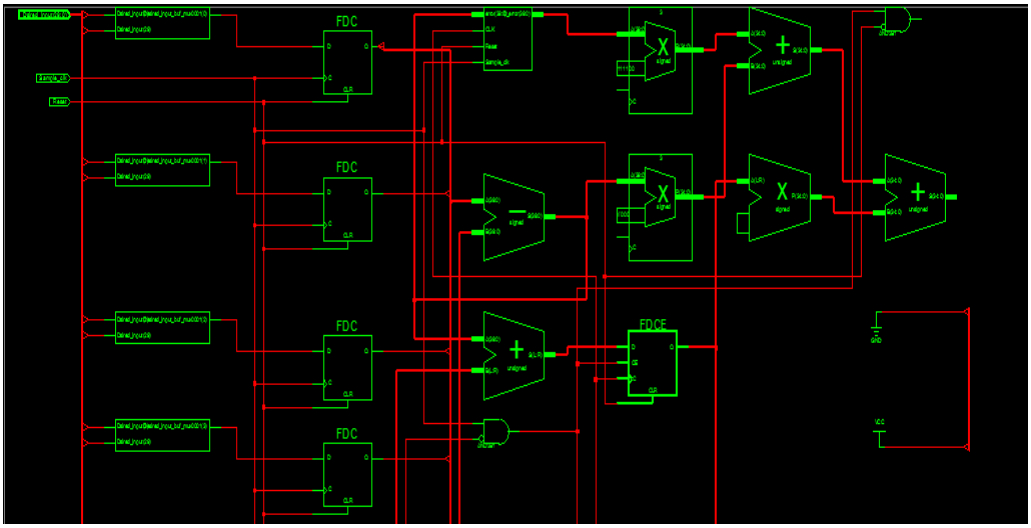


Figure 23. Interior View: FPGA-based PID Algorithm

Regarding to Figure 23, the PID algorithm have some buffers for desired and actual inputs. Figure 24 shows the interiors of buffer.

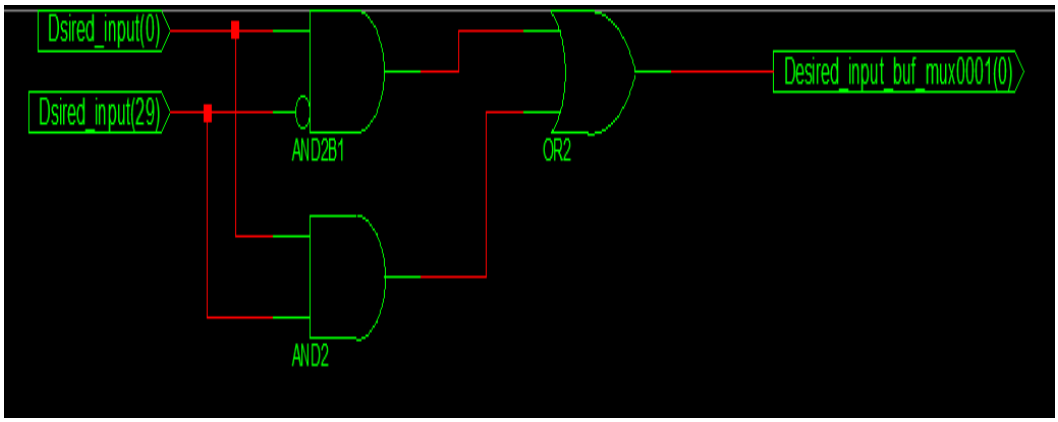


Figure 24. Interior View: FPGA-Based Buffer

The VHDL program of FPGA-based PID controller shows in Figure 25.

```
entity PID_Controler is
      Generic(Kp : std_logic_vector(7 downto 0) := B"11111010";
               Kv : std_logic_vector(7 downto 0) := B"00111100";
               Ki : std_logic_vector(7 downto 0) := B"00111100");
      Port(Actual_input : in  STD_LOGIC_VECTOR (29 downto 0);
        Dsired_input : in  STD_LOGIC_VECTOR (29 downto 0);
        PID_control : out  STD_LOGIC_VECTOR (34 downto 0);
        Reset : in  STD_LOGIC;
        CLK : in  STD_LOGIC;
        Sample_clk : in  STD_LOGIC);
end PID_Controler;
--------------------------------------------------------------------------------
architecture Behavioral of PID_Controler is
--------------------------------------------------------------------------------
------Signals definitions
--------------------------------------------------------------------------------
signal Actual_input_buf : std_logic_vector(39 downto 0);
signal Desired_input_buf : std_logic_vector(39 downto 0);
signal Error_gain : std_logic_vector(47 downto 0);
signal Error_diff : std_logic_vector(39 downto 0);
signal Error_integral : std_logic_vector(39 downto 0);
signal Error : std_logic_vector(39 downto 0);
signal Error_diff_gain : std_logic_vector(47 downto 0);
signal Error_integral_gain : std_logic_vector(47 downto 0);
signal PID_control_buf : std_logic_vector(39 downto 0);
```

Figure 25. VHDL Code: FPGA-based PID Algorithm

The device utilization summary shows in Figure 26.

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	219	14752	1%
Number of Slice Flip Flops	153	29504	0%
Number of 4 input LUTs	264	29504	0%
Number of bonded IOBs	98	304	32%
Number of MULT18X18SIOs	12	36	33%
Number of GCLKs	2	24	8%

Figure 26. Device Utilization Summaries

4. Result and Discussion

In this part, MATLAB based PID controller and FPGA-based PID controller are test. Figure 27 shows the trajectory following in PID controller and control free in first order delay system. In rise-time point of view, in certain situation, PID controller's rise-time is 0.2 second and control free's rise-time is about 3.939 second.

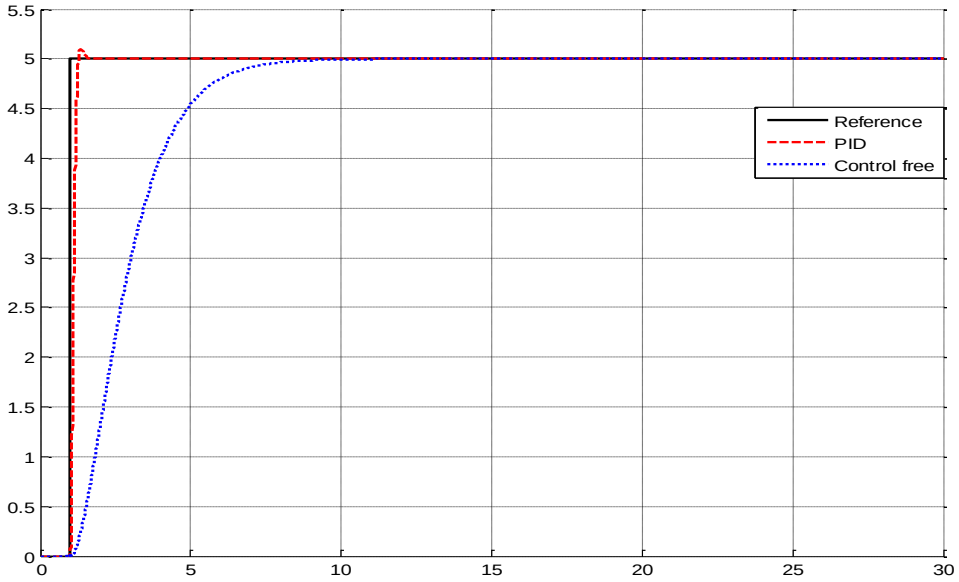


Figure 27. Trajectory Following: PID Controller and Control Free

Figure 28 shows the power of disturbance rejection in PID controller and control free for first order delay system. In presence of uncertainty, the robustness of PID controller is better than control free technique. Based on the following graph, the rate of overshoot has increased from 1% in certain condition to 45% and the rate of rise-time has increased from 0.2 second to 0.58 second in presence of uncertainty. The rise-time in control free FOD system in present of uncertainty has increased from 3.939 second to 19.46 second. Regarding to Figure 28, however PID algorithm is a good technique to reduce the delay time in certain and uncertain condition but it has variation in uncertain condition.

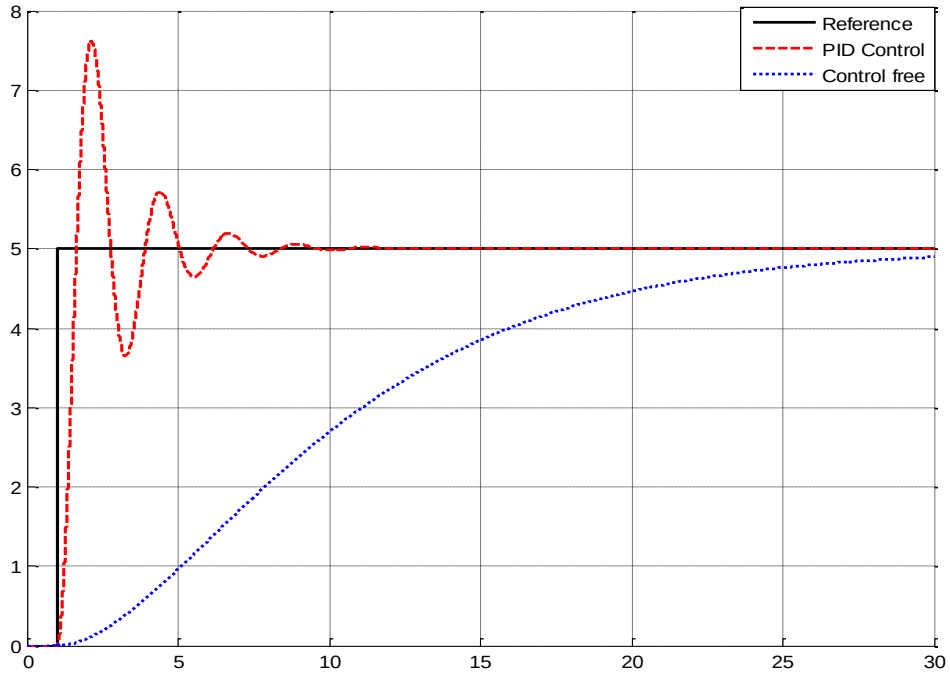


Figure 28. Disturbance Rejection: PID Controller and Control Free

FPGA Device Timing Detail: In this part synthesis report and timing, detail for PID controller is introduced.

Derivative Algorithm:

The first part in this analysis is derivative algorithm synthesize report. Figure 29 shows the HDL synthesize report in derivative algorithm.

```
HDL Synthesis Report

Macro Statistics
# Multipliers                          : 1
 40x5-bit multiplier                   : 1
# Adders/Subtractors                   : 1
 40-bit subtractor                     : 1
# Registers                            : 3
 1-bit register                        : 1
 40-bit register                       : 2
```

Figure 29. Hdl Synthesis Report: Derivative Algorithm

Figure 30 shows advanced HDL synthesis report in derivative algorithm.

```
Advanced HDL Synthesis Report

Macro Statistics
# Multipliers                                      : 1
 40x5-bit multiplier                               : 1
# Adders/Subtractors                               : 1
 40-bit subtractor                                 : 1
# Registers                                        : 81
 Flip-Flops                                        : 81
```

Figure 30. Advanced Hdl Synthesis Report: Derivative Algorithm

Figure 31 shows the device utilization summary in derivative algorithm.

```
Device utilization summary:
---------------------------

Selected Device : xa3s1600efgg400-4

 Number of Slices:                       67  out of  14752     0%
 Number of Slice Flip Flops:             81  out of  29504     0%
 Number of 4 input LUTs:                 67  out of  29504     0%
 Number of IOs:                          83
 Number of bonded IOBs:                  83  out of    304    27%
 Number of MULT18X18SIOs:                 3  out of     36     8%
 Number of GCLKs:                         1  out of     24     4%
```

Figure 31. Device Utilization Summary: Derivative Algorithm

Figure 32 shows the timing summary in derivative algorithm.

```
Timing Summary:
---------------
Speed Grade: -4

   Minimum period: 15.716ns (Maximum Frequency: 63.629MHz)
   Minimum input arrival time before clock: 16.466ns
   Maximum output required time after clock: 4.283ns
   Maximum combinational path delay: No path found
```

Figure 32. Timing Summary: Derivative Algorithm

Regarding to Figure 32, the Maximum frequency in this design is **60.73 MHz**. The timing detail regarding to derivative algorithm have been shown the following Figures. Figures 33, 34 and 35 have been shown the timing details in different data path. Figure 33 and 34 show the timing detail in data path between errors to differential data.

```
Delay:               15.716ns (Levels of Logic = 46)
  Source:            last_error_0 (FF)
  Destination:       diff_data_39 (FF)
  Source Clock:      CLK rising
  Destination Clock: CLK rising

  Data Path: last_error_0 to diff_data_39
--------------------------------------
Total                      15.716ns (13.799ns logic, 1.917ns route)
                                    (87.8% logic, 12.2% route)
```

Figure 33. Timing Detail: Derivative Algorithm (Data Path: Error (K+1) To Differential of Error)

Regarding the Figure 33, the delay time is 15.716 ns in 46 logic levels that 87.8% is logic and 12.2% is route delay.

```
---------------------------------------------------------------------------
Timing constraint: Default OFFSET IN BEFORE for Clock 'CLK'
  Total number of paths / destination ports: 435753 / 161
---------------------------------------------------------------------------
Offset:              16.466ns (Levels of Logic = 47)
  Source:            error<0> (PAD)
  Destination:       diff_data_39 (FF)
  Destination Clock: CLK rising

  Data Path: error<0> to diff_data_39
--------------------------------------
Total                    16.466ns (14.426ns logic, 2.040ns route)
                                  (87.6% logic, 12.4% route)
```

Figure 34. Timing Detail: Derivative Algorithm (Data Path: Error (K) To Differential of Error)

Regarding the Figure 34, the delay time is 16.466 ns in 47 logic levels that 87.6% is logic and 12.4% is route delay. The total number of paths is 435753 and total number of destination reports is 161.


```
Timing constraint: Default OFFSET OUT AFTER for Clock 'CLK'
  Total number of paths / destination ports: 40 / 40
-------------------------------------------------------------------------
Offset:              4.283ns (Levels of Logic = 1)
  Source:            diff_data_39 (FF)
  Destination:       D_error<39> (PAD)
  Source Clock:      CLK rising

  Data Path: diff_data_39 to D_error<39>
                               Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     FDCE:C->Q             1   0.591   0.420  diff_data_39 (diff_data_39)
     OBUF:I->O                 3.272          D_error_39_OBUF (D_error<39>)
    ----------------------------------------
    Total                      4.283ns (3.863ns logic, 0.420ns route)
                                       (90.2% logic, 9.8% route)
```

Figure 35. Timing Detail: Derivative Algorithm (Data Path: Differential of Error to D (Error) Buffers)

Integral Algorithm:

Figure 36 shows the HDL synthesize report in integral algorithm.

```
HDL Synthesis Report

Macro Statistics
# Multipliers                                          : 1
 40x5-bit multiplier                                   : 1
# Adders/Subtractors                                   : 1
 40-bit adder                                          : 1
# Registers                                            : 4
 1-bit register                                        : 1
 40-bit register                                       : 3
```

Figure 36. Hdl Synthesis Report: Integral Algorithm

Figure 37 shows advanced HDL synthesis report in integral algorithm.

```
Advanced HDL Synthesis Report

Macro Statistics
# Multipliers                                      : 1
 40x5-bit multiplier                               : 1
# Adders/Subtractors                               : 1
 40-bit adder                                      : 1
# Registers                                        : 121
 Flip-Flops                                        : 121
```

Figure 37. Advanced Hdl Synthesis Report: Integral Algorithm

Figure 38 shows the device utilization summary in integral algorithm.

```
Device utilization summary:
---------------------------

Selected Device : xa3s1600efgg400-4

 Number of Slices:                       81  out of  14752     0%
 Number of Slice Flip Flops:            121  out of  29504     0%
 Number of 4 input LUTs:                 69  out of  29504     0%
 Number of IOs:                          83
 Number of bonded IOBs:                  83  out of    304    27%
 Number of MULT18X18SIOs:                 3  out of     36     8%
 Number of GCLKs:                         1  out of     24     4%
```

Figure 38. Device Utilization Summary: Integral Algorithm

Figure 39 shows the timing summary in integral algorithm.

```
Timing Summary:
---------------
Speed Grade: -4

   Minimum period: 5.612ns (Maximum Frequency: 178.190MHz)
   Minimum input arrival time before clock: 15.599ns
   Maximum output required time after clock: 4.283ns
   Maximum combinational path delay: No path found
```

Figure 38. Timing Summary: Integral Algorithm

Regarding to Figure 39, the Maximum frequency in this design is 64.1 MHz. The timing detail for integral algorithm has been shown the following Figures. Figures 40, 41 and 42 have been shown the timing details in different data path. Figure 40 and 41 show the timing detail in data path between $\boldsymbol{e[k-2]}$ to $\boldsymbol{U[k]}$.

```
Timing constraint: Default period analysis for Clock 'CLK'
  Clock period: 5.612ns (frequency: 178.190MHz)
  Total number of paths / destination ports: 1020 / 240
-------------------------------------------------------------------
Delay:               5.612ns (Levels of Logic = 41)
  Source:            last_I_0 (FF)
  Destination:       I_error_sig_39 (FF)
  Source Clock:      CLK rising
  Destination Clock: CLK rising

  Data Path: last_I_0 to I_error_sig_39
---------------------------------------
Total                      5.612ns (5.113ns logic, 0.499ns route)
                                   (91.1% logic, 8.9% route)
```

Figure 39. Timing Detail: Integral Algorithm (Data Path: $\boldsymbol{e[k-2]}$ To $\boldsymbol{U[k]}$)

Regarding the Figure 39, the delay time is 5.612 ns in 41 logic levels that 91.1% is logic and 8.9% is route delay.

```
-------------------------------------------------------------------
Timing constraint: Default OFFSET IN BEFORE for Clock 'CLK'
  Total number of paths / destination ports: 206669 / 161
-------------------------------------------------------------------
Offset:              15.599ns (Levels of Logic = 30)
  Source:            error<16> (PAD)
  Destination:       I_error_sig_39 (FF)
  Destination Clock: CLK rising

  Data Path: error<16> to I_error_sig_39
---------------------------------------
Total                     15.599ns (13.436ns logic, 2.163ns route)
                                   (86.1% logic, 13.9% route)
```

Figure 40. Timing Detail: Integral Algorithm (Data Path: $\boldsymbol{e[k]}$ to $\boldsymbol{U[k]}$)

Regarding the Figure 40, the delay time is 15.599 ns in 30 logic levels that 86.1% is logic and 13.9% is route delay. The total number of paths is 206669 and total number of destination reports is 161.

```
Timing constraint: Default OFFSET OUT AFTER for Clock 'CLK'
  Total number of paths / destination ports: 40 / 40
--------------------------------------------------------------------------
Offset:              4.283ns (Levels of Logic = 1)
  Source:            I_error_39 (FF)
  Destination:       I_error<39> (PAD)
  Source Clock:      CLK rising

  Data Path: I_error_39 to I_error<39>
                               Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     FDE:C->Q             1   0.591   0.420  I_error_39 (I_error_39)
     OBUF:I->O                3.272          I_error_39_OBUF (I_error<39>)
    ----------------------------------------
    Total                     4.283ns (3.863ns logic, 0.420ns route)
                                      (90.2% logic, 9.8% route)
```

Figure 41. Timing Detail: Integral Algorithm

PID Algorithm:

Figure 42 shows the HDL synthesize report for PID algorithm.

```
---------------------------------------------------------------------
HDL Synthesis Report

Macro Statistics
# Multipliers                                          : 4
 40x5-bit multiplier                                   : 1
 40x8-bit multiplier                                   : 3
# Adders/Subtractors                                   : 5
 40-bit adder                                          : 3
 40-bit subtractor                                     : 2
# Registers                                            : 9
 1-bit register                                        : 2
 40-bit register                                       : 7
```

Figure 42. Hdl Synthesis Report: Pid Algorithm

Figure 43 shows advanced HDL synthesis report in PID algorithm.

```
Advanced HDL Synthesis Report

Macro Statistics
# Multipliers                                      : 4
 40x5-bit multiplier                               : 1
 40x8-bit multiplier                               : 2
 40x8-bit registered multiplier                    : 1
# Adders/Subtractors                               : 5
 35-bit adder                                      : 2
 40-bit adder                                      : 1
 40-bit subtractor                                 : 2
# Registers                                        : 224
 Flip-Flops                                        : 224
```

Figure 43. Advanced Hdl Synthesis Report: Pid Algorithm

Figure 44 shows the device utilization summary in PID algorithm.

```
Device utilization summary:
---------------------------

Selected Device : xa3s1600efgg400-4

 Number of Slices:                      219  out of  14752     1%
 Number of Slice Flip Flops:            153  out of  29504     0%
 Number of 4 input LUTs:                264  out of  29504     0%
 Number of IOs:                          98
 Number of bonded IOBs:                  98  out of    304    32%
    IOB Flip Flops:                      60
 Number of MULT18X18SIOs:                12  out of     36    33%
 Number of GCLKs:                         2  out of     24     8%
```

Figure 44. Device Utilization Summary: PID Algorithm

Figure 45 shows the timing summary in PD algorithm.

```
Timing Summary:
---------------
Speed Grade: -4

   Minimum period: 15.716ns (Maximum Frequency: 63.629MHz)
   Minimum input arrival time before clock: 4.683ns
   Maximum output required time after clock: 22.296ns
   Maximum combinational path delay: No path found
```

Figure 45. Timing Summary: Derivative Algorithm

Regarding to Figure 45, the Maximum frequency in this design is 44.85 MHz. however the rate of clock is 63.629 MHz but in PID algorithm the maximum frequency is 44.85MHz.

5. Conclusion

In this research, FPGA-based PID controller for first order delay system is design and analysis. First order delay system is a second order nonlinear, time variant and delay time system. In this type of system, reduce or eliminate the delay time is the main objective. To improve the time delay in this system, PID control algorithm is introduced. The first order delay system has been challenged in uncertainty and the time response has been changed from 3.939 second to 19.46 second. When PID controller is applied to first order delay system, caused to change the time response. The time response in PID algorithm has been changed from 0.2 second in certain condition to 0.58 second in uncertain condition (see Figures 27 and 28). The FPGA-based PID controller design based on derivative and integration algorithms. The maximum frequency of input clock pulse in this design is 63.629 MHz but the maximum output PID algorithm is about 44.85 MHz. In comparison with PD controller, PID controller is more stable but the maximum output frequency in PD (51.89 MHz) is better than PID (44.85 MHz).

References

[1] R. Faraji, A. Rouholamini, H. R. Naji, R. Fadaeinedjad and M. R. Chavoshian, "FPGA-Based Real Time Incremental Conductance Maximum Power Point Tracking Controller For Photovoltaic Systems", Power Electronics, IET, vol. vol. 7, no. 5, **(2014)**, pp. 1294-1304.

[2] S. Ghosh, R. K. Barai, S. Bhattarcharya, P. Bhattacharyya, S. Rudra, A. Dutta and R. Pyne, "An FPGA Based Implementation of a Flexible Digital PID Controller For a Motion Control System", In Computer Communication and Informatics *(ICCCI*), 2013 International Conference on, IEEE, January **(2013)**, pp. 1-6.

[3] K. Lochan and B. K. Roy, "Control of Two-link 2-DOF Robot Manipulator Using Fuzzy Logic Techniques: A Review:", In Proceedings of Fourth International Conference on Soft Computing for Problem Solving , Springer India, **(2015)**, pp. 499-511.

[4] J Li, L. Liu, Y. Wang and W Liang, "Adaptive Hybrid Impedance Control of Robot Manipulators with Robustness against Environment's Uncertainties", In Mechatronics and Automation (ICMA), 2015 IEEE International Conference on, IEEE, August **(2015)**, pp. 1846-1851.

[5] Y. Q. Bao and Y. Li, "FPGA-Based Design of Grid Friendly Appliance Controller", Smart Grid, IEEE Transactions on, vol. 5, no. 2, **(2014)**, pp. 924-931.

[6] B. Siciliano and O. Khatib, "Springer handbook of robotics", Springer Science and Business Media, (Eds), **(2008)**.

[7] R. A. DeCarlo, S. H. Zak and G. P. Matthews, "Variable Structure Control of Nonlinear Multivariable Systems: A Tutorial", Proceedings of the IEEE, vol. 76, no. 3, **(2002)**, pp. 212-232.

[8] K. D. Young, V. Utkin and U. Ozguner, "A Control Engineer's Guide To Sliding Mode Control", IEEE International Workshop on Variable Structure Systems, **(2002)**, pp. 1-14.

[9] O. Kaynak, "Guest Editorial Special Section on Computationally Intelligent Methodologies and Sliding-Mode Control", IEEE Transactions on Industrial Electronics, vol. 48, no. 1, **(2001)**, pp. 2-3.

[10] I. Boiko, L. Fridman, A. Pisano and E. Usai, "Analysis Of Chattering In Systems With Second-Order Sliding Modes", IEEE Transactions on Automatic Control, vol. 52, no. 11, **(2007)**, pp. 2085-2102.

[11] V. Utkin, "Variable Structure Systems With Sliding Modes", IEEE Transactions on Automatic Control,vol. 22, no. 2, **(2002)**, pp. 212-222.

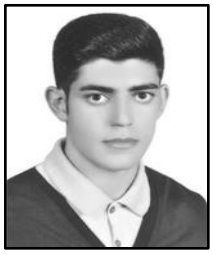
[12] P. I. Corke and B. Armstrong-Helouvry, "A Search For Consensus Among Model Parameters Reported For The PUMA 560 Robot", IEEE International Conference on Robotica and Automation, **(1994)**, pp. 1608-1613.

[13] F. Piltan, N. Sulaiman, M. H. Marhaban, A. Nowzary and M. Tohidian, "Design of FPGA-Based Sliding Mode Controller for Robot Manipulator", International Journal of Robotic and Automation, vol. 2 (3), **(2011)**, pp. 183-204.

[14] F. Piltan, N. Sulaiman, A. Jalali and K. Aslansefat, "Evolutionary Design of Mathematical Tunable FPGA Based MIMO Fuzzy Estimator Sliding Mode Based Lyapunov Algorithm: Applied to Robot Manipulator", International Journal of Robotics and Automation, vol. 2, no. 5, **(2011)**, pp. 317-343.

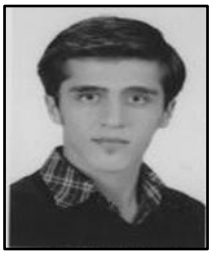
[15] F. Piltan, I. Nazari, S. Siamak and P. Ferdosali, "Methodology of FPGA-Based Mathematical error-Based Tuning Sliding Mode Controller", International Journal of Control and Automation, vol. 5, no. 1, **(2012)**, pp. 89-118.

Authors



Saman Namvarchi is currently Research Student at Institute of Advanced Science and Technology, Research and Training Center, IRAN SSP. He is research student of team (8 researchers) to design a Micro-electronic Based nonlinear controller for first order delay system since March, 2015. His current research interests are nonlinear control, artificial control system, Microelectronic Device, and HDL design.

Samira Soltani is currently research assistant at Institute of Advance Science and Technology, Research and Development Center, IRAN SSP. She is research assistant of team (47 researchers) to design and build of nonlinear control of industrial robot manipulator for experimental research and education and published about 5 Papers in this field since 2010 to 2012, research assistant (9 researchers) to design and implement intelligent tuning the rate of fuel ratio in internal combustion engine for experimental research and education and published 1 Journal papers since 2011 to 2013, research assistant (34 researchers) of filtering the hand tremors in flexible surgical robot for experimental research and education and published 2 journal papers in this field since 2012 to date, research assistant (21 researchers) to design high precision and fast dynamic controller for multi-degrees of freedom actuator for experimental research and education and published 1 journal papers in this field since 2013 to date, research assistant (22 researchers) to research of full digital control for nonlinear systems (*e.g.*, Industrial Robot Manipulator, IC Engine, Continuum Robot, and Spherical Motor) for experimental research and education and published 2 journal papers in this field since 2010 to date, and published more than 12 journal papers since 2011 to date. Her current research interests are nonlinear control, artificial control system and applied to FPGA, and IC engine modeling and control.



Iman Nazari is currently Research Assistant at Institute of Advanced Science and Technology, Research and Training Center, IRAN SSP. He is research assistant of team (8 researchers) to design a Micro-electronic Based nonlinear controller for first order delay system since March, 2015, research student (45 researchers) to Nonlinear control of Industrial Robot Manipulator for Experimental Research and Education from October 2010 to October 2011, and published 7 journal papers since 2011 to date. His current research interests are nonlinear control, artificial control system, Microelectronic Device, and HDL design

Ali Roshanzamir is currently Research Assistant at Institute of Advanced Science and Technology, Research and Training Center, IRAN SSP. He is research assistant of team (8 researchers) to design a Micro-electronic Based nonlinear controller for first order delay system since March, 2015, research student (45 researchers) to Nonlinear control of Industrial Robot Manipulator for Experimental Research and Education from June 2010 to June 2011, and published 5 journal papers since 2011 to date. His current research interests are nonlinear control, artificial control system, Microelectronic Device, and HDL design.

Arman Jahed is currently Research Assistant at Institute of Advanced Science and Technology, Research and Training Center, IRAN SSP. He is research assistant of team (8 researchers) to design a Micro-electronic Based nonlinear controller for first order delay system since March, 2015, research student (45 researchers) to Nonlinear control of Industrial Robot Manipulator for Experimental Research and Education from February 2012 to February 2013, and published 5 journal papers since 2012 to date. His current research interests are nonlinear control, artificial control system, Microelectronic Device, and HDL design.