“Performance Evaluation of Hybrid Fixed and Floating-Point PID Controller Architecture on Zynq-7000 FPGA”

P.Revathi [1], Y.L Ajay kumar[2], E.Bala krishna[3],

[1][1]*PG Scholar, [2]Professor,[3]Assistant Professor, Dept of ECE, Anantha Lakshmi institute of technology & sciences, Anantapuramu.*

***Abstract*—PID controllers are crucial for regulating various process variables such as temperature, flow, pressure, and speed in industrial control systems. Therefore, this paper aims to design and implement the real-time FPGA-based PI/PID controller, which is complex regarding memory issues, accuracy, resource utilization, and energy consumption. Thus, this research uses Matlab’s hybridized fixed- and floating-point approach to design the module Xilinx Vivado. The proposed design is realized on a Zynq-7000 FPGA board, and the results of the PI/PID controller implemented on the FPGA Zynq-7000 board model are stimulated. System generator tools are used to generate the hardware description language code. The Xilinx Vivado tool checks the module’s device utilization, timing analysis, and power consumption. The performance is improved in terms of dynamic range, speed, unlimited use of resources, efficiency, and less energy consumption.**

***Keywords*—PI/PID controller, Hybrid technique, Matlab/Simulink, FPGA Zynq-7000, Xilinx Vivado, System generator.**

I. Introduction

Numerous theories and design approaches are continually being developed in the field of automatic controllers. The most widely used controllers are the Proportional Integral (PI) and Proportional Integral Derivative (PID) controllers. They are considered the most common type of controller used in industry because they offer the best balance between cost and value [1]. There are several ways to implement the PI and PID controller to validate hardware, such as a microcontroller, digital signal processing, application-specific integrated circuit (ASIC), and field-programmable gate array (FPGA). The digital implementation is used to program the control algorithm within the chosen hardware, which is more flexible in tuning and advanced control algorithms. Among the digital ones, FPGA-based controllers offer advantages such as high speed, complex functionality, and low power consumption. However, conventional FPGA-based controller implementations have yet to concentrate on optimizing hardware resources [2]. These designs frequently necessitate a substantial quantity of multipliers and adders, thereby failing to leverage the memory-intensive capabilities of FPGAs effectively.

Joao Lima et al. proposed a methodology for an FPGA-based PID controller that uses fixed-point numerical representation. The MATLAB/Simulink environment models and evaluates the performance of different fixed-point representations with a given control process [3], best-precision fixed-point digital PID controller [4]. In [5], the researcher integrated a novel distributed arithmetic-based PID into an FPGA-based digital feedback control system, and the distributed arithmetic algorithms saved 80% of hardware resources and 40% of power consumption. This method also provides good closed-loop performance while using fewer resources, resulting in cost reduction, high speed, and low power consumption. The Xilinx Spartan 3E FPGA outperformed the Altera Cyclone FPGA board in terms of multipliers, adders/subtractors, power consumption, and speed on a low-power PID controller, according to [6]. The Spartan 3E also showed satisfactory results for memory usage, counters, and Look-up Tables (LUT). FPGA is implemented in the PID controller for the DC-DC converter for the low-voltage synchronous buck converter with the experimental validation of an FPGA-based position PI servo controller for the DC controller with dry friction [7].

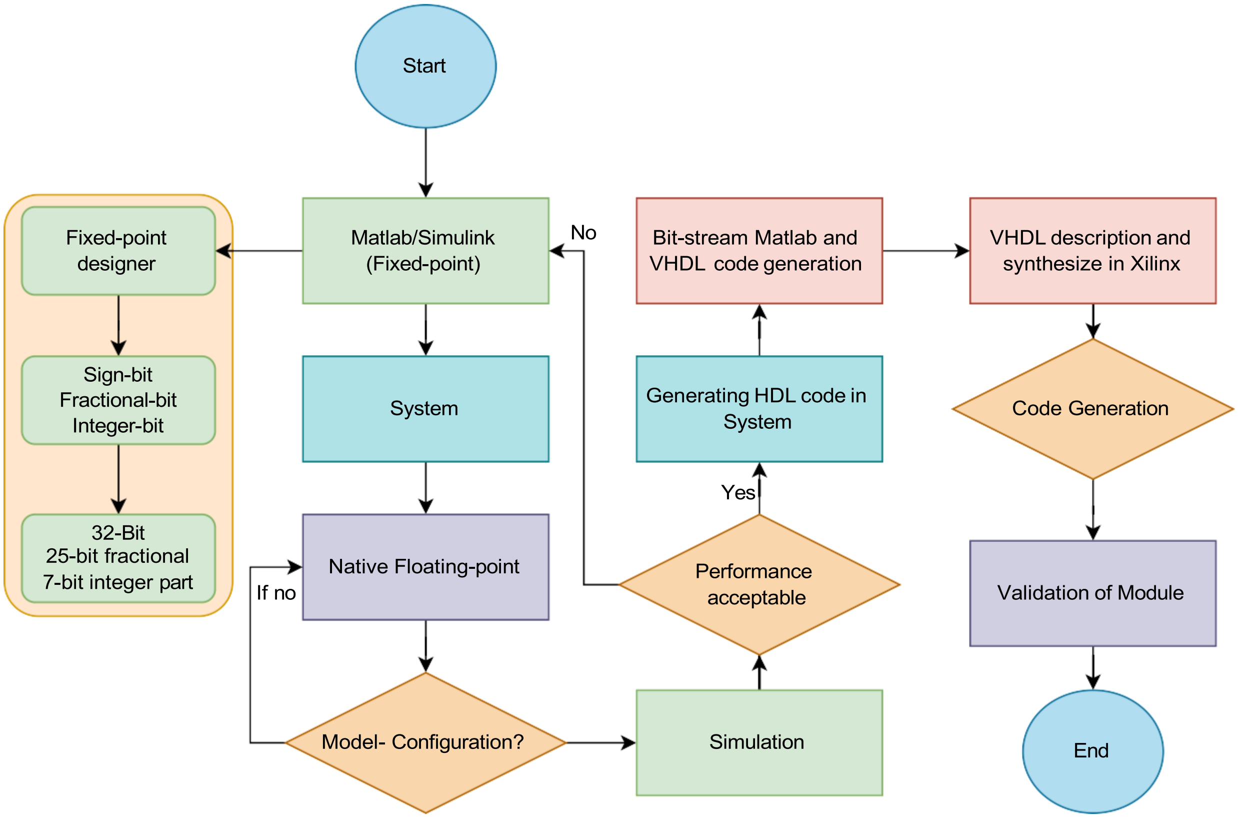


Fig. 1. Implementation flowchart of the proposed hybrid fixed and floating-point approach.

Simulation and experimental systems allow validation in open and closed loops, and the difference between simulation and experimental results shows very close dynamics in transient and steady-state responses [8]. An FPGA-based digital parallel PID controller is implemented for the axis control system where the Altera DEO nano board controls the five-axis robotic arm as a multi-axes system [9]. An FPGA-based robotic manipulator with dual PID controllers optimized using the Chien-Hrones-Reswick technique is proposed in [10] to eliminate overshoot and increase system energy conservation. In [11], the authors developed an FPGA-based PID controller device for diabetes patients that regulates blood pressure using Matlab/Simulink. The system generator toolbox from Xilinx ISE generates the Hardware Description Language (HDL) Code, and the Xilinx ISE tool is used to check the controller's device utilization and timing analysis. In [12], the author discussed about the implementation techniques based PID controller using the FPGA with the detailed performance is given which is based on approaching techniques.

Motivated by the above literature, this paper presents a hybrid fixed-and-floating technique implemented on FPGA-based PI/PID controllers. The figure shown in Fig. 1 outlines the proposed methodology for designing an FPGA-based PID controller module using Matlab/Simulink for simulation. The remaining sections of the paper are organized as follows: section II gives the detailed research methodology used in this article. Lastly, section III discusses the results obtained for the second-order plants and the controllers.

II. Methodology

A. PID Controller

The PID consists of three terms: Proportional (P), Integral (I), and Derivative (D), and each contributes a unique response to reduce the error signal [3], [5], [6].

$$C(s) = \frac{U(s)}{E(s)} = K_p + \frac{K_i}{s} + K_d s. \tag{1}$$

In the proportional control, $U\ (s)$ is changed directly to the error $E(s)$, where K_p is the proportional gain. The integral gain controls the output by integrating the error signal, where K_i means the integral gain, which is reciprocal with the integral time. Derivative gain K_d anticipates future errors and changes based on the rate of change of error, and it helps to reduce overshoot and oscillations [4].

B. FPGA Implementation of PID Controller on Zynq-7000

The proposed work involves implementing a PI/PID controller using Very High-Speed Integrated Circuit Hardware Description Language (VHDL) and Xilinx on Zynq-7000 FPGA, a programmable alternative to ASIC and SOC users. The FPGA offers configurable logic blocks, input-output blocks, and interconnects, making it ideal for designing a range of embedded systems. With its integrated 28nm programmable

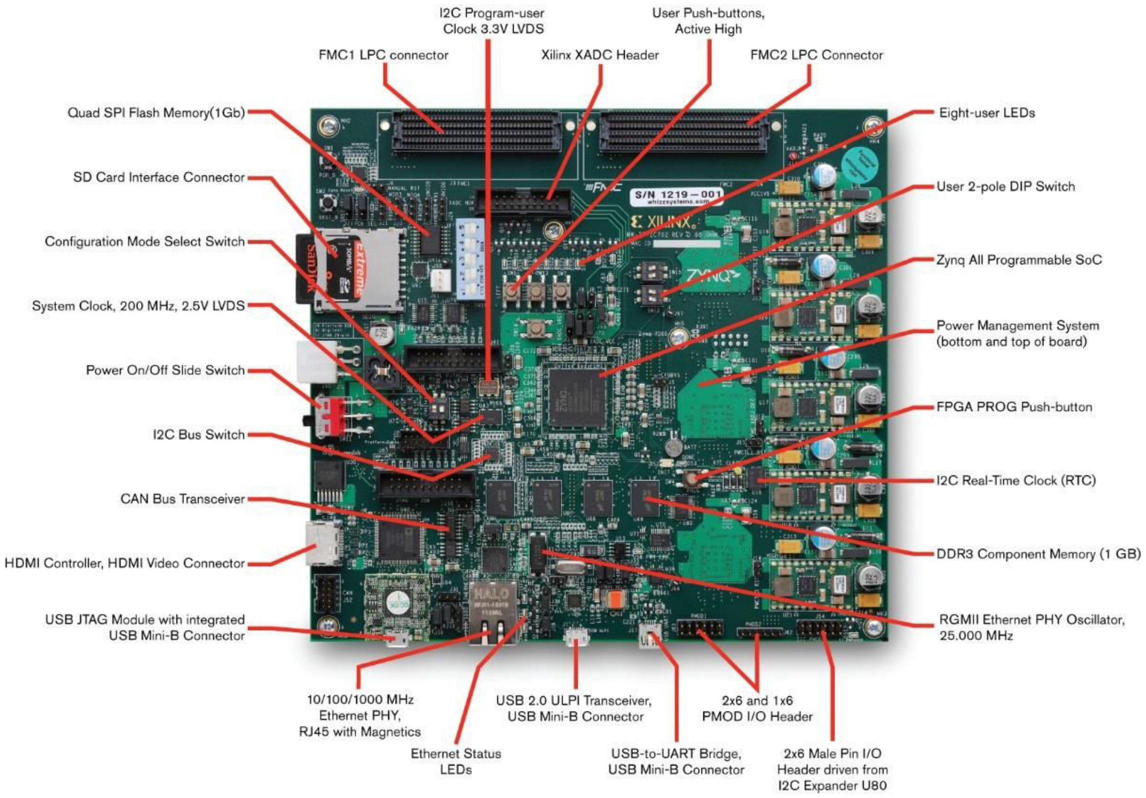

I2C Program-user Clock 3.3V LVDS
User Push-buttons, Active High
FMC1 LPC connector
Xilinx XADC Header
FMC2 LPC Connector
Quad SPI Flash Memory(1Gb)
SD Card Interface Connector
Configuration Mode Select Switch
System Clock, 200 MHz, 2.5V LVDS
Power On/Off Slide Switch
I2C Bus Switch
CAN Bus Transceiver
HDMI Controller, HDMI Video Connector
USB JTAG Module with integrated USB Mini-B Connector
Eight-user LEDs
User 2-pole DIP Switch
Zynq All Programmable SoC
Power Management System (bottom and top of board)
FPGA PROG Push-button
I2C Real-Time Clock (RTC)
DDR3 Component Memory (1 GB)
RGMII Ethernet PHY Oscillator, 25.000 MHz
10/100/1000 MHz Ethernet PHY, RJ45 with Magnetics
Ethernet Status LEDs
USB 2.0 ULPI Transceiver, USB Mini-B Connector
USB-to-UART Bridge, USB Mini-B Connector
2x6 and 1x6 PMOD I/O Header
2x6 Male Pin I/O Header driven from I2C Expander U80

Fig. 2. AMD Zynq ZC7020 FPGA board.

logic, Zynq-7000 surpasses discrete processors and FPGA systems in power and performance levels, offering 6.6M logic cells and transceivers ranging from 6.25Gb/s to 12.5Gb/s. Figure 2 shows the overall key components in the Zynq 7000 SoC ZC702 Evaluation Kit. This FPGA development platform is compact and suitable for portable projects like robots and mobile devices. It supports a wide range of I/O voltages from 1.2 to 3.3 V and has high-performance I/O support for 1.2 to 1.8 V. It has an industrial temperature range of -40°C to +100°C. It comes with DDR3 memory, offering higher operating performance and voltage than DDR2. Additionally, it features an Ethernet PHY for data transmission and reception, a Low Pin Count (LPC) bus for connecting low-bandwidth devices to personal computers, and an XADC core for connecting to AXI4 and providing control interface for the hard macro monitor XADC of the system on the Vertex-7, Kintex-7, and Zynq-7000 devices. [11].

C. *Hybrid Fixed and Floating-point Approach*

The process begins with designing a discrete PID controller, which is implemented on the discrete transfer function of the DC motor. The approach uses the continuous to discrete transformation formula to obtain values for the discrete transfer function while applying it in the Matlab command window while working with the discrete PID controller in the Z-domain form. In Matlab/Simulink, the PID parameters are tuned, and it uses the fixed-point designer toolbox to determine the bits needed for the implementation. The bits are divided into two parts: fractional and integer. The fractional part comprises 25 bits, while the integer part comprises 7 bits when implemented on the system (discrete transfer function) [5], [6], [8], [9].

Furthermore, the native-floating point approach is implemented in the system block (adhering to the IEEE 754 double format standard). This involves configuring the model in the model advisor toolbox, which is included in Matlab/Simulink. Additionally, the HDL code is generated into the system model, and the response of the PI-PID controller is examined via the scope and the Xilinx viewers, aided by Viti's composer. Finally, the HDL code is converted into VHDL and synthesized in the Xilinx Vivado to produce the code for the FPGA board Zynq-7000. At this point, the performance of the Zynq-7000 FPGA board in terms of energy consumption and resource utilization is assessed. A comprehensive design of the FPGA-based PI-PID controller is obtained, and the module is then validated, with code generated into the FPGA hardware model based on the PI-PID controller.

III. Results and Discussions

A. *Plant 1*

The second-order transfer function chosen for the performance analysis in the z-domain is as follows:

$$G(z) = \frac{0.1893z + 0.004675}{z^2 - 1.895z + 0.9048} \tag{2}$$

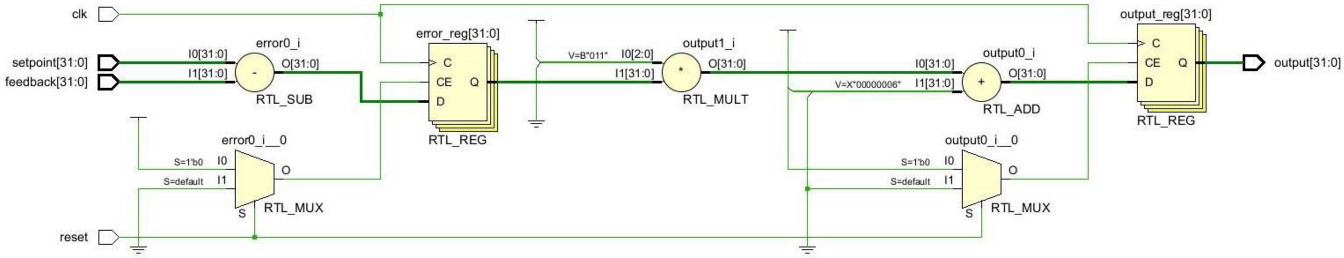


Fig. 3. Schematic design of RTL components description.

The system's tuned PI/PID controller parameters are $K_p = 0.5028$, $K_i = 0.5731$, and $K_d = 0.8788$. The Fixed-point approach is implemented in the model to determine the bits with the help of the fixed-point designer toolbox in MATLAB/Simulink. The bits are divided into two parts: fractional and integral. Twenty-five bits are included in the fractional part, while the integral depends on the 7 bits. The DTC (Data Type conversion) algorithm is applied to the model using the fixed point designer. While using the fixed-point designer tool, one needs to choose the scaling factors to convert the physical values into the fixed-point representation, and that DTC factor maintains the balance between the precision and range. Simulating the fixed-point DTC algorithm under various operating conditions helps us identify potential issues related to accuracy, saturation, and overflow, and it also gives stable control in all operating conditions.

The compatibility of the PI/PID controller model is verified using the model advisor toolbox in the Matlab/Simulink environment. Upon achieving adequate performance, the next step is to target the FPGA Zynq-7000 board and generate the HDL code into the model with 50 MHz frequency. The response of the PI/PID controller is then checked in the scope upon successful generation of the HDL code. Subsequently, the PI/PID controller's VHDL code is implemented in the Xilinx Vivado using the AMD Vitis composer, which is also available in the Matlab/Simulink environment. Further details on the design flow of the FPGA-7000 boards while implementing the PID controller can be found in Fig. 3, which provides a schematic description of the Register Transfer Level (RTL) components involved.

The circuit shows how resources are distributed in the FPGA to implement a 32-bit hybrid fixed-and-floating point technique. The resources used in the FPGA are listed in Table I. Input pins are distributed separately and shared among bits divided by adders and Muxes. The XOR gate is the only gate that shares the 2-input and uses the bits. More information about the input distribution can be found in Table II.

Slices of the look-up-tables and Logics are distributed separately according to Fig. 3, in which distribution of MUX, CARRY, OBUF (Output Buffer), and IBUF (Input buffer). OBUF drives the signal from the device to the external output pads, and IBUF is the single-ended signal. The RTL components' descriptive is given in Table III.

TABLE I
DEVICE UTILIZATION SUMMARY OF XILINX FPGA

Logic Utilization	Available	Used	Utilization
I/O pins	200	96	48.5%
Logic LUTs	40600	510	1.25%
Slice LUTS	40600	510	1.25%
Slice Register	81200	64	0.078%

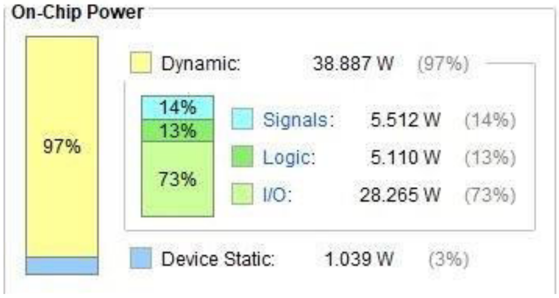
TABLE II
DISTRIBUTION OF INPUT PINS IN FPGA ZYNQ-7000

No: of Inputs	Bits	Adders	Muxes
2-input	28-bit	1	1
2- input	26-bit	-	1
2- input	27-bit	1	-
2- input	24-bit	1	1
2- input	23-bit	-	4
3- input	8-bit	2	-
2- input	8-bit	2	10
2- input	5-bit	1	3
6- input	5-bit	-	3
7- input	5-bit	-	3
4- input	5-bit	-	1
2-input	1-bit	-	40

TABLE III
DISTRIBUTION OF INPUT PINS IN FPGA ZYNQ-7000

Reference name	Used	Functional Category
LUT 6	308	LUT
LUT 3	108	LUT
LUT 5	85	LUT
LUT 4	65	LUT
IBUF	64	I/O
OBUF	32	I/O
LUT 2	30	LUT
CARRY 4	28	CARRY Logic
MUX F7	4	MUX fx
LUT 1	2	LUT

The FPGA-based PID controller's comprehensive design is illustrated in Fig. 3, and it has been developed utilizing the Xilinx Vivado design suite. The design comprises a 32-bit set point, and a feedback closed loop is integrated into the system. The closed-loop feedback mechanism regulates the system's output by comparing the output to the desired set point and applying the necessary corrective action. The Hybrid fixed-and-floating is successfully implemented on an FPGA-based PID controller using a Zynq-7000 board and in the last overall performance report in Fig. 4. The dynamic range of the hybrid technique gives better accuracy and less energy consumption.



On-Chip Power

Dynamic: 38.887 W (97%)

Signals: 5.512 W (14%)

Logic: 5.110 W (13%)

I/O: 28.265 W (73%)

Device Static: 1.039 W (3%)

Fig. 4. Power utilization report.

The implemented model utilizes a hybrid fixed-and-floating point technique. The PI/PID controller response shown in Fig. 5 is observed with a disturbance at 25 seconds. Analysis of the response indicates that the PID controller exhibits a faster settling time and less overshoot than the PI controller. Table IV presents traditional PI and PID controllers' comparative rise time and settling time performance. The PI controller's rise time is 1.82 seconds, whereas the PID controller showed a marginally better response with a rise time of 1.23 seconds. Regarding settling time performance, the PID controller exhibited faster settling and lower overshoot than the PI controller, with values of 0.98 seconds and 1.12 %.

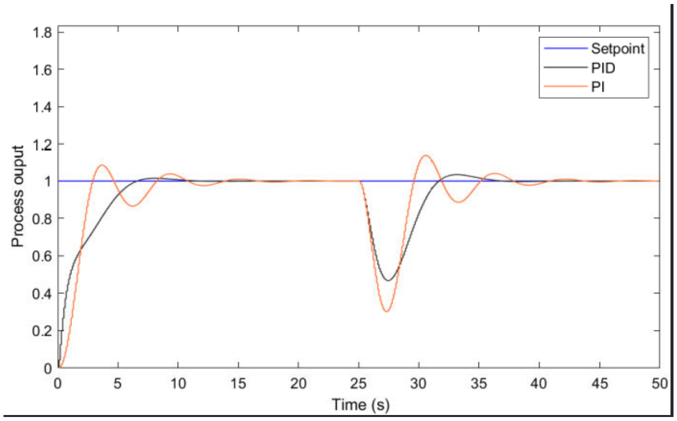


Fig. 5. Disturbance rejection and set-point tracking performance of PI and PID controllers in Plant 1.

TABLE IV
Numerical Performance Analysis of the Process Plant 1

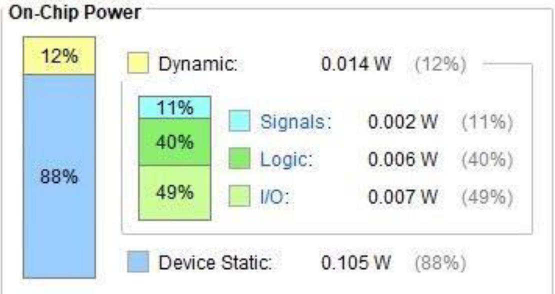
Controller	t_s	t_r	%OS
PI	43.12	1.82	1.12
PID	37.25	1.23	0.98

B. Plant 2

The Hybrid fixed-and-floating technique is also implemented on an FPGA-based PI/PID controller using the second-order transfer function in MATLAB/Simulink.

$$G(z) = \frac{0.004498z + 0.04069}{z^2 - 1.6555z + 0.7408} \tag{3}$$

The tuned PI/PID controller parameters in the hybrid technique for the implementation have been specified as $K_p = 0.3231$, $K_i = 1.425$, and $K_d = 0.0225$. Moreover, the hybrid technique divides the bits into fractional and integral parts. The fractional part consists of 24 bits, while the integral part has 8 bits, implemented using the fixed-point designer. The fixed-point designer is an essential tool that assists in designing fixed-point data types for Simulink models. The FPGA-based PI/PID controller implementation using Zynq-7000 boards and the overall performance of the FPGA is given in Fig. 6 in the power utilization report.



On-Chip Power

Dynamic: 0.014 W (12%)

Signals: 0.002 W (11%)

Logic: 0.006 W (40%)

I/O: 0.007 W (49%)

Device Static: 0.105 W (88%)

Fig. 6. Power utilization report.

Upon examining Fig. 7, it becomes apparent that the PI/PID controller has been integrated with a hybrid fixed-and-floating point implementation. Analysis of this response reveals that the PID controller exhibits a faster settling time than the PI controller, and its overshoot is also considerably less. The performance of traditional PI and PID controllers in terms of their rise time and settling time was compared and presented in Table V. The rise time of the PI controller is obtained as 1.525s, whereas the PID controller exhibited a slightly better response with a rise time of 1.132s. Regarding settling time, the PI controller demonstrated faster settling at 33.654 seconds and a lower overshoot than the PID controller, with respective values of 1.684 % and 1.932 %.

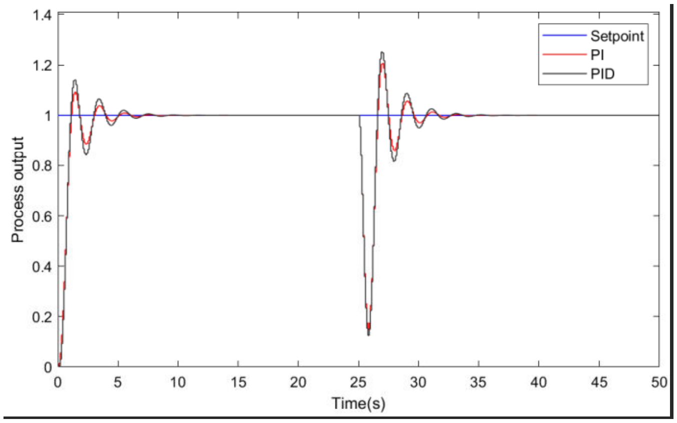


Fig. 7. Disturbance rejection and set-point tracking performance of PI and PID controllers in Plant 2.

The PI/PID controller model is first checked through the model advisor toolbox in Simulink to ensure its proper func-

TABLE V
NUMERICAL PERFORMANCE ANALYSIS OF THE PROCESS PLANT 2

Controller	t_s	t_r	%OS
PI	33.654	1.525	1.684
PID	35.978	1.132	1.932

tioning. Once confirmed, a targeted FPGA is allocated to the model Zynq-7000. The HDL code is then generated into a model for the VHDL description, with the frequency set to 100 MHz. This process is successfully implemented in MATLAB/Simulink. Subsequently, the AMD composer runs the program on the Xilinx Vivado. Finally, the program is generated on the Zynq-7000 FPGA board. The 32-bit RTL description of the components in the schematic design is implemented in Xilinx Vivado. Additionally, an overall utilization report is provided in Table VI. Input pins are distributed separately in the FPGA and shared with the bits, divided into adders and multiplexers. The XOR gate is the only gate that shares the 2-input and uses the bits and the distribution report of the input given in Table VII.

TABLE VI
DEVICE UTILIZATION SUMMARY OF XILINX FPGA

Logic Utilization	Available	Used	Utilization
I/O pins	200	98	49%
Logic LUTs	40600	66	0.16%
Slice LUTS	40600	66	0.16%
Slice Register	81200	64	0.078%

TABLE VII
DISTRIBUTION OF INPUT PINS IN FPGA ZYNQ-7000

No: of Inputs	Bits	Adders	Muxes
2-input	28-bit	2	2
2- input	26-bit	-	1
2-input	27-bit	1	2
2- input	24-bit	1	1
2- input	23-bit	-	4
3- input	25-bit	2	6
2- input	8-bit	2	10
2- input	5-bit	1	3
6- input	5-bit	-	3
7- input	5-bit	-	3
4- input	5-bit	-	1
2-input	1-bit	-	44

Slices of the look-up-tables and Logics are distributed separately according to Fig, in which distribution of MUX, CARRY, OBUF (Output Buffer), and IBUF (Input buffer). OBUF drives the signal from the device to the external output pads, and IBUF is the single-ended signal. The RTL components' descriptive Table VIII is given below.

IV. CONCLUSION

A PID controller model has been developed using the hybrid fixed and floating-point approaches implemented on the FPGA Zynq-7000 board. PID controller is used with the transfer function to validate the model with hybrid approaches in Matlab/Simulink. The VHDL language is used to program the FPGA in Xilinx Vivado, and the plant is composed by Viti's composer. Finally, the performance of this model seems to be better, and the dynamic range of the hardware is up to 97%. It uses more resources and gives more accuracy, speed, and less energy consumption.

TABLE VIII
RTL COMPONENTS UTILIZATIONS

Reference name	Used	Functional Category
LUT 6	20	LUT
LUT 3	15	LUT
LUT 5	22	LUT
LUT 4	5	LUT
IBUF	2	I/O
OBUF	2	I/O
LUT 1	2	LUT
LUT 2	7	LUT
CARRY 4	9	CARRY Logic
MUX F7	2	MUX fx

ACKNOWLEDGEMENTS

This research was funded by Short-Term Internal Research Funding (STIRF) with grant number 015LA0-048.

REFERENCES

[1] M. A. Fawwaz, K. Bingi, R. Ibrahim, P. A. M. Devan, and B. R. Prusty, "Design of PIDDα controller for robust performance of process plants," *Algorithms*, vol. 16, no. 9, p. 437, 2023.

[2] A. A. Aguirre, L. D. Mun˜oz, C. A. Mart´ın, M. J. Ram´ırez, and C. A. Salazar, "Design of digital PID controllers relying on FPGA-based techniques," *IFAC-PapersOnLine*, vol. 51, no. 4, pp. 936–941, 2018.

[3] J. Lima, R. Menotti, J. M. Cardoso, and E. Marques, "A methodology to design FPGA-based PID controllers," in *2006 IEEE International Conference on Systems, Man and Cybernetics*, vol. 3. IEEE, 2006, pp. 2577–2583.

[4] Y. Xu, K. Shuang, S. Jiang, and X. Wu, "Fpga implementation of a best-precision fixed-point digital PID controller," in *2009 International Conference on Measuring Technology and Mechatronics Automation*, vol. 3. IEEE, 2009, pp. 384–387.

[5] Y. F. Chan, M. Moallem, and W. Wang, "Design and implementation of modular FPGA-based PID controllers," *IEEE transactions on Industrial Electronics*, vol. 54, no. 4, pp. 1898–1906, 2007.

[6] B. Sreenivasappa and R. Udaykumar, "Design and implementation of FPGA based low power digital PID controllers," in *2009 International Conference on Industrial and Information Systems (ICIIS)*. IEEE, 2009, pp. 568–573.

[7] S. Chander, P. mod Agarwal, and I. Gupta, "Fpga-based PID controller for DC-DC converter," in *2010 Joint International Conference on Power Electronics, Drives and Energy Systems & 2010 Power India*. IEEE, 2010, pp. 1–6.

[8] L. F. Castano and G. A. Osorio, "Design of a FPGA based position PI servo controller for a DC motor with dry friction," in *2011 VII Southern Conference on Programmable Logic (SPL)*. IEEE, 2011, pp. 75–80.

[9] S¸ . Akkaya, O. Akbatı, and H. Go¨rgu¨n, "Multiple closed loop system control with digital pid controller using fpga," in *2014 International Conference on Control, Decision and Information Technologies (CoDIT)*. IEEE, 2014, pp. 764–769.

[10] P. Chotikunnan and R. Chotikunnan, "Dual design PID controller for robotic manipulator application," *Journal of Robotics and Control (JRC)*, vol. 4, no. 1, pp. 23–34, 2023.

[11] A. Kumar and R. Phadke, "Design of digital PID controller for blood glucose monitoring system," *International Journal of Engineering Research & Technology*, vol. 3, no. 12, pp. 307–11, 2014.

[12] A. Ali, K. Bingi, R. Ibrahim, P. A. M. Devan, and K. Devika, "A review on FPGA implementation of fractional-order systems and PID controllers," *AEU-International Journal of Electronics and Communications*, p. 155218, 2024.