ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645
Indian Journal of Science and Technology, *Vol 11(16), DOI: 10.17485/ijst/2018/v11i16/121762, April 2018*

Design and FPGA-Implementation of a PID Controller for Temperature Control in a Refrigeration System

Rim Ben Ali*, Mohamed Akram Jaballah, Emna Aridhi, Abdelkader Mami

Department of Physics, Faculté des Sciences de Tunis, Université de Tunis El Manar, Campus Universitaire Farhat Hached B.P. n° 94 - Rommana 1068 Tunis, Tunisie;
rim.benali@etudiant-fst.utm.tn, jaballahakram@gmail.com, aridhi_emna@yahoo.fr, abdelkader.mami@fst.rnu.tn

Abstract

Objectives: The universal controller PID (Proportional-Integra-Derivative) is widely used in automation systems. **Methods**: This work is focused to develop a PID controller for a refrigeration system in order to implement it in a Field Programmable Gate Array (FPGA) by using the Very High Speed Integrated Circuit Hardware Description Language (VHDL). Firstly, the nonlinear model of the refrigeration system is presented by a third order state system and simulated under Matlab/Simulink environment to develop its PID controller, which is aimed to adjust the indoor temperature. Further, the synthetizable program of the PID algorithm has been implemented on a Map Altera DE0-nano Kit using the Quartus II software. **Findings**: The performance of the proposed controller has been successfully validated with good tracking results. **Application**: The FPGA target presents a good solution to implement the PID algorithm.

Keywords: FPGA-Implementation, PID Controller, Quartus II, Refrigeration System, Temprature Control, Altera DE0-Nano Kit.

1. Introduction

The refrigeration system is used in many fields like lifestyle, industry, agriculture, etc. The complex dynamic system presents a high nonlinearity and properties of thermodynamic coupling, at the same time is expensive, while its power range is varied from 1 kW to above 1 MW[1]. Traditionally, it is designed with an on-off control in order to adjust the cooling demand[2]. Recently, the development of the control strategies has been largely enhanced the efficiency and reliability of the systems in order to regulate its internal temperature more efficiently and precisely. For this purpose, many control strategies have been used like the Model Predictive Control (MPC)[3,4], artificial intelligence control[5,6], LQG control[7], decoupling multivariable control[8], robust control[9] and the famous controller PID[10,11]. This latter is classified among the most known and used control technique in the industry regarding its precision, feasibility and simplicity[12]. However, the development of the controller applied to the refrigeration systems was the objective of several research works that results obtained prove its performance[13].

Furthermore, the implementation of the PID control algorithm on an FPGA target start by the development of the VHDL code in order to describe the design to be implemented. However, the FPGA provides a good solution to this issue that due to its advantages: embedding processor, low power consumption, programmable hard-

**Author for correspondence*

wired feature, shorter design cycle, fast time-to-market, higher density for the implementation of the digital system[14], and density in number of logic gates that can reach millions of equivalent logic gates (Xilinx, Altera), which is sufficient to implement a full 32-bit CPU together in a single device with most peripheries. On the other, the VHDL language is widely used by the designers and are supposed to be implemented on an FPGA structure due to its simplicity and flexibility in the hardware description of the FPGA circuits, which is presented as a platform very convenient capable to contain various programs and complex commands like the PID controller[15].

This work aims to develop the PID controller applied to the refrigeration system and to present the steps to implement the synthesizable VHDL program of the regulator using the fixed point representation. The paper is structured as follows: Section II gives the mathematical model describing the heat transfer in the refrigeration system. Section III is devoted to explaining the PID controller, which is simulated using Matlab/Simulink environment. Section IV illustrated the theories overview of the map Altera DE0-nano Kit. Sections V presented the simulation results of the VHDL algorithm described the PID controller by using Quartus II and the steps of the FPGA implementation. The last section is devoted to conclude this paper.

2. Modeling of the Refrigeration System

The refrigeration system is a thermal machine, which is generally composed of four elements: a compressor, an expansion valve, an evaporator and a condenser[16,17]. The heat transfer inside the system is shown by Figure 1.

In this study, the model of the refrigeration system aims to calculate the indoor temperature according to the temperature at the wall of the evaporator that can be determined using the following third order state system:

$$A = \begin{pmatrix} -2.986e-5 & 0.001438 & 0 \\ 2.986e-5 & -0.00613 & 0.001138 \\ 0 & 0.0001415 & -0.01462 \end{pmatrix}$$
$$B = \begin{pmatrix} 0 \\ 0.9926 \\ 0 \end{pmatrix}$$
$$C = \begin{pmatrix} 0 & 0.004584 & 0 \end{pmatrix} \quad (1)$$

The continuous and the discrete transfer functions of the system are given by the following equations:

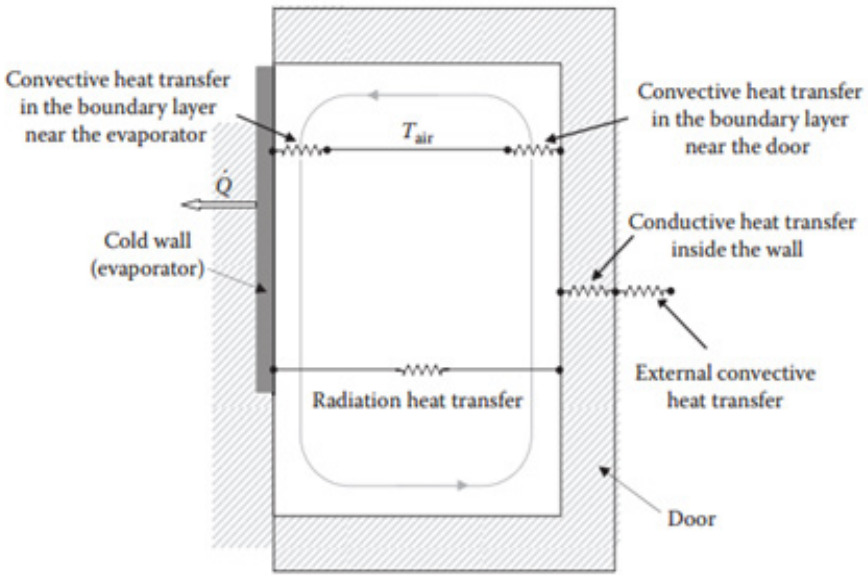

Figure 1. The heat exchange and airflow inside the refrigeration system[17].

$$F(p) = \frac{0.004551p^2 + 6.668.10^{-5}\,p + 1.987.10^{-9}}{p^3 + 0.02078p^2 + 9.006.10^{-5}\,p + 2.044.10^{-9}} \quad (2)$$

$$F(z) = \frac{0.1247z^2 - 0.205z + 0.08035}{z^3 + 2.476z^2 + 2.012z + 0.5361} \quad (3)$$

3. Control Design

The PID controller aims to reduce the error between the set-point and the measured value to enhance the system accuracy (temperature) by taking into account the speed

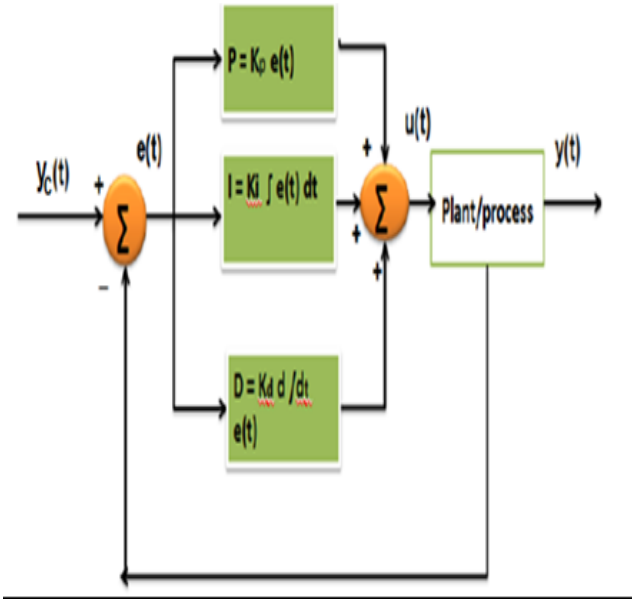

Figure 2. Diagram of the PID controller[18].

of the error variance between the value to be controlled and the set-point. It minimizes the disruption and adapts to new instructions to correct this error.

The technique of the PID controller requires specification of three parameters: the proportional gain (K_p), the integral gain (K_i) and the derivative gain (K_d). The primary design goals of the controller are:

Stabilizing a system potentially unstable.

Obtaining a good adjusting.

Minimizing the integral absolute tracking error.

Increasing the response time of the system.

3.1 Mathematical Equations of the PID Controller

Generally, the PID controller is represented by the following equation:[18]

$$u(t) = K_p\left(e(t) + \frac{1}{T_i}\int_0^t e(t)dt + T_d\, d\,\frac{e(t)}{dt}\right) \quad (4)$$

with:

K_p: the proportional gain.

T_d: the derivative time constant.

T_i: the integral time constant.

$e(t)$ the error signal.

$u(t)$ the output of the regulator.

The equation (3) can be transformed into a mathematical description in discrete time:[18]

$$u(k) = u(k-1) + K_p x_1(k) + K_i\, x_2(k) + K_d\, x_3(k) \quad (5)$$

with:

$$x_1(k) = e(k) - e(k-1) \quad (6)$$

$$x_2(k) = e(k) \quad (7)$$

$$x_3(k) = e(k) - 2e(k-1) + e(k-2) \quad (8)$$

K_i the gain of the integral term.

K_d: the gain of the derivative term.

Therefore, we can obtain the following equation of the PID:[19]

$$u(k) = u(k-1) + K_0 e(k) + K_1\ e(k-1) + K_2 e(k-2) \quad (9)$$

with:

$$K_0 = K_p + K_i\ + K_d\ ` \quad (10)$$

$$K_1 = -K_p + 2K_d \quad (11)$$

$$K_2 = K_d \quad (12)$$

3.2 PID Algorithm

Once the variables and parameters are initialized, the controller calculates at each sample time a new value for the adjusted signal. using the algorithm shown in Figure 3[20].

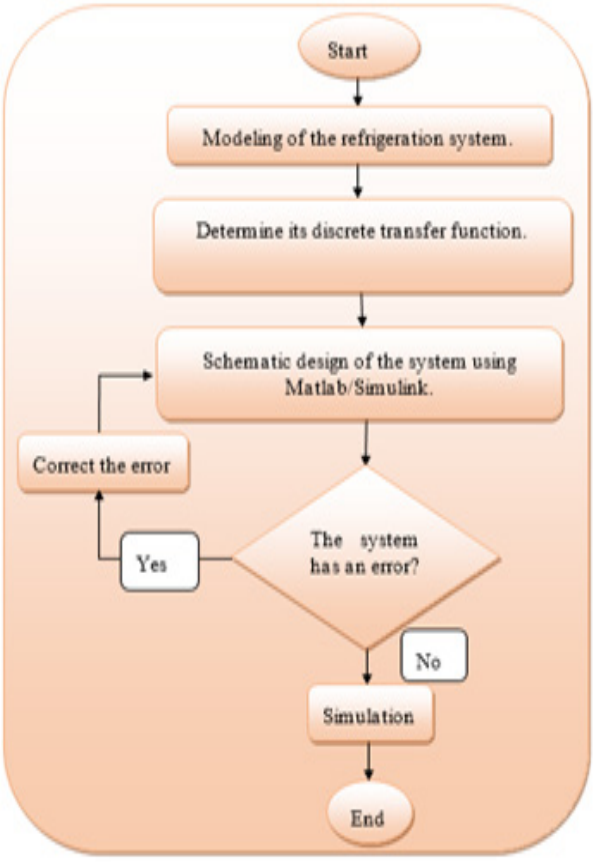

Figure 3. PID algorithm.

3.3 Determination of the PID Parameters

The PID controller has a three adjustable parameters such as: the constant of proportionality, the time derivative constant and the time constant of integration. However, many methods can be used to find the value of these parameters such as Ziegler-Nichols[21] and Takahashi[22]. In this study, the Takahashi method was used in a closed loop where we found:

The gain: $K_c = 1.248$.

The period of oscillations: $T_c = 60\ s$.

The, K_p, K_i and K_d are computed using the following equations:

$$K_p = 0.6K_c\left(1 - \frac{T_e}{T_c}\right) \quad (13)$$

$$\frac{K_p}{T_i} = \frac{1.2K_c}{T_c} \quad (14)$$

$$K_p T_d = \frac{0.3K_c T_c}{4} \quad (15)$$

$$K_i = \frac{T_e}{T_i} \quad (16)$$

$$K_d = \frac{T_d}{T_e} \quad (17)$$

with, T_e is the sample time, which is equal to 30 seconds. This gives:

$$K_p = 0.374; \qquad K_i = 2; \qquad K_d = 0.5$$

3.4 Simulation Results

The Simulink model of the PID controller applied to our process (refrigeration system), which is presented by the

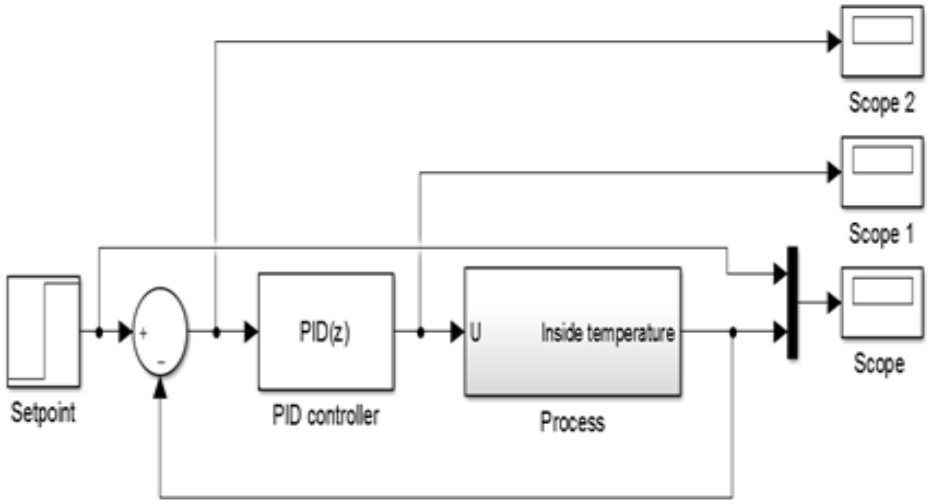

Figure 4. Simulink model of the refrigeration system controlled by the PID controller.

third order state system (Eq. 1), is shown in Figure 4.

The simulation results of the proposed model are illustrated in the Figures 5 and 6:

The set-point is fixed at:

- 3 °C from 0 to 4000 seconds.
- 6 °C from 4000 to 8000 seconds.

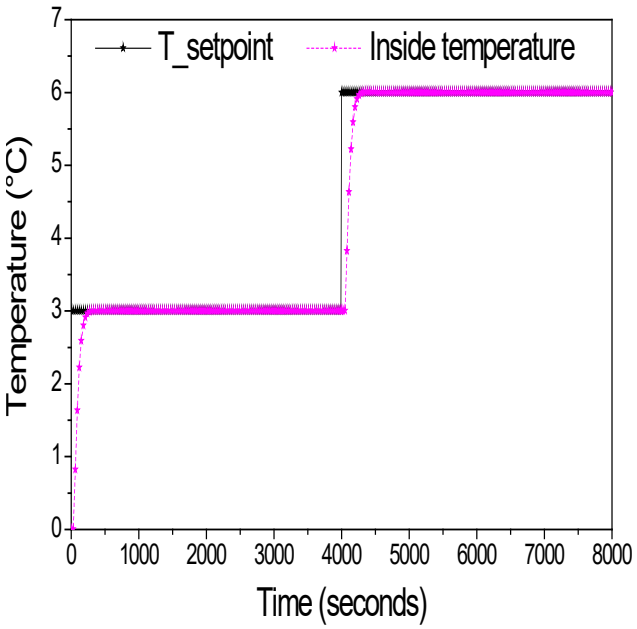

Figure 5. Variation of the set-point and the output temperature.

As has been shown in Figure 5, the internal temperature reaches the set-point value with a little error (Figure 6) and the generated control signal is shown in Figure 7.

The values of the root-mean-square error (RMS) between the superheat degree of the refrigeration system

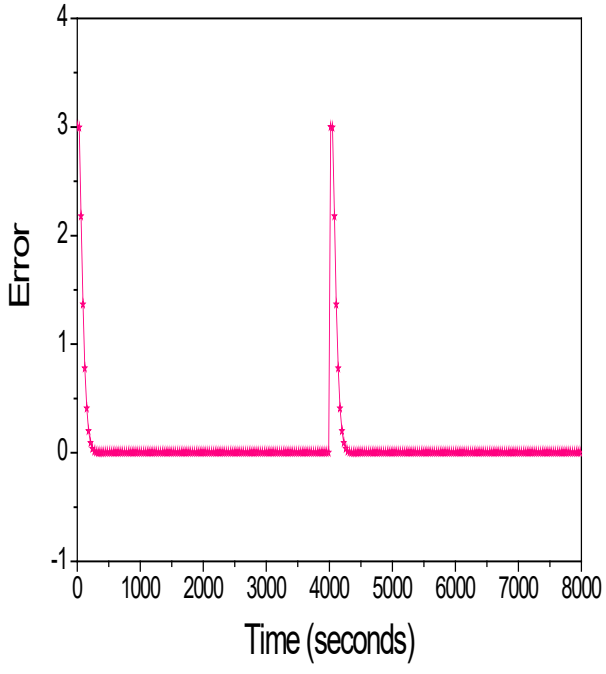

Figure 6. Error signal.

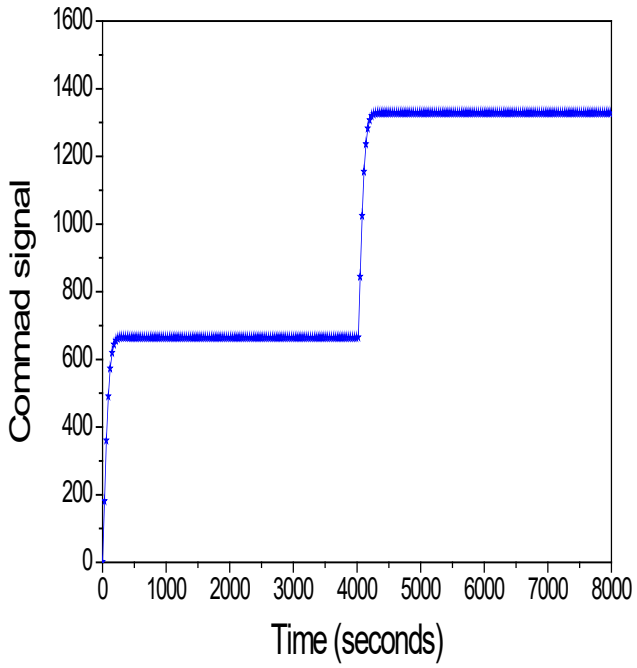

Figure 7. Control signal.

(T_{esh}) and its setting values ($\bar{T}_{esh}$) can be calculated using

Eq 18, in order to evaluate the performance of the control scheme in closed-loop[23].

$$RMS = \sqrt{\frac{\sum_{N_0}^{N_{end}} \left(\bar{T}_{esh} - T_{esh}\right)^2}{\left(N_{end} - N_0\right)}} \qquad (18)$$

with, N_0 is the starting point of the sampling range of the experiment and N_{end} is the ending point. In this case, RMS is equal to 0.0001 that show the efficiency of the PID controller to achieve the desired internal temperature with a little error.

4. Materials and Methods

This section is devoted to present the steps to follow in order to simulate and to implement the VHDL program of the PID controller. For this purpose, we have used:

- The Quartus II software: It provides a complete simulation environment for complex FPGA designs.
- The FPGA target: It is becoming a critical part of every real system requiring a large volume of logic gates with a short computation time[24]. They are a part of the ASIC family and most of them are designed with CMOS technology[15].

4.1 Map Altera DE0-Nano Kit

The FPGA technology (DE0-Nano board) has been chosen in this study, which is designed to minimize the propagation delay between the implemented logic and also to be configured as the user desire. The DE0-Nano is ideal for use and can be configured to implement any system design. It features a powerful Altera Cyclone IV FPGA with 22,320 logic elements. It is characterized by[25]:

- 32 MB of SDRAM.
- 64 Mb serial configuration memory device.
- 2 Kb EEPROM.

For connecting to real sensors, the DE0-Nano board includes:

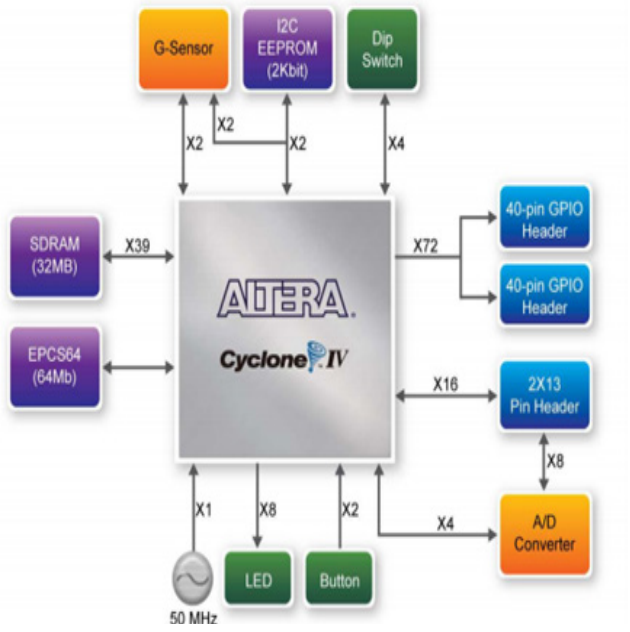

Figure 8. De0-nano board.

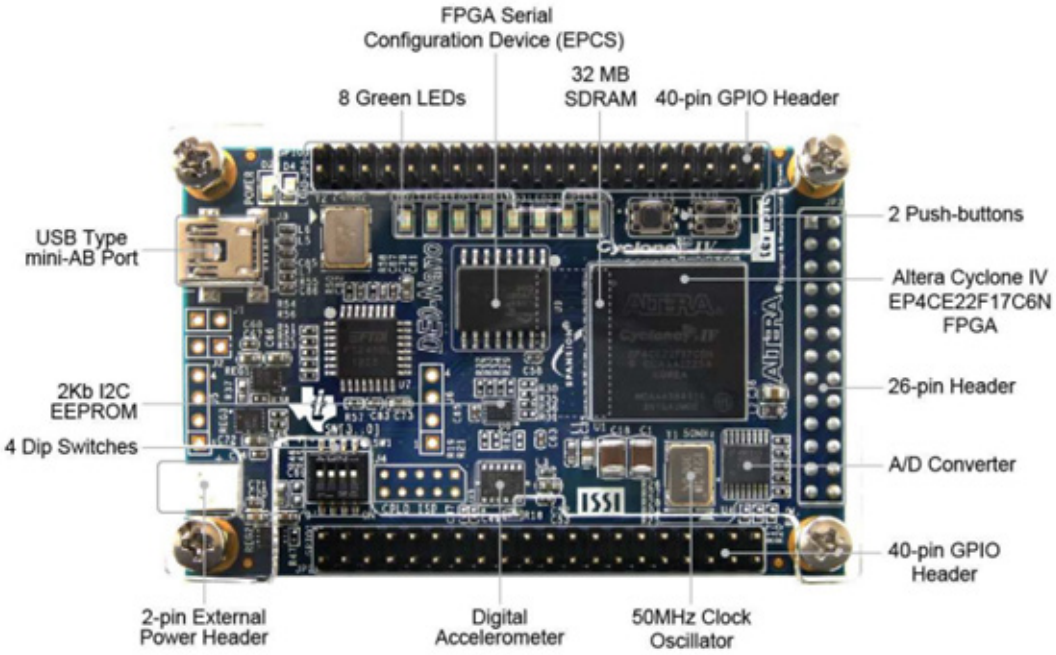

Figure 9. Top view of the De0-nano board[25].

- Analog Devices 13-bit.
- National Semiconductor 8-channel.
- 12-bit A/D converter.
- 3-axis accelerometer device.
- Built-in USB Blaster that is used for FPGA programming.
- Expansion headers.

Inputs and outputs include:

- Set of 4 dip-switches.
- 8 user LEDs.
- 2 pushbuttons.

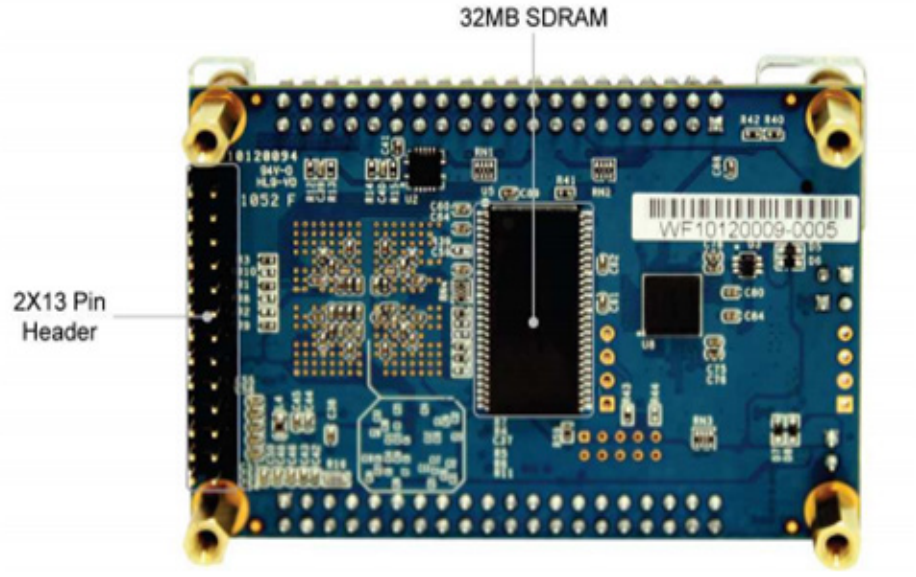

Figure 10. Bottom view of the De0-nano board[25].

Figure 8 showed the block diagram of the DE0-Nano board.

Figures 9 and10 showed the top and bottom views of the DE0-Nano board. They indicate the locations of the key components and the connectors.

4.2 VHDL Programming

HDL language is used to implement the programs and designs on an FPGA target. Especially, the VHDL language, which is used with different specific signal types. For the calculation inside architectures, we can use the fixed point "Sfixed/ufixed" or "unsigned/signed" types and we kept all the main program and its sub programs PORTS as "std_logic_vector".

In fact, a conversion of the fixed point signals to the "std_logic_vector" type, is necessary at the end of each subprogram according to the size of each signal. The first position of the decimal point is should be fixed by computing the number of bits corresponding to the integer part whereas the fractional part is the remaining bits. The overflow issue exists when using the "std_logic_vector" type and it can be solved by the use of fixed-point type. The size of each fixed point signal is chosen according to the existing variables in its expression. The fixed-point type allows to interpret the synthetiz able real type of this work better than the "std_logic_vector" type.

4.3 Configuration

Figure 11 mentioned the steps to follow in order to implement a VHDL program on an FPGA target.

It starts by writing the program using the description language (VHDL or Verilog) and finish by the FPGA configuration (downloading the bit stream file). The configuration data (LUT, distributed RAM, switch control, etc) are loaded with static stores (SRAM) of CMOS technology.But, they will be lost when the power supply is cut off.

5. PID Implementation

The equation of the PID controller can be written as follows[22]:

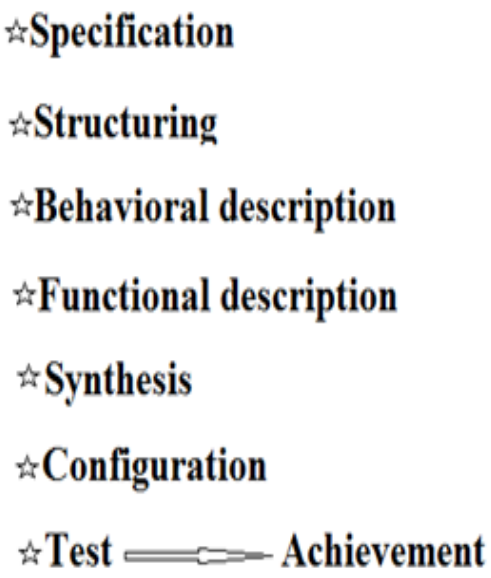

Figure 11. Configuration and impelmentation steps.

The continuous approximations of the PID controller:

$$u(t) = K_p\left(e(t) + \frac{1}{T_i}\int_0^t e(t)dt + T_d\frac{de(t)}{dt}\right) \tag{19}$$

The discrete approximations of the PID controller:

$$u[k-1] = K_p e[k-1] + K_i\sum_{j=0}^{k=1} e[j] + K_d\{e[k-1] - e[k-2]\} \tag{20}$$

$$u[k] = u[k-1] + (K_p + K_i + K_d)e[k] - (K_p + 2K_d)e[k-1] + K_d e[k-2] \tag{21}$$

$$u[k] = u[k-1] + K_0 e[k] + K_1 e[k-1] + K_d e[k-2] \tag{22}$$

In order to implement the PID controller on an FPGA target, it is necessary to cut the previous equations in the mathematical operations of addition and multiplication as follow:

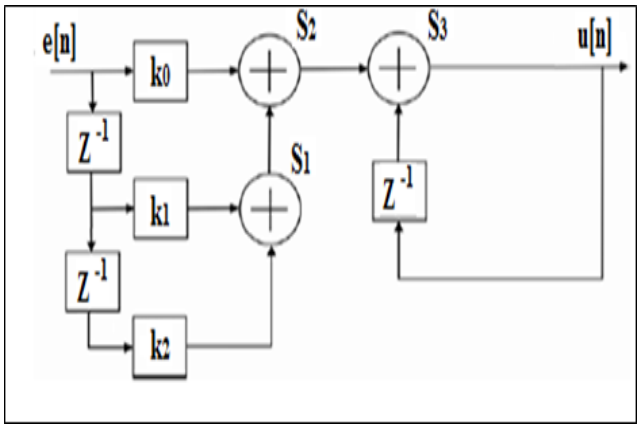

Figure 12. Hardware architecture of the PID controller.

$$p_0 = K_0 e[k] \tag{23}$$

$$p_1 = K_1 e[k-1] \tag{24}$$

$$p_2 = K_2 e[k-2] \tag{25}$$

$$S_1 = p_1 + p_2 \tag{26}$$

$$S_2 = p_0 + S_1 \tag{27}$$

$$S_3 = u[k] = u[k\text{-}1] + S_2 \tag{28}$$

Figure 12 shows the structure of the HDL description of the PID controller.

5.1 Implementation of the Digital Controller PID using Quartus II

This section presents the procedure to follow in order to implement the fixed point-VHDL algorithm on the FPGA target using Quartus II, which presents an EDA (Electronic Design Automation) of the FPGA. Figure 13 presents the initial interface of the software.

The VHDL design of the PID controller is carried out by using Eq. 22, which presents blocks of multiplication, addition, and delay modeled by the D scale of flips flops. It is developed by Quartz II software to be implemented on the Altera DE0-Nano board as shown in Figure 17.

The compilation and the synthesis process results of the program are shown in Figures 14 and 15.

The material resources used in the VHDL program are summarized in Figure 16.

The VHDL program of the PID controller has three inputs and one output:

Inputs: Clock (frequency 50 MHz), pidinput is "std_logic_vector (11 downto 0)" and key[0].

Outputs: the pidoutput is "std_logic_vector (11 downto 0)" and key[0]. The choice of 12 bits is due to the use of the DAC for displaying the output on the oscilloscope.

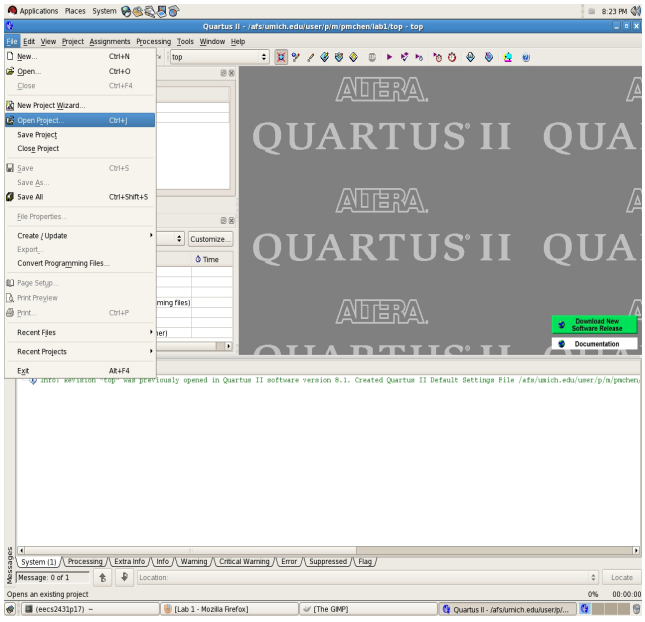

Figure 13. Initial interface of the Quartus II.

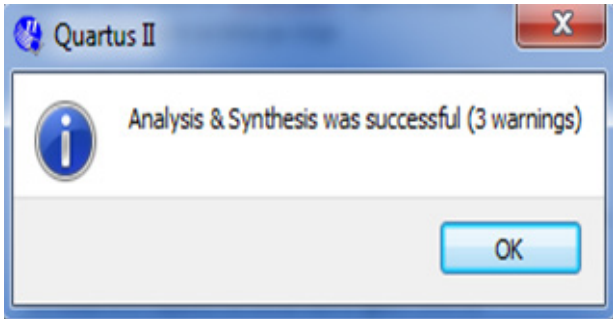

Figure 14. Compiler message.

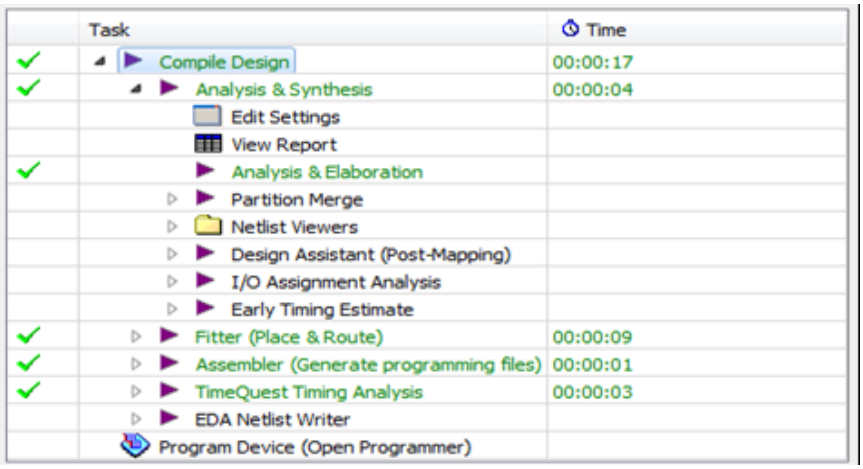

Figure 15. Synthesis process.

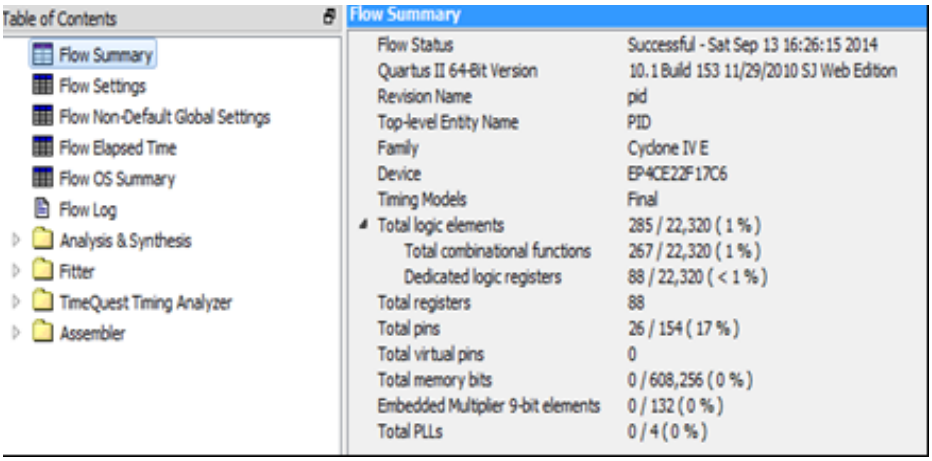

Figure 16. Report of compilation.

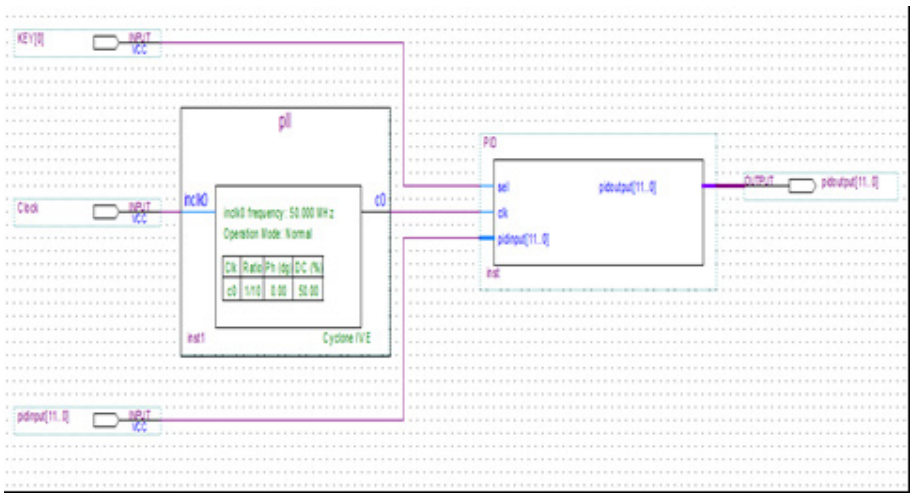

Figure 17. Global schematic description of the PID controller.

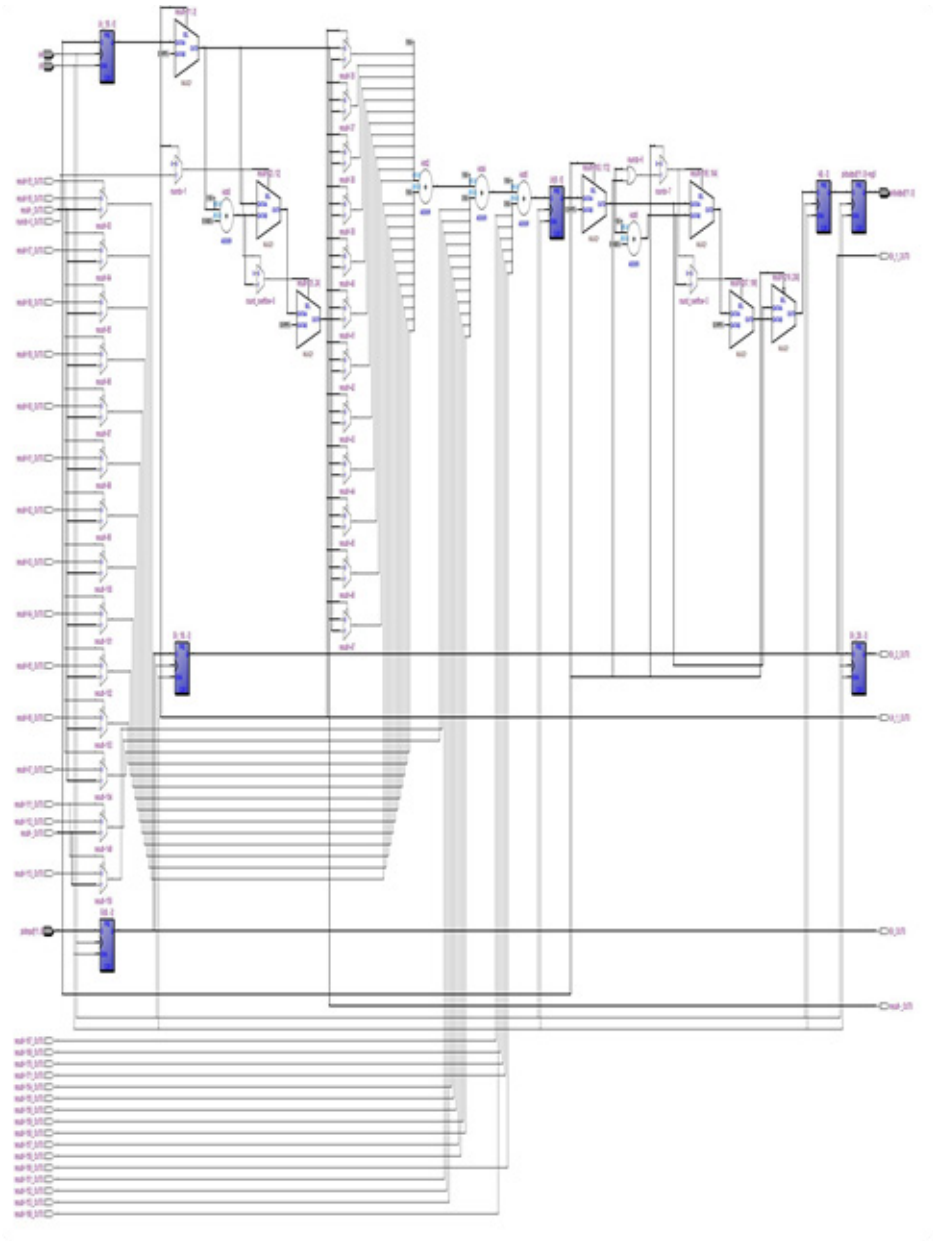

Figure 18. RTL view of the PID controller.

Therefore, we obtained a VHDL description (pid.VHD) as well as a schematic description (pid.BDF) ensuring the same task.

In fact, the corresponding RTL scheme that describes the VHDL design of the PID controller is illustrated in Figure 18.

To achieve a good performance in computation time, it is necessary to reduce the used port logic. Indeed, to complete the implementation, it must be started by making the specification of pines assigned to the ports of the inputs/outputs of our program after the creation of a "pid.cmp" file, which contains the location of the ports. This process is presented in Figure 19.

The final step of the implementation consists of programming the board through a JTAG cable. Firstly, we must connect the USB cable between the map DE0-

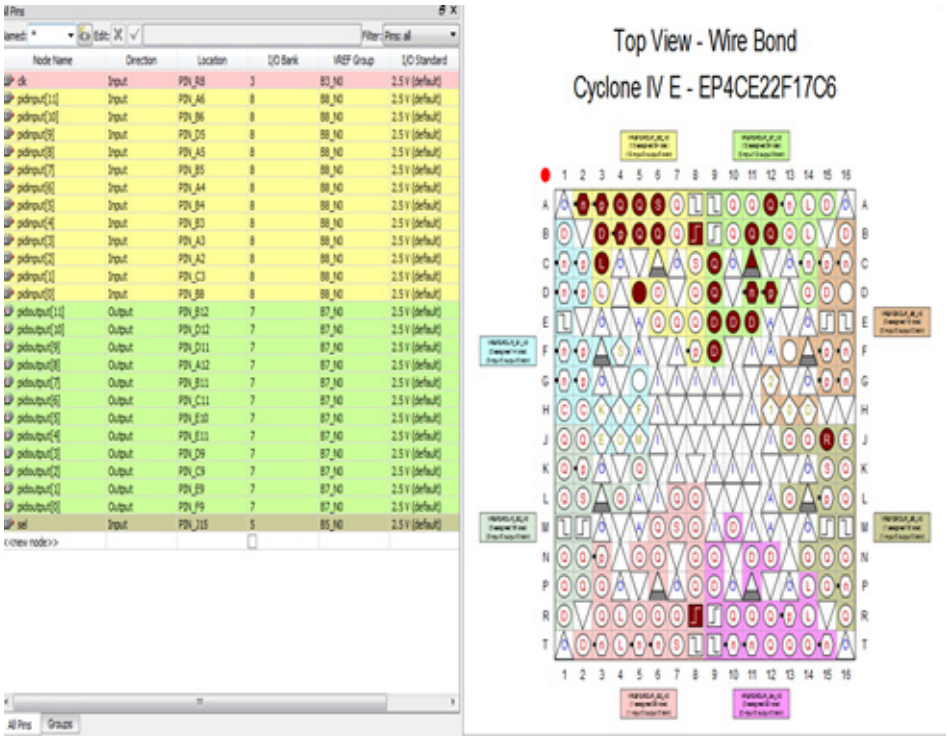

Figure 19. Location of the PID ports on the Altera board.

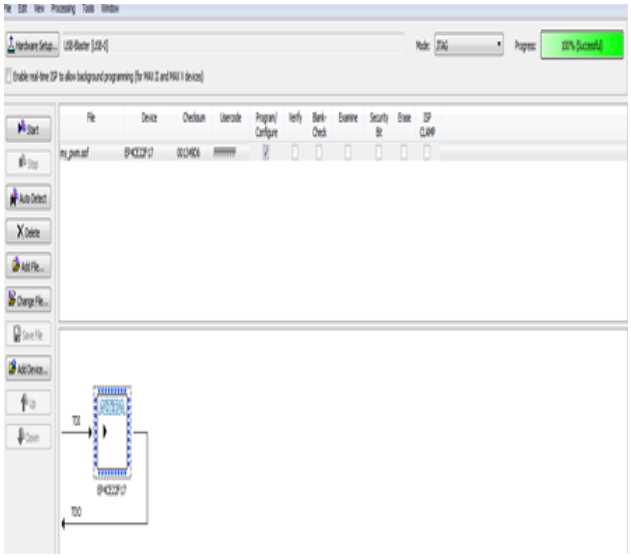

Figure 20. Configuration of the FPGA through the JTAG cable.

Nano and the computer and click on the button «Start » to perform the implementation. Therefore, a new tab is open, which is shown in Figure 20. Then, the bit stream file was successfully downloaded in the FPGA target. The displayed comment «100% Successful» indicates that the VHDL program of the PID controller is well implemented on the map Altera IV using the JTAG cable.

6. Conclusion

In this paper, a PID controller is simulated under Matlab/Simulink environment and its synthetizable VHDL algorithm is implemented on a Map Altera DE0-nano Kit. This kind of FPGA device is very convenient to implement the PID controller that is easy to program, to configure and to manipulate them through powerful and optimized software like Quartus II.

Therefore, three main factors are analyzed:

- Modelling: the thermal behavior of the refrigeration system has been modelled.
- Control: the PID controller is designed considering the dynamic modelling of the refrigeration system. However, the Takahashi method is used to deter-

mine its parameters, it ensures that the controlled variables track their set points.

- Implementation: the FPGA target is used to implement the PID algorithm, which presents a good choice due to its higher density for the digital system implementation.

The simulation results showed the efficiency of the PID controller to ensure a high performance and a good variation of the internal temperature according to the set-point. In future works, we will try to make a comparison between the VHDL program, already made, and the Matlab HDL Coder, to control the system in real time.

7. References

1. Bejarano G, Alfaya JA, Ortega MG, Vargas M. On the difficulty of globally optimally controlling refrigeration systems. Applied Thermal Engineering. 2017; 111:1143–57. Crossref.
2. Dantas TSS, Franco IC, Fileti AMF, Silva FV. Dynamic linear modeling of a refrigeration process with electronic expansion valve actuator. International Journal of Refrigeration. 2017; 75:311–21. Crossref.
3. Elliott MS, Rasmussen BP. Decentralized model predictive control of a multi-evaporator air conditioning system. Control Engineering Practice. 2013; 21(12):1665–77. Crossref.
4. Huang Y, Khajepour A, Bagheri F, Bahrami M. Optimal energy-efficient predictive controllers in automotive air-conditioning/refrigeration systems. Applied Energy. 2016; 184:605–18. Crossref.
5. Ardabili F, Mahmoudi F, Gundoshmian FT. Modeling and simulation controlling system of HVAC using fuzzy and predictive (radial basis function, RBF) controllers. Journal of Building Engineering. 2016; 6:301–8. Crossref.
6. Mohanraj M, Jayaraj S, Muraleedharan C. Applications of artificial neural networks for refrigeration, air-conditioning and heat pump systems - A review. Renewable and Sustainable Energy Reviews. 2012; 16(2):1340–58. Crossref.
7. Schurt LC, Hermes CJL, Neto AT. A model-driven multivariable controller for vapor compression refrigeration systems. International Journal of Refrigeration. 2009; 32(7):1672–82. Crossref.
8. Shen Y, Cai WJ, Li S. Normalized decoupling control for high-dimensional MIMO processes for application in room temperature control HVAC systems. Control Engineering Practice. 2010; 18(6):652–64. Crossref.
9. X. Xu, S. Wang, G. Huang. Robust MPC for temperature control of air-conditioning systems concerning on constraints and multi type uncertainties. Building Services Engineering Research and Technology. 2010; 31(1):39–55. Crossref.
10. Yabanova I, Keçebaş A. Development of ANN model for geothermal district heating system and a novel PID-based control strategy. Applied Thermal Engineering. 2013; 51(1–2):908–916. Crossref.
11. Salazar M, Méndez F. PID control for a single-stage transcritical CO2 refrigeration cycle. Applied Thermal Engineering. 2014; 67(1–2):429–38. Crossref.
12. Moradi H, Setayesh H, Alasty A. PID-Fuzzy control of air handling units in the presence of uncertainty. International Journal of Thermal Sciences. 2016; 109:123–35. Crossref.
13. Bi Q. Advanced controller auto-tuning and its application in HVAC systems. Control Engineering Practice. 2008; 8(6):633–44. Crossref.
14. Ramadan EA, El-Bardini M, Fkirin MA. Design and FPGA-implementation of an improved adaptive fuzzy logic controller for DC motor speed control. Ain Shams Engineering Journal. 2014; 5(3):803–16. Crossref.
15. Kulisz J, Chmiel M, Krzyzyk A, Rosół M. A hardware implementation of arithmetic operations for an FPGA-based programmable logic controller. IFAC-Papers On Line. 2015; 28(4):460–5. Crossref.
16. Yin X, Li S, Cai W. Enhanced-efficiency operating variables selection for vapor compression refrigeration cycle system. Computers and Chemical Engineering. 2015; 80:1–14. Crossref.
17. Amara BS, Flick D, Alvarez G, Laguerre O. Heat transfer and air flow in domestic refrigerators. 21st IIR International Congress of Refrigeration; 2013. p. 1–30.

18. Nema R, Thakur R, Gupta R. Design & Implementation of FPGA Based On PID Controller with Motor & Sensor. 2013; 1(3):11–13.
19. Wang X, Wang H. Design and implementation of intelligent PID control system of high-efficiency comber. Procedia Engineering. 2011; 24:355–9. Crossref.
20. Hamid NHA, Kamal MM, Yahaya FH. Application of PID controller in controlling refrigerator temperature. Proceedings of 2009 5th International Colloquium on Signal Processing and its Applications, CSPA; 2009. p. 378–84. Crossref.
21. Ekren O, Sahin S, Isler Y. Comparison of different controllers for variable speed compressor and electronic expansion valve. International Journal of Refrigeration. 2010; 33(6):1161–8. Crossref.
22. P. I. D. Controller [Internet]. [cited 2018 Mar 03]. Available from: https://en.wikipedia.org/wiki/PID_controller.
23. Yin X, Wang X, Li S, Cai W. Energy-efficiency-oriented cascade control for vapor compression refrigeration cycle systems. Energy. 2016; 116:1006–19. Crossref.
24. Serrano J. Introduction to FPGA design. 8th Workshop on Electronics for LHC Experiments; 2004. p. 231–47.
25. User Manual, DE0-Nano; 2013. p. 1–154.