FPGA implementation of predictive control

Emna Aridhi, Mehdi Abbas, Abdelkader Mami

► To cite this version:

Emna Aridhi, Mehdi Abbas, Abdelkader Mami. FPGA implementation of predictive control. 16th IEEE Mediterranean Electrotechnical Conference, Mar 2012, Tunis, Tunisia. 10.1109/MELCON.2012.6196411 . hal-01122797

HAL Id: hal-01122797

<https://hal.science/hal-01122797>

Submitted on 4 Mar 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

FPGA Implementation of Predictive Control

Emna Aridhi, Mehdi Abbes, Abdelkader Mami

Research Unit Analysis and Control Systems Modeling and Control Team-Photovoltaic Power Systems-FST
Campus University 2092 El Manar I.

Tunis, Tunisia.

aridhi_emna@yahoo.fr

mehdiabbes@yahoo.fr

abdelkader.mami@gmail.com

Abstract—In this paper, we propose an implementation of a synthesizable VHDL program of generalized predictive control (GPC) without constraints on a map XC3S700A Xilinx Starter Kit using the Xilinx ISE 10.1 software. The control strategy was applied to a second order state system. The VHDL language was used as a programming tool. Real variables were described with the fixed-point representation to overcome the overflow problems during the computations in the VHDL program. The use of FPGA circuits presents a good choice regarding to the problem of computation time encountered in predictive algorithms. A GPC Matlab program was also implemented in order to make a performance comparison. The simulation results show a good set-point tracking.

I. INTRODUCTION

The increasing complexity of control applications such as predictive algorithms, especially in real-time applications, requires the use of devices respecting real time constraints and minimizing the resources used by the role of optimization. This introduces the use of fast, flexible and optimized integrated circuits such as microcontrollers and FPGAs (Field Programmable Gate Array) circuits. FPGAs are considered more efficient, in the presence of constraints, than the microcontrollers. They are designed to overcome the problems encountered when using microcontrollers. It is due to their density in number of logic gates, their significantly high speed and the possibility of reconfiguration as desired by the user [6, 10, 12, 15].

The predictive control was applied at first to slow systems because of computing time, which requires increasing the sampling period. But with the development of technology, the computing time has become shorter [1].

At first, most implementations of the predictive algorithm, usually written in C, are done on microcontrollers. With technological advances, designers used other languages such as VHDL (Very High Speed Integrated Circuit Hardware Description Language), seen the simplicity and Flexibility of this language in the hardware description of the circuits, and more efficient circuits like FPGA.

Several worldwide studies [7, 3] led to implement the predictive control on FPGA circuits using different methods (writing in Handel C program and using the RS232 port for its implementation on the material target (RC10 card) [3] or using the Matlab HDL-Coder tool to generate synthesizable VHDL code which can be implemented on FPGA). Therefore, most of these methods agree with the use of VHDL code for describing the design to implement. It allows a minimization of logic gates number than the other methods (HDL coder, Handel C...). So, it increases the performance's design and decreases the consumption of logic gates.

In this paper, we present an implementation of a generalized predictive control (GPC) without constraints applied to a second order state system using a synthesizable VHDL fixed point program of the predictive algorithm. The results are compared to these obtained in Matlab.

The choice of fixed-point representation was owing to the ability of this type to present and interpret the not synthesizable real type better than the "std_logic_vector" type. The implementation was realized on a map XC3S700A based on FPGA using Xilinx ISE 10.1 software.

The paper is structured as follows: in Section II, we present a theoretical review of the GPC. Section III explains the FPGA choice and the Xilinx map. Section IV illustrates the implementation adopted method. For Sections V and VI, we present the controlled state system VHDL design, its simulations and results. The last section is devoted to conclude this paper.

II. THEORETICAL REVIEW OF THE GPC

In this section, we adopt the output deviation method [13] to represent the discrete state system, which is depending on the increments control [4].

$$\left\{ \begin{array}{l} \Delta x(k+1) = A.x(k) + B.u(k) \\ \Delta y(k) = C.\Delta x(k) \\ y(k) = y(k-1) + \Delta y(k) \end{array} \right\} \quad (1)$$

With $x(k) \in \mathbb{R}$.

A is the state matrix and B is the input vector [14].

$$A = \begin{pmatrix} -a_1 & 1 & 0 & \cdots \\ -a_2 & 0 & 1 & \cdots \\ \vdots & \vdots & \vdots & \ddots \\ -a_n & 0 & 0 & \cdots \\ -a_n & 0 & 0 & \cdots \end{pmatrix} ; B = \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_n - 2 \\ b_n - 1 \end{pmatrix} \quad (2)$$

C is the output vector, it is n elements.

$$C = (1 \ 0 \ \dots) \quad (3)$$

Δ is the integral action which it cancels the static error.

$$\Delta = 1 - z^{-1} \quad (4)$$

$\Delta x(k)$: The state increments vector at time k.

$\Delta y(k)$: The output increments vector at time k.

$\Delta u(k)$: The control increments vector at time k.

The predicted output along the prediction horizon H_p is written in the following matrix form [4].

$$Y = G \Delta U + F \quad (5)$$

Where

$Y = [y(k+1)/k, \dots, y(k+H_p/k)]^T$: The predicted outputs sequence along the prediction horizon H_p .

$\Delta U = [\Delta u(k), \Delta u(k+1), \dots, \Delta u(k+H_c-1)]^T$: The control increments sequence along the control horizon H_c .

Only the first computed control $\Delta u(k)$ will be assigned to the system to control.

The matrix G has dimension (H_p, H_c) , it is given by (6).

$$G = \begin{pmatrix} C.B & 0 & \cdots & \cdots & 0 \\ C.B + C.A.B & C.B & 0 & \cdots & 0 \\ C.B + C.A.B + C.A^2.B & C.B + C.A.B & C.B & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \sum_{j=1}^{H_p} C.A^{j-1}.B & \cdots & \cdots & \cdots & \sum_{j=1}^{H_p} C.A^{j-1}.B \end{pmatrix} \quad (6)$$

The vector F of H_p elements is written as follows:

$$F = \begin{pmatrix} y(k) \\ y(k) \\ \vdots \\ y(k) \end{pmatrix} \cdot \begin{pmatrix} C.A \\ C.A + C.A^2 \\ C.A + C.A^2 + C.A^3 \\ \vdots \\ \sum_{j=1}^{H_p} C.A^j \end{pmatrix} \Delta x(k) \quad (7)$$

The optimal control sequence is computed using the following quadratic criterion [13] which is given by (8).

$$J = \frac{1}{2} \left(\sum_{j=1}^{H_p} (y(k+j) - y_c(k+j))^2 + \lambda \sum_{j=1}^{H_c} \Delta u(k+j-1)^2 \right) \quad (8)$$

With

y_c is the set point.

y is the predicted output.

Δu is the control increment.

H_p is the prediction horizon.

H_c is the control horizon.

λ is the weighting factor.

It can also be written in matrix form as follows [13]

$$J = \frac{1}{2} [(Y - Y_c)^T (Y - Y_c) + \lambda \Delta U^T \Delta U] \quad (9)$$

Minimizing the equation (9), we obtain the optimal control sequence.

$$\Delta u = (G^T G + \lambda I)^{-1} G^T (Y_c - F) \quad (10)$$

III. CHOICE OF FPGAS AND THE XILINX MAP XC3S700A

This section is devoted to present an overview of the hardware and software development tools.

A. FPGAs

FPGAs (Field Programmable Gate Array) logic circuits are programmed by user, most of them are designed in CMOS technology and they are part of the ASIC family (Application Specific Integrated Circuit).

They are suitable platforms for design implementation of a real system requiring a large volume of logic gates and speed in computation time [6, 10, 12, 15].

They are composed by configurable logic blocks (CLB), flip-flops and programmable interconnections between these blocks.

A simplified diagram of a FPGA is shown in Fig. 1.

Figure 1. Simplified diagram of a target FPGA [12].

B. Choice of FPGA technology

Since the predictive algorithm is characterized by its complex and iterative aspect requiring the use of fast circuits in computing time and dense numbers of logic gates, we have chosen FPGA technology. These technologies are designed to minimize the propagation delay between the implemented logic, and to be configured as desired.

C. Map XC3S700A Starter Kit

This card belongs to the Spartan 3A family with a clock frequency of 50 MHz and a supply voltage of 5V. It is illustrated in Fig. 2.

Figure 2. Map XC3S700A Xilinx Starter Kit [11].

The Xilinx board contains mainly [11]:

- A programmable voltage generator (from 0V to 3.3V) through mobile jumpers.
- An analog capture circuit consists of two linear amplifiers with 8 bits programmable gain, in an LTC 6912-1 integrated circuit and two A / D converters each of 14 bits, in an LTC 1407A-1 integrated circuit and both of them are controlled by the FPGA through the SPI (Serial Peripheral Interface).
- Four D / A converters each of 12 bits grouped together in an LTC 2624 integrated circuit and controlled by the FPGA through the SPI.
- SPI serial Flash PROM provides serial transmission of data between the FPGA and other components such as A / D and D / A.

The Xilinx ISE 10.1 software was used in the designs implementation on FPGA; it executes all the necessary steps hierarchy as shown in Fig. 3.

Figure 3. Steps for implementing a design on FPGA [10].

IV. DEVELOPMENT APPROACH

In this section, we present the procedure to follow to implement the predictive algorithm on FPGA.

The predictive algorithm, which is implemented on FPGA, is written in VHDL with the fixed point type. This has the advantage to effectively represent the real type not synthesizable in VHDL better than the “std_logic_vector” type. A comparison results with Matlab is performed.

A. The fixed-point type

In many cases, tolerance system characteristics and performance of the material target platform don't justify and don't satisfy the computation of certain arithmetic operations. Moreover, they don't limit the overshoot in the results that may be incorrect when their size is chosen badly.

For this, we use the fixed-point representation. We should fix the first position of the decimal point by computing the number of bits (m) corresponding to the integer part. The fractional part (n) is the remaining bits. A number written in fixed point is mentioned in Fig. 4.

Figure 4. Format of the fixed-point representation [2].

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. For example, a fixed point (m=12, n=12) signal can represent a 12 bit vector (this is applied to the matrix A). The m and n length can increase according to the size and the real value of signal to represent.

The functions used to write VHDL fixed-point program are summarized in the following table.

TABLE I. VHDL FIXED POINT FUNCTIONS

VHDL Fixed Point Functions		
Fixing the position of the decimal point	Library and VHDL Packages	Types and Functions
$m = \text{round}(\log_2(A) + 0.5) + 1$ with A is the biggest number to represent in fixed point	IEEE_PROP OSD Fixed_pkg_c. vhd Fixed_float_t ypes_c.vhd	Ufixed (m-1 DOWNT0 - n) Sfixed (m-1 DOWNT0 - n)
		To_ufixed (std_logic_v ector value,m-1,- n) To_sfixed (std_logic_v ector value,m-1,- n) Resize (fixed- point result, m-1,-n) To_slv(fixed point value)

V. IMPLEMENTING A PREDICTIVE CONTROL WITHOUT CONSTRAINTS ON THE MAP XILINX SPARTAN 3A STARTER KIT XC3S700A

In this section, we present the VHDL predictive algorithm design, as well as the simulations and the corresponding RTL patterns.

A. System to control

The system to be controlled is a discrete second order state system [1]:

$$A = \begin{pmatrix} 0.93 & -0.01 \\ 0.04752 & 0.9964 \end{pmatrix}; B = \begin{pmatrix} 1 \\ 0 \end{pmatrix}; C = (-0.01 \quad 3.71) \quad (11)$$

With $H_p = 5$, $H_c = 1$ and $\lambda = 1$.

B. VHDL Programming

The VHDL predictive algorithm design applied to the state system (11) is developed by the Xilinx ISE10.1 software.

Figure 5. VHDL predictive control design.

The design includes:

- The “Set_point”: a vector of five 12 bits vectors.
- The “predictive_command_2nd_order”: the VHDL predictive algorithm block written in fixed point.

The two blocks are synchronized by the same clock “clk”.

The block diagram of the VHDL predictive algorithm block “predictive_command_2nd_order” is illustrated in Fig. 6.

Figure 6. The block diagram of the “predictive_command_2nd_order” VHDL block.

The VHDL predictive control program has four inputs and two outputs:

- Inputs: Clock (frequency 50 MHz), clock enable, reset and the set_point that is a vector of H_p elements where each is of “std_logic_vector (11 downto 0)” type: It is a periodic signal written in VHDL as a state machine.
- Outputs: The control and the output are two “std_logic_vector (11 downto 0)” vectors.

The choice of 12 bits is due to the use of the DAC for displaying the output on the oscilloscope.

The predictive controller program computes the matrices, the system status, the control, the control increment Δu and the output.

The “predictive_command_2nd_order” VHDL block contains the predictive algorithm presented by the following flowchart.

Figure 7. Chart of the predictive algorithm written in the “predictive_command_2nd_order” VHDL block.

$$\text{Where } H = \begin{pmatrix} C.A \\ C.A + C.A^2 \\ C.A + C.A^2 + C.A^3 \\ \vdots \\ \sum_{j=1}^{H_p} C.A^j \end{pmatrix}, L = \lambda I \text{ and } M = (G^T.G + \lambda.I)^{-1}.$$

For the matrix computation:

- It needs to specify new types of vectors for the matrices G (H_p, H_c), H ($H_p, 2$), L (H_c, H_c), M (H_c, H_c) and S ($H_p, 1$) ports: These are vectors where each element is a binary vector (std_logic_vector) whose size depends on the signal in question.
- The product $G^T * G$ is limited to a single term result of the multiplication of $G^T * G$ inside a “for loop” executed H_p time.

C. Signals Types

In the VHDL program, it was used different specific signals types: For the calculation inside architectures, we use the fixed point “Sfixed/ufixed” type or “unsigned/signed” types and we kept all the main program and its sub programs (calculating matrices, state, output, control...) PORTS as “std_logic_vector” or vector of vectors like the set point signal that is composed of five 12 bits vectors. In fact a conversion of the fixed point signals to the “std_logic_vector” type, according to the size of each signal in question, is necessary at the end of each sub program.

VI. SIMULATION PROCEDURE

In this section, we present the simulations of the programs carried out on Matlab and Xilinx ISE 10.1 with Xilinx ISE simulator as well as the interpretation results.

A. Matlab program

The Matlab program is an m-file. It follows the algorithm mentioned in the flowchart (Fig. 7). For the simulation, the set point is varied from 1.5 to 2.5 after 50 iterations. The set point should be multiplied by 1000 to be equal to the one of the VHDL program. It is used in the comparison with the VHDL results.

B. Simulation results

The VHDL program simulation using the ISE simulator is shown in Fig. 8. The set point varies between 1500 and 2500.

Figure 8. VHDL program simulation of the predictive controller by the ISE simulator.

The clock frequency is fixed at 50 MHz (20ns) which is equal to the clock frequency of the XC3S700A Starter Kit.

The output reaches the set point value after 46 iterations and predicts the set change of four iterations in the future.

When the set point becomes {1500,1500,1500,1500,2500}, the output begins to increase to the next set point value “2500” four iterations in the future (soon as the first element of the set signal changes), before that the set point becomes {2500,2500,2500,2500,2500}.

The size of the set point, the output and the control vectors is 12 bits (std_logic_vector(11 downto 0)); it is due to the use of the DAC.

The resulting curves of the Matlab and Xilinx ISE simulations are illustrated in Fig. 9.

Figure 9. Simulation results of the VHDL program with fixed point and Matlab program of the predictive algorithm.

C. Interpretation results

The simulation curves, obtained in both environments Matlab and ISE, show that the output signal predicts the change of the reference signal of H_p-1 iterations in the future thanks to the control signal that detects the changes in the set point before it goes to H_p iterations.

The difference, between curves, is due to the different interpretation of the fixed-point type used in the VHDL program and the floating point used in Matlab program.

In addition, the limited size of the VHDL signals has an effect in the calculation and therefore in the shape of the curves obtained.

The reference type adopted allows to be predicted in the future by the output. Indeed, at each sampling period, the output detects the change in the set point once the first of

five elements changes, while the other four have not changed. This method is used in VHDL, because there is no concept of future. It is a hardware language.

In case we fix the reference vector size in the VHDL program, the simulation will be, then, limited. Similarly, the implementation result will not be well visualized because the execution is very fast (ns). Except if the clock frequency of the FPGA board will be decreased.

D. RTL patterns of the predictive algorithm VHDL design

During the design synthesis, the XST (Xilinx Synthesis Technology) generates RTL patterns presented in Fig. 10 and the corresponding statistical report of the compensated hardware resources.

For each VHDL block like for example the output block, the XST generates its corresponding RTL scheme.

Figure 10. RTL patterns of the predictive algorithm.

It is noticed that the program didn't use more logic gates to achieve a good performance in computation time.

To complete the implementation process, Xilinx ISE executes the translation, the map and routing processes and configures the FPGA target by downloading the bit stream, by making on the card and by connecting the JTAG cable to the computer.

VII. CONCLUSION

An implementation of predictive control without constraints on a map XC3S700A, FPGA-based, is performed by writing a synthesizable VHDL fixed-point program. We compare VHDL signals to these obtained in Matlab. The use of fixed-point type provides good results because it solves

the overflow problems during the computations. This kind of FPGA devices is very convenient to implement predictive controls because of their important ability to compute complex algorithms in very short time. Besides, it is easy to program, configure and manipulate them through powerful and optimized software as Xilinx ISE.

In future works, we will try to make a comparison between the VHDL program, already made, and the Matlab HDL Coder, to control a real system and after that to implement a predictive control with constraints.

REFERENCES

- [1] A. Kheriji Abbes, F. Bouani and Mr. Ksouri. "A Microcontroller Implementation of Model Predictive Control". ICCESSE International Conference on Computer Electrical and Systems Sciences and Engineering, Paris, France, June 2011.
- [2] E. Bishop. "Fixed point package user's guide", 2008.
- [3] Leonidas G. Bleris, Mayuresh V. Kothare, Jesus Garcia and Mark G. Arnold. Embedded Model Predictive Control for System-on-a-Chip Application. Electrical and Computer Engineering Department, Chemical Engineering Department, Computer Science and Engineering Department. Lehigh University, Bethlehem, PA 18015, U.S.A. 7th International Symposium on Dynamics and Control of Process Systems, Boston, MA, July 2004.
- [4] W. Boualleg. "Commande prédictive des systèmes mono variables". Master thesis, Enit, 2008.
- [5] E. Granado Migliore. "Commande prédictive à base de programmation semi définie". National Institute of Applied Sciences in Toulouse, thesis in 2004.
- [6] S. Kilts. "Advanced FPGA Design Architecture, Implementation and optimization". Spectrum Design Solutions, Minneapolis, Minnesota, 2007
- [7] Lee, J.H, M. Morari and C.E. Garcia. "State-Space Interpretation of Model Predictive Control". Automatica, 1994.
- [8] K.V. Ling, S.P. Yue and J.M. Maciejowski. "Embedded Model Predictive Control (MPC) using a FPGA". Paper presented in the 17th World Congress The International Federation of Automatic Control Seoul, Korea, July 6-11.
- [9] S.Olaru. Contribution à l'étude de la commande prédictive sous contraintes par approche géométrique, Thesis at the University Paris XI Orsay in September 2005.
- [10] S.Renaud, P.Nouel et L.Bossuet. Conception des circuits numériques Language VHDL pour la synthèse et la modélisation, Courses ENSEIRB MATMECA Bordeaux, 2010.
- [11] Spartan-3A/3AN FPGA Starter Kit Board User Guide, Xilinx, 2008.
- [12] A.Tisserand et INRIA LIP Arénnaire. Introduction aux circuits FPGA, Laboratoire LIP ENS Lyon, December 2003.
- [13] K.Watanabet, K.Ikeda, T.Fukuda, S.G.Tzafestas. "Adaptative Generalized Predictive Control using A State-Space Approach", IEEE/RSJ International Workshop on Intelligent Robots and Systems IROS'91, p. 1609-1614, November 3-5, 1991, Osaka Japan. IEEE Cat. No.91TH0375-6.
- [14] Xilinx. Spartan 3 FPGA Family: Complete Datasheet, January 2005.
- [15] B. Zeidman: President Chalkboard Network. Introduction to FPGA Design and CLPD.