3rd International Conference on Automation, Control, Engineering and Computer Science (ACECS'16)
Proceedings of Engineering & Technology (PET)
pp. 540-546

FPGA Implementation of the Model Predictive Control: Application to a control system of water level

TELMOUDI BRINI Sirine[1], BOUZOUITA Badreddine[2] , BOUANI Faouzi[1]

[1] Tunis El Manar University, National Engineering School of Tunis, Tunisia
[2] University of Sousse, National Engineering School of Sousse, Tunisia
LR11ES20 Laboratory of Analysis Conception and Control of Systems, Tunis, Tunisia.

Sirinebrini@gmail.com
badreddine.bouzouita@enit.rnu.tn
Faouzi.bouani@enit.rnu.tn

***Abstract*—The design problem of Model Predictive Controller (MPC) needs to solve a computationally burden Quadratic Programming (QP) problem at each sample time. This paper provides an implementation of MPC algorithm into reconfigurable hardware such as a FPGA chip with a 20 MHz fixed clock. This implementation of MPC achieves comparable results to those obtained with an AMD Turiom (Tm) II Dual core 2.2 GHz PC which offers a promising platform to expand the field of application of this technique to control embedded systems. The efficiency of the proposed approach is demonstrated with a real time control of water level of a single tank system running on Nanoboard 3000XN FPGA platform.**

***Keywords*— FPGA, predictive control, quadratic optimization problem.**

I. INTRODUCTION

Predictive control is a theory that took advantage to exploit systems embarked power's on the control processes and also optimization algorithms evolution's which became faster. Indeed, this technique of control requires optimization, every sampling period's, a function cost to obtain the control law. For its capability of handling constraints intrinsically, MPC has been widely applied as an advanced control strategy to much complex industrial process such as energy renewable [10], power electronics [11] and road vehicles [6].

The progress integration process has led increasing number of transistors integrated on a chip. This is origin of vision systems on chip or single-chip systems commonly called SoC (System on Chip), which are inserted on same chip of different components, such as processors, specialized circuits (ASIC: Application Specific Integrated Circuit) memory banks and reconfigurable circuits (FPGA: Field Programmable Gate Array). Moreover, with its architecture with features such as pipelining and parallel computing, the FPGA can reach a processing speed much higher than software implementations [12][13]. This makes them particularly attractive for FPGA intensive computing tasks and thanks to technological advances in micro- electronics field's[9], searchers continues to grow and evolve with the new hardware design solutions, such as FPGAs or ASICs can be used as targets for control algorithms digital implementation's.

Predictive algorithms implementation's on FPGA attracted the interests of several researchers in these last year's such as: Bleris, Vouzis, Arnold, Kothare; He, KV.Ling; Ling Yue, Maciejowski, Monmasson and Cirstea. Monmasson and Cristea [5]. In fact, they gave an overview and some general directives onto the development of industrial control systems over FPGA. Bleris [2] used a coprocessor to accelerate the major part of the calculations in a sequential order. KV.Ling and He [3][7], developed a procedure by using co-conception tool's (co-design) MATLAB / Handel-C for fast prototyping and to implement MPC on FPGA, where an internal method's points is used to solve the on-line optimization. Lau, Yue, Ling, and Maciejowski in 2009 [8] compared the active set method with an internal point by its implementation on FPGA with the procedures Handel-C, and reflected that the active method is more effective for the problems of small scale.

This paper is a contribution to this latter line of research which is focused on the implementation of MPC into FPGA. It develops the integration hardware and software design model to implement an MPC algorithm based on ARIMA (Auto Regressive Integrated Moving Average) model on FPGA platforms for field controls. To make the design more flexible for different application we propose as technique of optimization the quadratic programming by the gradient method.

To demonstrate the efficacy of the proposed design, an example MPC system using the Nanoboard 3000XN device is applied to a control system of water level and achieves good control performance.

The remainder of the paper is organized as follows: section II presents theoretical overview of GPC. Section III gives the details of MPC algorithm implementation on FPGA and application to water control system. Section IV showcases experimental results. The conclusion is provided in section V.

II. THEORICAL REVIEW OF GPC

The words "Predictive Control" don't make reference to a specific technique of control but rather to a whole range of

ISSN: 2356-5608

strategies of control which are developed around certain common ideas which are the following ones.

A. Output prediction

All predictive control strategies require a numerical model in order to predict the future behavior of the process. In this paper, the control design is based on ARIMA model. This controller is known also as Generalized Predictive Control (GPC) because it can deal with open loop unstable system, no minimum-phase plant, and system with delay.

In this case, the output predictions are given by the following representation:

$$A\left(q^{-1}\right)\Delta y(k) = q^{-d}B\left(q^{-1}\right)\Delta u(k) \quad (1)$$

where $y(k)$ is the output system, $u(k)$ is the input system and d is the time delay. The term, $\Delta = 1 - q^{-1}$ is the integral action that allows the cancellation of the static error. $A(q^{-1})$ and $B(q^{-1})$ are polynomials of degrees respectively na and nb in backward shift operator q^{-1} :

$$A\left(q^{-1}\right) = 1 + \sum_{i=1}^{na} a_i q^{-i} \quad (2)$$

$$B(q^{-1}) = \sum_{i=1}^{nb} b_i q^{-i} \quad (3)$$

So, the output prediction $y(k + j)$ can be obtained using multiple recursion as follows:

$$\hat{y}(k+j/k) = \sum_{i=1}^{j} g_{j-i+1}\Delta u(k+i-1) + y_l(k+j) \quad \text{for j=1,…, } H_p \quad (4)$$

with:

$$\begin{cases} g_1 = b_1 \\ g_j = b_j + \sum_{i=1}^{j-1}(a_{i-1} - a_i)g_{j-i} \end{cases} \quad (5)$$

and:

$$\begin{cases} y_l(k+1) = y(k) - \sum_{i=1}^{na} a_i \Delta y(k+1-i) + \sum_{i=2}^{nb} b_i \Delta u(k+1-i) \\ y_l(k+j) = a_{j-1}y(k) + \sum_{i=j+1}^{nb} b_i \Delta u(k+j-i) + \sum_{i=1}^{min(j-1,na)} (a_{i-1} - a_i)y_l(k+j-i) \end{cases} \quad (6)$$

Consequently, we can put the relation (4) in the following matrix form:

$$Y = G\Delta U + Y_l \quad (7)$$

Such as:

Y : Vector outputs predicted on the prediction horizon Hp:

$$Y = \left[\hat{y}(k+1/k) \quad \hat{y}(k+2/k) \quad \dots \quad \hat{y}(k+H_p/k)\right]^T \quad (8)$$

ΔU : Vector increments future optimized controls:

$$\Delta U = \left[\Delta u(k) \quad \Delta u(k+1) \quad \dots \quad \Delta u(k+H_c-1)\right]^T \quad (9)$$

Y_l : Vector free responses due to increments of old orders:

$$Y_l = \left[y_l(k+1) \quad y_l(k+2) \quad \dots \quad y_l(k+H_p)\right]^T \quad (10)$$

Then the matrix G is distinguished as follows:

$$G = \begin{bmatrix} g_1 & & \\ g_2 & g_1 & \\ \vdots & & \\ g_{H_p} & \cdots & g_{H_p - H_c + 1} \end{bmatrix}; \; dimG = (H_p, H_c) \quad (11)$$

B. Control law

To indicate how well the process follows the desired trajectory, a cost function is formulated. This last minimizes the future errors between output signals and set-points, and the future increment control signals. Using the quadratic norm, the cost function is given by:

$$J_1 = \sum_{j=1}^{H_p} \left[y_c(k+j) - \hat{y}(k+j/k)\right]^2 + \lambda \sum_{j=0}^{H_c-1} \Delta u(k+j/k)^2 \quad (12)$$

with:

$\Delta u(k+j/k) = 0$ for $j \geq H_c$.

$\hat{y}(k+j/k)$: Predicted output.

$y_c(k+j)$: Set point.

λ : Weighting on the future increment control.

Δ : Incremental operator.

So the control law is obtained by the minimizing at each sample time of the following optimization problem:

$$\min_{\Delta U} J_1(\Delta U) \quad (13)$$

Using gradient method, the optimization problem (13), is reduced to solve the following equation:

$$\Delta J_1 = \frac{\partial J_1}{\partial \Delta U} = 2(G^T G + \lambda I_{H_c})\Delta U + 2G^T(Y_l - Yc) = 0 \quad (14)$$

Therefore, we obtain the following optimal solution:

$$\Delta U_{opt} = \left[G^T G + \lambda I_{H_c}\right]^{-1} G^T (Y_c - Y_l) \quad (15)$$

Note:

We need only calculate the first row of the matrix $\left[G^T G + \lambda I_{H_c}\right]^{-1}$ because only the first command will be applied to the process thus:

$$u(k) = u(k-1) + \Delta u(k) \quad (16)$$

where: $\Delta u(k) = \Delta U_{opt}(1)$

III. Applying the predictive control on a water system implemented on the Nanoboard 3000XN

In this section, we present the procedure to implement predictive algorithm on the Map Nanoboard 3000XN then it will applied on a control system of water level.

A. Nanoboard 3000XN

The Nanoboard 3000XN map with fixed Xilinx Spartan-3AN device (XC3S1400AN-4FGG676C) is crucial for the fast development of the systems embarked with Altium designer. It

is a platform of design material reprogrammable that exploits the power of one dedicated to electronic conception of high capacity, allowing the fast implementation, interactive and the debugging of our conceptions [4]. The map is constituted by several material modules which realize miscellaneous tasks according to designer's needs, quoting: Integrated color TFT LCD panel (240x320),SVGA interface (24-bit, 80MHz), Programmable clock (6 to 200MHz) and fixed clock (20MHz) – both available to FPGA user, 4-channel 8-bit ADC and 4-channel 8-bit DAC.

B. Single Tank System:

The process presented by Fig.1 consists of a rectangular tank which empties into a principal tank. We index these reservoirs respectively by 1 and 0. A pump allows the water supply of tank 1 by adjustment of the control input. We find also a power module which contains a power amplifier unit for the pump, a conditioning block of the sensor derived signals levels, flow and pressure.

➢ Estimation of system parameters:

First of all, we must know that general structure of process requires identification of order and system delay. From the step responses with a sampling period (Te) equals to 2s, allow to conclude that the process behavior can be characterized by a first order model:

$$y(k) = -a_1 y(k-1) + b_1 u(k-1) \tag{17}$$

where $y(k)$ is the level in the tank and $u(k)$ is the input control.

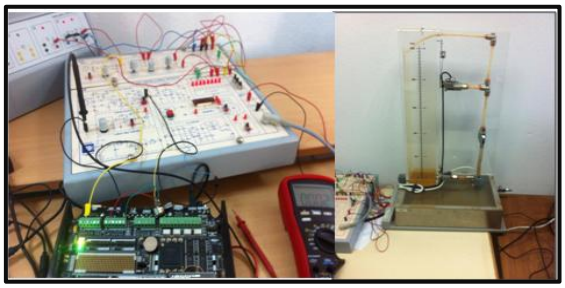

Fig. 1 System of water level

In order to determine the parameters of the model, we have exited the system with square-wave input. The response of the system is given by Fig.2, from it we note that the error between the estimated output and the actual output is somewhat remarkable and this is evident because of the nonlinearity of our process.

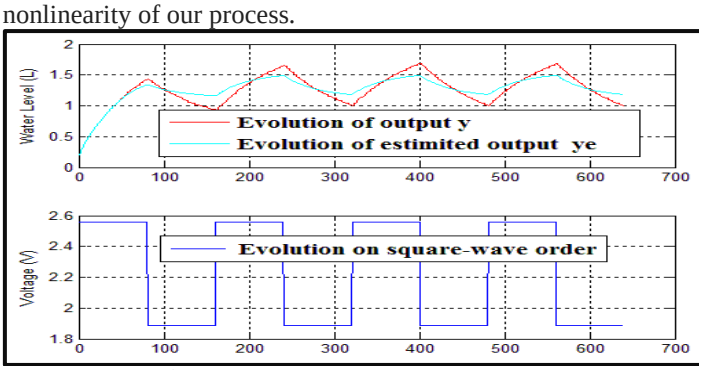

Fig. 2 Evolution of measured output, estimated output and square wave order

To estimate offline the parameters of the system, we opted for the recursive least squares method with fixed forgetting factor. The algorithm to be applied is the following:

$$\begin{aligned} e(k) &= y(k) - \hat{\theta}^T(k-1)\phi(k) \\ \hat{\theta}(k) &= \hat{\theta}(k-1) + P(k)\phi(k)e(k) \\ P(k) &= P(k-1) - \frac{P(k-1)\phi^T(k)P(k-1)}{1+\phi^T P(k-1)f(k)} \end{aligned} \tag{18}$$

With:

$\hat{\theta}(k) = [a_1, b_1]^T$: Vector of estimated parameters.

$e(k)$: Error of prediction in priori.

$\phi = [-y(k-1), u(k-1)]^T$: Vector of measures.

Then, the obtained model parameters are:

$$M : \begin{cases} a_1 = -0.97491 \\ b_1 = 0.033523 \end{cases} \tag{19}$$

The Fig.3 shows evolution of the model estimated parameters.

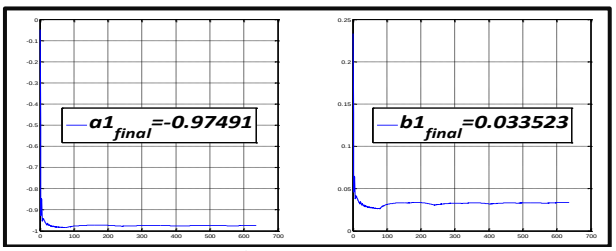

Fig.3 Evolution of the model estimated parameters.

C. MPC Design:

The design of the custom hardware solution is shown in Fig. 4. The predictive algorithm is implemented in C++ and compiled for a Nanoboard 3000XN XC3S1400AN-4FGG676C FPGA, which contains an A/D, D/A board to measure responses and output control actions and a TFT screen to show evolution of the output and control action.

- The first step of prediction process is: of a simple cycle t the FPGA receives output data y_t of the system from the analog to digital convertor A/D.
- In the next step the state observer prepare the data calculating the free responses y_l to feed into the QP solver.
- Then the QP is solved to gives a new control action U_{op}. This control action is sent to the Digital to Analog D/A circuit, which is loaded with the new value at the beginning of the next sampling cycle.
- The LCD screen is used to display evolution of output and control action.

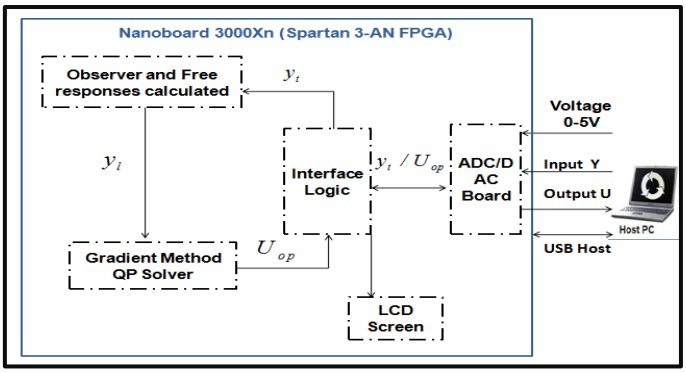

Fig. 4 MPC circuit design

D. Hard architecture with Altium designer

To elaborate the design shown in Fig. 4 with Altium designer we need to create a file called "openbus" presented Fig. 5 which allows the designer to build a system Processor-based with more rational and abstract manner. We used: a processor TSK3000 to assure the execution of software application which is a 32- bit RISC processor. Most of his instructions are 32 bit width and running in a single clock cycle. The latter was designed to simplify the development of 32 -bit systems targeted for FPGA implementation. It has greatly simplified memory structure interrupt handling to make the code simpler and also simplifies the connection of devices with support for wishbone bus. Moreover, a memory SRAM to store the data, a controller TFT_VGA to pilot the display of signals on LCD screen, tow converters ADC/DAC (analog to digital convertor/digital to analog convertors) to convert input/output data which are piloted by the SPI master controller, a soft terminal to retrieve the data in a text file such us output data and order data, the port IO for the input/output and two wishbone interconnect and arbiter.

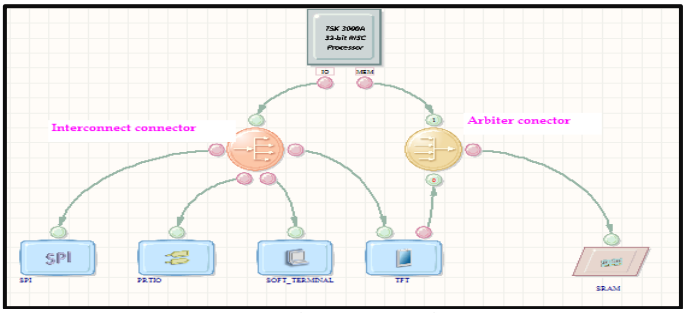

Fig. 5 Openbus System for the design of predictive control

E. Matrix operation :

Subtraction, multiplication, addition and inversion are the basic matrix operations of our algorithm, where matrices sizes are arbitrary. By taking into account limitation of FPGA resources (absence of matrix library), so we have to create the algorithms that calculate these operations especially inverse matrix because of its complexity.

To calculate the optimal solution defined by equation (15), the following matrixes must be calculated.

$$G[H_P][H_c],$$

$$M[H_c][H_c] = G^T G + \lambda I_{H_c},$$

$$Inv\_M[H_c][H_c] = M^{-1}$$

$$F[H_c][H_p] = Inv\_M.G^T$$

They are constants which can be obtained offline. Inversion matrix is obtained by using the Gaussian method [1].

F. Software Architecture with Altium designer: Predictive Algorithm

The selection of library is very important in the design of the application because the lack of one library can cause many errors at compilation. So choosing libraries should be well studied and the user must understand the value of each component before using it [9].

For our application we used the LCD module and the converters ADC/DAC of the card Nanoboard3000XN and our need requires the functions name of each libraries provides by the map shown in Table 1.

TABLE I
FUNCTIONS NAME OF EACH LIBRARIES USED IN THE PREDICTIVE ALGORITHM

Function Name	Role of peripherals
Grapigcs_open	Initialization of LCD screen
graphics_get_visible_canvas	Get screen active
graphics_fill_canvas	Give a color to the background of screen
adc084s021_open	Initialization of module ADC
dac084s085_open	Initialization of module DAC

In order to use the GPC controller algorithm, the steps illustrated in the flowchart, Fig. 6 must be followed.

➢ *Main code :*

Our code explains the predictive controller algorithm in many steps, citing:

- Calculating coefficient matrix of system response:

After declaration of system parameters, specific parameters, prediction horizon, control horizon, weighting factor and desired set point, matrix coefficients given by equation (4) will be calculated as follows on algorithm 1. This matrix is calculated offline and it's constant if we don't change any settings of systems.

Algorithm 1: calculate of matrix coefficients

1: Determination of vectors size **A** and **B** respectively **Sizea** " and " **sizeb** ".
2: Allocation of the first matrix element **G** to the first Vector element **B**: $G[0][0] = B[0]$.

3: Calculates of matrix G:
For k=2: H_p **do**
4: s=0.
5: **For** i=2: minimum (sizea,k) **do**
6: $S \leftarrow s + (a[i-2] - a[i-1]) * G[k-i][0]$.
7: End for
8: If k<=sizeb **then**
9: $G[K-1][0] = b[k-1] + s$.
10: Else
11: $G[k-1][0] = s$.
12: End If.
13: for j=2: minimum(H_c , k) **do**
14: $G[k][j] = G[k-1][j-1]$.
15: End For.
16: End For.

- Calculating Matrix $F = \left[G^T G + \lambda I_{H_C} \right]^{-1} G^T$:

Noting that only the first control increment of future control vector $\{u(k)\}$ will be applied to the system and the rest will be rejected because in the next moment the new output $\{y(k+1)\}$ is available that's why we only calculates the first line of matrix F . The calculation of this matrix is offline while it's constant and it's the most difficult to calculate from matrix operations presented in our code while it includes the inverse operation of $\left[G^T G + \lambda I_{H_c} \right]$. We used the Gauss algorithm to calculate inverse [1].

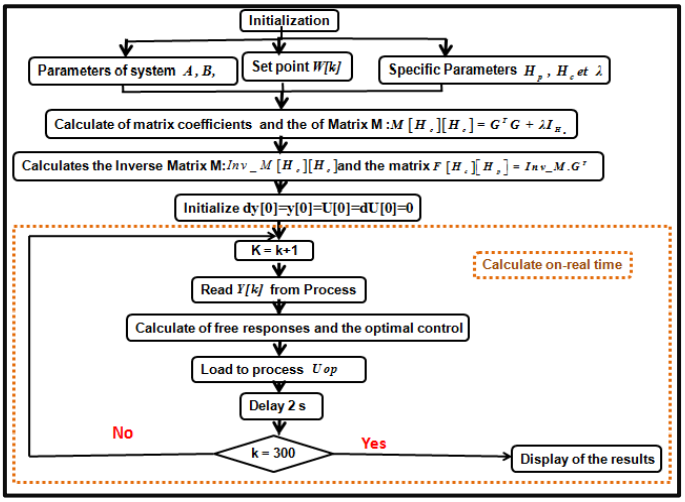

Fig. 6 Predictive algorithm

Also for implementation and for applying the predictive algorithm to the real process the total time for 300 iterations to calculate the optimal control is equal 10.887 min for a 2s sampling period.

IV. EXPERIMENTAL RESULTS

In this section, the proposed FPGA based MPC design is applied to a control system of water level.

The system to be controlled is a first order state system presented by the equation:

$$\frac{y(k)}{u(k)} = \frac{0.033523q^{-1}}{(1-0.97491q^{-1})}$$

A MPC controller was designed with a sampling interval of 2s. The following constraints:

$$0v \le U \le 5v$$

were also included.

The synthesis results of the MPC algorithm implementation on FPGA are very important for the conception and to know the occupation rate concerning the internal resources of the FPGA, such as the number of slices, clock etc. The hardware resources used to implement the MPC algorithm on a FPGA chips is shown in Table 2. About 50% of RAM on the FPGA chip was used and this value can be considered lightly lower compared to the large number of mathematical operations of the predictive algorithm, also 21% of Look-Up-Table (LUT) with 4 inputs are used. This is important for envisaging other more complex treatments.

TABLE II
SUMMARIZING MATERIAL RESOURCES USED FOR THE IMPLEMENTATION OF PREDICTIVE ALGORITHM FOR $H_p = 8, H_c = \lambda = 1$.

Used Logic	Used	Available	% of use
Number of Slice Flip Flops	2262	22528	10%
Number of slice	3074	11264	27%
Number of LUTs (4 inputs)	4942	22528	21%
Total LUTS (4 inputs)	5529	22528	23%
Number of block of inputs/outputs	145	502	28%
Number of BUFGMUXs	2	24	8%
Number of MULT18X18SIOs	2	32	6%
Number ofRAMB16BWEs	16	32	50%

Fig. 7, Fig. 8 and Fig.9 depict the evolution of water level Y and the input control U for different value of prediction horizon Hp (Hp = 3, Hp = 8 and Hp = 10). These results show that the water level reaches the retained set-point. We can see that the output response for Hp = 3 presents many fluctuations. For Hp equals to 8 (Fig. 8) , the closed loop response is more smooth and presents an overshoot. When we increase the Hp to 10, we note the disappearance of overshoot and the output response is also smooth.

In the second experiment, we aim to test the influence of the weighting control parameter λ. In Fig. 10, we decrease λ to 0.5 and for figure 13 the parameter λ is equal to 0.01. The prediction horizon and control horizon remain constant (Hp = 8 and Hc = 1).

When we compare the input signals depicted in Fig. 8, Fig. 9 and Fig. 11, it can be seen that the FPGA-GPC controller produces a better performance. In fact, the control law shows many fluctuation when we decrease the weighting parameter λ In these experiments, the average time required to compute the control input U for each sample time is 177,4 ms. Therefore,

we can conclude that the implementation of GPC on FPGA presents an alternative to control fast systems.

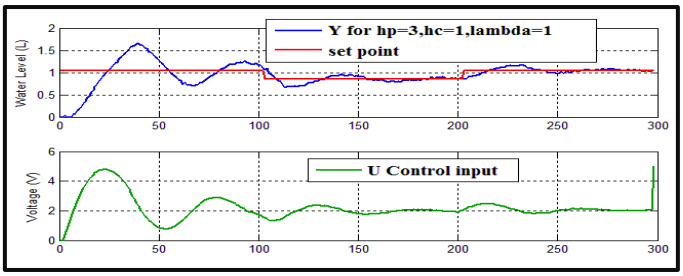

Fig. 7 Evolution of output and control Input for $H_p = 3, H_C = \lambda = 1$

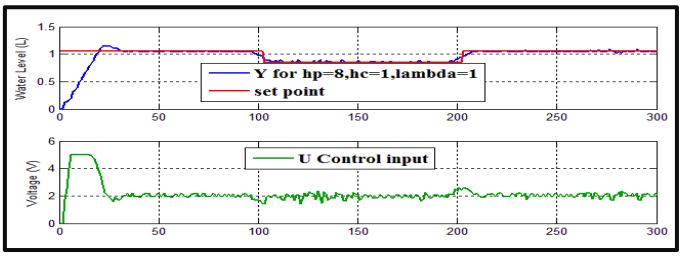

Fig. 8 Evolution of output and control Input for $H_p = 8, H_c = \lambda = 1$

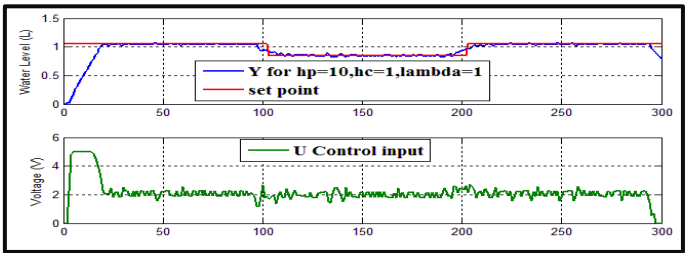

Fig. 9 Evolution of output and control Input for $H_p = 10, H_c = \lambda = 1$

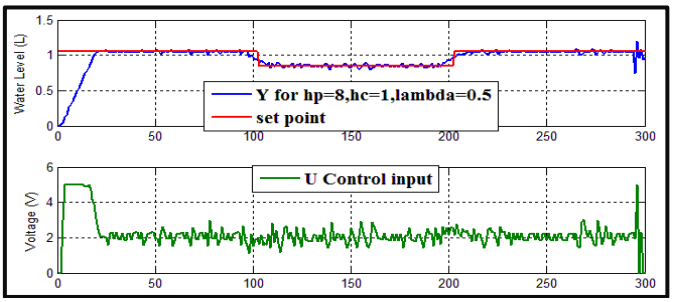

Fig. 10 Evolution of output and control Input for $H_p = 8, H_c = 1, \lambda = 0.5$

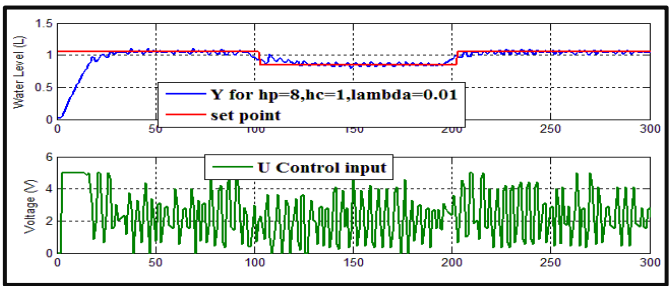

Fig.11 Evolution of output and control Input for $H_p = 8, H_c = 1, \lambda = 0.01$

V. CONCLUSION

This paper has proposed an integrated design of both hardware and software implementation of MPC on FPGA platforms. The MPC algorithm is based on ARIMA model using as optimization technique the quadratic programming (QP) based on the gradient method to obtain the control at each sample time. The performance of the MPC controller is tested on a water level system using Nanoboard 3000XN map with fixed Xilinx Spartan-3AN device. In this practical application, we have studied also the influence of the control law parameters. From the experimental results, we note that the implementation of MPC achieves comparable results to those obtained with an AMD Turiom (Tm) II Dual core 2.2 GHz PC which offers a promising platform to expand the field of application of this technique to control embedded systems, as well as the increase of prediction horizon Hp and of weighting factor give better performance in terms of fluctuation of the control action.

REFERENCES

[1] D.Pastre, *" Calculating the inverse of a matrix: method Gauss-Jordan"*, University René Descartes Faculty of Mathematics and Computer,Lesson of Numerical methods, chapter 2, 2004.

[2] G.Bleris, D.ouzis, G.Arnold, V.Kothare, "*A Co-Processor Platform for the Implementation of Control"*, American Control Conference 2006 (ACC06), Minneapolis, Minnesota, 14-16 June 2006.

[3] K.V.Ling, S.P.Yue, J.M.Macie Jowski, "*A FPGA implementation of model control",*American Control Conference Minneapolis, Minnesota, USA, June 2006.

[4] M.Ahmed, Y.Kong, "*Digital Traffic Controller on the Altium NanoBoard"*, IEEE Conference Publication International Symposium on Communications and Information Technologies (ISCIT), Gold Coast, QLD ,5 october 2012.

[5] M.Cirstea, E.Monmasson, "*FPGA Design Methodology for Industrial control Systems*", IEEE Transactions on Indistrial Electronics, Vol. 54, no. 4, pp. 1824-2842, Août 2007.

[6] M.Morari, M.Boatic, F.Borrelli, "*Hybrid Systems Modeling and Control"*, European Journal of Control, Vol.9, No 2-3, pp.177-189, 2003.

[7] M.H.He, KV.Ling, *"Model Predictive Contol on a Chip"*, The 5th International Conference on Control & Automation, Budapest, Hungary,June 26-29, 2005.

[8] M.S.K.Lau, S.P.Yue ,K.V.Ling, J.M.Maciejowski, *"A comparison of interior point and active set methods for FPGA implementation of model predective control*", Proc.European Control Conference, Budapest, August 2009.

[9] M.Vladislav, Z.Kamil, "*Design of high perfermance multimedia control system for UAV/UGV based on SOC/FPGA core*", Procedia Engineering 48, pp.402-408, Slovakia, Mars 2013.

[10] P.F.Odgaard, L.F.S.Larsen, R.Wisniewski,T.G.Hoygaard, *"on using Pareto optimality to tune a linear model predictive controller for wind turbines"*, Renewble Energy , vol 87, part 2, pp.884–891, October 2015.

[11] *P.Karamanakos,T.Gayer,S.Manias, "Direct Voltage Control Of DC-Dc Boost Convertes Using Enumeration-Based Model Predictive Control"*, IEEE Transactions on Power Electronics , Vol 29 , Issue: 2, pp. 968 – 978, Febrary 2014.

[12] Y.Nan,L.D.Lei, Z.Jun, *"Model predictive controller design and implementation on FPGA with application to motor servo system"*, Control Enginneering Practice, pp.1229-1235, 2012.

[13] W. Burleson, R. Tessier, *"Dynamically Parameterized Algorithms and Architectures to Exploit Signal Variations for Improved Performance and Reduced Power"*, in the Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, (ICASSP'01). Salt Lake City, Utah, May 2001.