FPGA MODELLING AND REAL-TIME EMBEDDED CONTROL DESIGN VIA LABVIEW SOFTWARE: APPLICATION FOR SWINGING-UP A PENDULUM

Wael Benrejeb, Olfa Boubaker

National Institute of Applied Sciences and Technology
Centre Urbain Nord BP 676, 1080 Tunis Cedex, Tunisia
Emails: olfa.boubaker@insat.rnu.tn

Submitted: May 15, 2012                 Accepted: July 20, 2012                Published: Sep. 1, 2012

Abstract- In this paper, Real-Time embedded control is designed via LabVIEW software for swinging-up a pendulum from its pending position to its upright position. Since the pendulum system has a typical nonlinear instable model, the control problem is achieved using the Astrom-Furuta energy control strategy. To overcome the complexities for the design and the real-Time implementation of the controller of the nonlinear system, FPGA and Real-Time Modules of LabVIEW software are used. A validation test is finally achieved using Proteus software via its Virtual Simulation Models (VSM). Simulation results show the capabilities of LabVIEW FPGA and Real-Time modules to customize control applications with flexible time control without VHDL coding or board design.

Index terms: Field programmable gate arrays, Programmable control, Software design, Inverted pendulum.
I. INTRODUCTION

For at least fifty years, the pendulum has been the most popular benchmark, among others, for teaching and researches in Control Theory and Mechatronics [1]. Different versions of the pendulum benchmark exist offering a variety of interesting control challenges. The most familiar types are the rotational single-arm pendulum [2], the double arm pendulum [3] and the cart inverted pendulum [4]. The principal control task considered in many research works is the swinging up from the stable equilibrium point to the unstable equilibrium [2, 4, 5, 6, 7, 8]. Though the structure of the inverted pendulum seems to be quite simple, the upswing control is much more difficult than it appears since the pendulum is a typically non linear system and many conventional techniques control are ineffective. Several real experimental models of the pendulum benchmark are performed [5, 9]. In the most cases, integrated software’s are used to support the design and the real-time implementation of controllers [10, 11].

On the other side, National Instruments® (NI) LabVIEW™ is considered, nowadays, as the most professional integrated software/Hardware platform [12]. Furthermore, more than 25 LabVIEW modules and toolkits are added to this software for specific applications [13]. In this framework, FPGA Module [14, 15] allow development of FPGA programs on NI reconfigurable Input/output RIO hardware targets for custom applications and control algorithms with loop rates up to 40 MHz without VHDL coding or board design. It executes multiple tasks simultaneously. In the same context, Real-Time Module [16, 17] allows designing, prototyping and deploying of Real-Time Controllers. Furthermore, to shorten time of designing and deploying it is also possible to take the advantage of the NI embedded platforms [18, 19]. Using LabVIEW graphical programming, we can then develop customized control applications on a desktop machine [20], and then download and execute the program to run on an independent NI hardware target.

In this paper, FPGA based real time control of an inverted pendulum using LabVIEW Software is investigated. The control objective is to swing up the pendulum from its instable equilibrium position to its stable position using the well known Astrom-Furuta Swing-up control strategy. FPGA and Real-Time modules of LabVIEW software will be used to design the nonlinear model of the inverted pendulum and to implement its switching control law.
II. PROBLEM FORMULATION

Consider the pendulum system shown by Fig. 1. The control problem considered is the swinging up the pendulum from the pending position to the upright position. The swing up of the pendulum is archived by a DC motor.

To find out the position and speed of the DC motor we will choose a square encoder with two outputs track to indicate the position and direction of rotation. The two square wave outputs (A and B) have the same number of pulses per revolution and are in quadrature relation to each other: A leads B by 90° for clockwise rotation and B leads A by 90° for counter-clockwise rotation. The pulse outputs indicate amount of shaft rotation, and the A / B phase difference indicates direction of rotation.

Let now denote, the angle between the vertical and the pendulum by $\theta$ (see Fig.2), which is positive in the clock-wise direction. $m$, $J$ and $l$ are the mass of the pendulum, the moment of inertia with respect to the pivot point and the distance from the pivot to the center of mass, respectively.
The acceleration of gravity is $g$ and the acceleration of the pivot is $u = ng$. Under some predefined assumptions, the equations of motion of the pendulum are described by [2]:

$$J \ddot{\theta} = mg l \sin \theta - m l \cos \theta$$  \hspace{1cm} (1)$$

The energy of the system can be deduced as:

$$E = 0.5 J_p \dot{\theta}^2 + m g l (\cos \theta - 1)$$  \hspace{1cm} (2)$$

where:

$$J_p = ml^2 + J$$  \hspace{1cm} (3)$$

Let $\omega_0 = m g l / J$. The normalized energy is deduced as:

$$E = m g l \left[ 0.5 \left( \frac{\dot{\theta}}{\omega_0} \right)^2 + \cos \theta - 1 \right]$$  \hspace{1cm} (4)$$

and the normalized state space system is described by:

$$\begin{align*}
\dot{x}_1 &= \dot{\theta} \\
\dot{x}_2 &= (\sin \theta - v \cos \theta) \omega_0
\end{align*}$$  \hspace{1cm} (5)$$
where \( x = [x_1, x_2]^T = [\theta, \dot{\theta}]^T \) is the state vector and such that:

\[
u = v \theta
\]  

(6)

where \( u \) is the control law described by the following form:

\[
u = \text{sign}_{ng} \left( k(E - E_0) \text{sign} \left( \dot{\theta} \cos \theta \right) \right)
\]  

(7)

\( \text{sign}_{ng} \) denotes a function which saturates at \( ng \) and \( E_0 \) the desired energy allowing the pendulum to reach the upright position. This strategy is essentially a bang-bang strategy for large errors and a proportional control for small errors. For small values of \( n \) the relation between \( n \) and \( k \) is approximately given by [2]:

\[
k \approx \frac{\pi}{2n - 1}
\]  

(8)

III. PROBLEM SOLUTION

A. NI LabVIEW Reconfigurable Platform

The NI LabVIEW technology of reconfigurable platform is based on four main parts (see Fig.3):

- A Real-Time processor,
- A reconfigurable FPGA
- An Input / Output module
- A graphical development software.

Combined, these four components provide the ability to quickly create a specific material for high performance and flexible time control.
Based on cost and technological constrains, we choose the Single-Board RIO 9632 for our application (see Fig.4). This device offer 32 Analogical Inputs, 4 Analogical outputs, 400 MHz for the processor Speed, 128 MB Memory (DRAM), 2M FPGA Size (Gates), an RS232 serial port for connection to peripherals and devices and more options.

B. LabVIEW Graphical Software Code
For rapid modeling and control prototyping, we use NI LabVIEW FPGA and Real-Time Modules [19, 21, 22, 23, 24]. FPGA module offer the advantage of graphical development for field-programmable gate array (FPGA) chips on NI reconfigurable hardware with loop rates up to 40 MHz and without VHDL coding or board design. Furthermore, multiple tasks can be simultaneously and deterministically executed and we can implement custom timing and triggering logic with 25 ns resolution.
LabVIEW Real-Time Module offers the possibility to design and deploy embedded Real-Time Controllers. To design and deploy the swing up control of the inverted pendulum, the following tasks are elaborated:

- Acquisition of the signals A & B from the encoder and generation of pulse width modulation (PWM) control signal to the DC motor via the FPGA Module.
- Design of the real-Time embedded control law (7) via Real-Time Module.

The FPGA and Real-Time control application is first developed on a desktop machine using a LabVIEW graphical program, and then downloaded and executed to run on the independent Single-Board RIO 9632 hardware.

Acquisition of the signals A and B from the encoder is achieved using the FPGA Module of LabVIEW software via the LabVIEW structure shown in Fig.5.a. In the previous code, an XOR Gate between the present values and previous values of channels A and B was considered to detect the change in the state. When the previous state of Channel A is the same as the current state of the channel B then the position is incremented, otherwise it is decremented (see Fig.6). A speedometer is also designed in this code (see tick/count variable). The PWM signal is generated and applied to the DC motor to swing up the pendulum via the FPGA module using the LabVIEW code shown in Fig.5.b. The two loops shown in Fig.5 are High-Speed and Parallel Loops.

To avoid using arithmetic operators in the FPGA module that can weigh the processing time, the design of the controller is achieved via Real-Time Module as follows: First, a program of speed computation is designed using the following relationship for the computed angular speed:

\[
\dot{\theta}_{\text{com}} = \frac{n \times 360}{4 \times N \times F}
\]

where:

\(\dot{\theta}_{\text{com}}\) : Computed angular velocity by the Real-Time Module

\(n\) : tick / count number delivered by the FPGA code

\(N\) : Number of pulses per revolution

\(F\) : FPGA frequency
Taking into account that $\min(\theta_{\text{com}}) = 0$, the duty cycle is then computed as:

\[
D = \frac{\text{mean}(\dot{\theta}_{\text{com}})}{\text{max}(\dot{\theta}_{\text{com}})}
\]

In the last LabVIEW project, five shared variables are used: the computed angular position, the computed angular speed, the PWM duty cycle, the normalized energy and the control law.

Fig.5. High-Speed, Parallel Loops in LabVIEW FPGA Module. (a) LabVIEW loop for acquisition of signals A & B from the encoder; (b) LabVIEW loop for generating PWM signal.
Fig. 6. Incrementing and decrementing the angular position signal.

Fig. 7. Computation of angular position and velocity via Real-Time toolkit.

The ordinary differential system (ODE) (5) is then designed using the LabVIEW Math Script Node Virtual Instrument (VI) as represented in Fig. 8.
The normalized Energy (4) and the control law (7) needed to solve the ODE system (5) are computed as shown by Fig. 9. The ODE system (5) is then solved using the ODE VI given in Fig. 10. The derivatives of the state variables are then deduced and used to compute the PWM signal exploited by FPGA code in Fig. 5.b.

Fig. 9. Computation of the normalized energy (4) and the control law (7)

Fig. 10. Solving the ordinary differential system (5)
C. Test Validation via Proteus VSM

In this section we present a test validation of the designed Real-Time control without using the NI Single-Board RIO 9632 device. We take advantage of the new version of Proteus software 7.7 with its Virtual Simulation Modelling (VSM) [25] as a Real-Time simulation tool for modelling the performance of PMW controller. The usage of Proteus VSM enables us shorter product development time and thus reducing development cost [26]. The Proteus VSM computed aided design (CAD) model is shown in Fig.11 where the microcontroller PIC 18F2550, the square encoder and the standard RS-232 are used to convert the angular speed generated by the LabVIEW FPGA Module into a PWM signal in order to rotate the shaft of the DC motor. Result of the test validation are derived for parameter \( k = 1 \). Fig. 12 shows transmitted and received signals via Proteus animation VSM interface whereas Fig.13 shows the encoder signal received by a LabVIEW User Interface via an RS232 standard. As we can see, VISA drivers running on the Real Time processor is used to communicate with the RS-232 port. The waveform graph shown by this User Interface show clearly that the encoder signal that will be exploited by FPGA/Real Time toolkits to compute the control law is well received. Furthermore, the profile of the encoder signal is correct in amplitude and time and corresponds exactly to the signal send by Proteus VSM interface.

![Fig.11. Design of test verification circuit via Proteus VSM](image-url)
Fig. 12. Received and transmitted signals via Proteus animation VSM interface.

Fig. 13. Encoder signal send by Proteus VSM and received by a LabVIEW User Interface via an RS232 standard.
The control law (7), shown in Fig.14 is different from that applied in simulation results in [2] since we have replaced the saturation function by the sign function. In this figure, we can clearly see the chattering effect in the control variable profile. On the other hand, we can remark that the velocity variable shown in Fig. 15 makes three switching to cancel. So, we can deduce that the control law swings the pendulum to its upright position after three switches. We can also observe that the control law continues to switch which can prove that the pendulum not falls down but it is maintained at its upright position with almost zero velocity.

VI. CONCLUSIONS

In this paper, Real-Time embedded control is designed via FPGA and Real-Time Modules of LabVIEW software for swinging-up the inverted pendulum via an inherent non linear control strategy. A validation test is achieved using Proteus software via its Virtual Simulation Models.
Simulation results show the capabilities of the proposed approach to customize the complex control application with flexible time control without VHDL coding or board design. In future works, complex nonlinear controllers [28, 29, 30] will be designed for robotic applications via LabVIEW software using the same implementation strategy presented in this paper.

REFERENCES


