FPGA Implementation of PID Controller Using Xilinx System Generator

Santosh Kumar.B
CSE Department, New Horizon College of Engineering, Bengaluru, Karnataka, India

Abstract—In this paper implementation of digital PID controller using Field Programmable Gate array (FPGA) is presented. Firstly, the paper adopts genetic algorithm to optimize the parameters of PID controller and introduces Spartan3e FPGA to implement the PID controller. Secondly, the closed-loop test system is constructed by DSP builder and Matlab/Simulink.

Keywords—FPGA, PID controller, Genetic Algorithm, Matlab, Xilinx ISE13.1, Spartan3e.

I. INTRODUCTION

An arrangement of physical components in such a manner to regulate itself or another system is a control system. This control system is composed of a plant which has to be regulated and a controller. The function of a controller is to obtain the desirable characteristics avoiding undesired characteristics. The controller can be configured in analogue or discrete. On a whole, the realization of discrete controller can be comprehends with the aid of microcontroller, microprocessor or FPGA. Where FPGA features speed, accuracy, power compactness, and cost improvement. The majority of the regulatory loops in the process industries use conventional PID controller. System generating tool is the industry’s leading high-level tool. It is possible to provide system modeling and automatic code generation from Simulink® and MATLAB®. It incorporates RTL, embedded, MATLAB and hardware components of a system.

II. DISCRETIZATION PID CONTROLLER

PID controller is traditionally been chosen by control system engineers due to their flexibility and reliable. A PID controller has proportional, integral and derivative terms that can be represented in continuous domain transfer function form as:

\[ C = K_p e + K_i \int e(t) \, dt + K_d \frac{de(t)}{dt} \]  \hspace{1cm} (2)

Taking Laplace transform of equation (2) result in,

\[ C(s) = K_p E(s) + \frac{K_i}{s} E(s) + K_d s E(s) \]  \hspace{1cm} (3)

Transforming the equation into discrete domain using Tustin approximation gives the transfer function of discrete PID controller.

\[ C(z) = G_p + G_i \frac{1+z^{-1}}{1-z^{-1}} + G_d \frac{1-z^{-1}}{1+z^{-1}} \]  \hspace{1cm} (4)

Where,

\[ G_p = K_p \]
\[ G_i = K_i \frac{T_s}{2} \]
\[ G_d = \frac{2K_d}{T_s + 2T_F} \]
\[ C = \frac{T_s - 2T_F}{T_s + 2T_F} \]

Where \( K_p, K_i \) and \( K_d \) are proportional, integral and derivative parameters respectively of digital PID controller and \( T_s \) is sampling time, \( T_F \) is derivative filter time.

The Figure 1 shows the PID controller corresponding to equation (4) in simulation.

The system under test is a DC motor whose speed is controlled by PID controller. Its transfer function is

\[ G(s) = \frac{12}{(0.0007 s^2 + 0.0539 s + 1.441)} \]  \hspace{1cm} (5)

Fig. 1 Simulation implementation of control system
The Ziegler–Nichols is an empirical method of tuning the controller. With the Z-N method the steady state characteristics is not met, where manually iterative tuning is required. So, the soft computing techniques are used to determine the optimal PID controller parameters to control the speed of DC motor. The soft computing technique i.e., Genetic algorithm has proved its excellence in giving better results by improving the steady state characteristics. Then PID controller is realized in Xilinx block set as shown in Fig.2.

III. RESULTS

The Fig.3 is response of open loop system without PID controller using MATLAB. The Fig.4 is response of closed loop system with PID controller using MATLAB. The Fig. 5 is a response of closed loop system with PID controller using DSP builder and Fig.6 Hardware response of the closed loop system with PID controller using FPGA. The comparison between ZN tuning method and GA tuning method is tabulated as table.

The synthesis using Xilinx ISE tools resulted in the following Synthesis report for a Spartan3e as target FPGA. The device utilization summary shown in Table 2 and a maximum frequency of 270.86 MHz is quiet faster for the control applications.
TABLE I
Comparison Between ZN Tuning Method and GA Tuning Method

<table>
<thead>
<tr>
<th>Tuning Method</th>
<th>Rise time(s)</th>
<th>Settling time(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZN method</td>
<td>1.3288</td>
<td>23.3607</td>
</tr>
<tr>
<td>GA method</td>
<td>1.084</td>
<td>11.9031</td>
</tr>
</tbody>
</table>

TABLE II
Device Utilization Summary

<table>
<thead>
<tr>
<th>Logic Utilization</th>
<th>Used</th>
<th>Available</th>
<th>Utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Slice Flip Flops</td>
<td>84</td>
<td>3,312</td>
<td>1%</td>
</tr>
<tr>
<td>Number of 4-input LUTs</td>
<td>289</td>
<td>3,312</td>
<td>3%</td>
</tr>
<tr>
<td>Number of occupied Slices</td>
<td>179</td>
<td>4,656</td>
<td>3%</td>
</tr>
<tr>
<td>Number of Slices containing only related logic</td>
<td>179</td>
<td>179</td>
<td>100%</td>
</tr>
<tr>
<td>Number of Slices containing unrelated logic</td>
<td>0</td>
<td>179</td>
<td>0%</td>
</tr>
<tr>
<td>Total Number of 4-input LUTs</td>
<td>325</td>
<td>3,312</td>
<td>3%</td>
</tr>
<tr>
<td>Number used as logic</td>
<td>289</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number used as a route-thru</td>
<td>36</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of bonded I/Os</td>
<td>65</td>
<td>232</td>
<td>28%</td>
</tr>
<tr>
<td>Number of BUFG/AOs</td>
<td>1</td>
<td>24</td>
<td>4%</td>
</tr>
<tr>
<td>Average Fanout of Non-Conn Nets</td>
<td>1.86</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Design statistics: Minimum period: 3.69ns (Maximum frequency: 270.86MHz)

IV. CONCLUSION

The PID controller for the DC motor speed control was successfully implemented on Spartan 3E FPGA kit. Found that genetic algorithm method is better compared to ZN tuning method in terms of rise time and settling time. Implementing PID controller on FPGA features speed, accuracy, power compactness, and cost improvement. The project methodology helps us to view and compare the simulation result and the hardware FPGA output on the same personal computer so that we can co-relate the simulation result with the real time hardware output. Future development would be designing IP core of PID controller with Auto calculation and tuning of parameters on PSOC.

REFERENCES

[7]. https://www.xilinx.com/