PID Controller Design for Field Programmable Gate Arrays

Document Number AE2005002.0

James Bonanno, P.E.
http://www.atlantixeng.com

Revision History

Release Date	Release Version	Version Number
March 2003	Initial Release	AE2003001-1
April 2003	Updated Release	AE2003001-2
Sept. 2005	Updated Release	AE2005002.0

I. Introduction

The digital PID controller has been used extensively in real time digital control for many decades. The PID is used extensively in fields such as servo/motor control, robotics, temperature control, and power electronics. It has a long history of development and very mature tuning rules. More recently, the PID has been adopted into variant forms that incorporate adaptive and non-linear controllers. Among these, fuzzy logic and adaptive control often are built around a “PID” base structure. Overall, the PID is an important tool for the embedded real-time digital controls designer.

Recently, the emergence of Field Programmable Gate Arrays and hardware description languages (HDLs) now allows for added dimensions of digital PID Controllers: *parallelism, programmable bit widths, and absolute determinism*. These features are readily accomplished with FPGA and ASIC technology, and these features distinguish FPGA implementations from traditional micro-controller and DSP micro-controllers. Real time control is enhanced with non-interrupt driven structures, and the certain, non-interrupt timing driven data-path process of a PID implemented in FPGA is significantly attractive. An engineer may decide to stack several PID controllers in parallel to run multiple processes within the same chip, for example.

The PID is a classic Infinite Impulse Response or IIR digital filter that typically incorporates additional decision logic. Signal processing and communications engineers have been implementing IIR filters into the FPGA fabric for quite some time now, particularly during the last five year. The decision logic that is present in a PID controller is a distinguishing factor from a standard type of IIR filter. For example, there is often decision logic to realize a clamping function that freezes the integral portion of the filter when the filter output reaches a large positive or negative value. This is often referred to as the positive or negative controller limit or saturation value. Thus, the modern PID is really an IIR filter with a finite state machine (FSM) realizing an anti-windup clamp function. Verilog or VHDL are Hardware Description Languages (HDL) that readily and easily synthesizes the PID controller.

The successful hardware implementation of a PID controller in FPGA technology must take into account overall performance criteria expected of the PID. These criteria are data throughput rate, logic area utilized, and ease of configuration and tuning. An automated test bench to test the different configurations of the PID assures the designer that indeed the PID will work in the real system as designed.

The first mechanism to design the controller for a FPGA target includes the selection of the data path and control process. Once the data and control elements are assembled into an appropriate digital form with a HDL such as VHDL or Verilog, an extensive process of testing and verification of the design using a tool such as Modelsim™ is the next step. The presence of a sound test bench allows the designer to not only examine whether the design meets performance requirements, but to optimize the design. This is where the designer can assess the proper operation of the PID based on different logic area design requirements.

For example, if the design is constrained to take less than 30 LUTs (Look Up Table) elements and have a data latency of 200 nano-seconds, then the designer will most likely choose a FSM (Finite State Machine) based filter design. The design would likely use a single-cycle multiplier and a finite state machine to sequence the entire filter, rather than having dedicated adders and multipliers at each stage in the filter for realizing the controller. These basic design principles and process may also be applied to other types of digital controllers when targeting FPGA technology, ranging from fuzzy logic based controllers to modern state variable filters utilizing state feedback.

The fundamental issues involved in data throughput versus logic area and the overall elegance of the design are common issues in the design of digital hardware controllers. The non-interrupt driven nature of a PID synthesized for a FPGA is a consequence of the HDL design. This, in turn, creates a strict level of determinism.

II. Example Hardware PI Controller Design

The PID controller can have a high, medium, or low level of data throughput. The resulting logic area utilization is nearly inversely proportional to the data throughput rate. For illustrative purposes, a high data-throughput rate Proportional-Integral PI Controller is first examined, and the digital design requirements to meet the performance criteria are discussed.

The discrete time difference equation for the PI controller is thus:

$$I(k) = (E(k)+E(k-1))*Ki*Ts/2 + I(k-1) \text{ (bilinear transform)}$$

$$P(k) = Kp*E(k)$$

For this first example, the data path elements are not reused. Instead, they are typically used once during the data path

processing of the error signal. The data-path elements include an adder/subtractor, multiplier, delay element, and comparator. These data-path elements are in turn combined into functional blocks. These functional blocks include a scaling operation (multiplier), discrete time integrator (adder and delay element), proportional gain (multiplier), and clamp function (comparator). The PI controller is shown in Figure 1 below.

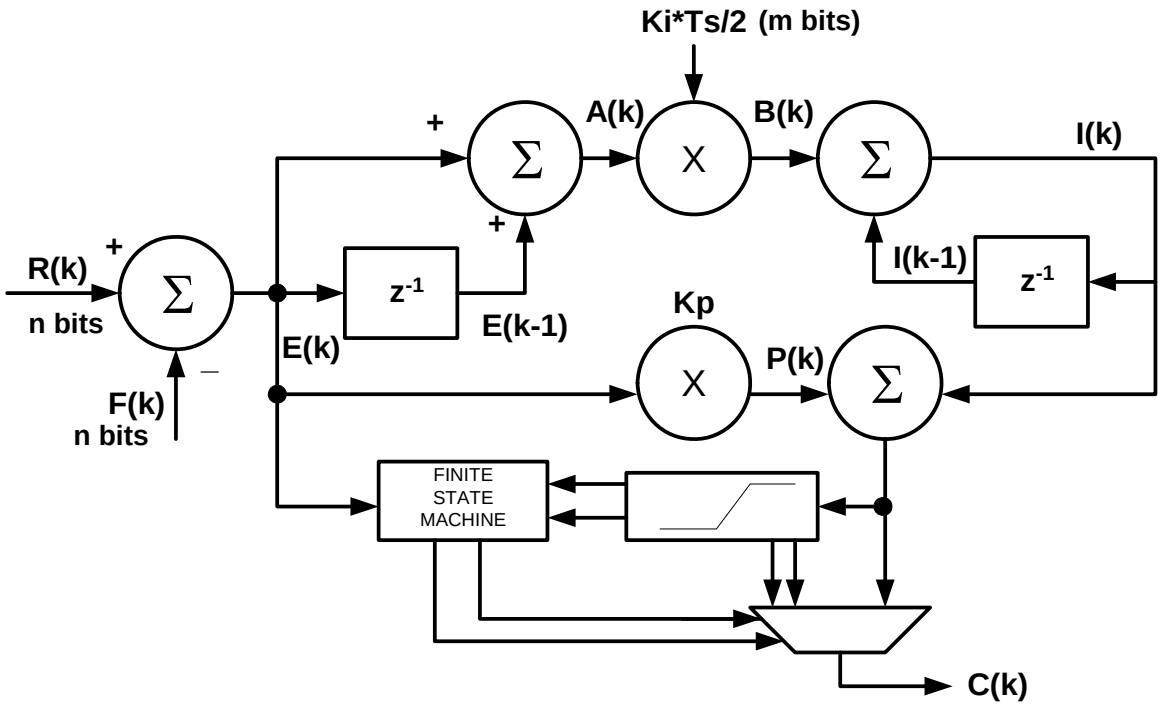


Figure 1: PI Controller Block Diagram

This design will achieve a maximum amount of data throughput if the all the data path elements are minimal latency. Allocating one clock for each data path element, the design will take seven clock cycles to produce the output. The bit widths deserved attention. For example, the error signal E(k) should have a bit width that prevents overflow from occurring. So in general, the E(k) signal should be n+1 bits where R(k) and F(k) are n bits. The analysis continues all the way through the data flow chain for the filter and is summarized in Table 1 below.

Signal	Bit Width	Comments
R(k)	N	Reference Signal
F(k)	N	Reference Signal
E(k)	N+1	Error Signal
A(k)	N+2	Error + Previous Error Signal
B(k)	(N+2)+M	Scaled A(k) by Ki*Ts/2
I(k)	(N+2)+M+1	Accumulate I(k) with B(k)

Table 1: Bit Width of PI Filter Intermediate Nodes

The results of the bit width reveal that higher bit width filters (say beyond 16 bits) are easily built in hardware without necessarily using a 32 bit data path or register. For example, a filter starting with 18 bits on R(k) and F(k) with a 16 bit coefficient for the integral gain would result in a 37 bit width output, which is easily realized in the flexible FPGA digital fabric while only using the necessary amount of bits. This is a nice feature of FPGA implementation.

III. Hardware PID Controller Testbench Process

An engineer knows that hardware can be tested in the laboratory with the proper stimulus and equipment to measure the response of the circuit under test. The same is true of digital hardware designed for FPGA with HDL's. However, the entire process is relegated to the computer, and the simulation can be assured of being timing accurate. This is always true if the post placed and routed design from the FPGA is put back into the simulator and run in the testbench. This insures that the HDL design under test has the appropriate delays inserted.

What is the best way to create a testbench for the PID controller? Well, consider the general model of the testbench process shown in Figure 2. This is a "closed loop" process with ideas familiar from Model Reference Control. Here, however, the ideas are used for testing of HDL modules written in either Verilog or VHDL.

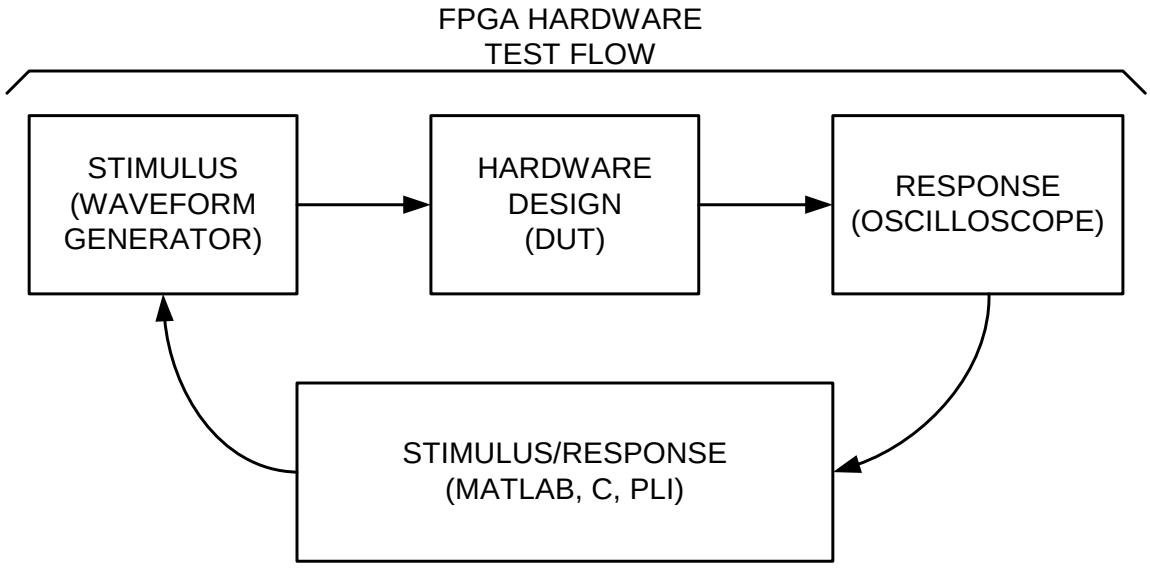
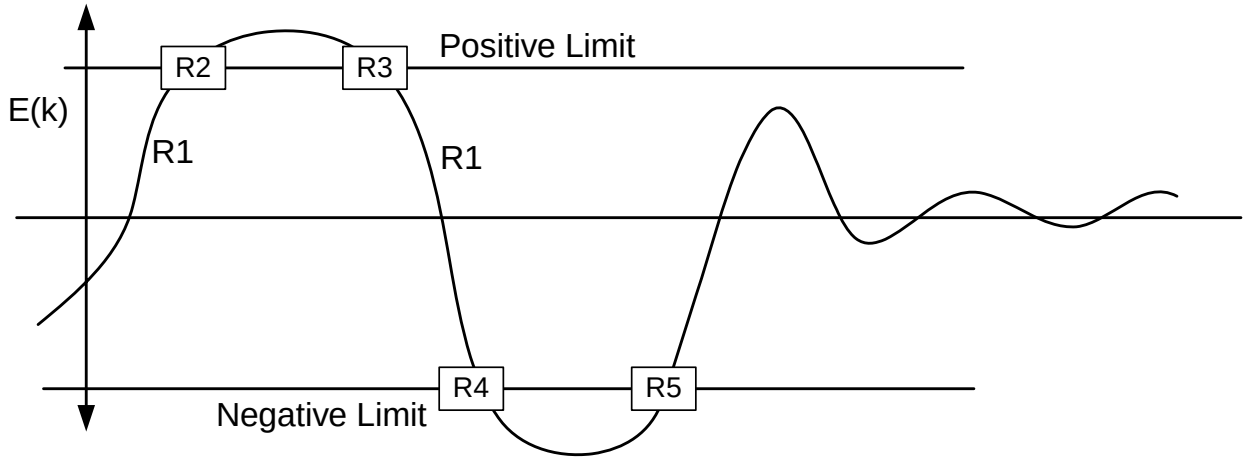


Figure 2: Generating RESPONSE and Performing ANALYSIS for FPGA Hardware

The PID controller must be tested properly. The analogy of taking a piece of hardware to the lab and testing it with a function generator and then measuring the results with a scope apply here as well. The laboratory is the computer, and if a digital design works on a tool such as Modelsim or Aldec, rest assured that it has a very high chance of working in the lab. Usually failure in the lab is the result of interconnect or technology issues. In terms of testing the PID controller, the testing bounds include the following:

1) Linear Region
2) Entry into Positive Clamp from Linear Region
3) Entry into Negative Clamp from Linear Region
4) Exit from Positive Clamp into Linear Region
5) Exit from Negative Clamp into Linear Region



If the controller successfully passes each of these tests, the integrity of the hardware is verified. In order to have flexibility in doing this, a Matlab or C file creates the stimulus ROM for the testbench as well as the “reference” stimulus and response waveforms. Overall, this is an important point. It has been found be experience that the most SUBTLE errors in the digital design will be uncovered if the ENTIRE functional basis is covered for the digital design unit. For example, one may have designed a state machine to handle the anti-windup function. The entry into the clamp region could have been designed as a latch. However, if it is an edge triggered latch rather than truly registered, once entry into the clamp region has occurred, exit cannot occur. Therefore, to test for such a possible condition, the stimulus should be created to exercise all bounds of the functional operation of the controller. The method for generating this stimulus will now be discussed.

V. Conclusions

A digital PID controller implemented in FPGA technology is a configurable controller in terms of latency, resolution, and parallelism. High bit width controllers can easily be implemented in FPGA technology with ease and reliability. In addition, *programmable bit width* digital filters are also achieved with FPGA technology. The speed or execution or latency of the controller can be precisely controlled with the amount of reuse of arithmetic elements such as multipliers and adders. Overall, the digital realization of PID controller is simplified with an exhaustive test bench. The FPGA based PID controller is an elegant solution for many applications in power electronic converter control, robotics, and motion control. This is particularly true when a design calls for parallelism either in the structure of an individual PID controller or “stacks” of PID controllers. The speed of execution of a FPGA based PID controller can be less than 100nsec if desired for high throughput requirements. This makes the digital PID an essentially “digital” version of an analog controller with all the flexibility and robustness of digital implementation. Thus, by creating the PID controller in a non VonNuemann style of architecture, a great deal of flexibility and power is obtained that the designer is sure to value as more sophisticated and reliable systems result from the implementation of such technology.