

# Implementing a PID Controller in an Actel FPGA

## Introduction

Almost any process requires control of the output to achieve the desired result. Such processes include attitude control of an aircraft, speed control of an elevator, voltage control of a power supply, and mixing and temperature control in a chemical plant. A commonly used control methodology for these examples and many others is proportional plus integral plus derivative (PID) control. PID control is a combination of these three types of control algorithms.

The history, derivation, and theory behind PID control are beyond the scope of this application note. Such topics are addressed in university-level courses and their associated textbooks<sup>1</sup>.

This application note discusses how a PID algorithm can be used to control the output voltages of a power supply as a system is powering up or in steady-state regulation. Also discussed is how the PID algorithm is implemented in an Actel Fusion FPGA as the board-level system management IC.

## A Closed-Loop, Controlled Ramping Power Supply

Many processors, DSPs, and FPGAs use multiple supply voltages. Manufacturers of such parts often recommend that the multiple supply voltages meet certain criteria when the device is powered up. Such criteria can be as follows:

1. Multiple supply voltages ramp-up monotonically (a linear ramp without spikes or dropouts).
2. Each supply is kept within  $\pm 0.5$  V of the other supply voltages as they ramp.

Historically, power supplies with this kind of output control have been implemented using analog circuitry, as shown in Figure 1.

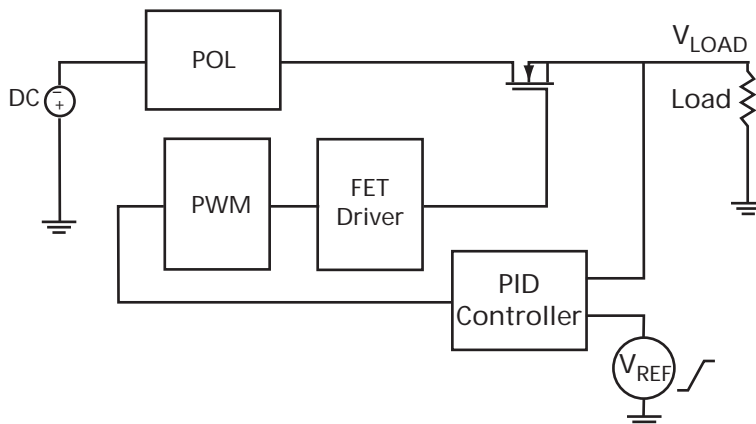


Figure 1 • Power Supply with Controlled Ramping of Output Voltage

Note that Figure 1 represents only one output voltage being ramped. To ramp multiple output voltages, each voltage being ramped requires one of the circuits shown.

1. Franklin, Powell, Emami-Naeini; "Feedback Control of Dynamic Systems," Addison Wesley, ISBN 0-201-11450-9.

## PID Control Implemented with Continuous Time (analog) Circuitry

The analog circuit implementation of a PID controller is shown in Figure 2.

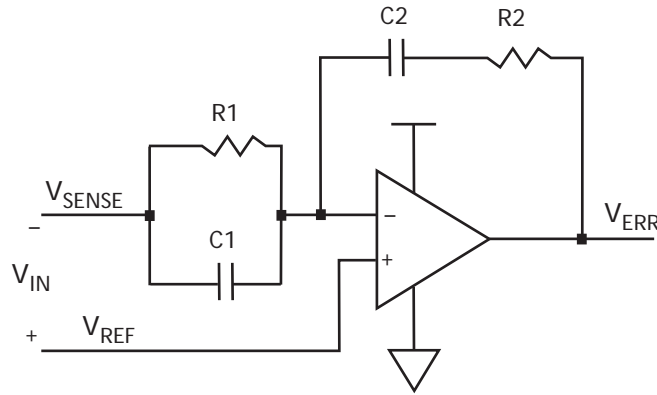


Figure 2 • PID Controller

EQ 1 is the equation for a PID controller in the continuous time domain.

$$V_{ERR}/V_{IN} = -Z2/Z1$$

EQ 1

Where

- $V_{IN}$  =  $V_{REF} - V_{SENSE}$
- $V_{SENSE}$  = The signal being sampled by the PID. In this example,  $V_{SENSE}$  is the power supply's  $V_{LOAD}$  signal
- $V_{REF}$  = The reference signal the PID has to follow or track
- $V_{ERR}$  = The output signal of the PID (error voltage)
- $Z2$  = The feedback impedance
- $Z1$  = The input impedance

EQ 1 can be moved into the frequency domain using LaPlace transforms. This gives EQ 2:

$$V_{ERR}/V_{IN} = - [(R2/R1) + (C1/C2) + (1/(R1 \times C2))] \times (1/s) + R2 \times C1s]$$

EQ 2

The coefficients in EQ 2 are commonly grouped and labeled as follows:

$$K_p = (R2/R1 + C1/C2)$$

$$K_i = (1/R1 \times C2)$$

$$K_d = R2 \times C1$$

The coefficients,  $K_p$ ,  $K_i$ , and  $K_d$  represent the coefficients of the proportional, integral, and derivative portions of the control algorithm.

Many software tools exist that help the designer to determine values for the  $R$ s and  $C$ s (or the  $K_p$ ,  $K_i$ , and  $K_d$ ) coefficients that will stabilize the output of the system being controlled. SPICE circuit simulation was used to determine the circuit values for the power supply controlled by the PID discussed in this application note. The values found were as follows:

$$C1 = 1 \text{ nf}$$

$$R1 = 1,000 \text{ Ohms}$$

$$C2 = 10 \text{ nf}$$

$$R2 = 10 \text{ K Ohms}$$

## PID Control Implemented with Discrete Time (digital) Circuitry

Since an FPGA is not an analog device, EQ 3 cannot be directly implemented in an FPGA. It must first be transformed into a discrete time algorithm that can be implemented in digital logic.

The transformation starts with the frequency domain equation (EQ 2 on page 2 is repeated here as EQ 3):

$$V_{ERR}/V_{IN} = -[(R2/R1) + (C1/C2) + (1/(R1 \times C2)) \times (1/s) + R2 \times C1s] \quad EQ 3$$

Using Z transforms in the “backward difference” method, EQ 3 is transformed into a velocity PID algorithm<sup>2</sup>. The result is EQ 4:

$$V_{ERR}(t) = V_{ERR}(t-1) + K_p \times [V_{IN}(t) - V_{IN}(t-1)] + (K_d/T_s) \times [V_{IN}(t) - (2 \times V_{IN}(t-1)) + V_{IN}(t-2)] + K_i \times V_{IN}(t) \quad EQ 4$$

Where

- $V_{ERR}(t)$  = Output at time t
- $V_{ERR}(t-1)$  = Output at time t - 1
- $V_{IN}(t)$  = Input at time t
- $V_{IN}(t-1)$  = Input at time t - 1
- $V_{IN}(t-2)$  = Input at time t - 2
- t = Present time step
- t - 1 = Time step immediately previous to t
- t - 2 = Time step immediately previous to t - 1
- $T_s$  = Sampling period

Plugging in the values from the circuit in Figure 2 on page 2 provides the following results:

$$K_p = (R2/R1 + C1/C2) = (10 \text{ K}/1000) + (1n/10n) = 10.1$$

$$K_i = (1/R1 \times C2) = 1/(1000 \times 10 \text{ n}) = 100 \text{ K}$$

$$K_d = R2 \times C1 = 10 \text{ K} \times 1 \text{ n} = 10 \mu$$

All values of the discrete time PID algorithm are now known except for  $T_s$ , the sampling period.  $T_s$  represents how often the PID samples its input signal. In this power supply,  $T_s$  is how often the PID samples the output of the power supply,  $V_{LOAD}$ .

The Actel Fusion FPGA has a built-in analog-to-digital converter (ADC). This ADC is used to sample the power supply output. The ADC conversion time is the amount of time required for the ADC to produce the digital result of what it sampled. Assuming the ADC digital result is fed directly to the PID input, the ADC conversion time becomes  $T_s$ , the sampling period of the PID controller. The recommended acquisition time for the ADC is 10  $\mu$ s/channel, which causes the conversion time to be 33.9  $\mu$ s. a 35  $\mu$ s conversion time was chosen to make the math simpler.

Now that  $K_p$ ,  $K_i$ ,  $K_d$ , and  $T_s$  are known, the discrete time algorithm becomes EQ 5:

$$V_{ERR}(t) = V_{ERR}(t-1) + 10.1 \times [V_{IN}(t) - V_{IN}(t-1)] + 0.28571 \times [V_{IN}(t) - 2 \times V_{IN}(t-1) + V_{IN}(t-2)] + 3.5 \times V_{IN}(t) \quad EQ 5$$

This algorithm can be implemented into digital logic.

2. M. Tham, "Discretised PID Controllers: Part of a Set of Study Notes on Digital Control, <http://lorien.ncl.ac.uk/ming/digicon/digimath/dpid1.htm>.

## V<sub>REF</sub> Implemented with Digital Logic

V<sub>REF</sub>, the voltage reference signal that the PID must track, can be implemented in digital logic with simple counters. The purpose of the PID is to control multiple power supply outputs so that they ramp monotonically and stay within +/-0.5 V of each other as they ramp. The power supply discussed in this application note has three output voltages: 1.2 V, 2.5 V, and 3.3 V. Thus V<sub>REF</sub> is actually three signals, each ramping from 0 to 1.2, 2.5, or 3.3 V at the same rate.

V<sub>IN</sub>, the input voltage to the PID, is given by EQ 6:

$$V_{IN} = V_{REF} - V_{SENSE}$$

EQ 6

where V<sub>SENSE</sub> is the power supply output voltage sensed by the ADC.

The Fusion ADC used to sense the output voltage is configured to be 8-bits. Therefore the output of the ADC is an 8-bit digital word representing the analog voltage it senses at the power supply outputs. For the above V<sub>IN</sub> equation to work in digital logic, V<sub>REF</sub> must also be an 8-bit digital word. To determine the 8-bit digital word equivalent of 1.2, 2.5, or 3.3 V for V<sub>REF</sub>, EQ 7 from the Fusion Datasheet is used:

$$ADC_{OUT} = (V_{IN} \times ADC_{INPUTSCALEFACTOR} / ADC_{REFERENCE}) \times 2^{ADC_{BITS} - 1}$$

EQ 7

Where

ADC<sub>OUT</sub> is the ADC output 8-bit digital word.

V<sub>IN</sub> is the voltage sampled by the ADC input.

ADC<sub>INPUTSCALEFACTOR</sub> is the scale factor applied of the input voltage depending on the range of the ADC<sub>INPUTSIGNAL</sub>. This is 1.25 when using the 0 to 2 V scale and 0.625 when using the 0 to 4 V scale.

ADC<sub>REFERENCE</sub> is the internal reference voltage used by the ADC in its conversion (2.56 by default).

ADC<sub>BITS</sub> is the number of bits of the ADC output word (8 for this example).

## Calculations

### For the 1.2 V Output

$$ADC_{OUT} = ((1.2) \times (1.25) / 2.56) \times (255) = 149.41 \cong 149 \Rightarrow 10010101$$

### For the 2.5 V Output

$$ADC_{OUT} = ((2.5) \times (0.625) / 2.56) \times (255) = 155.63 \cong 156 \Rightarrow 10011100$$

### For the 3.3 V Output

$$ADC_{OUT} = ((3.3) \times (0.625) / 2.56) \times (255) = 205.44 \cong 205 \Rightarrow 11001101$$

Thus, V<sub>REF</sub> is set to 10010101 to represent 1.2 V, 10011100 to represent 2.5 V, and 11001101 to represent 3.3 V.

Now that the digital word equivalents of the V<sub>REF</sub> values are known, the rate that V<sub>REF</sub> ramps to these values must be chosen. Many parts requiring voltage rail ramp control specify that the voltage should ramp within 100 ms. As shown in EQ 3 to EQ 5 on page 3, the PID used in conjunction with the Fusion ADC allows for considerable margin. The ramp-rate in this example has been selected to enable the PID and ADC to keep the output voltages stable and linear. The ADC has a sample rate four times that of the V<sub>REF</sub> increment rate. This oversampling technique is common, allowing for accurate voltage measurement and optimizes the PID control loop performance.

The ADC conversion time is 35 μs. To ensure adequate time for the PID to compensate for a particular value of V<sub>REF</sub>, the ADC was allowed to sample the output four times for each time V<sub>REF</sub> increases by one. Thus V<sub>REF</sub> increments by one every 4 × 35 μs = 140 μs. The 1.2 V V<sub>REF</sub> ramps from 0 to 149 in 150 increments or counts. The total ramp time is 140 μs × 150 = 21 ms. For this example the ramp-rate is 1.2 V in 25 ms, as this will provide some margin to ensure the ADC can sample the output 4 times before the next V<sub>REF</sub> increment. This increases the V<sub>REF</sub> increment time to 25 ms/150 = 166.66 μs.

The PID uses a 10 MHz input clock, driving a counter, to count up to 1,667 and then increment V<sub>REF</sub> by one, causing V<sub>REF</sub> to count from 0 to 149 in 25 ms. Recall that all three outputs ramp at the same rate, so they will meet the specification that each output should stay within +/-0.5 V of all others as they ramp. This leads to the calculations of ramp times and 10 MHz counter values for V<sub>REF</sub> of 1.2, 2.5, and 3.3 V shown in EQ 8 through EQ 10.

### For the 1.2 V Output

$$\text{Ramptime} = 1.2 \text{ V} \times (25 \text{ ms}/1.2 \text{ V}) = 0.025 \text{ s}$$

$$10 \text{ MHz counter value} = (0.025 \text{ s}/150) \times 10 \text{ MHz} = 1667 \cong 4.7x \text{ the } 35 \mu\text{s ADC conversion time}$$

EQ 8

### For the 2.5 V Output

$$\text{Ramptime} = 2.5 \text{ V} \times (25 \text{ ms}/1.2 \text{ V}) = 0.0521 \text{ s}$$

$$10 \text{ MHz counter value} = (0.0521 \text{ s}/157) \times 10 \text{ MHz} = 3317$$

EQ 9

### For the 3.3 V Output

$$\text{Ramptime} = 3.3 \text{ V} \times (25 \text{ ms}/1.2 \text{ V}) = 0.069 \text{ s}$$

$$10 \text{ MHz counter value} = (0.069 \text{ s}/206) \times 10 \text{ MHz} = 3337$$

EQ 10

The  $V_{REF}$  signals (1.2, 2.5, and 3.3) are each implemented using two counters, one toggling the other. For a  $V_{REF}$  of 1.2 V, the 10 MHz counter increments to 1,667 and then toggles the  $V_{REF}$  counter to increment by 1.  $V_{REF}$  1.2 will reach 149 in 25 ms. For a  $V_{REF}$  of 2.5 V, the 10 MHz counter increments to 3,317 and then toggles the  $V_{REF}$  counter to increment by 1.  $V_{REF}$  2.5 will reach 157 in 52.1 ms. For a  $V_{REF}$  of 3.3 V, the 10 MHz counter increments to 3,337 and then toggles the  $V_{REF}$  counter to increment by 1.  $V_{REF}$  3.3 will reach 206 in 69 ms. Each  $V_{REF}$  ramp-rate is 1.2 V/25 ms.

## Fusion PID Implementation Details

Shift and add techniques instead of multipliers were used to implement the coefficients of 10.1, 0.28571, and 3.5, to save processing time and logic gates. The discrete time PID algorithm shown in EQ 5 on page 3 was initially implemented in a Fusion device using 25-bit operands. The lower 8 bits were used to handle the calculations involving the fractional portion of the coefficients. The upper 18-bits handled the integral portion of the calculations. The PID algorithm was written in VHDL and took the form shown in EQ 11.

$$V_{ERR} = V_{ERRP} + A + B + C + D + E + F + G + V_{IN} + I + J + K$$

EQ 11

Where

$$\begin{aligned} V_{ERRP} &= V_{ERR}(t-1) \\ V_{IN} &= V_{IN}(t) \\ V_{INP} &= V_{IN}(t-1) \\ V_{INP2} &= V_{IN}(t-2) \\ V_{IN}V_{INP} &= V_{IN}(t) - V_{IN}(t-1) \\ V_{INP}X2 &= 2 \times V_{INP} \\ V_{INDT} &= V_{IN} - V_{INP}X2 + V_{INP2} \\ A &= 8 \times V_{IN}V_{INP} && \text{shift } V_{IN}V_{INP} \text{ left } 3 \\ B &= 2 \times V_{IN}V_{INP} && \text{shift } V_{IN}V_{INP} \text{ left } 1 \\ C &= 0.0625 \times V_{IN}V_{INP} && (1/16) \text{ shift } V_{IN}V_{INP} \text{ right } 4 \\ D &= 0.03125 \times V_{IN}V_{INP} && (1/32) \text{ shift } V_{IN}V_{INP} \text{ right } 5 \\ E &= 0.0078125 \times V_{IN}V_{INP} && (1/128) \text{ shift } V_{IN}V_{INP} \text{ right } 7 \\ F &= 2 \times V_{IN} && \text{shift } V_{IN} \text{ left } 1 \\ G &= 0.5 \times V_{IN} && \text{shift } V_{IN} \text{ right } 1 \\ I &= 0.25 \times V_{INDT} && \text{shift } V_{INDT} \text{ right } 2 \\ J &= 0.03125 \times V_{INDT} && (1/32) \text{ shift } V_{INDT} \text{ right } 5 \\ K &= 0.00390625 \times V_{INDT} && (1/256) \text{ shift } V_{INDT} \text{ right } 8 \end{aligned}$$

The actual coefficients became 10.1015625, 0.28515625, and 3.5. This implementation was tested in actual hardware and had the expected results.

## Refining the Fusion PID Implementation

The solution in EQ 11 on page 5 offers optimal results. Depending upon design performance requirements, the design can be optimized for implementation within an FPGA environment with minimal impact to accuracy. Tests were conducted where the algorithm coefficients were changed so that the number of bits needed to calculate the fractional portion of the coefficients could be reduced. Coefficients of 10, 0.5, and 3.5 also gave satisfactory results. This reduced the number of bits used for fractional calculations from 8 to 1 and thus reduced the operand size from 25 to 18-bits. Simulation also showed that operand values were unlikely to rise above 1,600 for this example. This allowed the operand to be reduced to 12-bits total, with the MSB used for a sign bit and the LSB used to calculate the fractional portion of the operand values. The resulting PID algorithm is shown in EQ 12.

$$V_{ERR} = V_{ERRP} + A + B + F + G + V_{IN} + P$$

EQ 12

Where

$$\begin{aligned} V_{ERRP} &= V_{ERR}(t-1) \\ V_{IN} &= V_{IN}(t) \\ V_{INP} &= V_{IN}(t-1) \\ V_{INP2} &= V_{IN}(t-2) \\ V_{IN}V_{INP} &= V_{IN}(t) - V_{IN}(t-1) \\ V_{INP}X2 &= 2 \times V_{INP} \\ V_{INDT} &= V_{IN} - V_{INP}X2 + V_{INP2} \\ A &= 8 \times V_{IN}V_{INP} && \text{shift } V_{IN}V_{INP} \text{ left 3} \\ B &= 2 \times V_{IN}V_{INP} && \text{shift } V_{IN}V_{INP} \text{ left 1} \\ F &= 2 \times V_{IN} && \text{shift } V_{IN} \text{ left 1} \\ G &= 0.5 \times V_{IN} && \text{shift } V_{IN} \text{ right 1} \\ P &= 0.5 \times V_{INDT} && \text{shift } V_{INDT} \text{ right 1} \end{aligned}$$

## Advantages of Implementing PID Control Using Digital Logic

The analog implementation of the PID controller in this example required one PID circuit for each output being controlled. In the power supply example referenced in this application note, three supply rails were being controlled, each requiring a PID for control. However, if the design is implemented within a programmable logic environment, the PID controller can be time shared between the three different voltage rails being controlled, reducing the cost of implementation. Also, the  $V_{REF}$  signal can be easily programmed into digital logic, further reducing circuit size and cost.

## List of Changes

The following table lists critical changes that were made in each revision of the document.

Revision*	Changes	Page
Revision 1 (June 2011)	Modified EQ 4 and EQ 5.	3

*Note:* \*The revision number is located in the part number after the hyphen. The part number is displayed at the bottom of the last page of the document. The digits following the slash indicate the month and year of publication.



Actel and the Actel logo are registered trademarks of Actel Corporation.  
All other trademarks are the property of their owners.



[www.actel.com](http://www.actel.com)

**Actel Corporation**

2061 Stierlin Court  
Mountain View, CA  
94043-4655 USA

**Phone** 650.318.4200

**Fax** 650.318.4600

**Actel Europe Ltd.**

River Court, Meadows Business Park  
Station Approach, Blackwater  
Camberley Surrey GU17 9AB  
United Kingdom

**Phone** +44 (0) 1276 609 300

**Fax** +44 (0) 1276 607 540

**Actel Japan**

EXOS Ebisu Bldg. 4F  
1-24-14 Ebisu Shibuya-ku  
Tokyo 150 Japan

**Phone** +81.03.3445.7671

**Fax** +81.03.3445.7668

[www.jp.actel.com](http://www.jp.actel.com)

**Actel Hong Kong**

Suite 2114, Two Pacific Place  
88 Queensway, Admiralty  
Hong Kong

**Phone** +852 2185 6460

**Fax** +852 2185 6488

[www.actel.com.cn](http://www.actel.com.cn)