Implementing Process Control Systems with Field Programmable Analog Arrays

By Chris Schene
chris.schene@anadigm.com

#ESC-206 at the Embedded Systems Conference – San Francisco 2004
Introduction

PID control has been around for several generations now, and over 90% of all industrial control loops are still implemented as some form of PID controller. The PID control algorithm has been thoroughly researched and is well understood. Numerous simulation, tuning, and analysis tools, and techniques are available to help analyze and design controllers for many commonly seen processes. In many cases a controller can be designed and tested using computerized, graphical, or plant model estimation techniques without the need for delving heavily into mathematical analysis of the system.

The emergence of dynamically reconfigurable programmable analog ICs has added to the options available to designers for implementing PID control loops. The field programmable analog array (FPAA) is well suited for embedded processes control because it can be loaded from an EEPROM or partially reprogrammed from a processor of any kind within a few micro seconds. Circuit parameters, such as gain and corner frequency, are typically accurate to within 1% or less, and the devices are practically immune to component drift due to aging and severe temperature changes that are seen in industrial environments. The ability of the FPAA to be reprogrammed on the fly makes it well suited for adaptive control.

In this paper we will discuss the basics of single-loop and adaptive PID control and show how these can be used to control processes with difficult dynamics. As examples, we will show a single-loop controller for a thermal process and adaptive control of a balance beam built with an FPAA. The FPAA used for these designs is accompanied by a graphical EDA tool set in which FPAA circuit functions, such as gain, filtering and rectifying are abstracted to functional software blocks known as Configurable Analog Modules (CAMs). These software modules are simply dragged on to the device work space, configured graphically, wired together and either downloaded to a test board, burned into an EPROM or loaded from a processor. No complex math, soldering or bread boarding of the FPAA circuit is required.

Single-Loop Control

In classical feedback control theory, for any closed-loop system the transfer function between two variables is given by the expression:

\[ v_1(s) = \frac{v_2(s) \cdot FF}{1 + FB} \]

Where \(v_1\) and \(v_2\) are the variables in question, \(FF\) is the product of all expressions in the forward path from \(v_1\) to \(v_2\), and \(FB\) is the product of all elements in the feedback path. From this definition we can derive the classical feedback control equations:

\[ y = \frac{(cpr + d - cpn)}{(1 + pc)} \]
\[ u = \frac{(cr + d - cn)}{(1 + pc)} \]
\[ e = \frac{(r - d - n)}{(1 + pc)} \]

Where \(c\) = the controller and \(p\) = the plant which we are controlling. \(y\) is the plant variable (a.k.a. PV), \(u\) is the control variable, and \(e\) is the error signal.

An examination of the equations reveals that the response of the plant variables is a function of three system inputs:

---

1. Class notes from Fall 2001 CHE 461 Process Control class, Dr Daniel Rivera, ASU
1) $r$ - is the reference variable that we are attempting to drive the plant response to. It is often referred to as the set point (SP).

2) $d$ - is the disturbance. This is typically a deterministic, but often unknown effect external to the system being controlled that will push the process away from $r(t)$. In the thermal process illustrated in this paper by a water heater, disturbances occur as heat is lost to the environment and ice is dropped into the water. For the adaptive control process illustrated by a balance beam, the $d(t)$ is friction and weights on the end of the beam.

3) $n$ - is noise. This is a random effect, such as white or pink noise, that will cause errors in the process. Note from the equations above that noise directly fights the control action and tends to make it difficult to keep the process tracking the set point or $r$ parameter.

It is possible to solve a single-loop control problem by working out the closed-loop transfer functions in detail. Fortunately, this rather tedious mathematical process is often not required for the well studied and ubiquitous PID controller. Simulation of the plant in any of the widely available commercial tools is a straightforward and convenient choice: Cadence, Simulink, and the FPAA tool set mentioned in this paper are systems in which simulation models can be created. Empirical tuning methodologies such as the Ziegler-Nichols open- and closed-loop tuning methods are an excellent choice. There are several controller design techniques (such as IMC, Cohen Coon, and minimum error integral methodologies), and there are many commercial tools available to assist in choosing correct tuning parameters as well. If it is necessary to derive the equations, often it is easier to use simulation or empirical methods first and then derive the equations from the experimental data.

The formula for the PID controller that is graphically shown in Figure 1 is:

$$u(s) = e(s)(K_p + \frac{1}{T_i} + sT_d)$$

While there is certainly wide variation in which control system criteria are used to decide which type of PID controller is most appropriate for a given application, some generally applicable principles can be stated:

1) **A “P” controller** - The proportional controller increases the speed of the system response but leads a non-zero steady-state offset from SP for all but pure capacity processes (integrators). Use P alone only when steady-state offset can be tolerated or on pure capacity processes. Liquid level tanks are often controlled with P control.

2) **A “PI” controller** – Corrects for offsets but tends to make the system more oscillatory as P increases. Less sensitive to noise than a PD or PID controller.

3) **A “PID” Controller** - When it is necessary to compensate for natural sluggishness of a process and signals have a low noise level, the PID controller is a good choice. Temperature loops can often have a significant process lag that can be partially compensated for with the “D” term to overcome the naturally slow response of a temperature process once it starts moving. The dynamics of the “D” term can be very hard to deal with in the presence of high noise levels.

4) **A “PD” controller** – Can be used to stabilize controllers with high P gain and allows for a smaller steady-state offset. Either PD or PID are often necessary for open-loop unstable processes. The PD controller will have a steady-state offset for all but pure capacity plants.

In industrial practice, there are a number of variations of the PID controller in addition to the parallel form listed above. It is common to take the derivative on the PV rather than the error signal and to make accommodations for integrator windup.

---

3 “*Quick and Efficient PID Control Design*”, Waters, Schene, de la Torre, 2003 Boston Embedded systems conference.

4 “*The PID Control Algorithm: How it works and How to tune it*” John A Shaw, 11,7,2001., 51-53

5 “*The PID Control Algorithm: How it works and How to tune it*” John A Shaw, 11,7,2001., 54

6 “*The PID Control Algorithm: How it works and How to tune it*” John A Shaw, 11,7,2001., 47

7 “*The PID Control Algorithm: How it works and How to tune it*” John A Shaw, 11,7,2001., 20

8 OGR (523-4)


Plant Models
In classical feedback control theory and the diagram in Figure 1, the process being controlled is called the “plant.”\(^{11}\) The mathematical definition of this “plant” includes the actual process and sometimes the effect of instrumentation, driver, and actuator circuits.

The model of most plants is characterized mathematically in one of four ways\(^{12}\)

1) State space models (differential and difference equations)
2) Transform domain (Laplace or Z transforms)
3) Frequency response form (also known as complex variable)
4) Convolution (impulse response) form.

Since these models are all expressing the same essential mathematical characteristics, transformation between these models is possible.

The theory most commonly used for control system analysis and design focuses on linear mathematical models.\(^{13}\) The mathematical characteristics of most plants, even if they are highly non-linear, can be characterized in linear form by well-known and tested control analysis techniques.\(^{14}\) Process models can be formulated in a number of ways from basic conservation of energy, mass, momentum and heat transfer principles to the purely empirical engineering discipline of System Identification.

The discipline of System Identification (SYSID), often referred to as empirical process modeling, involves treating a process essentially as a “black box” and stimulating it with one or more forcing functions. This makes it possible to identify the mathematical process shape and characteristics by studying the process response to the forcing function. Typical forcing functions include a step, ramp, pseudo random sequences, or repeating (oscillatory) simulation. No a priori knowledge of the process is required and there is a wide variety of computerized tools and thoroughly tested techniques available for this ID process.

A significant percentage of open-loop stable, or self-regulating, industrial processes can be adequately approximated as a first-order exponential with delay, which is modeled with the following transfer function in Laplace form and only requires three parameters:

\[
\frac{V_{out}(s)}{V_{in}(s)} = e^{-\theta s} \frac{K}{Ts+1},
\]

where:

1) \(\theta\) = the process lag, that is the time delay from the start of the forcing function until the first process movement is detected
2) \(K\) = the process gain after the output has stabilized, \(K = \frac{V_{out}(final)}{V_{in}(Step)}\)
3) \(T\) = the process time constant, or the time from the first process movement until the process reaches 63.2% of final value

This model is also referred to the “first-order plus dead time approximation.”\(^{15}\)

Controlling Processes With Difficult Dynamics
Processes that have these characteristics tend to be very difficult to control\(^{16}\) and are briefly discussed below, but the reader is directed to outside references for detailed study on these systems.

---

\(^{11}\) “Basic Control Systems Engineering”, Paul H Lewis, Chang Yang, Prentice Hall 1997, 7
\(^{12}\) “Process Dynamics, Modeling and Control”, Oggunnaite and Ray, 625, Chapter 4
\(^{13}\) “Process Dynamics, Modeling and Control”, Oggunnaite and Ray, 625
\(^{14}\) “Process Dynamics, Modeling and Control”, Oggunnaite and Ray, 625-629
\(^{15}\) “The PID Control Algorithm: How it works and How to tune it” John A Shaw, 11,7,2001, 42-5
\(^{16}\) “Process Dynamics, Modeling and Control”, Oggunnaite and Ray Chapter 17
**Time delay systems** - There is a time lag between the time a control action is applied and the time the effect of this action is reflected in the process variable. The Laplace transfer function of such a process will contain an $e^{-\theta s}$ term. The major difficulty in controlling these types of systems is that the controller is dealing with out-of-date information. There are a number of approaches to time-delay compensation such as the Smith predictor that are beyond the scope of this paper. The thermal process shown in this paper is a time delay system. One very straightforward approach to dealing with time-delay systems is to use a derivative term with high gain which will cause the process to move more quickly in the same direction once it does start moving.

**Inverse response systems** – These processes will initially move in the opposite direction of the final process value. Mathematically, this type of process action indicates the presence of at least one Right Half Plane zero in the process transfer function. The major problem presented by inverse response systems is the confusing scenario that is presented to the controller: a high gain will quickly move the process in the opposite direction that it should be controlled in, possibly resulting in an unstable situation. The addition of a negative derivative term will tend to alleviate this tendency toward instability during the initial movement but will fight the P and I control as the process moves in the correct final direction. Time delay systems and inverse response systems are also referred to as NMP (Non Minimum Phase) processes.

**Open-loop unstable systems** - These are systems with an RHP pole or a pole on the axis. These processes are unstable or conditionally unstable in their natural state, and so an upset in either direction may result in an unstable response. The major difficulty with this type of process is that it cannot be tested with standard process identification open-loop techniques. The balance beam process described in this paper, whose transfer function is of the form $K/t$, is open-loop unstable. Unless the force exerted by control variable exactly matches the weight, any slight difference will cause the beam to fall to one of the physical stops. Most open-loop unstable systems can be controlled quite well with properly chosen closed-loop parameters.

**Designing a Single-Loop Process Controller in an FPAA**

The FPAA combined with its graphical tool set is an excellent platform for implementing single-loop controllers because:

1) Component values in the device are nearly immune to parameter variation due to aging and temperature drift.
2) The graphical FPAA tool set has abstracted circuit functions to drag and drop functional modules that are simply configured graphically, wired together and downloaded to the target FPAA via an SPI interface. The circuits can be created at a block diagram level without the need to delve into complex math, write code, or focus on analog circuit details.
3) There is no limit on the number of times the device can be reprogrammed. Complete reprogramming takes less than 50 microseconds and partial updates take just a few microseconds.
4) No need to change discrete components when design is changed.
5) Partial FPAA device parameter updates can be done in real time in as little as a few microseconds via the ANSI compatible “C” code generated by the FPAA tool kit. This “C” code consists of “C” functions that the user calls to update FPAA parameters.
6) Designs can be changed and re-tested in minutes allowing for easy iteration of many different design choices.
7) Circuits are saved as software and designs can be changed in the field by simply loading a new circuit.
8) High level design tools are available to generate analog filters of arbitrary complexity and PID controllers.
9) The time delay through the FPAA, because the signal remains analog and does not incur digital conversion or processing delays, can be as low as a microsecond.

In this section we go though the design of a single-loop PI controller for a thermal process implemented in an FPAA. This thermal process has a delay of 73 seconds and is considered “difficult to control” due to the presence of that time lag.
**Instrumentation Interface for a Thermal Process**

Cost, accuracy requirements, harshness of the environment, and noise pick-up are some of the factors that must be taken into account when deciding on the type of instrumentation interface to be chosen. For the water heater example we needed to measure from 32°F to just above 140°F. For this purpose, we used a thermistor from a commercial outdoor thermometer. The characteristics of this thermistor were initially not known and were determined and calibrated in the following fashion:

1) The high and low resistance values were measured for the temperature extremes that the process will cover. Range was from about 32°F to 140°F, and the thermistor resistance varied from roughly 4 KΩ to 170 KΩ.

2) The thermistor was placed in series with two resistors that were approximately in the middle of the resistance range of the thermistor.

3) A voltage divider was then created (Figure 5) and a constant DC voltage applied to the voltage divider. This voltage was chosen with two particular points in mind: a) to keep the current below 100 µA to prevent self-heating of the thermistor and b) to scale the voltage from the sensor interface so that it is as close as possible to the internal voltage range of the FPAA. It is important to scale as closely as is reasonable to the actual range of the FPAA to avoid high gain at the input that will tend to amplify the resistor and input referred noise.

4) Finally, the instrument interface was calibrated by taking voltage vs. temperature measurements at various points along the temperature span of the process in question. Interpolation was done to determine the temperature vs. voltage values between the measurement points.

5) It is worth noting, and quite convenient, that for calibration purposes the absolute resistance values of either the thermistor or the resistor in series with it are actually immaterial! All that is required is the voltage vs. temperature data. Figure 4 shows the actual instrumentation curve.

![Figure 2. Temperature (°F) vs. thermistor voltage.](image-url)

**The Control Variable Interface for a Thermal Process**

In the thermal process example (Figure 5) being used in this paper, we are using a 50-W Darlington transistor pair to apply heat to the water. The Darlington pair is in turn buffered from the FPAA output cell through two transistors. The voltage transfer function of the power interface is shown in Figure 3. This transfer function is quite linear in the mid range, but very non-linear near saturation and cut-off. Further, since there is no cooling signal to bring the temperature down, we simply turn the driver off and let the process temperature drift downward. To adjust for these non-linearities, the inverse transfer function of the load line is programmed into an FPAA look-up table. This inverse transfer function is shown in Figure 4.
Implementing Process Control Systems with Field Programmable Analog Arrays

Determine the Plant Model of a Thermal Process

Figure 5 shows the water heater. Using a step forcing function, the open-loop gain of the process, $\Delta V_{\text{out}}/\Delta V_{\text{in}}$, is measured from the graph in Figure 6 as $2.42/-1.10 = -2.20$. The process time constant is obtained by measuring the amount of time it takes from the first process movement in response to the step until the process reaches 63.2% of its final value. In this particular example, that time constant is 944 seconds. In Figure 7, we have zoomed in to determine the process lag or “dead time.” The process lag is determined by measuring the time from the forcing input until the first process movement is seen on the graph. In this particular case, the process lag is 73 seconds.

We now have sufficient information to determine the transfer function of this process:

$V_{\text{out}}(s)/V_{\text{in}}(s) = -2.1e^{-73s}/(944s+1)$

It is important to keep in mind that the process transfer function can be changed by such factors as the rate of heat loss to the environment, the amount of water in the heater, and whether thermal contact is good. As might be expected, cutting the amount of liquid being heated will cut the time constant by the same percentage, assuming the rate of heat transfer from the controller is the same.
Designing the Controller for a Thermal Process

For the thermal process in this paper, a PI controller was chosen using IMC control techniques.\(^\text{17}\) Many of the common plants, especially exponential processes, have been analyzed at great length. It is well known that a first-order lag system can be controlled with a PI controller alone. The controller was built at a block diagram level with the FPAA device and is shown in detail in Figure 8:

Demonstrate Thermal Process Response to the Controller

Figure 7. Zoom in to view the first 300 seconds of the process to determine process lag or dead time.

Figure 8. Controller for the thermal process.

\(^\text{17}\) "Internal Model Control: A Comprehensive View", October 27th 1999, Rivera, Daniel E.,
Implementing Process Control Systems with Field Programmable Analog Arrays

Figure 9 shows the actual process response to set point changes. Note that when the PV is below the SP, the OP turns off since there is no negative control output. The process simply drifts to the set point due to heat loss to the environment. This controller controlled the temperature to within 0.2°F.

In Figure 10 the sharp breaks downward are due to ice cubes being dropped in the water. One of the purposes of a PID controller is to transfer process variance from the PV to the controller output. This is illustrated by the rapid variation in the OP in Figures 9 and 10, while the PV movement stays flat.

Adaptive Process Control With an FPAA

Brief Overview of Adaptive Control

Extensive research on adaptive control began in the 1950s to aid the design of autopilot systems for high-performance aircraft, but interest gradually waned, partly because the techniques available at that time were inadequate for the complexity of the problems. Much of the theoretical groundwork for adaptive control was done in the 1960s and 1970s, and by the 1980s adaptive controllers began to appear commercially due in no small part to the marriage of theory and experimentation that advances in microelectronics permitted. Today a significant percentage of commercially available controllers offer some form of adaptation to changing process variables for disturbances.

While both an adaptive control scheme and a single-loop feedback controller attempt to compensate for disturbances and process variations, the adaptive controller differs in that it has a second control function that adjusts control parameters automatically to adapt to changing conditions. The three most popular adaptive schemes are: (1) gain scheduling or scheduled adaptive control, (2) model reference adaptive control, and (3) self-tuning controllers.

(1) **Gain scheduling (scheduled adaptive control)** is a method by which the control parameters are determined ahead of time for various process-operating conditions. The concept of gain scheduling originated in the 1950s and was applied to flight control systems, in particular to the X-15. In industrial processes, gain scheduling is often used to vary the controllers for process start-up, shutdown, and steady-state operation. Scheduled adaptive control works best when all possibilities are known in advance.

(2) **Model Reference Adaptive Control** – This approach uses a closed-loop reference model of the process to predict how the process will respond to set point changes. The reference model is compared to the actual process set point response, and this information is used to adjust control parameters. This approach works...

---

20 “Process Dynamics, Modeling and Control”, Ogunnaike and Ray,, 676
well when a good reference model is available. The thermal controller in this paper could be made adaptive by this technique.

(3) **Self tuning adaptive control** – The self tuning controller estimates online and recursively the parameters of an approximate process model. This approach is very dependent on proper experimentation and parameter estimation techniques. It can work for a wide variety of situations and does not require detailed knowledge of the possibilities or even the process model, but suffers from the risk that poor model identification can result in unstable or runaway processes.

**Adaptive Control of a Balance Beam with an FPAA**

A balance beam can be used to illustrate a method of control of an open-loop unstable system and then applying automatic adaptive control techniques to this process to illustrate how an FPAA can be used for adaptive control.

The process to be controlled (Figure 11) is a balance beam which has a dish with weights in it at one end and a door magnet at the other end that is used to keep the beam at a desired set point.

![Figure 11. Balance beam schematic.](image)

The height of the weighted end of the balance beam is sensed by using a photo diode to sense light from an LED.\(^{21}\)

Analysis of the plant shows the balance beam to be an open-loop unstable system, having a transfer function of the form\(^{22}\):

\[
\frac{K}{t^2}
\]

since the pulling force on the armature by the electromagnet will be given by

\[
F_{\text{armature}} \sim I_c/R^2
\]

where \(I_c\) is the current in the coil and \(R\) is the distance between the magnet and the armature. There will also be a constant based on the physical properties of the material and voltage to current conversion via the power circuit (Figure 12).

Since \(x(t)\) (position) = \(\int \int\) acceleration dt, and \(a = F/m\) with mass fixed, the position \(x\) will be the second integral of the voltage output times a constant. Note that the light sensed from the LED will also be a function of \(1/R^2\), but to the first order, so the dynamic effects can be ignored.

\(^{21}\) “Process Dynamics, Modeling and Control”, Ogunnaike and Ray, 617 - 620
\(^{22}\) Anadigm Balance Beam report, October 9th 2003, Dave Lovell and Nick Hunter
The following equation was implemented in the FPAA device by placing and programming various configurable analog modules:

\[ e(s)*(Kp+Ki/s+Kds)*(G)/(\lambda_1s+1)(\lambda_2s+1)(\lambda_3s+1) \]

This circuit construction process is very fast, and different circuit designs can be created and downloaded in a few minutes. This facilitates rapid test and design verification. This circuit is shown in Figure 13.

Where \( Kp, Ki \) and \( Kd \) are PID constants and \( \lambda_1, \lambda_2 \) and \( \lambda_3 \) are time constants for the filters, note that conversion from time constant, \( \lambda \), to corner frequency of a filter = \( f_o = 1/(2\pi\lambda) \).

The FPAA tool set that was used to make the controller in Figure 13 will generate C code which will allow the circuit in the FPAA to be controlled from an SPI interface. Then the control algorithm can be made adaptive by...
programming some adaptive scheme, such as auto tuning, into the C code or some other code that is linked with the C code.

To create this adaptive controller the following steps were taken\(^\text{23}\): (1) The FPAA EDA tool was used to translate the circuit into ANSI compatible C code and (2) this C code was combined with C++ code and Visual Basic code that had a user interface and the adaptive algorithm built into it. The beam is controlled by a PC. The physical set up is shown in Figure 14.

![Figure 14.](image1.png)

**Experimental Results**

Figure 15 shows a screen snap shot of the auto tuning application and the controller’s response to a disturbance after the controller has been auto tuned. The two downward blips on the screen show the PV (position) when additional weight is dropped into the balance beam cup. Notice that the beam returns to the set point within one second.

![Figure 15.](image2.png)  
**Figure 15.** Response to disturbance.

![Figure 16.](image3.png)  
**Figure 16.** Increase D until system is critically damped.

---

\(^{23}\) The code and project reports will be made available at the 2004 Embedded systems show in San Francisco.
The adaptive algorithm is programmed as follows:

1) Reset P, I and D all to zero
2) Increase P until the beam oscillates
3) Increase D until the oscillations stop
4) Increase P until the beam is at a level position
5) Pulse the plant every two seconds to calculate its damping. Increase D until critical damping is achieved (Figure 16)
6) Pulse the plant every two seconds and increase I until optimum recovery time is achieved
7) System is tuned

**Conclusion**

The FPAA is an excellent platform for designing and implementing both single-loop and adaptive single-loop controllers. The engineer can design a PID controller at a block diagram level and implement, simulate, and test each circuit in minutes without the need to do complex mathematical calculations, choose discrete components, or focus on analog circuit details. Normal device tolerances are within 1%, and the circuit parameter values are immune to changes due to aging and temperature that occur with most analog circuits. Individual parameters, such as gain, can be updated in a few microseconds using the auto-generated C code the FPAA graphical tool kit produces.

We showed two actual controllers that were implemented using the FPAA device. The first was a single-loop PI controller of a non minimum phase (NMP) thermal plant where we described the design of the instrumentation interface, showed how to use the FPAA look-up table to linearize the driver load line, and derived the plant model transfer function using approximate process model techniques. The second plant was an adaptive control PID example of an open-loop unstable balance beam plant. This plant uses a magnet to keep the balance beam level even as we add more weight. The controller, which uses the auto generated C code generated from the FPAA graphical tool set combined with additional C++ and VB code, implements an adaptive auto tuning algorithm that determines and sets the proper PID tuning constants for the balance beam PID controller. Both of these plants present difficult process dynamics, and we showed graphs of actual closed-loop response of the plants to disturbances and set point changes.

The FPAA combined with its graphical tool set offers a flexible, intuitive, and dynamic platform for developing both static and adaptive single-loop controllers. The ability to change and cycle designs in a few minutes facilitates fast development and time to market and allows easy field updates of controllers.

**Acknowledgements**

Thanks to the following individuals who helped me with both my controls education, answered many questions and helped and this paper.

1) Dr Daniel Rivera of the Department of Chemical and Materials Engineering at Arizona State University for patiently answering my seemingly never ending list of questions.
2) Nick Hunter, Dave Lovell and Simon Dickenson for their work on the balance beam demo. They designed and constructed the balance beam hardware and software.
3) Tim Streit of Anadigm for his insight into the inner workings of the Anadigm ANX2XE04 devices.

---

1 "Process Dynamics, Modeling and Control", Ogunnaike and Ray, 665