

ミックスド・シグナル制御回路におけるPIDの活用、 マイクロコントローラで アルゴリズムの柔軟な実装を 実現する

著者：Eamon Neary (eamon.neary@analog.com)

はじめに

プロセス制御を行う際には、温度などプロセスに関連する指標（制御変数）の値を、目標となるセットポイントの値と比較することになります（図1）。その結果として得られた差分は、偏差信号 $e(t)$ としてコントローラに供給されます。コントローラは、その偏差信号を使用して制御信号 $u(t)$ を生成します。その制御信号に基づいてプロセスへの物理的な入力（操作変数）を操作することにより、偏差が安定的に低減されるように制御変数の値を変化させます。

上記のような制御を行うために一般的に用いられるのがPID（Proportional-Integral-Derivative：比例・積分・微分）コントローラです。同コントローラは、偏差から算出される3つの項の和を求めます。3つの項とは、シンプルなゲインの項（比例項）、偏差の積分値に比例する項（積分項）、偏差信号の変化率に比例する項（微分項）のことです。クローズド・ループにおいて、比例項には、瞬時値に比例する形で偏差を低減する効果があります。累積偏差である積分項は、偏差を緩やかにゼロに近づける効果をもたらします（また、保存された偏差により、偏差がゼロを通り過ぎて負の値になるまで駆動しようとする傾向を示します）。微分項は、偏差の変化率を使用して将来の値を予測することにより、比例項に対する応答を高速化する役割を果たします。加えて、微分項は積分項の遅れを補償することにより、ループの安定性を高める効果ももたらします。

これら3つの項を組み合わせれば、非常に正確で安定した制御を実現できます。但し、制御項は、特定のシステムにおいて最適な動作が得られるよう個別に調整（チューニング）する必要があります。プロセスによっては、遅れが生じる部分が多く、応答に大きな遅延が発生することがあります。その種のプロセスの制御は容易ではありません。そのため、シンプルなPIDコントローラが最も適しているのは、操作変数の変化に直ちに反応するプロセスだと言えます（多くの場合、プロセスに印加されるエネルギーの量または流量の制御に利用されます）。PID制御は、負荷が絶えず変化するシステムにとって有用な手法です。PIDコントローラは、セットポイントの頻繁な変更や、（周辺の条件や負荷の変化に伴う）制御変数の目標値からのずれに自動的に対応できるものでなければなりません。

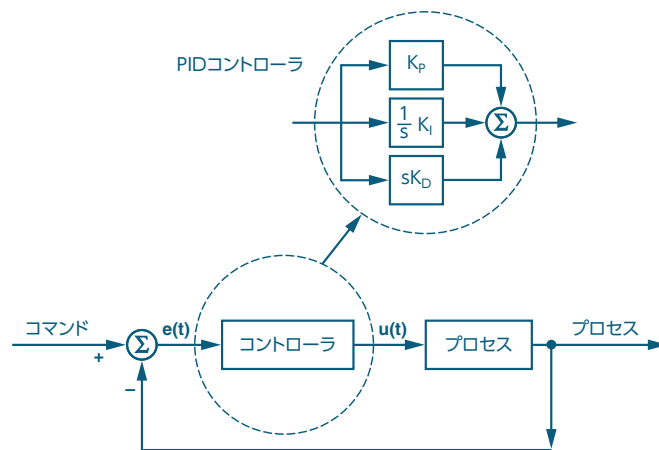


図1. PID制御を実現するループ

通常、低速なプロセス向けのPIDコントローラのパラメータは、速度を高めたシステムのモデルを使用することによって最初に取り得します。制御に向けた高度な戦略は数多く存在しますが、PIDコントローラは標準的なものと言えます。長年にわたる使用実績があり、十分に理解された産業用のコンポーネントであるため、非常に多くの産業用制御システムで使用されています。また、プロセスというのは不安定なものです。そのため、より洗練された制御機構の方が効果的であるとは限りません。特定のプロセスに対して適切にチューニングされたPIDコントローラは、優れた選択肢になり得ます。

以下、PIDの各項についてより詳しく説明しましょう。

比例制御

比例制御では、偏差に比例する補正項が適用されます。比例定数 K_p は、コントローラの比例ゲインとして知られています。ゲインが高くなると、システムはセットポイントの変化に対して高速に応答し、最終的な偏差（定常偏差）は小さくなります。但し、大きな減衰が生じることから、システムの安定性は低下します。ゲインが更に高くなると、オーバーシュートやリングングが生じ、最終的には減衰を伴わない発振に至ります。

積分制御

比例制御を利用すれば、偏差を大きく低減することができます。しかし、偏差をゼロにすることはできません。偏差をゼロにしたい場合には、制御関数に積分項を追加します。クローズド・ループにおいて、積分器は自身に入力される信号の平均レベルをゼロに保たなければなりません（さもなければ、その出力は無限に増加し、最終的には飽和するか、それ以上に悪い状態になります）。積分ゲインの定数 K_i が大きいほど、偏差は変化に応答して素早くゼロに近づきます（その後ゼロを通り過ぎて負の値になります）。 K_i を大きくしすぎると、発振や不安定な状態に陥る可能性があります。

微分制御

偏差の信号の時間微分（変化率）に比例する微分項を制御関数に追加すると、安定性が高まり、比例項／積分項を高いゲインで使用する場合に生じるオーバーシュートを低減することができます。また、偏差の変化を予測することによって、応答速度を高めることが可能になります。通常、そのゲインである減衰定数 K_D は、セットポイントまたは制御変数の変化に対する臨界減衰応答が得られるように調整することができます。減衰量が小さすぎると、比例制御によるオーバーシュートが残存してしまう可能性があります。一方、減衰量が大きすぎると応答が無意味に遅くなるおそれがあります。偏差の信号に現れる高周波のノイズが、微分器によって増幅されることにも注意を払わなければなりません。

ここで、上述した内容をまとめます。まず、比例（P）コントローラを利用することで立上がり時間が短くなり、定常偏差を低減することができます。但し、その偏差をゼロにすることはできません。一方、比例・積分（PI）コントローラを使用すれば、定常偏差をゼロにすることができます。但し、過渡応答が悪化してしまう可能性があります。PIDコントローラを採用すれば、システムの安定性を高め、オーバーシュートを低減し、過渡応答を改善することができます。制御ループのシステムにおいて、各項を大きく設定した場合の影響を表1にまとめました。

表 1. 制御ループにおける各項の影響

ゲインの定数	立上がり時間	オーバーシュート	セトリング時間	定常偏差
K_P	減少	増加	ほとんど変化なし	減少
K_I	減少	増加	増加	除去
K_D	ほとんど変化なし	減少	減少	ほとんど変化なし

3つの項の和は、以下の式で表されます。

$$u(t) = K_P e(t) + K_I \int e(t) dt + K_D \frac{de(t)}{dt}$$

これに対応する伝達関数は、以下のようになります。

$$K_P + \frac{K_I}{s} + K_D s = \frac{K_D s^2 + K_P s + K_I}{s}$$

図1のシステムにおいて、セットポイントの値と実際の出力の差が偏差信号 $e(t)$ です。この偏差信号がPIDコントローラに引き渡されます。それを受けて、PIDコントローラは、この偏差信号の微分値と積分値を計算します。3つの係数を代入して上記の和を求め、信号 $u(t)$ を生成します。

デジタル方式のPID制御

PIDのアルゴリズムは、産業分野のプロセス制御で広く用いられています。100年近く前から広く認められて採用されているものであり、もともとは空気圧の制御器に適用されていました。1940年代～1950年代には、アナログ・コンピュータによる制御システムの設計において、PID制御をモデル化するものとして電子回路が使われました。その後、PIDによるプロセス制御は、まずはアナログ・コントローラの形で、続いてデジタル・コントローラの形で広く使われるようになりました。例えば、8ビットのマイクロコントローラを使用してPIDのアルゴリズムを実行するといった具合です。その処理を実現するためのソフトウェアについては、多くの文献が公開されています。

以下では、まずデジタル方式のPIDコントローラを構成する基本的な要素について説明します。続いて、データ・アキュイジション用のSoC（System on Chip）であるMicroConverter[®]製品を紹介し、それらの製品を採用することにより、高いコスト効率でプロセス制御を実現することができます。

PIDによる制御ループは、例えば温度を狭い範囲で正確に維持する必要がある空調システムや冷蔵システムに適用されます。その場合、サーモスタットによるオン／オフ制御ではなく、連続的な監視と制御を実現できます。図2に示したのは、そうした制御システムの構成例です。このシステムでは、ファンの速度を連続的に調整し、低温源からの空気の流れを増減させることによって温度を一定に保ちます。

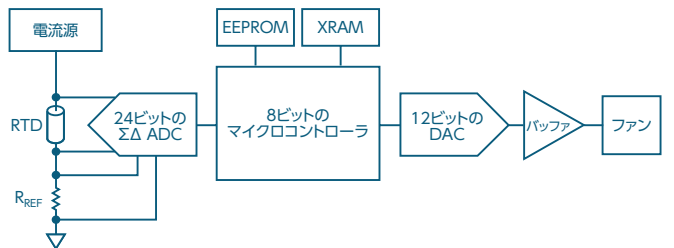


図 2. PID ベースの温度制御／換気システム。PID コントローラをディスクリート構成で実現する場合の例です。

このシステムに求められる働きは、ユーザが設定したセットポイントにできるだけ近い値に室温を保つことです。そのためには、室温を正確に測定した上で、それを補正できるようファンの速度を調整する必要があります。

図2のシステムでは、精度の高い電流源から抵抗性の温度センサー（サーミスタやRTD）に対して電流が供給されます。温度センサーは、目標となる温度に対応する値のリファレンス抵抗と直列に接続されています。A/Dコンバータ（ADC）は、リファレンス電圧とサーミスタの電圧の差を検出し、それをデジタル値に変換します。この値は温度の偏差を表します。8ビットのマイクロコントローラは、ADCから引き渡されたデータの処理と、PIDコントローラの機能を実現する役割を果たします。このマイクロコントローラにより、D/Aコンバータ（DAC）を介して駆動されるファンの速度が調整されます。マイクロコントローラは、外付けのプログラム・メモリとRAMを利用してプログラムを実行します。

比例制御 (P) のみを適用する場合、ファンの動作速度は、セットポイントに対する温度差と直接関連することになります。その場合、上で説明したとおり定常偏差をゼロにすることはできません。

積分項を追加してPI制御を行うようにすると、ファンの速度は周辺の温度に応じて上昇／下降します。周辺の温度は日中に上昇し、夕方には低下します。室温は、それらに伴う偏差を補正する形で調整されます。積分項を追加することによって、オフセット（定常偏差）は除去されます。但し、積分ゲインが大きすぎると、セットポイントを中心とする発振が生じるおそれがあります（なお、サーモスタットをオン／オフするタイプの温度制御システムでは必ず発振が生じます）。

上記の発振のおそれは、微分項を追加してPID制御を行うことでかなり抑制できます。微分項は、セットポイントからの偏差の変化率に応答します。扉や窓を一時的に開けた場合には温度が急に変化します。微分項は、そのような変化に対して素早く補正を実施できるようシステムを導きます。

このシステムは、SoCを使用することで、図3のように簡素化することができます。その結果、部品のコスト、アセンブリのコスト、基板上の面積を最小限に抑えることが可能になります。

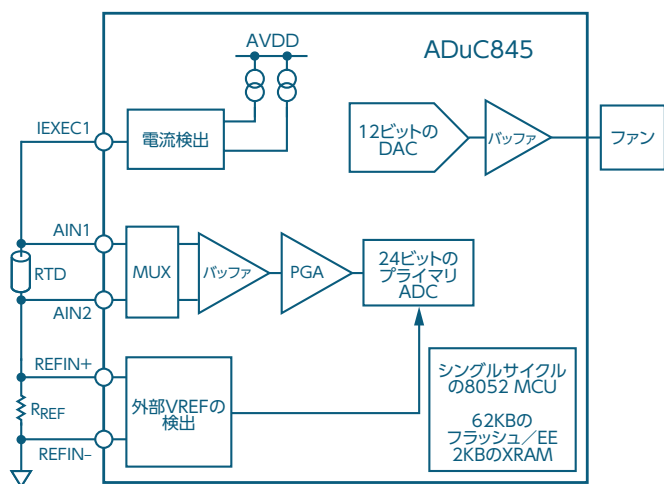


図3. SoCを利用した実装

図3では、SoCとしてMicroConverterである「ADuC845」を使用しています。同ICは、62Kバイトのフラッシュ／EEプログラム・メモリ、4Kバイトのフラッシュ・データ・メモリ、2KバイトのRAMを内蔵しています。フラッシュ・データ・メモリには、PID制御ループのチューニング済みの係数を保存することができます。また、シングルサイクルのコアは、PID制御ループの処理と一般的なタスクを同時に実行できるだけの十分な能力を備えています。

MicroConverter製品としては、分解能が12～24ビットのADCを集積したものが提供されています。24ビットのシグマ・デルタ ($\Sigma\Delta$) 型ADCを搭載するADuC845は、0.1℃の精度で温度を維持する必要があるシステムに最適な製品です。

PID制御ループが適したもう1つのアプリケーションとしては、セットポイント（サーボ）方式のモータ・コントロールが挙げられます（図4）。この種のアプリケーションでは、ユーザの入力によって定義された角度位置にモータを移動／維持／追従させる必要があります。図4の例では、ポテンショメータ (POT) の回転がユーザによる入力に相当します。

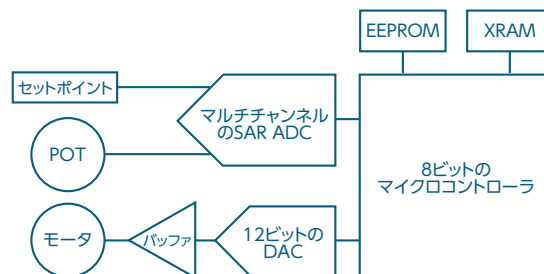


図4. ディスクリット構成のモータ・コントロール・システム

この種のシステムは、多くの部品を使ってディスクリット構成で実装することができます。ただ、ICベースのソリューションを採用すれば、先ほどの例と同様に、よりシンプルな実装を実現することが可能になります。図5に示したのは、MicroConverter製品を使用して構成したデモ用のシステムです。基板上に実装された回路により、セットポイントの入力となるポテンショメータの回転に追従するようポイントが操作されます。

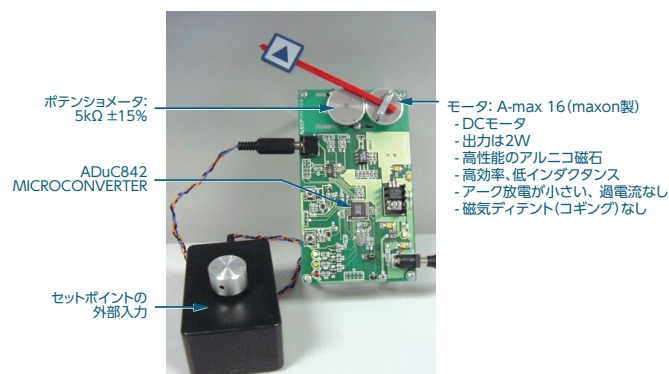


図5. SoCで構成したモータ・コントロール・システムの外観

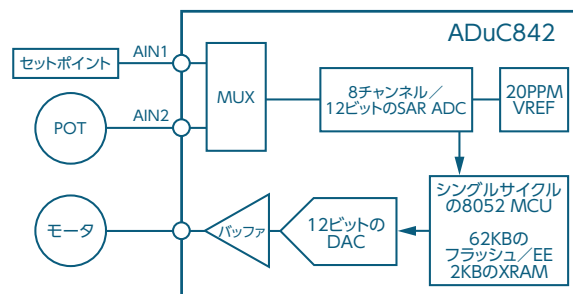


図6. SoCを利用した実装

図5のシステムでは、MicroConverter製品として「ADuC842」を採用しています。同ICには、図6に示した各種のブロックが集積されています。そのため、部品のコストとアセンブリのコストを抑えられます。また、演算用の電子回路の占有面積は小さく、信頼性は高くなります。図6を見れば、SoCを使用することでシステムのハードウェアが大幅に簡素化されることがわかるはずです。

デモ用システムの基板には、ADuC842以外に、ポテンショメータ用のバッファ・アンプ、モータを駆動する出力パワー・アンプ、低消費電力の電子回路に電力を供給する低ノイズのレギュレータ (5V)、より堅牢性の高いモータ用のレギュレータ (5V、ヒートシンク付き) が実装されています。その他にも、状態表示用のLED、リセット用ボタン、シリアル・データのダウンロード用ボタン、複数の受動部品が実装されています。

このシステムについては、PC上で実行可能なソフトウェアを用いて、PI制御に関するシミュレーションを実施しました。図7に示したのは、異なるチューニング・レベルに対するシステムの応答です。これを見れば、積分項がいかに重要であるかがわかります。

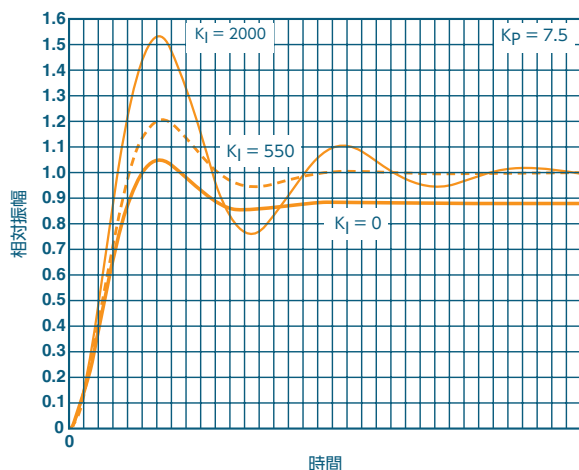


図7. PI制御の応答。3種類の積分項を使用したシミュレーション結果を示しています。K_Iが0の場合、1.0に対してオフセットが生じています。また、K_Iが2000の場合には軽微な減衰／発振が発生しかねないことがわかります。K_Iが550であれば、発振はほぼ抑えられます。

図8に示したのは、微分 (D) を加えた完全なPID制御ループの応答例です。システム全体のステップ応答が明らかに良化していることがわかります。ご覧のとおり、高速かつ正確で、適切に減衰する応答が得られています。オフセット、発振、オーバーシュートは生じていません。

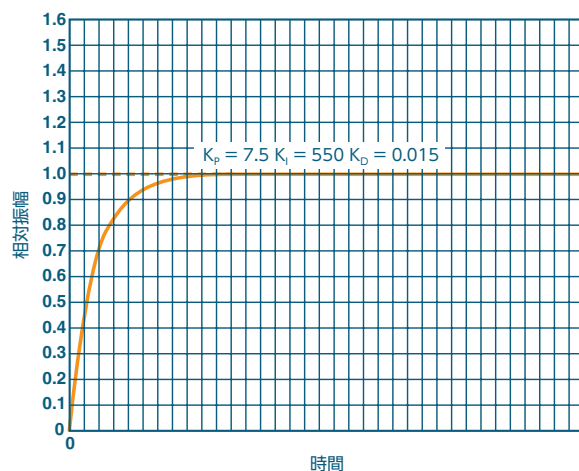


図8. PID制御を適用した場合の応答