

## MCP3551 デルタシグマ ADC の応用設計法

Author: Craig L. King  
Microchip Technology Inc.

### まえがき

MCP3551 デルタ - シグマ ADC は高分解能のコンバータです。このアプリケーションノートでは、このデバイスを使うときの、いろいろな設計テクニックについて論じています。まず典型的な応用回路について論じ、ノイズ解析の項が続きます。このデバイスは、他の高分解能の ADC と同じように、ノイズ電圧より小さな LSB サイズとなっています。このため、デバイスの（そしてシステムの）性能は、単にバイナリ出力列を見ているだけでは解析できません。データを集め、視覚的に結果を解析することが必要です。そのためには、回路設計をする時には、PC にデータを取り出す方法を用意しておくことが重要です。本アプリケーションノートでは、センサーやシステム性能評価を素早くするために、MCP3551 22-Bit Delta-Sigma ADC PICtail™ Demo Board の回路と DataView® ソフトウェアの使い方を説明し、同様に、デバイスを PICmicro® マイクロコントローラにインターフェースする方法を説明します。

DataView ソフトウェアは、ここで論ずる多くの問題に適切なヒストグラムや領域描画グラフにより、リアルタイムでシステムノイズ特性を視覚化して評価できるようにするものです。

アンチエイリアスフィルタの設計と入力セトリング時間の問題の項も含まれています。PICmicro マイクロコントローラ用のソフトウェアとハードウェア SPI™ 用のシリアル通信ファームウェアが、C とアセンブラで記述されて提供されています。C 言語で書かれたソフトウェア SPI™ のコードは、MCP3551 22-Bit Delta-Sigma ADC PICtail™ Demo Board の実行コードとしても提供されています。

### 標準的な接続

システムノイズ解析とデバッグ用として、MCP3551 22-Bit Delta-Sigma ADC PICtail™ Demo Board にセンサーを接続した MCP3551 デバイスの標準的な応用を図 1 に示します。

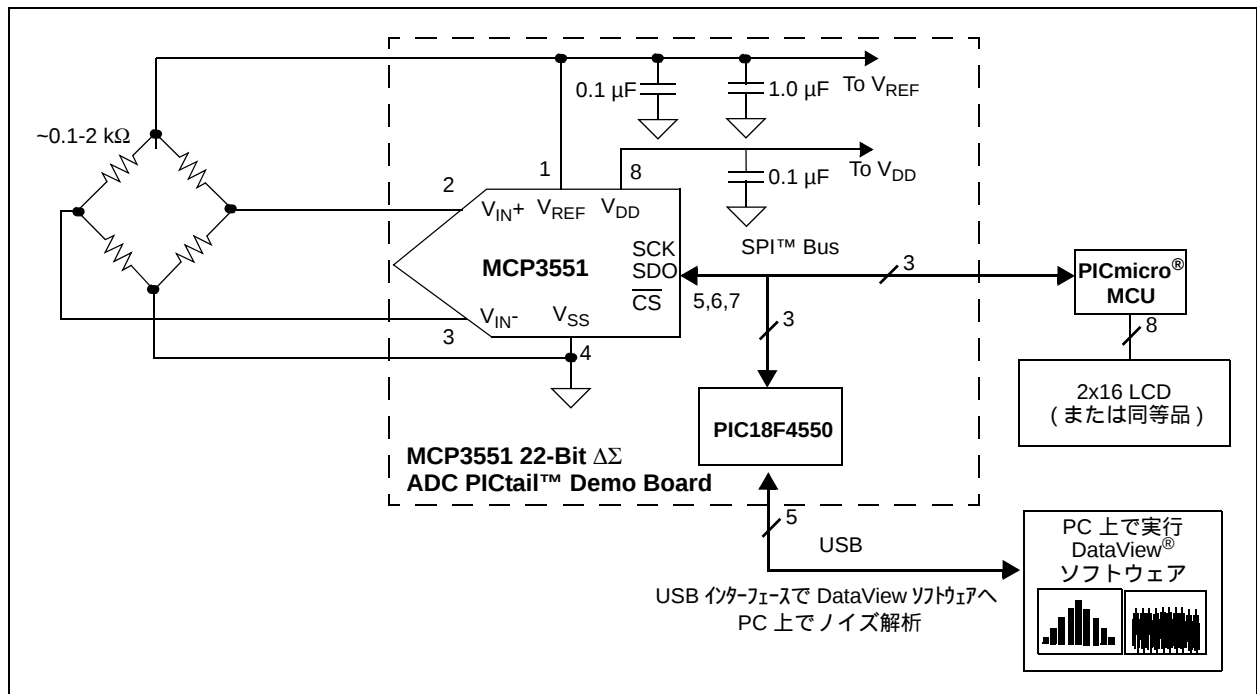


図 1: システムノイズとデバッグ用の典型的なブリッジセンサーの接続図

温度、圧力、重量などの物理的状態のセンサーは、その多くが図 1 のようなホイートストーンブリッジで構成されています。ブリッジは、1 から 4 の要素のどれでも物理的状態に感応するものにでき、できれば放射状の構成として、センサーと ADC リファレンス電圧の両方をシステムリファレンスでドライブするようにします。ひとつの例は General Electric 社の NovaSensor<sup>®</sup> という絶対圧力センサー (NPP-301) で、図 2 のように 4 要素の可変ブリッジとなっています。

MCP3551 ADC で設計するときには、まず最初のステップは、センサー性能の評価をして、システム全体分解能を上げるためにはどのようなステップにするか (もしあれば) を決定することです。多くの場合、MCP3551 デバイスで直接センサー出力をデジタル化する使い方ができ、他の外付けの信号補正回路を必要としません。

NPP-301 デバイスは、3V の電池で使ったとき、標準で 60 mV のフルスケール出力となっています。このデバイスの圧力範囲は 100 kPa となっています。MCP3551 の出力ノイズ規格は 2.5  $\mu\text{V}_{\text{RMS}}$  です。

下記式は圧力 (P、単位 : パスカル) と高度 (h、単位 : メートル) の関係の一次近似式です。

$$\log(P) \approx 5 - \frac{h}{15500}$$

フルスケールレンジが 60 mV で解像度が 2.5  $\mu\text{V}$  で使くと、直接デジタル化 (メートルで) したときの結果の解像度は 0.64 メートルとなり、およそ 2 フィートとなります。

この使い方は議論の例題として使っただけであることに注意して下さい。すなわち、最終的なシステム設計のときは、温度効果や一次近似による誤差を含めなければなりません。

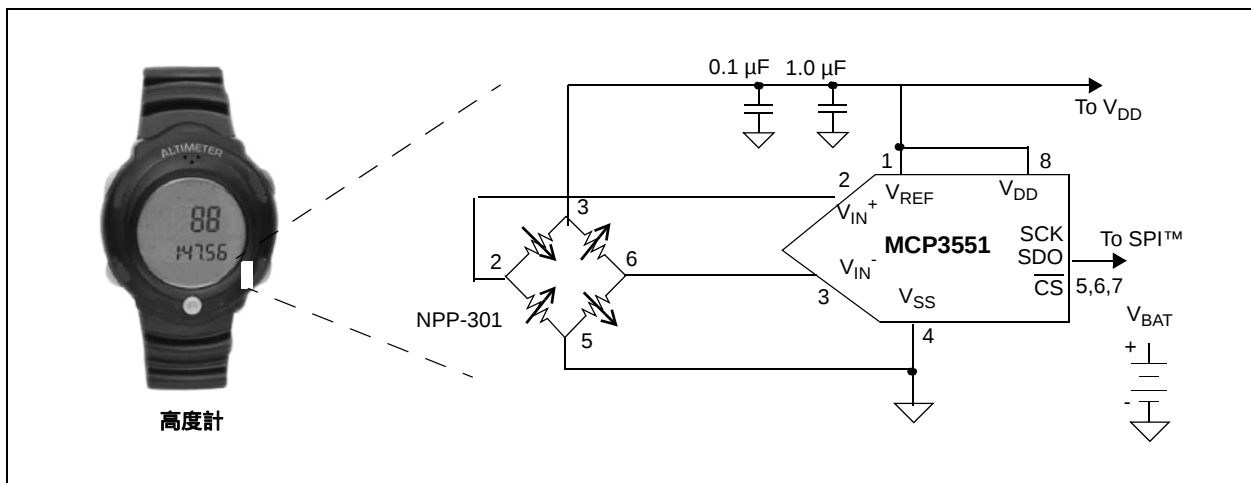


図 2: 直接デジタル化の例題。これは低電力の絶対圧力センサーモジュールで、GE 社の NovaSensor (NPP-301) シリーズで使われている低価格の表面実装の圧力センサーです。

MCP3551 のような高分解能 ADC は、低分解能 ADC と増幅段を含めた置き換えに使うことができます。図 3 のシステムブロック図は、典型的な信号処理回路です。この例では、要求精度は 12 ビットです。12 ビット ADC が選択され、変換前に信号増幅のため増幅段が必要となっていました。12 ビット精度を達成するため、ADC の全入力範囲を使わなければなりません。この例では、信号のコモンモードが変化するため、何らかのオフセット補正が必要で、加算アンプが必要になります。(つまり、信号は増幅前にセンター配置されなければなりません。)

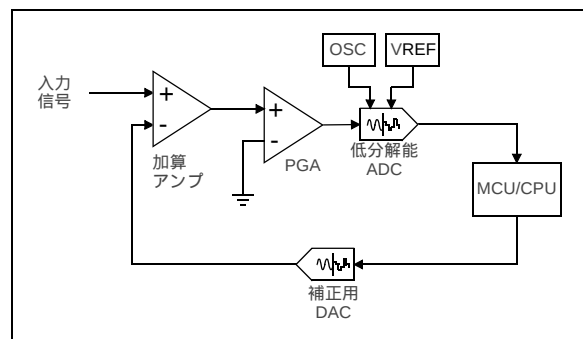


図 3: 低分解能 ADC と信号処理回路を使った応用例

この場合には、高分解能のMCP3551デバイスを使って信号処理回路全部を省略することができます。

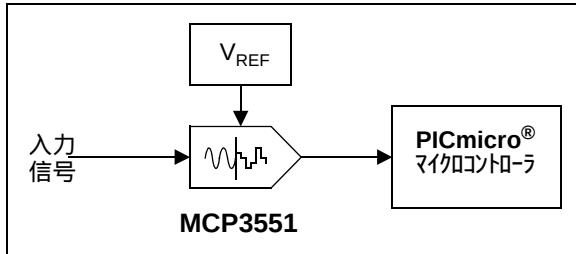


図 4: 高分解能 ADC を使って信号処理回路を省略する

高分解能 ADC の広いダイナミックレンジ (つまり MCP3551 の場合は 22 ビット) により増幅システムを省略することができます。上記の例では、12 ビット精度が要求でした。22 ビットのダイナミックレンジでは、ADC の入力レンジ内のどこにでも 12 ビット精度の範囲を置くことができます。図 5 は、 $V_{REF} = 2.5V$  (注: スケールは合わせてはいない) でこの比較を示しています。

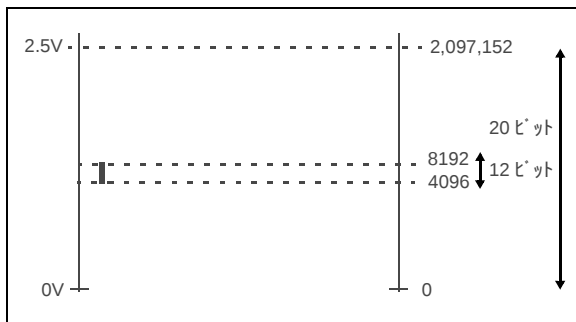


図 5: MCP3551/3 の広いダイナミックレンジと 12-bit ADC との比較 MCP3551/3

## ビットとノイズの解析

高分解能のコンバータでは、LSB の大きさはデバイスノイズより小さくなっています。(つまりデバイスからの出力コードには常にばらつきがあるということです。)この出力ノイズ規格は、出力コードの分布から計算で測定できます。出力コードの分布は、デバイスの有効解像度または等価ビット数 (ENOB) を定義します。出力コードの分布は、ある標準偏差を持っています。この標準偏差はデバイスの RMS ノイズ ( $\sigma$ ) です。RMS ノイズ (測定可能な最小の信号) とデバイスのフルスケール入力レンジ (測定できる最大信号) の比が、ADC の有効分解能となります。これを式 1 の定義で、2 進数の ENOB に変換します:

式 1:

$$ENOB = \frac{\ln(FSR / RMS \text{ Noise})}{\ln(2)}$$

注意すべきは、式 1 による ENOB (または有効分解能) の式は、純粋な DC 信号を前提としていることです。正弦波信号は、同じピークレベルに単調に広がったランダム信号より、AC 電力が 1.76 dB だけ高くなります。もし応用が AC 信号を扱うなら、ADC 性能は AC FFT を使って周波数領域でみることになります。これらの描画は、信号対ノイズ比 (SNR) または、信号対 (ノイズ + 歪) 比 (SINAD) を表します。しかし、これらは低バンド幅のデルタシグマのデータシートには通常ありません。

ENOB は本来、高 AC 入力の際に比べ、高 DC 入力の方が優れています。なぜなら AC 入力の際には、信号に不確定に加わる信号の位相が  $90^\circ$  に近くなると値が 0 に近くなってしまうからです。

標準 SNR (dB)  $= 6.02n + 1.76$  ( $V_{RMS} = V_{PEAK} / 2\sqrt{2}$ ) から求めるか、信号を純粋な正弦波とする) を使って ENOB を計算するには、式 2 を使います。計算結果の ENOB は、1.76 dB または 0.292 ビットだけ小さな ENOB となります。

式 2:

$$ER \text{ in bits rms} = \frac{20 \cdot \log\left(\frac{FSR}{RMS \text{ Noise}}\right)}{6.02}$$

100 mV フルスケール出力のセンサーを使ったとき、MCP3551 分解能ベースの ENOB は次の式で計算できます:

式 3:

$$ENOB = \frac{\ln((100mV)/(2.5\mu V))}{\ln(2)}$$

ここで:

$$ENOB = 15.3 \text{ bits}$$

MCP3551 の出力ノイズまたは有効分解能は、 $V_{REF} = 5V$  で 21.9 ビット RMS という規格です。予測されるピークノイズビット数 (ふらつかないビット数) は統計的解析で求められますが、これは次項で論じます。

注意すべきは、ADC の  $V_{REF}$  電圧を低くしても、デバイスの出力ノイズあるいは有効分解能は改善されないということです。これはノイズが入力構成品の熱雑音で占められているからです。

応用によっては、必要なシステム分解能を達成するためには信号増幅が必要になることもあります。本アプリケーションノートには、このような使い方で必要な信号処理回路の解析は含まれていません。

センサーそして最終的にはシステムの分解能を決定するには、全ての誤差を考慮しなければなりません。大部分の誤差の補正は使い方に依存しています。例えば、誤差規格が 0.01% の電池を考えてみて下さい。

補正をしないと、100 mV フルスケール出力のセンサーに対して MCP3551 の分解能が大変細かいのにもかかわらず、センサーがシステム全体の分解能を 13.2 ビットに抑えてしまいます。

定義によれば、ノイズは波形や姿形がない非周期性の信号です。このランダム性は、ガウス分布（正規分布）の RMS 測定のような統計的特性を扱うには最適です。システムを設計していて性能を計測したいときには、RMS ノイズの方がピークピークノイズより再現性が高くなります。図 6 は、2 つの異なる ADC 出力の分布を RMS と PEAK という異なる値で表したものです。

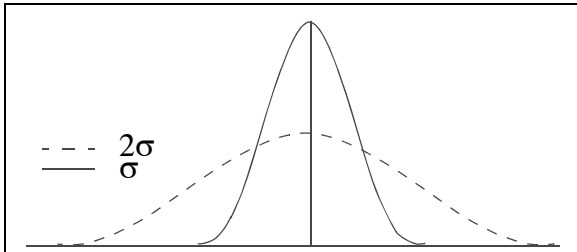


図 6: 2 つの出力の正規（ガウス）分布

DataView<sup>®</sup>ソフトウェアツールは、MCP3551 を使って、リアルタイムのヒストグラムを表示する視覚化ツールです。ソフトウェアは、その時の分布の RMS ノイズを計算します。さらに、分布に含まれるサンプル数を、平均の経験則として知られている方法で数値化し表示します。

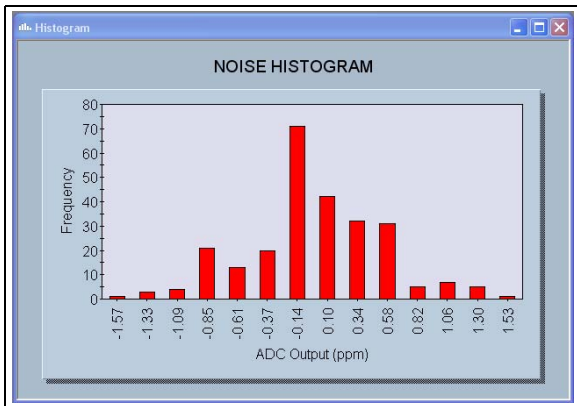


図 7: DataView<sup>®</sup>ソフトウェアは、システム性能をヒストグラム形式で表示する

前述の例では、RMS ノイズは 0.8 ppm でリファレンス電圧は 2.5V でした。このシステムでは、式 1 を使って ENOB は 21.6 となりました。

ソフトウェアはまた、領域描画ウィンドウを使って時間ベースのシステム解析を行うこともできます。この視覚化解析ツールにより、システムドリフトなどの時間ベースの誤差を解析することができます。

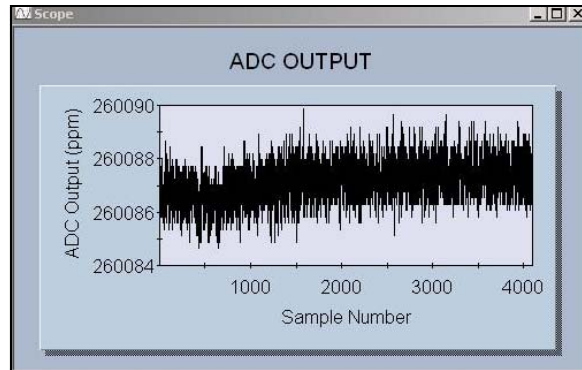


図 8: DataView ソフトウェアの領域描画表示

## デバッグポーリングとデータログ

DataView ソフトウェアは、USB のポーリング間隔をミリ秒から時間という広い範囲で変えられるという柔軟性があります。長期間のデータ解析を必要とする応用では、長期間の性能を見ることができるようシステムキャッシュを設定できます。DataView ソフトウェアの USB ポーリング間隔を変えることで、設計者は、高分解能システムに典型的な、長期変動というシステム問題を簡単に調べることができます。（図 9 参照）この特徴のさらなる情報については MCP3551 22-Bit Delta-Sigma ADC PICtail<sup>™</sup> Demo Board User's Guide (DS51579) を参照して下さい。

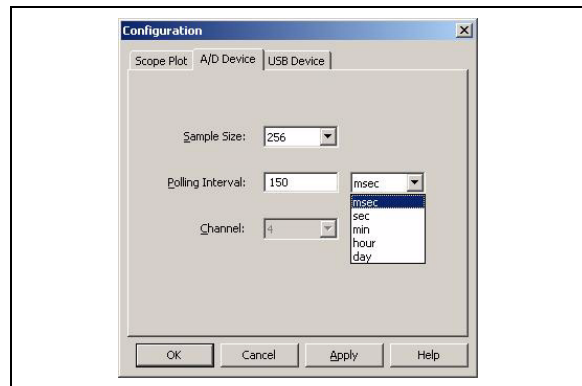


図 9: システム変動解析用の USB ポーリング間隔制御

## ピークノイズ予測と無雑音ビット

ピークピークノイズを計測したり予測することは、RMS ノイズを計測することよりかなり難しいことです。また、このピークピークノイズは、無雑音ビットまたはふらつきなしビットともいわれます。ここで、分布の中で発生する出力コードの可能性を予想してみましょう。分布が正規分布あるいはガウス分布（ノイズは完全にランダムと仮定）に従うという事実に基づいて、標準統計表を用いて表 1 が作成できます。最初の欄の乗数は、ピーク-RMS の比となります。この乗数（または比）は、信号の電力成分を解析するときの信号の波高因子としても知られています。しかし、ノイズを解析するときには、この乗数は、各用途の要求に基づいて選択すべきです。

システムのデバッグ中は、正しいピークピークウィンドウを求めるのに、統計学の経験則が一般的規則として適用できます。経験則によれば、正規分布に従うデータの 68% が平均の 1 標準偏差以内に収まり、95% が 2 標準偏差内に、99.7% が 3 標準偏差内に収まるとされています。デジタルシステム設計では、もっとも一般的な選択は平均の 3.3 標準偏差で 99.9% の確率です。この厳密さより厳しくするかゆるくしたシステム設計は、表 1 の他の RMS- ピーク比が波高因子を参照して下さい。

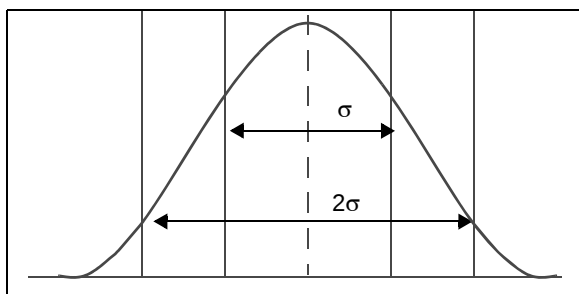


図 10: 平均からの  $n \sigma$  (標準偏差)、表 1 に基づく

例として、やや高信頼の無雑音ビットを要する応用を選択したとします。(例えば心不全用の電子式心臓細動除去器のフィードバックループなど)

DataView ソフトウェアを使って求めたシステムノイズは 0.5 ppm, RMS となります。

$$\begin{aligned}
 e_N(RMS) &= 0.5 \text{ ppm} \\
 e_N(p-p) &= e_N(RMS) \cdot K \\
 &= (0.5 \text{ ppm}) \cdot (10) \\
 &= 5.0 \text{ ppm} \\
 1 \text{ ppm} &= \frac{2^N \text{ codes}}{1000000} = \frac{2^{22}}{1000000} \\
 &= 4.194 \text{ codes} \\
 e_N(p-p) &= 5 \text{ ppm} = 20.97 \text{ codes} \\
 &= 21 \text{ codes}
 \end{aligned}$$

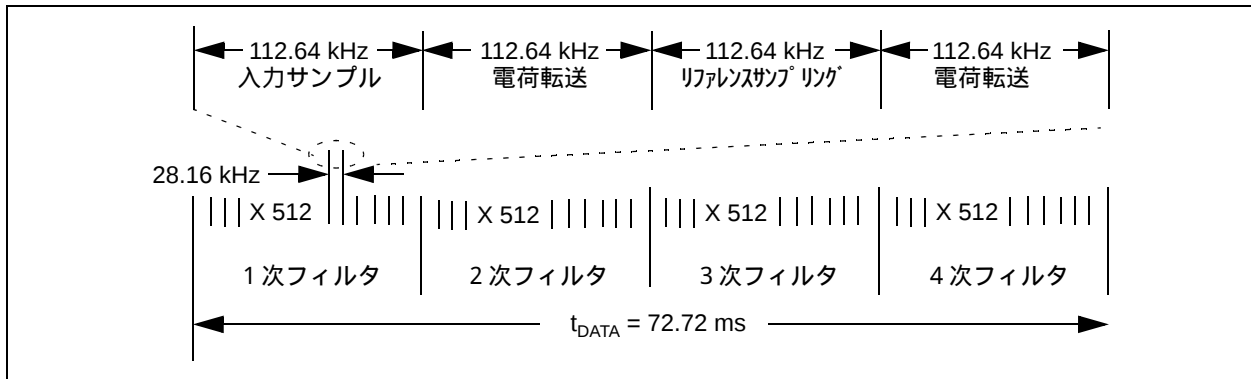
ここで、乗数は 5 を選択してやや控えめとし、結果ウィンドウは、21 個の出力コード幅をもっています。

表 1: 無雑音ビット予測用の信頼表

| 平均周辺の分布ウィンドウ<br>(ピークピークウィンドウ)                | ウィンドウ内に入る出力コードの確率 | ウィンドウ内に入らない出力コードの確率 |
|----------------------------------------------|-------------------|---------------------|
| 2.0 x RMS または 1 $\sigma$                     | 68%               | 32%                 |
| 3.0 x RMS または 1.5 $\sigma$                   | 87%               | 13%                 |
| 4.0 x RMS または 2 $\sigma$                     | 95.4%             | 4.6%                |
| 5.0 x RMS または 2.5 $\sigma$                   | 98.8%             | 1.2%                |
| 6.0 x RMS または 3 $\sigma$                     | 99.73%            | 0.27%               |
| <b>6.6 x RMS または 3.3 <math>\sigma</math></b> | <b>99.9%</b>      | <b>0.10%</b>        |
| 8.0 x RMS または 4 $\sigma$                     | 99.954%           | 0.046%              |
| 10.0 x RMS または 5 $\sigma$                    | 99.994%           | 0.006%              |

## アナログフロントエンド (AFE) の設計

アンチエイリアシングとセトリング時間の問題を見る前に、入力構成とデバイスの動作をまず評価しなければなりません。MCP3551 デバイスの入力ピンは、スイッチキャパシタタイプの入力です。入力ピンは直接デルタシグマ変調器に接続されています。ここでは内部発振器を 4 で割った周波数 ( $f_{INT}/4$ ) で、入力をオーバーサンプリングしています。結果はリファレンスと入力間で 4 相サンプリング方式となります。サンプリング時間  $t_{CONV}$  の間に、ADC は常時差動入力電圧とリファレンス電圧を比較し、この電荷を入力コンデンサに転送します。図 11 は MCP3551 デバイスを使ったときの、このタイミングの説明図です。



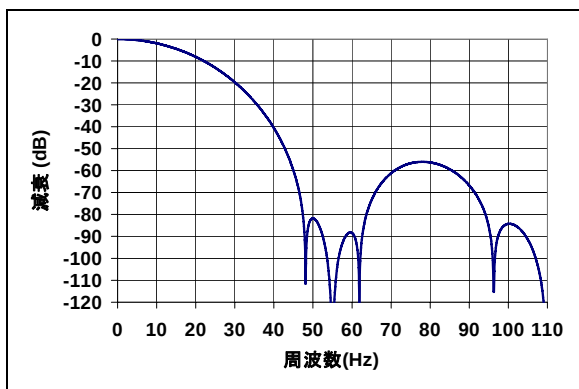
**図 11:** MCP3551 デバイスの内部タイミング。セトリング時間問題には電荷転送周波数の注視が必要。エイリアス問題には、28.16 kHz というオーバーサンプリング周波数に注視が必要。

## アンチエイリアスフィルタの設計

ADC のアーキテクチャにかかわらず、アンチエイリアスフィルタがしばしば必要とされます。デルタシグマ ADC も例外ではありません。図 12 の SINC フィルタ応答に基づけば、簡単で安価な RC フィルタだけで、オーバーサンプリング周波数付近の不必要な信号を取り除くには十分です。

MCP3551 デバイスは 28.16 kHz のオーバーサンプリング周波数となっています。MCP3553 デバイスは 30.72 kHz というオーバーサンプリング周波数となっていて、より高いデータレートまたはナイキスト周波数とするため、低いオーバーサンプリング比 (OSR) となっています。MCP3551 と MCP3553 デバイスのナイキストあるいは出力データレートは、それぞれ 13.75 Hz と 60 Hz です。

MCP3551 の SINC フィルタ応答は、図 12 のように周波数が上がると減衰が大きくなることを持っています。アンチエイリアスフィルタへの要求は、SINC フィルタの減衰特性を意識して選択する必要があります。



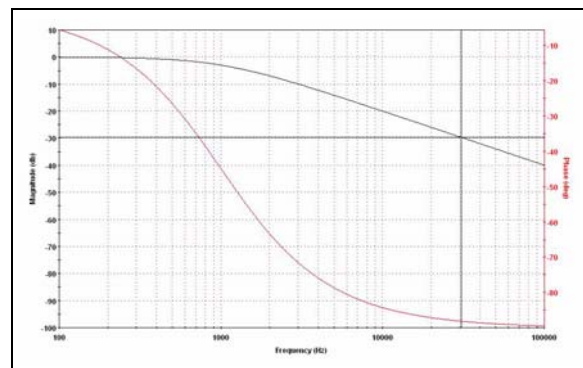
**図 12:** MCP3551 の改変型 SINC フィルタ

注意すべきは、不適切な部品を使うとフルスケールにならず、普通はより小さな振幅となることです。図 12 から、最大の SINC こぶはおよそ 60 dB 下がっ

ています。(エイリアス部では -20 dB です) 従って、アンチエイリアシングフィルタでさらに 20 dB を加えれば 100 dB を得られます。

マイクロチップのフリーの FilterLab<sup>®</sup> フィルタ設計ツールを使えば、簡単に単極の指定フィルタカットオフ周波数の RC 減衰器を評価でき、またエイリアスされた信号周波数成分の評価もできます。図 13 は、カットオフ周波数 1 kHz で、サンプリング周波数 30.72 kHz において 30 dB 以上として設計された RC を示しています。

注意すべきは、サンプリング周波数の整数倍で、SINC フィルタの応答が繰り返され、そこでは SINC フィルタ応答はゼロになることです。



**図 13:** FilterLab<sup>®</sup> フィルタ設計ツールで RC 応答を表示

## 入力インピーダンス

図 11 に示したようにデバイスの入力におけるスイッチング周波数は、内部発振器の周波数にすべてのフェーズで等しくなっています。入力ピンの抵抗は、スイッチング周波数に容量と等価容量 ( $C_{EQ}$ ) を乗じることで計算できます。結果の RC は、デバイスへ入力するときに必要とされるセトリング時間を決定します。

入力に何らかの RC が追加されると、デバイスの内部オーバーサンプリング内に入力信号が完全にセトリングされなくなる原因となります。ここで重要なことは、デルタシグマ構成で実行されるオーバーサンプリ

ングと平均化によって、ここに追加された追加 RC は毎回のオーバーサンプルされる電荷にかかわることで、結果のデバイス出力への影響は、変換オフセットとゲインのエラーになります。デバイスの直線性は悪くはありません。出力ノイズ特性も、入力抵抗によって追加される熱雑音が出力ノイズ規格を超えないと仮定すれば、悪くはありません。

オフセット・ゲインの影響の解析をしたいときには、内部 RC のセトリングタイム曲線を、希望するシステム精度と比べながら解析すればできます。図 14 は内部抵抗と容量 ( $R_{SW}$  と  $C_{EQ}$ ) だけの RC 充電カーブを示しています。

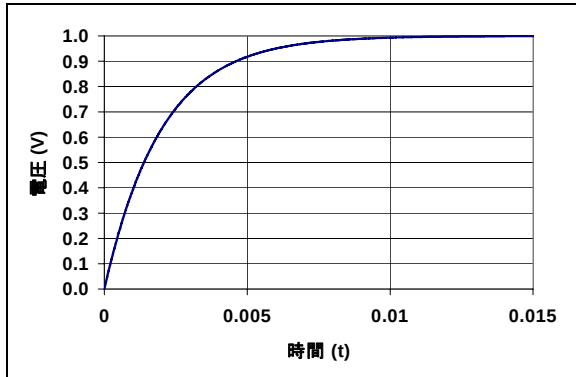


図 14: 標準 RC 曲線。入力信号が  $x$  ppm 以内にセットされるのに必要な時間を考慮しなければなりません。

絶対計測精度を得るためには、必要な内部電荷が定着されなければなりません。(これは、オフセットやゲイン調整をしていないシステムでは、最終電荷の割合として定義されます。)例えば、目標絶対精度が 1 ppm とすると、以下のセトリングタイムがなければなりません。それは RC 時定数 ( $T$ ) の倍数として表されます。

#### 式 4:

$$V_C = V_F \left( 1 - e^{-\frac{t}{\tau}} \right)$$

$$t = n\tau$$

$$V_e = 1 - \frac{V_C}{V_F}$$

$$V_C = V_F (1 - e^{-n})$$

$$e^{-n} = 1 - \frac{V_C}{V_F} = V_e$$

$$\begin{aligned} n &= -\ln(V_e) \\ &= -\ln(1\text{ppm}) \\ &= 13.8 \end{aligned}$$

この例では、1 ppm の絶対精度を得るためには、時定数の 14 倍の時間が定着を完全にするために必要とされます。式 4 を使うと、他の精度要求に対しても同じ計算が使えます。繰り返すと、補正されてオフセットやゲインエラーが無いシステムでは、セトリングタイムの解析は必須ではありません。なぜなら、各サンプルごとにデルタシグマでオーバーサンプリングし平均化しているからです。

## 通信ファームウェア

MCP3551 ADC はシリアル SPI™ デバイスです。本アプリケーションノートには、C とアセンブラ言語の両方で書かれたコードを含んでいます。MCP3551 22-Bit Delta-Sigma ADC PICtail™ Demo Board は、Data-View ソフトウェアと USB 経由で接続された PIC18F4550 を通して接続されます。またマイクロチップのC18コンパイラで書かれたコードが提供されています。使用している SPI 通信プロトコルの概要を下記に示します：

```
void Read3551(char *data)
{
    unsigned char n;
    data[2] = ReadSPI();
    data[1] = ReadSPI();
    data[0] = ReadSPI();
}

//MCU checks every 10 ms if conversion is
finished
if(AcquireData & gSampleFlag)
{
    CS_PTBoard_LOW();           //
    for(n=0;n<5;n++);
    if (SDIpin == 0)
        Read3551(outbuffer);
    CS_PTBoard_HIGH();         //
    gSampleFlag = 0;           //clear
    timeout indicator

    if(!mHIDTxIsBusy())
        HIDTxReport(outbuffer,3);
}
```

システムが DataView ソフトウェアから (PIC18F4550 に USB 経由で ASCII "S" を送ること) Acquire Data モードに設定されている間は、Acquire-Data フラグがセットされています。この間は、MCP3551 は常時変換を続けていて、読み出しデータは USB 経由で PC に送られ保存されます。

PIC18F4550 上で実行されるコードは、10 ms ごとに MCP3551 に対し CS の Low パルスを送ります。Timer1 のフラグと TimerCounter 変数がこの時間のセットに使われます。このこの Low パルスは、立上がりエッジが変換時間より短いときは、デバイスを単発変換モードに設定します。CS パルスが Low の間に、変換が終了したかどうか判定するために SDO 状態がテストされます。ピンが Low であれば、ファームウェアはそのデータを Read3551 サブルーチンを使って取り出します。Read3551() ルーチンは、3 回に分けて ReadSPI ルーチン呼び出し、22 ビットワードと 2 ビットのオーバーフローで含んだ 3 バイトを取り出します。3 バイトデータが戻されると Sample フラグがリセットされプロセスが完了します。

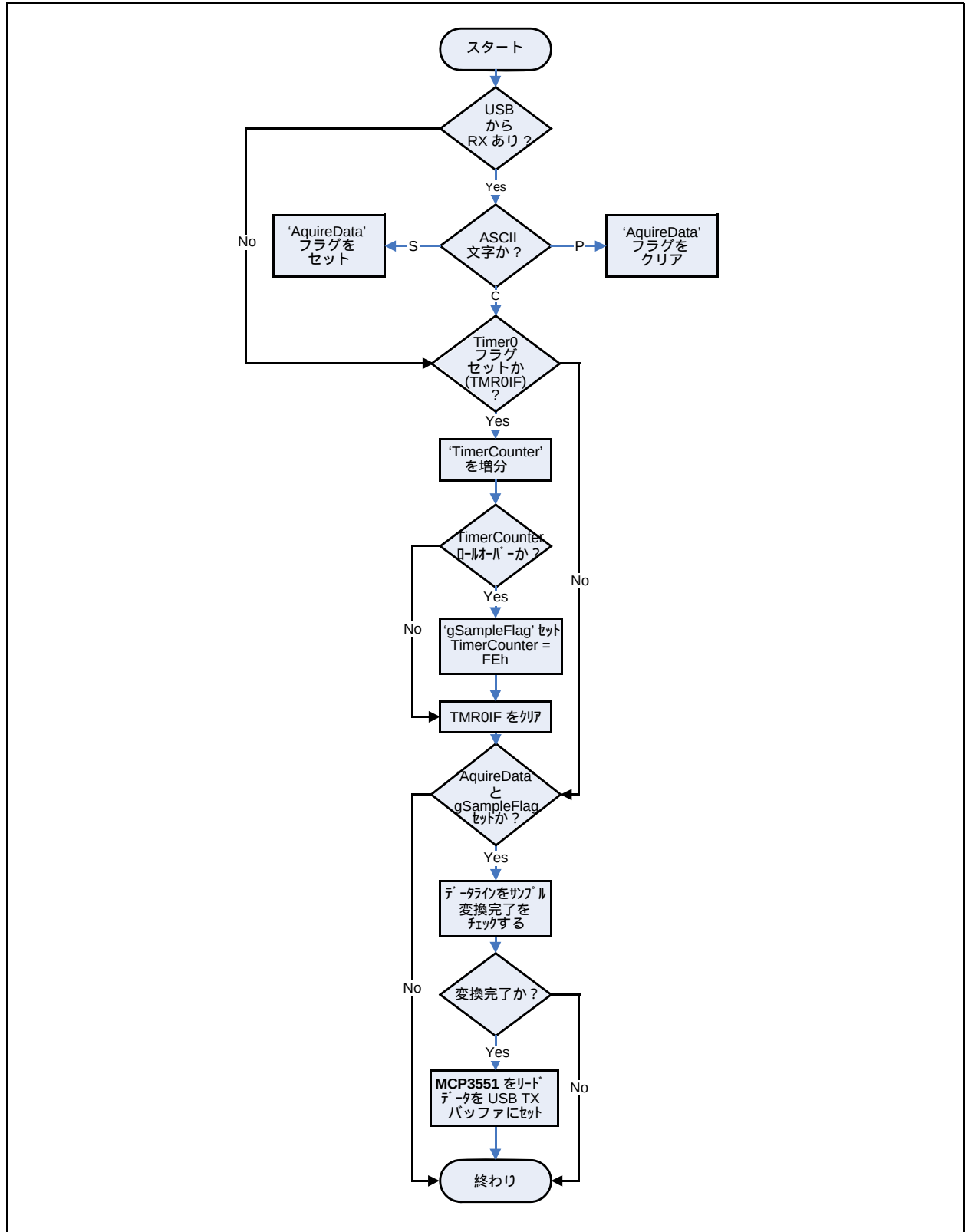


図 15: PIC18F4550 フローチャート。視覚による解析に供するため、USB TX バッファを DataView<sup>1</sup> ソフトウェアに送信する。

## 参考文献

- [1] "Delta-Sigma Data Converters Theory, Design and Simulation", Steven R. Norsworthy, Richard Schreier, Gabor C. Temes, IEEE Press, 1997, pp. 4-9.
- [2] AN9504, "A Brief Introduction to Sigma Delta Conversion", David Jarman, Intersil®, 1995.
- [3] "Modern Business Statistics", Ronald L. Iman, W.J. Conover, Second Edition, John Wiley & Sons, Inc., 1989.

## 付録 A: オーバーサンプリング解析

デルタシグマ ADC は何倍かのオーバーサンプリングデバイスです。高分解能、優れた電源周波数除去、わずかの外付け部品、低電力がこれによるメリットの一例です。高分解能のメリットは、よく混乱がありますが、単純なオーバーサンプリングだけの産物ではありません。

なぜ PICmicro® MCU の SAR ADC はオーバーサンプリングではないのか？

答えは簡単：ノイズの姿形です。高速逐次変換方式 (SAR) ADC の単純なオーバーサンプリングと結果の平均化だけでは、デルタシグマ ADC の高解像度性能は達成できません。オーバーサンプリングと平均化では、サンプル周波数を倍にするごとに 1/2 ビットの精度改善になるだけです。両アプローチのノイズ電力を比較することで、この比較の根拠となる理論をここに示します。

### 単純なオーバーサンプリング

一般的な量子化単位（または LSB）では、この量子内のノイズは完全にランダムと仮定するか、ホワイトノイズと仮定されます。従って、デジタルまたは量子化された出力 (ADC) の量子化ノイズ電力と RMS 量子化電圧は、下記式で求められます：

式 5:

$$e^2_{rms} = \frac{1}{q} \int_{-\frac{q}{2}}^{\frac{q}{2}} e^2 de = \frac{q^2}{12} \quad (V^2)$$
$$e_{rms} = \frac{q}{\sqrt{12}} \quad (V)$$

例えば、 $V_{REF}$  が 5V の 16 ビットコンバータでは、RMS 量子化ノイズは 22  $\mu$ V となります。

ナイキストの定理により、このノイズは 0 から  $f_s/2$  の周波数帯域に一様に存在することになり、またノイズのスペクトラム密度 ( $V/\sqrt{\text{Hz}}$ ) を求めることもできます。：

式 6:

$$E(f) = (e_{rms}) \sqrt{\frac{2}{f_s}} \left( \frac{V}{\sqrt{\text{Hz}}} \right)$$

測定対象となる周波数帯域 ( $f_0$ ) でのノイズ電力を決めるには、この必要な帯域に渡ってノイズを二乗し、積分しなければなりません。

## 式 7:

$$n_o^2 = (e(f)) \quad (V^2)$$

$$n_o = (e_{rms}) \left( \frac{2f_o}{f_s} \right)^{1/2} \quad (V)$$

ここで:

$$f_o < f_s/2$$

$f_s/2f_o$  が OSR (OSR: オーバーサンプリング比) であることを思い出せば、OSR を大きくすればノイズが OSR の平方根で減少するということが分かります。[1]

従って、サンプリング周波数を倍にするとわずか 3 dB しか性能改善されず、これは 0.5 ビット分解能増にしかありません。

デルタシグマ変調器は図 17 のように低周波ノイズを高周波に押しやることで、オーバーサンプリングの性能を向上させています。このデルタシグマ変調器のメリットは、ノイズの姿形にあります。一次のデルタシグマ変調器では、OSR を倍にするごとに、9 dB の精度改善つまり 1.5 ビット分解能改善となります。

蓄積器の出力は、入力信号に量子化により発生する誤差、つまり下記の図と式で表される量子化信号と同じものを加えたものになります:

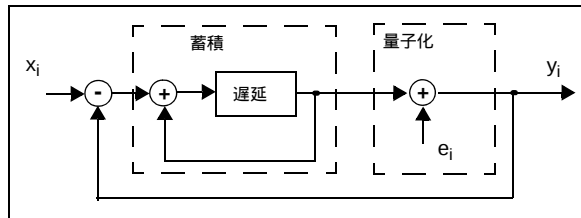


図 16: 一次デルタシグマ変調器のサンプルデータからみた等価図<sup>[1]</sup>

## 式 8:

$$y_i = x_{i-1} + (e_i - e_{i-1})$$

ノイズ密度 ( $e_i - e_{i-1}$ ) を求め、それを二乗し、最終的に使う帯域に渡って積分することで、再度ノイズ電力に変換します:

## 式 9:

$$n_o = e_{RMS} \frac{\pi}{\sqrt{3}} \left( \frac{2f_o}{f_s} \right)^{\frac{3}{2}}$$

デルタシグマ変調器は OSR を倍にするごとに 9 dB (または 1.5 ビット) 帯域ノイズが減少します。単純なオーバーサンプリングより 3 倍よくなります。

ノイズ姿形の性能を改善することは、高次のデルタシグマ変調器設計を使うことで実現できます。高次の変調器のノイズ電力は下記式で要約できます:

## 式 10:

$$n_o = e_{RMS} \frac{\pi^M}{\sqrt{2M+1}} \left( \frac{2f_o}{f_s} \right)^{M+\frac{1}{2}}$$

これによれば、M 次の変調器ではサンプリング周波数を倍にするごとに、ノイズは 3 (2M - 1) dB だけ減少します。例えば、M = 3 (MCP3551/3 デバイスは 3 次の変調器です) であれば、サンプリング周波数を倍にするごとに、分解能を 2.5 ビット改善できます。このアーキテクチャにより MCP3551/3 のようなデバイスを使えば、ノイズ性能を < 3 μV とすることが出来ます。図 17 に周波数領域のノイズ姿形を示します。ノイズは、オーバーサンプリング周波数 ( $f_s$ ) 付近の高い周波数に押しやられています。さらに注意すべきは、熱雑音も OSR を倍にする都度、標準平均化規則の 3 dB (1/2 ビット) 改善に従い、信号の一部として処理されることです。

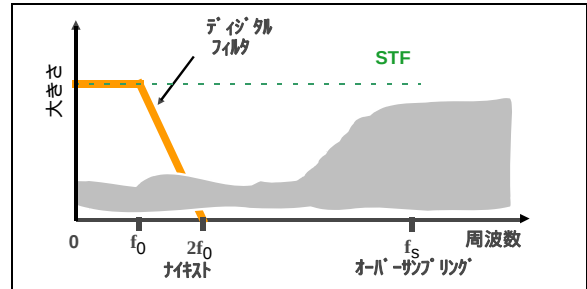


図 17: 必要な帯域で低ノイズレベルを達成したデルタシグマ変調器のノイズ姿形。これは高速 SAR ADC の単純オーバーサンプリングと平均化だけでは不可能。

(注: STF は Signal Transfer Function の略、すなわち信号伝達関数)

## 付録 B: ソフトウェア - SPI™ 通信のアセンブラプログラム

表 B-1: SPI \_ PIC18F252.ASM

```
=====
; Software License Agreement
;
; The software supplied herewith by Microchip Technology Incorporated
; (the "Company") for its PICmicro(r) microcontroller is intended and
; supplied to you, the Company's customer, for use solely and
; exclusively on Microchip PICmicro Microcontroller products. The
; software is owned by the Company and/or its supplier, and is
; protected under applicable copyright laws. All rights are reserved.
; Any use in violation of the foregoing restrictions may subject the
; user to criminal sanctions under applicable laws, as well as to
; civil liability for the breach of the terms and conditions of this
; license.
;
; THIS SOFTWARE IS PROVIDED IN AN "AS IS" CONDITION. NO WARRANTIES,
; WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, BUT NOT LIMITED
; TO, IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
; PARTICULAR PURPOSE APPLY TO THIS SOFTWARE. THE COMPANY SHALL NOT,
; IN ANY CIRCUMSTANCES, BE LIABLE FOR SPECIAL, INCIDENTAL OR
; CONSEQUENTIAL DAMAGES, FOR ANY REASON WHATSOEVER.
;
=====
; Filename:      840DSM.asm
=====
; Author:       Craig L. King
; Company:      Microchip Technology Inc.
; Revision:     1.00
; Date:        July 21, 2004
; Assembled using MPASM(tm) WIN compiler
=====
; Include Files:  p18f252.inc    V1.3
=====
;
=====

list p=18f252;list directive to define processor
#include <p18f252.inc>;processor specific definitions

; Change the following lines to suit your application.

_CONFIG1H    _CONFIG1H, _OSCS_OFF_1H & _HS_OSC_1H
_CONFIG2L    _CONFIG2L, _BOR_ON_2L & _PWRT_OFF_2L
_CONFIG2H    _CONFIG2H, _WDT_OFF_2H
_CONFIG3H    _CONFIG3H, _CCP2MX_OFF_3H
_CONFIG4L    _CONFIG4L, _STVR_OFF_4L & _LVP_OFF_4L & _DEBUG_OFF_4L
_CONFIG5L    _CONFIG5L, _CP0_OFF_5L & _CP1_OFF_5L & _CP2_OFF_5L & _CP3_OFF_5L
_CONFIG5H    _CONFIG5H, _CPB_OFF_5H & _CPD_OFF_5H
_CONFIG6L    _CONFIG6L, _WRT0_OFF_6L & _WRT1_OFF_6L & _WRT2_OFF_6L & _WRT3_OFF_6L
_CONFIG6H    _CONFIG6H, _WRTC_OFF_6H & _WRTB_OFF_6H & _WRTD_OFF_6H
_CONFIG7L    _CONFIG7L, _EBTR0_OFF_7L & _EBTR1_OFF_7L & _EBTR2_OFF_7L & _EBTR3_OFF_7L
_CONFIG7H    _CONFIG7H, _EBTRB_OFF_7H

; -----
; Constants

SPBRG_VAL    EQU        .64                ;set baud rate 19.2 for 20Mhz clock
; -----
```

表 B-1: SPI \_ PIC18F252.ASM (CONTINUED)

```

;Bit Definitions

GotNewData      EQU      0          ;bit indicates new data received

#define do       0              ;transmit bit
#define di       1              ;transmit bit
#define CLK      PORTC,2        ;clock
#define ADCS     PORTC,3        ;chip select
#define DIN      PORTC, 4       ;data in / READY
#define BITCOUNT 0x08

;-----
;Variables

                CBLOCK      0x000
                Flags        ;byte to store indicator flags
                RxData        ;data received
                TxData        ;data to transmit
                ParityByte    ;byte used for parity calculation
                ParityBit     ;byte to store received parity bit

                COUNT
                COUNT2
                COUNT3
                DLYCNT
                DLYCNT1
                DLYCNT2
                byte2
                byte1
                byte0

                ENDC

;-----
;This code executes when a reset occurs.

                ORG      0x0000 ;place code at reset vector

ResetCode:      bra      Main    ;go to beginning of program

;-----
;This code executes when a high priority interrupt occurs.

                ORG 0x0008      ;place code at interrupt vector

HighIntCode:    ;do interrupts here

                reset           ;error if no valid interrupt so reset

;-----
;This code executes when a low priority interrupt occurs.

                ORG      0x0018 ;place code at interrupt vector

LowIntCode:     ;do interrupts here

                reset           ;error if no valid interrupt so reset
;

```

表 B-1: SPI \_ PIC18F252.ASM (CONTINUED)

```
;          MAIN ROUTINE
;-----
;Main routine calls the receive polling routines and checks for a byte
;received. It then calls a routine to transmit the data back.
;
;
;This routine sets up the USART and then samples the MCP3551, and sends the 3 bytes out
;on the USART, THEN REPEAT
;
;-----

Main:      rcall   SetupSerial    ;set up serial port

           ;do other initialization here
           movlw   b'11010000'
           movwf   TRISC

MainLoop:

           ;go get the 3551 data

           rcall   Sample3551

           movff   byte2,TxData

           bsf     TXSTA,TX9D
           rcall   TransmitSerial ;go transmit the data

           movff   byte1,TxData

           bcf     TXSTA,TX9D
           rcall   TransmitSerial ;go transmit the data

           movff   byte0,TxData

           bcf     TXSTA,TX9D
           rcall   TransmitSerial ;go transmit the data

DoOtherStuff:;do other stuff here

           bra     MainLoop      ;go do main loop again

;-----
```

表 B-1: SPI \_ PIC18F252.ASM (CONTINUED)

```

;Check if data received and if so, place in a register and check parity.

ReceiveSerial: btfss   PIR1,RCIF      ;check if data received
               return                ;return if no data

               btfsc   RCSTA,OERR     ;if overrun error occurred
               bra     ErrSerialOverr ;then go handle error
               btfsc   RCSTA,FERR     ;if framing error occurred
               bra     ErrSerialFrame ;then go handle error

               movf    RCSTA,W         ;get received parity bit
               movwf   ParityBit      ;and save
               movf    RCREG,W        ;get received data
               movwf   RxData         ;and save

               rcall   CalcParity      ;calculate parity
               movf    ParityBit,W    ;get received parity bit
               xorwf   ParityByte,F   ;compare with calculated parity bit
               btfsc   ParityByte,0   ;check result of comparison
               bra     ErrSerlParity  ;if parity is different, then error
               bsf     Flags,GotNewData ;else indicate new data received
               return

;error because OERR overrun error bit is set
;can do special error handling here - this code simply clears and continues

ErrSerialOverr: bcf     RCSTA,CREN     ;reset the receiver logic
               bsf     RCSTA,CREN     ;enable reception again
               return

;error because FERR framing error bit is set
;can do special error handling here - this code simply clears and continues

ErrSerialFrame: movf    RCREG,W        ;discard received data that has error
               return

;error because parity bit is not correct
;can do special error handling here - this code simply clears and continues

ErrSerlParity: return                ;return without indicating new data

;-----
;Transmit data in WREG with parity when the transmit register is empty.

TransmitSerial: btfss   PIR1,TXIF      ;check if transmitter busy
               bra     $-2            ;wait until transmitter is not busy

               movf    TxData,W        ;get data to be transmitted

               movf    TxData,W        ;get data to transmit
               movwf   TXREG          ;transmit the data
               return

;-----

```

**表 B-1: SPI \_ PIC18F252.ASM (CONTINUED)**

```

;Calculate even parity bit.
;Data starts in working register, result is in LSB of ParityByte

CalcParity:    movwf    ParityByte        ;get data for parity calculation
               rrncf    ParityByte,W      ;rotate
               xorwf    ParityByte,W      ;compare all bits against neighbor
               movwf    ParityByte        ;save
               rrncf    ParityByte,F      ;rotate
               rrncf    ParityByte,F      ;rotate
               xorwf    ParityByte,F      ;compare every 2nd bit and save
               swapf    ParityByte,W      ;rotate 4
               xorwf    ParityByte,F      ;compare every 4th bit and save
               return

;-----
;Set up serial port.

SetupSerial:   movlw    0xc0              ;set tris bits for TX and RX
               iorwf    TRISC,F
               movlw    SPBRG_VAL        ;set baud rate
               movwf    SPBRG
               movlw    0x64             ;enable nine bit tx and high baud rate
               movwf    TXSTA
               movlw    0xd0             ;enable serial port and nine bit rx
               movwf    RCSTA
               clrf     Flags            ;clear all flags
               return

;*****
;*****
;-----
;
;Sample3551
;
;This is where you sample the MCP3551 and put your 22-bit
;answer into the following bytes:

;      byte 2      --      byte 1      --      byte 0
;      MSB                                     LSB
;
; This routine returns the data
;-----

```

表 B-1: SPI \_ PIC18F252.ASM (CONTINUED)

```

Sample3551

    clrf    byte2                ; reset input buffer
    clrf    byte1                ; reset input buffer
    clrf    byte0                ; reset input buffer

    bsf     CLK                  ; clock idle high
    bcf     ADCS                 ; INITIATE THE CONVERSION

    movlw   .6
    call    VAR1000TcyDELAY      ; delay 1ms

    bsf     ADCS                 ; CS HIGH (Single Conversion mode)

    movlw   .160
    call    VAR1000TcyDELAY      ; total delay 110ms (GREATER THAN TCONV, CAN BE REDUCED)
    call    VAR1000TcyDELAY      ; delay 160k Tcy
    call    VAR1000TcyDELAY      ; delay 250k Tcy
    call    VAR1000TcyDELAY      ; delay 250k Tcy

    bcf     ADCS                 ; GET THE CONVERSION DATA

    movlw   BITCOUNT
    movwf   COUNT                ; FIRST BYTE
FIRST_BYTE
    bcf     CLK                  ; drop clock for next bit
    bsf     CLK                  ; set clock to latch bit
    bcf     STATUS,C             ; pre-clear carry
    btfsc   DIN                  ; check for high or low bit
    bsf     STATUS,C             ; set carry bit
    rlc     byte2, f              ; roll the carry bit left into position
    decfsz  COUNT, f             ; decrement bit counter
    goto    FIRST_BYTE           ; get next bit

    movlw   BITCOUNT
    movwf   COUNT                ; SECOND BYTE
SECOND_BYTE
    bcf     CLK                  ; drop clock for next bit
    bsf     CLK                  ; set clock to latch bit
    bcf     STATUS,C             ; pre-clear carry
    btfsc   DIN                  ; check for high or low bit
    bsf     STATUS,C             ; set carry bit
    rlc     byte1, f              ; roll the carry bit left into place
    decfsz  COUNT, f             ; decrement bit counter
    goto    SECOND_BYTE          ; get next bit
    movlw   BITCOUNT
    movwf   COUNT                ; THIRD BYTE
THIRD_BYTE
    bcf     CLK                  ; drop clock for next bit
    bsf     CLK                  ; set clock to latch bit
    bcf     STATUS,C             ; pre-clear carry
    btfsc   DIN                  ; check for high or low bit
    bsf     STATUS,C             ; set carry bit
    rlc     byte0, f              ; roll the carry bit left into place
    decfsz  COUNT, f             ; decrement bit counter
    goto    THIRD_BYTE           ; get next bit
    bsf     CLK                  ; clock idles high

```

\_\_\_\_\_

[illegible]

---

## 付録 C: ハードウェア - SPI™ 通信の C 言語プログラム

表 C-1: MCP3551.C

```

/*****
 *
 *          Microchip USB C18 Firmware - MCP3551 PICTail(tm) Demo
 *
 *****/
 * FileName:      MCP3551.c
 * Dependencies:  See INCLUDES section below
 * Processor:     PIC18
 * Compiler:      C18 2.30.01+
 * Company:       Microchip Technology Inc.
 *
 * Software License Agreement
 *
 * The software supplied herewith by Microchip Technology Incorporated
 * (the Company) for its PICmicro(r) microcontroller is intended and
 * supplied to you, the Company's customer, for use solely and
 * exclusively on Microchip PICmicro(r) microcontroller products. The
 * software is owned by the Company and/or its supplier, and is
 * protected under applicable copyright laws. All rights are reserved.
 * Any use in violation of the foregoing restrictions may subject the
 * user to criminal sanctions under applicable laws, as well as to
 * civil liability for the breach of the terms and conditions of this
 * license.
 *
 * THIS SOFTWARE IS PROVIDED IN AN "AS IS" CONDITION. NO WARRANTIES,
 * WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, BUT NOT LIMITED
 * TO, IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
 * PARTICULAR PURPOSE APPLY TO THIS SOFTWARE. THE COMPANY SHALL NOT,
 * IN ANY CIRCUMSTANCES, BE LIABLE FOR SPECIAL, INCIDENTAL OR
 * CONSEQUENTIAL DAMAGES, FOR ANY REASON WHATSOEVER.
 *
 * Author          Date          Comment
 * ~~~~~
 * Pat Richards    xx/xx/xx      Original.
 *****/

/** I N C L U D E S *****/
#include <p18cxxx.h>
#include <usart.h>
#include <spi.h>
#include "system\types.h"
#include "system\usb\usb.h"

#include "io_cfg.h"          // I/O pin mapping
#include "user\MCP3551.h"

/** V A R I A B L E S *****/
#pragma udata
//byte old_sw2,old_sw3;

BOOL emulate_mode;
rom signed char dir_table[]={-4,-4,-4, 0, 4, 4, 4, 0};
byte movement_length;
byte vector = 0;

byte AcquireData = 1; //0 = STOP; 1 = Acquire
char buffer[3];

char inbuffer[BUF_SIZE];      // 8 byte input to USB device buffer
char outbuffer[BUF_SIZE];     // 8 byte output to USB device buffer
byte TimerCounter = 0xF0;
static unsigned char gSampleFlag;

```

表 C-1: MCP3551.C (CONTINUED)

```
/** P R I V A T E P R O T O T Y P E S *****/
void Read3551(char *data);
unsigned char ReadSPI( void );
void CheckBoardConnect(void);

/** D E C L A R A T I O N S *****/
#pragma code
void UserInit(void)
{
    byte i;

    CS_PTBoard_HIGH();          //Drive high
    tris_CS = 0; //Output
    OpenSPI(SPI_FOSC_16, MODE_11, SMPMID);

    TRISBbits.TRISB0 = 1;      //SDI
    TRISBbits.TRISB1 = 0;      //SCK

    //-----
    // initialize variables
    //-----
    for (i=0; i<BUF_SIZE; i++) // initialize input and output buffer to 0
    {
        inbuffer[i]=0;
        outbuffer[i]=0;
    }

    //Timer 0
    TMR0H = 0;                  //clear timer
    TMR0L = 0;                  //clear timer
    T0CONbits.PSA = 0;          //Assign prescaler to Timer 0
    T0CONbits.T0PS2 = 1;        //Setup prescaler
    T0CONbits.T0PS1 = 1;        //Will time out every 51 us based on
    T0CONbits.T0PS0 = 1;        //20 MHz Fosc
    T0CONbits.T0CS = 0;         //Increment on instuction cycle
} //end UserInit

/*****

* Function:      void ProcessIO(void)
*
* PreCondition:  None
*
* Input:         None
*
* Output:        None
*
* Side-Effects:  None
*
* Overview:      This function is a place holder for other user routines.
                  It is a mixture of both USB and non-USB tasks.
*
* Note:         None
*****/
```

表 C-1: MCP3551.C (CONTINUED)

```

void ProcessIO(void)
{
    char n;
    // User Application USB tasks
    if((usb_device_state < CONFIGURED_STATE) || (UCONbits.SUSPND==1)) return;

    if (HIDRxReport(inbuffer, 1))    // USB receive buffer has data
    {
        switch(inbuffer[0])          // interpret command
        {
            case START_ACQUISITION:    // 'S' START aquisition of data
                AquireData = 1;
                CS_PTBoard_LOW();        //Start conversion
                TimerCounter = 0xFF;
                break;

            case STOP_ACQUISITION:      // 'T' STOP aquisition of data
                AquireData = 0;
                break;

            case CHANNEL_SELECTION:     // 'C' A/D Channel Selection
                break;

            default:                    // unrecognized or null command
                ;
        }
    } // END switch(inbuffer[0])
} //END if (HIDRxReport(inbuffer, 1))

//Inst. cycle = 200 ns; TMR0IF sets every 51 us
if(INTCONbits.TMR0IF)
{
    TimerCounter++;
    if (!TimerCounter)                //if rolled over, set flag. User code will handle the rest.
    {
        TimerCounter = 0xFE;
        gSampleFlag = 1;
    }
    INTCONbits.TMR0IF = 0;
}

//MCU checks every 10 ms if conversion is finished
if(AquireData & gSampleFlag)
{
    CS_PTBoard_LOW();                //
    for(n=0;n<5;n++);
    if (SDIpin == 0)
        Read3551(outbuffer);
    CS_PTBoard_HIGH();                //
    gSampleFlag = 0;                  //clear timeout indicator

    if(!mHIDTxIsBusy())
        HIDTxReport(outbuffer,3);
}
} //end ProcessIO

```

表 C-1: MCP3551.C (CONTINUED)

```
/* *****  
 * Function:      void Read3551(char *data)  
 *  
 * PreCondition:  None  
 *  
 * Input:        Pointer to a string; must be three bytes min  
 *  
 * Output:       None  
 *  
 * Side-Effects:  None  
 *  
 * Overview:       
 *  
 * Note:          
 ***** */  
void Read3551(char *data)  
{  
    unsigned char n;  
    data[2] = ReadSPI();  
    data[1] = ReadSPI();  
    data[0] = ReadSPI();  
}  
  
/* *****  
 * Function:      CheckBoardConnect(void)  
 *  
 * PreCondition:  None  
 *  
 * Input:        None  
 *  
 * Output:       None  
 *  
 * Overview:     Checks if the Stimulus Board is attached. Future expansion.  
 *  
 * Side-Effects:  None  
 ***** */  
void CheckBoardConnect(void)  
{  
}  
  
/** EOF MCP3551.C ***** */
```

---

マイクロチップデバイスのコード保護機能に関する以下の点に留意ください。

- マイクロチップの製品は各製品独自のマイクロチップデータシートにある仕様を満たしています。
- 各製品ファミリーは、通常の状態ですべての方法で利用いただければ市場にある類似製品の中で最も安全なファミリーの一つとマイクロチップは信じております。
- 不正かつ非合法な方法を使ったコード保護機能の侵害があります。弊社の理解ではこうした手法は、マイクロチップデータシートにある動作仕様書以外の方法でマイクロチップ製品を使用することになります。こうした手法を使用した人は、ほとんどの場合、知的財産権の侵害となります。
- マイクロチップはコードの統合性に関心をお持ちの顧客とは協働させていただきます。
- マイクロチップまたは他のセミコンダクターメーカーがコードの安全性を保証したものではありません。コード保護は製品保護が「破られない」ということを保証するものではありません。

コード保護は常に進化します。マイクロチップは、当社製品のコード保護機能を継続的に改善することをお約束いたします。マイクロチップのコード保護機能を破ることは、デジタル・ミレニアム著作権法に違反します。こうした行為によるソフトウェアや著作権に関わる作品への不正アクセスがあった場合、同法に基づき賠償請求する権利があります。

---

本書の日本語版はユーザーの使用のために提供されます。Microchip Technology Inc. とその子会社、関連会社、すべての取締役、役員、職員、代理人は翻訳の間違いにより起こるいかなる責も負わないものとします。間違いが疑われる箇所については、Microchip Technology Inc. 発行のオリジナル文書を参照いただくようお願いいたします。

本書の日本語版はユーザーの使用のために提供されます。Microchip Technology Inc. とその子会社、関連会社、すべての取締役、役員、職員、代理人は翻訳の間違いにより起こるいかなる責も負わないものとします。間違いが疑われる箇所については、Microchip Technology Inc. 発行のオリジナル文書を参照いただくようお願いいたします。

本書に書かれているデバイスアプリケーション等に関する内容は、参考情報に過ぎません。ご利用のアプリケーションが仕様を満たしているかどうかについては、お客様の責任において確認をお願いします。これらの情報の正確さ、またはこれらの情報の使用に関し、マイクロチップテクノロジーインクはいかなる表明と保証を行うものではなく、また、一切の責任を負うものではありません。マイクロチップの明示的な書面による承認なしに、生命維持装置あるいは生命安全保障にマイクロチップの製品を使用することはすべて購入者のリスクとし、また購入者はこれによって起きたあらゆる損害、クレーム、訴訟、費用に関して、マイクロチップは擁護され、免責され、損害をうけないことに同意するものとします。知的財産権に基づくライセンスを暗示的に与えたものではありません。

## 商標

マイクロチップの名称とロゴ、マイクロチップのロゴ、Accuron、dsPIC、KEELOQ、microID、MPLAB、PIC、PICmicro、PICSTART、PRO MATE、PowerSmart、rfPIC、SmartShunt は米国及び他の国々において、マイクロチップテクノロジーインク の登録商標です。

Amplab、FilterLab、Migratable Memory、MXDEV、MXLAB、PICMASTER、SEEVAL、SmartSensor、The Embedded Control Solutions Company は、米国においてマイクロチップテクノロジーインク の登録商標です。

Analog-for-the-Digital Age、Application Maestro、dsPICDEM、dsPICDEM.net、dsPICworks、ECAN、ECONOMONITOR、FanSense、FlexROM、fuzzyLAB、In-Circuit Serial Programming、ICSP、ICEPIC、Linear Active Thermistor、MPASM、MPLIB、MPLINK、MPSIM、PICkit、PICDEM、PICDEM.net、PICLAB、PICtail、PowerCal、PowerInfo、PowerMate、PowerTool、rfLAB、rfPICDEM、Select Mode、Smart Serial、SmartTel、Total Endurance、UNI/O、WiperLock、及び Zena は、米国及び他の国々において、マイクロチップテクノロジーインク の登録商標とです。

SQTP は米国においてマイクロチップテクノロジーインクのサービスマークです。

本書に記載された上記以外の商標は、それぞれの会社の財産です。

著作権。© 2006 年マイクロチップテクノロジーインク、米国で印刷。無断複写・転載を禁じます。

 再生紙を使用。

**QUALITY MANAGEMENT SYSTEM**  
**CERTIFIED BY DNV**  
**== ISO/TS 16949:2002 ==**

マイクロチップは、ISO/TS-16949 を受けました。本社、アリゾナ州チャンドラーとテンベとカリフォルニア州マウンテンビューにあるデザイン及びウェア施設に対する 2003 年 10 月品質システム認証です。弊社の品質システムプロセスと手続きは、PICmicro® 8-bit MCUs、KEELOQ® コードホッピングデバイス、シリアル EEPROMs、マイクロペリフェラル、非揮発性メモリとアナログ製品を対象としています。更に、開発システムの設計及び製造に関するマイクロチップの品質システムは、2000 年に ISO9001 の認証を受けています。

## 全世界の販売及びサービス拠点

### AMERICAS

**Corporate Office**  
2355 West Chandler Blvd.  
Chandler, AZ 85224-6199  
Tel: 480-792-7200  
Fax: 480-792-7277  
Technical Support:  
<http://support.microchip.com>  
Web Address:  
[www.microchip.com](http://www.microchip.com)

**Atlanta**  
Alpharetta, GA  
Tel: 770-640-0034  
Fax: 770-640-0307

**Boston**  
Westborough, MA  
Tel: 774-760-0087  
Fax: 774-760-0088

**Chicago**  
Itasca, IL  
Tel: 630-285-0071  
Fax: 630-285-0075

**Dallas**  
Addison, TX  
Tel: 972-818-7423  
Fax: 972-818-2924

**Detroit**  
Farmington Hills, MI  
Tel: 248-538-2250  
Fax: 248-538-2260

**Kokomo**  
Kokomo, IN  
Tel: 765-864-8360  
Fax: 765-864-8387

**Los Angeles**  
Mission Viejo, CA  
Tel: 949-462-9523  
Fax: 949-462-9608

**San Jose**  
Mountain View, CA  
Tel: 650-215-1444  
Fax: 650-961-0286

**Toronto**  
Mississauga, Ontario,  
Canada  
Tel: 905-673-0699  
Fax: 905-673-6509

### ASIA/PACIFIC

**Australia - Sydney**  
Tel: 61-2-9868-6733  
Fax: 61-2-9868-6755

**China - Beijing**  
Tel: 86-10-8528-2100  
Fax: 86-10-8528-2104

**China - Chengdu**  
Tel: 86-28-8676-6200  
Fax: 86-28-8676-6599

**China - Fuzhou**  
Tel: 86-591-8750-3506  
Fax: 86-591-8750-3521

**China - Hong Kong SAR**  
Tel: 852-2401-1200  
Fax: 852-2401-3431

**China - Qingdao**  
Tel: 86-532-8502-7355  
Fax: 86-532-8502-7205

**China - Shanghai**  
Tel: 86-21-5407-5533  
Fax: 86-21-5407-5066

**China - Shenyang**  
Tel: 86-24-2334-2829  
Fax: 86-24-2334-2393

**China - Shenzhen**  
Tel: 86-755-8203-2660  
Fax: 86-755-8203-1760

**China - Shunde**  
Tel: 86-757-2839-5507  
Fax: 86-757-2839-5571

**China - Wuhan**  
Tel: 86-27-5980-5300  
Fax: 86-27-5980-5118

**China - Xian**  
Tel: 86-29-8833-7250  
Fax: 86-29-8833-7256

### ASIA/PACIFIC

**India - Bangalore**  
Tel: 91-80-4182-8400  
Fax: 91-80-4182-8422

**India - New Delhi**  
Tel: 91-11-5160-8631  
Fax: 91-11-5160-8632

**India - Pune**  
Tel: 91-20-2566-1512  
Fax: 91-20-2566-1513

**Japan - Yokohama**  
Tel: 81-45-471- 6166  
Fax: 81-45-471-6122

**Korea - Gumi**  
Tel: 82-54-473-4301  
Fax: 82-54-473-4302

**Korea - Seoul**  
Tel: 82-2-554-7200  
Fax: 82-2-558-5932 or  
82-2-558-5934

**Malaysia - Penang**  
Tel: 60-4-646-8870  
Fax: 60-4-646-5086

**Philippines - Manila**  
Tel: 63-2-634-9065  
Fax: 63-2-634-9069

**Singapore**  
Tel: 65-6334-8870  
Fax: 65-6334-8850

**Taiwan - Hsin Chu**  
Tel: 886-3-572-9526  
Fax: 886-3-572-6459

**Taiwan - Kaohsiung**  
Tel: 886-7-536-4818  
Fax: 886-7-536-4803

**Taiwan - Taipei**  
Tel: 886-2-2500-6610  
Fax: 886-2-2508-0102

**Thailand - Bangkok**  
Tel: 66-2-694-1351  
Fax: 66-2-694-1350

### EUROPE

**Austria - Wels**  
Tel: 43-7242-2244-399  
Fax: 43-7242-2244-393

**Denmark - Copenhagen**  
Tel: 45-4450-2828  
Fax: 45-4485-2829

**France - Paris**  
Tel: 33-1-69-53-63-20  
Fax: 33-1-69-30-90-79

**Germany - Munich**  
Tel: 49-89-627-144-0  
Fax: 49-89-627-144-44

**Italy - Milan**  
Tel: 39-0331-742611  
Fax: 39-0331-466781

**Netherlands - Druenen**  
Tel: 31-416-690399  
Fax: 31-416-690340

**Spain - Madrid**  
Tel: 34-91-708-08-90  
Fax: 34-91-708-08-91

**UK - Wokingham**  
Tel: 44-118-921-5869  
Fax: 44-118-921-5820