AVR055：内蔵RC発振器走行時校正に対する32kHzクリスタルの使い方

要点

- 32kHz外部クリスタルを使う校正
- 最大±2%精度でのRC周波数調整
- どの動作電圧と温度ででもRC発振器を調節
- 仕様内のどの周波数へもRC発振器を調節
- C言語で書かれた全ソース コード
- 調節可能なRC発振器と非同期タイマ/カウンタを持つ全てのAVRを支援
- 選択可能な校正クロック周波数

1. 序説

現在のAVRマイクロ コントローラの大多数は内蔵RC発振器での走行の可能性を提供します。内蔵RC発振器周波数は殆どのAVRに於いてデバイスのデータシートで指定される周波数の±2%内に校正することができ、いくつかのデバイスは±1%さえ達成可能です。

Atmelの工場で実行される校正は固定の動作電圧と温度で行われます。内蔵RC発振器の周波数が動作電圧と温度の両方によって影響を及ぼされるため、時に第2の校正を実行することが望まれます。内蔵RC発振器とOSCCAL校正レジスタを持つAVRデバイスは時計用32kHzクリスタルのような安定な周波数の外部信号を使って充分な精度で校正することができます。既定値(25℃、代表的に5V)と異なる、特定の応用環境に一致する温度または供給電圧でデバイスを走行する場合、または発振器を異なる周波数に調節することでも、この第2校正は有用であり得ます。

走行時内蔵発振器校正に対する外部の時計用クリスタル使用は、温度範囲全体に渡ってと、動作電圧と無関係に正確なシステム クロックを必要とする応用に対して効率的な費用の解決策になります。

この応用記述は非同期タイマ/カウンタの入力として外部32.768kHzクリスタルを使って内蔵RC発振器を校正する早くて正確な方法を記述します。

2. 動作の理屈 – 内蔵RC発振器

製品に於ける内蔵RC発振器は5Vまたは3.3Vのどちらかで校正されます。校正中に使われる動作電圧についての情報に関しては個別デバイスのデータシートを参照してください。工場校正の精度は±3%または±10%以内です(データシートを参照してください)。精度に対する設計の要求がAtmelによる工場での標準校正によって提供され得るものを越える場合、RC発振器の第2校正を実行することが可能です。これを行うことにより、±1%(工場校正で10%精度のものについては±2%)以内の周波数精度を得ることが可能です。従って第2校正は発振器の精度改善または周波数の仕立て上げを実行することができます。

3. クロック選択

AVRのヒューズ設定は使われるシステム クロック元を制御します。内蔵RC発振器を使うには対応するヒューズ設定が選択されなければなりません。ヒューズの概要はデータシートで利用可能です。外部クリスタルを使って校正するとき、システム クロックは内蔵発振器で走行していなければなりません。

4. クリスタル接続

殆どのAVRでは外部クリスタルの接続時にTOSC1/2ピンに外部コンデンサの接続が必要です。これは1.8Vでの動作能力を持つ全てのデバイスに適用します。他のデバイスではTOSC1/2ピン間に外部クリスタルを(他の部品なしで)直接接続することができます。クリスタル接続の詳細についてはデバイスのデータシートを参照してください。

8ビット AVR® マイクロ コントローラ

応用記述

本書は一般の方々の便宜のため有志により作成されたもので、Atmel社とは無関係であることを御承知ください。しおりのはじめにでの内容にご注意ください。

Rev. 8002D-07/08, 8002DJ4-02/21

5. 基準周波数

後続項はAVRﾏｲｸﾛ ｺﾝﾄﾛｰﾗで利用可能な内蔵RC発振器の概要を提供します。

いくつかのAVRは1つのRC発振器を持ち、一方他はそれらから選ぶための4つまでの異なるRC発振器を持ちます。周波数は1MHz～9.6MHzの範囲です。充分な精度の内蔵RC発振器にするためにAVRのI/O領域内に発振校正(OSCCAL)ﾚｼﾞｽﾀが存在します。OSCCALﾚｼﾞｽﾀは1ﾊﾞｲﾄ幅です。このﾚｼﾞｽﾀの目的は発振器周波数の調整を可能にすることです。この調整はRC発振器を校正する時に利用可能です。外部ｸﾘｽﾀﾙを使って校正する時にﾃﾞﾊﾞｲｽのﾀｲﾏ/ｶｳﾝﾀはｼｽﾃﾑ ｸﾛｯｸと非同期に動作する可能性を持たなければなりません。

ﾃﾞﾊﾞｲｽがAtmelによって校正されるとき、その校正ﾊﾞｲﾄがﾃﾞﾊﾞｲｽの識票列に格納されます。RC発振器周波数が工程に依存するため、この校正ﾊﾞｲﾄはﾃﾞﾊﾞｲｽ毎に変わり得ます。ﾃﾞﾊﾞｲｽが複数の発振器を持つ場合、RC発振器の各々に対する校正ﾊﾞｲﾄが識票列に格納されます。

既定のRC発振器校正ﾊﾞｲﾄは殆どのﾃﾞﾊﾞｲｽに於いて始動で自動的に識票列から取得されてOSCCALﾚｼﾞｽﾀ内に複写されます。例えば、ATmega8の既定ｸﾛｯｸ設定は1MHz内蔵RC発振器で、このﾃﾞﾊﾞｲｽについては1MHz RC発振器に対応する校正ﾊﾞｲﾄが始動で自動的に設定されます。ﾋｭｰｽﾞが変えられ、その結果として既定設定の代わりに4MHz発振器が使われる場合、校正ﾊﾞｲﾄは手動でOSCCALﾚｼﾞｽﾀに設定されなければなりません。識票列から4MHz校正ﾊﾞｲﾄを読み、従ってそれをﾌﾗｯｼｭ ﾒﾓﾘまたはEEPROMの位置へ格納するのに書き込みﾂｰﾙを使うことができ、そしてそれは走行時に主ﾌﾟﾛｸﾞﾗﾑによって読まれてOSCCAL内に複写されます。

OSCCALﾚｼﾞｽﾀを使う発振器調整に加え、いくつかのﾃﾞﾊﾞｲｽはｼｽﾃﾑ ｸﾛｯｸ前置分周器が特徴です。前置分周ﾚｼﾞｽﾀ(CLKPR)は予め定義された2のべき乗係数でｼｽﾃﾑ ｸﾛｯｸを分周するのに使うことができます。また、この前置分周器はAVRのﾋｭｰｽﾞを通して予め設定することができ、CKDIV8ﾋｭｰｽﾞのﾌﾟﾛｸﾞﾗﾑ(0)はCLKPRをｼｽﾃﾑ ｸﾛｯｸの8分周に設定します。これはﾃﾞﾊﾞｲｽが最高周波数仕様以下で動作することの保証を行うことができます。CLKPRはｼｽﾃﾑ ｸﾛｯｸ周波数を内部的に変更するために走行時に変更することができます。

発振器の基準周波数は分周されていない発振器周波数として定義されます。

6. RC発振器概要

経歴を通してAVRﾏｲｸﾛ ｺﾝﾄﾛｰﾗで様々なRC発振器が利用可能です。いくつかのﾃﾞﾊﾞｲｽ例と共にRC発振器の概要が**表6-1.**で見えます。この一覧は発振器形式によって分類され、そしてこれは発売時期による分類とも大体等価です。調整可能な発振器と非同期動作の可能性を持つﾃﾞﾊﾞｲｽ例だけがこの表で一覧にされます。支援されるﾃﾞﾊﾞｲｽの完全な一覧についてはｿｰｽ ｺｰﾄﾞ内の"device_specific.h"を参照してください。

表6-1. 発振器周波数と内蔵RC発振器付きﾃﾞﾊﾞｲｽの特徴（発振器版による分類）

発振器版番号	代表ﾃﾞﾊﾞｲｽ	RC発振器周波数 (MHz)	CKDIV8	CLKPR
2.0	ATmega163	1.0	×	×
3.0	ATmega16	1.0, 2.0, 4.0, 8.0	×	×
3.1	ATmega128 (**注1**)	1.0, 2.0, 4.0, 8.0	×	XDIV (**注2**)
4.0	ATmega169	8.0	○	○
5.0	ATmega169P	8.0	○	○

注1: ATmega103互換動作(M103C)ﾋｭｰｽﾞは非ﾌﾟﾛｸﾞﾗﾑ(1)にされなければなりません。このﾋｭｰｽﾞのﾌﾟﾛｸﾞﾗﾑ(0)は拡張I/Oの使用、従って内蔵発振器の調節を妨げます。

注2: 前置分周ﾚｼﾞｽﾀはこれらのﾃﾞﾊﾞｲｽでXDIVと名付けられています(**訳補**:分周係数もCLKPRと異なります)。

STK501ｱﾀﾞﾌﾟﾀを使ってAT90CANﾃﾞﾊﾞｲｽ(5.0版発振器)を校正する時に、±2%の精度限度内に校正するためにSTK501の32kHzｸﾘｽﾀﾙ以外の外部ｸﾘｽﾀﾙが必要とされます。外部ｸﾘｽﾀﾙ接続の詳細についてはﾃﾞﾊﾞｲｽのﾃﾞｰﾀｼｰﾄを参照してください。

6.1. 1.x版発振器

この版は校正することができるAVR用の最初の内蔵RC発振器です。けれども非同期動作が不可能です。このため、この版の発振器を持つﾃﾞﾊﾞｲｽは外部ｸﾘｽﾀﾙを使って校正することができず、**表6-1.**に現れません。

6.2. 2.x版発振器

この版は1.0MHzの周波数で提供されます。発振器周波数と動作電圧、温度間の依存性は1.x版と比べてかなり低減されています。

6.3. 3.x版発振器

この版は35.5k製法で製造された初期のﾃﾞﾊﾞｲｽに対して導入されました。

発振器ｼｽﾃﾑは多数の発振器周波数を提供するために拡張されています。1,2,4,8MHzの周波数を持つ4つの異なるRC発振器がﾃﾞﾊﾞｲｽに存在します。この版は識票列から1MHzの校正ﾊﾞｲﾄを自動的に設定するのが特徴です。4つの異なるRC発振器が存在する事実のため、識票列に4つの異なる校正ﾊﾞｲﾄが格納されています。既定の1MHz以外の周波数が望まれる場合、OSCCALﾚｼﾞｽﾀは走行時に対応する校正ﾊﾞｲﾄが設定されるべきです。

6.4. 4.x版発振器

8MHzの単一周波数が4.0版で提供されます。後期の4.x版についてはATtiny2313に対して4MHzと8MHz、ATtiny13に対して4.8MHzと9.6MHzの2つの周波数が提供されます。OSCCALﾚｼﾞｽﾀは変更され、その結果として選択した発振器に対する周波数調整に7ﾋﾞｯﾄだけが使われます。MSBは使われません。既定校正値の自動設定とｼｽﾃﾑ ｸﾛｯｸ前置分周器が存在します。

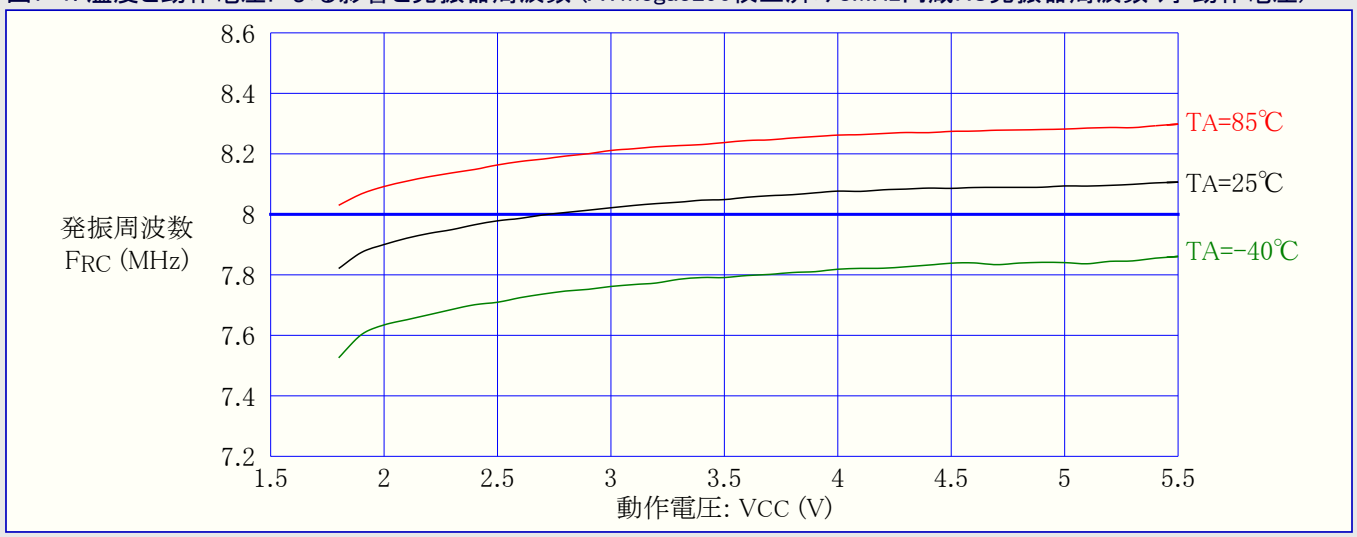
6.5. 5.x版発振器

8MHzの単一周波数が5.0版で提供されます。OSCCALﾚｼﾞｽﾀ内の全8ﾋﾞｯﾄが発振器周波数調整に使われます。既定校正値の自動設定とｼｽﾃﾑ ｸﾛｯｸ前置分周器が存在します。OSCCALﾚｼﾞｽﾀは2つの部分に分割されています。OSCCALのMSBは2つの重なっている周波数範囲の1つを選び、一方下位側7ﾋﾞｯﾄはこの範囲内で周波数を調整するのに使われます。

7. 発振器特性

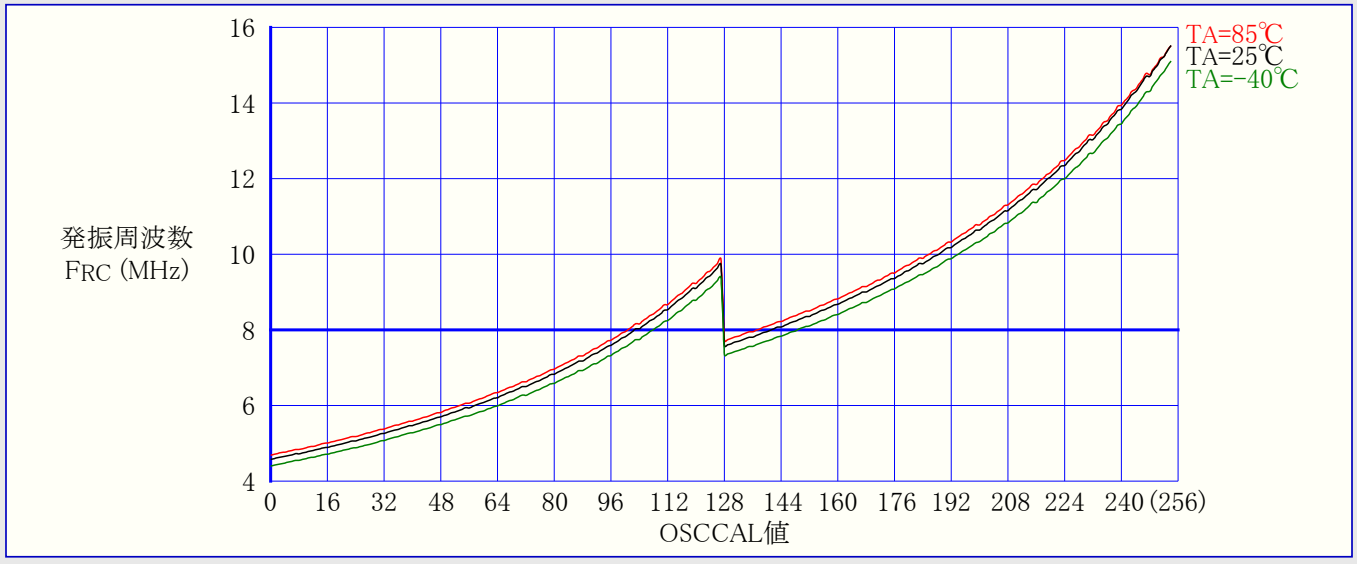
内蔵RC発振器の周波数は温度と動作電圧に依存します。この依存性の例はATmega3290の8MHz RC発振器の周波数を示す**図7-1.**で見られます。この図から見られるように、周波数は温度と動作電圧の上昇で増加します。これらの特性はﾃﾞﾊﾞｲｽ毎に変わるでしょう。特定ﾃﾞﾊﾞｲｽの詳細についてはそのﾃﾞｰﾀｼｰﾄを参照してください。

図7-1. 温度と動作電圧による影響と発振器周波数（ATmega3290校正済み8MHz内蔵RC発振器周波数 対 動作電圧）



調整可能な発振器付きの全ﾃﾞﾊﾞｲｽは発振器周波数を調整するためのOSCCALﾚｼﾞｽﾀを持ちます。OSCCALの値増加は周波数に於いて"擬似単調"増加に帰着します。これを擬似単調と呼ぶ理由はOSCCAL値の或る単一増加に関して、周波数が増加しない、または僅かに減少するからです。けれども、次の単一増加は常に再び周波数を増加します。換言すると、OSCCALﾚｼﾞｽﾀを1増加することは周波数を増加しないかもしれませんが、OSCCAL値を2増加することは常に周波数を増加します。この情報は与えられた周波数に合致する最良の校正値を見つける時に大いに関係します。OSCCAL値と発振器周波数間に関連する擬似単調の例は**図7-2.**で見られ、そしてこれはATmega3290の8MHz RC発振器です。いくつかのOSCCALﾚｼﾞｽﾀ(5.0版発振器を持つﾃﾞﾊﾞｲｽのOSCCALﾚｼﾞｽﾀ)がATmega3290と同様に発振器の調節に7ﾋﾞｯﾄを使うことに注意してください。**図7-2.**で見られるように2つの範囲が重なっています。他のﾃﾞﾊﾞｲｽは1つの連続する範囲に8(またはものによって7)ﾋﾞｯﾄ全てを使います。

図7-2. OSCCAL値の関数としてのATmega3290校正付きRC発振器周波数（8MHz内蔵RC発振器周波数 対 OSCCAL値）



全ての調整可能な発振器について、ﾃﾞｰﾀｼｰﾄで指定された基準周波数を10%よりももっと外れる発振器の調整は推奨されないことに注意することが重要です。これの理由はﾃﾞﾊﾞｲｽ内のﾌﾗｯｼｭ ﾒﾓﾘとEEPROMの書き込みに関連する内部ﾀｲﾐﾝｸﾞがRC発振器周波数に依存するからです。

RC発振器の基本特性を知れば、どの動作電圧とどの温度に於いても±1%の精度で基準周波数の10%以内で与えられた周波数にRC発振器を校正する効率的な校正ﾙｰﾁﾝを作ることが可能です。

7.1. 周波数安定時間

新しいOSCCAL値が設定されてしまうと、新しい周波数で安定させるために内蔵RC発振器に対して幾らかの時間がかかり得ます。この安定時間はRC発振器の各版で変わります。通常、発振器はOSCCALでの大きな変更よりも小さな変更に対してより早く安定します。この安定時間は5μsよりも長いことは決してありません。校正に関してどんな周波数測定をするのにも先立って、発振器を新しい周波数で安定させて置いてください。

8. 32.768kHzｸﾘｽﾀﾙを使う校正

校正は非同期ﾀｲﾏ/ｶｳﾝﾀが拡張I/O空間に配置されているか否かのどちらかに依存して6または7周期の繰り返しで実行されます。繰り返し毎に計数器が増加され、32kHz計時器の値が読まれます。(予め定義された)時計用32kHzｸﾘｽﾀﾙの計時数後、内蔵RC発振器に依存する速度の計数器は計時器値と比較されます。そして発振器校正(OSCCAL)ﾚｼﾞｽﾀの値は2分検索と近傍検索を使って望む校正周波数に従って書かれます。

繰り返し計数値と周波数間の関連は次式から得られます。

式8-1. 計数値計算

$$\text{繰り返し回数} \times \text{計数値} \times \text{RC周期} = \text{XTAL周期} \times \text{XTAL計時数}$$

ここで、

$$\text{RC周期} = \frac{1}{\text{望む周波数}} \qquad \text{XTAL周期} = \frac{1}{\text{XTAL周波数}}$$

9. 2分検索と近傍検索

3つの異なる校正法がこの応用記述に記述されています。これらは実行される検索方法が異なり、異なるｺｰﾄﾞ量と(実行)継続時間を持ちます。

7章ではいくつかの内蔵発振器が周波数範囲に於いて或る程度の周波数が重なる分割OSCCALﾚｼﾞｽﾀを持つことが指摘されていました。これらの発振器については全ての周波数に対する校正を保証するためにOSCCALの両範囲での検索が必要とされます。図7-2.は、例えば8MHzの周波数はATmega3290に於いて概ね104と144の両OSCCAL値に至るので、最適なOSCCAL値を得るために重なった周波数に対するOSCCALの両範囲での校正の比較が都合よいことかもしれないことを図解します。

9.1. 2分検索

2分検索は内蔵RC発振器に望む周波数を生成させるOSCCAL値を探すのに用いられます。2分検索は以下の方法で動きます。

1. 検索範囲の中央のOSCCAL値で始めて下さい。
2. 段階量を検索範囲の1/4に設定してください。
3. 現在の周波数が高すぎるかまたは低すぎるかを判定してください。
4. 現在の周波数が高すぎるならばOSCCALから段階量を引き、低すぎるならばOSCCALに段階量を足してください。周波数が丁度なら、更なる校正は全く必要とされないので、OSCCALを変更せずに中止してください。
5. 段階量を2で割ってください。
6. 段階量が0なら、検査は完了です。(後述される)近傍検索へ飛んでください。
7. **手順3.**へ飛んでください。

この検索法は厳密に単調な関連を持つﾃﾞｰﾀを検索する時(最悪実行時間になる時)に最適です。7章で言及したように、OSCCALと結果の発振器周波数間の関連は完全な単調ではありません。けれども、それは最適なOSCCAL値の近隣の値を探すための最も効率的な方法である2分検索に対して充分な近さです。この位置から最適値を探すのは容易です。

9.2. 近傍検索

この検索法では近接値が調べられなければなりません。この方法の使用に先行して2分検索が行われなければなりません。ｸﾛｯｸ周波数に於ける増加は2以上の値によるOSCCAL増加時に保証されるため、段階量が1の時の最終繰り返しでだけ問題が予測されます。この場合では増加されたOSCCAL値がｸﾛｯｸ周波数での減少の結果、またその逆も同様になるかもしれません。同じ方向でのもう1つ(都合2)は望む方向での周波数変更を保証します。けれどもこの段階が計数器に対する充分大きな最終段階の影響を与えるかもしれません。従って、最適なOSCCAL値を得ることを確実にするため、もう1つの段階までも行われるべきです。そしてそれらの繰り返しの各々のﾀｲﾏ/ｶｳﾝﾀ値が保存され、望むｸﾛｯｸ周期数と比較されます。望む周波数に最も近いｸﾛｯｸ周波数を生じるOSCCAL値が使われます。これは近傍検索として参照されます。近傍検索の実装には追加ｺｰﾄﾞが必要とされるため、これは要求精度がｺｰﾄﾞ量よりも重要でない時に省略することができます。

9.3. 単純検索

この方法は実装が非常に容易で、2分検索や近傍検索付きの2分検索に比べて減少されたｺｰﾄﾞ量を生じます。けれども、より高い実行時最悪状態を持ちます。この検索は次のように動きます。

1. 検索範囲の中央、換言するとOSCCAL分解能の1/2のOSCCAL値で始めて下さい。
2. 周波数が高すぎるなら、OSCCALから1を引いてください。周波数が低すぎるなら、OSCCALに1を足してください。発振器周波数が望む周波数に達しているなら、中止してください。
3. **手順2.**をn回繰り返してください。nはOSCCAL分解能の1/2です。

この最後の方法は先のものよりも明らかに遅いのですが、その小さなｺｰﾄﾞ量のため、少ないﾒﾓﾘ空間を占有します。この方法は微調整にも使うことができ、そしてそれは**手順1.**が省略され、校正は現在のOSCCALから走行します。この場合、繰り返し係数nはかなり減らすことができ、時には数段(回)が必要とされるだけかもしれません。大きな欠点は単純検索が校正の良好度を評価せず、従って完璧な校正が得られなければ、OSCCAL分解能の1/2に等しい校正周期が行われるまで続くことです。

9.4. 精度

検索の精度は使われる検索方法に依存します。

2分検索だけが使われる場合、最適なOSCCAL値が常に見つかるとは限りません。けれども、見つけた周波数は目的周波数の±1%に校正できるﾃﾞﾊﾞｲｽに対して、望む周波数の未だ±2%以内であるべきです。2分検索が単独で使われるとき、検索は望む精度制限内のｼｽﾃﾑ ｸﾛｯｸ周波数が得られた時に中止されます。これはOSCCALと内蔵RC発振器周波数間の擬似単調関係による最後の段階での問題を避けるために行われるべきです(**訳補**:そうでなければOSCCAL値の微小振動で無限反復の可能性があります)。

2分検索に加えて近傍検索が使われるとき、最適なOSCCAL値はそれが検索範囲の内側にあれば常に見つかります。そして結果のｸﾛｯｸ周波数はこの精度内に校正できるﾃﾞﾊﾞｲｽに関して、望む周波数の±1%以内でしょう。これを成し遂げるために、完全な一致が得られた、換言すると、与えられた外部ｸﾘｽﾀﾙでの計時量の間で計測されたCPU周期数が望む周波数と正確に一致する場合に検索が中止されるだけです。完全な一致を見つけることなく検索が終了されると、校正されたOSCCAL値は望む周波数に匹敵する最良値です。例え望むものにより近いｸﾛｯｸ周波数を生成するOSCCAL値があったとしても、その違いを測定することはできないでしょう。従って更なる検索は必要ありません。

精度は外部ｸﾘｽﾀﾙの計時量を増す(測定精度を改善する)ことによっても調整することができます。少ない計時は速い校正を与えますが、より低い精度です。或る限度まで数を増すことが精度を増します。OSCCALﾚｼﾞｽﾀの分解能が固定なので、より大きな計時量が必ずしも精度を上げないことに注意してください。測定精度の改善はOSCCAL(自体の)精度を改善しません。

10. 実装

本項はAVRに実装することができる走行時校正法を記述します。

C言語で書かれた動作実装が本応用記述に含まれます。ｿｰｽ ｺｰﾄﾞの完全な資料とｺﾝﾊﾟｲﾙ情報はｿｰｽ ｺｰﾄﾞと共に含まれる'readme.html'ﾌｧｲﾙを開くことによって得られます。

10.1. ﾊｰﾄﾞｳｪｱ

この応用は周波数安定性と経済性の両方で最適な選択になるように、時計用の32.768kHzｸﾘｽﾀﾙを発振器校正に使います。低周波数ｸﾘｽﾀﾙ用発振器はこの周波数で走行するｸﾘｽﾀﾙでの使用に対して設計されています。

時計用外部ｸﾘｽﾀﾙが安定時間を必要とすることに注意することが重要です。これは校正に先行してｸﾘｽﾀﾙが既に動作してない状況に適用します。

10.2. ｿﾌﾄｳｪｱ

後続項に於いて走行時校正を行うのに必要なﾌｧｰﾑｳｪｱが記述されます。

主な着想は或る周期数に対して独立した時計用32kHzｸﾘｽﾀﾙで計数器を非同期に走行することです。其の内単純なCPU繰り返しは語(計数器変数)を増加(+1)して終ります。内側繰り返しの周期数が既知で非同期計時器が正しいと仮定し、CPUは或る位置まで計数を行います。そして望む周波数に対応する数と比較し、OSCCALを通してCPU周波数を上下に調整することができます。

10.2.1. 検索範囲の定義

いくつかのﾃﾞﾊﾞｲｽは(内部)分割されたOSCCALﾚｼﾞｽﾀを持ちます。図7-2.で示されるように、ATmega3290のOSCCALﾚｼﾞｽﾀは1つが0～127、他が128～255の2つの範囲を持ちます。概ね7MHz～10MHz間の周波数へ校正する時に適切且つ確かな周波数を与えることが可能な2つのOSCCAL値があります。

校正を始め得る前に、そのﾃﾞﾊﾞｲｽのOSCCALﾚｼﾞｽﾀの分類について知ることが必要とされます。ﾏｸﾛは指定ﾃﾞﾊﾞｲｽに対してOSCCALﾚｼﾞｽﾀが2つに分けられているか否かを調べます。OSCCALが分けられている場合、検索は始めにOSCCALの下部が行われ、この始めの検索中に完璧な一致が得なれない限り、上部での検索が後続します。2つ目の検索後、2つの校正が比較され、最適な値が得られます。

可能な全ての周波数を含む1つの8(いくつかのﾃﾞﾊﾞｲｽでは7)ﾋﾞｯﾄ範囲を持つﾃﾞﾊﾞｲｽについては、1つの検索範囲だけが定義され、2重の検索と比較は必要ありません。

10.2.2. 校正初期化

校正を始める前にOSCCALﾚｼﾞｽﾀは検索範囲の1/2に等しい値によって書かれるべきです。2つの範囲のOSCCALを持つﾃﾞﾊﾞｲｽについては校正がOSCCAL下側で始まり、OSCCALは下部検索範囲の1/2を書かれます。これは2分検索を正しく動作させるために行われます。

非同期計時器はｸﾛｯｸ元として時計用外部ｸﾘｽﾀﾙを使うために正しく設定されなければなりません。計時器(ﾀｲﾏ/ｶｳﾝﾀ)がCPUｸﾛｯｸと非同期に時計用外部ｸﾘｽﾀﾙで駆動されるのを保証するため、非同期状態ﾚｼﾞｽﾀ(ASSR)の非同期動作(AS)ﾋﾞｯﾄは設定(1)されなければなりません。割り込みが使われる場合、正しい校正を保証するためにそれらは計数器(″Counter″)関数を使う前に禁止されなければなりません。

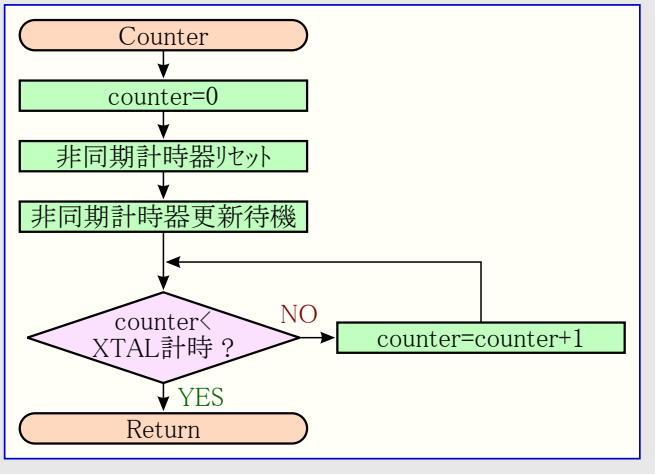
10.2.3. 計数値定義

望む周波数へ校正するとき、式8-1.内の計数値が必要とされます。この式はﾃﾞﾊﾞｲｽ固有繰り返し周期値を用いるﾏｸﾛとして実装されます。この繰り返し周期数は標準I/O領域に配置されたTCNTnﾚｼﾞｽﾀを持つﾃﾞﾊﾞｲｽに対して6、拡張I/O領域に配置されたTCNTnﾚｼﾞｽﾀを持つﾃﾞﾊﾞｲｽに対して7です。

今や計数値を得、計数器を増加するのに計数器関数を使い、そして校正を始めるために望む計数値を比較することができます。

計数器関数と非同期計時器の正しい動作を保証するために割り込みが禁止されることが重要です。

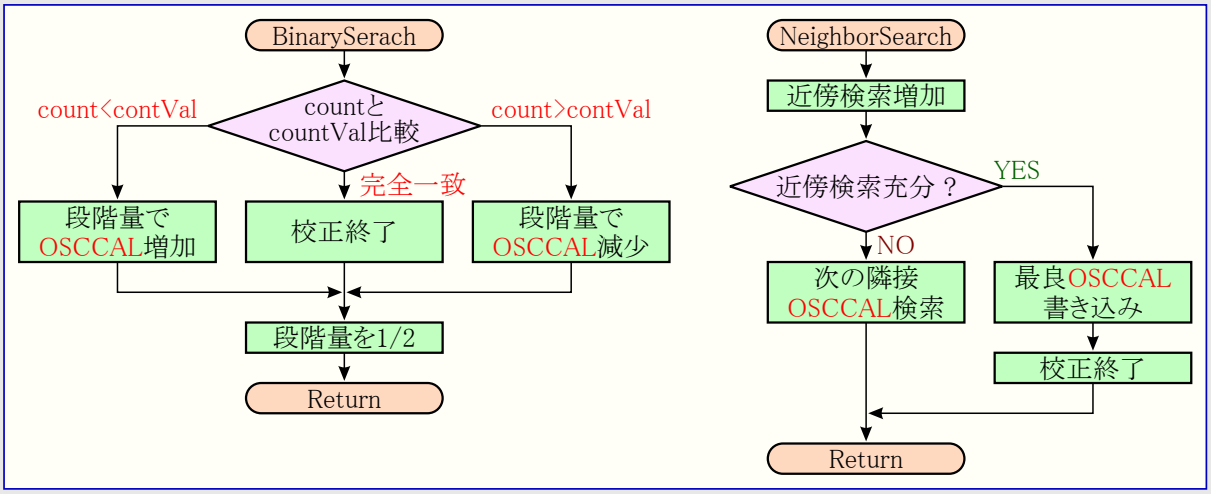
図10-1. 計数器関数



10.3. 検索

速い校正手順が必要とされる時に2分検索は他のどの検索法よりも良好に実行します。これを近傍検索と組み合わせたとき、OSCCAL値と発振器周波数間の関連が擬似単調なので素晴らしい精度が達成されます。この関連が厳密に単調でない時に2分検索は最適値に達することを保証しません。近傍検索は9.4.項で言及したように精度を大きく下げることなしに省略することができ、そしてこれはまた、走行時間を改善し、ｺｰﾄﾞ量を減らします。近傍検索は(2分検索で)完璧な校正が達成されているとき、常に省略されます。

図10-2. 2分検索関数と近傍検索関数



望む校正周波数のOSCCAL値が走行している周波数のOSCCAL値に近い時に気付くことができれば、単純検索法がより良い選択かもしれません。校正に先立ってOSCCALﾚｼﾞｽﾀを無変更のままにし、数段だけ(の検索)が必要かもしれません。これはｺｰﾄﾞ量を減らし、時間消費に関して2分検索よりも性能が勝り得るので、微調整や再校正のような場合に面白いかもしれません。

10.3.1. 校正

校正ルーチンは段階量が0に達するまで行い、または2分検索中に完璧な一致が起こる場合に終了します。段階量が0に達すると、近傍検索が始まります。最適なOSCCALは望む周波数に対応する計数値と計数器関数によって返された値間の差を評価することによって得られます。最適な値を得るため、検索終了時に最低偏差と対応するOSCCAL値が保存され、OSCCALﾚｼﾞｽﾀはその最低偏差に従って書かれます。

2つの範囲を含むOSCCALﾚｼﾞｽﾀのﾃﾞﾊﾞｲｽでは両方の範囲が望む周波数に一致する値を保持するかもしれませんが、1つの範囲は他(の範囲)よりももっと正確な校正を与えるかもしれません。可能な最良の解決策が得られることを保証するため、2つの校正が比較されます。1つのOSCCAL範囲を持つﾃﾞﾊﾞｲｽを使う時は2重検索の必要はありません。

図10-3. 2分検索と近傍検索を実行する校正ルーチン

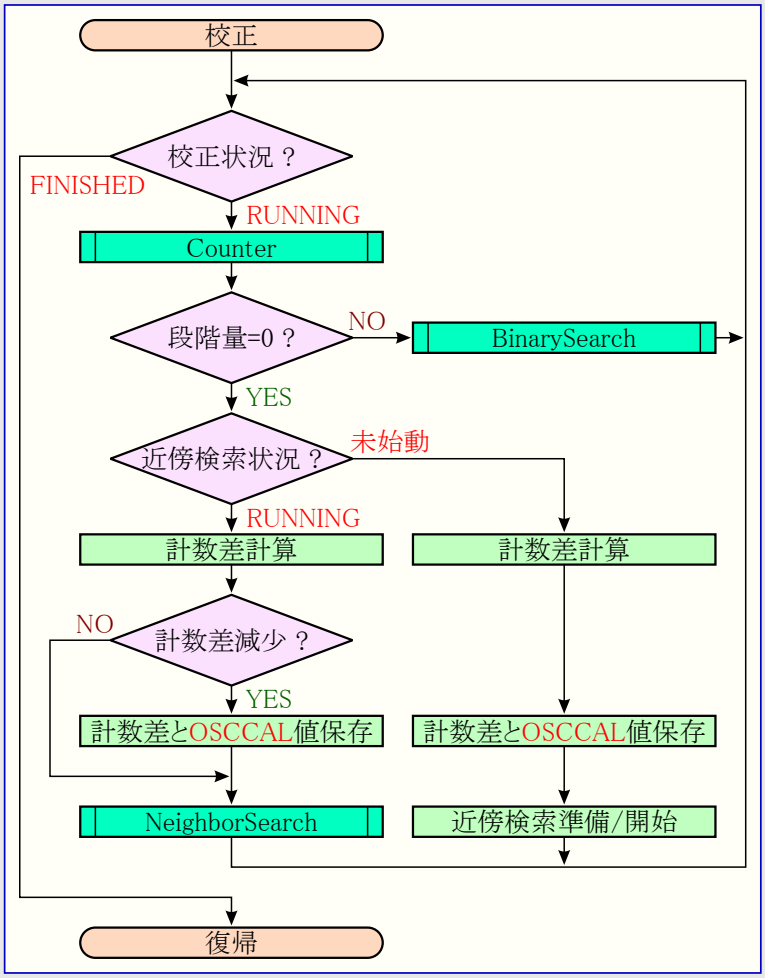
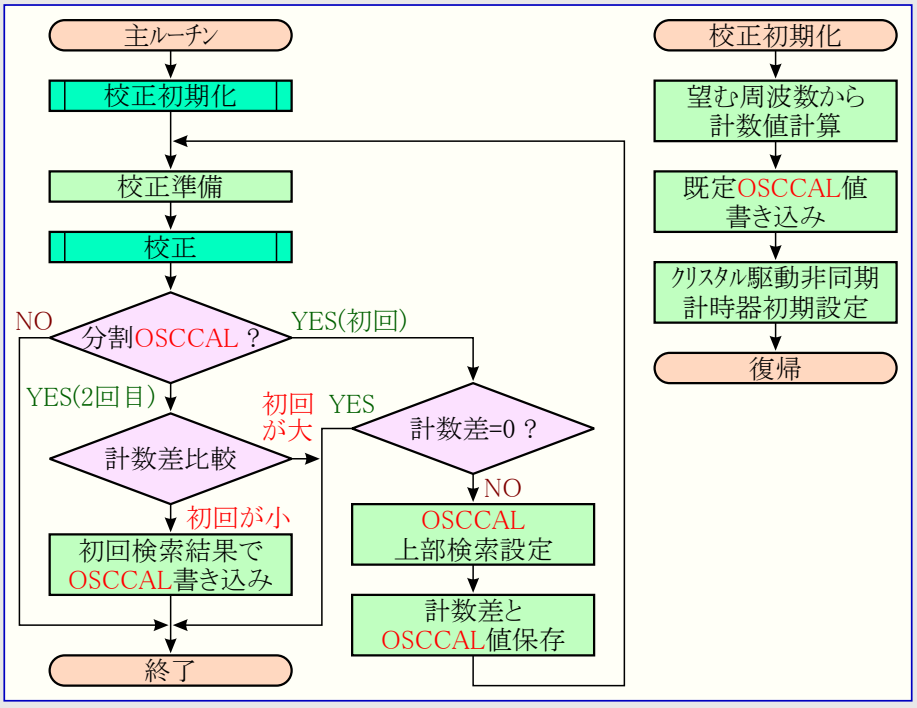


図10-4. 主ルーチン



10.4. コード量

この応用記述に含まれるソース コードは校正法、計時器分解能、使うOSCCALレジスタ形式に依存して、コンパイル時にかなり異なるコード量の結果になります。表10-1.は分割OSCCALレジスタを持つATmega3290と連続OSCCALレジスタを持つATmega32の2つのデバイスに対するいくつかのコード量の例を示します。

持続(時間)を説明するために校正周期も与えられます。校正周期は2分検索、近傍検索、単純検索を使い、OSCCALの書き込み有りまたは無しで計数器(関数)を走行することを意味します。

表10-1. コンパイルされたコード量と校正周期

デバイス	校正法	校正周期 (CPUクロック周期)	コード量 (バイト)
ATmega3290	単純検索	最大128	～270
	2分検索	最大14	～370
	近傍検索付き2分検索	最大22	～380
ATmega32	単純検索	最大128	～240
	2分検索	最大8	～280
	近傍検索付き2分検索	最大12	～290

10.5. 校正ファームウェアの性能

コードは効率に集中して書かれています。校正全体はかなり素早く実行されるべきです。従って性能は校正ファームウェアの容量と校正を完了するためにかかる時間に依存します。

校正ファームウェアは目的デバイスと校正に使われるインターフェースに依存して240～380バイトで、これは短いプログラミング時間にします。

校正ルーチンは近傍検索付きの2分検索を使って、ATmega3290に対して22、ATmega32に対して12(最悪)校正周期未満で完了されます。持続(時間)は2分検索中に完全な校正に至るか否かに依存します。

10.6. 校正クロック精度

校正の精度は外部校正クロックの精度に大きく依存します。従って使われるクリスタルの正確な周波数を測定して、インターフェース指定ソースファイル内にそれを入力することが重要です。時計用32.768kHzクリスタルの使用で最適な精度が達成されます。

11. 始める前に

ソース コードは調整可能な内蔵発振器と非同期動作付きのタイマ/カウンタを持つ現状の全てのAVRデバイスに適合します。これらのデバイスはソース コードと共に手に入る"device_specific.h"ヘッダ ファイルで一覧にされます。

www.atmel.comからAVR055用のソース コードをダウンロードし、それを解凍してください。

11.1. 校正ソース コード

全ての関数と主ルーチンは全て1つのファイル"calib_32kHz.c"に集められています。マクロ、校正指定値、フラグとデバイス指定定義は独立したヘッダ ファイルです。

11.1.1. ソース コード ファイル

根源ファイル"calib_32kHz.c"は以下のファイルを参照します。

1. デバイス指定ファイル"device_specific.h"。このファイルは根源ファイルが選択したデバイスに対応したレジスタとビットの正しい定義を使うことを保証します。
2. 校正インターフェース指定ファイル"calib_values.h"。このファイルは校正に関連する他の値と共に望む校正値を保持します。異なる校正法を選択するフラグだけでなく、いくつかのマクロも定義されます。

11.1.2. 校正実行

"calib_values.h"はコンパイル前に設定されなければならない多数のフラグを含みます。要求に対してソース コードを仕立て上げるために以下の手順を完了してください。

1. 近傍検索付き2分検索以外の検索法が望まれる場合、2つの"CALIBRATION_METHOD_xxxxx"行の1つの注釈を外してください。
2. 既定で1MHzに設定されている"CALIBRATION_FREQUENCY"値を修正することによって周波数を望む校正周波数に変更してください。
3. 時計用の32.768kHz以外のクリスタルが使われる場合、校正に使われるクリスタルに応じて"XTAL_FREQUENCY"を変更してください。
4. 精度と校正走行時間を増減するために"EXTERNAL_TICKS"が修正され得ます。±1%精度を保証するのに100計時が推奨されますが、この数は校正速度向上のために減らすことができます。非同期計時器(タイマ/カウンタ)のレジスタ容量のために最大値は255です。殆どのデバイスについては40計時が適してしますが、より大きな値は精度を向上させます。

本社

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
TEL 1(408) 441-0311
FAX 1(408) 487-2600

国外営業拠点

Atmel Asia
Unit 1-5 & 16, 19/F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
Hong Kong
TEL (852) 2245-6100
FAX (852) 2722-1369

Atmel Europe
Le Krebs
8, Rue Jean-Pierre Timbaud
BP 309
78054 Saint-Quentin-en-
Yvelines Cedex
France
TEL (33) 1-30-60-70-00
FAX (33) 1-30-60-71-11

Atmel Japan
104-0033 東京都中央区
新川1-24-8
東熱新川ﾋﾞﾙ 9F
ｱﾄﾒﾙ ｼﾞｬﾊﾟﾝ株式会社
TEL (81) 03-3523-3551
FAX (81) 03-3523-7581

製品窓口

ｳｪﾌﾞｻｲﾄ
www.atmel.com

技術支援
avr@atmel.com

販売窓口
www.atmel.com/contacts

文献請求
www.atmel.com/literature

本応用記述はAtmelのAVR055応用記述(doc8002.pdf Rev.8002D-07/08)の翻訳日本語版です。日本語では不自然となる重複する形容表現は省略されている場合があります。日本語では難解となる表現は大幅に意訳されている部分もあります。必要に応じて一部加筆されています。頁割の変更により、原本より頁数が少なくなっています。

必要と思われる部分には( )内に英語表記や略称などを残す形で表記しています。

青字の部分はﾘﾝｸとなっています。一般的に赤字の0,1は論理0,1を表します。その他の赤字は重要な部分を表します。