AN2632

tinyAVR® 1系でのIEC 60730等級B適合への指針

要点

- 等級B部品検査必要条件の一覧
- 一般等級B検査用ﾌｧｰﾑｳｪｱ ﾗｲﾌﾞﾗﾘ

説明

IEC 60730は製品の設計と動作の両面の多くを扱う家庭機器用安全規格です。この規格は安全が重要な装置に対する他の規格、例えばIEC 60335によっても参照されます。この規格でのｼｽﾃﾑ全体の遵守は動作が安全と認定されるべき機器に対して必要です。

この応用記述は規格の付属書Hに適合するための指針です。付属書Hは電気的制御に対する必要条件を扱います。IAR Embedded Workbench®とAVR® GCC用の認定されたﾌｧｰﾑｳｪｱ ﾗｲﾌﾞﾗﾘと使用例が供給されます。ﾌｧｰﾑｳｪｱ ﾗｲﾌﾞﾗﾘは規格の殆どの一般的な部分を網羅するように設定されています。検査を何時どう走らせるかと何の追加検査が必要とされるかは応用と実装選択の両方に依存します。一般的に、どの部分での障害も危険を引き起こすべきではありません。

「1. IEC 60730の付属書Hでの定義」章はIEC 60730規格からいくつかの定義を提示します。「2. 等級B必要条件」と「3. 構成要素検査」の章は等級Bｿﾌﾄｳｪｱに対する必要条件を記述します。「4. tinyAVR 1系用等級Bﾗｲﾌﾞﾗﾘ」はtinyAVR® 1系用にｲﾝｸﾙｰﾄﾞされる等級Bﾗｲﾌﾞﾗﾘとﾌｧｲﾙを紹介します。「5. ﾚｼﾞｽﾀ」、「6. ﾌﾟﾛｸﾞﾗﾑ ｶｳﾝﾀ」、「7. 割り込み処理」、「8. ｼｽﾃﾑ ｸﾛｯｸ」、「9. ﾒﾓﾘ」、「10. ｱﾅﾛｸﾞ入出力」の章はﾗｲﾌﾞﾗﾘ内の組み込み自己検査を詳細に記述します。「11. tinyAVR 1系での追加の安全機能」はtinyAVR® 1系で含まれる他の安全機能を提示します。

目次

1. IEC 60730の付属書Hでの定義

1.1. ソフトウェア等級

IEC 60730安全規格の付属書Hは機器用制御ソフトウェアの以下の3つの等級を定義します。

- 等級A – 装置の安全性に対して信頼されるべきことを意図されない制御機能 (H.2.21.1)
- 等級B – 機器でソフトウェア誤り以外の誤りが起きた場合の危険を防ぐことを意図されたコードを含むソフトウェア (H.2.21.2)
- 等級C – 他の保護的な装置を使うことなく危険を防ぐことを意図されたコードを含むソフトウェア (H.2.21.3)

保護的な制御機能で使われるソフトウェアは等級Bまたは等級Cのどちらかです。この応用記述は殆どの家電に適用される等級B制御を扱います。

1.2. 制御構造

等級B制御は定義された以下の3つの構造(H.11.12.2)の1つに従って設計することができます。

- 機能検査付き単一チャネル – それの動作に先立って検査データが機能部に導入される単一チャネル構造 (H.2.16.5)
- 周期的自己検査付き単一チャネル – 動作中に制御の構成要素が周期的に検査される単一チャネル構造 (H.2.16.6)
- 比較なしの2重チャネル – 指定された動作を実行するのに2つの相互に独立した機能手段を含む構造 (H.2.16.1)

用語のチャネルは機器でのMCU数を指します。単一チャネル構造はそれが最低費用を持つため好まれる構造の傾向があります。

機能検査付き単一チャネル構造はシステムが製造時点でだけ検査されることを意味します。この構造は機器が周期的な検査を必要とするどの構成要素も使われない場合にだけ使うことができます。周期的な検査はこれが製品の動作中に障害が検出されることを許すため、より安全な選択です。検査間の時間間隔は代表的に関連構成要素での障害に対して危険を引き起こすのにかかる時間よりも短い時間です。

比較なし2重チャネルは本質的に2つのMCUが違う作業で独立して動き、どちらかのMCUは他方が正しく動いていることを検査することができる構造です。1つの手法は2つの間で制御作業を共有するよりもむしろ1つのMCUを厳密に管理用に使うことです。

この応用記述は単一チャネル構造に集中します。

2. 等級B必要条件

等級B必要条件に準拠する機器に対して、制御ソフトウェアは表2-1.でのシステム構成要素に対して指定された障害を検出して処理しなければなりません。これらの検査と障害検出に加えて、ソフトウェアは認証を通過するために適切に資料化されなければなりません。これはプログラムの流れ、制御とデータの流れ、タイミング、障害樹状図、それと全般的な設計方針を含みます。

表2-1. 検査に対する構成要素と障害/異常の短い概要 (表 H.11.12.7)

要素	検査する構成要素	検査によって検出される代表的な異常
1	CPU	–
1.1	レジスタ	動かない
1.3	プログラム カウンタ	動かない
2	割り込み処理と実行	なし/過ぎる頻度の割り込み
3	クロック	不正な周波数
4	メモリ (注1)	–
4.1	不変メモリ	全単一ビット障害
4.2	可変メモリ	DC障害
4.3	アドレス指定	動かない
5	内部データ経路 (注1)	–
5.1	データ	動かない
5.2	アドレス指定	不正なアドレス
6	外部通信	–
6.1	データ	長すぎるハミング距離
6.3	タイミング	時間での不正な点
7	入出力周辺機能	–
7.1	デジタル入出力	H.27.1(注2)で指定される障害条件
7.2.1	A/DとD/Aの変換器	H.27.1(注2)で指定される障害条件
7.2.2	アナログ多重器	不正なアドレス指定
9	独自チップ(ASIC,GAL,ゲートアレーなど)	静的及び動的な機能特性外の何れかの出力

注1: これらはSRAM、フラッシュ メモリ、EEPROMが全て内部のためAVRに対して本質的に同じです。

注2: この表は様々な外部部品を一覧にして短絡や開放の障害が検出されなければならないかどうかを示します。

これらの検査のいくつかは必然的に応用依存です。例として、入出力周辺機能については入出力信号のもっともらしい検査を行うことが必要とされます。これは同様に応用とそれの実装の両方に依存します。従って、この応用記述で供給されるﾌｧｰﾑｳｪｱ ﾗｲﾌﾞﾗﾘは表2-1.の必要条件の全てを網羅し得ません。

3. 構成要素検査

tinyAVR® 1系に関連する表2-1.での個別構成要素のいくつかに対して条件を満たす障害検出方法と考察が本章で記述されます。

注: 機能検査が機器で使われる構成要素に対して条件を満たす方法下で一覧にされない場合に機能検査付き単一ﾁｬﾈﾙ構造は使うことができません。

3.1. CPUﾚｼﾞｽﾀ – 構成要素1.1

検査の目的：ﾚｼﾞｽﾀで動かないﾋﾞｯﾄを検出

CPUﾚｼﾞｽﾀはMCUの最も重要な部分で、欠陥ﾚｼﾞｽﾀで正しい動作が不可能なため検査されなければなりません。

障害検出に対して条件を満たす方法は静的ﾒﾓﾘ検査またはﾊﾟﾘﾃｨ検査付き語保護を用いる機能検査や自己検査です。ｲﾝｸﾙｰﾄﾞされるﾗｲﾌﾞﾗﾘでは、tinyAVR® 1系で選択肢にないﾊﾟﾘﾃｨ検査はﾊｰﾄﾞｳｪｱ実装を必要とするため、静的ﾒﾓﾘ検査が選ばれています。静的ﾒﾓﾘ検査は応用の必要条件に応じて機能検査または周期的自己検査のどちらかとして実行することができます。

3.2. ﾌﾟﾛｸﾞﾗﾑ ｶｳﾝﾀ – 構成要素1.3

検査の目的：ﾌﾟﾛｸﾞﾗﾑ ｶｳﾝﾀで動かないﾋﾞｯﾄを検出

正しく機能するﾌﾟﾛｸﾞﾗﾑ ｶｳﾝﾀはMCUで上手く走行するためにどのｿﾌﾄｳｪｱに対しても重要です。このような検査が最初の場所で正しく機能するためにﾌﾟﾛｸﾞﾗﾑ ｶｳﾝﾀを信頼するためにﾌﾟﾛｸﾞﾗﾑ ｶｳﾝﾀは動かないﾋﾞｯﾄに対して直接的に検査することができません。

障害検出に対して条件を満たす方法は、機能検査、周期的自己検査、独立時間枠監視、またはﾌﾟﾛｸﾞﾗﾑの流れの論理的な監視です。

好まれる解決策は応用の間接時間枠監視にｳｫｯﾁﾄﾞｯｸﾞを使うことです。ﾌﾟﾛｸﾞﾗﾑ ｶｳﾝﾀ障害が起きた場合、ｳｫｯﾁﾄﾞｯｸﾞ ﾀｲﾏは不適切な時でのﾘｾｯﾄまたは全くﾘｾｯﾄしないのどちらかで、結局ﾃﾞﾊﾞｲｽ ﾘｾｯﾄを引き起こします。それが不可欠な安全機能のため、tinyAVR® 1系のｳｫｯﾁﾄﾞｯｸﾞはﾌﾟﾛｸﾞﾗﾑ ｶｳﾝﾀ障害を捕まえるためにそれが信頼され得るのに先立って標準と窓の両動作で正しい動作に対して検査されなければなりません。

ｳｫｯﾁﾄﾞｯｸﾞが検査されてしまった後、常に窓動作で許可されるべきです。閉鎖期間は最低開放期間と同じ位、即ち、総期間の最低50%であるべきです。

3.3. 割り込み – 構成要素2

検査の目的：意図された割合で割り込みが発生して処理されることを確認

殆どの応用はそれらの動作に対して割り込みを信頼し、従ってそれらが意図されたその時にそれらが発生してそれによって処理されることを確認することが重要です。

障害検出に対して条件を満たす方法は機能検査や割り込みの時間枠監視を含みます。時間枠監視は一旦機器が使用中にこれが誤った動作の検出に役立つために推奨されます。どの割り込み検査も割り込み制御部を間接的に検査します。

3.4. ｸﾛｯｸ – 構成要素3

検査の目的：ｼｽﾃﾑ ｸﾛｯｸ周波数で意図せぬ変動を検出

MCUに対して正しいﾀｲﾐﾝｸﾞで正しく動作するために、ｼｽﾃﾑ ｸﾛｯｸの周波数は仕様内であることが確認されなければなりません。

障害検出に対して条件を満たす方法は周波数または時間枠の監視です。必要条件は正しい応用実行に対して問題を引き起こす周波数での発振がないのを検出することです。

AVR tinyAVR® 1系の事象ｼｽﾃﾑは外部ｸﾛｯｸ信号またはRTC(実時間計数器)によって起動されるﾀｲﾏ/ｶｳﾝﾀ捕獲を許し、参照基準ｸﾛｯｸとｼｽﾃﾑ ｸﾛｯｸ間の周波数比較を許します。

3.5. 静的RAM – 構成要素4.2, 4.3と5

検査の目的：SRAMとﾃﾞｰﾀ ﾊﾞｽでの動かないﾋﾞｯﾄと結合障害だけでなくどのｱﾄﾞﾚｽ指定の問題も検出

内部SRAMはﾃﾞｰﾀの揮発性記憶に使われ、これに関連するどの障害も機器制御に対して壊滅的であり得ます。

障害検出に対して条件を満たす方法は周期的検査やﾃﾞｰﾀ冗長性、例えば、ﾊﾟﾘﾃｨ ﾋﾞｯﾄを含みます。後者はこの目的用特定ﾊｰﾄﾞｳｪｱ実装なしでは厄介で、例えば、最善の選択として行進算法での周期的検査に任せます。

SRAM全体が1回で検査することができない場合、検査されるﾒﾓﾘ領域は結合障害の検出を許すために或る程度の重なりを持たなければなりません。

3.6. フラッシュ メモリとEEPROM – 構成要素4.1

検査の目的：不揮発性メモリでの全ての単一ビット障害を検出

全てのAVRマイクロ コントローラでは応用ソフトウェアがフラッシュ メモリに格納され、一方でEEPROMは例えば、デバイス特有設定と定数に使うことができます。デバイスに対して安全な動作のため、これらの不揮発性メモリは不正に対して調査されなければなりません。

障害検出に対して条件を満たす方法は単一または複数のチェックサムや単一ビット データ冗長性、例えばハードウェアでのパリティ検査を用いる周期的自己検査です。複数チェックサムはこれが一回でフラッシュ メモリまたはEEPROMの1領域に対して検査されることを許すため、推奨される選択です。方法は目的対象範囲に対してチェックサムを計算した後に不揮発性メモリの他の場所に格納された参照基準チェックサムとその結果の比較が続きます。

3.6.1. 自己プログラミングでの注意

書き込み中の電力障害のような事故がフラッシュ メモリを敢えて不正にするため、過酷な環境でのフラッシュ メモリの自己プログラミングの実行は推奨されません。

自己プログラミングが絶対的に必要なら、不慮の自己上書きが不可能なように施錠ビットによって保護されるブートローダでそれを行うことが推奨されます。この場合、ブートローダはそこでどのコードも実行されるのに先立ってフラッシュ メモリの応用領域のCRC検査を走行するべきです。

全てのtinyAVR® 1系デバイスはフラッシュ メモリ内のブートローダ領域が特徴です。

3.7. 外部通信 – 構成要素6

検査の目的：転送されるデータだけでなく、通信の流れとタイミングの正しさを確認

外部装置との通信は多くの応用の重要部分です。これは通信が雑音に対して影響を受けやすいため、障害の潜在的な供給元を代表し、通信線のどちらかの最終端が不正に動作するかもしれません。その結果、1つの装置での雑音や障害動作が別の装置で障害動作を引き起こさないことを保証するための処置が取られなければなりません。

転送されるデータでの障害を検出するための条件を満たす方法は繰り返し、CRC、ハミング符号のような複数ビット データ冗長性や規約検査です。通信タイミングでの障害を検出するための条件を満たす方法は時間枠監視または計画された送信です。通信の流れでの障害を検出するための条件を満たす方法はタイミングに対するそれらと同じですが、論理的監視も含みます。

要するに、制限時間、計画、転送冗長性と共に通信ドライバに基づいた状態機構が使われるべきです。

3.8. 入出力周辺機能 – 構成要素7

検査の目的：入出力が意図されるようであることと信号が正しく配線されていることを確認

全ての応用でアナログやデジタルの入出力が必要とされ、周辺機能またはMCUそれ自身のどちらかで障害動作を検出するのに使うことができます。

障害検出に対して条件を満たす方法は如何なる時でも意図される入力を得て望む出力を与えることを確認するソフトウェアを意味する妥当性検査です。

4. tinyAVR® 1系用等級Bライブラリ

ライブラリの目的はtinyAVR® 1系マイクロ コントローラに基づく安全で信頼性に足る応用の設計を簡単化することです。更に、このライブラリはお客様に対して設計と認証の処理を簡単化することが証明されています。

ライブラリは多くの異なる応用で自己検査単位部を組み込むのに充分な柔軟性があるように開発されています。使用者は応用がIEC 60730等級B規格に適合するように構成設定して使う責任があります。

ライブラリ コードはAVR GCCコンパイラとIAR™を支援します。使用者応用で自己診断ルーチンがどう組み込まれ得るかを示すために多くの例が含まれています。

ライブラリのソース コードはDoxygen(http://www.doxygen.org)で自動資料生成用に準備されています。この資料はこのL資料で提供される情報を補完します。

ソース コードはAtmel STARTコードを頼りません。コードのいくつかはそれら応用で動くためにどう設定されるかでの違いが有り得る周辺機能の構成設定を必要とします。それらの構成設定の全てが検査完了後にリセットされる訳ではありません。いくつかの検査に於いて、関数は変更したどのレジスタもリセットするように追加されています。これは使う必要がありませんが、検査後に変えられたレジスタままのレジスタの概要として扱うことができます。

検査例は釦と状態LEDを提供するためにEXT1ヘッダに接続されるOLED1 Xplainedを持つATtiny817 Xplained Proでコンパイルして試験されています。

4.1. 異常処理

可能な限り一般的であるべき検査単位部のために、使用者によって構成設定することができる多数の異常処理部が定義されています。異常は重要と非重要に分けられ、異常処理部に対する既定値を持ちます。

重要な異常は処理することができないそれら、例えば、ﾚｼﾞｽﾀ自己診断ﾙｰﾁﾝの結果を返すべきﾚｼﾞｽﾀが動かないﾋﾞｯﾄを持つ時です。重要な異常は既定によってCPUを中止して無限繰り返しの実行にして置きます。これはｳｫｯﾁﾄﾞｯｸﾞ ﾘｾｯﾄを引き起こすべきで、(この後にWDTとして参照される)ｳｫｯﾁﾄﾞｯｸﾞ ﾀｲﾏによって発行されるｼｽﾃﾑ ﾘｾｯﾄ後に取られるべき活動もまた構成設定することができます。

重要でない異常は、例えそれらが正しい動作から応用を妨げても、未だﾌﾟﾛｸﾞﾗﾑによって扱うことができるそれらです。例えば、ｱﾅﾛｸﾞ検査が失敗の場合、ﾌﾟﾛｸﾞﾗﾑはｼｽﾃﾑを安全な状態に置くために未だいくつかの活動を取ることができます。重要でない異常は既定によって全域異常ﾌﾗｸﾞを設定します。このﾌﾗｸﾞはclassb_errorと呼ばれ、ｼｽﾃﾑを安全な状態に置くために主応用によって使うことができます。これは例に従った手法です。

全ての検査で、classb_errorﾌﾗｸﾞは異常がない時に0で異常が見つかった時に0以外です。この異常ﾌﾗｸﾞは始動での既定値で上書きされることからSRAMでそれが使うﾒﾓﾘを防ぐ、NO_INIT属性を割り当てられます。ﾃﾞﾊﾞｲｽが電力を失わない限り、その値は維持されます。
classb_errorﾌﾗｸﾞは異常が見つかった時にこのﾗｲﾌﾞﾗﾘで1を書かれます。これは異常変数が何の以上が見つかったの情報も含むように変更することができます。殆どの異常処理が応用依存のため、それを行うための論理は含まれる検査に実装されていません。

4.2. ソース ファイル

ｿｰｽ ﾌｧｲﾙはwww.microchip.comを通して利用可能です。

ﾌﾟﾛｼﾞｪｸﾄは以下のﾌｧｲﾙを含みます。

- 共通ﾌｧｲﾙ:
  - oled1_xpro_attiny817.h – 例に対して釦とLEDを構成設定するための定義とｺｰﾄﾞ
  - classb_error_handler.h – 異常処理変数と異常処理関数
  - classb_compiler.h – IARとGCCの両方で等級Bﾗｲﾌﾞﾗﾘを動くようにするｺｰﾄﾞ
- ｱﾅﾛｸﾞ検査:
  - classb_analog.h – ADC、DAC、それとVREFの検査
  - main_analog.c – ｱﾅﾛｸﾞ検査用ｺｰﾄﾞ例
- CRC検査:
  - main_crc.c – CRC検査用実演応用
  - classb_crc.h – CRC検査用定義
  - classb_crc_hw.h – ﾊｰﾄﾞｳｪｱCRCSCAN単位部に対する簡単なｲﾝﾀｰﾌｪｰｽ
  - CRC_16bit_alg.h – 16ﾋﾞｯﾄCRC走査に基づく算法用ｺｰﾄﾞ
  - CRC_16bit_lookup.h – 参照表を用いることによって実行される16ﾋﾞｯﾄCRC走査用ｺｰﾄﾞ
  - CRC_32bit_alg.h – 32ﾋﾞｯﾄCRC走査に基づく算法用ｺｰﾄﾞ
  - CRC_32bit_lookup.h – 参照表を用いることによって実行される32ﾋﾞｯﾄCRC走査用ｺｰﾄﾞ
- 周波数検査:
  - classb_freq.h – 周波数検査用ｺｰﾄﾞ
  - classb_rtc.c – RTC構成設定ｺｰﾄﾞ
  - main_frequency.c – ｺｰﾄﾞ例
- 割り込み検査:
  - classb_interrupt_monitor.h – 割り込み監視検査用ｺｰﾄﾞ
  - main_interrupts.c – ｺｰﾄﾞ例
- ﾚｼﾞｽﾀ検査:
  - classb_cpu.h – ﾚｼﾞｽﾀ検査用ﾏｸﾛ
  - classb_cpu_gcc.c – GCCによってｺﾝﾊﾟｲﾙされる時に使われる検査ｺｰﾄﾞ
  - classb_cpu_gcc_asm.s – GCCﾚｼﾞｽﾀ検査用ｱｾﾝﾌﾞﾘ ｺｰﾄﾞ
  - classb_cpu_iar.c – IARによってｺﾝﾊﾟｲﾙされる時に使われる検査ｺｰﾄﾞ
  - classb_cpu_iar._asm.s – IARﾚｼﾞｽﾀ検査用ｱｾﾝﾌﾞﾘ ｺｰﾄﾞ
  - main_registers.c – ｺｰﾄﾞ例
- SRAM検査:
  - classb_sram.c – 行進X検査ｺｰﾄﾞ
  - classb_sram.h – 定義
  - main_sram.c – ｺｰﾄﾞ例
- WDT検査:
  - classb_wdt_test.c – WDT検査関数
  - classb_wdt_test.h – 初期化とmain移行の前に検査を走らせるのに使われる定義と属性
  - main_wdt.c – ｺｰﾄﾞ例

注: *.hファイルのいくつかは関数定義を含みます。これの理由はそしたらGCCがもっと容易にコードを最適化できるからです。これは特にインライン関数を持つ場合です。このように行うと、-flto最適化フラグが与えるものに匹敵するコード最適化の生成をGCCに許します。1つの違いは-flto任意選択がデバッグ情報を正しく保存しないことです。-flto任意選択は未だ多くの場合でより良いコード量を与えます。-Osと組み合わせた-flto任意選択はいくつかの場合でIARに匹敵する大きさ最適化を与えます。

5. レジスタ

この自己診断ルーチンではCPUレジスタが動かないビットと、レジスタ間ではなく個別レジスタ内のビットの各々間での結合障害について検査されます。基本的な算法は次のように動きます。レジスタの内容は検査後にレジスタ復元を可能にするようにスタックに押し込まれます。その後にレジスタのビットは状態変更を強制され、レジスタ内容がもう一度確認されます。

どちらかの確認後にレジスタの内容が不正の場合、異常フラグが掲げられます。検査が終わると、レジスタはスタックに押し込められた元の値で復元されます。レジスタ ファイル レジスタ(R0～R31)とI/Oレジスタの2つの形式のレジスタが検査されます。これらは独立したメモリ空間に配置され、違う命令でアクセスされます。

レジスタは以下の順(GCCコンパイラ実装)で検査されます。

1. 戻り値レジスタ　：R24, R25
2. 補助レジスタ　　：R31, R30
3. スタック ポインタ　：SPH, SPL
4. ステータス レジスタ：SREG (割り込みフラグを除く)
5. レジスタ　　：R29～R25とR23～R0

最初の3つの段階で、自己診断ルーチンに対して重要なレジスタでの検査が実行されます。

- 戻り値レジスタ　：この自己検査の結果を引き渡すのに使用
- 補助レジスタ1,2　：保存されなければならないレジスタの値を格納するのに使用
- スタック ポインタ　：主プログラムへ戻るのに必要
- ステータス レジスタ：CPU命令によって使われるフラグ

補助レジスタはそれらがスタック ポインタを検査するのに必要とされるため重要です。これらのレジスタで異常があった場合、デバイスは既定によって無限繰り返しの実行に留まります。けれども、代わりに異常処理部を呼ぶことが可能です。

最後の段階で、レジスタ ファイルの残りが検査されます。異常があった場合、自己診断ルーチンは異常処理部を呼んで検査の値を返します。けれども、どれかのレジスタで異常があった場合にデバイスが繰り返しで動作中止するように、重要なレジスタと同じ動きに構成設定することが推奨されます。どれかのレジスタが検査を失敗する場合、継続するコード実行は殆どの場合で推奨されません。

自己診断コードはアセンブリ コードでレジスタ操作を行うことがより簡単なためアセンブラで書かれます。

応用例はLED1をONにします。主プログラムは異常が見つかるまで複数回検査を走行する無限繰り返しです。繰り返し抜け出し条件に合致する異常が見つけられた場合、その異常が補助レジスタの場合を除き、LED1がOFFに切り替えられます。

注: この動きは実際の応用では推奨されません。検査は始動で走行されるべきで、以上が見つけられた場合に関数は活性にされるべきではありません。一般的に、レジスタ失敗の場合、デバイスがWDTをリセットすることがありそうもないため、WDTがデバイスをリセットするでしょう。

異常を模倣するため、自己診断ルーチンで中断点(ブレーク ポイント)を設定することができます。中断点で応用が停止した後、動かないビットを模倣するためにレジスタの内容を変更することができます。その後に実行を再開することができ、自己診断ルーチンは異常を検出するでしょう。

変更されたレジスタに応じて、CPUは直に自己診断ルーチン内の終わり無き繰り返しに移行するか、または全域異常変数が1に設定されるかのどちらかです。これはLEDをOFFに切り替えてその後に終わり無き繰り返しに移行させます。

6. プログラム カウンタ

プログラム カウンタは応用の間接時間枠監視にウォッチドッグ タイマ(WDT)を使って検査されます。

WDTは暴走や行き詰ったコードのような異常状況からの回復を許す、正しいプログラム動作を監視するためのシステム機能です。WDTはCPUから独立してクロック駆動される計時器です。これは予め定義された制限時間周期に構成設定され、許可されると常に走行します。

WDTが制限時間周期内にリセットされない場合、システム リセットを発行します。WDTリセットは応用コードからWDTリセット(WDR)命令を実行することによって行われます。加えて、tinyAVR® 1系のWDTは窓動作を持ちます。これはWDTがリセットされなければいけない間の総制限時間周期内に時間枠または窓を定義することを可能にします。WDTが窓の外側でリセットされる、早すぎる(窓が閉じられている)または遅すぎるのどちらかの場合、システム リセットが発行されます。標準動作と比べて、これは異常がWDTリセット命令の実効を絶えず引き起こす状況も捕らえることができます。従って、等級Bソフトウェアはこの窓動作を使うことと、閉鎖期間が総周期の最低50%であることが必要とされます。

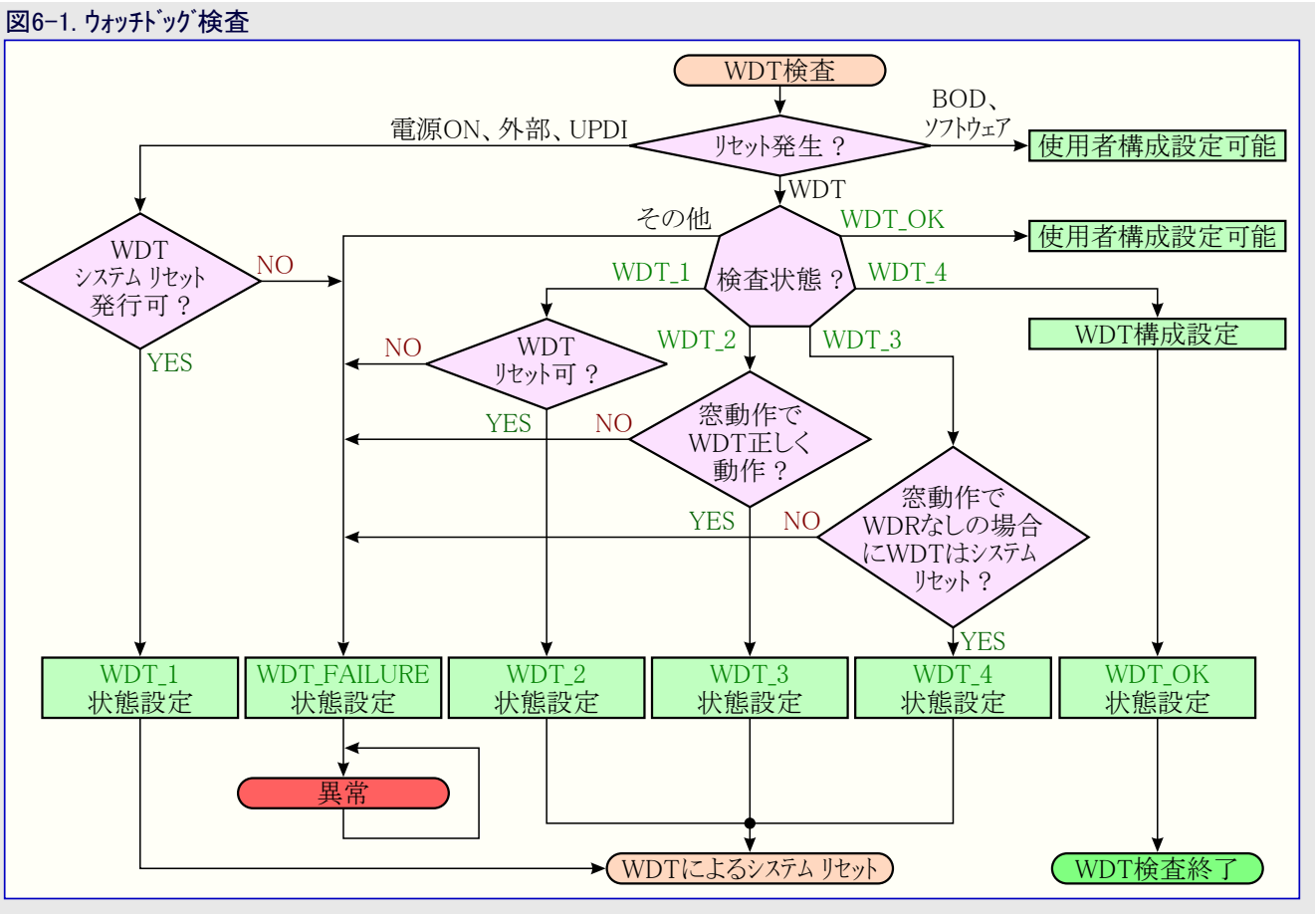
WDTは許可された場合に活動動作と全ての休止動作で走行します。これは非同期でCPUから独立したクロック元から走行し、例え主クロックが動かない場合でも動作を続けてシステム リセットを発行します。

tinyAVR® 1系にはWDT設定が事故によって変更され得ないことを保証する構成設定変更保護機構があります。

注: WDTと主クロックに同じ32kHzクロックを使うことが可能です。これが行われた場合、WDTは時間枠監視と周波数監視に対してもはや等級B必要条件を満たしません。

WDTはtinyAVR® 1系での統合安全機能のため、標準と窓の動作でこの単位部を検査する自己診断ルーチンが設計されています。これらの検査は主関数に先立つ応用の事前初期化領域でのリセット後に実行されます。この検査の流れ構成図は**図6-1.**で示されます。

図6-1. ウォッチドッグ検査



自己診断ルーチンは以下のことを確認します。

- 標準と窓の両動作でWDT時間超過後にシステム リセットが発行されます。
- 標準と窓の両動作でWDR命令を使ってWDTをリセットすることができます。
- デバイスは窓動作での時期を外したWDTリセットでリセットされます。

流れ構成図はWDT検査中にデバイスが何回もリセットされることを示します。検査段階の経緯を保つためにSRAM変数とデバイスのリセット フラグが自己診断ルーチンによって使われます。使用者は低電圧検出(BOD)やソフトウェア リセットの上で何を行うか、または検査が'WDT_OK'状態の時を監視することによって引き起こされるリセットをどう処理するかを構成設定することができます。

自己診断ルーチンはWDT発振器のタイミングを調べるためにタイマ/カウンタA型(TCA)を使います。TCAはWDTクロックと独立したクロック元を持ちます。TCAはWDTの周期を推定するのに使われ、プログラムはこの推定が間隔(T/2とT3/2)内であることを調べます。ここでのTはWDTの公称周期です。TCAはシステム クロックからクロック駆動されます。システム クロックは一般的に内部20MHzクロックです。このクロックはWDTによって使われるクロックよりも温度と電圧の変動に渡ってもっと安定です。

注: TCAはこの自己診断ルーチンによって暗黙的に検査され、TCAとWDT間の周波数での違いが50%よりも大きい場合に異常状態が設定されます。

意図される(異常なし)実行の流れは次のとおりです。

1. 電源ONまたは外部リセット後、WDTがシステム リセットを発行することができるのを調べます。検査状態を'WDT_1'に設定し、WDT時間超過によってシステムをリセットさせます。
2. WDTをリセットすることができることを調べます。検査状態を'WDT_1'に設定し、時間超過の前にWDTをリセットします。
3. 窓動作が正しく動くことを調べます。WDTを窓動作に変更します。最初に、窓が開くのを待って時間超過の前にWDTをリセットします。その後、状態を'WDT_3'に設定し、窓が閉鎖されている時にリセットを発行します。これはシステム リセットに帰着すべきです。
4. 検査状態を'WDT_4'に設定し、窓動作で構成設定されている間にWDT時間超過によってシステムをリセットさせます。
5. 応用設定に従ってWDTを構成設定します。検査状態を'WDT_OK'に設定し、主関数を続けます。

最初の段階はWDTがシステム リセットを発行することができることを確実にすることです。これはWDTを構成設定してそれがシステム リセットを発行するまで待つことによって行われます。加えて、検査の後の段階で必要とされるウォッチドッグ周期を予測するためにTCAが使われます。これは小さな周期(概ね1ms)でTCAを構成設定してシステム リセットまでTCA周期数を計数することによって行われます。プログラムが理論的な最大制限時間期間よりも長い間、システム リセットを待っているままの場合、異常状態が設定されます。

第2段階はWDTをリセットすることができることを確実にして、WDTのタイミングを調べることです。異常状態が設定されます。検査状態はこの検査段階が終わるまで異常状態です。推定したWDT周期が理論的な最小以上なことを確かめる調査が実行されます。WDTとTCA間の周波数での違いが間隔内であることを調べることは、両単位部が予期したように動いていると言う確信を提供します。次に、WDTが構成設定され、TCAはWDT周期の概ね3/4(WDT同期化遅延)待つように設定されます。これはWDTが予期したよりも早く経過しないことを調べます。その後、WDTはリセットされ、プログラムは新しいWDTリセットを発行するために再び一旦周期の3/4を待ちます。この理由はWDTをリセットする機構で何か問題があった場合にプログラムが2回目を待つ間にシステム リセットがあるからです。早期のシステム リセットは検査を異常状態にさせます。WDTリセットが確認された後、検査状態は'WDT_2'に設定され、プログラムは時間超過してシステム リセットを発行するWDTを待ちます。TCAが長すぎる間計数しているままの場合、異常が発行されます。

第3段階はWDTが窓動作で正しく動くことを調べます。これは窓動作でのWDTを構成設定することから成り、WDTがWDTリセットを発行する前に開放窓まで待ちます。WDTが開放窓だけだった場合、デバイス全体ではなく、WDTがリセットされます。この時点でデバイスがリセットされた場合、これはバックアップ開始時に検出されます。プログラムは続き、WDTリセット後にWDTは今や閉鎖窓に戻ります。検査常体が'WDT_3'に設定され、閉鎖窓の間にWDTのリセットが実行されます。窓が閉鎖されると、、WDTはシステム リセットを発行すべきです。システム リセットが発生しなければ、異常状態が設定されます。

第4段階では、WDTが再び窓動作で構成設定され、走行しているままにされます。WDTが予期するように動くなら、閉鎖窓と開放窓を通った後に時間超過を得てシステム リセットを実行するでしょう。妥当な時間内にシステム リセットが発生しない場合、異常状態が設定されます。

第5と最後の段階では、プログラムが単に窓動作でWDTを構成設定して'WDT_OK'状態を設定します。この後は構成設定した設定に従ってWDTをリセットするのは主応用の責任であることに注意してください。

検査が異常状態を設定する場合、使用者構成設定可能な異処理部を呼ぶことができます。既定により、デバイスは無限繰り返しの実行に留まります。これは信頼に足るソフトウェア応用に対して動いているWDTが重要なため、最も安全な選択と考えられます。

計数器と検査状態の変数はコンパイラがリセット後にそれらを初期化しないように宣言されます。このようにして、それらはリセットを渡って使うことができます。tinyAVR® 1系は検査の最初の反復かどうかを決めるのに使われるリセット発生を格納するレジスタを持ちます。

実演応用は釦に対して割り込みを構成設定し、LEDをONに切り替えてclassb_error変数が設定されない限りWDTがリセットされる繰り返しに留まります。この変数が設定されたなら、その後にLEDがOFFに切り替えられて応用が終わります。

釦が押されると、プログラムはWDT時間超過、従ってシステム リセットをもたらすWDTのリセットを停止します。この'予期せぬ'システム リセットは自己診断ルーチンによって見つけられるべきで、この実演ではclassb_error変数が設定され、WDTを構成設定して主応用を続けるように構成設定されます。釦押下後、LEDはOFFに切り替えられて応用は電源ONリセットまたは外部リセットまで反応しません。

6.1. 自己診断ルーチンはどう検査し得るか

自己診断ルーチンの最初の段階はシステム リセットがない限り異常状態を設定します。システム リセットを発行し得ることからそれを妨げるWDTでの問題はWDTを開始しないことによって簡単に再現することができます。異常状態が設定されてデバイスは動かなくなります。

WDTまたはタイマ/カウンタに対する発振器の周波数障害は中断点(ブレーク ポイント)を設定して、それが確信の区間の外側であるように、tc_count変数の値を変更することによって模倣することができます。

WDTリセット機構での障害はWDTがリセットされるコードの行を取り去ることによって模倣することができます。与えられたプログラムの構造で、自己診断ルーチンでの第2段階は、WDTが課された時間制限に従わない限り、異常状態を設定します。

WDT窓動作での障害はその動作形態の構成設定を取り去ることによって模倣することができます。コードはその後にWDTをリセットしてWDT周期の1/4間待ちます。その時点で、WDT周期は推定されるWDT周期(総制限時間周期は開放と閉鎖の期間の合計)以上でしょう。デバイスは異常状態を設定する前にリセットされません。

WDTによって発行されたシステム リセットに対して構成設定された活動は実演応用を通して検査することができます。釦押下はWDT超過をもたらします。その後に自己診断ルーチンは構成設定された活動を実行し、この場合はWDTを構成設定し、classb_error変数を設定してLEDをOFFに切り替えることです。

7. 割り込み処理

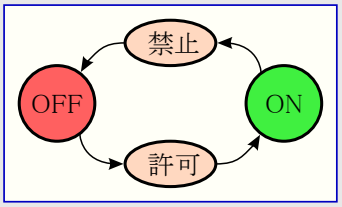
周辺機能での状態の変化である割り込み信号はﾌﾟﾛｸﾞﾗﾑ実行を変えるのに使うことができます。組み込み応用は例えば、実時間で事象に応答するために割り込みを使います。従って、ｼｽﾃﾑが割り込み機能での異常を検出できることが重要です。特に、等級B応用は割り込みが予期された毎に実行されない(または全くされない)かどうかを検出することができるべきです。

選ばれた方法は実時間計数器(RTC、CPUｸﾛｯｸから独立したｸﾛｯｸ)での時間枠監視です。要するに、監視されるどの割り込みもそれが実行される毎に計数器を増加しなければなりません。RTCは割り込み計数器が調べられる周期的な割り込みを生成します。どれかの計数器が構成設定可能な割り込み指定範囲外の場合、異常処理部が呼ばれます。

割り込み監視部はそれらの割り込みの登録されたものと活性にされたものの両方の頻度を調べます。割り込みの登録は調べる割り込みでの以下の情報、識別子、予期される頻度、許容変動を割り込み監視部に与えることを意味します。割り込み監視部はこの情報を持つﾃﾞｰﾀ構造体を作成します。割り込みを活性にすることは監視されるべき登録済み割り込みの調査を開始すべきことを監視部に告げることを意味します。割り込み監視部は各登録済み割り込みに対する状態変数を用いて個別割り込みに対してONまたはOFFに切り替えることができます。状態間で可能な遷移が**図7-1.**で示されます。

どの割り込みに対する既定状態もOFFで、この状態では割り込み監視部が割り込みの頻度を調べません。割り込みの状態は主応用が割り込み頻度の調査開始を監視部に望む時に許可(ENABLE)に設定されるべきです。次回に割り込み監視部が実行されて、その後に割り込み状態がONに変わります。これは割り込み計数器が割り込み監視部と同期されることを保証します。割り込み計数器が割り込み監視部の単一周期後で正確に増加を開始します。同様に、割り込みがもはや監視されるべきでないことを主応用が決めなら、割り込み状態はその後に禁止(DISABLE)に設定されるべきです。割り込み監視部は状態をOFFに切り替えて、次回にそれが実行されます。

図7-1. 割り込み状態



注: RTC周期毎に1つの許可/禁止要求だけであるべきです。

実装された割り込み監視部は主応用の堅牢さを更に増す2つの機能を持ちます。最初の1つは割り込みがONの場合にだけ増加する割り込み計数器で、従ってこの計数器は割り込みがOFFの間、0であるべきです。これは全ての登録済み割り込みに対して割り込み計数器によって調べられます。さもなければ、異常処理部が呼ばれます。2つ目の機能はCALSSB_STRICT定数が定義される場合にONでの割り込み許可やOFFでの割り込み禁止が異常処理部を呼ぶことです。

RTCは周期的に割り込み監視部を呼びます。全ての登録済み割り込みがそれらの状態に従って処理されます。活性な割り込みはそれらの頻度を調べられます。異常がなかった場合、割り込み監視部は計数器をﾘｾｯﾄします。異常が見つかった場合、異常処理部が呼ばれ、監視部は直ちに終わります(これは構成設定可能です)。不活性割り込みの計数器は整合性に対して0と比較されます。主応用が割り込みを活性にする要求を持つ場合、それの状態がONに設定され、その逆も同様です。状態がOFFの時に割り込み計数器が0に設定されることに注意してください。

割り込みを監視するためには以下の手順に従うべきです。

1. 割り込みは割り込みに識別子を提供することによってclassb_int_identifiersで宣言されるべきです。
2. 主応用はclassb_intmon_reg_int()を呼ぶことによって割り込みを登録しなければなりません。その後にその割り込みに対して構造体が作成されます。
3. RTCは周期的に割り込みを生成して割り込み監視部を呼び戻るように構成設定されなければなりません。
4. 監視されなければならない割り込みは各々の実行でclassb_intmon_increase()を呼ぶべきです。
5. 主応用は監視部が割り込みの調査を開始することを要求しなければなりません。これはclassb_intmon_set_state()で割り込み状態を許可(ENABLE)に変更することによって行われます。
6. いくつかの点で割り込みがもはや監視されるべきでない場合、主応用は状態を禁止(DISABLE)に変更することができます。

応用例では、ﾀｲﾏ/ｶｳﾝﾀ(TC)が監視部によって調べられる、溢れ割り込みを周期的に生成するように構成設定されます。加えて、応用は2つの釦割り込みを構成設定します。最初の物はTC割り込みの頻度を変更します。2つ目の物は監視部での割り込みを動作停止にします。応用が正しく動いていることを示すためにLEDがONに切り替えられます。応用は釦が押されない限り、LED ONでの繰り返しに留まります。最初の釦が押されると、TC割り込みの頻度が変わって、監視部が異常ﾌﾗｸﾞを設定するでしょう。主応用はその後に繰り返しを去って、LEDをOFFに切り替えます。監視部でのTC割り込みを動作停止にするために、2つ目の釦を押すことができます。この場合、最初の釦押下は未だ割り込み頻度の変更を引き起こしますが、監視部は異常を生成しません。

割り込み監視部は動いているRTCを信頼します。RTCはCPU周波数検査も支援し、それは関連するｿﾌﾄｳｪｱ単位部で調べられます。従って、RTCでの障害がないと仮定することができます。応用の信頼性を増すために、登録された割り込みは割り込み内で検査され得る計数器の最大値を持つことができます。計数器がこの値に達したなら、割り込み監視部は動いていないと仮定することができます。

監視部は以下のような各種方法で検査することができます。

- 割り込みを構成設定することができ、それから異常を生成するためにそれの頻度を変更することができます。これは応用例で実装されます。
- 割り込みの計数器はﾃﾞﾊﾞｯｸﾞ中に変更することができます。これは活性と不活性の両割り込みで行うことができます。
- CLASSB_STRICTｼﾝﾎﾞﾙが定義される場合、ﾌﾟﾛｸﾞﾗﾑは割り込みを2度、有効または無効にすることができ、これは異常を引き起こします。

8. システム クロック

自己診断ルーチンは実時間計数器(RTC:Real Time Counter)とタイマ/カウンタ(TC)を用いて実装されています。TCはこの周辺機能がCPUと同じクロック領域を持つために選ばれています。けれども、RTCは独立したクロック元からクロック駆動することができます。例えば、CPUは内部20MHz発振器から、RTCは内部32kHz発振器からクロック駆動することができます。

RTCは周期的に割り込みを生成するように構成設定されます。この割り込みに於いて、ソフトウェア16ビットTC溢れ計数器がそれの予期した値と比較され、それが与えられた構成設定可能な範囲内の場合に計数器値が解消されます。さもなければ、異常処理部が呼ばれます。

TC溢れ割り込みに於いて、16ビット溢れ計数器変数が増加されます。ソフトウェア計数器値を使う理由は、殆どの場合でTCがRTCよりもずっと高い周波数で走行するためです。TCはRTCよりも遥かに速く溢れます。この溢れ計数器が構成設定可能な閾値よりも大きい場合に異常処理部が呼ばれます。溢れ計数器がRTC割り込みで解消されるとすると、閾値よりも大きな溢れ計数器はTCの周波数と比較されるRTCの周波数の間で不整合があることを意味します。

自己診断単位部はシステム クロック、内部32kHz発振器、RTC、TCのどれかでの障害を検出します。検出されない異常の危険性は、RTCとTCの両クロックでの障害がそれらの周波数の正しい比率を与える(仮定の)筋書きに縮小されます。

応用例は以前の例と同様です。既定によって、tinyAVR 1系のデバイスは内部20MHz発振器から前置分周して落とされた3.3MHzで走行し、これは自己診断ルーチンのパラメータを構成設定する時に考慮されるシステム周波数です。釦が押されると、TC周期が変更されます。これは検査を失敗させてLEDがOFFに切り替えられます。

RTC周波数とRTC割り込み周期はRTCに関連するファイルで構成設定することができます。クロック元がCPUクロックから独立で、RTC設定に従ってRTC_INTERUP_PERIOD_TIME定数が定義されている限り、各種RTC構成設定は自己診断ルーチンと適合しています。TC単位部、前置分周器、(%での)検査に対する許容誤差、システム周波数を構成設定すること可能です。後者は主応用によって設定される実際のシステム周波数に対応しなければなりません。自己診断ルーチンはその設定に従ってクロック システムを変更しません。

この自己診断ルーチンを検査するにはいくつかの方法があります。

- 応用例は釦押下後にシステム周波数を変更することができます。
- システム周波数F_CPU定数は現実のシステム周波数と合わないように変更することができます。これは許容誤差値の変更と組み合わせることができます。
- RTCやTCでの問題を模倣するため、構成設定関数を注釈化することができます。

9. メモリ

本章は実装されているメモリと自己診断検査に対する標準的な必要条件を記述します。「9.1. 不変メモリ」項は可変メモリ、tinyAVR® 1系での内部フラッシュ メモリとEEPROMに言及し、「9.2. 可変メモリ」項は可変メモリ、即ち、SRAMに言及します。

9.1. 不変メモリ

不変メモリを検査するために巡回冗長検査(CRC:Cyclic Redundancy Check)が実装されています。これはデータでの予期せぬ異常を見つけるのに使われる異常検出技法です。この方法は一般的にデータ転送と、データとプログラムのメモリに存在するデータの正当性を決めるのに使われます。

CRC算法は入力データの流れやデータの塊を処理して、後で異常を検出するのに使うことができるチェックサム出力を生成します。これを行う一般的な2つの方法は以下から成ります。

- データのチェックサムを計算してそれを格納します。異常を検出するために同じデータで新しいチェックサムが計算され、前のものと比較されます。それらが異なる場合、異常があります。
- データのチェックサムを計算してそれをデータ領域に追加します。含まれたCRCチェックサムを加えたデータの計算されたチェックサムは一定のCRC値に帰着すべきです。新しいチェックサムが正しい値でない場合、データは変更されしまったか、または異常があります。

任意長のデータ塊に適用されるnビットCRCはnビットより長くないどの単一異常集中を検出し、全てのより長い異常集中の$1(1-2^{n})$の割合を検出します。データに異常がある場合、応用はいくつかの訂正活動を取るべきです。

ハードウェアとソフトウェアに基づく自己診断単位部が利用可能です。一般的に使われる以下の2つのCRC規格が支援されます。

- 16ビットCRC CCITT
- 32ビットCRC IEEE® 802.3

ソフトウェア実装は全てのtinyAVR® 1系デバイスによって使うことができます。この場合はCPUがデータを読んでCRCチェックサムを計算します。以下の2つのソフトウェア実装を選ぶことができます。

- 参照表：これは計算の速度向上のためにCRC参照表を用います。参照表はフラッシュ メモリの(16ビットに対して)512、(32ビットに対して)1024バイトを必要とします。
- 直接計算：これは多項式の除算を用いて各バイトに対してチェックサムを計算します。この版はフラッシュ メモリの空間を占有しませんが、参照表法よりも遅くなります。

ソフトウェア実装では、使われる32ビットCRC多項式が$EDB88320、初期剰余は$FFFFFFFFで、生成されたチェックサムはビット反転されて補数を取られます。使われるCCITT 16ビットCRC多項式は初期剰余として$0000を持つ$1021です。この場合、チェックサムはビット反転も補数もどちらもされません。

tinyAVR® 1系でのﾊｰﾄﾞｳｪｱ実装はﾌﾗｯｼｭ ﾒﾓﾘの内容だけ調査でき、EEPROMとSRAMはできません。計算されたCRCは16ﾋﾞｯﾄ幅で、CRC機構は調査されるﾌﾗｯｼｭ ﾒﾓﾘの最後の部分に存在すべきCRCﾁｪｯｸｻﾑに頼ります。誤りはISRを通して合図されるか、またはCRC完了後に調査されることが必要な状態ﾌﾗｸﾞのどちらかにすることができます。CRC単位部はﾌﾗｯｼｭ ﾒﾓﾘに対して完全なｱｸｾｽを持ち、同時にCPUは走行しません。CRCはﾋｭｰｽﾞ設定を通して始動でCPUが開始する前に動くようにもすることができ、そのように行うことが推奨されます。

CRC走査時間はどれ位のﾌﾗｯｼｭ ﾒﾓﾘが走査されるかと主ｸﾛｯｸ周波数に依存します。走査は16ﾋﾞｯﾄのﾃﾞｰﾀを3主ｸﾛｯｸ周期で処理します。加えて、開始と停止に対して大雑把に20主ｸﾛｯｸ周期の付随作業があります。応用の動作中にﾌﾗｯｼｭ ﾒﾓﾘの検査が必要とされる場合、応用が走査にどんなに時間がかかっても応答しないﾏｲｸﾛ ｺﾝﾋﾟｭｰﾀを処理することができる時に行われなければなりません。
EEPROMに格納されたﾃﾞｰﾀに対してﾁｪｯｸｻﾑを計算するのに以下の関数が利用可能です。

- CLASSB_CRC16_EEPROM_SW
- CLASSB_CRC32_EEPROM_SW

ﾌﾗｯｼｭ ﾒﾓﾘ内容のCRCを計算するのにはﾊｰﾄﾞｳｪｱ単位部が使われます。これ用のﾄﾞﾗｲﾊﾞはclassb_crc_hw.hで見つけることができます。

注: ﾊｰﾄﾞｳｪｱとｿﾌﾄｳｪｱの実装は同じCRC算法が同じﾁｪｯｸｻﾑを生成するように構成設定されています。けれども、処理時間にかなりの違いがあります。

応用例はﾌﾗｯｼｭ ﾒﾓﾘの完全性を調べるのにﾃﾞﾊﾞｲｽのCRCSCAN単位部を使います。CPUは検査が実行されている間動きません。釦は押された時にﾌﾗｯｼｭ ﾒﾓﾘのﾍﾟｰｼﾞを書くように構成設定されています。これが行われると、CRC走査が失敗して活性にされた遮蔽不可割り込み(NMI:Non-Maskable Interrupt)が実行してLEDをOFFに切り替えます。

注: NMIは不活性にすることができません。一旦異常が見つかると、絶えず活性に留まり、WDTが起動してﾃﾞﾊﾞｲｽをﾘｾｯﾄするまで、ISRのｺｰﾄﾞはISRを抜け出した直後に再び入られます。

注: CRCが失敗しないためにCRCﾁｪｸｻﾑは計算されてﾌﾗｯｼｭ ﾒﾓﾘに追加されることが必要です。これはAtmel Studioで事後構築命令を追加することによって達成することができます。この命令とそれを追加する方法は「AN2521 tinyAVR® 1系ﾃﾞﾊﾞｲｽでのCRCSCAN」で見つけることができます。

今一度の釦押下は無効です。全ての同様のCRC計算が同じﾁｪｯｸｻﾑをもたらすことを調べるために、それらを呼んで結果を比較することが可能です。ｿﾌﾄｳｪｱ実装(参照または直接の計算)用の算法は対応するﾍｯﾀﾞ ﾌｧｲﾙで設定することによって選ばれ、1つの実行中に1つの算法だけを呼ぶことができることに注意してください。或る算法や他のものを選ぶことがﾌﾗｯｼｭ ﾒﾓﾘ内容を変更するため、代わりにEEPROMに対する各種実行に渡ってﾁｪｯｸｻﾑが比較されるべきです。

9.2. 可変ﾒﾓﾘ

行進算法は可変ﾒﾓﾘを検査するために選ばれています。実装される特有更新算法は更新X検査として知られ、これは以下のように記述することができます。

⇳(w0);⇑(r0,w1);⇓(r1,w0);⇳(r0)

最初の段階は何れかの順で全てのﾒﾓﾘ位置に0を書くことです。第2段階は最低ｱﾄﾞﾚｽで始まる各ﾋﾞｯﾄで実行される以下のような3つの操作から成ります。

- ﾋﾞｯﾄを読んでそれが0であることを確認します。それが1なら、障害が発生しています。
- その位置に1を書きます。
- 次のﾋﾞｯﾄに対して繰り返します。

第3段階も(第2段階に対して)逆順のｱﾄﾞﾚｽで行われる以下のような3つ操作から成ります。

- ﾋﾞｯﾄを読んでそれが1であることを確認します。それが0なら、障害が発生しています。
- その位置に0を書きます。
- 次のﾋﾞｯﾄに対して繰り返します。

第4と最終の段階は何れかの順で全てのﾋﾞｯﾄが0であることを確認することから成ります。第2と第3の段階で使われる実際のｱﾄﾞﾚｽ順はそれらが正しく逆順である限り問題ないことに注意してください。この検査は第3と第4の段階が抜かれる最も一般的な行進C検査と同等です。

行進Xは以下の障害を検出することができます。

- ｱﾄﾞﾚｽ復号器障害
- 単一ｾﾙ障害: 動かない、変遷、ﾃﾞｰﾀ保持力の障害
- ﾒﾓﾘ ｾﾙ間の障害: 全てのではなく或る程度の結合障害(CFs:Coupling Faults)が可能

注: 可能な全ての結合障害の検出は直ちに行うことが非常に大変です。これに対してはいくつかの理由があり、それらの1つはSRAMの区画から来ています。この検査はSRAMの区画で動くことが必要なため、検査は区画間でいくつかのCFsがあるかを調べるためのｱｸｾｽを持ちません。例え区画が重なっていても、これは未だ保証されず、単に未検出となるCFsの危険を減らすだけです。

記述された行進検査はﾋﾞｯﾄ指向ﾒﾓﾘ(BOM:Bit-Oriented Memory)用に定義されています。AVRのSRAMは語指向ﾒﾓﾘ(WOM:Word-Oriented Memory)です。r0,r1,w0,W1は各々rD,r$\bar{D}$,wD,w$\bar{D}$で置き換え、ここでのDはどのﾃﾞｰﾀ背景にもでき、BOM行進検査は語間CFsを網羅するWOM行進検査に変えられます。我々の実装ではﾃﾞｰﾀ背景D=$00が選ばれています。

ﾒﾓﾘ ｾﾙ列の列内の物理的なﾋﾞｯﾄ位置を考慮することが重要です。提案された事故診断ﾙｰﾁﾝは背面の流れ($55,$AA,$33,$CC,$0F,$F0)を持つ追加更新要素を加えるように構成設定することができます。これは無制限語内CFs模式で考慮される語間状態CFsに対する網羅範囲を加えます。

SRAMでの応用ﾃﾞｰﾀでさえ検査を走らせることを可能とするために、ﾒﾓﾘは順番に検査される構成設定可能な領域数に分けられます。この検査の最も単純な動きはﾒﾓﾘ領域間に重複がない時です。この場合、全ての領域は最後の1つの可能性を除き、同じ大きさを持ちます。(緩衝部として参照される)最初のﾒﾓﾘ領域は予約されます。これは他の領域が検査されている間にそれらの内容を格納するために検査によって使われます。これは実装される更新検査が破壊的であると考えれば必要です。

行進X算法が同時に1つのﾒﾓﾘ領域で走行するとすれば、語間CFsが検出されない可能性を減らすのに、使用者構成設定可能なﾒﾓﾘ領域間の重複が存在します。ﾒﾓﾘ領域が検査される毎に、更に直前の領域の部分も検査されます。これはそれが先頭領域のため、緩衝部に適用されないことに注意してください。緩衝部の大きさは(重複しない)前の場合に対して拡張されることが必要です(それに応じて2つ目の領域の大きさが減らされます)。

ﾒﾓﾘ検査が応用でどう組み込まれ得るかを示す応用例が含まれます。ｼｽﾃﾑの正しい動きを合図するLEDはONで、その後に異常がない限り、ﾌﾟﾛｸﾞﾗﾑはSRAMﾒﾓﾘが検査される繰り返しに留まります。

自己診断ﾙｰﾁﾝは次のように検査することができます。

- 行進X算法を実行する関数で多数の中断点(ﾌﾞﾚｰｸ ﾎﾟｲﾝﾄ)を設定します。
- 語間結合障害の各種形式を形成するようにいくつかのﾒﾓﾘ位置の内容を変更します。

検査単位部はその後に異常ﾌﾗｸﾞを設定し、これは主繰り返しを抜け出してLEDがOFFに切り替えられることを応用にもたらします。

9.3. 更なる網羅範囲

SRAM、ﾌﾗｯｼｭ ﾒﾓﾘ、EEPROMがtinyAVR® 1系で内部ﾒﾓﾘであることを考慮すると、前に記述された自己診断ﾙｰﾁﾝはﾒﾓﾘ ｱﾄﾞﾚｽ指定と内部ﾃﾞｰﾀ経路に対する規格の仕様(表H.11.12.7の補助構成要素4.3と構成要素5)を網羅します。

10. ｱﾅﾛｸﾞ入出力

ﾗｲﾌﾞﾗﾘで利用可能なｱﾅﾛｸﾞ検査は内部信号だけで実行されます。ﾗｲﾌﾞﾗﾘでは以下のような異なる2つの検査関数が利用可能です。

1. ﾃﾞﾊﾞｲｽで内部的に配線されたDACによって生成されるﾚﾍﾞﾙのADC読み取り。各周辺機能用の参照基準電圧はﾊﾞﾝﾄﾞｷﾞｬｯﾌﾟ電圧から生成されます。
2. 上と同様ですが、ADC参照基準電圧としてVDDです。

全ての検査が等級B準拠のために走らせることが必要とされる訳ではありません。

検査1.と2.は非常に似ていますが、ﾃﾞﾊﾞｲｽが安定なVDDで動くなら、検査2.はやや大きいﾊﾞﾝﾄﾞｷﾞｬｯﾌﾟ電圧の偏差を見つける機会を持つため、やや広い網羅範囲を与えます。検査2.はVDDでのより高い程度の精度を必要とするため、電池応用やVDDが検査中に非常に雑音が多くなるような応用に使われるべきではありません。VDDに於いて雑音が多い、または或る回路から別の回路で大きな変動がある場合、ADCからの変換値のより高い許容範囲も必要とされるかもしれません。

検査1.は検査2.と3.よりも幾分簡単です。DACとADCの両方がﾊﾞﾝﾄﾞｷﾞｬｯﾌﾟ電圧から参照基準を得るため、ﾊﾞﾝﾄﾞｷﾞｬｯﾌﾟが不正かどうかを検査することができません。ﾊﾞﾝﾄﾞｷﾞｬｯﾌﾟによって供給される2つの独立した参照基準生成部が異なるまたは不正の場合を検出することができます。

全ての検査はDAC、ADC、参照基準電圧系を検査しますが、検査が失敗した場合にそれらのどれが失敗したかを告げることができません。例えば、これが使われる唯一の構成要素の場合で検査が正しいADCの動きを確認するのに使われる場合、DACやDACに対する参照基準生成部での異常の存在は検査失敗を引き起こし得ます。これはその後に例えこれが安全であっても、応用が実行されなくなることを引き起こし得ます。

ｺｰﾄﾞ例は検査状態を示す各々2つのLEDと共に継続する繰り返しで検査1.と検査2.を実行します。検査が失敗した場合、対応するLEDをOFFに切り替えます。

検査ﾙｰﾁﾝは以下のような異なる方法で確認することができます。

- ADCｸﾛｯｸはより速く変更することができます。
- 許容変換範囲は検査が失敗するように調節することができます。
- DACまたはADCのどちらかへの参照基準電圧を変更することができます。

例ではBUTTON1割り込みがADCへの参照基準電圧を1.1Vに変更(検査1.)し、一方でBUTTON2はADCとDACの両方の参照基準を1.1Vに変更(検査2.)します。これはADC変換結果を予想限度外にさせて、検査が失敗するでしょう。

11. tinyAVR 1系での追加の安全機能

tinyAVR® 1系は以下のように安全性を増すために使うことができるいくつかの追加機能を持ちますが、等級B必要条件のどれも直接的に扱いません。

- 構成設定変更保護(CCP:Configuration Change Protection)は制限時間ｺｰﾄﾞ手順に従わない場合に或るI/Oﾚｼﾞｽﾀの変更と不揮発性ﾒﾓﾘの読み書きを防ぎます。
- ﾋｭｰｽﾞは応用ｿﾌﾄｳｪｱによって容易に読んで確認することができます。
- 安全電圧以下での動作からﾃﾞﾊﾞｲｽを守る低電圧検出(BOD:Beown-Out Detection)、割り込み付きの設定可能な電圧ﾚﾍﾞﾙ監視部(VLM:Voltage Level Monitor)、電源ONﾘｾｯﾄ(POR:Power-On Reset)
- ｸﾛｯｸ ｼｽﾃﾑはｸﾛｯｸ元を切り替える前にに発振器の安定性を調べます。

12. 追補 – 用語と略語

- ﾊﾐﾝｸﾞ距離 – 2進数に対して、A XOR Bの結果での1の数として説明することができます。
- 静的ﾒﾓﾘ検査 – 不揮発性ﾒﾓﾘの検査
- 時間枠監視 – 設定された時間区間内で起こる事象のいくつかの継続する検査
- 行進算法 – RAM書き込みを順次行って後で複数段階で内容を確認する算法
- 結合障害(CFs:Coupling Faults) – 一方だけの変更時に両方を変更させる、違うﾒﾓﾘ区部間の繋がり

13. 謝辞

本資料での”IEC 60730”とこの規格からの他の全ての定義と部分は、ｽｲｽ、ｼﾞｭﾈｰﾌﾞ、IEC (www.iec.ch)、2007年 著作権© IEC 60730-1 3.2版を参照します。

著者はIEC 60730-1 3.2版(2007年)国際公開から情報を複製することの許諾に関して国際電気標準会議(IEC:International Electrotechnical Commission)に感謝します。

14. 参照と提唱する文献

- “IEC 60730-1：家庭と同様の使用のための自動的な電気的制御”3.2版2007年3月、国際電気標準会議
- Michael Lee BushnellとVishwani D. Agrawalによる“ﾒﾓﾘと混合された信号VLSI、ﾃﾞｼﾞﾀﾙ用電気的検査の重要性”
- “組み込み自己検査に対する設計者の手引き”、C. E. Stroud、2002年 Kluwer学術出版
- AVR040:電磁適合性(EMC)設計の考察
- AVR042:ﾊｰﾄﾞｳｪｱ設計の考察
- AN2521:tinyAVR® 1系ﾃﾞﾊﾞｲｽでのCRCSCAN

15. 改訂履歴

資料改訂	日付	注釈
A	2018年2月	初版資料公開
B	2018年10月	Atmel STARTへのﾘﾝｸを削除してmicrochip.comへのﾘﾝｸを追加。 「**tinyAVR 1系での追加の安全機能**」章にVLMを追加。
C	2019年7月	お客様のご意見に基づいてﾌﾟﾛｸﾞﾗﾑ ｶｳﾝﾀの話題を更新。
D	2019年12月	11.章から「ﾃﾞﾊﾞｲｽ仕様を超える動作条件を検出するのに内部温度感知器を使うことができます。」を削除。

Microchipｳｪﾌﾞ ｻｲﾄ

Microchipはhttp://www.microchip.com/で当社のｳｪﾌﾞ ｻｲﾄ経由でのｵﾝﾗｲﾝ支援を提供します。このｳｪﾌﾞ ｻｲﾄはお客様がﾌｧｲﾙや情報を容易に利用可能にするのに使われます。利用可能な情報のいくつかは以下を含みます。

- **製品支援** – ﾃﾞｰﾀｼｰﾄと障害情報、応用記述と試供ﾌﾟﾛｸﾞﾗﾑ、設計資源、使用者の手引きとﾊｰﾄﾞｳｪｱ支援資料、最新ｿﾌﾄｳｪｱ配布と保管されたｿﾌﾄｳｪｱ
- **全般的な技術支援** – 良くある質問(FAQ)、技術支援要求、ｵﾝﾗｲﾝ検討ｸﾞﾙｰﾌﾟ、Microchip設計協力課程会員一覧
- **Microshipの事業** – 製品選択器と注文の手引き、最新Microchip報道発表、ｾﾐﾅｰとｲﾍﾞﾝﾄの一覧、Microchip営業所の一覧、代理店と代表する工場

製品変更通知ｻｰﾋﾞｽ

Microchipの製品変更通知ｻｰﾋﾞｽはMicrochip製品を最新に保つのに役立ちます。加入者は指定した製品系統や興味のある開発ﾂｰﾙに関連する変更、更新、改訂、障害情報がある場合に必ず電子ﾒｰﾙ通知を受け取ります。

登録するにはhttp://www.microchip.com/pcnへ行って登録指示に従ってください。

お客様支援

Microchip製品の使用者は以下のいくつかのﾁｬﾈﾙを通して支援を受け取ることができます。

- 代理店または販売会社
- 最寄りの営業所
- 組み込み解決技術者(ESE:Embedded Solutions Engineer)
- 技術支援

お客様は支援に関してこれらの代理店、販売会社、またはESEに連絡を取るべきです。最寄りの営業所もお客様の手助けに利用できます。営業所と位置の一覧はこの資料の後ろに含まれます。

技術支援はhttp://www.microchip.com/supportでのｳｪﾌﾞ ｻｲﾄを通して利用できます。

Microchipﾃﾞﾊﾞｲｽ ｺｰﾄﾞ保護機能

Microchipﾃﾞﾊﾞｲｽでの以下のｺｰﾄﾞ保護機能の詳細に注意してください。

- Microchip製品はそれら特定のMicrochipﾃﾞｰﾀｼｰﾄに含まれる仕様に合致します。
- Microchipは意図した方法と通常条件下で使われる時に、その製品系統が今日の市場でその種類の最も安全な系統の1つであると考えます。
- ｺｰﾄﾞ保護機能を破るのに使われる不正でおそらく違法な方法があります。当社の知る限りこれらの方法の全てはMicrochipのﾃﾞｰﾀｼｰﾄに含まれた動作仕様外の方法でMicrochip製品を使うことが必要です。おそらく、それを行う人は知的財産の窃盗に関与しています。
- Microchipはそれらのｺｰﾄﾞの完全性について心配されているお客様と共に働きたいと思います。
- Microchipや他のどの半導体製造業者もそれらのｺｰﾄﾞの安全を保証することはできません。ｺｰﾄﾞ保護は当社が製品を”破ることができない”として保証すると言うことを意味しません。

ｺｰﾄﾞ保護は常に進化しています。Microchipは当社製品のｺｰﾄﾞ保護機能を継続的に改善することを約束します。Microchipのｺｰﾄﾞ保護機能を破る試みはﾃﾞｼﾞﾀﾙ ﾐﾚﾆｱﾑ著作権法に違反するかもしれません。そのような行為があなたのｿﾌﾄｳｪｱや他の著作物に不正なｱｸｾｽを許す場合、その法律下の救済のために訴権を持つかもしれません。

法的通知

ﾃﾞﾊﾞｲｽ応用などに関してこの刊行物に含まれる情報は皆さまの便宜のためにだけ提供され、更新によって取り換えられるかもしれません。皆さまの応用が皆さまの仕様に合致するのを保証するのは皆さまの責任です。**Microchipはその条件、品質、性能、商品性、目的適合性を含め、明示的にも黙示的にもその情報に関連して書面または表記された書面または黙示の如何なる表明や保証もしません。**Microchipはこの情報とそれの使用から生じる全責任を否認します。生命維持や安全応用でのMicrochipﾃﾞﾊﾞｲｽの使用は完全に購入者の危険性で、購入者はそのような使用に起因する全ての損害、請求、訴訟、費用からMicrochipを擁護し、補償し、免責にすることに同意します。他に言及されない限り、Microchipのどの知的財産権下でも暗黙的または違う方法で許認可は譲渡されません。

商標

品質管理ｼｽﾃﾑ

Microchipの品質管理ｼｽﾃﾑに関する情報についてはhttp://www.microchip.com/qualityを訪ねてください。

日本語© *HERO* 2019.

本応用記述はMicrochipのAN2632応用記述(DS00002632D-2019年12月)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。

世界的な販売とサービス

米国	亜細亜/太平洋	亜細亜/太平洋	欧州
本社 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 技術支援: http://www.microchip.com/support ｳｪﾌﾞｱﾄﾞﾚｽ: http://www.microchip.com	**ｵｰｽﾄﾗﾘｱ - ｼﾄﾞﾆｰ** Tel: 61-2-9868-6733 **中国 - 北京** Tel: 86-10-8569-7000 **中国 - 成都** Tel: 86-28-8665-5511 **中国 - 重慶** Tel: 86-23-8980-9588 **中国 - 東莞** Tel: 86-769-8702-9880 **中国 - 広州** Tel: 86-20-8755-8029 **中国 - 杭州** Tel: 86-571-8792-8115 **中国 - 香港特別行政区** Tel: 852-2943-5100 **中国 - 南京** Tel: 86-25-8473-2460 **中国 - 青島** Tel: 86-532-8502-7355 **中国 - 上海** Tel: 86-21-3326-8000 **中国 - 瀋陽** Tel: 86-24-2334-2829 **中国 - 深圳** Tel: 86-755-8864-2200 **中国 - 蘇州** Tel: 86-186-6233-1526 **中国 - 武漢** Tel: 86-27-5980-5300 **中国 - 西安** Tel: 86-29-8833-7252 **中国 - 廈門** Tel: 86-592-2388138 **中国 - 珠海** Tel: 86-756-3210040	**ｲﾝﾄﾞ - ﾊﾝｶﾞﾛｰﾙ** Tel: 91-80-3090-4444 **ｲﾝﾄﾞ - ﾆｭｰﾃﾞﾘｰ** Tel: 91-11-4160-8631 **ｲﾝﾄﾞ - ﾌﾟﾈｰ** Tel: 91-20-4121-0141 **日本 - 大阪** Tel: 81-6-6152-7160 **日本 - 東京** Tel: 81-3-6880-3770 **韓国 - 大邱** Tel: 82-53-744-4301 **韓国 - ｿｳﾙ** Tel: 82-2-554-7200 **ﾏﾚｰｼｱ - ｸｱﾗﾙﾝﾌﾟｰﾙ** Tel: 60-3-7651-7906 **ﾏﾚｰｼｱ - ﾍﾟﾅﾝ** Tel: 60-4-227-8870 **ﾌｨﾘﾋﾟﾝ - ﾏﾆﾗ** Tel: 63-2-634-9065 **ｼﾝｶﾞﾎﾟｰﾙ** Tel: 65-6334-8870 **台湾 - 新竹** Tel: 886-3-577-8366 **台湾 - 高雄** Tel: 886-7-213-7830 **台湾 - 台北** Tel: 886-2-2508-8600 **ﾀｲ - ﾊﾞﾝｺｸ** Tel: 66-2-694-1351 **ﾍﾞﾄﾅﾑ - ﾎｰﾁﾐﾝ** Tel: 84-28-5448-2100	**ｵｰｽﾄﾘｱ - ｳﾞｪﾙｽ** Tel: 43-7242-2244-39 Fax: 43-7242-2244-393 **ﾃﾞﾝﾏｰｸ - ｺﾍﾟﾝﾊｰｹﾞﾝ** Tel: 45-4450-2828 Fax: 45-4485-2829 **ﾌｨﾝﾗﾝﾄﾞ - ｴｽﾎﾟｰ** Tel: 358-9-4520-820 **ﾌﾗﾝｽ - ﾊﾟﾘ** Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79 **ﾄﾞｲﾂ - ｶﾞﾙﾋﾝｸﾞ** Tel: 49-8931-9700 **ﾄﾞｲﾂ - ﾊｰﾝ** Tel: 49-2129-3766400 **ﾄﾞｲﾂ - ﾊｲﾙﾌﾞﾛﾝ** Tel: 49-7131-72400 **ﾄﾞｲﾂ - ｶｰﾙｽﾙｰｴ** Tel: 49-721-625370 **ﾄﾞｲﾂ - ﾐｭﾝﾍﾝ** Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 **ﾄﾞｲﾂ - ﾛｰｾﾞﾝﾊｲﾑ** Tel: 49-8031-354-560 **ｲｽﾗｴﾙ - ﾗｰﾅﾅ** Tel: 972-9-744-7705 **ｲﾀﾘｱ - ﾐﾗﾉ** Tel: 39-0331-742611 Fax: 39-0331-466781 **ｲﾀﾘｱ - ﾊﾟﾄﾞﾊﾞ** Tel: 39-049-7625286 **ｵﾗﾝﾀﾞ - ﾃﾞﾙｰﾈﾝ** Tel: 31-416-690399 Fax: 31-416-690340 **ﾉﾙｳｪｰ - ﾄﾛﾝﾊｲﾑ** Tel: 47-72884388 **ﾎﾟｰﾗﾝﾄﾞ - ﾜﾙｼｬﾜ** Tel: 48-22-3325737 **ﾙｰﾏﾆｱ - ﾌﾞｶﾚｽﾄ** Tel: 40-21-407-87-50 **ｽﾍﾟｲﾝ - ﾏﾄﾞﾘｰﾄﾞ** Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 **ｽｳｪｰﾃﾞﾝ - ｲｪｰﾃﾎﾞﾘ** Tel: 46-31-704-60-40 **ｽｳｪｰﾃﾞﾝ - ｽﾄｯｸﾎﾙﾑ** Tel: 46-8-5090-4654 **ｲｷﾞﾘｽ - ｳｫｰｷﾝｶﾞﾑ** Tel: 44-118-921-5800 Fax: 44-118-921-5820