AVR® シミュレータ

AVR® シミュレータ

AVR®ｼﾐｭﾚｰﾀはどのﾊｰﾄﾞｳｪｱも使うことなくｺｰﾄﾞを走らせてﾃﾞﾊﾞｯｸﾞすることができるMicrochip AVRﾃﾞﾊﾞｲｽ用のｿﾌﾄｳｪｱ ｼﾐｭﾚｰﾀです。全ての命令、割り込み、ﾁｯﾌﾟ上の殆どの入出力単位部を含み、CPUをｼﾐｭﾚｰﾄします。

このｼﾐｭﾚｰﾀはﾃﾞﾊﾞｯｸﾞ目的対象としてMicrochip Studio応用とで動作します。これは走行、中断、ﾘｾｯﾄ、単一段階実行、中断点設定、変数監視のような通常のﾃﾞﾊﾞｯｸﾞ命令を使うことを使用者に許します。入出力、ﾒﾓﾘ、ﾚｼﾞｽﾀの表示がｼﾐｭﾚｰﾀを使って完全に整っています。

このｼﾐｭﾚｰﾀはﾊｰﾄﾞｳｪｱ設計から直接的に派生したﾃﾞﾊﾞｲｽのｿﾌﾄｳｪｱ模式に基づき、従って実ﾃﾞﾊﾞｲｽに対して周期精度です。

目次

1. Microchip Studioでのシミュレータの使用

1.1. 概要

シミュレータは他のどのデバッグ ツールとも同様に動くように作られたデバッグ ツールです。他のツールと同じ一覧で選ぶことができ、どのハードウェア接続も必要ないため、直ちに使うことができます。デバッグ作業が始められると、シュミュレータは選んだデバイスのシミュレータ模式を読み込みます。

シミュレータ模式の性能は実デバイスに比べて遅いですが、ソフトウェア作成のため、実デバイスでは利用できないいくつかの付加的デバッグの可能性を使用者に与えます。

1.2. デバッグ作業でのシミュレータの使用

シミュレータでのデバッグ作業が開始されると、選んだデバイスのシミュレータ模式が読み込まれ、使用者の応用で書かれます。応用メモリに応用が読み込まれた後、電源ONリセット(POR)が模式に適用されます。従って、シミュレータはPORリセット フラグ設定と共にリセット ベクタから開始します。デバッグ作業を始めるのに使用者が"Start Debugging/Continue (F5)(デバッグ開始/継続 (F5))"を使った場合、今やシミュレータ模式が走行して応用の実行を始めます。使用者が"Start Debugging and Break (Alt+F5)(デバッグを開始して中断 (Alt+F5))"を使った場合、シミュレータはmain()関数の始めに達するまで走行を開始してその後に中断を行います。

Microchip StudioのReset(リセット)釦はシミュレータにPORを適用します。電源ONリセット(POR)中にフラッシュ メモリとEEPROMの内容は失われません。Microchip Studioでリセットが行われている時は、一般的に実行がmain()関数の始めで中断します。リセットを実行する前に逆アセンブリ表示に切り換えることにより、実行はリセット ベクタで停止します。ウォッチドッグがリセットを生成するように設定された場合、リセット ベクタでの中断点(ブレーク ポイント)設定を通してそれを受け取ってシミュレータを走行させることができます。

走行しているシミュレータに取り付く、またはそれから取り外す/切断する方法はなく、背面で走行し続けます。プログラミング ダイアログでのように、シミュレートされるデバイスは使用者がデバッグを停止した時に存在しなくなります。これはMicrochip Studioの将来版で外されるかもしれない制限です。

このシミュレータがソフトウェア型のため、デバイス上のOCDシステムによって制限されず、ハードウェア ツールが共有しない次のような或る能力を持ちます。

- デバイスとOCDシステムの制限に関わらない無制限数の中断点
- 目的対象走行中の中断点の設定と削除
- 例えOCDシステムがなくても、全てのデバイスでデバッグができ、追跡を支援
- OCDシステムで至ることができない場所へのアクセス
- ハードウェアでの対象物、例えば周期計数器を持たない機能を提供
- 実ハードウェアと違い、シミュレータはMicrochip Studioのメモリ表示を使っての直接変更をフラッシュ メモリとEEPROMに許します。
- 未だ存在しないデバイスのシミュレートが可能(試供品が入手可能前の早期支援)

1.2.1. シミュレータでのI/O表示部使用

I/O表示部は一般的に他のツールと同じように動きます。けれども、I/Oアドレス指定法がシミュレータ模式の内部信号に割り当てられるため、少しの変わった点があります。これらは各種ハードウェア レジスタから読み込み専用、書き込み専用、読み書きのレジスタまたはレジスタ内のビット領域で起き、また、変わった点の書き込み動作形態を持ちます。一般的に、同じレジスタ内の各種ビット領域はそれらが異なるハードウェア位置に割り当てられるため、違うように動き得ます。

I/O表示部でレジスタが変更されると、レジスタの新しい値は目的対象に書かれ、その後に読み戻し、その読み戻した値がI/O表示部に示されます。この理由のため、示された値は様々な理由のために書かれた値と違うかもしれません。

- 読み書きアクセスを持つ通常のレジスタ/ビット領域では、それの値を変更すると、新しい値が直ちに示されます。
- いくつかの場合、読み書きアクセスが違うハードウェア位置になります。これの影響は例え(多くのデバイスでのUSARTのUDRレジスタのように)レジスタが書かれても見える影響がないレジスタ/ビット領域の変更を試みることや、(2重緩衝を持つレジスタのように)影響が1周期以上遅らされることが有り得ます。
- レジスタ/ビット領域が読み込み専用の場合、それを変更する試みは無効です。
- レジスタ/ビット領域が書き込み専用の場合、一般的に常に0として読みます。
- いくつかのレジスタ/ビット領域は設定、解除、切り替えのような特別な書き込み動作を持ちます。これは'1'の値の書き込みがビットでそれらの操作の1つを実行する一方で、'0'書き込みが無効なことを意味します(これはレジスタでの単一ビット変更に読み-変更-書き手順の必要をなくします)。度々、このようなレジスタが違うアドレスで通常のレジスタとして反映されます。これらの場合、シミュレータは一般的に例えそれが読み込み専用として記述されていても、この通常レジスタへの書き込みを許します。I/O表示部から変更を行う時に通常レジスタを使うことがよりもっと簡単です。
- 可能な時に、シミュレータは例えそれらが読み込み専用または書き込み専用として記述されていても、度々、レジスタ/ビット領域への読み/書きアクセスを許します。これの特に重要な1つの例が割り込み要求フラグです。これらは度々、読み込み専用を意図されますが、割り込み要求フラグが割り込みの発生であるようにハードウェアが設計されている場合(これは良くある場合)、シュミュレータからそれらへの書き込みを許すことが割り込みの起動を許します。この機能は要因生成、例えば、ADC割り込みは変換した値をADCデータ レジスタへ書くことを通して起動し、その後にADC割り込み要求フラグの設定(1)を通してADC割り込みを起動することができます(ADCデータ レジスタとADC割り込み要求フラグは一般的にデータシートで読み込み専用として記述されます)。

- いくつかの32ビット型では周辺機能単位部I/Oレジスタ アクセスのいくつかが内部信号を直接アクセスする代わりにチップ上のバスを使います。これらの場合、シミュレータはOCDに基づくエミュレータと同じ制限の対象になります。例えば、読み込み専用レジスタは読み込み専用に留まり、いくつかのレジスタは実際のレジスタが書かれ得る前に違うレジスタに書かれる保護様式が必要とされ、いくつかの周辺機能はそれらがアクセスされ得る前に明示的に許可されるそれらのクロックを持たなければならず、などです。デバイス毎の詳細は「2.2. デバイスと系列特有の問題」で見つけることができます。
- 時々、ハードウェア設計は妥当な努力での正しいビット領域割り当てを不可能にします。これは最も頻繁に書き込みアクセスに影響を及ぼしますが、時々、読み込みでもです。このような欠陥は「2.2. デバイスと系列特有の問題」で記述されます。
- 最後に、I/O表示部での無反応や他の欠陥があるレジスタはシミュレータ模式でのI/O割り当てでのバグによって起き得ます。

注意:

上記に関する注意

1. シミュレータI/O割り当てでのバグや欠落は模式の機能ではなく、レジスタのデバッガ表示部にだけ影響を及ぼします。目的対象で動いている応用はこのようなバグによって影響を及ぼされません。
2. この部分でI/O表示部について述べられた全てはメモリ表示部がI/O位置をアクセスするのに使われる時にも同様に真実です。この2つの間の違いは体裁だけです。
3. I/O表示部について一覧にされた制限は「1.2.2. シミュレータ刺激」で記述される刺激にも影響を及ぼします。

1.2.2. シミュレータ刺激

1.2.2.1. 序説

シミュレータ刺激はシミュレーション中で与えられた時間に何れかのレジスタやメモリ位置を読みまたは書きすることができる簡単なスクリプト ファイルを書く方法です。刺激ファイルはシミュレーション中の何時でも開始することができ、ファイルの最後、またはシミュレータ作業が終わる時まで継続します。

注: Atmel Studio 6.1で開始すると。AVR® Studio 4で見つかるファイル シミュレータが再導入されます。AVR Studio 4では2つの刺激変種があります。これはシミュレータ2で使わる最新のものです。古い方は支援されません。

ファイル シミュレータの特徴:

- タイミングは絶対周期計数器値に代わって遅延を単位として表されます。
- ポート レジスタだけでなく、どのI/Oレジスタも(シミュレート)に割り当てることができます。
- ビット単位割り当てを使ってI/Oレジスタの個別ビットをシミュレートすることができます。
- 指令が柔軟性を増します。
- 1つの刺激ファイルを開いて別の刺激ファイルを実行することができます。

1.2.2.2. Microchip Studioからの使用

刺激入力ファイルはMicrochip Studioのエディタのようなテキスト エディタを使って先に予め準備されなければなりません。刺激ファイルに対して.stim拡張子を使うことが推奨されます。デバッグを開始する前に、Project Properties(プロジェクト特性)頁のTool(ツール)タブで刺激ファイルが選ばれます。デバッグ作業中、メニュー選択のDebug(デバッグ)⇒Set Stimulifile(刺激ファイル設定)を使うことによって新しい刺激ファイルを選ぶことができます。

刺激生成部はメニュー選択のDebug(デバッグ)⇒Execute Stimulifile(刺激ファイル実行)を使ってMicrochip Studioから開始することができます。この任意選択は活動しているデバッグ作業中にだけ利用可能です。この動作と最後の刺激ファイルが閉じられることによって使い尽くされる刺激入力間の時間は刺激作業と呼ばれます。

刺激作業中、中断点、単一段階実行のような通常のデバッグ機能を使うことができます。刺激は応用プログラムが実行される(自由走行または単一段階実行)時に適用されます。刺激作業が終えられた後にデバッグ作業が継続される場合、(新しい刺激作業が開始されない限り)刺激なしで継続します。

デバッグ作業を終わる他に、明示的に活動している刺激作業を中止する方法はありません。

刺激作業からの出力はMicrochip Studio内のOutput(出力)枠に送られます。Show output from(からの出力を表示)の引き落とし一覧からFileStimuliProvider(ファイル刺激提供部)を選んでください。最初に刺激ファイルが実行される時にFileStimuliProvider(ファイル刺激提供部)が作成され、Microchip Studioが閉じられるまでその場所に留まります。出力はデバッグ作業に渡って保持されます。これが望まれないなら、手動で解消されなければなりません。出力枠形式の説明については「1.2.2.2.1. FileStimuliProvider出力枠形式」をご覧ください。

記録は刺激ファイルで指令を使うことによってのみ開始することができます。現在、記録を設定したり開始するためのGUI能力はありません。Microchip Studioは現在、出力枠ではなく、ファイルへの記録だけを支援します。

1.2.2.2.1. FileStimuliProvider出力枠形式

刺激作業が走行している時に、この作業からの全ての出力はFileStimuliProvider(ファイル刺激提供部)枠に出力されます。全ての出力行は#00000028形式の時刻印で始まります。これは10進数での周期計数器の値です。この枠で出力される3つの出力形式があり、これは以降に示されます。

刺激ファイルの開閉と作業の終わり

刺激ファイルが開かれるまたは閉じられる時に必ず出力で記録されます。この出力の最初の行は最初の刺激ファイルの開始です。これはこのように見えるかもしれません。

```
#000000000 Opened file 'C:¥Project¥Test¥test.stim' as [ 0]
```

大括弧内の数はファイルに割り当てられた0で始まるファイル番号です。この番号は他のメッセージでファイルを参照するのに使われます。別の刺激ファイルが開かれた場合、番号1を得、以下同様です。

ファイルが閉じられると、次のような刺激メッセージが現れます。

```
#000000028 Closed file 'C:¥Project¥Test¥test.stim' [ 0]
```

最後の刺激ファイルが閉じられてしまうと、刺激作業が終了され、それ以上刺激は生成されません。最後に閉じられたファイルが最初のファイルである必要がないことに注意してください。最後のファイルが閉じられると、以下のメッセージが生成されます。

```
#000000036 All stimuli files closed
```

指令送り返し

刺激ファイルからの全ての文はそれが実行される時に送り返されます。送り返された文は時刻印とファイル番号が先行し、このように見えるかもしれません。

```
#000000016 [ 0] PINB ^= 0x03
#000000016 [ 0] #4
#000000020 [ 0] PINA = 0x01
```

この例ではPINBとPINAへの割り当てが4周期による時間で分離され、全ての指令がファイル0から読まれます。

異常と警告のメッセージ

異常/警告メッセージは異常が検出されたファイル名と行番号を明示的に参照し、このように見えるかもしれません。

```
#000000006 [ 0] log foo.bar
Error: L:¥Project¥Test¥test.stim(6): Syntax error
```

1.2.2.3. 刺激ファイル形式

刺激ファイルは1行に対して1つの指令で刺激指令を含む単純なASCII平文ファイルです。注釈の他には遅延、割り当て、指示の3種類の指令だけがあります。

1.2.2.3.1. 遅延

遅延は#文字に後続するCPUクロック周期での遅延持続時間によって指定されます。#20は20クロック周期の遅延を意味します。遅延の使用が時間で指令を分離する唯一の方法です。遅延によって分離されない指令は同時に、即ち、同じクロック周期内で実行されます。現在の実装では刺激がCPU単一段階(実行)間でだけ評価され、これが複数周期命令の中間で終わる場合に指定されたよりも遅延が長いかもしれないことを意味します。

1.2.2.3.2. 割り当て

割り当てはI/Oレジスタへ新しい値を割り当てるのに使われます。演算子が**表1-1.**で一覧にされます。

表1-1. 刺激割り当て演算子

文	説明
目的対象 = 値	直接割り当て: 目的対象を値に等しく設定
目的対象 |= 値	ビット単位OR割り当て: 値で1のビットが目的対象で設定(1)され、残りのビットは無変化
目的対象 &= 値	ビット単位AND割り当て: 値で0のビットが目的対象で解除(0)され、残りのビットは無変化
目的対象 ^= 値	ビット単位XOR割り当て: 値で1のビットが目的対象で切り替え(反転)され、残りのビットは無変化

目的対象はI/O割り当てでのI/Oレジスタの数値メモリ アドレスにすることができます。tinyAVR®とmegaAVR®のように平坦なI/O構造を持つ単純なデバイスについてはデータシートで見つかるようなレジスタ名を使うこともできます。

複雑なI/O構造を持つデバイス(XMEGA®、UC3、SAM)については今の処アドレスを使うことが推奨されます。アドレスを決める最も簡単な方法はI/O表示部に持って来て望むレジスタを選ぶことです。アドレスはI/O表示部から複写することができます(望むレジスタを選択して右クリックし、"Copy Address(アドレス複写)"を選んでください)。

値はC構文に従った10進数、8進数、16進数で指定したどれかの数値定数にすることができ、また、*source形式を持つこともできます。これを使う時にsource構文はI/Oレジスタの名称またはメモリ アドレスです。現在の解釈部は式を支援しません。

```
GPIOR0 = *GPIOR1      // 許されます!。
GPIOR0 = *GPIOR1 + 1 // 許されません!。
```

1.2.2.3.3. 指示

指示は$文字によって始められ、指令が後続します。指示は刺激の実行と記録の様々な面の制御に使われます。
現在支援される指示が表1-2.で一覧にされます。

表1-2. 刺激指示

指示	引数 (注1)	説明
$stimulate	ファイル名	新しいファイルから刺激の読み込みを開始します。新しいファイルは現在のファイルと並行して読まれます。これは現在、刺激作業内で複数の刺激ファイルを開く唯一の方法です。
$quit		現在の刺激ファイルを閉じます。ファイルの残りは破棄され、ファイルが閉じられます。(ファイルの最後到着と同じ)
$break		ﾌﾟﾛｸﾞﾗﾑ実行を中断。刺激ファイルは開いたままで、刺激はﾌﾟﾛｸﾞﾗﾑ実行が再開される時に再開されます。
$repeat	回数	閉路繰り返しを開始。$endrep指示までを回数分繰り返します。
$endrep		閉路繰り返しの最後
$log	I/Oﾚｼﾞｽﾀ 遮蔽	ﾚｼﾞｽﾀ記録設定。遮蔽が指定された場合、記録は遮蔽でのﾋﾞｯﾄが変化した時にだけ更新されます。遮蔽は同じｱﾄﾞﾚｽに対する以前のどの遮蔽ともORされます。記録は$startlog指示が実行されるまで開始されません。
$unlog	I/Oﾚｼﾞｽﾀ 遮蔽	ﾚｼﾞｽﾀ記録停止。遮蔽が指定された場合、遮蔽でのﾋﾞｯﾄだけが記録を停止します。
$startlog	ファイル名 書き込み動作	その名称のファイルに記録開始。書き込み動作は任意選択で既定動作はファイルへ追加です。
$stoplog		記録停止
$fuse	ｱﾄﾞﾚｽ 値	ｱﾄﾞﾚｽのﾊﾞｲﾄに値を設定(注2)。ﾋｭｰｽﾞ ｱﾄﾞﾚｽは一般的に0から始まります。
$reset	形式	ﾃﾞﾊﾞｲｽ ﾘｾｯﾄ。可能なﾘｾｯﾄ形式が右で一覧にされます。
$memload	ファイル ｾｸﾞﾒﾝﾄ nocheck	ファイルの内容をﾒﾓﾘに読み込み。AVR設計ではﾃﾞｰﾀを読み込む場所を選ぶのにｾｸﾞﾒﾝﾄを指定することができます。指令の最後にnocheckを加えた場合、ファイルでのﾁｪｯｸｻﾑ誤りが無視されます。
$memdump	ファイル ｱﾄﾞﾚｽ 大きさ ｾｸﾞﾒﾝﾄ	ｱﾄﾞﾚｽで始まるﾒﾓﾘの内容をファイルに書き出し、大きさのﾊﾞｲﾄ数を書き出します。任意選択で、どのﾒﾓﾘを書きだすかを選ぶためにｾｸﾞﾒﾝﾄを指定します。

表1-3. 記録書き込み動作

形式	説明
a	ファイルへ追加 (既定)
o	どの既存ファイルへも上書き

表1-4. リセット形式

形式	説明
p	電源ONﾘｾｯﾄ (POR)
e	外部ﾘｾｯﾄ (EXT)
b	低電圧検出 (BOD)
s	ｽﾊﾟｲｸ (AVR XMEGAのみ、外部ﾘｾｯﾄと同じ)

表1-5. ﾒﾓﾘ ｾｸﾞﾒﾝﾄ

ｾｸﾞﾒﾝﾄ	説明
s	ﾃﾞｰﾀ ﾒﾓﾘ (既定)
f	ﾌﾗｯｼｭ ﾒﾓﾘ
e	EEPROM
i	I/O

表1-6. ﾒﾓﾘ ｾｸﾞﾒﾝﾄ

ｾｸﾞﾒﾝﾄ	説明
s	ﾃﾞｰﾀ ﾒﾓﾘ (既定)
f	ﾌﾗｯｼｭ ﾒﾓﾘ
e	EEPROM
i	I/O

注1: 複数の引数は空白(ｽﾍﾟｰｽ)によって分離されます。
注2: ﾋｭｰｽﾞ変更後、変更の効果を得るには電源ONﾘｾｯﾄが適用されなければなりません。
注3: この指令はAtmel Sudio 6.1 SP2で導入されました。(訳注:原書に於いて対応する注3なし)

記録の登録はどの理由に対しても記録されるI/Oﾚｼﾞｽﾀが値を変える時に必ず生成されます。記録形式は刺激形式に適合し、これは記録出力を刺激入力として使うことができることを意味します。
注: 相対ﾊﾟｽは最初の刺激ファイルのﾃﾞｨﾚｸﾄﾘに相対します。

1.2.2.3.4. 注釈

注釈は//で始められ、行の最後まで続きます。塊注釈(/* ～ */)は支援されません。

1.2.2.4. 既知の問題

刺激ファイル

注意:

レジスタ名とアドレス間の割り当ては一意のレジスタ名を持つ平坦なI/O構造のデバイスに対してだけ確実に動きます。ドット(.)表記は動きません。代わりに数値アドレスを使ってください。

- 32ビット デバイスのいくつかのレジスタの記録が支援されないかもしれません。これはデバイス毎の基準で記述されます。
- 割り当てでは演算子(=など)が空白(スペース)によって囲まれなければなりません。
- 刺激解釈部は刺激入力ファイルの最終行が改行で終端されていない場合に失敗します。
- 8ビット デバイスで16ビットまたは32ビットのレジスタ組への値割り当て、例えば、ADCへの割り当てはADCLとADCHに分けて割り当てなければなりません。「1.2.2.5. 刺激作業例」での例をご覧ください。
- 異常報告は望まれる多くを取り残しています。
- 現在の実装で刺激ファイルがCPU単一段階(実行)間でだけ評価されるため、刺激のタイミングは遅延仕様に比べて1または2周期外れるかもしれません。
- 開いている間に刺激ファイルを編集しようとする場合の共有違反

1.2.2.5. 刺激作業例

作業1: 簡単なAVR®プログラム

以下の例は最低2つの汎用入出力(GPIO)ポート(PORTAとPORTB)を持つどのATtinyとATmegaデバイスでも動くでしょう。I/O表示部を有効にしてPORTAとPORTBの単位部を示すように設定して走らせるべきです。

このプログラムはPORTAを出力として、PORTBを入力として設定し、その後にPINBに存在するものを何でも読んで、それを1増してPORTAでそれを出力することを繰り返します。

```
reset:  RJMP    start
start:  LDI     R16, 0xFF
        OUT     DDRA, R16       // PORTA => 出力
        CLR     R0
        OUT     DDRB, R0        // 入力 <= PORTB
loop:   IN      R0, PINB        // 何かの活動のために刺激を必要とします。
        INC     R0
        OUT     PORTA, R0
        RJMP    loop
```

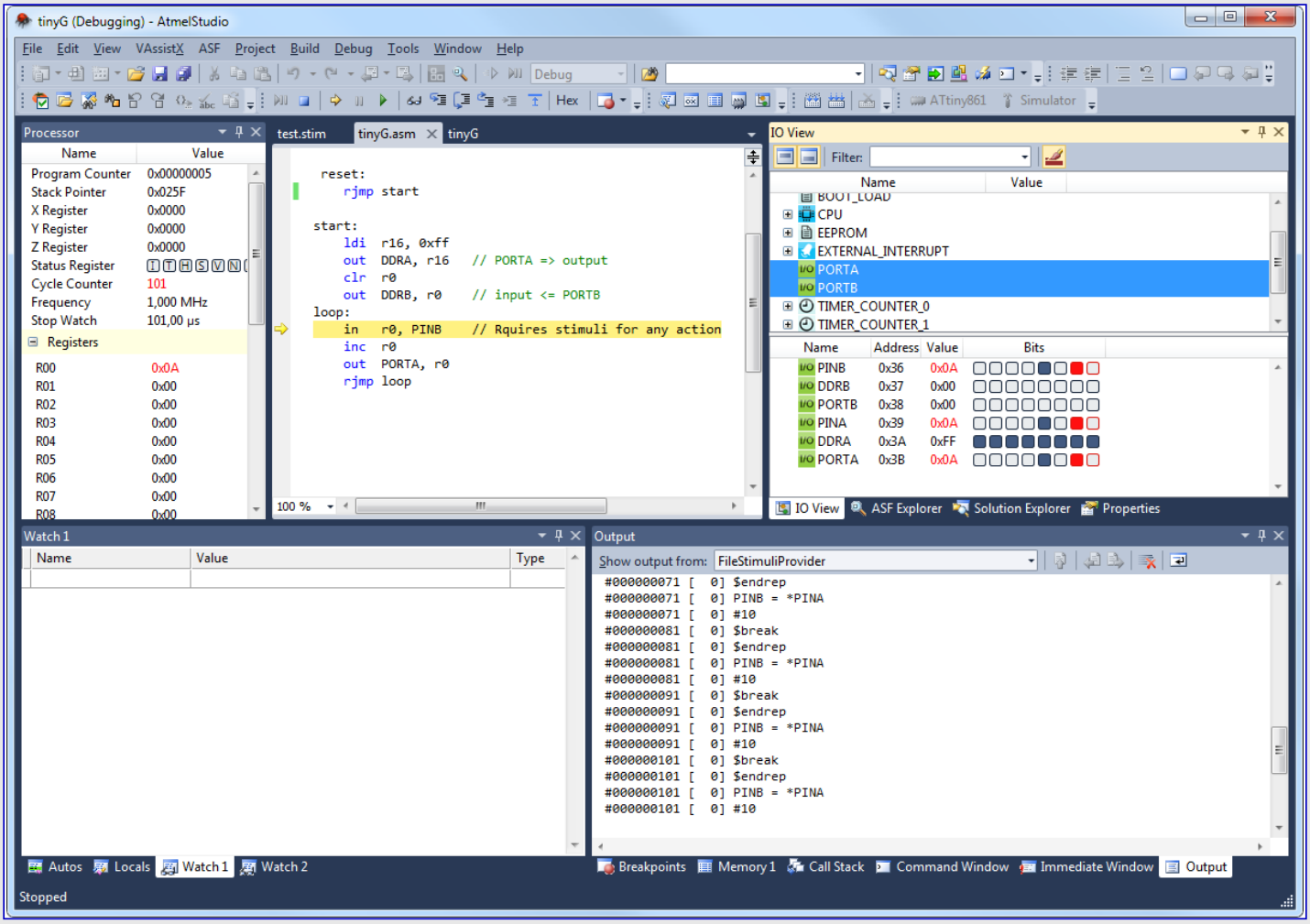
刺激なしで、このプログラムは永遠にPINBから0を読んでPORTAに1を出力します。

次に以下の刺激ファイルを適用してください。

```
// 刺激ファイル例、PINAでのども出力もPINBに戻して供給
#10
$repeat 10
    PINB = *PINA
    #10
    $break
$endrep
```

今やPINBは第10周期毎にPORTAからの出力によって駆動され、この影響はI/O表示部で直接見ることができます。繰り返しの内側の$break指示のため、プログラムはプログラムでのどの中断点もなしに10周期毎に停止します。

図1-1. 例からの刺激作業の捕獲画面



作業2: 簡単なAVR®記録

以下の例はATmega328P用に作られていますが、殆どのATtinyまたはATtmegaのﾃﾞﾊﾞｲｽで動くでしょう。このﾌﾟﾛｸﾞﾗﾑはTIMER0(ﾀｲﾏ/ｶｳﾝﾀ0)を開始し、その後、繰り返しに入り、常にPINDでの値に1を加算してその結果をPORTDに割り当てます。

```
#include<avr/io.h>

int main(void)
{
    DDRD = 0xFF;
    TCCR0B = (1<<CS00);
    while(1)
    {
        PORTD = PIND + 1;
    }
}
```

以下の刺激を適用すると、PORTDとPINDの値の記録を開始する記録ﾌｧｲﾙを作成します。20周期後、PINDとTCNT0の記録に変更し、更にもう20周期後、60周期間PINDだけの記録を継続します。

```
$log PORTD
$log PIND
$startlog mega328p_log_output.stim
#20
$unlog PORTD
$log TCNT0
#20
$unlog TCNT0
#60
```

```
$stoplog
$break
```

これは以下の内容でmega328p_log_output.stimと呼ばれるファイルを生成します。

```
#6
PORTD = 0x01
#1
PIND = 0x01
#4
PORTD = 0x02
#1
PIND = 0x02
#4
PORTD = 0x03
#1
PIND = 0x03
#3
TCNT0 = 0x10
TCNT0 = 0x11
#1
TCNT0 = 0x12
#1
PIND = 0x04
TCNT0 = 0x13
#1
TCNT0 = 0x14
#1
TCNT0 = 0x15
#1
TCNT0 = 0x16
#1
TCNT0 = 0x17
#1
PIND = 0x05
TCNT0 = 0x18
#1
～
```

作業3: PWM出力を記録

以下の例はATmega328P用に作られていますが、殆どのATtinyまたはATtmegaのﾃﾞﾊﾞｲｽで動くでしょう。このﾌﾟﾛｸﾞﾗﾑは高速PWM動作でTIMER0(ﾀｲﾏ/ｶｳﾝﾀ0)を開始します。PD6でのPWM信号の出力はPB0によって制御されます。PB0での'1'がPWM出力を許可し、PB0での'0'がそれを禁止します。

```
#include<avr/io.h>

int main(void)
{
    DDRB = 0x00;
    DDRD = 0xFF;
    PORTD = 0x00;
    OCR0A = 0x20;
    TCCR0A = (1<<COM0A0 | 1<<WGM01 | 1<<WGM00);
    TCCR0B |= (1<<CS00);
    while(1)
    {
        if(PINB & 0x01)
            TCCR0B |= (1<<WGM02);
        else
            TCCR0B &= ~(1<<WGM02);
    }
}
```

以下の刺激を適用すると、PINBとPINDの値の記録を開始する記録ファイルを作成します。20周期後、TCNT0の記録も開始します。5周期遅れて、PWM出力を開始するためにPB0に'1'を適用し、更にもう5周期後、TCNT0の記録をOFFにします。次の200周期間、PD6のPWM出力はPB0に'0'を設定することによってOFFにされるのに先立って記録されます。その後、それが中断するのに先立って更にもう200周期間継続します。

```
$log PINB
$log PIND
$startlog mega328p_log_output.stim
#20
// TIMER0計数器記録
$log TCNT0
#5
// PB0ﾋﾟﾝを'1'に設定、PWM出力開始
PINB |= 0x01
#5
// TIMER0計数器記録停止
$unlog TCNT0
#200
// PB0を'0'に設定、PWM出力停止
PINB &= 0xFE
#200
$stoplog
$break
```

これは以下の内容でmega328p_log_output.stimと呼ばれるファイルを生成します。

```
#20
TCNT0 = 0x09
TCNT0 = 0x0a
#1
TCNT0 = 0x0b
#1
TCNT0 = 0x0c
#1
TCNT0 = 0x0d
#1
TCNT0 = 0x0e
#1
TCNT0 = 0x0f
#1
PINB = 0x01
TCNT0 = 0x10
#1
TCNT0 = 0x11
#1
TCNT0 = 0x12
#1
TCNT0 = 0x13
#15
PIND = 0x40
#33
PIND = 0x00
#33
PIND = 0x40
#33
PIND = 0x00
#33
PIND = 0x40
#33
PIND = 0x00
#22
PINB = 0x00
#11
```

```
PIND = 0x40
#1
PIND = 0x00
```

1.3. プログラミング ダイアログでのシミュレータの使用

それが可能とは言え、プログラミング ダイアログでのシミュレータの使用は、シミュレータの揮発性のため、殆ど実用的な使用はありません。けれども、現実のどのハードウェア損傷の危険もなしに、または全てでどのハードウェアへの投資もすることなく、プログラミング ダイアログを調べる新規使用者に対して有用で有り得ます。フラッシュ メモリ内容、EEPROM内容、使用者識票、ヒューズ設定、施錠ビット設定を含めることができる製品ファイルを作成するのに使うこともできます。

シミュレータは以下の操作を支援します。

- デバイスIDの読み込み
- デバイス/メモリの消去
- メモリの書き込み、読み込み、確認
- ヒューズの書き込み、読み込み、確認
- 施錠ビットの書き込み、読み込み、確認

プログラミング ダイアログが開かれてデバイスが選ばれると、シミュレートされるデバイスが既定工場設定と空のメモリで開始します。シミュレータ内に書かれたどのデータも違うデバイスが選ばれる、またはプログラミング ダイアログが閉じられるまで保持されます。

1.4. シミュレータとハードウェア ツール間で鍵となる違い

Microchip Studioでシミュレータは基本的に他の何れのハードウェア ツールとも同様に扱われます。プログラミング ダイアログとデバッグ作業の両方に対して選ぶことができます。けれども、いくつかの鍵となる違いがあります。

- シミュレータは揮発性で、作業と作業の間でメモリを持たないことを意味します。1つの作業でシミュレータの(フラッシュ メモリ内容やヒューズのような)不揮発性メモリに書かれたどれもが、その作業終了後に忘れられます。作業終了時、シミュレートされたデバイスはまさに文字どおりに存在を止め、新しい作業が開始される時には新しくシミュレートされるデバイスが0から作成され、それの初期状態でその存在を開始します。特に、これはプログラミング ダイアログでヒューズを書いて、後でそれらのそのままのヒューズ設定でデバッグ作業を開始することができないことを意味します。新しい作業は常に既定ヒューズ設定で開始します。シミュレータと共にデバッグ作業で使うためのヒューズと他の任意設定はシミュレータ デバッグ作業中にだけ有効なシミュレータ特有特性頁を使って設定しなければなりません。作業と作業の間に全体または部分的なシミュレータの状態の保存を許す機能は将来に考慮されるかもしれません。
- シミュレータは選択可能な書き込みやデバッグのインターフェースを持ちません。これはシミュレートされるデバイスがソフトウェア模式によって実装され、この模式内の内部の全てのアクセスがソフトウェアAPI経由で行われ、物理的なインターフェースは伴われず、データを取得または書くのにクロック駆動される必要がないので、全てのアクセスが完全に非侵入型(**訳補**:制約要素がないことの意)だからです。
- 現在、同時に1つのシミュレータ実体だけを動かすことができます。また、シミュレータはハードウェア ツールの殆どが持つような通番が欠けています。
- シミュレータは実時間ではありません。これは(秒に対してシュミュレートしたCPU周期で測定した)シミュレーションの速度が実デバイスよりもかなり遅いことを意味ます。シミュレータは複数コアCPUで単一CPUコアだけを利用することができ、故により多くのコアへのPC更新はより速くしませんが、より大きなCPUキャッシュがより良い性能を示すことがあります。
- シミュレータはデバイスの完全な模式ではありません。デジタル論理回路が周期精度でシミュレートされる一方で、全てのアナログ周辺は現在欠けています。また、NVM(フラッシュ メモリとEEPROM)の模式は不完全です。不完全性の程度はデバイス間で変わり、「2. シミュレータでの既知の問題」をご覧ください。
- デバイス支援が完全ではありません。シミュレータはシミュレートする各デバイス/系統のソフトウェア模式に依存します。支援されるデバイスはシミュレータが選ばれる時にデバイス選択部で示されます。
- シミュレータは分離して走行し、シミュレートされるデバイスの周りがシミュレートされないことを意味します。シミュレータで現実の応用を走らせるために、シミュレートされるデバイスの入力に対して刺激が提供されなければなりません。「1.2.2. シミュレータ刺激」で示されるように、刺激は簡単な刺激ファイルによって提供されます。
- ATmega128シミュレータ模式について、ATmega103互換(M103C)ヒューズは既定でプログラム(0)されず、SUTはいくつかのデバイスで可能な最短値に設定されます。
- 殆どのtinyAVR®デバイスでは既定でSELFPRGENヒューズが非プログラム(1)にされ、動作でのSPMを防ぎます。SPMで動く時にこのヒューズを設定(0)してください。
- 多くのtinyAVRデバイスは交換ポート機能として外部リセットと、外部リセット ピンを禁止するためのRSTDISBLEヒューズを持ちます。(既定設定の)非プログラム(1)時、対応するポート ピンは(**訳補**:標準ポート機能として)意図されるように動かないでしょう。
- CKDIV8ヒューズは一般的にシミュレーション速度を増すために非プログラム(1)にされます。

1.5. AVR® Stuido 4とAVR23 Studioからの鍵となる違い

模式に基づくシミュレータは"シミュレータ2"としても知られるAVR Studio 4で使われた同じ技術を使います。更に32ビットUC2とArm®のデバイスの模式化も支援するように拡張されましたが、Arm模式は公には配布されません。

AVR32 Studioでの命令一式シミュレータは現在、Microchip Studioで利用できません。代わりに模式に基づくシミュレータが使われます。

2. シミュレータでの既知の問題

本章は現在のシミュレータの既知のバグと欠陥を一覧にします。括弧内で示される番号はバグ追跡システムでのバグ番号を参照します。

2.1. 全般的な問題

- シミュレータ設定ダイアログが未だ実装されていません(#13412)。
- AVR XMEGA® B1、UC3AとUC3Lを除くUC3系列、いくつかのtinyAVR®とmegaAVR®のデバイスは未だMicrochip Studioで支援されていません。
- AT90CAN*/ATmega*C*、AT90USB*/ATmega*U*、AT90PWM*、ATtiny87/167デバイスはシミュレータ模式によって決して支援されません。
- シミュレータによって外部メモリが支援されません(#7570、#9442)。
- 入出力ポートが出力として構成設定され、PORTxレジスタに書く時に、その変化が実チップでのように1周期遅れではなく、PINxレジスタで直ちに表示され得ます(#7188)。これはプログラムの実行ではなく、Microchip StudioのI/O表示部にだけ影響を及ぼします。
- AVR32 Studio命令一式シミュレータはMicrochip Studio 7.0で未だ実装されていません(#11557)。
- シミュレートされるデバイスが休止に置かれる場合、単一段階実行はプログラム計数器を進めません。実デバイスのように、この状態は何かが休止から起こすまで留まります。この動きの理由は休止時にCPUがコードを実行しないからです。代替はデバイス起き上がりまで反応しない単一段階実行を持ち、これは(おそらく決して起きない)それが起こるまでMicrochip Studio全体を無反応にするでしょう。
- シミュレートされるプログラムのコード内でのヒューズ設定はシミュレータの動作を妨げるかもしれません。特にOCDENヒューズとSUTヒューズはシミュレータを作業中止にさせるかもしれません。これはチップをリセットから脱するのにかかるクロック刻時数を計数するシミュレータでの制限時間によって起こされます。この制限時間が超過された場合、シミュレータはその作業を失敗します。

2.2. デバイスと系列特有の問題

2.2.1. tinyAVR®デバイス

- ATtiny40のRAMDRレジスタはMicrochip StudioのI/O表示部から書くことができません。対策: メモリ表示部経由で直接SRAMを書いてください。
- シミュレータでのATtiny10外部クロック選択はMicrochip Studioを無反応にさせます(#9349)。
- ATtiny25/45/85のPRRレジスタが動きません(#5584、PRRを持つ他のデバイスは動きます)。
- ATtiny25/45/85: 長すぎるウォッチドッグ制限時間(1MHzで64倍の長さ)。
- シミュレータ模式でシステム クロック前置分周器が含まれません。CLKPR書き込みはシステム クロックに影響を及ぼしません。(ATtiny4/5/9/10、ATtiny20、ATtiny40を除く全デバイス)
- CKDIV8ヒューズがプログラム(0)されている場合のデバッグ時にCLKPRが更新されません(#10515)。

2.2.2. megaAVR®と賢い電池デバイス

- ATmega16HVBのスタック ポインタは正しく初期化せず、応用によって初期化されなければなりません。これは実際のチップ共に問題で、シミュレータは単に現実を反映しているだけで、データシート障害情報章38.1.1.(改訂B)をご覧ください。
- ATmega169PA/165PA/329P/325P/3250P/3290P/649P/645P/6490P/6450Pのウォッチドッグ タイマが動きません(#9301)。
- シミュレータ模式でシステム クロック前置分周器が含まれません。CLKPR書き込みはシステム クロックに影響を及ぼしません。
- いくつかのATmegaデバイスで外部割り込み上昇端が両端で起動します。
- CKDIV8ヒューズがプログラム(0)されている場合のデバッグ時にCLKPRが更新されません(#10515)。

2.2.3. AVR® XMEGA®デバイス

- SPMを使うフラッシュ メモリと応用からのEEPROMの書き込み/消去はATxmegaデバイスで未だ実装されていません(#7611)。
- シミュレーションが他のデバイスに比べて遅いです。
- I/O表示部: WDT.CTRLレジスタのENABLEビット設定(1)によるウォッチドッグ許可が動きません。
- I/O表示部を通すCLK.CTRLレジスタ変更の試みがMicrochip Studioを無反応にします(XMEGA E5での割り当てられた読み込み専用)。
- XMEGA E5: 比較/捕獲(CCx)レジスタとTCxn.CTRLDがI/O表示部で読み込み専用。
- XMEGA E5: FAULTn.CTRLGレジスタがI/O表示部で読み込み専用。
- XMEGA E5: SPI.DATAレジスタがI/O表示部で読み込み専用。
- XMEGA E5: CRC.DATAINレジスタがI/O表示部で読み込み専用。

2.2.4. 32ビットAVR® UC3デバイス

- UC3A模式のI/O割り当てが未だ完了されていません。SMC、HMATRIX、FLASHC、MACB、SMC、DRAMC、INTC、PM、RTCのI/O単位部はMicrochip StudioのI/O表示部で常に0を示します。
- UC3デバイスのシミュレーションはUC3設計の複雑さと大きさのため、他のデバイスに比べて遅いです。

3. 改訂履歴

文書改訂	日付	注釈
A	2020年12月	初版文書公開

Microchipウェブ サイト

Microchipはwww.microchip.com/で当社のウェブ サイト経由でのオンライン支援を提供します。このウェブ 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** – データシートと障害情報、応用記述と試供プログラム、設計資源、使用者の手引きとハードウェア支援資料、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、オンライン検討グループ、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、セミナーとイベントの一覧、Microchip営業所の一覧、代理店と代表する工場

製品変更通知サービス

Microchipの製品変更通知サービスは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www.microchip.com/supportでのウェブ サイトを通して利用できます。

Microchipデバイス コード保護機能

Microchipデバイスでの以下のコード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デ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、その製品系統が安全であると考えます。
- Microchipデバイスのコード保護機能を破ろうとする試みに使われる不正でおそらく違法な方法があります。当社はこれらの方法がMicrochipのデ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とされると確信しています。これらのコード保護機能を破ろうとする試みは、おそらく、Microchipの知的財産権に違反することなく達成することはできません。
- Microchipはそれのコードの完全性について心配されている何れのお客様とも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のコードの安全を保証することはできません。コード保護は製品が”破ることができない”ことを当社が保証すると言うことを意味しません。コード保護は常に進化しています。Microchip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約束します。Microchipのコード保護機能を破る試みはデジタル 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

この刊行物含まれる情報はMicrochip製品を使って設計する唯一の目的にのために提供されます。デバイス応用などに関す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。

この情報はMicrochipによって「現状そのまま」で提供されます。Microchipは非侵害、商品性、特定目的に対する適合性の何れの黙示的保証やその条件、品質、性能に関する保証を含め、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。

如何なる場合においても、Microchipは情報またはその使用に関連するあらゆる種類の間接的、特別的、懲罰的、偶発的または結果的な損失、損害、費用または経費に対して責任を負わないものとします。法律で認められている最大限の範囲で、情報またはその使用に関連する全ての請求に対するMicrochipの全責任は、もしあれば、情報のためにMicrochipへ直接支払った料金を超えないものとします。生命維持や安全応用でのMicrochipデバ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www.microchip.com/qualityを訪ねてください。

日本語© *HERO* 2020.

本使用者の手引きはMicrochipのAVR®ｼﾐｭﾚｰﾀ使用者の手引き(DS50003042A-2020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: www.microchip.com/support ｳｪﾌﾞｱﾄﾞﾚｽ: www.microchip.com **ｱﾄﾗﾝﾀ** Duluth, GA Tel: 678-957-9614 Fax: 678-957-1455 **ｵｰｽﾁﾝ** TX Tel: 512-257-3370 **ﾎﾞｽﾄﾝ** Westborough, MA Tel: 774-760-0087 Fax: 774-760-0088 **ｼｶｺﾞ** Itasca, IL Tel: 630-285-0071 Fax: 630-285-0075 **ﾀﾞﾗｽ** Addison, TX Tel: 972-818-7423 Fax: 972-818-2924 **ﾃﾞﾄﾛｲﾄ** Novi, MI Tel: 248-848-4000 **ﾋｭｰｽﾄﾝ** TX Tel: 281-894-5983 **ｲﾝﾃﾞｱﾅﾎﾟﾘｽ** Noblesville, IN Tel: 317-773-8323 Fax: 317-773-5453 Tel: 317-536-2380 **ﾛｻﾝｾﾞﾙｽ** Mission Viejo, CA Tel: 949-462-9523 Fax: 949-462-9608 Tel: 951-273-7800 **ﾛｰﾘｰ** NC Tel: 919-844-7510 **ﾆｭｰﾖｰｸ** NY Tel: 631-435-6000 **ｻﾝﾎｾ** CA Tel: 408-735-9110 Tel: 408-436-4270 **ｶﾅﾀﾞ - ﾄﾛﾝﾄ** Tel: 905-695-1980 Fax: 905-695-2078	**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 - ヴェルス** 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: 45-4485-5910 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