付録．Ｂ　ＫＩＴ－ＡＲＭ９ｘｘ－ＴＰ（－Ｈ）　内部コマンド

本書は、ＫＩＴ－ＡＲＭ９ｘｘ－ＴＰ（－Ｈ）の内部コマンドについて記述しています。これらのコマンドは、デバッガの中でスルーコマンドとして使用できます。スルーコマンドの使用方法は各デバッガのマニュアルを参照ください。

ＰＡＲＴＮＥＲ／Ｗｉｎの場合

```
＞＆              ＜＜  スルーコマンドへの移行します。
＞＃ＥＮＶ        ＜＜  内部コマンドの入力です。
＞＆              ＜＜  スルーコマンドモードを終了します。
```

ＧＨＳ－Ｍｕｌｔｉの場合

ＲＴＥＳＥＲＶを接続後、ターゲット・ウインドウで直接入力できます。

コマンド一覧

以下のコマンドは一部の KIT でのみ有効なコマンドです。

これらのコマンドは、ご使用になりたい機能がデバッガ本体に有していない場合にのみ補助的にご使用ください。ご使用になるデバッガで同等の機能を有している場合にこれらのコマンドを発行した場合、デバッガとの間で競合をおこし、いずれかの動作が異常になる場合があります。[ETM]のコマンドが使用できるのは、CPU に ETM が実装されている場合に限ります。

コマンド書式

内部コマンドの基本書式を以下に示します。

コマンド名　パラメータ

＊パラメータ書式で［］は省略可能を示し、｜　は択一を意味します。

コマンド名はアルファベットの文字列で、パラメータとの間はスペースまたはタブで区切ります。パラメータはアルファベットの文字列または１６進数を指定し、各パラメータ間はスペースまたはタブで区切ります。（１６進数には演算子は使用できません。）

ａｃｃｅｎｖコマンド

[書式]
accenv [[!]jall] [[!]jsbp]

[パラメータ]
[!]jall: エミュレーション・メモリに割り付けしている空間に対するアクセス方法を指定します。jallはJTAG経由で行い、!jallはホストからエミュレーション・メモリへの直接アクセスを行います。(初期値)

> 備考：通常は、初期値でご使用ください。

[!]jsbp: エミュレーション・メモリに割り付けしている空間に対するソフトウェア・ブレークポイントを設定する際のアクセス方法を指定します。jsbpはJTAG経由で行い、!jsbpはホストからエミュレーション・メモリへの直接アクセスを行います。(初期値)
BIF-Dxx-xxが接続されている場合、jsbpでJTAG経由でソフトウェア・ブレークポイントの設定／解除を行う際のメモリ・ライト時は、エミュレーション・メモリを一時的に書込み可とし、非同期モードに変更してからメモリ・ライトを行います。したがって、romコマンドの書込み設定(wren)にも依存しませんので、!wrenのままで問題ありません。

> 備考：通常は、初期値でご使用ください。
> "accenv jsbp"に設定している場合、ブレーク中にユーザ・システム上のARMコア以外のデバイスがDMA等によりエミュレーション・メモリが割り付いている空間をアクセスした場合、そのアクセスは正常に行えない場合があります。

[機能]
エミュレーション・メモリに対するアクセス方法を指定するコマンドです。

> 備考：通常、エミュレーション・メモリに割付けた領域へのアクセスは、エミュレーション・メモリに対し、内部的にホストから直接アクセスを行い、エミュレーションしているCPUからアクセス(=JTAGアクセス)することはありません。このコマンドは、そのようなケースのアクセスを強制的にJTAGから行うようにするためのものです。
> 注意：jallを指定した状態で、大量のデータをエミュレーション・メモリにダウンロードすると、通常よりも時間がかかります。

[入力例]
accenv jall
　エミュレーション・メモリに対し、JTAG経由でアクセスします。
accenv jsbp
　エミュレーション・メモリ領域のソフトウェア・ブレークポイントの書換えをJTAG経由でアクセスします。
accenv !jall !jsbp
　通常のアクセスに戻します。

[注意事項]
このコマンドが使用できるのは、rte4win32 Ver5.11.B11、またはVer5.11.00以降のバージョンです。

ａｒｍｏｐｔコマンド

[書式]
armopt [[!]resetrecovery] [[!]resetautorun]

[パラメータ]
[!]resetrecovery: ユーザシステムからのリセットでICE制御レジスタがクリアされてしまうCPUにおいて、ユーザプログラム実行中にリセットが入った場合にその復元を行う時に指定します。!で復元は行いません。（初期値）
[!]resetautorun: resetrecoveryを指定した時に、自動的に実行を再開させる場合に指定します。!で自動的に実行は行いません。

[機能]
本コマンドはOMAP161x-TP用に用意されたものです。
OMAP_161xはCPUのリセット信号でICE制御レジスタがクリアされます。そのため、デバッグ中にユーザシステムからリセットを入れた場合、その後のICE制御が行えなくなりますので、それを回避するために用意されたコマンドです。

> 備考：これは実行中にリセットが入った時に機能するものです。ブレーク中にリセットが入れられた場合は、その後デバッガからリセットコマンドを発行してからデバッグを継続してください。尚、この機能を使用する為は、JTAG-IFのSRST-信号が双方向で機能することが必要です。

[入力例]
armopt resetrecovery resetautorun
resetrecovery機能をONし、リセット解除後、自動的に再実行をさせます。
armopt !esetrecovery !esetautorun
初期状態に戻します。

[注意事項]
このコマンドが使用できるのは、rte4win32 Ver5.11.B11、またはVer5.11.00以降のバージョンです。

ａｂｐ，ａｂｐ１，ａｂｐ２コマンド

[書式]
```
  abp[1|2] [ADDR [AMASK]] [data DATA [DMASK]] [user|privilege|both_mode]
           [[arm|thumb|both_iset] | [byte|hword|word|nosize]]
           [read|write|accs|exec] [ext|noext]
  abp{1|2} /del
```

[パラメータ]
```
  abp[1|2]:  abp1または、abp2の条件指定に先立ち入力します。
        ADDR [AMASK]: アドレス条件の指定
        ADDR:         アドレスを１６進数で指定します。
        AMASK:        アドレスのマスクデータを16進数で指定します。1のビットは、比較の対象に
                      なりません。
  data DATA [DMASK]: データ条件の指定
        DATA:         データを16進数で指定します。
        DMASK:        データのマスクデータを16進数で指定します。1のビットは、比較の対象にな
                      りません。
  user|privilege|both_mode:実行時の特権レベルを指定します。この指定はexecサイクルのみに有効
                      です。
        user:         ユーザモードの指定
        privilege:    特権モードの指定
        both_mode:    特権レベルを無視
  arm|thumb|both_iset:実行時の動作モードを指定します。この指定はexecサイクルのみに有効です。
        arm:          ARMモードを指定
        thumb:        thumモードの指定
        both_iset:    モードを無視
  read|write|accs|exec: サイクルの条件を指定します。
        read:         リードサイクルを指定します。
        write:        ライトサイクルを指定します。
        accs:         リードまたはライトサイクルを指定します。
        exec:         実行アドレスを指定します。データ条件は無視されます。
  byte|hword|word|nosize: アクセスサイズの指定します。この指定はexecサイクル以外で有効です。
        byte:         バイトアクセス(8-bit)を指定します。
        hword:        ハーフワードアクセス(16-bit)を指定します。
        word:         ワードアクセス(32-bit) を指定します。
        nosize:       無効を指定します。
  abp{1|2} /del: 条件の解除を行います。
        /del:         解除を指定します。
```

[機能]
```
   ２点ある、アクセス系のブレークポイントの設定または解除をします。
   実行アドレスの指定もできます。
   ソフトウェアブレークポイントと併用した場合、設定できるのは１点のみです。
```

[入力例]
```
   abp1 1000 exec both_mode both_iset
        1000h番地の実行にブレークを設定します。特権レベル、動作モードは共に無視します。
   abp2 1000 data 5555 0 read hword
        1000h番地からhwordで5555hをリードした時にブレークします。
   abp1 /del
        abp1の条件を解除します。
```

ｂｐｏｐｔ，ｂｐｏｐｔ２コマンド

[書式]
bpopt [or|seq|qnd]
bpopt2 [[!]trctrg]

[パラメータ]
or|and|seq: abp1とabp2の組合わせの条件を指定します。
or: abp1 又は、abp2のどちらかの発生でブレークします。
seq: abp1発生後、abp2が発生した時にブレークします。
and: abp1とabp2が同時に発生した時にブレークします。
abp1, abp2のアドレス、データ以外は同じ設定にし、
アドレスとデータのマスク条件で組合わせを指定します。
[!]trctrg: trctrg条件をブレーク条件として指定します。!で指定しません。

[機能]
abp1, abp2の組合わせ条件を指定します。

[入力例]
bpopt or
abp1 又は、abp2のどちらかの発生でブレークします。

ｅｎｖ，ｅｍｅｍｓｔａｔコマンド

[書式]
env [[!]auto] [jtag[xxx][.[yyy]]{M|K}] [[!]verify] [[!]int]
ememstat

[パラメータ]
[!]auto: 実行中にブレークポイントを設定した場合一時的にブレークしますが、その後の実行を自動的に行う場合に[Auto],行わない場合に[!auto]を指定します。
[jtag[xxx][.[yyy]]{M|K}]: JTAGクロックの周波数をMHz,またはKHzの単位で指定します。
指定は10KHzから125MHzの間の任意の値が可能ですが､設定されるのは指定値以下の以下の値に丸められます。実際の設定値は表示で確認できます。
RTE-2000-TP :[25MHz,12.5MHz,5MHz,2MHz,1MHz,500KHz,250KHz,100KHz]
RTE-2000H-TP:[125MHz,100MHz,80MHz,60MHz,50MHz,40MHz,30MHz,25MHz,12.5MHz,5MHz,2MHz,1MHz,500KHz,250KHz,100KHz,50KHz,25KHz,10KHz]

> 注意：通常は25MHzまたは、12.5MHzでご使用ください。1MHzより低い周波数を指定した場合、デバッガの動作が著しく遅くなったり、異常になる場合があります。
> 初期値は25MHzを上限とした動作する最高周波数に自動的に設定します。
> 初期値以上の値に設定する場合はCPUの許容範囲内で設定してください。
> CPUのスペック以上の周波数を設定した場合の動作は保証できません

[!]verify: メモリへの書込み時にリードアウトしてベリファイするかどうか指定します。!はベリファイしないを意味します。

> 備考：ROMをエミュレーションしている領域に対しも、CPUからアクセス(jread相当)しますので、ダウンロード時のテストにも有効です。但し、処理速度が遅くなります。

[!]int: int端子のマスク指定を指定します。!はマスクしないを意味します。

[機能]
envコマンドは、エミュレーション環境の設定と状態を表示します。
設定は変更が必要なパラメータだけを入力ください。入力の順序は任意です。
但し、同じパラメータを２回入力した場合は、後から入力した値が有効です。
ememstatコマンドは、E.MEM基板の実装状態を表示するコマンドです。
以下に表示例を示します。

```
env
Probe:
Probe:
 Unit      : RTE-2000(H)-TP
 Rom Probe : (use ememstat command)
 Emem Size : (use ememstat command)
CPU Settings:
 Auto Run     = ON (auto)
 JTAGCLOCK    = 25MHz (jtag25m)
 Verify       = verify off (!verify)
Signals Mask:
 INT          = NO MASK (!int)

ememstat
 Board_num  EMEM_Size  ROM_Probe
 ===================================
    ROM1     8Mbyte    Extend Type 2K
```

[入力例]
env verify int
Verify機能をONにし、intをマスクます。
env !int
intのマスクを解除します。
env jtag40m
JTAGクロックを40MHzに設定します。

ｈｅｌｐコマンド

[書式]
help [command]

[パラメータ]
command: コマンド名を指定します。
コマンド名を省略した場合、コマンドの一覧が表示されます。

[機能]
各コマンドのヘルプメッセージを表示します。

[使用例]
help map
mapコマンドのヘルプを表示します。

ｉｎｂ，ｉｎｈ，ｉｎｗコマンド

[書式]
inb [ADDR]
inh [ADDR]
inw [ADDR]

[パラメータ]
ADDR: 入力ポートのアドレスを16進数で指定します。

[機能]
inb,inh,inwは、アクセスサイズを区別して、リードを行ないます。
inbはバイト、inhはハーフ・ワード、inwはワード単位でアクセスします。

[使用例]
inb 1000
1000Hからバイト(8-bit)でリードします。
inh 1000
1000Hからハーフワード(16-bit)でリードします。
inw 1000
1000Hからワード(32-bit)でリードします。

ｉｎｉｔコマンド

[書式]
　init

[パラメータ]
　なし

[機能]
　ICEの環境を起動時の状態に初期化します。
　以下を除き、全ての環境設定値は初期化されます。
　・メモリキャッシュの除外エリア

ｊｒｅａｄコマンド

[書式]
jread [ADDR [LENGTH]]

[パラメータ]
ADDR:　　アドレスを１６進数で指定します。
LENGTH:　読み出すバイト数を１６進数で指定します。(max 100h)

[機能]
ROMコマンドで割り付けたROMエミュレーション領域をJTAG(CPU)から読み出すためのコマンドです。（通常のコマンドでは、ROMエミュレーション領域へのアクセスは内部のメモリに対し直接行っています。）

[使用例]
jread 100000 100
　　100000hから100hバイトをJTAG経由で読み出します。

ｎｃコマンド

[書式]
nc [[ADDR [LENGTH]]

[パラメータ]
ADDR: メモリキャッシュの除外エリアの開始アドレスを指定します。
LENGTH: メモリキャッシュの除外エリアのバイト数を指定します。
デフォルト値３２バイト、最少値３２バイト

[機能]
メモリ参照の高速化を図るため、ファームウェア内に８ブロック＊３２バイトのメモリリードキャッシュを持っています。同一アドレスのメモリ参照などは実際にはメモリをリードしません。Ｉ／Ｏを割り付けている空間では、このキャッシュ機能は実際の動作と矛盾しますので、このコマンドで除外エリアとして指定してください。メモリキャッシュの除外エリアは最大８ブロック指定でき、最少のブロックサイズは３２バイトです。

[使用例]
nc 10000 100
10000h番地から100バイトの領域をメモリキャッシュの除外エリアに指定します。

```
>nc 100000 100
 No Memory Cache Area
 No. Address  Length
 1  00100000   00000100
```

ｎｃｄコマンド

[書式]
ncd ブロック番号

[パラメータ]
ブロック番号: 削除するメモリキャッシュの除外エリアのブロック番号を指定します。

[機能]
メモリキャッシュの除外エリアを削除します。削除は各メモリキャッシュの除外エリアのブロック番号を指定します。

[使用例]
ncd 1
ブロック番号１をメモリキャッシュの除外エリアから削除します。

```
>nc 100000 100
 No Memory Cache Area
 No. Address  Length
 1  00100000   00000100

>ncd 1
 No Memory Cache Area
 No. Address  Length
```

ｎｓｂｐコマンド

[書式]
nsbp [[ADDR [LENGTH]]

[パラメータ]
ADDR: ソフトウェアブレーク禁止領域の開始アドレスを指定します。
LENGTH: ソフトウェアブレーク禁止領域のバイト数を指定します。
指定領域の最小単位はハーフワードバウンダリです。
また、指定できる領域の数は最大4ヶ所です。

[機能]
ソフトウェアブレークを禁止したい領域を指定します。
ブレークポイントを指定した場合、デバッガは暗黙的に対象アドレスに対し、メモリテスト（ライトアクセス）を行います。
一部のフラッシュROM等、ライトアクセスを行うことでメモリの状態が変り、正しいデータの読み出しが行えなくなる場合等に、ライトサイクルを禁止する目的で指定してください。
通常は、指定する必要はありません。

[使用例]
nsbp 10000 20000
10000h番地から20000バイトの領域をソフトウェアブレーク禁止領域に指定します。

```
>nsbp 100000 20000
Num Address  Length
01  00100000 00020000
```

ｎｓｂｐｄコマンド

[書式]
nsbpd [ブロック番号|/all]

[パラメータ]
ブロック番号: 削除するソフトウェアブレーク禁止領域のブロック番号を指定します。
/all: 全てのソフトウェアブレーク禁止領域を削除します。

[機能]
nsbpで指定したソフトウェアブレーク禁止領域を削除します。

[使用例]
nsbpd 1
ブロック番号１をソフトウェアブレーク禁止領域から削除します。

```
nsbp
Num Address  Length
01  00100000 00200000
02  00400000 00010000

>nsbpd 1
Num Address  Length
01  00400000 00010000
```

ｎｒｏｍコマンド

[書式]
nrom [[ADDR [LENGTH]]

[パラメータ]
ADDR: 強制ユーザ領域の開始アドレスを指定します。
LENGTH: 強制ユーザ領域のバイト数を指定します。
指定領域の最小単位は、以下の通りです。
エミュレーションしているROMのサイズに応じます。
8/16-bit：128k-byte単位
32-bit ：256k-byte単位
64-bit ：512k-byte単位
指定できる領域の数は最大4ヶ所です。

[機能]
ROMコマンドで指定したROMエミュレーション領域内の一部がユーザシステム上の資源にマップされていた場合にその領域を指定します。通常は指定する必要はありません。
指定領域に対する動作は以下の通りです。
デバッガからのアクセスは強制的にユーザシステムに対し行われるようになります。
実行中この領域へのアクセスサイクルでEMEMEN-信号はインアクティブ(Highレベル)になります。

[使用例]
nrom 0 20000
0h番地から20000バイトを強制ユーザ領域に指定します。

```
>nrom 0 20000
No. Address    Length
 1  00000000   00020000

>nrom 100000 40000
No. Address    Length
 1  00000000   00020000
 2  00100000   00040000
```

ｎｒｏｍｄコマンド

[書式]
nromd [ブロック番号|/all]

[パラメータ]
ブロック番号: 削除する強制ユーザ領域のブロック番号を指定します。
/all: 全ての強制ユーザ領域のブロックを削除します。

[機能]
nromで指定した強制ユーザ領域を削除します。

[使用例]
ncd 1
ブロック番号１を強制ユーザ領域から削除します。

```
>nrom 100000 40000
No. Address    Length
 1  00000000   00020000
 2  00100000   00040000

>nromd 1
No. Address    Length
 1  00100000   00040000
```

ｏｕｔｂ，ｏｕｔｈ，ｏｕｔｗコマンド

[書式]
outb [[ADDR] DATA]
outh [[ADDR] DATA]
outw [[ADDR] DATA]

[パラメータ]
ADDR:　　出力ポートのアドレスを16進数で指定します。
DATA:　　出力するデータを16進数で指定します。

[機能]
outb,outh,outwは、アクセスサイズを区別して、ライトを行ないます。
outbはバイト、outhはハーフ・ワード、outwはワード単位でアクセスします。

[使用例]
outb 1000 12
　　1000Hへバイトデータ：12hをlライトします。
outh 1000 1234
　　1000Hへハーフワードデータ：1234hをライトします。
outh 1000 12345678
　　1000Hへワードデータ：12345678hをライトします。

ｒｅｓｅｔコマンド

[書式]
reset

[パラメータ]
なし

[機能]
エミュレーションＣＰＵをリセットします。

ｒｏｍ１..ｒｏｍ４コマンド

[書式]

rom1 [ADDRESS [LENGTH]] [512k|1m|2m|4m|8m|16m|32m|64m|128m|256m] [rom8|rom16] [bus8|bus16|bus32|bus64] [[!]wren]

rom2 [ADDRESS [LENGTH]] [512k|1m|2m|4m|8m|16m|32m|64m|128m|256m] [rom8|rom16] [bus8|bus16] [[!]wren]

rom3 [ADDRESS [LENGTH]] [512k|1m|2m|4m|8m|16m|32m|64m|128m|256m] [rom8|rom16] [bus8|bus16|bus32] [[!]wren]

rom4 [ADDRESS [LENGTH]] [512k|1m|2m|4m|8m|16m|32m|64m|128m|256m] [rom8|rom16] [bus8|bus16] [[!]wren]

rom1: スロット#3に実装されたEMEM基板を含むモジュールに対する設定コマンドです。
rom2: スロット#4に実装されたEMEM基板を含むモジュールに対する設定コマンドです。
rom3: スロット#5に実装されたEMEM基板を含むモジュールに対する設定コマンドです。
rom4: スロット#6に実装されたEMEM基板を含むモジュールに対する設定コマンドです。

[パラメータ]

ADDR [LENGTH]: エミュレーションする領域を指定します。

ADDR: 開始アドレスを指定します。
エミュレートするROMの最下位のアドレス（ROMのバウンダリ）に合致していない場合、指定アドレス以下のアドレス領域は非エミュレーション領域になります。

LENGTH: エミュレートするROMのバイト数を指定します。

> 備考：ADDR, LENGTHで指定できる領域の最小単位は、エミュレーションしているROMのサイズに応じ、以下の通りです。
> ・8/16-bit：128k-byte単位
> ・32-bit　：256k-byte単位
> ・64-bit　：512k-byte単位

512k|1m|2m|4m|8m|16m|32m|64m|128m|256m:
１本のROMプローブでエミュレートするROMのBit容量を指定します。512K-bitから256M-bit(32M-Byte)までの値が指定できます。
例えば、27C1024の場合は、1mを指定します。

rom8|rom16: エミュレートするROMのデータビット数を指定します。
8bitと16bitが指定できます。DIP32のアダプタを使用する場合はrom8、DIP-40/42のアダプタ、及び16bit-標準ROMケーブルをそのまま使用する場合は、rom16を指定します。

bus8|bus16|bus32|bus64:
エミュレートするシステムの中でのROMのバスサイズを指定します。
8bit,16bit,32bit,64bitが指定できます。
>> [64-bit]は将来のためのパラメータです。（本KITでは使用しません）

[[!]wren]: Write Enable:エミュレーションメモリをRAMとして使用する場合の設定です。
wrenで書込み許可、!wrenで書込み禁止です。初期値は!wrenです。

[機能]

ROMエミュレーション環境の設定を行います。設定はADDRとLENGTHをペアで入力する以外は必要なパラメータだけ入力できます。入力の順序は任意です。但し、同じパラメータを２回入力した場合は、後から入力した値が有効です。初期値は、LENGTH = 0（使用しない）になっています。

［入力例］

>rom1 100000 300000 32m rom16 bus16 !wren

対象EMEM基板 ｽﾛｯﾄ位置	アドレス範囲	バス幅	ROM		ﾗｲﾄｲﾈｰﾌﾞﾙ
			バス幅	Bit数	
#3	100000 – 3fffff	16-bit	16-bit	32M-Bit	禁止

>rom2 140000 40000 2m rom16 bus16 wren

対象EMEM基板 ｽﾛｯﾄ位置	アドレス範囲	バス幅	ROM		ﾗｲﾄｲﾈｰﾌﾞﾙ
			バス幅	Bit数	
#4	140000 – 17ffff	16-bit	16-bit	2M-Bit	許可

>rom1 0 80000 2m rom rom16 bus32 !wren

対象EMEM基板 ｽﾛｯﾄ位置	アドレス範囲	バス幅	ROM		ﾗｲﾄｲﾈｰﾌﾞﾙ
			バス幅	Bit数	
#3+#4	000000 – 07ffff	32-bit	16-bit	2M-Bit	禁止

この時、rom2コマンドは発行しないでください。

＜備考＞

ｒｏｍコマンドで指定した領域における注意事項

ｒｏｍ１..ｒｏｍ４コマンドで指定した範囲へのデバッガからのアクセスは、ツール内部のエミュレーションメモリに対し直接アクセスしています。その結果、プロセッサから正しくROMにアクセスできない状態においても表示は正しく行われますので、デバッグ初期の段階ではjreadコマンド(CPUのバス経由で読み出すコマンド）を使用して読み出し確認するか、envコマンドでverifyをONにして書き込み（ダウンロード）を行うことをお勧めします。

ｒｏｍコマンドとEMEM基板の関係

romコマンド	バス幅	対象EMEM基板の ｽﾛｯﾄ位置	使用できないromコマンド
rom1	8-bit	#3	
	16-bit	#3	
	32-bit	#3+#4	rom2
	64-bit	#3+#4+#5+#6	rom2, rom3, rom4
rom2	8-bit	#4	
	16-bit	#4	
rom3	8-bit	#5	
	16-bit	#5	
	32-bit	#5+#6	rom4
rom4	8-bit	#6	
	16-bit	#6	

ｓｙｍｆｉｌｅ，ｓｙｍコマンド

[書式]
symfile FILENAME
sym [NAME]

[パラメータ]
symfile:　ファイル名を指定します。
sym:　　シンボルの先頭文字列を指定します。

[機能]
symfile コマンドは、FILENAMEで指定したelfファイルからシンボルを読み込みます。
対象となるのはグローバルシンボルだけです。
symコマンドは、読み込んだシンボルの表示（最大３０個）をします。

[使用例]
symfile c:¥test¥dry¥dry.elf
　　c:¥test¥dryのディレクトリからelfファイル:dry.elfのシンボルを読み込みます。
sym m
　　mから始まるシンボルを最大30個表示します。

ｖｅｃｔｏｒｂｐコマンド

[書式]
vectorbp [[!]reset] [[!]undef] [[!]soft_int] [[!]prefetch] [[!]data]
[[!]irq] [[!]fiq]

[パラメータ]

[!]reset:	リセット例外の発生でブレークします。!で解除します。
[!]undef]:	未定義命令例外の発生でブレークします。!で解除します。
[!]soft_int:	ソフトウェア割込み例外の発生でブレークします。!で解除します。
[!]prefetch:	プリフェッチアボート例外の発生でブレークします。!で解除します。
[!]data:	データアボート例外の発生でブレークします。!で解除します。
[!]irq:	IRQ割込み例外の発生でブレークします。!で解除します。
[!]fiq]:	FIQ割込み例外の発生でブレークします。!で解除します。

[機能]
vectorbpコマンドは例外発生時のベクタへの分岐でブレークする条件を設定するコマンドです。

[入力例]
vectorbp undef
未定義命令例外の発生でブレークします。
vectorbp !undef
未定義命令例外の発生でのブレークを解除します。

ｅｖｔコマンド

[書式]
evt [EVENT_NAME [[!]always|only|and|or] [rsca A_RESOURCE] [rscb B_RESOURCE]

[パラメータ]
EVENT_NAME: イベント名を指定します。
trctrg: トレーストリガ条件
trcenable: トレースイネーブル条件
viewdata: ビューデータ条件
count{1|2|3|4}: カウンタのデクリメント条件
reload{1|2|3|4}: カウンタのリロード条件
seq{1to2|1to3|2to1|2to3|3to1|3to2}: シーケンサ条件
extout{1|2|3|4}: 外部出力(External Output)条件
[[!]always|only|and|or]: リソースの組合わせを指定します。
always: イベントを常に有効とします。!で条件に従います。
only: A_RESOURCE条件だけでイベント成立とします。
and: A_RESOURCEとB_RESOURCEのAND条件をイベント成立とします。
or: A_RESOURCEとB_RESOURCEのOR条件をイベント成立とします。
rsca: A_RESOURCEの指定の直前に必要です。
rscb: B_RESOURCEの指定の直前に必要です。
A_RESOURCE,B_RESOURCE: リソースの指定です。B_RESOURCEだけに!条件を付けることはできません。
[!]rscsngl{1..16}: シングルコンパレータのリソースを指定します。!で否定条件です。
[!]rscrng{1..8}: レンジコンパレータのリソースを指定します。!で否定条件です。
[!]rsccnt{1|2|3|4}: カウンタを指定します。カウンタ値が条件成立。!で否定条件です。
[!]rscseq{1|2|3}: シーケンサのステートを指定します。!で否定条件です。
[!]rscmmd{1..16}: メモリマップデコーダを指定します。!で否定条件です。
[!]rscexti{1|2|3|4}: 外部入力(External Inputを指定します。!で否定条件です。

[機能]
トレースをコントロールするためにイベントに対し、そのリソースを指定するコマンドです。

[使用例]
evt trctrg only rsca rscsngl1
トレーストリガとしてシングルコンパレータのリソース１を指定します。

[備考]
設定できる各リソースの最大数は、rscstatコマンドで確認してください。これはCPUに組み込まれたETMの仕様に依存します。
メモリマップデコーダは、CPUの仕様に依存します。
ＥＴＭとトレースの関係は、付録. Ａ トレース機能の詳細を参照ください。

ｒｓｃｃｎｔ［1.. 4］コマンド

[書式]
rsccnt{1|2|3|4} [COUNT]

[パラメータ]
COUNT: カウンタの初期値を指定します。(0x1 - 0xFFFF)

[機能]
カウンタの初期値を設定するためのコマンドです。

[使用例]
rsccnt1 1000
カウンタ#1に0x1000を設定します。

[備考]
設定できるカウンタの最大数は、rscstatコマンドで確認してください。これはCPUに組み込まれたＥＴＭの仕様に依存します。
ＥＴＭとトレースの関係は、付録. Ａ トレース機能の詳細を参照ください。

ｒｓｃｓｎｇｌ［1.. １６］コマンド

[書式]
rscsngl{1..16} [ADDR] [amask{0|1|3}] [fetch|exec|exec_e|exec_n|accs|read|write]
[{nodata | {deq|dneq} DATA [DATA_MASK]}]

[パラメータ]
[ADDR] [amask{0|1|3}]:アドレス条件の指定
ADDR: アドレスを１６進数で指定します。
[amask{0|1|3}]: 下位2-bitのアドレスのマスクを指定します。
0-マスクなし、1-A0をマスク、3-A[1..0]をマスク
fetch|exec|exec_e|exec_n|accs|read|write: サイクル条件の指定
fetch: 命令フェッチを指定します。
exec: 命令の実行を指定します。
exec_e: 条件が成立した実行を指定します。
exec_n: 条件が成立しなった実行を指定します。
accs: readとwriteの両方のサイクルを指定します。
read: readサイクルを指定します。
write: writeサイクルを指定します。
nodata | {deq|dneq} DATA [DATA_MASK]}]: データ条件の指定
nodata: データ条件は無視します。
deq: データの一致条件を指定します。
dneq: データの不一致条件を指定します。
DATA: データを32-bit、6進数で指定します。
DMASK: データのマスクデータを32-bit、16進数で指定します。1のビットは、比較の対象になりません。

[機能]
シングルコンパレータとレンジコンパレータの各リソースの条件を指定するためのコマンドです。
レンジコンパレータは以下のシングルコンパレータの組合わせです。
rscrng1 – rscsngl1, rscsingl2
rscrng2 – rscsngl3, rscsingl4
rscrng3 – rscsngl5, rscsingl6
rscrng4 – rscsngl7, rscsingl8

[使用例]
rscsngl1 1000 amask3 write deq 12345678 0
シングルコンパレータ#1に対し、以下を設定します。
0x10000番地に対する0x12345678の書込み
rscsngl2 100 amask0 exec
シングルコンパレータ#2に対し、以下を設定します。
0x100番地の命令実行

[備考]
データ条件が設定できるのは、奇数のコンパレータだけです。また、設定できる最大数は、rscstatコマンドで確認してください。これはCPUに組み込まれたETMの仕様に依存します。
ＥＴＭとトレースの関係は、付録. Ａ トレース機能の詳細を参照ください。

ｒｓｃｓｔａｔコマンド

[書式]
rscstat

[パラメータ]
なし

[機能]
CPUに実装されているETMのリソース数を表示します。

[使用例]
以下は、一例です。

```
>rscstat
Number of Resources:
Address Comparator   (rscsngl) : 8
Data Comparator                : 2
Counter              (rsccnt)  : 2
Sequencer            (rscseq)  : 1
ContextID Comparator (rsccid)  : 0
Memory Map Decoder   (rscmmd)  : 8
External Input       (rscexti) : 4
External Output      (rscexto) : 1
```

[備考]
ＥＴＭとトレースの関係は、付録. Ａ トレース機能の詳細を参照ください。

ｔｒｃｅｎａｂｌｅコマンド

[書式]
```
trcenable [include|exclude] [rscsngl_{[1]..[g]}] [rscrng_{[1]..[4]}]
          [rscmmd_{[1]..[g]}]
          [[!]always] [start rscsngl_{[1]..[g]}] [stop rscsngl_{[1]..[g]}]
```

[パラメータ]
- include|exclude: 一致条件のモードの指定
  - include: リソース条件が一致した状態をトレースイネーブルとして指定。
  - exclude: リソース条件が一致しなかった状態をトレースイネーブルとして指定。
- rscsngl_{[1]..[g]}: 一致条件としてシングルコンパレータのリソースを指定します。
- rscrng_{[1]..[4]}: 一致条件としてレンジコンパレータのリソースを指定します。
- rscmmd_{[1]..[g]}: 一致条件としてメモリマップデコーダを指定します。
- [!]always] [start rscsngl_{[1]..[g]}] [stop rscsngl_{[1]..[g]}: 区間条件の指定
- [!]always: 区間トレース条件を常に有効として指定します。!で区間条件のリソースが有効になります。
- start rscsngl_{[1]..[g]: 区間トレース条件の開始条件として、シングルコンパレータのリソースを指定します。
- stop rscsngl_{[1]..[g]}: 区間トレース条件の停止条件として、シングルコンパレータのリソースを指定します。

[機能]

トレースを取り込む条件を指定するためのコマンドです。イベント条件(evtコマンド)とtrcenablrコマンドの一致条件と区間条件の全てが成立している時に行います。したがって、片方の条件だけを指定する場合は、他方は常に成立の状態にしておくことが必要です。（一致条件では、excludeを、区間条件では、alwaysを使います。）

[使用例]
```
evt trcenable always
trcenable include rscsngl_12 rscrng_2 always
```
シングルコンパレータ#1と#2、レンジコンパレータ#2に一致したサイクルをトレースイネーブルとして指定します。区間はalwaysとすることで無条件に成立状態にします。
```
trcenable exclude rscsngl_ rscrng_ start rscsngl_1 stop rscsngl_2
```
シングルコンパレータ#1を開始条件、#2を停止条件を区間トレース条件としてトレースイネーブルに指定します。イベントの一致条件は指定リソースを削除し、excludeを指定することで無条件に成立状態にします。

[備考]

リソースに設定できる最大数は、rscstatコマンドで確認してください。これはCPUに組み込まれたＥＴＭの仕様に依存します。

メモリマップデコーダは、CPUの仕様に依存します。

ＥＴＭとトレースの関係は、付録. Ａ トレース機能の詳細を参照ください。

ｖｉｅｗｄａｔａコマンド

[書式]
viewdata [!include | include [rscsngl_{[1]..[g]}] [rscrng_{[1]..[4]}]
[rscmmd_{[1]..[g]}]]
[exclude [rscsngl_{[1]..[g]}] [rscrng_{[1]..[4]}] [rscmmd_{[1]..[g]}]]

[パラメータ]
!include | include [rscsngl_{[1]..[g]}] [rscrng_{[1]..[4]}: 一致条件の指定
!include: 一致条件としてリソース条件を無視します。
include: 一致条件としてリソース条件を指定する時に必要です。
rscsngl_{[1]..[g]}: シングルコンパレータのリソースを指定します。
rscrng_{[1]..[4]}: レンジコンパレータのリソースを指定します。
rscmmd_{[1]..[g]}: メモリマップデコーダを指定します。
[exclude [rscsngl_{[1]..[g]}] [rscrng_{[1]..[4]}] [rscmmd_{[1]..[g]}: 不一致条件の指定
exclude: 不一致条件のリソース指定時に必要です。
rscsngl_{[1]..[g]}: シングルコンパレータのリソースを指定します。
rscrng_{[1]..[4]}: レンジコンパレータのリソースを指定します。
rscmmd_{[1]..[g]}: メモリマップデコーダを指定します。

[機能]
アクセスサイクルのデータをトレースに取り込む条件を指定するためのコマンドです。一致条件と不一致条件がそれぞれ指定でき、両方の条件が成立した状態で取り込みます。

[使用例]
viewdata include rscsngl_12 always
シングルコンパレータ#1と#2に一致したサイクルをviewdataとして指定します。

[備考]
リソースに設定できる最大数は、rscstatコマンドで確認してください。これはCPUに組み込まれたＥＴＭの仕様に依存します。
メモリマップデコーダは、CPUの仕様に依存します。
ＥＴＭとトレースの関係は、付録. Ａ トレース機能の詳細を参照ください。

ｆｉｆｏｆｕｌｌコマンド

[書式]
fifofull [include|exclude] [rscrng_{[1]..[4]}] [rscmmd_{[1]..[g]}]]

[パラメータ]
include|exclude: リソースの一致／不一致の条件指定
include: リソース条件が一致した状態をfifofullとして指定。
exclude: リソース条件が一致していない状態をfifofullとして指定。
rscrng_{[1]..[4]}] [rscmmd_{[1]..[g]}:
rscrng_{[1]..[4]}: レンジコンパレータのリソースを指定します。
rscmmd_{[1]..[g]}: メモリマップデコーダを指定します。

[機能]
完全モード(tronコマンド!realパラメータ)でトレースを取り込む条件を指定します。

[使用例]
fifofull include rscrng_2
レンジコンパレータ#2に一致したサイクルをfifofullとして指定します。

[備考]
リソースに設定できる最大数は、rscstatコマンドで確認してください。これはCPUに組み込まれた
ＥＴＭの仕様に依存します。
メモリマップデコーダは、CPUの仕様に依存します。
ＥＴＭとトレースの関係は、付録. Ａ トレース機能の詳細を参照ください。

ｔｅｎｖコマンド

[書式]
tenv [td_none|td_addr|td_data|td_both] [tclkdiv{1|2}] [fifoNN] [mmdNN]

[パラメータ]
td_none|td_addr|td_data|td_both: データトレースの種類を指定
td_none: データトレースなしを指定します。
td_addr: アドレスのみを指定します。
td_data: データのみを指定します。
td_both: アドレスとデータの両方を指定します。
tclkdiv{1|2}: 初期値でご使用ください。
fifoNN: 初期値でご使用ください。
mmdNN: メモリマップデコーダの制御レジスタの設定です。(0 – 255)

[機能]
トレースの環境を表示と設定を行うコマンドです。

[使用例]
tenv
設定状態の表示です。以下は、起動直後の初期値の表示例です。

```
>tenv
Trace Env Settings :
 Data Trace                = Address and Data (td_both)
 TRACECLK Div.             = 1/1 (tclkdiv1)
 FIFOFULL Min. Remain Level = 1 (fifo1)
 MMD Control Regiser Value  = 0 (mmd0)
 TRACEPKT Bit Width         = 8 bit (tdwidth8)
 Trace Out Context ID Bits  = Disable (cid0)
 tenv td_none
 データトレースを行いません。
```

[備考]
メモリマップデコーダの制御レジスタは、CPUの仕様に依存します。
ＥＴＭとトレースの関係は、付録. Ａ トレース機能の詳細を参照ください。

ｔｍｏｄｅコマンド

[書式]
tmode

[パラメータ]
なし

[機能]
トレースの設定状態を確認するためのコマンドです。

[使用例]
以下は、起動直後の初期値の表示例です。

```
>tmode
Trace Settings:
  Delay Count        = 01ffff
  Delay Mode         = Disable (!delay)
  Trace Mode         = Real Time (real)
  Ext Trigger        = Disable (noext)
Trace Env Settings:
 Data Trace                 = Address and Data (td_both)
 TRACECLK Div.              = 1/1 (tclkdiv1)
 FIFOFULL Min. Remain Level = 1 (fifo1)
 MMD Control Regiser Value  = 0 (mmd0)
 TRACEPKT Bit Width         = 8 bit (tdwidth8)
 Trace Out Context ID Bits  = Disable (cid0)
Event Settings:
 evt trctrg    !always
 evt trcenable always
 evt viewdata  always
Trace Enable Settings:
 Include/Exclude Regions:
   Mode                : Exclude (exclude)
   Single Comparator   : nothing
                         (rscsngl_)
   Range Comparator    : nothing
                         (rscrng_)
   Memory Map Decoder  : nothing
                         (rscmmd_)
 Trace Start/Stop Resource:
   Mode                : Always Start (always)
   Start Single Comp.  : nothing
                         (start rscsngl_)
   Stop Single Comp.   : nothing
                         (stop rscsngl_)
View Data Settings:
 Include Regions:
   Mode                : Include region Ignore (!include)
   Single Comparator   : nothing
                         (rscsngl_)
   Range Comparator    : nothing
                         (rscrng_)
   Memory Map Decoder  : nothing
                         (rscmmd_)
 Exclude Regions:
```

```
    Single Comparator    : nothing
                           (rscsngl_)
    Range Comparator     : nothing
                           (rscrng_)
    Memory Map Decoder   : nothing
                           (rscmmd_)
  FIFO Full Settings:
   Mode                : Include (include)
   Range Comparator    : nothing
                         (rscrng_)
   Memory Map Decoder  : nothing
                         (rscmmd_)
  Option Break 2 Settings :
   Trace Trigger Break : Disable (!trctrg)
```

[備考]

ＥＴＭとトレースの関係は、付録. Ａ トレース機能の詳細を参照ください。

ｔｒｏｎコマンド

[書式]
tron [DELAY] [[!]delay] [[!]real] [noext|posi|nega]

[パラメータ]
DELAY = 0..xxxx: ディレイカウンタ
トリガ成立後にメモリに取り込むフレーム数を１６進数で指定します。
[!]delay: 強制ディレイモードを指定します。!で通常のモードの指定に戻ります。強制ディレイモードでは、トレース開始後、ディレイカウンタ数分のトレースをした時点で強制的にトレースを終了するモードです。このモード中は、トリガイベントは無視されます。
[!]real: トレース中の実行モードを指定します。realでリアルタイム実行モードです。リアルタイム実行モードでは、トレース情報がオーバフローする場合があります。!で完全モード（非リアルタイム実行モード）になります。このモードでは、トレースのオーバフローは発生しませんが、実行速度が低下します
noext|nega|posi: トリガとして外部入力端子(EXI0)を指定します。
noext: EXI0をトリガとして使用しません。
posi: EXI0の立ち上がりエッジをトリガとして指定します。
nega: EXI0の立ち下がエッジをトリガとして指定します。

[機能]
トレースの設定とトレースバッファをクリアし、トレースの取り込みを開始します。

[使用例]
tron
初期値でtronした場合、トレースは強制的に開始し、トレースを強制的に終了するまでトレースします。ブレーク後trace表示させた場合、ブレーク直前の実行までの実行状態が表示できます。
tron delay 3fffd
初期値に対し強制ディレイモード(delay=on)でトレースを開始します。実行開始直後より、ディレイカウンタ値:0x3fffd分の取り込み後、トレースは自動的に終了します。強制ディレイモードでは、トリガは無視されます。
rscsngl1 1000 amask0 exec
evt trctrg onlya rsca rscsngl1
tron !delay 1ffff
0x1000番地の実行をトリガポイントとしてトレースを開始します。!delay, 1ffffは、変更していなければ指定する必要はありません。トリガ成立後は、ディレイカウンタ値:0x1ffffサイクル分取り込んだ後、トレースは自動的に終了します。その結果、トリガポイントの前後、約0x20000サイクルがトレースに入ります。
rscsngl1 1000
rscsngl2 2000
trcenable !always start rscsngl_1 stop rscsngl_2
tron
0x1000番地の実行直後より、トレースの取り込みを開始し、0x2000番地の実行で一時的にトレースの取り込みを中止します。

[備考]
トレースに関する詳細は、「付録.Ａ トレース機能の詳細」を参照ください。

ｔｒｏｆｆコマンド

[書式]
troff

[パラメータ]
なし

[機能]
トレースの取り込みを強制的に終了します。

ｔｒａｃｅコマンド

[書式]
trace [POS] [all|pc] [asm|ttag1|ttag2]

[パラメータ]
POS=±0..xxxx: トリガサイクル近辺または終了サイクルを０として、トレースの表示開始位置を１６進数で指定します。
all|pc|data: 取り込んだトレース情報の中から選択して表示するサイクルの指定します。
all: 全てのサイクル
pc: 実行サイクルのみ
asm|ttag1|ttag2: 表示種別を指定します。
Asm: アセンブラ表示のみ
ttag1: アセンブラ表示＋絶対時間でのタイムタグ表示
ttag2: アセンブラ表示＋相対時間でのタイムタグ表示

[機能]
トレースバッファの内容を表示します。
トレース中にこのコマンドを発行した場合、強制的に取り込みを終了し表示します。

[表示例]

```
> trace pc asm -8
  Cycle  Sub     Address  Code      Instruction                EXT  Stat
 -000008 ----    07c0072c 3000e5c4  STRB    r3,[r4]            1111 ID
 -000006 ----    07c00730 4002e1a0  MOV     r4,r2              1111 IE
 -000004 ----    07c00734 2001e242  SUB     r2,r2,#1           1111 IE
 -000002 ----    07c00738 0000e354  CMP     r4,#0              1111 IE
 +000002 ----    07c0073c fff88aff  BHI     7c00724            1111 IE
 +000004 ----    07c00724 4001e1a0  MOV     r4,r1              1111 IE
 +000006 ----    07c00728 1001e281  ADD     r1,r1,#1           1111 IE
 +000008 ----    07c0072c 3000e5c4  STRB    r3,[r4]            1111 ID
 +00000a ----    07c00730 4002e1a0  MOV     r4,r2              1111 IE
 +00000c ----    07c00734 2001e242  SUB     r2,r2,#1           1111 IE

> trace all asm -8
  Cycle  Sub     Address  Code      Instruction                EXT  Stat
 -000008 ----    07c0072c 3000e5c4  STRB    r3,[r4]            1111 ID
 -000008 0001    ------0f ------0b  [data access]                  DATA
 -000006 ----    07c00730 4002e1a0  MOV     r4,r2              1111 IE
 -000004 ----    07c00734 2001e242  SUB     r2,r2,#1           1111 IE
 -000002 ----    07c00738 0000e354  CMP     r4,#0              1111 IE
 +000002 ----    07c0073c fff88aff  BHI     7c00724            1111 IE
 +000004 ----    07c00724 4001e1a0  MOV     r4,r1              1111 IE
 +000006 ----    07c00728 1001e281  ADD     r1,r1,#1           1111 IE
 +000008 ----    07c0072c 3000e5c4  STRB    r3,[r4]            1111 ID
 +000008 0001    ------10 ------0b  [data access]                  DATA
 +00000a ----    07c00730 4002e1a0  MOV     r4,r2              1111 IE
 +00000c ----    07c00734 2001e242  SUB     r2,r2,#1           1111 IE

> trace ttag1 -8
  Cycle  Sub     Address  Code      Instruction                EXT  Stat
 -000008 ----    07c0072c 3000e5c4  STRB    r3,[r4]            1111 ID
                            time = 000,001,332,783.2uS
 -000008 0001    ------0f ------0b  [data access]                  DATA
 -000006 ----    07c00730 4002e1a0  MOV     r4,r2              1111 IE
```

```
                          time = 000,001,332,783.3uS
  -000004 ----    07c00734 2001e242  SUB     r2,r2,#1              1111 IE
                          time = 000,001,332,783.5uS
  -000002 ----    07c00738 0000e354  CMP     r4,#0                 1111 IE
                          time = 000,001,332,783.6uS
  +000002 ----    07c0073c fff88aff  BHI     7c00724               1111 IE
                          time = 000,001,332,784.1uS
  +000004 ----    07c00724 4001e1a0  MOV     r4,r1                 1111 IE
                          time = 000,001,332,784.2uS
  +000006 ----    07c00728 1001e281  ADD     r1,r1,#1              1111 IE
                          time = 000,001,332,784.4uS
  +000008 ----    07c0072c 3000e5c4  STRB    r3,[r4]               1111 ID
                          time = 000,001,332,784.5uS
  +000008 0001    ------10 ------0b  [data access]                     DATA

> trace ttag2
   Cycle  Sub     Address  Code      Instruction                   EXT  Stat
  -000008 ----    07c0072c 3000e5c4  STRB    r3,[r4]               1111 ID
                          time = 000,000,000,000.1uS
  -000008 0001    ------0f ------0b  [data access]                     DATA
  -000006 ----    07c00730 4002e1a0  MOV     r4,r2                 1111 IE
                          time = 000,000,000,000.1uS
  -000004 ----    07c00734 2001e242  SUB     r2,r2,#1              1111 IE
                          time = 000,000,000,000.2uS
  -000002 ----    07c00738 0000e354  CMP     r4,#0                 1111 IE
                          time = 000,000,000,000.1uS
  +000002 ----    07c0073c fff88aff  BHI     7c00724               1111 IE
                          time = 000,000,000,000.5uS
  +000004 ----    07c00724 4001e1a0  MOV     r4,r1                 1111 IE
                          time = 000,000,000,000.1uS
  +000006 ----    07c00728 1001e281  ADD     r1,r1,#1              1111 IE
                          time = 000,000,000,000.2uS
  +000008 ----    07c0072c 3000e5c4  STRB    r3,[r4]               1111 ID
                          time = 000,000,000,000.1uS
  +000008 0001    ------10 ------0b  [data access]                     DATA
  +00000a ----    07c00730 4002e1a0  MOV     r4,r2                 1111 IE
                          time = 000,000,000,000.1uS
  +00000c ----    07c00734 2001e242  SUB     r2,r2,#1              1111 IE
                          time = 000,000,000,000.1uS
  +00000e ----    07c00738 0000e354  CMP     r4,#0                 1111 IE
                          time = 000,000,000,000.2uS
  +000010 ----    07c0073c fff88aff  BHI     7c00724               1111 IE
                          time = 000,000,000,000.5uS
  +000012 ----    07c00724 4001e1a0  MOV     r4,r1                 1111 IE
                          time = 000,000,000,000.1uS
  +000014 ----    07c00728 1001e281  ADD     r1,r1,#1              1111 IE
                          time = 000,000,000,000.1uS
  +000016 ----    07c0072c 3000e5c4  STRB    r3,[r4]               1111 ID
                          time = 000,000,000,000.2uS
```

Cycle: トレースバッファ内の位置を16進数で相対的に表示しています。
トリガポイント位置の近辺または、トレースの最終フレームを０としています。

Sub: 命令実行以外のサイクルです。

Address: 実行アドレスまたは、バスサイクルのアドレスを表示します。

Code: 命令コードまたは、バスサイクルのデータを表示します。

Instruction:　　命令のニーモニックまたは、バスの種類を表示します。
EXT:　　外部入力端子EXI3..0の状態をビット列で表示します。
Stat:　　表示にもとになるトレースパケットの種別を表示します。

:　　トリガポイント（±10サイクル程度ずれる場合があります）
time =　タイムタグの表示

備考：タイムタグは、CPUから分岐情報が出力された時点のものです。分岐情報の出力は、実際の実行時間に対し遅れがあり、この遅れは一定ではありません。したがって、測定値には潜在的な誤差があります。また、実行直後の測定値は不定ですので無視してください。

［備考］
　トレースに関する詳細は、「付録. A トレース機能の詳細」を参照ください。

ｆｔｒａｃｅコマンド

[書式]
ftrace statpos endpos filname [trace_options]

[パラメータ]
statpos: ファイルに書き出すトレースポジションの開始位置
endpos: ファイルに書き出すトレースポジションの終了位置
filname:
trace_options: 以下のパラメータが指定できます。意味は、traceコマンドと同じです。
[all|pc] [asm|ttag1|ttag2]

[機能]
トレースバッファの内容をファイルに書き出します。

[注意]
このコマンドは、処理を開始しますと途中でキャンセルできませんので、パラメータの入力には十分ご注意ください。大きな範囲を指定した場合、処理に時間がかかります。

ｔｄａｔａ＿ｄｌｙコマンド

[書式]
tdata_dly [off|small|medium|large]

[パラメータ]
off: 補正しません。
small: 最小の補正をします。
medium: 中程度の補正をします。（初期値）
large: 最大の補正をします。

[機能]
トレースクロックに対するトレースデータのセットアップ時間を調整するためのコマンドです。
セットアップ時間はoffが一番小さく、largeが一番大きくなります。尚、実際のセットアップ値は使用するRTE-xxxx-TP本体やケーブルに依存しますので、各本体の仕様を確認ください。

[補足]
通常は初期値から変更する必要はありませんが、CPUやボードの状態によっては調整が必要になる場合があります。

[特記事項]
KIT-OMAP161x_ARM-TP(-H)の場合
以下にKIT-OMAP161x_ARM-TP(-H)の場合のパラメータを記します。

tdata_dly [off|small|medium|large|special1]
Special1: OMAP16xxに最適化した特別な設定になっていますので、変更しないでください。

ｖｅｒコマンド

［書式］
ver

［パラメータ］
なし

［機能］
ＩＣＥ制御用ファームウェアのバージョンを表示します。

ｗｏｒｋ＿ａｄｄｒコマンド

[書式]
work_addr [unused|emem|nosave|save] [PHYSICAL_ADDR]

[パラメータ]
command: コマンド名を指定します。
コマンド名を省略した場合、コマンドの一覧が表示されます。
unused|emem|nosave|save: ワーク領域の
unused: キャッシュ領域を使用しない場合に指定します。
emem: エミュレーションメモリをワーク領域に指定します。（初期値です）
エミュレーションメモリが接続され、正しく割りついている場合に限り指定できます。この場合、PHYSICAL_ADDRの指定は不要です。
メモリ内容は非破壊（保存される）で使用されます。
nosave: PHYSICAL_ADDRで指定したユーザシステム上のメモリを指定します。
メモリ内容は破壊的（保存されない）に使用されますので、余っている空間がある場合に指定します。
save: PHYSICAL_ADDRで指定したユーザシステム上のメモリを指定します。
メモリ内容は非破壊的（保存される）に使用されますので、メモリに空きがない場合に指定します。但し、セーブと書き戻しの操作が必要ですので、ブレークや実行開始前後の処理速度が少し遅くなる場合があります。
PHYSICAL_ADDR: ユーザシステム上のRAMのアドレスを指定します。いつでも読み書きできる状態にあることが必要です。

[機能]
キャッシュを使用している場合、デバッガの処理上の都合により、128バイトのユーザシステム上のメモリをワークとして使用する場合があります。その領域を指定するコマンドです。

[注意事項]
ユーザシステム上のRAMを指定する場合、いつでも正しくアクセスできるメモリを指定してください。

[使用例]
work_addr emem
エミュレーションメモリを使用します。
work_addr save 1000
0x1000番地から128バイトを非破壊で使用します。

[備考]
このコマンドは以下のKITでのみ有効です。
・KIT-ARM922T-TP(-H)
・KIT-OMAP_ARM-TP(^H)

ｂｏｏｔｗａｉｔコマンド

[書式]
bootwait [second]

[パラメータ]
second: 秒数を入力します。

[機能]
リセットコマンドにおいて、ICEからCPUへのリセット解除後、ICEのモニタに引き込むまでの待ち時間を設定するコマンドです。Altera社のExcaliburでは、リセット解除後、CPU自身の設定プログラムが実行されます。この実行が完了するまではCPUが本来の仕様で動作しませんので、この処理の時間を待つことが目的です。通常は2秒程度の時間を設定します。

[注意事項]
この設定値は、デバッガを抜けるまでクリアされません。（initコマンドではクリアされません。）

[使用例]
bootwait 2
2秒の待ち時間を設定します。

[備考]
このコマンドは以下のKITでのみ有効です。
・KIT-ARM922T-TP(-H)