付録. Ｂ　ＫＩＴ－ＥＰ３－ＴＰ-Ｈ　内部コマンド

本書は、KIT-EP3-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。

ＧＨＳ－Ｍｕｌｔｉの場合

ＲＴＥＳＥＲＶ２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

ご注意：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。

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。）

ａｂｐ，ａｂｐ１，ａｂｐ２，ａｂｐ３，ａｂｐ４コマンド

[書式]
```
abp [or[12]] [or34]
abp{1|2|3|4} [ADDR [AMASK]] [data DATA [DMASK]] [asid ASID|noasid] [aeq|aneq]
             [deq|dneq][read|write|accs] [byte|hword|word|nosize]
abp{1|2|3|4} /del
```

[パラメータ]
```
abp [or[12]] [or34]: abp1とabp2,abp3とabp4の組み合わせの条件を指定します。
     or|or12:   abp1 又は、abp2のどちらかの発生でブレークします。
     or34:      abp3 又は、abp4のどちらかの発生でブレークします。
abp{1|2|3|4}:  abp1-4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。
asid ASID|noasid: 将来の拡張用です。noasidでご使用ください。
   aeq|aneq:    アドレスの比較条件を指定します。
     aeq:       アドレスをイコールで比較します。
     aneq:      アドレスをノットイコールで比較します。
deq|dneq:       データの比較条件を指定します。
     deq:       データをイコールで比較します。
     dneq:      データをノットイコールで比較します。
exec|read|write|accs: 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。
byte|hword|word|nosize: アクセスサイズの指定します。
     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|3|4} /del: 条件の解除を行います。
     /del:      解除を指定します。
```

[機能]
```
 ４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します。
 実行アドレスの指定もできます。
```

[使用例]
```
 abp or
     abp1 or abp2 を指定します。
 abp2 1000 data 5555 0 aeq deq 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。
```

[備考]
```
   abpコマンドは、wpコマンドと資源を共有しています。そのため、wpで使用中のポイントは
   使用できません。
```

ｅｎｖ，ｅｍｅｍｓｔａｔコマンド

[書式]
```
env [[!]auto] [[!][verify]] [jtag[xxx][.[yyy]]{M|K}]
    [[!]resetz] [[!]hldrqz] [[!]stopz] [[!]waitz]
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。

[!]verify: メモリへの書き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-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。

[!]resetz: RESET信号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。

[!]hldrqz: HLDRQ信号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。

[!]stopz: STOP信号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。

[!]waitz: WAIT信号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DCUの状態を表示します。
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:
 Unit       : RTE-2000(H)-TP                  << RTE-2000(H)-TPが接続されています
 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:
 RESETZ       = NO MASK (!resetz)
 HLDRQZ       = NO MASK (!hldrqz)
 STOPZ        = NO MASK (!stopz)
 WAITZ        = NO MASK (!waitz)

>ememstat
 Board_num  EMEM_Size  ROM_Probe
```

```
      ==================================
        ROM1    32Mbyte    Extend Type 2K        << EMEMボードの実装状態を表示します。
```

[入力例]

env reset
RESETをマスクします。

env verify
Verify機能をONにします。

env jtag40m
JTAGクロックを40MHzに設定します。

ｅｍｏｄｅコマンド

[書式]
emode

[パラメータ]
なし

[機能]
イベントの設定状態を表示します。

[表示例]
以下は、初期状態の表示です。

```
Event Condition Settings:        << EVTコマンドの設定状態を表示
 evt brk      !seq
 evt seqclr   !seq
 evt seq1     !seq
 evt seq2     !seq
 evt seq3     !seq
 evt seq4     !seq
 evt secon    !seq
 evt secoff   !seq
 evt qualify  !seq
 evt tout     !seq
 evt match    !seq
Event Settings (execute):        << EVEコマンドの設定状態を表示
     ch Address   ASID   Cmp
 eve  1 /del
 eve  2 /del
 eve  3 /del
 eve  4 /del
 eve  5 /del
 eve  6 /del
 eve  7 /del
 eve  8 /del
Event Settings (access):         << EVAコマンドの設定状態を表示
     ch Address         Data    D_Mask   ASID   A_Cmp D_Cmp Kind  Size
 eva  1 /del
 eva  2 /del
 eva  3 /del
 eva  4 /del
 eva  5 /del
 eva  6 /del
Sequence Condigion Settings:     << SEQコマンドの設定状態を表示
 seq 1 step4
```

ｅｖａコマンド

[書式]
eva {1..6} [ADDR] [data DATA [MASK]] [asid ASID|noasid] [eq|lt|gt|neq|lte|gte|ign] [deq|dneq] [read|write|accs] [byte|hword|word|nosize] [/del]

[パラメータ]
eva {1..6}: アクセス系イベントのチャンネル(1-6)を指定します。
ADDR: アドレスを16進数で指定します。
data DATA [MASK]: データ条件の指定
DATA: データを16進数で指定します。
MASK: 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。
asid ASID|noasid: ASID条件の指定
ASID: ASID値を16進数で指定します。
noasid: ASIDを条件として使用しない指定です。
eq|lt|gt|neq|lte|gte|ign:
eq: ADDRで指定したアドレス = イベントアドレスで条件成立を指定します。
lt: ADDRで指定したアドレス > イベントアドレスで条件成立を指定します。
gt: ADDRで指定したアドレス < イベントアドレスで条件成立を指定します。
neq: ADDRで指定したアドレス != イベントアドレスで条件成立を指定します。
lte: ADDRで指定したアドレス => イベントアドレスで条件成立を指定します。
gte: ADDRで指定したアドレス =<イベントアドレスで条件成立を指定します。
ign: ADDRを比較条件として使用しない指定です。
deq|dneq: データの比較条件を指定します。
deq: データをイコールで比較します。
dneq: データをノットイコールで比較します。
read|write|accs: 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。
byte|hword|word|nosize: アクセスサイズの指定します。
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。
eva {1..6} /del:条件の解除を行います。
/del: 解除を指定します。

[機能]
アクセス系のイベントを設定します。指定したイベントは、EVTコマンドで統合して、ブレークやトレースの条件として使用できます。

[使用例]
eva 1 ffff000 data 55 00 byte read
デフォルトからの指定で、0xffff000番地から0x55のリードサイクルをEVA ch1に設定します。
eva 1 /del
EVA ch1の条件を解除します。

[備考]
トレースの条件で使用する場合、イベントがリードサイクル条件のときはデータ条件が無視されます。データ条件以外の条件が一致するリードサイクルがトレースの条件になります。

ｅｖｅコマンド

[書式]
eve {1..8} [ADDR] [asid ASID|noasid] [eq|lt|gt|neq|lte|gte|ign] [/del]

[パラメータ]
eve {1..8}: 実行系イベントのチャンネル(1-8)を指定します。
ADDR: アドレスを16進数で指定します。
asid ASID|noasid: ASID条件の指定
ASID: ASID値を16進数で指定します。
noasid: ASIDを条件として使用しない指定です。
eq|lt|gt|neq|lte|gte|ign:
eq: ADDRで指定したアドレス = イベントアドレスで条件成立を指定します。
lt: ADDRで指定したアドレス > イベントアドレスで条件成立を指定します。
gt: ADDRで指定したアドレス < イベントアドレスで条件成立を指定します。
neq: ADDRで指定したアドレス != イベントアドレスで条件成立を指定します。
lte: ADDRで指定したアドレス => イベントアドレスで条件成立を指定します。
gte: ADDRで指定したアドレス =<イベントアドレスで条件成立を指定します。
ign: ADDRを比較条件として使用しない指定です。
eve {1..8} /del:条件の解除を行います。
/del: 解除を指定します。

[機能]
実行系のイベントを設定します。指定したイベントは、EVTコマンドで統合して、
ブレークやトレースの条件として使用できます。

[使用例]
eve 1 1000
デフォルトからの指定で、0x1000番地の実行をEVE ch1に設定します。
eve 1 /del
EVE ch1の条件を解除します。

ｅｖｔコマンド

[書式]
```
 evt {brk|seqclr|seq1|seq2|seq3|seq4|secon|secoff|qualify|tout|match}
    [evep{[1][2][3]..[8]}] [ever{[1][3][5][7]}] [evap{[1][2][3]..[6]}]
    [evar{[1][3][5]}] [wp{[1][2][3][4]}] [[!]seq] [[!]evtin]
```

[パラメータ]

brk|seqclr|seq1|seq2|seq3|seq4|secon|secoff|qualify|tout|match:
イベントを統合する対象を指定します。

- brk: ブレーク条件を指定します。
- seqclr: シーケンシャル条件のクリア条件を指定します。
- seq1: シーケンシャル条件の初段の条件を指定します。
- seq2: シーケンシャル条件の2段目の条件を指定します。
- seq3: シーケンシャル条件の3段目の条件を指定します。
- seq4: シーケンシャル条件の4段目の条件を指定します。
- secon: トレースのセクション"ON"の条件を指定します。
- secoff: トレースのセクション"OFF"の条件を指定します。
- qualify: トレースのクォリファイの条件を指定します。
- tout: トリガ出力の条件を指定します。
- match: トレーストリガの条件を指定します。

evep{[1][2][3]..[8]}: eveコマンドで指定したイベントを単独でポイントとして指定します。数字をつけない場合、解除を意味します。
[1][2][3]..[8]: eveで指定したチェンネル番号と１対１で対応します。

ever{[1][3][5][7]}:eveコマンドで指定したイベントを複合してエリアとして指定します。
数字をつけない場合、解除を意味します。

- 1: eveで指定したチャンネル1と2の条件を範囲(and条件)として指定します。
- 3: eveで指定したチャンネル3と4の条件を範囲(and条件)として指定します。
- 5: eveで指定したチャンネル5と6の条件を範囲(and条件)として指定します。
- 7: eveで指定したチャンネル7と8の条件を範囲(and条件)として指定します。

evap{[1][2][3]..[6]}: evaコマンドで指定したイベントを単独でポイントとして指定します。
数字をつけない場合、解除を意味します。
[1][2][3]..[6]: evaで指定したチェンネル番号と１対１で対応します。

evar{[1][3][5]}: evaコマンドで指定したイベントを複合してエリアとして指定します。
数字をつけない場合、解除を意味します。

- 1: evaで指定したチャンネル1と2の条件を範囲(and条件)として指定します。
- 3: evaで指定したチャンネル3と4の条件を範囲(and条件)として指定します。
- 5: evaで指定したチャンネル5と6の条件を範囲(and条件)として指定します。

wp{[1][2][3][4]}: wpコマンドで指定したイベントを指定します。数字をつけない場合、解除を意味します。
[1][2][3][4]: wpで指定したポイント番号と１対１で対応します。

[!]seq: シーケンシャル条件を指定します。

- seq: シーケンシャル条件を指定します。!でシーケンシャル条件を解除します。
  seq関連(secclr,seq1,seq2,...)の条件には、指定できません。

[!]evtin: 外部イベント検出条件を指定します。!で指定しません。

[機能]
eve eva wpで指定したイベントを何に使うかを指定します。

[使用例]
```
evt brk evep1234 ever5 evap12 evar3
```
ブレーク用のイベントとして、eveで指定した1から4をポイントして、5と6を範囲条件とし、evaで指定した1から2をポイントとして、3,4を範囲として使用します。
```
evt brk evep ever evap evar
```
ブレーク用のイベントとして指定した、evep ever evap evarを解除します。

[備考]

seqon, secoff, qualifyに対し、evap, evarパラメータは指定できません。
wpパラメータはtoutでのみ指定できます。
seqを使用する場合には、seqclr, seq1, seq2, seq3, seq4には、evep/everで指定するイベントのみを指定してください。
evtinパラメータは、brk, secon, secoff, tout, matchで使用できます。
トレースのセクションやクォリファイに関する詳細は、本編のトレースの章を参照ください。

ｅｖｔｉｎｅｎｖコマンド

[書式]
evtinenv {inpe1|indma|evto}
[[!]toutpe1] [[!]toutdma]] [[!]evti]
[toutclk{1|2|3|4}] [dmatoutclk{1|2|3|4}]

[パラメータ]
inpe1|indma|evto: イベント統合部への生成要因を指定します。
inpe1: CPU(PE1)へのイベント入力の生成要因を指定します。
indma: DMAへのイベント入力の生成要因を指定します。
evto: 外部デバッグ装置へのイベントトリガ出力の生成要因を指定します。
[!]toutpe1: CPU(PE1)からのイベントトリガ出力要求を生成要因に指定します。!で指定しません。
[!]toutdma: DMAからのイベントトリガ出力要求を生成要因に指定します。!で指定しません。
[!]evti: 外部デバッグ装置からの外部イベント入力を生成要因に指定します。!で指定しません。
toutclk{1|2|3|4}: CPUのイベントトリガ出力の分周比を指定します。
それぞれ、1/1|1/2|1/3|1/4に対応します。
dmatoutclk{1|2|3|4}: DMAのイベントトリガ出力の分周比を指定します。
それぞれ、1/1|1/2|1/3|1/4に対応します。

[機能]
evt、dmaevtで設定したイベント統合部へのイベント入力の生成要因を設定します。

[備考]
inpe1ではtoutpe1を、indmaではtoutdmaを、そしてevtoではevtiを指定することはできません。

ｄｍａｅｖａコマンド

[書式]
```
dmaeva {1..6} [ADDR] [data DATA [MASK]] [eq|lt|gt|neq|lte|gte|ign]
    [deq|dneq] [exec|read|write|accs] [byte|hword|word|nosize]
    [ch CHNO|allch] [[!]evtin] [[!]csext] [[!]cspe1] [/del]
```

[パラメータ]
- dmaeva {1..6}: DMAイベントのチャンネル(1-6)を指定します。
  - ADDR: アドレスを16進数で指定します。
- data DATA [MASK]: データ条件の指定
  - DATA: データを16進数で指定します。
  - MASK: 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。
- eq|lt|gt|neq|lte|gte|ign:
  - eq: ADDRで指定したアドレス = イベントアドレスで条件成立を指定します。
  - lt: ADDRで指定したアドレス > イベントアドレスで条件成立を指定します。
  - gt: ADDRで指定したアドレス < イベントアドレスで条件成立を指定します。
  - neq: ADDRで指定したアドレス != イベントアドレスで条件成立を指定します。
  - lte: ADDRで指定したアドレス => イベントアドレスで条件成立を指定します。
  - gte: ADDRで指定したアドレス =<イベントアドレスで条件成立を指定します。
  - ign: ADDRを比較条件として使用しない指定です。
- deq|dneq: データの比較条件を指定します。
  - deq: データをイコールで比較します。
  - dneq: データをノットイコールで比較します。
- read|write|accs: 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。
- byte|hword|word|nosize: アクセスサイズの指定します。
  - 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。
- ch CHNO|allch: DMAチャンネル条件の指定
  - CHNO: DMAチャンネル番号を16進数で指定します。
    0～7F: DTSチャンネル0～127
    80～FE: DMACチャンネル0～126
  - allch: DMAチャンネルを条件を使用しない指定です。
- [!]evtin: 外部イベント入力を条件に指定します。!で指定しません。
- [!]csext: チップセレクト条件(外部リソース)を指定します。!で指定しません。
- [!]cspe1: チップセレクト条件(CPU(PE1)アクセス)を指定します。!で指定しません。
- eva {1..6} /del:条件の解除を行います。
  - /del: 解除を指定します。

[機能]

アクセス系のイベントを設定します。指定したイベントは、dmaevtコマンドで統合して、ブレークやトレースの条件として使用できます。

[使用例]
```
dmaeva 1 100000 data 55 00 byte read
```
0x100000番地から0x55のリードサイクルをdmaeva ch1に設定します。
```
dmaeva 1 /del
```
dmaeva ch1の条件を解除します。

ｄｍａｅｖｔコマンド

[書式]
```
 dmaevt {brk|tout|match|secon|secoff|qualify}
    [evap{[1][2][3]..[6]}] [evar{[1][3][5]}] [[!]evtin]
```

[パラメータ]
```
brk|seqclr|seq1|seq2|seq3|seq4|secon|secoff|qualify|tout|match:
                イベントを統合する対象を指定します。
    brk:        ブレーク条件を指定します。
    tout:       トリガ出力の条件を指定します。
    match:      トレーストリガの条件を指定します。
    secon:      トレースのセクション"ON"の条件を指定します。
    secoff:     トレースのセクション"OFF"の条件を指定します。
    qualify:    トレースのクォリファイの条件を指定します。
evap{[1][2][3]..[6]}: dmaevaコマンドで指定したイベントを単独でポイントとして指定し
                ます。数字をつけない場合、解除を意味します。
    [1][2][3]..[6]: dmaevaで指定したチェンネル番号と１対１で対応します。
evar{[1][3][5]}: dmaevaコマンドで指定したイベントを複合してエリアとして指定します。
                数字をつけない場合、解除を意味します。
    1:          evaで指定したチャンネル1と2の条件を範囲(and条件)として指定します。
    3:          evaで指定したチャンネル3と4の条件を範囲(and条件)として指定します。
    5:          evaで指定したチャンネル5と6の条件を範囲(and条件)として指定します。
[!]evtin:       外部イベント検出条件を指定します。!で指定しません。
```

[機能]
```
dmaevaで指定したイベントを何に使うかを指定します。
```

[使用例]
```
dmaevt brk evap13 ever5
    ブレーク用のイベントとして、dmaevaで指定した1と3をポイントして、5と6を範囲条件
    として使用します。
dmaevt brk evap evar
    ブレーク用のイベントとして指定した、evap evarを解除します。
```

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 env
env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。
最小値は３２バイトです。省略時は、初期値が６４Ｋバイトで、以降は前回のバイト数と同じになります。

[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。

[表示例]
初期値の表示です。

```
>nc
No Memory Cache Area
No. Address    Length
 1  0f000000   00180000
 2  0f1a0000   00e60000
 3  18000000   06c00000
 4  1ec08000   013f8000
```

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。初期値の領域は、決して削除しないでください。
変更した場合、コマンドでのIOへのアクセスで、正しい値が読み出せない場合があります。

[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。
＞＞一例ですので、実際には、変更しないでください。

```
>nc
No Memory Cache Area
No. Address    Length
 1  0f000000   00180000
 2  0f1a0000   00e60000
 3  18000000   06c00000
 4  1ec08000   013f8000

>ncd 1
No Memory Cache Area
No. Address    Length
 1  0f1a0000   00e60000
 2  18000000   06c00000
 3  1ec08000   013f8000
```

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)
また、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ROMケーブルの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]
nromd 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]
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	

ｓｅｑコマンド

[書式]
seq [PASS] [step{1|2|3|4}]

[パラメータ]
PASS: シーケンス条件の成立回数を１０進数で指定します。（max4096）
step{1|2|3|4}: シーケンスの段数を指定します。
step1: seq4->pass_count_decrement
step2: seq3->seq4->pass_count_decrement
step3: seq2->seq3->seq4->pass_count_decrement
step4: seq1->seq2->seq3->seq4->pass_count_decrement

[機能]
シーケンシャル条件の設定をします。
seq1～seq4の条件は、eve,eva,evtで指定します。
シーケンス途中でseqclr条件が成立した場合、そのシーケンスは最初に戻ります。

[使用例]
seq 100 step1
seq1->seq2->seq3->seq4の条件成立が100回成立した時にseqイベントが発生します。

ｓｓｗｏｎ，ｓｓｗｏｆｆコマンド

[書式]

```
ssw{on|off} [evap{1|2|3|4|5|6} {none|read|write|accs}]
        [evar{1|3|5} {none|read|write|accs}]
        [td{1|2|3|4} {none|read|write|accs}]
        [[!]wp{1|2|3|4}] [[!]owner]
        [[!]startpoint] [[!]section] [[!]qualify] [[!]match]
        [[!]program] [[!]directbranch] [[!]cmov]
        [mainswon_wp_{[1][2][3][4]}] [mainswoff_wp_{[1][2][3][4]}]
```

[パラメータ]

sswon: サブスイッチがON時にトレースに取り込むサイクルを指定するコマンドです。

sswoff: サブスイッチがoff時にトレースに取り込むサイクルを指定するコマンドです。

evap{1|2|3|4|5|6} {none|read|write|accs}:
evaコマンドで指定したポイント条件それぞれに対し、取り込むサイクルの種類を指定します。

- none ：取り込みません。
- read ：リードサイクルのみを取り込みます。
- write：ライトサイクルのみを取り込みます。
- accs ：リードとライトの両方のサイクルを取り込みます。

evar{1|3|5} {none|read|write|accs}:
evaコマンドで指定した範囲条件それぞれに対し、取り込むサイクルの種類を指定します。

- none ：取り込みません。
- read ：リードサイクルのみを取り込みます。
- write：ライトサイクルのみを取り込みます。
- accs ：リードとライトの両方のサイクルを取り込みます。

td{1|2|3|4} {none|read|write|accs}:
tdコマンドで指定したアドレス条件それぞれに対し、取り込むサイクルの種類を指定します。

- none ：取り込みません。
- read ：リードサイクルのみを取り込みます。
- write：ライトサイクルのみを取り込みます。
- accs ：リードとライトの両方のサイクルを取り込みます。

[!]wp{1|2|3|4}: wpコマンドで指定した条件によるメッセージ出力を取り込みます。!で取り込みません。

[!]owner: オーナーシップ・トレース・メッセージを取り込みます。!で取り込みません。

[!]startpoint: tspコマンドの検出メッセージを取り込みます。!で取り込みません。

[!]section: セクションON/OFFの検出メッセージを取り込みます。!で取り込みません。

[!]qualify: クオリファイイベントの検出メッセージを取り込みます。!で取り込みません。

[!]match: evtコマンドのマッチポイント・イベントの検出メッセージを取り込みます。!で取り込みません。

[!]program: 分岐等によるプログラムの不連続検出によるトレースメッセージを取り込みます。!で取り込みません。

[!]directbranch: 直接分岐によるトレースメッセージを取り込みます。!で取り込みません。

[!]cmov: cmov命令のトレースメッセージを取り込みます。!で取り込みません。

mainswon_wp_{[1][2][3][4]}:
メインスイッチをONに切り替えるwpコマンドのポイント条件を指定します。

mainswoff_wp_{[1][2][3][4]}:
メインスイッチをOFFに切り替えるwpコマンドのポイント条件を指定します。

[機能]

サブスイッチの状態によって、トレースに取り込むサイクルの種類を指定します。

[使用例]

初期値では、サブスイッチがONの時に全てのサイクルを取り込み、OFFの時にサイクルの取り込みを行わないように指定してあります。
これにより、任意の条件でトレースの取り込みをコントロールできます。
以下に初期値の状態を示します。

```
>sswon
Sub-switch ON Settings:
  Main Switch On Watch Point  = No Watchpoint (mainswon_wp_)
  Main Switch Off Watch Point = No Watchpoint (mainswoff_wp_)
  Data Trace Point 1 Trace   = Read and Write cycle (td1 accs)
  Data Trace Point 2 Trace   = Read and Write cycle (td2 accs)
  Data Trace Point 3 Trace   = Read and Write cycle (td3 accs)
  Data Trace Point 4 Trace   = Read and Write cycle (td4 accs)
  Event Access Point 1 Trace = Not Trace (evap1 none)
  Event Access Point 2 Trace = Not Trace (evap2 none)
  Event Access Point 3 Trace = Not Trace (evap3 none)
  Event Access Point 4 Trace = Not Trace (evap4 none)
  Event Access Point 5 Trace = Not Trace (evap5 none)
  Event Access Point 6 Trace = Not Trace (evap6 none)
  Event Access Range 1 Trace = Not Trace (evar1 none)
  Event Access Range 3 Trace = Not Trace (evar3 none)
  Event Access Range 5 Trace = Not Trace (evar5 none)
  Watch Point 1 Trace        = Disable (!wp1)
  Watch Point 2 Trace        = Disable (!wp2)
  Watch Point 3 Trace        = Disable (!wp3)
  Watch Point 4 Trace        = Disable (!wp4)
  Ownership Trace            = Disable (!owner)
  Start Point Trace          = Enable  (startpoint)
  Section Event Trace        = Enable  (section)
  Qualify Event Trace        = Enable  (qualify)
  Match Event Trace          = Enable  (match)
  Program Trace              = Enable  (program)
  Direct Branch Trace        = Enable  (directbranch)
  CMOV Trace                 = Enable  (cmov)Sub-switch ON Settings:

>sswoff
Sub-switch OFF Settings:
  Main Switch On Watch Point  = No Watchpoint (mainswon_wp_)
  Main Switch Off Watch Point = No Watchpoint (mainswoff_wp_)
  Data Trace Point 1 Trace   = Not Trace (td1 none)
  Data Trace Point 2 Trace   = Not Trace (td2 none)
  Data Trace Point 3 Trace   = Not Trace (td3 none)
  Data Trace Point 4 Trace   = Not Trace (td4 none)
  Event Access Point 1 Trace = Not Trace (evap1 none)
  Event Access Point 2 Trace = Not Trace (evap2 none)
  Event Access Point 3 Trace = Not Trace (evap3 none)
  Event Access Point 4 Trace = Not Trace (evap4 none)
  Event Access Point 5 Trace = Not Trace (evap5 none)
  Event Access Point 6 Trace = Not Trace (evap6 none)
  Event Access Range 1 Trace = Not Trace (evar1 none)
  Event Access Range 3 Trace = Not Trace (evar3 none)
  Event Access Range 5 Trace = Not Trace (evar5 none)
```

```
Watch Point 1 Trace       = Disable (!wp1)
Watch Point 2 Trace       = Disable (!wp2)
Watch Point 3 Trace       = Disable (!wp3)
Watch Point 4 Trace       = Disable (!wp4)
Ownership Trace           = Disable (!owner)
Start Point Trace         = Disable (!startpoint)
Section Event Trace       = Disable (!section)
Qualify Event Trace       = Disable (!qualify)
Match Event Trace         = Disable (!match)
Program Trace             = Disable (!program)
Direct Branch Trace       = Disable (!directbranch)
CMOV Trace                = Disable (!cmov)
```

ｄｍａｓｓｗｏｎ、ｄｍａｓｓｗｏｆｆコマンド

[書式]
```
dmassw{on|off} [evap{1|2|3|4|5|6} {none|read|write|accs}]
        [evar{1|3|5} {none|read|write|accs}]
        [td{1|2|3|4} {none|read|write|accs}]
        [startpoint [none|read|write|accs]] [[!]section] [[!]qualify] [[!]match]
        [[!]status]
```
[パラメータ]

dmasswon:　サブスイッチがON時にトレースに取り込むサイクルを指定するコマンドです。

dmasswoff:　サブスイッチがoff時にトレースに取り込むサイクルを指定するコマンドです。

evap{1|2|3|4|5|6} {none|read|write|accs}:
dmaevaコマンドで指定したポイント条件それぞれに対し、取り込むサイクルの種類を指定します。
- none ：取り込みません。
- read ：リードサイクルのみを取り込みます。
- write：ライトサイクルのみを取り込みます。
- accs ：リードとライトの両方のサイクルを取り込みます。

evar{1|3|5} {none|read|write|accs }:
dmaevaコマンドで指定した範囲条件それぞれに対し、取り込むサイクルの種類を指定します。
- none ：取り込みません。
- read ：リードサイクルのみを取り込みます。
- write：ライトサイクルのみを取り込みます。
- accs ：リードとライトの両方のサイクルを取り込みます。

td{1|2|3|4} {none|read|write|accs}:
dmatdコマンドで指定したアドレス条件それぞれに対し、取り込むサイクルの種類を指定します。
- none ：取り込みません。
- read ：リードサイクルのみを取り込みます。
- write：ライトサイクルのみを取り込みます。
- accs ：リードとライトの両方のサイクルを取り込みます。

startpoint [none|read|write|accs]: dmatspコマンドで指定したアドレス条件に対し、取り込むサイクルの種類を指定します。
- none ：取り込みません。
- read ：リードサイクルのみを取り込みます。
- write：ライトサイクルのみを取り込みます。
- accs ：リードとライトの両方のサイクルを取り込みます。

[!]section:　セクションON/OFFの検出メッセージを取り込みます。!で取り込みません。

[!]qualify:　クオリファイイベントの検出メッセージを取り込みます。!で取り込みません。

[!]match:　evtコマンドのマッチポイント・イベントの検出メッセージを取り込みます。!で取り込みません。

[!]status:　DMAステータス情報のトレースメッセージを取り込みます。!で取り込みません。

[機能]

サブスイッチの状態によって、トレースに取り込むＤＭＡサイクルの種類を指定します。

[使用例]

初期値では、サブスイッチがONの時に全てのサイクルを取り込み、OFFの時にサイクルの取り込みを行わないように指定してあります。
これにより、任意の条件でトレースの取り込みをコントロールできます。
以下に初期値の状態を示します。

```
>dmasswon
DMA Sub-switch ON Settings:
 Data Trace Point 1 Trace   = Read and Write cycle (td1 accs)
 Data Trace Point 2 Trace   = Read and Write cycle (td2 accs)
 Data Trace Point 3 Trace   = Read and Write cycle (td3 accs)
 Data Trace Point 4 Trace   = Read and Write cycle (td4 accs)
 Event Access Point 1 Trace = Not Trace (evap1 none)
 Event Access Point 2 Trace = Not Trace (evap2 none)
 Event Access Point 3 Trace = Not Trace (evap3 none)
 Event Access Point 4 Trace = Not Trace (evap4 none)
 Event Access Point 5 Trace = Not Trace (evap5 none)
 Event Access Point 6 Trace = Not Trace (evap6 none)
 Event Access Range 1 Trace = Not Trace (evar1 none)
 Event Access Range 3 Trace = Not Trace (evar3 none)
 Event Access Range 5 Trace = Not Trace (evar5 none)
 Start Point Trace          = Read and Write cycle (startpoint accs)
 Section Event Trace        = Enable  (section)
 Qualify Event Trace        = Enable  (qualify)
 Match Event Trace          = Enable  (match)
 DMA Status Trace           = Enable  (status)

>dmasswoff
DMA Sub-switch OFF Settings:
 Data Trace Point 1 Trace   = Not Trace (td1 none)
 Data Trace Point 2 Trace   = Not Trace (td2 none)
 Data Trace Point 3 Trace   = Not Trace (td3 none)
 Data Trace Point 4 Trace   = Not Trace (td4 none)
 Event Access Point 1 Trace = Not Trace (evap1 none)
 Event Access Point 2 Trace = Not Trace (evap2 none)
 Event Access Point 3 Trace = Not Trace (evap3 none)
 Event Access Point 4 Trace = Not Trace (evap4 none)
 Event Access Point 5 Trace = Not Trace (evap5 none)
 Event Access Point 6 Trace = Not Trace (evap6 none)
 Event Access Range 1 Trace = Not Trace (evar1 none)
 Event Access Range 3 Trace = Not Trace (evar3 none)
 Event Access Range 5 Trace = Not Trace (evar5 none)
 Start Point Trace          = Not Trace (startpoint none)
 Section Event Trace        = Disable (!section)
 Qualify Event Trace        = Disable (!qualify)
 Match Event Trace          = Disable (!match)
 DMA Status Trace           = Disable (!status)
```

ｓｆｒコマンド　※現在は使用できません

[書式]
sfr [reg [VAL]]
sfr2 [reg [VAL]]

[パラメータ]
reg:　ＳＦＲレジスタ名を指定します。
VAL:　ＳＦＲのレジスタ値を１６進数で指定します。

[機能]
ＳＦＲレジスタ値の設定と表示を行います。

[使用例]
sfr P0
P0レジスタの値を表示します。
sfr PM0 0
PM0レジスタに0hを設定します。

ｓｆｒｆｉｌｅコマンド　※現在は使用できません

[書式]
sfrfile {/|FILENAME}

[パラメータ]
/: sfrの登録を初期値に戻します。
FILENAME: NEC社が提供するデバイスファイル名を指定します。

[機能]
sfrfileコマンドは、FILENAMEで指定したデバイスファイルからSFR情報を抽出し登録します。
抽出したSFR情報はSFRコマンドで使用できるようになります。
抽出前に登録されていた情報は削除されます。

[使用例]
sfrfile c:¥test¥d800299.800
c:¥test¥のディレクトリからデバイスファイル: d800299.800を読み込みます。

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。

ｔｄ１.. ｔｄ４コマンド

[書式]
td{1|2|3|4} [LADDR [UADDR]] [asid ASID|noasid] [/del]

[パラメータ]
td{1|2|3|4}: td1 - td4の条件指定に先立ち入力します。
LADDR: 下限アドレスを16進数で指定します。
UADDR: 上限アドレスを16進数で指定します。
asid ASID|noasid: ASID条件の指定をします。
ASID: ASID値を16進数で指定します。
noasid: ASIDを条件として使用しない指定です。
/del: 指定した条件を解除します。

[機能]
トレースに取り込むデータアクセスサイクルのアドレス範囲条件を設定します。
下限アドレス<= アクセスベースアドレス<=上限アドレスが指定範囲になります。上限アドレスを省略した場合は、下限アドレスと同じアドレスを使用します。

[使用例]
td1 100000 100fff
100000h番地～100ffffh番地のアクセスサイクルがトレースに取り込みます。

[備考]
td1..td4で指定した領域のアクセスサイクルをトレースに表示させる為には、sswon/sswoffコマンドでそれぞれについて出力するアクセスサイクルの種類を指定する必要があります。

ｔｅｎｖコマンド

[書式]
```
tenv [tclkdiv{1|2|4|6|8|16}] [[!]ddr]
     [tdwidth{4|8|24}] [size{256k|512k|1m|2m|4m}] [[!]evti]
     [dtm{1|2|3|4|5|6}] [subor|suband] [[!]sss_st_off] [[!]qss_st_off]
     [[!]sss_on_dly1] [[!]sss_off_dly1]
      [lvresume{0..26}] [lvsuspend{2..28}]
```
[パラメータ]
- tclkdiv{1|2|4|6|8|16}: CPUの動作クロックに対するトレースクロックの分周率を指定します。それぞれ、1/1|1/2|1/4|1/6|1/8|1/16に対応します。
- tdwidth{4|8|24}: トレースデータの幅を指定します。4-bit|8-bit|24-bitに対応します。初期値から変更しないでください。
- size{256k|512k|1m|2m|4m}: トレースメモリの使用容量を指定します。それぞれ256K|512K|1M|2M|4Mバイトに対応します。
- subor: サブスイッチとして、セクション条件とクォリファイ条件のオアを指定します。
- suband: サブスイッチとして、セクション条件とクォリファイ条件のアンドを指定します。
- [!]sss_st_off: セクションのサブスイッチをOFF状態から開始します。!でON状態から開始します。
- [!]qss_st_off: クォリファイのサブスイッチをOFF状態から開始します。!でON状態から開始します。
- [!]sss_on_dly1:セクションのサブスイッチがONになるタイミングを条件成立１命令後に遅延させます。!で即時ONです。
- [!]sss_off_dly1:セクションのサブスイッチがOFFになるタイミングを条件成立１命令後に遅延させます。!で即時OFFです。
- [!]ddr: 初期値でご使用ください。
- [!]evti: 初期値でご使用ください。
- dtm{1|2|3|4|5|6}: 初期値でご使用ください。
- lvresume{0..26}: 初期値でご使用ください。
- lvsuspend{2..28}:初期値でご使用ください。

[機能]
トレースの環境設定を行います。

[使用例]
```
tenv subor
```
サブスイッチは、セクションとクォリファイのＯＲでトレースします。

[備考]
トレースに関する詳細は、「付録.A」を参照ください。

ｔｓｐ１，ｔｓｐ２コマンド

[書式]
tsp{1|2} [ADDR] [asid ASID|noasid] [/del]

[パラメータ]
tsp{1|2}:　　tsp1または、tsp2の条件指定に先立ち入力します。
ADDR:　　実行アドレスを１６進数で指定します。
asid ASID|noasid: ASID条件の指定をします。
　ASID:　　ASID値を16進数で指定します。
　noasid:　ASIDを条件として使用しない指定です。
/del:　　指定したアドレスを解除します。

[機能]
２点あるトレースのスタート・ポイント（アドレス）を指定します。
指定したポイントで、トレース情報の取り込みサイクルをかえることができます。
（取り込み条件の指定は、sswon, sswoffコマンドを参照ください）

[使用例]
tsp1 100000
　スタート・ポイント１に100000hの命令実行を指定します。

[備考]
トレースに関する詳細は、「付録.A」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表示します。

[表示例]
以下にデフォルト値を表示例として示します。

```
>tmode
Trace Settings (tron):
 Delay Count       = 0007ffff
 Trace Mode        = Real Time (real)
 Trace Enable Unit = PE1/DMA (pe1 !pe2 dma)
 Start Mode        = Force Start only PE1/DMA (force_1d)
 Delay Mode        = Disable (!delay)
 Ext Trigger       = Disable (noext)
Trace Env Settings :
 >--Setting for All Trace Unit (PE1/PE2/DMA)
  TRCCLK Div.     = 1/2 (tclkdiv2)
  TRCCLK Edge     = Both (ddr)
  Suspend Level   = 16 (lvsuspend16)
  Resume Level    = 6 (lvresume6)
  TDATA Width     = 8bit (tdwidth8)
  Trace Buffer    = 1M Cycles (size1m)
  Sync. by EVTI-  = Disable (!evti)
 >--Setting for Trace Unit of each Processor Element (PEx)
  Data Trace Message Type           = with PCM ,with Access-ID (dtm4)
  Sub switch                        = <section> or <qualify> (subor)
  Section Sub Switch at force start = on (!sss_st_off)
  Qualify Sub Switch at force start = on (!qss_st_off)
  Section Sub Switch turn on delay  = immediately (!sss_on_dly1)
  Section Sub Switch turn off delay = immediately (!sss_off_dly1)
Trace Start Point Settings:
      Address   ASID
 tsp1 /del
 tsp2 /del
```

[備考]
トレースに関する詳細は、「付録.A」を参照ください。

ｔｒｏｎコマンド

[書式]
```
tron [DELAY] [[!]delay] [[!]real] [[!]pe1] [[!]dma]
    [[[!]force] |[force_{1}{d}]] [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。!で非リアルタイム実行モードになります。このモードでは、オーバ
                 フローは発生しませんが、実行速度が低下します。
[!]pe1:          pe1(CPU)のトレースを有効にします。！で無効にします。
[!]dma           dmaのトレースを有効にします。！で無効にします。
[!]force:        トレースの開始条件としてtronの最初から強制的に開始を指定します。!で
                 強制開始を解除します。その場合は、sswon、sswoffの条件により開始しま
                 す。強制開始を指定した場合もsswon,sswoffは有効です。
                 pe1(CPU),dmaの全てが強制開始になります。
force_{1}{d}: tronの最初から強制的に開始するプロセッサを指定します。
                 1がpe1(CPU)、dがdmaを示します。
noext|nega|posi: トリガとして外部入力端子(EXIO)を指定します。
    noext:       EXIOをトリガとして使用しません。
    posi:        EXIOの立ち上がりエッジをトリガとして指定します。
    nega:        EXIO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:0x3ffd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。

[備考]
トレースに関する詳細は、「付録.A」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|cpu|pe1|dma] [asm|asmtime|asmclr] [subNN]

[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| cpu|pe1|dma: 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
cpu: CPUのサイクルのみ
pe1: CPUのサイクルのみ
dma: DMAサイクルのみ
asm|asmtime|asmclr: 表示種別を指定します。
asm: アセンブラ表示のみ
asmtime: アセンブラ表示＋絶対時間でのタイムタグ表示
asmclr: アセンブラ表示＋相対時間でのタイムタグ表示
subNN: 実際に取り込まれる一つの分岐情報から連続して逆アセンブルする命令数を16進数で指定します。初期値は100h(ex:sub100)です。

[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ます。

[表示例]
```
    >trace -4 asm
       Cycle  Sub      Address  Code     Instruction             EXT  Stat
      -000004 ---- PE1:1ec00014 0e010001 addi    0001h,r1,r1     1111 EXE
              0001     1ec00018 17260004 ld.h    0004h[r6],r2         SUB
              0002     1ec0001c 16020001 addi    0001h,r2,r2          SUB
              0003     1ec00020 1f060008 ld.b    0008h[r6],r3         SUB
              0004     1ec00024 1e030001 addi    0001h,r3,r3          SUB
              0005     1ec00028 1ec300ff andi    00ffh,r3,r3          SUB
              0006     1ec0002c 05ca     bnz     1ec00034h            SUB
              0007     1ec0002e 0000     nop                          SUB
      +000000 ---- PE1:         [ MTCH ]                        1111 WP
      +000002 ---- PE1:1ec00030 1e030001 addi    0001h,r3,r3     1111 EXE
              0001     1ec00034 0000     nop                          SUB
              0002     1ec00036 0f660001 st.w    r1,0000h[r6]         SUB
              0003     1ec0003a 0000     nop                          SUB
              0004     1ec0003c 17660004 st.h    r2,0004h[r6]         SUB
              0005     1ec00040 0000     nop                          SUB
              0006     1ec00042 1f460008 st.b    r3,0008h[r6]         SUB
              0007     1ec00046 0000     nop                          SUB
              0008     1ec00048 0000     nop                          SUB
              0009     1ec0004a 0000     nop                          SUB
      +000007 ---- PE1:1ec0004c e5a5     br      1ec00010h       1111 EXE (Bcond)
              0001     1e800010 0000     nop                          SUB

    >trace asmtime
       Cycle  Sub      Address  Code     Instruction             EXT  Stat
      -000004 ---- PE1:1ec00014 0e010001 addi    0001h,r1,r1     1111 EXE
                                 time = 000,000,005,576.0uS
              0001     1ec00018 17260004 ld.h    0004h[r6],r2         SUB
              0002     1ec0001c 16020001 addi    0001h,r2,r2          SUB
              0003     1ec00020 1f060008 ld.b    0008h[r6],r3         SUB
              0004     1ec00024 1e030001 addi    0001h,r3,r3          SUB
              0005     1ec00028 1ec300ff andi    00ffh,r3,r3          SUB
              0006     1ec0002c 05ca     bnz     1ec00034h            SUB
              0007     1ec0002e 0000     nop                          SUB
```

```
  +000000 ---- PE1:          [ MTCH ]                          1111 WP
                             time = 000,000,005,580.3uS
  +000002 ---- PE1:1ec00030 1e030001 addi     0001h,r3,r3      1111 EXE
                             time = 000,000,005,581.8uS
          0001      1ec00034 0000     nop                           SUB
          0002      1ec00036 0f660001 st.w     r1,0000h[r6]       SUB
          0003      1ec0003a 0000     nop                           SUB
          0004      1ec0003c 17660004 st.h     r2,0004h[r6]       SUB
          0005      1ec00040 0000     nop                           SUB
          0006      1ec00042 1f460008 st.b     r3,0008h[r6]       SUB
          0007      1ec00046 0000     nop                           SUB
          0008      1ec00048 0000     nop                           SUB
          0009      1ec0004a 0000     nop                           SUB
  +000007 ---- PE1:1ec0004c e5a5     br       1ec00010h        1111 EXE (Bcond)
                             time = 000,000,005,590.8uS
          0001      1e800010 0000     nop                           SUB

>trace asmclr
    Cycle  Sub      Address  Code     Instruction              EXT  Stat
  -000004 ---- PE1:1ec00014 0e010001 addi     0001h,r1,r1      1111 EXE
                             time = 000,000,000,000.0uS
          0001      1ec00018 17260004 ld.h     0004h[r6],r2       SUB
          0002      1ec0001c 16020001 addi     0001h,r2,r2        SUB
          0003      1ec00020 1f060008 ld.b     0008h[r6],r3       SUB
          0004      1ec00024 1e030001 addi     0001h,r3,r3        SUB
          0005      1ec00028 1ec300ff andi     00ffh,r3,r3        SUB
          0006      1ec0002c 05ca     bnz      1ec00034h          SUB
          0007      1ec0002e 0000     nop                         SUB
  +000000 ---- PE1:          [ MTCH ]                          1111 WP
                             time = 000,000,000,000.0uS
  +000002 ---- PE1:1ec00030 1e030001 addi     0001h,r3,r3      1111 EXE
                             time = 000,000,000,000.0uS
          0001      1ec00034 0000     nop                           SUB
          0002      1ec00036 0f660001 st.w     r1,0000h[r6]       SUB
          0003      1ec0003a 0000     nop                           SUB
          0004      1ec0003c 17660004 st.h     r2,0004h[r6]       SUB
          0005      1ec00040 0000     nop                           SUB
          0006      1ec00042 1f460008 st.b     r3,0008h[r6]       SUB
          0007      1ec00046 0000     nop                           SUB
          0008      1ec00048 0000     nop                           SUB
          0009      1ec0004a 0000     nop                           SUB
  +000007 ---- PE1:1ec0004c e5a5     br       1ec00010h        1111 EXE (Bcond)
                             time = 000,000,000,012.8uS
          0001      1e800010 0000     nop                           SUB

>trace 14 asm
  +000014 ---- DMA:--        [ DMA stat=0000000008 ]           1111 DMA STAT
  +000017 ---- DMA:1ec01000 ------00  [Byte   Read]            1111 RD DMA
  +00001c ---- DMA:1ec04000 ------00  [Byte   Write]           1111 WR DMA
  +000020 ---- PE1:1ec001a4 fd89     bnc      1ec00194h        1111 EXE (Bcond)
          0001      1ec00194 301d     mov      r29,r6             SUB
          0002      1ec00196 57060000 ld.b     0000h[r6],r10      SUB
          0003      1ec0019a 100a     mov      r10,r2             SUB
          0004      1ec0019c 7002     mov      r2,r14             SUB
          0005      1ec0019e 7288     shr      00000008h,r14      SUB
          0006      1ec001a0 05b1     bc       1ec001a6h          SUB
          0007      1ec001a2 1284     shr      00000004h,r2       SUB
  +000024 ---- DMA:1ec01000 ------01  [Byte   Read]            1111 RD DMA
  +00002a ---- DMA:1ec04000 ------01  [Byte   Write]           1111 WR DMA
  +00002e ---- DMA:1ec01002 ------02  [Byte   Read]            1111 RD DMA
  +000032 ---- PE1:1ec001a4 fd89     bnc      1ec00194h        1111 EXE (Bcond)
          0001      1ec00194 301d     mov      r29,r6             SUB
          0002      1ec00196 57060000 ld.b     0000h[r6],r10      SUB
          0003      1ec0019a 100a     mov      r10,r2             SUB
```

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: 分岐や実行命令数などの情報から解析して生成したサイクルの番号です。
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。
CPUサイクルの場合は、PE1:xxxxxxxx の形式でプロセッサとアドレスを表示します。
DMAサイクルの場合は、DMA:xxxxxxxx の形式で表示します。
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(メッセージ)の種別を表示します。

EXE	命令実行
EXE ()	命令実行、分岐要因あり （()は後述）
EXE <>	命令実行、条件判定結果あり （<>は後述）
WR CPU	メモリ書き込み発生(CPU)
RD CPU	メモリ読み出し発生(CPU)
WR DMA	メモリ書き込み発生(DMA)
RD DMA	メモリ読み出し発生(DMA)
OTM	タスク／プロセスID
WP	ウォッチポイント Code表示の部分にウォッチ要因(後述)を表示します。
ERR	エラーコード Code表示の部分にエラーコードを表示します。
DMA STAT	DMAステータス

"()"部分は次の文字列が入ります。
これは、分岐要因となった命令もしくは事象です。

NMI/INT	割り込みの発生によるもの
EXP/TRAP	例外の発生によるもの
RETI	当該命令によるもの
JMP	当該命令によるもの
JR	当該命令によるもの
JARL	当該命令によるもの
Bcond	当該命令によるもの
CALLT	当該命令によるもの
SWITCH	当該命令によるもの
DISPOSE	当該命令によるもの
CTRET	当該命令によるもの
SYSCALL	当該命令によるもの

"<>"部分は次の文字列が入ります。
これは、条件判定があった場合の結果です。

T	成立
NT	不成立

ウォッチ要因には次の文字列が入ります。

WP1..4	ウォッチポイント1～4
MTCH	マッチイベント
Q-ON	クオリファイ・サブスイッチ・ON
Q-OFF	クオリファイ・サブスイッチ・OFF
S-ON	セクション・サブスイッチ・ON
S-OFF	セクション・サブスイッチ・OFF

time =　　タイムタグの表示

備考:タイムタグは、CPUからトレース情報が出力された時点のものです。トレース情報の出力は、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」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|cpu|pe1|dma] [asm|asmtime|asmclr] [subNN]

[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。なお、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。

ｄｍａｔｄ１... ｄｍａｔｄ４コマンド

[書式]
```
dmatd{1|2|3|4} [LADDR [UADDR]] [ch CHNO|allch]
    [[!]csext] [[!]cspe1] [/del]
```

[パラメータ]
```
td{1|2|3|4}: dmatd1 - dmatd4の条件指定に先立ち入力します。
LADDR:          下限アドレスを16進数で指定します。
UADDR:          上限アドレスを16進数で指定します。
ch CHNO|allch: DMAチャネル条件の指定
    CHNO:       DMAチャネル番号を16進数で指定します。
                 0～7F: DTSチャンネル0～127
                80～FE: DMACチャンネル0～126
    allch:      DMAチャネルを条件を使用しない指定です。
[!]csext:       チップセレクト条件(外部リソース)を指定します。!で指定しません。
[!]cspe1:       チップセレクト条件(PE1アクセス)を指定します。!で指定しません。
/del:           指定した条件を解除します。
```

[機能]
トレースに取り込むＤＭＡアクセスサイクルのアドレス範囲条件を設定します。
下限アドレス<= アクセスベースアドレス<=上限アドレスが指定範囲になります。上限アドレスを省略した場合は、下限アドレスと同じアドレスを使用します。

[使用例]
```
dmatd1 100000 100fff
    100000h番地～100ffffh番地のアクセスサイクルがトレースに取り込みます。
```

[備考]
dmatd1..dmatd4で指定した領域のアクセスサイクルをトレースに表示させるためには、dmasswon/dmasswoffコマンドでそれぞれについて出力するアクセスサイクルの種類を指定する必要があります。

ｄｍａｔｅｎｖコマンド

[書式]
```
dmatenv [subor|suband] [[!]sss_st_off] [[!]qss_st_off]
     [[!]sss_on_dly1] [[!]sss_off_dly1] [dtm{1|2|3|4|5}]
```

[パラメータ]
- subor: サブスイッチとして、セクション条件とクォリファイ条件のオアを指定します。
- suband: サブスイッチとして、セクション条件とクォリファイ条件のアンドを指定します。
- [!]sss_st_off: セクションのサブスイッチをOFF状態から開始します。!でON状態から開始します。
- [!]qss_st_off: クォリファイのサブスイッチをOFF状態から開始します。!でON状態から開始します。
- [!]sss_on_dly1:セクションのサブスイッチがONになるタイミングを条件成立１命令後に遅延させます。!で即時ONです。
- [!]sss_off_dly1:セクションのサブスイッチがOFFになるタイミングを条件成立１命令後に遅延させます。!で即時OFFです。
- dtm{1|2|3|4|5}:初期値でご使用ください。

[機能]
ＤＭＡトレースの環境設定を行います。

[使用例]
```
dmatenv subor
```
サブスイッチは、セクションとクォリファイのＯＲでトレースします。

ｄｍａｔｓｐ１、ｄｍａｔｓｐ２コマンド

[書式]
dmatsp{1|2} [ADDR] [ch CHNO|allch] [[!]csext] [[!]cspe1] [/del]

[パラメータ]
dmatsp{1|2}: dmatsp1または、dmatsp2の条件指定に先立ち入力します。
ADDR: アドレスを１６進数で指定します。
ch CHNO|allch: DMAチャネル条件の指定
CHNO: DMAチャネル番号を16進数で指定します。
0～7F: DTSチャンネル0～127
80～FE: DMACチャンネル0～126
allch: DMAチャネルを条件を使用しない指定です。
[!]csext: チップセレクト条件(外部リソース)を指定します。!で指定しません。
[!]cspe1: チップセレクト条件(CPU(PE1)アクセス)を指定します。!で指定しません。
/del: 指定したアドレスを解除します。

[機能]
２点あるＤＭＡトレースのスタート・ポイント（アドレス）を指定します。
指定したポイントで、トレース情報の取り込みサイクルをかえることができます。
（取り込み条件の指定は、dmasswon, dmasswoffコマンドを参照ください）

[使用例]
dmatsp1 100000
スタート・ポイント１に100000hを指定します。

ｗｐコマンド

[書式]
```
  wp{1|2|3|4} [ADDR [AMASK]] [data DATA [DMASK]] [asid ASID|noasid] [aeq|aneq]
              [deq|dneq][exec|read|write|accs] [byte|hword|word|nosize]
  wp{1|2|3|4} /del
```

[パラメータ]
```
  wp{1|2|3|4}:    wp1-4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。
  asid ASID|noasid: 将来の拡張用です。noasidでご使用ください。
     aeq|aneq:    アドレスの比較条件を指定します。
       aeq:       アドレスをイコールで比較します。
       aneq:      アドレスをノットイコールで比較します。
  deq|dneq:       データの比較条件を指定します。
       deq:       データをイコールで比較します。
       dneq:      データをノットイコールで比較します。
  exec|read|write|accs: サイクルの条件を指定します。
       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。
       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。
  byte|hword|word|nosize: アクセスサイズの指定します。
       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。
  wp{1|2|3|4} /del: 条件の解除を行います。
       /del:      解除を指定します。
```

[機能]
```
   ４点あるウォッチポイントの設定または解除します。
   イベント、トレースの条件で使用します。
```

[使用例]
```
   wp1 1000 aeq exec
       1000h番地の実行にウォッチポイントを設定します。
   wp2 1000 data 5555 0 aeq deq read hword
       1000h番地からhwordで5555hをリードした時にウォッチポイントを検出します。
   wp1 /del
       wp1の条件を解除します。
```

[備考]
```
   wpコマンドは、abpコマンドと資源を共有しています。そのため、abpで使用中のポイントは
   使用できません。
   ウォッチポイントをトレースの条件で使用する場合は、サイクル条件は実行アドレスでなけ
   ればなりません。
```

ｔｉｍｅコマンド

[書式]
time

[パラメータ]
なし

[機能]
実行時間計測結果を時間で表示します。実行時間計測のタイマーはＣＰＵが実行を開始する毎に初期化され、ＣＰＵ実行中カウントされます。計測クロックは、JTAGCLKを２分周して使用され、ns単位に換算されて表示されます。
カウンタの有効桁は31-bitで、JTAGCLKが25MHzの時、最大約160秒までの実行時間が測定できます。

[備考]
測定値は実行の開始とブレークのオーバヘッド時間（数クロック）を含みます。

[使用例]
```
>time
Time = 10,320 (ns) (12.500000MHz) [Counter=00000081]  << JTAGCLK=25MHz時の表示です。
                        |                       |_Counter 値(Hex)
                        |_ 計測クロックの周波数(JTAGCLK の 1/2)
```

ｆｒｅａｄ、ｆｗｒｉｔｅ、ｆｆｉｌｌ、ｆｌｏａｄ、ｆｓａｖｅコマンド

[書式]
```
fread [ADDR [LENGTH]]
fwrite ADDR DATA0[ DATA1[ DATA2 [ DATA3...]]]
ffill ADDR LENGTH DATA
fload ADDR FILENAME
fsave ADDR LENGTH FILENAME
```

[パラメータ]

ADDR:	アドレスを１６進数で指定します。
LENGTH:	バイト数を１６進数で指定します。
DATA:	書き込みデータ
FILENAME:	ファイル名を指定します。

[機能]

fread [ADDR [LENGTH]]

ADDRで指定されたアドレスからLENGTHで指定された範囲をフライバイ・リードして表示します。

fwrite ADDR DATA0[DATA1[DATA2 [DATA3...]]]

ADDRで指定されたアドレスから順に、DATA0、DATA1、DATA2...とデータをフライバイ・ライトで書き込みます。書き込みデータのサイズはACCコマンドで設定したサイズになります。

ffill ADDR LENGTH DATA

ADDRで指定されたアドレスからLENGTHで指定された範囲全体に対して､DATAをフライバイ・ライトで書き込みます。書き込みデータのサイズはACCコマンドで設定したサイズになります。

fload ADDR FILENAME

ADDRで指定されたアドレスにFILENAMEのファイルの内容をフライバイ・ライトでダンロードします。

fsave ADDR LENGTH FILENAME

ADDRで指定されたアドレスからLENGTHで指定された範囲を、フライバイ・リードで読み出し、FILENAMEのファイルに書き出します。

[備考]

フライバイ・コマンドは、プロセッサが実行中に発行できるメモリアクセスコマンド郡です。これらのコマンドでアクセスできる範囲は、CPUの仕様として許可された空間に対してのみです。

プロセッサが実行中に発行できるため、フライバイ・リードでソフトウェア・ブレークポイントのアドレスを表示した場合は、プログラム・コードではなくブレーク命令が表示されます

ｖｅｒコマンド

[書式]
ver

[パラメータ]
なし

[機能]
ICE制御用のファームウェアのバージョンを表示します。