付録．Ｂ　ＫＩＴ－ＮＡ８５Ｅ２－ＴＰ（－Ｈ）　内部コマンド

本書は、ＫＩＴ－ＮＡ８５Ｅ２－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

ＲＴＥＳＥＲＶを接続後、ターゲット・ウインドウで直接入力できます。

ＰＡＲＴＮＥＲ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。

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

[パラメータ]
```
abp [or|seq]: abp1とabp2の組み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。
                abp1,及びabp2は実行アドレス条件で使用してください。
abp{1|2|3|4}:   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。
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。
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]
```
 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。
 abp1 1000 aeq exec
     1000h番地の実行にブレークを設定します。
 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。
```

ｅｎｖ，ｅｍｅｍｓｔａｔコマンド

[書式]

```
env [[!]auto] [[!][verify]] [jtag[xxx][.[yyy]]{M|K}]
    [[!]nmi0] [[!]nmi1] [[!]resetz] [[!]hldrqz] [[!]stopz]
    [[!]waitz] [[!]vsb] [[!]cpinit] [[!]vbresz]
    [romless|single0] [ilb{_hard|2|3|4|5|6|7|8|9}]
    [[!]iqueue] [dc_sizeN] [dc_wayN]
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-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。

[!]nmi0, [!]nmi1: nmi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。

[!]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。

[!]vsb: 初期値から変更しないでください。

[!]cpinit: 初期値から変更しないでください。

[!]vbresz: 初期値から変更しないでください。

romless|single0: 初期値から変更しないでください。

ilb{_hard|2|3|4|5|6|7|8|9}: 初期値から変更しないでください。

[!]iqueue: 命令キューヒット機能のON/OFFを指定します。
iqueueでON、!iqueueでOFFです。デフォルトはONです。
通常はONで使用し、E2AコアをICEする場合にのみOFFを指定してください。

dc_sizeN: データキャッシュのサイズをNで指定します。Nの単位はK-Byteで、整数を入力してください。初期値はdc_size8(8K-Byte)です。初期値以外のキャッシュを使用する場合は、正しい値を必ず設定してご使用ください。

dc_wayN: データキャッシュのWAY数をNで指定します。初期値はdc_way4(4-way)です。初期値以外のキャッシュを使用する場合は、正しい値を必ず設定してご使用ください。

> 備考：[!]iqueue, dc_sizeN, dc_wayNは、rte4win32 ver5.10.xxから追加されたパラメータです。

[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)
 CPU Mode     = single0 (single0)             << ChkRTE2.exeの指定に依存します
Signals Mask:
 NMI0         = NO MASK (!nmi0)
 NMI1         = NO MASK (!nmi1)
 RESETZ       = NO MASK (!resetz)
 HLDRQZ       = NO MASK (!hldrqz)
 STOPZ        = NO MASK (!stopz)
 WAITZ        = NO MASK (!waitz)
 VMWAIT/VMLAST/VMAHLD
              = NO MASK (!vsb)
 CPINIT       = MASK (cpinit)
 VBRESZ       = NO MASK (!vbresz)
iLB Bus Setting:
 Latency      = depend on hardware signal (ilb_hard)
Instruction Queue Setting:
 I-Queue      = Enable  (iqueue)
Data Cache Settings:
 Size         = 8Kbyte (dc_size8)
 Way          = 4 (dc_way4)

>ememstat
 Board_num  EMEM_Size  ROM_Probe
 ===================================
    ROM1    32Mbyte    Extend Type 2K        << EMEMボードの実装状態に依存します。
```

[入力例]

env reset !nmi1
　RESETをマスクし、NMI1をマスクしません。
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: 将来の拡張用です。noasidでご使用ください。
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。
ava 1 /del
EVA ch1の条件を解除します。

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: 将来の拡張用です。noasidでご使用ください。
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。
ave 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]} [[!]seq]
```

[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。

[!]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。

[機能]

eve eva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。
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。
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。

ｅｘｔｂｒｋコマンド

[書式]
extbrk [disable|posi|nega]

[パラメータ]
disable: 本機能を使用しない時に指定します。（初期値）
posi: 立ち上がりエッジを検出してブレーク要求を出します。
nega: 立ち下がりエッジを検出してブレーク要求を出します。

[機能]
外部入力信号(EXTコネクタ:RSV-IN0)から入力する信号を使って、実行をブレークする機能の指定を行う為のコマンドです。

[使用例]
extbrk posi
立ち上がりエッジを検出してブレーク要求を出します。

備考：この機能を使用する場合は、JTAG/N-Wireコネクタ にDBINT信号が接続されていなければなりません。

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

ｉｎｔ＿ｍａｓｋコマンド

[書式]
int_mask [intNN {enable|disable}]

[パラメータ]
intNN: NNが0～63でint0～int63を指定します。
enable|disable
enable: maskを解除します。
disable: maskを設定します。

[機能]
int00～int63の割込みのマスクを指定します。指定した割り込みはCPUに受け付けられなくなります。

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、最小値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。

[表示例]
初期値の表示です。

```
>nc
No Memory Cache Area
No. Address    Length
 1  1ffff000   00001000
```

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  00100000   03ef0000
 2  0ffff000   00001000

>ncd 1
 No Memory Cache Area
 No. Address  Length
 1  03fff000   00001000
```

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]
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} [{exec_{[0]..[e]}|exec_default}]
            [evap{1|2|3|4|5|6} {none|read|write|accs|readp|writep|accsp}]
            [evar{1|3|5} {none|read|write|accs|readp|writep|accsp}]
            [all_cycle {none|read|write|accs|readp|writep|accsp}]
```
[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。

exec_{[0]..[e]}:実行系のトレースに取り込むサイクルを指定します。
番号との対応付けは、以下の通りです。取り込みを制限した場合、トレースの逆アセンブル表示は正しく行えない場合があります。

0:Interrupt, 1:Exception, 2:RETI, 3:JMP, 4:JR, 5:JARL,
6:Condition Jump(not taken), 7:Condition Jump(taken),
8:CALLT, 9:SWITCH, a:DISPOSE, b:CTRET,
c:tp, d:evt_match

exec_default:　全てのサイクルを取り込みます。（'exec_0123456789abcd'と等価）
通常この状態で使用してください。

evap{1|2|3|4|5|6} {none|read|write|accs|readp|writep|accsp}:
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。
- readp：リードサイクルとその実行サイクルを取り込みます。
- writep:ライトサイクルとその実行サイクルを取り込みます。
- accsp：リードとライトのサイクルとその実行サイクルを取り込みます。

evar{1|3|5} {none|read|write|accs|readp|writep|accsp}:
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。
- readp：リードサイクルとその実行サイクルを取り込みます。
- writep:ライトサイクルとその実行サイクルを取り込みます。
- accsp：リードとライトのサイクルとその実行サイクルを取り込みます。

all_cycle {none|read|write|accs|readp|writep|accsp}:
無条件に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。
- readp：リードサイクルとその実行サイクルを取り込みます。
- writep:ライトサイクルとその実行サイクルを取り込みます。
- accsp：リードとライトのサイクルとその実行サイクルを取り込みます。

[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:
 Trace execute cycle           = exec_0123456789abcd (exec_default)
 evap1 Trace cycle (evap1)     = No cycle (none)
 evap2 Trace cycle (evap2)     = No cycle (none)
 evap3 Trace cycle (evap3)     = No cycle (none)
 evap4 Trace cycle (evap4)     = No cycle (none)
 evap5 Trace cycle (evap5)     = No cycle (none)
 evap6 Trace cycle (evap6)     = No cycle (none)
 evar1 Trace cycle (evar1)     = No cycle (none)
 evar3 Trace cycle (evar3)     = No cycle (none)
 evar5 Trace cycle (evar5)     = No cycle (none)
 All access cycle  (all_cycle) = No cycle (none)

>sswoff
Sub-switch OFF Settings:
 Trace execute cycle           = exec_
 evap1 Trace cycle (evap1)     = No cycle (none)
 evap2 Trace cycle (evap2)     = No cycle (none)
 evap3 Trace cycle (evap3)     = No cycle (none)
 evap4 Trace cycle (evap4)     = No cycle (none)
 evap5 Trace cycle (evap5)     = No cycle (none)
 evap6 Trace cycle (evap6)     = No cycle (none)
 evar1 Trace cycle (evar1)     = No cycle (none)
 evar3 Trace cycle (evar3)     = No cycle (none)
 evar5 Trace cycle (evar5)     = No cycle (none)
 All access cycle  (all_cycle) = No cycle (none)
```

[備考]

サブスイッチに関する詳細は、「付録.Ａ トレース機能の詳細」を参照ください。

ｓｓｗｅｎｖコマンド

[書式]
sswenv [[!]tsp1] [[!]tsp2] [[!]secon] [[!]secoff] [[!]qualify]

[パラメータ]
[!]tsp1: tsp1をサブスイッチ・オン条件に指定します。!で指定しません。
[!]tsp2: tsp2をサブスイッチ・オフ条件に指定します。!で指定しません。
[!]secon: セクションON条件をサブスイッチ・オン条件に指定します。!で指定しません。
[!]secoff: セクションOFFをサブスイッチ・オフ条件に指定します。!で指定しません。
[!]qualify: クオリファイ条件をサブスイッチ・オン/オフ条件に指定します。
!で指定しません。

[機能]
サブスイッチの条件の指定をおこないます。

[使用例]
sswenv tsp1 tsp2
tsp1をサブスイッチ・オン条件に、tsp2をサブスイッチ・オフ条件に指定します。

ｓｆｒ，ｓｆｒ２コマンド

[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。

レジスタとして使用できる名称は以下の通りです。

＜sfrコマンドでアクセスできるレジスター覧＞

```
SFR (R/W):
  CSC0 CSC1 BPC BSC BEC BHC VSWC
  BTSC CLOK CLOKL CLOKH CADL CADH CCNT MOPR MOPRL MOPRH DXSA0 DXSA0L
  DXSA0H DXDA0 DXDA0L DXDA0H DXSA1 DXSA1L DXSA1H DXDA1 DXDA1L DXDA1H
  DXSA2 DXSA2L DXSA2H DXDA2 DXDA2L DXDA2H DXSA3 DXSA3L DXSA3H DXDA3 DXDA3L
  DXDA3H DXBC0 DXBC0L DXBC0H DXBC1 DXBC1L DXBC1H DXBC2 DXBC2L DXBC2H
  DXBC3 DXBC3L DXBC3H DXADC0 DXADC1 DXADC2 DXADC3 DXC DXCHC0 DXCHC1 DXCHC2
  DXCHC3 IMR0 IMR0L IMR0H IMR1
  IMR1L IMR1H IMR2 IMR2L IMR2H IMR3 IMR3L IMR3H PIC0 PIC1 PIC2 PIC3 PIC4
  PIC5 PIC6 PIC7 PIC8 PIC9 PIC10 PIC11 PIC12 PIC13 PIC14 PIC15 OVFIC0
  OVFIC1 CCCIC00 CCCIC01 CCCIC10 CCCIC11 CMDIC0 CMDIC1 CMDIC2 CMDIC3
  CMDIC4 CMDIC5 DMAIC0 DMAIC1 DMAIC2 DMAIC3 DMERIC SEIC0 SRIC0 STIC0
  SEIC1 SRIC1 STIC1 JVIC0 JVIC1 JVIC2 JVIC3
  JVIC4 JVIC5 JVIC6 JVIC7 JVIC8 JVIC9 JVIC10 JVIC11 PSC
  P0 P1 P2 P3 PM0 PM1 PM2 PM3 BCT0
  BCT1 DWC0 DWC1 BCC ASC BCP LBS LBC0 LBC1 FWC FIC BMC PRC MCT AHC SCR0
  RFS0 SCR1 RFS1 SCR2 RFS2 SCR3 RFS3 SCR4 RFS4 SCR5 RFS5 SCR6 RFS6 SCR7
  RFS7 CMD0 TMCD0 TTMD0 CMD1 TMCD1
  TTMD1 CMD2 TMCD2 TTMD2 CMD3 TMCD3
  TTMD3 CMD4 TMCD4 TTMD4 CMD5 TMCD5
  TTMD5 CCC00 CCC01 TMCC00 TMCC01
  SESC0 TTMC00 TTMC01 CCC10 CCC11 TMCC10 TMCC11 SESC1 TTMC10 TTMC11 NMC
  ILBEN PCIIC JAVC NPRL CPINITR DTFR0
  DTFR1 DTFR2 DTFR3 DXSTEP ASIM0
  TXB0 CKSR0 BRGC00 TREG00 TREG01 TREG02 TXS0 ASIM1 TXB1 CKSR1 BRGC10
  TREG10 TREG11 TREG12 TXS1 PWMC0
  PWMB0 PWMT00 PWMT01 PWMC1 PWMB1 PWMT10 PWMT11 PRS1TES0
  PRS1TES2
SFR (W):
  PRCMD
SFR (R):
  DXWST DXRST1 DXRST2 DXBSC DXBEC
  ISPR TMD0 TMD1 TMD2 TMD3 TMD4
  TMD5 TMC0 TTMC02 TMC1 TTMC12
  MDMNT0 MDMNT1 RXB0 ASIS0 ASIF0
  RXB1 ASIS1 ASIF1 PRS1TES1_
```

＜sfr2コマンドでアクセスできるレジスター覧＞
現在、対象のレジスタはありません。

[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

[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。

ｔｅｎｖコマンド

[書式]
tenv [subor|suband] [[!]sss_st_off] [[!]qss_st_off]
[[!]sss_on_dly1] [[!]sss_off_dly1]
[[!]add_pc] [lvresume{0..62}] [lvsuspend{1..62}]
[nonbranchNN] [[!]phold] [[!]once] [tdwidth{4|8|16|24|48}]
[tclkdiv{1|2|4}] [[!]trcce] [[!]debug]

[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。
[!]add_pc: Start/Match/Overflow時にPCを出力します。!で出力しません。
lvresume{0..62}:初期値でご使用ください。
lvsuspend{1..62}:初期値でご使用ください。
nonbranchNN: 連続したアドレスの実行が継続した場合にPC情報をトレースに取り込む間隔を指定します。NNは、 0 – fffの範囲で指定します。 NN=0 は無限大を示します。通常は、初期値"0"でご使用ください。
[!]phold 完全モード（ノンリアルタイムモード）でトレース中、実行が停止したときにその状態を知らせるパケットをトレースに取り込みます。!で取り込みません。
[!]once: トリガ出力としてトリガ条件成立時に１回だけ出力します。
!では、条件成立時に毎回出力します。
tdwidth{4|8|16|24|48}:トレースデータの幅を指定します。
それぞれ、4-bit|8-bit|16-bit|24-bit|48-bitに対応します。
tclkdiv{1|2|4}: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に対応します。
[!]trcce: トレースの圧縮機能をON/OFFします。
trcceでON、!traceで強制的にOFFです。

補足：[!]trcceパラメータは、rte4win32のバージョン5.10.02以上で対応されたものです。

[!]debug: 初期値 (!debug)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 dmatrc
サブスイッチは、セクションとクォリファイのオアでトレースします。

[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。

ｔｐコマンド

[書式]
tp [ADDR] [asid ASID|noasid] [/del]

[パラメータ]
ADDR: 偶数アドレスを１６進数で指定します。（A0は、常にＯに補正されます）
asid ASID|noasid: 将来の拡張用です。noasidでご使用ください。
/del: 指定したアドレスを解除します。

[機能]
トレースのトリガポイントを指定します。
トレースは、トリガポイントを基点にしてその前後の実行状態を取り込むことができます。

[使用例]
tp 100000
10000hの命令実行をトリガポイントとして指定します。

[注意事項]
tronコマンドでdelay modeが指定されている場合、トリガポイントの指定は無視されます。
この場合、tron !delayと入力してdelay modeを解除してください。
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。

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: 将来の拡張用です。noasidでご使用ください。
/del: 指定したアドレスを解除します。

[機能]
２点あるトレースのセクション・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
セクション・ポイント１に100h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 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   = 0000ffff
 Trace Mode    = Real Time (real)
 Start Mode    = Force Start (force)
 Delay Mode    = Disable (!delay)
 Ext Trigger   = Disable (noext)
Trace Env Settings :
 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)
 Add PC infomation                 = Disable (!add_pc)
 Non-branch    = None (nonbranch0)
 Resume Level  = 0 (lvresume0)
 Suspend Level = 1 (lvsuspend1)
 PHOLD         = Disable (!phold)
 ONCE          = Disable (!once)
 TDATA Width   = 16bit (tdwidth16)
 TRCCLK Div.   = 1/2 (tclkdiv2)
 Debug Mode    = Disable (!debug)
Trace Switch Point Settings:
      Address   ASID
 tsp1 /del
 tsp2 /del
Trigger Point Settings:
    Address  ASID
 tp /del
```

[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。

ｔｒｏｎコマンド

[書式]
tron [DELAY] [[!]delay] [[!]real] [[!]force] [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。
[!]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、tsp1の条件により開始します。強制開始を指定した場合もtsp1, tsp2は、有効です。
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: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。
tp 1000
tron !delay 1ffff
tpの条件成立をトリガポイントとしてトレースを開始します。!delay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。

tsp1 1000
tsp2 2000
tp 1800
tron !force
トレースパケットの取り込み条件は、tsp1の条件成立後はsswonコマンドの指定値に、tsp2の条件成立後はsswoffコマンドの指定値になります。初期値では、sswonコマンドでパケットの取り込み、sswoffで取り込みの停止を指定していますので、この設定では、tsp1で指定した0x1000番地の実行直後より、トレースの取り込みを開始し、tsp2で指定した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。この間にtpで指定した0x1800の実行があった場合、それをトリガポイントとして、ディレイサイクル値（初期値0x1ffff）分のパケットをトレースして取り込みを終了します。

```
tsp1 /del
tsp2 /del
tron force
    tsp1, tsp2を解除して、強制開始でトレースを開始します。
```

[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|data] [asm|ttag1|ttag2] [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|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
data: データサイクルのみ
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
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80h(ex:sub80)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 asm -10
   Cycle  Sub  Address     Code     Instruction              EXT  Stat
 H-000011 ---- 00:00001e32 500f     mov      r15,r10         1111 JMPD  Bcond
  -000011 0001 00:00001e34 7a5f     add      ffffffffh,r15   1111 SUB
  -000011 0002 00:00001e36 51e0     cmp      r0,r10          1111 SUB
 H-00000e ---- 00:00001e38 ddcb     bh       00001df0h       1111 JMPS  Bcond
 H-00000b ---- 00:00001df0 57e90001 ld.hu    0000h[r9],r10   1111 JMPD  Bcond
  -00000b 0001 00:00001df4 880a     mov      r10,r17         1111 SUB
  -00000b 0002 00:00001df6 8002     mov      r2,r16          1111 SUB
  -00000b 0003 00:00001df8 89f0     cmp      r16,r17         1111 SUB
  -000008 ---- 00:04020000 ----4d53 [Hword  Read]           1111 DATAR
 H-000002 ---- 00:00001dfa 15a2     bz       00001e1eh       1111 JMPSS Bcond
* +000000 ---- --:-------- -------- [Trace Trigger]          1111 MATCH
 H+000001 ---- 00:00001e1e 4a42     add      00000002h,r9    1111 JMPD  Bcond
  +000001 0001 00:00001e20 1241     add      00000001h,r2    1111 SUB
  +000001 0002 00:00001e22 00c2     zxh      r2              1111 SUB
  +000001 0003 00:00001e24 6002     mov      r2,r12          1111 SUB
  +000001 0004 00:00001e26 5e80ffff ori      ffffh,r0,r11    1111 SUB
  +000001 0005 00:00001e2a 61eb     cmp      r11,r12         1111 SUB
 H+000003 ---- 00:00001e2c 05ba     bnz      00001e32h       1111 JMPSS Bcond
 H+000005 ---- 00:00001e32 500f     mov      r15,r10         1111 JMPD  Bcond
  +000005 0001 00:00001e34 7a5f     add      ffffffffh,r15   1111 SUB

> trace ttag1
   Cycle  Sub  Address     Code     Instruction              EXT  Stat
 H-000011 ---- 00:00001e32 500f     mov      r15,r10         1111 JMPD  Bcond
                                 time = 000,000,002,027.5uS
  -000011 0001 00:00001e34 7a5f     add      ffffffffh,r15   1111 SUB
  -000011 0002 00:00001e36 51e0     cmp      r0,r10          1111 SUB
 H-00000e ---- 00:00001e38 ddcb     bh       00001df0h       1111 JMPS  Bcond
                                 time = 000,000,002,027.6uS
 H-00000b ---- 00:00001df0 57e90001 ld.hu    0000h[r9],r10   1111 JMPD  Bcond
                                 time = 000,000,002,027.6uS
  -00000b 0001 00:00001df4 880a     mov      r10,r17         1111 SUB
  -00000b 0002 00:00001df6 8002     mov      r2,r16          1111 SUB
  -00000b 0003 00:00001df8 89f0     cmp      r16,r17         1111 SUB
  -000008 ---- 00:04020000 ----4d53 [Hword  Read]           1111 DATAR
```

```
                              time = 000,000,002,027.6uS
 H-000002 ---- 00:00001dfa 15a2     bz       00001e1eh        1111 JMPSS Bcond
                              time = 000,000,002,027.7uS
* +000000 ---- --:-------- -------- [Trace Trigger]           1111 MATCH
                              time = 000,000,002,027.7uS
 H+000001 ---- 00:00001e1e 4a42     add      00000002h,r9     1111 JMPD  Bcond
                              time = 000,000,002,027.7uS
  +000001 0001 00:00001e20 1241     add      00000001h,r2     1111 SUB
  +000001 0002 00:00001e22 00c2     zxh      r2               1111 SUB
  +000001 0003 00:00001e24 6002     mov      r2,r12           1111 SUB
  +000001 0004 00:00001e26 5e80ffff ori      ffffh,r0,r11     1111 SUB
  +000001 0005 00:00001e2a 61eb     cmp      r11,r12          1111 SUB
 H+000003 ---- 00:00001e2c 05ba     bnz      00001e32h        1111 JMPSS Bcond
                              time = 000,000,002,027.7uS
 H+000005 ---- 00:00001e32 500f     mov      r15,r10          1111 JMPD  Bcond
                              time = 000,000,002,027.8uS
  +000005 0001 00:00001e34 7a5f     add      ffffffffh,r15    1111 SUB
  +000005 0002 00:00001e36 51e0     cmp      r0,r10           1111 SUB
 H+000007 ---- 00:00001e38 ddcb     bh       00001df0h        1111 JMPS  Bcond
                              time = 000,000,002,027.8uS

> trace ttag2
   Cycle  Sub  Address     Code     Instruction              EXT  Stat
 H-000011 ---- 00:00001e32 500f     mov      r15,r10          1111 JMPD  Bcond
                              time = 000,000,000,000.0uS
  -000011 0001 00:00001e34 7a5f     add      ffffffffh,r15    1111 SUB
  -000011 0002 00:00001e36 51e0     cmp      r0,r10           1111 SUB
 H-00000e ---- 00:00001e38 ddcb     bh       00001df0h        1111 JMPS  Bcond
                              time = 000,000,000,000.1uS
 H-00000b ---- 00:00001df0 57e90001 ld.hu    0000h[r9],r10    1111 JMPD  Bcond
                              time = 000,000,000,000.0uS
  -00000b 0001 00:00001df4 880a     mov      r10,r17          1111 SUB
  -00000b 0002 00:00001df6 8002     mov      r2,r16           1111 SUB
  -00000b 0003 00:00001df8 89f0     cmp      r16,r17          1111 SUB
  -000008 ---- 00:04020000 ----4d53 [Hword  Read]            1111 DATAR
                              time = 000,000,000,000.0uS
 H-000002 ---- 00:00001dfa 15a2     bz       00001e1eh        1111 JMPSS Bcond
                              time = 092,216,911,462.5uS
* +000000 ---- --:-------- -------- [Trace Trigger]           1111 MATCH
                              time = 000,000,000,000.0uS
 H+000001 ---- 00:00001e1e 4a42     add      00000002h,r9     1111 JMPD  Bcond
                              time = 000,000,000,000.0uS
  +000001 0001 00:00001e20 1241     add      00000001h,r2     1111 SUB
  +000001 0002 00:00001e22 00c2     zxh      r2               1111 SUB
  +000001 0003 00:00001e24 6002     mov      r2,r12           1111 SUB
  +000001 0004 00:00001e26 5e80ffff ori      ffffh,r0,r11     1111 SUB
  +000001 0005 00:00001e2a 61eb     cmp      r11,r12          1111 SUB
 H+000003 ---- 00:00001e2c 05ba     bnz      00001e32h        1111 JMPSS Bcond
                              time = 000,000,000,000.0uS
 H+000005 ---- 00:00001e32 500f     mov      r15,r10          1111 JMPD  Bcond
                              time = 000,000,000,000.1uS
```

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。

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。

TRGSTARTON	STARTパケット発生、サブスイッチがONになった
TRGSTARTOFF	STARTパケット発生、サブスイッチがOFFになった
MATCH	MATCHパケット発生

OVF	オーバーフロー発生
TRCEND	TRCENDパケット発生
JMPD ◇	JMPDパケット発生（◇は後述）
JMPDS ◇	JMPDSパケット発生（◇は後述）
JMPS ◇	JMPSパケット発生（◇は後述）
JMPSS ◇	JMPSSパケット発生（◇は後述）
OPCODE	オペコード・アクセス（実行）発生
DATAW	メモリ書き込み発生（トレース・パケット）
DATAR	メモリ読み出し発生（トレース・パケット）
SFRW	SFR書き込み発生（バストレース）
SFRR	SFR読み出し発生（バストレース）
SUB	サブサイクル

"◇"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
BcondNT	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
STORE	データトレースでWithPCが指定されている場合
LOAD	データトレースでWithPCが指定されている場合
FSTART	トレースの強制スタート

*:	トリガポイント（多少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|data] [asm|ttag1|ttag2] [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。

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。