APPENDIX D. KIT-V850E/MA3-IE INTERNAL COMMANDS

This appendix describes the KIT-V850E/MA3-IE internal commands. These commands can be used as through commands in the debugger. For an explanation of using through commands, refer to the manual provided with the debugger.

With GHS-Multi

The through commands can be directly input in the target window after RTESERV has been connected.

With PARTNER

```
>&          << Enter through command mode.
>#ENV       << Enter an internal command.
>&          << Exit from through command mode.
```

Commands

Note These commands can be used only if the debugger does not provide equivalent functions. If these commands are issued when the debugger provides equivalent functions, a contention may occur between KIT-V850E/MA3-IE and the debugger, causing either device to malfunction.

Command syntax

The basic syntax for the KIT-V850E/MA3-IE internal commands is described below:

command-name parameter(s)

* In parameter syntax, a parameter enclosed in brackets ([ ]) is omissible. A horizontal line (|) indicates that one of the parameters delimited by it must be selected.

A command name must be an alphabetic character string, and be separated from its parameter(s) by a space or tab. A parameter must be an alphabetic character string or hexadecimal number, and be delimited by a space or tab. (A hexadecimal number cannot contain operators.)

abp, abp1, and abp2 commands

[Format]

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

[Parameters]

abp [or|and|seq]:	Specifies a condition for combination of abp1 and abp2.
or:	Break occurs if either abp1 or abp2 occurs.
and:	Break occurs if both abp1 and abp2 occur at the same time. A mask condition is used.
seq:	Break occurs if abp2 occurs after abp1.
abp{1|2}:	Input before the condition of abp1 or abp2 is specified.
ADDR [AMASK]:	Specifies an address condition.
ADDR:	Specifies addresses in hexadecimal number.
AMASK:	Specifies the mask data of an address in hexadecimal. Bits that are 1 will not be compared.
data DATA [DMASK]:	Specifies a data condition.
DATA:	Specifies data in hexadecimal.
DMASK:	Specifies the mask data of data in hexadecimal. Bits that are 1 will not be compared.
asid ASID|noasid:	For future expansion. Use noasid.
aeq|aneq:	Specifies an address comparison condition.
aeq:	Compares address for equality.
aneq:	Compares address for non-equality.
deq|dneq:	Specifies a data comparison condition.
deq:	Compares data for equality.
dneq:	Compares data for non-equality.
exec|read|write|accs:	Specifies a cycle condition.
exec:	Specifies an executable address. A data condition is ignored.
read:	Specifies a read cycle.
write:	Specifies a write cycle.
accs:	Specifies a read or write cycle.
byte|hword|word|nosize:	Specifies access size.
byte:	Specifies byte access (8 bits).
hword:	Specifies half-word access (16 bits).
word:	Specifies word access (32 bits).
nosize:	Specifies invalidity.
abp{1|2} /del:	Clears a condition.
/del:	Specifies deletion of a condition.

[Function]

These commands set or delete access breakpoints.
Up to two access breakpoints can be set.
They can specify execution addresses.

[Examples]

abp or

abp1 or abp2 is specified.

abp1 1000 aeq exec

A breakpoint for execution of address 1000h is set.

abp2 1000 data 5555 0 aeq deq read hword

Break occurs when 5555h is read in hword from address 1000h.

abp1 /del

The condition set by abp1 is deleted.

env and ememstat commands

[Format]

env [[!]auto] [[!][verify]] [jtag{25|12|5|2|1|500|250|100}]
[ram{4|12|28|60}]
[[!]nmi] [[!]intwdt] [[!]resetz] [[!]hldrqz] [[!]waitz]

ememstat

[Parameters]

[!]auto: If a breakpoint is set during execution, the breakpoint causes a temporary break. Choose [auto] to automatically perform the subsequent execution. Choose [!auto] to suppress it.

[!]verify: Specifies whether the verification after writing memory is set. Enter ! if it is not to be set.

jtag{25|12|5|2|1|500|250|100}:
Specifies the JTAG clock for internal N-Wire. Each number corresponds to the following JTAG clock.
[25 MHz|12.5 MHz|5 MHz|2 MHz|1 MHz|500 kHz|250 kHz|100 kHz]

Remark Usually, use 25 MHz (default). If the frequency lower than 1 MHz is specified, the debugger might be slowed down in operation speed or might malfunction.

[ram{4|12|28|60}]: Specifies the capacity of the internal RAM.
Each number corresponds to the following capacity:
[4 KB|12 KB|28 KB|60 KB]

[!]nmi: Specifies whether the nmi signal is to be masked. Enter ! if it is not to be masked.

[!]intwdt: Specifies whether the internal intwdt signal is to be masked. Enter ! if it is not to be masked.

[!]resetz: Specifies whether the RESET- signal is to be masked. Enter ! if it is not to be masked.

[!]hldrqz: Specifies whether the HLDRQ- signal is to be masked. Enter ! if it is not to be masked.

[!]waitz: Specifies whether the WAIT- signal is to be masked. Enter ! if it is not to be masked.

[Function]

The env command sets the emulation environment and displays the DCU status.
Enter only those parameters that need to be changed. Parameters may be entered in any order.
If the same parameter is entered twice, only the last entry is valid.
The ememstat command displays the mounting status of the E.MEM board.
Display examples are shown below:

```
>env
Probe:
  Unit       : RTE-2000-TP          <<Displays the main unit connected.
  Rom Probe  : (use ememstat command)
  Emem Size  : (use ememstat command)
```

```
CPU Settings:
   Auto Run       = ON (auto)
   JTAGCLOCK  = 25MHz (jtag25)
   Verify         = verify off (!verify)
   CPU Mode     = single (single)
   RAM Size      = 60K (ram60)
Signals Mask:
   NMI            = NO MASK (!nmi)
   INTWDT       = NO MASK (!intwdt)
   RESETZ       = NO MASK (!resetz)
   HLDRQZ      = NO MASK (!hldrqz)
   WAITZ          = NO MASK (!waitz)

>ememstat
   Board_num   EMEM_Size   ROM_Probe
    ==================================
      ROM1      32Mbyte      Extend Type 2K  <<Depends on the EMEM board mounting status.
```

[Examples]

env resetz !nmi

RESET is masked while NMI is not masked.

env verify

Sets the Verify function to ON.

emode command

[Format]

emode

[Parameter]

None

[Function]

The emode command displays the event setting status.

[Example]

The initial status is shown below as an example:

```
Event Condition Settings:          <<Displays the setting status of the evt command.
  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):          <<Displays the setting status of the eve command.
      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):           <<Displays the setting status of the eva command.
      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 Condition Settings:       <<Displays the setting status of the seq command.
  seq 1 step4
```

eva command

[Format]

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]

[Parameters]

eva {1..6}:	Specifies an access event channel (1 to 6).
ADDR:	Specifies the address in hexadecimal.
data DATA [MASK]:	Specifies a data condition.
DATA:	Specifies data in hexadecimal.
MASK:	Specifies mask data for the data in hexadecimal. Bits that are 1 will not be compared.
asid ASID|noasid:	For future expansion. Use noasid.
eq|lt|gt|neq|lte|gte|ign:	
eq:	Specifies that the condition is satisfied when the event address is equal to the address specified for ADDR.
lt:	Specifies that the condition is satisfied when the event address is smaller than the address specified for ADDR.
gt:	Specifies that the condition is satisfied when the event address is greater than the address specified for ADDR.
neq:	Specifies that the condition is satisfied when the event address is not equal to the address specified for ADDR.
lte:	Specifies that the condition is satisfied when the event address is smaller than or equal to the address specified for ADDR.
gte:	Specifies that the condition is satisfied when the event address is greater than or equal to the address specified for ADDR.
ign:	Specifies that ADDR is not used as a comparison condition.
deq|dneq:	Specifies a data comparison condition.
deq:	Compares data for equality.
dneq:	Compares data for non-equality.
read|write|accs:	Specifies a cycle condition.
read:	Specifies a read cycle.
write:	Specifies a write cycle.
accs:	Specifies a read or write cycle.
byte|hword|word|nosize:	Specifies access size.
byte:	Specifies byte access (8 bits).
hword:	Specifies half-word access (16 bits).
word:	Specifies word access (32 bits).
nosize:	Specifies invalidity.
eva {1..6} /del:	Clears a condition.
/del:	Specifies deletion of a condition.

[Function]

The eva command sets an access event. The specified event can be combined with a condition using the evt command to be used as a break or trace condition.

[Examples]

eva 1 ffff000 data 55 00 byte read

A cycle for reading 0x55 starting at address 0xffff000 is set for eva ch1 with using the default values for other parameters.

eva 1 /del

The condition of eva ch1 is cleared.

eve command

[Format]

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

[Parameters]

eve {1..8}: Specifies an execution event channel (1 to 8).

ADDR: Specifies the address in hexadecimal.

asid ASID|noasid: For future expansion. Use noasid.

eq|lt|gt|neq|lte|gte|ign:

eq: Specifies that the condition is satisfied when the event address is equal to the address specified for ADDR.

lt: Specifies that the condition is satisfied when the event address is smaller than the address specified for ADDR.

gt: Specifies that the condition is satisfied when the event address is greater than the address specified for ADDR.

neq: Specifies that the condition is satisfied when the event address is not equal to the address specified for ADDR.

lte: Specifies that the condition is satisfied when the event address is smaller than or equal to the address specified for ADDR.

gte: Specifies that the condition is satisfied when the event address is greater than or equal to the address specified for ADDR.

ign: Specifies that ADDR is not used as a comparison condition.

eve {1..8} /del: Clears a condition.

/del: Specifies deletion of a condition.

[Function]

The eve command sets an execution event. The specified event can be combined with a condition using the evt command to be used as a break or trace condition.

[Examples]

eve 1 1000

Execution of the instruction at address 0x1000 is set for eve ch1 using the default values for other parameters.

eve 1 /del

The condition of eve ch1 is cleared.

evt command

[Format]

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]

[Parameters]

brk|seqclr|seq1|seq2|seq3|seq4|secon|secoff|qualify|tout|match:
Specifies a condition with which the event is to be combined.

- brk: Specifies a break condition.
- seqclr: Specifies a sequential clear condition.
- seq1: Specifies a first-step sequential condition.
- seq2: Specifies a second-step sequential condition.
- seq3: Specifies a third-step sequential condition.
- seq4: Specifies a fourth-step sequential condition.
- secon: Specifies a trace section on condition.
- secoff: Specifies a trace section off condition.
- qualify: Specifies a trace qualify condition.
- tout: Specifies a trigger output condition.
- match: Specifies a trace trigger condition.

evep{[1][2][3]..[8]}: Specifies the corresponding event specified by the eve command as a point by itself. Specifying this parameter with no numeric characters cancels the setting.

- [1][2][3]..[8]: Each number corresponds to a channel number specified by eve.

ever{[1][3][5][7]}: Specifies each pair of events specified by the eve command as an area. Specifying this parameter with no numeric characters cancels the setting.

- 1: Specifies the conditions of channels 1 and 2 specified by eve as a range (and condition).
- 3: Specifies the conditions of channels 3 and 4 specified by eve as a range (and condition).
- 5: Specifies the conditions of channels 5 and 6 specified by eve as a range (and condition).
- 7: Specifies the conditions of channels 7 and 8 specified by eve as a range (and condition).

evap{[1][2][3]..[6]}: Specifies the corresponding event specified by the eva command as a point by itself. Specifying this parameter with no numeric characters cancels the setting.

- [1][2][3]..[6]: Each number corresponds to a channel number specified by eva.

evar{[1][3][5]}: Specifies each pair of events specified by the eva command as an area. Specifying this parameter with no numeric characters cancels the setting.

- 1: Specifies the conditions of channels 1 and 2 specified by eva as a range (and condition).
- 3: Specifies the conditions of channels 3 and 4 specified by eva as a range (and condition).
- 5: Specifies the conditions of channels 5 and 6 specified by eva as a range (and condition).

[!]seq: Specifies a sequential condition.

- seq: Specifies a sequential condition. Enter ! to cancel the sequential condition. ! cannot be specified for a seq-related condition (seqclr, seq1, seq2, seq3, or seq4).

[Function]

The evt command specifies the use of each event specified by eve or eva.

[Examples]

evt brk evep1234 ever5 evap12 evar3

As break events, the events specified for channels 1 to 4 by eve are used as points; those specified for channels 5 and 6 as a range condition; those specified for channels 1 and 2 by eva as points; and those specified for channels 3 and 4 as a range.

evt brk evep ever evap evar

The events specified for evep, ever, evap, and evar as break events are canceled.

[Remark]

For the details of the sequential conditions, see the description of the seq command.

For the details of the trace section and qualify conditions, see Appendix C, "Details of Trace Functions".

extbrk command

[Format]

extbrk [disable|posi|nega]

[Parameters]

disable:	Disables this capability (default).
posi:	Break request at positive edge detection
nega:	Break request at negative edge detection

[Function]

The extbrk command specifies the break request using external input signal (pin 4 of EXT connector (EXI1)).

[Example]

extbrk posi

A break is requested at positive edge detection.

> **Remark** To use this capability, the DBINT signal needs to be connected to the JTAG/N-Wire connector.

help command

[Format]

help [command]

[Parameter]

command: Specifies the name of the command for which you require help.
If this parameter is omitted, a list of commands is displayed.

[Function]

The help command displays a help message for a specified command.

[Example]

help map

A help message for the map command is displayed.

inb, inh, and inw commands

[Format]

inb [ADDR]
inh [ADDR]
inw [ADDR]

[Parameter]

ADDR: Specifies the address of an input port in hexadecimal.

[Function]

The inb, inh, and inw commands read the I/O space in different sizes.
The inb command accesses I/O space in bytes, inh in half words, and inw in words.

[Example]

inb 1000
The I/O space is read in bytes (8-bit units), starting at 1000H.
inh 1000
The I/O space is read in half words (16-bit units), starting at 1000H.
inw 1000
The I/O space is read in words (32-bit units), starting at 1000H.

init command

[Format]

init

[Parameter]

None

[Function]

The init command initializes KIT-V850E/MA3-IE. All environment values are initialized.
A memory cache rejection area is not initialized.

jread command

[Format]

jread [ADDR [LENGTH]]

[Parameters]

ADDR: Specifies an address in hexadecimal.

LENGTH: Specifies the number of bytes to be read, in hexadecimal. (Max.: 100h)

[Function]

The jread command reads the ROM emulation area allocated by the ROM command, via JTAG (the CPU). (Access to the ROM emulation area by ordinary commands is performed directly on internal memory.)

[Example]

jread 100000 100

100h bytes, starting at 100000h, are read via JTAG.

nc command

[Format]

nc [[ADDR [LENGTH]]

[Parameters]

ADDR: Specifies the start address of a memory cache rejection area.

LENGTH: Specifies the length of the memory cache rejection area in bytes.
The default value is 32 bytes. The allowable minimum value is also 32 bytes.

[Function]

To ensure quick memory access, KIT-V850E/MA3-IE provides a memory read cache of 8 blocks*32 bytes in the ICE. When the same memory address is accessed more than once, the read operation is not actually performed. This cache operation conflicts with the actual operation when an I/O unit is mapped onto memory. In such a case, specify a memory cache rejection area by using the nc command. Up to eight blocks can be specified as a memory cache rejection area. The allowable minimum block size is 32 bytes.

Areas other than the ROM and RAM areas are specified as memory cache rejection areas by default. Usually, the specification of memory cache rejection areas does not need to be changed.

[Example]

The initial status is shown below as an example:

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

ncd command

[Format]

ncd block-number

[Parameter]

block-number: Specifies the block number for a memory cache rejection area to be deleted.

[Function]

The ncd command deletes a memory cache rejection area. Specify the block number corresponding to the memory cache rejection area to be deleted. Do not delete any default memory cache rejection area.

If an default memory cache rejection area is deleted, accessing an I/O space by a command may fail to read correct values.

[Example]

ncd 1

Block 1 is deleted from the memory cache rejection area.

>>This is just an example. Do not delete the block actually.

```
>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 command

[Format]

nsbp [[ADDR [LENGTH]]

[Parameters]

ADDR: Specifies the start address of a software break prohibition area.

LENGTH: Specifies the length of a software break prohibition area in bytes.
The minimum unit of a specification area is the boundary of half word.
The number of the areas which can be specified is a maximum of four.

[Function]

The nsbp command specifies an area to forbid a software break.
When a breakpoint is specified, a debugger implicitly performs a memory test (write access) to an object address.
The state of some flash ROM may change by performing write access and right data may not be read. When this happens, please forbid a software break by this command to prohibit use of write cycles.
Usually, it is not necessary to specify.

[Example]

nsbp 10000 20000

A 20000-byte area, starting at 10000h, is specified as a software break prohibition area.

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

nsbpd command

[Format]

nsbpd [block-number|/all]

[Parameters]

block-number: Specifies the block number of the software break prohibition area to be deleted.

/all: Specifies all software break prohibition area to be deleted.

[Function]

The nsbpd command deletes the software break prohibition area specified by nsbp.

[Example]

nsbpd 1

Block 1 is deleted from a software break prohibition area.

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

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

nrom command

[Format]

nrom [[ADDR [LENGTH]]

[Parameters]

ADDR: Specifies the start address of a forced user area.

LENGTH: Specifies the length of a forced user area in bytes.

The minimum unit of the a specification area is as follows.

Depends on the size of the ROM being emulated.

8/16 bits: 128K bytes

32 bits: 256K bytes

(64 bits: 512K bytes)

The number of areas which can be specified is a maximum of four.

[Function]

The nrom command specifies the area when part of ROM emulation area specified by ROM command is mapped to other resources on a user system. Usually, it is not necessary to specify this command. The operations for the specified area are as follows.

- An access from the debugger is forcibly made to the user system.
- The EMEMEN- signal of the ROM cable is deasserted inactive (high level) in the cycle for accessing this area during execution.

[Example]

nrom 0 20000

A 20000-byte area, starting at 0h, is specified as a forced user area.

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

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

nromd command

[Format]

nromd [block-number|/all]

[Parameters]

block-number: Specifies the block number for the forced user area to be deleted.

/all: Specifies all the forced user area to be deleted.

[Function]

The nromd command deletes the forced user area specified by nrom.

[Example]

nromd 1

Block 1 is deleted from the forced user area.

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

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

outb, outh, and outw commands

[Format]

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

[Parameters]

ADDR: Specifies the address of an output port in hexadecimal.
DATA: Specifies the data to be output in hexadecimal.

[Function]

The outb, outh, and outw commands write data to the I/O space in different sizes.
The outb command accesses the I/O space in bytes, outh in half words, and outw in words.

[Examples]

outb 1000 12
Byte data 12h is written to 1000H in the I/O space.
outh 1000 1234
Half word data 1234h is written to 1000H in the I/O space.
outw 1000 12345678
Word data 12345678h is written to 1000H in the I/O space.

reset command

[Format]

reset

[Parameter]

None

[Function]

The reset command resets the CPU.

rom1..rom4 commands

[Format]

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

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

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

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

rom1: This command performs setting of a module including the EMEM board mounted to slot #3.

rom2: This command performs setting of a module including the EMEM board mounted to slot #4.

rom3: This command performs setting of a module including the EMEM board mounted to slot #5.

rom4: This command performs setting of a module including the EMEM board mounted to slot #6.

[Parameters]

ADDR [LENGTH]: Specifies an area to be emulated.

ADDR: Specifies a start address. An error occurs if the specified start address does not match the lowest address of the ROM to be emulated (boundary of the ROM).

LENGTH: Specifies the number of bytes of the ROM to be emulated. (Must be specified in boundary units of 4 bytes.)

512k|1m|2m|4m|8m|16m|32m|64m|128m|256m:
Specifies the bit size of the ROM to be emulated.
Sizes from 512K to256M bits can be specified. For the 27C1024, for example, specify 1M bit.

rom8|rom16: Specifies the number of data bits of the ROM to be emulated.
Either 8 bits or 16 bits can be specified. If a DIP-32-ROM cable is used, choose rom8; if a DIP-40/42-ROM or STD-16BIT-ROM cable is used, choose rom16.

bus8|bus16|bus32|bus64:
Specifies the ROM bus size in the system to be emulated. 8 bits, 16 bits, 32 bits, or 64 bits can be specified.
>> [bus32, bus64] are parameters for future use. (They are not used with KIT-V850E/MA3-IE.)

[[!]wren]: Write Enable: This setting is for using the emulation memory as RAM. wren enables writing, and !wren disables writing. The default value is !wren.

[Function]

The rom1 to rom4 commands set the ROM emulation environment of RTE-2000-TP. ADDR and LENGTH must be input in pairs. Input other parameters only when their values need to be changed. Parameters may be entered in any order. If the same parameter is entered twice, only the last entry is valid. The initial value of LENGTH is 0 (not used).

[Examples]

rom1 100000 40000 2m rom16 bus16 !wren

Slot position of EMEM board	Address range	Bus width	ROM		Write enable
			Bus width	Bits	
#3	100000 - 13ffff	16 bits	16 bits	2M bits	Disabled

rom2 140000 40000 2m rom16 bus16 wren

Slot position of EMEM board	Address range	Bus width	ROM		Write enable
			Bus width	Bits	
#4	140000 - 17ffff	16 bits	16 bits	2M bits	Enabled

rom1 0 80000 2m rom16 bus32 !wren

Slot position of EMEM board	Address range	Bus width	ROM		Write enable
			Bus width	Bits	
#3 + #4	000000 - 07ffff	32 bits	16 bits	2M bits	Disabled

Do not issue the rom2 command at this time.

<Remark>

Note on area specified by rom command

Access to the range specified by the rom1..rom4 commands from the debugger is a direct access to the emulation memory in the tool. As a result, display is performed correctly even if the processor cannot correctly access ROM. It is therefore recommended to read and check data by using the jread command (that reads data viathe CPU bus) or write data by setting verify to ON with the env command (download) in the initial stage of debugging.

Relationship between rom command and EMEM board

rom command	Bus width	Slot position of EMEM board	Unusable rom command
rom1	8 bits	#3	
	16 bits	#3	
	32 bits	#3 + #4	rom2
	64 bits	#3 + #4 + #5 + #6	rom2, rom3, rom4
rom2	8 bits	#4	
	16 bits	#4	
rom3	8 bits	#5	
	16 bits	#5	
	32 bits	#5 + #6	rom4
rom4	8 bits	#6	
	16 bits	#6	

seq command

[Format]

seq [PASS] [step{1|2|3|4}]

[Parameters]

PASS: Specifies in decimal the number of times the sequence condition is to be satisfied.

step{1|2|3|4}: Specifies the number of steps in the sequence.

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

[Function]

The seq command sets the sequential conditions.

Use eve, eva, and evt to specify conditions for seq1 to seq4.

When the seqclr condition is satisfied during a sequence, the sequence is executed from the beginning.

[Example]

seq 100 step1

A seq event occurs when conditions seq1 -> seq2 -> seq3 -> seq4 are satisfied 100 times.

sswon and sswoff commands

[Format]

ssw{on|off} [{exec_{[0]..[e]}|exec_default}]
[td{1|2|3|4|5|6|7|8} {none|read|write|accs|readp|writep|accsp}]
[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}]
[evep{1|2|3|4|5|6|7|8} {enable|disable}]
[ever{1|3|5|7} {enable|disable}]

[Parameters]

sswon: This command specifies a cycle in which trace data is to be loaded when the sub-switch is on.

sswoff: This command specifies a cycle in which trace data is to be loaded when the sub-switch is off.

exec_{[0]...[e]}: Specifies a cycle in which data is to be loaded as execution trace.
Each number corresponds to a cycle as follows. If trace data of some execution cycles is not loaded, disassembled trace data display may not be performed correctly.
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, e:opecode

exec_default: Loads trace data in all cycles. (Equivalent to "exec_0123456789abcd".) Usually, use this option.

td{1|2|3|4|5|6|7|8} {none|read|write|accs|readp|writep|accsp}:
Specifies the type of cycle in which trace data is to be loaded for each condition specified by the td command.

- none: Does not load trace data.
- read: Loads trace data in read cycles only.
- write: Loads trace data in write cycles only.
- accs: Loads trace data in read and write cycles.
- readp: Loads trace data in read cycles and their execution cycles.
- writep: Loads trace data in write cycles and their execution cycles.
- accsp: Loads trace data in read and write cycles, and their execution cycles.

evap{1|2|3|4|5|6} {none|read|write|accs|readp|writep|accsp}:
Specifies the type of cycle in which trace data is to be loaded for each point condition specified by the eva command.

- none: Does not load trace data.
- read: Loads trace data in read cycles only.
- write: Loads trace data in write cycles only.
- accs: Loads trace data in read and write cycles.
- readp: Loads trace data in read cycles and their execution cycles.
- writep: Loads trace data in write cycles and their execution cycles.
- accsp: Loads trace data in read and write cycles, and their execution cycles.

evar{1|3|5} {none|read|write|accs|readp|writep|accsp}:

Specifies the type of cycle in which trace data is to be loaded for each range condition specified by the eva command.

- none: Does not load trace data.
- read: Loads trace data in read cycles only.
- write: Loads trace data in write cycles only.
- accs: Loads trace data in read and write cycles.
- readp: Loads trace data in read cycles and their execution cycles.
- writep: Loads trace data in write cycles and their execution cycles.
- accsp: Loads trace data in read and write cycles, and their execution cycles.

all_cycle {none|read|write|accs|readp|writep|accsp}:

Specifies the type of cycle in which trace data is to be loaded unconditionally.

- none: Does not load trace data.
- read: Loads trace data in read cycles only.
- write: Loads trace data in write cycles only.
- accs: Loads trace data in read and write cycles.
- readp: Loads trace data in read cycles and their execution cycles.
- writep: Loads trace data in write cycles and their execution cycles.
- accsp: Loads trace data in read and write cycles, and their execution cycles.

evep{1|2|3|4|5|6|7|8} {enable|disable}:

Specifies whether to load data for each point condition specified by the eve command.

When enable is specified, data is loaded.

ever{1|3|5|7} {enable|disable}:

Specifies whether to load data for each range condition specified by the eve command.

When enable is specified, data is loaded.

[Function]

The sswon and sswoff commands specify the types of cycles in which trace data is to be loaded according to the sub-switch status.

[Example]

By default, trace data in all cycles is to beloaded when the sub-switch is on and trace data in no cycles is to be loaded when it is off.

These commands can be used to control the loading of trace data under any desired conditions.

The default settings are shown below.

```
>sswon
Sub-switch ON Settings:
  Trace execute cycle                = exec_0123456789abcd (exec_default)
  td1 Trace cycle      (td1)         = No cycle (none)
  td2 Trace cycle      (td2)         = No cycle (none)
  td3 Trace cycle      (td3)         = No cycle (none)
  td4 Trace cycle      (td4)         = No cycle (none)
  td5 Trace cycle      (td5)         = No cycle (none)
  td6 Trace cycle      (td6)         = No cycle (none)
  td7 Trace cycle      (td7)         = No cycle (none)
  td8 Trace cycle      (td8)         = No cycle (none)
  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)
  evep1 Trace          (evep1)       = Disable (disable)
  evep2 Trace          (evep2)       = Disable (disable)
  evep3 Trace          (evep3)       = Disable (disable)
  evep4 Trace          (evep4)       = Disable (disable)
  evep5 Trace          (evep5)       = Disable (disable)
  evep6 Trace          (evep6)       = Disable (disable)
  evep7 Trace          (evep7)       = Disable (disable)
  evep8 Trace          (evep8)       = Disable (disable)
  ever1 Trace          (ever1)       = Disable (disable)
  ever3 Trace          (ever3)       = Disable (disable)
  ever5 Trace          (ever5)       = Disable (disable)
  ever7 Trace          (ever7)       = Disable (disable)
```

```
>sswoff
Sub-switch OFF Settings:
  Trace execute cycle                = exec_
  td1 Trace cycle       (td1)        = No cycle (none)
  td2 Trace cycle       (td2)        = No cycle (none)
  td3 Trace cycle       (td3)        = No cycle (none)
  td4 Trace cycle       (td4)        = No cycle (none)
  td5 Trace cycle       (td5)        = No cycle (none)
  td6 Trace cycle       (td6)        = No cycle (none)
  td7 Trace cycle       (td7)        = No cycle (none)
  td8 Trace cycle       (td8)        = No cycle (none)
  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)
  evep1 Trace           (evep1)      = Disable (disable)
  evep2 Trace           (evep2)      = Disable (disable)
  evep3 Trace           (evep3)      = Disable (disable)
  evep4 Trace           (evep4)      = Disable (disable)
  evep5 Trace           (evep5)      = Disable (disable)
  evep6 Trace           (evep6)      = Disable (disable)
  evep7 Trace           (evep7)      = Disable (disable)
  evep8 Trace           (evep8)      = Disable (disable)
  ever1 Trace           (ever1)      = Disable (disable)
  ever3 Trace           (ever3)      = Disable (disable)
  ever5 Trace           (ever5)      = Disable (disable)
  ever7 Trace           (ever7)      = Disable (disable)
```

[Remark]

For the details of the sub-switch, see Appendix C, "Details of Trace Functions".

sfr command

[Format]

sfr [reg [VAL]]

[Parameters]

reg: Specifies an SFR register name.

VAL: Specifies the value for an SFR register in hexadecimal.

The following names can be used as register names:

<Registers that can be accessed by the sfr command>

SFR (R/W):

SFR (R/W):

PAL PALL PALH PAH PAHL PAHH PDL PDLL PDLH PCS PCT PCM PCD PBD PMAL
PMALL PMALH PMAH PMAHL PMAHH PMDL PMDLL PMDLH PMCS PMCT PMCM PMCD PMBD
PMCAL PMCALL PMCALH PMCAH PMCAHL PMCAHH PMCDL PMCDLL PMCDLH PMCCS
PFCCS
PMCCT PFCCT PMCCM PMCCD PMCBD CSC0 CSC1 BEC VSWC
DSA0L DSA0H DDA0L DDA0H DSA1L DSA1H DDA1L DDA1H
DSA2L DSA2H DDA2L DDA2H DSA3L DSA3H DDA3L DDA3H DBC0
DBC1 DBC2 DBC3 DADC0 DADC1 DADC2 DADC3 DCHC0
DCHC1 DCHC2 DCHC3 IMR0 IMR0L
IMR0H IMR1 IMR1L IMR1H IMR2 IMR2L IMR2H IMR3 IMR3L IMR3H WDTIC P00IC0
P00IC1 P00IC4 P00IC5 P10IC6 P10IC7 P01IC0 P01IC1 P01IC2 P01IC3 P11IC4
P11IC5 P02IC1 P02IC2 P12IC4 P12IC5 P12IC6 P13IC0 P13IC1 P13IC2 P13IC3
P13IC4 P13IC7 P05IC0 P05IC1 CMICD0 CMICD1 CMICD2 CMICD3 CM10IC0 CM10IC1
OVPIC0 OVQIC OVPIC1 OVPIC2 DMAIC0 DMAIC1 DMAIC2 DMAIC3 SEIC0 SRIC0
STIC0 SEIC1 SRIC1 STIC1 SEIC2 SRIC2 STIC2 SEIC3 SRIC3 STIC3 ADIC PSC
ADM0 ADM1 ADM2 ADTS DA0CS0 DA0CS1
DA0M P0 P1 P2 P3 P4 P5 PM0 PM1
PM2 PM3 PM4 PM5 PMC0 PMC1 PMC2
PMC3 PMC4 PMC5 PMC7 PFC0 PFC1 PFC2
PFC3 PFC4 PFC5 BCT0 BCT1 DWC0
DWC1 BCC ASC BCP LBS LBC0 LBC1 FWC FIC BMC PRC AHC SCR1
RFS1 SCR3 RFS3 SCR4 RFS4 SCR6 RFS6 CMD0
TMCD0 CMD1 TMCD1 CMD2 TMCD2 CMD3
TMCD3 TMENC10 CM100 CM101 CC100
CC101 CCR10 TUM10 TMC10 SESA10 PRM10 TQ0CTL0
TQ0CTL1 TQ0IOC0 TQ0IOC1 TQ0IOC2 TQ0OPT0 TQ0CCR0 TQ0CCR1 TQ0CCR2 TQ0CCR3
TQ0OPT1 TQ0OPT2 TQ0IOC3 TQ0DTC HZA0CTL0
HZA0CTL1 TP0CTL0 TP0CTL1 TP0IOC0 TP0IOC1 TP0IOC2 TP0OPT0 TP0CCR0 TP0CCR1
TP1CTL0 TP1CTL1 TP1IOC0 TP1IOC1 TP1IOC2 TP1OPT0 TP1CCR0 TP1CCR1
TP2CTL0 TP2CTL1 TP2IOC0 TP2IOC1 TP2IOC2 TP2OPT0 TP2CCR0 TP2CCR1
OSTS WDCS WDTM PFCE0 PFCE1 PFCE2
PFCE3 PFCE5 SYS DTFR0 DTFR1 DTFR2
DTFR3 PSMR CKC PCC WDRES CORAD0

CORAD0L CORAD0H CORAD1 CORAD1L CORAD1H CORAD2 CORAD2L CORAD2H CORAD3
CORAD3L CORAD3H CORCN DTOC DIFC
DAKW FLPMC UA0CTL0 UA0CTL1 UA0CTL2
UA0OPT0 UA0STR UA0TX UA1CTL0 UA1CTL1 UA1CTL2 UA1OPT0 UA1STR UA1TX UA2CTL0
UA2CTL1 UA2CTL2 UA2OPT0 UA2STR UA2TX UA3CTL0 UA3CTL1 UA3CTL2 UA3OPT0
UA3STR UA3TX INTF0 INTF1 INTF2
INTF3 INTF5 NMIF INTR0 INTR1 INTR2
INTR3 INTR5 NMIR CB0CTL0 CB0CTL1
CB0CTL2 CB0STR CB0TX CB0TXL CB1CTL0 CB1CTL1 CB1CTL2 CB1STR CB1TX CB1TXL
CB2CTL0 CB2CTL1 CB2CTL2 CB2STR CB2TX CB2TXL IIC
IICC SVA IICCL IICX IICF PRSM PRSCM

SFR (W):

PRCMD PFCMD

SFR (R):

ISPR ADCR0 ADCR1 ADCR2 ADCR3
ADCR4 ADCR5 ADCR6 ADCR7 ADCR0H ADCR1H ADCR2H ADCR3H ADCR4H ADCR5H
ADCR6H
ADCR7H P7 P8 TMD0 TMD1 TMD2 TMD3
STATUS10 TQ0CNT TQ0DTT1 TQ0DTT2
TQ0DTT3 TP0CNT TP1CNT TP2CNT
PFS UA0RX UA1RX UA2RX UA3RX CB0RX
CB0RXL CB1RX CB1RXL CB2RX CB2RXL

[Function]

The sfr command sets and displays the value of the SFR register.

[Examples]

sfr P0

The value of the P0 register is displayed.

sfr PM0 0

The value 0h is set in the PM0 register.

symfile and sym commands

[Format]

symfile FILENAME

sym [NAME]

[Parameters]

FILENAME: Specifies file name.

NAME: Specifies first character string in the symbols to be displayed.

[Function]

The symfile command reads symbols from the elf file specified by the FILENAME parameter.

Only global symbols can be read.

The sym command displays up to 30 symbols that have been read.

[Examples]

symfile c:\test\dry\dry.elf

Symbols are read from the elf file dry.elf in the c:\test\dry directory.

sym m

Up to 30 symbols that begin with "m" are displayed.

td1 .. td8 commands

[Format]

td{1|2|3|4|5|6|7|8} [ADDR [MASK]] [asid ASID|noasid] [/del]

[Parameters]

td{1|2|3|4|5|6|7|8}:	Input before the condition of a command from td1 to td8 is specified.
ADDR:	Specifies an address in hexadecimal.
MASK:	Specifies the mask data of an address in hexadecimal. Bits that are 1 are not subject to comparison. Only bits 9 through 2 are valid.
asid ASID|noasid:	For future expansion. Use noasid.
/del:	Clears the specified address.

[Function]

The td1.. td8 commands set the conditions of the data access cycles to be recorded by trace. These conditions can be used as trace loading conditions and triggers.

[Example]

td1 100000 ff

The access cycle of address 1000xxh is loaded to trace.

[Remark]

To display trace data in the access cycle of the area specified by td1 to td8, specify the type of access cycle in which trace data is to be output for the sswon/sswoff command.

For the details of the trace, see Appendix C, "Details of Trace Functions".

tenv command

[Format]

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}] [[!]debug]

[Parameters]

subor:	Specifies OR of the section and qualify conditions as the sub-switch.
suband:	Specifies AND of the section and qualify conditions as the sub-switch.
[!]sss_st_off:	Starts the section sub-switch in the off state. Enter ! to start the sub-switch in the on state.
[!]qss_st_off:	Starts the qualify sub-switch in the off state. Enter ! to start the sub-switch in the on state.
[!]sss_on_dly1:	Delays setting the section sub-switch to on one instruction after the condition is satisfied. Enter ! to immediately set the sub-switch to on.
[!]sss_off_dly1:	Delays setting the section sub-switch to off one instruction after the condition is satisfied. Enter ! to immediately set the sub-switch to off.
[!]add_pc:	Outputs PC for Start, Match, and Overflow. Enter ! to specify that PC is not to be output.
lvresume{0..62}:	Use the default value.
lvsuspend{1..62}:	Use the default value.
nonbranchNN:	Specifies the trace loading interval for PC information when the execution of the instructions at contiguous addresses is continued. For NN, specify a value between 0 and fff. NN = 0 means infinity. Usually, use the default value (0).
[!]phold:	Loads a packet indicating that execution is stopped during tracing in the complete (non-real-time) mode. Enter ! to load no packet.
[!]once:	Outputs trigger output once when the trigger condition is satisfied. Enter ! to output trigger output each time the trigger condition is satisfied.
tdwidth{4|8|16|24|48}:	Specifies the width of trace data. Each number corresponds to 4 bits, 8 bits, 16 bits, 24 bits, or 48 bits. For KIT-V850E/MA3-IE, use only the default value (48 bits).
tclkdiv{1|2|4}:	Specifies the frequency division rate of the trace clock to the CPU operation clock. Each number corresponds to 1/1 or 1/2. For KIT-V850E/MA3-IE, use only the default value (1/1).
[!]debug:	Use the default value (!debug).

[Function]

The tenv command sets the trace environment.

[Example]

tenv subor dmatrc

OR of the section and qualify conditions is used as the sub-switch.

[Remark]

For the details of the trace, see Appendix C, "Details of Trace Functions".

tp command

[Format]

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

[Parameters]

ADDR:	Specifies an even-numbered address in hexadecimal. (A0 is always corrected to 0.)
asid ASID|noasid:	For future expansion. Use noasid.
/del:	Clears the specified address.

[Function]

The tp command specifies a trace trigger point.

Trace is used to monitor the execution status before and after a trigger point.

[Example]

tp 100000

The execution of the instruction at 100000h is specified as a trigger point.

[Note]

If delay mode is specified with the tron command, the trigger point specification is ignored.

Delay mode can be canceled by entering tron !delay.

For the details of the trace, see Appendix C, "Details of Trace Functions".

tsp1 and tsp2 commands

[Format]

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

[Parameters]

tsp{1|2}:	Input before the condition of tsp1 or tsp2 is specified.
ADDR:	Specifies an execution address in hexadecimal.
asid ASID|noasid:	For future expansion. Use noasid.
/del:	Clears the specified address.

[Function]

The tsp1 and tsp2 commands specify the section points (addresses) of the two trace points.
The cycle in which the trace information is to be loaded can be changed by using the specified point.
(For information on how to specify the loading condition, see the description of the sswon and sswoff commands.)

[Example]

tsp1 100

The execution of the instruction at 100h is specified to section point 1.

[Remark]

For the details of the trace, see Appendix C, "Details of Trace Functions".

tmode command

[Format]

tmode

[Parameter]

None

[Function]

The tmode command displays the trace setting status.

[Example]

The default status is shown below as an example:

```
>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)
  TD1  Trigger    = Disable (!td1 )
  TD2  Trigger    = Disable (!td2 )
  TD3  Trigger    = Disable (!td3 )
  TD4  Trigger    = Disable (!td4 )
  TD5  Trigger    = Disable (!td5 )
  TD6  Trigger    = Disable (!td6 )
  TD7  Trigger    = Disable (!td7 )
  TD8  Trigger    = Disable (!td8 )
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 information                  = Disable (!add_pc)
  Non-branch      = None (nonbranch0)
  Resume Level = 0 (lvresume0)
  Suspend Level = 0 (lvsuspend0)
  PHOLD           = Disable (!phold)
  ONCE            = Disable (!once)
  TDATA Width     = 48bit (tdwidth48)
  TRCCLK Div.     = 1/1 (tclkdiv1)
  Debug Mode      = Disable (!debug)
```

```
Trace Switch Point Settings:
          Address   ASID
  tsp1 /del
  tsp2 /del
Trigger Point Settings:
       Address  ASID
  tp  /del
Data Trace Settings:
          Address  A_Mask   ASID
  td1   /del
  td2   /del
  td3   /del
  td4   /del
  td5   /del
  td6   /del
  td7   /del
  td8   /del
```

[Remark]

For the details of the trace, see Appendix C, "Details of Trace Functions".

tron command

[Format]

tron [DELAY] [[!]delay] [[!]real] [[!]force] [noext|posi|nega]
[[!]td{1|2|3|4|5|6|7|8}]

[Parameters]

DELAY = 0..3fffd delay counter	Specifies the number of frames in memory that are to be loaded in response to a trigger, in hexadecimal.
[!]delay:	Specifies forced delay mode. Enter !delay to return to normal mode. In forced delay mode, trace is forcibly terminated when the number of frames specified by the delay counter are traced after trace starts. In this mode, trigger events are ignored.
[!]real:	Specifies the execution mode during trace. real specifies the real-time execution mode. The trace information may overflow in real-time execution mode. ! specifies the non-real-time execution mode. An overflow does not occur in this mode, but the execution speed drops.
[!]force:	Specifies the forced tracestart immediately after the tron command is issued as the trace start condition. Enter ! to cancel the forced start. In this case, trace is started according to the condition specified by tsp1. When forced start is specified, tsp1 and tsp2 are also valid.
noext|posi|nega:	Specifies an external input pin (EXI0) as a trigger.
noext:	Does not use EXI0 as a trigger.
posi:	Uses the rising edge of EXI0 as a trigger.
nega:	Uses the falling edge of EXI0 as a trigger.
[!]td1:	Specifies Trace Data Condition 1 (td1) as a trigger. ! clears the setting.
[!]td2:	Specifies Trace Data Condition 2 (td2) as a trigger. ! clears the setting.
[!]td3:	Specifies Trace Data Condition 3 (td3) as a trigger. ! clears the setting.
[!]td4:	Specifies Trace Data Condition 4 (td4) as a trigger. ! clears the setting.
[!]td5:	Specifies Trace Data Condition 5 (td5) as a trigger. ! clears the setting.
[!]td6:	Specifies Trace Data Condition 6 (td6) as a trigger. ! clears the setting.
[!]td7:	Specifies Trace Data Condition 7 (td7) as a trigger. ! clears the setting.
[!]td8:	Specifies Trace Data Condition 8 (td8) as a trigger. ! clears the setting.

[Function]

The tron command clears the trace buffer and the settings of trace, and begins loading trace data.

[Examples]

tron

When tron is specified using the default values, trace is forcibly started and continues until forcibly terminated. Trace data displayed after a break shows the execution status until the execution immediately before the break.

tron delay 3fffd

Trace is started in the forced delay mode (delay=on) with using the default values for other parameters. Trace data in as many cycles as specified by the delay counter value (0x3fffd) is loaded immediately after the start of execution, and trace is automatically terminated. In the forced delay mode, trigger events are ignored.

td1 3ffb800 0
sswon td1 accs
tron !delay td1 1ffff

Trace is started when the condition of td1 is satisfied as a trigger point. !delay does not need to be specified if not changed. After the trigger condition is satisfied, trace data in as many cycles as the delay counter value (0x1ffff) is loaded, and trace is automatically terminated. As the result, trace data in about 0x20000 cycles before and after the trigger point is loaded. The sswon command is required to display trace data in the cycles related to td1.

tp 1000
tron !delay 1ffff

Trace is started when the condition specified by tp is satisfied as the trigger point. !delay does not need to be specified if not changed. After the trigger condition is satisfied, trace data in as many cycles as the delay counter value (0x1ffff) is loaded, and trace is automatically terminated. As the result, trace data in about 0x20000 cycles before and after the trigger point is loaded.

tsp1 1000
tsp2 2000
tp 1800
tron !force

As the trace packet loading condition, the value specified in the sswon command is used after the condition specified by tsp1 is satisfied and the value specified in the sswoff command is used after the condition specified by tsp2 is satisfied. By default, the sswon command specifies packet loading and the sswoff command specifies the stop of loading. According to this setting, trace loading is started immediately after the execution of the instruction at address 0x1000 specified by tsp1 and is temporarily stopped at the execution of the instruction at address 0x2000 specified by tsp2. During this period, the instruction at address 0x1800 specified by tp may be executed. In this case, the execution is used as the trigger point. As many packets as the delay cycle value (default value: 0x1ffff) are traced and loading is terminated.

tsp1 /del
tsp2 /del
tron force

tsp1 and tsp2 are canceled and trace is started in the forced start mode.

[Remark]

For the details of the trace, see Appendix C, "Details of Trace Functions".

troff command

[Format]

troff

[Parameter]

None

[Function]

The troff command forcibly terminates the loading of trace data.

trace command

[Format]

trace [POS] [all|pc|data] [asm|ttag1|ttag2] [subNN]

[Parameters]

POS=±0..1ffff:	Specifies the trace display start position in hexadecimal, assuming the vicinity of a trigger cycle or the ending cycle to be 0.
all|pc|data:	Specifies the cycle in loaded trace information that is to be displayed.
all:	All cycles
pc:	Execution cycles only
data:	Data cycles only
asm|ttag1|ttag2:	Specifies the display type.
asm:	Displays assembled listing.
ttag1:	Displays assembled listing and Time Tag in absolute time format.
ttag2:	Displays assembled listing and Time Tag in relative time format.
subNN:	Number of instructions to be disassembled in succession from an information item to actually be loaded (hexadecimal). The initial value is 80h (sub80).

[Function]

The trace command displays the contents of the trace buffer.

Issuing this command during trace forcibly terminates the loading process.

[Display]

```
> trace asm -5
    Cycle      Sub  Address       Code       Instruction              EXT   Stat
  _MchkFill8+002ah:
    -000005  ----   00:00000766 fd8b       bh     _MchkFill8+001ah(00000756h)
                                                                     1111  JMPS  Bcond
    -000003  ----   00:00000756 1241       add    00000001h,r2      1111  JMPD  Bcond
    -000003  0001                00:00000758    7e02ffff           addi  ffffh,r2,r15     1111
             SUB
    -000001  ----   00:0000075c 674f0000 st.b   r12,0000h[r15]     1111  JMPS  STORE
  * -000001  ----    --:04000000 ------0b [Byte Write (TD1)]        1111  DATAW
    -000001  0002                00:00000760    700a               mov   r10,r14         1111
             SUB
    -000001  0003                00:00000762    525f               add   ffffffffh,r10   1111
             SUB
    -000001  0004                00:00000764    71e0               cmp   r0,r14 1111  SUB
    +000001  ----   00:00000766 fd8b       bh     _MchkFill8+001ah(00000756h)
                                                                     1111  JMPS  Bcond
    +000003  ----   00:00000756 1241       add    00000001h,r2      1111  JMPD  Bcond
    +000003 0001 00:00000758 7e02ffff     addi   ffffh,r2,r15      1111  SUB

> trace ttag1
    Cycle      Sub  Address       Code       Instruction              EXT   Stat
  _MchkFill8+002ah:
    -000005  ----   00:00000766 fd8b       bh     _MchkFill8+001ah(00000756h)
                                                                     1111  JMPS  Bcond
                                    time = 000,002,151,164.7uS
    -000003  ----   00:00000756 1241       add    00000001h,r2      1111  JMPD  Bcond
```

```
                                    time = 000,002,151,164.8uS
   -000003  0001                   00:00000758     7e02ffff          addi   ffffh,r2,r15      1111
            SUB
   -000001   ----   00:0000075c  674f0000  st.b    r12,0000h[r15]  1111  JMPS  STORE
                                    time = 000,002,151,164.8uS
 * -000001   ----    --:04000000  ------0b  [Byte   Write (TD1)]    1111  DATAW
   -000001  0002                   00:00000760     700a              mov    r10,r14           1111
            SUB
   -000001  0003 00:00000762 525f        add    ffffffffh,r10       1111  SUB
   -000001  0004 00:00000764 71e0        cmp    r0,r14              1111  SUB
   +000001   ----   00:00000766 fd8b        bh     _MchkFill8+001ah(00000756h)
                                                                     1111  JMPS  Bcond
                                    time = 000,002,151,164.9uS
   +000003   ----   00:00000756 1241        add    00000001h,r2    1111  JMPD  Bcond
                                    time = 000,002,151,164.9uS

> trace ttag2
    Cycle      Sub  Address      Code       Instruction               EXT    Stat
  _MchkFill8+002ah:
   -000005   ----   00:00000766 fd8b        bh     _MchkFill8+001ah(00000756h)
                                                                     1111  JMPS  Bcond
                                    time = 000,000,000,000.0uS
   -000003   ----   00:00000756 1241        add    00000001h,r2    1111  JMPD  Bcond
                                    time = 000,000,000,000.1uS
   -000003  0001                   00:00000758     7e02ffff          addi   ffffh,r2,r15      1111
            SUB
   -000001   ----   00:0000075c  674f0000  st.b    r12,0000h[r15]  1111  JMPS  STORE
                                    time = 000,000,000,000.0uS
 * -000001   ----    --:04000000  ------0b  [Byte   Write (TD1)]    1111  DATAW
   -000001  0002                   00:00000760     700a              mov    r10,r14           1111
            SUB
   -000001  0003                   00:00000762     525f              add    ffffffffh,r10     1111
            SUB
   -000001  0004                   00:00000764     71e0              cmp    r0,r14 1111  SUB
   +000001   ----   00:00000766 fd8b        bh     _MchkFill8+001ah(00000756h)
                                                                     1111  JMPS  Bcond
                                    time = 000,000,000,000.1uS
   +000003   ----   00:00000756 1241        add    00000001h,r2    1111  JMPD  Bcond
                                    time = 000,000,000,000.0uS
   +000003 0001                    00:00000758     7e02ffff          addi   ffffh,r2,r15      1111
            SUB
```

Cycle:	Relative positions in the trace buffer are displayed in hexadecimal. The vicinity of the trigger point or the trace end frame is assumed to be 0.
Sub:	Cycle numbers generated by analyzing branching and number-of-executed-instruction information.
Address:	Execution addresses or bus cycle addresses are displayed.
Code:	Instruction code or bus cycle data is displayed.
Instruction:	Instruction mnemonics or bus types are displayed.
EXT:	The states of external input pins EXI3 to EXI0 are displayed as bit strings.
Stat:	The types of trace packets on which display is based are displayed.
	TRGSTARTON START packet is generated. Sub-switch is set to ON.
	TRGSTARTOFF START packet is generated. Sub-switch is set to OFF.

MATCH	MATCH packet is generated.
OVF	Overflow occurs.
TRCEND	TRCEND packet is generated.
JMPD <>	JMPD packet is generated. (< > will be explained later.)
JMPDS <>	JMPDS packet is generated. (< > will be explained later.)
JMPS <>	JMPS packet is generated. (< > will be explained later.)
OPCODE	Op code access (execution) occurs.
DATAW	Memory write occurs (trace packet).
DATAR	Memory read occurs (trace packet).
SFRW	SFR write occurs (bus trace).
SFRR	SFR read occurs (bus trace).
SUB	Sub-cycle

"<>" indicates the following character strings. It indicates an instruction or an event that has caused branch.

NMI/INT	By occurrence of interrupt
EXP/TRAP	By occurrence of exception
RETI	By corresponding instruction
JMP	By corresponding instruction
JR	By corresponding instruction
JARL	By corresponding instruction
BcondNT	By corresponding instruction
Bcond	By corresponding instruction
CALLT	By corresponding instruction
SWITCH	By corresponding instruction
DISPOSE	By corresponding instruction
CTRET	By corresponding instruction
STORE	When WithPC is specified for data trace
LOAD	When WithPC is specified for data trace
FSTART	Forced start of trace

* mark: Trigger point (may shift slightly).
time = Displays Time Tag

Remark The Time Tag reflects a value when the CPU outputs branch information. The output of branch information has some delay from the time of actual execution, and the delay is not constant. Thus, the measurement value of the Time Tag has some error. Especially, please ignore the measurement result immediately after the execution, as it is unreliable.

[Remark]

For the details of the trace, see Appendix C, "Details of Trace Functions".

ftrace command

[Format]

ftrace statpos endpos filname [trace_options]

[Parameters]

statpos:	Start trace position to be written into a file
endpos:	End trace position to be written into a file
filname:	Input a file name
trace_options:	The following parameters are available. The meaning is the same as for the trace command. [all|pc|data] [asm|ttag1|ttag2] [subNN]

[Function]

The ftrace command writes the contents of the trace buffer into a file.

[Note]

Carefully enter the parameters for this command because the command cannot be canceled once executed. If a wide range is specified, processing takes time.

time command

[Format]

time

[Parameter]

None

[Function]

The time command displays the time as the result of execution time measurement. The execution time measurement timer is initialized each time the CPU starts execution and is keeping the time during the execution of the CPU. The measurement clock frequency i s 50 MHz and the resolution is 20 ns. The time converted to ns units is displayed.

[Remark]

The measurement time includes the overhead times (several clocks) at the start of execution and breaks.

[Example]

```
>time
 Time = 6,600 (ns) (50.000000MHz) [Counter=0000014a]                    |
                                                  |_Counter value (hexadecimal)
                                  |_Measurement clock frequency (always 50 MHz)
```

ver command

[Format]

ver

[Parameter]

None

[Function]

The ver command displays the version of KIT-V850E/MA3-IE.