TEMU

Command Line Reference

Version latest, 2026-03-19

© Terma GmbH, Germany, 2015-2023. Proprietary and intellectual rights of Terma GmbH, Germany are involved in the subject-matter of this material and all manufacturing, reproduction, use, disclosure and sales rights pertaining to such subject-matter are expressly reserved. This material is submitted for a specific purpose as agreed in writing, and the recipient by accepting this material agrees that this material will not be used, copied or reproduced in whole or in part, nor its content (or any part thereof) revealed in any manner or to any third party, except own staff, to meet the purpose for which it was submitted and subject to the terms of the written agreement.

Table of Contents

Chapter 1. CLI

This is the reference document for the command line interface and TEMU scripting language *TScript*.

TScript is a language dedicated to managing the TEMU object system. It has been designed to be ergonomically when runnning TEMU in interactive mode. As such, unlike general purpose scripting language, the *TScript* language has direct syntactic support for the TEMU object system and global visibility on all objects in the object system.

1.1. History

1.1.1. TEMU 2

TEMU 2 supported rudimentary command line processing, based on the *libedit* tokeniser and command line completion mechanism.

The command line interpreter was line or command based and did not have any support for e.g. expressions or blocks.

Scripting in *TScript* then was essentially producing a sequence of commands without the ability to branch or introduce functions or expressions.

1.1.2. TEMU 3

As time went on more power was deemed to be needed. Several standard scripting language was investigated, but they all had the same problem. That is to deal with the TEMU object system in a natural way (with autocompletion, etc), required some significant changes to the language sources.

Compared to the TEMU 2 interpreter, the following features had been identified:

- Expressions to avoid repeating values.
- Defining complex commands in the scripting language in order to e.g. simulate boot software.
- Conditional execution.
- Promotion of command failures to status codes, in order to avoid stopping batch session due to failed commands.
- Ability to invoke class specific commands.

The new *TScript* in TEMU 3 supports the use cases above, by introducing the following features:

Expressions

Full support for operator priority based expressions such as:

```
temu> 6 + 2 * 18
```

```
18
```

Ability to use property values in expressions:

```
# Use actual ROM size instead of repeating it
mem0.map addr=0x00000000 length=rom0.size iface=rom0:MemAccessIface

# Simulate boot requirement of LEON processors where
# boot software is expected to set
# the scaler reload to CPU frequency in MHz minus one
gpTimer0.scalerReload = cpu0.freq / 1000000 - 1
```

Custom Commands in TScript

Custom commands can be defined automating boot actions and other sequences:

```
defcmd boot(file: path) {
  load obj=mem0 file=${file} start-addr-var=START
  cpu0.setPC pc=${START}
}

boot file=foo.elf
```

Conditional Execution

TScript now supports if statements.

```
if cpu0.freq < 1000000 {
  raise "Processor is too slow"
} else {
  echo "Everything is great"
}
```

Converting Command Failures to Status Codes

It is possible to convert a command failure to status code using the try expression:

```
try load obj=mem0 file=nonexistingfile
0

if !try load obj=mem0 file=nonexistingfile {
  echo "Could not load file"
}
```

Command Methods

TEMU now allows the registration of commands methods, a command method is a command attached to a class.

They are invoked on objects.

```
# Run CPU pregs command
cpu0.pregs

# Run map command in memory space
mem0.map addr=0x00000000 length=0x1000 iface=ram0:MemAccessIface
```

These command methods make it easier to add class specific commands, in addition the syntax is simplified. It is not needed having to pass an explicit object parameter.

1.2. Language Overview

1.2.1. File Extensions

By convention *TScript* is written in files using the extension **`.temu.`** `TEMU also recognise.ts`, and `.tscript` but the `ts`-extension may clash with TypeScript.

1.2.2. Comments

TScript supports comments using either of the single line shell style (`\#`) or C style (`//`). The shell-style comments has been supported since TEMU 2 while the C-style comments were added in TEMU 3.

C-style comments are the preferred style since TEMU 3.

1.2.3. Commands and Command Methods

Commands are stand alone global commands. This was the only type of command in TEMU 2. TEMU 3 introduces command method, a command that is associated to a specific class and invoked on objects. The difference is that when interacting with objects using a traditional command, the object must be passed explicitly and the command must verify the type or compliance of the object to what the command expects, for a command method, this is not the case since they can only be invoked on objects that are already compliant to whatever rules are imposed on the object.

1.2.4. Meta Classes and Class Objects

In TEMU 3, *TScript* has been extended with meta classes and class objects. This primarily means that an object can be created using the class objects `new` method.

The `new` method does by default only take one argument (the object name) but classes can (in C or C++) customise the `new` method by querying for it by name from the class objects meta class and adding named required or optional arguments to the method reference.

Each registered class has a class object with the name of the class, in addition each class object is associated with a meta class, with the name `@{CLASS_NAME}`. In addition a root meta class responsible for constructing class and meta class pairs exists, this root class is named `@MetaClass`.

You should normally not be interacting with the meta classes directly, but they exist there to maintain property and methods registries for the class objects.

1.2.5. Variables

Variables are assigned as: `foo=123` and referenced as `${foo}` or `$foo`

1.2.6. Expressions

Tscript supports expressions. The operators supported depends on the type of the left and right hand sides.

Operator	Kind	Types
+	Binary	integer, float
-	Binary, Unary	integer, float
/	Binary	integer, float, paths
*	Binary	integer, float
./	Unary	paths
<<	Binary	integers
>>	Binary	integers
|	Binary	integers
&	Binary	integers
~	Unary	integers
!	Unary	integers

1.2.7. Conditional Statements

Tscript supports if-statements.

```
if expr {
  // statements
}
```

The conditional expression does not need to be parenthesised. However, the curly braces are mandatory.

1.2.8. Loops

Not yet implemented.

To anticipate the future implementation of loops, the keywords `for`, `foreach` and `while` have been reserved.

1.2.9. Commands and Functions

Commands are entities that can fail. A command halts execution of a script if running non interactivelly. Functions however are expressions and cannot fail.

Commands can be converted to expression values which will prevent non-interactive scripts from terminating by using the `try` keyword which converts the result of the command to a 1 for success and 0 for failure.

Built-in Functions

Function name	
`defined(var)`	Return 1 if var is defined. 0 if not defined.

1.2.10. Command Definitions

It is possible to define commands both using the TEMU API and by defining commands in a TScript file.

```
defcmd mycmdname(a: string, b: path) {
  echo $a
  echo $b
}

mycmdname a=hello b=file.txt
```

A command can be failed using the raise statement.

A common use of a command would be to implement boot software emulation. The following example shows how a LEON3 processor can be booted:

```
defcmd boot(app: path) {
  if try load obj=mem0 file=${app} start-addr-var=startpc {
    if defined(startpc) {
      cpu0.setPC pc=$startpc
    }
    // Scaler reload register should be freq in MHz - 1
    grTimer0.scalerReload = cpu0.freq / 1e6 - 1

    // Prepare stack and frame pointer registers
    cpu0.setReg reg="%fp" value=0x40000000 + ram0.size - 4
    cpu0.setReg reg="%sp" value=0x40000000 + ram0.size - 4 - 92
  } else {
    raise "Failed to load file"
  }
}
```

Command Argument Types

The argument list is in the form of name type pairs, the argument name comes first, followed by a colon and then the typename. A number of types are supported at the moment:

- int: integer type
- real: double precision floating point
- string: string type
- path: string type (path indicates to auto completion to complete paths)
- object: temu object type
- iface: interface reference
- prop: property reference

The arguments are available as scoped variables in the command body.

1.2.11. Exceptions

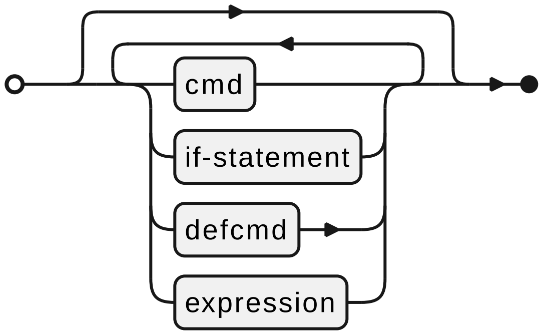
The raise statement raises an exception in a script and fails the current command. It takes a string as a parameter. This string will be printed out by the CLI as an error.

Exceptions can be caugth using the try expression. This expression converts the exception to a boolean.

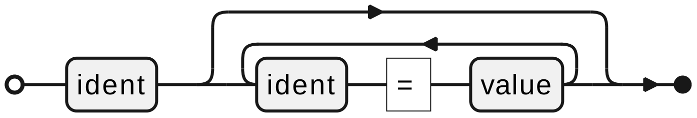
```
  if try mem0.load file=foo.elf {
    // Success
  } else {
    raise "Could not load file!"
  }
```

1.3. Grammar

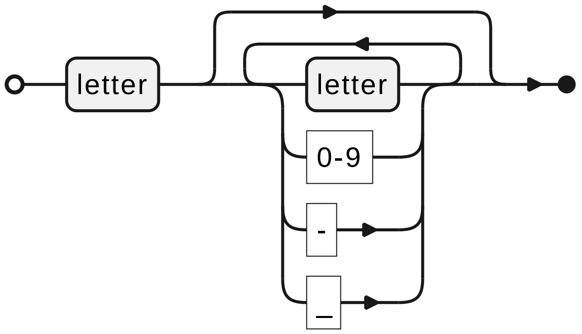
tscript



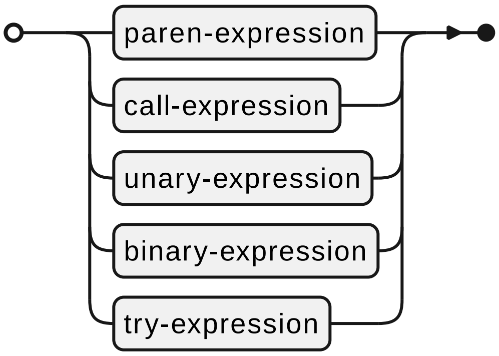
cmd



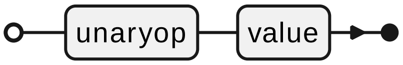
ident



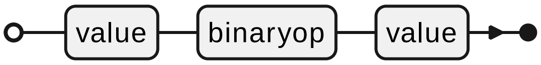
expression



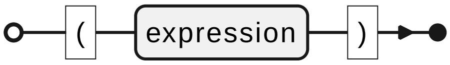
unary-expression



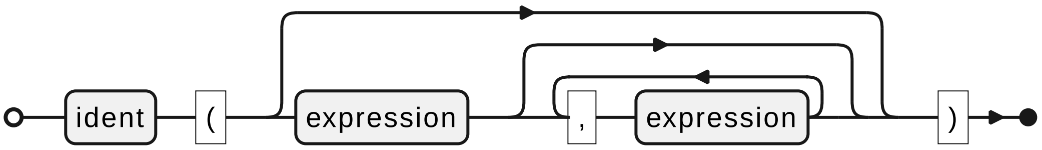
binary-expression



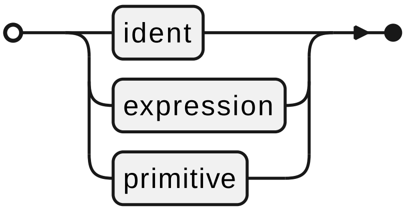
paren-expression



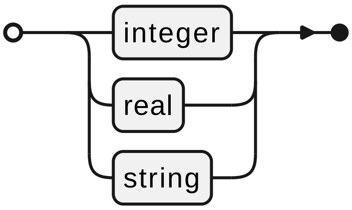
call-expression



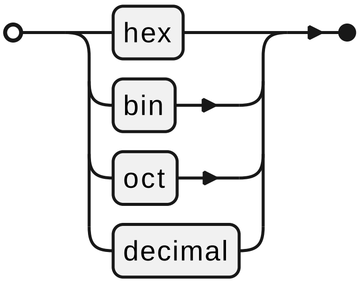
value



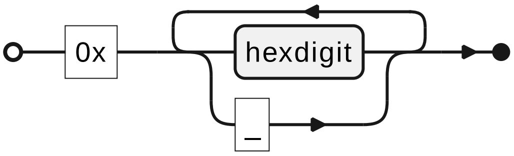
primitive



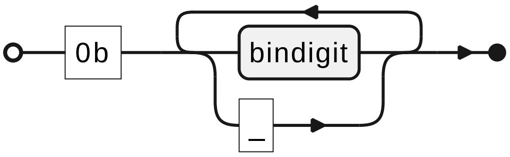
integer



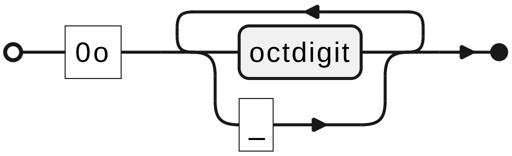
hex



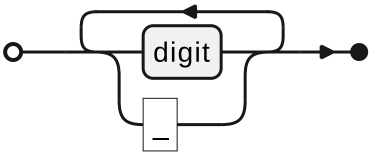
bin



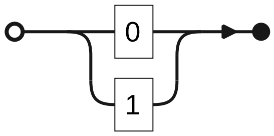
oct



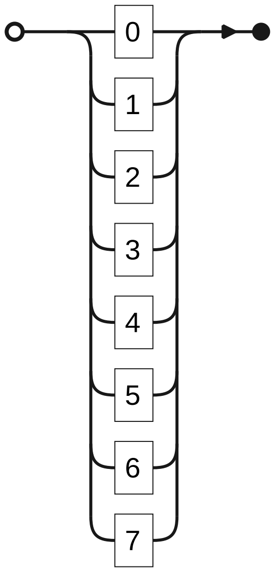
decimal



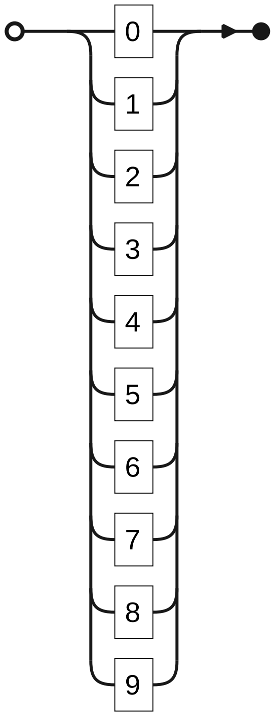
bindigit

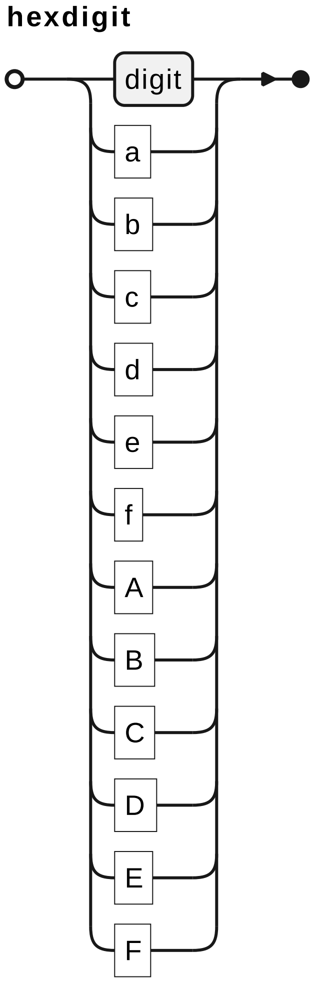


octdigit



digit





1.4. Command Reference

1.4.1. Commands

?

Alias for Section 1.4.1.237.

add-eth-delay

Add delay to eth link

Arguments

Option	Type	Description
link	object	Ethernet link where to use the delay
device	object	Device to delay
seconds	double	Delay in seconds

Aliases

eth-delay

alias

set a command alias

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
command	string	
name	string	

as

Alias for Section 1.4.1.142.

assemble

Alias for Section 1.4.1.142.

b

Alias for Section 1.4.1.21.

bp-del

Alias for Section 1.4.1.19.

bp-list

Alias for Section 1.4.1.20.

bp-set

Alias for Section 1.4.1.21.

break

Alias for Section 1.4.1.21.

break-del

Alias for Section 1.4.1.19.

break-delete

Alias for Section 1.4.1.17.

break-list

Alias for Section 1.4.1.20.

break-print

Alias for Section 1.4.1.18.

break-set

Alias for Section 1.4.1.21.

break-uart

Create breakpoint using uart logging

Arguments

Option	Type	Description
uart	object	Uart to listen
regex	string	String to find in uart logging.

Aliases

uart-break

break-uart-delete

Print all breakpoints for an object

Arguments

Option	Type	Description
uart	object	Uart to listen.
index	integer	Index of breakpoint to delete.

Aliases

break-delete

break-uart-list

Print all breakpoints for an object

Arguments

Option	Type	Description
uart	object	Break model object.

Aliases

break-print

breakpoint-delete

delete breakpoint

Arguments

Option	Type	Description
obj	object	
id	integer	
addr	integer	

Aliases

del, bp-del, break-del

breakpoint-list

list current breakpoints

ℹ This command takes no named arguments.

Aliases

bp-list, break-list

breakpoint-set

set breakpoint

Arguments

Option	Type	Description
obj	object	
addr	integer	

Aliases

break, b, bp-set, break-set

can-bus-connect

Connect a CAN device to a CAN bus

Arguments

Option	Type	Description
bus	interface	bus interface
dev	interface	device interface

Aliases

can-connect

can-bus-disconnect

Disconnect device from CAN bus

Arguments

Option	Type	Description
bus	interface	CAN bus interface
dev	interface	CAN device interface

Aliases

can-disconnect

can-connect

Alias for Section 1.4.1.22.

can-disconnect

Alias for Section 1.4.1.23.

checkpoint

Alias for Section 1.4.1.227.

checkpoint-restore

Alias for Section 1.4.1.226.

checkpoint-save

Alias for Section 1.4.1.227.

class-generate-doc

Generate documentation for class

This command does not have any aliases.

Arguments

Option	Type	Description
class	class	
file	path	
embed	integer	

class-generate-reg-images

Generate register packetdiag images for class

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
class	class	
dir	path	

class-info

Show info about class

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
class	class	

class-list

List all classes

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

clear-faulty

Alias for Section 1.4.1.143.

component-forest

Print component forrest

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

config-bool

Print configuration entry

This command does not have any aliases.

This command takes no named arguments.

config-float

Print configuration entry

This command does not have any aliases.

This command takes no named arguments.

config-int

Print configuration entry

This command does not have any aliases.

This command takes no named arguments.

config-str

Print configuration entry

This command does not have any aliases.

This command takes no named arguments.

config-uint

Print configuration entry

This command does not have any aliases.

This command takes no named arguments.

connect

Alias for Section 1.4.1.169.

connect-timesource

Alias for Section 1.4.1.179.

cov-write

Write branch counters to file

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
file	path	Path to file to write

cpu-disable-mode-switch-events

Disable generation on mode switch events.

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
cpu	object	Processor.

cpu-disable-trap-events

Disable generation on trap events.

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
cpu	object	Processor.

cpu-enable-mode-switch-events

Enable generation of mode switch events. Note that no event handler will be installed. By default the events will not be reported for performance reasons. To for example get a message for every mode switch, a mode switch event listner must also be installed by e.g. a plugin.

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
cpu	object	Processor.
print	integer	Whether to print on mode switch.

cpu-enable-trap-events

Enable generation of trap events. Note that no event handler will be installed. By default the events will not be reported for performance reasons. To for example get a message for every trap, a trap handler event listner must also be installed by e.g. a plugin.

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
cpu	object	Processor.
print	integer	Whether to print on trap.

cpu-raise-trap

Raise trap as if triggered by current instruction

Arguments

Option	Type	Description
cpu	object	Processor.
trap	integer	Trap number.

Aliases

raise-trap

cpu-reset

Resets the given CPU

Arguments

Option	Type	Description
cpu	object	Processor.
warm	integer	Set to 1 for warm reset.

Aliases

reset

cpu-set-pc

Set the program counter (and Next-PC)

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
cpu	object	Processor to set program counter in.
pc	integer	The new PC value.

cpu-set-reg

set register

Arguments

Option	Type	Description
cpu	object	Processor.
reg	string	Register name.
value	integer	Value to set.

Aliases

set-reg

cpu-show-regs

Pretty print registers.

Arguments

Option	Type	Description
cpu	object	Processor.
decimal	integer	Print in decimal

Aliases

regs

del

Alias for Section 1.4.1.19.

delwp

Alias for Section 1.4.1.262.

deserialise

Alias for Section 1.4.1.226.

deserialize

Alias for Section 1.4.1.226.

diag-err

Alias for Section 1.4.1.57.

diag-error

Emit a diagnostic error.

> ℹ This command takes no named arguments.

Aliases

diag-err, err

diag-warn

Alias for Section 1.4.1.59.

diag-warning

Emit a diagnostic warning.

> ℹ This command takes no named arguments.

Aliases

diag-warn, warn

dis

Alias for Section 1.4.1.146.

disable-binary-translation

Disable binary translation for all processors that supports it.

> ℹ This command does not have any aliases.

> ℹ This command takes no named arguments.

disassemble

Alias for Section 1.4.1.146.

echo

Write string to stdout

> ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

elf-load

Alias for Section 1.4.1.106.

enable-binary-translation

Enable binary translation for all processors supporting it.

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

env

Alias for Section 1.4.1.250.

err

Alias for Section 1.4.1.57.

eth-connect

Alias for Section 1.4.1.72.

eth-delay

Alias for Section 1.4.1.2.

eth-disconnect

Alias for Section 1.4.1.73.

eth-enable-capture

Alias for Section 1.4.1.74.

ethernet-connect

Connect phy to ethernet link

Arguments

Option	Type	Description
link	object	Ethernet link
device	interface	Ethernet device

Aliases

eth-connect

ethernet-disconnect

Disconnect phy to ethernet link

Arguments

Option	Type	Description
link	object	Ethernet link
device	interface	Ethernet device

Aliases

eth-disconnect

ethernet-enable-capture

Enable capture to file

Arguments

Option	Type	Description
link	object	Ethernet link
file	path	Path to capture file
named-pipe	integer	Create a named pipe for wireshark

Aliases

eth-enable-capture

event-list

List all events, sources and event ids

> ℹ This command does not have any aliases.

> ℹ This command takes no named arguments.

exec

Alias for Section 1.4.1.235.

exec-path-append

Append search path for the exec command

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

exec-path-list

List search paths used by the exec command

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

exec-path-prepend

Prepend search paths for the exec command

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

exit

Alias for Section 1.4.1.238.

exp-pcr

Alias for Section 1.4.1.107.

exp-pcregs

Alias for Section 1.4.1.107.

exp-pr

Alias for Section 1.4.1.108.

exp-pregs

Alias for Section 1.4.1.108.

exp-set-field

Alias for Section 1.4.1.109.

exp-write-field

Alias for Section 1.4.1.110.

experimental-debug-add-path

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
path	path	

experimental-debug-break

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
ctxt	string	Debugging context (if omitted, using current)
cpu	object	CPU (for relative breakpoints)
loc	string	Breakpoint location
addr	integer	Breakpoint address

experimental-debug-demute-break

Unmute break point, when hit a message will be printed.

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
ctxt	string	Debugging context (if omitted, using current)
id	integer	Break point ID

experimental-debug-dispose-ctxt

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
ctxt	string	Name of debugging context

experimental-debug-enable-auto-resume

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

experimental-debug-ignore-break

Set break point action to resume simulator. I.e. sim should ignore it.

This command does not have any aliases.

Arguments

Option	Type	Description
ctxt	string	Debugging context (if omitted, using current)
id	integer	Break point ID

experimental-debug-list-contexts

This command does not have any aliases.

This command takes no named arguments.

experimental-debug-list-cu

This command does not have any aliases.

Arguments

Option	Type	Description
ctxt	string	Debugging context (if omitted, using current)

experimental-debug-list-ranges

This command does not have any aliases.

Arguments

Option	Type	Description
ctxt	string	Debugging context (if omitted, using current)

experimental-debug-list-source

This command does not have any aliases.

Arguments

Option	Type	Description
cpu	object	CPU whose PC will be used to list source
addr	integer	Address to list source around
lines	integer	Lines around address (default 5)

experimental-debug-list-variables

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
ctxt	string	Debugging context (if omitted, using current)
cpu	object	Processor

experimental-debug-load-ctxt

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
mem	object	Memory space object
file	path	Path to file to load

experimental-debug-mute-break

Mute break point, when hit no message will be printed.

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
ctxt	string	Debugging context (if omitted, using current)
id	integer	Break point ID

experimental-debug-print-linenum-prog

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
ctxt	string	Debugging context (if omitted, using current)
func	string	function

experimental-debug-read-variable

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
ctxt	string	Debugging context (if omitted, using current)
cpu	object	Processor
variable	string	Variable name

experimental-debug-remap-path

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
from	string	
to	path	

experimental-debug-set-context

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
ctxt	string	Debugging context name

experimental-debug-simulate-break

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
ctxt	string	Debugging context (if omitted, using current)
cpu	object	CPU triggering the break
addr	integer	Breakpoint address

experimental-debug-stop-break

Set break point action to stop simulator.

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
ctxt	string	Debugging context (if omitted, using current)
id	integer	Break point ID

experimental-elf-load

Load ELF file for debugging new ELF loader.

Arguments

Option	Type	Description
file	path	

Aliases

elf-load

experimental-print-class-registers

EXPERIMENTAL!!! Print registers docs for class

Arguments

Option	Type	Description
class	class	

Aliases

exp-pcregs, exp-pcr

experimental-print-registers

EXPERIMENTAL!!! Print registers

Arguments

Option	Type	Description
obj	object	

Aliases

exp-pregs, exp-pr

experimental-register-set-field

EXPERIMENTAL!!! Set register field

Arguments

Option	Type	Description
reg	register	
regid	integer	
field	string	
value	integer	

Aliases

exp-set-field

experimental-register-write-field

EXPERIMENTAL!!! Write register field

Arguments

Option	Type	Description
reg	register	
field	string	
value	integer	

Aliases

exp-write-field

get-attr

Alias for Section 1.4.1.147.

graph

Alias for Section 1.4.1.171.

h

Alias for Section 1.4.1.237.

help

Alias for Section 1.4.1.237.

i

Alias for Section 1.4.1.172.

i2c-bus-connect-target

Connect a target device to a I2C bus

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
bus	interface	I2C bus interface
dev	interface	I2C target device interface
devaddr	integer	Device address/ID

i2c-bus-disconnect-target

Disconnect a target device from a I2C bus

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
bus	interface	I2C bus interface
dev	interface	I2C device interface
devaddr	integer	Device address/ID

import

Alias for Section 1.4.1.191.

import-env

Alias for Section 1.4.1.248.

import-global

Alias for Section 1.4.1.192.

info

Alias for Section 1.4.1.172.

init-scheduler

Select she scheduler to use

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
variant	string	

interrupt-lower

Lower interrupt

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
iface	interface	
irq	integer	

interrupt-raise

Raise interrupt

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
iface	interface	
irq	integer	

license

Alias for Section 1.4.1.128.

license-agreement

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

license-checkout

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
pool	string	License pool ID

license-info

Show info about your license

ℹ This command takes no named arguments.

Aliases

license

license-release

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

load

Alias for Section 1.4.1.149.

log-debug

Emit debug log message. Debug messages are only emitted in the asserts builds of TEMU.

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

log-disable-colours

Disable logging with colors to stdout/stderr

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

log-enable-colours

Enable logging with colors to stdout/stderr

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

log-error

Emit error log message

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

log-fatal

Emit fatal log message

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

log-info

Emit info log message

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

log-level

Show and set log level Log level is set by giving one of: fatal, error, warning, info, trace or debug as a positional argument

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
obj	object	object to set log level for
category	string	logging category to modify
severity	string	severity to log (debug, info, warning, error or fatal)

log-set-file

Set file for TEMU log. Setting file to '<default>' will restore default behaviour. The log file will be appended to if it already exist.

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
file	path	

log-trace

Emit trace log message

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

log-warn

Emit warning log message

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

map

Alias for Section 1.4.1.150.

memory-assemble

assemble instruction

Arguments

Option	Type	Description
cpu	object	
instr	string	
addr	integer	
count	integer	

Aliases

as, assemble

memory-clear-faulty

Clear multiple event upset / uncorrectable bit

Arguments

Option	Type	Description
obj	object	Memory space object
addr	integer	Physical address
len	integer	Length of data block to clear faulty attribute

Aliases

meu-clear, clear-faulty

memory-clear-upset

Clear single event upset bit

Arguments

Option	Type	Description
obj	object	Memory space object
addr	integer	Physical address
len	integer	Length of data block to clear SEU attribute

Aliases

seu-clear

memory-clear-user

Clear user definable attribute in memory space

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
obj	object	Memory space object
num	integer	User attribute number (1-3)
addr	integer	Physical address
len	integer	Length of data block to clear user attribute

memory-disassemble

disassemble memory range

Arguments

Option	Type	Description
cpu	object	
func	string	Function to disassemble
va	integer	
pa	integer	
addr	integer	
count	integer	number of instructions

Aliases

dis, disassemble

memory-get-attrs

Retrieves the attribute for the specified memory address

Arguments

Option	Type	Description
obj	object	Memory space object
addr	integer	Physical address

Aliases

get-attr

memory-ir-dump

Dumps IR to stdout

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
memspace	object	
addr	integer	
count	integer	
cpu	integer	

memory-load

Load file to memory. Set the address parameter to treat the file as binary, otherwise the type will be auto-detected.

Arguments

Option	Type	Description
obj	object	
file	path	
addr	integer	Address to load file to (if set, treating file as binary)
dwarf	integer	enable experimental dwarf parsing
start-addr-var	string	Variable name to set start address in.
use-pa	integer	Use physical load address instead of virtual (supported by ELF files)

Aliases

load

memory-map

Maps object in a memory space. An object can be either given with the object argument in which case the default memory access interface of MemAccessIface will be used. If 'iface' is specified instead, an explicit memory access interface can be provided. This is usefulf for devices that multiple memory ranges to map.

Arguments

Option	Type	Description
addr	integer	
length	integer	
memspace	object	
object	object	
iface	interface	
cacheable	integer	

Aliases

map

memory-read

Show the contents of the memory

Arguments

Option	Type	Description
obj	object	
addr	integer	physical address
length	integer	length in bytes
elemsize	integer	data element size (default = 0 = u8, 1 = u16, 2 = u32, 3 = u64)

Aliases

read, x

memory-set-faulty

Simulate multiple event upset / uncorrectable

Arguments

Option	Type	Description
obj	object	Memory space object
addr	integer	Physical address
len	integer	Length of data block to set faulty attribute

Aliases

meu, meu-set, set-faulty

memory-set-upset

Simulate single event upset

Arguments

Option	Type	Description
obj	object	Memory space object
addr	integer	Physical address
len	integer	Length of data block to set SEU attribute

Aliases

seu, seu-set

memory-set-user

Set user definable attribute in memory space

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
obj	object	Memory space object
num	integer	User attribute number (1-3)
addr	integer	Physical address
len	integer	Length of data block to set user attribute

memory-translate

Translate virtual to physical address

Arguments

Option	Type	Description
cpu	object	
addr	integer	

Aliases

walk

memory-write

write memory contents

Arguments

Option	Type	Description
obj	object	
addr	integer	
length	integer	
elemsize	integer	
data	integer	

Aliases

write, wr

meu

Alias for Section 1.4.1.152.

meu-clear

Alias for Section 1.4.1.143.

meu-set

Alias for Section 1.4.1.152.

mil-std-1553-connect

Connect remote terminal to 1553 bus

Arguments

Option	Type	Description
bus	object	bus object
rt	object	remote terminal object
addr	integer	rt address

Aliases

milbus-connect

mil-std-1553-disconnect

Disconnect remote terminal from 1553 bus

Arguments

Option	Type	Description
bus	object	bus object
addr	integer	rt address

Aliases

milbus-disconnect

mil-std-1553-setbc

Set the bus-controller object for the 1553 bus

Arguments

Option	Type	Description
bus	object	bus object
bc	object	bus controller object

Aliases

milbus-setbc

milbus-connect

Alias for Section 1.4.1.160.

milbus-disconnect

Alias for Section 1.4.1.161.

milbus-setbc

Alias for Section 1.4.1.162.

notification-list

List all notifications

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

object-assembly-complete

Mark the end of object assembly in a script. This will emit the temu.object-assembly-complete notification.

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

object-assembly-started

Mark the start of object assembly in a script. This will ensure temu.object-assembly-complete is published.

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

object-connect

connect two objects

Arguments

Option	Type	Description
a	property	Interface reference property ('obj.propname').
b	interface	Interface target ('obj:ifacename')

Aliases

connect

object-create

Create new object

> ℹ This command does not have any aliases.

Arguments

Option	Type	Description
class	class	
name	string	
args	string	

object-generate-graph

Generate the object graph as a dot file. If display is set to non-zero, the graph will be rendered and displayed using eog and a random file name will be generated in the current dir. NOTE: You need eog and graphviz installed to actually display the graph.

Arguments

Option	Type	Description
file	path	
display	integer	

Aliases

graph

object-info

Show object information

The use and/or disclosure,etc. of the contents of this document (or any part thereof) is subject to the restrictions referenced on the front page.

Arguments

Option	Type	Description
obj	object	
base	string	hex(default), bin, dec or oct

Aliases

info, i

object-list

List all objects

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

object-print

Pretty print object

Arguments

Option	Type	Description
obj	object	

Aliases

print, pp

object-prop-read

read property value (with side-effects from getter)

Arguments

Option	Type	Description
prop	property	
index	integer	

Aliases

read-prop

object-prop-write

write property value (with side-effects)

Arguments

Option	Type	Description
prop	property	
val	integer	
value	string	
fval	double	
index	integer	

Aliases

write-prop

object-reset

Resets the given object

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
obj	object	Object supporting the reset iface
warm	integer	Set to 1 for warm reset.

object-run

Run the machine or cpu for the given time. If time is omitted, the machine or CPU will run forever (or until a non-normal stop (halt, break, watch hit etc.))

Arguments

Option	Type	Description
obj	object	Object to run (cpu), runs scheduler otherwise.
cycles	integer	Time in cycles (only for CPUs)
time	double	Time in seconds.
perf	integer	Set to 1 to report emulator performance (MIPS-rating).
pc	integer	Program counter for start address of all CPUs.

Aliases

run

object-set-timesource

Set time source for object

Arguments

Option	Type	Description
obj	object	
ts	object	

Aliases

set-timesource, set-ts, connect-timesource

object-step

Step processor

Arguments

Option	Type	Description
obj	object	Object to step (if cpu object is to be stepped).
cpuidx	integer	CPU to step in a machine. Defaults to current CPU.
steps	integer	Number of steps, default is 1.
perf	integer	Set to 1 to report emulator performance of step operation.
pc	integer	PC to start stepping from.

Aliases

step

object-time

Get the simualted time for a machine or cpu

Arguments

Option	Type	Description
obj	object	Object to get time from.

Aliases

time

object-trace

Trace machine or processor

Arguments

Option	Type	Description
obj	object	Object to step (machine or cpu).
cpuidx	integer	CPU to step in a machine. Defaults to current CPU.
steps	integer	Number of steps, default is 1.
cpuid	integer	CPU index.
pc	integer	PC to start stepping from.
symbols	path	ELF file where to query symbols, optional
ignore	string	Comma separated symbol list to ignore traces from
call-graph	integer	Limit traces to function changes

Aliases

trace

objsys-check-sanity

Check sanity of object system

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

objsys-delete-all-objects

Erase all objects

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

objsys-delete-all-objects-and-classes

Erase all objects and classes

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

objsys-deserialise

Alias for Section 1.4.1.226.

objsys-serialise

Alias for Section 1.4.1.227.

pbanks

Print info about all register banks.

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
obj	object	Object to print bank info about.

plugin-append-path

Add plugin path

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

plugin-list

Show loaded plugins

ℹ This command does not have any aliases.

ℹ This command takes no named arguments.

plugin-load

Load plugin into temu

ℹ This command takes no named arguments.

Aliases

import

plugin-load-global

Load plugin into temu making symbols available globally

This command takes no named arguments.

Aliases

import-global

plugin-remove-path

Remove plugin path

This command does not have any aliases.

This command takes no named arguments.

plugin-show-paths

Show plugin paths

This command does not have any aliases.

This command takes no named arguments.

power-off

Powers off the model

This command does not have any aliases.

Arguments

Option	Type	Description
obj	object	Object.

power-on

Powers on the model

This command does not have any aliases.

Arguments

Option	Type	Description
obj	object	Object.

power-state

Prints the power state of the model

This command does not have any aliases.

Arguments

Option	Type	Description
obj	object	Object.

pp

Alias for Section 1.4.1.174.

pregs

Print register contents.

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
obj	object	Object to print registers for.
bank	string	Register bank, leave unset to print all of them.
reg	string	Register name to limit printout to just one register.

print

Alias for Section 1.4.1.174.

q

Alias for Section 1.4.1.238.

quit

Alias for Section 1.4.1.238.

raise-trap

Alias for Section 1.4.1.47.

read

Alias for Section 1.4.1.151.

read-prop

Alias for Section 1.4.1.175.

reg-enable

Enable register notification.

This command does not have any aliases.

Arguments

Option	Type	Description
obj	object	Object name
bank	string	Bank name
reg	string	Register name
notification	string	Notification name (pre-read, post-read, pre-write, post-write, pre-reset or post-reset).

reg-write

Write register value.

Arguments

Option	Type	Description
obj	object	Object name
bank	string	Bank name
reg	string	Register name
value	integer	Value to write.

Aliases

wreg

regs

Alias for Section 1.4.1.51.

reset

Alias for Section 1.4.1.48.

restore

Alias for Section 1.4.1.226.

run

Alias for Section 1.4.1.178.

save

Alias for Section 1.4.1.227.

script-run

Run script

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
file	path	
lang	string	
script	string	

serial-log-to-pcap

Start uart logging to pcap file

Arguments

Option	Type	Description
uart	object	Uart to listen
file	path	PCAP file for logging

Aliases

uart-log

serial-stop-logging-to-pcap

Stop uart logging to the pcap file

Arguments

Option	Type	Description
uart	object	Uart to stop pcap logging

Aliases

stop-uart-log

serialise

Alias for Section 1.4.1.227.

serialize

Alias for Section 1.4.1.227.

set

Alias for Section 1.4.1.251.

set-faulty

Alias for Section 1.4.1.152.

set-reg

Alias for Section 1.4.1.50.

set-timesource

Alias for Section 1.4.1.179.

set-ts

Alias for Section 1.4.1.179.

seu

Alias for Section 1.4.1.153.

seu-clear

Alias for Section 1.4.1.144.

seu-set

Alias for Section 1.4.1.153.

snapshot-restore

reload a snapshot from file

Arguments

Option	Type	Description
file	path	Path of snapshot to restore.
inline	integer	Alias for inplace.
inplace	integer	Restore without deleting objects, preserves pointer integrity for external code.

Aliases

deserialise, deserialize, restore, objsys-deserialise, checkpoint-restore

snapshot-save

save a snapshot to file

Arguments

Option	Type	Description
file	path	Path to snapshot file to save.

Aliases

serialise, serialize, save, checkpoint, objsys-serialise, checkpoint-save

source

Alias for Section 1.4.1.235.

spw-bus-connect

Connect two SpaceWire ports

Arguments

Option	Type	Description
port1	interface	port of the first device
port2	interface	port of the other end device

Aliases

spw-connect

spw-bus-disconnect

Disconnect two SpaceWire ports

Arguments

Option	Type	Description
port1	interface	port of the first device
port2	interface	port of the other end device

Aliases

spw-disconnect

spw-connect

Alias for Section 1.4.1.229.

spw-disconnect

Alias for Section 1.4.1.230.

step

Alias for Section 1.4.1.180.

stop-uart-log

Alias for Section 1.4.1.215.

temu-exec

exec temu command line interface script

Arguments

Option	Type	Description
file	path	

Aliases

source, exec

temu-generate-help

generate help document

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
file	path	File to write help in.

temu-help

show temu help

Arguments

Option	Type	Description
command	string	

Aliases

?, h, help

temu-quit

quit temu

This command takes no named arguments.

Aliases

quit, q, exit

temu-version

Show temu version

This command takes no named arguments.

Aliases

version

time

Alias for Section 1.4.1.181.

trace

Alias for Section 1.4.1.182.

tsnap-read

Read TEMU Snapshot File

This command does not have any aliases.

Arguments

Option	Type	Description
file	path	Path to snapshot

tsnap-write

Write TEMU Snapshot File

This command does not have any aliases.

Arguments

Option	Type	Description
file	path	Path to snapshot

uart-break

Alias for Section 1.4.1.16.

uart-log

Alias for Section 1.4.1.214.

unset

Alias for Section 1.4.1.247.

var-delete

delete variable

Arguments

Option	Type	Description
var	string	

Aliases

var-unset, unset

var-init-from-environment

Import environmental variables into the TEMU command line. If var is not given, all environment variables are imported.

Arguments

Option	Type	Description
var	string	Environment variable to import

Aliases

import-env

var-print

print variable

ℹ This command does not have any aliases.

Arguments

Option	Type	Description
var	string	

var-print-all

print all variables

This command takes no named arguments.

Aliases

vars, env

var-set

set a variable

Arguments

Option	Type	Description
var	string	
val	string	

Aliases

set

var-unset

Alias for Section 1.4.1.247.

vars

Alias for Section 1.4.1.250.

version

Alias for Section 1.4.1.239.

w

Alias for Section 1.4.1.264.

walk

Alias for Section 1.4.1.155.

warn

Alias for Section 1.4.1.59.

watch

Alias for Section 1.4.1.264.

watch-del

Alias for Section 1.4.1.262.

watch-list

Alias for Section 1.4.1.263.

watch-set

Alias for Section 1.4.1.264.

watchpoint-delete

delete watchpoint

Arguments

Option	Type	Description
id	integer	

Aliases

delwp, wp-del, watch-del

watchpoint-list

list current watchpoints

ℹ This command takes no named arguments.

Aliases

wp-list, watch-list

watchpoint-set

Set watchpoint (default is write watchpoint) use kind=(r|w|rw) to change behaviour

Arguments

Option	Type	Description
obj	object	
addr	integer	

Option	Type	Description
len	integer	
kind	string	

Aliases

watch, w, wp-set, watch-set

wp-del

Alias for Section 1.4.1.262.

wp-list

Alias for Section 1.4.1.263.

wp-set

Alias for Section 1.4.1.264.

wr

Alias for Section 1.4.1.156.

wreg

Alias for Section 1.4.1.207.

write

Alias for Section 1.4.1.156.

write-prop

Alias for Section 1.4.1.176.

x

Alias for Section 1.4.1.151.