NAME

ddb - configure DDB kernel debugger properties

SYNOPSIS

ddb capture [**-M -core**] [**-N -system**] **print**
ddb capture [**-M -core**] [**-N -system**] **status**
ddb script *scriptname*
ddb script *scriptname=script*
ddb scripts
ddb unscript *scriptname*
ddb *pathname*

DESCRIPTION

The **ddb** utility configures certain aspects of the ddb(4) kernel debugger from user space that are not configured at compile-time or easily via sysctl(8) MIB entries.

To ease configuration, commands can be put in a file which is processed using **ddb** as shown in the last synopsis line. An absolute *pathname* must be used. The file will be read line by line and applied as arguments to the **ddb** utility. Whitespace at the beginning of lines will be ignored as will lines where the first non-whitespace character is '#'.

OUTPUT CAPTURE

The **ddb** utility can be used to extract the contents of the ddb(4) output capture buffer of the current live kernel, or from the crash dump of a kernel on disk. The following debugger commands are available from the command line:

capture [**-M** *core*] [**-N** *system*] **print**
Print the current contents of the ddb(4) output capture buffer.

capture [**-M** *core*] [**-N** *system*] **status**
Print the current status of the ddb(4) output capture buffer.

SCRIPTING

The **ddb** utility can be used to configure aspects of ddb(4) scripting from user space; scripting support is described in more detail in ddb(4). Each of the debugger commands is available from the command line:

script *scriptname*
Print the script named *scriptname*.

script *scriptname=script*
Define a script named *scriptname*. As many scripts contain characters interpreted in special ways by the shell, it is advisable to enclose *script* in quotes.

scripts
List currently defined scripts.

unscript *scriptname*
Delete the script named *scriptname*.

EXIT STATUS

The **ddb** utility exits 0 on success, and >0 if an error occurs.

EXAMPLES

The following example defines a script that will execute when the kernel debugger is entered as a result of a break signal:

```
ddb script kdb.enter.break="show pcpu; bt"
```

The following example will delete the script:

```
ddb unscript kdb.enter.break
```

For further examples, see the ddb(4) and textdump(4) manual pages.

SEE ALSO

ddb(4), mac_ddb(4), textdump(4), sysctl(8)

HISTORY

The **ddb** utility first appeared in FreeBSD 7.1.

AUTHORS

Robert N M Watson

BUGS

Ideally, **ddb** would not exist, as all pertinent aspects of ddb(4) could be configured directly via sysctl(8).