2.9 DIT header

DIT is the tool used to build kernel images for download. Like the L4 kernel itself, it has an information page describing the layout of various programs and data that were part of the downloaded kernel image. It consists of two parts, the initial header followed by zero of more file headers as specified by the initial header. The initial headers format follows below.

vaddr end $_{(32)}$	+20
file end $_{(32)}$	+16
phdr num $_{(32)}$	+12
phdr size $_{(32)}$	+8
phdr off $_{(32)}$	+4
"dhdr" $_{(32)}$	+0

phdr off The offset from the beginning of this header to where the file headers start.

phdr size The size of each of the file headers.

phdr num The number of file headers that follow.

file end The offset to the end of the kernel image file. For DIT internal use only.

vaddr end The end of currently used physical memory space. This includes the L4 kernel and all other programs and data in the downloaded kernel image.

Each of the file headers is laid out as follows.

flags $_{(32)}$	+28
entry $_{(32)}$	+24
size $_{(32)}$	+20
base $_{(32)}$	+16
name string $_{(32)}$	+12
name string $_{(32)}$	+8
name string $_{(32)}$	+4
name string $_{(32)}$	+0

name string Null terminated string containing name of program or data file (truncated to 16 characters).

base The base address of the program or data file .

size The size of the program or data.

entry The start address of the program if it is executable, zero otherwise.

flags Miscellaneous flags defined below.

~ (30)	b	r

r If set the kernel runs this program as part of the initial servers upon startup. If not set, the program or data has simply been loaded into memory and has not been invoked.

b If set this program is designated as the resource manager. In this case, the initial server is given rights to creation of new tasks.

Appendix A

DITE

A.1 Name

dite — tool for creating multi-object images

A.2 Synopsis

dite [**-hVvqdl**] [[**-Rxr**] [**-b** *address*]] [**-E** *address*] *filename*] **-o** *target*

A.3 Description

DITE (*Downloadable Image Tool Extended*) is a tool for combining several object files (usually executables, but it isn't picky) into a single downloadable image. This image is usually appended to a bootloader, or downloaded directly to the target machine.

DITE currently understands ELF and raw formats, although other formats can be added with relative ease.

A.4 Options

DITE options may be either the traditional POSIX one letter options, or the GNU style long options. POSIX style options start with a single "-", while GNU long options start with "--".

A.4.1 General Options

`-h, --help` Prints out a list of options and some usage examples.

`-V, --version` Prints out the name and version.

`-v, --verbose` Increases the verbosity level. For example, displays the address and offset of images in the output image.

`-q, --quiet` Disables any non-error output.

A.4.2 Target Options

-n, --nodit Do not include a DIT header in the output image.

-t format, --target format Specifies the format of the output image. This takes the form *object-architecture-endianness*.

Note that partial target formats are allowed; the other information will be inferred from the first non-raw input file. If this option is not included, the output format is that of the first non-raw input file. For a complete list of available formats, execute `dite -h`.

-o filename, --output filename This specifies the output file.

-D segment, --dit-segment segment This specifies the segment in which the DIT header will be placed (starting from 0). The default segment is the last segment.

-O offset, --offset offset[1] This specifies the offset by which each segment will be moved. This does not effect any data in the DIT header.

-d, --dump This prints the contents of the DIT segment to stdout.

-l, --link Prints the next free address in the image.

-u, --update-sigma0[2] This updates the kernel info page in the second segment of the output image. This is used with `--raw` so that sigma0 knows where to find the DIT page.

A.4.3 Source File Options

-a pad, --align pad This pads the input image with zeroes up to the specified alignment.

-R. --raw Specifies that the file will not be included in the output image as-is.

-e address, --entry address Specifies the entry point for the image. This is only required for those input formats which do not specify this information (i.e. raw).

-b address, --base address This specifies the base of the object. This is different to `--offset` in that it *does* affect the data in the DIT header. This option is primarily used with the `--raw` option.

-x, --execute This sets the execute flag in the DIT header.

-r, --resource This sets the resource flag in the DIT header.

A.5 Notes

For output formats which contain an entry point, the entry point of the first source file is used, if the target does not exist. If the target does exist, its entry point is used.

A.6 Examples

The following are examples of the usage of `dite`.

This is an example of how to create a file called *kernel* from a file called *kernel64*.

Note the use of the `-O` flag; the PMON bootloader uses the virtual address field rather than the physical address field to load the image. The `-O` flag specifies that the image needs to be loaded at 0xffffffff80000000 plus the normal addresses.

```
dite -t elf32-mips64-big -D 2 -O 0xffffffff80000000 -o ker-
nel kernel64
```

This command takes the file *kernel* generated in the previous example and adds the file *serial* specifying it as a initial server.

```
dite -o kernel-serial -O 0x8000000 kernel -x serial
```

The following example creates an image called *kernel-serial* containing the raw file *l4pal* with an entry address of 0x200000 and a base address of 0x20000 (i.e. it constitutes 1 segment in the output file at address 0x200000), the file *sigma0* and the file *serial*, which is an initial server and the resource manager.

```
dite -R -e 0x200000 -b 0x200000 l4pal sigma0 -xr serial -o kernel-
serial
```

A.7 Bugs

The `-O` flag is an abomination.

On 32 bit architectures, any address larger than 0xffffffff may be incorrectly read (this is due to the use of *strtoul* to convert the string).

DITE will silently allow overlapping images.

A.8 Authors

Simon Winwood (`sjw@cse.unsw.edu.au`)

The latest version of dite may be found at `http://www.cse.unsw.edu.au/~sjw/code/dite/`.

A.9 See Also

`ld(1)`