There is a shell demo application that comes with MutekH.

This applications uses two MutekH features as its core:

- The in-kernel lua binding
- The libtermui terminal interface with completion

Moreover, this application has optional parts that can demonstrate other services of the kernel. Depending on your configuration, more commands may be available. We'll see the complete list of features that can be built below.

Hardware host

This application targets most of the current targets of the MutekH kernel:

- SoCLib hosts (ARM, Mips, PPC), just use it from the Arch/Soclib/Tutorial platform
- SAM7 hardware
- As an Unix process
- Natively on x86

Features

VFS access

When you use the `CONFIG_VFS` token, you'll have access to all file-related functions.

They are the usual shell commands (`ls`, `cat`, `cd`, `pwd`, `mkdir`, `rm`, `mv`, `ln`, `hexdump`), except they use a lua-ish syntax (with arguments like a function call).

Example:

```
[lua:] mkdir("foo")
[lua:] cd("foo")
[lua:foo] ls()
[lua:foo] append("file.txt", "abcd")
[lua:foo] ls()
file.txt [reg] 4
[lua:foo] cat("file.txt")
abcd[lua:foo] rm("file.txt")
[lua:foo] ls()
[lua:foo]
```

Il you compile other filesystem drivers (macro:CONFIG_DRIVER_FS_FAT16, macro:CONFIG_DRIVER_FS_ISO9660, macro:CONFIG_DRIVER_FS_DEVFS), you may even use `mount()`.

```
[lua:] mkdir("dev")
[lua:] mount("devfs", "dev")
[lua:] cd("dev/child0")
[lua:child0] ls()
cpus-Ppc,405@0 [dir] 0
cpus-Ppc,405@1 [dir] 0
cpus-Ppc,405@2 [dir] 0
cpus-Ppc,405@3 [dir] 0
tty@0 [dir] 0
block@0 [dir] 0
xicu@0-out@0 [dir] 0
xicu@0-out@1 [dir] 0
```

```
xicu@0-out@2 [dir] 0
xicu@0-out@3 [dir] 0
xicu@0 [dir] 0
memory@0 [dir] 0
[lua:child0] cd("tty@0")
[lua:tty@0] ls()
handle [reg] 0
[lua:tty@0] append("handle", "message on tty")
message on tty[lua:tty@0]
```

You may watch the internal state of the VFS with

```
[lua:] vfs_dump()
VFS dump for root 0x7f40b420, fsroot: 0x7f40b3a0, refcount: 3
 + 5 "" 0x7f40b420 (0x7f40b420), lu: 6, open: 0, close: 0, stat: 0
   + 1 "test.txt" 0x7f40cae0 (0x7f40b420), lu: 0, open: 0, close: 0, stat: 0
   + 2 "cdrom" 0x7f40be40 (0x7f40b420), lu: 0, open: 0, close: 0, stat: 0
[lua:]
```

SD/MMC driver

If you are using some hardware with a SD/MMC slot, you may use the following command:

```
[lua:] sd_mmc_rehash()
```

Cryptographic functions

MutekH contains a cryptographic API. It is enabled with the macro:CONFIG_LIBCRYPTO

The shell can be used to apply some cryptographic operations on files. The first command to test is a hash of a file. This example requires macro:CONFIG_LIBCRYPTO_MD5:

```
[lua:] md5("test.txt")
md5: 58 a8 78 59 71 21 2e f1 37 f0 4b 05 df 26 ea 15
```

Timer functions

MutekH has a built-in timer. If you enabled support for the timer (with the macro:CONFIG_MUTEK_TIMERMS token), and if you have a proper timer device setup, you may use the following commands:

```
[lua:] printlater(12000, "silly delayed message")
[lua:] cat("test.txt")
silly delayed message
test contents line 1
contents line test 2
[lua:]
```

LCD functions

Libelf / Librtld functions

Libdsrl functions