Summary

Building and running MutekH requires some development tools. Most should be available from your operating system packages. A script to easily get other specific tools installed is provided.

Required tools

Repository access

The MutekH source code is available from its mercurial repository. You will need ?mercurial (>=1.5) to fetch the source:

```
hg clone http://www.mutekh.org/hg/mutekh
```

MutekH rely on the ?Subrepository feature of mercurial. Some external projects subrepositories require ?subversion to be installed so they are fetched automatically. The hg/.hgsub file contains the external repositories list. If some source files are missing, ensure your mercurial version is not outdated or perform additional checkouts by hand.

Toolchain

Building MutekH requires the following standard software packages:

- The ?GNU Compiler Collection (gcc) (>=4.4.2).
- ?GNU make (>=3.81).
- A ?perl script interpreter (>=5.8).
- A python 2 interpreter.

Some builds may require the following additional tools, depending on the target architecture and build configuration:

- The flattened device tree compiler (dtc): This tool comes with toolchains.
- The heterogeneous linker found in hg/tools/hlink, for heterogeneous platforms only.

Target simulation tools

You may need real hardware or a simulator to run MutekH (Arch/Emu target doesn't need one though):

- ?Qemu to run MutekH, available in most GNU/Linux distributions.
- ?Bochs an other x86 emulator, available in most GNU/Linux distributions.
- ?SoCLib to experiment with various multiprocessor platforms (Mips, PowerPc, Arm, ...). A precompiled SoCLib platform is available ?here for test purpose. We suggest building your own platforms by installing SoCLib (see ?soclib:InstallationNotes).

All this simulators can be with the ?GNU debugger.

You may need extra tools to prepare bootable kernel images for some targets:

- GNU mtools or mkisofs to create a x86 bootable disk images, available in most GNU/Linux distributions.
- GNU grub or etherboot to boot compiled kernel images, included in boot image in hg/tools/ directory.

The hg/tools/x86_cdrom.sh and hg/tools/x86_floppy.sh scripts are available to easily create boot disk images.

Getting the tools

Some tools are readily available in most GNU/Linux distributions.

Other tools require being build with a specific set of options and customized for a particular target. MutekH comes with a script to build and install these tools for you.

Building tools from source

The tools/crossgen.mk script is able to download, patch, build and install required specific tools for you.

The following commands display help and default configuration:

```
 $ cd .../mutekh
 $ tools/crossgen.mk
[display some help]
 $ tools/crossgen.mk config
[display default configuration]
```

The following example shows how to get a little endian Mips cross-compiler installed under `~/mutekh`:

```
 $ tools/crossgen.mk TARGET=mipsel PREFIX=$HOME/mutekh toolchain
```

This script can install the following tools for you:

- GNU Compiler Collection (`gcc`),
- GNU Binutils,
- GNU Debugger (`gdb`),
- Device Tree Compiler (`dtc`),
- Bochs x86 emulator,
- Qemu processor emulator,
- Modified GNU coreutils timeout command, used for testsuite (`testwrap`)

Precompiled tools binaries

Some precompiled toolchains are avaialble ?here as static i386 Linux binaries for convenience and quick start purpose. It should work on any GNU/Linux i386 and x86_64 distributions. It was configured for installation in `/opt/mutekh`.

We still recommend building your own up to date tools from source if you plan to work with MutekH.