Building examples applications

MutekH comes with some examples applications available in trunk/mutekh/examples. This page briefly explains how to build these examples.

Required tools

Building MutekH requires the following standard software packages:

- A GNU compiler or cross-compiler. See below.
- GNU make (>=3.81), available in most GNU/Linux and BSD operating systems.
- A perl interpreter (>=5.8), available in most GNU/Linux and BSD operating systems.
- A python interpreter, available in most GNU/Linux and BSD operating systems.

Some builds may require the following additional tools

- The flattened device tree compiler (dtc): ?http://git.jdl.com/gitweb/ . This tool is included in precompiled toolchains.
- The heterogeneous linker found in trunk/mutekh/tools/hlink, for heterogeneous platforms only.

You may need real hardware or a simulator to run MutekH:

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

You may need extra tools to deals with 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 trunk/mutekh/tools/ directory.

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

GNU toolchain

MutekH comes with a script to build a complete cross-compilation toolchain for you. 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.

We suggest building your own toolchain if you plan to work with MutekH. The tools/crossgen.mk script is able to download, build and install toolchains for you.

There is an inline help:

```
 $ tools/crossgen.mk
[prints some help]
```

You can try a line like this one to get a Mips cross-compiler installed under `~/gnu`:

```
$ tools/crossgen.mk all TARGET=mipsel-unknown-elf PREFIX=$HOME/gnu
```

Building examples

Each example comes with its own config file which is used to configure the MutekH kernel build. This file contains application specific configuration to enable kernel features.

Some other configuration options are related to target architecture. Some ready to use configuration sets for specific targets are factored in the trunk/mutekh/examples/common directory. These configuration files are organized in sections that can be enabled from the build command line. Look at the chosen example config file to determine if it contains custom standalone configuration or if it relies on common configuration sets by including files from trunk/mutekh/examples/common.

Please refer to the BuildSystem page for in depth description of the build system.

Some working examples are listed in trunk/mutekh/examples/README file.

You are encouraged to read platform specific tutorials and subscribe to the ?mutekh-users mailing list to get help or report issues.

Using standalone and specific configuration file

Here is a make invocation for the hello example using a custom and standalone config file which targets x86 Linux process (see QuickStartUnix):

```
$ make CONF=examples/hello/config_emu
```

Relying on common configuration files

Here are make invocations for various target architectures to build examples which are using common configuration files:

- As unix user process, on x86_64 machine running Linux:

  ```
  $ make CONF=examples/hello/config BUILD=emu-linux-x86_64
  ```

- To build a x86 machine (PC) bootable kernel

  ```
  $ make CONF=examples/hello/config BUILD=ibmpc-x86
  ```

- For SoCLib simulator, Mips32 Little endian, for caba-vgmn-mutekh_soclib_tutorial or caba-vgmn-mutekh_kernel_tutorial platforms:

  ```
  $ make CONF=examples/hello/config BUILD=soclib-mips32el:pf-tutorial
  ```

- Heterogeneous builds for SoCLib simulator, Mips32 and Arm processors for caba-vgmn-mutekh_kernel_tutorial platform:

  ```
  $ make kernel-het CONF=examples/hello_het/config BUILD=pf-het EACH=soclib-mips32el:soclib
  ```