

# Installation Notes

## Prepare the environment

You'll need:

- A C++ compiler, preferably g++
- A working SystemC implementation
  - ♦ [?OSCI implementation](#)
  - ♦ SystemCass
- Binutils and BFD for your target CPU, see [Cross Compiler](#)
- A [?Subversion](#) client
- A recent [?Python](#) interpreter
- A bourne-shell compatible, like bash
- [?SDL](#) (for graphic utilities)

## Getting SoCLib

```
$ cd where/to/put/soclib
$ svn co https://www.soclib.fr/svn/trunk/soclib soclib
```

Put soclib/bin in your \$PATH, preferably add this line in your shell's startup scripts.

```
$ export PATH=$PATH:where/to/put/soclib/bin
```

## Compiling tools

Some tools need compilation before use:

```
$ cd where/to/put/soclib/utils/src
$ make
$ make install
```

## Configuration

### SystemC

You may edit [SoclibConf SoCLib's configuration file]. Out of the box, the only thing the configuration needs is setting an environment variable pointing to your SystemC implementation. Again this may preferably reside in your shell's startup scripts:

```
$ export SYSTEMC=/path/to/systemc
```

If you want to check, you should have a listing close to this one:

```
$ ls $SYSTEMC
AUTHORS  ChangeLog  LICENSE  README      docs      include
COPYING  INSTALL    NEWS     RELEASNOTES  examples  lib-linux
```

## Cross-compilation tools

By default, platform examples expect cross-toolchains compiled as described in [Cross Compiler](#). I.e. it expects `mipsel-unknown-elf-gcc`, `powerpc-unknown-elf-gcc` and `mb-gcc` (Xilinx ships a Microblaze compiler named it this way).

If you already have cross-toolchains compiled on your host, you can declare them in `~/.soclib/soft_compilers.conf`. For each architecture in `mipsel`, `powerpc` and `microblaze`, you may define:

```
<arch>_CC_PREFIX = ...
<arch>_CFLAGS = ...
<arch>_LDFLAGS = ...
```

For instance, if you want to use a mips cross-compiler configured for Linux (GNU+Glibc), you can declare:

```
mipsel_CC_PREFIX = mipsel-linux-elf-
mipsel_CFLAGS = -nostdinc
mipsel_LDFLAGS = -nostdlib
```

`nostdinc` and `nostdlib` disable default libraries (Glibc) from compilation and linking.

## Other paths

You should have cross-compilers in you path as well. For instance you should have a generic mipsel compiler toolsuite available as `mipsel-unknown-elf-*`.

If they are not in the `$PATH`, add them in:

```
$ export PATH=$PATH:/path/to/compiler/suite/bin
```

## Testing

Let's compile a simple platform:

```
$ cd /path/to/soclib/platforms/timer_4mips
$ make
[...]
$ ./system.x
```

If ever this fails, see if [SoclibConf SoCLib's configuration file] may help you.

You may even want to test more SoCLib:

```
$ cd /path/to/soclib/platforms/
$ make test
```