

Building Software via Shared Knowledge

José R. Herrero, Juan J. Navarro

Computer Architecture Department
Universitat Politècnica de Catalunya

Barcelona, Spain
{josepr,juanjo}@ac.upc.es

Goals

- Ease process of writing Makefiles

HOW?

Reusing usual rules and definitions

HOW?

Including prepared Makefiles automatically

- Ease development and build process on heterogenous environments

HOW?

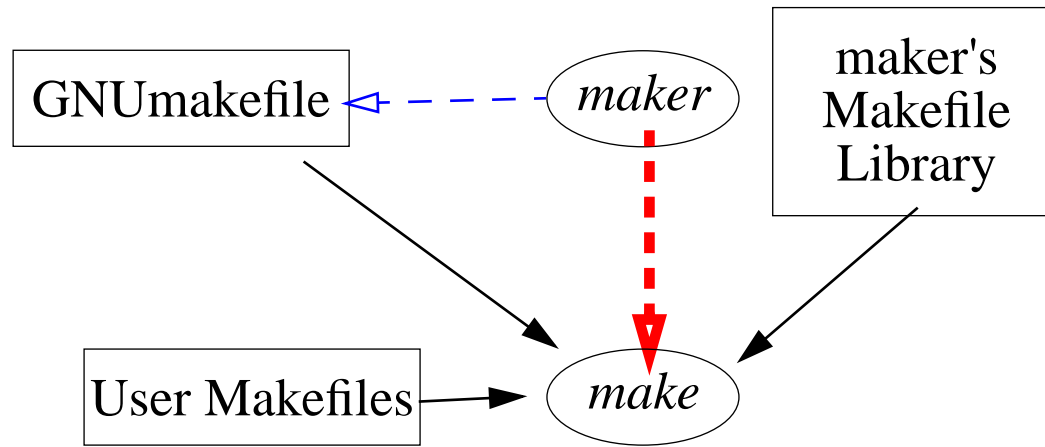
Handling platform dependant information separately

HOW?


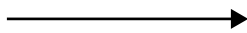

Using platform description Makefiles

Building the targets in platform specific directories (Build Tree)

System Architecture.



Legend:

-  File creation
-  Input file
-  Program call

Automatic inclusion of files

1. GNUmakefile (Preparation Makefile: Initializes PRJROOT, PRJCWD, MAKEFLAGS, MAKEFILES, VPATH)
2. Makefile.sys (System-wide configuration file: requires ARCH)
3. Makefile.prj (Project-wide configuration file)
4. Makefile.cfg (Directory-wide customization)
5. Makefile.lib (Makefile library: common targets and rules)
6. Makefile.vpath (Configuration of header file search path)
7. Pdesc.\$(ARH) (Platform description)
8. Makefile.deps (Dependency file ‘.d’ inclusion)
9. modules (Optional parts of library)
10. Makefile (Directory Makefile: targets & source file dependencies)

User files



Example: Makefiles for a simple directory structure

File simulator/Makefile

```
SUBDIRS=lib
TARGETS =

all: $(TARGETS)
```

File simulator/lib/Makefile

```
all: $(TARGETS)

$(LIBNAME).a: $(LIBNAME).a($(MODS))
```

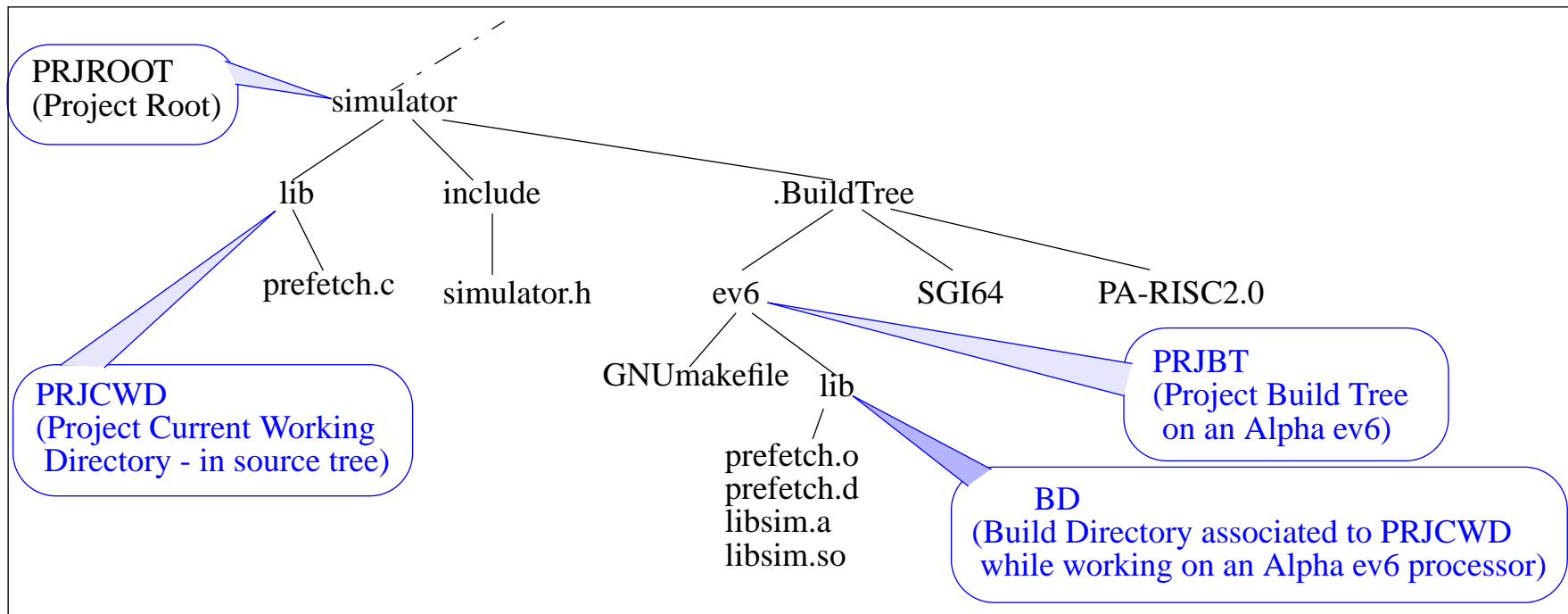
File simulator/lib/Makefile.cfg

```
LIBNAME = libsim
TARGET_LIBRARIES = $(LIBNAME).a $(LIBNAME).so
INSTALL_LIST_LIBRARIES_SYSDEP = $(TARGET_LIBRARIES)
FSRC = stride.F
CSRC = branch.c prefetch.c
MODS = $(FSRC:.F=.o)
TARGETS = $(TARGET_LIBRARIES)
```

Directory structure

```
simulator
├── Makefile
├── lib
│   ├── Makefile
│   ├── Makefile.cfg
│   ├── branch.c
│   ├── prefetch.c
│   └── stride.F
└── include
    └── simulator.h
```

Example: Directory structure



Features

- Platform specific build tree

BuildTree/\$(ARCH), --bd

- File searches

VPATH, USRINC, File interposition

- Dynamic dependency generation

CSRC, FSRC

- Recursion into subdirectories

SUBDIRS

- Debugging environment

--dbg, .gdbinit, .dbxinit

- Testing environment

xterm, PRJROOT, PATH, LD_LIBRARY_PATH

...