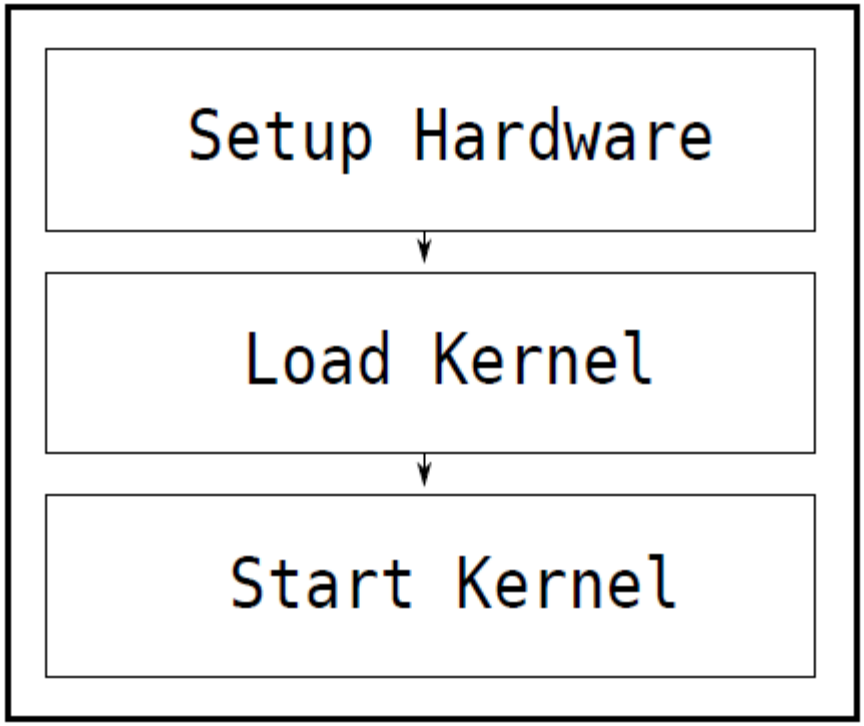
Embedded Linux

VEDA Solutions
Inspired by open Source

Boot loader Basics
Overview



Boot loader Basics
Use Cases

Development:

- Load Kernel Over Ethernet/USB

Standalone:

- Load Kernel From NAND/NOR/MMC/USB

Boot loader Basics

Example 1: Good old PC

BIOS:

- Located in ROM / NOR Flash
- Performs Hardware Setup
- Loads Bootloader from HDD

Bootloader:

- Executes from RAM
- Loads Kernel from HDD

Boot loader Basics

Example 2: Embedded NOR flash

```
[SoC + RAM + NOR Flash + I/O]
```

Bootloader:

- Located in NOR Flash
- Performs Hardware Setup
- Loads Kernel from NOR Flash

Boot loader Basics

Example 3: Embedded NOR flash + NAND/MMC

```
[SoC + RAM + NOR Flash + NAND Flash / MMC + I/O]
```

Bootloader:

- Located in NOR Flash
- Performs Hardware Setup
- Loads Kernel from NAND Flash / MMC

Boot loader Basics

Example 4: Embedded NAND Flash/MMC

```
[SoC + RAM + NAND Flash / MMC + I/O]
```

Mask ROM:

- Located inside SoC
- Loads Bootloader from NAND Flash / MMC

Bootloader:

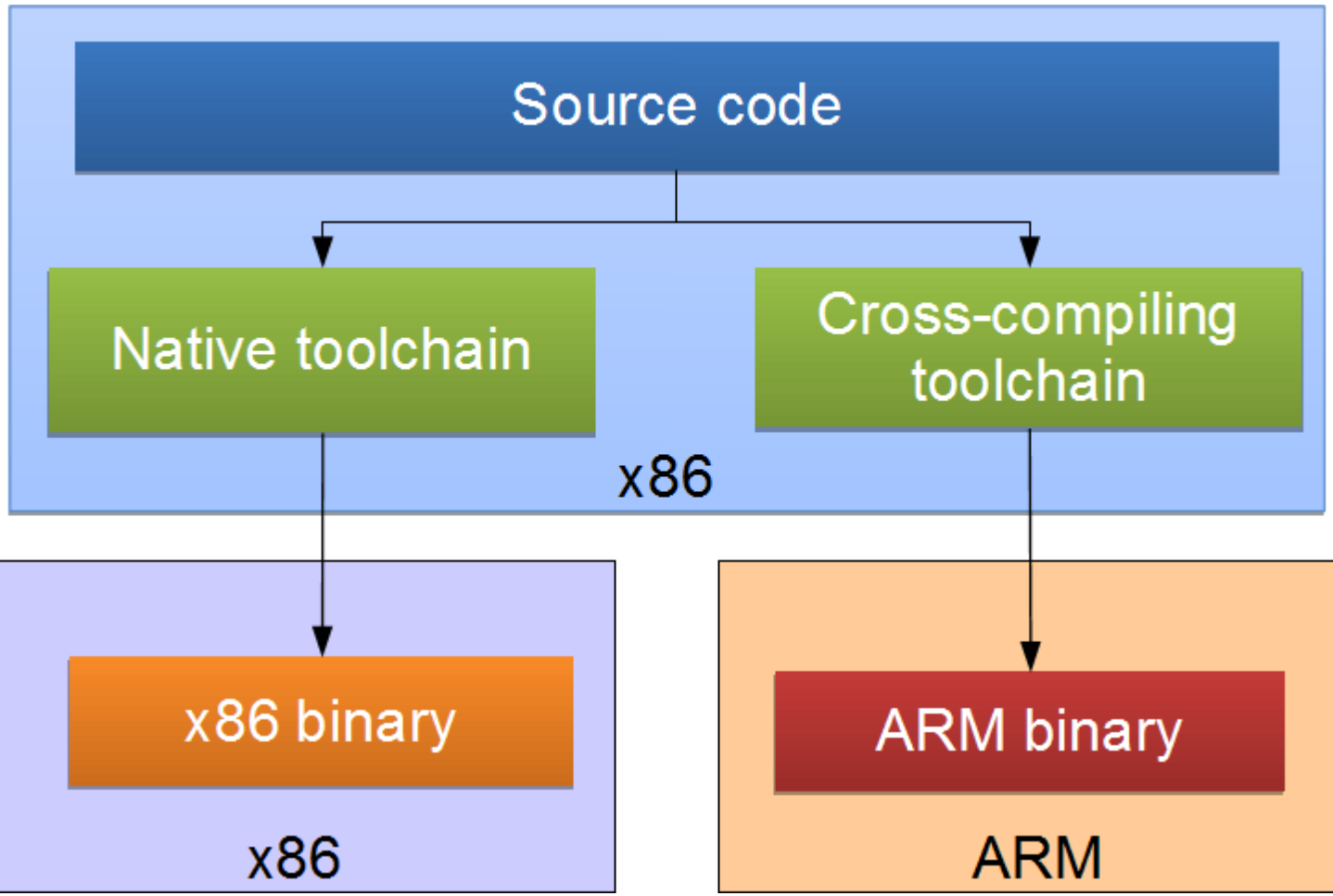
- Executes from On-chip RAM
- Performs Hardware Setup
- Loads Kernel from NAND Flash / MMC

The Four Major Elements

- Tool Chain
- Boot Loader
- Kernel
- User Space

Cross-compiling toolchains

Cross-compiling toolchains



Compilation machine

Execution machine

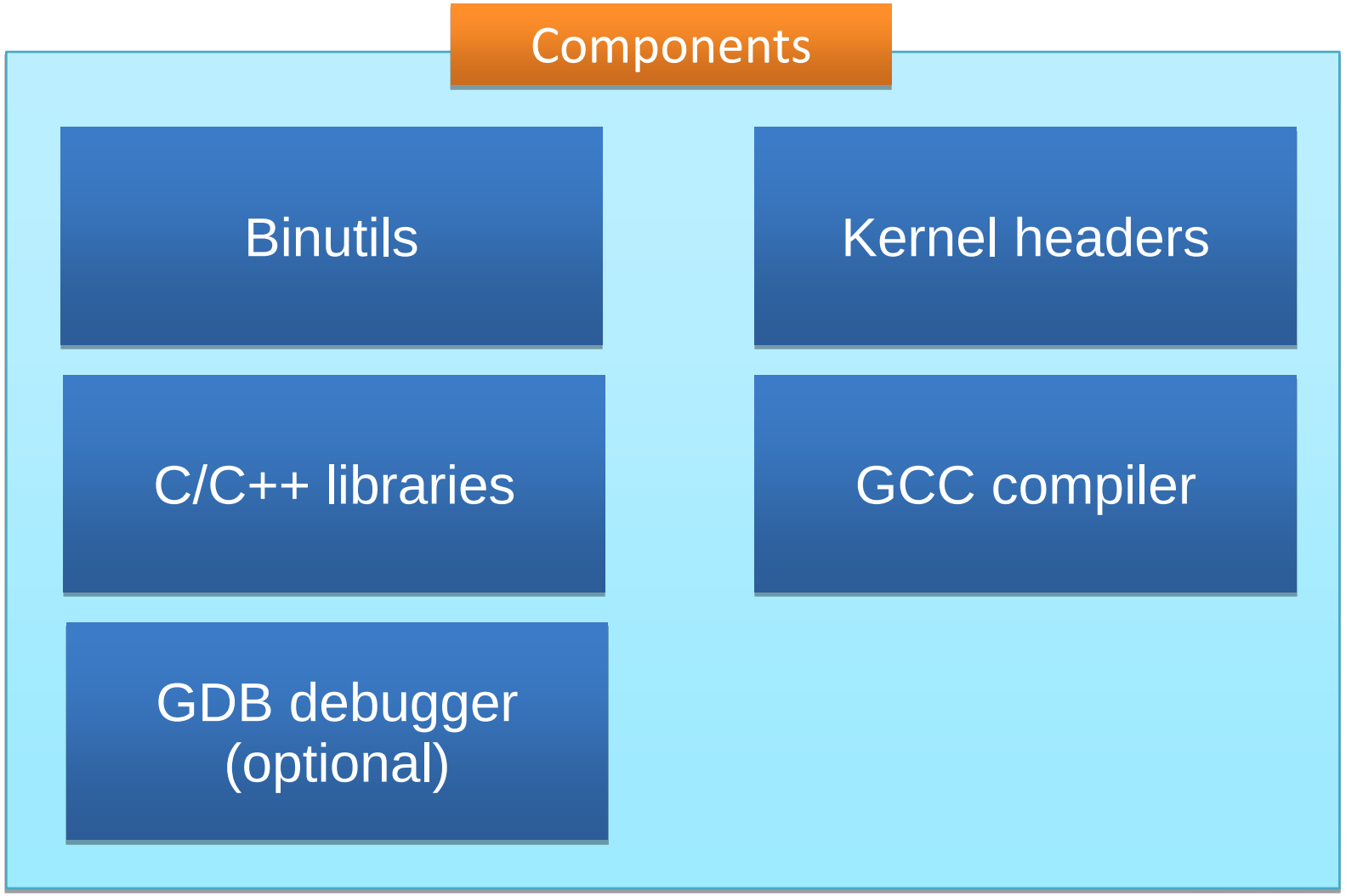
Types of Tool Chain

- Native: run compiler on target board
  - If your target board is not fast enough or doesn't have enough memory or storage, use an emulator e.g. qemu
- Cross: compile on one machine, run on another
  - Most common option

The Tool Chain

- You can't do anything until you can produce code for your platform
- A tool chain consists of at least
  - binutils: GNU assembler, linker, etc.
  - gcc: GNU C compiler
  - C library (libc): the interface to the operating system
  - gdb: debugger

Cross-compiling toolchains



GCC compiler

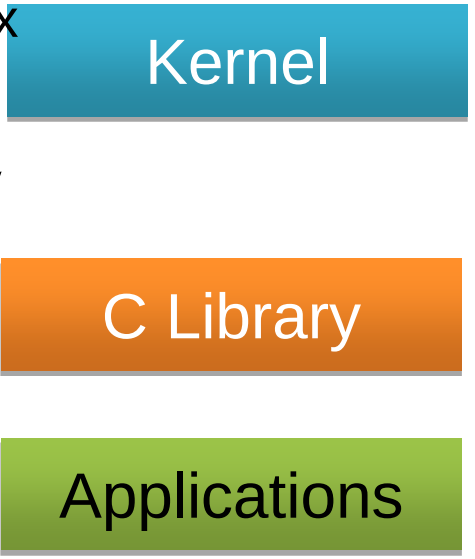
- ✓ GNU C Compiler, the famous free software compiler
- ✓ Can compile C, C++, Ada, Fortran, Java, Objective-C, Objective-C++, and generate code for a large number of CPU architectures, including ARM, AVR, Blackfin, CRIS, FRV, M32, MIPS, MN10300, PowerPC, SH, v850, i386, x86_64, IA64, Xtensa, etc.
- ✓ http://gcc.gnu.org/
- ✓ Available under the GPL license, libraries under the LGPL.

Binutils

- **Binutils** is a set of tools to generate and manipulate binaries for a given CPU architecture
- as, the assembler, that generates binary code from assembler source code
- ld, the linker
- ar, ranlib, to generate .a archives, used for libraries
- objdump, readelf, size, nm, strings, to inspect binaries. Very useful analysis tools !
- strip, to strip useless parts of binaries in order to reduce their size
- http://www.gnu.org/software/binutils/
- GPL license

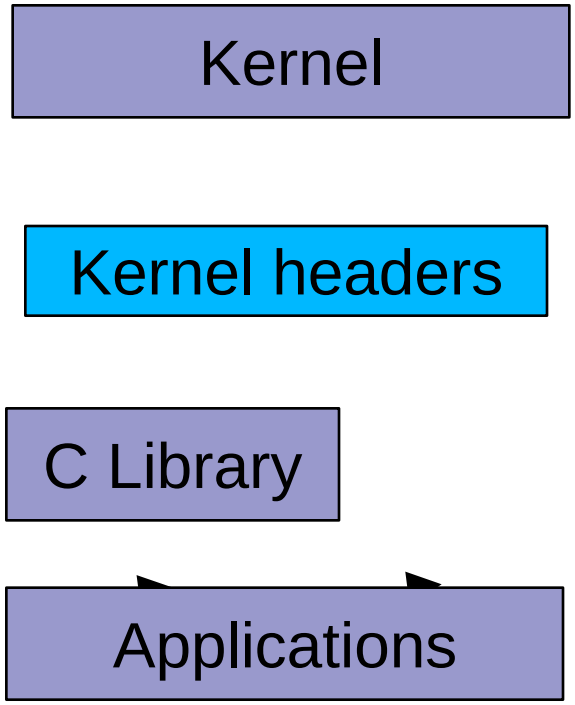
C library

- ✓ The C library is an essential component of a Linux system
- ✓ Interface between the applications and the kernel
- ✓ Provides the well-known standard C API to ease application development
- ✓ Several C libraries are available:
  glibc, uClibc, eglibc, dietlibc, newlib, etc.
- ✓ The choice of the C library must be made at the time of the cross-compiling toolchain generation, as the GCC compiler is compiled against a specific C library.



Kernel headers

- ✓ The C library and compiled programs needs to interact with the kernel
- ✓ Available system calls and their numbers
- ✓ Constant definitions
- ✓ Data structures, etc.
- ✓ Therefore, compiling the C library requires kernel headers, and many applications also require them.



Available in <linux/...> and <asm/...> and a few other directories corresponding to the ones visible in include/ in the kernel sources

Criteria for selecting a tool chain

- Good support for your processor
  - e.g. for ARM A-8 core, armv4 compilers work OK but armv7t works better
- Appropriate C library
- Up-to-date
- Good support (community or commercial)
- Other goodies, e.g.
  - Cross-compiled libraries and programs
  - Development tools for tracing, profiling, etc.

Tool Chain Examples

Free, minimal

	URL	Architectures
Codesourcery G++ Lite	www.codesourcery.com	ARM, MIPS, PPC, SH

Free, binary

	URL	Architectures
Angstrom	www.angstrom-distribution.org	ARM, PPC, AVR32, SH
Debian	www.debian.org	ARM, PPC
Ubuntu	www.ubuntu.com	ARM
Denx ELDK	www.denx.de	PPC (ARM, MIPS)

Toolchain building utilities

Another solution is to use utilities that automate the process of building the toolchain

- ✓ Same advantage as the pre-compiled toolchains: you don't need to mess up with all the details of the build process
- ✓ But also offers more flexibility in terms of toolchain configuration, component version selection, etc.
- ✓ They also usually contain several patches that fix known issues with the different components on some architectures
- ✓ Identical principle: shell scripts or Makefile that automatically fetch, extract, configure, compile and install the different components

Tool Chain Examples

Free, integrated build environment

	URL	Architectures
Buildroot	www.buildroot.org	ARM, PPC, MIPS
OpenEmbedded	www.openembedded.org	ARM, PPC, AVR32, SH
LTIB	www.bitshrine.org	ARM, PPC

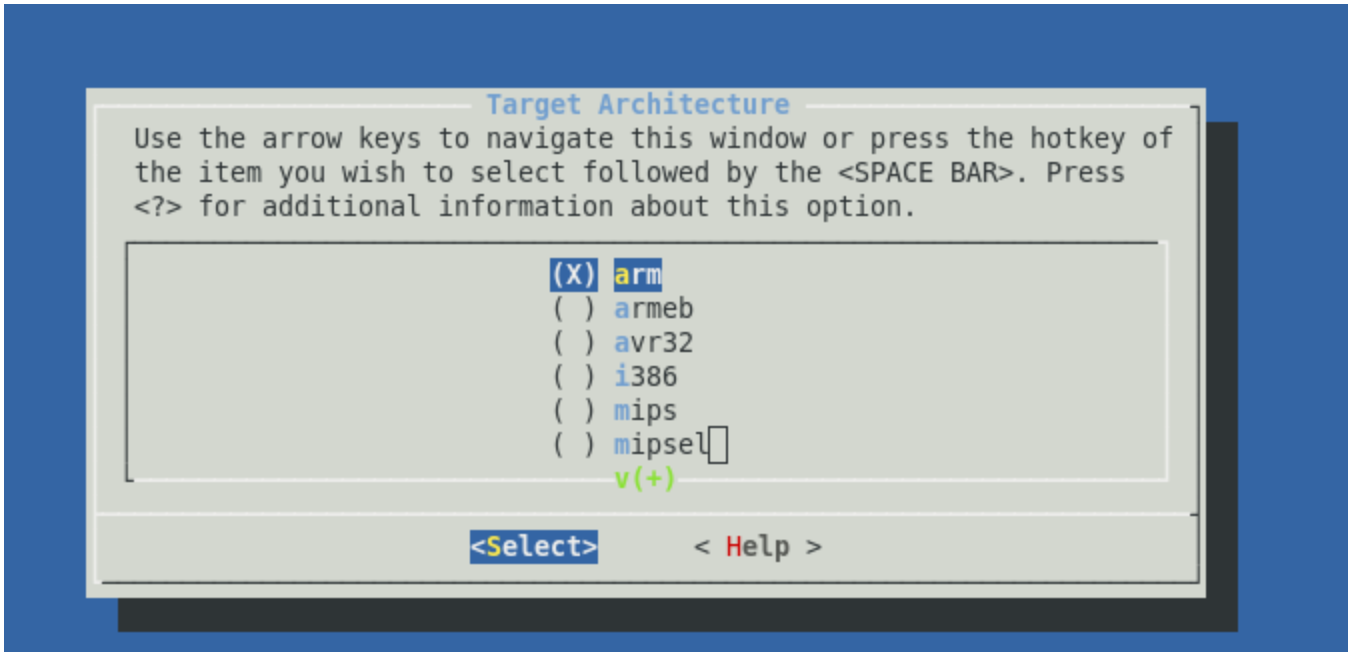
Commercial

	URL	Architectures
MontaVista Linux	www.mvista.com	
Timesys LinuxLink	linuxlink.timesys.com	
Windriver Linux	www.windriver.com	
LynuxWorks BlueCat Linux	www.lynuxworks.com	
Sysgo ElinOS	www.sysgo.com	

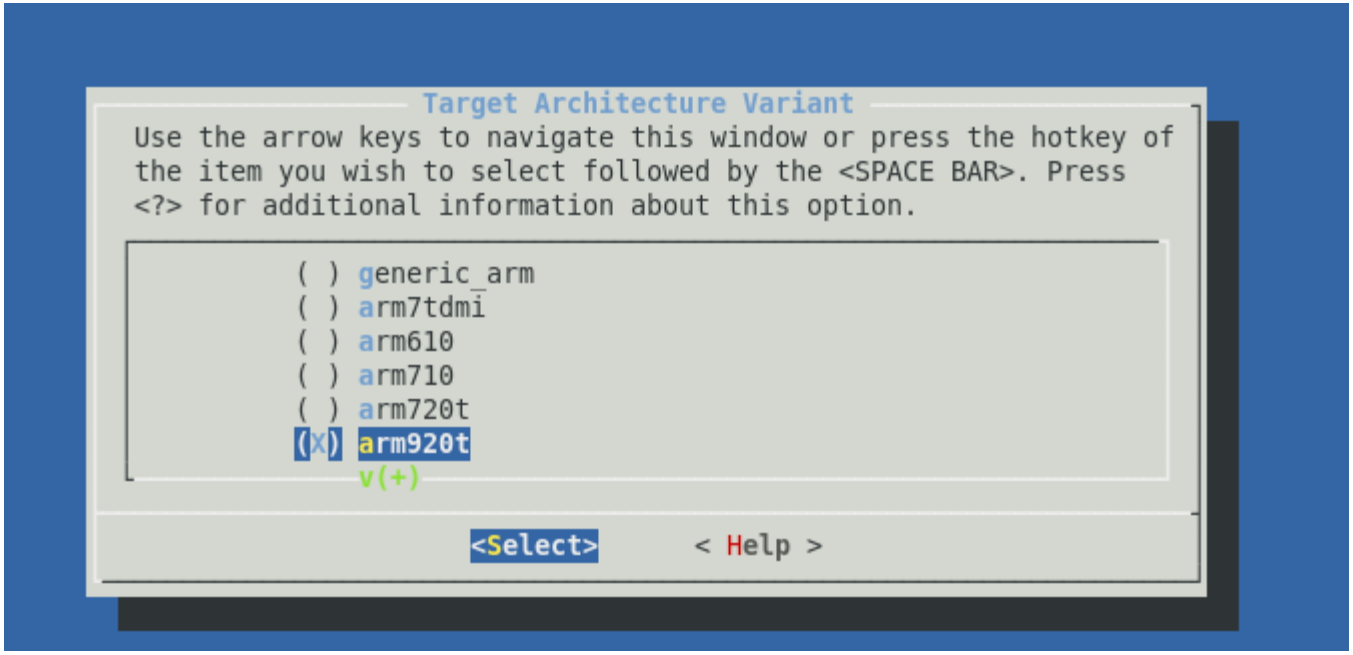
Buildroot

Steps to build cross toolchain for arm AT920T using buildroot

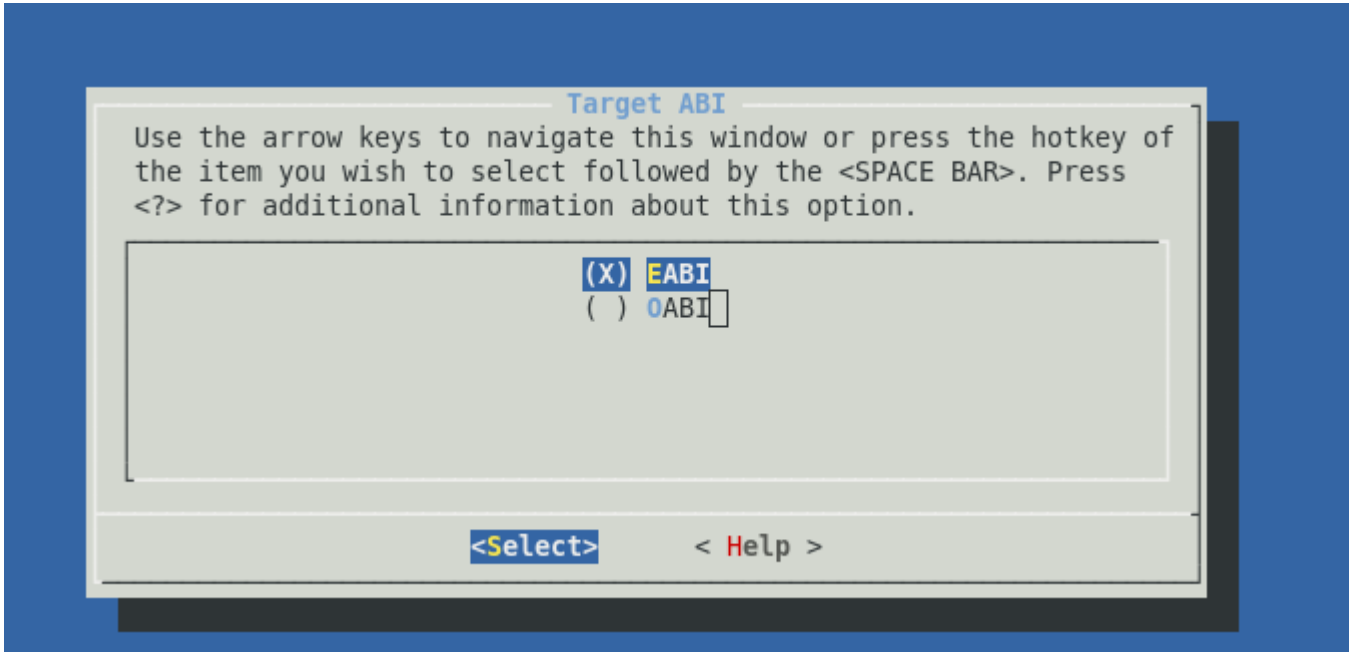
1. untar buildroot source
2. tar xvf buildroot-2010.11.tar.gz
3. make menuconfig



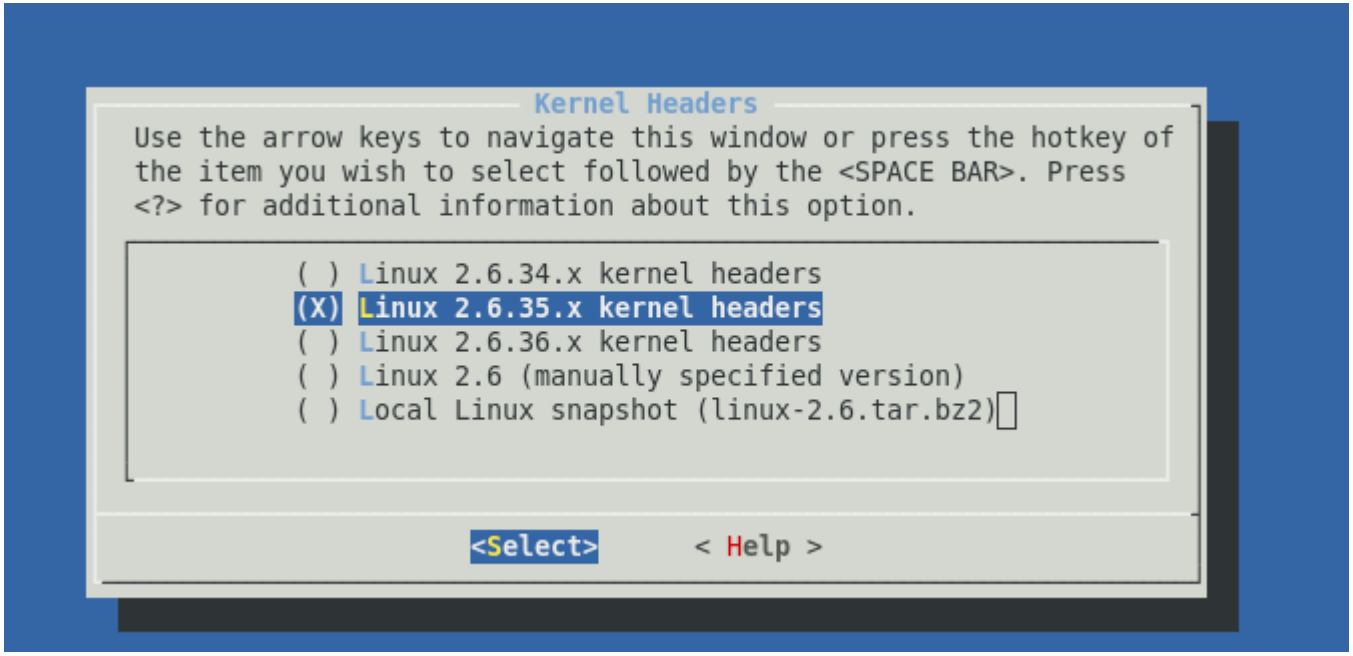
Buildroot


Target Architecture Variant
Use the arrow keys to navigate this window or press the hotkey of the item you wish to select followed by the <SPACE BAR>. Press <?> for additional information about this option.
( ) generic_arm
( ) arm7tdmi
( ) arm610
( ) arm710
( ) arm720t
(X) arm920t
v(+)
<Select>
< Help >

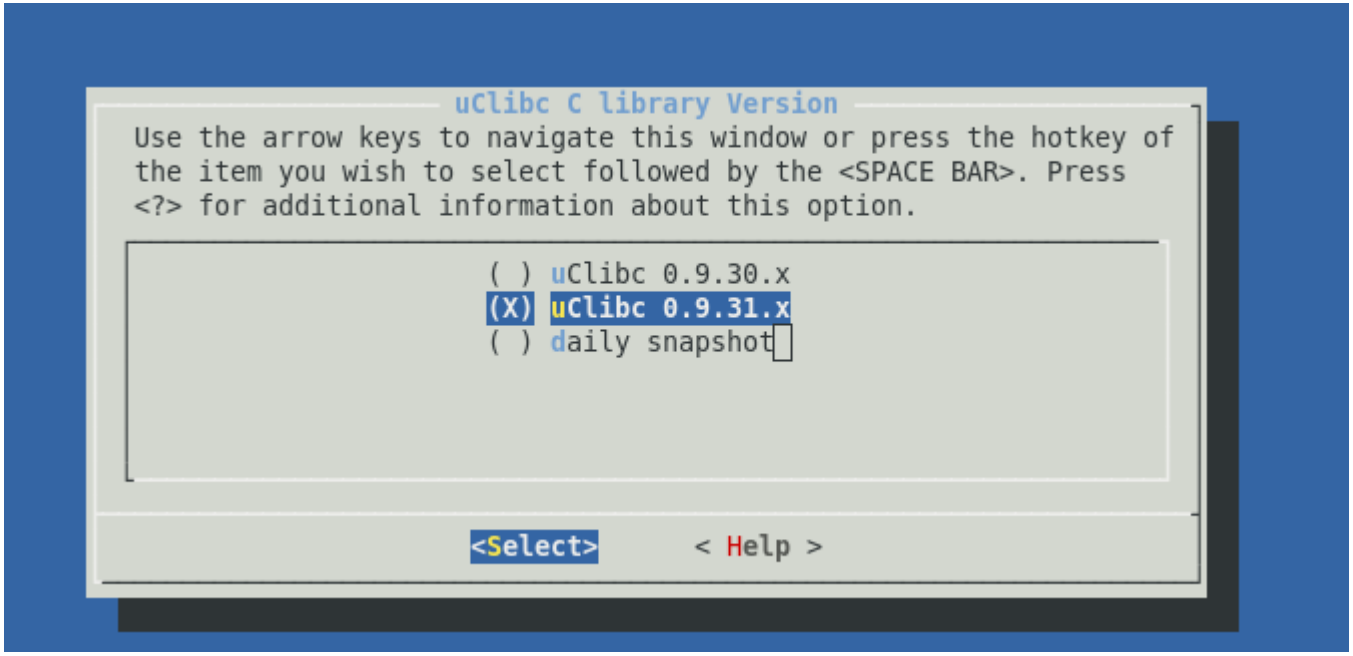
Buildroot


Target ABI
Use the arrow keys to navigate this window or press the hotkey of the item you wish to select followed by the <SPACE BAR>. Press <?> for additional information about this option.
(X) EABI
( ) OABI
<Select>
< Help >

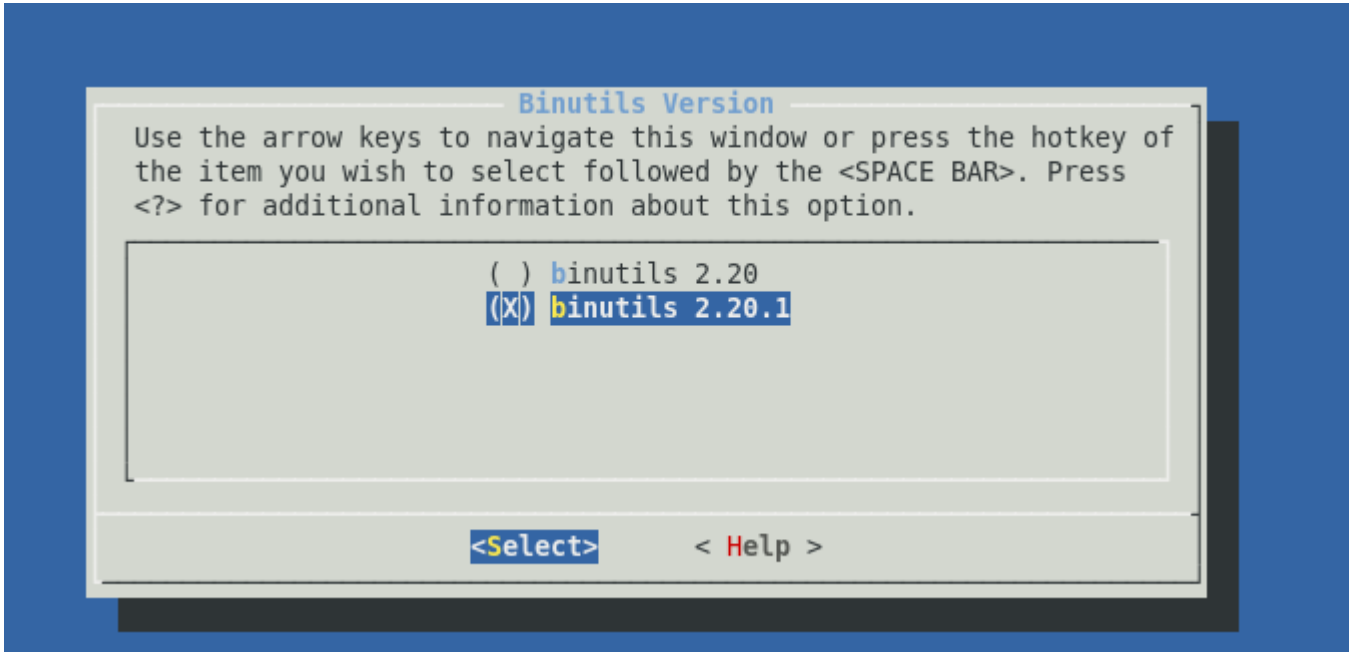
Buildroot



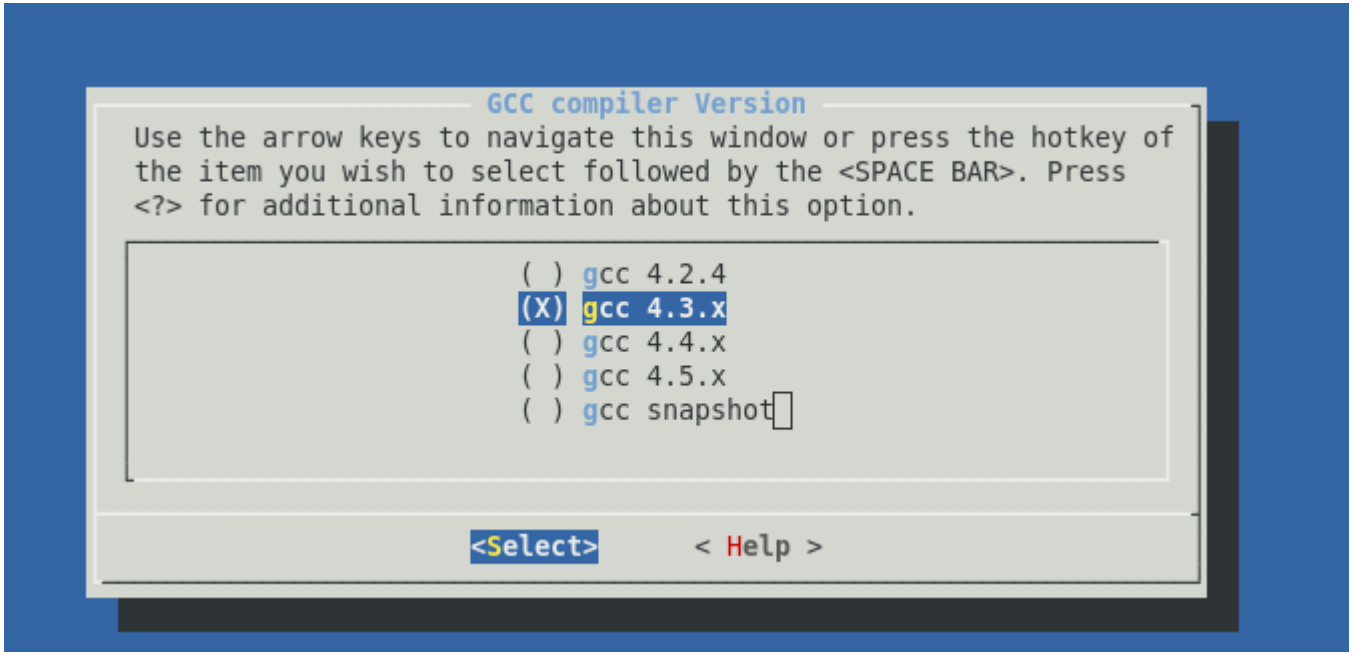
Buildroot



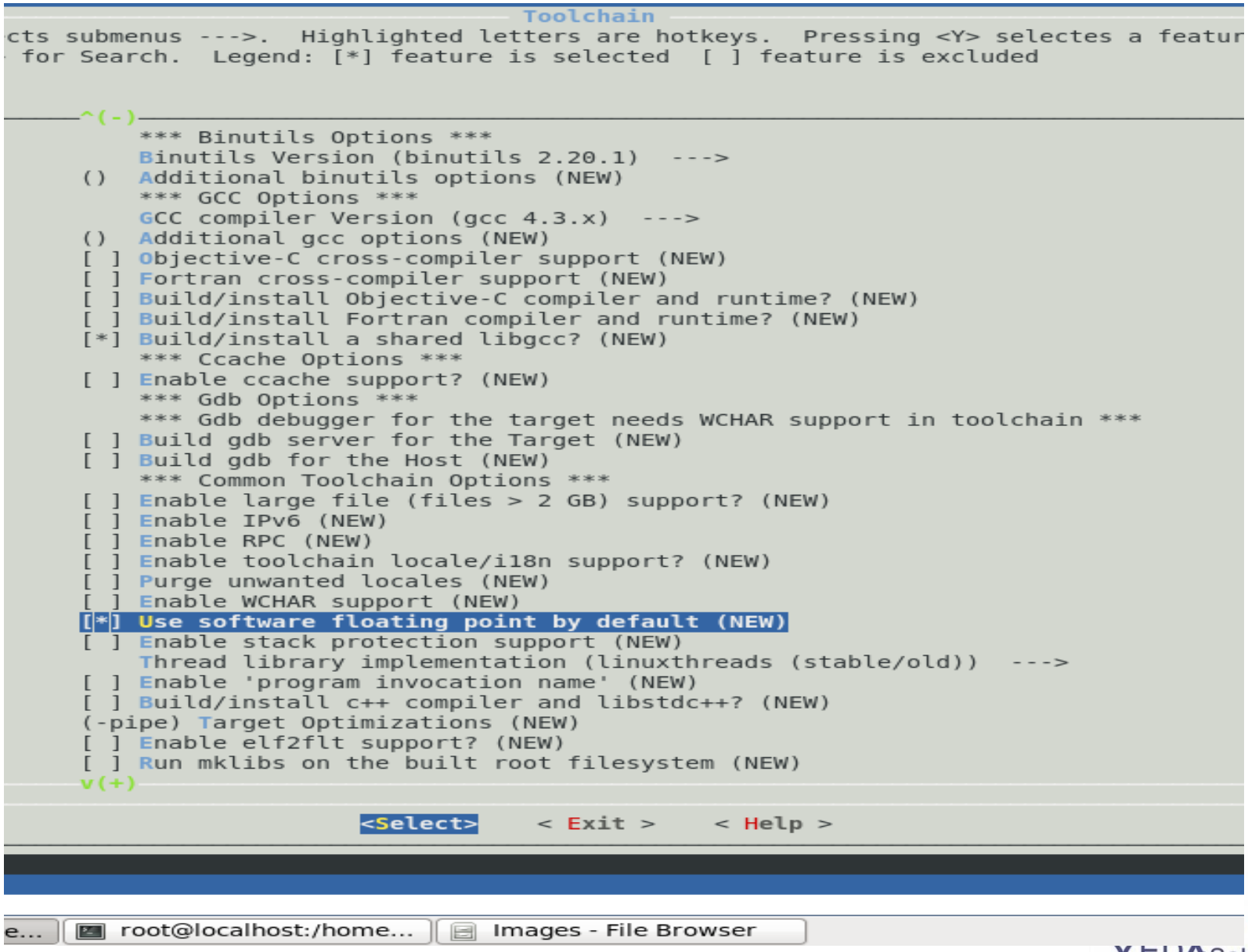
Buildroot


Binutils Version
Use the arrow keys to navigate this window or press the hotkey of the item you wish to select followed by the <SPACE BAR>. Press <?> for additional information about this option.
( ) binutils 2.20
(X) binutils 2.20.1
<Select>
< Help >

Buildroot


GCC compiler Version
Use the arrow keys to navigate this window or press the hotkey of the item you wish to select followed by the <SPACE BAR>. Press <?> for additional information about this option.
( ) gcc 4.2.4
(X) gcc 4.3.x
( ) gcc 4.4.x
( ) gcc 4.5.x
( ) gcc snapshot
<Select>
< Help >

Buildroot


Toolchain
cts submenus --->. Highlighted letters are hotkeys. Pressing <Y> selectes a featur
for Search. Legend: [*] feature is selected [ ] feature is excluded
*** Binutils Options ***
Binutils Version (binutils 2.20.1) --->
() Additional binutils options (NEW)
*** GCC Options ***
GCC compiler Version (gcc 4.3.x) --->
() Additional gcc options (NEW)
[ ] Objective-C cross-compiler support (NEW)
[ ] Fortran cross-compiler support (NEW)
[ ] Build/install Objective-C compiler and runtime? (NEW)
[ ] Build/install Fortran compiler and runtime? (NEW)
[*] Build/install a shared libgcc? (NEW)
*** Ccache Options ***
[ ] Enable ccache support? (NEW)
*** Gdb Options ***
*** Gdb debugger for the target needs WCHAR support in toolchain ***
[ ] Build gdb server for the Target (NEW)
[ ] Build gdb for the Host (NEW)
*** Common Toolchain Options ***
[ ] Enable large file (files > 2 GB) support? (NEW)
[ ] Enable IPv6 (NEW)
[ ] Enable RPC (NEW)
[ ] Enable toolchain locale/i18n support? (NEW)
[ ] Purge unwanted locales (NEW)
[ ] Enable WCHAR support (NEW)
[*] Use software floating point by default (NEW)
[ ] Enable stack protection support (NEW)
Thread library implementation (linuxthreads (stable/old)) --->
[ ] Enable 'program invocation name' (NEW)
[ ] Build/install c++ compiler and libstdc++? (NEW)
(-pipe) Target Optimizations (NEW)
[ ] Enable elf2flt support? (NEW)
[ ] Run mklibs on the built root filesystem (NEW)
<Select> < Exit > < Help >
root@localhost:/home...
Images - File Browser

make

On success, cross-tool-chain gets installed under
$<buildroot_src>/output/staging/usr/bin/ directory with a “arm-linux-” prefix

Thank you