

## Building the LINUX kernel for LEON

---

*LINUX build process overview*

*Written by Konrad Eisele, Daniel Hellstrom*

*LinuxBuild  
Version 1.0.2  
April 2012*

---

## **LINUX build process overview**

Konrad Eisele, Daniel Hellstrom

Copyright © 2012 Aeroflex Gaisler AB

# Table of Contents

1. LINUXBUILD .....	1
1.1. Introduction .....	1
1.2. Requirements .....	1
1.2.1. Installing Toolchain .....	1
1.3. Download Location .....	2
2. Installing .....	3
2.1. Buildroot distribution .....	4
2.2. Debian distribution .....	4
3. Configuring .....	6
3.1. Multiple Linux repositories .....	7
3.2. Save/Load configurations .....	8
3.3. Buildroot .....	10
3.3.1. Network Configuration .....	11
4. Building .....	12
5. Upgrading .....	13
6. Support .....	14

# 1. LINUXBUILD

## 1.1. Introduction

This document describes how the Linuxbuild utility is used to build one or more of the components listed below. The Linuxbuild utility consists of small Makefile-scripts for building and configuring some of the Linux tools that Aeroflex Gaisler provides patches for.

- Linux Kernel
- Buildroot user-space root file system
- Linux Kernel RAM image
- PROM/FLASH image

The Linuxbuild utility is a quick way of getting started with Linux development for the LEON architecture, it ties different components together to build a complete Linux environment. Each component is designed to be used separately from each other, one can see Linuxbuild as an example utility that provides a quick way of getting started with Linux development using the different tools and components. Currently the following components are supported in Linuxbuild.

- Linux Kernel + LEON Linux patches
- LEON Linux RAM loader (**mklinuximg**)
- Buildroot + LEON patches
- MKPROM2

Settings for standard LEON Linux configurations are available within the Linuxbuild package and can also be created by the user. Multiple build directories can be managed in order to test different configuration. The predefined configurations can be found in the `gaisler/configs` directory.

Note also that since each component is configured separately it is sometimes needed to set the same configuration option in multiple locations.

## 1.2. Requirements

- SPARC/LEON Linux Toolchain (Buildroot can be used to build a toolchain)
- MKPROM2 - for creating PROM/FLASH images
- wget
- git
- Internet access

Buildroot requires a number of tools such as bison, flex, msgfmt, makeinfo, etc. please see respective tool's homepage for requirements.

### 1.2.1. Installing Toolchain

Unless a custom toolchain is built with `crosstool-ng` or the Buildroot tool, the standard SPARC/LEON Linux toolchain should be installed before proceeding.

The GCC-4.4.2 multilib based toolchain is downloaded from the Aeroflex Gaisler FTP server at `ftp://ftp.gaisler.com/gaisler.com/linux/linux-2.6/toolchains/sparc-linux-4.4.2/`.

The toolchain is installed into the `/opt` directory creating the resulting directory `/opt/sparc-linux-x.y.z-toolchains/multilib`. The `bin` directory containing `sparc-linux-gcc` should be added to the `PATH` variable:

```
$ export PATH=/opt/sparc-linux-4.4.2-toolchains/multilib/bin:$PATH
$ which sparc-linux-gcc
/opt/sparc-linux-4.4.2-toolchains/multilib/bin/sparc-linux-gcc
```

Note that the toolchain path is hardcoded and cannot be installed to another directory.

---

### 1.3. Download Location

*Table 1.1. Build steps*

Tool	Download location
LINUXBUILD	<a href="ftp://gaisler.com/gaisler.com/linux/linux-2.6/linuxbuild/linuxbuild-x.y.z.tar.bz2">ftp://gaisler.com/gaisler.com/linux/linux-2.6/linuxbuild/linuxbuild-x.y.z.tar.bz2</a>
Linux toolchain	<a href="ftp://ftp.gaisler.com/gaisler.com/linux/linux-2.6/toolchains/sparc-linux-4.4.2/">ftp://ftp.gaisler.com/gaisler.com/linux/linux-2.6/toolchains/sparc-linux-4.4.2/</a>
MKROM2	<a href="ftp://ftp.gaisler.com/gaisler.com/mkprom2/linux/mkprom2-2.0.36.tar.gz">ftp://ftp.gaisler.com/gaisler.com/mkprom2/linux/mkprom2-2.0.36.tar.gz</a>

## 2. Installing

Before installing Linuxbuild the SPARC/LEON Linux toolchain must be installed, unless Buildroot is used to build a uClibc toolchain. See Section 1.2.1.

After downloading the Linuxbuild package it is extracted using **tar -xf**, and the components are installed the first time by using the "upgrade" functionality of the KConfig GUI:

```
$ tar -xf linuxbuild-x.y.z.tar.bz2
$ cd linuxbuild-x.y.z
$ make xconfig
... do SELECT/INSTALL/UPGRADE PACKAGES...
```

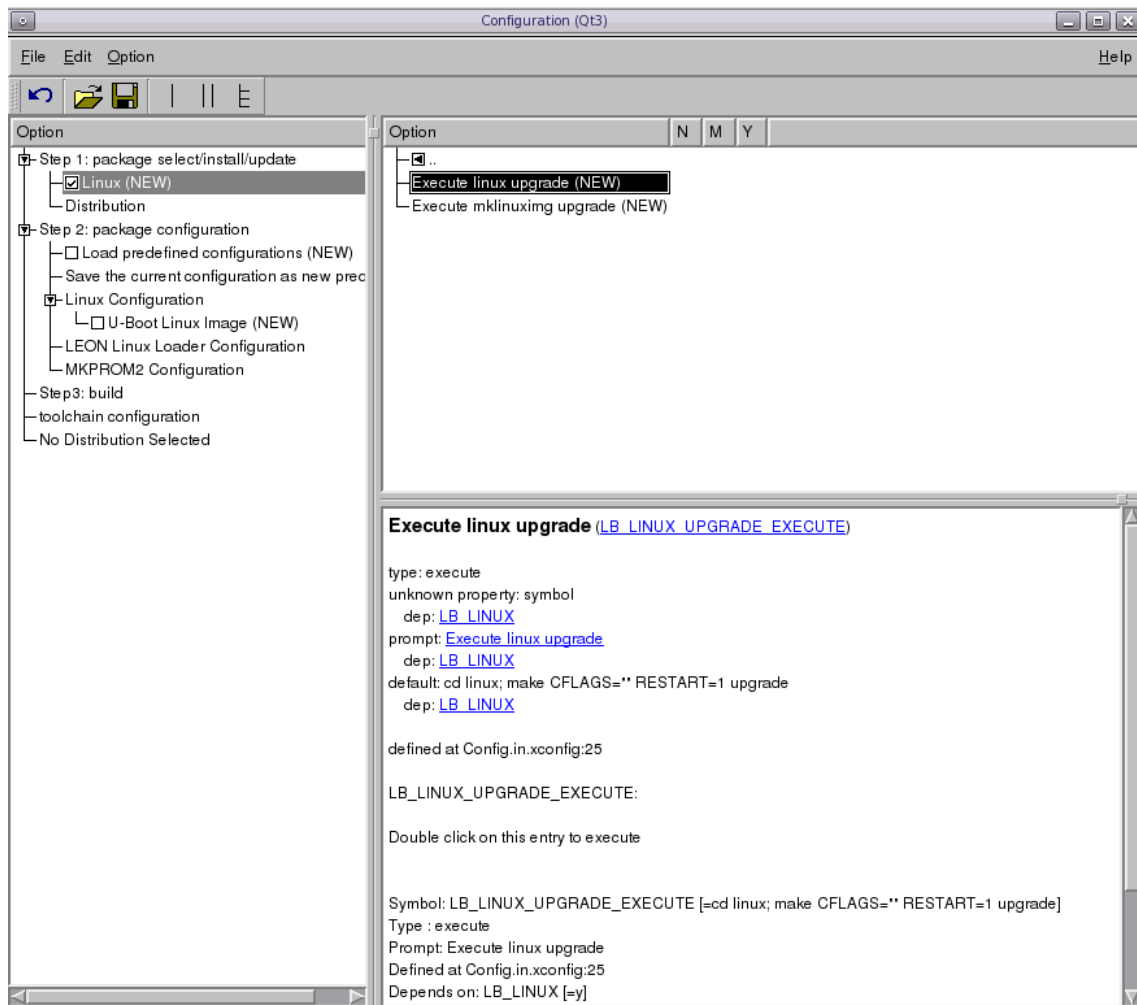


Figure 2.1. Selecting Linux components for installing/upgrading

The configurator is modified from the original Kconfig to execute commands from within the GUI. I.e. when double-clicking on "Execute linux upgrade" a dialog will pop up asking weather to exeute the upgrade command in a new xterm or in the parent shell:

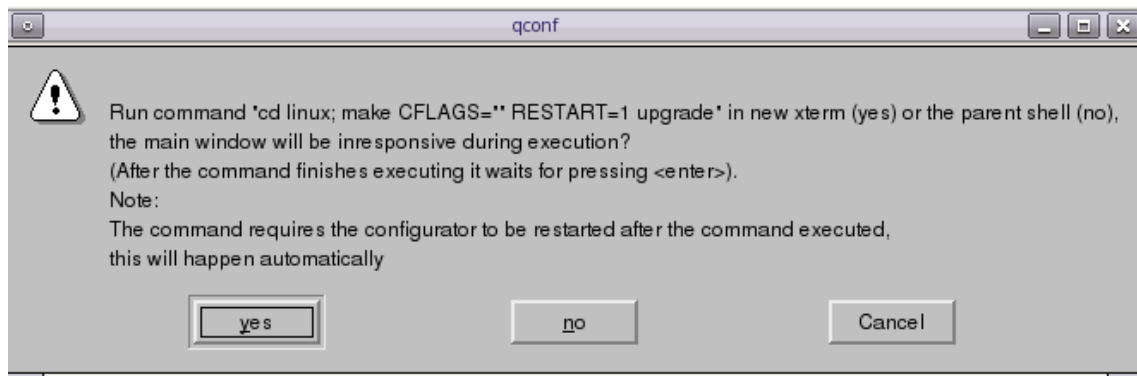


Figure 2.2. Dialog asking weather to execute a command from within the GUI

When doing install/upgrade and when permanently saving a configuration the configurator is restarted automatically so that the configuration change is shown in the configurator.

## 2.1. Buildroot distribution

Below picture shows the upgrade command for the Buildroot sources:

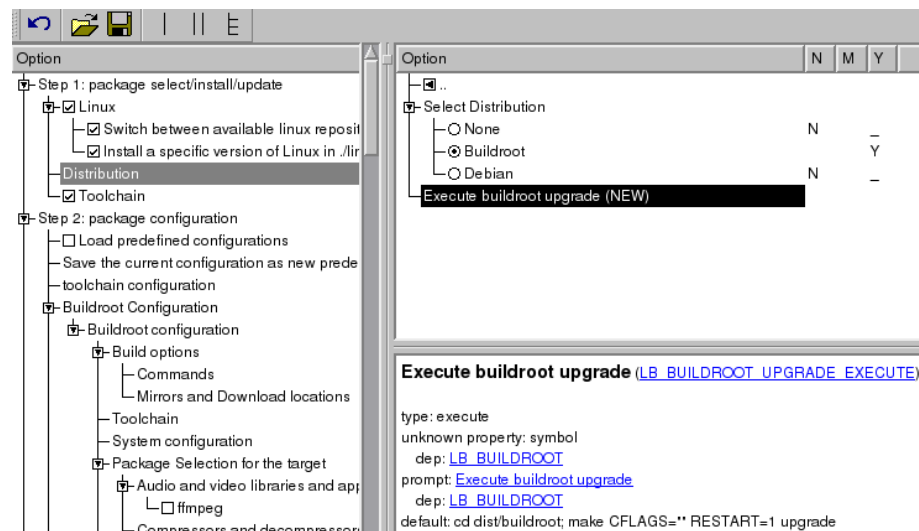


Figure 2.3. Installing/upgrading distribution components

## 2.2. Debian distribution

A Sparc32 wheezy Debian package repository has been created at <http://developer.gaisler.net/debian>. The process of retrieving the packages and installing them is done using the Multistrap utility from <http://wiki.debian.org/Multistrap>. To use multistrap by hand you would create a configuration file m-sparc.config

```
[General]
arch=sparc
directory=/tmp/dist-multistrap-w
cleanup=true
noauth=true
unpack=true
aptsources=Grip Updates
debootstrap=Debian

[Debian]
packages=
source=http://developer.gaisler.net/debian
suite=wheezy
```

and issue a

```
$multistrap -f m-sparc.config
```

The command run from Linuxbuild does some more steps:

- On a system that lacks apt-get download and install under [linux-build]/dist/debian/usr the apt and dpkg suite
- Multistrap requires perl modules Config::Auto, Parse::Debian::Packages and Locale::gettext. Linuxbuild will try to install them from CPAN.
- Download and install under [linux-build]/dist/debian/usr the newest version of tar and fakeroot
- Run multistrap
- Patch the created image directory with various /etc/\* to be able to boot
- Create 2 versions, one [linux-build]/dist/debian/dist-multistrap-w-nfs that is big and that can be used to NFS boot, the other version [linux-build]/dist/debian/dist-multistrap-w-cpio that has some files removed to reduce size (still 20 MB).

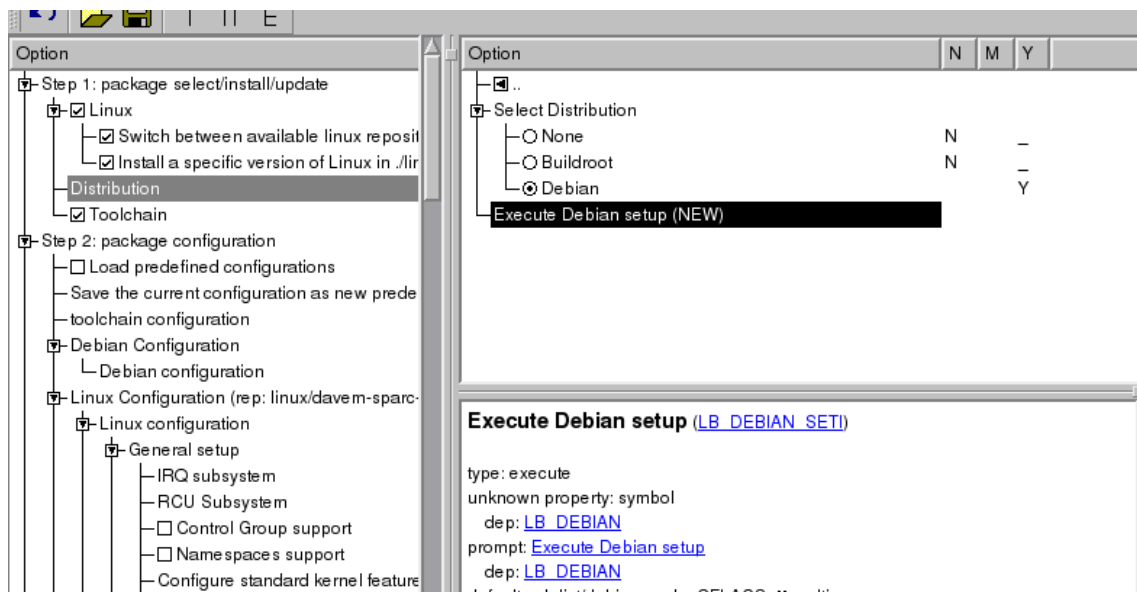


Figure 2.4. Setup from the debian package repository

After the setup the root cpio file should be located at [linuxbuild]/dist/debian/rootfs.cpio.



### 3. Configuring

After the selected components have been downloaded or upgraded, Linuxbuild and each selected component is configured using one of the following make targets (**make xconfig**):

- xconfig - Qt based GUI (Qt-3/4 libs required)
- gconfig - GTK based GUI
- menuconfig - ncurses based terminal interface

The buildroot and Linux configuration trees are displayed as subtrees inside the main screen:

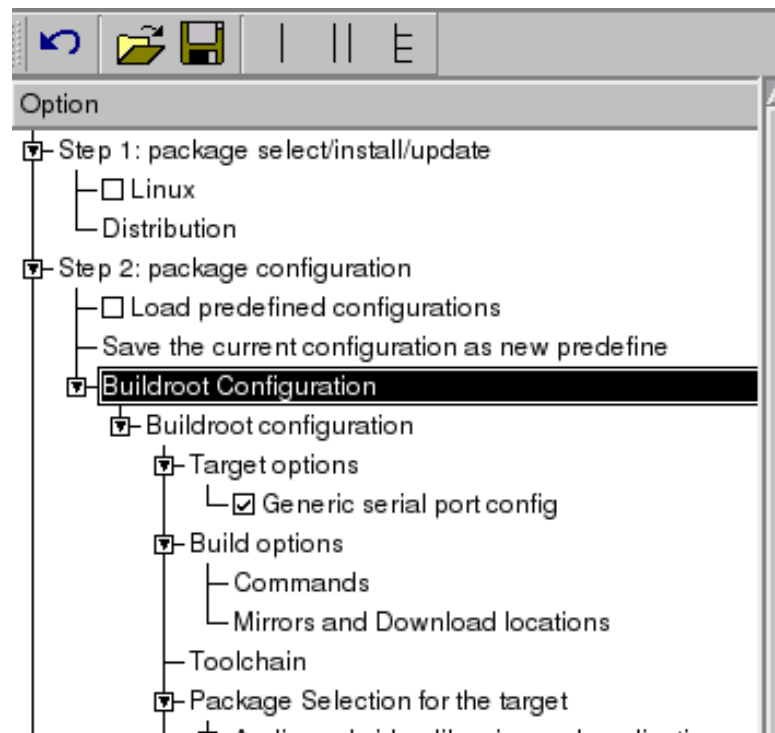


Figure 3.1. The buildroot configuration tree inside the main configuration tree

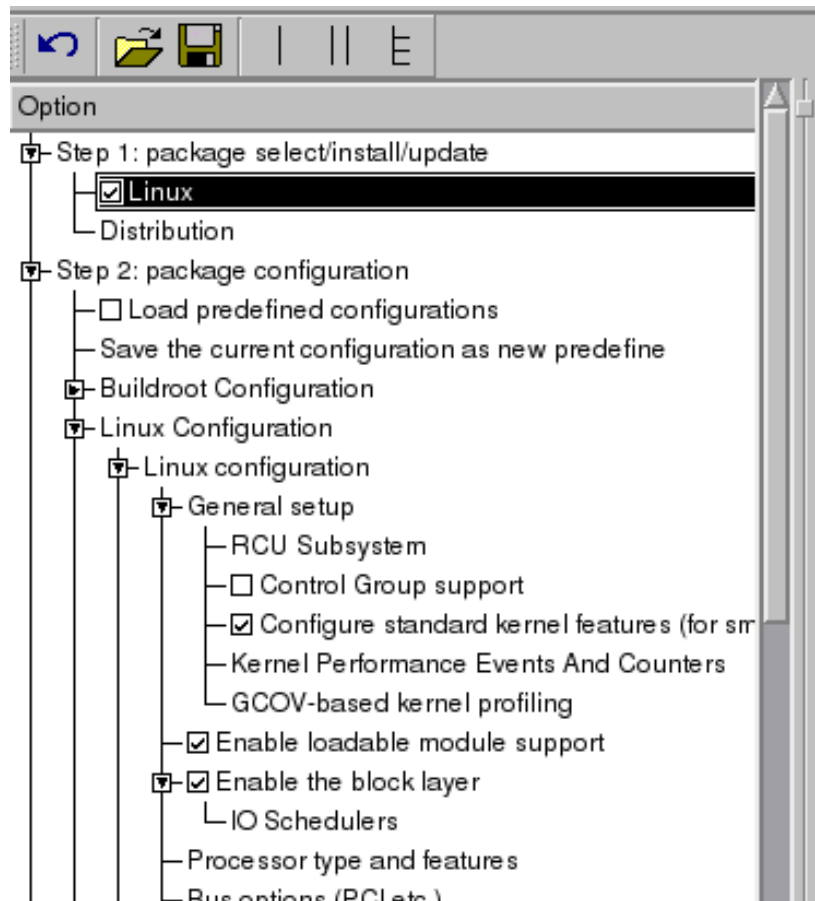


Figure 3.2. The LINUX configuration tree inside the main configuration tree

Prepared LEON configurations can be found in `gaisler/configs`, they can be loaded by Linuxbuild under the "Save/Load Configuration" in the GUI. Loading a configuration updates the current configuration of all components. For this to work all enabled components must have been installed/updated.

Note that since each component is configured separately it is sometimes needed to set the same configuration option more than once in different configuration GUIs.

### 3.1. Multiple Linux repositories

When starting the configurator the subdirectory `./linux/` is scanned for available git repositories. The symlink `./linux/linux-work` will point to the active Linux repository, which is also the one where the Linux configuration tree will be loaded from. To switch `./linux/linux-work` execute the "Execute switch Linux repository" entry as shown below:

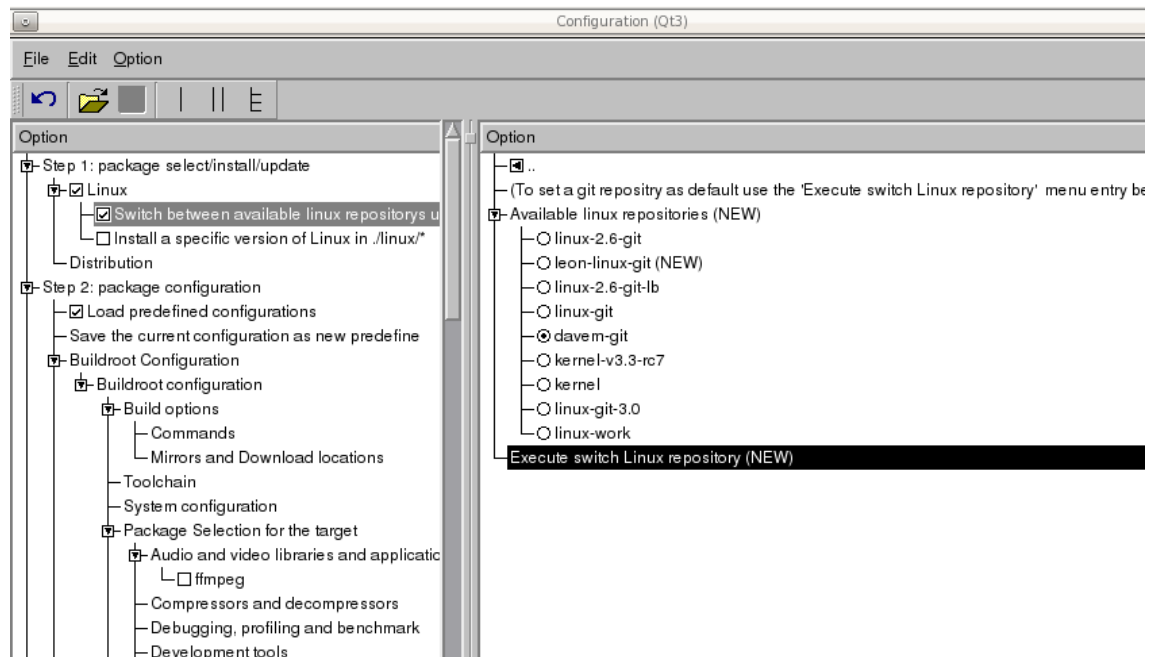


Figure 3.3. Select the repository to set as default

## 3.2. Save/Load configurations

To save the current configuration first specify a name:

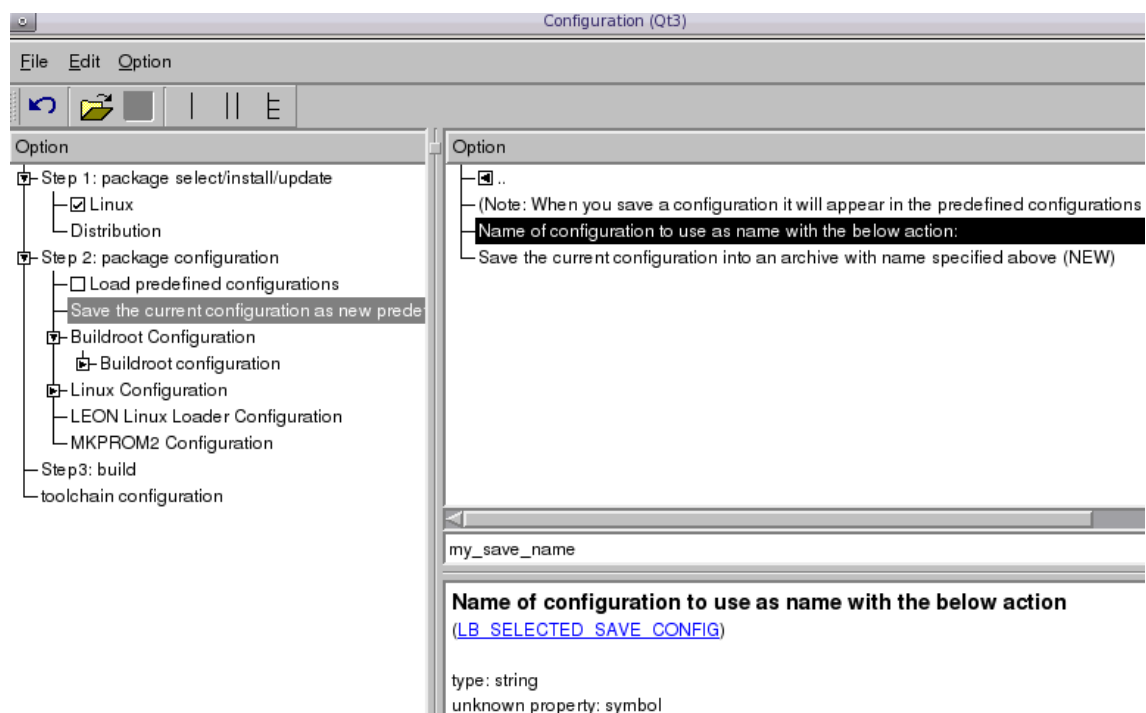


Figure 3.4. Specify the name of the configuration to save

After that execute the save command:

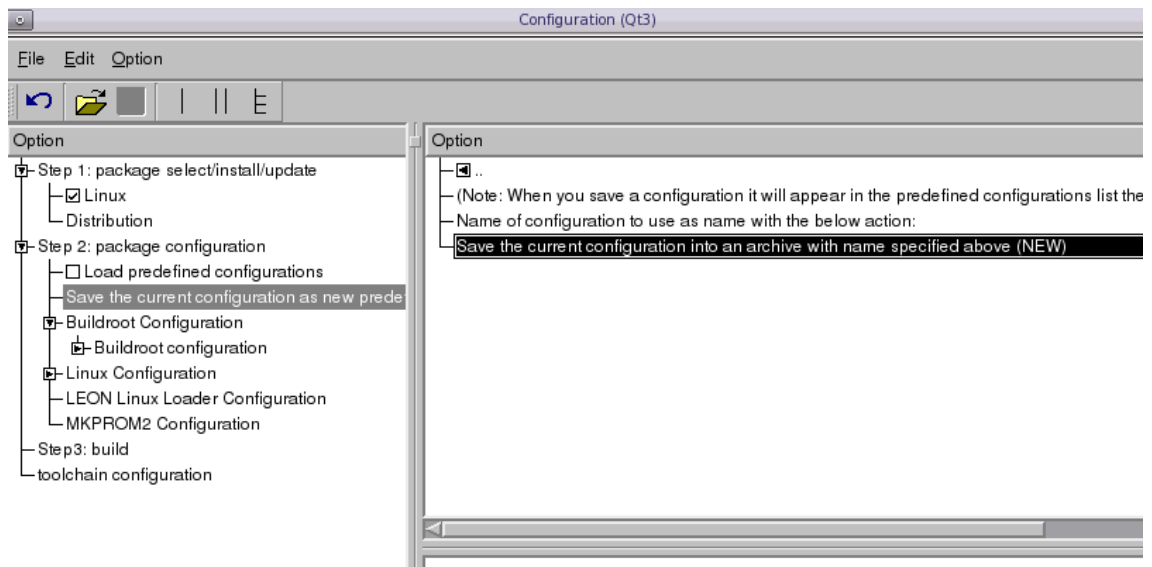


Figure 3.5. Execute the save command

The configurator will restart automatically. After restart the newly saved configuration is accessible in the "Load predefined configurations" entry:

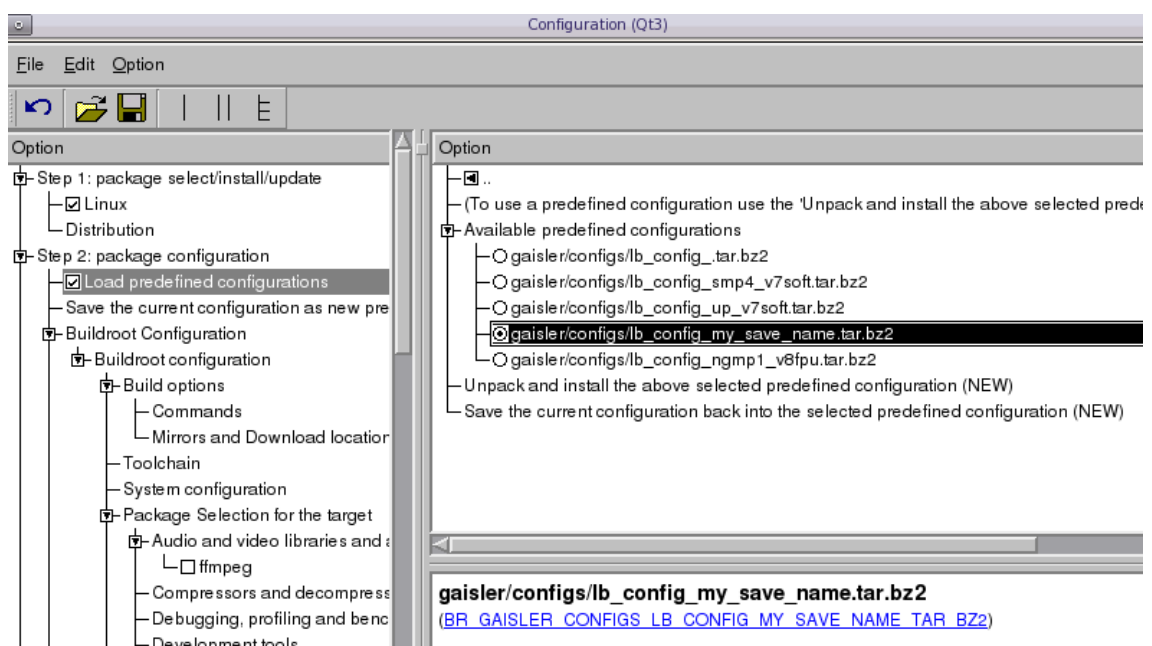


Figure 3.6. List of saved configurations

You can unpack and load the selected configuration:

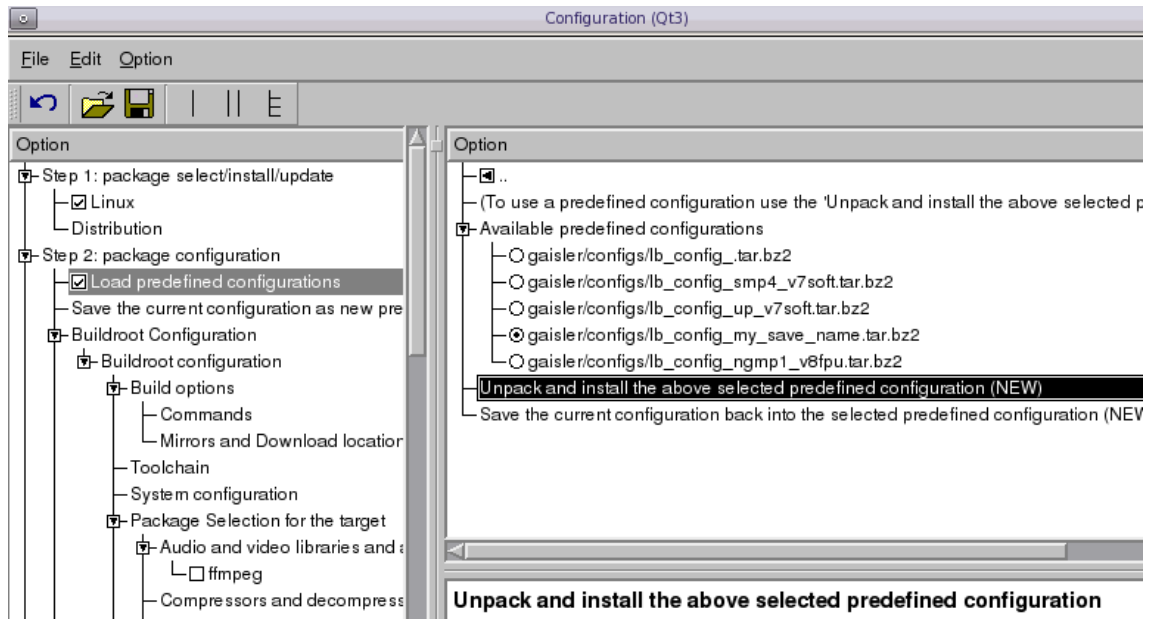


Figure 3.7. Load a saved configuration

### 3.3. Buildroot

Buildroot is used to build user-space applications and tool chains, Aeroflex Gaisler does not distribute uClibc tool chains at the time of writing, in order to build one self Buildroot can be used. In either case a toolchain used to build user-space applications must be selected, see the table below for a number of prepared configurations. The Buildroot configuration GUI is entered by selecting LB\_BUILDROOT\_CONFIGURE (first selecting Buildroot LB\_BUILDROOT).

Table 3.1. Prepared Buildroot Toolchain Configurations

Config	Builds Toolchain	Libc	Target	Toolchain Location
sfleon_shared_basic	YES	uClibc	soft-float/v7	Buildroot builddir/output/staging
sfleov8_shared_basic	YES	uClibc	soft-float/v8	Buildroot builddir/output/staging
hfleon_shared_basic	YES	uClibc	hard-float/v7	Buildroot builddir/output/staging
hfleov8_shared_basic	YES	uClibc	hard-float/v8	Buildroot builddir/output/staging
hfleov8_glibc_basic	NO	GLIBC	hard-float/v8	/opt/sparc-linux-toolchains/hfleov8
sfleon_glibc_basic	NO	GLIBC	soft-float/v7	/opt/sparc-linux-toolchains/sfleon
sfleon_multilib_glibc_basic	NO	GLIBC	soft-float/v7	/opt/sparc-linux-toolchains/multilib
sfleov8_multilib_glibc_basic	NO	GLIBC	soft-float/v8	/opt/sparc-linux-toolchains/multilib
hfleon_multilib_glibc_basic	NO	GLIBC	hard-float/v7	/opt/sparc-linux-toolchains/multilib
hfleov8_multilib_glibc_basic	NO	GLIBC	hard-float/v8	/opt/sparc-linux-toolchains/multilib

### 3.3.1. Network Configuration

After building the Buildroot file system the first time the file system content is located in the Buildroot build directory `build-br/target`. Adding network settings for the network interfaces can be done by editing the `/etc/network/interfaces` file, for example setting `eth0` in DHCP and `eth1` to a static IP address is done by editing the interfaces file as follows:

```
# Configure Loopback
auto lo
iface lo inet loopback

# Do DHCP for ETH0
auto eth0
iface eth0 inet dhcp

# Static IP for ETH1
auto eth1
iface eth1 inet static
    address 192.168.1.207
    network 192.168.1.0
    netmask 255.255.255.0
    broadcast 192.168.1.255
    gateway 192.168.1.1
```

## 4. Building

After configuring the Linuxbuild and all the selected components the build process is started by typing **make build** or by doubleclicking the build command inside the GUI:

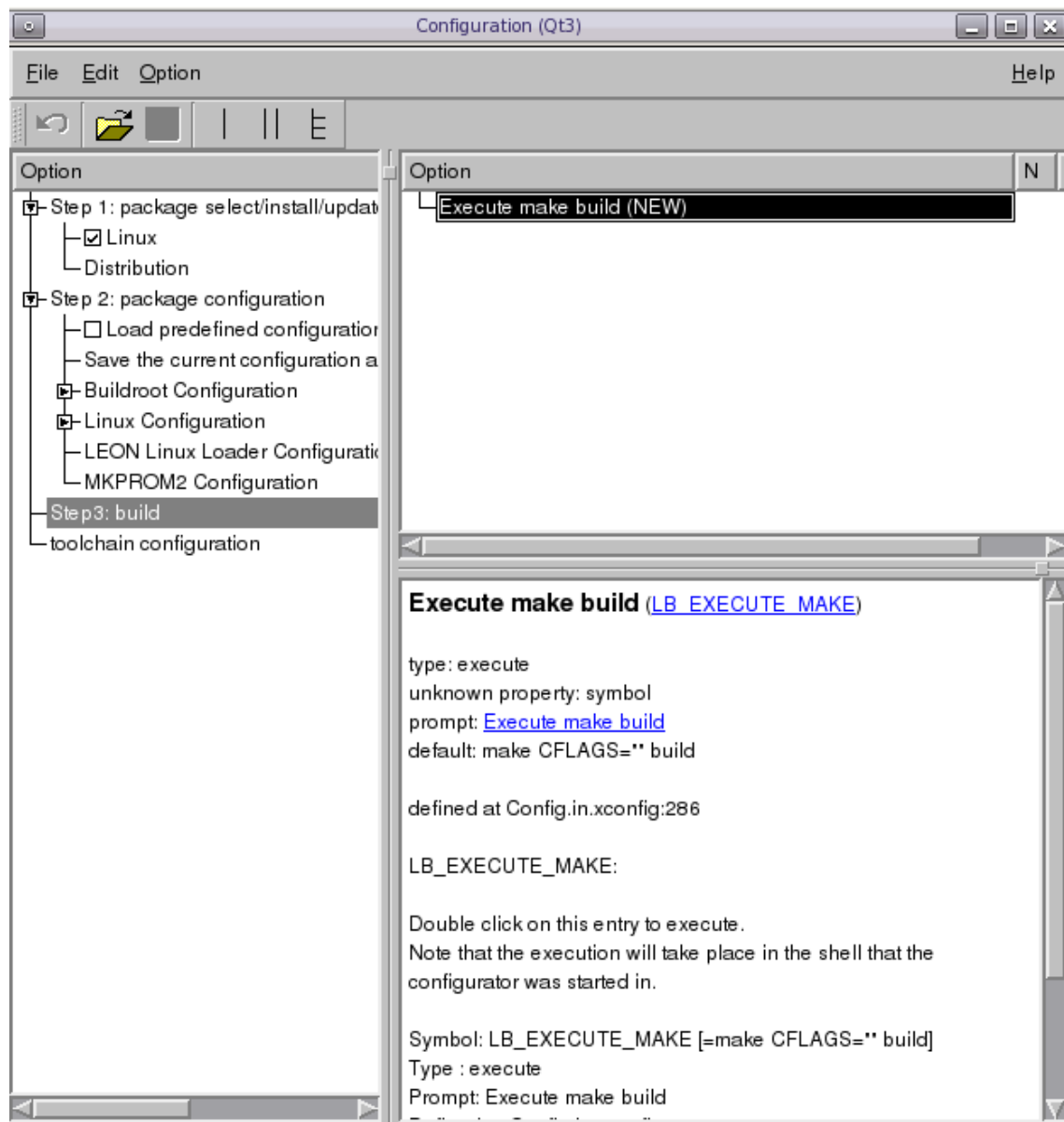


Figure 4.1. Execute build from within the GUI

If the build process fail it may be due to that a required tool is missing. The resulting images and file system images are found in the `output /` directory.

```
$ make build
...
```

## 5. Upgrading

Individual components can be upgraded when Aeroflex Gaisler release new updated packages, the upgrade process can be started from the KConfig GUI (make xconfig).

If a package requires the Linuxbuild itself to be upgraded the user must do that manually. So save time the `linux/linux-git` and `dist/buildroot/buildroot-git` trees can be moved from the old Linuxbuild directory to the new, also the current configuration can be saved from the xconfig GUI and loaded into the new Linuxbuild installation.



---

## 6. Support

For Support, contact the Aeroflex Gaisler support team at [support@gaisler.com](mailto:support@gaisler.com).