crosstool-ng详解

版本：v2.0

Crifan Li

摘要

本文主要介绍了什么是crosstool-ng，crosstool-ng的名字的写法的含义，为何要有crosstool-ng，crosstool-ng的各种特点，crosstool-ng的使用的基本逻辑；

下载crosstool-ng，安装crosstool-ng到当前系统环境中；

接着介绍了crosstool-ng的常见文件夹的结构组织，关于crosstool-ng本身的一些背景知识和逻辑，比如用ct-ng help查看所拥有的功能，用ct-ng list-steps查看本身的build过程分成哪几步，用ct-ng list-samples查看当前已包含哪些默认的示例配置，用ct-ng show-tuple查看当前配置的是什么交叉编译器，用ct-ng version查看当前crosstool-ng的版本，然后介绍了crosstool-ng本身的配置的使用的基本逻辑，包含使用已有的crosstool-ng的示例配置和根据需要去修改你的crosstool-ng的配置

然后极其详细的解释了crosstool-ng中的各种配置参数含义和用法，主要包含几个方面：crosstool-ng工具相关的配置参数，比如多线程编译以节省时间，从之前错误的那一步恢复继续编译，出错时不立刻退出等，再介绍crosstool-ng核心配置详解，包括crosstool-ng本身相关的重要参数，比如源码包下载保存路径，工作路径，目标安装路径，和交叉编译相关的核心参数，包括举例说明Architecture level, Emit assembly for CPU, Tune for CPU如何去配置，以及其他重要参数，比如MMU，大小端Endianness，浮点处理单元FPU，CFLAGS参数，Linux内核版本，binutils版本，gcc版本，C库的选择，线程模型的选择等。

接着介绍用ct-ng build编译crosstool-ng的过程，然后总结了crosstool-ng常见问题及解决办法，包括几方面：安装crosstool-ng工具本身的，crosstool-ng在配置阶段的，crosstool-ng在编译阶段的

最后介绍了与crosstool-ng相关的一些事项，比如crosstool-ng的作者的主页，linaro对crosstool-ng的支持，Buildroot支持crosstool-ng。

本文提供多种格式供：

在线阅读	HTML [1]	HTMLs [2]	PDF [3]	CHM [4]	TXT [5]	RTF [6]	WEBHELP [7]
下载（7zip压缩包）	HTML [8]	HTMLs [9]	PDF [10]	CHM [11]	TXT [12]	RTF [13]	WEBHELP [14]

HTML版本的在线地址为：

http://www.crifan.com/files/doc/docbook/crosstool_ng/release/html/crosstool_ng.html

有任何意见，建议，提交bug等，都欢迎去讨论组发帖讨论：

http://www.crifan.com/bbs/categories/crosstool_ng/

[1] http://www.crifan.com/files/doc/docbook/crosstool_ng/release/html/crosstool_ng.html
[2] http://www.crifan.com/files/doc/docbook/crosstool_ng/release/htmls/index.html
[3] http://www.crifan.com/files/doc/docbook/crosstool_ng/release/pdf/crosstool_ng.pdf
[4] http://www.crifan.com/files/doc/docbook/crosstool_ng/release/chm/crosstool_ng.chm
[5] http://www.crifan.com/files/doc/docbook/crosstool_ng/release/txt/crosstool_ng.txt
[6] http://www.crifan.com/files/doc/docbook/crosstool_ng/release/rtf/crosstool_ng.rtf
[7] http://www.crifan.com/files/doc/docbook/crosstool_ng/release/webhelp/index.html
[8] http://www.crifan.com/files/doc/docbook/crosstool_ng/release/html/crosstool_ng.html.7z
[9] http://www.crifan.com/files/doc/docbook/crosstool_ng/release/htmls/index.html.7z
[10] http://www.crifan.com/files/doc/docbook/crosstool_ng/release/pdf/crosstool_ng.pdf.7z
[11] http://www.crifan.com/files/doc/docbook/crosstool_ng/release/chm/crosstool_ng.chm.7z
[12] http://www.crifan.com/files/doc/docbook/crosstool_ng/release/txt/crosstool_ng.txt.7z
[13] http://www.crifan.com/files/doc/docbook/crosstool_ng/release/rtf/crosstool_ng.rtf.7z
[14] http://www.crifan.com/files/doc/docbook/crosstool_ng/release/webhelp/crosstool_ng.webhelp.7z

修订历史		
修订 2.0	2013-09-04	crl
1. 将帖子内容整理过来 2. 添加了关于配置的详细解释 3. 添加了关于常见错误的总结		

crosstool-ng详解:

Crifan Li

版本：v2.0

出版日期 2013-09-04

[15] http://www.crifan.com/files/doc/docbook/soft_dev_basic/release/html/soft_dev_basic.html#cc_by_nc

目录

范例清单

前言

1. 本文目的

本文目的在于，介绍关于crosstool-ng的各种方面，包括如何下载安装和使用，如何配置和编译等等。

使得对于crosstool-ng不熟悉的人，都可以很快上手，可以实现自己用crosstool-ng，编译出来一个自己的交叉编译器。

2. 待完成

第 1 章 crosstool-ng简介

1.1. 什么是crosstool-ng

crosstool-ng，全称是crosstool Next Generation，即下一代crosstool，即crosstool的升级版

可能有人会问，什么是crosstool？

crosstool，是个交叉编译器的制作工具。

可能又有人问，什么是交叉编译器，什么又是交叉编译器的制作工具？

那你可以去看我的教程：

交叉编译详解 [1]

中的详细解释：

借助别人的工具去制作交叉编译器 [2]

另外，关于crosstool的详细解释，可参见：

crosstool [3]

知道了crosstool-ng是个交叉编译器的制作工具后，再来解释，crosstool-ng的由来：

crosstool-ng的官网是：

crosstool-ng [4]

crosstool-ng是作者Yann E. MORIN，在Dan Kegel[5]写的crosstool[6]的基础上，做了全新的升级。

目标在于，在制作交叉编译器方面，更加容易使用。

1.2. crosstool-ng的名字的写法

其实，crosstool-ng的官网中，都已经专门强调和解释了，关于crosstool-ng，这个名字的，在不同情况下的写法：

Refering to crosstool-NG [7]

下面简要翻译一下为：

crosstool-ng名称的写法和含义.

- crosstool-NG

  此项目的标准名称，全名，长名称。

[1] http://www.crifan.com/files/doc/docbook/cross_compile/release/html/cross_compile.html
[2] http://www.crifan.com/files/doc/docbook/cross_compile/release/html/cross_compile.html#use_tool_build_crosscompiler
[3] http://www.crifan.com/files/doc/docbook/cross_compile/release/html/cross_compile.html#crosscompile_tool_crosstool
[4] http://crosstool-ng.org/
[5] http://kegel.com/
[6] http://kegel.com/crosstool/
[7] http://crosstool-ng.org/#refering_to_crosstool-ng

crosstool和NG之间用短横线连接

crosstool小写

NG大写

- CT-NG

  全部大写

  是全称crosstool-NG的缩写

  CT和NG之间用短横线连接

- ct-ng

  全称缩写后的小写版本，全部小写

  一般是在前端，即使用此crosstool-NG这个工具的时候，其工具名

  后面你会在第 2 章 下载和安装crosstool-ng中看到，安装完毕crosstool-NG后

  接着去使用此crosstool-NG，都是用的是ct-ng，比如

```
ct-ng help
ct-ng menuconfig
ct-ng build
```

  等等

一般来说，建议用长名称crosstool-NG，而不要用短名CT-NG在邮件主题中（为了使标题不要太长），可以使用短名CT-NG

crosstool-ng版本的写法和含义.

- crosstool-NG X.Y.Z

  长名称

  crosstool-NG和版本号X.Y.Z之间有空格

- crosstool-ng-X.Y.Z

  长名称的小写版

  名称和版本之间用短横线连接

  一般用于发布tar包的时候

- crosstool-ng-X.Y.Z+hg_id

  长名称的小写版本，加上短横线，加上版本号，加上加号，再加上Hg的ID（其可以通过ct-ng version而得到）

  用于区分发行版（releases）和快照版（snapshots）

下文中，如无特殊说明，为了省事，都还是采用crosstool-ng的写法。

1.3. 为何要有crosstool-ng

简言之就是：

之前的crosstool，做的不是足够好，所以才有人（Yann E. MORIN）弄出了个更好这个：crosstool-ng

再详细解释一下就是：

crosstool，对于之前制作交叉编译器来说，算数属于首选的工具，已经很不错了。

但是随着时间的发展，其相对来说，还是没有足够好，足够好用。

加上，后者缺乏足够的维护。

所以后来Yann E. MORIN就去弄了个next generation的crosstool，叫做crosstool-ng。

其宗旨就是：让交叉编译器的制作，更加简单，让制作交叉编译器的工具，更加好用。

至于，如何变得简单好用了，就是后面要去介绍的：第 1.4 节 “crosstool-ng的特点”了。

1.4. crosstool-ng的特点

下面，根据我的了解，整理一下，crosstool-ng，作为一个交叉编译器的制作工具，所具有的优点：

1.4.1. 支持（类似于Linux内核配置的那个）menuconfig

menuconfig，用的最为广泛，支持度最好，最好用。

menuconfig可以称为：即简单，又好用。

用过的都知道。

关于make menuconfig，不熟悉的可以参考：

【整理】make config，make menuconfig，make oldconfig，make xconfig，make defconfig，make gconfig [8]

1.4.2. 支持足够多，且越来越多的架构（architecture）

目前已经支持众多了架构了：

Alpha, ARM, AVR32 (EXP), Blackfin (EXP), MIPS, OpenRISC/or32(+), PowerPC, s390, SPARC (EXP), SuperH (EXP), x86

且以后会支持越来越多。

相比较而言，之前的交叉编译器制作工具，支持的架构，比crosstool-ng更多，好像没几个。

1.4.3. 支持工具链中可选多种不同的C库等模块

支持基于uClibc, glibc或eglibc的工具链

支持其他类型的，也很容易添加

详见：第 3.3.2.2.9 节 “crosstool-ng中的配置参数：C库的选择”

[8] http://www.crifan.com/make_config_make_menuconfig_make_oldconfig_make_xconfig_make_defconfig/

1.4.4. 支持不同目标OS平台

支持2种目标运行环境：

Linux

bare metal

详见：交叉编译器名字中的kernel部分[9]

1.4.5. 补丁仓库

为许多种现存的，已知bug的，需要打包的各种模块，提供了一个补丁仓库。

如此可以实现：可以自动去帮你打上，已知的，需要打补丁的各种模块

从而，避免了各种编译的错误，制作出来的交叉编译器的各种问题

极大程度上的节省了你的精力和时间，和降低了制作交叉编译器的难度

1.4.6. 支持不同的线程模型

支持2种线程模型：

NPTL

linuxthreads

详见：第 3.3.2.2.10 节 “crosstool-ng中的配置参数：线程模型的选择”

1.4.7. 支持软浮点和硬件浮点

详见：第 3.3.2.2.4 节 “crosstool-ng中的配置参数：浮点处理单元FPU”

1.4.8. 支持multlib的工具链（暂处于试验阶段）

1.4.9. 支持众多调试相关的内容

支持很多用来调试（debug）的工具：

本地的和跨平台的gdb，gdbserver

调试库：dmalloc, duma

以及一些其他功能，比如从编译失败的那一步，恢复重新编译，从而节省你的大量的时间和精力

详见：第 3.3.1 节 “crosstool-ng工具相关的配置参数”中的：

第 3.3.1.1 节 “多线程编译以节省时间”

第 3.3.1.2 节 “从之前错误的那一步恢复继续编译”

1.4.10. 提供示例配置

背景是：对于交叉编译器的配置，涉及到各个模块的各个版本，自己配置不好的话，很容易出现各种问题，比如版本不兼容

[9] http://www.crifan.com/files/doc/docbook/cross_compile/release/html/cross_compile.html#crosscompiler_kernel_part

而此处提供了，很多个，已经经过验证的配置，针对各个架构（CPU）的各种默认配置

如此，你就可以利用已有的配置，从而简化你的配置的复杂度，极大的降低了你制作交叉工具链的难度，时间，精力。

详见：第 3.1.3 节 “用ct-ng list-samples查看当前已包含哪些默认的示例配置”

1.4.11. 支持多种主机编译环境：各种Linux发行版，Cygwin等

crosstool-ng，不仅支持，普通的各种Linux发行版，比如常见的Ubuntu

还支持Cygwin[10]

这意味着，在纯的Windows环境下

（用Cygwin模拟Linux环境，在Cygwin下制作交叉编译器等等，在Cygwin下用交叉编译器去编译uboot，kernel等等）

也可以去折腾嵌入式开发了

1.5. crosstool-ng的使用的基本逻辑

在详细介绍，然后下载和安装crosstool-ng，如何配置，如何编译crosstool-ng之前，需要先解释一下，crosstool-ng，这个工具，的使用的基本逻辑。

其实，这部分的内容，作者在主页：

crosstool-NG: Download and usage [11]

中，已经解释过了，只是相对比较言简意赅，不是足够详细。

此处，就来详细的解释一下：

在使用crosstool-ng之前，实际上，要明白，有两个概念，不要混淆了：

1. 下载，编译，安装crosstool-ng工具本身（到当前的Linux，Cygwin等系统中，供后续使用）crosstool-ng，作为一个工具，尤其是在Linux领域里

   很明显，也有对应的源码包，即：也需要，通过下载对应的源码，去编译源码，然后再安装到当前的系统中

   其整体的逻辑，和下载，编译，安装，别的，Linux下的软件包，没有啥特殊区别。

Linux下安装第三方工具的基本逻辑

Linux的世界中，使用某个工具软件，和Windows下，不太一样

更多的时候，是自己从源码中编译并安装

所以，需要自己找到自己要的源码包，即下载，然后去配置，再去编译和安装。

即：下载，配置，编译，安装

对于这方面的逻辑，不是很熟悉的话，可参考我总结的：

软件开发基础之linux下安装某个软件的逻辑和过程[12]

[10] http://www.crifan.com/files/doc/docbook/cygwin_intro/release/html/cygwin_intro.html
[11] http://crosstool-ng.org/#download_and_usage
[12] http://www.crifan.com/soft_dev_basic_linux_install_software/

2. 用crosstool-ng这个工具（即ct-ng）去制作（配置和编译）所需要的交叉编译器
在安装了crosstool-ng这个工具后，就可以去使用ct-ng了。

   然后才是去，配置，和编译，以便制作出对应的相应的交叉编译器

   即，用crosstool-ng这个，制作交叉编译器的工具，去制作交叉编译器，的基本步骤，包括：

   a. 配置
   即，为了你要制作的交叉编译器，去配置各种参数

      比如CPU是arm还是mips还是其他，运行目标平台是Linux还是bare metal，所用的C库是elibc，还是glibc，还是uclibc等，等等的配置。

      选用合适的配置，一般的做法，主要有两种：

      - 直接利用已有的配置
      crosstool-ng，本身已经支持很多种体系结构的配置

        很多时候，你可以直接使用已有的配置，而基本无需更改了。

        对应的命令一般是：

        ```
        ct-ng some_default_config
        ```

      - 利用已有的，最接近你的需求的配置，再手动去改改
      有时候，你所需要编译的交叉编译器，和已有的各种配置中，都没有合适的，没有满足你的需求的

        那么就要你，去找一个，和你的需求最接近的一个配置

        利用该配置，然后再去修改一些参数，达到最终你想要的效果

        对应的命令一般是：

        ```
        ct-ng some_default_config
        ct-ng menuconfig
        ```

   b. 编译
   等所有的配置都完毕后，就可以去编译，生成你所需要的交叉编译器了。

      对应的命令一般是：

      ```
      ct-ng build
      ```

   当然，在配置和编译之前和过程中，你需要搞懂很多东西，比如：

   你自己的处理器是什么，所要使用的目标系统是什么，要用什么C库等

   并且还要，根据这些已有信息，搞清楚如何去配置crosstool-ng等等。

   这也就是，此文接下来要解释的内容。

第 2 章 下载和安装crosstool-ng

在使用crosstool-ng，为我们制作交叉编译器之前，需要下载crosstool-ng的源码，并且编译安装crosstool-ng，到我们的当前的（Linux或Cygwin等）系统中。

2.1. 下载crosstool-ng

去crosstool-ng的下载页面：

http://crosstool-ng.org/download/crosstool-ng/

中，下载最新版的crosstool-ng。

截至本文此刻，最新版本是：

crosstool-ng-1.18.0.tar.bz2 [1]

crosstool-ng的最新版本不是下载页面最下面的那个crosstool-ng-1.9.3的版本

之前自己第一次开始去折腾crosstool-ng的时候，从主页

http://crosstool-ng.org/

中找到了：

crosstool-ng的下载页面 [2]

然后，按照正常的逻辑，页面一直下拉到最低端，然后去下载那个：

crosstool-ng-1.9.3.tar.bz2 [3]

以为，理所当然的是最新的版本了，然后接着去折腾：

【记录】在Cygwin下编译gcc-3.4.5-glibc-2.3.6的arm-xscale-linux-gnu交叉编译器 [4]

结果弄到后来，很多错误：

【未解决】crosstool-ng编译出错：Installing C library headers中的make[3]: xxx/sys-root/usr/include/sys/uio.h] Error 1 [5]

然后最后实在不行了，然后无意间发现，原先最新的版本，不是之前的，crosstool-ng-1.9.3，而是：crosstool-ng-1.18.0

对应的，也是在上述所说的下载页面中，下载地址是：

crosstool-ng-1.18.0.tar.bz2 [6]

然后后来才是用此，真正最新的版本，去折腾的：

[1] http://crosstool-ng.org/download/crosstool-ng/crosstool-ng-1.18.0.tar.bz2
[2] http://crosstool-ng.org/download/crosstool-ng/
[3] http://crosstool-ng.org/download/crosstool-ng/crosstool-ng-1.9.3.tar.bz2
[4] http://www.crifan.com/cygwin_compile_gcc_3-4-5_glibc_2-3-6_arm_xscale_linux_gnu_cross_compiler/
[5] http://www.crifan.com/crosstool_ng_installing_c_library_headers_sys_root_usr_include_sys_uio_h_error_1/
[6] http://crosstool-ng.org/download/crosstool-ng/crosstool-ng-1.18.0.tar.bz2

【记录】重试使用最新版本1.18.0的crosstool-ng去配置和编译xscale的交叉编译器 [7]

然后，本来都想去报告作者的，其此下载页面，搞得文件排序，和一般的，不太一样啊，

一般的下载页面，都是保证最下面或最上面是最新版本，方便用户下载的。

对应的，文件排序，是要按照数字大小排序，而不是此处的按照ascii字符排序。

然后后来发现，作者貌似自己已经意识到此问题了，所以，在下载页面，又多了个空文件，放在下载页面最开始，名字就叫做：

00-LATEST-is-1.18.0

以此，来通知来下载的人，不要搞错了。

而实际上，误把crosstool-ng-1.9.3当做最新版本的人，我之前也在网络上看到有过的。

所以，即使是如此的，小问题，实际上，也是需要作者，处理的更好，以更方便用户使用的。

否则，就容易出现这类的问题了。

在此，实际上，还是建议作者，如果可以，还是去修改服务器，保证文件排序是可以通过先ascii字符排序，对于文件中的数字部分，即各个版本号，按照数字的值去排序

注：Total Commander中，是支持此种排序的。感兴趣的去看：

【已解决】total commander里面的文件名是中文加数字而无法正常排序 [8]

2.2. 安装crosstool-ng

在介绍将crosstool-ng安装到当前系统之前，先要说一下：

目前crosstool-ng，除了支持普通的Linux发行版，比如Ubuntu之外，还支持Cygwin环境的。

而我们此处的目标是：把crosstool-ng安装到当前（Linux或Cygwin等）系统中。

所以，接下来的步骤，相对其实很简单，还是常见的：

```
./configure --prefix=/opt/crosstool-ng
make
make install
```

步骤而已。

详细过程，可参考：

详见：【记录】在Ubuntu下用crosstool-ng编译xscale的交叉工具链[9]

详见：【记录】重试使用最新版本1.18.0的crosstool-ng去配置和编译xscale的交叉编译器[10]

[7] http://www.crifan.com/reuse_latest_version_crosstool_ng_to_config_and_compile_for_xscale/
[8] http://www.crifan.com/total_commander_cn_name_plus_number_can_not_sort/
[9] http://www.crifan.com/under_ubuntu_use_crosstool_ng_build_xscale_cross_compiler_toolchain/
[10] http://www.crifan.com/reuse_latest_version_crosstool_ng_to_config_and_compile_for_xscale/

虽然，安装crosstool-ng本身，这个过程中，相对很简单。

但是：

- Ubuntu
  由于在Ubuntu下，默认系统没有安装很多开发工具，导致也还是会遇到一对缺少某工具的情况

  详见：第 5.1.1 节 “缺少一些开发相关的工具”

- Cygwin
  虽然Cygwin只是针对Linux系统的模拟，而不是真正的Linux

  但是，由于我之前安装了所有的开发工具，所以，此处对于Ubuntu下出现的缺少某工具的问题

  倒真的还是很少出现的。

在上述配置，编译，安装crosstool-ng之后，还要做一个事情，那就是：

要把对应的crosstool-ng，这个工具，对应的可执行文件：ct-ng所在的路径，加入到你的环境变量中去。

我一般所用的步骤是：修改当前用户的.bashrc然后将最后一行的PATH变量中，加上此处对应的路径/opt/crosstool-ng/bin，即可。

修改后的.bashrc中最后一行，带PATH的值，类似于这样：

```
PATH=$PATH:/opt/crosstool-ng/bin
```

具体过程，详见：

【记录】在Ubuntu下用crosstool-ng编译xscale的交叉工具链 [11]

【记录】在Cygwin下编译gcc-3.4.5-glibc-2.3.6的arm-xscale-linux-gnu交叉编译器 [12]

当安装crosstool-ng期间出问题

当安装crosstool-ng期间出问题

就先去看看，后面所总结的，常见的错误总结：

第 5.1 节 “安装crosstool-ng工具本身的阶段常见问题及解决办法”

[11] http://www.crifan.com/under_ubuntu_use_crosstool_ng_build_xscale_cross_compiler_toolchain/
[12] http://www.crifan.com/cygwin_compile_gcc_3-4-5_glibc_2-3-6_arm_xscale_linux_gnu_cross_compiler/

第 3 章 配置crosstool-ng：ct-ng menuconfig

相关旧帖：【总结】crosstool-ng使用心得和注意事项[1]

crosstool-ng中关于配置，此处，主要介绍两个大的方面：

1. crosstool-ng本身的配置的使用的基本逻辑
   关于crosstool-ng的配置本身，其实也有一定的使用逻辑和技巧。

   这些逻辑和技巧，实际上，作者在其主页：

   Using a released version[2]

   中已经解释过了，此处，是加上自己实际折腾的经验的基础上

   再次整理出来，解释清楚：第 3.2 节 “crosstool-ng本身的配置的使用的基本逻辑”

2. crosstool-ng的配置期间，各种配置参数的含义，如何去配置
   很明显，由于配置参数极多，并且，不同版本中，也可能发生细微的变化，所以不可能面面俱到的解释所有的配置参数的详细的含义。

   但是，又很明显，一些常见的参数，核心的参数，重要的参数，则至始至终，都不会改变的，而且也是必须要搞懂的。

   所以，此处，主要介绍crosstool-ng中，常见的，核心的那些参数。

   而这些常见的，核心的参数，根据其相关性，又主要分两方面：

   - crosstool-ng作为交叉编译器制作工具本身的使用方面的配置参数
     详见：第 3.3.1 节 “crosstool-ng工具相关的配置参数”
   - 去用crosstool-ng制作交叉编译器，和交叉编译器所相关的配置参数
     详见：第 3.3.2 节 “crosstool-ng核心配置详解”

使用ct-ng之前的一些说明

在开始使用crosstool-ng这个工具之前，即使用ct-ng之前，有个事情要提醒一下：

最好，或者说，一般常见的做法是：

单独为后续使用crosstool-ng制作交叉编译去，而建立一个单独的文件夹

即，执行后续的ct-ng menuconfig，ct-ng build等命令，所处在的路径

比如：我之前，除了，解压crosstool所得的文件夹crosstool-ng-1.18.0之外，去建立了一个对应的文件夹：crosstool-ng-1.18.0_build

同时，为了后续crosstool-ng下载对应的各个包，也建立了对应的src和x-tools两个文件夹。

然后，此刻的目录结构就是：

[1] http://www.crifan.com/summary_crosstool_ng_note/
[2] http://crosstool-ng.org/#using_a_released_version

```
crosstool-ng-1.18.0❶
crosstool-ng-1.18.0_build❷
src❸
x-tools❹
crosstool-ng-1.18.0.tar.bz2❺
```

❶ crosstool-ng的源码包：❺，解压后的文件夹

包含了crosstool-ng的相关源码

❷ 专门为后期使用crosstool-ng去建立交叉编译器，的编译（build），而专门建立的文件夹

对应的，后续的ct-ng menuconfig，ct-ng build等命令，都是在此文件夹下执行的。

❸ 为crosstool-ng中，后续需要下载各种软件的源码包，而准备的，

crosstool-ng中，在开始执行build之后，会去下载对应的源码包，都会存放到这个文件夹下

❹ 这个文件夹，是，用crosstool-ng所生成的交叉编译器，所在的路径。

对应的配置中，会有，类似于：

```
(${HOME}/develop/crosstool-ng/x-tools/${CT_TARGET}) Prefix directory
```

的配置，用来指定生成的交叉编译器，存放在何处。

此时，就是去设置为此处对应的路径即可。

❺ 很明显，这个就是之前我在折腾crosstool-ng-1.18.0时，去下载的源码包了。

对应的上面的❶，就是此源码包解压后的路径。

如此，后续的编译，才显得，相对更加有组织，不至于显得结构太混乱。

3.1. 关于crosstool-ng本身的一些背景知识和逻辑

在去使用和crosstool-ng之前，需要了解一些，和crosstool-ng本身的一些逻辑和背景知识。

然后才能对于后面的配置和编译，如何使用crosstool-ng，有个真正的理解。

3.1.1. 用ct-ng help查看所拥有的功能

正如Linux世界中的常见逻辑，crosstool-ng中，也可以通过help，查看到，其本身具有哪些功能：

```
CLi@PC-CLI-1 ~/develop/crosstool-ng/crosstool-ng-1.18.0_build
$ ct-ng help
This is crosstool-NG version 1.18.0

Copyright (C) 2008  Yann E. MORIN <yann.morin.1998@free.fr>
```

```
This is free software; see the source for copying conditions.
There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A
PARTICULAR PURPOSE.

See below for a list of available actions, listed by category:

Configuration actions:
  menuconfig         - Update current config using a menu based program
  oldconfig          - Update current config using a provided .config as base
  extractconfig      - Extract to stdout the configuration items from a
                     build.log file piped to stdin
  defconfig          - Save current config as a mini-defconfig to ${CONFIG}
  olddefconfig       - Update config from a mini-defconfig ${CONFIG}
                     (default: ${CONFIG}=./defconfig)
  saveconfig         - Save current config as a preconfigured target
  show-tuple         - Print the tuple of the currently configured toolchain

Preconfigured toolchains (#: force number of // jobs):
  list-samples       - prints the list of all samples (for scripting)
  show-<sample>      - show a brief overview of <sample> (list with list-samples)
  <sample>           - preconfigure crosstool-NG with <sample> (list with list-samples)
  build-all[.#]      - Build *all* samples (list with list-samples) and install in
                     ${CT_PREFIX} (which you must set)

Build actions (#: force number of // jobs):
  build[.#]          - Build the currently configured toolchain
  list-steps         - List all build steps

Clean actions:
  clean              - Remove generated files
  distclean          - Remove generated files, configuration and build directories

Distribution actions:
  wiki-samples       - Print a DokuWiki table of samples
  updatetools        - Update the config tools
  tarball            - Build a tarball of the configured toolchain

Environment variables (see /opt/crosstool-ng/share/doc/crosstool-ng/ct-ng.1.18.0/0 - Table of
 content.txt):
  STOP=step          - Stop the build just after this step (list with list-steps)
  RESTART=step       - Restart the build just before this step (list with list-steps)
  CT_PREFIX=dir      - install samples in dir (see action "build-all", above).
  V=0|1|2            - 0 => show only human-readable messages (default)
                     1 => show only the commands being executed
                     2 => show both

Use action "menuconfig" to configure your toolchain
Use action "build" to build your toolchain
Use action "version" to see the version
See "man 1 ct-ng" for some help as well
```

正如此，看了help之后，才有后面的，各种功能的用法。比如：

查看当前有哪些（默认的）示例配置，可以用：

```
ct-ng list-samples
```

查看单个的某个示例配置的核心参数，用：

```
ct-ng show-<sample>
```

直接借用（使用，调用）该默认配置，用：

```
ct-ng <sample>
```

等等。

关于其他的更多用法，后续会详细解释的。

3.1.2. 用ct-ng list-steps查看本身的build过程分成哪几步

如上所述，用list-steps，可以查看到，当前，用crosstool-ng去build，去制作交叉编译器，具体分成哪些步骤：

```
CLi@PC-CLI-1 ~/develop/crosstool-ng/crosstool-ng-1.18.0_build
$ ct-ng list-steps
Available build steps, in order:
  - libc_check_config
  - companion_libs_for_build
  - binutils_for_build
  - companion_libs_for_host
  - binutils_for_host
  - cc_core_pass_1
  - kernel_headers
  - libc_start_files
  - cc_core_pass_2
  - libc
  - cc_for_build
  - cc_for_host
  - libelf_for_target
  - binutils_for_target
  - debug
  - test_suite
  - finish
Use "<step>" as action to execute only that step.
Use "+<step>" as action to execute up to that step.
Use "<step>+" as action to execute from that step onward.
```

知道了，此处分了多少步骤，具体有哪些步骤之后

后面才有可能去实现，从之前出错的哪一步，恢复，继续编译。

3.1.3. 用ct-ng list-samples查看当前已包含哪些默认的示例配置

如第 3.1.1 节 “用ct-ng help查看所拥有的功能” 中所述，我们可以通过

```
ct-ng list-samples
```

查看到，当前crosstool-ng中，已经有了哪些默认的，帮我们配置好的哪些配置：

```
CLi@PC-CLI-1 ~/develop/crosstool-ng/crosstool-ng-1.18.0
$ ct-ng list-samples
Status  Sample name
  MKDIR config.gen
  IN    config.gen/arch.in
  IN    config.gen/kernel.in
  IN    config.gen/cc.in
  IN    config.gen/binutils.in
  IN    config.gen/libc.in
  IN    config.gen/debug.in
[L.X]   alphaev56-unknown-linux-gnu
[L.X]   alphaev67-unknown-linux-gnu
[L.X]   arm-bare_newlib_cortex_m3_nommu-eabi
[L.X]   arm-cortex_a15-linux-gnueabi
[L..]   arm-cortex_a8-linux-gnueabi
[L..]   arm-davinci-linux-gnueabi
[L..]   armeb-unknown-eabi
[L.X]   armeb-unknown-linux-gnueabi
[L.X]   armeb-unknown-linux-uclibcgnueabi
[L..]   arm-unknown-eabi
[L..]   arm-unknown-linux-gnueabi
[L.X]   arm-unknown-linux-uclibcgnueabi
[L.X]   armv6-rpi-linux-gnueabi
[L.X]   avr32-unknown-none
[L..]   bfin-unknown-linux-uclibc
[L..]   i586-geode-linux-uclibc
[L.X]   i586-mingw32msvc,i686-none-linux-gnu
[L.X]   i686-nptl-linux-gnu
[L.X]   i686-unknown-mingw32
[L.X]   m68k-unknown-elf
[L.X]   m68k-unknown-uclinux-uclibc
[L.X]   mips64el-n32-linux-uclibc
[L.X]   mips64el-n64-linux-uclibc
[L.X]   mips-ar2315-linux-gnu
[L..]   mipsel-sde-elf
[L..]   mipsel-unknown-linux-gnu
[L.X]   mips-malta-linux-gnu
[L..]   mips-unknown-elf
[L.X]   mips-unknown-linux-uclibc
[L..]   powerpc-405-linux-gnu
[L.X]   powerpc64-unknown-linux-gnu
[L..]   powerpc-860-linux-gnu
[L.X]   powerpc-e300c3-linux-gnu
[L.X]   powerpc-e500v2-linux-gnuspe
[L..]   powerpc-unknown_nofpu-linux-gnu
[L..]   powerpc-unknown-linux-gnu
[L..]   powerpc-unknown-linux-uclibc
[L.X]   s390-ibm-linux-gnu
[L.X]   s390x-ibm-linux-gnu
[L..]   sh4-unknown-linux-gnu
[L..]   x86_64-unknown-linux-gnu
[L..]   x86_64-unknown-linux-uclibc
[L.X]   x86_64-unknown-mingw32
 L (Local)       : sample was found in current directory
 G (Global)      : sample was installed with crosstool-NG
 X (EXPERIMENTAL): sample may use EXPERIMENTAL features
 B (BROKEN)      : sample is currently broken
```

如此，后续就可以去使用这些默认配置了。

3.1.4. 用ct-ng show-tuple查看当前配置的是什么交叉编译器

在后续的配置（和编译）之后，可以通过show-tuple，去查看到当前配置的交叉编译器是啥样的：

比如，我之前在折腾：

【记录】重试使用最新版本1.18.0的crosstool-ng去配置和编译xscale的交叉编译器 [3]

后，对应的结果是：

```
CLi@PC-CLI-1 ~/develop/crosstool-ng/crosstool-ng-1.18.0_build
$ ct-ng show-tuple
arm-xscale-linux-gnueabi
```

就知道当前，配置的交叉编译器，是：arm-xscale-linux-gnueabi

3.1.5. 用ct-ng version查看当前crosstool-ng的版本

类似的，可以通过version查看当前的crosstool-ng工具本身的版本：

```
CLi@PC-CLI-1 ~/develop/crosstool-ng/crosstool-ng-1.18.0_build
$ ct-ng version
This is crosstool-NG version 1.18.0

Copyright (C) 2008  Yann E. MORIN <yann.morin.1998@free.fr>
This is free software; see the source for copying conditions.
There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A
PARTICULAR PURPOSE.
```

即，此处的crosstool-ng的版本是：1.18.0

3.2. crosstool-ng本身的配置的使用的基本逻辑

从之前介绍的第 3.1.3 节 “用ct-ng list-samples查看当前已包含哪些默认的示例配置”，我们可以看出

对于常见的架构，比如arm，mips，powerpc等等，都有了很多的，已经帮我验证过，可以正常编译的示例配置了

所以，我们接下来，主要就是：

[3] http://www.crifan.com/reuse_latest_version_crosstool_ng_to_config_and_compile_for_xscale/

搞懂自己借用哪个配置，然后调用默认配置，然后再确认一下配置，根据自己的情况去改一改，就差不多，就配置好了。

下面介绍，如何去配置crosstool-ng的主要的两种情况：

- 如果已有同样（类似的）配置，那么直接拿过来用即可
- 如果没有需要的配置，则找到一个最为接近的配置，借用后，再去修改为自己所要的配置

下面就来详细阐述其含义：

3.2.1. 使用已有的crosstool-ng的示例配置

很多时候，我们用crosstool-ng去编译我们要的交叉编译器的时候，所针对的CPU，目标运行环境，想要用的C库等等，

很可能，和之前上面看到的某个配置，是一样的。

此时，我们所要做的事情，其实就是直接拿来用即可。

比如，我们想要针对cortex-a8这款CPU，去编译一个交叉编译器，那么你就会发现，其实默认已有此配置了：

```
arm-cortex_a8-linux-gnueabi
```

在使用之前，也是可以先去看看，该配置的主要参数是哪些：

```
CLi@PC-CLI-1 ~/develop/crosstool-ng/crosstool-ng-1.18.0_build
$ ct-ng show-arm-cortex_a8-linux-gnueabi
[G..]   arm-cortex_a8-linux-gnueabi
    OS              : linux-3.7.3
    Companion libs : gmp-4.3.2 mpfr-2.4.2 ppl-0.10.2 cloog-ppl-0.15.9 libelf-0.8.13
    binutils        : binutils-2.20.1a
    C compiler      : gcc-4.4.6 (C,C++)
    C library       : glibc-2.9 (threads: nptl)
    Tools           : dmalloc-5.5.2 duma-2_5_15 gdb-6.8a ltrace-0.5.3 strace-4.5.19
```

然后，你就可以直接去调用此配置了：

```
ct-ng arm-cortex_a8-linux-gnueabi
```

实际输出，是类似于这样的：

```
CLi@PC-CLI-1 ~/develop/crosstool-ng/crosstool-ng-1.18.0_build
$ ct-ng arm-cortex_a8-linux-gnueabi
  LN    config
  MKDIR config.gen
  IN    config.gen/arch.in
  IN    config.gen/kernel.in
  IN    config.gen/cc.in
  IN    config.gen/binutils.in
  IN    config.gen/libc.in
  IN    config.gen/debug.in
```

```
  CONF  config/config.in
#
# configuration saved
#

*************************************************************

Initially reported by: Yann E. MORIN
URL: http://ymorin.is-a-geek.org/

*************************************************************

Now configured for "arm-cortex_a8-linux-gnueabi"
```

一些实际折腾的过程，可参考：

【记录】重试使用最新版本1.18.0的crosstool-ng去配置和编译xscale的交叉编译器 [4]

接下来，你就可以去进入配置：

```
ct-ng menuconfig
```

去确认一下，是否是完全符合你的要求。

如果有什么个别的配置不满意，再去修改为你自己想要的即可。

3.2.2. 根据需要去修改你的crosstool-ng的配置

很多时候，已经的哪些示例配置，未必完全符合你的需要，这时候，你就只能是：

找个和你的情况最接近的一个示例配置，然后再去基于此配置，去修改成你所要的。

比如，我之前折腾，为TQ2440的S3C2440的CPU，其是arm920t的核

去配置crosstool-ng的时候，就发现，默认的配置中，没有完全合适的。

然后就一点点去调查，而最后搞清楚了，在已有的，arm相关的配置中，davinci，是相对来说，比较接近此处的arm920t的了。

所以，就去调用davinci的默认配置：

```
ct-ng arm-davinci-linux-gnueabi
```

然后再按照自己的需求，进入：

```
ct-ng menuconfig
```

后，去根据自己的需要，再去确认配置，和修改配置，为自己所需要的。

详细过程参见：

【记录】Cygwin下为arm920t配置crosstool-ng [5]

[4] http://www.crifan.com/reuse_latest_version_crosstool_ng_to_config_and_compile_for_xscale/
[5] http://www.crifan.com/cygwin_crosstool_ng_do_ct_ng_menuconfig_config_for_arm920t/

3.3. crosstool-ng的配置参数详解

如上，借用已有的配置后，接下来就是去：

```
ct-ng menuconfig
```

然后针对里面的配置选项，去根据自己的需要，去修改了。

由于crosstool-ng的功能太多，配置选项太多，

所以，此处，也不太可能，针对每个参数都详细解释其含义。

不过，对于一些核心参数，是我们必须要了解清楚的，

所以，下面就针对，一些核心参数，比较有用的参数，来详细解释解释：

3.3.1. crosstool-ng工具相关的配置参数

crosstool-ng中，有不少有用的参数，是和，crosstool-ng这个工具本身，有关的

其目的在于，在为你使用此工具时，提供使用上的便利

比如设置多线程，以加快编译速度，节省编译时间，

之前编译在某步出错，然后重新重头编译又需要耗费大量的无谓的时间，而提供了从之前错误的某步中恢复，继续编译，从而节省你的时间

在编译出错时，不退出，而提供机会给你修复错误，修复后，然后可以继续编译

如此等等的参数，目的就一个：

让制作交叉编译器，这个费事费力的活，变得尽可能的省时省力

下面，就来介绍一下这些参数，如何配置，以及详细的含义和用法。

3.3.1.1. 多线程编译以节省时间

直接给出示例配置：

```
Paths and misc options
    (4) Number of parallel jobs
```

其help的说明为：

```
  ┌──────────────────────────────────── Number of parallel jobs ─────────────────────────────────────┐
  │ CT_PARALLEL_JOBS:                                                                               │
  │                                                                                                 │
  │ Number of jobs make will be allowed to run concurently.                                         │
  │ Set this higher than the number of processors you have, but not too high.                       │
```

```
| A good rule of thumb is twice the number of processors you have.          |
|                                                                           |
| Enter 1 to have only one job at a time.                                   |
|                                                                           |
| Enter 0 to set automatically based on how many processors the host has.   |
|                                                                           |
| Symbol: PARALLEL_JOBS [=4]                                                |
| Type  : integer                                                           |
| Prompt: Number of parallel jobs                                           |
|   Defined at config/global/build-behave.in:8                              |
|   Depends on: !BACKEND [=n]                                               |
|   Location:                                                               |
|     -> Paths and misc options                                             |
```

此处，设置为4，意思是：

同时运行4个进程（去执行crosstool-ng的编译过程）

很明显，多线程去编译的话，会大大缩短整个的编译时间。

此数字的设置，当然，不能随便胡乱设置。

比较常见的做法是：

CPU内核数 x 2

比如，你的是Intel的双核的CPU，那么此值就是：

2 x 2 =4

了。

不用去ct-ng build.N

此处，设置好此参数后，后续直接去build：

```
ct-ng build
```

即可自动以对应的多线程去编译了。

无需后续手动再在build上参数，即无需：

```
ct-ng build.4
```

内部是用make -jN去实现的

此处的多线程的参数

内部，是把对应的数字，传递给make，即实现：

```
make -j4
```

的效果，然后make就会以对应的4个线程去执行了。

关于make的-j参数，不了解的可参考：

【整理】Linux下的make命令使用心得 [6]

[6] http://www.crifan.com/summary_usage_about_make_linux_command/

3.3.1.2. 从之前错误的那一步恢复继续编译

示例配置：

```
Paths and misc options
   [*] Debug crosstool-NG
   [*]   Save intermediate steps
```

Save intermediate steps的help的说明为：

```
 ┌──────────────────────────────────── Save intermediate steps
─────────────────────────────────────┐
 │ CT_DEBUG_CT_SAVE_STEPS:                                          |
 |                                                                  |
 │ If you say 'y' here, then you will be able to restart crosstool-NG at       |
 │ any step.                                                        |
 |                                                                  |
 │ It is not currently possible to restart at any of the debug facilities.     |
 │ They are treated as a whole.                                     |
 |                                                                  |
 │ To get the full list os steps, run: ct-ng list-steps             |
 |                                                                  |
 │ Symbol: DEBUG_CT_SAVE_STEPS [=y]                                 |
 │ Type  : boolean                                                  |
 │ Prompt: Save intermediate steps                                  |
 |   Defined at config/global/ct-behave.in:79                       |
 |   Depends on: DEBUG_CT [=y]                                      |
 |   Location:                                                      |
 |     -> Paths and misc options                                    |
 |       -> Debug crosstool-NG (DEBUG_CT [=y])                      |
```

此处，先要选上：Debug crosstool-NG

表示，使用crosstoo-ng的调试方面的功能

再去选上：Save intermediate steps

意思是：

编译完毕每一步之后，都会保存对应的状态

如此，就可以实现对应的，从之前出错的步骤恢复而继续编译的效果了。

当然，实现此效果的前提是，你参考之前的第 3.1.2 节 “用ct-ng list-steps查看本身的build过程分成哪几步”，而搞清楚了，crosstool-ng的编译，其本身分了哪些步骤。

然后直到自己是在哪一步出的错，然后才可以实现回复出错的步骤，而继续编译。

总的逻辑是：

找到之前编译，最后成功的那一步（last successful step）。

然后去执行：

```
ct-ng last_successful_step+
```

就可以继续恢复继续编译了。

例 3.1. 从出错的那一步恢复重新继续编译

比如，我之前折腾：

【已解决】crosstool-ng在Installing C library headers & start files期间出错：Makefile:240: *** mixed implicit and normal rules. Stop. [7]

期间，当时出错的输出信息是：

```
================================================================
[INFO ]  Installing MPC for host
[EXTRA]    Configuring MPC
[EXTRA]    Building MPC
[EXTRA]    Installing MPC
[INFO ]  Installing MPC for host: done in 182.22s (at 27:10)
[EXTRA]  Saving state to restart at step 'binutils_for_host'...
[INFO ]
 ================================================================
[INFO ]  Installing binutils for host
[EXTRA]    Configuring binutils
[ERROR]    configure: error: cannot create configure.lineno; rerun with a POSIX shell
[ERROR]
[ERROR]  >>
[ERROR]  >>  Build failed in step 'Installing binutils for host'
[ERROR]  >>        called in step '(top-level)'
[ERROR]  >>
[ERROR]  >>  Error happened in: CT_DoExecLog[scripts/functions@257]
[ERROR]  >>        called from: do_binutils_backend[scripts/build/binutils/binutils.sh@205]
[ERROR]  >>        called from: do_binutils_for_host[scripts/build/binutils/binutils.sh@92]
[ERROR]  >>        called from: main[scripts/crosstool-NG.sh@632]


Current command:
  'CFLAGS=-O2 -g -pipe ' 'CXXFLAGS=-O2 -g -pipe ' 'LDFLAGS= ' '/home/CLi/develop/
crosstool-ng/crosstool-ng-1.18.0_build/.build/src/binutils-2.22/configure' '--build=i686-
build_pc-cygwin' '--host=i686-build_pc-cygwin' '--target=arm-xscale-linux-gnueabi' '--
prefix=/home/CLi/develop/crosstool-ng/x-tools/arm-xscale-linux-gnueabi' '--disable-werror'
 '--enable-ld=yes' '--enable-gold=no' '--with-pkgversion=crosstool-NG 1.18.0' '--disable-
multilib' '--disable-nls' '--with-float=softfp' '--with-sysroot=/home/CLi/develop/crosstool-ng/
x-tools/arm-xscale-linux-gnueabi/arm-xscale-linux-gnueabi/sysroot'
exited with error code: 1
Please fix it up and finish by exiting the shell with one of these values:
    1  fixed, continue with next build command
    2  repeat this build command
    3  abort build

ct-ng:~/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/
build/build-binutils-host-i686-build_pc-cygwin>
```

[7] http://www.crifan.com/crosstool_ng_makefile_mixed_implicit_and_normal_rules_stop/

此处，我们能看到的是，在当前的：

```
[INFO ]  Installing binutils for host
```

而出错了，而对应的，在此之前的，最后一个正确编译的阶段，是对应着有输出：

```
[EXTRA]  Saving state to restart at step 'binutils_for_host'...
```

的那一步，即：

```
binutils_for_host
```

此时，你就可以用：

```
ct-ng binutils_for_host+
```

而实现：

（给定，之前最后正常执行的那一步（last successful step），往后，）从上述出错的那一步，继续正常编译的效果了。

即，所谓的：restore from fail step

此处不是ct-ng last_fail_step+

此处，之前不熟悉此功能的用法，结果去用：

```
ct-ng last_fail_step+
```

的方法，即用：

```
ct-ng cc_core_pass_1+
```

想要回复错误，继续执行，结果就出现：

```
[ERROR]  The previous build did not reach the point where it could be restarted at xxx
```

之类的错误了。

详见：

【已解决】crosstool-ng在Installing C library headers & start files期间出错：Makefile:240: *** mixed implicit and normal rules. Stop.[8]

RESTART=step的参数估计也可以实现同样的从错误那步恢复继续编译的效果

从之前的第 3.1.1 节 “用ct-ng help查看所拥有的功能”中可以看出

估计在build的时候，指定对应的RESTART=step

```
ct-ng build RESTART=last_successful_step
```

应该也可以实现类似的效果的。

有空的话，可以去试试。

[8] http://www.crifan.com/crosstool_ng_makefile_mixed_implicit_and_normal_rules_stop/

3.3.1.3. 出错时不立刻退出

说实话，对于上面的那个：第 3.3.1.2 节 “从之前错误的那一步恢复继续编译”的功能，已经很强大了

而crosstool-ng，工具做的就是好。

因为，除此之外，又进一步的支持这个，出错时不立刻退出，的功能。

配置如下：

```
Paths and misc options
   [*] Debug crosstool-NG
   [*]   Interactive shell on failed commands
```

Interactive shell on failed commands的help的说明为：

```
 ┌──────────────────────── Interactive shell on failed commands
─────────────────────────┐
 │ CT_DEBUG_INTERACTIVE:                                        |
 |                                                              |
 | If you say 'y' here, then an interactive shell will be spawned for          |
 | each failed command.                                         |
 |                                                              |
 | This shell will have the same environment that the failed command           |
 | was run with, and the working directory will be set to the directory        |
 | the failed command was run in.                               |
 |                                                              |
 | After you fix the issue, you can exit the interactive shell with any        |
 | of these exit codes:                                         |
 |   1  the issue was fixed, continue the build with the next command          |
 |   2  the issue was fixed, re-run the failed command                         |
 |   3  abort the build                                         |
 |                                                              |
 | Note: '2' is only possible for commands run via CT_DoExecLog, though.       |
 |                                                              |
 | Symbol: DEBUG_INTERACTIVE [=y]                               |
 | Type  : boolean                                              |
 | Prompt: Interactive shell on failed commands                 |
 |   Defined at config/global/ct-behave.in:121                  |
 |   Depends on: DEBUG_CT [=y]                                  |
 |   Location:                                                  |
 |     -> Paths and misc options                                |
 |       -> Debug crosstool-NG (DEBUG_CT [=y])                  |
```

此处，参数Interactive shell on failed commands的含义是：

直译为：当编译（命令）出错时，提供交互式的shell

其具体实际效果是：

正常情况下，当编译出错，会直接退出crosstool-ng的编译环境，回到当前的命令行。

而设置了此参数后，当编译失败时，不是直接退出编译

而是提供一个交互式的shell

然后你可以（另开一个终端）去修复你的问题

然后修复问题后，再通过

```
exit N
```

实现对应的返回N的值，达到对应的效果：

- 1
  （不重新执行，之前错误的那个命令）

  而接着直接继续编译

- 2
  重新执行之前错误的那个命令

  然后接着继续执行

  常用于：

  当某个.c文件编译出错了，然后你另起一个终端，去修改了该.c文件，解决了错误。

  然后就可以通过

  ```
  exit 2
  ```

  而使得，重新执行命令，重新编译该.c文件，而使得可以正常编译，继续编译。

- 3
  直接退出编译

  相当于，直接按Ctrl+C，而中断编译

例 3.2. 如何在编译失败后，修复错误，然后继续接着编译

比如，我之前折腾

【记录】crosstool为xscale编译(ct-ng build)过程 [9]

时，遇到编译出错后，就回到当前命令行了：

```
================================================================================
[INFO ]  Installing static core C compiler
[EXTRA]    Configuring static core C compiler
[EXTRA]    Building static core C compiler
[EXTRA]    Installing static core C compiler
[INFO ]  Installing static core C compiler: done in 1282.54s (at 67:22)
[EXTRA]  Saving state to restart at step 'libc_headers'...
[INFO ]
================================================================================
[INFO ]  Installing C library headers
[EXTRA]    Configuring C library
```

[9] http://www.crifan.com/crosstool_xscale_ct_ng_build_process/

```
[EXTRA]    Installing C library headers
[ERROR]    make[3]: *** [/home/CLi/develop/crosstool-ng/x-tools/arm-xscale-linux-gnueabi/
arm-xscale-linux-gnueabi//sys-root/usr/include/sys/uio.h] Error 1
[ERROR]    make[2]: *** [misc/install-headers] Error 2
[ERROR]    make[1]: *** [install-headers] Error 2
[ERROR]    Build failed in step 'Installing C library headers'
[ERROR]    Error happened in '/opt/crosstool-ng/lib/ct-ng-1.9.3/scripts/functions' in function
 'CT_DoExecLog' (line unknown, sorry)
[ERROR]        called from '/opt/crosstool-ng/lib/ct-ng-1.9.3/scripts/build/libc/glibc.sh' at line
 # 175 in function 'do_libc_headers'
[ERROR]        called from '/opt/crosstool-ng/lib/ct-ng-1.9.3/scripts/crosstool-NG.sh' at line #
 597 in function 'main'
[ERROR]    Look at '/home/CLi/develop/crosstool-ng/x-tools/arm-xscale-linux-gnueabi/
build.log' for more info on this error.
[ERROR]  (elapsed: 76:07.10)
[76:12] / /opt/crosstool-ng/bin/ct-ng:143: recipe for target `build' failed
make: *** [build] Error 2

CLi@PC-CLI-1 ~/develop/crosstool-ng/crosstool-ng-1.9.3_build
$
```

没有给你机会去修复错误，然后继续编译。

而你能做的，做节省时间的方式，也最多只是，参考：第 3.3.1.2 节 “从之前错误的那一步恢复继续编译”去，在解决了问题之后，去：

```
ct-ng libc_headers+
```

从而达到，从出错的步骤，恢复后继续编译的效果，不再重复之前的已经成功编译的那些步骤。

但是，即便是这样，对于编译最开始的部分：

```
CLi@PC-CLI-1 ~/develop/crosstool-ng/crosstool-ng-1.9.3_build
$ ct-ng build
[INFO ]  Performing some trivial sanity checks
[INFO ]  Build started 20130724.093552
[INFO ]  Building environment variables
[EXTRA]  Preparing working directories
```

等步骤，还是需要花点时间的，还是有点重复浪费的感觉。

所以，此处，选中了上面所说的那个"Interactive shell on failed commands"选项后，

当编译出错后，会出现类似于：

【已解决】Ubuntu中crosstool-ng编译出错：scripts/unifdef.c:209:25: error: conflicting types for ‘getline’ [10]

中的这样的效果：

[10] http://www.crifan.com/ubuntu_crosstool_ng_ct_ng_build_error_scripts_unifdef_c_error_conflicting_types_for_getline/

```
[INFO ]
 ==================================================================
[INFO ]  Installing kernel headers
[EXTRA]    Installing kernel headers
[ERROR]    /home/crifan/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/linux-
custom/scripts/unifdef.c:209:25: error: conflicting types for 'getline'
[ERROR]    make[3]: *** [scripts/unifdef] Error 1
[ERROR]    make[2]: *** [headers_install] Error 2
[ERROR]    make[1]: *** [headers_install] Error 2
[ERROR]
[ERROR]  >>
[ERROR]  >>  Build failed in step 'Installing kernel headers'
[ERROR]  >>        called in step '(top-level)'
[ERROR]  >>
[ERROR]  >>  Error happened in: CT_DoExecLog[scripts/functions@257]
[ERROR]  >>        called from: do_kernel_install[scripts/build/kernel/linux.sh@112]
[ERROR]  >>        called from: do_kernel_headers[scripts/build/kernel/linux.sh@91]
[ERROR]  >>        called from: main[scripts/crosstool-NG.sh@632]


Current command:
  'make' '-C' '/home/crifan/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/linux-
custom' 'O=/home/crifan/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-
linux-gnueabi/build/build-kernel-headers' 'ARCH=arm' 'INSTALL_HDR_PATH=/home/crifan/
develop/crosstool-ng/x-tools/arm-xscale-linux-gnueabi/arm-xscale-linux-gnueabi/sysroot/
usr' 'V=0' 'headers_install'
exited with error code: 2
Please fix it up and finish by exiting the shell with one of these values:
    1  fixed, continue with next build command
    2  repeat this build command
    3  abort build

ct-ng:~/develop/crosstool-ng/crosstool-ng-1.18.0_build>
```

即，有对应的

```
Please fix it up and finish by exiting the shell with one of these values
```

那些信息。

然后，当你去解决了对应的问题后：

此处，只是去修改对应的.build/src/linux-custom/scripts/unifdef.c，即可。

然后，就可以通过输入：

```
exit 2
```

就可以实现：

重新执行此处刚才出错的命令，然后继续重新去编译刚才出错的那个.build/src/linux-custom/scripts/unifdef.c文件，

从而实现：

当编译出错时，也不立刻退出crosstool-ng的编译，

允许你去修复对应的问题，然后修复完毕后，可以接着继续编译。

因而实现了：

避免重复的，之前的各个步骤，包括最开始的环境检查等等的时间

真正的，彻底的，最大程度上的，节省你的时间和精力。

注意是exit加上数字，不是直接输入数字

此处，由于是，交互式shell（interactive shell）

所以，当你在修复问题后，想要，继续编译，

则是需要，退出当前交互式shell，回去继续编译的。

而退出shell的话，就需要用到对应的：

```
exit N
```

表示退出，且返回值是N

而不是：直接输入N

3.3.2. crosstool-ng核心配置详解

crosstool-ng中有很多配置参数。

其中一些核心参数，是交叉编码方面，比如搞清楚，和配置正确的参数。

下面就来详细解释这些参数的含义和说明如何配置。

3.3.2.1. crosstool-ng本身相关的重要参数

此处介绍，和crosstool-ng，为了交叉编译，作为工具本身方面的一些常见的，重要的参数的配置。

3.3.2.1.1. 源码包下载保存路径

常见配置如下：

```
Paths and misc options
   (${HOME}/develop/crosstool-ng/src) Local tarballs directory
```

Local tarballs directory的help的说明为：

```
  ┌──────────────────────────────── Local tarballs directory
────────────────────────────────┐
  │ CT_LOCAL_TARBALLS_DIR:                                    │
  │                                                           │
```

```
| If you have previously downloaded the tarballs, enter the PATH where        |
| you stored them here.                                                       |
|                                                                             |
| Symbol: LOCAL_TARBALLS_DIR [=${HOME}/develop/crosstool-ng/src]              |
| Type  : string                                                              |
| Prompt: Local tarballs directory                                            |
|   Defined at config/global/paths.in:5                                       |
|   Depends on: !BACKEND [=n]                                                 |
|   Location:                                                                 |
|     -> Paths and misc options                                               |
```

即，你之前配置好的各种模块，包括C库，GCC，binutils等等所有的东西，下载下来，都放在此文件夹中。

对应的路径配置，可以参考前面介绍的：❸

3.3.2.1.2. 工作路径

所有的东西，都放在其下，包括：

- 编译时所生成的arm-xscale-linux-gnueabi文件夹
- 所有模块解压的源码在其下的src路径下

一般使用默认配置，如下：

```
Paths and misc options
   (${CT_TOP_DIR}/.build) Working directory
```

Working directory的help的说明为：

```
 ┌──────────────────────────────────────── Working directory ─────────────────────────────────────────┐
 | CT_WORK_DIR:                                                                                       |
 |                                                                                                    |
 | Set this to the directory where all build actions will be done.                                    |
 |                                                                                                    |
 | The default is "${CT_TOP_DIR}/.build", and leaving this option                                     |
 | empty will also use the default.                                                                   |
 |                                                                                                    |
 | You should not need to change that, except in one very peculiar                                    |
 | setup:                                                                                             |
 |  - your crosstool-NG source directory is on the network                                            |
 |  - you configured crosstool-NG with --local                                                        |
 | This kind of setup is a pain, as any action involving source file                                  |
 | access would have to go through the wire. In this case, you should                                 |
 | set CT_WORK_DIR to point to a path local to your machine, to avoid                                 |
 | any network overhead.                                                                              |
 |                                                                                                    |
 | Do *NOT* change it if you don't know better.                                                       |
 |                                                                                                    |
```

```
| Symbol: WORK_DIR [=${CT_TOP_DIR}/.build]                              |
| Type  : string                                                        |
| Prompt: Working directory                                             |
|   Defined at config/global/paths.in:31                                |
|   Depends on: !BACKEND [=n]                                           |
|   Location:                                                           |
|     -> Paths and misc options                                         |
```

3.3.2.1.3. 目标安装路径

改为你自己的，对应的x-tools/${CT_TARGET}的路径：

```
Paths and misc options
  (${HOME}/develop/crosstool-ng/x-tools/${CT_TARGET}) Prefix directory
```

Prefix directory的help的说明为：

```
┌──────────────────────────────────────── Prefix directory ─────────────────────────────────────────┐
| CT_PREFIX_DIR:                                                                                    |
|                                                                                                   |
| This is the path the toolchain will run from.                                                     |
|                                                                                                   |
| Symbol: PREFIX_DIR [=${HOME}/develop/crosstool-ng/x-tools/${CT_TARGET}]                           |
| Type  : string                                                                                    |
| Prompt: Prefix directory                                                                          |
|   Defined at config/global/paths.in:52                                                            |
|   Depends on: !BACKEND [=n]                                                                       |
|   Location:                                                                                       |
|     -> Paths and misc options                                                                     |
```

此处对应的x-tools路径，可参考之前的解释：❹

3.3.2.2. 交叉编译相关的核心参数

3.3.2.2.1. crosstool-ng中的配置参数：Architecture level, Emit assembly for CPU, Tune for CPU

crosstool-ng中的Architecture level, Emit assembly for CPU, Tune for CPU，分别对应的是gcc中的-march=xxx, -mcpu=xxx, -mtune=xxx

而这几个参数，是相对最最核心的，最最重要的配置了。

例 3.3. 举例说明Architecture level, Emit assembly for CPU, Tune for CPU如何去配置

比如，我之前的折腾：

【记录】为arm的xscale配置crosstool-ng [11]

中的：

【整理】crosstool中的Architecture level(CT_ARCH_ARCH),Emit assembly for CPU(CT_ARCH_CPU),Tune for CPU(CT_ARCH_TUNE)的值，对于arm的xscale应该是填写什么 [12]

以及后来的折腾：

【整理】crosstool-ng中的Architecture level，Emit assembly for CPU，Tune for CPU对于TQ2440的S3C2440的ARM920T填写何值 [13]

而搞清楚了自己的xscale的CPU所对应的配置是：

- Architecture level=CT_ARCH_ARCH=-march=armv5te
- Emit assembly for CPU=CT_ARCH_CPU=-mcpu=xscale
- Tune for CPU=CT_ARCH_TUNE=-mtune=strongarm110

以及，后来的折腾：

【记录】Cygwin下为arm920t配置crosstool-ng [14]

中的：

【整理】crosstool-ng中的Architecture level，Emit assembly for CPU，Tune for CPU对于TQ2440的S3C2440的ARM920T填写何值 [15]

去搞懂了，S3C2440的对应的配置是：

- Architecture level=CT_ARCH_ARCH=-march=armv4t
- Emit assembly for CPU=CT_ARCH_CPU=-mcpu=arm9tdmi
- Tune for CPU=CT_ARCH_TUNE=-mtune=arm920t

交叉编译时的核心参数

其实，如上所说的这几个参数

对应的，都是gcc中的参数，

即对应的-march=xxx，-mcpu=xxx，-mtune=xxx

是属于，在交叉编译领域，差不多都会遇到的，相对比较通用的，配置参数。

所以，针对这几个参数的详细讲解，专门放在了另外一个关于交叉编译的教程里面，做了极其详尽的解释。

先要搞懂这几个参数的详细含义，可参考对应的内容：

[11] http://www.crifan.com/crosstool_config_for_arm_xscale/
[12] http://www.crifan.com/summary_crosstool_march_mcpu_mtune_for_arm_xscale/
[13] http://www.crifan.com/crosstool_ng_architecture_level_emit_assembly_for_cpu_tune_for_cpu_for_tq2440_s3c2440_arm920t/
[14] http://www.crifan.com/cygwin_crosstool_ng_do_ct_ng_menuconfig_config_for_arm920t/
[15] http://www.crifan.com/crosstool_ng_architecture_level_emit_assembly_for_cpu_tune_for_cpu_for_tq2440_s3c2440_arm920t/

crosstool-ng中交叉编译前缀的命名规则 [16]

3.3.2.2.2. crosstool-ng中的配置参数：MMU

crosstool-ng中，对于MMU的配置，此处如下：

```
Target options
  [*] Use the MMU
```

Use the MMU的help的说明为：

```
 ┌──────────────────────────────────────────── Use the MMU ─────────────────────────────────────────────┐
 │ CT_ARCH_USE_MMU:                                                                                     │
 │                                                                                                      │
 │ If your architecture has an MMU and you want to use it,                                              │
 │ say 'Y' here.                                                                                        │
 │                                                                                                      │
 │ OTOH, if you don't want to use the MMU, or your arch                                                 │
 │ lacks an MMU, say 'N' here.                                                                          │
 │                                                                                                      │
 │ Note that some architectures (eg. ARM) has variants that                                             │
 │ lacks an MMU (eg. ARM Cortex-M3), while other variants                                               │
 │ have one (eg. ARM Cortex-A8).                                                                        │
 │                                                                                                      │
 │ Symbol: ARCH_USE_MMU [=y]                                                                            │
 │ Type  : boolean                                                                                      │
 │ Prompt: Use the MMU                                                                                  │
 │   Defined at config/target.in:87                                                                     │
 │   Depends on: ARCH_SUPPORTS_BOTH_MMU [=y]                                                            │
 │   Location:                                                                                          │
 │     -> Target options                                                                                │
 │   Selected by: ARCH_alpha [=n] && GEN_CHOICE_ARCH [=y] && \                                          │
 │ ARCH_alpha_AVAILABLE [=y] || ARCH_s390 [=n] && GEN_CHOICE_ARCH [=y] && \                             │
 │ ARCH_s390_AVAILABLE [=y] || ARCH_powerpc [=n] && GEN_CHOICE_ARCH [=y] && \                           │
 │ ARCH_powerpc_AVAILABLE [=y] || ARCH_sparc [=n] && GEN_CHOICE_ARCH [=y] && \                          │
 │ ARCH_sparc_AVAILABLE [=y] || ARCH_mips [=n] && GEN_CHOICE_ARCH [=y] && \                             │
 │ ARCH_mips_AVAILABLE [=y] || ARCH_avr32 [=n] && GEN_CHOICE_ARCH [=y] && \                             │
 │ ARCH_avr32_AVAILABLE [=y] || ARCH_sh [=n] && GEN_CHOICE_ARCH [=y] && \                               │
 │ ARCH_sh_AVAILABLE [=y] || ARCH_x86 [=n] && GEN_CHOICE_ARCH [=y] && \                                 │
 │ ARCH_x86_AVAILABLE [=y]                                                                              │
```

此处，已经选上了MMU，是因为之前对于CPU部分，是在：

【记录】为arm的xscale配置crosstool-ng [17]

[16] http://www.crifan.com/files/doc/docbook/cross_compile/release/html/cross_compile.html#crosstool_ng_crosscompiler_naming_rule

[17] http://www.crifan.com/crosstool_config_for_arm_xscale/

设置为xscale的，而xscale的CPU，全都是有MMU的。

所以，此处，crosstool-ng（根据内部已经实现好的依赖关系），已经自动帮你选上MMU了。

需要了解的是，现在多数的CPU，都是已经支持MMU的。

只有之前的很老的CPU，有的才不支持MMU。

3.3.2.2.3. crosstool-ng中的配置参数：大小端Endianness

crosstool-ng中，对于MMU的配置，此处如下：

```
Target options
   Endianness: (Little endian)  --->
```

Endianness的help的说明为：

```
 ┌──────────────────────────────────────────── Endianness:
──────────────────────────────────────────┐
 │ There is no help available for this option.                        │
 │ Prompt: Endianness:                                                │
 │   Defined at config/target.in:112                                  │
 │   Depends on: ARCH_SUPPORTS_BOTH_ENDIAN [=y]                       │
 │   Location:                                                        │
 │     -> Target options                                              │
 │   Selected by: ARCH_SUPPORTS_BOTH_ENDIAN [=y] && m                 │
```

此处，将Endianness设置为小端（Little endian）

一般，多数情况下，都是小端。

不论设置大端（Big Endian）还是小端（Little endian）

都要确保本身CPU是支持此种模式的。

比如，之前，也是经过确认，才知道，xscale，本身是，既支持大端，也支持小端的：

【记录】为arm的xscale配置crosstool-ng [18]

所以，此处设置小端，也才是确保，CPU本身是支持的。

关于大端和小端的区别的，不熟悉的可参考：

大端(Big Endian)与小端(Little Endian)详解 [19]

3.3.2.2.4. crosstool-ng中的配置参数：浮点处理单元FPU

crosstool-ng中的配置中，同样会涉及到，当前的CPU，是否有FPU，浮点处理单元。

[18] http://www.crifan.com/crosstool_config_for_arm_xscale/
[19] http://www.crifan.com/big_endian_big_endian_and_small_end_little_endian_detailed/

一般来说，多数CPU都是没有的。

比如，我之前的折腾：

【记录】Cygwin下为arm920t配置crosstool-ng [20]

就发现，当然的S3C2440dearm920t的话，是不支持FPU的。

所以，此处，就设置为对应的，软件浮点了：

```
Target options
  () Use specific FPU
     Floating point: (softfp (FPU))  --->
```

Use specific FPU的help的说明为：

```
┌──────────────────────────────────────── Use specific FPU
─────────────────────────────────────────┐
│ CT_ARCH_FPU:                                                 │
│                                                              │
│ On some targets (eg. ARM), you can specify the kind of FPU to emit          │
│ code for.                                                    │
│                                                              │
│ This is the configuration flag --with-fpu=XXX, and the runtime flag         │
│ -mfpu=XXX.                                                   │
│                                                              │
│ See below wether to actually emit FP opcodes, or to emulate them.           │
│                                                              │
│ Pick a value from the gcc manual for your choosen gcc version and your      │
│ target CPU.                                                  │
│                                                              │
│ Leave blank if you don't know, or if your target architecture does not      │
│ offer this option.                                           │
│                                                              │
│ Symbol: ARCH_FPU [=]                                         │
│ Type  : string                                               │
│ Prompt: Use specific FPU                                     │
│   Defined at config/target.in:274                            │
│   Depends on: ARCH_SUPPORTS_WITH_FPU [=y]                    │
│   Location:                                                  │
│     -> Target options                                        │
```

Floating point的help的说明为：

```
┌──────────────────────────────────────── Floating point:
─────────────────────────────────────────┐
│ There is no help available for this option.                  │
```

[20] http://www.crifan.com/cygwin_crosstool_ng_do_ct_ng_menuconfig_config_for_arm920t/

```
  | Prompt: Floating point:                                          |
  |   Defined at config/target.in:294                                |
  |   Depends on: ARCH_SUPPORTS_WITH_FLOAT [=y]                      |
  |   Location:                                                      |
  |     -> Target options                                            |
  |   Selected by: ARCH_SUPPORTS_WITH_FLOAT [=y] && m                |
```

可见，此处的FPU，就是对应着gcc中的参数：-mfpu=XXX

对应的，如果是你：

crosstool-NG in Linaro [21]

中所举例的Cortex-A9 with NEON的CPU的话，那么此处的设置就是：

```
Target options
   (neon) Use specific FPU
      Floating point: (hardware (FPU))  --->
```

了，而此时，你也可以看到，配置选择中，又多了个，关于后缀hf的选项，你也可以选择该项：

```
Target options
   [*]   append 'hf' to the tuple (EXPERIMENTAL)
```

append 'hf' to the tuple的help的说明为：

```
  ┌──────────────────────── append 'hf' to the tuple (EXPERIMENTAL)
──────────────────────┐
  | CT_ARCH_ARM_TUPLE_USE_EABIHF:                                    |
  |                                                                  |
  | Is you say 'y' here, then the tuple for the toolchain will end   |
  | up with *eabihf, instead of the usual *eabi.                     |
  |                                                                  |
  | *eabihf is used to denote that the toolchain *is* using the      |
  | hard-float ABI, while *eabi is just an indication of using the   |
  | soft-float ABI.                                                  |
  |                                                                  |
  | Ie. all one can say is:  *eabihf   hard-float ABI                |
  |                                                                  |
  | Saying 'n' here does *not* impact the ability of the toolchain to|
  | generate hard-float instructions with the hard-float ABI. It is a|
  | purely cosmetic thing, used by distros to differentiate their    |
  | hard-float-ABI-using ports from their soft-float-ABI-using ports.|
  | (eg. Debian Wheezy and above).                                   |
```

[21] https://wiki.linaro.org/WorkingGroups/ToolChain/Using/CrosstoolNg

```
|                                                     |
| This is an option, as not all versions of gcc/binutils do support         |
| such tuple, and fail to build with *eabihf. Stock gcc version up          |
| to, and including 4.7.2 have an issue or another with *eabihf.            |
|                                                     |
| This option is here for the future.                           |
|                                                     |
| Say 'n', unless you are trying to fix gcc to properly recognise           |
| the *eabihf tuples.                                       |
|                                                     |
| Symbol: ARCH_ARM_TUPLE_USE_EABIHF [=y]                            |
| Type  : boolean                                       |
| Prompt: append 'hf' to the tuple (EXPERIMENTAL)                   |
|   Defined at config/arch/arm.in.2:64                        |
|   Depends on: ARCH_arm [=y] && ARCH_FLOAT_HW [=y] && ARCH_ARM_EABI [=y]\  |
|  && EXPERIMENTAL [=y]                                       |
|   Location:                                           |
|     -> Target options                                     |
|       -> Use EABI (ARCH_ARM_EABI [=y])                            |
```

3.3.2.2.5. crosstool-ng中的配置参数：CFLAGS参数

crosstool-ng中，支持设置对应的CFLAGS参数，以传递给底层的gcc的。

经过之前的调查：

【整理】crosstool中Target CFLAGS参数的含义和如何设置 [22]

结论是：

一般设置为-O，即可。

```
Target options
   (-O) Target CFLAGS
```

Target CFLAGS的help的说明为：

```
 ┌──────────────────────────────────────── Target CFLAGS
──────────────────────────────────────────┐
| CT_TARGET_CFLAGS:                                 |
|                                                     |
| Used to add specific options when compiling libraries of the toolchain,   |
| that will run on the target (eg. libc.so).                |
|                                                     |
| Note that the options above for ARCH, ABI, CPU, TUNE and FPU will be      |
| automatically used. You don't need to specify them here.          |
|                                                     |
```

[22] http://www.crifan.com/crosstool_gcc_cflags_o/

```
  | Leave blank if you don't know better.                                  |
  |                                                                        |
  | Symbol: TARGET_CFLAGS [=-O]                                            |
  | Type  : string                                                         |
  | Prompt: Target CFLAGS                                                  |
  |   Defined at config/target.in:341                                      |
  |   Location:                                                            |
  |     -> Target options                                                  |
```

3.3.2.2.6. crosstool-ng中的配置参数：Linux内核版本

目前我的理解是：

对于你进行嵌入式开发时，所选择的对应的Linux版本，应该和此处交叉编译器配置的Linux内核版本要一致。

比如，我之前折腾：

【记录】重试使用最新版本1.18.0的crosstool-ng去配置和编译xscale的交叉编译器 [23]

时，去选择了，自定义版本的Linux内核：linux-2.6.19.1.tar.bz2

所以，此处，就先去把对应的，自己的linux-2.6.19.1.tar.bz2，放到对应的路径下：

Cygwin下面的：

```
/home/CLi/develop/crosstool-ng/src/linux-2.6.19.1.tar.bz2
```

而当前自己的HOME路径是：

```
/home/CLi
```

所以就可以去设置为：

```
Operating System
    Linux kernel version (custom tarball or directory)  --->
        custom tarball or directory
    (${HOME}/develop/crosstool-ng/src/linux-2.6.19.1.tar.bz2) Path to custom source, tarball or
 directory
```

Path to custom source, tarball or directory的help的说明为：

```
  ┌──────────────────────────────────── Path to custom source, tarball or directory
──────────────────────────────────────┐
  | CT_KERNEL_LINUX_CUSTOM_LOCATION:                                                 |
  |                                                                                  |
  | Enter here the path to the tarball of your full kernel tree or                   |
  | kernel directory                                                                 |
  |                                                                                  |
```

[23] http://www.crifan.com/reuse_latest_version_crosstool_ng_to_config_and_compile_for_xscale/

```
  | Symbol: KERNEL_LINUX_CUSTOM_LOCATION [=${HOME}/develop/crosstool-ng/src/
linux-2.6.19.1.tar.bz2]   |
  | Type  : string                                                          |
  | Prompt: Path to custom source, tarball or directory                     |
  |   Defined at config/kernel/linux.in:148                                 |
  |   Depends on: <choice> && KERNEL_LINUX_CUSTOM [=y]                      |
  |   Location:                                                             |
  |     -> Operating System                                                 |
  |       -> Linux kernel version (<choice> [=y])                           |
  |         -> custom tarball or directory (KERNEL_LINUX_CUSTOM [=y])       |
```

如此，之后编译的时候，就可以找到你自己设置的Linux内核，并解压，去加载对应的头文件等内容了。

3.3.2.2.7. crosstool-ng中的配置参数：binutils版本

binutils，是编译交叉编译器时，一个比较重要的组件。

对应的版本的选择，一般还是按照之前使用示例配置时，选择的为准。

如果后期编译出错，实在解决不了的时候，再考虑换个其他的版本试试。

我目前的配置，供参考：

```
Binary utilities
    binutils version (2.22)  --->
```

binutils version的help的说明为：

```
┌──────────────────────────────── binutils version ─────────────────────────────┐
│ There is no help available for this option.                                  │
│ Prompt: binutils version                                                     │
│   Defined at config/binutils/binutils.in:5                                   │
│   Depends on: BINUTILS_binutils [=y]                                         │
│   Location:                                                                  │
│     -> Binary utilities                                                      │
│   Selected by: BINUTILS_binutils [=y] && m                                   │
```

3.3.2.2.8. crosstool-ng中的配置参数：gcc版本

crosstool-ng中的gcc的版本选择，也是属于极其重要的参数。

此处我的gcc版本的选择，还是按照之前的逻辑，选用示例配置中的gcc的版本的，此处是4.6.0版本的：

并且，对应着，此处，如果没有特殊需求，则只是选择C++的，即可。

即，剩余那些语言：Fortran，Java，ADA，Objective-C，Objective-C++等，都无需选择。

我此处的配置如下，供参考：

```
C compiler
    C compiler (gcc)  --->
    gcc version (4.6.0)  --->
    [*] C++
    [ ] Fortran
    [ ] Java
    [ ] ADA (EXPERIMENTAL)
    [ ] Objective-C (EXPERIMENTAL)
    [ ] Objective-C++ (EXPERIMENTAL)
    ()  Other languages (EXPERIMENTAL)
```

Linaro版本的gcc

Linaro版本的gcc，crosstool-ng现在也支持了。

所以，对应的可以看到，在之前选上：

```
Paths and misc options
    [*] Try features marked as EXPERIMENTAL
```

之后，此处的gcc中，就可以去选上：

```
C compiler
    [*] Show Linaro versions
```

然后就可以去"gcc version"中，找到有多个Linaro版本的gcc了：

```
|  |                ( ) linaro-4.7-2013.01                |  |
|  |                ( ) 4.7.2                           |  |
|  |                ( ) 4.7.1                           |  |
|  |                ( ) 4.7.0                           |  |
|  |                ( ) linaro-4.6-2013.01                |  |
|  |                ( ) 4.6.3                           |  |
|  |                ( ) 4.6.2                           |  |
|  |                ( ) 4.6.1                           |  |
|  |                (X) 4.6.0                            |  |
|  |                ( ) linaro-4.5-2012.03                |  |
|  |                ( ) 4.5.3                           |  |
|  |                ( ) 4.5.2                           |  |
|  |                ( ) 4.5.1                           |  |
|  |                ( ) 4.5.0                           |  |
|  |                ( ) linaro-4.4-2011.02-0               |  |
```

```
|  |                ( ) 4.4.7                |  |
|  |                ( ) 4.4.6                |  |
|  |                ( ) 4.4.5                |  |
|  |                ( ) 4.4.4                |  |
|  |                ( ) 4.4.3                |  |
|  |                ( ) 4.4.2                |  |
|  |                ( ) 4.4.1                |  |
|  |                ( ) 4.4.0                |  |
|  |                ( ) 4.3.6                |  |
|  |                ( ) 4.3.5                |  |
|  |                ( ) 4.3.4                |  |
|  |                ( ) 4.3.3                |  |
|  |                ( ) 4.3.2                |  |
|  |                ( ) 4.3.1                |  |
|  |                ( ) 4.2.4                |  |
```

然后，如果需要的话，可以去选上某个Linaro版本的gcc，比如：

```
C compiler
    gcc version (linaro-4.7-2013.01)  --->
```

关于Linaro不熟悉的，可参考：

【整理】Linaro简介 [24]

3.3.2.2.9. crosstool-ng中的配置参数：C库的选择

crosstool-ng中，关于C库的选择，也是个重要的问题。

常见的有三种C库：glibc，eglibc，uclibc

不了解的，可参考：【整理】uclibc,eglibc,glibc之间的区别和联系[25]

一般选择glibc或者是eglibc。

我此处，选择的是glibc，对应的版本选择，也是使用之前示例配置中的2.9

配置如下：

```
C-library
    C library (glibc)  --->
    glibc version (2.9)  --->
```

3.3.2.2.10. crosstool-ng中的配置参数：线程模型的选择

线程模型主要分两种：linuxThreads和nptl

[24] http://www.crifan.com/summary_linaro_intro/
[25] http://www.crifan.com/relation_between_uclibc_glibc_eglibc/

相关区别，可参考这个帖子：Linux 线程模型的比较：LinuxThreads 和 NPTL[26]

现在多数都选择，默认的nptl

配置如下：

```
C-library
    Threading implementation to use: (nptl)  --->
```

[26] http://www.ibm.com/developerworks/cn/linux/l-threading.html

第 4 章 编译crosstool-ng：ct-ng build

相关旧帖：【总结】crosstool-ng使用心得和注意事项[1]

在之前配置完毕crosstool-ng后，就可以去编译了。

```
ct-ng build
```

然后，如果顺利的话，过一定的时间，你就可以编译完成，得到对应的交叉编译器了。

ct-ng build总共所耗时间

此处crosstool-ng的编译总体花费时间，取决于你的总体配置，机器性能等因素。

比较老的一些电脑，估计会在3,4个小时，甚至更多的时间。

目前多数电脑的性能，平均在1到2个小时前后。

比如，我之前的一次编译：

【记录】在Ubuntu下用crosstool-ng编译xscale的交叉工具链 [2]

就花费了72分钟。

当ct-ng build期间出问题

当然，编译期间，一般来说，多少都会出现一些问题。

对于一些常见的问题，可以去参考后面所总结的：

第 5.2 节 “crosstool-ng在编译阶段常见问题及解决办法”

例 4.1. 正常输出的log信息

此处，贴上，我之前：

【记录】在Ubuntu下用crosstool-ng编译xscale的交叉工具链 [3]

的成功编译的输出的内容（去除之前错误，合并正常的输出后的结果）：

```
crifan@ubuntu:~/develop/crosstool-ng/crosstool-ng-1.18.0_build$ ct-ng build
[INFO ]  Performing some trivial sanity checks
[INFO ]  Build started 20130807.235739
[INFO ]  Building environment variables
[EXTRA]  Preparing working directories
[EXTRA]  Installing user-supplied crosstool-NG configuration
[EXTRA]
 ================================================================================
[EXTRA]  Dumping internal crosstool-NG configuration
```

[1] http://www.crifan.com/summary_crosstool_ng_note/
[2] http://www.crifan.com/under_ubuntu_use_crosstool_ng_build_xscale_cross_compiler_toolchain/
[3] http://www.crifan.com/under_ubuntu_use_crosstool_ng_build_xscale_cross_compiler_toolchain/

```
[EXTRA]    Building a toolchain for:
[EXTRA]      build  = i686-pc-linux-gnu
[EXTRA]      host   = i686-pc-linux-gnu
[EXTRA]      target = arm-xscale-linux-gnueabi
[EXTRA]  Dumping internal crosstool-NG configuration: done in 0.09s (at 00:02)
[INFO ]
 =================================================================
[INFO ]  Retrieving needed toolchain components' tarballs
[EXTRA]    Using 'linux-custom' from custom location
[INFO ]  Retrieving needed toolchain components' tarballs: done in 0.30s (at 00:02)
[INFO ]
 =================================================================
[INFO ]  Extracting and patching toolchain components
[EXTRA]    Extracting 'linux-custom'
[EXTRA]    Extracting 'gmp-5.0.2'
[EXTRA]    Patching 'gmp-5.0.2'
[EXTRA]    Extracting 'mpfr-3.1.0'
[EXTRA]    Patching 'mpfr-3.1.0'
[EXTRA]    Extracting 'mpc-0.9'
[EXTRA]    Patching 'mpc-0.9'
[EXTRA]    Extracting 'libelf-0.8.13'
[EXTRA]    Patching 'libelf-0.8.13'
[EXTRA]    Extracting 'binutils-2.22'
[EXTRA]    Patching 'binutils-2.22'
[EXTRA]    Extracting 'gcc-4.6.0'
[EXTRA]    Patching 'gcc-4.6.0'
[EXTRA]    Extracting 'glibc-2.9'
[EXTRA]    Patching 'glibc-2.9'
[EXTRA]    Extracting 'glibc-ports-2.9'
[EXTRA]    Patching 'glibc-ports-2.9'
[EXTRA]    Extracting 'dmalloc-5.5.2'
[EXTRA]    Patching 'dmalloc-5.5.2'
[EXTRA]    Extracting 'gdb-6.8a'
[EXTRA]    Patching 'gdb-6.8a'
[EXTRA]    Extracting 'ncurses-5.9'
[EXTRA]    Patching 'ncurses-5.9'
[EXTRA]    Extracting 'expat-2.1.0'
[EXTRA]    Patching 'expat-2.1.0'
[EXTRA]    Extracting 'ltrace-0.5.3'
[EXTRA]    Patching 'ltrace-0.5.3'
[EXTRA]    Extracting 'strace-4.5.19'
[EXTRA]    Patching 'strace-4.5.19'
[INFO ]  Extracting and patching toolchain components: done in 89.30s (at 01:32)
[EXTRA]  Saving state to restart at step 'libc_check_config'...
[EXTRA]  Saving state to restart at step 'companion_libs_for_build'...
[EXTRA]  Saving state to restart at step 'binutils_for_build'...
[EXTRA]  Saving state to restart at step 'companion_libs_for_host'...
[INFO ]
 =================================================================
[INFO ]  Installing GMP for host
[EXTRA]    Configuring GMP
[EXTRA]    Building GMP
[EXTRA]    Installing GMP
[INFO ]  Installing GMP for host: done in 50.61s (at 02:23)
[INFO ]
 =================================================================
[INFO ]  Installing MPFR for host
[EXTRA]    Configuring MPFR
```

```
[EXTRA]    Building MPFR
[EXTRA]    Installing MPFR
[INFO ]  Installing MPFR for host: done in 21.73s (at 02:45)
[INFO ]
 ================================================================
[INFO ]  Installing MPC for host
[EXTRA]    Configuring MPC
[EXTRA]    Building MPC
[EXTRA]    Installing MPC
[INFO ]  Installing MPC for host: done in 8.38s (at 02:53)
[EXTRA]  Saving state to restart at step 'binutils_for_host'...
[INFO ]
 ================================================================
[INFO ]  Installing binutils for host
[EXTRA]    Configuring binutils
[EXTRA]    Building binutils
[EXTRA]    Installing binutils
[INFO ]  Installing binutils for host: done in 71.16s (at 04:05)
[EXTRA]  Saving state to restart at step 'cc_core_pass_1'...
[INFO ]
 ================================================================
[INFO ]  Installing pass-1 core C compiler
[EXTRA]    Configuring gcc
[EXTRA]    Building gcc
...
[INFO ]
 ================================================================
[INFO ]  Installing kernel headers
[EXTRA]    Installing kernel headers
...
[INFO ]  Installing kernel headers: done in 972.34s (at 39:34)
[EXTRA]  Saving state to restart at step 'libc_start_files'...
[INFO ]
 ================================================================
[INFO ]  Installing C library headers & start files
[EXTRA]    Configuring C library
[EXTRA]    Installing C library headers
[EXTRA]    Installing C library start files
[INFO ]  Installing C library headers & start files: done in 23.14s (at 40:02)
[EXTRA]  Saving state to restart at step 'cc_core_pass_2'...
[INFO ]
 ================================================================
[INFO ]  Installing pass-2 core C compiler
[EXTRA]    Configuring gcc
[EXTRA]    Building gcc
[EXTRA]    Installing gcc
[INFO ]  Installing pass-2 core C compiler: done in 297.35s (at 45:04)
[EXTRA]  Saving state to restart at step 'libc'...
[INFO ]
 ================================================================
[INFO ]  Installing C library
[EXTRA]    Configuring C library
[EXTRA]    Building C library
[EXTRA]    Installing C library
[INFO ]  Installing C library: done in 286.37s (at 49:56)
[EXTRA]  Saving state to restart at step 'cc_for_build'...
[EXTRA]  Saving state to restart at step 'cc_for_host'...
```

```
[INFO ]
 ==================================================================
[INFO ]  Installing final compiler
[EXTRA]    Configuring gcc
[EXTRA]    Building gcc
[EXTRA]    Installing gcc
[INFO ]  Installing final compiler: done in 377.10s (at 56:25)
[EXTRA]  Saving state to restart at step 'libelf_for_target'...
[INFO ]
 ==================================================================
[INFO ]  Installing libelf for the target
[EXTRA]    Configuring libelf
[EXTRA]    Building libelf
[EXTRA]    Installing libelf
[INFO ]  Installing libelf for the target: done in 3.99s (at 56:42)
[EXTRA]  Saving state to restart at step 'binutils_for_target'...
[INFO ]
 ==================================================================
[INFO ]  Installing binutils for target
[EXTRA]    Configuring binutils for target
[EXTRA]    Building binutils' libraries (libiberty bfd) for target
[EXTRA]    Installing binutils' libraries (libiberty bfd) for target
[INFO ]  Installing binutils for target: done in 44.91s (at 57:40)
[EXTRA]  Saving state to restart at step 'debug'...
[INFO ]
 ==================================================================
[INFO ]  Installing dmalloc
[EXTRA]    Configuring dmalloc
[EXTRA]    Building dmalloc
[EXTRA]    Installing dmalloc
[INFO ]  Installing dmalloc: done in 6.77s (at 57:59)
[INFO ]
 ==================================================================
[INFO ]  Installing cross-gdb
[EXTRA]    Configuring cross-gdb
[EXTRA]    Building cross-gdb
...
[INFO ]  Installing cross-gdb: done in 687.38s (at 69:26)
[INFO ]
 ==================================================================
[INFO ]  Installing native gdb
[EXTRA]    Building static target ncurses
[EXTRA]    Building static target expat
[EXTRA]    Configuring native gdb
[EXTRA]    Building native gdb
[EXTRA]    Installing native gdb
[INFO ]  Installing native gdb: done in 140.70s (at 71:47)
[INFO ]
 ==================================================================
[INFO ]  Installing gdbserver
[EXTRA]    Configuring gdbserver
[EXTRA]    Building gdbserver
[EXTRA]    Installing gdbserver
[INFO ]  Installing gdbserver: done in 4.65s (at 71:52)
[INFO ]
 ==================================================================
[INFO ]  Installing ltrace
[EXTRA]    Copying sources to build dir
```

```
[EXTRA]    Configuring ltrace
[EXTRA]    Building ltrace
[EXTRA]    Installing ltrace
[INFO ]  Installing ltrace: done in 2.37s (at 71:54)
[INFO ]
 ================================================================================
[INFO ]  Installing strace
[EXTRA]    Configuring strace
[EXTRA]    Building strace
[EXTRA]    Installing strace
[INFO ]  Installing strace: done in 9.33s (at 72:04)
[EXTRA]  Saving state to restart at step 'test_suite'...
[EXTRA]  Saving state to restart at step 'finish'...
[INFO ]
 ================================================================================
[INFO ]  Cleaning-up the toolchain's directory
[INFO ]    Stripping all toolchain executables
[EXTRA]    Installing the populate helper
[EXTRA]    Installing a cross-ldd helper
[EXTRA]    Creating toolchain aliases
[EXTRA]    Removing access to the build system tools
[EXTRA]    Removing installed documentation
[INFO ]  Cleaning-up the toolchain's directory: done in 2.03s (at 72:35)
[INFO ]  Build completed at 20130808.011013
[INFO ]  (elapsed: 72:34.05)
[INFO ]  Finishing installation (may take a few seconds)...
[72:35] / crifan@ubuntu:~/develop/crosstool-ng/crosstool-ng-1.18.0_build$
```

想要在编译的时候输出所有的信息

根据之前第 3.1.1 节 “用ct-ng help查看所拥有的功能”中的介绍，通过加上V=2参数，可以实现，即输入，默认就输出的，人类可读的信息，也输出内部执行的那些命令。

如果有需求的话，为了编译出错时，可以直接方便的看到所正在执行的命令，是哪个出错了。

（就不用去看对应的，当前文件夹下面的build.log了）

那么可以自己去试试，加上V=2的效果，即：

```
ct-ng build V=2
```

的效果。

第 5 章 crosstool-ng常见问题及解决办法

相关旧帖：【总结】crosstool-ng使用心得和注意事项[1]

5.1. 安装crosstool-ng工具本身的阶段常见问题及解决办法

安装crosstool-ng工具本身，也常会遇到一些小问题。

5.1.1. 缺少一些开发相关的工具

比如，在configure和make等时候，提示你缺少一些工具，版本等不满足要求之类的错误。

这时候，基本的逻辑都是：

搞清楚当前缺少的是哪个（版本的）工具，然后去安装上去，即可。

当cygwin下缺失某软件包时请重新安装Cygwin且全选整个Develop那组的所有软件包

注意，如果这些问题，是发生在Cygwin下，那么，则需要你去安装对应的软件包。

其中，比较好的做法是：

在安装Cygwin的时候，对于Developmen那个分类，全部都安装。

详见：

Cygwin详解 [2]

5.1.1.1. configure: error: missing required tool: bison

在Ubuntu下，解决办法是：

```
sudo apt-get install bison
```

详见：【记录】在Ubuntu下用crosstool-ng编译xscale的交叉工具链[3]

5.1.1.2. configure: error: missing required tool: flex

在Ubuntu下，解决办法是：

```
sudo apt-get install flex
```

[1] http://www.crifan.com/summary_crosstool_ng_note/
[2] http://www.crifan.com/files/doc/docbook/cygwin_intro/release/html/cygwin_intro.html
[3] http://www.crifan.com/under_ubuntu_use_crosstool_ng_build_xscale_cross_compiler_toolchain/

详见：【记录】在Ubuntu下用crosstool-ng编译xscale的交叉工具链[4]

5.1.1.3. configure: error: missing required tool: gperf

在Ubuntu下，解决办法是：

```
sudo apt-get install gperf
```

详见：【记录】在Ubuntu下用crosstool-ng编译xscale的交叉工具链[5]

5.1.1.4. configure: error: missing required tool: makeinfo

在Ubuntu下，解决办法是：

```
sudo apt-get install texinfo
```

详见：【已解决】Ubuntu中用sudo apt-get install安装某软件，结果出错：E: Unable to locate package makeinfo[6]

5.1.1.5. configure: error: could not find GNU awk

在Ubuntu下，解决办法是：

```
sudo apt-get install gawk
```

详见：【记录】在Ubuntu下用crosstool-ng编译xscale的交叉工具链[7]

5.1.1.6. configure: error: could not find GNU libtool >= 1.5.26

在Ubuntu下，解决办法是：

```
sudo apt-get install libtool
```

详见：【已解决】Ubuntu下去配置crosstool-ng时出现：configure: error: could not find GNU libtool >= 1.5.26[8]

5.1.1.7. configure: error: could not find GNU automake >= 1.10

在Ubuntu下，解决办法是：

```
sudo apt-get install automake
```

详见：【记录】在Ubuntu下用crosstool-ng编译xscale的交叉工具链[9]

[4] http://www.crifan.com/under_ubuntu_use_crosstool_ng_build_xscale_cross_compiler_toolchain/
[5] http://www.crifan.com/under_ubuntu_use_crosstool_ng_build_xscale_cross_compiler_toolchain/
[6] http://www.crifan.com/ubuntu_apt_get_install_e_unable_to_locate_package/
[7] http://www.crifan.com/under_ubuntu_use_crosstool_ng_build_xscale_cross_compiler_toolchain/
[8] http://www.crifan.com/ubuntu_crosstool_ng_configure_error_could_not_find_gnu_libtool_large_than_1_5_26/
[9] http://www.crifan.com/under_ubuntu_use_crosstool_ng_build_xscale_cross_compiler_toolchain/

5.1.1.8. configure: error: could not find curses header, required for the kconfig frontends

在Ubuntu下，解决办法是：

```
sudo apt-get install libncurses5-dev
```

详见：【记录】在Ubuntu下用crosstool-ng编译xscale的交叉工具链[10]

5.1.2. 在Cygwin下出现的一些问题

除了上述所说的，在Linux以及Cygwin下都可能会出现的一些问题之外，

在Cygwin环境下，还会出现一些，和Cygwin特定相关的一些问题：

5.1.2.1. zconf.tab.o:zconf.tab.c:(.text+0x162a): undefined reference to `_libintl_gettext'

错误现象：

cygwin下make编译crosstool-ng出错：

```
CLi@PC-CLI-1 ~/develop/crosstool-ng/crosstool-ng-1.18.0
$ make
  SED    'ct-ng'
  SED    'scripts/crosstool-NG.sh'
  SED    'scripts/saveSample.sh'
  SED    'scripts/showTuple.sh'
  GEN    'config/configure.in'
  GEN    'paths.mk'
  GEN    'paths.sh'
  DEP    'nconf.gui.dep'
  DEP    'nconf.dep'
  DEP    'lxdialog/checklist.dep'
  DEP    'lxdialog/inputbox.dep'
  DEP    'lxdialog/util.dep'
  DEP    'lxdialog/textbox.dep'
  DEP    'lxdialog/yesno.dep'
  DEP    'lxdialog/menubox.dep'
  DEP    'mconf.dep'
  DEP    'conf.dep'
  BISON  'zconf.tab.c'
  GPERF  'zconf.hash.c'
  LEX    'lex.zconf.c'
  DEP    'zconf.tab.dep'
  CC     'zconf.tab.o'
  CC     'conf.o'
  LD     'conf'
zconf.tab.o:zconf.tab.c:(.text+0x162a): undefined reference to `_libintl_gettext'
zconf.tab.o:zconf.tab.c:(.text+0x47fe): undefined reference to `_libintl_gettext'
```

[10] http://www.crifan.com/under_ubuntu_use_crosstool_ng_build_xscale_cross_compiler_toolchain/

```
zconf.tab.o:zconf.tab.c:(.text+0x56ec): undefined reference to `_libintl_gettext'
zconf.tab.o:zconf.tab.c:(.text+0x58be): undefined reference to `_libintl_gettext'
zconf.tab.o:zconf.tab.c:(.text+0xc70b): undefined reference to `_libintl_gettext'
zconf.tab.o:zconf.tab.c:(.text+0xc719): more undefined references to `_libintl_gettext' follow
/usr/lib/gcc/i686-pc-cygwin/4.5.3/../../../../i686-pc-cygwin/bin/ld: zconf.tab.o: bad reloc
 address 0x110 in section `.data'
/usr/lib/gcc/i686-pc-cygwin/4.5.3/../../../../i686-pc-cygwin/bin/ld: final link failed: Invalid
 operation
collect2: ld returned 1 exit status
Makefile:108: recipe for target `conf' failed
make[2]: *** [conf] Error 1
Makefile:160: recipe for target `build-lib-kconfig' failed
make[1]: *** [build-lib-kconfig] Error 2
Makefile:120: recipe for target `build' failed
make: *** [build] Error 2
```

原因：

Cygwin下虽然已经安装了intl库，但是此处make编译的时候，没有链接进去，所以报错。

解决办法：

修改对应的Makefile文件：

```
crosstool-ng-1.18.0\kconfig\Makefile
```

去添加对应的intl库，即改为：

```
# What’ s needed to build  ‘conf’
conf_SRC = conf.c
conf_OBJ = $(patsubst %.c,%.o,$(conf_SRC))
conf_DEP = $(patsubst %.o,%.dep,$(conf_OBJ))
$(conf_OBJ) $(conf_DEP): CFLAGS += $(INTL_CFLAGS)
conf: LDFLAGS += -lintl

# What’ s needed to build  ‘mconf’
mconf_SRC = mconf.c
mconf_OBJ = $(patsubst %.c,%.o,$(mconf_SRC))
mconf_DEP = $(patsubst %.c,%.dep,$(mconf_SRC))
$(mconf_OBJ) $(mconf_DEP): CFLAGS += $(NCURSES_CFLAGS) $(INTL_CFLAGS)
#mconf: LDFLAGS += $(NCURSES_LDFLAGS)
mconf: LDFLAGS += -lintl $(NCURSES_LDFLAGS)

# What’ s needed to build  ‘nconf’
nconf_SRC = nconf.c nconf.gui.c
nconf_OBJ = $(patsubst %.c,%.o,$(nconf_SRC))
nconf_DEP = $(patsubst %.c,%.dep,$(nconf_SRC))
#$(nconf_OBJ) $(nconf_DEP): CFLAGS += $(INTL_CFLAGS) -I/usr/include/ncurses
#nconf: LDFLAGS += -lmenu -lpanel -lncurses
```

```
$(nconf_OBJ) $(nconf_DEP): CFLAGS += -I/usr/include/ncurses/ $(INTL_CFLAGS)
nconf: LDFLAGS += -lintl -lmenu -lpanel -lncurses
```

此处也同时添加了ncurses库，解决了undefined reference to `_wattrset'的问题

此问题的解决，是在折腾：

【已解决】cygwin中编译crosstool-ng出错：zconf.tab.o:zconf.tab.c:(.text+0x162a): undefined reference to `_libintl_gettext' [11]

时，去参考：

Re: crosstool-NG Cygwin Build Fails [12]

而找到的解决办法。

可以看出，其中，在添加intl的同时，也同时去添加了对应的ncurses那个库。

那是因为，后期，同样会出现类似的错误第 5.1.2.2 节 “lxdialog/menubox.o:menubox.c:(.text+0x7d): undefined reference to `_wattrset'” 的

所以，此处，也加上对应的ncurses库，一并解决了类似的问题。

此处不要通过make时加LDFLAGS参数去添加-lintl和-lcurses

其实，上面贴出来的，修改makefile的办法，的确已经解决了此处的，缺少intl和缺少ncurses的库的两个问题了。

只是，后来，在另外一个Cygwin环境下去折腾同样的make编译crosstool-ng的时候：

【记录】Cygwin下配置编译和安装crosstool-ng [13]

由于之前解决上面这个问题的时候：

【已解决】Cygwin 1.7.17下make编译crosstool-ng出错：zconf.tab.o:zconf.tab.c:(.text+0x162a): undefined reference to `_libintl_gettext' [14]

用的解决办法是：

不去修改makefile，而直接给make加上LDFLAGS参数加上-lintl：

```
make LDFLAGS="-lintl"
```

当时也是可以解决此问题的。

并且，接着去解决了：第 5.1.2.2 节 “lxdialog/menubox.o:menubox.c:(.text+0x7d): undefined reference to `_wattrset'”

用的方法也是，继续给make的LDFLAGS添加对应的-lcurses的参数：

```
make LDFLAGS="-lintl -lcurses"
```

[11] http://www.crifan.com/crosstool_ng_cygwin_zconf_tab_o_zconf_tab_c_text_0x162a_undefined_reference_to_libintl_gettext/
[12] http://sourceware.org/ml/crossgcc/2012-05/msg00012.html
[13] http://www.crifan.com/cygwin_crosstool_ng_configure_make_make_install
[14] http://www.crifan.com/crosstool_ng_cygwin_zconf_tab_o_zconf_tab_c_text_0x162a_undefined_reference_to_libintl_gettext-2/

然后，后来就出现，相对来说，至少到目前为止，没有完全搞懂的问题：

【已解决】Cygwin下make编译crosstool-ng出错：nconf.o:nconf.c:(.text+0×373): undefined reference to `_free_item'[15]

觉得是属于比较诡异的问题。

而最终的解决办法是：

不要用make时给定LDFLAGS参数的方式去添加intl和curses的库

还是用上述的办法，修改对应的makefile，添加对应的intl的库，调整对应的

```
$(INTL_CFLAGS) -I/usr/include/ncurses
```

变成：

```
-I/usr/include/ncurses/ $(INTL_CFLAGS)
```

即可彻底解决此诡异的问题。

在该诡异问题的根本原因没有搞清楚之前，别人如果遇到类似的错误，还是参照此处，修改makefile去解决类似的问题。

5.1.2.2. lxdialog/menubox.o:menubox.c:(.text+0x7d): undefined reference to `_wattrset'

错误现象：

cygwin下make编译crosstool-ng出错：

```
Administrator@PC-20130611GART /home/develop/crosstool-ng/crosstool-ng-1.18.0
$ make LDFLAGS="-lintl"
  GEN    'config/configure.in'
  GEN    'paths.mk'
  GEN    'paths.sh'
  LD     'conf'
  CC     'lxdialog/menubox.o'
  CC     'lxdialog/yesno.o'
  CC     'lxdialog/textbox.o'
  CC     'lxdialog/util.o'
  CC     'lxdialog/inputbox.o'
  CC     'lxdialog/checklist.o'
  CC     'mconf.o'
  LD     'mconf'
```

[15] http://www.crifan.com/crosstool_ng_cygwin_nconf_o_nconf_c_text_0x373_undefined_reference_to_free_item/

```
lxdialog/menubox.o:menubox.c:(.text+0x7d): undefined reference to `_wattrset'
lxdialog/menubox.o:menubox.c:(.text+0x97): undefined reference to `_wmove'
......
mconf.o:mconf.c:(.text+0x13de): undefined reference to `_ncwrap_stdscr'
mconf.o:mconf.c:(.text+0x13e6): undefined reference to `_getcurx'
collect2: ld returned 1 exit status
Makefile:100: recipe for target `mconf' failed
make[2]: *** [mconf] Error 1
Makefile:160: recipe for target `build-lib-kconfig' failed
make[1]: *** [build-lib-kconfig] Error 2
Makefile:120: recipe for target `build' failed
make: *** [build] Error 2

Administrator@PC-20130611GART /home/develop/crosstool-ng/crosstool-ng-1.18.0$
```

原因：

Cygwin下虽然已经安装了ncurses库，但是此处make编译的时候，没有链接进去，所以报错。

解决办法：

参见：第 5.1.2.1 节 “zconf.tab.o:zconf.tab.c:(.text+0x162a): undefined reference to `_libintl_gettext'” 中的此处也同时添加了ncurses库，解决了undefined reference to `_wattrset'的问题，去修改makefile而解决。

注意：参见之前此处不要通过make时加LDFLAGS参数去添加-lintl和-lcurses的解释，不要通过给make加LDFLAGS的方式去解决此问题，而是如上，使用修改makefile去解决此问题。

5.1.2.3. nconf.c:1521:2: error: lvalue required as left operand of assignment

错误现象：

cygwin下make编译crosstool-ng出错：

```
Administrator@PC-20130611GART /home/develop/crosstool-ng/crosstool-ng-1.18.0
$ make LDFLAGS="-lintl -lcurses"
  GEN    'config/configure.in'
  GEN    'paths.mk'
  GEN    'paths.sh'
  LD     'mconf'
  CC     'nconf.o'
nconf.c: In function 'main':
nconf.c:1521:2: error: lvalue required as left operand of assignment
Makefile:95: recipe for target `nconf.o' failed
make[2]: *** [nconf.o] Error 1
Makefile:160: recipe for target `build-lib-kconfig' failed
make[1]: *** [build-lib-kconfig] Error 2
Makefile:120: recipe for target `build' failed
make: *** [build] Error 2
```

原因：

nconf.c中（不知何故）无法找到对应的宏ESCDELAY，导致此处的代码：

```
ESCDELAY = 1;
```

变成一个未知的变量的左赋值，所以报此错误。

解决办法：

把对应的出错的文件：
crosstool-ng-1.18.0\kconfig\nconf.c

修改为：

```
//ESCDELAY = 1;
set_escdelay(1);
```

详见：

【已解决】Cygwin下make编译crosstool-ng出错：nconf.c:1521:2: error: lvalue required as left operand of assignment [16]

5.1.2.4. bash: ct-ng: command not found

之前参考别人的教程：

Cygwin下用crosstool-ng 编译交叉工具链 [17]

去折腾crosstool-ng的时候，在Cygwin下面，遇到一个问题：

同样的去执行对应的命令：

```
echo "PATH=$PATH:/new/added/some/path" >> ~/.bashrc
```

但是echo到.bashrc中的PATH的值，却是展开的。

从而，导致.bashrc中的内容，太乱，以至于无法正常执行。

所以，经过一番折腾，才发现，原来是，在Cygwin中，echo的值，包含环境变量的话，是需要转义的，即写为：

```
echo "PATH=\$PATH:/new/added/some/path" >> ~/.bashrc
```

才可以的。

[16] http://www.crifan.com/crosstool_ng_cygwin_nconf_c_error_lvalue_required_as_left_operand_of_assignment/
[17] http://hi.baidu.com/wanghaishanren/item/22a7976ad8583392c5d24984

【已解决】Cygwin中安装完crosstool-ng后，运行ct-ng help出错：bash: ct-ng: command not found [18]

5.1.2.5. Win7下无法编辑保存Cygwin中的文件

实际上，此问题，不属于此处crosstool-ng方面的问题，只是和cygwin有关。

如果有人遇到类似的问题：

cygwin的文件，在Win7中（通过资源管理器）打开后，用编辑器打开，可以编辑，但是去保存时，无法保存。

则可以去参考Cygwin中的专门的解释：

Cygwin下的文件可以编辑但是无法保存 [19]

5.2. crosstool-ng在编译阶段常见问题及解决办法

5.2.1. 不论何种平台下编译crosstool-ng常见错误及解决办法

不论是在Cygwin还是在Linux类系统中，都常见的错误：

5.2.1.1. scripts/unifdef.c:209:25: error: conflicting types for ‘getline'

错误现象：

cygwin下ct-ng build去编译crosstool-ng出现：

```
CLi@PC-CLI-1 ~/develop/crosstool-ng/crosstool-ng-1.18.0_build
$ ct-ng build
......
[INFO ]
 ================================================================================
[INFO ]  Installing kernel headers
[EXTRA]    Installing kernel headers
[ERROR]    /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/linux-
custom/scripts/unifdef.c:209:25: error: conflicting types for 'getline'
[ERROR]    make[3]: *** [scripts/unifdef] Error 1
[ERROR]    make[2]: *** [headers_install] Error 2
[ERROR]    make[1]: *** [headers_install] Error 2
[ERROR]
[ERROR]  >>
[ERROR]  >>  Build failed in step 'Installing kernel headers'
[ERROR]  >>        called in step '(top-level)'
```

[18] http://www.crifan.com/cygwin_crosstool_ng_ct_ng_help_bash_ct_ng_command_not_found/
[19] http://www.crifan.com/files/doc/docbook/cygwin_intro/release/html/cygwin_intro.html#file_edit_can_not_save

```
[ERROR]  >>
[ERROR]  >>  Error happened in: CT_DoExecLog[scripts/functions@257]
[ERROR]  >>        called from: do_kernel_install[scripts/build/kernel/linux.sh@112]
[ERROR]  >>        called from: do_kernel_headers[scripts/build/kernel/linux.sh@91]
[ERROR]  >>        called from: main[scripts/crosstool-NG.sh@632]
......
```

原因：

Linux源码中的scripts/unifdef.c中的函数：getline，和别处重复定义了。

此问题，算是一个，很普遍遇到的，很早之前就出现的问题了。

解决办法：

把getline随便改个别的名字，即可。比如改为get_line

具体做法：

去修改对应的文件；
crosstool-ng-1.18.0_build\.build\src\linux-custom\scripts\unifdef.c

改为：

```
static void             flushline(bool);
//static Linetype         getline(void);
static Linetype         get_line(void);
static Linetype         ifeval(const char **);

......

/*
 * The driver for the state machine.
 */
static void
process(void)
{
    Linetype lineval;

    for (;;) {
        linenum++;
        //lineval = getline();
        lineval = get_line();
        trans_table[ifstate[depth]][lineval]();
        debug("process %s -> %s depth %d",
            linetype_name[lineval],
            ifstate_name[ifstate[depth]], depth);
    }
}

/*
 * Parse a line and determine its type. We keep the preprocessor line
 * parser state between calls in the global variable linestate, with
```

```
 * help from skipcomment().
 */
static Linetype
//getline(void)
get_line(void)
{
```

即可。

详见：【已解决】crosstool-ng在Installing kernel headers时出错：[ERROR] /xxx/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/linux-custom/scripts/unifdef.c:209:25: error: conflicting types for ‘getline’[20]

5.2.1.2. Build failed in step ‘Extracting and patching toolchain components’ , in function ‘CT_DoExecLog’ (line unknown, sorry)

错误现象：

cygwin下ct-ng build去编译crosstool-ng出现：

```
[ERROR]    Build failed in step 'Extracting and patching toolchain components'
[ERROR]    Error happened in '/opt/crosstool-ng/lib/ct-ng-1.9.3/scripts/functions' in function
 'CT_DoExecLog' (line unknown, sorry)
[ERROR]        called from '/opt/crosstool-ng/lib/ct-ng-1.9.3/scripts/functions' at line # 681 in
 function 'CT_Extract'
[ERROR]        called from '/opt/crosstool-ng/lib/ct-ng-1.9.3/scripts/build/debug/300-gdb.sh'
 at line # 65 in function 'do_debug_gdb_extract'
[ERROR]        called from '/opt/crosstool-ng/lib/ct-ng-1.9.3/scripts/build/debug.sh' at line #
 28 in function 'do_debug_extract'
[ERROR]        called from '/opt/crosstool-ng/lib/ct-ng-1.9.3/scripts/crosstool-NG.sh' at line #
 569 in function 'main'
[ERROR]    Look at '/home/CLi/develop/crosstool-ng/x-tools/arm-xscale-linux-gnueabi/
build.log' for more info on this error.
[ERROR]  (elapsed: 10:01.80)
[10:06] / /opt/crosstool-ng/bin/ct-ng:143: recipe for target `build' failed
make: *** [build] Error 2
```

原因：

此处遇到此问题的原因，好像是：

由于之前下载过程中，由于某种原因，比如网络不稳定等，而导致下载下来的源码包不是完整的

从而导致后续的解压和打补丁出错。

解决办法：

去确定，你当前正在处理，即正在解压和打包的是哪个软件包

[20] http://www.crifan.com/crosstool_ng_error_linux_scripts_unifdef_c_error_conflicting_types_for_getline/

然后去重新下载一个对应的，完整的无误的，该版本的软件包

放到对应的目录下即可。

比如：此处正在处理的是ncurses的5.7版本的源码包

对应的文件是：ncurses-5.7.tar.gz

所以去：

1. 删除

   crosstool-ng\crosstool-ng-1.9.3_build\targets\src

   中的.ncurses-5.7.extracting和ncurses-5.7.tar.gz

2. 删除

   crosstool-ng\crosstool-ng-1.9.3_build\targets\tarballs

   中的ncurses-5.7.tar.gz

3. 拷贝最新的，你刚下载好的，确保是完整的ncurses-5.7.tar.bz2到：

   crosstool-ng\crosstool-ng-1.9.3_build\targets\src

如果你遇到类似问题，参考上面的步骤，去试试，应该就可以解决此类问题了。

详见：【已解决】crosstool编译出错：Build failed in step 'Extracting and patching toolchain components', in function 'CT_DoExecLog' (line unknown, sorry)，at line # 681 in function 'CT_Extract'[21]

5.2.1.3. [libgcc/./_powisf2.o] Error 1，/tmp/cc7Xh6xp.s:21: Error: selected processor does not support ARM mode `fmsr s14,r0'

错误现象：

cygwin下ct-ng build去编译crosstool-ng出现：

```
CLi@PC-CLI-1 ~/develop/crosstool-ng/crosstool-ng-1.18.0_build
$ ct-ng build
......
[INFO ]
 ================================================================================
[INFO ]  Installing pass-1 core C compiler
[EXTRA]    Configuring gcc
[EXTRA]    Building gcc
[ERROR]    make[3]: *** [libgcc/./_powisf2.o] Error 1
[ERROR]    make[3]: *** [libgcc/./_powidf2.o] Error 1
[ERROR]    make[2]: *** [stmp-multilib] Error 2
[ERROR]    make[1]: *** [all-gcc] Error 2
......
```

[21] http://www.crifan.com/crosstool_build_failed_in_step_extracting_and_patching_toolchain_components_in_function_ct_doexeclog_line_unknown_sorry/

对应build.log中的详细内容为：

```
[ALL ]    /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/build/build-cc-core-pass-1/./gcc/xgcc -B/home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/build/build-cc-core-pass-1/./gcc/ -B/home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/buildtools/arm-xscale-linux-gnueabi/bin/ -B/home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/buildtools/arm-xscale-linux-gnueabi/lib/ -isystem /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/buildtools/arm-xscale-linux-gnueabi/include -isystem /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/buildtools/arm-xscale-linux-gnueabi/sys-include -O2  -g -Os -DIN_GCC -DCROSS_COMPILE   -W -Wall -Wwrite-strings -Wstrict-prototypes -Wmissing-prototypes -Wold-style-definition  -isystem ./include  -fPIC  -g   -DIN_LIBGCC2 -D__GCC_FLOAT_NOT_NEEDED  -Dinhibit_libc -I. -I. -I/home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/gcc-4.2.2/gcc -I/home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/gcc-4.2.2/gcc/.  -I/home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/gcc-4.2.2/gcc/../include   -I/home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/gcc-4.2.2/gcc/../libcpp/include  -I/home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/gcc-4.2.2/gcc/../libdecnumber -I../libdecnumber -DL_powixf2 -c /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/gcc-4.2.2/gcc/libgcc2.c -o libgcc/./_powixf2.o
[ALL ]    /tmp/cc7Xh6xp.s: Assembler messages:
[ALL ]    /tmp/cc7Xh6xp.s:21: Error: selected processor does not support ARM mode `fmsr s14,r0'
[ALL ]    /tmp/cc7Xh6xp.s:27: Error: selected processor does not support ARM mode `flds s15,.L16'
[ALL ]    /tmp/cc7Xh6xp.s:28: Error: selected processor does not support ARM mode `fcpyseq s13,s15'
[ALL ]    /tmp/cc7Xh6xp.s:29: Error: selected processor does not support ARM mode `fcpysne s13,s14'
[ALL ]    /tmp/cc7Xh6xp.s:39: Error: selected processor does not support ARM mode `fmuls s14,s14,s14'
[ALL ]    /tmp/cc7Xh6xp.s:41: Error: selected processor does not support ARM mode `fmulsne s13,s13,s14'
[ALL ]    /tmp/cc7Xh6xp.s:49: Error: selected processor does not support ARM mode `fldslt s15,.L16'
[ALL ]    /tmp/cc7Xh6xp.s:50: Error: selected processor does not support ARM mode `fdivslt s13,s15,s13'
[ALL ]    /tmp/cc7Xh6xp.s:52: Error: selected processor does not support ARM mode `fmrs r0,s13'
[ALL ]    libgcc.mk:223: recipe for target `libgcc/./_powisf2.o' failed
[ERROR]    make[3]: *** [libgcc/./_powisf2.o] Error 1
[ALL ]    make[3]: *** Waiting for unfinished jobs....
[ALL ]    /tmp/ccSq3fkb.s: Assembler messages:
[ALL ]    /tmp/ccSq3fkb.s:21: Error: selected processor does not support ARM mode `fmdrr d6,r0,r1'
[ALL ]    /tmp/ccSq3fkb.s:23: Error: selected processor does not support ARM mode `fldd d7,.L16'
[ALL ]    /tmp/ccSq3fkb.s:29: Error: selected processor does not support ARM mode `fcpydeq d5,d7'
[ALL ]    /tmp/ccSq3fkb.s:30: Error: selected processor does not support ARM mode `fcpydne d5,d6'
[ALL ]    /tmp/ccSq3fkb.s:40: Error: selected processor does not support ARM mode `fmuld d6,d6,d6'
```

```
[ALL  ]    /tmp/ccSq3fkb.s:42: Error: selected processor does not support ARM mode `fmuldne
 d5,d5,d6'
[ALL  ]    /tmp/ccSq3fkb.s:50: Error: selected processor does not support ARM mode `flddlt
 d7,.L16'
[ALL  ]    /tmp/ccSq3fkb.s:51: Error: selected processor does not support ARM mode `fdivdlt
 d5,d7,d5'
[ALL  ]    /tmp/ccSq3fkb.s:54: Error: selected processor does not support ARM mode `fmrrd
 r0,r1,d5'
[ALL  ]    libgcc.mk:226: recipe for target `libgcc/./_powidf2.o' failed
[ERROR]    make[3]: *** [libgcc/./_powidf2.o] Error 1
[ALL  ]    make[4]: `crtend.o' is up to date.
[ALL  ]    make[4]: `crtbeginS.o' is up to date.
[ALL  ]    make[4]: `crtendS.o' is up to date.
[ALL  ]    make[4]: `crtbeginT.o' is up to date.
[ALL  ]    make[4]: Leaving directory `/home/CLi/develop/crosstool-ng/crosstool-
ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/build/build-cc-core-pass-1/gcc'
[ALL  ]    make[3]: Leaving directory `/home/CLi/develop/crosstool-ng/crosstool-
ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/build/build-cc-core-pass-1/gcc'
[ALL  ]    Makefile:1540: recipe for target `stmp-multilib' failed
[ERROR]    make[2]: *** [stmp-multilib] Error 2
[ALL  ]    rm gcc.pod
[ALL  ]    make[2]: Leaving directory `/home/CLi/develop/crosstool-ng/crosstool-
ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/build/build-cc-core-pass-1/gcc'
[ALL  ]    Makefile:4314: recipe for target `all-gcc' failed
[ERROR]    make[1]: *** [all-gcc] Error 2
[ALL  ]    make[1]: Leaving directory `/home/CLi/develop/crosstool-ng/crosstool-
ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/build/build-cc-core-pass-1'
[ERROR]
[ERROR]  >>
[ERROR]  >>  Build failed in step 'Installing pass-1 core C compiler'
[ERROR]  >>        called in step '(top-level)'
```

原因：

至今未知

感觉好像是版本不兼容之类的问题。

解决办法：

后来是把gcc从当前的4.2.2，换成了4.6.0，就可以避免此问题了。

详见：

【未解决】crosstool-ng编译arm的xscale时，编译gcc-4.2.4时出错：[libgcc/./_powisf2.o] Error 1，/tmp/cc7Xh6xp.s:21: Error: selected processor does not support ARM mode `fmsr s14,r0' [22]

5.2.1.4. .build/arm-xscale-linux-gnueabi/buildtools/include/cloog/ppl_backend.h:54: undefined reference to `_ppl_finalize'

错误现象：

cygwin下ct-ng build去编译crosstool-ng出现：

[22] http://www.crifan.com/crosstool_ng_gcc_4_2_2_libgcc_powisf2_o_error_1_tmp_cc7xh6xp_s_error_selected_processor_does_not_support_arm_mode_fmsr_s14_r0/

```
CLi@PC-CLI-1 ~/develop/crosstool-ng/crosstool-ng-1.18.0_build
$ ct-ng build
......
[INFO ]
 ================================================================
[INFO ]  Installing pass-1 core C compiler
[EXTRA]    Configuring gcc
[EXTRA]    Building gcc
[ERROR]    make[2]: *** [cc1-dummy.exe] Error 1
[ERROR]    make[1]: *** [all-gcc] Error 2
[ERROR]
[ERROR]  >>
[ERROR]  >>  Build failed in step 'Installing pass-1 core C compiler'
[ERROR]  >>        called in step '(top-level)'
......
```

对应build.log中的详细内容为：

```
[ALL  ]    mv tmp-specs specs
[ALL  ]    : > tmp-libgcc.mvars
[ALL  ]    echo LIB1ASMFUNCS = '_udivsi3 _divsi3 _umodsi3 _modsi3 _bb_init_func
 _call_via_rX _interwork_call_via_rX _lshrdi3 _ashrdi3 _ashldi3 _arm_negdf2 _arm_addsubdf3
 _arm_muldivdf3 _arm_cmpdf2 _arm_unorddf2 _arm_fixdfsi _arm_fixunsdfsi _arm_truncdfsf2
 _arm_negsf2 _arm_addsubsf3 _arm_muldivsf3 _arm_cmpsf2 _arm_unordsf2 _arm_fixsfsi
 _arm_fixunssfsi _arm_floatdidf _arm_floatdisf _arm_floatundidf _arm_floatundisf _clzsi2 _clzdi2
 _aeabi_lcmp _aeabi_ulcmp _aeabi_ldivmod _aeabi_uldivmod _dvmd_lnx' >> tmp-libgcc.mvars
[ALL  ]    echo LIB1ASMSRC = 'arm/lib1funcs.asm' >> tmp-libgcc.mvars
[ALL  ]    echo LIB2FUNCS_ST = '_eprintf __gcc_bcmp' >> tmp-libgcc.mvars
[ALL  ]    libbackend.a(graphite.o): In function `cloog_finalize':
[ALL  ]    /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-
gnueabi/buildtools/include/cloog/ppl_backend.h:54: undefined reference to `_ppl_finalize'
[ALL  ]    /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-
gnueabi/buildtools/lib/libcloog.a(domain.o): In function `cloog_build_ppl_cstr':
[ALL  ]    /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:333: undefined reference to `_ppl_new_Constraint'
[ALL  ]    /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:337: undefined reference to `_ppl_new_Constraint'
[ALL  ]    /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-
gnueabi/buildtools/lib/libcloog.a(domain.o): In function `cloog_translate_constraint':
[ALL  ]    /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:365: undefined reference to `_ppl_new_Coefficient'
[ALL  ]    /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:366: undefined reference to
 `_ppl_new_Linear_Expression_with_dimension'
[ALL  ]    /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:370: undefined reference to
 `_ppl_assign_Coefficient_from_mpz_t'
[ALL  ]    /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:371: undefined reference to
 `_ppl_Linear_Expression_add_to_coefficient'
[ALL  ]    /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:376: undefined reference to
 `_ppl_assign_Coefficient_from_mpz_t'
```

```
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:378: undefined reference to
`_ppl_Linear_Expression_add_to_inhomogeneous'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:379: undefined reference to `_ppl_delete_Coefficient'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:385: undefined reference to `_ppl_delete_Linear_Expression'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-
gnueabi/buildtools/lib/libcloog.a(domain.o): In function `cloog_translate_constraint_matrix':
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:455: undefined reference to
`_ppl_new_C_Polyhedron_from_space_dimension'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-
gnueabi/buildtools/lib/libcloog.a(domain.o): In function `cloog_translate_constraint_matrix_1':
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:444: undefined reference to
`_ppl_Polyhedron_add_constraint'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:445: undefined reference to `_ppl_delete_Constraint'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-
gnueabi/buildtools/lib/libcloog.a(domain.o): In function `cloog_translate_oppose_constraint':
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:407: undefined reference to `_ppl_new_Coefficient'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:408: undefined reference to
`_ppl_new_Linear_Expression_with_dimension'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:413: undefined reference to
`_ppl_assign_Coefficient_from_mpz_t'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:414: undefined reference to
`_ppl_Linear_Expression_add_to_coefficient'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:420: undefined reference to
`_ppl_assign_Coefficient_from_mpz_t'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:421: undefined reference to
`_ppl_Linear_Expression_add_to_inhomogeneous'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:422: undefined reference to `_ppl_delete_Coefficient'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:430: undefined reference to `_ppl_delete_Linear_Expression'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-
gnueabi/buildtools/lib/libcloog.a(domain.o): In function `cloog_initialize':
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:112: undefined reference to `_ppl_initialize'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.echo
 LIB2FUNCS_EXCLUDE = '' >> tmp-libgcc.mvars
[ALL ]  build/src/cloog-ppl-0.15.11/source/ppl/domain.c:118: undefined reference to
`_ppl_restore_pre_PPL_rounding'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:124: undefined reference to `_ppl_set_error_handler'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:130: undefined reference to
`_ppl_io_set_variable_output_function'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-
gnueabi/buildtools/lib/libcloog.a(domain.o): In function `cloog_translate_ppl_polyhedron_1':
```

```
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:511: undefined reference to
`_ppl_Polyhedron_get_minimized_constraints'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:512: undefined reference to
`_ppl_new_Constraint_System_const_iterator'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:513: undefined reference to
`_ppl_new_Constraint_System_const_iterator'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:516: undefined reference to `_ppl_Constraint_System_begin'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:517: undefined reference to `_ppl_Constraint_System_end'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:519: undefined reference to
`_ppl_Constraint_System_const_iterator_increment'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:518: undefined reference to
`_ppl_Constraint_System_const_iterator_equal_test'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:521: undefined reference to
`_ppl_Constraint_System_const_iterator_dereference'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:522: undefined reference to `_ppl_Constraint_type'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:525: undefined reference to
`_ppl_Polyhedron_space_dimension'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:540: undefined reference to `_ppl_Constraint_System_begin'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:540: undefined reference to `_ppl_Constraint_System_end'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:541: undefined reference to
`_ppl_Constraint_System_const_iterator_equal_test'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:548: undefined reference to `_ppl_new_Coefficient'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:549: undefined reference to
`_ppl_Constraint_System_const_iterator_dereference'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:551: undefined reference to `_ppl_Constraint_type'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:553: undefined reference to `_ppl_Constraint_type'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:557: undefined reference to `_ppl_Constraint_coefficient'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:558: undefined reference to `_ppl_Coefficient_to_mpz_t'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:566: undefined reference to
`_ppl_Constraint_inhomogeneous_term'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:567: undefined reference to `_ppl_Coefficient_to_mpz_t'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:569: undefined reference to `_ppl_delete_Coefficient'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:571: undefined reference to `_ppl_Constraint_type'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:590: undefined reference to `_ppl_Constraint_type'
```

```
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:542: undefined reference to
`_ppl_Constraint_System_const_iterator_increment'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:552: undefined reference to `_ppl_Constraint_type'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:597: undefined reference to
`_ppl_delete_Constraint_System_const_iterator'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:598: undefined reference to
`_ppl_delete_Constraint_System_const_iterator'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-
gnueabi/buildtools/lib/libcloog.a(domain.o): In function `cloog_domain_extend':
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:1842: undefined reference to
`_ppl_Polyhedron_add_space_dimensions_and_embed'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:1843: undefined reference to
`_ppl_Polyhedron_map_space_dimensions'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1845: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-
gnueabi/buildtools/lib/libcloog.a(domain.o): In function `cloog_domain_project':
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:1787: undefined reference to
`_ppl_Polyhedron_remove_space_dimensions'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1789: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-
gnueabi/buildtools/lib/libcloog.a(domain.o): In function `cloog_domain_polyhedron_compare':
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1250: undefined reference to `_ppl_Polyhedron_is_empty'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1257: undefined reference to `_ppl_Polyhedron_is_empty'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:1264: undefined reference to
`_ppl_new_C_Polyhedron_from_C_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:1265: undefined reference to
`_ppl_new_C_Polyhedron_from_C_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:1276: undefined reference to
`_ppl_new_Coefficient_from_mpz_t'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:1277: undefined reference to
`_ppl_new_Linear_Expression_with_dimension'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:1278: undefined reference to
`_ppl_Linear_Expression_add_to_coefficient'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1279: undefined reference to `_ppl_new_Generator'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:1280: undefined reference to
`_ppl_Polyhedron_add_generator'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:1281: undefined reference to
`_ppl_Polyhedron_add_generator'
```

[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1282: undefined reference to `_ppl_delete_Generator'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1283: undefined reference to `_ppl_delete_Linear_Expression'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1284: undefined reference to `_ppl_delete_Coefficient'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1287: undefined reference to `_ppl_new_C_Polyhedron_from_C_Polyhedron'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1288: undefined reference to `_ppl_Polyhedron_intersection_assign'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1289: undefined reference to `_ppl_delete_Polyhedron'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1290: undefined reference to `_ppl_delete_Polyhedron'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1292: undefined reference to `_ppl_Polyhedron_is_empty'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1300: undefined reference to `_ppl_new_C_Polyhedron_from_C_Polyhedron'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1301: undefined reference to `_ppl_new_C_Polyhedron_from_C_Polyhedron'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1302: undefined reference to `_ppl_delete_Polyhedron'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1303: undefined reference to `_ppl_delete_Polyhedron'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1305: undefined reference to `_ppl_Polyhedron_intersection_assign'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1306: undefined reference to `_ppl_Polyhedron_intersection_assign'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1311: undefined reference to `_ppl_new_C_Polyhedron_from_C_Polyhedron'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1317: undefined reference to `_ppl_Polyhedron_add_constraint'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1318: undefined reference to `_ppl_delete_Constraint'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1330: undefined reference to `_ppl_Polyhedron_is_empty'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1339: undefined reference to `_ppl_delete_Polyhedron'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1340: undefined reference to `_ppl_new_C_Polyhedron_from_C_Polyhedron'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1341: undefined reference to `_ppl_new_C_Polyhedron_from_C_Polyhedron'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1367: undefined reference to `_ppl_Polyhedron_add_constraint'
[ALL] /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1368: undefined reference to `_ppl_delete_Constraint'

```
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1374: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:1375: undefined reference to
 `_ppl_new_C_Polyhedron_from_C_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:1401: undefined reference to
 `_ppl_Polyhedron_add_constraint'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1402: undefined reference to `_ppl_delete_Constraint'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1408: undefined reference to `_ppl_Polyhedron_is_empty'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1419: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:1420: undefined reference to
 `_ppl_new_C_Polyhedron_from_C_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1424: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:1425: undefined reference to
 `_ppl_new_C_Polyhedron_from_C_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1430: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1431: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1432: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1433: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1434: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-
gnueabi/buildtools/lib/libcloog.a(domain.o):/home/CLi/develop/crosstool-ng/crosstool-
ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1252: more undefined
 references to `_ppl_delete_Polyhedron' follow
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-
gnueabi/buildtools/lib/libcloog.a(domain.o): In function `cloog_domain_polyhedron_compare':
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:1326: undefined reference to
 `_ppl_Polyhedron_add_constraint'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1327: undefined reference to `_ppl_delete_Constraint'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1410: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1411: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1412: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1413: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1414: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-
gnueabi/buildtools/lib/libcloog.a(domain.o):/home/CLi/develop/crosstool-ng/crosstool-
ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1332: more undefined
 references to `_ppl_delete_Polyhedron' follow
```

```
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/buildtools/lib/libcloog.a(domain.o): In function `cloog_domain_convex':
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:813: undefined reference to `_ppl_Polyhedron_upper_bound_assign'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:814: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:820: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/buildtools/lib/libcloog.a(domain.o): In function `cloog_pol_from_matrix':
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:641: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/buildtools/lib/libcloog.a(domain.o): In function `cloog_domain_union':
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1064: undefined reference to `_ppl_Polyhedron_contains_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1070: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1072: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1066: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1072: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1104: undefined reference to `_ppl_Polyhedron_contains_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1110: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1112: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1106: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1112: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/buildtools/lib/libcloog.a(domain.o): In function `cloog_domain_intersection':
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1164: undefined reference to `_ppl_Polyhedron_intersection_assign'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1166: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:1168: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/buildtools/lib/libcloog.a(domain.o): In function `cloog_domain_simplify':
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:903: undefined reference to `_ppl_new_Pointset_Powerset_C_Polyhedron_from_space_dimension'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:909: undefined reference to `_ppl_Pointset_Powerset_C_Polyhedron_add_disjunct'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-ppl-0.15.11/source/ppl/domain.c:910: undefined reference to `_ppl_delete_Polyhedron'
```

```
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:917: undefined reference to
`_ppl_new_Pointset_Powerset_C_Polyhedron_from_space_dimension'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:923: undefined reference to
`_ppl_Pointset_Powerset_C_Polyhedron_add_disjunct'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:924: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:929: undefined reference to
`_ppl_Pointset_Powerset_C_Polyhedron_simplify_using_context_assign'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:933: undefined reference to
`_ppl_new_Pointset_Powerset_C_Polyhedron_const_iterator'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:935: undefined reference to
`_ppl_new_Pointset_Powerset_C_Polyhedron_const_iterator'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:936: undefined reference to
`_ppl_Pointset_Powerset_C_Polyhedron_const_iterator_begin'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:937: undefined reference to
`_ppl_Pointset_Powerset_C_Polyhedron_const_iterator_end'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:942: undefined reference to
`_ppl_Pointset_Powerset_C_Polyhedron_const_iterator_dereference'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:939: undefined reference to
`_ppl_Pointset_Powerset_C_Polyhedron_const_iterator_increment'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:938: undefined reference to
`_ppl_Pointset_Powerset_C_Polyhedron_const_iterator_equal_test'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:947: undefined reference to
`_ppl_delete_Pointset_Powerset_C_Polyhedron_const_iterator'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:948: undefined reference to
`_ppl_delete_Pointset_Powerset_C_Polyhedron_const_iterator'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:949: undefined reference to
`_ppl_delete_Pointset_Powerset_C_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:950: undefined reference to
`_ppl_delete_Pointset_Powerset_C_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-
gnueabi/buildtools/lib/libcloog.a(domain.o): In function `cloog_domain_difference':
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:1207: undefined reference to
`_ppl_Polyhedron_add_constraint'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1208: undefined reference to `_ppl_delete_Constraint'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1210: undefined reference to `_ppl_delete_Polyhedron'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
src/cloog-ppl-0.15.11/source/ppl/domain.c:1217: undefined reference to
`_ppl_Polyhedron_add_constraint'
[ALL ]  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1218: undefined reference to `_ppl_delete_Constraint'
```

```
[ALL  ]    /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/cloog-
ppl-0.15.11/source/ppl/domain.c:1220: undefined reference to `_ppl_delete_Polyhedron'
[ALL  ]    collect2: ld returned 1 exit status
[ALL  ]    Makefile:1666: recipe for target `cc1-dummy.exe' failed
[ERROR]    make[2]: *** [cc1-dummy.exe] Error 1
[ALL  ]    make[2]: *** Waiting for unfinished jobs....
[ALL  ]    echo LIBGCOV = '_gcov _gcov_merge_add _gcov_merge_single _gcov_merge_delta
  _gcov_fork _gcov_execl _gcov_execlp _gcov_execle  _gcov_execv _gcov_execvp
 _gcov_execve  _gcov_interval_profiler _gcov_pow2_profiler _gcov_one_value_profiler
 _gcov_indirect_call_profiler _gcov_average_profiler _gcov_ior_profiler  _gcov_merge_ior' >>
 tmp-libgcc.mvars
[ALL  ]    echo LIB2ADD = '$(gcc_srcdir)/config/arm/bpabi.c $(gcc_srcdir)/config/arm/
unaligned-funcs.c  $(gcc_objdir)/addsf3.c $(gcc_objdir)/divsf3.c $(gcc_objdir)/eqsf2.c
 $(gcc_objdir)/gesf2.c $(gcc_objdir)/lesf2.c $(gcc_objdir)/mulsf3.c $(gcc_objdir)/negsf2.c
 $(gcc_objdir)/subsf3.c $(gcc_objdir)/unordsf2.c $(gcc_objdir)/fixsfsi.c $(gcc_objdir)/floatsisf.c
 $(gcc_objdir)/floatunsisf.c $(gcc_objdir)/adddf3.c $(gcc_objdir)/divdf3.c $(gcc_objdir)/eqdf2.c
 $(gcc_objdir)/gedf2.c $(gcc_objdir)/ledf2.c $(gcc_objdir)/muldf3.c $(gcc_objdir)/negdf2.c
 $(gcc_objdir)/subdf3.c $(gcc_objdir)/unorddf2.c $(gcc_objdir)/fixdfsi.c $(gcc_objdir)/floatsidf.c
 $(gcc_objdir)/floatunsidf.c $(gcc_objdir)/extendsfdf2.c $(gcc_objdir)/truncdfsf2.c' >> tmp-
libgcc.mvars
[ALL  ]    echo LIB2ADD_ST = '$(gcc_srcdir)/config/arm/linux-atomic.c  ' >> tmp-libgcc.mvars
[ALL  ]    echo LIB2ADDEH = '$(gcc_srcdir)/config/arm/unwind-arm.c $(gcc_srcdir)/config/
arm/libunwind.S $(gcc_srcdir)/config/arm/pr-support.c $(gcc_srcdir)/unwind-c.c $(gcc_srcdir)/
emutls.c  ' >> tmp-libgcc.mvars
[ALL  ]    echo LIB2ADDEHSTATIC = '$(gcc_srcdir)/config/arm/unwind-arm.c $(gcc_srcdir)/
config/arm/libunwind.S $(gcc_srcdir)/config/arm/pr-support.c $(gcc_srcdir)/unwind-c.c
 $(gcc_srcdir)/emutls.c  ' >> tmp-libgcc.mvars
[ALL  ]    echo LIB2ADDEHSHARED = '$(gcc_srcdir)/config/arm/unwind-arm.c $(gcc_srcdir)/
config/arm/libunwind.S $(gcc_srcdir)/config/arm/pr-support.c $(gcc_srcdir)/unwind-c.c
 $(gcc_srcdir)/emutls.c  ' >> tmp-libgcc.mvars
[ALL  ]    echo LIB2_SIDITI_CONV_FUNCS = '' >> tmp-libgcc.mvars
[ALL  ]    echo LIBUNWIND = '  ' >> tmp-libgcc.mvars
[ALL  ]    echo SHLIBUNWIND_LINK = '' >> tmp-libgcc.mvars
[ALL  ]    echo SHLIBUNWIND_INSTALL = '' >> tmp-libgcc.mvars
[ALL  ]    echo FPBIT = '' >> tmp-libgcc.mvars
[ALL  ]    echo FPBIT_FUNCS = '_pack_sf _unpack_sf _addsub_sf _mul_sf _div_sf
 _fpcmp_parts_sf _compare_sf _eq_sf _ne_sf _gt_sf _ge_sf  _lt_sf _le_sf _unord_sf _si_to_sf
 _sf_to_si _negate_sf _make_sf  _sf_to_df _sf_to_tf _thenan_sf _sf_to_usi _usi_to_sf' >> tmp-
libgcc.mvars
[ALL  ]    echo LIB2_DIVMOD_FUNCS = '_divdi3 _moddi3 _udivdi3 _umoddi3 _udiv_w_sdiv
 _udivmoddi4' >> tmp-libgcc.mvars
[ALL  ]    echo DPBIT = '' >> tmp-libgcc.mvars
[ALL  ]    echo DPBIT_FUNCS = '_pack_df _unpack_df _addsub_df _mul_df _div_df
 _fpcmp_parts_df _compare_df _eq_df _ne_df _gt_df _ge_df  _lt_df _le_df _unord_df _si_to_df
 _df_to_si _negate_df _make_df  _df_to_sf _df_to_tf _thenan_df _df_to_usi _usi_to_df' >> tmp-
libgcc.mvars
[ALL  ]    echo TPBIT = '' >> tmp-libgcc.mvars
[ALL  ]    echo TPBIT_FUNCS = '_pack_tf _unpack_tf _addsub_tf _mul_tf _div_tf  _fpcmp_parts_tf
 _compare_tf _eq_tf _ne_tf _gt_tf _ge_tf  _lt_tf _le_tf _unord_tf _si_to_tf _tf_to_si _negate_tf
 _make_tf  _tf_to_df _tf_to_sf _thenan_tf _tf_to_usi _usi_to_tf' >> tmp-libgcc.mvars
[ALL  ]    echo DFP_ENABLE = '' >> tmp-libgcc.mvars
[ALL  ]    echo DFP_CFLAGS='' >> tmp-libgcc.mvars
[ALL  ]    echo D32PBIT='' >> tmp-libgcc.mvars
[ALL  ]    echo D32PBIT_FUNCS='_addsub_sd _div_sd _mul_sd _plus_sd _minus_sd  _eq_sd
 _ne_sd _lt_sd _gt_sd _le_sd _ge_sd  _sd_to_si _sd_to_di _sd_to_usi _sd_to_udi  _si_to_sd
 _di_to_sd _usi_to_sd _udi_to_sd  _sd_to_sf _sd_to_df _sd_to_xf _sd_to_tf  _sf_to_sd _df_to_sd
 _xf_to_sd _tf_to_sd  _sd_to_dd _sd_to_td _unord_sd _conv_sd' >> tmp-libgcc.mvars
[ALL  ]    echo D64PBIT='' >> tmp-libgcc.mvars
```

```
[ALL ]    echo D64PBIT_FUNCS='_addsub_dd _div_dd _mul_dd _plus_dd _minus_dd  _eq_dd
 _ne_dd _lt_dd _gt_dd _le_dd _ge_dd  _dd_to_si _dd_to_di _dd_to_usi _dd_to_udi  _si_to_dd
 _di_to_dd _usi_to_dd _udi_to_dd  _dd_to_sf _dd_to_df _dd_to_xf _dd_to_tf  _sf_to_dd _df_to_dd
 _xf_to_dd _tf_to_dd  _dd_to_sd _dd_to_td _unord_dd _conv_dd' >> tmp-libgcc.mvars
[ALL ]    echo D128PBIT='' >> tmp-libgcc.mvars
[ALL ]    echo D128PBIT_FUNCS='_addsub_td _div_td _mul_td _plus_td _minus_td  _eq_td
 _ne_td _lt_td _gt_td _le_td _ge_td  _td_to_si _td_to_di _td_to_usi _td_to_udi  _si_to_td _di_to_td
 _usi_to_td _udi_to_td  _td_to_sf _td_to_df _td_to_xf _td_to_tf  _sf_to_td _df_to_td _xf_to_td
 _tf_to_td  _td_to_sd _td_to_dd _unord_td _conv_td' >> tmp-libgcc.mvars
[ALL ]    echo GCC_EXTRA_PARTS = 'crtbegin.o crtbeginS.o crtbeginT.o crtend.o crtendS.o' >>
 tmp-libgcc.mvars
[ALL ]    echo SHLIB_LINK = '$(GCC_FOR_TARGET) -O2  -g -Os -DIN_GCC -
DCROSS_DIRECTORY_STRUCTURE  -W -Wall -Wwrite-strings -Wstrict-prototypes -
Wmissing-prototypes -Wcast-qual -Wold-style-definition  -isystem ./include  -fPIC -Wno-
missing-prototypes  -g   -DIN_LIBGCC2 -D__GCC_FLOAT_NOT_NEEDED  -Dinhibit_libc
 -shared -nodefaultlibs  -Wl,--soname=@shlib_base_name@.so.1  -Wl,--version-
script=@shlib_map_file@  -o @multilib_dir@/@shlib_base_name@.so.1.tmp @multilib_flags@
  @shlib_objs@ @libgcc_libm@ -lc &&  rm -f @multilib_dir@/@shlib_base_name@.so
 &&  if [ -f @multilib_dir@/@shlib_base_name@.so.1 ]; then  mv -f @multilib_dir@/
@shlib_base_name@.so.1  @multilib_dir@/@shlib_base_name@.so.1.backup;  else true; fi &&
  mv @multilib_dir@/@shlib_base_name@.so.1.tmp @multilib_dir@/@shlib_base_name@.so.1
 &&  (echo "/* GNU ld script";  echo "    Use the shared library, but some functions are only
 in";  echo "    the static library.  */";  echo "GROUP ( @shlib_base_name@.so.1 libgcc.a )"  ) >
 @multilib_dir@/@shlib_base_name@.so' >> tmp-libgcc.mvars
[ALL ]    echo SHLIB_INSTALL = '$(mkinstalldirs) $(DESTDIR)$(slibdir)@shlib_slibdir_qual@;
  /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/tools/bin/install -c -m
 644 @multilib_dir@/@shlib_base_name@.so.1  $(DESTDIR)$(slibdir)@shlib_slibdir_qual@/
@shlib_base_name@.so.1;  rm -f $(DESTDIR)$(slibdir)@shlib_slibdir_qual@/
@shlib_base_name@.so;  /home/CLi/develop/crosstool-ng/crosstool-
ng-1.18.0_build/.build/tools/bin/install -c -m 644 @multilib_dir@/@shlib_base_name@.so
 $(DESTDIR)$(slibdir)@shlib_slibdir_qual@/@shlib_base_name@.so' >> tmp-libgcc.mvars
[ALL ]    echo SHLIB_EXT = '.so' >> tmp-libgcc.mvars
[ALL ]    echo SHLIB_MKMAP = '$(gcc_srcdir)/mkmap-symver.awk  ' >> tmp-libgcc.mvars
[ALL ]    echo SHLIB_MKMAP_OPTS = '' >> tmp-libgcc.mvars
[ALL ]    echo SHLIB_MAPFILES = '$(gcc_srcdir)/libgcc-std.ver $(gcc_srcdir)/config/libgcc-
glibc.ver $(gcc_srcdir)/config/libgcc-glibc.ver $(gcc_srcdir)/config/arm/libgcc-bpabi.ver  ' >>
 tmp-libgcc.mvars
[ALL ]    echo SHLIB_NM_FLAGS = '-pg' >> tmp-libgcc.mvars
[ALL ]    echo LIBGCC2_CFLAGS = '-O2  -g -Os -DIN_GCC -DCROSS_DIRECTORY_STRUCTURE
  -W -Wall -Wwrite-strings -Wstrict-prototypes -Wmissing-prototypes -Wcast-qual -Wold-
style-definition  -isystem ./include  -fPIC -Wno-missing-prototypes  -g   -DIN_LIBGCC2 -
D__GCC_FLOAT_NOT_NEEDED  -Dinhibit_libc' >> tmp-libgcc.mvars
[ALL ]    echo TARGET_LIBGCC2_CFLAGS = '-fPIC -Wno-missing-prototypes' >> tmp-
libgcc.mvars
[ALL ]    echo LIBGCC_SYNC = '' >> tmp-libgcc.mvars
[ALL ]    echo LIBGCC_SYNC_CFLAGS = '' >> tmp-libgcc.mvars
[ALL ]    echo CRTSTUFF_CFLAGS = '-O2 -g -Os -DIN_GCC -DCROSS_DIRECTORY_STRUCTURE
  -W -Wall -Wwrite-strings -Wstrict-prototypes -Wmissing-prototypes -Wcast-qual
 -Wold-style-definition  -isystem ./include  -I. -I. -I/home/CLi/develop/crosstool-ng/
crosstool-ng-1.18.0_build/.build/src/gcc-4.4.6/gcc -I/home/CLi/develop/crosstool-ng/
crosstool-ng-1.18.0_build/.build/src/gcc-4.4.6/gcc/.  -I/home/CLi/develop/crosstool-
ng/crosstool-ng-1.18.0_build/.build/src/gcc-4.4.6/gcc/../include   -I/home/CLi/develop/
crosstool-ng/crosstool-ng-1.18.0_build/.build/src/gcc-4.4.6/gcc/../libcpp/include -I/home/
CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/
buildtools/include -I/home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/
arm-xscale-linux-gnueabi/buildtools/include -I/home/CLi/develop/crosstool-ng/crosstool-
ng-1.18.0_build/.build/src/gcc-4.4.6/gcc/../libdecnumber -I/home/CLi/develop/crosstool-ng/
crosstool-ng-1.18.0_build/.build/src/gcc-4.4.6/gcc/../libdecnumber/dpd -I../libdecnumber   -I/
```

```
home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/
buildtools/include -DCLOOG_PPL_BACKEND   -g0  -finhibit-size-directive -fno-inline-functions
 -fno-exceptions  -fno-zero-initialized-in-bss -fno-toplevel-reorder -fno-tree-vectorize  -
Dinhibit_libc' >> tmp-libgcc.mvars
[ALL  ]    echo CRTSTUFF_T_CFLAGS = '' >> tmp-libgcc.mvars
[ALL  ]    echo CRTSTUFF_T_CFLAGS_S = ' -fPIC' >> tmp-libgcc.mvars
[ALL  ]    mv tmp-libgcc.mvars libgcc.mvars
[ALL  ]    rm gcc.pod
[ALL  ]    make[2]: Leaving directory `/home/CLi/develop/crosstool-ng/crosstool-
ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/build/build-cc-core-pass-1/gcc'
[ALL  ]    Makefile:4872: recipe for target `all-gcc' failed
[ERROR]    make[1]: *** [all-gcc] Error 2
[ALL  ]    make[1]: Leaving directory `/home/CLi/develop/crosstool-ng/crosstool-
ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/build/build-cc-core-pass-1'
[ERROR]
[ERROR]  >>
[ERROR]  >>  Build failed in step 'Installing pass-1 core C compiler'
[ERROR]  >>        called in step '(top-level)'
```

原因：

好像是之前的gcc中的cloog方面的bug

解决办法：

最后是使用 gcc 4.6.0，可以正常编译，可以避免此问题。

详见：

【未解决】cygwin下编译xscale的crosstool-ng时，gcc出错：[cc1-dummy.exe] Error 1 [23]

5.2.1.5. Makefile:240: *** mixed implicit and normal rules. Stop.

错误现象：

cygwin下ct-ng build去编译crosstool-ng出现：

```
CLi@PC-CLI-1 ~/develop/crosstool-ng/crosstool-ng-1.18.0_build
$ ct-ng build
......
[INFO ]  Installing kernel headers: done in 1854.80s (at 72:22)
[EXTRA]  Saving state to restart at step 'libc_start_files'...
[INFO ]
 ================================================================================
[INFO ]  Installing C library headers & start files
[EXTRA]    Configuring C library
[EXTRA]    Installing C library headers
[ERROR]    make[2]: *** [manual/install-headers] Error 2
[ERROR]    make[1]: *** [install-headers] Error 2
[ERROR]
[ERROR]  >>
[ERROR]  >>  Build failed in step 'Installing C library headers & start files'
```

[23] http://www.crifan.com/crosstool_ng_cygwin_gcc_cc1_dummy_exe_error_1_gcc/

```
[ERROR]  >>       called in step '(top-level)'
[ERROR]  >>
[ERROR]  >>  Error happened in: CT_DoExecLog[scripts/functions@257]
[ERROR]  >>       called from: do_libc_backend_once[scripts/build/libc/glibc-eglibc.sh-
common@380]
[ERROR]  >>       called from: do_libc_backend[scripts/build/libc/glibc-eglibc.sh-
common@143]
[ERROR]  >>       called from: do_libc_start_files[scripts/build/libc/glibc-eglibc.sh-
common@60]
[ERROR]  >>       called from: main[scripts/crosstool-NG.sh@632]
......
```

对应build.log中的详细内容为：

```
[ALL  ]    make[3]: Entering directory `/home/CLi/develop/crosstool-ng/crosstool-
ng-1.18.0_build/.build/src/glibc-2.9/gnulib'
[ALL  ]    make[3]: `install-headers' is up to date.
[ALL  ]    make[3]: Leaving directory `/home/CLi/develop/crosstool-ng/crosstool-
ng-1.18.0_build/.build/src/glibc-2.9/gnulib'
[ALL  ]    /usr/bin/make  subdir=wctype -C wctype ..=../ install-headers
[ALL  ]    make[3]: Entering directory `/home/CLi/develop/crosstool-ng/crosstool-
ng-1.18.0_build/.build/src/glibc-2.9/wctype'
[ALL  ]    /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/tools/bin/install -
c -m 644 wctype.h /home/CLi/develop/crosstool-ng/x-tools/arm-xscale-linux-gnueabi/arm-
xscale-linux-gnueabi/sysroot/usr/include/wctype.h
[ALL  ]    make[3]: Leaving directory `/home/CLi/develop/crosstool-ng/crosstool-
ng-1.18.0_build/.build/src/glibc-2.9/wctype'
[ALL  ]    /usr/bin/make  subdir=manual -C manual ..=../ install-headers
[ALL  ]    make[3]: Entering directory `/home/CLi/develop/crosstool-ng/crosstool-
ng-1.18.0_build/.build/src/glibc-2.9/manual'
[ALL  ]    Makefile:240: *** mixed implicit and normal rules.  Stop.
[ALL  ]    make[3]: Leaving directory `/home/CLi/develop/crosstool-ng/crosstool-
ng-1.18.0_build/.build/src/glibc-2.9/manual'
[ALL  ]    Makefile:207: recipe for target `manual/install-headers' failed
[ERROR]    make[2]: *** [manual/install-headers] Error 2
[ALL  ]    make[2]: Leaving directory `/home/CLi/develop/crosstool-ng/crosstool-
ng-1.18.0_build/.build/src/glibc-2.9'
[ALL  ]    Makefile:12: recipe for target `install-headers' failed
[ERROR]    make[1]: *** [install-headers] Error 2
[ALL  ]    make[1]: Leaving directory `/home/CLi/develop/crosstool-ng/crosstool-
ng-1.18.0_build/.build/arm-xscale-linux-gnueabi/build/build-libc-startfiles'
[ERROR]
[ERROR]  >>
[ERROR]  >>  Build failed in step 'Installing C library headers & start files'
[ERROR]  >>       called in step '(top-level)'
[ERROR]  >>
[ERROR]  >>  Error happened in: CT_DoExecLog[scripts/functions@257]
[ERROR]  >>       called from: do_libc_backend_once[scripts/build/libc/glibc-eglibc.sh-
common@380]
[ERROR]  >>       called from: do_libc_backend[scripts/build/libc/glibc-eglibc.sh-
common@143]
[ERROR]  >>       called from: do_libc_start_files[scripts/build/libc/glibc-eglibc.sh-
common@60]
[ERROR]  >>       called from: main[scripts/crosstool-NG.sh@632]
```

原因：

make是3.82的，不识别，旧的，显示和隐式的规则混合在一起的混合规则

所以才报此错误。

解决办法：

而这种makefile中混合规则，老版本的make，即3.81版本的make，是支持的。

所以，总的来说，有两种解决办法：

5.2.1.5.1. 手动修改每个Makefile中对应的混合规则->使得3.82的make可以识别

很明显此办法，的确可以解决此类问题。

比如我之前这里：

【已解决】编译crosstool-ng在Installing kernel headers时出错：/home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/linux-custom/Makefile:1444: *** mixed implicit and normal rules. Stop.[24]

就是这样手工改makefile的。

但是此办法的缺点是：

crosstool-ng项目中，涉及到这种混合规则的Makefile中的写法，有N多个

如果你要是每个都手动改，那就累死了。

并且需要你对Makefile的语法和规则，有足够深入的理解，也才能确保修改为正确的写法

所以：

此法，不适合，本身对于Makefile就不是很熟悉的，大部分的人

更好的办法是，下面要介绍的这个第 5.2.1.5.2 节 “把3.82的make换成3.81版本的make”把此处3.82的make想办法换成3.81版本的make，就可以了。

5.2.1.5.2. 把3.82的make换成3.81版本的make

此法，很明显，属于：一劳永逸的办法

把make换成3.81的，之前的makefile中，N多处的混合规则，就不用手动修改，即可识别，正常编译了。

对应的，把3.82的make，换成3.81的make，其实也有两种办法：

5.2.1.5.2.1. 把你当前的（Linux或Cygwin等）系统环境中的3.82的make换成3.81的make

此法，也是我之前，在Cygwin中折腾crosstool-ng时，去尝试过的：

【已解决】Cygwin下配置Linux内核出错：Makefile:419: *** mixed implicit and normal rules. Stop.[25]

但是很明显，此法，代价太大：

[24] http://www.crifan.com/crosstool_ng_installing_kernel_headers_recipe_for_target_headers_install_failed_build_src_linux_custom/

[25] http://www.crifan.com/cygwin_make_linux_kernel_makefile_mixed_implicit_and_normal_rules_stop/

使得整个系统中的make，都从3.82降级到3.81了。

总的来说，也不算好事。

并且，像是Cygwin中，去更换某个版本的软件包的话，相对来说，还比较麻烦。

而Linux系统，比如Ubuntu中，如果你用的源中没有3.81的make的话，还涉及到先换源

总之，就是很麻烦。

所以，一般来说，都还是用下面介绍的这个更好的办法：

第 5.2.1.5.2.2 节 “用crosstool-ng中提供的3.81的make（而不用更换系统环境中的3.82的make）”

5.2.1.5.2.2. 用crosstool-ng中提供的3.81的make（而不用更换系统环境中的3.82的make）

最终，后来通过：

【已解决】crosstool-ng在Installing C library headers & start files期间出错：Makefile:240: *** mixed implicit and normal rules. Stop.[26]

而去通过更改crosstool-ng中的配置，得以实现：

直接使用crosstool-ng中提供的那个3.81的make

而实现：

不仅解决了此处的问题：

```
Makefile:xxx: *** mixed implicit and normal rules. Stop.
```

而且，无需更改当前系统的3.82的make

也无需去手动更改出错的每一个makefile了。

对应的做法是：

ct-ng menuconfig

去如下配置：

```
Paths and misc options  —>
   [*] Try features marked as EXPERIMENTAL
Companion tools  —>
   [*] Build some companion tools
   [*]   make
```

Try features marked as EXPERIMENTAL对应的help帮助内容为：

```
  ┌──────────────────────────────── Try features marked as EXPERIMENTAL
─────────────────────────────────┐
  │ CT_EXPERIMENTAL:                                                    │
  │                                                                     │
```

[26] http://www.crifan.com/crosstool_ng_makefile_mixed_implicit_and_normal_rules_stop/

```
| If you set this to Y, then you will be able to try very experimental                |
| features.                                                                           |
|                                                                                     |
| Experimental features can be one of:                                                |
|   - working, in which case you should tell me it is!                                |
|   - buggy, in which case you could try patching and send me the result              |
|   - unfinished, in which case you could try hacking it and send me the result       |
|   - non-existant, in which case you could also try hacking it in and send me        |
|     the result                                                                      |
|                                                                                     |
| Symbol: EXPERIMENTAL [=y]                                                           |
| Type  : boolean                                                                     |
| Prompt: Try features marked as EXPERIMENTAL                                         |
|   Defined at config/global/ct-behave.in:18                                          |
|   Location:                                                                         |
|     -> Paths and misc options                                                       |
```

以及：

Build some companion tools对应的help帮助内容为：

```
┌──────────────────────────────────── Build some companion tools ─────────────────────────────────────┐
| CT_COMP_TOOLS:                                                                      |
|                                                                                     |
| Crosstool-NG relies on some external tools to be recent enough, namely:             |
|   make = 3.81 (in some cases)                                                       |
|   m4 >= 1.4.12                                                                      |
|   autoconf >= 2.63                                                                  |
|   automake >= 1.10.2                                                                |
|   libtool >= 2.2.4                                                                  |
|                                                                                     |
| If your system has older versions, we can build them for you,                       |
| but you are strongly encouraged to update your system instead!                      |
|                                                                                     |
| Symbol: COMP_TOOLS [=y]                                                             |
| Type  : boolean                                                                     |
| Prompt: Build some companion tools                                                  |
|   Defined at config/companion_tools.in:8                                            |
|   Depends on: EXPERIMENTAL [=y]                                                     |
|   Location:                                                                         |
|     -> Companion tools                                                              |
```

如此，再去继续重新编译，可以看到对应的编译输出信息：

```
[EXTRA]    Extracting 'make-3.81'
[EXTRA]    Patching 'make-3.81'
[EXTRA]
 ================================================================
[EXTRA]    Installing make
[02:14] |
```

即，去下载并编译对应的3.81的make，供后续使用了。

至此，才算是比较完美的解决此处的问题。

详见：

【未解决】crosstool-ng编译arm的xscale时，编译gcc-4.2.4时出错：[libgcc/./_powisf2.o] Error 1，/tmp/cc7Xh6xp.s:21: Error: selected processor does not support ARM mode `fmsr s14,r0' [27]

5.2.2. Cygwin下编译crosstool-ng常见错误及解决办法

此处是，在Cygwin下才会出现的一些，关于编译crosstol-ng时遇到的问题和解决办法：

5.2.2.1. Cygwin下编译crosstool-ng时导致内存泄露

我之前在：

Win7 x64 + Cygwin 1.7.24

下，去编译crosstool-ng：

【记录】重试使用最新版本1.18.0的crosstool-ng去配置和编译xscale的交叉编译器 [28]

结果，期间出现多次的：

Cannot allocate memory

详见：

【未解决】crosstool-ng编译出错：Installing C library headers中的make[3]: xxx/sys-root/usr/include/sys/uio.h] Error 1 [29]

【未解决】crosstool-ng编译出错：gcc-4.2.2/gcc/regrename.c:1646:12: error: 'IFCVT_ALLOW_MODIFY_TEST_IN_INSN' undeclared (first use in this function) [30]

【未解决】firefox的CPU占用率太高而多次崩溃 [31]

【未解决】cygwin下编译xscale的crosstool-ng时，gcc出错：[cc1-dummy.exe] Error 1 [32]

最终，而推测出的：

[27] http://www.crifan.com/crosstool_ng_gcc_4_2_2_libgcc_powisf2_o_error_1_tmp_cc7xh6xp_s_error_selected_processor_does_not_support_arm_mode_fmsr_s14_r0/
[28] http://www.crifan.com/reuse_latest_version_crosstool_ng_to_config_and_compile_for_xscale
[29] http://www.crifan.com/crosstool_ng_installing_c_library_headers_sys_root_usr_include_sys_uio_h_error_1/
[30] http://www.crifan.com/crosstool_ng_gcc_4_2_2_gcc_regrename_c_error_ifcvt_allow_modify_test_in_insn_undeclared_first_use_in_this_function/
[31] http://www.crifan.com/firefox_crash_and_cpu_high/
[32] http://www.crifan.com/crosstool_ng_cygwin_gcc_cc1_dummy_exe_Error_1_gcc/

【未解决】用cygwin编译crosstool-ng，好像有内存泄露(memory leak) [33]

但是，截止现在，其实也不完全确定：

到底是cygwin中的（比如ld）有bug

还是crosstool-ng对于此处的cygwin不太兼容？

而导致的此处的内存泄露，内存被消耗光的问题。

不过，和此对比的是：

后来又去Ubuntu 13.04中，去编译crosstool-ng：

【记录】Ubuntu下用crosstool-ng为xscale建立交叉编译器arm-xscale-linux-gnueabi-gcc [34]

确定，是没有出现内存占用极高，而导致内存不够的问题。

所以，此处截止目前的结论是：

此处在Win7 x64 + Cygwin 1.7.24，去编译crosstool-ng，出现了内存占用极高

而导致系统卡死，无法继续编译

应该就是：内存泄露

但是原因，到底是cygwin本身的问题，还是crosstool-ng不太兼容cygwin

至今仍未知。

已知的是：crosstool-ng在Ubuntu 13.04中，编译是很顺利的，不会出现此内存泄露的现象。

5.2.2.2. Your file system in ‘xxx/targets’ is *not* case-sensitive!

错误现象：

cygwin下ct-ng build去编译crosstool-ng出现：

```
[ERROR]  Your file system in  '/home/CLi/develop/crosstool-ng/crosstool-ng-1.9.3_build/
targets'  is *not* case-sensitive!
```

原因：

windows系统默认是大小写不分的

而crosstool-ng，一般是在Linux类系统中运行的，是区分大小写的

比如两个文件文件名的字符可能相同，但是大小写不同，就是两个不同文件

而如果Windows中不区分大小写，那么就会导致文件冲突。

所以此处报错，说你的环境是不区分大小写的。

[33] http://www.crifan.com/cygwin_compile_crosstool_ng_may_memory_leak/
[34] http://www.crifan.com/ubuntu_crosstool_ng_cross_compile_for_xscale_arm_xscale_linux_gnueabi/

解决办法：

Windows下，修改注册表：

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\kernel
\obcaseinsensitive
```

将值从1改为0。

详见：【已解决】cygwin下crosstool去build出错：Performing some trivial sanity checks,Your file system in xxx is *not* case-sensitive![35]

已managed的方式挂载cygwin好像也可以解决此问题

对于此问题，之前好像在某处看到说是

以managed的方式去mount cygwin，好像也是可以解决此问题的

目前没去深入研究，只是听说。供参考。

5.2.3. Linux下编译crosstool-ng常见错误及解决办法

此处是，在Linux类系统下才会出现的一些，关于编译crosstol-ng时遇到的问题和解决办法：

5.2.3.1. gcc-4.2.2/gcc/regrename.c:1646:12: error: ‘IFCVT_ALLOW_MODIFY_TEST_IN_INSN’ undeclared (first use in this function)

错误现象：

cygwin下ct-ng build去编译crosstool-ng出现：

```
CLi@PC-CLI-1 ~/develop/crosstool-ng/crosstool-ng-1.18.0_build
$ ct-ng build
[INFO ]  Performing some trivial sanity checks
......
[INFO ]
 ================================================================================
[INFO ]  Installing pass-1 core C compiler
[EXTRA]    Configuring gcc
[EXTRA]    Building gcc
[ERROR]    /home/CLi/develop/crosstool-ng/crosstool-ng-1.18.0_build/.build/src/gcc-4.2.2/
gcc/regrename.c:1646:12: error: 'IFCVT_ALLOW_MODIFY_TEST_IN_INSN' undeclared (first use
 in this function)
[ERROR]    make[2]: *** [regrename.o] Error 1
[ERROR]    make[1]: *** [all-gcc] Error 2
...
```

[35] http://www.crifan.com/crosstool_cygwin_build_error_your_file_system_not_case_sensitive/

原因：

暂未清楚深层次的原因。

解决办法：

去修改对应的文件：
crosstool-ng-1.18.0_build\.build\src\gcc-4.2.2\gcc\regrename.c

去注释掉对应的代码，变成：

```
/* Added for targets (AVR32) which supports test operands to be modified
   in cond_exec instruction. For these targets we cannot make a change to
   the test operands if one of the test operands is an output operand This beacuse
   changing the test operands might cause the need for inserting a new test
   insns in the middle of a sequence of cond_exec insns and if the test operands
   are modified these tests will fail.
*/

// if ( IFCVT_ALLOW_MODIFY_TEST_IN_INSN
     // && predicated )
  // {
    // int insn_skipped = 0;
    // rtx test = COND_EXEC_TEST (PATTERN (insn));

    // /* Check if the previous insn was a skipped predicated insn with the same
       // test as this predicated insns. If so we cannot do any modification to
       // this insn either since we cannot emit the test insn because the operands
       // are clobbered. */
    // if ( prev_pred_insn_skipped
         // && (rtx_equal_p (test, prev_pred_test)
             // || rtx_equal_p (test, reversed_condition (prev_pred_test))) )
      // {
        // insn_skipped = 1;
      // }
    // else
      // {
        // /* Check if the output operand is used in the test expression. */
        // for (i = 0; i < n_ops; ++i)
          // if ( recog_data.operand_type[i] == OP_INOUT
               // && reg_mentioned_p (recog_data.operand[i], test) )
            // {
              // insn_skipped = 1;
              // break;
            // }

      // }

    // prev_pred_test = test;
    // prev_pred_insn_skipped = insn_skipped;
    // if ( insn_skipped )
      // {
        // if (insn == BB_END (bb))
          // break;
        // else
          // continue;
```

```
        // }
      // }
    // else
      // {
        // prev_pred_insn_skipped = 0;
      // }
```

即可。

详见：【未解决】crosstool-ng编译出错：gcc-4.2.2/gcc/regrename.c:1646:12: error: ‘IFCVT_ALLOW_MODIFY_TEST_IN_INSN’ undeclared (first use in this function)[36]

[36] http://www.crifan.com/crosstool_ng_gcc_4_2_2_gcc_regrename_c_error_ifcvt_allow_modify_test_in_insn_undeclared_first_use_in_this_function/

第 6 章 与crosstool-ng相关的一些事项

相关旧帖：【总结】crosstool-ng使用心得和注意事项[1]

6.1. crosstool-ng的作者的主页

后来无意间发现，其实crosstool-ng的作者Yann E. MORIN，简称ymorin，其有个主页的：http://ymorin.is-a-geek.org/

其中，很容易看出：ymorin.is-a-geek，即：ymorin is a geek，翻译为中文即为：ymorin是个极客，即技术上比较能折腾的家伙,^_^

然后，对应该主页中，有此crosstool-ng的主页链接：http://crosstool-ng.org/

然后也可以看到，其也参考了别的一些项目的，比如：BusyBox[2]中的modprobe。

6.2. crosstool-ng支持Linaro的gcc

Linaro，是一个组织，其维护了一堆的基础库等内容，方便其他人使用。包括Ubuntu等Linux的发行版，Android等等。

其所维护的东西，除了基础的gcc库等常见内容之外，还包括此crosstool-ng。

关于如何在crosstool-ng中，使用linaro的gcc，可参考：

Linaro版本的gcc

和：

crosstool-NG in Linaro [3]

6.3. Buildroot支持crosstool-ng

Buildroot，默认是使用自己的交叉编译器，去制作整个Linux系统，包括编译kernel，编译uboot，制作rootfs的。

而最新的Buildroot，已经支持，使用第三方的交叉编译器，去制作整个Linux系统。

其中，就已经支持了，用crosstool-ng的交叉编译器。

详见Buildroot中的配置说明：

Buildroot [4]

[1] http://www.crifan.com/summary_crosstool_ng_note/
[2] http://www.busybox.net/
[3] https://wiki.linaro.org/WorkingGroups/ToolChain/Using/CrosstoolNg
[4] http://www.crifan.com/files/doc/docbook/buildroot_intro/release/html/buildroot_intro.html

参考书目

[1] crosstool[1]

[2] crosstool-ng[2]

[3] Refering to crosstool-NG[3]

[4] Using a released version[4]

[5] crosstool-NG: Download and usage[5]

[6] http://ymorin.is-a-geek.org/

[7] crosstool-NG in Linaro[6]

[8] BusyBox[7]

[9] Cygwin下用crosstool-ng 编译交叉工具链[8]

[10] 【总结】crosstool-ng使用心得和注意事项[9]

[11] 交叉编译详解[10]

[12] 借助别人的工具去制作交叉编译器[11]

[13] crosstool[12]

[14] 【整理】make config，make menuconfig，make oldconfig，make xconfig，make defconfig，make gconfig[13]

[15] 软件开发基础之linux下安装某个软件的逻辑和过程[14]

[16] 【记录】在Cygwin下编译gcc-3.4.5-glibc-2.3.6的arm-xscale-linux-gnu交叉编译器[15]

[17] 【未解决】crosstool-ng编译出错：Installing C library headers中的make[3]: xxx/sys-root/usr/include/sys/uio.h] Error 1[16]

[18] 【记录】重试使用最新版本1.18.0的crosstool-ng去配置和编译xscale的交叉编译器[17]

[19] 【已解决】total commander里面的文件名是中文加数字而无法正常排序[18]

[20] 【记录】在Ubuntu下用crosstool-ng编译xscale的交叉工具链[19]

[1] http://kegel.com/crosstool/
[2] http://crosstool-ng.org/
[3] http://crosstool-ng.org/#refering_to_crosstool-ng
[4] http://crosstool-ng.org/#using_a_released_version
[5] http://crosstool-ng.org/#download_and_usage
[6] https://wiki.linaro.org/WorkingGroups/ToolChain/Using/CrosstoolNg
[7] http://www.busybox.net/
[8] http://hi.baidu.com/wanghaishanren/item/22a7976ad8583392c5d24984
[9] http://www.crifan.com/summary_crosstool_ng_note/
[10] http://www.crifan.com/files/doc/docbook/cross_compile/release/html/cross_compile.html
[11] http://www.crifan.com/files/doc/docbook/cross_compile/release/html/cross_compile.html#use_tool_build_crosscompiler
[12] http://www.crifan.com/files/doc/docbook/cross_compile/release/html/cross_compile.html#crosscompile_tool_crosstool
[13] http://www.crifan.com/make_config_make_menuconfig_make_oldconfig_make_xconfig_make_defconfig/
[14] http://www.crifan.com/soft_dev_basic_linux_install_software/
[15] http://www.crifan.com/cygwin_compile_gcc_3-4-5_glibc_2-3-6_arm_xscale_linux_gnu_cross_compiler/
[16] http://www.crifan.com/crosstool_ng_installing_c_library_headers_sys_root_usr_include_sys_uio_h_error_1/
[17] http://www.crifan.com/reuse_latest_version_crosstool_ng_to_config_and_compile_for_xscale/
[18] http://www.crifan.com/total_commander_cn_name_plus_number_can_not_sort/
[19] http://www.crifan.com/under_ubuntu_use_crosstool_ng_build_xscale_cross_compiler_toolchain/

[21] 【已解决】Cygwin中安装完crosstool-ng后，运行ct-ng help出错：bash: ct-ng: command not found[20]

[22] 【记录】Cygwin下为arm920t配置crosstool-ng[21]

[23] 【整理】Linux下的make命令使用心得[22]

[24] 【已解决】crosstool-ng在Installing C library headers & start files期间出错：Makefile:240: *** mixed implicit and normal rules. Stop.[23]

[25] 【记录】crosstool为xscale编译(ct-ng build)过程[24]

[26] 【已解决】Ubuntu中crosstool-ng编译出错：scripts/unifdef.c:209:25: error: conflicting types for ‘getline’[25]

[27] 【记录】为arm的xscale配置crosstool-ng[26]

[28] 【整理】crosstool中的Architecture level(CT_ARCH_ARCH),Emit assembly for CPU(CT_ARCH_CPU),Tune for CPU(CT_ARCH_TUNE)的值，对于arm的xscale应该是填写什么[27]

[29] 【整理】crosstool-ng中的Architecture level，Emit assembly for CPU，Tune for CPU对于TQ2440的S3C2440的ARM920T填写何值[28]

[30] crosstool-ng中交叉编译前缀的命名规则[29]

[31] 大端(Big Endian)与小端(Little Endian)详解[30]

[32] 【整理】crosstool中Target CFLAGS参数的含义和如何设置[31]

[33] Cygwin详解[32]

[34] 【已解决】Ubuntu中用sudo apt-get install安装某软件，结果出错：E: Unable to locate package makeinfo[33]

[35] 【已解决】Ubuntu下去配置crosstool-ng时出现：configure: error: could not find GNU libtool >= 1.5.26[34]

[36] 【已解决】cygwin下crosstool去build出错：Performing some trivial sanity checks,Your file system in xxx is *not* case-sensitive![35]

[37] Buildroot[36]

[38] 【整理】Linaro简介[37]

[20] http://www.crifan.com/cygwin_crosstool_ng_ct_ng_help_bash_ct_ng_command_not_found/
[21] http://www.crifan.com/cygwin_crosstool_ng_do_ct_ng_menuconfig_config_for_arm920t/
[22] http://www.crifan.com/summary_usage_about_make_linux_command/
[23] http://www.crifan.com/crosstool_ng_makefile_mixed_implicit_and_normal_rules_stop/
[24] http://www.crifan.com/crosstool_xscale_ct_ng_build_process/
[25] http://www.crifan.com/ubuntu_crosstool_ng_ct_ng_build_error_scripts_unifdef_c_error_conflicting_types_for_getline/
[26] http://www.crifan.com/crosstool_config_for_arm_xscale/
[27] http://www.crifan.com/summary_crosstool_march_mcpu_mtune_for_arm_xscale/
[28] http://www.crifan.com/crosstool_ng_architecture_level_emit_assembly_for_cpu_tune_for_cpu_for_tq2440_s3c2440_arm920t/
[29] http://www.crifan.com/files/doc/docbook/cross_compile/release/html/cross_compile.html#crosstool_ng_crosscompiler_naming_rule
[30] http://www.crifan.com/big_endian_big_endian_and_small_end_little_endian_detailed/
[31] http://www.crifan.com/crosstool_gcc_cflags_o/
[32] http://www.crifan.com/files/doc/docbook/cygwin_intro/release/html/cygwin_intro.html
[33] http://www.crifan.com/ubuntu_apt_get_install_e_unable_to_locate_package/
[34] http://www.crifan.com/ubuntu_crosstool_ng_configure_error_could_not_find_gnu_libtool_large_than_1_5_26/
[35] http://www.crifan.com/crosstool_cygwin_build_error_your_file_system_not_case_sensitive/
[36] http://www.crifan.com/files/doc/docbook/buildroot_intro/release/html/buildroot_intro.html
[37] http://www.crifan.com/summary_linaro_intro/

[39] 【整理】crosstool和crosstool-ng[38]

[38] http://www.crifan.com/crosstool_vs_crosstool_ng/