1. Requirements

1.1 Hardware

1.1.1 CPU: Intel 2.5G I5 x64;

1.1.2 Memory: 16G DDR3;

1.1.3 Disk: 30GB;

1.1.4 Internet connected;

1.2 Software

1.2.1 OS: Ubuntu 12.04(64 bit);

1.2.2 Java JDK 6;

1.2.3 Eclipse;

1.2.4 ADT;

1.2.5 CDT;

1.2.6 Android SDK;

1.2.7 Android NDK;

1.2.8 Linux Source;

1.2.9 Android Source.

2. Initializing a Build Environment

Attention: Demonstration is in Ubuntu 12.04(32 bit).

2.1 Installing the JDK

The *Sun JDK* is no longer in Ubuntu's main package repository. In order to download it, you need to add the appropriate repository and indicate to the system which JDK should be used.

Input the following command in the Terminal:

```
$ sudo gedit /etc/apt/sources.list
```

Add:

```
$ deb http://us.archive.ubuntu.com/ubuntu/ hardy multiverse
```

Input in terminal:

```
$ sudo add-apt-repository "deb http://archive.canonical.com/ lucid partner"
$ sudo apt-get update
$ sudo apt-get install sun-java6-jdk
```

2.2 Installing the Android SDK

Download the SDK from Web Browser:

http://developer.android.com/sdk/index.html

Choose the *adt-bundle-linux-x86-20130219.zip*.

Linux 32-bit	adt-bundle-linux-x86-20130219.zip	418664018 bytes	e56ebb5c8eb84eb3227cf7c255373f4b

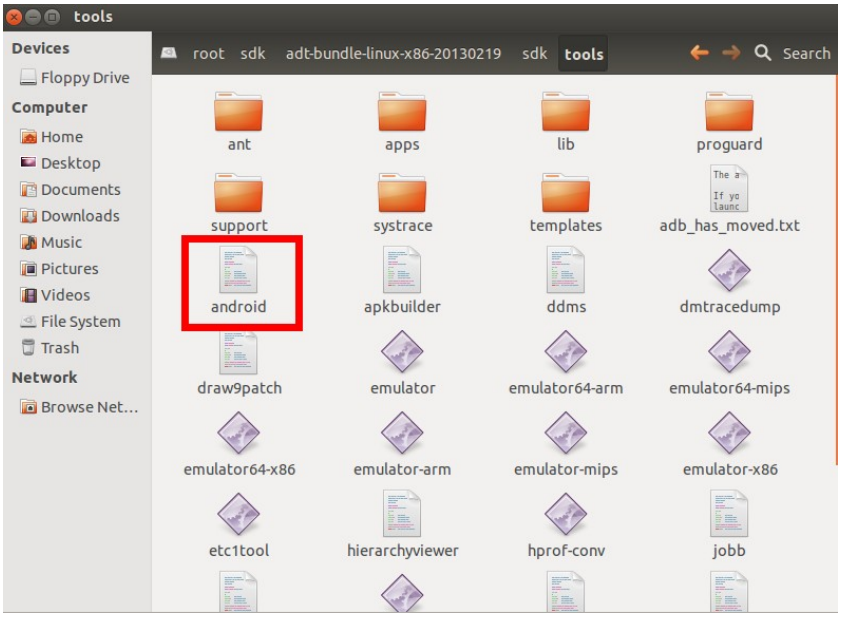
Input in the terminal:

```
$ sudo mkdir ~/sdk
```

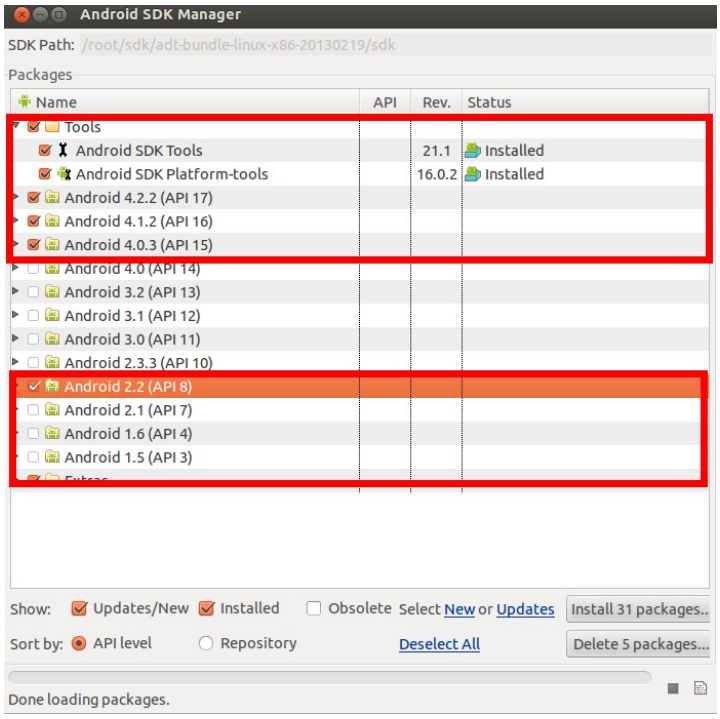
Copy the *adt-bundle-linux-x86-20130219.zip* to the */root/sdk* and extract the zip.

Enter the folder */root/sdk/adt-bundle-linux-x86-20130219/sdk/tools*.

Double click the script file *android*.



It will show the *Android SDK Manager*. Choose the package with ticks.



Then click the *Install XX packages* button on the right. It will start download the required packages.

After the installation, add the SDK tools to *PATH Environment Variable*.

Input in the terminal:

```
$ gedit /etc/profile
```

Input in the “profile”

```
$ export ANDROID_SDK_PATH=/root/sdk/adt-bundle-linux-x86-20130219/sdk
$ export PATH=.:$ANDROID_SDK_PATH/platform-tools:$PATH
```

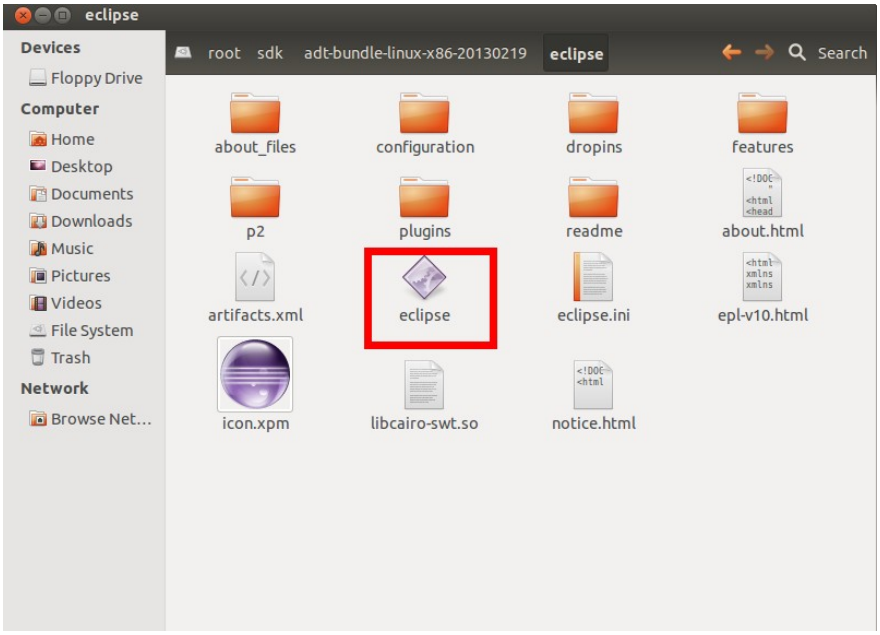
Save and close the *profile*.

Input in the terminal:

$ source /etc/profile

2.2 Installing the Eclipse

With the download of *adt-bundle-linux-x86-20130219.zip*, we can find the *Eclipse* in the */root/sdk/adt-bundle-linux-x86-20130219/eclipse*.



Double click the *eclipse* can run it.

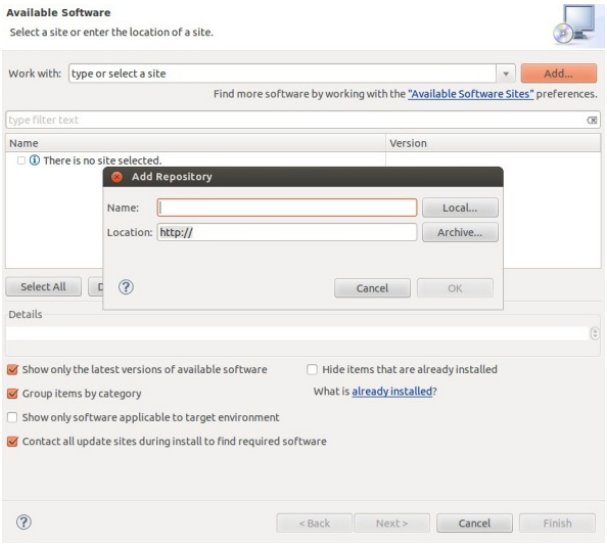
2.2.1 Installing the ADT online

Android Development Tools (ADT) is a plugin for the Eclipse IDE that is designed to give you a powerful, integrated environment in which to build Android applications.

ADT extends the capabilities of Eclipse to let you quickly set up new Android projects, create an application UI, add packages based on the *Android Framework API*, debug your applications using the Android SDK tools, and even export signed (or unsigned) .apk files in order to distribute your application.

Run the eclipse and click the *help>Install New Software.*

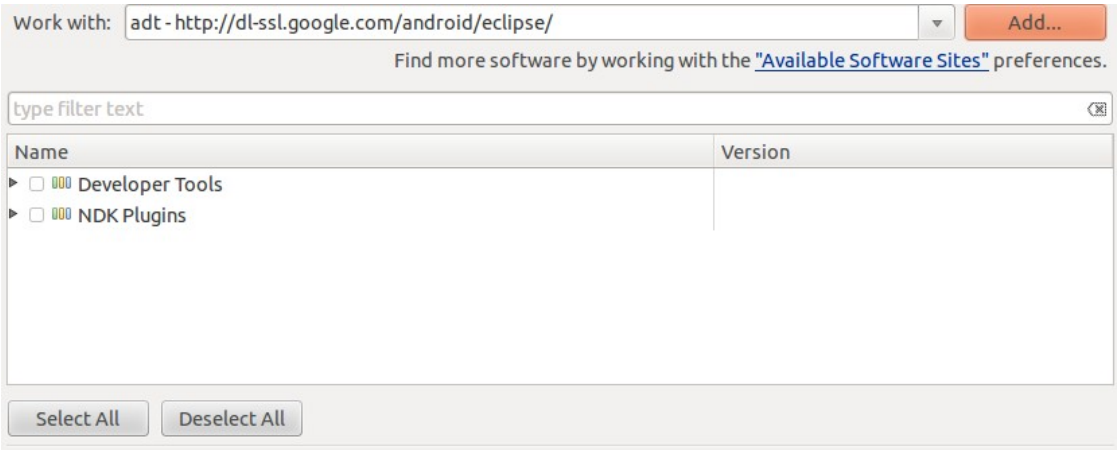
Then click the *Add* button.



Input the Name *ADT*.

Input the Location *https://dl-ssl.google.com/android/eclipse*.

Click *OK*. Then it will show the choice for installation.



Click *Select All* and *Next*.

2.2.2 Installing the NDK

The NDK is a toolset that allows you to implement parts of your app using native-code languages such as C and C++. For certain types of apps, this can be helpful so you can reuse existing code libraries written in these languages, but most apps do not need the Android NDK.

Download the NDK from Web Browser:

http://developer.android.com/sdk/ndk/index.html

Choose the *android-ndk-r8e-linux-x86.tar.bz2*.

Linux 32-bit (x86)	android-ndk-r8e-linux-x86.tar.bz2	461526099 bytes	26d774b0884bcd98de08eb4de41ab532

Input in the terminal:

```
$ sudo mkdir ~/ndk
```

Copy the *android-ndk-r8e-linux-x86.tar.bz2* to the */root/sdk* and extract the zip.

Input in the terminal:

```
$tar –xvf android-ndk-r8e-linux-x86.tar.bz2
```

Finally, add the NDK folder to *PATH Environment Variable*.

2.2.3 Installing the CDT

The CDT Project provides a fully functional C and C++ Integrated Development Environment based on the Eclipse platform. Features include: support for project creation and managed build for various toolchains, standard make build, source navigation, various source knowledge tools, such as type hierarchy, call graph, include browser, macro definition browser, code editor with syntax highlighting, folding and hyperlink navigation, source code refactoring and code generation, visual debugging tools, including memory, registers, and disassembly viewers.

The installation of CDT is the same as ADT.

But the Location of CDT is:

http://download.eclipse.org/tools/cdt/releases/indigo

2.3 Installing the Cross Compiling Environment

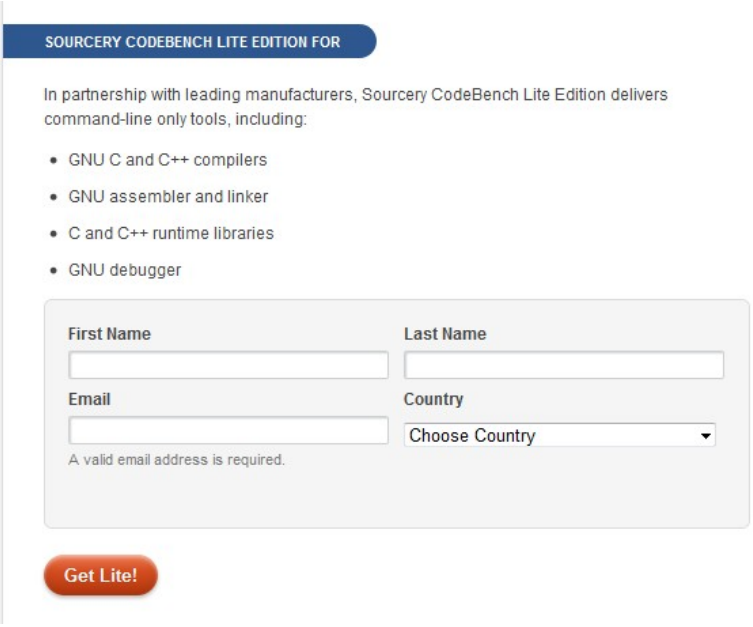
The CPU X86 is using the CISC, while the ARM processor is RISO. Therefore, the cross compiling environment should be installed.

The CodeSourcery is an good choice for the cross compiling. It can be download from the following website.

http://www.mentor.com/embedded-software/sourcery-tools/sourcery-codebench/editions/lite-edition/

Choose the *ARM Processor GNU/Linux release*.

It will need your personal information. Finally, system will send you an Email with the download URL.



Click the *Download Sourcery CodeBench Lite 2012.09-64*.

Choose the *IA32 GNU/Linux Installer.*

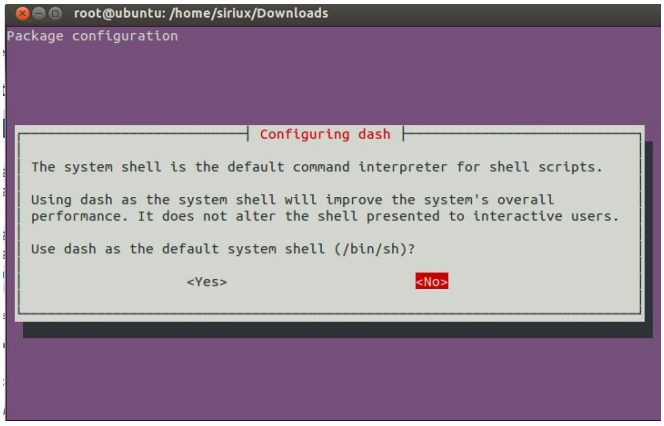
Download	MD5 Checksum
Recommended Packages	
IA32 GNU/Linux Installer	c6b76a7214e3eb404e6d8dadf4b2aab3
IA32 Windows Installer	20583b80d0be222a41615d75fe776039
Advanced Packages	
IA32 GNU/Linux TAR	b5c21a5b546a27912c9f7fd8ac9b4729
IA32 Windows TAR	7c876cb5ab7fb109f5efc33f488649fc
Source TAR	52456cf4d7bb81e267f91f30649f60a5

After download, enter the folder in terminal and use the following command:

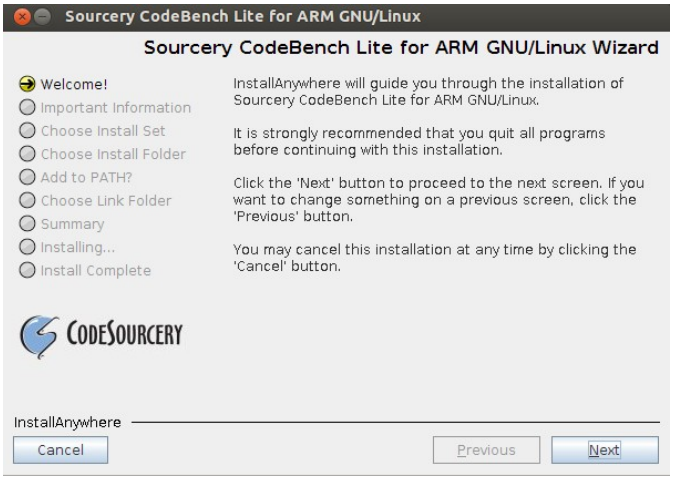
$ ln –s /lib/i386-linux-gnu/libc.so.6 /lib

$ dpkg-reconfigure dash

Choose *NO*.



$ sh arm-2012.09-64-arm-none-linux-gnueabi.bin



Add the CodeSourvery to PATH.

Input in the terminal:

$ gedit /etc/profile

Input in the *profile*.

$export PATH=.:$/root/CodeSourcery/Sourcery_CodeBench_Lite_for_ARM_GNU_LIinux/bin:$PATH

Save and close the *profile*.

Input in the terminal:

```
$ source /etc/profile
```

3. Test the first cross compiling environment

Program a simple first.c file.

```
#include<stdio.h>
int main()
{
    printf("Hello world\n");
    return 0;
}
```

Input in the terminal:

```
$ arm-none-linux-gnueabi-gcc –static –o first first.c
$ adb push first /data/local
$ adb shell
$ cd /data/local
$ ./first
```

```
root@android:/data/local # ls
first
tmp
word_count.ko
word_count.o
root@android:/data/local # ./first
Hello world
```

Suggestions for modification: