Bayonne + Linux + Dialogic HOWTO

Adrian Sai-wah TAM (swtam@school.net.hk)

Hong Kong School Net, The Chinese University of Hong Kong

Version 0.2: September 2, 2002

1 Document History

v0.2 (2002-09-02) With the help of SR 5.1 for Linux Installation HOWTO from Mr. Gerry Gilmore, enhanced the RedHat installation details. Some tricks on Debian's installation is also added.

v0.1 (2002-08-29) First version

2 Overview

Dialogic is a subsidiary of Intel Corporation. It produces a series of telephony devices for computers. The Dialogic devices are proud of its generic software interface – all Dialogic device share the same API and device drivers (a.k.a. System Release), as said by Intel. However, the Dialogic's API is not a very efficient tool for developing telephony applications, especially in terms of the Rapid Application Development tilde.

Bayonne, a software project of GNU, is aimed to provide a generic framework for telephony hardware. It provides a high level abstraction so that all different hardware uses the same script to do the same thing (of course, provided that the hardware supports such features required). The script that Bayonne uses is simple and easy to understand. With the use of Bayonne, we can develop a telephony application easily.

Bayonne do not support all the telephony devices on the market. But lucky, Dialogic series is one of the brand it supports. The following describes how you can do with Dialogic and Bayonne on the Linux platform, from the hardware set up to application development.

3 Hardware installation

The Dialogic devices usually have a board identification switch on the board. The switch is something like a dial, with hexadecimal numbers 0 to F printed on it. This switch is for identifying and distinguishing the board from others when you have more than one board installed in a system. Every Dialogic board should have a unique identification number, but if you don't care about which board and which board, you can just make every board the identification number 0, which then, automatic assignment is applied base on the PCI bus and slot number. If you are not using the automatic assignment, remember to record the identification number and the corresponding board models. These are needed during the driver installation.

4 Install System Release on Linux system

The Linux system can be any distribution. But I would prefer classifying the distributions into RedHat-alike and Debian-alike. Their difference is somewhat large. But in this document, you only need to know one thing: Do you use RPM for installation of new packages. If not, you just read the Debian-alike part.

4.1 Installation for RedHat-alike distributions

I have tried the RedHat version 7.3 as well as Mandrake 9.0 beta 1 and beta 3. Installation on RedHat is easy. Just as the Intel says, install Linux Stream package first, then do the Dialogic system release installation. However, I found that installation on Mandrake is a little bit difficult. I don't know it is the problem of Mandrake-modified kernel or the instability of beta distributions. However, there is still ways to solve it. Anyway, if you found that your RedHat-alike distribution has some problem, do the tasks according to the paragraphs followed. If you are RedHat, you can probably skip section 4.1.1 and go from 4.1.3.

4.1.1 Kernel upgrade

The Dialogic system release version 5.1 is tailored for kernel version 2.4 series. Therefore, it would be best to keep up with the most current 2.4 kernel. I have not tried the installation on kernel 2.2 and the development kernel 2.5, please tell me the result if you tried this.

You should first, go the `ftp://ftp.kernel.org/pub/linux/kernel/v2.4/` and get the kernel tar ball. The one I'd tried is version 2.4.19. After you get the tar ball (say, linux-2.4.19.tar.bz2), you extract it into somewhere like `/usr/src`, like follows:

```
cd /var/tmp
wget ftp://ftp.kernel.org/pub/linux/kernel/v2.4/linux-2.4.19.tar.bz2
cd /usr/src
tar jxf /var/tmp/linux-2.4.19.tar.bz2
```

After extracting the tar ball, you should make the new kernel and use it. It is quite easy to do even if it is your first time to compile kernel. You just `chdir` into the kernel source directory and than `make menuconfig`, choose the options you needed, than `make bzImage`; `make modules`; `make modules_install`; and finally, `make install`. I don't want to describe in too detail, but you can consult The Linux Kernel HOWTO (`http://www.tldp.org/HOWTO/Kernel-HOWTO.html`) at Linux Documentation Project (`http://www.tldp.org/`) for details on kernel compilation. The most important thing here is: enable the Linux telephony support during you `make menuconfig`, either in compiled-in or module is okay. After the compilation and installation, remember to reboot and use the new kernel before proceed.

4.1.2 Afraid of make bzImage?

If you don't want to make your kernel from the tar ball, or you are novice enough that do not know how to do it, or you have any reason don't want to do it, you can also use the RedHat package manager for the kernel upgrade purpose.

Dialogic System Release requires a clean and pure kernel. That means, it would be better if you are using RedHat because RedHat seldom modify the kernel. That was my experience that Mandrake 8.2 beta 1 has some problem in the kernel that would give tons of harmless but noisy messages during the SR's modules load. The System Release 5.1 requires kernel version of 2.4.9+, therefore, if you are using RedHat 7.2, your kernel must be upgraded. If you are using Red Hat 7.3, that's great! You basically needs no upgrade of kernel. However, the make sure you have the following packages installed:

- kernel-2.4.18-3, or kernel-smp-2.4.18-3 if you are using a system with parallel processors
- kernel-source-2.4.18-3
- kernel-headers-2.4.18-3
- some version of gcc, gmake, devel libraries like glibc-devel, runtime libraries like glibc and libstdc++, ...

These packages are required because you need to compile something with the kernel source code. You just need to "`rpm -i`" or "`rpm -U`" them. The Red Hat users of older version may also use these RPM packages and upgrade your existing ones. Go to http://www.redhat.com/ or http://www.rpmfind.net/ for details on where to get these RPM packages. Again, if you have upgraded the kernel, don't forget to re-run the lilo or grub and reboot before proceed! RPM do not do these for you!

4.1.3 Install Linux Streams

Linux Streams (LiS) is important to Dialogic System Release. It uses LiS to make the driver for your system. So the first thing you have to do is to go to: `ftp://ftp.gcom.com/pub/linux/src/LiS/` and get the latest Linux Stream package. The lastest one at the time I write this document is version 2.14.6. Same as downloading and installing the kernel, you just download the tar ball (say, `LiS-2.14.6.tar.gz`) and extract everything into somewhere like `/usr/src` (recommended).

After that, you go to `/usr/src/LiS-2.14.6` and run `make`. Then you will have an interactive configuration session which is same as running `./configure`. Following is the questions you found:

1. How do you want to configure STREAMS?

   It is recommended to run in Linux kernel. Although it is the choice for running in user level, but it is recommended and required by the Dialogic System Release to make it a kernel mode program.

2. Are you using the native Linux C compiler or are you crossing-compiling using a different compiler?

   Well...are you cross-compiling? I haven't tried and don't know whether cross compiling works. Hereinafter, I assumed you are using a native compiler, just as what I did.

3. Enter directory location of your kernel source

   Usually the configuration program is smart enough to find the source directory of the kernel that you are using. If it fails, tell it the right one and press the enter key.

4. Which version of the kernel are you building for?

   It is the version number string of your kernel. If you have given the right source directory, it can normally find the string. This should be exactly correct because if you have given a wrong version string, the kernel may refuse to load the LiS modules.

5. Do you intend to run LiS on this machine with the currently running kernel?

   Normally you should choose yes.

6. How do you want to build STREAMS for the Linux kernel?

   You can choose to make LiS a part of the kernel (linked with the kernel) or a separate module that loads on your demand. However, you should have no choice but make it a module. It is required by the Dialogic System Release.

7. After building STREAMS, do you want kernel loadable modules installed?

   I have chosen yes to let the make program move the modules into `/lib/modules/2.4.19/misc`, just because I don't want to run `make modules_install` once again in the kernel source.

8. Enter location of your kernel module directory

   Give the location where the modules installed, usually it gives a correct path as default.

9. When you make STREAMS, do you want to use backward compatible constants in the file stropts.h?

I don't know much about this issue. I just used the default, UnixWare/Solaris compatible instead of LiS backward compatible constants.

10. When you make STREAMS, do you want to use Solaris style cmn_err?
    As well, I don't know much about this issue. I just used the default, SVR4 style.

11. When you make STREAMS, do you want to compile for source level debugging?
    As I don't want tons of messages jamming my screen or the /var/log/message file, I disabled it.

12. Do you want to use shared libraries?
    I have tried to make it statically linked. But something mysterious happened. I, threfore, recommends use the dynamic linking.

That's all of the questions. After that, you should see the LiS compiles and installs. In summary, if you are lucky enough (usually, in fact), you just need to press enter for several times and use all the default values would be okay. After LiS compiles, remember to run once again the `make install` command to make the LiS helper scripts copied into `/usr`.

4.1.4 Obtain and Install System Release

First of all, please go to Intel's web (http://resource.intel.com/telecom/support/releases/unix/index.html) and download the "System Release 5.1 for Red Hat Linux 7.1/7.2 on Intel Architecture" (SR 5.1). You will find a file named `LINUX_SR5.1.tgz` of 140 MB.

Extract it into a directory, you will than find 15 RPM files and 6 other files. If you are using RPM for package management, just run `./install.sh` and you will see a menu of 6 items. The first two are for SpringWare boards and the later four are for DM3 boards. You just need to follow the instruction of menu and install the required package.

For example, I am using a D/480JCT-1T1 card and therefore I choose `1` to indicate that I am using a card of SpringWare series. It will than show you a list of packages to be installed and install the RPM packages by using the "`rpm -i`" command. After the installation is complete without error, you will see the menu again and in this case, you can choose `Q` to quit. Further details on installation can be obtained from Intel. There is a document called "Installation and Configuration Guide for Linux" at the above URL describing how to do installations of the System Release 5.1 on RedHat (and alike systems).

Some remarks here: Unless you know specifically that you have *Antares* hardware, make sure not to select option 2 in the menu of install.sh. That option will create some entries in your config file that will clutter your screen with all kinds of mean, nasty, ugly messages during initialization. (Thanks Gerry)

After the installation of the SR 5.1, configuration is required. The configuration process is aimed to identify the model of board you are using, the IRQ it should be used, create the kernel modules for use as the device driver for the Dialogic devices and so on. Under Red Hat-alike system, it is quite easy. You just go to the SR 5.1 directory where you just extracted the 140 MB tar ball, there is a shell script called `config.sh`. Run it and you will see that the first question asked is whether you have a DM3 type board. If you answer no (that's me!), you will be prompted for whether you have other type of Intel Dialogic Board. Answering these few question is very important as wrong answer will give (may be) miserable results.

After these, you should follow the instructions appeared. It will help you copy the required files into `/usr/src/LiS/` and build a kernel modules (with installation to `/lib/modules/`*version*`/misc/`). It will also help creating configuration files in /usr/dialogic/conf/ by asking what models you have and what parameters (IRQs and board IDs, for example) you want to use. Please follow sections 2.4 to 5.9 of the "Installation and Configuration Guide for Linux" for better descriptions.

4.1.5 Testing if it works

Rebooting is never a good practice for Linux systems! The real Linux user only reboots at two occations: first is doing something with the kernel, and the second is dealing with the hardware which means you must stop the power. After installing the Dialogic SR, you don't need to reboot. Just call the following commands:

```
. /etc/profile.d/orbacus.sh
. /usr/dialogic/bin/setenv.sh
/etc/rc.d/init.d/orbacus start
/etc/rc.d/init.d/dialogic start
```

and you will have the same effect as rebooting the system.

All the Dialogic files relies in `/usr/dialogic/`, for example, the executables are at `/usr/dialogic/bin/`. You can try go there and call `dlstart` – a script for starting the Dialogic devices. If everything goes fine, it will download something onto the Dialogic hardware and started some daemons with some diagnostic information shown. After that, you can call the program `detect` in the same directory which will than tell you about what's installed in your system. You should normally see the model and board ID of the telephony boards. If what I say is happened, congradulation! You've make it. If not, try to read the Installation Guide or call the vendor for help. Probably it is the problem of your kernel version or you have chosen a wrong model during the post-install configuration. If it still fails, try think your system as some scratch system and follow the solution for Debian-alike distributions.

4.2 Installation for Debian-alike (or non-RPM) distributions

Seems that Intel belives Linux is equivalent to Red Hat. It looks do not understand what Linux really is. However, frankly, I don't like RedHat as many Linux gurus. Debian is my favoriate Linux favor, but it do not have RPM given in parallel with Debian Package system. I have no patience to install the RPM for it as well. Therefore, after reading the "Installing Dialogic Boards in Slackware Linux" by Diego Betancor on May 2001 (`http://diego.betancor.com/dialogicinlinux/`), I decided to do it by hand.

4.2.1 Prepare the kernel and Linux Streams

Just as described in sections 4.1.1 and 4.1.3, you should first keep the kernel up to date and with telephony options enabled. Then give the Linux Streams installed as modules.

4.2.2 Installing the Dialogic System Releases

The first step is the same as the Red Hat-alike installation: please get the 140 MB package first. Then extract it into some directory. At there, just call the `install.sh` script. It should fail definitely because we have no RPM system. But never mind, I just need the output. The install script will tell you what packages you should install. For example, in my case, I am installing a SpringWare board and it told me that I should have `DLGCcom`, `DLGCcsp`, `DLGCdev`, `DLGCgc`, `DLGCooc`, `DLGCparms`, `DLGCpri` packages. Then, assuming you have the `rpm2cpio` and `cpio` utilities, convert all the required RPMs into cpio formats. In my case, I did this:

```
for pkgs in com csp dev gc ooc parms pri; do
    rpm2cpio DLGC${pkgs}-*.i386.rpm > DLGC${pkgs}.cpio
done
```

Then, you have two choice. If you are safety-oriented, make a directory, say, `/tmp/sr51/` and call the following command. If you are risk-taking, just go directly to the root directory and take:

```
for pkgs in com csp dev gc ooc parms pri; do
    cpio -id < /path/to/files/DLGC${pkgs}.cpio
done
```

Basically, if you call `cpio` in `/tmp/sr51`/, you will see that there is a `./usr/dialogic/` directory and a `./etc/` directory. Intel did not use the full power of RPM, but just make it as an alternative of tar ball. So we just need to copy everything into root (`cp -R * /`) is okay.

Debian is somewhat differs from Red Hat. Its start up script is at `/etc/init.d/` instead of `/etc/rc.d/init.d/`, so you have to move the dialogic startup script and other files to correct locations. I would suggest to make a symbolic link as there is still some scripts in `/usr/dialogic/bin` believes there is a `/etc/rc.d/init.d/orbacus` script. Commands are as follows:

```
ln -s /etc/rc.d/init.d/orbacus /etc/init.d/orbacus
ln -s /etc/rc.d/rc3.d/S85orbacus /etc/rc3.d/S85orbacus
ln -s /etc/rc.d/rc5.d/S85orbacus /etc/rc5.d/S85orbacus
```

After that, you have to make the dynamic link libraries available for the use of the SR 5.1 binaries and header files available for the compilation of drivers. What you have to do is call the following commands at anywhere:

```
for dyn in /usr/dialogic/lib/* /usr/dialogic/ooc/lib/*; do
    ln -s ${dyn} /usr/lib/`basename ${dyn}`
done
for header in /usr/dialogic/inc/*.h; do
    ln -s ${header} /usr/include/`basename ${header}`
done
```

Now, you have created 58+105=163 soft links. Then run this list of commands accordingly:

```
ln -s /bin/pidof /sbin/pidof
ln -s /usr/lib/libxerces_c1_5.so /usr/lib/libxerces.so
mkdir -p /var/dialogic/log
chmod 766 /var/dialogic /var/dialogic/log
ln -s /var/dialogic/log /usr/dialogic/log
cp /usr/dialogic/init.d/dialogic /etc/init.d/dialogic
ln -s /etc/init.d/dialogic /etc/rc3.d/S90dialogic
ln -s /etc/init.d/dialogic /etc/rc5.d/S90dialogic
ln -s /etc/init.d/dialogic /etc/rc0.d/K90dialogic
ln -s /etc/init.d/dialogic /etc/rc6.d/K90dialogic
cp /usr/dialogic/cfg/RtfConfig.xml /usr/dialogic/cfg/DefaultRtfConfig.xml
chmod -w /usr/dialogic/cfg/DefaultRtfConfig.xml
echo '. /etc/profile.d/orbacus.sh' >> /etc/profile
echo '. /usr/dialogic/bin/setenv.sh' >> /etc/profile
```

which creates a symbolic links to make Debian compatible with the Red Hat nomenclature and do some post-installation script that RPM will do. Follows that, create a file `/usr/dialogic/cfg/.master.cfg` with the following content (2 lines):

```
00200  vox  sram  1  2
00201  ant  an    1  0
```

Now you have already finished most of the jobs. Next, is to go back to source directory of the SR 5.1 which your `config.sh` locates, and modify the `config.sh` script at line 20 where defines variable `TEST_PKG` to “`rpm -q`”. Because we do not use rpm as the package management tool, the command `rpm -q packagename` must fail and gives error return value. To make the powerful `config.sh` script runs, we need the `$TEST_PKG packagename` return successfully. You can, therefore, modify the `$TEST_PKG` into anything that gives successful return values. What I did is make it into "`echo`", which will definitely return 0 whatever you give.

After your modification, just execute `config.sh` and do the configuration as in section 4.1.4 et voila!

4.2.3 Problems?

One thing very important: If your `/etc/init.d/orbacus` script fails and says relocation error and do not understand the symbol `__dynamic_cast_2`, it is not Debian's fault! The orbacus stuff are used to run on Red Hat and it seems that the Red Hat's dynamic library is required. Two possible solutions to this is (any one will work):

1. Use strip to kill the `__dynamic_cast_2` symbol in `/usr/dialogic/ooc/lib/libOB.so`, command line is:
   `strip --strip-symbol=__dynamic_cast_2 /usr/dialogic/oob/lib/libOB.so.4.0.5`
2. Backup your original library file and copy the file `/usr/lib/libstdc++-libc6.2-2.so.3` from Red Hat (or Mandrake) distribution and overwrite the original

The less-dirty solution is under investigation. Please give me some time (and hints if you have).

Alright, back to the basic. After the installation, you should first test the system according to section 4.1.5. If it works, that's great and skip to the next section for the works on Bayonne. If not, probably you need to change something in /usr/dialogic/cfg/. The most critical configuration file is dialogic.cfg (for SpringWare only). In most case, you need to have a "Dialog/HD=YES" in the section [Genload - All Boards], which the config.sh and mkcfg scripts always missed. For others parameters usable in the dialogic.cfg file, you should consult section 3.3 of the Installation and Configuration Guide. The following is my configuration file for the system with one VFX/PCI card for your reference:

```
[Genload - All Boards]
Dialog/HD=YES
BusType=SCBus
SCBusClockMaster=AUTOMATIC
SCBusClockMasterSource=AUTOMATIC
#[Genload - ID 0] /* T1/E1 voice HD */
#Feature=dpd_in
#ParameterFile=spandti.prm
#[Genload - ID 1] /* T1/E1 voice HD */
#ISDNProtocol=ne1
```

Sometimes, (in fact, most of the time) you may not have the file /etc/rc.d/init.d/functions which the Red Hat-alike systems have. This function is useless but give some fancy output on loading the start up scripts in /etc/rc.d/init.d/*. If it is the case, your orbacus start up script will not run properly. The solution is simple:

Delete the lines (line 12, 47, 69) which imports the `functions` script and call to `echo_success` functions.

5 Bayonne's matter

5.1 Making Bayonne from Scratch

The main portal of Bayonne is at the GNU: `http://www.gnu.org/software/bayonne/`. As described in the Bayonne User Manual, you needs four more components to make Bayonne works, namely the CommonC++, ccAudio, ccRTP, ccScript. These four components can be obtained via HTTP and FTP from ftp.gnu.org or any other GNU FTP mirrors. At the time I write this document, the latest versions available are: CommonC++2 1.0.1 (`commoncpp2-1.0.1.tar.gz`), ccAudio 1.0 RC 1 (ccaudio-1.0rc1.tar.gz), ccRTP 0.9.1 (ccrtp-0.9.1.tar.gz), ccScript 2.2.1 (`ccscript-2.2.1.tar.gz`). The latest version of Bayonne is 1.0.0 (bayonne-1.0.0.tar.gz).

After obtaining all the tar balls, extract and install the CommonC++ first, then the ccAudio, ccRTP and ccScript in any other, and finally extract and install the Bayonne package. The following screen dumps shows what's being done:

```
# ls -l
bayonne-1.0.0.tar.gz     ccrtp-0.9.1.tar.gz       commoncpp2-1.0.1.tar.gz
ccaudio-1.0rc1.tar.gz    ccscript-2.2.1.tar.gz
# for x in *; do tar zxf $x; done
# ls -l
```

```
bayonne-1.0.0/          ccrtp-0.9.1/          commoncpp2-1.0.1/
bayonne-1.0.0.tar.gz    ccrtp-0.9.1.tar.gz    commoncpp2-1.0.1.tar.gz
ccaudio-1.0rc1/         ccscript-2.2.1/
ccaudio-1.0rc1.tar.gz   ccscript-2.2.1.tar.gz
# cd commoncpp2-1.0.1
# ./configure --prefix=/usr
....
# make && make install
....
# cd ../ccaudio-1.0rc1
# ./configure --prefix=/usr
....
# make && make install
....
# cd ../ccrtp-0.9.1
# ./configure --prefix=/usr
...
# make && make install
...
# cd ../ccscript-2.2.1
# ./configure --prefix=/usr
...
# make && make install
...
# cd ../bayonne-1.0.0
```

```
# ./configure --prefix=/usr
...
# make && make install
...
```

In some case, just as what I encountered, you may found some library is missing and some packages do not compile. Basically it should not happen but once it happens, give a symbolic link in `/usr/lib` to fix the problem. Some `Makefile` just mix up the filename of libraries for the linker. (I've failed to make bayonne 1.0rc1 because the `libperl.so` do not exist but I have `libperl.so.5.6.1`, make a symbolic link will do.)

After the Bayonne and its depending packages are installed, you can have a try to see if it works. Just start the Dialogic daemon (/usr/dialogic/init.d/dialogic), and the Bayonne daemon (/etc/rc.d/init.d/bayonne). Then you will find that a demo program starts. Dial to the telephony card and have a try.

5.2 Global Call

Bayonne is not the only way to drive Dialogic devices. Intel provides another suite of development software called Global Call for such a purpose. Global Call is a suite of API, however, it is different from the older, low-level C-API. The Global Call can make us share the same code base between all different Dialogic devices, no matter it is a DM3 or SpringWire board, it is digital or analog. Nevertheless, it is not the purpose of discussing Global Call in this document. If you are interested, please refer to http://resource.intel.com/telecom/support/releases/protocols/GCProtocols20/index.html for details.

6 Creating your own Bayonne Application

Pending