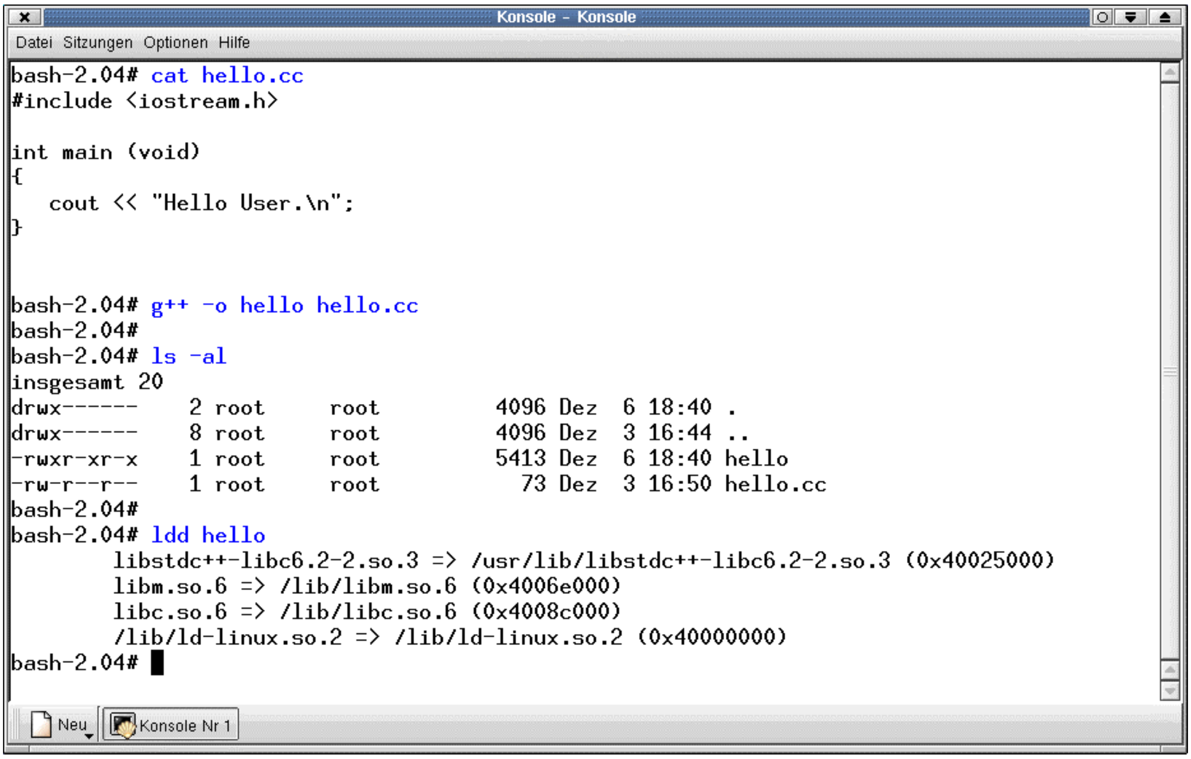
How to Program the Linux-based ADNP/1486 with C++

C++ programs need additional libraries. Please make sure that your ADNP/1486 is using a Linux configuration with JFFS space. The document describes how to find out which libraries are necessary and how to install these libraries within the ADNP/1486 JFFS file space.

- **1. Step**: Edit your C++ source code. Save your C++ source codes in files with the extension **.cc** (i.e. **hello.cc**).



```
bash-2.04# cat hello.cc
#include <iostream.h>

int main (void)
{
   cout << "Hello User.\n";
}


bash-2.04# g++ -o hello hello.cc
bash-2.04#
bash-2.04# ls -al
insgesamt 20
drwx------    2 root     root         4096 Dez  6 18:40 .
drwx------    8 root     root         4096 Dez  3 16:44 ..
-rwxr-xr-x    1 root     root         5413 Dez  6 18:40 hello
-rw-r--r--    1 root     root           73 Dez  3 16:50 hello.cc
bash-2.04#
bash-2.04# ldd hello
        libstdc++-libc6.2-2.so.3 => /usr/lib/libstdc++-libc6.2-2.so.3 (0x40025000)
        libm.so.6 => /lib/libm.so.6 (0x4006e000)
        libc.so.6 => /lib/libc.so.6 (0x4008c000)
        /lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
bash-2.04# █
```

- **2. Step:** Run the Linux/GNU C++ compiler and build a executable from your C++ source code file. The name of C++ compiler is **g++**. The following command line assumes that **hello.cc** is your C++ source code file and **hello** the name of the executable.

  ```
  g++ -o hello hello.cc
  ```

- **3. Step:** Check with the **ldd** utility program the names of the dynamic link libraries, which are necessary to run your executable on the ADNP/1486.

  ```
  ldd hello
  ```

  We assume that your executable needs **libc.so.6**, **libm.so.6** and **libstdc++-libc6.2-2.so.3**. The library **libc.so.6** is already present within the ADNP/1486 root file system (see directory **/lib**).

- **4. Step:** Check the symbolic links within the **/lib** directory of your development system and find the file name of the real library file. We assume that **libstdc++-libc6.2-2.so.3** is a symbolic link to **libstdc++-3-libc6.2-2-2.10.0.so.**

- **5. Step:** Transfer the missing libraries **libm.so.6** and **libstdc++-3-libc6.2-2-2.10.0.so** with FTP direct to the JFFS space (see directory **/mnt**). Make sure that your FTP client owns the necessary write access rights.

- **6. Step:** Check the new libraries in **/mnt**. It is necessary to have the name details.

- **7. Step:** Build two symbolic links to the new libraries in **/mnt**. Setup a Telnet session to the ADNP/1486. Make sure to get superuser rights for this session. Execute the following commands within your Telnet session.

  ```
  cd /lib
  ln -s /mnt/libstdc++-3-libc6.2-2-2.10.0.so libstdc++-libc6.2-2.so.3
  ln -s /mnt/libm.so.6 libm.so.6
  ```

  The **ln** command builds in this case a link within the **/lib** directory of the ADNP/1486 to the real library file in the **/mnt** directory (JFFS space of the ADNP/1486).

- **8. Step:** Transfer the executable from your development system to the ADNP/1486 and run this executable. If your executable needs other or more libraries you get an error message. Repeat the steps 3 to 7 for any additional library. Finally rebuild your RIMAGE.GZ to have the symbolic links permanent within **/lib**.