- Instead of Copying: Cross-Compiling -

Goal:
Use this exercise to learn how to include target libraries automatically on the host and how to avoid tedious and errorprone copying of multiple files and directories.

Exercise 8.1: Preparation (1)
Check that the following packets

- **`libsndfile1, libsndfile1-dev, sndfile-programs, hexer`**
- **`nfs-kernel-server`**

are correctly installed on your target.
Now export your root directory on the target to your host (and only to your host) using:

- **`exportfs`**

On your host, import this directory by mounting it on **`/mnt/rootfs`** .
Hint 1 regarding NFS on Raspian-Jessie: The system startup under Jessie has moved to **`systemd`**. This results in a nasty circular dependency, which prevents the proper startup of **`rpcbind/portmapper`**. So in case of NFS issues you might want to check the execution status of **`rpcbind`**. If necessary, re-start **`rpcbind`** and then re-start the nfs-Server, both manually. To automate this, you want to put this into **`/etc/rc.local`**, so it works after rebooting.
Hint 2 regarding NFS-Client running on Ubuntu: When using **`mount.nfs`** pls. make sure to add the Option **`–overs=3`** . Without it, NFSv4 is used by default, which is based on the Kerberos authentication system. This might result in extremely long timeout intervals (>4min) when connecting.

Exercies 8.2: Preparation (2)
Now look on the target for the header-File **`sndfile.hh`**. Which absolute file path do we need to configure on the host? Add it as **`#include`** path into Eclipse. Now set the **`sysroot`** option as a linker flag in Eclipse, appropriately to your mount path.

Exercies 8.3: Again: My first sine wave – this time with cross-compilation
After this setup procedure, we are able to cross compile the project of exercise 7.2. Create a run- and debug configuration for your target and execute the program both on host and on the target. Compare the content of the WAV files created on the host and on the target, e.g using **`diff`**.