**1.Übung**

**1.Aufgabe:**

3.

**TFTP = Trivial File Transfer Protocol:**

TFTP ist ein sehr einfaches Dateiübertragungsprotokoll. Es unterstützt lediglich das

Lesen und Schreiben von Dateien. Nicht vorhanden sind viele Funktionen des mächtigeren

FTP wie etwa Rechtevergabe mittels chmod.

**DHCP = Dynamic Host Configuration Protocol:**

DHCP ist ein Kommunikationsprotokoll, um die Zuweisung der Netzwerkkonfiguration an

Clients durch einen Server zu ermöglichen.

**NFS = Network File System:**

Das NFS ist ein von Sun Microsystems entwickeltes Protokoll, das den Zugriff auf Dateien über ein Netzwerk ermöglicht.

Dabei werden die Dateien nicht wie z. B. bei FTP übertragen, sondern die Benutzer können auf Dateien, die sich auf einem

entfernten Rechner befinden, so zugreifen, als ob sie auf ihrer lokalen Festplatte abgespeichert wären.

**2.Aufgabe:**

1./2.

Versionen:

root@debian:~/pxadev# gcc --version

gcc (Debian 4.4.5-8) 4.4.5

Copyright (C) 2010 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO

warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

root@debian:~/pxadev# arm-linux-gcc --version

arm-linux-gcc (GCC) 3.3.2

Copyright (C) 2003 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO

warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

root@debian:~/pxadev# arm-linux-ld --version

GNU ld version 2.14.90.0.7 20031029

Copyright 2002 Free Software Foundation, Inc.

This program is free software; you may redistribute it under the terms of

the GNU General Public License. This program has absolutely no warranty.

root@debian:~/pxadev# arm-linux-as --version

GNU assembler 2.14.90.0.7 20031029

Copyright 2002 Free Software Foundation, Inc.

This program is free software; you may redistribute it under the terms of

the GNU General Public License. This program has absolutely no warranty.

This assembler was configured for a target of `arm-linux'.

root@debian:~/pxadev# arm-linux-gdb --version

GNU gdb 6.7.1

Copyright (C) 2007 Free Software Foundation, Inc.

License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>

This is free software: you are free to change and redistribute it.

There is NO WARRANTY, to the extent permitted by law. Type "show copying"

and "show warranty" for details.

This GDB was configured as "--host=i686-pc-linux-gnu --target=arm-linux".

root@debian:~/pxadev#

3.

root@debian:/media/USB Basti/EOS# file HelloWorldGCC

HelloWorldGCC: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.18, not stripped

root@debian:/media/USB Basti/EOS# file HelloWorldARM

HelloWorldARM: ELF 32-bit LSB executable, ARM, version 1, dynamically linked (uses shared libs), for GNU/Linux 2.0.0, not stripped

in computing, the Executable and Linkable Format (ELF, formerly called Extensible Linking Format) is a common standard file format for executables, object code, shared libraries, and core dumps. First published in the System V Application Binary Interface specification,[1] and later in the Tool Interface Standard,[2] it was quickly accepted among different vendors of Unix systems. In 1999 it was chosen as the standard binary file format for Unix and Unix-like systems on x86 by the 86open project.

The first is compilation which turns source code into object modules.

The second, linking, is what combines object modules together to form an executable.

The distinction is made for, among other things, allowing third party libraries to be included in your executable without you seeing their source code (such as libraries for database access, network communications and graphical user interfaces), or for compiling code in different languages (C and assembly code for example) and then linking them all together.

When you statically link a file into an executable, the contents of that file are included at link time. In other words, the contents of the file are physically inserted into the executable that you will run.

When you link dynamically, a pointer to the file being linked in (the file name of the file, for example) is included in the executable and the contents of said file are not included at link time. It's only when you later run the executable that these dynamically linked files are bought in and they're only bought into the in-memory copy of the executable, not the one on disk.

if you compile an executable with gcc's -g flag, it contains debugging information. That means for each instruction there is information which line of the source code generated it, the name of the variables in the source code is retained and can be associated to the matching memory at runtime etc. Strip can remove this debugging information and other data included in the executable which is not necessary for execution in order to reduce the size of the executable.

4.

root@debian:/media/USB Basti/EOS/Uebung1/HelloWorld# file HelloWorld.o

HelloWorld.o: ELF 32-bit LSB relocatable, ARM, version 1, not stripped

nur kompilieren, kein linken

5.

/etc/exports

/root/pxadev/rootfs 192.168.1.0/255.255.255.0(rw,no\_root\_squash,no\_subtree\_check)

3.Aufgabe

PC:

root@debian:/tmp/RandomNumbers# time ./RandomNumbers

real 0m0.129s

user 0m0.128s

sys 0m0.000s

ARM:

root@colibri:/var/tmp# time ./RandomNumbersARM

real 0m 6.61s

user 0m 4.49s

sys 0m 0.10s

tmux