Hochschule für Technik und Wirtschaft
Studiengang Kommunikationsinformatik
Prof. Dr.–Ing. Damian Weber

Rechnernetze

5. Übung

Aufgabe (Tic–Tac–Toe mit UDP)

Ziel der Aufgabe ist es, das Tic–Tac–Toe–Programm aus der vorigen Übung um Netzwerkkommunikation mittels UDP zu ergänzen. Es sollen zwei Spieler über das Netzwerk Tic–Tac–Toe spielen können.

Sie müssen hierzu einen Socket erzeugen. Ein Socket ist ein `int`–Wert, der für das Senden und Empfangen von Daten über das Netzwerk benutzt wird. Der Aufruf

```
socket(AF_INET,SOCK_DGRAM,0)
```

erzeugt einen solchen Socket, speichern Sie diesen in einer `int`–Variable. Siehe hierzu auch die Manualpage der `socket()` Funktion. Die dort enthaltenen `#include`–Hinweise stellen auch die im folgenden benötigten Datenstrukturen bereit.

Der erzeugte Socket muß, um Senden *und* empfangen zu können, an eine Netzwerkadresse gebunden werden, diese besteht aus Adreßfamilie, IP–Adresse und einem frei wählbaren Port > 1024. Hierfür gibt es eine Struktur `struct sockaddr_in`, die folgendermaßen deklariert ist:

```
(siehe /usr/include/netinet/in.h)

struct sockaddr_in {
sa_family_t    sin_family; /* address family: AF_INET */
short int      sin_port;   /* port in network byte order */
```

```
struct in_addr sin_addr;  /* internet address */
...
};
```

Hierbei ist `in_addr` als

```
struct in_addr {
unsigned int s_addr;     /* address in network byte order */
};
```

deklariert.

Füllen Sie die Struktur(en), konsultieren Sie die Manualpage von `bind()` und rufen Sie die `bind()`–Funktion auf.

Jetzt können Sie über diesen Socket mit Hilfe der

- `sendto()` Funktion UDP–Pakete an einen Empfänger schicken
- `recvfrom()` Funktion UDP–Pakete am angegebenen Port empfangen.

Sie benötigen bei `sendto()` IP–Adresse und Port des Empfängers. Bei `recvfrom()` erhalten Sie neben den empfangenen Daten auch IP–Adresse und Port des Senders.

Beachten Sie, daß IP–Adresse (32 Bit) und Port (16 Bit) beim Aufruf der genannten Funktionen in *network byte order* vorliegen müssen.

Unterteilen Sie Ihren Testvorgang in drei Schritte:

- Testen Sie Ihr Programm zunächst innerhalb zweier Fenster an Ihrem Arbeitsplatzrechner (IP `127.0.0.1`).
- Testen Sie danach Ihr Programm auf zwei verschiedenen Rechnern des ISL–Labors, indem Sie sich mit *ssh* auf einem weiteren Rechner einloggen.
- Versuchen Sie, daß auch die Kommunikation mit der Bigendian–Maschine `stl-s-stud` richtig funktioniert.