Mikrorechentechnik II

INT-4

**Kommunikation mit Raspberry Pi**

Versuchsprotokoll

Gruppe 18

4. Juli 2022

vorgelegt von

Name: Vorname

Hengstler Jakob

Hanusch Dustin

Pavlov Ivo

Brantz Tobias

Betreuer: Dr. Roland Schingnitz

Inhaltsverzeichnis

1. Aufgabenstellung des Praktikums

2. Erklärung des Programms

2.1. Flussdiagramm

2.2. Erläuterung der Dateien

3. Probleme während des Praktikums

4. Erläuterung eines interessanten Ausschnittes im Code

A. Quellcode

1. Aufgabenstellung des Praktikums

Die Aufgabe im Rahmen dieses Versuchs besteht darin, ein Client- und ein Serverprogramm in der Programmiersprache C zu erstellen, die über das Ethernet Netzwerk kommunizieren sollen. Dabei liest das Client-Programm periodisch Temperaturwerte von einem Sensor ein, wandelt diese in ein geeignetes Format und sendet sie an das Server-Programm. Dieses nimmt die Daten entgegen und gibt sie in geeigneter Weise auf einem OLED-Matrixdisplay aus.

Die Aufgabenstellung beinhaltet folgende Teilaufgaben:

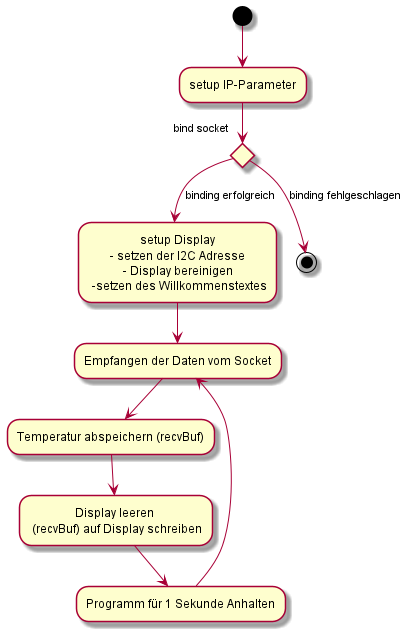
* Senden und Empfangen von Daten über das Ethernet-Netzwerk-Interface
* Abfragen des Temperatursensors über das SPI-Interface des Raspberry Pi
* Ansteuerung eines OLED-Displays über das I2C-Interface des Raspberry Pi

Zur Lösung dieser Teilaufgaben ist es erforderlich, die entsprechenden Hardwaredokumentationen zu SPI-, I2C- und Netzwerkinterface des Raspberry Pi sowie die zugehörigen Dokumentationen zu den benötigten Softwarebibliotheken zu verstehen.

2. Erklärung des Programms

2.1. Flussdiagramm

Server.c:



Client.c

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

2.2. Erläuterung der Dateien

Die zwei unterschiedlichen Dateien sind jeweils die Programme für den Server und den Client. Der Server stellt das empfangene Datum zudem auf einem Display dar, der Client liest die Temperatur aus einem angeschlossenem Sensor aus.

3. Probleme während des Praktikums

Zunächst muss erwähnt werden, dass wir leider nicht alle relevanten Dateien zur Verfügung hatten. Wir haben uns, so gut es ging, an den in der Anleitung gegebenen Code-Beispielen orientiert und für das SSD1306 Display uns eine öffentliche Bibliothek von GitHub heruntergeladen.

Da es bei dem Versuch mehr um das Verständnis des Ethernet-Protokolls ging und weniger um die Programmierung in C, gab es nur kleinere Probleme während der Programmierung. Diese bezogen sich vor allem auf die Einrichtung der Toolchain in den verschiedenen IDEs, welche wir benutzten. Bei der Programmierung des Servers gab es eine kleine Schwierigkeit beim Socket-Binding (Zeile 71, „server.c“), was nötig ist, um Daten zu empfangen.

Während der Durchführung des Praktikums gab es noch einen Speicherfehler, welcher sich nach Einsatz des Debuggers auf eine Methode zurückführen ließ, welche im C99-Standart nicht verfügbar war, bzw. nicht korrekt ausgeführt werden konnte.

Zusammenfassend ist festzuhalten, dass die Probleme im Wesentlichen dort herrührten, dass wir uns vorher nur wenig bis gar nicht mit der verwendeten Hardware/Software (v.a. Ethernet-Protokoll) befasst hatten und uns dementsprechend erst einarbeiten mussten.

4. Erläuterung eines interessanten Ausschnittes im Code

Im Folgenden wird die Funktion zum Senden von Paketen über UDP näher erläutert:

void send\_data(float temp){  
 char msg[30] ;  
 sprintf(msg, “%.1f”, temp);  
 unsigned int addrlen = sizeof (server\_addr);  
  
 long count = sendto(sockid,msg,strlen(msg),0,(struct sockaddr\*)&server\_addr,addrlen);  
 printf("count: %li\n",count);  
 status = count;  
}

Die Funktion nimmt einen Parameter vom Typ „float“ entgegen und hat keinen Rückgabewert. Es wird jener Datentyp verwendet, da die Funktion, welche die Temperatur vom Sensor ermittelt keine ganze Zahl zurückgibt.

Zunächst wird ein char-Array als Buffer definiert. Danach wird der übergebene float in Character konvertiert und in den Buffer geschrieben. Von diesem wird nun noch die Länge ermittelt. Mit der „sendto“-Funktion aus der socket-Bibliothek wird nun das Array an den Server übertragen. Die Funktion benötigt die ID des vorher bereitgestellten Sockets, das Array, die Länge des Arrays die Adresse des Servers und die Länge der Adresse.

Der Rückgabewert entspricht der tatsächlich übertragenen Zeichenanzahl. Diese wird dann auf der Konsole ausgegeben, um zu überprüfen, ob die Übertragung korrekt stattgefunden hat. Der Wert wird zudem noch in die Variable „status“ geschrieben, welche zu einem späteren Zeitpunkt benötigt wird.

A. Quellcode

1) client.c

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

2) server.c

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung Ein Bild, das Text enthält.

Automatisch generierte Beschreibung