Programmeren in C op de Raspberry Pi

Communicatie via WiFi (een hoofdstuk als voorbeeld)

Dogan Ibrahim (Verenigd Koninkrijk)

Gewoonlijk wordt de Raspberry Pi geprogrammeerd met behulp van Python. Python is een zeer krachtige taal, maar C wordt toch meer gebruikt onder programmeurs. Programmeren in C wordt ook onderwezen op bijna alle technische scholen en universiteiten. Dit artikel laat zien hoe we kunnen programmeren in C op de RPi, zodat u kunt zien wat daar bij komt kijken voor het ontwikkelen van uw eigen hardware-gebaseerde projecten. U kunt aan de slag in uw thuislab, gewapend met een boek en een Raspberry Pi. We gaan het gewoon proberen!

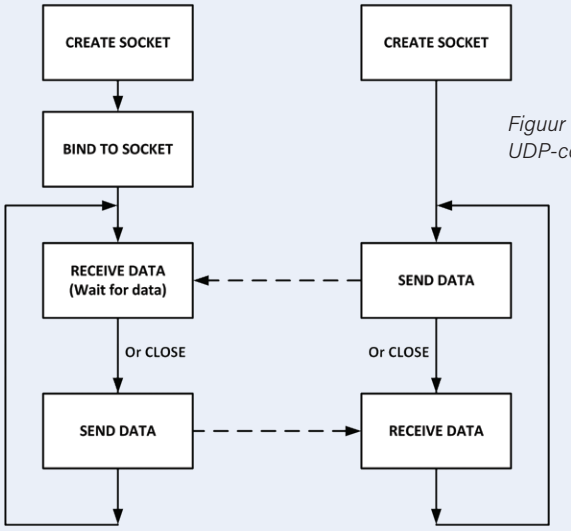
Opmerking van de redactie: Dit artikel is een gedeelte uit het 376 pagina's tellende boek C Programming on Raspberry Pi — Develop innovative hardware-based projects in C. *Het is enigszins bewerkt en aangepast om beter in de stijl van Elektor Magazine te passen. Omdat dat een gedeelte is uit een grotere publicatie, kunnen er in dit artikel verwijzingen voorkomen naar andere hoofdstukken in het boek. De auteur en de redactie hebben hun best gedaan om dit te vermijden, en zijn altijd bereid om op vragen in te gaan. U vindt hun contactgegevens in het tekstkader* **Vragen of opmerkingen?**

UDP en TCP/IP

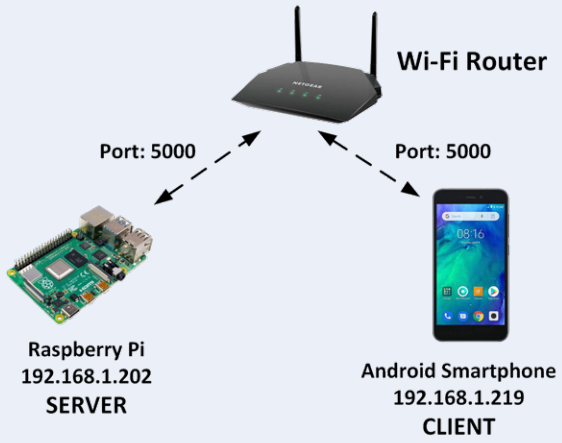
Communicatie over een WiFi-verbinding gaat volgens een client/server-model. Datapakketten worden verstuurd en ontvangen via sockets. Over het algemeen wacht de server op een verbinding van een client en als die verbinding is gemaakt, communiceren ze over en weer. Daarbij wordt vooral gebruik gemaakt van twee protocollen: UDP en TCP. TCP is een protocol waarbij een verbinding wordt opgebouwd, waarlangs de pakketten gegarandeerd worden afgeleverd. De pakketten krijgen volgnummers en de ontvangst van elk pakket wordt bevestigd, zodat ze ook in de juiste volgorde aankomen. Door deze manier van werken is TCP meestal langzaam maar betrouwbaar: de pakketten komen gegarandeerd aan. Bij UDP wordt geen verbinding opgebouwd. De pakketten hebben geen volgnummers en het is niet gegarandeerd dat ze hun bestemming bereiken. UDP heeft minder overhead dan TCP en is daardoor sneller. In **tabel 1** hebben we enkele van de verschillen tussen de TCP en UDP op een rijtje gezet.

Tabel 1: Communicatie via TCP en UDP

TCP	UDP
Pakketten hebben volgnummers en de ontvangst van elk pakket wordt bevestigd	Er is geen ontvangstbevestiging
Langzaam	Snel
Geen verlies van pakketten	Pakketten kunnen zoekraken
Veel overhead	Weinig overhead
Vraagt meer resources	Vraagt minder resources
Er wordt een verbinding opgebouwd	Er wordt geen verbinding opgebouwd
Lastiger te programmeren	Eenvoudiger te programmeren
Voorbeelden: HTTP, HTTPS, FTP	Voorbeelden: DNS, DHCP, Computer games



Figuur 1: UDP-communicatie.



Figuur 2: Blokschema van het project.

UDP-communicatie

In **figuur 1** ziet u UDP-communicatie over een WiFi-link.

Server

1. Maak een UDP-socket
2. Koppel het socket met het server-adres.
3. Wacht totdat er een datagram-pakket van een client binnenkomt.
4. Verwerk het datagram-pakket.
5. Stuur een antwoord naar de client of sluit het socket.
6. Ga terug naar stap 3 (als het socket niet gesloten is).

Client

1. Maak een UDP-socket
2. Stuur een bericht naar de server.
3. Wacht op een antwoord van de server.
4. Verwerk het antwoord.
5. Ga terug naar stap 2, of sluit het socket.

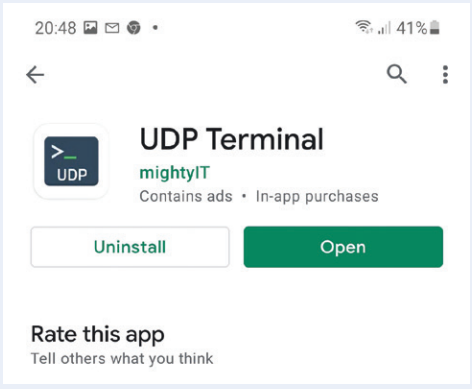
Project 1: Communiceren met een Android-smartphone met UDP (RPi is de server)

In dit project gebruiken we UDP-communicatie om data te ontvangen van en te zenden naar een Android smartphone. In dit project is Raspberry Pi de server en de Android-smartphone de client.

Doel: Dit project laat zien hoe we UDP-communicatie kunnen opbouwen tussen een Raspberry Pi en een Android-smartphone.

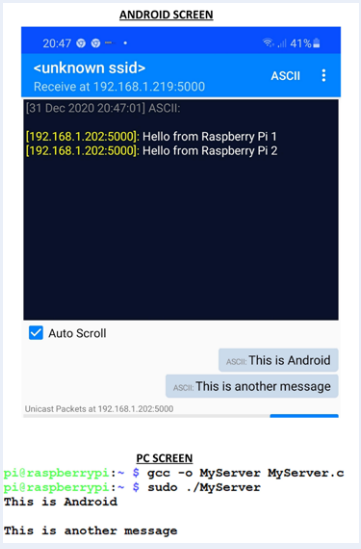
Blokschema: In **figuur 2** ziet u de structuur van het project. De Raspberry Pi en Android-smartphone communiceren via een WiFi-router.

Programmalisting: Het programma *MyServer.c* is weergegeven in **listing 1** en is te vinden in het softwarepakket bij het boek. Ga naar [1], scroll omlaag naar *Downloads* en klik op *Software_C Programming on Raspberry Pi*. Sla het .zip-bestand lokaal op en pak het uit, of pak alleen *MyServer.c* uit.



Figuur 3: UDP Terminal-apps op een Android-smartphone.

Figuur 4: Voer de poortnummers in.



Figuur 5: Voorbeeld van het programma in actie.

Aan het begin van *MyServer.c* worden de benodigde headerfiles ge-include in het programma. Het bericht *Hello from Raspberry Pi* wordt opgeslagen in het character-array `msg`. Er wordt een UDP-socket aangemaakt door de functie socket() aan te roepen. De handle wordt opgeslagen in de variabele `sock`. Dan worden de details over de server (Raspberry Pi) gegeven. Het adres is `INADDR_ANY` dus elke computer op hetzelfde netwerk met poort 5000 zal verbinden met de Raspberry Pi. De details over de server worden met de opgegeven poort gekoppeld met de functie `bind`.

De rest van het programma draait in een lus. Binnen die lus wordt de functie `recvfrom` aangeroepen om te wachten op de aankomst van een datapakket van de client-computer (Android-smartphone). Dit is een blokkerende call: de functie blijft wachten totdat er data is ontvangen van de client. Het programma sluit af als het karakter `x` wordt ontvangen van de client. Er wordt een `NULL`-karakter toegevoegd aan de ontvangen data en de data wordt met behulp van de functie `printf` weergegeven op het scherm van de Raspberry Pi. Een integer-variabele `count` wordt omgezet in een karakter en aan het eind van het character-array `msg` geplakt. Dit wordt naar de client gestuurd. De eerste keer geeft de client dus *Hello from Raspberry Pi 1* weer. De tweede keer krijgt hij *Hello from Raspberry Pi 2* enzovoort. Het socket wordt gesloten vlak voor het einde van het programma. We kunnen het programma als volgt compileren en uitvoeren:

```
gcc –o MyServer MyServer.c
sudo ./MyServer
```

In de Play Store zijn veel gratis UDP-programma's te vinden. In dit project gebruiken we *UDP Terminal* van *mightyIT* (zie **figuur 3**). Geef het poortnummer en IP-adres van uw Android-smartphone in en klik op *Start Terminal* zoals in **figuur 4**. In **figuur 5** ziet u het programma in actie. Hier zijn zowel het scherm van de client als dat van de server weergegeven.
Opmerking: u kunt het IP-adres van uw Raspberry Pi vinden met het commando `ifconfig`.

Listing 1: MyServer.c

```
/*------------------------------------------------------------------
     RASPBERRY PI - ANDROID SMARTPHONE COMMUNICATION
     ==============================================
This is UDP program. The program receives and then sends messages
to a smartphone over the UDP socket. Program terminates when
character x is sent from the smartphone.
This is the UDP server program, communicating over port 5000.

Author: Dogan Ibrahim
File  : MyServer.c
Date  : December 2020
------------------------------------------------------------------*/
#include <netinet/in.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#define Port 5000
#define BUFFSIZE 1024
//
// Start of MAIN program
//
int main(void)
{
    int sock, len, num, count = 1;
    char buffer[BUFFSIZE];
    struct sockaddr_in serveraddr, clientaddr;
    char msg[26] = “Hello from Raspberry Pi   “;
    sock = socket(AF_INET, SOCK_DGRAM, 0);
    len = sizeof(clientaddr);
    serveraddr.sin_family = AF_INET;
    serveraddr.sin_addr.s_addr = INADDR_ANY;
    serveraddr.sin_port = htons(Port);
    bind(sock,(struct sockaddr *)&serveraddr, sizeof(serveraddr));
    while(1)
    {
        num = recvfrom(sock, buffer, BUFFSIZE,MSG_WAITALL,
              (struct sockaddr *)&clientaddr, &len);
        if(buffer[0] == ’x’)break;
        buffer[num] = ’\0’;
        printf(“%s\n”,buffer);
        sprintf(&msg[24], “%d”,count);
        count++;
        sendto(sock, &msg, strlen(msg), MSG_CONFIRM,
              (struct sockaddr *)&clientaddr, len);
    }
    close(sock);
}
```

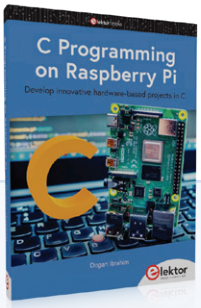
210346-03

Vragen of opmerkingen?

Als u vragen of opmerkingen hebt over dit artikel, email dan de auteur via d.ibrahim@btinternet.com of Elektor via editor@elektor.com

Met bijdragen van:

Tekst: **Dogan Ibrahim**
Redactie: **Jan Buiting**
Lay-out: **Giel Dols**
Vertaling: **Evelien Snel**



GERELATEERDE PRODUCTEN

- Book: D. Ibrahim, *C Programming on Raspberry Pi* (Elektor 2021) (SKU 19703)
  www.elektor.nl/19703
- E-Book: D. Ibrahim, *C Programming on Raspberry Pi* (Elektor 2021) (SKU 19704)
  www.elektor.nl/19704