

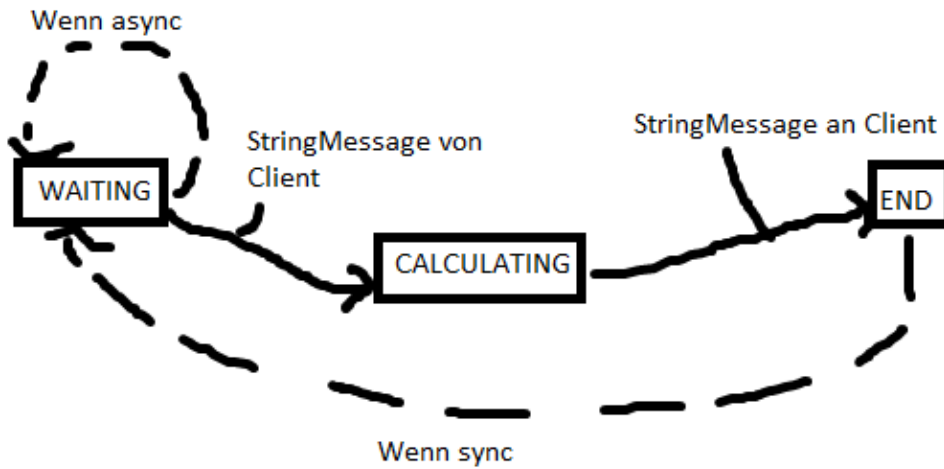
Protokoll:

Client schickt Message an Server

Server schickt Message zurück

Zustandsdiagramm:

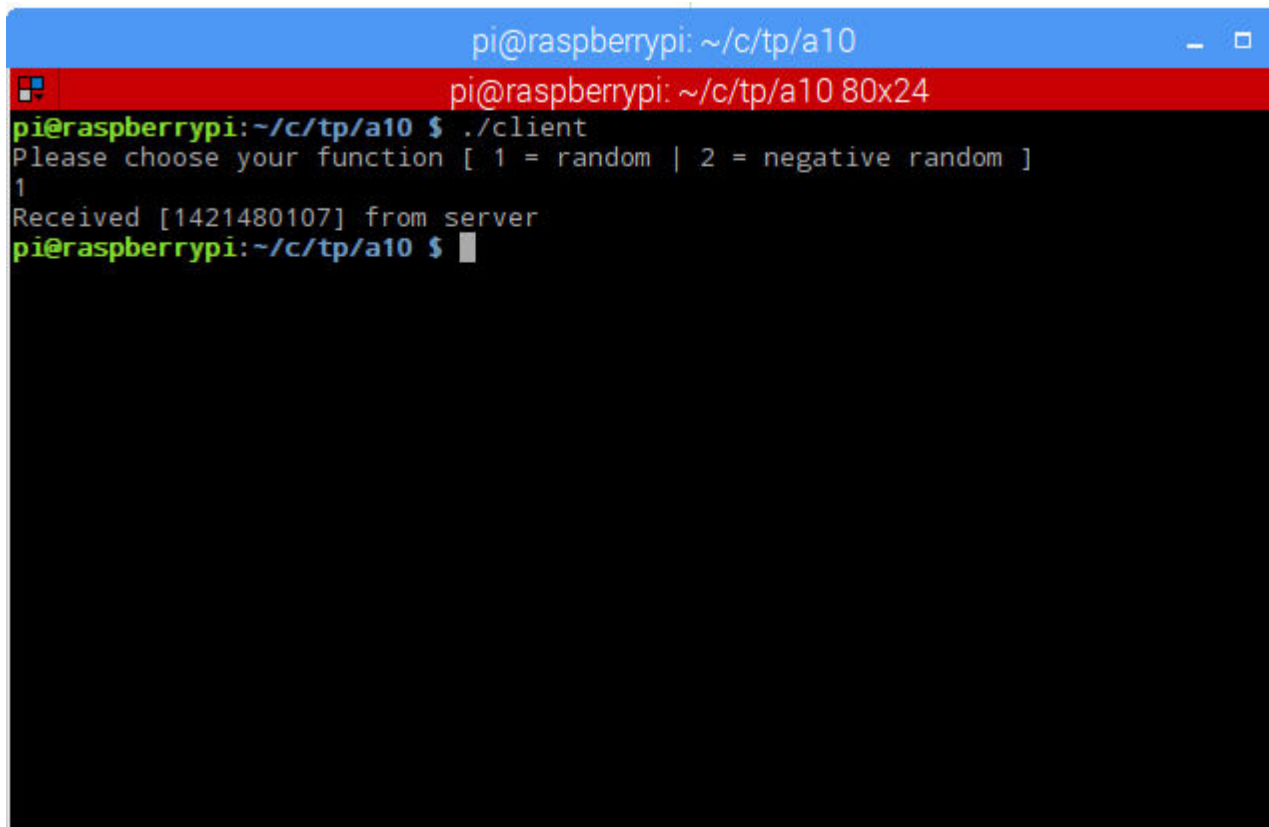
Server



Client



Test:

A terminal window titled 'pi@raspberrypi: ~/c/tp/a10' with a red title bar. The terminal shows the execution of './client'. It prompts the user to choose a function (1 = random, 2 = negative random). The user enters '1'. The terminal then displays 'Received [1421480107] from server' and returns to the prompt.

```
pi@raspberrypi: ~/c/tp/a10
pi@raspberrypi: ~/c/tp/a10 80x24
pi@raspberrypi:~/c/tp/a10 $ ./client
Please choose your function [ 1 = random | 2 = negative random ]
1
Received [1421480107] from server
pi@raspberrypi:~/c/tp/a10 $
```

Ausführung:

Einfach über CLI mit ./server bzw. ./seqServer bzw. ./client starten
Der Client muss dann entweder "1" oder "2" eingeben, um die Funktion auszuwählen.
Danach kann eine 10 Sekündige Wartezeit erwartet werden.

Kompilierung:

Ausführbare Dateien können durch
make [server | seqServer | client]
erstellt werden.

Antworten:

Offensichtlich ist die parallele Berechnungsmethode zwar nicht schneller, senkt aber Wartezeiten von anderen Clients, die nach dem ersten Client starten. Bei der sequentiellen Abarbeitung müssen diese erst auf die Beendigung der vorherigen warten, bei der parallelen Berechnung kommen alle sofort dran. Allerdings ist die maximale Anzahl an Threads begrenzt. Zu viele Threads behindern den Prozessor zu stark und könnten sogar zum Absturz des Programms führen.