**Part 1:**

Compare response time and CPU load:

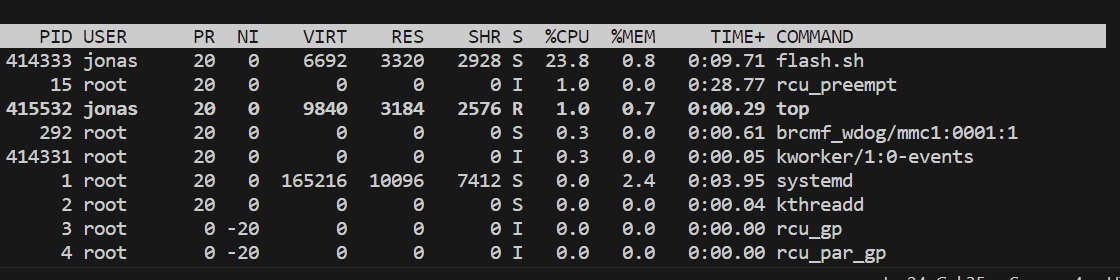
1. **Sysfs from shell script**

Response Time: Ø 4.28 ms

CPU Load: 23.8 %

Results change if CPU fully loaded: Ø 1,97 ms

Top:



Top with load: Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Oscilloscope picture:

Ein Bild, das Screenshot, Menschliches Gesicht, Person, Quadrat enthält.

Automatisch generierte Beschreibung

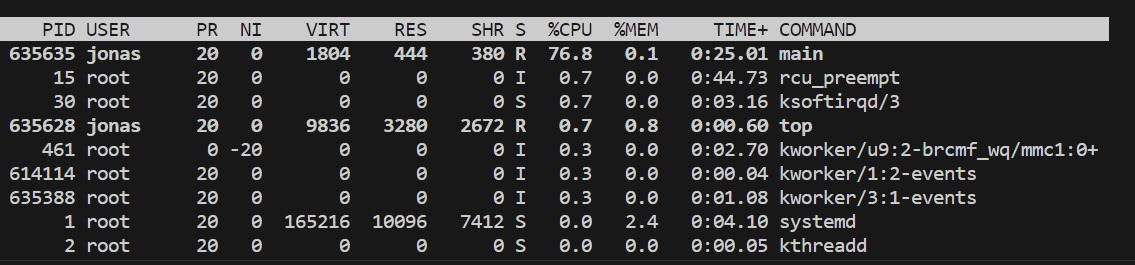
1. **Sysfs from C++ application polling pin status using a timed read()**

Response Time: Ø 213.9 µs

CPU Load: 76.8 %

Results change if CPU fully loaded: Ø 205.3 µs

Top:

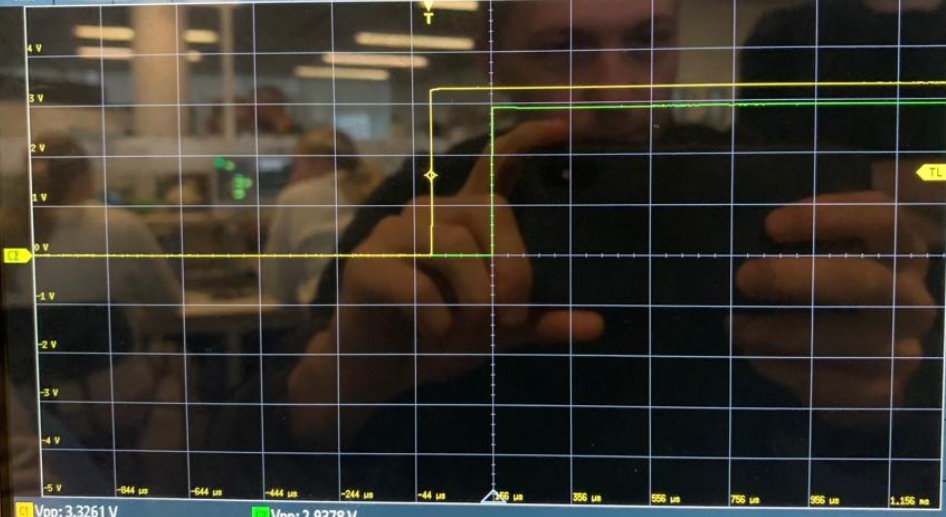


Top with load:

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Oscilloscope picture:



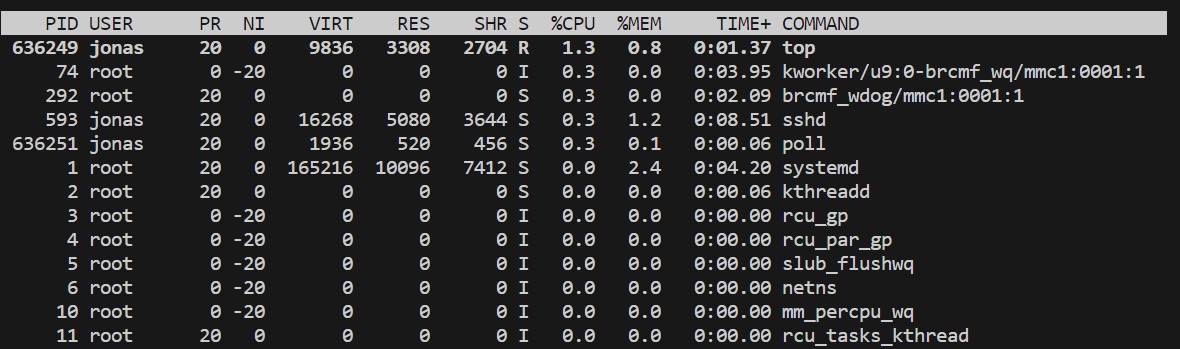
1. **Sysfs from C++ application using poll() function provided by kernel**

Response Time: Ø 397.1 µs

CPU Load: 0.3 %

Results change if CPU fully loaded: Ø 262.7 µs

Top:

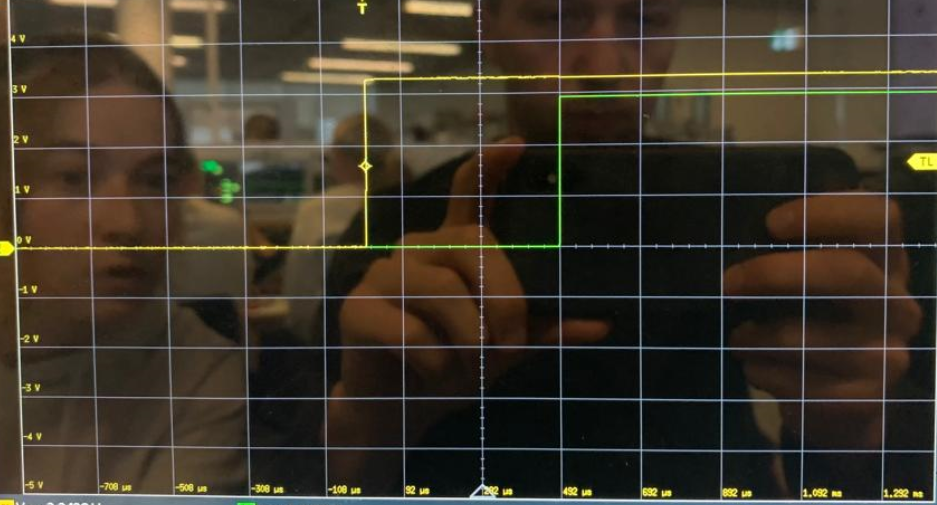


Top with Load:

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Oscilloscope picture:



1. **Kernel modul using interrupts**

Response Time: Ø 18.91 µs

CPU Load: 0%

Results change if CPU fully loaded: Ø 11.03 µs

Top:

Ein Bild, das Text, Screenshot, Karte Menü enthält.

Automatisch generierte Beschreibung

Top with load:

Ein Bild, das Text, Screenshot, Software, Zahl enthält.

Automatisch generierte Beschreibung

Oscilloscope picture:Ein Bild, das Screenshot, Text, Multimedia-Software, Software enthält.

Automatisch generierte Beschreibung

* **Which mechanism would suffice for counting encoder pulses?**  
  The encoder period is 664.45 µs, so every mechanism except from the *Sysfs from shell script* is valid for the counting.

**Part 2:**

**Speed print:**

ref. rps: 1.000000, act. rps: 0.952381, duty: 0.026125  
ref. rps: 1.000000, act. rps: 0.952381, duty: 0.026125  
ref. rps: 1.000000, act. rps: 1.428571, duty: -0.235129  
ref. rps: 1.000000, act. rps: 0.952381, duty: 0.026125  
ref. rps: 1.000000, act. rps: 0.952381, duty: 0.026125  
ref. rps: 1.000000, act. rps: 0.952381, duty: 0.026125  
ref. rps: 1.000000, act. rps: 0.952381, duty: 0.026125  
ref. rps: 1.000000, act. rps: 0.952381, duty: 0.026125  
ref. rps: 1.000000, act. rps: 0.952381, duty: 0.026125  
ref. rps: 1.000000, act. rps: 1.428571, duty: -0.235129  
ref. rps: 1.000000, act. rps: 0.952381, duty: 0.026125  
ref. rps: 1.000000, act. rps: 0.952381, duty: 0.026125  
ref. rps: 1.000000, act. rps: 0.952381, duty: 0.026125  
ref. rps: 1.000000, act. rps: 0.952381, duty: 0.026125  
ref. rps: 1.000000, act. rps: 0.952381, duty: 0.026125  
ref. rps: 1.000000, act. rps: 0.952381, duty: 0.026125  
ref. rps: 1.000000, act. rps: 1.428571, duty: -0.235129  
ref. rps: 1.000000, act. rps: 0.952381, duty: 0.026125  
ref. rps: 1.000000, act. rps: 0.952381, duty: 0.026125  
ref. rps: 1.000000, act. rps: 0.952381, duty: 0.026125  
ref. rps: 1.000000, act. rps: 0.952381, duty: 0.026125

**Control rate:**

**Duty Cycle:**

Peaks down are the jitter.

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

The results weren`t affected in our case. In the table it is visible that the injected busy task takes up 99.3% CPU load. However, the controller worked fine.

Injecting the busy task before starting the main task to control the motor would result in the motor not spinning at all.

GitHub: https://github.com/Marlenexyz/EMBE-Group

YouTube: https://www.youtube.com/watch?v=buABB3gQtAQ