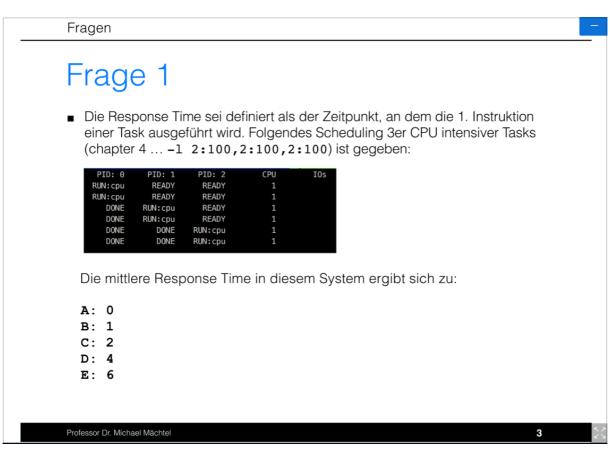
#uni/semester3/Betriebssysteme/vorlesung20_10_2020

- 1. Wie kann der user mode den system calls Parameter eingeben, wenn er gar kein Zugang zu der Funktion hat?
 - → Die werden in kernel stack geladen.
- 2. Steht in der Trap table nur Pointer auf Systemfunktionen?
 - → Jaein
- 3. Beim Context Wechsel wird floating point unit nicht gesichert, da der Kernel floating point frei läuft.
- 4. Man kann str c Signale and laufende Prozesse überschreiben.

Fragen



Fragen

Frage 2

Bewerten Sie folgende Aussage:
 Die C Systembibliothek ist Teil des Betriebssystemkernels.

Richtig

Falsch

Polling options

→ Falsch

Fragen

Frage 3

- Welche Aussage ist richtig?
 - A: fork erzeugt einen Kindprozess, der den Programmcode ab Zeile 1 des Programms ausführt.
 - B: fork erzeugt einen Kindprozess, der eine exakte Kopie des Elternprozesses ist.
 - C: Der fork Systemaufruf darf nicht mit dem Fehlercode -1 returnieren, da sonst die Systemstabilität gefärdet ist.

→ B, C auf jeden Fall falsch, es gefährdet nicht die Systemstabitilität. Die Benutzer Stabilität vielleicht schon.

Andere Lösung wie man richtig die Zeit berechnen kann.

```
struct timespec * diff_time( struct timespec before, struct timespec after,
struct timespec *result )
```

```
{
if (result==NULL)
    return NULL;
if ((after.tv_sec<before.tv_sec) || ((after.tv_sec==before.tv_sec) &&</pre>
     (after.tv_nsec<=before.tv_nsec))) { /* after before before */</pre>
      result.tv_sec = result.tv_nsec = 0;
}
result->tv_sec = after.tv_sec - before.tv_sec;
result->tv_nsec= after.tv_nsec- before.tv_nsec;
if (result->tv_nsec<0) {</pre>
     result->tv_sec--;
     /* result->tv_nsec is negative, therefore we use "+" */
     result->tv_nsec = NANOSECONDS_PER_SECOND+result->tv_nsec;
}
    return result;
}
```

Fragen die ich noch hätte:

- → Warum ist context switch kleiner, als der systemcall?
- → Warum wenn man println verwendet, wird dass nur zuerst in Parent ausgeführt, und dann im child. Mit sleep, wird das dann abwechselnd gemacht.