Betriebssysteme, Übung 6

Prof. Dr. Jan Dünnweber

Verteilte Systeme und Betriebssysteme

Quelltext für Lese- und Schreib-Prozesse in UNIX

```
#include <unistd.h>
#include <errno.h>
#include <stdio.h>
#include <string.h>
#include <sys/mman.h>
int main(void) {
  pid_t pid; /* variable to record process id of child */
  char *shared_memory;   /* shared memory base address */
  int i_parent, i_child; /* index variables            */
  int value;             /* value read by child        */
  shared_memory = mmap( 0, sizeof(int), PROT_READ | PROT_WRITE,
                    MAP_ANONYMOUS | MAP_SHARED, -1, 0 );
  if (shared_memory == (char *)-1) { fprintf( stderr, "%s\n",
     strerror(errno) ); return errno;  }
```

Quelltext (Fortsetzung)

```
if ((pid = fork( )) < 0) { /* apply fork and check for error */
  fprintf(stderr, "%s\n", strerror(errno)); return errno;  }
if (0 == pid) { /* child process */
  printf ("The␣child␣process␣begins.\n");
  for (i_child = 0; i_child < 10; i_child++) {
    sleep(1); /* wait for memory to be updated */
    value = *shared_memory;
    printf("Child's␣report:␣current␣value␣=␣%2d\n", value); }
  printf("The␣child␣is␣done\n");
} else {  /* ... */
```

Quelltext (Fortsetzung)

```
/* ... parent process: */
int childExitStatus;
printf("The␣parent␣process␣begins.\n");
for (i_parent = 0; i_parent < 10; i_parent++) {
  /* write into shared memory */
  *shared_memory = i_parent * i_parent;
  printf("Parent's␣report:␣current␣index␣=␣%2d\n", i_parent );
  sleep(1); /* wait for child to read value */ }
wait(&childExitStatus);
printf("The␣parent␣is␣done\n"); }
return 0; }
```

Experimente mit Shared Memory

- Übersetzen Sie den Quelltext von den vorigen Folien, untersuchen Sie die Ausgabe und erklären Sie diese
- Entfernen sie den sleep-Aufruf aus dem Kindprozess und beschreiben Sie den Effekt
- Fügen Sie den sleep-Aufruf im Kindprozess wieder hinzu und untersuchen Sie, was passiert, wenn Sie diesen im Elternprozess entfernen
- Überlegen Sie sich, wie die Koordination der beiden Prozesse mittels *Sperren* anstelle des sleep hätte implementiert werden können
  - Machen Sie sich mit der Funktionsweise von Sperren vertraut: `http://de.wikipedia.org/wiki/Spinlock`
  - **Tipp:** Der Elternprozess sollte in das shared-memory word schreiben, wenn es den Wert -1 enthält. Anderenfalls sollte es der Kindprozess auslesen