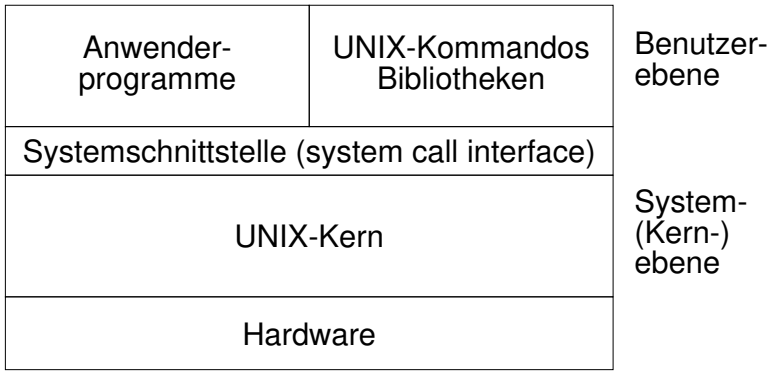
3 Cache-Speicher

C. Schimmel: "UNIX Systems for Modern Architectures", Addison-Wesley Publishing Company, 1994.

3.1 Überblick über relevante UNIX-Interna

❏ Grobstruktur

Benutzerebene - Systemebene



❏ Programm

Enthält auszuführende Befehlsfolge

❏ Prozeß

Ausführung eines Programms,
besteht aus Programm, Datenbereich, Hardwarezustand und Adreßraum

❏ Thread

Sequentielle Befehlsausführung in einem Prozeß,
besteht aus Programm, Datenbereich, Hardwarezustand

© 1996, Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

Reproduktion jeder Art oder Verwendung dieser Unterlage zu Lehrzwecken außerhalb der Universität Erlangen-Nürnberg ist ohne Genehmigung des Autors nicht zulässig

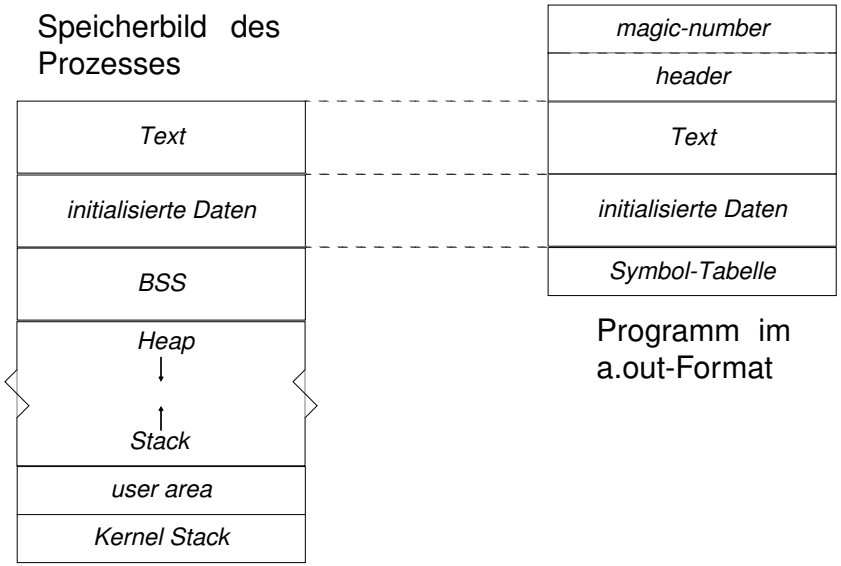
❏ Auszug aus der Systemschnittstelle eines Solaris-Kerns (/usr/include/sys/syscall.h):

```
#define     SYS_syscall   0
#define     SYS_exit      1
#define     SYS_fork      2
#define     SYS_read      3
#define     SYS_write     4
#define     SYS_open      5
#define     SYS_close     6
#define     SYS_wait      7
#define     SYS_creat     8
#define     SYS_link      9
#define     SYS_unlink    10
#define     SYS_exec      11
#define     SYS_chdir     12
#define     SYS_time      13
#define     SYS_mknod     14
#define     SYS_chmod     15
#define     SYS_chown     16
#define     SYS_brk       17
#define     SYS_stat      18
#define     SYS_lseek     19
#define     SYS_getpid    20
#define     SYS_mount     21
#define     SYS_umount    22
```


© 1996, Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

Reproduktion jeder Art oder Verwendung dieser Unterlage zu Lehrzwecken außerhalb der Universität Erlangen-Nürnberg ist ohne Genehmigung des Autors nicht zulässig

❏ Speicherorganisation eines Prozesses

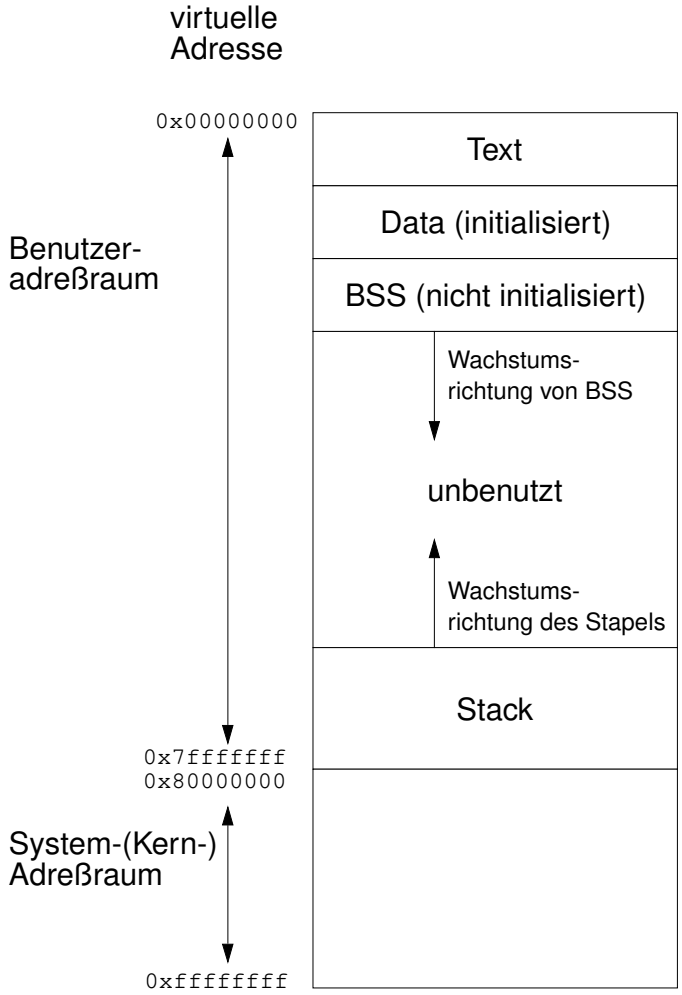


© 1996, Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

Reproduktion jeder Art oder Verwendung dieser Unterlage zu Lehrzwecken außerhalb der Universität Erlangen-Nürnberg ist ohne Genehmigung des Autors nicht zulässig

❏ Der Adreßraum von Prozessen

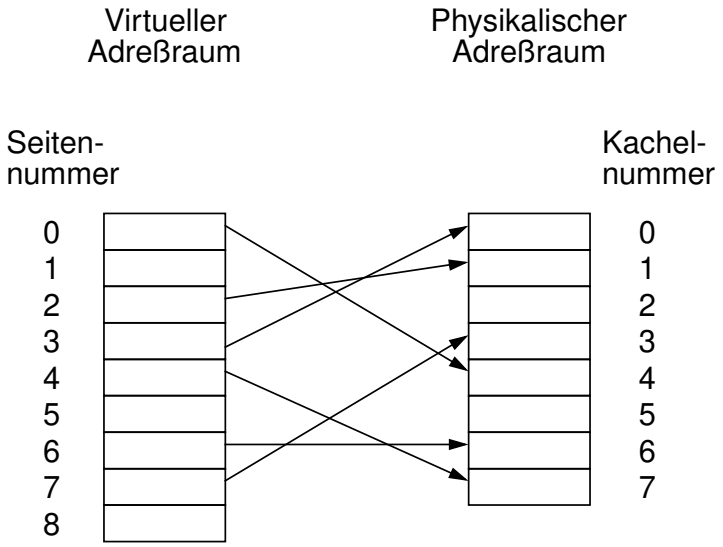
Jeder Prozeß besitzt einen eigenen Adreßraum



© 1996, Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

Reproduktion jeder Art oder Verwendung dieser Unterlage zu Lehrzwecken außerhalb der Universität Erlangen-Nürnberg ist ohne Genehmigung des Autors nicht zulässig

❏ Adreßraumabbildung



© 1996, Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

Reproduktion jeder Art oder Verwendung dieser Unterlage zu Lehrzwecken außerhalb der Universität Erlangen-Nürnberg ist ohne Genehmigung des Autors nicht zulässig

❏ Kontext-Umschaltung

Sichern und Neueinstellen

- der Prozessorregister
- des Befehlszählers
- des Kellerzeigers
- der Adreßraumabbildung

❏ Thread-Umschaltung

Sichern und Neueinstellen

- der Prozessorregister
- des Befehlszählers
- des Kellerzeigers

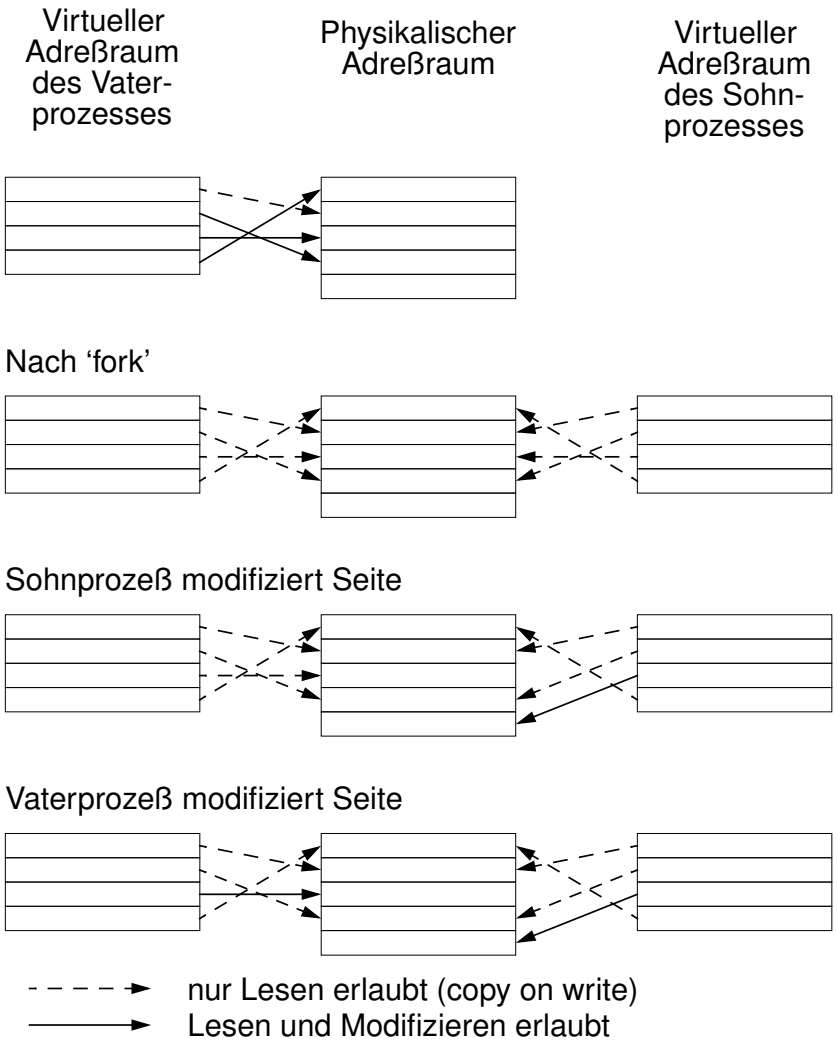
aber Beibehaltung der Adreßraumabbildung

© 1996, Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

Reproduktion jeder Art oder Verwendung dieser Unterlage zu Lehrzwecken außerhalb der Universität Erlangen-Nürnberg ist ohne Genehmigung des Autors nicht zulässig

❑ Systemaufrufe zur Speicher- und Prozeßverwaltung

• Der ‘fork’-Aufruf unter Benutzung von ‘copy on write’



© 1996, Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

Reproduktion jeder Art oder Verwendung dieser Unterlage zu Lehrzwecken außerhalb der Universität Erlangen-Nürnberg ist ohne Genehmigung des Autors nicht zulässig

• Der ‘exec’-Aufruf

Ändert den Programmteil eines Prozesses, die Prozessorregister, den Befehlszähler und die Adreßraumabbildung.

behält den Zustand bzgl. der meisten Unix-Dienste bei, wie geöffnete Dateien, ‘home directory’, ...

• Der ‘exit’-Aufruf

Bewirkt zusammen mit der Tilgung eines Prozesses die Freigabe seines Adreßraums und das Abkoppeln von Unix-Diensten, z. B. durch Schließen seiner geöffneten Dateien.

Überführung in den Zombie-Zustand

• Der ‘sbrk’- und ‘brk’-Aufruf

Verändern die Größe des BSS-Segmentes.

sbrk: Parameter gibt Veränderung der Segmentlänge an

brk: Parameter gibt die kleinste, von BSS nicht mehr benutzte virtuelle Adresse an

© 1996, Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

Reproduktion jeder Art oder Verwendung dieser Unterlage zu Lehrzwecken außerhalb der Universität Erlangen-Nürnberg ist ohne Genehmigung des Autors nicht zulässig

- Gemeinsamer Speicher (shared memory)

  Einrichtung, indem in mehreren Adreßraum-abbildungen Verweise auf die gleiche Kachel eingetragen werden, möglicherweise unter unterschiedlichen virtuellen Adressen.

  Wird vor allem zur schnellen Kommunikation zwischen Prozessen genutzt.

  Systemaufrufe:

  shared memory operations

```
char *shmat(int shmid, char *shmaddr,
            int shmflg)
int shmdt(char *shmaddr)
```

  shared memory control operations

```
int shmctl (int shmid, int cmd,
            struct shmid_ds *buf)
```

  get shared memory segment identifier

```
int shmget(key_t key, int size, int shmflg)
```

- E/A-Operationen

  gepuffert - ungepuffert

- ‘mapped files’

  Verwendbar wie Felder eines Prozesses

  Wirken logisch ähnlich wie ‘shared memory’

© 1996, Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

Reproduktion jeder Art oder Verwendung dieser Unterlage zu Lehrzwecken außerhalb der Universität Erlangen-Nürnberg ist ohne Genehmigung des Autors nicht zulässig