


```
struct sched_param {
pthread_attr_t schedschedparam ( &a, &p ); // read parameter
// set p.sched_priority
pthread_attr_t schedschedparam ( &a, &p );
pthread_create ( &id, &a, pthread_function, argument );
pthread_attr_destroy ( &a ); // destroy attributes
}
```

7. MUTEX UND SEMAPHORE

Jeder Thread hat seinen eigenen Instruction-Pointer und Stack-Pointer. Wenn Ergebnisse von der *Systemaufrufen* abhängigen Prozeduren abhängen, spricht man von einer *Race Condition* an. Threads müssen *synchronisiert* werden, damit keine *Race Condition* entsteht.

Critical Section: Code-Bereich, in dem Daten mit anderen Threads *geteilt* werden. Muss unbedingt synchronisiert werden.

Atomare Instruktionen: Eine atomare Instruktion kann vom Prozessor *unterbrechungsfrei* ausgeführt werden. *Achtung:* Selbst einzelne Assembly-Instruktionen nicht immer *atomar*

Anforderungen an Synchronisations-Mechanismen: Gegenseitiger Ausschluss (Nur ein Thread darf in Critical Section sein), **Fortschritt** (Entw. wenn in die Critical Section darf, muss in kritischer Zeit getroffen werden), **Begrenztes Warten** (Thread darf nicht überleben, bevor er in die Critical Section darf).

Implementierung: Nur mit HW-Unterstützung möglich. Es gibt zwei atomare Operationen, Test-And-Set (liest einen int auf und returnt den vorherigen Wert: test_and_set(int &target) {int value = *target; *target = 1; return value;}) und Compare-and-Swap (Überschreibt ein int auf einem spezialisierten Wert, wenn dieser dem erwarteten Wert entspricht: compare_and_swap (int &int, int expected, int new, &int) {int value = *int; *int = expected; if (&int == new, &int) return value;})

7.1. SEMAPHORE

Enthält Zähler $z \geq 0$. Wird nur über *Post(v)* (Erhöhe z um v) und *Wait(v)* (Zugriff wenn $z > 0$, verringert z um v und fährt fort. Wenn $z = 0$, setzt den Thread in waiting, bis anderer Thread z erhöht).

*int sem_wait (sem_t *sem);* *int sem_post (sem_t *sem);* *int sem_trywait (sem_t *sem);* *int sem_timedwait (sem_t *sem, struct timespec *abs_timeout);* Sind wie *sem_wait*, aber *brechen* ab, falls Dekrement nicht durchgeführt werden kann. *sem_trywait* bricht sofort ab, *sem_timedwait* nach der angegebenen Zeitdauer. *int sem_destroy (sem_t *sem);* *Entfernt Speicher*, den das OS mit *sem* *assoziiert* hat.

semaphore free = n;
semaphore used = 0;
// Producer
while (1) {
 // Warte, falls Customer zu langsam
 WAIT (free); // Hat es Platz in Queue?
 produce_item (&buffer[i], ...);
 POST (used); // 1 Element mehr in Queue
 m = (w+1) % BUFFER_SIZE;
}
// Consumer
while (1) {
 // Warte, falls Producer zu langsam
 WAIT (used); // Hat es Elemente in Queue?
 consume (&buffer[i], ...);
 POST (free); // 1 Element weniger in Queue
 n = (n+1) % BUFFER_SIZE;
}

7.2. MUTEXE

Ein Mutex hat einen *binden Zustand* z , der nur durch zwei Funktionen verändert werden kann: *Acquire (Mutex &m = 0, setz z auf 1 und fahre fort. Wenn z = 1, blockiert den Thread, bis z = 0), Release (setz z = 0).* Auch als non-blocking-Function: *int pthread_mutex_trylock (pthread_mutex_t *mutex)*

// Beispiel Initialisierung
pthread_mutex_t mutex; // global
int main() {
 // Beispiel Verwendung in Threads
 *void *thread_function (void *args) {*
 while (running) { ...
 // Enter critical section:
 pthread_mutex_lock (&mutex);
 // Run threads & wait for them to finish
 pthread_mutex_unlock (&mutex);
 // Leave critical section:
 pthread_mutex_unlock (&mutex); ...}}

Priority Inversion: Ein *hoch-priorisierter* Thread C wartet auf eine Ressource, die von einem *niedrig-priorisierten* Thread A gehalten wird. Ein Thread mit Priorität zwischen diesen beiden Threads erhält den Prozessor. Kann mit *Priority Inheritance* gelöst werden: Die Priorität von A wird temporär auf die Priorität von C gesetzt, damit der Mutex schnell wieder freigegeben wird.

8. SIGNALS, PIPES UND SOCKETS

8.1. SIGNALS

Signale ermöglichen es, einen Prozess von *aussen* zu unterbrechen, wie ein *Interrupt*. *Unterbrechen* des gerade laufenden Prozesses/Threads, Auswahl und Ausführen der *Signal-Handler-Funktionen*, *Fortssetzen* des Prozesses. Werden über ungültige Instructions oder Abbruch auf Seitens Benutzer ausgelöst. Jeder Prozess hat pro Signal einen Handler.

Alarm: *Ignore-Handler* (ignoriert das Signal), *Terminate-Handler* (benötigt das Programm), *Abnormal-Terminate-Handler* (benötigt Programm und, wenn es nicht funktioniert, direkt wie waiting), *SIGSTOP* (Wie *SIGQUIT*, aber kann nicht ignoriert oder abgefangen werden), *SIGCONT* (Setzt den Prozess fort)

Signale von der Shell senden: *kill 1234 5678* sendet SIGTERM an Prozess 1234 und 5678

*int sigaction (int signal, struct sigaction *new, struct sigaction *old);* Definiert Signal-Handler für *signal*, wenn *new* $\neq 0$. (Eigene Signal-Handler definieren via *sigaction* statt *sa_handler*: zu laufende Funktion, *sa_mask*: Blockierte Signale während Ausführung, *sa_handler* nur durch *sigset(0)*-Funktion: *sigemptyset*, *sigfillset*, *sigdelset*, *sigisset*, *sigismember*)

8.2. PIPES

Eine geöffnete Datei entspricht einem *Eintrag* in der *File-Descriptor-Tabelle (FDT)* im Prozess. Zugriff über *File-API* (nur *read* und *write*). Das OS speichert die *Eintrag der Prozess-FD* in einem *Verweis auf die globale FDT*. Bei *fork(0)* wird die FDT auch kopiert.

int dup (int source_fd); *int dup2 (int source_fd, int destination_fd);* *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup (int source_fd); *int _dup2 (int source_fd, int destination_fd);* *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup3 (int source_fd, int destination_fd, int flags); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup4 (int source_fd, int destination_fd, int flags, int fd); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup5 (int source_fd, int destination_fd, int flags, int fd, int fd2); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup6 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup7 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup8 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup9 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup10 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup11 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup12 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup13 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup14 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup15 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup16 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup17 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup18 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup19 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup20 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup21 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup22 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup23 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup24 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup25 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup26 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup27 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup28 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup29 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup30 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup31 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup32 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup33 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29, int fd30); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup34 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29, int fd30, int fd31); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup35 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29, int fd30, int fd31, int fd32); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup36 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29, int fd30, int fd31, int fd32, int fd33); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup37 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29, int fd30, int fd31, int fd32, int fd33, int fd34); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup38 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29, int fd30, int fd31, int fd32, int fd33, int fd34, int fd35); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup39 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29, int fd30, int fd31, int fd32, int fd33, int fd34, int fd35, int fd36); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup40 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29, int fd30, int fd31, int fd32, int fd33, int fd34, int fd35, int fd36, int fd37); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup41 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29, int fd30, int fd31, int fd32, int fd33, int fd34, int fd35, int fd36, int fd37, int fd38); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup42 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29, int fd30, int fd31, int fd32, int fd33, int fd34, int fd35, int fd36, int fd37, int fd38, int fd39); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup43 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29, int fd30, int fd31, int fd32, int fd33, int fd34, int fd35, int fd36, int fd37, int fd38, int fd39, int fd40); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup44 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29, int fd30, int fd31, int fd32, int fd33, int fd34, int fd35, int fd36, int fd37, int fd38, int fd39, int fd40, int fd41); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup45 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29, int fd30, int fd31, int fd32, int fd33, int fd34, int fd35, int fd36, int fd37, int fd38, int fd39, int fd40, int fd41, int fd42); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup46 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29, int fd30, int fd31, int fd32, int fd33, int fd34, int fd35, int fd36, int fd37, int fd38, int fd39, int fd40, int fd41, int fd42, int fd43); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup47 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29, int fd30, int fd31, int fd32, int fd33, int fd34, int fd35, int fd36, int fd37, int fd38, int fd39, int fd40, int fd41, int fd42, int fd43, int fd44); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup48 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29, int fd30, int fd31, int fd32, int fd33, int fd34, int fd35, int fd36, int fd37, int fd38, int fd39, int fd40, int fd41, int fd42, int fd43, int fd44, int fd45); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup49 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29, int fd30, int fd31, int fd32, int fd33, int fd34, int fd35, int fd36, int fd37, int fd38, int fd39, int fd40, int fd41, int fd42, int fd43, int fd44, int fd45, int fd46); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup50 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29, int fd30, int fd31, int fd32, int fd33, int fd34, int fd35, int fd36, int fd37, int fd38, int fd39, int fd40, int fd41, int fd42, int fd43, int fd44, int fd45, int fd46, int fd47); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup51 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29, int fd30, int fd31, int fd32, int fd33, int fd34, int fd35, int fd36, int fd37, int fd38, int fd39, int fd40, int fd41, int fd42, int fd43, int fd44, int fd45, int fd46, int fd47, int fd48); *Duplizieren* den File-Descriptor *source_fd*. *dup2* alloziert einen neuen FD, *dup2* überschreibt *destination_fd*.
int _dup52 (int source_fd, int destination_fd, int flags, int fd, int fd2, int fd3, int fd4, int fd5, int fd6, int fd7, int fd8, int fd9, int fd10, int fd11, int fd12, int fd13, int fd14, int fd15, int fd16, int fd17, int fd18, int fd19, int fd20, int fd21, int fd22, int fd23, int fd24, int fd25, int fd26, int fd27, int fd28, int fd29, int fd30, int fd31, int fd32, int fd33, int fd34, int fd35, int fd36, int fd