Implementation of a user level threading.

Threads:

Thread ThreadCreate (void(*start_funct)(void *), void *args)

This function creates a new Thread. The parameter start_func is the function in which the new thread starts executing. The parameter args is passed to the start function. This routine does not pre-empt the invoking thread. In others words the parent (invoking) thread will continue to run; the child thread will sit in the ready queue.

---

void ThreadYield(void)

Suspends execution of invoking thread and yield to another thread. The invoking thread remains ready to execute—it is not blocked. Thus, if there is no other ready thread, the invoking thread will continue to execute.

---

void ThreadYield(void)

Suspends execution of invoking thread and yield to another thread. The invoking thread remains ready to execute—it is not blocked. Thus, if there is no other ready thread, the invoking thread will continue to execute.

---

int ThreadJoin(Thread thread)

Joins the invoking function with the specified child thread. Returns 0 on success. It returns -1 on failure. Failure occurs if specified thread is not an immediate child of invoking thread.

---

void ThreadJoinAll(void)

Waits until all children have terminated. Returns immediately if there are no active children.

---

void ThreadExit(void)

Terminates the invoking thread. Note: all Threads are required to invoke this function. Do not allow functions to “fall out” of the start function.

Semaphores:

Semaphore SemaphoreInit(int initv)

Create a semaphore. Set the initial value to initv, which must be non-negative. A positive initial value has the same effect as invoking SemaphoreSignal the same number of times. On error it returns NULL.

---

void SemaphoreSignal(Semaphore sem)

Signal semaphore sem. The invoking thread is not pre-empted.

---

void SemaphoreWait(Semaphore sem)

Wait on semaphore sem.

---

int SemaphoreDestroy(Semaphore sem)

Destroy semaphore sem. Do not destroy semaphore if any threads are blocked on the queue.

Return 0 on success, -1 on failure.