Universidade Federal do Rio Grande do Sul
Instituto de Informática
INF01142 - Sistemas Operacionais I N - 2018/1 - Turma B
Prof. Dr. Sérgio Luis Cechin
Trabalho Prático I - Documentação

Integrantes:
Cassio Ramos - 193028
Giovani Tirello - 252741
Maicon Vieira - 242275

1. **Decrição Geral:**

A biblioteca cthread fornece um conjunto de funções que possibilitam a programação com threads, desde a sua criação, execução, sincronização, término, trocas de contexto, entre outras funcionalidades. A ideia por trás da implementação dessas funções está descrita abaixo.

2. **Funções:**

- cidentify:

Função básica que retorna o nome e o número do cartão de cada integrante do grupo. Se o valor passado como parâmetro 'size' for menor que o tamanho da string que contém a identificação dos integrantes, é retornada uma mensagem de erro;

- ccreate:

Primeiramente é testado se é a primeira vez que uma thread é criada, pois se for verdade, é necessário inicializar o escalonador e as variáveis de filas (apto, apto suspenso, bloqueado, bloqueado suspenso e executando). Em seguida, cria a thread e inicializa seus campos e seu contexto. Por fim, insere a thread na fila de aptos.

- cyield:

A função testa se existe uma thread em execução e, caso verdade, insere-a na fila de aptos, remove-a da fila de execução e faz a troca de contexto para o escalonador, que irá escolher qual próxima thread será executada.

- cjoin:

A função recebe o identificador da thread cujo término deve ser aguardado. Assim, analisa se a fila de execução não está vazia, salva thread da fila de execução e, caso não seja um auto-join, procura nas filas de apto, apto suspenso, bloqueado e bloqueado suspenso para saber se a thread do parâmetro tid existe (thread ainda não terminou execução). Se encontra, procura na fila de bloqueado e de bloqueado suspenso se existe alguma thread esperando pela thread do parâmetro tid, se não encontrou o join é feito, inserindo a thread na fila de bloqueado, removendo da fila de execução e fazendo a troca de contexto. Para fazer o join, foi necessário adicionar um campo da struct TCB_t chamado waitingThreadID, que receberá o parâmetro tid passado para a função cjoin caso seja possível fazer o join.

- csuspend:

A função recebe o identificador da thread a ser suspensa. Primeiramente, vverifica se não é um auto-suspend. Após, procura na fila de apto e na fila de bloqueado se encontra a thread recebida como parâmetro; se encontrou na fila de apto, remove dessa fila, adiciona na fila de apto suspenso e muda seu estado; se encontro na fila de bloqueado, remove dessa fila, adiciona na fila de bloqueado suspenso e muda seu estado.

- cresume:

A função recebe o identificador da thread a ser resumida. Primeiramente, verifica se não é um auto-resume. Após, procura na fila de apto suspenso e na fila de bloqueado suspenso se encontra a thread recebida como parâmetro; se encontrou na fila de apto suspenso, remove dessa fila, adiciona na fila de apto e muda seu estado; se encontro na fila de bloqueado suspenso, remove dessa fila, adiciona na fila de bloqueado e muda seu estado.

- csem_init:

Essa função basicamente associa o parâmetro ‘count’ ao respectivo campo do semáforo recebido como parâmetro, aloca o semáforo na memória e inicializa a fila do semáforo.

- cwait:

Essa função decrementa o campo ‘count’ do semáforo e verifica se esse campo é maior que zero; se for, o recurso está livre e a thread pode continuar execução, se não, salva a thread em execução, muda seu estado para bloqueado, remove da fila de execução, adiciona na fila de bloqueado, adiciona na fila do semáforo e faz a troca de contexto.

- csignal:

Essa função incrementa o campo ‘count’ do semáforo e procura a thread da fila do semáforo na fila de bloqueado; se encontra, remove thread da fila do semáforo e da fila de bloqueado, adiciona na fila de apto e muda estado da thread para apto; se não encontra na fila de bloqueado, procura na fila de bloqueado suspenso e, se encontra, remove thread da fila do semáforo e da fila de bloqueado suspenso, adiciona na fila de apto suspenso e muda estado da thread para apto suspenso.

- dispatcher:

Caso a thread atualmente em execução terminou sua execução, essa função remove a thread da fila de execução, muda seu estado para término e procura nas filas de bloqueado e bloqueado suspenso se existe alguma thread que estava esperando por essa thread terminar; se encontrar, move a thread ou da fila de bloqueado para a fila de apto, ou da fila de bloqueado suspenso para a fila de apto suspenso; após isso, libera a thread que terminou execução da memória e escolhe outra thread na fila de apto para ser executada. Se thread em execução não terminou execução, apenas move essa thread para a fila de apto e move outra thread da fila de apto para a fila de execução.

- init_scheduler:

Essa função cria todas as filas de execução, apto, apto suspenso, bloqueado e bloqueado suspenso, inicializa o contexto do escalonador e cria a thread main com identificador igual a zero, inserindo-a na fila de execução.

3. **Testes e biblioteca:**

Mesmo que a biblioteca libcthread.a e alguns programas de teste tenham sido disponibilizados, não foi possível rodar os programas de teste para garantir a funcionalidade esperada das funções implementadas.