Projeto 01 - Troca de Contextos

Kelvin James
Rafael Lammel Marinheiro
Sistemas de Informação - UTFPR
06 de maio de 2019

1. Questões

1.1 Objetivo e parâmetros das funções

- **getcontext()**: Tem como parâmetro um ponteiro do tipo ucontext_t, fazendo com que ele aponte para o contexto atualmente ativo; Em sucesso retorna 0, em falha retorna -1;

- **setcontext()**: Tem como parâmetro um ponteiro constante do tipo ucontext_t; Ela restaura o contexto passado por esse ponteiro; Em sucesso não retorna nada, em falha retorna -1; O contexto tem que ter sido obtido pela função getcontext() ou makecontext(); Se pela função getcontext(), ele continua a execução do código; caso tenha sido obtido por makecontext(), vai executar a função que makecontext() recebe por parâmetro antes de prosseguir com o código;

- **swapcontext()**: Tem como parâmetros um ponteiro de ucontext_t e um ponteiro constante de ucontext_t. A função salva o contexto atual, que é apontado pelo primeiro parâmetro, e ativa o contexto para o qual o segundo parâmetro aponta;

- **makecontext( )**: Modifica o contexto da variável ucontext_t (inicializada com getcontext( )) passada como argumento, fazendo com que ao fazer a restauração dessa variável, seja chamada a função definida como argumento no makecontext( ).

1.2. Significados dos campos da estrutura ucontext_t

- **uc_stack**: A pilha onde está o contexto;
- **uc_link**: Ponteiro para o próximo contexto a ser executado quando este retornar.

1.3. Explicando as linhas onde as funções são chamadas ou a estrutura de ucontext_t é modificada em pingpong.c

Obs.: Como o enunciado não dizia qual versão do projeto referenciar, foi utilizado a referência de linhas para o código mais recente da entrega dos projetos A (Projeto 06 com prioridades).

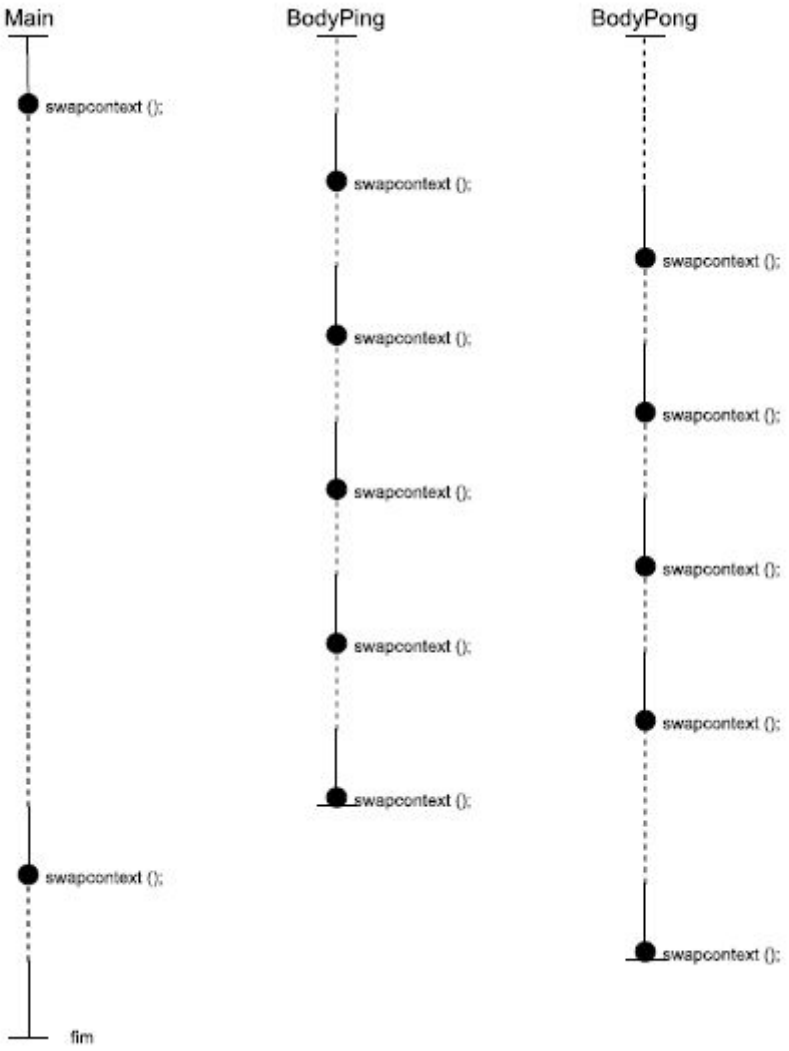
- Linha 63: É chamado a função getcontext() e passado a ela um ponteiro como parâmetro. Este ponteiro passa a apontar para o contexto atual. Esta linha salva o contexto atual no ponteiro, dentro da função task_create.

- Linhas 68 até 71: Aqui fazemos a manipulação nas estruturas ucontext_t. Na ordem de linhas:
  uc_stack.ss_sp: Recebe a base de memória alocada;
  uc_stack.ss_size: Recebe o tamanho da memória alocada;
  uc_stack.ss_flags: Recebe 0, não estamos usando flags no projeto;
  uc_link: Também recebe 0 pois não estamos usando essa função;

- Linha 83: Aqui é chamado a função makecontext(). Assim como getcontext(), ela é chamada dentro de task_create. Depois que a tarefa é salva em um ponteiro pelo getcontext(), este ponteiro é usado na função makecontext() para colocar a função da tarefa (recebida por task_create) para ser executada da próxima vez que a tarefa for chamada.

- Linha 120: A chamada de swapcontext() no código. Ele é chamado dentro da função task_switch, a qual recebe um ponteiro da próxima tarefa a ser executa, e utiliza swapcontext() para salvar a tarefa atual e executar a tarefa recebida por parâmetro.

1.4 Diagrama de Tempo



2. Referências

1. The Open Group, The Open Group Publication Server, 1997. Referência de “ucontext_t”. Disponível em: https://pubs.opengroup.org/onlinepubs/7908799/xsh/ucontext.h.html
2. GNU, Manual de Referências GNU C. “Complete Context Control”. Disponível em: https://www.gnu.org/software/libc/manual/html_node/System-V-contexts.html