Sistemas Embarcados - Trabalho Prático II

O segundo trabalho prático da disciplina de Sistemas Embarcados consiste na implementação de um programa distribuído para resolver jogos. Juntamente com o enunciado, são disponibilizados os seguintes recursos para o desenvolvimento do trabalho:

- HellfireOS (*https://www.github.com/sjohann81/hellfireos*);
- Simulador MPSoC (junto com o HellfireOS);
- Programa que resolve jogos de labirinto de tamanhos diversos (15 jogos são resolvidos pelo programa);

Sugere-se que inicialmente o programa fornecido seja portado para o ambiente do HellfireOS. Essa versão pode ser usada como base para testes, uma vez que essa apresentará os resultados corretos. Essa versão pode ter o tempo de execução medido para comparações com a versão distribuída utilizando a função *_read_us()*, que retorna uma leitura de tempo com precisão de microssegundos.

Versão distribuída: Observe as aplicações exemplo que envolvem o uso de NoC do HellfireOS. As principais primitivas da API a serem usadas são: *_read_us()*, *hf_spawn()*, *hf_cpuid()*, *hf_selfid()*, *hf_comm_create()*, *hf_send()*, *hf_recvprobe()* e *hf_recv()*. Como alternativa para comunicação, pode ser utilizada a API para RPC (funções *hf_register()* e *hf_call()*). É necessário que a API para comunicação seja compreendida antes do trabalho iniciar. Familiarize-se com a troca de mensagens usando os conceitos de endereçamento por processador, porta e canal e compreenda o modelo implementado no *kernel* para que sua aplicação seja eficiente, explorando os conceitos de computação distribuída em um ambiente embarcado.

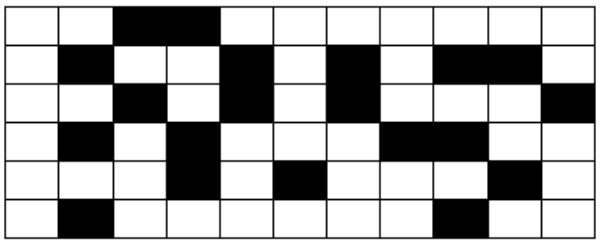
Para a versão distribuída a ser desenvolvida, a configuração do *kernel* deve ser modificada, assim como do simulador. É sugerido que sejam usadas 3 configurações entre 9 e 25 processadores (malhas 3x3, 3x4, 4x4, 4x5 e 5x5).

Lembre que o modelo de programação é distribuído e sem compartilhamento de memória, portanto é responsabilidade da aplicação distribuir os dados a serem processados (para tarefas *worker* em outros processadores) e coletar os resultados das soluções. O objetivo do trabalho é reduzir o tempo de processamento do algoritmo, e o grupo deve decidir a melhor estratégia para isso.

Entrega: Juntamente com a implementação, deve ser entregue um relatório detalhando a mesma, a abordagem utilizada para paralelizar uma solução para o problema com o objetivo de explorar a estrutura de processamento baseada em um MPSoC, o ganho em tempo de processamento quando comparado com uma solução sequencial (usar 3 configurações de MPSoC), o volume de dados enviado pela rede e as técnicas que foram utilizadas para resolver os problemas de distribuição de dados e sincronização. O trabalho deve ser realizado em grupos de dois ou três integrantes e entregue via Moodle.

Anexo - Labirinto

O objetivo do jogo é, partindo de um ponto inicial de um labirinto, chegar no destino final. Um labirinto possui o formato semelhante ao do desenho:



Um labirinto pode ser representado por uma matriz retangular L, onde o elemento l_{ij} vale 0 ou -1 conforme a casa correspondente do labirinto seja uma passagem livre ou uma parede, respectivamente.

Um modo para resolver esse problema consiste em marcar com o número k onde $(k = 1, 2, 3...)$ todas as casas livres que estejam exatamente a $k - 1$ passos de distância do destino, pelo caminho mais curto possível. Suponha que a cada passo seja possível se deslocar por apenas uma casa na vertical ou horizontal. Então, rotula-se inicialmente a posição do destino com 1 e para cada $k >= 2$ examinam-se todas as casas livres do labirinto, marcando-se

com k aquelas ainda não marcadas e que sejam adjacentes a alguma casa marcada com $k-1$.

A marcação continua até ser atingido um valor de k (28 no exemplo abaixo) tal que nenhuma casa esteja em condições de ser marcada. Ao final da marcação teremos a seguinte matriz, supondo o destino em [5,10]:

26	27	-1	-1	12	11	10	9	10	11	12
25	-1	0	0	-1	12	-1	8	-1	-1	13
24	25	-1	0	-1	13	-1	7	6	5	-1
23	-1	21	-1	15	14	15	-1	-1	4	3
22	21	20	-1	16	-1	16	17	18	-1	2
23	-1	19	18	17	18	17	18	-1	2	1

O caminho mais curto até o destino pode então ser determinado partindo-se da posição inicial e passando a cada etapa para uma casa adjacente e cuja numeração seja menor do que a atual. Por exemplo, partindo de [0,0] é necessário percorrer pelo menos 26 casas para chegar ao destino: [0,0], [1,0], [2,0], [3,0], [4,0], [4,1], [4,2], ..., [4,10], [5,10]. Nos labirintos disponibilizados, existe uma borda externa e as posições inicial e final são respectivamente o canto superior esquerdo e inferior direito.