T8: Geração de Imagem em Paralelo com CUDA

Disciplina: elc139 - Programação Paralela
Aluno: Adriano Luís de Almeida

Parte 1

cudaMallocManaged faz a alocação de memoria para substituir camada malloc do C e retorna um ponteiro acessível em qualquer processador

cudaMallocManaged(&pic, frames*width*width*sizeof(unsigned char));

Parte 1.1

Como pode ser notado na chamada da função criada e nomeada como calculate, o número de frames é usado para a criação de threads, de acordo com o enunciado.
Dentro do método calculate é utilizado mais duas chamadas como demonstrado abaixo:

calculate<<<1, frames>>>(width, frames, pic);

Parte 1.1

Dentro do método calculate é utilizado mais duas chamadas como demonstrado abaixo:

```
int index = threadIdx.x;
int offset = blockDim.x;
```

threadIdx.x Índice de segmento dentro do bloco
blockDim.x Contém as dimensões do bloco

Parte 1.1

Após a execução de toda o método calculate **cudaDeviceSynchronize();** forçará o programa a garantir que os **kernels/memcpys** dos fluxos estejam completos antes de continuar.

```
cudaDeviceSynchronize();
```

Parte 1.2

- Para rodar os testes foi criado um *notebook* no Google Colaboratory disponível !AQUI!

- Lembre-se: Para iniciar, faça uma cópia deste notebook clicando em File -> Save a copy in Drive. Faça o restante da prática usando sua cópia. Em Runtime -> Change runtime type, habilite o uso de GPU.

Parte 1.2

WEVE				WAVECUDA1		
width	frames	time sec.		width	frames	time sec.
1024	100	5.4348		1024	100	0.6502
1024	200	11.0191		1024	200	0.6784
2048	100	21.6637		2048	100	2.0165
2048	200	42.2622		2048	200	2.1749

Referências

- NVIDIA. CUDA C Programming Guide. 2019
- Mark Harris. An Even Easier Introduction to CUDA. 2017
- Mark Harris. Unified Memory for CUDA Beginners. 2017