Implementação da ferramenta Cuda para calcular o valor de PI

Robson Liesner de Lima Junior
Instituto Federal Catarinense - IFC
Videira, SC - Brasil
Email: robsonlljr@gmail.com

Resumo— Cuda é uma plataforma de computação paralela e um modelo de programação desenvolvido pela NVIDIA para computação geral em unidades de processamento gráfico (GPUs). Com o cuda implementado em conjunto com o código, pode acelerar drasticamente os aplicativos de computação aproveitando o poder das GPUs. Com isso, foi desenvolvido o cálculo do PI em que a parte sequencial da carga de trabalho é executada na CPU que é otimizada para desempenho de thresh único, enquanto a parte de computação intensiva do aplicativo é executada em milhares de núcleos da GPU em paralelo

I. Introdução

Inicialmente para implementar o Cuda é necessário ter uma placa de vídeo compatível com a ferramenta, uma versão com suporte do Microsoft Windows, uma versão com suporte do Microsoft Visual Studio e o NVIDIA CUDA Toolkit. Cuda foi desenvolvido com vários objetivos de design em mente:

- Fornece um pequeno conjunto de extensões para linguagens de programação padrão, como C, que permitem uma implementação direta de algoritmos paralelos. Com CUDA C/C++, o programador pode se concentrar na tarefa de paralelização dos algoritmos em vez de gastar tempo em sua implementação.
- Suporta computação heterogênea em que os aplicativos usam a CPU e a GPU. As partes seriais dos aplicativos são executadas na CPU e as partes paralelas são transferidas para a GPU. Como tal, o CUDA pode ser aplicado de forma incremental a aplicativos existentes. a CPU ea GPU são tratados como dispositivos separados que possuem seus próprios espaços de memória. Essa configuração também permite computação simultânea na CPU e GPU sem contenção de recursos de memória.

II. Metodologia

Basicamente, a resolução do cálculo do PI foi utilizado 2 funções, uma para calcular a área do objeto e outra para identificar se o valor recebido é um número ímpar ou par para assim se chegar no valor aproximado. Além das funções complementares se tem a MAIN() que faz a iniciação da ferramenta Cuda com a criação de 'cudaEvent_t' e alocação de memória com 'cudaMalloc'.

```
__device__ float calcularArea(float inicio, float final, float base)
{
    float medio = (inicio + final) / 2;
    float altura = 4 / (1 + (medio) * (medio));
    return base * altura;
}
```

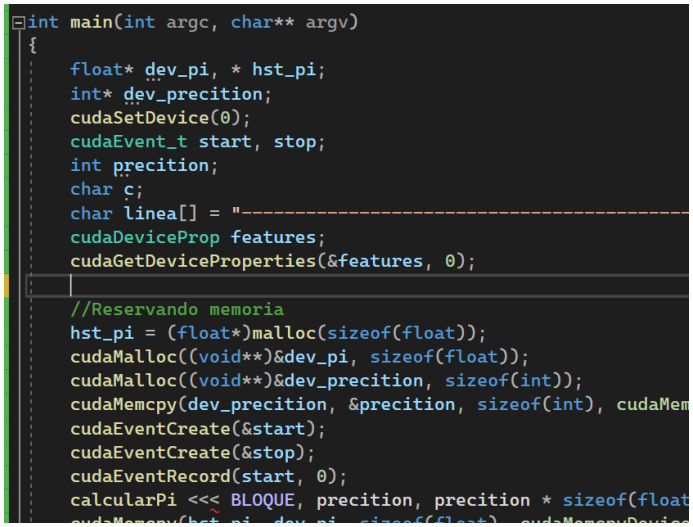
```
__global__ void calcularPi(float* pi, int* precisaoEscolhida)
{
    int lancamentos = *precisaoEscolhida;
    int identificador = threadIdx.x;
    float inicio;
    float final;
    extern __shared__ float area[];
    float superVar = 1 / (float)lancamentos;
    cuda_SYNCTHREADS();

    int* array;
    if (lancamentos % 2 != 0)
    {
        if (identificador == (lancamentos - 1))
        {
            array[0] += array[identificador];
        }
    }
    cuda_SYNCTHREADS();

    int salto = lancamentos / 2;

    while (salto)
    {
        if (identificador < salto)
        {
            area[identificador] = array[0] + array[identificador];
        }
    }
    cuda_SYNCTHREADS();

    if (identificador == 0)
    {
        *pi = area[identificador];
    }
}
```



```
int main(int argc, char** argv)
{
    float* dev_pi, * hst_pi;
    int* dev_precition;
    cudaSetDevice(0);
    cudaEvent_t start, stop;
    int precition;
    char c;
    char linea[] = "----------------------------------------
    cudaDeviceProp features;
    cudaGetDeviceProperties(&features, 0);

    //Reservando memoria
    hst_pi = (float*)malloc(sizeof(float));
    cudaMalloc((void**)&dev_pi, sizeof(float));
    cudaMalloc((void**)&dev_precition, sizeof(int));
    cudaMemcpy(dev_precition, &precition, sizeof(int), cudaMem
    cudaEventCreate(&start);
    cudaEventCreate(&stop);
    cudaEventRecord(start, 0);
    calcularPi <<< BLOQUE, precition, precition * sizeof(float
```

REFERÊNCIAS

CORPORATION, Nvidia. **CUDA Installation Guide for Microsoft Windows**. Disponível em: https://docs.nvidia.com/cuda/cuda-installation-guide-microsoft-windows/index.html. Acesso em: 01 fev. 2022.

CORPORATION, Nvidia. **CUDA C++ Programming Guide**. Disponível em: https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html. Acesso em: 01 fev. 2022.