

ARQUITECTURA DE COMPUTADORES

1.CÓDIGO

```
/*
 * ARQUITECTURA DE COMPUTADORES
 * 2º Grado en Ingenieria Informatica
 *
 * ENTREGA Básico 5:
 * >> Cálculo de  $\pi$  mediante reducción paralela
 *
 * Alumno: Antonio Alonso Briones
 * Fecha: 16/10/2024
 */

// includes
#include <stdio.h>
#include <stdlib.h>
#include <cuda_runtime.h>
#include <math.h>
#include <device_launch_parameters.h>
// Función para verificar si un número es potencia de 2
int esPotenciaDeDos(int x)
{
    return (x != 0) && ((x & (x - 1)) == 0);
}

// Kernel para calcular  $\pi$  utilizando reducción paralela
__global__ void calcular_pi(float* datos, float* resultado, int N)
{
    int myID = threadIdx.x;

    // Cada hilo calcula su término de la serie
    if (myID < N)
    {
        float n = (float)(myID + 1);
        datos[myID] = 1.0f / (n * n);
    }

    // Sincronización de hilos
    __syncthreads();

    // Reducción paralela
    int salto = N / 2;
    while (salto > 0)
    {
        if (myID < salto)
        {
            datos[myID] += datos[myID + salto];
        }
        // Sincronización de hilos
        __syncthreads();
        salto /= 2;
    }

    // El hilo 0 escribe el resultado final
    if (myID == 0)
    {
        *resultado = datos[0];
    }
}
```

```

    }
}

int main(int argc, char** argv)
{
    int N;
    printf("Introduce el número de términos (potencia de 2): ");
    scanf("%d", &N);

    // Verificar que N sea potencia de 2
    if (!esPotenciaDeDos(N))
    {
        printf("El número de términos debe ser una potencia de 2.\n");
        return 1;
    }

    // Obtener las propiedades del dispositivo
    cudaDeviceProp deviceProp;
    cudaGetDeviceProperties(&deviceProp, 0);

    // Cálculo del número total de núcleos (SP)
    int cudaCores;
    int SM = deviceProp.multiProcessorCount;
    int major = deviceProp.major;
    int minor = deviceProp.minor;
    const char* archName;
    switch (major)
    {
    case 1: // Tesla
        archName = "TESLA";
        cudaCores = 8;
        break;
    case 2: // Fermi
        archName = "FERMI";
        if (minor == 0)
            cudaCores = 32;
        else
            cudaCores = 48;
        break;
    case 3: // Kepler
        archName = "KEPLER";
        cudaCores = 192;
        break;
    case 5: // Maxwell
        archName = "MAXWELL";
        cudaCores = 128;
        break;
    case 6: // Pascal
        archName = "PASCAL";
        if (minor == 1 || minor == 2)
            cudaCores = 128;
        else
            cudaCores = 64;
        break;
    case 7: // Volta y Turing
        if (minor == 0)
            archName = "VOLTA";
        else
            archName = "TURING";
        cudaCores = 64;
        break;
    case 8: // Ampere
        archName = "AMPERE";

```

```

        cudaCores = 64;
        break;
    case 9: // Hopper
        archName = "HOPPER";
        cudaCores = 128;
        break;
    default: // Arquitectura desconocida
        archName = "DESCONOCIDA";
        cudaCores = 0;
        break;
}
int totalCores = SM * cudaCores;

// Mostrar las propiedades del dispositivo
printf("\n> Propiedades del dispositivo seleccionado:\n");
printf("Nombre: %s\n", deviceProp.name);
printf("Arquitectura CUDA: %s\n", archName);
printf("Capacidad de cómputo: %d.%d\n", major, minor);
printf("Número de multiprocesadores: %d\n", SM);
printf("Número de núcleos CUDA (%d x %d): %d\n", SM, cudaCores, totalCores);
printf("Máximo número de hilos por bloque: %d\n", deviceProp.maxThreadsPerBlock);

// Verificar que N no exceda el máximo de hilos por bloque
if (N > deviceProp.maxThreadsPerBlock)
{
    printf("El número de términos (%d) excede el máximo número de hilos por bloque (%d).\n", N, deviceProp.maxThreadsPerBlock);
    // Ajustar N al máximo permitido y que sea potencia de 2
    int temp = deviceProp.maxThreadsPerBlock;
    while (!esPotenciaDeDos(temp))
    {
        temp--;
    }
    N = temp;
    printf("Ajustando N a %d para ser potencia de 2.\n", N);
}

printf("Número de hilos lanzados: %d\n", N);

// Declaración de punteros para host y device
float* hst_datos;
float* dev_datos;
float hst_resultado;
float* dev_resultado;

// Reserva de memoria en el host
hst_datos = (float*)malloc(N * sizeof(float));

// Reserva de memoria en el device
cudaMalloc((void**)&dev_datos, N * sizeof(float));
cudaMalloc((void**)&dev_resultado, sizeof(float));

// Lanzamiento del kernel
printf("\n> Calculando...\n");
calcular_pi << <1, N >> > (dev_datos, dev_resultado, N);

// Copiar el resultado al host
cudaMemcpy(&hst_resultado, dev_resultado, sizeof(float), cudaMemcpyDeviceToHost);

// Calcular la aproximación de  $\pi$ 

```

```

float pi_aprox = sqrtf(6.0f * hst_resultado);

// Calcular el error relativo
float pi_ref = 3.141593f;
float error = fabsf(pi_ref - pi_aprox) / pi_ref * 100.0f;

// Mostrar resultados
printf("\n> RESULTADOS:\n");
printf("Valor de  $\pi$ : %.6f\n", pi_ref); // Valor de  $\pi$  conocido (6 decimales)
printf("Aproximación de  $\pi$ : %.6f\n", pi_aprox);
printf("Error relativo: %.6f%%\n", error);

// Liberar memoria
free(hst_datos);
cudaFree(dev_datos);
cudaFree(dev_resultado);

// Salida
printf("*****\n");
printf("<pulsa [INTRO] para finalizar>");
getchar();
getchar();
return 0;
}

```

2. RESULTADO DE LA COMPILACIÓN

```

Salida
Mostrar salida de: Compilación
Compilación iniciada a las 23:23...
1>----- Operación Compilar iniciada: Proyecto: Básico 5, configuración: Debug x64 -----
1>Compiling CUDA source file kernel.cu...
1>
1>C:\Users\alolan\source\repos\Básico 5>"C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v12.6\bin\nvcc.exe"
1>kernel.cu
1>tmpxft_0000933c_00000000-7_kernel.cudaf1.cpp
1>Básico 5.vcxproj -> C:\Users\alolan\source\repos\Básico 5\x64\Debug\Básico 5.exe
===== Compilación: 1 correcto, 0 erróneo, 0 actualizado, 0 omitido =====
===== Compilar completado a las 23:23 y tardó 02,355 segundos =====
|

```

3. RESULTADO DE LA DEPURACIÓN

```
kerneltwo -x
```

```
(Ámbito global)
```

```
Salida
```

```
Mostrar salida de: Depurar
```

```
'Básico 5.exe' (Win32): 'C:\Users\aloan\source\repos\Básico 5\src\Debug\Básico 5.exe' cargado. Símbolos cargados.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\ntdll.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\kernel32.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\KernelBase.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\vcruntime140.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\ucrtbased.dll' cargado.
```

```
E! subprocesso 18884 terminó con código 0 (0x0).
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\nvcuda.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\advapi32.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\msvcrt.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\sechost.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\bcrypt.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\rpcr4.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\gdi32.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\win32u.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\gdi32full.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\msvc_p_win.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\ucrtbase.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\USER32.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\imm32.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\DXCore.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\DriverStore\FileRepository\invhmi.inf_and64_c4739b7ed91feb94\nvuda64.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\shlwapi.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\version.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\masn1.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\cryptnet.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\crypt32.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\cryptbase.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\wldp.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\combase.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\oleaut32.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\dvrstore.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\devobj.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\cfgmgr32.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\cfgmgr32.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\cfgmgr32.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\cfgmgr32.dll' descargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\cfgmgr32.dll' descargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\nvapi64.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\setupapi.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\shell32.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\ole32.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\DriverStore\FileRepository\invhmi.inf_and64_c4739b7ed91feb94\nvdxdmml64.dll' cargado.
```

```
'Básico 5.exe' (Win32): 'C:\Windows\System32\kernel.appcore.dll' cargado.
```

```
E! subprocesso 17256 terminó con código 321225786 (0xc000013a).
```

```
E! subprocesso 38288 terminó con código 321225786 (0xc000013a).
```

```
E! subprocesso 37140 terminó con código 321225786 (0xc000013a).
```

```
E! subprocesso 16584 terminó con código 321225786 (0xc000013a).
```

```
E! subprocesso 28820 terminó con código 321225786 (0xc000013a).
```

```
E! subprocesso 27256 terminó con código 321225786 (0xc000013a).
```

```
E! subprocesso 35568 terminó con código 321225786 (0xc000013a).
```

```
E! programa '[17204] Básico 5.exe' terminó con código 321225786 (0xc000013a).
```

4. SALIDA POR PANTALLA

```
C:\Users\aloan\source\repos\ X + v
Introduce el número de términos (potencia de 2): 512

> Propiedades del dispositivo seleccionado:
Nombre: NVIDIA GeForce RTX 3050 Laptop GPU
Arquitectura CUDA: AMPERE
Capacidad de cómputo: 8.6
Número de multiprocesadores: 16
Número de núcleos CUDA (16 x 64): 1024
Máximo número de hilos por bloque: 1024
Número de hilos lanzados: 512

> Calculando...

> RESULTADOS:
Valor de p: 3.141593
Aproximación de p: 3.139729
Error relativo: 0.059332%
*****
<pulsa [INTRO] para finalizar>
```