Memoria Constante

Dra. María Guadalupe Sánchez Cervantes

- Usamos la memoria constante para datos que no cambiarán en el curso de la ejecución de un kernel.
- El hardware de NVIDIA provee 64KB de memoria constante.

Declaración de la memoria constante

- El mecanismo para declarar un dato en la memoria constante es similar a declarar un buffer como memoria compartida.

```
__constant__ int s[35];
```

Ejercicio – Constante-

```
__constant__ int datos[1024];

__global__ void kernel(int *d_dst) {
   int tId = threadIdx.x + blockIdx.x * blockDim.x;
   d_dst[tId] = datos[tId];
}


int main(int argc, char **argv) {
   int *d_datos;
   cudaMalloc((void**)&d_datos, sizeof(int) * 1024);
```

```
  int *test = new int[1024];
  memset(test, 0, sizeof(int) * 1024);

  for (int i = 0; i < 1024; i++) {
    test[i] = i;
  }

  cudaMemcpyToSymbol(datos, test, sizeof(int) * 1024);
  kernel<<< 1, 1024 >>>(d_datos);

  free(test);
  cudaFree(d_datos);


  return 0;
}
```

Captura del tiempo

```
cudaEvent_t start,stop;
cudaEventCreate(&start);
cudaEventCreate(&stop);
cudaEventRecord(start,0);
.............
cudaEventRecord(stop,0);
cudaEventSynchronize(stop);
float elapsedTime;
cudaEventElapsedTime(&elapsedTime,start,stop);
printf("Time to generate: %3.1f ms\n",elapsedTime);
cudaEventDestroy(start);
cudaEventDestroy(stop);
```

Ejercicio a resolver

1. Añadir al *ejercicio de constante*, el tiempo de ejecución del kernel.
2. Realizar la copia de los datos de la memoria global al host e imprimirlo.
3. Añadir un nuevo kernel haciendo uso solo de la memoria global.
4. Comparar el tiempo entre la utilización de memoria constante y la memoria global.