
Universidad Nacional de Córdoba

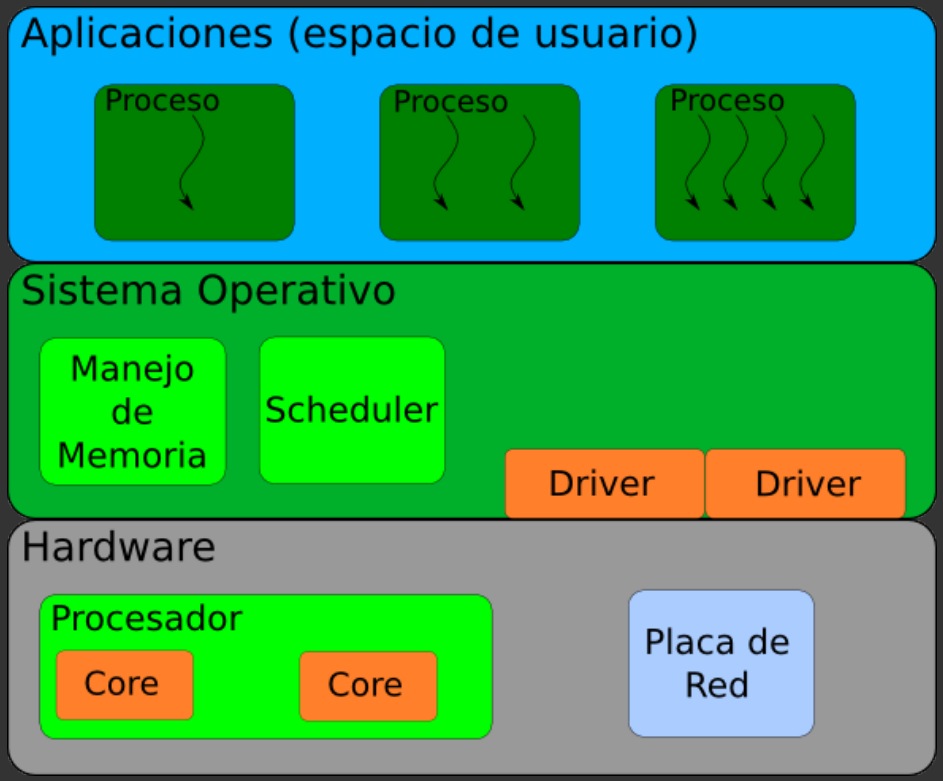
FaMAF | GPGPU Computing Group


# Honrarás tus recursos y tu hardware

Matías Bellone

# PCs

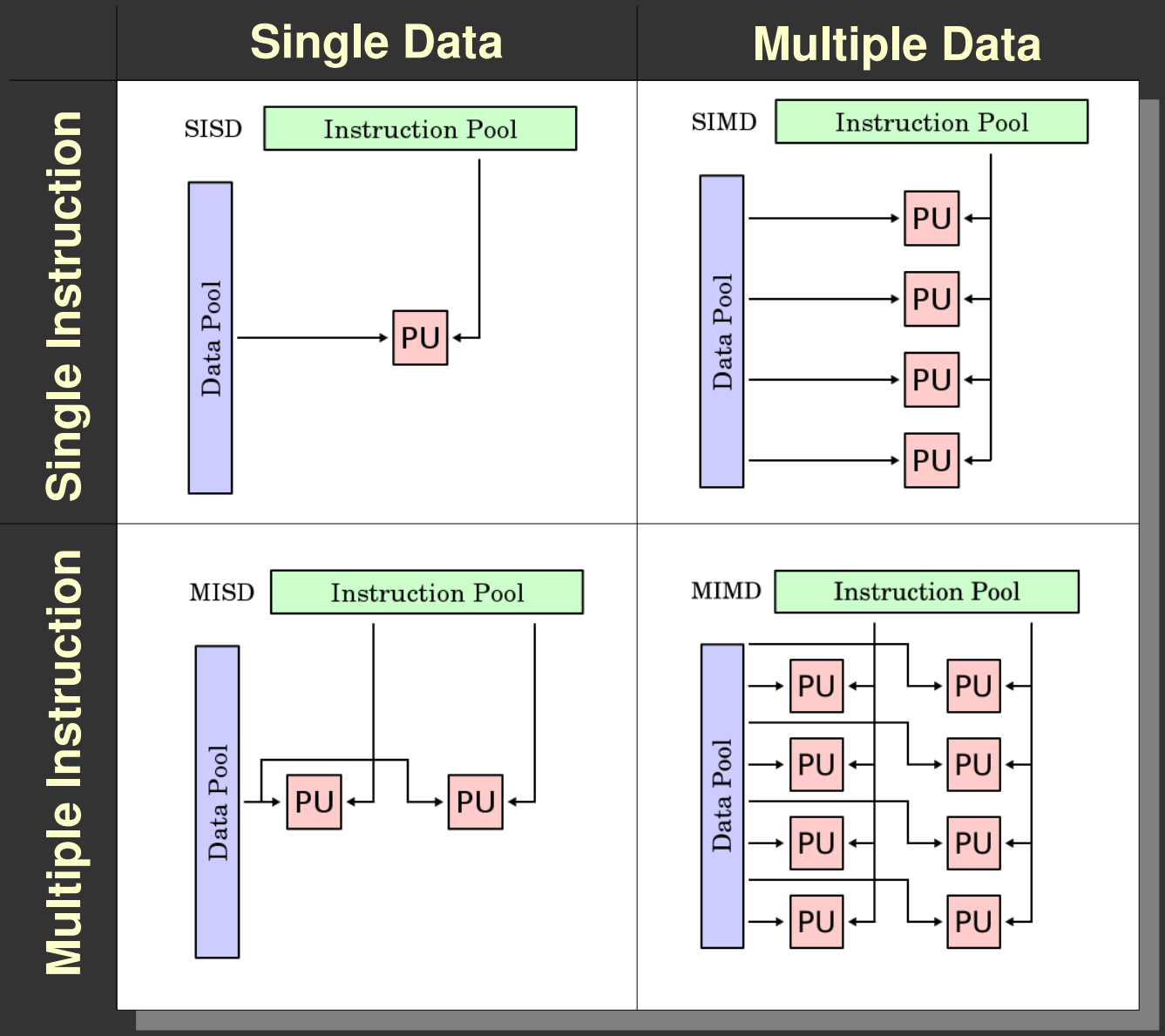
- Muy estructurada
- Muy versátil
- Estructura base



# Clusters

- Muchas PCs trabajando en conjunto
- Organización particular
  - Sistema de archivos
  - Memoria
  - Comunicación
- La arquitectura determina un modelo de paralelismo ideal (o necesario)

# Flynn's Taxonomy

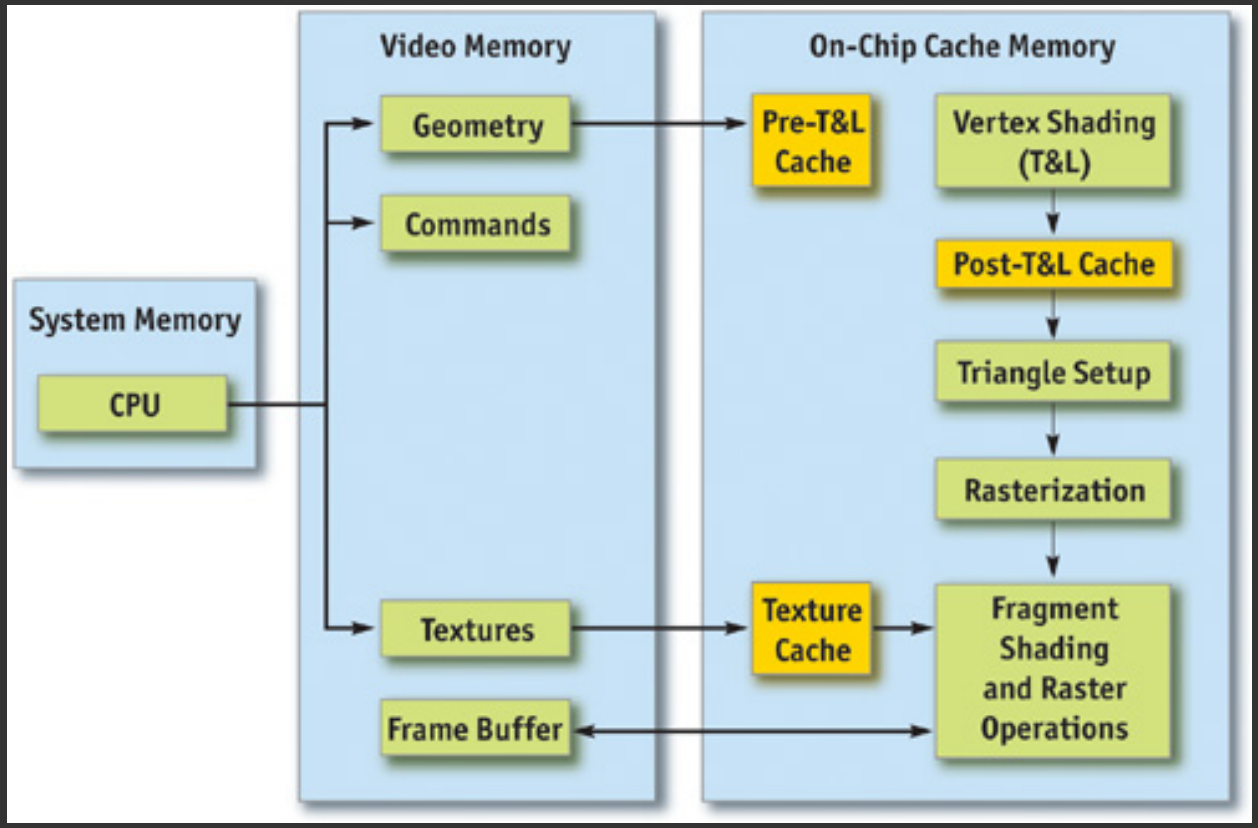


# Co-procesadores

- Ayudar al procesador con una tarea
- Más popular: co-procesador matemático
  - Para XTs, 386s y 486s
- Después vinieron otros
  - Sonido
  - Video

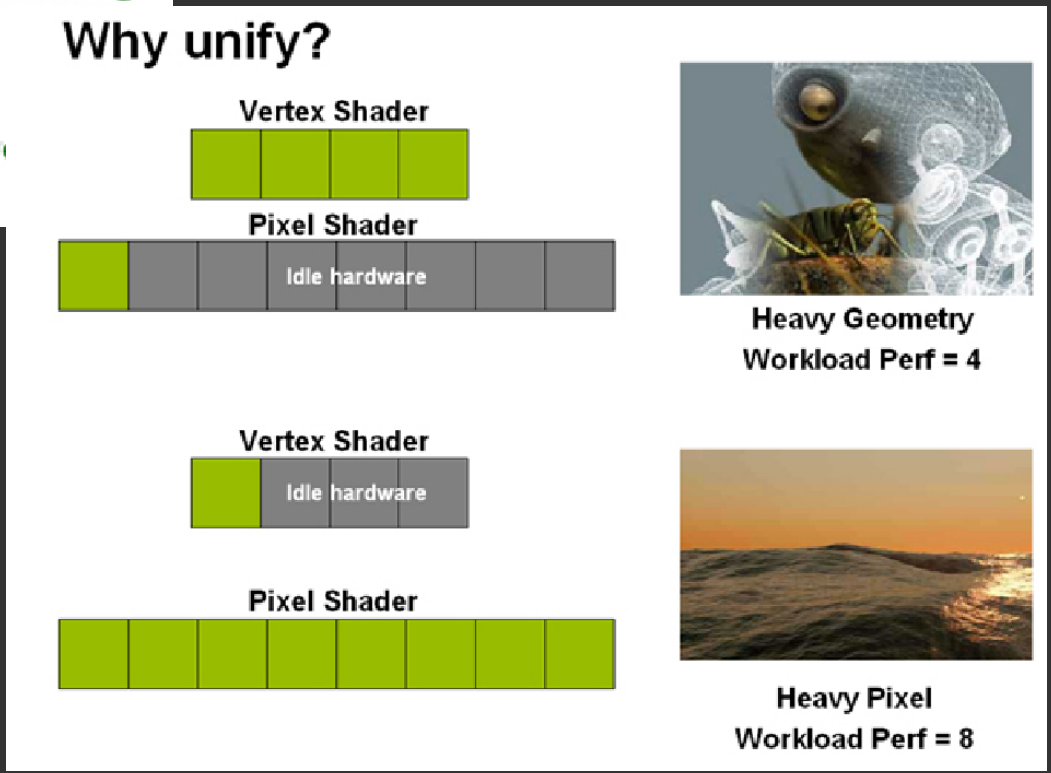
# Aceleradoras de Video

- Co-procesadores vectoriales
- APIs de Video
  - OpenGL
  - DirectX

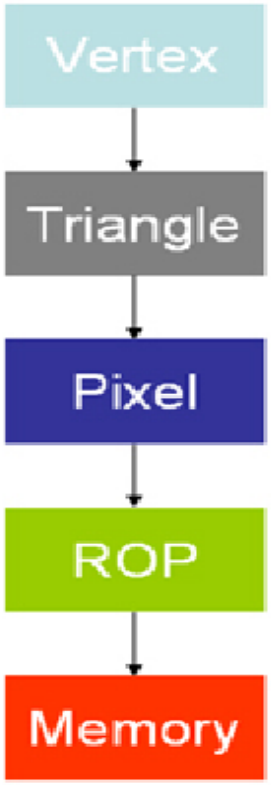


# GPUs


Vertex
T&L evolved to vertex shading
Triangle
Triangle, point, line – setup
Pixel
Flat shading, texturing, eventually
Pixel shading
ROP
Blending, Z-buffering, antialiasing
Memory
Wider and faster over the y


Why unify?
Vertex Shader
Pixel Shader
Idle hardware
Heavy Geometry
Workload Perf = 4
Vertex Shader
Idle hardware
Pixel Shader
Heavy Pixel
Workload Perf = 8

# GPUs



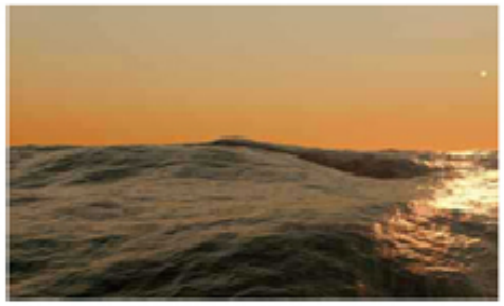

T&L evolved to vertex shading

## Why unify?

**Unified Shader**

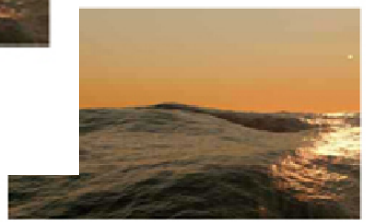

**Heavy Geometry**
**Workload Perf =12**

**Unified Shader**

**Heavy Pixel**
**Workload Perf = 12**

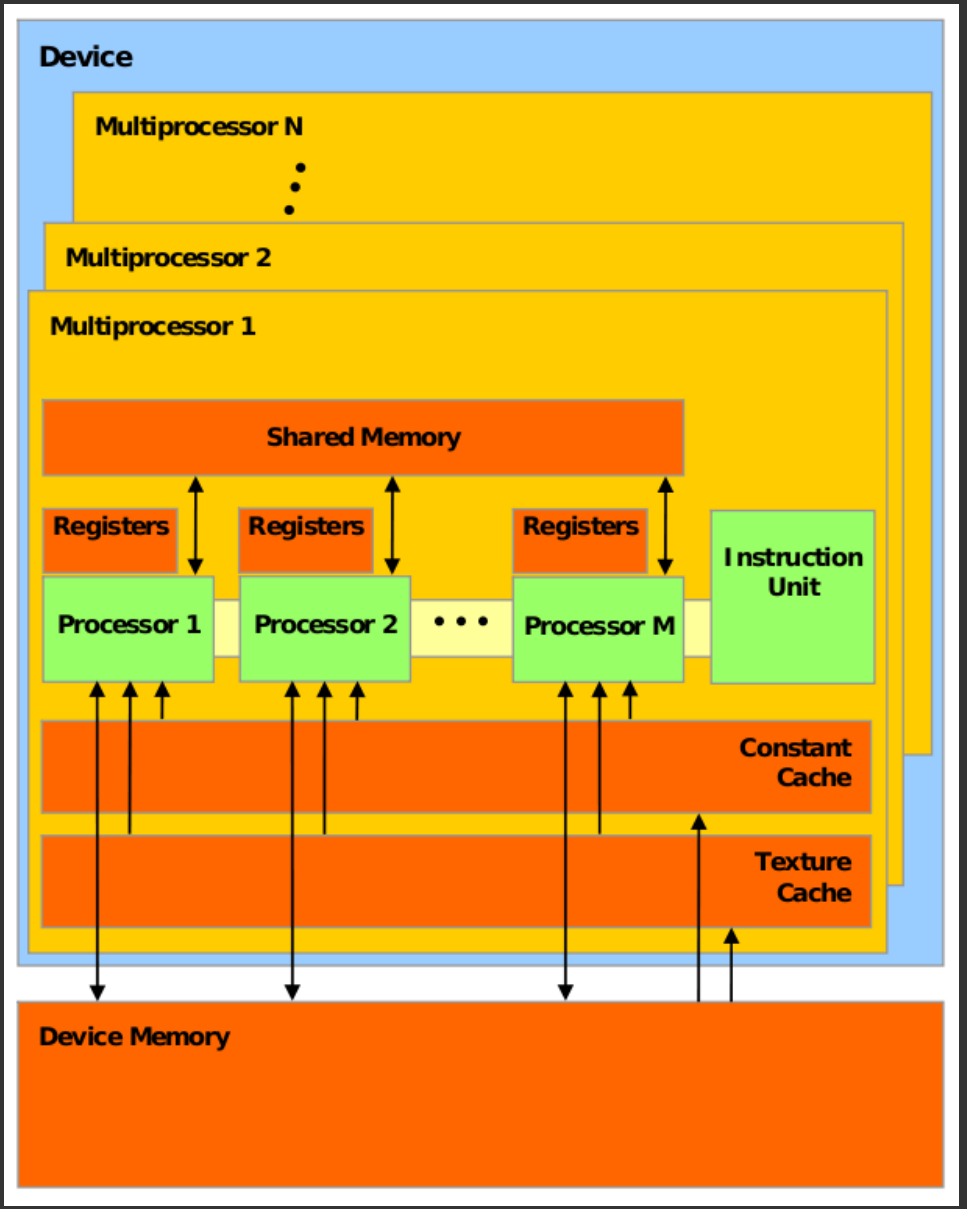
**Heavy Geometry**
**Workload Perf = 4**



**Heavy Pixel**
**Workload Perf = 8**

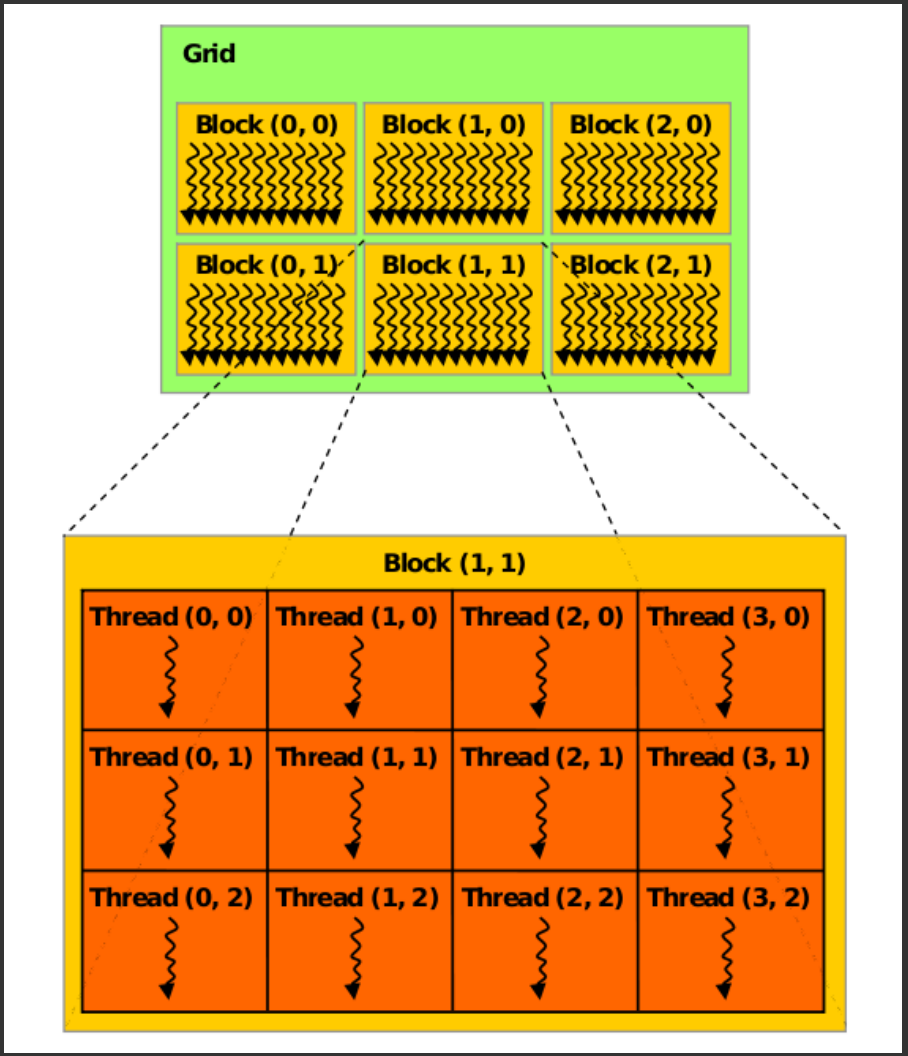
# SIMT

- Multi-Procesadores
  - Muchas FPUs
  - Una sola unidad de instrucción
  - Pocas DPUs
- Memoria
  - Memoria general
  - Memoria específica
- Scheduler de 0 overhead (en hardware)



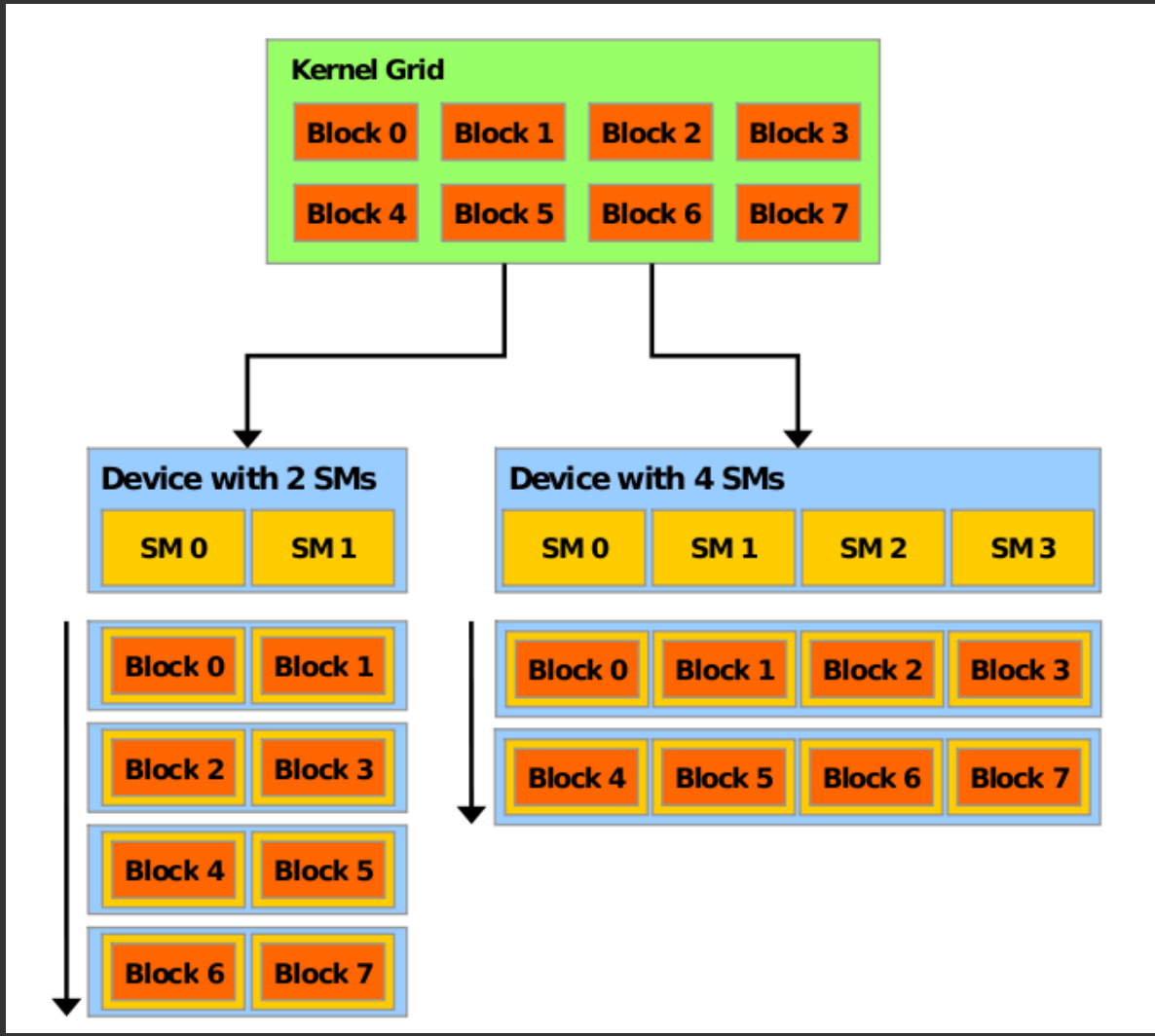
# Grillas de Threads

- Organización de threads en bloques
- Organización de bloques en una grilla
- Facilita el mapeo de threads a datos
- Límites máximos impuestos por la arquitectura



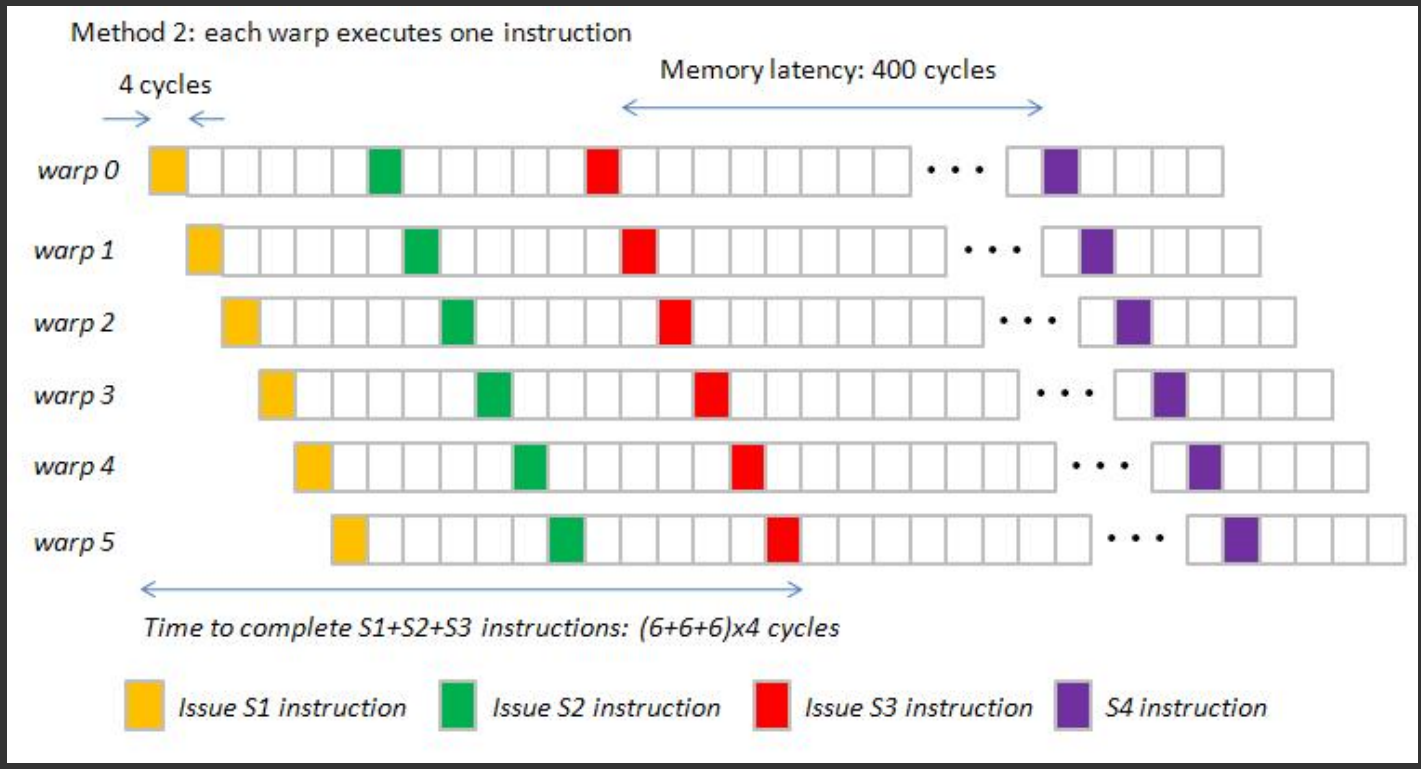
# Bloques y MPs

- Bloque: unidad lógica de trabajo
- Se van a asignar tantos bloques como se pueda a un MP
- Thread: unidad mínima de ejecución
- Warp: unidad mínima de sincronización



# Occupancy

- El acceso a memoria es caro
- Los threads son baratos
- Esconder latencia y aprovechar poder de procesamiento



# Limitaciones

- Primitivas de sincronización
- Manejo de memoria
- No soporta todo C
  - nada de struct y union
  - nada de punteros a funciones (inlining)
  - sólo arrays uni-dimensionales
- Propietario (Hardware y Software)

# Soluciones

- GPU como dumb-worker
- Es una tecnología en evolución
  - Mejora del lenguaje
  - Agregado de funcionalidad
  - Mejora en el hardware
- Existen soluciones equiparables
  - OpenCL
  - Ocelot