Arquitecturas Paralelas (Parte 2)

Computación de alto rendimiento

Objetivos

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Amdahl Gustafson
Ejercicios

Presentación de prácticas

- Diferenciación entre multiprocesadores y multicomputadores.
- El papel de GPUs y FPGAs como aceleradores en la Computación de Alto Rendimiento (CAR).
- Conceptos de benchmarking para optimizar el rendimiento.

Introducción a los Multiprocesadores

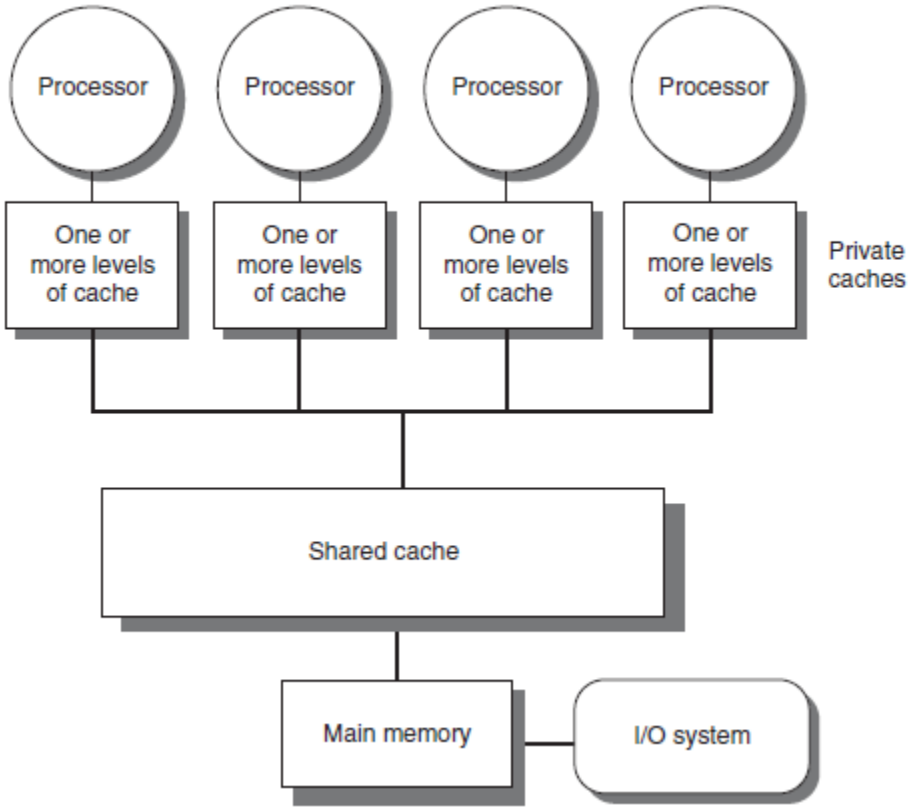
Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Amdahl Gustafson
Ejercicios
Presentación de prácticas

- **Multiprocesadores (SMP y NUMA)**
  - **Multiprocesadores:** Arquitecturas donde varios procesadores comparten una única memoria común.
  - **Modelos principales:**
    - **SMP** (Symmetric Multiprocessing): Todos los procesadores tienen acceso uniforme a la memoria.
    - **NUMA** (Non-Uniform Memory Access): El tiempo de acceso a la memoria varía según su proximidad al procesador.
  - **Ventajas**:
    - Baja latencia en el acceso a la memoria.
    - Programación más sencilla en comparación con sistemas distribuidos.

Introducción a los Multiprocesadores

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Amdahl Gustafson
Ejercicios
Presentación de prácticas

- **SMP (Symmetric Multiprocessing):** Todos los procesadores tienen acceso uniforme a la memoria.



Introducción a los Multiprocesadores

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Amdahl Gustafson
Ejercicios
Presentación de prácticas

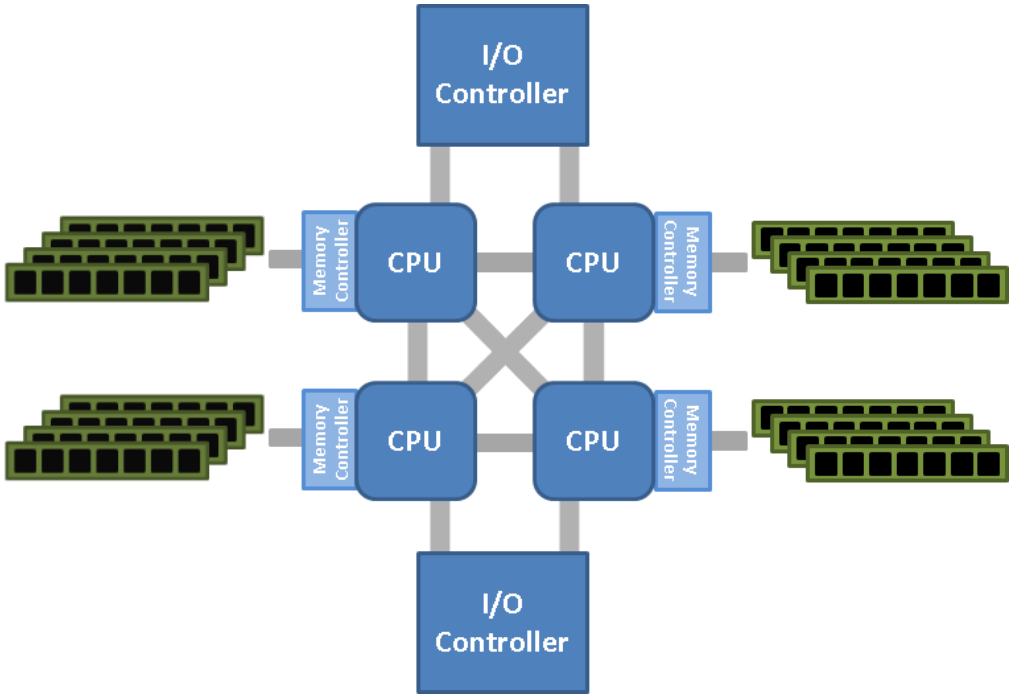
- **Arquitectura de SMP (Multiprocesamiento Simétrico)**
  - **Definición**: Los sistemas SMP permiten que varios procesadores accedan de forma equitativa a una misma memoria compartida.
    - El tiempo de acceso a la memoria es uniforme para todos los procesadores.
    - Los procesadores suelen comunicarse a través de la memoria sin necesidad de mensajes explícitos.
  - **Características:**
    - **Baja latencia:** Los procesadores acceden a la memoria sin esperas significativas.
    - **Escalabilidad limitada:** Al aumentar el número de procesadores, la contención por el acceso a la memoria se convierte en un problema.
  - **Ventaja clave:** Fácil de programar con herramientas como OpenMP, ya que no requiere gestionar mensajes entre procesadores.

Introducción a los Multiprocesadores

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Amdahl Gustafson
Ejercicios

Presentación de prácticas

- **NUMA (Non-Uniform Memory Access):** El tiempo de acceso a la memoria varía según su proximidad al procesador.



Introducción a los Multiprocesadores

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Amdahl Gustafson
Ejercicios
Presentación de prácticas

- **NUMA (Non-Uniform Memory Access): El tiempo de acceso a la memoria varía según su proximidad al procesador.**
  - **Definición**: En NUMA, cada procesador tiene su propia memoria local, a la que accede rápidamente.
    - El acceso a la memoria de otros procesadores es posible, pero con mayor latencia.
    - Ideal para sistemas con un gran número de procesadores, ya que reduce la contención de memoria.
  - Ventajas de NUMA**:**
    - Mejora la escalabilidad en comparación con SMP.
    - Reduce los cuellos de botella al distribuir la carga de memoria.
  - Problema potencial:
    - La programación es más compleja, ya que se deben optimizar los accesos a memoria remota para evitar penalizaciones de latencia.

Introducción a los Multiprocesadores

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Amdahl Gustafson
Ejercicios
Presentación de prácticas

- **Comparación SMP vs. NUMA**

CARACTERÍSTICA	SMP (Symmetric Multiprocessing)	NUMA (Non-Uniform Memory Access)
Acceso a memoria	Uniforme: el tiempo de acceso es el mismo para todos los procesadores.	No uniforme: el tiempo de acceso depende de la proximidad del procesador.
Escalabilidad	Limitada por la contención de la memoria compartida.	Alta, debido a la reducción de la contención.
Coherencia de memoria	Necesaria mediante protocolos como MESI.	Necesaria, pero puede optimizarse según la topología.
Aplicaciones típicas	Bases de datos transaccionales, servidores de aplicaciones.	Simulaciones científicas, procesamiento masivo de datos.

Introducción a los Multiprocesadores

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Amdahl Gustafson
Ejercicios

Presentación de prácticas

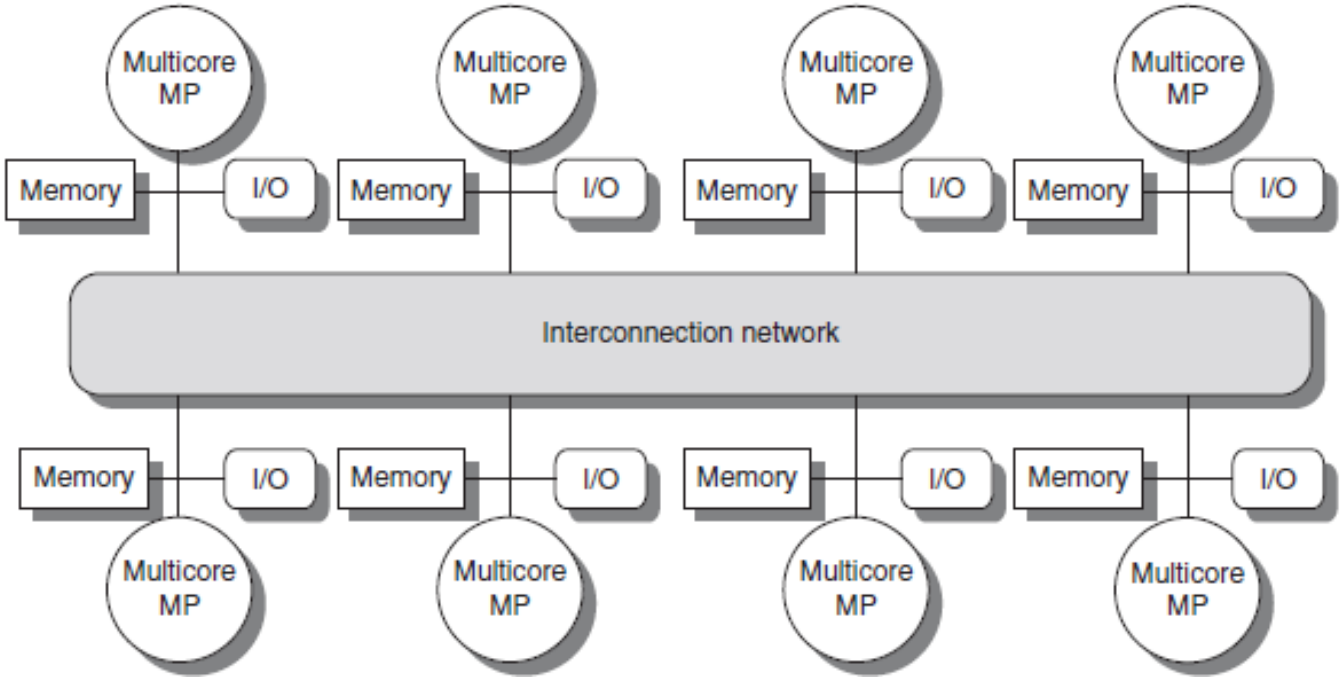
- **Problemas y limitaciones en Multiprocesadores**
- **Problemas de escalabilidad:**
  La competencia por la **memoria** compartida puede degradar el rendimiento cuando se añaden más procesadores.
- **Condiciones de carrera:**
  Ocurren cuando dos o más procesadores acceden simultáneamente a los mismos datos sin la sincronización adecuada.
- **Técnicas de optimización:**
- Uso de secciones críticas para evitar conflictos
- Balanceo de carga dinámico para optimizar la distribución de tareas.

NOTA. Éstos problemas iremos abordándolos en sesiones posteriores

Introducción a los Multicomputadores

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Amdahl Gustafson
Ejercicios
Presentación de prácticas

- **Definición de Multicomputadores**



Introducción a los Multicomputadores

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Amdahl Gustafson
Ejercicios
Presentación de prácticas

- **Definición de Multicomputadores**
  - Un **multicomputador** está compuesto por varios nodos independientes, cada uno con su propio procesador y memoria local.
  - A diferencia de los multiprocesadores (memoria compartida), los multicomputadores se basan en **memoria distribuida**.
  - Los nodos se comunican mediante el paso de mensajes con protocolos como **MPI** (Message Passing Interface))
- **Ventajas clave:**
  - Alta escalabilidad, ya que es fácil añadir más nodos si afectar al rendimiento global.
  - Tolerancia a fallos: Si un nodo falla, el sistema global puede continuar funcionando.
- **Desventaja:**
  - Mayor latencia debido a la comunicación entre nodos.
  - Mayor complejidad por el manejo de paso de mensajes

Multiprocesadores Vs Multicomputadores

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Amdahl Gustafson
Ejercicios
Presentación de prácticas

- **Comparación entre Multiprocesadores y Multicomputadores**

Característica	Multiprocesadores	Multicomputadores
Memoria	Compartida entre todos los procesadores	Distribuida, cada nodo tiene la suya
Comunicación	Mediante acceso directo a la memoria	Paso de mensajes explícito
Escalabilidad	Limitada por la competencia por la memoria	Alta: se pueden añadir más nodos
Latencia	Baja	Alta (depende de la red de interconexión)
Programación	Más sencilla	Más compleja, requiere gestión explícita de mensajes

Balanceo de Carga en Multiprocesadores y Multicomputadores

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Amdahl Gustafson
Ejercicios

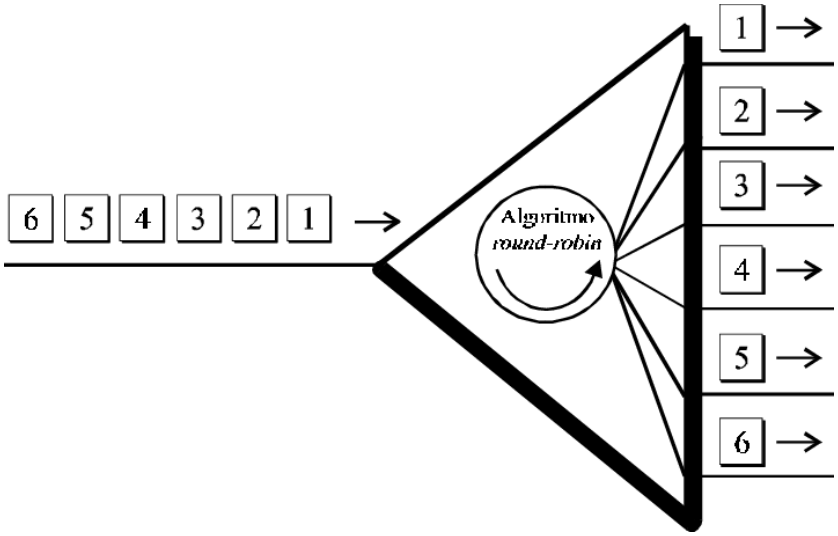
Presentación de prácticas

- **En multiprocesadores:**
  Las tareas deben distribuirse equitativamente entre los procesadores para evitar cuellos de botella en el acceso a la memoria.
- **En multicomputadores:**
  Se debe asegurar que **tanto las tareas como los datos** estén distribuidos de forma uniforme para evitar que un nodo esté sobrecargado mientras otros están inactivos.
- **Técnicas de balanceo de carga:**
  - **Estático:** Las tareas se asignan antes de la ejecución.
  - **Dinámico:** Las tareas se asignan o reasignan en tiempo de ejecución.

Balanceo de Carga en Multiprocesadores y Multicomputadores

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Amdahl Gustafson
Ejercicios
Presentación de prácticas

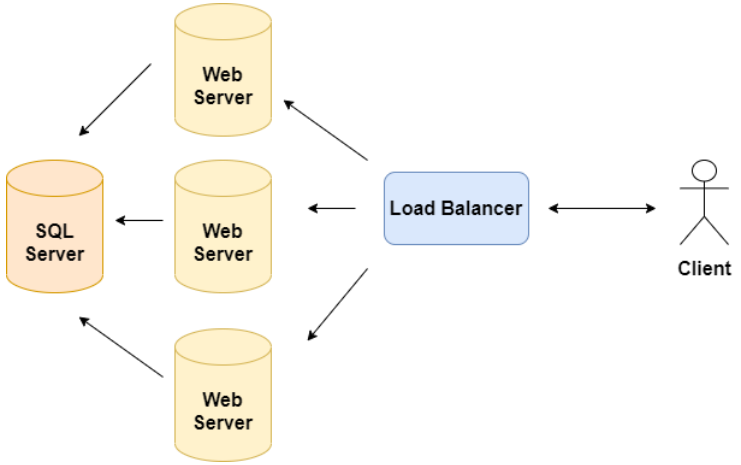
- **Estático:** Las tareas se asignan antes de la ejecución.
- Distribución del tráfico se realiza según reglas predefinidas que no cambian en tiempo real.
- Este método no considera el estado actual de los servidores o recursos.
- Ejemplo: algoritmo **"round robin",** donde las solicitudes se asignan secuencialmente a cada servidor en un orden cíclico.
- Este enfoque es sencillo y funciona bien cuando los servidores tienen capacidades similares y la carga es predecible.



Balanceo de Carga en Multiprocesadores y Multicomputadores

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Amdahl Gustafson
Ejercicios
Presentación de prácticas

- **Dinámico:** Las tareas se asignan o reasignan en tiempo de ejecución.
- Ajusta la distribución del tráfico en función del estado actual de los servidores como la **carga de CPU**, el uso de **memoria** o el número de **conexiones activas**.
- Por ejemplo, el algoritmo de **"Least Connections"** dirige las nuevas solicitudes al servidor con la menor cantidad de conexiones activas en ese momento.
- Este enfoque es más adaptable en entornos con cargas variables.



Visita recomendada: https://aws.amazon.com/es/what-is/load-balancing

GPUs y su Arquitectura Paralela

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Amdahl Gustafson
Ejercicios
Presentación de prácticas

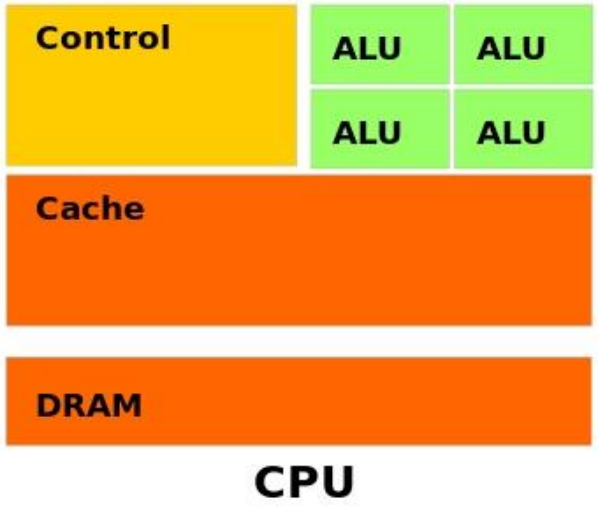
- **Características clave de las GPUs:**
  - Miles de núcleos trabajando en paralelo.
  - Optimizadas para operaciones vectoriales y de punto flotante.
  - Lenguajes de programación: CUDA (NVIDIA) y OpenCL.
- Casos de uso:
  - Redes neuronales profundas.
  - Renderizado de gráficos.
  - Simulaciones físicas.

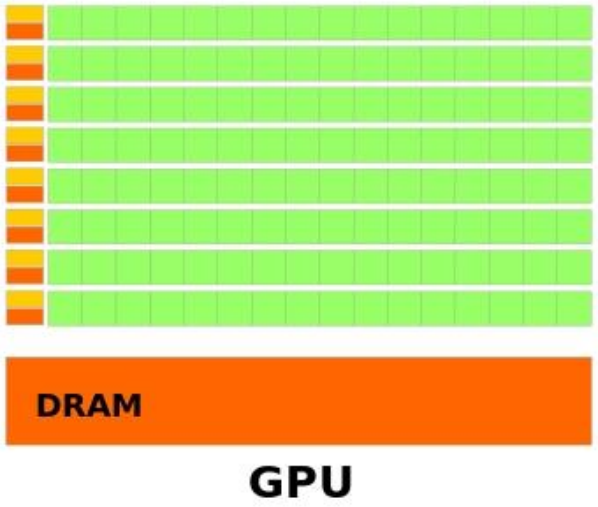
Visita recomendada: https://aws.amazon.com/es/what-is/load-balancing/?utm_source=chatgpt.com

GPUs y su Arquitectura Paralela

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Amdahl Gustafson
Ejercicios
Presentación de prácticas

- **Características clave de las GPUs:**

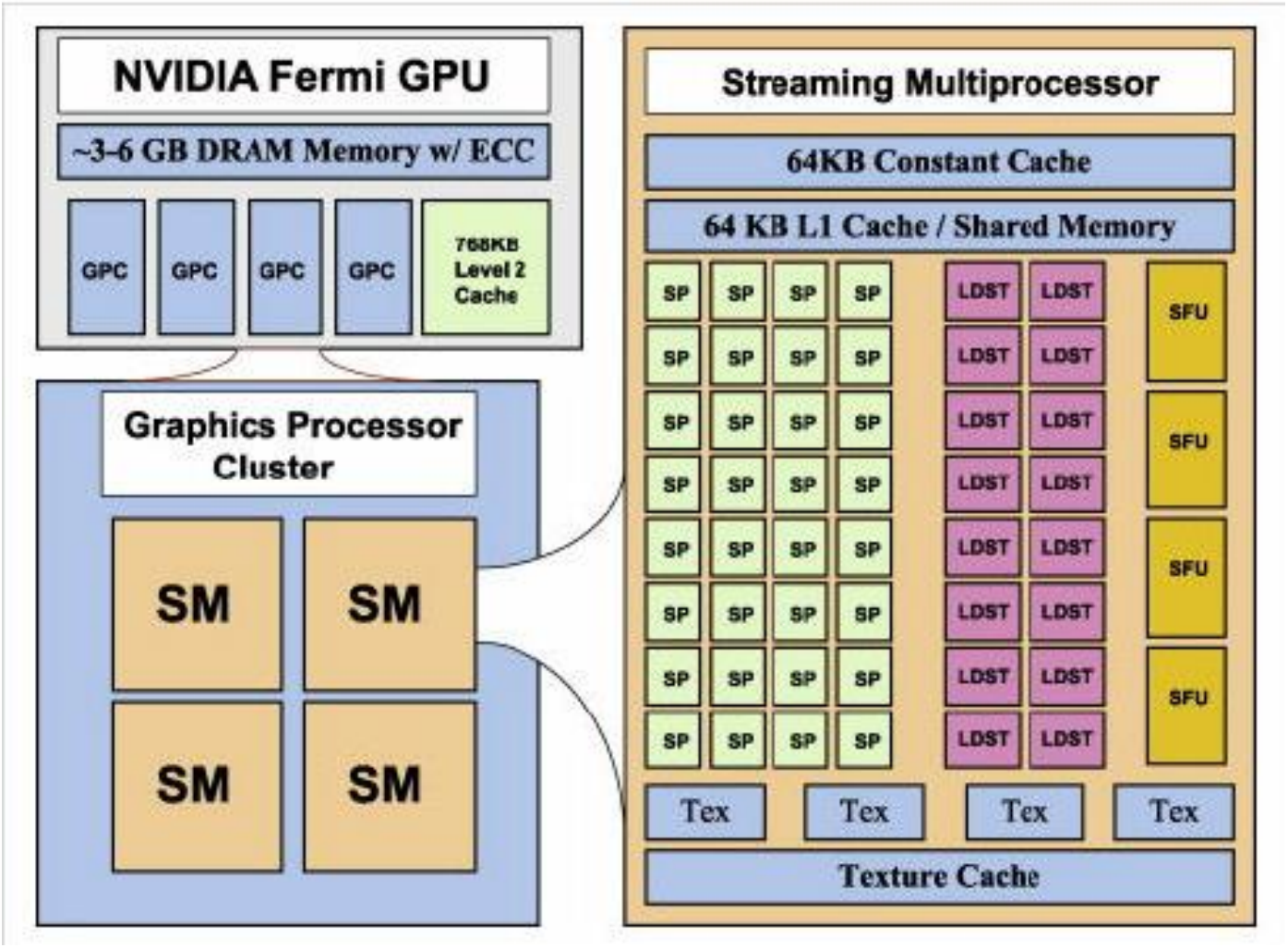




GPUs y su Arquitectura Paralela

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Amdahl Gustafson
Ejercicios
Presentación de prácticas

- **Características clave de las GPUs:**



GPUs y su Arquitectura Paralela

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Amdahl Gustafson
Ejercicios
Presentación de prácticas

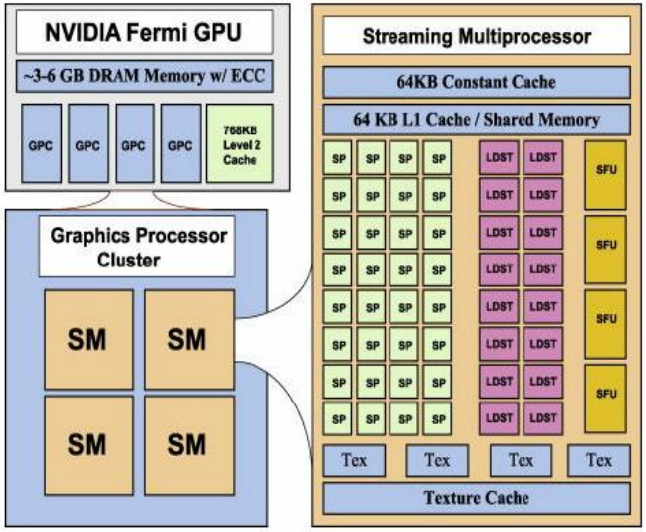
1. NVIDIA Fermi GPU (izquierda):

DRAM Memory (~3-6 GB): Memoria global donde se almacenan los datos principales.

768 KB **Level 2 Cache**: Caché que reduce el tiempo de acceso a la memoria global.

GPC (Graphics Processor Cluster): Agrupa varios SMs y gestiona cálculos.

SM (Streaming Multiprocessor): Unidad que contiene núcleos CUDA para la computación en paralelo.



2. Streaming Multiprocessor (derecha):

64 KB **Constant Cache**: Almacena valores constantes para accesos rápidos.

64 KB **L1 Cache** / Shared Memory: Memoria de baja latencia compartida entre hilos del bloque.

SP (Streaming Processor): **Núcleo CUDA** que ejecuta instrucciones en paralelo.

LD/ST Units: Gestiona la carga y almacenamiento de datos entre la memoria y los registros.

SFU (Special Function Units): Realiza operaciones matemáticas complejas (trigonométricas, raíces, etc.).

Tex (Texture Cache): Caché para acceso rápido a texturas y datos gráficos.

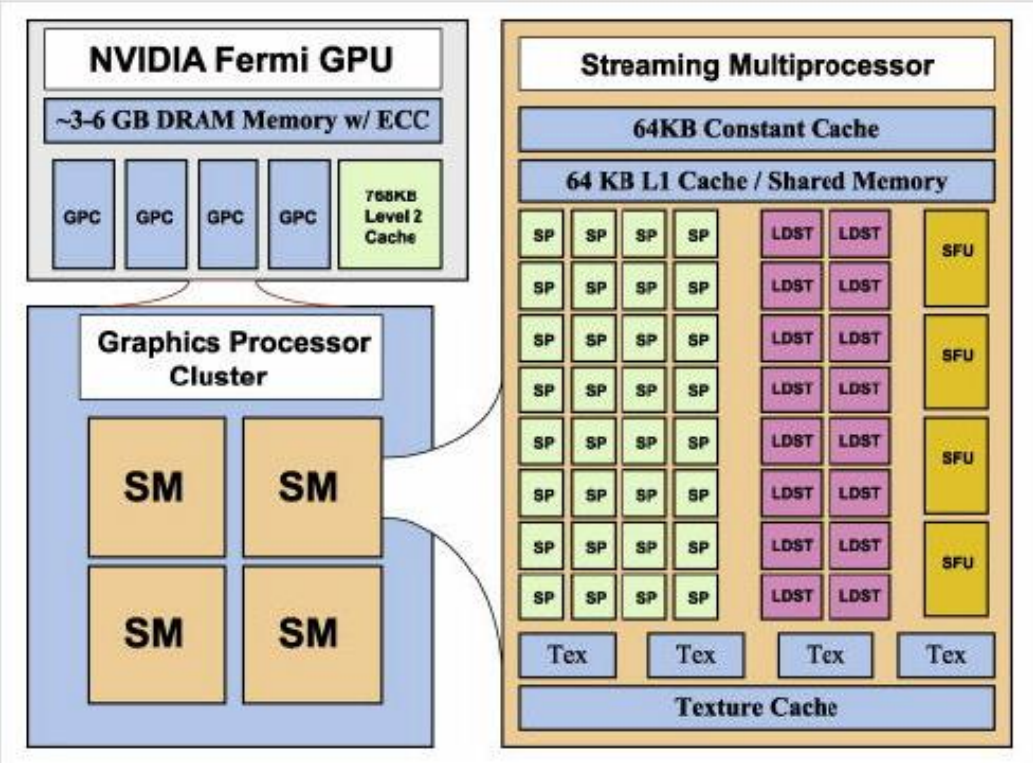
Ejemplo: Un modelo como **GeForce GTX 480** tiene **15 SMs** con **480 núcleos CUDA** en total.

GPUs y su Arquitectura Paralela

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Ejercicios
Presentación de prácticas

- **Ejemplo práctico:**

  Para procesar una imagen de la cara de una persona y aplicar **reconocimiento facial** utilizando una GPU como la que describe el esquema de **NVIDIA Fermi**, cada parte de la GPU tendría un papel específico

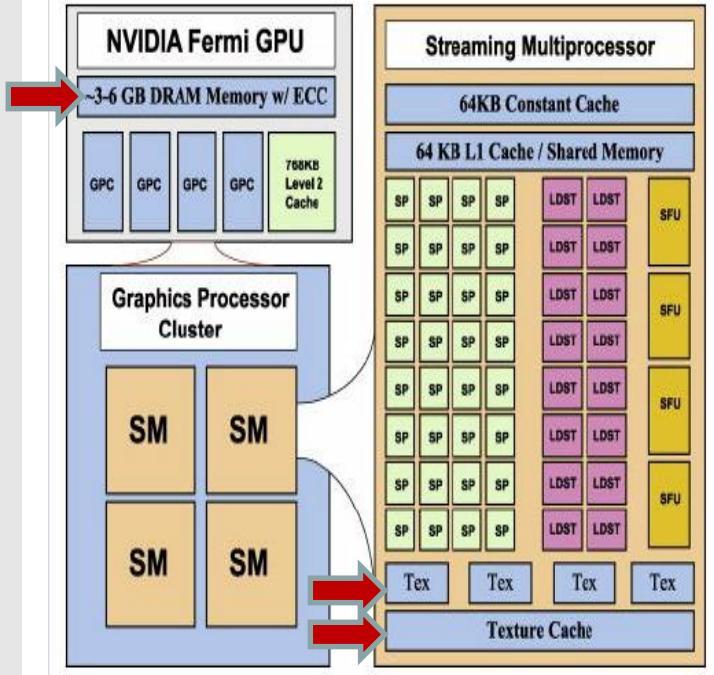


Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Ejercicios
Presentación de prácticas

1. Entrada de la imagen y su preparación

- La imagen de la cara llega en formato digital (1024x1024)
- Antes de ser procesada, se convierte en una matriz de píxeles (p-ej 1024x1024).
- Cada píxel contiene información de **color** o **luminosidad**.

- **DRAM Memory** (~3-6 GB): Se almacena la imagen original y los datos intermedios (como características detectadas) en esta memoria principal.
- **Texture Cache (Tex)**: Si trabajamos con características basadas en texturas, como bordes o patrones locales, esta cache puede optimizar el acceso rápido a pequeños bloques de la imagen.

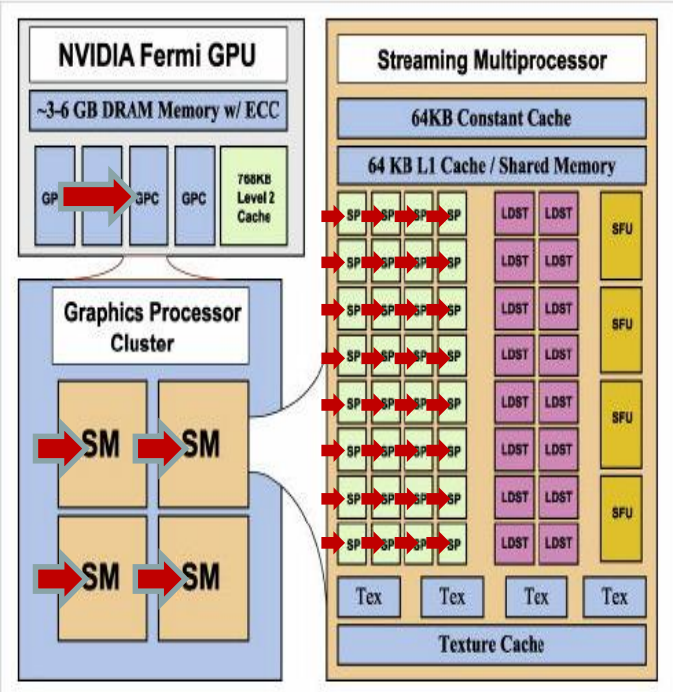


Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Ejercicios
Presentación de prácticas

2. Descomposición de la imagen en tareas paralelas

Para el reconocimiento facial, dividimos la imagen en **pequeños bloques** (fragmentos de **16x16 píxeles**) y asignamos cada bloque a un grupo de **hilos paralelos**.

- **GPC** (Graphics Processor Cluster): Coordina el trabajo de varios **SMs** (Streaming Multiprocessors), asignando **fragmentos** de la imagen a **diferentes partes de la GPU**.
- **SMs**: Cada SM se encarga de procesar **un bloque** de la imagen.
- **SPs** (Streaming Processors): Dentro de cada SM, múltiples núcleos SP aplican un **filtro de detección** de bordes (como Sobel) para encontrar las líneas de la nariz, ojos, y boca

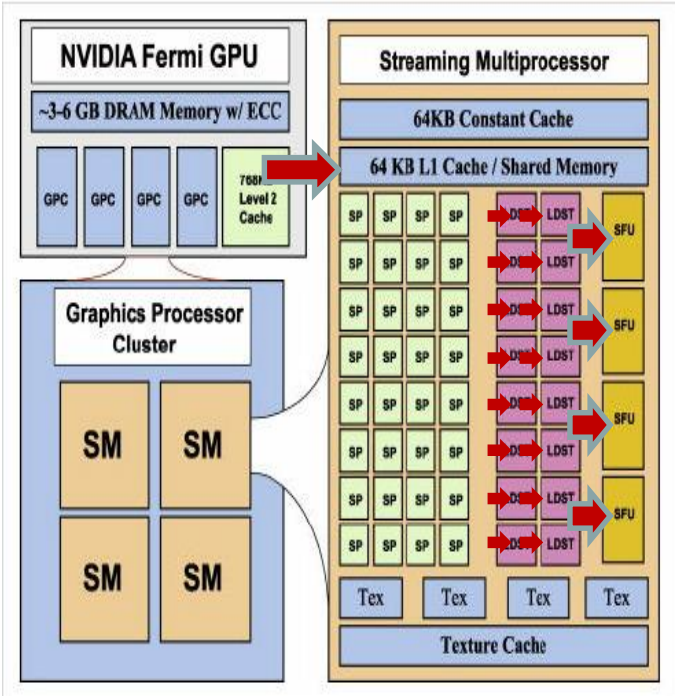


Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Ejercicios
Presentación de prácticas

3. Extracción de características

Durante esta fase, se detectan características importantes en la imagen, como la forma de los ojos, la distancia entre ellos, o el contorno de la cara.

- **SFU** (Special Function Units): Estas unidades ejecutan funciones matemáticas complejas como raíces cuadradas, logaritmos o funciones trigonométricas, que son necesarias en cálculos como la **detección de bordes**, el **escalado** o la **transformación de la imagen**.
- **LD/ST Units:** Los datos intermedios se mueven constantemente entre los registros de los **SPs** y la **L1 Cache**
- **L1 Cache / Shared Memory**: Los **datos temporales y las características intermedias** (por ejemplo, bordes detectados o fragmentos filtrados) **se almacenan** aquí **para reducir la latencia.**

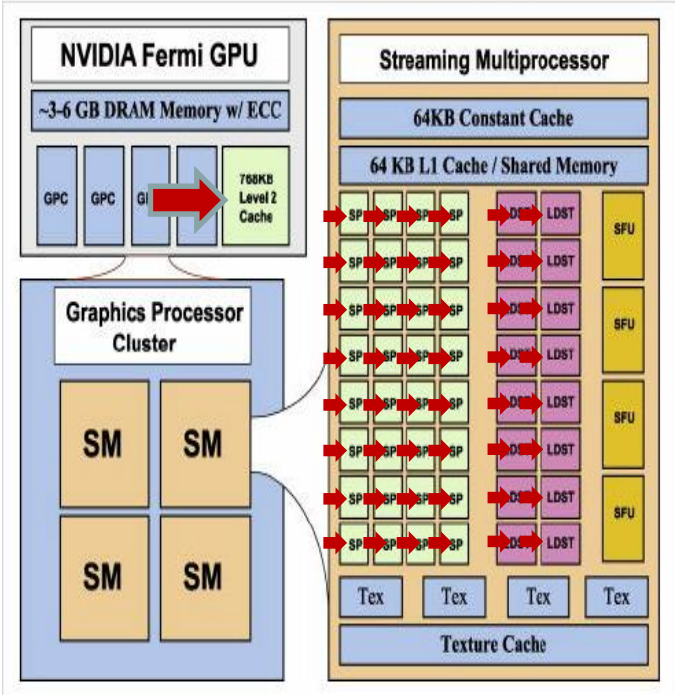


Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Ejercicios
Presentación de prácticas

4. Comparación con la base de datos

Una vez detectadas las características, se comparan con una **base de datos preentrenada** que contiene **rostros conocidos**.

- **L2 Cache (768 KB):** Almacena **parámetros del modelo** y datos intermedios.
- **LD Units:** Cargan los parámetros necesarios del modelo de reconocimiento desde la **L2** Cache o la **DRAM** Memory **hacia** los **SPs**.
- **SPs**: Ejecutan **en paralelo** las operaciones de **comparación**.
- Si el modelo utiliza una **red neuronal convolucional** (**CNN**), los **LD** Units cargan los **pesos** de la red desde la memoria, y los **SPs** aplican el modelo a las características detectadas.

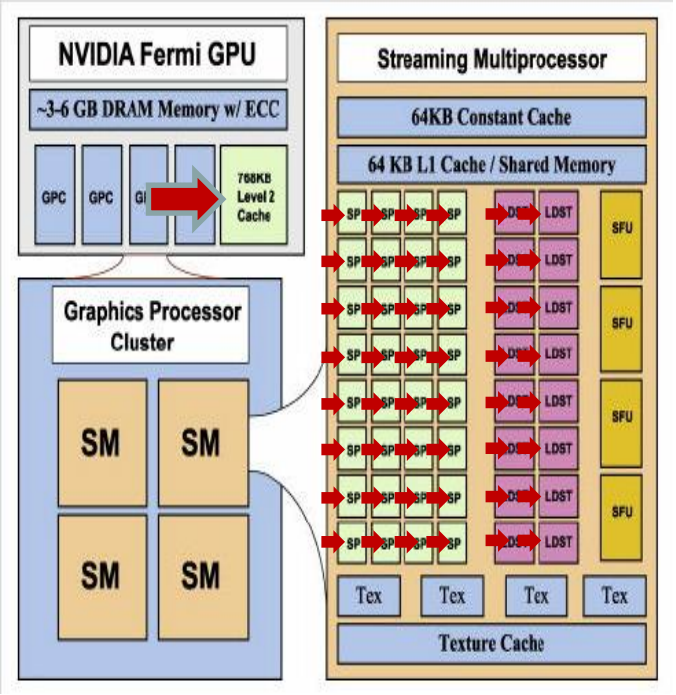


Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Ejercicios
Presentación de prácticas

5. Salida del resultado

El resultado final (por ejemplo, "**Juan Pérez identificado**") se almacena en la memoria para su uso por el sistema externo.

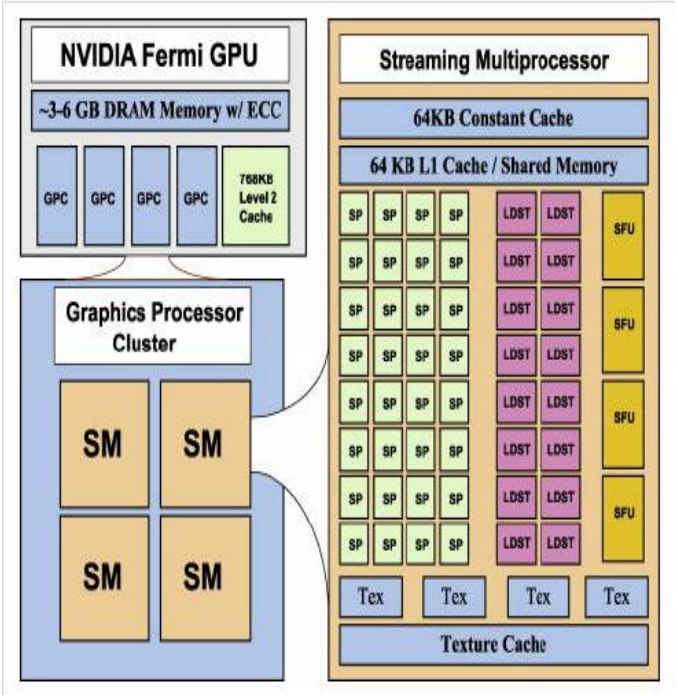
- **ST Units**: Almacenan el **resultado final** en la **DRAM Memory**.
- **DRAM Memory** (~3-6 GB): **Contiene el resultado** del procesamiento listo para ser enviado al sistema de control.



Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Ejercicios
Presentación de prácticas

Resumen del papel de los LD/ST Units en cada paso:

1. **Cargar (LD):** Los bloques de imagen son cargados desde la DRAM Memory hacia la L1 Cache y los registros de los SPs.
2. **Mover datos intermedios (LD/ST):** Durante el procesamiento, los bordes detectados y otras características se mueven entre los registros de los SPs y la shared memory.
3. **Cargar parámetros del modelo (LD):** Los pesos del modelo de reconocimiento se cargan desde la L2 Cache hacia los SPs.
4. **Almacenar el resultado final (ST):** El resultado del reconocimiento se almacena en la DRAM Memory para su uso externo.

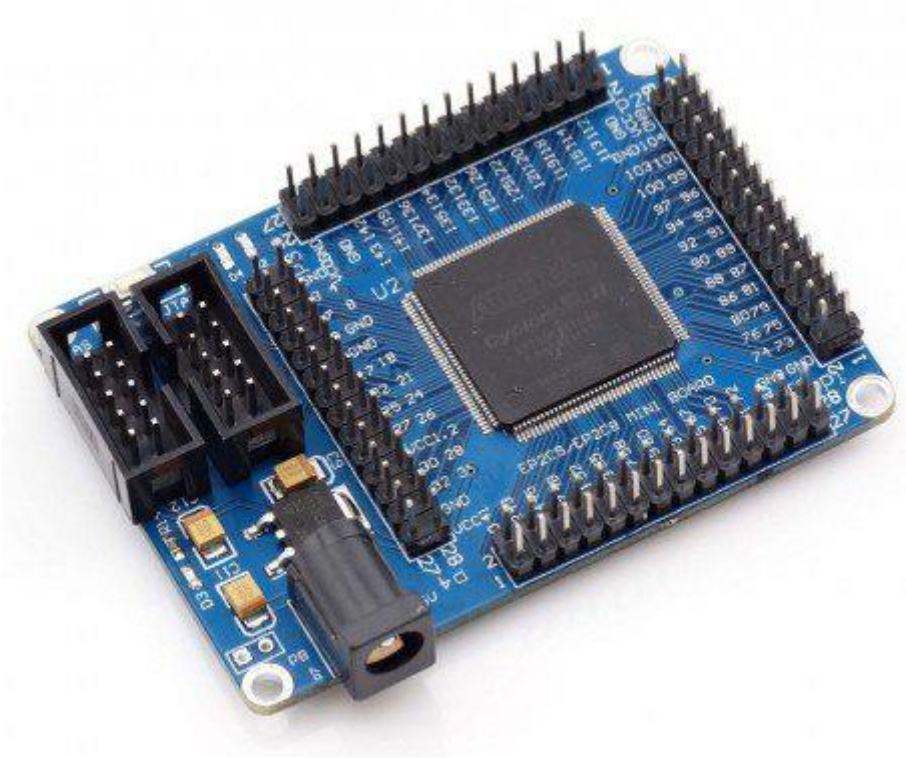


FPGAs - Personalización del Hardware

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Ejercicios

Presentación de prácticas

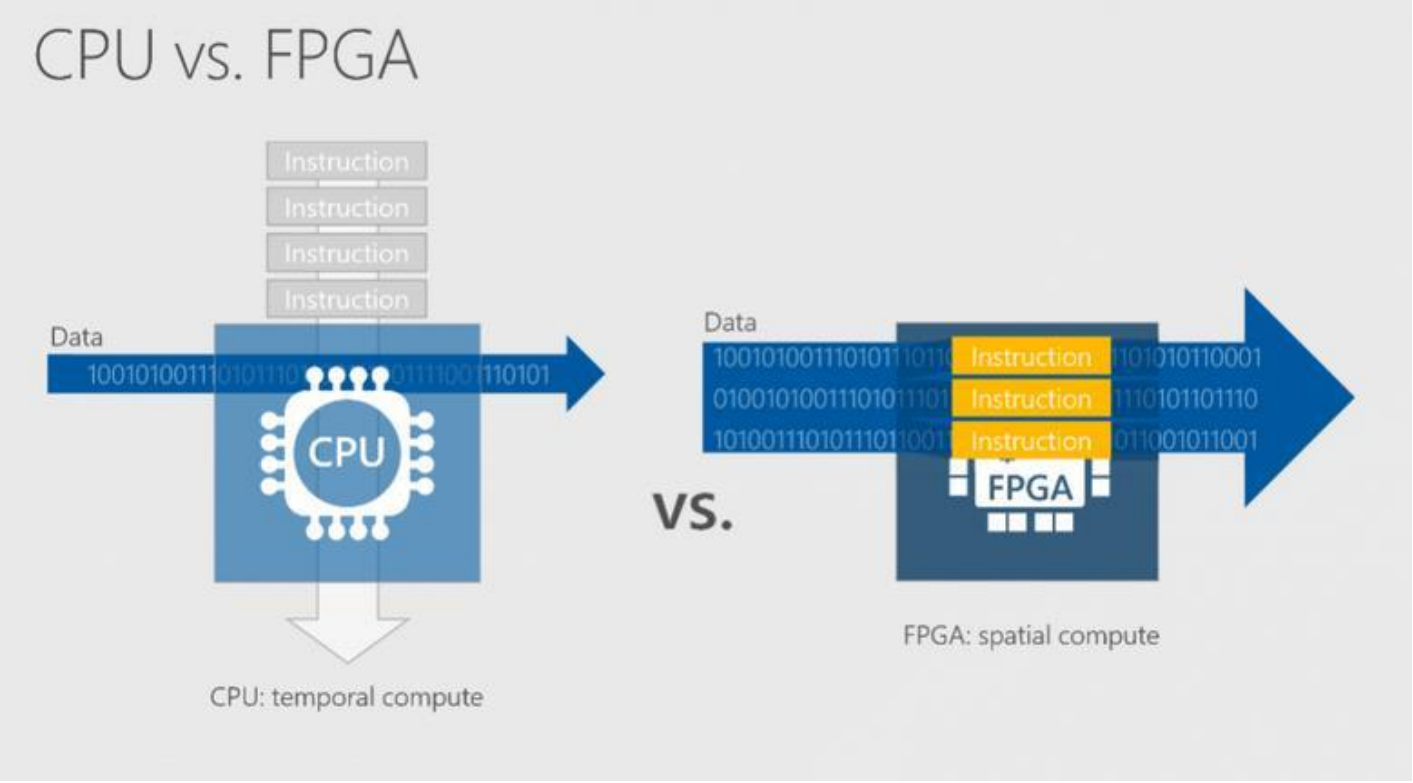
- **¿Qué es una FPGA?**



FPGAs - Personalización del Hardware

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Ejercicios
Presentación de prácticas

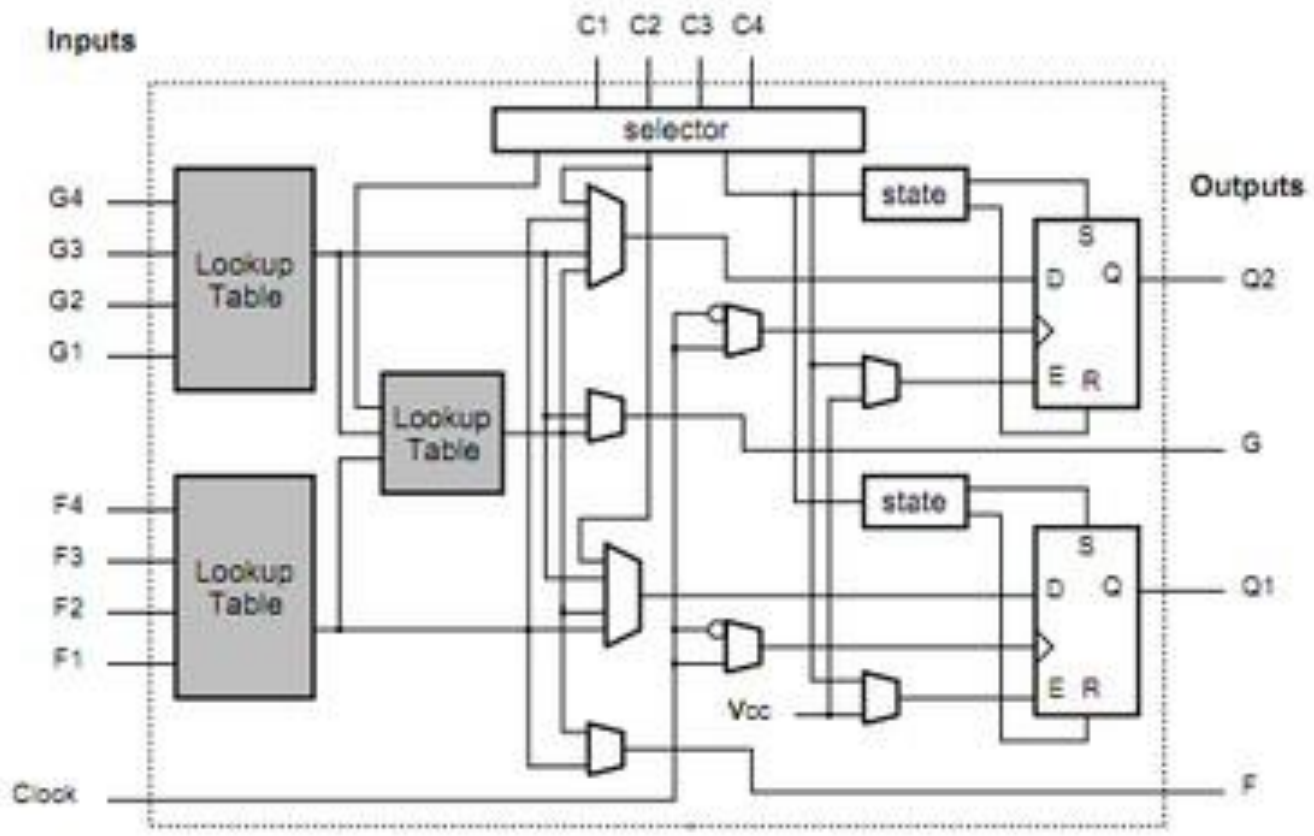
- Comparación General de FPGAs y CPUs



FPGAs - Personalización del Hardware

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Ejercicios
Presentación de prácticas

- Esquema interno FPGA (ejemplo)



FPGAs - Personalización del Hardware

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Ejercicios
Presentación de prácticas

- **Definición y Características**
  - **FPGA** (Field Programmable Gate Array): **Circuitos integrados reconfigurables** que permiten personalizar su funcionamiento después de la fabricación.
  - Se programan mediante **lenguajes de descripción de hardware** (HDL).
  - Los **bloques lógicos configurables**, como las Lookup Tables (LUTs) y los flip-flops, son las partes principales que se deben programar desde un ordenador usando software especializado (por ejemplo, Xilinx Vivado o Intel Quartus).
  - Son ideales para aplicaciones específicas o prototipos.
  - Permiten un **alto nivel de paralelismo**, lo que las hace útiles en telecomunicaciones, procesamiento de señales y sistemas embebidos.

FPGAs - Personalización del Hardware

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Ejercicios
Presentación de prácticas

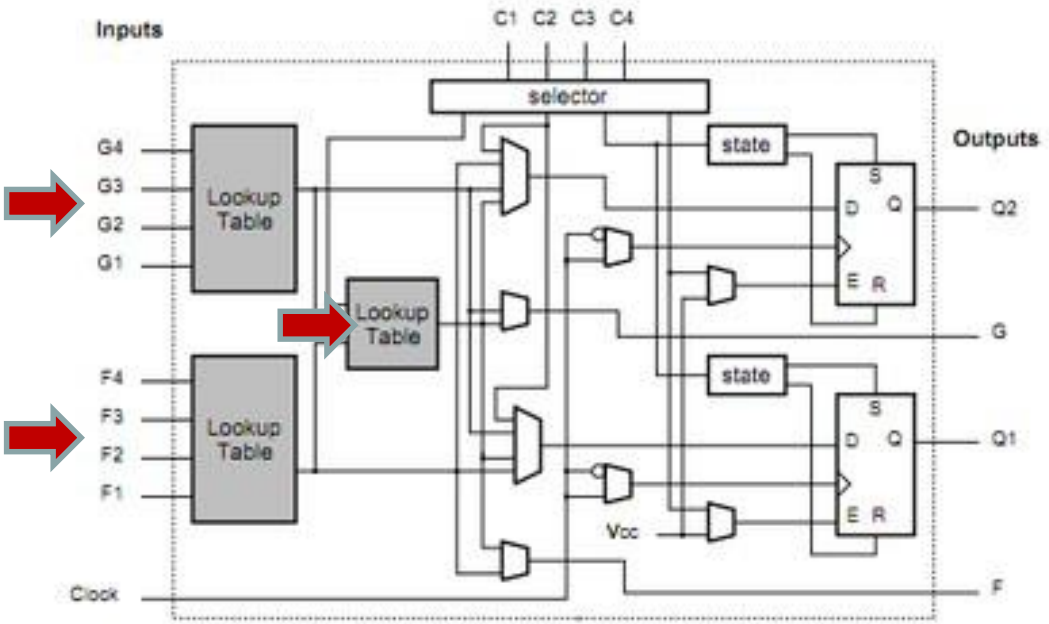
- **Ejemplo práctico (ADAS)**
  - En automoción, las FPGAs son esenciales en los Sistemas Avanzados de Asistencia al Conductor (ADAS).
  - Permiten procesar **rápidamente** datos de múltiples **sensores** como cámaras, radares y lidars.
  - Estos dispositivos ejecutan tareas críticas como el **reconocimiento de objetos** (peatones, señales de tráfico), detección de **carriles** y **monitorización** del entorno.
  - Su **flexibilidad** permite adaptar algoritmos específicos y mejorar funcionalidades, lo que las hace ideales para pruebas y mejoras **sin rediseñar el hardware** continuas en tecnologías de conducción autónoma y asistencia segura.

FPGAs - Personalización del Hardware

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Ejercicios
Presentación de prácticas

- **Ejemplo práctico (ADAS)**

1. **Captura de datos de sensores:** Los datos de cámaras (imágenes), radares (distancias) y lidars (mapas 3D) se envían a las entradas de la FPGA.
2. **Procesamiento en LUTs** (Lookup Tables): Las LUTs ejecutan operaciones como **detección de bordes**, **clasificación de objetos** y **filtrado de ruidos**. Por ejemplo, se detecta si un objeto es un peatón o una señal de tráfico.

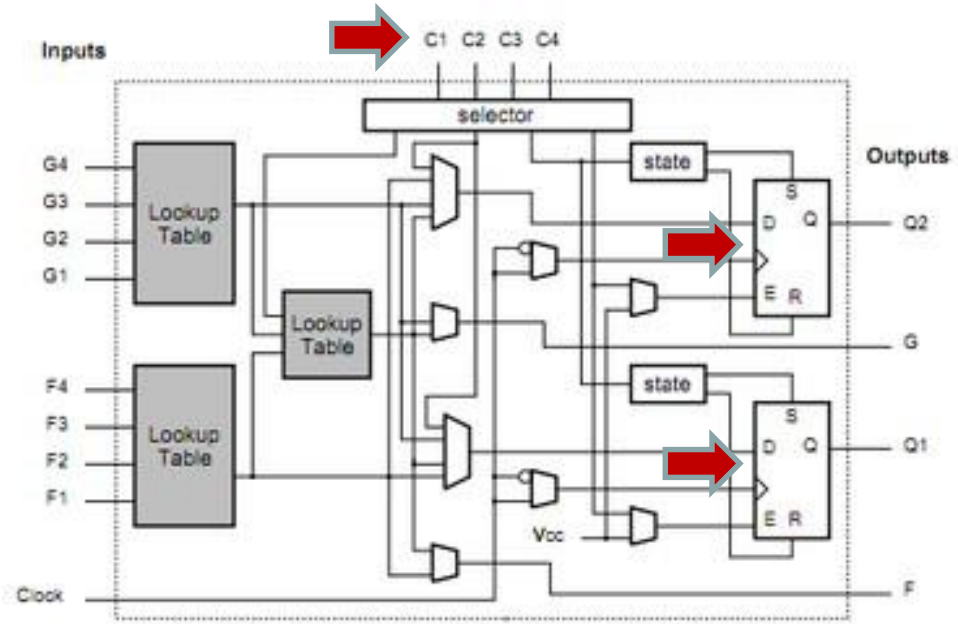


FPGAs - Personalización del Hardware

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Ejercicios
Presentación de prácticas

- **Ejemplo práctico (ADAS)**

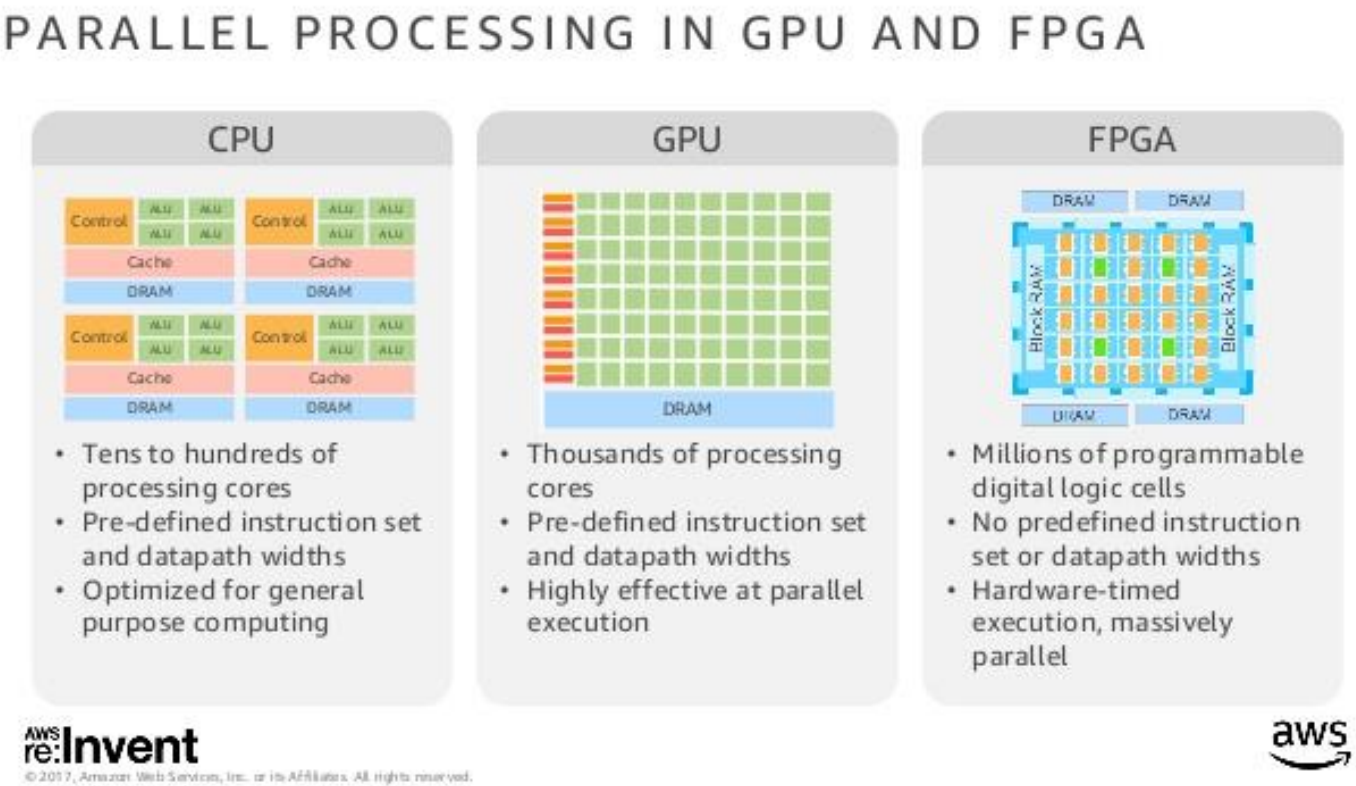
3. **Combinar resultados mediante el selector**: El selector (**C1, C2..)decide** qué información debe pasar a la siguiente etapa o combinarse con otros módulos.
4. **Almacenamiento temporal en flip-flops:** Los resultados intermedios (por ejemplo, detección de carriles) se almacenan y sincronizan con el reloj para su procesamiento ordenado.
5. **Cálculo de decisiones:** Las FPGA analizan los datos procesados para emitir decisiones (como frenar o cambiar de carril) en tiempo real.



FPGAs - Personalización del Hardware

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Ejercicios
Presentación de prácticas

- Comparación General de GPUs, FPGAs y CPUs



FPGAs - Personalización del Hardware

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Ejercicios
Presentación de prácticas

- Comparación General de GPUs, FPGAs y CPUs

Característica	GPUs	FPGAs	CPUs
Paralelismo	Alto	Medio	Bajo
Eficiencia energética	Moderada	Alta	Moderada
Flexibilidad	Media	Alta	Alta
Casos de uso	Simulaciones, redes neuronales	Criptografía, procesamiento en tiempo real	Tareas generales

Benchmarking: Concepto y Tiempos de Ejecución

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Ejercicios
Presentación de prácticas

- **Definición:**
  - El benchmarking en sistemas paralelos mide el rendimiento mediante la comparación de tiempos de ejecución al ejecutar tareas con distintas configuraciones o cargas.
  - Su objetivo es identificar cuellos de botella y mejorar la eficiencia.
- **Tiempos de Ejecución:**
  - **Tiempo Total:**
  - Representa el tiempo necesario para completar una tarea de principio a fin.
  - Incluye todas las fases del proceso y es un indicador global del rendimiento.
  - **Tiempos Parciales:**
  - Son mediciones intermedias (por ejemplo, entre subprocesos o etapas del algoritmo).
  - Sirven para identificar dónde se producen retrasos (cuellos de botella),

Ejercicios

Objetivos
Multiprocesadores
Multicomputadores
Comparativa
Balanceo carga
GPU arquitectura
FPGAs
Comparativa
Benchmarking
Ejercicios
Presentación de prácticas

- ¿Dudas o Preguntas ?
- A continuación puedes practicar los conceptos básicos vistos en ésta sesión mediante algunos ejercicios tipo que puedes encontrar en Moodle