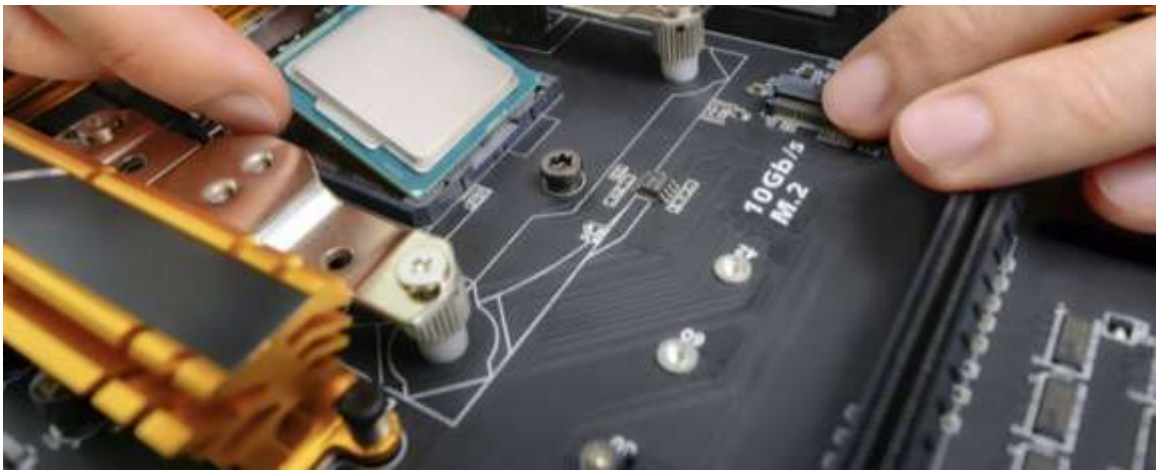
Universidad Europea

LAUREATE INTERNATIONAL UNIVERSITIES

Máquinas de alto rendimiento

Contenido

Presentación



En 1966, Flynn estableció una clasificación del paralelismo que todavía hoy es utilizada para describir las **arquitecturas de procesamiento intenso**. La clasificación de Flynn se basa en dos características:

- **Número de instrucciones que ejecutan en paralelo:** una instrucción (single instruction) o múltiples (multiple instructions).
- **Número de fuentes de datos que se tratan en paralelo:** una entrada (single data) o múltiples (multiple data).

Sobre la base de estas características, la clasificación de Flynn establece **cuatro tipos** de arquitecturas según el paralelismo utilizado en instrucciones y datos:

Tipo	Descripción	Ejemplos
SISD	*Single Instruction, Single Data.*	Monoprocesador convencional.
SIMD	*Single Instruction, Multiple Data.*	Procesador vectorial, procesador en Array.
MISD	*Multiple Instruction, Single Data.*	No existen arquitecturas de este tipo.
MIMD	*Multiple Instruction, Multiple Data.*	Sistemas multinúcleo y multiprocesador.

En este recurso no profundizaremos en ninguna, pero aprenderemos la estructura, el funcionamiento y la aplicación de las siguientes **arquitecturas avanzadas**:

- Procesadores vectoriales.
- Arquitecturas escalares y VLIW.
- Arquitecturas en array.
- Arquitecturas de procesamiento de gráficos (GPU).
- Arquitecturas neuronales.
- Arquitecturas sistólicas.
- DP Pipeline.
- Transputers.

Objetivos

Los **objetivos** que se pretenden alcanzar con este recurso son los siguientes:

- Identificar este **tipo de arquitecturas** y a saber cuál es la **más adecuada** para cada dominio de aplicación.
- Indagar sobre las **tendencias actuales en diseño** de ordenadores de alto rendimiento.
- Analizar las **diferencias existentes** entre cada tipo de arquitecturas avanzadas.

Procesadores vectoriales

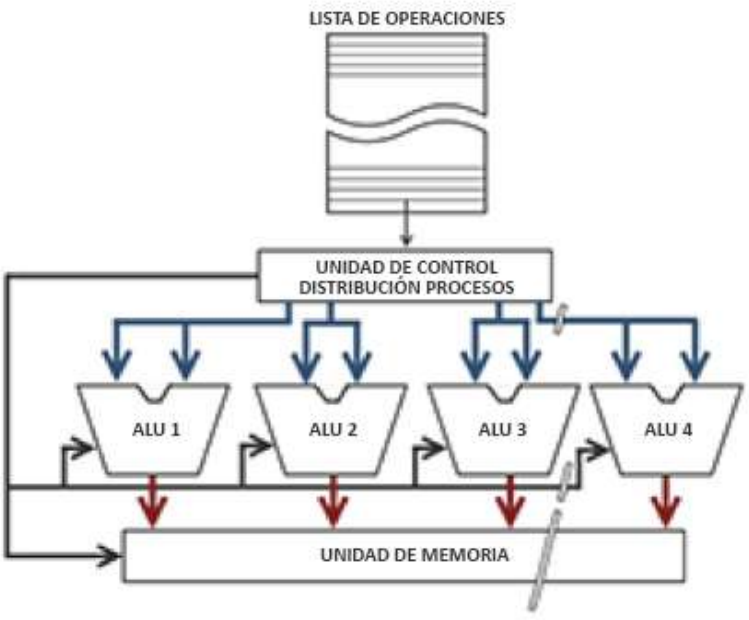
Cuanto mayor es una organización empresarial, más crecen las responsabilidades y el volumen de ésas. Por tanto, la evolución natural de las organizaciones les lleva a ir repartiendo las responsabilidades sobre las actividades que se van desarrollando.

Los **procesadores vectoriales** son un ejemplo de **procesadores SIMD** (un solo flujo de instrucciones, múltiples flujos de datos). Se utilizan normalmente en aplicaciones que requieren **procesamiento intenso sobre estructuras de datos**, como el tratamiento de imágenes. Históricamente se han asociado los procesadores vectoriales a los computadores de la familia CRAY, creados por la compañía CRAY Inc. (CRAY, 2016).

Su principal característica es que la **ALU** (Unidad Aritmético-Lógica) está **segmentada** en varias ALU que pueden trabajar de forma secuencial sobre los datos, o en paralelo. En la figura se muestra un esquema de un procesador vectorial. Las ALU pueden ser vectoriales o escalares (operan con registros vectores o registros).

El controlador puede **distribuir las operaciones y datos** a las diferentes ALU para que operen en paralelo. La ALU en lugar de trabajar solamente con registros individuales, el procesador vectorial almacena los operandos y los resultados de las operaciones en vectores (registros vectores).

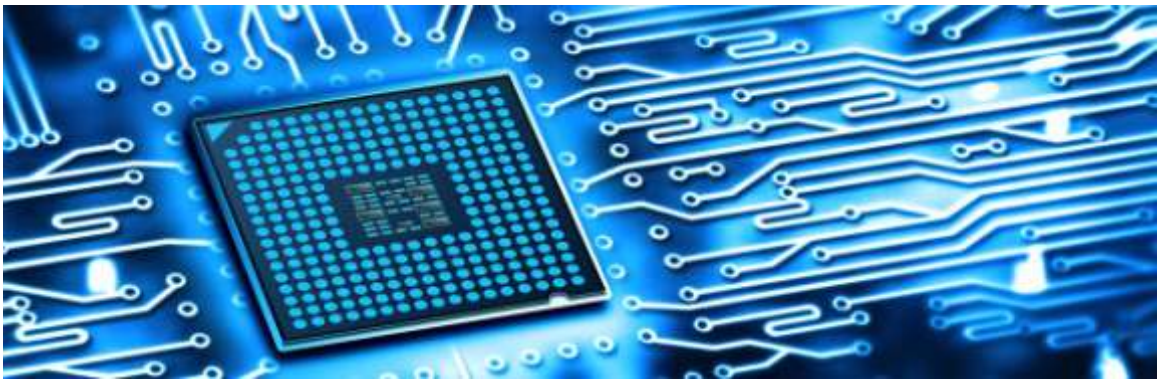
Los procesadores vectoriales fueron la base de los **supercomputadores** entre los años 80 y 90. Los procesadores vectoriales se utilizan sobre todo para trabajar con **grandes estructuras de datos**, como aplicaciones que requieren tratamiento de imágenes.



En detalle -Instrucciones vectoriales

Los procesadores vectoriales incluyen **operaciones vectoriales** en su juego de instrucciones, entre las que destacamos las siguientes:

- **addv.d:** suma dos vectores con valores de doble precisión.
- **addvs.d y mulvs.d:** suma un registro escalar a cada miembro de un vector.
- **mulvs.d:** multiplica un registro escalar a cada miembro de un vector.
- **lv:** carga de datos de doble precisión en un vector.
- **sv:** almacenamiento en memoria de los datos de doble precisión de un vector.

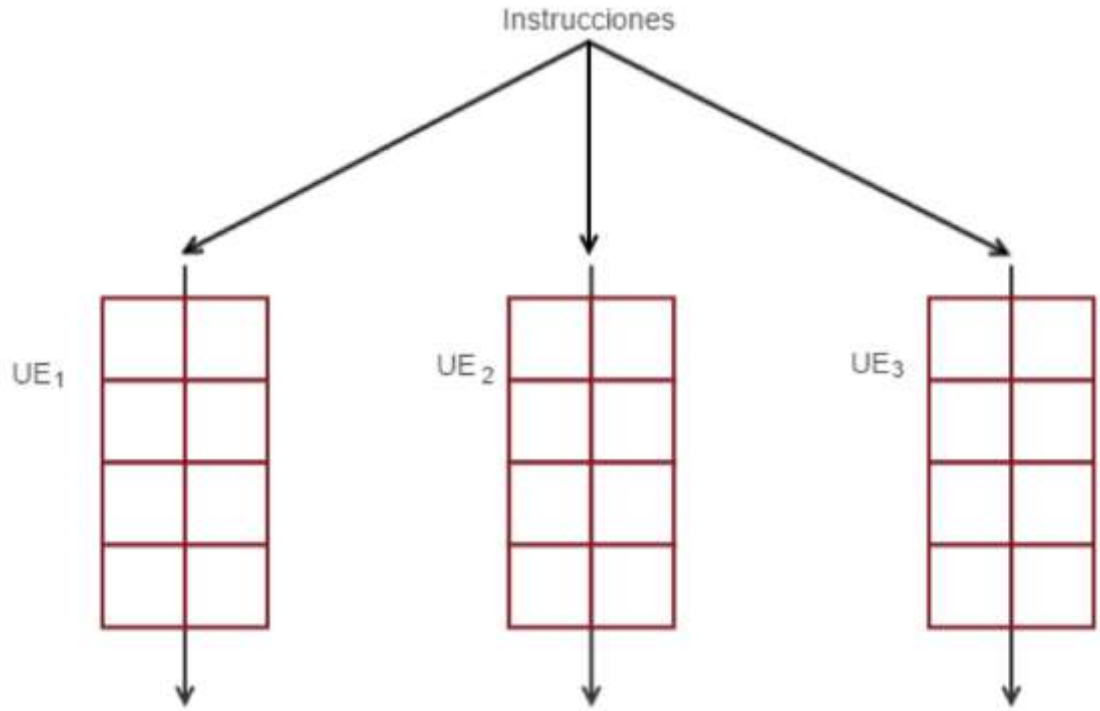


Procesadores superescalares y VLIW

Por **procesador escalar** entendemos aquel procesador que tiene **varias unidades de ejecución**, lo que permite paralelizar la ejecución de instrucciones, aunque la distribución instrucciones se hace de forma secuencial.

En los procesadores superescalares existen varias unidades de ejecución segmentadas que trabajan de forma paralela ejecutando **instrucciones independientes**. La distribución de instrucciones se hace en **paralelo**.

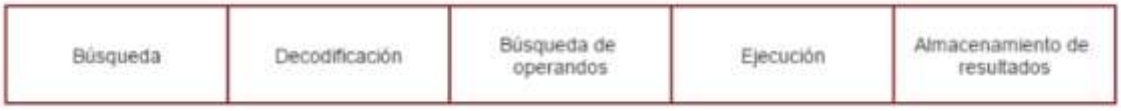
En la figura inferior se muestra un esquema de un procesador superescalar:



Los procesadores **VLIW** siguen la **misma estructura** y funcionamiento que los procesadores superescalares. La **diferencia** entre un procesador VLIW y un procesador superescalar es que el VLIW trabaja con **instrucciones multi-operación independientes entre sí**. Los procesadores superescalares utilizan hardware para detectar y resolver o minimizar el impacto de las dependencias entre instrucciones.

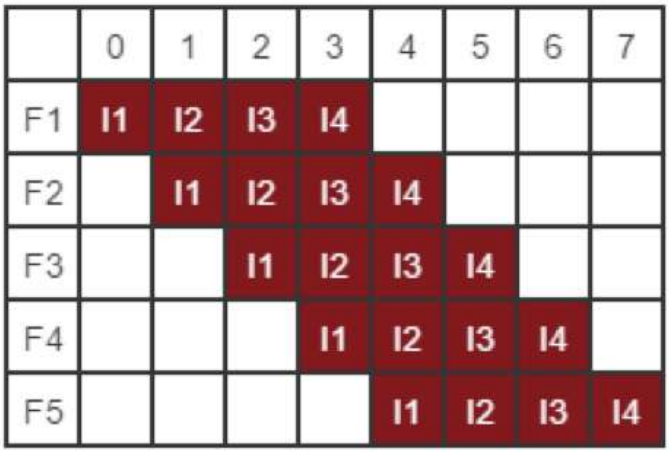
Superescalares versus segmentados

Imaginemos un **procesador segmentado de cinco etapas**:

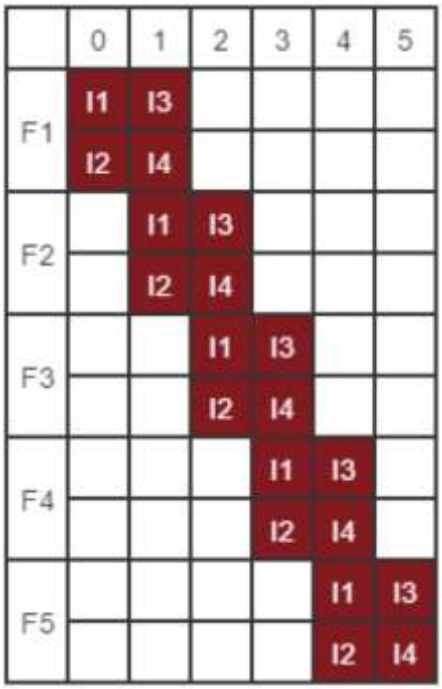


Si comparamos la ejecución de **cuatro instrucciones independientes** en un procesador segmentado y en un procesador superescalar de dos unidades de ejecución segmentadas, el **resultado** sería el siguiente:

Ejecución de cuatro instrucciones en un procesador segmentado



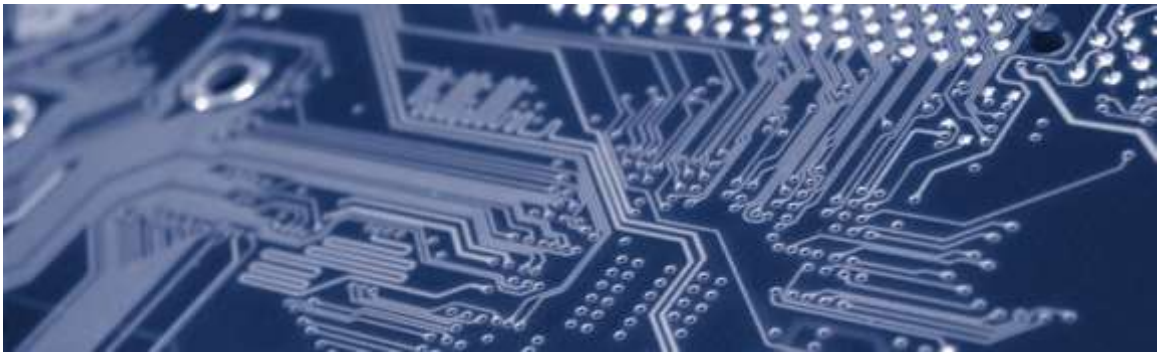
Ejecución de cuatro instrucciones en un procesador superescalar



La mejora respecto de ejecutar cuatro instrucciones en un procesador superescalar, frente a uno segmentado es la siguiente:

Speed-up = 8 / 6 = 1,33 veces más rápido → mejora del 33%.

Si el programa constara de **1 millón de instrucciones independientes** → Speed-up = 1.000.004 / 500.004 = 2 → mejora del 100%.



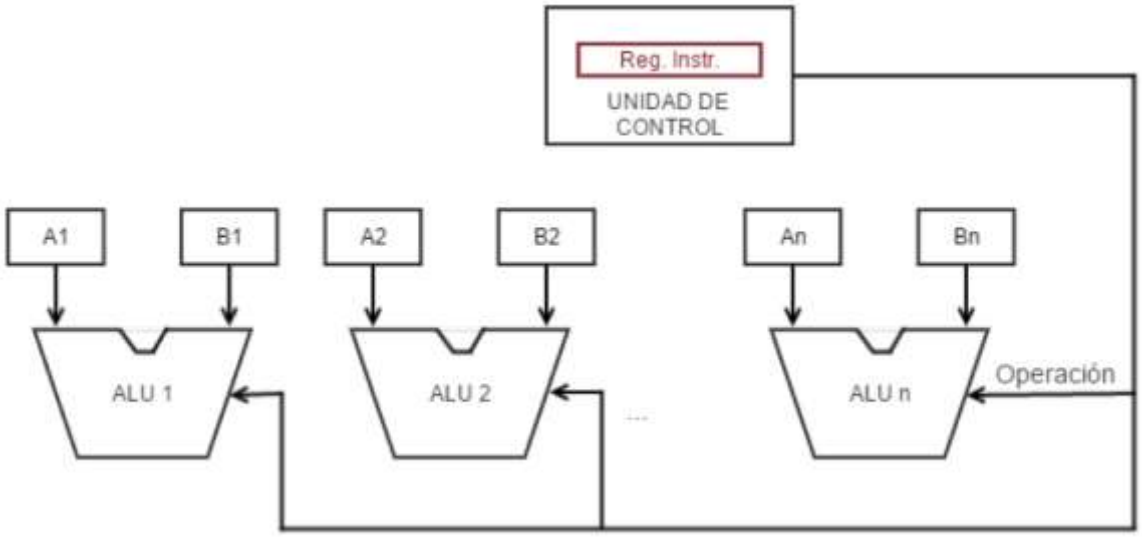
Procesadores en array

Los **procesadores en array** se componen de un **único procesador con varias unidades aritmético-lógicas (ALU).**

Los procesadores en array son arquitecturas de tipo SIMD, con una única unidad de control donde todas las ALU **ejecutan la misma instrucción** (ver figura).

Cada ALU ejecuta con su propia fuente de datos, lo que hace que el grado de paralelismo depende el número de unidades de ejecución.

La principal diferencia entre un procesador en array y un procesador vectorial, es que los procesadores vectoriales pueden ejecutar **instrucciones independientes en paralelo** y con **registros vectores** en lugar de registros escalares, lo que hace que el **rendimiento** de los procesadores vectoriales sea muy superior a los procesadores en array.



Procesador en array

GPU (Graphics Processor Unit)

El **tratamiento de gráficos e imágenes** es una de las tareas que **más recursos consume** en un computador. Para mejorar esta característica en los procesadores, surgieron las unidades de procesamiento gráfico (GPU o Graphics Processor Unit).

Las GPU son un **complemento acelerador** a los procesadores para mejorar el rendimiento del computador cuando se trabaja con aplicaciones de diseño gráfico o videojuegos. Las GPU son "miniprocesadores" que solo pueden ejecutar ciertas operaciones que operan con estructuras de datos matriciales.

La combinación de **CPU-GPU** convierten a las computadoras en **arquitecturas multiprocesadores heterogéneas**, donde la CPU (Central Processing Unit) ejecuta todo aquello que no puede hacer la GPU cuando se trabaja con imágenes.

Para programar las GPU se utilizan unas interfaces de programación de alto nivel específicas para la generación y modificación de imágenes tridimensionales, llamadas **API gráficas**. Estas API constan de primitivas para dibujar vértices, líneas, polígonos, figuras 3D basadas en triángulos y en la generación de imágenes y sombras basadas en píxeles.

Las API gráficas **más utilizadas** en la actualidad son:

- OpenGL (OpenGL, 2016): es multiplataforma.
- Microsoft Direct X: en entornos Microsoft Windows (Microsoft, 2016).

Arquitecturas neuronales

Las **arquitecturas neuronales (ANN)** son computadoras de propósito específico que se utilizan para **automatizar tareas complejas** que no son sencillas de implementar con un algoritmo secuencial.

Este tipo de arquitecturas se denominan "neuronales" porque trabajan **simulando el funcionamiento de las neuronas** en el sistema nervioso humano (ver figura inferior), donde los nodos se conectan entre sí formando "redes neuronales". Cada nodo ejecuta cálculos sencillos pero no lineales, es decir, el resultado puede variar en diferentes ejecuciones.

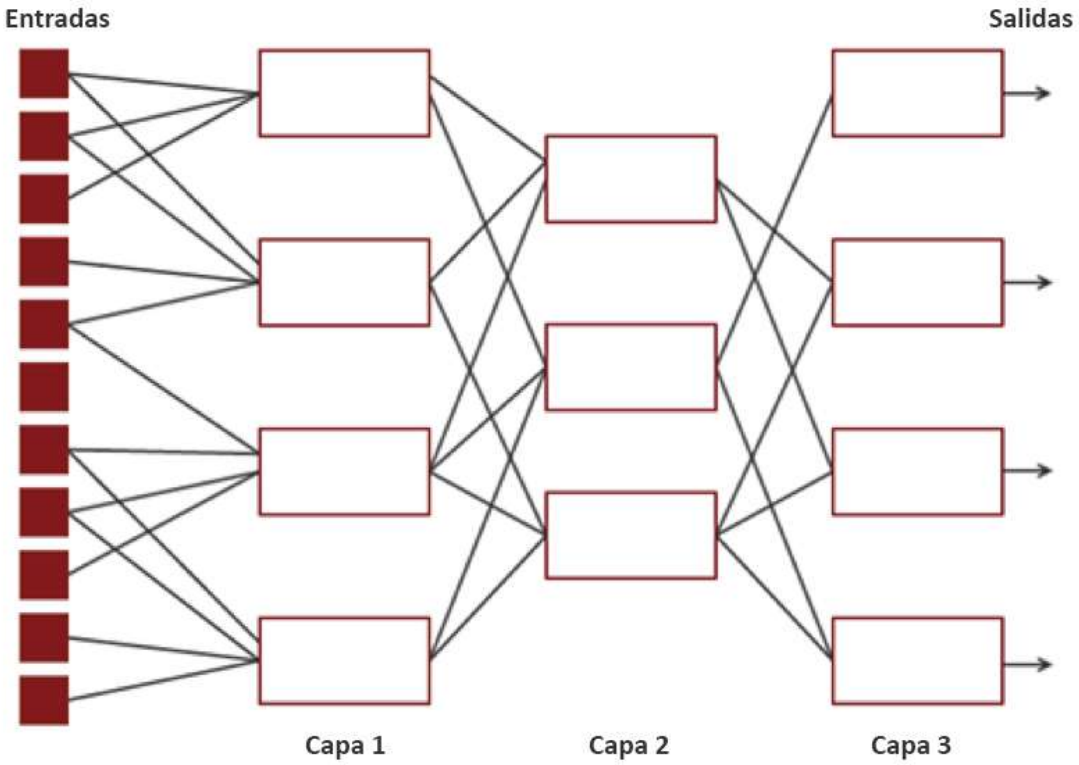
La arquitectura debe pasar un **proceso de entrenamiento** antes de funcionar correctamente.

Existen **tres tipos de aplicaciones** de las arquitecturas neuronales:

Se recibe un elemento en forma de conjunto de datos y se clasifica el tipo de ejemplo recibido.

Se recibe un elemento en forma de conjunto de datos y los datos van evolucionando hasta generar una salida. Se utiliza para cambiar la forma de representación de los datos (traducir de un formato a otro).

Una posible aplicación de las arquitecturas neuronales es la compresión de datos para reducir su tamaño.

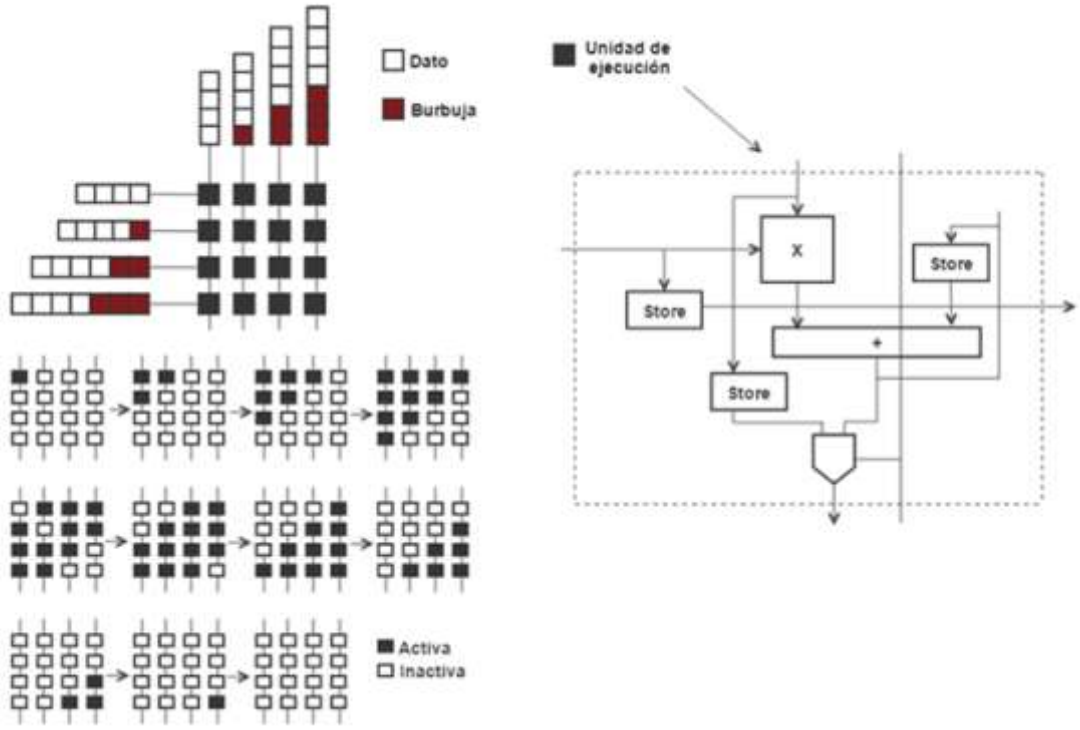


Arquitecturas sistólicas

Las **arquitecturas sistólicas** son arquitecturas **paralelas** de propósito específico. El término "sistólico" viene de la forma en la que trabajan **simulando el sistema de bombeo del corazón**.

Los procesadores sistólicos constan de **varias unidades de ejecución** idénticas que se comunican entre sí punto a punto para mejorar el rendimiento del procesador, ya que este modelo de interconexión reduce tiempo de transferencia de datos entre unidades enlazadas y reducir así la necesidad de almacenar información en la memoria central.

Una posible aplicación de una arquitectura sistólica es la **multiplicación de matrices de 4x4** de una forma óptima:



Multiplicación de matrices en un procesador sistólico

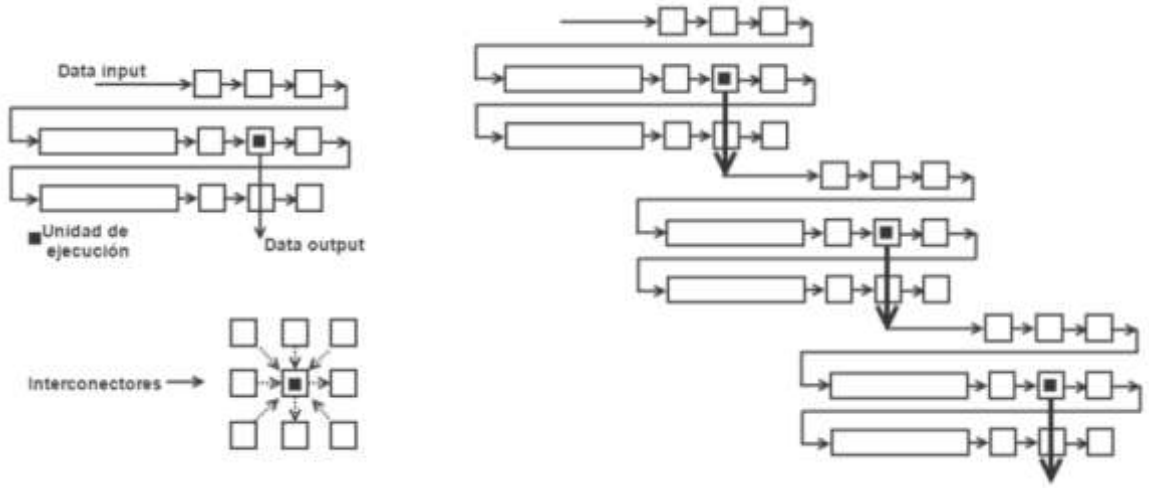
Arquitecturas DP pipeline

Las arquitecturas **DP pipeline** son **similares a las arquitecturas sistólicas** en el sentido de que ambas se utilizan para **aplicaciones específicas**, y ambas tienen la capacidad de dividir una operación compleja sobre una estructura de datos en pequeñas operaciones. Los resultados de cada operación ejecutada en una unidad de ejecución se utilizan como **entradas de datos** en la siguiente operación ejecutada en otro nodo.

Estas arquitecturas ofrecen un alto rendimiento en el **tratamiento de imágenes**.

En la figura inferior, se representa **una aplicación de tratamiento de imágenes**, donde los **datos de entrada son píxeles** con un color concreto, y se modifican los píxeles implementado una función que tiene en cuenta el color de los píxeles de alrededor.

En la figura de la izquierda se representa la **ejecución del cálculo de un pixel**, mientras que en la figura de la derecha se representa una ejecución en cadena (pipeline) para tratar una imagen.



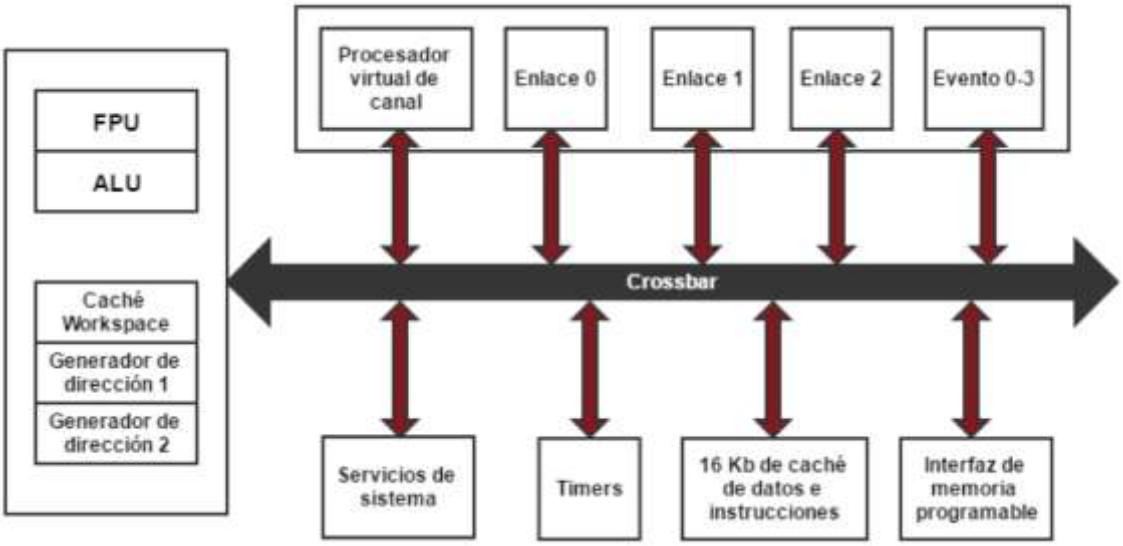
Transputers

Los **transputers** son un tipo de **arquitecturas MIMD de tipo multicomputador**, donde los procesadores se sitúan en chips separados.

Están diseñados para implementar una **programación paralela y síncrona**, optimizando la comunicación entre procesos.

Las arquitecturas **DP pipeline** son **similares a las arquitecturas sistólicas** en el sentido de que ambas se utilizan para **aplicaciones específicas**, y ambas tienen la capacidad de dividir una operación compleja sobre una estructura de datos en pequeñas operaciones. Los resultados de cada operación ejecutada en una unidad de ejecución se utilizan como **entradas de datos** en la siguiente operación ejecutada en otro nodo.

Posteriormente, se mejoró el rendimiento del computador en **ejecuciones secuenciales** y el sistema de interconexión (*red crossbar*) entre procesadores, añadiendo mecanismos de enrutamiento. Estas arquitecturas se centran más en **mejorar las comunicaciones** que en el rendimiento de los procesadores.



Ejemplo de arquitectura basada en transputers: T9000

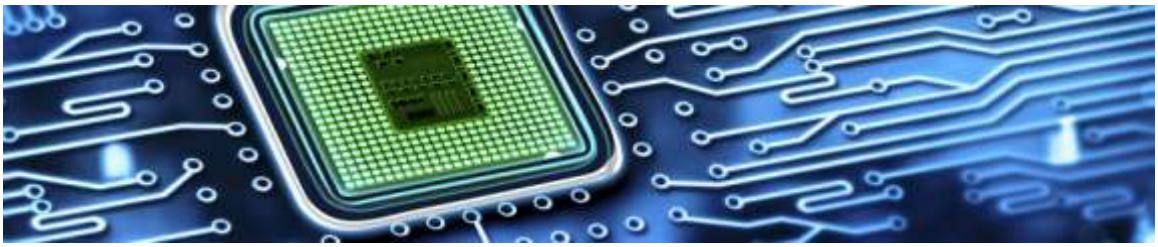
Resumen

En este recurso hemos conocido **arquitecturas especiales** utilizadas para procesamiento intenso como los procesadores vectoriales, procesadores superescalares, procesadores en array, GPU, arquitecturas neuronales, arquitecturas sistólicas, arquitecturas DP pipelined y transputers.

Algunas de estas arquitecturas se utilizan para **aplicaciones específicas** como las GPU, neuronales, sistólicas y pipeline.

Es importante que un **ingeniero informático** sea capaz de:

- Identificar estas arquitecturas.
- Conocer su funcionamiento.
- Saber cuáles son sus posibles aplicaciones.
- Saber decidir cuál es la arquitectura más adecuada a cada situación, teniendo en cuenta el tipo de aplicación al que se destinará la máquina.



Referencias bibliográficas

- CRAY (2016). CRAY. Acceso web: http://www.cray.com/
- Marcellus D.H (1984). Systems programming for Computers. Prentice Hall.
- Microsoft (2016). Acceso web: www.microsoft.com,
- Newell, A. and Simon, H. 19, 3 (Mar. 1976), pp. 113-126. Computer science as empirical inquiry: symbols and search. Communications of the ACM.
- OpenGL (2016). OpenGL. Acceso Web: www.opengl.org
- Patterson D., Hennessy J. (2011). Estructura y Diseño de Computadores: Interfaz Hardware/Software. Editorial Reverté. Cuarta Edición. Traducido y adaptado de: Patterson & Hennessy, Computer Organization and Design: Hardware/Software Interface. Forth Edition. 2009. Ed. Elsevier.
- Sony Computer Entertainment Europe (2016). PlayStation. Acceso web: www.playstation.com
- Tanembaum, A.S. (1999). Structured Computer Organization (4th. ed.). Englewood Cliffs N.J. Prentice-Hall.