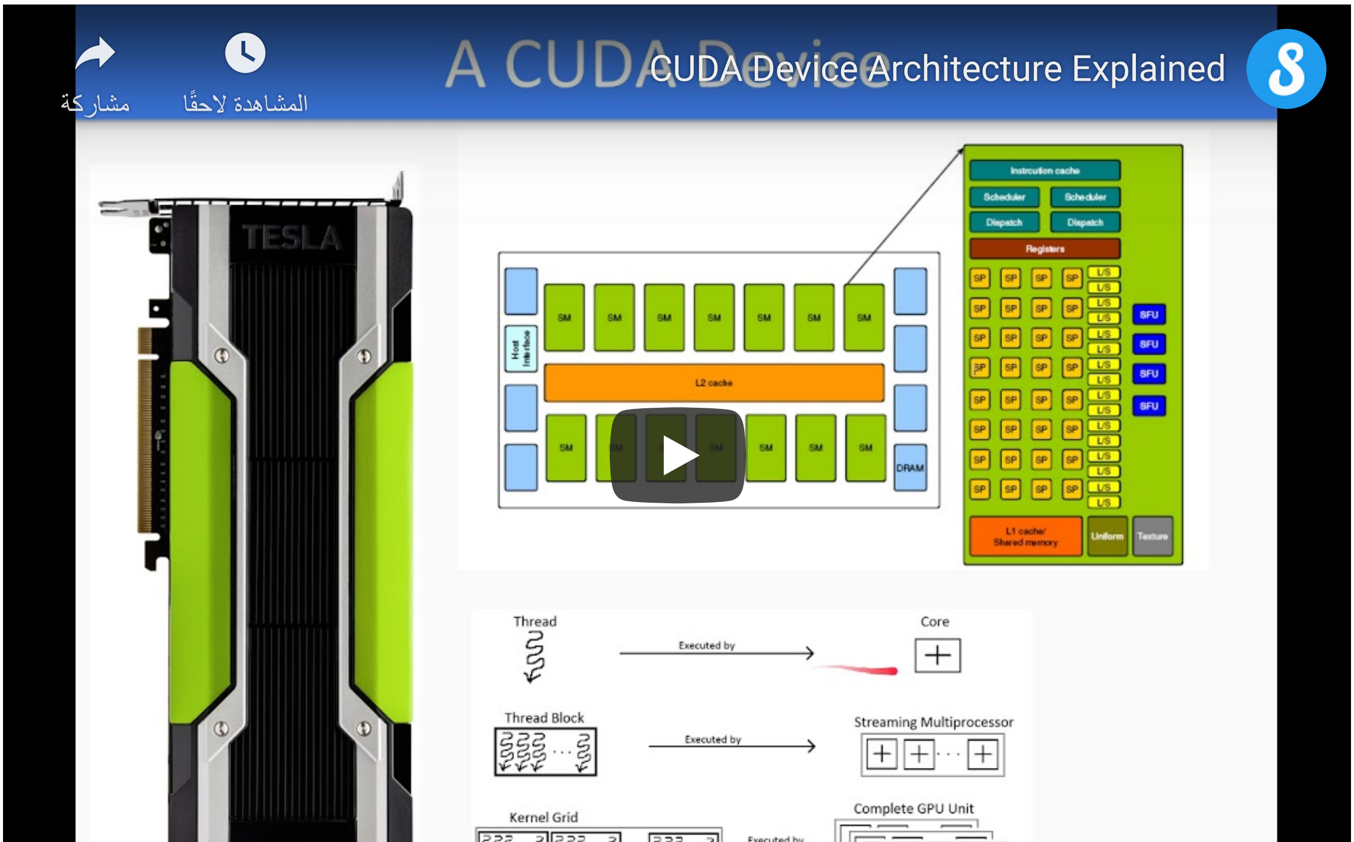
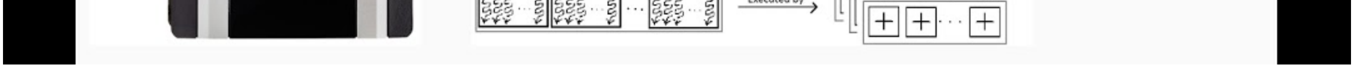
GPUs - graphics processing units

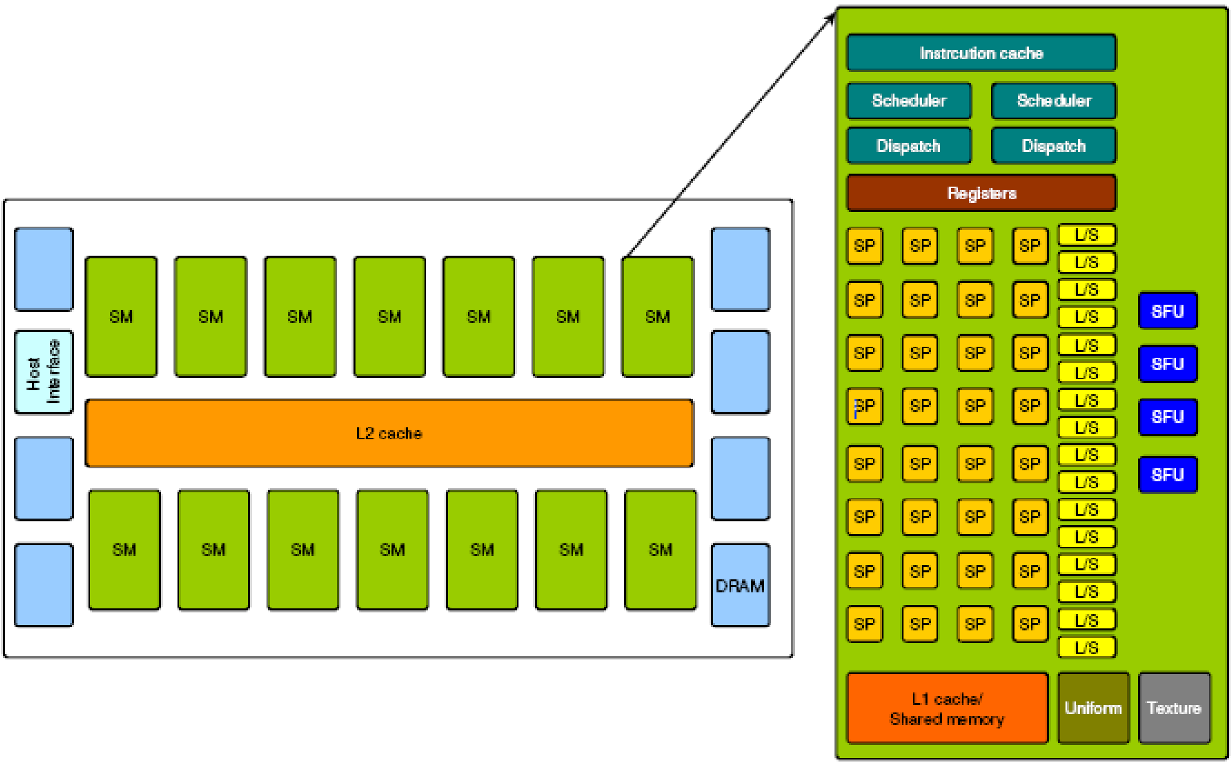
The term **GPU** was popularized by nVIDIA in 1999, who marketed a computer graphics card called GeForce 256 as “The world’s first GPU”. However, the card was mainly designed for rendering of high-end computer graphics and enhancing computer-based gaming performance. In contrast, today’s GPUs also provide an inexpensive platform for developing and executing high-performance non-graphical applications. The development of general-purpose applications on GPUs is often termed as **GPGPU (General Purpose Computing on GPUs)** and a number companies have started to produce GPUs that are capable of general purpose computation.

Among the various kinds of GPUs, we have specifically chosen nVIDIA’s **Tesla** is a popular development platform, however latest GPUs like **Fermi K80, Pascal, Volta** are also not very different in terms of basics. All these devices allow programmers to develop applications using an easily programmable C-like language, called **Compute Unified Device Architecture (CUDA)**.



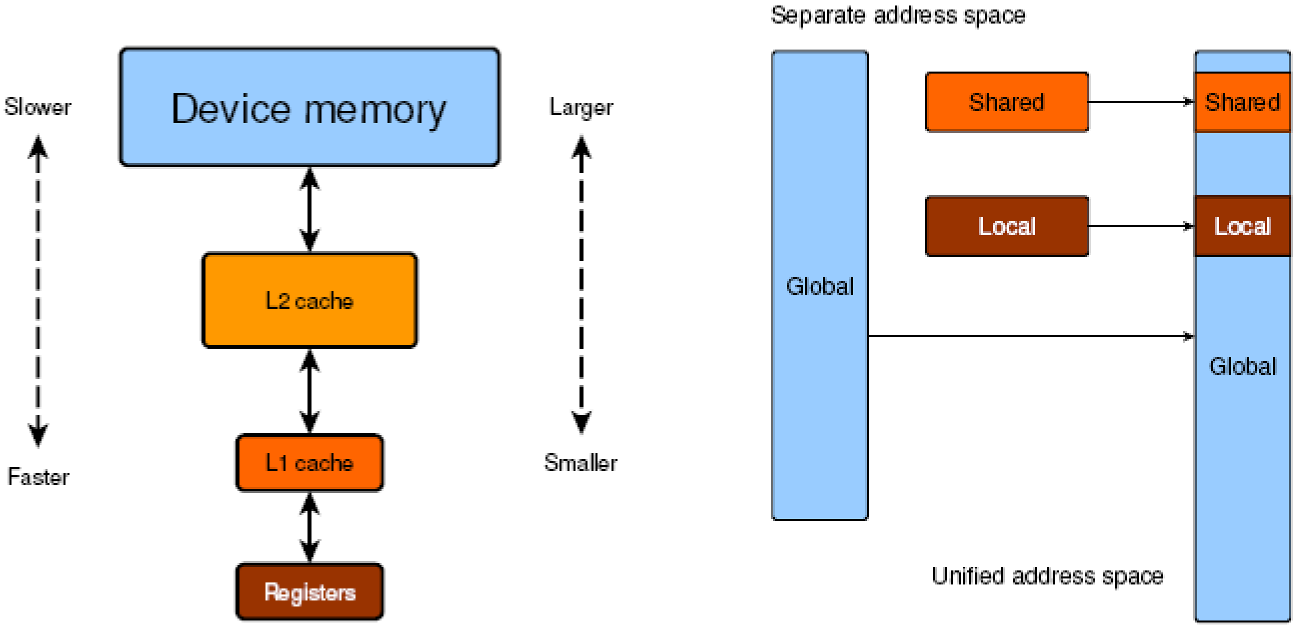


CUDA Device



Nvidia's Tesla unified GPU architecture

A schematic diagram of the Tesla’s unified GPU architecture is shown in the figure above. It illustrates the framework of a Tesla C2050 device. The device has a total of 448 **streaming-processor (SP)** cores organized as a group of 32 stream processors (also called CUDA cores), in 14 **streaming multiprocessors (SM)**. Each core here executes a sequential thread in a so-called **SIMT (Single Instruction, Multiple Thread)** fashion and all the threads in a same wrap execute the same instruction at the same time, where a wrap is a group of 32 threads. Tesla supports up to 32 active warps on each SM. If one warp stalls at any conditional operation, then it selects another ready warp so that the cores can remain active.



(b) Memory hierarchy

(c) Memory addressing

GPU memory

Similar to other GPUs, the Tesla device has a hierarchy of on-board memory, such as a small and fast programmable **L1 cache/shared memory (16 – 48 KBs)**, a fully coherent **L2 cache memory (512–768 KBs)** and a relatively large on-board DRAM or device **main memory (3–6 GBs)**. The L1 cache is attached to each multiprocessor and shared among the comprising cores, where the unified L2 cache is shared across the device. The main task of L2 memory is to minimize the effect of the long latency of device DRAM. There are also some register files (128 KBs), texture and constant caches within each SM. At the software level, all the memory levels are unified into a single continuous address space.

There exist a number of application programming interfaces (API) that can enable programmers to access the device memory and develop GPU-based applications.