Multicore Computers

Introduction

Multicore refers to an architecture in which a single physical processor incorporates the core logic of more than one processor. A single integrated circuit is used to package or hold these processors. These single integrated circuits are known as a die. Multicore architecture places multiple processor cores and bundles them as a single physical processor. The objective is to create a system that can complete more tasks at the same time, thereby gaining better overall system performance.

This technology is most commonly used in multicore processors, where two or more processor chips or cores run concurrently as a single system. Multicore-based processors are used in mobile devices, desktops, workstations and servers.

Hardware Performance Issues

Microprocessor systems have experienced a steady, exponential increase in execution performance for decades. This increase is due to refinements in the organization of the processor on the chip, and the increase in the clock frequency.

1. **Increase in Parallelism**
   The organizational changes in processor design have primarily been focused on increasing instruction-level parallelism, so that more work could be done in each clock cycle.

   Pipelining: Individual instructions are executed through a pipeline of stages so that while one instruction is executing in one stage of the pipeline, another instruction is executing in another stage of the pipeline.

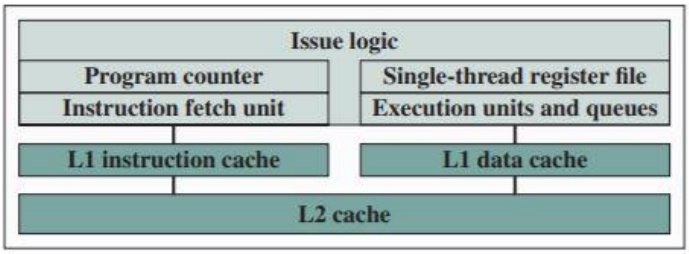
   In the case of pipelining, simple three-stage pipelines were replaced by pipelines with five stages, and then many more stages, with some implementations having over a dozen stages. There is a practical limit to how far this trend can be taken, because with more stages, there is the need for more logic (hardware), more interconnections and more control signals.

2. **Alternative Chip Organization**

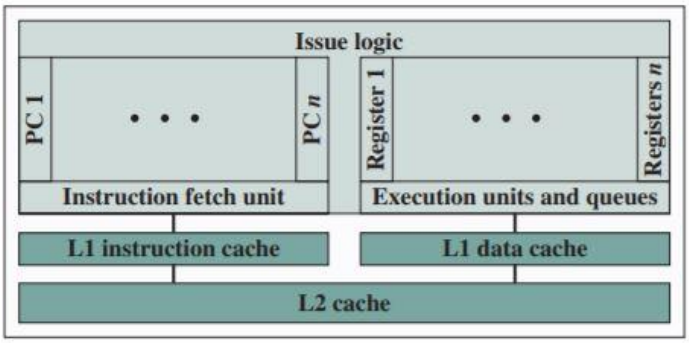
   Superscalar: Multiple pipelines are constructed by replicating execution resources. This enables parallel execution of instructions in parallel pipelines, so long as hazards are avoided.
   With superscalar organization, performance increase can be achieved by increasing the number of parallel pipelines. There are limitations, as the number of pipelines increases. More logic is required to manage hazards and to stage instruction resources. A single thread of execution reaches the point where hazards and resource dependencies prevent the full use of the multiple pipelines available. As the complexity of managing multiple threads over a set of pipelines limits the number of threads and number of pipelines that can be effectively utilized.

Simultaneous multithreading (SMT): Register banks are replicated so that multiple threads can share the use of pipeline resources.

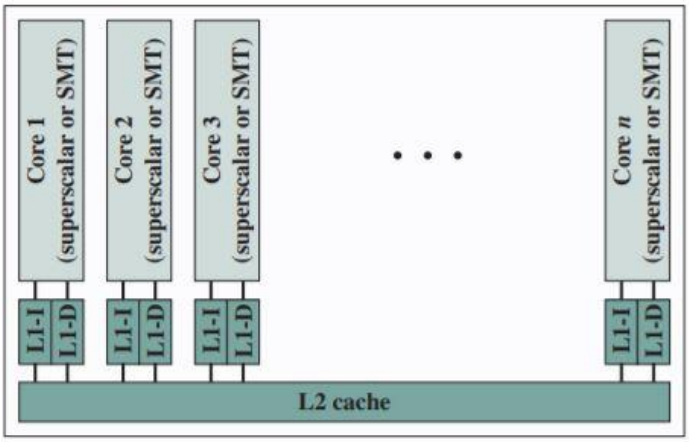
The increase in complexity to deal with all of the logical issues related to very long pipelines, multiple superscalar pipelines, and multiple SMT register banks means that increasing amounts of the chip area is occupied with coordinating and signal transfer logic.
This increases the difficulty of designing, fabricating, and debugging the chips.



(a) Superscalar



(b) Simultaneous multithreading



(c) Multicore

3. **Power Consumption**

   To maintain the higher performance, the number of transistors per chip rise and high clock frequencies. Unfortunately, power requirements have grown exponentially as chip density and clock frequency have risen. One way to control power density is to use more of the chip area for cache memory. Memory transistors are smaller and have a power density an order of magnitude lower than that of logic.

   The recent decades the Pollack's rule was observed, which states that performance increase is roughly proportional to square root of increase in complexity. If you double the logic in a processor core, then it delivers only 40% more performance. The use of multiple cores has the potential to provide near-linear performance improvement with the increase in the number of cores.

   Power considerations provide another motive for moving toward a multicore organization. The chip has such a huge amount of cache memory, it becomes unlikely that any one thread of execution can effectively use all that memory.

   In SMT, a number of relatively independent threads or processes has a greater opportunity to take full advantage of the cache memory.

Software Performance Issues

A detailed examination of the software performance issues related to multicore organization is huge task.

The performance benefits of a multicore organization depend on the ability to effectively exploit the parallel resources.

Consider, single application running on a multicore system. 20 The law assumes a program in which a fraction (1-f) of the execution time involves code serial and a fraction f that involves code is infinitely parallelizable with no scheduling overhead.

A number of classes of applications benefit directly from the ability to scale throughput with the number of cores. Multithreaded native applications: Multithreaded applications are characterized by having a small number of highly threaded processes.

Examples of threaded applications include Lotus Domino or Siebel CRM (Customer Relationship Manager). Multiprocess applications: Multiprocess applications are characterized by the presence of many single-threaded processes. Examples of multi-process applications include the Oracle database, SAP, and PeopleSoft.

Java language greatly facilitate multithreaded applications. Java Virtual Machine is a multithreaded process that provides scheduling and memory management for Java applications. Java applications that can benefit directly from multicore resources include application servers such as Sun's Java Application Server, BEA's Weblogic, IBM's Websphere etc. All applications that use a Java 2 Platform, Enterprise Edition (J2EE platform) application server can immediately benefit from multicore technology.

Multiinstance applications: Even if an individual application does not scale to take advantage of a large number of threads, it is still possible to gain advantage from multicore architecture by running multiple instances of the application in parallel. If multiple application instances require some degree of isolation, then virtualization technology can be used to provide each of them with its own separate and secure environment.

Multicore Organization

The main variables in a multicore organization are as follows:

- The number of core processors on the chip.
- The number of levels of cache memory.
- The amount of cache memory that is shared

General Organization of Multicore Systems

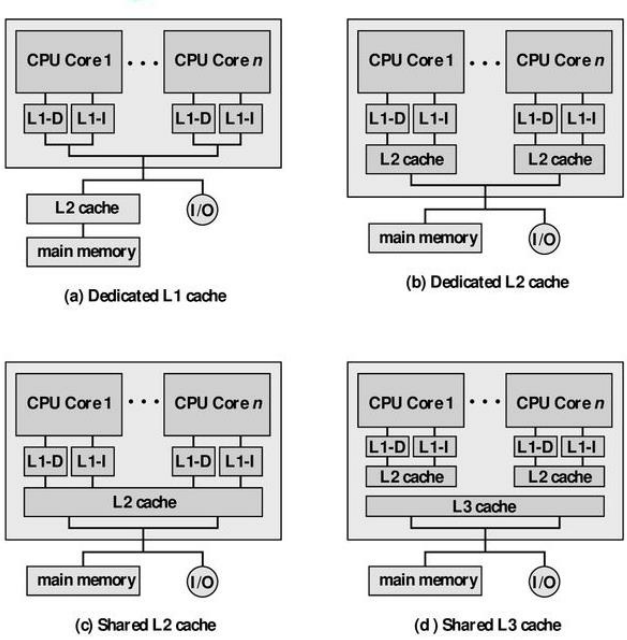


Figure (a) shows an organization found in some of the earlier multicore computer chips and is still seen in embedded chips. In this organization, the only on-chip cache is L1 cache, each core having its own dedicated L1 cache. Almost invariably, the L1 cache is divided into instruction and data caches. An example of this organization is the ARM11 MP core.

Figure (b) is the organization in which there is no on-chip cache sharing. In this, there is enough area available on the chip to allow for L2 cache. An example of this organization is AMD Opteron.

Figure (c) shows allocation of chip space to memory, but with the use of a shared L2 cache. Intel Core Duo has this organization.

The amount of cache memory available on the chip continues to grow, performance considerations to grow, performance considerations dictate splitting off a separate, shared L3 cache with dedicated L1 and L2 cache for each core processor. Intel core i7 is an example of shared L3 cache organization.

Benefits of the shared cache architecture

- Efficient usage of the last-level cache If one core idles, the other core takes all the shared cache
- Reduces resource underutilization
- Allows more data-sharing opportunities for threads running on separate cores that are sharing cache
- One core can pre-/post-process data for the other core
- Alternative communication mechanisms between cores
- Reduce cache-coherency complexity
- Reduced false sharing because of the shared cache
- Less workload to maintain coherency, compared to the private cache architecture
- The same data only needs to be stored once
- Effective data sharing between cores allows data requests to be resolved at the shared-cache level instead of going all the way to the system memory

Dual Core and Quad Core Processors

ISSUE :	DUAL CORE	QUAD CORE
No of cores or Data lines	Dual core has 2 processing Cores	Quad core has 4 processing cores
Speed	Dual core is less powerful in terms of speed.	Quad core is faster
Task	Dual core does not support Multi-tasking like quad core	Quad core is designed for multi tasking
Heat	Dual core is lighter and no heat is generated when working.	Quad core processors generates heat which can heat up the device
Energy consumption	Dual core consumes less power	Quad core consumes more power
Video	Dual core lacks in Graphics	Quad core is better equipped to handle high quality graphics

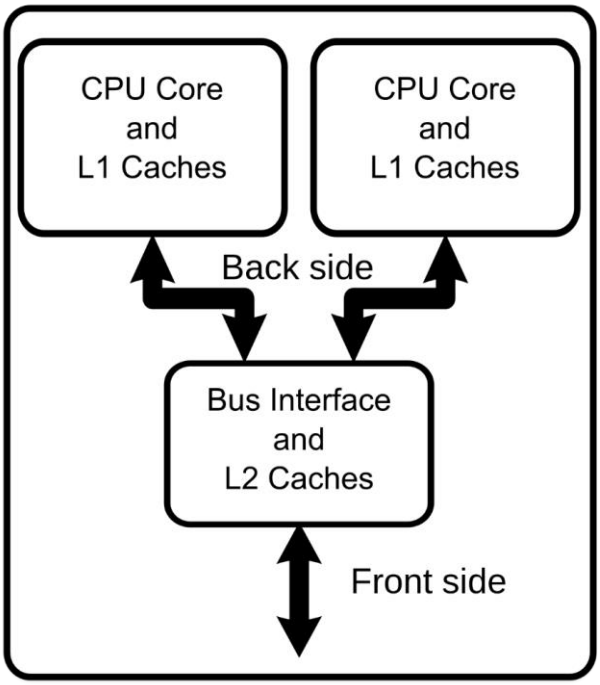


Figure: Dual Core Organization

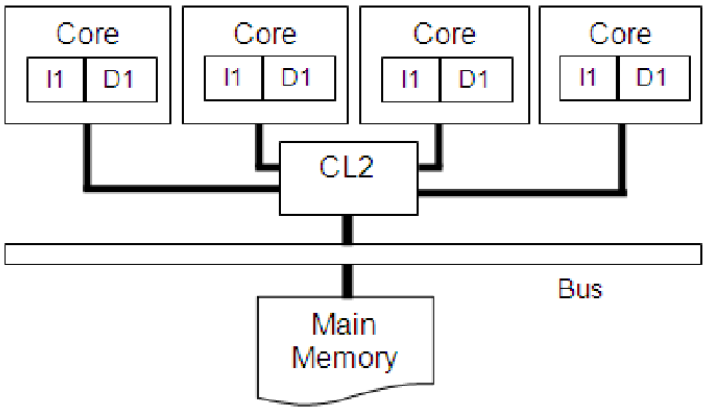


Fig: Quad Core Organization

Power Efficient Processor

- In last few years, power PC dissipation has become an important design issue in context of performance in the design of new computer systems.
- Whereas in the past, the primary job of the computer architect was to translate improvement in operating frequency and transistor count into performance. Now, power efficiency must be considered at every step of the design process.
- Following points are considered while designing power efficient processor :
    - Latency: time until task completion.
    - Throughput: amount of work done per unit time.
- Designing power efficient processors, various techniques are used:
    - Pipelining
    - Multicore processors
    - Multilevel cache