

# Math Co-processor

---

- ▶ Intel family of math coprocessor are generally labelled as 80x87, which support 80x86 processors in computing.
- ▶ Instruction sets and programming of all devices are almost identical
- ▶ Intel family of math-coprocessors is able to
  - ***add***
  - ***subtract***
  - ***multiply***
  - ***divide***
  - ***find square root***
  - ***calculate partial tangent***
  - ***calculate partial arctangent***
  - ***logarithms***

# Intel Processors and Co-processors

---

Processor	Corresponding Co-processor
8086/8088	8087
80186/80188	80187
80286	80287
80386	80387SX, 80387DX
80486SX	80487SX

# CISC and RISC Processors

---

## ▶ **CISC (Complex Instruction Set Computers):**

- ▶ CISC was developed to make compiler development simpler.
- ▶ It shifts most of the burden of generating machine instructions to the processor.
  - ▶ ***For example***, instead of having to make a compiler write long machine instructions to calculate a square-root, a CISC processor would have a built-in ability to do this.

# CISC and RISC Processors

---

## ▶ **RISC (Reduced Instruction Set Computers):**

- ▶ RISC is a type of microprocessor architecture that utilizes a small, highly-optimized set of instructions, rather than a more specialized set of instructions often found in other types of architectures.

## ▶ **History**

The first RISC projects came from IBM, Stanford, and UC-Berkeley in the late 70s and early 80s. The IBM 801, Stanford MIPS, and Berkeley RISC 1 and 2 were all designed with a similar philosophy which has become known as RISC.

# CISC and RISC Processors

---

The main characteristics of **CISC** microprocessors are:

- ▶ Extensive instructions.
- ▶ Complex and efficient machine instructions.
- ▶ Extensive addressing capabilities for memory operations.
- ▶ Relatively few registers in use.

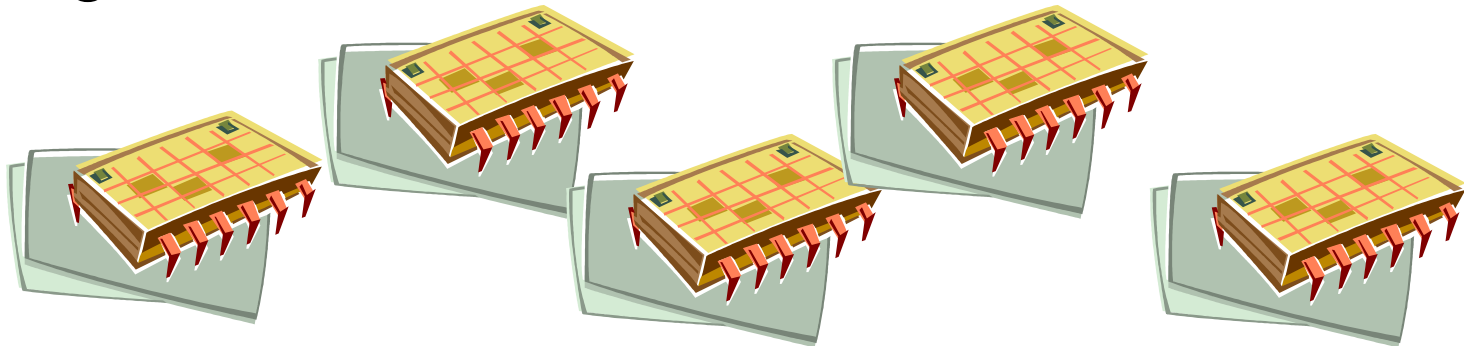
In comparison, **RISC** processors are more or less the opposite of the above:

- ▶ Reduced instruction set.
- ▶ Less complex, simple instructions.
- ▶ *Pipelining* is used to allow for simultaneous execution of parts, or stages, of instructions
- ▶ Many symmetric registers are used.

# Age of Multi-core Processors

---

- ▶ From 1986 – 2002, microprocessors were speeding like a rocket, increasing in performance an average of 50% per year.
- ▶ Since then, it's dropped to about 20% increase per year.
- ▶ **An Intelligent Solution:**
  - ▶ Instead of designing and building faster microprocessors, put multiple processors on a single integrated circuit.



# Is Multi-core Processors is the Solution?

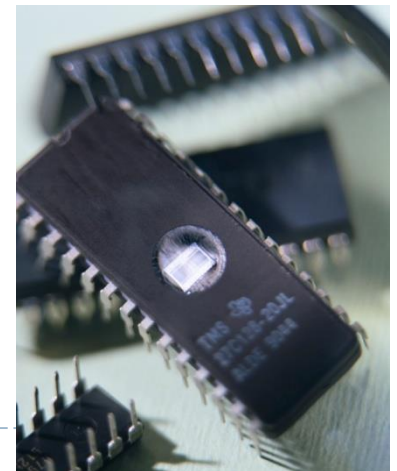
---

- ▶ Up to now, performance increases have been attributable to increasing density of transistors in multiple microprocessors.
- ▶ But there are inherent problems:
  - ▶ *Smaller transistors = faster processors*
  - ▶ *Faster processors = increased power consumption*
  - ▶ *Increased power consumption = increased heat*
  - ▶ *Increased heat = unreliable processors*

# Multi-core Processor and Parallel Programming

---

- ▶ Adding more processors doesn't help much if programmers aren't aware of them...
- ▶ ... or don't know how to use them.
- ▶ Serial programs don't benefit from this approach (in most cases).
- ▶ Move away from single-core systems **to multi-core processors and Introduce parallelism!!!**
- ▶ “Core” = central processing unit (CPU)





# Parallel Programming Concept for Multi-core Processors

---

- ▶ Compute n values and add them together.
- ▶ **Serial solution:**

```
sum = 0;
for (i = 0; i < n; i++) {
    x = Compute_next_value(. . .);
    sum += x;
}
```

- ▶ n iteration requires to add.

# Parallel Programming Concept for Multi-core Processors

- ▶ 8 cores,  $n = 24$ , then the calls to
- ▶ 1,4,3, 9,2,8, 5,1,1, 5,3,7, 2,5,0, 4,1,8, 6,5,1, 2,3,9
- ▶ The Master Core runs the serial program

```
if (I'm the master core) {  
    sum = my_x;  
    for each core other than myself {  
        receive value from core;  
        sum += value;  
    }  
} else {  
    send my_x to the master;  
}
```

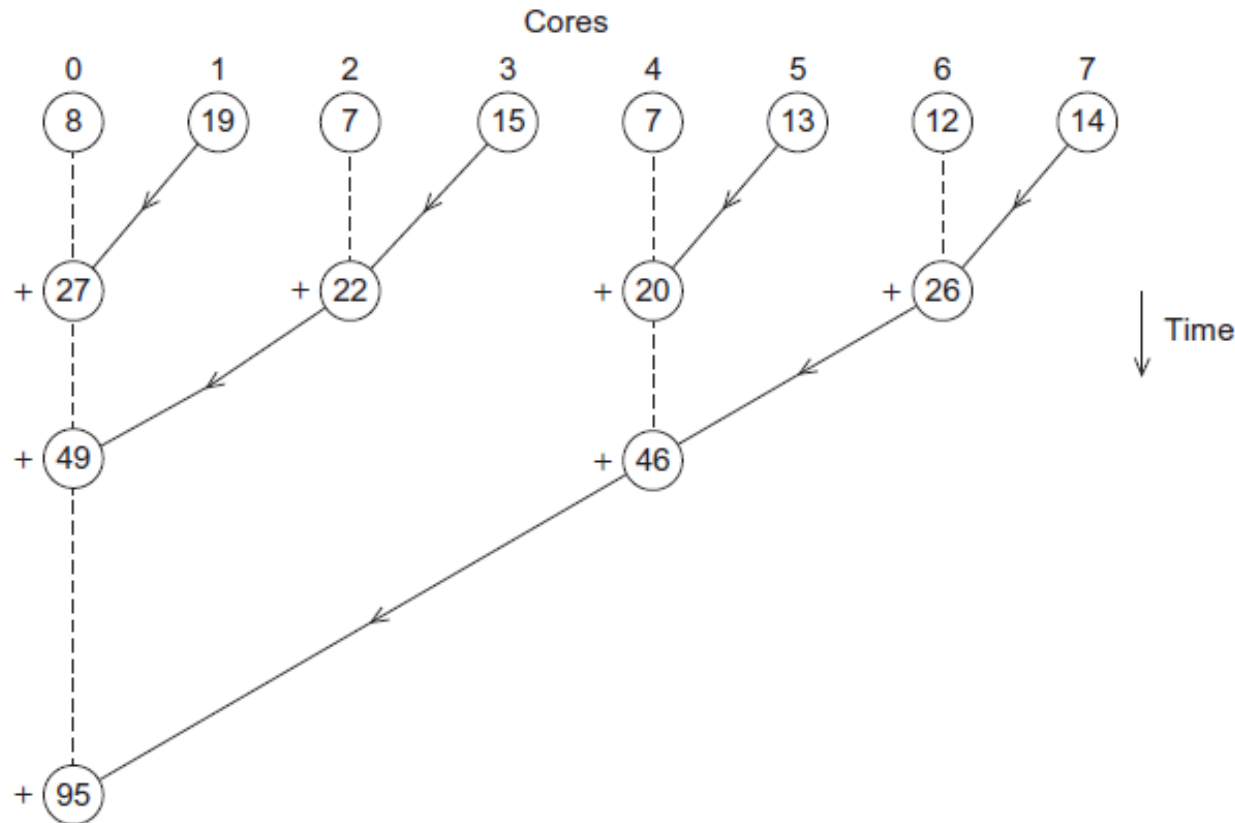
Core	0	1	2	3	4	5	6	7
my_sum	8	19	7	15	7	13	12	14

Core	0	1	2	3	4	5	6	7
my_sum	<b>95</b>	19	7	15	7	13	12	14

# Parallel Programming Concept for Multi-core Processors

But wait! There's a much better way to compute the global sum.



# Parallel Programming Concept for Multi-core Processors

---

## ► **Analysis**

- In the first example, the master core performs 7 receives and 7 additions.
- In the second example, the master core performs 3 receives and 3 additions.
- The improvement is more than a factor of 2!
- **Imagine, If we have 1000 cores:**
  - The first example would require the master to perform 999 receives and 999 additions.
  - The second example would only require 10 receives and 10 additions.
- That's an improvement of almost a factor of 100!

# Salient Features: Dual Core and Core 2 Duo

---

## ► **Architecture**

- *Dual Core*: The dual core is the older of the two having an architecture that sports two cores on one processor, but its older technology puts it in a position of disadvantage.
- *Core 2 Duo*: The core 2 duo has two cores on the same processor. However while the architecture sounds vaguely the same, it is more advanced than the dual core processors.

# Salient Features: Dual Core and Core 2 Duo

---

## ► **Performance**

- *Dual Core*: This is without a doubt one of the best Intel processors, however it lacks the bite in terms of performance. Acceptable enough clock speeds of about 2.33 GHz for the best ones.
- *Core 2 Duo*: This newer processor beats the dual core in all bench-marking tests and therefore performance-wise, is definitely the better pick. Slams the opposition clocking speeds of 3.33 GHz clock as seen of the higher end ones.

# Salient Features: Core i3

---

- ▶ Entry level processor
- ▶ 2-4 Cores
- ▶ 4 Threads

(a **thread** of execution is the smallest sequence of programmed instructions that can be managed independently by a scheduler, which is typically a part of the operating system. Multiple threads can exist within one process, executing concurrently and sharing resources such as memory, while different processes do not share these resources.)

- ▶ Hyper-Threading (efficient use of processor resources)
- ▶ 3-4 MB Cache
- ▶ 32 nm Silicon (less heat and energy)
- ▶ Core i3 processors do support 64-bit versions of Windows
- ▶ **Suitability:**
  - ▶ If you use your computer for basic tasks such as word processing, email, surfing the web, etc., a Core i3 processor is more than enough to handle all of that with ease

# Salient Features: Core i5

---

- ▶ Mid range processor
- ▶ 2-4 Cores
- ▶ 4 Threads
- ▶ Turbo Mode (turn off core if not used)
- ▶ Hyper-Threading (efficient use of processor resources)
- ▶ 3-8 MB Cache
- ▶ 32-45 nm Silicon (less heat and energy)
- ▶ Suitability:
  - ▶ Core i5's offer enough performance to do stuff like video editing and gaming, and more than enough performance to do basic stuff like-word processing, internet surfing, and email.



# Salient Features: Core i7

---

- ▶ High end processor
- ▶ 4-8 Cores
- ▶ 8 Threads
- ▶ Turbo Mode (turn off core if not used)
- ▶ Hyper-Threading (efficient use of processor resources)
- ▶ 4-8 MB Cache
- ▶ 32-45 nm Silicon (less heat and energy)
  
- ▶ Suitability:
  - ▶ Core i7's offer enough performance to do stuff like video editing and gaming.

# Salient Features: Core i9

(10<sup>th</sup> and upper Generation Processor)

---

- ▶ 8-16 cores
- ▶ Supports Hyper-Threading
- ▶ Upto 16 MB of Smart Cache
- ▶ 1400 Billions Transistor
- ▶ Turbo-mode compatible
  
- ▶ Suitability:
  - ▶ Core i9's offer enough performance to do stuff like video editing and gaming.