Cloud Computing Concepts

CS3132

Dr. Anand Kumar Mishra

NIIT University

ISA
Microarchitecture
Processor (CPU)

Details

ISA - Instruction Set Architecture

- The ISA specifies what the processor is capable of doing
- To command the computer, you need to speak its language and the instructions are the words of a computer's language and the instruction set is basically its vocabulary
  - Unless you know the vocabulary and you have a very good vocabulary, you cannot gain good benefits out of the machine

ISA - Instruction Set Architecture

- The instruction set architecture is the specification of **what the computer can do** and **the machine has to be fabricated** in such a way that it will execute whatever has been specified in your ISA
- The only way that you can talk to your machine is through the ISA
- ISA defines the set of instructions that the processor can execute, as well as the format of data that is passed between the processor and memory

ISA - Instruction Set Architecture

- The ISA specifies what the processor is capable of doing
- An Instruction Set Architecture (ISA) is part of the abstract model of a computer that defines how the CPU is controlled by the software
- The ISA acts as an interface between the hardware and the software, specifying both what the processor is capable of doing as well as how it gets done
- The ISA provides the only way through which a user is able to interact with the hardware

ISA - Instruction Set Architecture

- An ISA contains
  - the functional definition of storage locations (registers, memory) & operations (add, multiply, branch, load, store, etc)
  - precise description of how to invoke & access them
- ISA does not contain non-functional aspects
  - How operations are implemented
  - Which operations are fast and which are slow
  - Which operations take more power and which take less
- Instructions are bit-patterns. Hardware interprets as commands

ISA - Instruction Set Architecture

An ISA contains

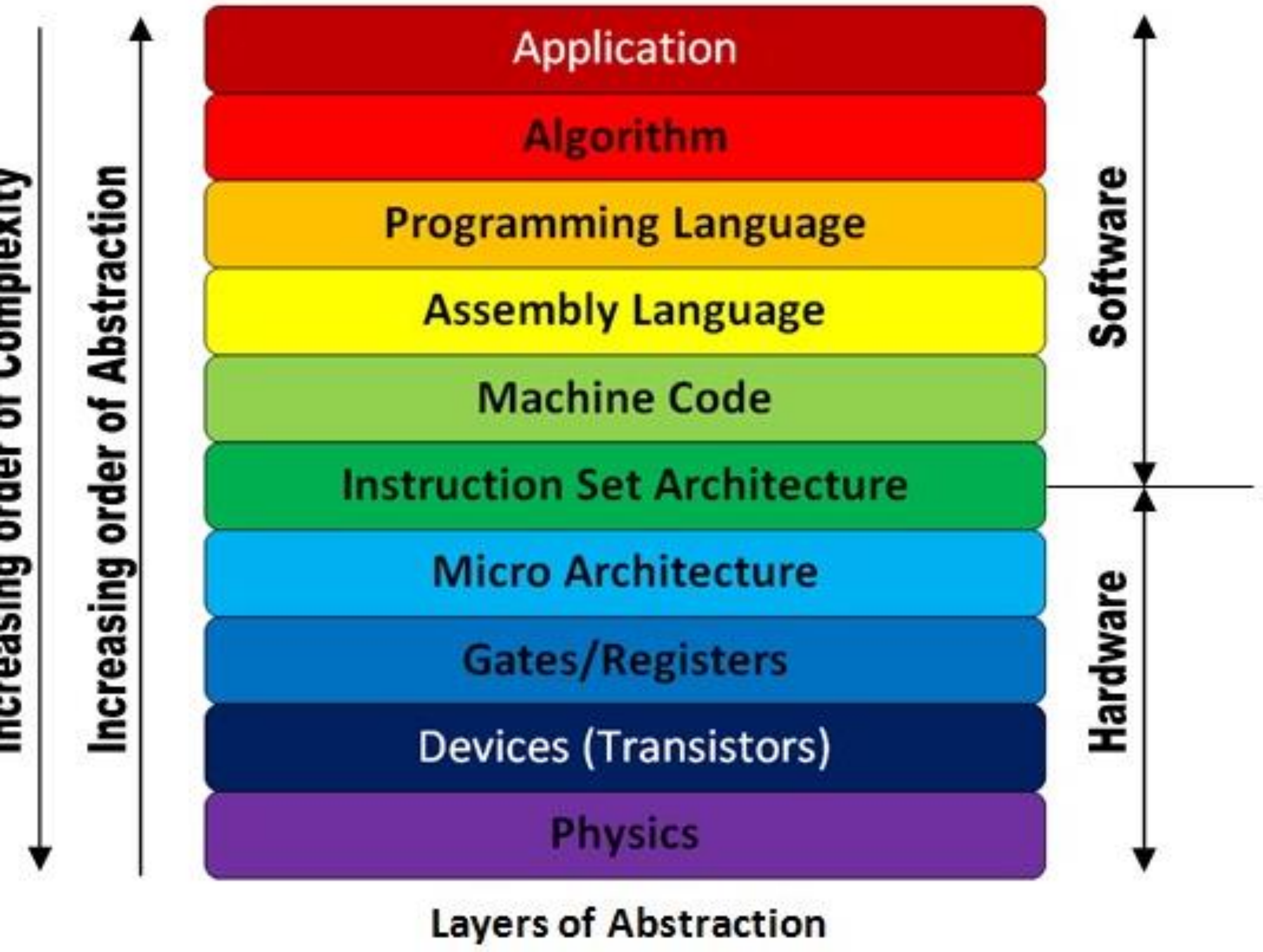
- the functional definition of storage locations (registers, memory) & operations (add, multiply, branch, load, store, etc)
- precise description of how to invoke & access them
- What instructions are available?
- What addressing modes are available?
- What is the format of data?
- How many and what kind of registers are available? {Example with x86 register}
- What condition codes, if any, are defined?
- How are exceptions handled?
- How are interrupts handled?

ISA does not contain non-functional aspects

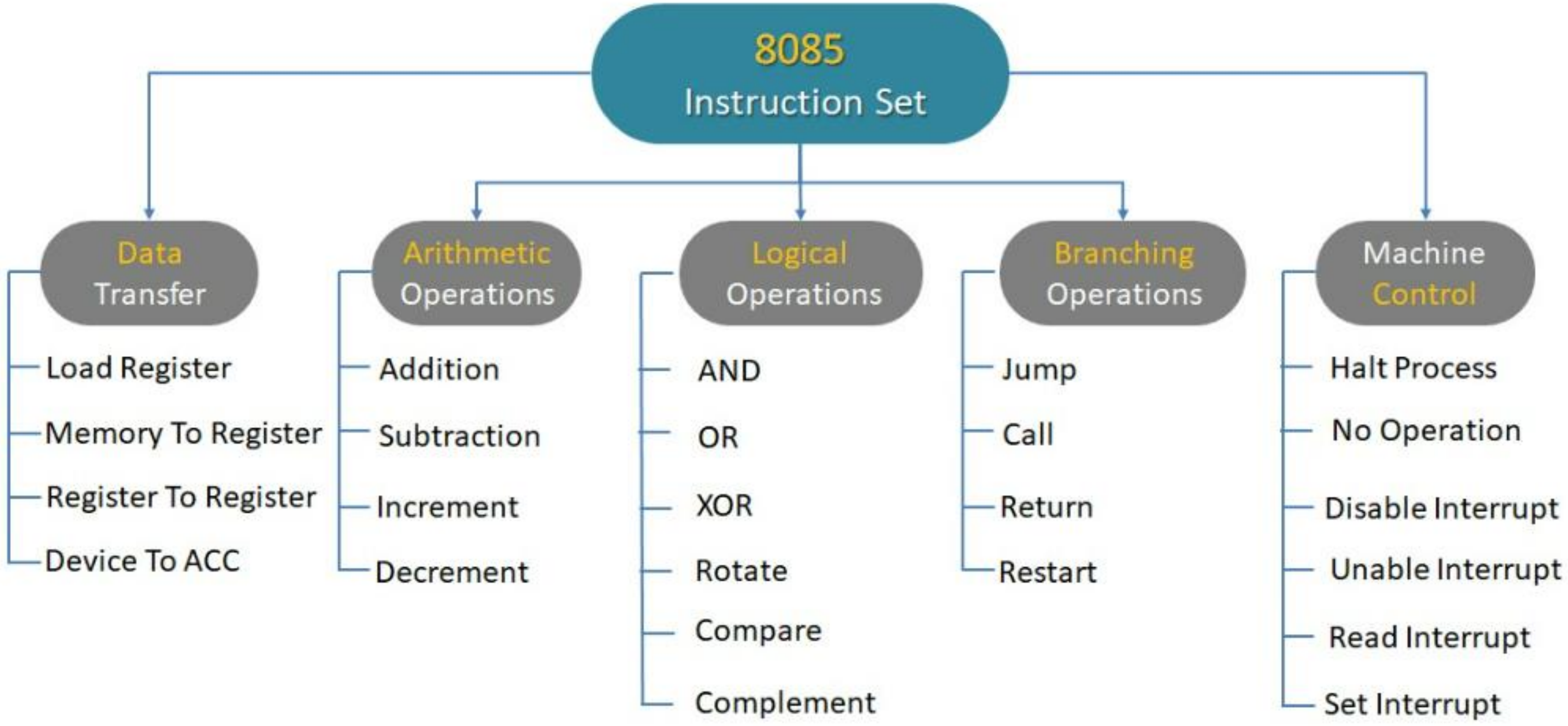
- How operations are implemented
- Which operations are fast and which are slow
- Which operations take more power and which take less
- What are procedure calling conventions?
- What are cache replacement policies?
- What happens on a page fault?

ISA - Instruction Set Architecture

- The ISA defines:
    - the supported data types,
    - the registers,
    - how the hardware manages main memory,
    - key features (such as virtual memory),
    - which instructions a microprocessor can execute, and
    - the input/output model of multiple ISA implementations
- The ISA can be extended by adding instructions or other capabilities, or by adding support for larger addresses and data values



Microarchitectural level lies just below the **ISA** level and hence is concerned with the implementation of the basic operations to be supported by the Computer as defined by the **ISA**


8085 Instruction Set
Data Transfer
Load Register
Memory To Register
Register To Register
Device To ACC
Arithmetic Operations
Addition
Subtraction
Increment
Decrement
Logical Operations
AND
OR
XOR
Rotate
Compare
Complement
Branching Operations
Jump
Call
Return
Restart
Machine Control
Halt Process
No Operation
Disable Interrupt
Unable Interrupt
Read Interrupt
Set Interrupt

Examples of ISAs include

- x86:
  - The x86 ISA is used by most personal computers and servers.
- ARM:
  - The ARM ISA is used in most mobile devices and embedded systems.
- RISC-V:
  - The RISC-V ISA is an open-source ISA that is gaining popularity in a variety of applications.

x86 Architecture

- The Intel x86 processor uses complex instruction set computer (CISC) architecture
- The X86 architecture is a computer language instruction set executed by a microprocessor

x86 Architecture - Registers

- The x86 architecture consists of the following unprivileged integer registers.

eax	Accumulator
ebx	Base register
ecx	Counter register
edx	Data register - can be used for I/O port access and arithmetic functions
esi	Source index register
edi	Destination index register
ebp	Base pointer register
esp	Stack pointer

X86 Architecture - Data Types

- byte: 8 bits
- word: 16 bits
- dword: 32 bits
- qword: 64 bits (includes floating-point doubles)
- tword: 80 bits (includes floating-point extended doubles)
- oword: 128 bits

x86 Architecture - Flags

- These are single-bit registers and have a variety of uses

Flag Code	Flag Name	Value	Flag Status	Description
of	Overflow Flag	0 1	**nvov**	No overflow - Overflow
df	Direction Flag	0 1	**updn**	Direction up - Direction down
if	Interrupt Flag	0 1	**diei**	Interrupts disabled - Interrupts enabled
sf	Sign Flag	0 1	**plng**	Positive (or zero) - Negative
zf	Zero Flag	0 1	**nzzr**	Nonzero - Zero
af	Auxiliary Carry Flag	0 1	**naac**	No auxiliary carry - Auxiliary carry
pf	Parity Flag	0 1	**pepo**	Parity odd - Parity even
cf	Carry Flag	0 1	**nccy**	No carry - Carry
tf	Trap Flag			If **tf** equals 1, the processor will raise a STATUS_SINGLE_STEP exception after the execution of one instruction. This flag is used by a debugger to implement single-step tracing. It should not be used by other applications.
iopl	I/O Privilege Level			I/O Privilege Level This is a two-bit integer, with values between zero and 3. It is used by the operating system to control access to hardware. It should not be used by applications.

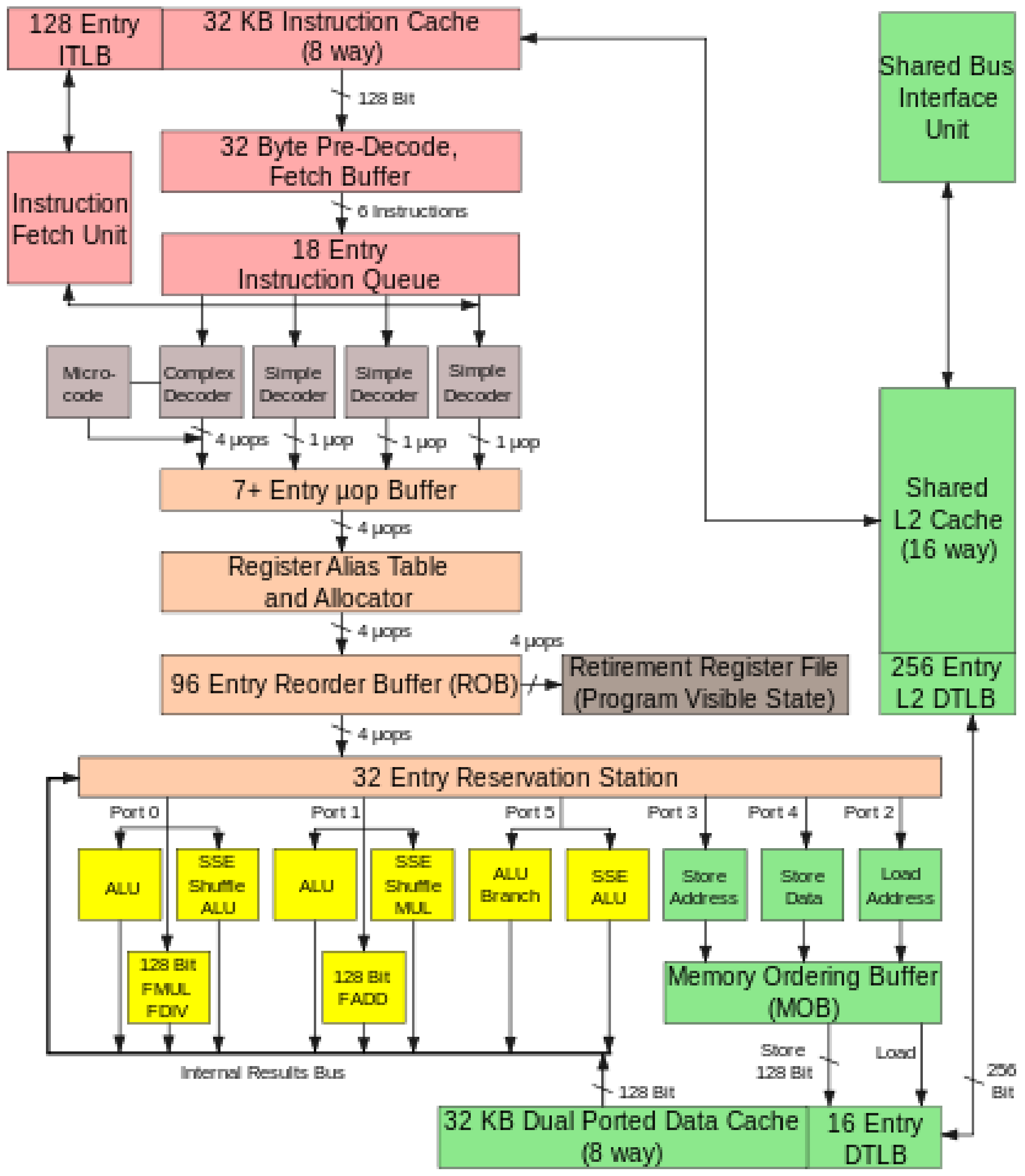
Microarchitecture

- A microarchitecture is a concrete (real) implementation of an ISA
- Micro-architecture includes things like:
  - Pipeline length and layout
  - Number and sizes of caches
  - Cycle counts for individual instructions
  - Which optional features are implemented
- The microarchitecture of a processor is the design of the processor at the lowest level
- It describes how the processor's components are interconnected and how they operate

Microarchitecture

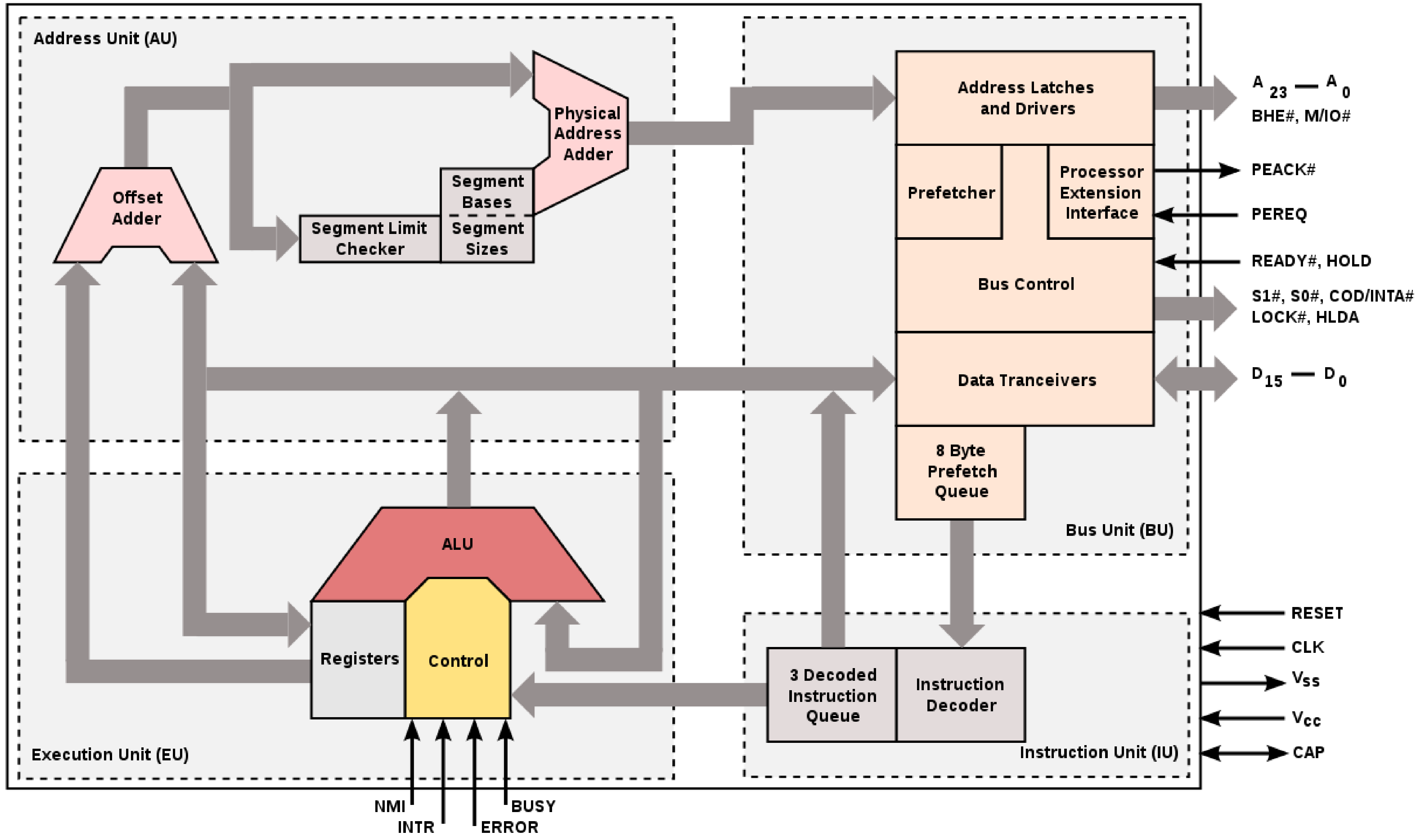
- The microarchitecture of a machine is usually represented as **diagrams**
  - that describe the interconnections of the various microarchitectural elements of the machine,
    - which may be anything from single gates and registers, to complete arithmetic logic units (ALUs) and even larger elements.

Diagram of the Intel Core 2 microarchitecture



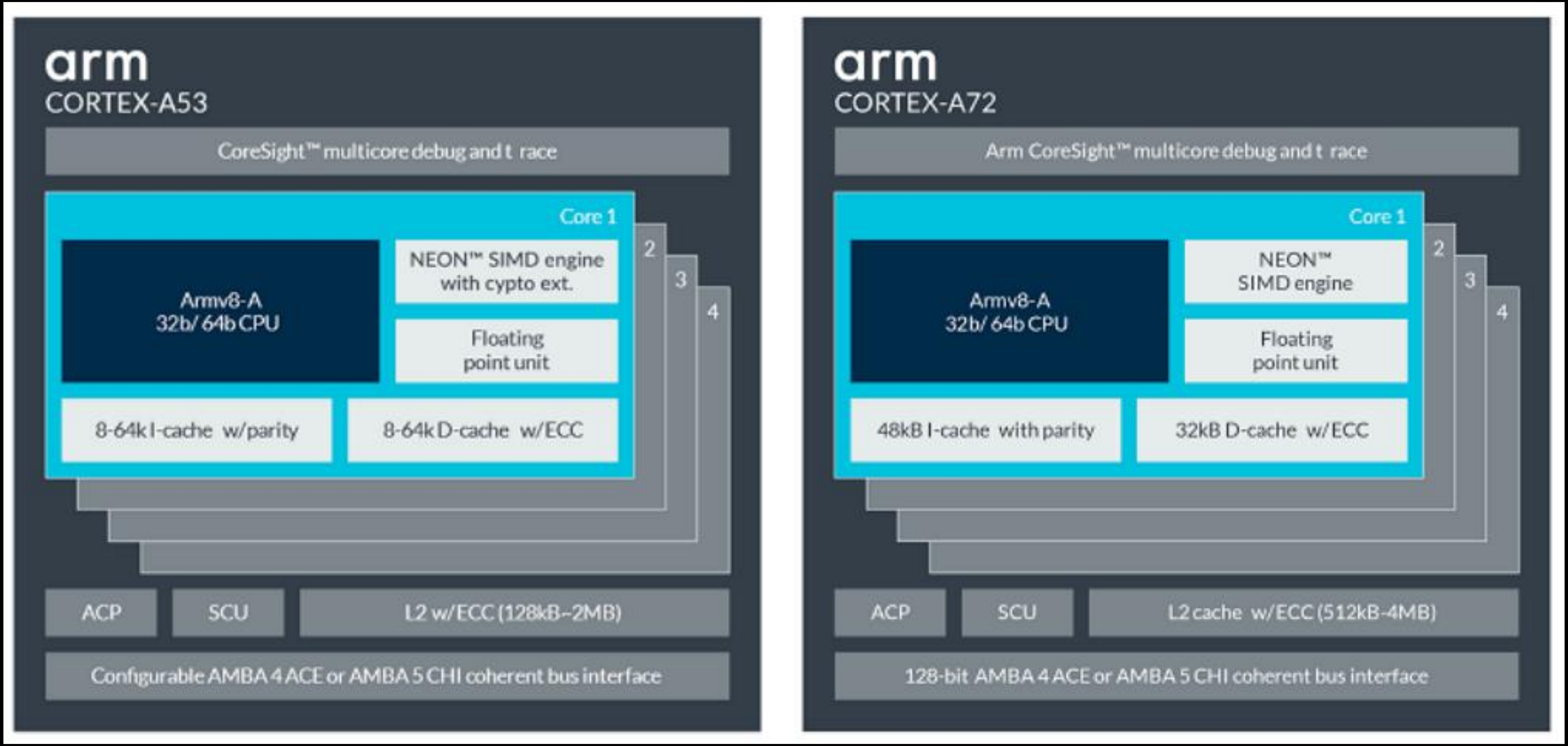
Intel Core 2 Architecture

Intel 80286 architecture



Microarchitecture

- The microarchitecture of a processor is implemented to implement a particular ISA
- Same ISA can be implemented with different microarchitectures
  - For example, the x86 ISA has been implemented by many different companies over the years, each with its own unique microarchitecture



Cortex-A53 and Cortex-A72 are both implementations of the Armv8-A architecture. This means that they have the same architecture, but they have very different micro-architectures

Architecture	Cortex-A53	Cortex-A72
Target	Optimized for power efficiency	Optimized for performance
Pipeline	8 stages In-order	15+ stages Out-of-order
Caches	L1 I cache: 8KB - 64KB L1 D cache: 8KB - 64KB L2 cache: optional, up to 2MB	L1 I cache: 48KB fixed L1 D cache: 32KB fixed L2 cache: mandatory, up to 2MB

Microarchitecture

- The components of a microarchitecture are the fundamental building blocks of a processor. They include:
  - Arithmetic logic unit (ALU):
    - The ALU performs arithmetic and logical operations on data.
  - Control unit:
    - The control unit reads instructions from memory and controls the execution of those instructions.
  - Register file:
    - The register file stores temporary data that is being used by the processor.
  - Cache:
    - The cache stores frequently accessed data to improve the performance of the processor.

Microarchitecture

- Other components of a microarchitecture may include:
- Branch prediction unit:
  - The branch prediction unit predicts which branch of a conditional statement will be taken. This can improve performance by allowing the processor to prefetch the instructions that will be executed next.
- Out-of-order execution unit:
  - The out-of-order execution unit allows the processor to execute instructions in any order, regardless of the order in which they appear in the program. This can improve performance for certain types of applications.

Processor

- known as a central processing unit (CPU)
- Processor is the physical implementation of a microarchitecture
    - It is the chip that is installed in a computer and that executes instructions
- It is responsible for executing instructions and processing data

Processor

- The processor **includes all of the components of the microarchitecture**, as well as other components such as:
  - **Memory controller**:
    - The memory controller manages the flow of data between the processor and memory
  - **Input/output (I/O) controller**:
    - The I/O controller manages the flow of data between the processor and peripheral devices such as disks, networks, and displays

Examples of Processors

- Intel Core i7
  - The Intel Core i7 is a high-performance processor that is used in desktop computers and servers.
- AMD Ryzen 7
  - The AMD Ryzen 7 is a high-performance processor that is used in desktop computers and servers
- Apple M1
  - The Apple M1 is a high-performance processor that is used in Apple Mac computers

https://cs.lmu.edu/~ray/notes/cpu/

Intel Processors

Architecture	**Microarchitecture**	**Processors**
IA-32	P6	Pentium Pro, Pentium II, Pentium |||
IA-32	Pentium M	Pentium M
IA-32	Enhanced Pentium M	Core Solo, Core Duo, Xeon LV
IA-32	NetBurst	Pentium 4, Pentium 4F
x86-64 or Intel 64	NetBurst	Pentium 4F, Pentium D, Pentium Extreme Edition, Xeon 5000, Xeon 7100
Intel 64	Intel Core	Pentium Dual Core, Core 2 Duo, Core 2 Quad, Core 2 Extreme
Intel 64	Nehalem, Sandy Bridge, Ivy Bridge	Core i3, Core i5, Core i7

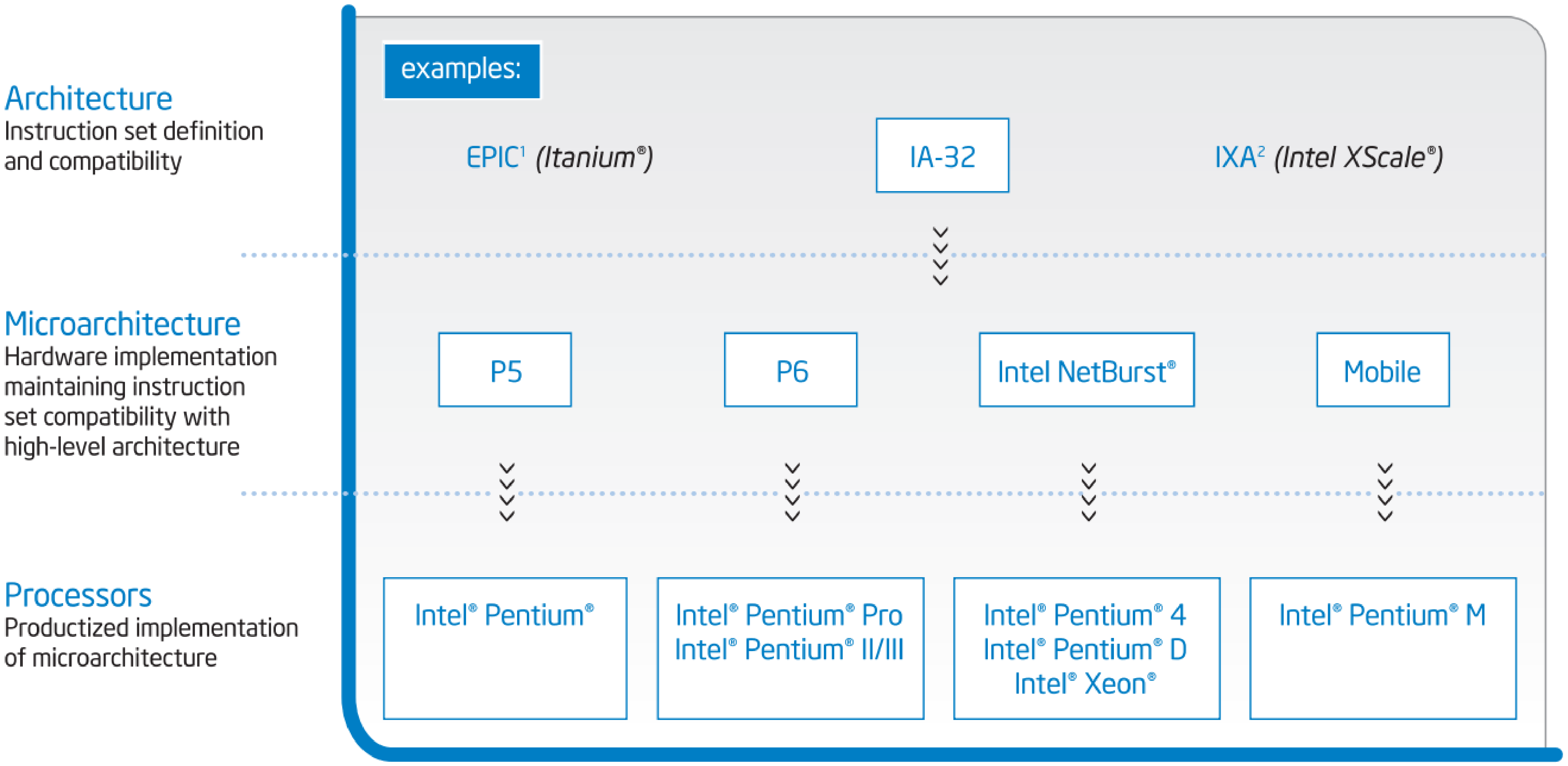
Difference between the microarchitecture and the processor

- Microarchitecture:
  - The microarchitecture of a processor defines how the processor's components are interconnected and how they operate
  - For example, the microarchitecture may define whether the processor has a pipeline, a cache, and a branch prediction unit.
- Processor:
  - The processor is the physical implementation of the microarchitecture
  - For example, the processor may be a quad-core processor with a 64-bit architecture and a 3 MB cache.

Relationship between ISA, microarchitecture, and processor

- The ISA defines the set of instructions that the processor can execute, and the microarchitecture is how the processor is designed to execute those instructions
- The processor is the physical implementation of the microarchitecture
- **ISA is the what**
- **Microarchitecture is the how**
  - **How the ISA is implemented**
- **Processor is the physical embodiment of the how**

Microarchitecture History



1. EPIC (Explicitly Parallel Instruction Computing)

2. IXA (Intel® Internet Exchange Architecture)

Relationship between ISA, microarchitecture, and processor

- ISA (instruction set architecture) or “architecture”
  - the set of instructions supported by a processor
- Microarchitecture concepts deal with how the ISA is implemented
  - concepts such as instruction pipelining, branch prediction, out of order execution are all employed to achieve an efficient realization of the ISA
- Example:
  - ARMv7 architecture refers to a particular ISA
  - On the other hand, the term **core** refers to an implementation of ISA using a microarchitecture
  - **ARM cortex A-5 core** is an implementation of ARMv7-A ISA using a particular microarchitecture

Relationship between ISA, microarchitecture, and processor

- Different processors may support same ISA, but may have different microarchitecture
- Example
  - Intel and AMD both implement x86 ISA using different microarchitectures
- x86 was a CISC architecture while ARM (Advanced RISC machines) was a RISC architecture