Abstraction in System Architecture

Lecture 42, v01

John G. Artus

BSEE

MSSE

INCOSE ESEP

Practical Example of Abstraction

- Conceptual abstractions may be formed by filtering the information content of a concept or an observable phenomenon, selecting only those aspects which are relevant for a particular purpose (www.wikipedia.org)
- Suppose you are planning a road trip from Los Angeles to New York
  - Which if these two maps would best serve your purpose of conceptualizing the path from origin to destination?

https://www.maptrove.ca/united-states/best/usa-interstate-highways-wall-map.html

https://highwayactof1956.weebly.com/map-of-the-us-interstate-highway-system.html

- The map on the left has just too much unnecessary detail
- The map on the right (an abstraction of the one on the left) has the right amount of detail to do the initial planning

In the initial planning of a solution, we step away from the details (abstraction) in order to better understand the bigger picture of the problem we are facing

Practical Example of Abstraction (continued)

- Suppose you were planning two overnight stays in Santa Fe, NM and Indianapolis, IN
- These more detailed maps would then be appropriate for that lower-level detail planning
  - These maps are more detailed (less abstract) than the larger US Interstate map

https://www.istockphoto.com/es/vector/nuevo-m%C3%A9xico-mapa-de-carretera-gm149364916-1561754

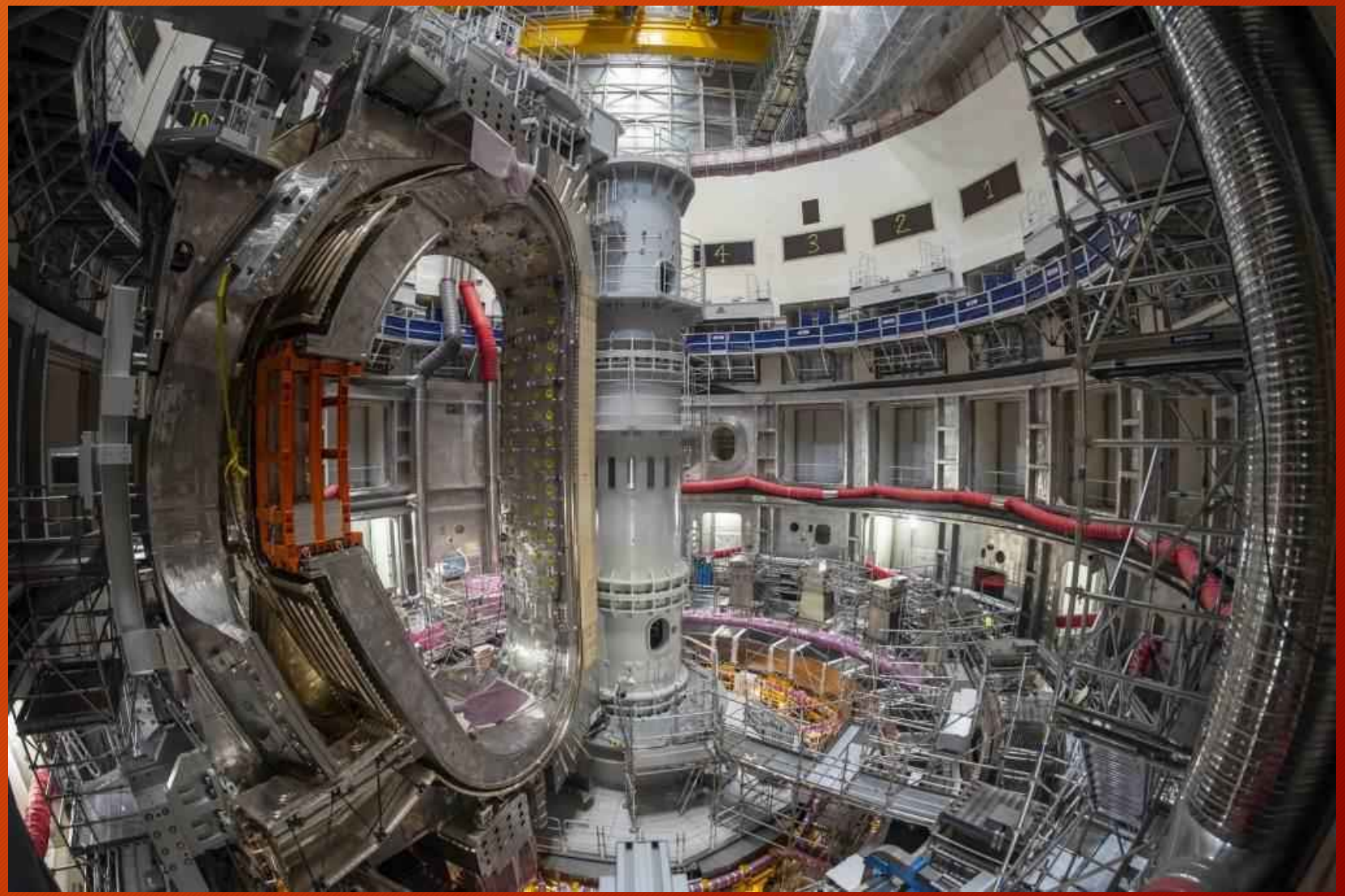
https://www.istockphoto.com/es/vector/mapa-de-carretera-indiana-gm164884358-1746841

As we dive deeper into the solution,
we need to begin adding detail (becoming less abstract)

Consider the ITER Fusion Reactor Being Built in France

- The design of the ITER reactor started well before any construction began
- It began with a concept for a reactor
- Considering all the machinery and software that goes into such a complex system, can you imagine how the concept initially developed
- Well, scientists and engineers didn't just start by making a long list of parts to go and buy – the project is massively complex
- To handle this design complexity, they had to ABSTRACT the concept into easily manageable parts

Heart of the ITER Facility: The Tokamak Reactor

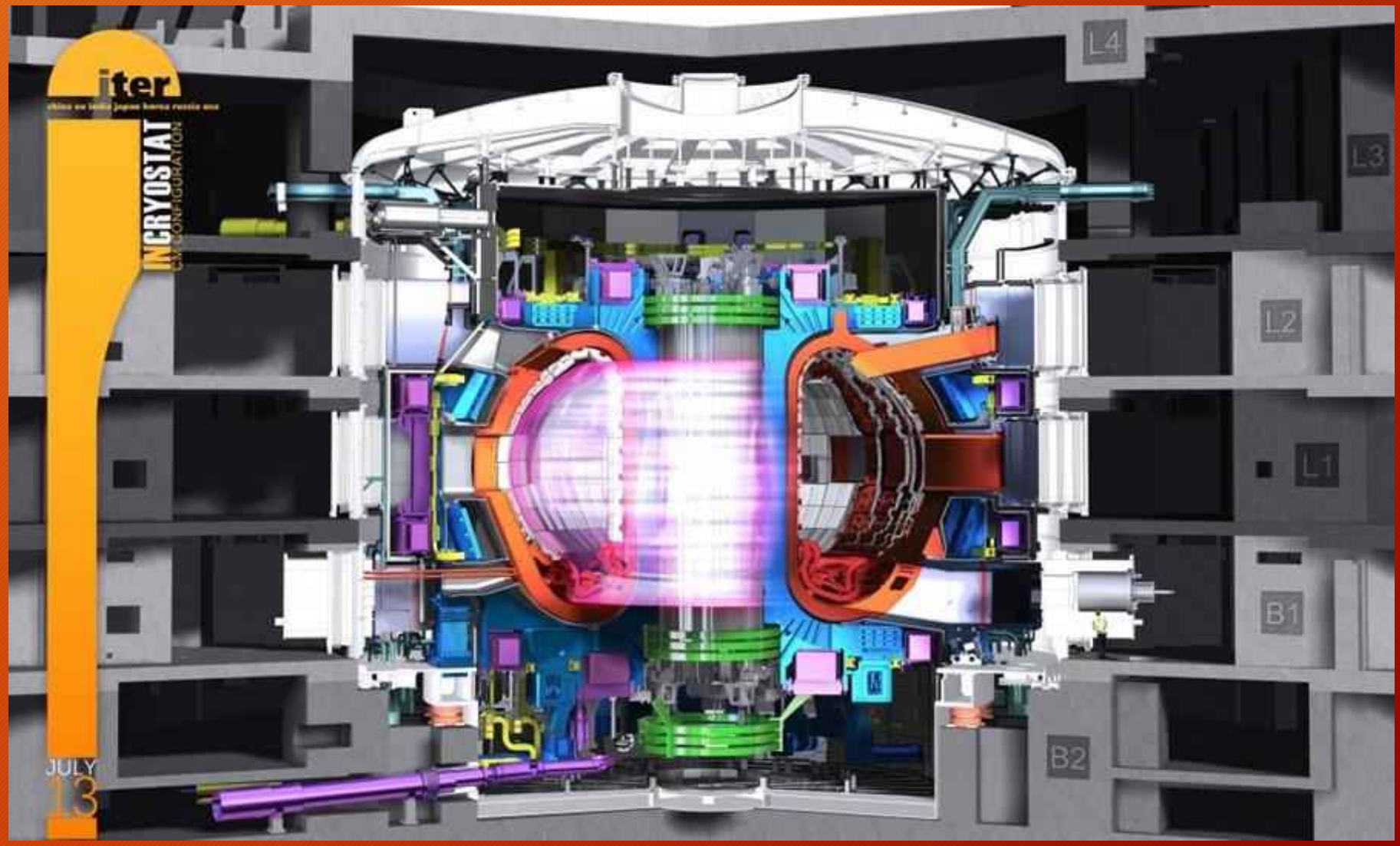


All ITER images from www.iter.org

The ITER Fusion Reactor is a Very Complex Undertaking

- The Tokamak Reactor is the heart of the ITER
- Over 23,000 Tons
- Over 10,000,000 parts (just the reactor)
- This is a very complex design requiring many design teams from around the world coordinating their contributions to this massive effort
- When beginning to conceptualize a solution (a design), these teams had to ABSTRACT away the details in order to develop the top-level concept for how the whole system is going to work

Concept of the ITER Tokamak Reactor in Operation

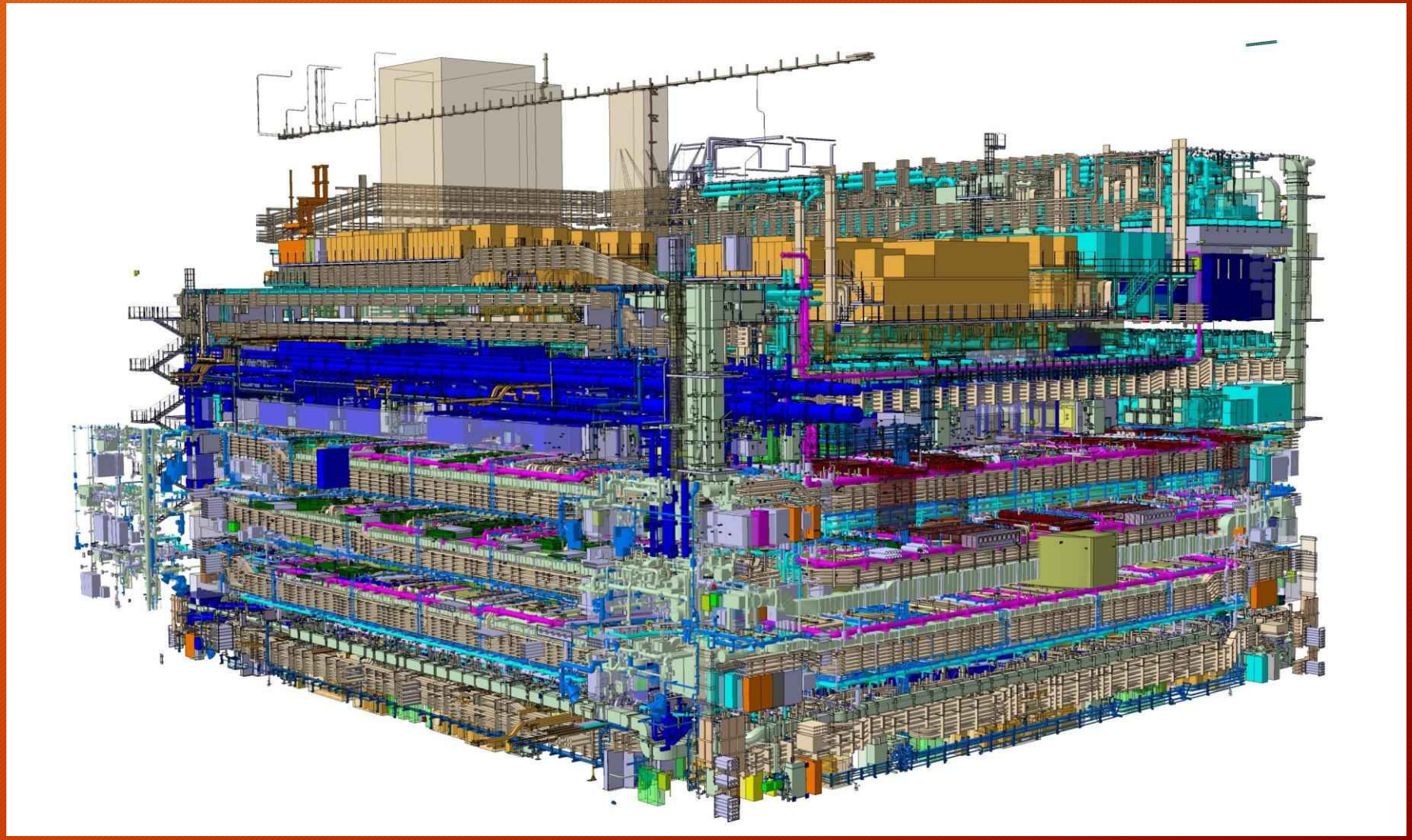


All ITER images from www.iter.org

A Project of this Level of Complexity Requires Abstraction

- In addition to the reactor itself, MAJOR supporting infrastructure has to be designed and built
- The total engineering effort on this project is absolutely staggering
- One can imagine that all this design effort does not happen all at once
- The entire system needs to be abstracted and decomposed in an organized way to ensure the entire project comes together in the end (is integrated) as smoothly as possible

Concept of the 7-Level ITER Tokamak Building



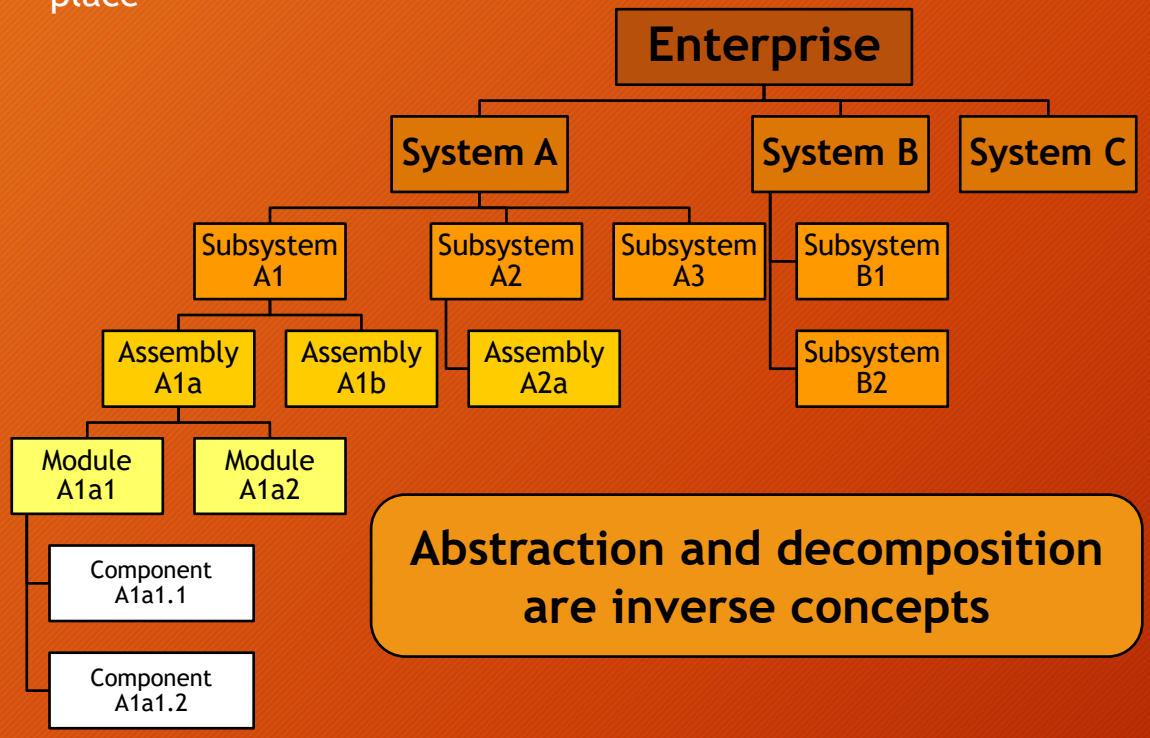
All ITER images from www.iter.org

Overview of Abstraction

- An abstraction is a model of the real system
  - These models connect the idea (concept) of a system to reality by removing sufficient detail so that a workable solution can be developed
- No matter the level of complexity, abstraction is a useful process to employ when conceptualizing a solution
- Abstraction is the process of ignoring certain details of a system concept in order to
  - Simplify the problem
  - To proceed in step-wise fashion among design levels of increasing system detail
  - This approach allows designers to only comprehend one level of complexity at a time
  - This helps facilitate the specification, design, and implementation of a system
- After gaining sufficient understanding at the higher levels of abstraction, we can then decompose the system into its subordinate elements at lower levels of the hierarchy tree
  - In doing so, we are adding detail (becoming less abstract)

Decomposition

- Most hard problems are tackled by taking a “Divide and Conquer” approach
  - Meaning, to divide the whole problem into smaller, more manageable problems
  - This is a common and practical approach taken in engineering in general, and in system architecture in particular
  - Decomposition in system architecture is accomplished by subdividing the system into smaller and smaller parts
- Decomposition naturally forms layers representing different levels of complexity
  - Each layer describes the same solution, but at increasing levels of detail
  - These layers are then mapped onto each other
  - In this way, high level abstractions are materialized by lower level abstractions until a simple realization (at the lowest level) can take place



Abstraction and decomposition are inverse concepts

More Abstract (Less Detail)

Less Abstract (More Detail)

Some things to note:

1. In this example, only one branch of the tree is populated down to low levels
2. Names used to describe the various levels are non-standard (everyone uses their own naming scheme)
3. Part IDs used in this example are also non-standard

Decomposition (continued)

- The goal of decomposition is to
  - Identify lower-level system elements that interact with each other in well-defind, and well-behaved ways
  - Lower-level child elements all support their parent's primary purpose (high cohesion)
- If this is achieved, then
  - Different teams can work on different system elements independently
  - Cross-communication among teams is minimized (low coupling)
  - High chance of successful design in which system elements work together harmoniously
- If decomposition is done in a counterproductive way, then problems could arise during integration of the elements into the whole system
  - This often occurs when jumping too far between levels in a hierarchy
  - Jumping from little system detail at higher levels of the higerarchy to too much detail in one jump
  - Individual team solutions for child element designs may not combine properly into the desired parent solution
- Abstraction is a way of performing decomposition by changing the level of detail to be considered
  - The goal is to properly distribute the jumps in detail in a balanced way at each successive lower level

How Decomposition Relates To Abstraction

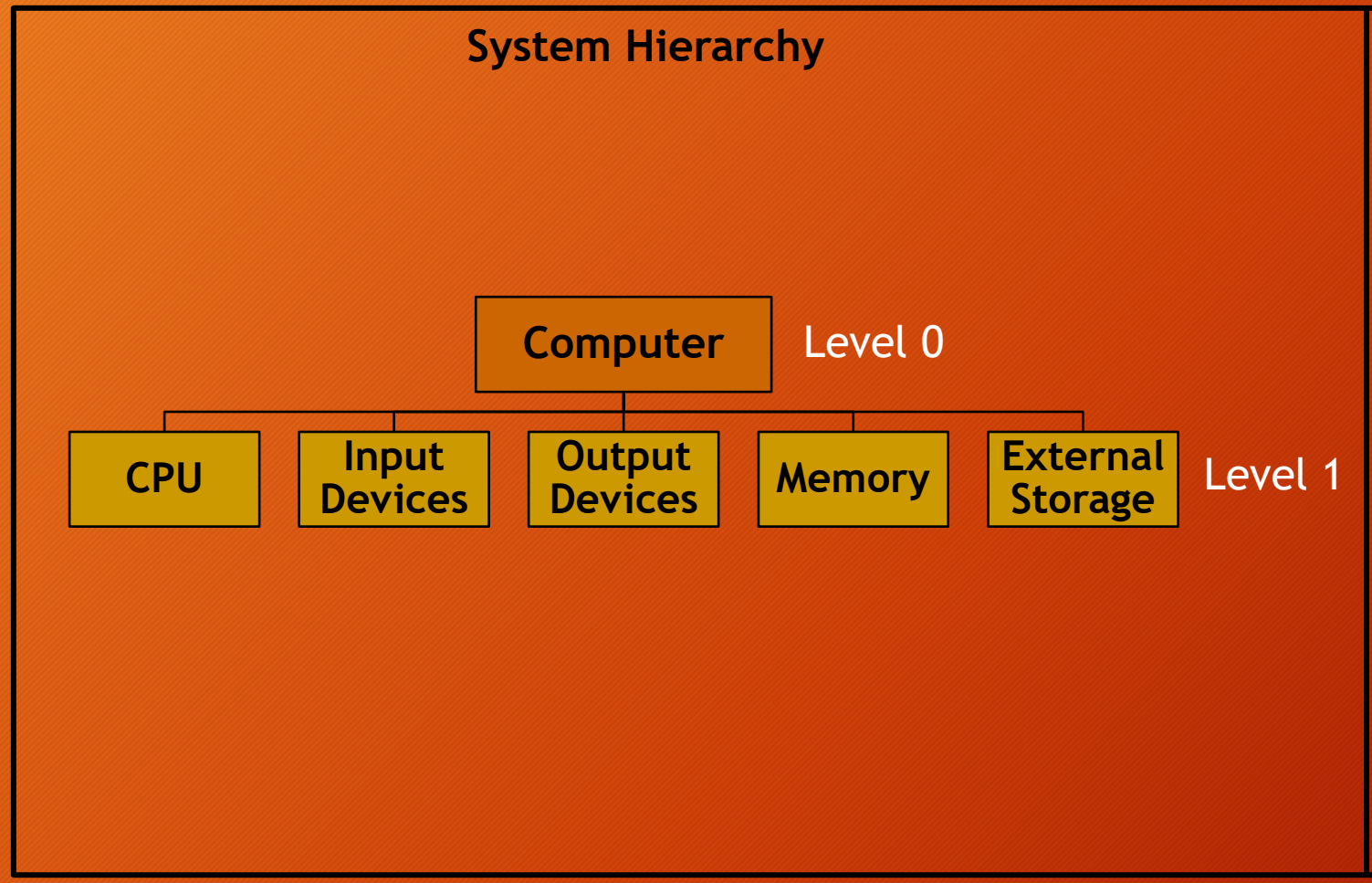
- The process of transforming one abstraction into a more detailed abstraction is called refinement
  - The new abstraction can be referred to as a refinement of the original one
  - Complex features of one abstraction are simplified into abstractions at lower levels of detail
  - Abstractions and their refinements typically do not coexist in the same system description
  - The system is normally described one level of detail at a time
  - Information management at multiple levels of detail is one of the goals of abstraction/decomposition
- Precisely what is meant by a more detailed abstraction is defined by the architect
  - A good architectural description will represent substitutability of concepts smoothly from one level of abstraction to another
- Decomposition occurs when one abstraction is split into smaller, more detailed abstractions
- Composition (integration) occurs when two abstractions are used to define another higher abstraction
  - Example: when multiple subsystem descriptions support (are consistent with) the description of the whole system
- Good abstractions can be very useful while bad abstractions can be very harmful
  - A good abstraction leads to reusable components
  - It simplifies interactions and localizes details and their operations into well defined units
  - Bad abstractions lead to customized components that are not reusable

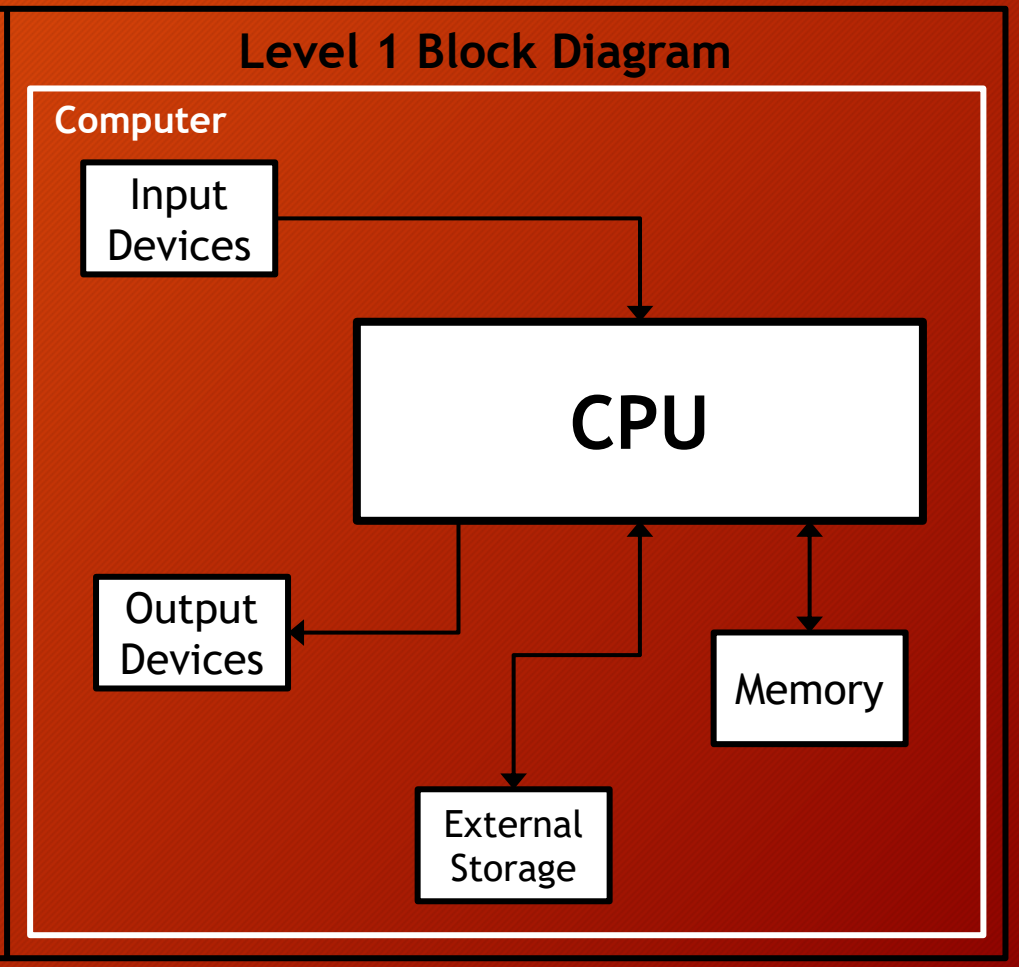
Decomposition Example: Computer

- In this example, we will decompose a simple computer design along one branch of decomposition
  - We start at Level 0, the highest level which addresses the system itself: the Computer
  - This is the highest level of abstraction of the concept of a Computer as is possible
  - All detail is removed

Decomposition Example: Computer

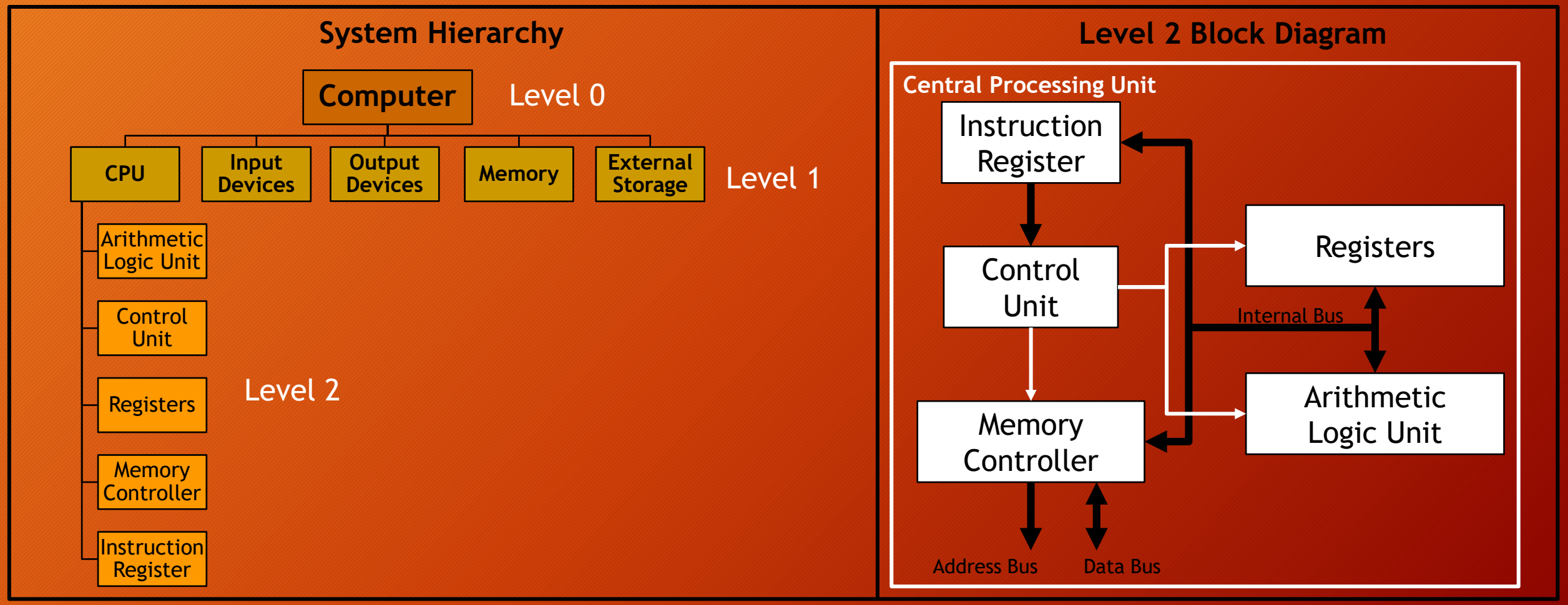
- At Level 1, we decompose the Computer into its basic components
  - It is important to select the appropriate level of decomposition
  - Getting too detailed too early can lead to complications when the system is integrated





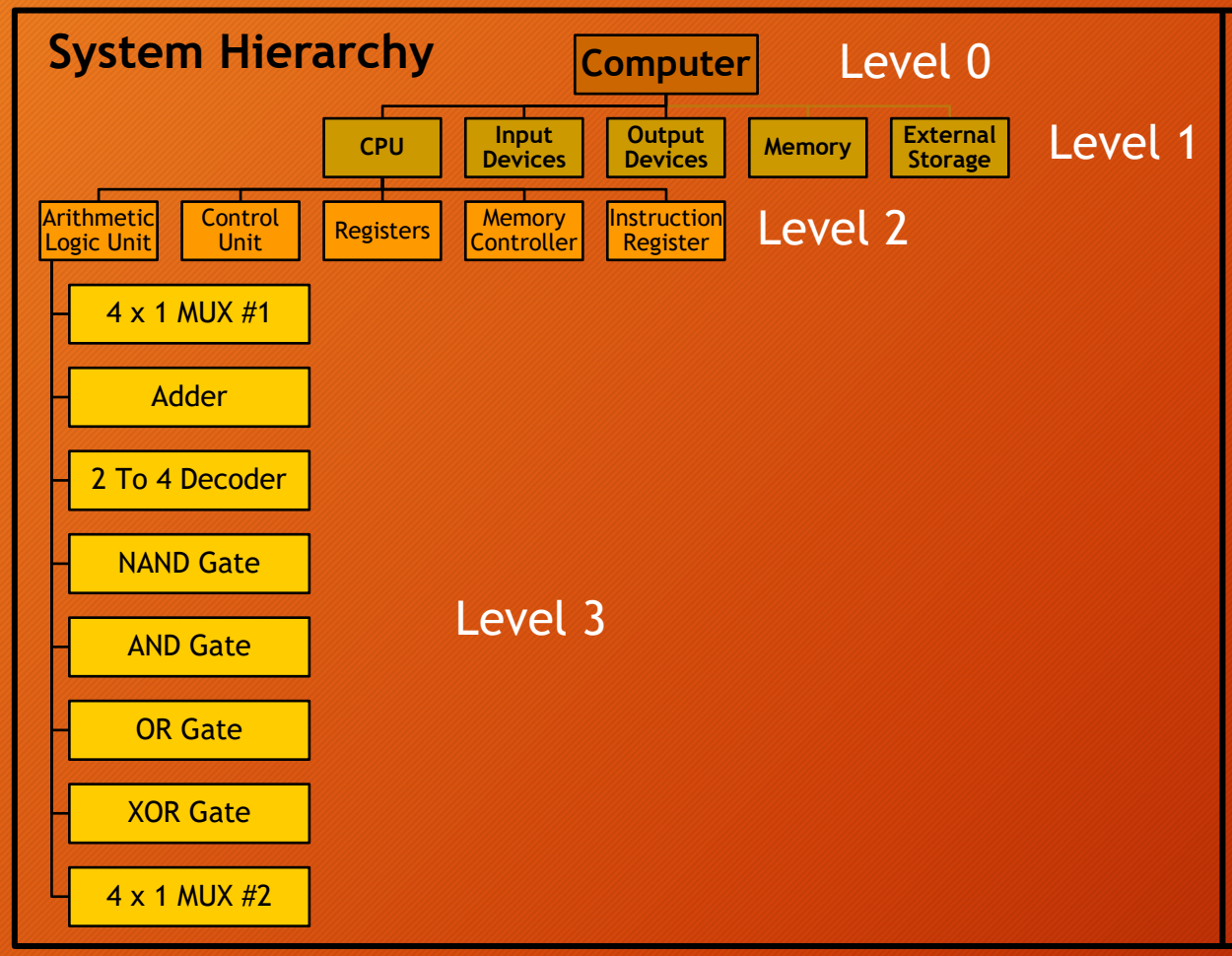
Decomposition Example: Computer

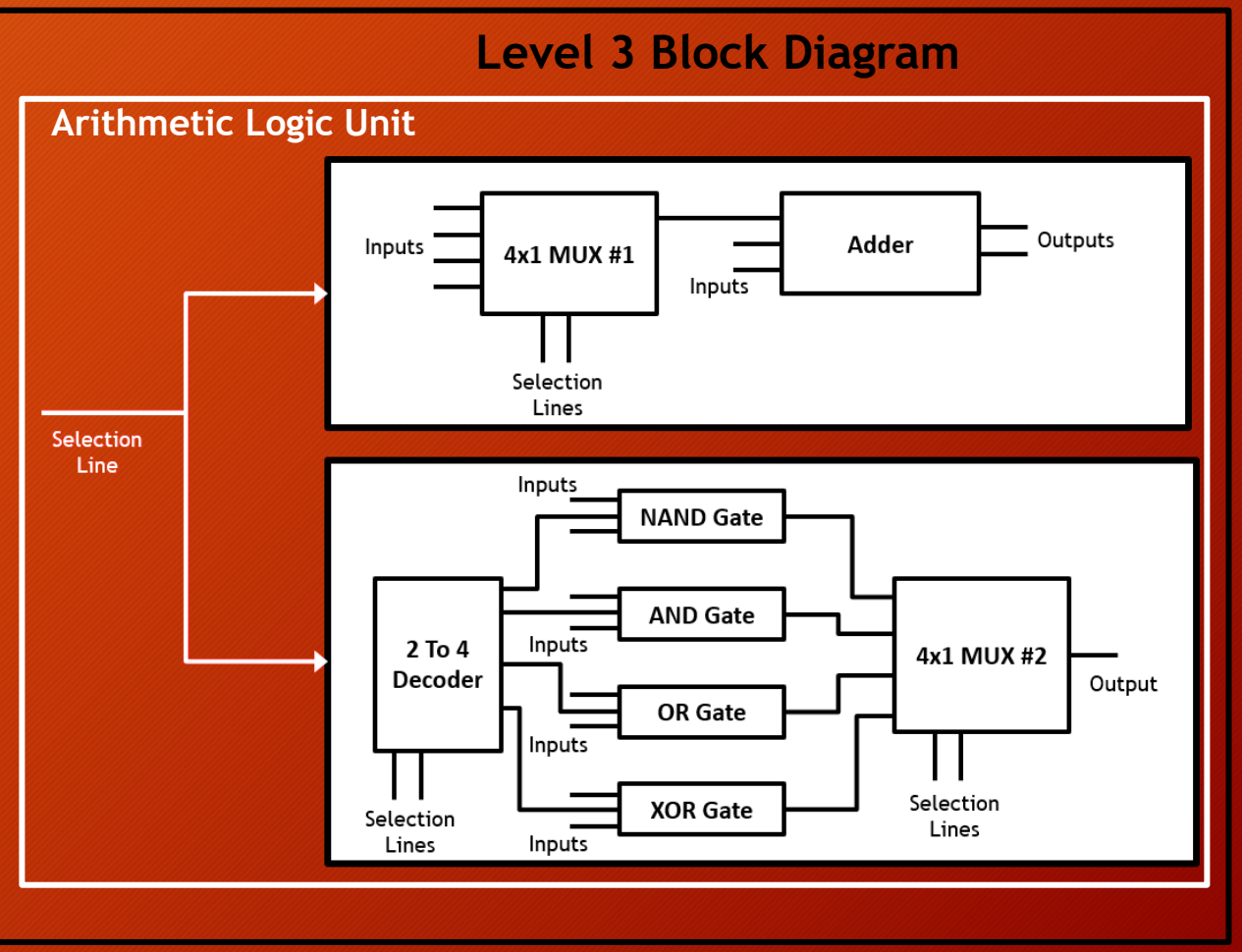
- In this example, further decomposition focuses on one major component of the Computer: The CPU
  - Here the CPU is broken into its major elements



Decomposition Example: Computer

- From here, the ALU is further decomposed into its major element

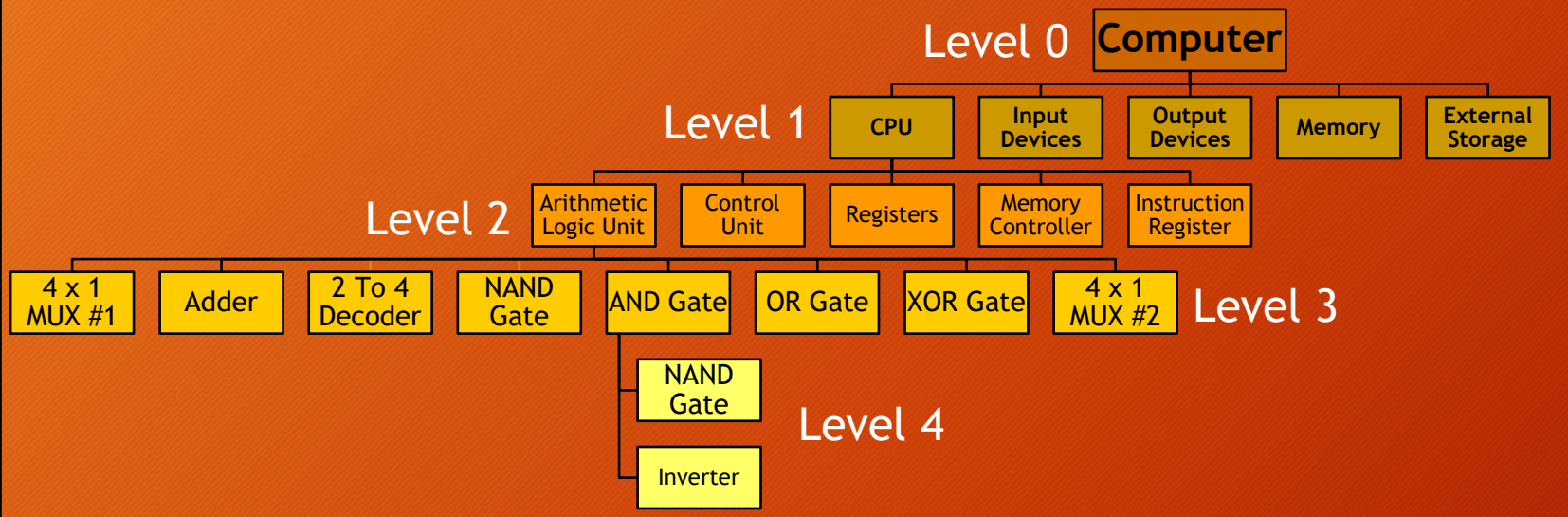




Decomposition Example: Computer

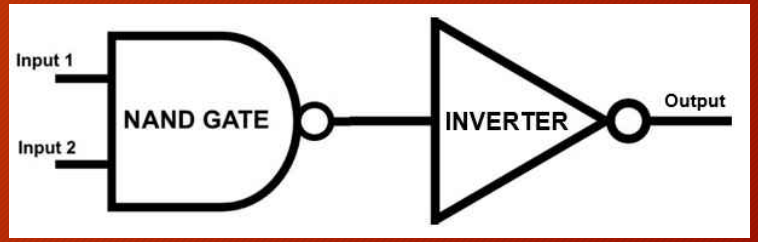
- From here, the AND Gate is further decomposed into its major element

System Hierarchy



Level 4 Block Diagram

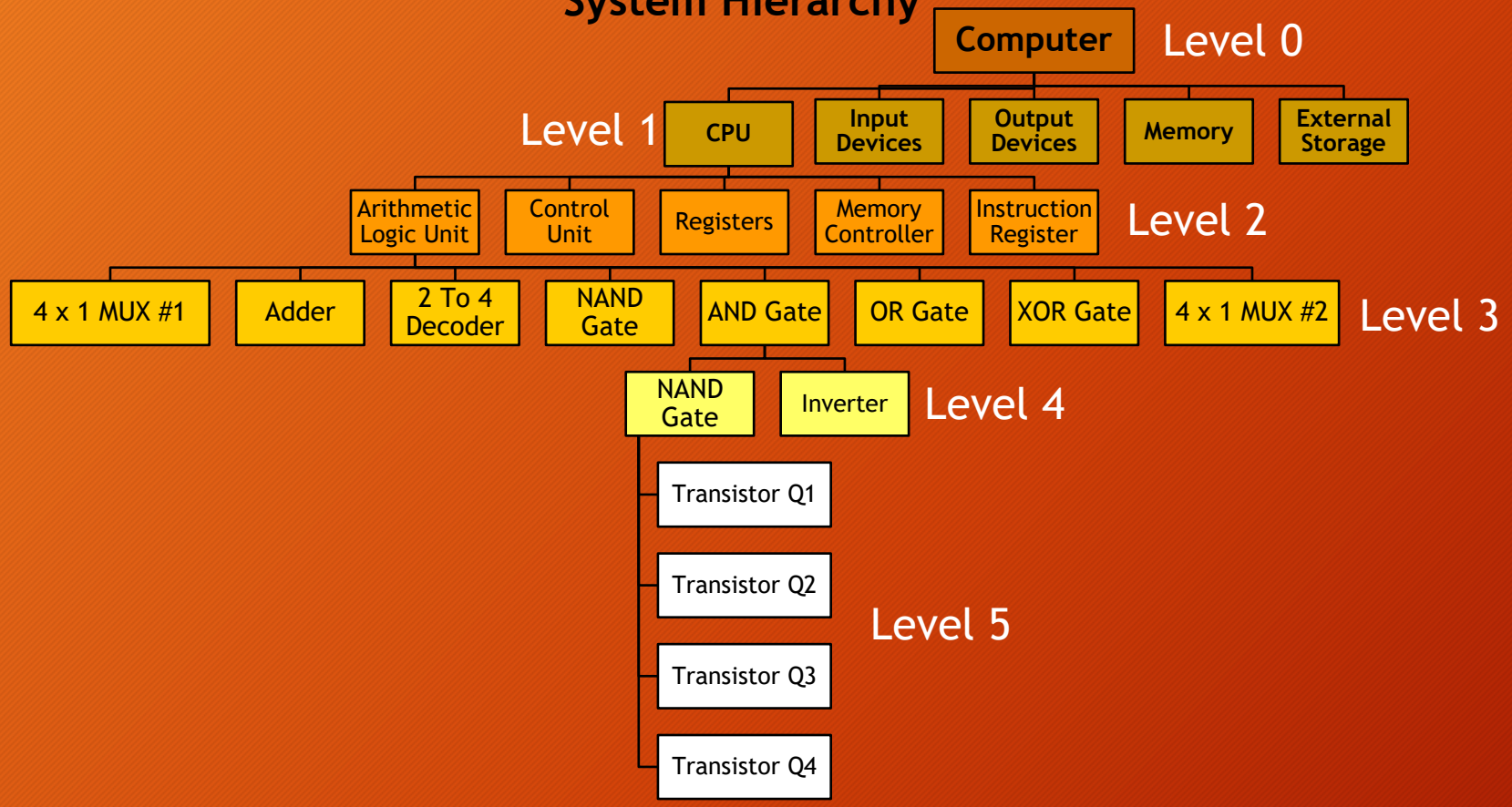
AND Gate



Decomposition Example: Computer

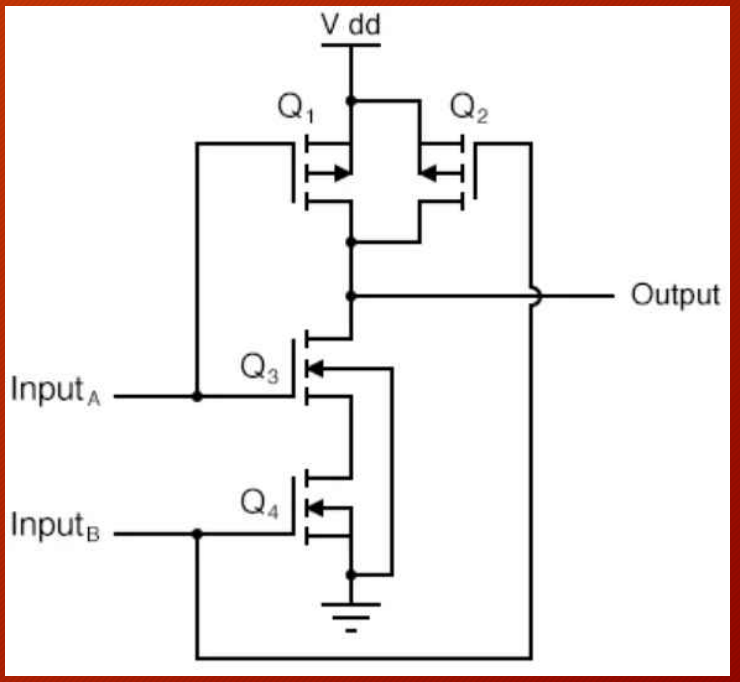
- Finally, the NAND Gate is decomposed into its basic elements: transistors
  - From this point, it is determined that no gain is to be made by further decomposition

System Hierarchy



Level 5 Block Diagram

CMOS NAND Gate



Overall Summary

- Abstraction and Decomposition are related, inverse concepts commonly used in the architectural design of complex systems
- Understanding their importance and use is critical to ensuring a productive representation of the system during the design phase
- The benefits (or alternatively, the costs) of the proper or improper use of abstraction/decomposition during the design phase will become apparent during system integration

References

- Burback, R, of Stanford University. (1998), *Abstraction*, http://infolab.stanford.edu/~burback/watersluice/node147.html
- Liskov, B. and Guttag, J. (1986), *Abstraction and Specification in Program Development*, The MIT Press, Cambridge, Massachusettes
- Pothier, J. (2014), *Abstract Thinking*, https://sites.tufts.edu/eeseniordesignhandbook/2014/abstract-thinking/
- Wikipedia Article (2022), *Abstraction*, https://en.wikipedia.org/wiki/Abstraction