

# Digital Design Methodology



## A Complex Information System Implementation

- Software based approach?
- Hardware based approach?
- Embedded system approach?
- Mixed design approach?



## Design Requirements & Specification

- Most concept projects include a body of information that describes the product or output of the project's work effort
- This information deals with the objectives of the final product, defined in the project **requirements**
- Any rules for creating the product, defined in the project **specifications**.

by Abramov B.

3

## Specification & Implementation

- Specification is related to **what** the system does.
  - ✓ The specification of a system refers to a description of its functionality and of other characteristics (**constraints**) required for its use.
- Implementation is related to **how** the system performs the operation.
  - ✓ An implementation of a system refers to how the system is constructed from simpler components. In the case of digital devices, the implementation is a digital network that consists of the interconnection of digital modules. This network can be defined at several levels depending on the complexity of the primitive modules used.

by Abramov B.

4

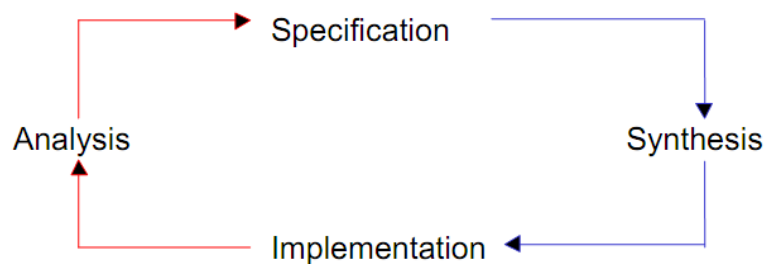
## Design Constraints

- Physical constraints: size, weight etc.
- Functional constraints: performance, throughput, power consumption, etc.
- Environment constraints: humidity, temperature etc.
- Resource constraints: cost, time to market etc.

by Abramov B.

5

## Design Process

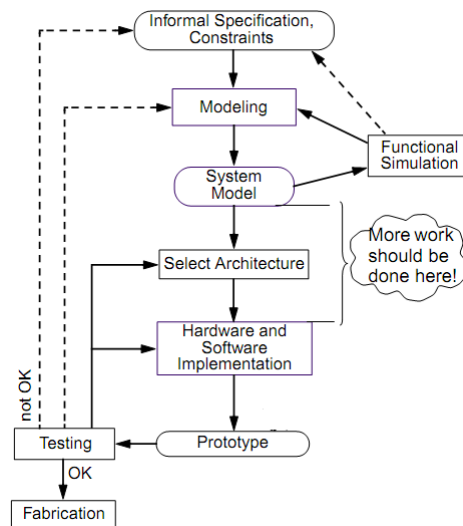


- The analysis of a system has an objective the determination of its specification from an implementation.
- The synthesis consists of obtaining an implementation that satisfies the specification of a system

by Abramov B.

6

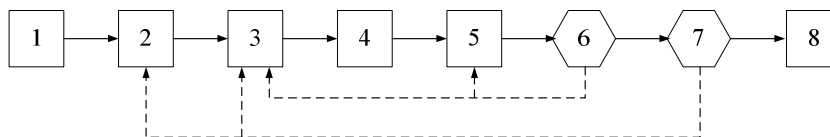
## Typical Design Flow



by Abramov B.

7

## Modeling



1. Task definition
2. Characterization of object that to be modeled
3. Object characteristics reduction
4. Decision on the model representation
5. Formalization
6. The model substantiation
7. Match analysis of the modeled object and modeling purposes
8. Data collection from model behavior

by Abramov B.

8

## Two main Paradigm

- Procedural paradigm (aka programming paradigm)
  - ✓ Flow charts
  - ✓ BDT
  - ✓ BDD,
  - ✓ ASM
  - ✓ Split ASM
- Declarative paradigm (aka structural paradigm)
  - ✓ Logic equations
  - ✓ STG
  - ✓ FSM



by Abramov B.

9

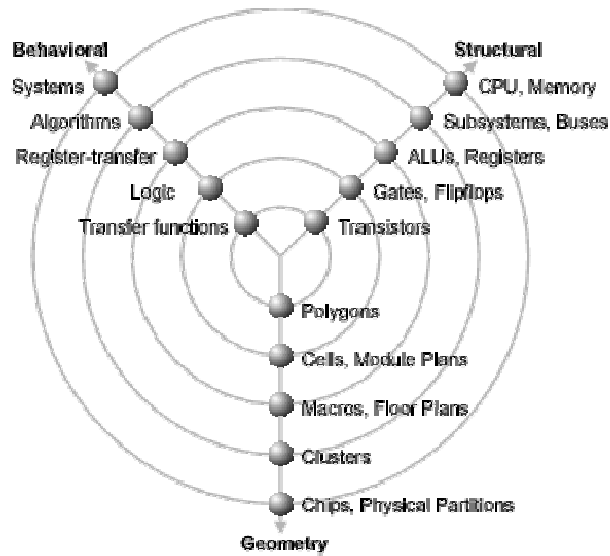
## Structural vs. Algorithmic?

- High-Level System Diagram
  - ✓ Context of the design
    - Inputs and Outputs
    - Throughput/rates
    - Algorithmic requirements
- Algorithm Description
  - ✓ Mathematical Description
  - ✓ Performance Criteria
    - Accuracy
    - Optimization constraints
  - ✓ Implementation constraints
    - Area
    - Speed

by Abramov B.

10

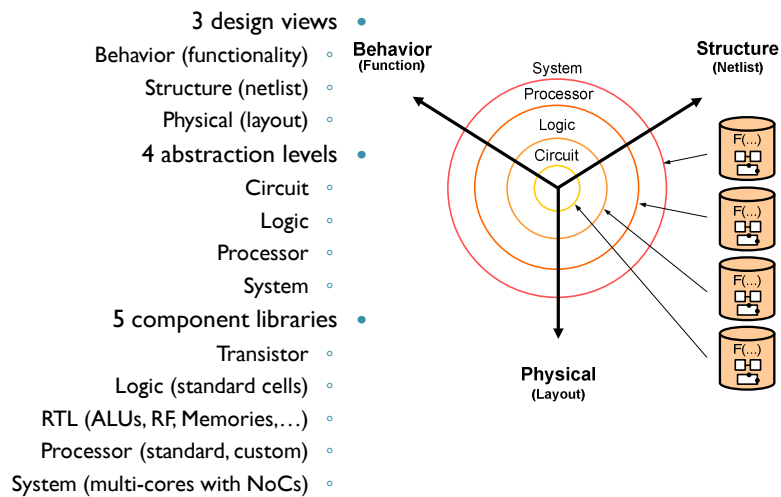
# Gajski-Kuhn (Y) chart



by Abramov B.

11

# Y chart



by Abramov B.

12

## Gajski-Kuhn three domains

- Behavior- describes the temporal and functional behavior of a system.
- Structure - A system is assembled from subsystems. Here the different subsystems and their interconnection to each other is contemplated for each level of abstraction.
- Geometry (Physical) - information about the size, the shape and the physical placement.

by Abramov B.

13

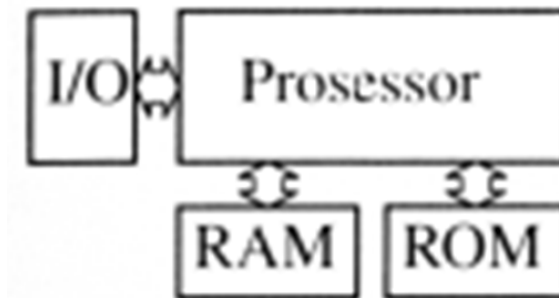
## Gajski-Kuhn five levels

- Architectural - A system's requirements and its basic concepts for meeting the requirements are specified here.
- Algorithmic - The "how" aspect of a solution is refined. Functional descriptions about how the different subsystems interact, etc. are included.
- Functional (register-transfer) - Detailed descriptions of what is going on, from what register over which line to where a data is transferred, is the contents of this level.
- Logic - The single logic cell is in the focus here, but not limited to AND, OR gates, also Flip-Flops and the interconnections are specified.
- Circuit - The actual hardware level. The transistor with its electric characteristics is used to describe the system. Information from this level printed on silicon results in the chip.

by Abramov B.

14

## Abstractions Levels: System



by Abramov B.

15

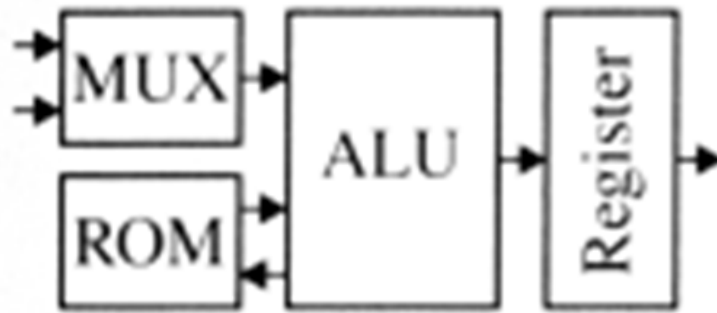
## Abstractions Levels: Algorithmic

```
A := A * B + F
  IF (C = TRUE) THEN
    A := A + 2 * F
  ELSE A := A - 1
ENDIF
```

by Abramov B.

16

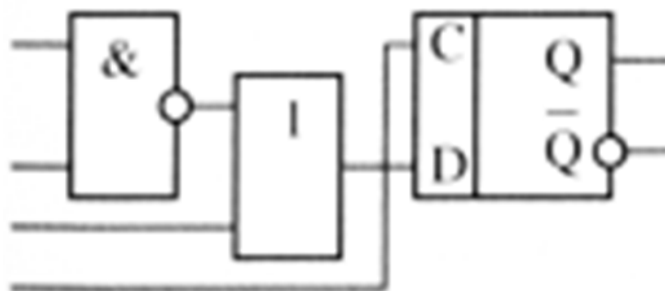
## Abstractions Levels: Register-Transfer



by Abramov B.

17

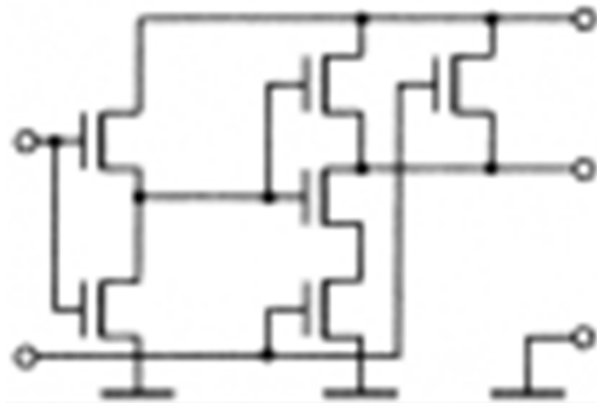
## Abstractions Levels: Logic



by Abramov B.

18

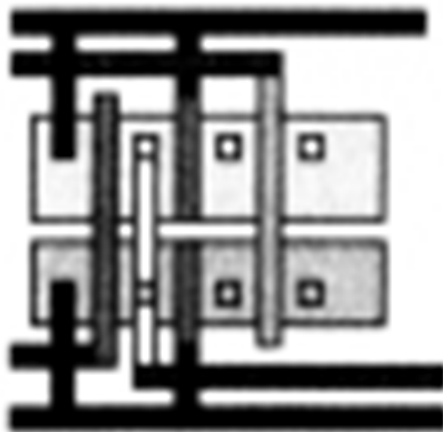
## Abstractions Levels: Transistor



by Abramov B.

19

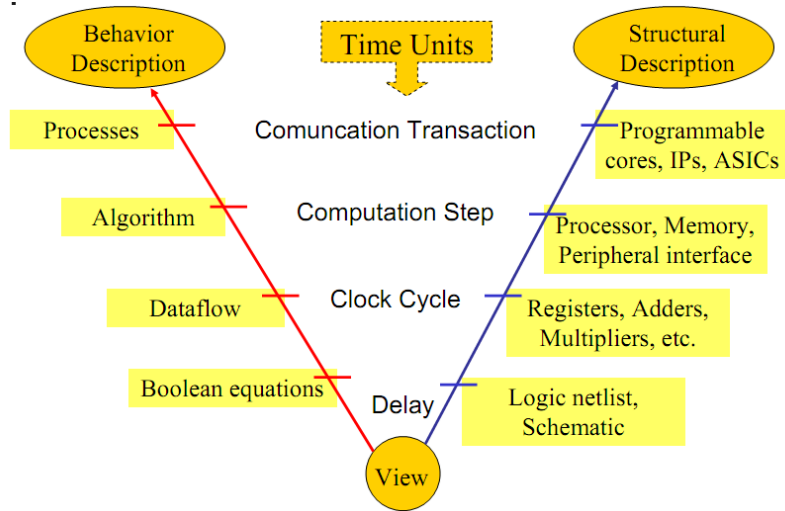
## Abstractions Levels: Geometry



by Abramov B.

20

## Timing Units at Different Levels



by Abramov B.

21

## Design Partition "Divide et Impera"

- Top Down Design Strategy
  - ✓ Big Pictures (since Plato)
- Bottom Up Design Strategy
  - ✓ Small Steps (since Aristotle)

by Abramov B.

22

## Design Partition (Divide et Impera)

- What kinds of objects are there?
- What sorts of properties can the objects have?
- What sorts of relationships can the objects have?
- What sorts of events or processes can occur involving those objects?
- What sorts of problems can arise involving those objects?
- What algorithms, or procedures are required to solve those problems?

by Abramov B.

23

## Top Down Design Strategy

- Complex problems can be solved using **top-down design strategy** :
  - ✓ We break the problem into parts
  - ✓ Then break the parts into parts
  - ✓ Each of the parts will be easy to do

by Abramov B.

24

## Top Down Design Strategy

- Breaking the problem into parts helps us to clarify what needs to be done.
- At each step of refinement, the new parts become less complicated and, therefore, easier to figure out.
- Parts of the solution may turn out to be reusable.
- Breaking the problem into parts allows more than one person to work on the solution.