# EE382M.20: System-on-Chip (SoC) Design

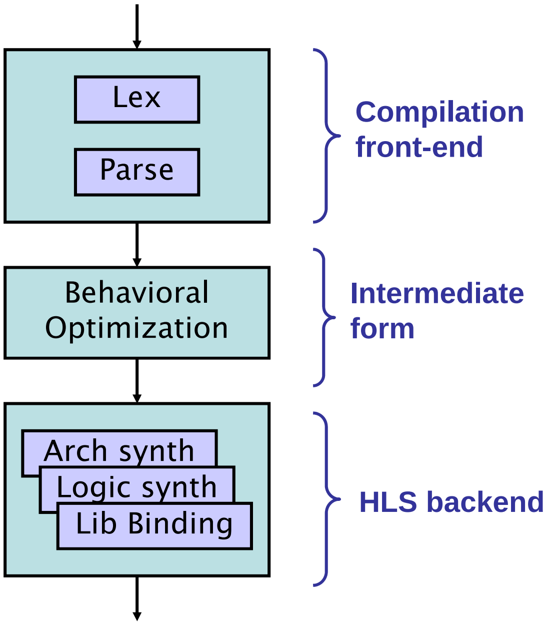
## Lecture 11 – High-Level Synthesis Algorithms

*Sources:*
*Rajesh Gupta, UC San Diego*

Andreas Gerstlauer
Electrical and Computer Engineering
University of Texas at Austin
gerstl@ece.utexas.edu

The University of Texas at Austin
Electrical and Computer Engineering
*Cockrell School of Engineering*

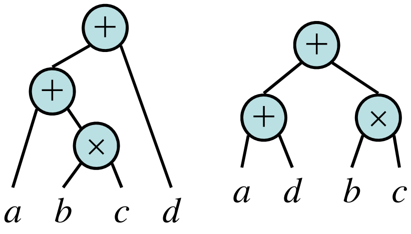
## High-Level Synthesis Flow


Lex
Parse
Compilation front-end
Behavioral Optimization
Intermediate form
Arch synth
Logic synth
Lib Binding
HLS backend

- **Expressions**

$$x = a + b \times c + d$$

- **Parse tree**

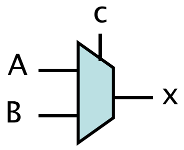


- **Data-flow graph (DFG)**
- **Control/data-flow of basic blocks (CDFG)**

*Source: R. Gupta*

## Behavioral Optimization

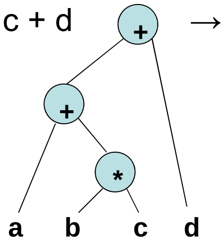
- **Data-flow transformations from software compilation**
  - Tree height reduction
    - – Balance expression tree, expose parallelism
  - Constant and variable propagation (a = 1;  c = 2 * b; → c = 2;)
  - Common sub-expression elimination (a=x+y; c=x+y; → c = a;)
  - Dead-code elimination
  - Operator strength reduction (e.g., *4 → << 2)
- **Control-flow transformations for hardware**
  - Conditional expansion
    - – If (c) then x=A else x=B
    - ➢ compute A and B in parallel, x=(C)?A:B
  - Loop expansion
    - – Instead of three iterations of a loop, replicate the loop body three times

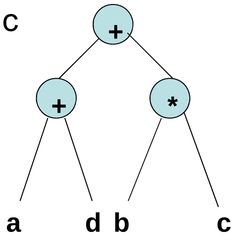


*Source: R. Gupta*

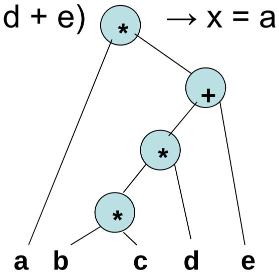
## Tree-Height Reduction

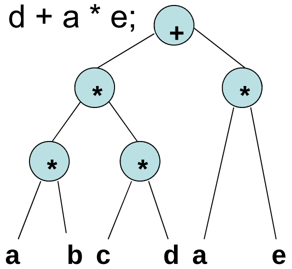
- **Commutativity and associativity**
  - x = a + b * c + d → x = (a + d) + b * c





- **Distributivity**
  - x = a * (b * c * d + e) → x = a * b * c * d + a * e;





## Architectural Synthesis

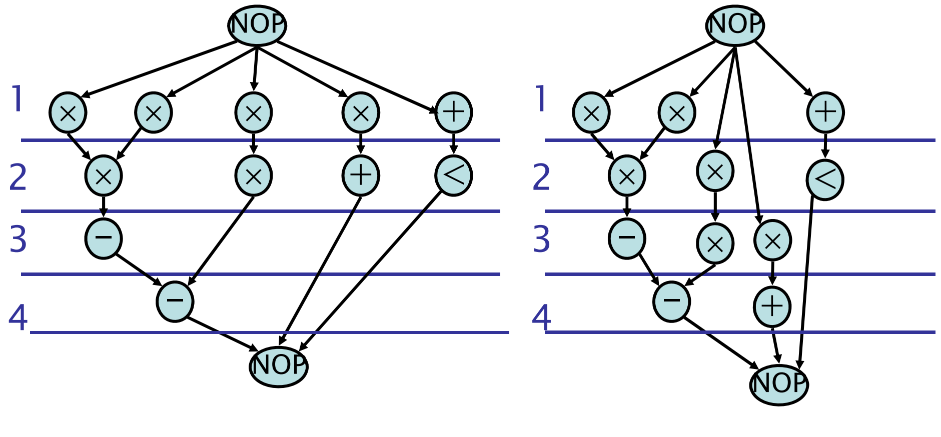
- **Deals with "computational" behavioral descriptions**
  - Behavior as sequencing graph
    (called dependency graph, or data flow graph DFG)
  - Hardware resources as library elements
    - Pipelined or non-pipelined
    - Resource performance in terms of execution delay
  - Constraints on operation timing
  - Constraints on hardware resource availability
  - Storage as registers, data transfer using wires

- **Objective**
  - Generate a synchronous, single-phase clock circuit
  - Might have multiple feasible solutions (explore tradeoff)
  - Satisfy constraints, minimize objective:
    - Maximize performance subject to area constraint
    - Minimize area subject to performance constraints

*Source: R. Gupta*

## Synthesis in Temporal Domain

- **Scheduling and binding in different order or together**
  - Schedule is a mapping of operations to time slots (cycles)
  - Scheduled sequencing graph is a labeled graph



*Source: R. Gupta*

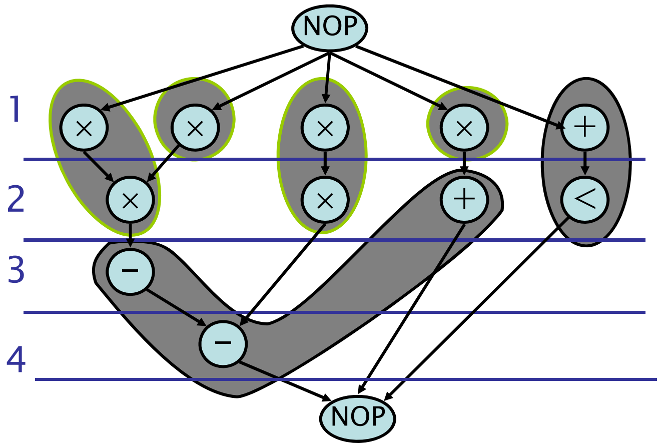
## Operation Types

- **For each operation, define its *type***
- **For each resource, define a resource type, and a delay (in terms of # cycles)**
- **T is a relation that maps an operation to a resource type that can implement it**
  - $T : V \rightarrow \{1, 2, ..., n_{res}\}$
- **More general case:**
  - A resource type may implement more than one operation type (e.g., ALU)
- **Resource binding:**
  - Map each operation to a resource with the same type
  - Might have multiple options

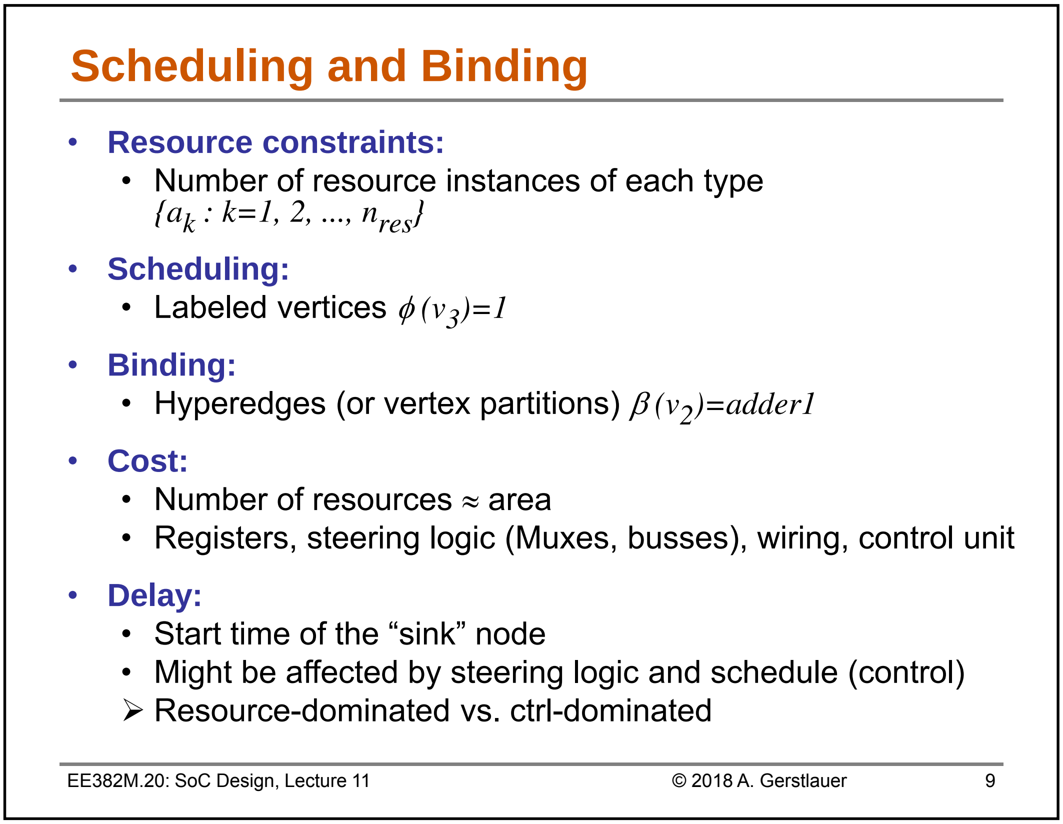
*Source: R. Gupta*

## Synthesis in Spatial Domain

- **Resource sharing**
  - More than one operation bound to same resource
  - Operations have to be serialized
  - Can be represented using hyperedges (define vertex partition)

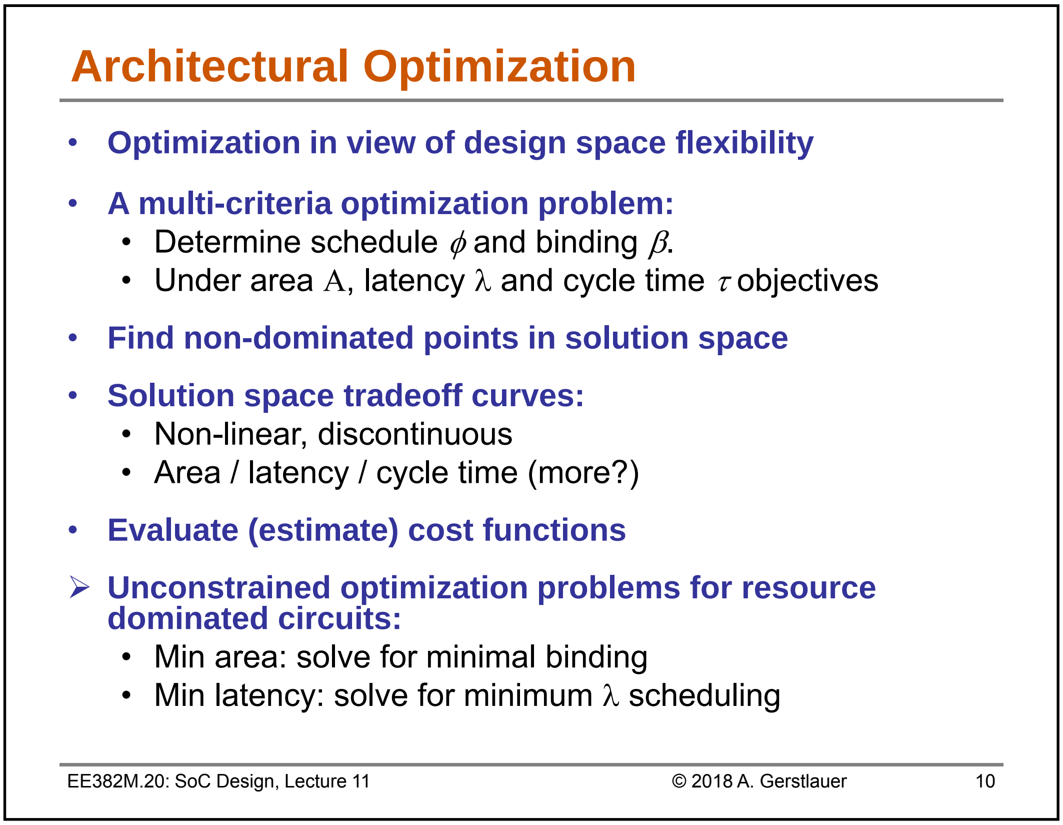


*Source: R. Gupta*



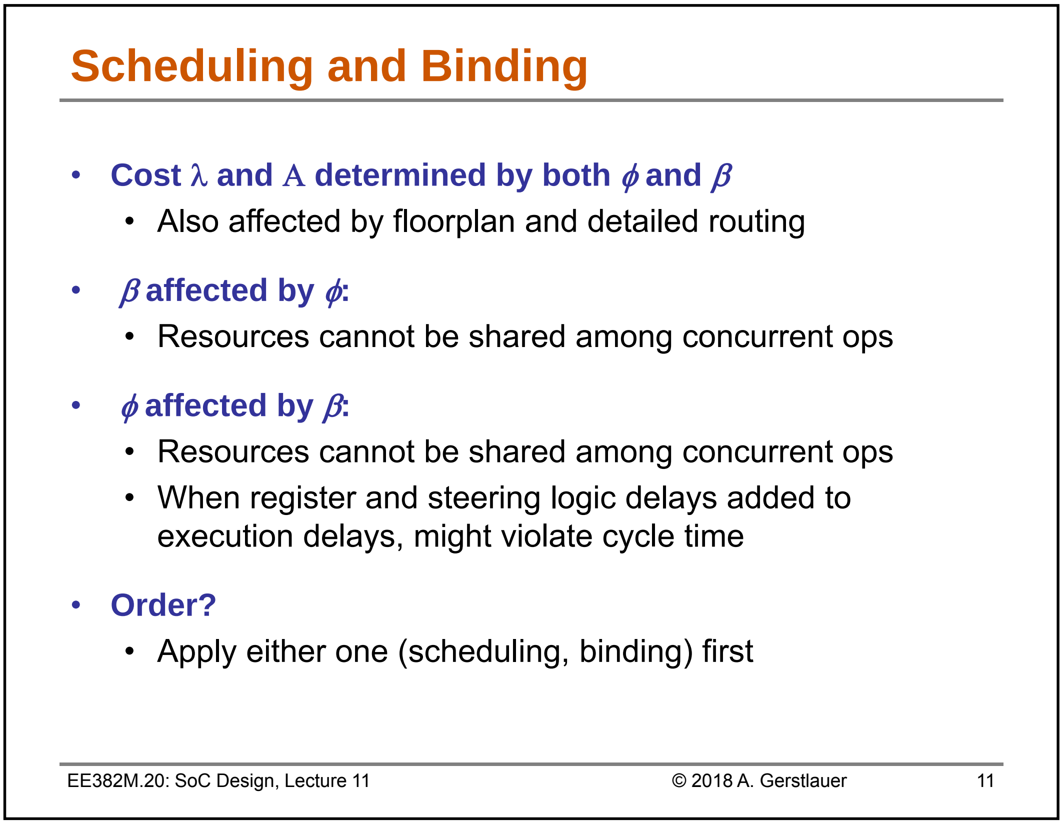
## Scheduling and Binding

- **Resource constraints:**
  - Number of resource instances of each type $\{a_k : k=1, 2, \ldots, n_{res}\}$
- **Scheduling:**
  - Labeled vertices $\phi(v_3)=1$
- **Binding:**
  - Hyperedges (or vertex partitions) $\beta(v_2)=adder1$
- **Cost:**
  - Number of resources ≈ area
  - Registers, steering logic (Muxes, busses), wiring, control unit
- **Delay:**
  - Start time of the “sink” node
  - Might be affected by steering logic and schedule (control)
  - ➢ Resource-dominated vs. ctrl-dominated



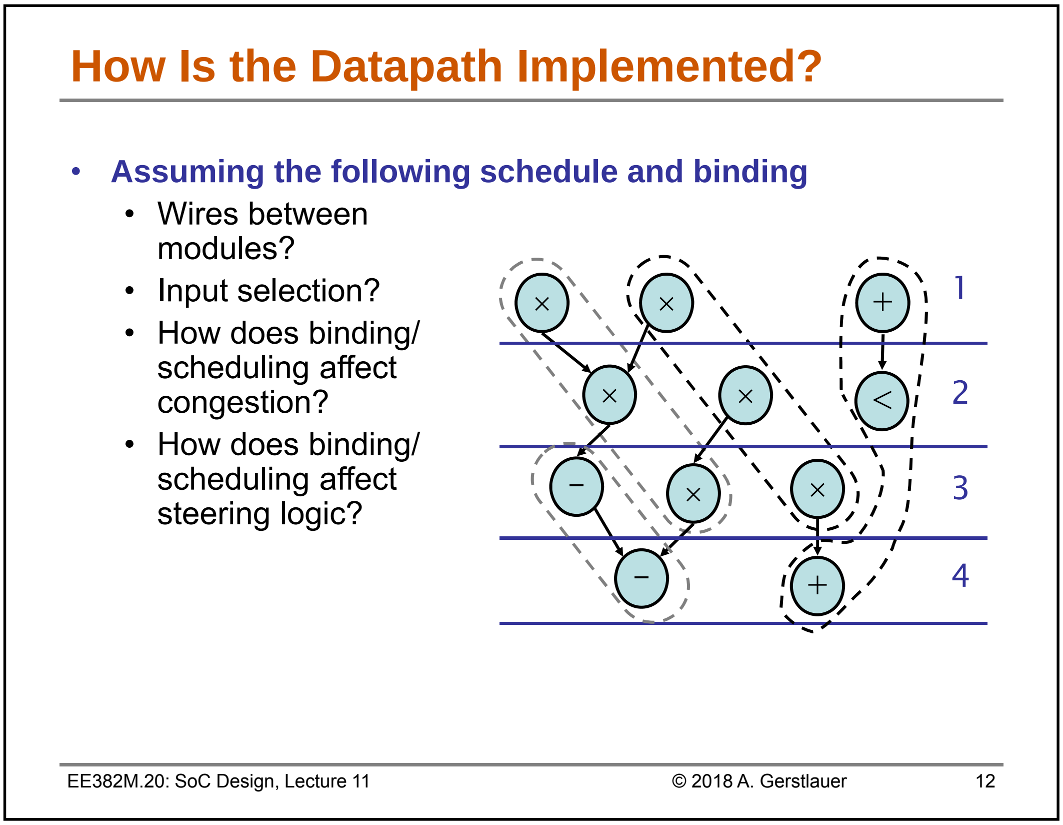
## Architectural Optimization

- **Optimization in view of design space flexibility**
- **A multi-criteria optimization problem:**
  - Determine schedule $\phi$ and binding $\beta$.
  - Under area $A$, latency $\lambda$ and cycle time $\tau$ objectives
- **Find non-dominated points in solution space**
- **Solution space tradeoff curves:**
  - Non-linear, discontinuous
  - Area / latency / cycle time (more?)
- **Evaluate (estimate) cost functions**
- ➢ **Unconstrained optimization problems for resource dominated circuits:**
  - Min area: solve for minimal binding
  - Min latency: solve for minimum $\lambda$ scheduling



## Scheduling and Binding

- **Cost $\lambda$ and $\mathrm{A}$ determined by both $\phi$ and $\beta$**
  - Also affected by floorplan and detailed routing
- **$\beta$ affected by $\phi$:**
  - Resources cannot be shared among concurrent ops
- **$\phi$ affected by $\beta$:**
  - Resources cannot be shared among concurrent ops
  - When register and steering logic delays added to execution delays, might violate cycle time
- **Order?**
  - Apply either one (scheduling, binding) first



## How Is the Datapath Implemented?

- **Assuming the following schedule and binding**
  - Wires between modules?
  - Input selection?
  - How does binding/ scheduling affect congestion?
  - How does binding/ scheduling affect steering logic?