Behavioral Synthesis

Herman Schmit

Problem Statement

- Input:
  - Data Flow Graph
    - We won't discuss control flow
  - Cell Library (e.g. fast adder, slow adder, etc.)
    - area, speed
  - Constraints or cost function:
    - area, speed
- Output:
  - Data path and control to implement

Input: Data Flow Graph

- True data dependencies in flow

Input: Cell Library

- Assume units have known execution times
  - Not data dependent

Behavioral Synthesis Process

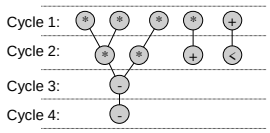
- Scheduling:
  - Assign operators to time slots
  - Don't violate constraints
- Allocation:
  - Assign operators to physical units
  - Re-use hardware for different operators
  - Minimize registers and interconnect
- Simultaneous scheduling and allocation

Scheduling

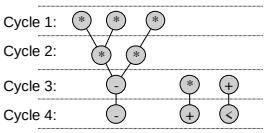
- Respect data dependencies
- Respect resource constraints
  - Don't schedule 3 multiplications if only 2 multipliers
- Respect timing constraints
- Scheduling problem is NP-complete
  - with timing and resource constraints
- Heuristics: List, Force-directed, SA

Scheduling Example 1: Unconstrained

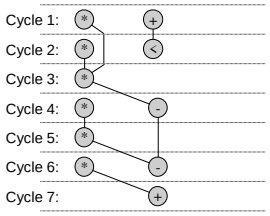
ASAP



ALAP



Scheduling Example 2: Resource Constrained



1 Multiplier
1 Adder
1 Comparitor
1 Subtractor

List Scheduling

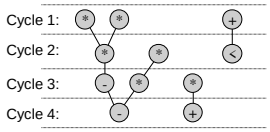
- Keep a list of "ready" operations
  - Prioritize the list based on heuristic
  - Schedule any ready operation in any available operator that can perform that operation
- Priority heuristics:
  - Most dependent operations
  - Deepest dependent tree
  - Least flexible schedule

Allocation

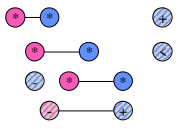
- Given scheduled DFG
- Assign nodes to resources to minimize resources
- Construct conflict graph:
  - Edge between nodes whose schedules conflict
- Solve by graph coloring

Allocation Example

Scheduled DFG



Conflict Graph (w/ coloring)



Not shown: conflicts due to operation incompatibility.
* -> Multipliers
{-,+,<} -> ALUs

Simultaneous Scheduling and Allocation

- Allocate available resources for all cycles:

	Mult 1	Mult 2	ALU 1	ALU 2
Cycle 1:				
Cycle 2:				
Cycle 3:				
Cycle 4:				

- Placement in space & time
  - Respect data dependencies
- SA and Force-directed approaches

Devadas & Newton Move Set

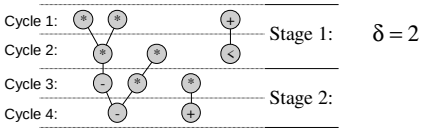
1. Interchange two operations
   swap placement
2. Displace one operation

} Satisfy constraints

3. Interchange variables on symmetric op

????

- Not clear that this can reach every design
  – Might depend on initial design state

Pipelined Datapaths

- How many cycles between new data?
  – δ = 1 (e.g. PipeRench)
    - Cannot do any functional unit sharing
    - Only "disjoint" operators can share FUs
  – δ < Schedule length
    - Can share within each stage, not between