

What's to be covered

- Hardware modeling
- Architectural synthesis
- Scheduling algorithms
- Resource sharing and binding
- Two-level combinational logic optimization
- Multiple-level combinational logic optimization
- Sequential logic optimization
- Cell library binding

- Advance Topics

1

CS220: Synthesis of Digital System, Fall 06

Design Specification and Description

2

10/5/2006

Structures

(a)

(b)

(c)

m2

m1: n1,n2,n3
 m2: n1,n2
 m3: n2,n3

3

CS220: Synthesis of Digital System, Fall 06

Logic networks

- Consists of:
 - Local function
 - Ports (I/O)
 - Directional nets
- Mixed behavioral/structural
 - May be more compact than SOP
- Very important modeling structure to understand!!!

4

CS220: Synthesis of Digital System, Fall 06

State Diagrams

- A set of primary input patterns, X .
- A set of primary output patterns, Y .
- A set of states, S .
- A state transition function, $\delta : X \times S \rightarrow S$.
- An output function, $\lambda : X \times S \rightarrow Y$ for Mealy models or $\lambda : S \rightarrow Y$ for Moore models.
- An initial state.

5

CS220: Synthesis of Digital System, Fall 06

Dataflow graph

- Vertices = operations
- Edges = data dependencies

```

xl = x + dx;
ul = u - (3*x*u*dx) - (3*y*dx);
yl = y + u*dx;
c = xl < a;
    
```

6

CS220: Synthesis of Digital System, Fall 06

(Technology Independent) Optimization

- Semantic preserving
- Implementation independent
- Dataflow based transformation
 - Tree-height reduction
 - Constant and variable propagation
 - Common subexpression elimination
 - Dead code elimination
 - Operator strength reduction
 - Code motion
- Control flow based transformation
 - Model expansion
 - Conditional expansion
 - Loop expansion

CS220: Synthesis of Digital System, Fall 06 7

Tree height reduction

$(a+b*c)+d = (a+d)+b*c$
 $a*(b*c*d+e) = a*b*c*d+a*e$

CS220: Synthesis of Digital System, Fall 06 8

Other dataflow optimization

- Constant and variable propagation
 - $a=0; b=a+1; c=2*b \Rightarrow a=0; b=1; c=2$
 - $a=x; b=a+1; c=2*a \Rightarrow a=x; b=x+1; c=2*x$
 - Remove $a=x$, if a is not used again...
- Common subexpression elimination
 - $a=x+y; b=a+1; c=x+y \Rightarrow a=x+y; b=a+1; c=a$
- Dead code elimination
 - $a=x; b=x+1; c=2*x; a=c$
- Operator strength reduction
 - $a=x^2; b=3*x \Rightarrow a=x*x; t=x<<1; b=x+t$
- Code motion
 - for ($i=1; i \leq a*b$) { ... } $\Rightarrow t=a*b$; for ($i=1; i \leq t$) { ... }
 - If a and b are not updated in the loop

CS220: Synthesis of Digital System, Fall 06 9

Control optimization

- Model expansion
 - $x=a+b; y=a*b; z=foo(x,y); foo=(p,q)\{t=q-p; return(t)\}$
 - Expands to $x=a+b; y=a*b; z=y-x$
 - I.e. inlining
- Conditional expansion
 - On Boolean operations
 - $y=ab; \text{if}(a)\{x=b+d\}\text{else}\{x=bd\}$
 - Expands to $y=ab; x=y+d(a+b)$
- Loop unrolling
 - $x=0; \text{for}(i=1; i \leq 3; i++)\{x=x+a[i]\}$
 - Unroll to $x=a[1]+a[2]+a[3]$

CS220: Synthesis of Digital System, Fall 06 10

Dataflow graph

- Vertices = operations
- Edges = data dependencies

$x1 = x + dx;$
 $u1 = u - (3 * x * u * dx) - (3 * y * dx);$
 $y1 = y + u * dx;$
 $c = x1 < a;$

CS220: Synthesis of Digital System, Fall 06 11

Sequencing graph

- Useful to represent datapath and control
- Extended dataflow graphs
 - Hierarchy
 - Control-flow commands
 - Branching
 - Iteration

CS220: Synthesis of Digital System, Fall 06 12

Hierarchy

13

CS220: Synthesis of Digital System, Fall 06

Branching

14

CS220: Synthesis of Digital System, Fall 06

Loop

15

CS220: Synthesis of Digital System, Fall 06

Semantics of sequencing graphs

- Marking of vertices
 - Waiting for execution
 - Executing
 - Have completed execution
- Execution Semantics
 - An operation can be fired as soon as all its immediate predecessors have completed execution

16

CS220: Synthesis of Digital System, Fall 06

Vertex attributes

- Area cost
- Delay cost
 - Propagation delay
 - Execution delay
- Data-dependent execution delays
 - Bounded
 - E.g. branching
 - Unbounded
 - E.g. iteration, synchronization

17

CS220: Synthesis of Digital System, Fall 06

Architecture Synthesis

10/5/2006

18

Dataflow graph

- Vertices = operations
- Edges = data dependencies

$x1 = x + dx;$
 $u1 = u - (3 * x * u * dx) - (3 * y * dx);$
 $y1 = y + u * dx;$
 $c = x1 < a;$

19

Example

- Dedicated binding
 - 6 Mult, 5 ALU

20

Example

- Shared binding
 - Bound operations must not execute concurrently
 - 4 mult, 2 ALU

21

Estimation

- (Functional) resource dominated circuit
 - Area: sum of the area of the resources
 - Latency: Start time of the sink operation
- Example
 - Area: Mult 5, ALU 1, overhead 1
 - Delay: Mult 35ns, ALU 25ns cyclotime 40ns
 - Dedicated resource: 36 units 4 cycles
 - One Mult one ALU: 7 units 7 cycles

22

Estimation for non resource-dominated circuit

- Cycle time and area also affected by
 - Register
 - Usage, setup time, propagation time
 - Steering logics
 - Bus and multiplexer usage, bus delay, propagation time
 - Wiring
 - Area and delay
 - Control
 - Area and delay, critical path
- All are affected by scheduling and binding!!!
 - Mutually dependent

23

Approaches to architectural optimization

- Area: how many MULT s and ALUs are used
 - Heavily dependent on binding
- Latency: How many “clock” till the output is ready
 - Heavily dependent on scheduling
- Cycle-time: clock period
- Delay=Cycle-time*Latency

- Area/latency trade-off
 - For some values of the cycle-time
- Cycle-time/latency trade-off
 - For some binding (area)
- Area/cycle-time trade-off
 - For some schedule (latency)

24

Approaches to architectural optimization

- Multiple-criteria optimization problem
 - Area, latency, cycle-time
- Determine Pareto optimal points
 - Implementation such that no other has all parameters with inferior values
- Draw trade-off curves
 - Discontinuous and highly nonlinear

CS220: Synthesis of Digital System, Fall 06 25

Some possible scheduling

CS220: Sy 26

Example: area/latency trade-off

- Area ≤ 20
 - Mult 5
 - ALU 1
 - Overhead 1
- Latency ≤ 8
- Mult 35ns
- ALU 25ns
- Fix Cycle time
 - 40ns
 - 30ns

CS220: Synthesis of Digital System, Fall 06 27

Example: area/latency trade-off

- Area ≤ 20
 - Mult 5
 - ALU 1
 - Overhead 1
- Latency ≤ 8
- Mult 35ns
- ALU 25ns
- Cycle time of 40ns
 - Unit Delay model
 - 1M 1A (L=7, A=7)
 - 1M 2 A (L=7, A=8)

CS220: Synthesis of Digital System, Fall 06 28

Example: area/latency trade-off

- Area ≤ 20
 - Mult 5
 - ALU 1
 - Overhead 1
- Latency ≤ 8
- Mult 35ns
- ALU 25ns
- Cycle time of 40ns
 - Unit Delay model
 - 1M 1A (L=7, A=7)
 - 1M 2 A (L=7, A=8)
 - 2M 1A (L=5, A=12)
 - 2M 2A (L=4, A=13)

CS220: Synthesis of Digital System, Fall 06 29

Example: area/latency trade-off

- Area ≤ 20
 - Mult 5
 - ALU 1
 - Overhead 1
- Latency ≤ 8
- Mult 35ns
- ALU 25ns
- Cycle time of 40ns
 - Unit Delay model
 - 1M 1A (L=7, A=7)
 - 1M 2 A (L=7, A=8)
 - 2M 1A (L=5, A=12)
 - 2M 2A (L=4, A=13)

CS220: Synthesis of Digital System, Fall 06 30

Example: area/latency trade-off

- Area ≤ 20
 - Mult 5
 - ALU 1
 - Overhead 1
- Latency ≤ 8
- Mult 35ns
- ALU 25ns
- Cycle time of 30ns
 - Mult need 2 cycles
 - 1 M ($L \geq 13$)

31

CS220: Synthesis of Digital System, Fall 06

Example: area/latency trade-off

- Area ≤ 20
 - Mult 5
 - ALU 1
 - Overhead 1
- Latency ≤ 8
- Mult 35ns
- ALU 25ns
- Cycle time of 30ns
 - Mult need 2 cycles
 - 1 M ($L \geq 13$)
 - 2M 1A ($L=8, A=12$)

32

CS220: Synthesis of Digital System, Fall 06

Example: area/latency trade-off

- Area ≤ 20
 - Mult 5
 - ALU 1
 - Overhead 1
- Latency ≤ 8
- Mult 35ns
- ALU 25ns
- Cycle time of 30ns
 - Mult need 2 cycles
 - 1 M ($L \geq 13$)
 - 2M 1A ($L=8, A=12$)
 - 3M 1A ($L=7, A=17$)
 - 3M 2A ($L=6, A=18$)
 - 4M ($A \geq 21$)

33

CS220: Synthesis of Digital System, Fall 06

Example: area/latency trade-off

- Area ≤ 20
 - Mult 5
 - ALU 1
 - Overhead 1
- Latency ≤ 8
- Mult 35ns
- ALU 25ns
- Cycle time of 30ns
 - Mult need 2 cycles
 - 1 M ($L \geq 13$)
 - 2M 1A ($L=8, A=12$)
 - 3M 1A ($L=7, A=17$)
 - 3M 2A ($L=6, A=18$)
 - 4M ($A \geq 21$)

34

CS220: Synthesis of Digital System, Fall 06

Example: cycle-time/latency trade-off

- Cycle time 20-50ns
- Latency ≤ 8
- Fixed "Area"
 - 6 Multiplier 5 ALU
 - 1 multiplier 1 ALU

35

CS220: Synthesis of Digital System, Fall 06

Example: cycle-time/latency trade-off

- M: 35ns
- A: 25ns
- 6 Multiplier 5 ALU
 - 20-24 ns cycle time
 - M, A has 2 cycle delay
 - Latency=8
 - 25-34 ns cycle time
 - M has 2 cycle delay
 - A has 1 cycle delay
 - Latency=6
 - 35-49 ns cycle time
 - M, A has 1 cycle delay
 - Latency=4
 - 50 ns cycle time
 - 2 A can be chained
 - Latency=3

36

CS220: Synthesis of Digital System, Fall 06

Example: cycle-time/latency trade-off

- M: 35ns
- A: 25ns
- Latency ≤ 8
- 1 multiplier 1 ALU
 - Cycle time must be 35 or larger

CS220: Synthesis of Digital System, Fall 06 37

Example: cycle-time/latency trade-off

- Cycle time 20-50ns
- Latency ≤ 8
- 6 Multiplier 5 ALU
 - 20-24 ns cycle time
 - M, A has 2 cycle delay
 - Latency=8
 - 25-34 ns cycle time
 - M has 2 cycle delay
 - Latency=6
 - 35-49 ns cycle time
 - Latency=4
 - 50 ns cycle time
 - 2 A can be chained
 - Latency=3
- 1 multiplier 1 ALU
 - Cycle time must be 35 or larger

CS220: Synthesis of Digital System, Fall 06 38

Datapath synthesis

- Resource binding
 - Assigning operations to resources
- Connectivity synthesis
 - Connection of resources to
 - Multiplexers busses and registers
 - Control unit interface
 - I/O ports

CS220: Synthesis of Digital System, Fall 06 39

CS220: Synthesis of Digital System, Fall 06 40

Connecting things up

CS220: Synthesis of Digital System, Fall 06 41

Control synthesis

- Synthesis of the control unit
- Logic model
 - Synchronous FSM
- Physical implementation
 - Microcode (ROM, PLA)
 - Hard-wired FSM
 - Distributed FSM

CS220: Synthesis of Digital System, Fall 06 42

Control synthesis

- Synthesize circuit that
 - Executes scheduled operations
 - Provides synchronization
 - Supports
 - Iteration
 - Branching
 - Hierarchy
 - Interfaces
- Assumption
 - Synchronous implementation
 - Control unit is a FSM (or connection of FSMs)

43

Example

44

High-level synthesis of pipelined circuits

- Pipeline circuits
 - Concurrent execution of operations on different data sets
 - Increase throughput
 - I/O data rate
 - Preserve latency
- Applicable to General purpose processors
- Digital signal processors

45

Example

- Fix the latency at 4
- Unpipelined implementation
 - Given the "best" scheduling
 - Require only 2 Mult 2 ALU
- Cycle-time is 40 ns
 - Mult delay is 35, plus some overhead
 - Throughput is $1/(40ns*4) = 6.25M$ Hz, or 6.25 M I/O per second

46

Example

- Fix the latency at 4
- 2-stage pipelined implementation
 - Required dedicated binding
 - 3 Mult 3 ALU
- Cycle-time is 40 ns
 - Throughput is $1/(40ns*2) = 12.5M$ Hz, or 12.5 M I/O per second

47

Example

- Fix the latency at 4
- Fully pipelined implementation
 - Required dedicated binding
 - 6 Mult 5 ALU
- Cycle-time is 40 ns
 - Throughput is $1/40ns = 25M$ Hz, or 25 M I/O per second

48

Scheduling Algorithm

10/5/2006 49

Example

CS220: Synthesis of Digital System, Fall 06

Simplest model

- All operations have bounded delays
 - Cycle-time is given
- All delays are in cycles
 - Cycle-time is given
- No constraints
 - No bounds on area
- Goal
 - Minimize latency

CS220: Synthesis of Digital System, Fall 06 51

Minimum-latency unconstrained scheduling problem

- Given a set of ops V with integer delays D and a partial order on the operations E :
- Find an integer labeling of the operations $\phi : V \rightarrow \mathbb{Z}^+$, such that
 - $t_i = \phi(v_i)$,
 - $t_i \geq t_j + d_j \quad \forall i, j \text{ s.t. } (v_j, v_i) \in E$
 - And t_n is minimum

CS220: Synthesis of Digital System, Fall 06 52

ASAP scheduling algorithm

```

ASAP (  $G_s(V; E)$  ) {
  Schedule  $v_0$  by setting  $t_0^S = 1$ ;
  repeat {
    Select a vertex  $v_i$  whose pred. are all scheduled;
    Schedule  $v_i$  by setting  $t_i^S = \max_{j:(v_j, v_i) \in E} t_j^S + d_j$ ;
  }
  until ( $v_n$  is scheduled) ;
  return ( $t^S$ );
}
    
```

CS220: Synthesis of Digital System, Fall 06 53

Example

CS220: Synthesis of Digital System, Fall 06 54

ALAP scheduling algorithm

```

ALAP( $G_s(V, E), \bar{\lambda}$ ) {
  Schedule  $v_0$  by setting  $t_0^L = \bar{\lambda} + 1$ ;
  repeat {
    Select vertex  $v_i$  whose succ. are all scheduled;
    Schedule  $v_i$  by setting  $t_i^L = \min_{j:(v_i, v_j) \in E} t_j^L - d_i$ ;
  }
  until ( $v_0$  is scheduled) ;
  return ( $t^L$ );
}
    
```

CS220: Synthesis of Digital System, Fall 06

Example

CS220: Synthesis of Digital System, Fall 06

Remarks

- ALAP solves a latency-constrained problem
- Latency bound can be set to latency computed by ASAP algorithm
- Mobility
 - Defined for each operation
 - Difference between ALAP and ASAP schedule
- Slack on the start time

CS220: Synthesis of Digital System, Fall 06

Example

- Operation with zero mobility
 - $\{v_1, v_2, v_3, v_4, v_5\}$
 - Critical path.
- Operations with mobility 1
 - $\{v_6, v_7\}$
- Operation with mobility 2
 - $\{v_8, v_9, v_{10}, v_{11}\}$

CS220: Synthesis of Digital System, Fall 06