SYNCHRONOUS-LOGIC OPTIMIZATION

©*Giovanni De Micheli*

Stanford University

Outline

- Structural optimization methods:

  - Peripheral retiming

  - Synchronous-logic transformations.

  - Synchronous *don't cares* .

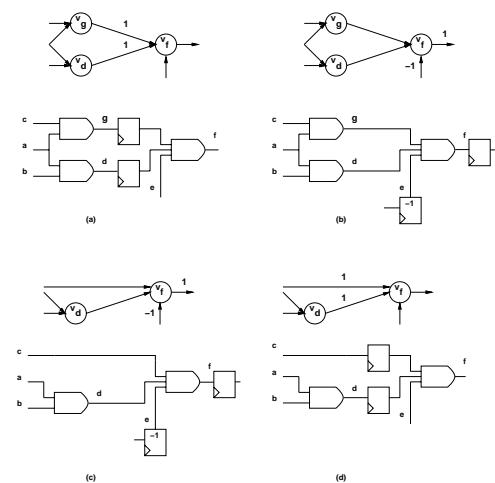- Relations between state-based models and structural models.

Peripheral retiming

- Alternate retiming and comb. synthesis.

- Move register position to periphery:

  - Maximize the scope of combinational logic.

  - Borrow and release synchronous delays.

- Optimize combinational logic.

- Return borrowed synchronous delays.

Example

Assumption for peripheral retiming

- The network graph is acyclic.

- There are no two paths from an input to an output vertex with different weights.

- There exists integer vectors
  - $\mathbf{a} \in Z^{|V^I|}$ and $\mathbf{b} \in Z^{|V^O|}$, such that

  - $w(v_i, \ldots, v_j) = a_i + b_j$ for all paths $(v_i, \ldots, v_j)$ with $v_i \in V^I, v_j \in V^O$.

- Remarks:
  - Applicable to pipelined networks.
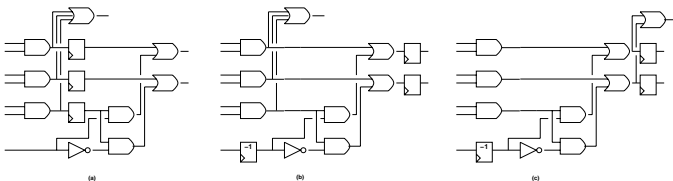 - Extensible to circuits with feedback by using partitioning.

Logic transformations and peripheral retiming

- Apply combinational logic transformations.

- Requirement:

  - No negative weight on I/O paths to guarantee that circuit can be retimed again.
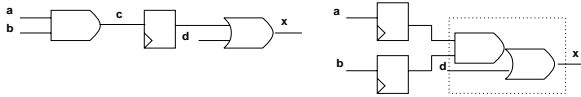
- Reject some transformations.

Example

Algebraic synchronous logic transformations

- Combine transformations with retiming.

- Transform combinational logic expressions:

  - *Within* register boundaries.

  - *Across* register boundaries.

- Extension of *algebraic* transformations.
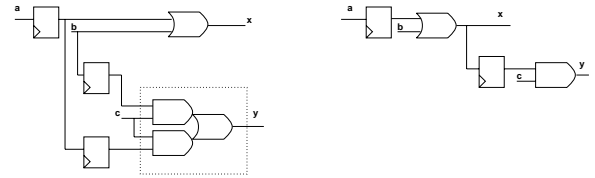
Example of synchronous elimination

$$c = ab; \ x = d + c@1 \qquad\qquad x = d + a@1b@1$$

Example of synchronous substitution

$$x = a@1 + b; \ y = a@2c + b@1c$$

$$x = a@1 + b; \ y = x@1c$$

Boolean synchronous logic transformations

- Boolean function minimization:

  - Functions of delayed variables.

  - *Explicit* synchronous *don't care* conditions.

  - Extension of combinational methods.

- Boolean relation minimization:

  - Problem generalization.

  - *Implicit don't care* conditions.

Extension of classic *don't care* conditions to the synchronous domain

- *Controllability don't care* sets:

  - Conditions that cannot occur:

    * Due to *external* connections.

    * Due to *internal* connections.

- *Observability don't care* sets:

  - Conditions such that a variable is not observed at present or in the future.

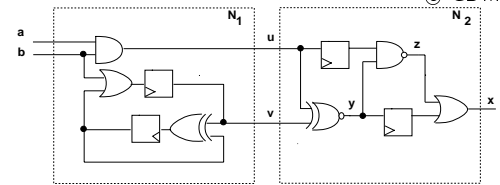   * At some *external* port.

    * At some *internal* gate.

Explicit *don't care representation*
© GDM

- Synchronous literal: literal with time label.

- Synchronous product: product of sync. literals.

- *Don't care* sets:

  - Sums of synchronous products.

  - Time invariant component.

  - Time dependent component.

Example
© GDM

- Initialization:

  - $(b^{(-4)}, b^{(-3)}, b^{(-2)}) = (1, 0, 1)$.

- Transient *don't care* condition:

  - $v'^{(-3)} + v'^{(-1)}$.

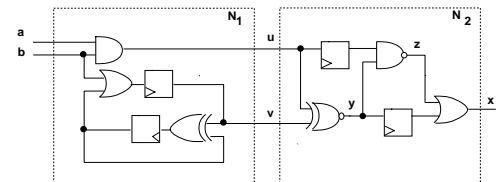- Time-invariant *don't care* condition:

  - $u^{(n)} v'^{(n+1)}$.

Synchronous logic optimization using explicit *don't care* sets.
© GDM

- Compute local *don't care* sets:

  - Extensions of *controllability*, *observability* algorithms for combinational circuits.

- Optimize functions w.r. to local *don't care* sets:

  - Rename time-labeled variables.
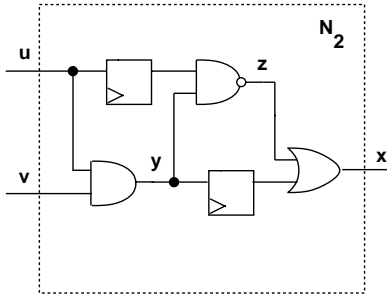
  - Use two-level minimizers.

Example
© GDM

- Replace $EXNOR$ gate by an AND gate.

  - Perturbation:
    * $\delta^{(n)} = (u^{(n)} v^{(n)} + u'^{(n)} v'^{(n)}) \oplus (u^{(n)} v^{(n)}) = u'^{(n)} v'^{(n)}$

- Compute local *don't care* set:

  - $\widetilde{DC}_y$ contains $u'^{(n-1)} u'^{(n)} + u^{(n-1)} v'^{(n)}$.

- Perturbation is bounded by *don't care* set:

  - Replacement feasible !
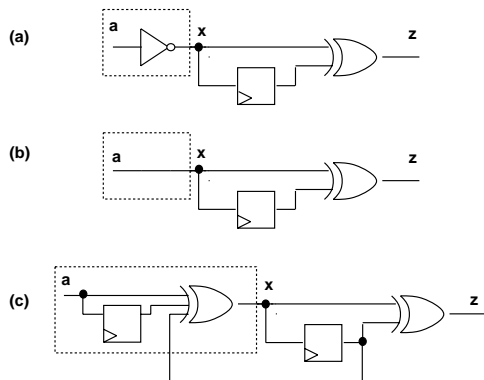
Example

Implicit *don't care* conditions

- Explicit *don't care* sets do not represent all degrees of freedom for optimization.

- There are some feasible simplifications that require a more complex model.

  - Synchronous Boolean relation models.

  - Implicit *don't care* condition representations.

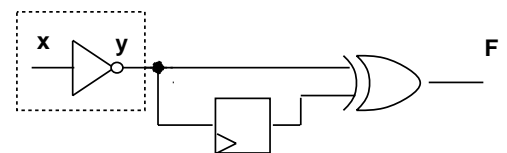- Specialized algorithms.

Example
Path reconvergence with unequal delays

Example of implicit representation

- Equating terminal behavior:

  - $F = x'_n \oplus x'_{n-1} = y_n \oplus y_{n-1}$

  - $(x'_n \oplus x'_{n-1}) \overline{\oplus} (y_n \oplus y_{n-1}) = 1$

- Example of solutions:

  - $y_n = x_n$

  - $y_n = x'_n$

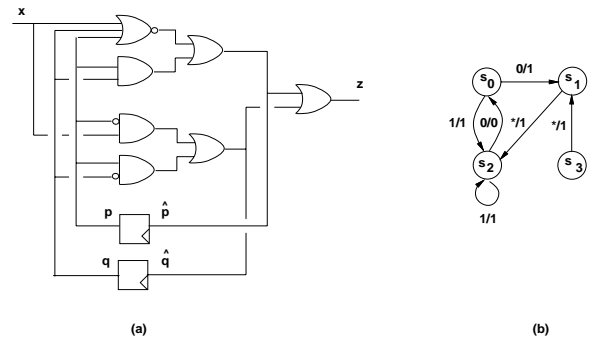 - $y_n = x_n \oplus x_{n-1} \oplus y_{n-1}$

Relating the structural to the state-based models.

© GDM

- State encoding:

  – Maps the state-based representation into a structural one.

- State extraction:

  – Recovers the state information from a structural model.

- Remark:

  – A circuit with n register may have 2^n states.

  – Unreachable states.

State Extraction

© GDM

(a) (b)

- State variables: p, q.

- Initial state: $p = 0, \quad q = 0$.

- Four possible states.

State Extraction

© GDM

- Reachability analysis.

  – Given a state
  determine which states are reachable for some inputs.

  – Given a state subset
  determine the reachable state subset.

  – Start from initial state.

  – Stop when convergence is reached.

- Notation:

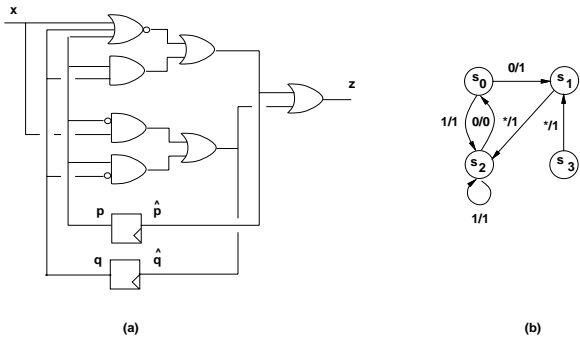 – A state (or state subset) is an expression over the state variables.

Reachability analysis

© GDM

- State transition function: **f**

- Initial state: r_0.

- States reachable from r_0:

  – Image of r_0 under **f**.

- States reachable from set r_k:

  – Image of r_k under **f**.

- Iteration:

  – $r_{k+1} = r_k \cup$ (image of r_k under **f**).

- Convergence:

  – $r_{k+1} = r_k$ for some k.

Example

(a) (b)

- Initial state $r_0 = p'q'$.

- The state transition function $\mathbf{f} = \begin{bmatrix} x'p'q' + pq \\ xp' + pq' \end{bmatrix}$.

Example

- Image of $p'q'$ under $\mathbf{f}$:

  - When $(p = 0; q = 0)$, $\mathbf{f}$ reduces to $\begin{bmatrix} x' \\ x \end{bmatrix}$

  - Range is $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$ and $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$

- States reachable from the reset state:

  - $(p = 1; q = 0)$ and $(p = 0; q = 1)$.

  - $r_1 = p'q' + pq' + p'q = p' + q'$.

- States reachable from r_1: $\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

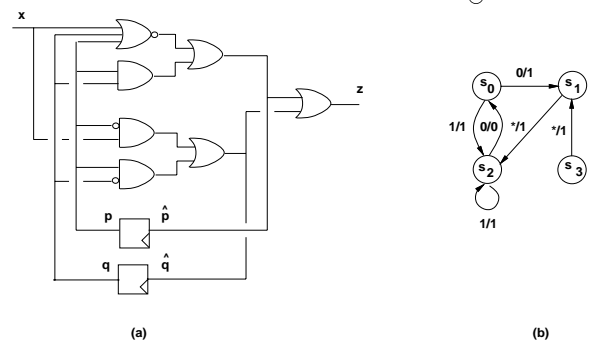- Convergence: $s_0 = p'q'$; $s_1 = pq'$; $s_2 = p'q$.

Completing the extraction

- Determine state set (vertex set).

- Determine transitions (edge set) and I/O labels.

  - Inverse image computation.

  - Look at conditions that lead you into a given state.

Example

(a) (b)

- Transitions into $s_0 = p'q'$.

  - Patterns that make $\mathbf{f} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

  - $(x'p'q' + pq)'(xp' + pq')' \;=\; x'p'q$

  - Transition from state $s_2 = p'q$ under input x'.

Remarks

- Extraction is performed efficiently with BDDs.

- Model the transition relation $\chi(\mathbf{i}, \mathbf{x}, \mathbf{y})$ with BDD.

  - Links possible triples of

    * (input, state, next-state).

- Image of r_k (where r_k depends on $\mathbf{x}$).

  - $\mathcal{S}_{\mathbf{i}, \mathbf{x}}(\chi(\mathbf{i}, \mathbf{x}, \mathbf{y}) \cdot r_k(\mathbf{x}))$.

Summary
Optimization of synchronous circuits

- State-based models:

  - Classic FSM optimization.

- Structural models:

  - Retiming.

  - Peripheral retiming.

  - Algebraic and Boolean transformations.

- Still area of active research.