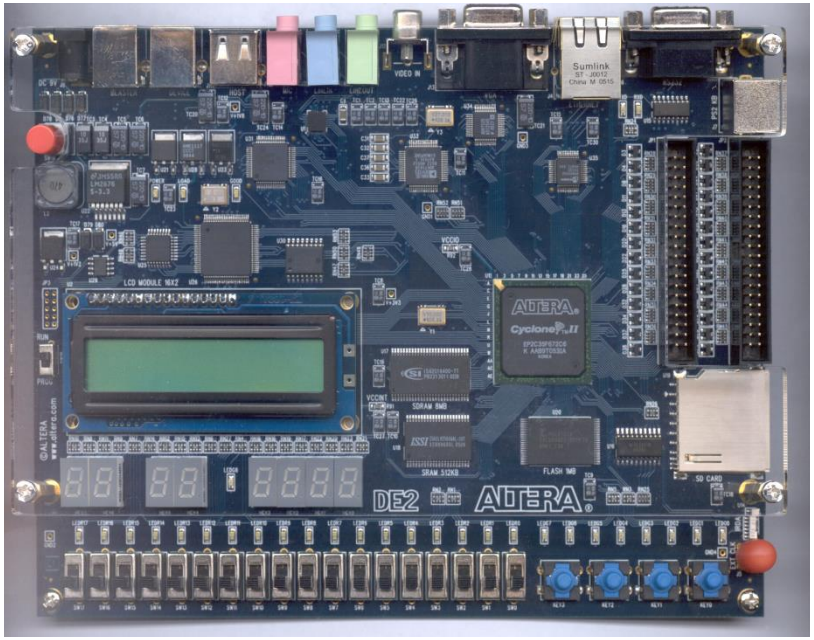
CPE 322
Digital Hardware Design Fundamentals

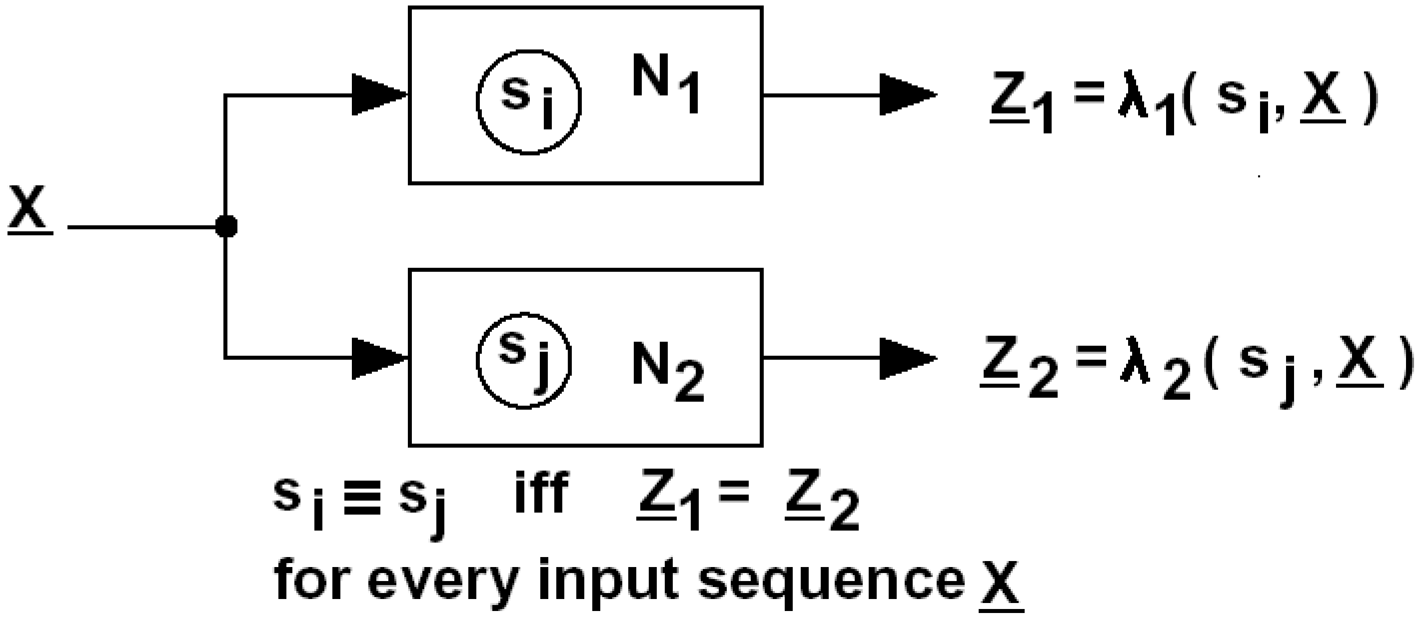
Electrical and Computer Engineering

State Reduction in Mealy and Moore Finite State Machines



Equivalent States

- Two state are equivalent if we cannot tell them apart by observing input and output sequences



Definition: Two states are equivalent si==sj only and only if, for every input sequence X, the output sequences Z1 and Z2 are the same.

Not practical => try all sequences (what is the length of sequence?)

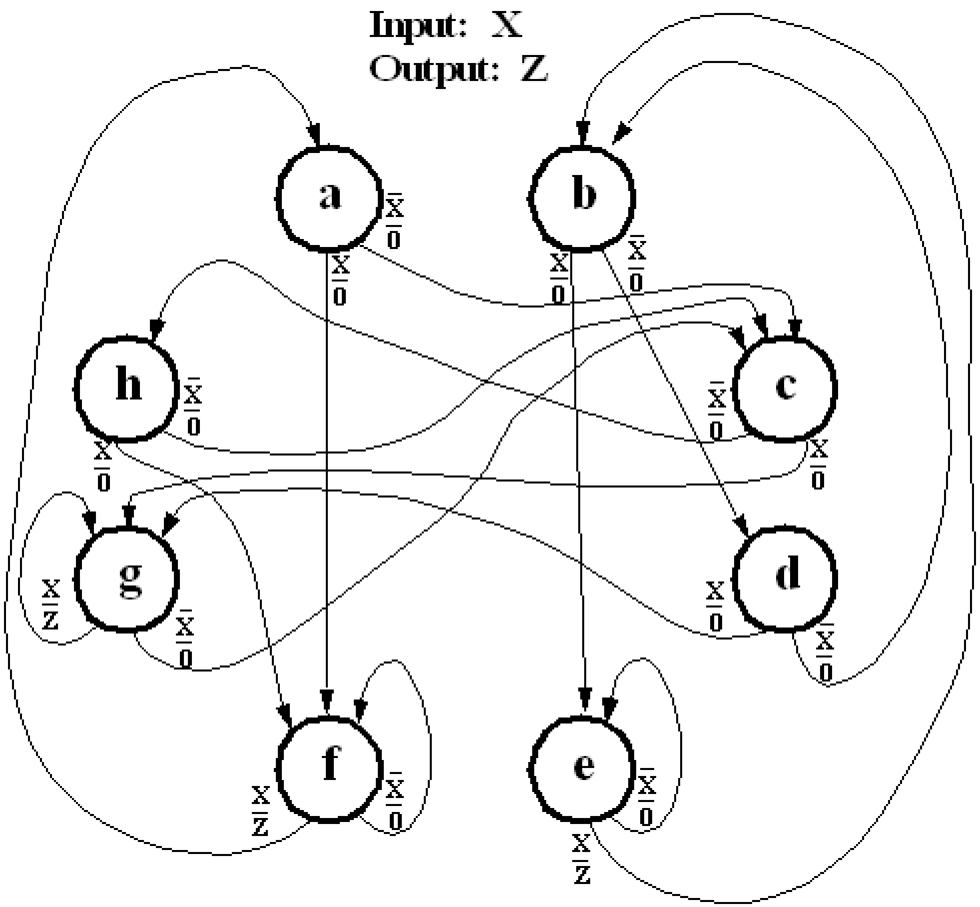
State Equivalence Theorem

- Two states, Si and Sj are *equivalent* (i.e. Si == Sj) if and only if for every single input X, the outputs are the same and the next states are found to be equivalent.
- Only one representation of the state is needed for each set of equivalent states found.
  - The others can be removed.

Equivalent State Determination Methodology

- – If two states are the same state then they are equivalent.
- – Other equivalent states are found by systematically examining each two state combination present in the FSM to determine if they are equivalent based upon the application of the State Equivalence Theorem.
- – This is a multiphase operation with state pairs that are found to be not equivalent being removed from further consideration in the next phase.
- – States whose equivalence cannot be determined are passed on to the next phase.
- – When there is no further non-equivalent states found in a phase then the process has found all distinct states.
  - Any remaining two-state combinations must be equivalent.

State Table Reduction Example

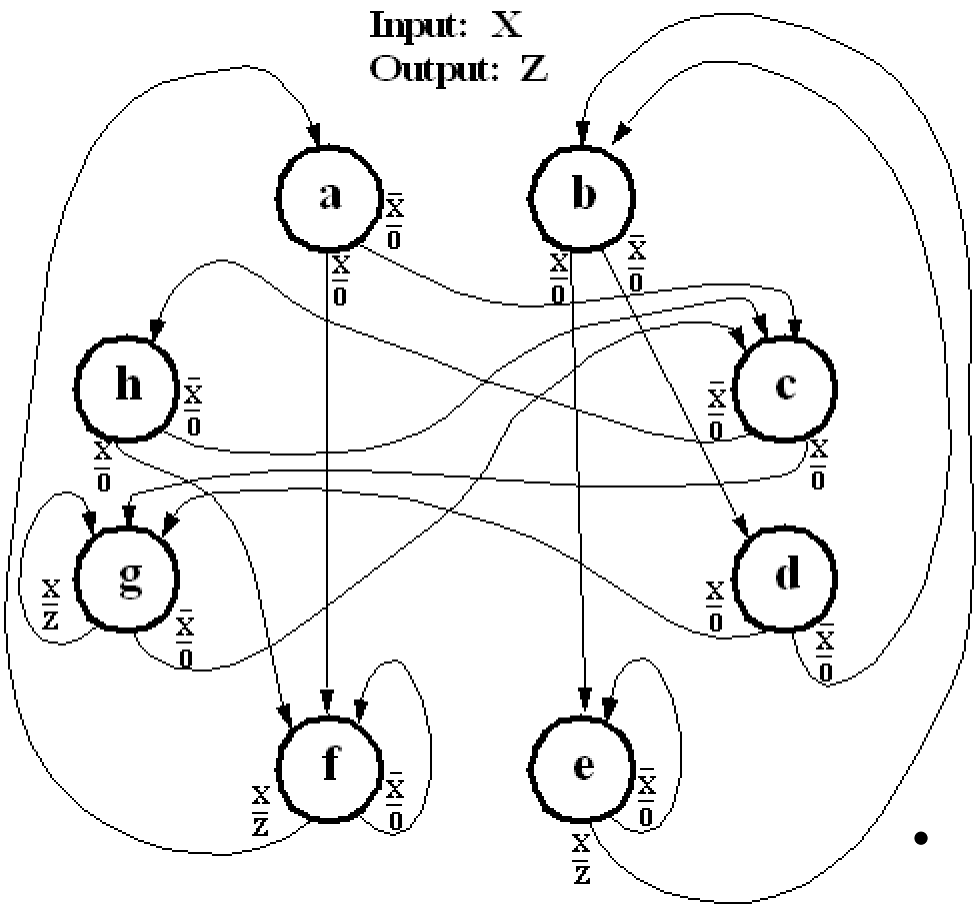


Extended State Transition Graph

Present State	Next State X = 0	Next State X = 1	Present Output X = 0	Present Output X = 1
a	c	f	0	0
b	d	e	0	0
c	h	g	0	0
d	b	g	0	0
e	e	b	0	1
f	f	a	0	1
g	c	g	0	1
h	c	f	0	0

State Table

State Table Reduction Example



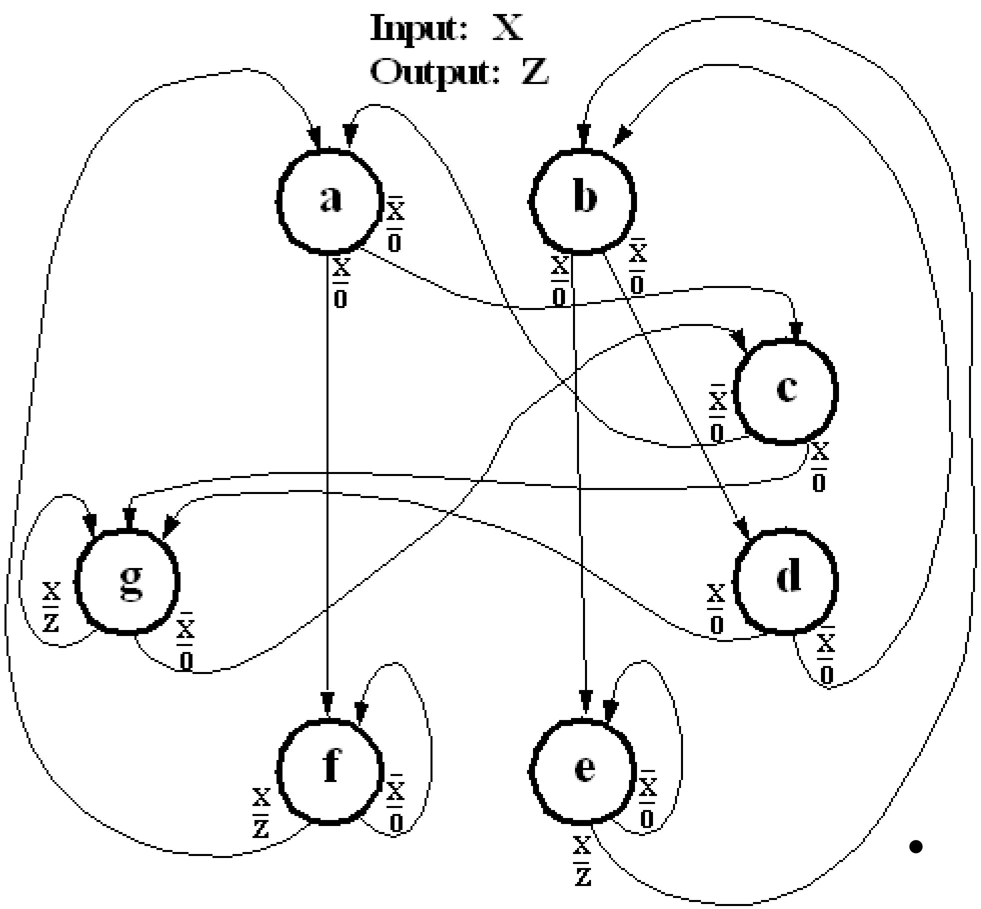
Extended State Transition Graph

Present State	Next State X = 0	Next State X = 1	Present Output X = 0	Present Output X = 1
a	c	f	0	0
b	d	e	0	0
c	~~h~~ a	g	0	0
d	b	g	0	0
e	e	b	0	1
f	f	a	0	1
g	c	g	0	1
~~h~~	~~c~~	~~f~~	~~0~~	~~0~~

State Table

- By direct application of the Equivalent State Theorem for states **a** and **h**.

State Table Reduction Example



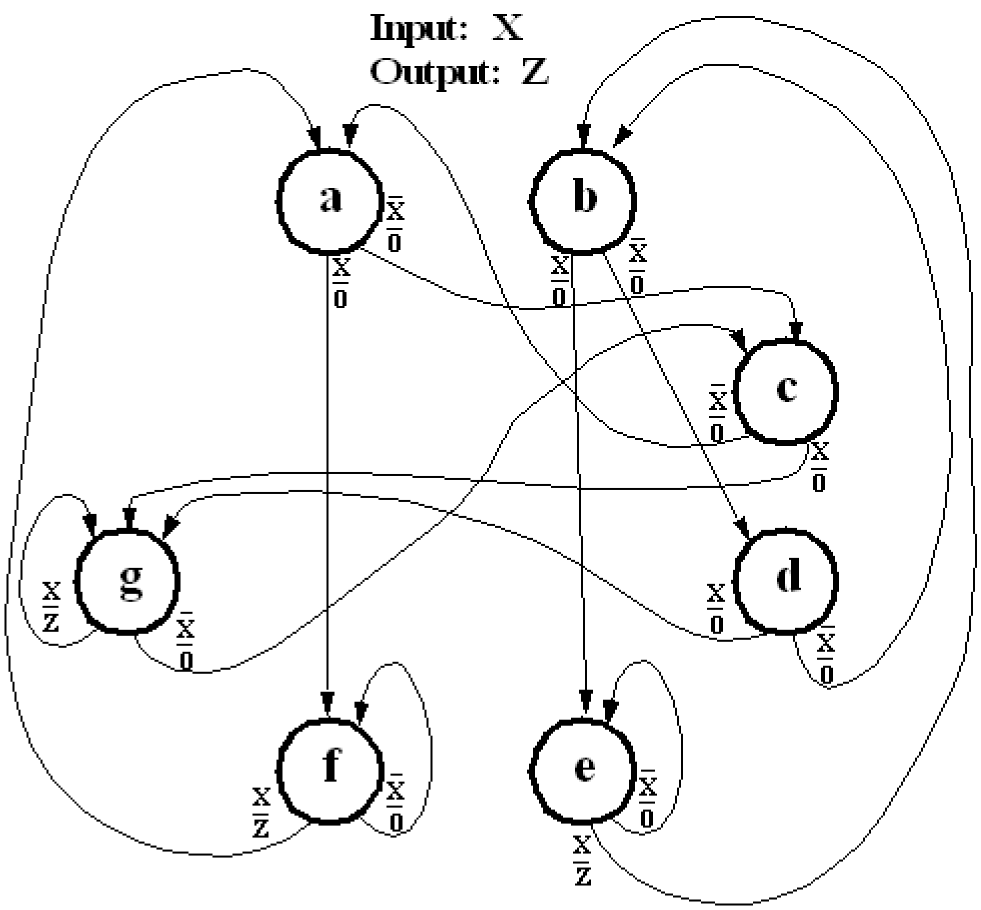
Extended State Transition Graph

Present State	Next State X = 0	1	Present Output X = 0	1
a	c	f	0	0
b	d	e	0	0
c	~~h~~ a	g	0	0
d	b	g	0	0
e	e	b	0	1
f	f	a	0	1
g	c	g	0	1
~~h~~	~~c~~	~~f~~	~~0~~	~~0~~

State Table

- By direct application of the Equivalent State Theorem for states **a** and **h**.

State Table Reduction Example



Extended State Transition Graph

Present State	Next State X = 0	1	Present Output X = 0	1
a	c	f	0	0
b	d	e	0	0
c	a	g	0	0
d	b	g	0	0
e	e	b	0	1
f	f	a	0	1
g	c	g	0	1

State Table

State Table Reduction Example

- This reduction was done by inspection but further state reduction requires an iterative evaluation of the states.

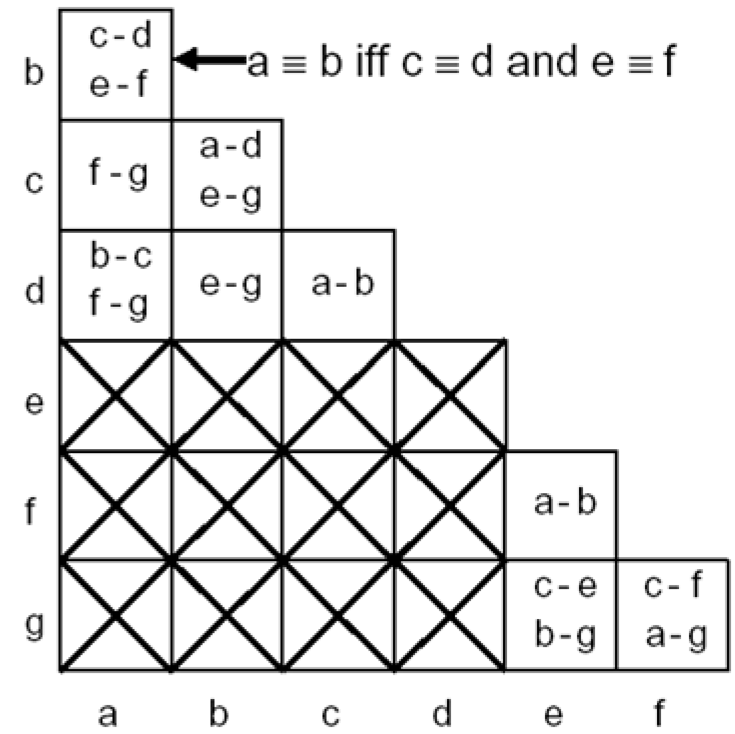
Implication Table Construction

- Evaluating the two state equivalence comparison process is aided by the use of an implication table
  - This is in effect a lower triangular portion of a square matrix where both dimensions represent the number of states in the FSM.
  - On an N state FSM representation:
    - The y axis proceeds from State 2 to State N.
    - The x axis goes from State 1 to State N-1.
  - Each Square of the Implication table should be labeled with the conditions necessary for state equivalence for the two states associated with the (row,column) pair.
    - This is obtained from the State Table or STG.
  - State pairs that cannot be equivalent are marked by placing an X in the (row/column) square on the implication table.

State Table Reduction

Present State	Next State X = 0	Next State X = 1	Present Output X = 0	Present Output X = 1
a	c	f	0	0
b	d	e	0	0
c	a	g	0	0
d	b	g	0	0
e	e	b	0	1
f	f	a	0	1
g	c	g	0	1

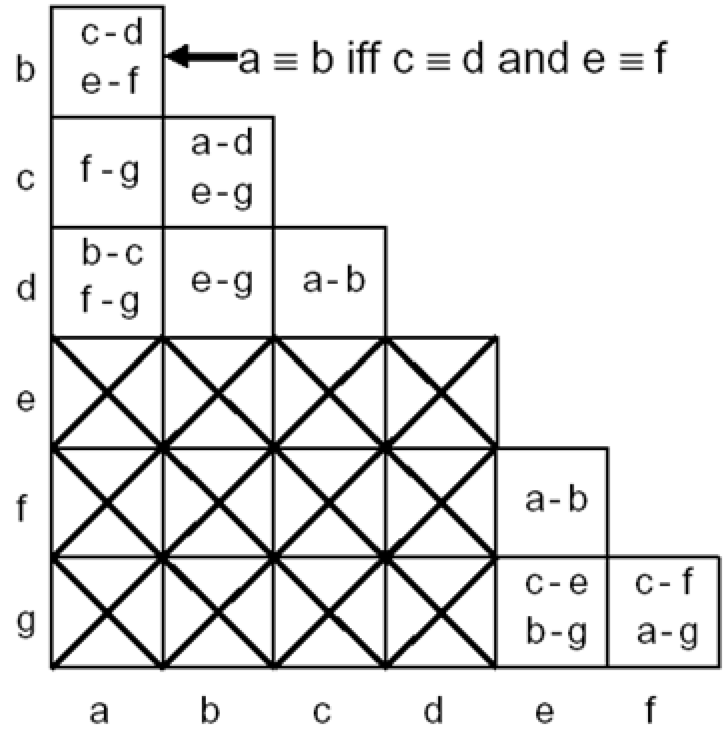
State Table



- State combinations whose outputs differ are not equivalent so the corresponding square is marked with an X
- Other Squares contain the next state requirements for equivalency.
  - For example States a and b have the same output => they are same iff c==d and f==e. We say c-d and e-f are *implied pairs* for a-b. They may or may not be equivalent – can not tell in this phase

State Table Reduction

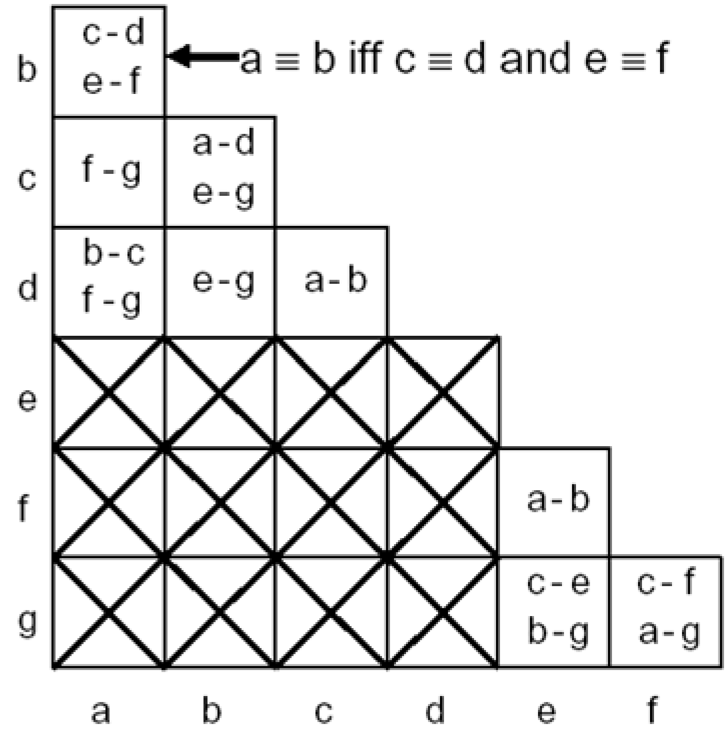
- Consider square (b,a) to be equivalent
  c == d && e == f. Can't determine
  remain implied pairs.
- Consider square (c,a) to be equivalent
  f == g. Can't determine
  remain implied pairs.
- Consider square (d,a) to be equivalent
  b == c && f == g. Can't determine
  remain implied pairs.
- Consider square (c,b) to be equivalent
  a == d && e == g. Can't determine
  remain implied pairs.
- Consider square (d,b) to be equivalent
  e== g. Can't determine
  remain implied pairs.
- Consider square (d,c) to be equivalent
  a == b Can't determine
  remain implied pairs.



- Consider square (f,e) to be equivalent
  a == b Can't determine
  remain implied pairs.

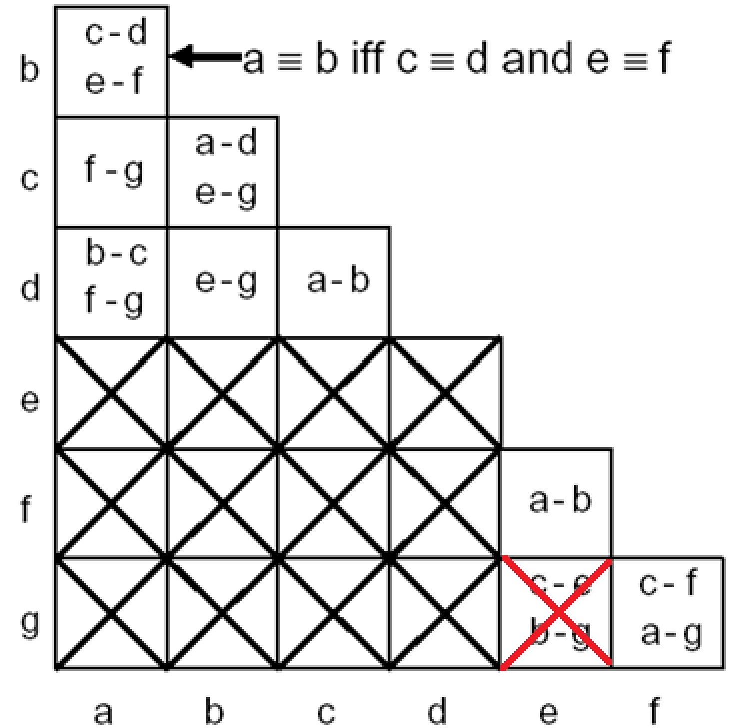
State Table Reduction

- Consider square (g,e) to be equivalent
  c == e && b == g. This is **not** true.
  c !== e since it has an X in the
  (e,c) square [same is true for
  (b,g) square but only one is
  needed to declare f not equivalent
  to g].



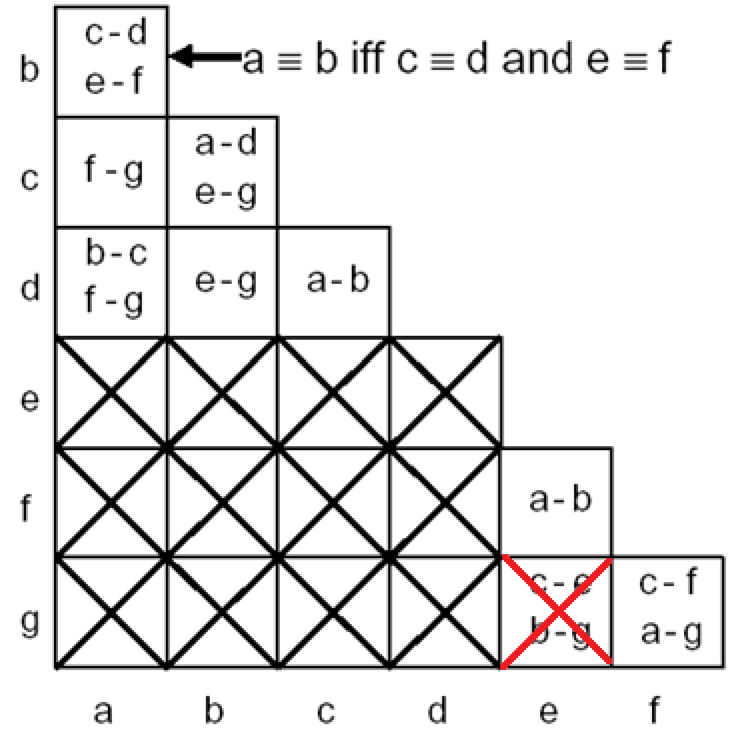
State Table Reduction

- Consider square (g,e) to be equivalent
  c == e && b == g. This is **<u>not</u>** true. c !== e since it has an X in the (e,c) square [same is true for (b,g) square but only one is needed to declare f not equivalent to g].
- X is placed in (g,e) location.



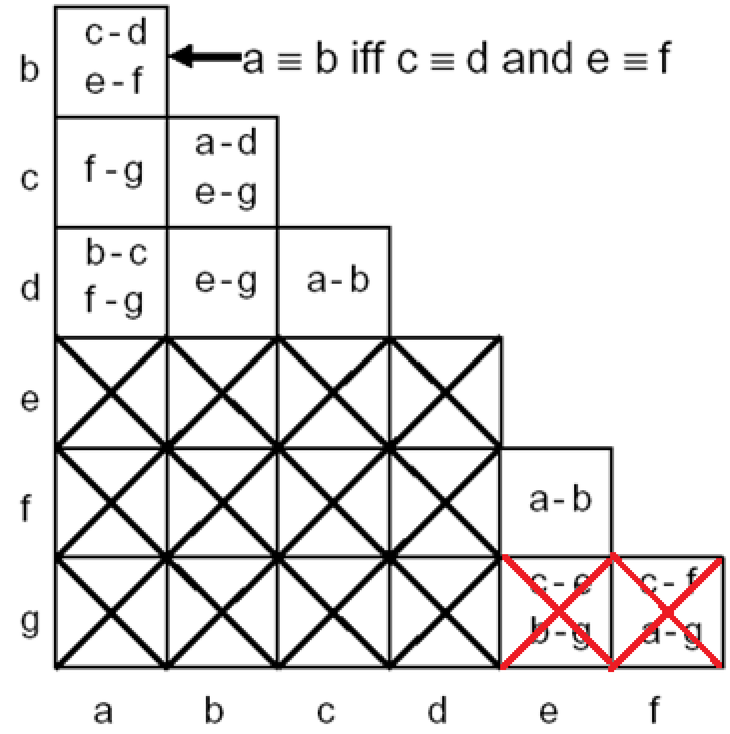
State Table Reduction

- Consider square (g,f) to be equivalent
  e == f && a == g. This is **<u>not</u>** true.
  a !== g since it has an X in the
  (g,a) square.



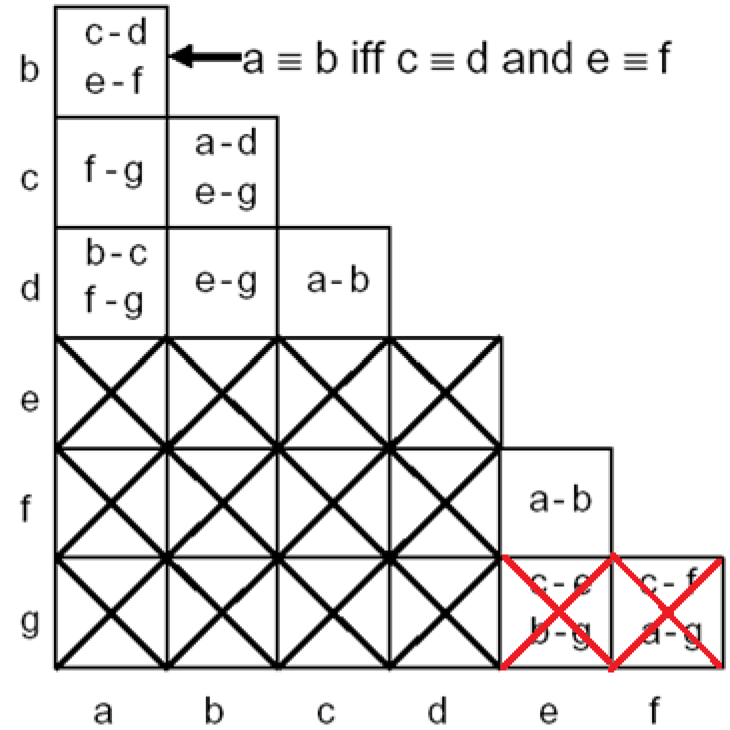
State Table Reduction

- Consider square (g,f) to be equivalent
  e == f && a == g. This is **not** true.
  a !== g since it has an X in the
  (g,a) square.
- Place an X in the (g,f) location.



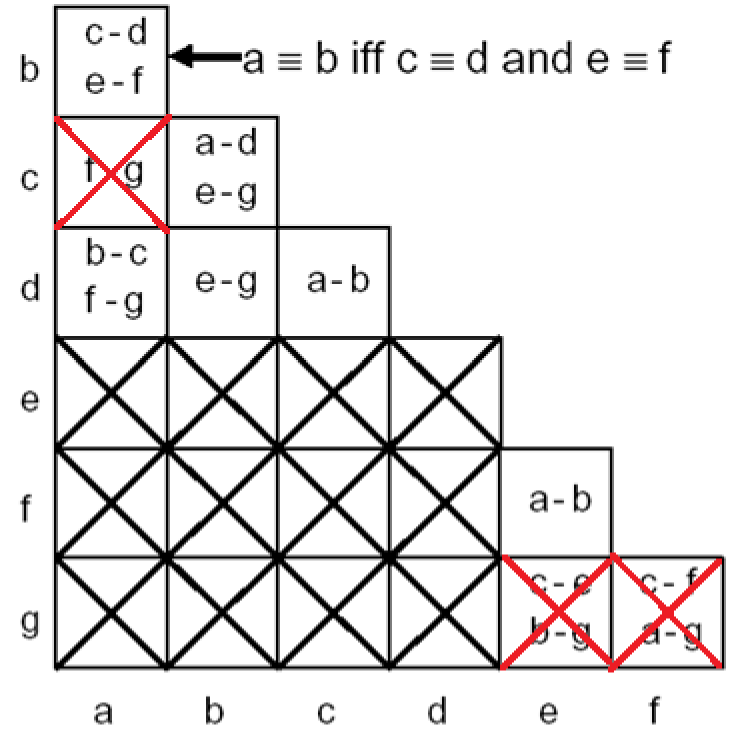
State Table Reduction

- Consider square (b,a) to be equivalent
  c == d && e == f. Can’t determine
  remain implied pairs.
- Consider square (c,a) to be equivalent
  f == g. This is **<u>not</u>** true.
  f !== g since it now has an X in the
  (f,g) square.



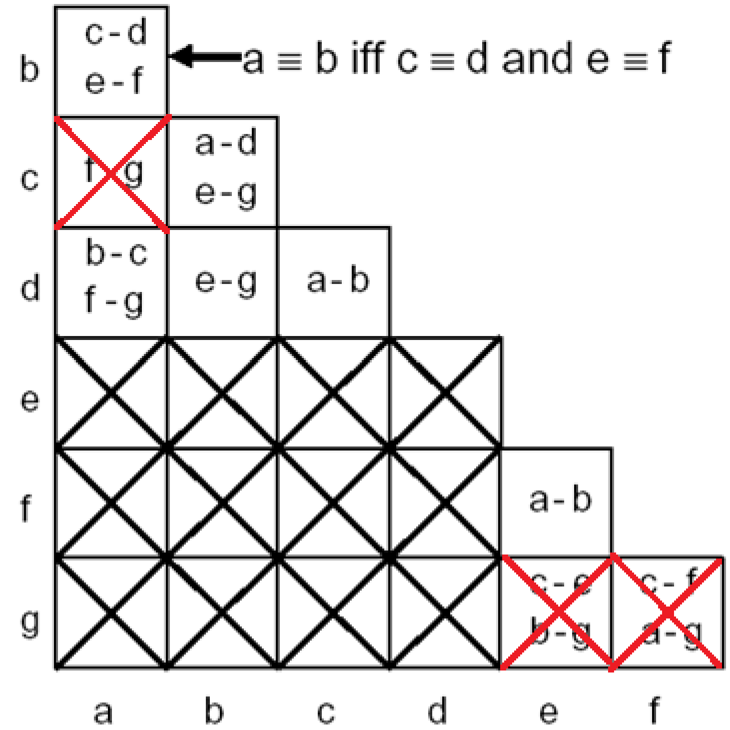
State Table Reduction

- Consider square (b,a) to be equivalent
  c == d && e == f. Can't determine
  remain implied pairs.
- Consider square (c,a) to be equivalent
  f == g. This is **not** true.
  f !== g since it now has an X in the
  (f,g) square.
- Place an X in the (c,a) square.



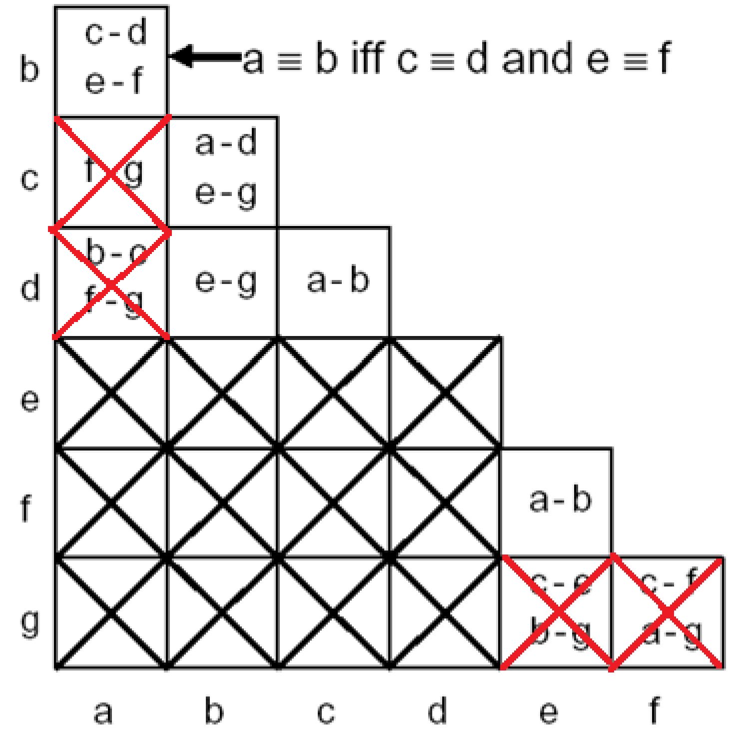
State Table Reduction

- Consider square (d,a) to be equivalent
  b==c && f == g. This is **not** true.
  f !== g since it now has an X in the
  (f,g) square.



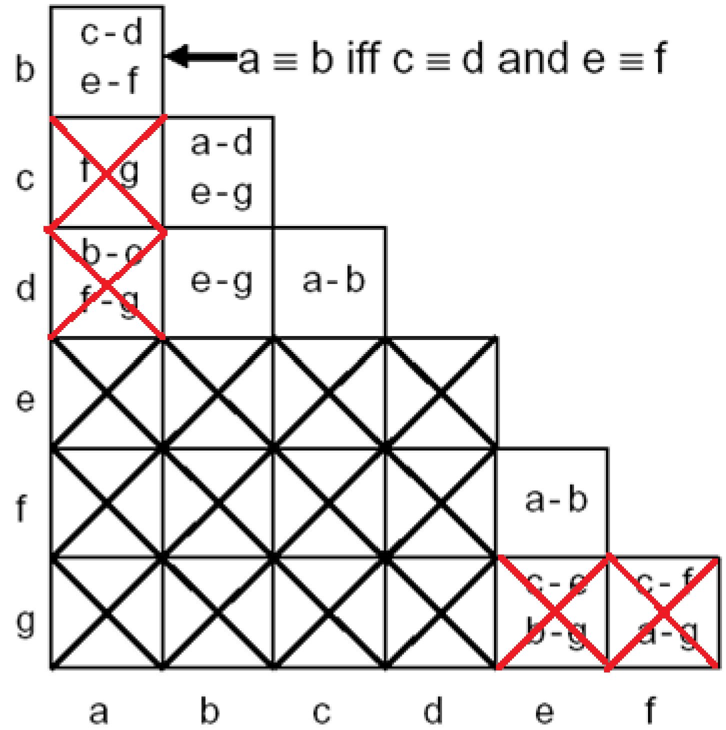
State Table Reduction

- Consider square (d,a) to be equivalent
  b==c && f == g. This is **not** true.
  f !== g since it now has an X in the
  (f,g) square.
- Place an X in the (d,a) square.



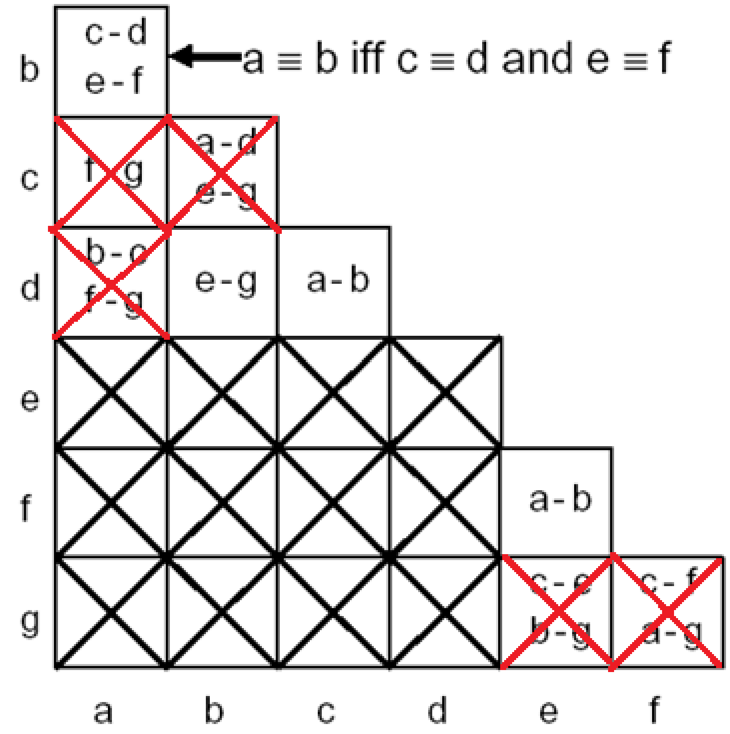
State Table Reduction

- Consider square (c,b) to be equivalent a==d && e == g. This is **not** true. a !== d and e !==g since they now both have an X in the (d,a) and (g,e) squares.



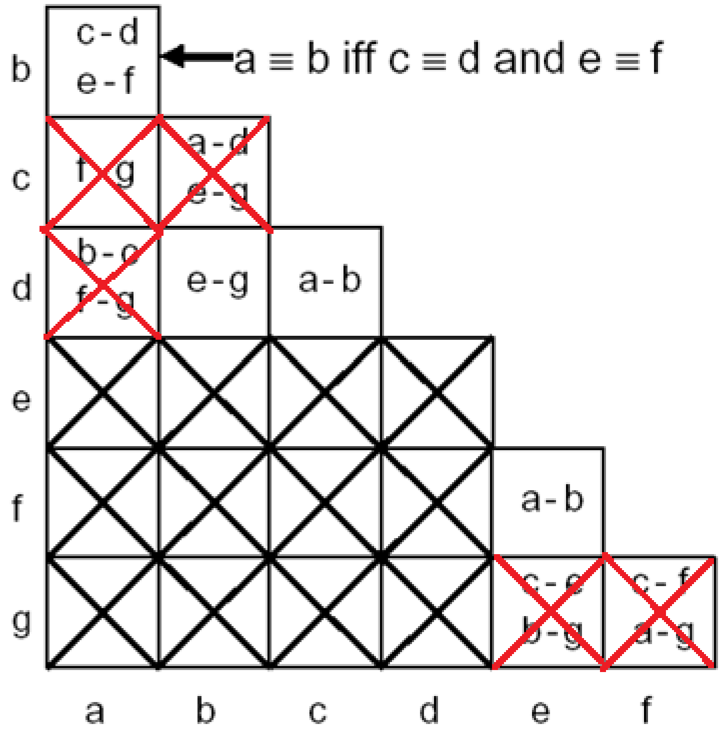
State Table Reduction

- Consider square (c,b) to be equivalent a==d && e == g. This is **<u>not</u>** true. a !== d and e !==g since they now both have an X in the (d,a) and (g,e) squares.
- Place an X in the (c,b) square.



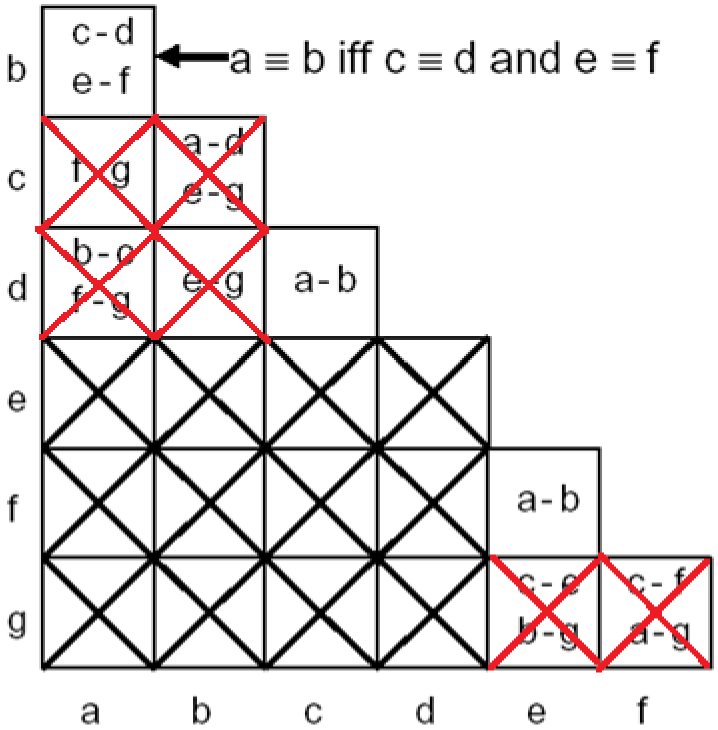
State Table Reduction

- Consider square (d,b) to be equivalent
  e == g. This is **not** true.
  e !==g since it now has
  an X in the (g,e) square.



State Table Reduction

- Consider square (d,b) to be equivalent
  e == g. This is **not** true.
  e !==g since it now has
  an X in the (g,e) square.
- Place an X in the (d,b) square.

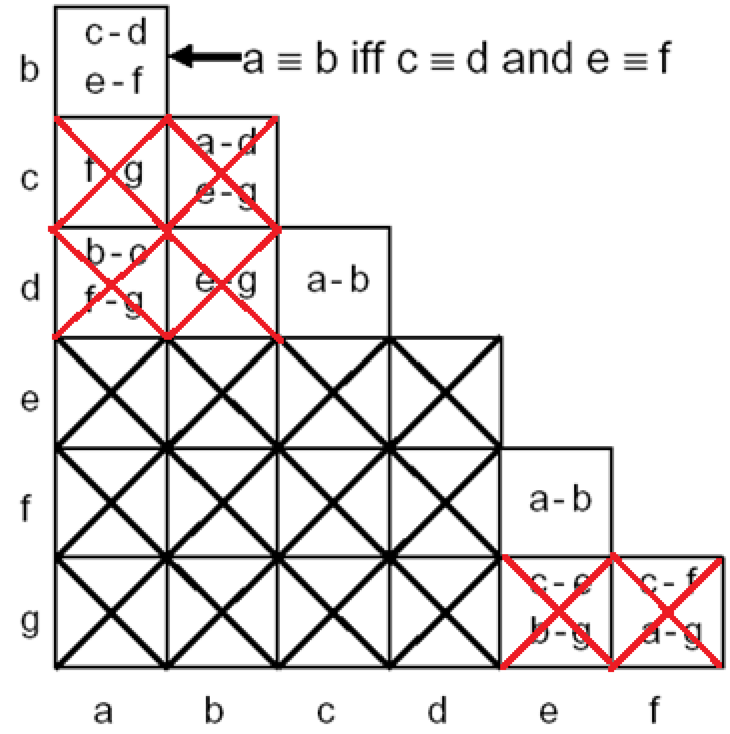


State Table Reduction

- Consider square (d,c) to be equivalent
  a == b Can't determine
  remain implied pairs.

- Consider square (f,e) to be equivalent
  a == b Can't determine
  remain implied pairs.

- Since at least one non-equivalence was found need to re-evaluate all other non-resolved implied pairs.



- Considering all the implied pair squares (b,a), (d,c) and (f,e) results in no new non-equivalences. Thus all non-equivalences have been found and the process stops.
- Remaining implied pairs are equivalent states.
  - (i.e. a==b, c==d, and e==f)

State Table Reduction

Present State	Next State X = 0	Next State X = 1	Present Output X = 0	Present Output X = 1
a	c	f	0	0
b	d	e	0	0
c	a	g	0	0
d	b	g	0	0
e	e	b	0	1
f	f	a	0	1
g	c	g	0	1

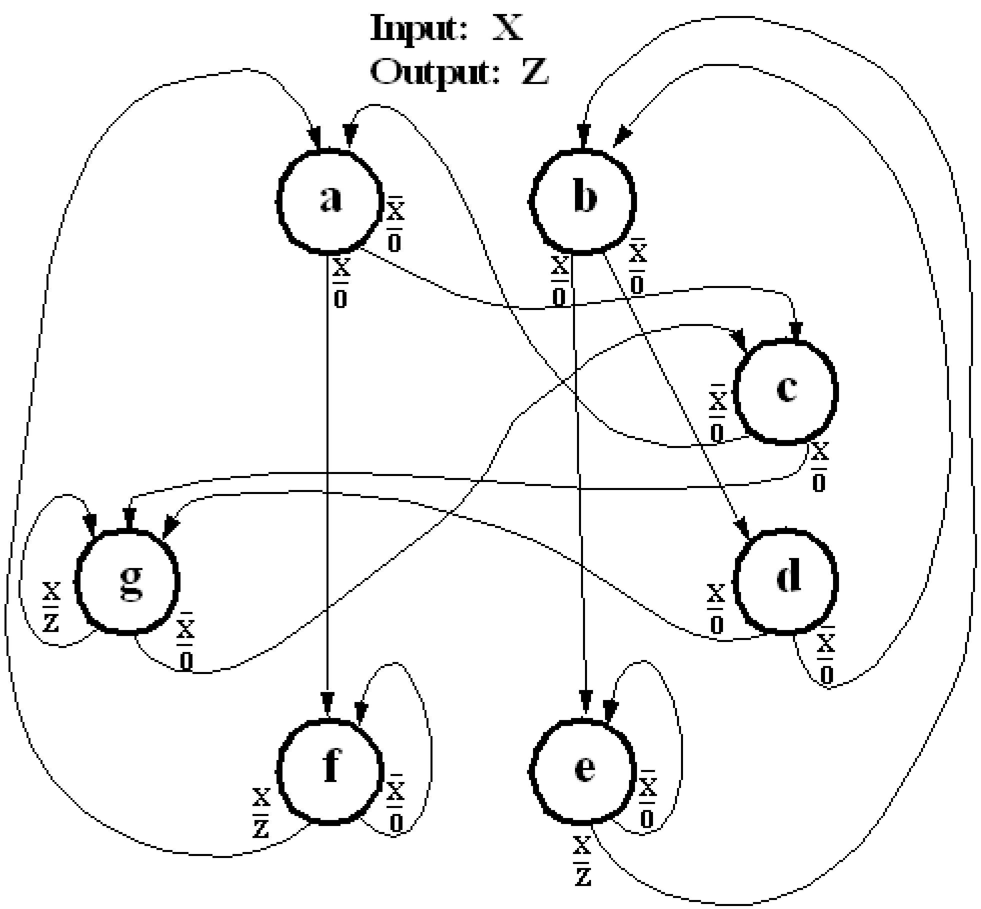
State Table

$a \equiv b$, $c \equiv d$, $e \equiv f$

Present State	X = 0	1	X = 0	1
a	c	e	0	0
c	a	g	0	0
e	e	a	0	1
g	c	g	0	1

Final Reduced Table

State Reduction Example

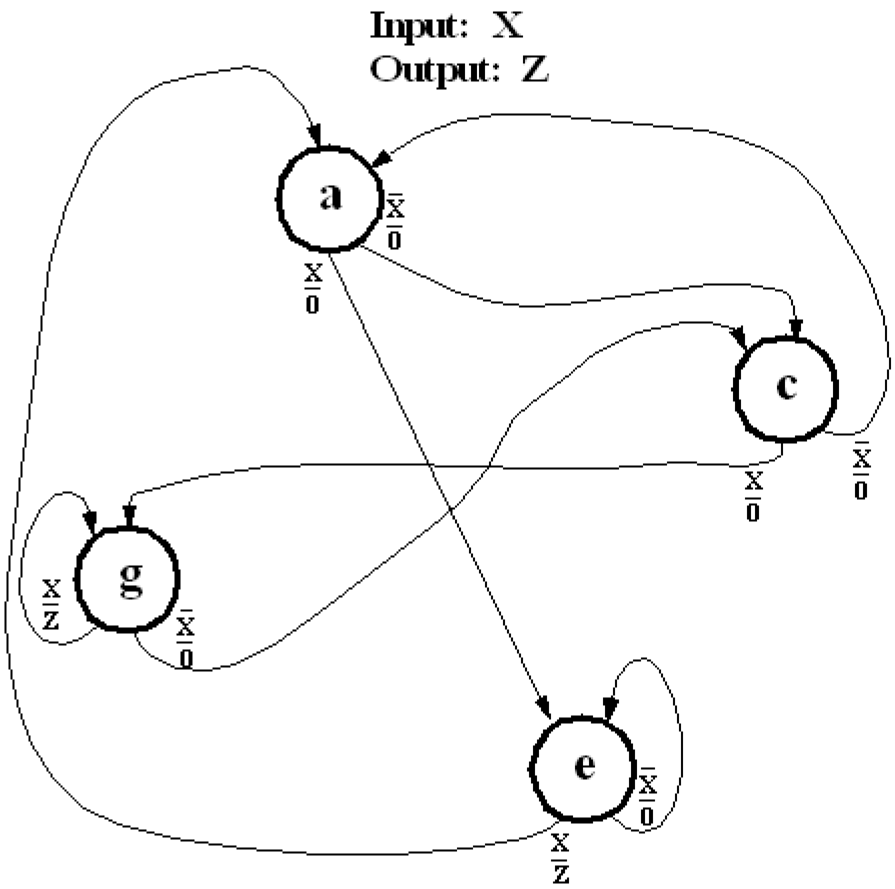


Extended State Transition Graph

Present State	Next State X = 0	Next State X = 1	Present Output X = 0	Present Output X = 1
a	c	f	0	0
b	d	e	0	0
c	a	g	0	0
d	b	g	0	0
e	e	b	0	1
f	f	a	0	1
g	c	g	0	1

State Table

State Reduction Example



Extended State Transition Graph
(Reduced Representation)

a ≡ b, c ≡ d, e ≡ f

Present State	X = 0	1	X = 0	1
a	c	e	0	0
c	a	g	0	0
e	e	a	0	1
g	c	g	0	1

Final Reduced Table

Implication Table Method

- 1. Construct a chart that contains a square
for each pair of states.
- 2. Compare each pair in the state table. If the outputs associated with states i and j are different, place an X in square i-j to indicate that i!=j.
If outputs are the same, place the implied pairs in square i-j.
If outputs and next states are the same (or i-j implies only itself), i==j.
- 3. Go through the implication table square by square.
If square i-j contains the implied pair m-n, and square m-n contains X, then i!=j, and place X in square i-j.
- 4. If any Xs were added in step 3, repeat step 3 until no more Xs are added.
- 5. For each square i-j that does not contain an X, i==j.