State Reduction

Equivalent States

- Two states are equivalent if- merging them does not change machines external behaviour.
- *Or*, two states are equivalent if-

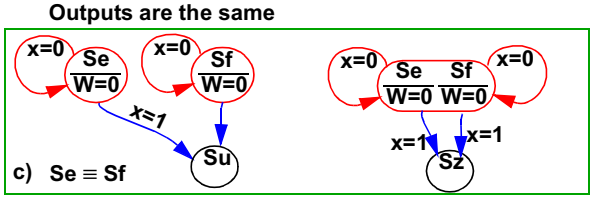
1. There outputs are equivalent.
2. Their next states are the same, or equivalent, or would be equivalent if the states were merged, for all inputs.

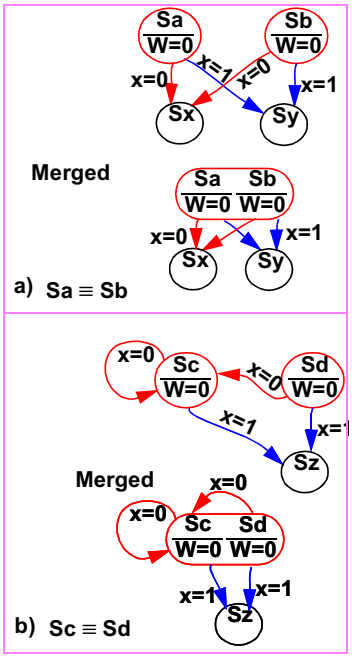
Examples a) and b)

Next states the same
Outputs the same

Example c)

Next states are equivalent if present states merged.
Outputs are the same





Equivalent states

If you take a finite state machine, put it in a green box, and look at only the outputs and inputs (including reset and clock). Then two states in the machine are equivalent if merging the states causes no change in what you can observe from watching all possible sequences of inputs and outputs.

An alternate definition which is equivalent

Two states are equivalent if their outputs and their next states are equivalent.

The tricky part:

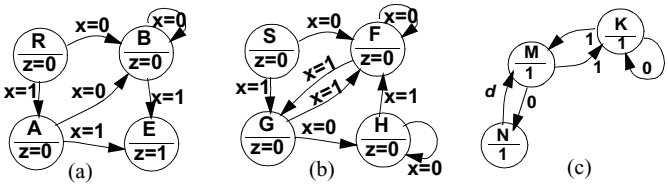
If merging the two states makes their next states equivalent, then the two states are equivalent

Note that the "same or equivalent" must be for all possible input combinations.

25.• PROBLEM

Which states are equivalent?

In a)? Explain.
In b)? Explain.
In c)? This one is hard. It was taken from the example on Slide 57.



A Machine With Excess States

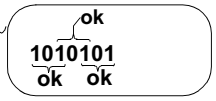
A "101" Sequence Detector With Overlap

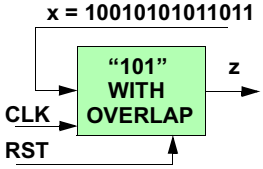
Specification

x=serial bit coming in at clock rate

z = 1 after sequence 101 is detected

Sequences may overlap





Step 1

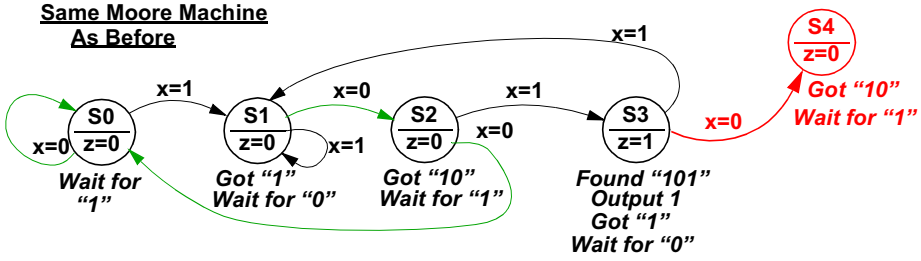
Block diagram, waveforms.

x	1	0	0	1	0	1	0	1	0	1	1	0	1	1	
z	-	0	0	0	0	0	1	0	1	0	1	0	0	1	0

Moore Output

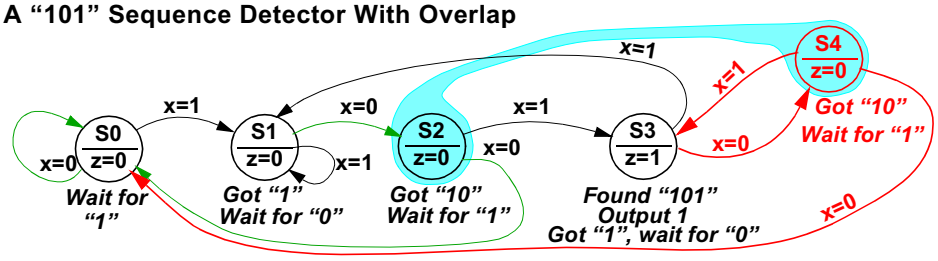
Same Moore Machine As Before

New to Catch Overlap



Merging States

A "101" Sequence Detector With Overlap



S2 and S4 Look like they can merge.

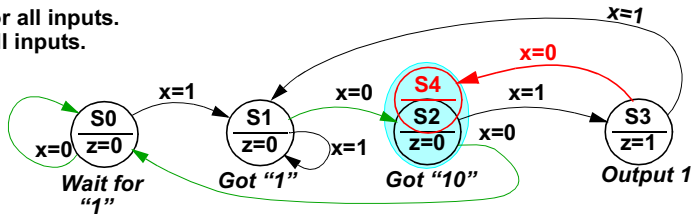
Can they?

Check

State	Next State x=0	Next State x=1	Output
S2	S0	S3	0
S4	S0	S3	0

Next states the same for all inputs.
Outputs the same for all inputs.

Merged State Graph



101 Machine

Merge Table or Staircase Diagram

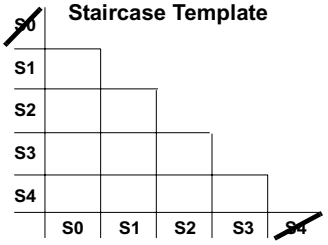
Overlapping 101 State Table

State	Next State x=0	Next State x=1	output z
S0	S0	S1	0
S1	S2	S1	0
S2	S0	S3	0
S3	S4	S1	1
S4	S0	S3	0

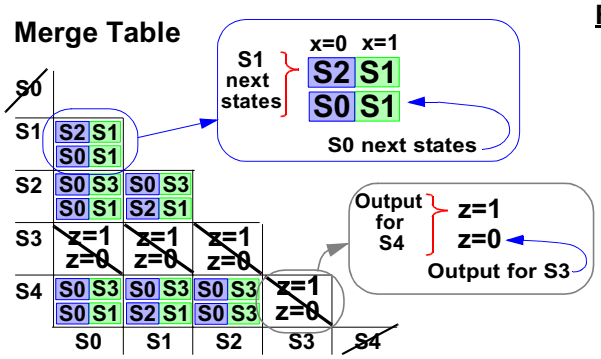
Make template
- Write states along axes
- Cross out top state S0
- Cross out right most state S4

Box for every possible merger.

Staircase Template



Merge Table



Fill in the boxes

For each intersection
i.e for each potential state merger.

1) Check outputs
If different ▨. Can't merge.

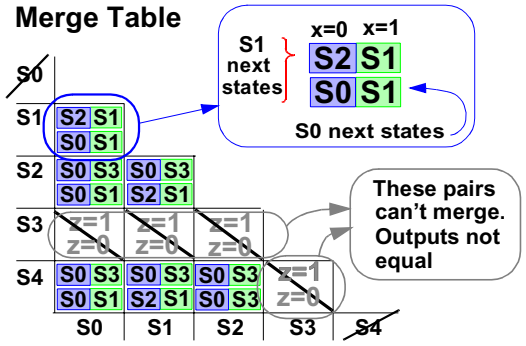
2) Fill in next state info.
Follow format shown

Left State's next states
Bottom State's next states

Merge Table for 101 Machine

Merge Table: Iteration 1

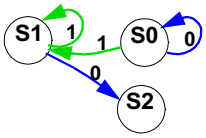
Merge Table

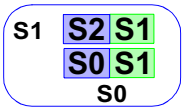


3. Step through Merge Table

Can S1 and S0 be merged?

Only if next states can be merged.



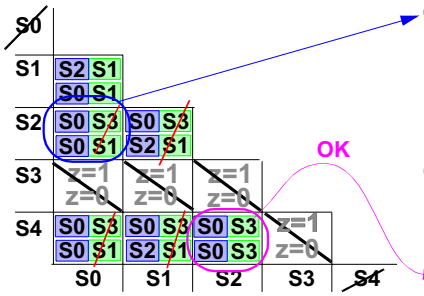


Can S1 merged with S1? (OK)

Can S2 merged with S0?

(Don't know yet, leave till later.)

Continue Stepping through Merge Table



Can S2 and S0 be merged?

Only if S0 can merge with S0. (OK)

and S3 can merge with S1.

(No! S1 and S3 have different outputs)

Strike out merger → S0 S3 / S0 S1

Continue

S2 and S1 can't merge. → S0 S3 / S2 S1

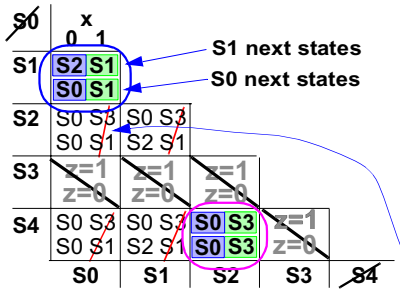
S4 and S0 can't merge. → S0 S3 / S0 S1

S4 and S1 can't merge. → S0 S3 / S2 S1

But clearly S4 and S2 can merge!

Merge Table: Iteration 2

Merge Table after 1 iteration

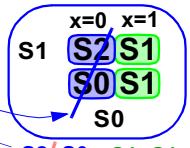


Review of last iteration

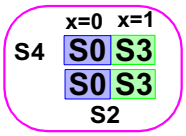
Scratched mergers with different outputs.
These can never be equivalent.
S3 output is different from S0, S1, S2

4) Iteration 2

Check all states that are not scratched.
Are next states still potentially mergable?



S2≡S0 S1≡S1
Scratch S1 and S0 merger



S0≡S0 S3≡S3
Still OK

Merged 101 Table

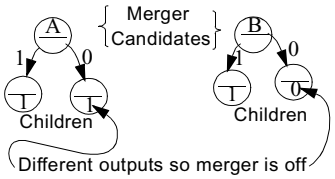
Merge Table: 2nd Iteration

The 0th iteration removes all potential mergers in which the states have different outputs.

The 1st iteration removes all potential mergers in which the children[1] of the two states have different outputs (for the same input). .

The 2nd iteration will eliminate potential mergers with conflicting outputs in the children of the children.

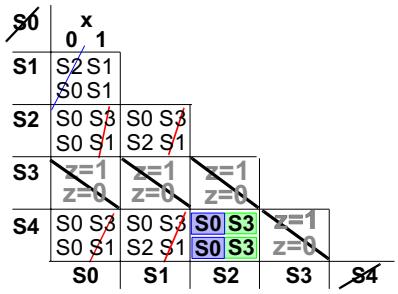
From this you might be able to infer that if a state machine has only one output for all states, (the output never changes), then it is equivalent to a one state machine.



1. The next states of a state are sometimes called the *children* of the state.

Merge Table: Iteration 4

Merge Table after 2 iterations



5) Iteration 3

Nothing can change

No need for more iterations

Conclude: Merge S4 and S2

6) Make a merged state table

Combine states S2 and S4 -> S24

Change S2 or S4 in old table to S24

Original State Table

State	Next State x=0	Next State x=1	output z
S0	S0	S1	0
S1	S2	S1	0
S2	S0	S3	0
S3	S4	S1	1
S4	S0	S3	0

Merged State Table

State Q	Next State Q^+ x=0	Next State Q^+ x=1	output z
S0	S0	S1	0
S1	S24	S1	0
S24	S0	S3	0
S3	S24	S1	1
~~S4~~			

New merged state

Merge Table: Final Iteration

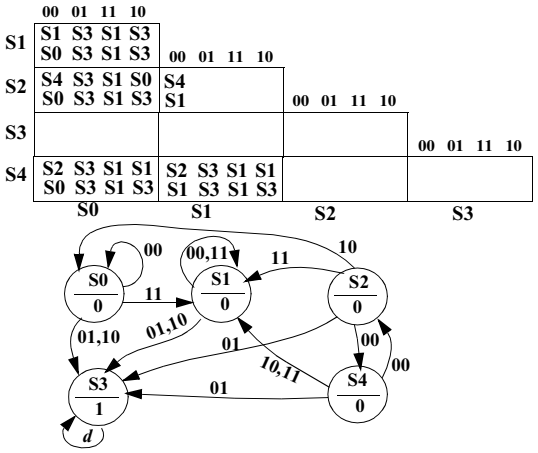
Merge Table for a two-input circuits

25.A•PROBLEM

a) Complete the merge table for the two-input state graph shown.

b) Use the merge table to show which potential mergers can be eliminated.

c) Merge the states and construct the state table for the revised machine. (3 states)



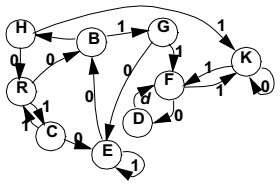
State Reduction With a Merge Table

A More Complex Table With Mealy Outputs

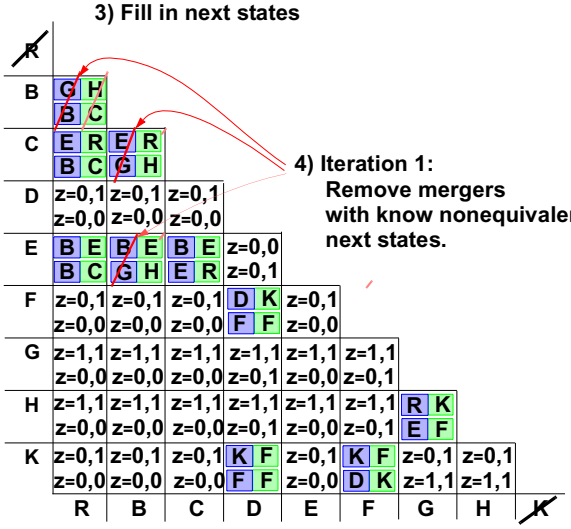
Find the Reduced State Table

Mealy State Table

State	Next State		output z	
	x=0	x=1	x=0	x=1
R	B	C	0	0
B	G	H	0	0
C	E	R	0	0
D	F	F	0	1
E	B	E	0	0
F	D	K	0	1
G	E	F	1	1
H	R	K	1	1
K	K	F	0	1



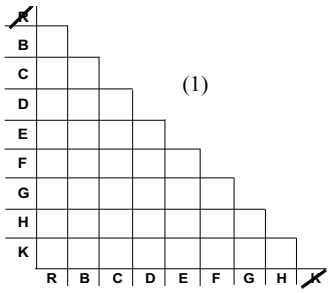
1) Build Staircase
2) Remove mergers with conflicting outputs
3) Fill in next states



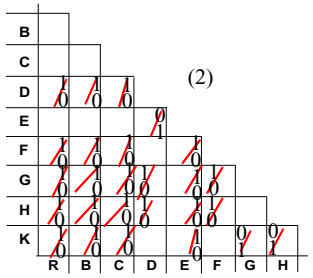
4) Iteration 1: Remove mergers with know nonequivalent next states.

More Complex Merged State Table

1. Draw the staircase. Besure to cross out the states ate the end of the axes



2. Cross out all squares where the outputs differ between the row and column states. States are not mergeable if there is any difference in their outputs.
   H outputs are only compatable with G.
   D, F and K are only compatable with each other.



3. In the squares that are left, fill in the next states for each row (top of square), and each column (bottom of square). See slide above.

4. Start the 1st iteration.

State Reduction With a Merge Table(Cont)

- Iteration 2: take out C-R and E-C

	R	B	C	D	E	F	G	H	K
B	G H B C								
C	E R B C	E R G H							
D	z=0,1 z=0,0	z=0,1 z=0,0	z=0,1 z=0,0						
E	B E B C	B E G H	B E E R	z=0,0 z=0,1					
F	z=0,1 z=0,0	z=0,1 z=0,0	z=0,1 z=0,0	D K F F	z=0,1 z=0,0				
G	z=1,1 z=0,0	z=1,1 z=0,0	z=1,1 z=0,0	z=1,1 z=0,1	z=1,1 z=0,0	z=1,1 z=0,1			
H	z=1,1 z=0,0	z=1,1 z=0,0	z=1,1 z=0,0	z=1,1 z=0,1	z=1,1 z=0,0	z=1,1 z=0,1	R K E F		
K	z=0,1 z=0,0	z=0,1 z=0,0	z=0,1 z=0,0	K F F F	z=0,1 z=0,0	K F D K	z=0,1 z=1,1	z=0,1 z=1,1	

- Iteration 3: take out E-R

because E-C was taken out last iteration.

	R	B	C	D	E	F	G	H	K
B	G H B C								
C	E R B C	E R G H							
D									
E	B E B C	B E G H	B E E R						
F				D K F F					
G									
H								R K E F	
K				K F F F		K F D K			

Iteration 4: take out G-H because E-R was just removed

F can merge with D
D can merge with K
K can merge with F

Do a 3-way merge D-F-K

Merged State Table

Iteration 2

C-R cannot merge unless E and B can merge. But E-B merge was crossed out on the 1st iteration.
E-C cannot merge unless B and E can merge. But E-B merge was crossed out on the 1st iteration.

Iteration 3

E-R cannot merge unless E and C can merge. But E-C merge was crossed out on the 2nd iteration.

Iteration 4

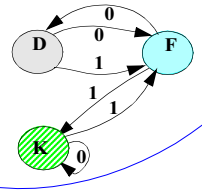
G-H cannot merge unless E and R can merge. But E-R merge was crossed out on the 3rd iteration

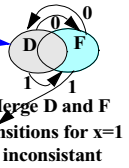
.

The 3-way Merge

F and D might merge if F and D are merged.
This is self fulfulling.
F and D might merge if K and F are merged.
If D and F alone are merged
The "0" transition arrows both go to the new DF state. OK!
The "1" transition arrows go different places. NO GOOD!
The one "1" transitions are OK only if F and K are also merged.

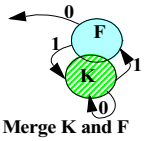
	D	F
F	D K	F F
D		





Merge D and F transitions for x=1 inconsistant

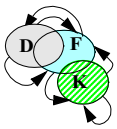
Similarly merging F and K alone gives inconsistant x=0 transitions.



Merge K and F transitions for x=0 inconsistant

Must merge all three states.
Then all transitions go to the one merged state.
This is consistant.

Merge K, D and F transitions consistant

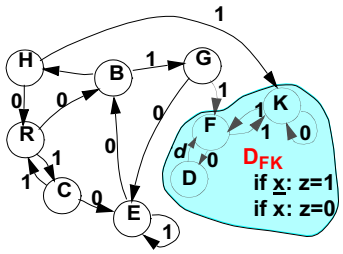


State Reduction With a Merge Table(Cont²)

- **Merging the Table**
  - Make new state D_{FK}
  - Replace state D with D_{FK}.
  - Remove states F and K.
  - In the next state columns replace D,F, or K with D_{FK}.

Continue on with the next job -

State Assignment.



Merged State Table

State	Next State		output z	
	x=0	x=1	x=0	x=1
R	B	C	0	0
B	G	H	0	0
C	E	R	0	0
D_{FK}	D_{FK}	D_{FK}	0	1
E	B	E	0	0
F				
G	E	D_{FK}	1	1
H	R	D_{FK}	1	1
K				

Merge Table

26.• PROBLEM

Design a Moore machine which will give out a 1 whenever it receives the input sequence 010 or 110. Overlapped sequences should be detected.

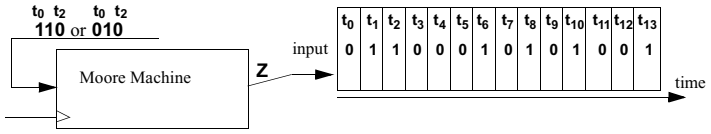
a) Fill in the outputs for the input sequence given in the figure. Time t_0 comes before t_1 etc.

b) Design the state graph. Most people will get 7 states.

c) Check for equivalent states. Show the merge (staircase) table. (If you have a 21 square staircase, 10 squares will have incompatable outputs, another 8 should have incompatable next states)

d) If state A merges with state B, call the merged state AB. Draw the new reduced 4-state graph.

e) Do **not** draw K-maps or find the circuit.



Summary:

Merging a Mealy table.

Mergers are not obvious.

Several iterations may be needed to find all incompatable next-states.