L10 – State Machine Design Topics

States Machine Design

- Other topics on state machine design
  - Equivalent sequential machines
  - Incompletely specified machines
  - One Hot State Machines

- Ref: text Unit 15.4, 15.5, 15.8

Copyright 2012 - Joanne DeGroat, ECE, OSU

Equivalent State Machines

- So far have seen that equivalent states in the state table of a sequential machine are equivalent and can be removed.
- How about the equivalence between two sequential circuits?
  - *Two sequential circuits are equivalent if they are capable of doing the same work.*

Copyright 2012 - Joanne DeGroat, ECE, OSU

Formally

- Definition 15.2
  - Sequential circuit N_1 is equivalent to sequential circuit N_2 if for each state p in N_1, there is a state q in N_2 such that $p \equiv q$, and conversely, for each state s in N_2 there is a state t in N_1 such that $s \equiv t$.
  - Simply said they have the same states which can be seen if the circuit is small enough.
- An implication table can be used to prove this.

Copyright 2012 - Joanne DeGroat, ECE, OSU

An example

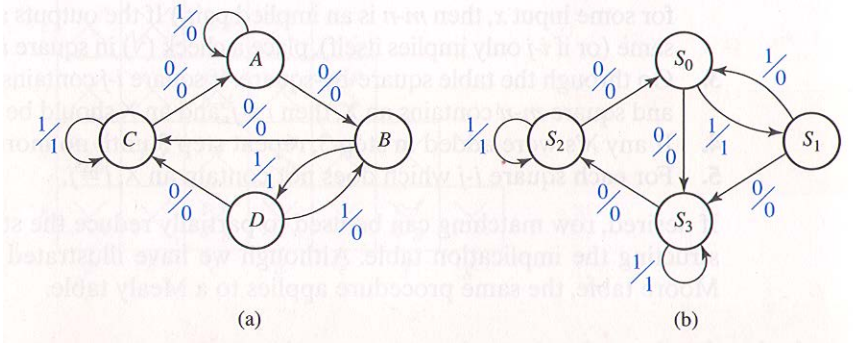
- State tables and state graphs of two sequential machines. Figure 15-6 from the text.
- Equivalent?

N_1

	$X=0$	1	$X=0$	1
A	B	A	0	0
B	C	D	0	1
C	A	C	0	1
D	C	B	0	0

N_2

	$X=0$	1	$X=0$	1
S_0	S_3	S_1	0	1
S_1	S_3	S_0	0	0
S_2	S_0	S_2	0	0
S_3	S_2	S_3	0	1



Copyright 2012 - Joanne DeGroat, ECE, OSU

Proving equivalence

- Again will use an implication table.
  - Only this time, it is the full square.
  - Along bottom are the states of one machine
  - Along the side are the states of the second.
- Start by removing output incompatible states.

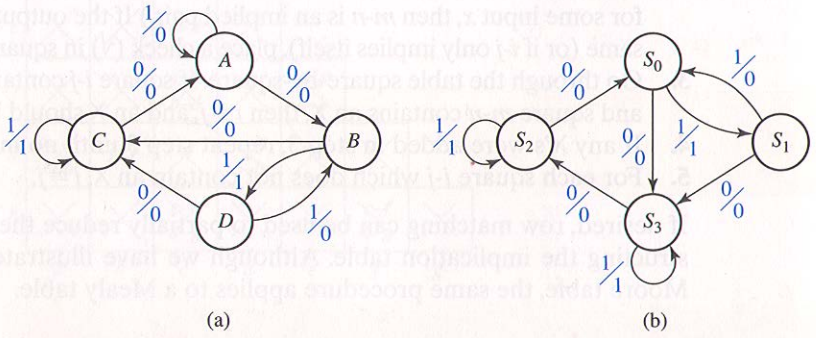
Copyright 2012 - Joanne DeGroat, ECE, OSU

The equivalence implication table

- X squares where the outputs are incompatible
- Enter implied equivalence pairs for remaining states.

	N_1			
	$X=0$	1	$X=0$	1
A	B	A	0	0
B	C	D	0	1
C	A	C	0	1
D	C	B	0	0

	N_2			
	$X=0$	1	$X=0$	1
S_0	S_3	S_1	0	1
S_1	S_3	S_0	0	0
S_2	S_0	S_2	0	0
S_3	S_2	S_3	0	1



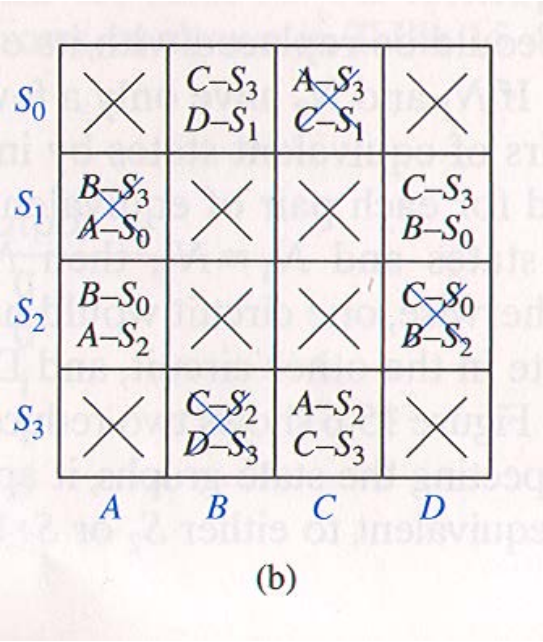
	A	B	C	D
S_0	X	$C–S_3$ $D–S_1$	$A–S_3$ $C–S_1$	X
S_1	$B–S_3$ $A–S_0$	X	X	$C–S_3$ $B–S_0$
S_2	$B–S_0$ $A–S_2$	X	X	$C–S_0$ $B–S_2$
S_3	X	$C–S_2$ $D–S_3$	$A–S_2$ $C–S_3$	X

(a)

Copyright 2012 - Joanne DeGroat, ECE, OSU

Step 2

- Go back through and remove the implied equivalence pairs that were Xed on the first pass. Continue until no further Xs are entered.
- If there is one square not Xed in each row and each column, the state machines are equivalent. (When both are minimal)
- Consider problem 15-17 in text Does this work if the state tables are of different size?



Copyright 2012 - Joanne DeGroat, ECE, OSU

Problem 15.17

- The problem statement

15.17 Circuits N and M have the state tables that follow.

(a) Without first reducing the tables, determine whether circuits N and M are equivalent.

(b) Reduce each table to a minimum number of states, and then show that N is equivalent to M by inspecting the reduced tables.

M

	$X=0$	1	
S_0	S_3	S_1	0
S_1	S_0	S_1	0
S_2	S_0	S_2	1
S_3	S_0	S_3	1

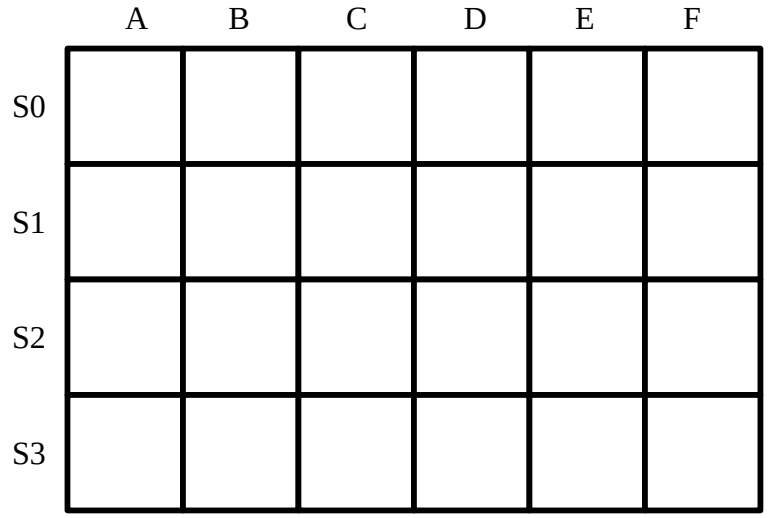
N

	$X=0$	1	
A	E	A	1
B	F	B	1
C	E	D	0
D	E	C	0
E	B	D	0
F	B	C	0

Copyright 2012 - Joanne DeGroat, ECE, OSU

Problem 15.17

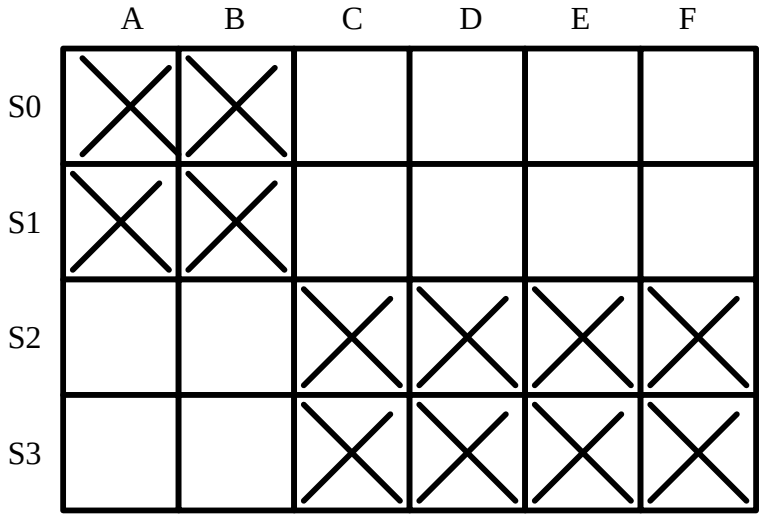
- Can be Worked on board
- Or here in the slides
  - Start with an equivalence implication table



Copyright 2012 - Joanne DeGroat, ECE, OSU

Output compatible

- Go through and X output incompatible states



Copyright 2012 - Joanne DeGroat, ECE, OSU

Next State

- Fill in the next state pairs on the table

	A	B	C	D	E	F
S0	X	X	S3 – E S1 – D	S3 – E S1 – C	S3 – B S1 – D	S3 – B S1 – C
S1	X	X	S0 – E S1 – D	S0 – E S1 – C	S0 – B S1 – D	S0 – B S1 – C
S2	S0 – E S2 - A	S0 – F S2 – B	X	X	X	X
S3	S0 – E S3 - A	S0 – F S3 – B	X	X	X	X

Copyright 2012 - Joanne DeGroat, ECE, OSU

1st Pass through table

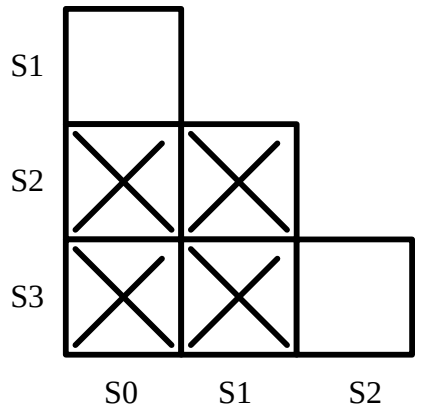
- Check implied pairs (S3-E)x (S0-B)x
  - Remainder are compatible. It seems machine has redundant states

	A	B	C	D	E	F
S0	X	X	~~S3 – E~~ S1 – D	~~S3 – E~~ S1 – C	S3 – B S1 – D	S3 – B S1 – C
S1	X	X	S0 – E S1 – D	S0 – E S1 – C	~~S0 – B~~ S1 – D	~~S0 – B~~ S1 – C
S2	S0 – E S2 - A	S0 – F S2 – B	X	X	X	X
S3	S0 – E S3 - A	S0 – F S3 – B	X	X	X	X

Copyright 2012 - Joanne DeGroat, ECE, OSU

Minimize both machines?

- Start with the Sx machine – can it be minimized? If so, what are implications?



15.17 Circuits N and M have the state tables that follow.

(a) Without first reducing the tables, determine whether circuits N and M are equivalent.

(b) Reduce each table to a minimum number of states, and then show that N is equivalent to M by inspecting the reduced tables.

M

	$X=0$	1	
S_0	S_3	S_1	0
S_1	S_0	S_1	0
S_2	S_0	S_2	1
S_3	S_0	S_3	1

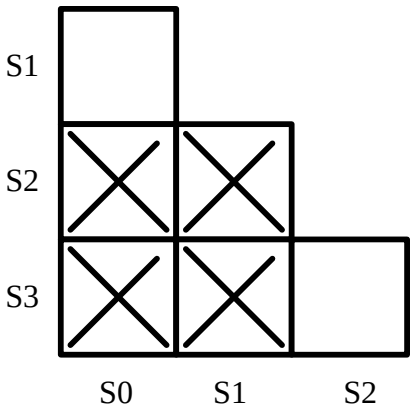
N

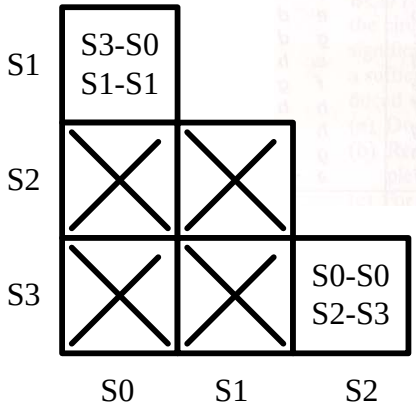
	$X=0$	1	
A	E	A	1
B	F	B	1
C	E	D	0
D	E	C	0
E	B	D	0
F	B	C	0

Copyright 2012 - Joanne DeGroat, ECE, OSU

Minimize both machines

- Implied Next States





15.17 Circuits N and M have the state tables that follow.

(a) Without first reducing the tables, determine whether circuits N and M are equivalent.

(b) Reduce each table to a minimum number of states, and then show that N is equivalent to M by inspecting the reduced tables.

M

	$X=0$	1	
S_0	S_3	S_1	0
S_1	S_0	S_1	0
S_2	S_0	S_2	1
S_3	S_0	S_3	1

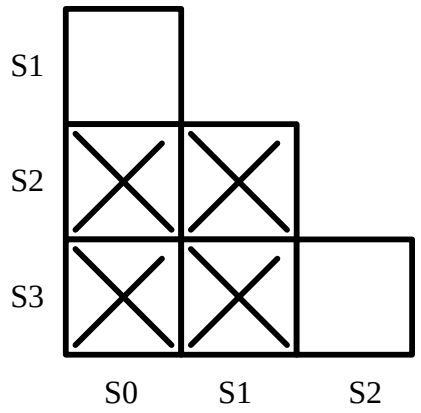
N

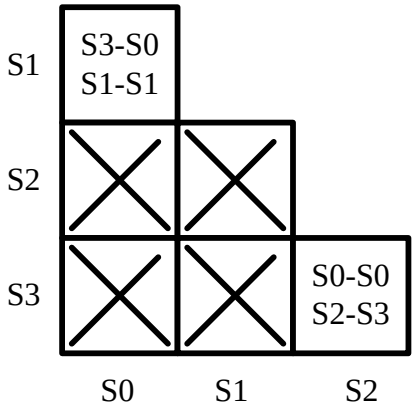
	$X=0$	1	
A	E	A	1
B	F	B	1
C	E	D	0
D	E	C	0
E	B	D	0
F	B	C	0

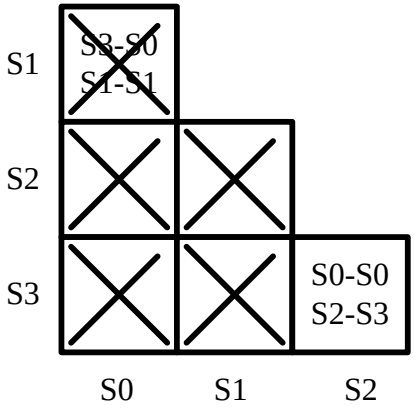
Copyright 2012 - Joanne DeGroat, ECE, OSU

Minimize both machines

- Can it be reduced?



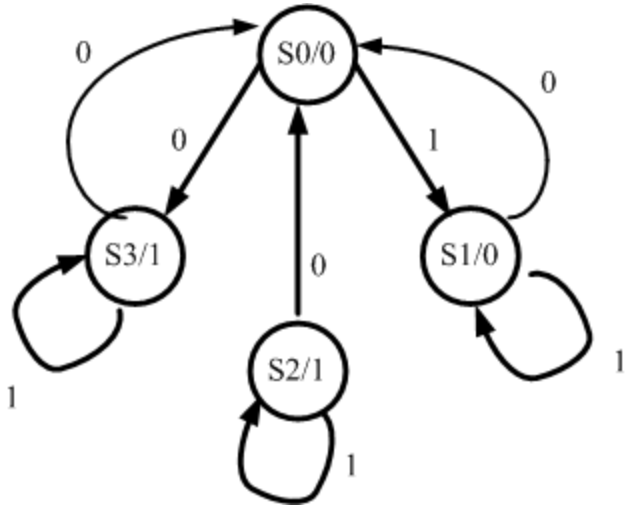




Copyright 2012 - Joanne DeGroat, ECE, OSU

Reduced machine

- Can be seen from state graph
- States S2 and S3 are equivalent – in fact S2 is not reachable unless the machine comes up in that state at startup and it can never reach S2 again.



Copyright 2012 - Joanne DeGroat, ECE, OSU

Now for the A,B,..,E machine

- Start with incompatible outputs

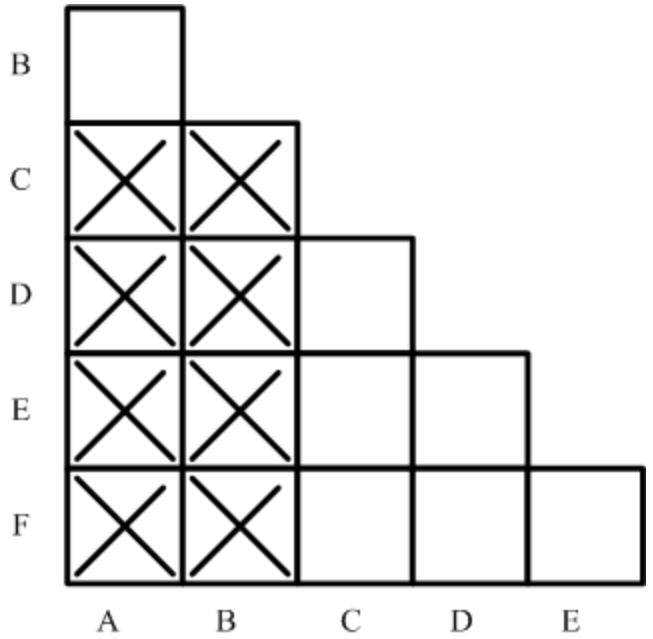
15.17 Circuits N and M have the state tables that follow.

(a) Without first reducing the tables, determine whether circuits N and M are equivalent.

(b) Reduce each table to a minimum number of states, and then show that N is equivalent to M by inspecting the reduced tables.

	M		
	$X = 0$	1	
S_0	S_3	S_1	0
S_1	S_0	S_1	0
S_2	S_0	S_2	1
S_3	S_0	S_3	1

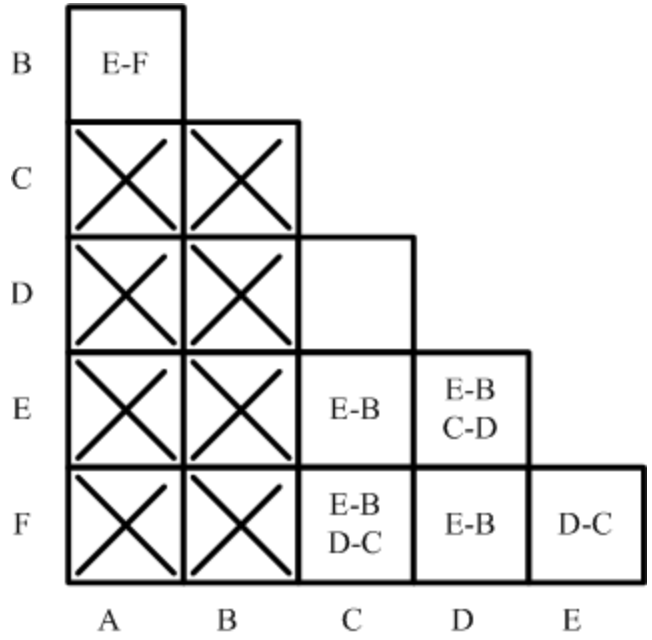
	N		
	$X = 0$	1	
A	E	A	1
B	F	B	1
C	E	D	0
D	E	C	0
E	B	D	0
F	B	C	0

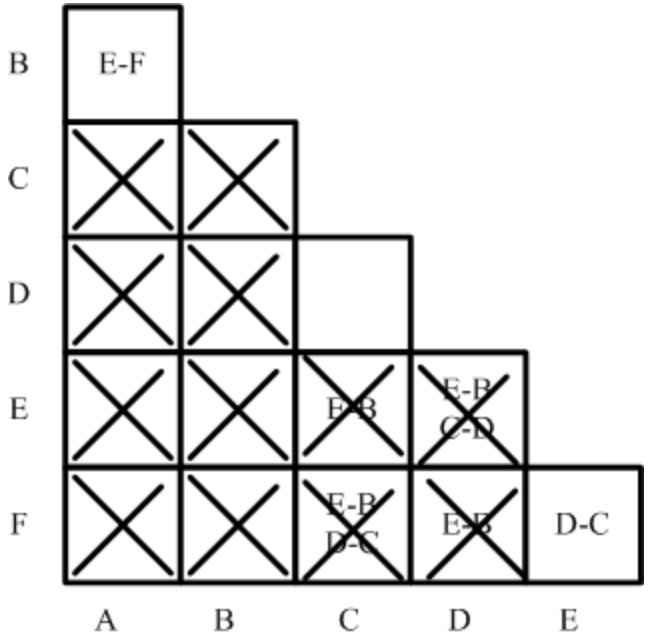


Copyright 2012 - Joanne DeGroat, ECE, OSU

And then implied next state

- And run though algorithm





Copyright 2012 - Joanne DeGroat, ECE, OSU

Result is reduced state table

- Find that
  - D ≡ C
  - E ≡ F
  - A ≡ B
- So state table reduces to AND for Sx version

	X=0	X=1	OUTPUT
A	E	A	1
C	E	C	0
E	A	C	0

	X=0	X=1	OUTPUT
S3	S0	S3	1
S1	S0	S1	0
S0	S3	S1	0

Copyright 2012 - Joanne DeGroat, ECE, OSU

Significance

- Now consider the equivalence implication table. What is the implication if S2 replaces state S2 and S3?

	A	B	C	D	E	F
S0	X	X	~~S3 – E~~ ~~S1 – D~~	~~S3 – E~~ ~~S1 – C~~	S3 – B S1 – D	S3 – B S1 – C
S1	X	X	S0 – E S1 – D	S0 – E S1 – C	~~S0 – B~~ ~~S1 – D~~	~~S0 – B~~ ~~S1 – C~~
S2	S0 – E S2 - A	S0 – F S2 – B	X	X	X	X
S3	S0 – E S3 - A	S0 – F S3 – B	X	X	X	X

Copyright 2012 - Joanne DeGroat, ECE, OSU

Incompletely Specified

- Incompletely Specified State Tables
  - State tables that contain don't cares.
  - Results in reduced logic

- Determining the best way to fill in the don't cares is another of the *n-p complete* problems.
- For this course do the most logical approach.

Copyright 2012 - Joanne DeGroat, ECE, OSU

One Hot

- CPLDs and FPGAs have a good number of F/Fs onboard. The F/Fs are there whether they are used or not, so a circuit with the minimum number of F/Fs is not the ultimate objective.
- For these devices the objective is to reduce the total number of logic cells used and the interconnection between cells.
- One hot encoding is one approach to have shorter signal paths and reduce logic cells.

Copyright 2012 - Joanne DeGroat, ECE, OSU

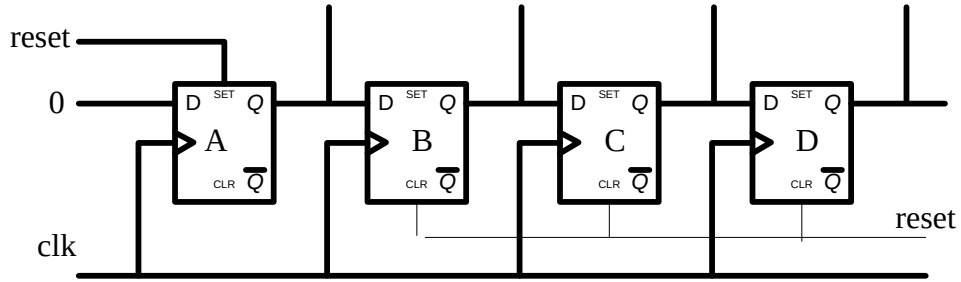
What is one hot?

- One hot is a method where a flip flop is used for each state in the state machine. A state machine with *n* states will require *n* flip flops in its realization.
- One hot realization is excellent for controllers that step through a set sequence of linear steps.
- Text gives example of a multiplier controller state graph which is not linear.

Copyright 2012 - Joanne DeGroat, ECE, OSU

Linear one hot

- Linear one hot sequential controllers requires no next state logic.
- On Reset the output of the 4 F/F is 1000
  - On clocks 0100, then 0010, then 0001, then 0000



Copyright 2012 - Joanne DeGroat, ECE, OSU

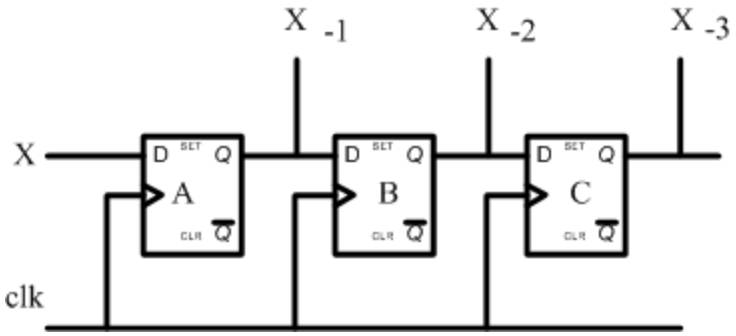
One hot use

- Have been use in such things as
  - Successive approximation A-D converters
  - Various automotive control systems
  - Automated machinery control systems
- Also commonly used in processor controllers
  - Process controller states
    - F1,F2,F3,F4,F5,F6,F7,F8 always followed by
    - E0,E1,E2,E3,E4,E5,E6,E7 if direct addressing
    - E20,E21,E22,E23,E24,E25,E26,E27 if indirect

Copyright 2012 - Joanne DeGroat, ECE, OSU

One hot application

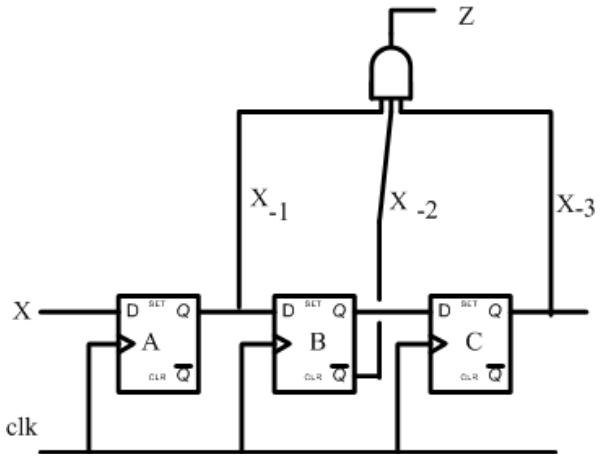
- One hot could have been used in the sequence detector problems
  - Detect an input sequence ending in 101.
  - Construct a shift register that holds the last 3 inputs of an input X.



Copyright 2012 - Joanne DeGroat, ECE, OSU

The full circuit

- Desire Z=1 when X_{-1} X_{-2} X_{-3} is 101.
- Simply construct the combinational logic with inputs from the F/F outputs.



Copyright 2012 - Joanne DeGroat, ECE, OSU

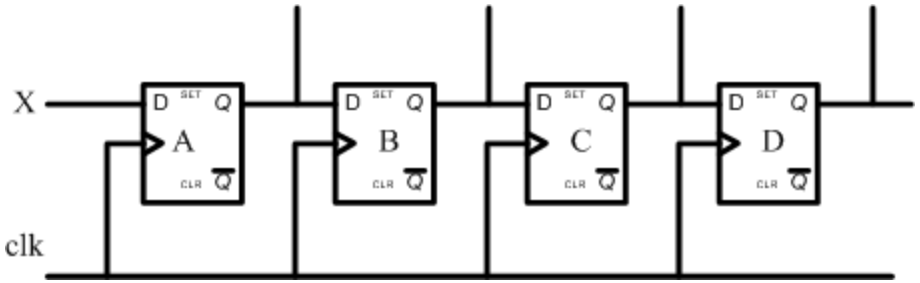
Compare the gates

- Traditional implementation for sequence detector (from text)
  - 2 F/Fs
  - 2 2-input AND gates
  - 1 INV
- One hot implementation
  - 3 F/Fs
  - 1 3-input AND gate

Copyright 2012 - Joanne DeGroat, ECE, OSU

Another example

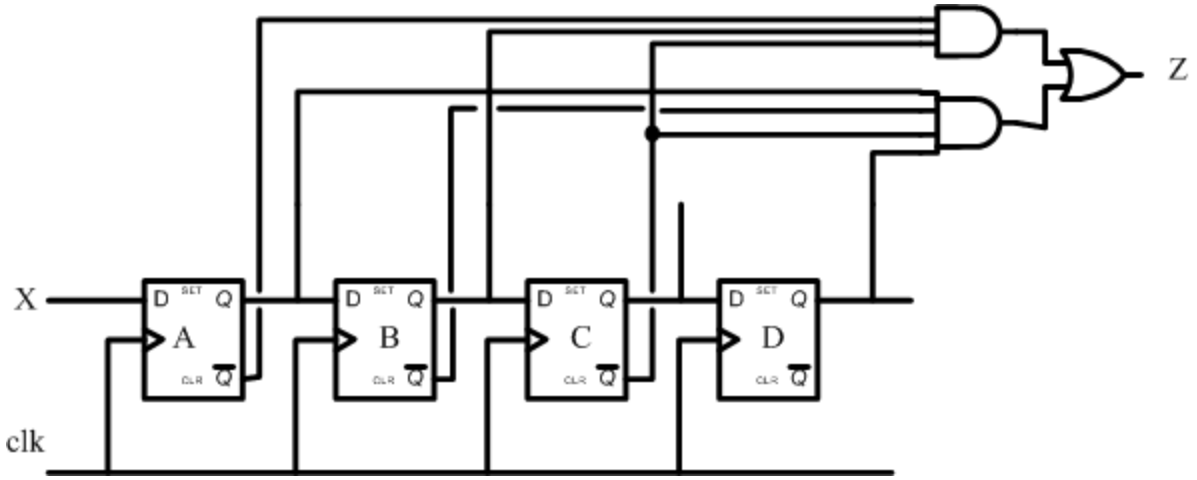
- Design a sequence detector that detects input sequences ending in 010 or 1001. Z = 1 when a sequence is detected.
- Start with a 4 bit shift register to hold the last 4 inputs.



Copyright 2012 - Joanne DeGroat, ECE, OSU

Now add Z generation logic

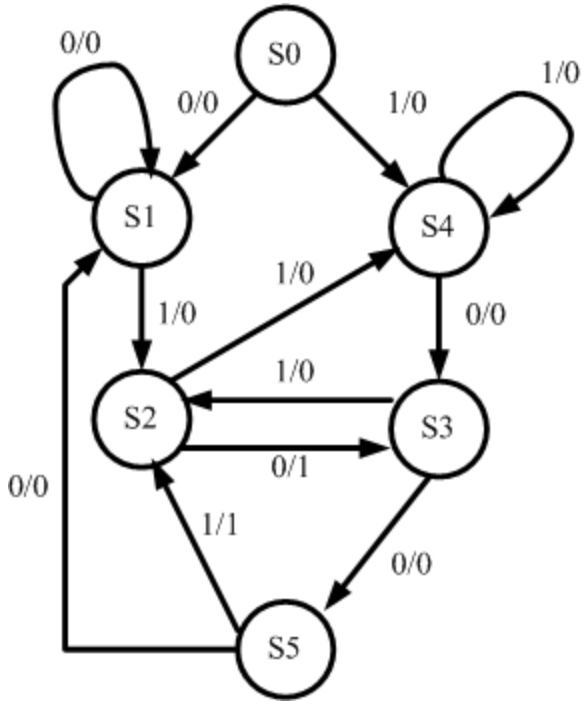
- Construct the combinational logic for Z



Copyright 2012 - Joanne DeGroat, ECE, OSU

Implementation comparison

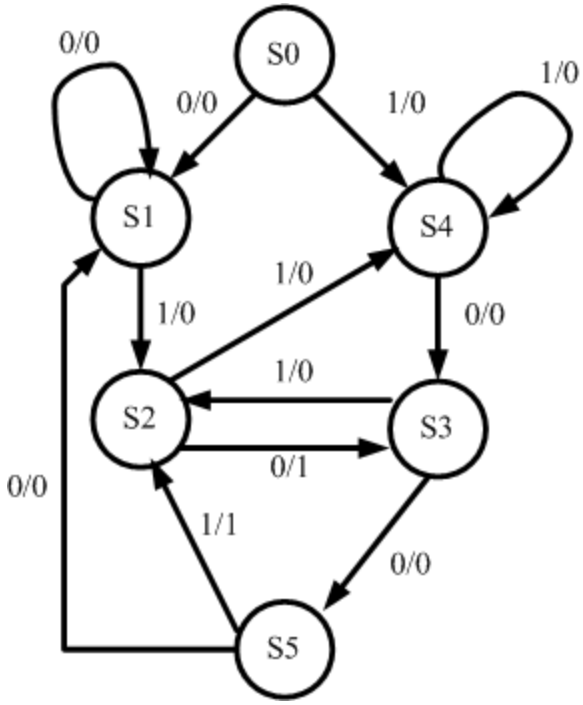
- Traditional
  - 3 F/Fs
  - Need to work problem
  - more than 1 hot
- One hot
  - 4 F/Fs
  - 2 AND gates (1- 3 inp, 1- 4 inp)
  - 1 OR gate (2 inp)



Copyright 2012 - Joanne DeGroat, ECE, OSU

The state table

- For the Mealy Machine



	NEXT STATE		OUTPUT	
Present State	**X=0**	**X=1**	**X=0**	**X=1**
S0	S1	S4	0	0
S1	S1	S2	0	0
S2	S3	S4	1	0
S3	S5	S2	0	0
S4	S3	S4	0	0
S5	S1	S2	0	1

Copyright 2012 - Joanne DeGroat, ECE, OSU

Comparison

- This was worked to gates in lect 9.
  - 3 D F/Fs
  - 2 – 3 input AND gates
  - 3 – 2 input AND gates
  - 2 – 2 input OR gates
  - 1 – 3 input OR gate
- Versus one hot
  - 4 F/Fs
  - 1 – 3 input AND gate
  - 1 – 4 input AND gate
  - 1 – 2 input OR
- Comments from class??????

Copyright 2012 - Joanne DeGroat, ECE, OSU

Lecture summary

- Have looked at state machine equivalence.
- Incompletely specified machine implicaiton.
- One hot encoding and how it may not be all that bad an alternative.

Copyright 2012 - Joanne DeGroat, ECE, OSU