CS47 - Lecture 16

Kaushik Patra
(kaushik.patra@sjsu.edu)

1

- Combinational Circuit Design
- Sequential Circuit Design

[Chapter 5 of Logic & Computer Design Fundamentals, 4th Edition, M. Morris Mano, Charles R. Kime]

Combinational Circuit Design ...

2

Steps of Logic Design

- Write the specification.
- Formulate the function in Truth table.
- Optimize the Boolean function into reduce form
- Map the function into available logic gates.
- Verify the logic circuit.

3

- Writing down specification of the target circuit functionality is very important. Clear specification will identify many subtle helpful clues, like don't care terms.
- Once the specification is available, truth table of the logic function must be constructed.
- The Boolean logic function is reduced and optimized using the Boolean algebra technique and K-Map (any may other).
- One a reduced Boolean function is available, map the function with existing logic gates in the technology library. This step is called technology mapping. For example, the library may only contain INV, NAND and NOR. The Boolean function must be expressed in terms of the corresponding Boolean operation.
- Once the logic circuit is implemented, it is important to verify it's correctness and match the result against truth table.

Example Specification

- The excess-3 code for a decimal digit is the binary combination corresponding to the decimal digit plus 3. e.g. for a decimal digit 5 (binary 0101) the excess-3 code is 5+3=8, which is 1000 in 4-bit binary. Implement a logic circuit which will translate a 4-bit BCD (Binary Coded Decimal), where each decimal digit is converted into corresponding binary digit, into its excess-3 code.

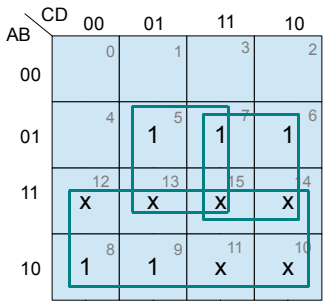
-

Truth Table Formation

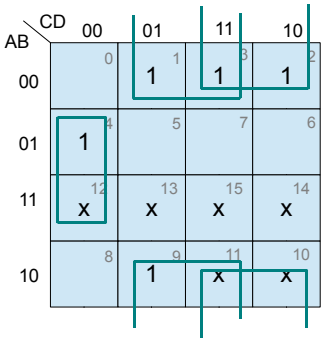
Decimal Digit	Input BCD				Output Excess-3			
	A	B	C	D	W	X	Y	Z
0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	1	0	0
2	0	0	1	0	0	1	0	1
3	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	1	1
5	0	1	0	1	1	0	0	0
6	0	1	1	0	1	0	0	1
7	0	1	1	1	1	0	1	0
8	1	0	0	0	1	0	1	1
9	1	0	0	1	1	1	0	0

5

Optimization



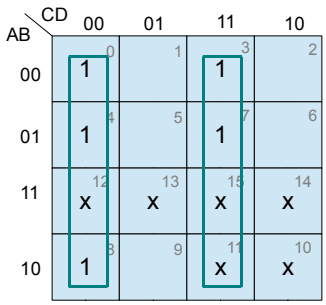
W = A + BC + BD



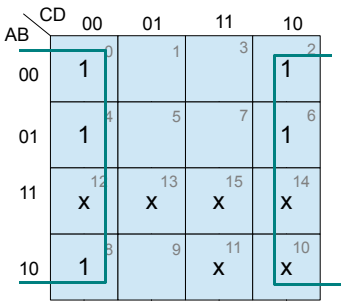
X = B'C + B'D + BC'D'

6

Optimization



Y = CD + C'D'



Z = D'

7

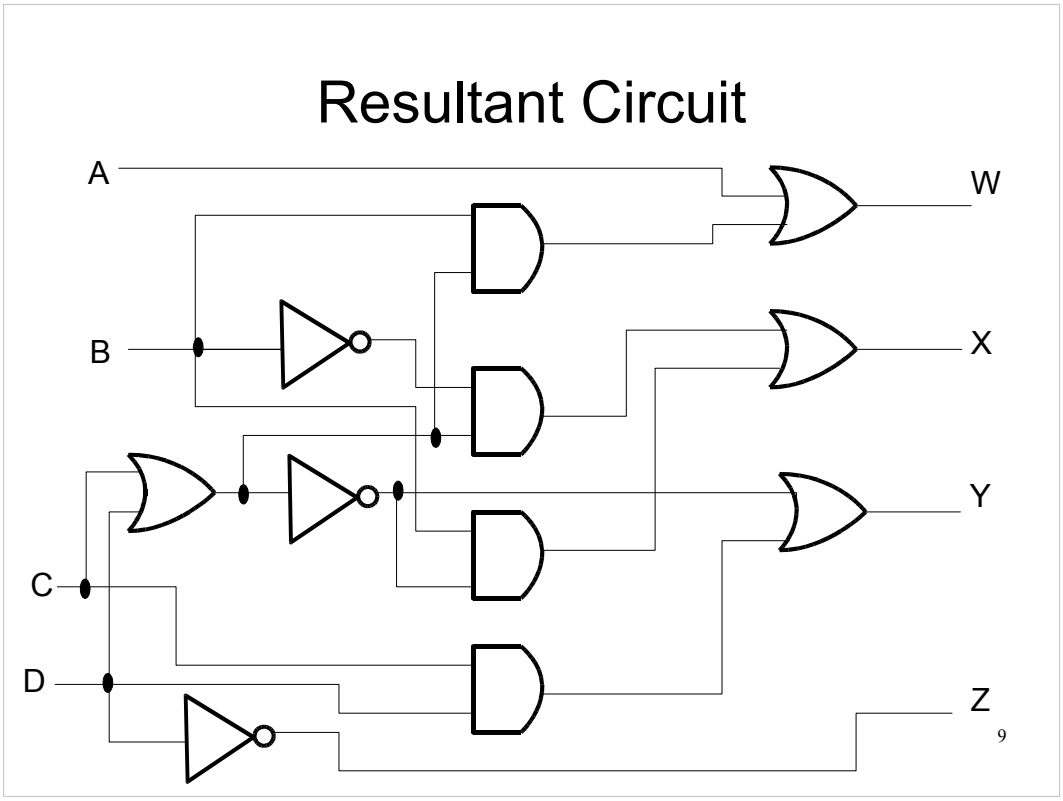
-

Optimization

- W = A + BC + BD = A + B(C+D)
- X = B'C + B'D + BC'D' = B'(C+D) + B(C+D)'
- Y = CD + C'D'
- Z = D'

- Common factor term T = C+D
  - W = A + BT
  - X = B'T + BT'

8



- It is possible to map all the gates representing non-complemented functions into fundamental logic gates INV, NAND, NOR.

Sequential Circuit Design ...

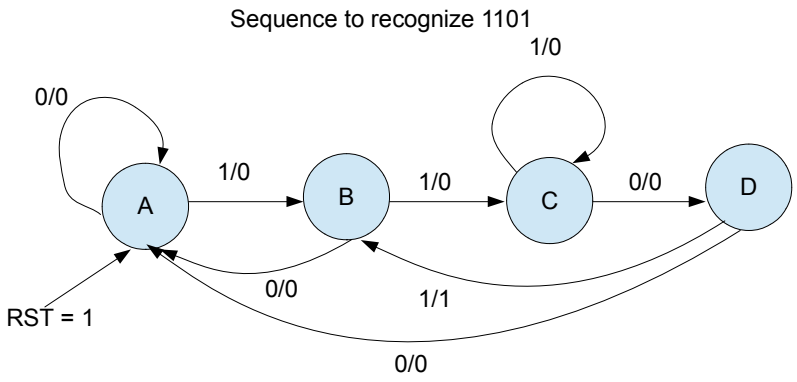
Steps of Sequential Design

- **Specification**: Collect and write the specification
- **Formulation:** Obtain state diagram and state table.
- **State Assignment:** Assign binary code to the states and convert state table to truth table.
- **Output & Flipflop Input Equation:** Select flipflop type and determine the input equation (next state equation) for the flipflop.
- **Optimization**: Optimize the flipflop input equation and output equation.
- **Technology Mapping:** Map the obtained equation in terms of logic gates available for the technology.
- **Verification:** Verify the correctness of the final circuit.

Example Problem

- The sequence recognizer has one input X and one output Z. It has reset applied directly to the flipflop to initialize circuit to all zero. The circuit should recognize the occurrence of the sequenced bit of '1101' on incoming bit stream through X. Once a pattern 1101 is recognized output Z should produce 1, otherwise output will remain 0.

State Diagram



State A	'0' is seen.
State B	'1' is seen
State C	'11' is seen
State D	'110' is seen

13

- For the given problem, the circuit must remember what has been visited for last three times and decide if it is matching the given patter with the current value of X.

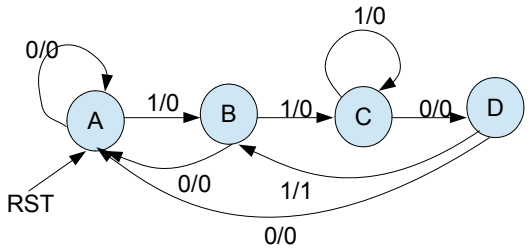
- A circle denotes a state of the machine, and transitions are marked with arrow to show state transitions. Each of this arrows or edges are marked with X/Z denoting what input caused the transition and what is the corresponding output value.

- The machine starts with state A which denotes that last value seen was 0. Upon receiving 0s it will stay at A and output will be 0. Once a 1 comes in, the state is changed to B and output is 0. The state B denotes that last value was 1. Upon receiving 0 in B, state changes to A since '10' can not be a start of '1101'. Upon receiving 1 in B, state goes to C which indicates that the last two values were 1 (a pattern '11' came). State will not be changed from C if further incoming values are 1, because getting more 1 means that at least the last 2 values were '11'. The state C will be changed to D upon receiving 0. This state D denotes that a pattern '110' has been seen. From this state D, transition will happen to B on getting 1 and output will 1 since it indicates that a pattern 1101 is seen. It goes to B because state B denotes that last value seen was '1'. If 0 is received at state D, the state will change to A because A denotes that the last value seen was 0.

- RST is the first state entry signal and it goes to state A with RST at 1. There is from state noted in the state diagram for RST=1. This means, the circuit will come to state A from what ever state it may be in.

State Table Construction

Present State	Next State		Output Z	
	X=0 RST=0	X=1 RST=0	X=0 RST=0	X=1 RST=0
A	A	B	0	0
B	A	C	0	0
C	D	C	0	0
D	A	B	0	1



14

- To construct state transition table, we list down the states in one column as present state of the machine. The next group of columns indicate the next state transition from the present state upon various combination of the primary input. There is another group of columns indicates the output value as a function of the present state and the input combinations.

State Assignments

Present State	Next State		Output Z	
	X=0 RST=0	X=1 RST=0	X=0 RST=0	X=1 RST=0
00	00	01	0	0
01	00	11	0	0
11	10	11	0	0
10	00	01	0	1

- All states are assigned in gray code
  - A = 00
  - B = 01
  - C = 11
  - D = 10

15

- At this state assignment steps we encode each state, denoted with letter, with some binary value. There are several way to assign binary value to a state. Some of them are as following.
  - Using counting order, e.g. A=00, B=01, C=10, D=11.
  - Using gray code, e.g. A=00, B=01, C=11, D=10.
  - Using one hot assignments, e.g. A=0001, B=0010, C=0100, D=1000

- If there are m number of states and minimum number of bit needed is n, then $2^n \geq m$.

- One hot assignment needs total m bit to denote m state. It needs maximum number of bits, but in many cases it reduce the logic circuit complexity for next state determination.

- There is no hard rules and methods to choose assignment strategies for state. A number of methods have been developed based on heuristics, however, there are beyond scope of this class.

- For this specific problem, coding has been done for the states using gray code. Intuitively, this coding method may work good since K-Map uses gray code, and we'll derive the next state logic using K-Map.

Truth Table Construction

State Table

Present State	Next State		Output Z	
	X=0 RST=0	X=1 RST=0	X=0 RST=0	X=1 RST=0
00	00	01	0	0
01	00	11	0	0
11	10	11	0	0
10	00	01	0	1

$A_{t+1} = D_A(A_t,B_t,X) = \Sigma m(3,6,7)$

$B_{t+1} = D_B(A_t,B_t,X) = \Sigma m(1,3,5,7)$

$Z(A_t,B_t,X) = \Sigma m(5)$

State Variable A, B at present state | Primary Input | State Variable A, B at next state | Primary Output

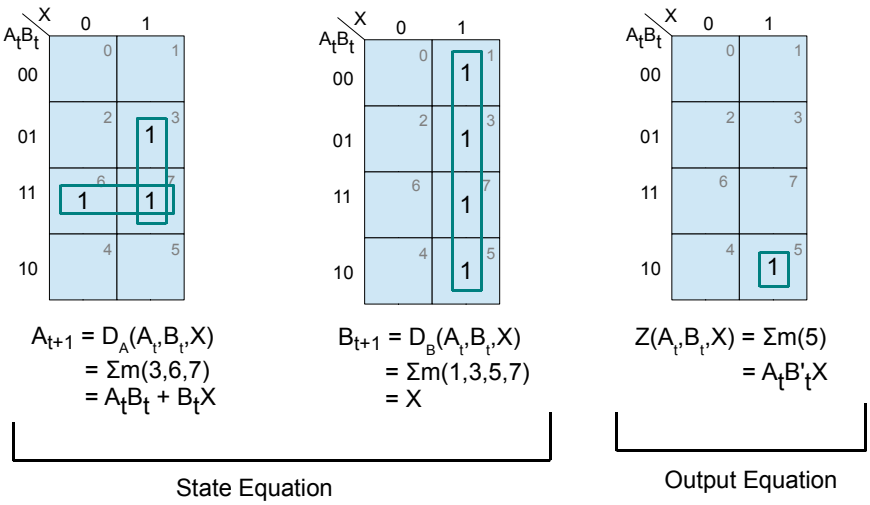
	A_t	B_t	X	A_{t+1}	B_{t+1}	Z
m0	0	0	0	0	0	0
m1	0	0	1	0	**1**	0
m2	0	1	0	0	0	0
m3	0	1	1	**1**	**1**	0
m6	1	1	0	**1**	0	0
m7	1	1	1	**1**	**1**	0
m4	1	0	0	0	0	0
m5	1	0	1	0	**1**	**1**

Truth Table

16

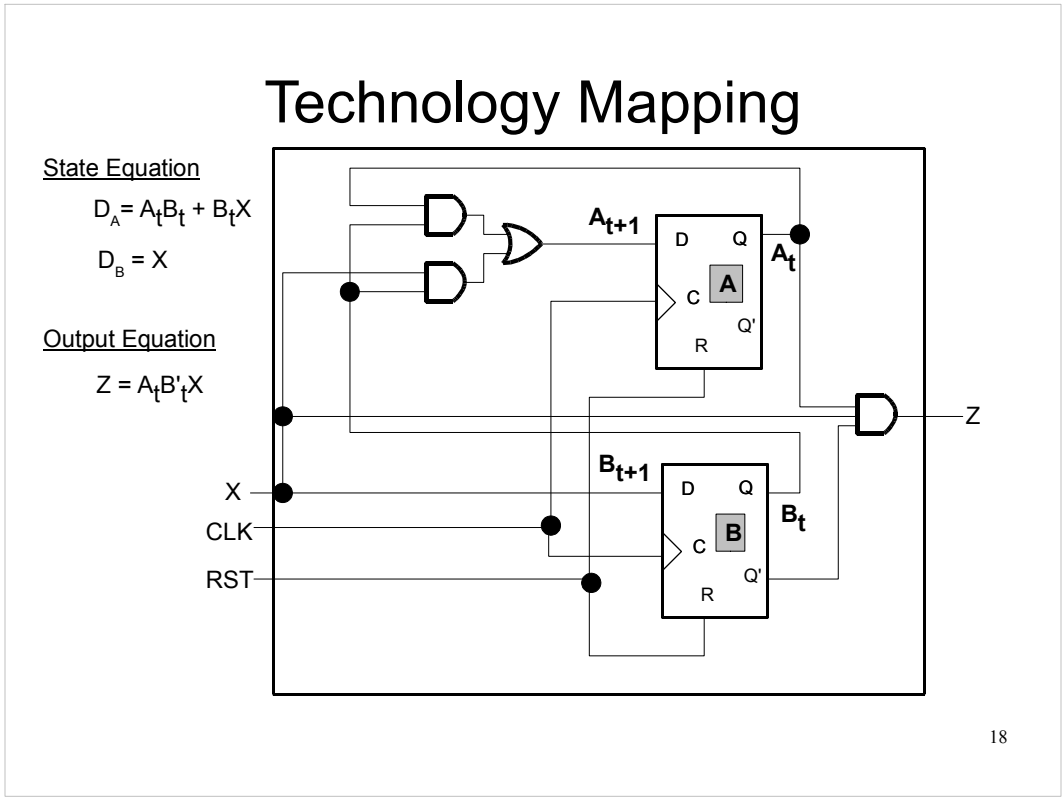
- Once the state assignments are done, the state table is rearranged to make truth table. We take each bit of the present state and the primary input as input for next state logic and the output logic.
- In this example, the present state bits are assumed in variables A_t and B_t (it is not related anyway to the state letter code A, B, C, D – we could have name these two variables A_t and B_t as K_t and V_t). These variables are also called state variable.The input for the truth tables are A_t , B_t and X. The outputs are A_{t+1} , B_{t+1} and Z. The bit combination A_{t+1} B_{t+1} will denote the encoded next state.
- From the truth table minterms are identified for each of the output variable and written in SOP format.

Optimization using K-map



17

- K-map optimization is used to determine reduced equation for each of the truth table output variable.
- The reduced equation for the state variable is also called state equation.



- We are assuming all types of gates are available for technology mapping of these equations.

- To store the two state bits we use two D-flipflops, one for state bit A and other for state bit B. The output of these two flops provide the current state. Based on current state and the primary input, the next state is determined through the combinational Logic circuit. This next state information will be available on the D-pins of the state storage flops which will be latched in the next clock cycle.

- The output Z is also determined using the current state bits and the primary input bit.

CS47 - Lecture 16

Kaushik Patra
(kaushik.patra@sjsu.edu)

19

- Combinational Circuit Design
- Sequential Circuit Design

[Chapter 5 of Logic & Computer Design Fundamentals, 4th Edition, M. Morris Mano, Charles R. Kime]