Synchronous Sequential Logic

* Contains a memory ckt that is able to save outputs (present state) and use it as the input again for the next time.
* Sequential Skts
  + Synchronous – is a system whose behavior depends on input signals at discrete instants of time. Synchronization is achieved by clock pulses (EX: Flip Flops = storring 1 bit of info)
  + Asynchronous – is a system whose behavior depends on the input signals at any instant of time.
* Storage Element
  + Latches – stores 1 bit of info (designed before flip flops)
  + SR latch – has 2 elements r and s. The output of one feeds back into the other. After R is outputed then its pulled and put back into s
  + D Latch was able to remove the forbidden state that can occur in the SR latch – She drew the ckt on the board

Read on the T Flip Flop lol wasn’t paying attention

* T Flip Flop Characterstic table

|  |  |
| --- | --- |
| T | Q(T+1) |
| 0 | Q(T) No change |
| 1 | Q'(T) Compliment |

* + Q(T+1) = Q'T + QT'
    - T (exclusive or) Q
* Characterstic Table for D Flip Flop

|  |  |
| --- | --- |
| D | Q(T+1) |
| 0 | 0 Reset |
| 1 | 1 Set |

* She Drew a big d flip flop ckt on board
  + A(T+1) = A(T)X(T) + B(T)X(T) => A(T+1) = Ax + Bx
  + B(T+1) = A'(T)X(T) => B(T+1) = A'x
  + Y = [A(T) + B(T)]x'(T) => Y = (A+B)x'
  + State Table

|  |  |  |  |
| --- | --- | --- | --- |
| Present State  A B | Input  x | Next State  A B | Output  y |
| 00 | 0 | 0 0 | 0 |
| 00 | 1 | 0 1 | 0 |
| 01 | 0 | 0 0 | 1 |
| 01 | 1 | 1 1 | 0 |
| 10 | 0 | 0 0 | 1 |
| 10 | 1 | 1 0 | 0 |
| 11 | 0 | 0 0 | 1 |
| 11 | 1 | 1 0 | 0 |

* + She drew a state diagram based off the truth table
* She drew a big T Flip Flop ckt on board
  + T(A) = Bx
  + T(B) = x
  + Y = AB
  + State Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Present State  A B | Input  X | Inputs to T F\_Fs  Ta Tb | Output  Y | Next State  A B |
| 00 | 0 | 0 0 | 0 | 0 0 |
| 00 | 1 | 0 1 | 0 | 0 1 |
| 01 | 0 | 0 0 | 0 | 0 1 |
| 01 | 1 | 1 1 | 0 | 1 0 |
| 10 | 0 | 0 0 | 0 | 1 0 |
| 10 | 1 | 0 1 | 0 | 1 1 |
| 11 | 0 | 0 0 | 1 | 1 1 |
| 11 | 1 | 1 1 | 1 | 0 0 |

* + She drew a state diagram based off truth table
* State Reduction
  + 2^n means you need N flip flops.
  + 1) State Table
  + 2) Check each row in table to find equiv rows
  + 3) If there are equiv rows then delete until only one row
  + 4) Substitute for the states deleted in other rows with the equiv states
    - State Table

|  |  |  |
| --- | --- | --- |
| Preset State | Next State  X = 0 | X = 1 | Output  X=0|X=1 |
| a | A b | 0 0 |
| b | C d | 0 0 |
| c | A d | 0 0 |
| d | E f | 0 1 |
| e | A f | 0 1 |
| f | G f | 0 1 |
| g | A f | 0 1 |

Delete G and substitute e for the g's in the table. Then F and D are the same so delete D and replace f's with D

* 1. 5 states => 2^3 so 3 F\_F's
  2. She redrew the state diagram without the gs and fs
* Designing a Synchronous CKT
  1. Word description of the ckt behavior (State diagram, or state table)
  2. Obtain the state table
  3. Reduce the number of states – Row reduction.
  4. Assign the binary values to each state if given letter symbols to the states
  5. Determine the number of flip flops needed
  6. Choose the type of F\_F needed
  7. From state table derive excitations and output tables
  8. USe the Kmaps derive Output functions
  9. Draw Logic Diagram
* EX: Design clocked sequential ckt for state diagram below using T F\_Fs
  + She drew the state diagram on the board
  + State Table

|  |  |
| --- | --- |
| Present State  A B | Next State  X=0|X=1  A B | AB |
| 00 | 00 01 |
| 01 | 10 01 |
| 10 | 10 11 |
| 11 | 11 00 |

2 Flip flops needed

* + Excitation Table

|  |  |  |  |
| --- | --- | --- | --- |
| Present State  A B | Input x | Next State  A B | Inputs to T F\_F  Ta Tb |
| 00 | 0 | 0 0 | 0 0 |
| 00 | 1 | 0 1 | 0 1 |
| 01 | 0 | 1 0 | 1 1 |
| 01 | 1 | 0 1 | 0 0 |
| 10 | 0 | 1 0 | 0 0 |
| 10 | 1 | 1 1 | 0 1 |
| 11 | 0 | 1 1 | 0 0 |
| 11 | 1 | 0 0 | 1 1 |

* + Kmaps
  + TA

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A \BX | 00 | 01 | 11 | 10 |
| 0 |  |  |  | 1 |
| 1 |  |  |  | 1 |

* + - Ta = Bx'
  + TB

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A \BX | 00 | 01 | 11 | 10 |
| 0 |  | 1 |  | 1 |
| 1 |  | 1 |  | 1 |

* + - Tb = B'x + Ax + A'Bx'
  + Draw the ckt diagram last – She drew it on the board