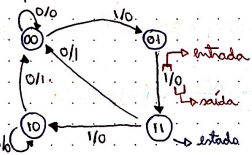
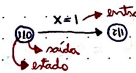
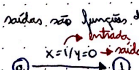
síntese de circuitos sequenciais

- circuitos lógicos sequenciais
  - possui algum elemento de memória
  - estado atual → está salvo em um conjunto de FF's
  - próximo estado → tempo (t+1)
  - saídas → função do estado atual + entradas
- diagrama de estados
  - 

- modelo de moore → máquina de estados finitos
  - saídas são função apenas dos estados
  - 

  - saída é "ficar" → depende apenas do estado atual e <u>muda sempre na borda do clock</u>
- modelo de mealy → máquina de estados finitos
  - saídas são função das entradas e dos estados
  - 

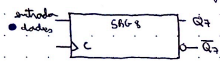
  - saída também depende da entrada e <u>muda diretamente com a entrada</u>

registradores

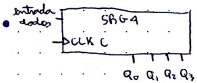
- definição → circuito digital com 2 funções lógicas { armazenamento de dados / movimentação de dados

- registradores de deslocamento com entrada e saída serial / saída paralela

  - recebe um bit de cada vez em uma série e gera saída em série] → saída serial



  - recebe um bit de cada vez em uma única linha e gera informações de forma paralela



contadores

- contadores assíncronos

  - as saídas dos FF's não mudam de estado exatamente com o mesmo sincronismo que o pulso de clock

  - nº de estados = módulo do contador = 2^N → N = nº de FF's

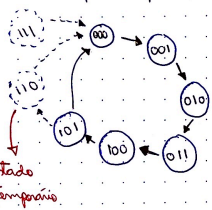
  - t_{pd} → tempo de atraso entre a aplicação do pulso e a mudança da saída → 5 a 10 ns
    ↳ o emésimo FF requer $N \cdot t_{pd}$ para responder

  - evitar problemas de atraso → $T_{clock} \geq N \cdot t_{pd}$ e $f_{clock} \leq \frac{1}{N \cdot t_{pd}}$
    (T_{clock} = período entre os pulsos)

  - glitches → valores de contagem errados entre as transições de estados

contadores

- contadores de módulo $< 2^n$
  - "eu tinha n flip-flops, mas não quero usar todos esses estados: quero parar antes e recomeçar mais cedo"
  - força o contador a { contar até um valor menor / resetar quando chegar nesse valor }
  - C B A
    0 0 0
    0 0 1
    0 1 0
    0 1 1
    1 0 0
    1 0 1
    1 1 0



  - ★ o estado 110 é temporário
  - ★ o contador conta, de fato, de 000 até 101 (módulo 6)
  - ★ o estado 110 é após um spike
  - procedimento geral para construção
    1. determine o menor número de FFs necessários de modo que $2^N > X$
    2. conecte as saídas de uma porta NAND às entradas assíncronas CLEAR de todos os FFs
    3. determine quais são os FFs em nível ALTO no valor X e conecte as saídas correntes desses FFs às entradas da porta NAND
- contadores síncronos (paralelos)
  - os FFs são disparados simultaneamente (em paralelo) pelos pulsos de clock
  - apenas o FF A (LSB) tem suas entradas J e K permanentemente em nível alto
- contadores síncronos ~~(paralelos)~~ - vantagens
  - os FFs não ~~têm~~ tem o atraso de propagação somado → atraso total = t_{pd} do FF + t_{pd} da NAND
  - um contador síncrono tem atraso ~~mínimo~~ modo "simples" de um flip-flop somente quando todos os flip-flops de ordem menor estiverem em nível lógico alto
- contadores síncronos decrescentes
  - basta usar as saídas invertidas dos FFs

Contadores

- CIs de contadores síncronos

$\overline{CLR}$	$\overline{LOAD}$	ENP	ENT	CLK	função
L	X	X	X	X	clear assíncrono
L	X	X	X	↑	clear síncrono
H	L	X	X	↑	carga síncrona
H	H	H	H	↑	contagem crescente
H	H	L	X	X	sem mudança
H	H	X	L	X	sem mudança

projeto de contadores e multiplicadores

- projeto de contadores síncronos

  - circuitos de contadores síncronos podem ser projetados para gerar qualquer sequência de contagem
  - não usar entradas assíncronas dos FFs → não gera efeitos
  - para se projetar contadores síncronos, deve-se identificar as entradas de controle de cada FF em cada estado do contador
  - preciso encontrar as expressões lógicas das entradas dos FFs para daí montar a tabela de estados.
  - autocorretor → estados retornam para um estado da sequência normal.
  - as entradas de todos os FFs precisam estar com os valores que levam para o próximo estado no momento em que houver a transição ativa do clock

- etapas do projeto de contadores síncronos

  1. determine o nº de FFs necessários e a sequência desejada
  2. desenhe o diagrama de transição de estados mostrando todos os estados
  3. construa uma tabela de transição de estados listando todos os estados atuais e os próximos
  4. acrescente a esta tabela uma coluna para cada entrada j e k
  5. projete os circuitos lógicos necessários para gerar os níveis requeridos
  6. implemente as expressões finais

Multiplicador

- ideia geral
  - realizamos shifts para a direita do produto parcial
  - quando { o bit = 1, lendo da direita para esquerda, soma e RS
    o bit = 0, lendo da direita para esquerda, RS

- operações no multiplicador
  1. o multiplicando é carregado no registrador B
  2. o multiplicador é carregado no registrador Q
  3. C||A são inicializados em 0 quando Q vale 1
  4. os produtos parciais são formados no registradores C||A||Q.
  5. cada bit do multiplicador, começando com LSB, é processado.
  6. o conteúdo de C||A||Q é deslocado para direita.
     - se ocorrer overflow → o vai-um é recuperado do FFC durante o deslocamento para direita
  7. 5 e 6 se repetem até que a condição Counter P = 0

- controle do multiplicador
  - Estado IDLE { saídas da multiplicação anteriores são mantidas
    entrada G é usada como condição para iniciar a multiplicação
    C, A, P são inicializados
  - Estado MULO { a adição condicional é realizada baseada no valor Q_0
  - Estado MUL1 { deslocamento para a direita
    verifica a contagem final P = 0