11. Lógica binária e Álgebra de Boole

operações lógicas

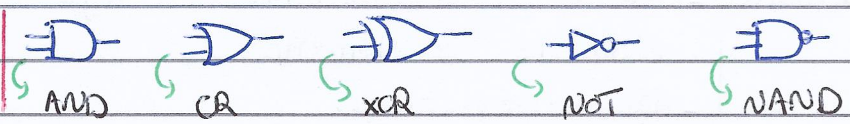
- A AND B = $A \cdot B$
- A NAND B = $\overline{AB}$
- A OR B = $A + B$
- A XOR B = $A \oplus B = A\bar{B} + \bar{A}B$
- A XNOR B = $\overline{A \oplus B} = AB + \bar{A}\bar{B}$
- NOT A = $\bar{A}$

portas lógicas

nome dado aos circuitos lógicos que implementam as funções lógicas (em circuitos elétricos)

entradas
- uma → NOT
- 2 ou + → AND, OR, XOR, NAND, NOR, ...

representações

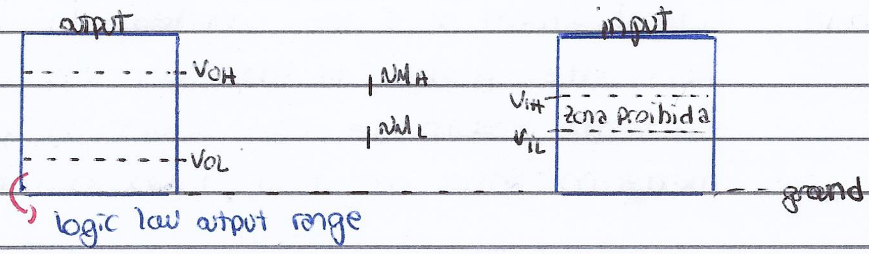


níveis lógicos

tensões
- 0 → 0 V (ground)
- 1 → 5 V (tensão de alimentação)

ruído: tudo o que degrade a qualidade do sinal
resistência, imperfeições da fonte...

margem de ruído: margem extra que os receptores têm para absorver eventuais desvios, ruídos, etc



$$NM_H = V_{OH} - V_{IH} \qquad NM_L = V_{IL} - V_{OL}$$

tipos de circuitos

lógica
- combinatória: sem memória (ñ há ciclos); saídas determinadas por entradas atuais
- sequencial: com memória; saídas podem ser determinadas por ent passadas

circuito lógico

composição
- entradas
- saídas

nós (no caso de sub-circuitos) → internos qnd ligam circuito a outro

especificações
- funcionais → descreve relações lógicas entre entradas e saídas, podendo ser especificado por tabela de verdade ou equação Booleana.
- temporais → Δt entre mudança de entrada e estabilização da saída ($T = \frac{1}{f}$)

obs: Álgebra de Boole considera regimes estacionários (despreza variações temporais)
≠
componentes eletrônicos são contínuos e não discretos

circuito combinatório
- cada elemento tem de ser combinatório
- cada nó é uma entrada ou liga a uma e uma só saída de um elemento
- não existem ciclos

funções lógicas

representação
- tabela de verdade
- equação algébrica
- circuito lógico

equação Booleana: permite especificação funcional da saída em função das entradas. $S = F(A, B, C, n)$

álgebra de Boole

complemento $\bar{A}$
literal: variável ou o seu complemento $A, \bar{A}$
implicante: produto de literais AB, $\bar{B}C$, ABC
mintermo: produto que inclui todas as variáveis ABC, $\bar{A}BC$
maxtermo: soma que inclui todas as variáveis $(A+B+C)$

precedência → NOT > AND > OR

tabela de verdade: que descreve um problema lógico pode sempre ser descrita como
- n entradas → 2^n linhas

soma de produtos (SOP) primeira F.C
produto de somas (POS) segunda F.C

SoP $\Sigma(n,y)$ $n,y \in \mathbb{N}$	a cada linha da t.v. corresponde um mintermo cada mintermo assume o valor 1 na sua linha apenas somam-se os mintermos que assumem o valor 1
PoS $\Pi(n,y)$ $n,y \in \mathbb{N}$	mintermo → maxtermo // 1 → 0 → dual das SoP

axiomas — obs todos têm um dual (troca de 0 por 1 e vice-versa, mantendo o teorema verdadeiro)

valor binário	$B = 0$ se $\neq 1$	$B = 1$ se $\neq 0$
negação	$\overline{0} = 1$	$\overline{1} = 0$
and	$0.0 = 0$ // $1.1 = 1$	$1+1 = 1$ // $0+0 = 0$
	$0.1 = 1.0 = 0$	$1+0 = 0+1 = 1$

teoremas

	identidade	$B.1 = B$	$B+0 = B$
	el. absorvente	$B.0 = 0$	$B+1 = 1$
vip	idempotência	$B.B = B$	$B+B = B$
vip	involução	$\overline{\overline{B}} = B$	
	complementaridade	$B.\overline{B} = 0$	$B+\overline{B} = 1$

teoremas

	comutatividade	$B.C = C.B$	$B+C = C+B$
	associatividade	$BC(D) = B(CD)$	$(B+C)+B = B(C+B)$
vip	distributividade	$(BC)+(BD) = B(C+\ )$	$(B+C)(B+D) = B+(CD)$
vip	absorção	$B(B+C) = B$	$B+(BC) = B$
vip	adjacência	$(BC)+(B\overline{C}) = B$	$(B+C)(B+\overline{C}) = B$
vip	consenso	$(BC)+(\overline{B}D)+(CD)$	$(B+C)(\overline{B}+D)(C+D)$
		$=(BC)+(\overline{B}D)$	$=(B+C)(\overline{B}+D)$

obs para demonstrar os teoremas podemos utilizar a indução perfeita, que consiste em demonstrar na tabela de verdade que se verifica para todos os valores das variáveis

leis de Morgan — negação do produto é a soma das negações

bubble pushing — troca de OR/AND por AND/OR
empurram-se as bolhas

representação em circuitos | convenções | entradas à esquerda / cima
saídas à direita / baixo
usar linhas retas
fluxo das gates é da esq para a dir

don't care | podemos simplificar a tabela de verdade usando o símbolo X para representar os casos em que a saída é independente de alguma entrada

circuitos de prioridades
circuitos onde se evitam conexões devido aos don't care

conjunto completo de operadores | conjunto de operadores que permite a implementação de qualquer função booleana

exemplos | {AND, OR, NOT}
{AND, NOT}
{OR, NOT}
{NAND} → leis de Morgan a SoP
{NOR}

obs menos portas facilita trabalho de quem projeta o circ

formas canónicas | 1ª SoP → AND - OR
2ª PoS → OR - AND
3ª SoP → NAND - NAND
4ª PoS → NOR - NOR

mapas de Karnaugh - representar t.v. num mapa, desenhando círculos sempre que houver 1's em células adjacentes (2^n elementos)
- na expressão final usamos apenas os literais que são invariantes em cada círculo

obs entre cada linha / coluna só pode variar 1 bit
linha de baixo é vizinha da de cima (esq / dir)
cantos são vizinhos