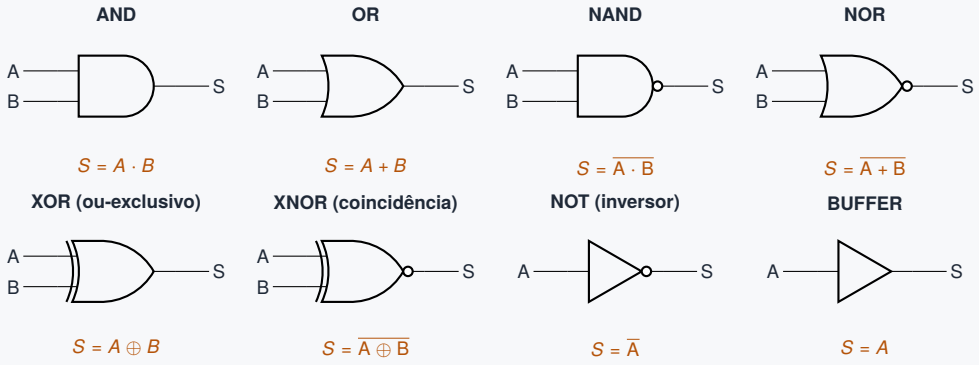
Notação de circuitos digitais — frente: combinacional

1. Portas lógicas — símbolo, expressão e tabela-verdade



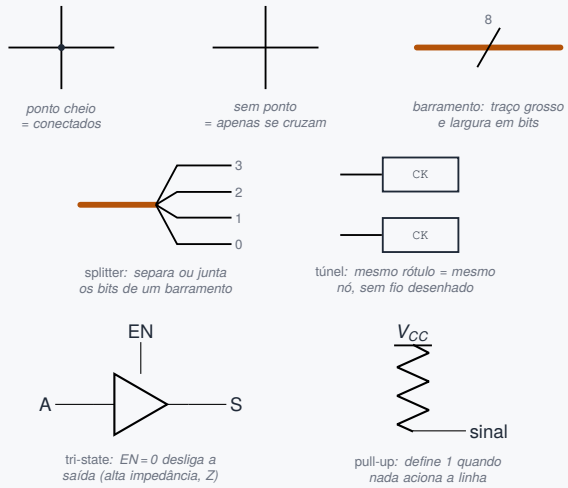
A	B	AND	OR	NAND	NOR	XOR	XNOR
0	0	0	0	1	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	1	0	0	0	0

A	NOT	BUF
0	1	0
1	0	1

a bolha ○ na saída inverte: NAND é AND + bolha, NOR é OR + bolha, XNOR é XOR + bolha. Bolha na entrada *inverte aquele sinal antes de ele entrar na porta.*

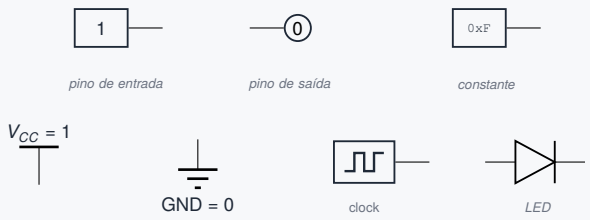
Complemento: $\overline{A}$ no desenho, `A'` quando o traço não for viável (e-mail, código). **NAND e NOR são universais**: qualquer função sai só com uma delas.

2. Fios, junções e barramentos



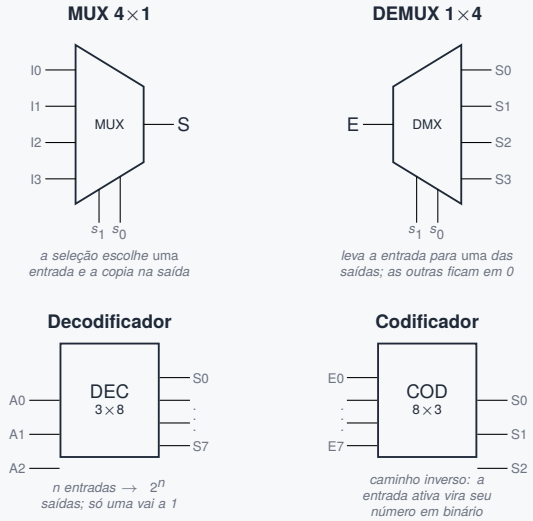
Fio sem fonte fica **indefinido** (X); dois valores na mesma linha dão **conflito**. No Logisim os dois aparecem pela cor do fio (quadro 12).

3. Entradas, saídas, constantes e *clock*



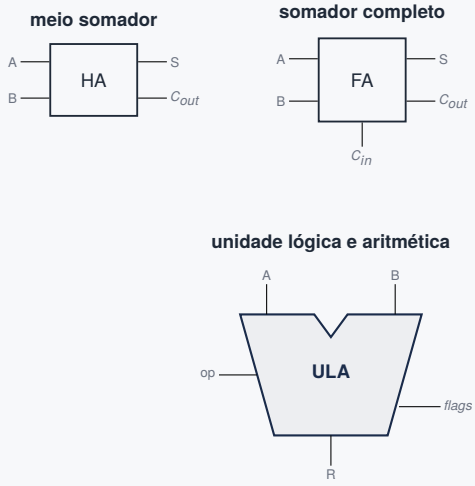
Nível lógico: **0** = baixo (0 V, falso), **1** = alto (V_{CC}). **Z** (alta impedância) e **X** (indefinido) não são níveis lógicos: são estados do fio.

4. Seleção e decodificação



O trapézio é a forma canônica de MUX e DEMUX: a base maior fica do lado com mais linhas e a seleção entra pela base inclinada, nunca junto com os dados.

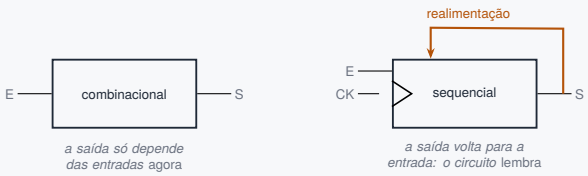
5. Aritmética: somadores e ULA



A	B	C_{in}	S	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

O V no topo é a marca da ULA: dois operandos por cima, resultado por baixo, código da operação pela lateral. *Flags* típicas: Z (zero), C (*carry*), N (negativo), V (*overflow*).

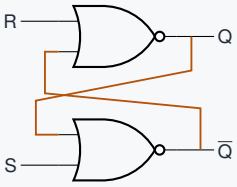
6. O que torna um circuito sequencial



Só aqui aparecem o *clock* (diz *quando* a saída pode mudar) e as entradas assíncronas *Preset*/*Clear* (mudam a saída sem esperar o *clock*).

7. *Latch* — sensível a nível

RS com NOR

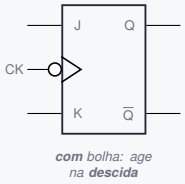


S	R	Q
0	0	mantém
0	1	0 (*reset*)
1	0	1 (*set*)
1	1	proibido

Com o nível de habilitação ativo o *latch* segue a entrada — inclusive o ruído. Com NAND tudo se inverte: entradas $\overline{S}$ e $\overline{R}$, ativas em 0, e a combinação proibida passa a ser 0 0.

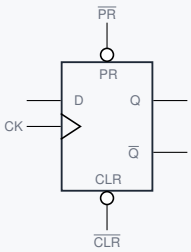
8. *Flip-flop* — disparado por borda

JK



com** bolha: age na **descida

D



sem** bolha: age na **subida

JK		Q_{n+1}
0	0	Q_n (mantém)
0	1	0
1	0	1
1	1	$\overline{Q_n}$ (inverte)

D	Q_{n+1}
0	0
1	1

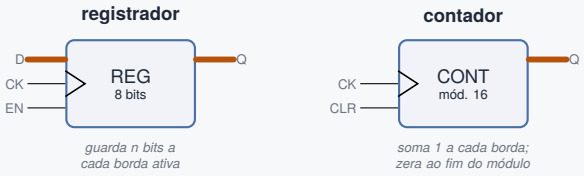
copia D

T	Q_{n+1}
0	Q_n
1	$\overline{Q_n}$

divide por 2

A **cunha** ▷ no *clock* é o que distingue *flip-flop* de *latch*. $\overline{PR}$ e $\overline{CLR}$ são **assíncronas** e não podem valer 0 ao mesmo tempo. T = JK com J = K; D = JK com K = $\overline{J}$.

9. Registradores e contadores

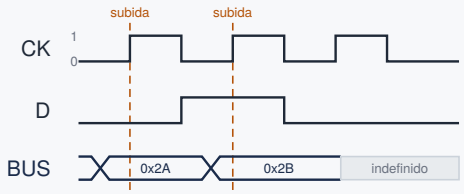


Azul claro = memória, em toda figura da disciplina. O registrador de deslocamento é a mesma caixa, com entrada serial de um lado e *n* saídas paralelas do outro.

10. Memórias RAM e ROM

Capacidade = $2^{\text{bits de endereço}} \times$ largura da palavra. **A** = endereço, **D** = dados (bidirecional na RAM). Barra em cima ($\overline{WE}$, $\overline{CS}$) = **ativa em 0**.

11. Cronogramas (*timing diagrams*)



O tempo corre da esquerda para a direita; um degrau é uma transição. A tracejada marca a **borda ativa** — só ali a saída de um *flip-flop* muda. Barramento: duas linhas cruzando a cada troca de valor; o hachurado é *don't care*.

12. Do símbolo ao Logisim — onde achar cada componente

AND, OR, NAND, NOR, XOR, XNOR → Gates · `...` `Gate`. *Number of Inputs*, *Data Bits*, *Gate Size*, *Negate* por entrada (é a bolha).
NOT, BUFFER, *tri-state* → Gates · `NOT Gate`, `Buffer`, `Controlled Buffer/Inverter`. *Data Bits*, *Control Line*.
Pino de entrada / de saída → Wiring · `Pin`. *Output?* (não = entrada), *Data Bits*, *Label*.
Constante, V_{CC}, GND → Wiring · `Constant`, `Power`, `Ground`. *Value*, *Data Bits*.
Clock → Wiring · `Clock`. *High/Low Duration*; simular com Ctrl+K ou Ctrl+T.
Túnel → Wiring · `Tunnel`. *Label* — mesmo rótulo, mesmo nó.
***Splitter* e barra de bits** → Wiring · `Splitter`. *Fan Out*, *Bit Width In*, distribuição dos bits.
***Pull-up*/*down*, sonda de valor** → Wiring · `Pull Resistor` (*Pull Direction*), `Probe` (*Radix*).
LED, botão, display → Input/Output · `LED`, `Button`, `Hex Digit Display`. Cor, *Active On High*.
MUX, DEMUX → Plexers · `Multiplexer`, `Demultiplexer`. *Select Bits*, *Data Bits*, *Include Enable?*
Decodificador, codificador → Plexers · `Decoder`, `Priority Encoder`. O codificador do Logisim é **de prioridade**.
Somador de *n* bits, subtrator, comparador, deslocador → Arithmetic · `Adder`, `Subtractor`, `Comparator`, `Shifter`. O `Adder` já traz C_{in} e C_{out}.
***Flip-flop* SR, D, T, JK** → Memory · `S-R/D/T/J-K Flip-Flop`. *Trigger*: *rising*/*falling edge*; *Preset* e *Clear* nos cantos.
Registrador, contador, deslocamento → Memory · `Register`, `Counter`, `Shift Register`. *Trigger*, *Maximum Value* (o módulo), *Number of Stages*.
RAM, ROM → Memory · `RAM`, `ROM`. *Address Bit Width*, *Data Bit Width*; conteúdo pelo *Poke Tool*.
Meio somador e somador completo → *não existem prontos* — montar com XOR + AND, como no quadro 5.
***Latch* RS e D** → *não existem prontos* — montar com `NAND`/`NOR` (quadro 7); o Logisim só traz *flip-flops* de borda.
ULA → *não existe pronta* — montar com `Adder` + portas + `Multiplexer` escolhendo a operação.
Cronograma → menu *Simulate* · *Logging* escolhe os sinais e exporta; o Logisim Evolution tem *Chronogram*.

Cor do fio é diagnóstico, não decoração: cinza = sem valor, azul = indefinido (X), verde-escuro = 0, verde-claro = 1, preto = barramento, vermelho = conflito, laranja = largura incompatível — os dois últimos são erro de montagem.

Bibliografia: CAPUANO (Ac.5012157) · TOCCI (Ac.131146) · SILVA (Ac.5063593).