

Resumo Circuitos

Introdução

Existem alguns tipos de circuitos:

Combinacionais

- Saidas dependem **unicamente** das entradas
- Lógica combinacional e álgebra de boole

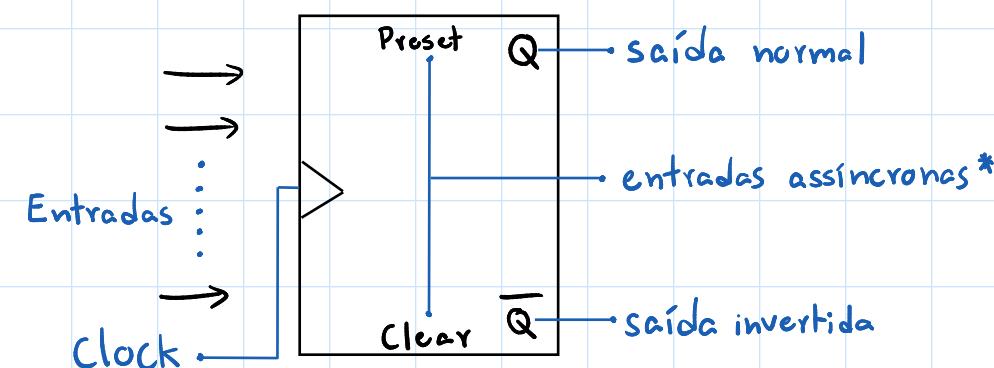
Biestáveis

- Flip-flops e latches são circuitos sequenciais mais elementares.
- Representam uma unidade de bit.
- Biestáveis pois possuem dois estados lógicos estáveis (0 e 1).

Sequenciais

- Possui realimentação da saída para a entrada.
- Saidas dependem ^(também) de estados anteriores
 - ↳ Síncronos (Tocci chama de flip-flop)
 - Comandados por clock
 - ↳ Assíncronos (Tocci chama de latch)
 - (para a maioria dos autores, ambos são flip-flops)

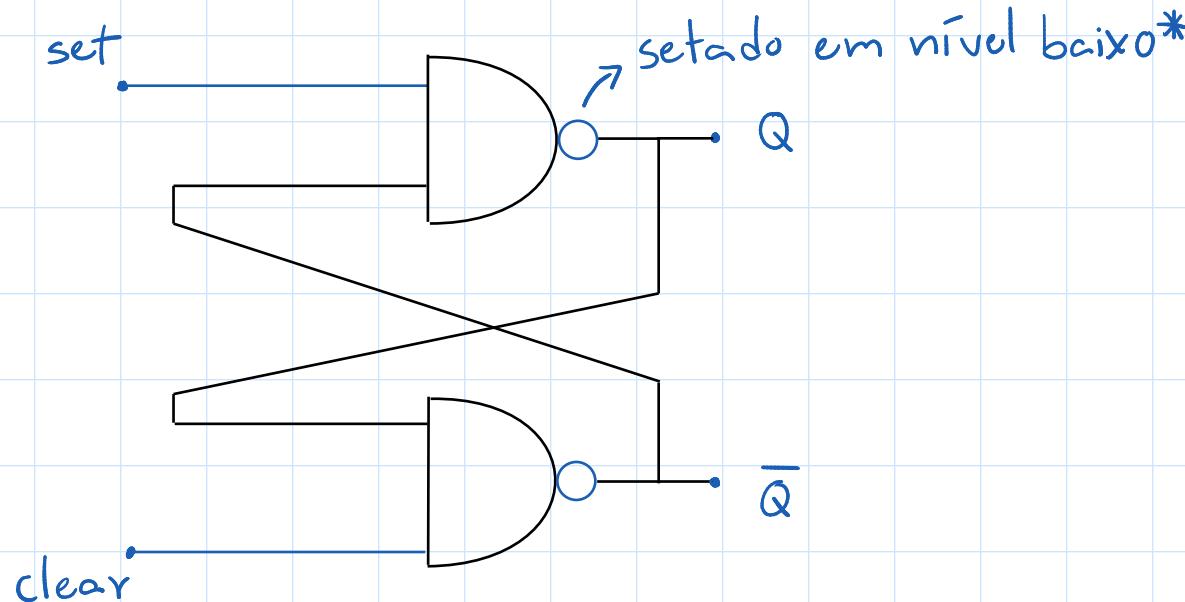
Anatomia de um flip-flop



* As entradas assíncronas tem prioridade MÁXIMA, acima até mesmo do clock. $\text{Preset} = 1$
 $\text{Clear} = 0$

Latches

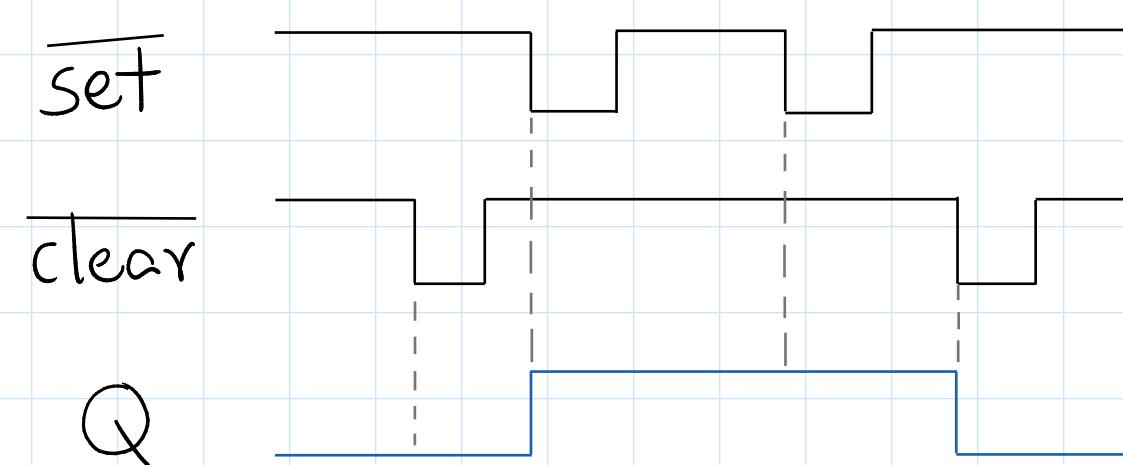
Latch SC
 (ativada em nível baixo)



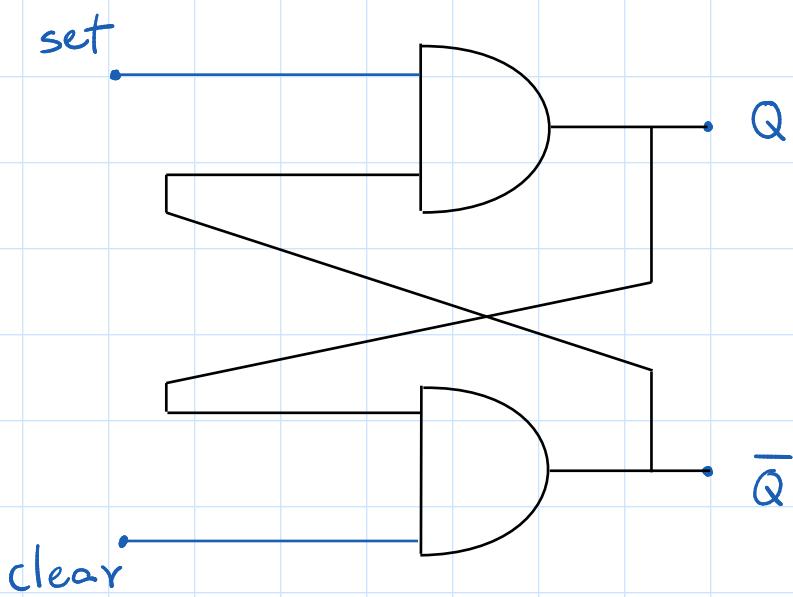
set	clear	Q (saída)
1	1	não muda
0	1	$Q = 1$
1	0	$Q = 0$
0	0	índigo <small>(comportamento imprevisível)</small>

* Por causa dos inversores, essa latch tem o set e clear ativados em nível baixo, ou seja, quando o nível for 0, o comando é dado.

Exemplo de diagrama
 (latch sc ativada em nível baixo)

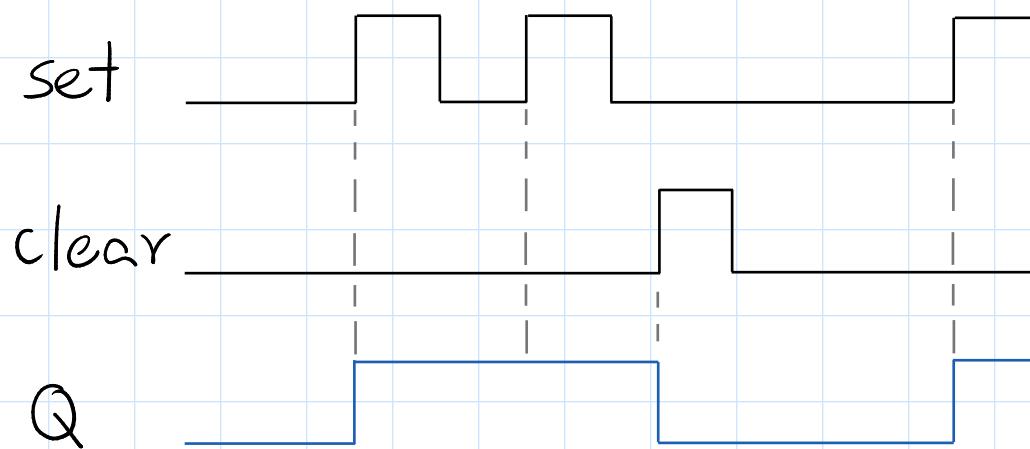


Latch SC
(ativada em nível alto)



set	clear	Q (saída)
1	1	inválido (comportamento imprevisível)
0	1	$Q = 0$
1	0	$Q = 1$
0	0	não muda

Exemplo de diagrama
(latch sc ativada em nível alto)



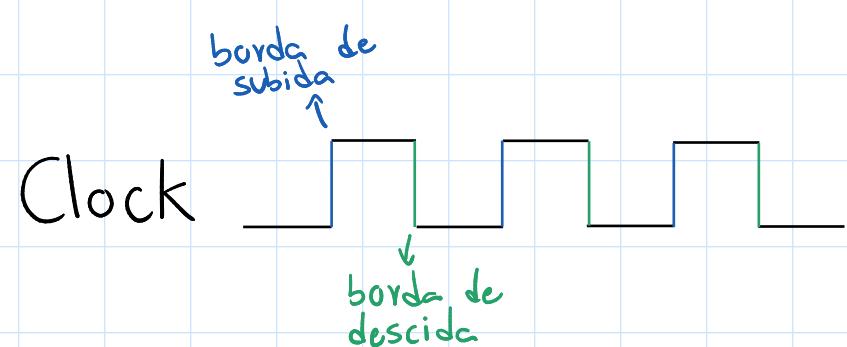
OBS: é possível criar outros latches SC com o mesmo funcionamento usando outras portas lógicas usando as leis de De Morgan. Por exemplo substituindo a porta $\neg(A \wedge B)$ por $\neg(\neg A \vee \neg B)$.

IMPORTANTE!

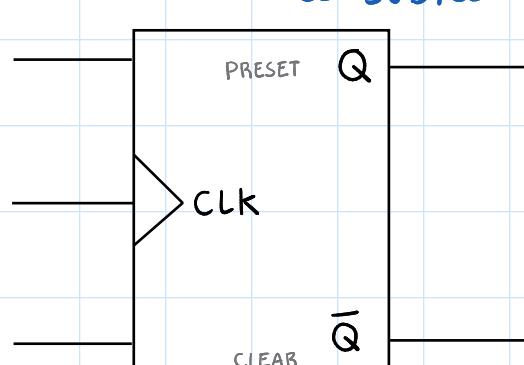
- * Latch SC também pode ser chamado de Latch SR ou Flip-Flop SR
- * S = set C = clear (sinônimo de R = reset)

Clock

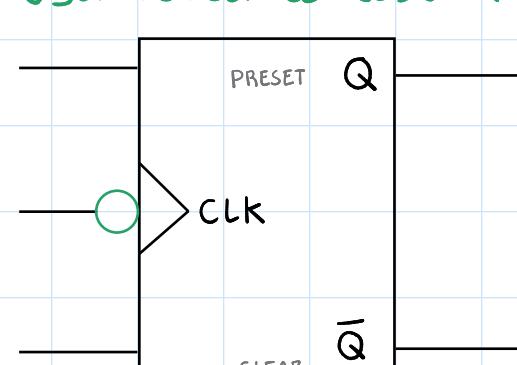
Sistemas digitais podem ser síncronos ou assíncronos, nos assíncronos as mudanças de estado ocorrem imediatamente assim que há mudança nas entradas, já nos síncronos existe o sinal de clock, que é um pulso constante que determina quando que o estado do circuito será verificado e atualizado.



usa borda de subida:



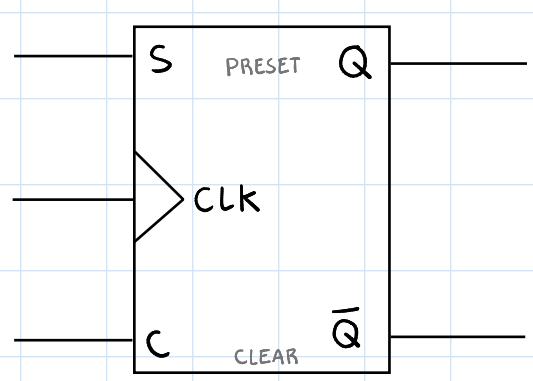
usa borda de descida:



Flip-Flops

São latches com clock.

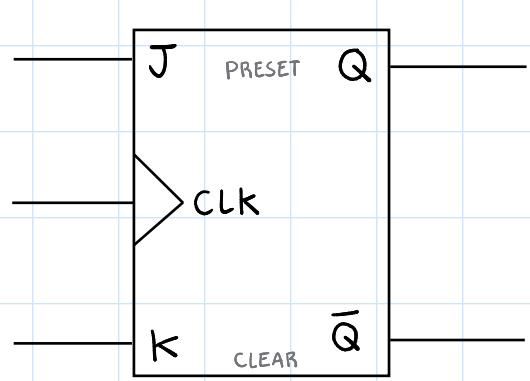
Flip-Flop SC



S	C	CLK	saída
1	1	↑	inválido (comportamento imprevisível)
0	1	↑	$Q = 0$
1	0	↑	$Q = 1$
0	0	↑	não muda

(igual à latch SC, mas com clock)

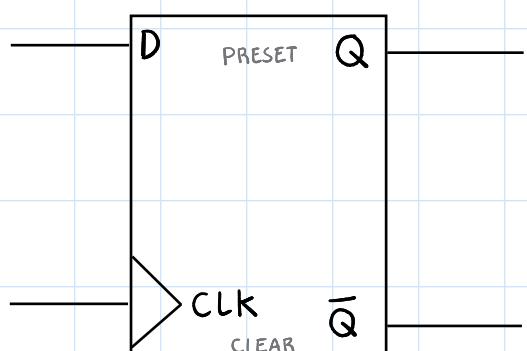
Flip-Flop JK



J	K	CLK	saída
1	1	↑	comutação
0	1	↑	$Q = 0$
1	0	↑	$Q = 1$
0	0	↑	não muda

(J funciona como set e K como clear, mas quando são ativados simultaneamente comutam o estado)

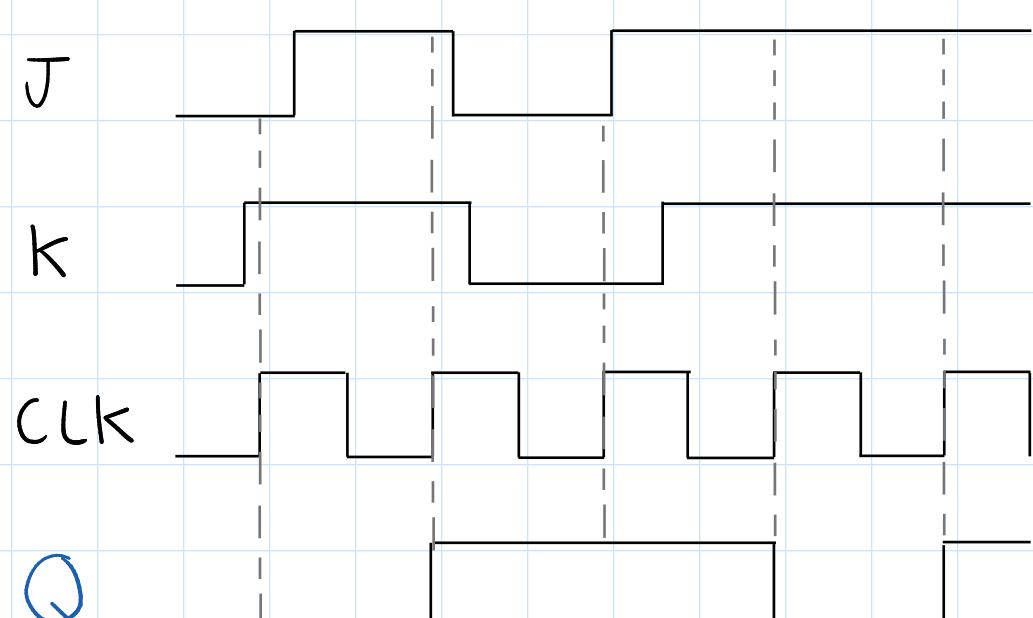
Flip-Flop D



D	CLK	saída
0	↑	0
1	↑	1

(apenas deixa o estado passar, respeitando o clock)

Diagrama de exemplo (Flip-Flop JK)

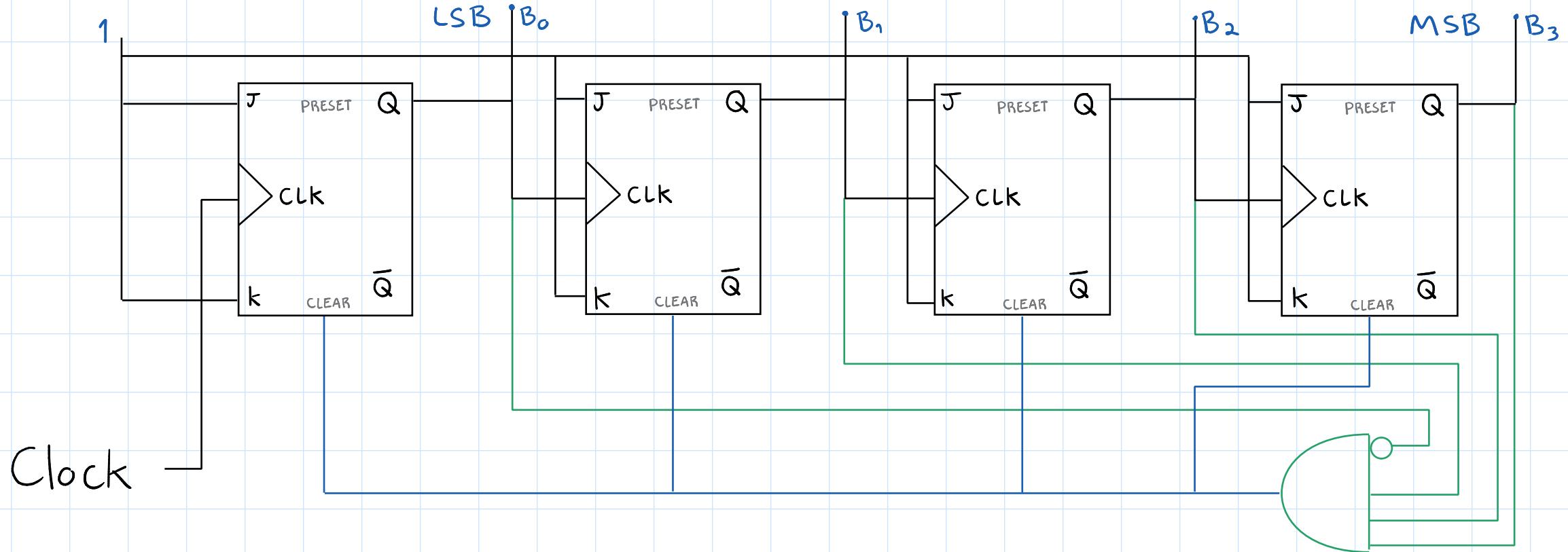


Contadores

Dois tipos: síncronos (todos os flip-flops recebem o clock) e assíncronos (apenas o primeiro recebe).

Contadores Assíncronos

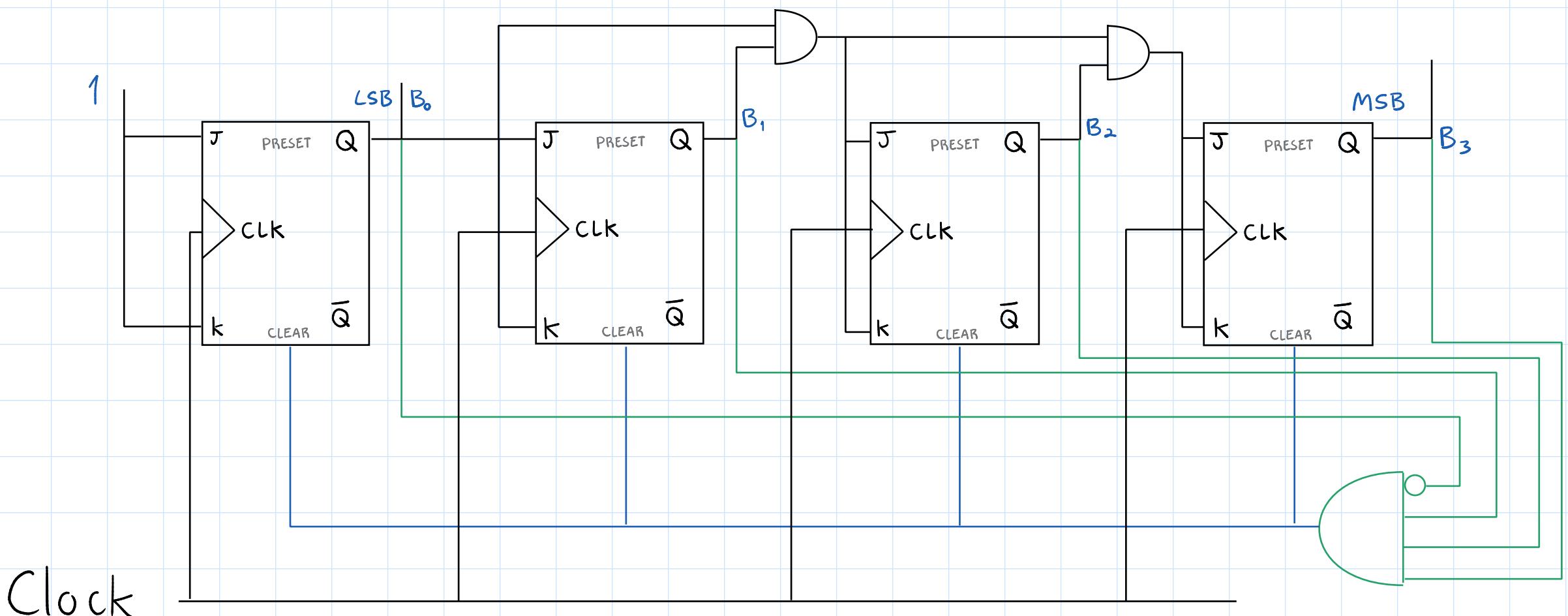
Contador com reset no 14 : (conta de 0 a 13)



Para todos os flip-flops, J e k recebem nível alto. O primeiro recebe o pulso do clock e os demais recebem a saída do flip-flop anterior como clock.

Contadores Síncronos

Contador com reset no 14 : (conta de 0 a 13)



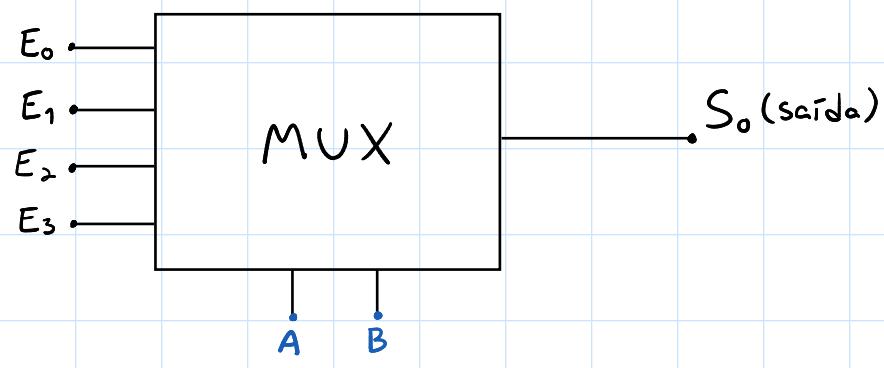
Todos os flip-flops recebem o pulso de clock, porém só o primeiro recebe nível alto em J e k. Nos demais, o J e o K dependem da conjunção das saídas de todos os flip-flops anteriores.

Multiplex e Demultiplex

Ambos servem para o controle de saídas e entradas.

Multiplex

Possui várias entradas mas uma só saída.

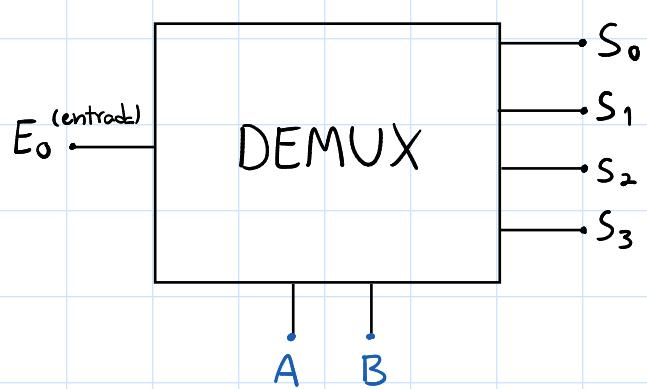


As variáveis A e B fazem a seleção de qual entrada passará para a saída. 2^n = número de entradas possíveis, sendo n o número de variáveis.

A	B	Entrada selecionada
0	0	E0
0	1	E1
1	0	E2
1	1	E3

Demultiplex

Possui uma entrada e várias saídas.

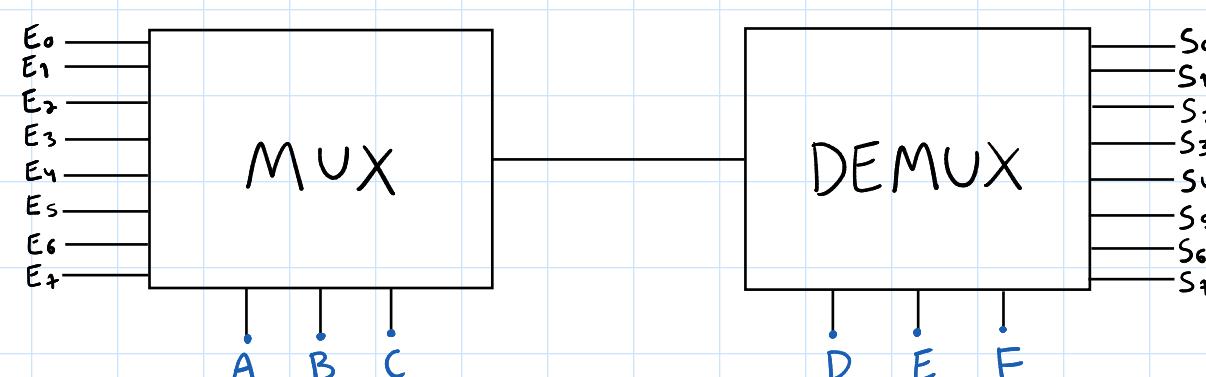


As variáveis A e B fazem a seleção de qual saída receberá a entrada. 2^n = número de saídas possíveis, sendo n o número de variáveis.

A	B	Saída selecionada
0	0	S0
0	1	S1
1	0	S2
1	1	S3

Exemplo de exercício:

Quais devem ser os valores das variáveis para que a entrada E5 saia em S3?



$$S = 101$$

$$E_5 = \begin{smallmatrix} 1 & 0 & 1 \\ A & B & C \end{smallmatrix}$$

$$3 = 011$$

$$S_3 = \begin{smallmatrix} 0 & 1 & 1 \\ D & E & F \end{smallmatrix}$$

Resposta: A=1, B=0, C=1,
D=0, E=1, F=1

Memórias

Memória ROM (Read Only Memory)

Pode ser lida, mas não reprogramada facilmente.

* MROM (máscara)

- Programada por máscara quando fabricada, não pode ser reprogramada.

* EPROM (eletricamente programável)

- Pode ser programada, apagada e reprogramada indefinidamente.
- Programável com eletricidade.
- Apagável com ultravioleta.

* FLASH

- Mais baratas que os EEPROM
- Mais usadas atualmente (SSDs)

* PROM (programável)

- Pode ser programada uma única vez.
- Possui fusíveis que são seletivamente queimados durante a programação.

* EEPROM (eletricamente apagável e programável)

- Igual à EPROM, mas pode ser reprogramada eletricamente, sem necessidade de ultravioleta.

Usos das ROMs:

- Firmwares (CDs, brinquedos, máquinas, eletrodomésticos...)

Memória RAM (Random Access Memory)

Usadas para armazenamento temporário de dados. Diferentemente das ROMs, são voláteis e seus dados se perdem quando param de receber energia.

* Dinâmicas (DRAM)

- Se fundamentam no uso de capacitores MOS, que são periodicamente recarregados para que não percam as informações, processo chamado de refresh.
- Baixo consumo de energia.
- Mais custo-benefício, mas não tão rápidas como as estáticas.
- Tipos: DIP, SIMM, DIMM, DDR.

* Estáticas (SRAM)

- Se baseiam em flip-flops e precisam de alimentação constante.
- Mais rápidas, porém mais caras.

Aumento da Capacidade de Memórias

Todo módulo de memória tem um tamanho de palavra e uma quantidade de palavras, para descobrir a capacidade em bits da memória basta multiplicar estas duas informações. Além disso cada palavra precisa ter um endereço, a quantidade de bits de endereçamento deve prover combinações suficientes para que todas as palavras sejam representadas.

$$2^e = \text{número de palavras}$$

e = quantidade de bits de endereçamento

$$n \cdot q = \text{capacidade em bits}$$

n = nº de palavras

q = quantidade de palavras

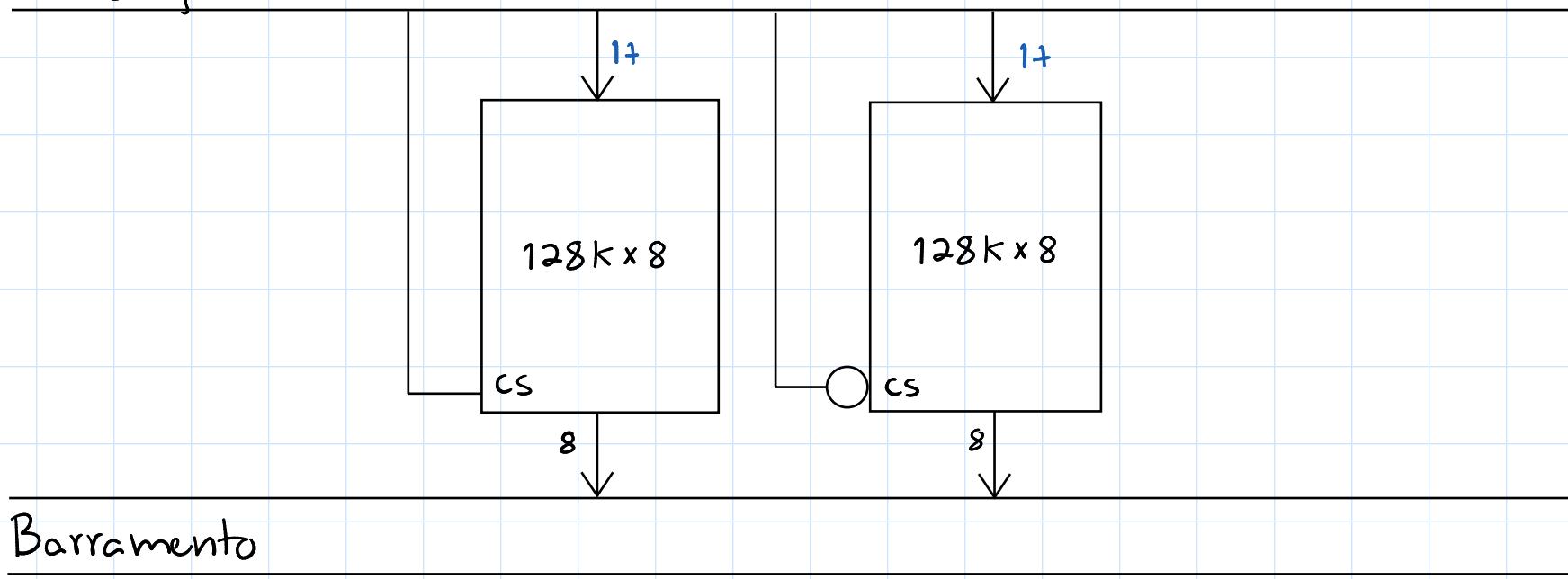
Podemos criar arranjos de módulos de memória para aumentar a quantidade e/ou tamanho das palavras.

Para aumentar a quantidade de palavras basta adicionar um novo módulo trabalhando separadamente. Além disso, um circuito de chip select para que os dois módulos não "falem" um por cima do outro.

Ex: $128\text{K} \times 8 \rightarrow 256\text{K} \times 8$

$$128 \cdot 1024 = 131072 = 2^{17}$$

Endereçamento ($17 + 1$ (para o chip select))

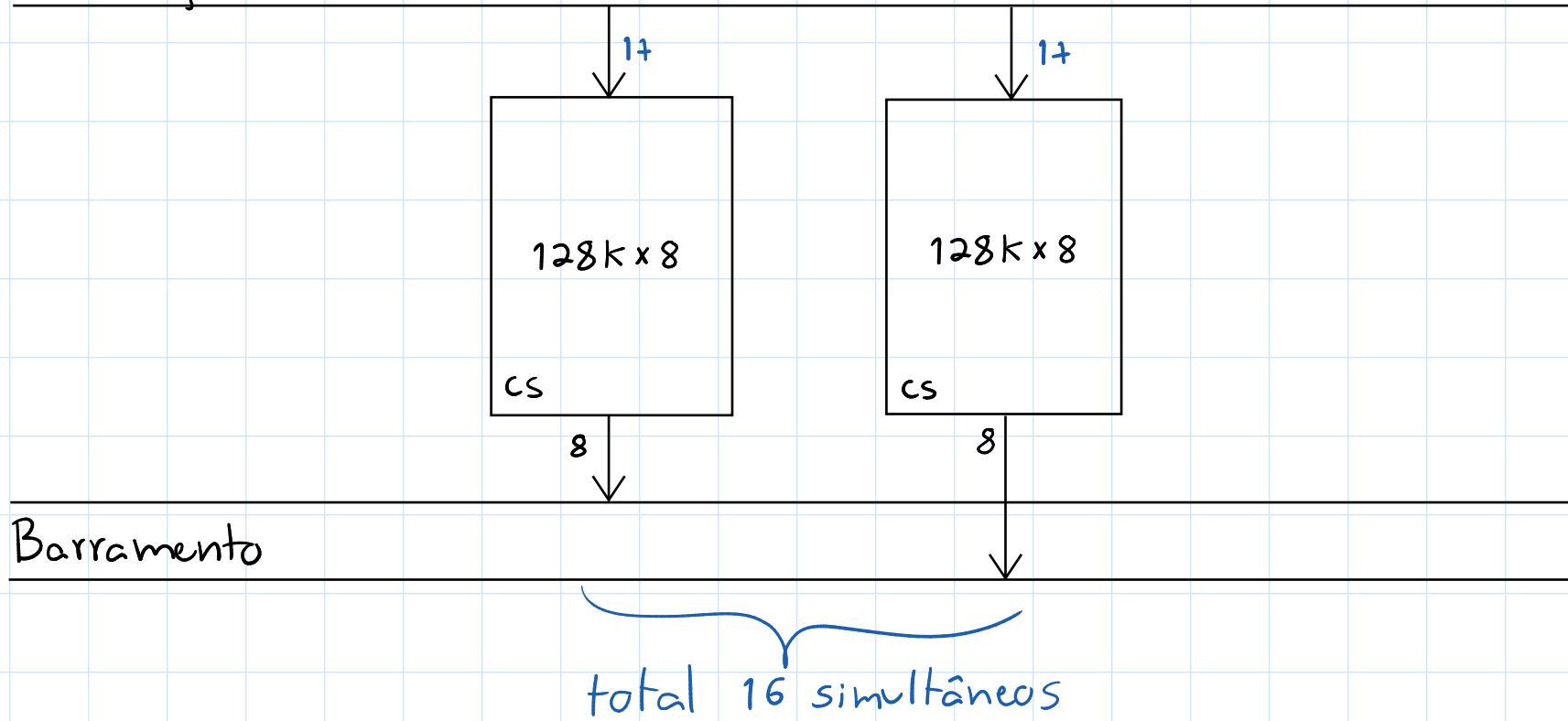


Para aumentar o tamanho das palavras precisamos colocar os dois módulos trabalhando simultaneamente, já que cada um armazenará metade de cada palavra.

Ex: $128\text{K} \times 8 \rightarrow 128\text{K} \times 16$

$$128 \cdot 1024 = 131072 = 2^{17}$$

Endereçamento (17)



Ex com ambos:

$128\text{K} \times 8 \rightarrow 256\text{K} \times 16$

$$128 \cdot 1024 = 131072 = 2^{17}$$

Endereçamento ($17 + 1$ (para o chip select))

