Professor ***Marco Câmara***

Arquitetura e Organização de Computadores

UCSAL 2025-01

Por que estudar isso ?

O *hardware* influencia em diversos aspectos

***Complexidade**: um computador é mais barato que muitos equipamentos específicos.*

***Capilaridade**: que tal um Instagram só para celular topo de linha?*

***Interação entre dispositivos**: desbloquear o Windows pelo celular.*

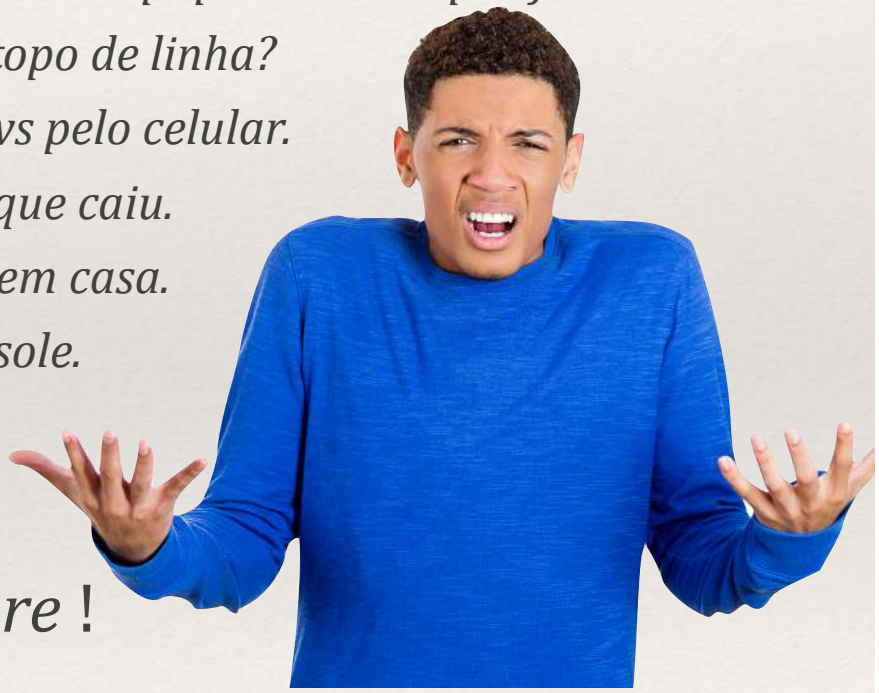
***Viabilidade**: alertar o serviço médico para um idoso que caiu.*

***Interação com o ambiente**: ligar o ar-condicionado em casa.*

***Interação com o Usuário**: celular controlando a console.*

***Custo**: "internet das coisas" em "coisas baratas"?*

***Funcionalidade**: o Waze roda no seu notebook?*



Eu sou do *software*, e não do *hardware* !

***Softwares* rodam no *hardware*, não se esqueça !**

Automação e Informação

O trabalho humano era baseado na força física

Apoio: ferramentas, forças naturais e animais.

Máquinas só apareceram na Revolução Industrial

Realizavam trabalhos de força física, velocidade ou precisão;

O trabalho intelectual não era o objetivo;

Apoio: outros humanos.

No último século, máquinas no trabalho intelectual

Computadores são máquinas cada vez mais eficazes neste trabalho;

Um dia assumirão todo o trabalho humano?

A matéria-prima e o produto final mudaram

Agora as máquinas manipulam informações.

Analógico X Digital

Informações não têm existência física

Livros, Fotografias, DVDs e Pendrives não são informações, e sim mídias;

*Informações precisam ser **representadas.***

Informações podem ser Digitais ou Analógicas

Informações Digitais

Valores discretos, variações finitas e completude limitada;

Poucas informações do nosso dia-a-dia são digitais (contagens, datas, gênero, estado civil, informações de estado de dispositivos, por exemplo).

Informações Analógicas

Valores contínuos, variações infinitas, completude;

O mundo está cheio de informações analógicas - todas as principais grandezas.

Por que informações digitais ?

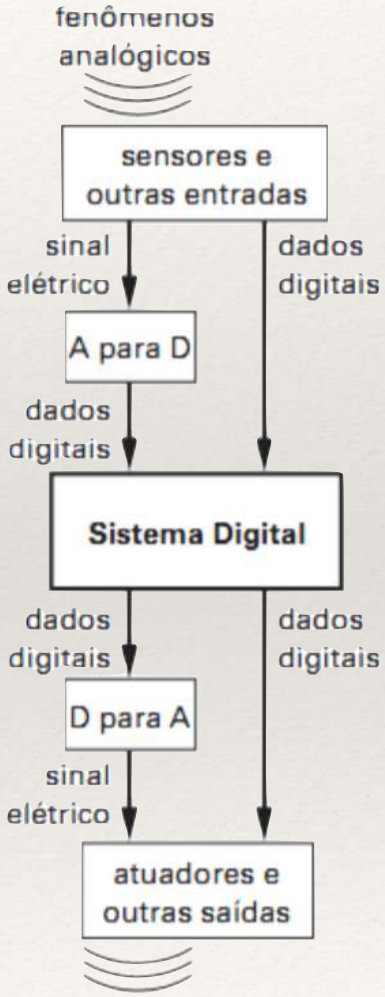
O objetivo é a simplicidade

Captar, armazenar, processar e entregar informações analógicas normalmente é mais complicado;

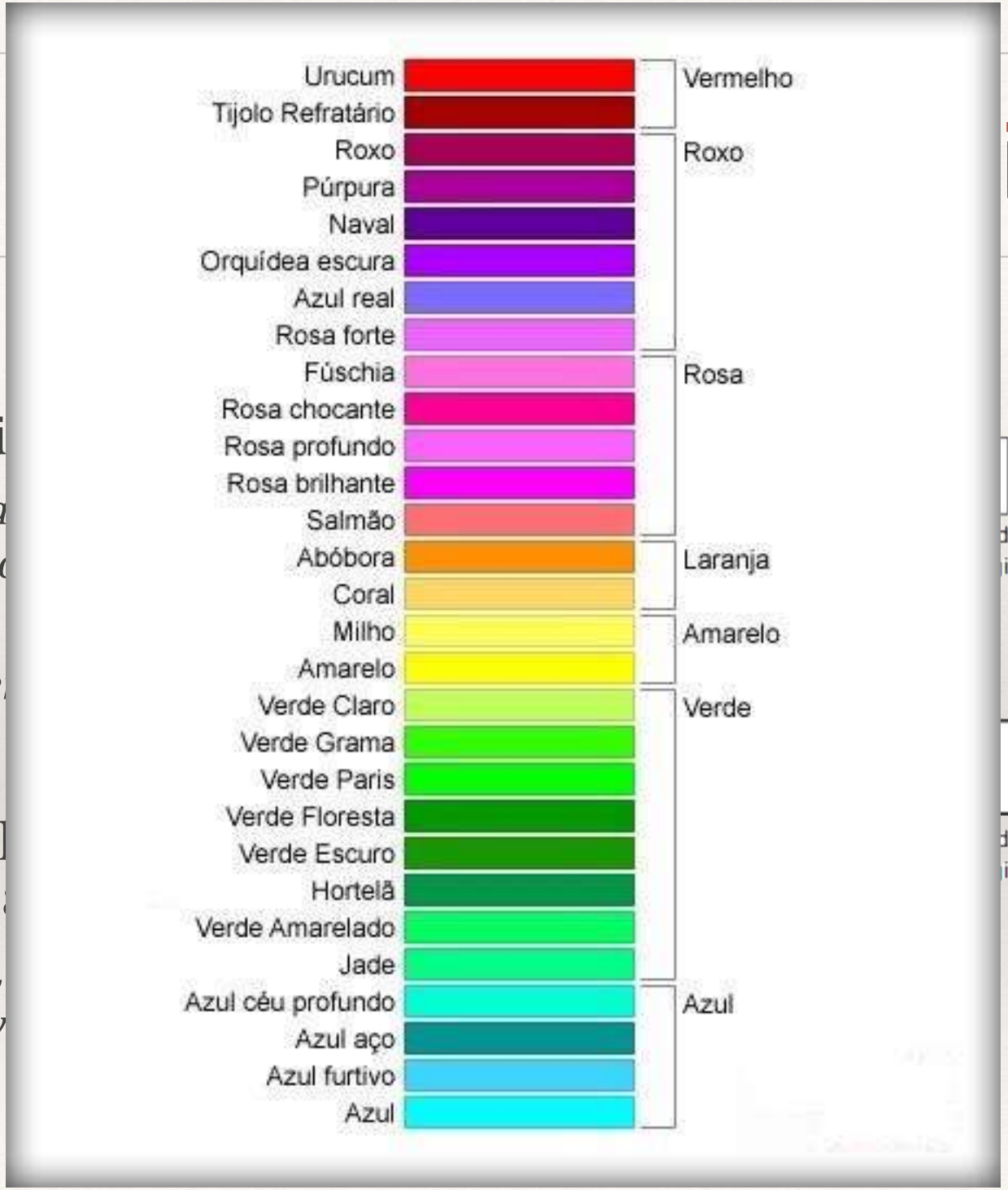
A exceção é a interpretação de informações pelo ser humano;

Computadores precisam lidar com informações analógicas

Captar, converter, processar e armazenar, e muitas vezes converter de novo para devolver no mesmo formato;



Por que



O objetivo é a si

Captar, armazena
informações anal
complicado;

A exceção é a inte
ser humano;

Computadores
informações an

Captar, converter,
muitas vezes conv
mesmo formato;

Representando Informações Numéricas

- Nos primórdios, a contagem
  - *Quantas cabeças de gado?*
  - *Quantas "luas" durará a viagem?*
  - *Quantos guerreiros a tribo adversária possui?*
- Avaliação qualitativa
  - *"Nenhum", "poucos" ou "muitos";*
- Comparação com os dedos das mãos
- Representações para quantidades maiores
  - *Métodos ineficazes, como os números romanos e unidades baseadas no corpo humano*; (você já tentou *somar dois números romanos*?)
  - *Sistema de numeração Indo-Arábico*

O corpo humano como referência

O cúbito, utilizado pelos egípcios

- Distância entre o cotovelo e a ponta do dedo médio do faraó: 52,4cm;
- A referência mudava para a Suméria (39,5cm) e Assíria (54,9cm);
- As pirâmides foram construídas com base nisto !

Milha, Jarda, Pé e Polegada etc

- Unidades utilizadas na Inglaterra e EUA, mas algumas ainda têm alcance mundial;
- **Milha** (*mile*): 1000 passos de um centurião romano: 1.609,34m;
- **Jarda** (*yard*): distância entre o nariz e a ponta do polegar com o braço esticado (com base nas medidas do Rei Henrique I): 3 pés, ou 91,44cm;
- **Pé** (foot): 12 polegadas, ou 30,48cm;
- **Polegada** (inch): 2,54cm;

Há uma tendência clara de convergência internacional para o SI (Sistema Internacional de medidas)

- O uso de sistemas diferentes causa diversos problemas;
- A NASA perdeu uma sonda (MCO - Mars Climate Orbiter) por conta disto.

Sistemas de Numeração

Notação Posicional

Conceito de "Base"

- *Qualquer quantidade de símbolos igual ou maior que 2;*
- *Sistema atual é decimal; computadores usam o binário;*
- *"Contando" em outras bases*

Unidades de Armazenamento Binárias

- *bit, nibble, Byte e múltiplos*

Conversão entre bases

- *Sistema Posicional*
- *Métodos simplificados*

Sistemas de Numeração

Notação

Conceito

Qual

Siste

"Con

Unidade

bit, n

Convers

Siste

Méto

Ninguém acertou as 6 dezenas da Megasena. O valor acumulado foi

R$ 104.250.368,91

Sistemas de Numeração

Notação

Conceito

Qual

Siste

"Con

Unidade

bit, n

Convers

Siste

Méto

Ninguém acertou as 6 dezenas da Megasena. O valor acumulado foi

R$ 104.250.368,91

Oito reais

Sistemas de Numeração

Notação

Conceito

Qual

Siste

"Con

Unidade

bit, n

Convers

Siste

Méto

R$ 104.250.368,91

Uma centena de milhão de reais (1×10^8 reais)
Zero dezenas de milhão de reais (0×10^7 reais)
Quatro milhões de reais (4×10^6 reais)
Duas centenas de milhares de reais (2×10^5 reais)
Cinco dezenas de milhares de reais (5×10^4 reais)
Zero milhares de reais (0×10^3 reais)
Três centenas de reais (3×10^2 reais)
Seis dezenas de reais (6×10^1 reais)
Oito reais (8×10^0 reais)
Nove décimos de real (9×10^{-1} reais)
Um centésimo de real (1×10^{-2} reais)

Mas como é que calculamos o valor do número?

Sistemas de Numeração

Notação

Conceito

Qual

Siste

"Con

Unidade

bit, n

Convers

Siste

Méto

R$ 104.250.368,91

Valor do algarismo

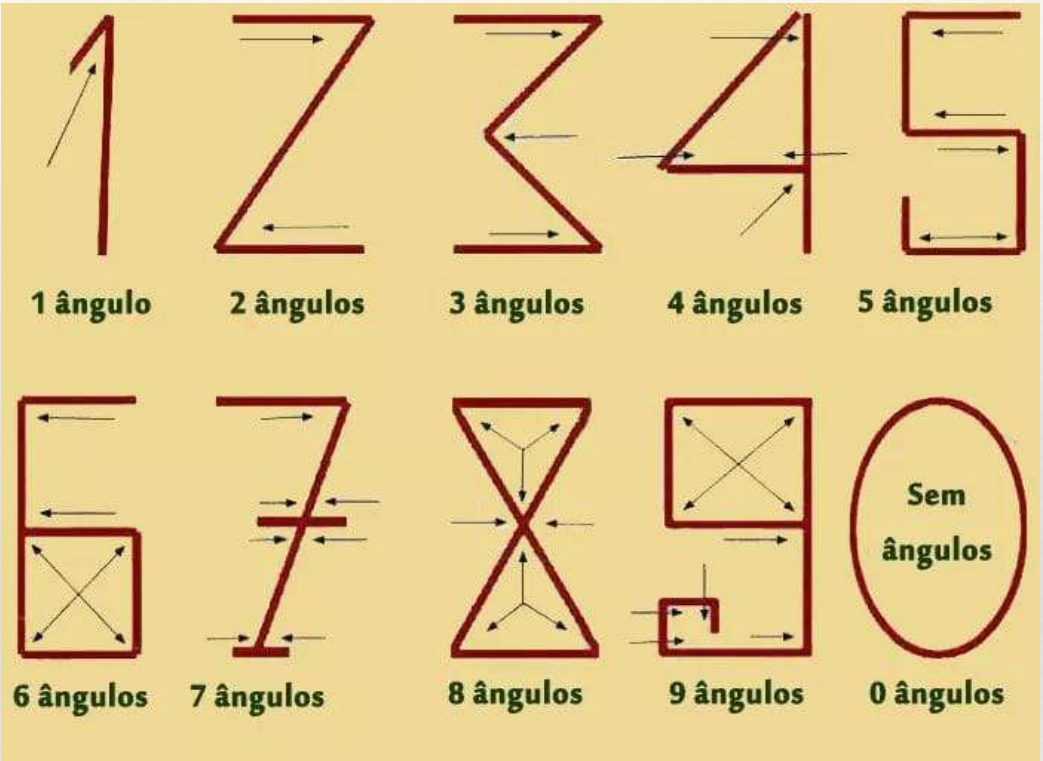
Uns (1 x 10^8 reais)
Zes (0 x 10^7 reais)
Quatro milhões de reais (4 x 10^6 reais)
Duas centenas de milhares de reais (2 x 10^5 reais)
Cinco dezenas de milhares de reais (5 x 10^4 reais)
Zero milhares de reais (0 x 10^3 reais)
Três centenas de reais (3 x 10^2 reais)
Seis dezenas de reais (6 x 10^1 reais)
Oito reais (8 x 10^0 reais)
Nove décimos de real (9 x 10^{-1} reais)
Um centésimo de real (1 x 10^{-2} reais)

Valor do Algarismo

Por que usamos os símbolos atuais para representar as quantidades?

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Para todo “mistério”, existe uma explicação fácil e interessante;



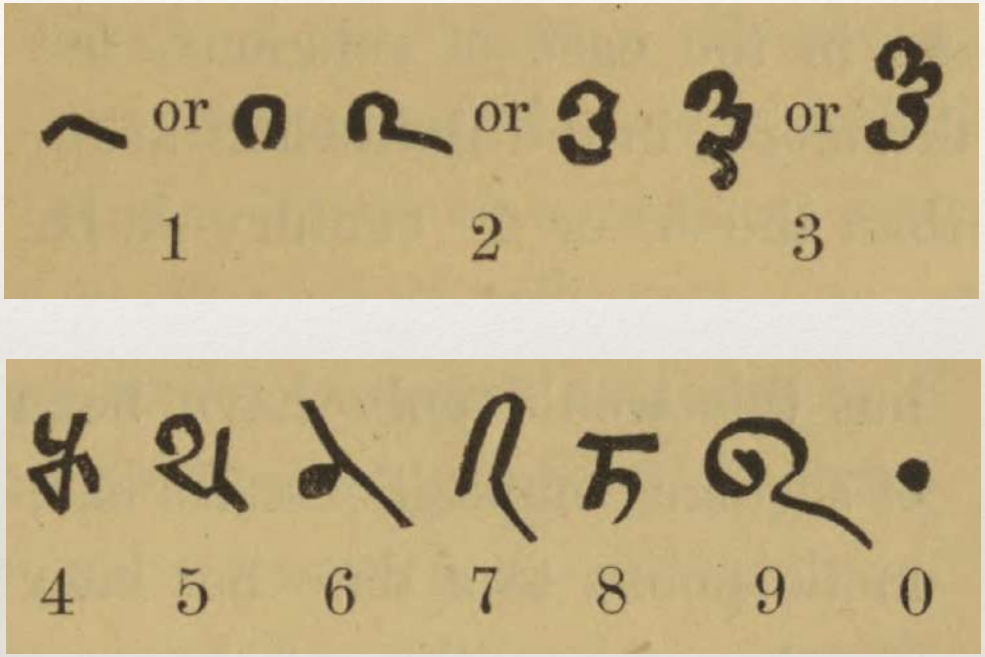
Valor do Algarismo

Por que usamos os símbolos atuais para representar as quantidades?

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Para todo "mistério", existe uma explicação fácil e interessante;

E uma explicação verdadeira, mas nem sempre tão interessante ...



Números Bakhshali

Sistemas de Numeração

Notação

Conceito

Qual

Siste

"Con

Unidade

bit, n

Convers

Siste

Méto

8 7 6 5 4 3 2 1 0 -1 -2

R$ 104.250.368,91

Uma centena de milhão de reais (1 x 10^8 reais)
Zero dezenas de milhão de reais (0 x 10^7 reais)
Quatro milhões de reais (4 x 10^6 reais)
Duas centenas de milhares de reais (2 x 10^5 reais)
Cinco dezenas de milhares de reais (5 x 10^4 reais)
Zero milhares de reais (0 x 10^3 reais)
Três centenas de reais (3 x 10^2 reais)
Seis dezenas de reais (6 x 10^1 reais)
Oito reais (8 x 10^0 reais)
Nove décimos de real (9 x 10^{-1} reais)
Um centésimo de real (1 x 10^{-2} reais)

10^5

Valor da posição

Sistemas de Numeração

Notação

Conceito

Qual

Siste

"Con

Unidade

bit, n

Convers

Siste

Méto

R$ 104.250.368,91

$1 \times 10^{8}+0 \times 10^{7}+4 \times 10^{6}+2 \times 10^{5}+5 \times 10^{4}+0 \times 10^{3}+3 \times 10^{2}+6 \times 10^{1}+8 \times 10^{0}+9 \times 10^{-1}+1 \times 10^{-2}$

$$\text{Valor do número} = \sum \text{Valores de Posição}$$

$$\text{Valor de Posição} = V_a \times \text{Base}^{p}$$

V_a = Valor do algarismo
p = Posição

Sistemas de Numeração

Notação

Conceito

Qual

Siste

"Con

Unidade

bit, n

Convers

Siste

Méto

R$ 104.250.368,91

Uma centena de milhão de reais (1 x 10^8 reais)
Zero dezenas de milhão de reais (0 x 10^7 reais)
Quatro milhões de reais (4 x 10^6 reais)
Duas centenas de milhares de reais (2 x 10^5 rea
Cinco dezenas de milhares de reais (5 x 10^4 reais)
Zero milhares de reais (0 x 10^3 reais)
Três centenas de reais (3 x 10^2 reais)
Seis dezenas de reais (6 x 10^1 reais)
Oito reais (8 x 10^0 reais)
Nove décimos de real (9 x 10^{-1} reais)
Um centésimo de real (1 x 10^{-2} reais)

10^5

Base

Por que a base 10?

10 dedos da mão;

Primeiro método de representação numérica?

Estamos culturalmente treinados a utilizá-lo em tudo;

Para os dispositivos computacionais, não é o mais simples;

Outras bases de numeração

Divisão do tempo: bases 60, 12 e 2?

- Segundos, minutos e horas: Sumérios;
- Sumérios usavam o sistema sexagesimal e duodecimal;
- 60 - o número mágico, divisível por 1, 2, 3, 4 e 5

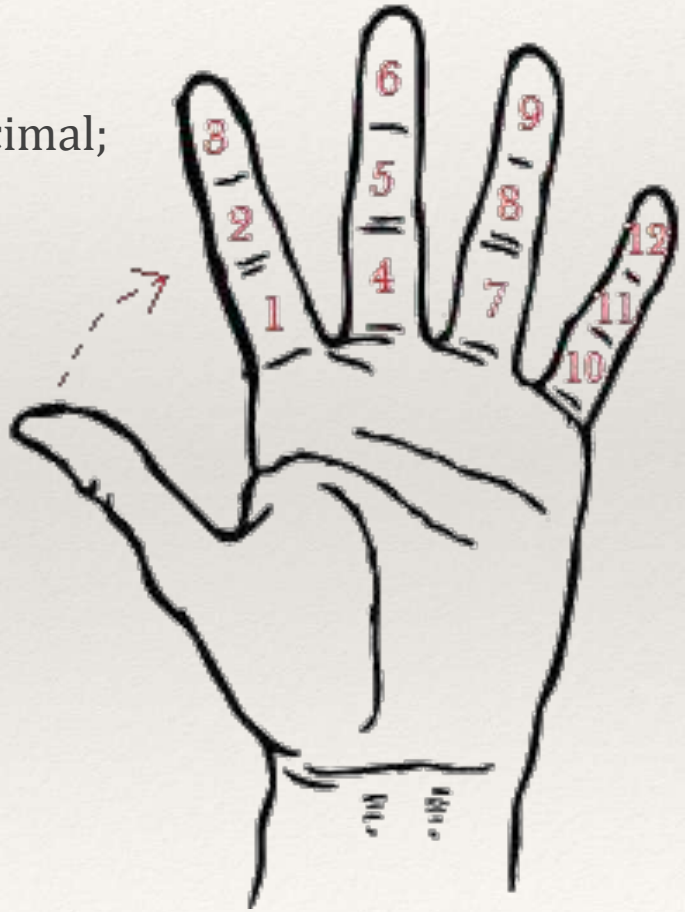
Outras bases de numeração

Divisão do tempo: bases 60, 12 e 2?

- Segundos, minutos e horas: Sumérios;
- Sumérios usavam o sistema sexagesimal e duodecimal;
- 60 - o número mágico, divisível por 1, 2, 3, 4 e 5

12 divisões do dedo humano, e 60 ÷ 5

- 12 horas dividiam a parte clara e escura dos dias
- Até hoje utilizamos as 12 horas, e 2 turnos



Outras bases de numeração

Divisão do tempo: bases 60, 12 e 2?

- Segundos, minutos e horas: Sumérios;
- Sumérios usavam o sistema sexagesimal e duodecimal;
- 60 - o número mágico, divisível por 1, 2, 3, 4 e 5

12 divisões do dedo humano, e 60 ÷ 5

- 12 horas dividiam a parte clara e escura dos dias
- Até hoje utilizamos as 12 horas, e 2 turnos

Huygens cria o minuto (1670?)

- Cientista Holandês brilhante
- Inventor do pêndulo
- 1800: o pequeno de segunda ordem

E se usássemos outra base?

Vamos supor oito dedos. Como seria a contagem usando a notação posicional, e os mesmos símbolos?

0	10	21	101
1	11	22	102
2	12	23	103
3	13	24	104
4	14	[...]	105
5	15	75	106
6	16	76	107
7	17	77	110
...	20	100	111
			...

A base binária

De todas as bases, a mais simples é a binária

- Tem apenas dois símbolos, a menor quantidade possível;
- Por outro lado, há um aumento significativo do número de dígitos;
- Computadores têm facilidade para lidar com números extensos, porém simples;
- Humanos têm dificuldade para lidar com números extensos.

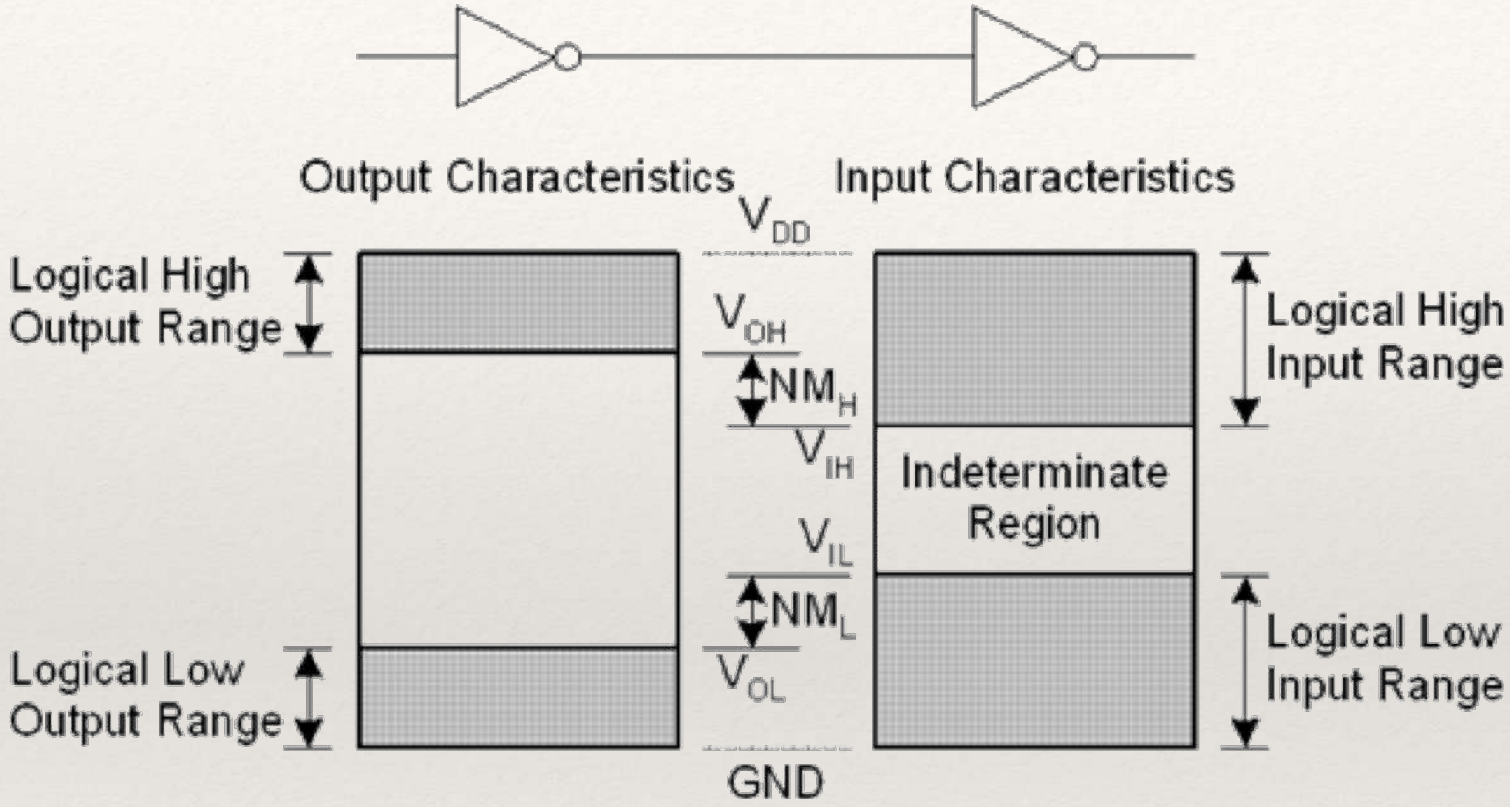
Quantidades representadas não mudam

- A base só afeta a representação.

Bases potências de 2 simplificam a representação

- Bases mais comuns nesta função são o octal e o hexadecimal.

Representação Binária



Representação interna de valores binários

- Faixas de Valores de 0 e 1
- Faixas grandes suportam erros
- O risco está na "região indeterminada", que dificilmente é utilizada

Bases potências de 2

Números binários grandes são difíceis para humanos. Por isto é comum usarmos representações em bases que são potências de 2, como 8(2^3), ou mais frequentemente, 16(2^4). Vejam como exemplo, novamente, o saldo acumulado da Mega Sena, em binário, e suas representações em octal e hexadecimal:

110 001 101 101 011 110 000 000 000 . 111 010 001 111 010 111 ⇒ Binário

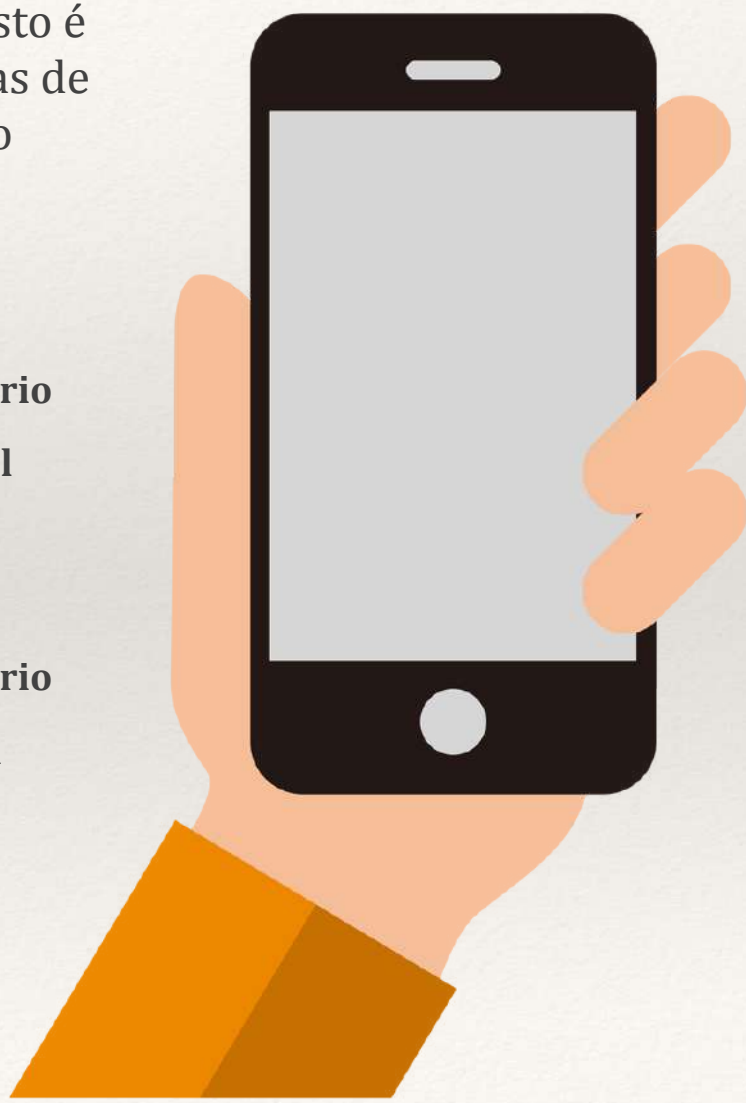
6 1 5 5 3 6 0 0 0 . 7 2 1 7 2 7 ⇒ Octal

$(615536000.721727)_8$

110 0011 0110 1011 1100 0000 0000 . 1110 1000 1111 0101 11 ⇒ Binário

6 3 6 B C 0 0 . E 8 F 5 ⇒ Hexa

$(636BC00.E8F5)_{16}$, ou simplesmente 636BC00.E8F5h



Unidades Binárias

A contagem em binário envolverá sempre muitos dígitos, devido à pequena quantidade de símbolos disponíveis;

O valor acumulado no prêmio da Megasena, por exemplo (R$ 104.250.368,91, que já era bem grande), ficaria assim em binário:

11000110110101111000000000.11101000111101011...

E como ficam os múltiplos em binário?

Não temos mil, milhão, bilhão etc, e sim:

1 nibble = 4 bits;

1 Byte = 8 bits;

1 KiloByte = 2^{10} (1024) Bytes;

1 MegaByte = 2^{10} KB = 2^{20} Bytes

1 GigaByte = 2^{30} Bytes;

1 TeraByte = 2^{40} KB

1 PetaByte = 2^{50} Bytes;

[...]

Os múltiplos com potências de 2 só valem para o Byte ! Todas as demais medidas utilizam os múltiplos decimais (Ex. 1 Megapixel = 1 milhão de pixels, 1 Gigabit = 1 bilhão de bits etc).

0
1
10
11
100
101
110
111
1000
1001
1010
1011
1100
[...]

E se usássemos outra base?

Vamos supor oito dedos. Como seria a contagem usando a notação posicional, e os mesmos símbolos?

0	10	21	101
1	11	22	102
2	12	23	103
3	13	24	104
4	14	[...]	105
5	15	75	106
6	16	76	107
7	17	77	110
...	20	100	111
			...

Que quantidade o número $(111)_8$ representa na base 10?

$1 \times 8^2 + 1 \times 8^1 + 1 \times 8^0 =$

$1 \times 64 + 1 \times 8 + 1 \times 1 =$

$64 + 8 + 1 = 73$

E se usássemos outra base?

Vamos supor oito dedos. Como seria a contagem usando a notação posicional, e os mesmos símbolos?

0	10	21	101
1	11	22	102
2	12	23	103
3	13	24	104
4	14	[…]	105
5	15	75	106
6	16	76	107
7	17	77	110
…	20	100	111
			…

E no caso oposto? Como representar a quantidade 62 na base 8?

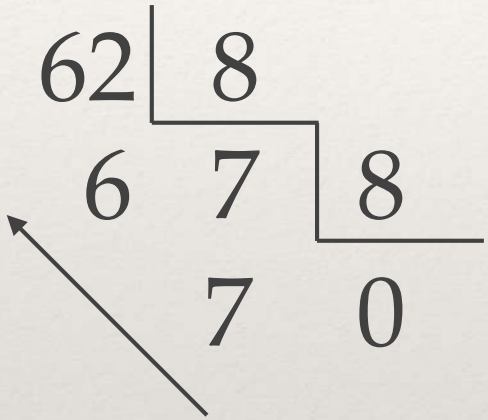
$8^3 = 512$; 62 é menor que 512, logo não há dígito na posição 3;

$8^2 = 64$; 62 é menor que 64, logo não há dígito na posição 2;

$8^1 = 8$; 62 é maior que 8; procuro o maior produto de 8 menor que 62;
8 x 7 = 56 ⇒ 7 é o algarismo na posição 1. Mas ainda faltam 6!

$8^0 = 1$; 6 = 1 x 6 ⇒ 6 é o algarismo na posição 1

Método simplificado



$$62 = (76)_8$$

Divide-se repetidamente o número na base 10 pelo valor da base, até obter o quociente zero, preservando-se os restos de cada divisão;

O número na nova base será formado pelos restos lidos de baixo para cima;

Na prática, o método apenas reproduz o conceito da Notação Posicional.

Representação Binária

- Binários podem representar qualquer número real (R)
  - Isso inclui números naturais (N), inteiros (Z), racionais (Q) e irracionais (I);
  - A representação de cada um destes números na memória, e nas unidades aritméticas pode ser diferente;
  - Cabe ao programador escolher a representação mais adequada em função dos valores que serão manipulados ou armazenados;
- Inteiros negativos são expressos em “Complemento de 2"
  - Bit mais significativo é reservado para o sinal (1 = Negativo);
  - Para representar um número negativo, subtraímos 1 do número positivo correspondente, e invertemos todos os bits do número;
  - Veja um exemplo com o número -5:

0101 (5 em binário) → 0100 (subtraindo-se 1 do número) →

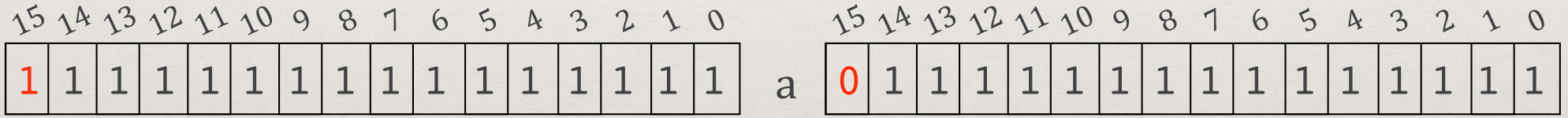
1011 (invertendo todos os bits)

Representação Binária

Números negativos (cont.)

- Lembrar que um bit é reservado para o sinal;
- Variáveis inteiras para armazenar números negativos são *signed*. Se não for armazenar números negativos, pode ser *unsigned*;

`signed int`: -32.768 a 32.767; `unsigned int`: 0 a 65.536



Números racionais e irracionais podem ser representados de duas formas

- Ponto decimal fixo;
- Ponto decimal flutuante.

Representação Binária

Números fracionários com ponto fixo

- O exemplo da Mega-Sena utilizou este formato - seguir notação posicional.

Números com ponto flutuante

- Similar à notação científica; permite números precisos com poucos dígitos;
- Para números mais precisos, mais dígitos na mantissa;
- Para números maiores (ou menores), mas dígitos no expoente;
- Mantissa e expoente podem ser negativos (complemento de 2);
- Mantissa de números positivos começa com $(0,1)_2$; Positivos com $(1,0)_2$;

Distribuição de sinal, mantissa e expoente

- Posição e número de bits ocupados por cada parte segue diversos formatos;
- IEEE754 é uma proposta de norma;
- A complexidade não pára por aí. Computadores dispõem de processadores específicos só para processar este tipo de dado.

Representação Binária

Dízimas são um caso curioso

- Valores representados por dízimas em uma base podem não ser em outra;
- O número 0,1 em decimal é caso clássico. Alguns programas de computador, e até calculadoras mostram o valor 0,099999999...
- O valor acumulado da Mega Sena, por exemplo, tem duas casas decimais no sistema decimal, mas é uma dízima em binário;
- Um cuidado recomendável é aproximar resultados para o número de casas decimais exigidas, evitando falhas.

Binários também representam letras, símbolos e caracteres especiais

- Basicamente associamos os itens a valores binários específicos;
- A Tabela ASCII é o caso mais comum.

Tabela ASCII

A tabela tem aspectos curiosos, como o fato das minúsculas serem representadas pelo código das maiúsculas + código do espaço em branco;

Desafios como ordenação alfabética na verdade são operações aritméticas !

Bin	Oct	Dec	Hex	Sinal
0010 0000	040	32	20	(espaço)
0010 0001	041	33	21	!
0010 0010	042	34	22	"
0010 0011	043	35	23	#
0010 0100	044	36	24	$
0010 0101	045	37	25	%
0010 0110	046	38	26	&
0010 0111	047	39	27	'
0010 1000	050	40	28	(
0010 1001	051	41	29	)
0010 1010	052	42	2A	*
0010 1011	053	43	2B	+
0010 1100	054	44	2C	,
0010 1101	055	45	2D	-
0010 1110	056	46	2E	.
0010 1111	057	47	2F	/
0011 0000	060	48	30	0
0011 0001	061	49	31	1
0011 0010	062	50	32	2
0011 0011	063	51	33	3
0011 0100	064	52	34	4
0011 0101	065	53	35	5
0011 0110	066	54	36	6
0011 0111	067	55	37	7
0011 1000	070	56	38	8
0011 1001	071	57	39	9
0011 1010	072	58	3A	:
0011 1011	073	59	3B	;
0011 1100	074	60	3C	<
0011 1101	075	61	3D	=
0011 1110	076	62	3E	>
0011 1111	077	63	3F	?

Bin	Oct	Dec	Hex	Sinal
0100 0000	100	64	40	@
0100 0001	101	65	41	A
0100 0010	102	66	42	B
0100 0011	103	67	43	C
0100 0100	104	68	44	D
0100 0101	105	69	45	E
0100 0110	106	70	46	F
0100 0111	107	71	47	G
0100 1000	110	72	48	H
0100 1001	111	73	49	I
0100 1010	112	74	4A	J
0100 1011	113	75	4B	K
0100 1100	114	76	4C	L
0100 1101	115	77	4D	M
0100 1110	116	78	4E	N
0100 1111	117	79	4F	O
0101 0000	120	80	50	P
0101 0001	121	81	51	Q
0101 0010	122	82	52	R
0101 0011	123	83	53	S
0101 0100	124	84	54	T
0101 0101	125	85	55	U
0101 0110	126	86	56	V
0101 0111	127	87	57	W
0101 1000	130	88	58	X
0101 1001	131	89	59	Y
0101 1010	132	90	5A	Z
0101 1011	133	91	5B	[
0101 1100	134	92	5C	\
0101 1101	135	93	5D	]
0101 1110	136	94	5E	^
0101 1111	137	95	5F	_

Bin	Oct	Dec	Hex	Sinal
0110 0000	140	96	60	`
0110 0001	141	97	61	a
0110 0010	142	98	62	b
0110 0011	143	99	63	c
0110 0100	144	100	64	d
0110 0101	145	101	65	e
0110 0110	146	102	66	f
0110 0111	147	103	67	g
0110 1000	150	104	68	h
0110 1001	151	105	69	i
0110 1010	152	106	6A	j
0110 1011	153	107	6B	k
0110 1100	154	108	6C	l
0110 1101	155	109	6D	m
0110 1110	156	110	6E	n
0110 1111	157	111	6F	o
0111 0000	160	112	70	p
0111 0001	161	113	71	q
0111 0010	162	114	72	r
0111 0011	163	115	73	s
0111 0100	164	116	74	t
0111 0101	165	117	75	u
0111 0110	166	118	76	v
0111 0111	167	119	77	w
0111 1000	170	120	78	x
0111 1001	171	121	79	y
0111 1010	172	122	7A	z
0111 1011	173	123	7B	{
0111 1100	174	124	7C	|
0111 1101	175	125	7D	}
0111 1110	176	126	7E	~

A Máquina da Informação

As primeiras máquinas eram mecânicas

A máquina de Anticítera, datada do século I A.C., é o mais antigo computador analógico conhecido

27 engrenagens de bronze, feitas a mão;

Representam a órbita da Lua, de outros planetas e do próprio Sol;

Empregava uma engrenagem diferencial, supostamente inventada apenas no século XVI;

Relógios, que são mecanismos similares, só foram criados no século XVIII;

A maior engrenagem tinha 140 milímetros de diâmetro e originalmente teve 223 dentes.

A Máquina da Informação

As primeiras máquinas eram mecânicas

A máquina de Anticítera, datada do século I A.C., é o mais antigo computador analógico conhecido

27 engrenagens de bronze, feitas a mão;

Representam a órbita da Lua, de outros planetas e do próprio Sol;

Empregava uma engrenagem diferencial, supostamente inventada apenas no século XVI;

Relógios, que são mecanismos similares, só foram criados no século XVIII;

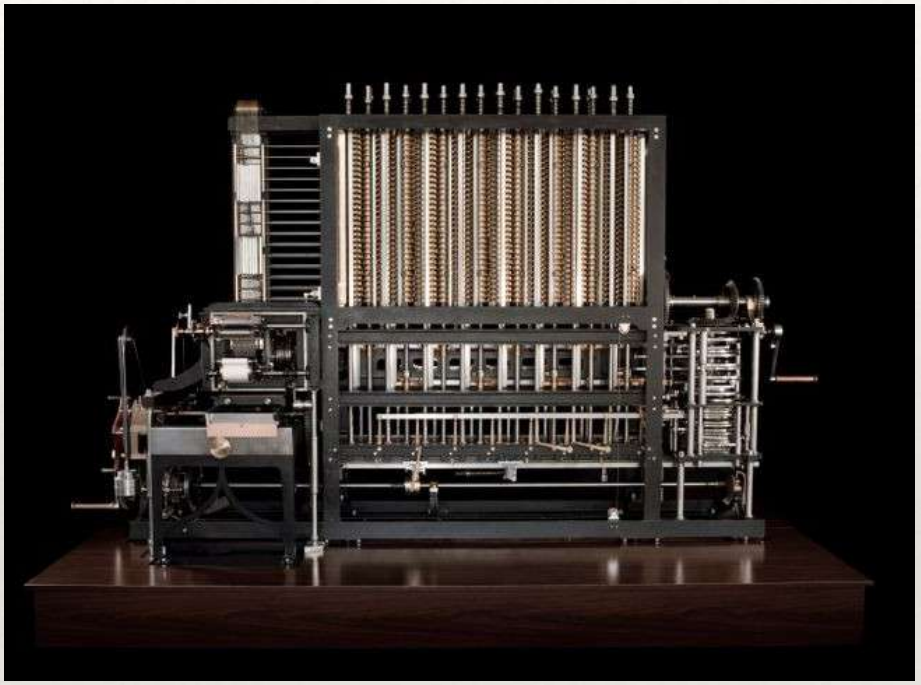
A maior engrenagem tinha 140 milímetros de diâmetro e originalmente teve 223 dentes.

Máquina de Babbage

Primeira máquina “programável”;

Possuía componentes atuais, como ULA (Unidade Lógica e Aritmética), Memória para 1.000 números de 40 dígitos decimais;

Foi desenvolvida em 1833, porém acabou não sendo implementada na época.



Computadores Eletromecânicos

O conceito do computador moderno surgiu em um trabalho de Alan Turing, em 1936;

A “Bomba eletromecânica”, ou simplesmente “Bombe”, foi desenvolvida em 1940 como parte dos esforços de guerra para quebrar a criptografia feita pela máquina nazista “Enigma";

Reconheceram a máquina do filme “O jogo da imitação” ?

Blocos Funcionais Eletromecânicos

Componentes Básicos dos primeiros dispositivos

Interruptores (chaves) como dispositivos de entrada;

Relês para "cascateamento" de saídas como novas entradas;

Lâmpadas como dispositivos de saída.

Representação de Valores Digitais

Chave (ou circuito) fechado = 1; Chave aberta = 0;

Lâmpada acesa = 1; Lâmpada apagada = 0.

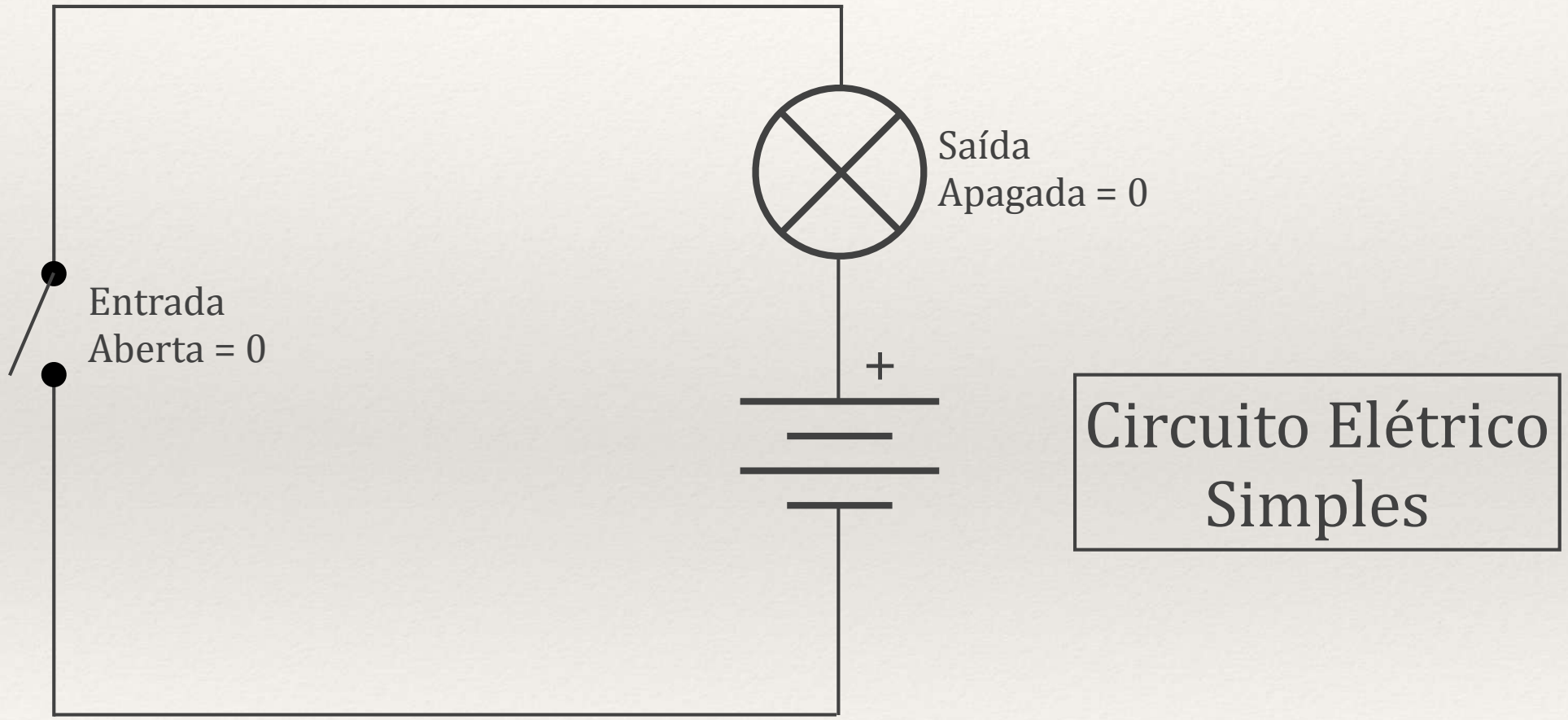
Problemas

Confiabilidade dos componentes (ex. lâmpada queimada era um 0?)

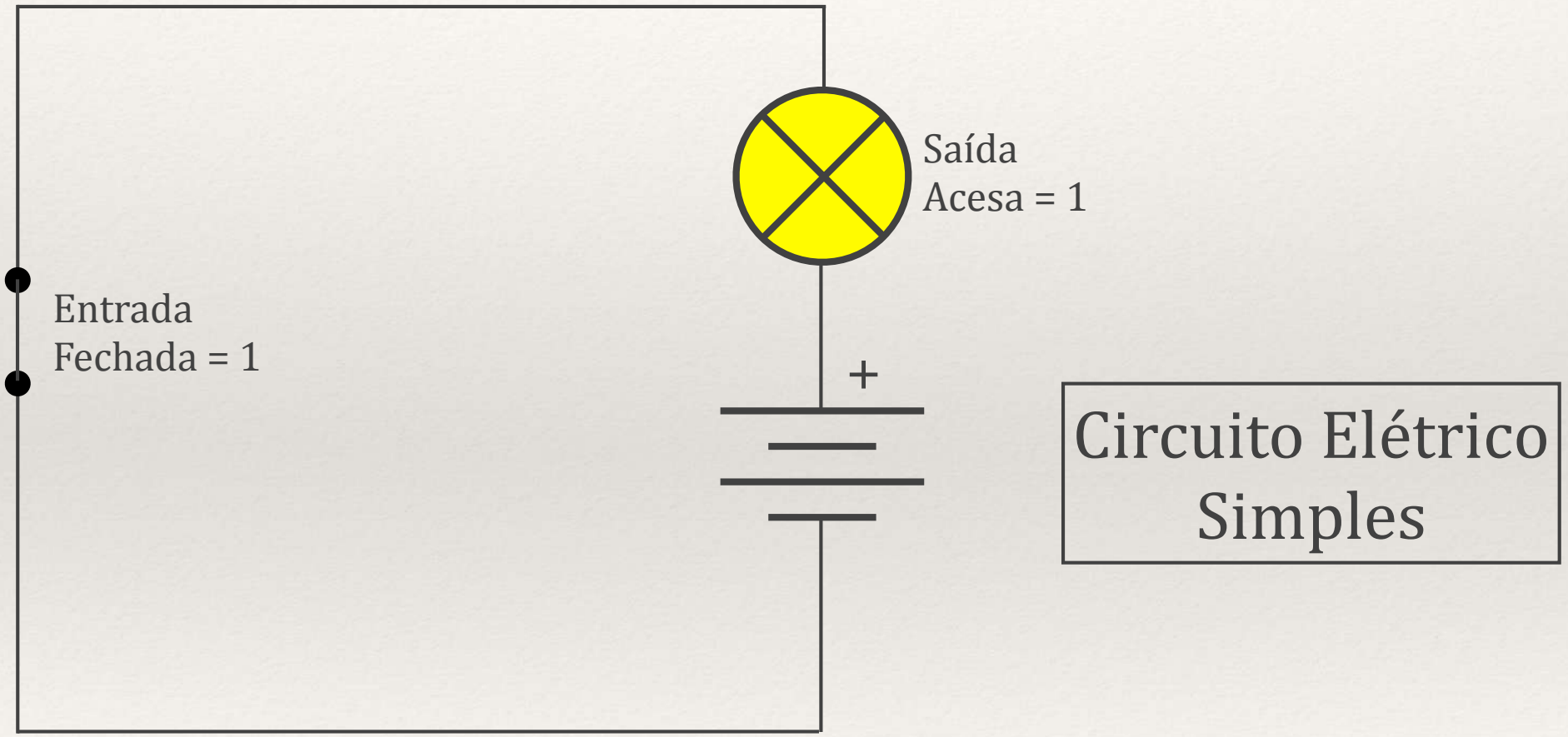
Performance limitada por desempenho mecânico dos relês;

Muito espaço ocupado, mesmo por circuitos simples.

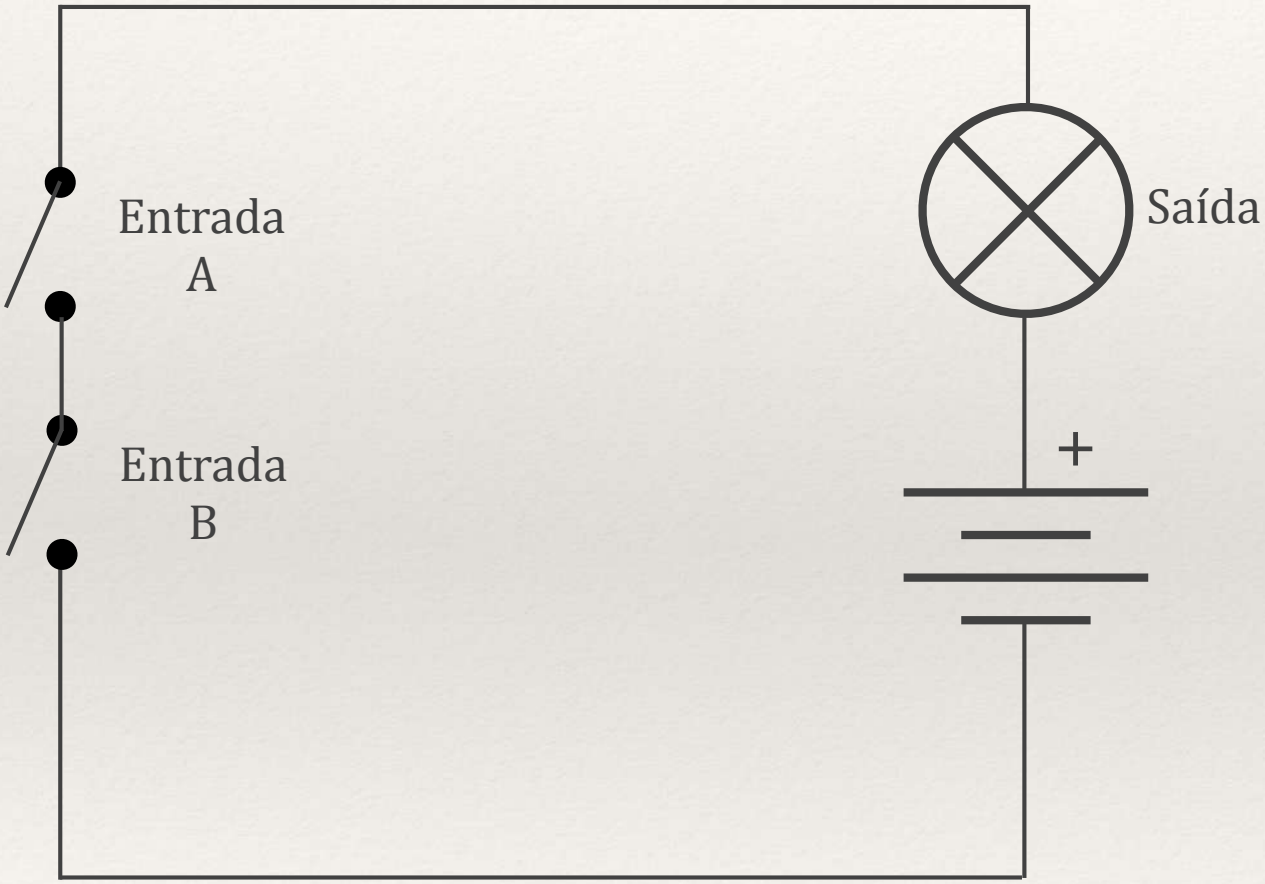
Blocos Funcionais Eletromecânicos



Blocos Funcionais Eletromecânicos



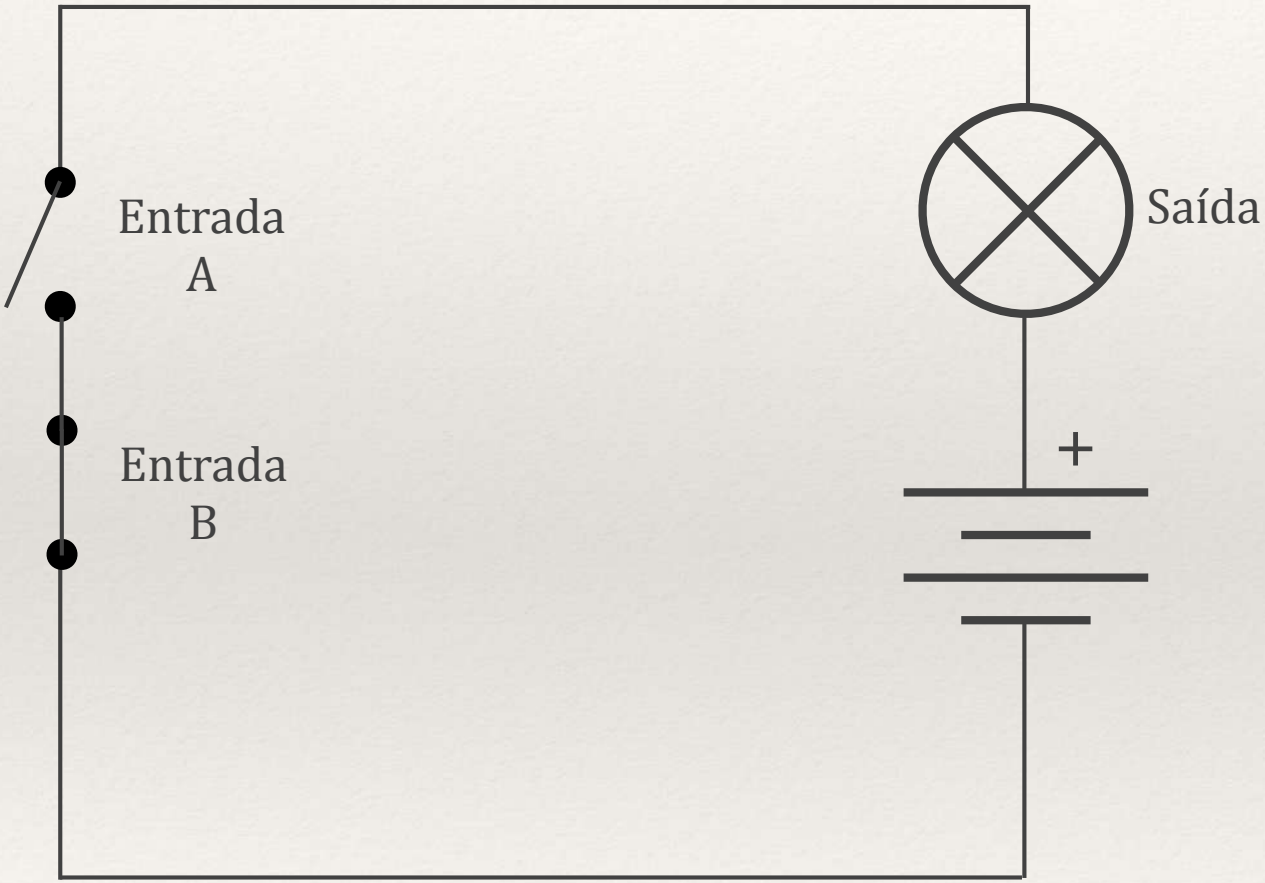
Blocos Funcionais Eletromecânicos



Lógica “E”
(AND)

A	B	S
0	0	0

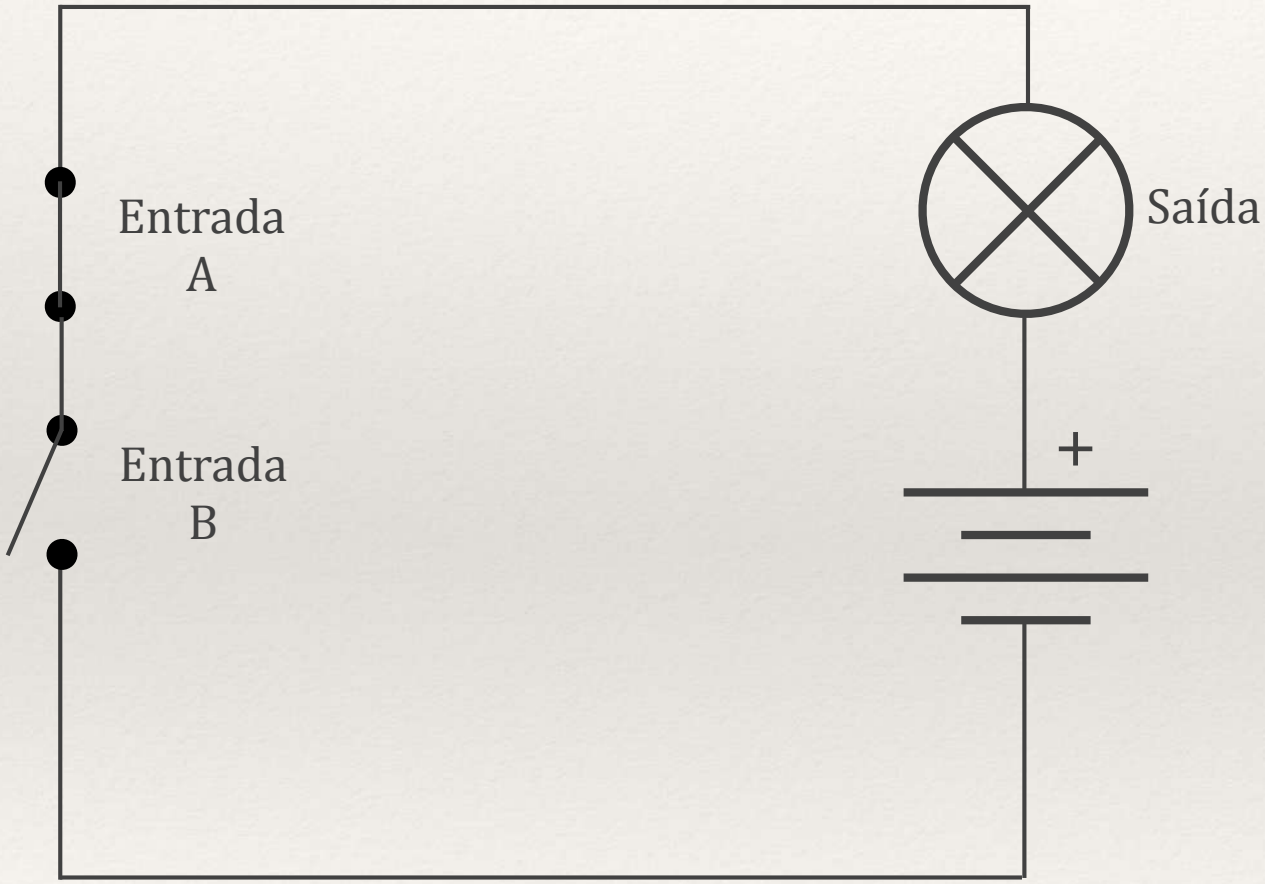
Blocos Funcionais Eletromecânicos



Lógica “E”
(AND)

A	B	S
0	0	0
0	1	0

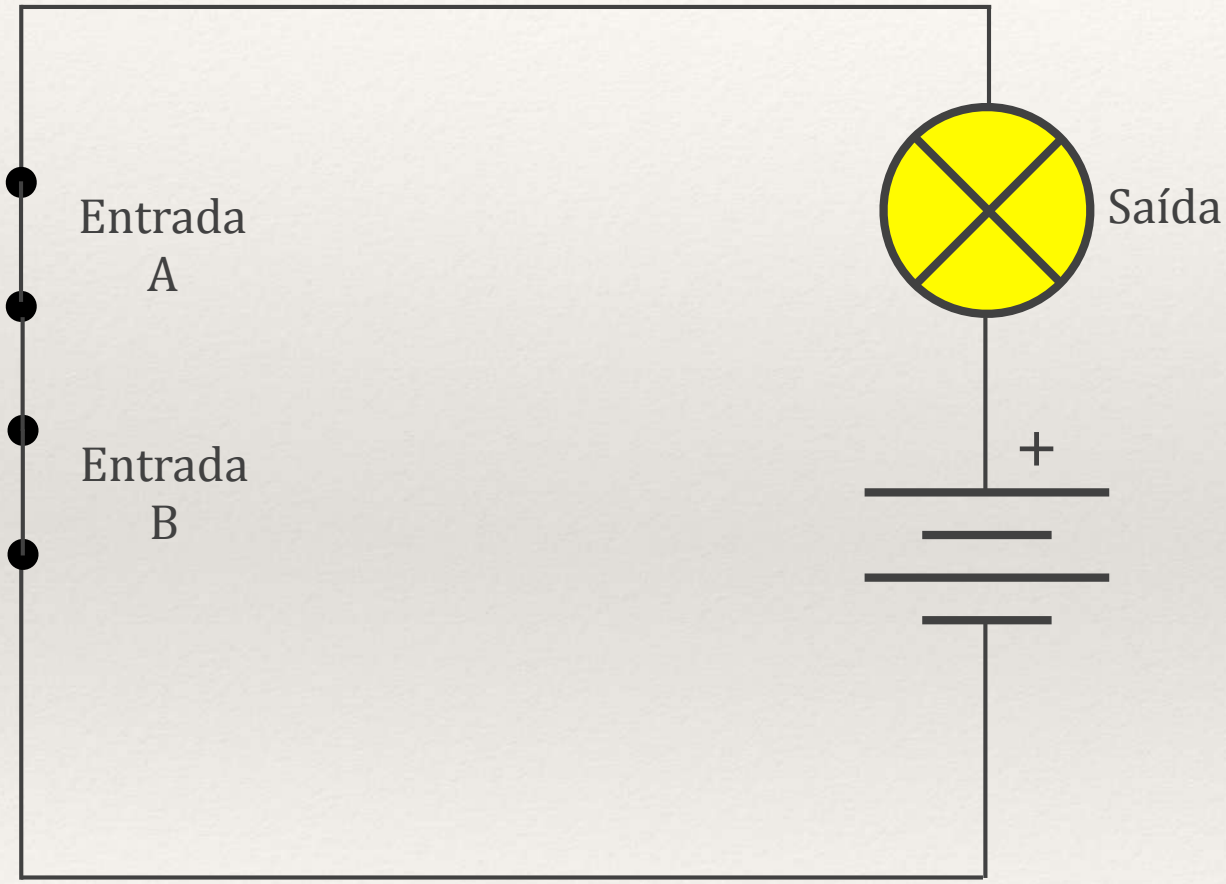
Blocos Funcionais Eletromecânicos



Lógica “E”
(AND)

A	B	S
0	0	0
0	1	0
1	0	0

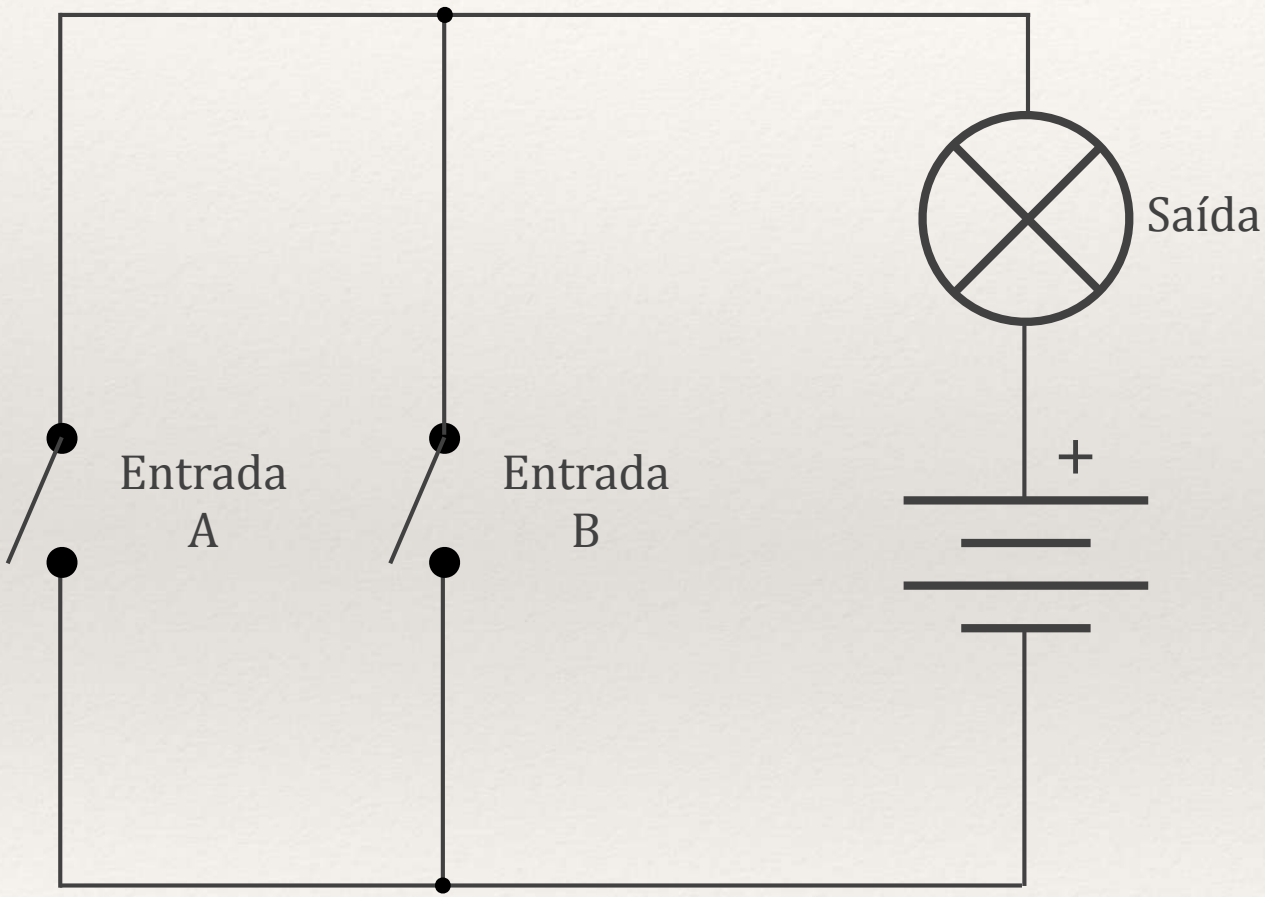
Blocos Funcionais Eletromecânicos



Lógica “E”
(AND)

A	B	S
0	0	0
0	1	0
1	0	0
1	1	1

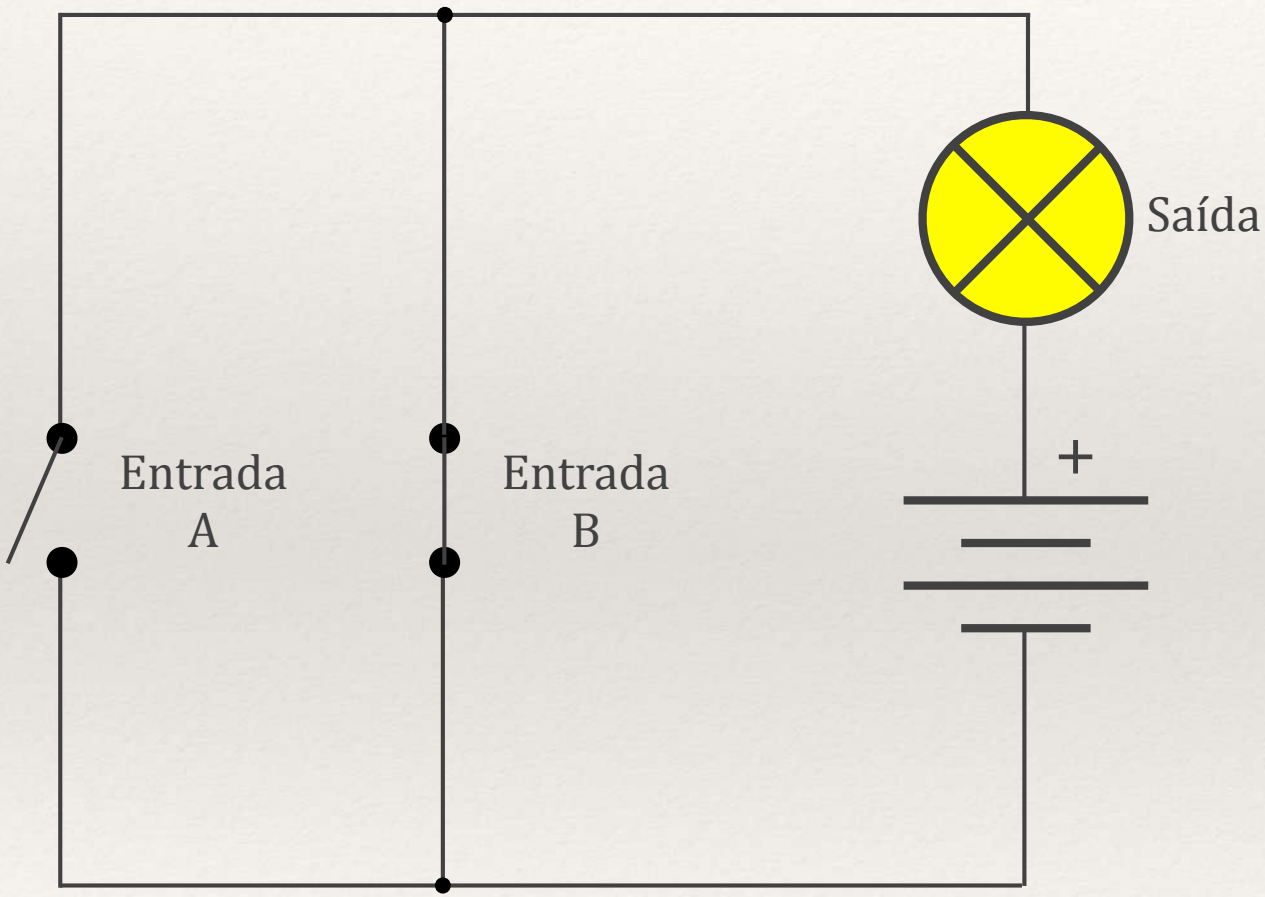
Blocos Funcionais Eletromecânicos



Lógica “OU”
(OR)

A	B	S
0	0	0

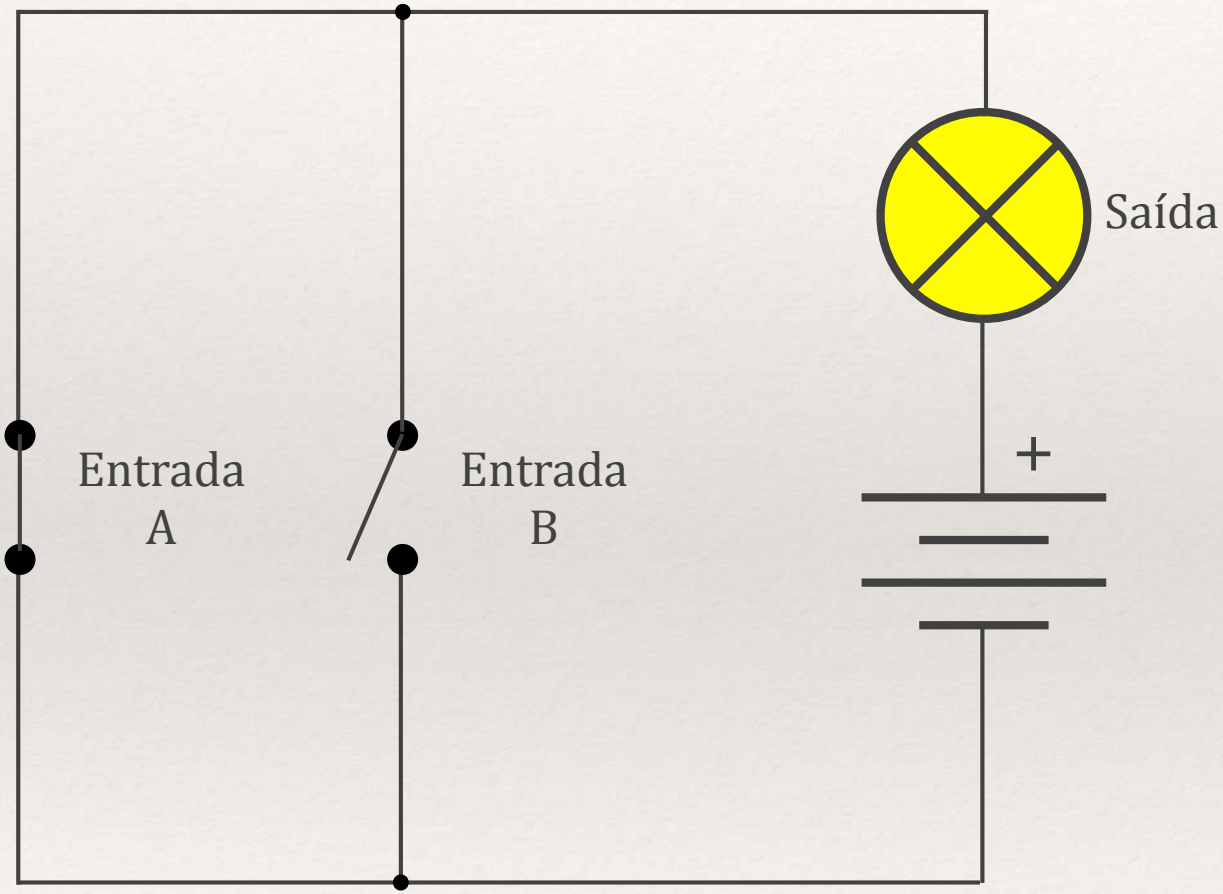
Blocos Funcionais Eletromecânicos



Lógica “OU”
(OR)

A	B	S
0	0	0
0	1	1

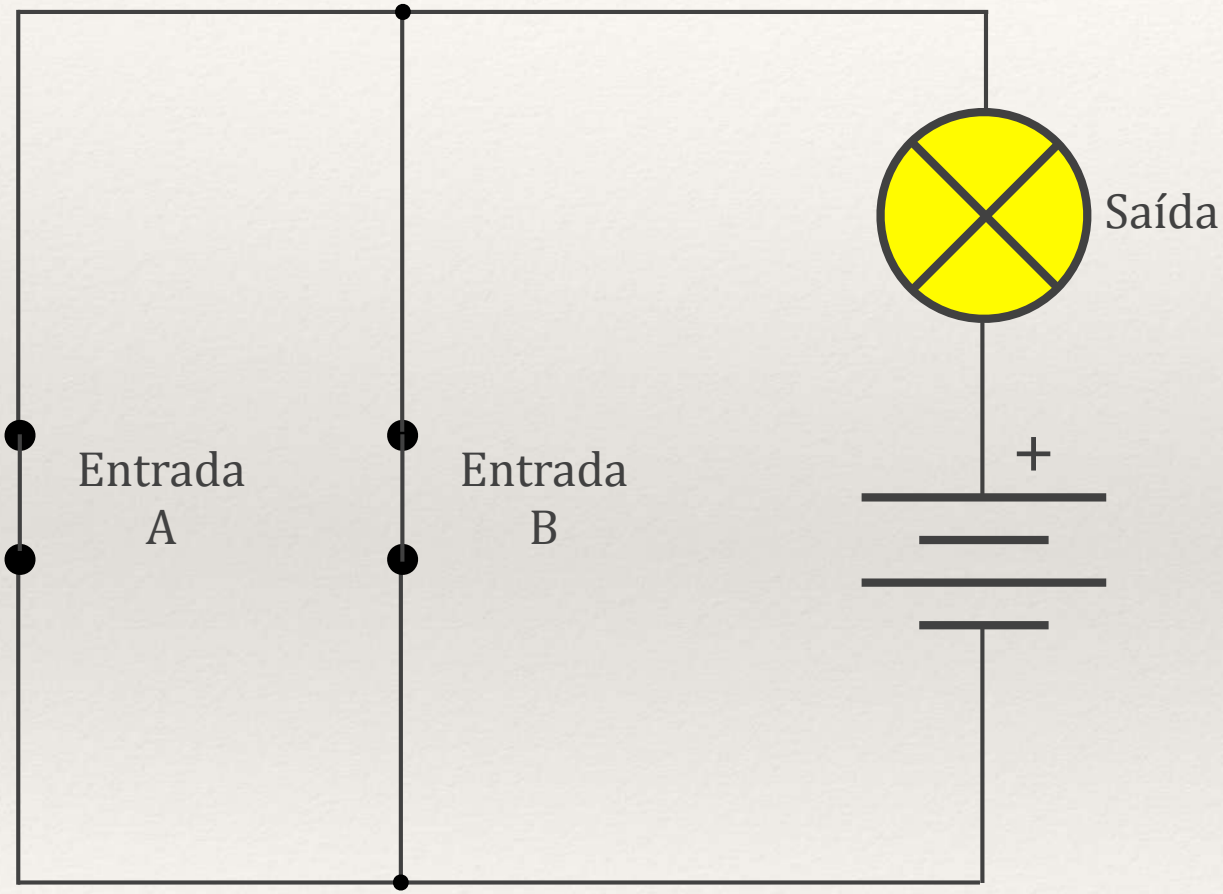
Blocos Funcionais Eletromecânicos



Lógica "OU"
(OR)

A	B	S
0	0	0
0	1	1
1	0	1

Blocos Funcionais Eletromecânicos



Lógica "OU"
(OR)

A	B	S
0	0	0
0	1	1
1	0	1
1	1	1

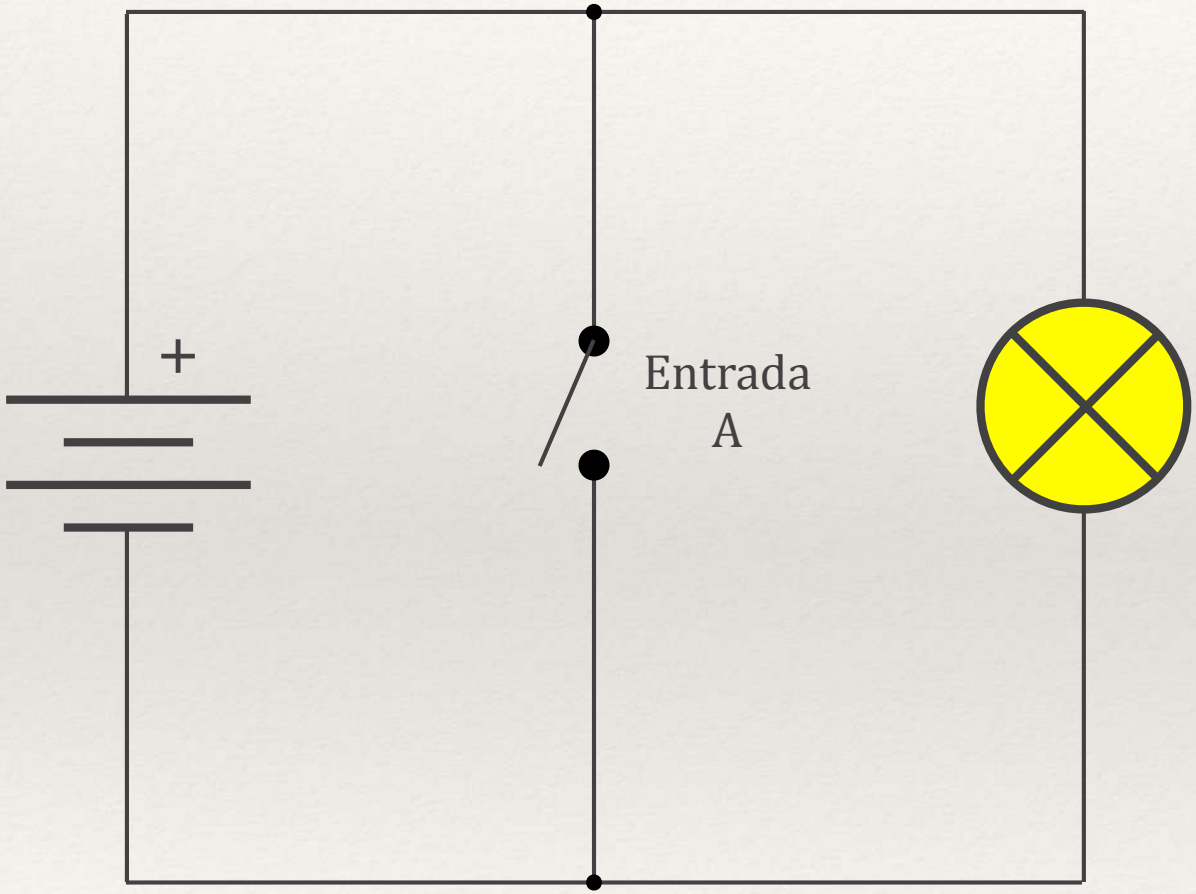
AND é o oposto de OR ?

Na verdade, não …

Precisamos de um operador de “negação”;

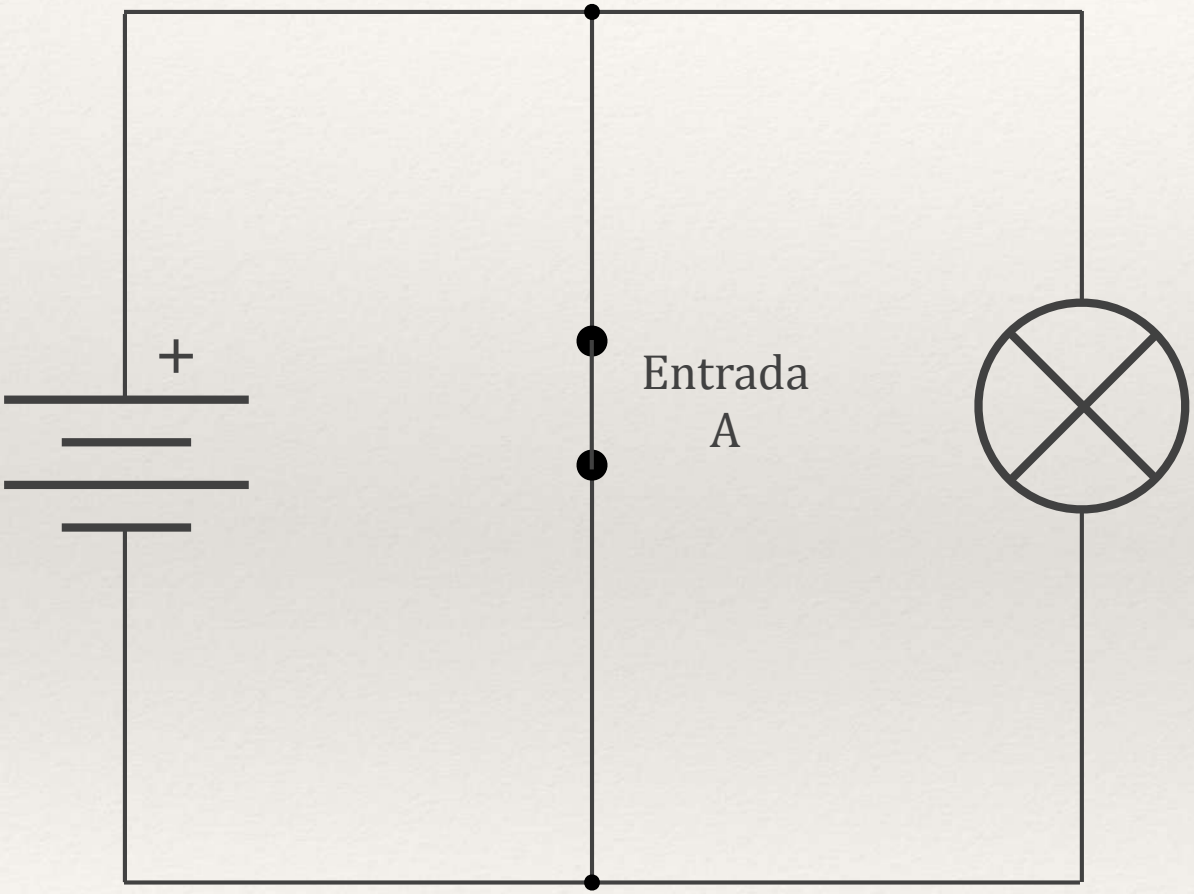
Como implementar?

Bloco NÃO Funcional



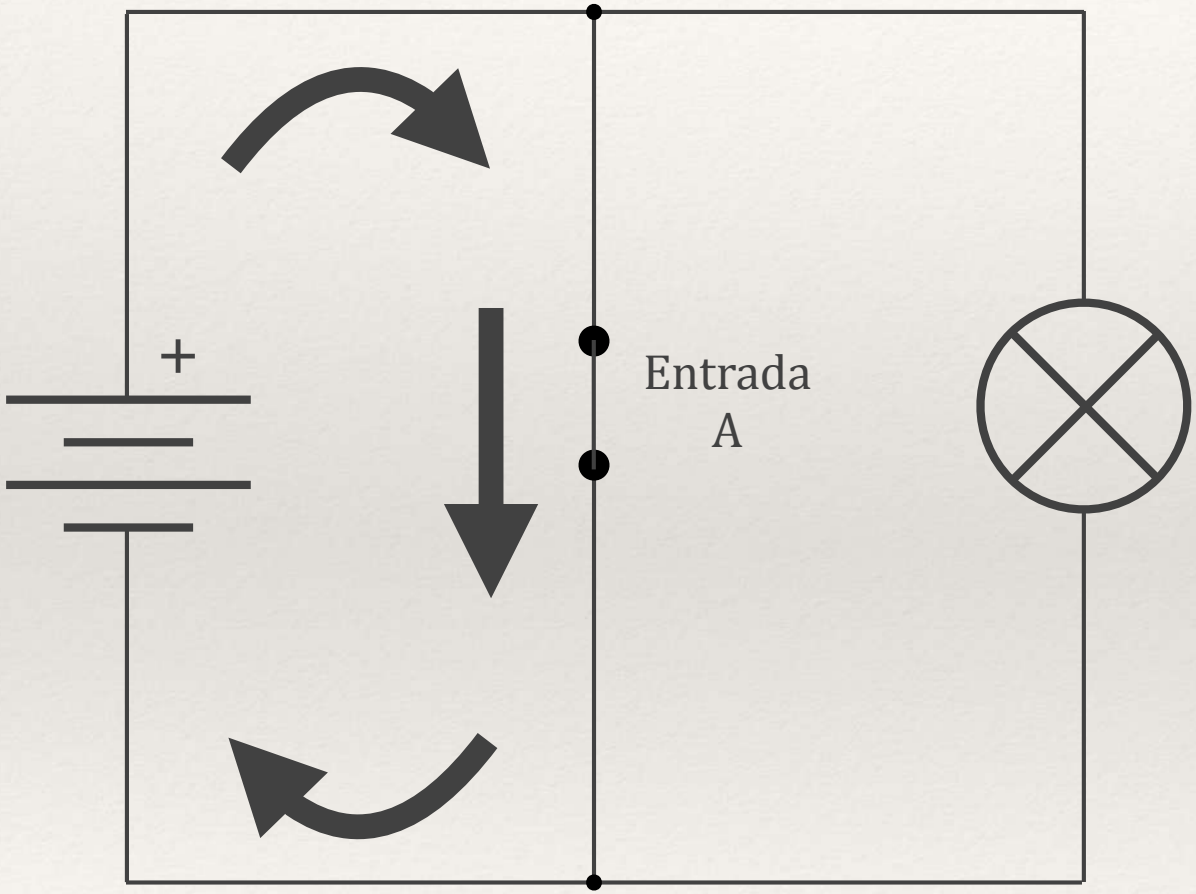
A	S
0	1

Bloco NÃO Funcional



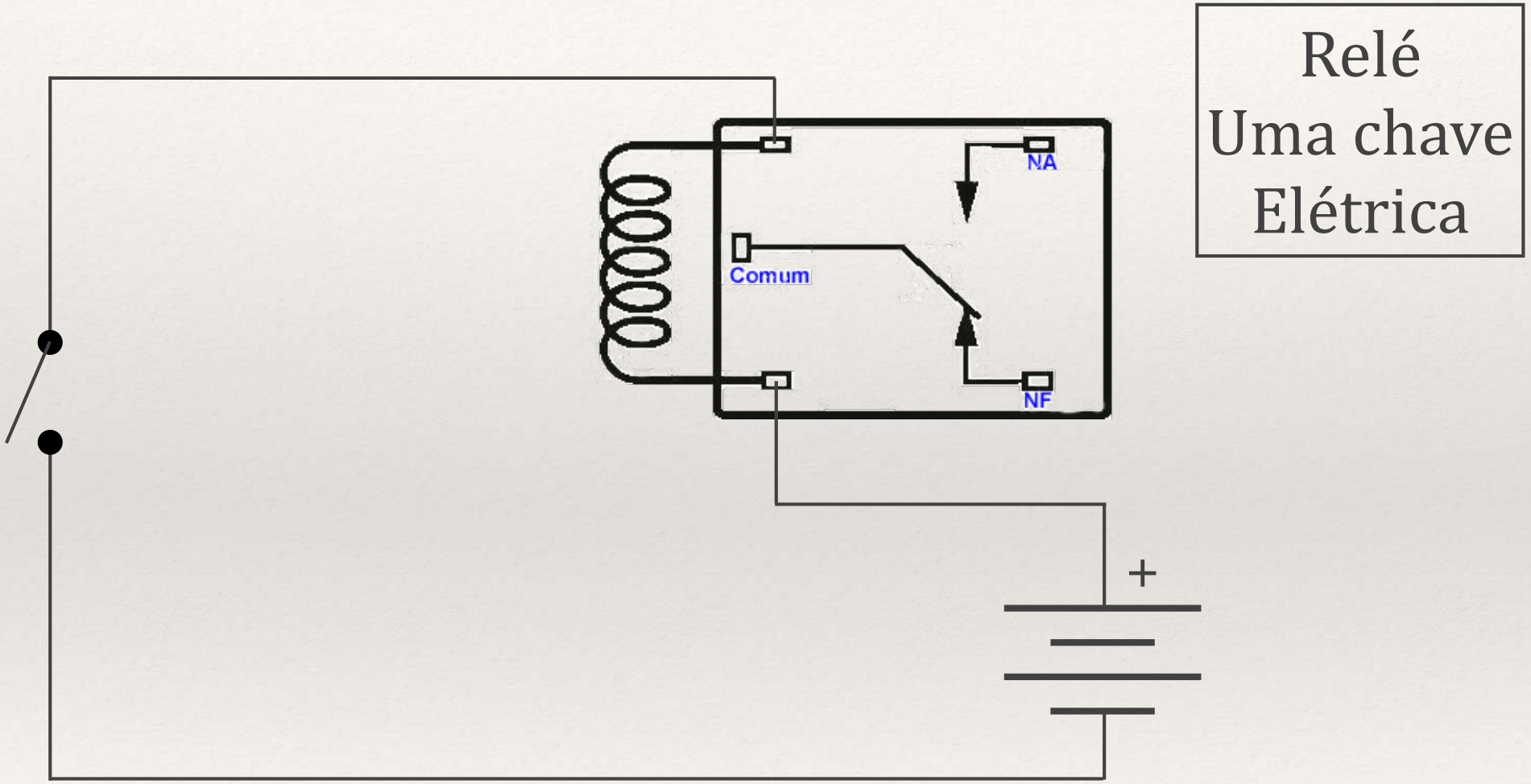
A	S
0	1
1	0

Bloco NÃO Funcional

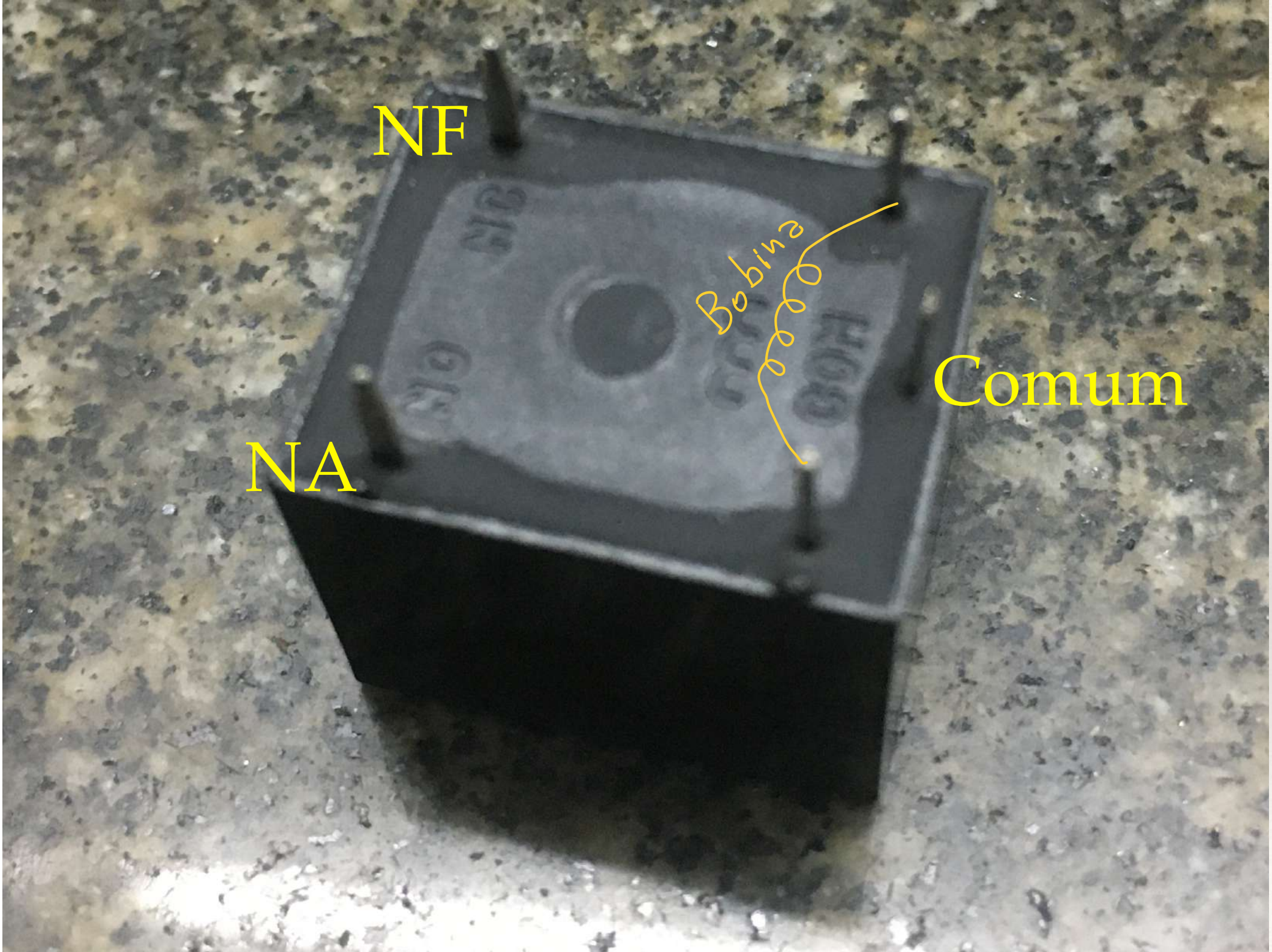


A	S
0	1
1	0

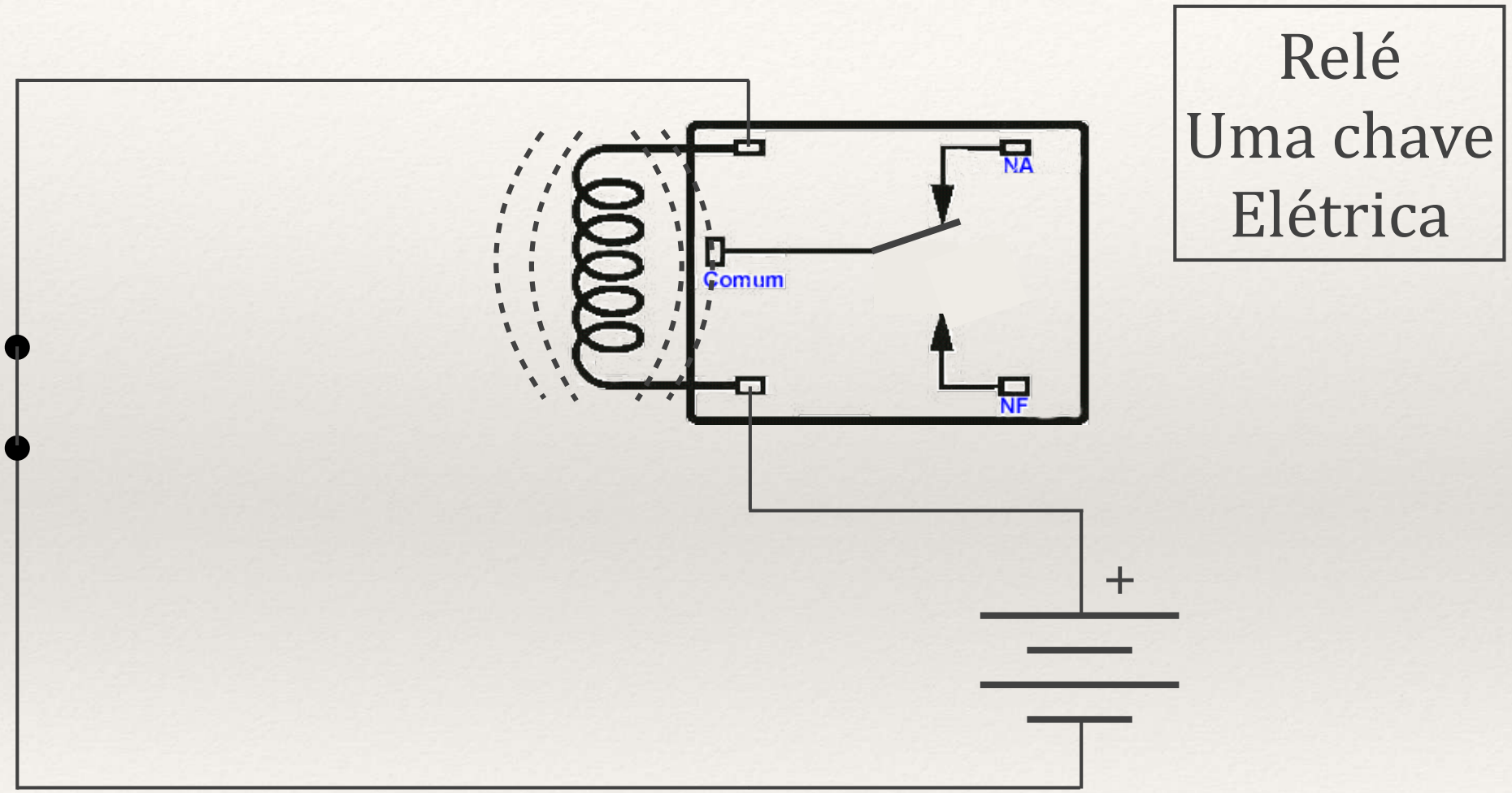
Blocos Funcionais Eletromecânicos



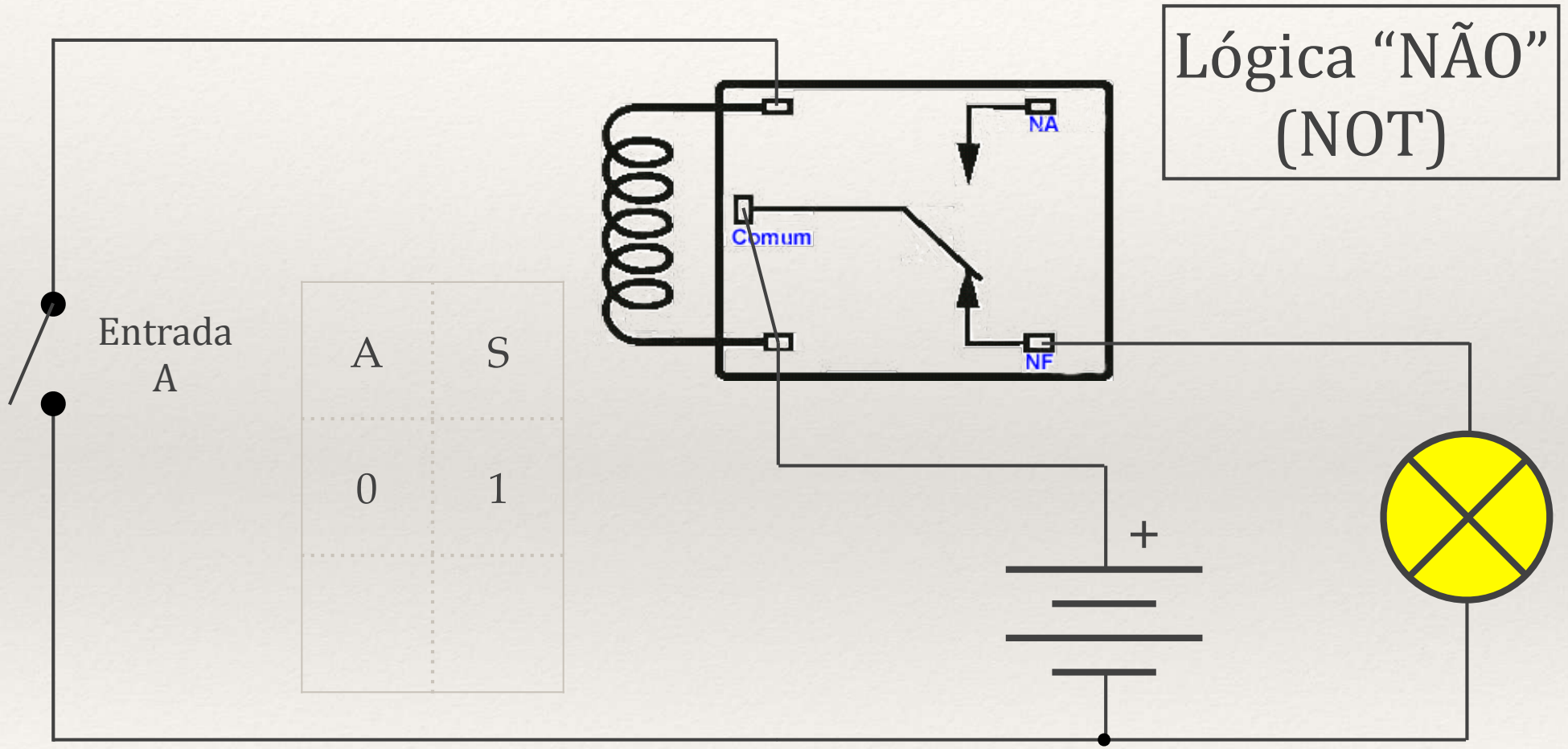

HRS4H-S-DC12V
NO:10A 250V~
NC:6A 250V~
10A 120VAC
10A 24VDC
R C G


NF
Bobina
Comum
NA

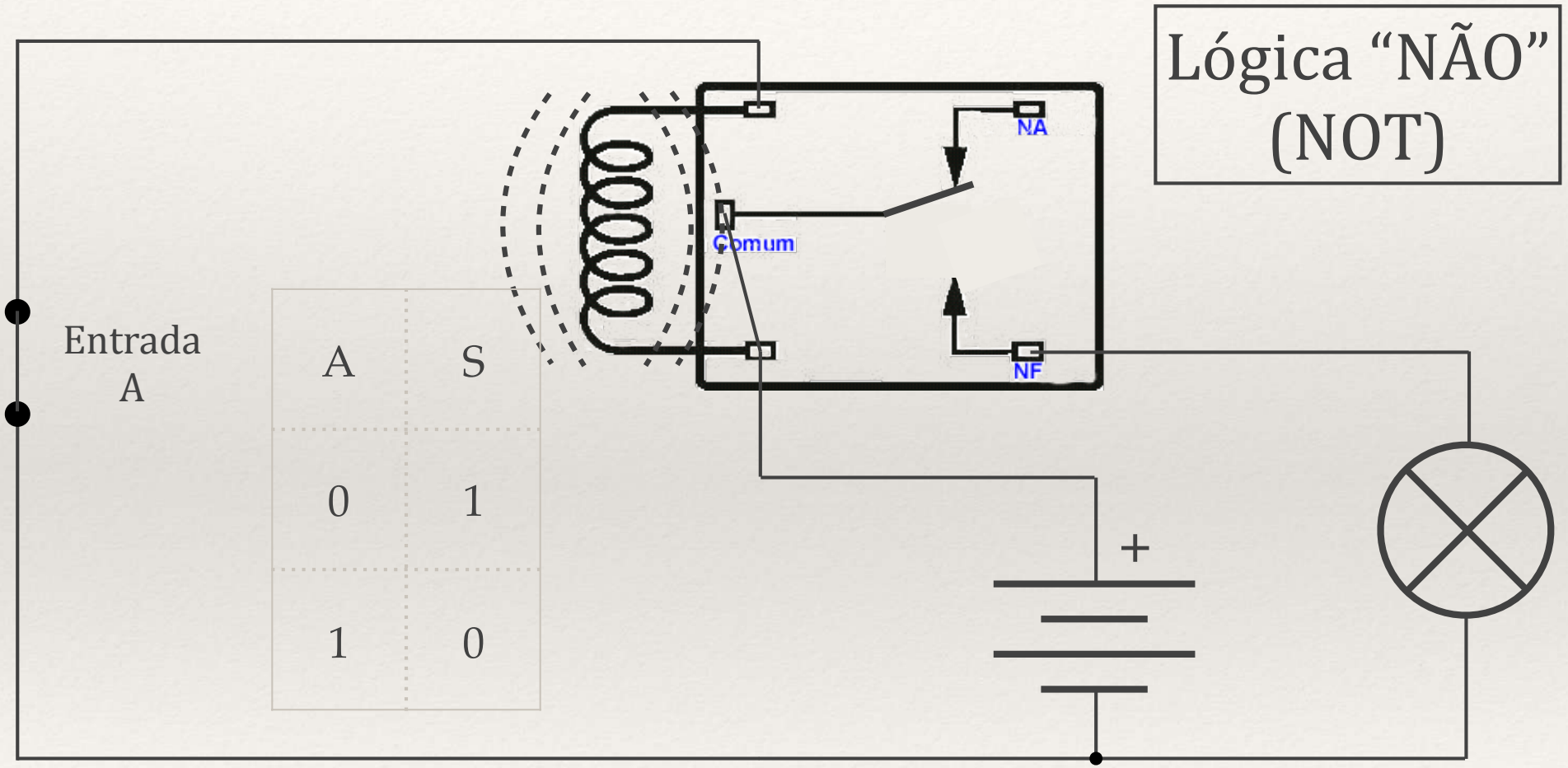
Blocos Funcionais Eletromecânicos



Blocos Funcionais Eletromecânicos



Blocos Funcionais Eletromecânicos



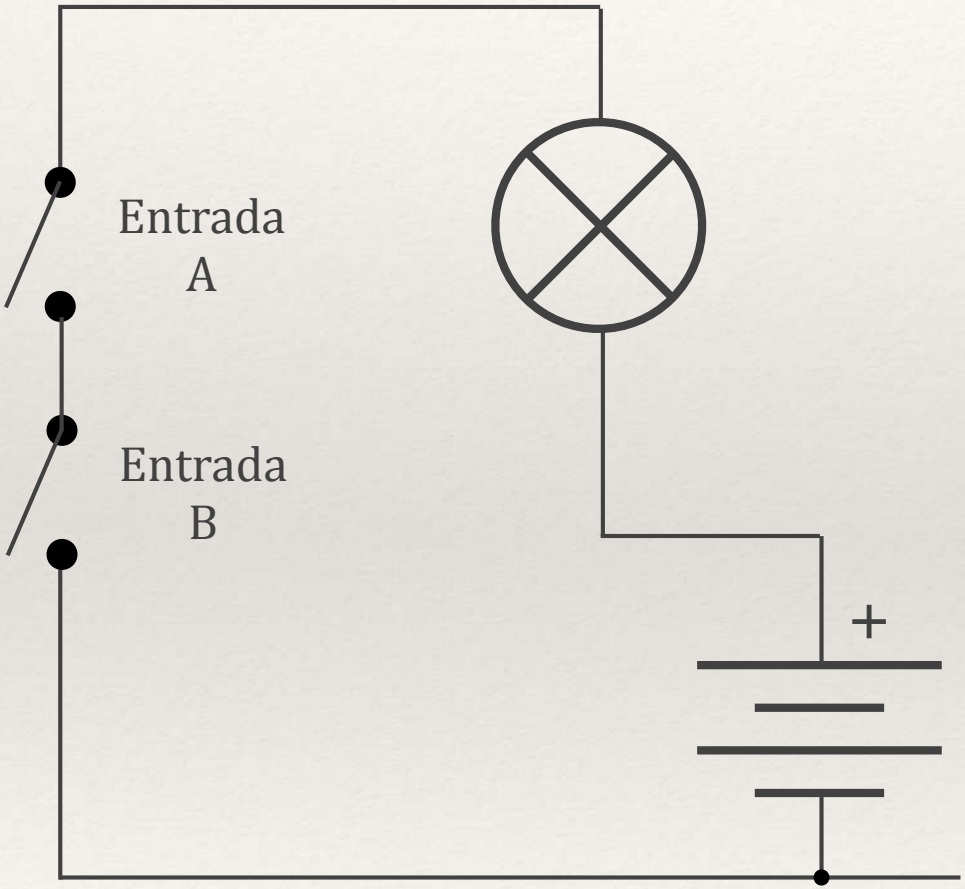
Outra função do Relé

Permite concatenar circuitos, ligando a saída de um na entrada do outro;

Isso permite criar circuitos mais complexos e funcionais;

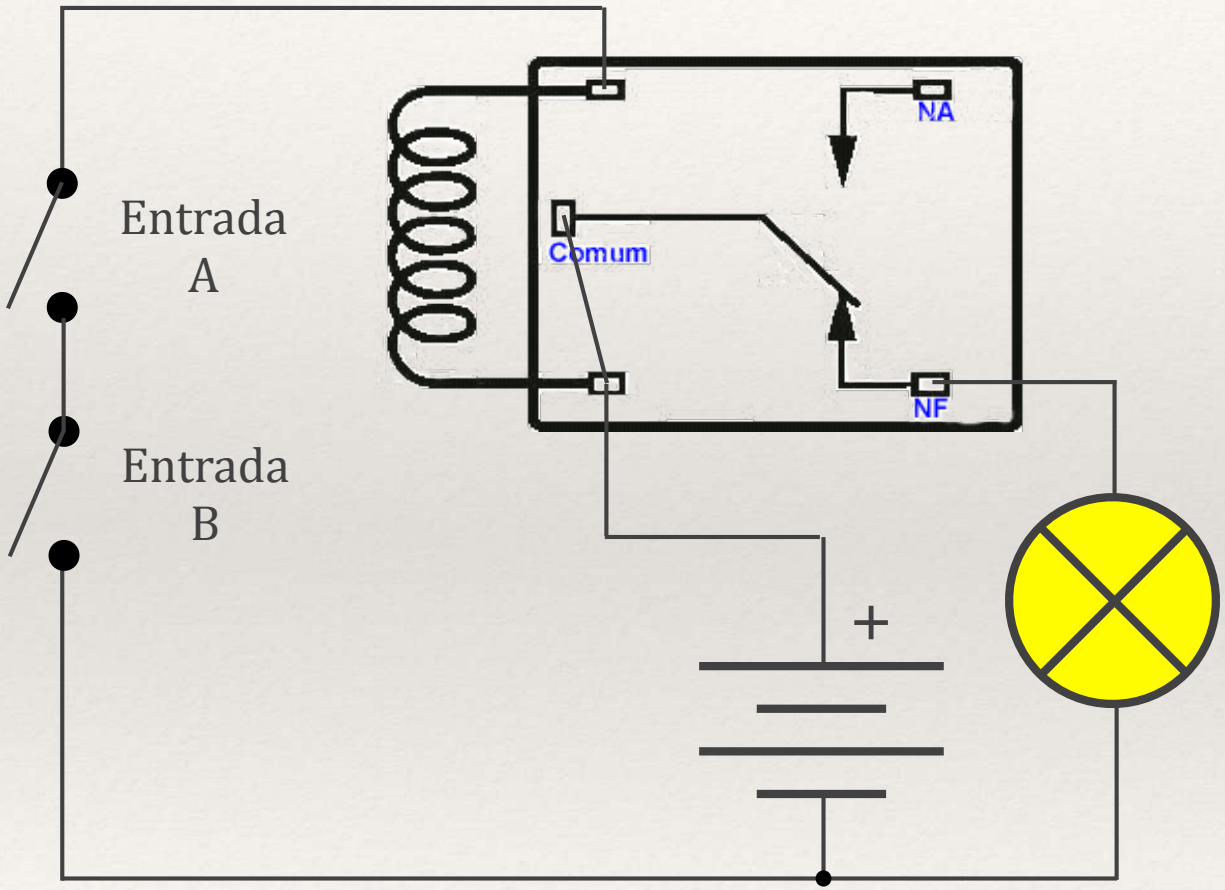
Por exemplo, para criar um circuito “oposto" do AND, precisamos conectar a saída de um AND na entrada de um NOT, criando o (NOT) (AND), ou simplesmente NAND.

Blocos Funcionais Eletromecânicos



Transformando uma Lógica “E” (AND) em “Não E” (NAND) usando um Relé...

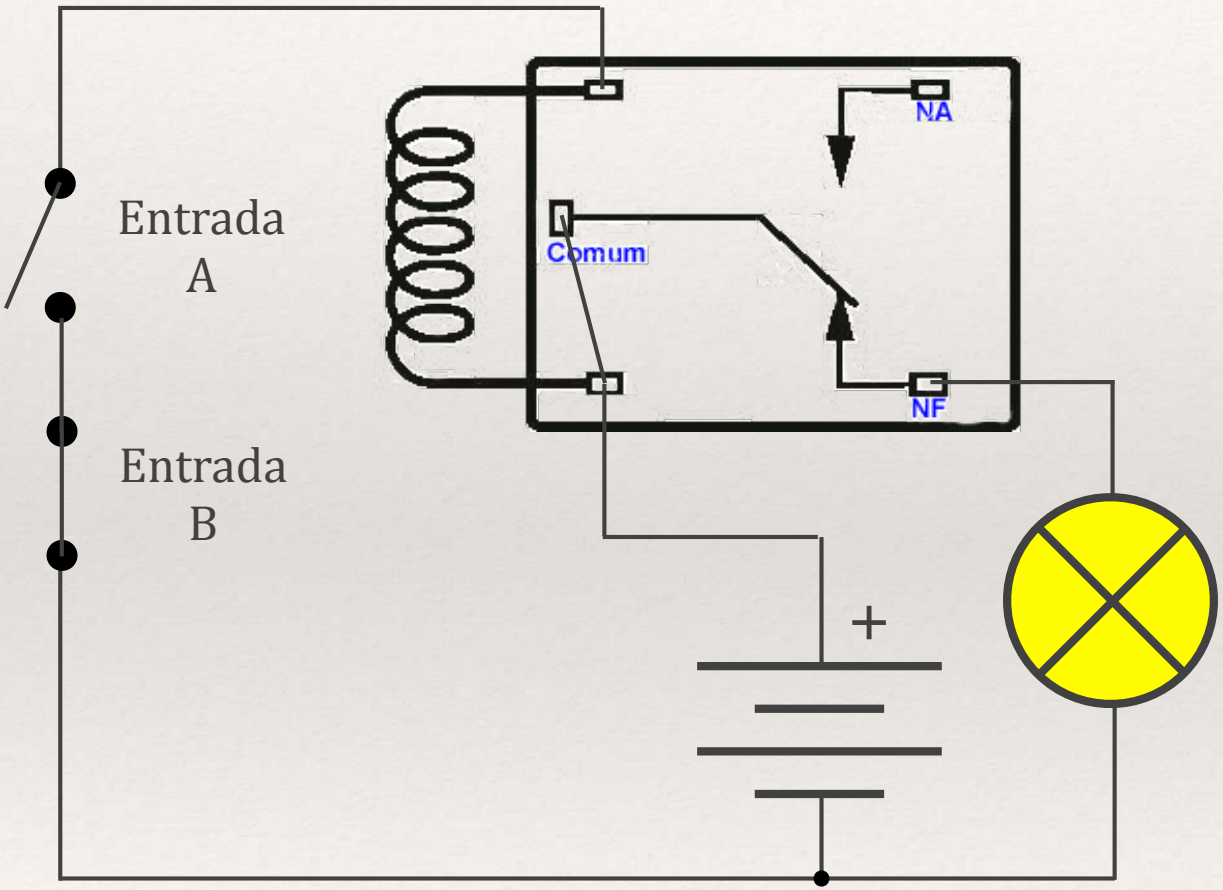
Blocos Funcionais Eletromecânicos



Lógica “Não E”
(NAND)

A	B	S
0	0	1

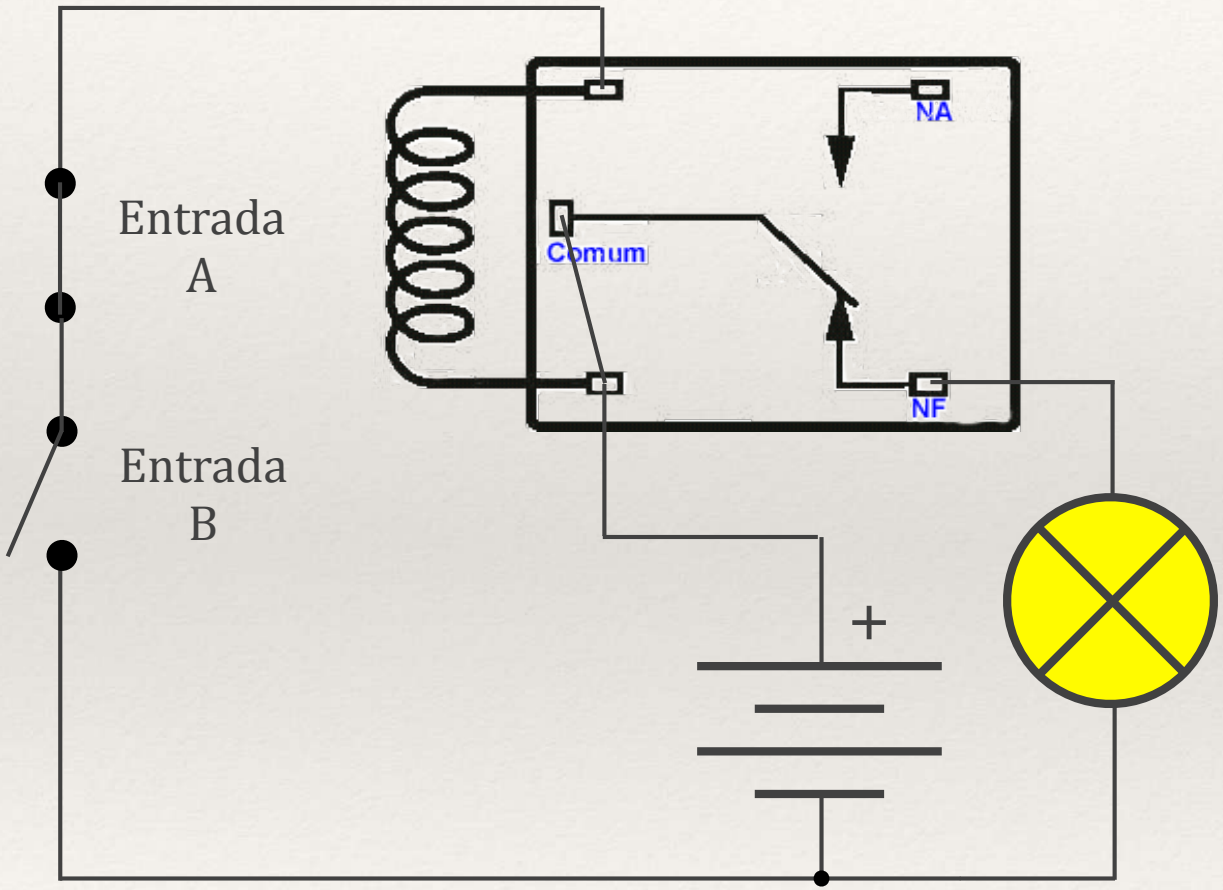
Blocos Funcionais Eletromecânicos



Lógica “Não E”
(NAND)

A	B	S
0	0	1
0	1	1

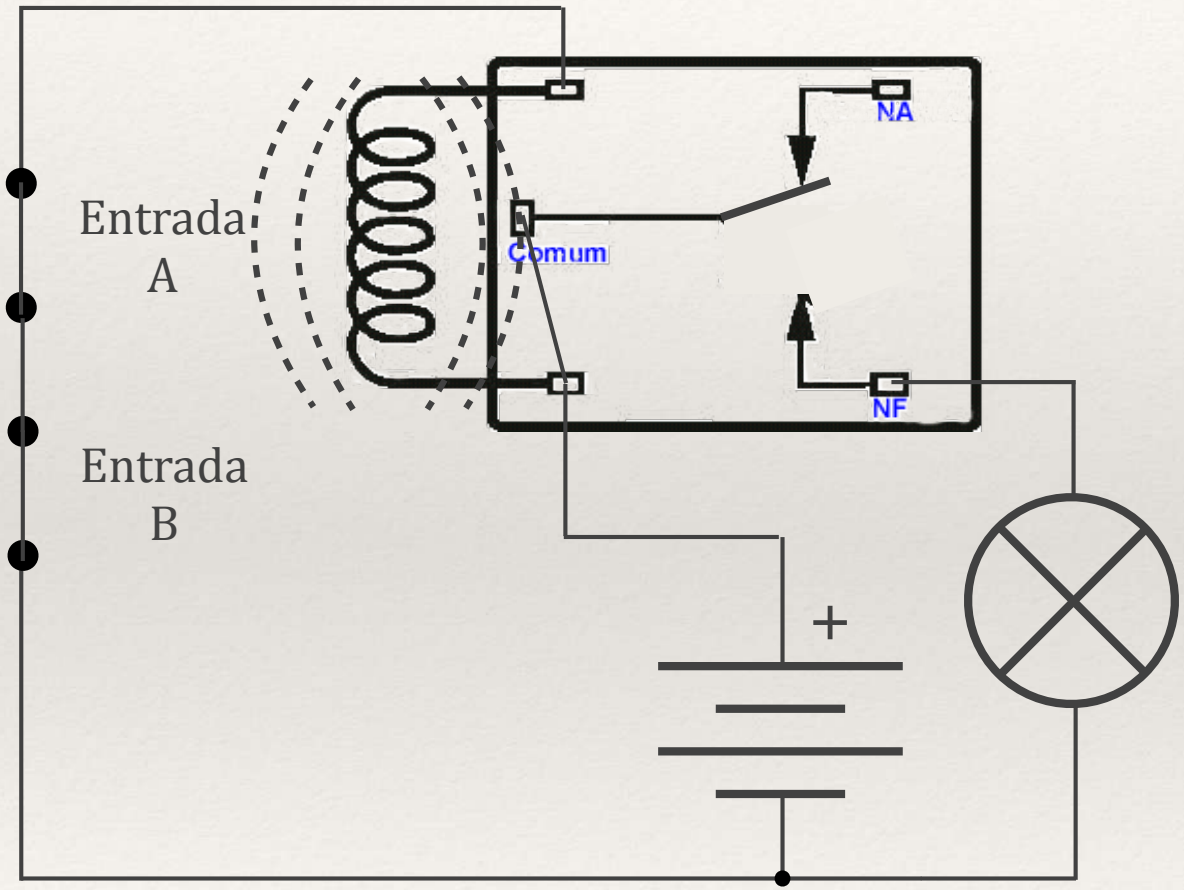
Blocos Funcionais Eletromecânicos



Lógica “Não E” (NAND)

A	B	S
0	0	1
0	1	1
1	0	1

Blocos Funcionais Eletromecânicos



Lógica “Não E” (NAND)

A	B	S
0	0	1
0	1	1
1	0	1
1	1	0

Computador com Relés

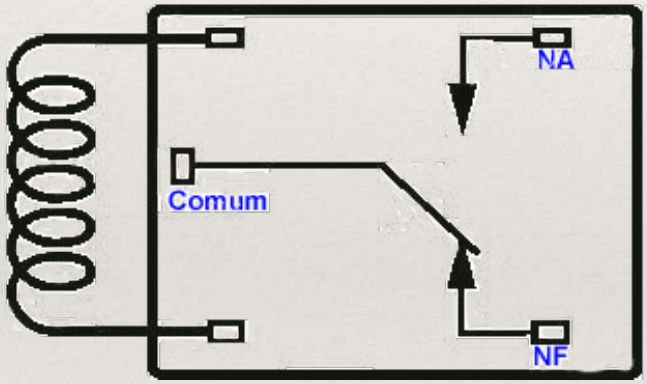
Concatenando milhares de circuitos, é possível construir Unidades Lógicas e Aritméticas, Memória, e todos os blocos componentes de um computador;

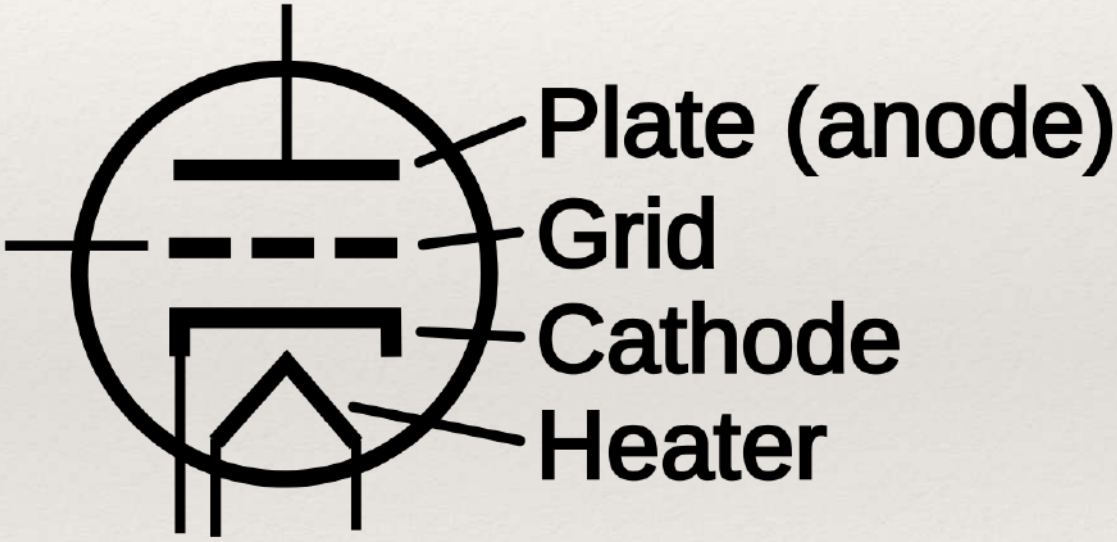
No entanto, isso vai ficar MUITO grande (um processador Intel i3 possui 382 MILHÕES de “relés”);

Outro problema é a performance e consumo de energia elétrica;

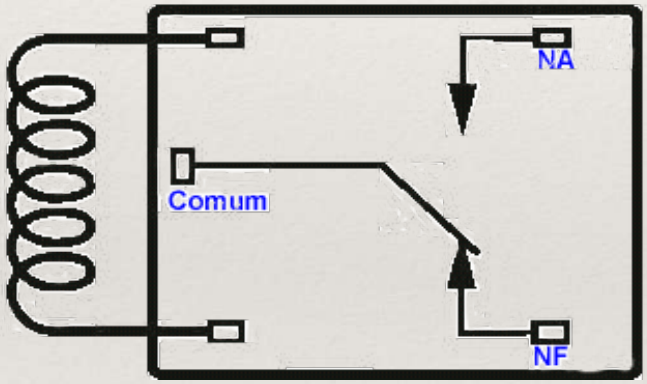
Como resolver?

Relé X Válvula a Vácuo





Relé X Válvula a Vácuo

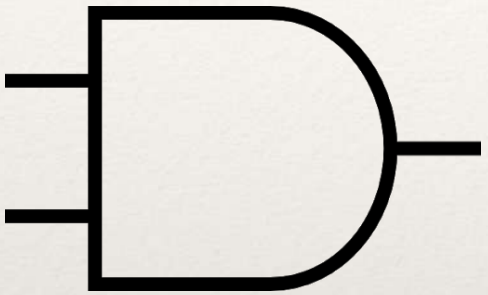


5,6 cm

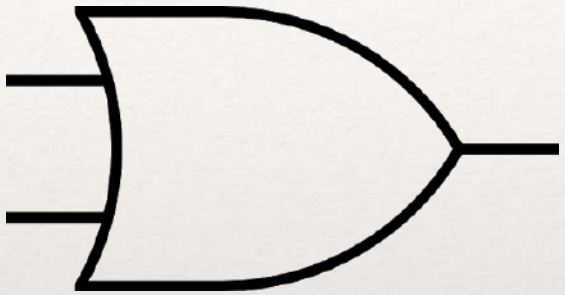
Válvula a Vácuo x Transistor

382 milhões de Transistores

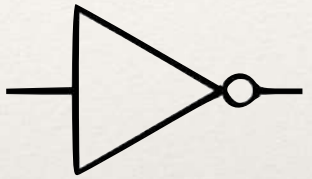
Portas Lógicas



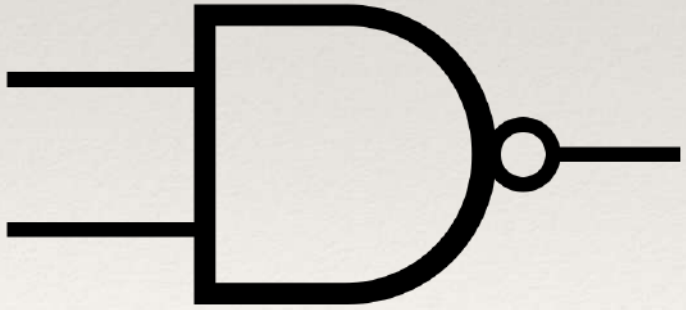
AND



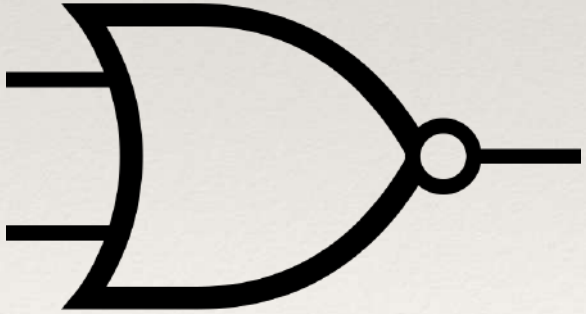
OR



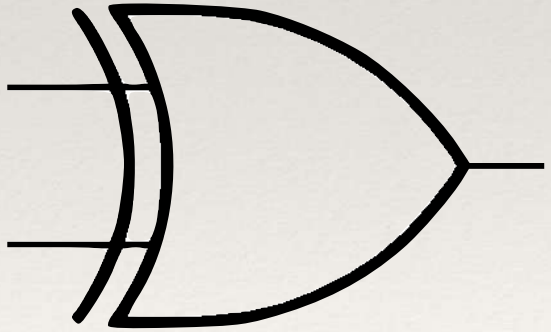
NOT



NAND

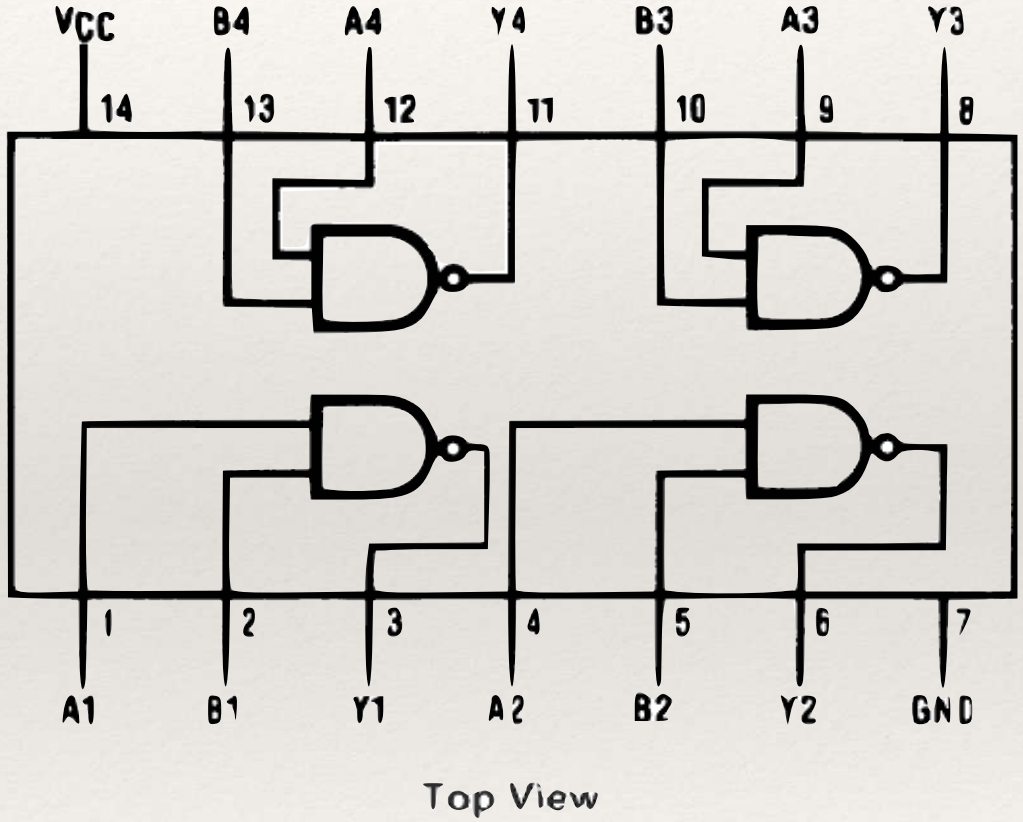


NOR



XOR

Circuito Integrado 74HC00



Top View

Circuitos Aritméticos

Um computador, tal como vimos, é construído com a concatenação de diversos blocos básicos;

Uma das partes importantes em um computador é a ULA (Unidade Lógica e Aritmética), que é capaz de realizar, entre outras coisas, cálculos aritméticos;

Como implementar um circuito "somador" em binário?

$$\begin{array}{r} {\scriptstyle 1\quad\ 1\ 1\ 1} \\ 1\,0\,0\,1\,0\,1\,1\,1 \\ 0\,1\,0\,1\,0\,1\,0\,1 \\ \hline 1\,1\,1\,0\,1\,1\,0\,0 \end{array} +$$

A	B	S
0	0	0
0	1	1
1	0	1
1	1	0

1
(vai um)

Circuitos Aritméticos

Um computador, tal como vimos, é construído com a concatenação de diversos blocos básicos;

Uma das partes importantes em um computador é a ULA (Unidade Lógica e Aritmética), que é capaz de realizar, entre outras coisas, cálculos aritméticos;

Como implementar um circuito "somador" em binário?

$$\begin{array}{r} {\scriptstyle 1\quad\ 1\ 1\ 1} \\ 1\,0\,0\,1\,0\,1\,1\,1 \\ 0\,1\,0\,1\,0\,1\,0\,1 \\ \hline 1\,1\,1\,0\,1\,1\,0\,0 \end{array} +$$

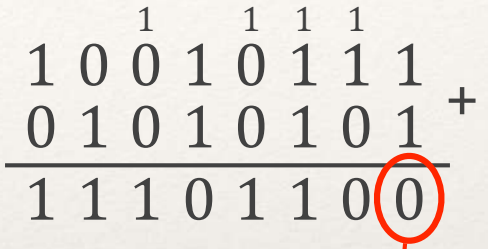
A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Circuitos Aritméticos

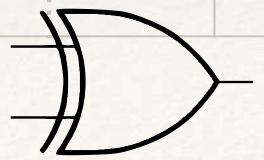
Um computador, tal como vimos, é construído com a concatenação de diversos blocos básicos;

Uma das partes importantes em um computador é a ULA (Unidade Lógica e Aritmética), que é capaz de realizar, entre outras coisas, cálculos aritméticos;

Como implementar um circuito "somador" em binário?



A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

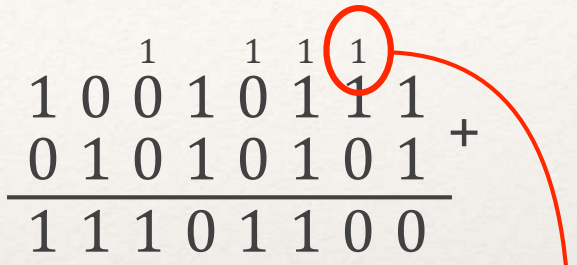


Circuitos Aritméticos

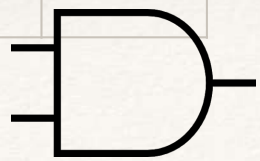
Um computador, tal como vimos, é construído com a concatenação de diversos blocos básicos;

Uma das partes importantes em um computador é a ULA (Unidade Lógica e Aritmética), que é capaz de realizar, entre outras coisas, cálculos aritméticos;

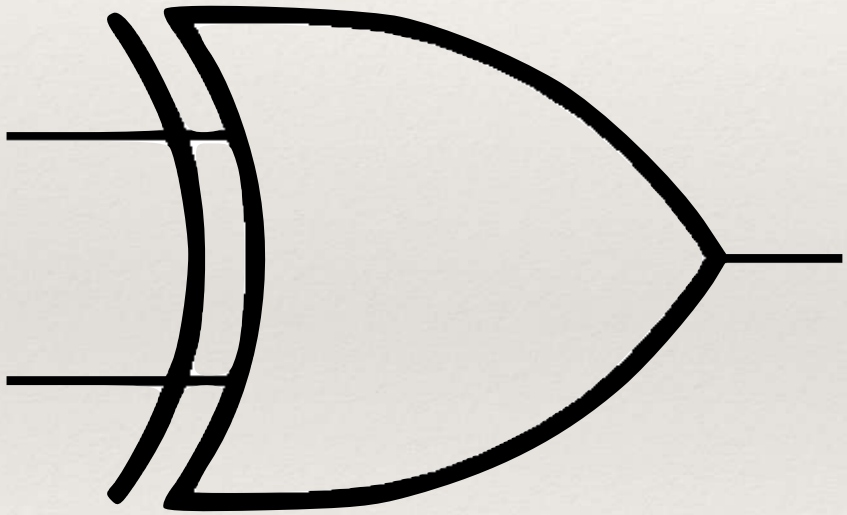
Como implementar um circuito "somador" em binário?



A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



Circuitos Aritméticos

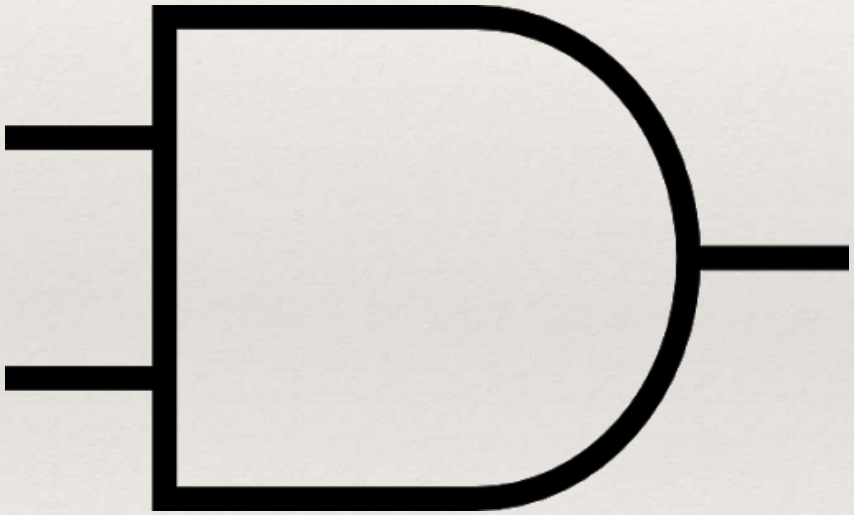


$$\begin{array}{r} {\scriptstyle 1\quad\ 1\ 1\ 1} \\ 1\,0\,0\,1\,0\,1\,1\,1 \\ 0\,1\,0\,1\,0\,1\,0\,1 \\ \hline 1\,1\,1\,0\,1\,1\,0\,0 \end{array} +$$

A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

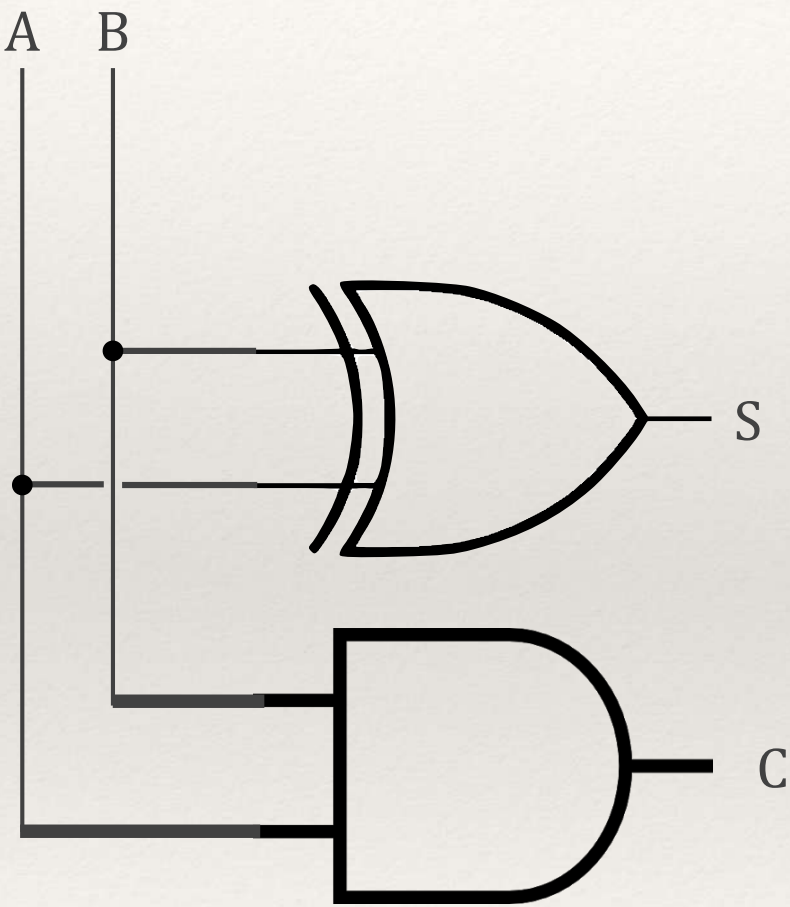
Circuitos Aritméticos

$$\begin{array}{r} {\scriptstyle 1\quad\ 1\ 1\ 1} \\ 1\,0\,0\,1\,0\,1\,1\,1 \\ 0\,1\,0\,1\,0\,1\,0\,1 \\ \hline 1\,1\,1\,0\,1\,1\,0\,0 \end{array} +$$



A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Circuitos Aritméticos

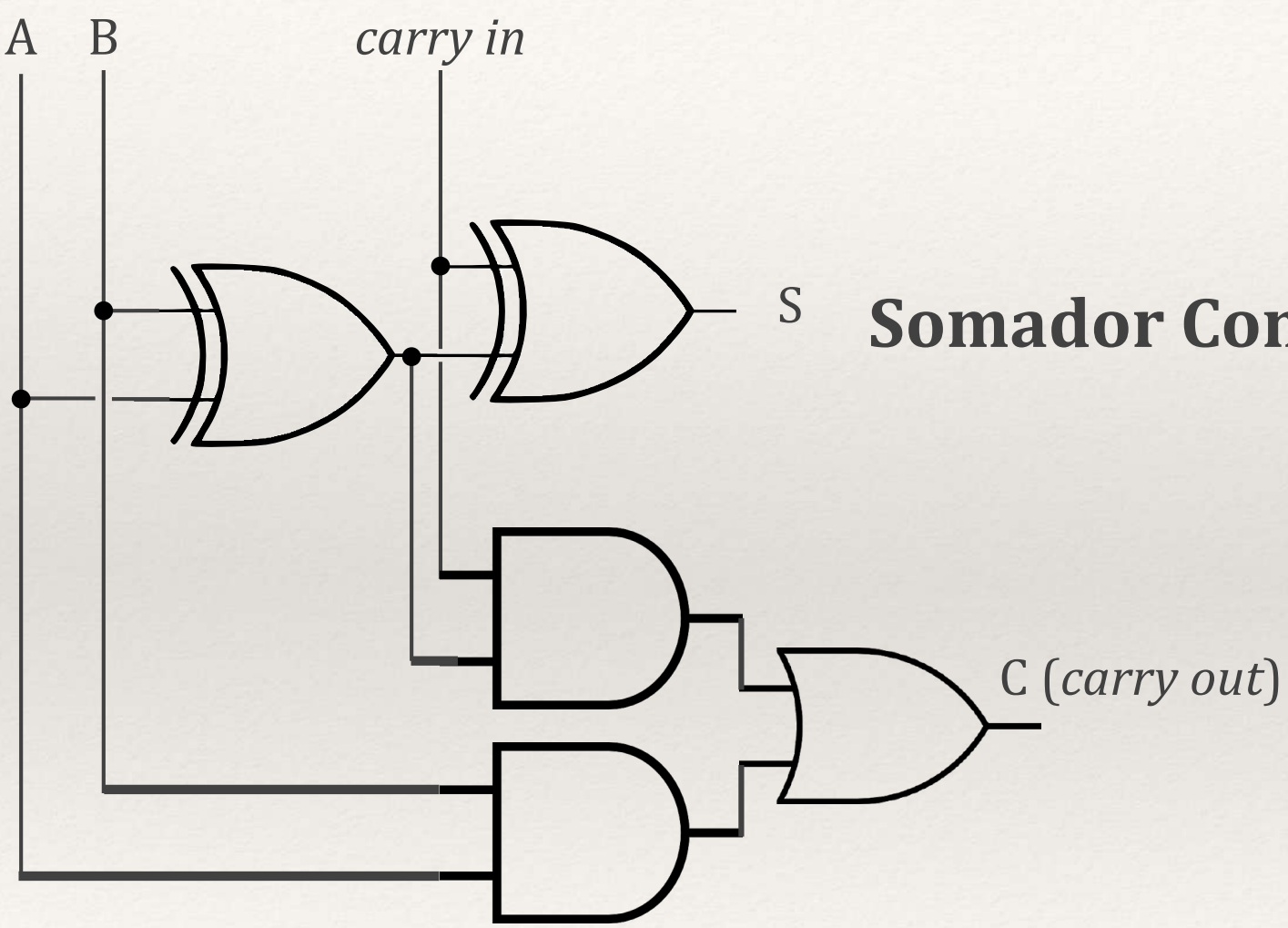



```
    1   1 1 1 ?
  1 0 0 1 0 1 1 1
  0 1 0 1 0 1 0 1 +
-----------------
  1 1 1 0 1 1 0 0
```

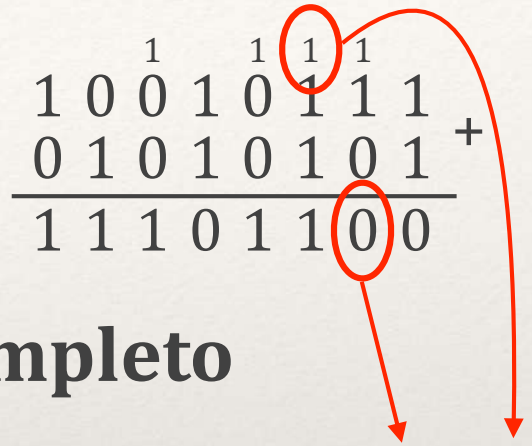
Soma apenas dois dígitos
(não recebe "Vai Um")

Meio Somador

Circuitos Aritméticos



Somador Completo

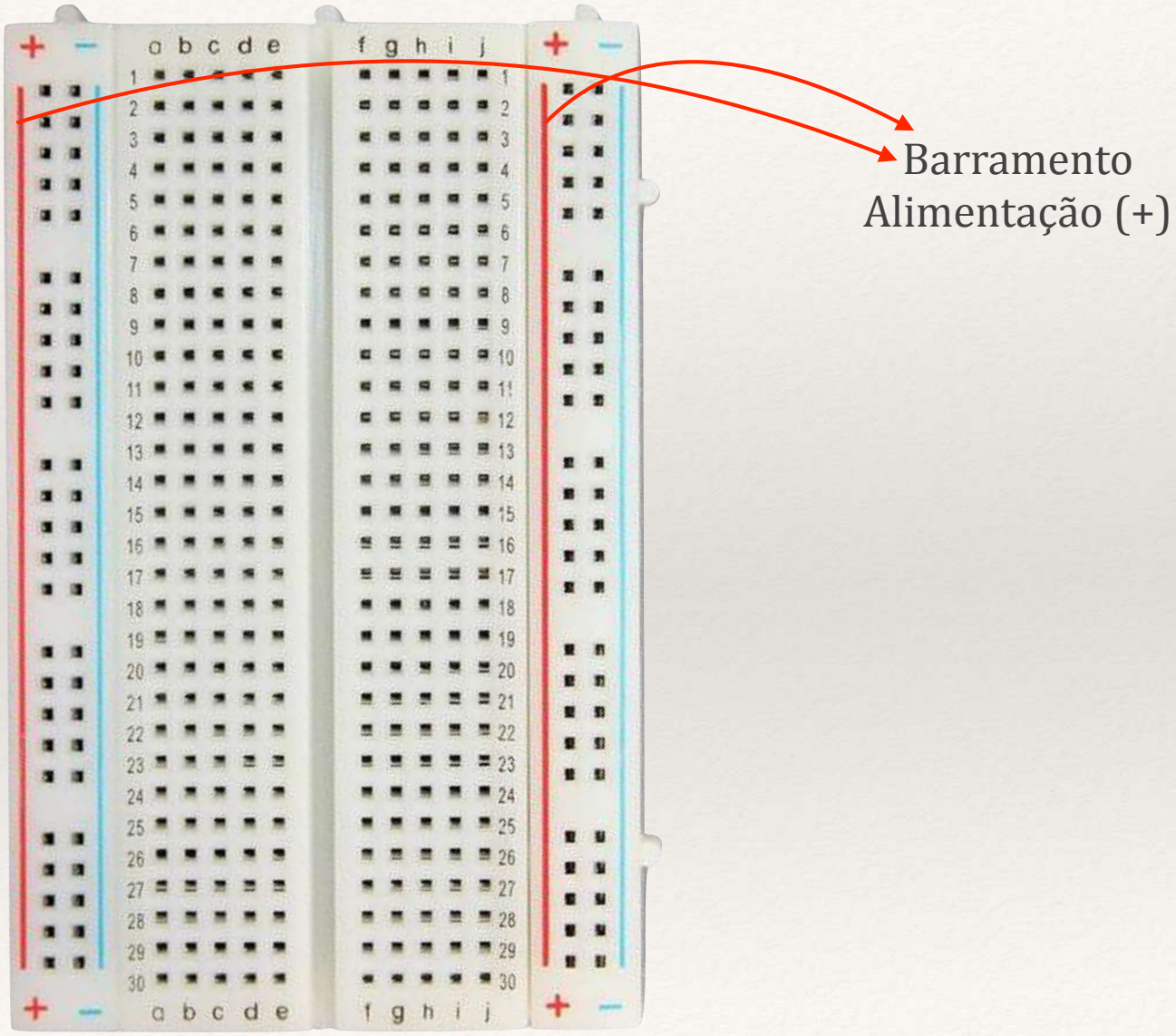


A	B	Ci	S	Co
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

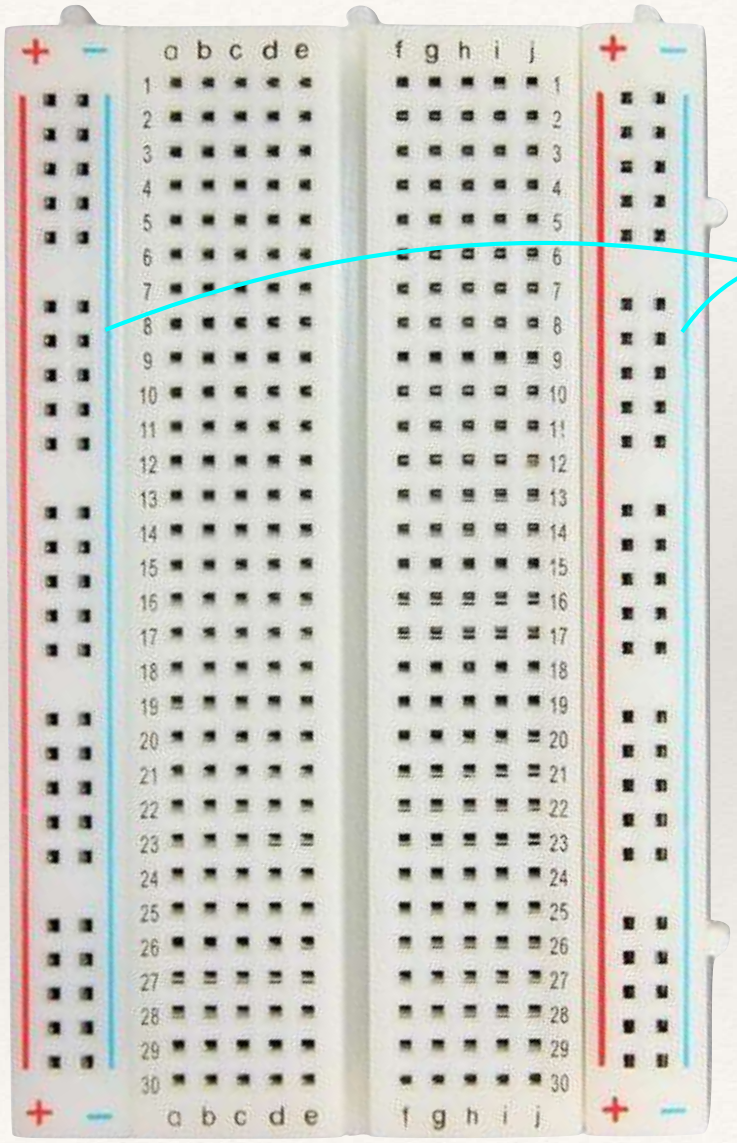
Conceitos p/Aula Prática

Protoboard: conexões e alimentação elétrica

Laboratório: Protoboard

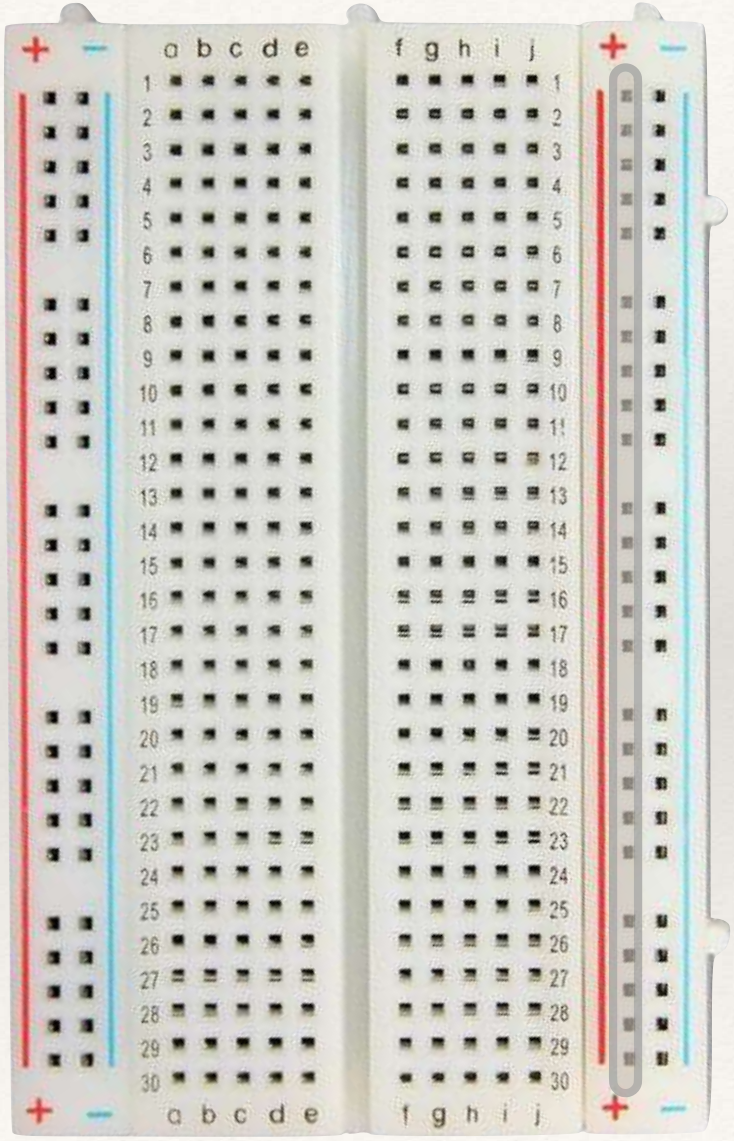


Laboratório: Protoboard



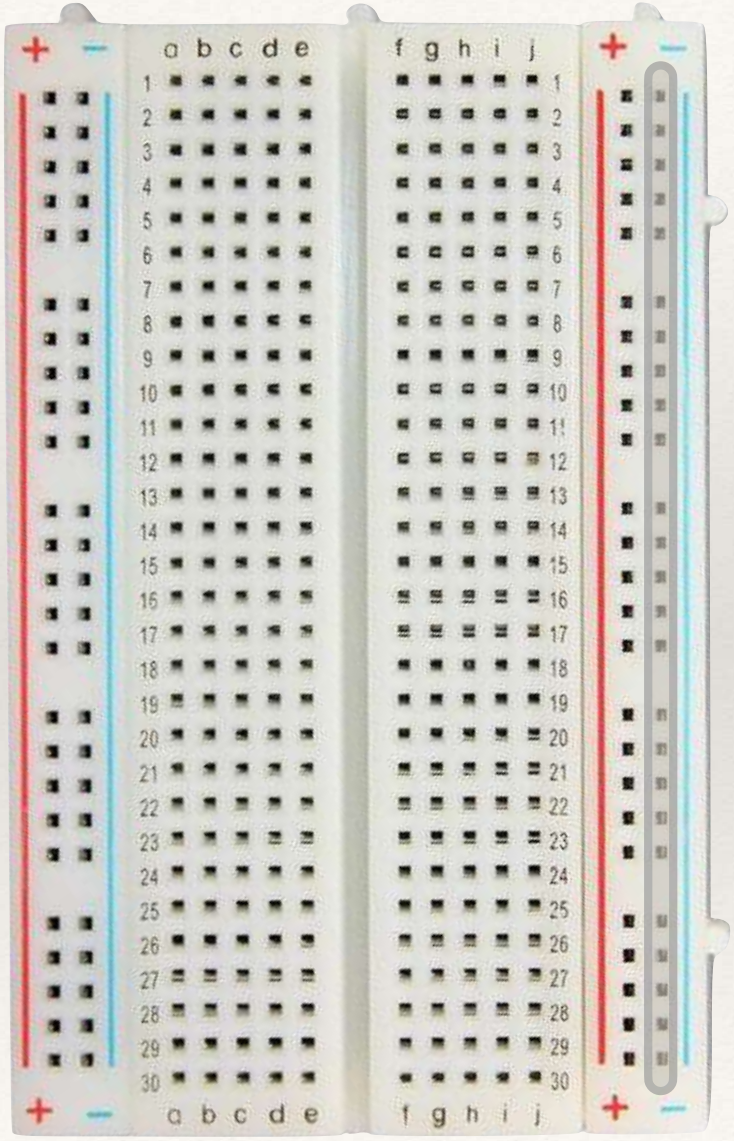
Barramento
Alimentação (-)
GND (ground), **Terra**

Laboratório: Protoboard



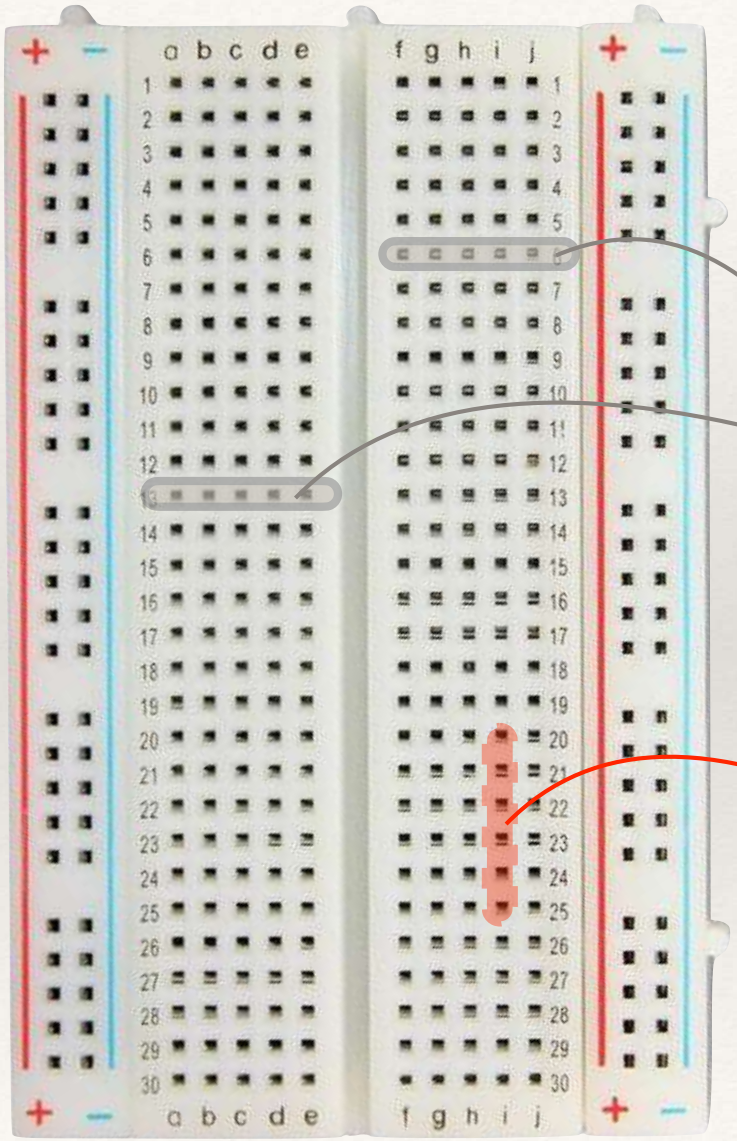
Todos os orifícios estão eletricamente conectados !

Laboratório: Protoboard



Todos os orifícios estão eletricamente conectados !

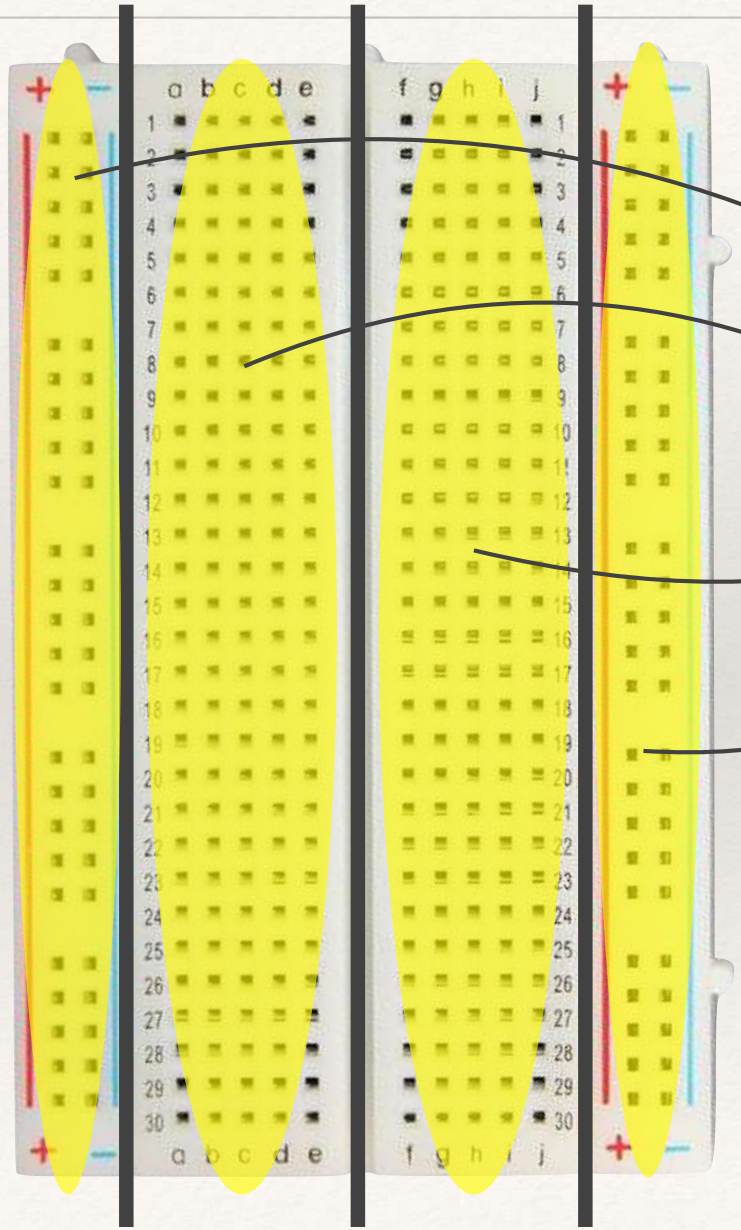
Laboratório: Protoboard



Aqui os cinco orifícios de cada linha estão eletricamente conectados !

Não há conexão vertical !

Laboratório: Protoboard



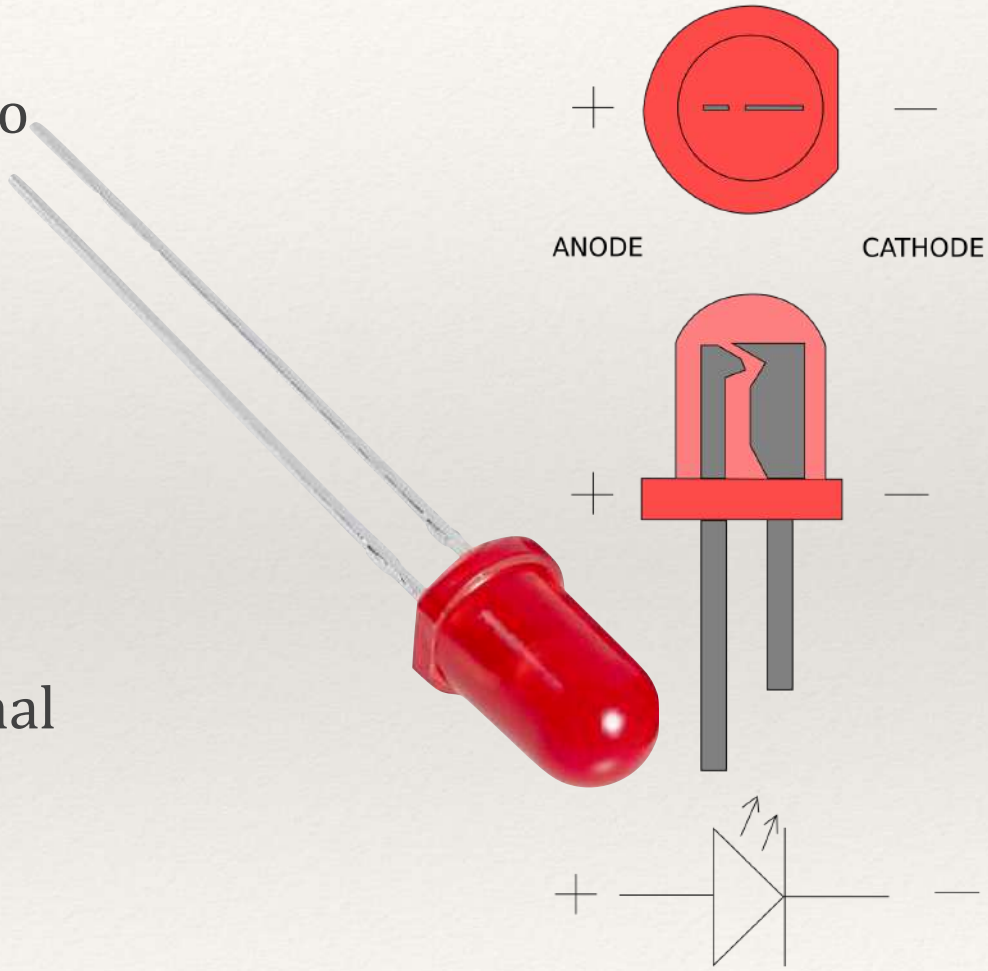
Não há conexão elétrica entre estas quatro áreas diferentes

LED - *Light-Emitting Diode*

Emite luz por eletroluminescência quando circula corrente por sua junção semicondutora;

Não limita a corrente circulante, logo exige resistor em série para não ser danificado;

Como é um DIODO, só permite a circulação de corrente em um sentido (seta do símbolo indica o sentido formal da corrente).

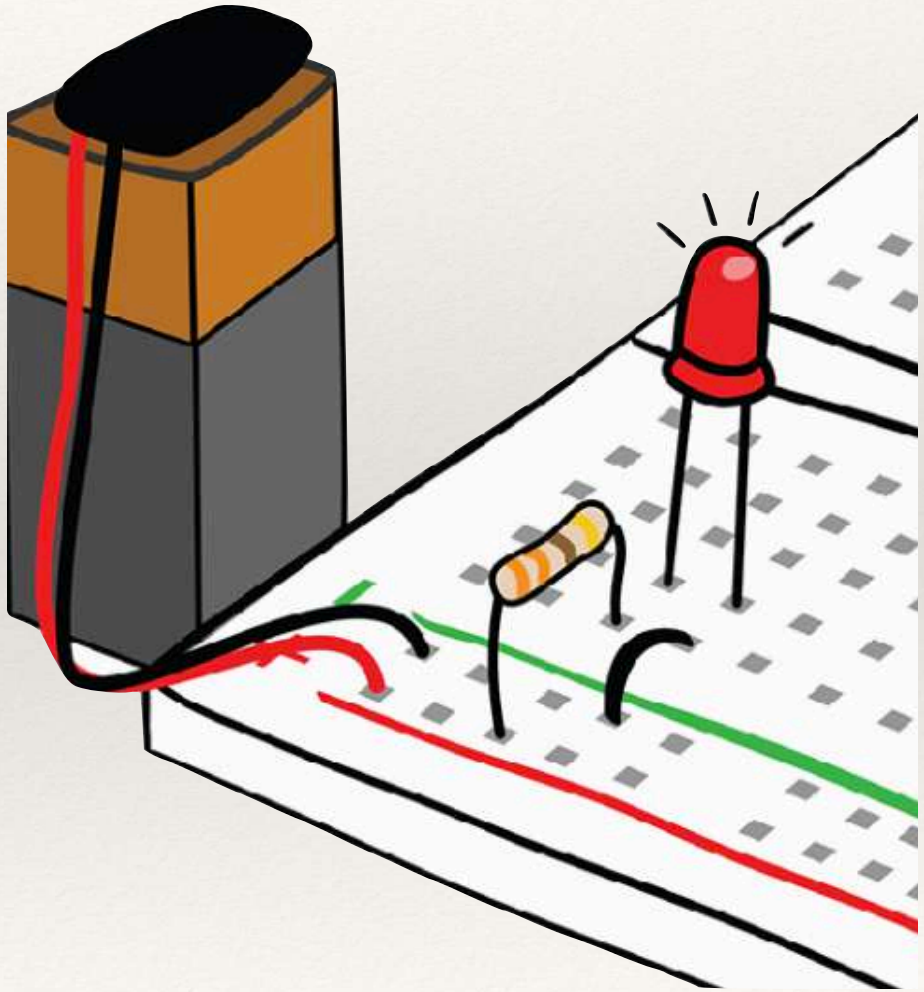


Acendendo um LED

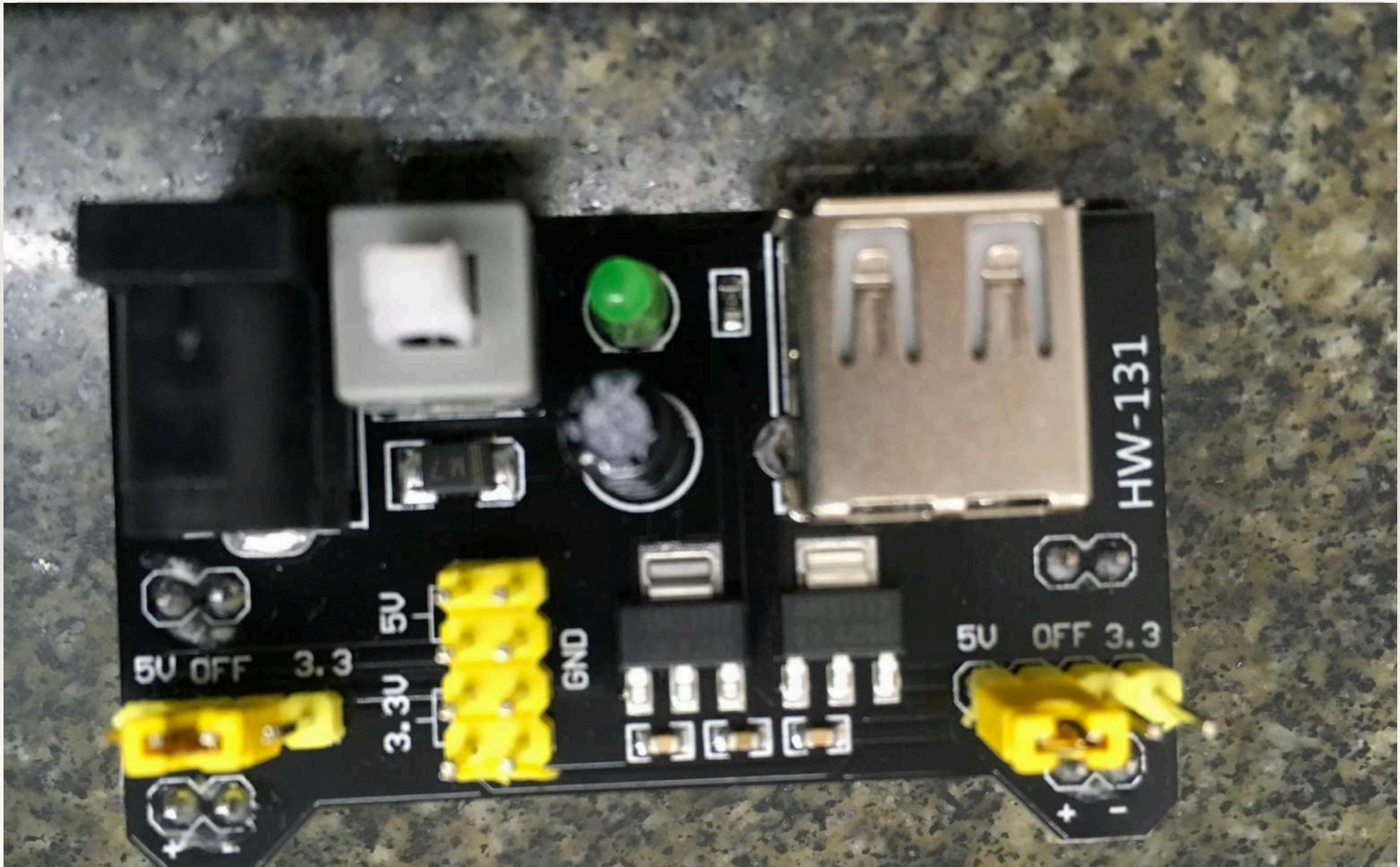
Os terminais da bateria de 9V foram conectados ao barramento de alimentação

- Vermelho = positivo
- Preto = negativo

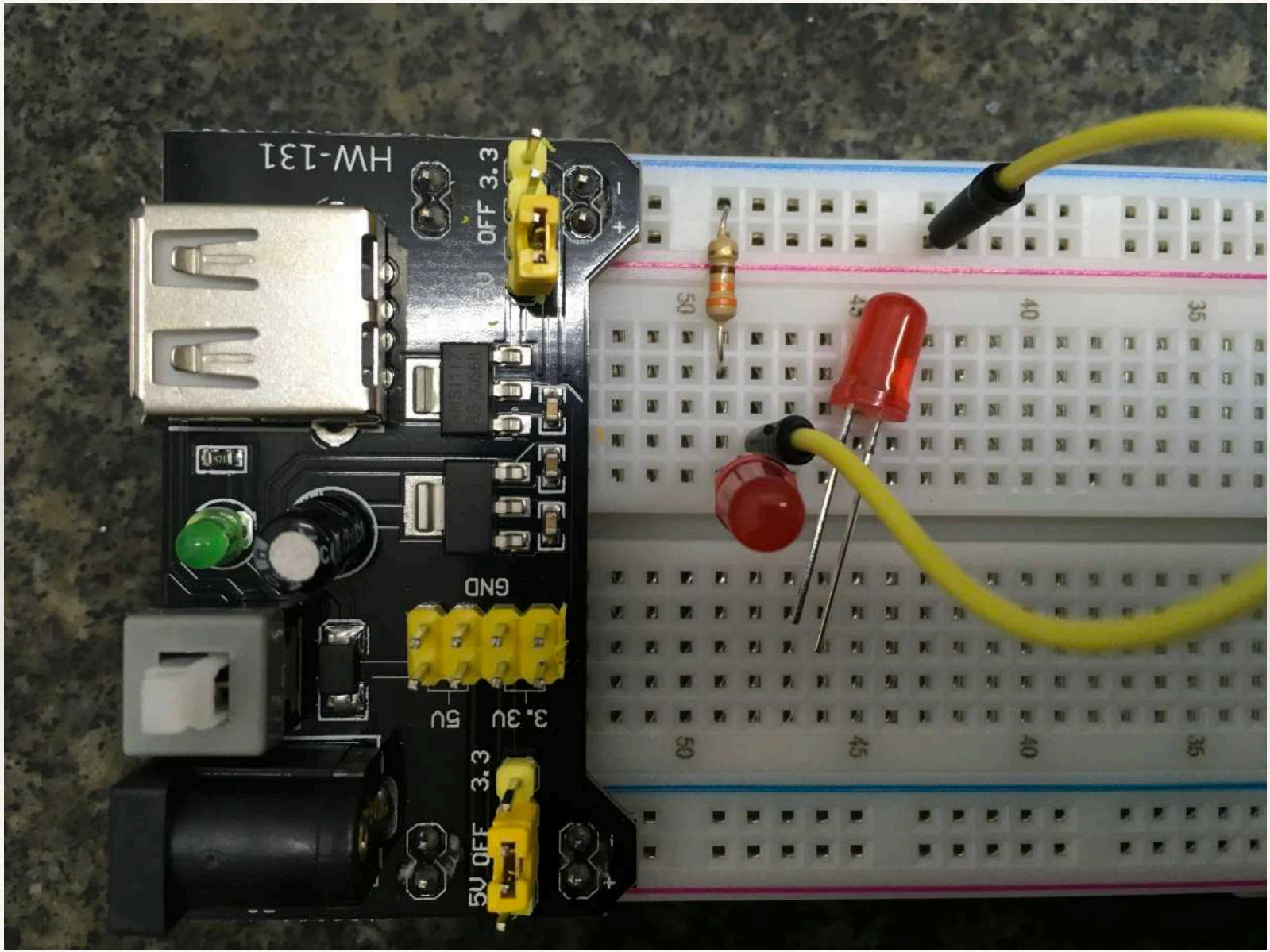
O LED está ligado aos dois pólos, com um resistor para controlar a corrente



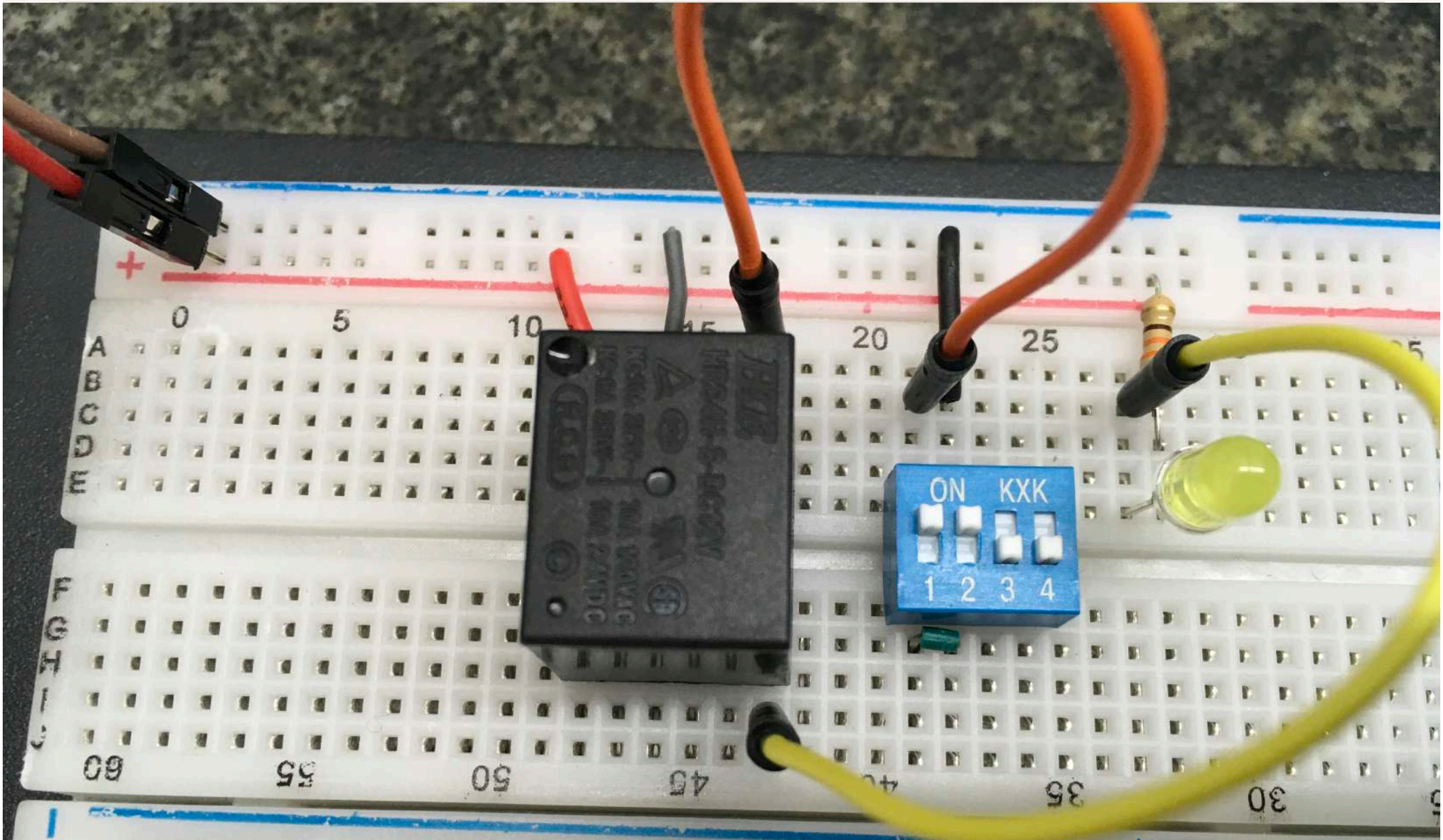
Substituindo a Pilha



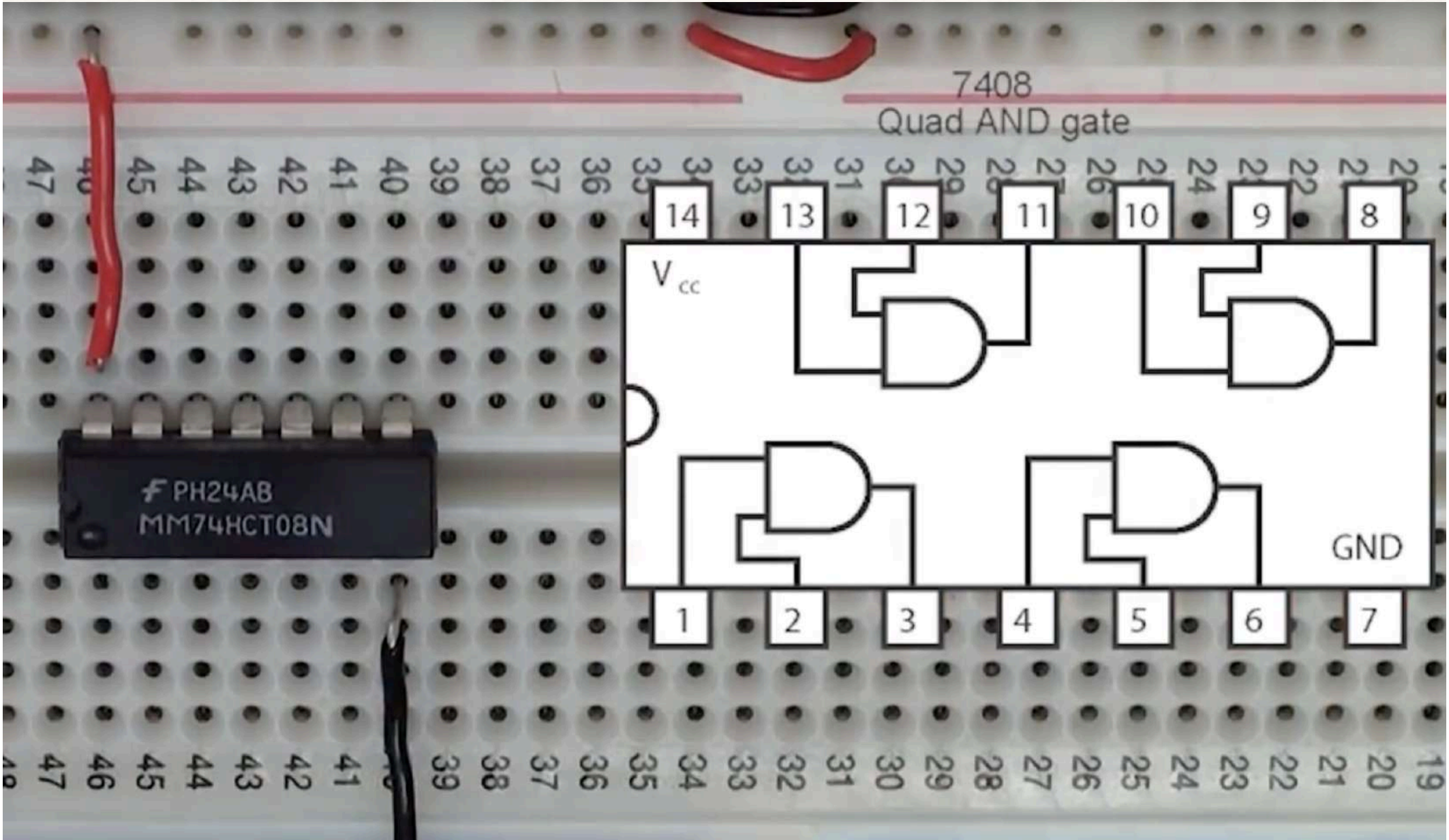
Acendendo um LED

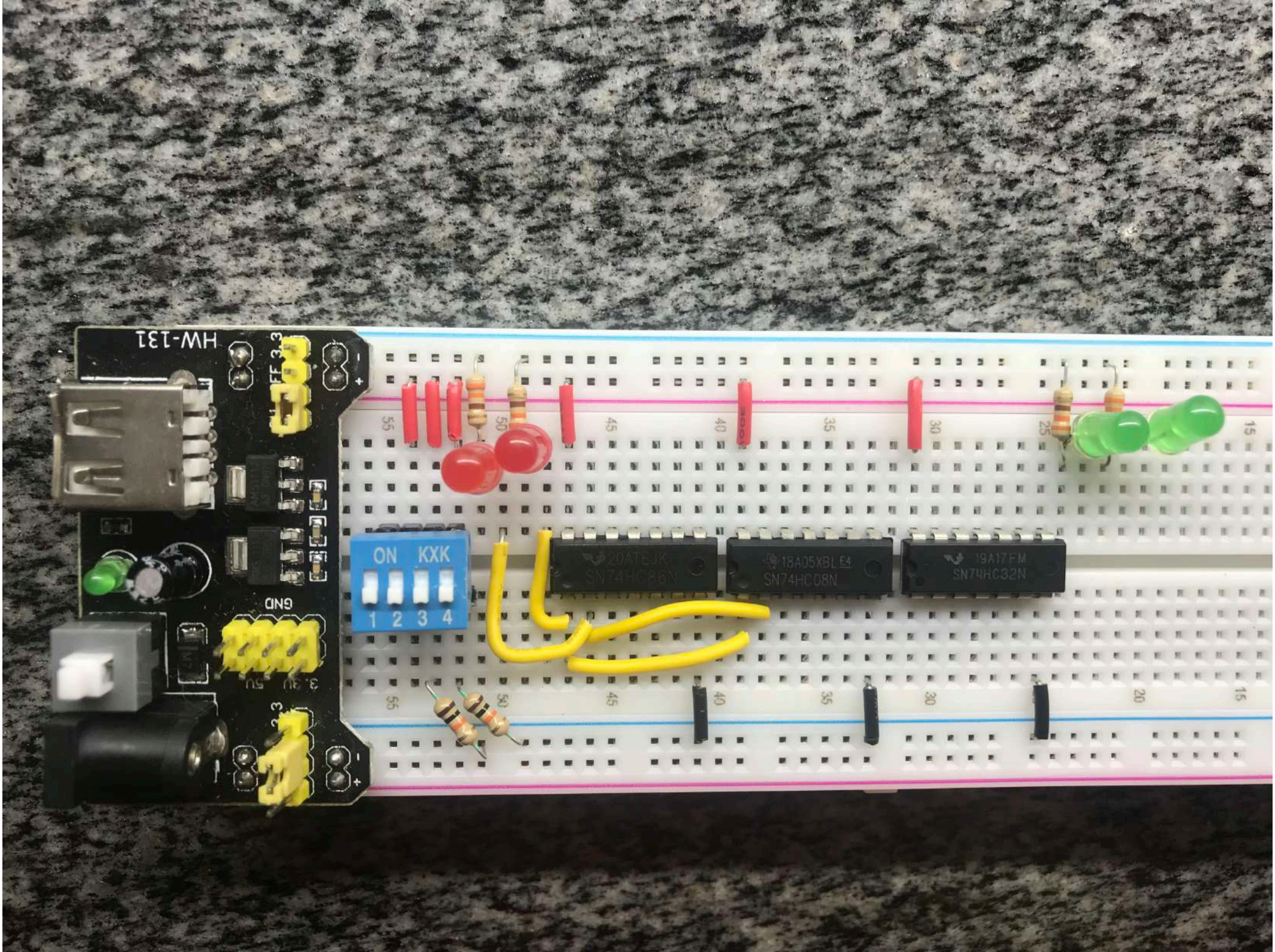


Porta NAND com Relê

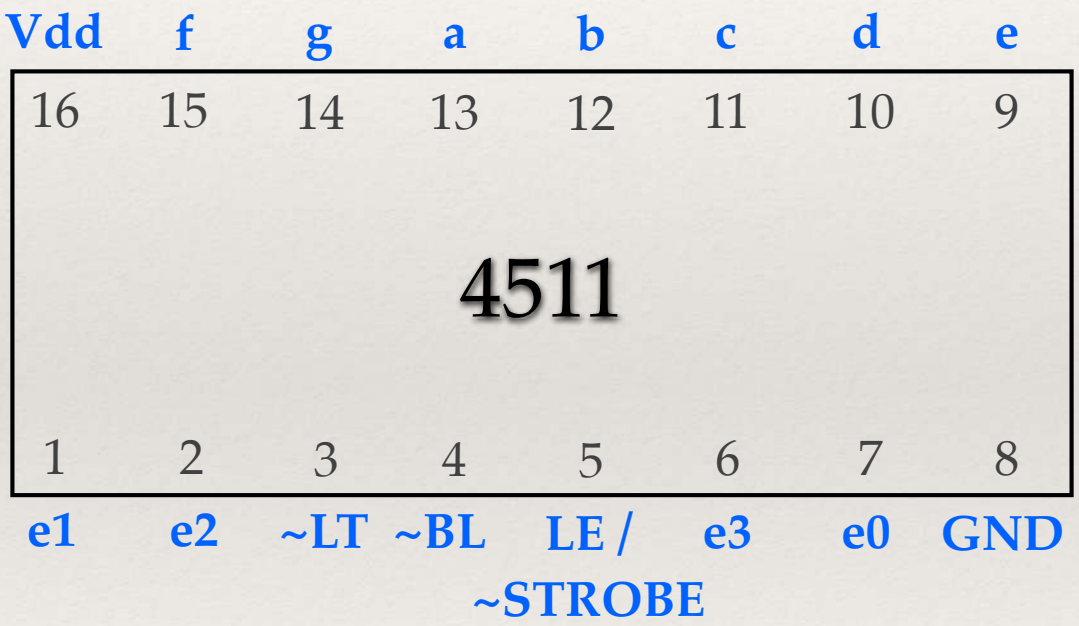


Ligando um Circuito Integrado (7408)

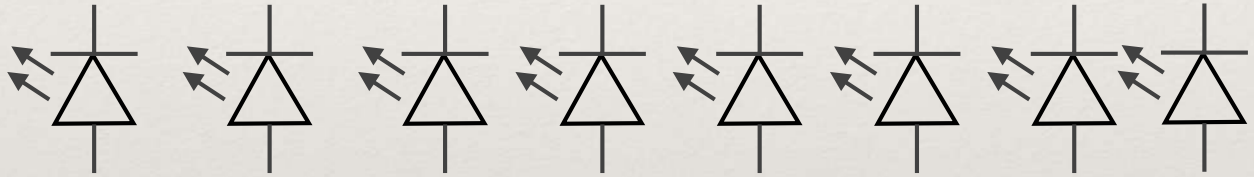



HW-131
ON KXK
1 2 3 4
SN74HC08N
SN74HC32N

BCD>7 segmentos (4511)



Display 7 segm. (5611AH)



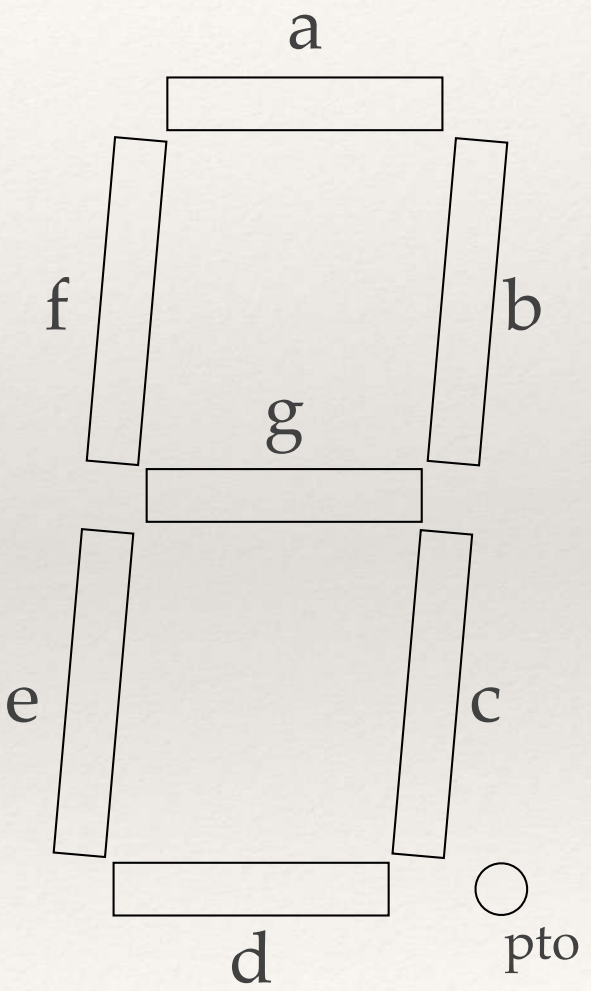
Síntese de Circuitos Lógicos

Display de sete segmentos

- Representa números decimais (e hexadecimais) com base em sete LEDs dispostos em um arranjo específico;
- Muito utilizado para representação de valores.

Situação clássica de síntese de circuito lógico

- Como converter um número binário (ou BCD) no acendimento dos segmentos correspondentes?
- Cada segmento tem seu circuito lógico.



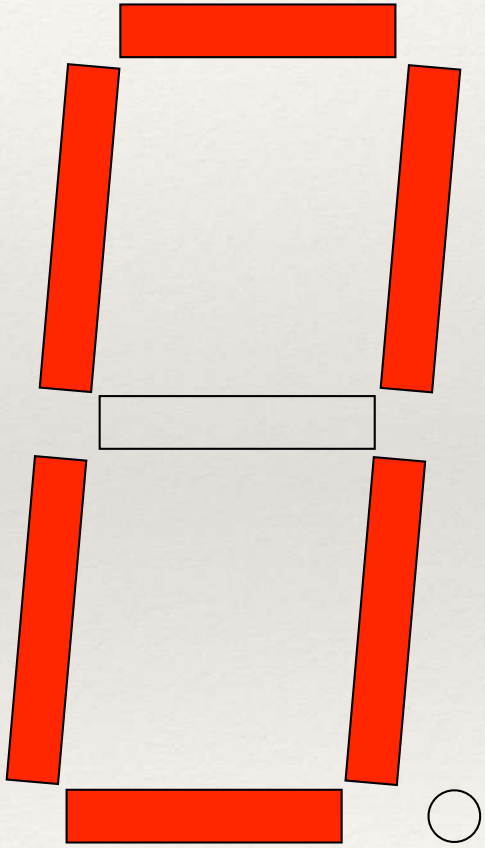
Síntese de Circuitos Lógicos

Display de sete segmentos

- Representa números decimais (e hexadecimais) com base em sete LEDs dispostos em um arranjo específico;
- Muito utilizado para representação de valores.

Situação clássica de síntese de circuito lógico

- Como converter um número binário (ou **BCD**) no acendimento dos segmentos correspondentes?
- Cada segmento tem seu circuito lógico.



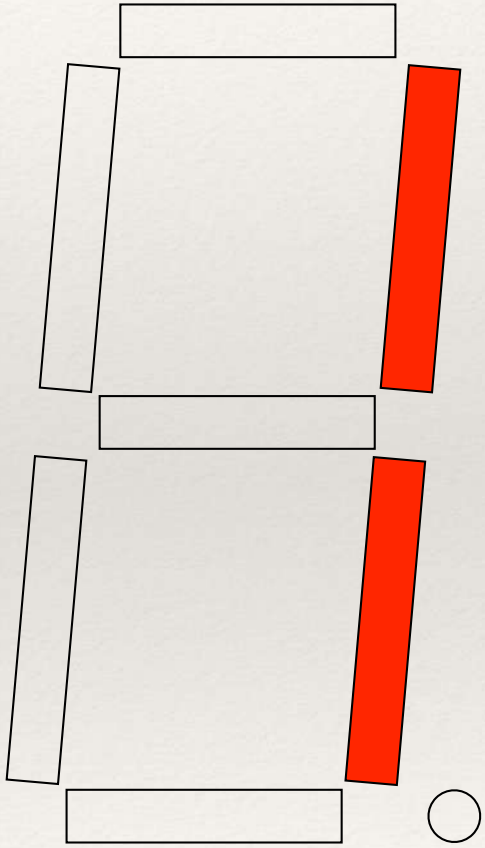
Síntese de Circuitos Lógicos

Display de sete segmentos

- Representa números decimais (e hexadecimais) com base em sete LEDs dispostos em um arranjo específico;
- Muito utilizado para representação de valores.

Situação clássica de síntese de circuito lógico

- Como converter um número binário (ou **BCD**) no acendimento dos segmentos correspondentes?
- Cada segmento tem seu circuito lógico.



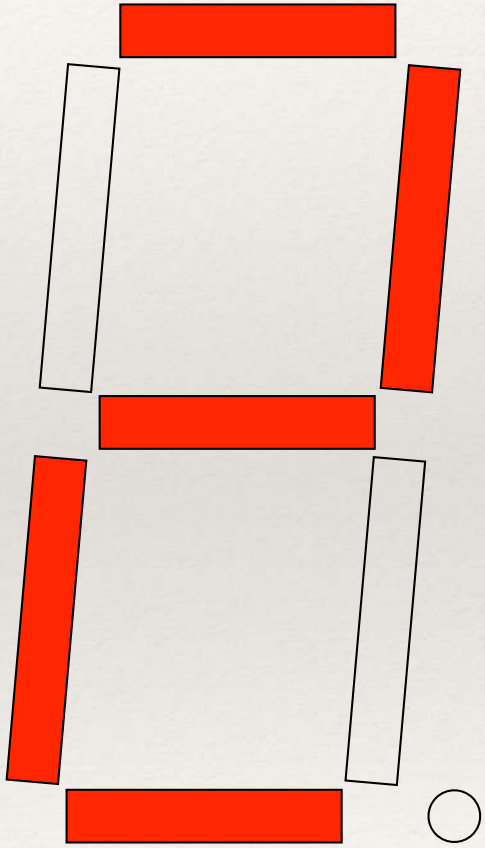
Síntese de Circuitos Lógicos

Display de sete segmentos

- Representa números decimais (e hexadecimais) com base em sete LEDs dispostos em um arranjo específico;
- Muito utilizado para representação de valores.

Situação clássica de síntese de circuito lógico

- Como converter um número binário (ou **BCD**) no acendimento dos segmentos correspondentes?
- Cada segmento tem seu circuito lógico.



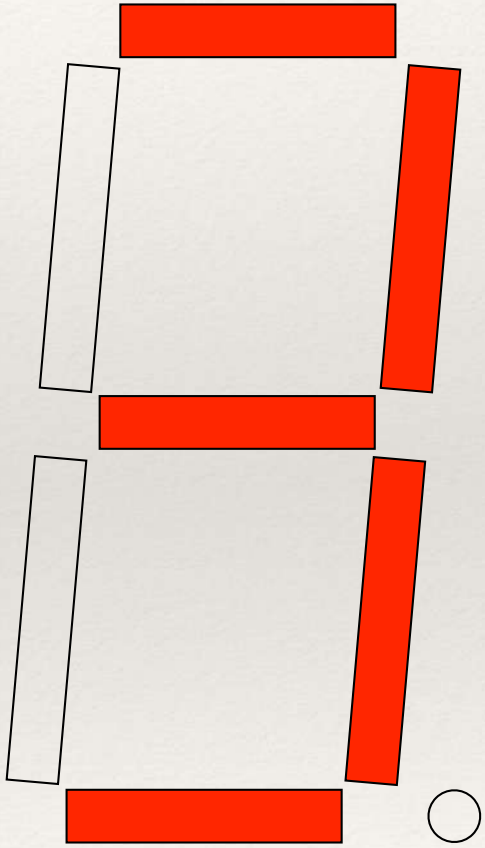
Síntese de Circuitos Lógicos

Display de sete segmentos

- Representa números decimais (e hexadecimais) com base em sete LEDs dispostos em um arranjo específico;
- Muito utilizado para representação de valores.

Situação clássica de síntese de circuito lógico

- Como converter um número binário (ou **BCD**) no acendimento dos segmentos correspondentes?
- Cada segmento tem seu circuito lógico.



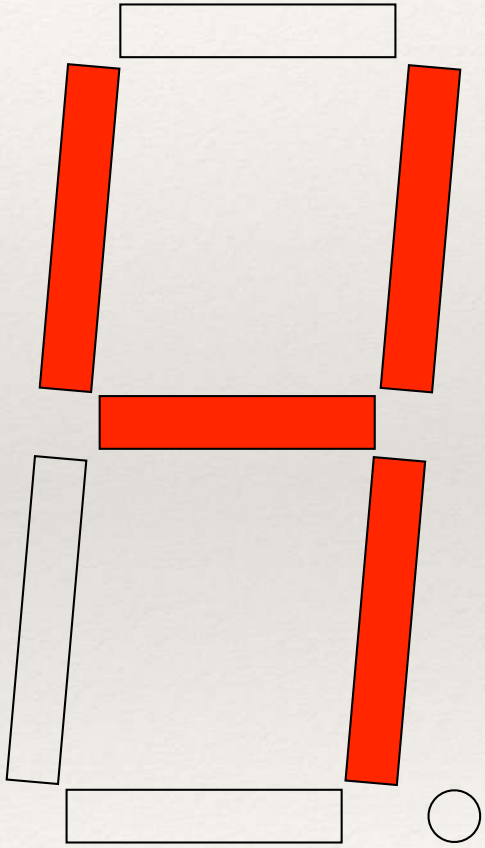
Síntese de Circuitos Lógicos

Display de sete segmentos

- Representa números decimais (e hexadecimais) com base em sete LEDs dispostos em um arranjo específico;
- Muito utilizado para representação de valores.

Situação clássica de síntese de circuito lógico

- Como converter um número binário (ou **BCD**) no acendimento dos segmentos correspondentes?
- Cada segmento tem seu circuito lógico.



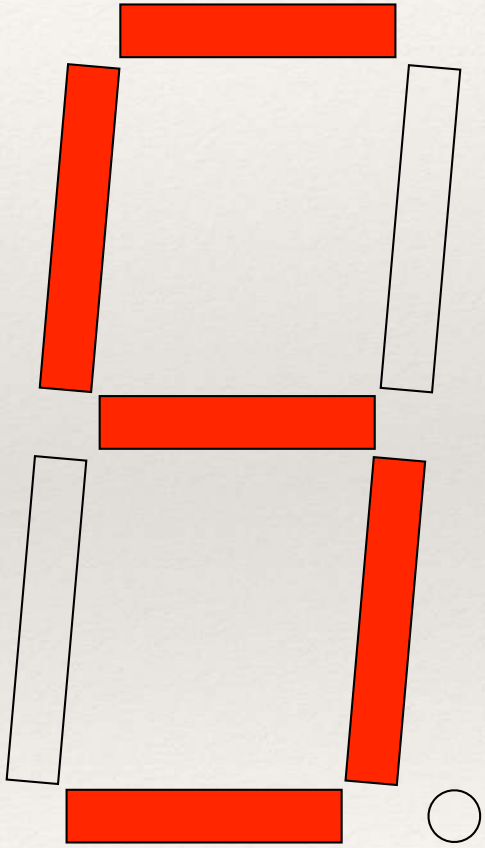
Síntese de Circuitos Lógicos

Display de sete segmentos

- Representa números decimais (e hexadecimais) com base em sete LEDs dispostos em um arranjo específico;
- Muito utilizado para representação de valores.

Situação clássica de síntese de circuito lógico

- Como converter um número binário (ou **BCD**) no acendimento dos segmentos correspondentes?
- Cada segmento tem seu circuito lógico.



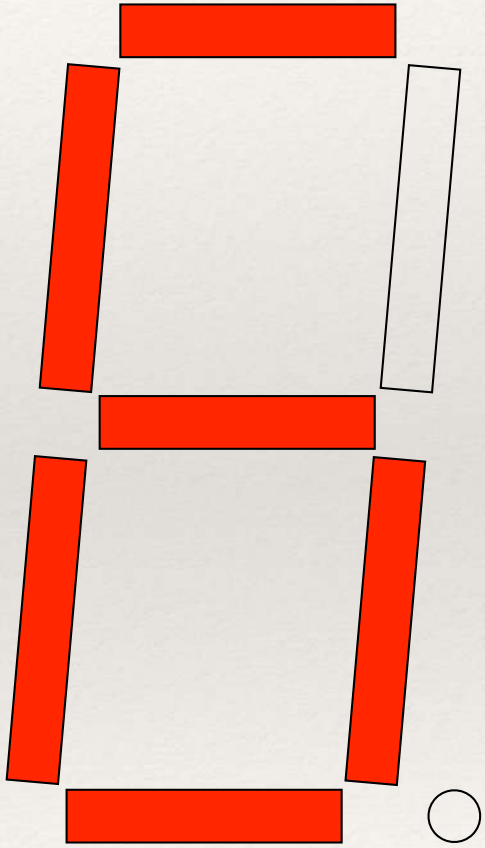
Síntese de Circuitos Lógicos

Display de sete segmentos

- Representa números decimais (e hexadecimais) com base em sete LEDs dispostos em um arranjo específico;
- Muito utilizado para representação de valores.

Situação clássica de síntese de circuito lógico

- Como converter um número binário (ou **BCD**) no acendimento dos segmentos correspondentes?
- Cada segmento tem seu circuito lógico.



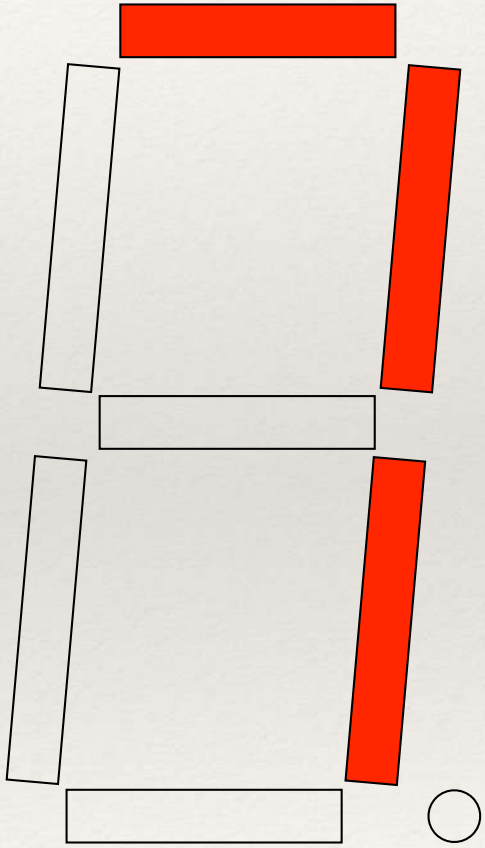
Síntese de Circuitos Lógicos

Display de sete segmentos

- Representa números decimais (e hexadecimais) com base em sete LEDs dispostos em um arranjo específico;
- Muito utilizado para representação de valores.

Situação clássica de síntese de circuito lógico

- Como converter um número binário (ou **BCD**) no acendimento dos segmentos correspondentes?
- Cada segmento tem seu circuito lógico.



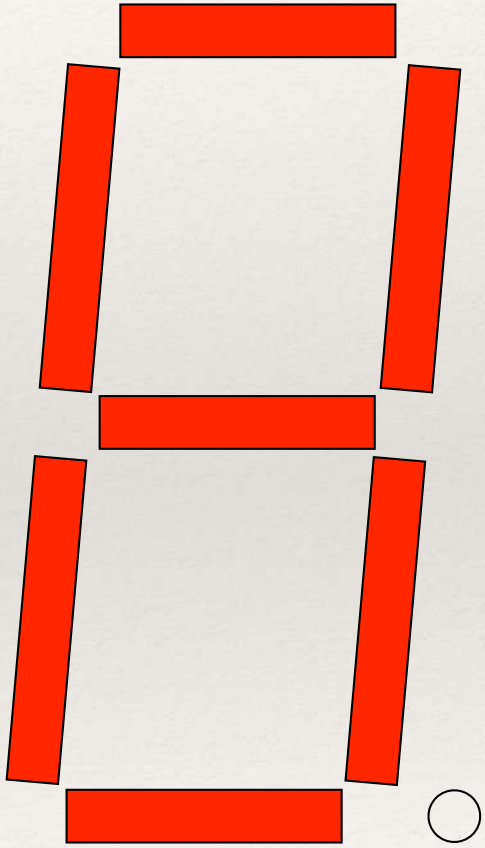
Síntese de Circuitos Lógicos

Display de sete segmentos

- Representa números decimais (e hexadecimais) com base em sete LEDs dispostos em um arranjo específico;
- Muito utilizado para representação de valores.

Situação clássica de síntese de circuito lógico

- Como converter um número binário (ou **BCD**) no acendimento dos segmentos correspondentes?
- Cada segmento tem seu circuito lógico.



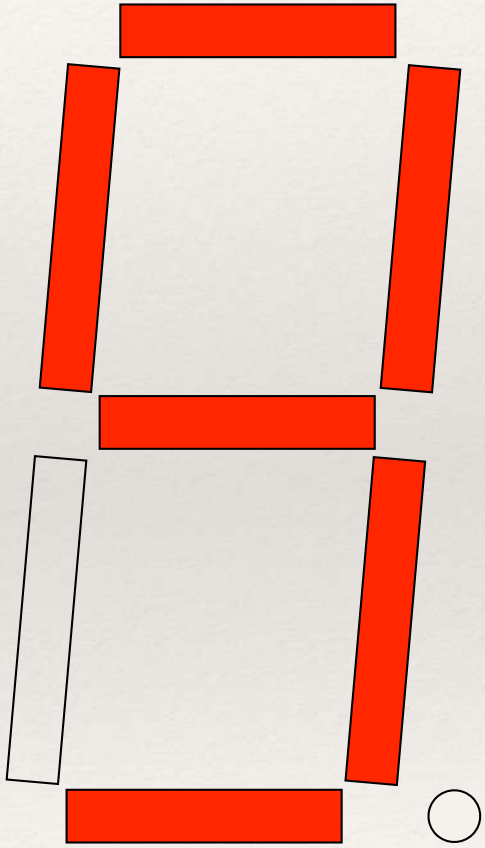
Síntese de Circuitos Lógicos

Display de sete segmentos

- Representa números decimais (e hexadecimais) com base em sete LEDs dispostos em um arranjo específico;
- Muito utilizado para representação de valores.

Situação clássica de síntese de circuito lógico

- Como converter um número binário (ou **BCD**) no acendimento dos segmentos correspondentes?
- Cada segmento tem seu circuito lógico.



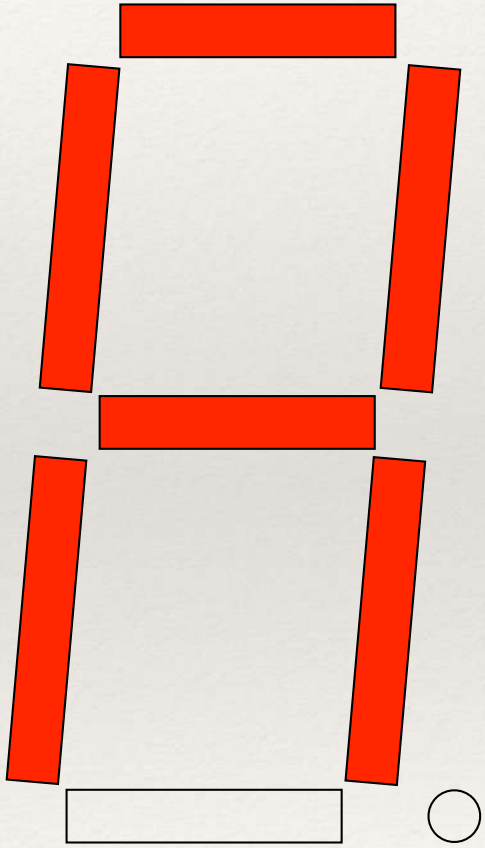
Síntese de Circuitos Lógicos

Display de sete segmentos

- Representa números decimais (e **hexadecimais**) com base em sete LEDs dispostos em um arranjo específico;
- Muito utilizado para representação de valores.

Situação clássica de síntese de circuito lógico

- Como converter um número binário (ou BCD) no acendimento dos segmentos correspondentes?
- Cada segmento tem seu circuito lógico.



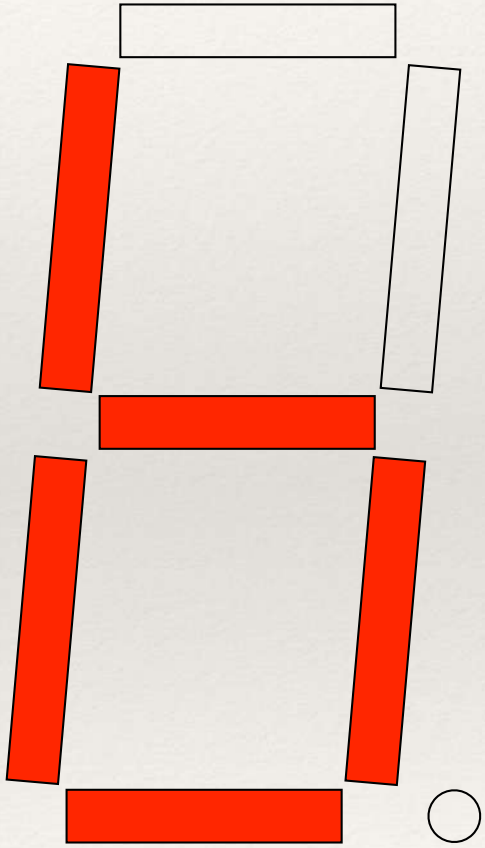
Síntese de Circuitos Lógicos

Display de sete segmentos

- Representa números decimais (e **hexadecimais**) com base em sete LEDs dispostos em um arranjo específico;
- Muito utilizado para representação de valores.

Situação clássica de síntese de circuito lógico

- Como converter um número binário (ou BCD) no acendimento dos segmentos correspondentes?
- Cada segmento tem seu circuito lógico.



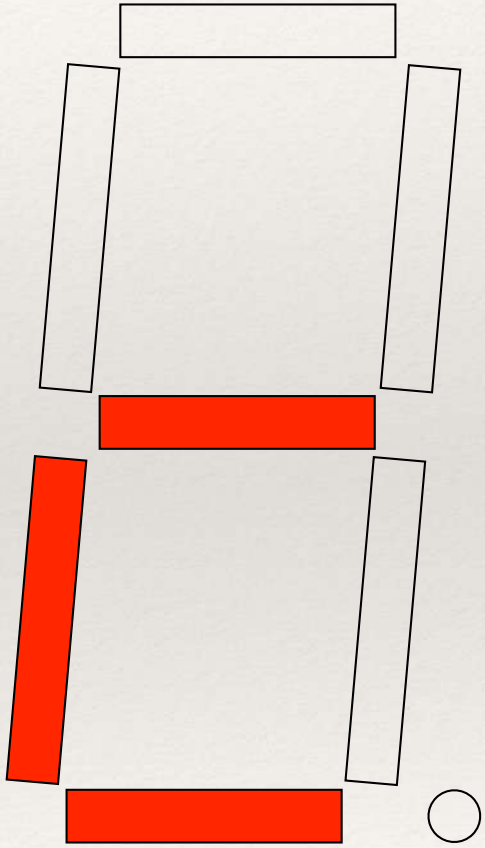
Síntese de Circuitos Lógicos

Display de sete segmentos

- Representa números decimais (e **hexadecimais**) com base em sete LEDs dispostos em um arranjo específico;
- Muito utilizado para representação de valores.

Situação clássica de síntese de circuito lógico

- Como converter um número binário (ou BCD) no acendimento dos segmentos correspondentes?
- Cada segmento tem seu circuito lógico.



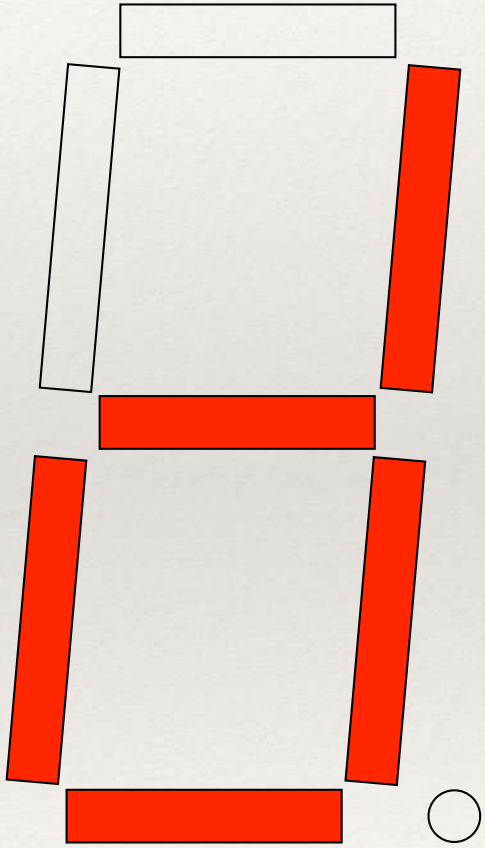
Síntese de Circuitos Lógicos

Display de sete segmentos

- Representa números decimais (e **hexadecimais**) com base em sete LEDs dispostos em um arranjo específico;
- Muito utilizado para representação de valores.

Situação clássica de síntese de circuito lógico

- Como converter um número binário (ou BCD) no acendimento dos segmentos correspondentes?
- Cada segmento tem seu circuito lógico.



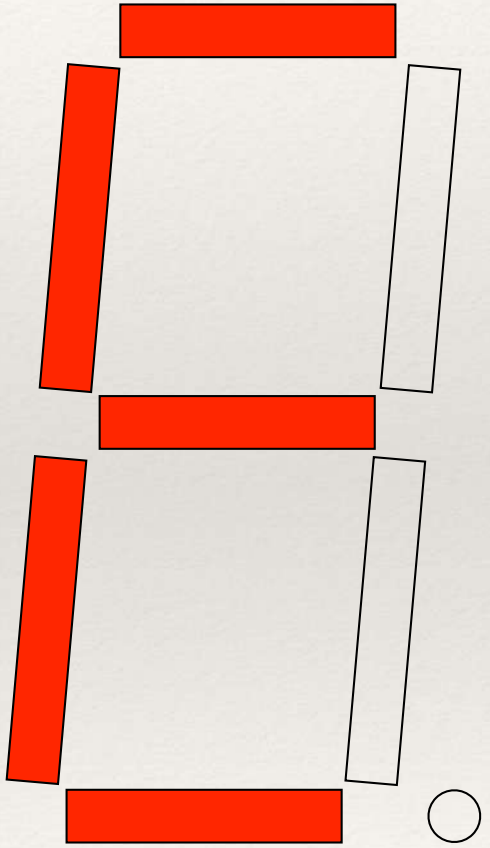
Síntese de Circuitos Lógicos

Display de sete segmentos

- Representa números decimais (e **hexadecimais**) com base em sete LEDs dispostos em um arranjo específico;
- Muito utilizado para representação de valores.

Situação clássica de síntese de circuito lógico

- Como converter um número binário (ou BCD) no acendimento dos segmentos correspondentes?
- Cada segmento tem seu circuito lógico.



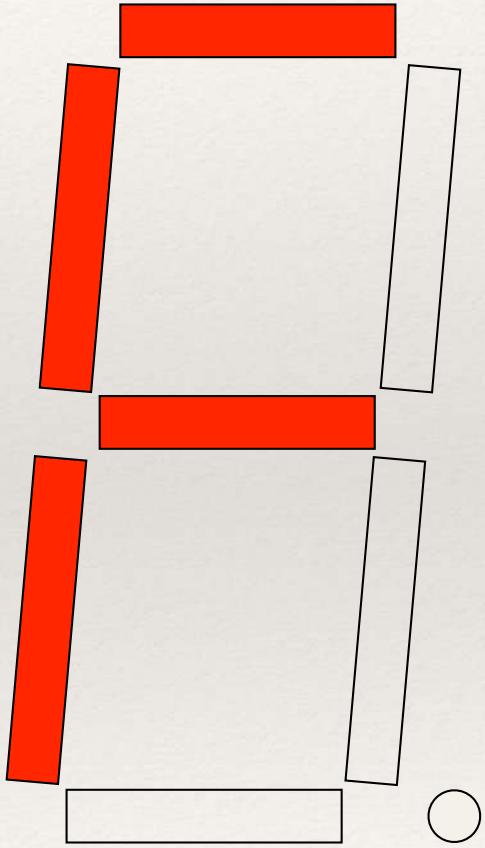
Síntese de Circuitos Lógicos

Display de sete segmentos

- Representa números decimais (e **hexadecimais**) com base em sete LEDs dispostos em um arranjo específico;
- Muito utilizado para representação de valores.

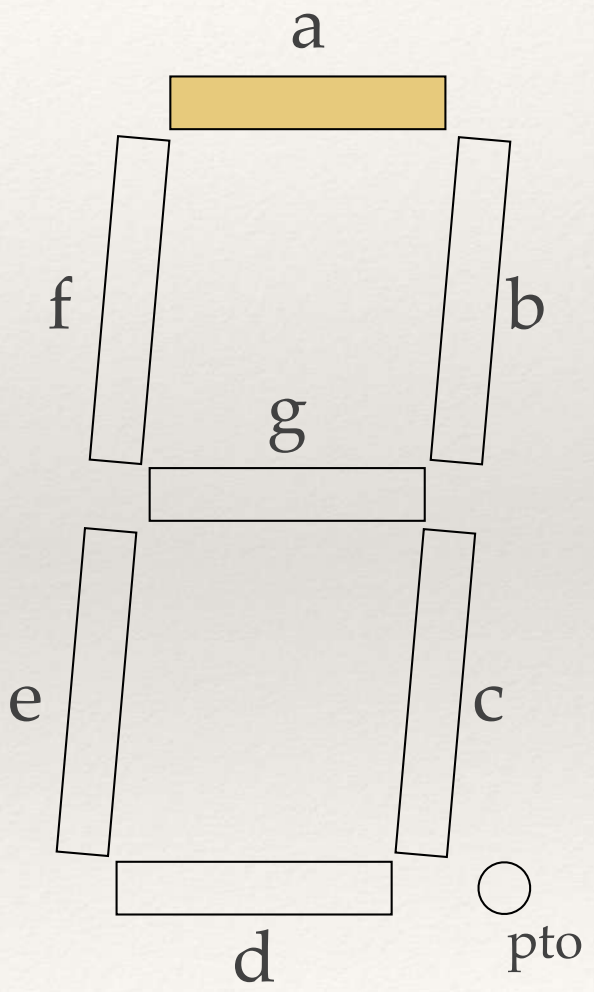
Situação clássica de síntese de circuito lógico

- Como converter um número binário (ou BCD) no acendimento dos segmentos correspondentes?
- Cada segmento tem seu circuito lógico.



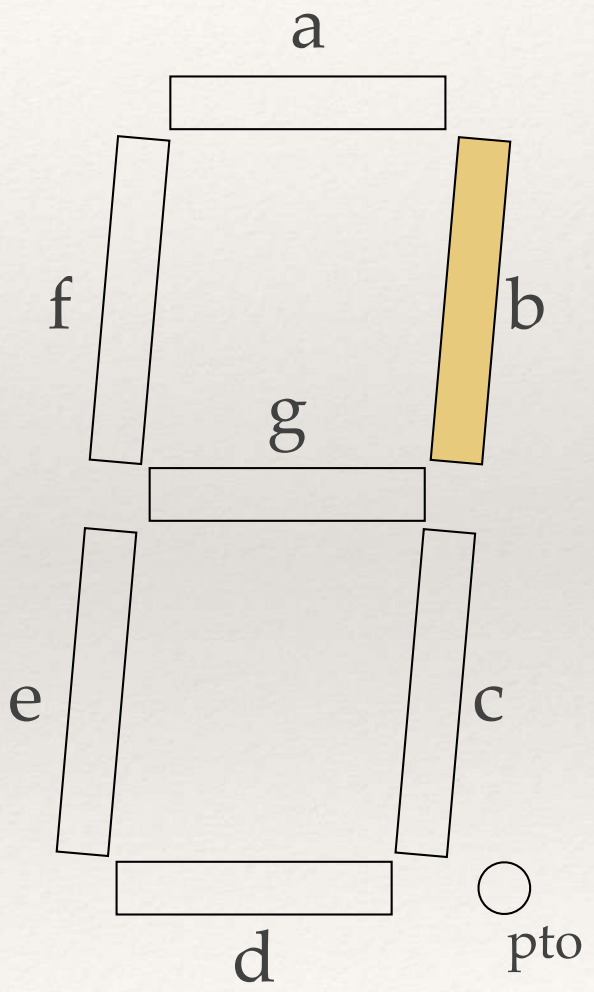
Síntese de Circuitos Lógicos

A	B	C	D	S	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1
1	0	1	0	A	1	1	1	0	1	1	1
1	0	1	1	B	0	0	1	1	1	1	1
1	1	0	0	C	0	0	0	1	1	0	1
1	1	0	1	D	0	1	1	1	1	0	1
1	1	1	0	E	1	0	0	1	1	1	1
1	1	1	1	F	1	0	0	0	1	1	1



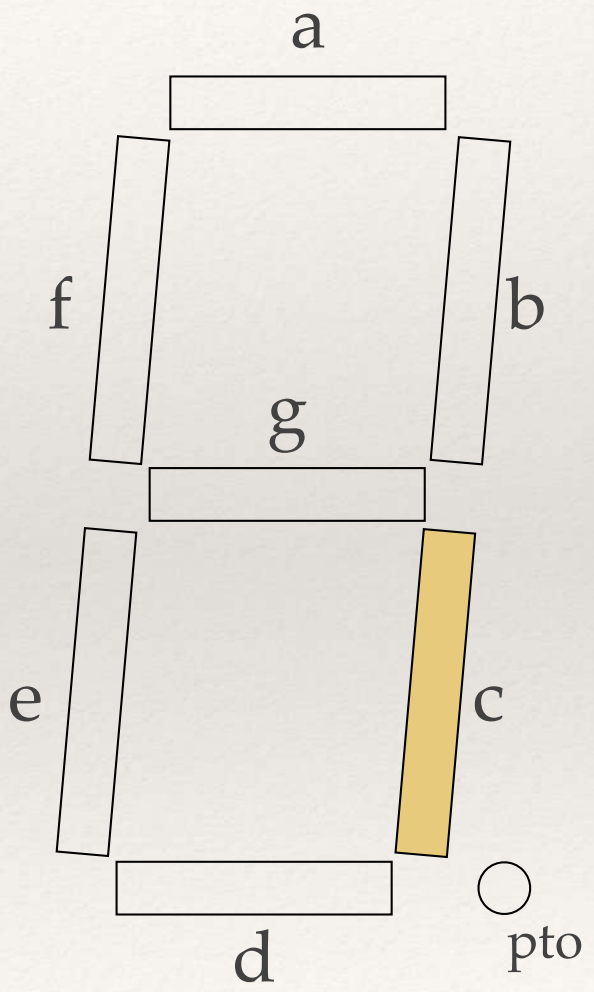
Síntese de Circuitos Lógicos

A	B	C	D	S	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1
1	0	1	0	A	1	1	1	0	1	1	1
1	0	1	1	B	0	0	1	1	1	1	1
1	1	0	0	C	0	0	0	1	1	0	1
1	1	0	1	D	0	1	1	1	1	0	1
1	1	1	0	E	1	0	0	1	1	1	1
1	1	1	1	F	1	0	0	0	1	1	1



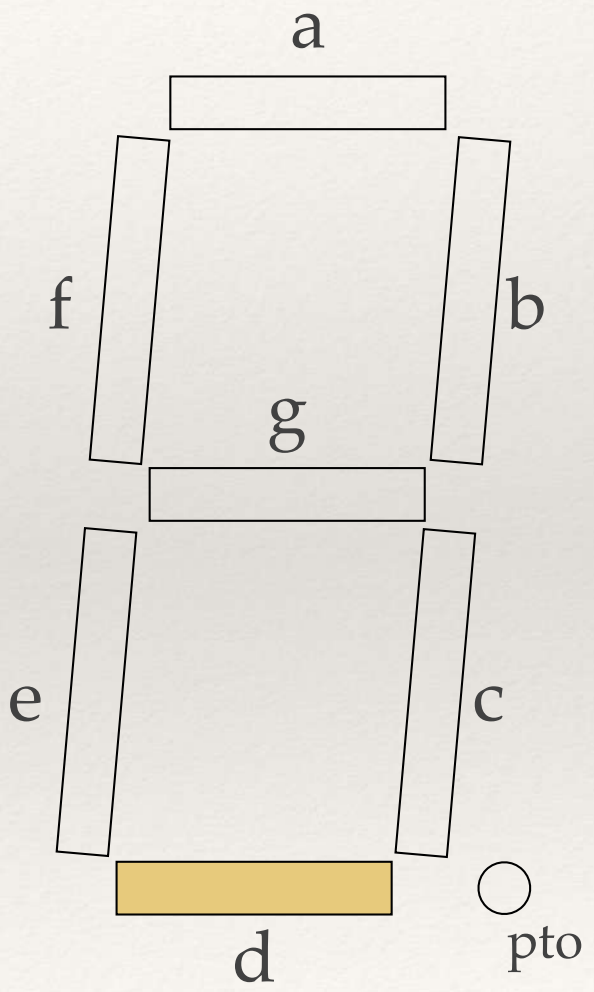
Síntese de Circuitos Lógicos

A	B	C	D	S	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1
1	0	1	0	A	1	1	1	0	1	1	1
1	0	1	1	B	0	0	1	1	1	1	1
1	1	0	0	C	0	0	0	1	1	0	1
1	1	0	1	D	0	1	1	1	1	0	1
1	1	1	0	E	1	0	0	1	1	1	1
1	1	1	1	F	1	0	0	0	1	1	1



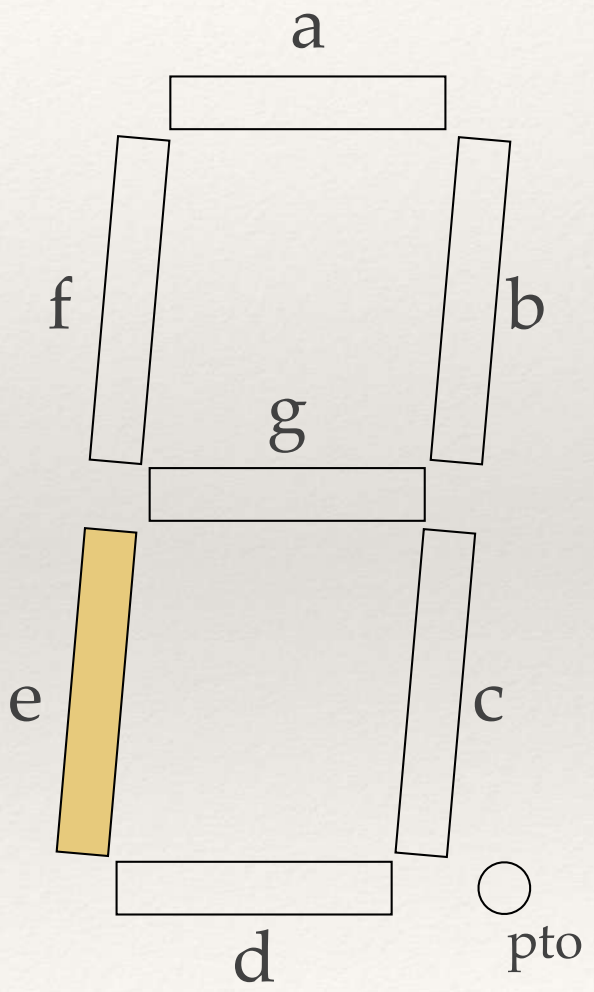
Síntese de Circuitos Lógicos

A	B	C	D	S	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1
1	0	1	0	A	1	1	1	0	1	1	1
1	0	1	1	B	0	0	1	1	1	1	1
1	1	0	0	C	0	0	0	1	1	0	1
1	1	0	1	D	0	1	1	1	1	0	1
1	1	1	0	E	1	0	0	1	1	1	1
1	1	1	1	F	1	0	0	0	1	1	1



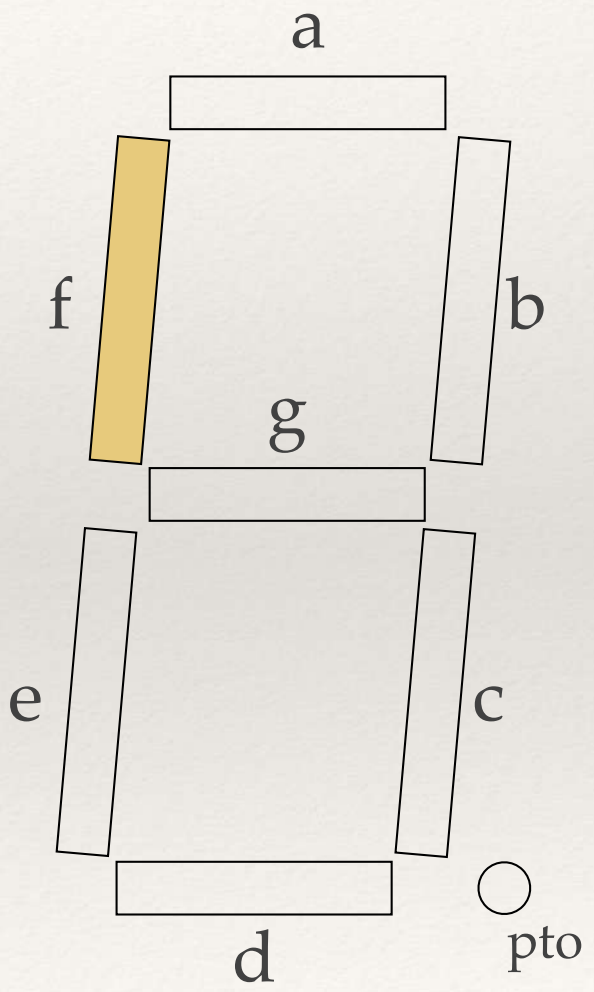
Síntese de Circuitos Lógicos

A	B	C	D	S	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1
1	0	1	0	A	1	1	1	0	1	1	1
1	0	1	1	B	0	0	1	1	1	1	1
1	1	0	0	C	0	0	0	1	1	0	1
1	1	0	1	D	0	1	1	1	1	0	1
1	1	1	0	E	1	0	0	1	1	1	1
1	1	1	1	F	1	0	0	0	1	1	1



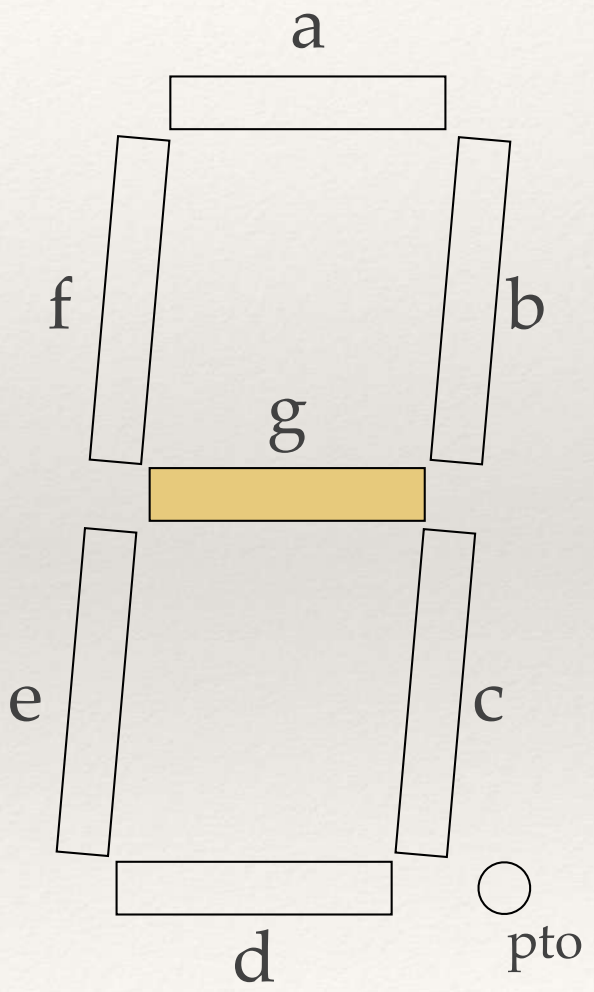
Síntese de Circuitos Lógicos

A	B	C	D	S	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1
1	0	1	0	A	1	1	1	0	1	1	1
1	0	1	1	B	0	0	1	1	1	1	1
1	1	0	0	C	0	0	0	1	1	0	1
1	1	0	1	D	0	1	1	1	1	0	1
1	1	1	0	E	1	0	0	1	1	1	1
1	1	1	1	F	1	0	0	0	1	1	1



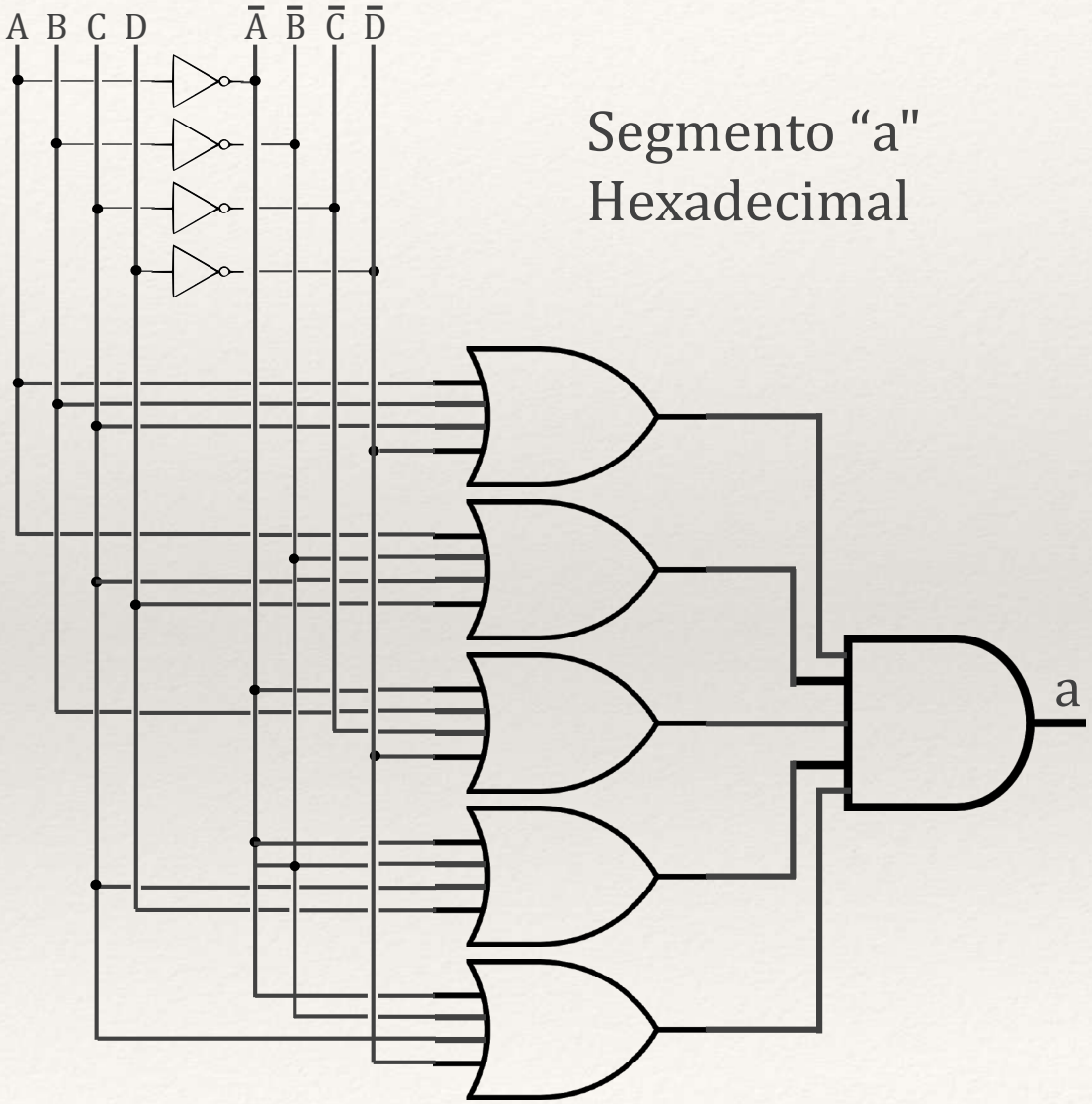
Síntese de Circuitos Lógicos

A	B	C	D	S	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1
1	0	1	0	A	1	1	1	0	1	1	1
1	0	1	1	B	0	0	1	1	1	1	1
1	1	0	0	C	0	0	0	1	1	0	1
1	1	0	1	D	0	1	1	1	1	0	1
1	1	1	0	E	1	0	0	1	1	1	1
1	1	1	1	F	1	0	0	0	1	1	1



Síntese de Circuitos Lógicos

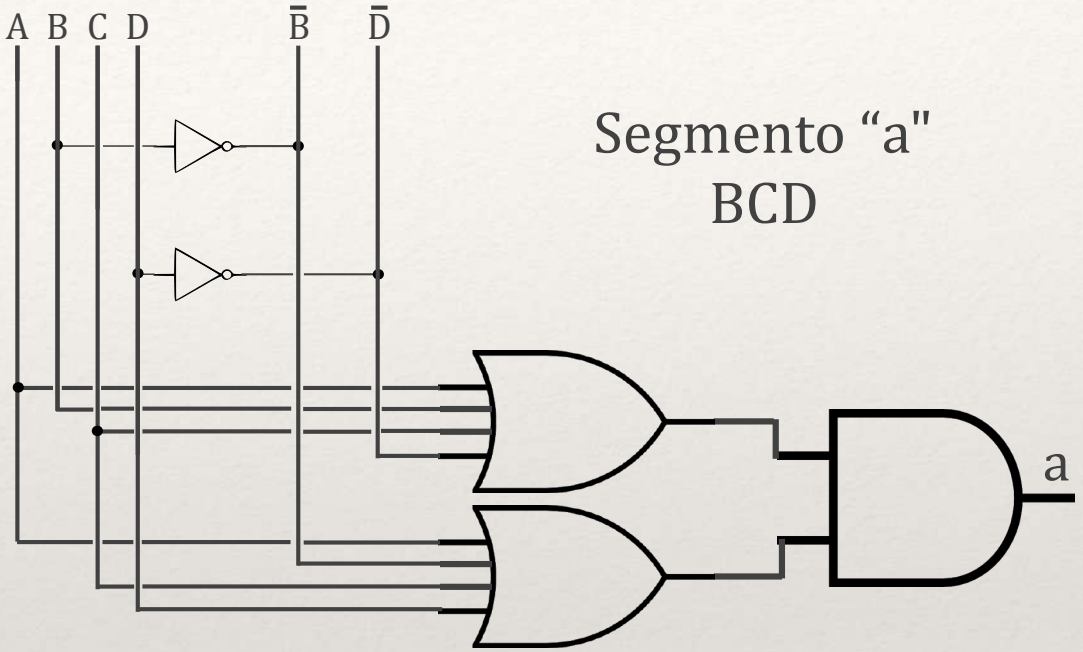
A	B	C	D	S	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1
1	0	1	0	A	1	1	1	0	1	1	1
1	0	1	1	B	0	0	1	1	1	1	1
1	1	0	0	C	0	0	0	1	1	0	1
1	1	0	1	D	0	1	1	1	1	0	1
1	1	1	0	E	1	0	0	1	1	1	1
1	1	1	1	F	1	0	0	0	1	1	1



Síntese de Circuitos Lógicos

A	B	C	D	S	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1
1	0	1	0	A	x	x	x	x	x	x	x
1	0	1	1	B	x	x	x	x	x	x	x
1	1	0	0	C	x	x	x	x	x	x	x
1	1	0	1	D	x	x	x	x	x	x	x
1	1	1	0	E	x	x	x	x	x	x	x
1	1	1	1	F	x	x	x	x	x	x	x

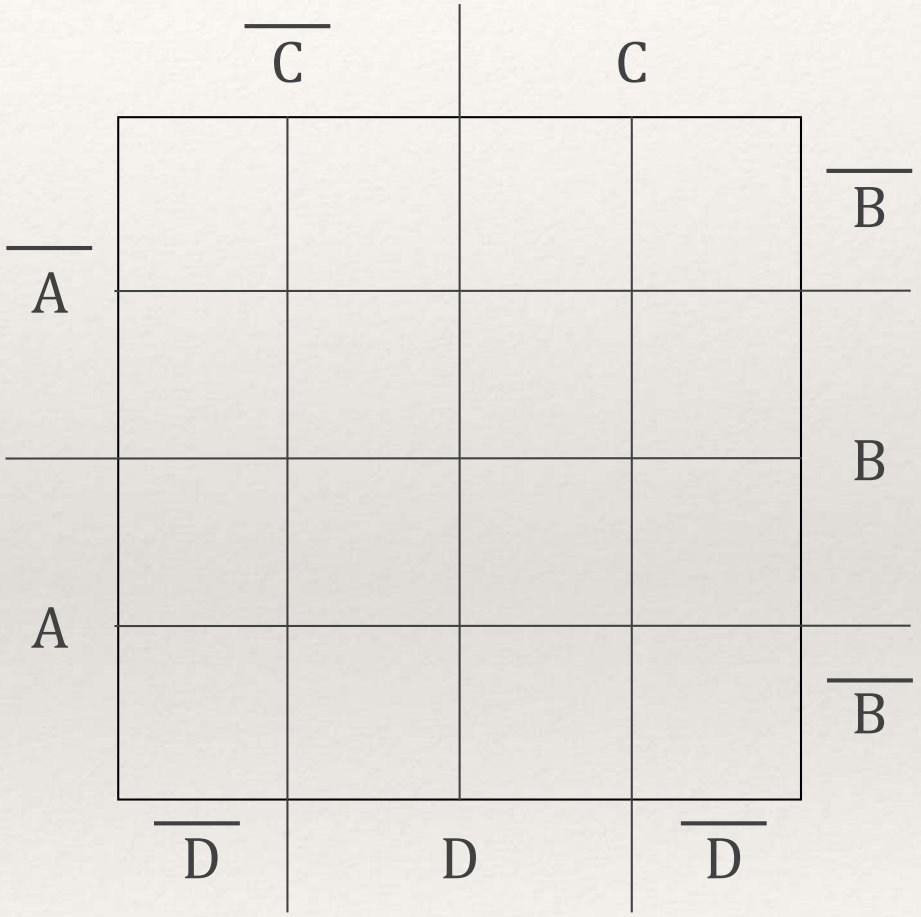
Condições Irrelevantes



Segmento “a" BCD

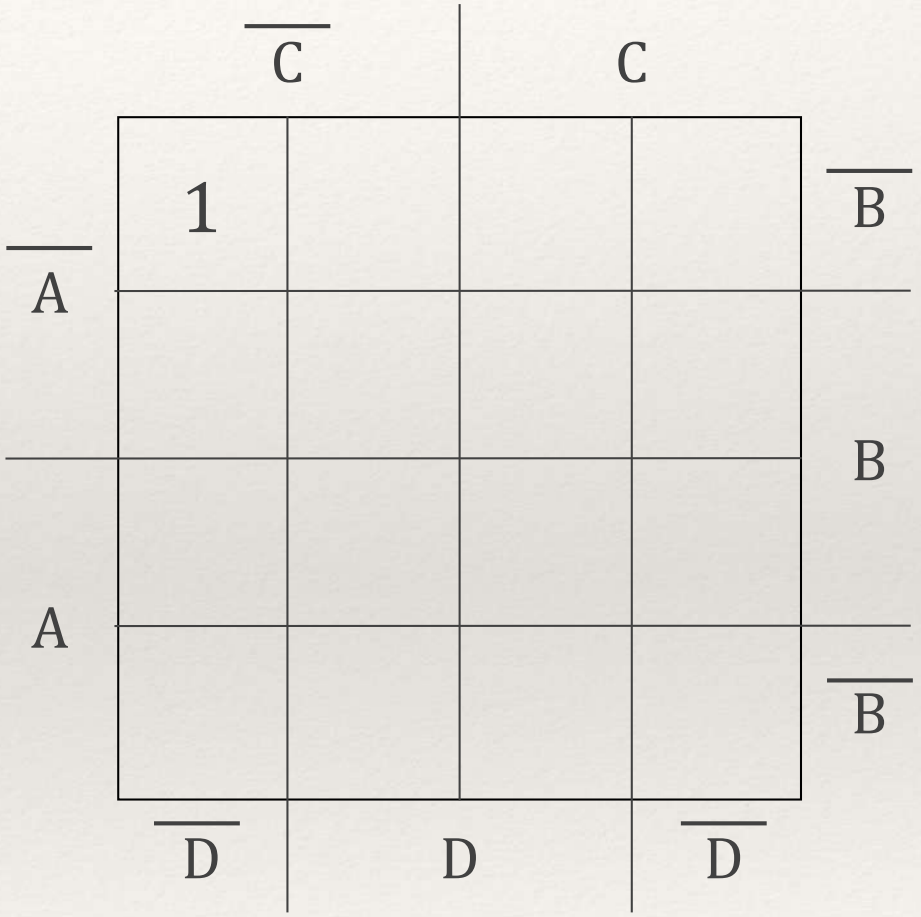
Simplificação de Circuitos Lógicos

A	B	C	D	S	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1
1	0	1	0	A	x	x	x	x	x	x	x
1	0	1	1	B	x	x	x	x	x	x	x
1	1	0	0	C	x	x	x	x	x	x	x
1	1	0	1	D	x	x	x	x	x	x	x
1	1	1	0	E	x	x	x	x	x	x	x
1	1	1	1	F	x	x	x	x	x	x	x



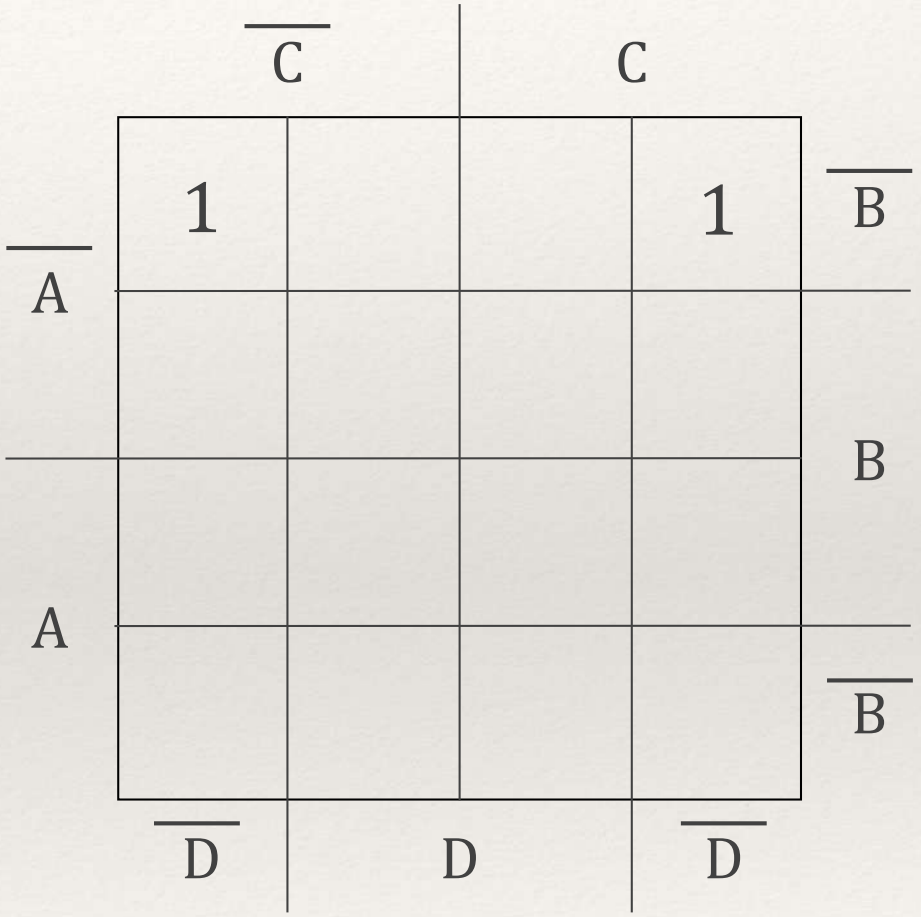
Simplificação de Circuitos Lógicos

A	B	C	D	S	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1
1	0	1	0	A	x	x	x	x	x	x	x
1	0	1	1	B	x	x	x	x	x	x	x
1	1	0	0	C	x	x	x	x	x	x	x
1	1	0	1	D	x	x	x	x	x	x	x
1	1	1	0	E	x	x	x	x	x	x	x
1	1	1	1	F	x	x	x	x	x	x	x



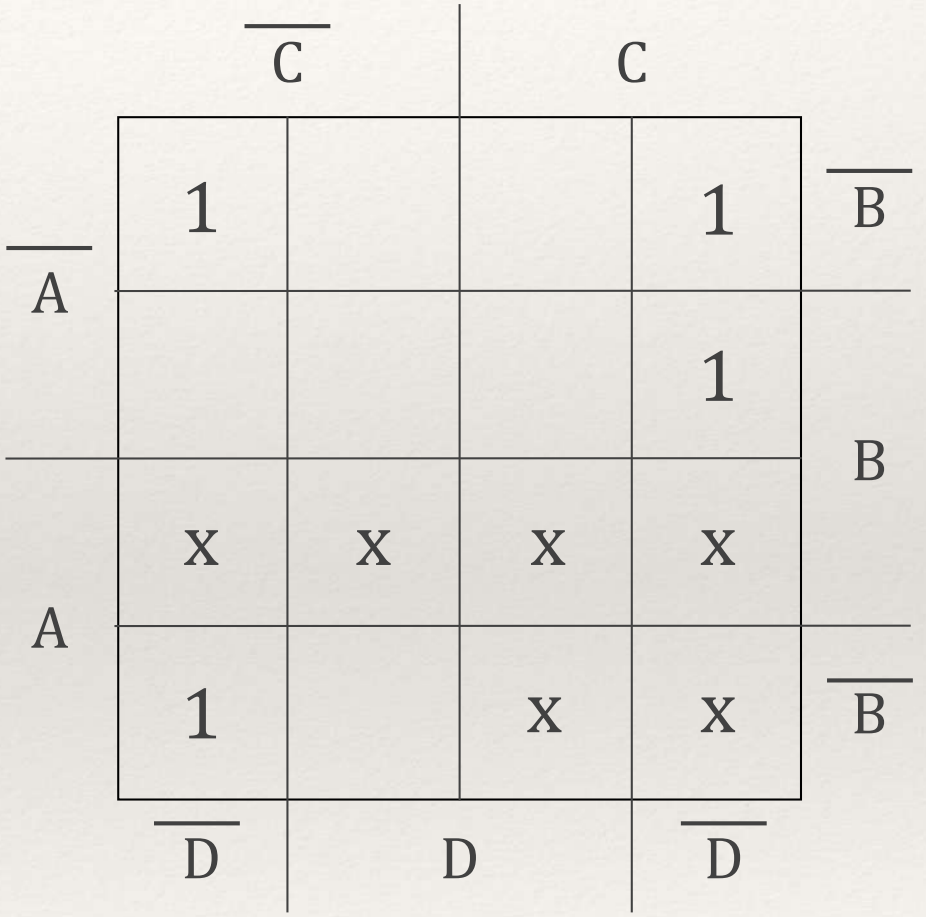
Simplificação de Circuitos Lógicos

A	B	C	D	S	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1
1	0	1	0	A	x	x	x	x	x	x	x
1	0	1	1	B	x	x	x	x	x	x	x
1	1	0	0	C	x	x	x	x	x	x	x
1	1	0	1	D	x	x	x	x	x	x	x
1	1	1	0	E	x	x	x	x	x	x	x
1	1	1	1	F	x	x	x	x	x	x	x



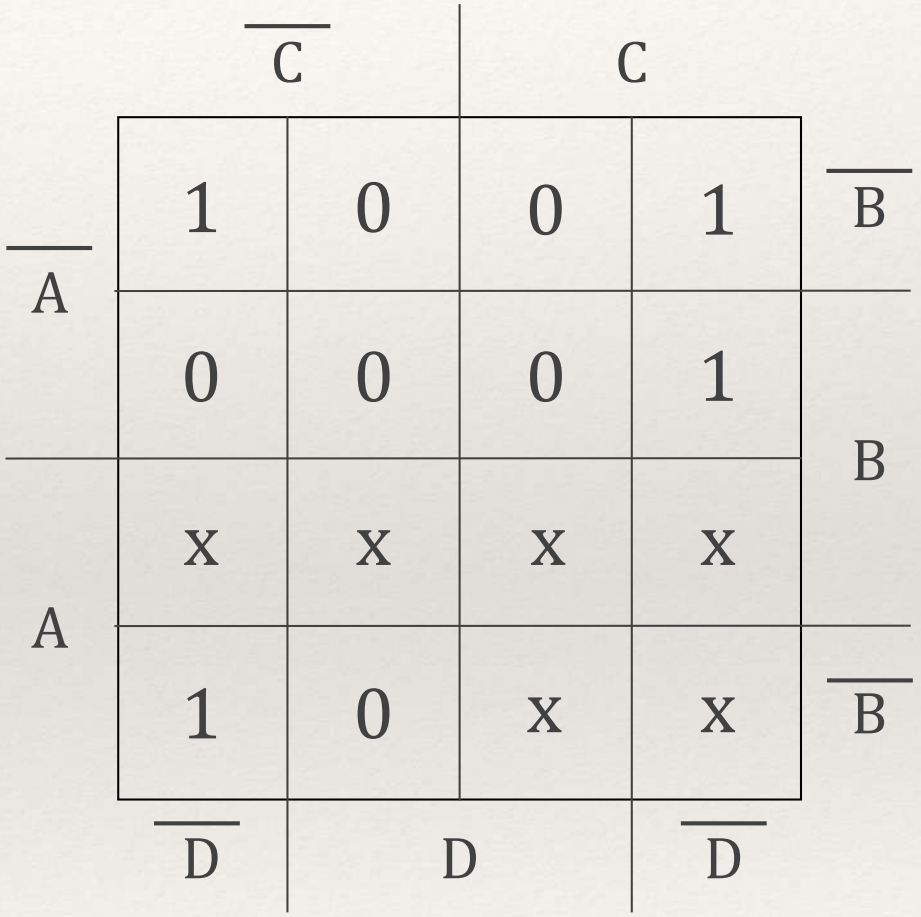
Simplificação de Circuitos Lógicos

A	B	C	D	S	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1
1	0	1	0	A	x	x	x	x	x	x	x
1	0	1	1	B	x	x	x	x	x	x	x
1	1	0	0	C	x	x	x	x	x	x	x
1	1	0	1	D	x	x	x	x	x	x	x
1	1	1	0	E	x	x	x	x	x	x	x
1	1	1	1	F	x	x	x	x	x	x	x



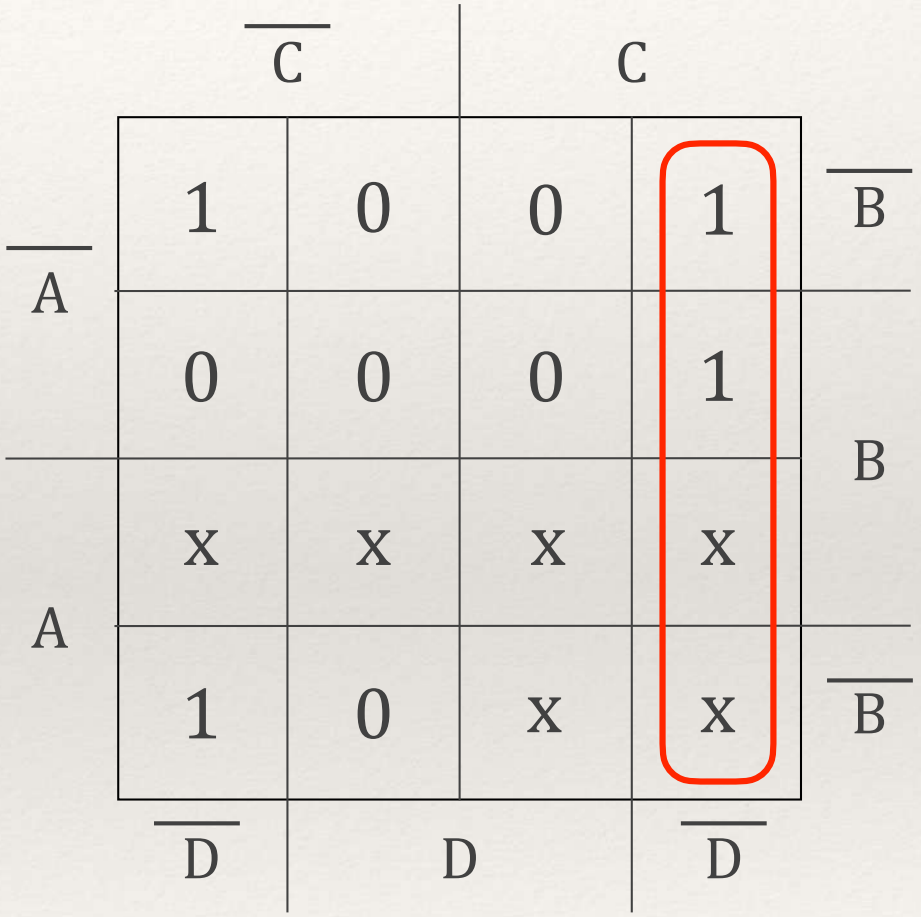
Simplificação de Circuitos Lógicos

A	B	C	D	S	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1
1	0	1	0	A	x	x	x	x	x	x	x
1	0	1	1	B	x	x	x	x	x	x	x
1	1	0	0	C	x	x	x	x	x	x	x
1	1	0	1	D	x	x	x	x	x	x	x
1	1	1	0	E	x	x	x	x	x	x	x
1	1	1	1	F	x	x	x	x	x	x	x



Simplificação de Circuitos Lógicos

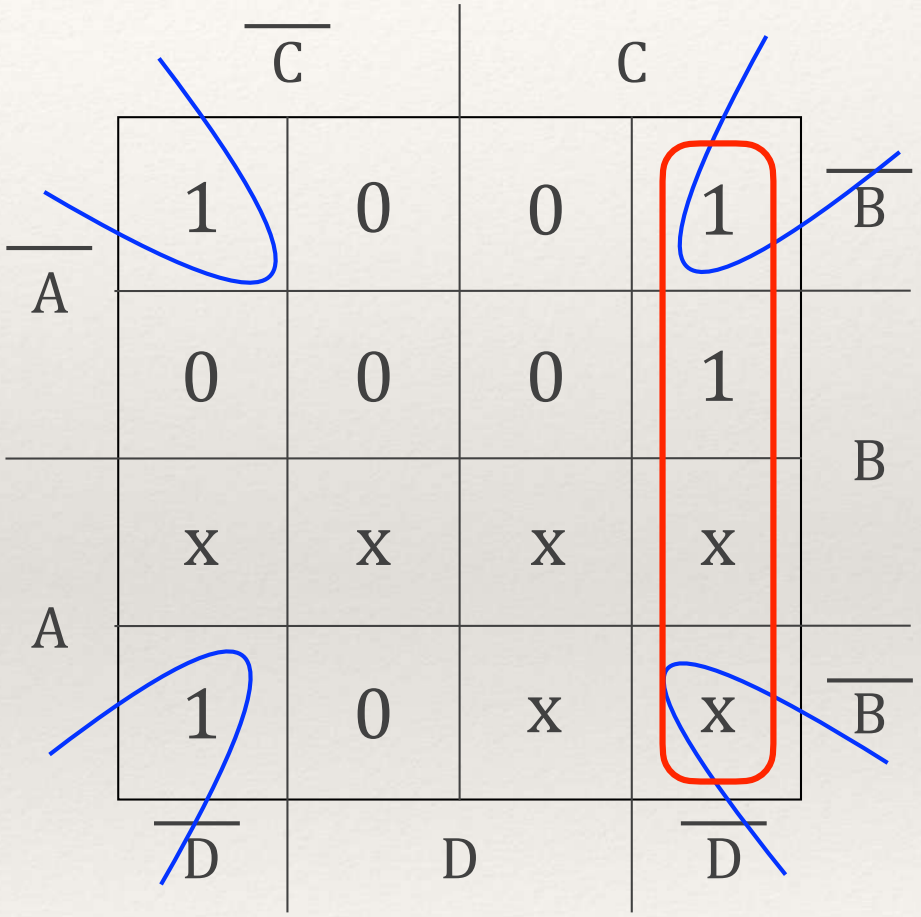
A	B	C	D	S	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1
1	0	1	0	A	x	x	x	x	x	x	x
1	0	1	1	B	x	x	x	x	x	x	x
1	1	0	0	C	x	x	x	x	x	x	x
1	1	0	1	D	x	x	x	x	x	x	x
1	1	1	0	E	x	x	x	x	x	x	x
1	1	1	1	F	x	x	x	x	x	x	x



$C \cdot \overline{D} +$

Simplificação de Circuitos Lógicos

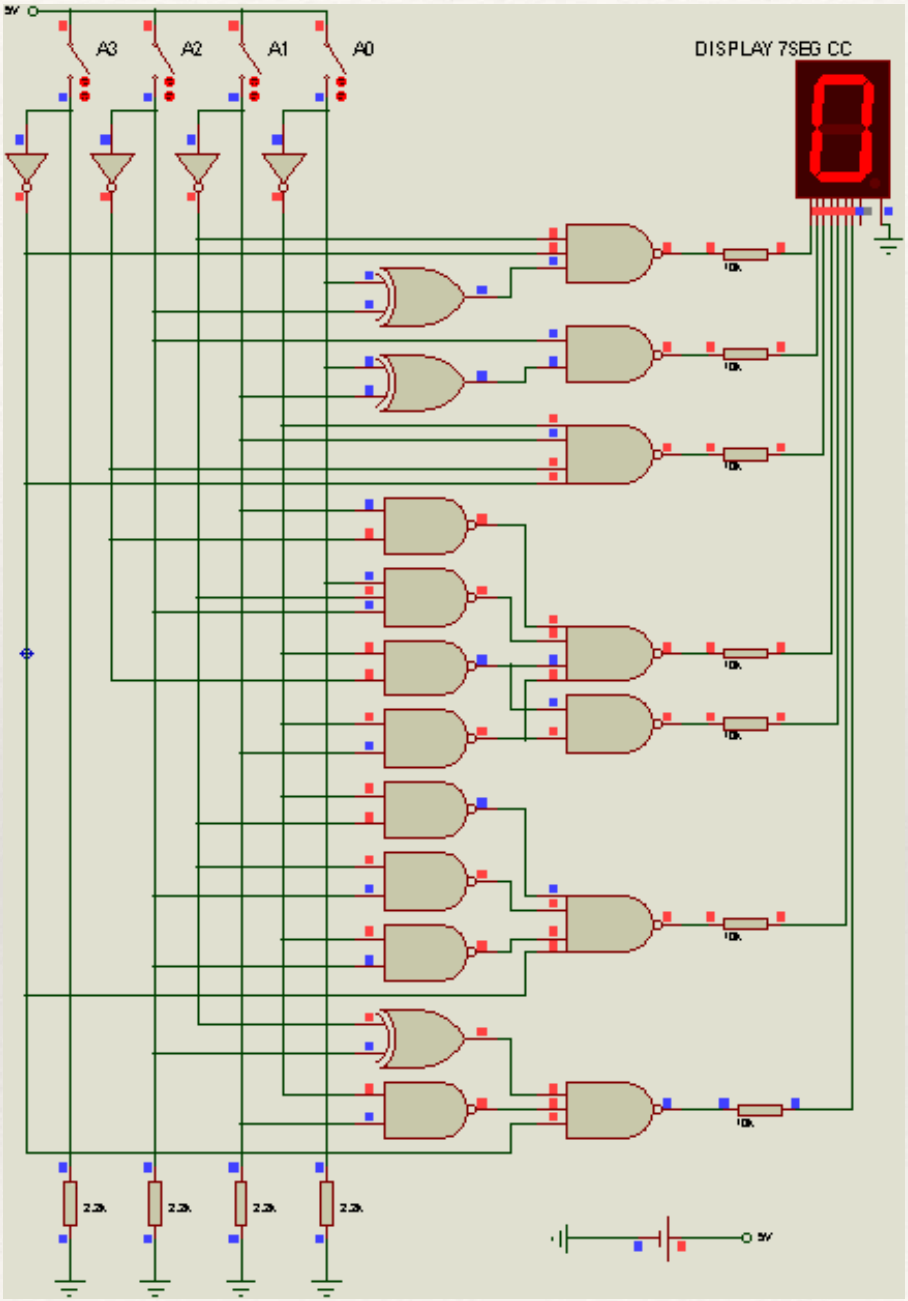
A	B	C	D	S	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1
1	0	1	0	A	x	x	x	x	x	x	x
1	0	1	1	B	x	x	x	x	x	x	x
1	1	0	0	C	x	x	x	x	x	x	x
1	1	0	1	D	x	x	x	x	x	x	x
1	1	1	0	E	x	x	x	x	x	x	x
1	1	1	1	F	x	x	x	x	x	x	x



$C \cdot \overline{D} + \overline{B} \cdot \overline{D}$ = segmento "e"

Circuito após simplificação

A	B	C	D	S	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1
1	0	1	0	A	x	x	x	x	x	x	x
1	0	1	1	B	x	x	x	x	x	x	x
1	1	0	0	C	x	x	x	x	x	x	x
1	1	0	1	D	x	x	x	x	x	x	x
1	1	1	0	E	x	x	x	x	x	x	x
1	1	1	1	F	x	x	x	x	x	x	x



Síntese de Circuitos Lógicos I

Dada uma tabela verdade, como obter o circuito lógico equivalente?

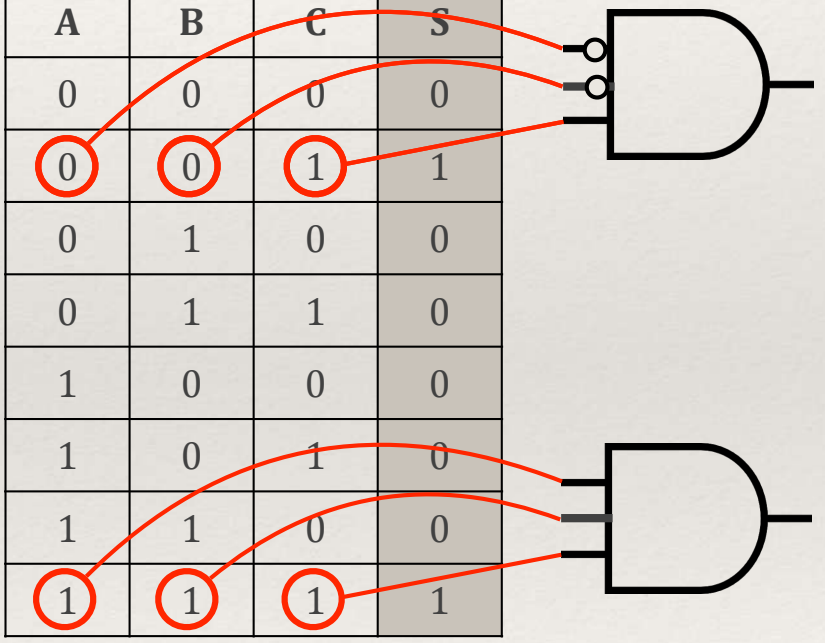
Nos dois exemplos da Lista de Exercícios 3, há apenas uma saída "diferente";

O resultado pode ser estendido

- Saídas 1 -> Portas AND;
- Saídas 0 -> Portas OR.

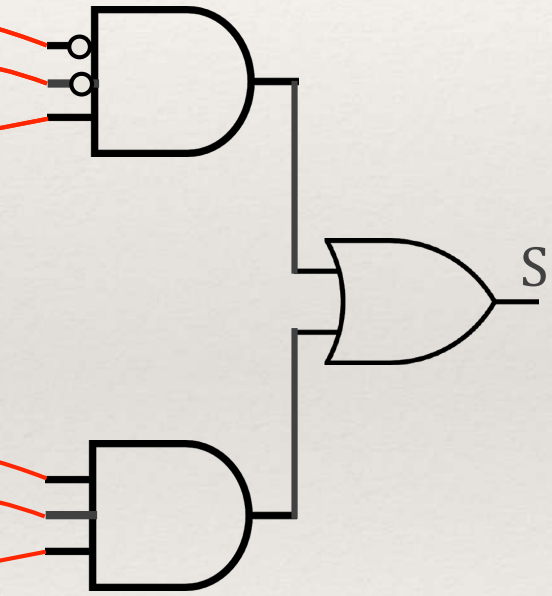
Estratégia correta é escolher a solução com menor número de portas.

A	B	C	S
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1



Síntese de Circuitos Lógicos I

A	B	C	S
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

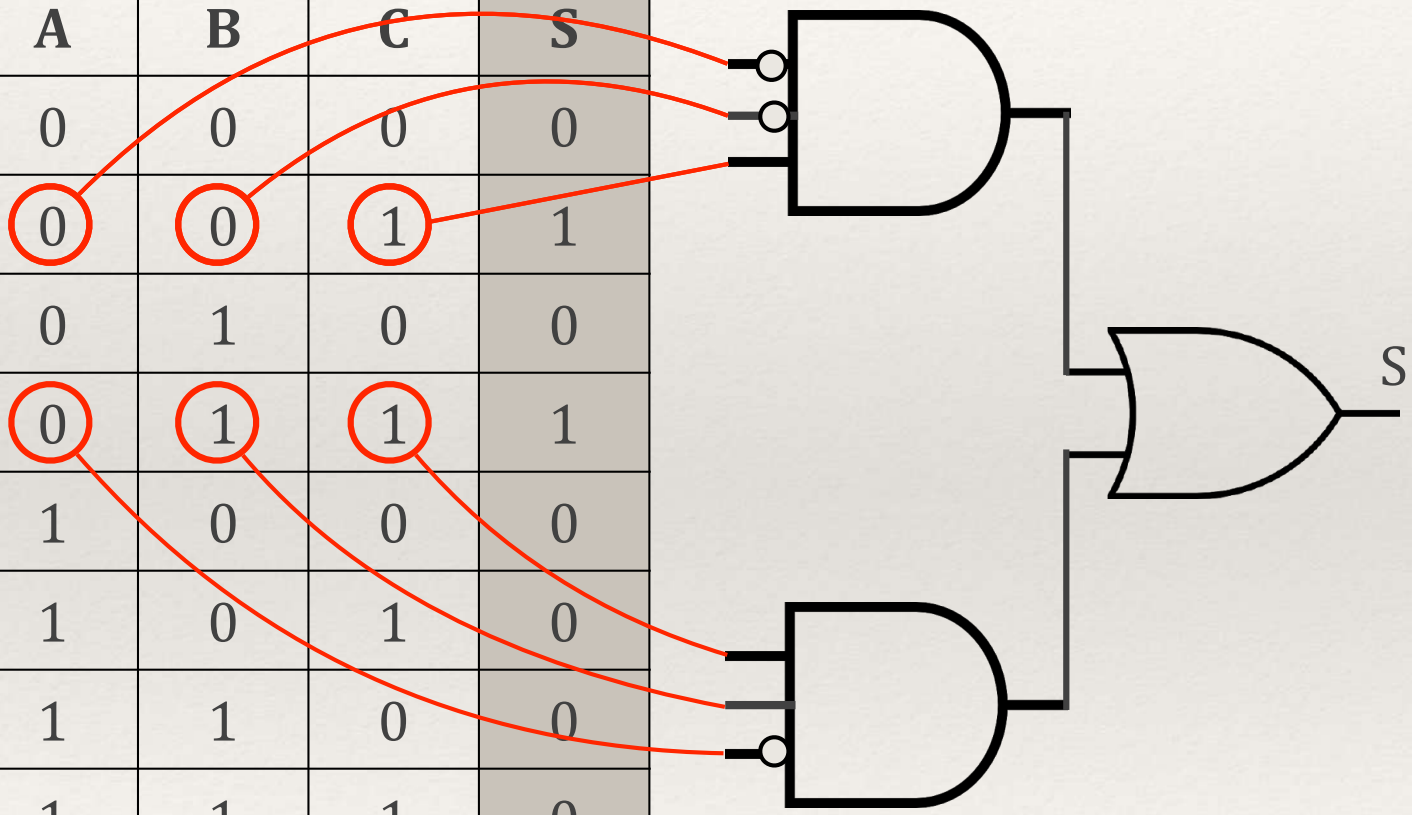


Exercício 3 da Lista 3

Síntese e simplificação de um circuito lógico com 3 entradas.

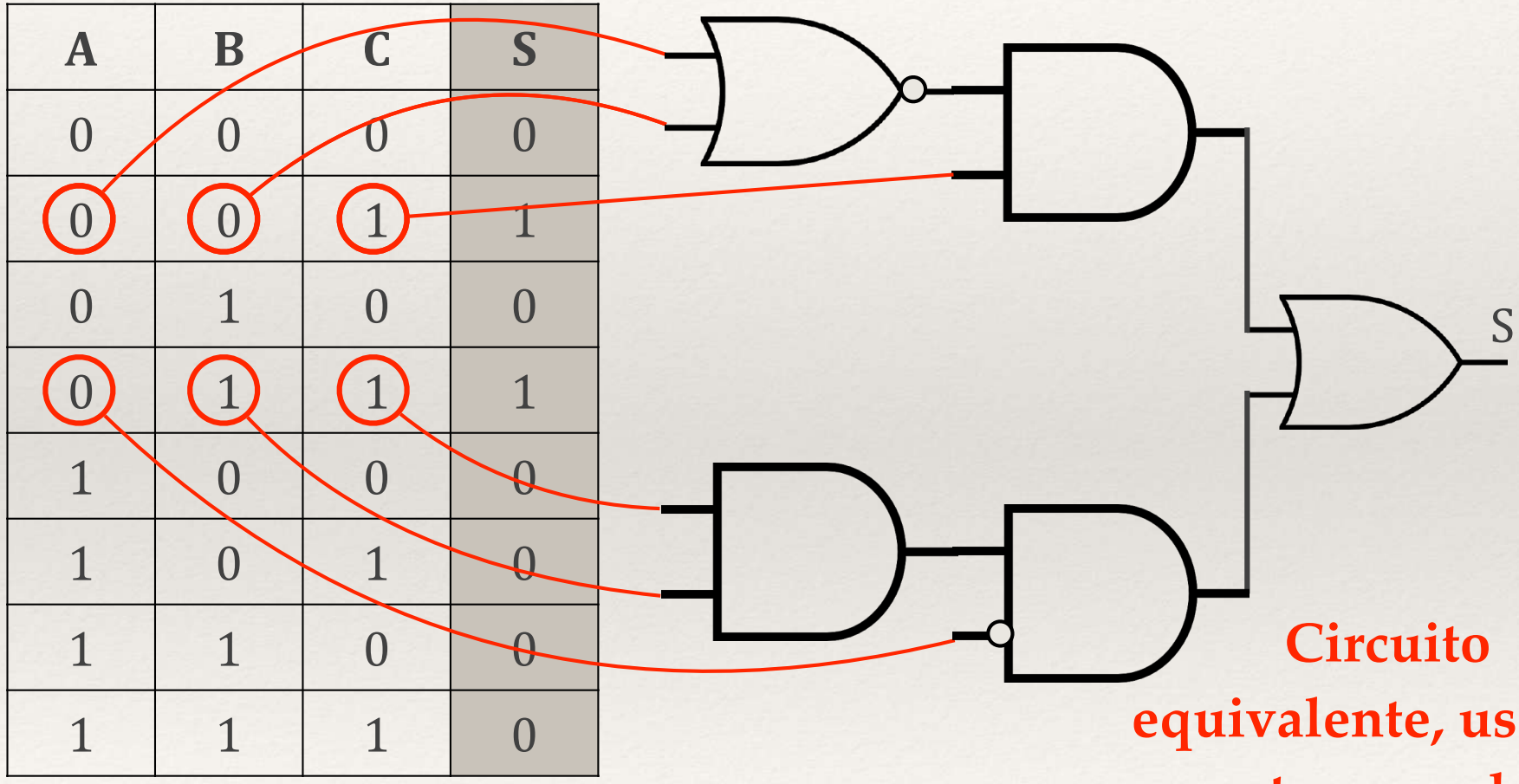
Síntese de Circuitos Lógicos II

A	B	C	S
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0



Síntese de Circuitos Lógicos II

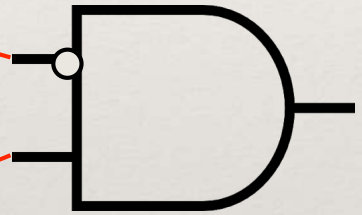
A	B	C	S
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0



Circuito equivalente, usando portas com duas entradas

Simplificação do Circuito

A	B	C	S
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

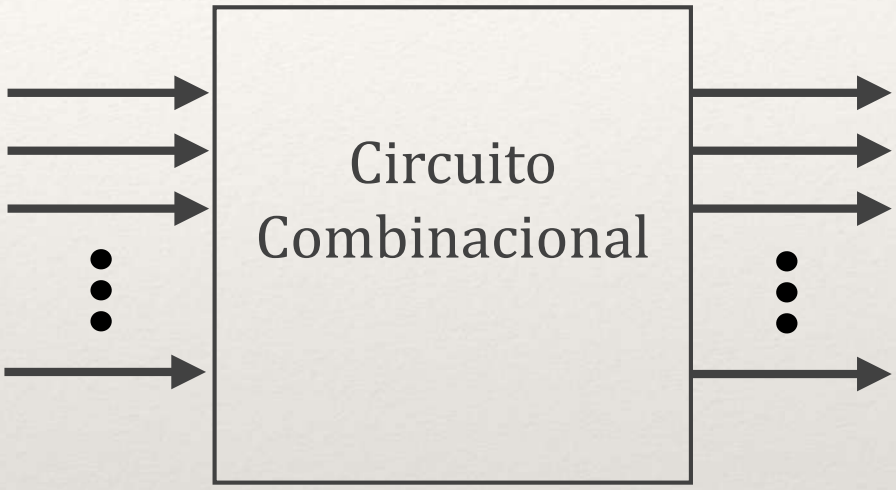


Circuitos Combinacionais

A partir do valor atual de “n" entradas determina-se o valor das “m" saídas;

O valor de cada uma das “m” saídas é determinada por uma expressão lógica;

Não existe realimentação (saídas não podem funcionar como entradas).

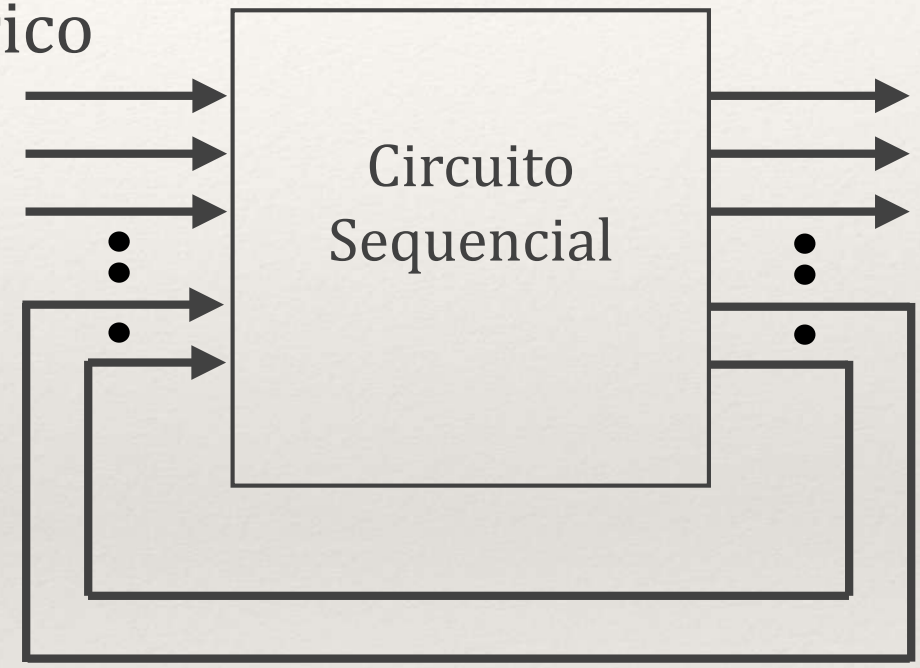


Circuitos Sequenciais

A partir do valor atual e histórico das “n" entradas pode ser determinado o valor de “m" saídas;

Uma ou parte das “m" saídas pode ser utilizada como entrada (realimentação);

Muitas operações computacionais exigem circuitos sequenciais. Ex: multiplicação, memória etc.



Circuitos Sequenciais

Dispositivos computacionais utilizam circuitos combinacionais e sequenciais;

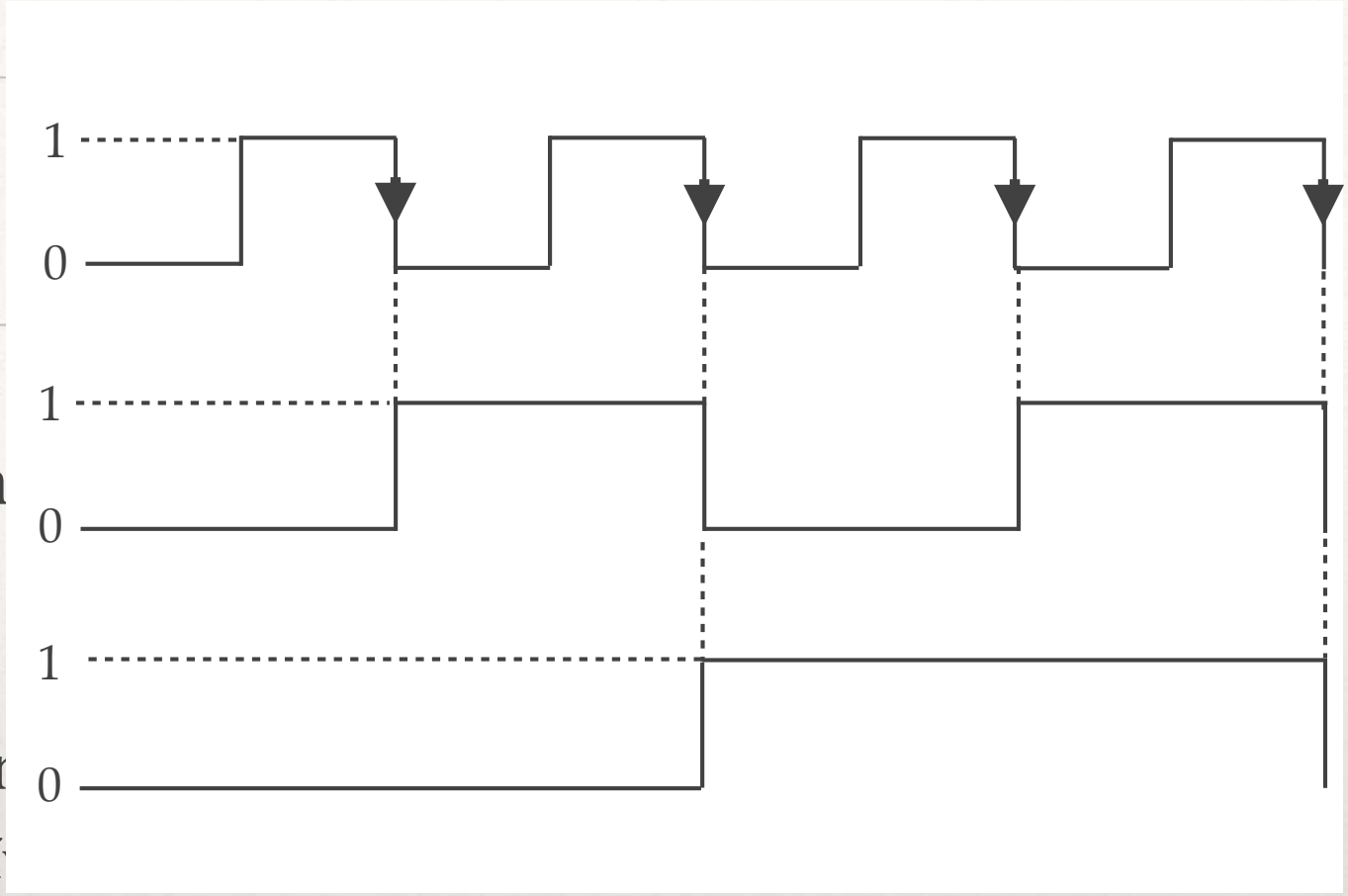
Circuitos sequenciais podem ser dispositivos de armazenamento (memórias), *latches* e *flip-flops*

- *Latches*: sensíveis ao nível lógico;
- *Flip-flops*: sensíveis à borda (subida ou descida).

Circuitos sequenciais compõem memórias estáticas, registradores de deslocamento, contadores etc.

O estudo detalhado dos circuitos sequenciais foge do escopo de nossa disciplina;

Circuitos



Dispositivos computa
e sequenciais;

Circuitos sequenciais
armazenamento (mer

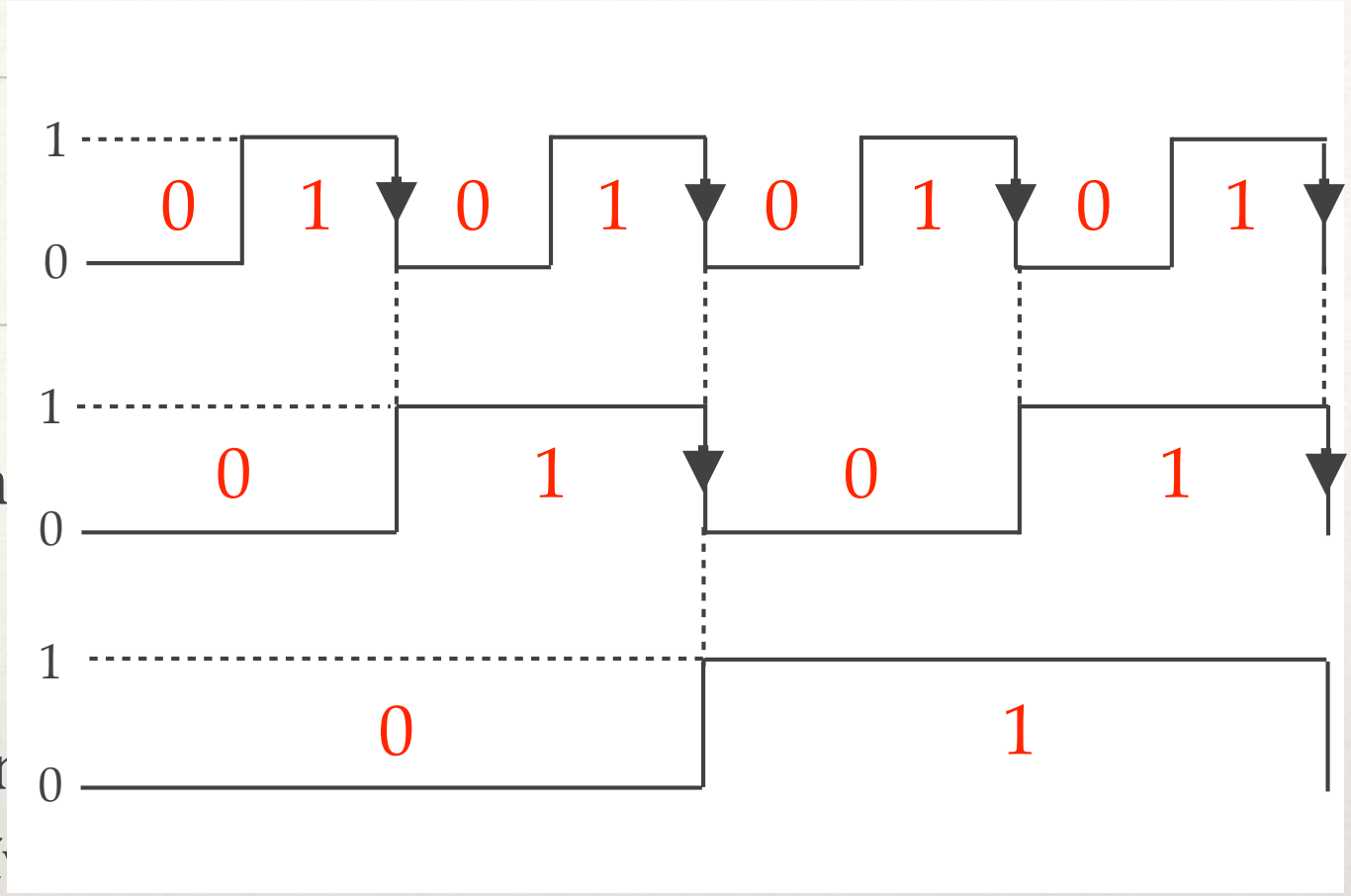
Latches: sensíveis ao ní

Flip-flops: sensíveis **à borda (subida ou descida)**.

Circuitos sequenciais compõem memórias estáticas, registradores de deslocamento, contadores etc.

O estudo detalhado dos circuitos sequenciais foge do escopo de nossa disciplina;

Circuitos



Dispositivos computa
e sequenciais;

Circuitos sequenciais
armazenamento (mer

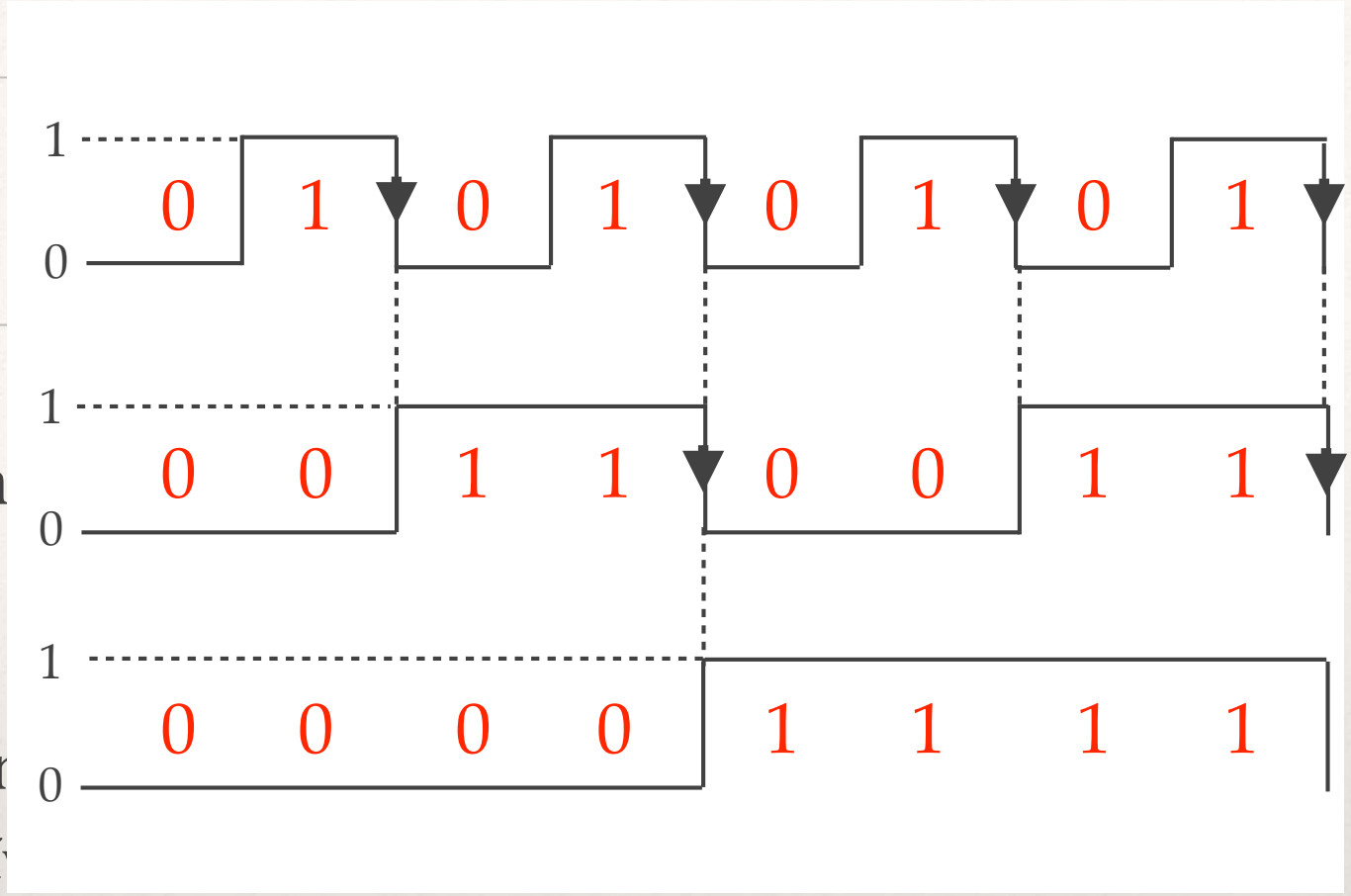
Latches: sensíveis ao ní

Flip-flops: sensíveis **à borda (subida ou descida)**.

Circuitos sequenciais compõem memórias estáticas, registradores de deslocamento, contadores etc.

O estudo detalhado dos circuitos sequenciais foge do escopo de nossa disciplina;

Circuitos



Dispositivos computa
e sequenciais;

Circuitos sequenciais
armazenamento (mer

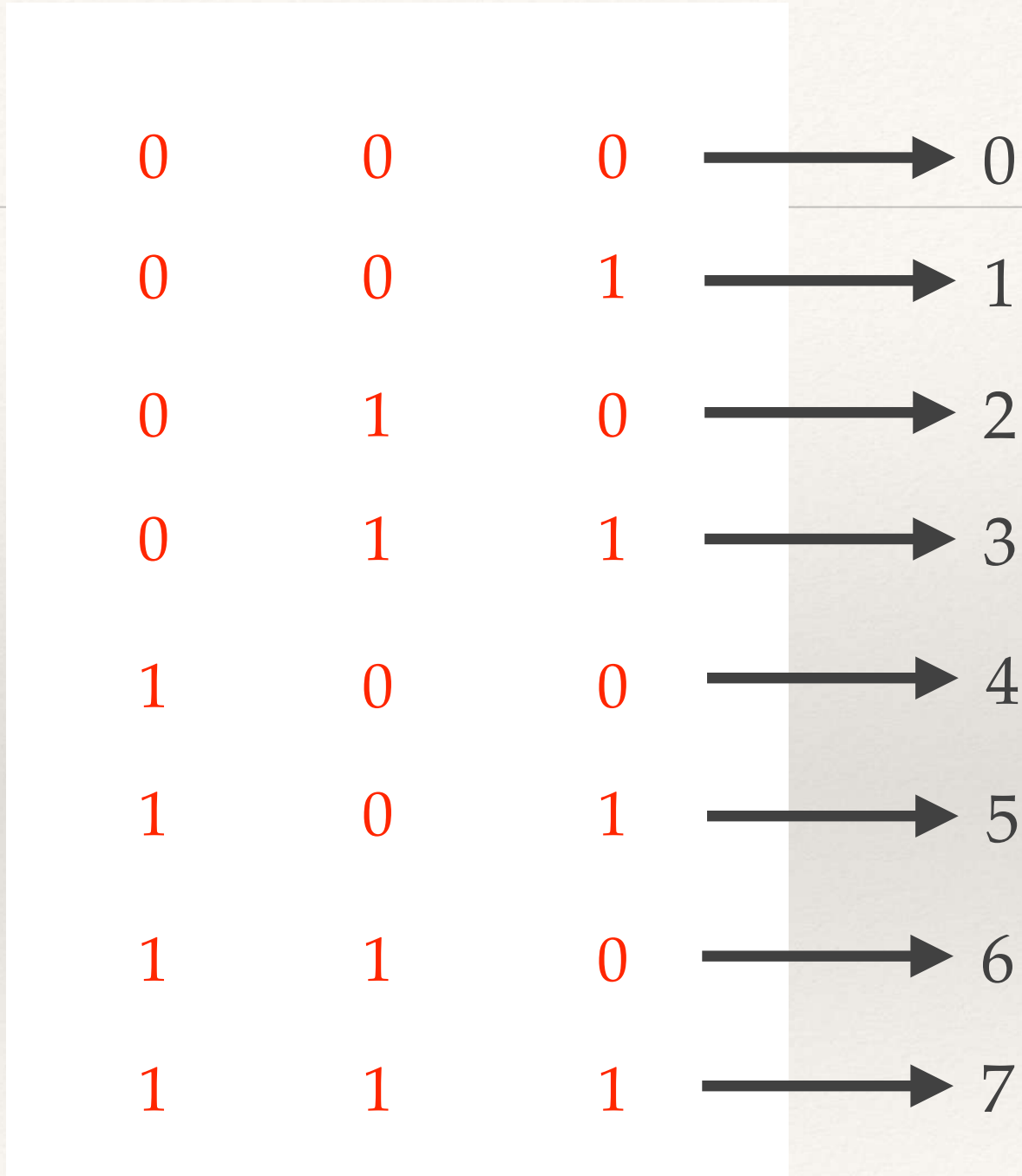
Latches: sensíveis ao ní

Flip-flops: sensíveis **à borda (subida ou descida)**.

Circuitos sequenciais compõem memórias estáticas, registradores de deslocamento, contadores etc.

O estudo detalhado dos circuitos sequenciais foge do escopo de nossa disciplina;

0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1


0 0 0 0
0 0 1 1
0 1 0 2
0 1 1 3
1 0 0 4
1 0 1 5
1 1 0 6
1 1 1 7

Estrutura de um Computador

O ENIAC (*Electronic Numerical Integrator and Computer*) foi o primeiro computador de uso geral;

- Era decimal !
- Sua programação era feita literalmente de forma manual (chaves e cabos)

O IAS (Institute of Advanced Study), desenvolvido por John von Neumann (1946-1951) foi a primeira máquina com programas armazenados em memória.

- Seu projeto é base da maioria dos computadores em uso até hoje!

Estrutura de um Computador

IAS (von Neumann)

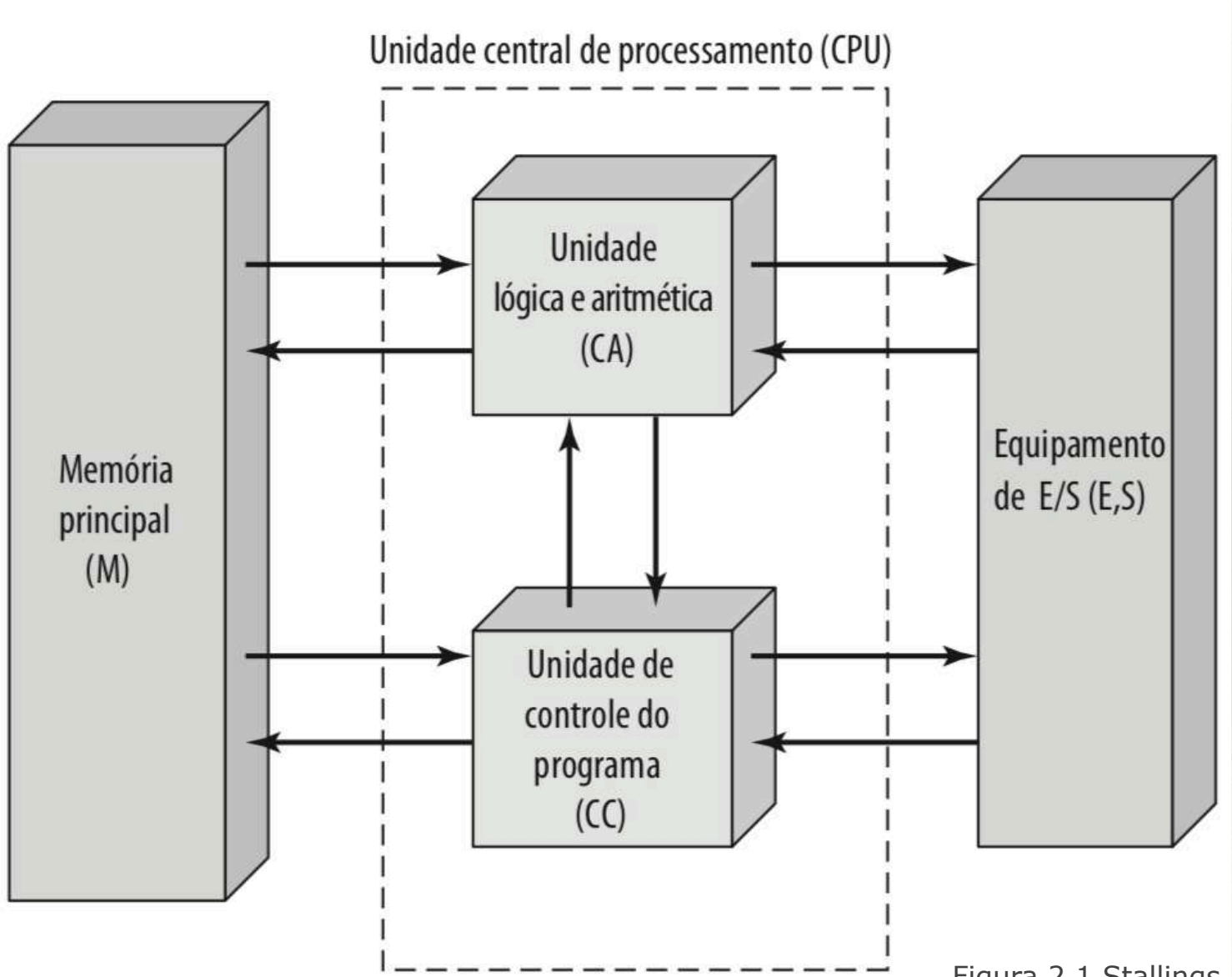


Figura 2.1 Stallings

O IAS

1.000 posições de memória (palavras) com 40 bits;

Dados e instruções armazenados na memória em binário;

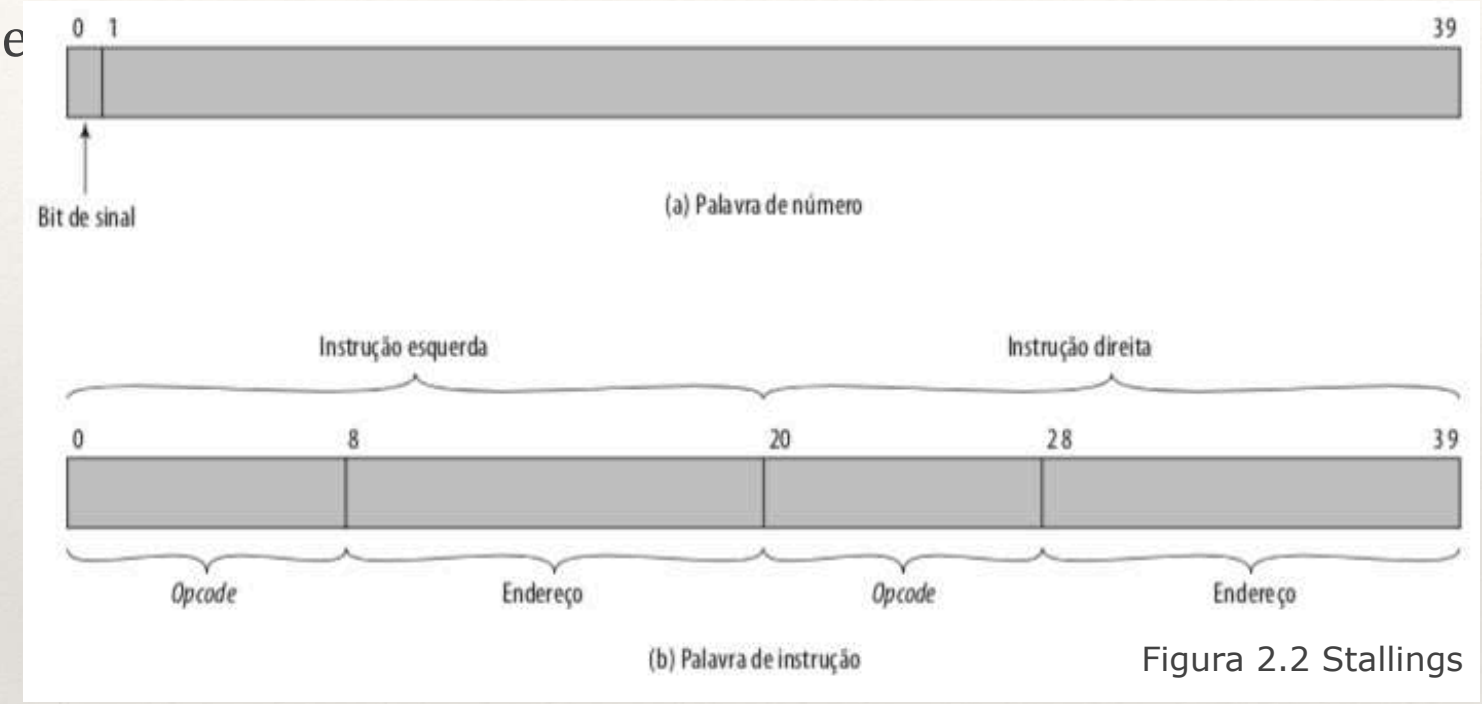


Figura 2.2 Stallings

Cada número é representado por um bit de sinal e um valor de 39 bits;

Uma palavra também pode conter duas instruções de 20 bits (8 bits com o código de operação - *opcode*) e um endereço de 12 bits designando uma das palavras na memória (numeradas de 0 a 999).

Estrutura de Alto Nível

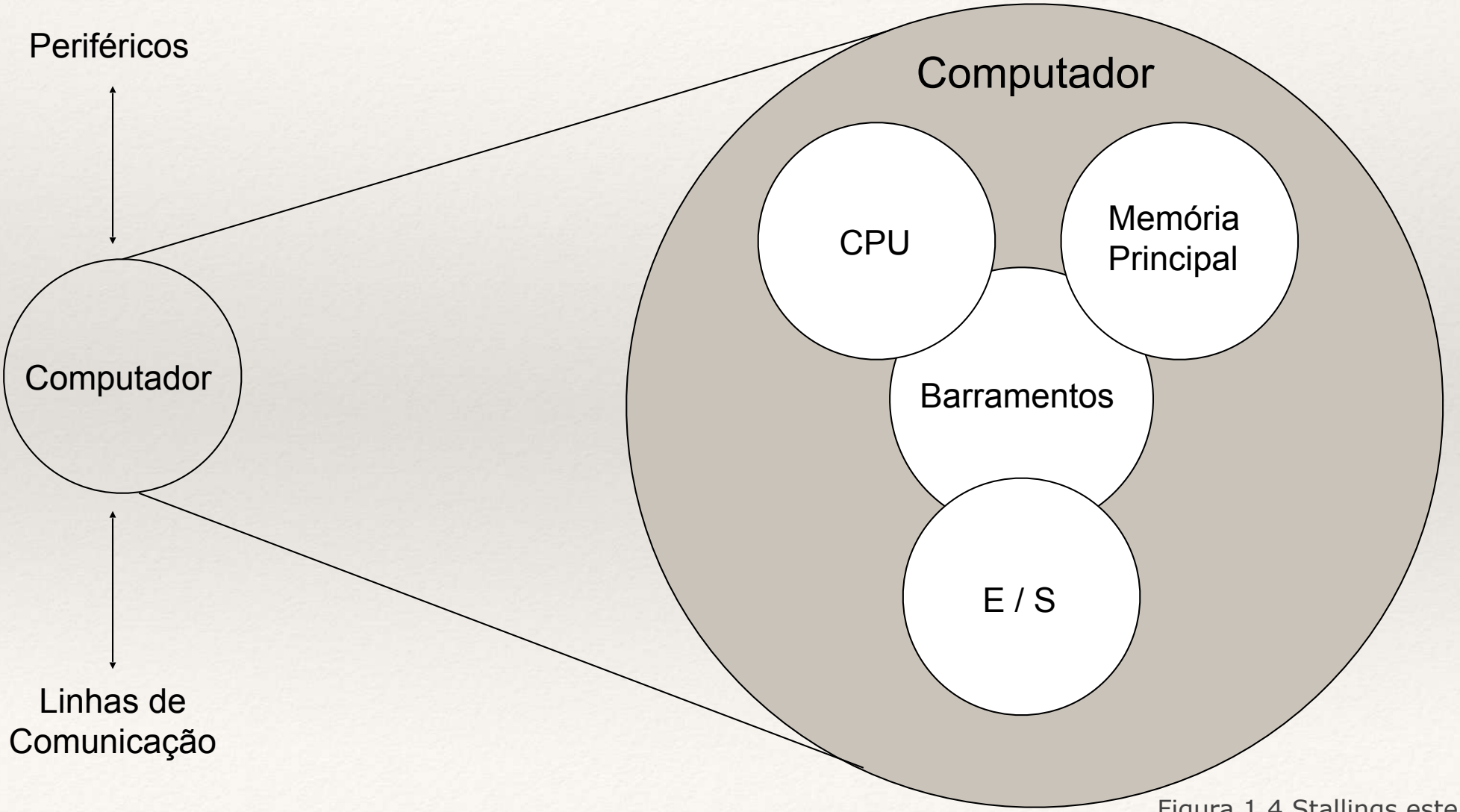


Figura 1.4 Stallings estendida

Estrutura da CPU

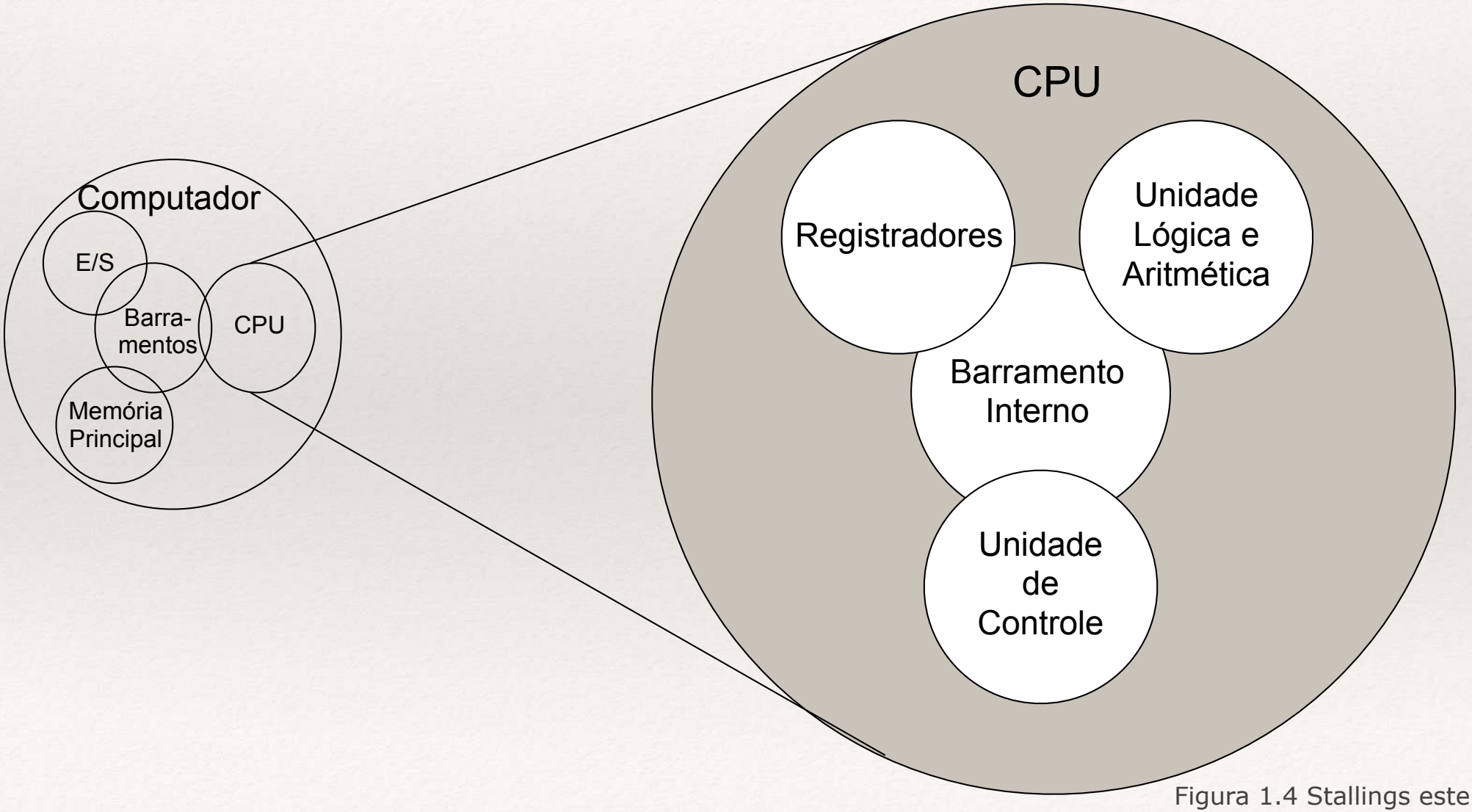


Figura 1.4 Stallings estendida

Estrutura da Unidade de Controle

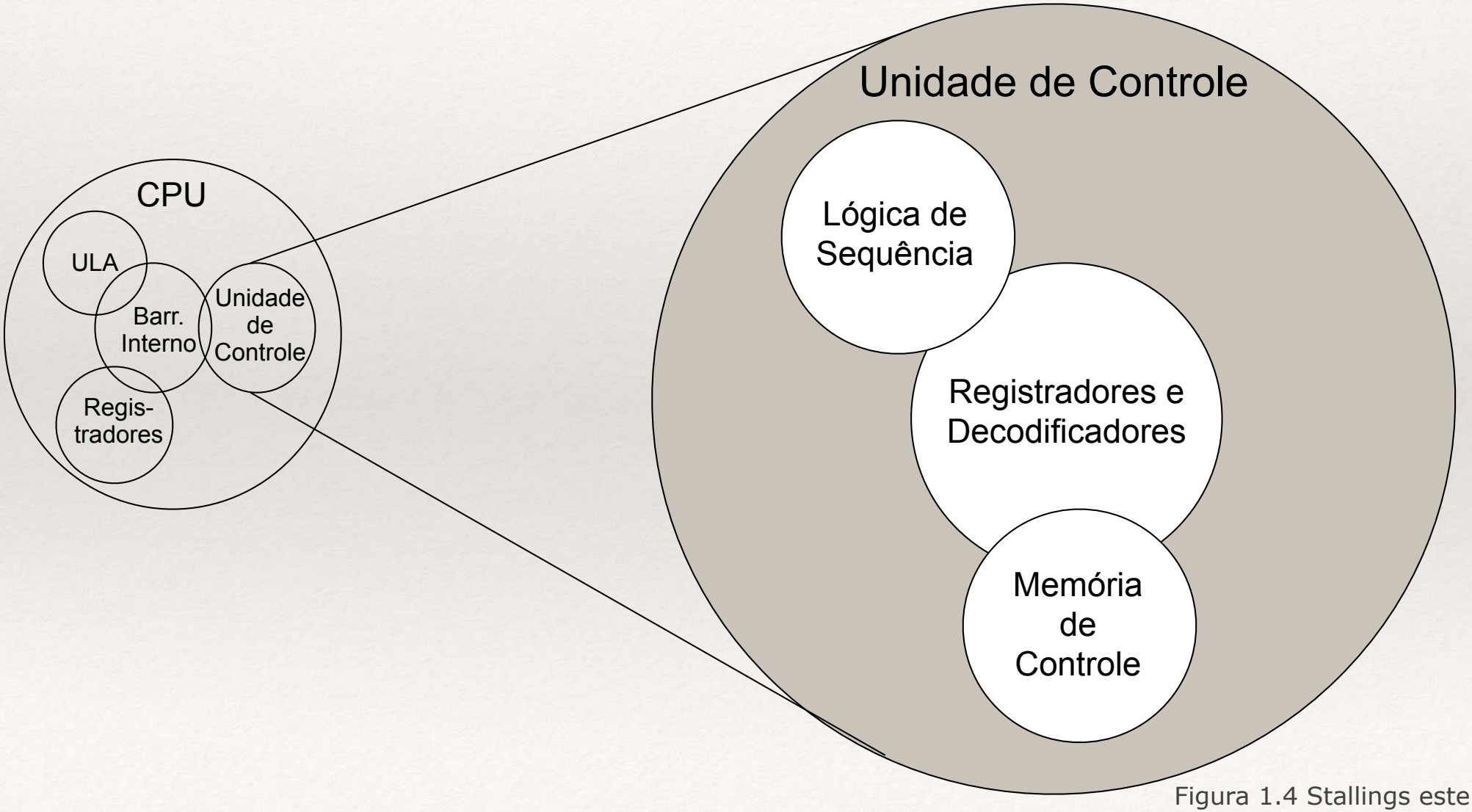


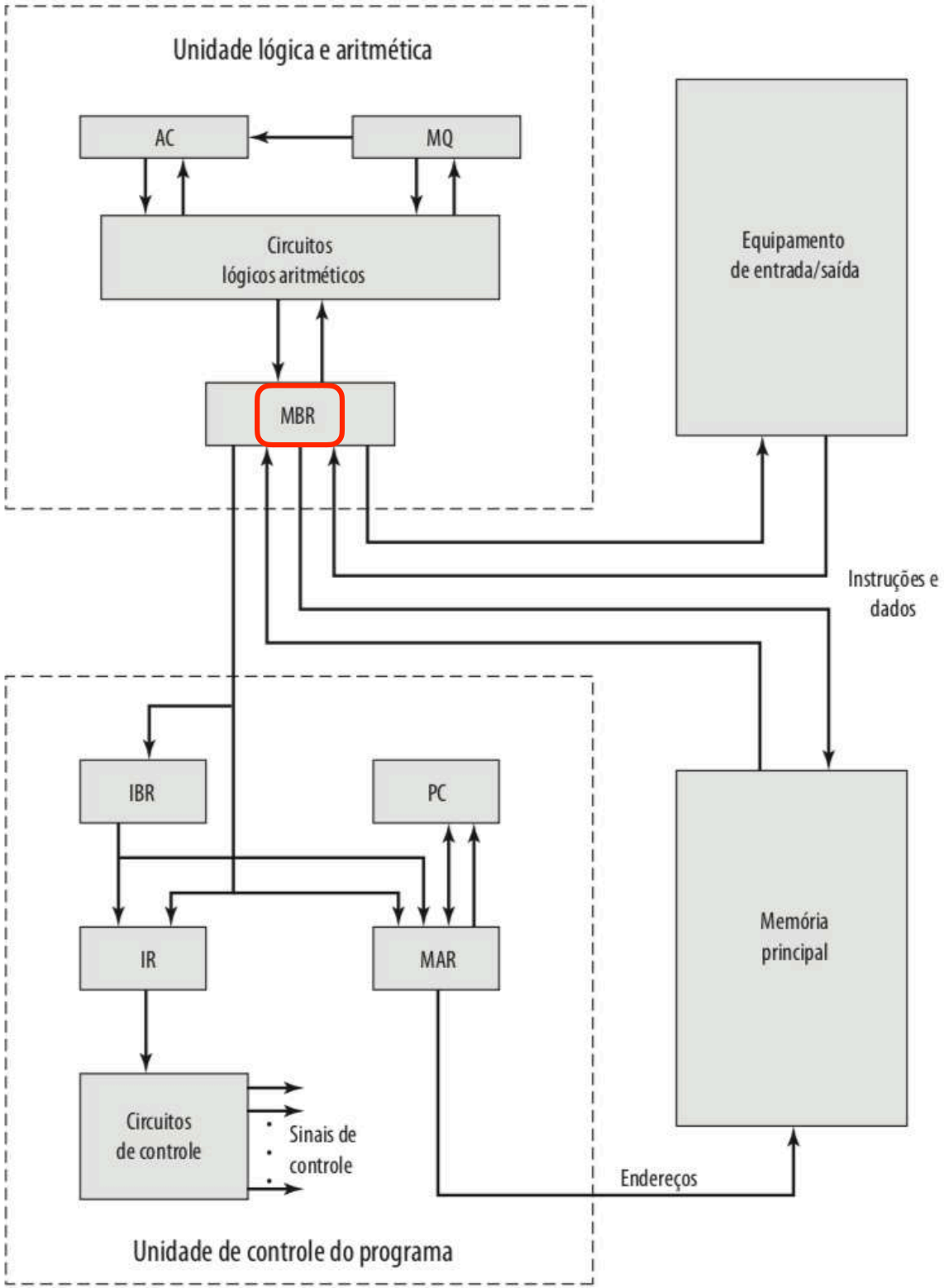
Figura 1.4 Stallings estendida

Operando o IAS

MBR

Memory Buffer Register

Contém uma palavra a ser armazenada na memória ou enviada para um dispositivo de E/S.

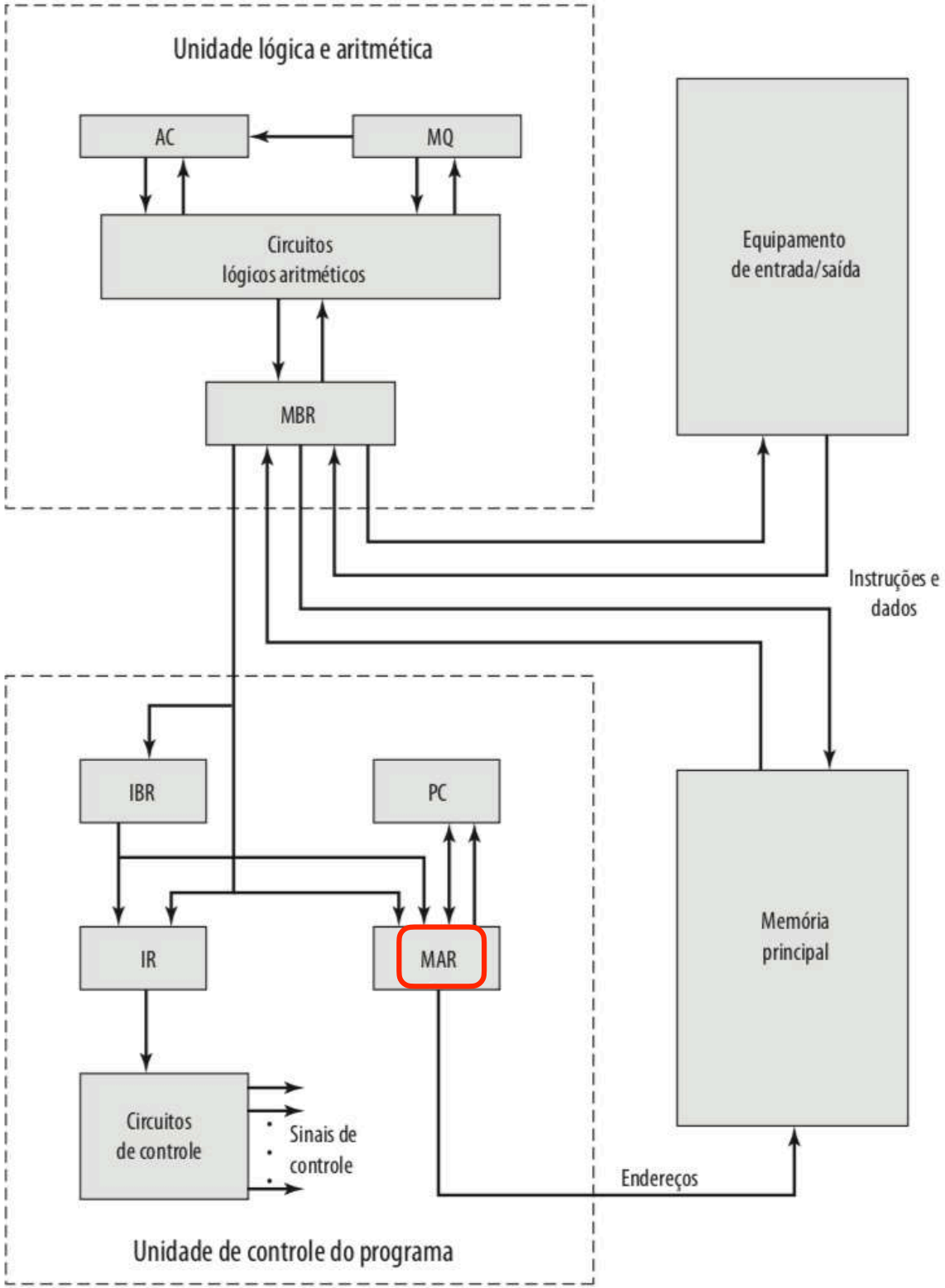


Operando o IAS

MAR

Memory Address Register

Especifica o endereço, na memória, da palavra a ser escrita ou lida no MBR.

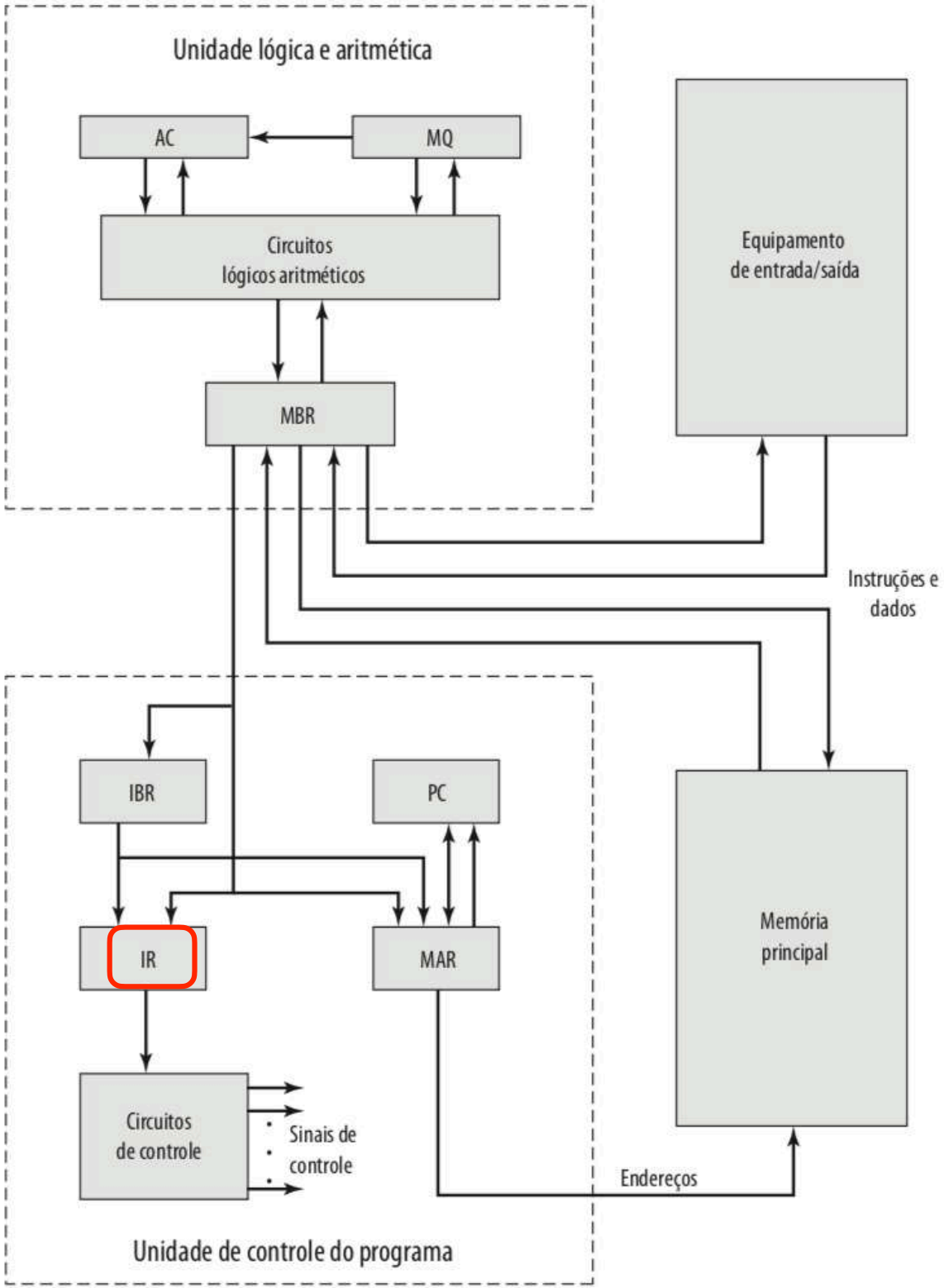


Operando o IAS

IR

Instruction Register

Contém o *opcode* de 8 bits da instrução que está sendo executada.

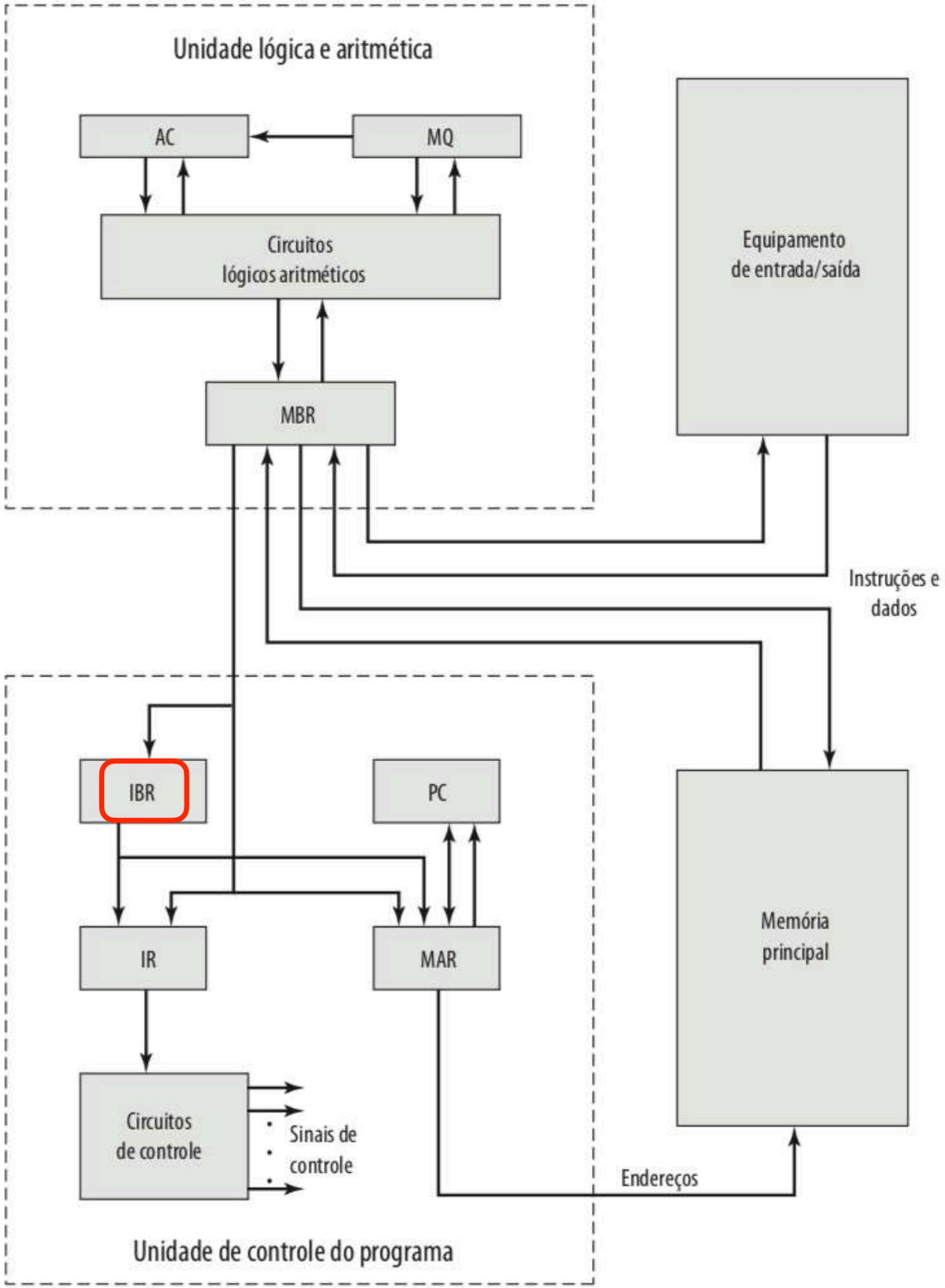


Operando o IAS

IBR

Instruction Buffer Register

Mantém a próxima instrução a ser executada.

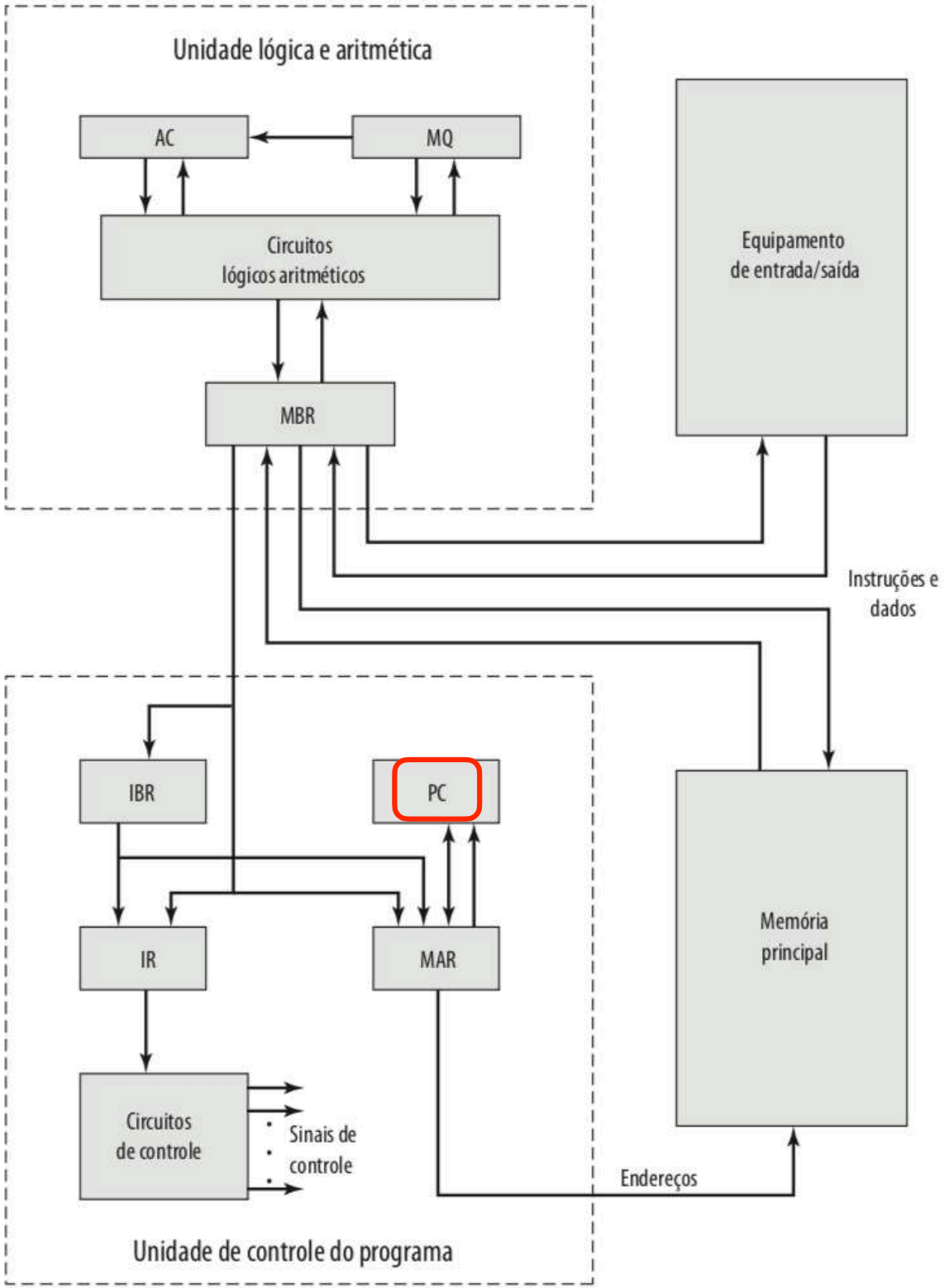


Operando o IAS

PC

Program Counter

Contém o endereço do próximo par de instruções a ser lido na memória.

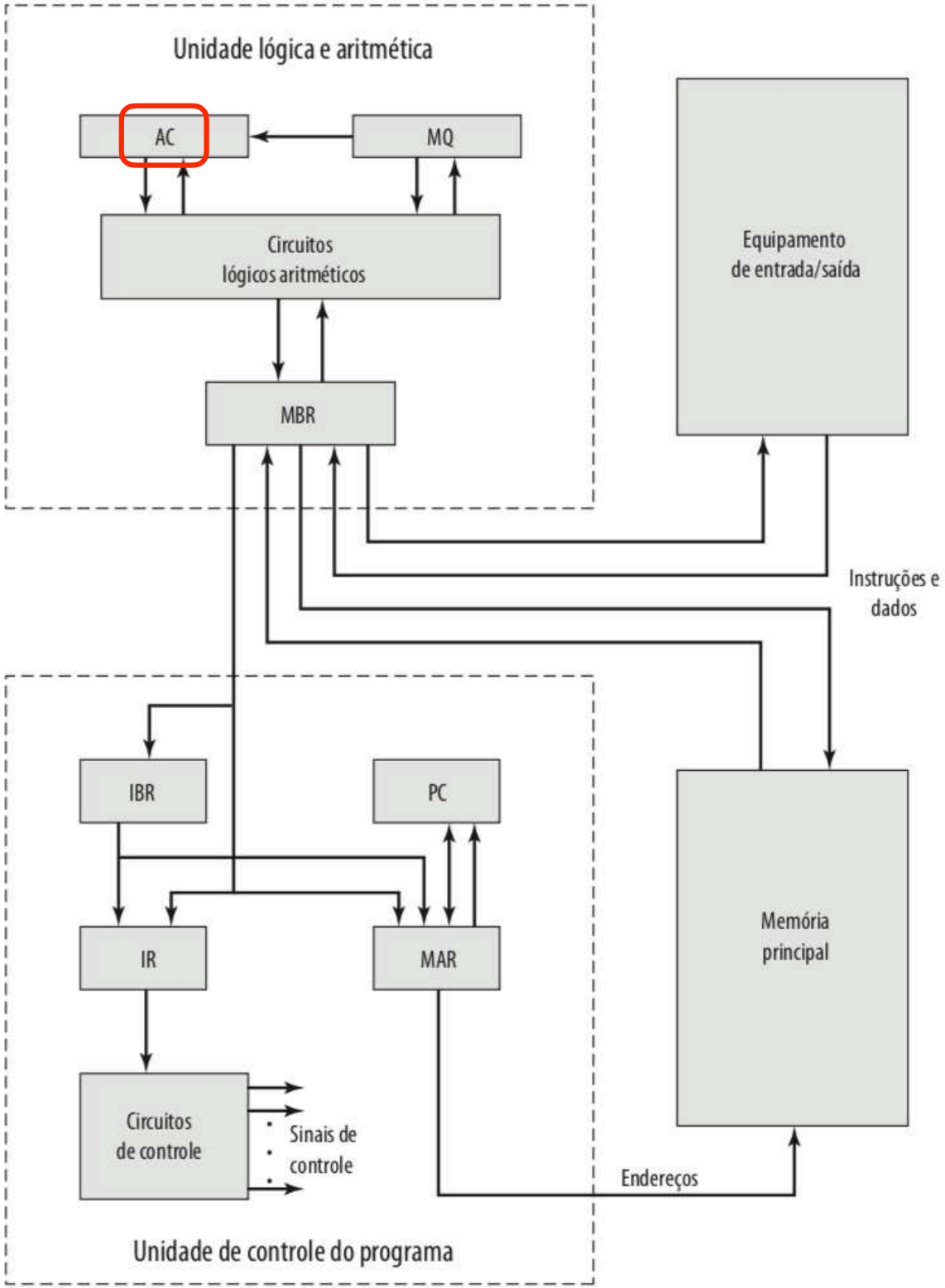


Operando o IAS

AC

Acumulador

Mantém operandos temporariamente.



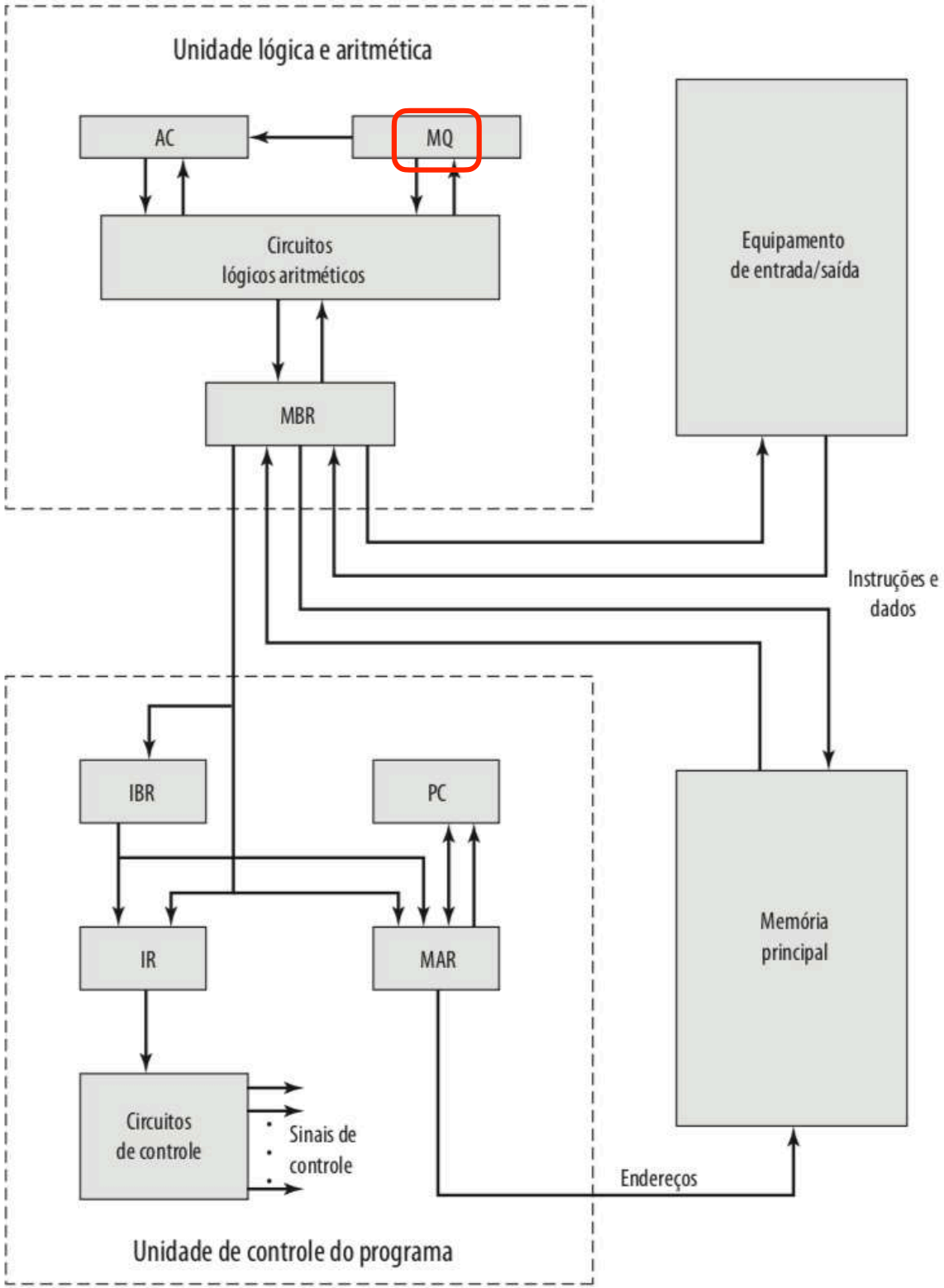
Operando o IAS

MQ

Multiplier Quotient

Mantém operandos e também resultados temporários.

Ex.: uma multiplicação de dois números de 40 bits é um número de 80 bits.



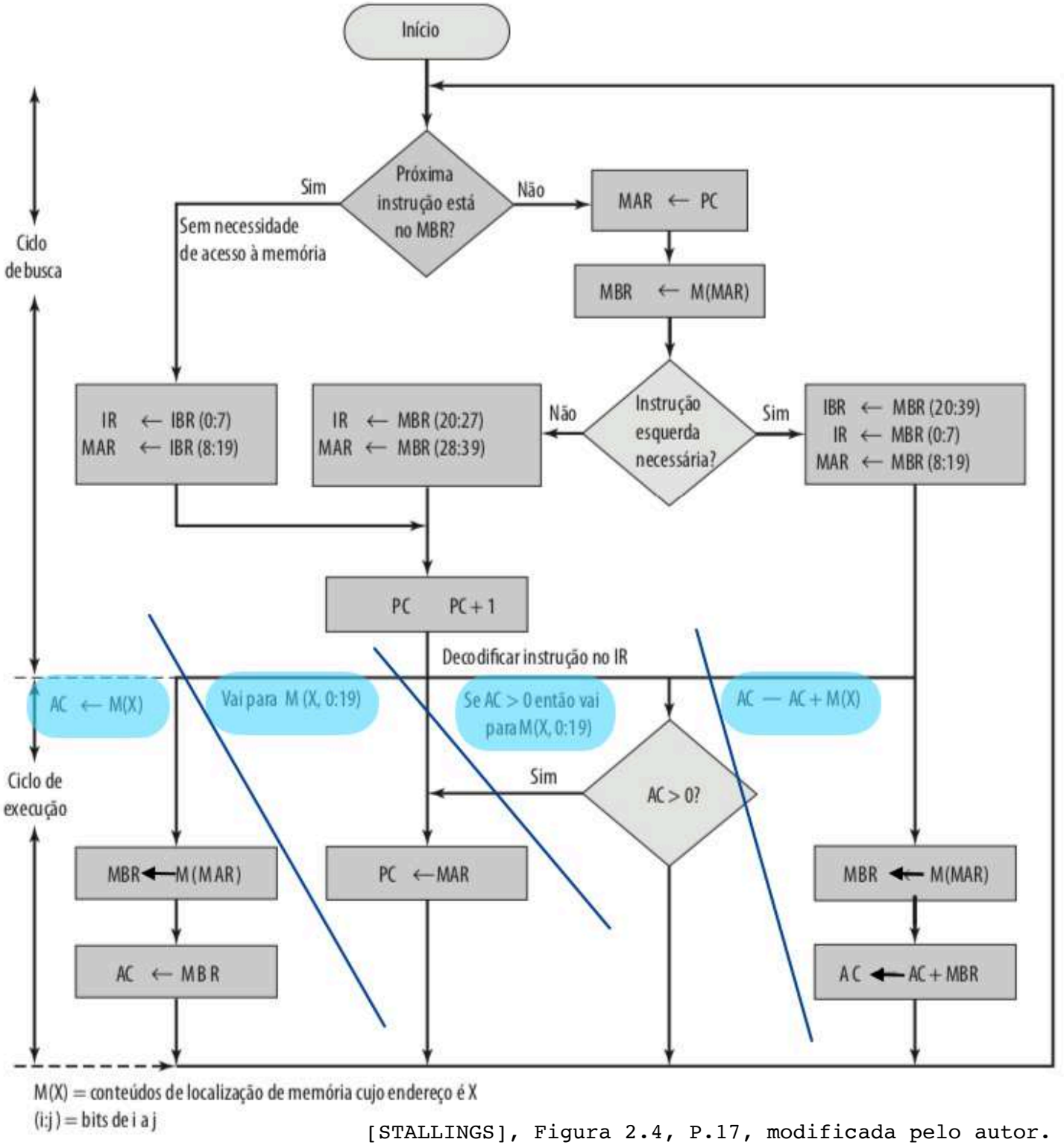
Ciclo de Instrução no IAS

Sub-ciclos

- Busca (*Fetch*)
- Execução

4 Instruções

- AC ← M(x)
- GOTO M(x esq.)
- IF(AC>0), GOTO M(x esq.)
- AC ← AC + M(x)



[STALLINGS], Figura 2.4, P.17, modificada pelo autor.

Conjunto de Instruções

5 grupos;

21 instruções;

Assembly.

Tipo de instrução	*Opcode*	Representação simbólica	Descrição
Transferência de dados	00001010	LOAD MQ	Transfere o conteúdo de MQ para AC
	00001001	LOAD MQ,M(X)	Transfere o conteúdo do local de memória X para MQ
	00100001	STOR M(X)	Transfere o conteúdo de AC para o local de memória X
	00000001	LOAD M(X)	Transfere M(X) para o AC
	00000010	LOAD – M(X)	Transfere – M(X) para o AC
	00000011	LOAD |M(X)|	Transfere o valor absoluto de M(X) para o AC
	00000100	LOAD – |M(X)|	Transfere -|M(X)| para o acumulador
Desvio incondicional	00001101	JUMP M(X,0:19)	Apanha a próxima instrução da metade esquerda de M(X)
	00001110	JUMP M(X,20:39)	Apanha a próxima instrução da metade direita de M(X)
Desvio condicional	00001111	JUMP+ M(X,0:19)	Se o número no AC for não negativo, apanha a próxima instrução da metade esquerda de M(X)
	00010000	JUMP+ M(X,20:39)	Se o número no AC for não negativo, apanha a próxima instrução da metade direita de M(X)
Aritmética	00000101	ADD M(X)	Soma M(X) a AC; coloca o resultado em AC
	00000111	ADD |M(X)|	Soma |M(X)| a AC; coloca o resultado em AC
	00000110	SUB M(X)	Subtrai M(X) de AC; coloca o resultado em AC
	00001000	SUB |M(X)|	Subtrai |M(X)| de AC; coloca o resto em AC
	00001011	MUL M(X)	Multiplica M(X) por MQ; coloca os bits mais significativos do resultado em AC; coloca bits menos significativos em MQ
	00001100	DIV M(X)	Divide AC por M(X); coloca o quociente em MQ e o resto em AC
	00010100	LSH	Multiplica o AC por 2; ou seja, desloca à esquerda uma posição de bit
	00010101	RSH	Divide o AC por 2; ou seja, desloca uma posição à direita
Modificação de endereço	00010010	STOR M(X,8:19)	Substitui campo de endereço da esquerda em M(X) por 12 bits mais à direita de AC
	00010011	STOR M(X,28:39)	Substitui campo de endereço da direita em M(X) por 12 bits mais à direita de AC

[STALLINGS, William. Arquitetura e Org.Computadores – 8ªEd. Tabela 2.1, p.18

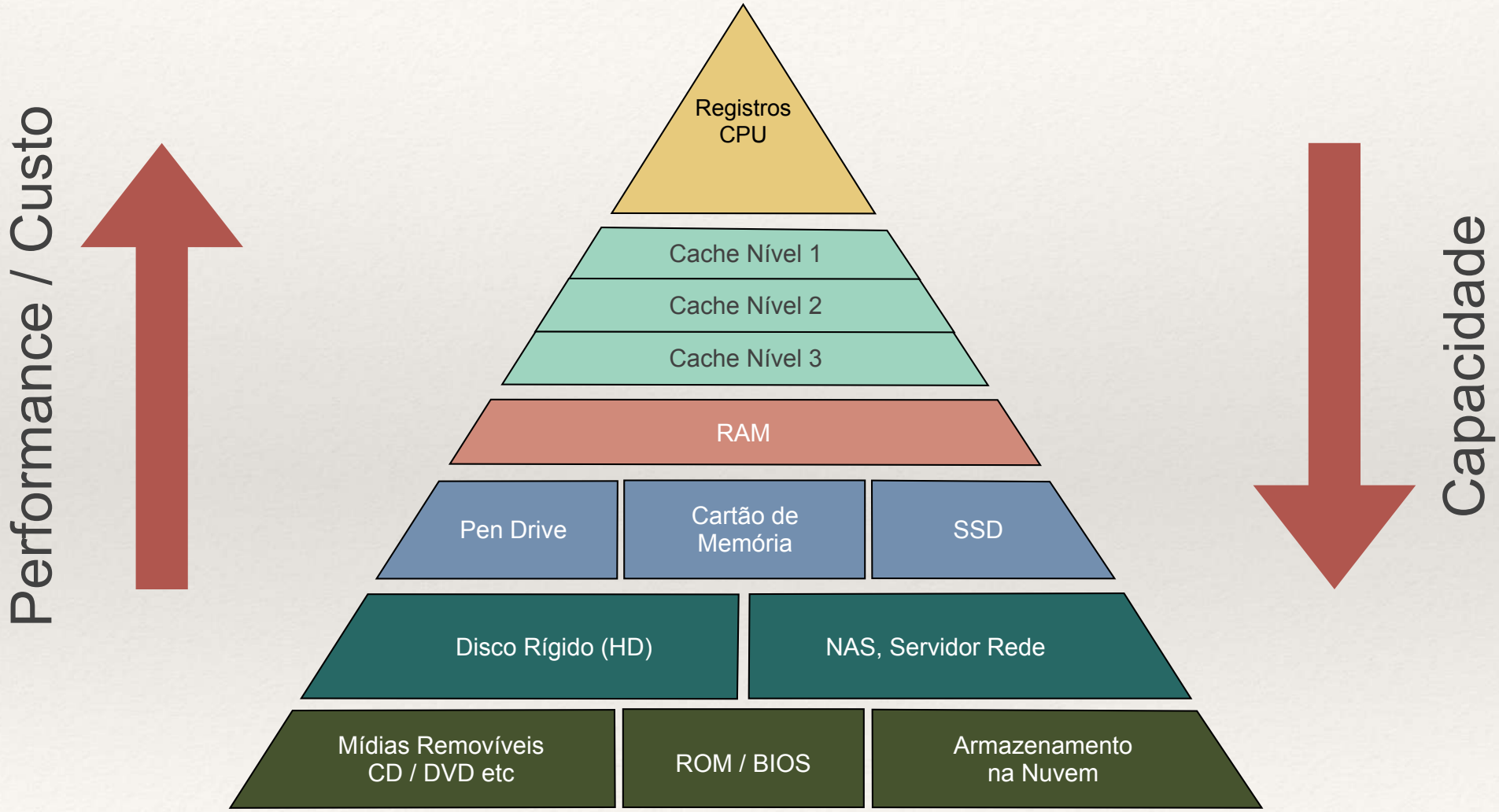
Memória - Características

Tabela 4.1 Principais características dos sistemas de memória do computador

Localização	**Desempenho**
Interna (por exemplo, registradores do processador, memória principal, cache)	Tempo de acesso
	Tempo de ciclo
Externa (por exemplo, discos ópticos, discos magnéticos, fitas)	Taxa de transferência
Capacidade	**Tipo físico**
Número de palavras	Semicondutor
Número de bytes	Magnético
Unidade de transferência	Óptico
Palavra	Magneto-óptico
Bloco	**Características físicas**
Método de acesso	Volátil/não volátil
Sequencial	Apagável/não apagável
Direto	**Organização**
Aleatório	Módulos de memória
Associativo	

[STALLINGS, William. Arquitetura e Org.Computadores - 8ªEd. Tabela 4.1, p.90

Hierarquia de Memória



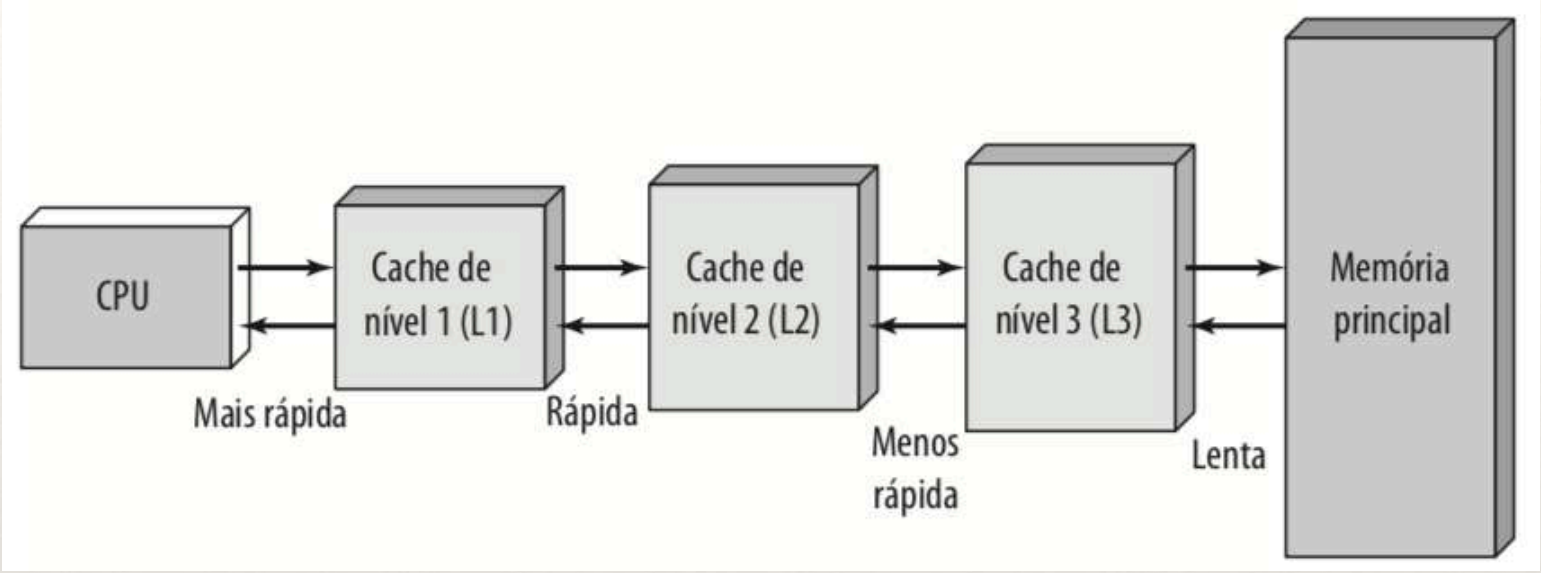
Fonte: Autor

Comparativo de Performance

Nível	1	2	3	4	5
Nome	registradores	cache	memória principal	disco de estado sólido	disco magnético
Tamanho típico	< 1 KB	< 16 MB	< 64 GB	< 1 TB	< 10 TB
Tecnologia de implementação	memória personalizada com várias portas CMOS	CMOS SRAM em chip ou fora do chip	CMOS SRAM	memória flash	disco magnético
Tempo de acesso (ns)	0,25 – 0,5	0,5 – 25	80 – 250	25.000 – 50.000	5.000.000
Largura de banda (MB/s)	20000 – 100000	5000 – 10000	1000 – 5000	500	20 – 150

[SILBERSCHATZ, Abraham] Fundamentos de Sistemas Operacionais, Figura 1.11, Cap.1

A Memória *Cache*



[STALLINGS, William. Arquitetura e Org.Computadores - 8ªEd. Figura 4.3, p.96

O objetivo é acelerar o acesso às instruções e dados, evitando o acesso à memória principal;

A memória *cache* é constituída de níveis, sendo que os mais próximos da CPU são normalmente mais rápidos, e menores;

Com a evolução dos processadores, o número de níveis cresceu.

Processor	Type	Year of Introduction	L1 Cache[a]	L2 Cache	L3 Cache
IBM 360/85	Mainframe	1968	16–32 kB	—	—
PDP-11/70	Minicomputer	1975	1 kB	—	—
VAX 11/780	Minicomputer	1978	16 kB	—	—
IBM 3033	Mainframe	1978	64 kB	—	—
IBM 3090	Mainframe	1985	128–256 kB	—	—
Intel 80486	PC	1989	8 kB	—	—
Pentium	PC	1993	8 kB/8 kB	256–512 kB	—
PowerPC 601	PC	1993	32 kB	—	—
PowerPC 620	PC	1996	32 kB/32 kB	—	—
PowerPC G4	PC/server	1999	32 kB/32 kB	256 kB to 1 MB	2 MB
IBM S/390 G6	Mainframe	1999	256 kB	8 MB	—
Pentium 4	PC/server	2000	8 kB/8 kB	256 kB	—
IBM SP	High-end server/ supercomputer	2000	64 kB/32 kB	8 MB	—
CRAY MTA[b]	Supercomputer	2000	8 kB	2 MB	—
Itanium	PC/server	2001	16 kB/16 kB	96 kB	4 MB
Itanium 2	PC/server	2002	32 kB	256 kB	6 MB
IBM POWER5	High-end server	2003	64 kB	1.9 MB	36 MB
CRAY XD-1	Supercomputer	2004	64 kB/64 kB	1 MB	—
IBM POWER6	PC/server	2007	64 kB/64 kB	4 MB	32 MB
IBM z10	Mainframe	2008	64 kB/128 kB	3 MB	24–48 MB
Intel Core i7 EE 990	Workstation/ server	2011	6 × 32 kB/ 32 kB	1.5 MB	12 MB
IBM zEnterprise 196	Mainframe/ server	2011	24 × 64 kB/ 128 kB	24 × 1.5 MB	24 MB L3 192 MB L4

[STALLINGS, William. Arquitetura e Org.Computadores - 8ªEd. Tabela 4.3, p.100

Política de escrita no *cache*

Como a memória *cache* é menor que a memória principal, blocos da memória *cache* são frequentemente substituídos;

Blocos apenas “lidos" não causam conflitos, mas ...

Como resolver a atualização de blocos alterados?

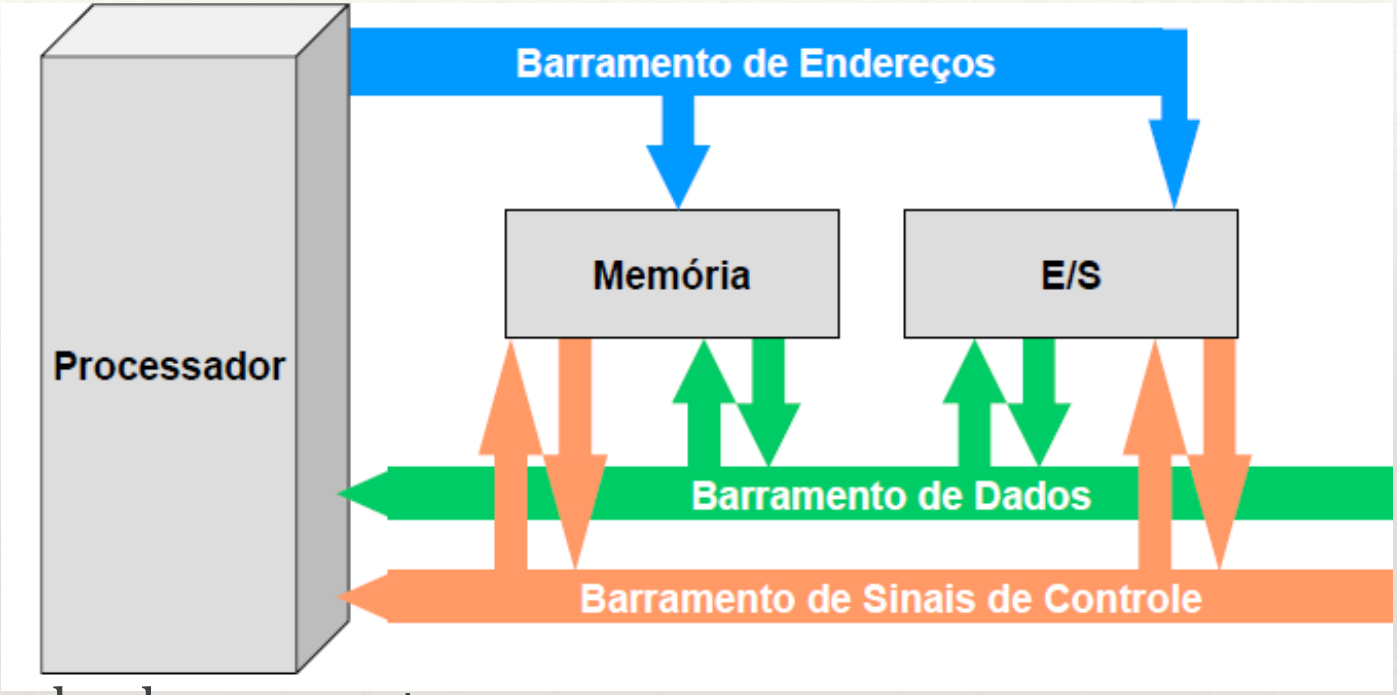
- No *cache*? E na memória principal?
- Compartilhamento por periféricos? E por múltiplos processadores, cada qual com seu *cache*?

Barramentos

Os componentes trocam informações;

São 3 os barramentos de um computador:

- Dados
- Endereços
- Controle



As setas indicam o fluxo de informações pelos barramentos:

- A Memória pode ser escrita ou lida (barramento de dados);
- O mesmo acontece com os periféricos de Entrada/Saída;
- A CPU apenas escreve no barramento de endereços, mas lê e escreve no barramento de dados;
- Todos escrevem e lêem no barramento de controle.

Máquinas Paralelas

Além da máquina de Von Neumann, outras arquiteturas são discutidas, e algumas vezes implementadas;

Muitas vezes comparam o cérebro humano com computadores. Estes últimos são mais rápidos em muitas coisas, mas não superaram a capacidade humana

- A resposta seria o paralelismo? (temos bilhões de neurônios operando em paralelo)
- Arquiteturas paralelas com muitas CPUs simples não poderiam ser superiores?

Processamento Paralelo exige:

- Sincronismo;
- Balanceamento de carga (equalitária ou especializada).

O desafio é de *hardware*, ou de *software*?

Máquinas Paralelas

Projetadas para executar múltiplas tarefas independentes simultaneamente:

- Servidores na Internet (Ex. serviços da Google);
- Transações Bancárias.

Processamento vetorial:

- Soma de dois vetores: cada processador trata de um par de elementos;

Pipeline (divisão de tarefas)

Taxonomia de Flynn

Instruções \ Fluxo de Dados	Único	Múltiplo
Único	SISD Single Instruction, Single Data	SIMD Single Instruction, Multiple Data
Múltiplo	MISD Multiple Instruction, Single Data	MIMD Multiple Instruction, Multiple Data

Taxonomia de Flynn

Single Instruction Single Data:

Computadores de Von Neumann: microprocessadores tradicionais e Computadores de Grande Porte;

Algumas arquiteturas alternativas

Máquinas "Harvard" (arquitetura com múltiplas áreas de memória e barramentos, utilizada nos microcontroladores);

Sistemas *pipeline* e superescalares.

Single Instruction Multiple Data

Multiple Instruction Single Data

Multiple Instruction Multiple Data

Taxonomia de Flynn

Single Instruction Single Data

Single Instruction Multiple Data:

- Uma Unidade de Controle, múltiplas ULA (Unidade Lógica e Aritmética);
- Processadores Vetoriais, Super-computadores;
- Utilizadas em Computadores convencionais para processamento multimídia (Intel Core i7 possui a extensão *SSE - Streaming SIMD Extensions*);

Multiple Instruction Single Data

Multiple Instruction Multiple Data

Taxonomia de Flynn

Single Instruction Single Data

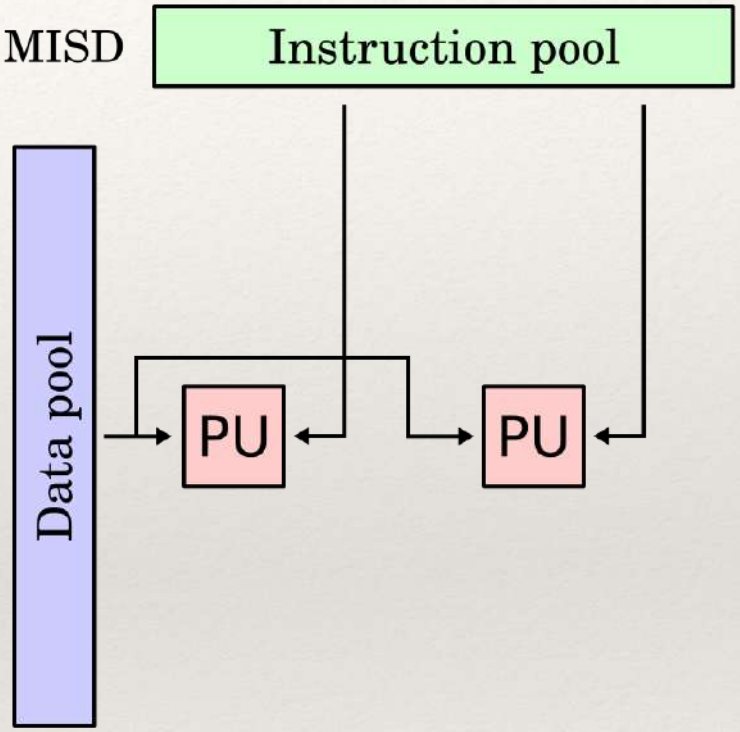
Single Instruction Multiple Data

Multiple Instruction Single Data:

Não existem no mercado computadores que operem nesse modelo;

Há alguns exemplos raros em sistemas de controle complexos (aeroespaciais?).

Multiple Instruction Multiple Data



Taxonomia de Flynn

Single Instruction Single Data

Single Instruction Multiple Data

Multiple Instruction Single Data

Multiple Instruction Multiple Data:

Múltiplos processadores e múltiplos Computadores;

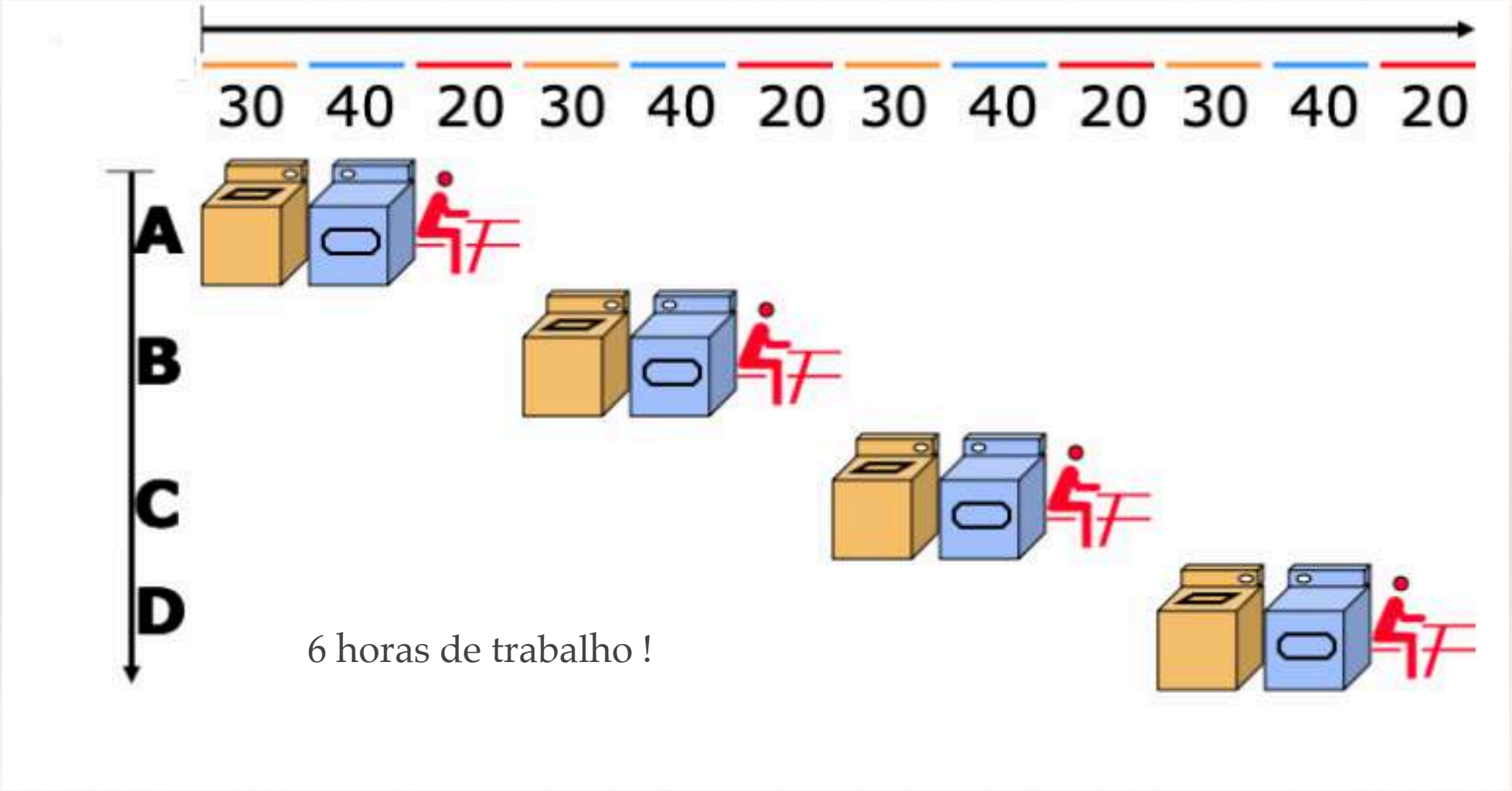
Categoria mais comum no estudo dos "Sistemas Distribuídos”.

Pipeline

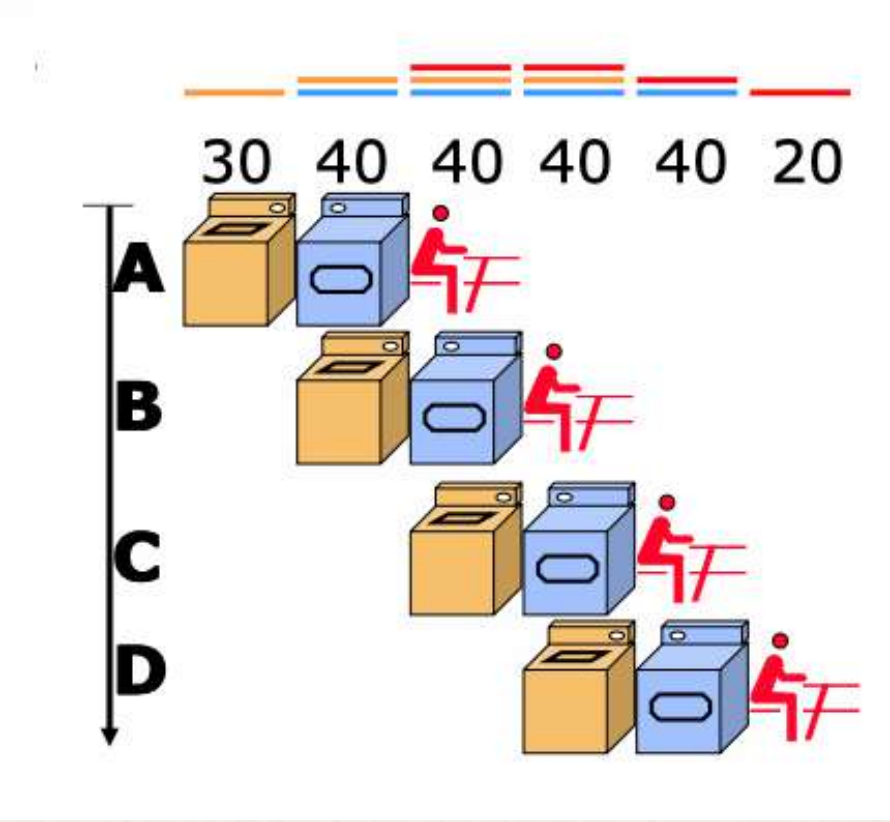
Ana, Beto, Carol e Damásio precisam lavar suas roupas na lavanderia do prédio, mas só existe uma lavadora, uma secadora e uma mesa para dobrar as roupas;

A lavagem demora 30 minutos, a secagem 40 minutos, e para dobrar são necessários mais 20 minutos. Ou seja, 1 hora e meia para tudo.

Sem *pipeline*

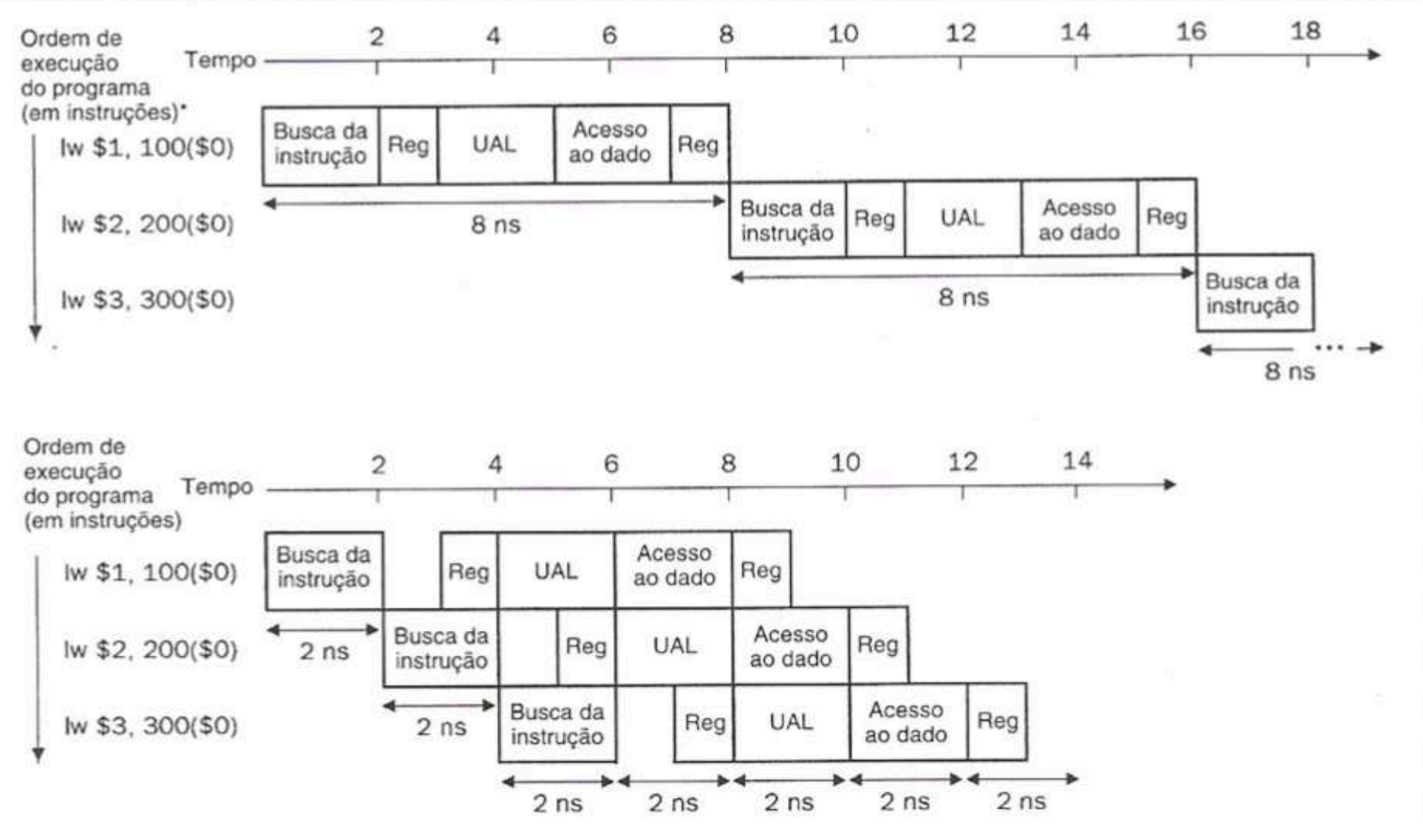


Com *pipeline*



3:30h de trabalho !

Pipeline - ciclos de instrução



Pipeline e Multiescalares

A operação *pipeline* tipicamente exige mudanças na arquitetura Von Neumann

- Múltiplos barramentos (dados e endereço)
- Memórias independentes para dados e instruções

As arquiteturas superescalares são uma evolução do *pipeline*, por permitirem a execução de múltiplas instruções, nem sempre em estágios similares do *pipeline*. Estas exigem mudanças ainda maiores na arquitetura de hardware

- Unidade de busca de instruções, com algoritmo preditivo de desvios;
- Unidade de decodificação com leitura simultânea de diversos registradores de instrução;
- Múltiplas ULAs e de aritmética de ponto flutuante.

Dispositivos de E/S

Tratamento de E/S *round-robin*

- Ciclo infinito de leitura;
- Tempo de acesso, interatividade e "bufferização".

Interrupções

- Controle via *hardware*;
- Rotinas de "exceção".

Computação Física

A interação do usuário com dispositivos computacionais acontece pelos dispositivos de entrada e saída convencionais, como teclado, telas sensíveis ao toque, mouse e dispositivo apontadores, microfones e alto-falantes;

No entanto, o poder dos dispositivos computacionais pode ser aplicado diretamente em soluções “físicas" no mundo real, com a medição e sensibilidade a eventos, e também com a atuação sobre os mesmos, sem intervenção humana;

Dispositivos computacionais que fazem este trabalho podem, muitas vezes, ser mais simples e de baixo custo. É o caso dos microcontroladores.

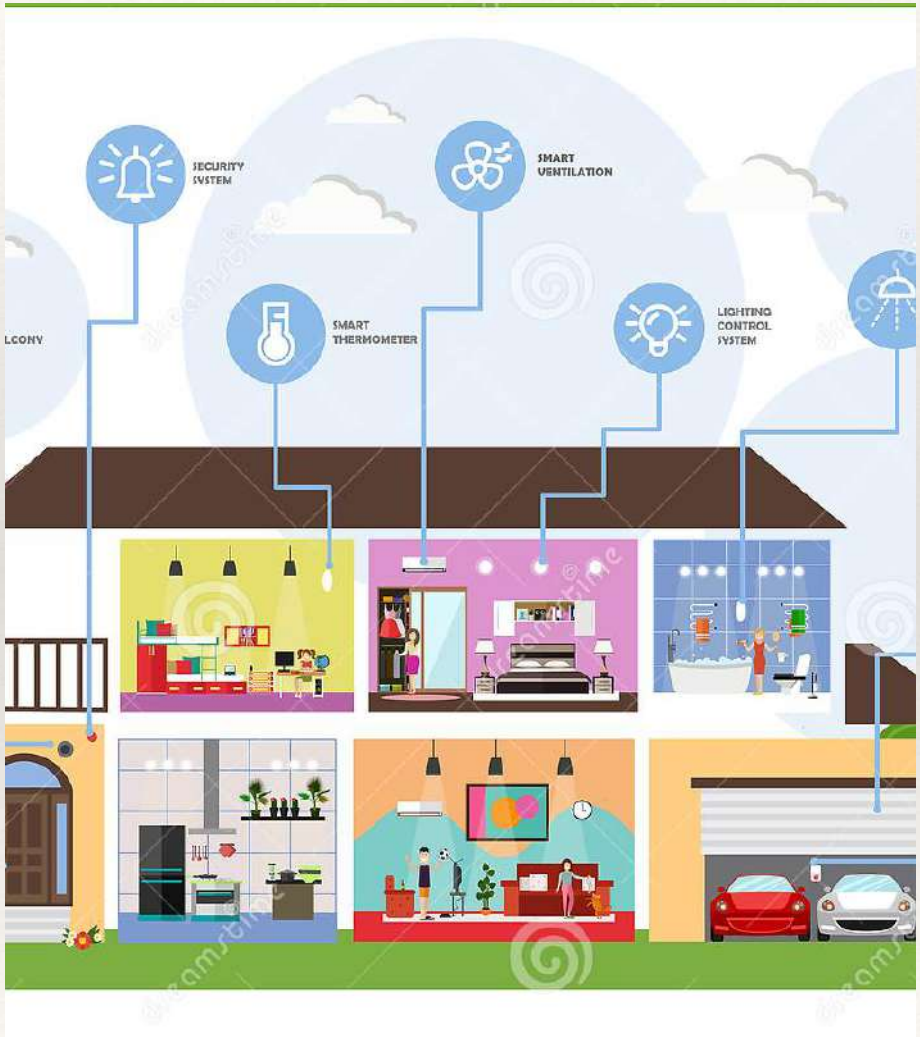
Aplicações Típicas

Segurança Patrimonial

- Sensores de presença;
- Abertura de portas e janelas;
- Simulação de presença;
- Acionamento de Alarmes.

Automação Residencial

- Iluminação e Sonorização;
- Aquecedores e Ar-Condicionado;
- Interação por Voz;
- Roteiros Pré-programados.



Aplicações Típicas

Automação de Sistemas

Doméstico, Profissional ou Industrial;

Controle, Segurança e Economia.

Um exemplo: Bomba D'Água

Controle: ligar e desligar nos momentos corretos;

Segurança: evitar contato humano;

Economia: proteção do equipamento, e controle do consumo.

Sensores: ativação e desativação por medição de nível dos reservatórios, medição do fluxo d'água para evitar operação seca e controlar consumo de água, medição da temperatura de operação, medição da tensão e corrente elétrica, medição do nível de ruído durante a operação.

Aplicações Típicas

Robótica

Sensores diversos: acelerômetro, pressão, radar, visão computacional, vibração;

Interação com o ambiente e usuários;

Aplicações com diferentes níveis de complexidade, como mapeamento de ambientes internos e externos, aprendizado de máquina etc;

Pesquisas em andamento e mercado gigantesco - é o momento de aprender!

Como implementar

Placas de Protótipo com microcontroladores (ex.Arduíno)

- Não possuem sistema operacional, e sim *firmware;*
- Programas (*sketchs*) são carregados pelo *bootloader* pré-instalado;
- Oferecem portas analógicas e digitais;
- Oferecem memória e performance limitadas, porém suficientes.

SBCs (*Single Board Computers*) (ex. Raspberry)

- Rodam SOs típicos, como Android, Linux e Windows;
- Oferecem I/O padronizados, como HDMI, USB, Ethernet;
- Oferecem memória e performance de processamento mais elevadas.

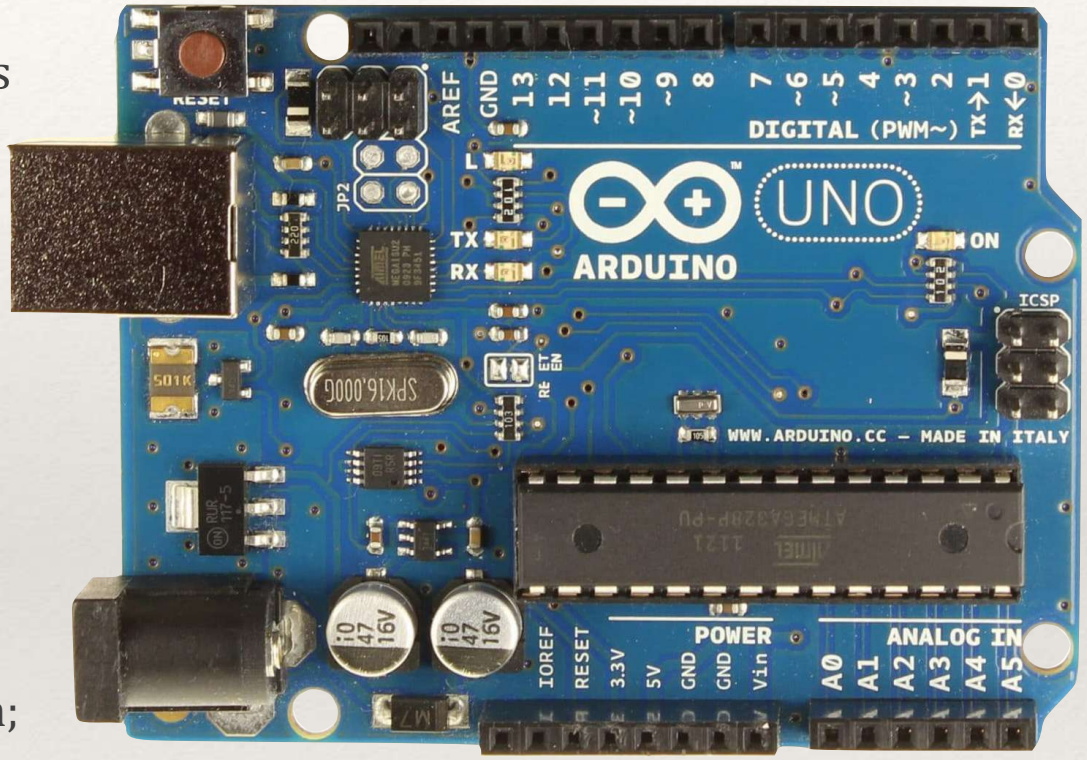
Conhecendo o Arduíno Uno R3

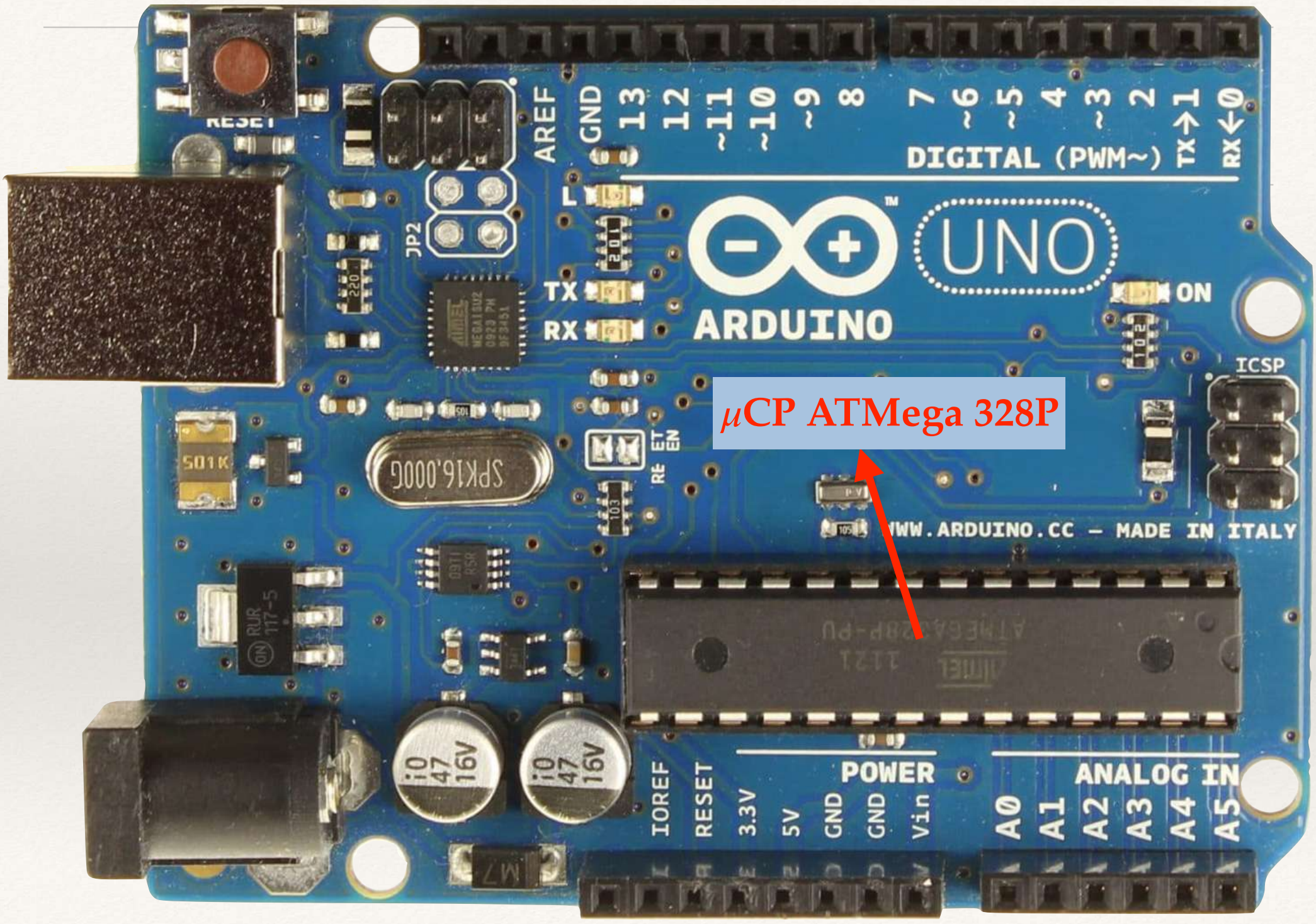
Tudo começou na Itália em 2005;

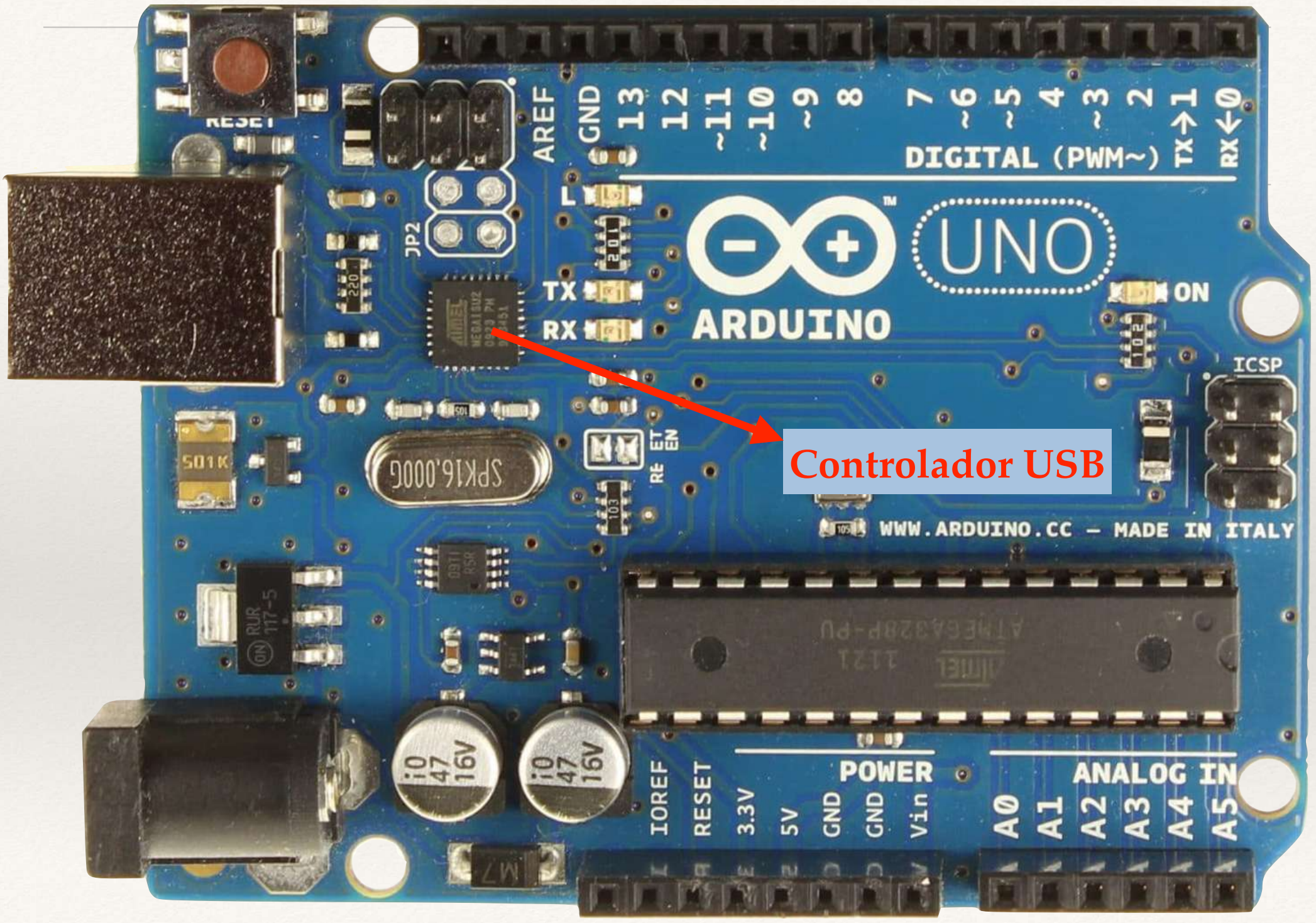
Tecnologia aberta (existem diversos fabricantes produzindo produtos similares);

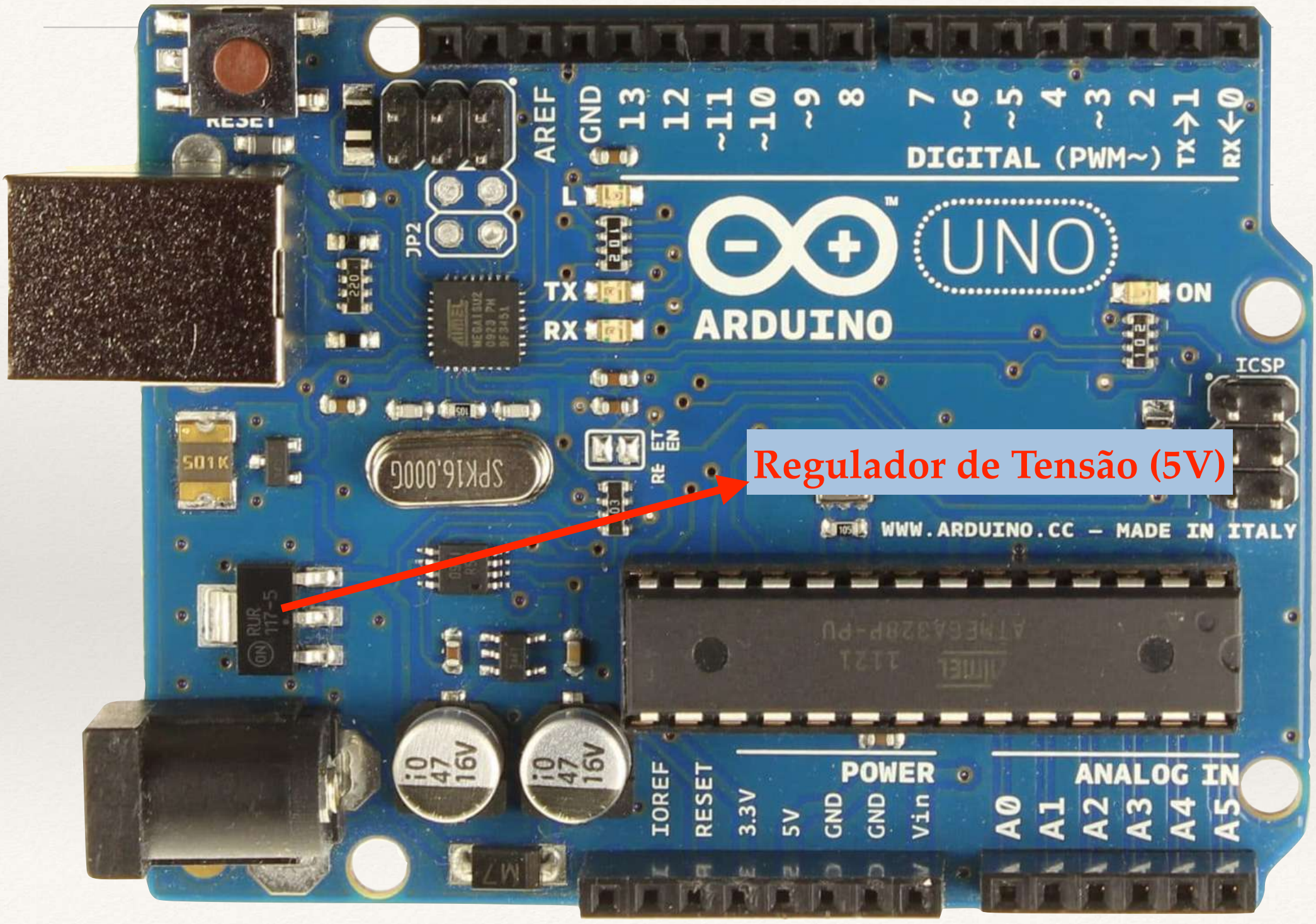
São diversos dispositivos. Vamos destacar o **Arduíno Uno versão 3:**

- Tensão de Operação: 5V
- 14 pinos de I/O (6 PWM);
- 6 entradas analógicas;
- Processador ATmega328P;
- 1 KB EPROM, 2KB SRAM, 32KB Flash;
- Ambiente de desenvolvimento gratuito e disponível para diversas plataformas (Windows. Mac, Linux). Não há suporte oficial para celulares e tablets (iOS ou Android).



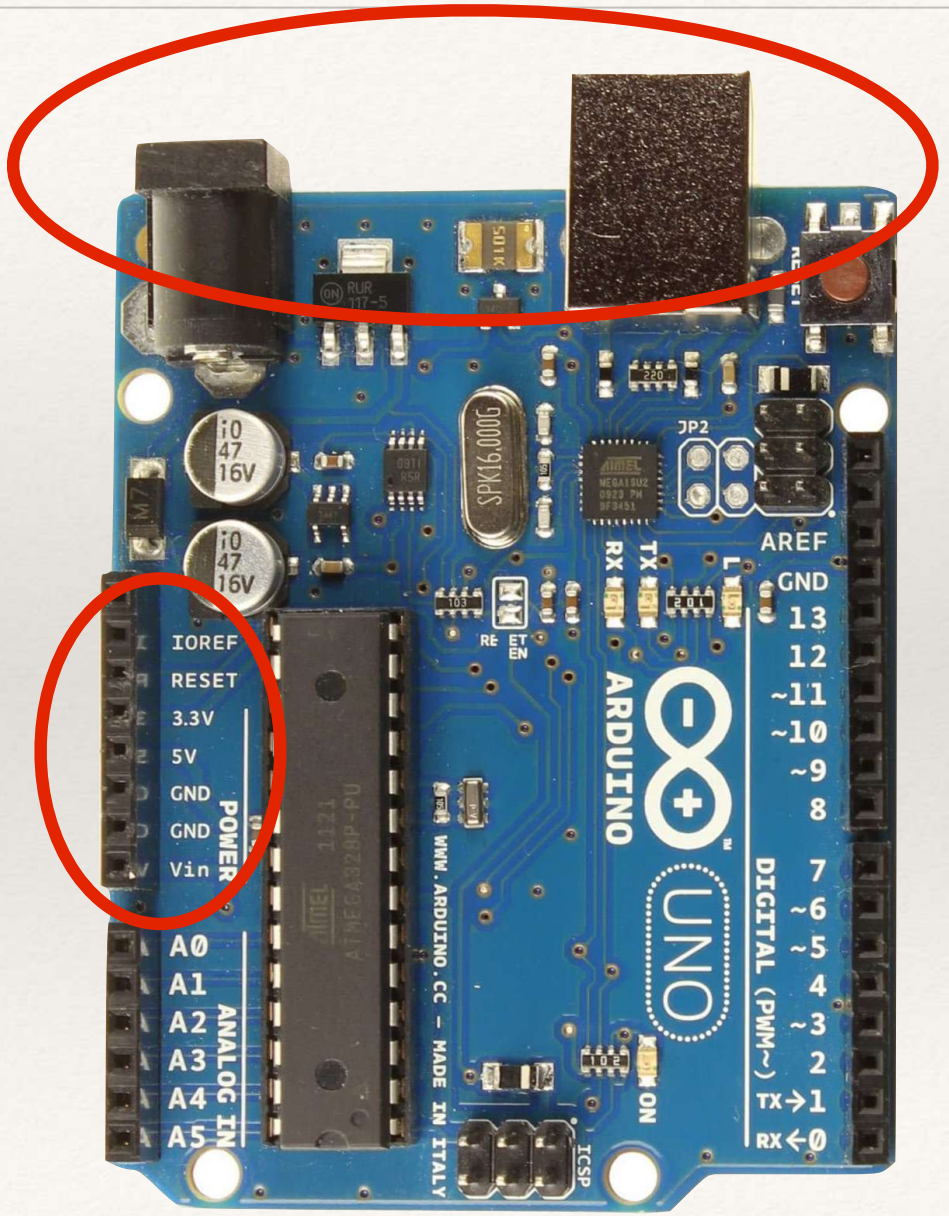

μCP ATMega 328P
ARDUINO
UNO
DIGITAL (PWM~)
POWER
ANALOG IN
ICSP
ON
TX
RX
WWW.ARDUINO.CC – MADE IN ITALY


Controlador USB
DIGITAL (PWM~)
ARDUINO
UNO
ON
ICSP
POWER
ANALOG IN
WWW.ARDUINO.CC – MADE IN ITALY


Regulador de Tensão (5V)
DIGITAL (PWM~)
ARDUINO
UNO
ON
ICSP
POWER
ANALOG IN
WWW.ARDUINO.CC – MADE IN ITALY

Portas do Arduíno Uno

- Alimentação
- Portas Analógicas
- Portas Digitais

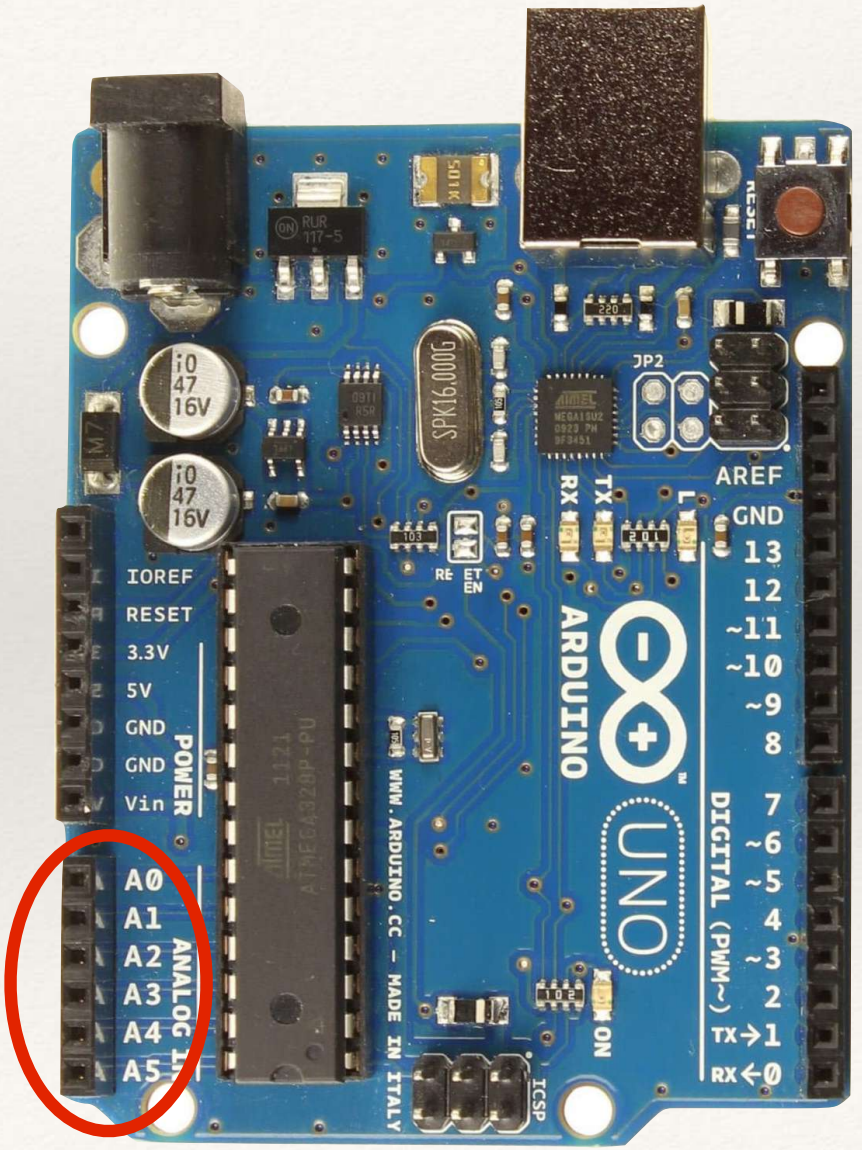


Portas do Arduíno Uno

Alimentação

Portas Analógicas

Portas Digitais

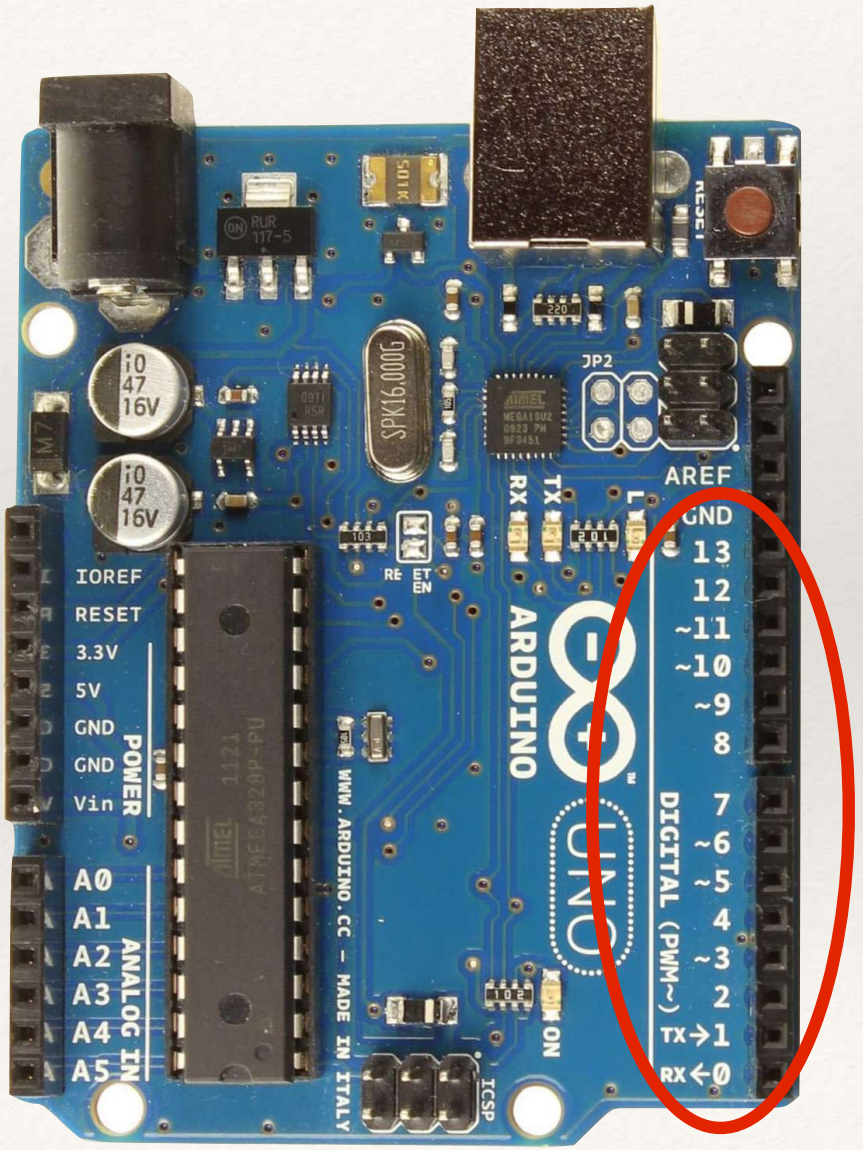


Portas do Arduíno Uno

Alimentação

Portas Analógicas

Portas Digitais



Portas de Alimentação

Alim. / Comunicação

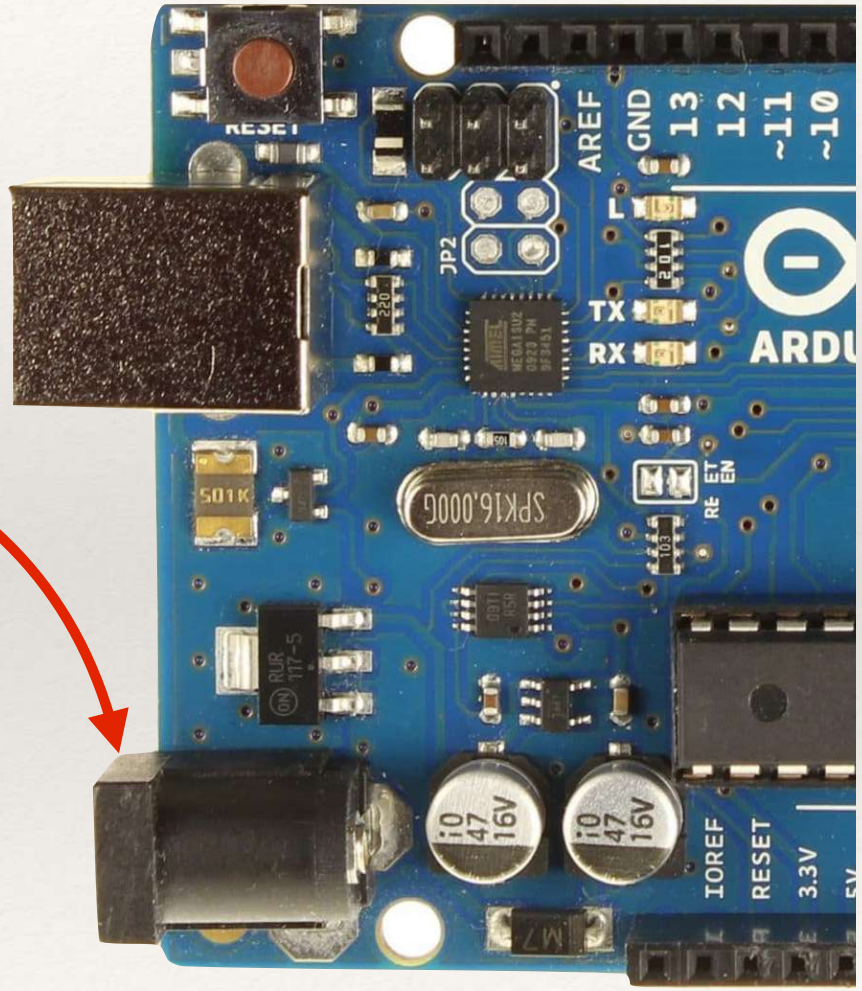
Entrada USB

Provê comunicação com o computador (e alimentação).

Entrada Fonte DC Externa

Provê alimentação a partir de fonte de 5 a 17V, <100mA;

Não se recomenda >12V (aquecimento componentes).



Portas Analógicas

Portas Digitais

Portas de Alimentação

Alimentação (pinos)

Saídas de 5V e 3,3V

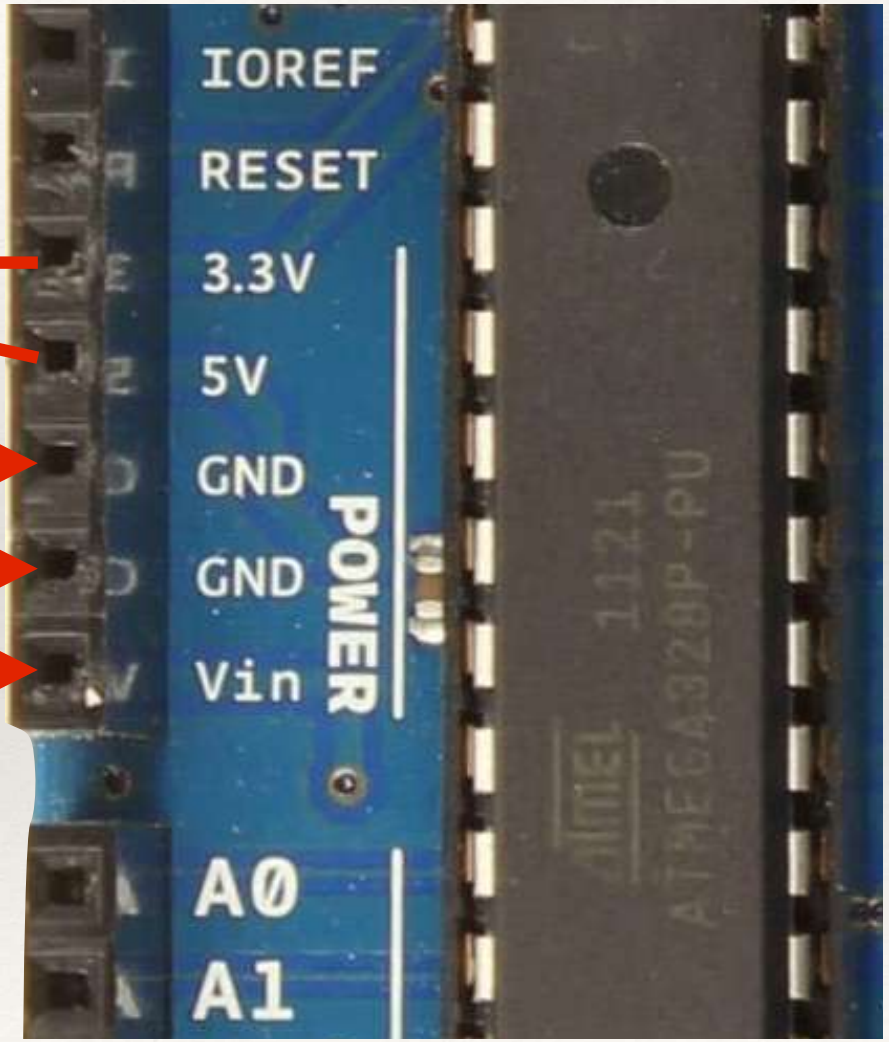
Corrente máxima de 40mA

Entrada Vin

Alimentação de 5V ≥100mA

Portas Analógicas

Portas Digitais



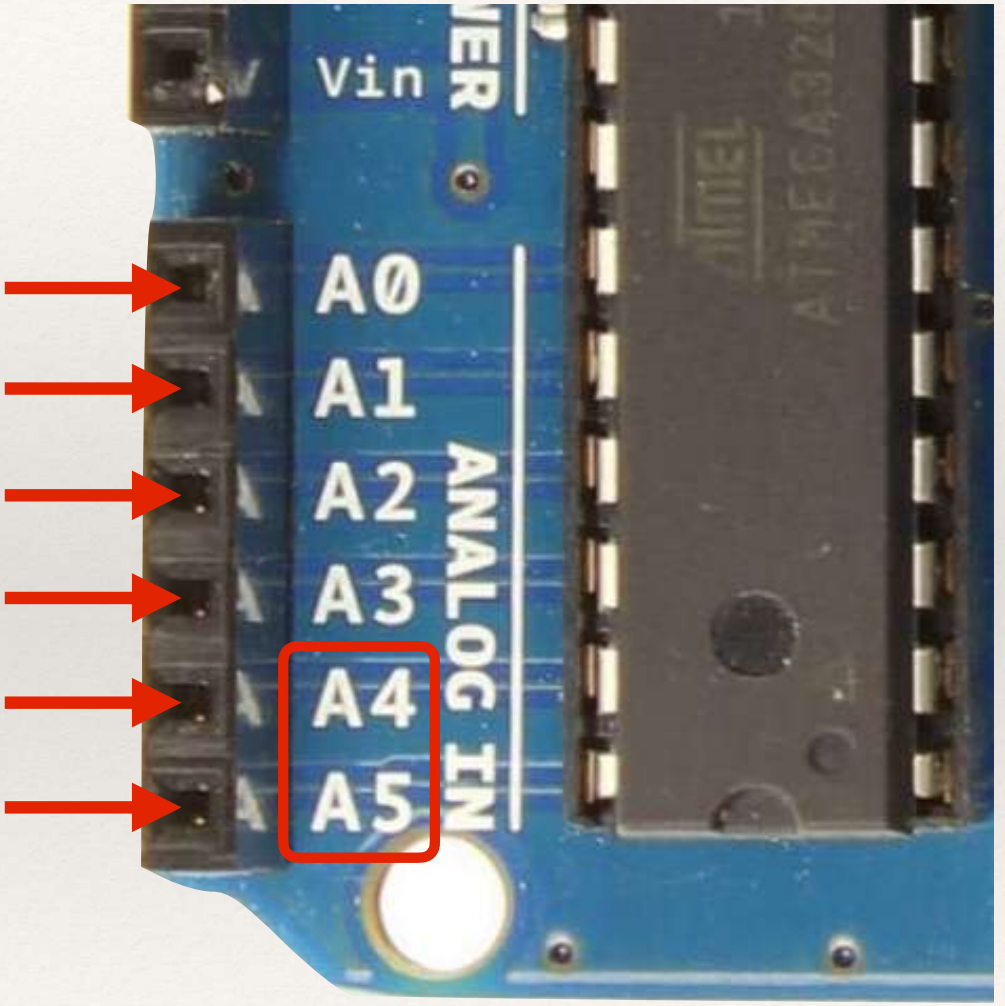
Portas Analógicas

Alimentação

Portas Analógicas

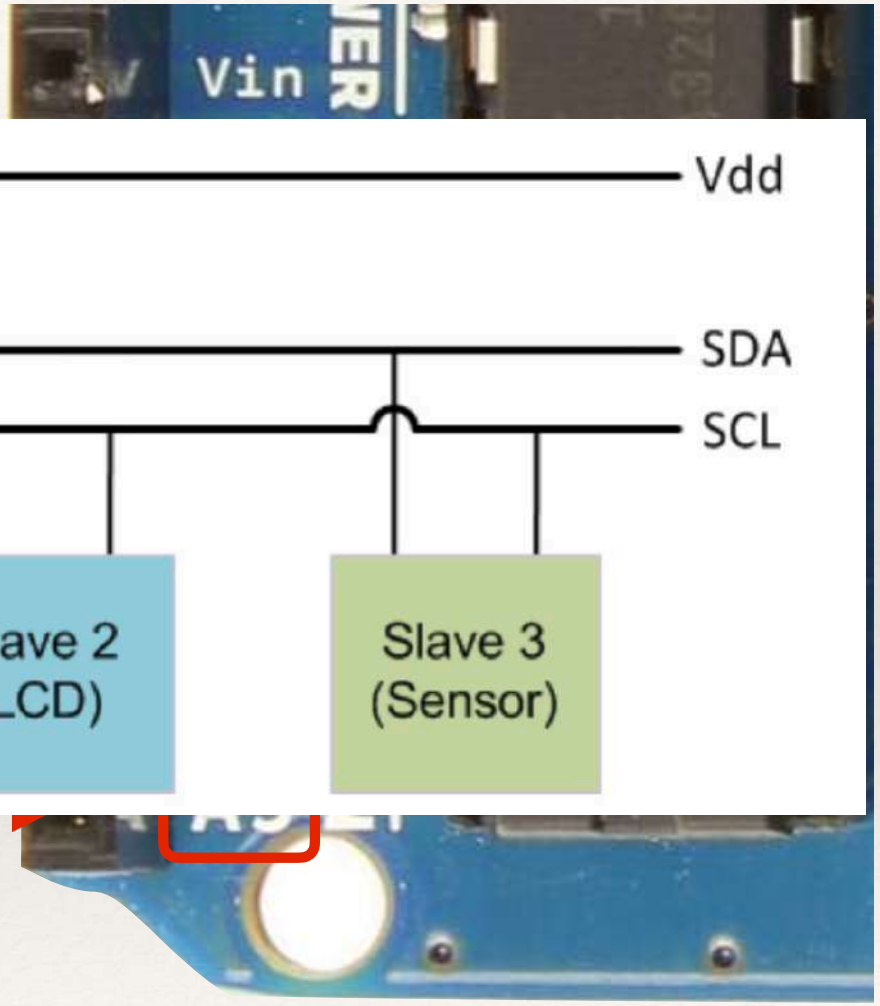
- Entradas com conversor A/D
  - Valores de 10 bits (0~1023)
  - 0 = **0V**
  - 1023 = **5V** (IOREF reduz valor máximo)
- Funções Especiais
  - I2C (SCL e SDA): portas A5 e A4

Portas Digitais



Portas Analógicas

Alimentação

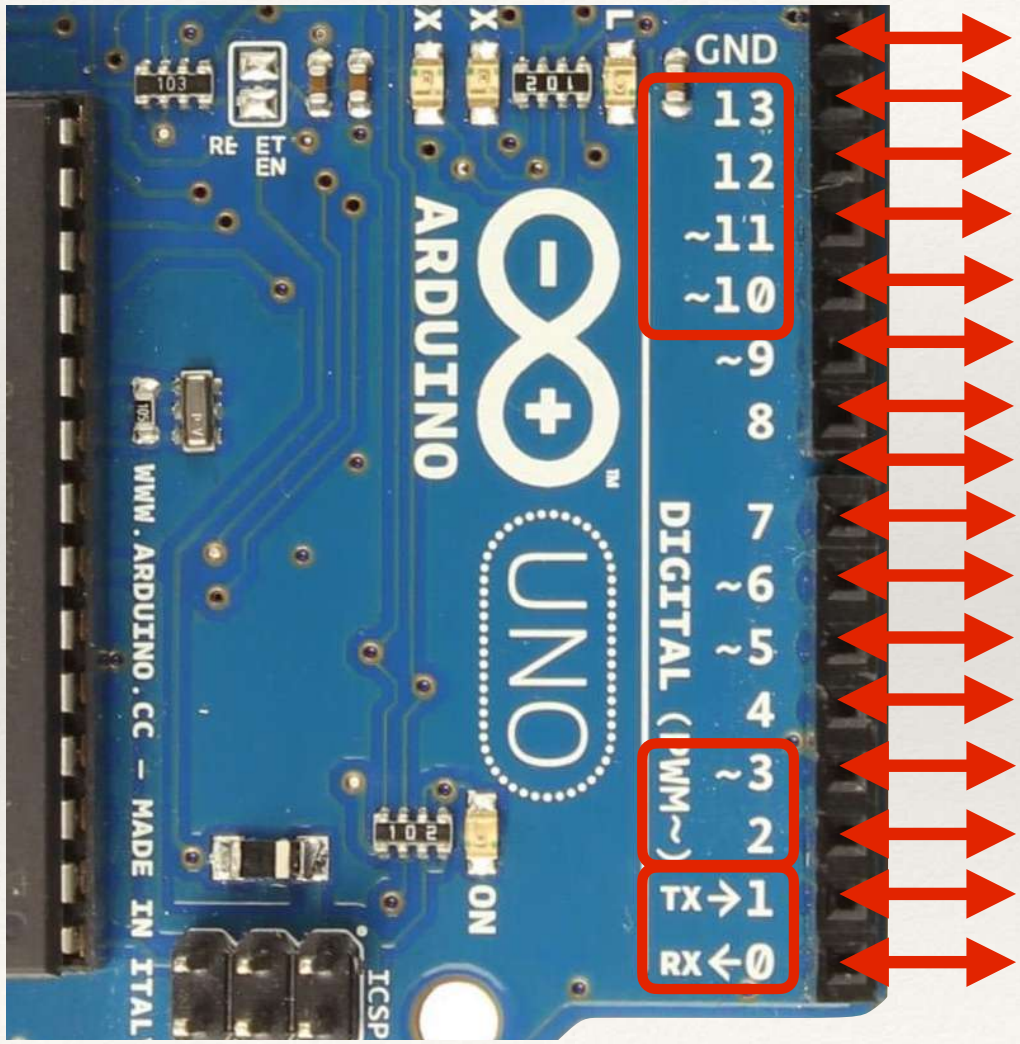


P

I2C (SCL e SDA): portas A5 e A4

Portas Digitais

Portas Digitais



Alimentação

Portas Analógicas

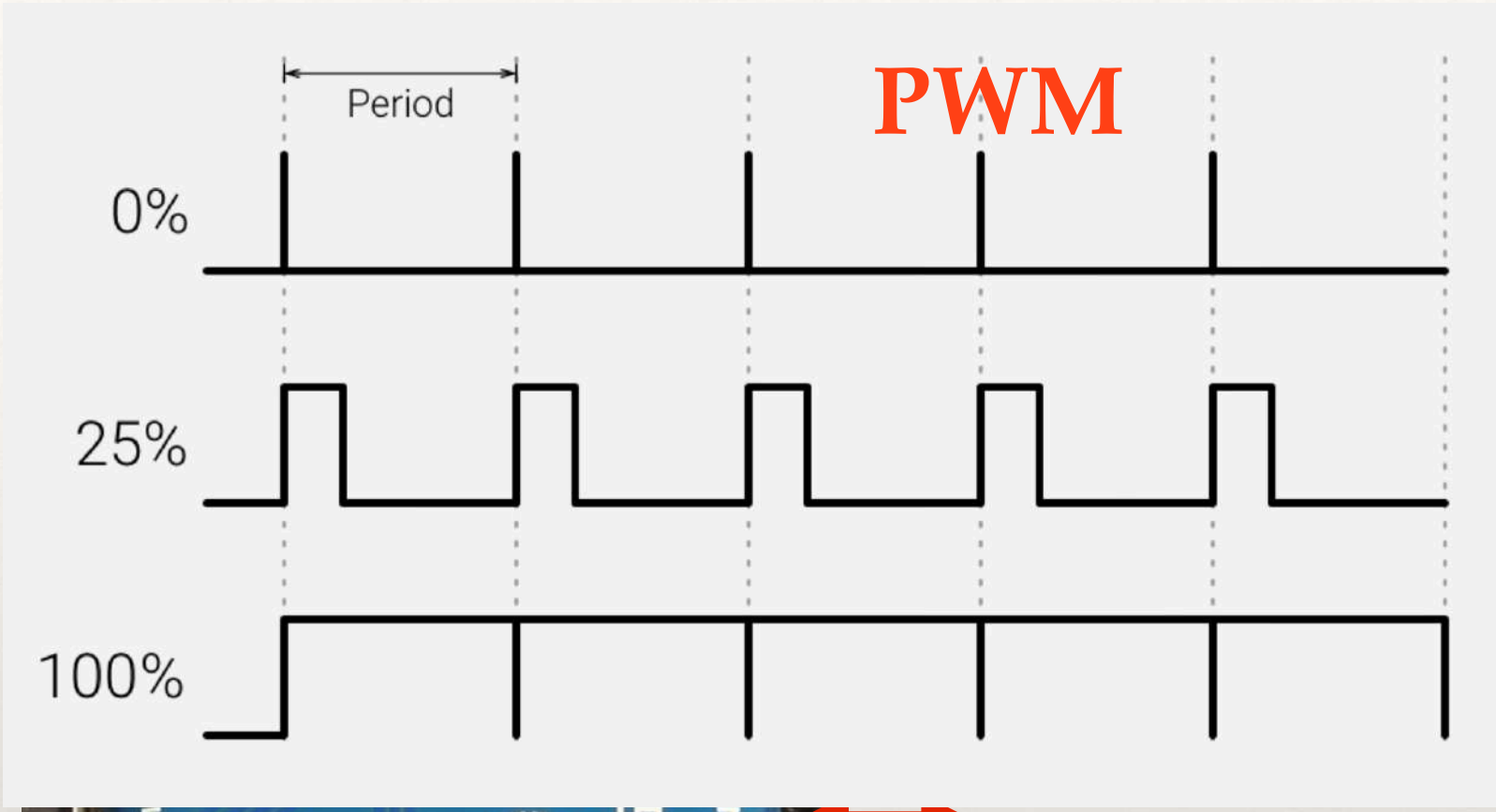
Portas Digitais

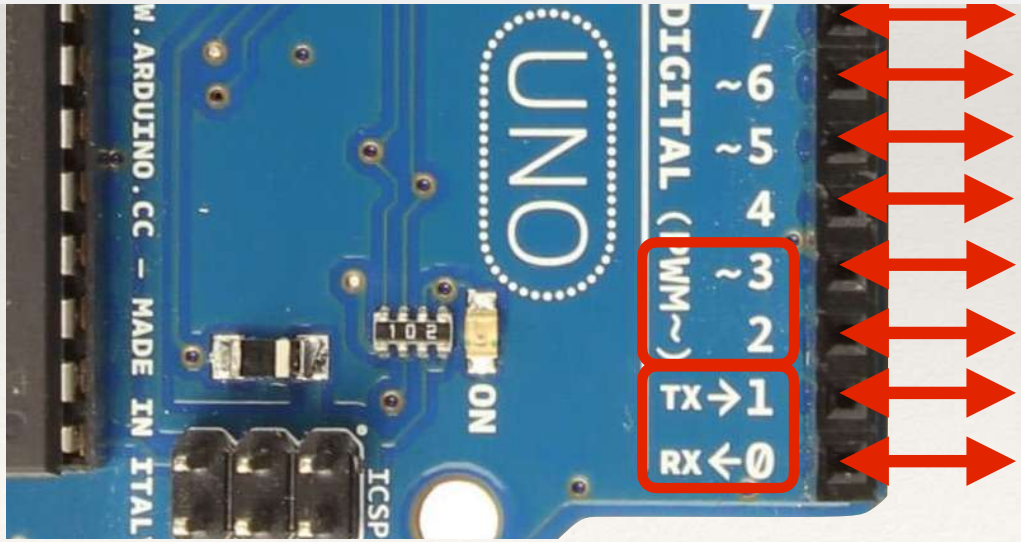
- Entrada ou Saída
  - Configurada via SW
  - Saídas marcadas com ~ são **PWM** (*Pulse Width Modulation*)
  - < 20mA (máx.40 mA); total 200mA
- Funções especiais
  - **UART**: Rx/Tx (portas 0 e 1)
  - **Interrupções**: INT0 e 1 (portas 2 e 3)
  - **SPI**: SS,MOSI,MISO,SCK(p. 10~13)
  - **Pino 13**: LED interno

PWM

igitais

s





Saídas marcadas com ~ são PWM (*Pulse Width Modulation*)

< 20mA (máx.40 mA); total 200mA

Funções especiais

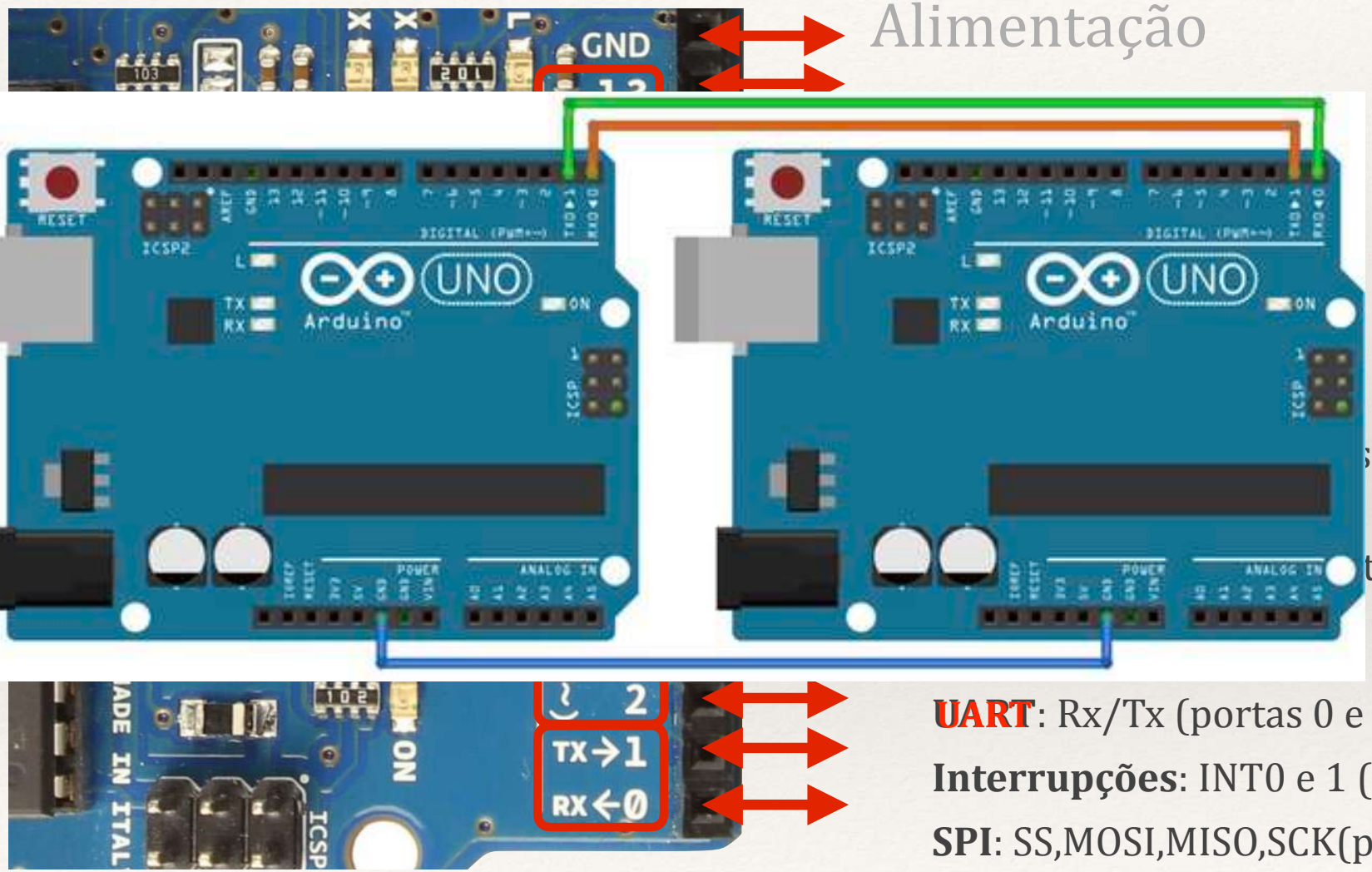
UART: Rx/Tx (portas 0 e 1)

Interrupções: INT0 e 1 (portas 2 e 3)

SPI: SS,MOSI,MISO,SCK(p. 10~13)

Pino 13: LED interno

Portas Digitais



Alimentação

são PWM (*Pulse*

tal 200mA

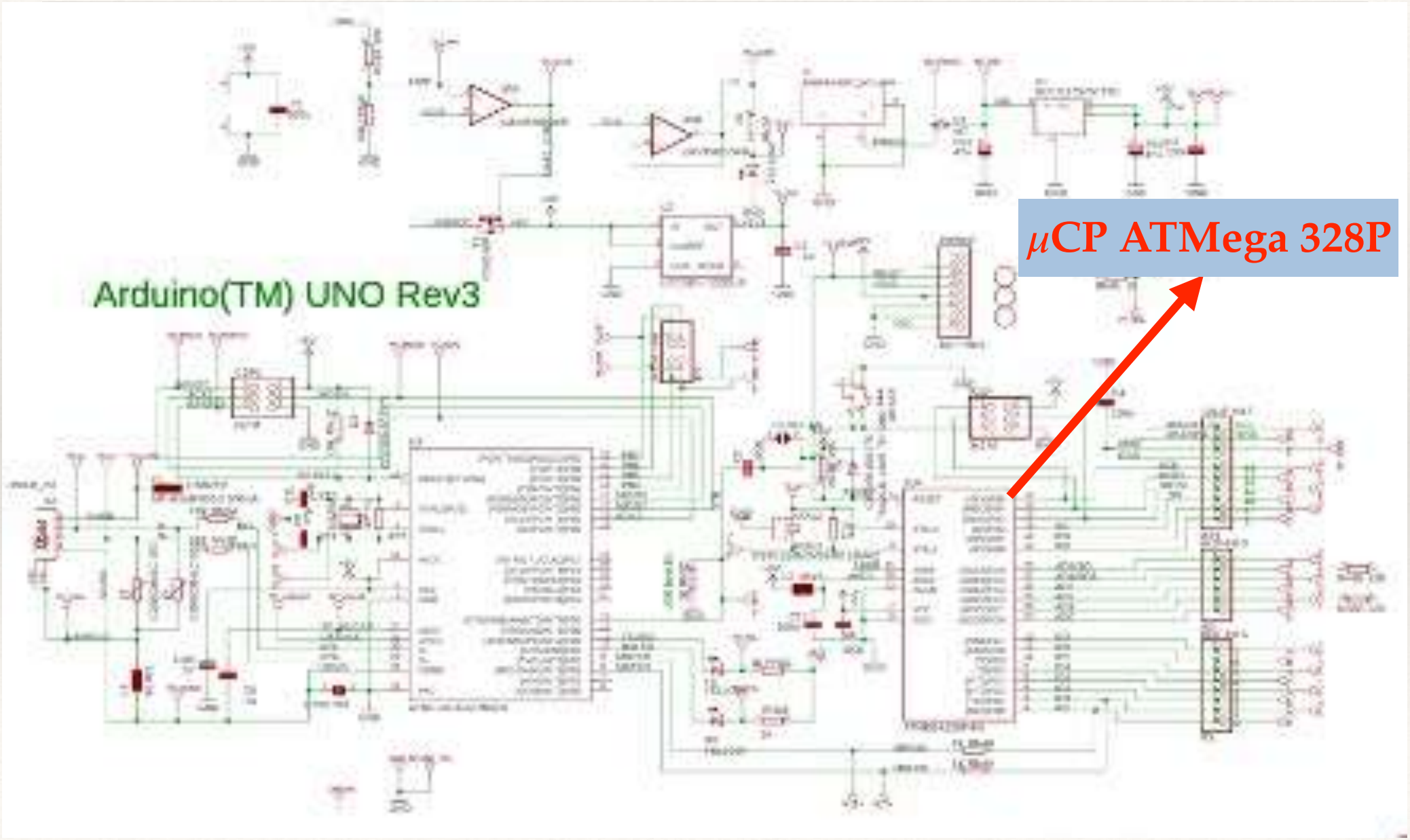
UART: Rx/Tx (portas 0 e 1)

Interrupções: INT0 e 1 (portas 2 e 3)

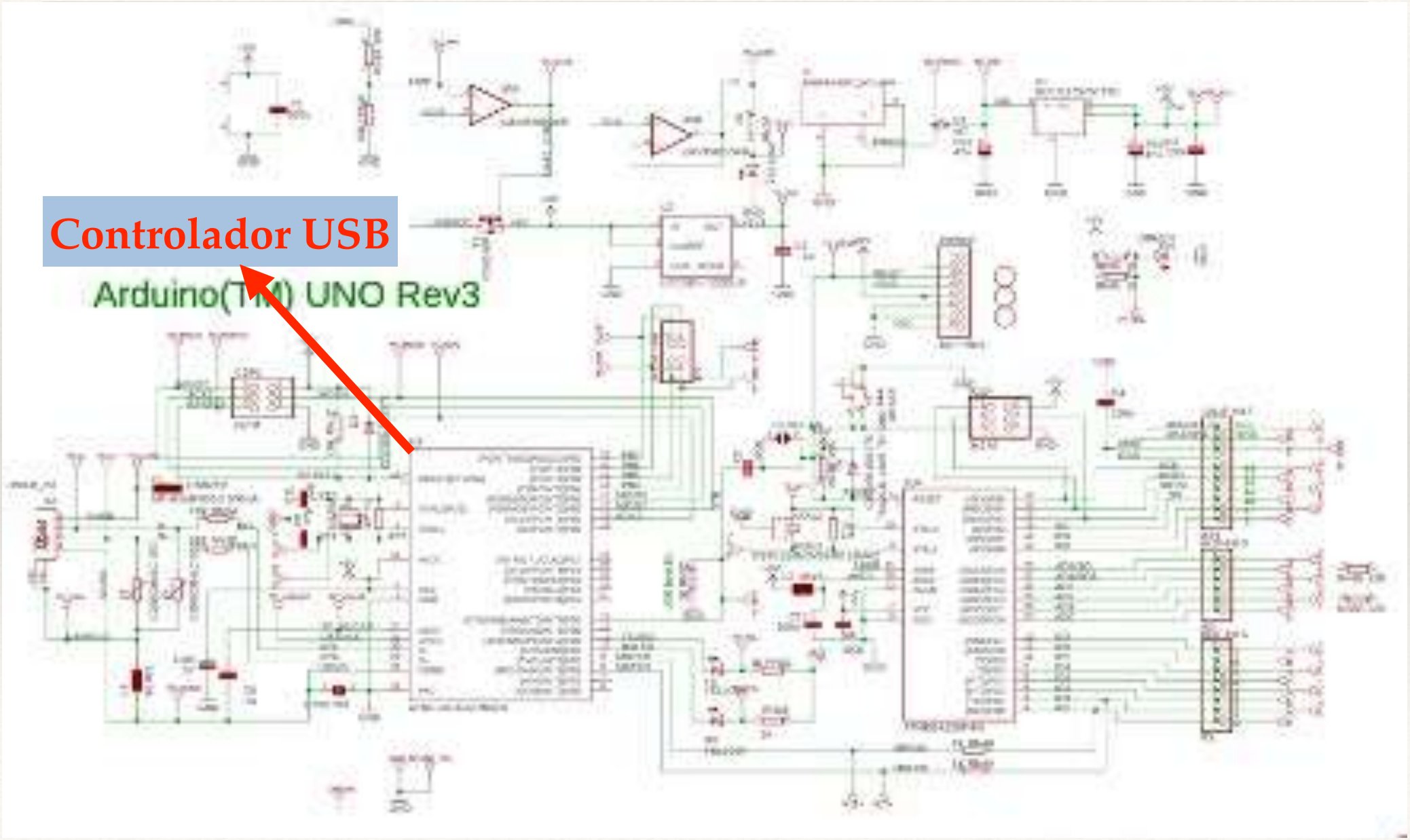
SPI: SS,MOSI,MISO,SCK(p. 10~13)

Pino 13: LED interno

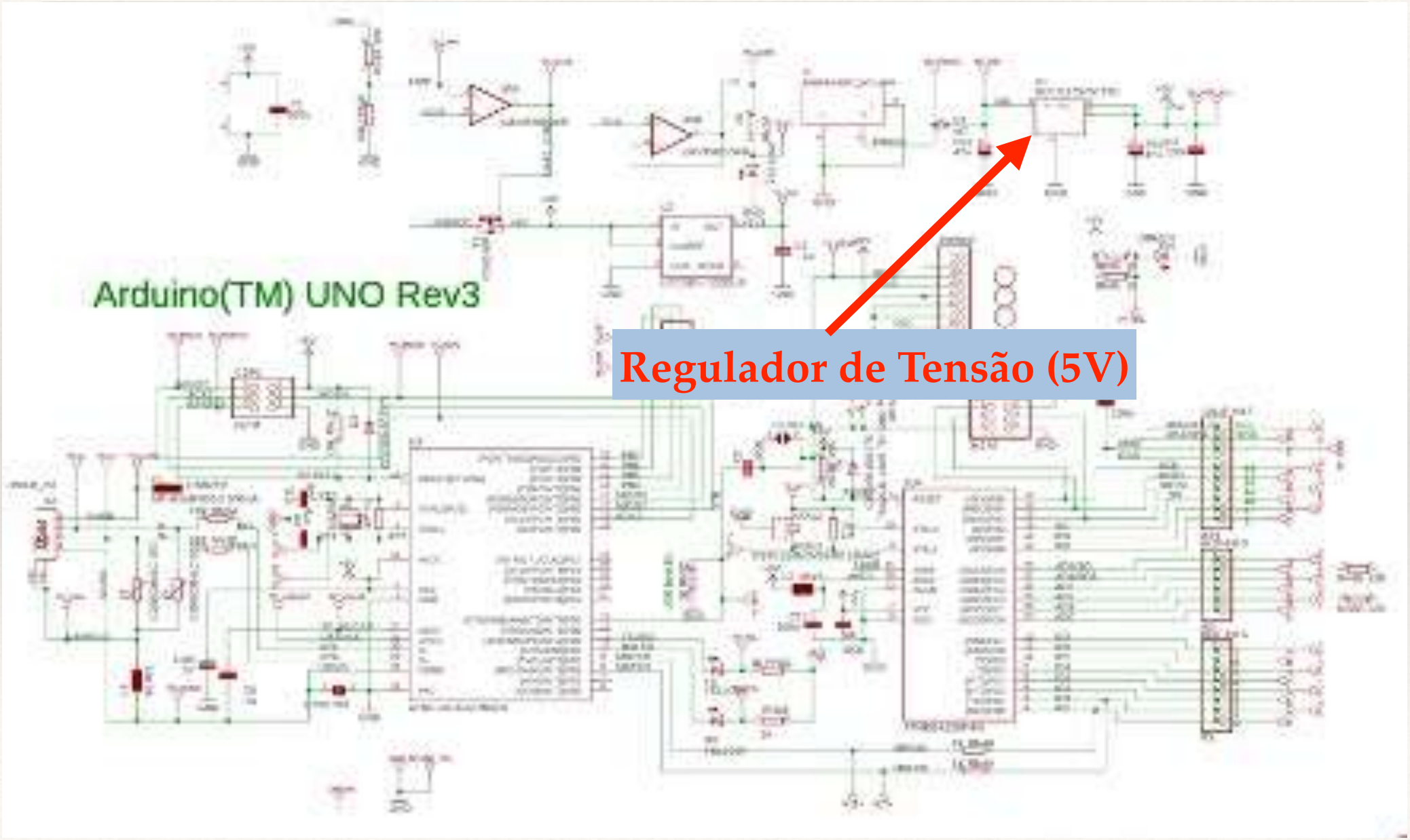
Por dentro do Arduíno Uno R3



Por dentro do Arduíno Uno R3



Por dentro do Arduíno Uno R3



Outras placas de Protótipo

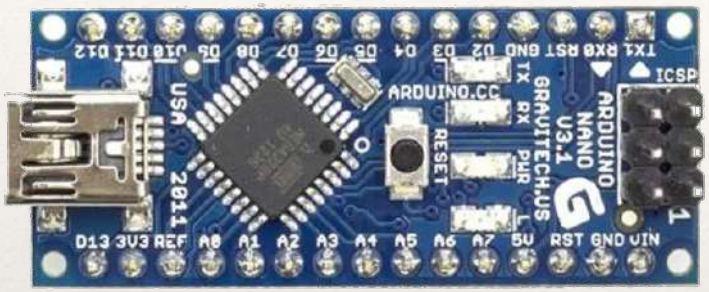
Há outros “Arduínos”:

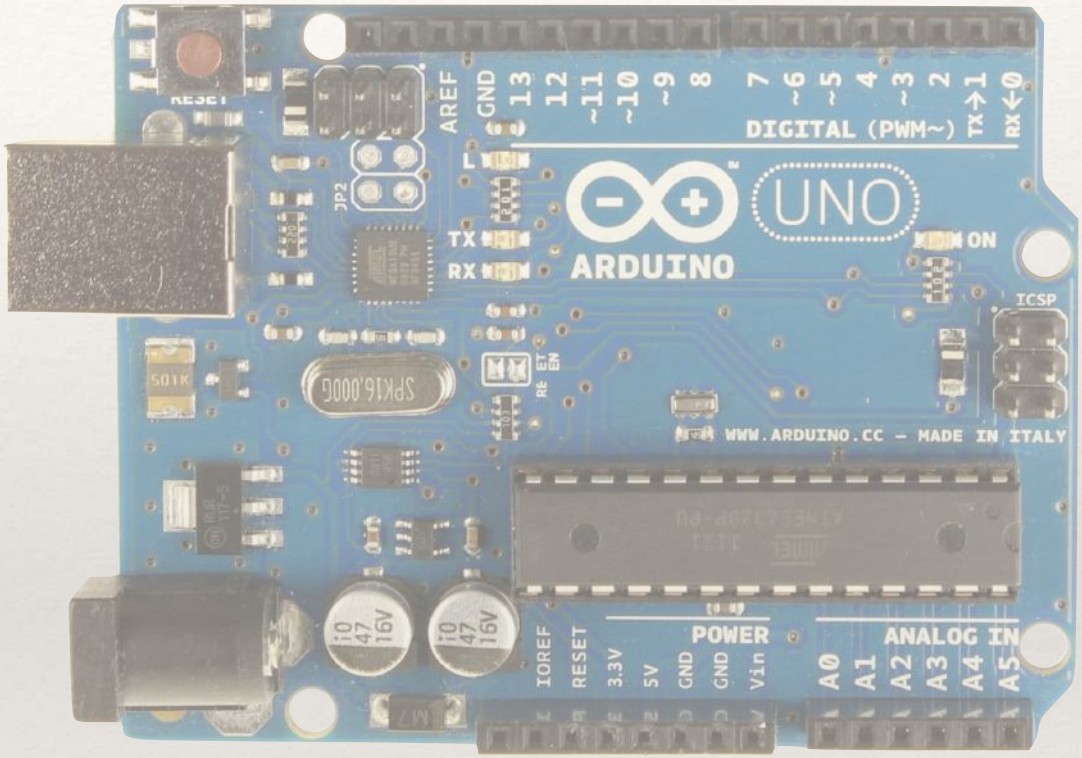
Nano (*)	Mega (*)	LilyPad (*)
Uno R4 Minima	Uno R4 Wi-Fi	

Há também outras linhas, como:

- **NodeMCU** (baseado no controlador ESP8266): baixo custo e recursos de comunicação
- **ESP32**: baixo custo, recursos de comunicação avançados e alta capacidade
- **Teensy 4** (baseado no ARM Cortex-M7 de 600MHz): alta performance
- Placas baseadas no **MSP430**, da Texas Instruments: baixo consumo
- Placas baseadas no **STM32**: versatilidade

Arduíno Nano (descontinuado)





Mesmos recursos do Uno

Hardware

Mesmo processador;

Mesma memória;

Entradas e Saídas

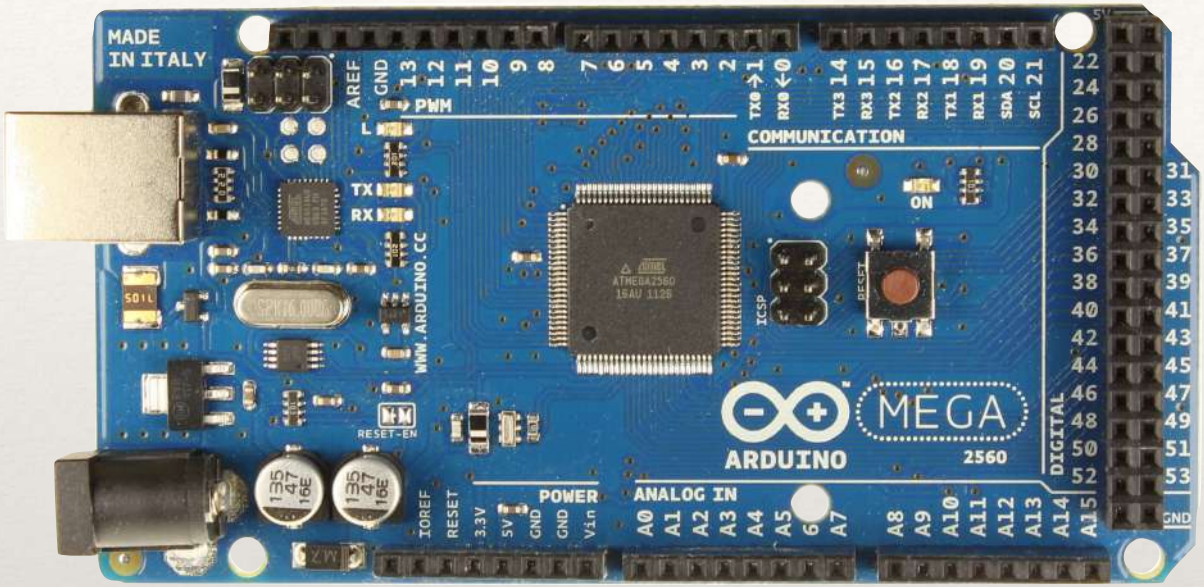
Layout diferente de pinos

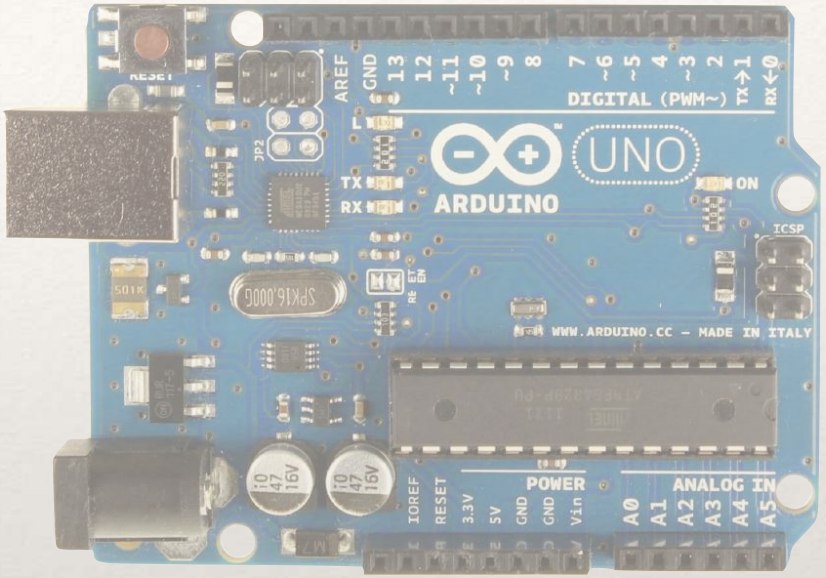
Ex. 8 portas analógicas, só tem 1 GND;

Não tem pinos de conexão

Exige soldagem, ou, se instalados pinos, *protoboard*;

Arduíno Mega (descontinuado)





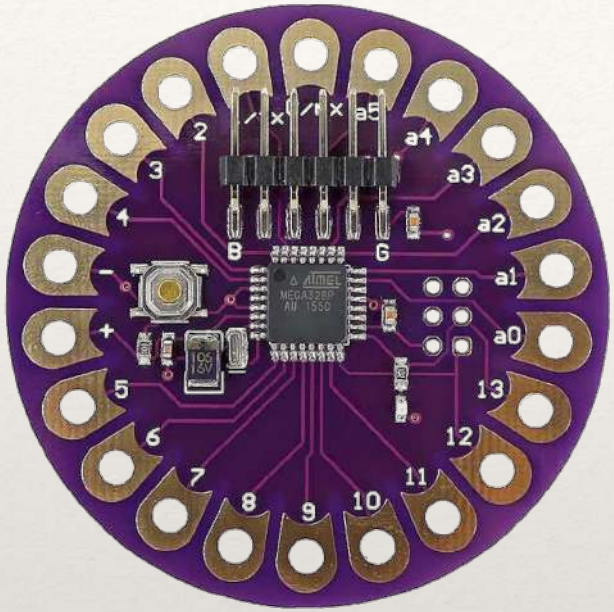
Microcontrolador e Memória

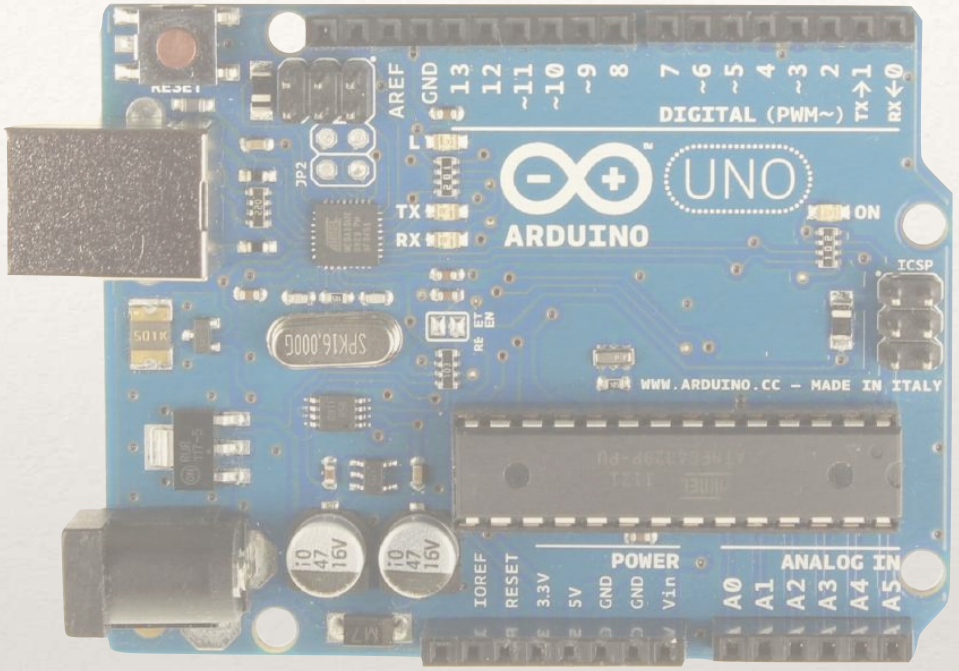
- ATMega2560;
- SRAM de 8KB, EEPROM de 4KB;
- Memória Flash de 256KB;
- *Clock* de 16MHz.

Entradas e Saídas

- 54 portas de E/S digitais;
  (15 portas PWM)
- 16 portas de entrada analógica;
- 4 UARTs e pinos compatíveis (Uno).

Arduíno LilyPad (descontin.)





μCP, Memória e Alimentação

ATMega328V ou ATMega168V;

SRAM de 1KB, EEPROM de 1KB;

Memória Flash de 16KB;

Clock de 8MHz; opera entre 2,7 e 5,5V.

Entradas e Saídas

14 portas de E/S digitais;
(6 portas PWM)

6 portas de entrada analógica.

Arduíno Uno R4 Minima

Compatível com a versão R3 em diversos aspectos (pinagem e tensão, por exemplo);

Novo microcontrolador (RA4M1 Remessas) contendo um ARM Cortex-M4 com *clock* de 48MHz e unidade de ponto flutuante;

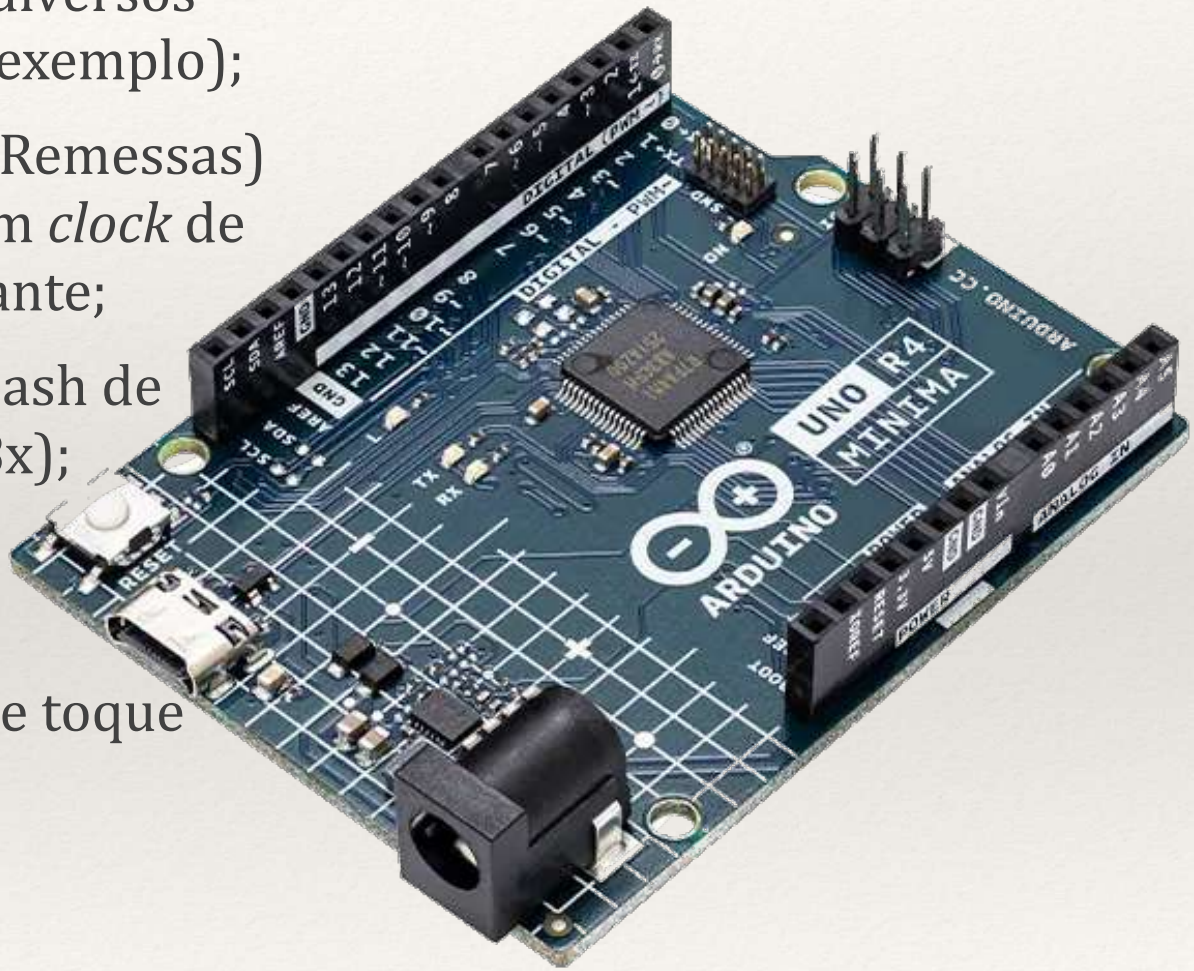
Memória SRAM de 32KB (16x), flash de 256KB (8x) e EEPROM de 8KB (8x);

Nova fonte de alimentação chaveada (6 a 24V) com 1,2A;

RTC, DAC (12 bits), HID, sensor de toque capacitivo e suporte a CAN (com transceptor interno);

ADC agora tem 14 bits;

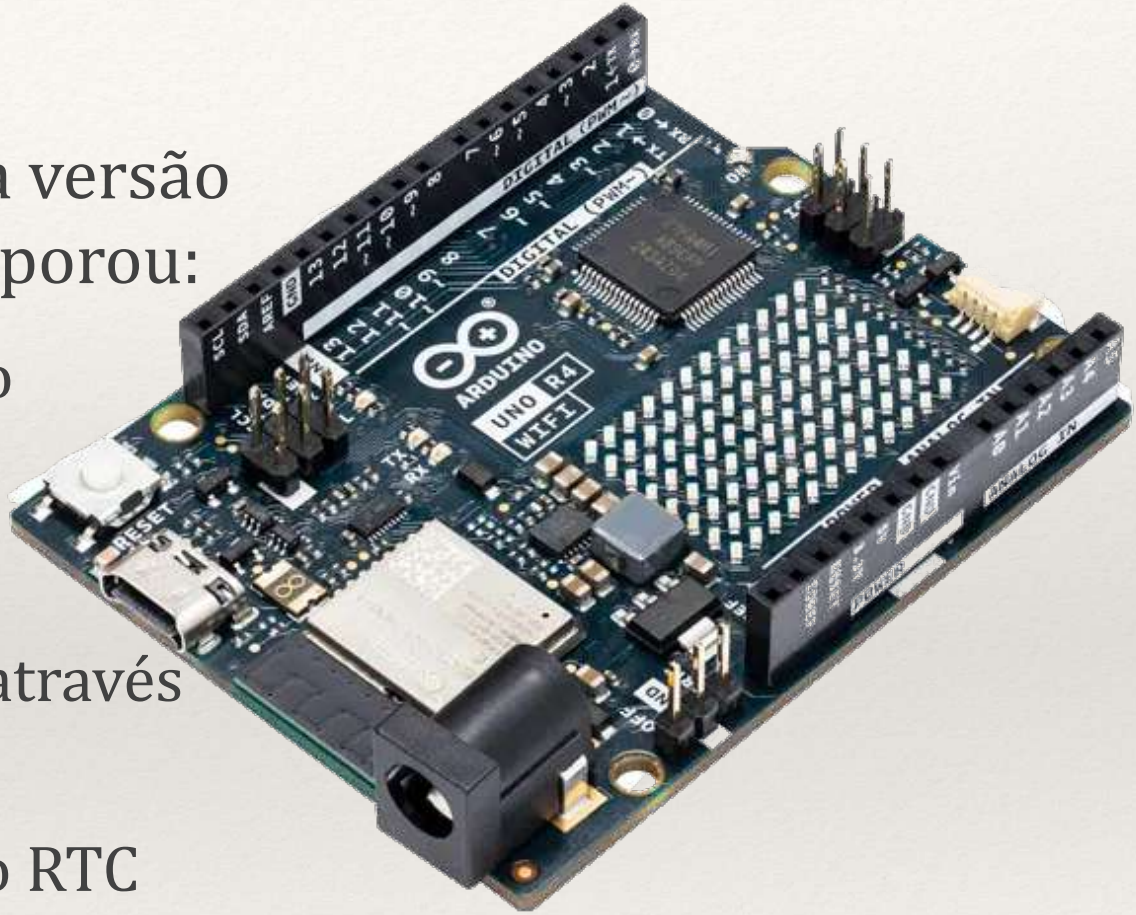
USB C com performance USB2.0



Arduíno Uno R4 Wi-Fi

Além de todos os recursos da versão Minima, a versão Wi-Fi incorporou:

- Módulo ESP32-S3, oferecendo suporte a BT e Wi-Fi;
- Matriz de 96 LEDs;
- Segunda porta I2C, acessível através de conector Qwiic;
- Conector para alimentação do RTC em separado.



Outras placas de Protótipo

Há outros "Arduínos":

Nano (*)	Mega (*)	LilyPad (*)
Uno R4 Minima	Uno R4 Wi-Fi	

Há também outras linhas, como:

NodeMCU (baseado no controlador ESP8266): baixo custo e recursos de comunicação

ESP32: baixo custo, recursos de comunicação avançados e alta capacidade

Teensy 4 (baseado no ARM Cortex-M7 de 600MHz): alta performance

Placas baseadas no **MSP430**, da Texas Instruments: baixo consumo

Placas baseadas no **STM32**: versatilidade

Conhecendo o ESP32

Desenvolvido pela Expressif

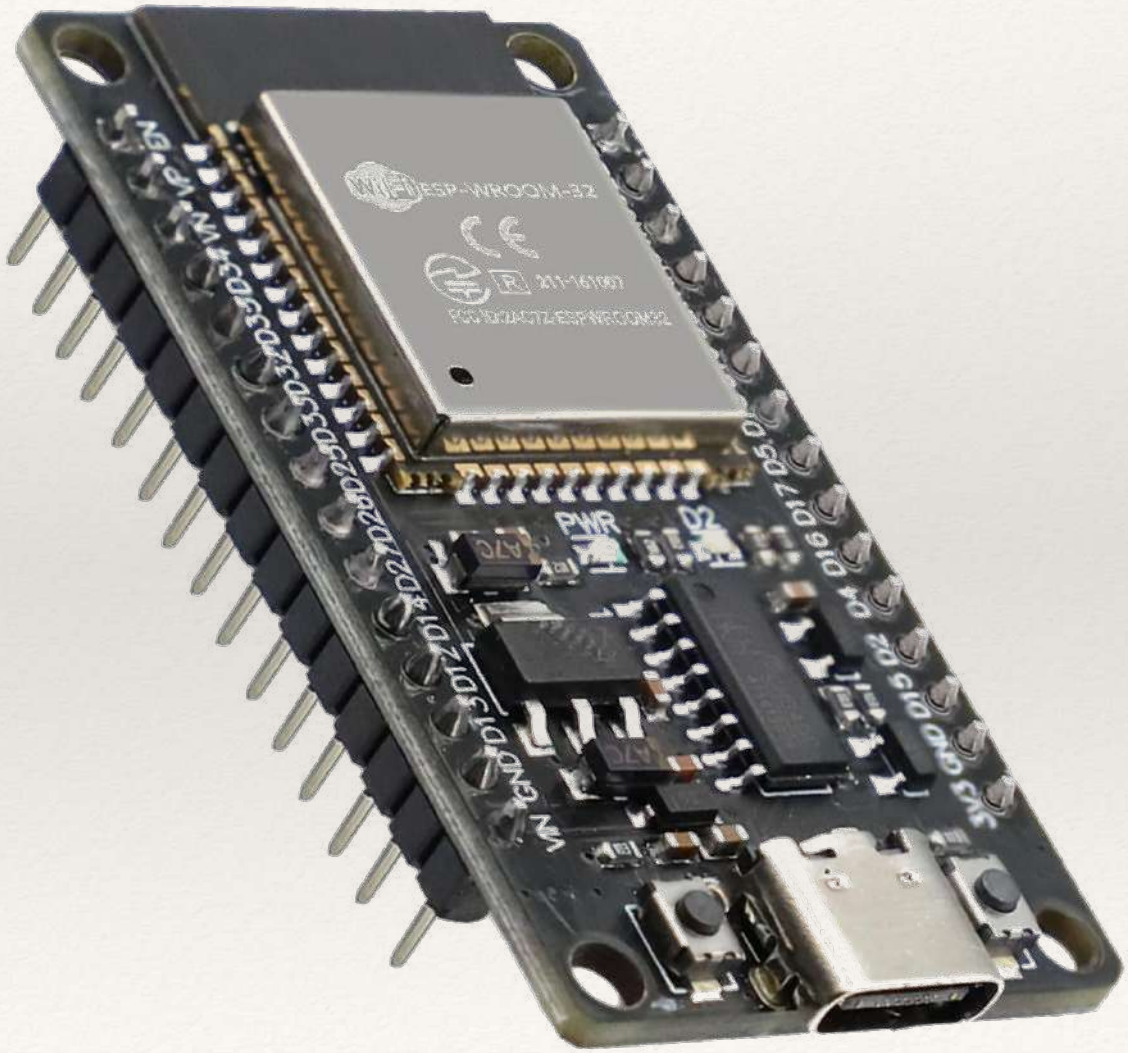
- Baixo custo (~US$ 6)
- Selecionei ESP32 WROOM 32D (são muitos)

Aspectos Técnicos:

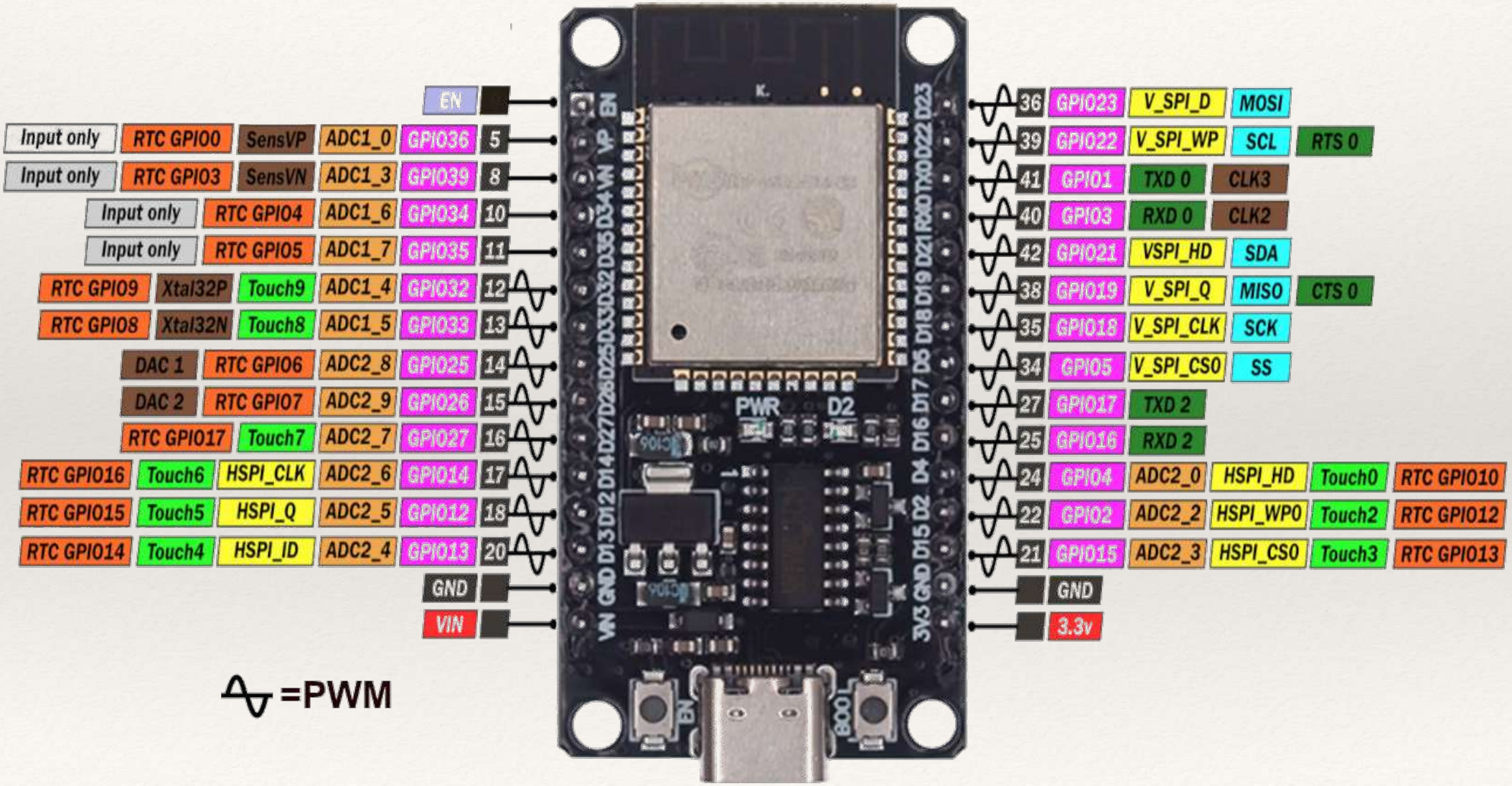
- Tensão de Operação: 3,3V
- Dimensões: 50 x 28mm
- Tensilica Xtensa Dual-Core 32-bit LX6
- 448KB ROM, 520KB SRAM, 4MB Flash
- Suporta Wi-Fi, Bluetooth
- Múltiplas interfaces I/O: ADCs, DACs, UART, SPI, I2C, PWM, entre outras.

40 GPIOs

- 14 completamente livres
- 4 apenas para leitura
- 1 apenas para saída
- 21 reservados, com uso limitado



ESP32 WROOM 32D (pinos)



Programando o Arduíno Uno

Software gratuito;

Programas se chamam "*sketchs*";

Duas partes:

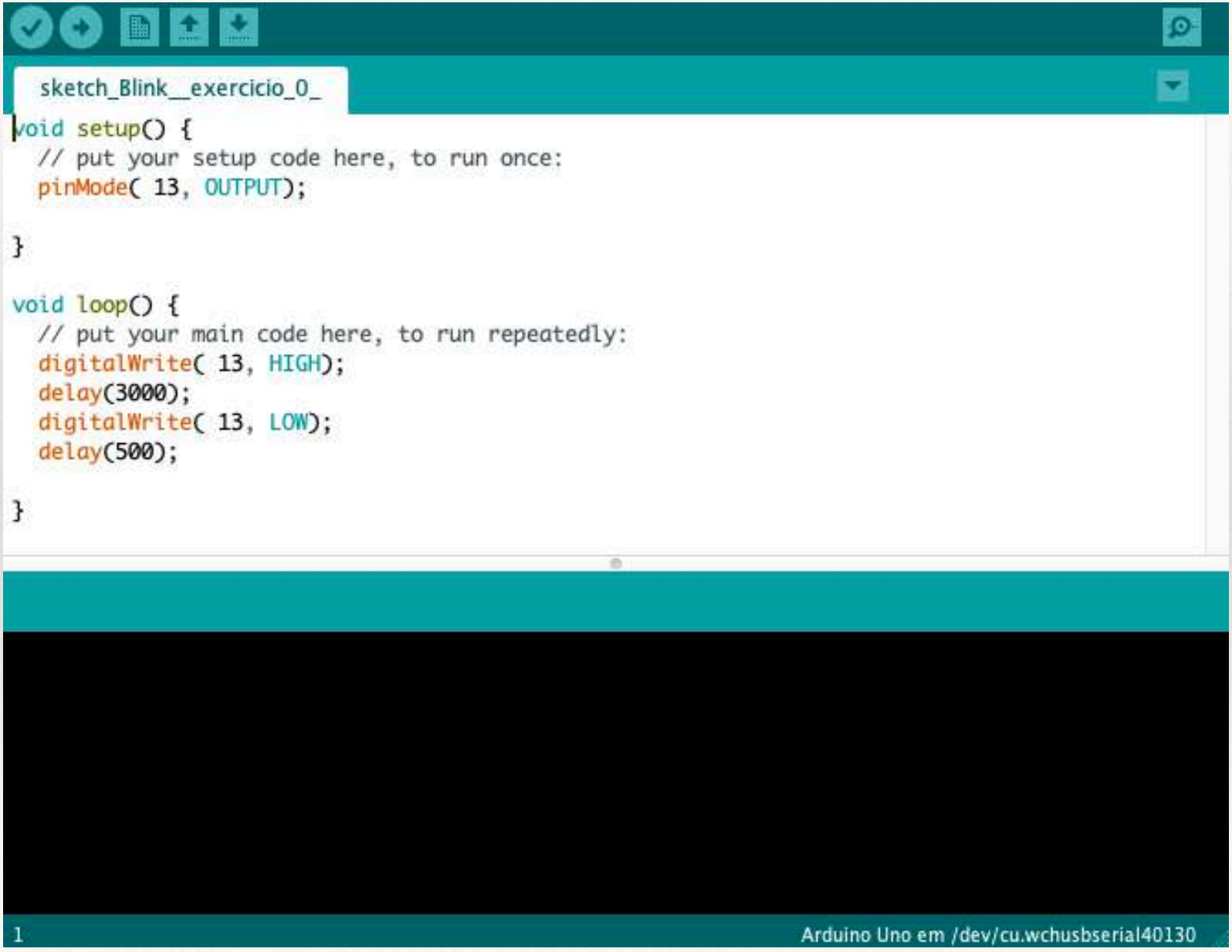
- setup() - executa apenas 1 vez;
- loop() - executa continuamente.

Após compilar, é necessário carregar na placa de protótipo, tipicamente utilizando a porta USB;

Este método é chamado de "compilação cruzada", e é adotado também para a programação de celulares, por exemplo.

* Curiosidade: o primeiro programa em um Arduíno não é o "Hello World", e sim o "Blink".

Apresentando o “Blink"



Programação - Estrutura

Estrutura Básica

loop( ) setup ( )

Estrutura Avançada

#define #include // /* ... */ { ... }

Controle

if ... else switch ... case do ... while while for

Operadores de Comparação

== != < > <= >=

Operadores Booleanos

! && ||

Programação - I/O

Digitais

digitalRead( ) digitalWrite ( ) pinMode()

Analógicas

analogRead() analogWrite() analogReference()

Comunicação

Serial.? Stream.? Keyboard.? Mouse.?

Tempo

delay() delayMicroseconds() micros() millis()

Avançadas

tone() shiftIn() shiftOut() pulseIn() pulseInLong() noTone()

Programação - Limitações

Não há Sistema Operacional

Recursos precisam ser oferecidos pelo programador. Isso implica que conceitos básicos, como multitarefa, por exemplo, não existem;

Ambiente de desenvolvimento

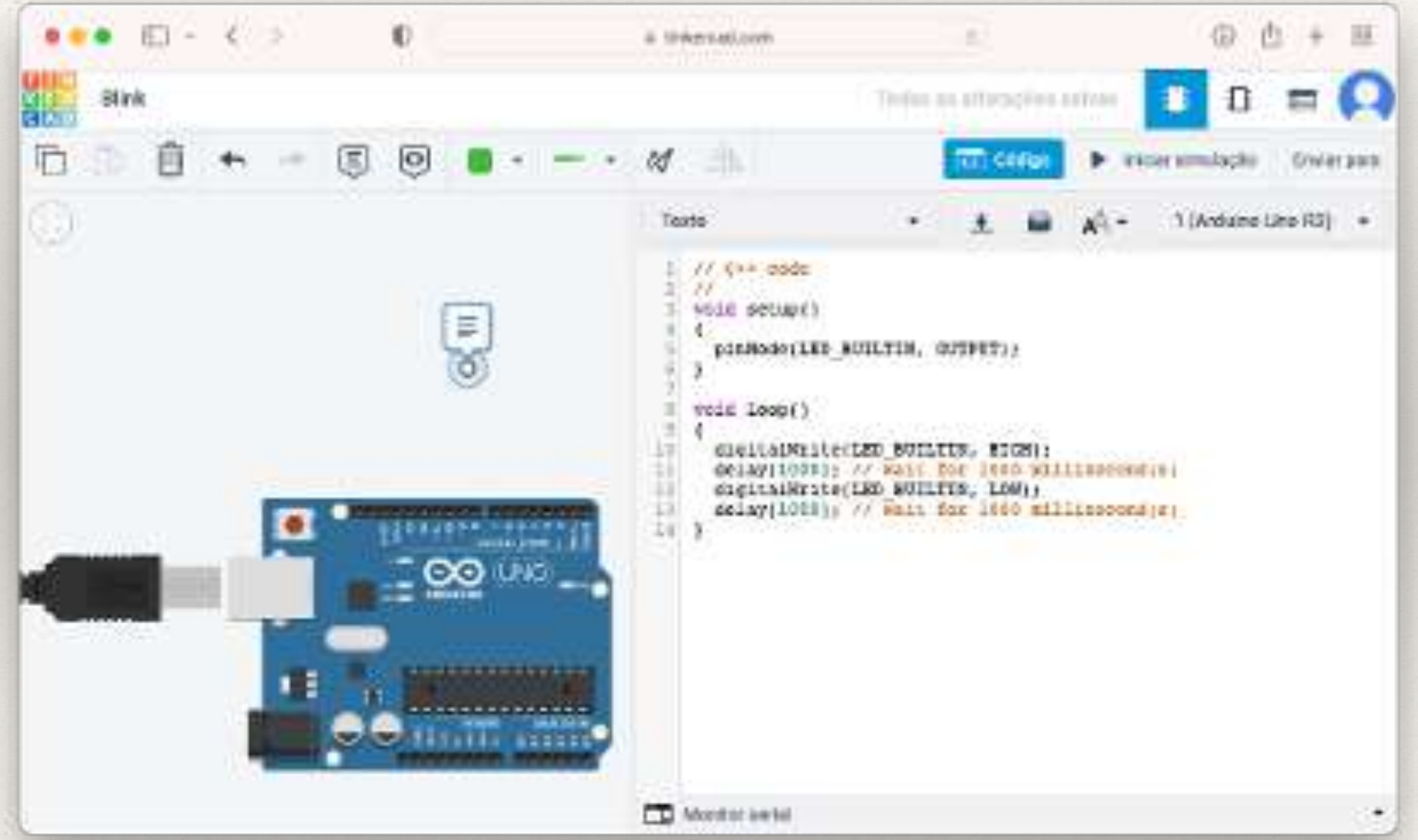
O ambiente é básico, e facilidades como ferramentas de debug, por exemplo, são inexistentes, ou primárias;

Limites de *hardware* (Arduíno Uno v3)

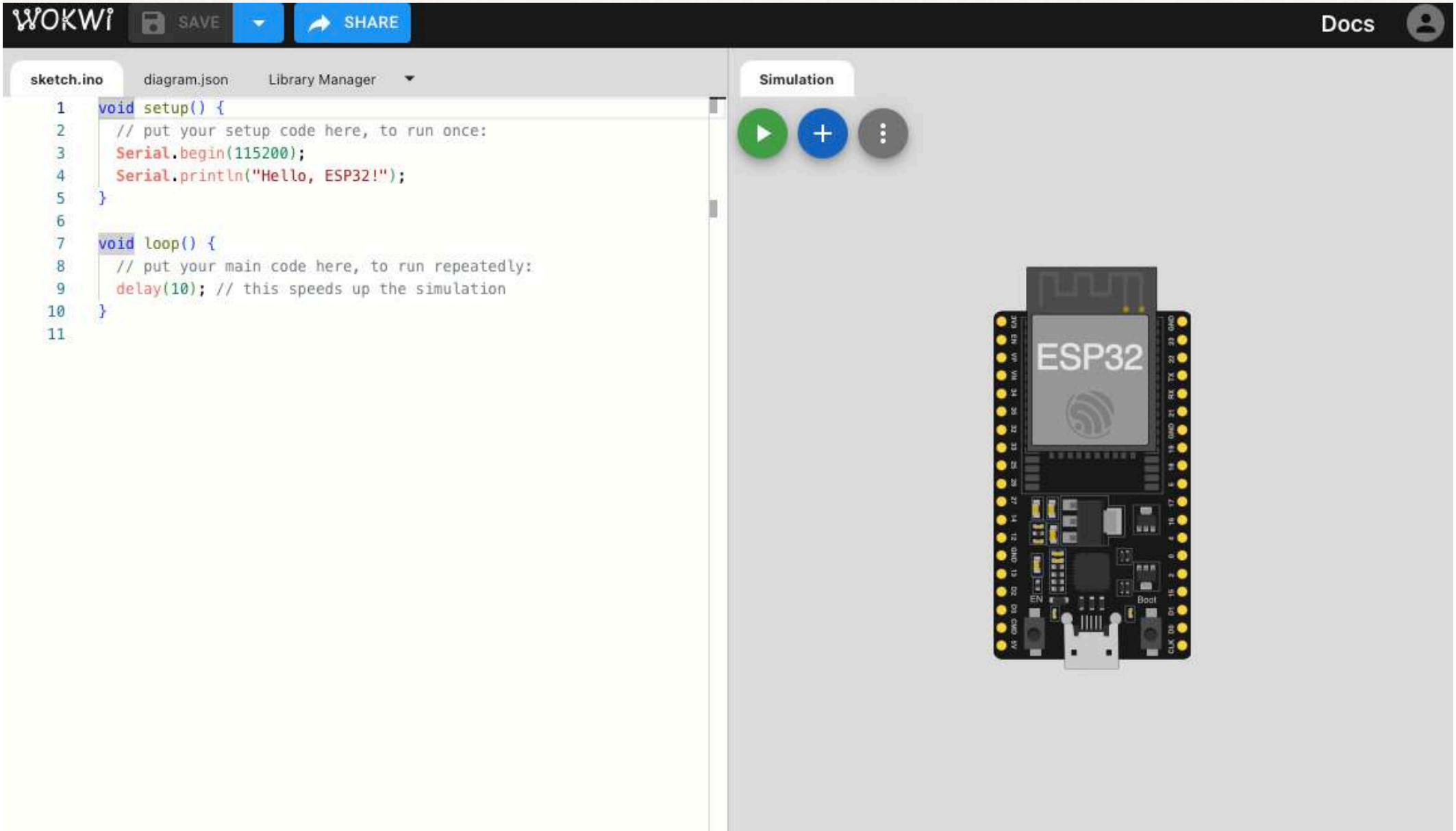
Performance limita (algumas) aplicações;

32KB de flash; 2KB RAM (500 pixels?); 1 KB EEPROM (100k ciclos de escrita)

Programação - Simuladores

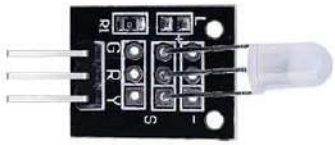


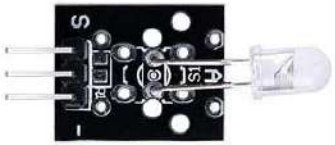
Programação - Simuladores



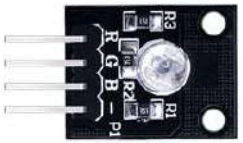
Dispositivos de Entrada e Saída

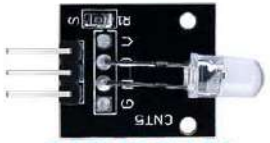

JOYSTICK

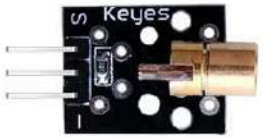

TWO-COLOR


IR EMISSION

MEMBRANE SWITCH

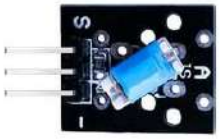

RGB LED


7 COLOR FLASH



LASER EMIT


SMD RGB


TILT-SWITCH

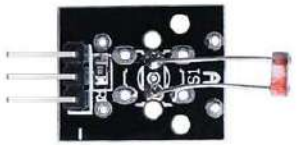
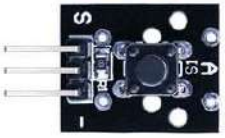
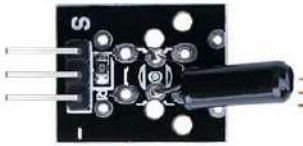


PHOTO-RESISTOR

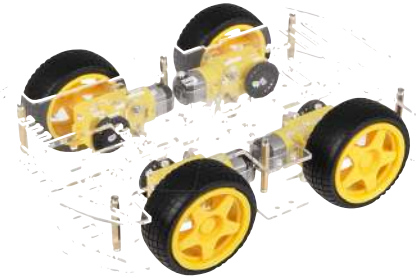
ULTRASONIC SENSOR

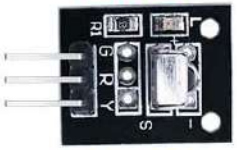

BUTTON

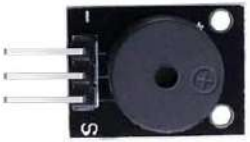
ACTIVE BUZZER


SHOCK


WATER LEVEL SENSOR

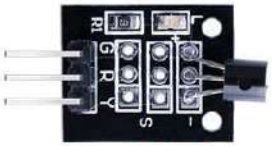



IR RECEIVER


PASSIVE BUZZER


DS 1307 RTC MODULE


LCD 1602 MODULE


18B20 TEMP


ROTARY ENCODER


RELAY


HC-SR501


GY-521


POWER SUPPLY

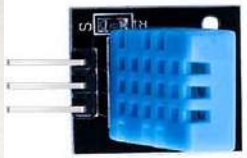
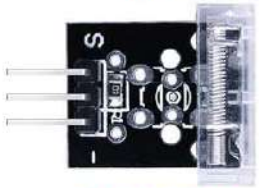

TEMP AND HUMIDITY


PHOTO-INTERRUPTER


TAP MODULE

AVOIDANCE

TRACKING


MAGNETIC SPRING


DIGITAL TEMPERATURE


FLAME


LINEAR HALL


BIG SOUND


METAL TOUCH

SMALL SOUND

Dispositivos de Saída

LEDs

- Saída analógica típica para testes;
- Admite modulação via PWM;
- Vermelho, Verde, Amarelo, Azul, Branco e RGB (3 saídas).

Dispositivos de Entrada

Chaves e Botões;

Teclados (ex. teclado de membrana);

Sensores digitais (chuva, presença e movimento, chave magnética, gases, sensor de digital etc);

Sensores analógicos (potenciômetro, umidade, pressão, iluminação, temperatura, ultrasom, tensão, corrente, campo magnético etc);

Sensores de pulsos (fluxo d'água, contadores etc);

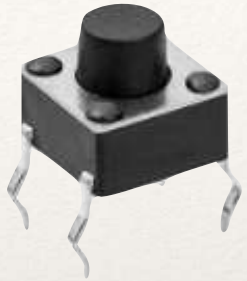
Dispositivos especiais (giroscópio, acelerômetro, RTC).

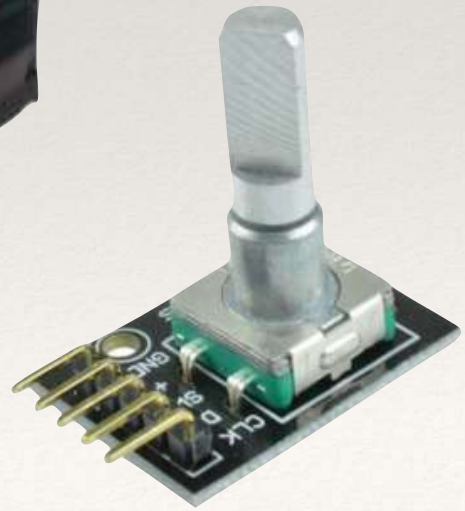
Dispositivos de Entrada

- Chaves e Botões
  - Método básico de entrada;
  - Chaves momentâneas, fixas e *encoders*;
  - *Pull Up* ou *Pull Down*
  - Efeito *Bouncing*
    - *Debouncing* por SW ou HW

Dispositivos de Entrada

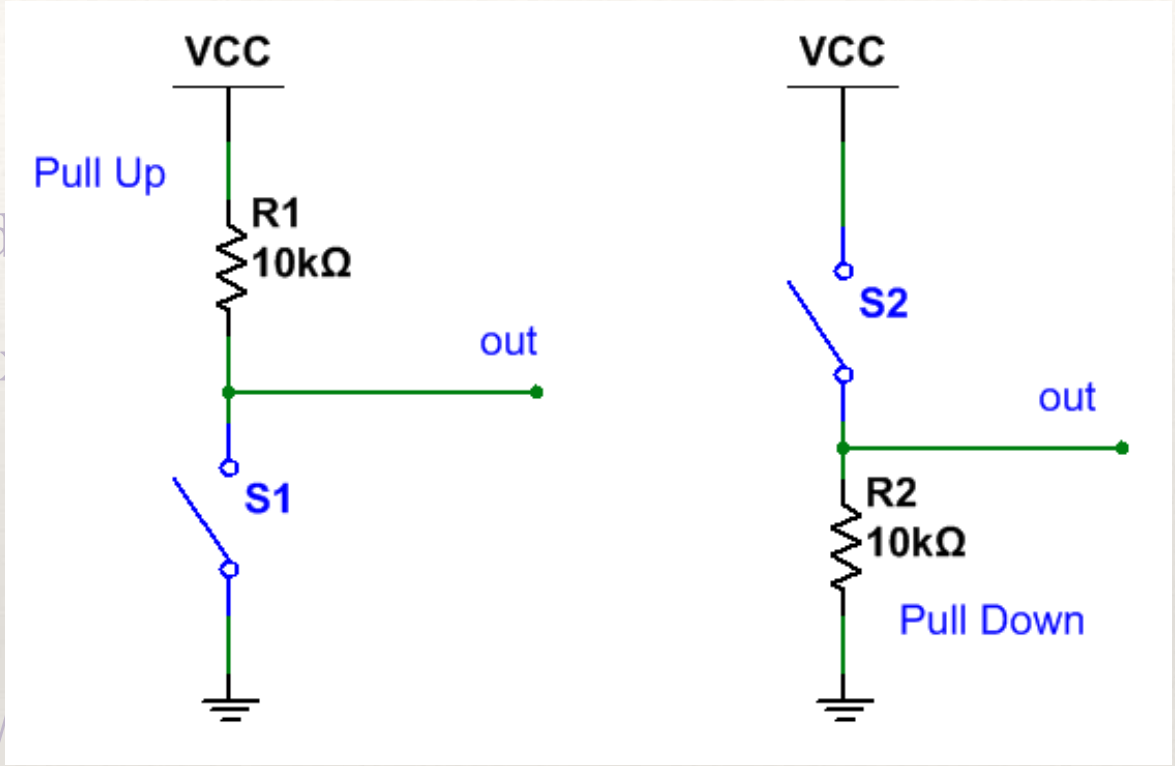
- Chaves e Botões
  - Método básico de entrada;
  - Chaves momentâneas, fixas e *encoders*;
  - *Pull Up* ou *Pull Down*
  - Efeito *Bouncing*
    - *Debouncing* por SW ou HW





Dispositivos de Entrada

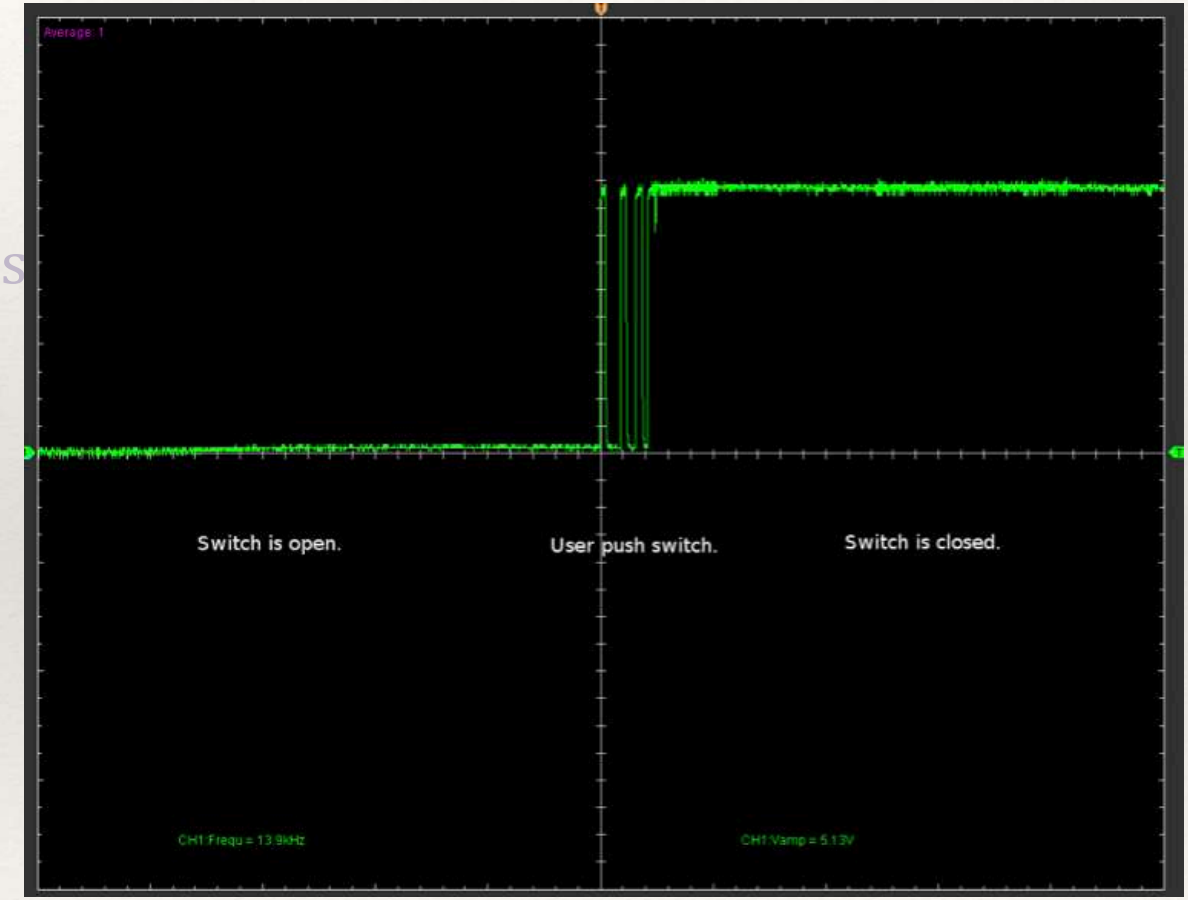
- Chaves e Botões
  - Método básico de entrad
  - Chaves momentâneas, fi
  - *Pull Up* e *Pull Down*
  - Efeito *Bouncing*
    - *Debouncing* por SW ou HV



Dispositivos de Entrada

Chaves e Botões

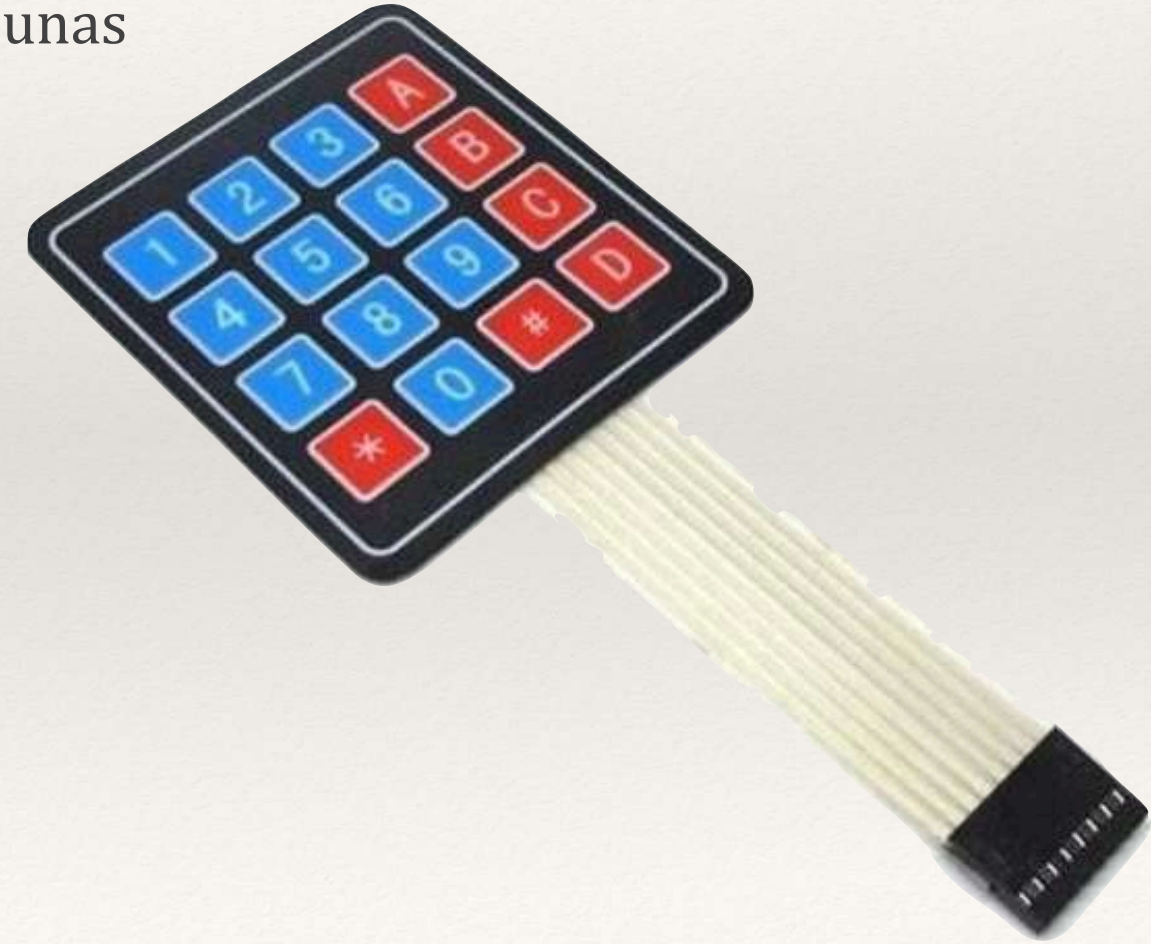
- Método básico de entrada;
- Chaves momentâneas, fixas
- *Pull Up* ou *Pull Down*
- Efeito *Bouncing*
  - *Debouncing*: SW ou HW



Dispositivos de Entrada

Teclados (ex. teclado de membrana)

- Cruzamento de Linhas X Colunas
- Varredura via SW
- Analisar combinações?

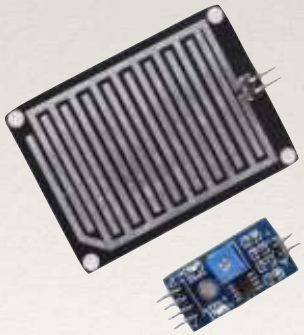


Dispositivos de Entrada

Sensores digitais (chuva, presença e movimento, chave magnética, gases, leitura de digitais etc)

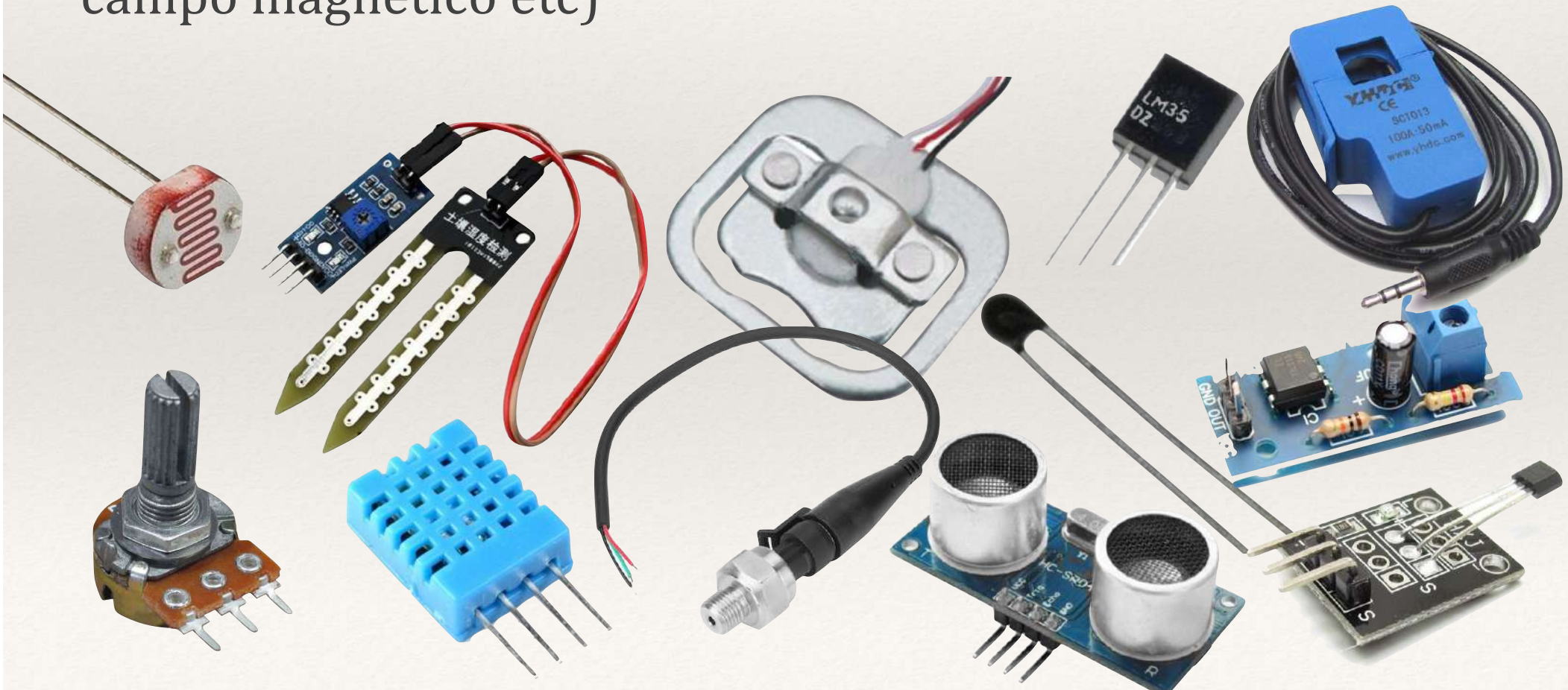
Funcionam como chaves (sem *bouncing*);

Alguns permitem ajuste da sensibilidade para ativação;



Dispositivos de Entrada

Sensores analógicos (iluminação, potenciômetro, umidade, peso, pressão, temperatura, ultrasom, corrente, tensão, campo magnético etc)



Sensores “medem” uma tensão elétrica

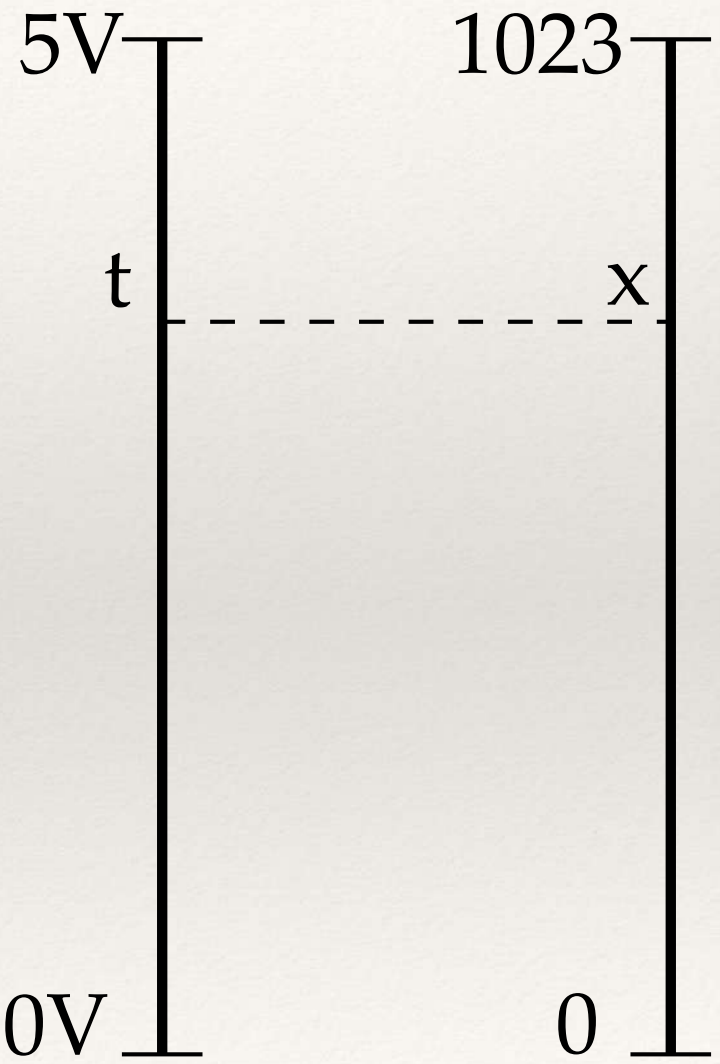
Tensão elétrica é injetada em uma porta analógica do microcontrolador

- Tensão elétrica precisa estar dentro dos limites do microcontrolador;
- No Arduíno, pode estar entre 0 e 5V.

Na porta analógica, a tensão é convertida para um número inteiro pelo conversor analógico/digital interno;

Número é determinado pela quantidade de bits do conversor analógico digital

- No Arduíno, o conversor é de 10bits
- $2^{10} = 1024$



A tensão é convertida para um número

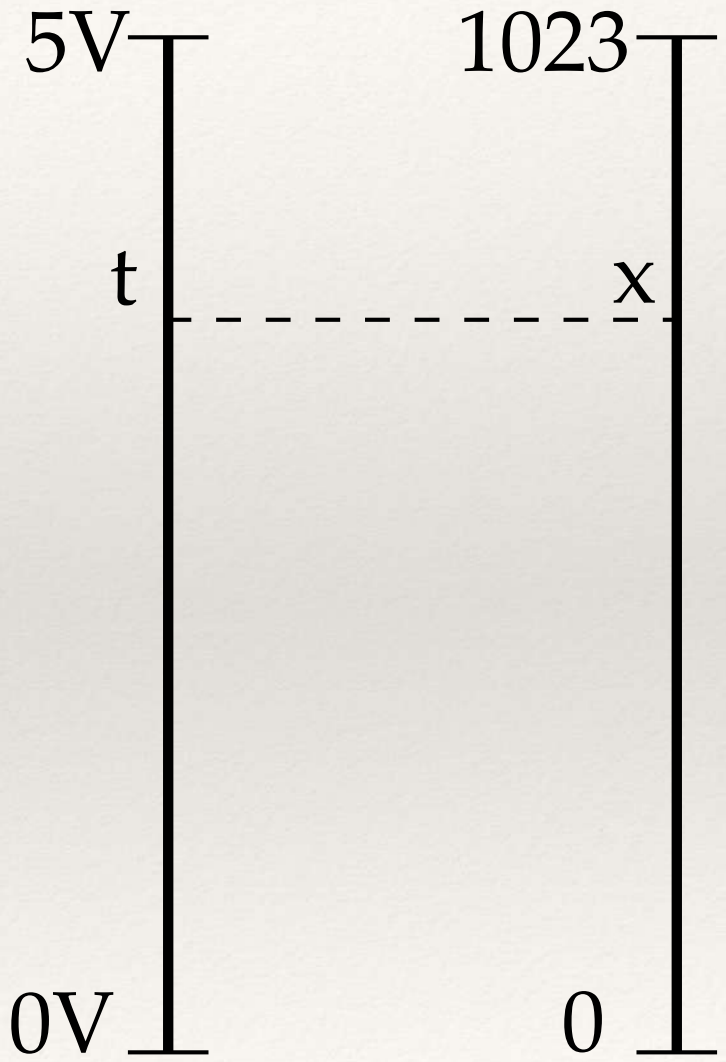
Medições são proporcionais graças à linearidade do sensor;

Conversão é básica, por regra de 3

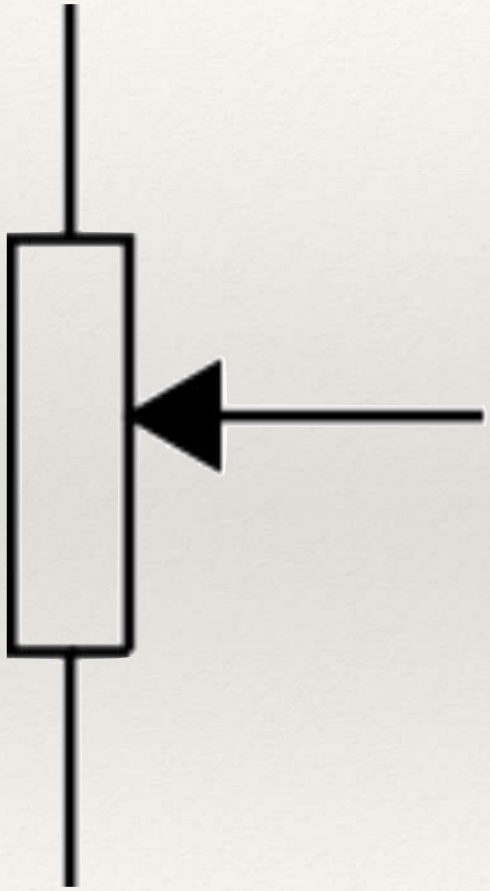
$$\frac{t - 0}{5 - 0} = \frac{x - 0}{1023 - 0}$$

$$\frac{t}{5} = \frac{x}{1023}$$

$$t = \frac{5 \cdot x}{1023}$$



Entrada: Divisor de Tensão



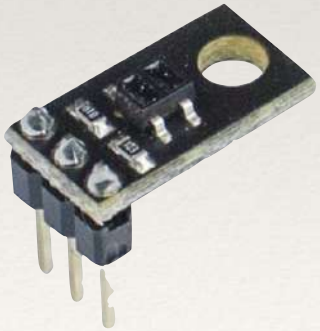
Com a circulação de corrente pelo potenciômetro, a tensão no cursor varia de acordo com a posição do mesmo;

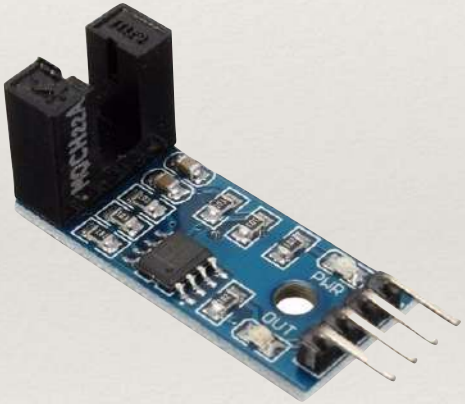
A posição do cursor determina, então, a tensão obtida;

O potenciômetro funciona como um sensor de posição angular, e assim está sendo usado no projeto.

Dispositivos de Entrada

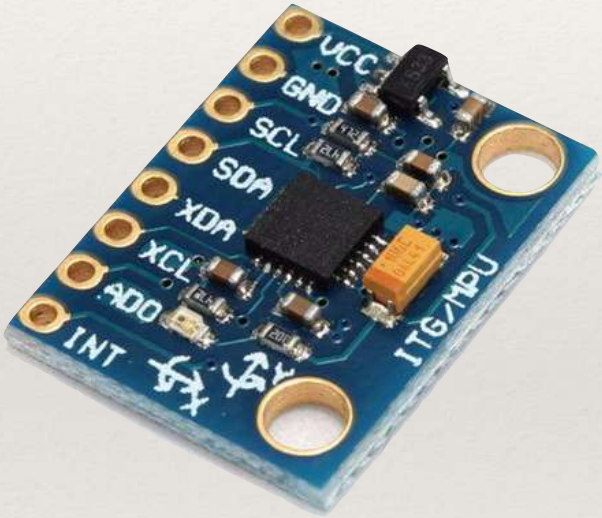
Sensores de pulsos (fluxo d’água, contadores etc)

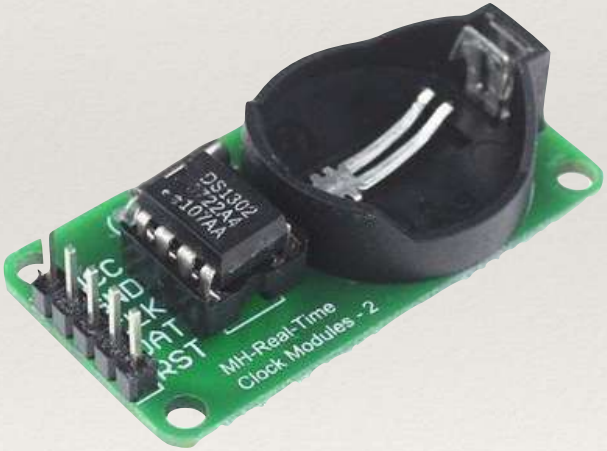




Dispositivos de Entrada

Dispositivos especiais (giroscópio, acelerômetro, RTC)



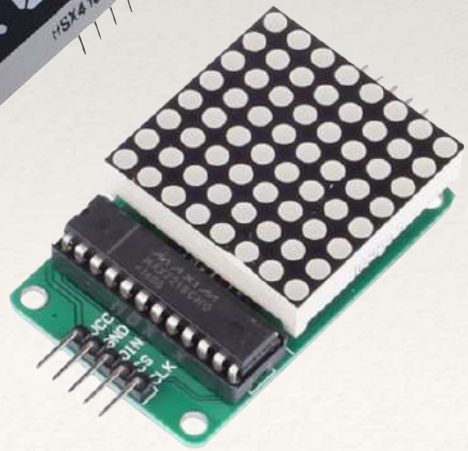


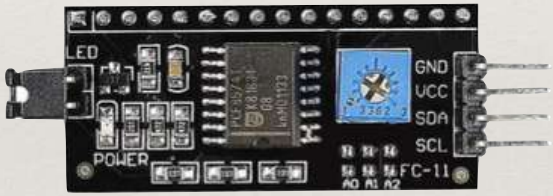
Dispositivos de Saída

Displays (7 segm., matriciais, LCD, TFT, OLED)

Interface convencional ou de rede (i2C, por exemplo);

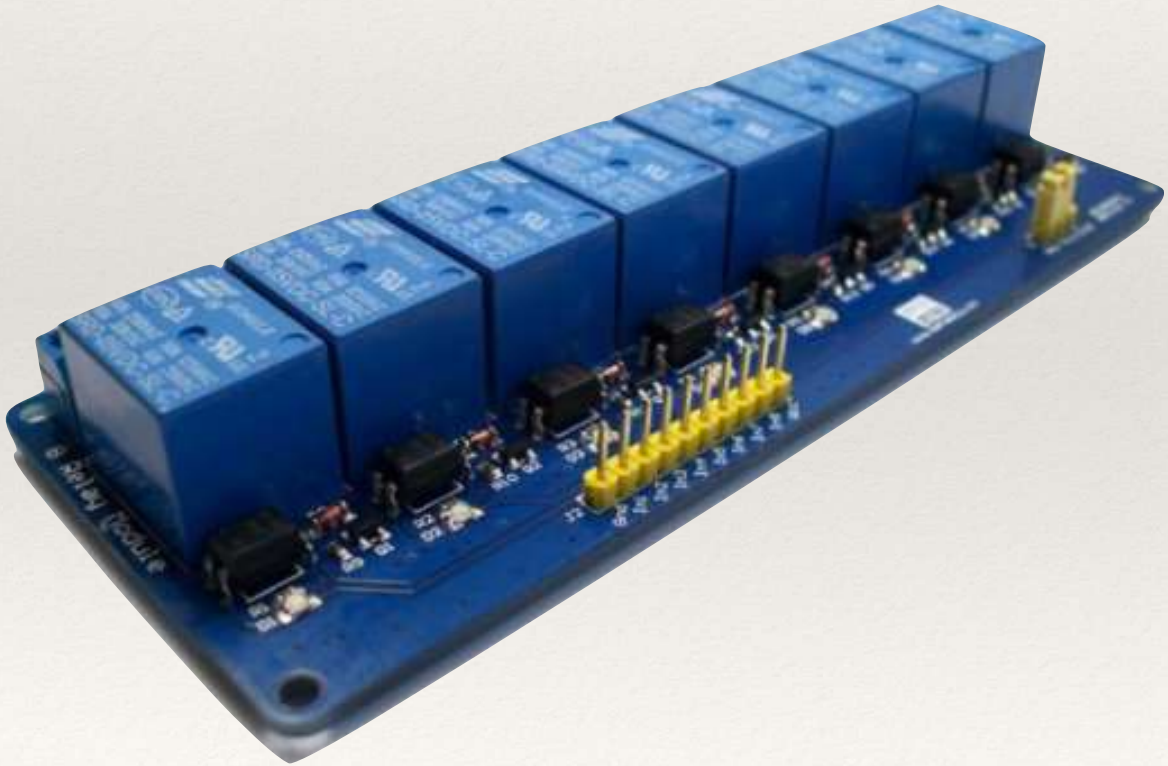
Cátodo ou Anodo comum;

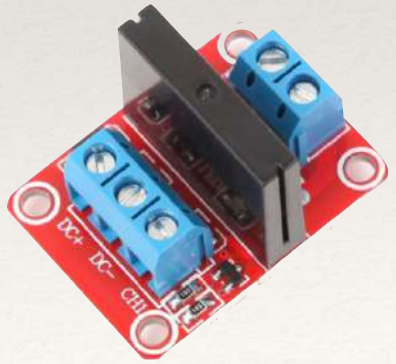




Dispositivos de Saída

Relés (acionam cargas diversas)





Dispositivos de Saída

Servos, motores convencionais e de passo

Dispositivos Entrada/Saída

RS-232, USB;

Ethernet cabeada;

Infravermelho;

RF, Bluetooth, Wi-Fi, RFiD;

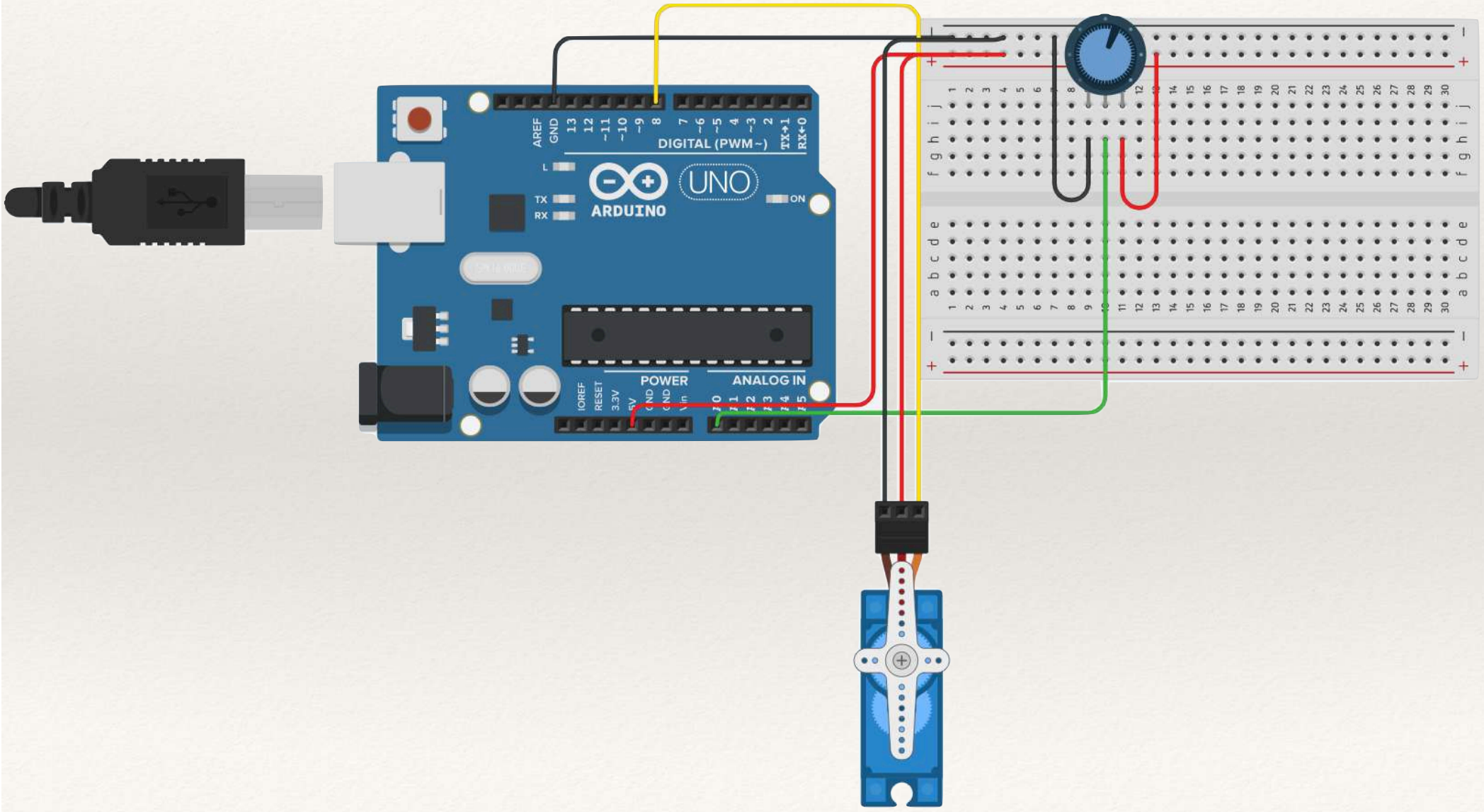
Módulos de Memória externa.

Servo Motor Angular

Além da alimentação, o terceiro pino deve ser interligado diretamente a uma saída digital do controlador;

A biblioteca do dispositivo permite acionar diretamente o servo para um ângulo específico.

Projeto: Seguidor de Potenciômento



Projeto: Seguidor Potenciômento

```
#include <Servo.h>
Servo servo1;
const int pinServo=8;
const int POT=0;
int valpot = 0;
int angleServo = 0;

void setup() {
  servo1.attach(pinServo);
}
void loop() {
  valpot = analogRead(POT);

  angleServo=map(valpot,0,1023,0,180);

  servo1.write(angleServo);
  delay(15);
}
```