Professor ***Marco Câmara***

Sistemas Embarcados

UCSAL 2024-02

Aula 01

Definição

Um **sistema embarcado** é um sistema computadorizado construído para uma aplicação específica. Por isto mesmo, normalmente não suporta itens que não estejam relacionados à execução da aplicação. Seu *hardware* e *software* têm limitações.

O *hardware* normalmente é mais lento para reduzir o consumo de energia, e oferece apenas recursos mínimos essenciais para suporte ao seu conjunto específico de periféricos.

O *software* tipicamente precisa ter comportamento determinístico (sempre oferece o mesmo tempo de execução), e, em certos casos, em tempo real (reagindo sempre de forma imediata a eventos). Em alguns sistemas, o *software* precisa ser tolerante a falhas, com degradação controlada (considere por exemplo um satélite ou um marcador preso ao corpo de uma baleia). Em outros, o *software* precisa comunicar o erro imediatamente após o primeiro sinal de falha(considere um monitor cardíaco).

[**Making Embedded Systems** - White, Elecia]

Definição

Um **sistema embarcado** é um sistema eletrônico microprocessado que, após ser programado, possui uma função específica que geralmente não pode ser alterada. Uma impressora, por exemplo, mesmo possuindo um processador que poderia ser utilizado para qualquer tipo de atividade, tem sua funcionalidade restrita apenas à impressão de páginas. Um computador de propósito geral, no entanto, pode ser utilizado num instante como um ambiente de entretenimento, em outro como estação de trabalho, ou até mesmo um telefone.

[**Programação de Sistemas Embarcados** - Almeida, Rodrigo M A]

Definição

Um **sistema embarcado** é um sistema eletrônico microprocessado, completamente encapsulado, dedicado ao dispositivo ou sistema que ele controla. Diferentemente de computadores de propósito geral, como o computador pessoal, um sistema embarcado realiza um conjunto de tarefas predefinidas, geralmente com requisitos específicos. Já que o sistema é dedicado a tarefas específicas, através de engenharia pode-se otimizar o projeto reduzindo tamanho, recursos e custo do produto.

[Wikipedia]

Definição (+ algumas)

Interface com o usuário

Sistemas embarcados podem operar sem nenhuma ou muito pouca interação com seus usuários.

Omnipresentes

Praticamente todo equipamento elétrico, mecânico e químico atual é um sistema embarcado;

Carregadores de celular, eletrodomésticos, semáforos, automóveis, aviões, motores industriais, equipamentos médicos etc.

Diversos níveis de complexidade e custo

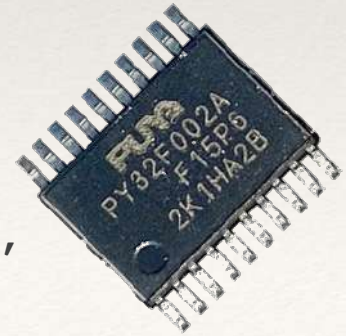
Sistemas embarcados controlam desde um eletrodoméstico simples, como um liquidificador, até um foguete recuperável da SpaceX.

Complexidade e Custo

Na área de TI, a variedade é comum em qualquer tipo de dispositivo

- Computadores possuem grande variação de complexidade e custo;
- No entanto, sistemas embarcados oferecem um espectro ainda maior.

XQR5VFX130, utilizado na indústria aeroespacial, custa **US$ 40,000**



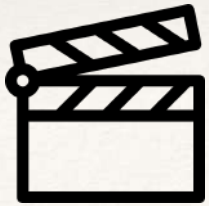
PY32F002A, utilizado em sistemas simples, custa **US$ 0.08**

40000 / 0,08 = 500.000 vezes mais caro

Motivação e história

Sempre houve demanda pela automação de tarefas repetitivas, por diversos motivos. Entre eles:

- *Aumento da performance*
- *Redução de Custos*
- *Padronização*
- *Segurança*



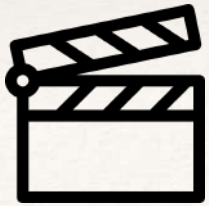
Motivação e história

Sempre houve demanda pela automação de tarefas repetitivas, por diversos motivos. Entre eles:

- *Aumento da performance*
- *Redução de Custos*
- *Padronização*
- *Segurança*

A revolução industrial praticamente foi definida por esta demanda

- *Autômatos complexos serviram de inspiração para as máquinas de tear*
- *Os autômatos durante algum tempo eram dispositivos mecânicos: roda d'água, relógio, semáforos, e até robôs !*



Motivação e história

Sempre houve demanda pela automação de tarefas repetitivas, por diversos motivos. Entre eles:

- *Aumento da performance*
- *Redução de Custos*
- *Padronização*
- *Segurança*

A revolução industrial praticamente foi definida por esta demanda

- *Autômatos complexos serviram de inspiração para as máquinas de tear*
- *Os autômatos durante algum tempo eram dispositivos mecânicos: roda d'água, relógio, semáforos, e até robôs !*

Dispositivos mecânicos acabaram sendo substituídos por eletrônicos, e depois digitais, e finalmente microprocessados

- *Redução de custos pela escala de produção;*
- *Maior simplicidade de projeto, fabricação e manutenção.*

Motivação e história

Sempre houve demanda pela automação de tarefas repetitivas, por diversos motivos. Entre eles:

Aumento da performance

Padronização

A revolução industrial pra

Autômatos complexos servir

Os autômatos durante algu

relógio, semáforos, e até rob

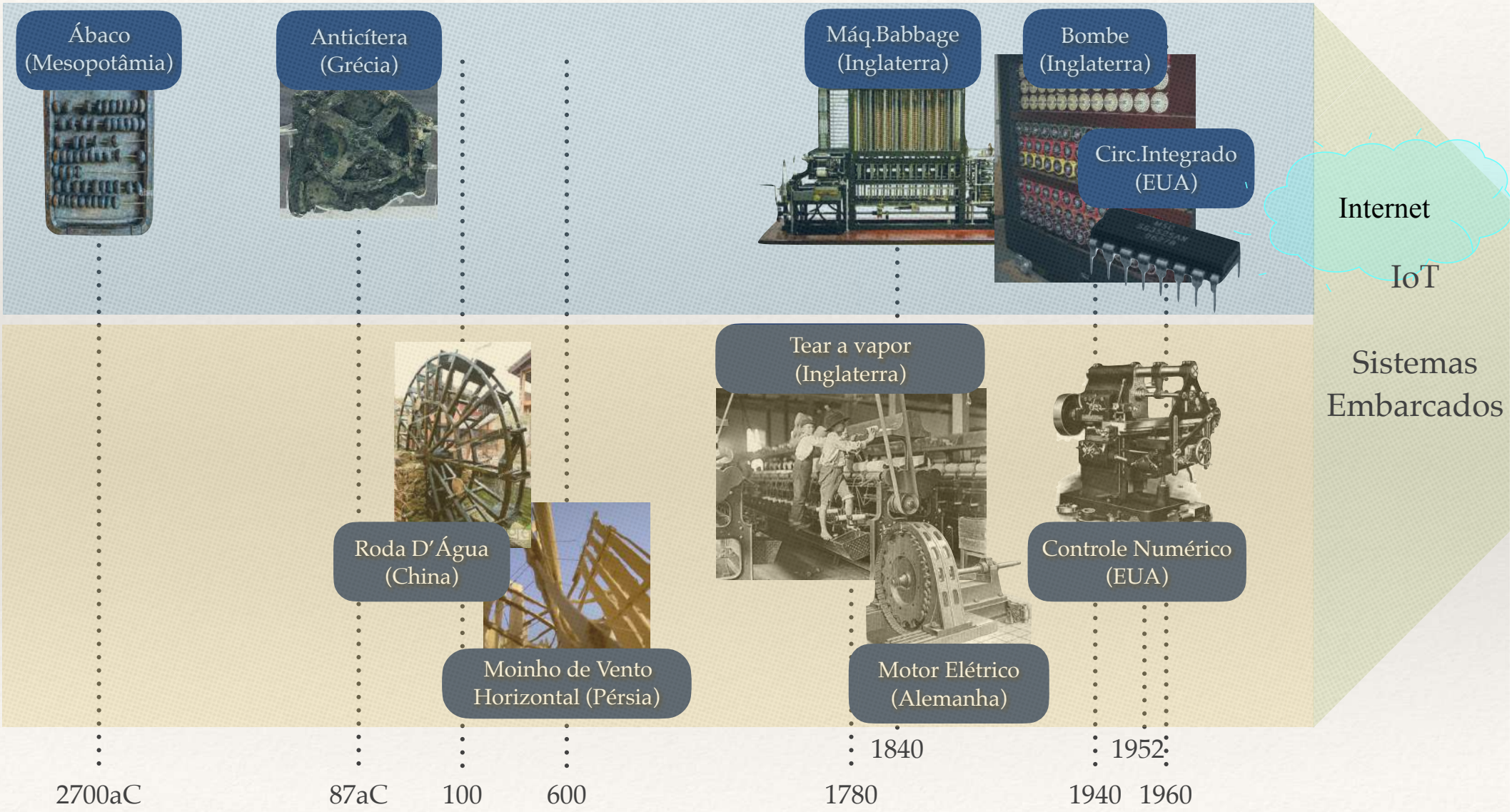
Dispositivos mecânicos aca
eletrônicos, e depois digitais, e finalmente microprocessados

Redução de custos pela escala de produção;

Maior simplicidade de projeto, fabricação e manutenção.

Alguns autores adotam uma classificação dos sistemas embarcados em função da arquitetura dos processadores (pequena ou média escala, ou “complexa). Outros usam a “geração" dos dispositivos (1ª a 4ª gerações). Como este tipo de classificação inevitavelmente ficará defasada à medida em que evoluem os dispositivos, não adotaremos esta estratégia.

Um pouco de história



Eletrônica na Automação

Os benefícios da eletricidade foram rapidamente aproveitados

- Motores, válvulas, solenóides;
- Botões, Relês, Chaves Magnéticas;
- Circuitos elétricos.

Circuitos eletrônicos analógicos foram o próximo passo

- Automação “inteligente”;
- Maior confiabilidade, menor ocupação de espaço etc.

Chegada da eletrônica digital e a Lógica Fixa

- Portas lógicas permitiam lógicas complexas, e até programação sequencial;
- Comportamento mais estável, obedecendo à lógica digital e binária;

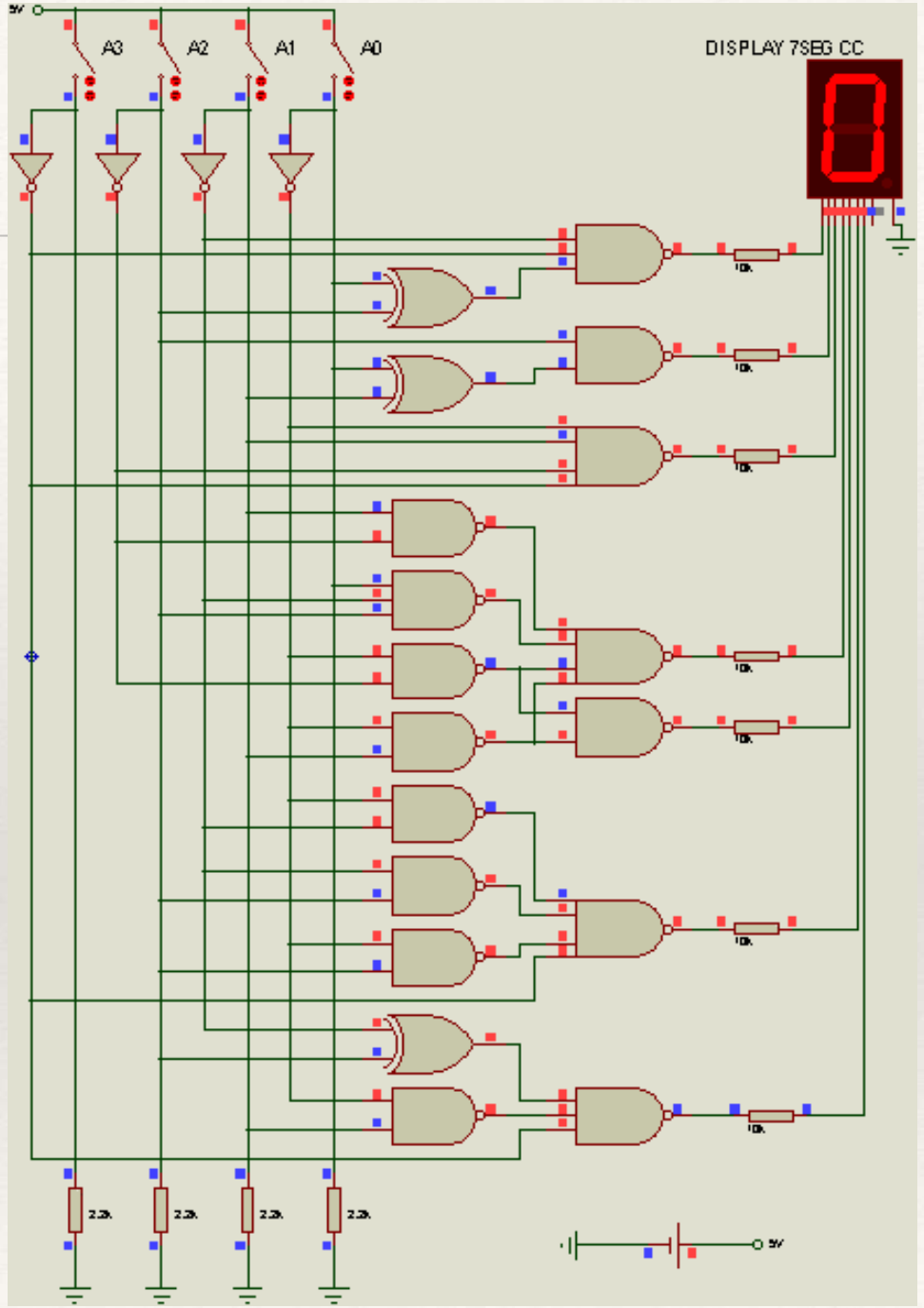
Lógica Fixa

Funcionalidade Digital

- É virtualmente possível implementar qualquer lógica específica através de projeto combinacional/sequencial;
- Pode-se obter níveis elevados de performance.

Pontos negativos

- Baixa flexibilidade e portabilidade;
- Complexidade gera custos elevados;
  - Projeto, Implementação e Manutenção
- Tipicamente tem dimensões maiores.



Lógica Fixa (implementação)

Discreta

- Operações lógicas básicas (E, OU, XOR etc);
- Componentes eletrônicos analógicos complementares.

Integrada

- Funcionalidade implementada totalmente, ou em grande parte, em circuito integrado específico;
- Pode utilizar inclusive circuitos digitais programáveis, como FPGA (*Field Programmable Gate Array*).

Sistemas Microprocessados

Flexibilidade

- Lógica implementada por SW;
- Um único microcontrolador admite infinitos projetos;
- Recursos embutidos permitem novas funcionalidades talvez inviáveis em sistemas de Lógica Fixa;
- Permitem atualizações pós-venda.

Baixo custo

- Não é raro que microcontroladores substituam dezenas de componentes discretos;
- Consumo energético e espaço ocupado são reduzidos.

Sistemas Microprocessados ⇒ SISTEMAS EMBARCADOS. ou "embutidos"

A História mais recente ...

Microprocessadores

1968: surge a Intel.

4 bits: Intel 4004, 4040

8 bits: Intel 8008, 8080, 8085

16 bits: Intel 8086, 8088, 80186 & 80188, 80286

32 bits: 80386, 80486, Intel Pentium, Pentium Pro, II, II xeon, III, IV, Dual Core

64 bits: Intel Core 2, i3, i5, i7, i9

Microcontroladores

1971: TMS1000, da Texas

1976: Intel 8048

1980: Intel 8051

1990 ➠ : surgiram diversos fornecedores (Atmel, Microchip, Expressif etc)

Classificação de Sistemas embarcados

Autônomos ou “isolados"

Em rede;

De tempo real;

Móveis.

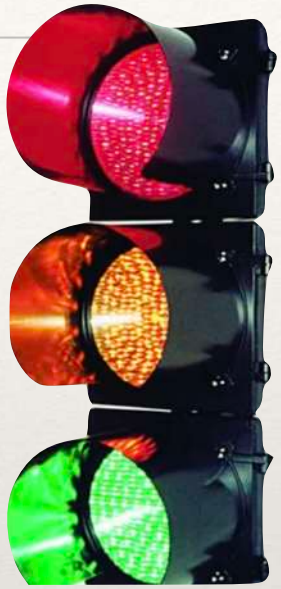
* **As classificações não são mutuamente exclusivas**

Classificação de Sistemas embarcados

Autônomos ou “isolados"

São independentes, não precisam de servidores ou acesso à internet;

Recursos de rede, quando existem, são locais, para acesso a sensores, atuadores, ou a sub-sistemas.



Em rede;

De tempo real;

Móveis.

Classificação de Sistemas embarcados

Autônomos ou “isolados";

Em rede

Dependem de recursos externos, seja como servidores, ou mesmo para comunicação com outros sistemas independentes;

Incorporam a **IoT (Internet das Coisas)**, que devido à sua abrangência e diversidade, deve-se tratar como um tema à parte;

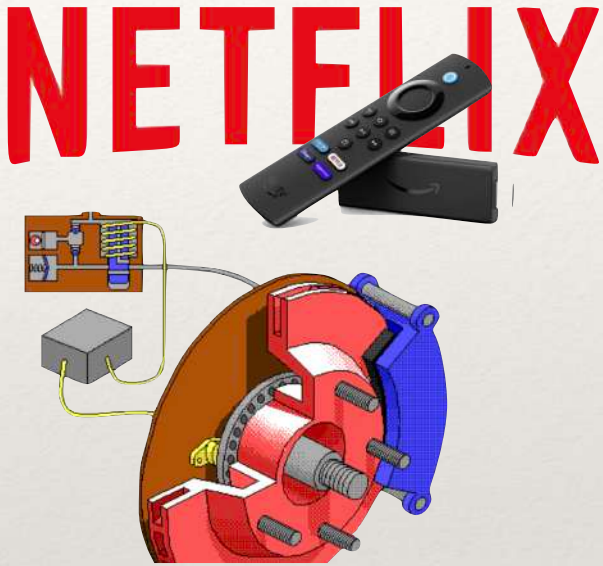
Representam um desafio para a segurança.

De tempo real;

Classificação de Sistemas embarcados

Autônomos ou “isolados”;

Em rede;



De tempo real

Precisam obedecer a requisitos de tempo de resposta:

Sistemas *Soft Real Time* operam mesmo diante de falhas na obediência a estes requisitos;

Sistemas *Hard Real* *Time* têm requisitos rígidos, que não podem ser desobedecidos.

Móveis.

Classificação de Sistemas embarcados

Autônomos ou “isolados”;

Em rede;

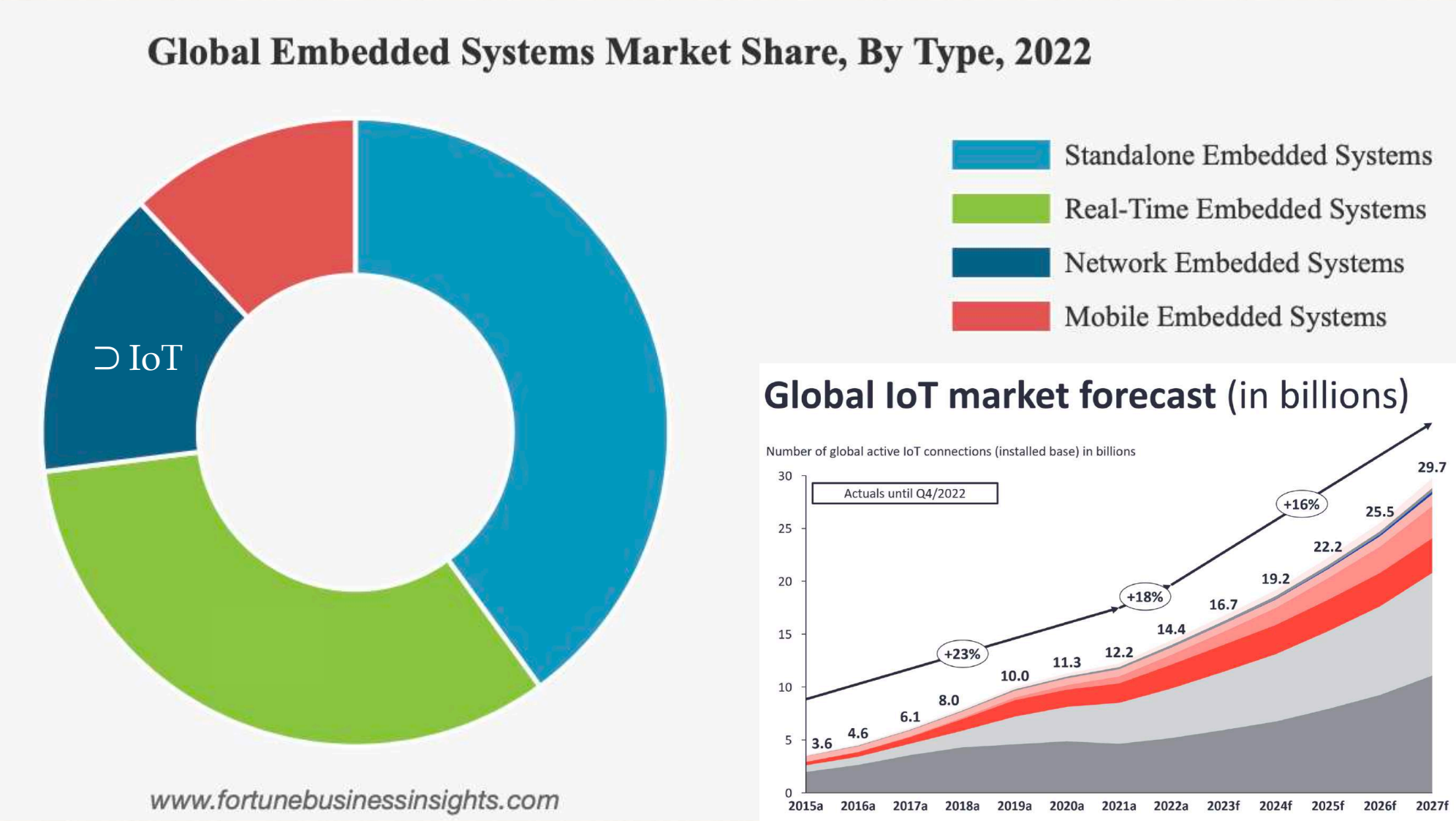
De tempo real;

Móveis

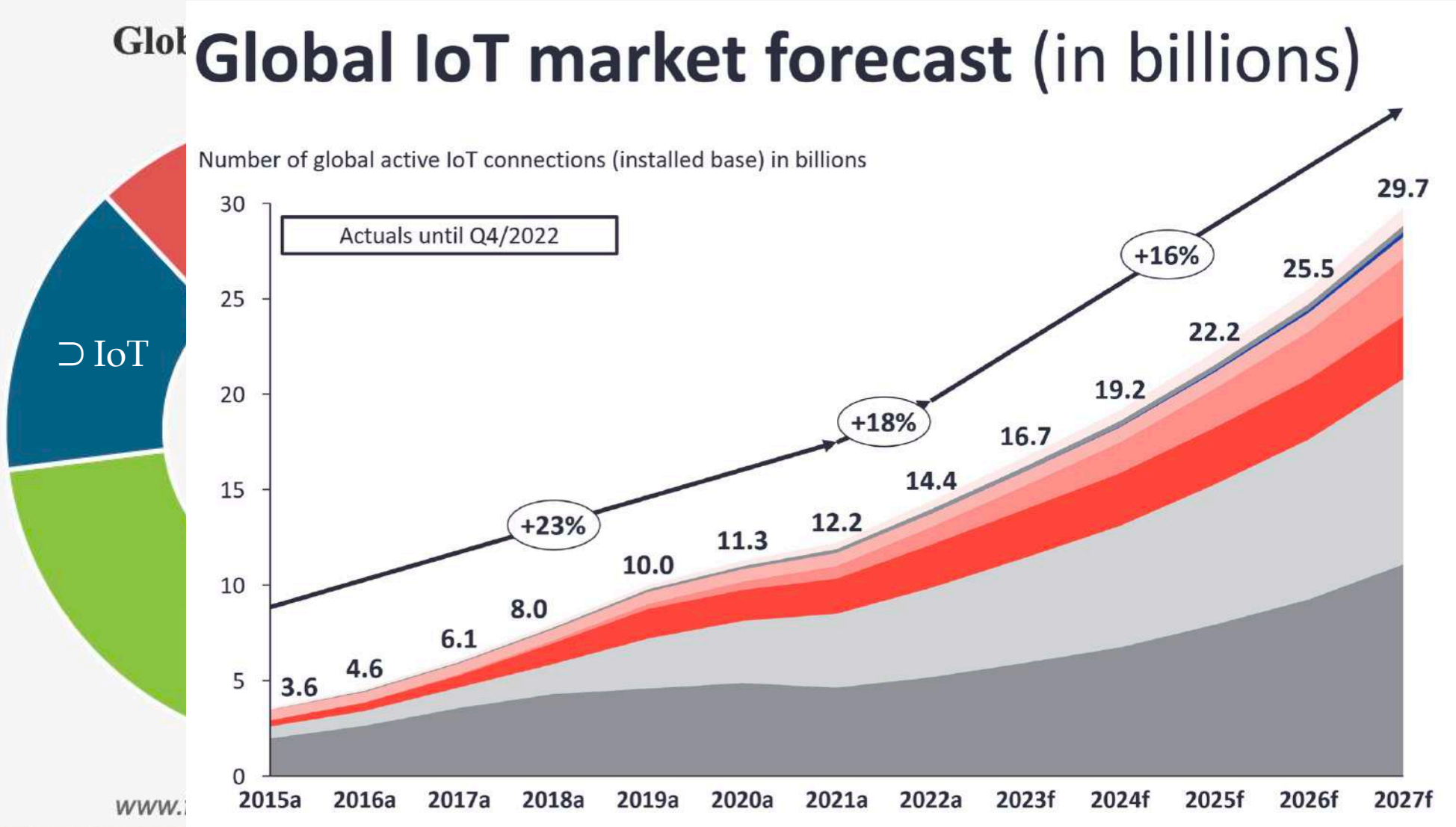
Tipicamente estão em dispositivos de pequeno porte, portáteis e com restrições severas de consumo energético;

Celular é um sistema embarcado ?!?

Classificação de Sistemas embarcados



Classificação de Sistemas embarcados



* IoT Analytics Research 2023

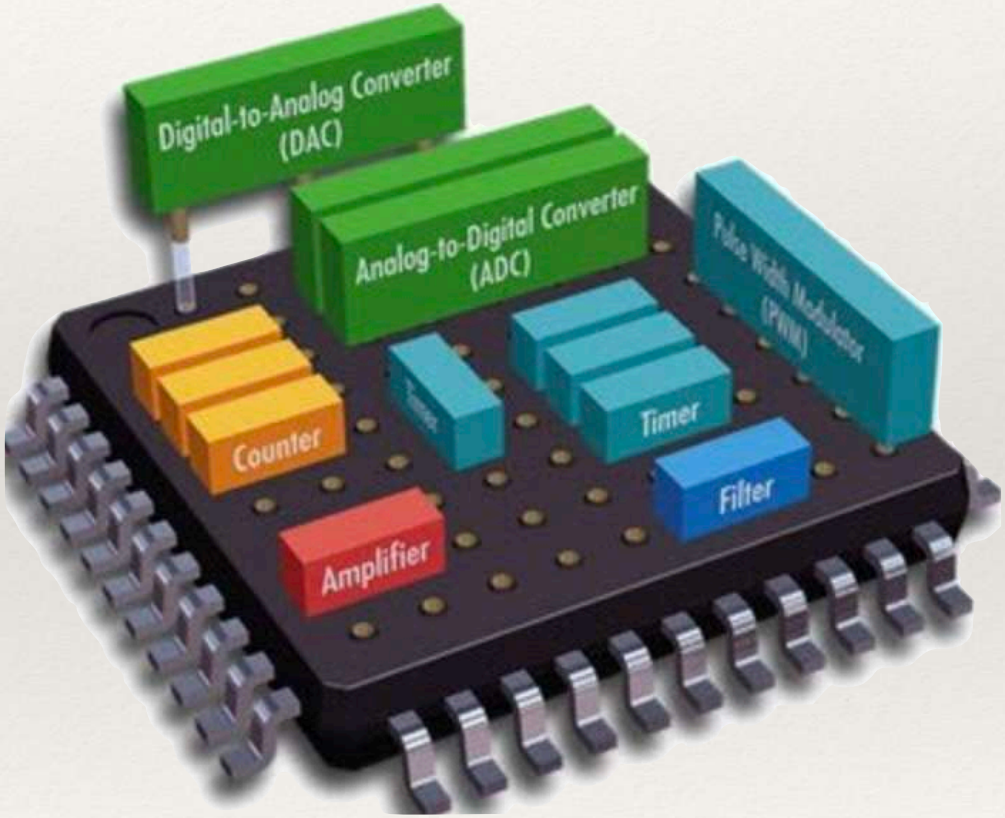
Aplicações de Sistemas embarcados

Automotivas

Industriais

Médicas

Outras aplicações



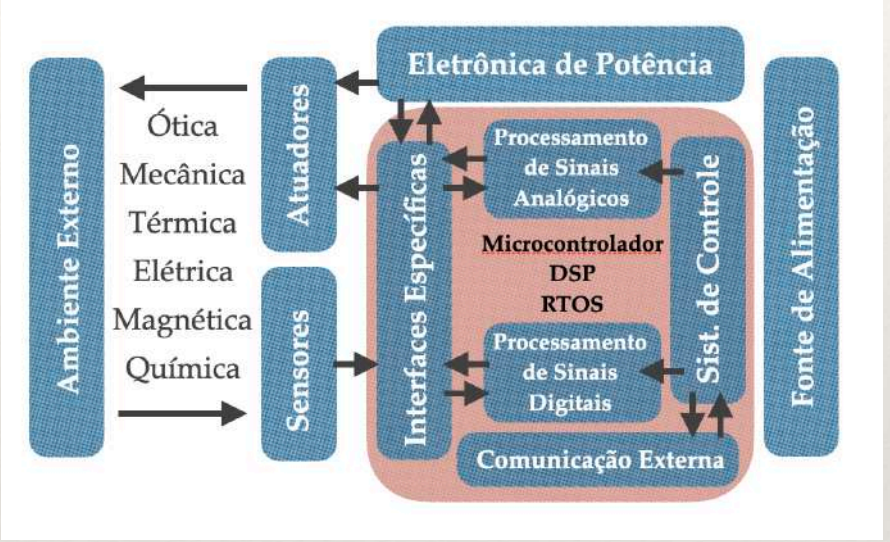
Aplicações de Sistemas embarcados

Automotivas

Aplicação comum dos Sistemas embarcados;

Envolve praticamente todos os componentes de um sistema completo;

Nos anos 70, o único equipamento eletrônico de um automóvel era o rádio !



Industriais

Médicas

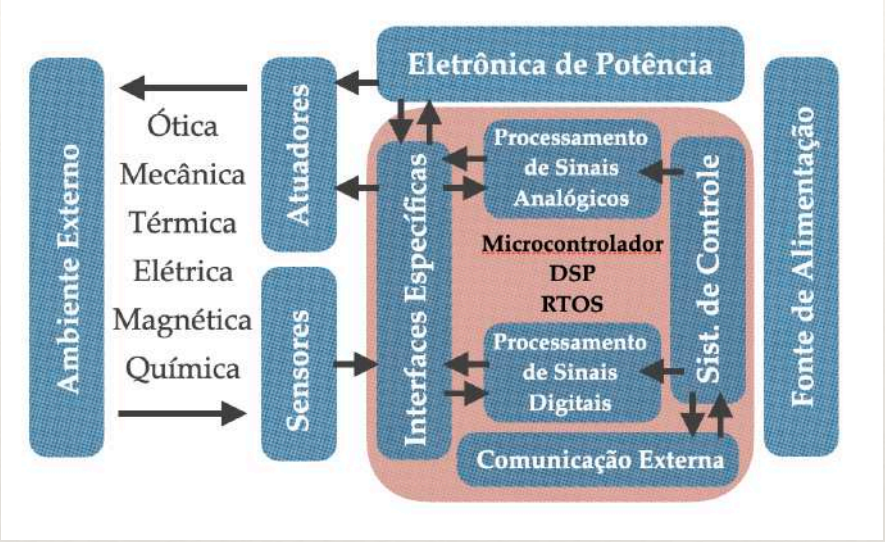
Aplicações de Sistemas embarcados

Automotivas

Hoje qualquer automóvel possui um Sistema embarcado (o Celta já tinha !);

Temos ignição e injeção eletrônica, computador de bordo, ABS, EBD, Airbag, suspensão ativa, travas elétricas, direção elétrica assistida, travas elétricas, ar-condicionado, *cruize control* ...

Tipicamente são múltiplos sistemas de diferentes fornecedores, interligados.



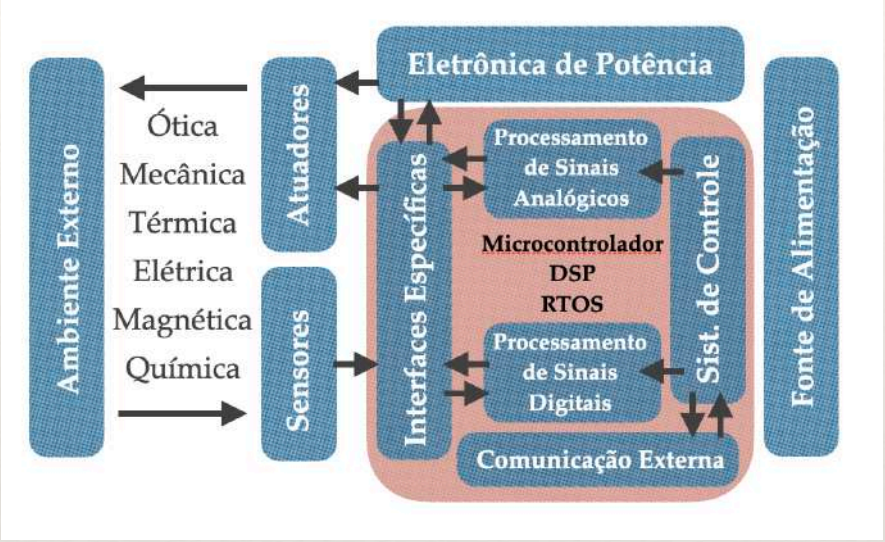
Industriais

Aplicações de Sistemas embarcados

Automotivas

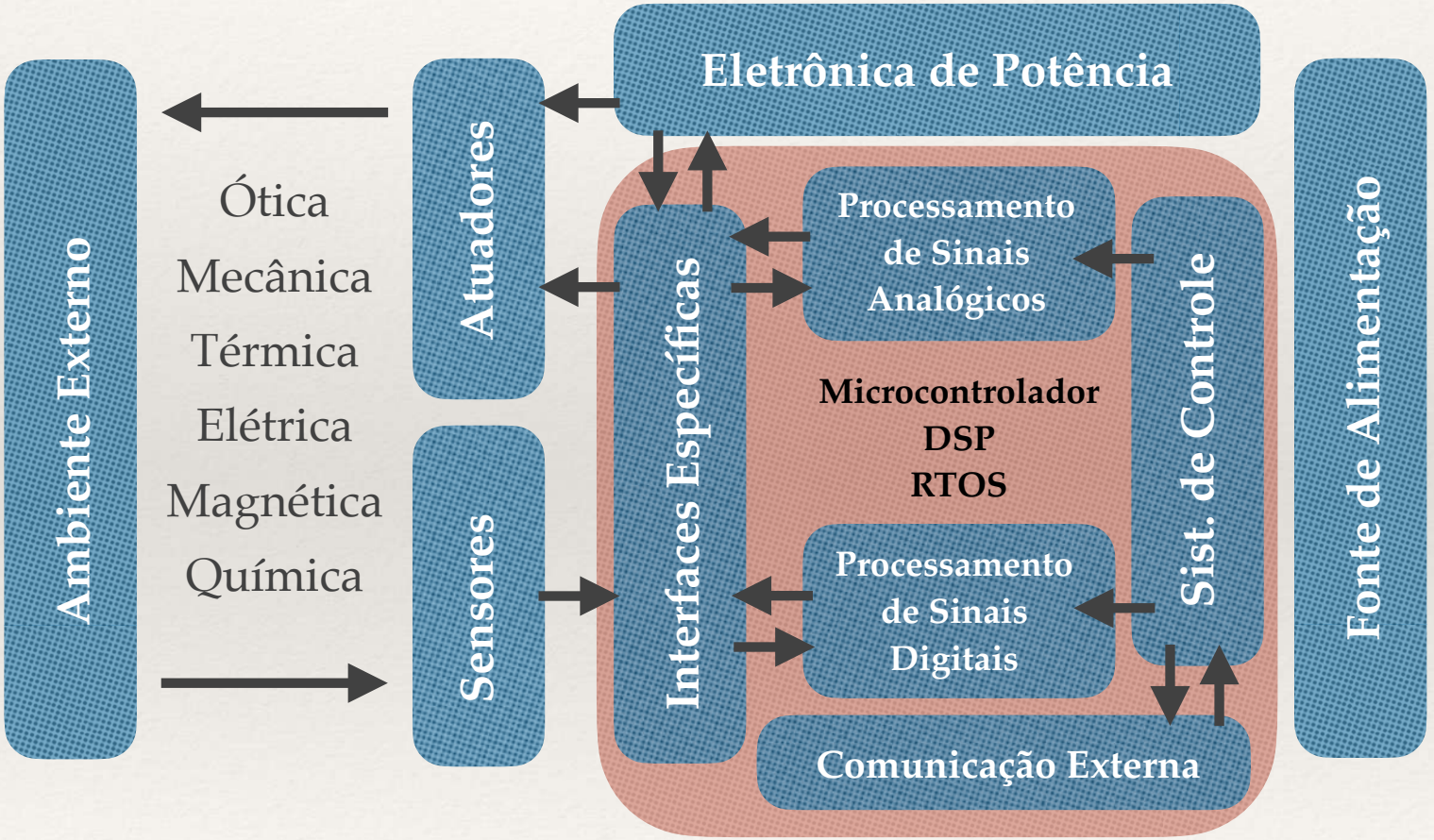
"Montadora" produz carroceria e motor. Os demais componentes são de fabricantes diversos (painel da Visteon, Freio da Bosch, Controle do Motor da Magnet Marelli, Ar-condicionado da Denso etc). Isto exige uma rede padronizada interligando os componentes do Sistema embarcado;

Protocolo CAN, porta OBD...



Industriais

Aplicações de Sistemas embarcados



Aplicações de Sistemas embarcados

Automotivas

Industriais

Especialistas identificam quatro Revoluções Industriais. A primeira ocorreu no final do século 18. A segunda, no final do século 19, está relacionada à aplicação da eletricidade e a criação da linha de montagem;

É na terceira, do final do século 20, que foram adotados os Sistemas embarcados, inclusive com os primeiros esforços de desenvolvimento da **Internet das Coisas** (IoT);

Especialistas dizem que estamos passando pela 4ª Revolução Industrial, a chamada **Indústria 4.0**. Os destaques agora são a CPS, a AI, o ML, o Big Data e a Robótica.

Médicas

Aplicações de Sistemas embarcados

Automotivas

Industriais

Especialistas identificam quatro Revoluções Industriais. A primeira ocorreu no final do [illegible] relacionada à aplicação da eletricidade

Cyber-Physical Systems: o conceito envolve a **integração** dos dispositivos e máquinas físicos com sistemas digitais "inteligentes". Esta permite o monitoramento em tempo real, análise e controle dos processos industriais.

[illegible]dotados os Sistemas embarcados, [illegible]vimento da **Internet das Coisas** (IoT);

[illegible]a 4ª Revolução Industrial, a chamada **Indústria 4.0**. Os destaques agora são a CPS, a AI, o ML, o Big Data e a Robótica.

Médicas

Aplicações de Sistemas embarcados

Automotivas

Industriais

A Indústria 4.0 obviamente considera a evolução do IoT, inclusive integrada à Computação em Nuvem;

Outra tecnologia considerada em alguns casos é a *Digital Twin*, para simulação, testes e manutenção preditiva;

Aplicações Industriais possuem exigências específicas quanto à operação em ambientes ruidosos, segurança e estabilidade.

Médicas

Aplicações de Sistemas embarcados

Automotivas

Industriais

Médicas

- Requisitos críticos de estabilidade e segurança;
- Utilizados em *smartwatches*, além de dispositivos para exames, monitoramento, geração de sinais vitais, próteses, robôs cirúrgicos etc.

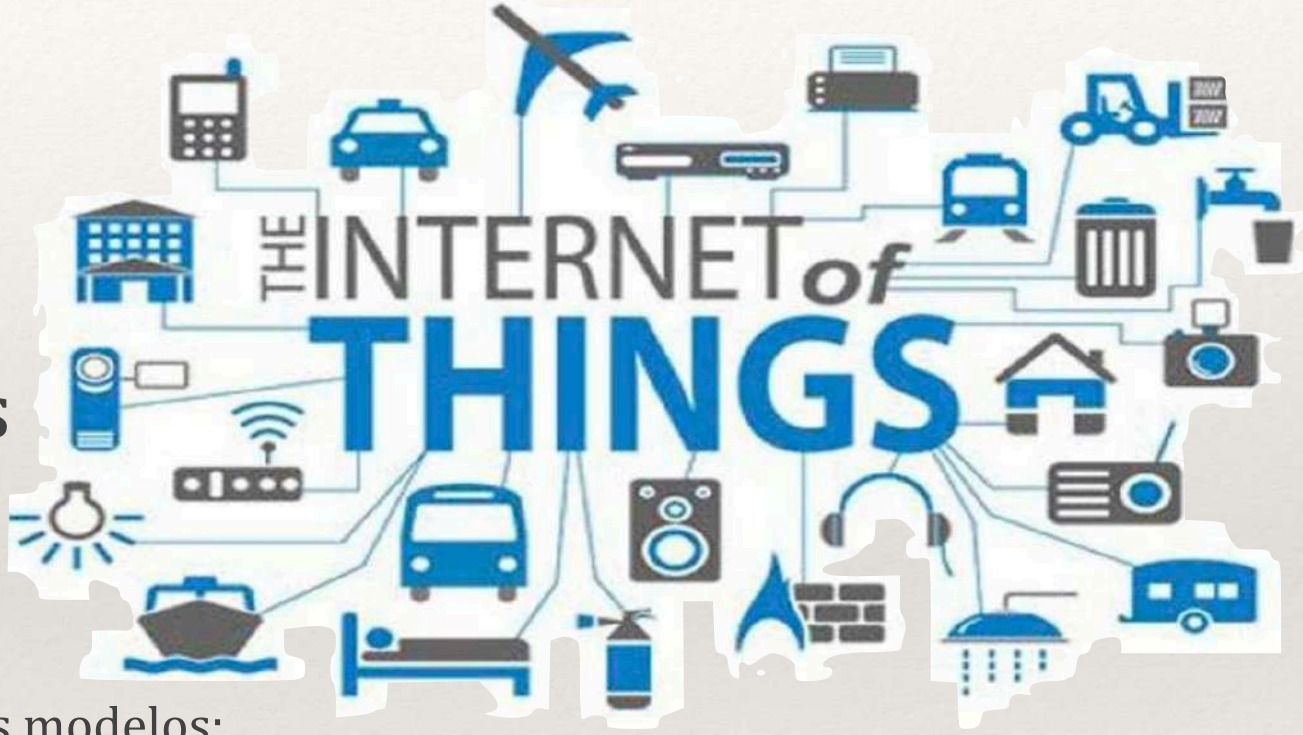
Outras aplicações

Aplicações de Sistemas embarcados

Industriais

Médicas

Outras aplicações



- Carros autônomos;
- Automação Residencial;
- *Drones* dos mais diversos modelos;
- Equipamentos e armamento militares;
- Equipamentos de Rede e Telecomunicações (*switch*, roteador, sistemas redundantes etc);
- Robôs diversos (do aspirador doméstico às maravilhas da Boston Dynamics);

Aula 02

Introdução ao IoT

Características

As principais características da Internet das Coisas são:

- A IoT pode ser caracterizada como uma rede mundial de coisas/objetos/dispositivos interconectados que se comportam como entidades ativas [Roman et al. 2011b];
- As coisas (dispositivos) na IoT, muitas vezes, possuem restrições de recursos como memória RAM ou ROM, poder de processamento e energia [Hummen et al. 2013];
- Mecanismos de comunicação de alguns dispositivos, na maioria das vezes sem fio, possuem baixa potência de transmissão e baixa taxa de dados [Mahalle et al. 2010];

Introdução ao IoT

Características

As principais características da Internet das Coisas são:

- Integra coisas (dispositivos) heterogêneos, o que demanda uma preocupação em relação a interoperabilidade entre estes [Atzori et al. 2010,Mahalle et al. 2012];
- Pode ser caracterizada como um ambiente contendo um grande número computadores ou dispositivos invisíveis que colaboram com o usuário, ou seja, um ambiente pervasivo e ubíquo [Hanumanthappa e Singh 2012];
- Na IoT, os usuários podem interagir com as coisas em seu ambiente físico e virtual de diversas maneiras [Mahalle et al. 2012].

Introdução ao IoT

Características

Informações retiradas do artigo:
https://wiki.inf.ufpr.br/maziero/lib/exe/fetch.php?media=ceseg:2013-sbseg-mc4.pdf

- A rede possui uma topologia dinâmica, pois muitos nós entram e saem da rede com frequência [Mahalle et al. 2012,Hanumanthappa e Singh 2012];
- Há uma grande quantidade de coisas (dispositivos) com ciclo curto de vida, o que exige uma alta capacidade de gerenciamento [Fongen 2012];

IoT e as Redes

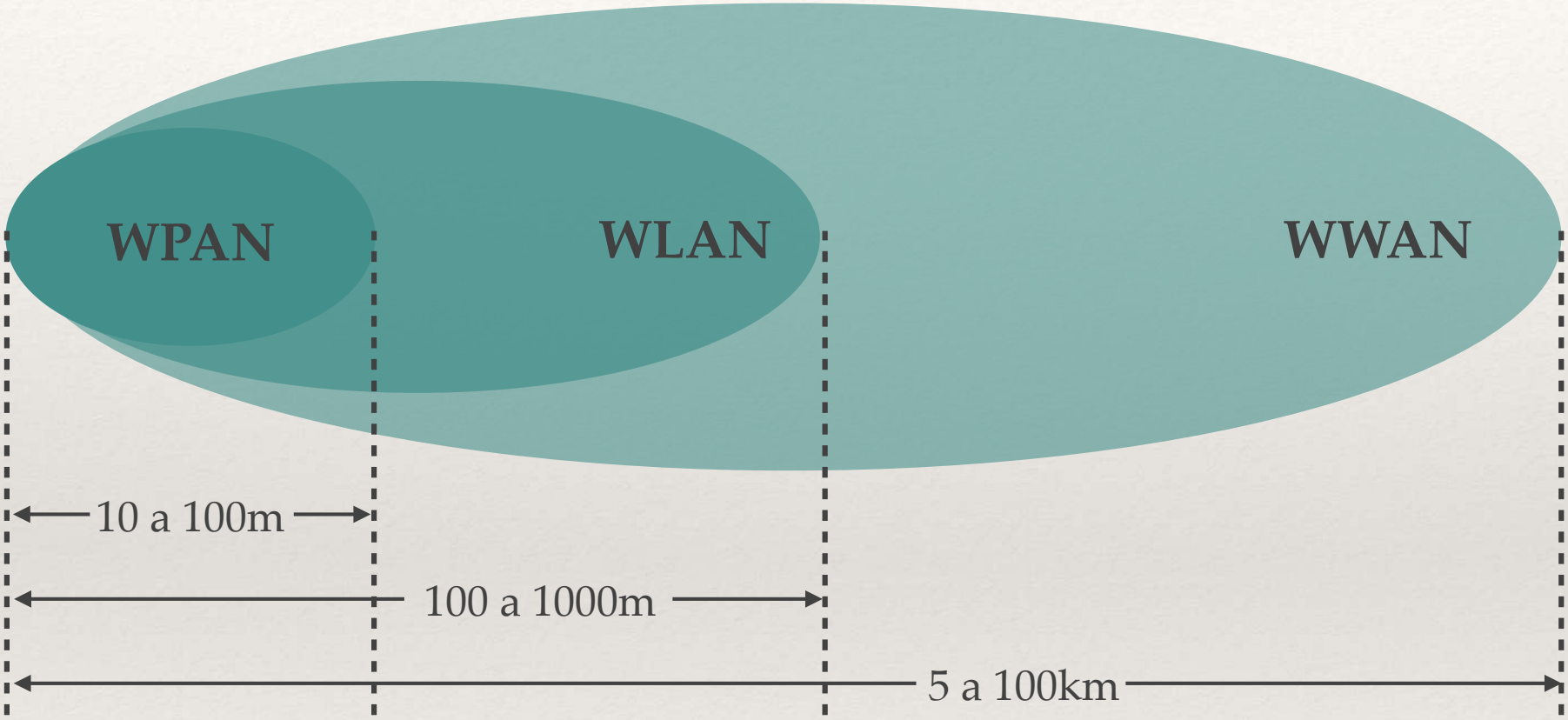
É inegável que a tecnologia IoT depende da qualidade das redes de comunicação, embora existam outros aspectos também importantes, como os Sistemas Operacionais e aplicações suportadas;

A tecnologia IoT adota múltiplos protocolos, alguns já existentes antes do seu surgimento, como o Wi-Fi, e outros bem ajustados às suas características:

- Baixa demanda de largura de banda;
- Baixo consumo energético;
- Acesso típico sem fio.

Tecnologias como BlueTooth, RFID, ZigBee, NFC e LoWPAN estão entre elas.

IoT e as Redes (alcance)



A IoT normalmente está associada a redes com baixo alcance. No entanto, alguns protocolos mais recentes têm expandido seu alcance, como a LoRaWAN, por exemplo;

Novas aplicações como telemetria, medição de consumo residencial de água, luz e gás, gestão de trânsito e agropecuária podem ser possíveis.

IoT X segurança

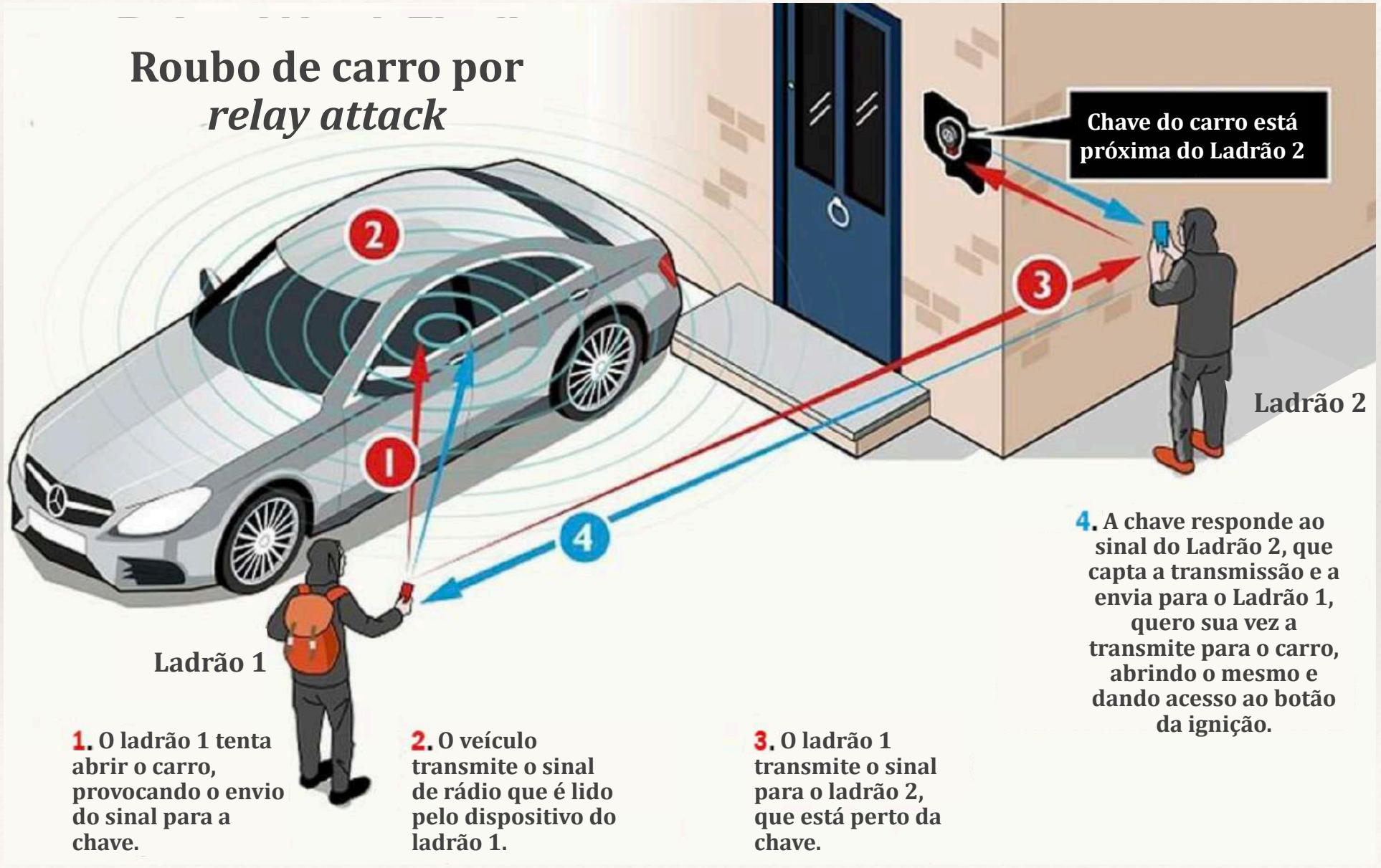
O conceito da IoT é conectar “tudo”, e quanto mais acessos à rede existirem, maiores são os riscos de segurança;

Métodos criptográficos seguros exigem processamento rápido e vigilância contínua, o que atenta contra as características dos dispositivos IoT !

- Capacidade limitada de processamento;
- Baixo consumo energético.

São inúmeros os casos de crimes que exploram as vulnerabilidades dos sistemas IoT.

IoT X segurança



Home › Notícias › Segurança

Furto de carros “sem chave” está se tornando comum no Brasil

Por Felipe Demartini | Editado por Claudio Yuge | 08 de Abril de 2022 às 19h49

Imagem: Felipe Ribeiro/Canaltech

Uma brecha de segurança em modelos de veículos que utilizam a tecnologia keyless para permitir a abertura de portas e partida do motor sem a necessidade de uso da chave,

1. O ladrão 1 tenta abrir o carro, provocando o envio do sinal para a chave.

2. O veículo transmite o sinal de rádio que é lido pelo dispositivo do ladrão 1.

3. O ladrão 1 transmite o sinal para o ladrão 2, que está perto da chave.

onde ao
ão 2, que
nissão e a
Ladrão 1,
vez a
a o carro,
esmo e
dando acesso ao botão da ignição.

Como ladrões de carros estão virando 'hackers' para roubar modelos mais modernos

É importante manter o software do veículo atualizado, assim como já fazemos com telefones e computadores, para evitar invasões.

Por Doug Jacobson*, BBC

18/08/2023 10h07 · Atualizado há 6 meses

abrir o carro, provocando o envio do sinal para a chave.

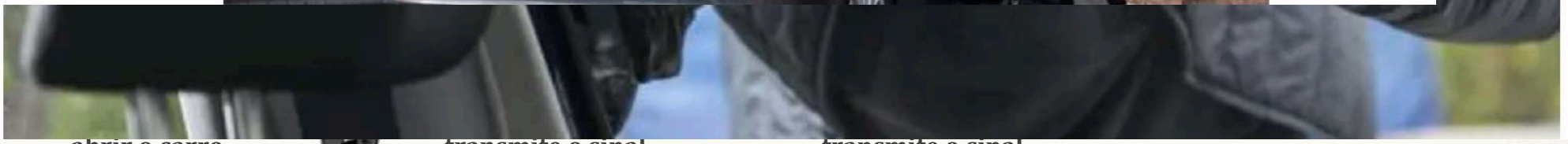
transmite o sinal de rádio que é lido pelo dispositivo do ladrão 1.

transmite o sinal para o ladrão 2, que está perto da chave.

Carros › Manutenção e Segurança

'Relay attack': em novo golpe, bandidos clonam a chave do carro a distância

Alessandro Reis • Do UOL, em São Paulo (SP)
05/10/2023 04h00



abrir o carro, provocando o envio do sinal para a chave.

transmite o sinal de rádio que é lido pelo dispositivo do ladrão 1.

transmite o sinal para o ladrão 2, que está perto da chave.

IoT X segurança

AirTags são 'a arma favorita de perseguidores', afirma processo contra a Apple

16/10/2023 às 10:57 • 1 min de leitura

COMPARTILHE

Atualização do Apple AirTag traz recurso 'anti-stalking' ao dispositivo de rastreio

22 de dezembro de 2022 4

O **Apple Airtag** recebeu uma nova **atualização de firmware** com a chegada de um recurso anti-stalking. Ou seja, uma característica que pode ajudar as pessoas a não serem perseguidas ou vigiadas. Esse update é o **2.0.24 que chegou em novembro**, mas só em dezembro é que finalmente atingiu 100% dos aparelhos da maçã.

Notícias > Gadgets

Android ganha proteção contra stalkers que usam o Apple AirTag

Google libera ferramenta nativa que avisa sobre rastreadores por perto. Usuário também encontra dicas de como desativar o equipamento oculto.

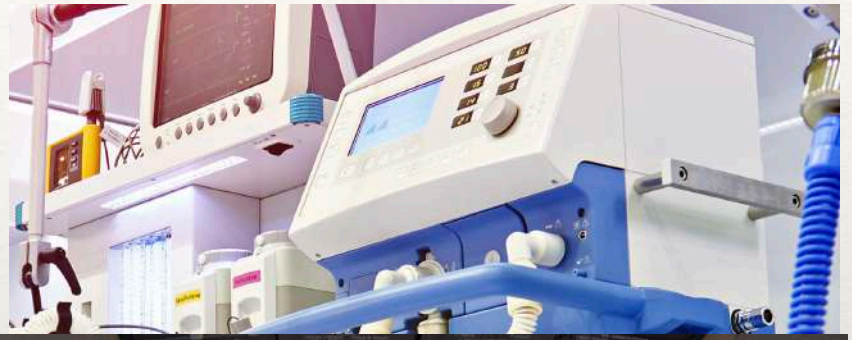
Por Thássius Veloso
10/08/2023 às 12:53

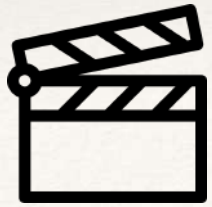
IoT X segurança

Outras implementações também foram alvos de ataque, embora não sejam tão bem documentadas na imprensa. Quais seriam os próximos alvos?

- Sistemas Médicos (assassinato)?
- Sistemas Aviônicos (terrorismo)?

É claro que é necessária precaução, mas também não devemos exagerar ...





HW de Sistemas Embarcados

No sistema embarcado, o HW é um conjunto de componentes

- Dispositivo "processador";
- Atuadores;
- Fontes de alimentação;
- Sensores diversos;
- Interfaces e protocolos de comunicação;
- Componentes eletrônicos diversos.

Escolher corretamente cada um dos componentes é essencial

- A portabilidade pode ser inviável;
- O *hardware* determina os parâmetros de projeto, inclusive de *software*.

Outro fator importante são as condições ambientais

- Acesso, proteção, temperatura, vibração, EMI etc

Além de tudo isso, tem o SW !

- Ele é a verdadeira “inteligência" da automação.

Processador

Um sistema embarcado de automação moderno tipicamente exige um processador. Temos algumas opções:

CISC Intel i9-13900K
14 núcleos
20 threads
24MB cache
Clock 2.6GHz até 5.4GHz
Consumo de 45W
64 GB DDR5
SSD 1TB PCIe NVM M.2

Processador

Um sistema embarcado de automação moderno tipicamente exige um processador. Temos algumas opções:

RISC Appl- M3 Max
14 ...úcleos
40 núcleos GPUs
16 núcleos neurais
Clock 2.7GHz até 4.1GHz
C...sumo de 78W
36 GB Memória U...ificada
SSD 1TB

Processador

Um sistema embarcado de automação moderno tipicamente exige um processador;

São máquinas maravilhosas, rápidas e modernas, MAS ...

Desempenho e capacidade extraordinária de armazenamento não são tão importantes neste tipo de aplicação !

Quais são os requisitos típicos de um processador para um sistema de automação ?

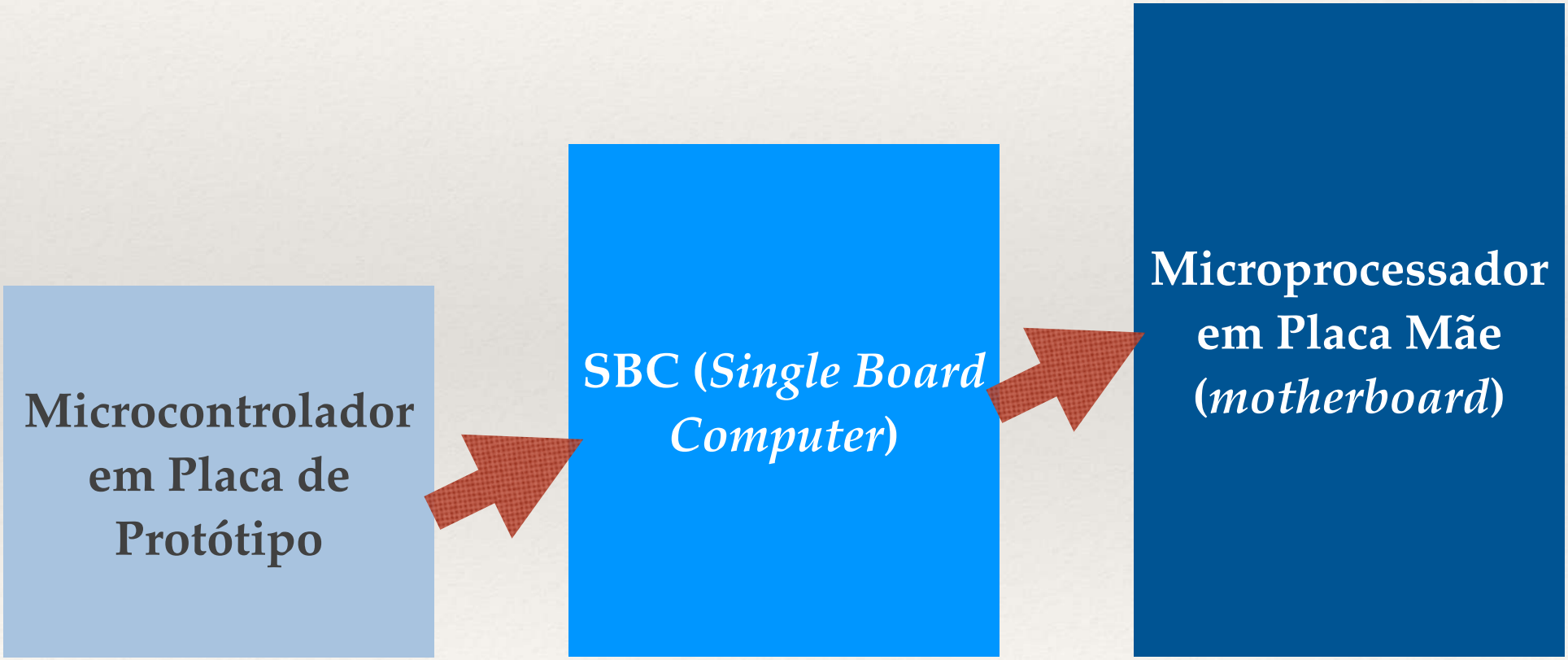
Consumo baixo de energia;

Dimensões reduzidas;

Custo acessível.

Processador

Não faz sentido aumentar complexidade, e com isto consumo, tamanho e custo ...



Processador

Microprocessadores

- Tipicamente baseados na arquitetura Von Neumann;
- Tipicamente divide-se em ULA, Unidade de Controle e registradores;
- Exige componentes externos essenciais, como memórias, barramentos, *clock*, interfaces de E/S etc;
- Projetados para múltiplos usos;
- Devido a esta dependências, sistemas microprocessados ocupam mais espaço.

Microcontroladores

- Tipicamente baseados na arquitetura de Harvard;
- Além dos componentes presentes em um microprocessador, condensa periféricos como memória, *clock*, interfaces de E/S diversas, conversores A/D e D/A, e portas de acesso GPIO;
- Projetados para uso específico;
- Tipicamente são limitados em performance e capacidade para redução do consumo, tamanho e custo.

Arquitetura Básica

Von Neumann

Conceito desenvolvido pela equipe de John Von Neumann no IAS (*Institute for Advanced Studies*) da Universidade de Princeton em 1946;

Tornou-se um padrão de fato, inspirando projetos até os dias atuais (**"Esta máquina é Von Neumann ?"**)

Estabeleceu o padrão de armazenamento do código executável na memória do dispositivo;

Executa uma instrução em dois ciclos;

Possui um barramento de cada tipo (dados, endereços e controle).

Harvard

Conceito desenvolvido para o computador Mark I, desenvolvido por Howard Aiken em 1937.

Separa memórias de uso específico (dados e instruções), que são acessíveis por barramentos diferentes;

Comum em microcontroladores, e em processadores DSP (*Digital Signal Processing*);

Executa uma instrução em um ciclo;

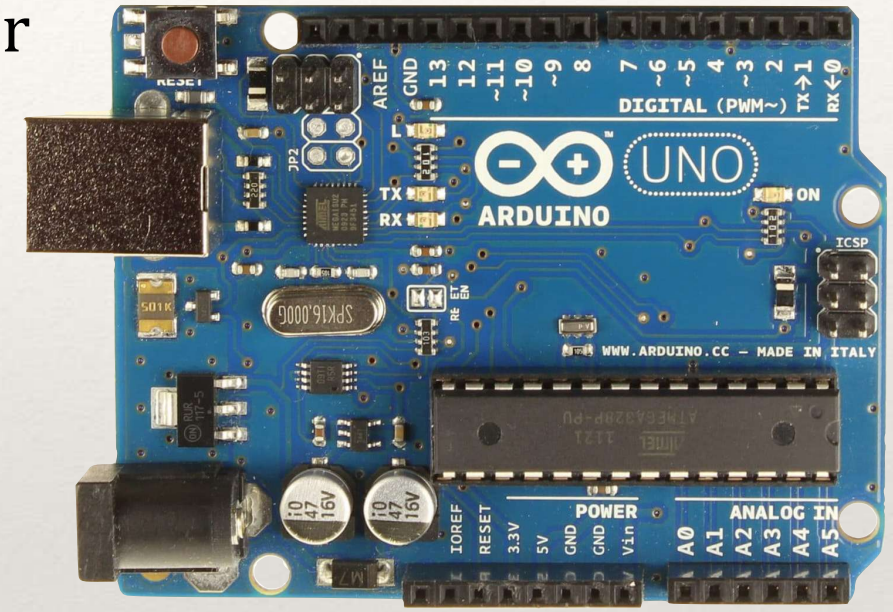
Permite a implementação do *pipelining*, técnica de processamento paralelo facilitada pelos barramentos independentes.

Microcontroladores

Microcontrolador x Microprocessador

Microprocessadores são baseados na arquitetura von Neumann, enquanto que os **Microcontroladores** são baseados na arquitetura Harvard. Nesta última, a memória de programa está separada da memória de dados;

Microprocessadores tipicamente exigem diversos componentes auxiliares externos, inclusive a memória;



Placa de Protótipo Arduíno Uno

Microcontroladores tipicamente possuem características mais simples, são menores e e consomem menos energia;

Microprocessadores possuem relógios (*clock*) muito mais rápidos e estáveis, oferecendo poder computacional mais elevado.

Microcontroladores

Tamanho e consumo

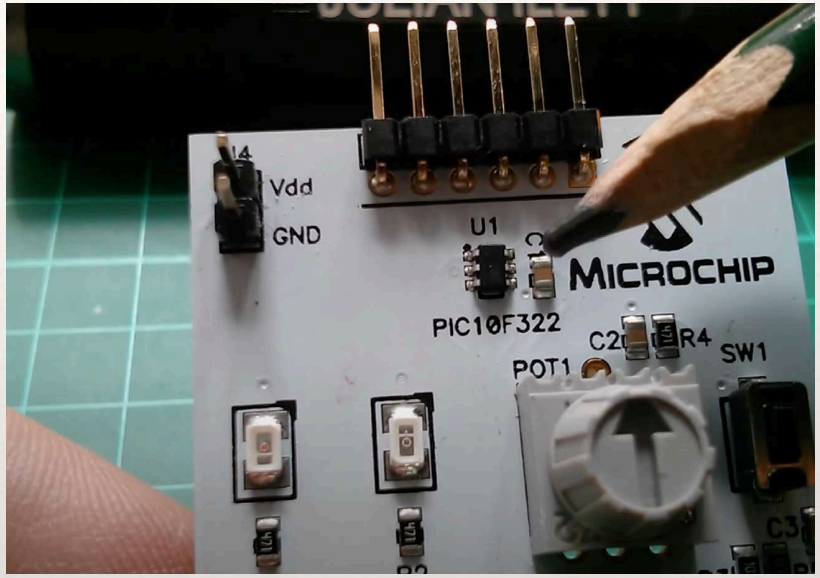
Estes são aspectos fundamentais em projetos embarcados, determinando inclusive a viabilidade, ou não, de algumas aplicações;

Tamanho

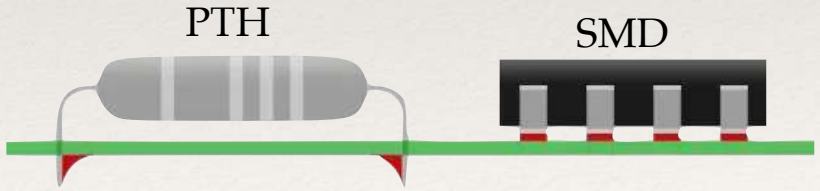
Atualmente, com a miniaturização em larga escala dos componentes, o número de componentes internos não impõe limites mínimos às dimensões do CI;

No entanto, reduzir o número de pinos afeta o tamanho significativamente, mas dificulta as interfaces, limitando as aplicações;

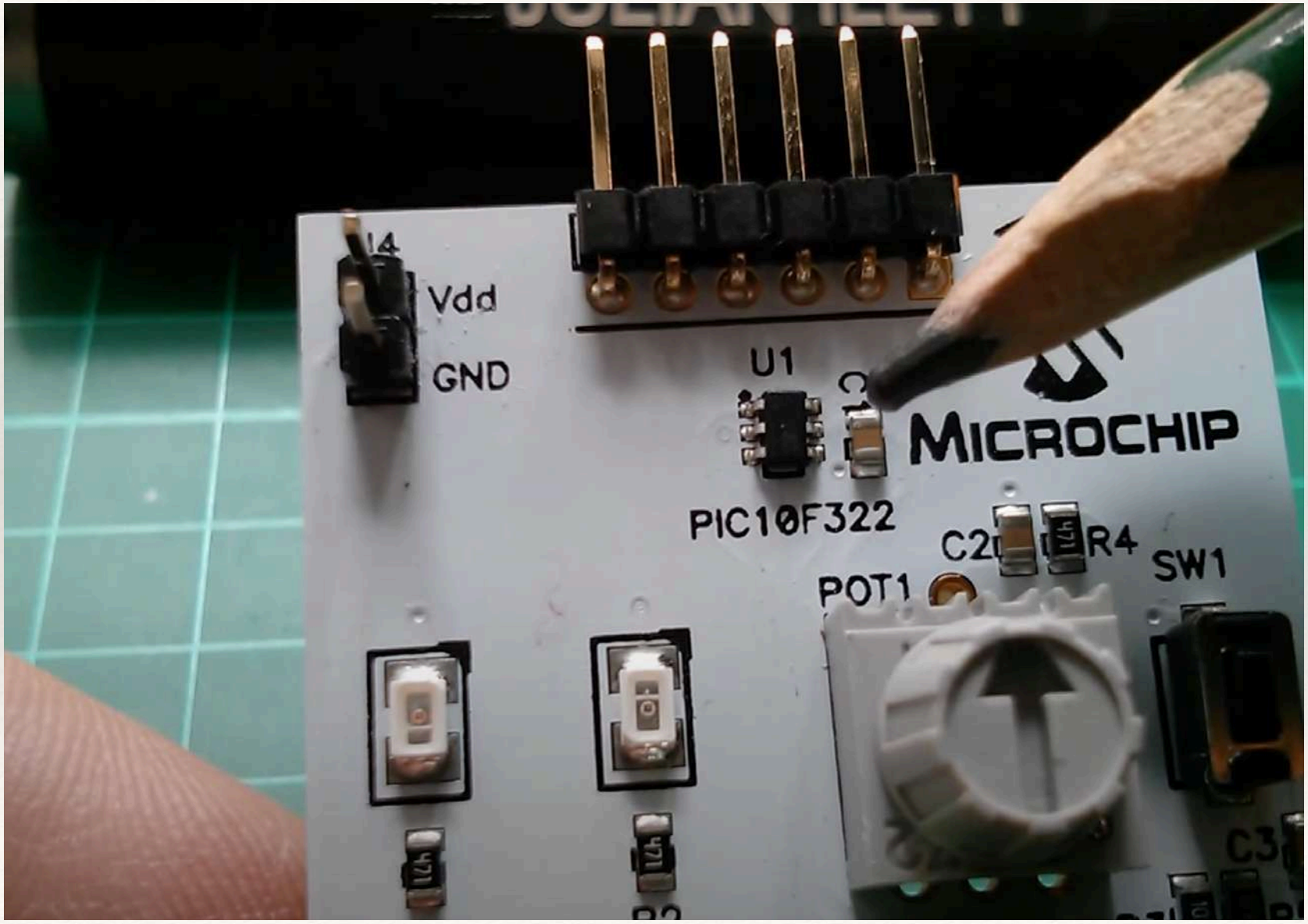
Outro fator é o tipo de pino (PTH ou SMD)
PTH - *Plated-Through Hole*
SMD - *Surface Mount Device*



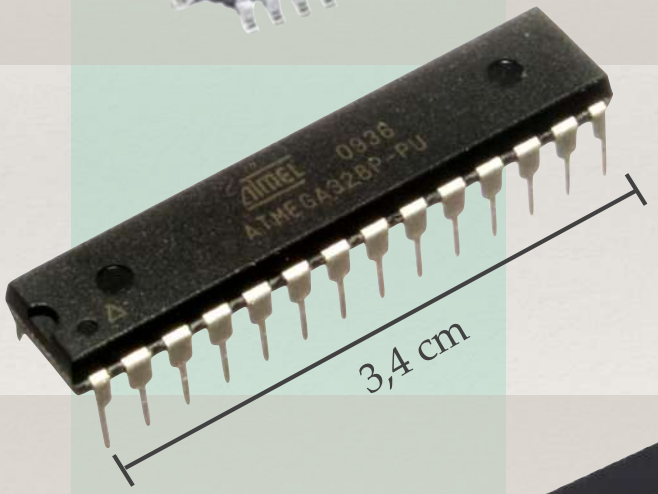
Placa de Protótipo PIC10F322



Microcontroladores



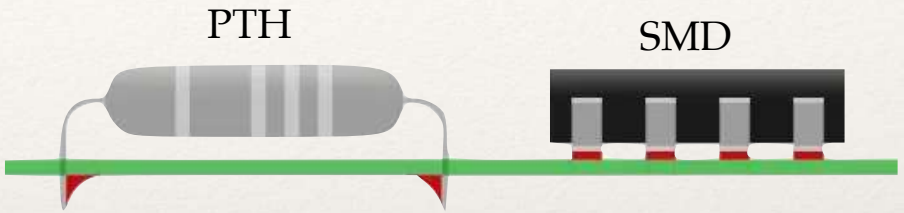
Comparativo Dimensional

	PIC10F322	ATTiny85	ATMega328P	ESP32 SoC
	6 pinos	8 pinos	28 pinos	48 pinos
SMD				
PTH				
QFN				

Conexões

Outro fator importante é a forma de conexão ao circuito:

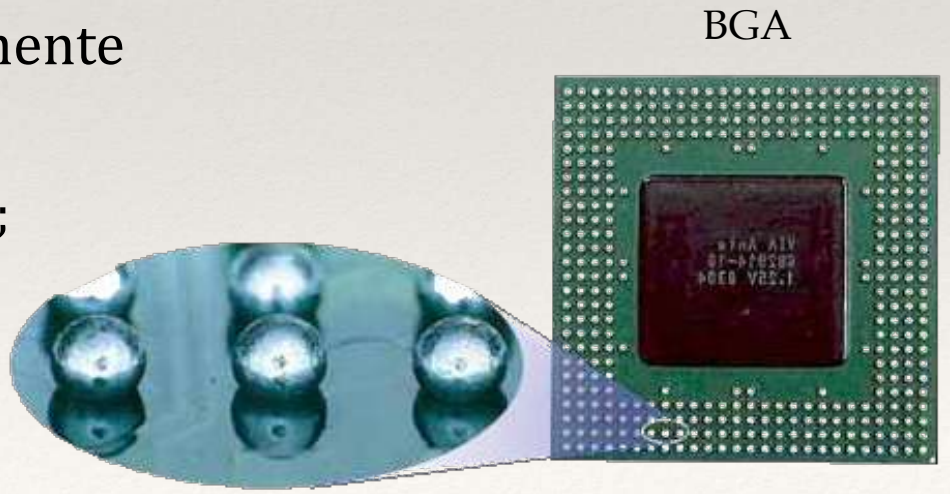
PTH - *Plated-Through Hole*
SMD - *Surface Mount Device*



Tamanho dos componentes SMD é muito menor, especialmente com encapsulamento BGA (*Ball Grid Array*) pela redução do espaço ocupado pelos pinos.

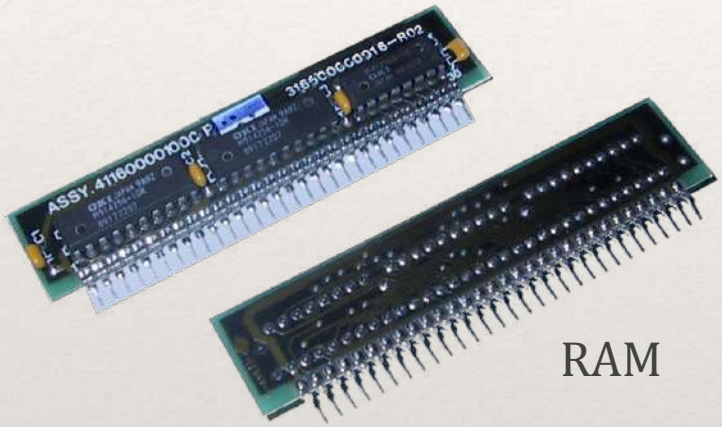
Tamanho afeta positiva ou negativamente diversos aspectos:

Dissipação de calor;
Interferência;
Manutenção.
Custo de Produção;
Confiabilidade;



Encapsulamentos (PTH)

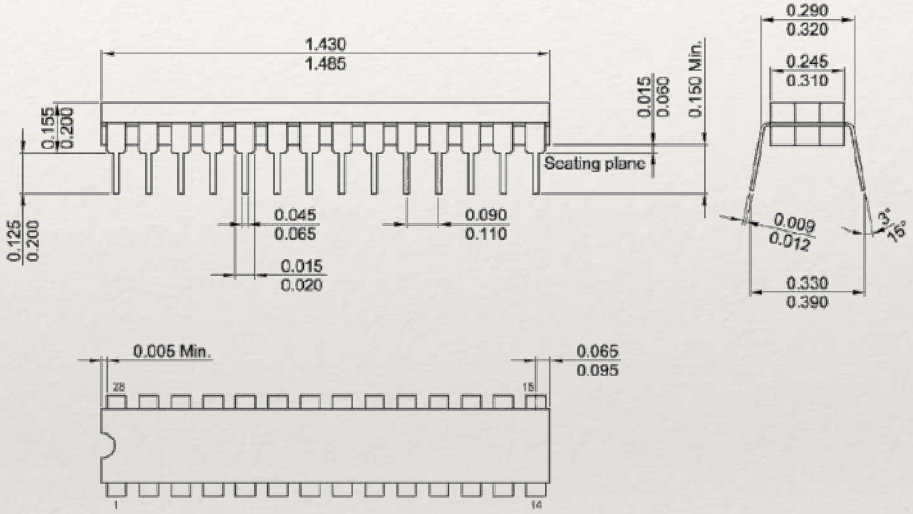
SIP - *Single In-Line Package*

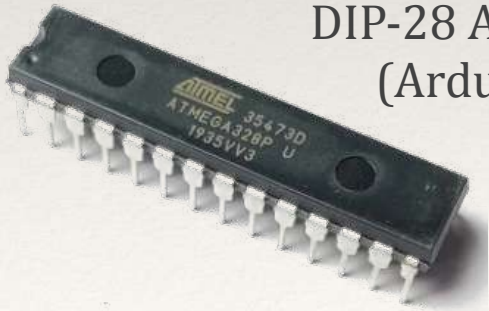


RAM

Resistor Network

DIP - *Dual In-Line Package*

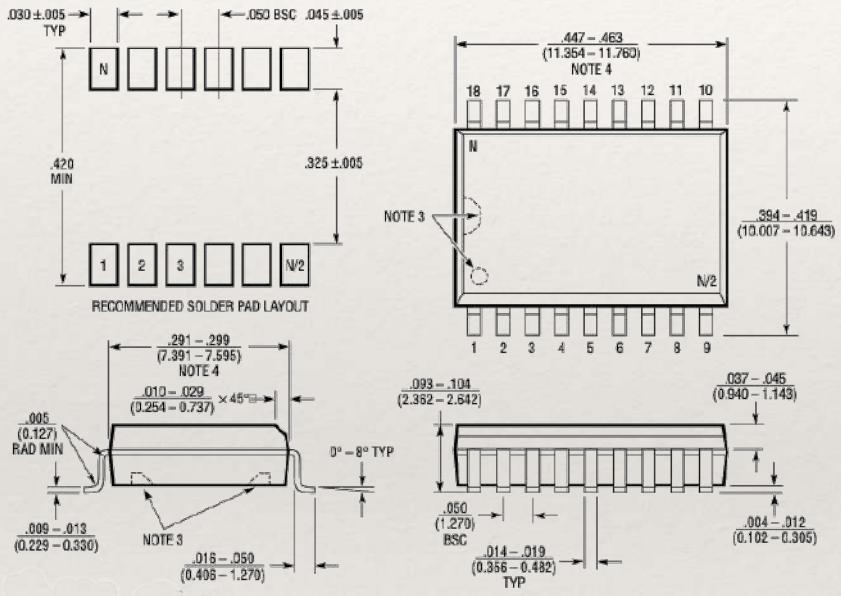


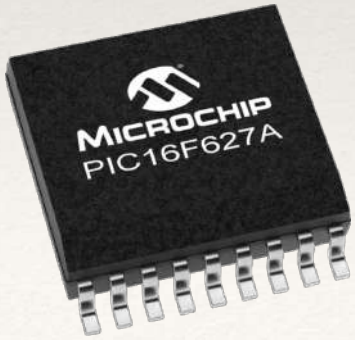


DIP-28 ATMega328P
(Arduíno Uno)

Encapsulamentos (SMD)

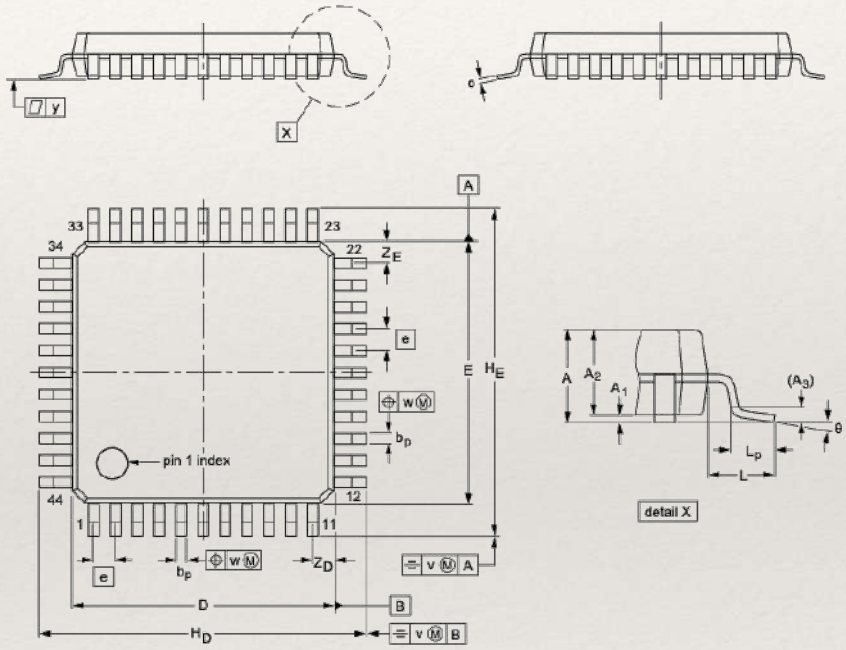
SOIC - *Small Outline Integrated Circuit*

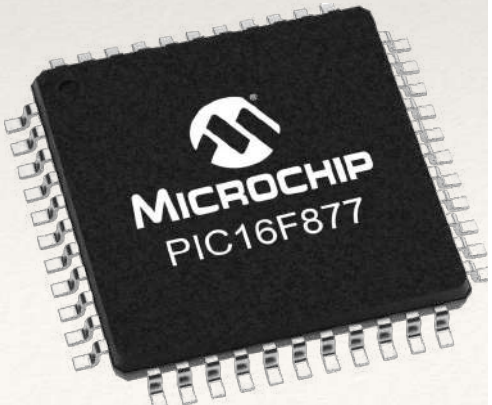




SOIC-18
PIC16F627A

QFP - *Quad Flat Package*

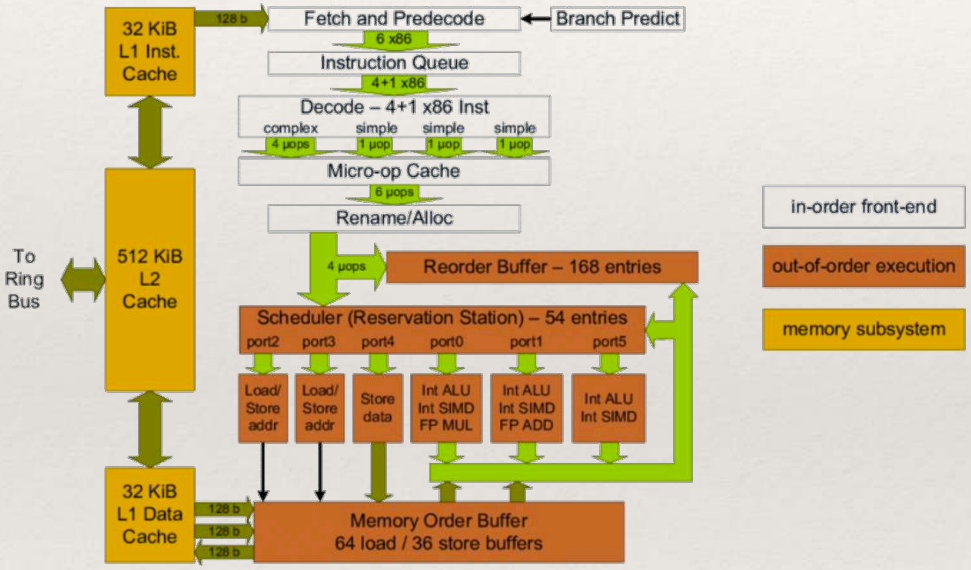




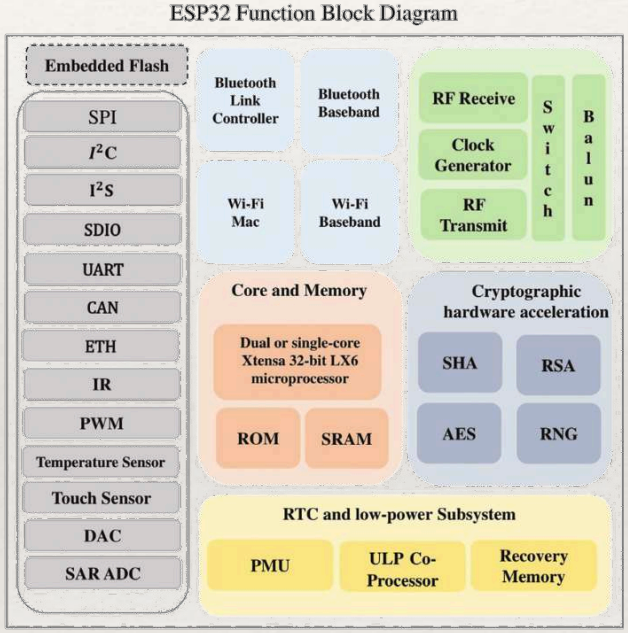
QFP-44
PIC16F877

Opções de Processador

Microprocessadores



Microcontroladores



Microprocessadores

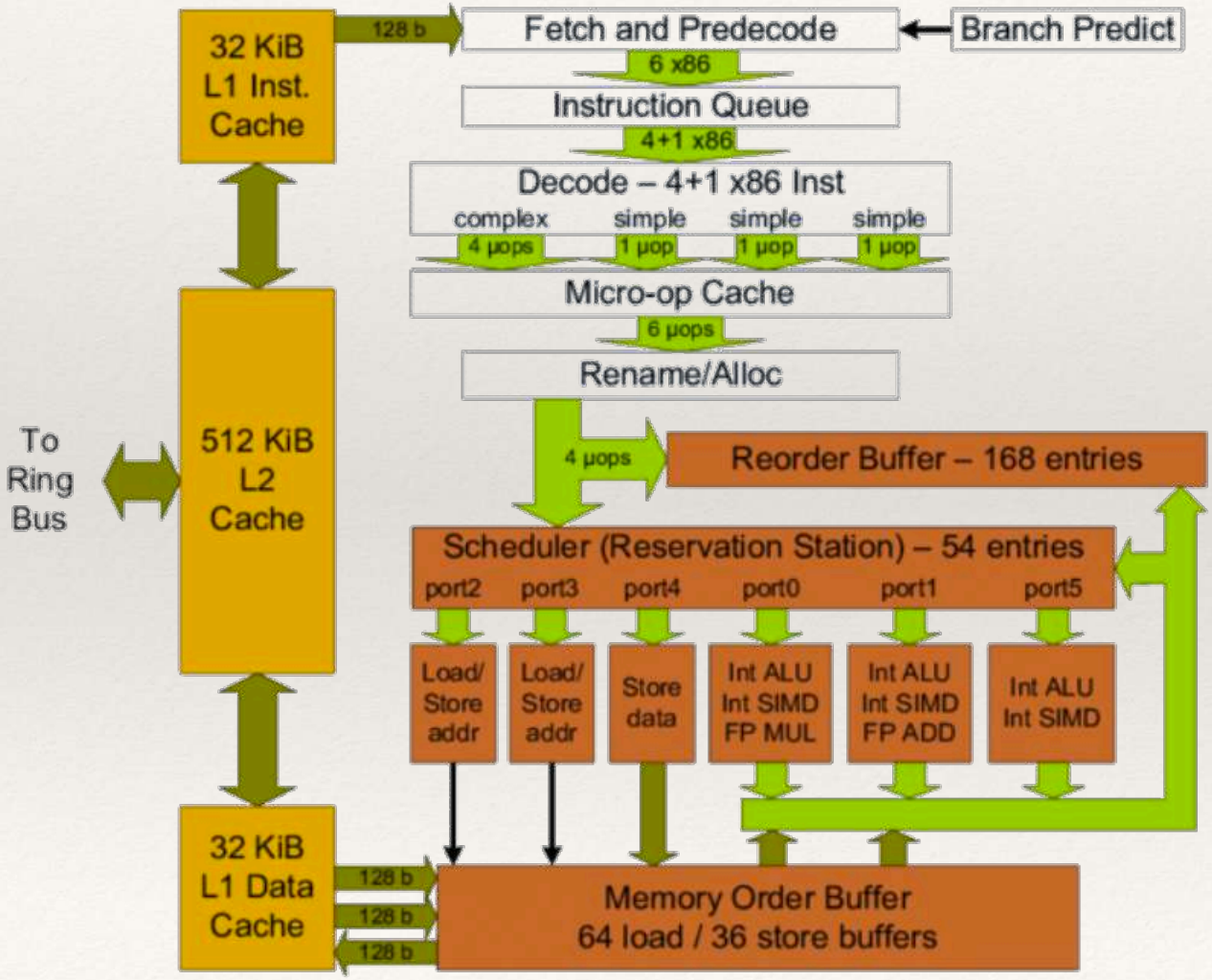


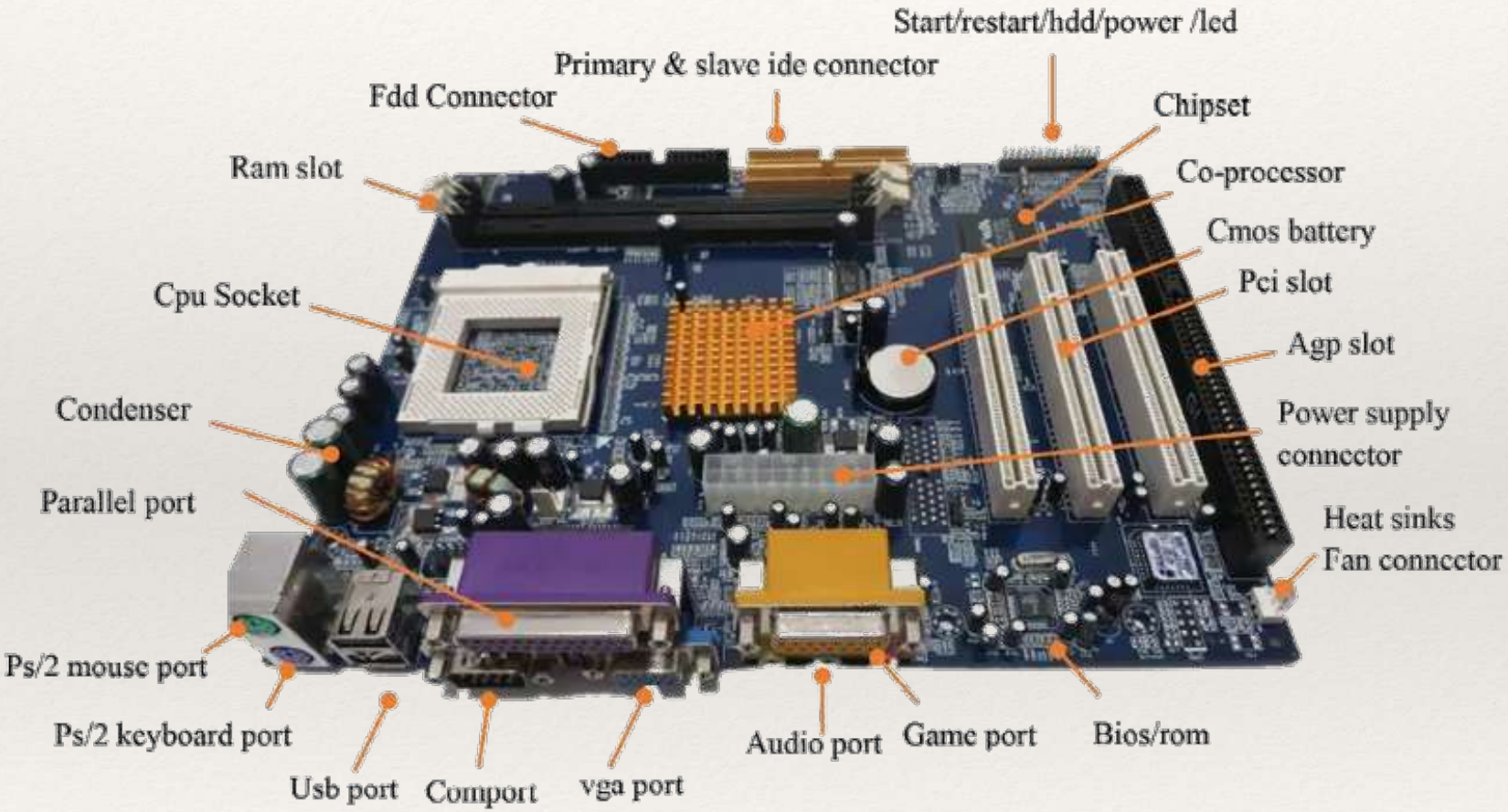
Diagrama funcional em blocos de um processador Intel (*Sandy Bridge*)

in-order front-end

out-of-order execution

memory subsystem

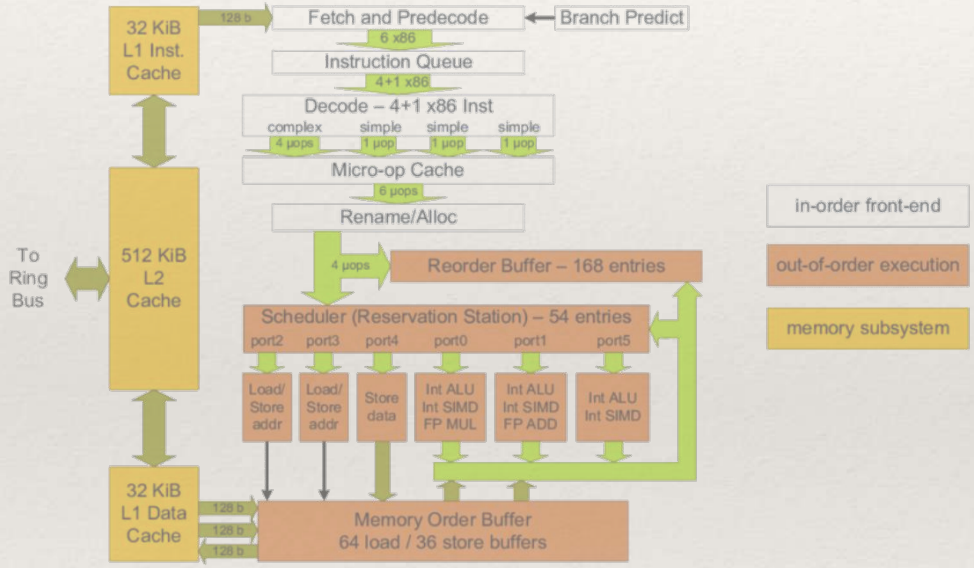
Microprocessadores



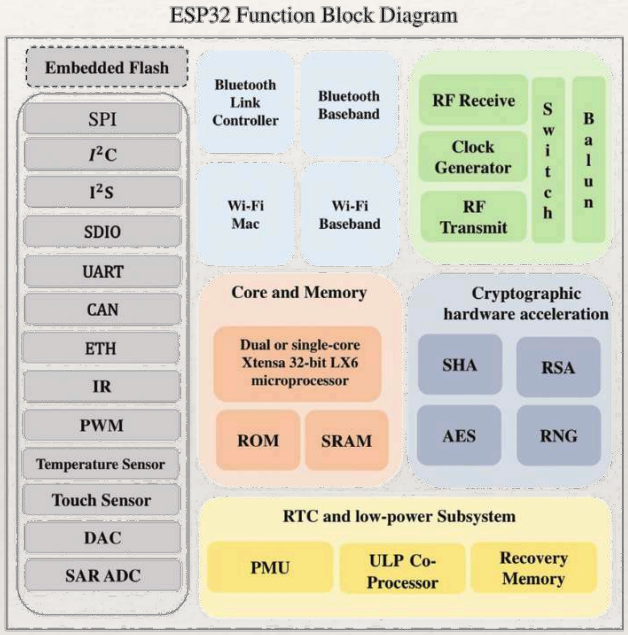
https://bit.ly/3xf8Vr3

Opções de Processador

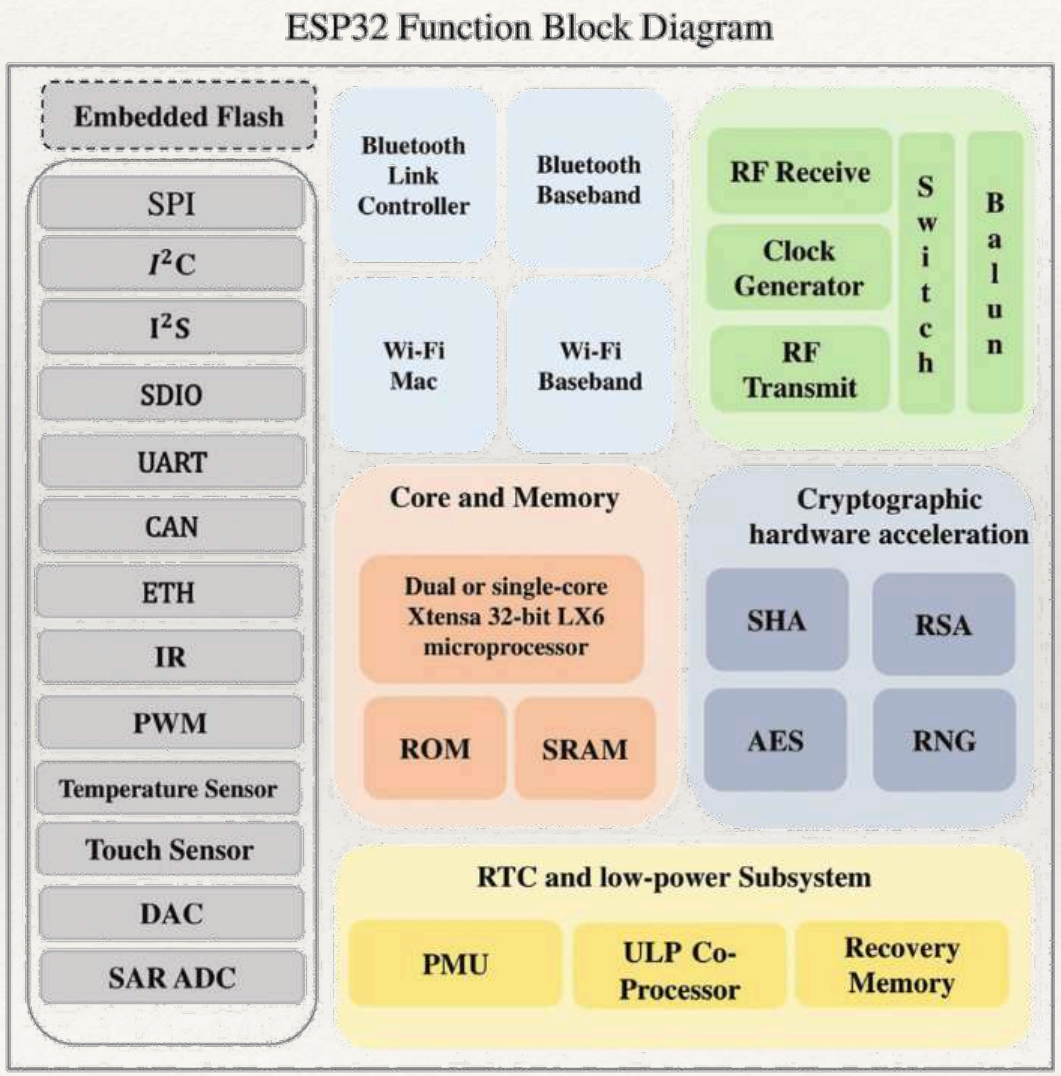
Microprocessadores



Microcontroladores



Microcontroladores



Conceito de SoC (*System on a Chip*)

*** Na foto, uma placa de um kit de desenvolvimento baseado no ESP32.**

Placas de Protótipo

Um processador exige companhia !

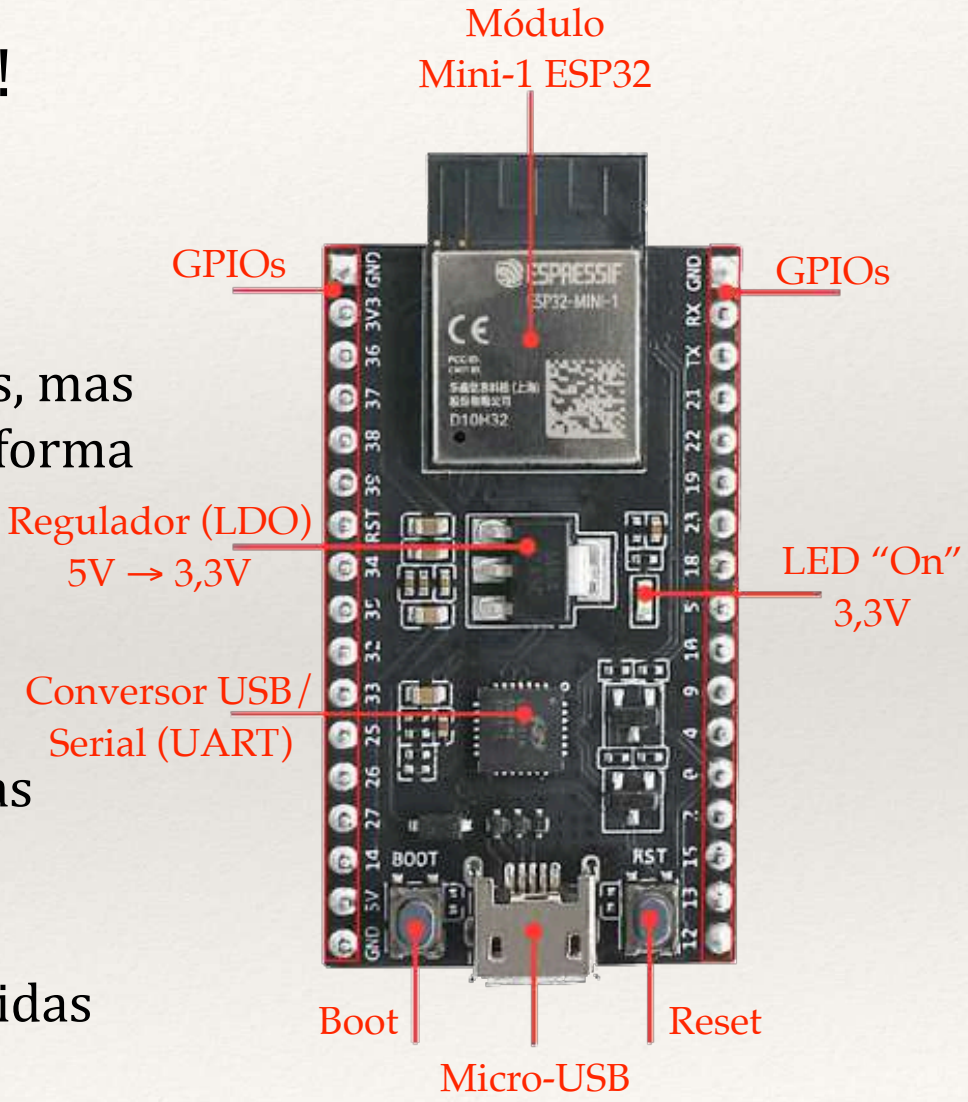
Microprocessadores precisam de muitos componentes externos essenciais à sua operação;

Microcontroladores são menos exigentes, mas mesmo assim não podem ser usados de forma isolada.

Placas de Protótipo

Também chamadas de “kits de desenvolvimento” (*development kits*), elas permitem testar protótipos com menor esforço;

Na foto temos uma das placas desenvolvidas com o EXP32 da Espressif.

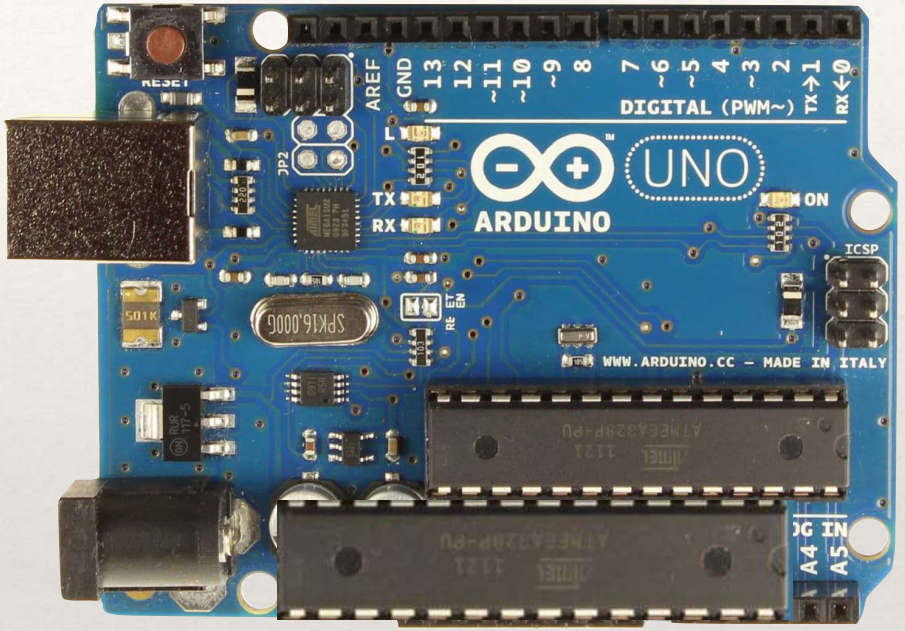


Placa de Protótipo EXP32

Microcontroladores

O Arduíno Uno v3 certamente é a placa de protótipo mais conhecida no mundo. Ela utiliza o microcontrolador ATMega328P.

O ATMega328P será estudado mais detalhadamente ao longo da disciplina.

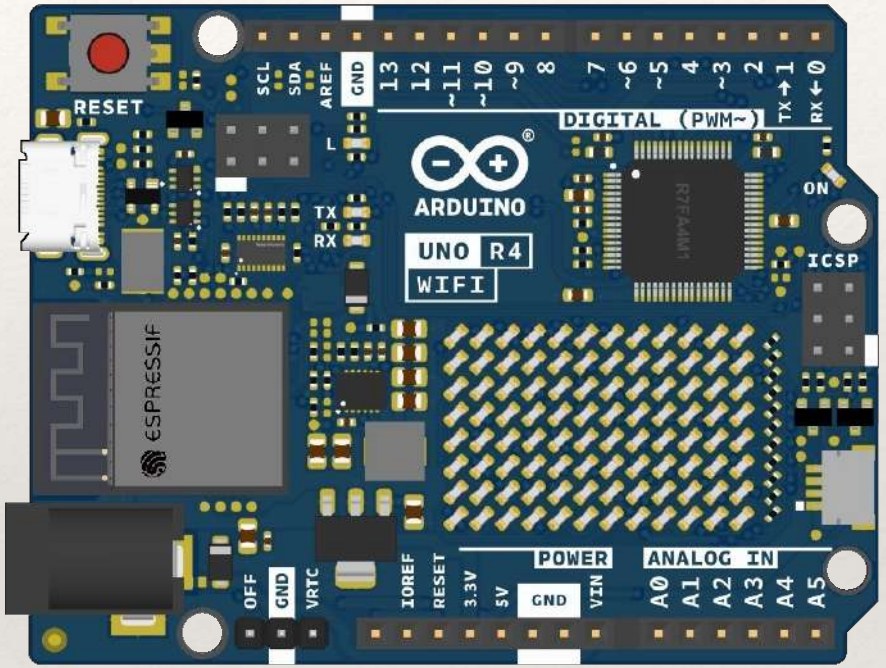


Microcontroladores

O Arduíno Uno v4 pretende suceder a versão 3. No entanto, ela utiliza um outro microcontrolador, o Renesas RA4M1 32-bit ARM® Cortex®;

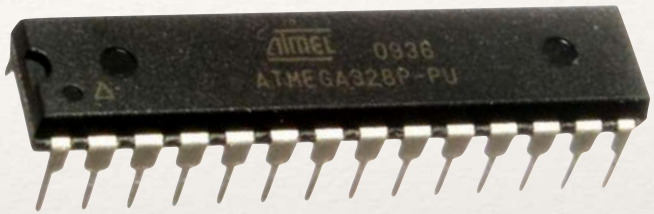
Todas as características específicas precisam ser revistas;

Por isto é muito importante diferenciar microcontroladores e placas de protótipo.



ATMega328P

Microcontrolador do Arduíno Uno



Bastante utilizado em todo o mundo em instituições de ensino, e por hobbistas, para projetos dos mais diversos tipos;

Desenvolvido pela Atmel, comprada pela Microchip em 2016, está disponível em formatos PTH e SMP;

No projeto Arduíno, o microcontrolador é instalado em uma **placa de protótipo** que oferece recursos adicionais, facilitando a implementação de projetos sem conhecimentos muito aprofundados de eletrônica;

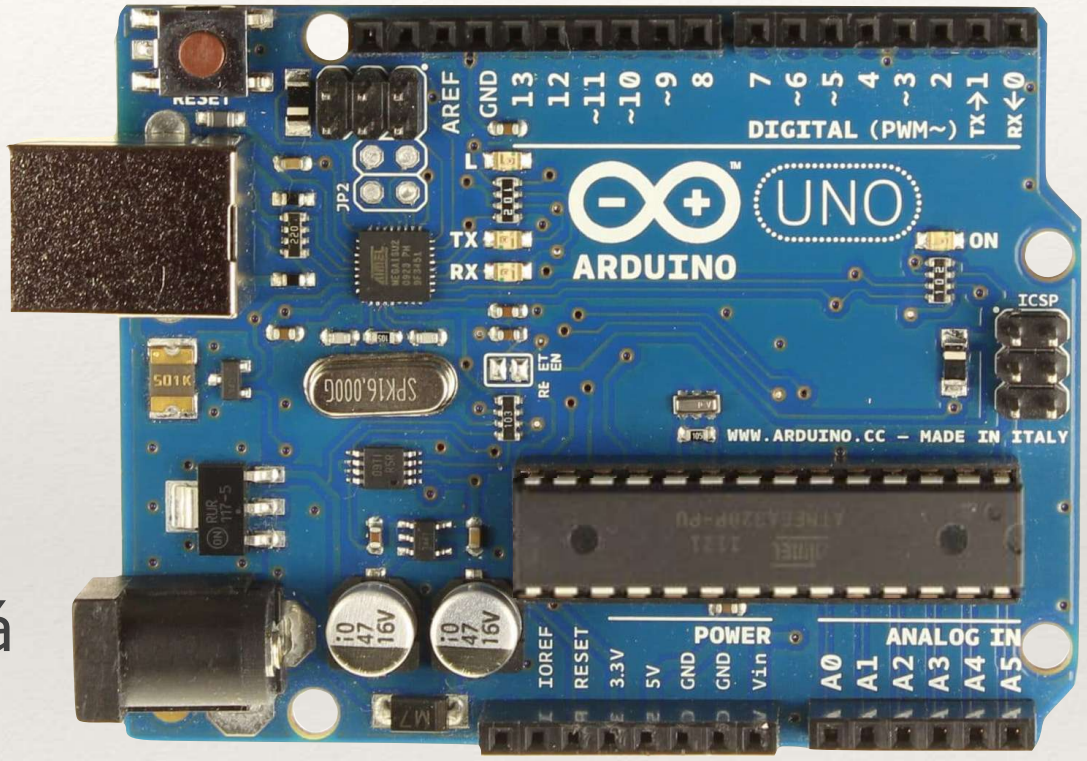
Arduíno, portanto, é o nome da placa de protótipo, e não do microcontrolador !

Conhecendo o Arduíno Uno R3

Tudo começou na Itália em 2005;

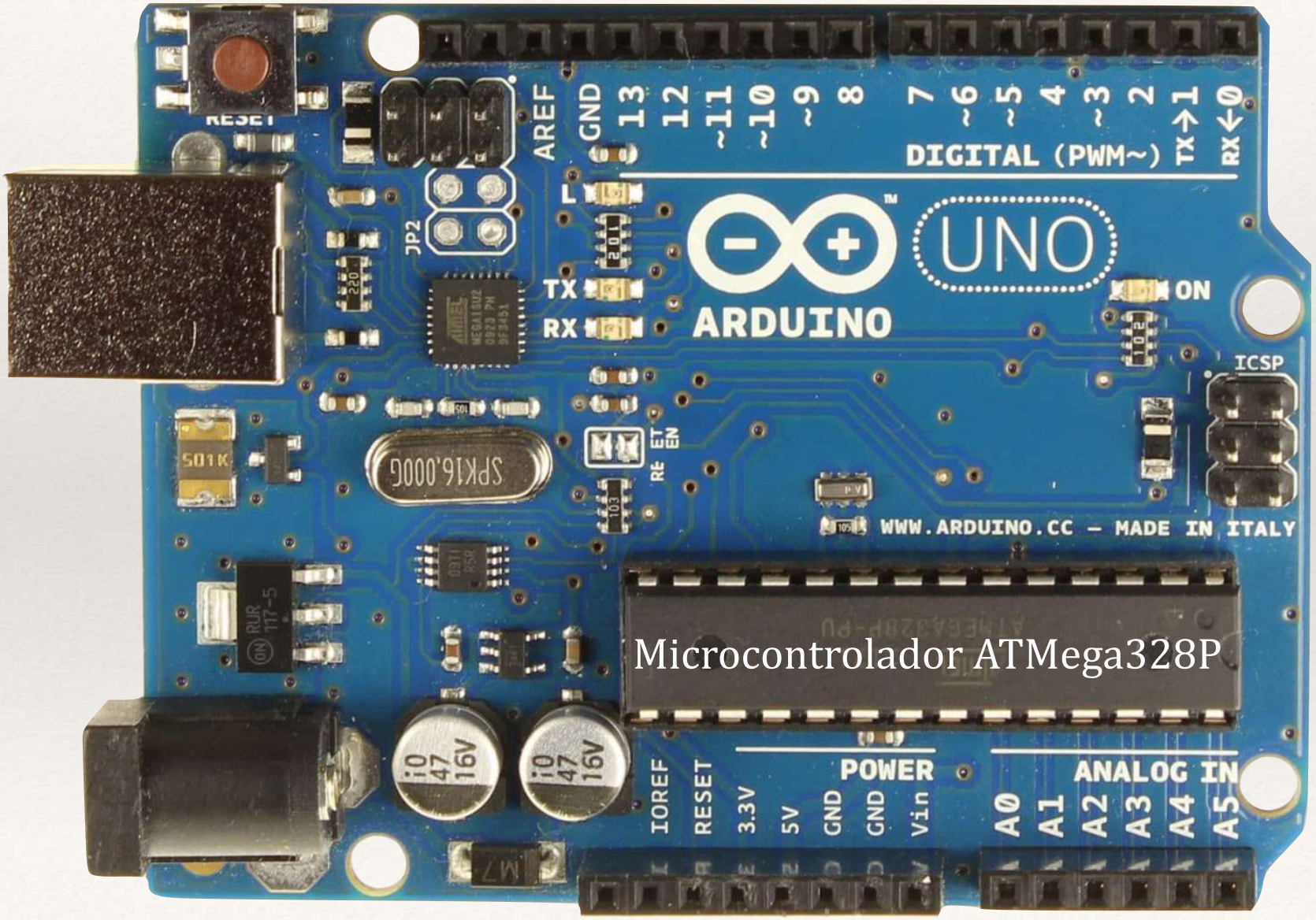
Tecnologia aberta (existem diversos fabricantes produzindo produtos similares);

O mais correto é chamar de "**Projeto Arduíno**", já que são diversos dispositivos;



Na foto, temos o **Arduíno Uno versão 3**. A versão mais atual desta placa é a 4. Além dela, o projeto tem diversos outros modelos com recursos variados.

Conhecendo o Arduíno Uno R3

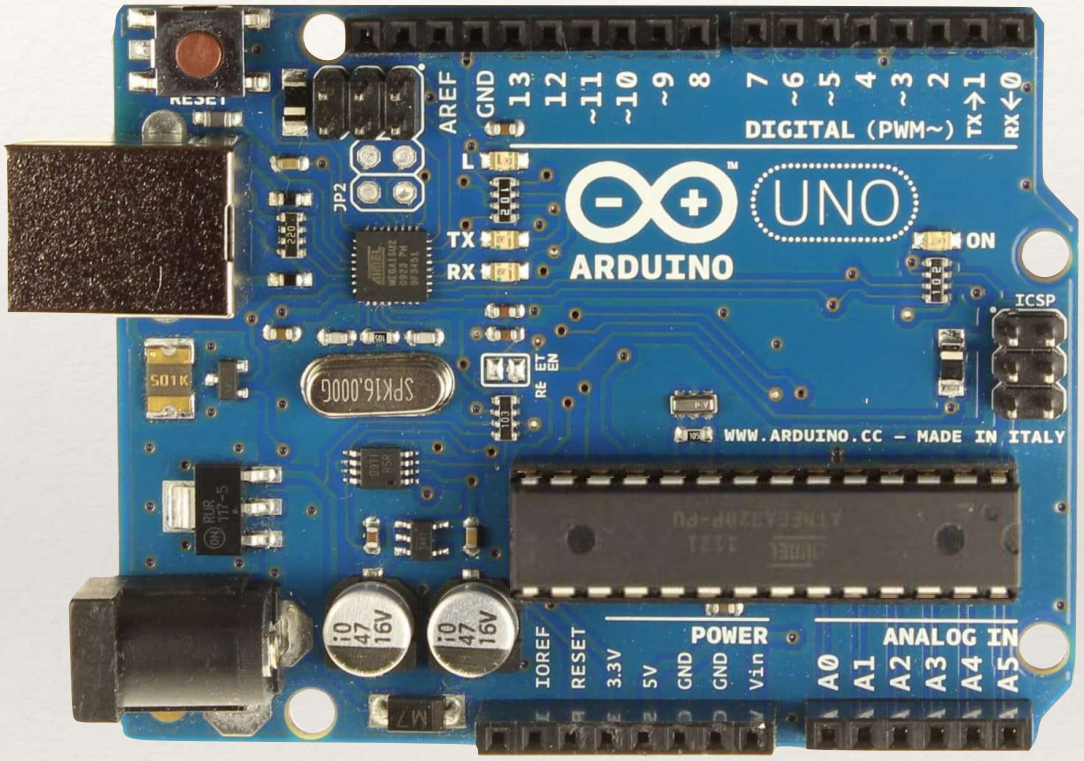


Conhecendo o Arduíno Uno R3

Uma placa de protótipo contém componentes adicionais, além do microcontrolador;

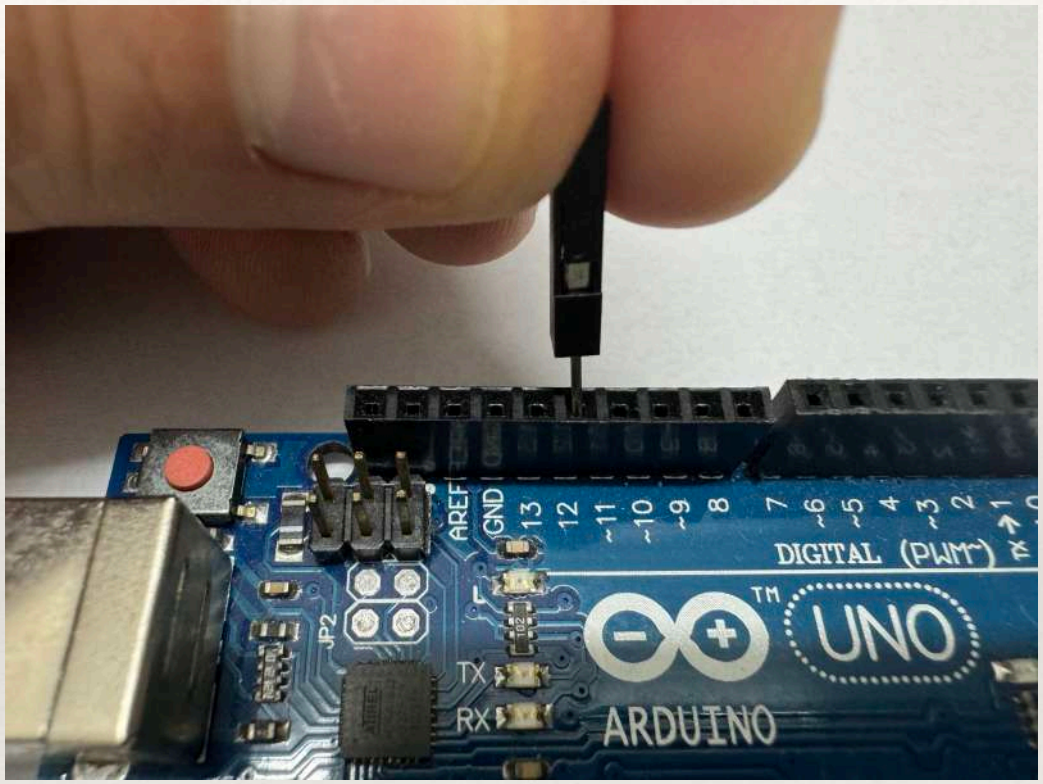
No caso do Arduíno Uno, temos:

- Conexões diversas;
- Botão de Reset;
- Controladora USB;
- *Clock* Externo a Quartzo;
- Regulador de Tensão (5V);
- Componentes eletrônicos diversos.



Conexões do Arduíno Uno R3

O Uno oferece conectores fêmea para fios *jumper Dupont*, sem uso de solda ou mesmo uma *protoboard*.



Embora simples, estas conexões têm alguns problemas:

- Contatos instáveis, especialmente para sistemas mais complexos;
- Baixa condutividade;
- Suportam correntes pequenas.

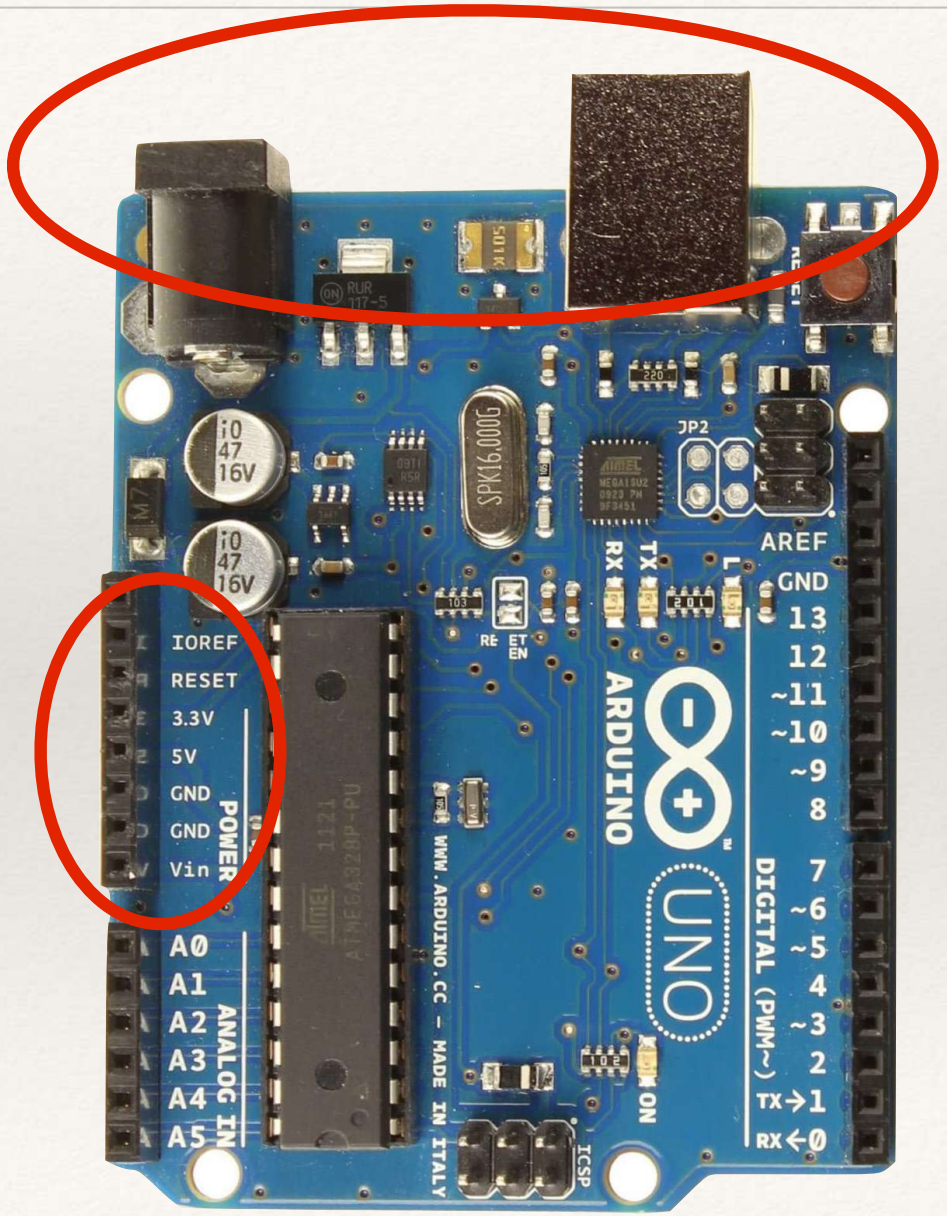
Por outro lado, outras placas de protótipo podem exigir o uso de *protoboards*. Projetos complexos podem exigir inclusive uma placa de circuito impresso, mesmo para protótipos.

Conexões do Arduíno Uno R3

Alimentação

Portas Analógicas

Portas Digitais

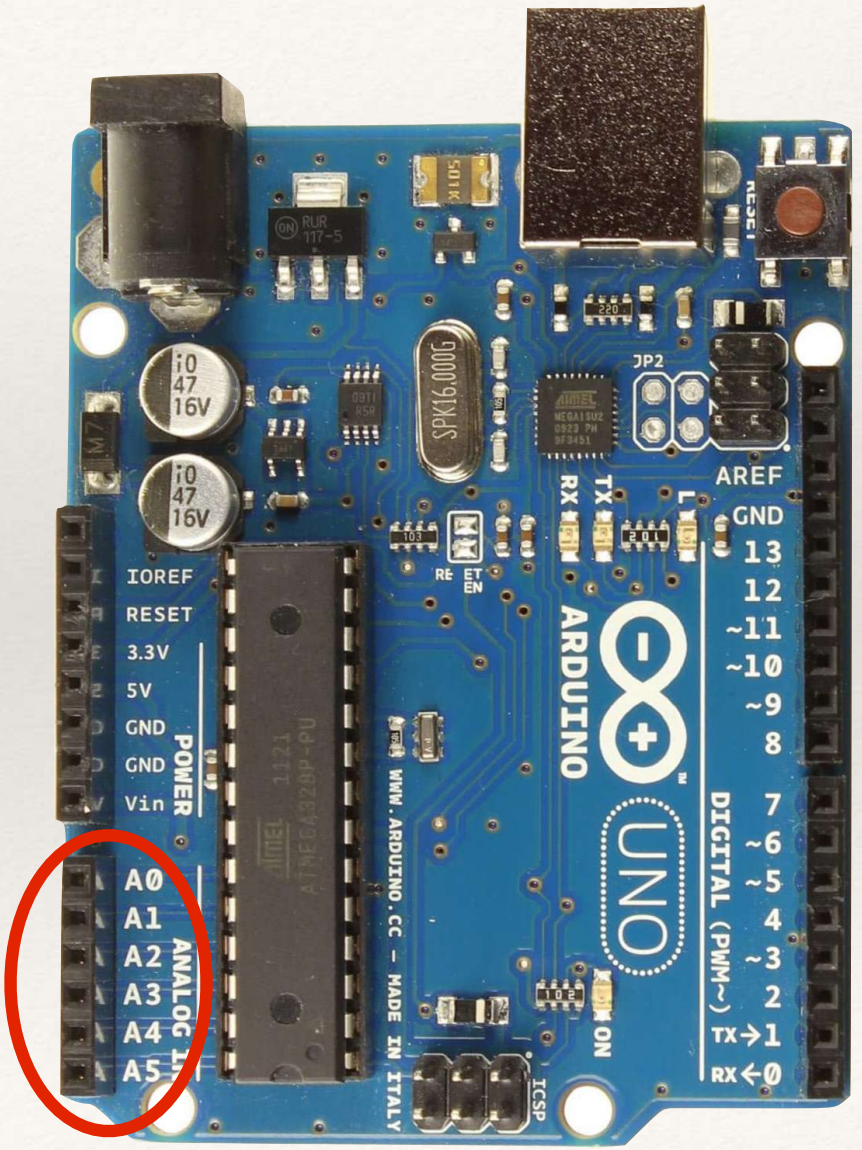


Conexões do Arduíno Uno R3

Alimentação

Portas Analógicas

Portas Digitais

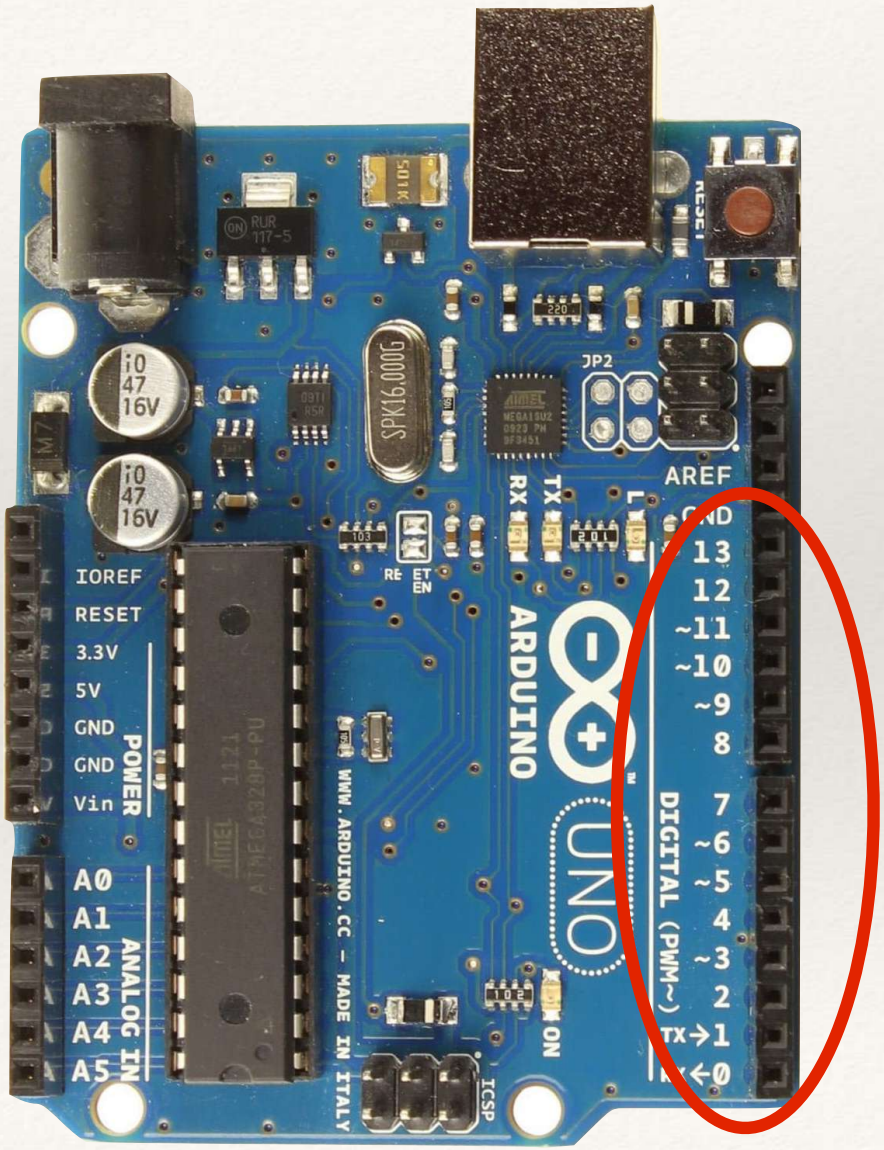


Conexões do Arduíno Uno R3

Alimentação

Portas Analógicas

Portas Digitais



Conexões do Arduíno Uno R3

Alim. / Comunicação

Entrada USB

Provê comunicação com o computador (e alimentação).

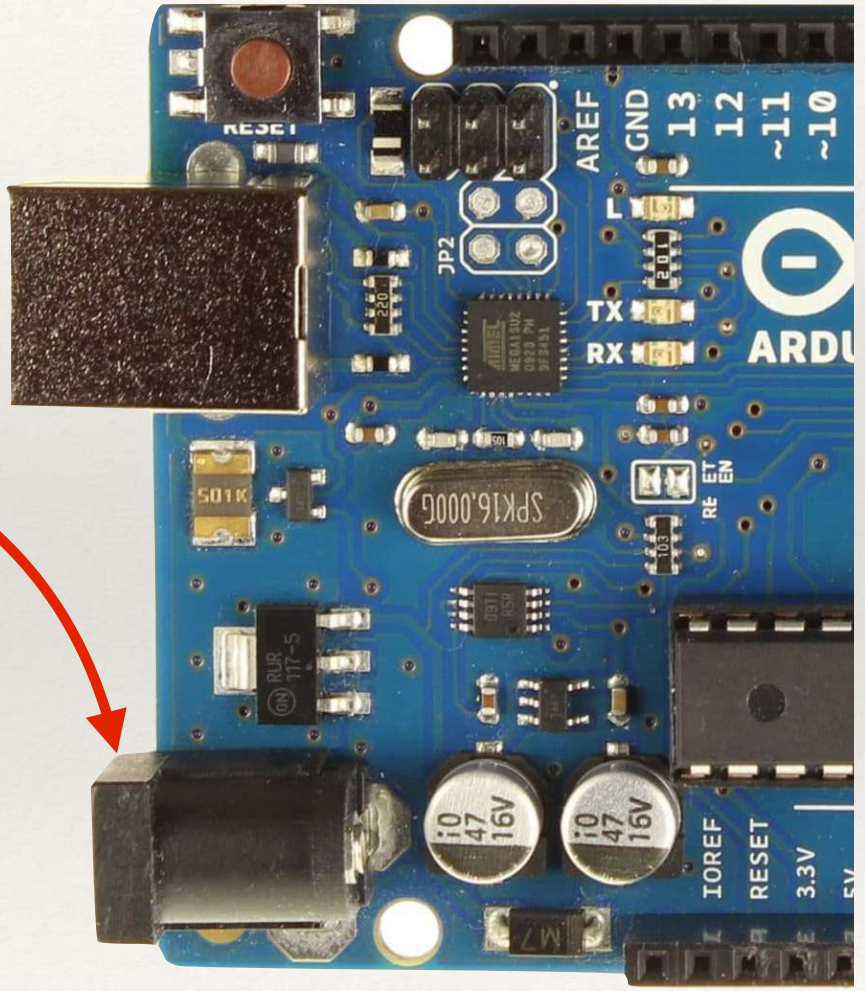
Entrada Fonte DC Externa

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

Não se recomenda >12V (aquecimento componentes).

Portas Analógicas

Portas Digitais



Conexões do Arduíno Uno R3

Alimentação (pinos)

Saídas de 5V e 3,3V

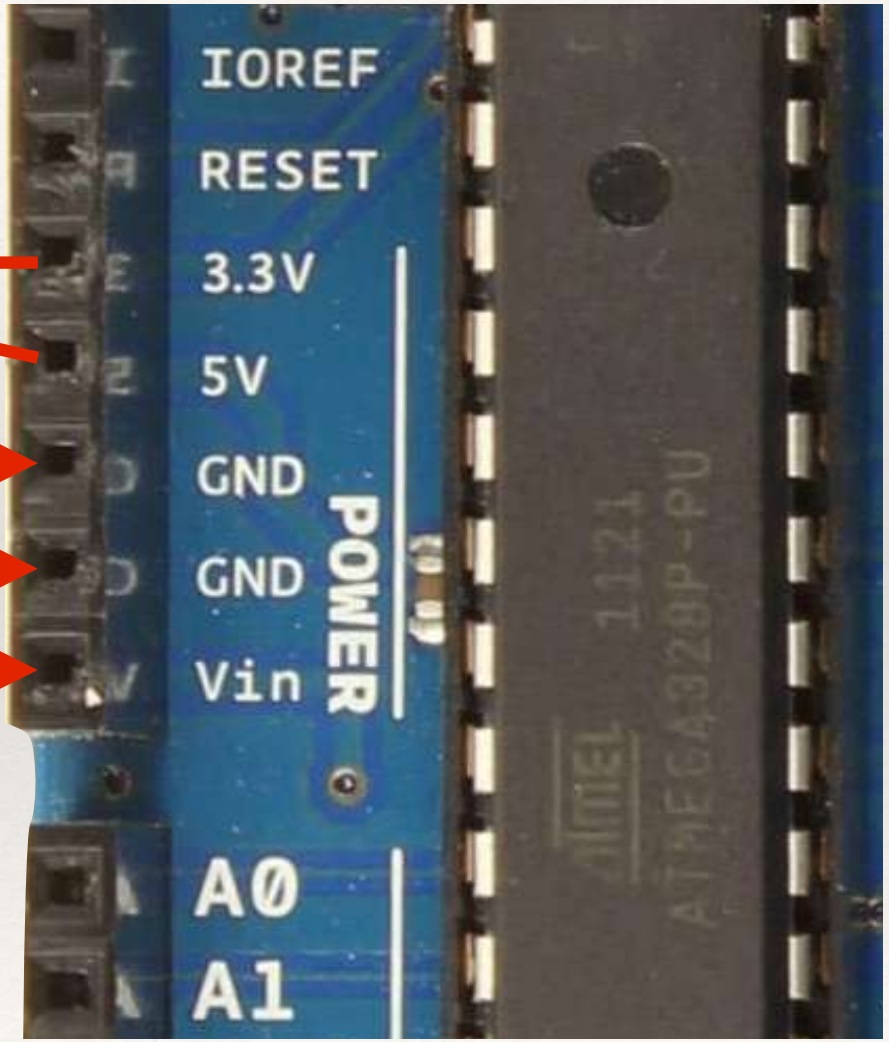
Corrente máxima de 40mA

Entrada Vin

Alimentação de 5V ≥100mA

Portas Analógicas

Portas Digitais

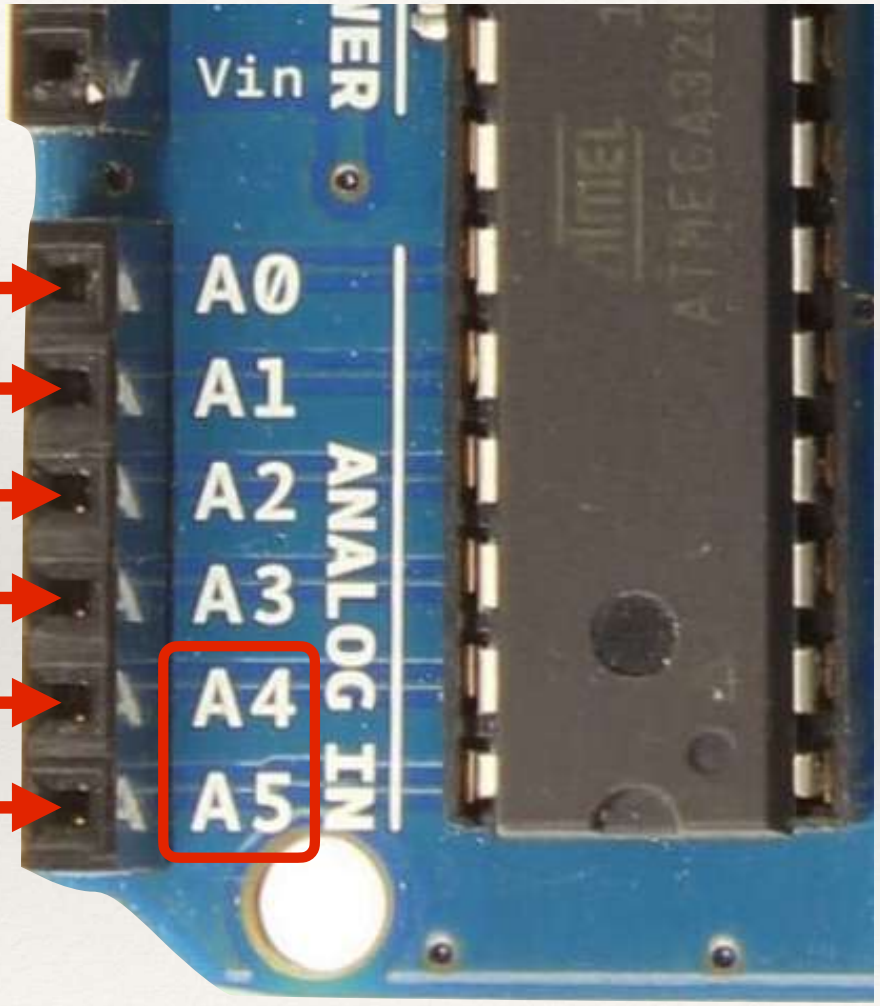


Conexões do Arduíno Uno R3

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

Conexões do Arduíno Uno R3

A

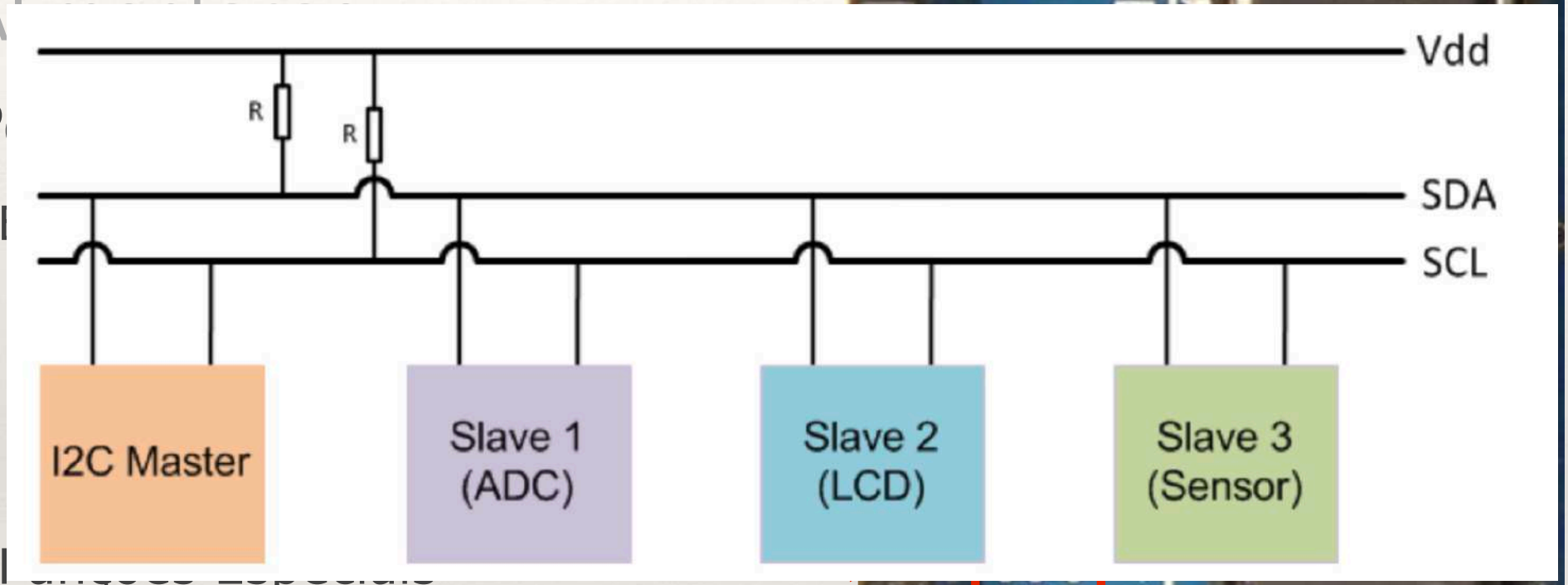
P

I

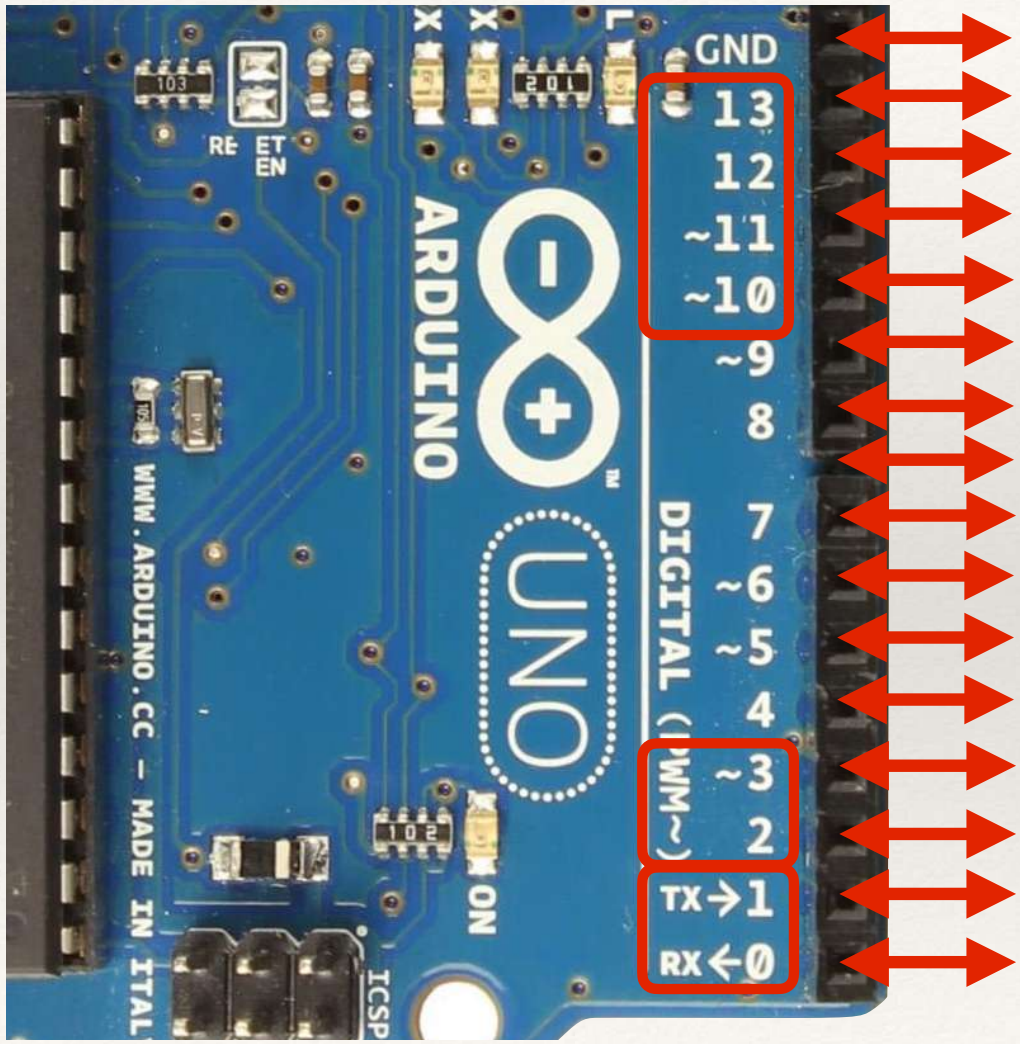
Funções Especiais

I2C (SCL e SDA): portas A5 e A4

Portas Digitais



Conexões do Arduíno Uno R3



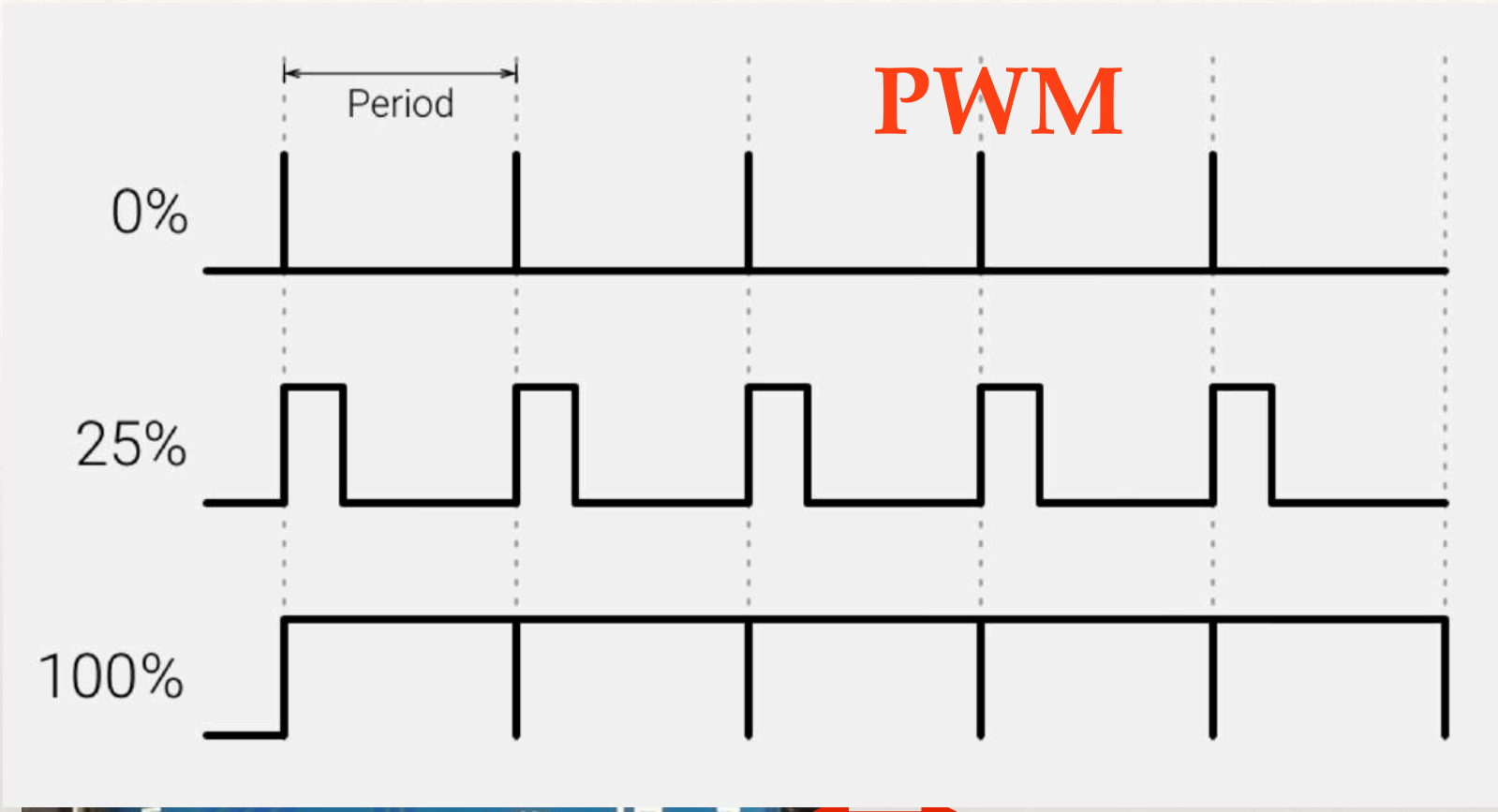
Alimentação

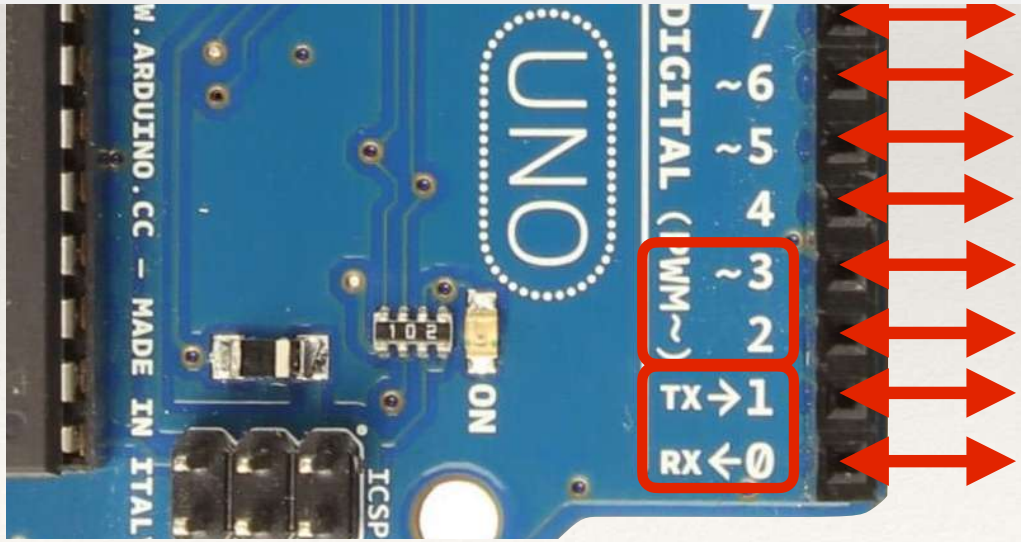
Portas Analógicas

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





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

< 20mA (máx.40 mA); total 200mA

Funções especiais

UART: Rx/Tx (portas 0 e 1)

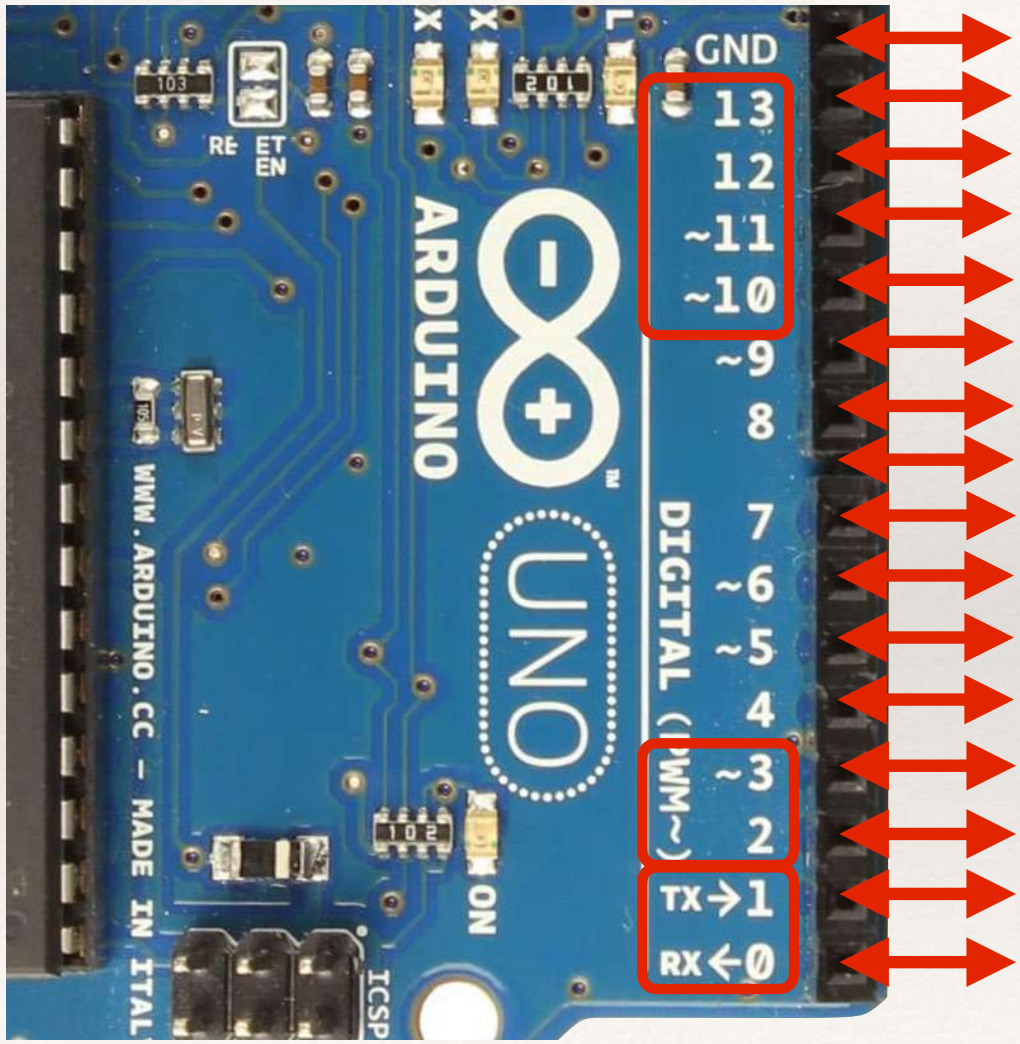
Interrupções: INT0 e 1 (portas 2 e 3)

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

Pino 13: LED interno

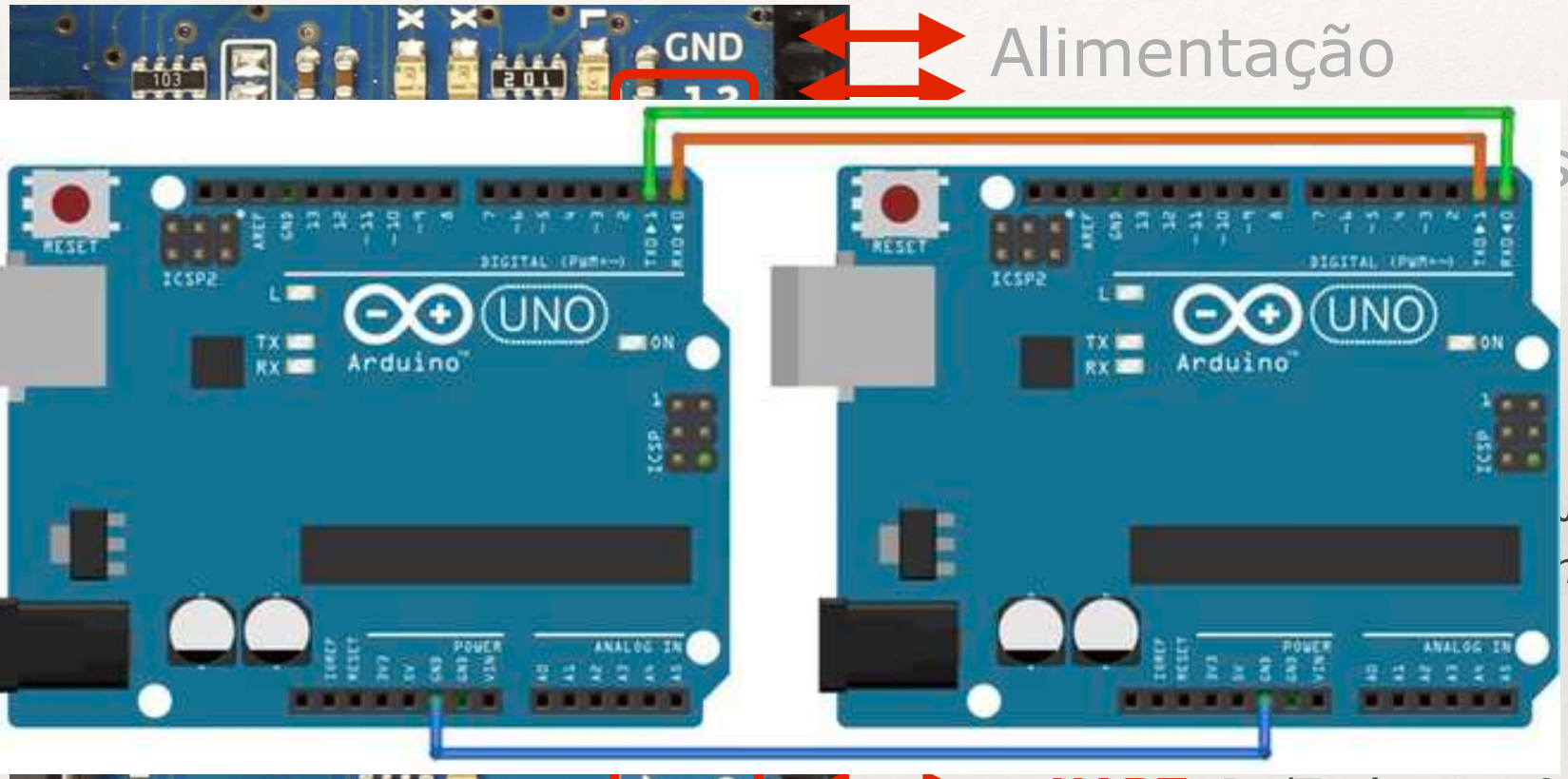
Aula 03

Conexões do Arduíno Uno R3



- 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

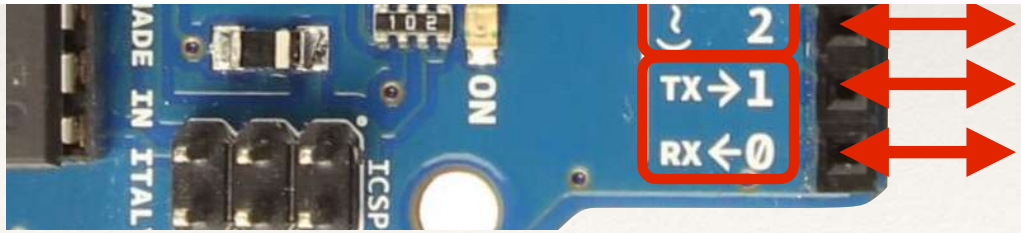
Conexões do Arduíno Uno R3



Alimentação

~ são PWM

total 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

Conhecendo o Arduíno Uno R3

Lembrar que:

- A porta USB é prioritariamente uma porta de comunicação, e não de alimentação !
- Apenas pequenos projetos devem ser alimentados pela placa de protótipo;
- Eventuais problemas elétricos podem afetar o computador conectado.

Conhecendo o Arduíno Uno R3

O *clock* sincroniza processador, barramentos e periféricos;

Performance:

- Se for baixa, pode tornar a operação lenta demais - porém reduz consumo;
- Se for alta, pode impor tempos de resposta impossíveis para os componentes - porém é mais ágil;

Estabilidade:

- Se não existirem operações síncronas, as exigências são baixas (1 ~ 5 %);
  - ★ **Um relógio atrasa até 14 minutos por dia com 1% de estabilidade.**
- Para aplicações síncronas, é um item crítico;
  - ★ **Ex: ADC, DAC e comunicação de dados.**

Conhecendo o Arduíno Uno R3

Existem algumas opções para implementação do *clock*:

- Oscilador RC interno;
- Oscilador RC externo;
- Oscilador com cristal de quartzo.

As opções diferem na estabilidade, tamanho, consumo e custo;

Fatores externos (ou internos) afetam a estabilidade:

- Temperatura, alimentação, interferências elétricas e vibração;

O Arduíno Uno R3 usa um cristal de 16MHz em seu oscilador.

Conhecendo o Arduíno Uno R3

A alimentação é crítica:

- Queda de tensão por descarga da bateria, ou falhas de projeto, podem provocar alteração de comportamento;
- Ruído injetado na alimentação é uma das fontes de instabilidade mais difíceis de diagnosticar. O próprio circuito digital é uma fonte de ruído;
- Um surto de tensão, mesmo pequeno, pode danificar diversos componentes.

Como evitar problemas?

- Escolha cuidadosa das fontes de alimentação;
- Capacitores de desacoplamento individuais para cada CI, instalados bem próximos dos mesmos.

Microcontroladores

Microcontrolador	PIC10F322	ATTiny85	ATMega328P	ESP32 SoC
Arquitetura	8 bits	8 bits	8 bits	32 bits Dual-Core
GPIO	4	6	23	34
Program Memory (Flash)	512 B	8 KB	32 KB	16 MB
Data Memory (SRAM)	64 B	512 B	2 KB	520 KB
EEPROM	128 B	512 B	1 KB	4MB
Clock Máximo	16 MHz	20 MHz	16 MHz	240 MHz
UART	Não	Não	1	3
i2C	Não	Não	1	2
SPI	Não	1	1	4
PWM	2	2	6	16
ADC	3 x 8-bits	4 x 10-bits	8 x 10-bits	18 x 12-bits
Contadores	2 x 8-bits	2 x 8-bits	2 x 8-bits + 1 x 16-bits	8 x 16-bits
Consumo (modo ativo)	25uA (1 MHz)	300uA (1 MHz)	1,5mA (4 MHz)	260mA
Custo unitário (US$)	0,89	1,42	2,66	3,46

Eletrônica em SE

Desafios de um projeto de HW

- Não é um conhecimento nativo de um engenheiro de *software*;
- A rotina de desenvolvimento em HW é bem diferente;
- Testes podem ser destrutivos - a verificação prévia precisa ser exaustiva.

Integração HW & SW

- HW instável provocará problemas, independente da qualidade do SW;
- Problemas no SW podem danificar o HW.

Diversidade de componentes

- Ao contrário do desenvolvimento de SW, não há um conjunto fixo de instruções e comandos;
- Surgem novos sensores, atuadores e protocolos quase diariamente.

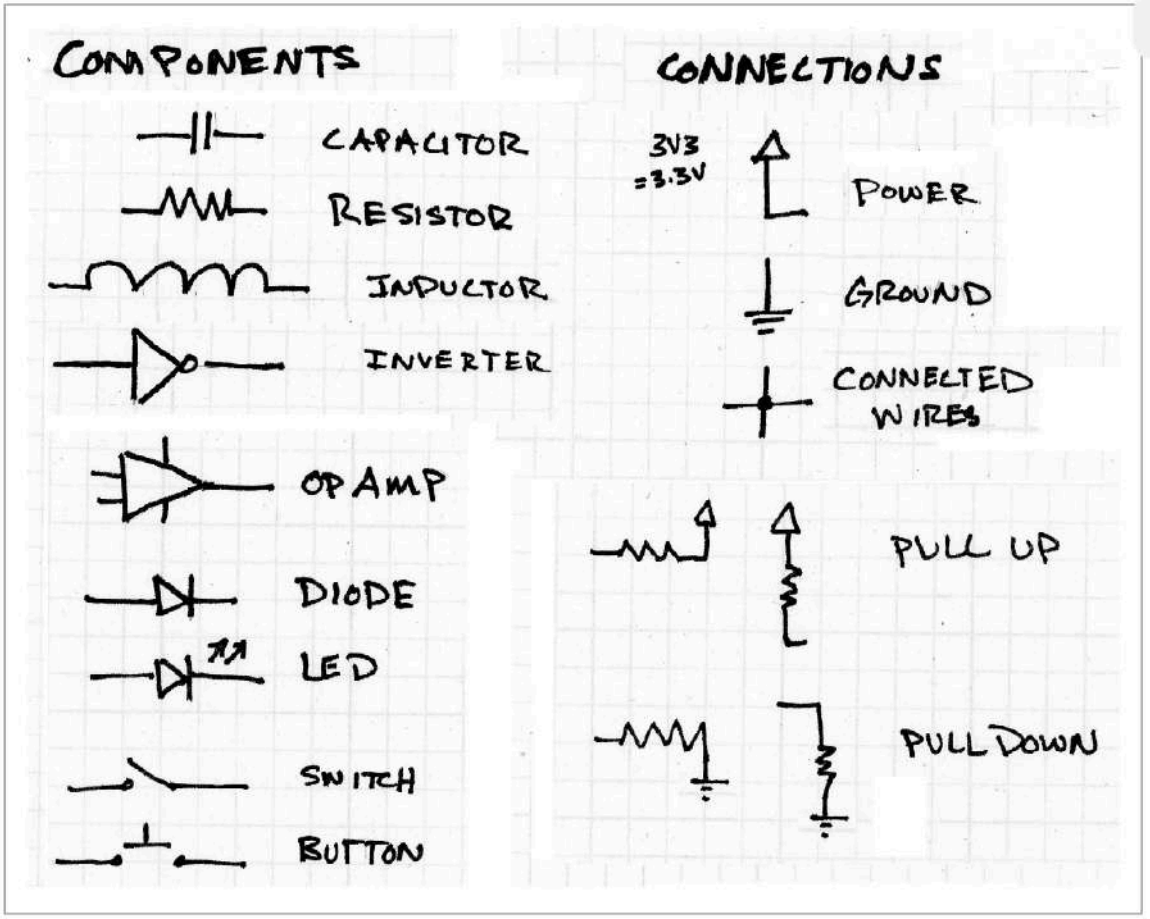
Conceitos Básicos

Grandezas:

- Tensão, Corrente, Campos elétrico e magnético;

Componentes:

- Resistores, Capacitores e Indutores;
- Diodos e transistores;
- Amplificadores Operacionais;
- Portas Lógicas;



Tensão elétrica

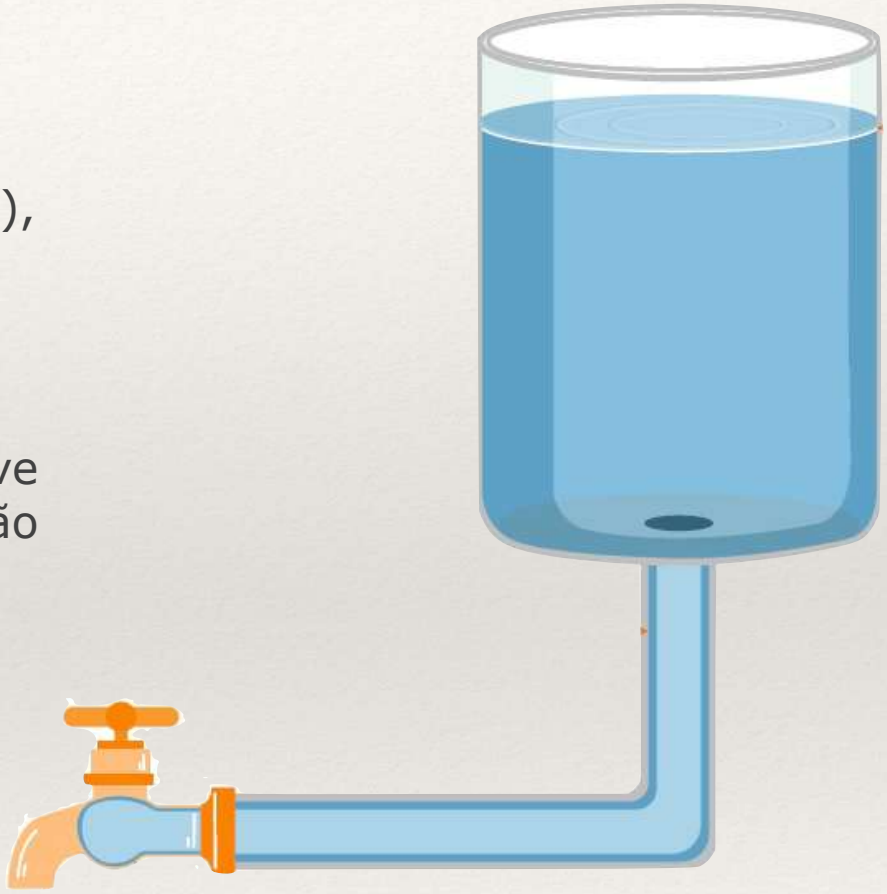
É a diferença de potencial entre dois pontos, medida em Volts

- Tipicamente um dos pontos é o “terra" (GND), com a tensão de zero volts;
- Quando contínua, pode ser positiva ou negativa; Também pode ser alternada;
- Provoca a circulação de corrente, que se move do ponto de tensão mais alto para o de tensão mais baixa.

Pode ser causada por 3 efeitos

- Campo elétrico estático;
- Campo eletromagnético variável;
- Corrente sob ação de um campo magnético.

Analogia mecânica: energia potencial.



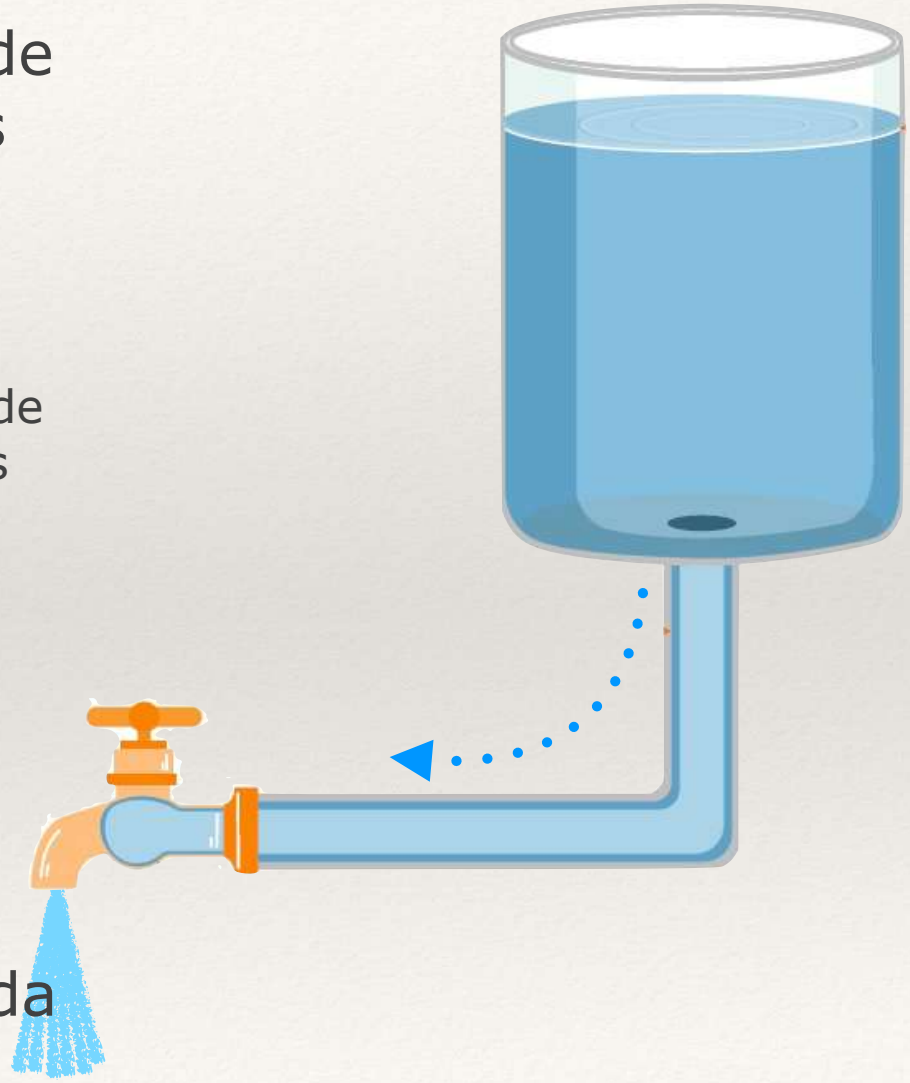
Corrente elétrica

Fluxo ordenado de cargas dentro de um condutor, medido em Ampères

- Para ser condutor, o material tipicamente tem elétrons livres em movimento desordenado;
- Ocorre quando provocada pela existência de uma tensão elétrica entre as extremidades do condutor;
- Quando contínua, o sentido do fluxo determina o seu sinal; pode também ser alternada.

Analogia mecânica: energia cinética;

Resistores se opõem à circulação da corrente elétrica.



Campo Elétrico

É um campo de força criado pela eletricidade estática

- Se inicia na carga positiva e termina na carga negativa;
- É provocado pela ação de cargas elétricas (elétrons, prótons e íons);

Um campo elétrico variável provoca um campo magnético, e vice-versa;

Capacitores se opõem à alteração de um campo elétrico.

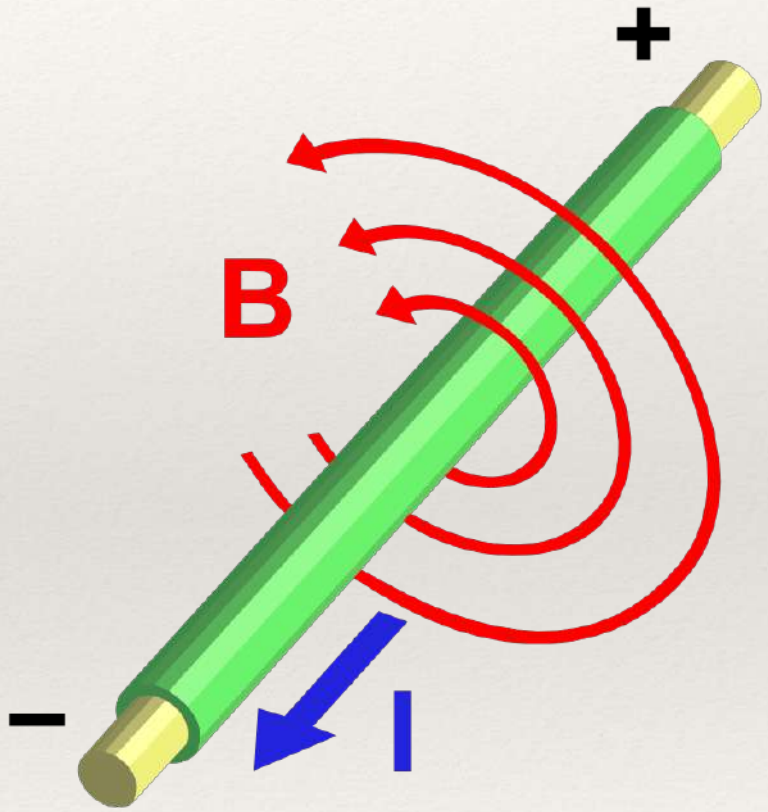
Campo Magnético

É um campo de força criado por materiais ferromagnéticos ou pela circulação de corrente elétrica

- Formam círculos concêntricos perpendiculares à circulação da corrente;
- A orientação do campo pode ser representada pela “regra da mão direita".

Um campo elétrico variável provoca um campo magnético, e vice-versa;

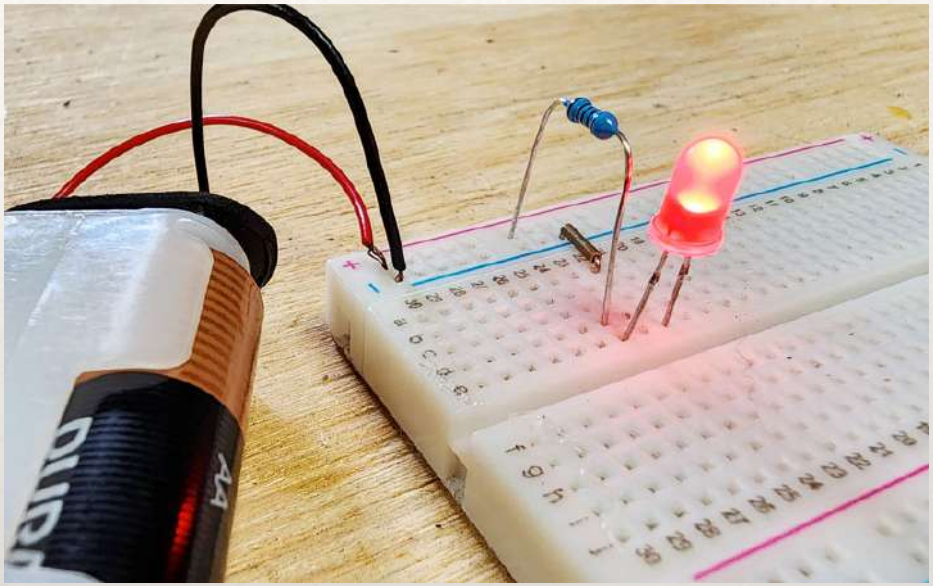
Indutores se opõem à alteração de um campo magnético.



Resistor

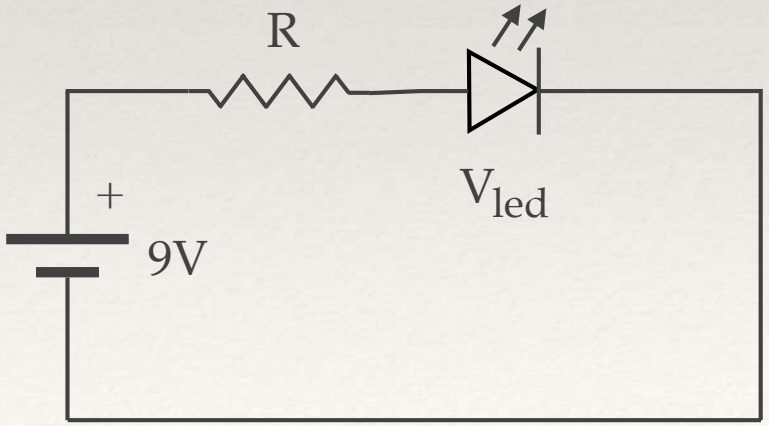
Oferece resistência à passagem da corrente, e é medido em ohms (Ω)

- Remove energia do circuito transformando-a em potência térmica, que portanto precisa ser dissipada;
- Reduz a corrente em circulação;
- Provoca uma queda de tensão entre as suas extremidades.

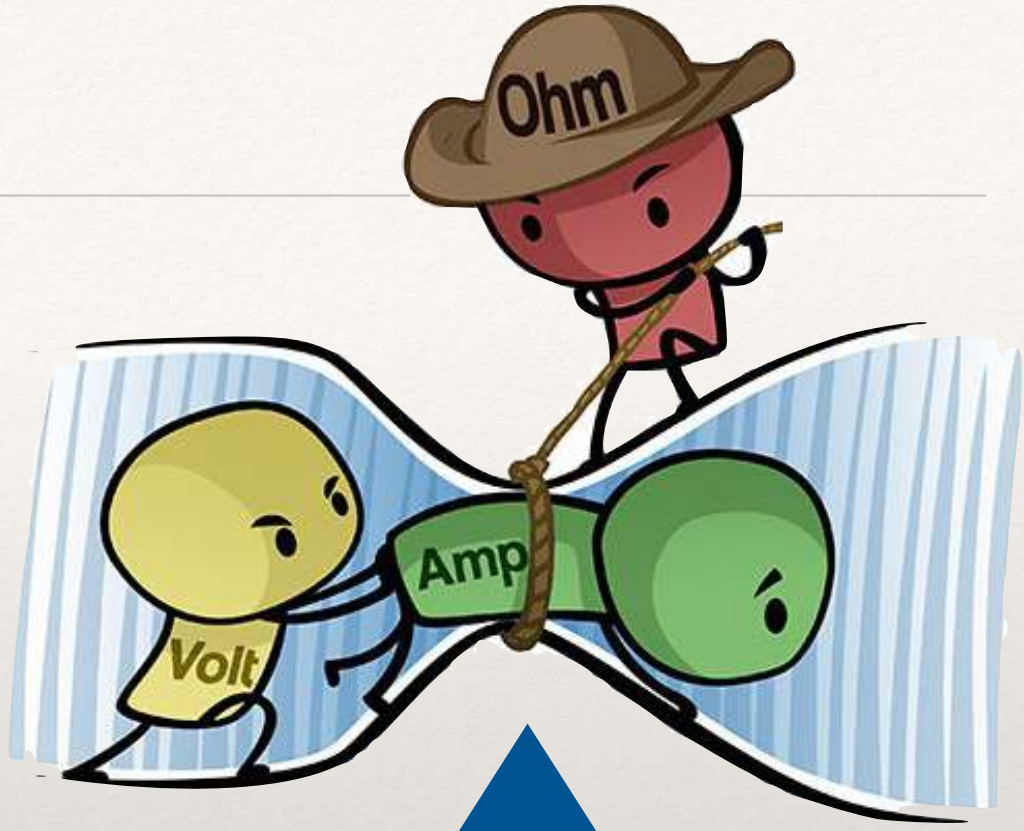


Está relacionado à tensão e a corrente em um circuito com base na Lei de Ohm

- Desenvolvida pelo físico alemão Georg Simon Ohm em 1827, baseada no trabalho de Fourier;



Resistor

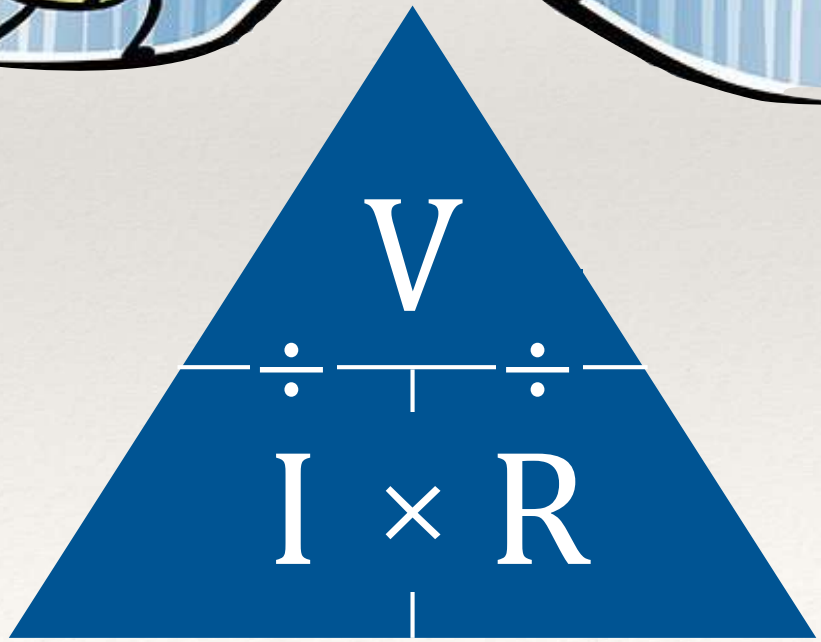


Oferece resistência à passagem da corrente, e é medido em ohms (Ω)

- Remove energia do circuito transformando-a em potência térmica, que portanto precisa ser dissipada;
- Reduz a corrente em circulação;
- Provoca uma queda de tensão entre as suas extremidades.

Está relacionado à tensão e a corrente em um circuito com base na Lei de Ohm

- Desenvolvida pelo físico alemão Georg Simon Ohm em 1827, baseada no trabalho de Fourier;



Tipos de Resistores

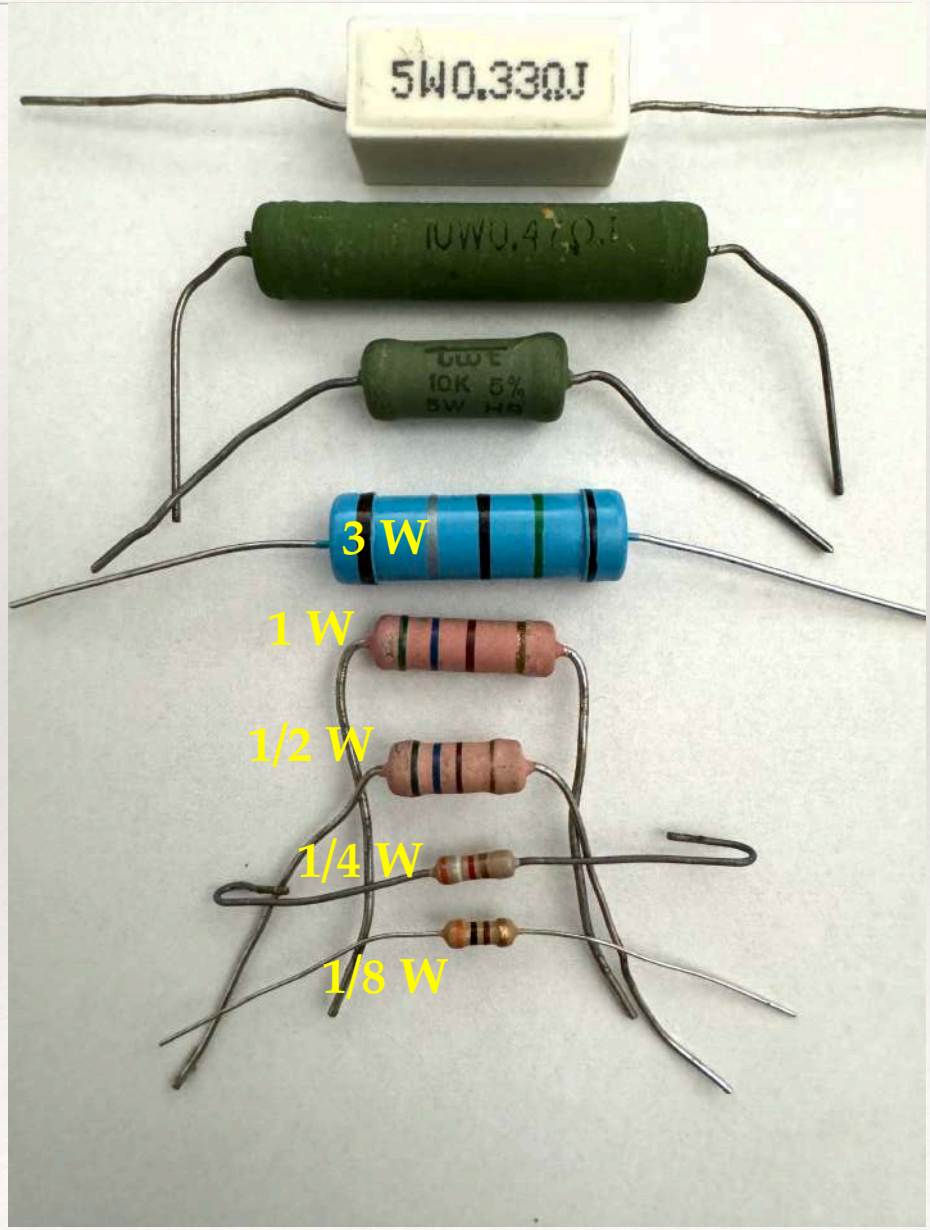
Resistores PTH - Potência

Tem relação com o tamanho

- Resistores menores dissipam menos potência;
- A temperatura (inclusive externa) afeta o valor da resistência !

Em alguns casos, pode estar gravada no corpo;

Projetos devem contemplar operação abaixo do limite.



Resistores SMD - Potência

Invólucro	Tamanho em polegadas	Tamanho em milímetros	Potência
0201	0,024 "× 0,012"	0,6 milímetros × 0,3 milímetros	1/20W
0402	0.04 "× 0,02"	1,0 milímetro × 0,5 milímetros	1/16W
0603	0,063 "× 0,031"	1,6 milímetros × 0,8 milímetros	1/16W
0805	0,08 "× 0,05"	2,0 milímetro × 1,25 milímetros	1/10W
1206	0,126 "× 0,063"	3,2 milímetro × 1,6 milímetros	1/8W
1210	0. 12 "× 0,10",	3,2 milímetros × 2,6 mm	1/4W
1812	0, 18 "× 0,12",	4,6 mm × 3,0 milímetro	1/3W
2010	0,20 "× 0,10"	5,08 milímetros × 2,6 mm	1/2W
2.512	0,25 "× 0,12"	6,35 milímetros × 3,0 milímetros	1W

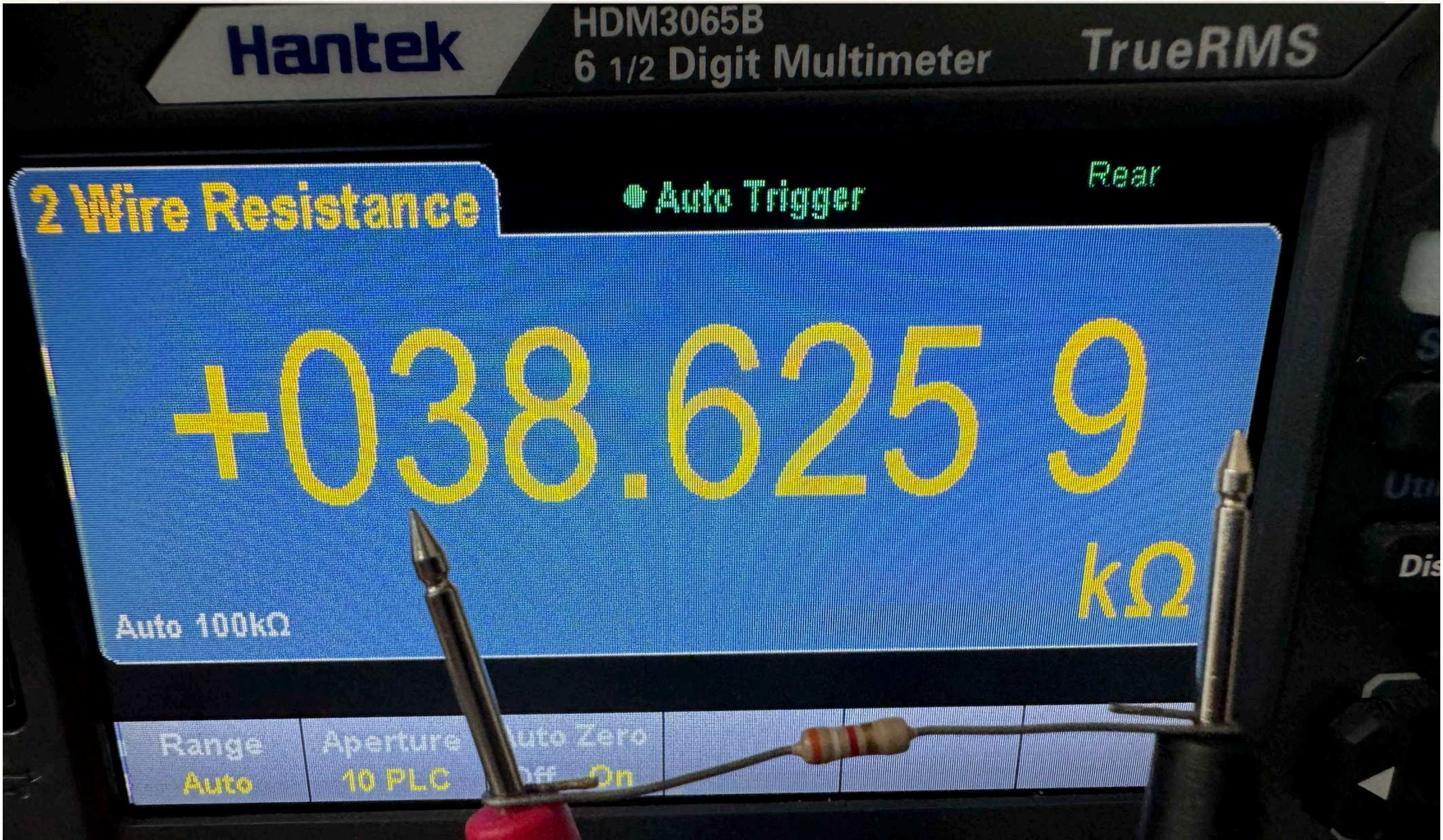
Resistores - Resistência em Ω

Cor	1ª Faixa	2ª Faixa	3ª Faixa	Multiplicador	Tolerância
Preto	0	0	0	x 1 Ω	
Marrom	1	1	1	x 10 Ω	+/- 1%
Vermelho	2	2	2	x 100 Ω	+/- 2%
Laranja	3	3	3	x 1K Ω	
Amarelo	4	4	4	x 10K Ω	
Verde	5	5	5	x 100K Ω	+/- .5%
Azul	6	6	6	x 1M Ω	+/- .25%
Violeta	7	7	7	x 10M Ω	+/- .1%
Cinza	8	8	8		+/- .05%
Branco	9	9	9		
Dourado				x .1 Ω	+/- 5%
Prateado				x .01 Ω	+/- 10%

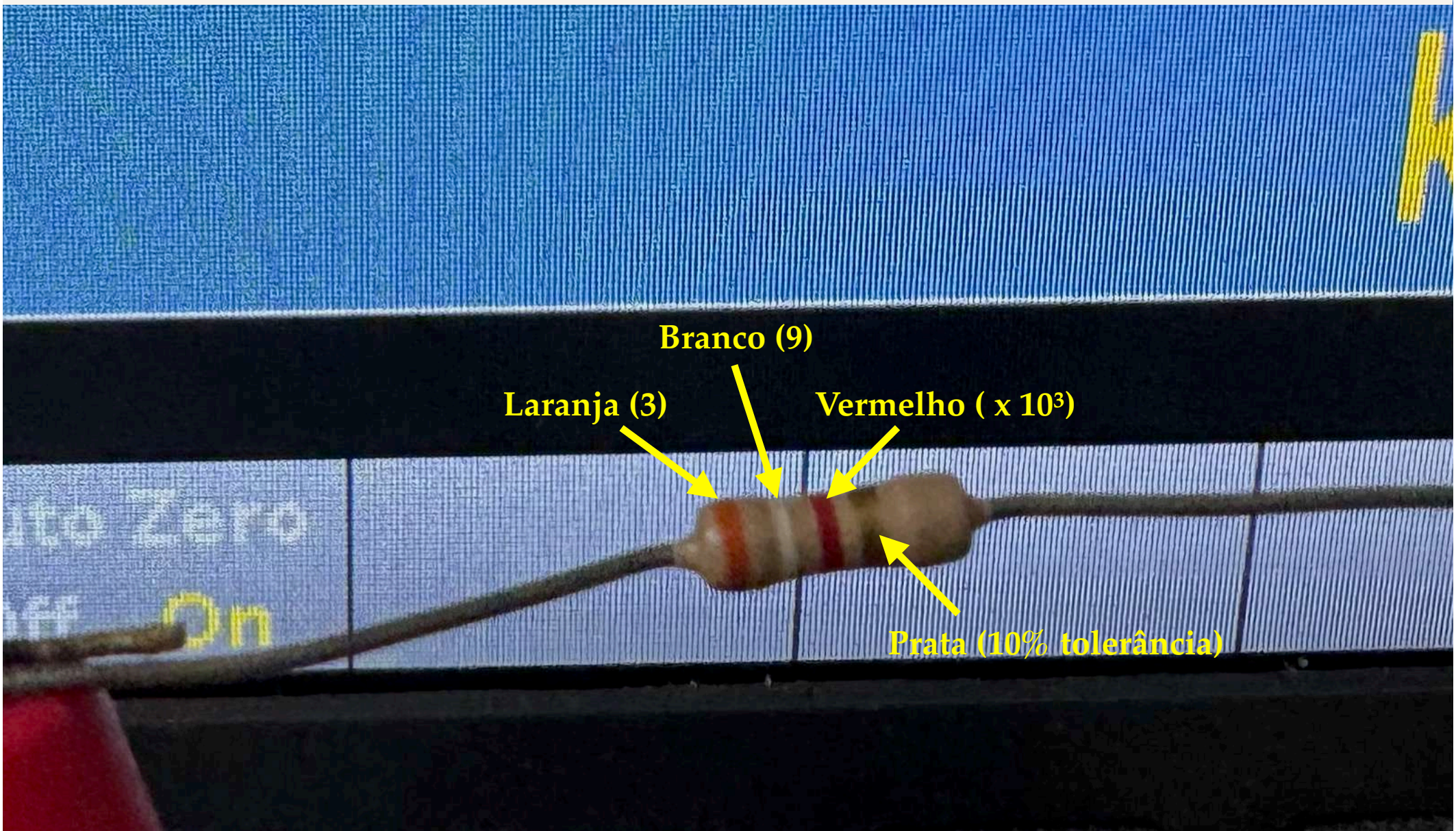
(código válido para maior parte dos resistores PTH)

3 faixas (20% tolerância), 4 faixas (comum), 5 faixas (precisão)

Resistores - Resistência em Ω



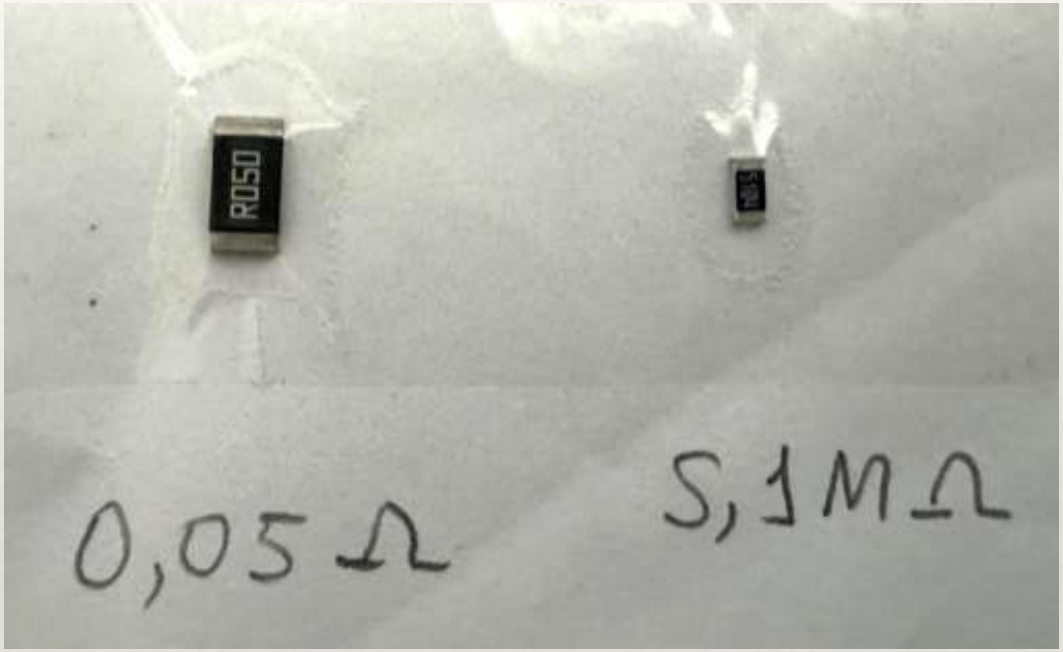
Resistores - Resistência em Ω



Resistores - Resistência em Ω

Em resistores SMD, o valor normalmente vem gravado no corpo

- Tipicamente usam um código numérico de 3 dígitos;
- Tal como as duas primeiras faixas de um resistor PTH, os dois primeiros números indicam os dígitos significativos, o terceiro é o multiplicador;
- "104" por exemplo é igual a 100KΩ.
- Se houverem quatro dígitos, vale a regra para resistores PTH com cinco faixas. Os três primeiros são dígitos significativos, e o último é o multiplicador.



Resistores - Resistência em Ω

Em resistores SMD, o valor normalmente vem gravado no corpo

- Tipicamente usam um código numérico de 3 dígitos;
- Tal como as duas primeiras faixas de um resistor PTH, os dois primeiros números indicam os dígitos significativos, o terceiro é o multiplicador;
- "104" por exemplo é igual a 100KΩ.
- Se houverem quatro dígitos, vale a regra para resistores PTH com cinco faixas. Os três primeiros são dígitos significativos, e o último é o multiplicador.

0,05 Ω - 1W

Tamanho 2512 (6,35 x 3mm)

Resistores - Resistência em Ω

Em resistores SMD, o valor normalmente vem gravado no corpo

- Tipicamente usam um código numérico de 3 dígitos;
- Tal como as duas primeiras faixas de um resistor PTH, os dois primeiros números indicam os dígitos significativos, o terceiro é o multiplicador;
- "104" por exemplo é igual a 100KΩ.
- Se houverem quatro dígitos, vale a regra para resistores PTH com cinco faixas. Os três primeiros são dígitos significativos, e o último é o multiplicador.

5,1 MΩ - 1/4W

Tamanho 1206 (3,2 x 1,6mm)

Aula 04

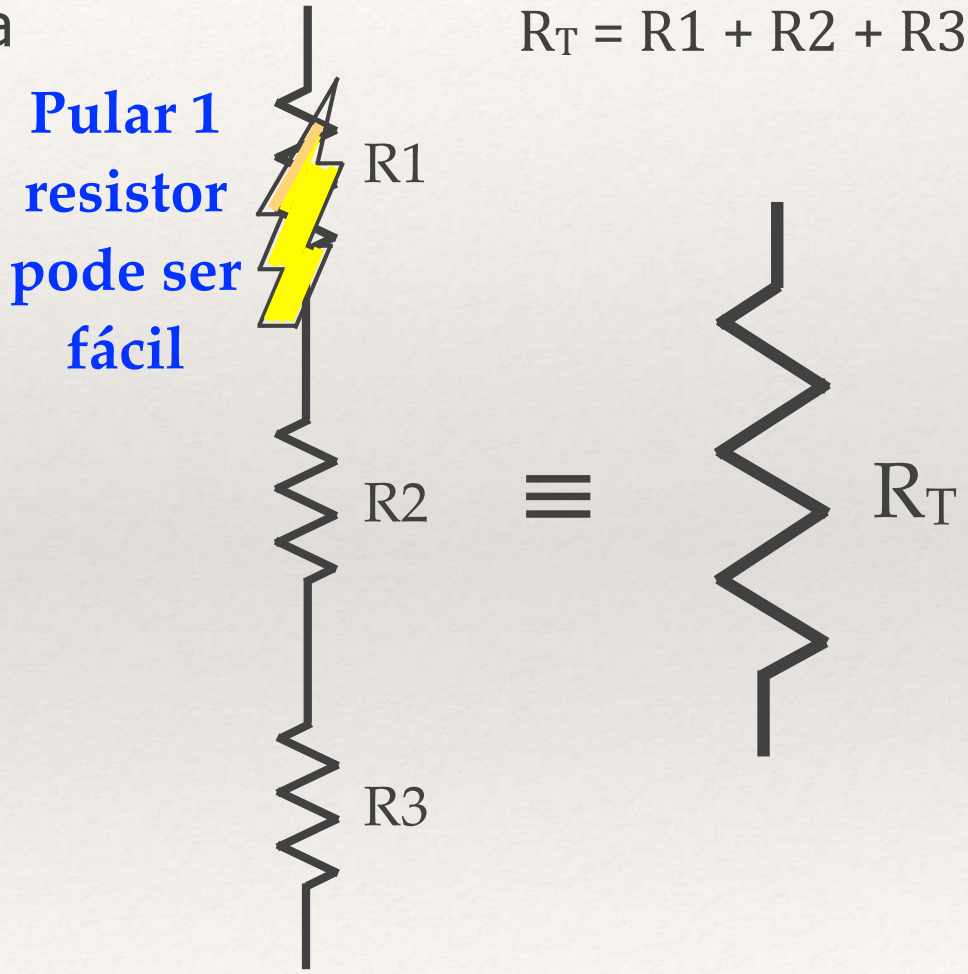
Resistores em Série

A resistência total é igual à soma das resistências individuais;

O arranjo permite:

- Distribuir a potência dissipada;
- Substituir resistências individuais;
- Aumentar a proteção contra descargas elétricas.

$$R_T = R1 + R2 + R3$$



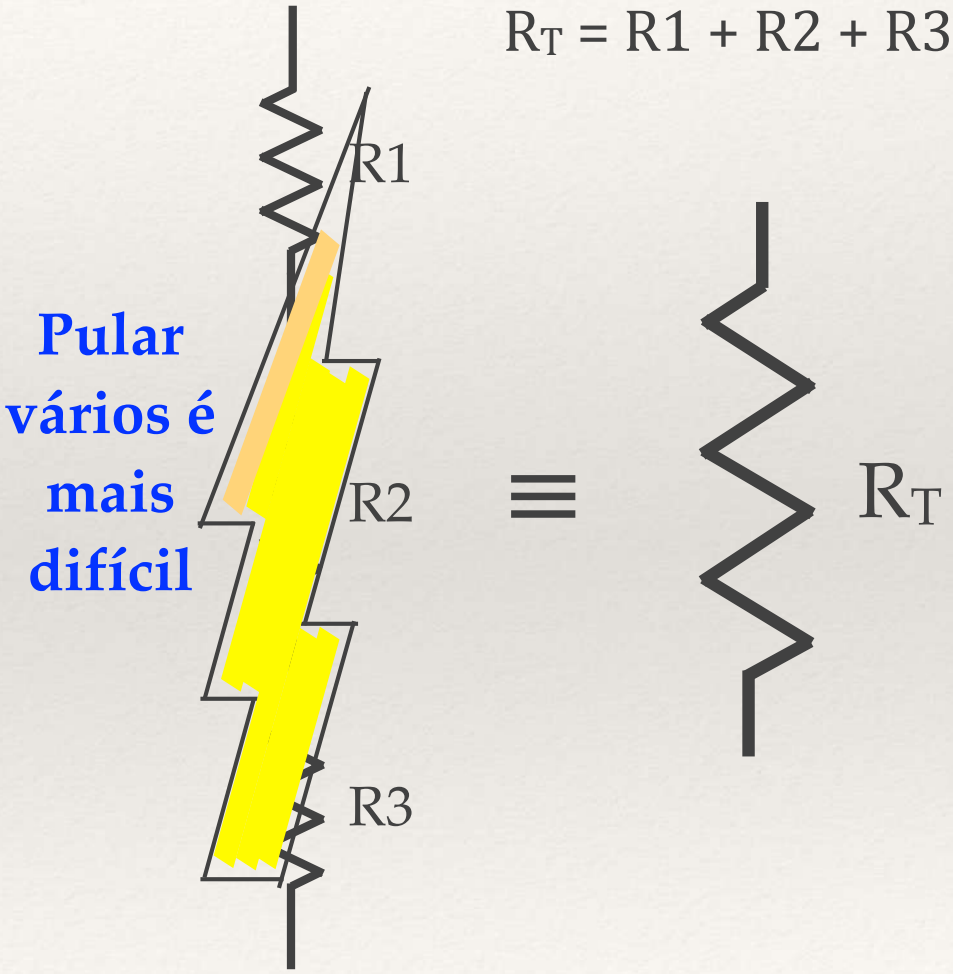
Resistores em Série

A resistência total é igual à soma das resistências individuais;

O arranjo permite:

- Distribuir a potência dissipada;
- Substituir resistências individuais;
- Aumentar a proteção contra descargas elétricas.

$$R_T = R1 + R2 + R3$$



Resistores em Série

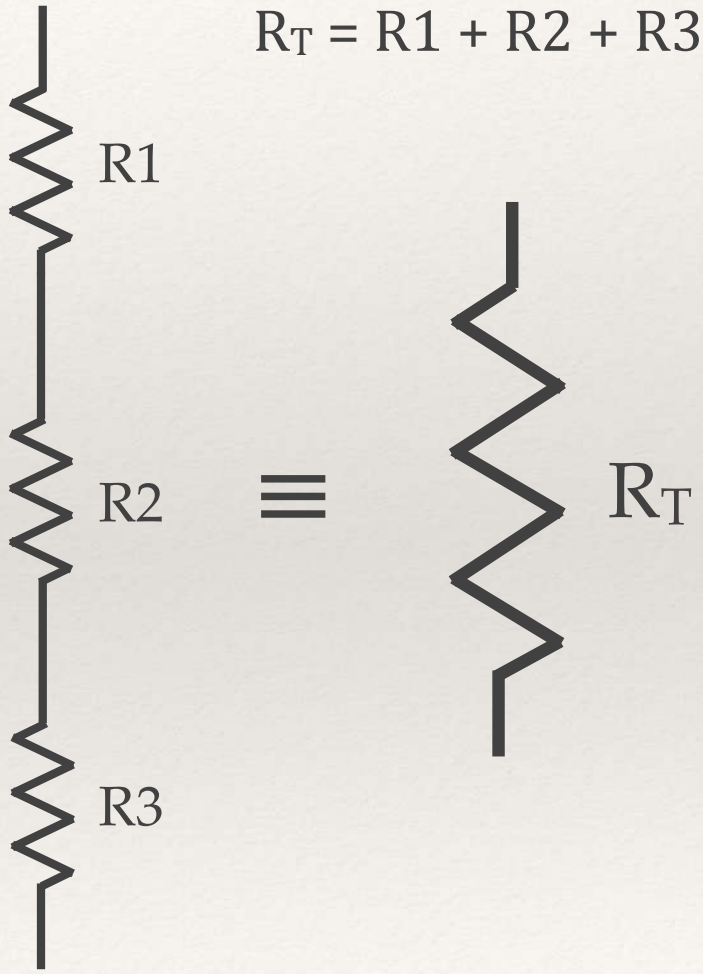
A resistência total é igual à soma das resistências individuais;

O arranjo permite:

- Distribuir a potência dissipada;
- Substituir resistências individuais;
- Aumentar a proteção contra descargas elétricas.

Existem, no entanto, aplicações mais "nobres", como o divisor de tensão.

$R_T = R1 + R2 + R3$



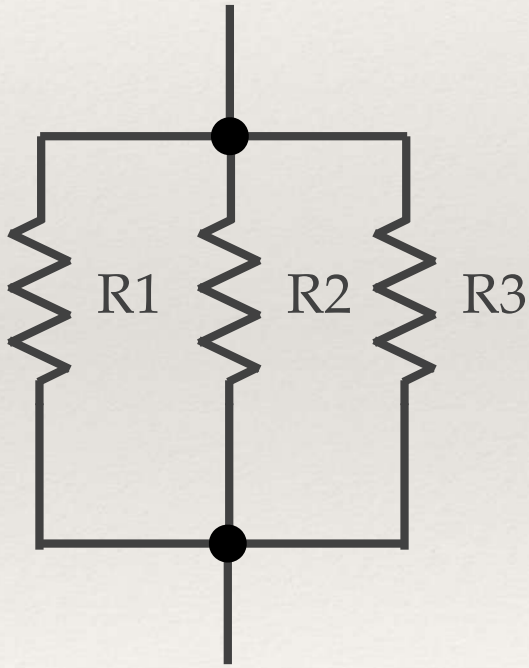
Resistores em Paralelo

A resistência total é menor que a menor das resistências individuais;

$$\frac{1}{Rt} = \frac{1}{R1} + \frac{1}{R2} + \frac{1}{R3}$$

O arranjo permite:

- Distribuir a potência dissipada;
- Substituir resistências individuais.



R - Valores Comerciais

Norma IEC 60063:1963 estabeleceu as “séries E"

* (IEC - *International Electrotechnical Comission*)

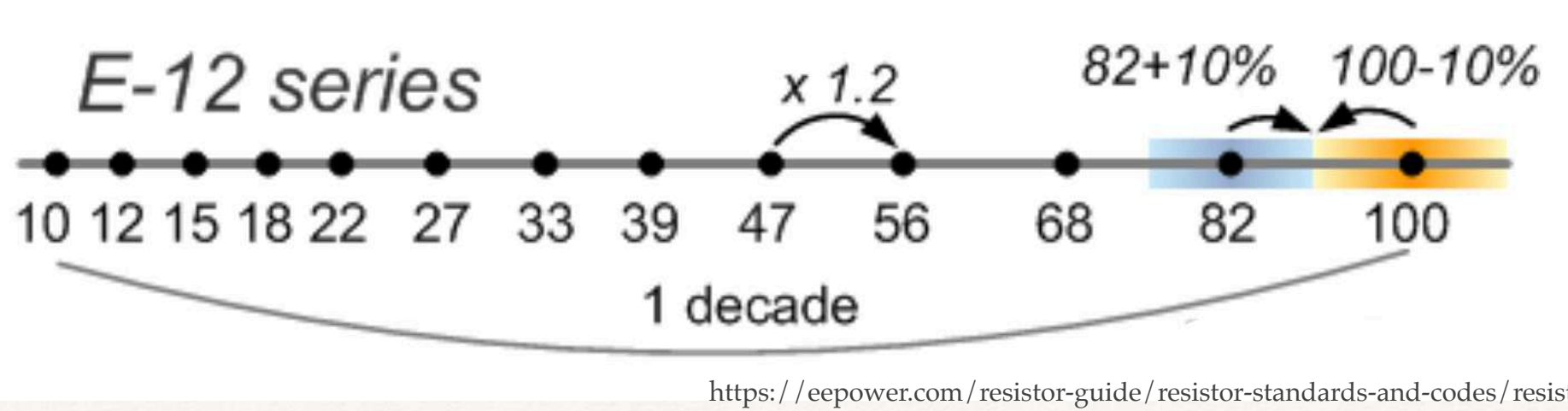
Cada década (0,1-1; 1-10 …) está dividida em “n" espaços logarítmicos, a depender da tolerância:

20% de tolerância, 5 espaços (5 x 20% = 100%) e 6 valores (série E-6);

10% de tolerância envolve 12 valores (série E-12). Veja o cálculo:

$$10^{\frac{1}{12}} = 1{,}21$$

Cada valor é 21% maior que o anterior, com ajustes de aproximação e escala.



https://eepower.com/resistor-guide/resistor-standards-and-codes/resistor-values/#

R - Valores Comerciais

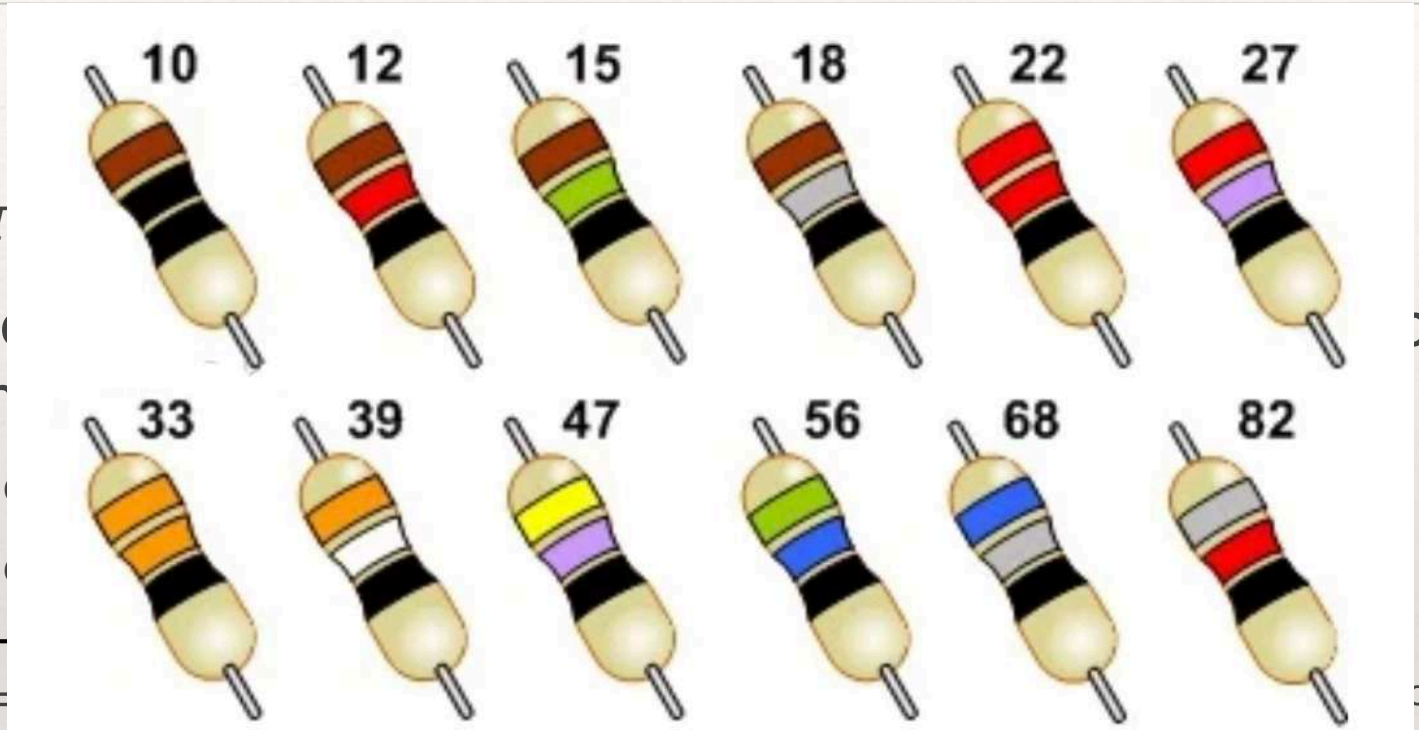
Norma

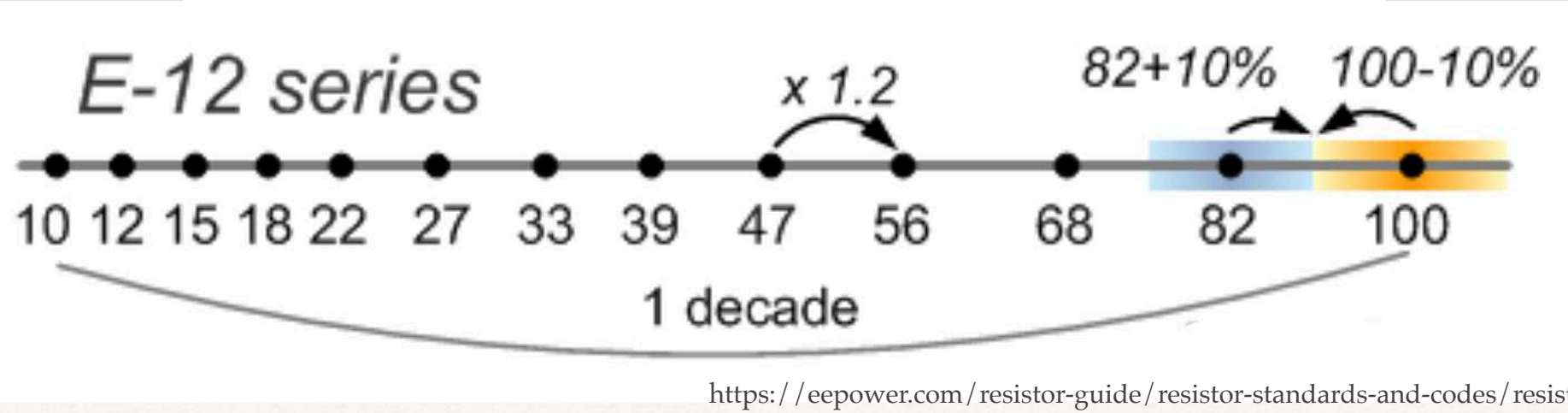
* (IEC - *I*

Cada d… …paços logarítr…

- 20% d… …E-6);
- 10% d…

$10^{\frac{1}{12}}$ = … …o e escala.





https://eepower.com/resistor-guide/resistor-standards-and-codes/resistor-values/#

R - Valores Comerciais

As séries valem para outros componentes:

- Capacitores;
- Indutores;
- Diodos Zener.

A partir da série E-48 os componentes têm valores com 3 dígitos;

Outras séries:

- E-96;
- E-192.

Série E-6		
10	15	22
33	47	68

Série E-12 (10%)					
10	12	15	18	22	27
33	39	47	56	68	82

Série E-24 (5% e 1%)							
10	11	12	13	15	16	18	20
22	24	27	30	33	36	39	43
47	51	56	62	68	75	82	91

Série E-48 (2% e 1%)											
100	105	110	115	121	127	133	140	147	154	162	169
178	187	196	205	215	226	237	249	261	274	287	301
316	332	348	365	383	402	422	442	464	487	511	536
562	590	619	649	681	715	750	787	825	866	909	953

https://eepower.com/resistor-guide/resistor-standards-and-codes/resistor-values/#

R - Acendendo um LED

Resistência de R:

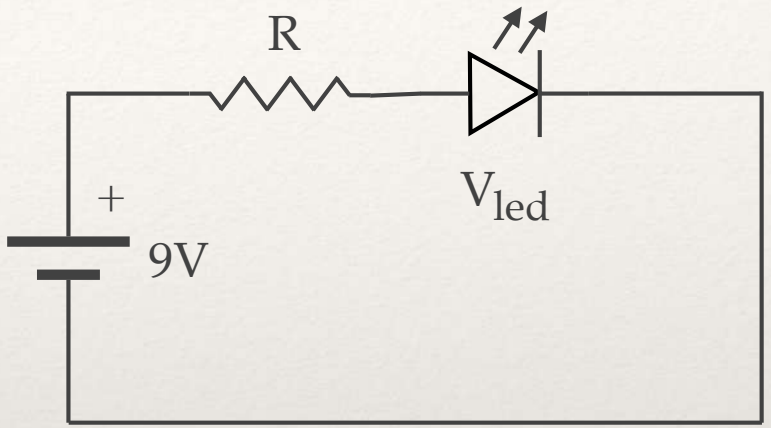
O cálculo é aplicação direta da lei de Ohm:

Tensão: 9 - V_{led} = 9 - 2 = 7 V

Corrente: 10mA

R = 7 / 0,01 = 700 **Ω**

Valor comercial mais próximo é **680Ω**



Potência de R:

i = 7 / 680 = 0,01A (10mA)

P = R x i^2 = 680 x $0,01^2$ = 0,068 W

Valor comercial é **1/8 W** (0,125 W)

R - de 5V para 3,3V

Divisor de tensão

Corrente baixa entre o GPIO de saída no Arduíno e o terra;

5% da corrente máxima (20mA) = 1mA;

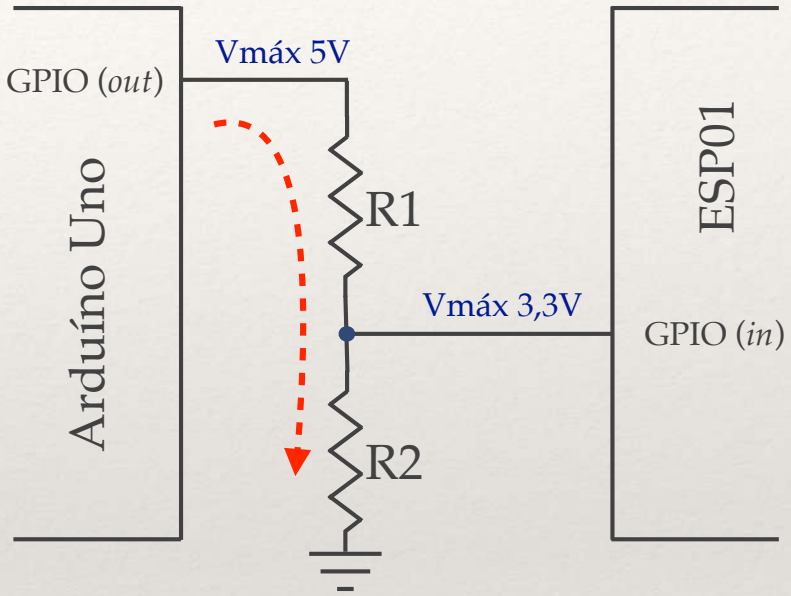
(R1 + R2) = 5V / 1mA = 5 KΩ

R2 / (R1 + R2) = 3,3 / 5

R2 = 0,66 * 5KΩ ≦ **3,3KΩ** (valor comercial)

R1 = 5KΩ - 3,3KΩ ≧ 1,7KΩ

R1 = **1,8KΩ** (valor comercial)



R - de 5V para 3,3V

Divisor de tensão

Corrente baixa entre o GPIO de saída no Arduíno e o terra;

5% da corrente máxima (20mA) = 1mA;

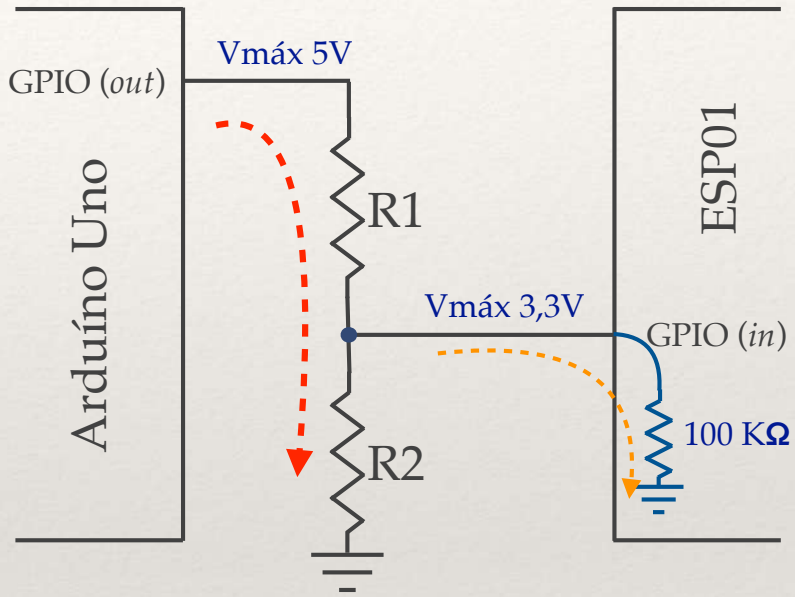
(R1 + R2) = 5V / 1mA = 5 KΩ

R2 / (R1 + R2) = 3,3 / 5

R2 = 0,66 * 5KΩ ≦ **3,3KΩ** (valor comercial)

R1 = 5KΩ - 3,3KΩ ≧ 1,7KΩ

R1 = **1,8KΩ** (valor comercial)



Corrente na carga

Entradas GPIO são de alta impedância (entre 100 KΩ e 1 MΩ);

R2 em paralelo com 100 KΩ = (1 / 3,3KΩ + 1 / 100KΩ)$^{-1}$ = 3,2 KΩ;

Variação de 3% para o menor valor de impedância.

R - *Pull-Up / Pull-Down*

O problema das entradas desconectadas

É importante SEMPRE definir um valor padrão para uma entrada desconectada;

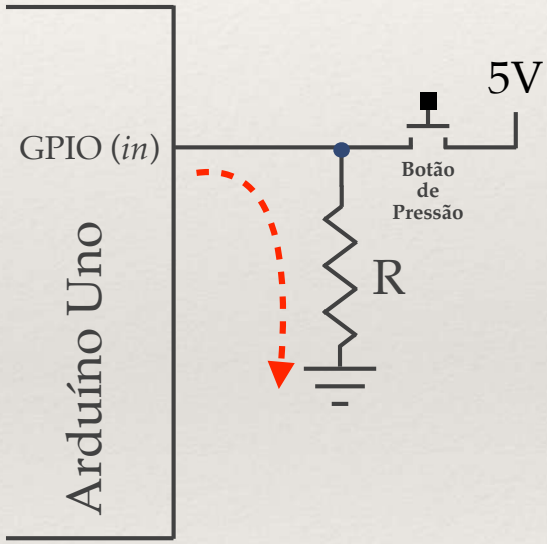
Ruídos elétricos podem determinar o valor de uma entrada desconectada;

5% da corrente máxima (20mA) = 1mA;

R = 5V / 1mA = 4,7 KΩ

Pull-Down: pino funciona como fonte (*source*);

Pull-Up: pino funciona como dreno (*sink*)



R - *Pull-Up / Pull-Down*

O problema das entradas desconectadas

É importante SEMPRE definir um valor padrão para uma entrada desconectada;

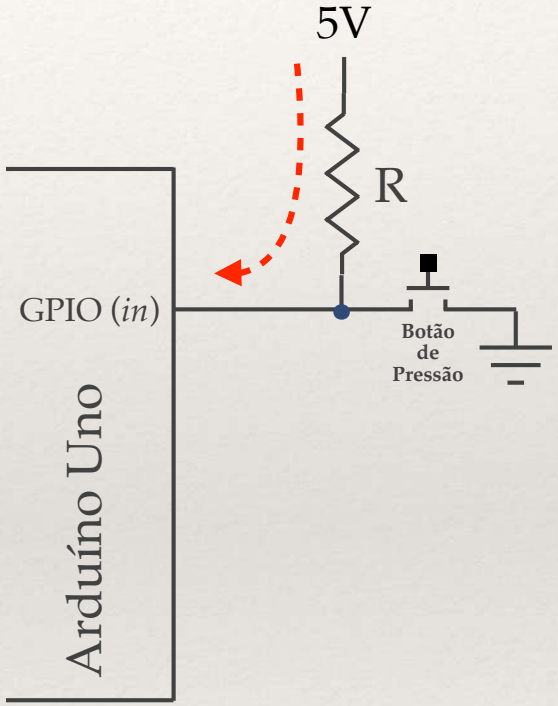
Ruídos elétricos podem determinar o valor de uma entrada desconectada;

5% da corrente máxima (20mA) = 1mA;

R = 5V / 1mA = 4,7 KΩ

Pull-Down: pino funciona como fonte (*source*);

***Pull-Up*: pino funciona como dreno (*sink*)**

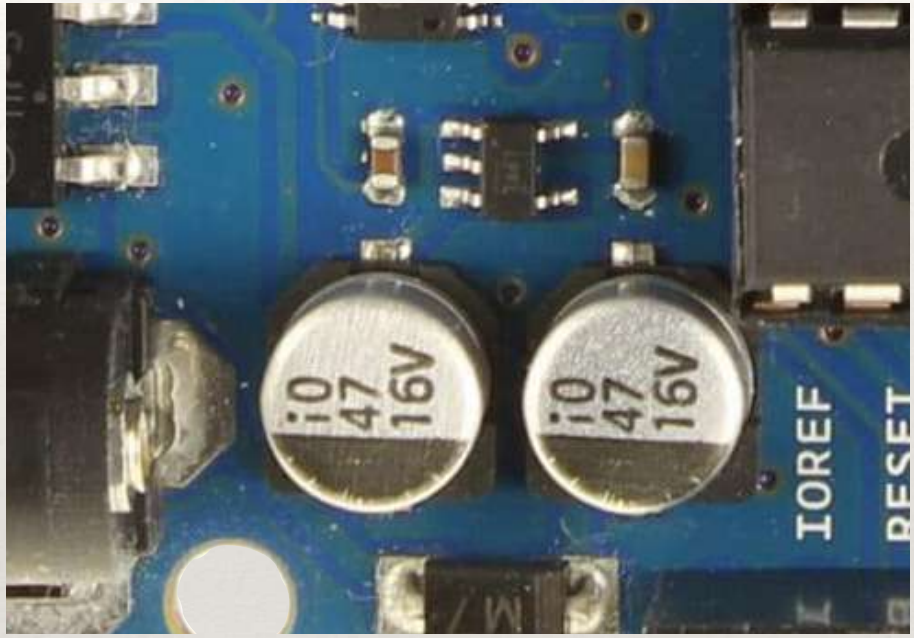


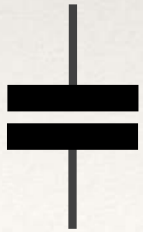
Aula 05

Capacitor

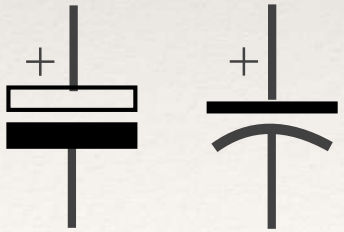
Acumula tensão sob a forma de campo elétrico interno do componente, e é medido em faradays (F), tipicamente sub-múltiplos

- Lembra uma bateria com carga e descarga muito rápida;
- Diversas aplicações na eletrônica analógica, com algumas particularmente importantes em sistemas embarcados.





Capacitor Convencional

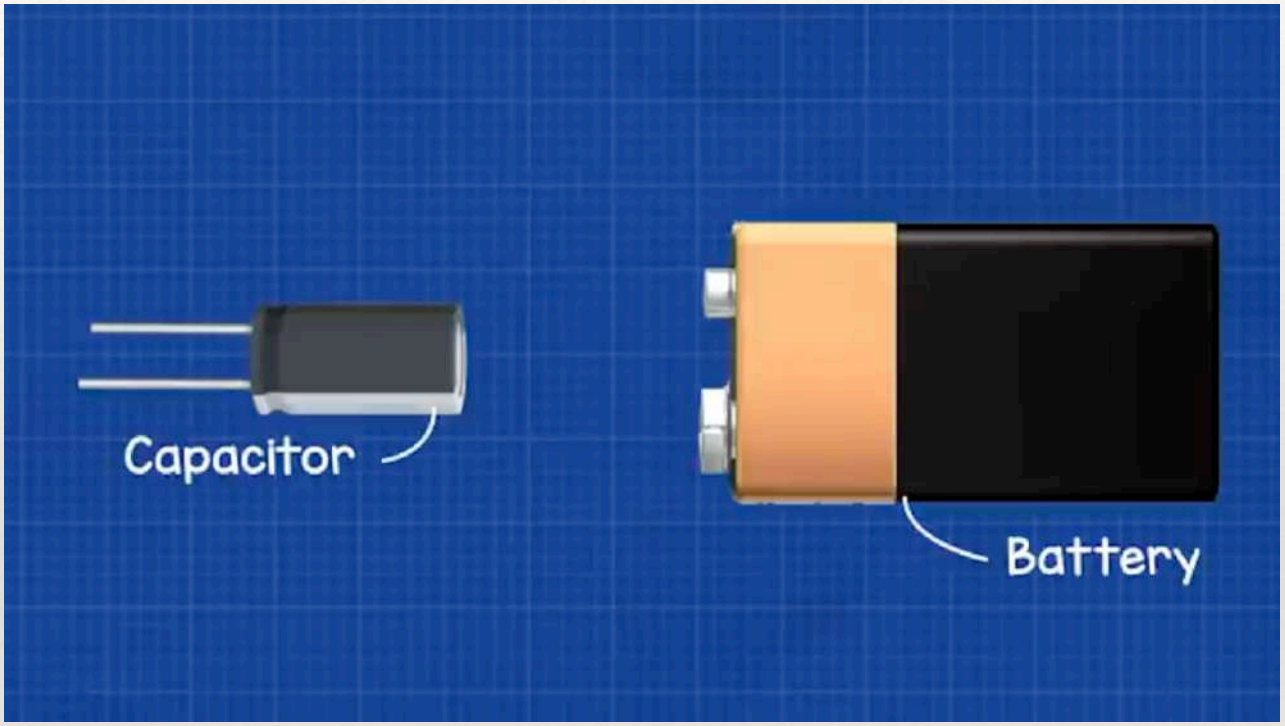


Capacitor Polarizado

Capacitor - Operação

Quando conectado a uma fonte de tensão, acumula carga

- Não há circulação de corrente entre os terminais, e sim apenas o acúmulo de carga;
- A carga fica disponível até que circule corrente de descarga no sentido contrário;



- Normalmente é especificado pela capacitância e tensão de operação. Algumas características específicas do tipo de capacitor também podem influenciar a aplicação;
- A tensão de operação determina o tamanho.

Tipos de Capacitor

1. Eletrolítico

   Alta capacitância, polarizado, possui pequena indutância parasita

2. Poliéster

   Autorenerativo

3. Cerâmico

   Um dos mais comuns

4. Tântalo

   Substituem os eletrolíticos, polarizados, + vida útil, - espaço

5. Mica

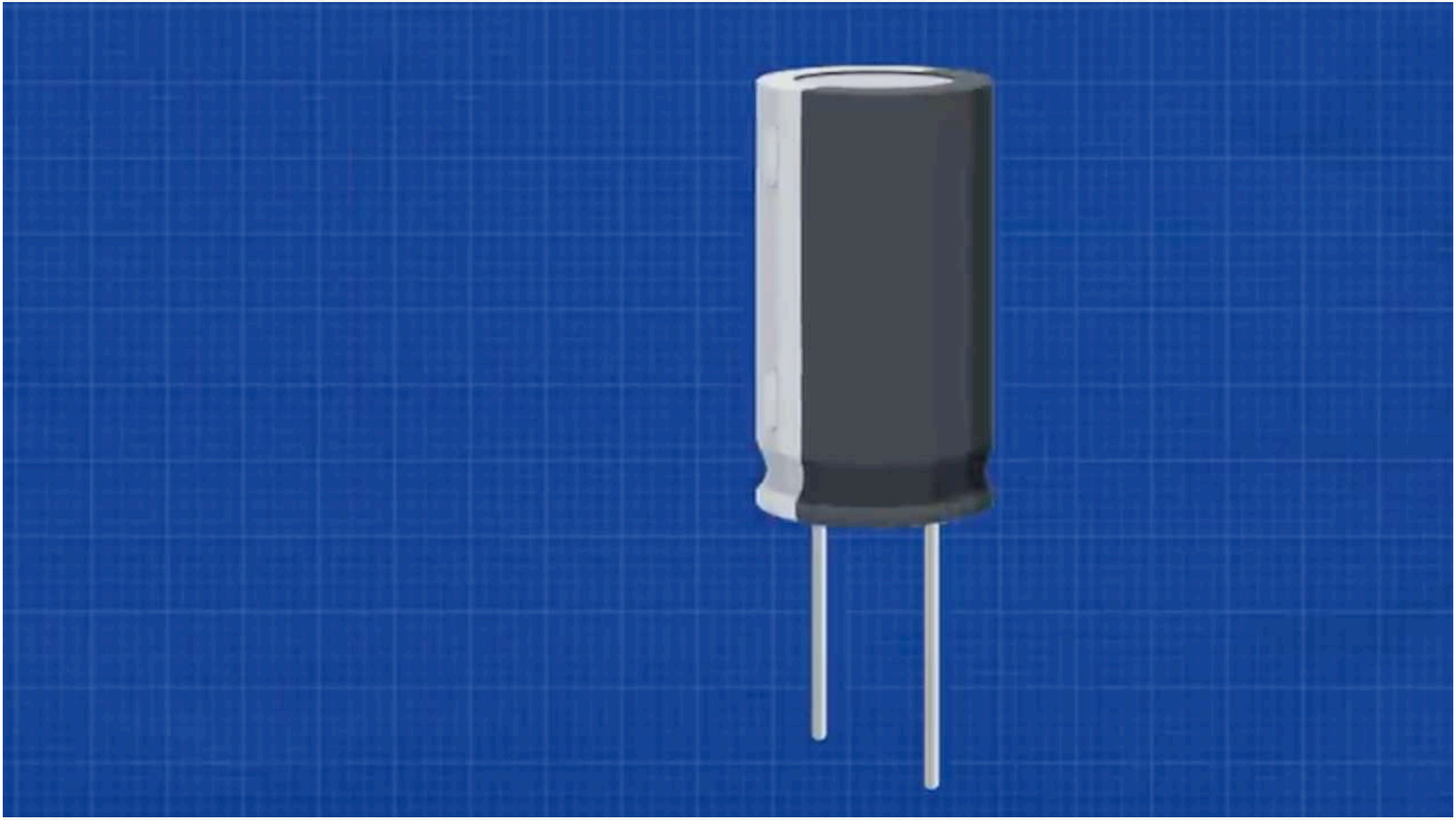
   Muito estável, usado em circuitos que exigem precisão

6. SMD

7. Variáveis

https://www.mundodaeletrica.com.br/tipos-de-capacitores/

Tipos de Capacitor



https://www.youtube.com/watch?v=ucEiEic-kZ4&t=412s

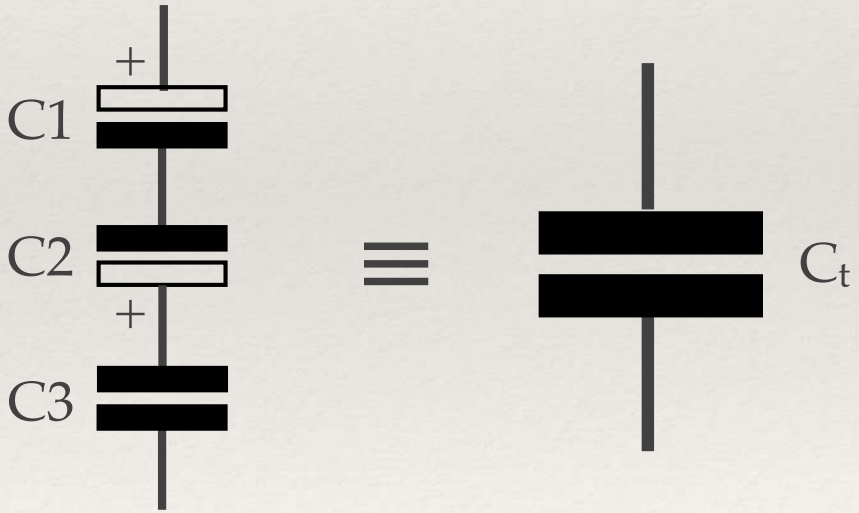
Capacitores em Série

A capacitância total é menor que a menor das capacitâncias individuais;

$$\frac{1}{Ct} = \frac{1}{C1} + \frac{1}{C2} + \frac{1}{C3}$$

O arranjo permite:

- Distribuir a tensão acumulada;
- Substituir capacitâncias individuais;
- Eliminar a polaridade (ver C1 e C2).



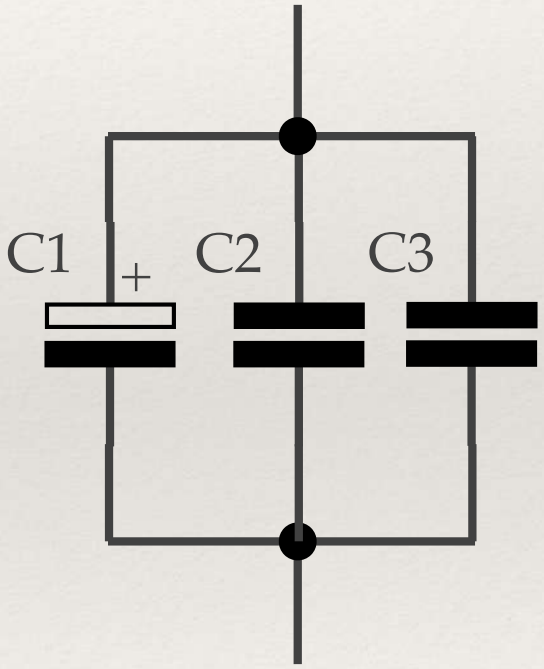
Capacitores em Paralelo

A capacitância total é a soma das capacitâncias individuais;

$$C_T = C1 + C2 + C3$$

O arranjo permite:

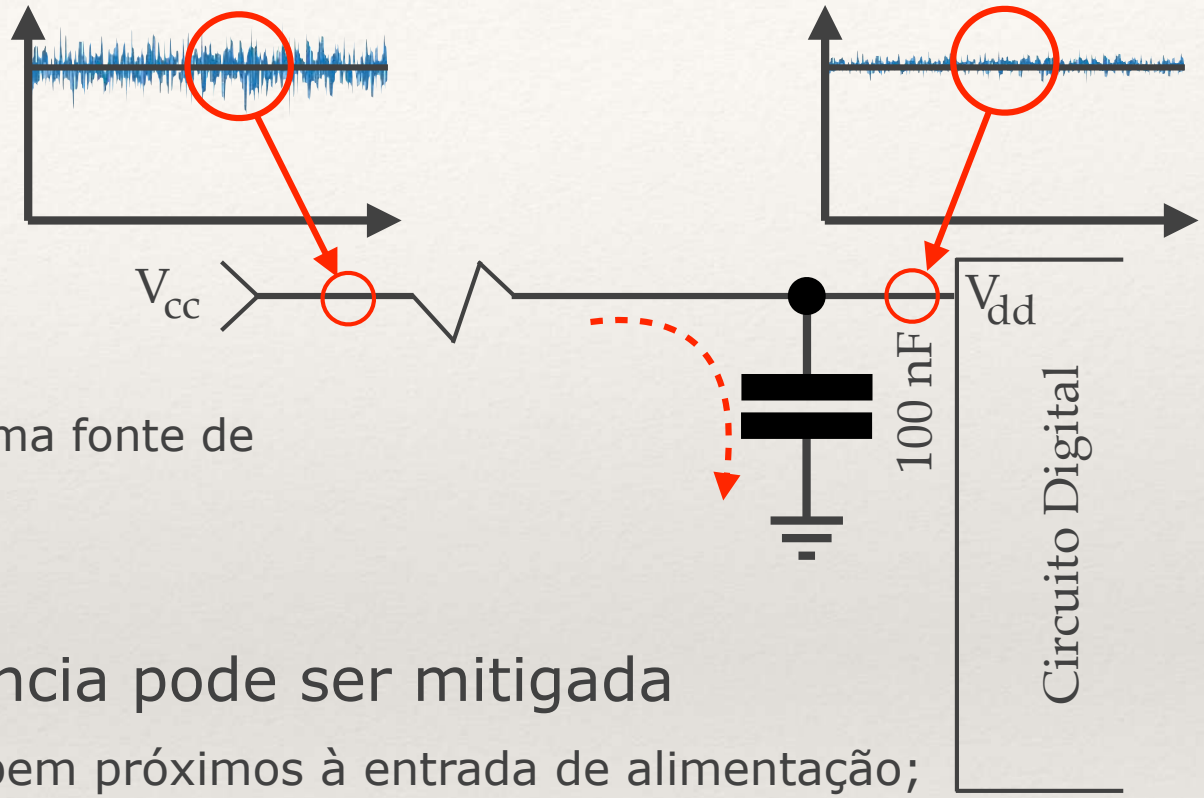
- Substituir capacitares individuais;
- Agregar características de dois tipos diferentes de capacitor (na figura, C1 é eletrolítico e C2 cerâmico).



Capacitor - Desacoplamento

É comum encontramos interferências na linha de alimentação, derivadas:

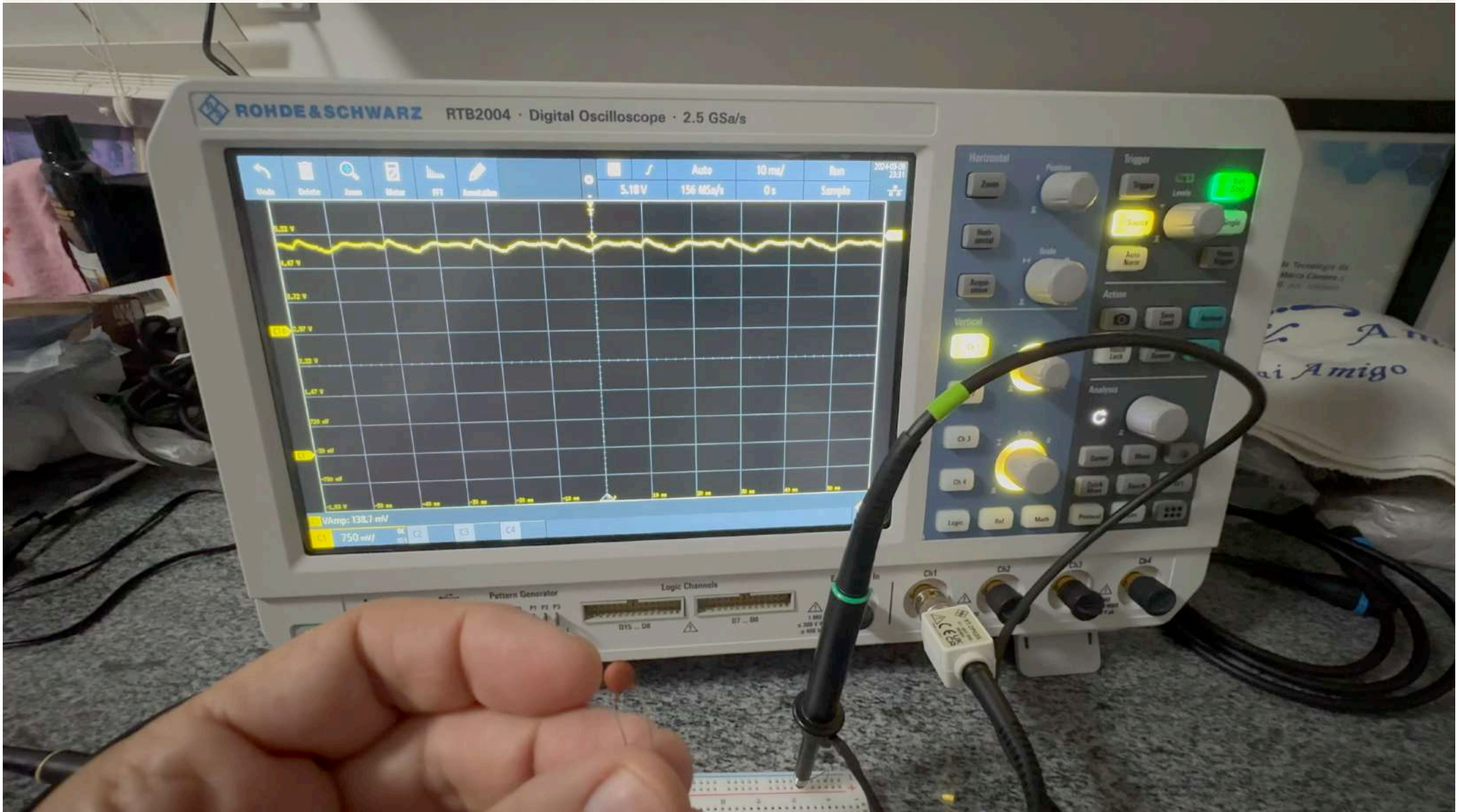
- do próprio circuito;
- de equipamentos ligados na mesma fonte de alimentação;
- de interferência eletromagnética.



Interferência de alta frequência pode ser mitigada

- Capacitores de 100nF instalados bem próximos à entrada de alimentação;
  (cálculo foge do escopo da disciplina)
- Proteção advém da carga e descarga instantânea do capacitor;
- Microcontroladores e circuitos integrados digitais são particularmente sensíveis a este problema.

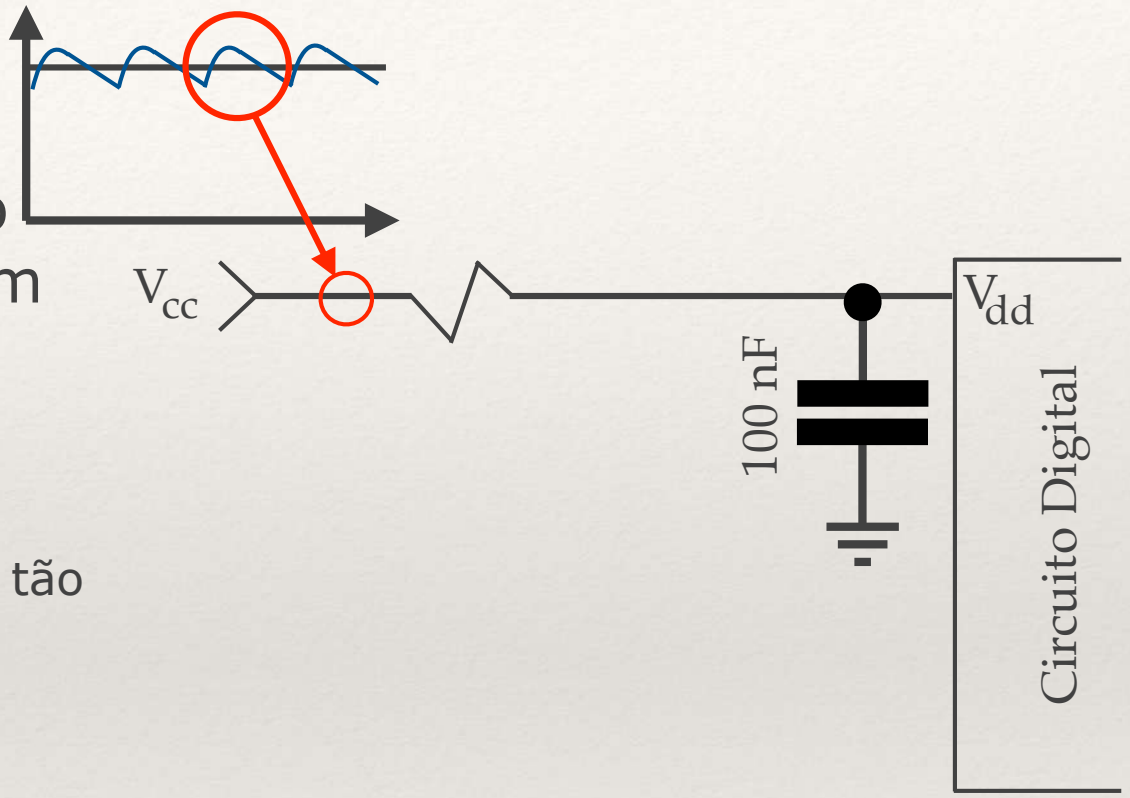
Capacitor - Desacoplamento



Capacitor - Estabiliza tensão

Se houver instabilidade de baixa frequência na tensão de alimentação, usamos um capacitor maior

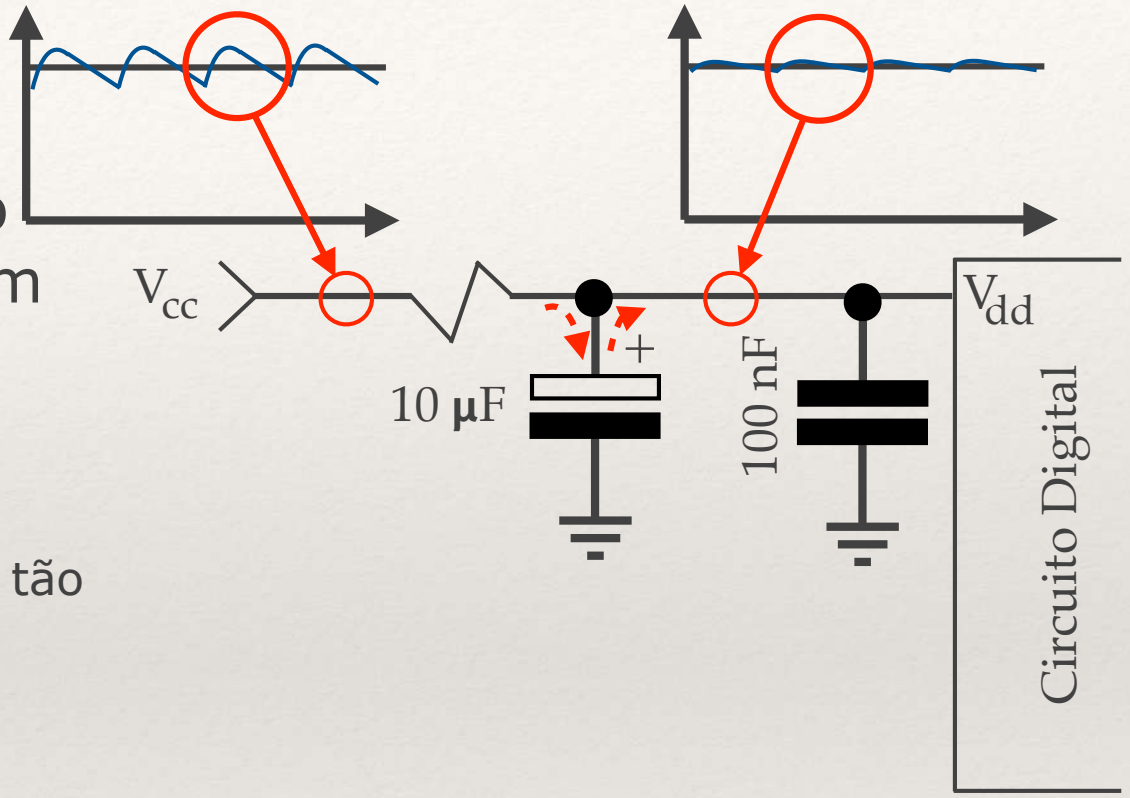
- Isolado, ou em conjunto com anterior;
- Capacitores eletrolíticos não são tão eficientes em altas frequências;



Capacitor - Estabiliza tensão

Se houver instabilidade de baixa frequência na tensão de alimentação, usamos um capacitor maior

- Isolado, ou em conjunto com anterior;
- Capacitores eletrolíticos não são tão eficientes em altas frequências;



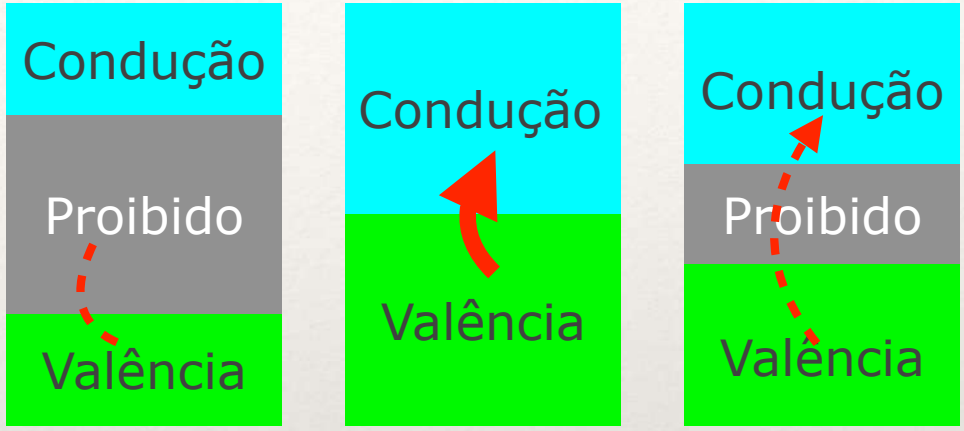
Além destas aplicações ...

- Capacitores são essenciais em diversos circuitos analógicos;
- Compõem filtros dos mais diversos tipos, fontes de alimentação, acopladores e desacopladores, sintonizadores, e dezenas de outras aplicações.

Semicondutores

Em um átomo, os elétrons podem ocupar determinadas órbitas ou níveis de energia

- São sete: k, l, m, n, o, p e q
- O nível mais externo é chamado de nível de valência;
- Para conduzir uma corrente elétrica, os elétrons precisam saltar do nível de valência para o nível de condução.



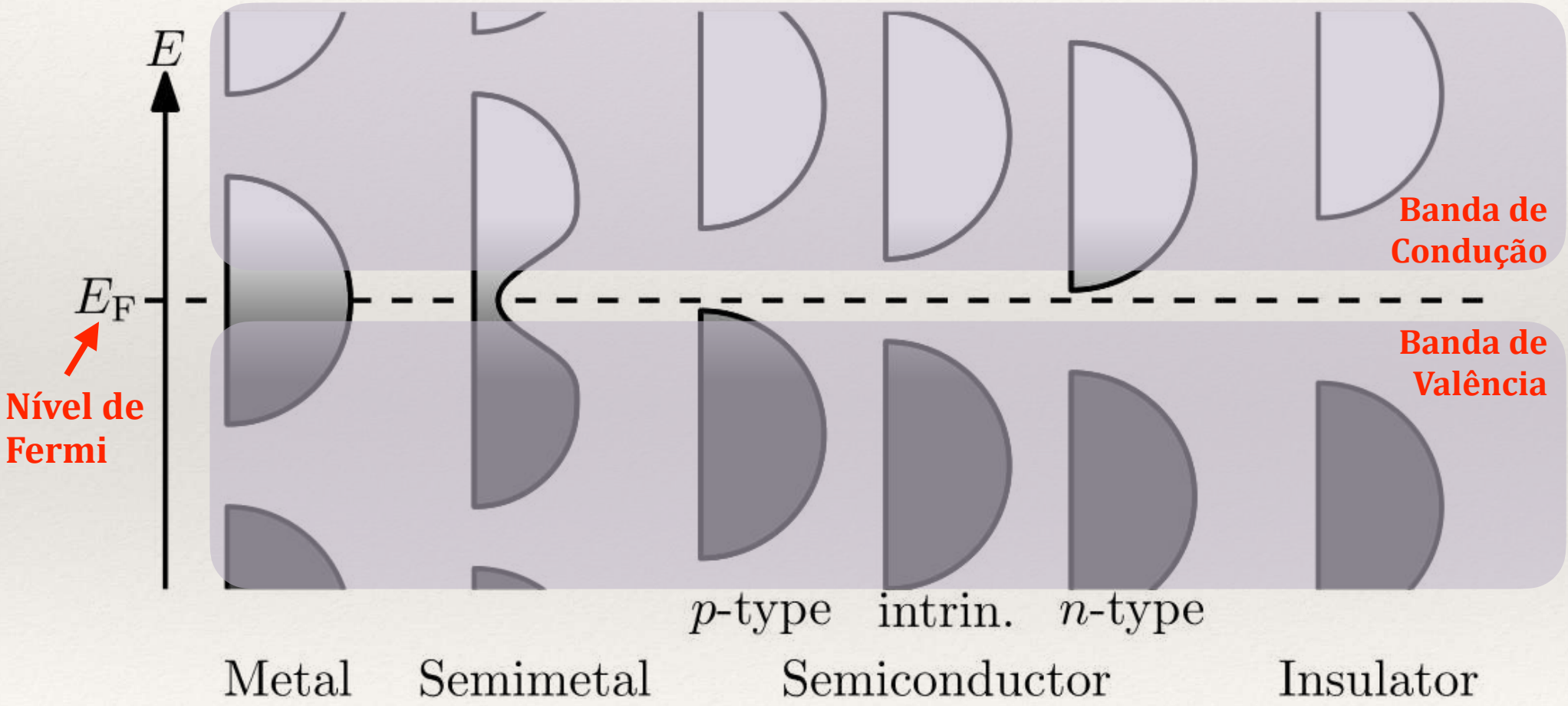
Isolante **Condutor** **Semicondutor**

A existência de níveis ou bandas de energia “proibidas" entre a condução e a valência determinam o tipo de material

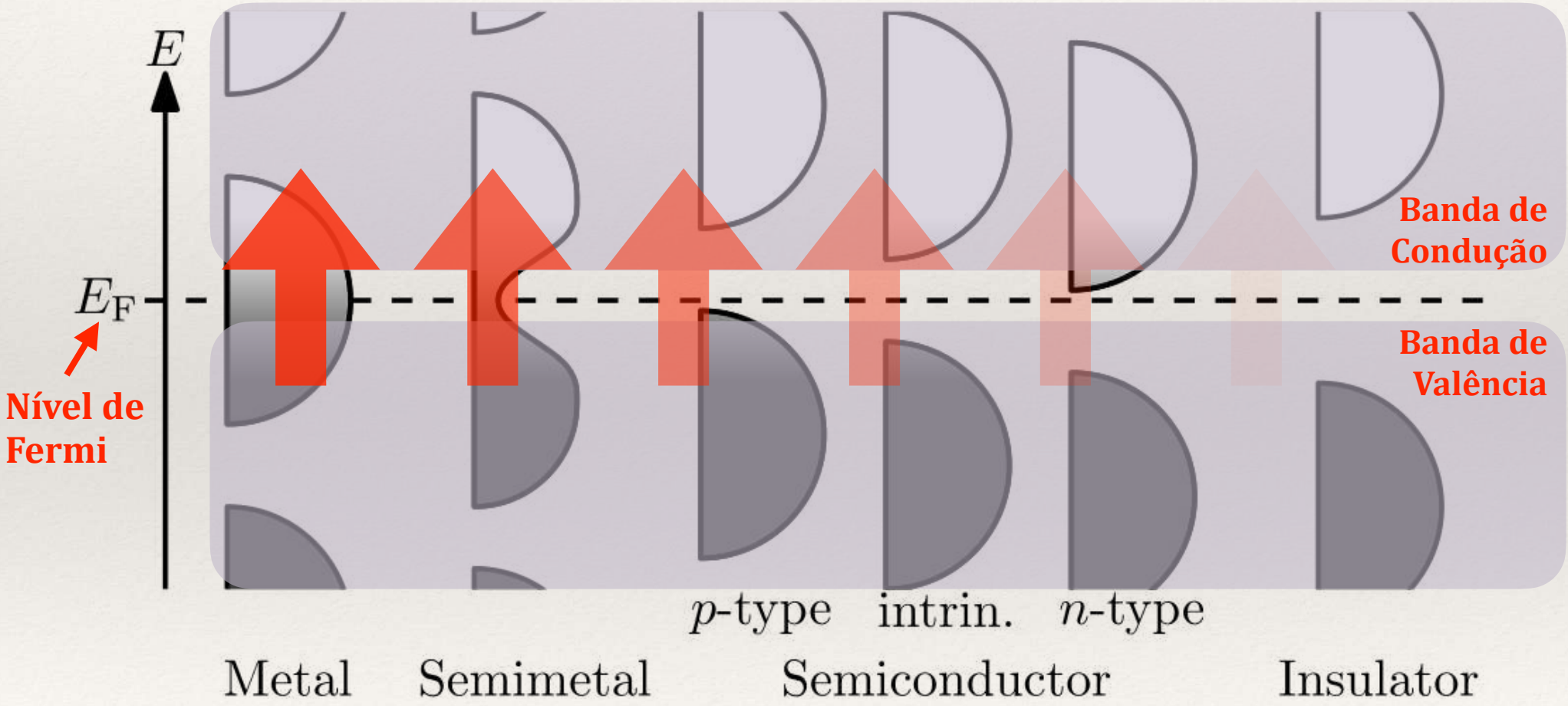
- Não existe banda proibida: **CONDUTOR**
- Existe uma banda proibida de grande dimensão: **ISOLANTE**
- Existe uma banda proibida de pequena dimensão: **SEMICONDUTOR**

https://www.feis.unesp.br/Home/departamentos/engenhariaeletrica/capitulo_1_diodos#:~:text=Germânio é outro elemento tetravalente,e de outros componentes semicondutores.&text=Falta de elétrons livres no,constituir-se numa vantagem enorme.

Bandas de Valência e Condução



Bandas de Valência e Condução



Semicondutores

Para nosso estudo, interessa conhecer os semicondutores

- Banda de valência próxima da banda de condução;
- Temperatura, choques mecânicos, luz, radiação e, obviamente, tensão e corrente elétrica podem fazer os elétrons migrarem de banda. Com “elétrons livres”, o material funciona como um condutor;
- O elemento semicondutor mais comum é o Silício, que possui 4 elétrons na camada de valência. É o segundo elemento mais comum na Terra, mas outros materiais podem ser utilizados, como o Germânio;

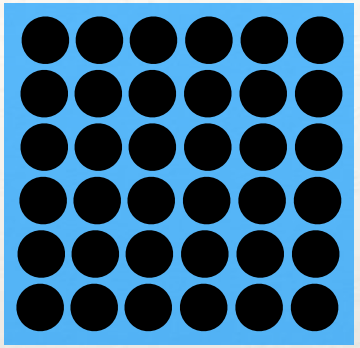
Semicondutores tipo “N" e tipo “P"

- São produzidos pela dopagem do Silício com “impurezas";
- Tipo “N”: dopagem com átomos pentavalentes como o Fósforo (sobra um elétron por átomo), tornando o material propenso a liberar elétrons;
- Tipo “P”: dopagem com átomos trivalentes como o Alumínio (falta um elétron por átomo), tornando o material propenso a receber elétrons.

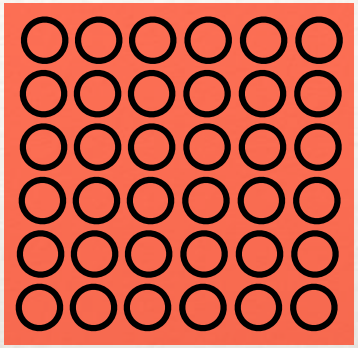
Junção Semicondutora

Obtida pela junção de dois materiais dopados de forma diferente

- Na junção, ocorre migração dos elétrons em excesso do material N para o material P, formando a camada de depleção;
- Diferença de potencial (≈ 0,7V no silício) impede a continuidade de circulação de corrente.
- * No germânio, a tensão é de 0,3V.



Tipo “N"

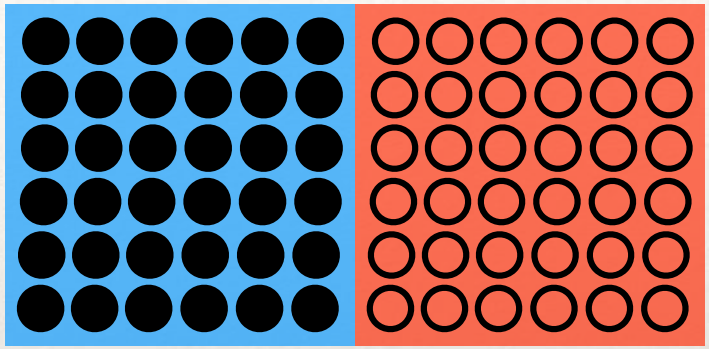


Tipo “P”

Junção Semicondutora

Obtida pela junção de dois materiais dopados de forma diferente

- Na junção, ocorre migração dos elétrons em excesso do material N para o material P, formando a camada de depleção;
- Diferença de potencial (≈ 0,7V no silício) impede a continuidade de circulação de corrente.
- * No germânio, a tensão é de 0,3V.

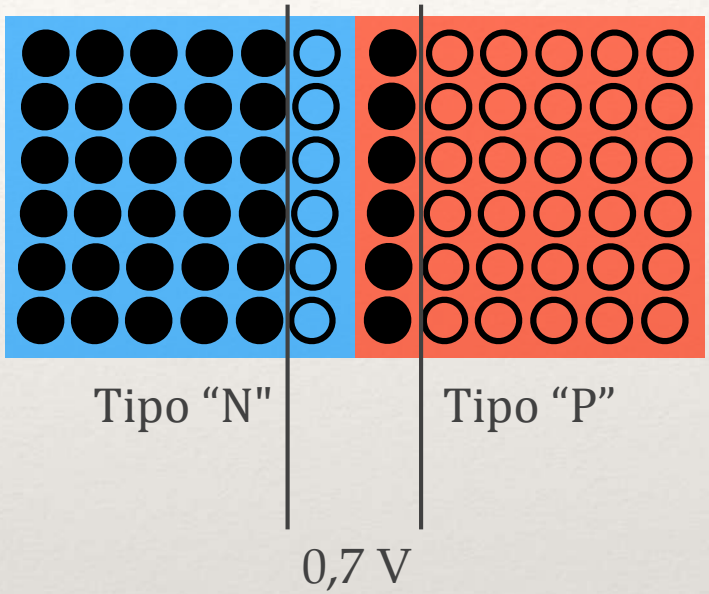


Tipo “N" Tipo “P”

Junção Semicondutora

Obtida pela junção de dois materiais dopados de forma diferente

- Na junção, ocorre migração dos elétrons em excesso do material N para o material P, formando a camada de depleção;
- Diferença de potencial (≈ 0,7V no silício) impede a continuidade de circulação de corrente.
- * No germânio, a tensão é de 0,3V.

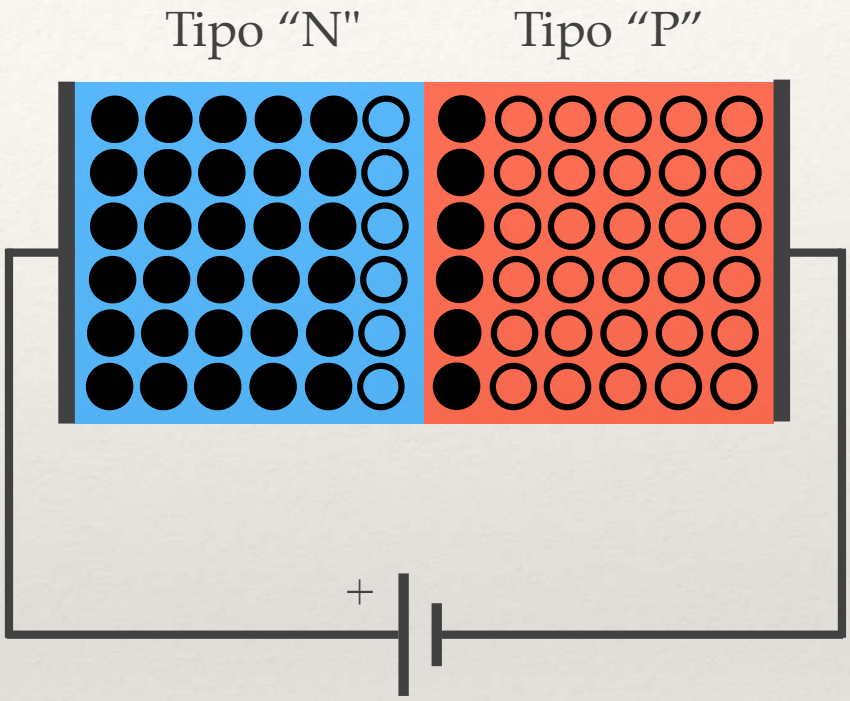


Junção Semicondutora

E quando a junção é polarizada?

Com o material N ligado ao positivo, aumenta a camada de depleção, e não circula corrente;

Invertendo, desde que a tensão seja maior que a diferença de potencial da junção, haverá circulação de corrente.

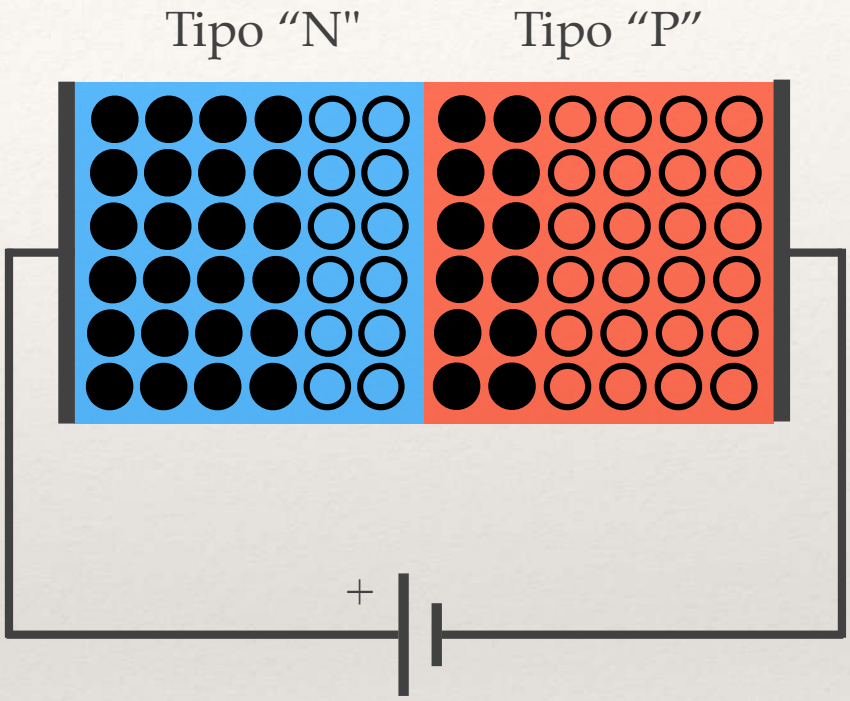


Junção Semicondutora

E quando a junção é polarizada?

Com o material N ligado ao positivo, aumenta a camada de depleção, e não circula corrente;

Invertendo, desde que a tensão seja maior que a diferença de potencial da junção, haverá circulação de corrente.

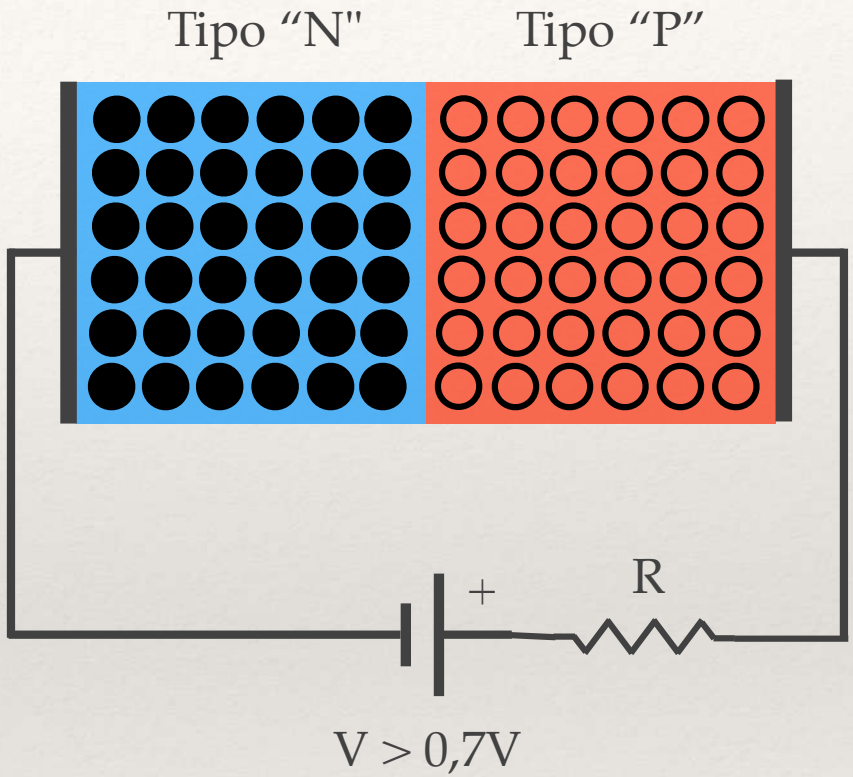


Junção Semicondutora

E quando a junção é polarizada?

Com o material N ligado ao positivo, aumenta a camada de depleção, e não circula corrente;

Invertendo, desde que a tensão seja maior que a diferença de potencial da junção, haverá circulação de corrente.

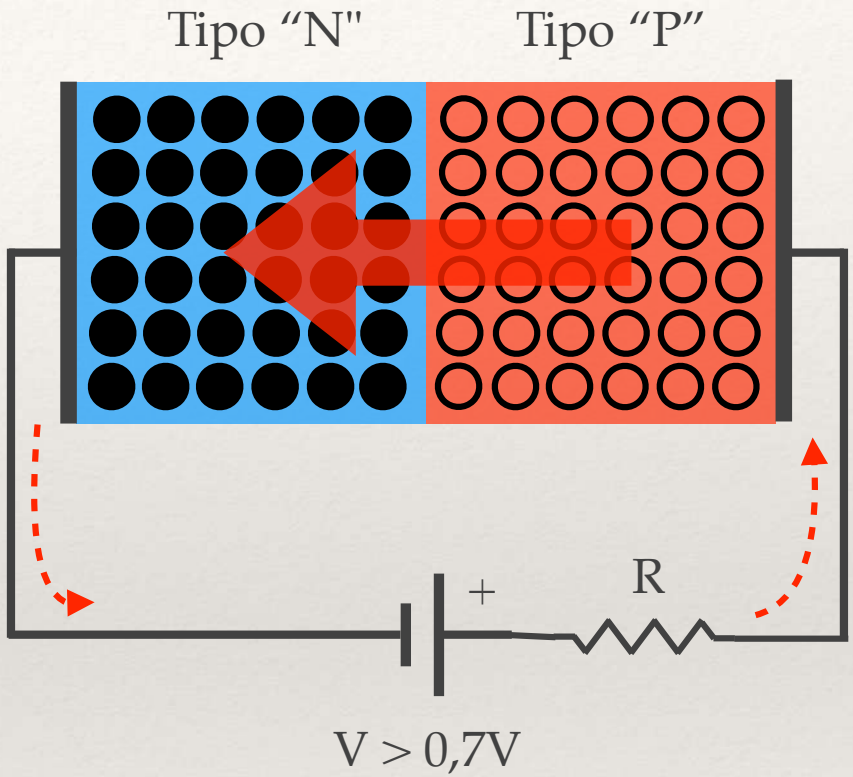


Junção Semicondutora

E quando a junção é polarizada?

Com o material N ligado ao positivo, aumenta a camada de depleção, e não circula corrente;

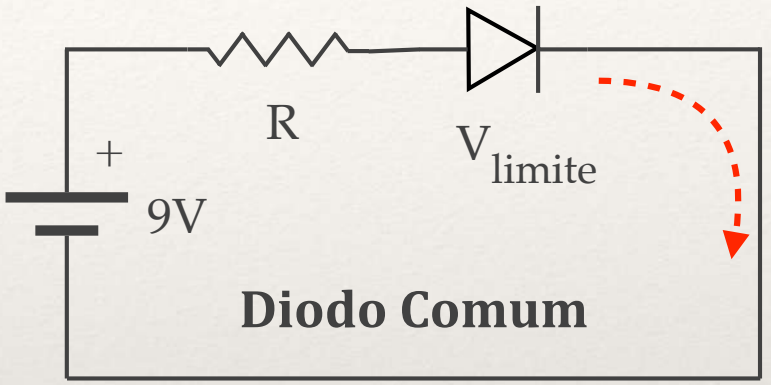
Invertendo, desde que a tensão seja maior que a diferença de potencial da junção, haverá circulação de corrente.

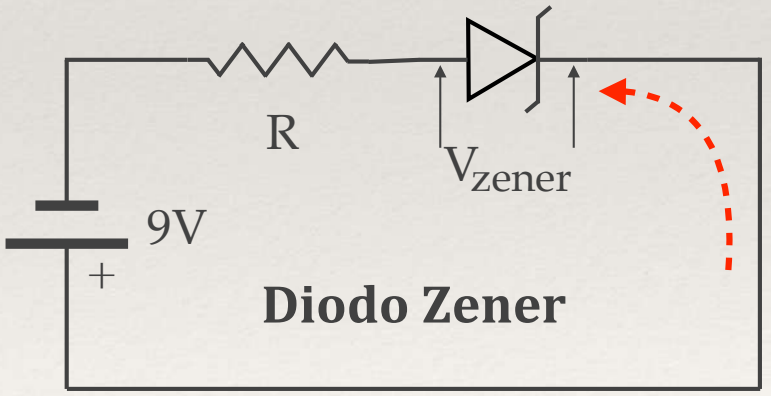


Diodo: usando a junção P-N

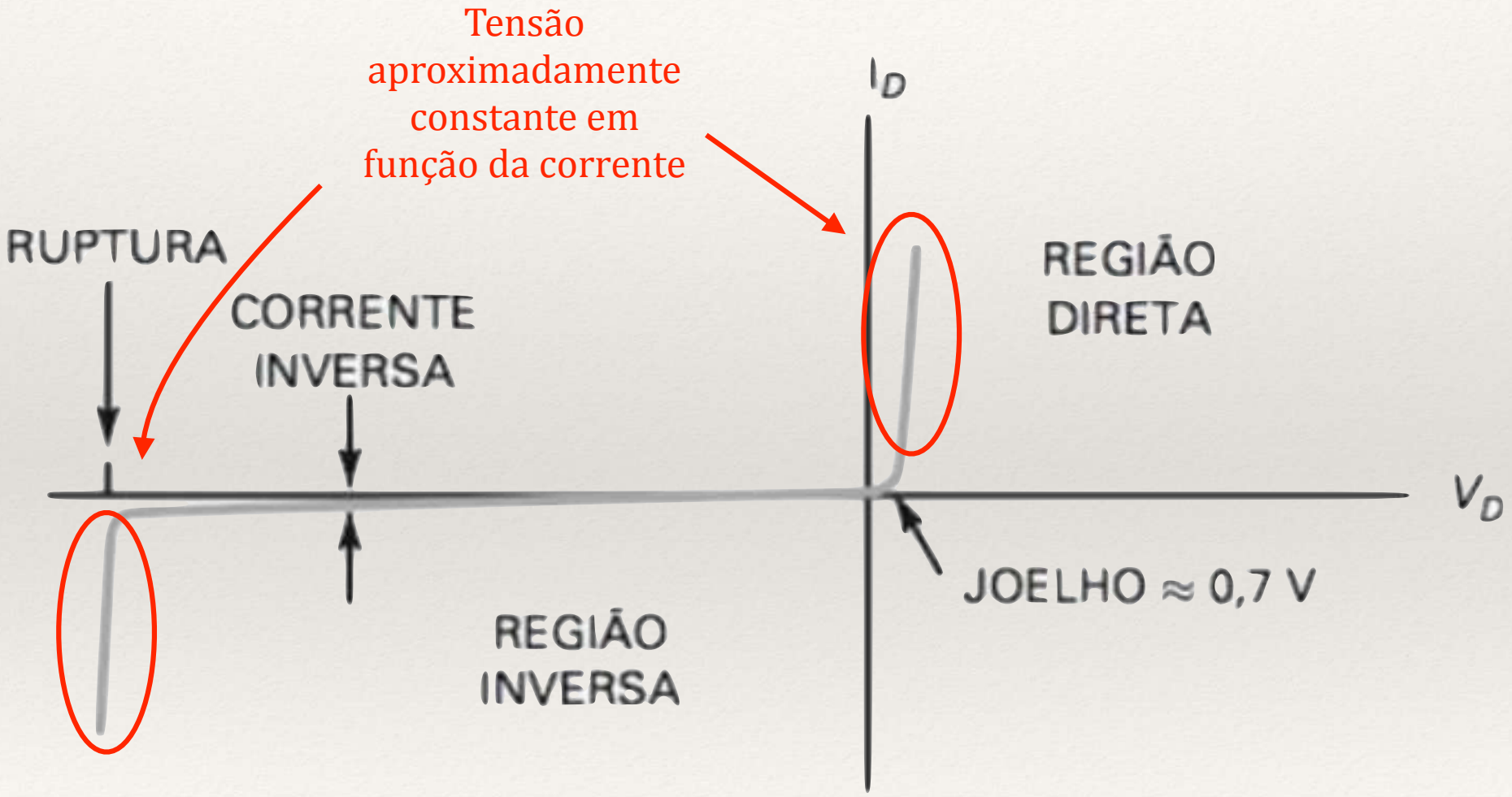
Só permite a circulação de corrente em um sentido

- Exige tensão mínima limite (≈ 0,7V);
- Corrente só circula no sentido contrário se a tensão aplicada ultrapassar a tensão de ruptura;
- Um tipo específico de diodo aproveita este efeito (chamado de “efeito zener”) para outras aplicações.





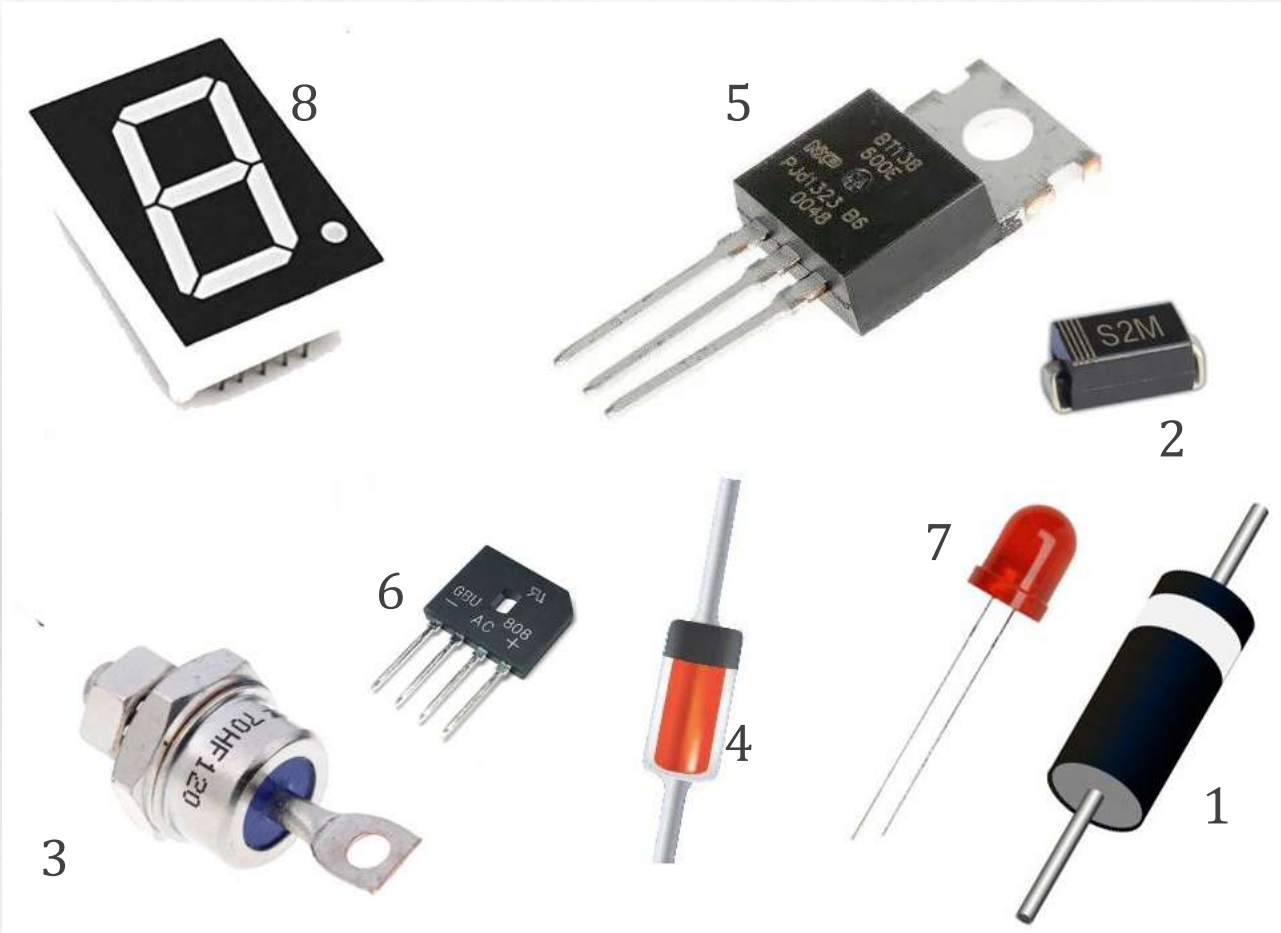
Diodo: curva característica



https://www.researchgate.net/figure/Figura-2-Curva-caracteristica-do-diodo-Fonte-2-Para-boa-parte-dos-diodos-trabalhar_fig2_309644229

Tipos de Diodo

1. Retificador
   Fontes de alimentação, suporta altas correntes;
2. SMD
3. Alta potência
   Para correntes elevadas;
4. Sinal
   Utilizados em circuitos de alta frequência. Frequentemente são de germânio;
5. Duplo
   2 diodos em contra-posição;
6. Ponte retificadora
   Arranjo de 4 diodos para fontes de alimentação
7. LED
8. Display de 7 segmentos



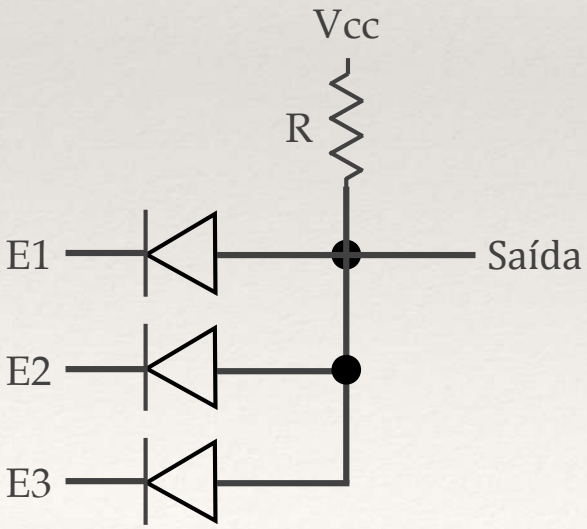
Diodo: portas lógicas simples

Porta AND (E)

Qualquer entrada (E1 a E3) no nível zero provoca o nível zero na saída;

Apenas se todas as entradas estiverem em nível um garantimos a saída no nível 1;

O resistor impede um curto-circuito.

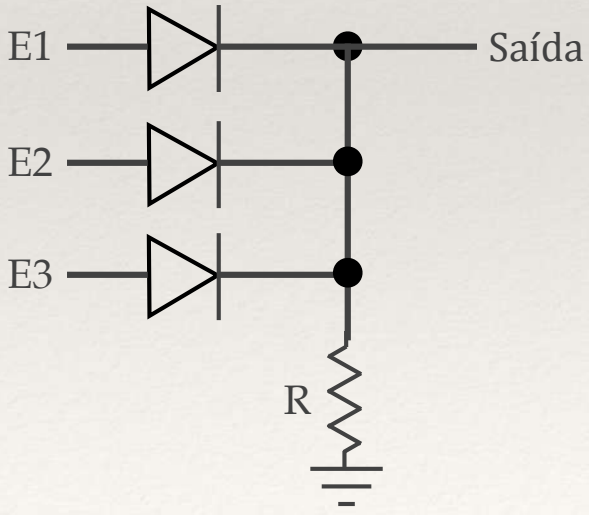


Porta OR (OU)

Qualquer entrada (E1 a E3) no nível um provoca o nível um na saída;

Apenas se todas as entradas estiverem no nível zero garantimos a saída no nível zero;

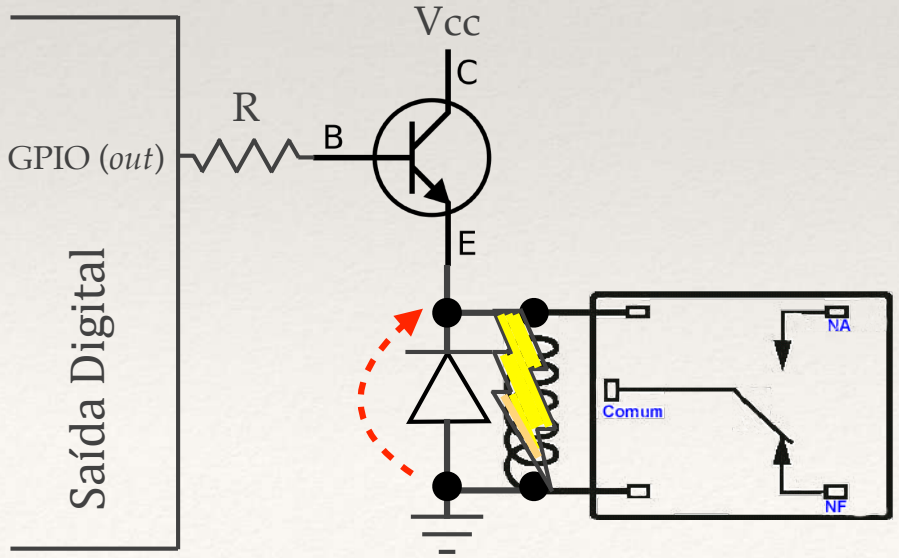
O resistor impede um curto-circuito.



Diodo: proteção de circuitos

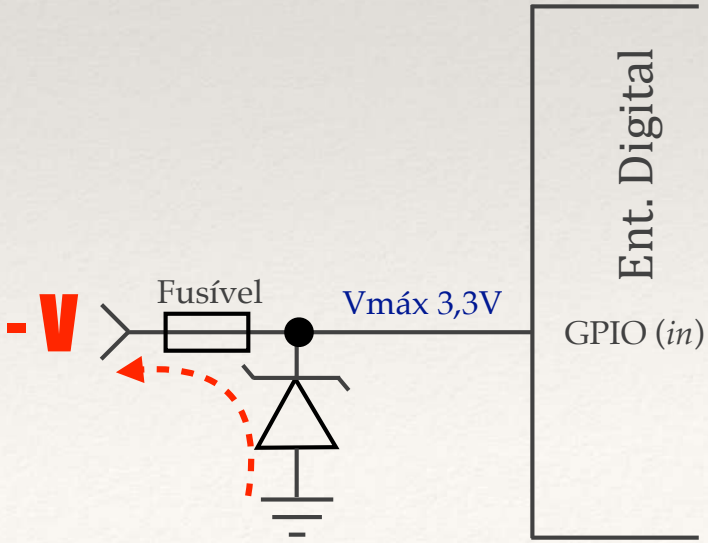
Em circuitos digitais, são utilizados para proteção

- Diodos comuns impedem polarização invertida;
- Proteção funciona inclusive para pulsos reversos de alta tensão (relês).



Diodos zener protegem entradas digitais sensíveis

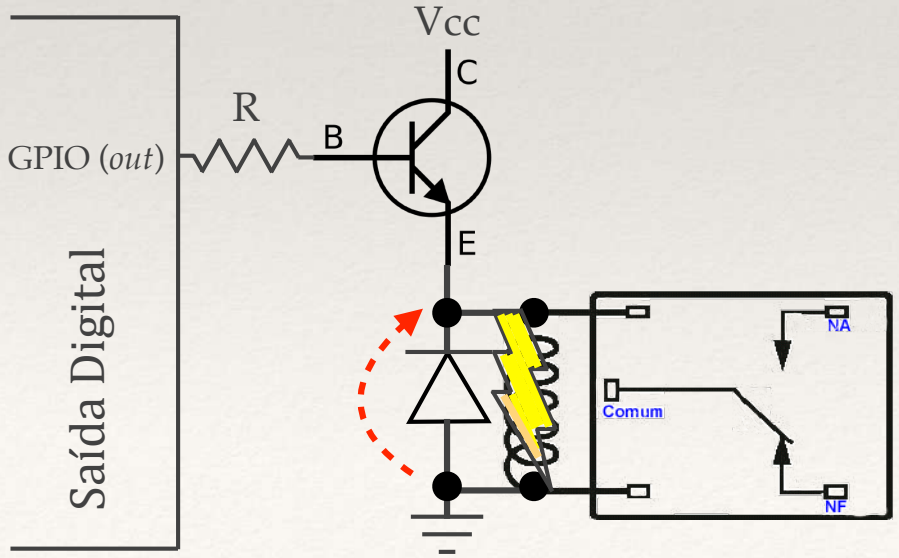
- Proteção contra inversão de polaridade;
- Proteção contra sobretensões;
- Fusível em série contra sobrecorrentes.



Diodo: proteção de circuitos

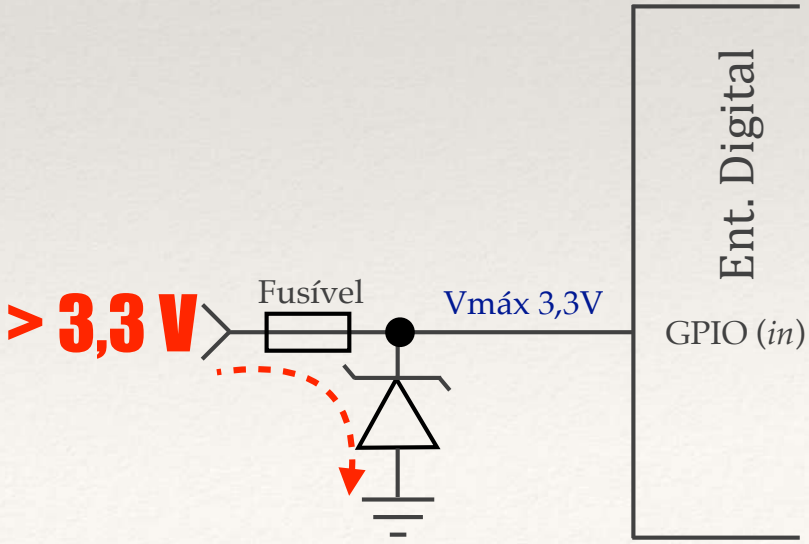
Em circuitos digitais, são utilizados para proteção

- Diodos comuns impedem polarização invertida;
- Proteção funciona inclusive para pulsos reversos de alta tensão (relês).



Diodos zener protegem entradas digitais sensíveis

- Proteção contra inversão de polaridade;
- Proteção contra sobretensões;
- Fusível em série contra sobrecorrentes.

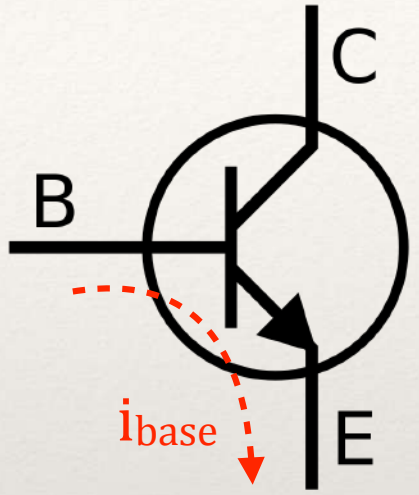


Aula 05

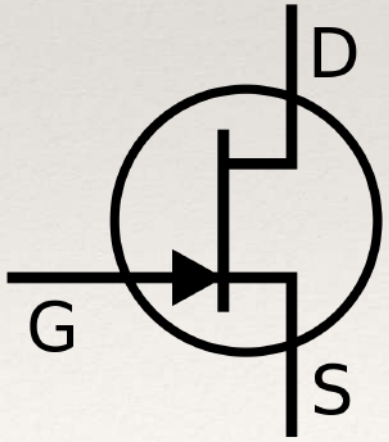
O Transístor

São dois os principais tipos:

- Bipolar de Junção: controlado pela corrente que circula pela base;
- Efeito de Campo: controlado pela tensão presente no *gate*.



Bipolar NPN

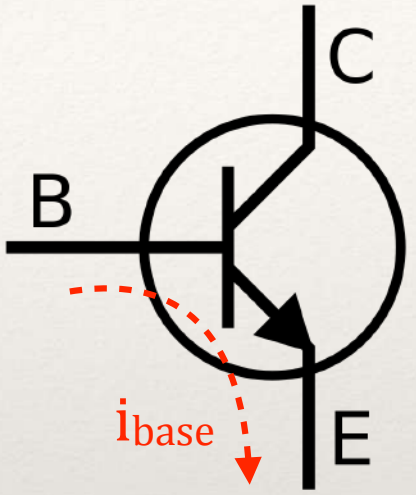


J-FET Canal N

O Transístor

São dois os principais tipos:

- Bipolar de Junção: controlado pela corrente que circula pela base;
- Efeito de Campo: controlado pela tensão presente no *gate*.



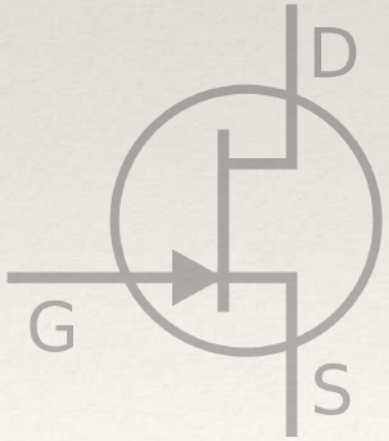
Bipolar NPN

Transistores Bipolares

- Podem ser do tipo NPN ou PNP;

Aplicações Típicas

- Chaveadores;
- Amplificadores.

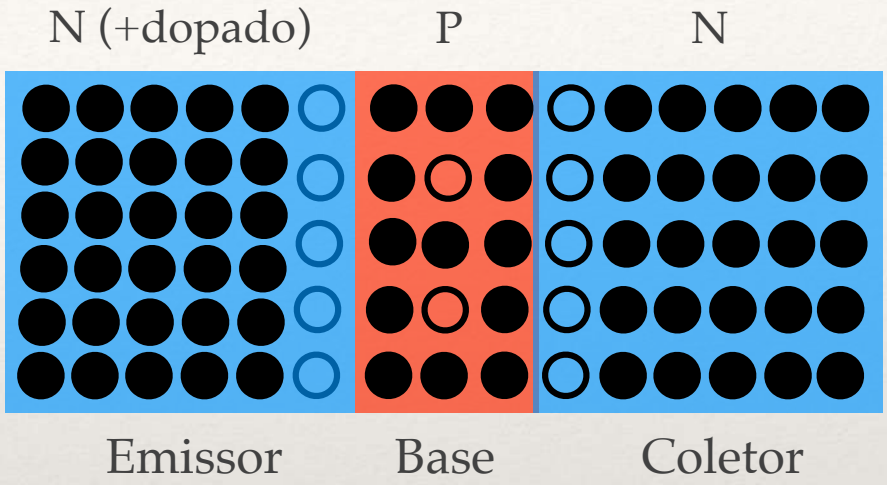


J-FET Canal N

Transístor: duas junções

Similar ao diodo, mas...

- Temos duas camadas de depleção;
- O emissor tem uma dopagem maior. Com isso, a camada de depleção entre base e emissor libera elétrons em excesso;



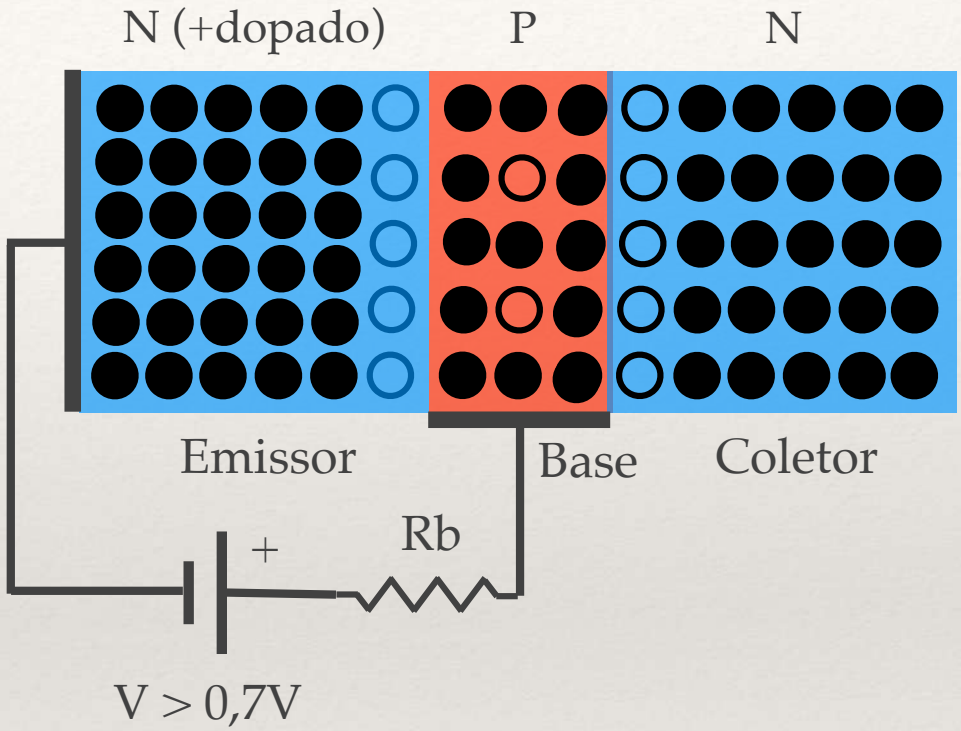
Transístor: duas junções

Similar ao diodo, mas...

- Temos duas camadas de depleção;
- O emissor tem uma dopagem maior. Com isso, a camada de depleção entre base e emissor libera elétrons em excesso;

Se fizermos circular uma corrente entre a base e o emissor:

- A camada de depleção correspondente desaparecerá, e ainda ficarão elétrones livres devido à diferença na dopagem;



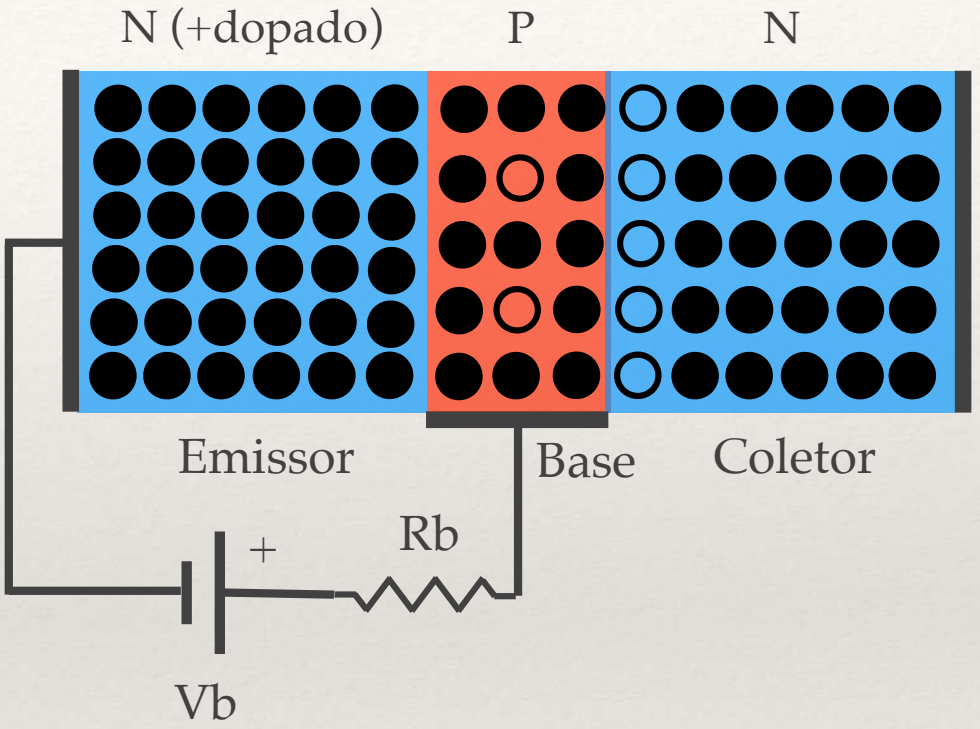
Transístor: duas junções

Similar ao diodo, mas...

- Temos duas camadas de depleção;
- O emissor tem uma dopagem maior. Com isso, a camada de depleção entre base e emissor libera elétrons em excesso;

Se fizermos circular uma corrente entre a base e o emissor:

- A camada de depleção correspondente desaparecerá, e ainda ficarão elétrones livres devido à diferença na dopagem;



Transístor: duas junções

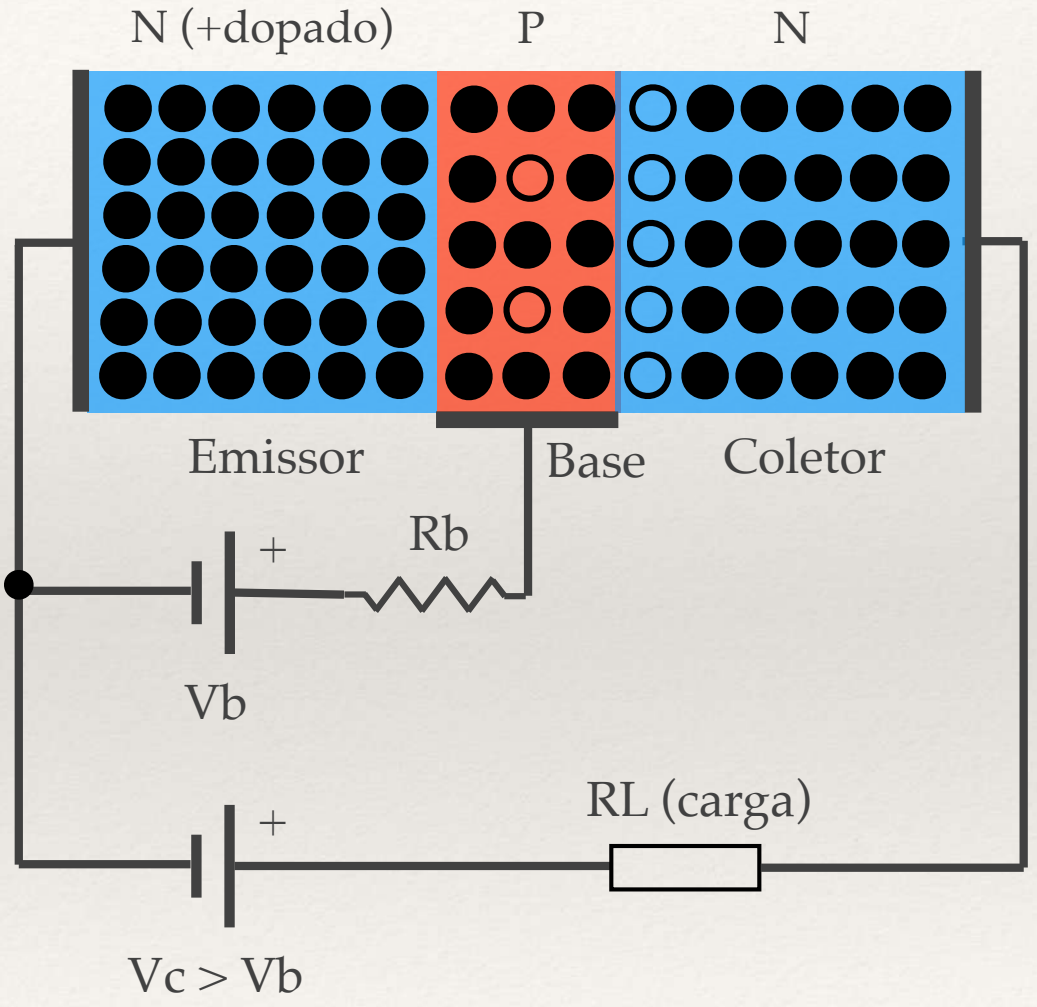
Similar ao diodo, mas...

- Temos duas camadas de depleção;
- O emissor tem uma dopagem maior. Com isso, a camada de depleção entre base e emissor libera elétrons em excesso;

Se fizermos circular uma corrente entre a base e o emissor:

- A camada de depleção correspondente desaparecerá, e ainda ficarão elétrones livres devido à diferença na dopagem;

Uma corrente entre coletor e base será controlada pela corrente de base.



Transístor: duas junções

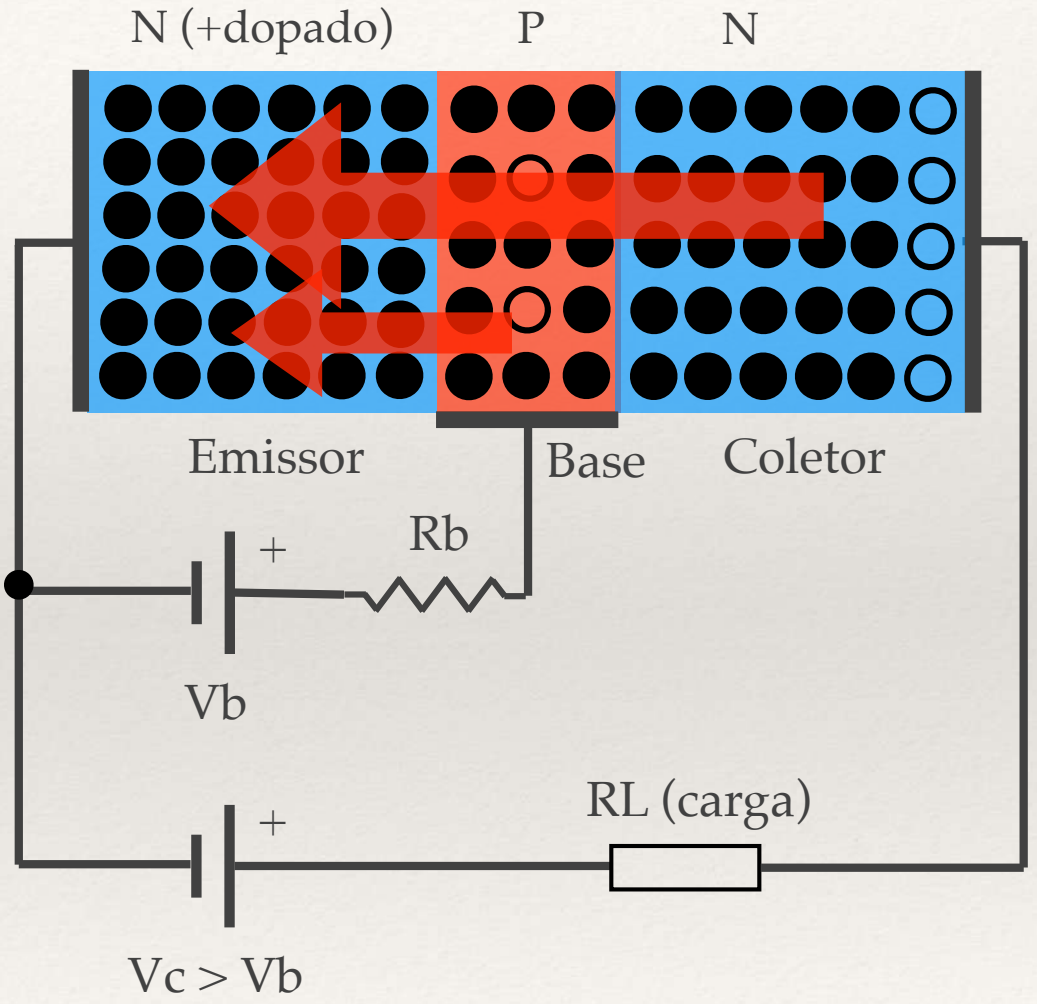
Similar ao diodo, mas...

- Temos duas camadas de depleção;
- O emissor tem uma dopagem maior. Com isso, a camada de depleção entre base e emissor libera elétrons em excesso;

Se fizermos circular uma corrente entre a base e o emissor:

- A camada de depleção correspondente desaparecerá, e ainda ficarão elétrones livres devido à diferença na dopagem;

Uma corrente entre coletor e base será controlada pela corrente de base.

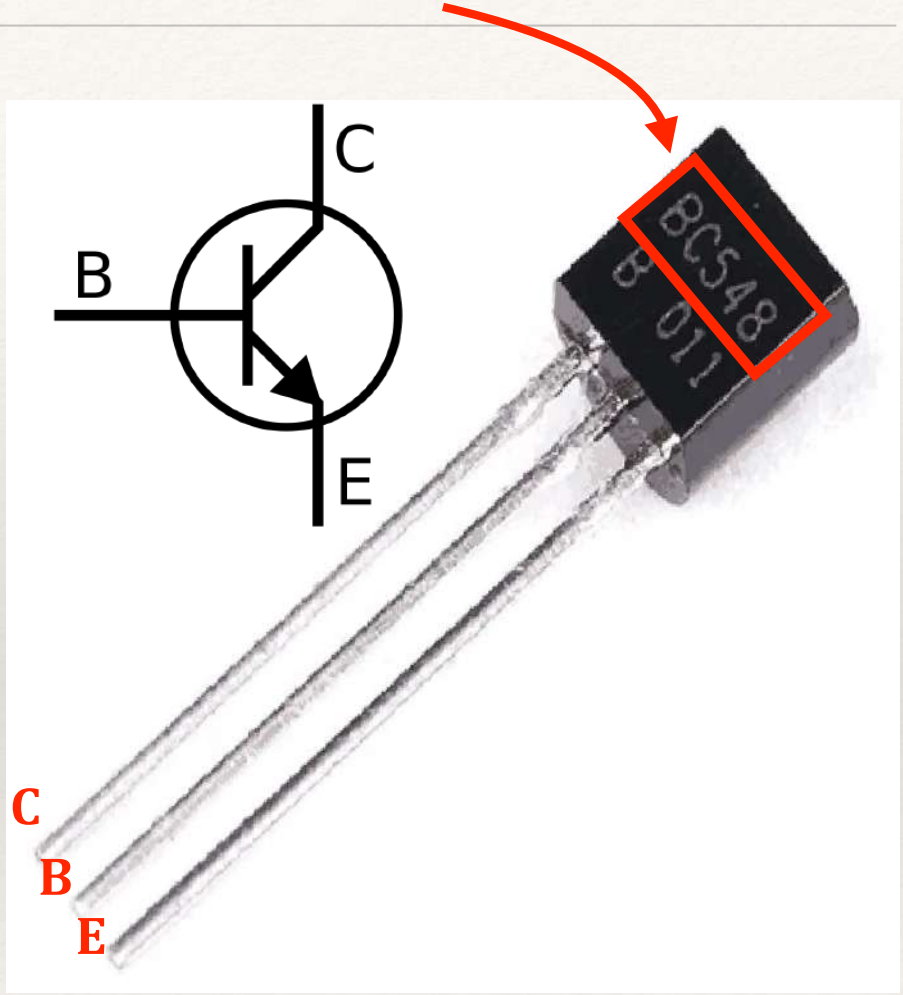


Transístor

Especificações estão associadas ao código do componente

- Código normalmente vem impresso no corpo do transistor, além de número do lote e outras especificações;
- Na figura ao lado, temos o BC548, transístor NPN popular de baixa potência.

Na folha de dados (*datasheet*), encontramos, entre outras coisas, a "pinagem" do componente.



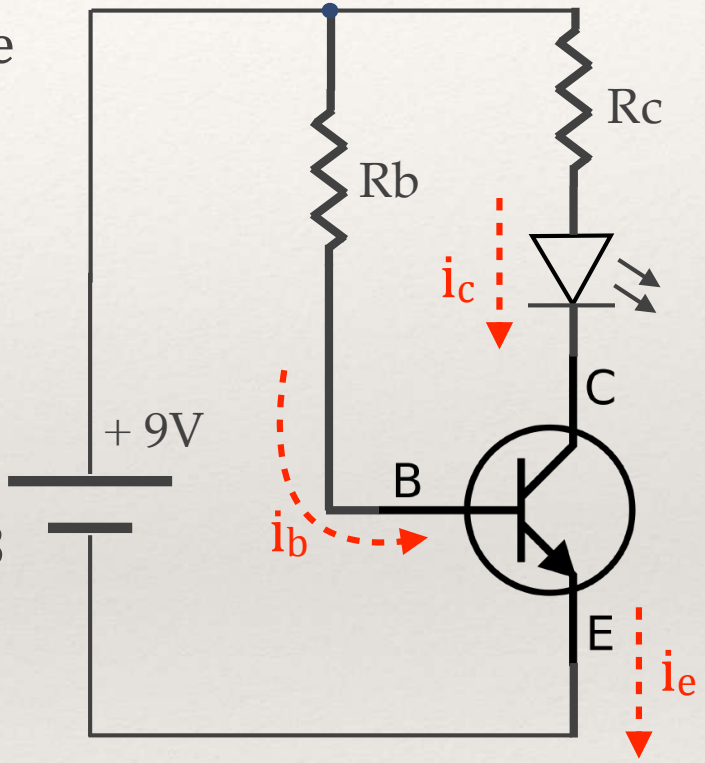
Polarizando um Transístor

"Polarizar" um transístor é projetar as correntes que devem passar pelos seus terminais

- $i_c = \beta \times i_b$ (β é o ganho do transístor)
- $i_e = i_c + i_b$
- $V_{be} \approx 0{,}7$ V (queda na junção)
- No PNP, o sentido de circulação da corrente se inverte.

A depender dos valores, o transistor pode ficar em 3 diferentes "zonas":

- Corte, Ativa (amplificação) e Saturação;
- Para operar como chave, nos interessa as zonas de corte e saturação;
- Valor prático para i_b é 10% de i_c para operar na zona de saturação independente de β.



Eletrônica nos μCUs

Em ambientes embarcados, a necessidade de conexão com ambientes externos determina aplicação de conceitos de eletrônica;

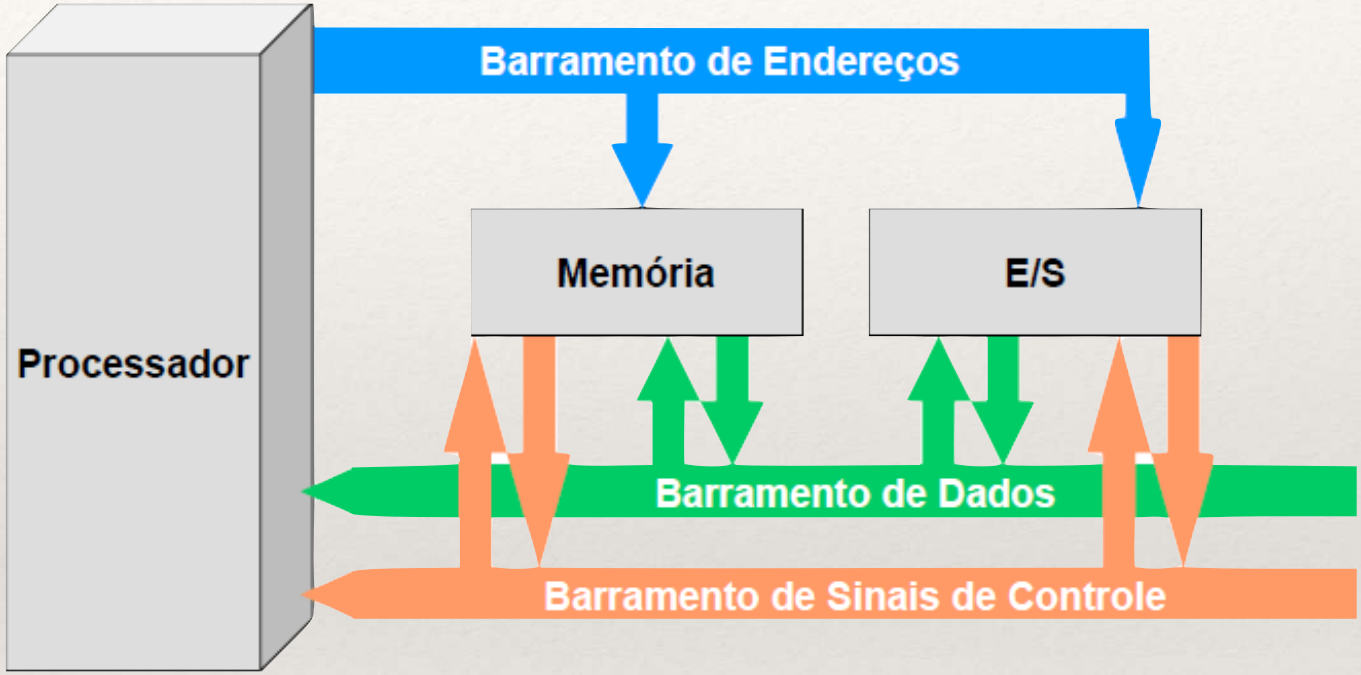
Além de microcontroladores, é típico encontrar diversos componentes externos fundamentais.

Barramentos

Os componentes trocam informações;

Na arquitetura de Von Neumann são 3 os barramentos:

- Endereços
- Dados
- Controle



As setas indicam o fluxo de informações pelos barramentos:

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

Barramentos

Barramento de Endereços

Fan-out no barramento de endereços:

Supondo 16 GBytes de memória, e palavra de 64 bits, temos:

$$\frac{16x1024x1024x1024x8}{64} = 1024x1024x1024x2 \approx 2 \text{ bilhões de palavras}$$

Cada uma das palavras precisa ler o barramento de endereços para identificar o destinatário.

Mesmo para correntes de entrada baixas (Ex.10uA), teríamos um consumo de $2 \times 10^9 \times 10^{-6} = 2000A$?!?

O problema exige soluções específicas !

Barramentos

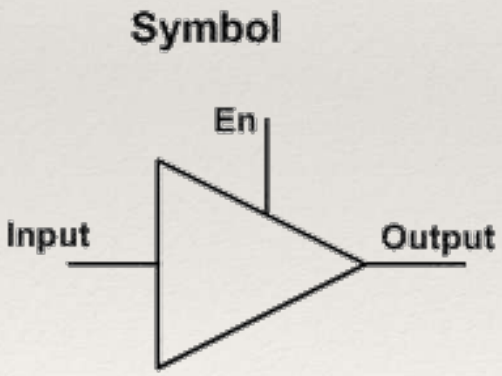
Barramento de Dados

Tri-state no barramento de dados:

Como duas saídas diferentes de memória podem compartilhar o mesmo barramento em uma leitura da CPU? (curto-circuito ?!?)

O terceiro estado (desconectado) é essencial;

Apenas dois dispositivos ficam “ativos”: UM emissor e um receptor.



Truth Table

En	Input	Output
0	X	Hi-Z
1	0	0
1	1	1

Fig. 3 Tri-state buffer symbol and truth table

Barramentos

Barramento de Sinais de Controle

Alguns sinais tipicamente presentes no barramento de Controle:

BREQ (*Bus Request*): requisição de barramentos por um periférico que passará a controlar o ambiente. Ex: DMA (*Direct Memory Access*)

CLK (Clock): sinal de *clock*;

INTR (*Interrupt Request*): solicitação de interrupção externa;

R/$\overline{\mathbf{W}}$ (*Read/*$\overline{\textit{Write}}$): identifica o tipo de operação que será feita na memória ou periférico. Em algumas arquiteturas, há pinos separados;

RDY (Ready): sinaliza a conclusão de alguns procedimentos;

RST (*Reset*): reinicialização dos componentes.

É comum termos sinais ativos em nível zero, o que é representado pela barra acima do sinal correspondente.

Barramentos

Vantagens

- A ampliação do número de dispositivos e usuários é trivial;
- Baixo custo.

Desvantagens

- Compartilhamento pode provocar retenção de tráfego, com queda de desempenho, e possível *jitter* associado.

Dispositivos Adicionais

Circuitos/*chips* específicos podem ser incluídos em um projeto

- *Buffers*
  - Permite a interconexão de circuitos com performances diferentes;
  - Podem ser unidirecionais ou bidirecionais.
- *Latches*
- *Drivers*
- *Clock* externo
- Memórias
- Periféricos

Dispositivos Adicionais

Circuitos/*chips* específicos podem ser incluídos em um projeto

- *Buffers*
- *Latches*
  - Agrega a capacidade de “memorizar" temporariamente o valor de saída;
  - São apenas unidirecionais.
- *Drivers*
- *Clock* externo
- Memórias
- Periféricos

Dispositivos Adicionais

Circuitos/*chips* específicos podem ser incluídos em um projeto

- *Buffers*
- *Latches*
- *Drivers*
  - Compatibiliza saídas digitais de baixa potência com cargas elevadas;
  - Em algumas situações, agregam circuitos analógicos especializados para controle de cargas específicas, como motores *brushless*, de passo, sistemas trifásicos etc.
- *Clock* externo
- Memórias
- Periféricos

Dispositivos Adicionais

Circuitos/*chips* específicos podem ser incluídos em um projeto

- *Buffers*
- *Latches*
- *Drivers*
- *Clock* externo
  - É muito comum a existência de circuitos internos de *clock* nos microcontroladores, mas a demanda por maior exatidão e precisão pode exigir circuitos externos de maior qualidade.
- Memórias
- Periféricos

Dispositivos Adicionais

Circuitos/*chips* específicos podem ser incluídos em um projeto

- *Buffers*
- *Latches*
- *Drivers*
- *Clock* externo
- Memórias:
  - Esta é uma demanda essencial nos microprocessadores;
  - Não é suportada por microcontroladores, e é complexa até em SBCs como o Raspberry Pi.
- Periféricos

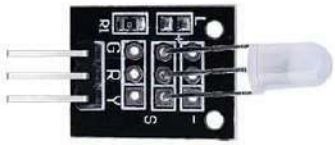
Dispositivos Adicionais

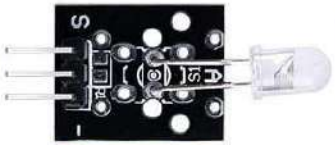
Circuitos/*chips* específicos podem ser incluídos em um projeto

- *Buffers*
- *Latches*
- *Drivers*
- *Clock* externo
- Memórias
- Periféricos:
  - Até mesmo em placas de protótipo às vezes são oferecidos RPCs, interfaces de rede, sensores de temperatura, transmissores de rádio, displays etc.

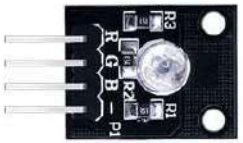
Dispositivos de Entrada e Saída


JOYSTICK

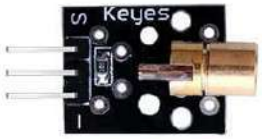

TWO-COLOR


IR EMISSION

MEMBRANE SWITCH

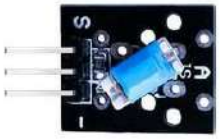

RGB LED


7 COLOR FLASH



LASER EMIT


SMD RGB


TILT-SWITCH

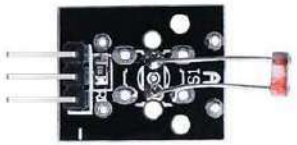
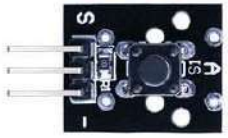
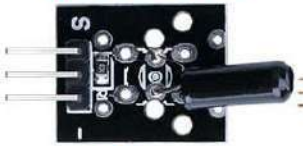


PHOTO-RESISTOR

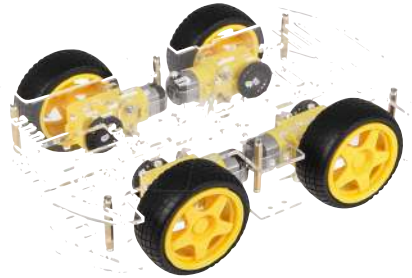
ULTRASONIC SENSOR

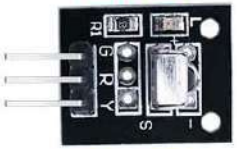

BUTTON

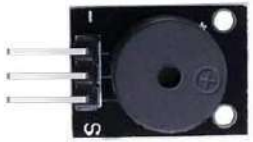
ACTIVE BUZZER


SHOCK


WATER LEVEL SENSOR

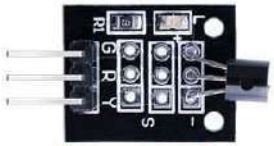



IR RECEIVER


PASSIVE BUZZER


DS 1307 RTC MODULE


LCD 1602 MODULE


18B20 TEMP


ROTARY ENCODER


RELAY


HC-SR501


GY-521


POWER SUPPLY

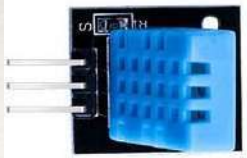
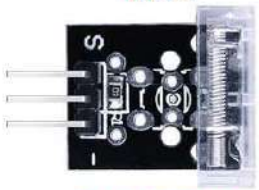

TEMP AND HUMIDITY


PHOTO-INTERRUPTER


TAP MODULE


TRACKING


AVOIDANCE


MAGNETIC SPRING


DIGITAL TEMPERATURE


FLAME


LINEAR HALL


BIG SOUND


METAL TOUCH

SMALL SOUND

Interface na Prática

A tensão elétrica dos sinais trocados com o mundo externo codificam a informação transferida

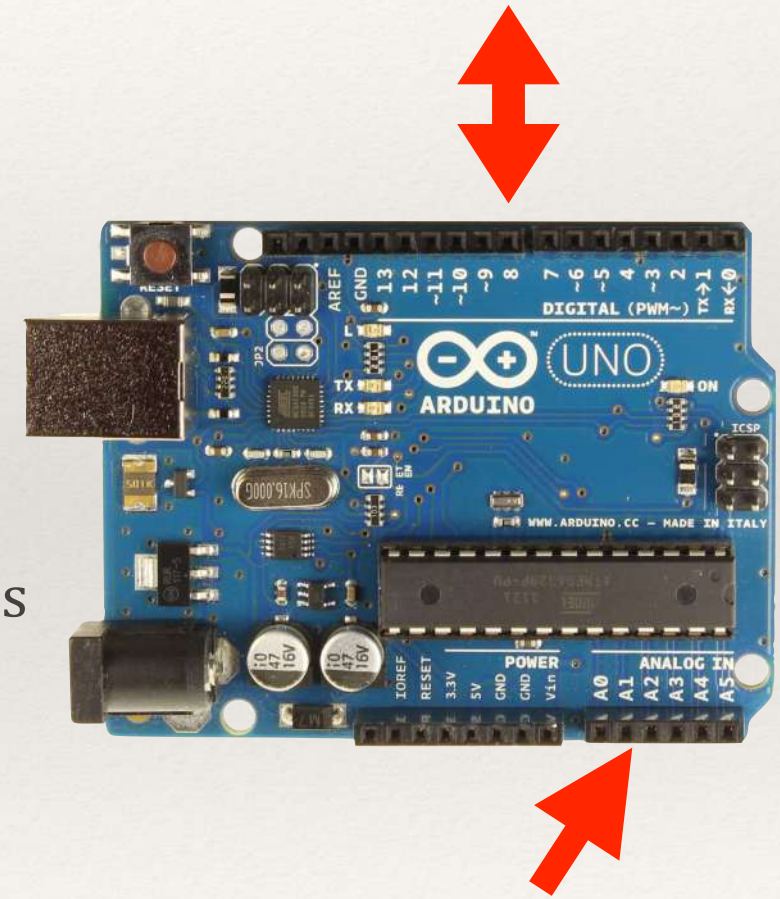
Portas digitais ESCREVEM E LEEM sinais codificados digitalmente

- Tensão alta (HIGH) = 1;
- Tensão baixa (LOW) = 0;

Portas analógicas LEEM sinais codificados analogicamente

- O valor da tensão indica o valor analógico;
- Teoricamente, existem “infinitos" valores (na prática não. Por quê?)

Sinais Elétricos
(valores de tensão digitais)



Sinais Elétricos
(valores de tensão analógicos)

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).

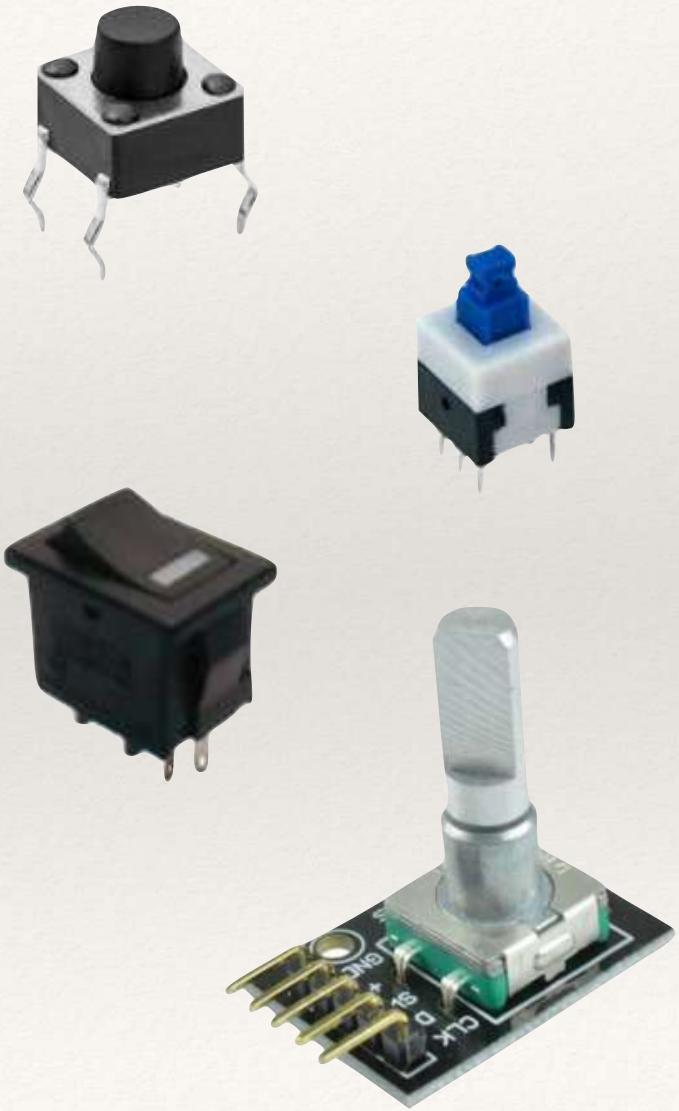
Dispositivos de Entrada

- Chaves e Botões
  - Método básico de entrada;
  - Chaves momentâneas, fixas e *encoders*;
  - Foto-isoladores;
  - *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*;
- Foto-isoladores;
- *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;

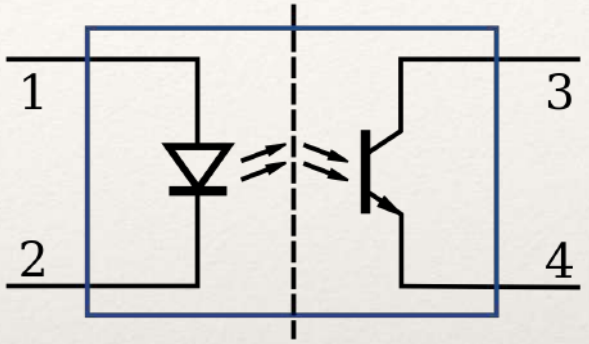
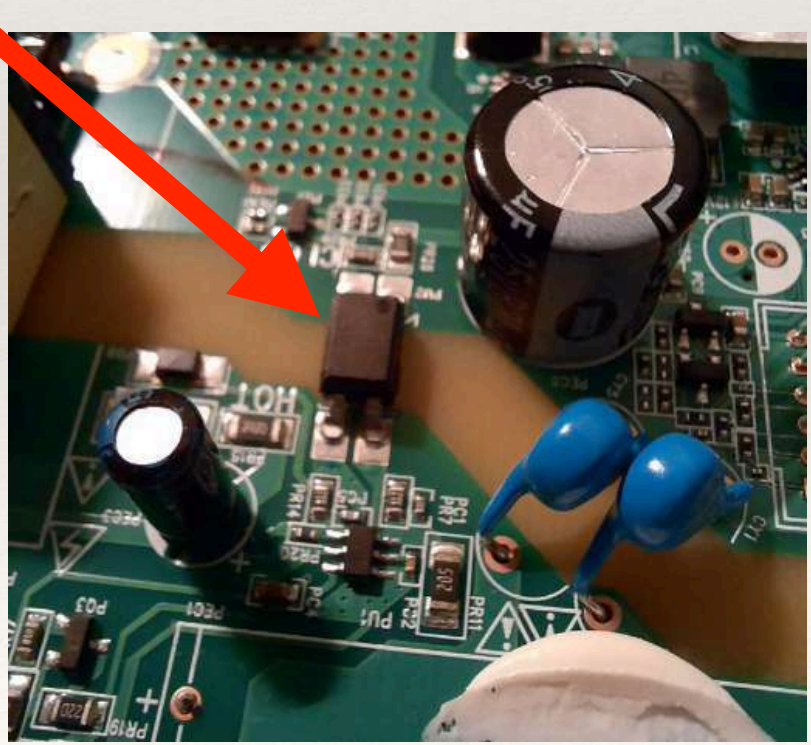


Foto-isoladores

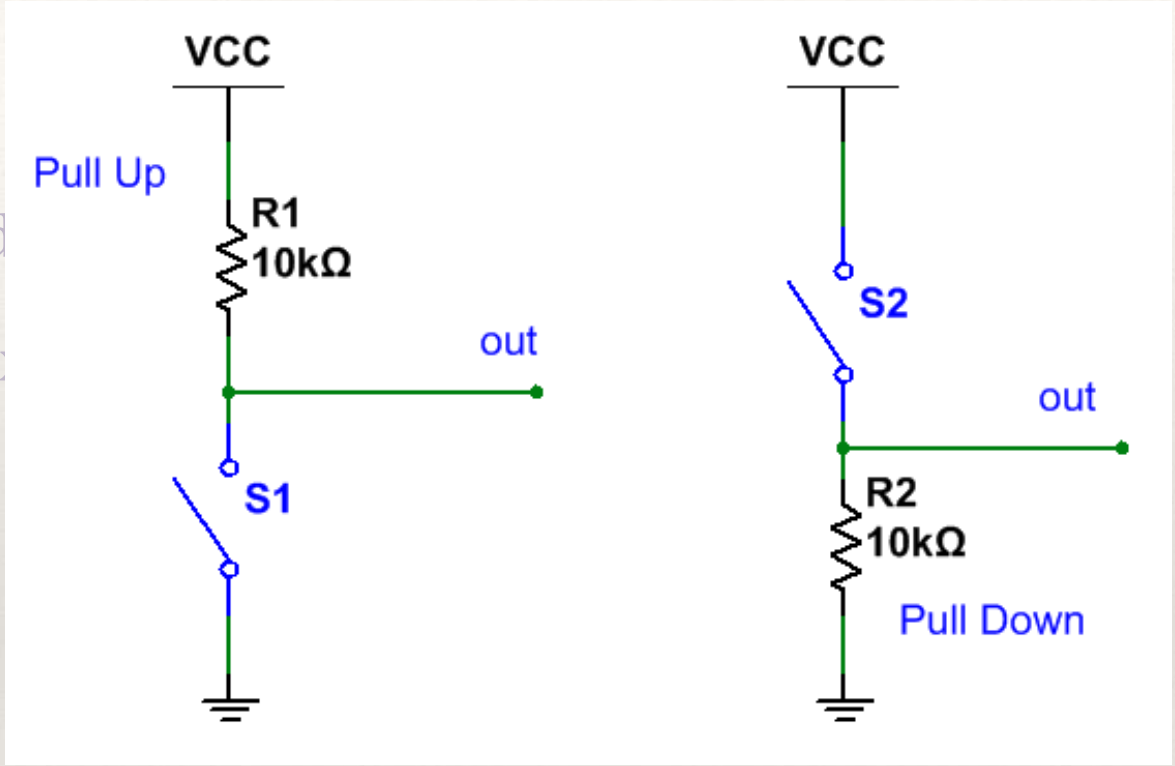
- Existem sob a forma de foto-diodos, foto-transistores etc;
- No foto-transistor do exemplo, quem excita a base é a luz do LED interno;
- Embora tipicamente operem como chave, podem atuar como amplificadores;
- Promovem isolamento típico de alguns milhares de volts (5.300V), com resistência de $10^{11}\Omega$.
- Na placa do exemplo, notem o espaço de isolamento.



Figuras: https://en.wikipedia.org/wiki/Opto-isolator; Especificação: TIL111

Dispositivos de Entrada

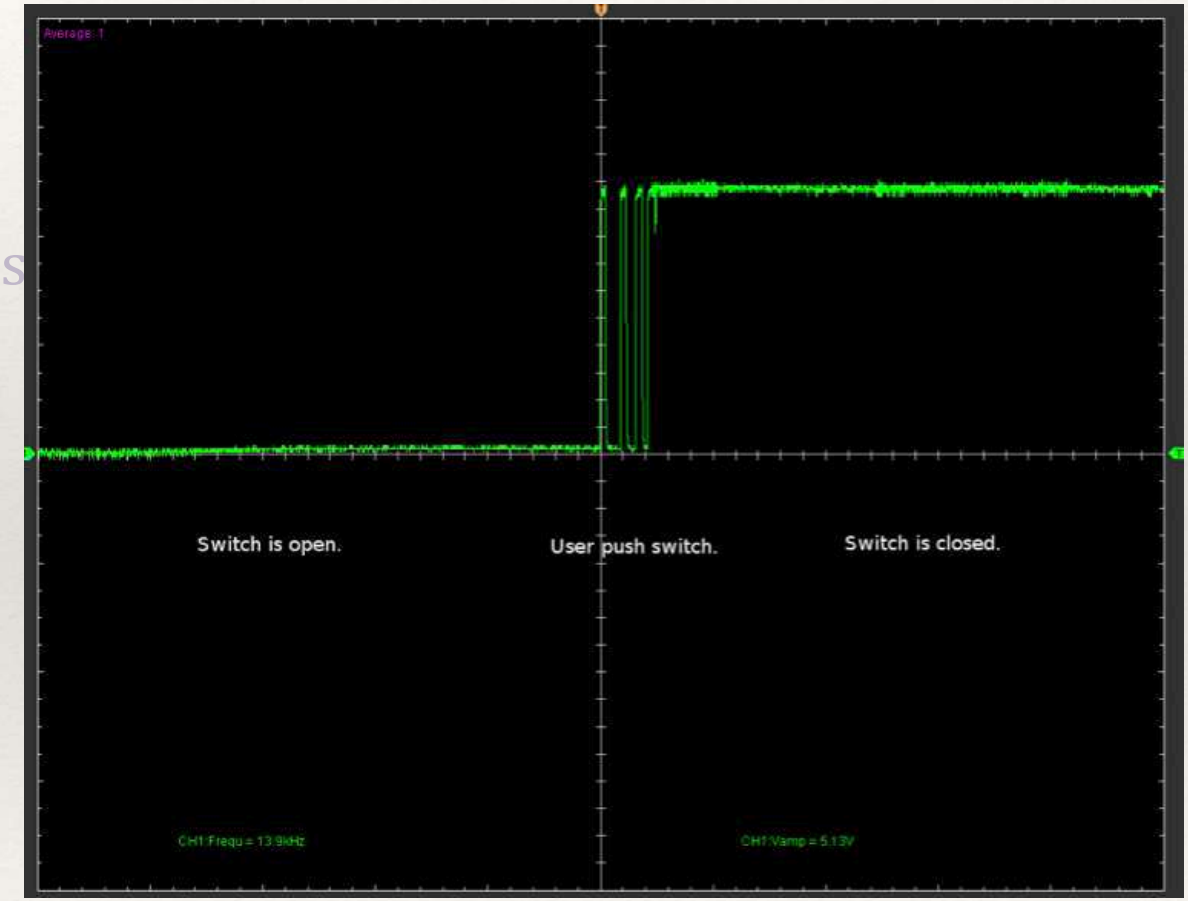
- Chaves e Botões
  - Método básico de entrad
  - Chaves momentâneas, fi
  - Foto-isoladores;
  - *Pull Up* e *Pull Down*
  - Efeito *Bouncing*
    - *Debouncing* por SW ou HW



Dispositivos de Entrada

Chaves e Botões

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



https://docs.arduino.cc/built-in-examples/digital/Debounce/

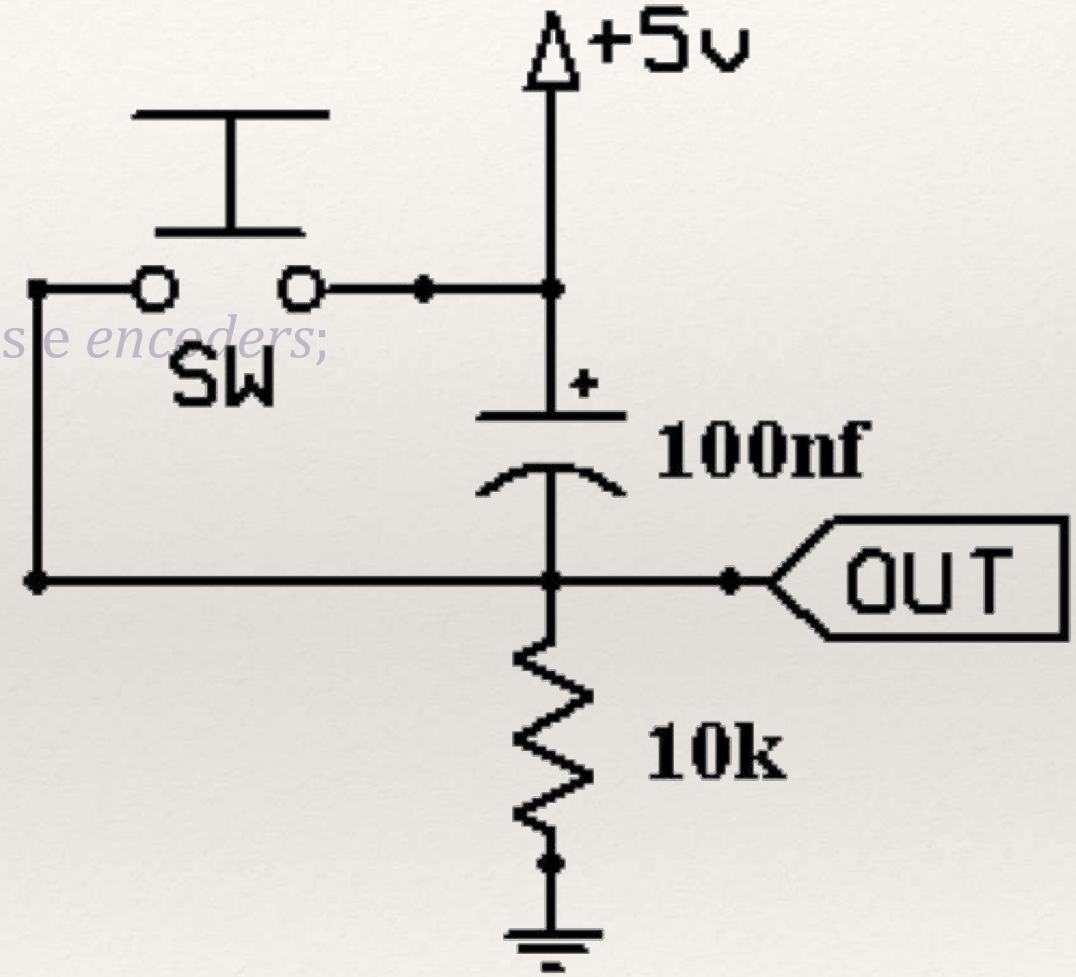
Dispositivos de Entrada

Chaves e Botões

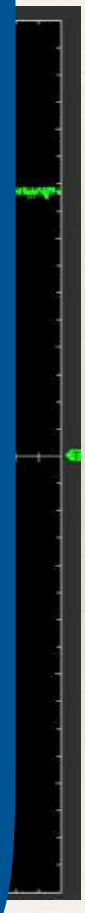
- Método básico de entrada;
- Chaves momentâneas, fixas e *encoders*;
- Foto-isoladores;
- *Pull Up* e *Pull Down*

Efeito *Bouncing*

- *Debouncing* por HW ou SW

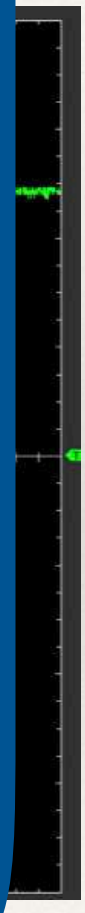


```
// Constantes para indicar número dos pinos utilizados
const int buttonPin = 2;
const int ledPin = 13;

// Variables will change:
int ledState = HIGH;          // status atual do pino de saída (LED)
int buttonState;              // status atual do pino de entrada
int lastButtonState = LOW;  // status anterior do pino de entrada

unsigned long lastDebounceTime = 0;  // ultimo momento em que o pino mudou de status
unsigned long debounceDelay = 50;    // tempo para o "debounce"

void setup() {
  pinMode(buttonPin, INPUT);
  pinMode(ledPin, OUTPUT);

  digitalWrite(ledPin, ledState);    // Define status atual da saída
}
```



https://docs.arduino.cc/built-in-examples/digital/Debounce/

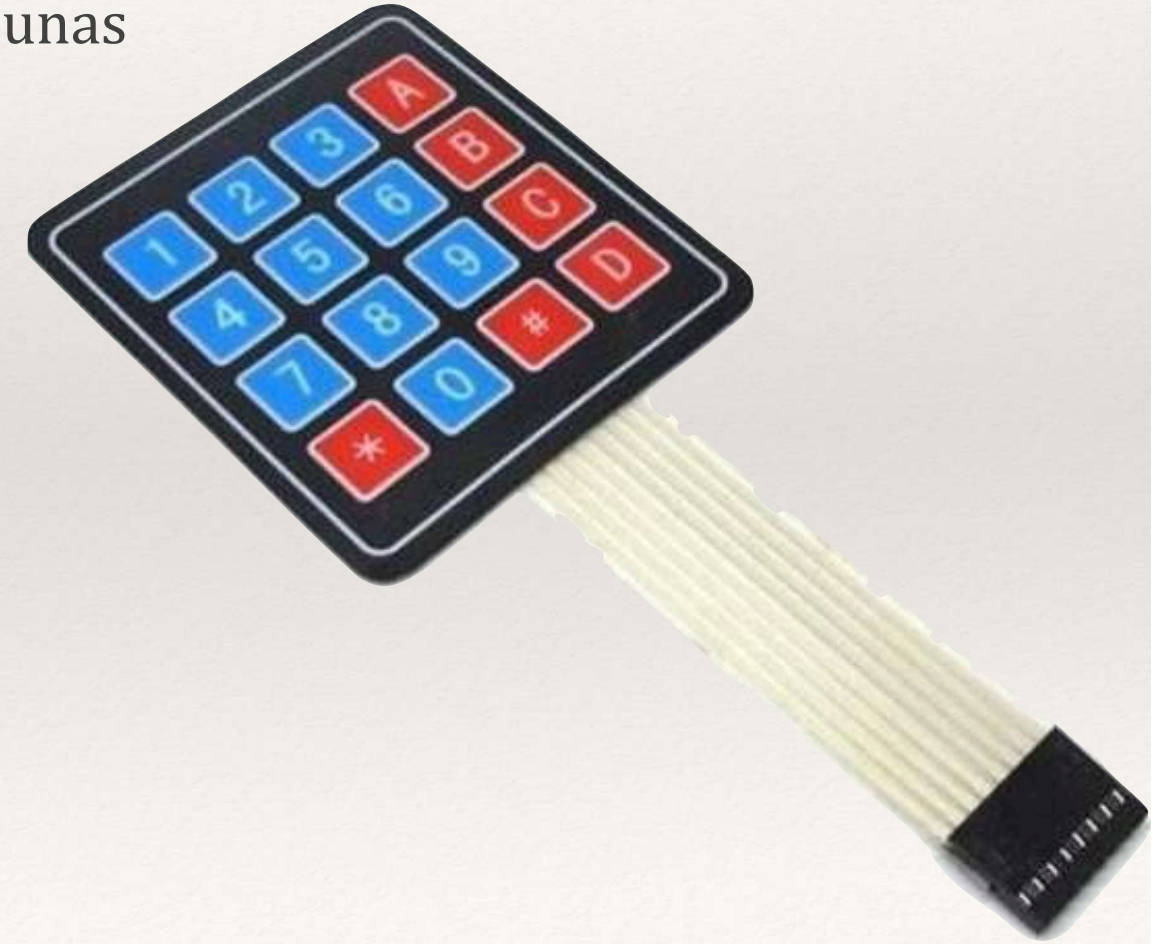
```
void loop() {
  // Lê o status atual do botão de entrada
  int reading = digitalRead(buttonPin);

  // Se o status do botão mudou, seja por ação do operador, seja por ruído, comece a contar o tempo
  if (reading != lastButtonState) lastDebounceTime = millis();

  // Se decorreu o tempo definido para o “debounce”
  if ((millis() - lastDebounceTime) > debounceDelay) {

    // Se o status mudou
    if (reading != buttonState) {
      buttonState = reading;

      // Altere o status do LED apenas se o status for HIGH
      if (buttonState == HIGH) {
        ledState = !ledState;
      }
    }
  }

  // Defina o status do LED
  digitalWrite(ledPin, ledState);

  // Salva a leitura para a próxima execução do loop
  lastButtonState = reading;
}
```



Dispositivos de Entrada

Teclados (ex. teclado de membrana)

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

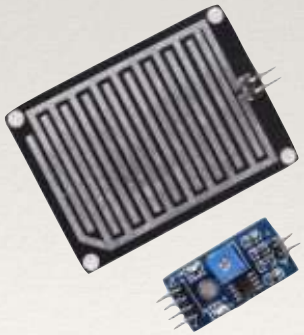


Dispositivos de Entrada

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 Saída

LEDs

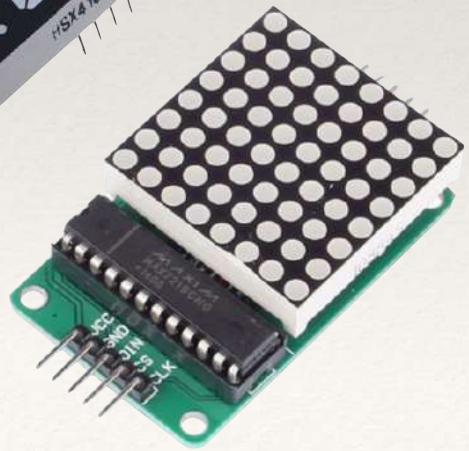
- Saída analógica típica para testes;
- Admite modulação via PWM;
- Vermelho, Verde, Amarelo, Azul, Branco e RGB (3 saídas).

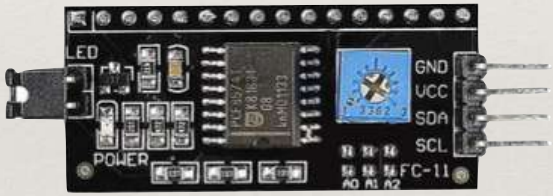
Dispositivos de Saída

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

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

Cátodo ou Anodo comum;



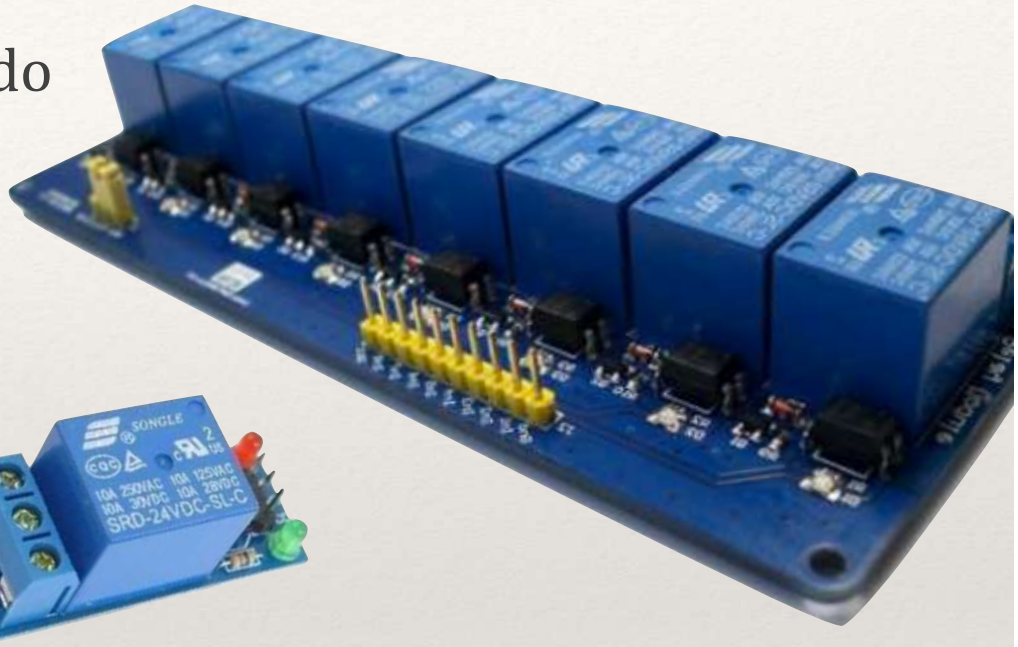


Dispositivos Saída (Relês)

Nem todo dispositivo pode ser acionado por um transístor

Cargas AC ou de alta potência;

Necessidade de isolamento;



Relês são dispositivos mecânicos

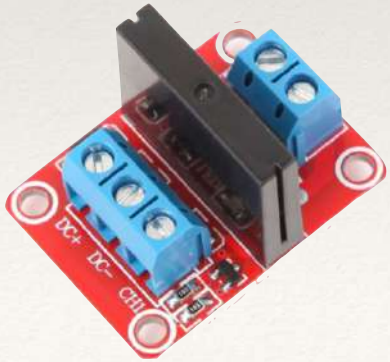
- ⬆ Promovem grande isolamento;
- ⬆ Suportam cargas DC ou AC;
- ⬇ São lentos;
- ⬇ Consomem potência significativa;
- ⬇ Vida útil é baixa
  Suportam até 300.000 acionamentos (típ)

"Relês" de estado sólido

- ⬆ São rápidos;
- ⬆ Consomem pouca potência;
- ⬆ Vida útil elevada;
  Suportam até 10.000.000 de acionamentos (típ)

Tem isolamento inferior aos relês

Suportam cargas AC



[Moraes, Cícero Couto] Engenharia de Automação Industrial 2ªEd. p.27

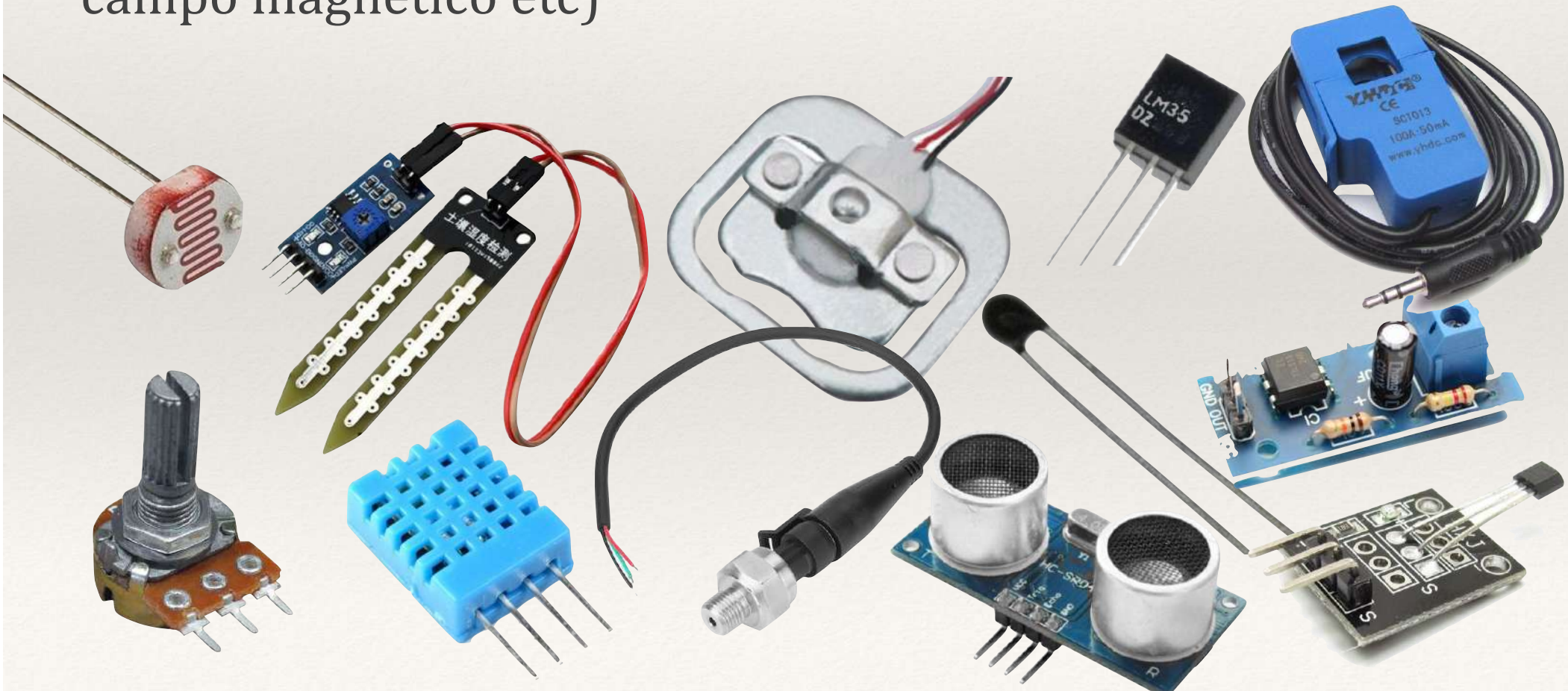
Dispositivos de Saída

Servos, motores convencionais e de passo

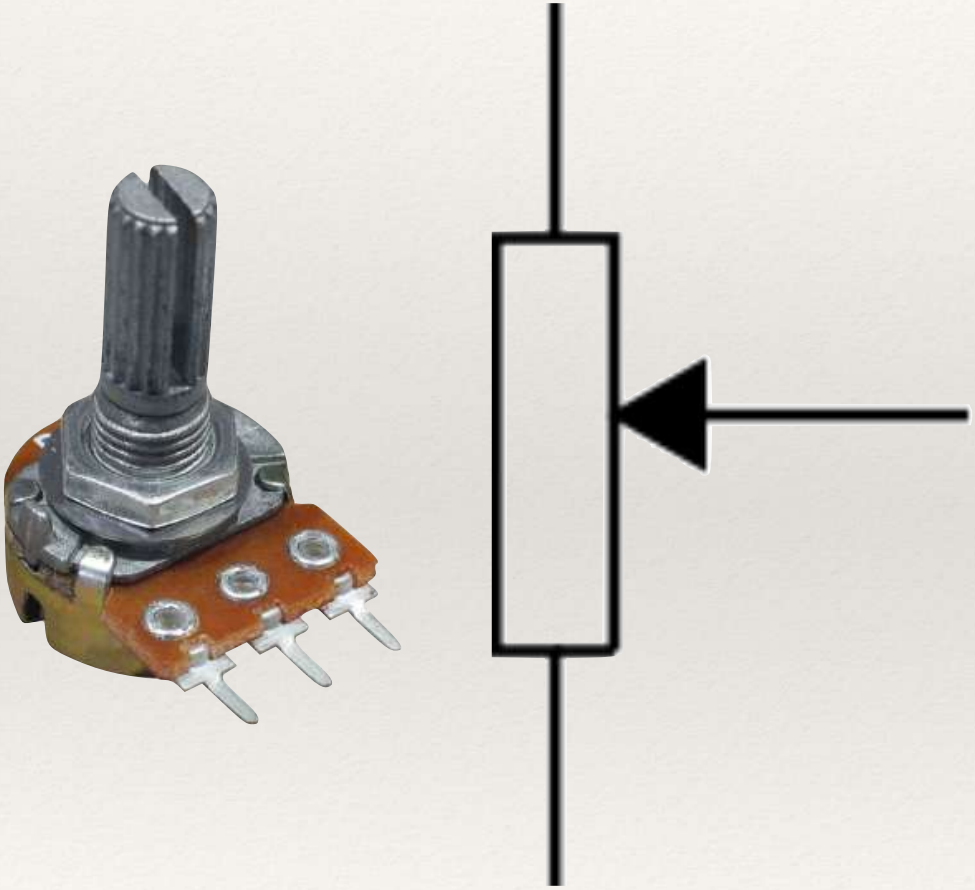
Aula 06

Dispositivos de Entrada

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



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 é utilizado em diversas situações.

Sensores “medem” tensão elétrica

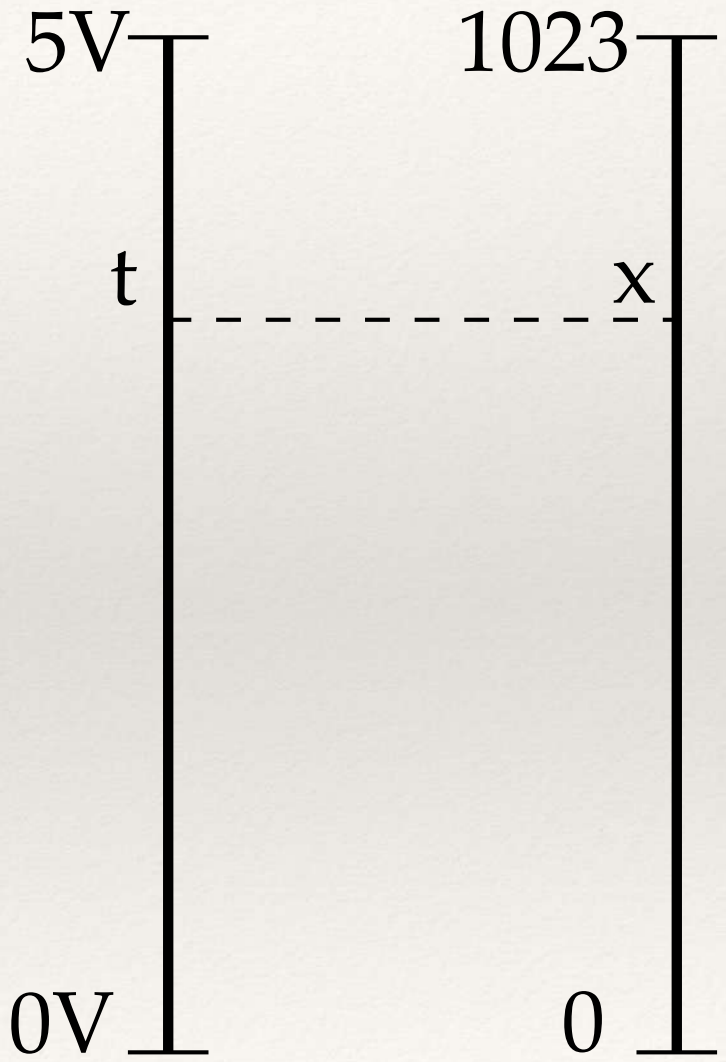
Tensão elétrica é injetada em uma porta analógica do microcontrolador

- 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
- $2^{10} = 1024$



Tensão é convertida para um número

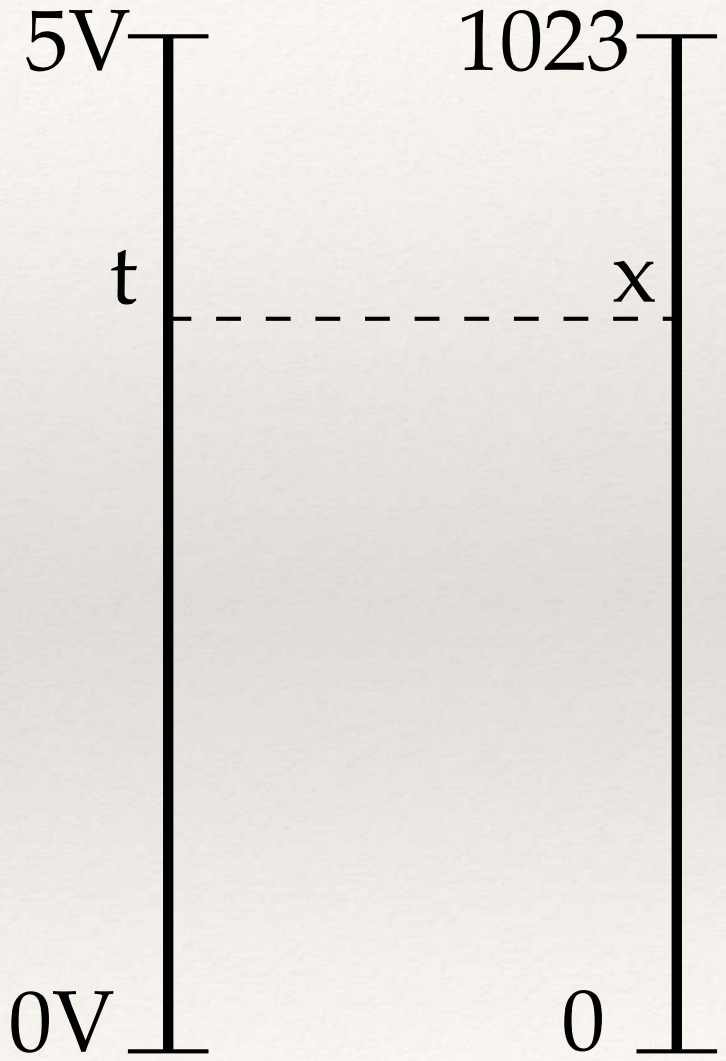
Medições são proporcionais graças à linearidade* do sensor;

Conversão é básica, por regra de 3

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

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

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



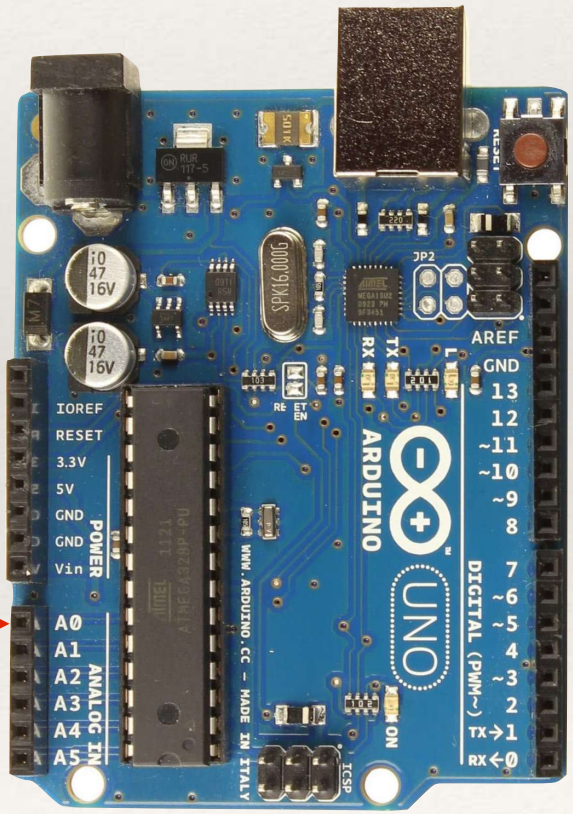
* Veremos depois

Medição analógica básica

Medição de Tensões Disponíveis no próprio Arduíno

- Sem componentes externos;
- Avaliar as medições com base no valor numérico medido pelo ADC;

Alimentação via USB



Entrada a ser medida

Medição analógica básica

Os resultados serão vistos no Monitor Serial
Utilizaremos a entrada A0 do Arduíno

Salvamos o valor lido anteriormente

Leitura analógica da entrada

Mostra apenas se houver alteração

Número de amostragens < 10 por segundo

```
void setup() {

  Serial.begin(115200);
  pinMode(A0,INPUT);

}

int entrada, entradaAnterior;

void loop() {

  entrada = analogRead(A0);

  if( entrada != entradaAnterior ) {
    Serial.println( entrada );
    entradaAnterior = entrada;
  }

  delay(100);

}
```

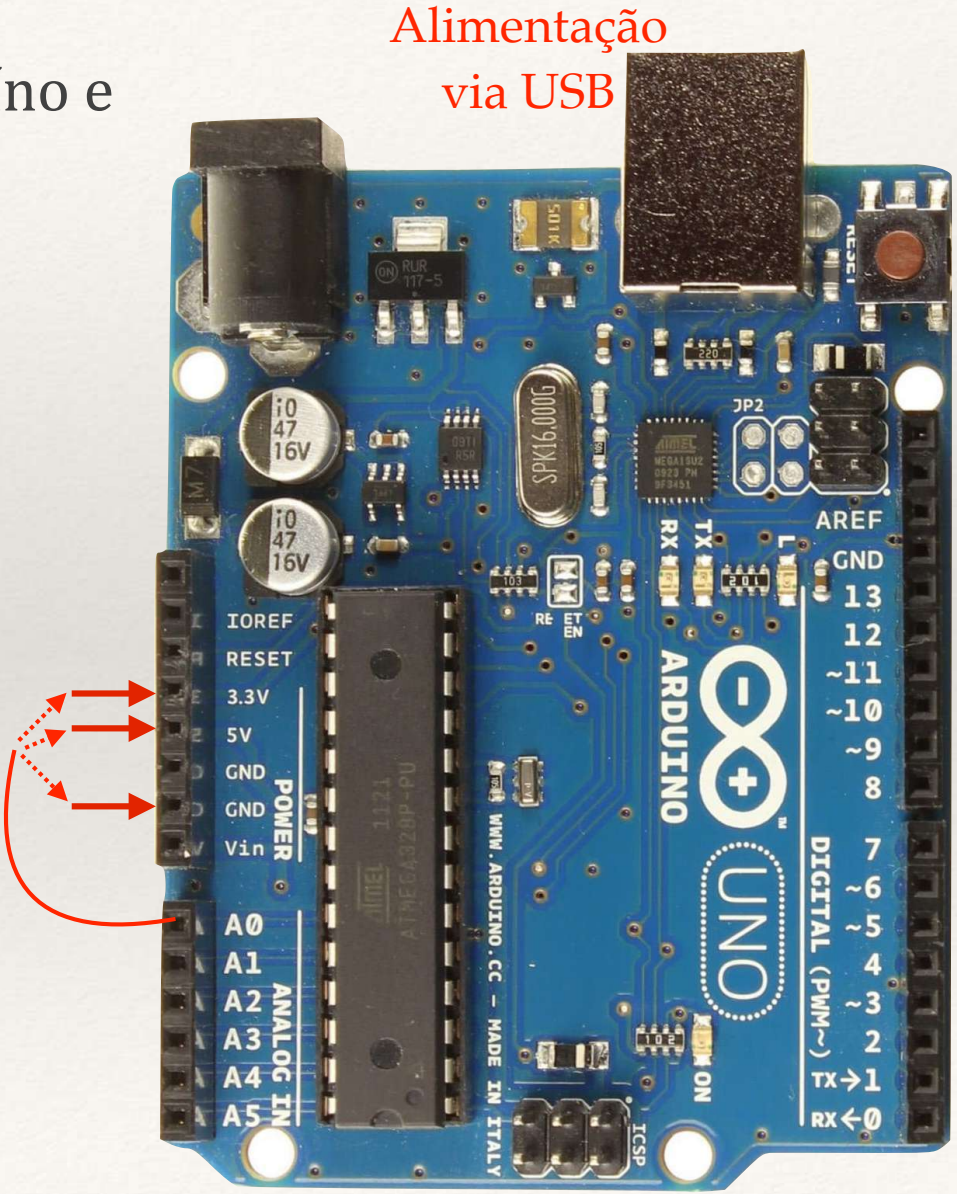
Medição analógica básica

Circuito básico, com apenas um Arduíno e um cabo

- Simplicidade, sem equip. externos.

Escolha da tensão de entrada:

- 5V (Vcc)
- 3,3V (saída de alimentação)
- 0V (GND)



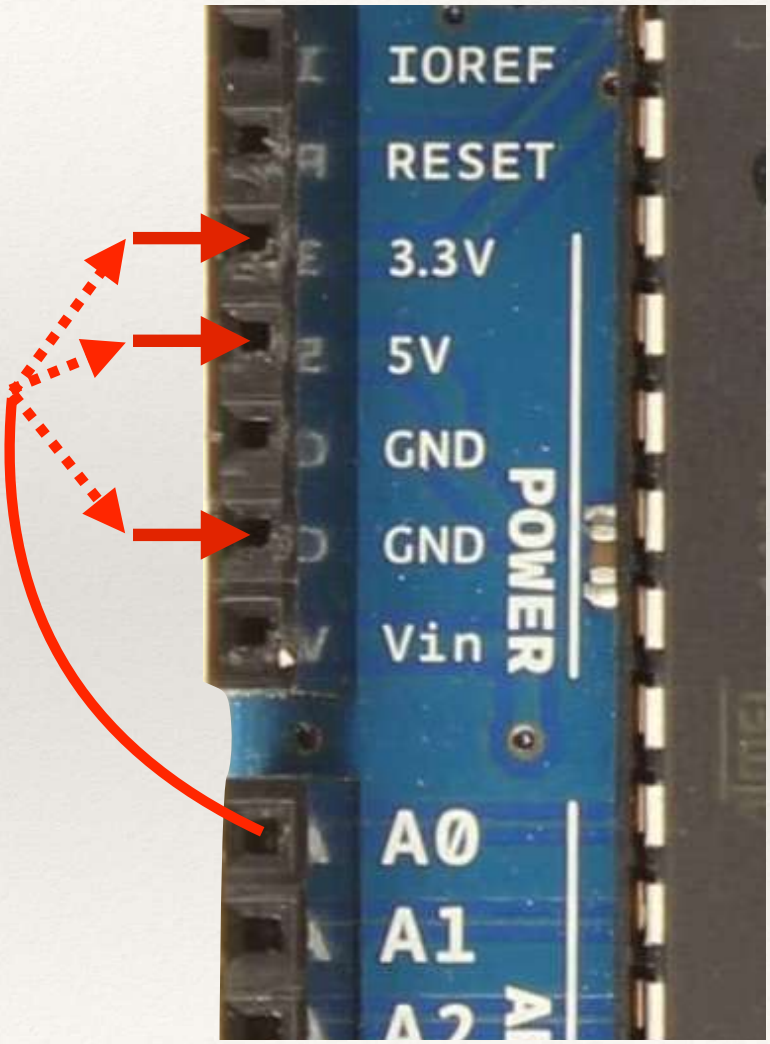
Medição analógica básica

Circuito básico, com apenas um Arduíno e um cabo

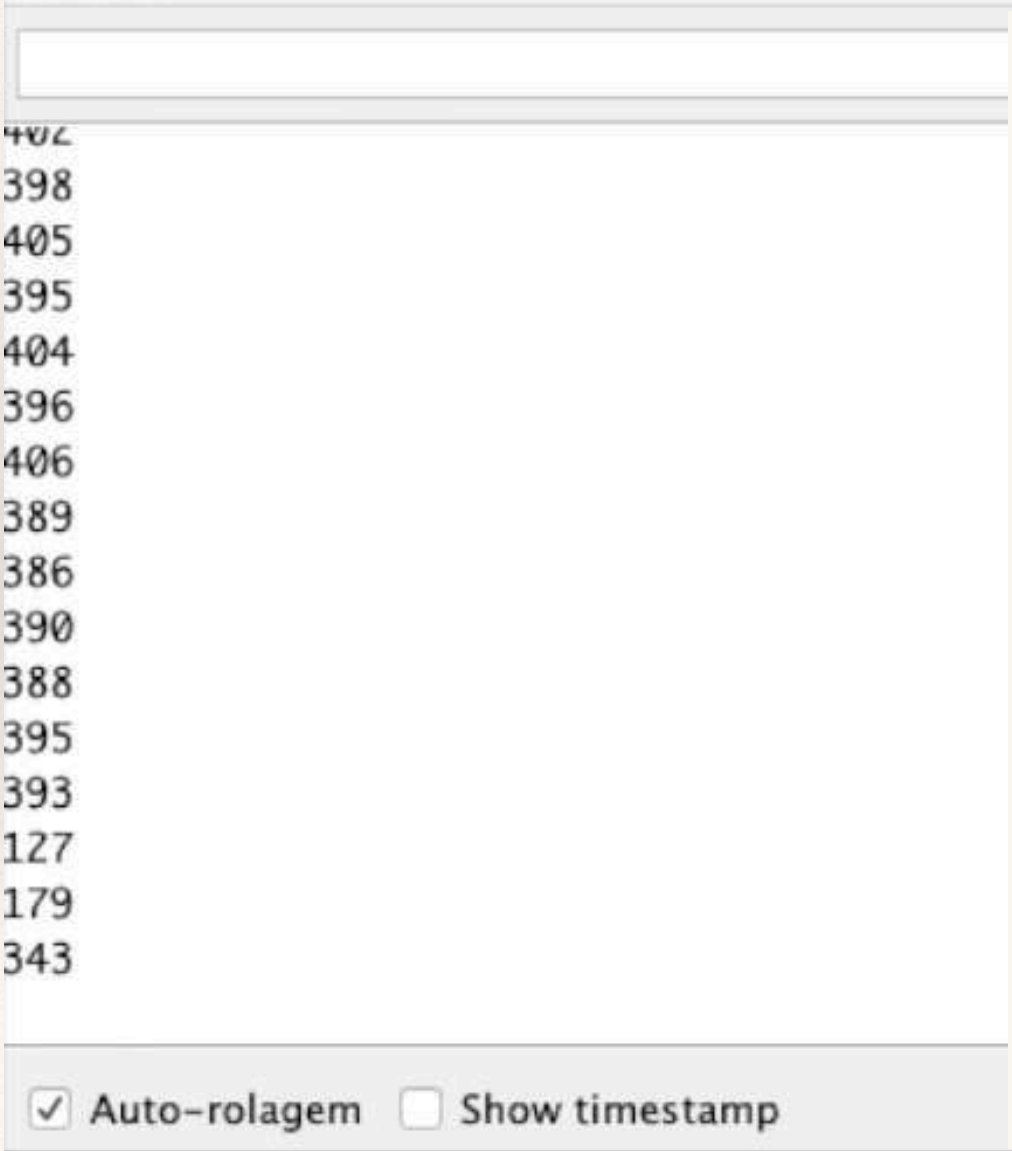
- Simplicidade, sem equip. externos.

Escolha da tensão de entrada:

- 5V (Vcc)
- 3,3V (saída de alimentação)
- 0V (GND)



Medição analógica básica



O que percebemos?

- Entradas desconectadas são “dominadas” pelo ruído;
- Há alguma instabilidade quando medimos a tensão de 3,3V.

Especificação básica

- Conversor A/D de 10 bits (0~1023).

Cuidado com a conexão !

- Interligar portas erradas pode danificar o Arduíno, e até a porta USB do seu computador !

Instabilidade

A instabilidade é um sintoma da falta de precisão;

- Mas o que é “precisão" ?

A precisão, exatidão e a resolução, além da linearidade, são parâmetros importantes, e serão posteriormente;

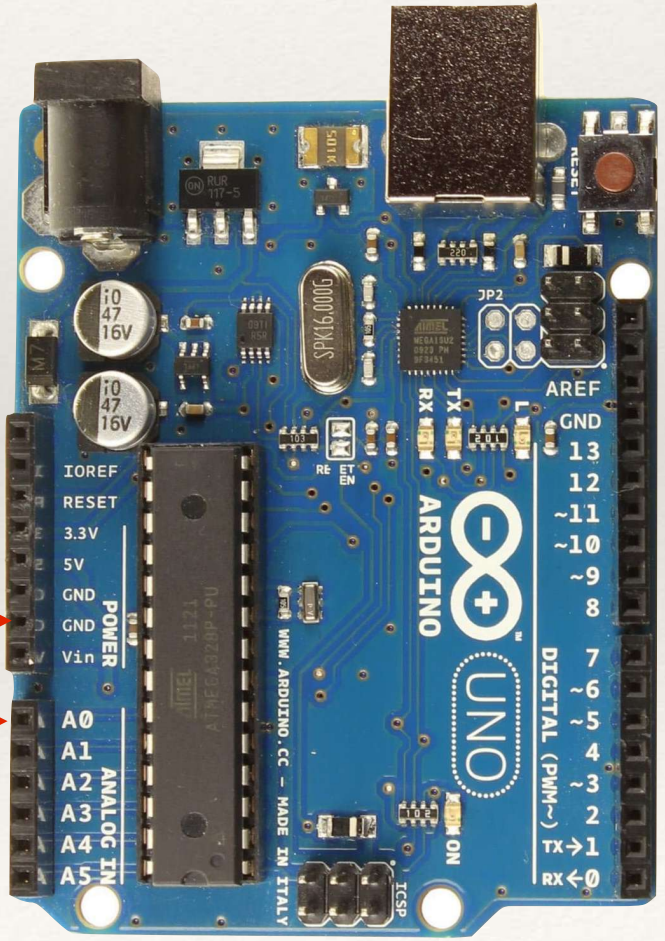
A imprecisão costuma afetar mais os valores intermediários de uma escala (no caso do experimento, seus efeitos apareceram mais na medição dos 3,3V).

Medição analógica típica

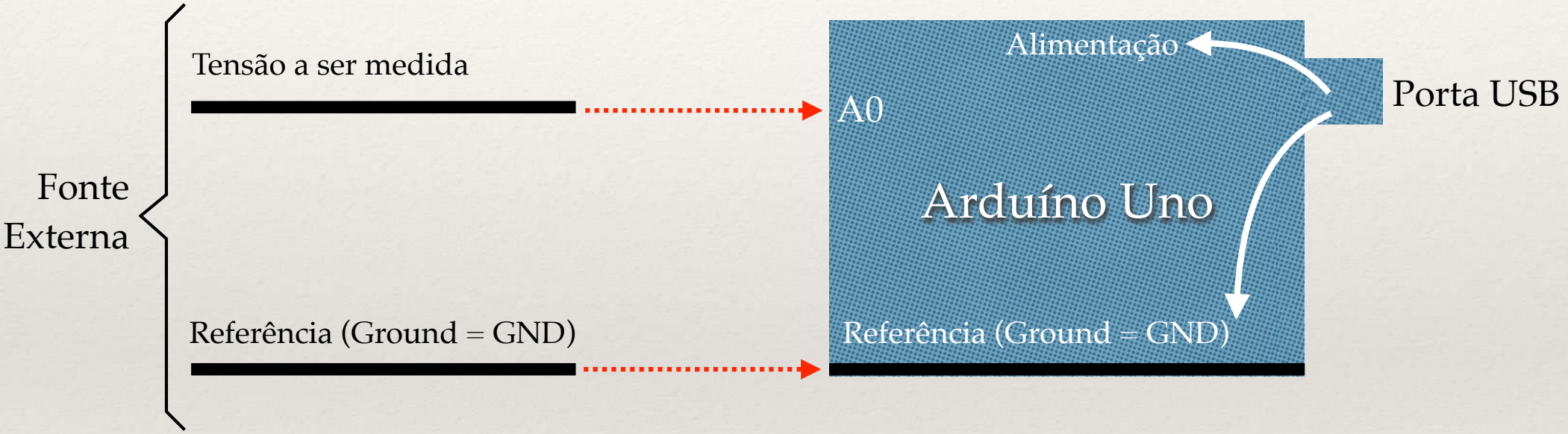
Medição de Tensão Externa 0~5V;

- Sem ajuste de escala;
- Referência: Vcc (5V);
- Medição numérica e de tensão;
- Apenas o Arduíno, sem componentes externos.

Alimentação via USB



Medição analógica típica



Medição analógica típica

O valor da tensão não é inteiro

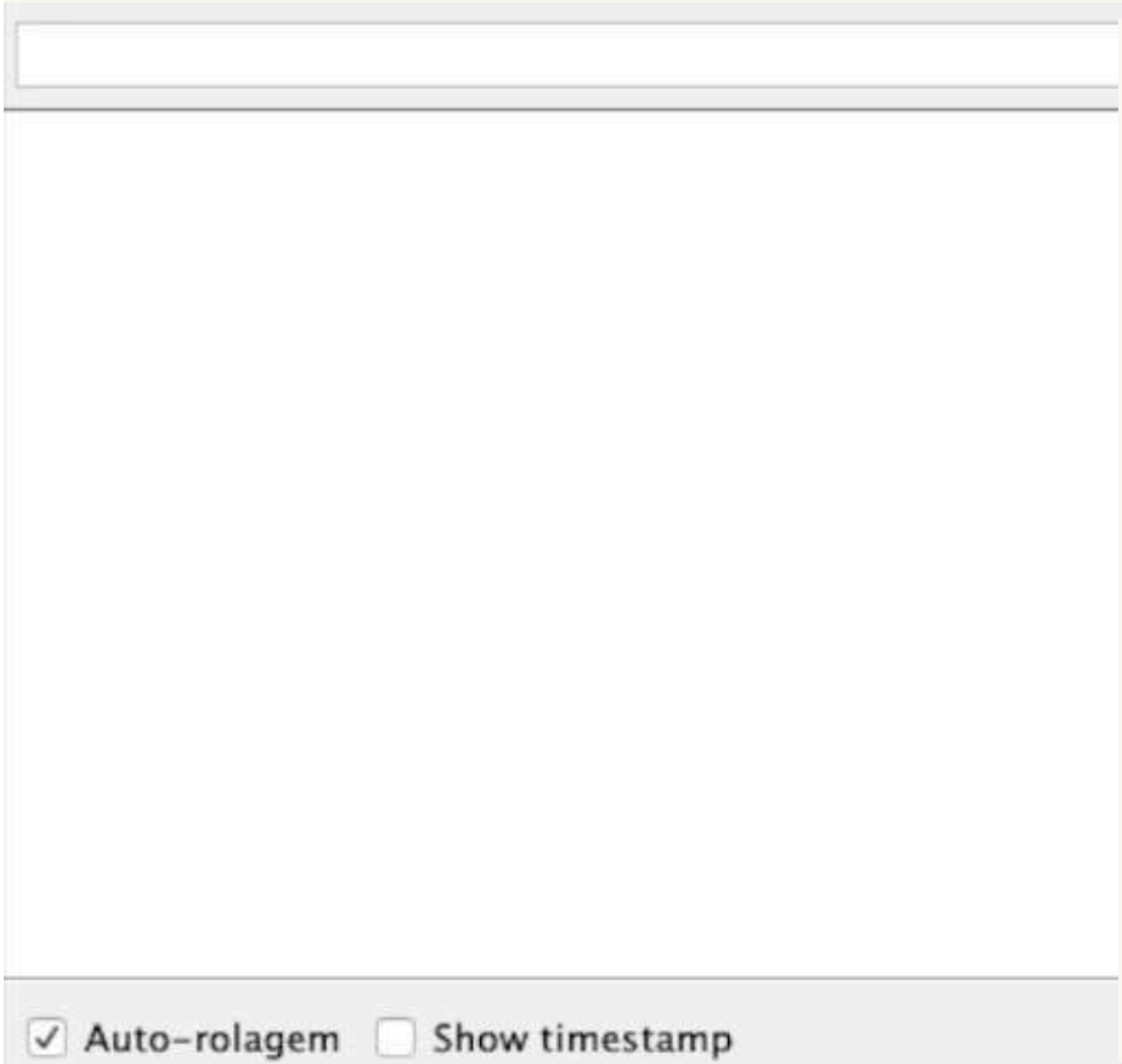
Cálculo da tensão

(usamos a casa decimal para garantir o processamento das constantes como “float”)

O valor da tensão terá 2 casas decimais

```
void setup() {

  Serial.begin(115200);
  pinMode(A0,INPUT);

}

int entrada, entAnt;
float tensao;

void loop() {

  entrada = analogRead(A0);
  tensao = 5.0 * entrada / 1023.0;

  if( entrada != entAnt ) {
    Serial.print( entrada );
    Serial.print( ": ");
    Serial.print(tensao,2);
    Serial.println("V");
    entAnt = entrada;
  }

  delay(100);

}
```

Medição analógica típica



O que percebemos?

- A exatidão é razoável;
- As tensões foram representadas com 2 dígitos decimais;
- Percebe-se alguma instabilidade nos valores medidos, exceto para 0 e 5V.

Cuidado importante !

- Não há qualquer proteção da entrada, logo tensões negativas, ou superiores a 5V podem danificar o Arduíno !

Casas Decimais

O número de casas decimais não é só uma “escolha”; ele depende diretamente da resolução;

Outro parâmetro também importante é o “número de contagens”;

Vamos ver estes parâmetros ?

Aula 07

Precisão e Exatidão

A informação é instável e imprecisa

- Nota-se também um erro de medição, mesmo que façamos uma média;
- Vamos entender cada um dos parâmetros?

Pouco preciso,
pouco exato

Precisão e Exatidão

Muito preciso, pouco exato

Pouco preciso, pouco exato

Precisão não tem relação com a obtenção do valor exato

Se o valor medido for fixo, medições precisas retornarão valores sempre muito próximos;

Ou seja, para valores fixos, a medição será estável, mesmo que errada.

Precisão e Exatidão

Exatidão não tem relação com precisão (ou estabilidade)

Muitas vezes medições exatas perdem precisão devido à presença de erros aleatórios (ruído branco, ou aleatório);

Nestes casos, a média dos valores medidos é muito próxima do valor real.

Pouco preciso, pouco exato

Pouco preciso, muito exato

Precisão e Exatidão

Exatidão não tem relação com precisão (ou estabilidade)

Muitas vezes medições exatas perdem precisão devido à presença de erros aleatórios (ruído branco, ou aleatório);

Nestes casos, a média dos valores medidos é muito próxima do valor real.

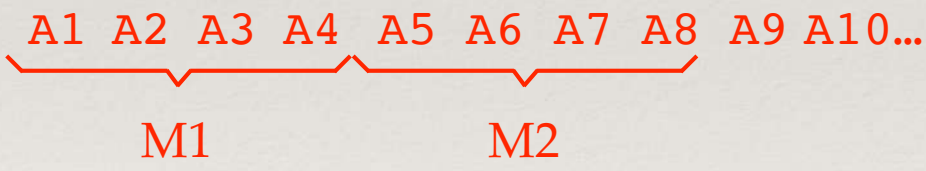
Pouco preciso,
pouco exato

Muito preciso
muito exato

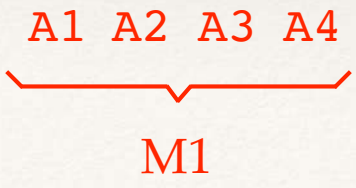
Precisão

Se a causa for um ruído com características aleatórias

- Cálculo da média pode resolver;
- Solução Típica: média simples;
- Número de amostragens depende do desvio padrão



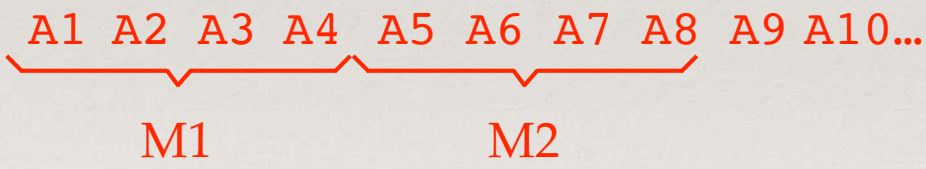
- Médias móveis normalmente são melhores devido à performance



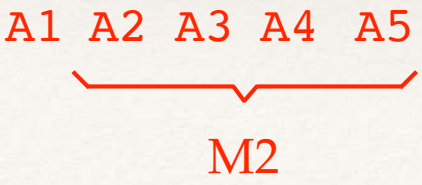
Precisão

Se a causa for um ruído com características aleatórias

- Cálculo da média pode resolver;
- Solução Típica: média simples;
- Número de amostragens depende do desvio padrão



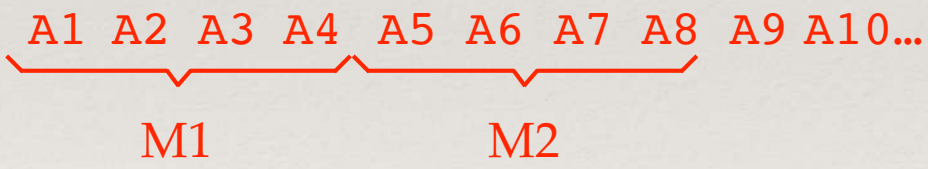
- Médias móveis normalmente são melhores devido à performance



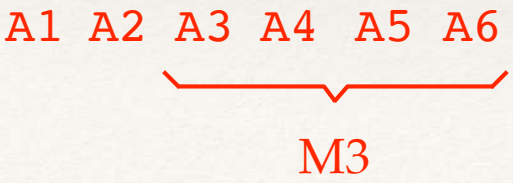
Precisão

Se a causa for um ruído com características aleatórias

- Cálculo da média pode resolver;
- Solução Típica: média simples;
- Número de amostragens depende do desvio padrão



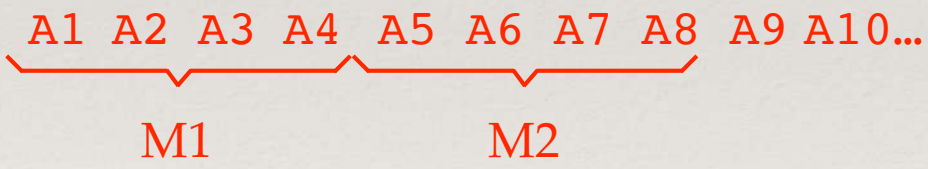
- Médias móveis normalmente são melhores devido à performance



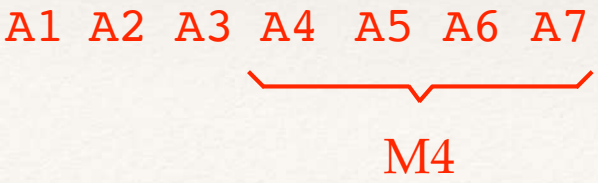
Precisão

Se a causa for um ruído com características aleatórias

- Cálculo da média pode resolver;
- Solução Típica: média simples;
- Número de amostragens depende do desvio padrão



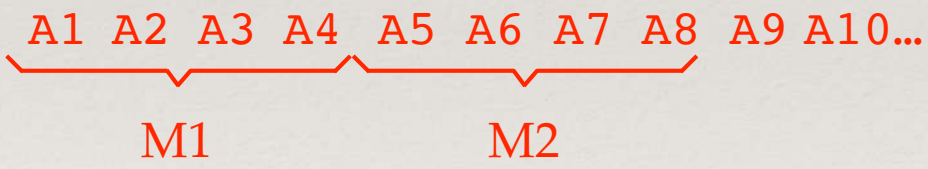
- Médias móveis normalmente são melhores devido à performance



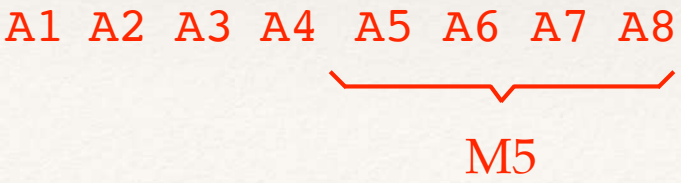
Precisão

Se a causa for um ruído com características aleatórias

- Cálculo da média pode resolver;
- Solução Típica: média simples;
- Número de amostragens depende do desvio padrão



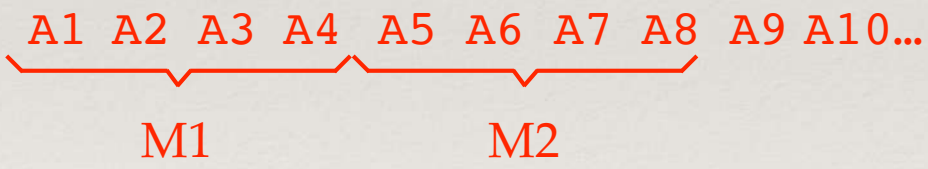
- Médias móveis normalmente são melhores devido à performance



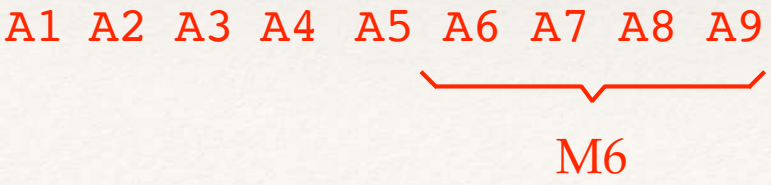
Precisão

Se a causa for um ruído com características aleatórias

- Cálculo da média pode resolver;
- Solução Típica: média simples;
- Número de amostragens depende do desvio padrão



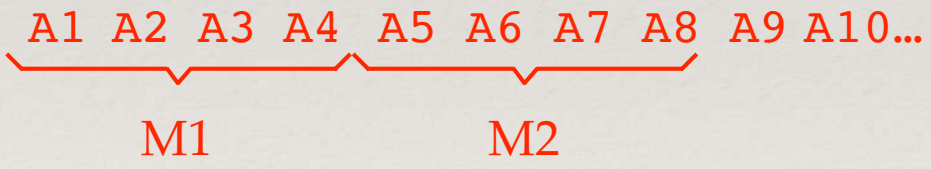
- Médias móveis normalmente são melhores devido à performance



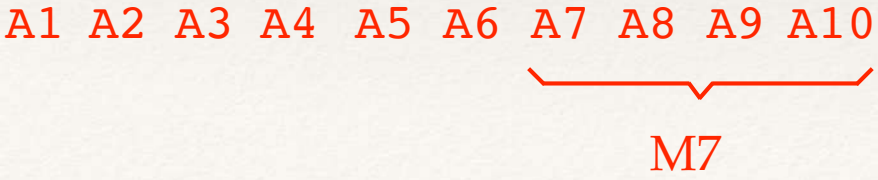
Precisão

Se a causa for um ruído com características aleatórias

- Cálculo da média pode resolver;
- Solução Típica: média simples;
- Número de amostragens depende do desvio padrão



- Médias móveis normalmente são melhores devido à performance



Exatidão

Normalmente a inexatidão é um problema mais grave

- A informação correta não está disponível;
- Se houver precisão, no entanto, a comparação com outra medição mais exata pode determinar um padrão para correção do erro;

A exatidão é um percentual

- Ex.: **± (2% + 2)** significa 2% de exatidão + 2. Se estiver lendo 10,0, o valor pode estar entre 9,78 e 10,22.

Desafio: Média Móvel

Como aumentar a precisão no experimento?

Vamos implementar o algoritmo de Média Móvel?

Média Móvel

Guarda medição no vetor

Incrementa o vetor de forma circular 0 ⟳ 16

Acumula as 16 últimas medições

Calcula a média dos valores acumulados

```
void setup() {

  Serial.begin(115200);
  pinMode(A0,INPUT);

}

int entrada, entradaAnterior;
int posicao = 0, i, medicoes[16], soma;
float tensao;

void loop() {

  entrada = analogRead(A0);

  medicoes[ posicao ] = entrada;
  if( posicao < 16 ) posicao++;
  else posicao = 0;

  soma = 0;
  for( i=0 ; i < 16 ; i++ )
    soma += medicoes[ i ];

  tensao = (soma / 16 ) * 5.0 / 1023.0;

  if( entrada != entradaAnterior ) {
    Serial.print( entrada );
    Serial.println( ": " );
    Serial.print( tensao,2 );
    Serial.println( "V" );
    entradaAnterior = entrada;
  }

  delay(100);

}
```

Aula 07

Linearidade

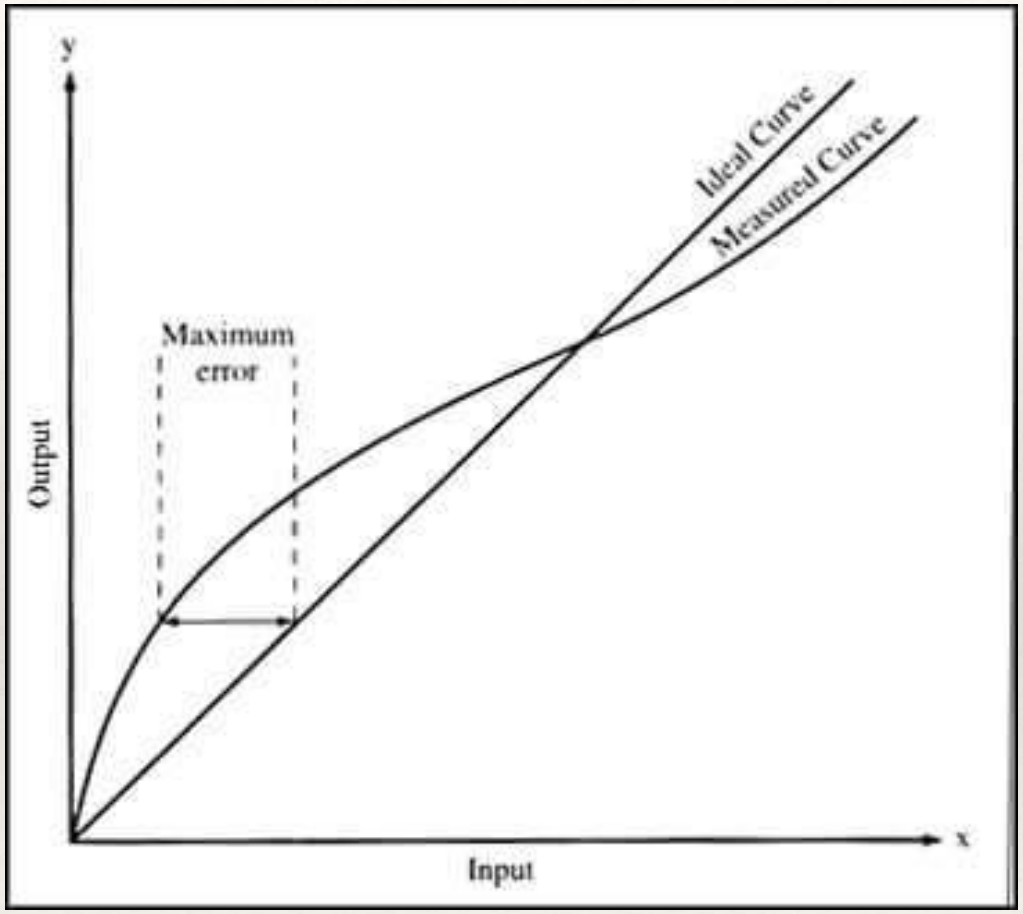
O comportamento da leitura nem sempre corresponde ao comportamento real do valor medido

A não-linearidade provoca erros de medição em parte dos pontos avaliados;

É comum encontrar a não-linearidade nos extremos da faixa de valores mensuráveis;

Em algumas situações, pode valer a pena limitar a faixa de valores medidos, e com isso a resolução;

Em casos específicos, pode-se corrigir a não-linearidade matematicamente.



Precisão, Exatidão e Linearidade

Nenhum destes parâmetros tem relação obrigatória com medições digitais !

Qualquer tipo de medida, inclusive com instrumentos analógicos ou manuais, é afetada pela precisão, exatidão e linearidade;

No entanto, medições digitais em sistemas automatizados também precisam ser avaliadas nestes aspectos.

Fatores Universais

- Referência
- Ruídos
- Escala

Fatores em Medições Digitais

- Resolução
- Taxa de Amostragem

Referência

Qualquer medição baseia-se em uma referência;

Microcontroladores e sensores podem usar tipicamente uma das seguintes fontes de referência:

- Tensão de Alimentação;
- Tensão de Referência Interna;
- Tensão de Referência Externa.

Referências possuem exatidão e precisão pré-definidas

- Quanto melhor a referência, melhor a medição (e maior o custo);
- Fatores externos, como a temperatura, muitas vezes afetam as referências.

Fontes de Referência

Diodo Zener

- Tensões diversas;
- Tomar cuidado com a linearidade;
- Afetado pela temperatura (700ppm/^{o}C)

Fontes de Referência

Referências Programáveis de Precisão

Ex.: TL431 (Texas)

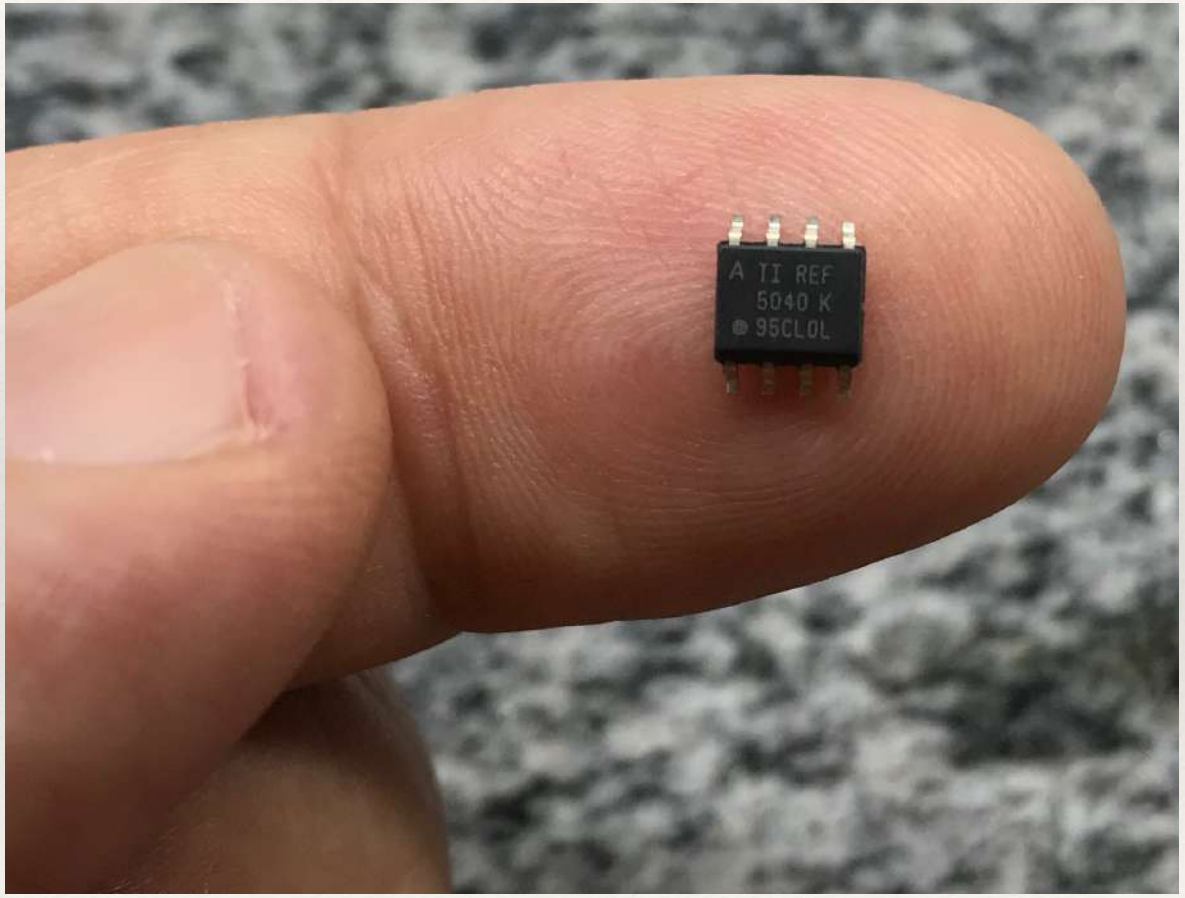
- Tensão ajustável de 2,5 a 36V;
- 0,5% de Exatidão;
- Afetado pela temperatura (600ppm/^{o}C).

Fontes de Referência

Chips especializados

Ex.: 5040AIDG4 (Texas)

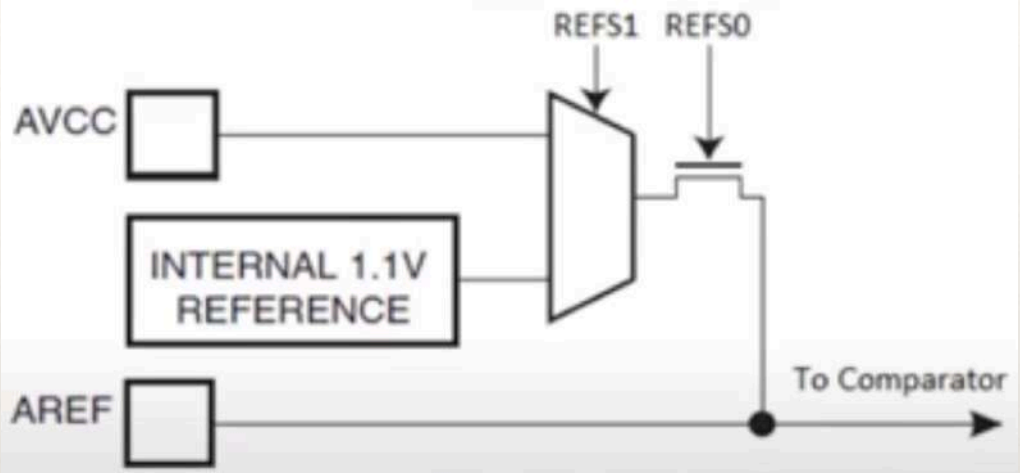
- 4,096 Volts
- 0,05% de exatidão;
- 3 ppm/^{o}C



Ex: Referência no Arduíno

Por padrão, o Arduíno utiliza a tensão de alimentação (5V) como referência;

- Outras referências exigem *software* e até *hardware* específico;
- Após mudança, as primeiras medições podem ser imprecisas.



analogReference(DEFAULT) : utiliza Vcc, e varia com ela.

analogReference(INTERNAL) : utiliza referência interna de 1,1V

- Medir valor efetivo (medir pino AREF durante operação normal).

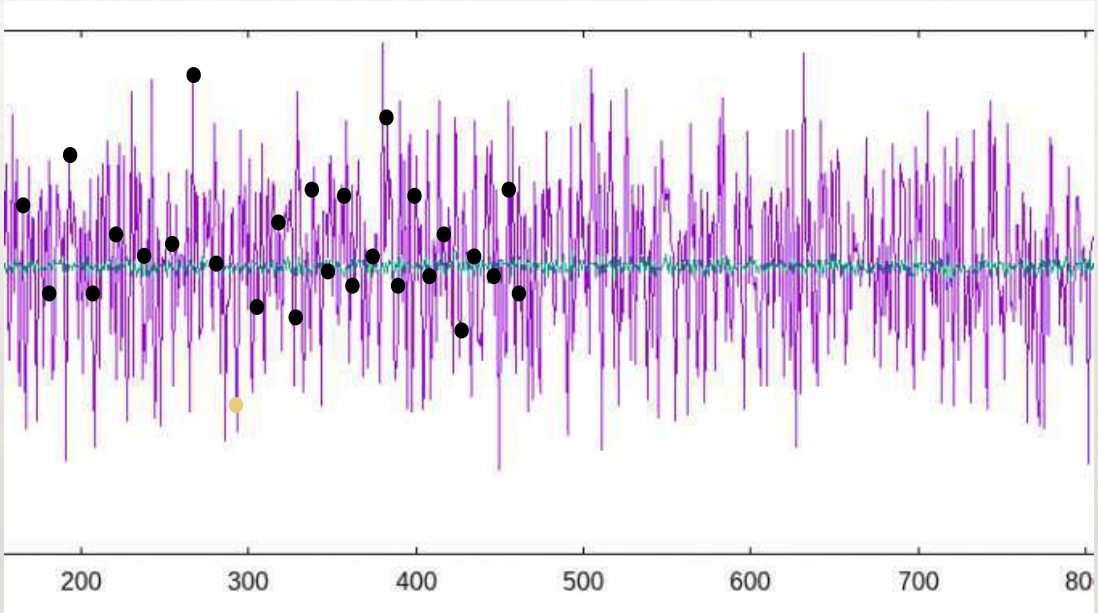
analogReference(EXTERNAL) : utiliza tensão de entrada no pino AREF

- 0V < Referência <= 5V (a qualidade da referência determinará a exatidão das medidas);
- Não usar analogRead() antes de mudar a referência, pois isso pode danificar o Arduíno.

Tratando o ruído

A melhor opção é buscar a redução do ruído no sensor, circuito ou meio ambiente

- Qualidade do sensor e componentes;
- Proximidade de Fontes de ruído;
- Conexão e cabos;
- Alimentação elétrica;



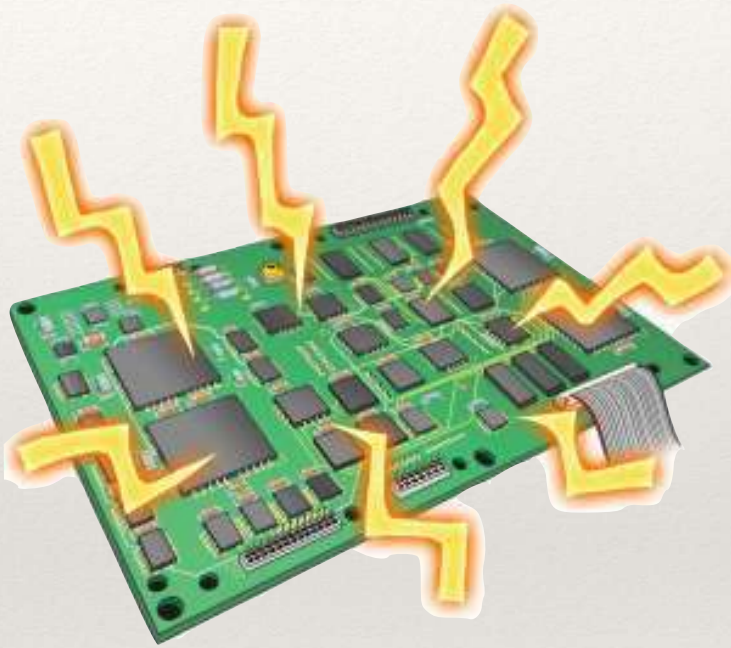
Ruídos tipicamente afetam a precisão

- Quando aleatórios, ruídos podem ser reduzidos por algoritmos de média;
- Se o desvio padrão for elevado, o número de amostragens necessárias pode ser grande.

Ruído causa problemas graves

Fontes de ruído de alta intensidade podem provocar problemas ainda mais sérios

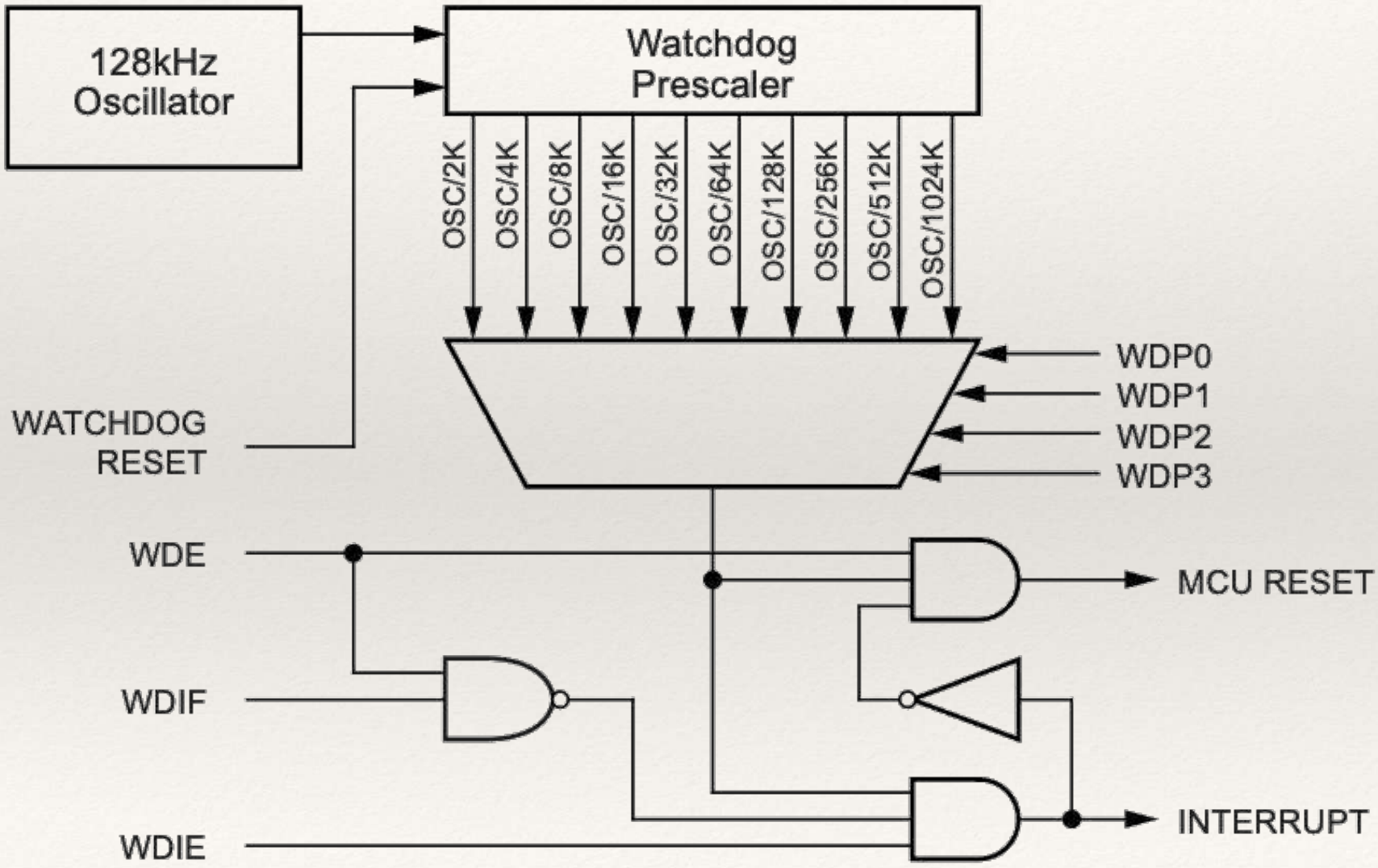
- Comum em indústrias, e outra aplicações, como sistemas automotivos embarcados;
- Ruídos podem impedir a operação de ADCs;
- Ruídos podem alterar registros internos do MCU;
- Ruídos podem provocar “travamento".



Design e projeto eletrônico podem reduzir impactos

- Blindagem e aterramento;
- Layout da PCB e outras questões;
- Como evitar o travamento em aplicações embarcadas de automação?

WDT (*Watch Dog Timer*)



Datasheet do ATMega328P

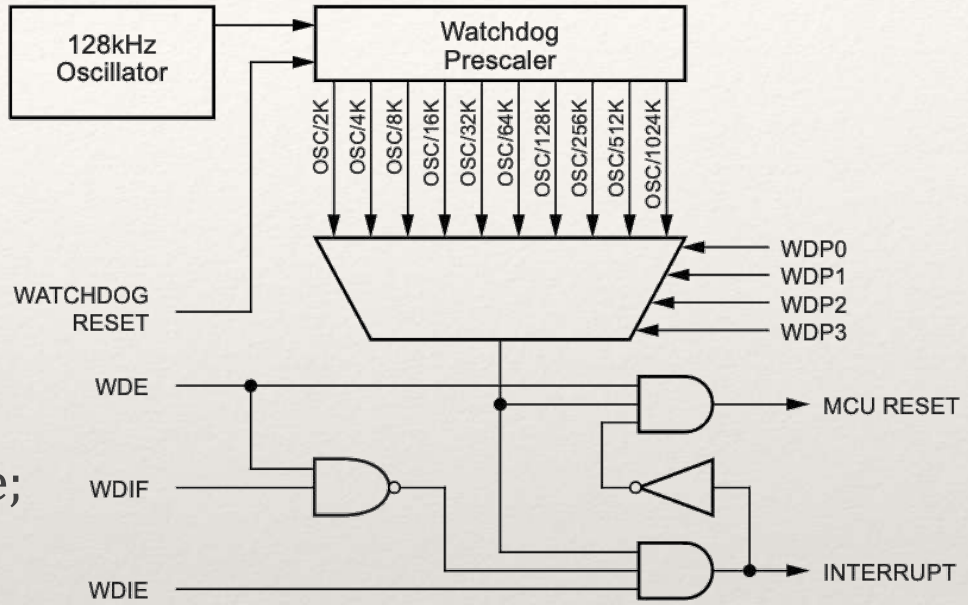
WDT (*Watch Dog Timer*)

MCUs sujeitas a "travamento"

- Espera por dispositivo que não responde;
- Endereço da instrução corrompido, etc.

Operação básica do WDT:

- Contador baseado em *clock* independente;
- Ao atingir contagem determinada, envia sinal de RESET ao MCU;



- MCU evita o RESET enviando ele mesmo um sinal de RESET para o WDT;
- Enquanto o MCU enviar periodicamente o sinal, tudo funciona bem;
- Diante da eventual falha no envio do sinal (por um travamento), o WDT reinicializa o MCU, permitindo que tudo volte à operação.

Escalas

Adequam a leitura à faixa de valores das entradas

- Devem ser dimensionadas para manter a resolução máxima;
- Tipicamente usam Divisores de Tensão;
- Avaliar impedâncias;
- Cuidado com a qualidade dos componentes.

Outros "detalhes"

- Unificar o "zero volts";
- Medições diferenciais;
- Algumas escalas não começam em zero;
- Medição de valores negativos;
- Proteção de entradas.

Escalas

Adequam a leitura à faixa de valores das entradas

- Devem ser dimensionadas para manter a resolução máxima;
- Tipicamente usam Divisores de Tensão;
- Avaliar impedâncias;
- Cuidado com a qualidade dos componentes.

Outros "detalhes"

- Unificar o "zero volts";
- Medições diferenciais;
- Algumas escalas não começam em zero;
- Medição de valores negativos;
- Proteção de entradas.

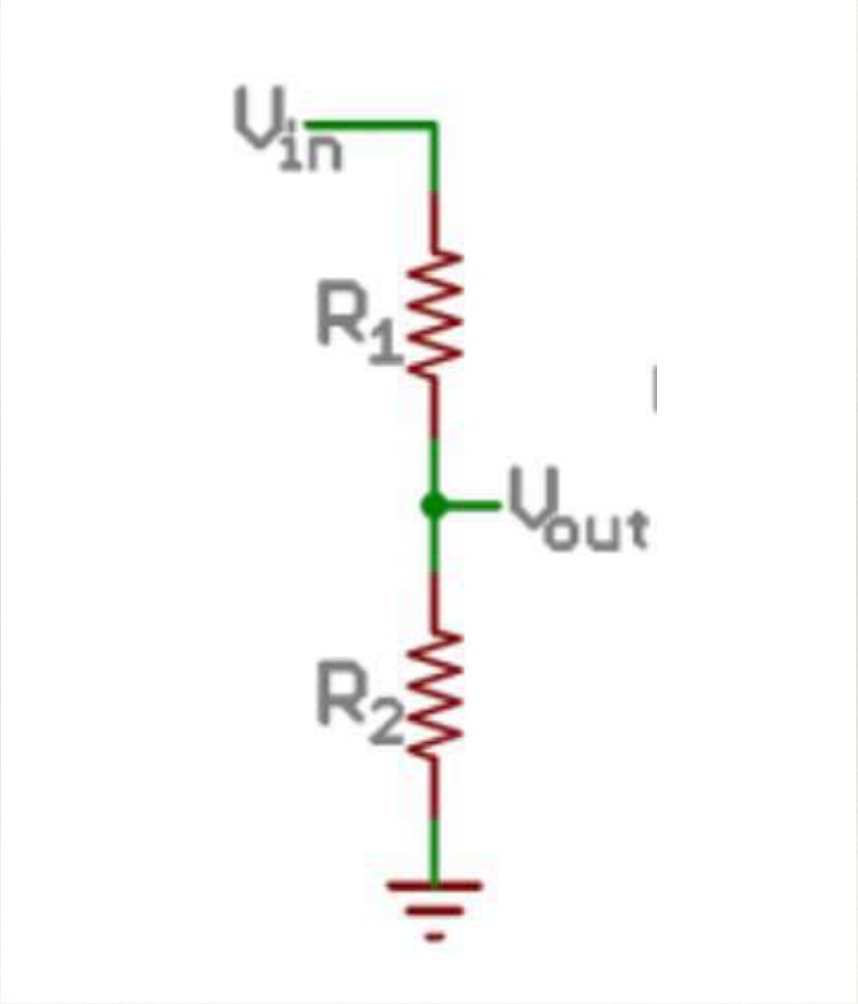
Escalas

Adequam a leitura à faixa de valores das entradas

- Devem ser dimensionadas para manter a resolução máxima;
- Tipicamente usam Divisores de Tensão;
- Avaliar impedâncias;
- Cuidado com a qualidade dos componentes.

Outros "detalhes"

- Unificar o "zero volts";
- Medições diferenciais;
- Algumas escalas não começam em zero;
- Medição de valores negativos;
- Proteção de entradas.



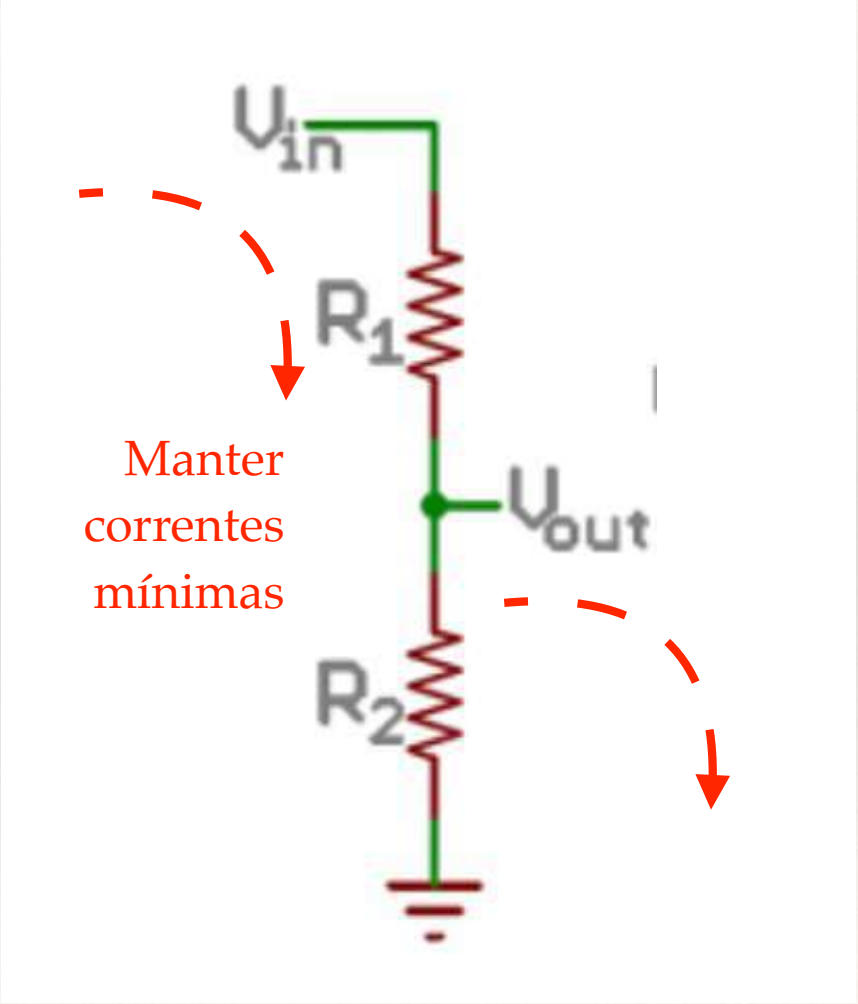
Escalas

Adequam a leitura à faixa de valores das entradas

- Devem ser dimensionadas para manter a resolução máxima;
- Tipicamente usam Divisores de Tensão;
- Avaliar impedâncias;
- Cuidado com a qualidade dos componentes.

Outros "detalhes"

- Unificar o "zero volts";
- Medições diferenciais;
- Algumas escalas não começam em zero;
- Medição de valores negativos;
- Proteção de entradas.



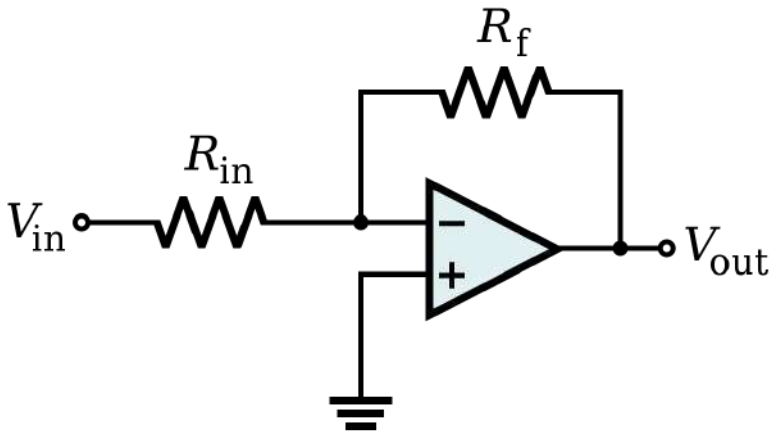
Escalas

Adequam a leitura à faixa de valores das entradas

- Devem ser dimensionadas para manter a resolução máxima;
- Tipicamente usam Divisores de Tensão;
- Avaliar impedâncias;
- Cuidado com a qualidade dos componentes.

Outros "detalhes"

- Unificar o "zero volts";
- Medições diferenciais;
- Algumas escalas não começam em zero;
- Medição de valores negativos;
- Proteção de entradas.



Escalas

Adequam a leitura à faixa de valores das entradas

- Devem ser dimensionadas para manter a resolução máxima;
- Tipicamente usam Divisores de Tensão;
- Avaliar impedâncias;
- Cuidado com a qualidade dos componentes.

Outros "detalhes"

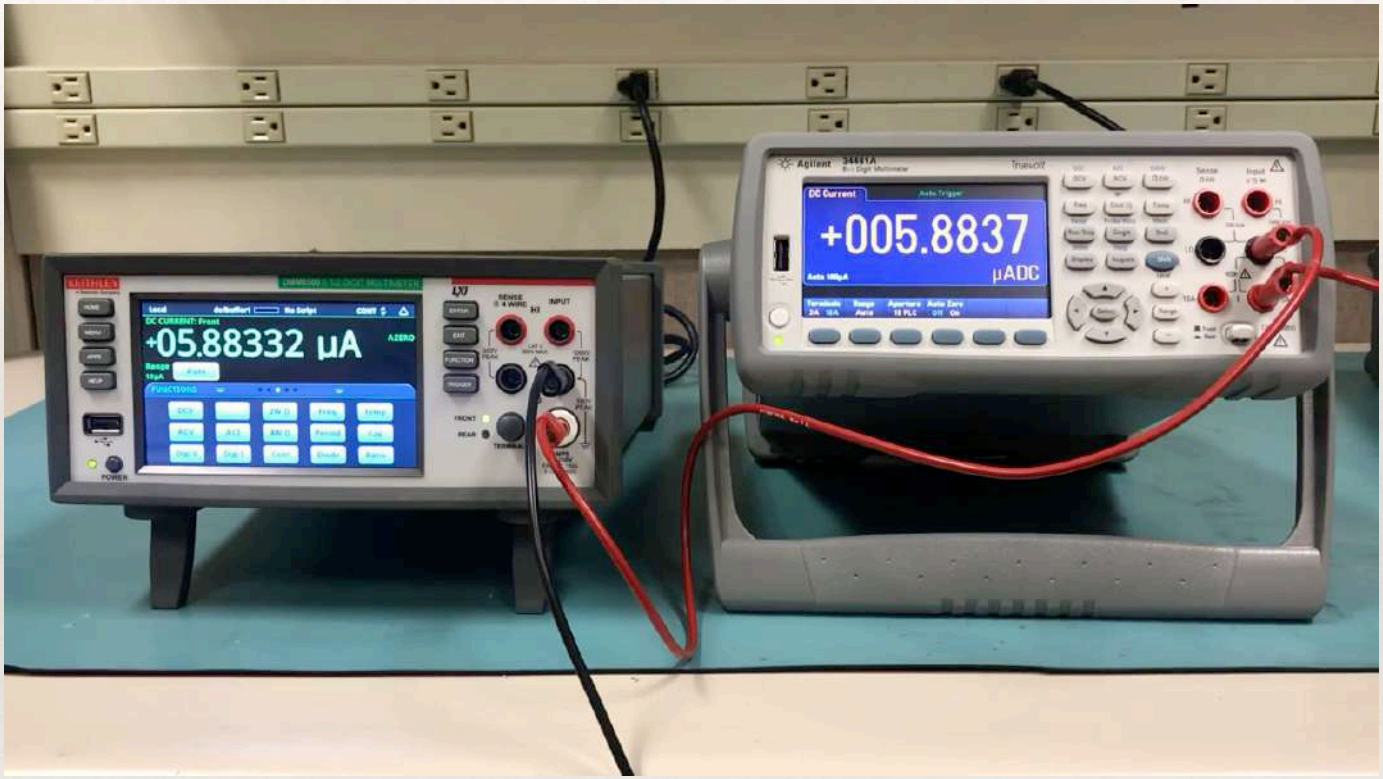
- Unificar o "zero volts";
- Medições diferenciais;
- Algumas escalas não começam em zero;
- Medição de valores negativos;
- Proteção de entradas.

Resolução

Parâmetro mais usado em medições digitais;

Medida em bits, é a capacidade de identificar valores diferentes, mesmo que próximos;

Outra forma de avaliar é a “contagem”: quantos diferentes valores podem ser representados ?



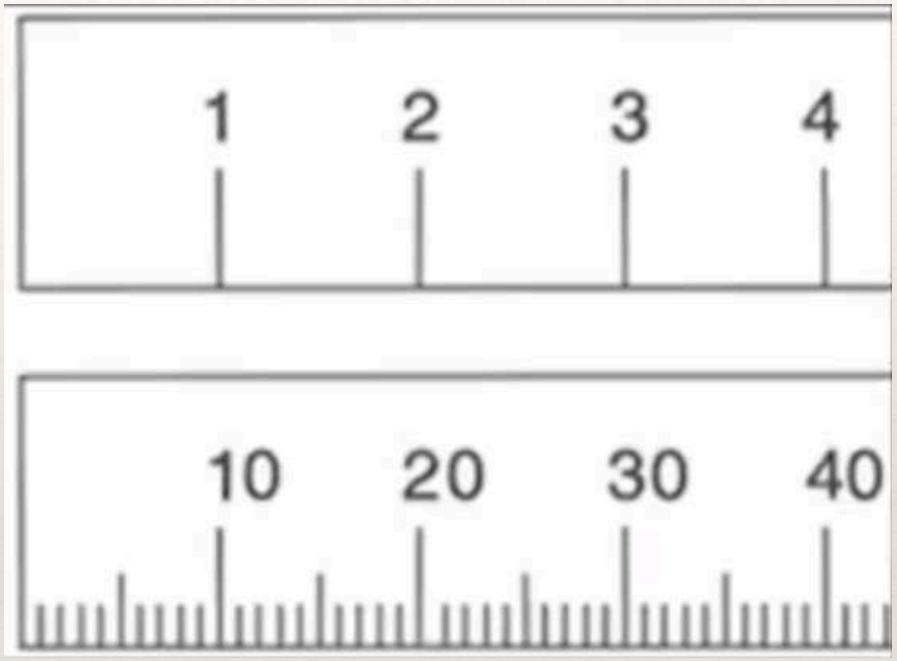
Quem determina a resolução é o ADC (Conversor Analógico / Digital);

Vídeo mostra a “disputa" de dois equipamentos de alta resolução. Notem que, mesmo possuindo a mesma quantidade de dígitos, há diferenças.

Resolução

Existe resolução em medidas analógicas ?

Em medições digitais, a resolução é definida em bits, já que o sinal de entrada precisa ser convertido para um número digital e binário;

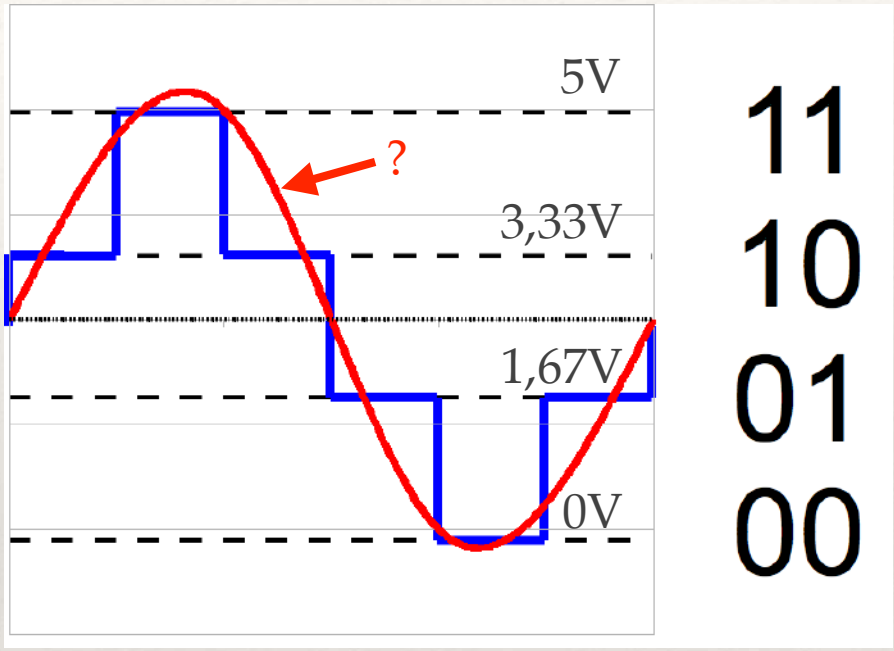


Resolução

Existe resolução em medidas analógicas ?

Em medições digitais, a resolução é definida em bits, já que o sinal de entrada precisa ser convertido para um número digital e binário;

Resolução de 2 bits: 4 valores diferentes;



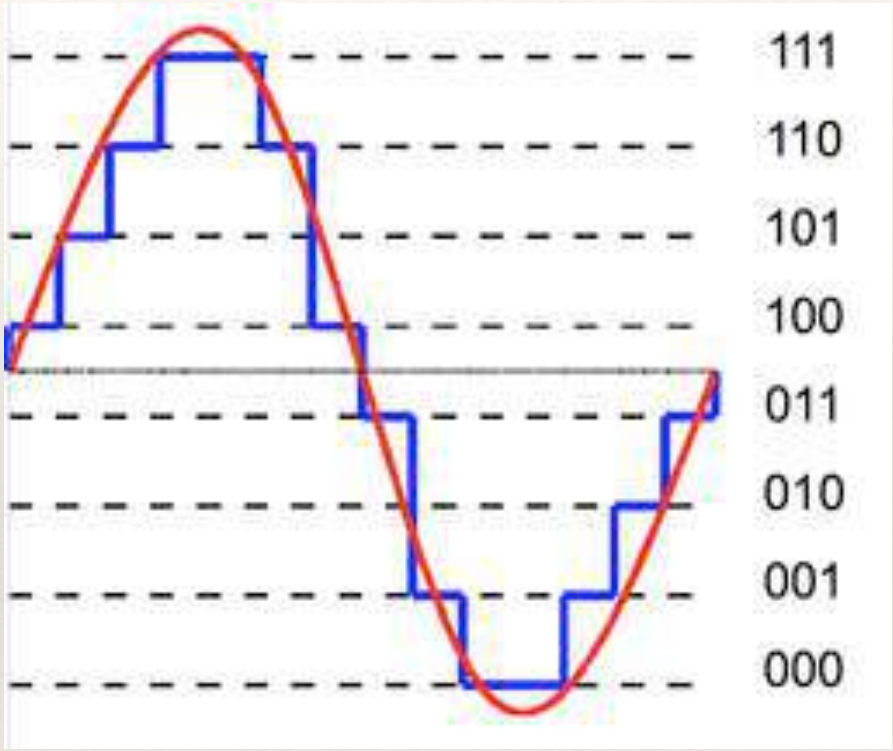
Resolução

Existe resolução em medidas analógicas ?

Em medições digitais, a resolução é definida em bits, já que o sinal de entrada precisa ser convertido para um número digital e binário;

Resolução de 2 bits: 4 valores diferentes;

Resolução de 3 bits: 8 valores diferentes;



Resolução

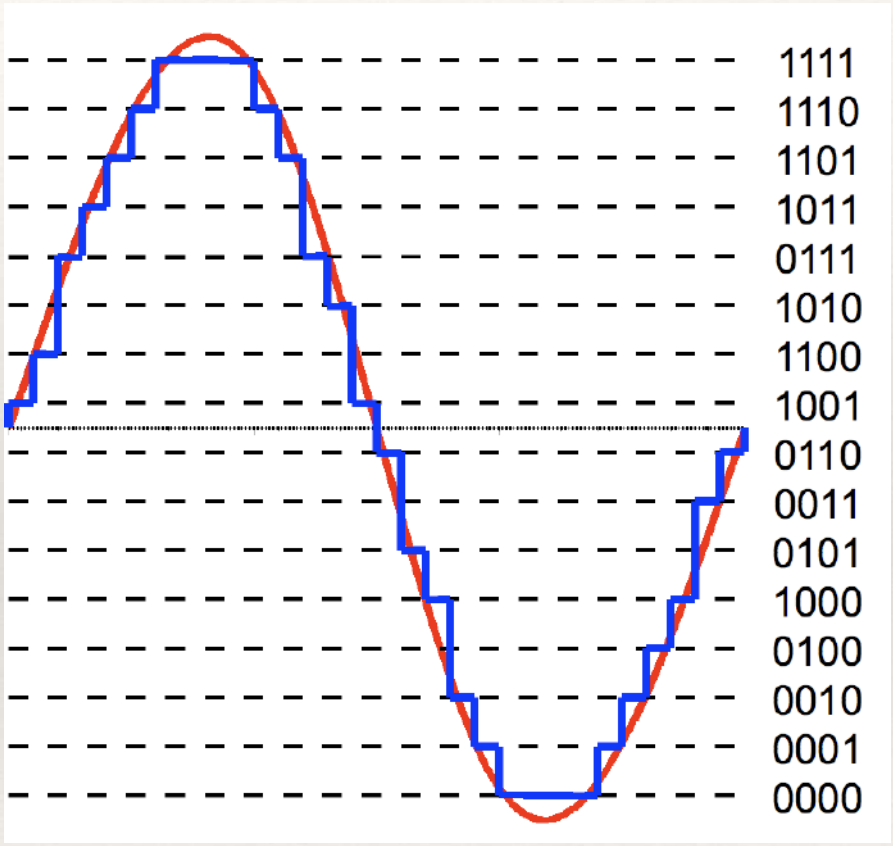
Existe resolução em medidas analógicas ?

Em medições digitais, a resolução é definida em bits, já que o sinal de entrada precisa ser convertido para um número digital e binário;

Resolução de 2 bits: 4 valores diferentes;

Resolução de 3 bits: 8 valores diferentes;

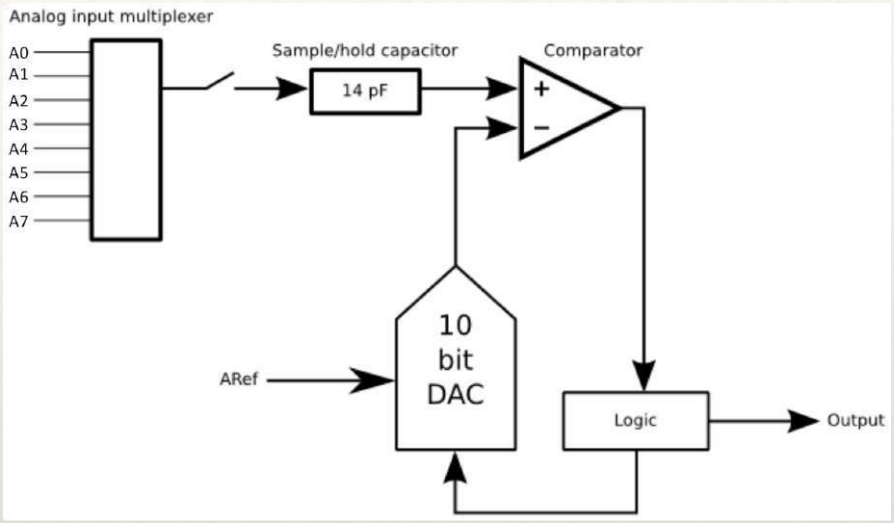
Resolução de 4 bits: 16 valores diferentes;



Resolução

Existe resolução em medidas analógicas ?

Em medições digitais, a resolução é definida em bits, já que o sinal de entrada precisa ser convertido para um número digital e binário;



Resolução de 2 bits: 4 valores diferentes;

Resolução de 3 bits: 8 valores diferentes;

Resolução de 4 bits: 16 valores diferentes;

Arduíno: 10 bits de resolução no ADC

São 4,89mV entre dois valores (5V / 1023). Isso implica em 3 dígitos (0,00 a 5,00V), pois o terceiro dígito (mV) não pode ser usado. Como são 500 valores, temos “500 contagens”;

Com a referência de 1,1V, teríamos 1mV. Isso implicaria em 4 dígitos, ou 1100 "contagens".

Resolução - como calcular

E se, dados os limites de uma medição, eu precisar calcular a resolução?

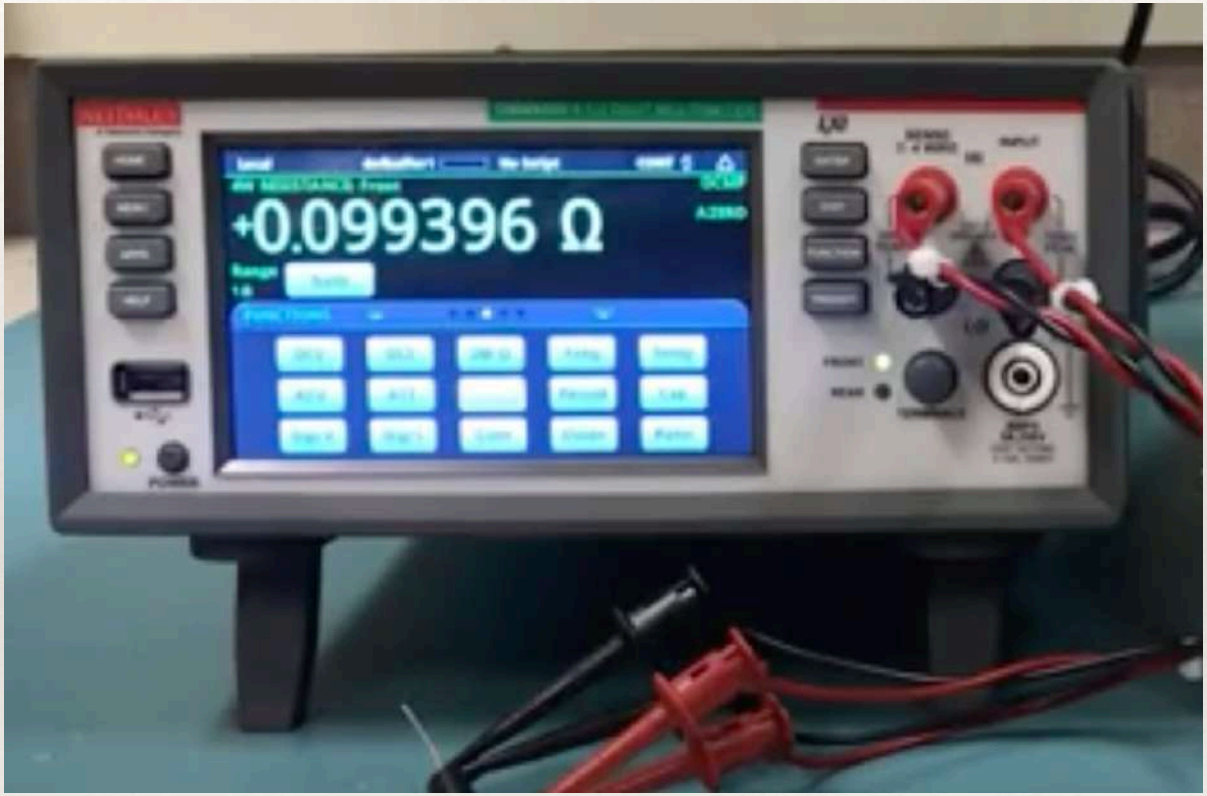
Qual a resolução necessária para termos “x" contagens?

No vídeo, os equipamentos possuem 6 1/2 dígitos, logo:

De 0 a 1.999.999: 2 milhões de valores;

2^{21} = 2.097.152 → são 21 bits;

Você não vai conseguir fazer com um Arduíno.



Resolução - como calcular

Outro exemplo: termômetro digital;

Medir temperaturas até 42,0^{0}C

De 0,0 até 42,0: 421 valores;

$2^9 = 512 \rightarrow$ são 9 bits;

Este dá para fazer com o Arduíno!

Aula 08

E/S Analógicas X Digitais

Grandezas Digitais e Analógicas precisam ser processadas constantemente;

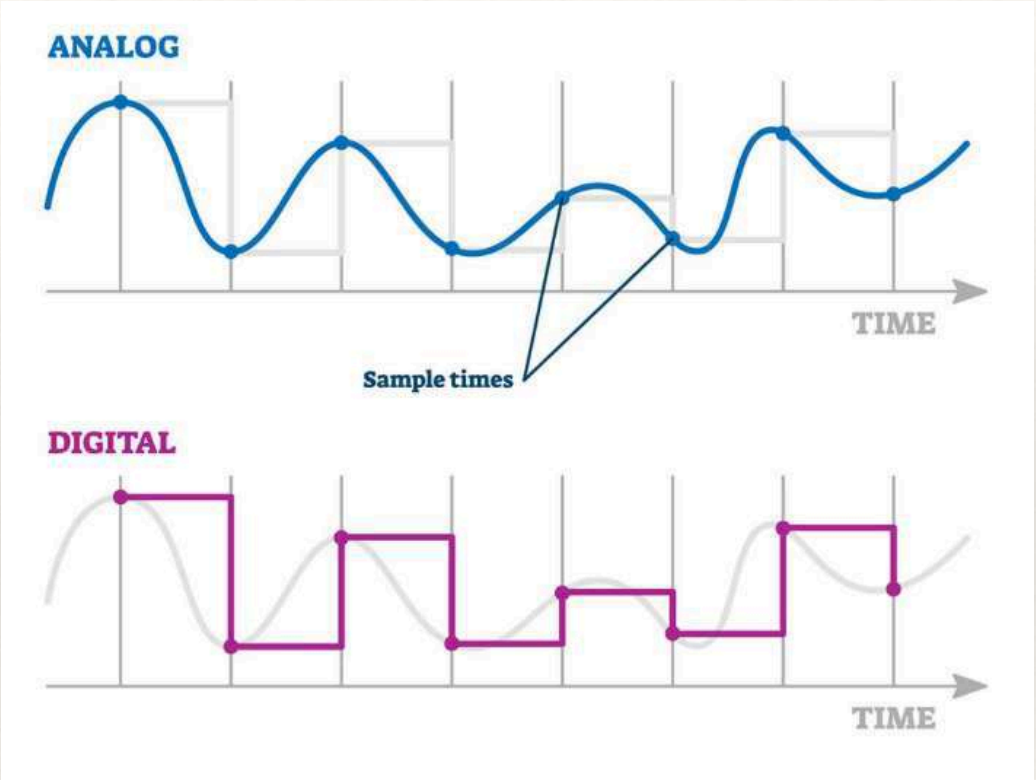
Para manipular informações analógicas, há dois desafios:

Leitura:

- ADCs (*Analog to Digital Converters*);

Escrita:

- DACs (*Digital to Analog Converters*);
- PWM (*Pulse Width Modulation*).



ADC: como funciona?

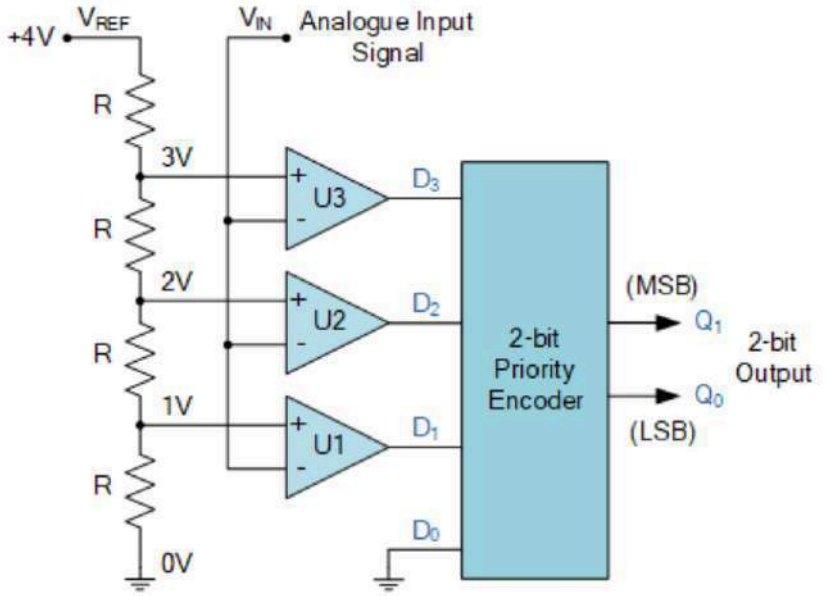
Arquitetura Paralela

- Um divisor resistivo divide a tensão de referência máxima em intervalos;
- As saídas do divisor resistivo são comparadas com o valor da entrada analógica através de “n" comparadores, gerando uma tabela verdade com “n" saídas;
- As “n" saídas são convertidas para o número binário correspondente.

A arquitetura é rápida (tempo real), mas sua complexidade é muito grande para resoluções maiores;

Outras arquiteturas são mais adequadas para resoluções mais altas

- Pipeline, SAR, Sigma-Delta
- Estas arquiteturas geram retardo, e são muito dependentes da linearidade, ruído, exatidão e precisão.



Analogue Input Voltage (V_{IN})	Comparator Outputs				Digital Outputs	
	D_3	D_2	D_1	D_0	Q_1	Q_0
0 to 1 V	0	0	0	**0**	0	0
1 to 2 V	0	0	**1**	X	0	1
2 to 3 V	0	**1**	X	X	1	0
3 to 4 V	**1**	X	X	X	1	1

ADC: como funciona?

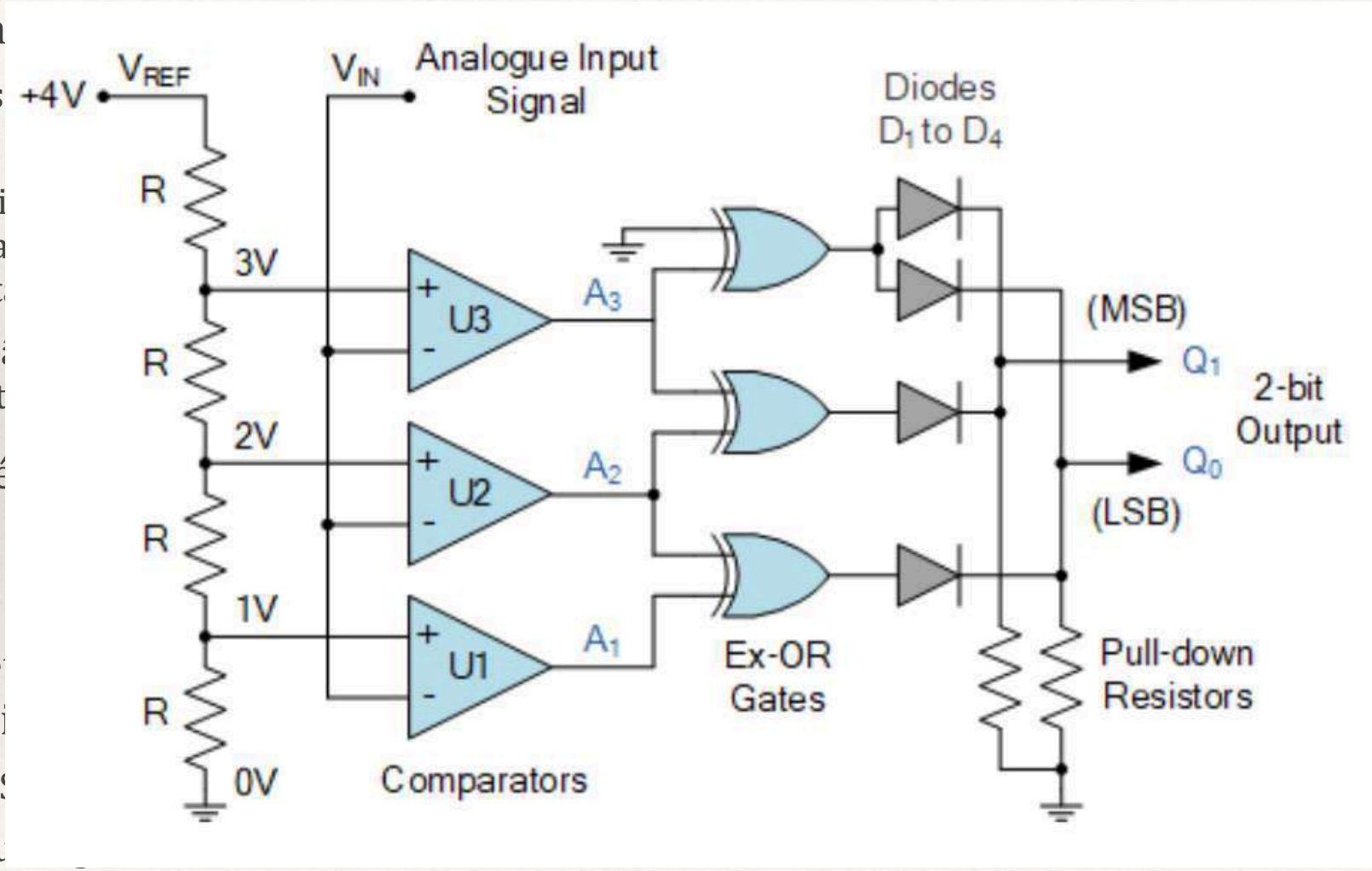
Arquitetura Pa

- Um divisor res
em intervalos;
- As saídas do di
da entrada ana
gerando uma t
- As “n" saídas s
correspondent

A arquitetura é
complexidade
maiores;

Outras arquite
resoluções mai

- Pipeline, SAR, S
- Estas arquitetu
dependentes da linearidade, ruído, exatidão e precisão.

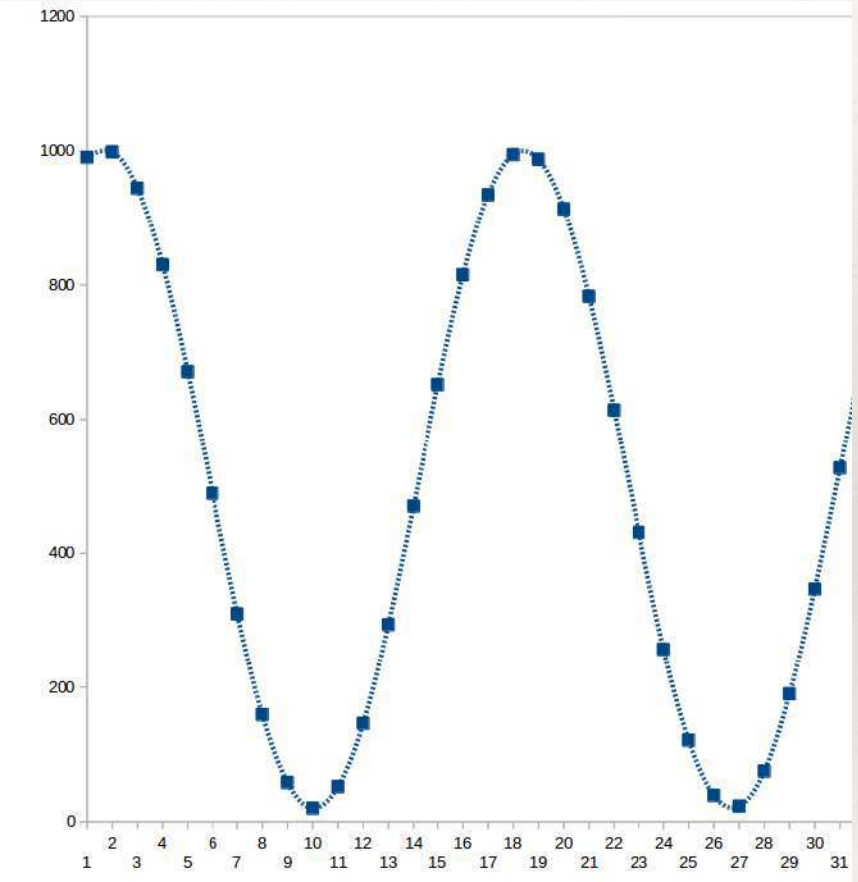


Amostragem

As medições são realizadas em um determinado momento, onde o valor medido é registrado;

A quantidade de medições em um determinado período é chamada de taxa de amostragem

- É necessária uma frequência de amostragem mínima para conseguirmos efetivamente representar um determinado sinal, e suas variações com o tempo;
- O teorema de Nyquist diz que, para uma amostragem infinita, uma frequência de amostragem pelo menos 2 vezes superior à maior componente de frequência do sinal é capaz de representar adequadamente o sinal original;
- Sinais digitais, devido às suas harmônicas, podem ter frequências de amostragem ideais muito elevadas.

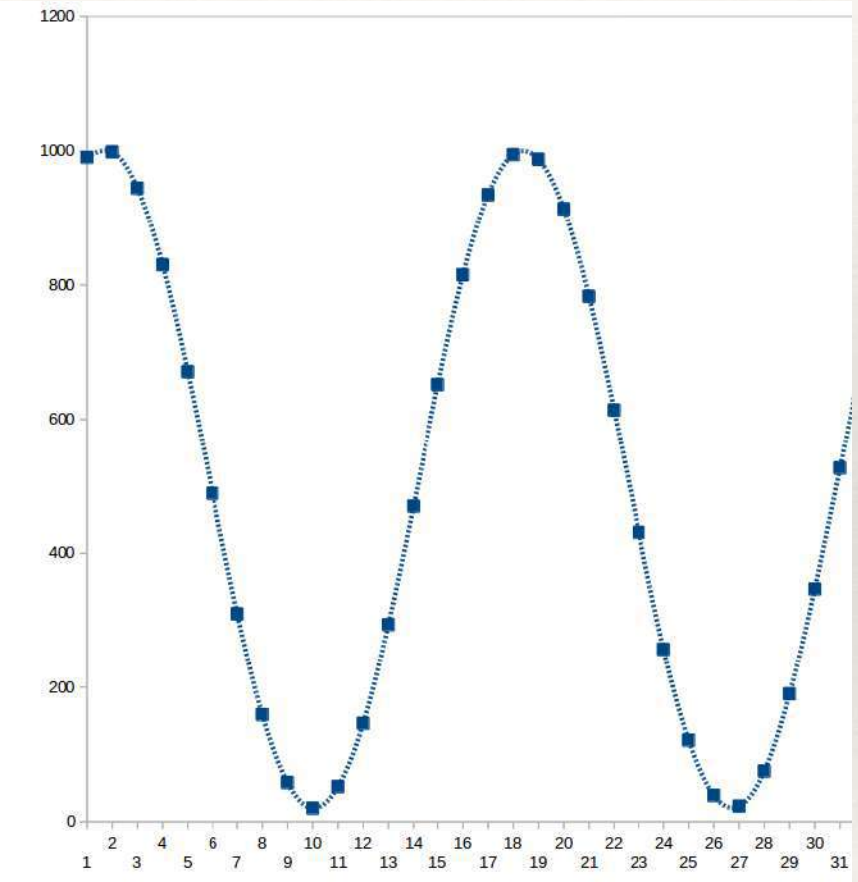


A frequência de amostragem está relacionada à performance do ADC.

Amostragem

A taxa de amostragem está associada à aplicação

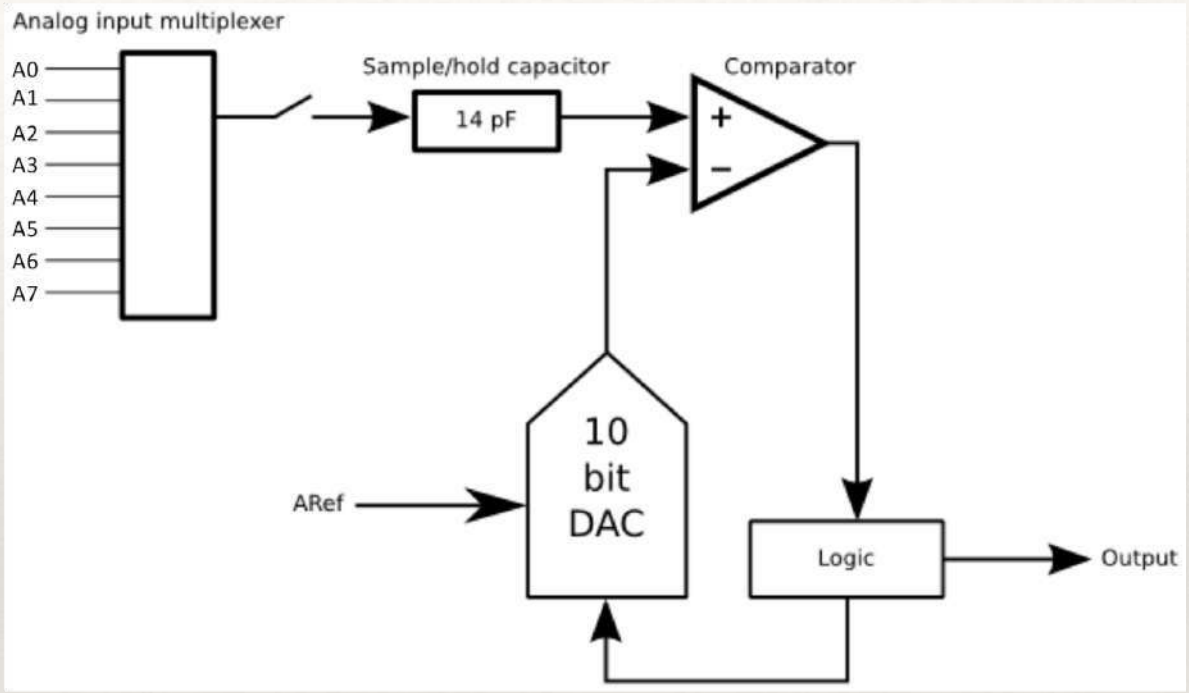
- Se você quer ler um valor fixo, ou com variação lenta, a taxa pode ser menor;
- Se você deseja analisar o comportamento de um sinal que varia rapidamente, a taxa precisa ser maior;
- Um voltímetro tem taxa típica de 4 SA/s (sem considerar cálculo de média);
- Um osciloscópio tem taxa típica de 1 GSA/s (você não vai conseguir fazer isto com um Arduíno).



O ADC do Arduíno

Hardware do ADC

- Uma amostragem é, na verdade, uma comparação;
- Capacitor para "filtro de ruído analógico";
- O filtro é mais eficiente para ruídos de frequência mais alta.



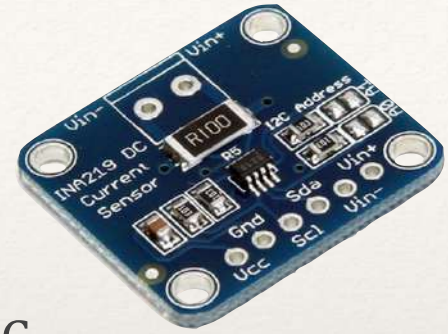
Multiplexação de 8 diferentes entradas analógicas

Poderíamos ter mais?

Tudo é uma questão de performance e aplicação.

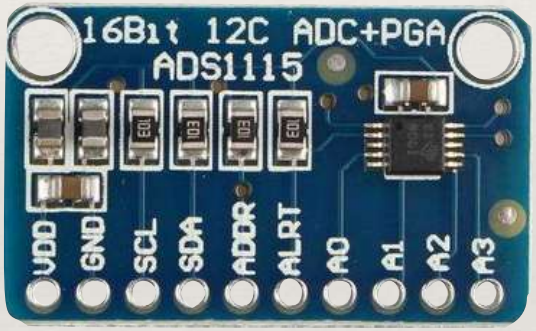
ADC externo

INA219 - 12 bits com interface i2C



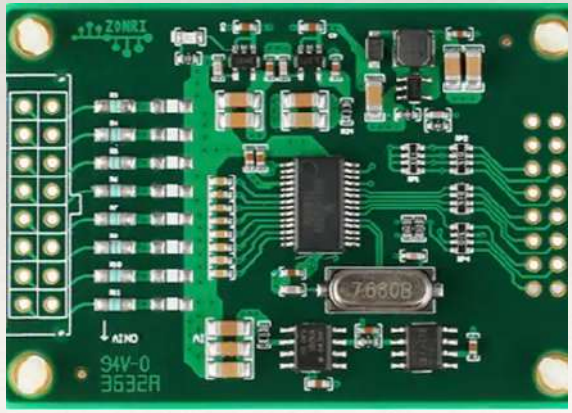
- Até 128 SA/s;
- Leitor de Tensão e Corrente.

ADS1115 - 16 bits com interface i2C



- Até 860 SA/s;
- 4 canais de entrada ou ...
- 2 canais diferenciais;
- Amplificador de ganho programável.

ADS1252 - 24 bits com interface SPI



DAC: como funciona?

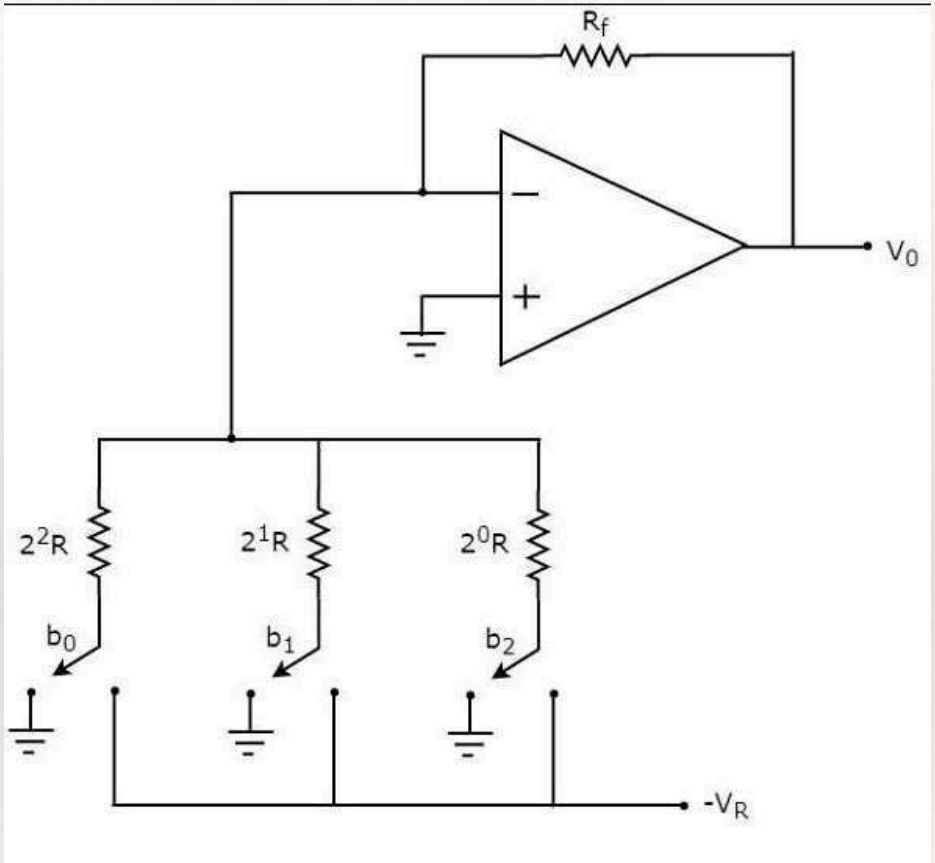
Um DAC converte uma entrada digital em uma saída de “n" níveis, simulando um valor analógico

- Quanto mais bits, “mais analógico";
- Propriedades: resolução, monotonicidade, linearidade, exatidão e tempo de acomodação.

Existem basicamente 2 tipos de DAC

- DAC por Resistores Ponderados;
- DAC por Rede R-2R em escada.

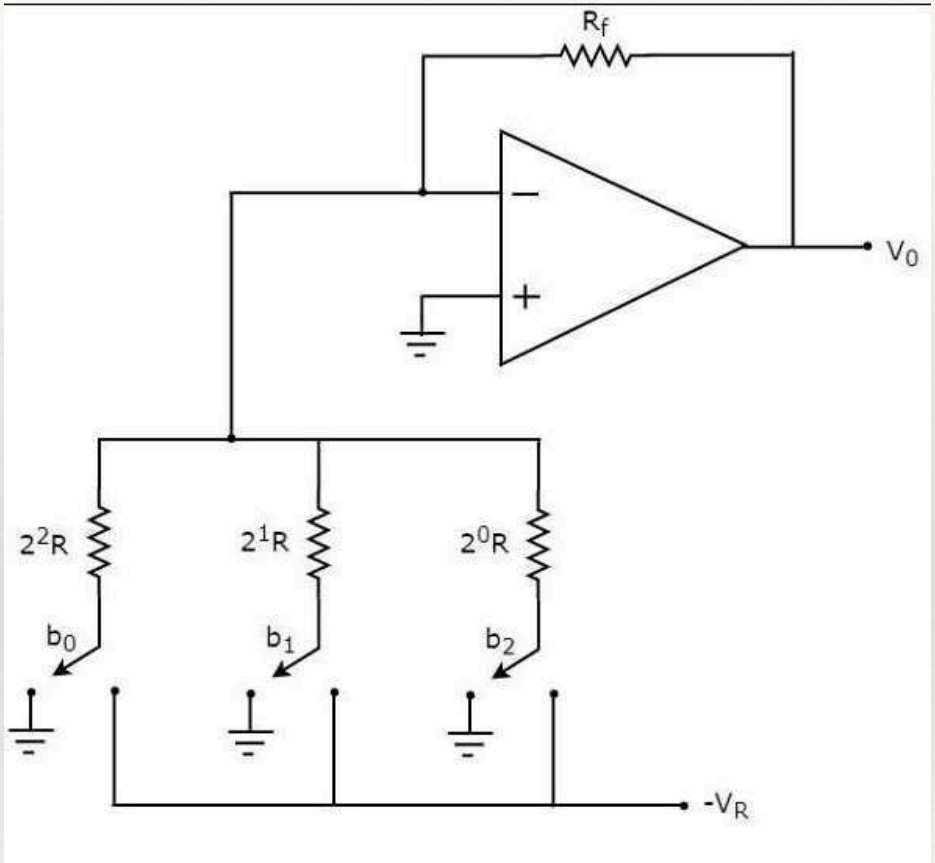
O Arduíno Uno <u>não possui</u> DAC integrado



DAC: como funciona?

DAC por Resistores Ponderados

- Sequência de resistores acompanham as potências de 2 representadas por cada dígito do número;
- Complexidade associada à precisão dos resistores, especialmente para muitos dígitos;
- Na figura temos um exemplo com entrada de 3 bits.

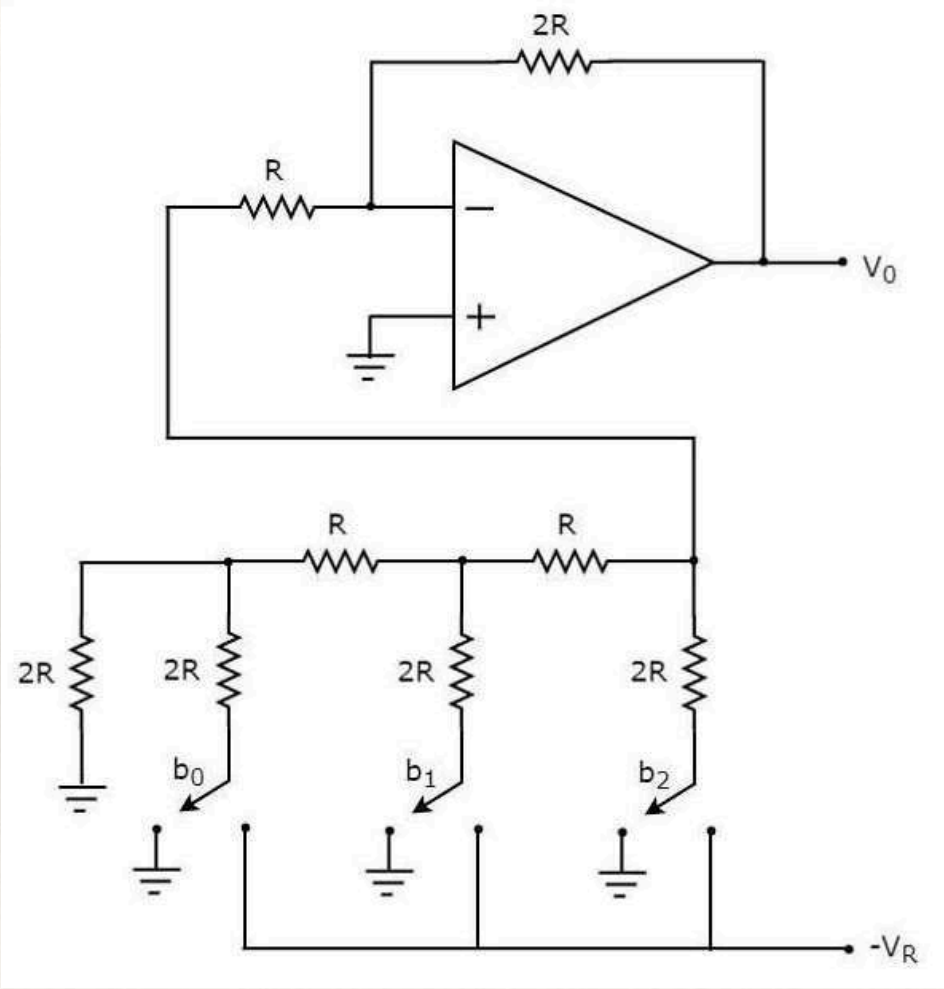


DAC: como funciona?

DAC por Rede R-2R em escada

A vantagem clara está nos dois únicos valores para os resistores, facilitando a construção;

Na figura temos um exemplo com entrada de 3 bits.



Parâmetros

Resolução: número de bits

Monotonicidade: crescimento constante

Linearidade

Exatidão

Tempo de acomodação: tempo necessário para a saída representar uma mudança entre o menor e maior valor binário na entrada.

PWM: como funciona?

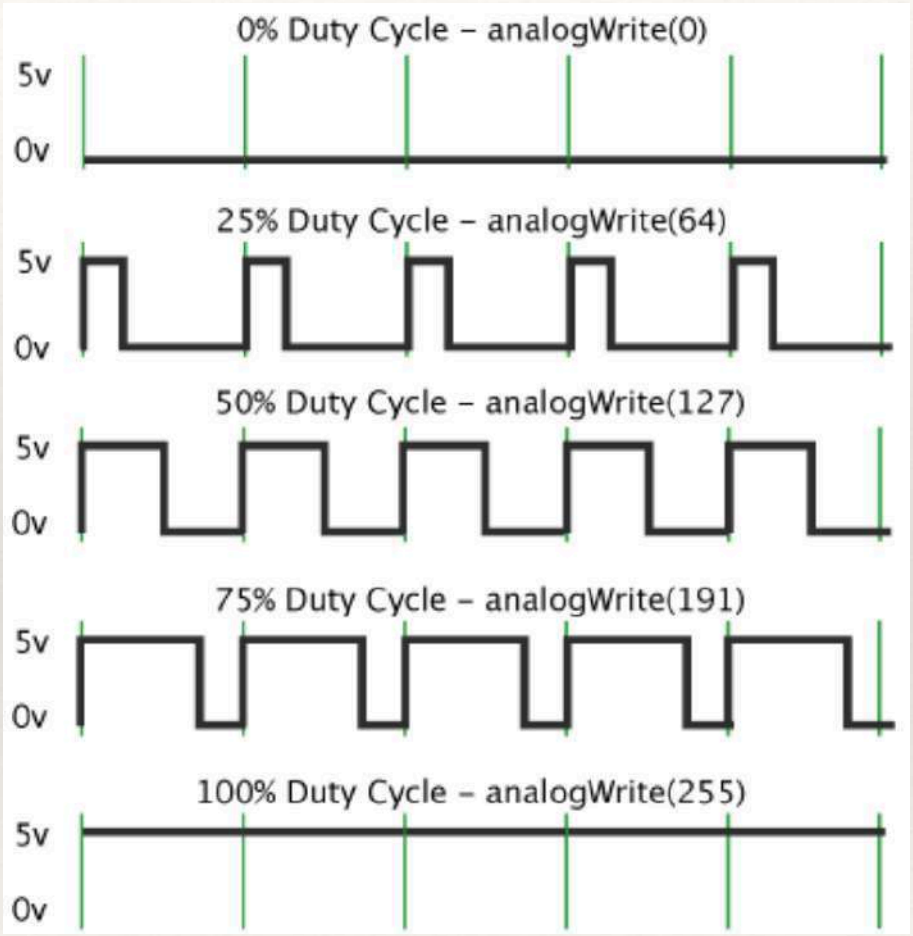
Pulse Width Modulation: Modulação por Largura de Pulso;

Opção mais comum para simular uma saída analógica

- Microcontrolador controla o percentual de tempo da saída no nível 1;
- O percentual é chamado de “ciclo de trabalho”, ou *Duty Cycle*;
- O efeito “analógico" vem da tensão média na saída.

Arduíno Uno

- Frequência típica 490Hz, mas pode subir até 4MHz;
- Resolução para ajuste do Duty Cycle: 8 bits

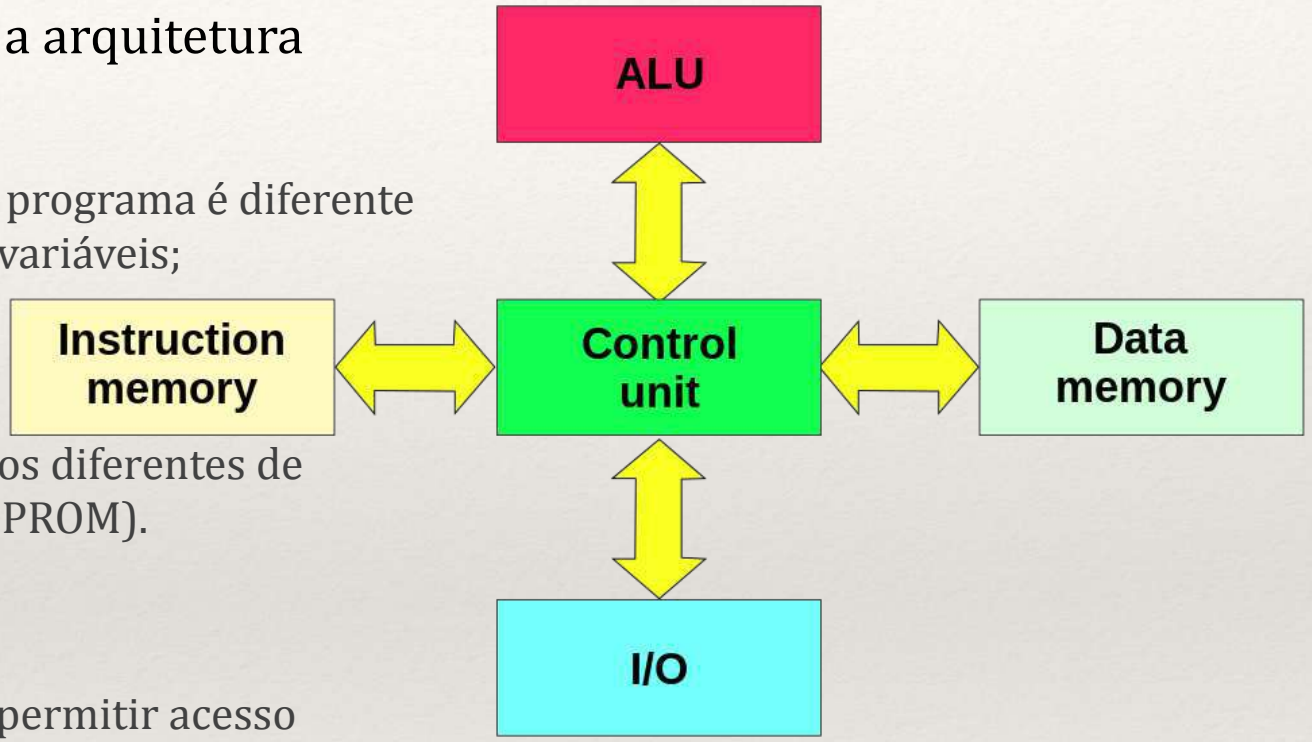


Aula 09

Memória no ATMega328P

O ATMega328P não segue a arquitetura padrão Von Neumann

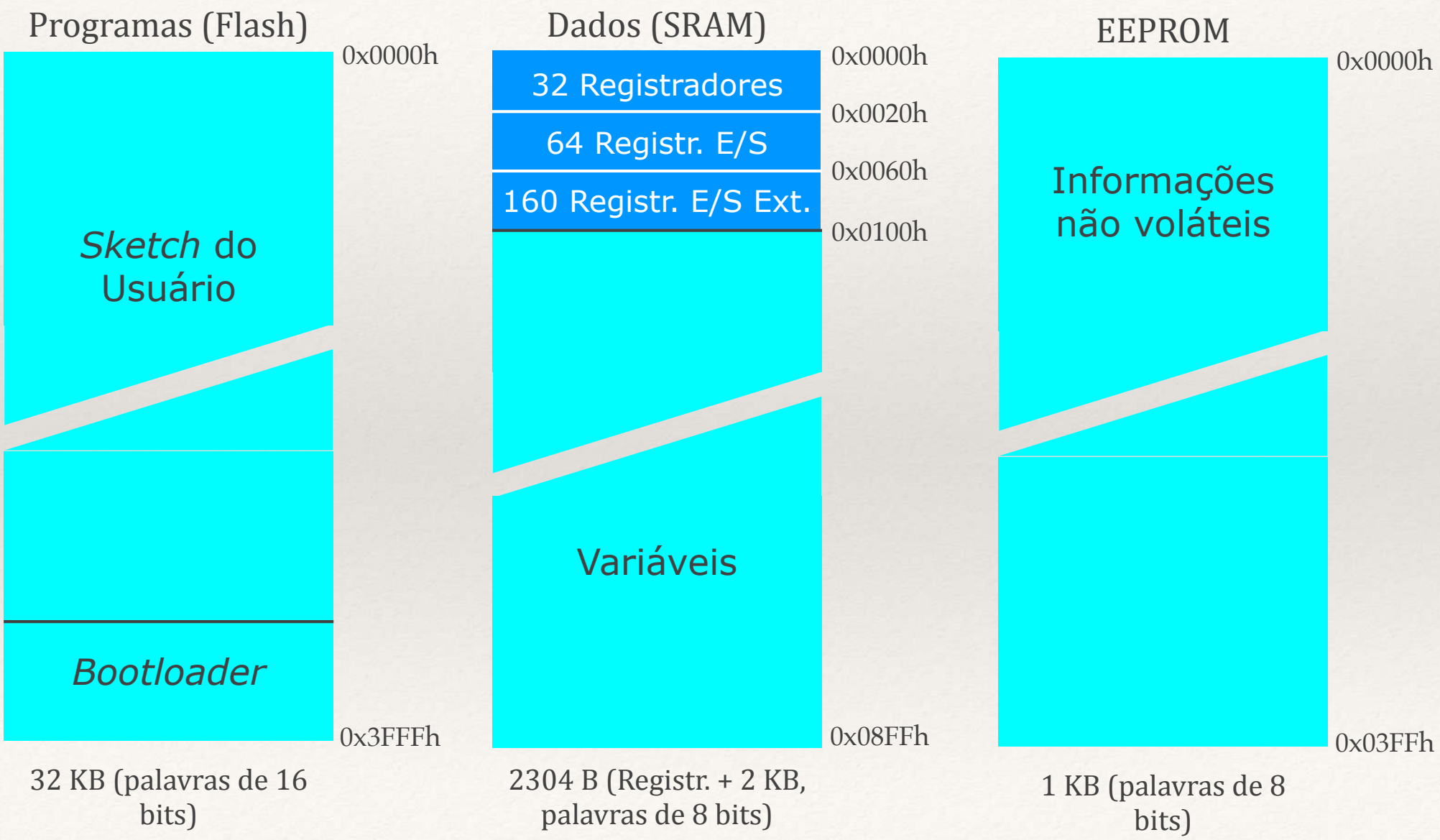
- A Memória que armazena o programa é diferente da memória que contém as variáveis;
- São 3 tipos de memória diferentes;
- O ATMega328P possui 3 tipos diferentes de memória (Flash, SRAM e EEPROM).



"Arquitetura Harvard"

- Ganho de performance por permitir acesso simultâneo às instruções e aos dados;
- Tamanho que contém uma instrução (16 bits) pode ser diferente da palavra de dados (8 bits);
- Processadores modernos misturam Von Neumann com Harvard, através do uso do cache.

Memória no ATMega328P



ATMega328P em blocos

Os registradores controlam toda a operação do ATMega328P

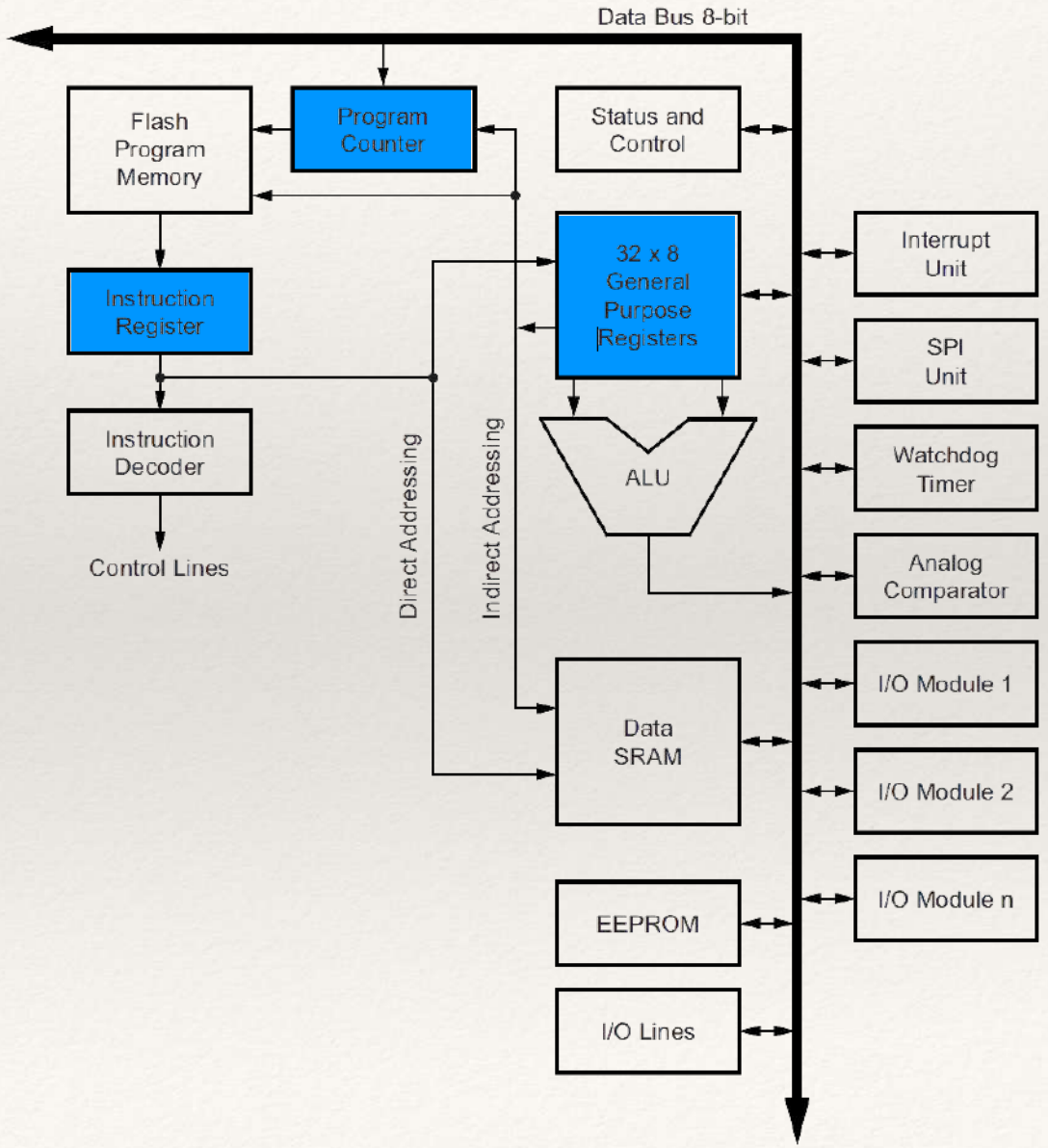
- Muitos são acessados indiretamente quando usamos o conjunto de instruções padrão do Arduíno;

Alguns recursos exigem acesso direto

- Interrupções de Relógio;
- Alteração de frequência PWM;
- Manipulação “criativa" de portas.

Acesso via registradores

- Códigos menores, e mais rápidos;
- Manipulação e consulta binária.



Acesso Binário

Bits dos registradores agregam funções na mesma palavra

- Alterações e consultas bit-a-bit;
- Operações lógicas binárias: E, OU, NÃO, deslocamento.

Registradores de 8 ou 16 bits

- Registradores de 16 bits tipicamente são tratados byte a byte.

Ativando bits (*set bits*)

- Diretamente pela carga da palavra;
- Carga de bit com operação OU (*OR*).

Desativando bits (*reset bits*)

- Desativação de bit com operação E (*AND*).

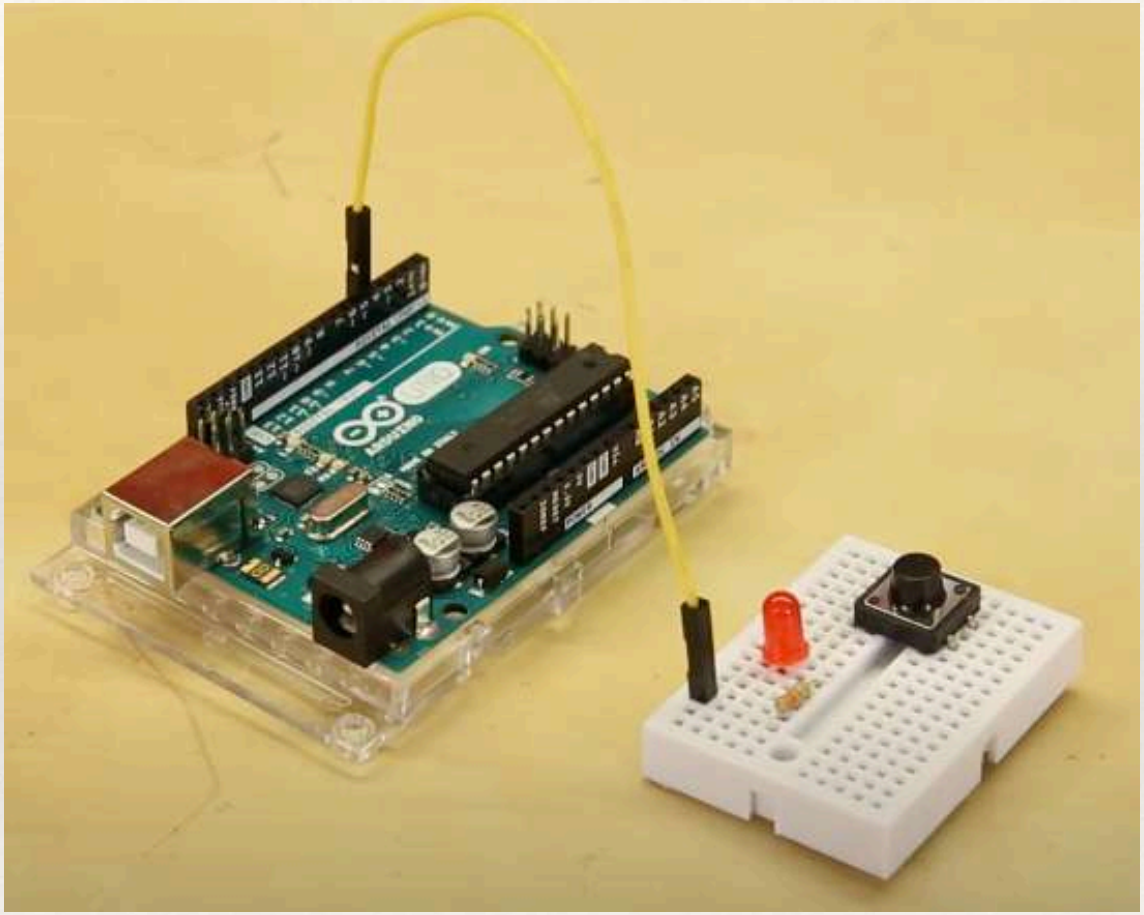
Acesso Binário - exemplo

Projeto básico com Arduíno Uno

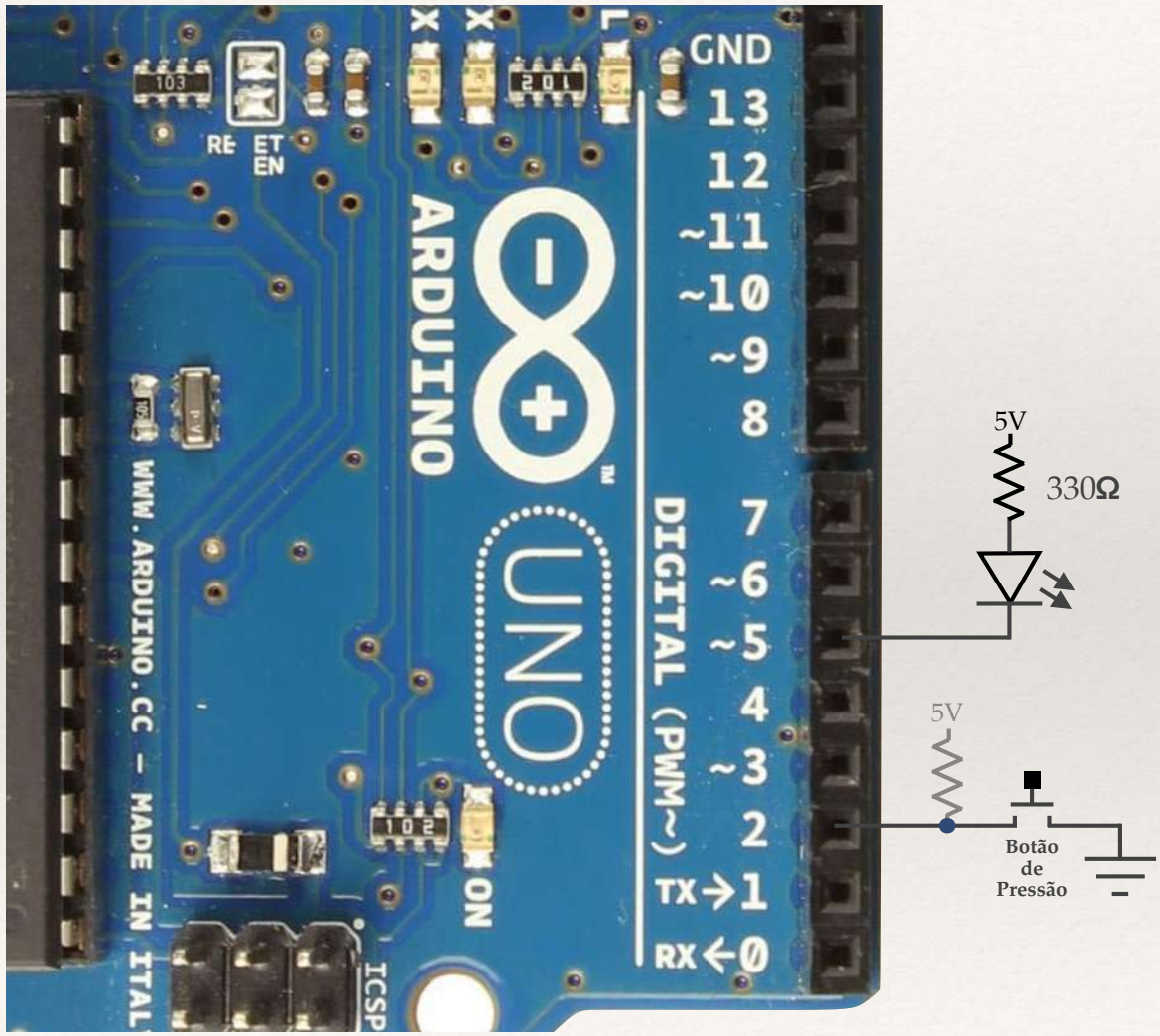
- Ler botão;
- Se apertado, acende LED; caso contrário, apaga.

Desenvolvimento do teste

- Teste inicial com código em C padrão do Arduíno;
- Migração de parte do código, desta vez manipulando diretamente os registradores.



Acesso Binário - exemplo



Componentes alimentados pelo Arduíno Uno

- LED (digital 5);
- Botão de Leitura (digital 2);
- Pull-Up interno.

Projeto Básico

Sketch define entrada e saída

- Pino 2 utilizado para leitura do botão, com uso de PullUp interno;
- Pino 5 utilizado para ativar o LED.

Função básica Arduíno

- `pinMode();`
- Opera indiretamente com registradores.

Esquemático Arduíno Uno

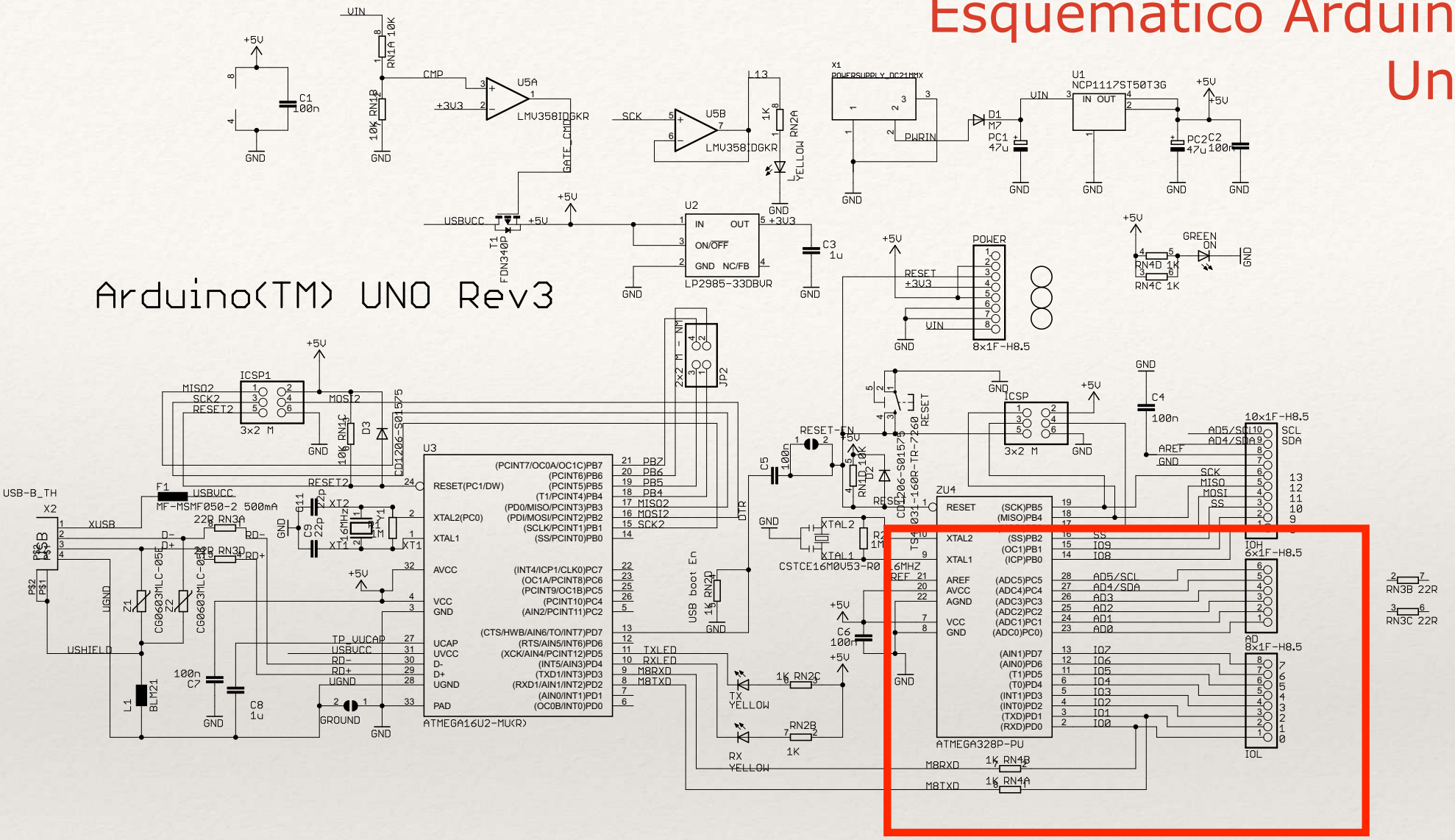
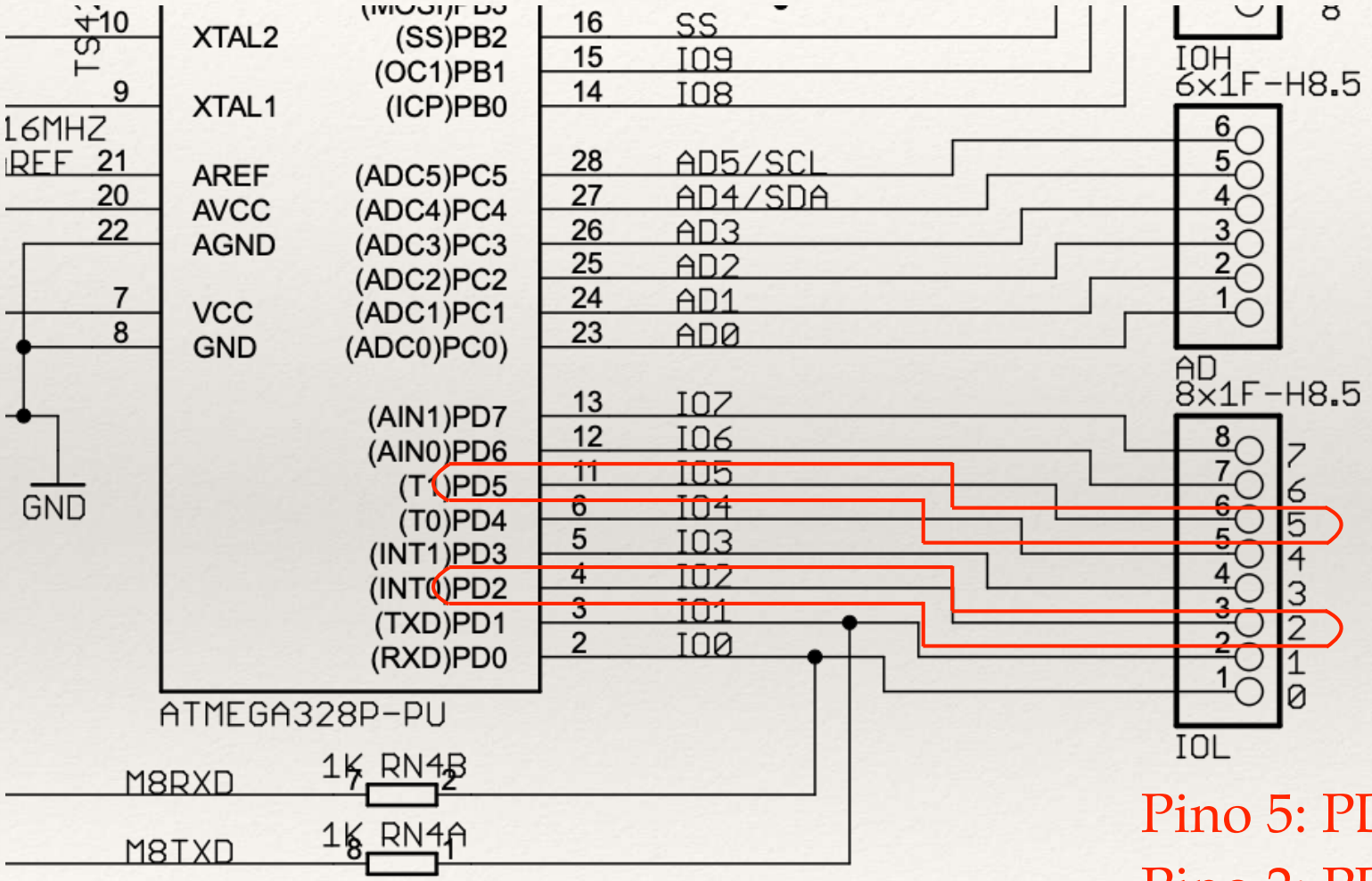


Diagrama esquemático do Arduíno Uno v3

Registradores pinos 2, 5 ?

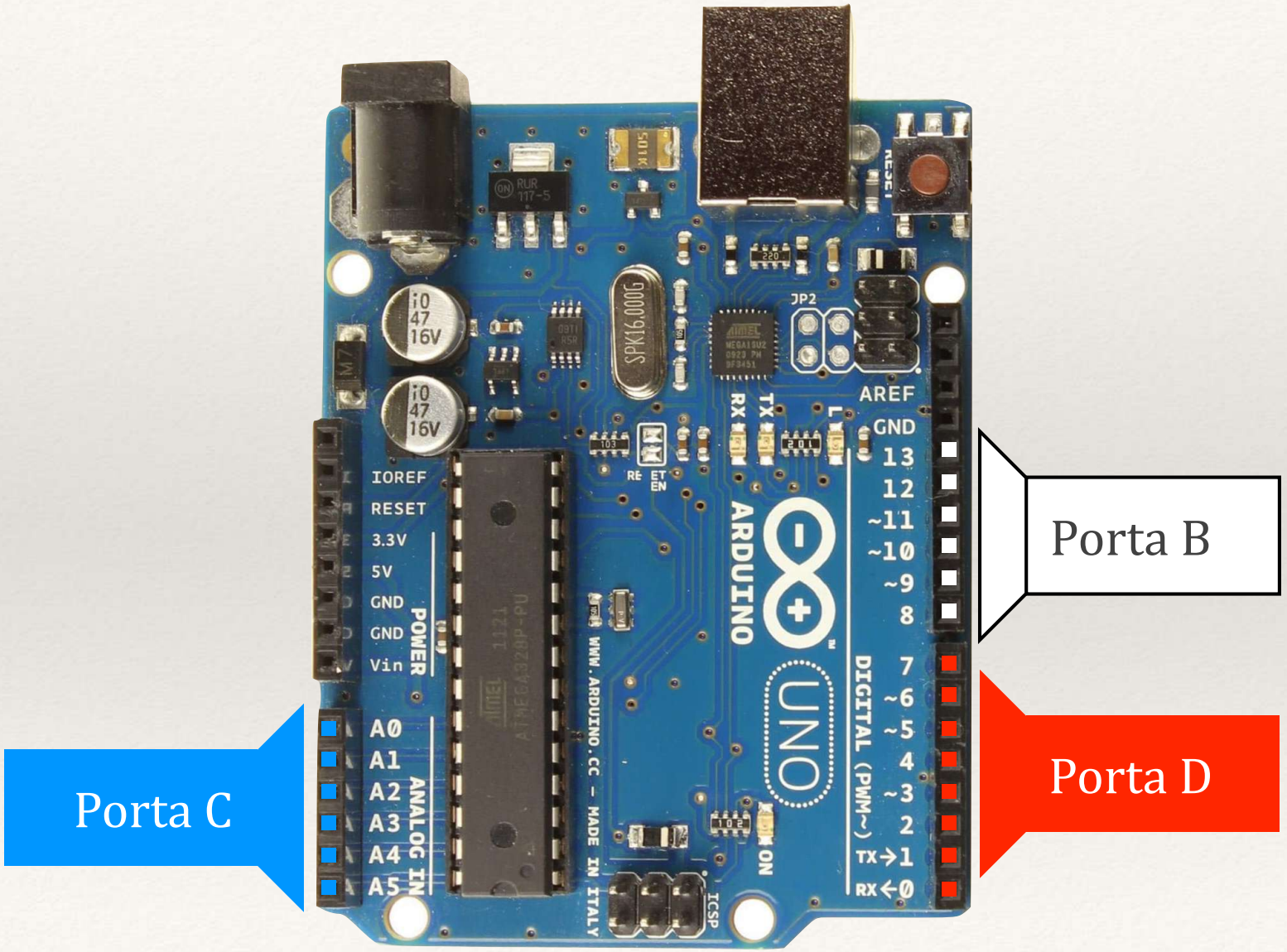


Pino 5: PD5

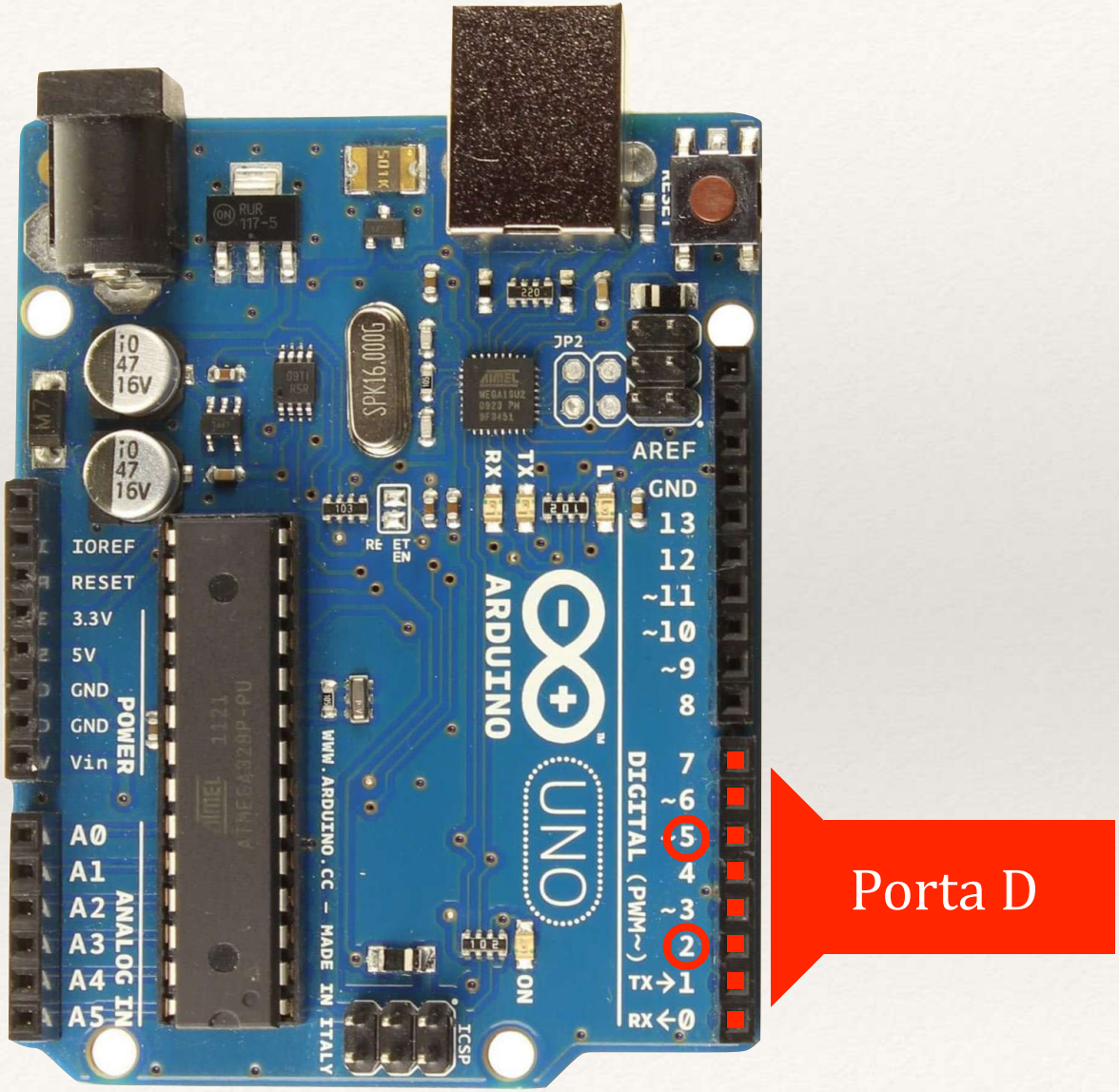
Pino 2: PD2

(ambos na **porta D**)

Arduíno: portas B, C e D



Arduíno: portas B, C e D



Registradores - Porta D

PORTD – The Port D Data Register

Bit	7	6	5	4	3	2	1	0	
0x0B (0x2B)	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	PORTD
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

DDRD – The Port D Data Direction Register

Bit	7	6	5	4	3	2	1	0	
0x0A (0x2A)	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	DDRD
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

PIND – The Port D Input Pins Address

Bit	7	6	5	4	3	2	1	0	
0x09 (0x29)	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	PIND
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

Datasheet do ATMega328P

Registradores - Porta D

PORTD – The Port D Data Register

Bit	7	6	5	4	3	2	1	0	
0x0B (0x2B)	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	PORTD
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

DDRD – The Port D Data Direction Register

0x0A								DDD0	DDRD
Read/Write								R/W	
Initial Value								0	

O registrador DDR determina quais são os pinos que serão utilizados como entrada (valor 0) ou saída (valor 1)

PIND – The Port D Input Pins Address

Bit	7	6	5	4	3	2	1	0	
0x09 (0x29)	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	PIND
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

Datasheet do ATMega328P

Registradores - Porta D

O registrador PORT determina, no caso de SAÍDA, qual o valor a ser apresentado; no caso de entrada, habilita o resistor interno de *PullUp* (1), ou desabilita (0)

DDRD – The Port D Data Direction Register

Bit	7	6	5	4	3	2	1	0	
0x0A (0x2A)	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	DDRD
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

PIND – The Port D Input Pins Address

Bit	7	6	5	4	3	2	1	0	
0x09 (0x29)	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	PIND
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

Datasheet do ATMega328P

Registradores - Porta D

PORTD – The Port D Data Register

Bit	7	6	5	4	3	2	1	0	
0x0B (0x2B)	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	PORTD
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

DDRD – The Port D Data Direction Register

Bit	7	6	5	4	3	2	1	0	
0x0A (0x2A)	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	DDRD
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

PIND – The Port D Input Pins Address

							1	0	
0x0							PIND1	PIND0	PIND
Re							R	R	
Init							N/A	N/A	

O registrador PIN, que deve ser tratado apenas para leitura, possui o status atual de cada pino

Datasheet do ATMega328P

Acesso Binário - exemplo

Sketch define pino 5 como saída e pino 2 como entrada

1º passo: carregar 1 no bit 5 do registrador DDRD;

2º passo: carregar 1 no bit 2 do registrador PORTD;

Atribuindo o valor da palavra

Em binário:

```
DDRD  = B00100000;
PORTD = B00000100;
```

Ou, em decimal:

```
DDRD  = 32;
PORTD =  2;
```

Acesso Binário - exemplo

Atribuindo o valor ao BIT !

Operação OU com conteúdo anterior:

```
DDRD  |= B00100000;
PORTD |= B00000100;
```

Embora funcione, este procedimento força a criação de uma constante, e de uma variável provisória (lento?);

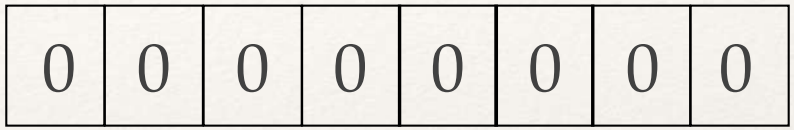
Existe um procedimento mais comum para este tipo de operação, que utiliza o registro de deslocamento.

Acesso Binário - exemplo

Ativando o bit 5 em DDRD:

Criamos uma palavra com o bit desejado ativo:

```
( 1 << pinoLed )
```

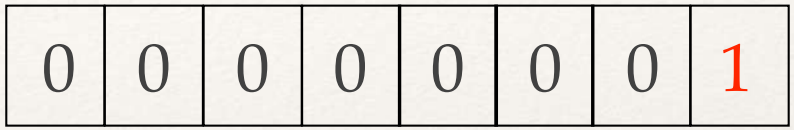


Acesso Binário - exemplo

Ativando o bit 5 em DDRD:

Criamos uma palavra com o bit desejado ativo:

```
( 1 << pinoLed )
```

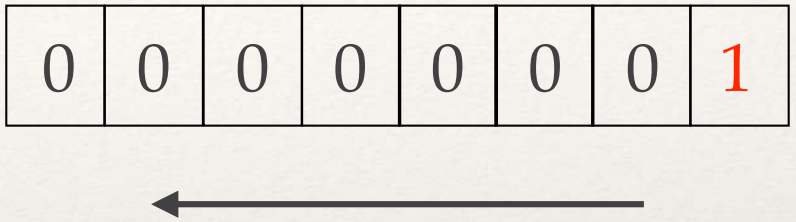


Acesso Binário - exemplo

Ativando o bit 5 em DDRD:

Criamos uma palavra com o bit desejado ativo:

```
( 1 << 5 )
```

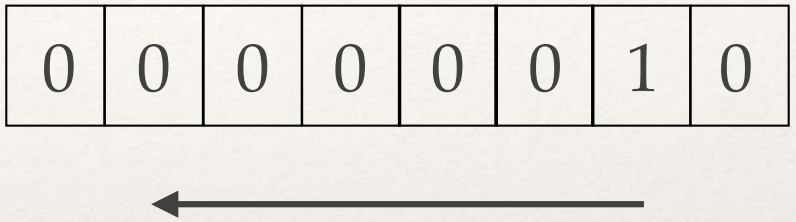


Acesso Binário - exemplo

Ativando o bit 5 em DDRD:

Criamos uma palavra com o bit desejado ativo:

```
( 1 << 4 )
```

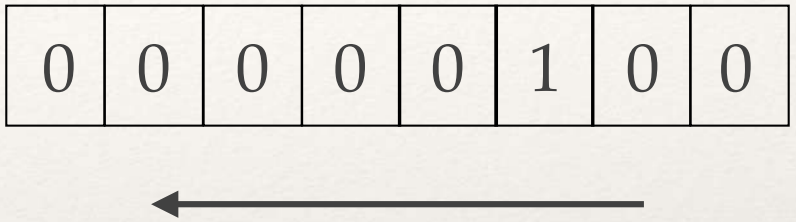


Acesso Binário - exemplo

Ativando o bit 5 em DDRD:

Criamos uma palavra com o bit desejado ativo:

```
( 1 << 3 )
```

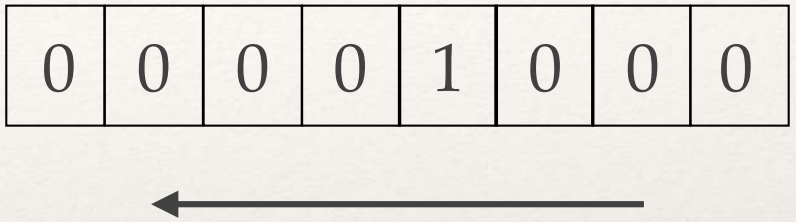


Acesso Binário - exemplo

Ativando o bit 5 em DDRD:

Criamos uma palavra com o bit desejado ativo:

```
( 1 << 2 )
```

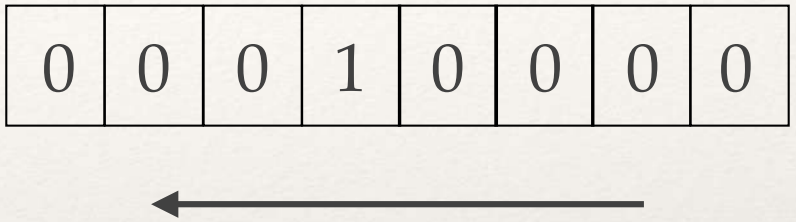


Acesso Binário - exemplo

Ativando o bit 5 em DDRD:

Criamos uma palavra com o bit desejado ativo:

```
( 1 << 1 )
```

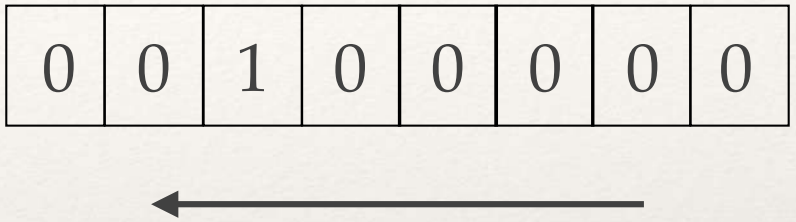


Acesso Binário - exemplo

Ativando o bit 5 em DDRD:

Criamos uma palavra com o bit desejado ativo:

```
( 1 << 0 )
```



Acesso Binário - exemplo

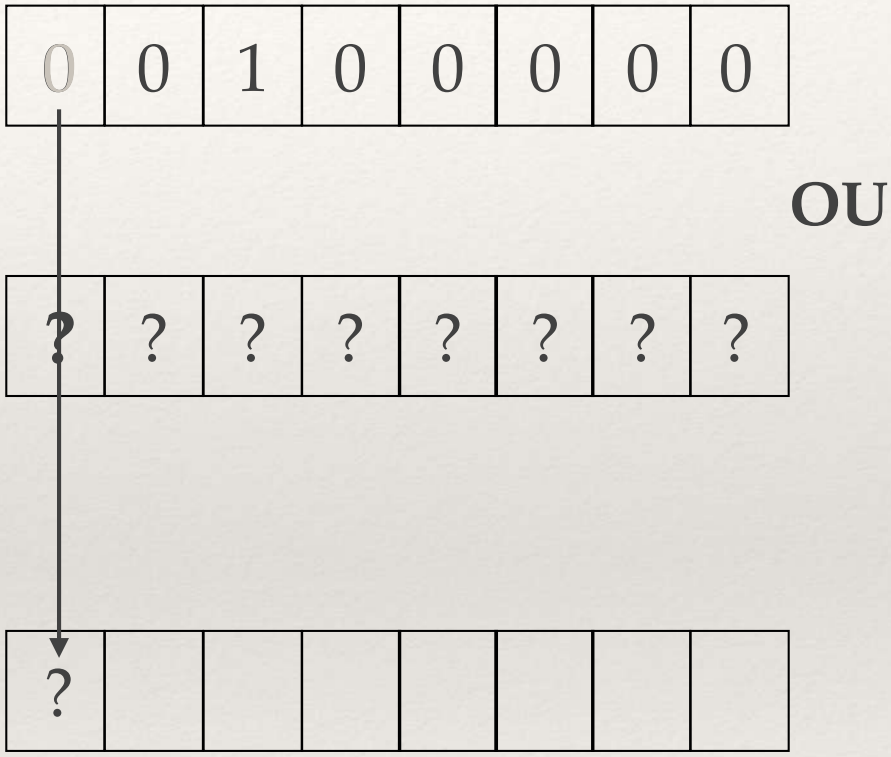
Ativando o bit 5 em DDRD:

Criamos uma palavra com o bit desejado ativo:

```
(1<<pinoLed)
```

Fazemos uma operação OU (|) com o valor atual de DDRD:

```
DDRD =( 1 << pinoLed ) | DDRD;
```



Acesso Binário - exemplo

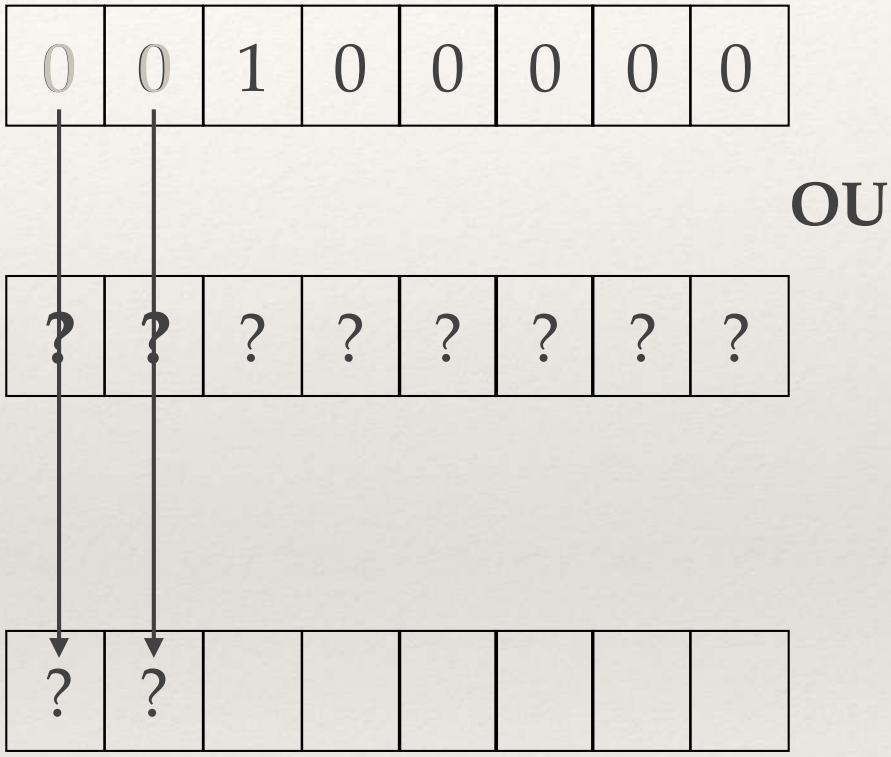
Ativando o bit 5 em DDRD:

Criamos uma palavra com o bit desejado ativo:

```
(1<<pinoLed)
```

Fazemos uma operação OU (|) com o valor atual de DDRD:

```
DDRD =( 1 << pinoLed ) | DDRD;
```



Acesso Binário - exemplo

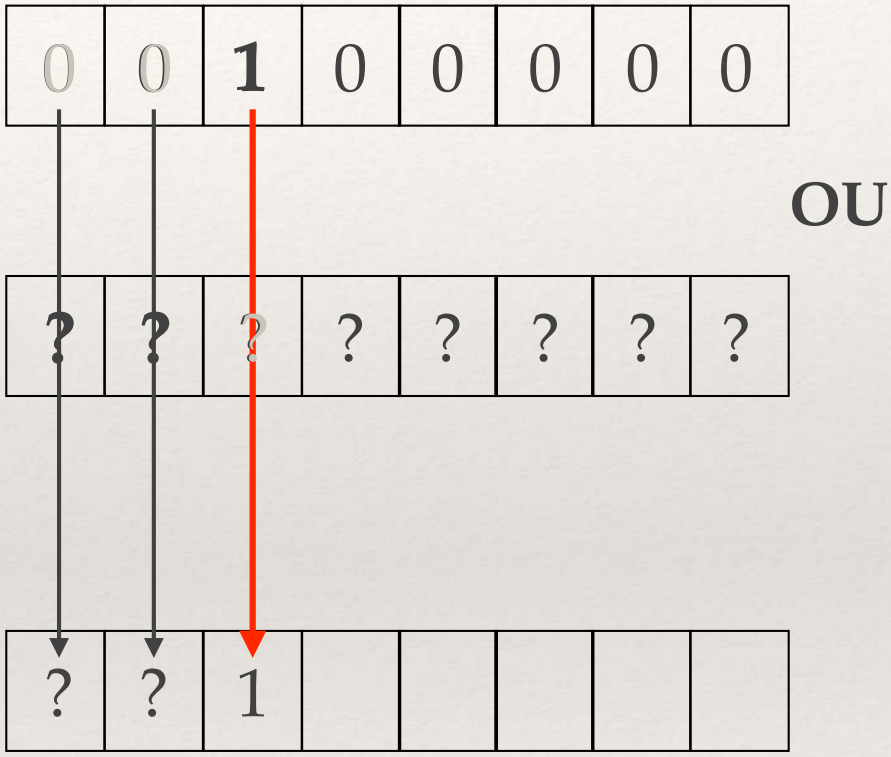
Ativando o bit 5 em DDRD:

Criamos uma palavra com o bit desejado ativo:

```
(1<<pinoLed)
```

Fazemos uma operação OU (|) com o valor atual de DDRD:

```
DDRD =( 1 << pinoLed ) | DDRD;
```



Acesso Binário - exemplo

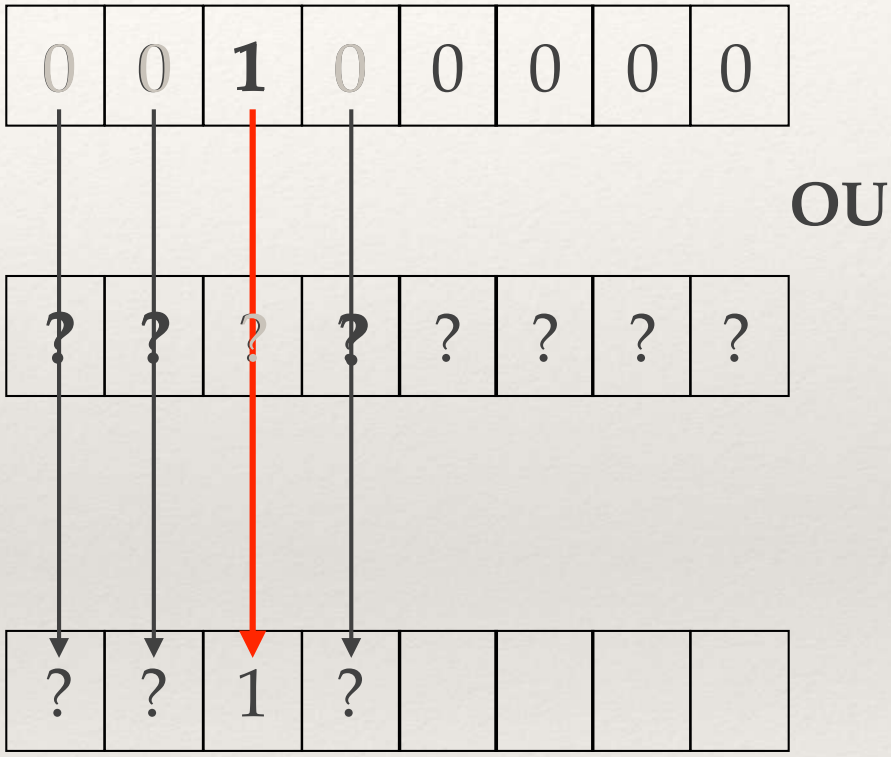
Ativando o bit 5 em DDRD:

Criamos uma palavra com o bit desejado ativo:

```
(1<<pinoLed)
```

Fazemos uma operação OU (|) com o valor atual de DDRD:

```
DDRD =( 1 << pinoLed ) | DDRD;
```



Acesso Binário - exemplo

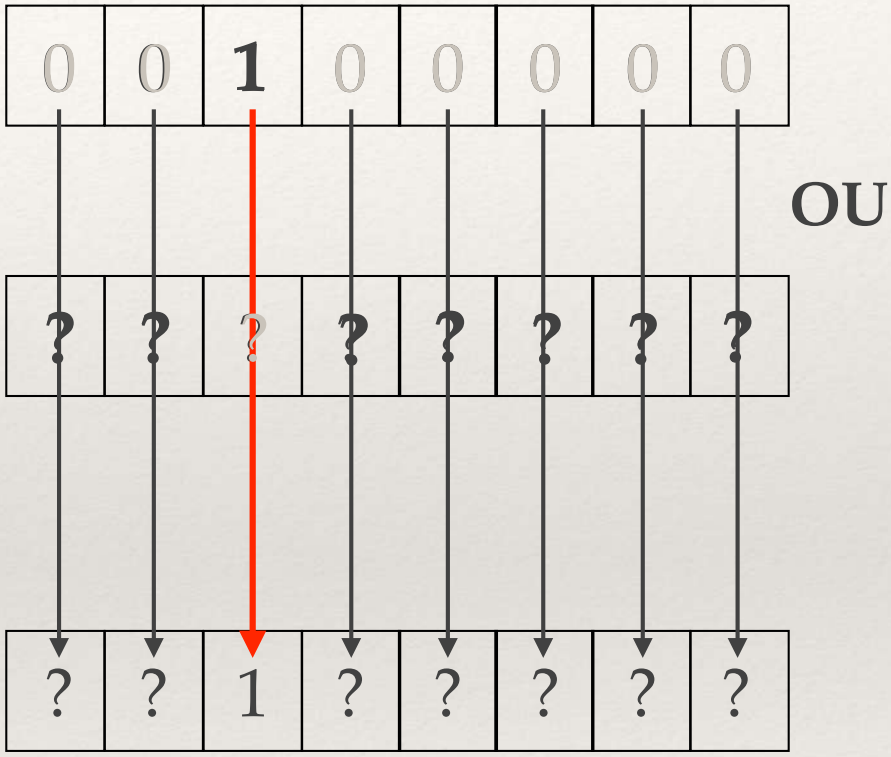
Ativando o bit 5 em DDRD:

Criamos uma palavra com o bit desejado ativo:

```
(1<<pinoLed)
```

Fazemos uma operação OU (|) com o valor atual de DDRD:

```
DDRD =( 1 << pinoLed ) | DDRD;
```



Acesso Binário - exemplo

Ativando o bit 5 em DDRD:

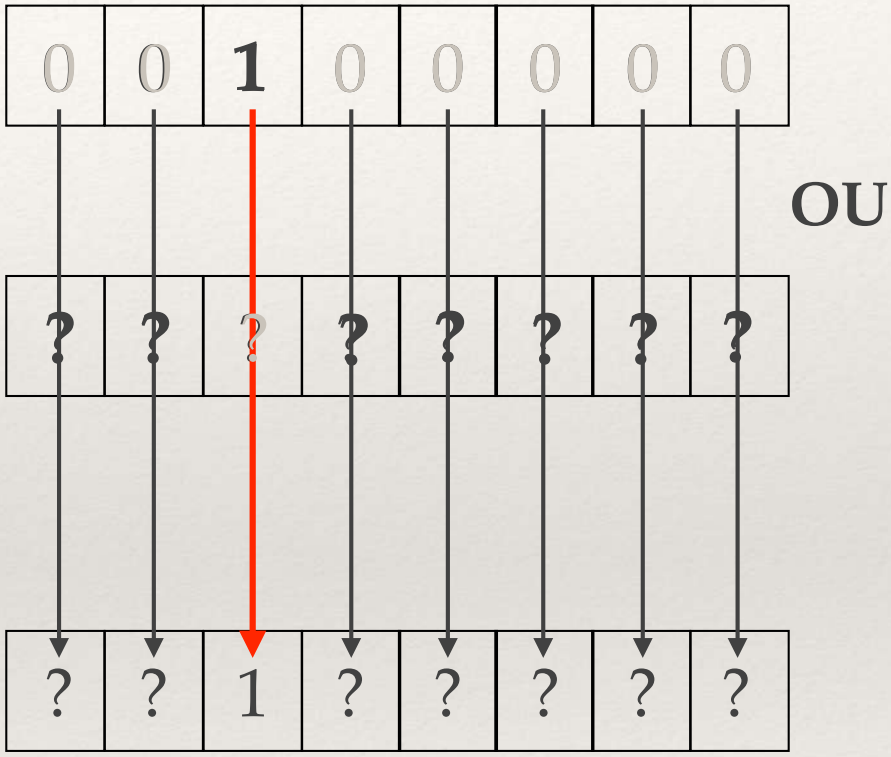
Criamos uma palavra com o bit desejado ativo:

```
(1<<pinoLed)
```

Fazemos uma operação OU (|) com o valor atual de DDRD:

```
DDRD =( 1 << pinoLed ) | DDRD;
```

Todos os bits de DDRD são mantidos, exceto o bit 5, que assume o valor 1.



Acesso Binário - exemplo

Desativando o bit 5 em DDRD:

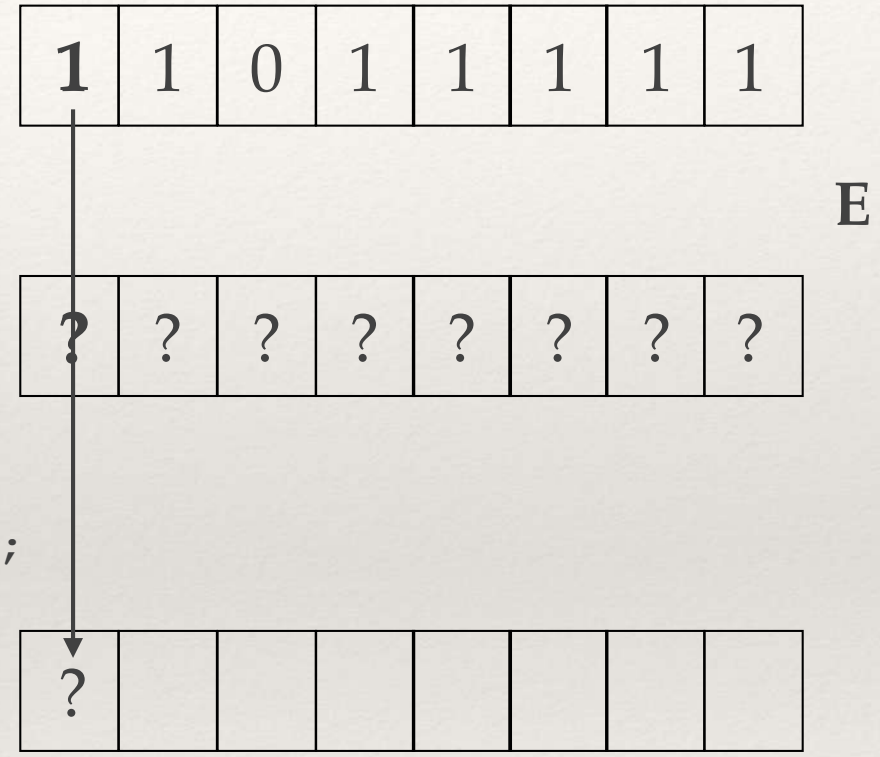
Criamos uma palavra com o bit desejado ativo:

`(1<<pinoLed)`

Invertemos a palavra bit-a-bit (~);

Fazemos uma operação E (&) com o valor atual de DDRD:

```
DDRD = ~( 1 << pinoLed ) & DDRD;
```



Acesso Binário - exemplo

Desativando o bit 5 em DDRD:

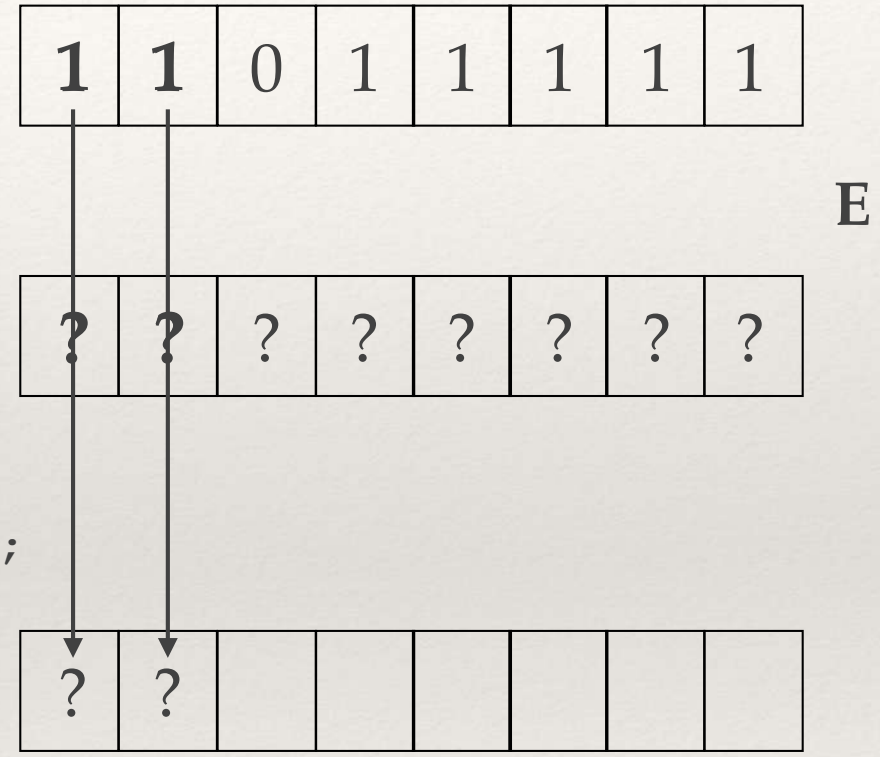
Criamos uma palavra com o bit desejado ativo:

`(1<<pinoLed)`

Invertemos a palavra bit-a-bit (~);

Fazemos uma operação E (&) com o valor atual de DDRD:

```
DDRD = ~( 1 << pinoLed ) & DDRD;
```



Acesso Binário - exemplo

Desativando o bit 5 em DDRD:

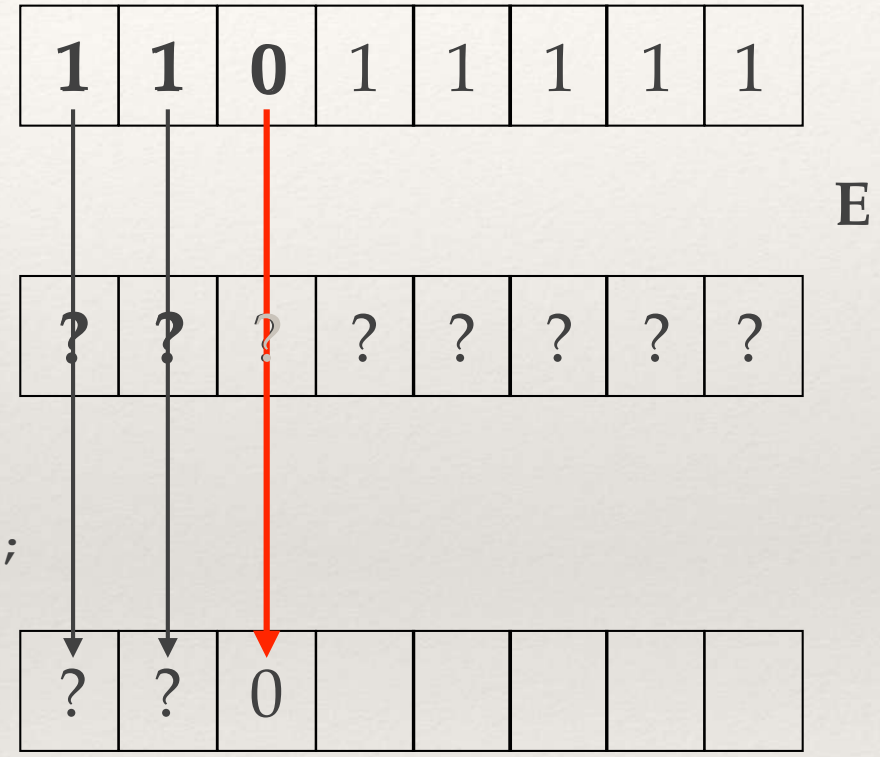
Criamos uma palavra com o bit desejado ativo:

`(1<<pinoLed)`

Invertemos a palavra bit-a-bit (~);

Fazemos uma operação E (&) com o valor atual de DDRD:

```
DDRD = ~( 1 << pinoLed ) & DDRD;
```



Acesso Binário - exemplo

Desativando o bit 5 em DDRD:

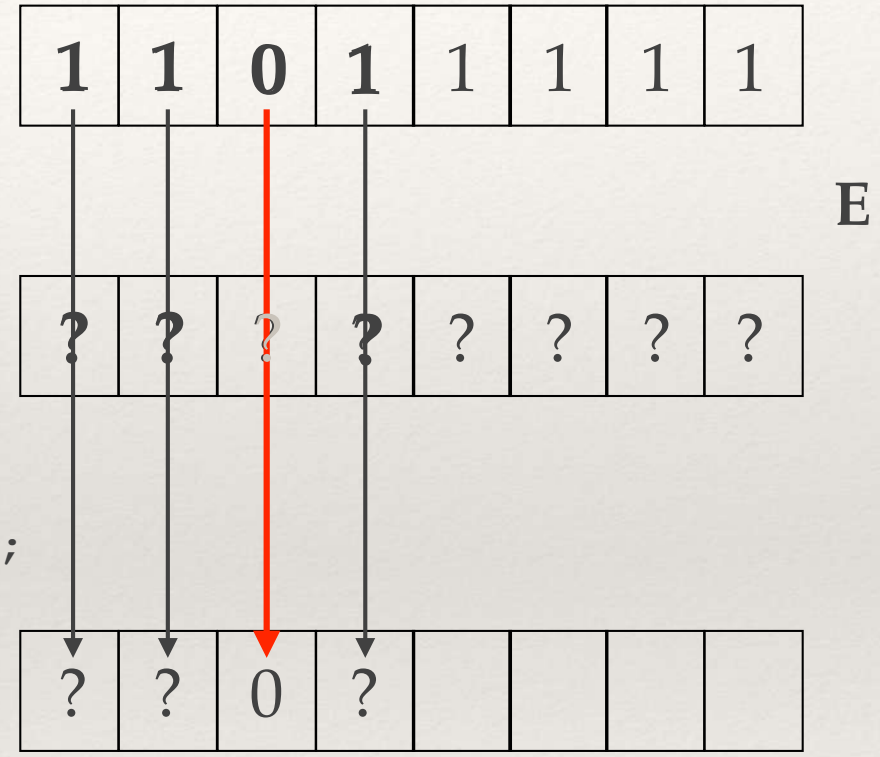
Criamos uma palavra com o bit desejado ativo:

`(1<<pinoLed)`

Invertemos a palavra bit-a-bit (~);

Fazemos uma operação E (&) com o valor atual de DDRD:

```
DDRD = ~( 1 << pinoLed ) & DDRD;
```



Acesso Binário - exemplo

Desativando o bit 5 em DDRD:

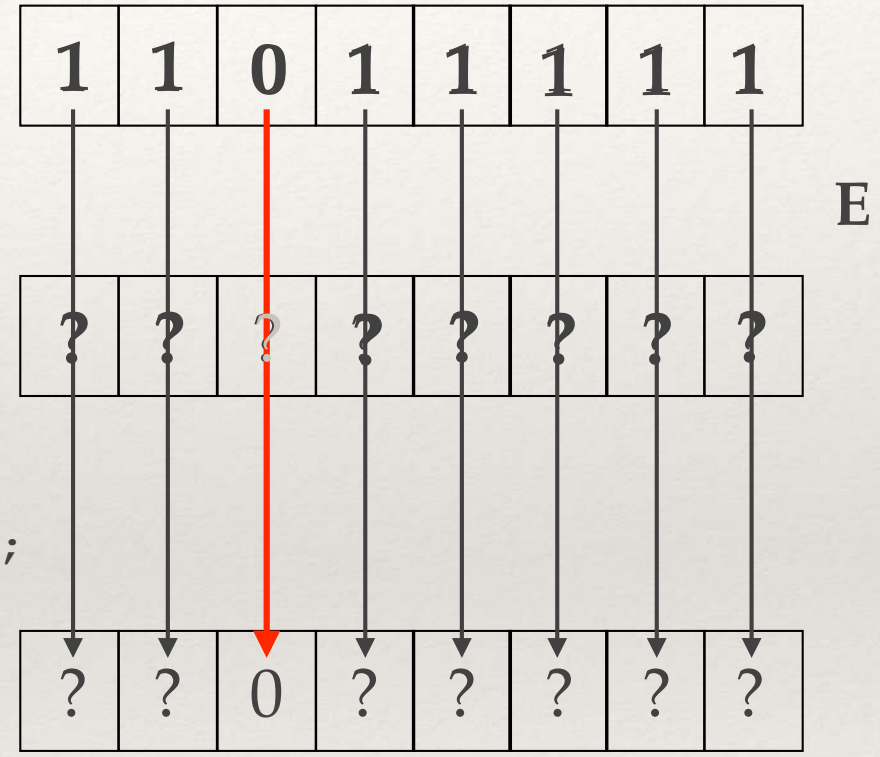
Criamos uma palavra com o bit desejado ativo:

`(1<<pinoLed)`

Invertemos a palavra bit-a-bit (~);

Fazemos uma operação E (&) com o valor atual de DDRD:

```
DDRD = ~( 1 << pinoLed ) & DDRD;
```



Todos os bits de DDRD são mantidos, exceto o bit 5, que assume o valor 0.

Acesso Binário - exemplo

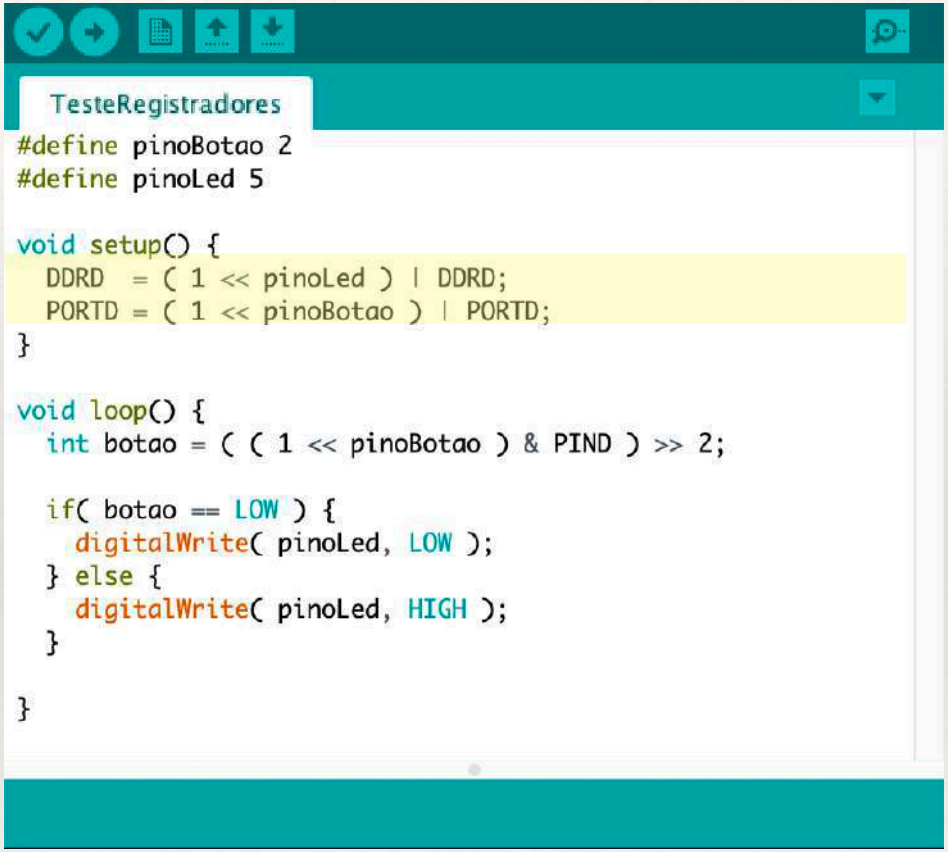
Para ativar o bit 2 de PORTD em DDRD, o processo é similar

- Criamos uma palavra com o bit desejado ativo:

  `(1<<pinoBotao)`

- Fazemos uma operação OU (|) com o valor atual de PORTD:

  `PORTD =( 1 << pinoBotao ) | PORTD;`



Acesso Binário - exemplo

Para leitura do Botão, utilizamos o registro PIND

Criamos uma palavra colocando 1 no bit a ser lido:

```
( 1 << pinoBotao )
```

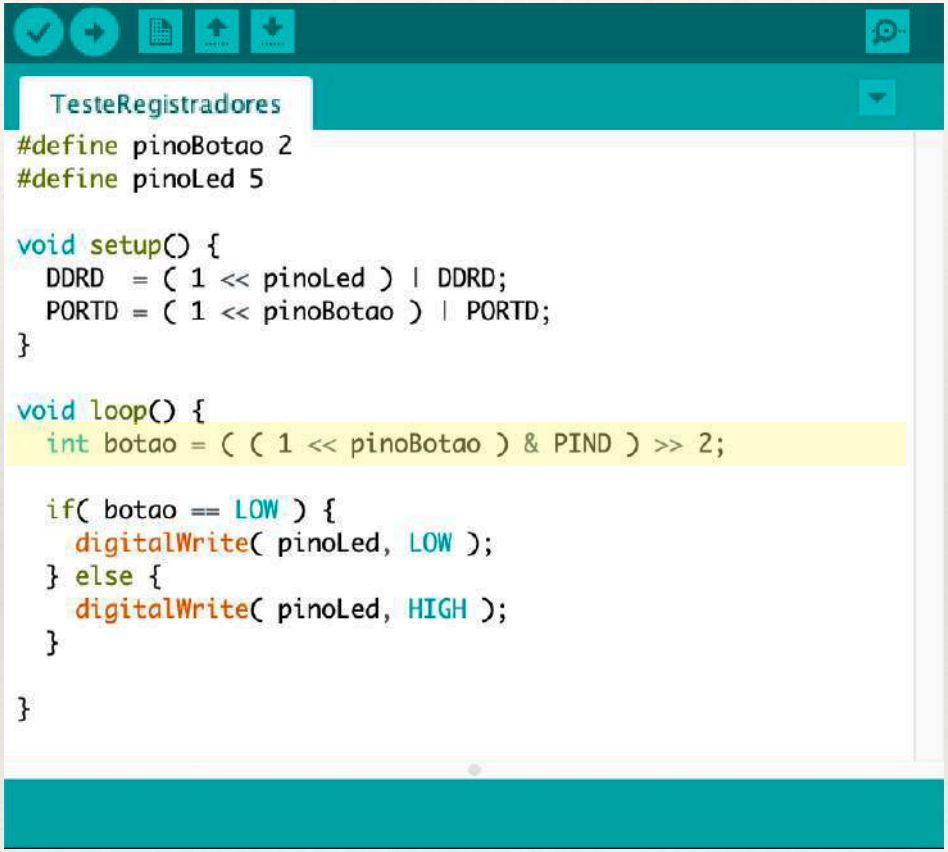
Fazemos uma operação E (&) com o valor atual de PIND:

```
( 1 << pinoBotao ) & PIND;
```

Revertemos o deslocamento para colocar o bit testado na posição zero:

```
botao = ( ( 1 << pinoBotao ) & PIND ) >> 2;
```

O valor será 1 (verdadeiro) ou 0 (falso), a depender do valor lido na posição desejada do PIND.



Acesso Binário - exemplo

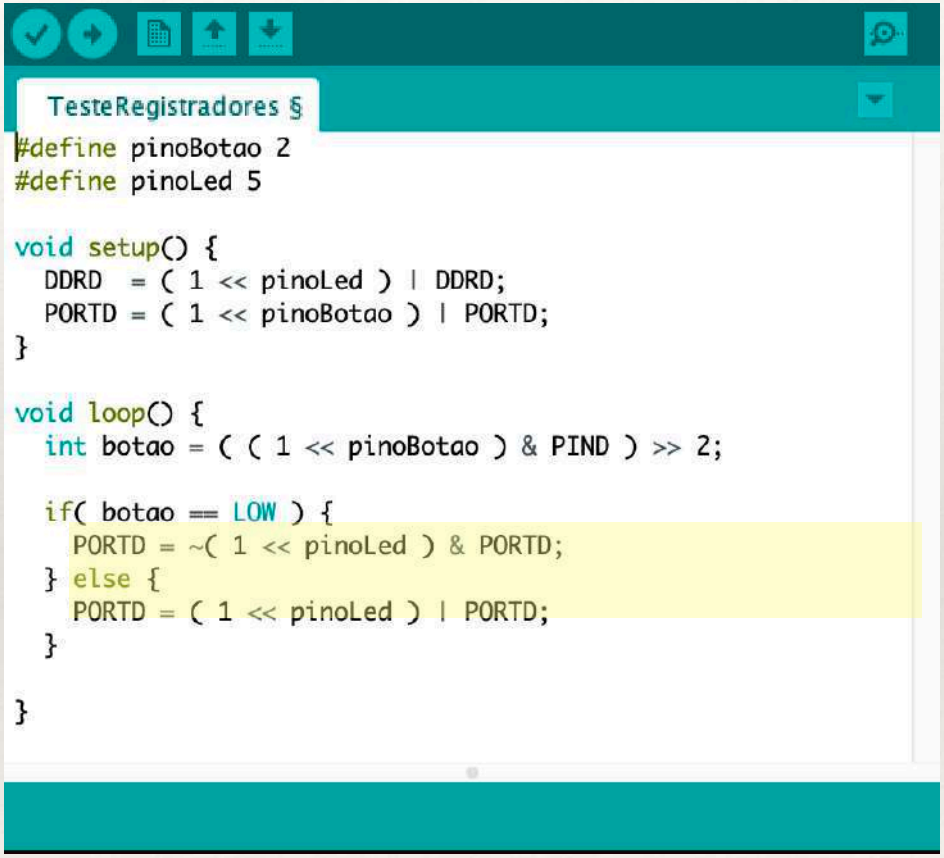
Para acender ou apagar o LED, os processos são bem similares

Criamos uma palavra com o bit desejado ativo:

```
( 1 << pinoLed )
```

Fazemos uma operação OU (|) com o valor atual de PORTD:

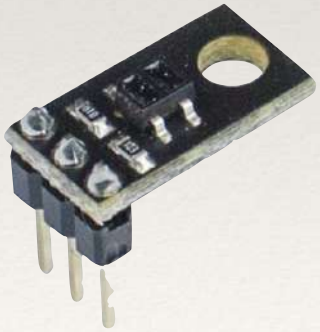
```
if( botão == LOW ) {
    PORTD = ~( 1 << pinoLed )
& PORTD;
} else {
    PORTD =( 1 << pinoLed ) |
PORTD;
}
```

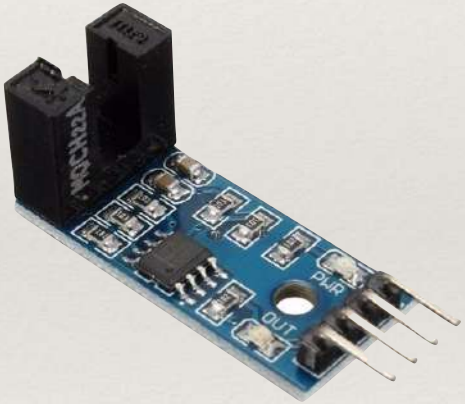


Aula 10

Dispositivos de Entrada

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





Interrupções

Uma das formas mais eficazes de controlar operações de entrada e saída;

Podem ser acionadas por eventos internos (ex. *timers*) ou externos (ex. sinais elétricos);

Provocam o desvio de execução do *sketch* para uma *Interrupt Service Routine* (ISR) específica. Ao término da ISR, a execução retorna ao ponto original;

A ISR deve ser construída com alguns cuidados especiais.

Interrupções ATMega328P

Um microcontrolador normalmente possui vetores de interrupções

- Associam cada interrupção a uma (ISR);
- A tabela ao lado mostra os vetores do ATMega328P; alguns deles estão associados a 3 diferentes *timers* (0, 1 e 2);
- Em todos há 2 funções de comparação, e uma de *overflow*.

Vector	Program Address	Source	Interrupt Definition
1	0	RESET	External pin, power-on reset, brown-out reset and watchdog system reset
2	2	INT0	External interrupt request 0
3	4	INT1	External interrupt request 1
4	6	PCINT0	Pin change interrupt request 0
5	8	PCINT1	Pin change interrupt request 1
6	A	PCINT2	Pin change interrupt request 2
7	C	WDT	Watchdog time-out interrupt
8	E	TIMER2 COMPA	Timer/Counter2 compare match A
9	10	TIMER2 COMPB	Timer/Counter2 compare match B
10	12	TIMER2 OVF	Timer/Counter2 overflow
11	14	TIMER1 CAPT	Timer/Counter1 capture event
12	16	TIMER1 COMPA	Timer/Counter1 compare match A
13	18	TIMER1 COMPB	Timer/Counter1 compare match B
14	1A	TIMER1 OVF	Timer/Counter1 overflow
15	1C	TIMER0 COMPA	Timer/Counter0 compare match A
16	1E	TIMER0 COMPB	Timer/Counter0 compare match B
17	20	TIMER0 OVF	Timer/Counter0 overflow
18	22	SPI, STC	SPI serial transfer complete
19	24	USART, RX	USART Rx complete
20	26	USART, UDRE	USART, data register empty
21	28	USART, TX	USART, Tx complete
22	2A	ADC	ADC conversion complete
23	2C	EE READY	EEPROM ready
24	2E	ANALOG COMP	Analog comparator
25	30	TWI	2-wire serial interface
26	32	SPM READY	Store program memory ready

Datasheet do ATMega328P

Interrupções ATMega328P

Os 3 *timers* também estão associados a registradores específicos

Registradores são modificados automaticamente, sem a necessidade de código específico;

Modificações dependem direta ou indiretamente do Clock do sistema.

30. Register Summary (Continued)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
(0xB4)	OCR2B	Timer/Counter2 output compare register B								131
(0xB3)	OCR2A	Timer/Counter2 output compare register A								131
(0xB2)	TCNT2	Timer/Counter2 (8-bit)								131
0x28 (0x48)	OCR0B	Timer/Counter0 output compare register B								
0x27 (0x47)	OCR0A	Timer/Counter0 output compare register A								
0x26 (0x46)	TCNT0	Timer/Counter0 (8-bit)								
(0x8B)	OCR1BH	Timer/Counter1 - Output compare register B high byte								111
(0x8A)	OCR1BL	Timer/Counter1 - Output compare register B low byte								111
(0x89)	OCR1AH	Timer/Counter1 - Output compare register A high byte								111
(0x88)	OCR1AL	Timer/Counter1 - Output compare register A low byte								111
(0x87)	ICR1H	Timer/Counter1 - Input capture register high byte								112
(0x86)	ICR1L	Timer/Counter1 - Input capture register low byte								112
(0x85)	TCNT1H	Timer/Counter1 - Counter register high byte								111
(0x84)	TCNT1L	Timer/Counter1 - Counter register low byte								111

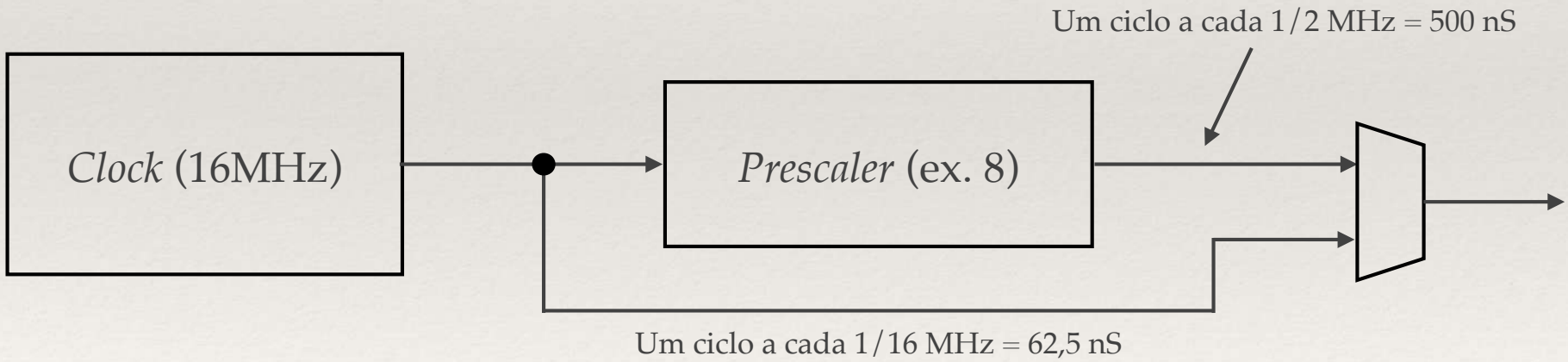
} 16 bits

Datasheet do ATMega328P

Interrupções ATMega328P

No Arduíno Uno, o clock é de 16 MHz

- Cada microcontrolador tem sua própria taxa de clock;
- O clock pode ser dividido por valores pré-definidos (*prescaler*);
- O clock pode ter seu valor comparado para gerar interrupções .



Datasheet do ATMega328P

Interrupções ATMega328P

No Arduíno Uno, o clock é de 16 MHz

Cada microcontrolador tem sua própria taxa de clock;

O clock pode ser dividido por valores pré-definidos (*prescaler*);

O clock pode ter seu valor comparado para gerar interrupções .

Timer	Tamanho	Interrupções Possíveis	Uso no Arduíno Uno
0	8 bits (0 ~ 255)	Comparação *Overflow*	delay(), millis(), micros() analogWrite() pinos 5 e 6
1	16 bits (0 ~ 65535)	Comparação *Overflow* *Captura de Entrada*	Funções para servos analogWrite() pinos 9 e 10
2	8 bits (0 ~ 255)	Comparação *Overflow*	tone() analogWrite() pinos 3 e 11

Datasheet do ATMega328P

Interrupções ATMega328P

Definindo o *prescaler* (ex. timer 1)

- O timer 1 não afeta millis();
- Processo é similar nos demais timers.

(*) Datasheet do ATMega328P

TCCR1B – Timer/Counter1 Control Register B

Bit	7	6	5	4	3	2	1	0	
(0x81)	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	TCCR1B
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Table 15-6. Clock Select Bit Description

CS12	CS11	CS10	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	$clk_{I/O}$/1 (no prescaling)
0	1	0	$clk_{I/O}$/8 (from prescaler)
0	1	1	$clk_{I/O}$/64 (from prescaler)
1	0	0	$clk_{I/O}$/256 (from prescaler)
1	0	1	$clk_{I/O}$/1024 (from prescaler)
1	1	0	External clock source on T1 pin. Clock on falling edge.
1	1	1	External clock source on T1 pin. Clock on rising edge.

Aula 11

Interrupções (prática)

Para testar as interrupções com *timers*, vamos modificar o clássico “Blink”;

O código original pode ser obtido em

`Arquivos→Exemplos→Basics→Blink`

Vamos iniciar o processo testando o código original

LED_BUILTIN = 13

Interrupções (prática)

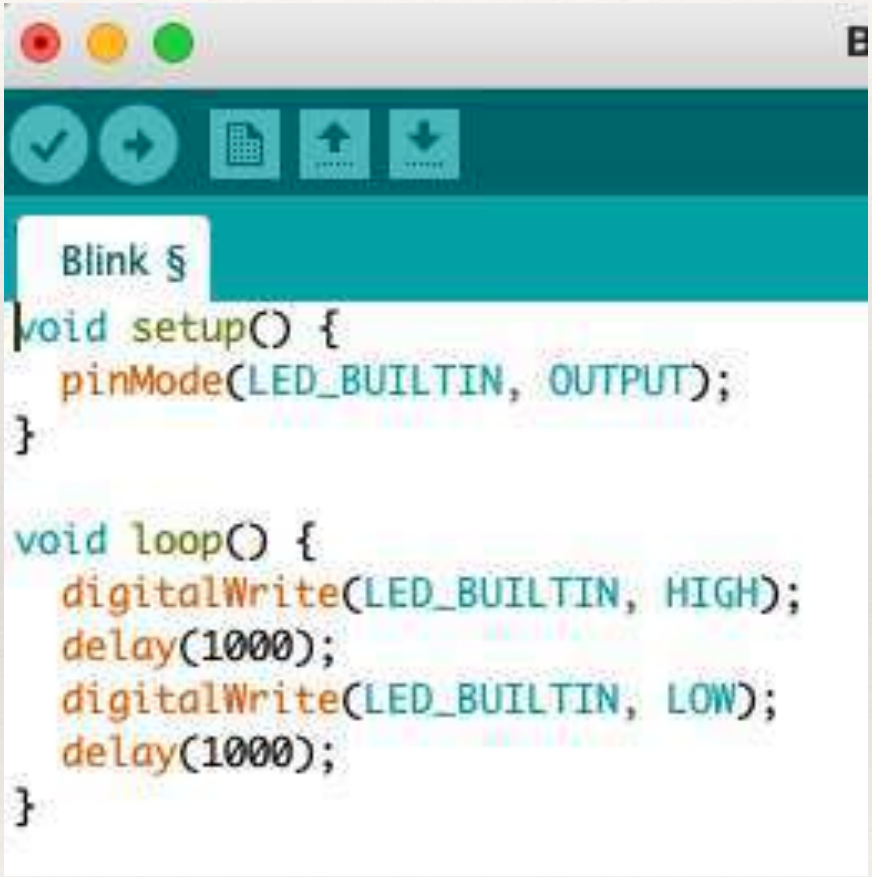
Para testar as interrupções com *timers*, vamos modificar o clássico “Blink”;

O código original pode ser obtido em

`Arquivos→Exemplos→Basics→Blink`

Vamos iniciar o processo testando o código original

LED_BUILTIN = 13



Esquemático Arduíno Uno

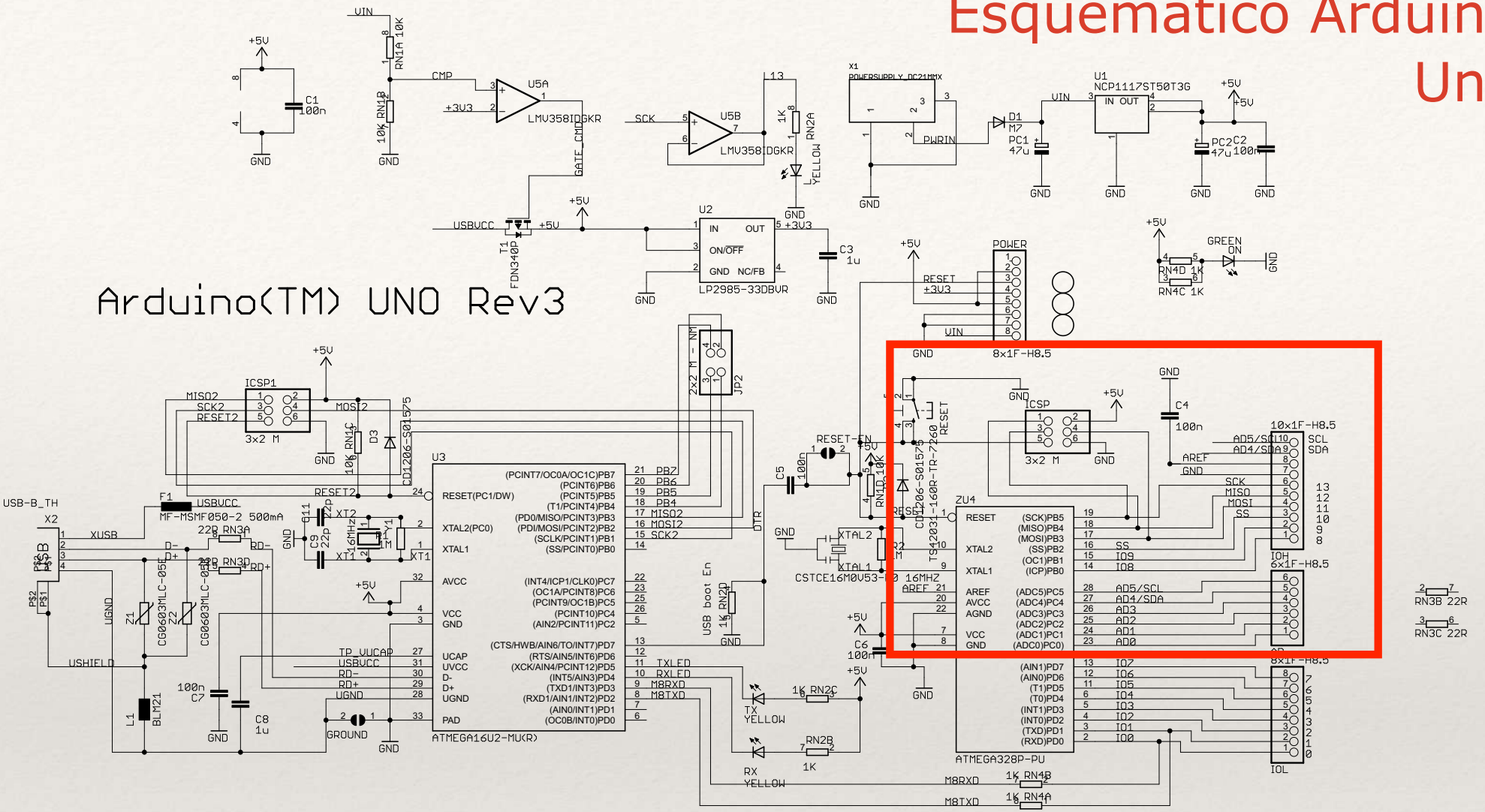


Diagrama esquemático do Arduíno Uno v3

Esquemático Arduíno Uno

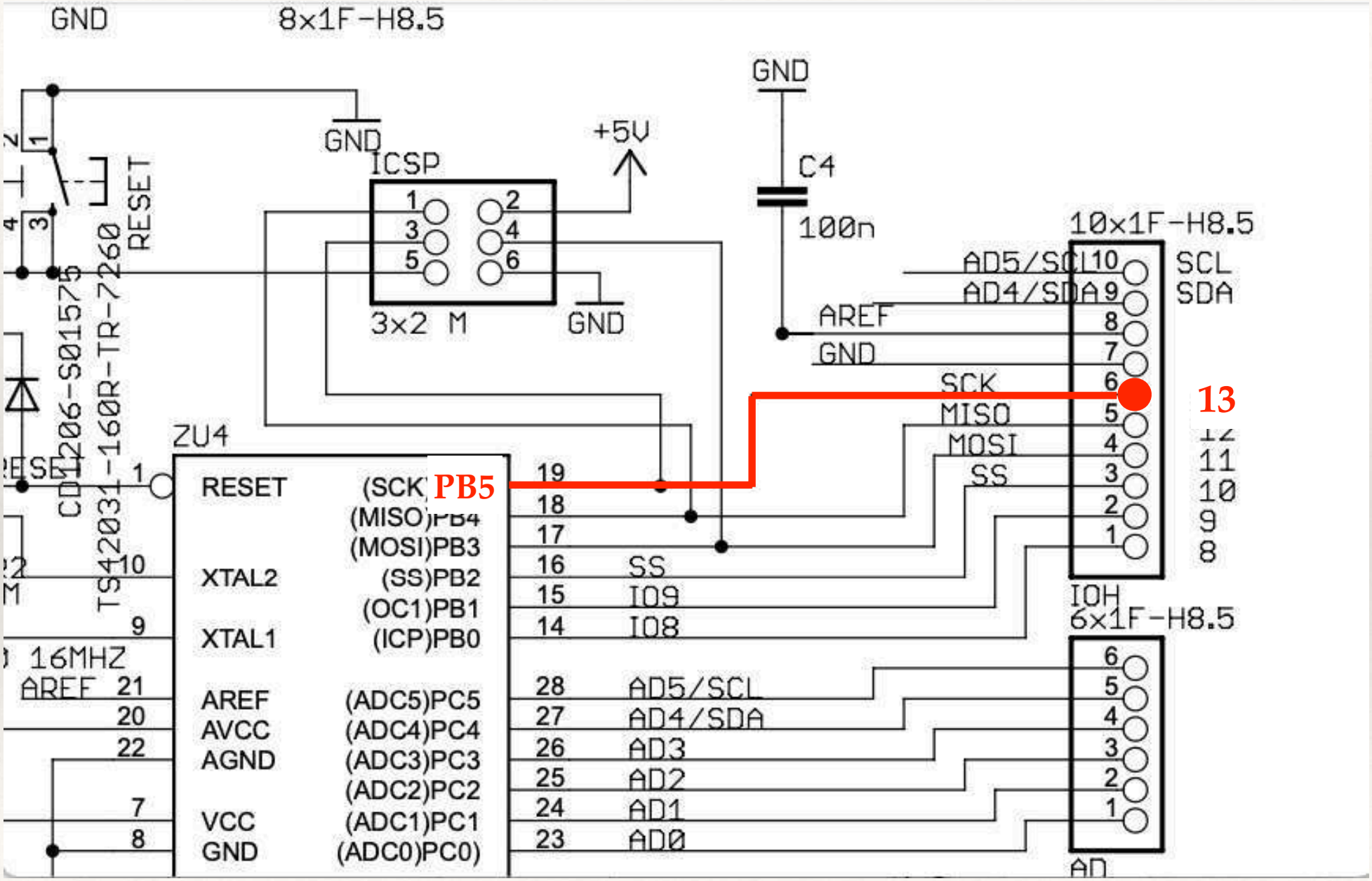


Diagrama esquemático do Arduíno Uno v3

Interrupções (prática)

Temos o mesmo código, mas agora com registradores;

Outra opção é apenas inverter o valor do registrador

PORTB ^= (1 << PB5);

A operação OU-Exclusivo (^) inverte o valor do bit.

Função *delay(1000)* mantém o processador “ocupado" por 1s;

Vamos usar o *timer1* para substituir a função !

Interrupções (prática)

Temos o mesmo código, mas agora com registradores;

Outra opção é apenas inverter o valor do registrador

PORTB ^= (1 << PB5);

A operação OU-Exclusivo (^) inverte o valor do bit.

Função *delay(1000)* mantém o processador “ocupado" por 1s;

Vamos usar o *timer1* para substituir a função !

Preparando o *timer1*

Embora o TCCR1A não seja necessário, ele precisa sofrer reset;

- O Arduíno grava o TCCR1A por padrão para operar o PWM;
- Basta o comando **TCCR1A = 0;**

TCCR1A – Timer/Counter1 Control Register A

Bit	7	6	5	4	3	2	1	0	
(0x80)	COM1A1	COM1A0	COM1B1	COM1B0	–	–	WGM11	WGM10	TCCR1A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Preparando o *timer1*

Uma mudança do LED a cada segundo

Clock de 16MHz = 16 milhões de ciclos por segundo

Contagem máxima do timer1 é 65.536

```
16.000.000 / 64    = 250.000  ⇒ inválido (> contagem máxima)
16.000.000 / 256   =  62.500  ⇒ válido (escolhido) !
16.000.000 / 1.024 =   15.625 ⇒ válido !
```

Prescaler = 256; Contador = 62.500;

Table 15-6. Clock Select Bit Description

CS12	CS11	CS10	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	$clk_{I/O}$/1 (no prescaling)
0	1	0	$clk_{I/O}$/8 (from prescaler)
0	1	1	$clk_{I/O}$/64 (from prescaler)
1	0	0	$clk_{I/O}$/256 (from prescaler)
1	0	1	$clk_{I/O}$/1024 (from prescaler)
1	1	0	External clock source on T1 pin. Clock on falling edge.
1	1	1	External clock source on T1 pin. Clock on rising edge.

Preparando o *timer1*

Definindo o prescaler

TCCR1B |= (1 << CS12);

TCCR1B &= ~(1 << CS11);

TCCR1B &= ~(1 << CS10);

Para usar o contador

Definimos a interrupção pela comparação com o registro A;

TIMSK1 = (1 << OCIE1A);

TIMSK1 – Timer/Counter1 Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0	
(0x6F)	–	–	ICIE1	–	–	OCIE1B	OCIE1A	TOIE1	TIMSK1
Read/Write	R	R	R/W	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Input Capture

Overflow

Preparando o *timer1*

Carga do registro A

Definimos constantes para facilitar o acesso aos registradores de 16 bits:

```
const uint16_t t1_carga = 0;
const uint16_t t1_comparador = 62500;
```

Registradores são carregados dentro da ISR:

```
ISR( TIMER1_COMPA_vect ) {
    TCNT1 = t1_carga;
    PORTB ^= ( 1 << PB5 );
}
```

Também é importante habilitar as interrupções

```
sei();
```

Código Final

Estrutura loop() só possui delay()

- Nenhum código afetará a operação do blink();

ISR() pode ser ainda menor

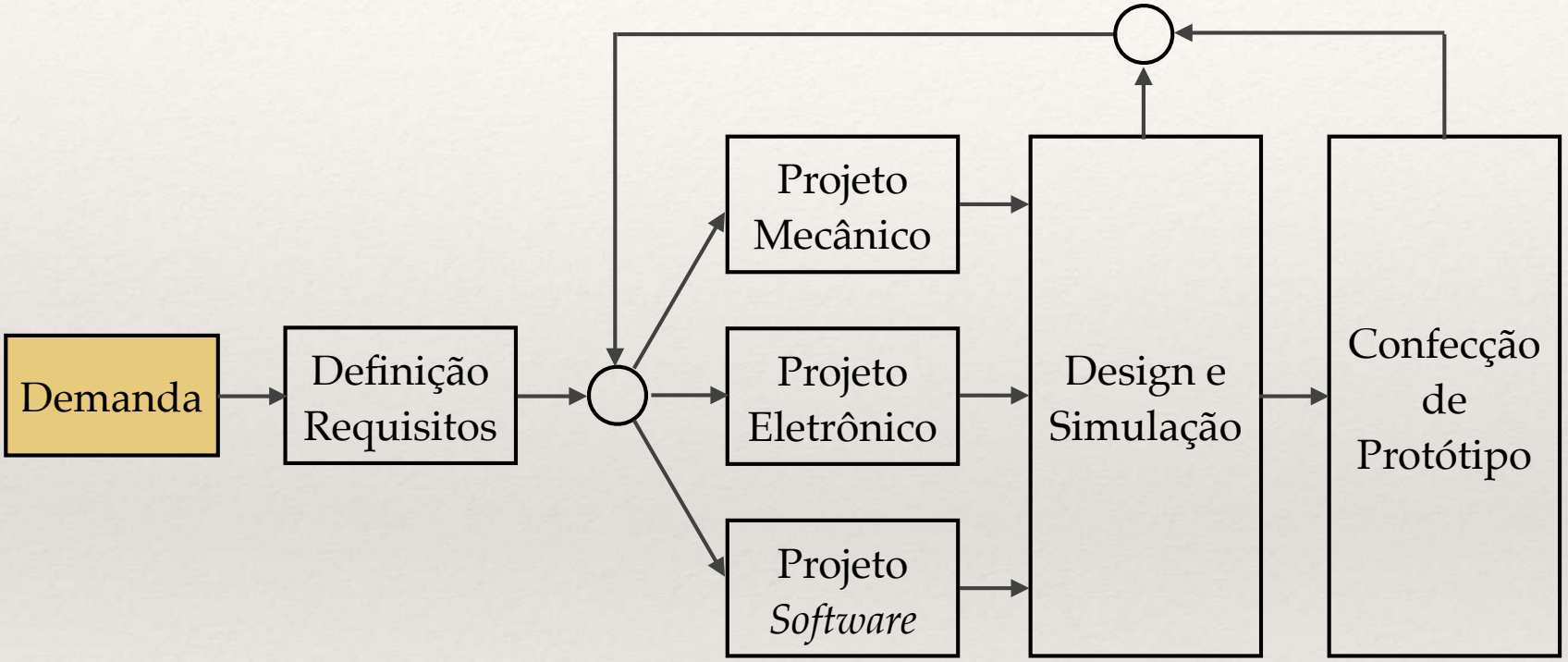
- Modo CTC pode ser habilitado;
- *timer1* zera ao alcançar registro A;
- Objetivo é fazer ISR() o menor possível.

BlinkComInterrERegistr

```
const uint16_t t1_carga = 0;
const uint16_t t1_comparador = 62500;

void setup() {

  // Define o pino 13 como OUTPUT
  DDRB |= (1 << PB5 );

  // Impõe RESET ao registro A de controle do timer1
  TCCR1A = 0;

  // Define o prescaler em 256
  TCCR1B |= (1 << CS12 );
  TCCR1B &= ~(1 << CS11 );
  TCCR1B &= ~(1 << CS10 );

  // Zera o timer1, e define o valor para comparação
  TCNT1 = t1_carga;
  OCR1A = t1_comparador;


  // Define a interrupção por comparação com o registro A
  TIMSK1 = (1 << OCIE1A );

  sei();

}

void loop() {

  delay(1000);

}

ISR( TIMER1_COMPA_vect ) {

  TCNT1 = t1_carga;

  // Ativa a porta PB5
  PORTB ^= (1 << PB5 );

}
```

(*) Código inspirado no material da SparkFun “Level Up Your Arduíno Code: Timer Interrupts”

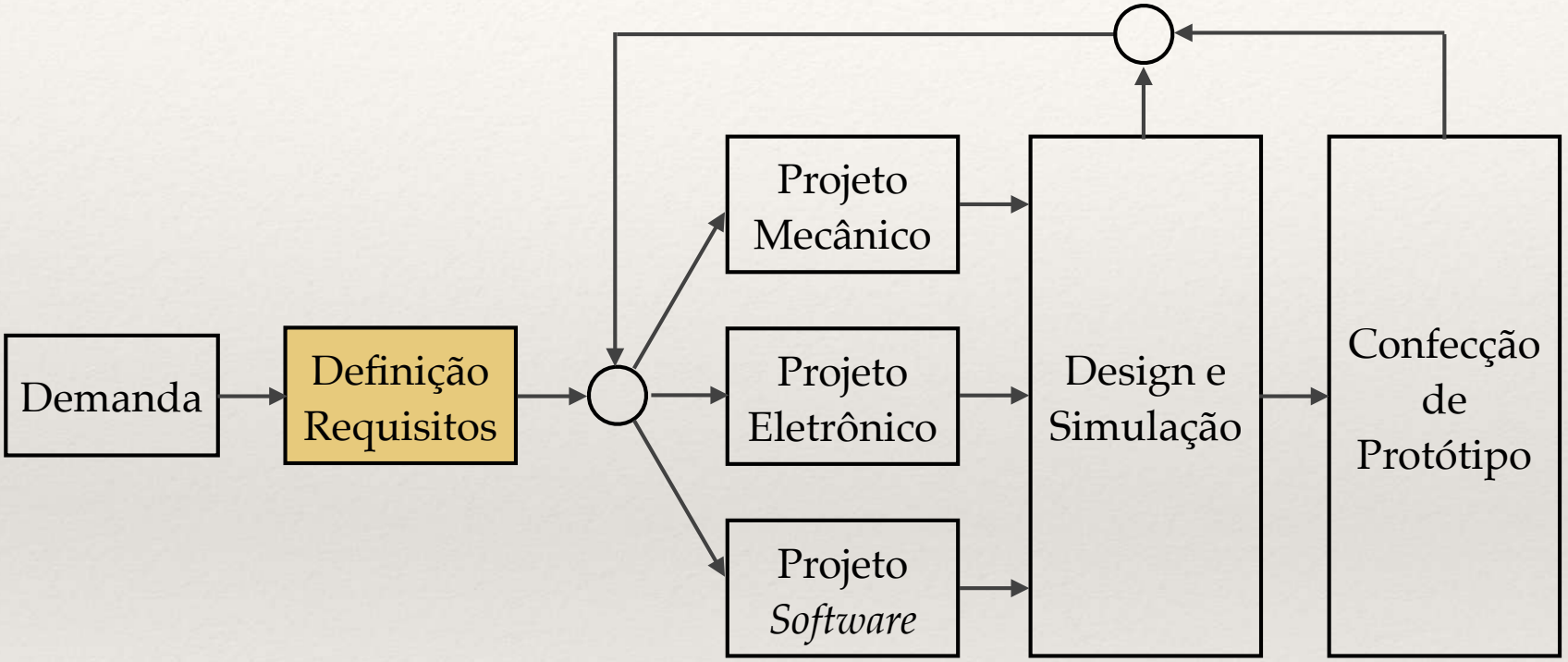
Fases do Projeto de SE



A demanda por SE hoje está em todas as áreas

Esta ficando difícil encontrar um equipamento meramente mecânico ou eletrônico;
Sistemas embarcados melhoram as funcionalidades, aumentam a vida útil, reduzem custo;
Caminho similar ao da informatização, da internet, e agora da IA.

Fases do Projeto de SE



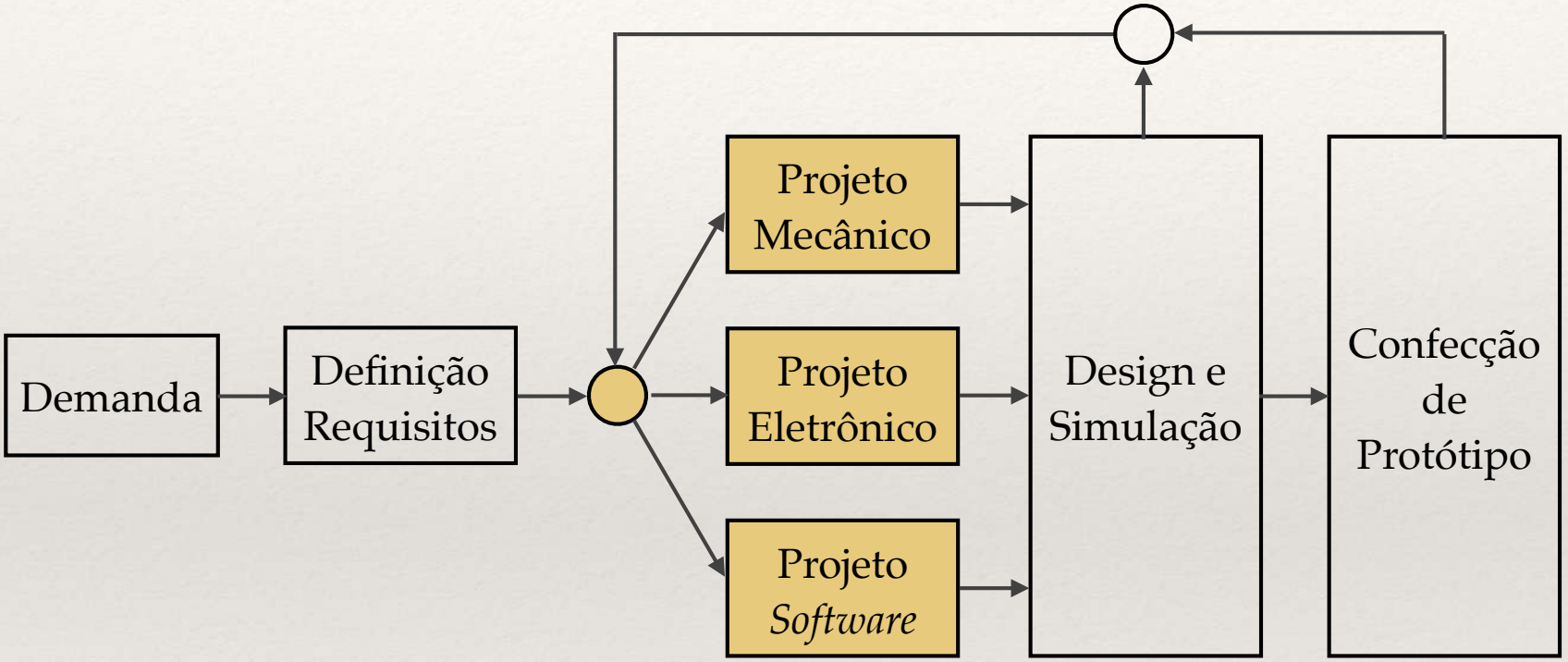
Definição de requisitos

Envolve diversas disciplinas;

Exige um trabalho bem cuidadoso, para garantir longevidade ao equipamento;

Pode mudar inclusive durante o ciclo de vida do produto (atualização de *software*).

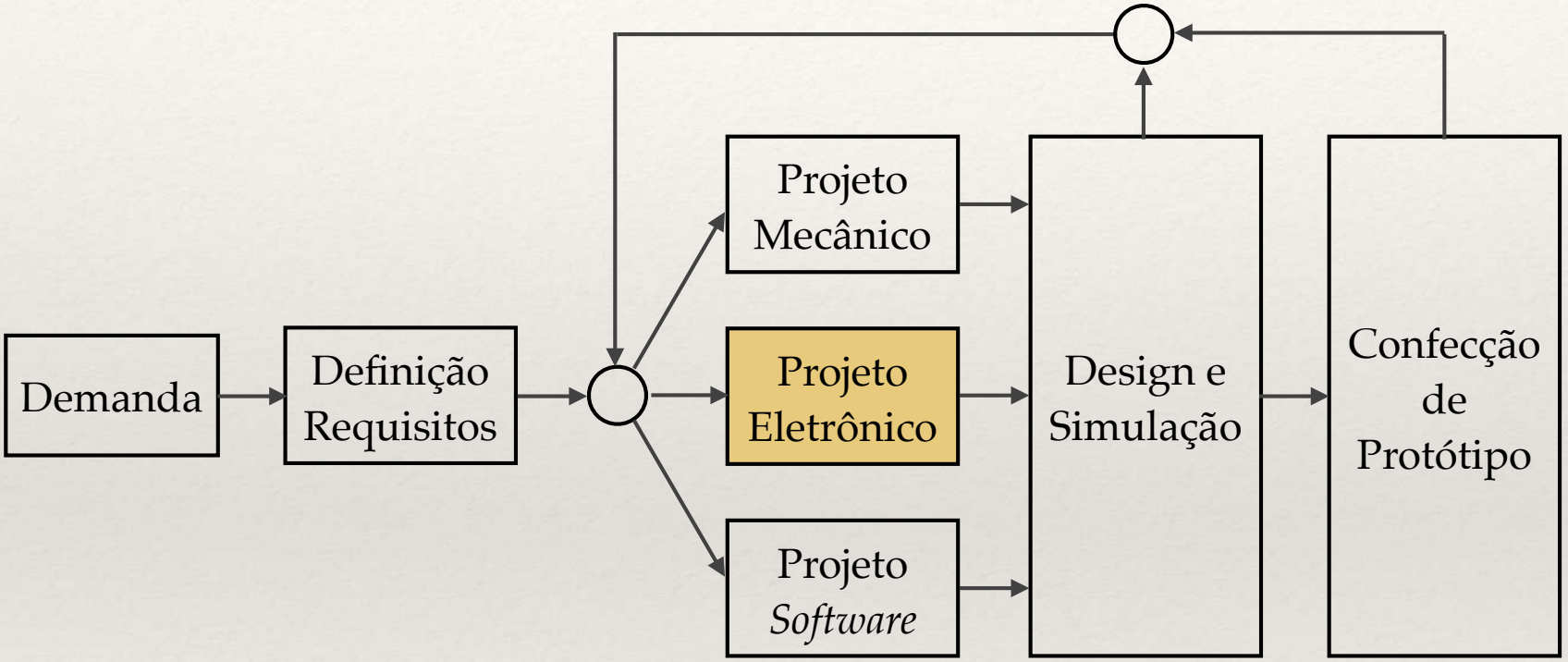
Fases do Projeto de SE



Projetos multidisciplinares

Podem inclusive envolver outras áreas (química, biofísica, agronomia, medicina etc);
Desenvolvidos de forma virtual com auxílio de ferramentas CAx (CAD, CAM, CAE etc);
Há forte interação entre projetos (espaço ocupado, peso, consumo de energia etc).

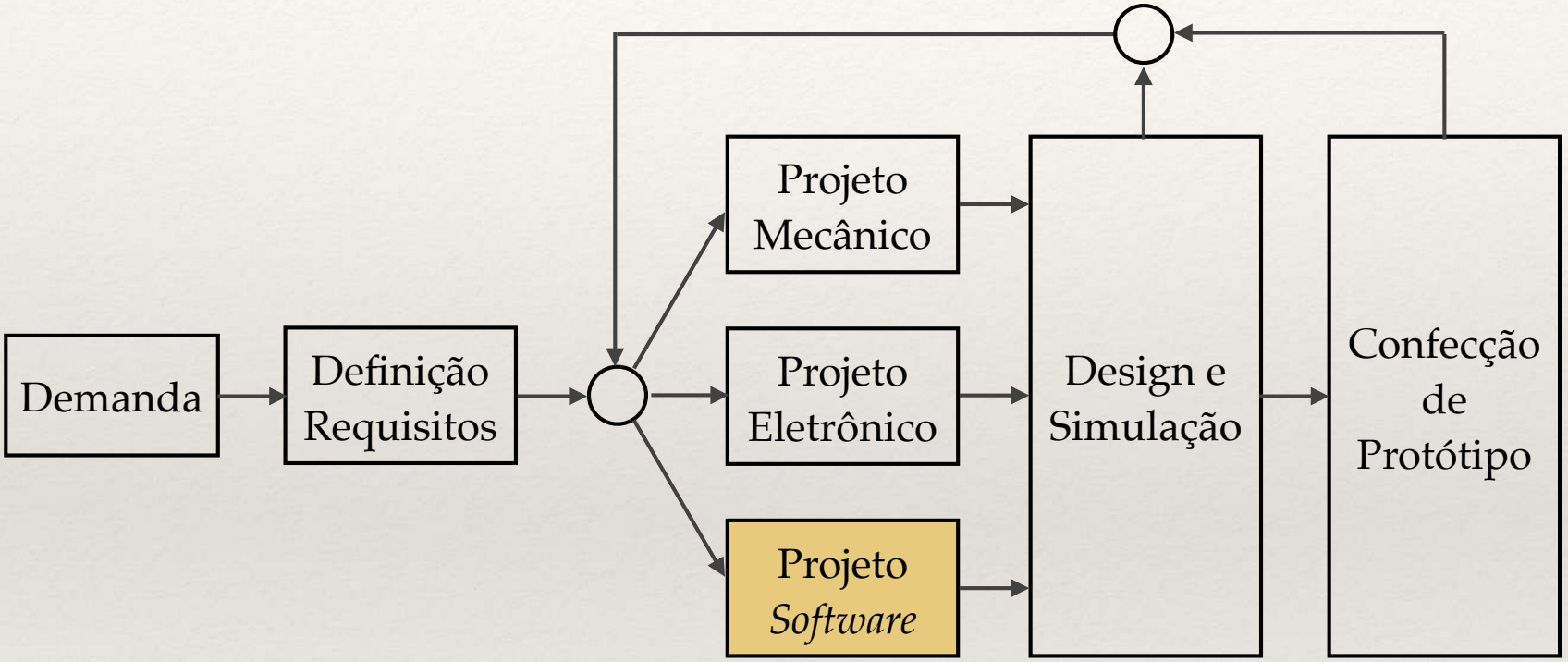
Fases do Projeto de SE



Projeto Eletrônico

Também é multidisciplinar (elétrica, eletrônica, microcontroladores, sensores, controle etc);
Envolve definição da plataforma (microcontrolador?), sensores, atuadores e componentes;
Integração forte com o projeto de *software*.

Fases do Projeto de SE



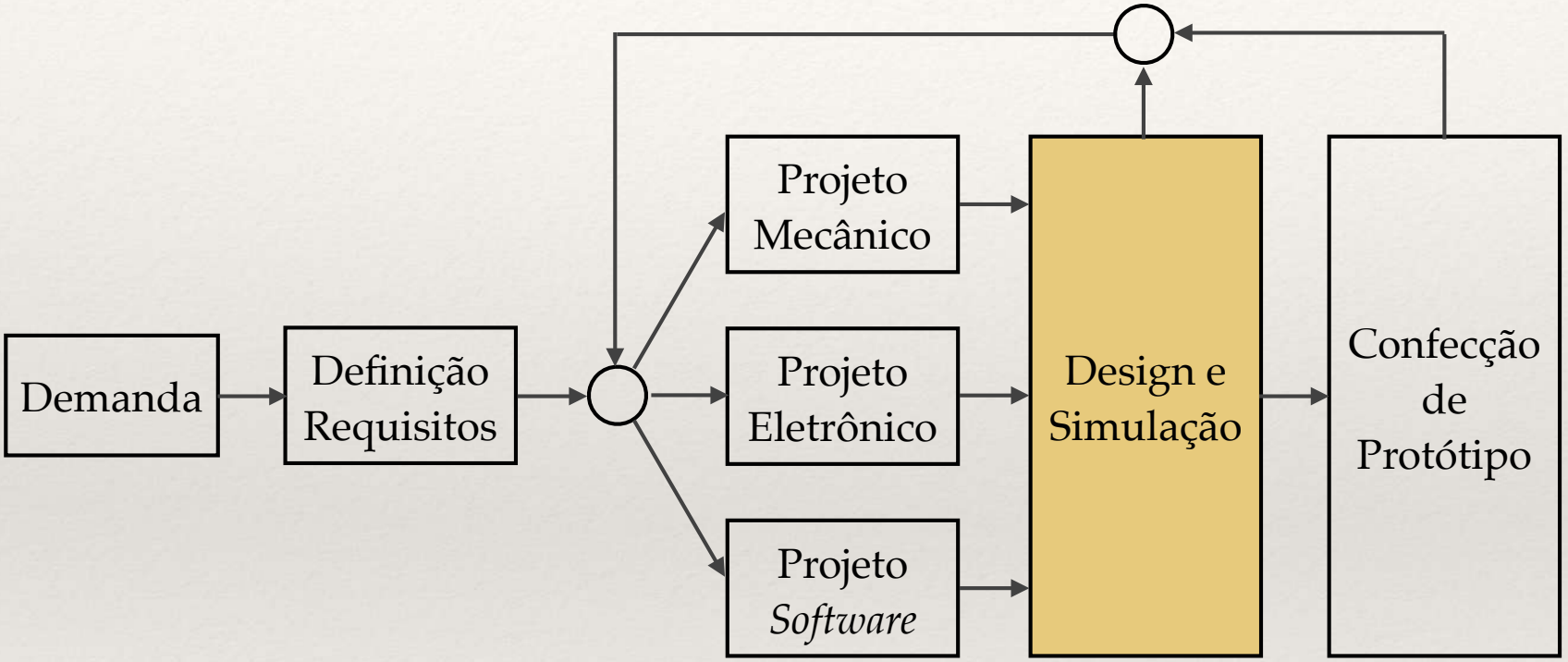
Projeto *Software*

Fortemente afetado pelos requisitos do projeto, e escolhas no projeto eletrônico;

Exige definições e sofre efeitos bem diferentes dos demais projetos;

É o nosso objeto de estudo nesta aula.

Fases do Projeto de SE



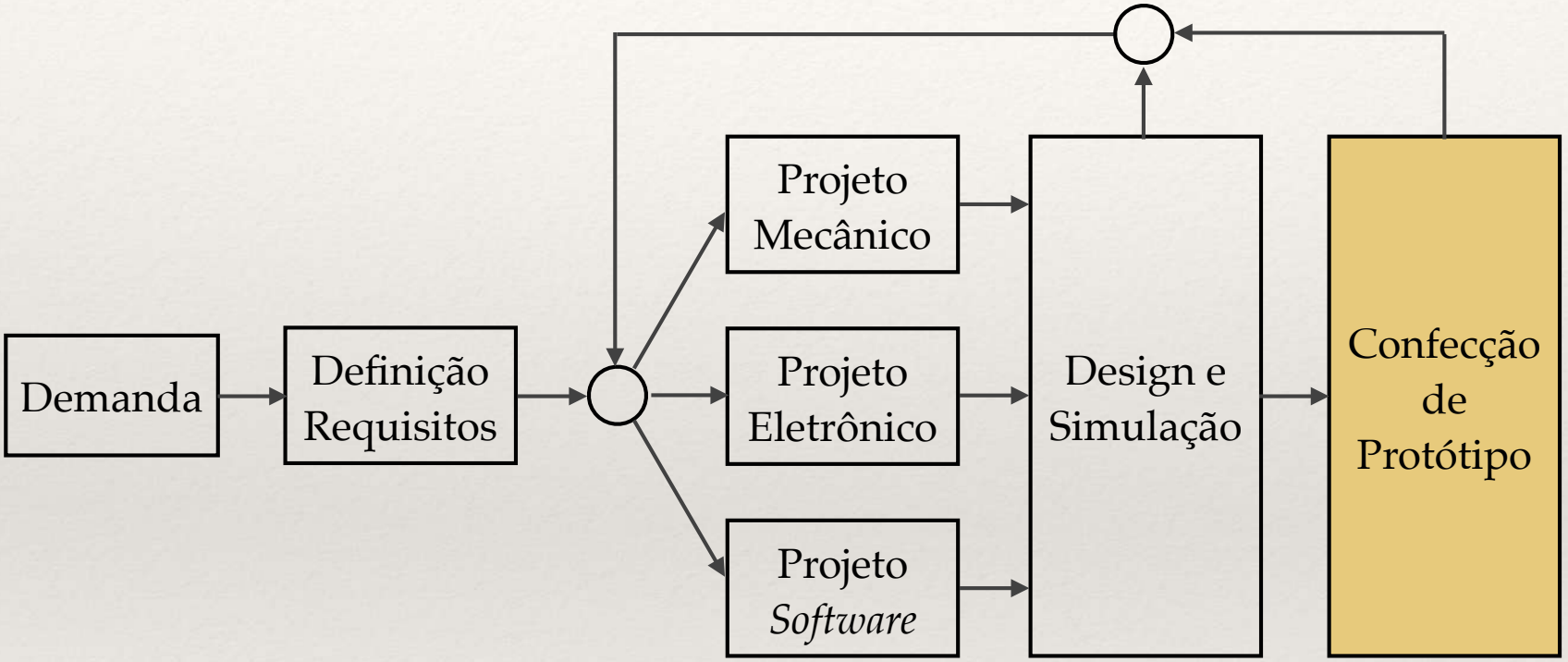
Design e Simulação

Há riqueza de ferramentas de *software* para apoio nesta etapa de projeto;

Tipicamente é possível simular a maior parte dos componentes do projeto;

Comportamentos observados na simulação podem afetar o projeto, exigindo ajustes.

Fases do Projeto de SE

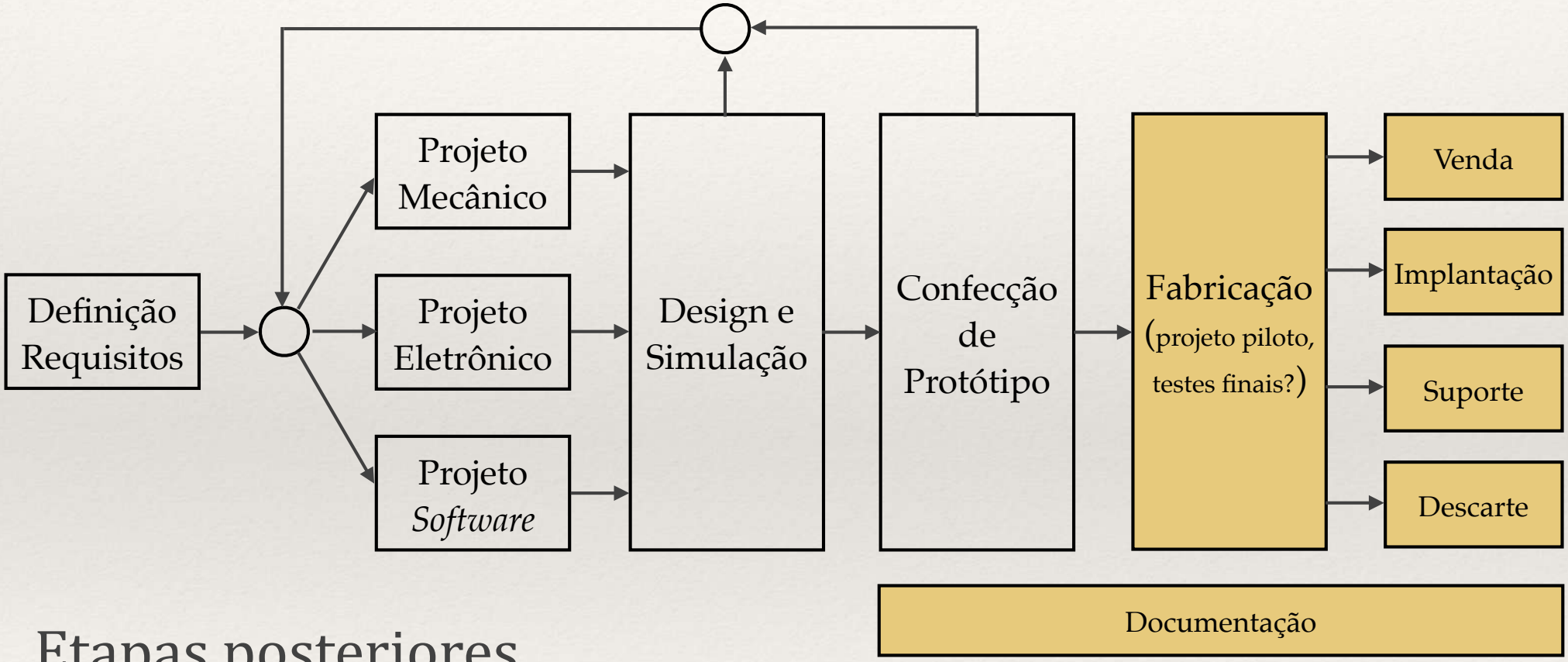


Confecção de Protótipo

Tipicamente, a confecção de um modelo físico e operacional do projeto é essencial;

O comportamento do modelo ainda pode afetar o projeto, exigindo modificações.

Fases do Projeto de SE



Etapas posteriores

Obviamente o trabalho não acaba no protótipo;

Um projeto exige muitas outras preocupações, que podem inclusive afetar as primeiras etapas

Software em SE

Quem executará o *software*?

Definição tipicamente ocorre durante o projeto de *hardware*;

Soluções proprietárias (PLCs);

Software em SE

Quem executará o *software*?

Definição tipicamente ocorre durante o projeto de *hardware*;

Os dez maiores fabricantes:

	Fabricantes	Marca
1	Siemens	Simatic
2	Rockwell Automation	Allen Bradley
3	Mitsubishi Electric	Melsec
4	Schneider Electric	Modicon
5	Omron	Sysmac
6	Emerson Electric (GE)	RX3i & VersaMax (GE Fanuc)
7	Keyence	KV & V-8000
8	ABB (B&R Automation)	AC500 X20 & X90
9	Bosch	Rexroth ICL
10	Hitachi	EH & H

https://ladderlogicworld.com/plc-manufacturers/#google_vignette

Software em SE

Norma IEC 61131-3 (1993 ~ 2013)

- Promover interoperabilidade entre diferentes fabricantes de PLCs;
- Define, entre outras coisas, 5 linguagens de programação padrão
  - LD (*Ladder Diagram*) - gráfica;
  - FBD (*Function Block Diagram*) - gráfica;
  - ST (*Structured Text*) - textual;
  - IL (*Instruction List*) - textual (descontinuada);
  - SFC (*Sequencial Function Chart*) - gráfica.

Tecnologia, no entanto, continua fechada

- Não há acesso ao *software*, nem à arquitetura de *hardware*;
- Restrições levaram à busca por plataformas abertas;
- Motivação para estudo dos microcontroladores em sistemas de automação, ou seja, nossa disciplina !

Software em SE

Microcontroladores utilizados em PLCs

Em sistemas menores, linha PIC e ARM Cortes M3 ou M4, ou FPGAs;

Em sistemas de grande porte, ARM Cortes A8 e A9, e microprocessadores.

Tecnologia aberta

Promove a redução significativa de custos;

Garante flexibilidade, por não depender de soluções proprietárias.

OpenPLC - uma iniciativa de tecnologia aberta

Iniciativa do projeto de doutorado de Thiago Rodrigues Alves

Suporta as 5 linguagem padrão definidas pela IEC 61131-3

Pode ser executada nas plataformas PIC, Arduíno, ESP, Raspberry Pi etc.

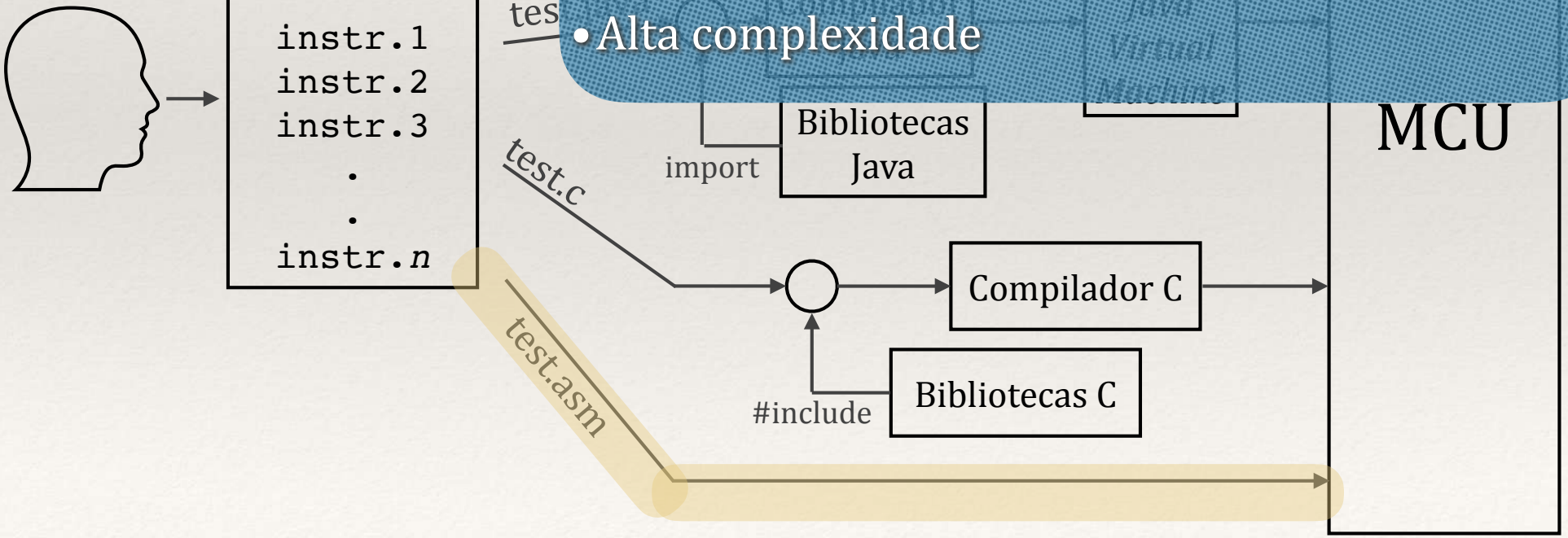
https://autonomylogic.com/

https://www.youtube.com/watch?v=cNg-tXZynJE

Programa “aberto" MCUs

Programação *Assembly*

- Não há necessidade de compilação, exceto para conversão dos mnemônicos em instruções de linguagem de máquina com o "Assembler";
- Maior performance e menor código possível;
- Baixíssima portabilidade;
- Alta complexidade



Conjunto de Instruções

Mnemonic, Operands		Description	Cycles	14-Bit Instruction Word MSb			LSb	Status Affected
BYTE-ORIENTED FILE REGISTER OPERATIONS								
ADDWF	f, d	Add W and f	1	00	0111	dfff	ffff	C,DC,Z
ANDWF	f, d	AND W with f	1	00	0101	dfff	ffff	Z
CLRF	f	Clear f	1	00	0001	lfff	ffff	Z
CLRW	-	Clear W	1	00	0001	0xxx	xxxx	Z
COMF	f, d	Complement f	1	00	1001	dfff	ffff	Z
DECF	f, d	Decrement f	1	00	0011	dfff	ffff	Z
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	00	1011	dfff	ffff	
INCF	f, d	Increment f	1	00	1010	dfff	ffff	Z
INCFSZ	f, d	Increment f, Skip if 0	1(2)	00	1111	dfff	ffff	
IORWF	f, d	Inclusive OR W with f	1	00	0100	dfff	ffff	Z
MOVF	f, d	Move f	1	00	1000	dfff	ffff	Z
MOVWF	f	Move W to f	1	00	0000	lfff	ffff	
NOP	-	No Operation	1	00	0000	0xx0	0000	
RLF	f, d	Rotate Left f through Carry	1	00	1101	dfff	ffff	C
RRF	f, d	Rotate Right f through Carry	1	00	1100	dfff	ffff	C
SUBWF	f, d	Subtract W from f	1	00	0010	dfff	ffff	C,DC,Z
SWAPF	f, d	Swap nibbles in f	1	00	1110	dfff	ffff	
XORWF	f, d	Exclusive OR W with f	1	00	0110	dfff	ffff	Z
BIT-ORIENTED FILE REGISTER OPERATIONS								
BCF	f, b	Bit Clear f	1	01	00bb	bfff	ffff	
BSF	f, b	Bit Set f	1	01	01bb	bfff	ffff	
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	01	10bb	bfff	ffff	
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	01	11bb	bfff	ffff	

PIC16F877 Instruction Set https://ww1.microchip.com/downloads/en/DeviceDoc/31029a.pdf

Conjunto de Instruções

Define instruções em “linguagem de máquina” reconhecidas pelo dispositivo

- A programação tipicamente é realizada em “assembly”, que representa cada instrução com uma sigla (mneumônico);
- Veremos outras formas mais simples de programar um microcontrolador ao longo da disciplina.

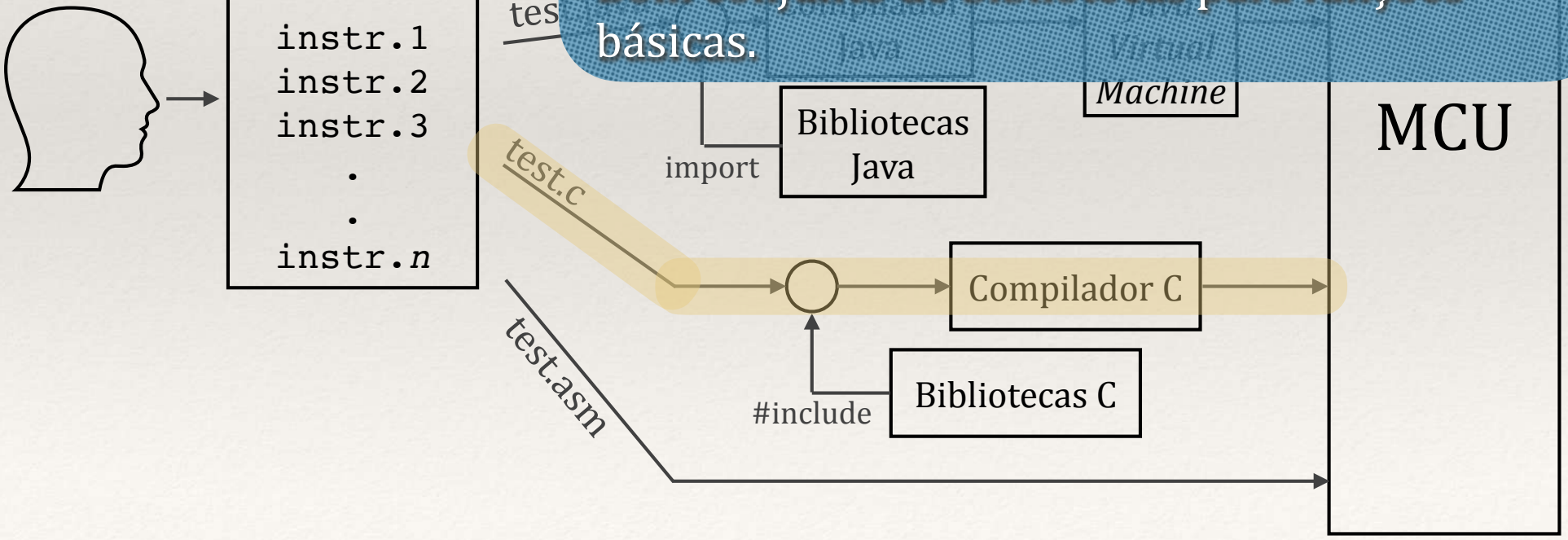
Arquiteturas RISC x CISC

- *(Reduced / Complete) Instruction Set Computer*;
- O PIC16F877, por exemplo, possui apenas 35 instruções (RISC), divididas em seis grupos;
- RISC tipicamente oferece performance maior, porém traz desafios para a programação.

Programa “aberto" MCUs

Programação C, C++

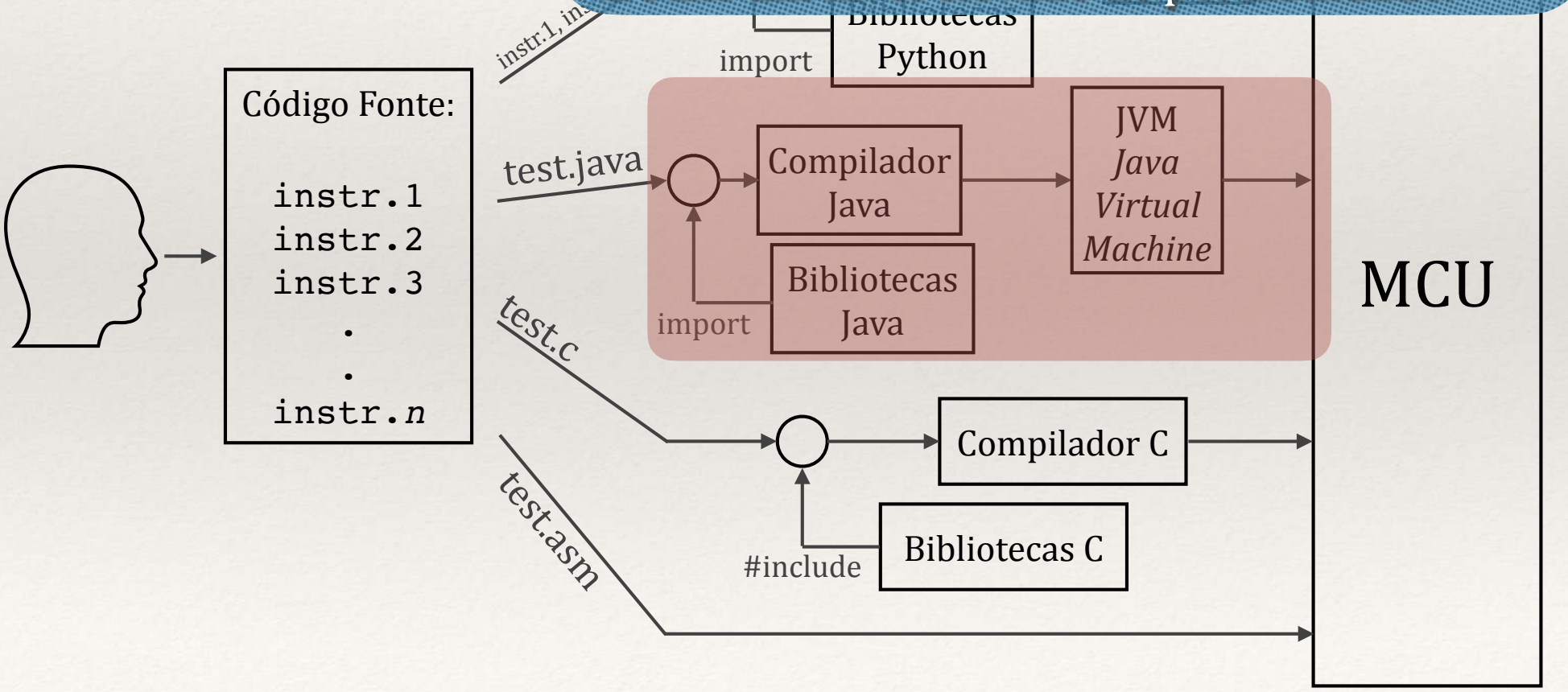
- Boa performance e código reduzido;
- Boa portabilidade, embora aceite código específico;
- Complexidade baixa;
- Bom conjunto de bibliotecas para funções básicas.



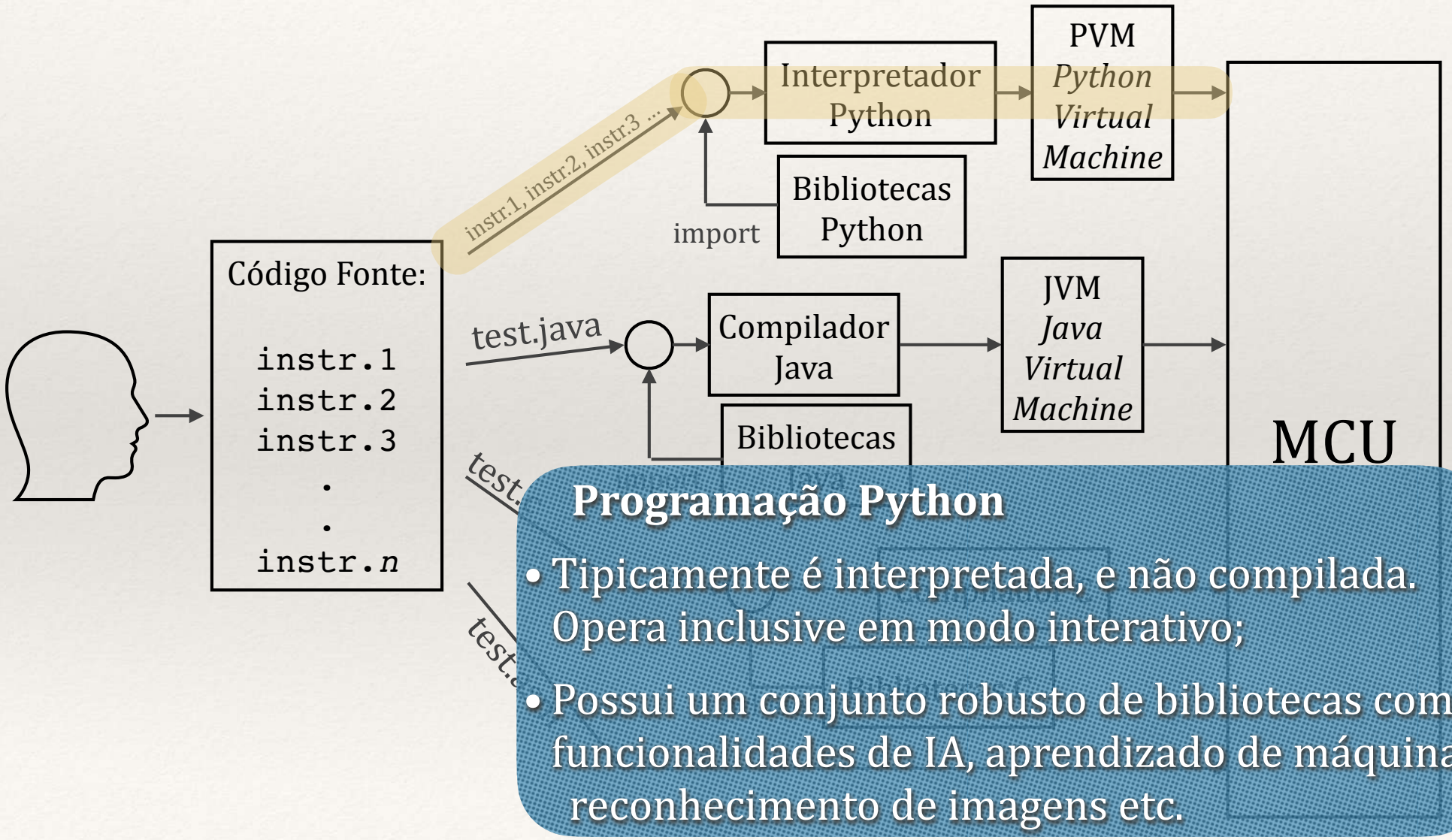
Programa "aberto" MCUs

Programação Java

- Incomum na programação de SE;
- Incorpora o conceito de Máquina Virtual, que trouxe um nível elevado de portabilidade.



Programa “aberto" MCUs



Compiladores especiais

Cross Compilers

Criam códigos executáveis para dispositivos diferentes de onde o compilador foi executado;

Tipicamente utilizado na programação de MCUs e outros dispositivos que não suportam adequadamente a operação de um compilador;

Também são utilizados na programação de celulares, *smartwatches* etc.

Source-to-source Compilers

Transforma um código fonte escrito em uma linguagem de programação para outra linguagem;

Existem exemplos para converter LD (*Ladder Diagram*) para C++, por exemplo.

Software em SE

Parâmetros importantes a serem avaliados

Performance: quanto mais baixo o nível, mais rápido;

Ocupação de Memória: quanto mais baixo o nível, menor ocupação;

Grau de abstração: quanto mais alto o nível, maior a abstração;

Portabilidade: linguagens de alto nível e Máquinas Virtuais ajudam;

Reuso & Bibliotecas: maior a participação de mercado ajuda;

Documentação e recursos de apoio: idem acima.

Cabe ao projetista a análise

Custo e tempo de desenvolvimento/testes;

Direitos autorais.

Outros componentes de *SW*

Software Básico

Bootloader

- Nos computadores, carrega o SO da unidade de disco;
- Software de inicialização, tipicamente associado a uma placa de protótipo;
- Configuração Básica de módulos internos (Wi-Fi, UART etc)
- Permite a carga de sketches;

Sistema Operacional

- Tipicamente ausente nos MCUs, pode ser encontrado nos SBC (*Single Board Computers*);
- Incorpora funcionalidades importantes, porém consome recursos para sua execução.

RTOS (*Real Time Operational System*)

- Para microcontroladores, especialmente em aplicações multitarefa e em tempo real;
- Trata problemas de compartilhamento de recursos, sincronismo e priorização.

Integração com recursos na nuvem

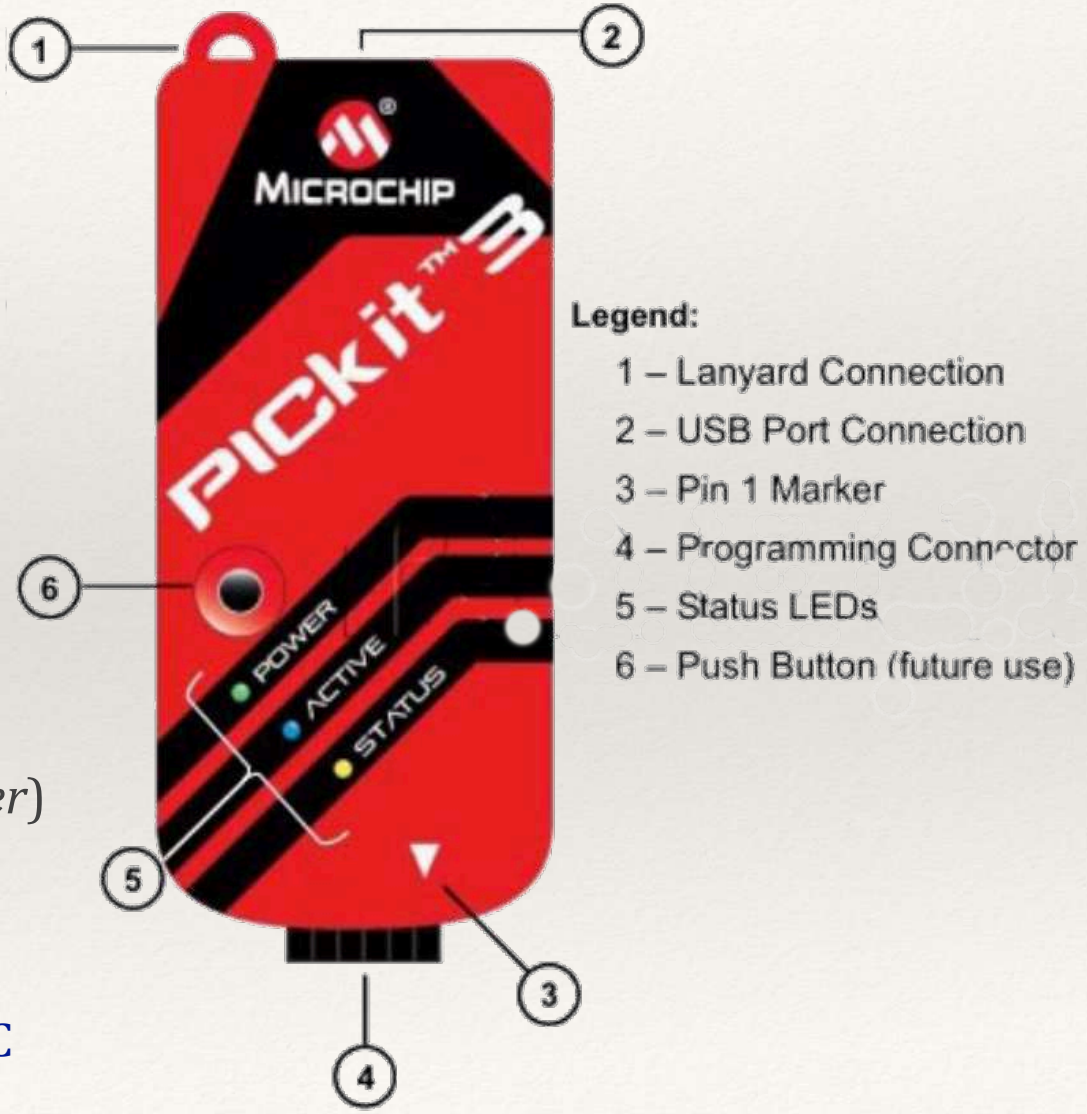
Diversos recursos são oferecidos na nuvem, especialmente para IoT.

Carga do SW

Microcontroladores, mesmo instalados em placas de protótipo, não possuem interfaces acessíveis diretamente pelo programador;

Há diferentes formas de carregar o *software*:

- Acesso direto via USB (placas de protótipo);
- Acesso via *device programmer* (*)
- Acesso via ISP (*In-Circuit Programmer*)



(*) Na foto, o PICkit 3, *device programmer* popular para microcontroladores da linha PIC

Via USB

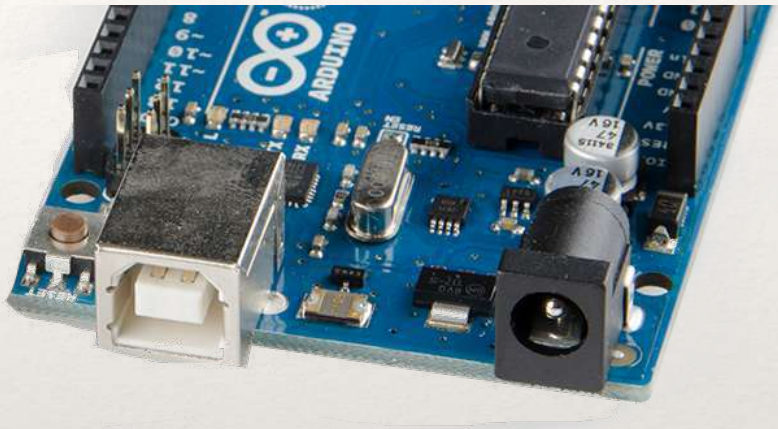
Algumas placas de protótipo oferecem portas USB integradas

- Linha Arduíno;
- ESP8266, ESP32

Algumas oferecem conexões para adaptadores USB externos

- Ex.: ESP01 (ao lado)

Basta conectar a um computador com uma IDE instalada.



Via *Device Programmer*

Permite conexão direta do *chip*

- Diretamente;
- Através de protoboard, ou na placa de protótipo.

Fabricantes oferecem diversas opções

- PICkit, ATMega *Programmer* etc

Via ISP

Utilize uma placa de protótipo como bridge

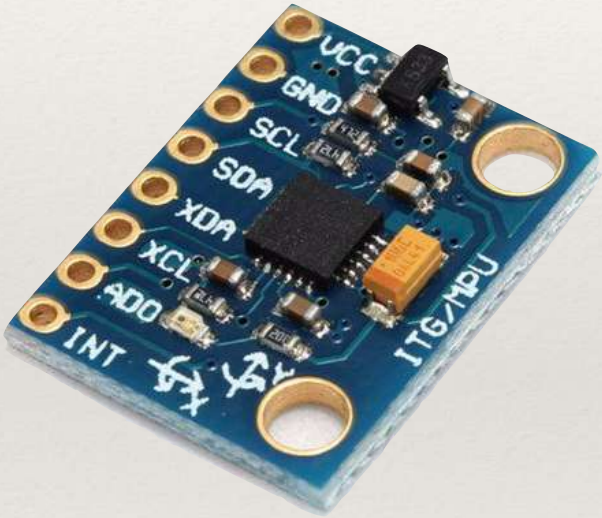
- Arduíno Uno;
- IDE oferece opção nativa.

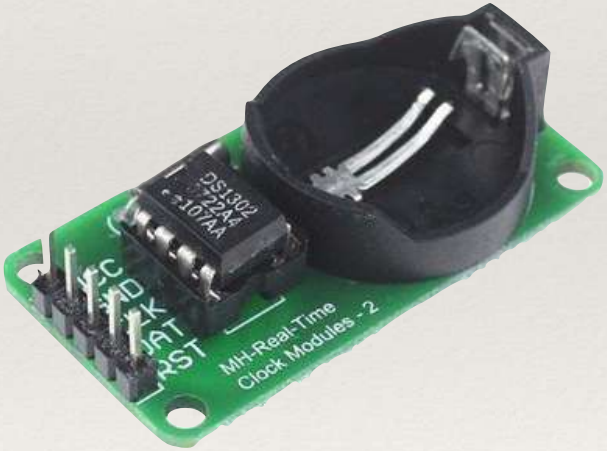
Interligação é feita através de portas SPI.

Aula 12

Dispositivos de Entrada

Dispositivos especiais (giroscópio, acelerômetro, RTC)





Comunicação em SE

Em Sistemas Embarcados, microcontroladores e sensores precisam se comunicar

- Sinais elétricos (portas analog./digitais);
- Interfaces cabeadas;
- Interfaces sem fio.

Há diversos padrões internacionais

UART	ZigBee	Bluetooth
i2C	Ethernet	CAN
SPI	Wi-Fi	LoRa

Também há soluções proprietárias

- Ex. ESP-Now

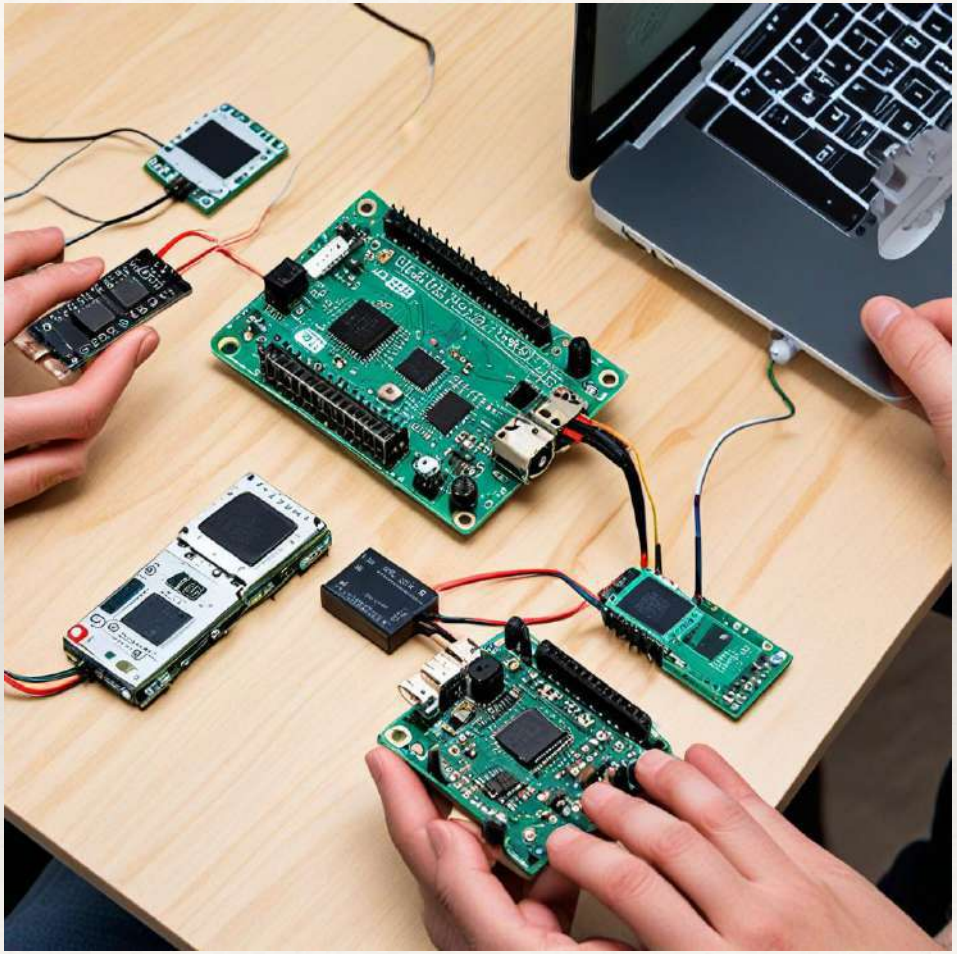


Ilustração gerada no imagine.art (IA)

Comunicação - Protocolos

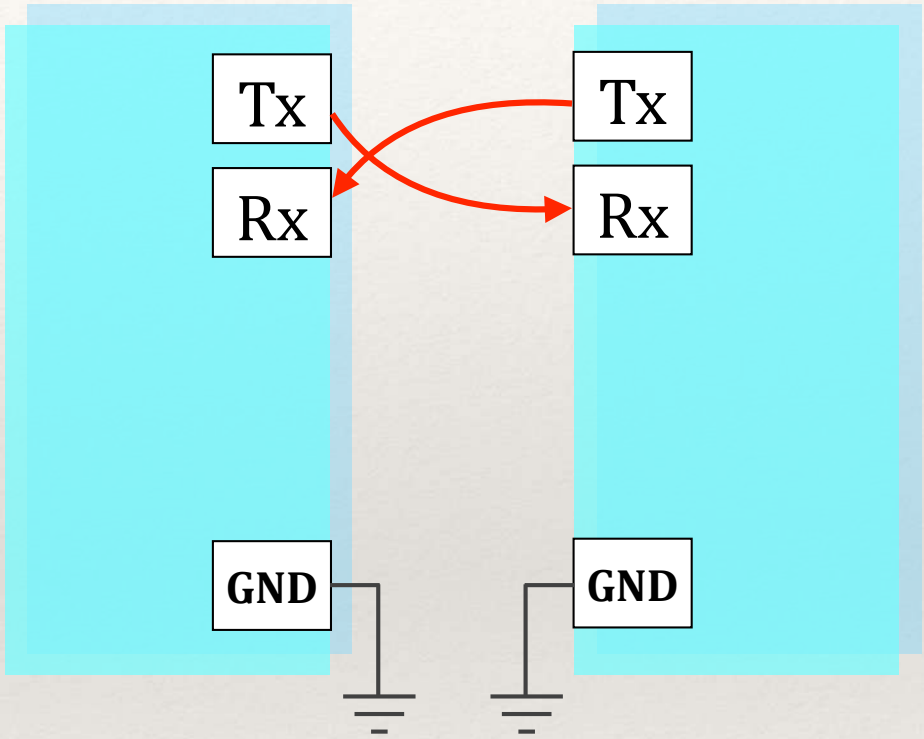
Protocolo	Mídia	Taxa	Alcance	Aplicação Típica
UART	Cabeada	< 5Mbps	1 Km	Muito simples, 2 fios, baixa performance, dois pontos, assíncrono, full-duplex, compatibilidade universal.
i2C	Cabeada	< 5Mbps	100 m	Simples, barramento 2 fios, baixa performance, até 1008 pontos, síncrono, Half-duplex, múltiplos controladores.
SPI	Cabeada	< 65Mbps	10 m	Complex.Média, barramento 4 fios, alta performance, “n" pontos (?), síncrono, full-duplex, 1 controlador.
ZigBee	Sem fio	< 250 Kbps	100 m	Simples, topologias estrela, árvore ou *mesh*, baixa performance, nº pontos (topologia ?) e baixo consumo.
CAN	Cabeada	< 1 Mbps	1 Km	Complex.Média, barramento 2 fios, média performance, “n" pontos, assíncrono, full-duplex, 1 controlador.
LoRa	Sem fio	< 5 Kbps	10 Km	Simples, topologia estrela ou *mesh*, baixíssima performance, “n” pontos, baixo consumo.
ESP-Now	Sem fio	< 1 Mbps	480 m	Simples, topologia estrela ou ponto a ponto, alta performance, 20 pontos, baixo consumo.

A opção UART

É um protocolo clássico, base de todas as comunicações seriais, como o RS-232, por exemplo;

É assíncrono (não utiliza *clock*);

Simplex, *Half* ou *Full-duplex*;



A opção UART

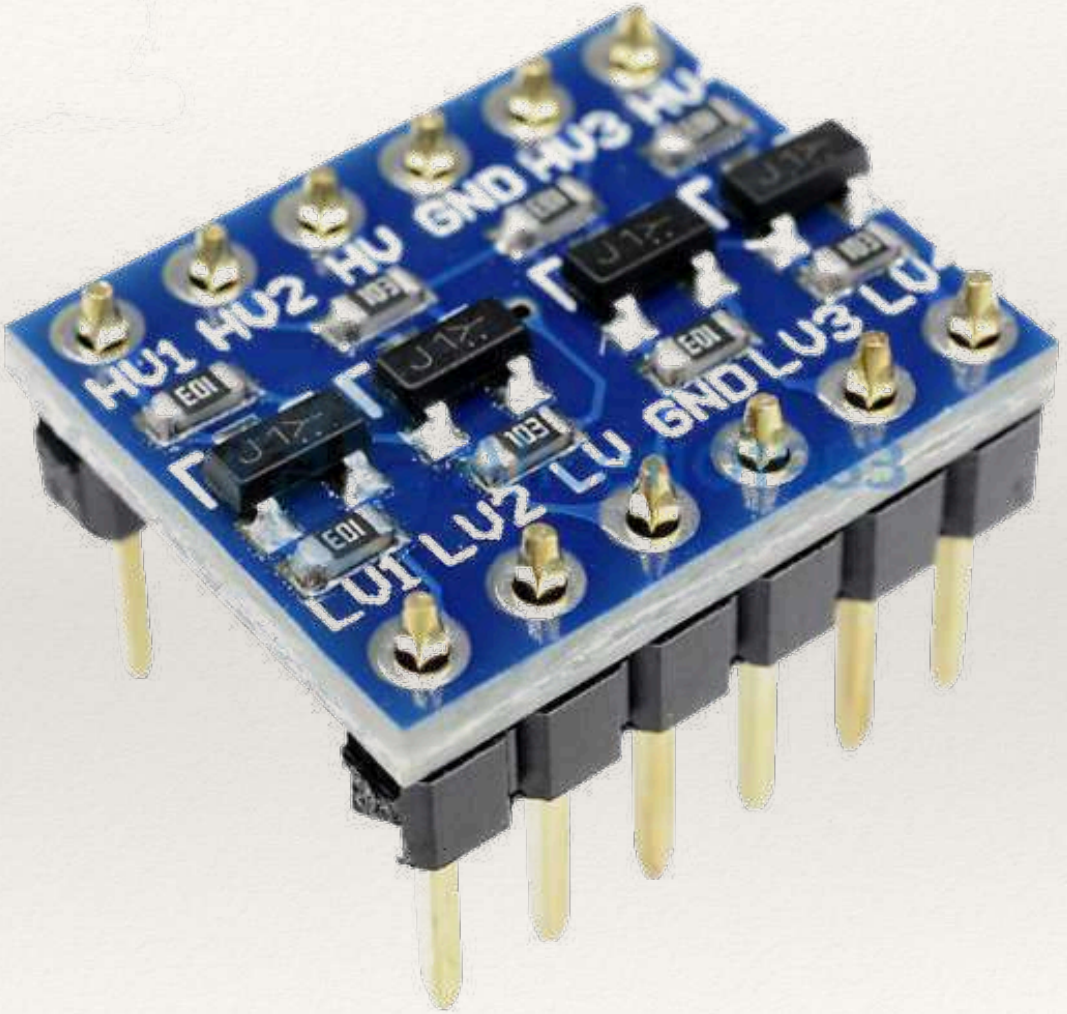
É um protocolo clássico, base de todas as comunicações seriais, como o RS-232, por exemplo;

É assíncrono (não utiliza *clock*);

Simplex, *Half* ou *Full-duplex*;

Compatibilizar níveis de tensão

- Microcontroladores e sensores nem sempre usam o mesmo nível de tensão;
- 3,3 V → 5V : ok ?
- 5V → 3,3V : não interligar !
- Divisores de tensão ou conversores de nível (foto) podem ser utilizados.



A opção UART

É um protocolo clássico, base de todas as comunicações seriais, como o RS-232, por exemplo;

É assíncrono (não utiliza *clock*);

Simplex, *Half* ou *Full-duplex*;

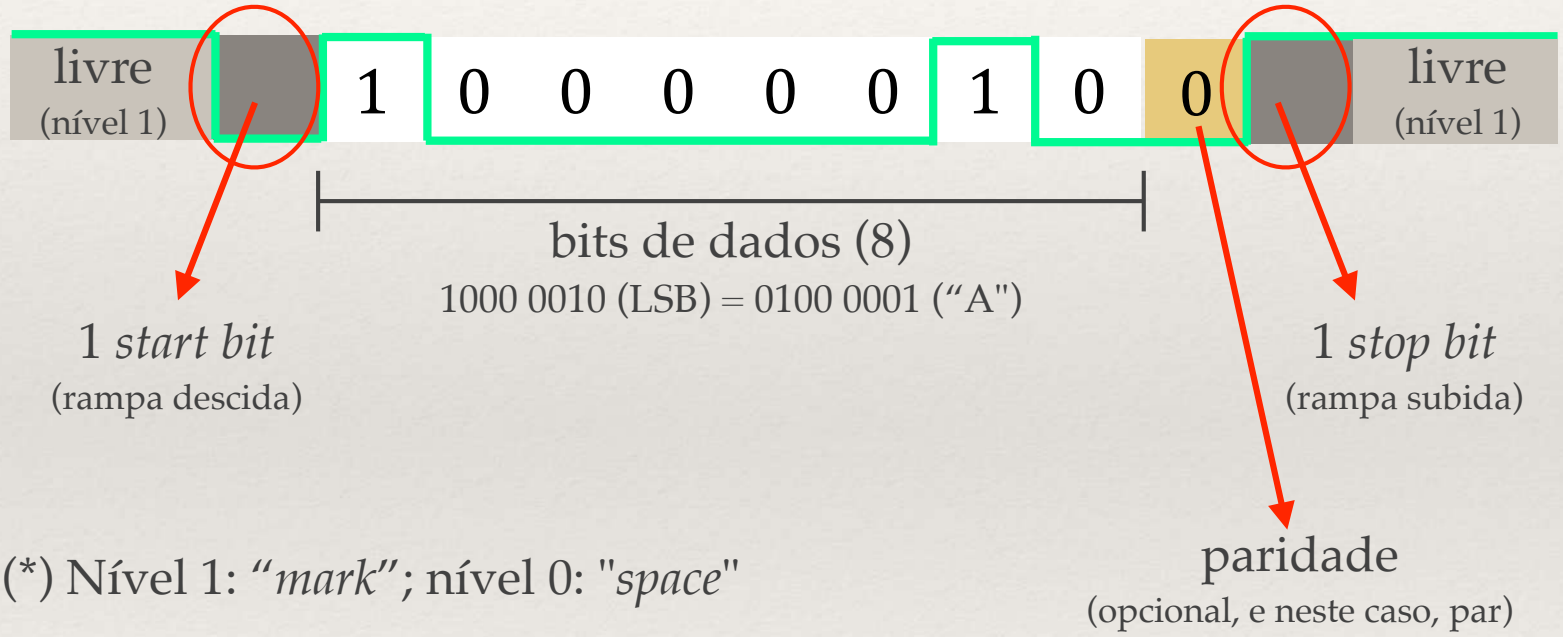
Compatibilizar níveis de tensão;

São 3 as informações de configuração:

- Taxa de transferência;
- Comprimento em bits (5 a 9, típico 7-8);
- *start* e *stop* bits;

Taxas Típicas
4.800 bps
9.600 bps
19.200 bps
57.600 bps
115.200 bps

UART Formato de quadro



(*) Nível 1: “*mark*”; nível 0: "*space*"

UART Programação

Tomando como exemplo o Arduíno, temos os seguintes procedimentos para transmissão e recepção:

Transmissão:

Funciona como uma impressão serial;

O processo envolve:

```
Serial.begin( taxa,[configuração] )
Função Serial.print p/ transmissão.
```

Recepção:

Varredura frequente do *buffer* de entrada, ou por interrupção.

O processo envolve:

```
Serial.begin( taxa,[configuração] )
Verificar Serial.available()
Armazenar mensagem em variável
inChar = Serial.read()
```

(*) Configuração Padrão: 8 bits de dados, 1 de parada, sem paridade (SERIAL_8N1)

A opção I²C

Protocolo criado pela Philips em 1982

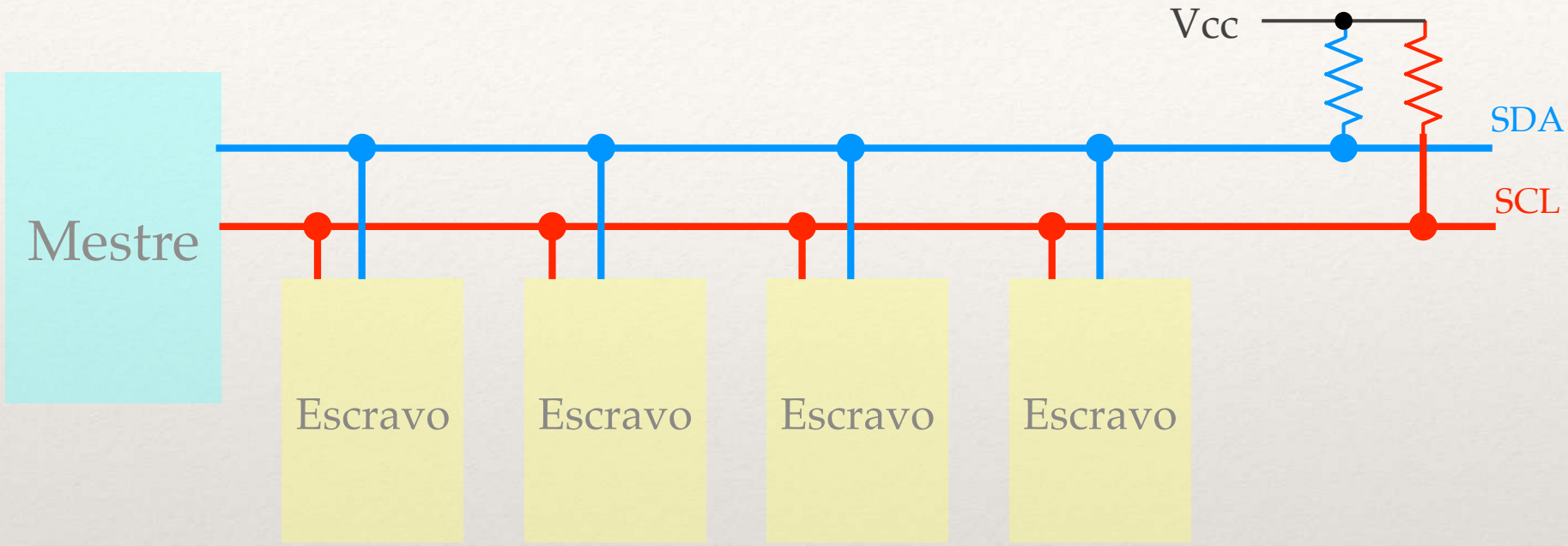
- O I²C não define todos os seus aspectos;
- Nome, conector, e até mesmo tensão de operação são variáveis;
- É muito comum internamente em celulares e outros dispositivos.

Opera em diversos modos

Standard (Sm)	100Kbps	High-Speed (Hs)	1,7Mbps
Fast (Fm)	400Kbps	High-Speed (Hs)	3,4Mbps
Fast plus (Fm+)	1Mbps	Ultra-Fast (UFm)	5Mbps (Unidirecional)

Utiliza 2 fios: SCL (clock) e SDA (serial data)

I²C - Topologia

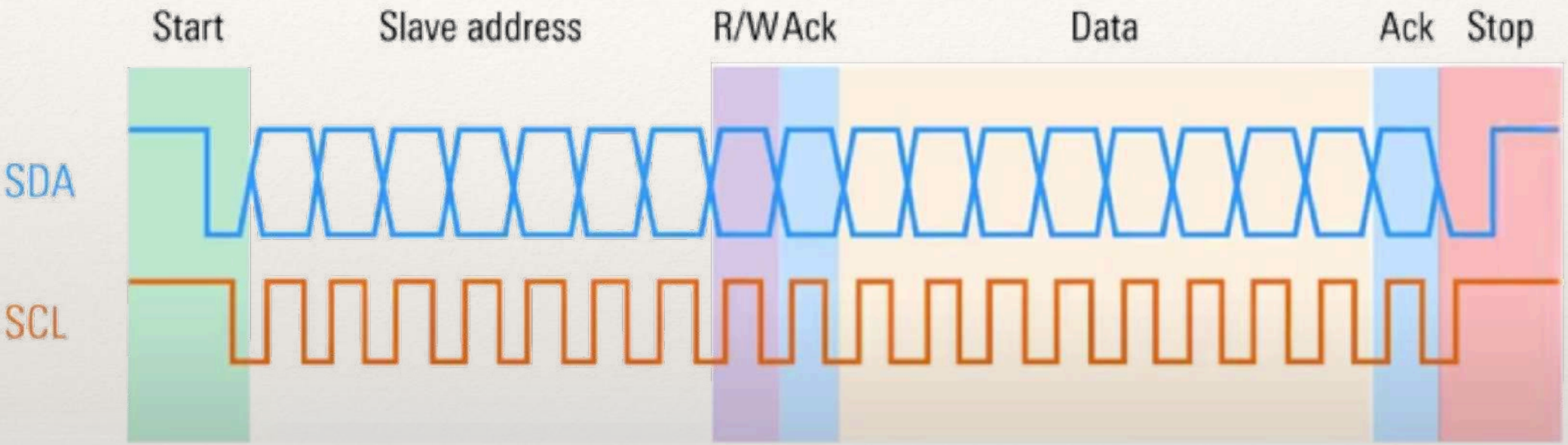


Barramento único

Admite entrada e saída de dispositivos a qualquer momento;

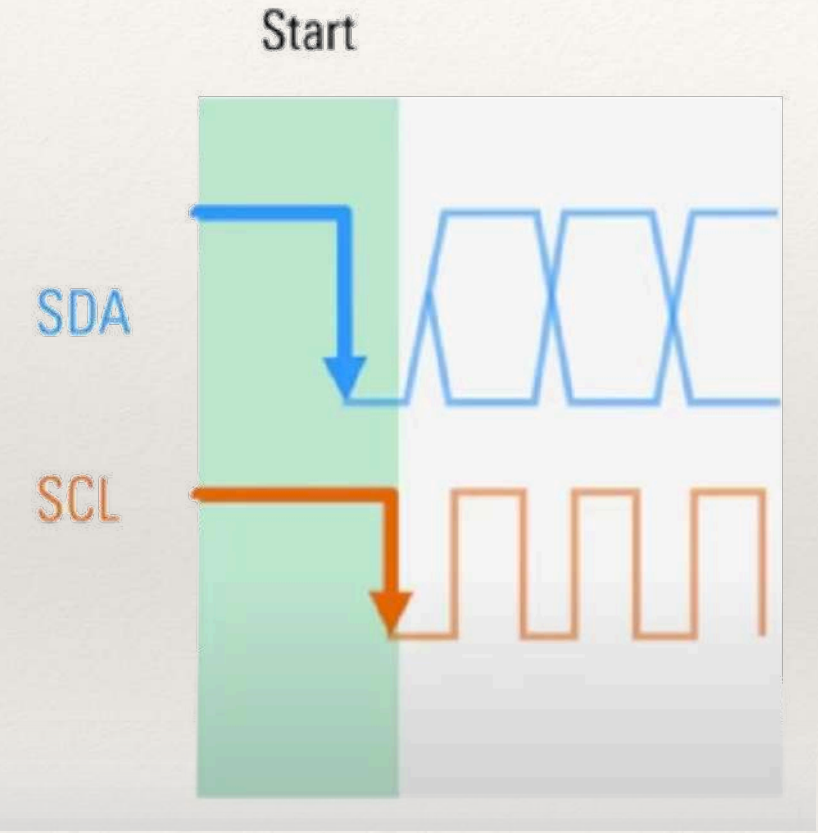
Cada dispositivo possui um endereço que o identifica.

I²C - Formato de Quadro



(*) Slides obtidos no vídeo "Understanding I2C" da Rohde Schwarz

I²C - Formato de Quadro

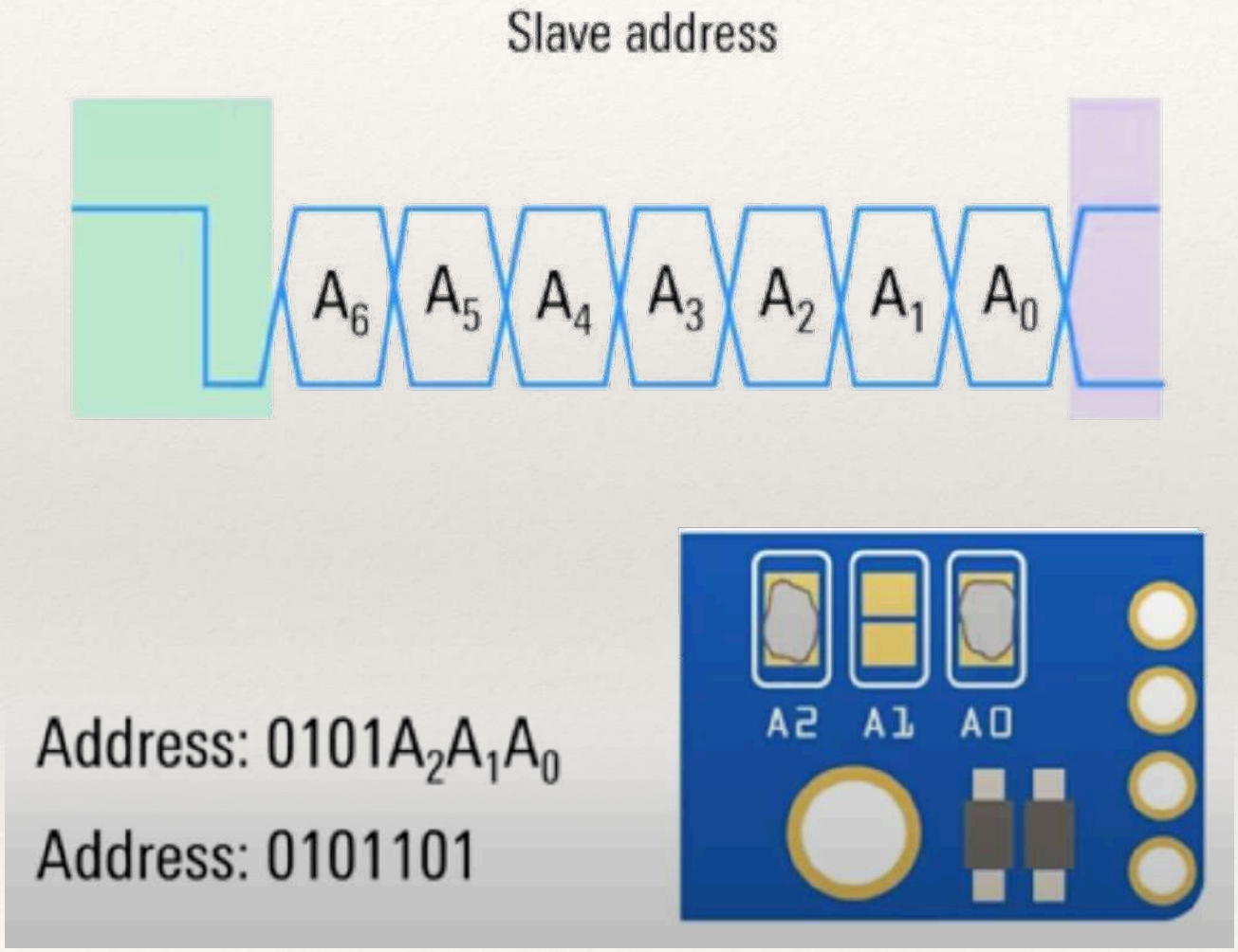


O valor “1" identifica que o barramento está livre;

Sequência determina *star bit*

- SDA desce, seguido por SCL;

I²C - Formato de Quadro



Cada dispositivo tem um endereço de 7 bits

- Começa pelo MSB;
- Endereço pode ser fixo, ou modificado por alterações na placa de circuito impresso do dispositivo.

I²C - Formato de Quadro

O bit R/W vem logo após o endereço, e é definido pelo Master

- 0 → Mestre escreve no escravo;
- 1 → Mestre lê o escravo;

I²C - Formato de Quadro

Definido pelo receptor, vai para zero para confirmação

- 1 → Condição livre do barramento = NACK;

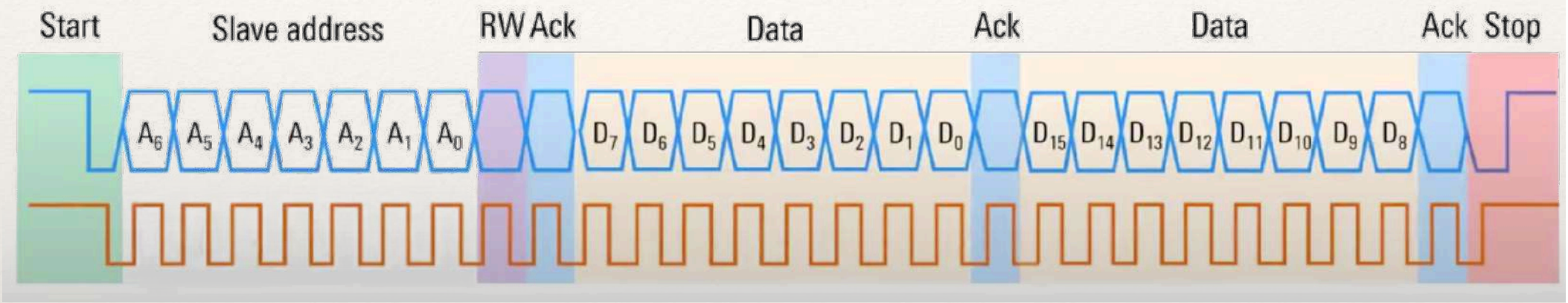
ACK após dados

- Confirme recebimento dos dados;

ACK após endereço do escravo

- Escravo com aquele endereço existe;
- Pronto para ler / escrever, de acordo com R/W.

I²C - Topologia



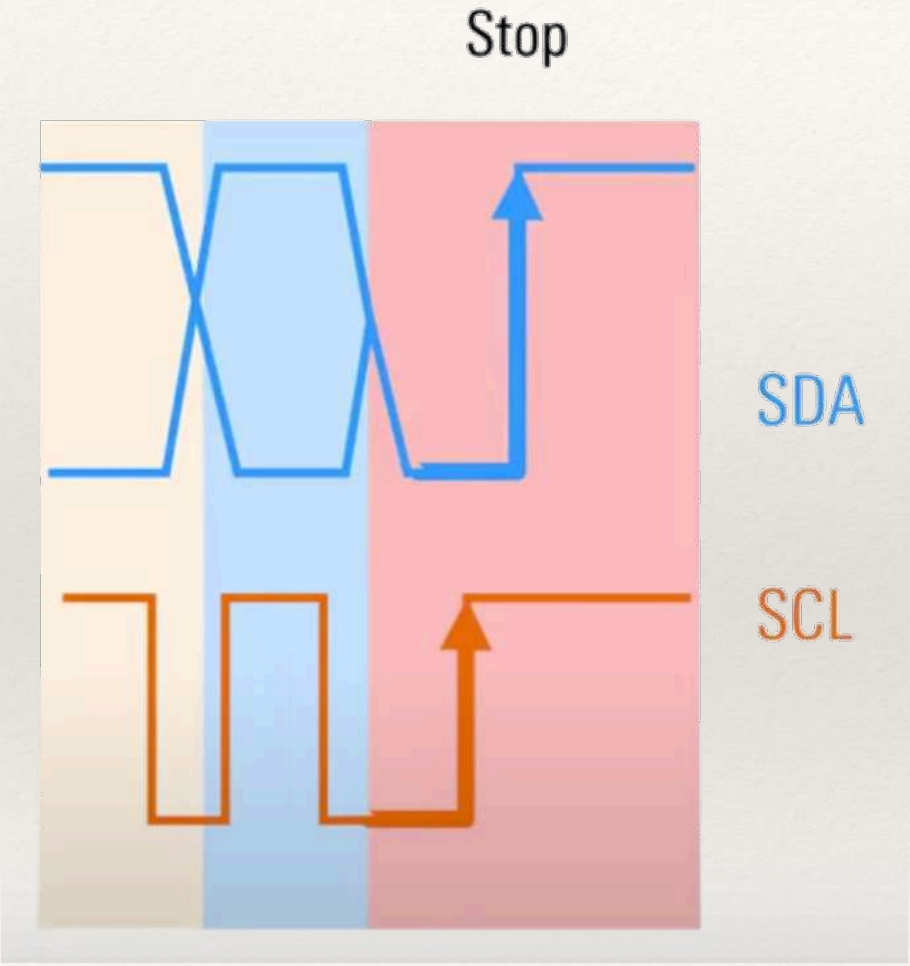
É comum encaminhar mais de um bloco de dados

- Cada bloco individual tem sua confirmação (Ack) específica;

O I2C não define conteúdos, qualquer bloco pode transmitir qualquer coisa, mas ...

- ... é comum, na escrita de registradores de periféricos, que o primeiro bloco contenha o encereço, e o segundo os dados a serem escritos, por exemplo.

I²C - Formato de Quadro



Tudo ocorre de forma similar ao *start bit*;

A sequência inversa determina *stop bit*

- SCL sobe, seguido por SDA;

I²C Programação

Tomando como exemplo o Arduíno, temos os seguintes procedimentos para transmissão e recepção:

Transmissão:

O processo envolve:

```
#include wire.h
wire.begin( endereço )
wire.write( mensagem )
```

Recepção:

O processo envolve:

```
#include wire.h
wire.begin( endereço )
Char = wire.read
```

A opção ESP NOW

Protocolo proprietário da Expressif, fabricante do ESP8266 e ESP32

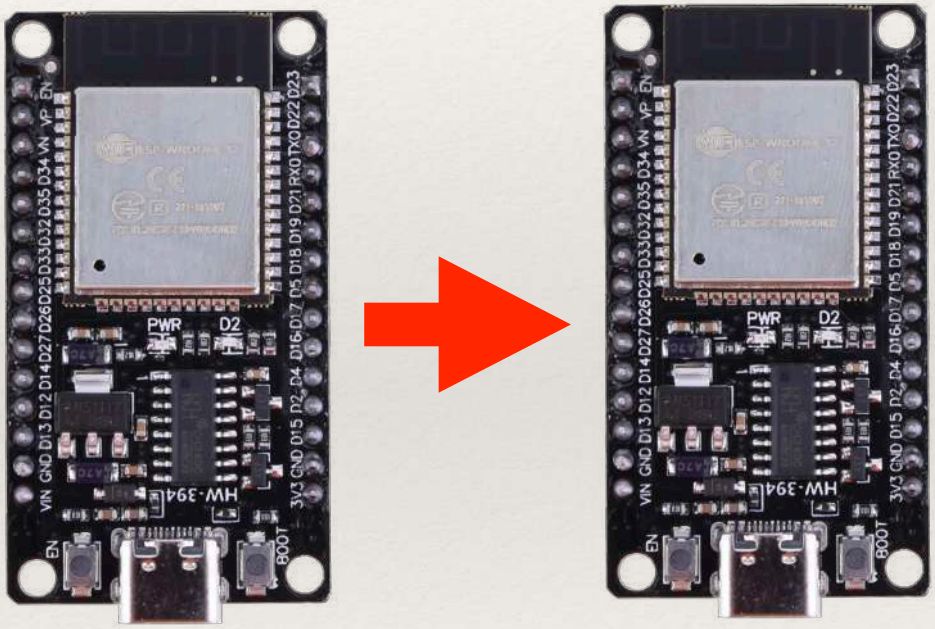
- Utiliza o transceptor de 2,4 GHz já existente para Bluetooth e Wi-Fi;
- Utiliza um padrão de transmissão similar aos teclados e mouses sem fio;
- Não exige infraestrutura Wi-Fi (roteador, por exemplo);
- Aceita comunicação uni e bidirecional com conexões persistentes;

Algumas limitações

- Maior pacote deve ter 250 Bytes (mas permanece útil para SE);
- Podem haver entre 6 e 20 estações, a depender do modo de operação, e do uso de criptografia.

ESP-NOW Unidirecional

Um emissor (*iniciator*) e um receptor (*responder*);

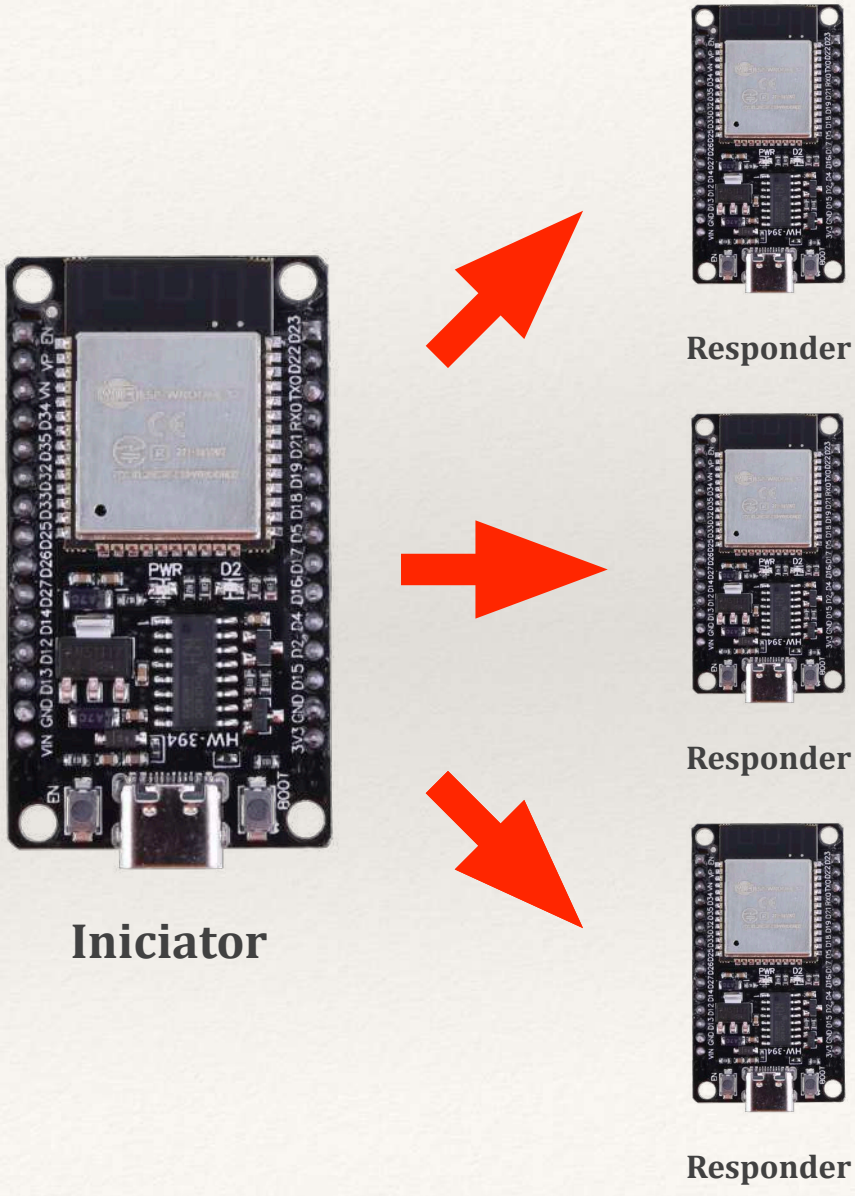


Iniciator **Responder**

ESP-NOW Unidirecional

Um emissor (*iniciator*) e um receptor (*responder*);

Um emissor e vários receptores individualmente, ou em *broadcast* (para todos);

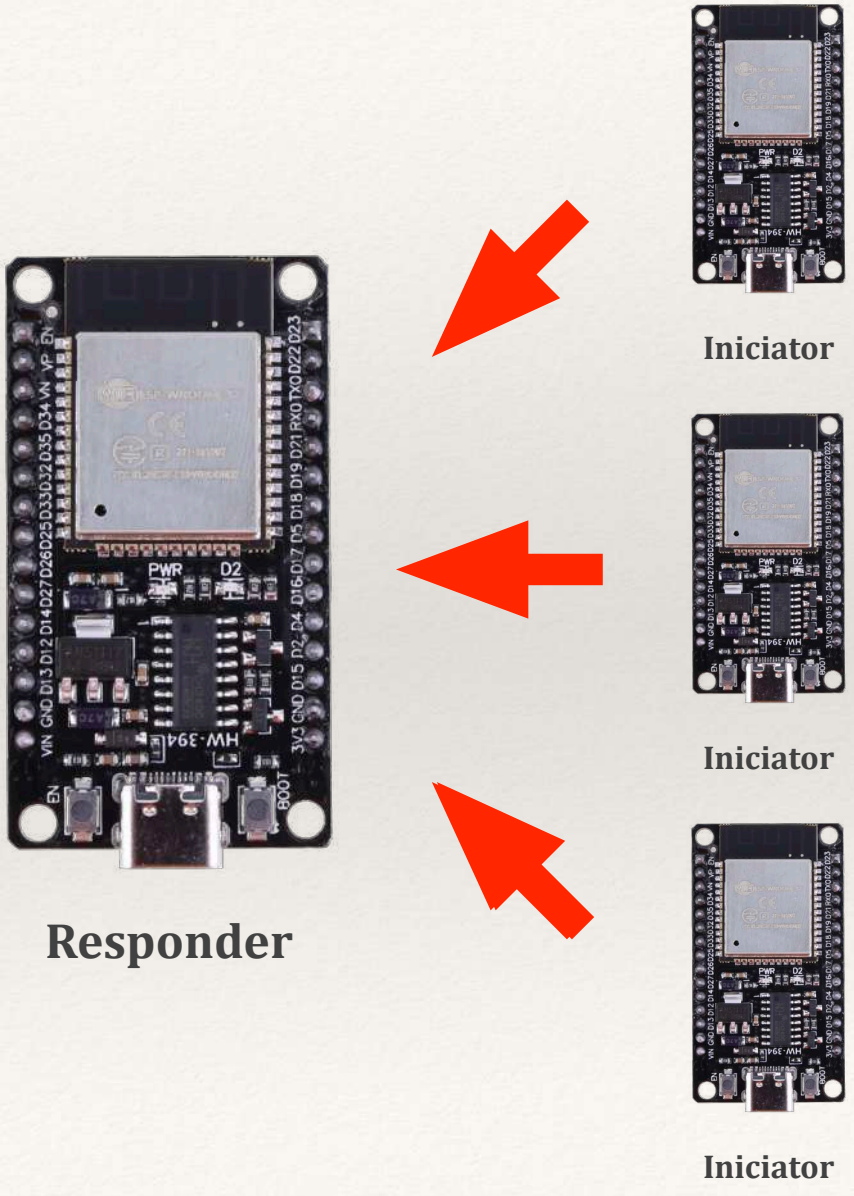


ESP-NOW Unidirecional

Um emissor (*iniciator*) e um receptor (*responder*);

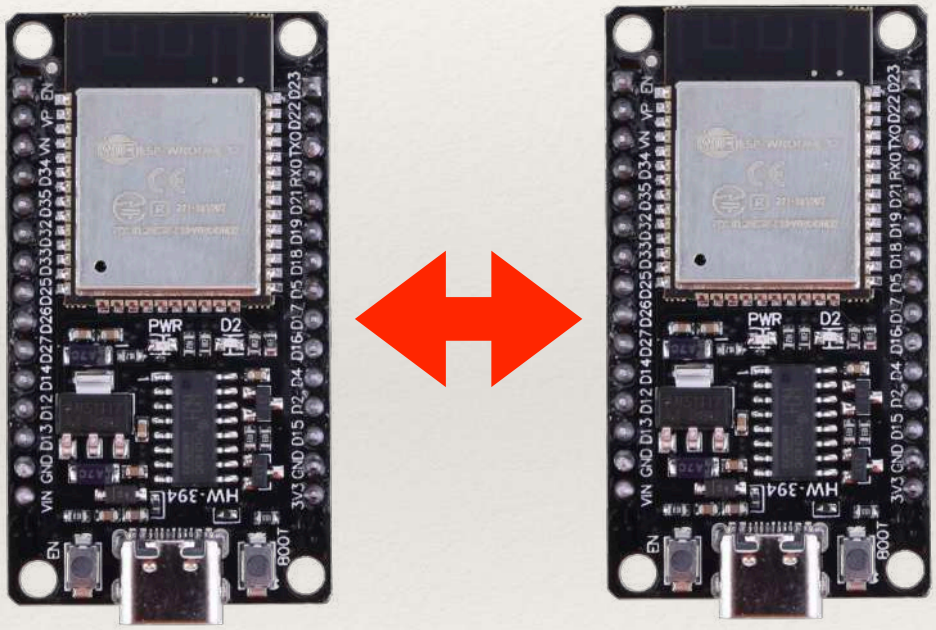
Um emissor e vários receptores individualmente, ou em *broadcast* (para todos);

Vários emissores individualmente ou simultaneamente;



ESP-NOW Bidirecional

Dois dispositivos podem transmitir e ler ao mesmo tempo, atuando simultaneamente como emissor e receptor;



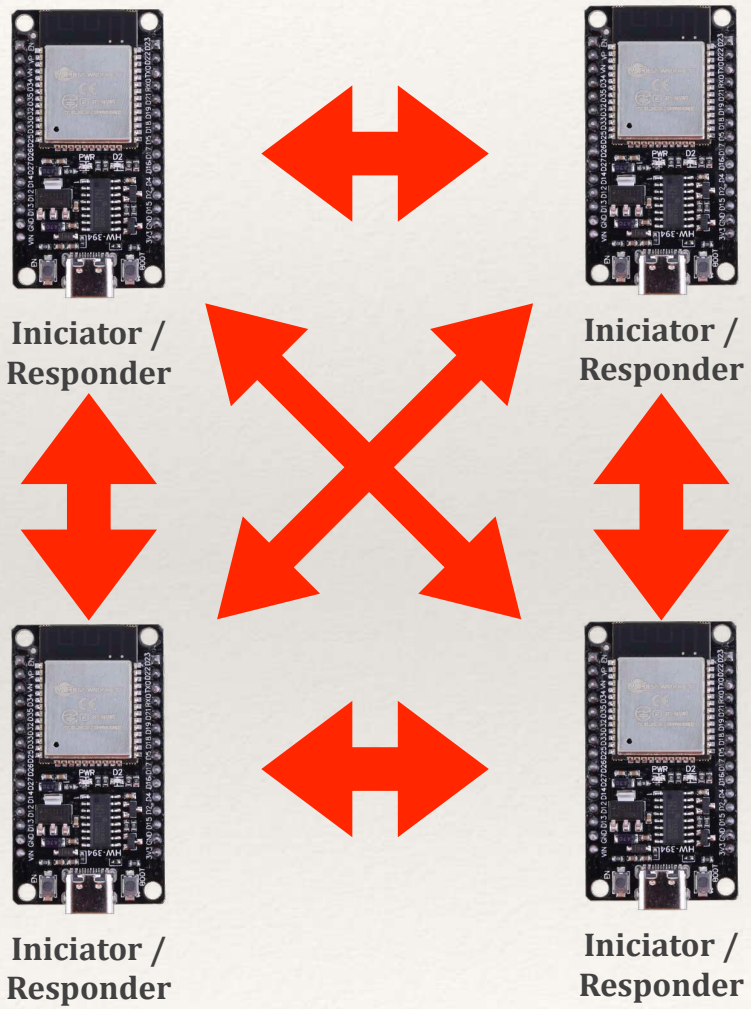
Iniciator / Responder

Iniciator / Responder

ESP-NOW Bidirecional

Dois dispositivos podem transmitir e ler ao mesmo tempo, atuando simultaneamente como emissor e receptor;

Todos os dispositivos podem se comunicar com qualquer outro nas duas direções.



ESP-NOW Endereçamento

Utilizar o *MAC Address* do dispositivo para identificá-lo como destinatário;

O *MAC Address* é um número hexadecimal de 6 dígitos único gravado pelo fabricante;

(*) O ESP-NOW não suporta endereços de *broadcast* (FF:FF:FF:FF:FF:FF), mas existe forma de fazer um *pseudo-broadcast*.

Para obter o endereço, pode-se usar o *sketch* ao lado, ou mesmo verificar as mensagens de inicialização do dispositivo.

MostraMACAddress

```
/*
  ESP32 MAC Address printout
  MostraMACAddress.ino
  Mostra o endereço MAC do ESP32 no Terminal Serial

  Sketch obtido em:
  DroneBot Workshop 2022 ©
  https://dronebotworkshop.com
*/

// Inclui biblioteca WiFi
#include "WiFi.h"

void setup() {

  // Configura o terminal serial com taxa de 115200 bps
  // Lembrar de configurar a mesma taxa no Terminal Serial da IDE do Arduino
  Serial.begin(115200);

  // Coloca o ESP32 no modo "Station"
  WiFi.mode(WIFI_MODE_STA);

  // Mostra o endereço MAC no Terminal Serial
  Serial.print("MAC Address: ");
  Serial.println(WiFi.macAddress());
}

void loop() {

}
```

Funções de *callback*

Para a comunicação usando o ESP-NOW, utilizamos duas funções, chamadas na transmissão e recepção:

`esp_now_register_send_cb()`

Utilizada na transmissão, retorna *status* de envio e recepção;

O processo envolve:

- `Iniciar o ESP-NOW`
- `Registrar a função de callback`
- `Adicionar um receptor (MAC Address)`
- `Enviar a mensagem`

`esp_now_register_rcb_cb()`

Utilizada na recepção, retorna os dados recebidos;

O processo envolve:

- `Iniciar o ESP-NOW`
- `Registrar a função de callback`
- `Salvar a mensagem recebida`

Receptor

```
/*
  ESP-NOW - RecebeMedicoesTemperatura
  Teste do ESP-NOW com recepção múltipla
  Recebe informações de temperatura de sensores conectados a outras ESP32 via ESP-NOW

  Código inspirado na publicação de:
  DroneBot Workshop 2022
  https://dronebotworkshop.com
*/

// Inclui as Bibliotecas necessárias
#include <WiFi.h>
#include <esp_now.h>

// Define estrutura de dados (campo "a" do tipo "float" não foi utilizado)
typedef struct struct_message {
    //float a;
    float b;
    int c;
} struct_message;

// Cria objeto com base na estrutura
struct_message myData;

// função de "Callback"
void OnDataRecv(const uint8_t * mac, const uint8_t *incomingData, int len)
{
  // Lê os dados recebidos
  memcpy(&myData, incomingData, sizeof(myData));

  // Mostra dados no Monitor Serial
  Serial.print("Sensor ");
  Serial.print(myData.c);
  Serial.print(": ");
  Serial.println(myData.b);
  Serial.println("");
}
```

```
void setup() {
  // Configura taxa do Monitor Serial
  Serial.begin(115200);

  // Inicializa o Wi-Fi do ESP32 no modo "Estação"
  WiFi.mode(WIFI_STA);

  // Inicializa o ESP-NOW
  if (esp_now_init() != 0) {
    Serial.println("Erro inicializando o ESP-NOW");
    return;
  }

  // Registra a função de "callback"
  esp_now_register_recv_cb(OnDataRecv);

}

void loop() {
  }
```

Transmissor

```
/*
  ESP-NOW EnviaTemperatura
  Teste do ESP-NOW com transmissão múltipla
  Envia valor da temperatura para um ESP32 central via ESP-NOW

  Código inspirado na publicação de:
  DroneBot Workshop 2022
  https://dronebotworkshop.com
*/

// Inclui as Bibliotecas necessárias
#include <WiFi.h>
#include <esp_now.h>

// Defini constante indicando o pino onde o sensor LM35 está conectado,
// ou seja, ao GPIO 34 (Analog ADC1_CH6)
const int pinoLM35 = 34;

// Identificador do transmissor (deve ser alterado para cada transmissor)
int ident = 1;

// Define o endereço MAC do "Responder"
uint8_t broadcastAddress[] = {0xE8, 0x6B, 0xEA, 0xD0, 0xE2, 0x18};

// Define estrutura de dados (campo "a" do tipo "float" não foi utilizado)
typedef struct struct_message {
  //float a;
  float b;
  int c;
} struct_message;

// Cria objeto com base na estrutura
struct_message myData;

// Register peer
esp_now_peer_info_t peerInfo;

// função de "Callback"
void OnDataSent(const uint8_t *macAddr, esp_now_send_status_t status)
{
  Serial.print("Status de envio do último pacote: ");
  Serial.println(status == ESP_NOW_SEND_SUCCESS ? "Entrega bem sucedida" : "Entrega falhou");
}
```

```
void setup() {

  // Configura tax do Monitor serial
  Serial.begin(115200);
  delay(100);   // Aguarda estabilização (testar necessidade posteriormente)

  // Inicializa o Wi-Fi o ESP32 em modo "Estação"
  WiFi.mode(WIFI_STA);

  // Inicializa o ESP-NOW
  if (esp_now_init() != 0) {
    Serial.println("Erro inicializando o ESP-NOW");
    return;
  }

  // Registra a função de "callback"
  esp_now_register_send_cb(OnDataSent);

  memcpy(peerInfo.peer_addr, broadcastAddress, 6);
  peerInfo.channel = 0;
  peerInfo.encrypt = false;

  if (esp_now_add_peer(&peerInfo) != ESP_OK) {
    Serial.println("Falha ao adicionar destinatário");
    return;
  }

}

void loop() {

  int i, leituraADC;

  leituraADC = analogRead( pinoLM35 );

  myData.b = leituraADC * (3300.0 / 4096);
  myData.c = ident;

  // Envia dados
  esp_now_send(broadcastAddress, (uint8_t *) &myData, sizeof(myData));

  // Aguarda 2 seg para nova leitura do sensor
  delay(2000);
}
```

Aula 13

Protocolos para IoT

Assim como no OSI, os protocolos IoT também são dispostos em camadas. Há modelos alternativos:

- **3 camadas**: percepção, rede e aplicação;
- **4 camadas**: percepção, suporte, rede e aplicação;
- **5 camadas**: percepção, transporte, processamento, aplicação e negócios.

Diversidade grande de protocolos, alguns novos, aplicados em IoT. Nas camadas mais altas, entre outros, temos:

- **HTTP**: *Hiper Text Transfer Protocol*
- **MQTT**: *Message Queuing Telemetry Transport*

HTTP (*Hiper Text Transfer Protocol*)

O protocolo utiliza o TCP, e vem sendo aproveitado para outras aplicações diferentes do objetivo inicial (navegadores WEB)

- Simplicidade de especificação de protocolo
- Estabilidade operacional

Tradicionalmente, o HTTP operava com requisições e respostas

- Cada acesso envolvia o estabelecimento de uma conexão, transferência das informações e encerramento da conexão;
- Para garantir aplicações em tempo real / interativas, surgiu o *HTTP Websockets* e o *HTTP Push*;

HTTP (*Hiper Text Transfer Protocol*)

HTTP Websockets

- Permite conexões persistentes e bidirecionais;

HTTP Push

- Também chamado de HTTP Streaming, permite que servidores enviem respostas aos seus clientes independentes de requisições;
- Permite atualizações em tempo real e comunicações orientadas a eventos;

Aplicação

- O uso do HTTP é particularmente interessante quando a aplicação tem interação via WEB;

MQTT (*Message Queuing Telemetry Transport*)

Criado em 1999 por Andy Standor-Clark (IBM) e Arlen Nipper (Cirrus Link)

Baseado no modelo de Publicação / Assinatura

- Define dois componentes: os **clientes** e os ***brokers***;
- **Clientes** podem publicar ou acessar informações
  - O acesso tipicamente se dá através de uma estrutura de tópicos pré-definida, onde clientes são “assinantes” ou através de filtros específicos;
- ***Brokers*** atuam como intermediários, responsáveis por receber informações "publicadas", e entregar estas informações para os clientes “assinantes”
  - Disponíveis em plataformas abertas e gratuitas (Ex. Mosquitto);

Este modelo “desacoplado" habilita aplicações eficientes e escaláveis

- Suporte a ambientes com comunicação instável ?
- Independência dos processos de envio e recepção;
- Não é necessário um POST para cada cliente, como no HTTP.

MQTT (*Message Queuing Telemetry Transport*)

Utiliza o TCP, com recursos de QoS

- Níveis: 0 - "at most once”; 1 -"at least once”; 2 -"exactly once”
- Conexões persistentes

Protocolo leve, com *payload* binário

- Uma mensagem MQTT pode ter apenas 2 Bytes !
- Clientes/dispositivos de baixo consumo;
- Taxas de transferência reduzidas (Ex.LoRa);
- Habilita aplicações pervasivas;

HTTP x MQTT

	HTTP WEBSOCKETS	MQTT
Modelo	Full-duplex, Bidirecional	Publicação / Assinatura
Endereçamento	URIs	Tópicos
Protocolo de Transporte	TCP	TCP, UDP
Acompanhamento Cliente	N/D	Mensagens
Retenção de Mensagens	N/D	Sim
Modo de envio de mensagens	Bidirecional	Assíncrono
Arquitetura	Cliente/Servidor	Cliente/Servidor
Confiabilidade das Mensagens	WebSockets	QoS níveis 1 e 2

Fim