



Residência  
Tecnológica  
em Sistemas  
Embarcados

# Conversor Analógico-Digital

Unidade 4 | Capítulo 8

Executores:



Coordenação:



Iniciativa:



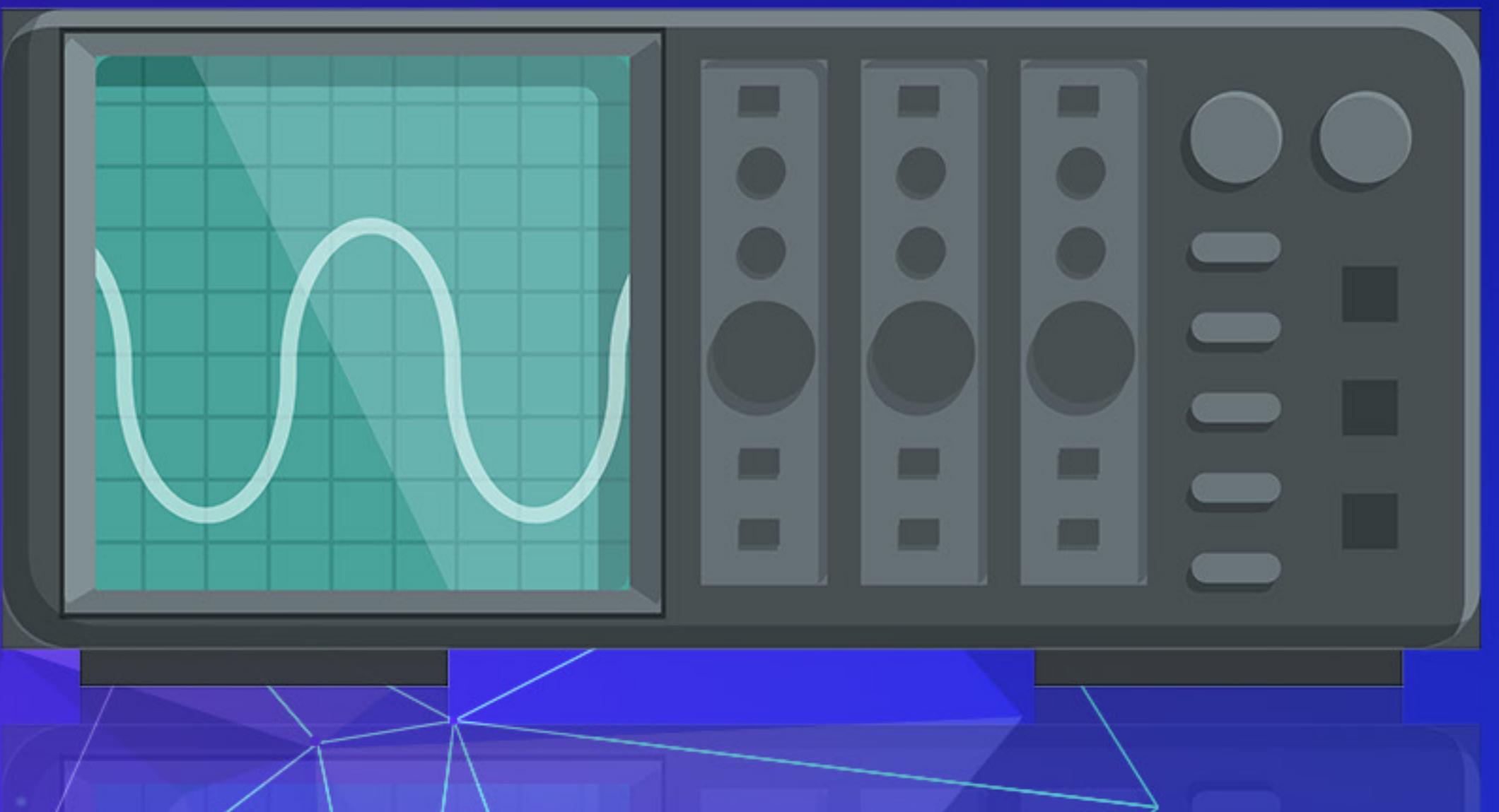
MINISTÉRIO DA  
CIÊNCIA, TECNOLOGIA  
E INOVAÇÃO

UNIÃO E RECONSTRUÇÃO

BRASIL

# Sumário

- Objetivos
- Revisão
- Visão Geral
- ADC no RP2040
- Configurações
- Exemplos de Código
- Principais Pontos
- Conclusão



# Objetivos

- Entender o processo de conversão A/D.
- Compreender o funcionamento do módulo A/D.
- Conhecer os diferentes modos de configuração do módulo conversor A/D.
- Configurar e implementar o uso do módulo A/D, inclusive em conjunto com o módulo PWM.



# Revisão

Nas últimas aulas você aprendeu:

- Usar e configurar o clocke temporizadores
- A utilizar as interfaces de comunicação serial
- A configurar e aplicar o módulo PWM

# Visão Geral

## O que é um Conversor Analógico-Digital (ADC)?

- Dispositivo que converte sinais analógicos (variáveis contínuas) em sinais digitais (variáveis discretas)
- Microcontroladores só lidam com sinais digitais
  - » Precisamos do ADC para tratar sensores analógicos

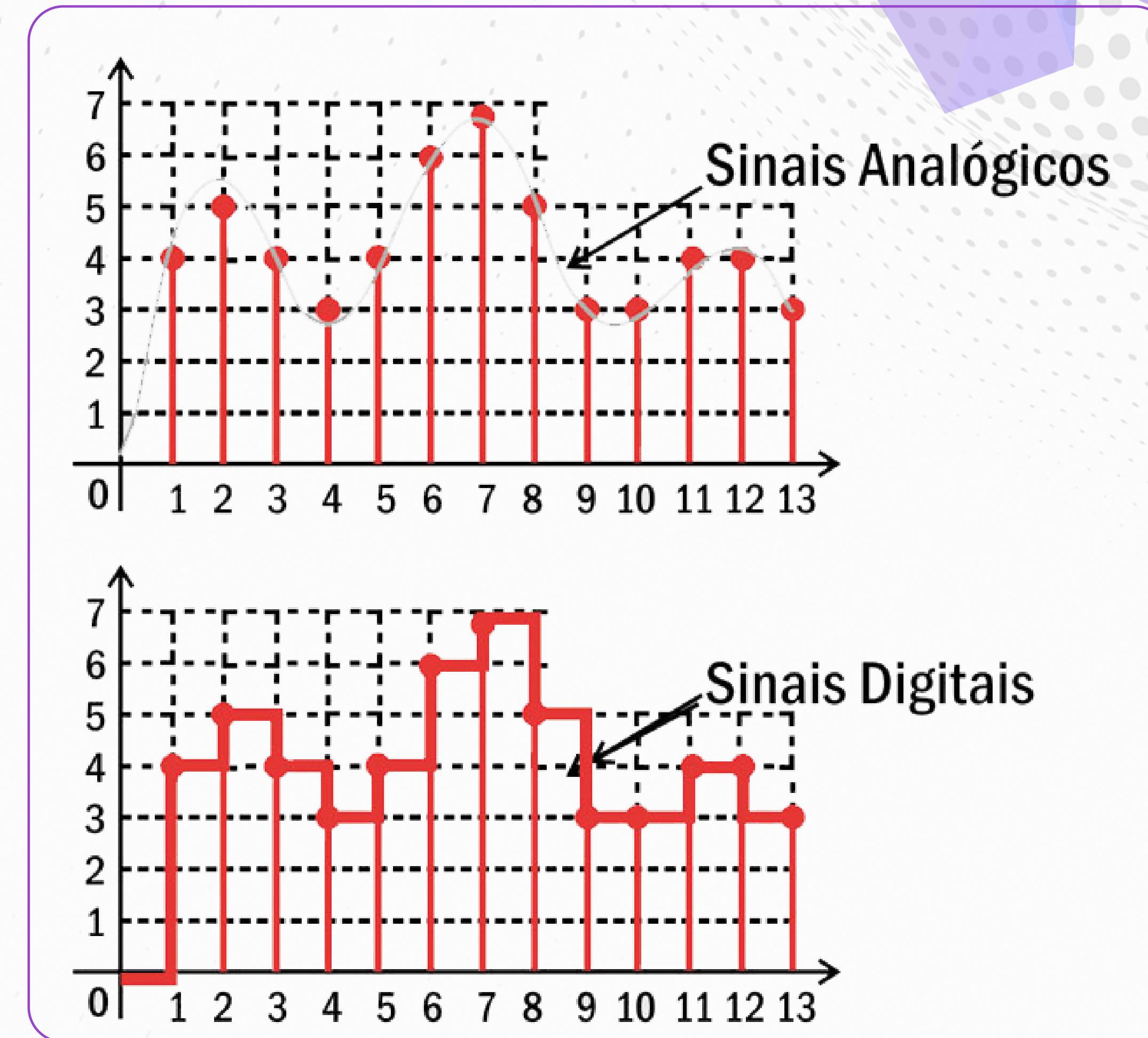


Fig. 1. Comparação sinal analógico e digital  
Fonte: sistemas-automotivos-osciloscópio-03-2017-1.jpg (605×590) (oficinabrasil.com.br)

A imagem apresenta dois gráficos que comparam sinais analógicos e digitais.  
Gráfico superior (Sinais Analógicos):

- Linha contínua com pontos: Uma linha que se move suavemente para cima e para baixo, conectando todos os pontos. Essa linha representa um sinal analógico, que pode assumir qualquer valor dentro de um intervalo contínuo.

- Pontos: Cada ponto na linha representa um valor específico do sinal em um determinado momento.

Gráfico inferior (Sinais Digitais):

- Linhas retas: Linhas retas que conectam pontos em diferentes alturas, formando um degrau. Essa forma representa um sinal digital, que assume apenas valores discretos (ou seja, valores específicos dentro de um conjunto finito).

- Degraus: Os degraus indicam que o sinal digital muda abruptamente de um valor para outro, sem transições suaves.

Comparação entre os gráficos:

- Sinais analógicos: São contínuos e podem assumir infinitos valores dentro de um intervalo. Exemplos de sinal analógico incluem a temperatura, a pressão e a luz.

- Sinais digitais: São discretos e assumem apenas valores específicos, geralmente representados por 0 e 1. Exemplos de sinal digital incluem os dados armazenados em um computador.

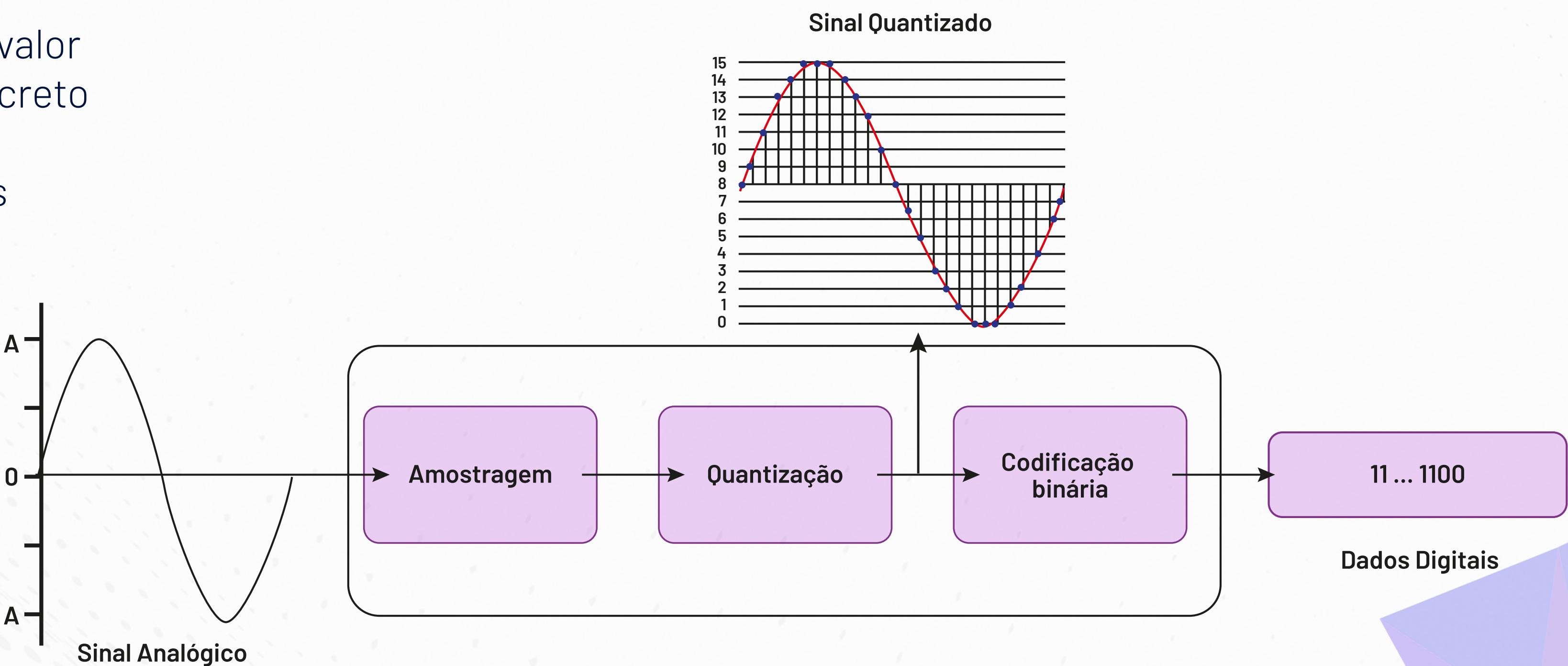
# Visão Geral

## Princípios de Funcionamento

- Amostragem
  - » Processo no qual o ADC lê o sinal analógico em períodos pré-definidos.
  - » Maior afrequênciade amostragem, mais preciso é a representação do sinal
- Quantização
  - » Aproxima (arredonda) o valor para um valor digital discreto
    - \* Erro de Quantização
    - \* Maior a resolução mais níveis de quantização
- Codificação binária
  - » Mapeia os valore

**Fig. 2. Processo de conversão de um sinal analógico em digital.**  
Fonte: [https://miro.medium.com/v2/resize:fit:815/0\\*\\_RcOXGzoUkIBJevR.png](https://miro.medium.com/v2/resize:fit:815/0*_RcOXGzoUkIBJevR.png)

O texto descreve o processo de conversão de um sinal analógico (contínuo) para um sinal digital (discreto).  
Esse processo ocorre em três etapas principais:  
1. Amostragem: O sinal analógico é medido em intervalos regulares de tempo, criando uma série de pontos de dados.  
2. Quantização: Esses pontos de dados são arredondados para valores predefinidos, transformando o sinal em um conjunto de níveis discretos.  
3. Codificação: Os valores discretos são convertidos em uma sequência de bits (zeros e uns), que é a representação digital do sinal original.  
Em resumo, o texto explica como um sinal físico contínuo (como o som ou a luz) pode ser transformado em uma representação numérica (digital) para ser processado por dispositivos eletrônicos.



# Visão Geral

## Características Importantes

- Resolução
  - » Define a precisão do ADC
    - \* 8 bits, 10 bits, 12 bits, dentre outras
- Taxa de Amostragem
  - » Quantas amostras podem ser feitas
    - \* Medida em sps (samples per second)
- Tensão de Referência
  - » Intervalo de tensões que o ADC pode medir

# ADC no RP2040

## O RP2040 tem um ADC interno

- Conversão por Aproximações Sucessivas
- 500 ksps (amostras por segundo) com clock de 48 MHz
- Resolução de 12-bits
- Cinco entradas
  - » 4 pinos externos (GP26-GP29)
  - » 1 sensor de temperatura
- Interrupção/DMA
  - » FIFO 8 amostras

Fig. 3. Diagrama de conversor analógico-digital e suas conexões com diferentes entradas.

Fonte: pág. 561. <https://datasheets.raspberrypi.com/rp2040/rp2040-datasheet.pdf>

O diagrama mostra como um microcontrolador pode coletar dados de diferentes sensores e convertê-los em um formato que possa ser processado por um computador.

Elementos-chave:

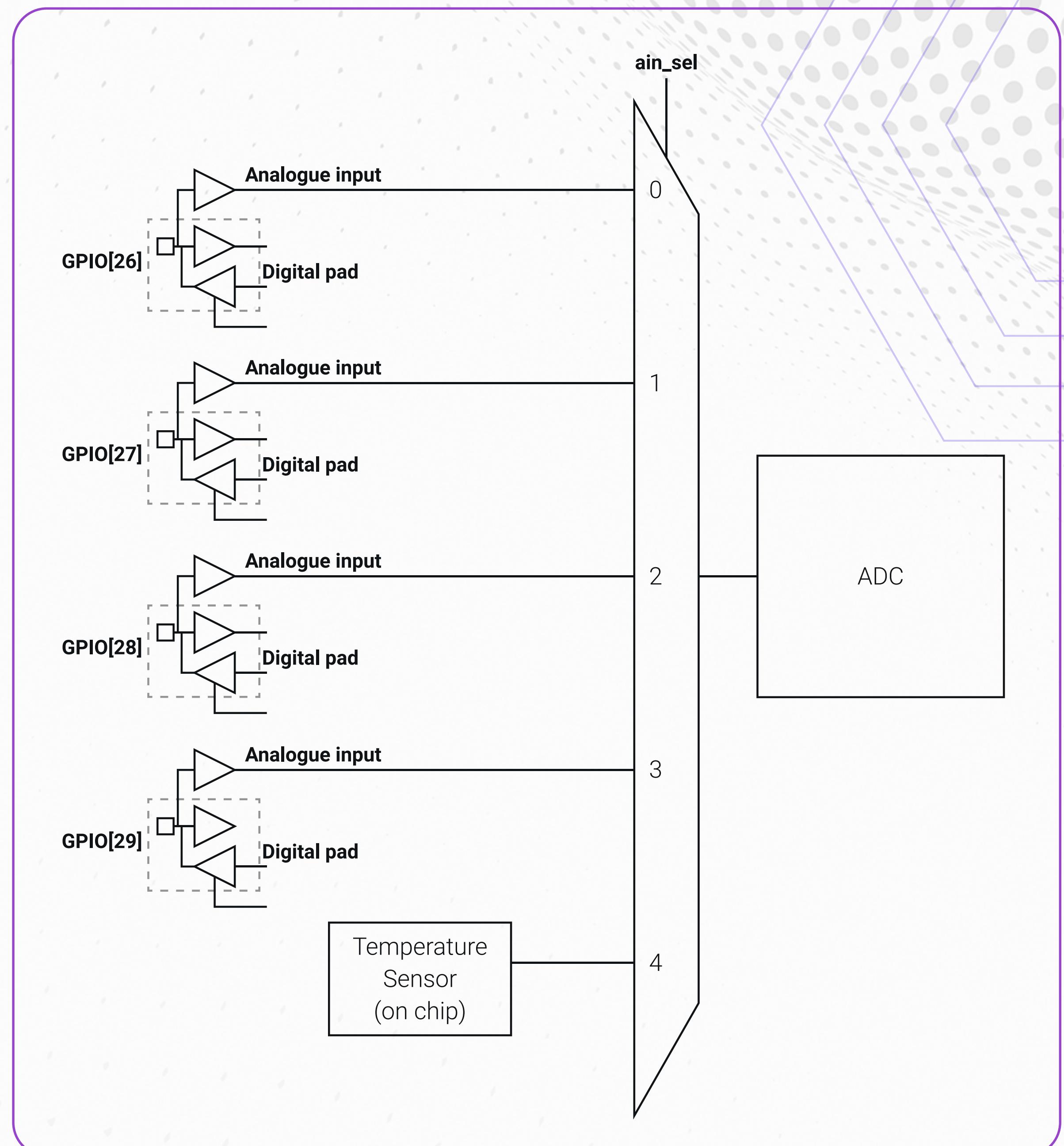
- Entradas Analógicas: São pontos de conexão onde os sinais analógicos (como a voltagem de um sensor) são inseridos.
- Seletor de Entrada: Permite escolher qual entrada analógica será processada a cada momento.
- Conversor Analógico-Digital (ADC): Transforma o sinal analógico contínuo em um valor digital discreto, que o microcontrolador pode entender.
- Sensor de Temperatura: Um exemplo de sensor analógico que pode ser conectado ao microcontrolador.
- Microcontrolador: Utiliza o valor digital obtido para realizar diversas tarefas, como controlar outros dispositivos ou exibir informações em uma tela.

Funcionamento:

1. Seleção da Entrada: O microcontrolador escolhe qual sensor quer ler.

2. Conversão: O sinal analógico do sensor selecionado é convertido em um número digital.

Processamento: O microcontrolador utiliza esse número para tomar decisões ou realizar ações.



# ADC no RP2040

## Conversor SAR

- O processo de conversão é feito por aproximações sucessivas
  - » Um conversor digital-analógico (DAC) gera valores analógicos que são comparados ao valor amostrado
  - \* Um comparador determina se o valor gerado pelo DAC é maior (1) ou menor (0) que o valor amostrado
    - O processo se repete com o DAC gerando valores analógicos que são frações da tensão de referência
    - Processo lento
      - > Leva 96 ciclos de clock para gerar os 12-bits
      - > A precisão é baixa
        - Apenas os 9 bits mais significativos são precisos!

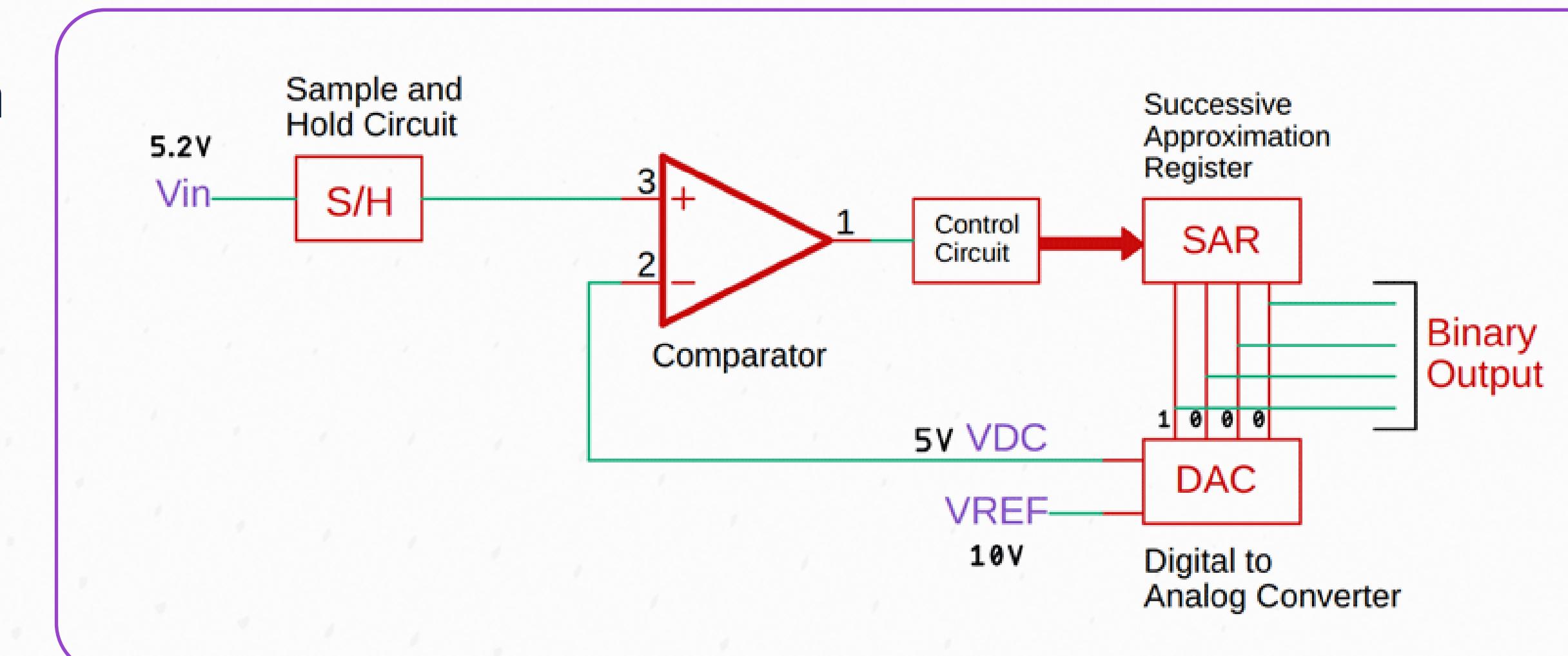


Fig. 4. Diagrama de um Conversor Analógico-Digital (ADC).  
Fonte: <https://circuitdigest.com/sites/default/files/inlineimages/u3/Successive-Approximation-ADC-Working.png>

Imagine um termômetro antigo de mercúrio. A altura da coluna de mercúrio muda de forma contínua conforme a temperatura varia. Essa é uma medida analógica. Agora, pense em um termômetro digital. Ele mostra a temperatura como um número, por exemplo, 25 graus. Esse número é uma representação digital da temperatura. O diagrama que você viu mostra como um circuito eletrônico transforma uma medida analógica (como a tensão de um sensor) em um valor digital.

Como funciona:

- Amostragem: O circuito "congela" o valor da medida analógica em um determinado momento.
- Comparação: O valor congelado é comparado com diferentes valores de referência.
- Aproximação: O circuito "chuta" um valor digital e verifica se está próximo do valor real. Se não estiver, ajusta o chute até encontrar o valor mais próximo.
- Resultado: O valor digital final é uma representação precisa do valor analógico original.

# Configurações

O ADC possui dois modos de funcionamento:

- One-Shot
  - » Realiza uma única conversão por vez
    - \* É preciso
      - Inicializar o módulo ADC -adc\_init()
      - Configurar GPIO com entrada analógica -adc\_gpio\_init(gpio)
      - Selecionar a entrada analógica -adc\_select\_input(channel)
      - Realizar a conversão -v = adc\_read()
- Free-Running
  - » Inicia novas conversões de forma automática
  - » Amostra de forma sequencial as entradas analógicas pré-selecionadas
  - » Armazena em uma FIFO de oito entradas
  - » Integração com o DMA

# Exemplos de Códigos

## Leitura de Entrada ADC

- Vamos fazer a leitura do Joystick da BitDogLab
  - » O Joystick gera dois sinais analógicos que representam os eixos X e Y
    - \* GP26 -Canal Analógico 0
      - VRX
    - \* GP27 -Canal Analógico 1
      - VRY
  - \* Há ainda uma entrada digital, que representa se o controle foi pressionado
    - GP22
- » Nesse exemplo, iremos ler astrês entradas e usar o console serial para mostrar os seus valores

# Exemplos de Códigos

- Crie o projeto no VS Code com auxílio do plugin do Pi Pico
- Faça a alteração abaixo no CMakeLists.txt
  - » Adicionar a biblioteca hardware\_adc para configurar o console pela USB

```
# Modify the below lines to enable/disable output over UART/USB
pico_enable_stdio_uart(joystick 0)
pico_enable_stdio_usb(joystick 1)

target_link_libraries(joystick pico_stdlib hardware_adc)

pico_add_extra_outputs(joystick)
```

# Exemplos de Códigos

## Código da Aplicação

```
#include <stdio.h>
#include "hardware/adc.h"
#include "pico/stdlib.h"

const int VRX = 26;
const int VRY = 27;
const int ADC_CHANNEL_0 = 0;
const int ADC_CHANNEL_1 = 1;
const int SW = 22;
```

# Exemplos de Códigos

## Código da Aplicação

```
void setup()
{
    // Initialize chosen serial port
    stdio_init_all();
    adc_init();
    adc_gpio_init(VRX);
    adc_gpio_init(VRY);
    gpio_init(SW);
    gpio_set_dir(SW, GPIO_IN);
    gpio_pull_up(SW);
}
```

# Exemplos de Códigos

## Código da Aplicação

```
int main()
{
    uint16_t vrxi_value, vryi_value, swi_value;
    setup();
    printf("Joystick\n");
    while (1)
    {
        adc_select_input(ADC_CHANNEL_0);
        sleep_us(2);
        vrxi_value = adc_read();
        adc_select_input(ADC_CHANNEL_1);
        sleep_us(2);
        vryi_value = adc_read();
        swi_value = !gpio_get(SW);
        printf("X: %u, Y: %u, Botão: %d\n", vrxi_value, vryi_value, swi_value);
        sleep_ms(10000);
    }
}
```

# Exemplos de Códigos

## Controle de intensidade dos LEDs com o JoyStick

- Vamos controlar a intensidade do LED RGB usando PWM a partir dos comandos do Joystick
  - » Procedimento
    - \* Configurar os pinos analógicos e as saídas digitais (LEDs azul e verde)
    - \* Configurar dois PWMS
      - Wrap de 4096
      - Level(dutycycle) igual ao valor lido do Joystick
  - » Vamos fazer um loop de 100 ms
    - \* Realizar a leitura das entradas analógicos
    - \* Ajuste do duty cycle

# Exemplos de Códigos

Crie o projeto no VS Code com auxílio do plugin do Pi Pico

- Altere o CMakeLists.txt
  - » Adicione as bibliotecas do ADC e PWM

```
# Modify the below lines to enable/disable output over UART/USB
pico_enable_stdio_uart(joystick 0)
pico_enable_stdio_usb(joystick 1)

target_link_libraries(joystick pico_stdlib hardware_adc hardware_pwm)

pico_add_extra_outputs(joystick)
```

# Exemplos de Códigos

## Código da Aplicação

```
#include <stdio.h>          // Biblioteca padrão de entrada e saída
#include "hardware/adc.h"    // Biblioteca para manipulação do ADC no RP2040
#include "hardware/pwm.h"    // Biblioteca para controle de PWM no RP2040
#include "pico/stdlib.h"      // Biblioteca padrão do Raspberry Pi Pico

// Definição dos pinos usados para o joystick e LEDs
const int VRX = 26;          // Pino de leitura do eixo X do joystick (conectado ao ADC)
const int VRY = 27;          // Pino de leitura do eixo Y do joystick (conectado ao ADC)
const int ADC_CHANNEL_0 = 0;   // Canal ADC para o eixo X do joystick
const int ADC_CHANNEL_1 = 1;   // Canal ADC para o eixo Y do joystick
const int SW = 22;            // Pino de leitura do botão do joystick
```

# Exemplos de Códigos

## Código da Aplicação

```
const int LED_B = 13;          // Pino para controle do LED azul via PWM
const int LED_R = 11;          // Pino para controle do LED vermelho via PWM
const float DIVIDER_PWM = 16.0; // Divisor fracional do clock para o PWM
const uint16_t PERIOD = 4096;   // Período do PWM (valor máximo do contador)
uint16_t led_b_level, led_r_level = 100; // Inicialização dos níveis de PWM para os LEDs
uint slice_led_b, slice_led_r;    // Variáveis para armazenar os slices de PWM
                                //correspondentes aos LEDs
```

# Exemplos de Códigos

## Código da Aplicação

```
// Função para configurar o joystick (pinos de leitura e ADC)
void setup_joystick()
{
    // Inicializa o ADC e os pinos de entrada analógica
    adc_init();          // Inicializa o módulo ADC
    adc_gpio_init(VRX); // Configura o pino VRX (eixo X) para entrada ADC
    adc_gpio_init(VRY); // Configura o pino VRY (eixo Y) para entrada ADC

    // Inicializa o pino do botão do joystick
    gpio_init(SW);        // Inicializa o pino do botão
    gpio_set_dir(SW, GPIO_IN); // Configura o pino do botão como entrada
    gpio_pull_up(SW);      // Ativa o pull-up no pino do botão (para evitar flutuações)
}
```

# Exemplos de Códigos

## Código da Aplicação

```
// Função para configurar o PWM de um LED (genérica para azul e vermelho)
void setup_pwm_led(uint led, uint *slice, uint16_t level)
{
    gpio_set_function(led, GPIO_FUNC_PWM); // Configura o pino do LED como saída PWM
    *slice = pwm_gpio_to_slice_num(led); // Obtém o slice do PWM associado ao pino do LED
    pwm_set_clkdiv(*slice, DIVIDER_PWM); // Define o divisor de clock do PWM
    pwm_set_wrap(*slice, PERIOD); // Configura o valor máximo do contador (período
    pwm_set_gpio_level(led, level); // Define o nível inicial do PWM para o LED
    pwm_set_enabled(*slice, true); // Habilita o PWM no slice correspondente ao LED
}
```

# Exemplos de Códigos

## Código da Aplicação

```
// Função de configuração geral
void setup()
{
    stdio_init_all();                                // Inicializa a porta serial para saída
    setup_joystick();                               // Chama a função de configuração do joystick
    setup_pwm_led(LED_B, &slice_led_b, led_b_level); // Configura o PWM para o LED azul
    setup_pwm_led(LED_R, &slice_led_r, led_r_level); // Configura o PWM para o LED vermelho
}
```

# Exemplos de Códigos

## Código da Aplicação

```
// Função para ler os valores dos eixos do joystick (X e Y)
void joystick_read_axis(uint16_t *vrx_value, uint16_t *vry_value)
{
    // Leitura do valor do eixo X do joystick
    adc_select_input(ADC_CHANNEL_0); // Seleciona o canal ADC para o eixo X
    sleep_us(2); // Pequeno delay para estabilidade
    *vrx_value = adc_read(); // Lê o valor do eixo X (0-4095)

    // Leitura do valor do eixo Y do joystick
    adc_select_input(ADC_CHANNEL_1); // Seleciona o canal ADC para o eixo Y
    sleep_us(2); // Pequeno delay para estabilidade
    *vry_value = adc_read(); // Lê o valor do eixo Y (0-4095)
}
```

# Exemplos de Códigos

## Código da Aplicação

```
int main()
{
    uint16_t vrx_value, vry_value, sw_value; // Variáveis para armazenar os valores do joystick
    setup(); // Chama a função de configuração
    printf("Joystick-PWM\n"); // Exibe uma mensagem inicial via porta serial
    // Loop principal
    while (1)
    {
        joystick_read_axis(&vrx_value, &vry_value); // Lê os valores dos eixos do joystick
        // Ajusta os níveis PWM dos LEDs de acordo com os valores do joystick
        pwm_set_gpio_level(LED_B, vrx_value); // Ajusta o brilho do LED azul com o valor do eixo
        pwm_set_gpio_level(LED_R, vry_value); // Ajusta o brilho do LED vermelho com o valor do eixo

        // Pequeno delay antes da próxima leitura
        sleep_ms(100); // Espera 100 ms antes de repetir o ciclo
    }
}
```

# Principais Pontos

- O ADC é um dispositivo que na interface do mundo analógico com o mundo digital
- Um ADC possui várias características como resolução, taxa de amostragem, tensão referencial
- O RP2040 possui um ADC interno do tipo SAR
  - » 12 bits de resolução
  - » 500 ksps
- Aprendemos como usar o SDK para criar aplicações que usam o ADC e outros módulos como o PWM

# Conversor Analógico-Digital

Unidade 4 | Capítulo 8

---

## Fonte de imagens:

<https://br.freepik.com>  
<https://oficinabrasil.com.br/uploads/imagens/Tecnicas/sistemas-automotivos-oscilloscopio-03-2017-1.jpg>  
[https://miro.medium.com/v2/resize:fit:815/0\\*\\_Rc0XGzoUk1BjevR.png](https://miro.medium.com/v2/resize:fit:815/0*_Rc0XGzoUk1BjevR.png)  
<https://datasheets.raspberrypi.com/rp2040/rp2040-datasheet.pdf>  
<https://circuitdigest.com/sites/default/files/inlineimages/u3/Successive-Approximation-ADC-Working.png>