

UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
GEDRE - INTELIGÊNCIA EM ILUMINAÇÃO

gedre
inteligência
em iluminação

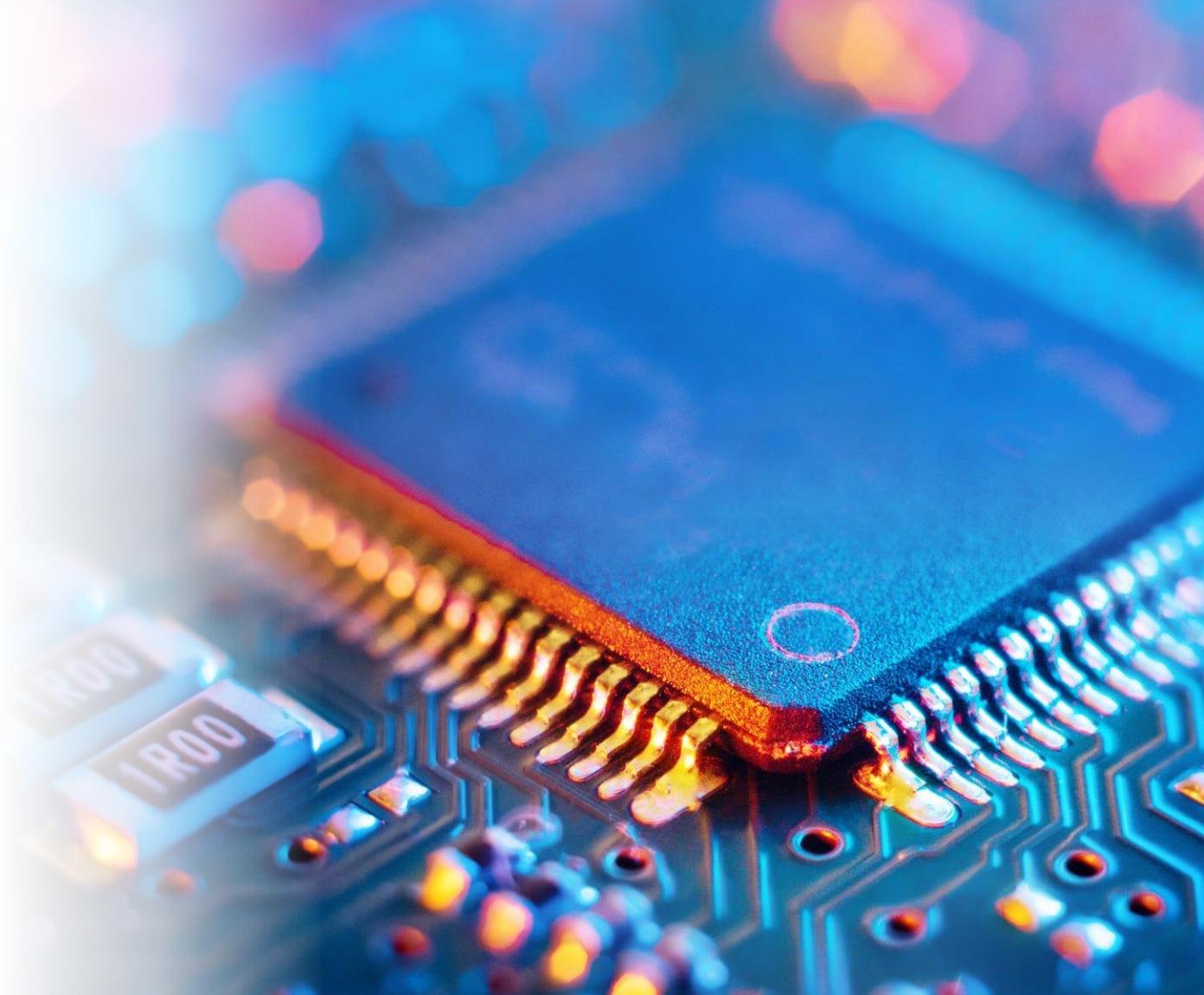
Minicurso de Microcontroladores

Prof. Renan Duarte

Março de 2025

Visão geral do minicurso

1. Conceitos básicos
2. Instalação dos softwares
3. Projeto base
4. System Clock
5. GPIO
6. Interrupções
7. Timer
8. ADC
9. PWM
10. Desafio final



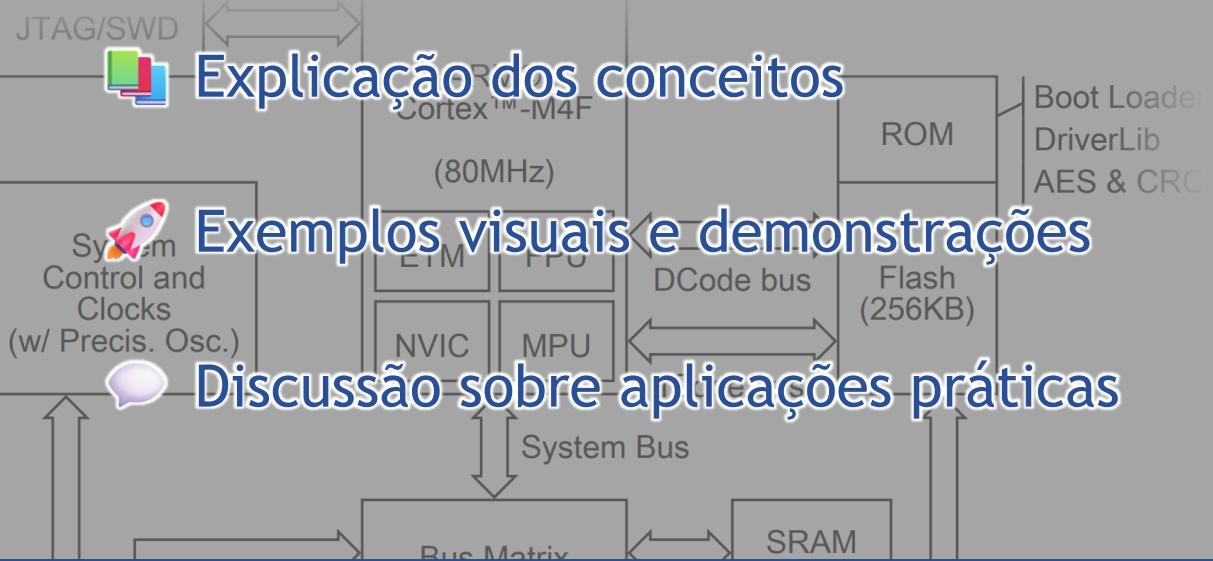
Visão geral do minicurso

Parte teórica

Pequena parte das aulas

Introdução ao tema da aula

Explicação dos conceitos



Parte prática

Maior parte das aulas

Implementação de código

Testes e depuração do programa

Experimentação com hardware

Análise dos resultados



renan.duarte@gedre.ufsm.br

Av. Roraima 1000 - Prédio 10 - Sala 440 - Santa Maria - RS



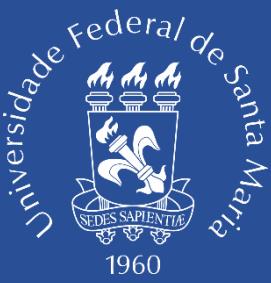
www.ufsm.br/gedre



+55 55 3220 9492



gedre.ufsm



UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
GEDRE - INTELIGÊNCIA EM ILUMINAÇÃO

gedre
inteligência
em iluminação

Aula 1 - Conceitos básicos

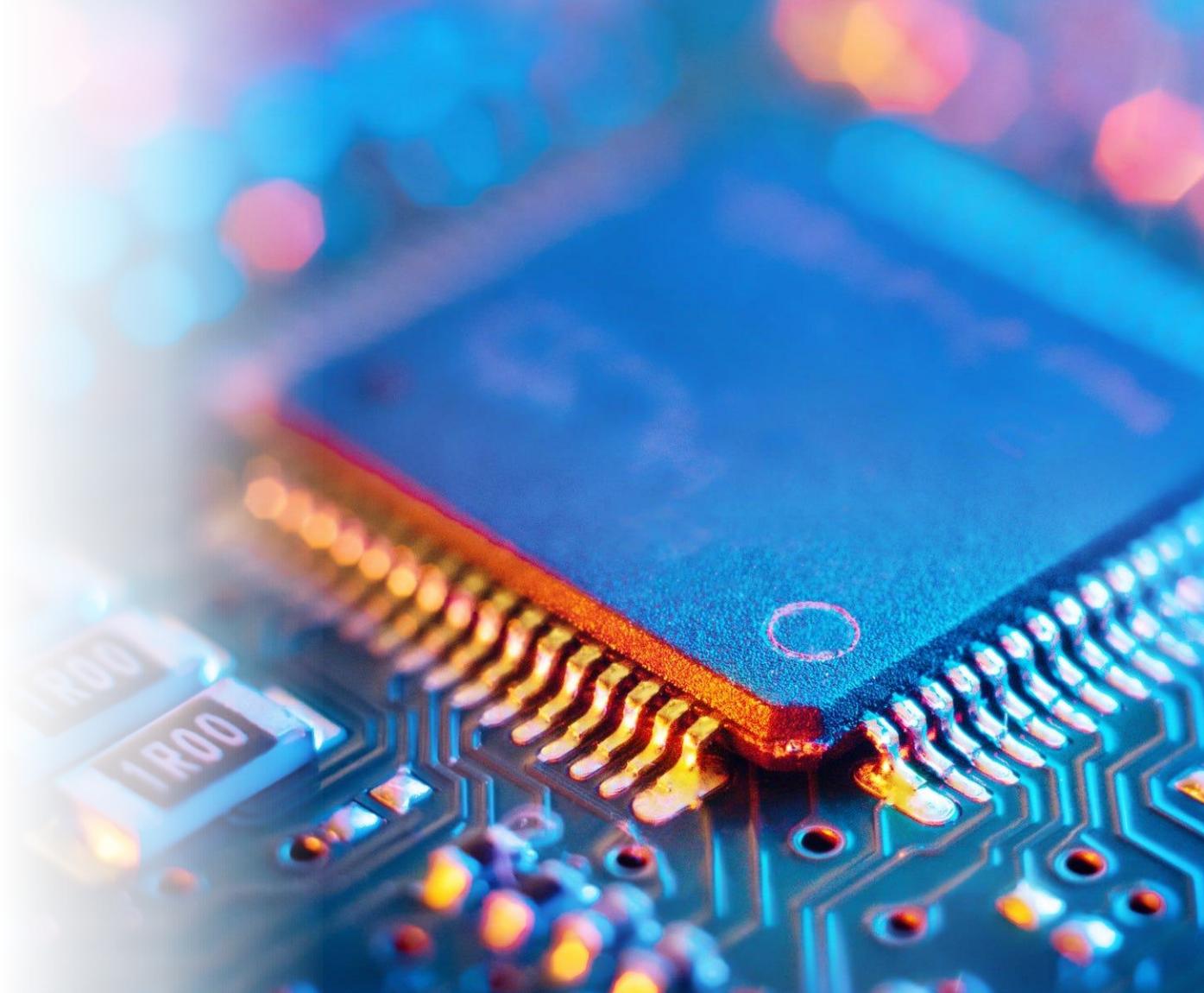
Prof. Renan Duarte

Minicurso de microcontroladores

Março de 2025

Conteúdo

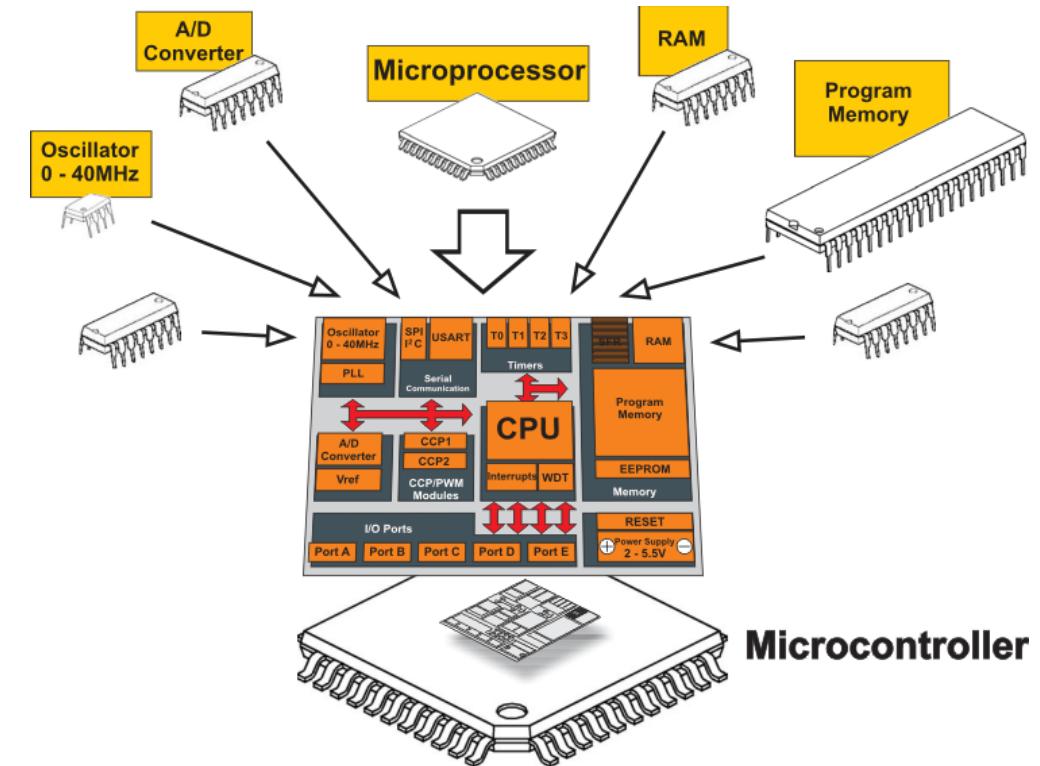
- Introdução
- Microcontrolador TM4C123GH6PM
- TM4C123G Launchpad
- Code Composer Studio
- Tiva Ware



O que é um microcontrolador?

Chip que integra CPU, memória e periféricos

- Executa tarefas específicas de controle e automação
- Pode ler entradas (sensores) e controlar saídas (motores, LEDs, etc.)
- Usado em dispositivos embarcados (carros, eletrodomésticos, IoT)
- Programado para operar de forma autônoma



O que é um microcontrolador?

Característica	Microprocessador	Microcontrolador
Função Primária	Computação de propósito geral	Controlo de sistemas embarcados específicos
Nível de Integração	CPU como componente principal; requer externos	CPU, memória e periféricos integrados num único chip
Memória	Principalmente externa ao chip da CPU	Integrada no mesmo chip da CPU
Periféricos	Externos ao chip da CPU	Integrados no mesmo chip da CPU
Flexibilidade	Alta; capaz de executar diversas tarefas	Limitada à tarefa específica para a qual foi programado
Consumo de Energia	Geralmente mais alto	Geralmente mais baixo
Custo	Geralmente mais alto (considerando o sistema completo)	Geralmente mais baixo
Aplicações Típicas	Computadores pessoais, servidores, etc.	Eletrodomésticos, automóveis, dispositivos IoT, etc.

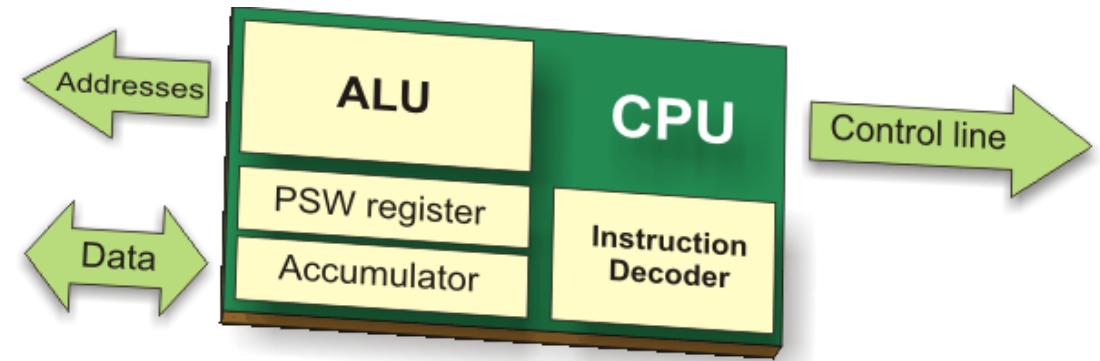
Componentes de um microcontrolador

Unidade central de processamento (CPU)

- Circuito responsável por executar as instruções do programa através de operações lógicas, aritméticas, de controle e entrada/saída

Funções principais da CPU

- Buscar (fetch) instruções da memória
- Decodificar as instruções
- Executar operações e controlar periféricos



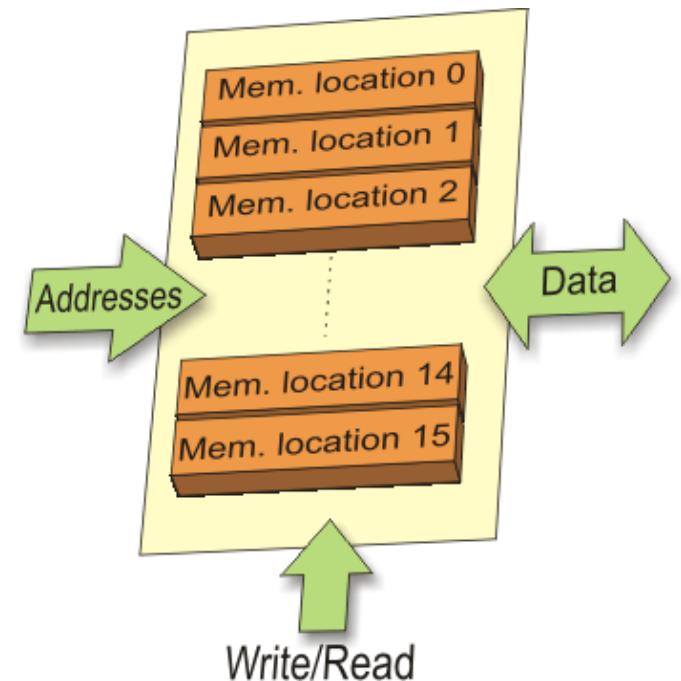
Componentes de um microcontrolador

Memória

- Espaço externo à CPU destinado a armazenar informações

Tipos de Memória

- Memória RAM: volátil, usada para dados temporários
- Memória Flash: não volátil, armazena o programa
- EEPROM: usada para armazenar configurações permanentes



Componentes de um microcontrolador

Registrador

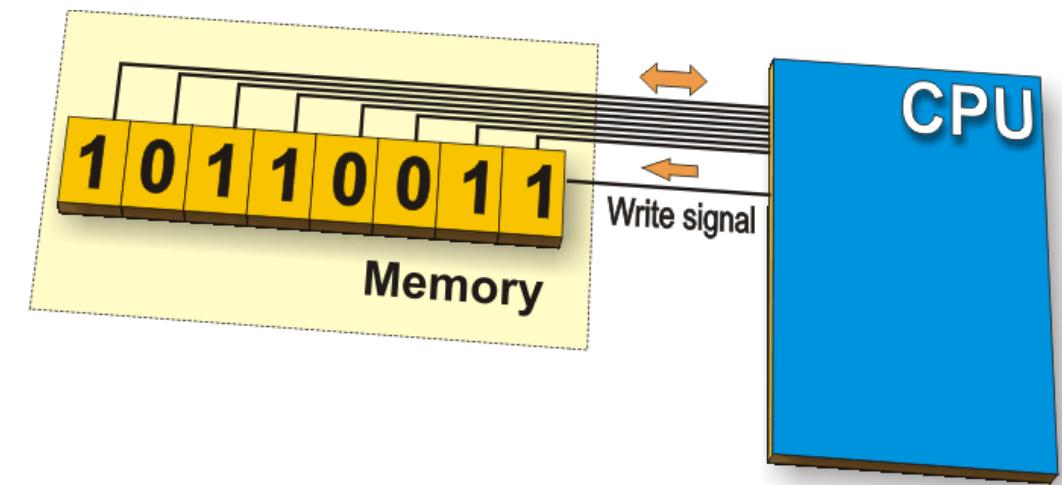
- Unidade de memória de n bits muito rápida dentro da CPU que armazena temporariamente dados e instruções em processamento

Funções principais

- Guardar operandos para operações aritméticas e lógicas
- Armazenar endereços de memória
- Controlar o estado do processamento (ex.: registrador de status)

Exemplos comuns de registradores:

- PC (Program Counter): guarda o endereço da próxima instrução
- IR (Instruction Register): guarda a instrução atual



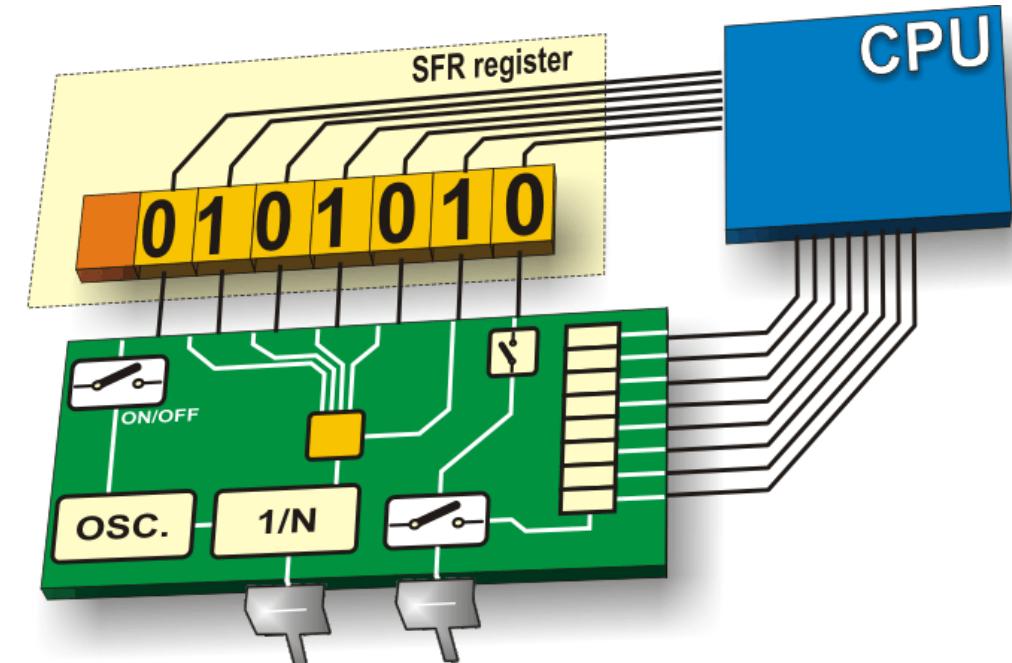
Componentes de um microcontrolador

Registrador de função especial (SFR)

- Registradores que controlam funções específicas do microcontrolador, como temporizadores, comunicação serial, interrupções, etc.

Exemplos de SFRs comuns

- TIMERx: configuração de temporizadores
- USARTx: controle da comunicação serial
- ADCx: controle do conversor analógico-digital
- PORTx: configuração de entradas e saídas digitais



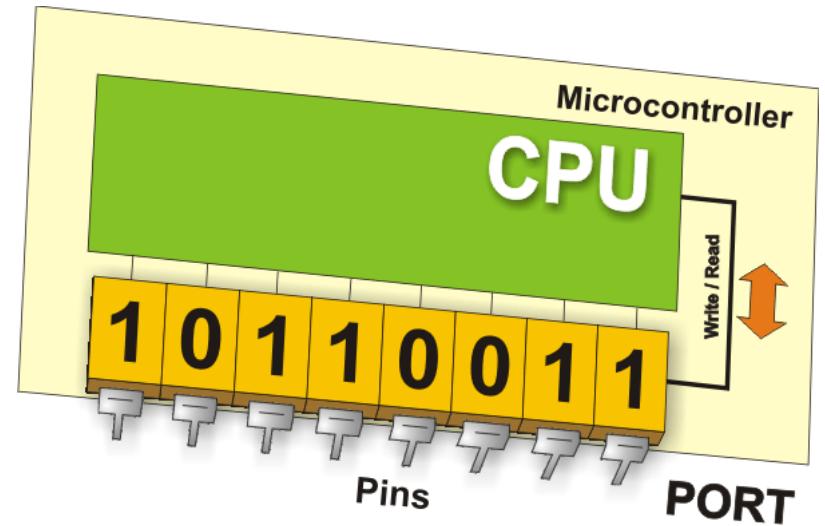
Componentes de um microcontrolador

Portas de entrada/saída (IOs ou GPIOs)

- Pinos genéricos de entrada (I) ou saída (O)
- Cada pino pode ser configurado como entrada (receber sinal) ou saída (enviar sinal)
- O software pode alterar dinamicamente o comportamento dos pinos

Exemplos de uso

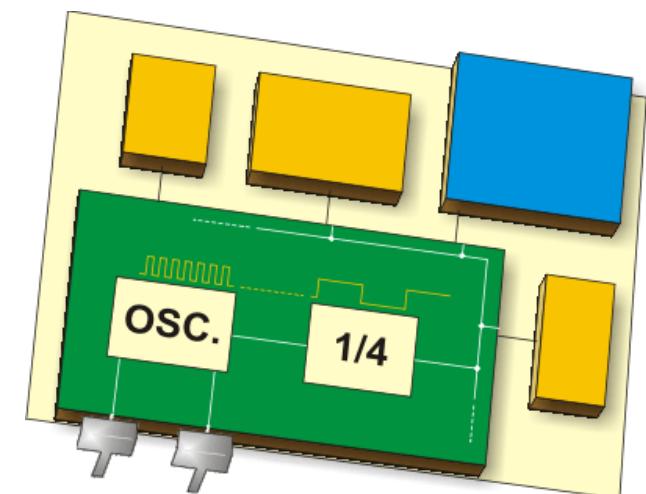
- LED
- Botões
- Sensores
- Motores



Componentes de um microcontrolador

Relógio de Sistema (System Clock)

- Frequência na qual as instruções do programa são executadas pela CPU
- A frequência do clock determina a velocidade de operação do microcontrolador
- Clocks mais rápidos = mais operações por segundo, mas também maior consumo de energia
- Um mesmo microcontrolador pode ter múltiplos clocks distintos para diferentes periféricos ou para diferentes momentos da operação (sleep, deepsleep, etc.)



Fabricantes



 山东华翼微电子技术股份有限公司
Shandong Huayi Micro-Electronics Technology Co.,Ltd.

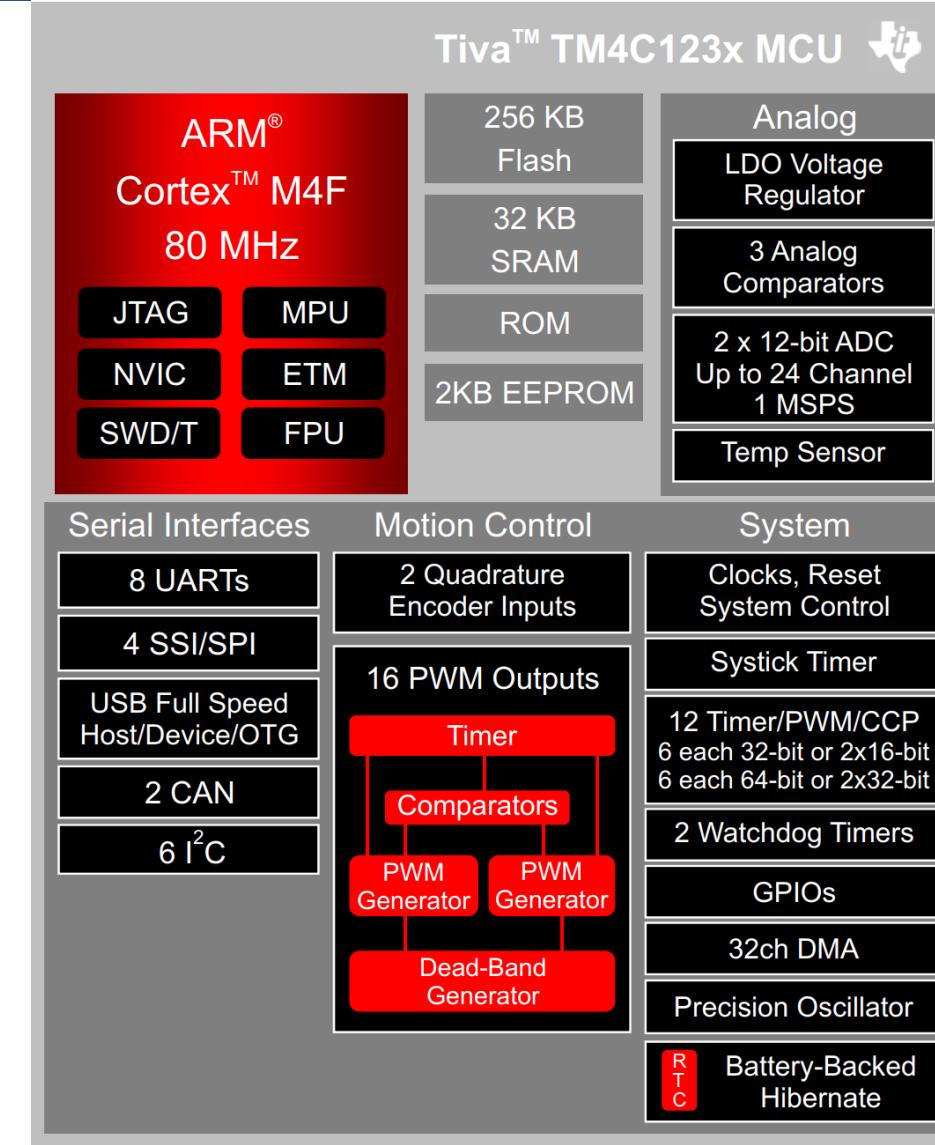


Microcontrolador TM4C123GH6PM

Parte da família TMC4C123x

- Núcleo ARM Cortex-M4F (80 MHz)
- Unidade de ponto flutuante (FPU)
- Memória Flash: 256 KB
- RAM: 32 KB
- EEPROM: 2 KB
- GPIOs: 43 pinos

<https://www.ti.com/lit/ds/symlink/tm4c123gh6pm.pdf>

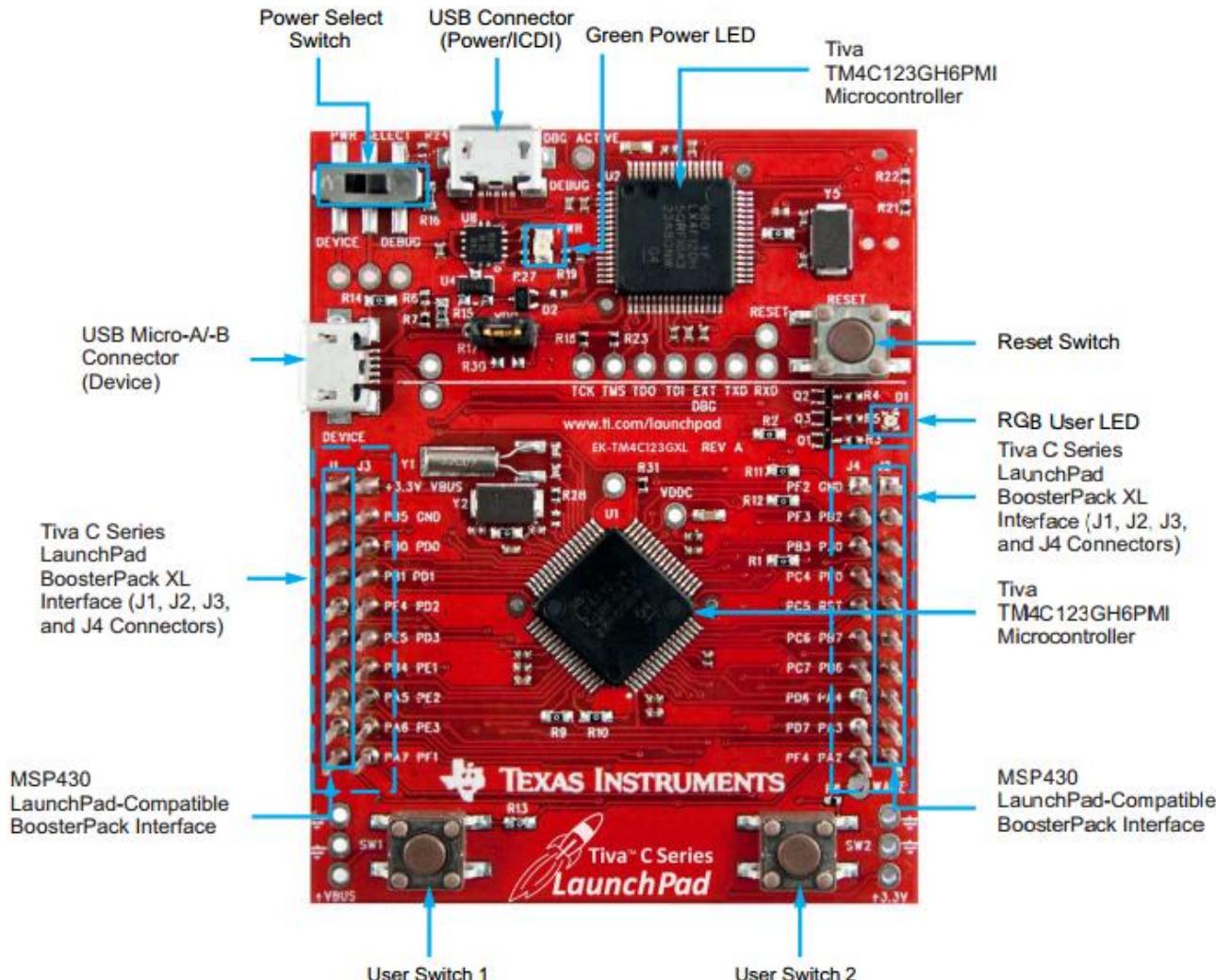


TM4C123G Launchpad

Placa de desenvolvimento

- Ideal para aprendizado, prototipagem e projetos embarcados
- Interface de programação e depuração embutida (ICDI - In-Circuit Debug Interface)
- Acesso fácil aos pinos do microcontrolador (headers laterais)
- LEDs e push-buttons integrados para testes rápidos
- Alimentação via USB ou fonte externa
- Compatível com TivaWare e diversas IDEs (Code Composer Studio, Keil, etc.)

<https://www.ti.com/lit/pdf/spmu296>



Code Composer Studio

Software oficial da Texas Instruments



- Gratuito e sem limitações de memória
- Compatível com toda a linha de microcontroladores e DSPs da Texas
- Permite programação, depuração e análise de desempenho em um único ambiente
- Baseado na IDE Eclipse
 - Interface bastante similar à de vários outros softwares baseados em eclipse

O que é?

- Biblioteca de funções concebida para simplificar e acelerar o desenvolvimento de projetos nos microcontroladores da linha Tiva

Características

- Vasta documentação
- Funções já gravadas na memória ROM do microcontrolador (não ocupam “espaço” de código)

<https://www.ti.com/lit/ug/spmu298e/spmu298e.pdf>

Referências essenciais

Sempre que vamos trabalhar com qualquer microcontrolador, devemos estudar:

- Datasheet do microcontrolador - Parâmetros elétricos, mapas de registradores, etc.
<https://www.ti.com/lit/ds/symlink/tm4c123gh6pm.pdf>
- Manual da placa de desenvolvimento (se houver) - Conexões, recursos, etc.
<https://www.ti.com/lit/pdf/spmu296>
- Manual da SDK utilizada (se utilizada) - Funções, parâmetros, retornos, etc.
<https://www.ti.com/lit/ug/spmu298e/spmu298e.pdf>





renan.duarte@gedre.ufsm.br

Av. Roraima 1000 - Prédio 10 - Sala 440 - Santa Maria - RS



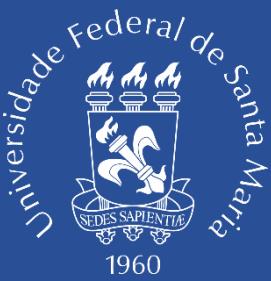
www.ufsm.br/gedre



+55 55 3220 9492



gedre.ufsm



UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
GEDRE - INTELIGÊNCIA EM ILUMINAÇÃO

gedre
inteligência
em iluminação

Aula 2 - Instalação dos softwares

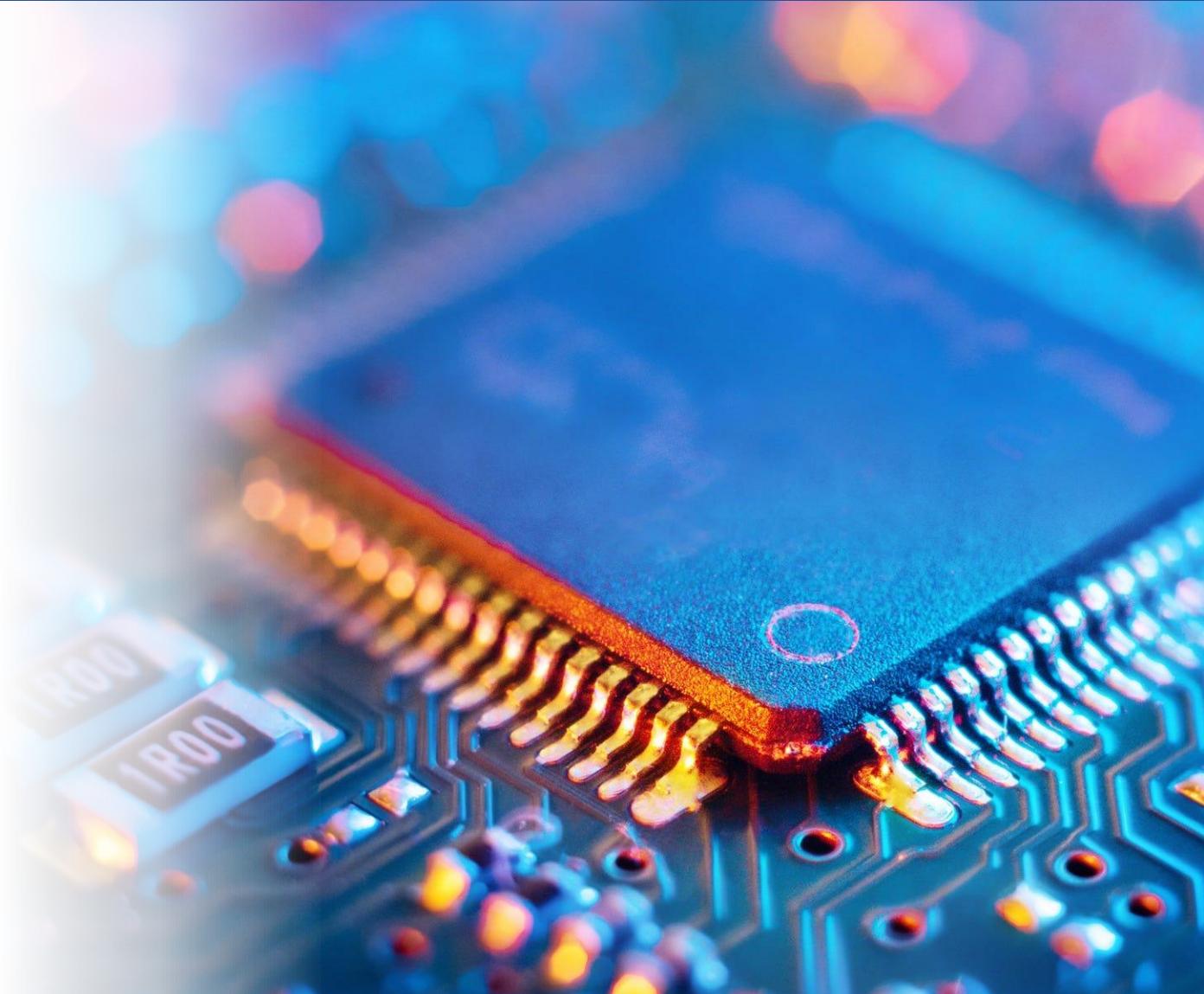
Prof. Renan Duarte

Minicurso de microcontroladores

Março de 2025

Conteúdo

- Links para download
- Instalação Code Composer Studio
- Instalação TivaWare



Links para download

Code Composer Studio

- <https://www.ti.com/tool/CCSTUDIO>

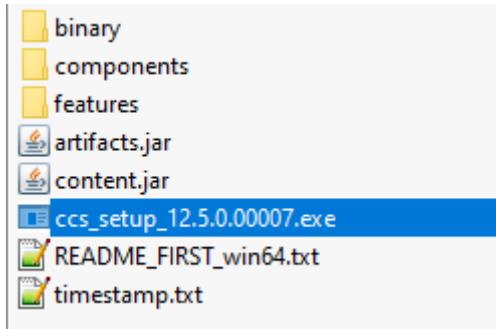
SW-TM4C – TivaWare for C Series Software (Requer login e formulário)

- <https://www.ti.com/tool/SW-TM4C>

Instalação Code Composer Studio

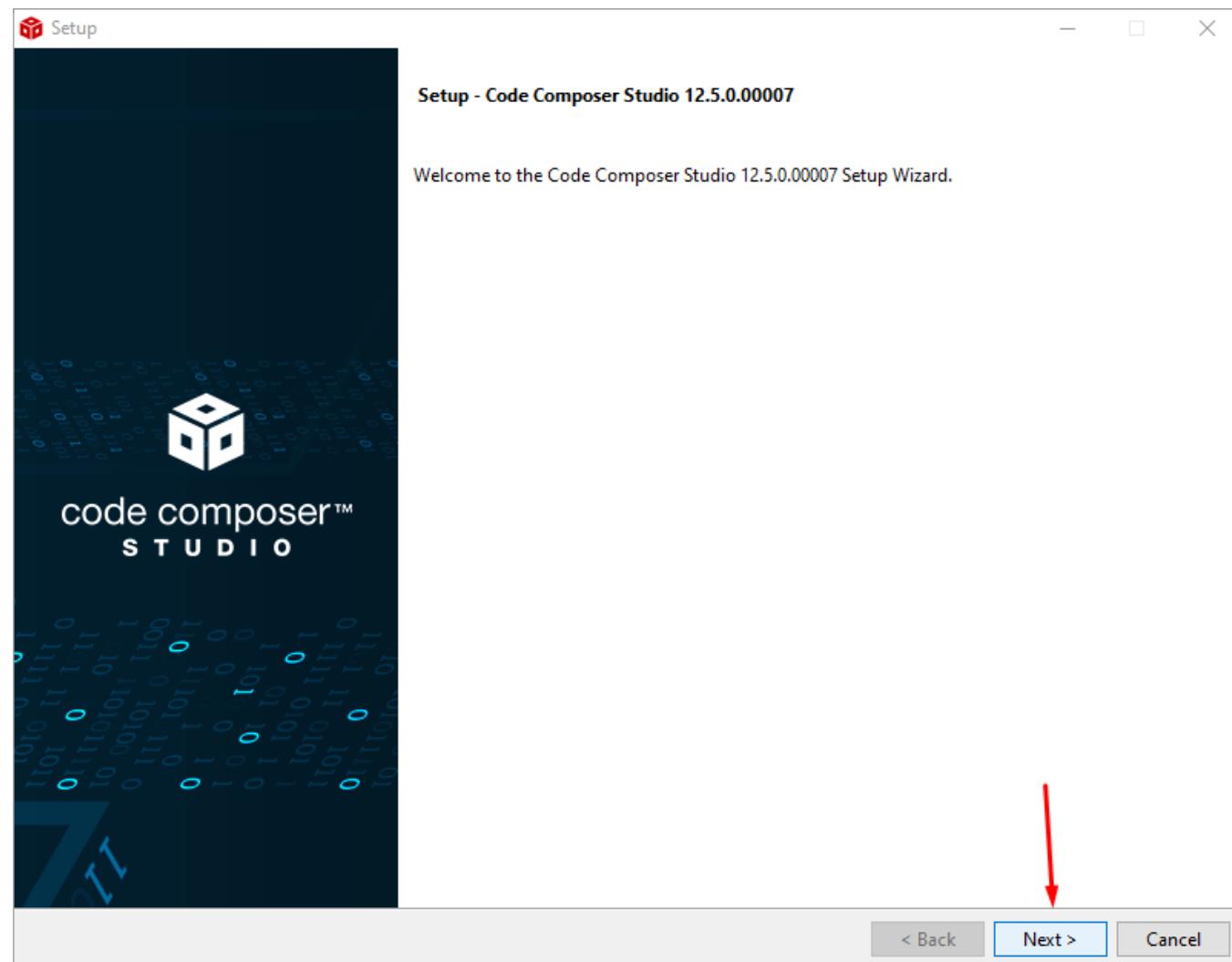
Observação

Para instalar o Code Composer Studio, é necessário extrair o conteúdo do arquivo zip e executar o setup:

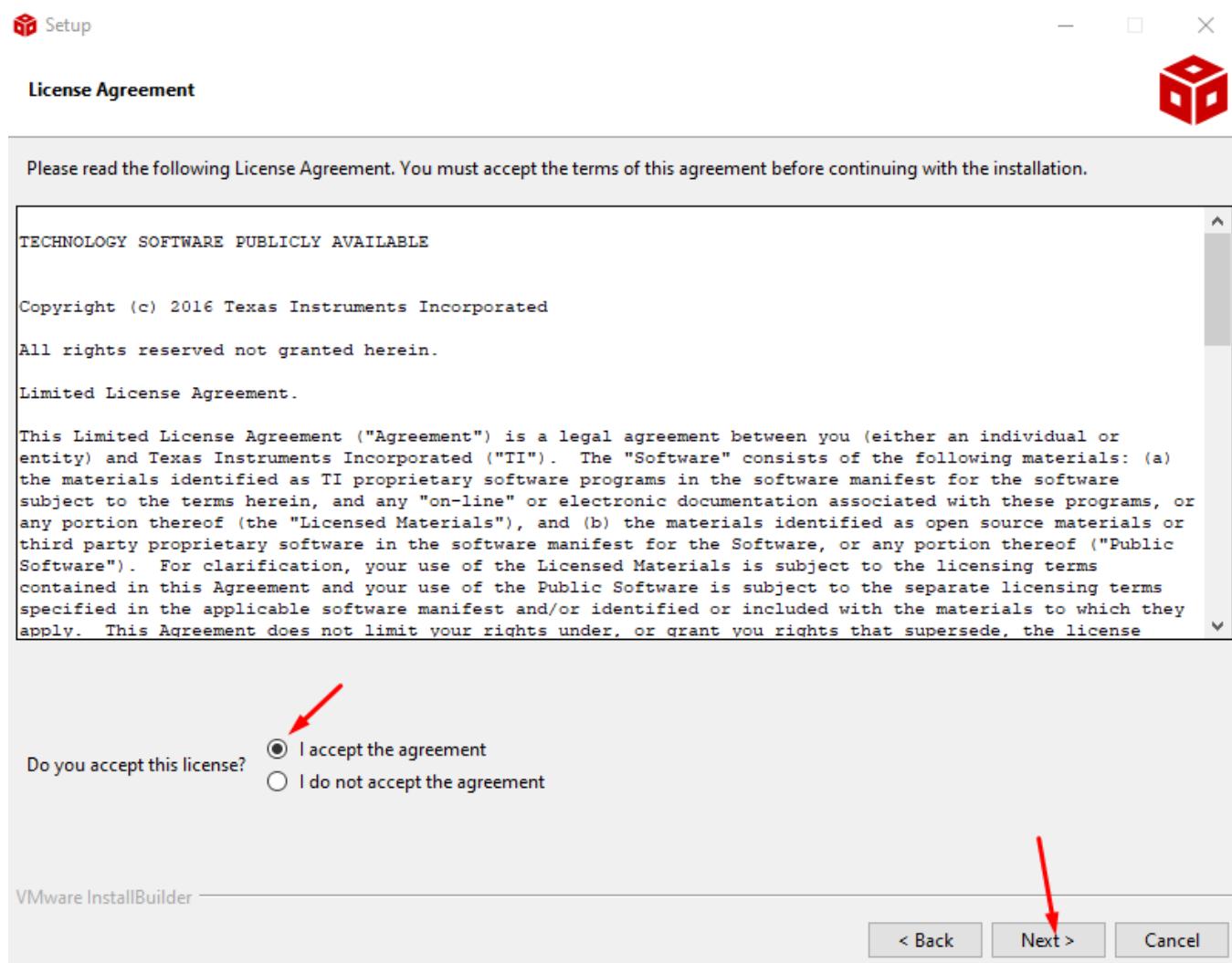


Executar direto do arquivo zip não funcionará!

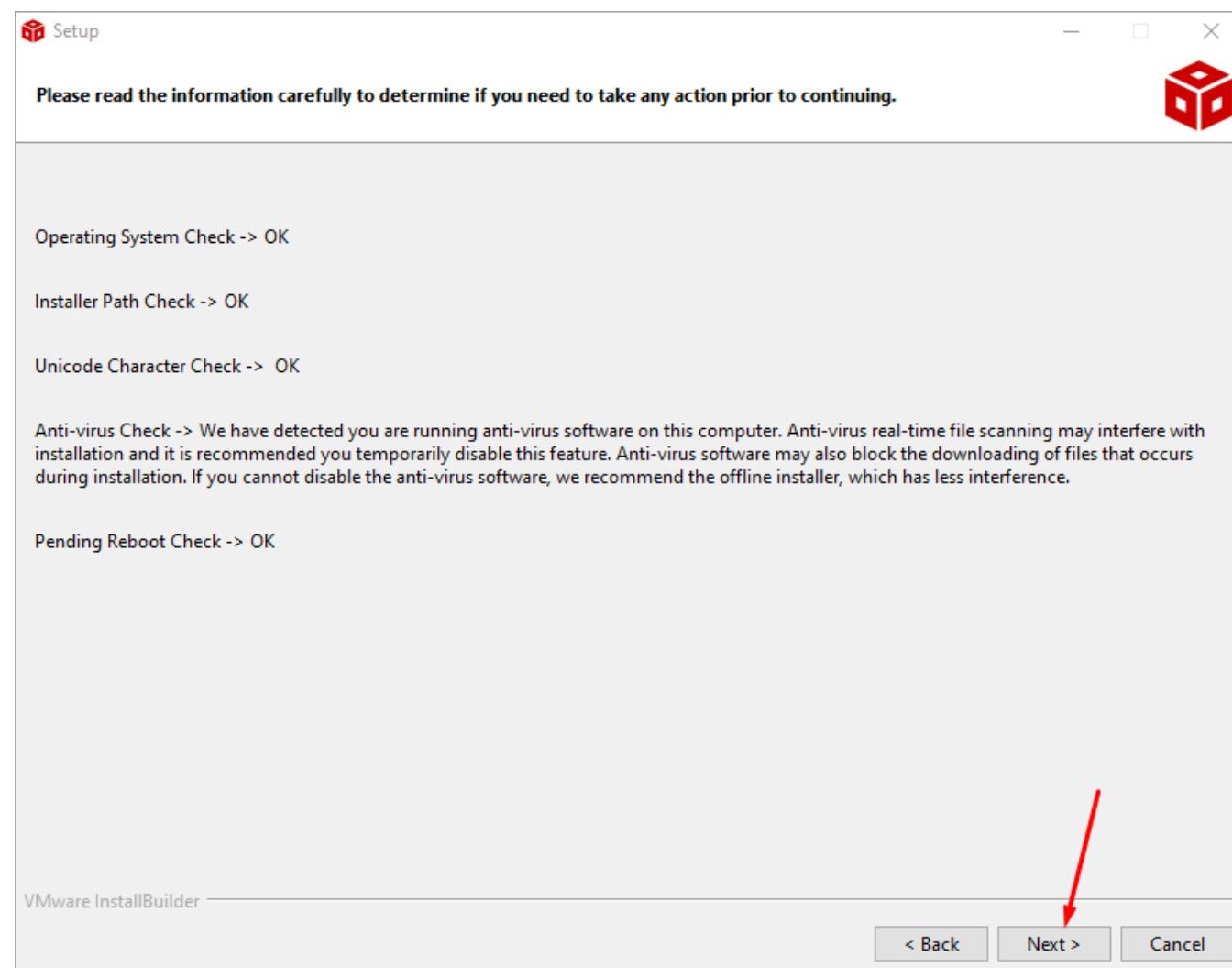
Instalação Code Composer Studio



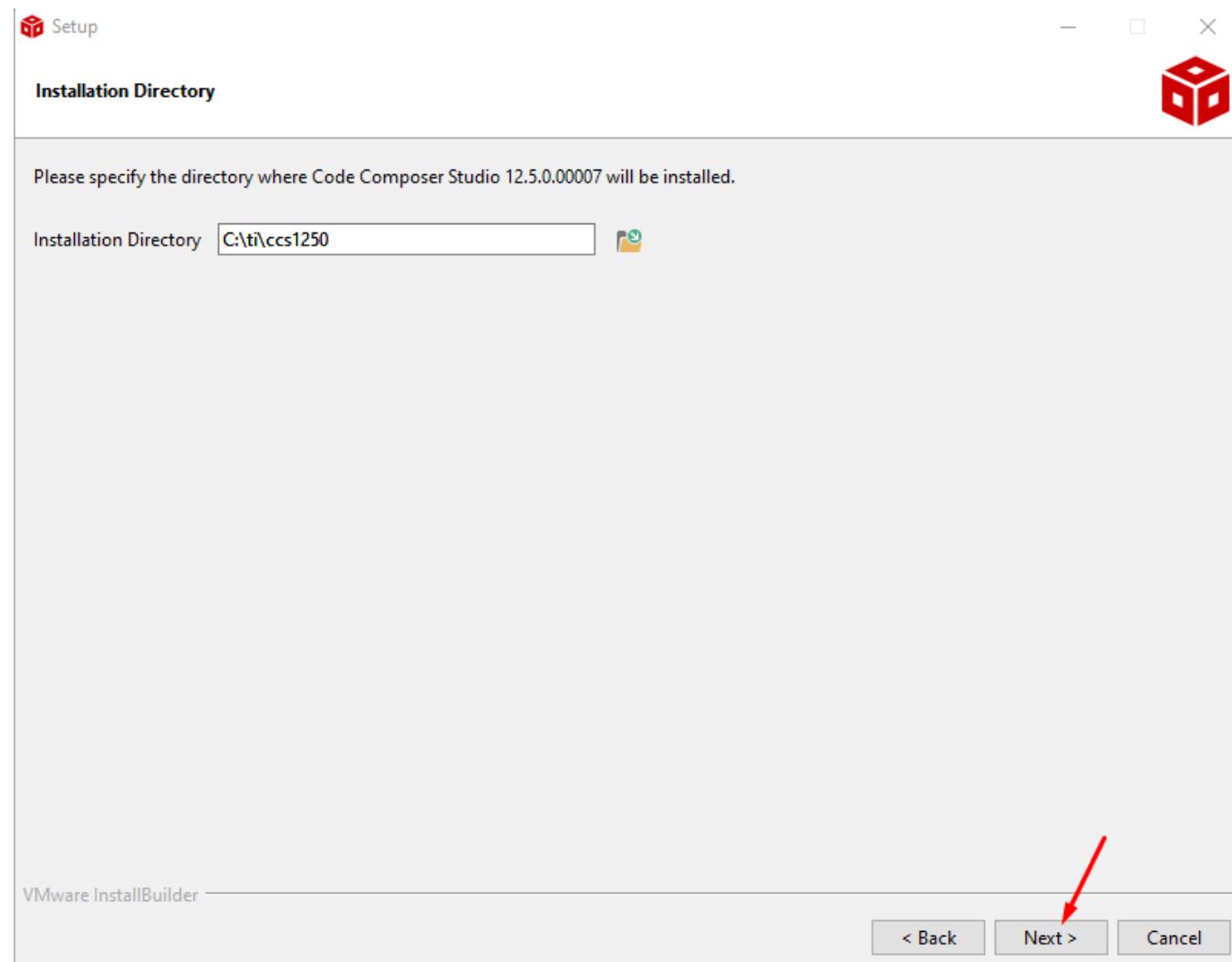
Instalação Code Composer Studio



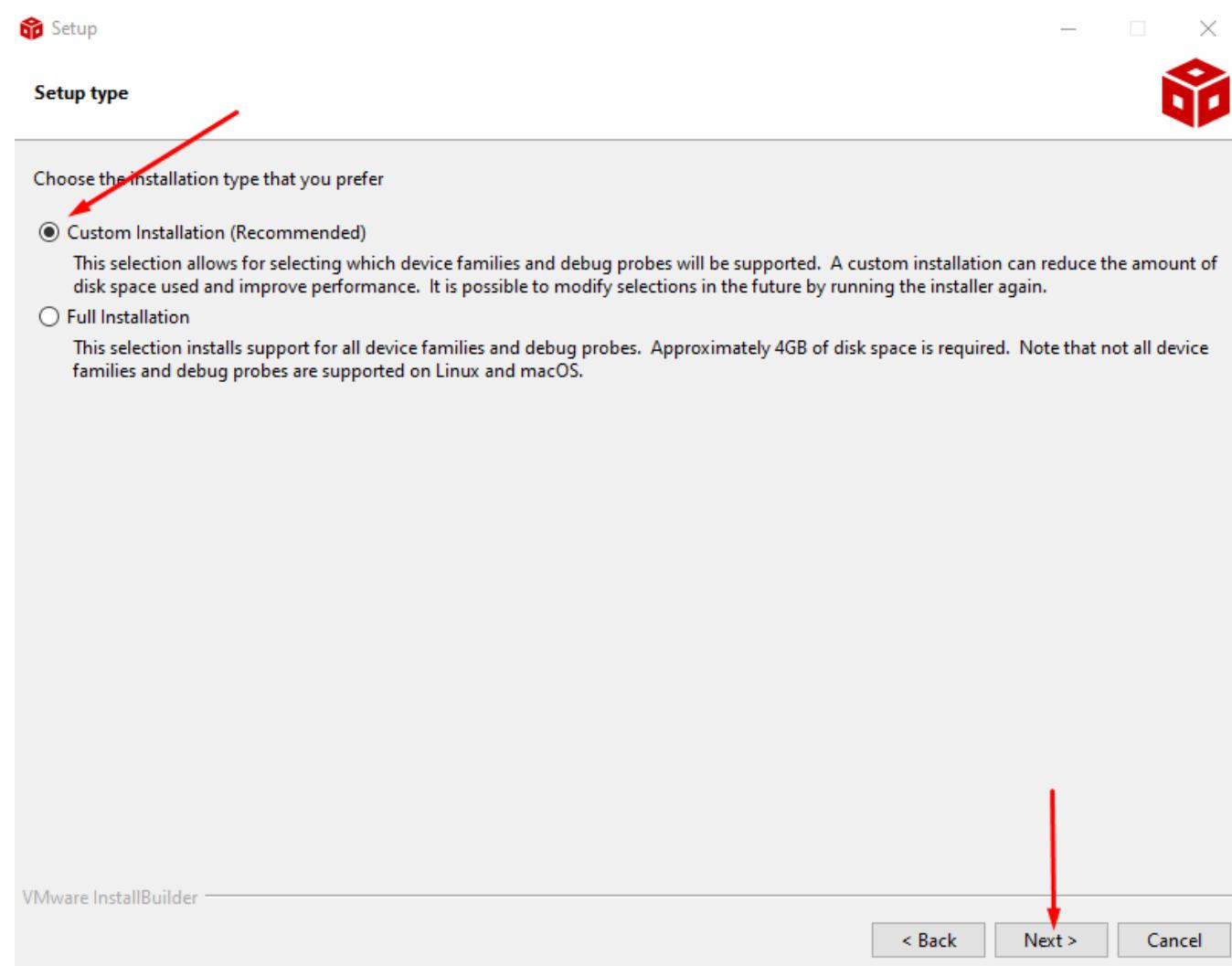
Instalação Code Composer Studio



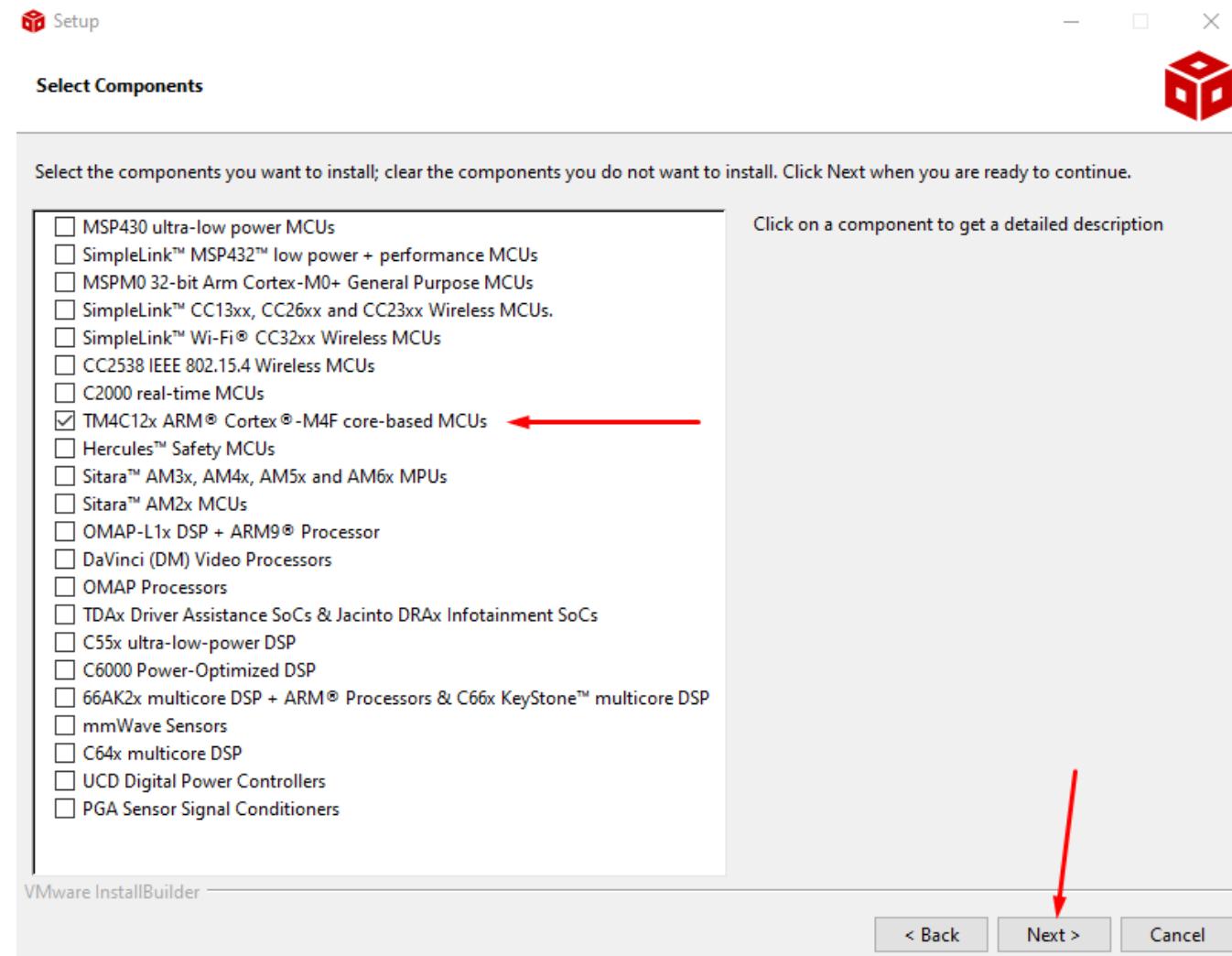
Instalação Code Composer Studio



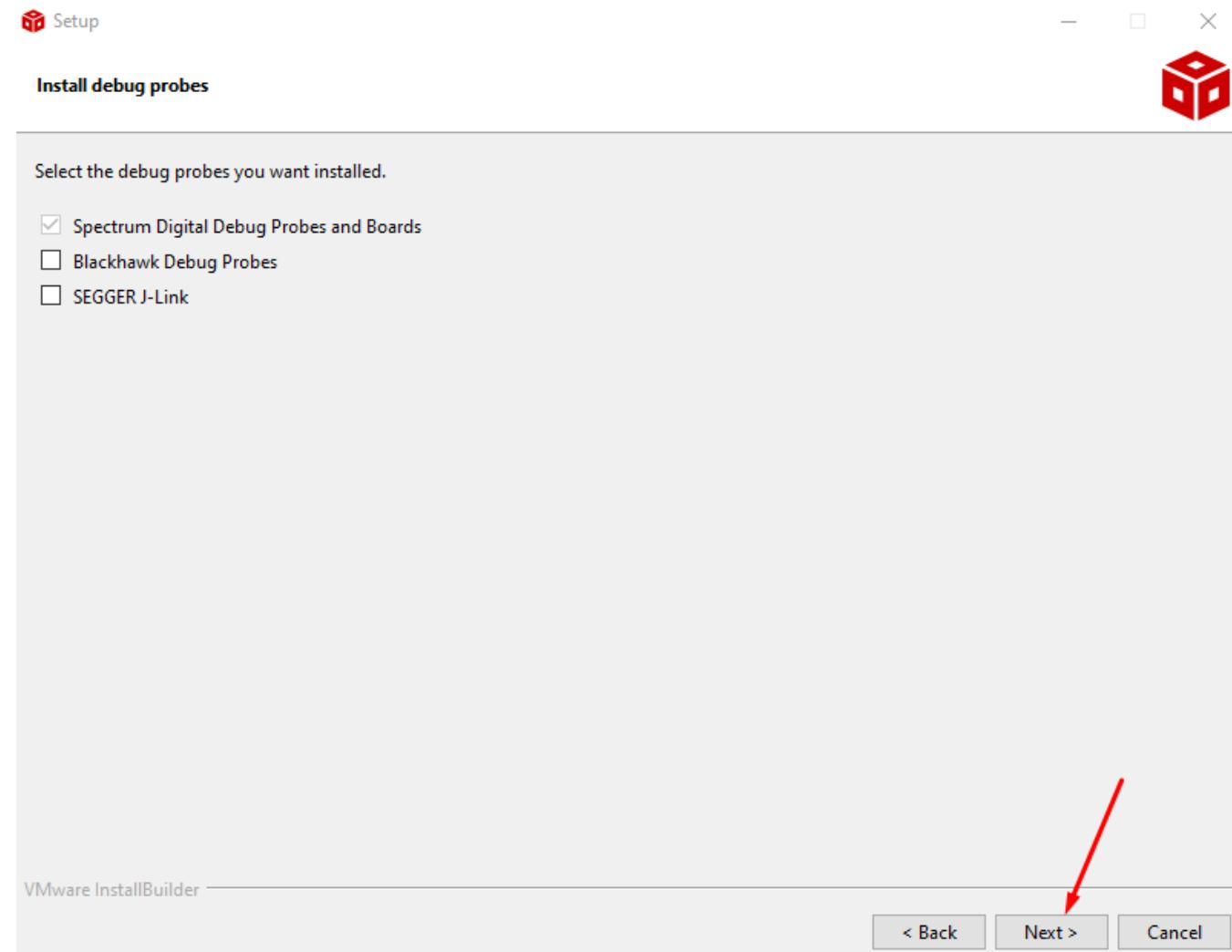
Instalação Code Composer Studio



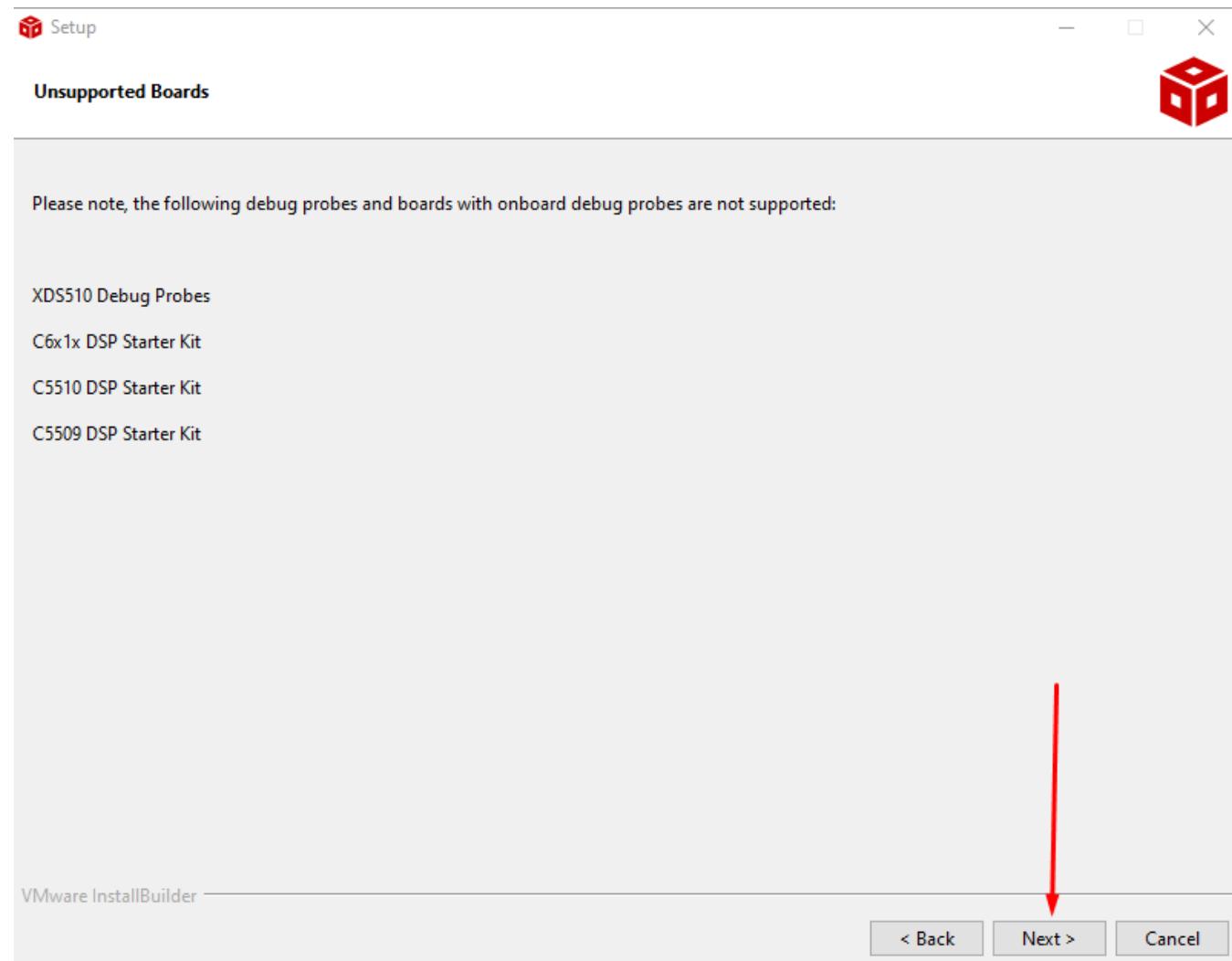
Instalação Code Composer Studio



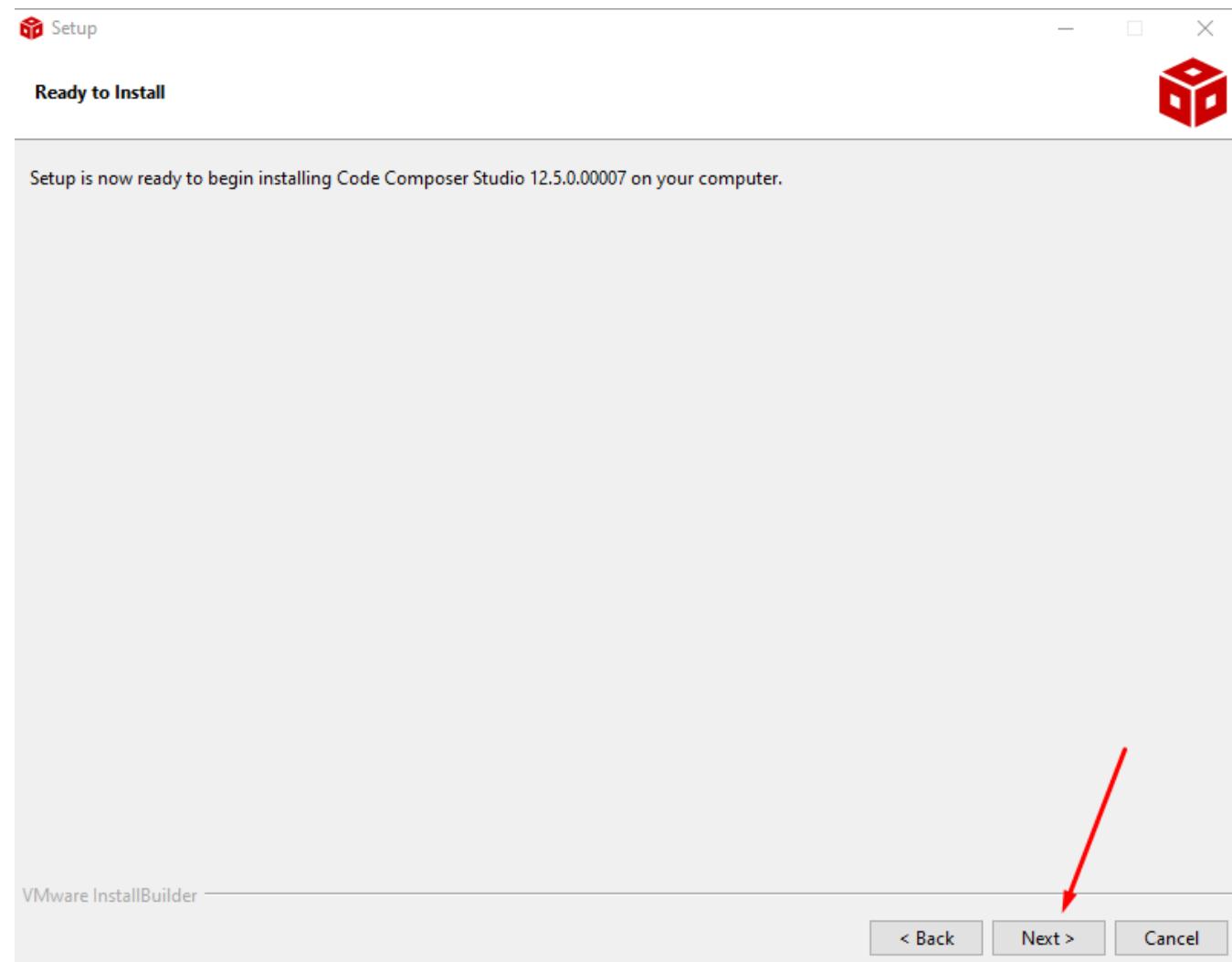
Instalação Code Composer Studio



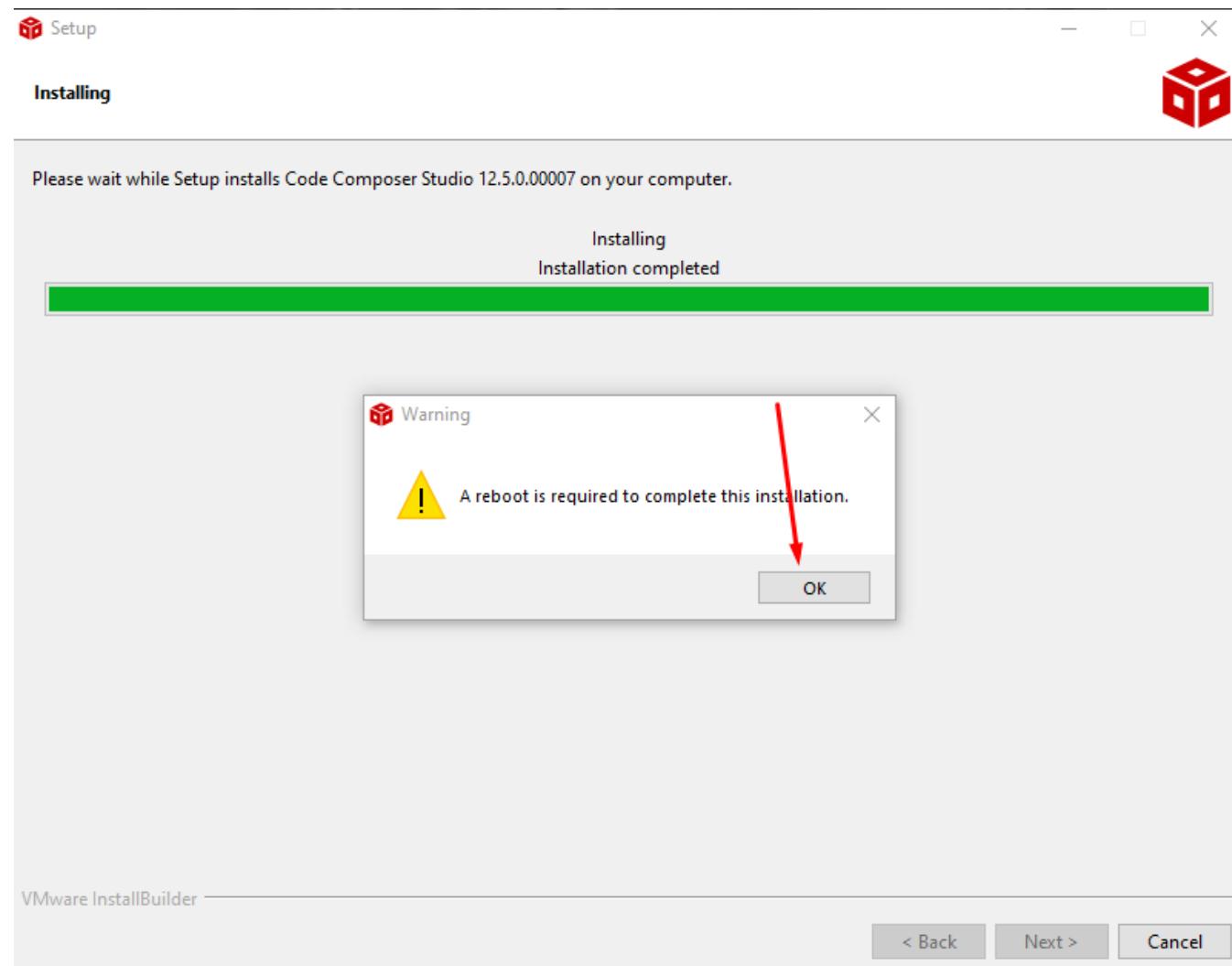
Instalação Code Composer Studio



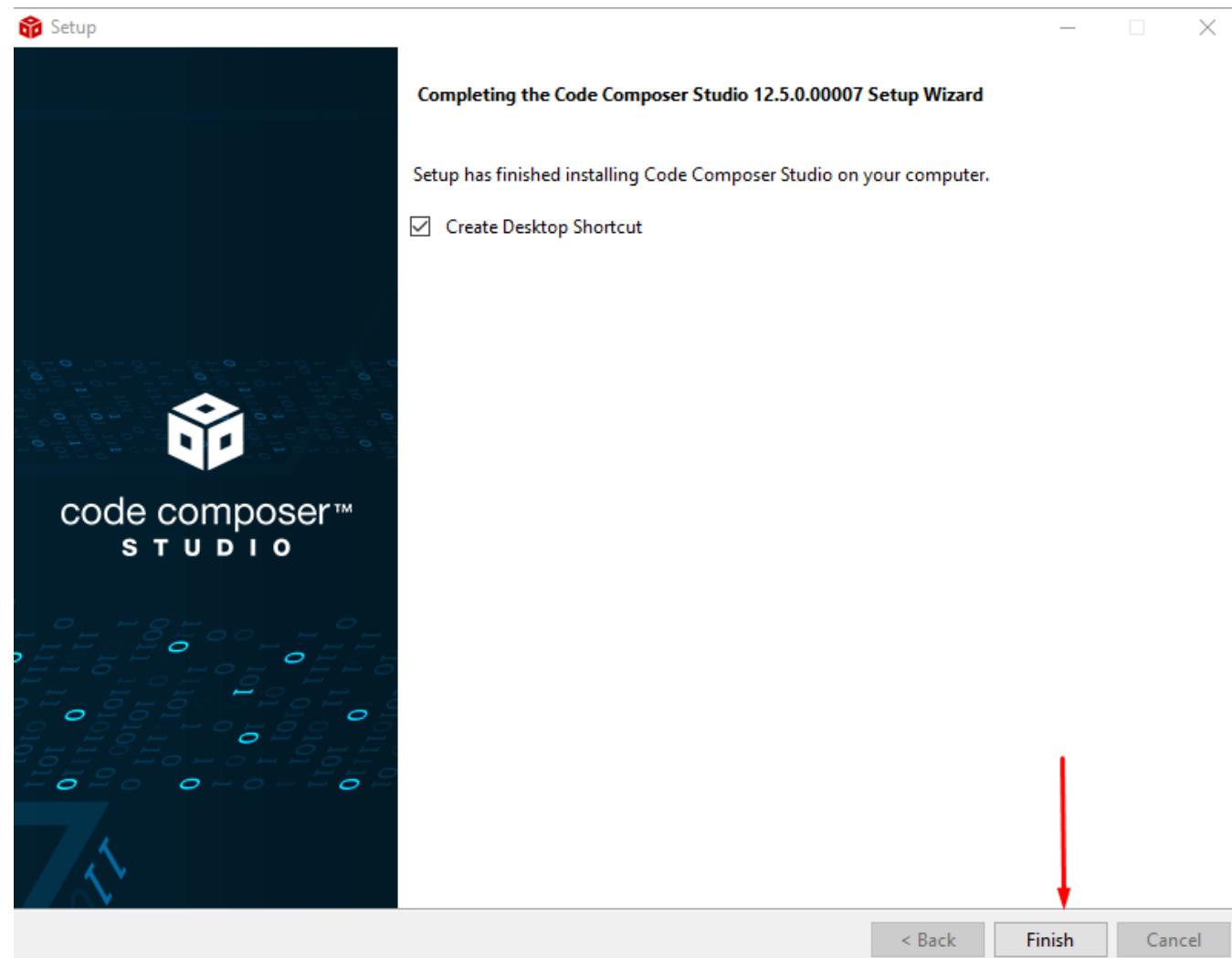
Instalação Code Composer Studio



Instalação Code Composer Studio



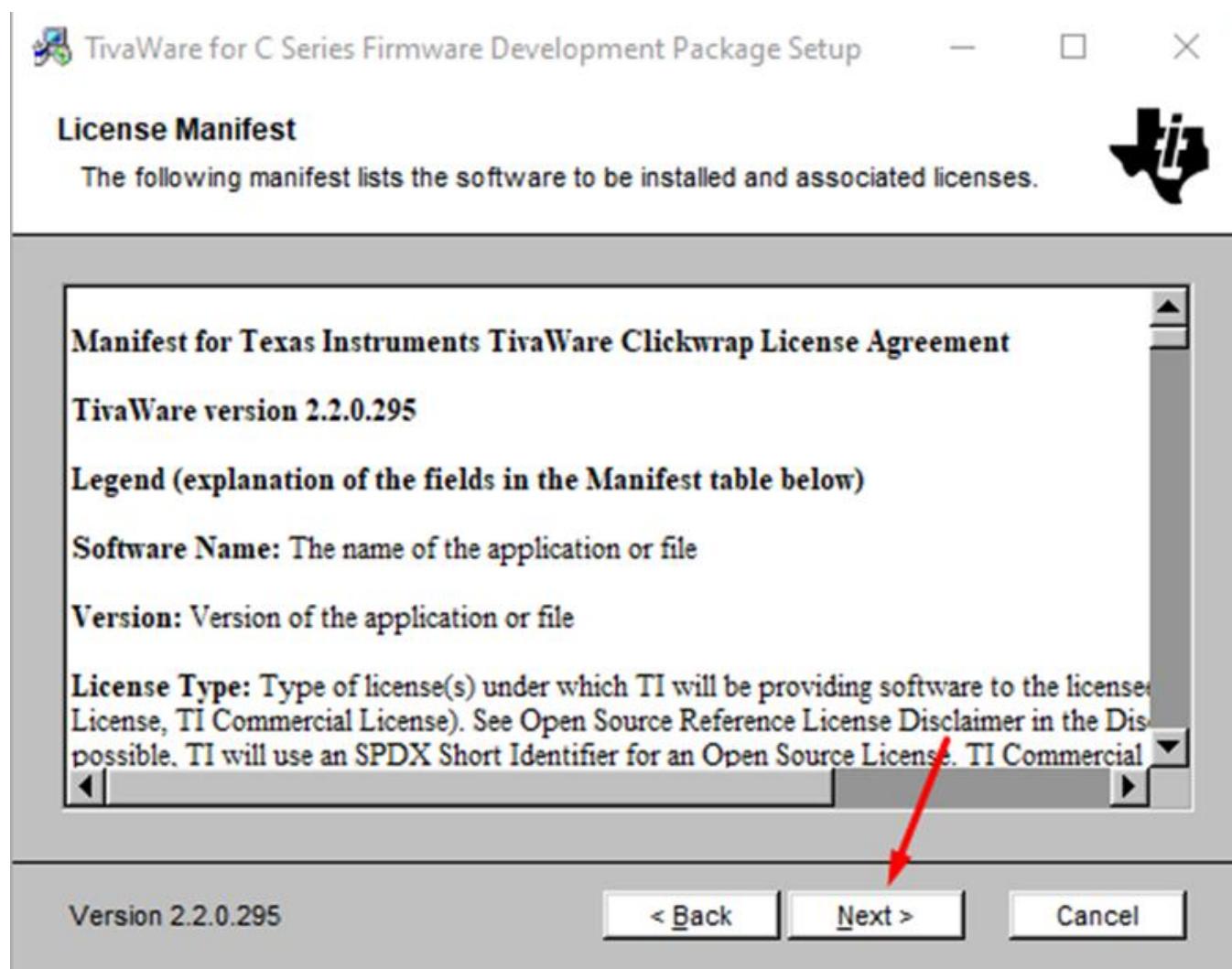
Instalação Code Composer Studio



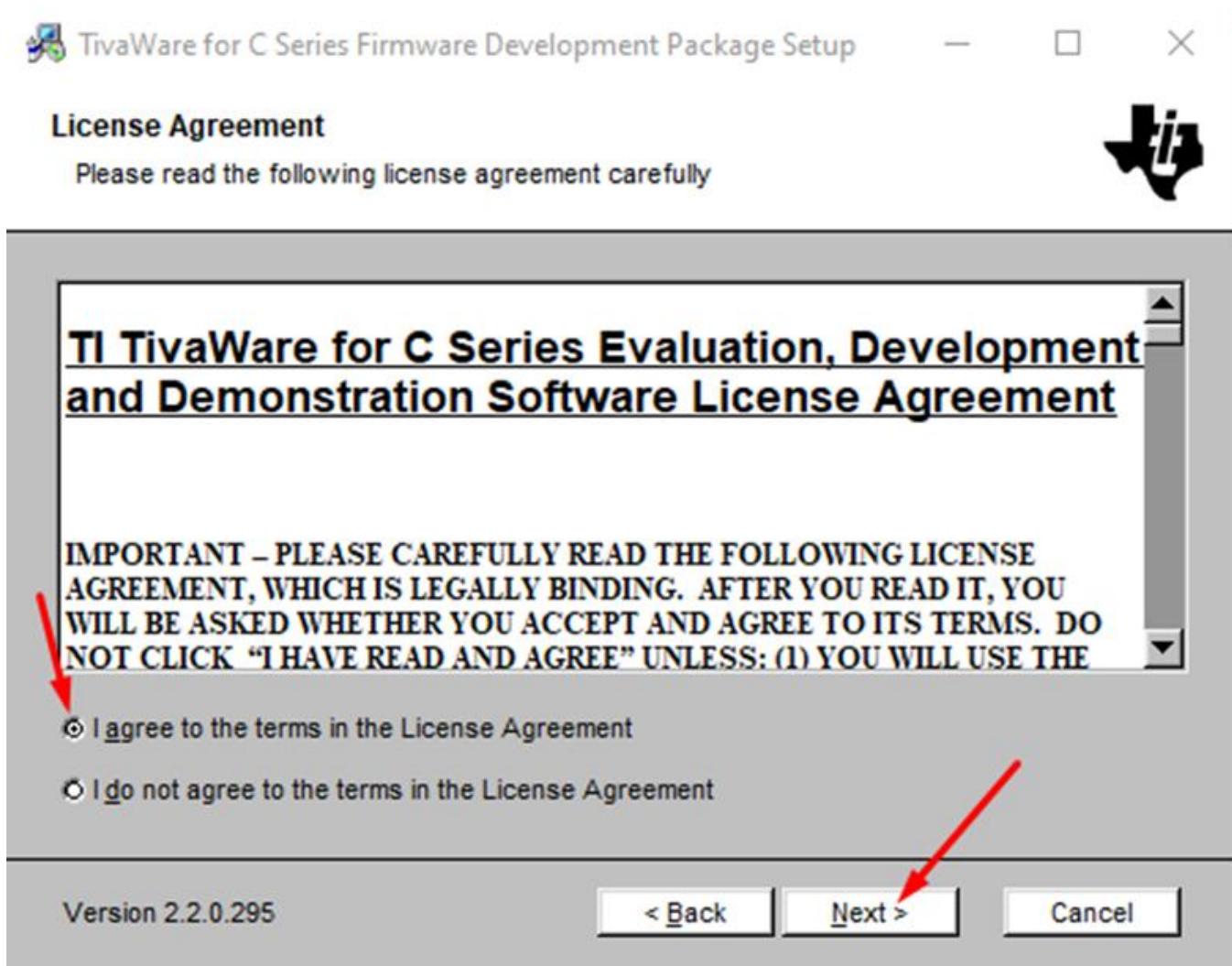
Instalação TivaWare



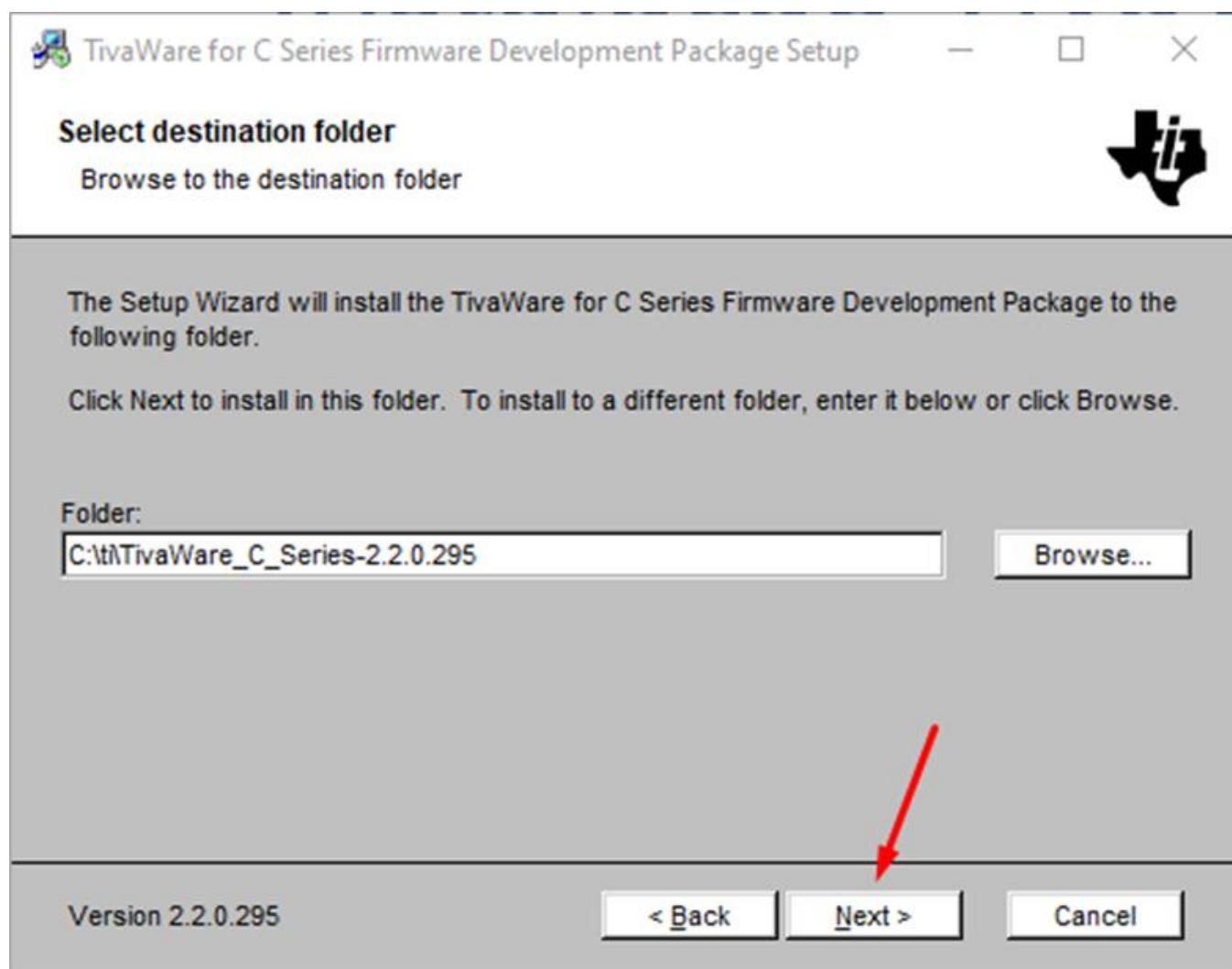
Instalação TivaWare



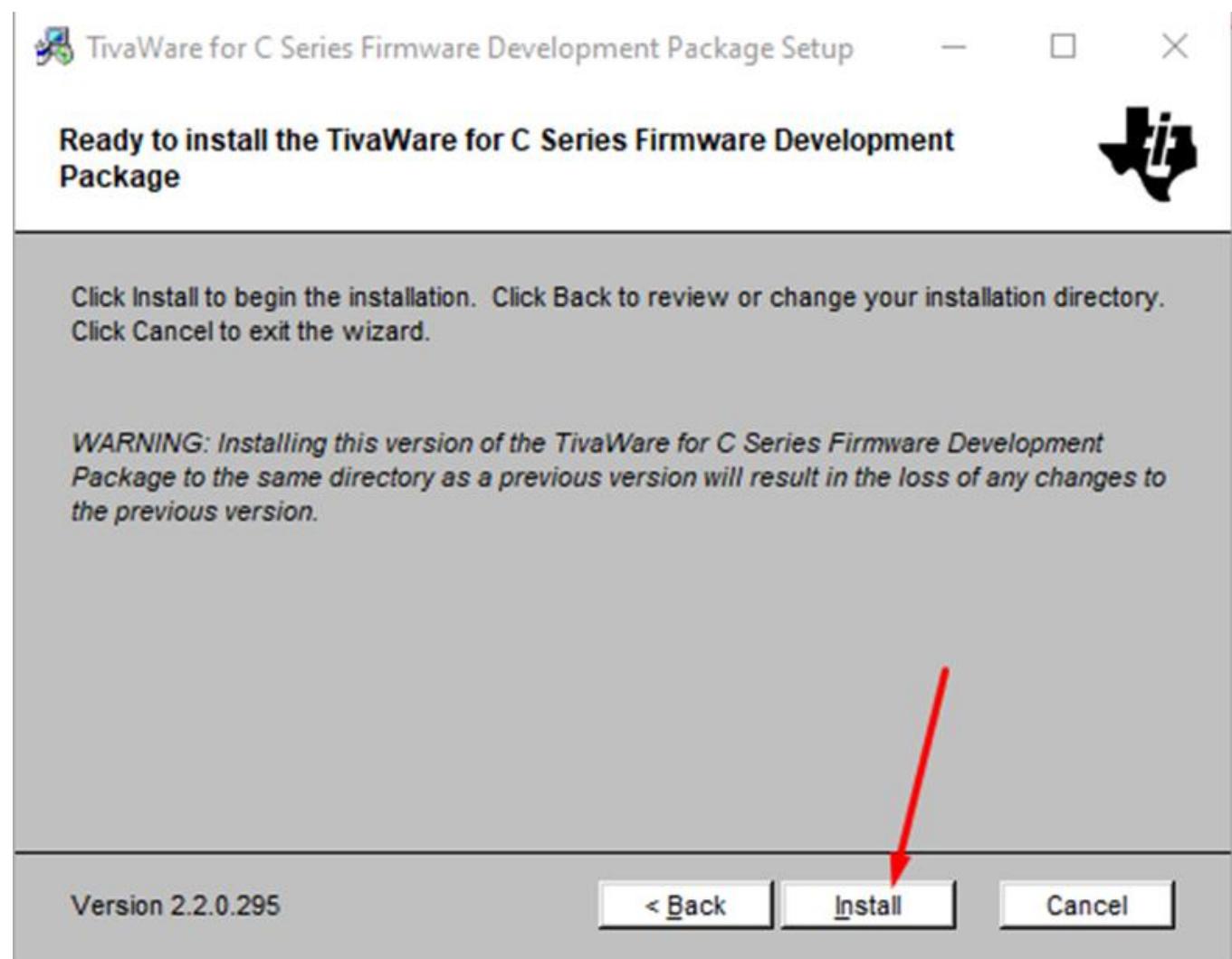
Instalação TivaWare



Instalação TivaWare



Instalação TivaWare



Instalação TivaWare





renan.duarte@gedre.ufsm.br

Av. Roraima 1000 - Prédio 10 - Sala 440 - Santa Maria - RS



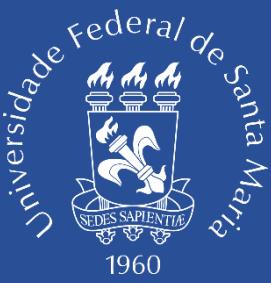
www.ufsm.br/gedre



+55 55 3220 9492



gedre.ufsm



UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
GEDRE - INTELIGÊNCIA EM ILUMINAÇÃO

gedre
inteligência
em iluminação

Aula 3 - Projeto Base

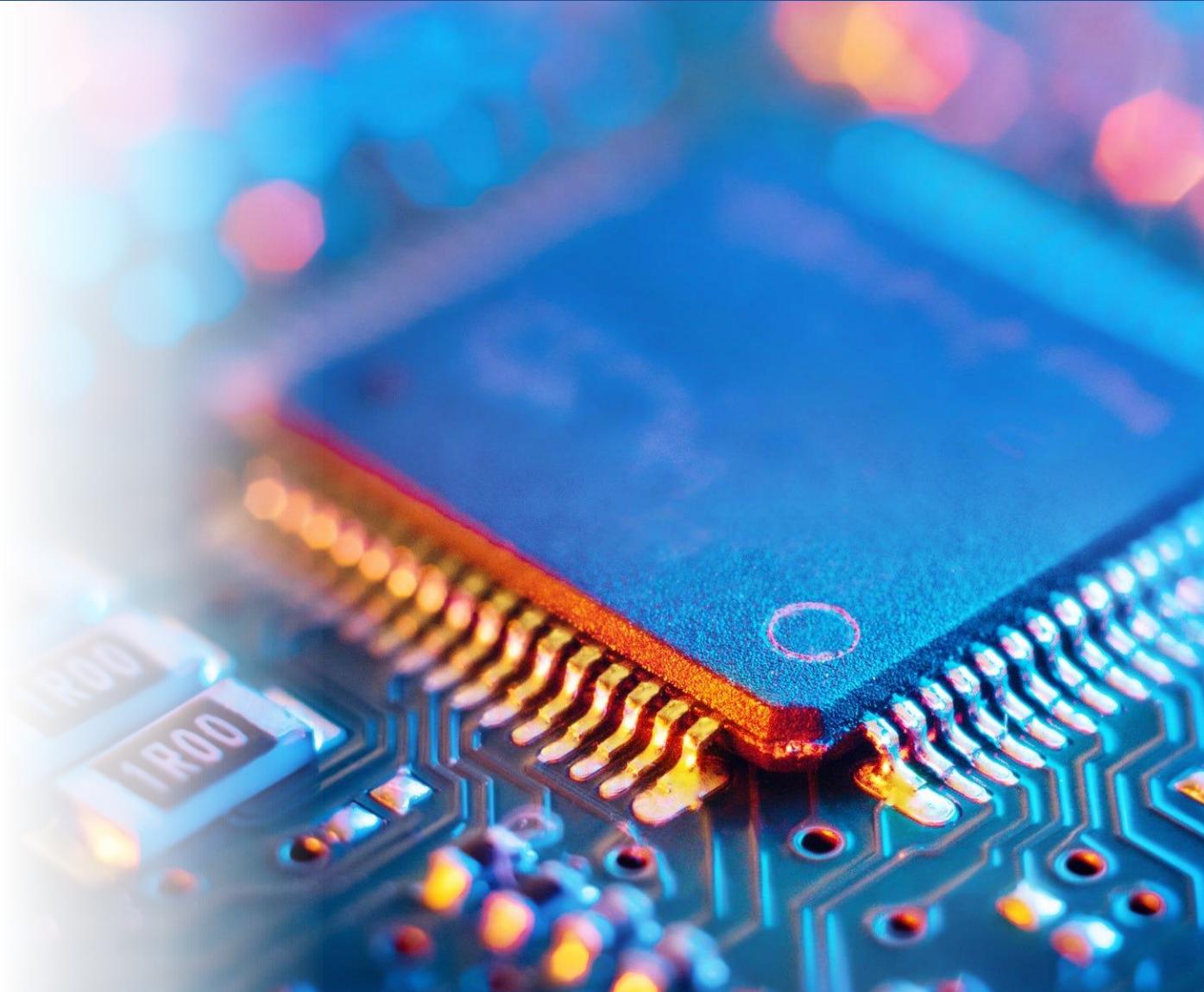
Prof. Renan Duarte

Minicurso de microcontroladores

Março de 2025

Conteúdo

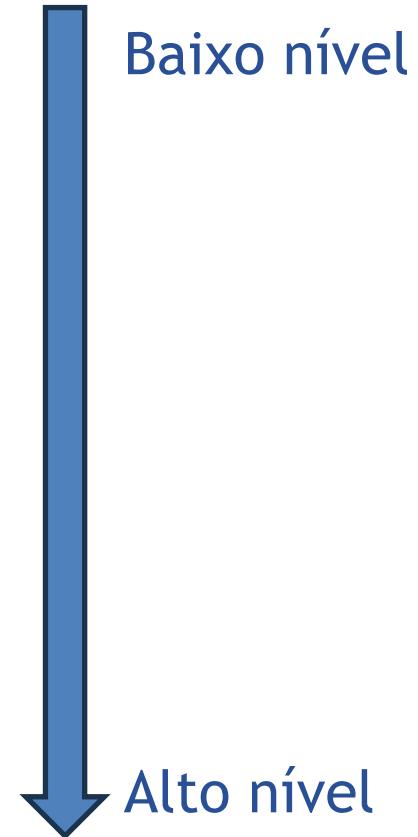
- Visão geral da IDE
- Criando o Projeto Base
- Configurando as bibliotecas
- Compilando e testando



Programação de Microcontroladores

Níveis de Abstração

1. Código de máquina
2. Assembly
3. C (ou CPP) + endereços de memória
4. C (ou CPP) + arquivo header do microcontrolador
5. C (ou CPP) + biblioteca de funções (SDK)



Referência interessante: <https://youtu.be/9FTUa-2eIDU>

Programação de Microcontroladores

Níveis de Abstração

Quanto mais baixo o nível, mais controle e eficiência, mas mais difícil e propenso a erro.

Quanto mais alto o nível, mais fácil e rápido, mas às vezes menos eficiente.

Programação de Microcontroladores

1. Código de máquina

O que é?

- Programar diretamente no formato de código de máquina (binário/hexadecimal) que o processador entende

Como funciona?

- Cada instrução do processador é representada por uma sequência de bits ([OP codes](#))
- Essas sequências podem ser representadas em hexadecimal (ex: 0xE3A02001)

Por que é difícil escrever à mão?

- Cada instrução precisa ser convertida manualmente para seu código binário/hexadecimal
- Exige conhecimento profundo do conjunto de instruções da CPU
- Um erro em um único bit pode fazer o programa falhar completamente
- Programar assim é extremamente lento, complexo e propenso a erros

Programação de Microcontroladores

2. Assembly

Em assembly, precisamos conhecer cada instrução que a CPU é capaz de executar e o endereço de memória dos registradores do microcontrolador:

```
; Inicializa o clock para GPIOF  
LDR R1, =0x400FE108 ; Carrega o endereço do registrador em R1  
MOVS R0, #32          ; Move o valor 32 (100000) para R0  
STR R0, [R1]           ; Armazena o valor de R0 no endereço apontado por R1
```

Série de vídeos muito interessantes sobre o assunto:

<https://www.youtube.com/watch?v=LnzuMJLZRdU&list=PLowKtXNTBypFbtuVMUVXNR0z1mu7dp7eH>

Programação de Microcontroladores

3. C (ou CPP) + endereços de memória

Aqui já utilizamos uma linguagem mais “amigável”, mas ainda precisamos conhecer o endereço dos registradores na memória:

```
// Inicializa o clock para GPIOF  
*((volatile uint32_t *)0x400FE108) = 0b100000;
```

Bit 5 do registrador RCGC2 controla o clock da porta GPIOF

RCGC2 tem 32 bits e fica no endereço (página 460 do datasheet):

- Base 0x400.F.E000
- Offset 0x108

Programação de Microcontroladores

4. C (ou CPP) + arquivo header do microcontrolador

Nessa abordagem, não precisamos conhecer os endereços dos registradores. O próprio fabricante do microcontrolador já fornece um arquivo header com todas as definições (HAL):

```
// Arquivo header do micro
#include "inc/tm4c123gh6pm.h"

// Inicializa o clock para GPIOF
SYSCTL_RCGC2_R = SYSCTL_RCGC2_GPIOF;
```

No header:

```
#define SYSCTL_RCGC2_R          (*((volatile uint32_t *)0x400FE108))
#define SYSCTL_RCGC2_GPIOF        0x00000020
```

Programação de Microcontroladores

5. C (ou CPP) + biblioteca de funções (SDK)

Nesse modo, as funções necessárias já nos são fornecidas. Precisamos apenas utilizá-las:

```
// Inicializa o clock para GPIOF  
SysCtlPeripheralEnable (SYSCTL_PERIPH_GPIOF);
```

Do manual da TivaWare:

26.2.2.32 SysCtlPeripheralEnable

Enables a peripheral.

Prototype:

```
void  
SysCtlPeripheralEnable(uint32_t ui32Peripheral)
```

Parameters:

ui32Peripheral is the peripheral to enable.

Description:

This function enables a peripheral. At power-up, all peripherals are disabled; they must be enabled in order to operate or respond to register reads/writes.

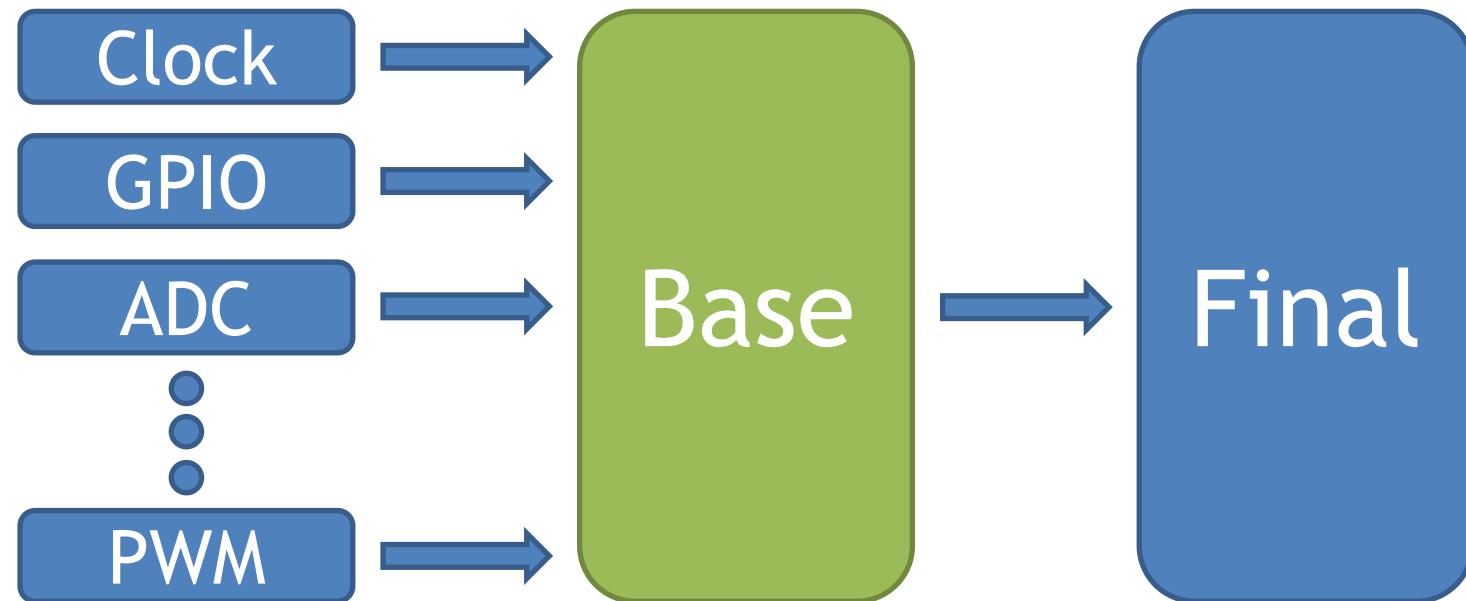
The `ui32Peripheral` parameter must be only one of the following values:

<code>SYSCTL_PERIPH_CAN0,</code>	<code>SYSCTL_PERIPH_ADC0,</code>	<code>SYSCTL_PERIPH_ADC1,</code>
<code>SYSCTL_PERIPH_CAN1,</code>	<code>SYSCTL_PERIPH_EEPROM0,</code>	<code>SYSCTL_PERIPH_CCM0,</code>
<code>SYSCTL_PERIPH_COMP0,</code>	<code>SYSCTL_PERIPH_EPI0,</code>	<code>SYSCTL_PERIPH_EMAC,</code>
<code>SYSCTL_PERIPH_EPHY,</code>	<code>SYSCTL_PERIPH_GPIOA,</code>	<code>SYSCTL_PERIPH_GPIOB,</code>
<code>SYSCTL_PERIPH_GPIOB,</code>	<code>SYSCTL_PERIPH_GPIOC,</code>	<code>SYSCTL_PERIPH_GPIOD,</code>
<code>SYSCTL_PERIPH_GPIOE,</code>	<code>SYSCTL_PERIPH_GPIOF,</code>	<code>SYSCTL_PERIPH_GPIOG,</code>
<code>SYSCTL_PERIPH_GPIOH,</code>	<code>SYSCTL_PERIPH_GPIOJ,</code>	<code>SYSCTL_PERIPH_GPIOK,</code>
<code>SYSCTL_PERIPH_GPIOI,</code>	<code>SYSCTL_PERIPH_GPIOM,</code>	<code>SYSCTL_PERIPH_GPION,</code>
<code>SYSCTL_PERIPH_GPIOJ,</code>	<code>SYSCTL_PERIPH_GPIOQ,</code>	<code>SYSCTL_PERIPH_GPIOR,</code>

Projeto Base

Objetivo:

- Criar um projeto que servirá de base para todos os demais projetos

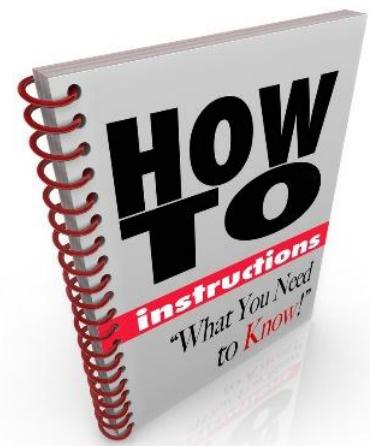


Nele, configuraremos a IDE, indicaremos onde o compilador deve buscar os arquivos necessários e como faremos a gravação e teste do código.

Projeto Base

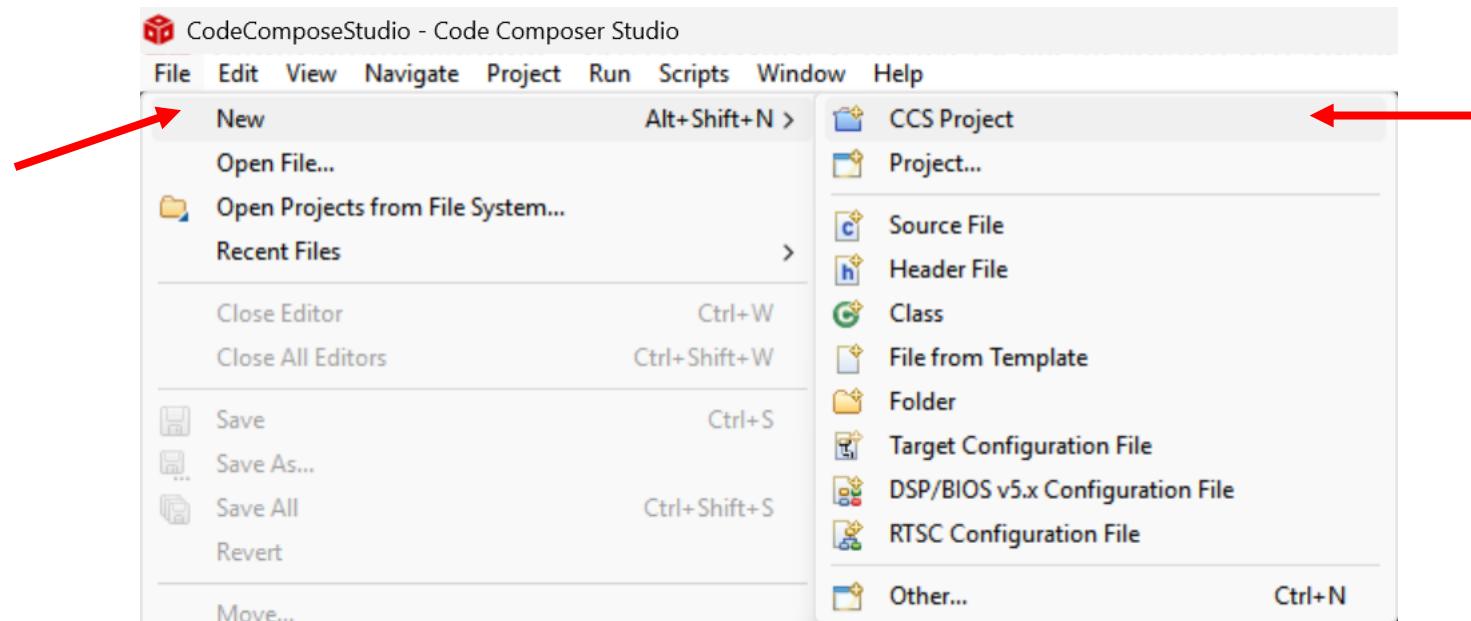
Passo a passo:

1. Criar um novo projeto
2. Adicionar as bibliotecas
3. Informar ao compilador onde buscar as bibliotecas no momento da compilação
4. Compilar o projeto

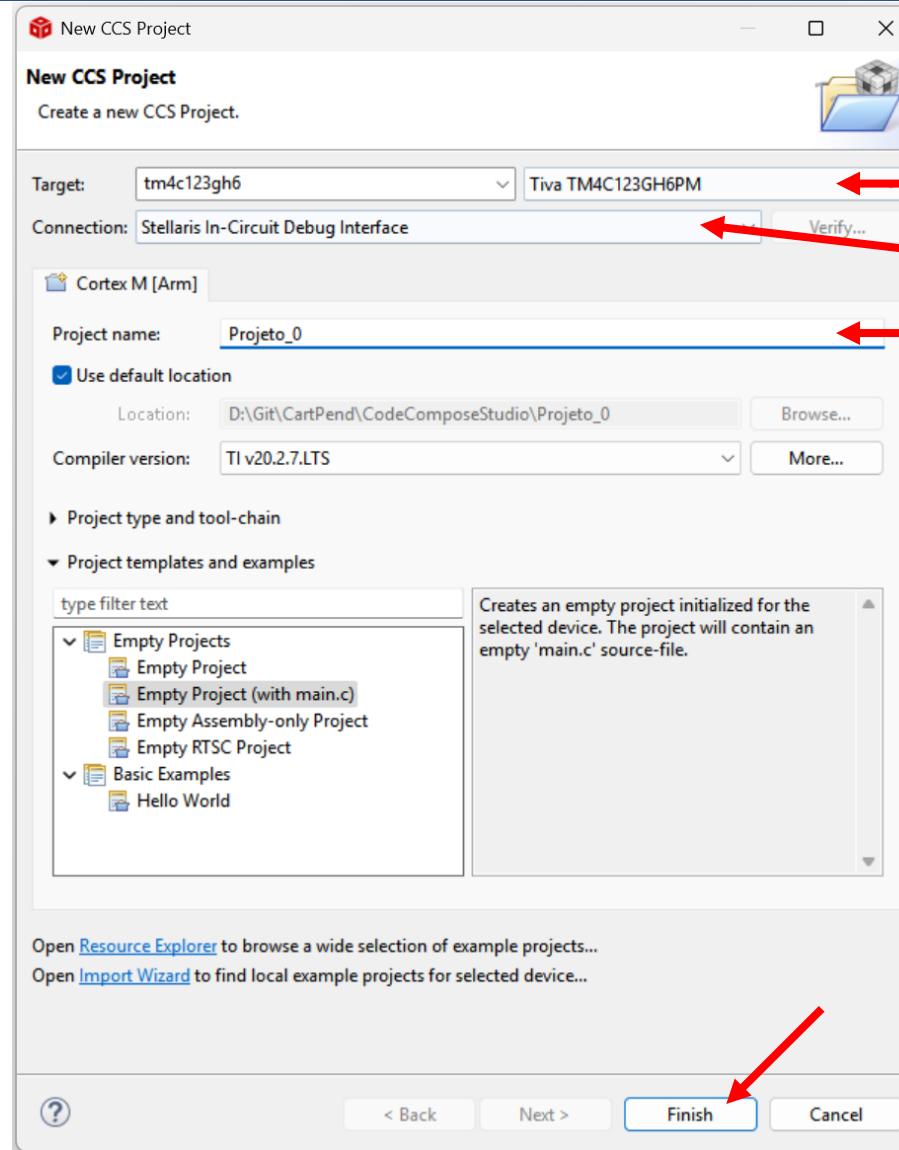


Projeto Base

1. Criar um novo projeto



Projeto Base

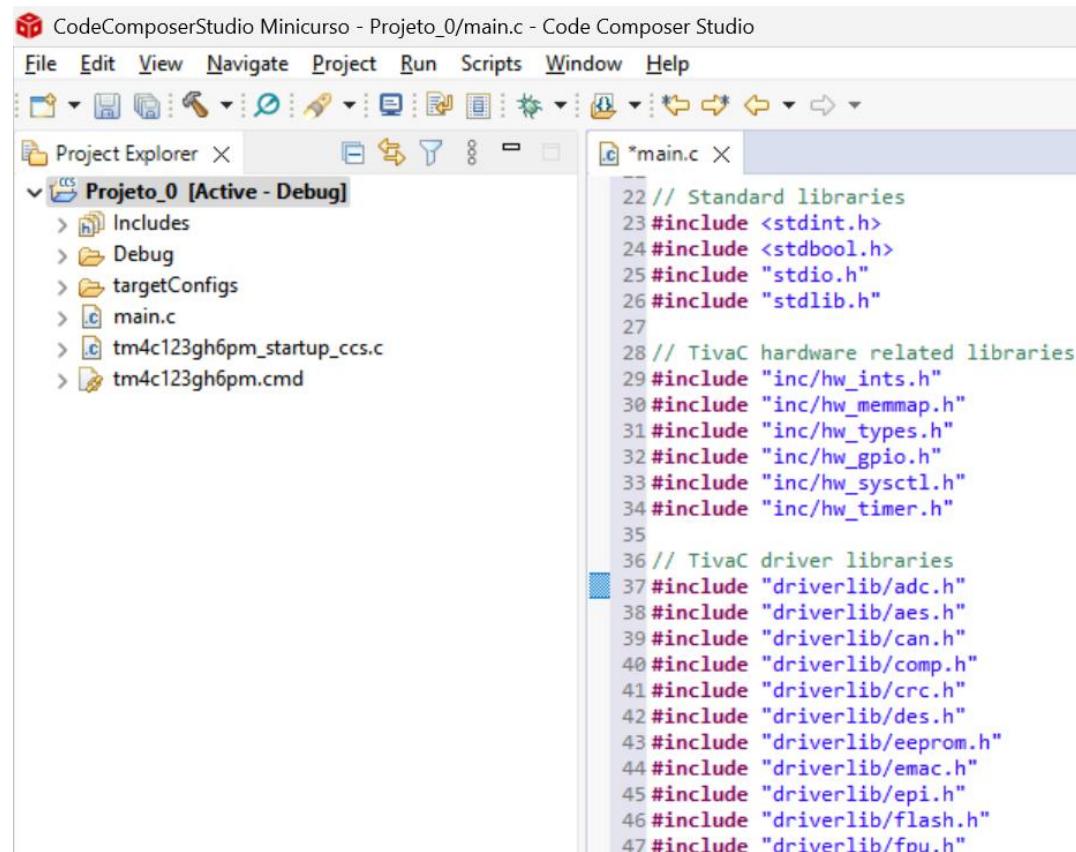


Microcontrolador usado
Forma de conexão
Nome do projeto

Projeto Base

2. Adicionar as bibliotecas

Substituir o conteúdo do arquivo “main.c” pelo do arquivo “main_0_base.c” do repositório do minicurso.



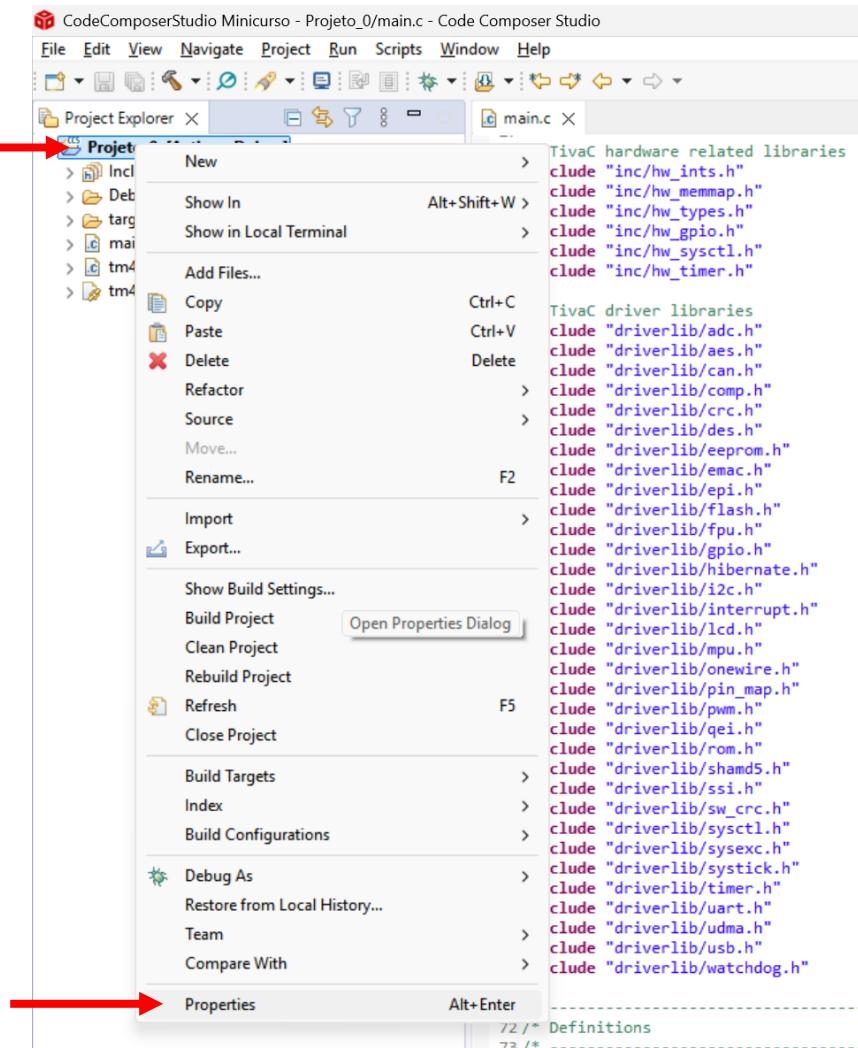
The screenshot shows the Code Composer Studio interface with the following details:

- Project Explorer:** Shows the project structure for "Projeto_0 [Active - Debug]". It includes:
 - Includes
 - Debug
 - targetConfigs
 - main.c (selected)
 - tm4c123gh6pm_startup_ccs.c
 - tm4c123gh6pm.cmd
- Code Editor:** Displays the content of the selected file, "main.c". The code lists numerous header file includes, categorized by library type:
 - // Standard libraries
 - #include <stdint.h>
 - #include <stdbool.h>
 - #include "stdio.h"
 - #include "stdlib.h"
 - // TivaC hardware related libraries
 - #include "inc/hw_ints.h"
 - #include "inc/hw_memmap.h"
 - #include "inc/hw_types.h"
 - #include "inc/hw_gpio.h"
 - #include "inc/hw_sysctl.h"
 - #include "inc/hw_timer.h"
 - // TivaC driver libraries
 - #include "driverlib/adc.h"
 - #include "driverlib/aes.h"
 - #include "driverlib/can.h"
 - #include "driverlib/comp.h"
 - #include "driverlib/crc.h"
 - #include "driverlib/des.h"
 - #include "driverlib/eeprom.h"
 - #include "driverlib/emac.h"
 - #include "driverlib/epi.h"
 - #include "driverlib/flash.h"
 - #include "driverlib/fpu.h"

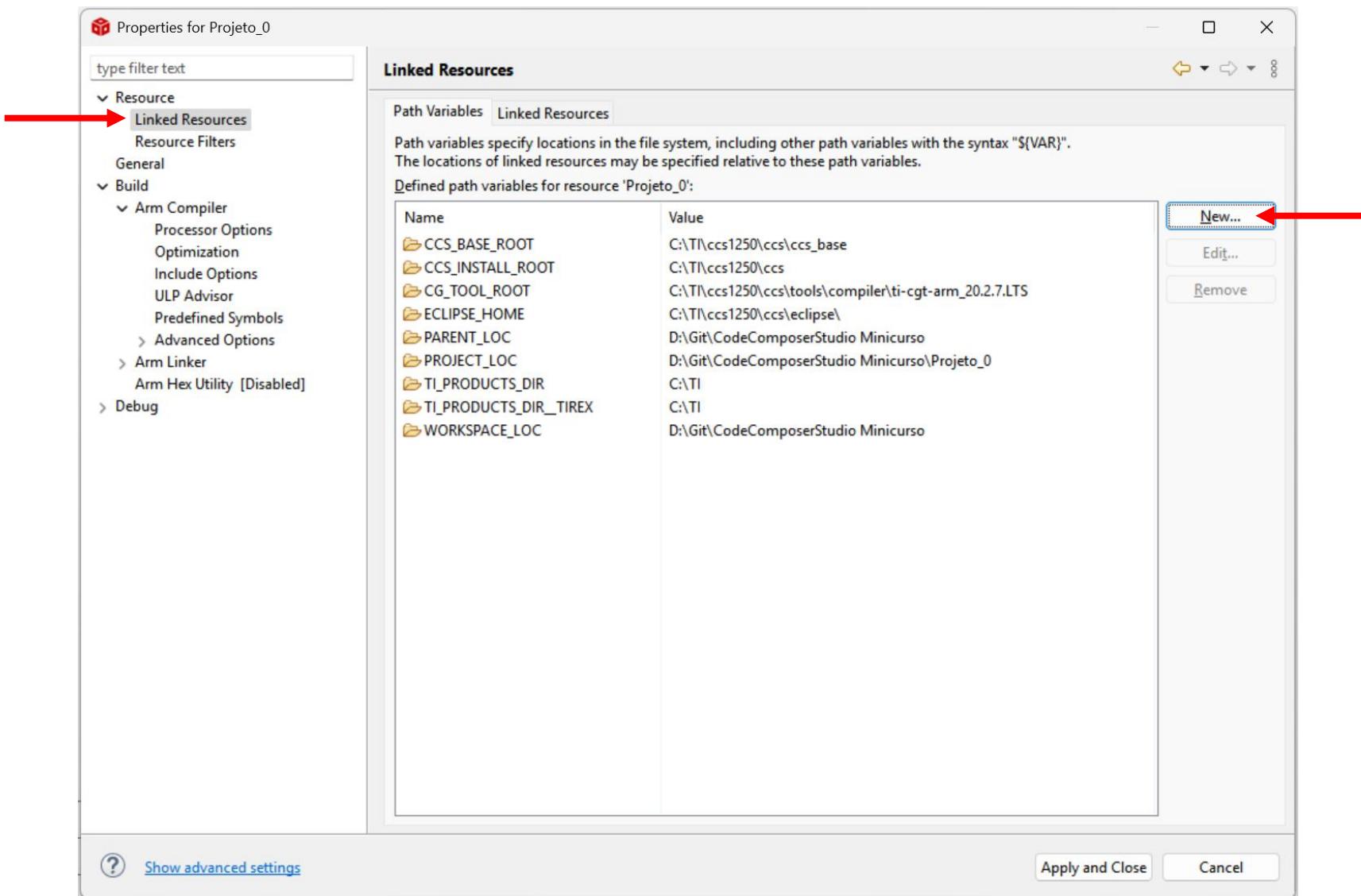
Projeto Base

3. Informar ao compilador onde buscar as bibliotecas no momento da compilação

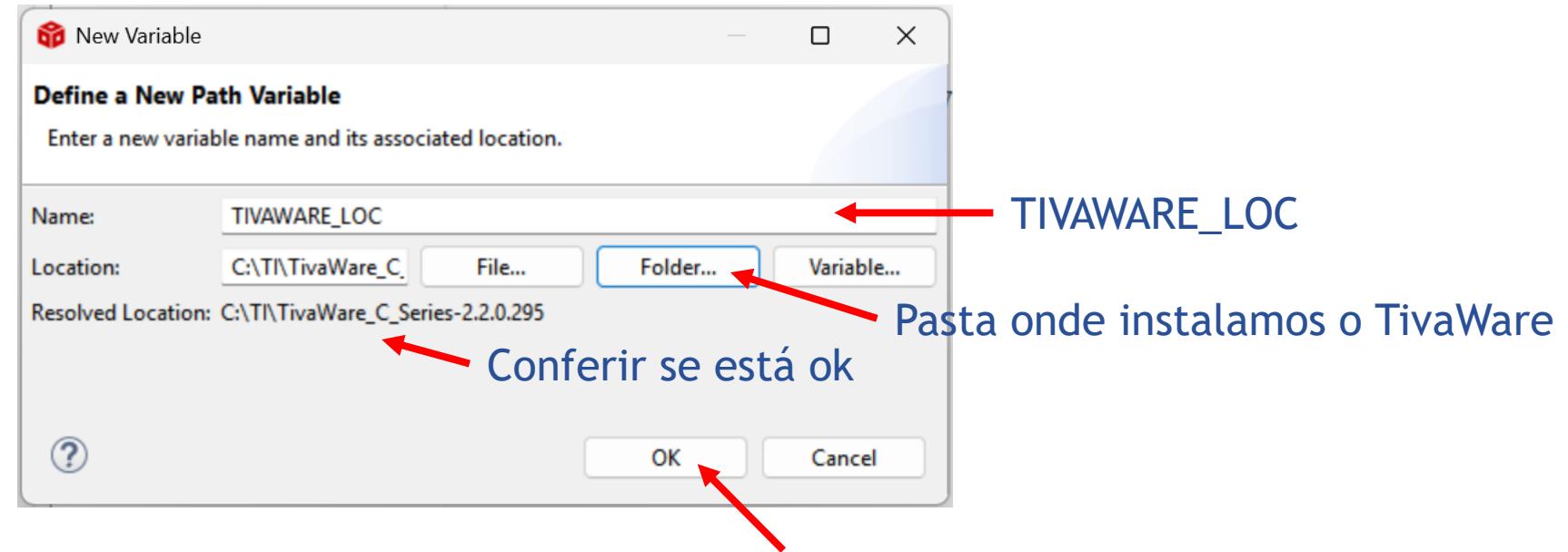
Botão direito



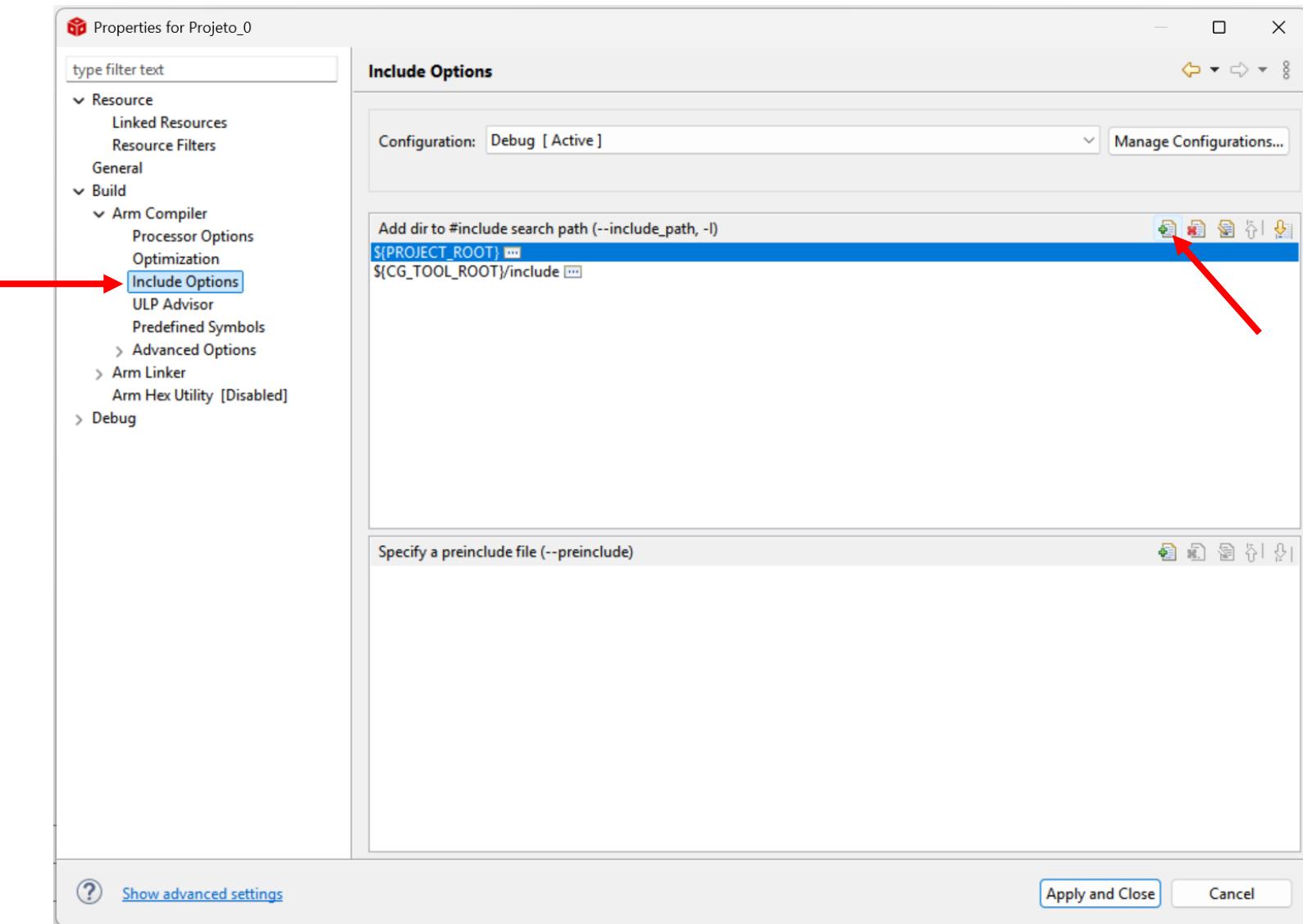
Projeto Base



Projeto Base

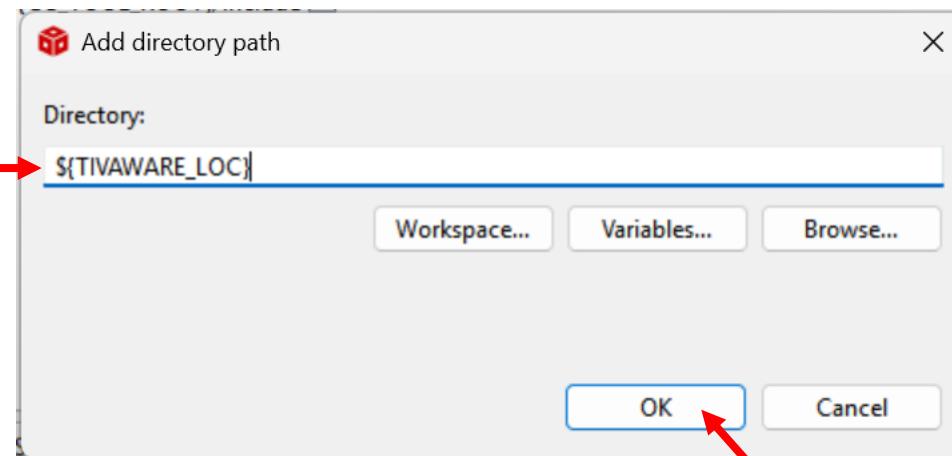


Projeto Base

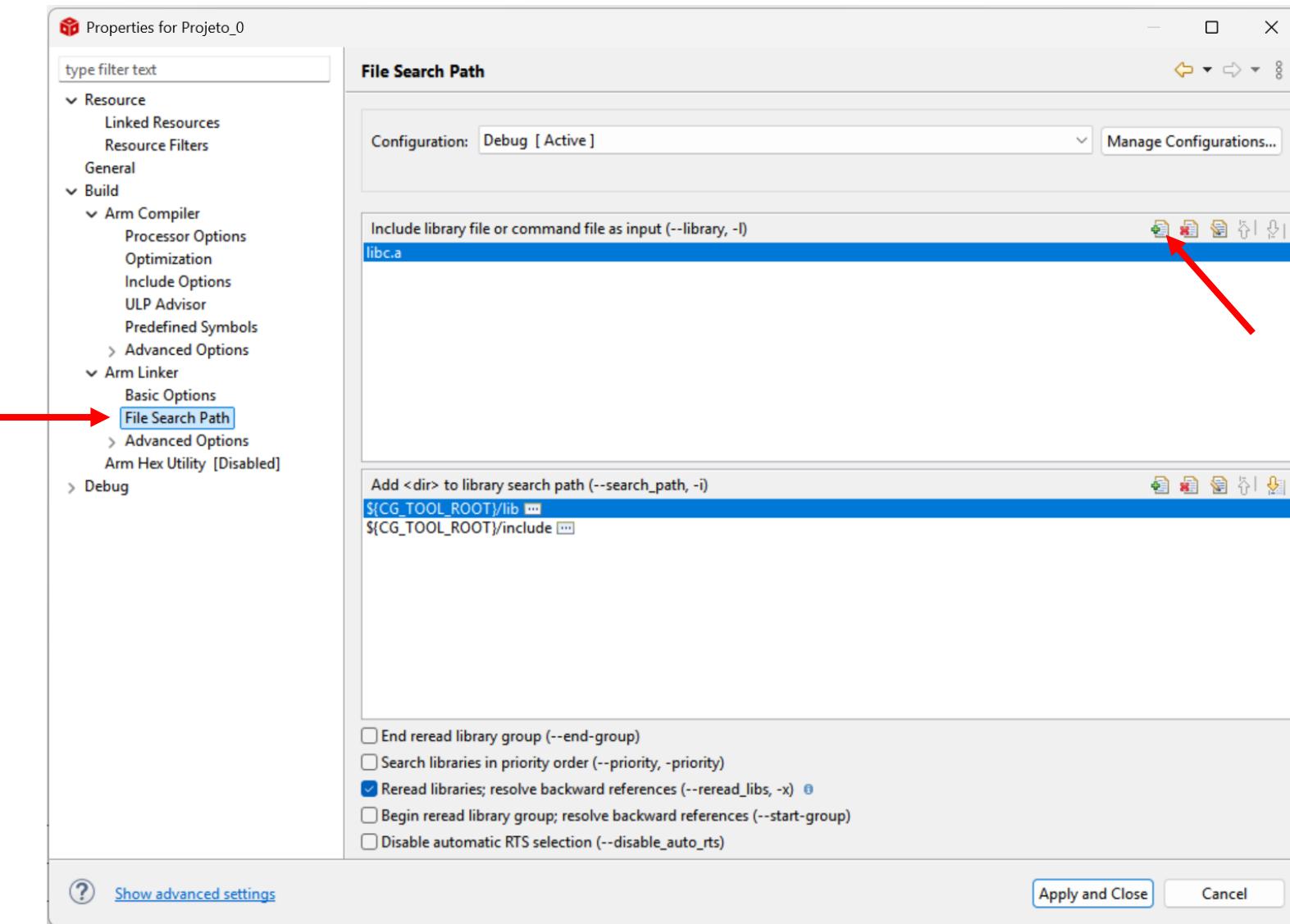


Projeto Base

`${TIVWARE_LOC}`

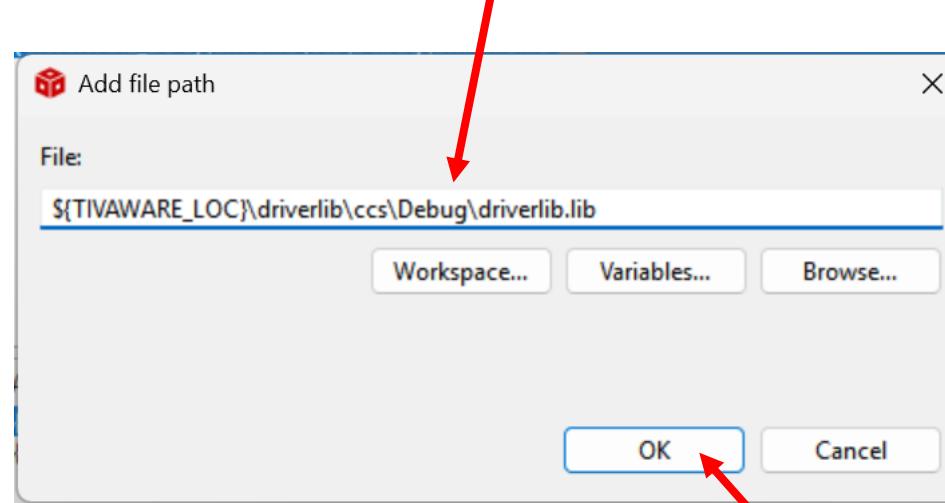


Projeto Base

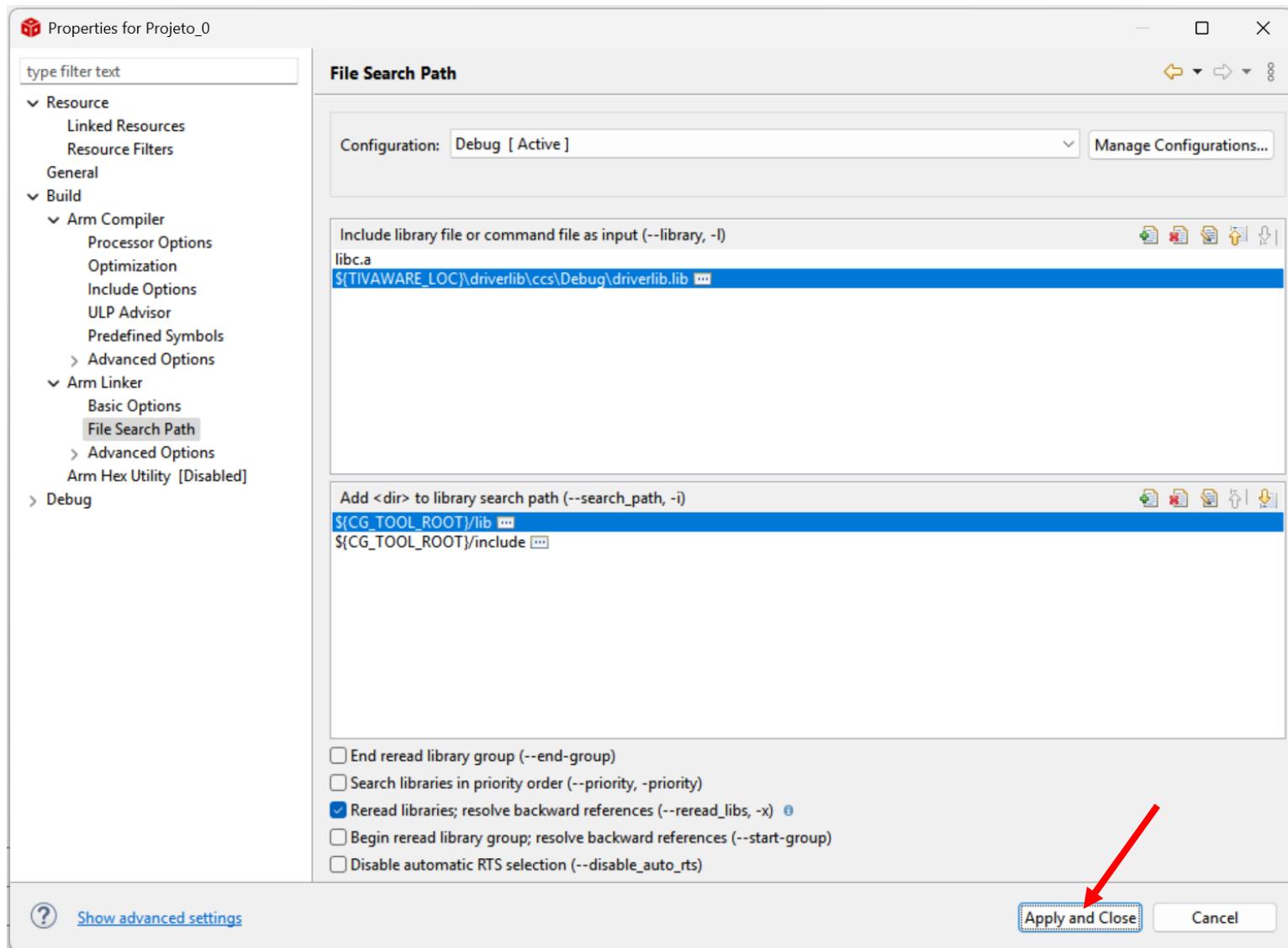


Projeto Base

`${TIVWARE_LOC}\driverlib\ccs\Debug\driverlib.lib`



Projeto Base



Projeto Base

4. Compilar o projeto

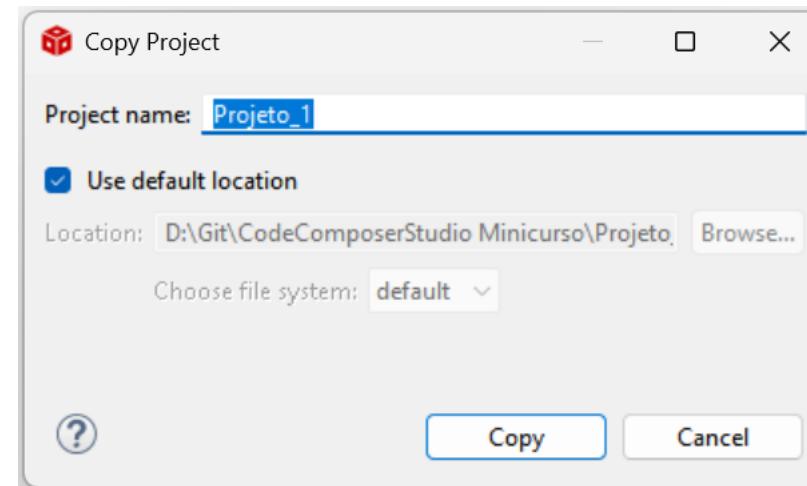
Se todas as bibliotecas forem encontradas pelo compiladores, teremos 0 erros na compilação



Projeto Base

Tudo pronto!

Para os próximos exercícios, precisamos apenas copiar e colar o projeto base dando a ele um novo nome. Todas as configurações já estarão feitas.





renan.duarte@gedre.ufsm.br

Av. Roraima 1000 - Prédio 10 - Sala 440 - Santa Maria - RS



www.ufsm.br/gedre



+55 55 3220 9492



gedre.ufsm

Aula 4 - Sistema de clock

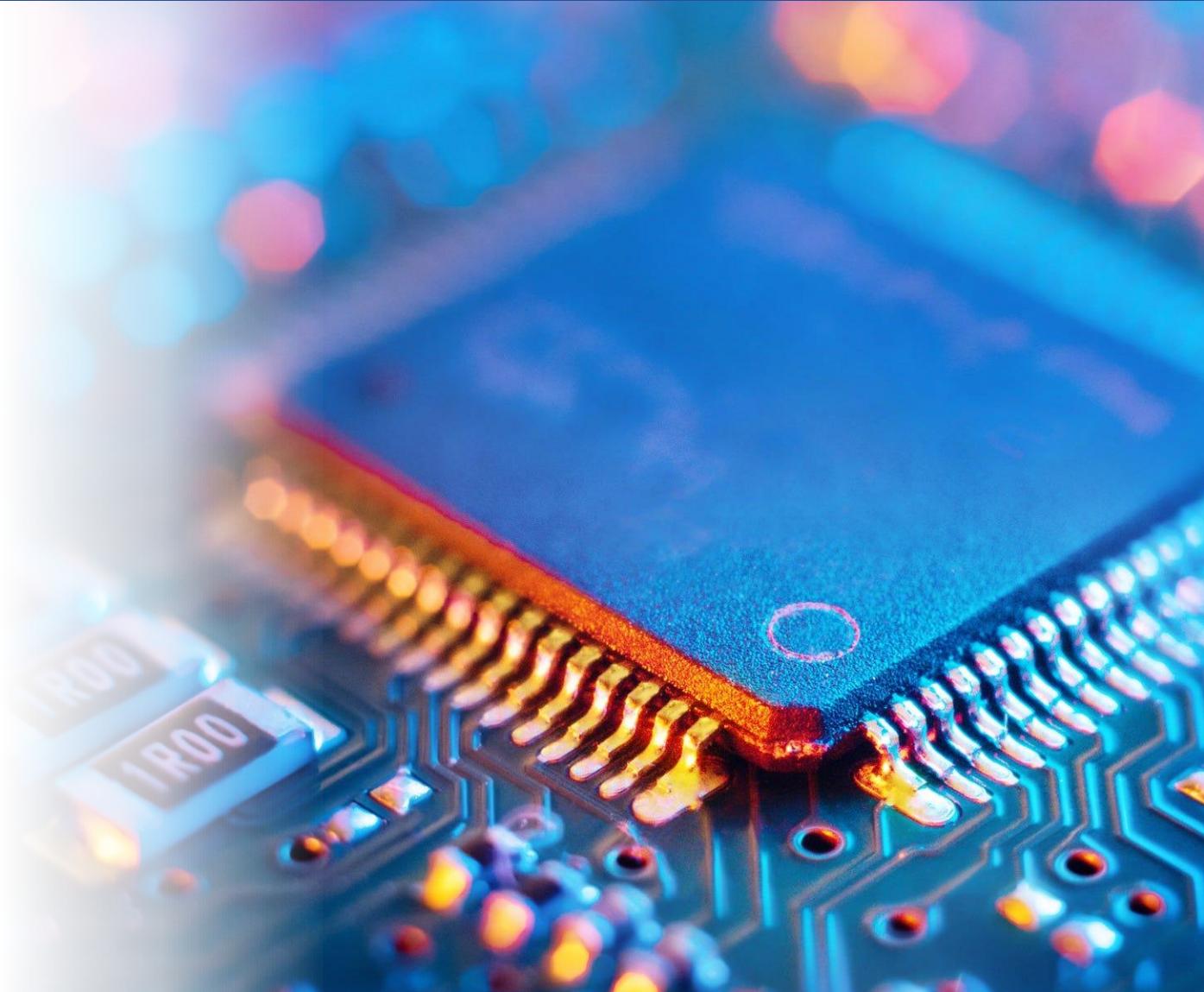
Prof. Renan Duarte

Minicurso de microcontroladores

Março de 2025

Conteúdo

- Sistema de clock do TM4C123GH6
- Projeto 1 - Sistema de clock



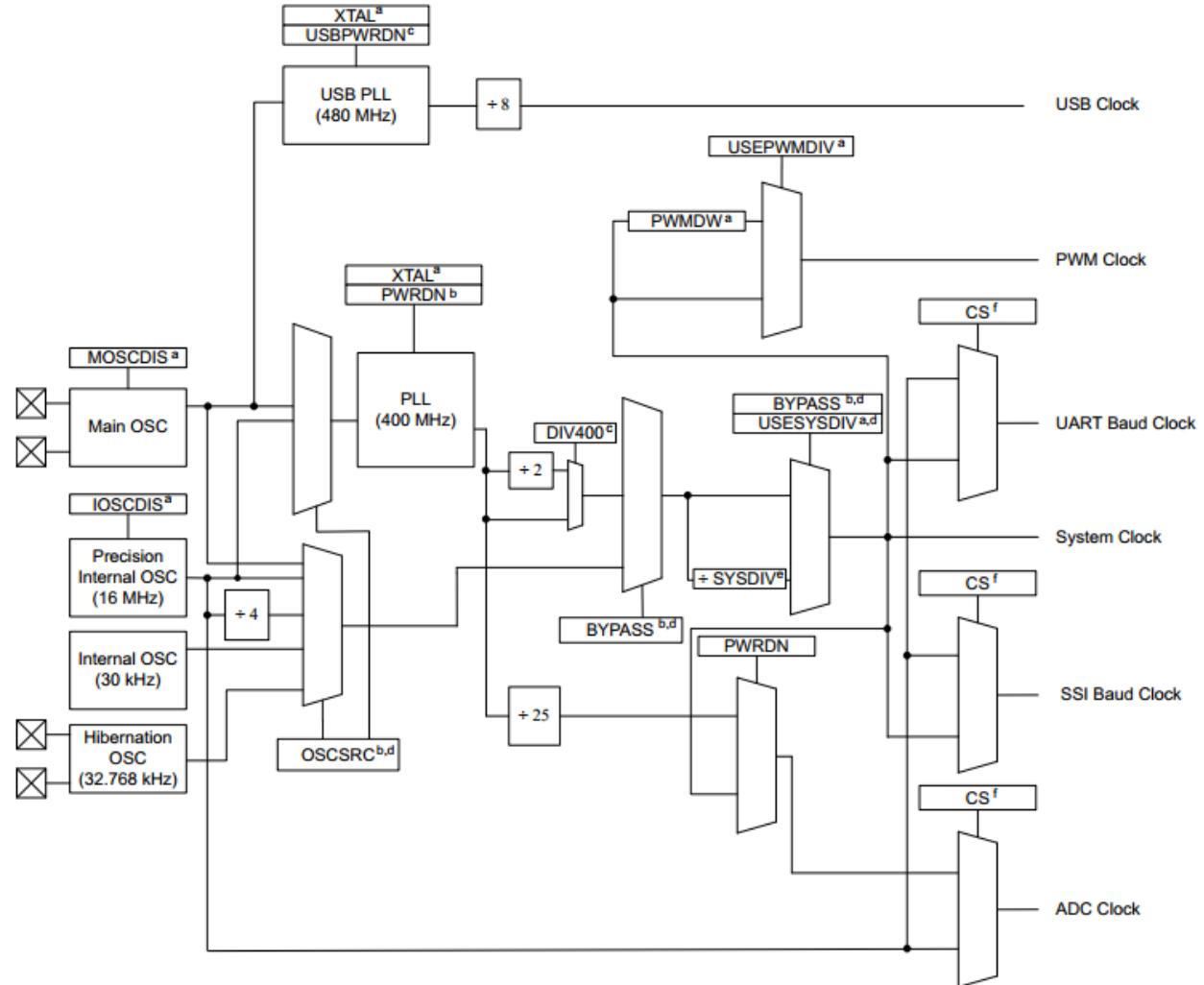
Sistema de clock do TM4C123GH6

Fontes de clock fundamentais

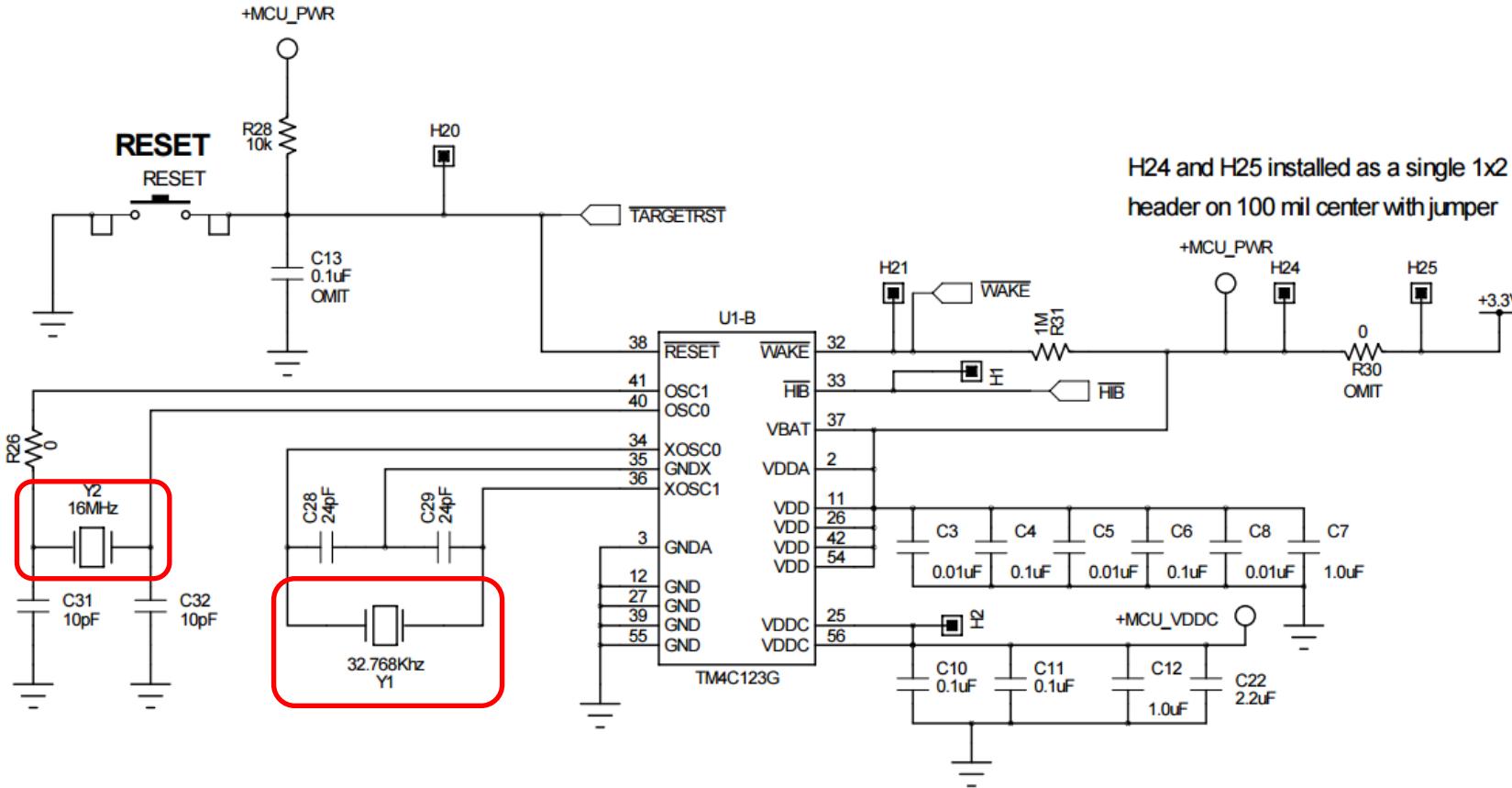
- Oscilador Interno de Precisão (PIOSC)
 - 16 MHz $\pm 1\%$ na temperatura ambiente
 - Usado após o POR
- Oscilador Principal (MOSC):
 - Fonte de externa ou cristal (5-25 MHz)
 - Fornece clock para o USB PLL
- Oscilador Interno de BF (LFIOSC):
 - Usado no modo deep-sleep para economia de energia
 - Reduz a comutação interna, permitindo desligar o MOSC
- Clock de Hibernação:
 - Oscilador de 32,768 kHz para RTC

Clock interno do sistema (SysClk)

- Derivado de qualquer fonte de clock



Sistema de clock do TM4C123GH6



Projeto 1 - Sistema de clock

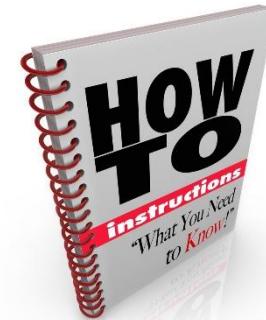
Objetivo:

- Configurar e medir o clock da CPU



Passo a passo:

- Definir a origem do clock (Cristal externo)
- Ligar o PLL (200 MHz)
- Configurar o divisor de clock





renan.duarte@gedre.ufsm.br

Av. Roraima 1000 - Prédio 10 - Sala 440 - Santa Maria - RS



www.ufsm.br/gedre



+55 55 3220 9492



gedre.ufsm

Aula 5 - GPIOs

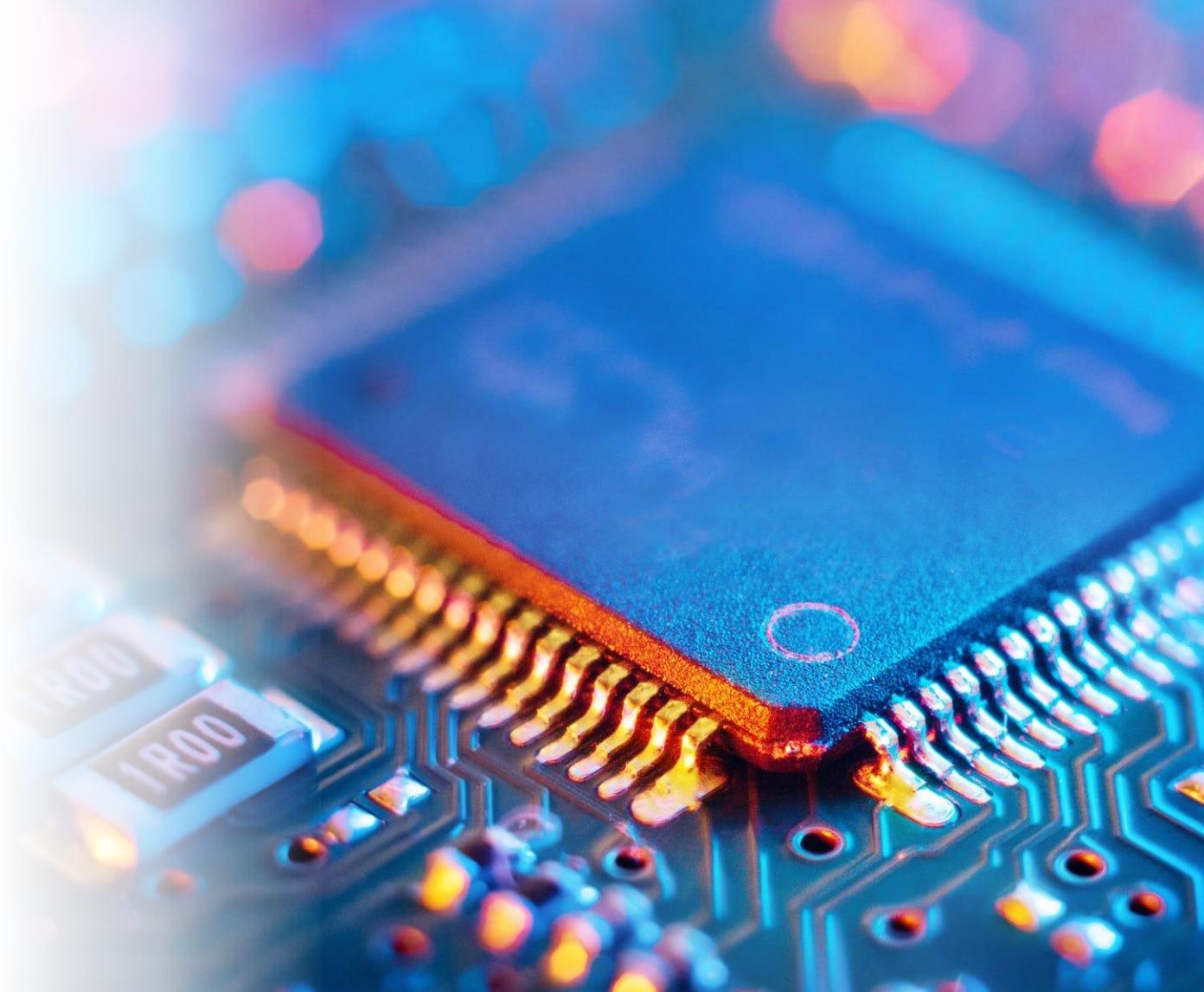
Prof. Renan Duarte

Minicurso de microcontroladores

Março de 2025

Conteúdo

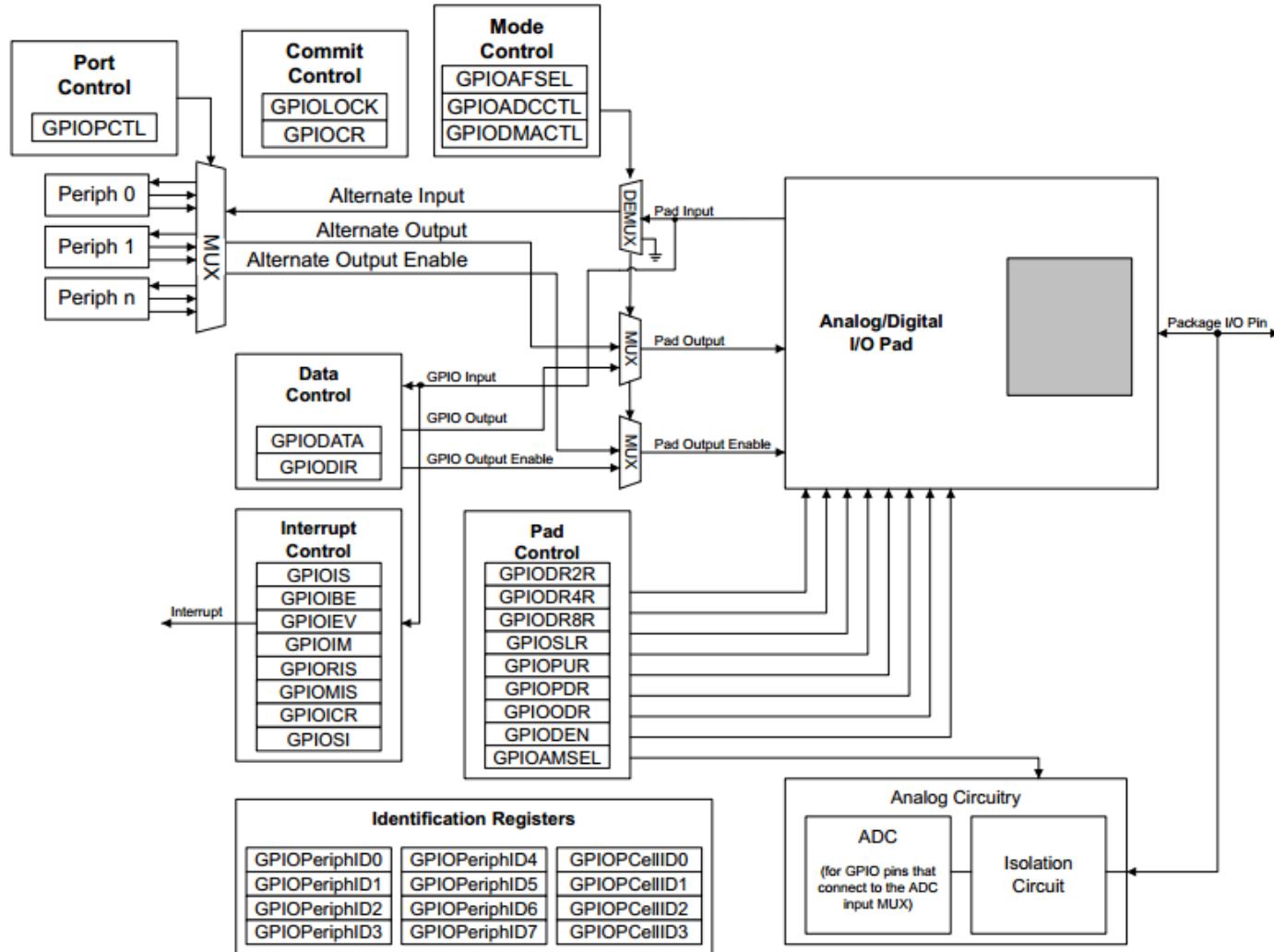
- GPIOs do TM4C123GH6
- Projeto 2 - GPIO



GPIOs do TM4C123GH6

Características

- Entradas e saídas 3.3V (maioria dos pinos é tolertante à 5V)
- Interrupções em todos os portas
- Todos os pinos podem ser configurados com:
 - Schmitt Trigger
 - Pull Up ou Pull Down
 - Coletor Aberto
 - Controle de corrente (2, 8 ou 18mA)



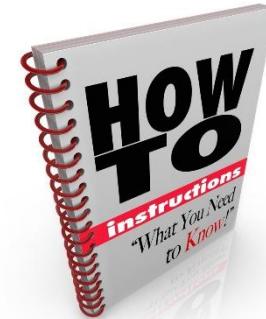
Projeto 2 - GPIO

Objetivo:

- Ler o estado de pinos de entrada
- Definir o estado de pinos de saída

Passo a passo:

- Habilitar periférico
- Configurar portas como IO
- Ler entradas e definir saídas





renan.duarte@gedre.ufsm.br

Av. Roraima 1000 - Prédio 10 - Sala 440 - Santa Maria - RS



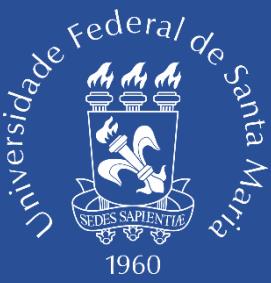
www.ufsm.br/gedre



+55 55 3220 9492



gedre.ufsm



UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
GEDRE - INTELIGÊNCIA EM ILUMINAÇÃO

gedre
inteligência
em iluminação

Aula 6 - Interrupções

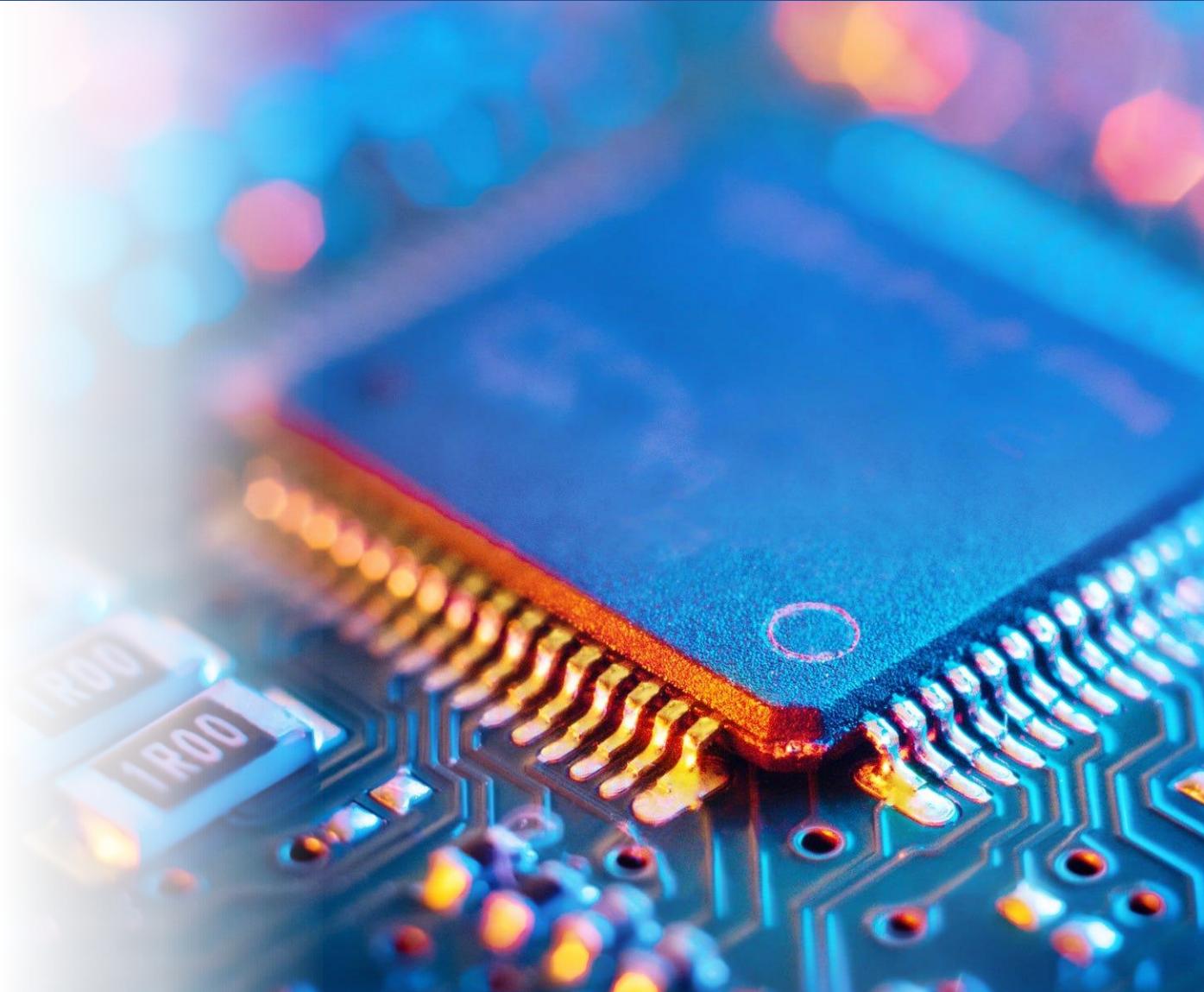
Prof. Renan Duarte

Minicurso de microcontroladores

Março de 2025

Conteúdo

- Introdução
- Interrupções no TM4C123GH6
- Projeto 3 - Interrupções



Interrupções

O que são?

- Eventos que interrompem a execução normal do programa para atender uma situação urgente
- Permitem resposta rápida a eventos externos ou internos

Como funcionam?

- Quando ocorre uma interrupção, o microcontrolador pausa o que estava fazendo
- Executa uma rotina (função) especial chamada ISR (Interrupt Service Routine)
- Após o atendimento, volta para onde parou

Interrupções no TM4C123GH6

Controle de Interrupções

- Usa o NVIC (Nested Vectored Interrupt Controller) integrado ao ARM Cortex-M4F

Otimizações

- Tail-chaining: permite atendimento de interrupções consecutivas sem salvar/restaurar estado entre elas
- Processamento de interrupção rápido e determinístico
 - 12 ciclos normais, 6 ciclos com tail-chaining (sem FPU)

Recursos Adicionais

- 8 níveis de prioridade para 7 exceções (handlers do sistema) e 78 interrupções
- Repriorização dinâmica de interrupções via software
- Interrupção externa não-mascarável (NMI) para aplicações críticas
- Manipulação automática de registradores por hardware (eficiência no atendimento)

Projeto 3 - Interrupções

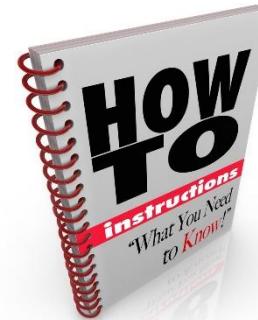
Objetivo:

- Utilizar interrupções para monitorar entradas
- Definir o estado de pinos de saída



Passo a passo:

- Habilitar periférico
- Configurar portas como IO
- Configurar interrupção
- Entrar no laço infinito
- Atuar somente quando houver interrupção





renan.duarte@gedre.ufsm.br

Av. Roraima 1000 - Prédio 10 - Sala 440 - Santa Maria - RS



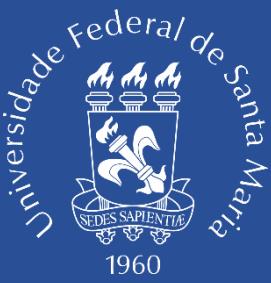
www.ufsm.br/gedre



+55 55 3220 9492



gedre.ufsm



UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
GEDRE - INTELIGÊNCIA EM ILUMINAÇÃO

gedre
inteligência
em iluminação

Aula 7 - Timer

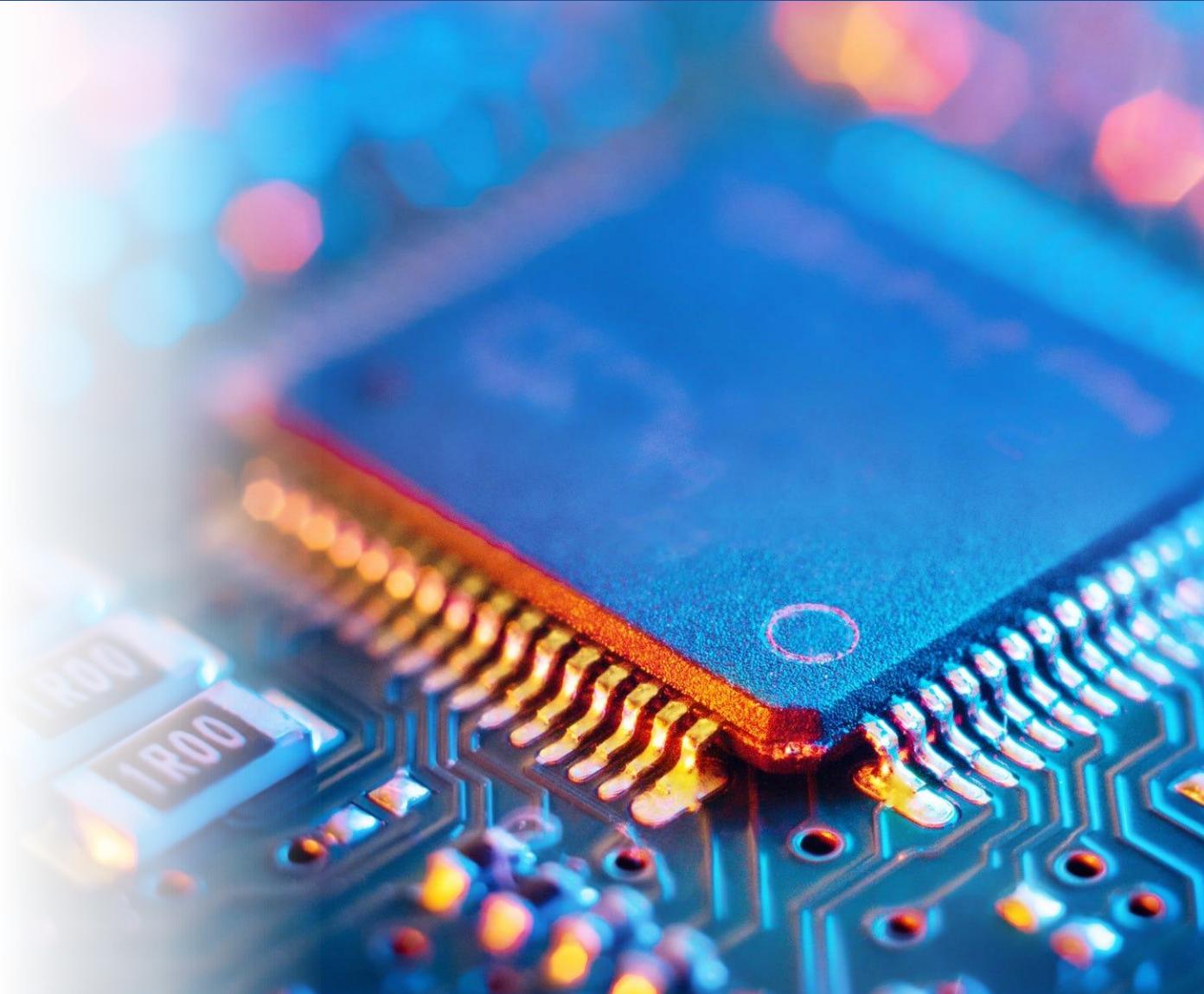
Prof. Renan Duarte

Minicurso de microcontroladores

Março de 2025

Conteúdo

- Introdução
- Timers do TM4C123GH6
- Projeto 4 - Timer



Timers

São contadores de tempo dentro do microcontrolador

Para que servem?

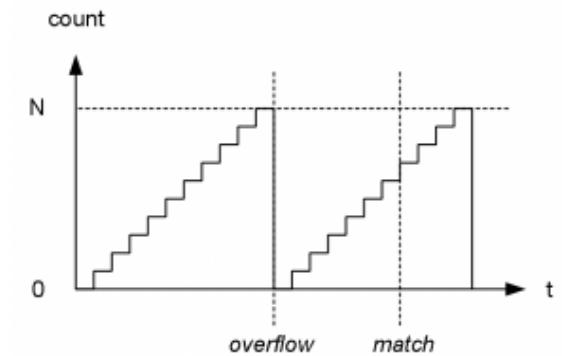
- Medir intervalos de tempo
- Gerar atrasos (delays)
- Controlar eventos periódicos
- Contar pulsos ou eventos externos

Como funcionam?

- Conta "ticks" do clock até atingir um valor programado
- Conta "ticks" do clock entre enventos

Modos comuns

- Contador (mede eventos)
- Temporizador (mede tempo)
- PWM (gera sinais de controle)



Timers do TM4C123GH6

Timers de propósito geral

- 6 módulos de 32 bits
- 6 módulos de 64 bits

Tipos de operação

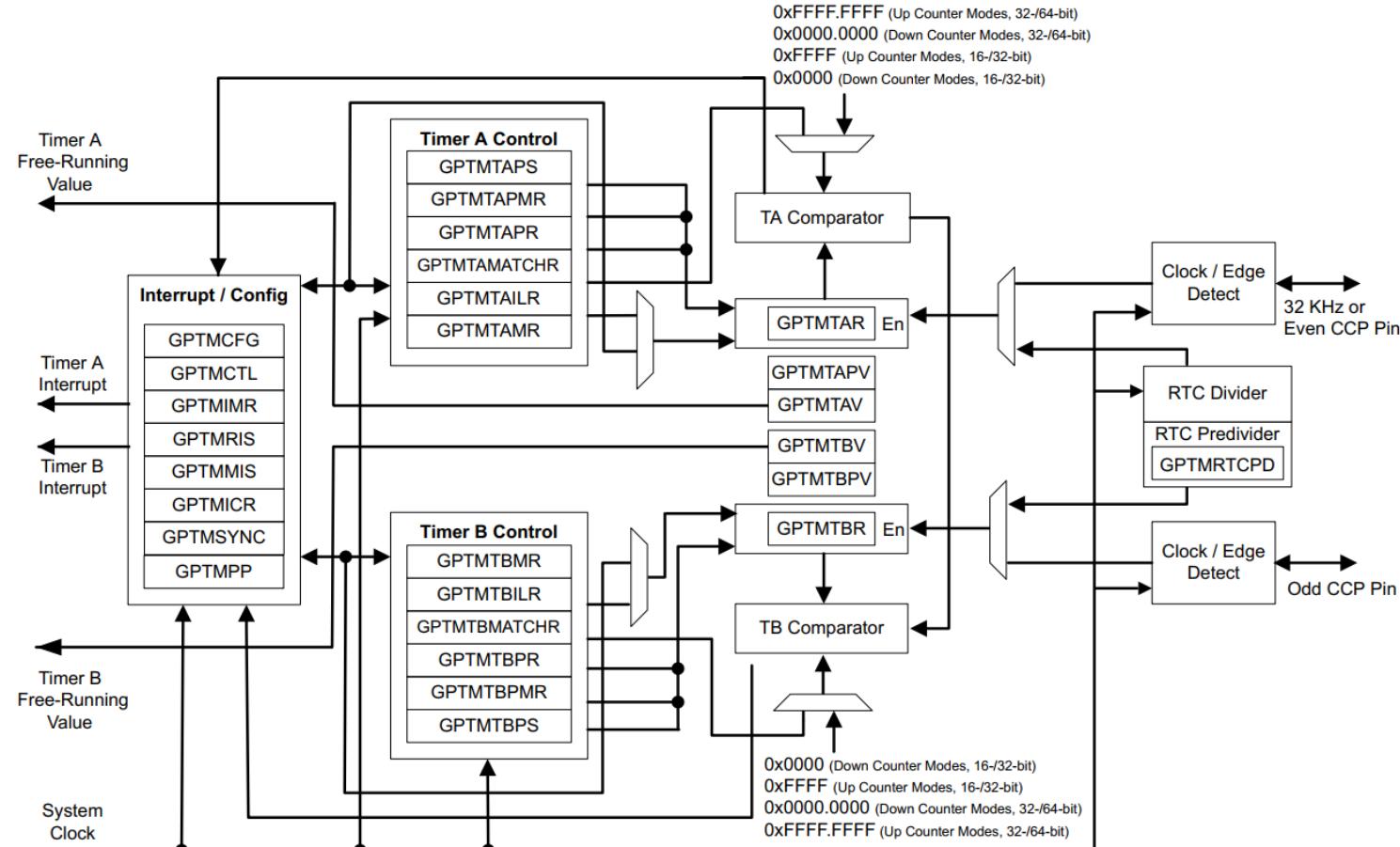
- Temporizador
- Contador de eventos

Modos de funcionamento

- Período único (One-shot)
- Periódico (Periodic)
- Captura de eventos (Capture)
- PWM (Pulse Width Modulation)

Características principais

- Operação independente ou combinada (16, 32 ou 64 bits)
- Geração de interrupções



Timers do TM4C123GH6

Timer	Up/Down Counter	CCP0 Pin	CCP1 Pin
16/32-Bit Timer 0	Timer A	T0CCP0	-
	Timer B	-	T0CCP1
16/32-Bit Timer 1	Timer A	T1CCP0	-
	Timer B	-	T1CCP1
16/32-Bit Timer 2	Timer A	T2CCP0	-
	Timer B	-	T2CCP1
16/32-Bit Timer 3	Timer A	T3CCP0	-
	Timer B	-	T3CCP1
16/32-Bit Timer 4	Timer A	T4CCP0	-
	Timer B	-	T4CCP1
16/32-Bit Timer 5	Timer A	T5CCP0	-
	Timer B	-	T5CCP1
32/64-Bit Wide Timer 0	Timer A	WT0CCP0	-
	Timer B	-	WT0CCP1
32/64-Bit Wide Timer 1	Timer A	WT1CCP0	-
	Timer B	-	WT1CCP1
32/64-Bit Wide Timer 2	Timer A	WT2CCP0	-
	Timer B	-	WT2CCP1
32/64-Bit Wide Timer 3	Timer A	WT3CCP0	-
	Timer B	-	WT3CCP1
32/64-Bit Wide Timer 4	Timer A	WT4CCP0	-
	Timer B	-	WT4CCP1
32/64-Bit Wide Timer 5	Timer A	WT5CCP0	-
	Timer B	-	WT5CCP1

Projeto 4 - Timer

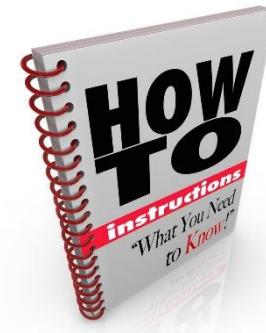
Objetivo:

- Gerar uma interrupção de timer a cada 1 segundo
- Alterar estado das saídas a cada interrupção
- Refazer utilizando o SysTick timer



Passo a passo:

- Habilitar periférico
- Configurar saídas digitais
- Configurar timer
- Configurar ação da interrupção
- Ligar timer
- Alterar estado das saídas a cada interrupção





renan.duarte@gedre.ufsm.br

Av. Roraima 1000 - Prédio 10 - Sala 440 - Santa Maria - RS



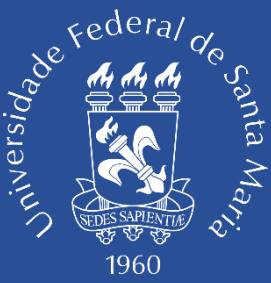
www.ufsm.br/gedre



+55 55 3220 9492



gedre.ufsm



UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
GEDRE - INTELIGÊNCIA EM ILUMINAÇÃO

gedre
inteligência
em iluminação

Aula 8 - ADC

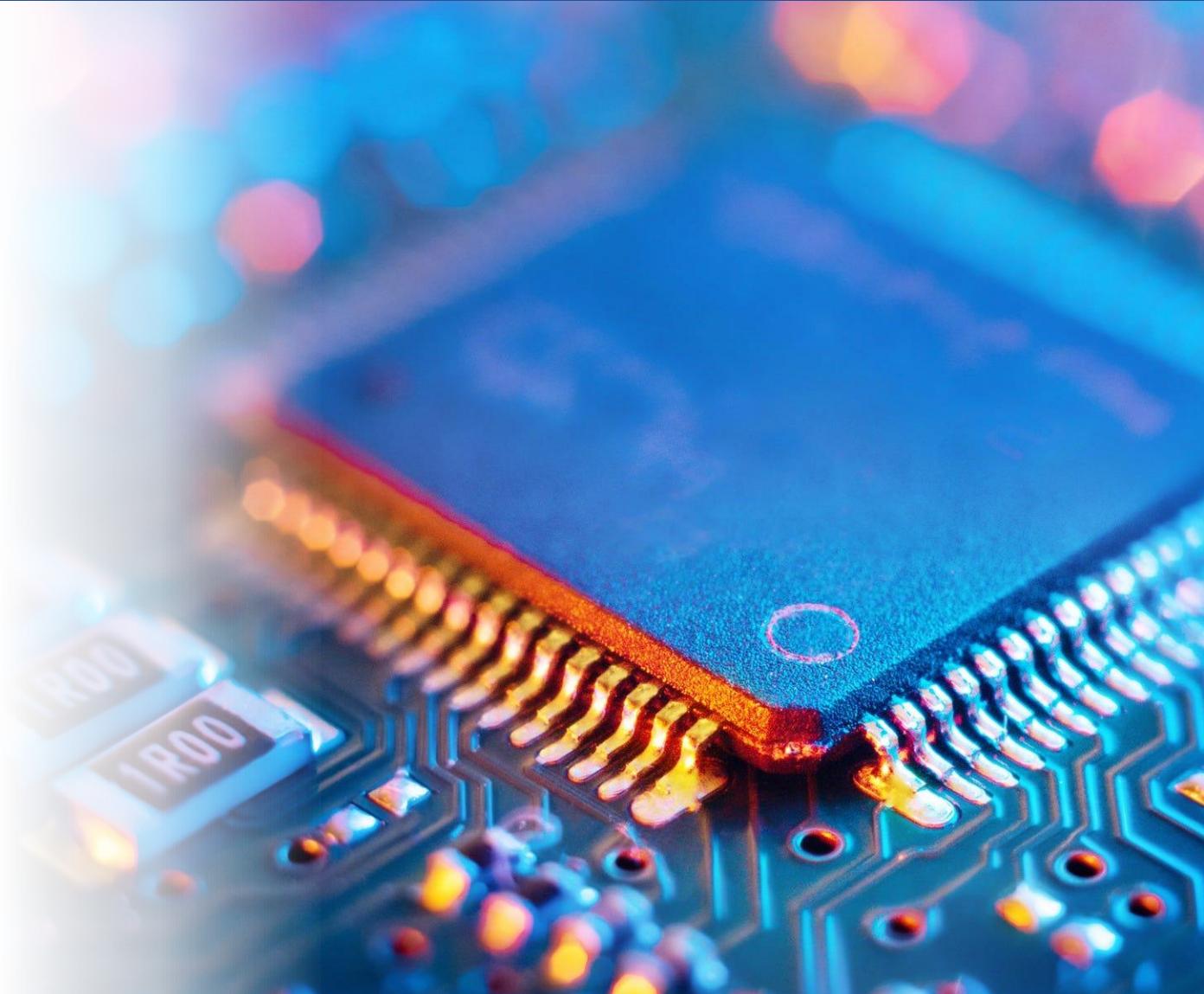
Prof. Renan Duarte

Minicurso de microcontroladores

Março de 2025

Conteúdo

- Introdução
- ADC no TM4C123GH6
- Projeto 5 - ADC



Conversão analógico-digital

O que é?

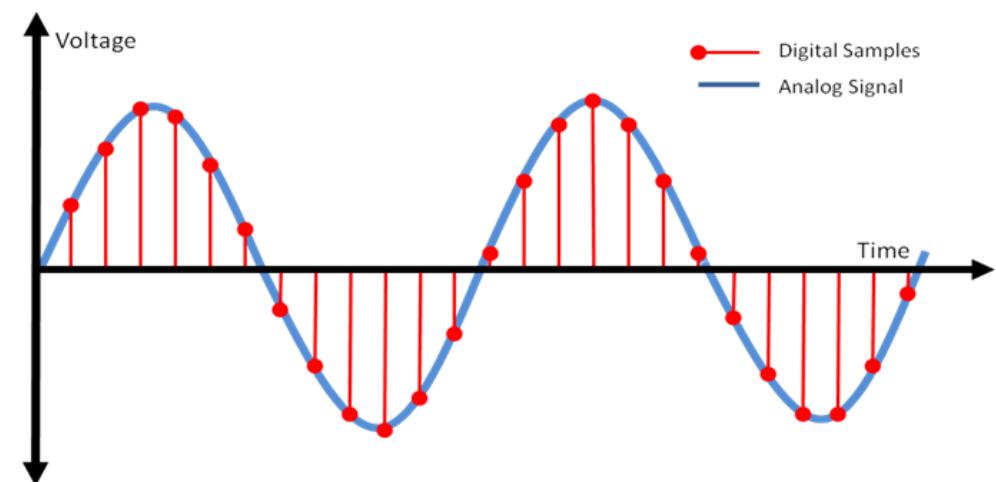
- Transforma sinais do mundo real (análogicos) em números digitais que o microcontrolador entende

Exemplo

- Temperatura, luz, som → Viram valores digitais

Como funciona

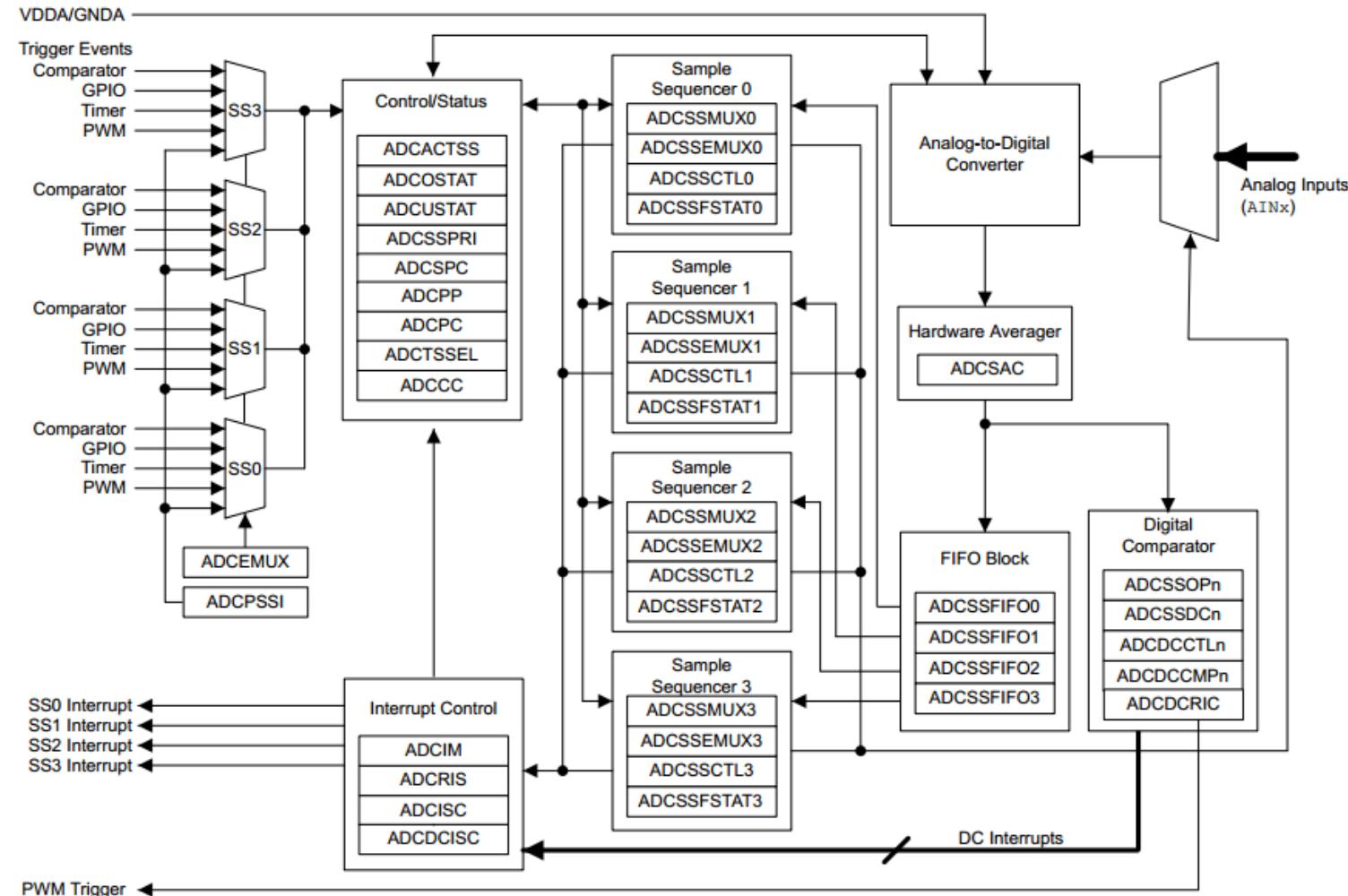
- O ADC lê a tensão de entrada
- Converte essa tensão em um número (por exemplo, de 0 a 4095 em um ADC de 12 bits)



ADC no TM4C123GH6

Características

- 2 módulos ADC
- 12 entradas analógicas
- 12 bits de resolução (valores de 0 a 4095)
- Até 1 milhão de amostras por segundo (1 MSPS)
- Pode-se configurar sequências de canais.
- Pode começar via software, timers ou eventos externos.
- Média em hardware



ADC no TM4C123GH6

Entradas analógicas

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type ^a	Description
AIN0	6	PE3	I	Analog	Analog-to-digital converter input 0.
AIN1	7	PE2	I	Analog	Analog-to-digital converter input 1.
AIN2	8	PE1	I	Analog	Analog-to-digital converter input 2.
AIN3	9	PE0	I	Analog	Analog-to-digital converter input 3.
AIN4	64	PD3	I	Analog	Analog-to-digital converter input 4.
AIN5	63	PD2	I	Analog	Analog-to-digital converter input 5.
AIN6	62	PD1	I	Analog	Analog-to-digital converter input 6.
AIN7	61	PD0	I	Analog	Analog-to-digital converter input 7.
AIN8	60	PE5	I	Analog	Analog-to-digital converter input 8.
AIN9	59	PE4	I	Analog	Analog-to-digital converter input 9.
AIN10	58	PB4	I	Analog	Analog-to-digital converter input 10.
AIN11	57	PB5	I	Analog	Analog-to-digital converter input 11.

Sequenciadores

Sequencer	Number of Samples	Depth of FIFO
SS3	1	1
SS2	4	4
SS1	4	4
SS0	8	8

ADC no TM4C123GH6

Média em hardware

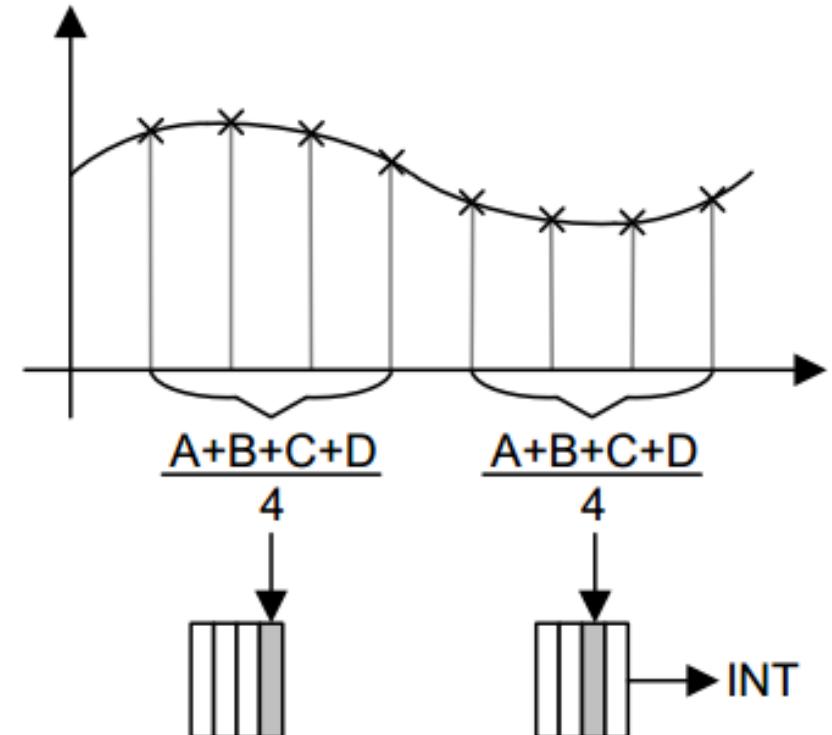
- O ADC pode fazer a média de várias amostras automaticamente, sem precisar de código
- Reduz ruídos e flutuações no sinal analógico
- Mais precisão sem gastar processamento da CPU

Como funciona?

- O ADC tira várias amostras do mesmo sinal
- Calcula a média e entrega um valor mais estável

Configurações possíveis

- Média de 4, 8, 16 ou 32 amostras



Projeto 5 - ADC

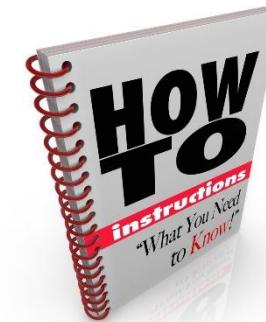
Objetivo:

- Converter um valor analógico de uma entrada
- Calcular a tensão correspondente
- Repetir utilizando duas entradas analógicas



Passo a passo:

- Habilitar periférico
- Configurar entrada analógica
- Configurar conversor AD
- Efetuar conversão
- Calcular o valor de tensão correspondente





renan.duarte@gedre.ufsm.br

Av. Roraima 1000 - Prédio 10 - Sala 440 - Santa Maria - RS



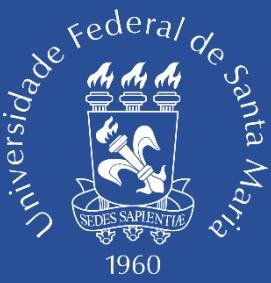
www.ufsm.br/gedre



+55 55 3220 9492



gedre.ufsm



UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
GEDRE - INTELIGÊNCIA EM ILUMINAÇÃO

gedre
inteligência
em iluminação

Aula 9 - PWM

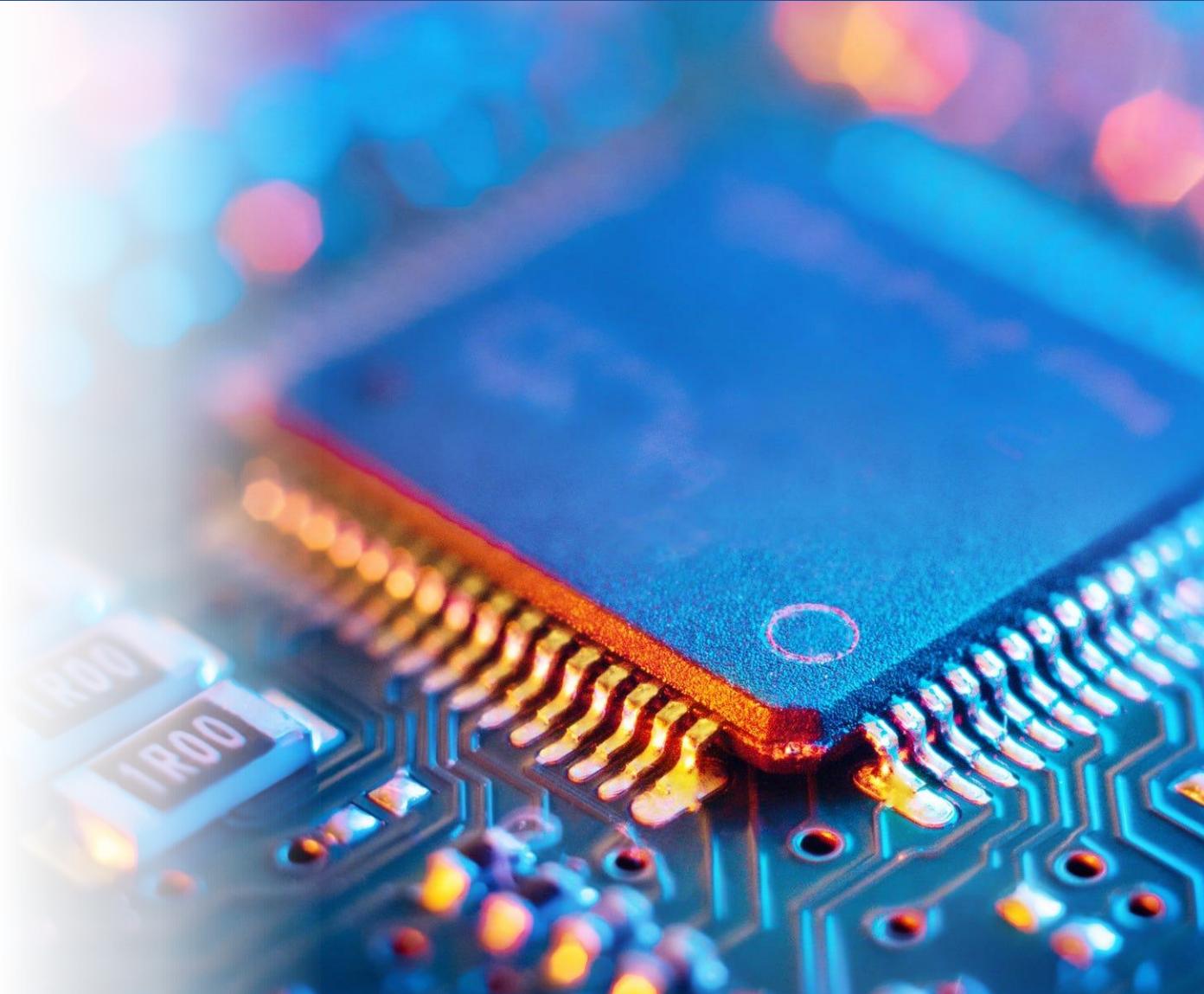
Prof. Renan Duarte

Minicurso de microcontroladores

Março de 2025

Conteúdo

- Introdução
- PWM no TM4C123GH6
- Projeto 6 - PWM



Modulação por largura de pulso - PWM

O que é?

- Tipo de conversão digital-analógica em que o nível desejado é modulado pela largura de um pulso digital

Como funciona?

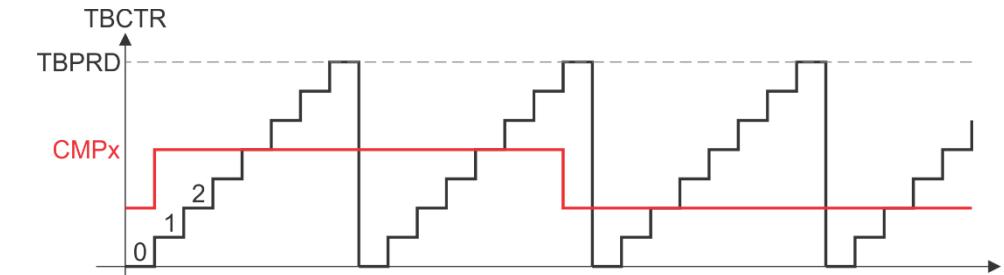
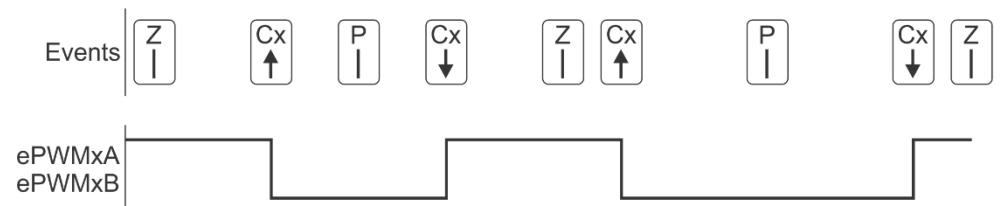
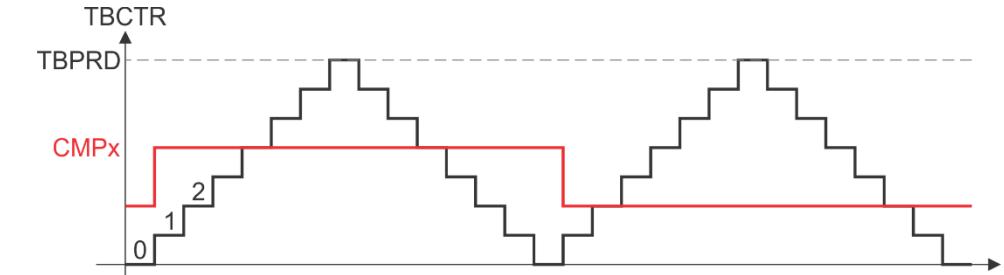
- Compara o valor de contagem de um timer periódico com um valor pré-definido

Parâmetros importantes

- Frequência: velocidade dos pulsos
- Duty Cycle: porcentagem do tempo em nível alto
- Tipo: PWM centrado, PWM up (ou down)

Aplicações

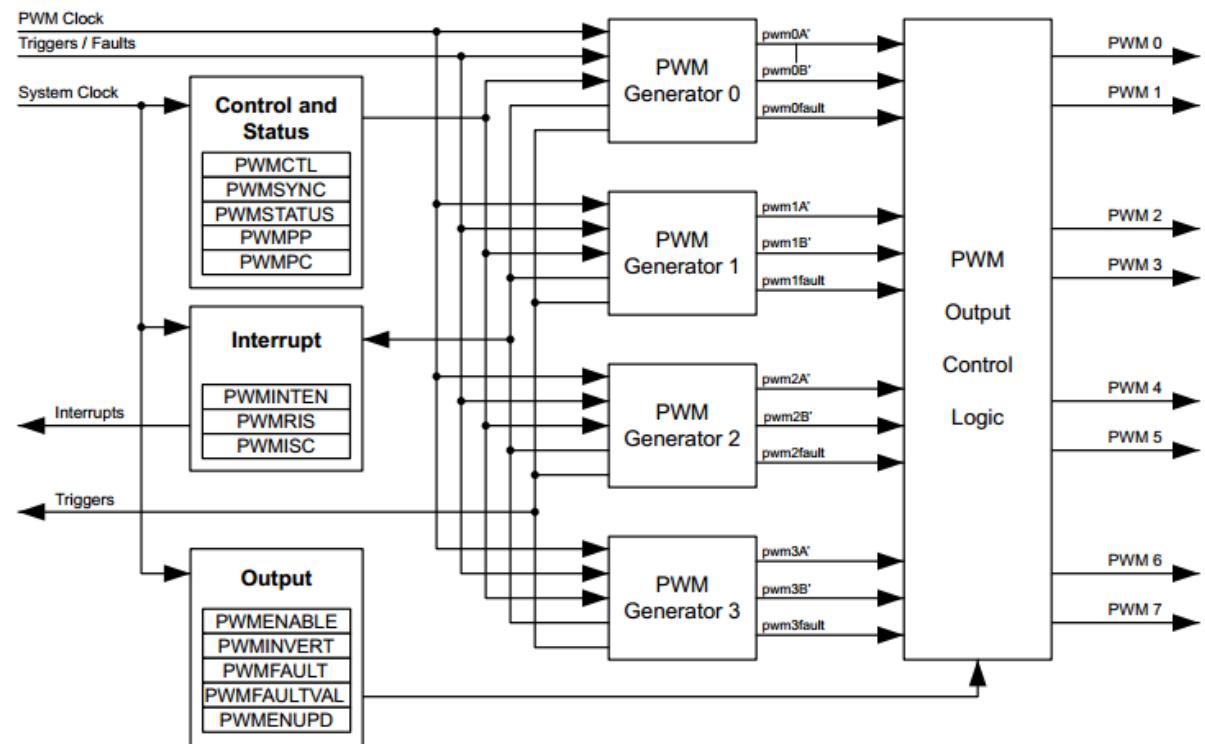
- Controle de motores
- Regulagem de brilho de LEDs
- Controle de conversores de energia



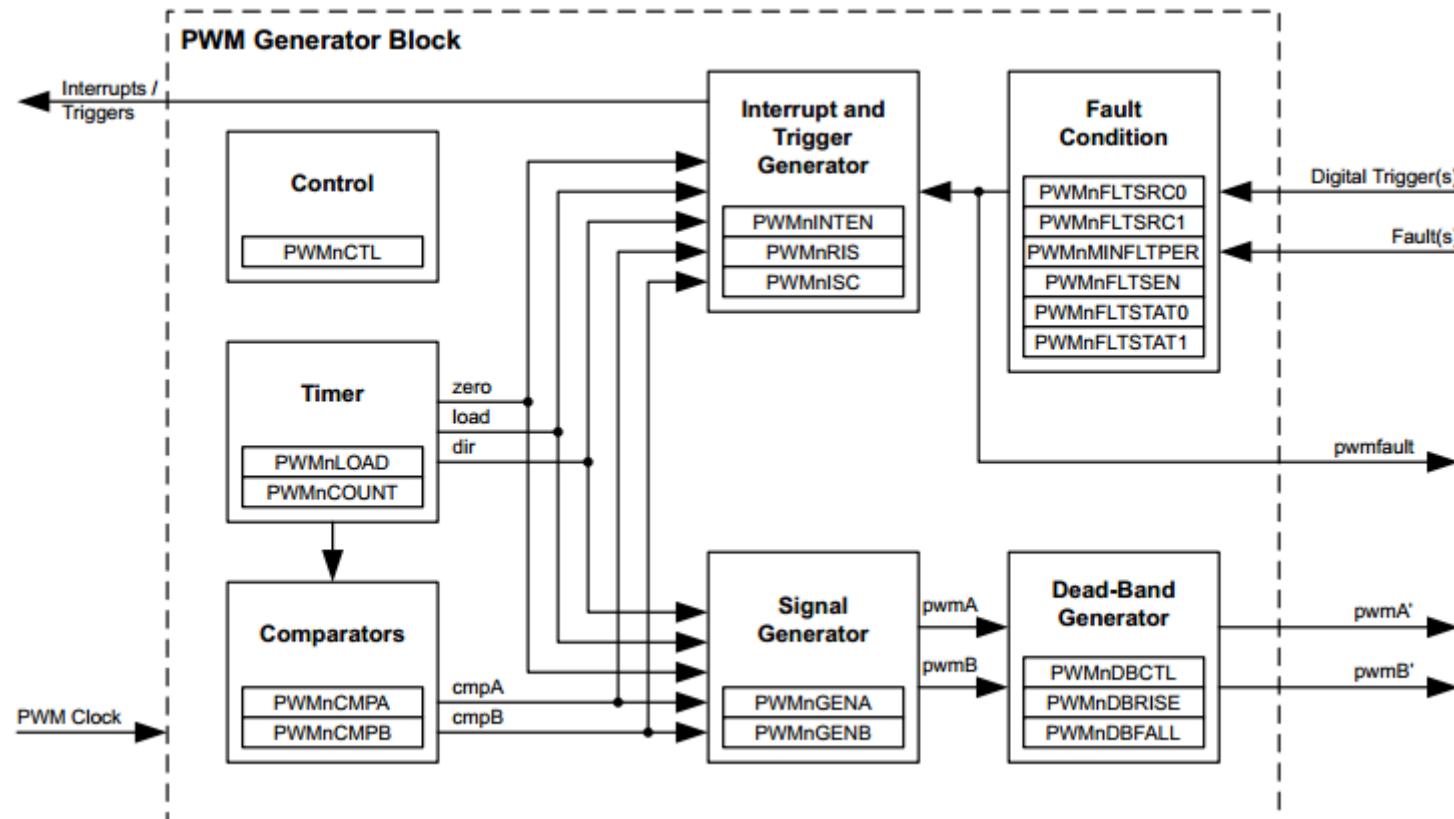
PWM no TM4C123GH6

Características

- 2 módulos PWM (PWM0 e PWM1)
- 4 geradores por módulo (cada gerador controla 2 saídas)
- Até 8 saídas PWM independentes
- Resolução de duty cycle de 16 bits
- Frequência ajustável via divisores do clock
- Suporte a modos de atualização síncrona e assíncrona
- Pode ser acionado por software ou eventos externos (por exemplo, timers)
- Compatível com dead-band generation



PWM no TM4C123GH6



PWM no TM4C123GH6

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type ^a	Description
M0FAULT0	30 53 63	PF2 (4) PD6 (4) PD2 (4)	I	TTL	Motion Control Module 0 PWM Fault 0.
M0PWM0	1	PB6 (4)	O	TTL	Motion Control Module 0 PWM 0. This signal is controlled by Module 0 PWM Generator 0.
M0PWM1	4	PB7 (4)	O	TTL	Motion Control Module 0 PWM 1. This signal is controlled by Module 0 PWM Generator 0.
M0PWM2	58	PB4 (4)	O	TTL	Motion Control Module 0 PWM 2. This signal is controlled by Module 0 PWM Generator 1.
M0PWM3	57	PB5 (4)	O	TTL	Motion Control Module 0 PWM 3. This signal is controlled by Module 0 PWM Generator 1.
M0PWM4	59	PE4 (4)	O	TTL	Motion Control Module 0 PWM 4. This signal is controlled by Module 0 PWM Generator 2.
M0PWM5	60	PE5 (4)	O	TTL	Motion Control Module 0 PWM 5. This signal is controlled by Module 0 PWM Generator 2.
M0PWM6	16 61	PC4 (4) PD0 (4)	O	TTL	Motion Control Module 0 PWM 6. This signal is controlled by Module 0 PWM Generator 3.
M0PWM7	15 62	PC5 (4) PD1 (4)	O	TTL	Motion Control Module 0 PWM 7. This signal is controlled by Module 0 PWM Generator 3.
M1FAULT0	5	PF4 (5)	I	TTL	Motion Control Module 1 PWM Fault 0.
M1PWM0	61	PD0 (5)	O	TTL	Motion Control Module 1 PWM 0. This signal is controlled by Module 1 PWM Generator 0.
M1PWM1	62	PD1 (5)	O	TTL	Motion Control Module 1 PWM 1. This signal is controlled by Module 1 PWM Generator 0.
M1PWM2	23 59	PA6 (5) PE4 (5)	O	TTL	Motion Control Module 1 PWM 2. This signal is controlled by Module 1 PWM Generator 1.
M1PWM3	24 60	PA7 (5) PE5 (5)	O	TTL	Motion Control Module 1 PWM 3. This signal is controlled by Module 1 PWM Generator 1.
M1PWM4	28	PF0 (5)	O	TTL	Motion Control Module 1 PWM 4. This signal is controlled by Module 1 PWM Generator 2.
M1PWM5	29	PF1 (5)	O	TTL	Motion Control Module 1 PWM 5. This signal is controlled by Module 1 PWM Generator 2.
M1PWM6	30	PF2 (5)	O	TTL	Motion Control Module 1 PWM 6. This signal is controlled by Module 1 PWM Generator 3.
M1PWM7	31	PF3 (5)	O	TTL	Motion Control Module 1 PWM 7. This signal is controlled by Module 1 PWM Generator 3.

Projeto 6 - PWM

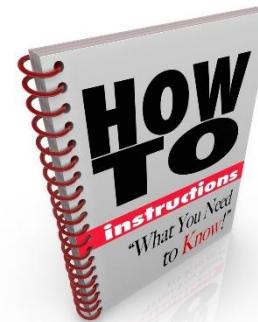
Objetivo:

- Obter um sinal PWM
- Variar a razão cíclica e observar os resultados



Passo a passo:

- Configurar PF1 como saída PWM
- Definir período do PWM
- Configurar módulo PWM
- Habilitar módulo PWM
- Variar razão cíclica



Projeto 6 - PWM - Desafio

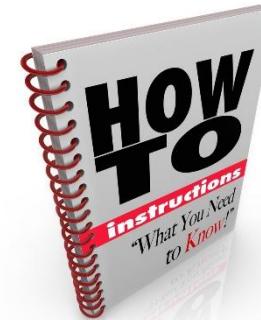
Objetivo:

- Gerar 3 sinais PWM para controlar o LED RGB da placa
- Obter cores diferentes variando a razão cíclica



Passo a passo:

- Configurar PF1, PF2, PF3 como saída PWM
- Definir período do PWM
- Configurar módulos PWM (módulo 1, saídas 5, 6 e 7)
- Habilitar módulos PWM
- Variar razão cíclica (3 canais)





renan.duarte@gedre.ufsm.br

Av. Roraima 1000 - Prédio 10 - Sala 440 - Santa Maria - RS



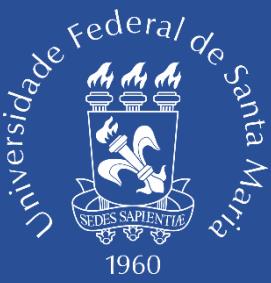
www.ufsm.br/gedre



+55 55 3220 9492



gedre.ufsm



UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
GEDRE - INTELIGÊNCIA EM ILUMINAÇÃO

gedre
inteligência
em iluminação

Aula 10 - Desafio final

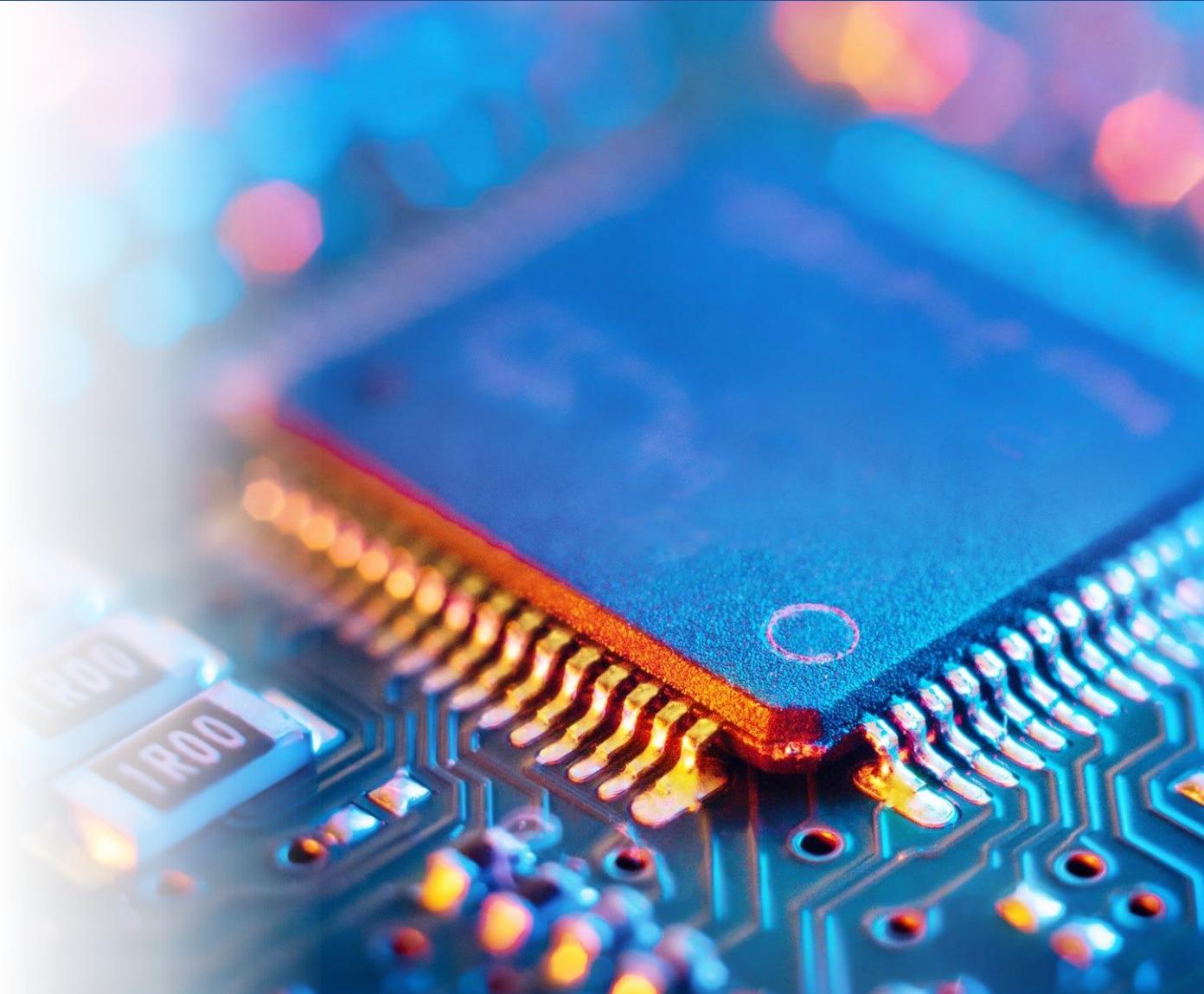
Prof. Renan Duarte

Minicurso de microcontroladores

Março de 2025

Conteúdo

- Introdução
- Projeto 7 - Desafio final

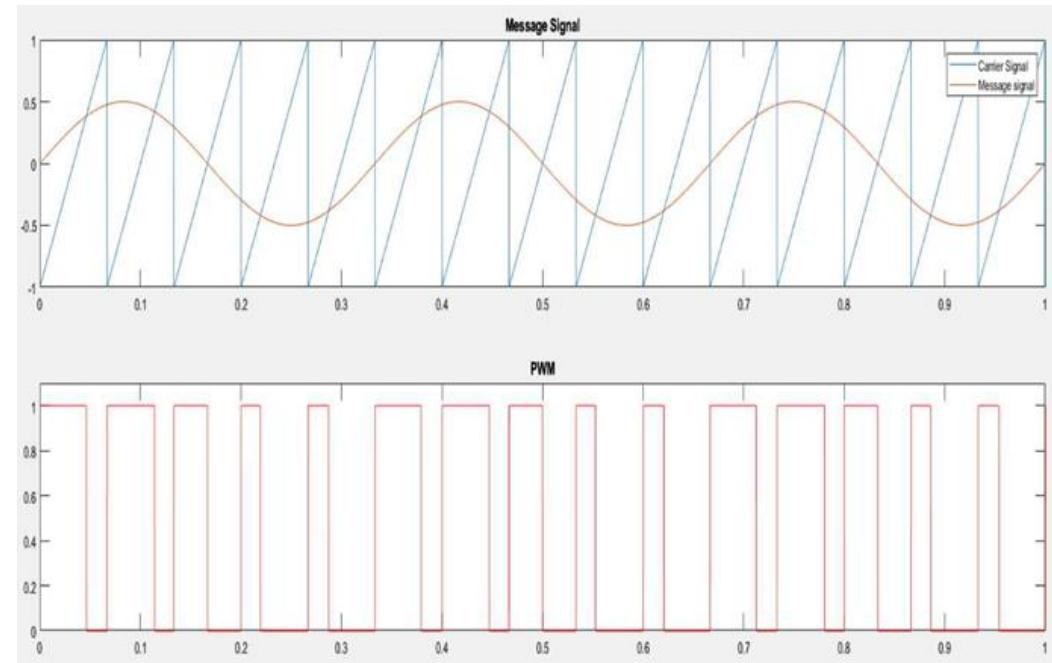


Gerador de sinais com modulador PWM

Se variarmos a razão cíclica de um PWM de acordo com outra forma de onda, estamos modulando esse sinal na largura dos pulsos.

Após, podemos filtrar o PWM e obter novamente o sinal original.

Se gerarmos digitalmente essa forma de onda desejada, podemos criar um gerador de sinais.



Projeto 7 - Desafio final



Objetivo:

- Aplicar os conceitos vistos em todo o custo
- Criar um gerador de sinais com o modulador PWM:
 - Deve gerar: Senoide, senoide retificada, triangular e onda quadrada
 - Botão da placa deve alterar a forma de onda
 - Potenciômetro (ADC) deve controlar a frequência da onda (10 à 100 Hz)
- Usar 312,5kHz como frequência do PWM
- Usar 1kHz como frequência de amostragem (timer)



renan.duarte@gedre.ufsm.br

Av. Roraima 1000 - Prédio 10 - Sala 440 - Santa Maria - RS



www.ufsm.br/gedre



+55 55 3220 9492



gedre.ufsm