

Configuração do VS Code para Raspberry Pi Pico W

Unidade 3 | Capítulo 4

Prof. Jorge Wattes



Executores:



Coordenação:



Iniciativa:



Sumário

Introdução	3
Raspberry Pi Pico	14
Monitor Serial	15
Em Resumo	17
Conclusão	18
Referências	19

Boas-vindas e introdução

Olá, estudantes, sejam bem-vindos à aula sobre como configurar o ambiente integrado de desenvolvimento VS Code para programar o microcontrolador RP2040 da placa Raspberry Pi Pico W.

Ao final desta aula, espero que você seja capaz de configurar o VS Code para programação de sistemas embarcados através da placa Raspberry Pi Pico. Também, que você consiga realizar comunicação serial com ela, tendo finalidade de depuração. Além disso, você poderá programar, na prática, sua placa BitDogLab.

3.1 Raspberry Pi Pico

Configuração do VS Code para Raspberry Pi Pico.

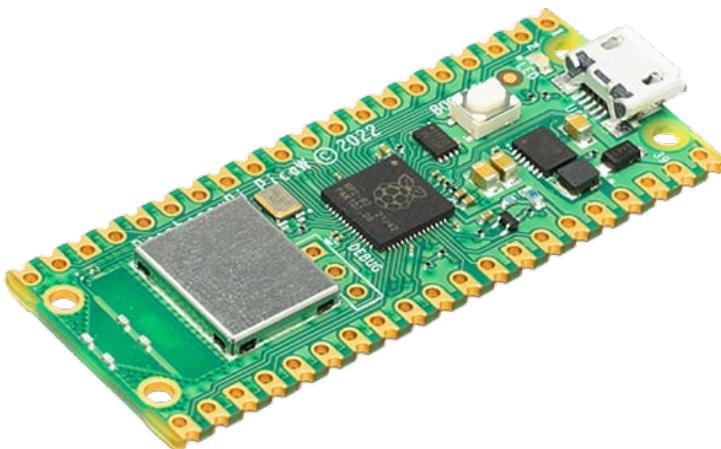


Figura 1 - Raspberry Pi Pico

FIGURA 1:#PRACEGOVER: A imagem mostra uma placa eletrônica verde, que é uma Raspberry Pi Pico W. Esta é uma placa de desenvolvimento de baixo custo e pequeno porte, utilizada em projetos de eletrônica e programação. A placa tem formato retangular e apresenta componentes eletrônicos soldados em sua superfície. No centro, há um microcontrolador RP2040, um chip quadrado responsável pelo processamento. Do lado esquerdo, um módulo de metal prateado que representa o chip de Wi-Fi. A placa também contém pinos dourados em suas laterais, usados para conexão com outros dispositivos ou sensores. Há um botão branco no canto superior direito, próximo a uma porta micro USB, que é utilizada para alimentação e comunicação com um computador.

Fonte: <https://www.robocore.net/placa-raspberry-pi/raspberry-pi-pico-w>

Para executar um código C em um microcontrolador, é essencial utilizar um compilador e um ambiente de desenvolvimento configurado para

o hardware específico. Antes de implementar o código diretamente no microcontrolador, é comum realizar testes para garantir que o código funcione conforme planejado.

Nesse contexto, o uso de ferramentas de simulação se torna crucial, pois permite verificar o comportamento do código sem a necessidade de hardware físico, reduzindo custos e otimizando o processo de desenvolvimento.

A plataforma online <https://wokwi.com/> é uma excelente opção para simulação, oferecendo suporte para diversos modelos de microcontroladores, incluindo o Raspberry Pi Pico, que será a base do nosso curso. Através desta plataforma, poderemos testar e validar nossos códigos de forma eficiente, antes de transferi-los para o hardware real.

O Raspberry Pi Pico é uma plataforma acessível e poderosa, ideal para cursos de sistemas embarcados com foco na programação Bare Metal em linguagem C. Ele é equipado com o microcontrolador RP2040, um chip dual-core ARM Cortex-M0+ que oferece 264KB de RAM, sendo capaz de executar tarefas complexas em tempo real.

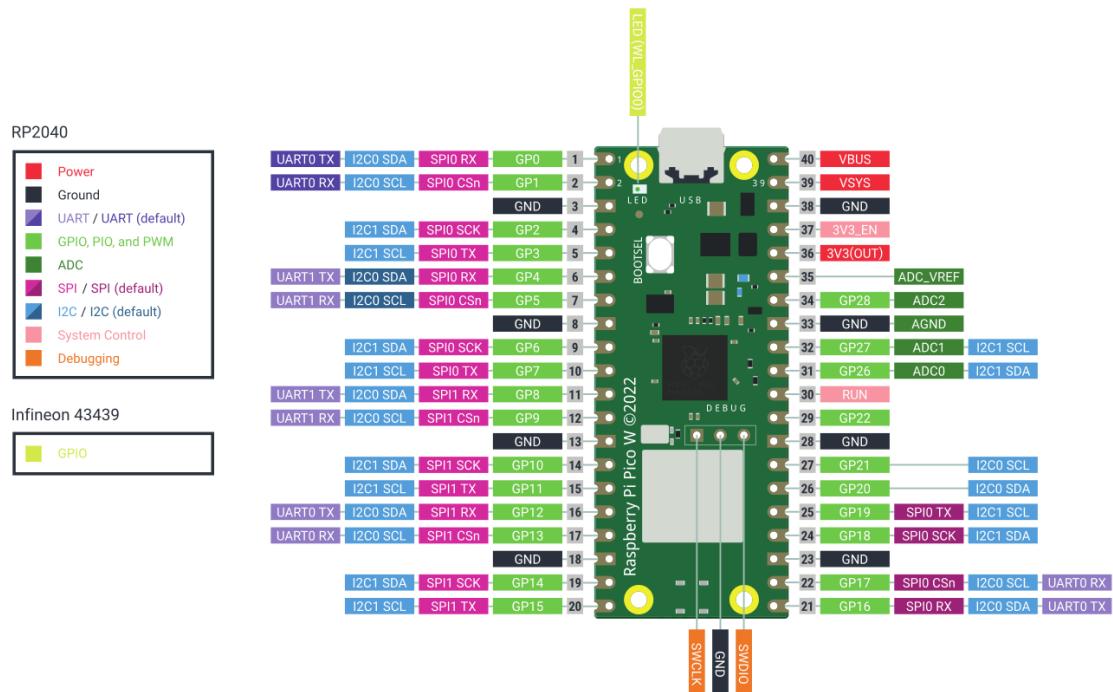


Figura 2 - Pinout do Raspberry Pi Pico

Fonte: <https://www.alexwilson.dev/learning-in-public/gpio-on-the-raspberry-pi-pico>

Este curso utilizará o Raspberry Pi Pico para explorar a programação de baixo nível, permitindo que os alunos interajam diretamente com o hardware. Os periféricos do RP2040, como GPIO, I2C, SPI, UART e ADC, serão fundamentais para a construção de projetos práticos, onde os alunos aprenderão a controlar dispositivos externos, como LEDs, sensores e motores, sem a dependência de sistemas operacionais ou bibliotecas de alto nível.

O foco em Bare Metal garante que você compreenda o funcionamento do hardware, adquirindo habilidades essenciais para otimização de recursos e controle preciso em aplicações embarcadas. Este aprendizado direto com o hardware, combinado com a versatilidade dos periféricos do Raspberry Pi Pico, prepara os alunos para enfrentar desafios técnicos em diversas áreas da engenharia eletrônica e computação.

Utilizaremos também a BitDogLab, uma placa de desenvolvimento educacional repleta de periféricos que conta com um Raspberry Pi Pico W. Conteúdos a respeito da podem ser explorados em seu repositório ?? (<https://github.com/BitDogLab/BitDogLab>).

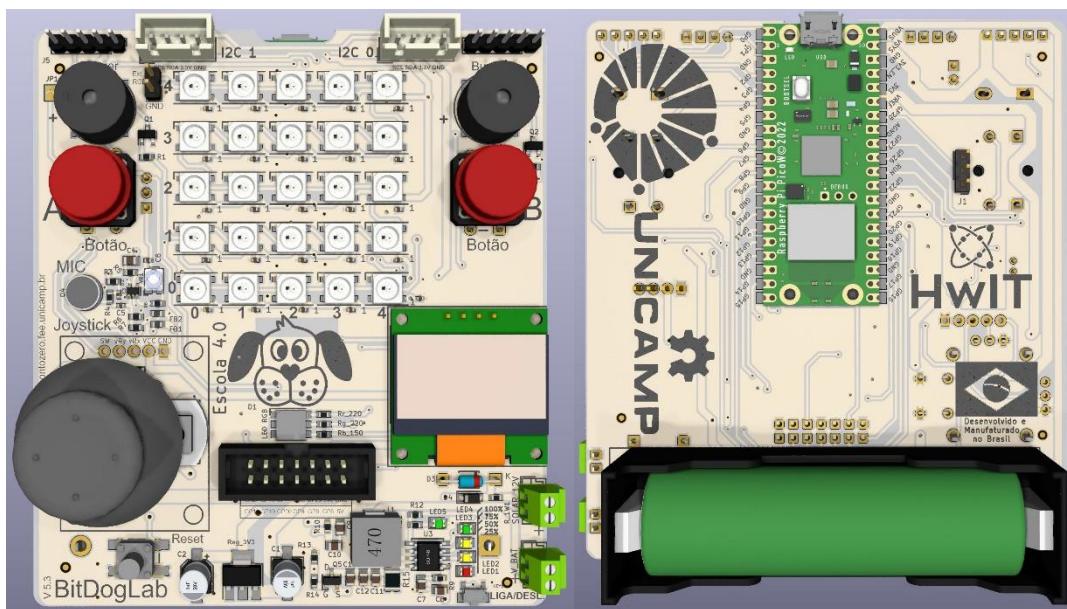


Figura 3 - Pinout do Raspberry Pi Pico

Figura 3 #pracegover: A imagem mostra uma placa de desenvolvimento eletrônico de dois lados, com componentes variados dispostos em cada uma das partes. **Lado esquerdo da imagem:** A placa à esquerda exibe diversos componentes interativos. Na parte superior, há dois conectores I2C próximos a uma fileira de botões à direita e à esquerda. No canto inferior esquerdo, há um joystick com tampa cinza e ao lado está o texto "BitDogLab", junto a um microfone identificado como MIC. Na parte central, pode-se ver um pequeno display LCD colorido para exibição de informações e vários circuitos e conectores. Também há um conector para baterias e componentes de controle de energia na parte inferior direita.

Lado direito da imagem: A segunda placa apresenta o logotipo da Unicamp e a marca "HwIT" na parte superior. No centro, está fixado um Raspberry Pi Pico, já mencionado anteriormente, que se conecta aos circuitos da placa. A parte inferior contém uma bateria cilíndrica verde, instalada em um suporte preto, fornecendo alimentação para o dispositivo. Diversos circuitos e conectores estão visíveis ao redor da placa, indicando sua complexidade e versatilidade em projetos eletrônicos.

Fonte: <https://hwit.com.br/produto/bitdoglab/>

A seguir, vamos ver o passo a passo necessário para configurar o VS Code para a programação do Raspberry Pi Pico na placa BitDogLab.

PROGRAMANDO O RASPBERRY PI PICO

1º passo: Instalação do VS Code

Acesse o link [🔗 https://code.visualstudio.com/download](https://code.visualstudio.com/download) e baixe a última versão do VS Code de acordo com a versão de seu sistema operacional. Durante a instalação preste atenção no segundo passo, deixe a opção Adicionar ao Path marcada. Ao concluir a instalação, reinicie seu computador.

2º passo: Instalação Compilador ARM GCC

Acesse o Link [🔗 https://developer.arm.com/Tools%20and%20Software/GNU%20Toolchain](https://developer.arm.com/Tools%20and%20Software/GNU%20Toolchain) e baixe a última versão do AArch32 bare-metal target na versão .exe, e execute-o para iniciar a instalação. Observe o diretório de instalação (Ex: C:\Program Files (x86)\Arm GNU Toolchain arm-none-eabi\13.3 rel1).

Link direto para o download do programa:

[🔗 https://developer.arm.com/-/media/Files/downloads/gnu/13.3.rel1/binrel/arm-gnu-toolchain-13.3.rel1-mingw-w64-i686-arm-none-eabi.exe](https://developer.arm.com/-/media/Files/downloads/gnu/13.3.rel1/binrel/arm-gnu-toolchain-13.3.rel1-mingw-w64-i686-arm-none-eabi.exe)

What's new in 13.3.Rel1

This release is based on GCC 13.3

In this release

Windows (mingw-w64-i686) hosted cross toolchains
AArch32 bare-metal target (arm-none-eabi)

- [arm-gnu-toolchain-13.3.rel1-mingw-w64-i686-arm-none-eabi.zip](#)
- [arm-gnu-toolchain-13.3.rel1-mingw-w64-i686-arm-none-eabi.zip.asc](#)
- [arm-gnu-toolchain-13.3.rel1-mingw-w64-i686-arm-none-eabi.zip.sha256asc](#)
- [arm-gnu-toolchain-13.3.rel1-mingw-w64-i686-arm-none-eabi.exe](#)
- [arm-gnu-toolchain-13.3.rel1-mingw-w64-i686-arm-none-eabi.exe.asc](#)
- [arm-gnu-toolchain-13.3.rel1-mingw-w64-i686-arm-none-eabi.exe.sha256asc](#)

Figura 4 – Página de Download do AArch32 bare-metal target

Figura 4 #pracegover: A imagem mostra uma seção de uma página de lançamento de software, especificamente da versão 13.3.Rel1 de uma ferramenta de compilação GCC 13.3. O título "What's new in 13.3.Rel1" aparece no topo, seguido de uma descrição que indica que essa versão é baseada no GCC 13.3. Abaixo, há uma lista de arquivos disponíveis para download, projetados para o sistema Windows (mingw-w64-i686) e direcionados a toolchains de cruzamento para AArch32 bare-metal target (arm-none-eabi).

No seção de arquivos disponíveis para download, cada link tem o nome do arquivo e a extensão, incluindo as seguintes opções:

[arm-gnu-toolchain-13.3.rel1-mingw-w64-i686-arm-none-eabi.zip](#): arquivo compactado principal.

[arm-gnu-toolchain-13.3.rel1-mingw-w64-i686-arm-none-eabi.zip.asc](#): assinatura ASC para o arquivo ZIP.

[arm-gnu-toolchain-13.3.rel1-mingw-w64-i686-arm-none-eabi.zip.sha256asc](#): assinatura SHA256 para verificar a integridade do arquivo ZIP.

[arm-gnu-toolchain-13.3.rel1-mingw-w64-i686-arm-none-eabi.exe](#): arquivo executável principal da ferramenta.

[arm-gnu-toolchain-13.3.rel1-mingw-w64-i686-arm-none-eabi.exe.asc](#): assinatura ASC para o arquivo EXE.

[arm-gnu-toolchain-13.3.rel1-mingw-w64-i686-arm-none-eabi.exe.sha256asc](#): assinatura SHA256 para o arquivo EXE.

Cada link fornece acesso a uma versão diferente do compilador, com os arquivos em formatos .zip e .exe, além de suas respectivas assinaturas digitais para verificação. A organização dos links facilita o acesso direto aos arquivos relacionados ao lançamento dessa versão específica.

Fonte: <https://developer.arm.com/downloads/-/arm-gnu-toolchain-downloads/13-2-rel1>

No final da instalação, marque a caixa de seleção de **Add Path to environment Variable.**



Figura 5 – Telas do processo de instalação da ferramenta **Arm GNU Toolchain 13.3.rel1 arm-none-eabi** no Windowset

Figura 5 #pracegover: A imagem mostra duas capturas de tela do processo de instalação da ferramenta Arm GNU Toolchain 13.3.rel1 arm-none-eabi no Windows.

Primeira janela (à esquerda): A primeira janela é a tela de boas-vindas do instalador, com o título "Bem-vindo ao Instalador do Arm GNU Toolchain 13.3.rel1 arm-none-eabi". O texto informa que o instalador guiará o usuário pelo processo de instalação e recomenda fechar todos os outros aplicativos antes de iniciar o instalador. Também menciona que pode ser necessário reiniciar o computador após a instalação. Abaixo da mensagem, há dois botões: "Próximo >" (para continuar o processo) e "Cancelar" (para interromper a instalação).

Segunda janela (à direita): A segunda janela mostra a tela final do processo de instalação, com o título "Completando a instalação do Arm GNU Toolchain 13.3.rel1 arm-none-eabi". O texto indica que a instalação foi concluída com sucesso e que a ferramenta já está instalada no computador. Abaixo, existem quatro opções marcáveis:

Mostrar o Readme: para exibir o arquivo de instruções pós-instalação.

Launch openocd.bat: para executar um script de inicialização.

Add path to environment variable: para adicionar o caminho da ferramenta às variáveis de ambiente do sistema.

Add registry information: para incluir informações no registro do Windows.

Na parte inferior, há dois botões: "Voltar" (para revisar etapas anteriores) e "Concluir" (para finalizar o instalador). A cor predominante do fundo das janelas é azul, com o ícone padrão de instalação no canto superior esquerdo.

Fonte: autor

3º passo: Instalação do Raspberry Pi Pico SDK

Acesse o link: [🔗 https://github.com/raspberrypi/pico-setup-windows/releases](https://github.com/raspberrypi/pico-setup-windows/releases) e baixe a última versão do `pico-setup-windows-x64-standalone.exe` e do `openocd-x64-standalone.zip`.

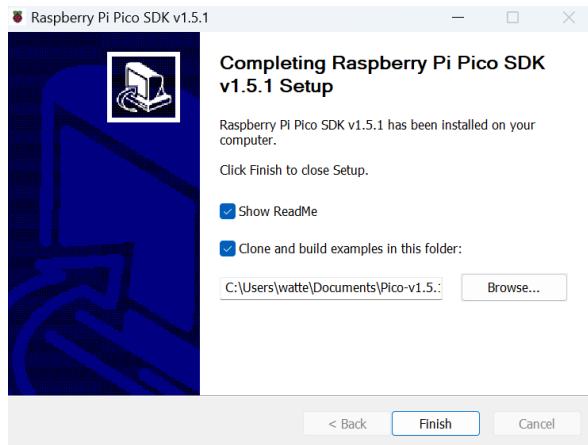


Figura 6 - Tela de finalização do instalador do Raspberry Pi Pico SDK v1.5.1 no Windows

Figura 6 #pracegover: A imagem mostra a tela de finalização do instalador do **Raspberry Pi Pico SDK v1.5.1** no Windows.

O título da janela é "**Completing Raspberry Pi Pico SDK v1.5.1 Setup**", indicando que a instalação foi concluída com sucesso. A mensagem abaixo informa que o **Raspberry Pi Pico SDK v1.5.1** foi instalado no computador e instrui o usuário a clicar em "**Finish**" (Concluir) para fechar o instalador.

Abaixo dessa mensagem, há duas opções que o usuário pode marcar:
Show ReadMe: para exibir o arquivo ReadMe (documento com informações sobre a instalação).
Clone and build examples in this folder: para clonar e construir exemplos no diretório indicado, com a opção de selecionar um caminho diferente através do botão **Browse...**

Na parte inferior da janela, há três botões: "**Back**" (para voltar a uma etapa anterior), "**Finish**" (para concluir a instalação), e "**Cancel**" (para cancelar o processo). A janela tem um fundo azul escuro com o ícone padrão de instalação no canto superior esquerdo.

4º passo: Criar as variáveis de ambiente

Acesse o Explorador de Arquivos, acesse propriedades de Meu Computador, e clique em Configurações avançadas do sistema e em seguida "Variáveis de Ambiente ...". Selecione path, clique em editar. Irá abrir uma nova tela, depois clique em novo, copie e cole os endereços abaixo.

Adicione ao PATH as seguintes linhas:

C:\Program Files(x86)\Arm GNU Toolchain arm-none-eabi\13.3 rel1\bin

C:\Program Files\Raspberry Pi\Pico SDK v1.5.1\pico-sdk\toolchain\13.2 Rel1\bin

Adicione as seguintes variáveis do sistema:

Variável	Valor
PICO_SDK_PATH	C:\Program Files\Raspberry Pi\Pico SDK v1.5.1
PICO_TOOLCHAIN_PATH	C:\Program Files\Raspberry Pi\Pico SDK v1.5.1\pico-sdk\toolchain

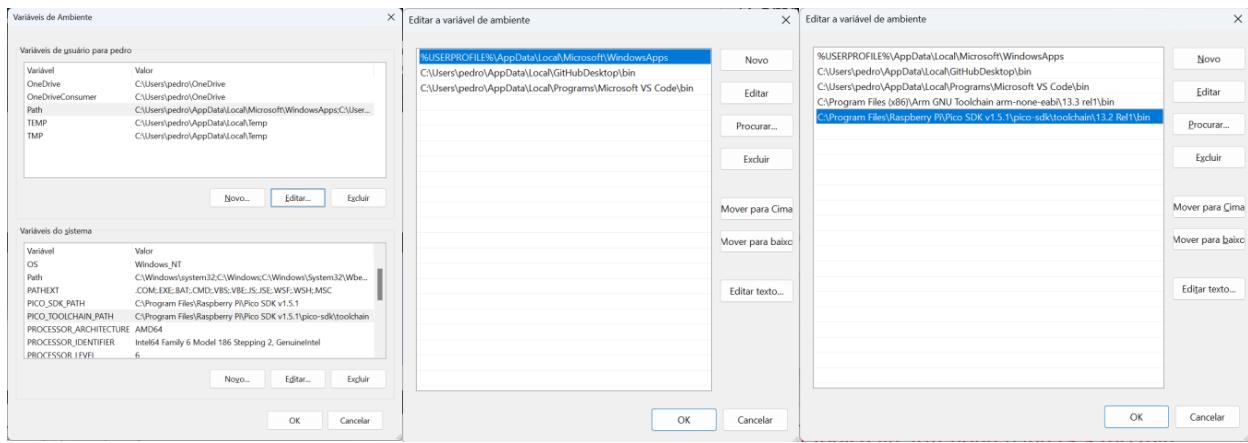


Figura 7 - Configuração de variáveis de ambiente

A imagem mostra três janelas diferentes relacionadas à configuração de variáveis de ambiente no sistema operacional Windows.

Primeira janela (à esquerda): A janela principal é a tela de **Variáveis de Ambiente**, onde é possível ver duas listas: uma para variáveis de ambiente do usuário e outra para variáveis de ambiente do sistema. Na parte superior, está a seção de variáveis do usuário, listando variáveis como OneDrive, TEMP, e TMP, que apontam para diretórios temporários no sistema. Na parte inferior, há a seção de variáveis do sistema, contendo variáveis como Path, PROCESSOR_ARCHITECTURE, e outras que influenciam a execução de programas. Existem três botões disponíveis: **Novo**, **Editar** e **Excluir** para adicionar, modificar ou remover variáveis.

Segunda janela (ao centro): A segunda janela mostra a edição da variável de ambiente Path do usuário. A variável Path contém uma lista de diretórios onde o sistema deve procurar executáveis. A lista de diretórios inclui pastas como C:\Users\pedro\AppData\Local\Microsoft\WindowsApps e outras pastas relacionadas a ferramentas de desenvolvimento, como o Microsoft VS Code\bin. Há opções de **Novo**, **Editar**, **Procurar...**, **Excluir**, e botões para mover os itens da lista para cima ou para baixo.

Terceira janela (à direita): A terceira janela também mostra a edição da variável Path, agora com um novo diretório adicionado à lista: C:\Program Files\Raspberry Pi\Pico-sdk e C:\Program Files(x86)\Arm Toolchain..., que referenciam os locais de instalação do SDK do Raspberry Pi e do Arm GNU Toolchain. Assim como na segunda janela, há opções para adicionar ou remover diretórios, além de mover os caminhos para cima ou para baixo na ordem de prioridade.

Fonte: autor

5º passo: Plugins VS Code

Abra o VS Code, acesse o menu de extensão do VS Code e instale os seguintes Plugins:

- C/C++ (Já instalado anteriormente)

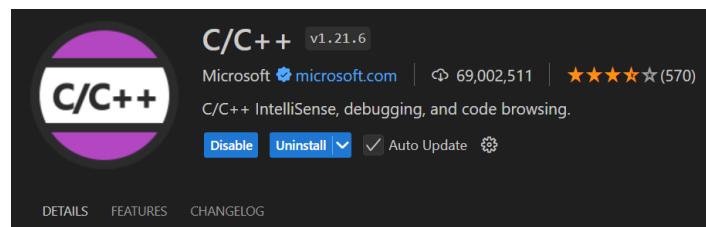


Figura 8 – Plugin C/C++

Figura 8 #pracegover: A imagem exibe a extensão **C/C++ v1.21.6**, desenvolvida pela **Microsoft**, disponível no Visual Studio Code. A extensão tem **69.002.511 downloads** e uma avaliação de **4,4 estrelas** com base em **570 avaliações**. Ela oferece suporte para IntelliSense, depuração e navegação de código em C/C++. Abaixo, há opções para desabilitar, desinstalar e habilitar **atualizações automáticas** da extensão.

Fonte: <https://microhobby.com.br/blog/2023/08/13/depurando-aplicacoes-qt-no-vs-code-com-debug-helpers/>

• CMake

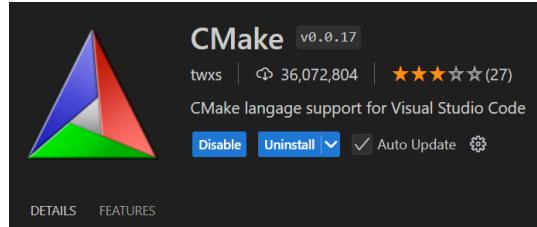


Figura 9 - CMake

Figura 9 #pracegover: Esta imagem apresenta a extensão **CMake v0.0.17**, desenvolvida por **twxs**, disponível no Visual Studio Code, com **36.072.804 downloads**. A extensão possui uma avaliação de **3,2 estrelas**, baseada em **27 avaliações**. A descrição indica que ela fornece suporte para a linguagem CMake no Visual Studio Code. As opções incluem **desabilitar**, **desinstalar** e **atualizações automáticas** habilitadas.

Fonte: <https://medium.com/@mehmet-mert-gunduz/enhance-your-c-c-development-top-vs-code-extensions-for-maximized-productivity-a3a2a968d628>

• CMake Tools

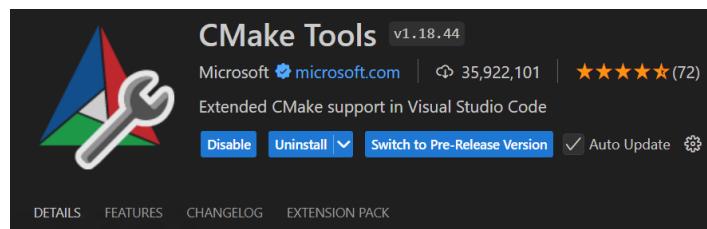


Figura 10 – Cmake Tools

#pracegover: imagem exibe a extensão **CMake Tools v1.18.44**, também desenvolvida pela **Microsoft**, com **35.922.101 downloads** e uma avaliação de **4,9 estrelas**, com base em **72 avaliações**. Essa extensão oferece suporte estendido para o CMake no Visual Studio Code. As opções disponíveis são **desabilitar**, **desinstalar**, **mudar para versão pré-lançamento** e **atualizações automáticas** habilitadas.

Fonte: <https://medium.com/@mehmet-mert-gunduz/enhance-your-c-c-development-top-vs-code-extensions-for-maximized-productivity-a3a2a968d628>

Clique no ícone de engrenagem e depois em Extension Settings para acessar as configurações do Plugin:

- Digite “**Cmake Path**” e confira se o nome é **cmake**.

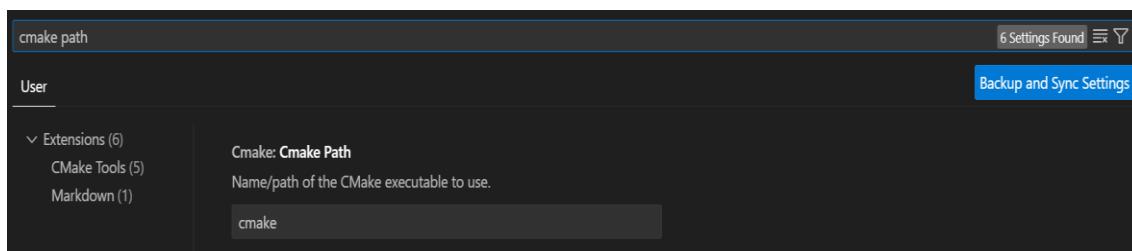
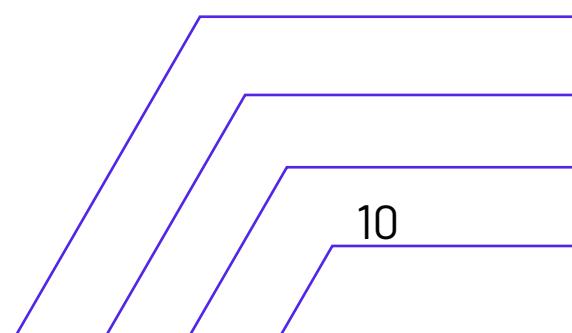


Figura 11 – Cmake Path

Figura 11 #pracegover: A imagem mostra uma tela de configurações do Visual Studio Code, onde a busca por “cmake path” foi realizada. No topo, há uma barra de busca com o texto “cmake path”, e ao lado direito, há um botão azul rotulado “Backup and Sync Settings”. Abaixo, na seção “User”, está listada a extensão CMake Tools, que oferece a opção de definir o caminho do executável CMake. Na caixa de texto, o caminho definido para o executável CMake é simplesmente “cmake”, permitindo que o sistema encontre o executável correspondente no ambiente de desenvolvimento.

Fonte: autor



- Digite Configure Environment, clique em Add Item, em Item digite PICO_SDK_PATH e em Value digite o caminho da pasta pico-sdk (C:\Program Files\Raspberry Pi\Pico SDK v1.5.1).

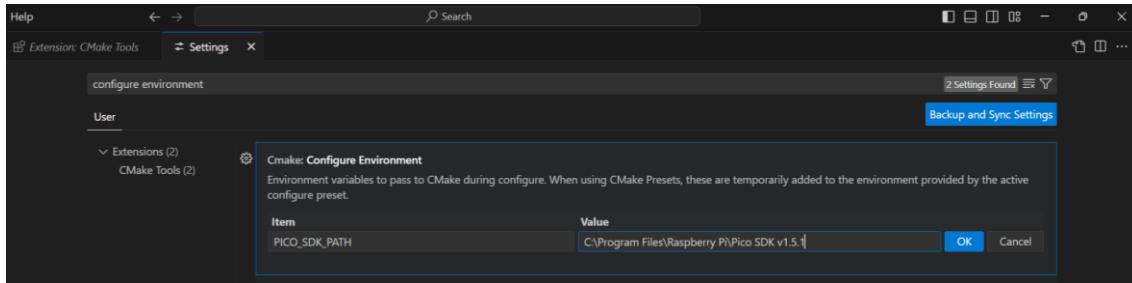


Figura 12 - Environment

Figura 12 #pracegover: A imagem mostra outra configuração no Visual Studio Code, onde a busca por "configure environment" foi realizada. A tela está configurada para ajustar as variáveis de ambiente que o CMake utiliza durante a configuração. Na lista de variáveis de ambiente, o item destacado é PICO_SDK_PATH, que aponta para o diretório "C:\Program Files\Raspberry Pi\pico-sdk v1.5.1". O campo permite que o usuário ajuste esse caminho conforme necessário para integrar o SDK do Raspberry Pi Pico ao ambiente CMake. A interface tem um botão azul de confirmação OK e um botão de Cancelar no canto inferior direito.

Fonte: autor

- Em Generator digite NMake Makefiles.

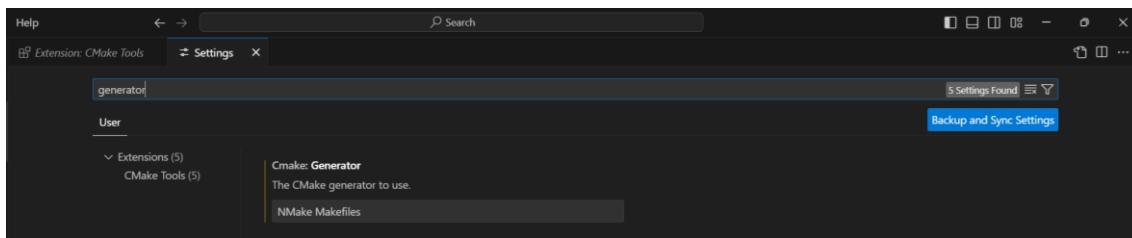


Figura 13 - Generator

Figura 13 #pracegover: Esta imagem mostra outra busca dentro das configurações do Visual Studio Code, desta vez com o termo "generator". A tela exibe a configuração do CMake Generator, que define qual gerador de arquivos de build o CMake deve usar. O campo "CMake Generator" contém a escolha NMake Makefiles, que é um dos geradores usados pelo CMake para criar arquivos Makefiles no ambiente NMake. A interface de configurações tem uma aparência limpa, com o botão "Backup and Sync Settings" no canto superior direito.

- Raspberry Pi Pico

Ainda no menu de extensão do VS Code (Atalho Ctrl+Shift+X) e instale o seguinte Plugin:

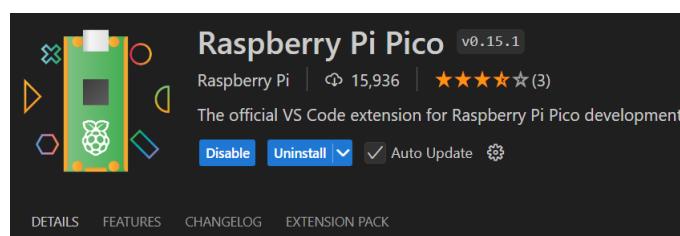


Figura 14 - Plugin Raspberry Pi Pico

Figura 14 #pracegover: imagem mostra a extensão Raspberry Pi Pico v0.15.1 no Visual Studio Code. O ícone da extensão exibe uma ilustração de um Raspberry Pi Pico com várias formas geométricas coloridas ao redor. A extensão é desenvolvida pela Raspberry Pi e possui 15.936 downloads, com uma avaliação média de 4,7 estrelas baseada em 3 avaliações. A descrição indica que esta é a extensão oficial para o desenvolvimento com o Raspberry Pi Pico. Existem botões para desabilitar a extensão, desinstalar e uma caixa de seleção marcada para atualização automática.

Utilizando o plugin do Raspberry Pi Pico é possível criar e compilar códigos. Para testar o funcionamento do ambiente, clique no menu do Plugin Pi Pico, do lado esquerdo. Clique em New Project From Examples, busque pelo exemplo Blink, board type: Pico W, local de sua preferência e finalize em Create. Irá abrir uma nova janela com o nome blink.

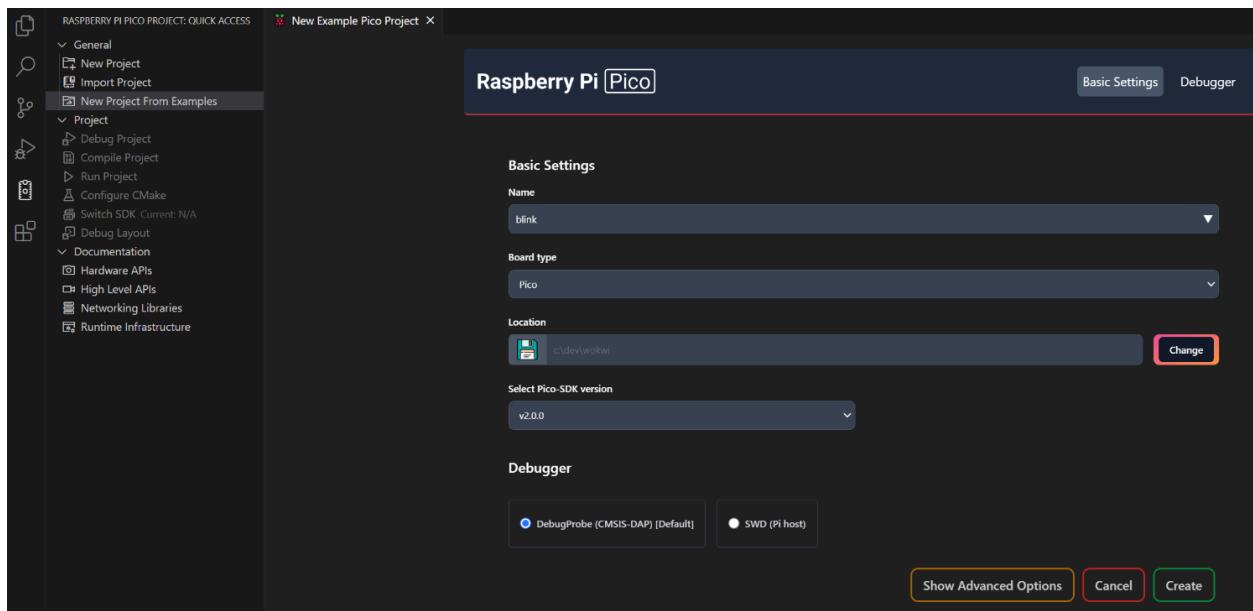


Figura 15 – interface de criação de um novo projeto no Raspberry Pi Pico Project

Figura 15 #pracegover: A imagem mostra a interface de criação de um novo projeto no **Raspberry Pi Pico Project**, dentro de um ambiente de desenvolvimento integrado (IDE). À esquerda, há um painel de navegação com várias opções, como **New Project**, **Import Project**, **New Project From Examples**, e outras categorias de configuração relacionadas ao projeto Pico.

À direita, está a tela principal intitulada **New Example Pico Project**, com as configurações básicas para iniciar um novo projeto. A seção **Basic Settings** inclui campos como:

Name: atualmente preenchido com o nome "blink".

Board Type: selecionado como Pico.

Location: o caminho do diretório está mostrado como "C:\dev\work..." com a opção de alterá-lo por meio do botão **Change**.

Select Pico-SDK version: a versão do SDK selecionada é **v2.0.0**.

No seção **Debugger**, o modo de depuração padrão é o **DebugProbe (CMSIS-DAP)**, com a opção alternativa **SWD (Pi host)**.

No canto inferior direito, há três botões: **Show Advanced Options** (para opções adicionais de configuração), **Cancel** (em vermelho), e **Create** (em verde) para criar o projeto. O design da interface é moderno, com fundo escuro e texto claro, proporcionando uma visualização clara das opções disponíveis.

Fonte: autor

Após a criação do arquivo, verificar se a pasta blink foi criada no diretório que você selecionou. Parabéns, mais um passo conquistado!

• NA PRÁTICA

Pode ser necessário reiniciar o VS Code ou o computador após a instalação dos Plugins para garantir o correto funcionamento!

6º passo: Instalando o Driver da placa Raspberry Pi Pico

Acesse o site do software Zadig (<https://zadig.akeo.ie/>) e realize o download e execute o mesmo.

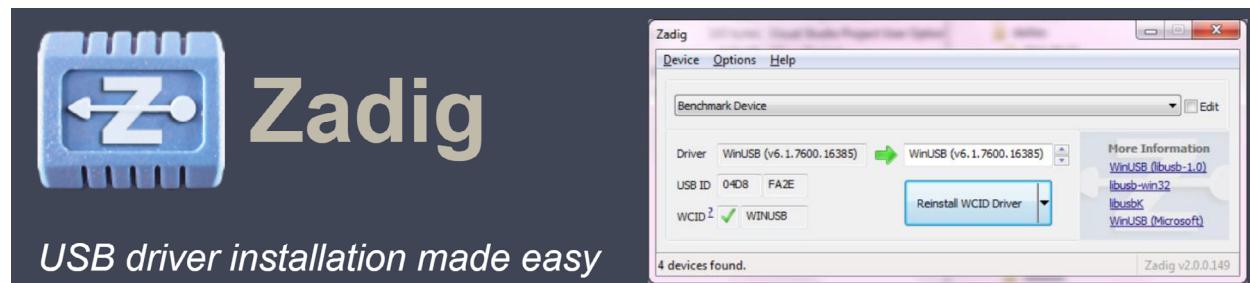


Figura 16 – Zadig

Figura16 #pacegover: imagem exibe o logotipo e a interface do programa **Zadig**, que facilita a instalação de drivers USB. À esquerda, há o logotipo em tons de azul, com um "Z" estilizado entre duas setas opostas, e a palavra "Zadig" em grande destaque ao lado, acompanhada pela frase "USB driver installation made easy". À direita, aparece uma janela do programa, mostrando uma lista de dispositivos USB e permitindo a seleção de drivers. O botão principal da janela é "Reinstall WCID Driver", destacando a funcionalidade do programa.

Fonte: autor

Para instalar o driver, é necessário primeiro conectar o Raspberry Pi Pico em modo BOOTSEL. Para isso: Desconecte a placa do PC e conecte o cabo usb da placa ao computador **enquanto** mantém pressionado o botão BOOTSEL e só então remova o dedo do botão.

• IMPORTANTE

- Lembre-se de conectar a placa em modo BOOTSEL toda vez que desejar alterar o código nela.
- A placa BitDogLab possui bateria, para entrar no modo BOOTSEL, é necessário desenergizar a placa, isso pode ser feito dando um clique duplo no botão de energizar ou simplesmente removendo a bateria.

Com o Raspberry Pi Pico conectado em modo BOOTSEL, execute o Zadig e navegue pelos dispositivos até o RP2 Boot (Interface 1), feito isso, selecione na caixa de seleção, a versão do drive ser instalado: WinUSB.

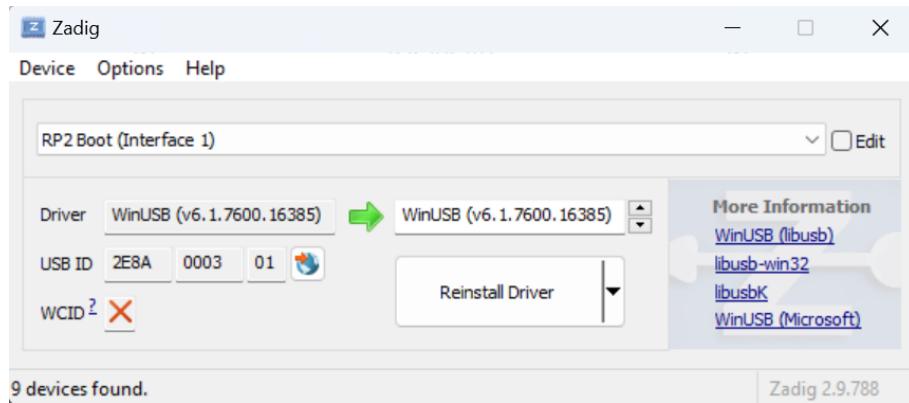


Figura 17 - Zadig

Figura17 #pracegover: imagem apresenta uma captura de tela de outra janela do Zadig. O dispositivo listado é "RP2 Boot (Interface 1)", e o driver selecionado é WinUSB (versão 6.1.7600.16385). Abaixo do driver, há informações detalhadas sobre o ID USB (2E8A, 0003, 01) e a opção de reinstalar o driver. O lado direito da janela fornece links para mais informações sobre os drivers, como WinUSB e libusb.

Fonte: autor

7º passo: Programando o Raspberry Pi Pico

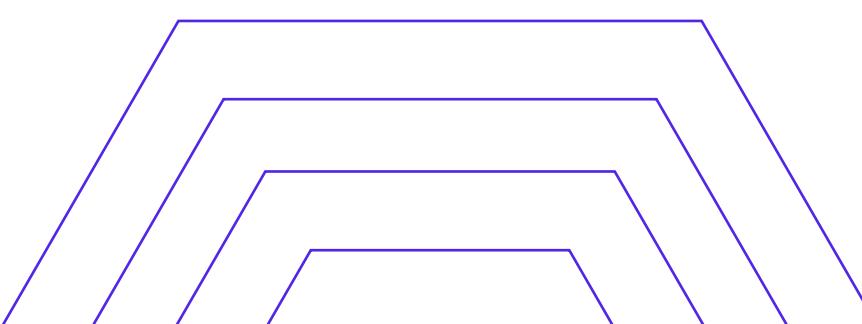
Após criar o projeto, volte ao menu do plugin Pi Pico e clique em Compile Project, isso irá gerar os arquivos de programação do microcontrolador. Para programar o microcontrolador RPi Pico clique em Run Project.

• IMPORTANTE

Caso a placa não esteja conectada em modo BOOTSEL, a placa não será encontrada.

Para conectar em modo BOOTSEL, desconecte a placa do PC e conecte o cabo usb da placa ao computador enquanto mantem pressionado o botão BOOTSEL e só então remova o dedo do botão.

Se o LED do Raspberry Pi Pico começar a piscar 2x por segundo, significa que sua configuração foi feita corretamente, **parabéns!**



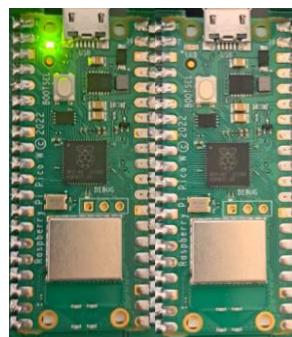


Figura 18 - Demonstração do LED do Raspberry Pi Pico

Figura18 #pracegover: Esta imagem mostra duas placas Raspberry Pi Pico lado a lado. Ambas estão conectadas e apresentam componentes eletrônicos visíveis, incluindo o conector micro-USB na parte superior de cada uma. A placa à esquerda tem um pequeno LED verde aceso, indicando que está ligada e funcional. A cor predominante das placas é verde, com componentes prateados e pretos.

Fonte: autor

Para desenvolver outros códigos, acesse o repositório de exemplos do Raspberry Pi Pico ([🔗 Link](#)) faça o download do mesmo e estude os códigos que utilizam periféricos e componentes que você deseje utilizar.

Quanto ao código básico do Blink, você poderá, por exemplo, alterar em que o LED pisca através da constante LED_DELAY_MS, presente na linha 16 do arquivo blink.c.

MONITOR SERIAL

1º passo: Instalar o Putty

Para realizar a comunicação via porta serial com o computador, é necessário que o mesmo possua um software apropriado para a recepção de dados. Recomenda-se o uso do Putty ([🔗 Link](#)), contudo, fica a seu critério utilizar outros softwares semelhantes.

2º passo: Configure o CMakeLists.txt

Para habilitar a comunicação serial, é necessário alterar o arquivo CMakeLists.txt, adicionando as seguintes linhas ao final do código: (obs: onde main é o nome do arquivo principal, no caso, main.h)

```
pico_enable_stdio_usb(main 1)
pico_enable_stdio_uart(main 0)
```

3º passo: Modificando o código principal:

No arquivo principal (nome_do_projeto.c), adicione a biblioteca stdio inserindo o seguinte trecho de código ao início do arquivo:

- #include <stdio.h>

Initialize as funções da biblioteca stdio executando a seguinte função loco após o início da função main():

- stdio_init_all();

4º passo: Configuração de Driver para monitor serial:

Utilizando o Zadig ([Link](#)), configure uma das interfaces do RP2 Boot (Interface 0) como USB Serial (CDC) mantendo uma como WinUSB para permitir a programação dele.

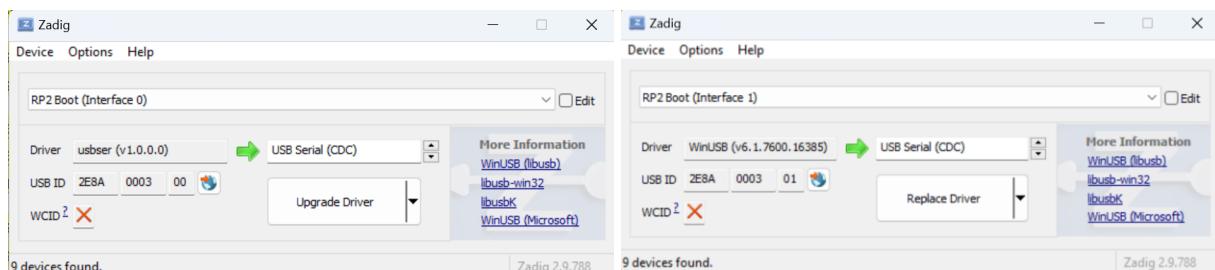


Figura 19 - Zadig

Figura19 #pracegover: imagem apresenta duas capturas de tela do Zadig lado a lado. Ambas as janelas listam o dispositivo "RP2 Boot", mas com diferentes interfaces: Interface 0 e Interface 1. Em uma das interfaces, o driver é **usbser** (versão 1.0.0.0), que será atualizado para **USB Serial(CDC)**. Na outra, o driver é **WinUSB**, também sendo alterado para **USB Serial(CDC)**. O USB ID é detalhado em ambas as janelas, assim como o WCID marcado com um "X" vermelho, indicando que não está presente.

Fonte: autor

5º passo: Realizando o Monitoramento serial com o Putty:

Através do Gerenciador de Dispositivos (clique com o botão direito no ícone Windows), identifique qual porta (COM) está se conectando ao Raspberry Pi Pico. Isso pode ser feito conectando-se e desconectando-se a placa para identificar qual COM aparece quando a placa é conectada.

Initialize o Putty, selecione comunicação Serial, defina a Serial line com a COM correta e Speed para 115200 e clique em Open para executar o monitor serial.

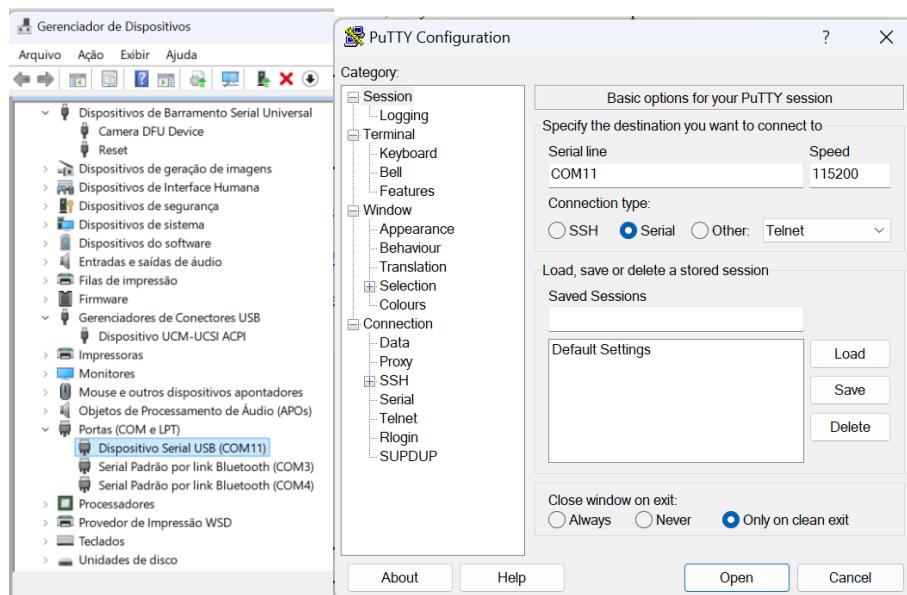


Figura 20 – Putty

Figura 20 #pracegover: A imagem exibe duas janelas lado a lado. À esquerda, temos o Gerenciador de Dispositivos do Windows, destacando a porta serial "Dispositivo Serial USB (COM11)" na lista de controladores. À direita, a janela de configuração do PuTTY, onde o usuário está configurando uma conexão serial para o COM11 com uma taxa de transmissão de 115200. O PuTTY é utilizado para comunicação em dispositivos serials, com várias opções de configuração exibidas na barra lateral da janela.

Fonte: autor

Todos os comandos de `printf('texto')` enviados pelo Raspberry serão exibidos nessa interface e podem ser utilizados para identificar a execução de partes de código ou mesmo imprimir valores de variáveis que se deseja observar.

Em resumo

Aprendemos como configurar o ambiente de desenvolvimento VS Code para compilar e programar projetos feitos para a placa de desenvolvimento do nosso curso.

Configuramos a comunicação através da porta serial virtual com o PC, ferramenta fundamental na depuração de códigos e na transferência de dados para o PC.

Não deixe para configurar seu PC depois! Faça o quanto antes e aproveite para explorar o código Blink e outros exemplos disponíveis no repositório do RPi Pico ([🔗 https://github.com/raspberrypi/pico-examples](https://github.com/raspberrypi/pico-examples))

Conclusão

Na próxima unidade, vamos começar a explorar periféricos internos da RPi Pico e da placa BitDogLab.

Aproveite esse tempo para começar a imaginar as possibilidades de projeto que você consegue implementar através da placa BitDogLab.

Muito obrigado e até a próxima!

Referências

GITHUB. Programando o Raspberry Pi Pico. Disponível em: <https://github.com/raspberrypi/pico-setup-windows/blob/master/docs/tutorial.md>. Acesso em: 3 out. 2024.

RASPBERRY PI FOUNDATION. Getting Started with Raspberry Pi Pico. Disponível em: <https://datasheets.raspberrypi.com/pico/getting-started-with-pico.pdf>. Acesso em: 3 out. 2024.

WASEEM, Shahzaib. How to Use VS Code for Raspberry Pi Pico and Pico-W. Medium, 2023. Disponível em: <https://medium.com/@shahzaib.waseem27/how-to-use-vs-code-for-raspberry-pi-pico-and-pico-w-c72e9692a238>. Acesso em: 3 out. 2024.

MAKER HERO. Programando a Raspberry Pi Pico no Visual Studio Code. MakerHero, 2022. Disponível em: <https://www.makerhero.com/blog/programando-a-raspberry-pi-pico-no-visual-studio-code/>. Acesso em: 3 out. 2024.

LEARN EMBEDDED SYSTEMS. USB Serial Communication with Raspberry Pi Pico. Disponível em: <https://learnembeddedsystems.co.uk/pico-usb-serial-code>. Acesso em: 3 out. 2024.

Obrigado

