Banco de Informações de Hardware (BIH) da BitDogLab

A BitDogLab, uma iniciativa do Projeto Escola 4.0 da Unicamp, é uma ferramenta educacional dedicada à eletrônica e computação. Baseada na Raspberry Pi Pico H ou W, permite aos usuários explorar, montar e programar utilizando componentes montados na sua placa e também externos conectados de forma organizada e segura. Selecionados meticulosamente, os componentes promovem um aprendizado mão na massa, incentivando os usuários a aprimorar habilidades de programação e eletrônica de maneira sinérgica e progressiva.

Esta plataforma enriquecedora oferece uma experiência vibrante, imergindo os usuários em um ambiente de aprendizado colorido, auditivo e sinestésico. Além disso, a BitDogLab é otimizada para programação assistida por modelos de linguagem de grande escala (LLM), como o GPT-4, facilitando um aprendizado mais intuitivo e assistido por um incansável tutor.

Orientada à educação pré-universitária, a BitDogLab visa catalisar a incorporação tecnológica educacional, fornecendo uma ferramenta robusta e flexível que se integra de maneira única à jornada de aprendizado dos estudantes.

Um diferencial da BitDogLab é que seu projeto é totalmente aberto, permitindo que seja livremente copiada, fabricada, montada e melhorada pelos usuários. Mais informações em: https://github.com/Fruett/BitDogLab

Conexões e Configurações de Hardware:

As conexões da Raspberry Pi pico com outros componentes estão realizadas da seguinte forma:

Um **LED RGB** catodo comum tem o eletrodo do vermelho lidado no GPIO 12 através de um resistor de 220 ohm, o pino de verde está ligado no GPIO 13 também através de um resistor de 220 ohm e o pino do azul no GPIO 11 através de um resistor de 150 ohm.

Um botão, identificado como **Botão A**, está conectado no GPIO5 da Raspberry Pi Pico. O outro terminal do botão está conectado ao GND da placa.

Outro botão, identificado como **Botão B**, está conectado no GPIO6 da Raspberry pi pico. O outro terminal do botão também está conectado ao GND da placa.

Outro botão, identificado como **RESET**, está conectado no RUN (pino número 30) da Raspberry pi pico. O outro terminal do botão também está conectado ao GND da placa.

Um buzzer passivo, identificado como **Buzzer A**, está conectado - através de um transistor - no GPIO21 da Raspberry pi pico.

Outro buzzer passivo, identificado como **Buzzer B**, está conectado no GPIO4 da Raspberry pi pico.

O pino in de uma **matriz de LEDs 5050** RGB de 5 linhas por 5 colunas tipo WS2812B (Neopixel) está conectada ao GPIO7.

Um **joystick analógico** tipo KY023 tem a saída VRy conectada ao GPIO26 e a saída VRx ao GPIO27. Seu botão SW está conectada ao GPIO22, o outro terminal do botão está no GND.

Um **display OLED** 128 colunas por x 64 linhas de 0,96 polegadas com comunicação I2C, tem seu pino SDA conectado ao GPIO14 e o pino SCL com o GPIO15. Esse display exige o carregamento da biblioteca ssd1306.py na Raspiberry pi pico.

Um módulo **microfone** de eletreto com saída analógica está conectado ao GP28. O nível médio do sinal de saída é 1,65 V. A tensão na saída do módulo varia de 0V até 3,3V

Um conector Insulation-Displacement Connector (**ICD**) box de 14 pinos é usado para expanção de hardware e está assim conectado com a Raspberry Pi Pico: pino 1 com o GND, pino 2 com o 5V, pino 3 com 3V3, pino 4 com GPIO10, pino 5 com o GPIO28, pino 6 com o GPIO9, pino 7 com GND analogico. pino 8 com o GPIO8, pino 9 com o GPIO17, pino 10 com o GND, pino 11 com o GPIO16, pino 12 com GPIO19, pino 13 com o GND, pino 14 com o GPIO18.

Uma barra de terminais que permite a conexão de garras do tipo jacaré está assim conectada ao Raspberry Pi Pico: os eletrodos DIG 0, 1, 2 e 3 estão conectados aos GPIOs 3, 2, 1 e 0, respectivamente. Além disso esta barra de terminais possui mais 5 eletrodos conectados ao: GND analógico, GPIO 28, GND, 3V3 e 5V da Raspberry Pi Pico.

As orientações a seguir são úteis no caso do usuário optar por programar em Micropython.

Sugerimos que as seguintes bibliotecas sejam importadas conforme a necessidade do usuário:

```
from machine import PWM, Pin

import neopixel

import time

import random

from machine import Pin, SoftI2C, ADC

from ssd1306 import SSD1306_I2C

import math
```

```
# Configuração do OLED

i2c = SoftI2C(scl=Pin(15), sda=Pin(14))

oled = SSD1306_I2C(128, 64, i2c)
```

```
# Número de LEDs na sua matriz 5x5
NUM_LEDS = 25
# Inicializar a matriz de NeoPixels no GPIO7
np = neopixel.NeoPixel(Pin(7), NUM_LEDS)
# Definindo a matriz de LEDs
LED_MATRIX = [
    [24, 23, 22, 21, 20],
    [15, 16, 17, 18, 19],
    [14, 13, 12, 11, 10],
    [05, 06, 07, 08, 09],
    [04, 03, 02, 01, 00]
]
```