KAMAMI

KAmodRPi Pico Prototyping Platform

KAMAMI

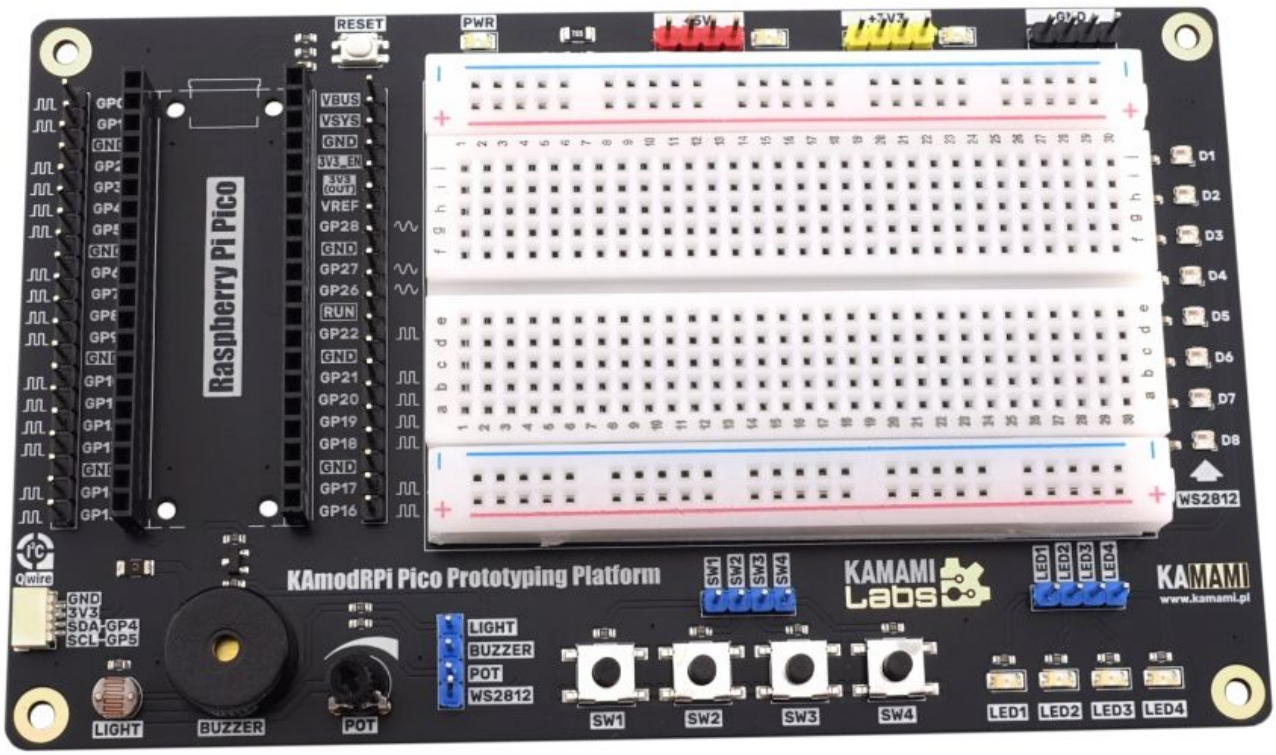
Rev. 20260420064622
Źródło: https://wiki.kamamilabs.com/index.php?title=KAmodRPi_Pico_Prototyping_Platform

Table of contents

Description

KAmodRPi Pico Prototyping Platform - Universal Prototyping System for Raspberry Pi Pico

The KAmodRPi Pico Prototyping Platform is a multifunctional expansion board for the Raspberry Pi Pico series modules, designed to streamline prototyping processes. The device integrates the most commonly used electronic components with a breadboard area, creating an organized workspace. Thanks to the broken-out GPIO connectors and built-in peripherals, users can instantly build and test circuits without the need to prepare basic input/output elements every time. With a dedicated reset button and an I²C Qwire connector, working with modern sensors becomes seamless and intuitive. It is a refined prototyping system that combines operational safety, provided by intelligent power management, with full freedom to expand with external components.



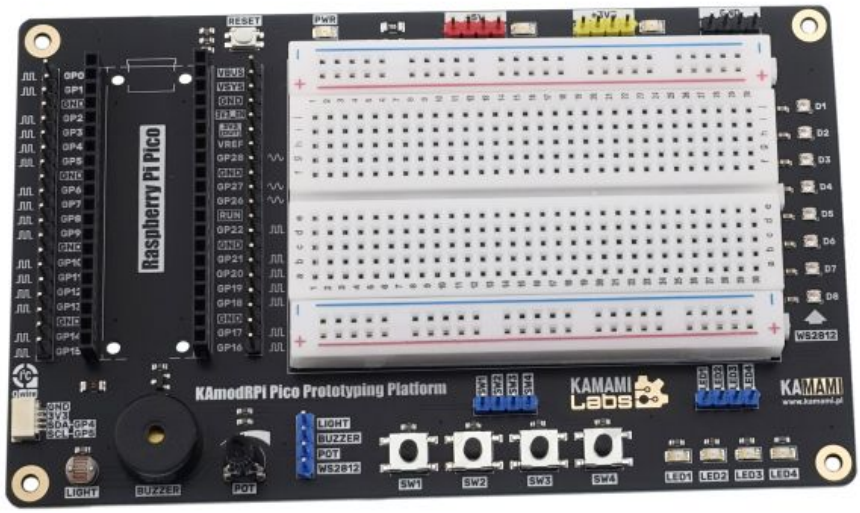
Basic Parameters

- Standard 2×20 pin female header for Raspberry Pi Pico, Pico W, Pico 2
- 20-pin male headers with all RPi Pico signals broken out and labeled (2.54 mm pitch)
- I²C Qwire connector (JST SH 4-pin 1 mm), compatible with Qwiic / STEMMA QT
- 400-point prototyping breadboard
- 4 red LEDs
- 8 addressable WS2812 RGB LEDs
- 4 monostable push-buttons
- Built-in potentiometer and photoresistor for ADC testing
- Integrated piezoelectric transducer for acoustic signaling
- Built-in RESET button connected to the microcontroller's RUN pin
- Built-in peripherals can be disconnected using the included jumper wires, allowing full use of GPIO pins for other purposes
- LED indicators for each power line (PWR, 5V, 3.3V)
- Supply voltage: 5V (from the Raspberry Pi Pico USB port)
- Integrated polymer fuses on power lines for the Qwire connector and +5V / +3V3 goldpins

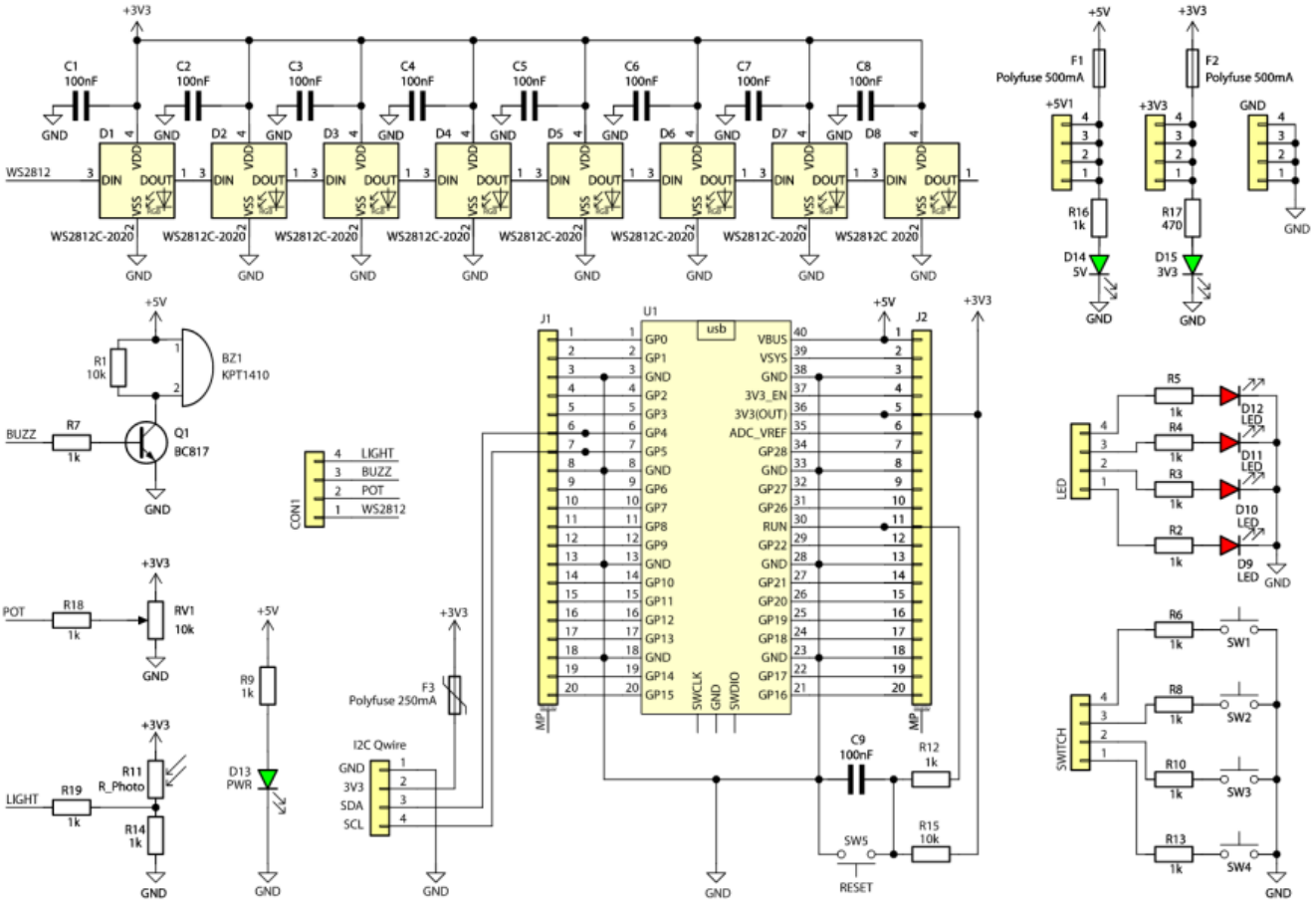
- Dimensions: 85 × 140 mm

Standard Equipment

Code	Description
KAmodRPi Pico Prototyping Platform	• Assembled and tested module • 5 x rubber feet
Prototyping Breadboard	• 400-point breadboard, dimensions 82x55x10 mm
Jumper Wires	• Set of 40 female-to-female jumper wires, various colors, 17 cm length
Symmetrical 2.54 mm connectors	• 1×40 goldpin header, male-to-male, symmetrical – ideal for creating connections on breadboards.



Electrical Schematic

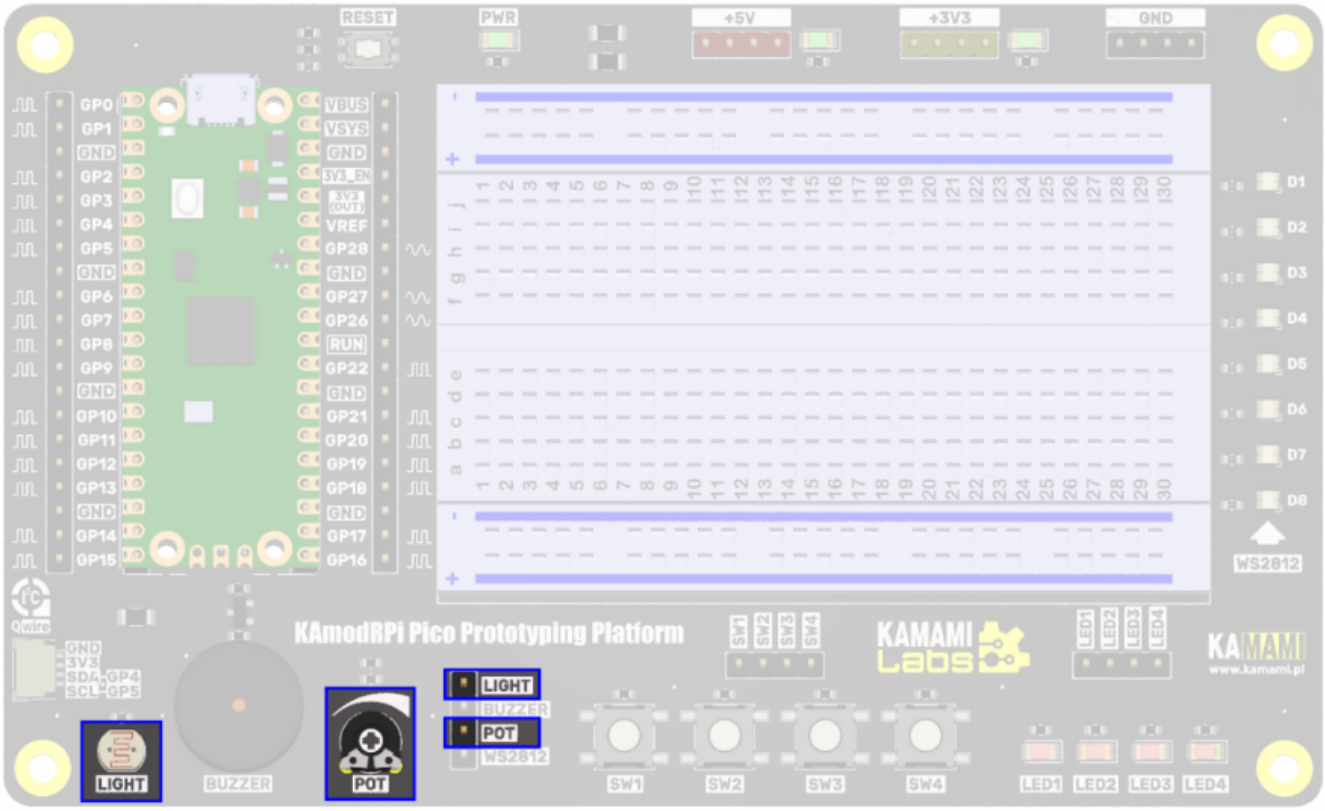


Potentiometer and Photoresistor

The board features two integrated analog signal sources for learning and testing ADC (Analog-to-Digital Converter) functions.

- **Potentiometer (POT)**: Allows manual voltage adjustment from 0 to 3.3V. Ideal for simulating sensor readings (e.g., temperature), adjusting LED brightness, or controlling servos.
- **Photoresistor (LIGHT)**: Changes its resistance based on the light intensity falling on the sensor. Allows for creating automatic light control systems or day/night detection.

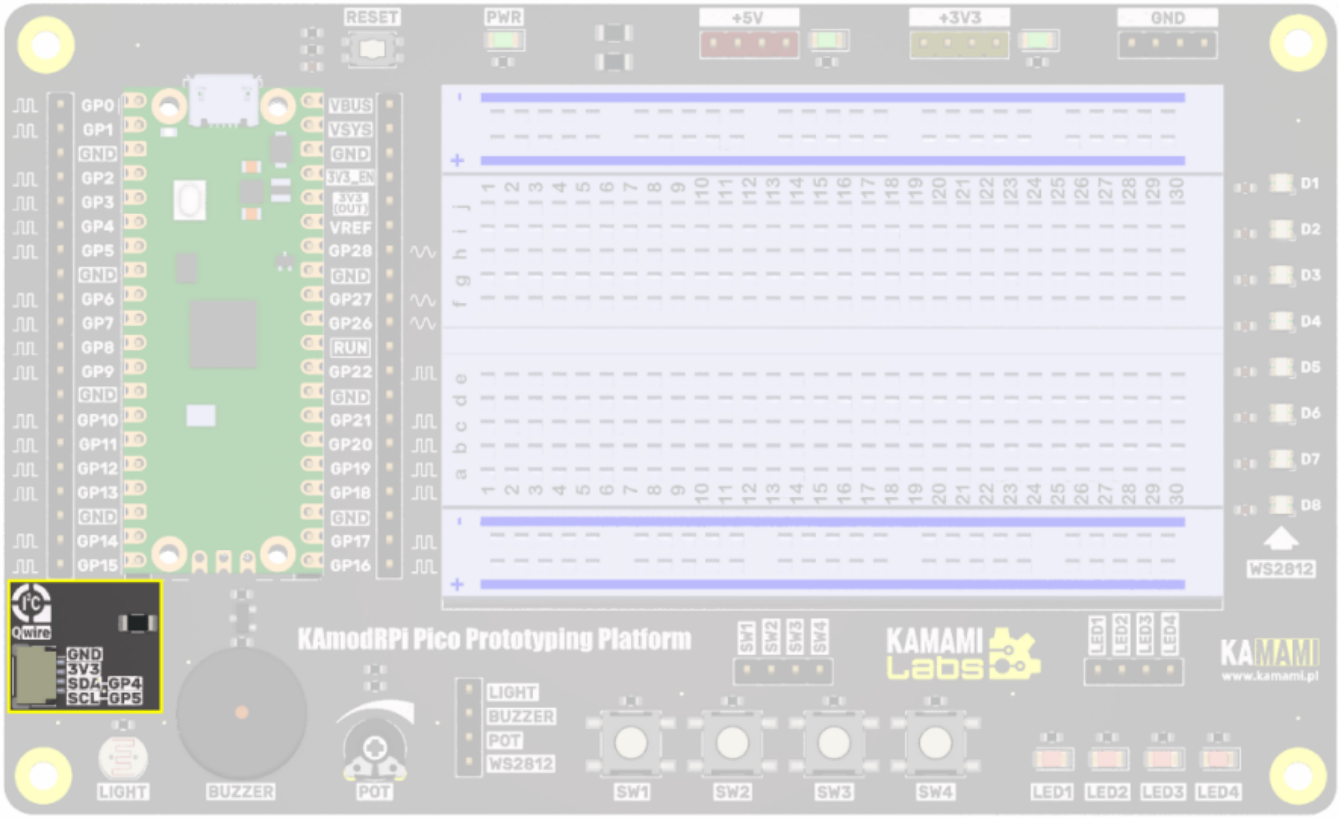
Note: These signals should be connected to Pico pins that support ADC functionality (standard GP26 and GP27). ADC pins are highlighted on the board with a wave symbol ∿ for quick identification.



I²C Interface

The I²C interface is broken out to a Qwire connector compatible with Qwiic / STEMMA QT. The Qwiic connector is designed for rapid prototyping with external modules. Additionally, the power line of this connector has an independent polymer fuse with a 250mA trip threshold.

- Standard: JST SH 4-pin 1mm connector compatible with the KAmod Qwire, SparkFun Qwiic, and Adafruit STEMMA QT ecosystems.
- Communication: Uses the I²C bus (SDA-GP4, SCL-GP5 lines) and 3.3V power.
- Advantage: Allows daisy-chaining multiple sensors, OLED displays, or motor controllers using a single standard cable.

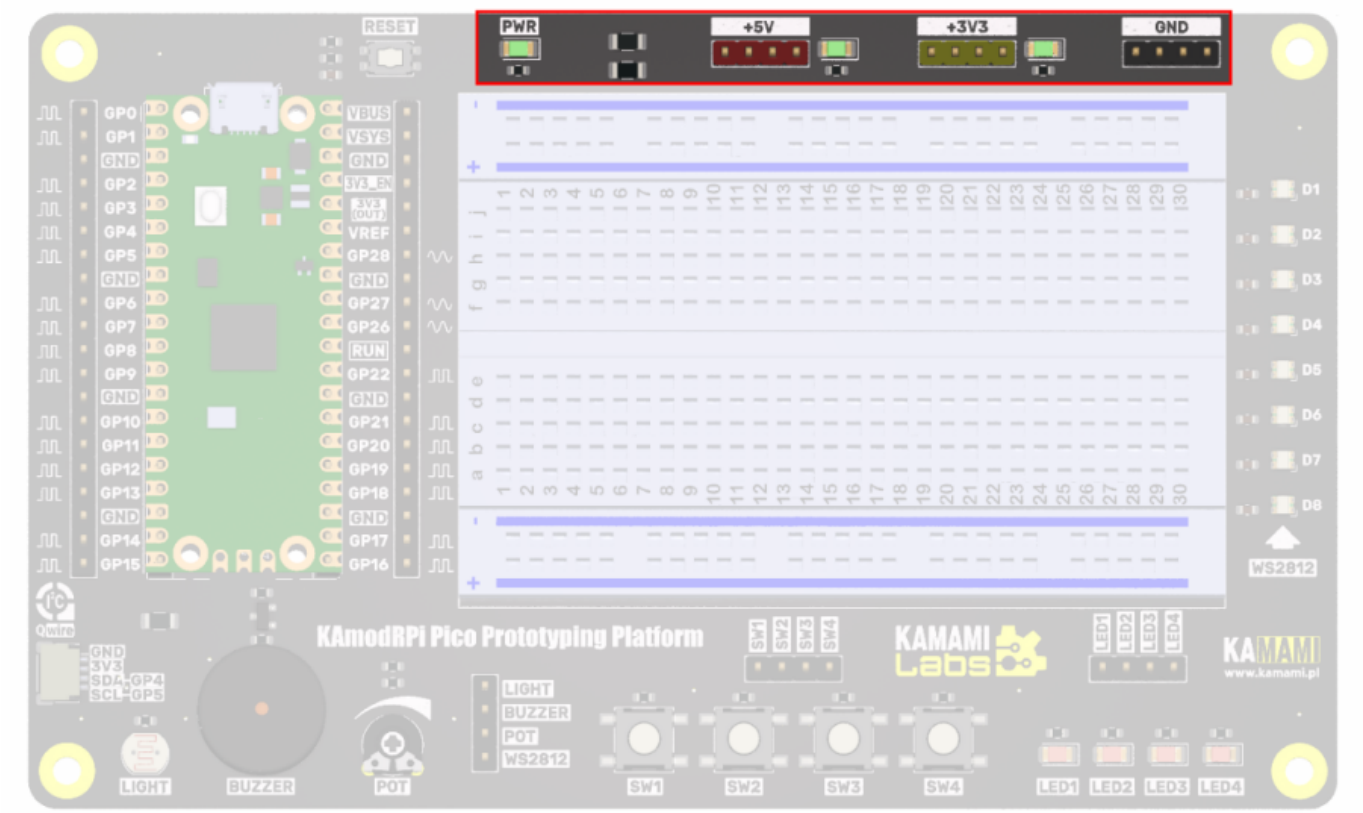


Power Supply

The KAmodRPi Pico Prototyping Platform is powered by 5V from the Raspberry Pi Pico's USB port. The platform is equipped with a voltage monitoring system and damage protection, making it a safe tool for learning and experimentation.

- **Visual Voltage Control:** Three dedicated LEDs indicate the status of key power lines:
  - **PWR**: Signals the presence of voltage directly from the USB port (5V before protection).
  - **+5V**: Indicates proper power to the main 5V rail (5V after protection).
  - **+3V3**: Confirms stable operation of the microcontroller's voltage regulator (3.3V after protection).
- **Fuse Protection:** Power lines broken out to the +5V and +3V3 goldpins are protected by polymer fuses. In the event of a short circuit or overload on the breadboard, the fuse automatically cuts off the current. These fuses protect both the base board and the connected Raspberry Pi Pico from the effects of accidental shorts or excessive current consumption in the prototype. Once the cause of the failure is removed, the element regenerates itself, restoring power without the need to replace any parts.

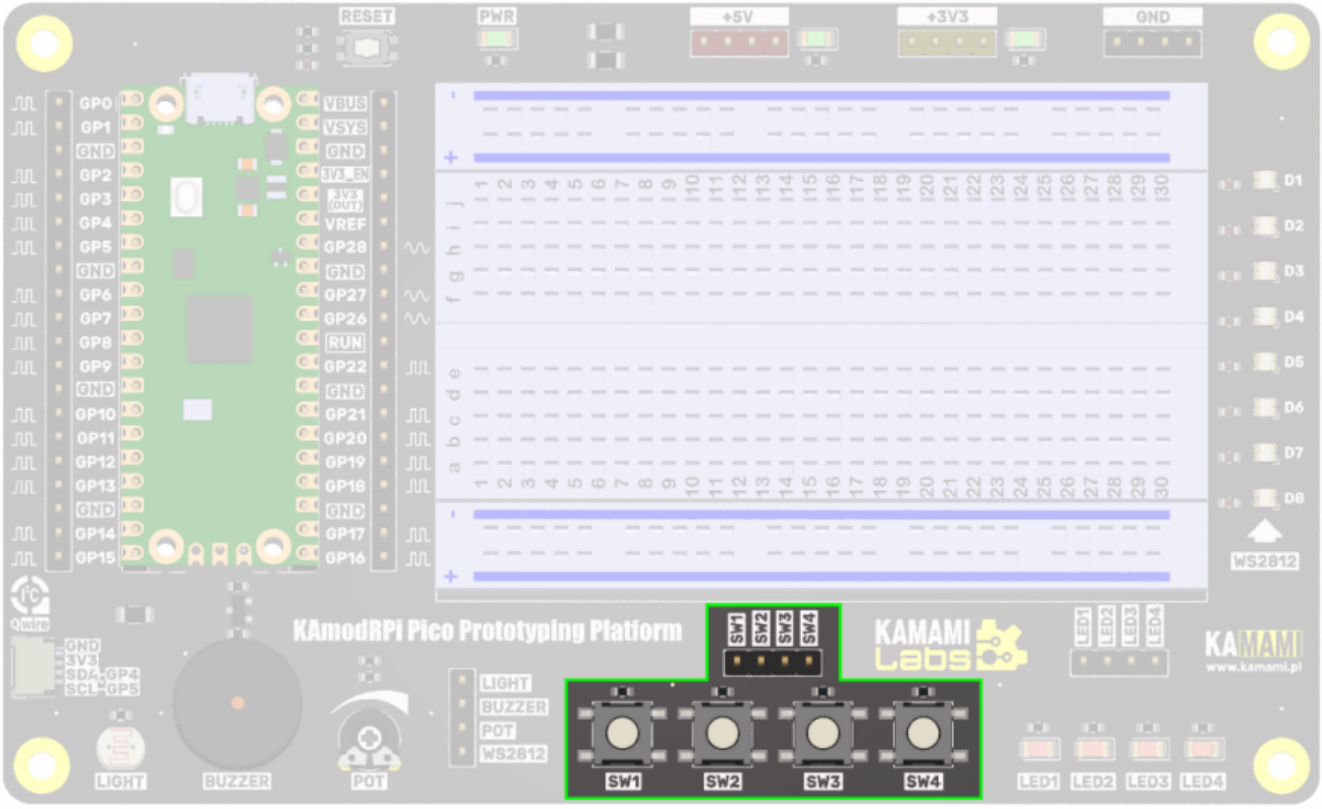
Important: The power rails on the goldpin headers have protection up to 500 mA. If your project requires more current (e.g., multiple powerful servos or long RGB LED strips), we recommend using an external power source to avoid tripping the fuses and temporarily disconnecting peripherals.



Buttons

The KAmodRPi Pico Prototyping Platform module is equipped with four monostable push-buttons (SW1 – SW4) of the microswitch type for user interaction.

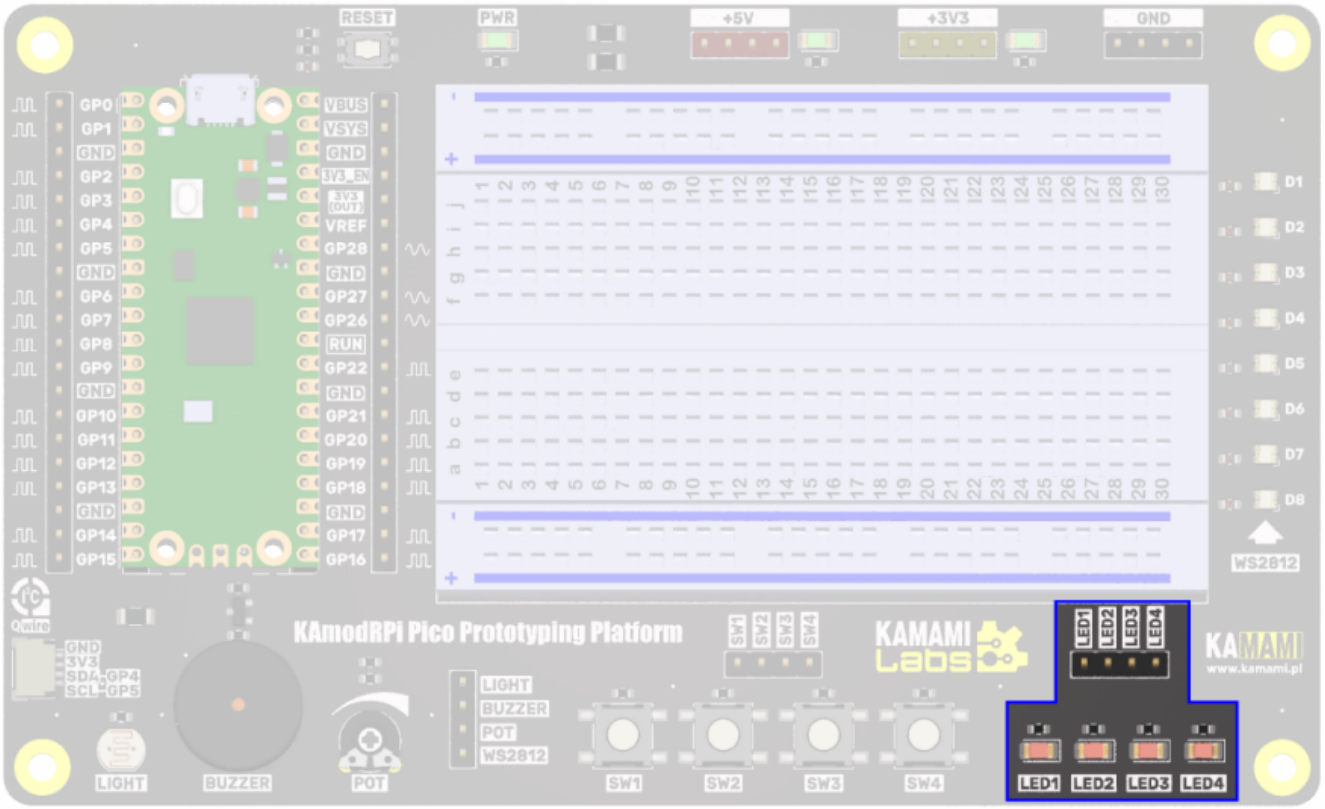
- Operation: The buttons connect the GPIO pins to ground (GND).
- Programming: In the program code, pins should be configured as inputs with the internal pull-up resistor enabled. A low state (0) means the button is pressed, and a high state (1) means it is idle.
- Application: Creating control interfaces, switching operating modes, or triggering events.



LEDs

The board features four red LEDs (LED1 – LED4) located next to the buttons, facilitating the creation of intuitive visual interfaces.

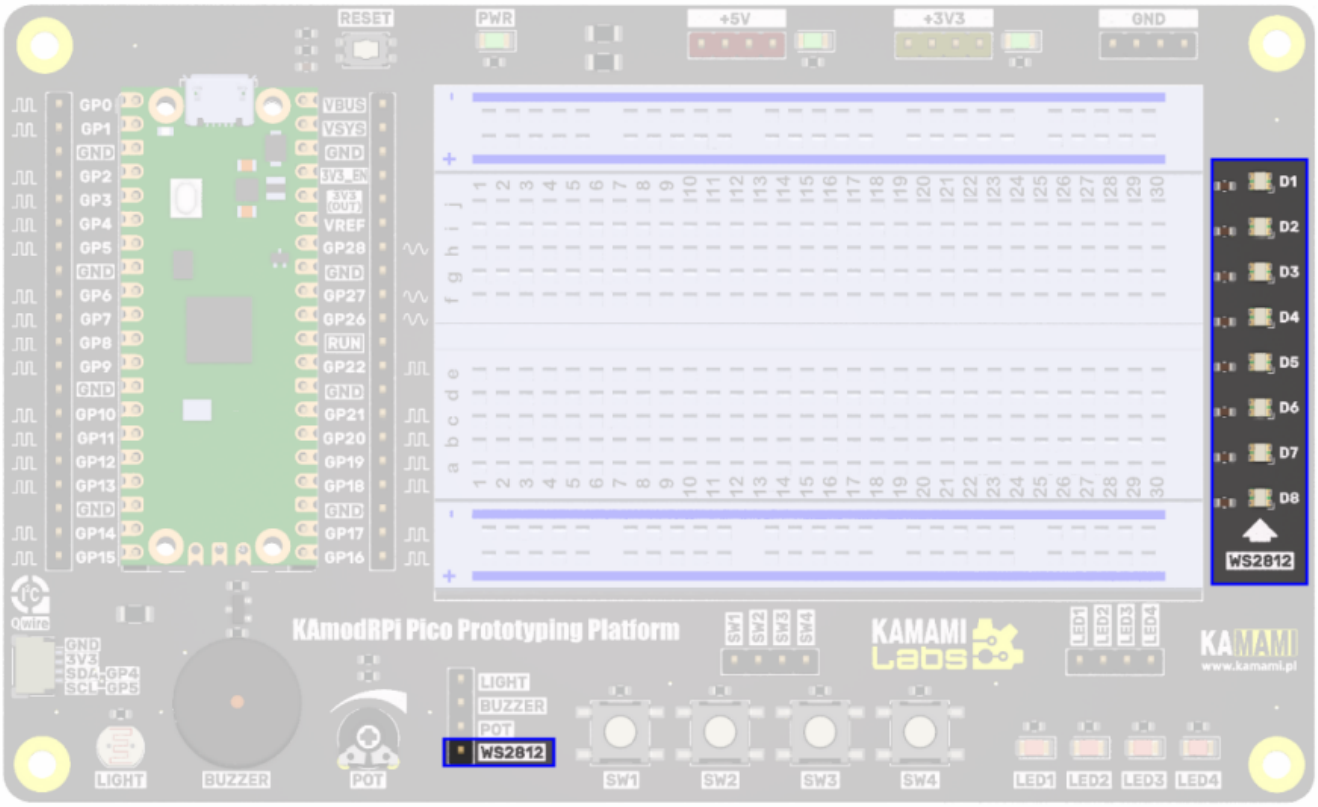
- Operation: LEDs are controlled by a digital signal. An LED turns on when a high state (1) is applied to the corresponding GPIO pin.
- Configuration: Each LED has an integrated current-limiting resistor, allowing direct connection to microcontroller pins without risk of damage.
- Application: Indicating operating states, debugging code, and informing about device activity.



WS2812 LEDs

The board is equipped with a strip of eight addressable WS2812 LEDs (Neopixel standard), allowing for the display of any colors from the RGB palette.

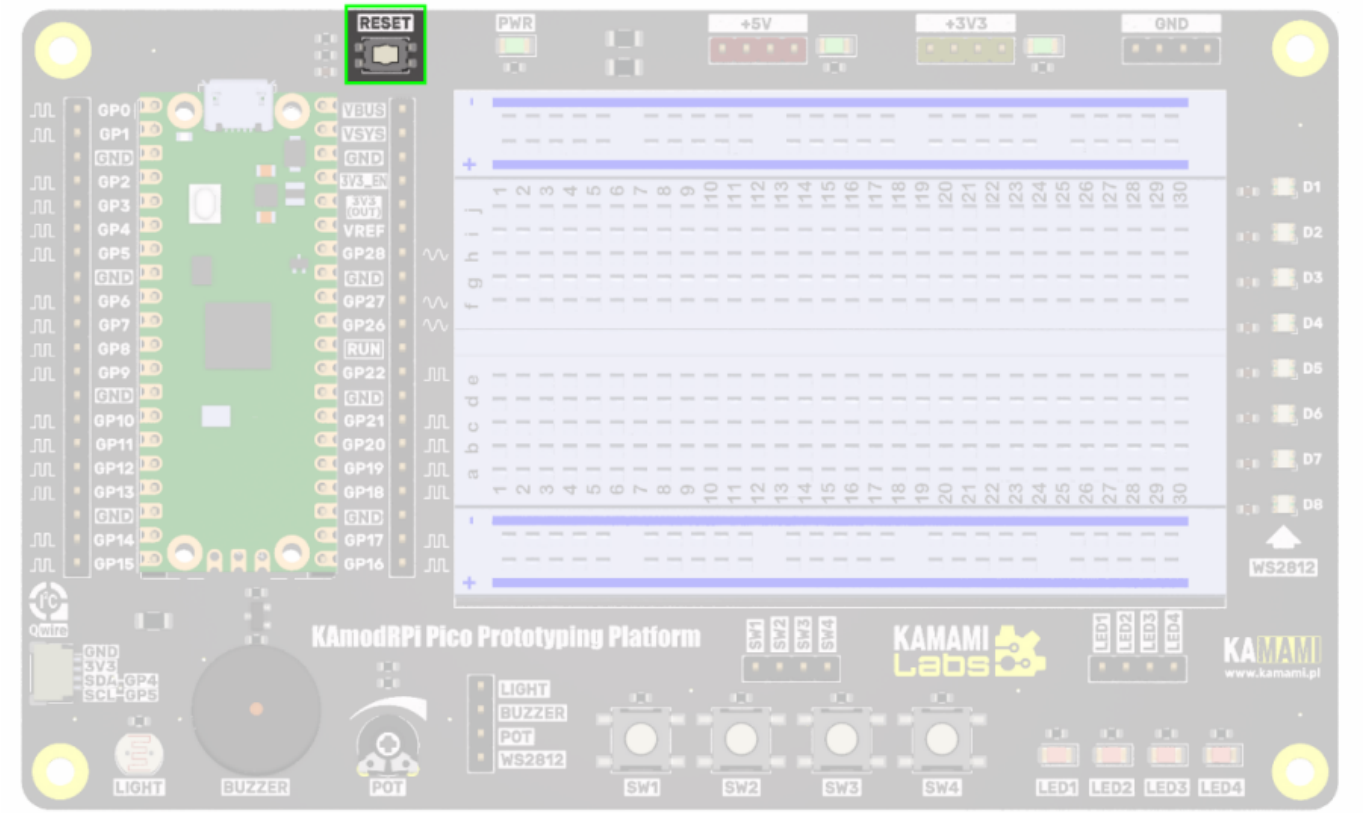
- Operation: All 8 LEDs are controlled serially using only one GPIO pin. Each LED can light up independently with a different color and brightness.
- Programming: A library supporting the one-wire protocol is required (e.g., Adafruit NeoPixel for Arduino or the built-in neopixel module in MicroPython).
- Application: Creating striking illuminations, visualizing sensor data (e.g., changing color based on temperature), or signaling system status.



RESET Button

The dedicated RESET button is a convenience missing from the standard Raspberry Pi Pico module.

- Operation: The button is connected to the microcontroller's RUN pin and ground (GND). Pressing it causes an immediate processor interrupt and program restart.
- Advantage: Eliminates the need for frequent unplugging and plugging of the USB cable to restart the system or enter firmware upload mode (when used in combination with the BOOTSEL button on the Pico module).
- Safety: Allows for a quick stop of the device in case of prototype malfunction.



Example Program

A test program has been prepared for both the Arduino IDE and MicroPython (simplified version) to comprehensively test all functions of the KAmodRPi Pico Prototyping Platform. It demonstrates interaction between analog sensors and output elements (LED, Buzzer, RGB).
Arduino Requirements:

- Board Manager: `Raspberry Pi Pico/RP2040/RP2350`
- Libraries: `Adafruit NeoPixel`

MicroPython Requirements:

- The script uses standard MicroPython libraries (`machine, neopixel`)

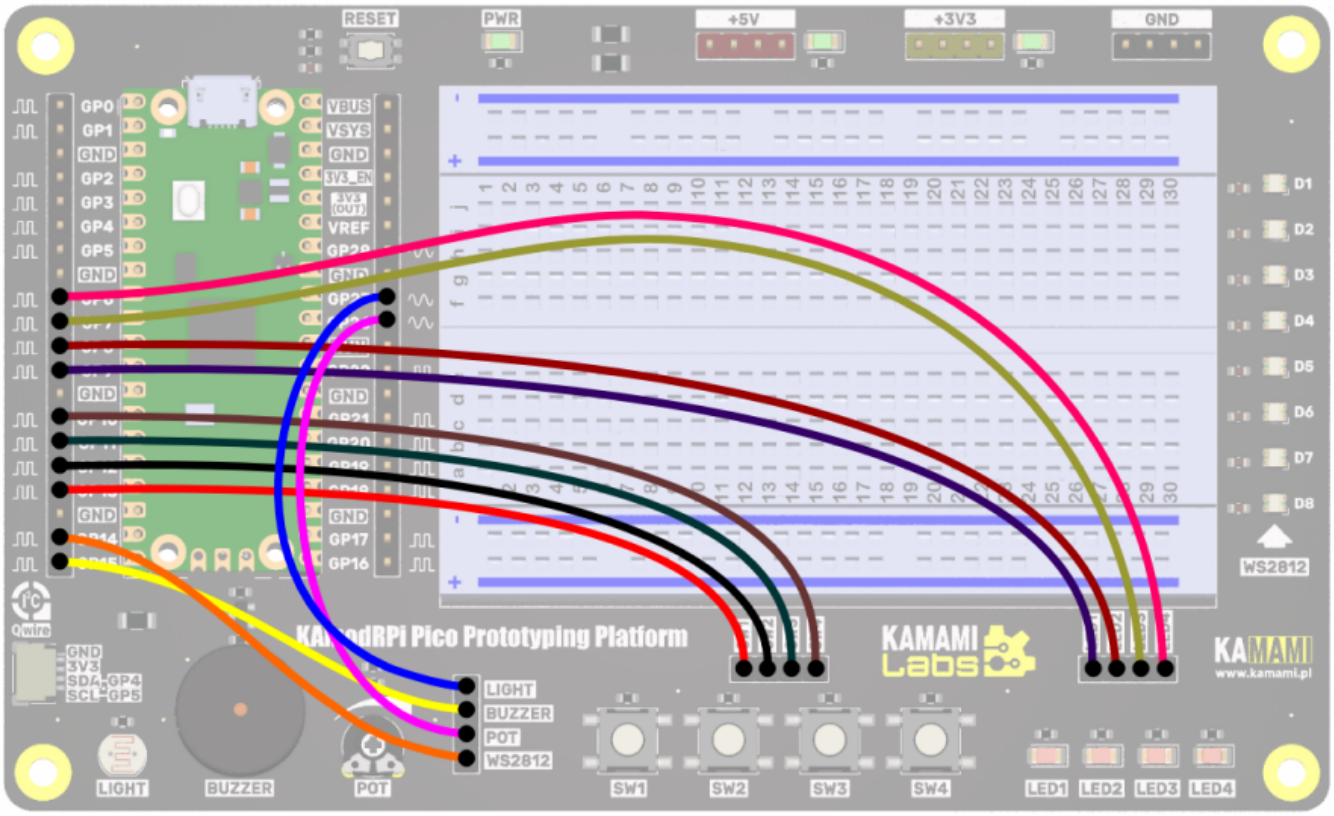
Program Operation:

1. Start Sequence
   • Upon startup and serial port initialization, the program performs a characteristic sound sequence using the buzzer, signaling readiness.
2. Intelligent LED Control (Digital Section)
   • Automatic Mode: Red LEDs perform a smooth "knight rider" wandering effect. Animation speed is dynamically adjusted by the photoresistor – the brighter it is, the faster the LEDs change state.
   • Manual Mode: Pressing any of the 4 buttons pauses the animation, and the LEDs respond directly to the button states.
3. Interactive Buzzer
   • Each of the four buttons is assigned to a different sound frequency (from 500 Hz to 2000 Hz). Pressing a button generates a unique tone.
4. RGB LED Strip (WS2812)
   • The program cyclically fills the 8-LED strip with red, green, and blue colors.

- The fill speed and color changes are controlled in real-time via the built-in potentiometer.

5. Serial Monitor
   - The program sends a KAmodRPi Pico Prototyping Platform greeting to the serial port (115200 bps) at each restart to verify the computer connection.
6. GPIO Signal Mapping

Component	Signal Type	GPIO
Buttons (SW1-SW4)	Digital (PULL_UP)	GP10, GP11, GP12, GP13
LEDs (LED1-LED4)	Digital (High = ON)	GP6, GP7, GP8, GP9
WS2812 (RGB)	One-Wire (Serial)	GP14
Buzzer	PWM	GP15
Potentiometer (POT)	Analog (ADC)	GP26
Photoresistor (LIGHT)	Analog (ADC)	GP27
I²C (Qwire)	I²C	GP4 (SDA), GP5 (SCL), 3V3 protected 250 mA



Arduino Test Program:

```
//Board manager URL:
https://github.com/earlephilhower/arduino-pico/releases/download/global/package_rp2040_index.j
son
//Library required: Adafruit NeoPixel

#include <Adafruit_NeoPixel.h>

//---------------------------------
#define LED1    6
#define LED2    7
#define LED3    8
#define LED4    9
```

```
#define LED_ON      1
#define LED_OFF      0
#define LED_ALL     4

#define SW1     10
#define SW2     11
#define SW3     12
#define SW4     13
#define SW_ON        0
#define SW_OFF       1
#define SW_ALL       4

#define POT_PIN     26
#define LIGHT_PIN   27

#define WS2812_PIN  14
#define NUMPIXELS   8
#define BUZZ_PIN    15

//--------------------------------
int led_sw_man(int delay);
int ws_man(int delay);
void buzz_show(int d);

int buttons;
int adj;
int buzz_time;

Adafruit_NeoPixel pixels(NUMPIXELS, WS2812_PIN, NEO_GRB + NEO_KHZ800);

void setup() {
  Serial.begin(115200);
  delay(2000);
  Serial.println("KAmodRPi Pico Prototyping Platform");

  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(LED3, OUTPUT);
  pinMode(LED4, OUTPUT);

  pinMode(SW1, INPUT_PULLUP);
  pinMode(SW2, INPUT_PULLUP);
  pinMode(SW3, INPUT_PULLUP);
  pinMode(SW4, INPUT_PULLUP);

  pinMode(BUZZ_PIN, OUTPUT);

  pixels.begin();
  pixels.show();
  buzz_show(200);
}

void loop() {
  adj = analogRead(LIGHT_PIN);
  buttons = led_sw_man(adj/16);

  if (buttons > 0) {
    buzz_time = 10;
    if (buttons == 1) tone(BUZZ_PIN, 500);
```

```
    if (buttons == 2) tone(BUZZ_PIN, 1000);
    if (buttons == 4) tone(BUZZ_PIN, 1500);
    if (buttons == 8) tone(BUZZ_PIN, 2000);
  }
  if (buzz_time > 0){
      buzz_time--;
      if (buzz_time == 0) noTone(BUZZ_PIN);
  }

  adj = analogRead(POT_PIN);
  ws_man(adj/16);

  delay(10);
}

int ws_man(int delay){
  static int ws_i = 0;
  static int ws_led = 0;
  static int ws_color = 0;

  if (ws_i < delay){
    ws_i++;
  } else {
    ws_i=0;
    for (uint l = 0; l<NUMPIXELS; l++){
      if (ws_led >= l){
        if (ws_color == 0) pixels.setPixelColor(l, 255, 0, 0);
        if (ws_color == 1) pixels.setPixelColor(l, 0, 255, 0);
        if (ws_color == 2) pixels.setPixelColor(l, 0, 0, 255);
      }
    }
    pixels.show();
    ws_led++;
    if (ws_led >= NUMPIXELS) {
      ws_led = 0;
      ws_color++;
      if (ws_color > 2) ws_color = 0;
    }
  }
  return 1;
}

int led_sw_man(int delay){
  static int led_i = 0;
  static int led_pos = 0;
  static int led_dir = 0;
  static int led_map[LED_ALL] = {LED1, LED2, LED3, LED4};
  int push_map = 0;
  static int sw_map[SW_ALL] = {SW1, SW2, SW3, SW4};

  for (int i=0; i<SW_ALL; i++){
    if (digitalRead(sw_map[i]) == SW_ON){
      push_map |= (1<<i);
    }
  }

  if (push_map > 0){
    for (int i=0; i<SW_ALL; i++){
      digitalWrite(led_map[i], (push_map & (1<<i)) ? LED_ON : LED_OFF);
```

```
    }
  } else {
    if (led_i < delay){
      led_i++;
    } else {
      led_i=0;
      digitalWrite(led_map[led_pos], LED_OFF);
      if (led_dir == 0){
        led_pos++;
        if (led_pos >= LED_ALL){
          led_pos = LED_ALL - 2;
          led_dir = 1;
        }
      } else {
        led_pos--;
        if (led_pos < 0){
          led_pos = 1;
          led_dir = 0;
        }
      }
      digitalWrite(led_map[led_pos], LED_ON);
    }
  }
  return push_map;
}

void buzz_show(int d){
  tone(BUZZ_PIN, 1000); delay(d); noTone(BUZZ_PIN); delay(d);
  tone(BUZZ_PIN, 1000); delay(d); noTone(BUZZ_PIN); delay(d);
  tone(BUZZ_PIN, 1000); delay(d/4); noTone(BUZZ_PIN); delay(d/4);
  tone(BUZZ_PIN, 1000); delay(d); noTone(BUZZ_PIN); delay(d);
}
```

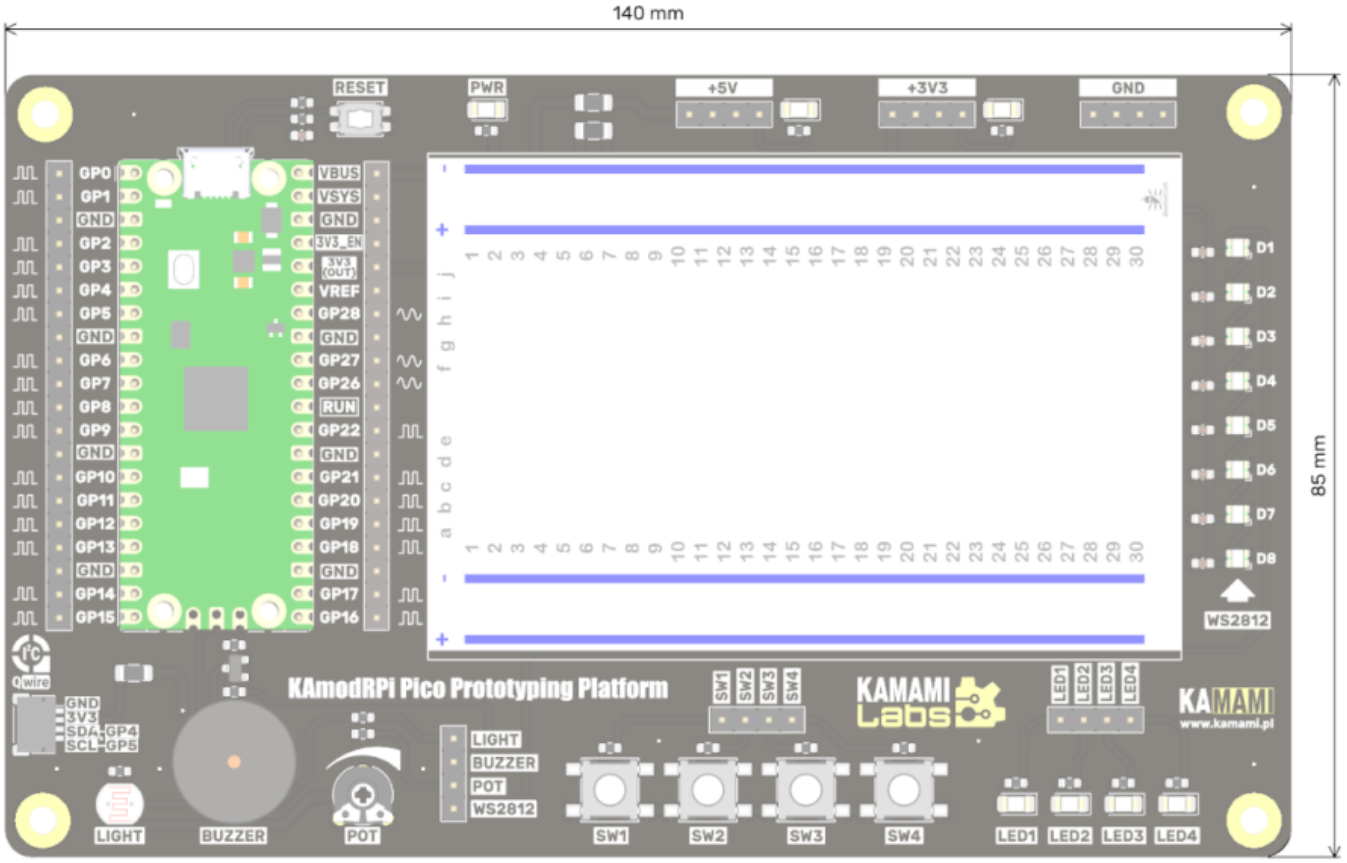
Simplified MicroPython Test Program:

```
import machine
import utime
import neopixel

# --- PIN Configuration ---
led_pins = [6, 7, 8, 9]
leds = [machine.Pin(p, machine.Pin.OUT) for p in led_pins]

sw_pins = [10, 11, 12, 13]
buttons = [machine.Pin(p, machine.Pin.IN, machine.Pin.PULL_UP) for p in sw_pins]

pot = machine.ADC(26)
ldr = machine.ADC(27)

buzzer = machine.PWM(machine.Pin(15))
pixels = neopixel.NeoPixel(machine.Pin(14), 8)

def play_tone(freq, duration):
    if freq > 0:
        buzzer.freq(freq)
        buzzer.duty_u16(32768) # 50% duty cycle
```

```
    utime.sleep_ms(duration)
    buzzer.duty_u16(0)

print("RPi Pico Prototyping Platform - Start Test")
play_tone(1000, 100)

while True:
    light_val = ldr.read_u16()
    speed = max(50, light_val // 100)
    for i in range(4):
        if buttons[i].value() == 0:
            leds[i].value(1)
            buzzer.freq(500 * (i + 1))
            buzzer.duty_u16(1000)
        else:
            leds[i].value(0)
    if all(b.value() == 1 for b in buttons):
        buzzer.duty_u16(0)

    pot_val = pot.read_u16()
    brightness = pot_val // 256
    for i in range(8):
        pixels[i] = (brightness, 0, 255 - brightness)
    pixels.write()

    utime.sleep_ms(10)
```

Dimensions

The dimensions of the KAmodRPi Pico Prototyping Platform module are 85 x 140 mm. The board includes 4 mounting holes with a 3 mm diameter for stable attachment to a case or workbench.



Links

- CAD Model (STEP)

BTC Korporacja
05-120 Legionowo
ul. Lwowska 5
tel.: (22) 767-36-20
faks: (22) 767-36-33
e-mail:
sprzedaz@kamami.pl
https://kamami.pl