Meet the ADIY PICO

The ADIY PICO is a small yet powerful microcontroller board based on the RP2040 chip. It's designed for learning, experimenting, and building projects — from simple LED blinkers to advanced IoT devices. With multiple pins, built-in memory, and USB support, the Pico is the "brain" that will control all the sensors, modules, and displays in your kit. Before you can start coding with the Arduino IDE, read the following instructions below

How it works with Arduino IDE:

First, you install the Raspberry Pi Pico boards package inside Arduino IDE.
When you upload a sketch, Arduino automatically converts (compiles) it to binary and then sends it to the Pico.
The Pico doesn't need a special .uf2 firmware in this case — Arduino handles everything.
The only time you'll use BOOTSEL mode is if the board doesn't auto-reset or if you want to recover it. In that case, the Arduino IDE will create a .uf2 file and upload it through BOOTSEL automatically

SETUP OF ADIY PICO FOR ARDUINO IDE

Here's the step-by-step guide for installing Raspberry Pi Pico support in the Arduino IDE:

Step 1 – Open Arduino IDE Make sure you are using Arduino IDE 2.x (recommended) or at least Arduino IDE 1.8.19 .

Step 2 – Open Preferences Go to File → Preferences (Windows/Linux) or Arduino IDE → Preferences (Mac). In the window that opens, look for the box "Additional Board Manager URLs".

Step 3 – Add the Pico Board URL Paste this URL into the box:
https://github.com/earlephilhower/arduino-pico.git

Step 4 – Open Board Manager Go to Tools → Board → Board Manager. In the search bar, type: rp2040 Find "Raspberry Pi RP2040 Boards (by Earle Philhower) Click Install.

Step 5 – Enter Bootloader Mode & Upload Sketch
Press and hold the **BOOTSEL** button on your Raspberry Pi Pico, then connect it to your computer using a USB cable. Release the button once a new storage drive (usually named **RPI-RP2**) appears on your computer. This means the board is now in bootloader mode, ready to receive code. You can now upload your first sketch from the Arduino IDE.

Step 6- Connect your Raspberry Pi Pico via USB: Go to Tools → Board and select: Raspberry Pi Pico.

Step 7 – Select Port: Go to Tools → Port and choose the COM port that shows up when the Pico is connected. On Windows, it appears as "USB Serial Device". On Mac/Linux, it shows as /dev/tt

Step 8 – Upload a Test Sketch Open File → Examples → Basics → Blink. Click the Upload button. Now your Pico is fully set up for Arduino IDE

Interfacing Raspberry Pi Pico with IR Sensor

🔹 Components Needed

- Raspberry Pi Pico
- IR sensor module (obstacle detection type)
- Jumper wires(female to female_

Pin Connections

- **IR Sensor VCC → Pico 3.3V**
- **IR Sensor GND → Pico GND**
- **IR Sensor OUT → Pico GPIO15 (any GPIO can be used)**

Working

- The IR sensor emits IR rays; if an object is in front, rays reflect and the sensor gives **LOW** on output pin.
- If no object, output stays **HIGH**.

Code

```
int irSensorPin = 15;  // IR sensor connected to GPIO 15 (works for Pico)
int irValue;

void setup() {
  Serial.begin(9600);       // Start Serial Monitor
  pinMode(irSensorPin, INPUT); // Set pin as input
}

void loop() {
  irValue = digitalRead(irSensorPin); // Read sensor value

  if (irValue == LOW) {  // Object detected (LOW signal)
    Serial.println("Object Detected!");
  } else {
    Serial.println("No Object");
  }

  delay(100);  // Small delay for readability
}
```

Interfacing Raspberry Pi Pico with MQ2 Gas Sensor

Components Needed

- Raspberry Pi Pico
- MQ2 Gas Sensor module
- Jumper wires

MQ2 Pins

- **VCC → Pico 3.3V (or 5V if module supports)**
- **GND → Pico GND**
- **DO (Digital Out) → Pico GPIO14 (any GPIO)**
- **AO (Analog Out) → Not directly usable with Pico** (since Pico doesn't have true ADC >1v unless conditioned. MQ2 output can be connected to Pico's ADC pin with a voltage divider if analog reading is required).

For beginner use, we'll use **DO pin** (digital HIGH/LOW signal).

Working

- MQ2 senses smoke/gas.
- If gas concentration crosses threshold set by onboard potentiometer, DO pin becomes **LOW**.
- Otherwise, stays **HIGH**.

Code

```
int mq2Pin = 14;  // MQ2 sensor connected to GPIO 14 (works for Pico/Arduino)
int mq2Value;

void setup() {
  Serial.begin(9600);        // Start Serial Monitor
  pinMode(mq2Pin, INPUT);    // Set pin as input
}

void loop() {
  mq2Value = digitalRead(mq2Pin);  // Read sensor value

  if (mq2Value == LOW) {  // Gas detected (LOW signal)
    Serial.println("Gas/Smoke Detected!");
  } else {
    Serial.println("Air is Clean");
  }

  delay(500);  // 0.5 second delay for readability
}
```