



Introduction to Embedded Systems, Lecture 5

# Practical ES design with Raspberry Pi Pico in Arduino IDE

Dmitry Zaitsev

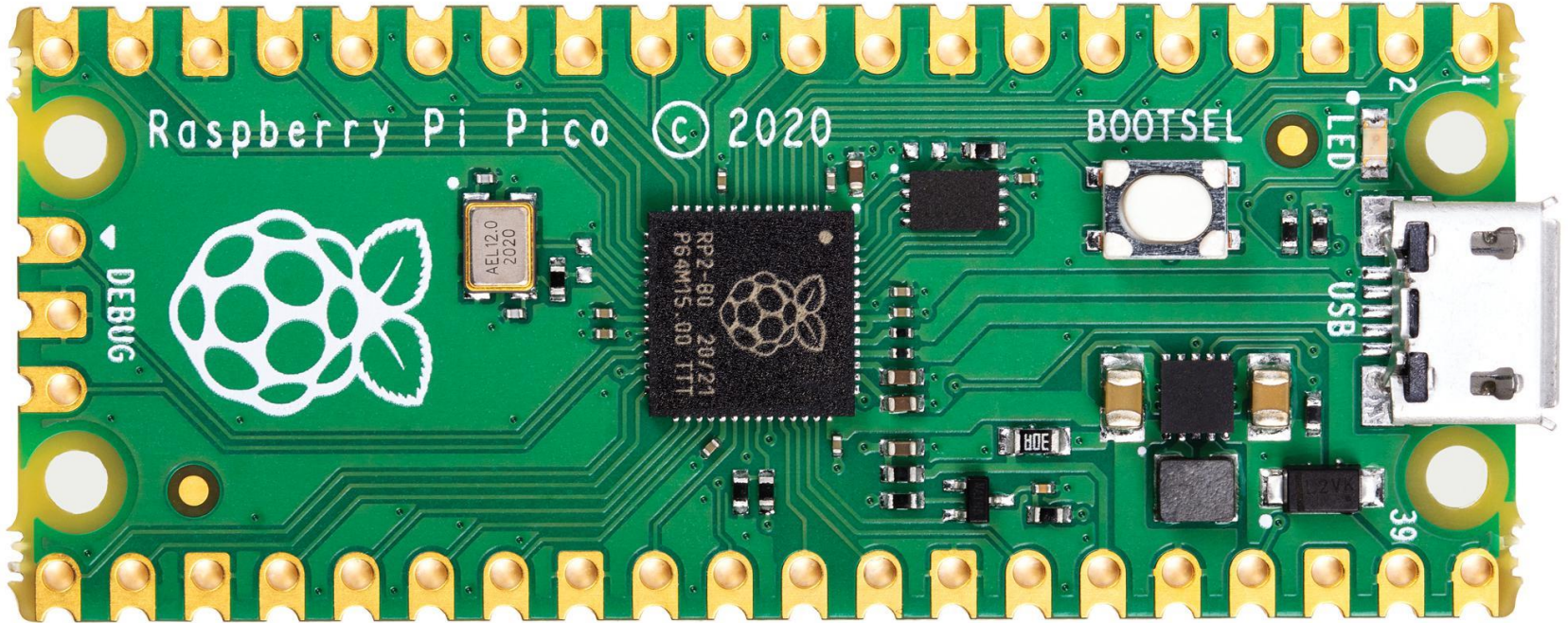
<http://daze.ho.ua>

# Raspberry Pi Pico

The Raspberry Pi Pico is a microcontroller with:

- an RP2040 chip featuring 2 x ARM Cortex-M0+ cores clocked at 133MHz;
- 256KB RAM;
- 2 MB of onboard Flash, 30 GPIO pins;
- micro-USB port and 40 main user pins
- 21mm x 51.3mm

# Raspberry Pi Pico



# Key Features of Raspberry Pi Pico

- RP2040 microcontroller with 2MB Flash memory that can be (re)programmed via USB or via the Serial Wire Debug (SWD) port
- Micro-USB B port for power and data
- 40 pin 21×51 ‘DIP’ style 1mm thick PCB with 0.1” through-hole pins also with edge castellations
  - Exposes 26 multi-function 3.3V General Purpose I/O (GPIO)
  - 23 GPIO are digital-only and 3 are ADC capable
  - Can be surface mounted as a module
- 3-pin ARM Serial Wire Debug (SWD) port
- Uses an on-board buck-boost SMPS – allows for flexible options for powering the unit from micro-USB, external supplies or batteries

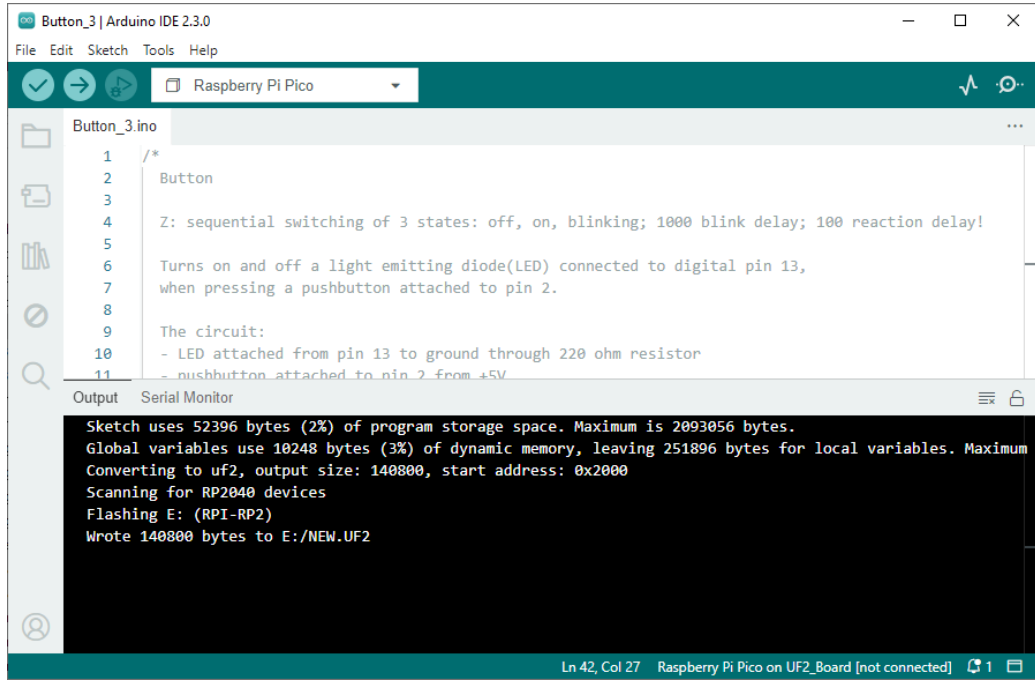
# Toolset: Arduino IDE

- Open source hardware specifications
- Load profiles of various microcontrollers
- Software development in C
- Compile and upload into Pico via USB
- Serial interface
- Debugging facilities

# Install Arduino IDE

- Download from <https://www.arduino.cc/>
- Install Arduino IDE
- Install profile of required equipment
- Attach microcontroller
- Design software-hardware specification: *sketch*
- Or use Arduino in cloud <https://cloud.arduino.cc/>

# Arduino IDE sketch + board (via USB)



Button\_3 | Arduino IDE 2.3.0

File Edit Sketch Tools Help

✓ → 🎵 📁 Raspberry Pi Pico 🔍

Button\_3.ino

```
1  /*
2  3  Button
4  5  Z: sequential switching of 3 states: off, on, blinking; 1000 blink delay; 100 reaction delay!
6  7  Turns on and off a light emitting diode(LED) connected to digital pin 13,
8  9  when pressing a pushbutton attached to pin 2.
10 11 The circuit:
12 13 - LED attached from pin 13 to ground through 220 ohm resistor
14 15 - pushbutton attached to pin 2 from +5V
```

Output Serial Monitor

Sketch uses 52396 bytes (2%) of program storage space. Maximum is 2093056 bytes.  
Global variables use 10248 bytes (3%) of dynamic memory, leaving 251896 bytes for local variables. Maximum :  
Converting to uf2, output size: 140800, start address: 0x2000  
Scanning for RP2040 devices  
Flashing E: (RPI-RP2)  
Wrote 140800 bytes to E:/NEW.UF2

Ln 42, Col 27 Raspberry Pi Pico on UF2\_Board [not connected]



# Download and install Arduino IDE

Software | Arduino

arduino.cc/en/software

PROFESSIONAL EDUCATION STORE

Search on Arduino.cc

SIGN IN

HARDWARE SOFTWARE CLOUD DOCUMENTATION COMMUNITY BLOG ABOUT

## Arduino Web Editor

Start coding online and save your sketches in the cloud. The most up-to-date version of the IDE includes all libraries and also supports new Arduino boards.

[CODE ONLINE](#) [GETTING STARTED](#)



## Over-the-Air Updates

DISCOVER MORE

## Downloads



### Arduino IDE 2.3.2

The new major release of the Arduino IDE is faster and even more powerful! In addition to a more modern editor and a more responsive interface it features autocompletion, code navigation, and even a live debugger.

For more details, please refer to the [Arduino IDE 2.0 documentation](#).

Nightly builds with the latest bugfixes are available through

#### DOWNLOAD OPTIONS

**Windows** Win 10 and newer, 64 bits  
**Windows** MSI installer  
**Windows** ZIP file

**Linux** AppImage 64 bits (X86-64)  
**Linux** ZIP file 64 bits (X86-64)

**macOS** Intel, 10.14: "Catalina" or newer, 64 bits  
**macOS** Apple Silicon, 11: "Big Sur" or newer, 64 bits

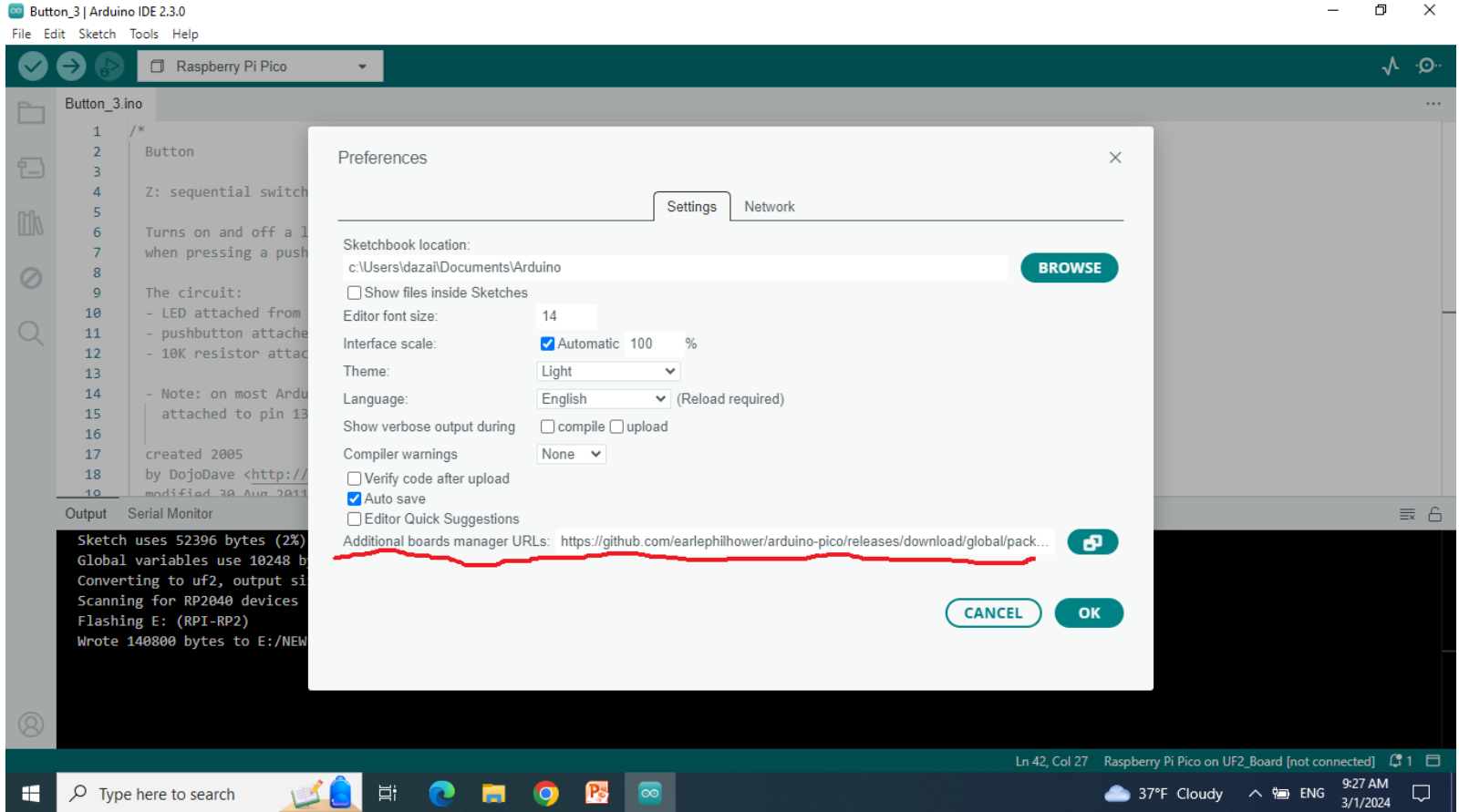
[Release Notes](#)

[Help](#)

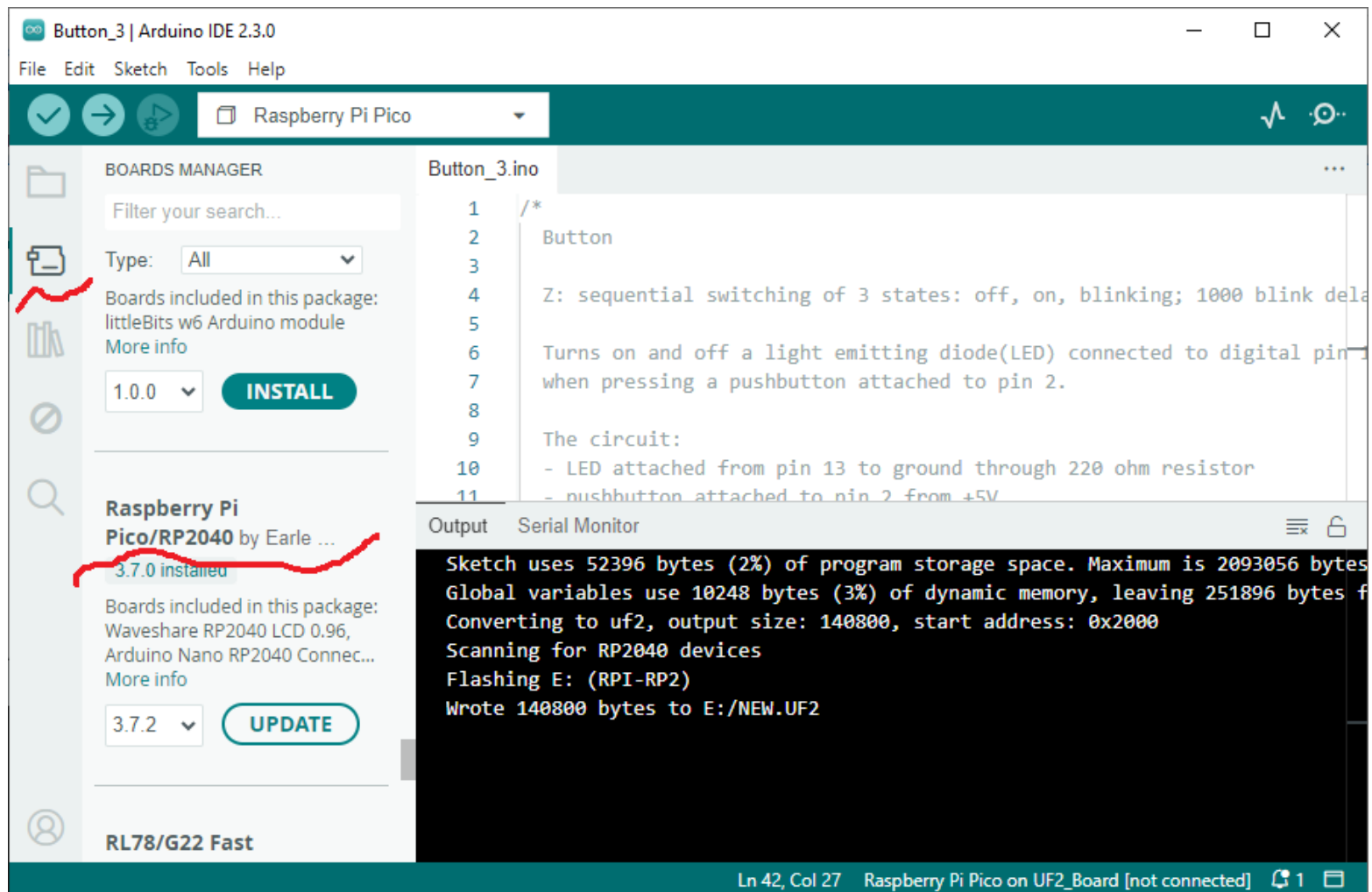


# Add RP Pico Board Support Package

[https://github.com/earlephilhower/arduino-pico/releases/download/global/package\\_rp2040\\_index.json](https://github.com/earlephilhower/arduino-pico/releases/download/global/package_rp2040_index.json)



# Install RP Pico boards



The screenshot displays the Arduino IDE 2.3.0 interface. The top menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar shows icons for checking, running, and uploading code, along with a dropdown menu currently set to 'Raspberry Pi Pico'.

The **BOARDS MANAGER** panel on the left is active. It features a search filter, a 'Type' dropdown set to 'All', and a list of installed boards. The **Raspberry Pi Pico/RP2040** by Earle F. Francis III is highlighted, with version **3.7.0** installed. Below it, other boards like 'Waveshare RP2040 LCD 0.96' and 'Arduino Nano RP2040 Connect' are listed. The **RL78/G22 Fast** board is also visible at the bottom.

The main editor window shows the **Button\_3.ino** sketch. The code includes a comment describing the sketch's function: sequential switching of 3 states (off, on, blinking) with a 1000ms delay. It also includes a circuit diagram description: an LED connected to pin 13 to ground through a 220 ohm resistor, and a pushbutton connected to pin 2 from +5V.

The **Serial Monitor** at the bottom displays the output of the sketch. The output text is as follows:

```
Sketch uses 52396 bytes (2%) of program storage space. Maximum is 2093056 bytes
Global variables use 10248 bytes (3%) of dynamic memory, leaving 251896 bytes f
Converting to uf2, output size: 140800, start address: 0x2000
Scanning for RP2040 devices
Flashing E: (RPI-RP2)
Wrote 140800 bytes to E:/NEW.UF2
```

The status bar at the bottom indicates the current location is Ln 42, Col 27, and the board is 'Raspberry Pi Pico on UF2\_Board [not connected]'.

# Choose RP Pico board

Button\_3 | Arduino IDE 2.3.0

File Edit Sketch Tools Help

Boards Manager

Filter your search...

Type: All

Boards included in this package:  
littleBits w6 Arduino module  
More info

1.0.0 **INSTALL**

**Raspberry Pi Pico/RP2040** by Earle...

3.7.0 installed

Boards included in this package:  
Waveshare RP2040 LCD 0.96,  
Arduino Nano RP2040 Connec...  
More info

3.7.2 **UPDATE**

RL78/G22 Fast

Button\_3.ino

```
1 /*
2  Button
3
4  Z: sequential switching of 3 states: off, on, blinking; 1000 blink delay
5
6  Turns on and off a light emitting diode(LED) connected to digital pin 13
7  when pressing a pushbutton attached to pin 2.
8
9  The circuit:
10  - LED attached from pin 13 to ground through 220 ohm resistor
11  - pushbutton attached to pin 2 from +5V
```

Output Serial Monitor

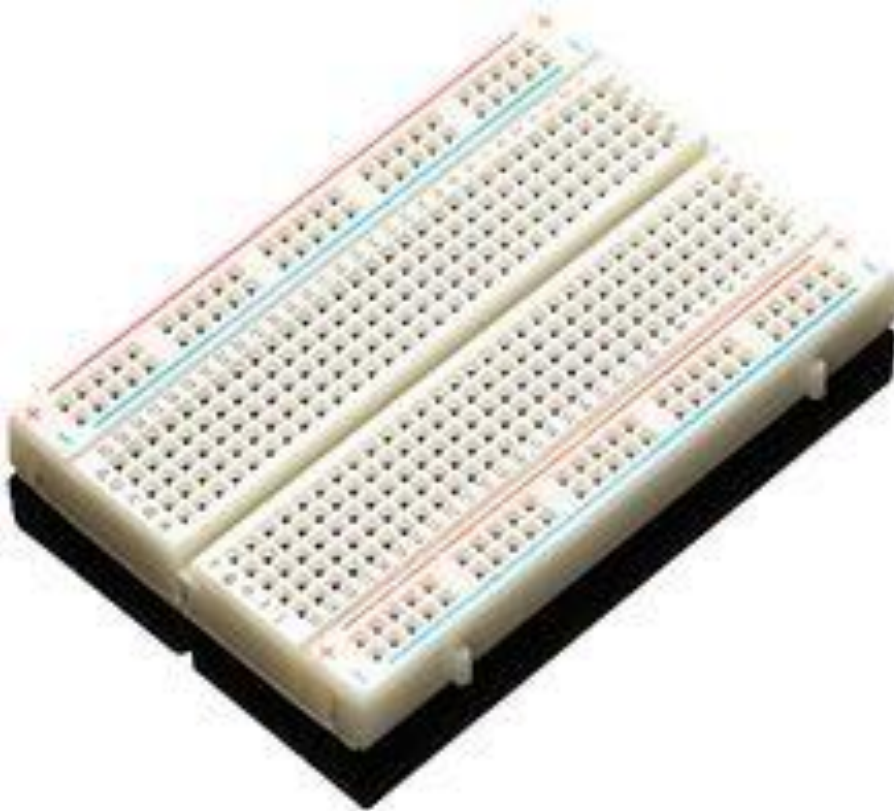
Sketch uses 52396 bytes (2%) of program storage space. Maximum is 2093056 bytes  
Global variables use 10248 bytes (3%) of dynamic memory, leaving 251896 bytes free  
Converting to uf2, output size: 140800, start address: 0x2000  
Scanning for RP2040 devices  
Flashing E: (RPI-RP2)  
Wrote 140800 bytes to E:/NEW.UF2

Ln 42, Col 27 Raspberry Pi Pico on UF2\_Board [not connected] 1

# Basics of embedded system hardware prototyping (drafting)

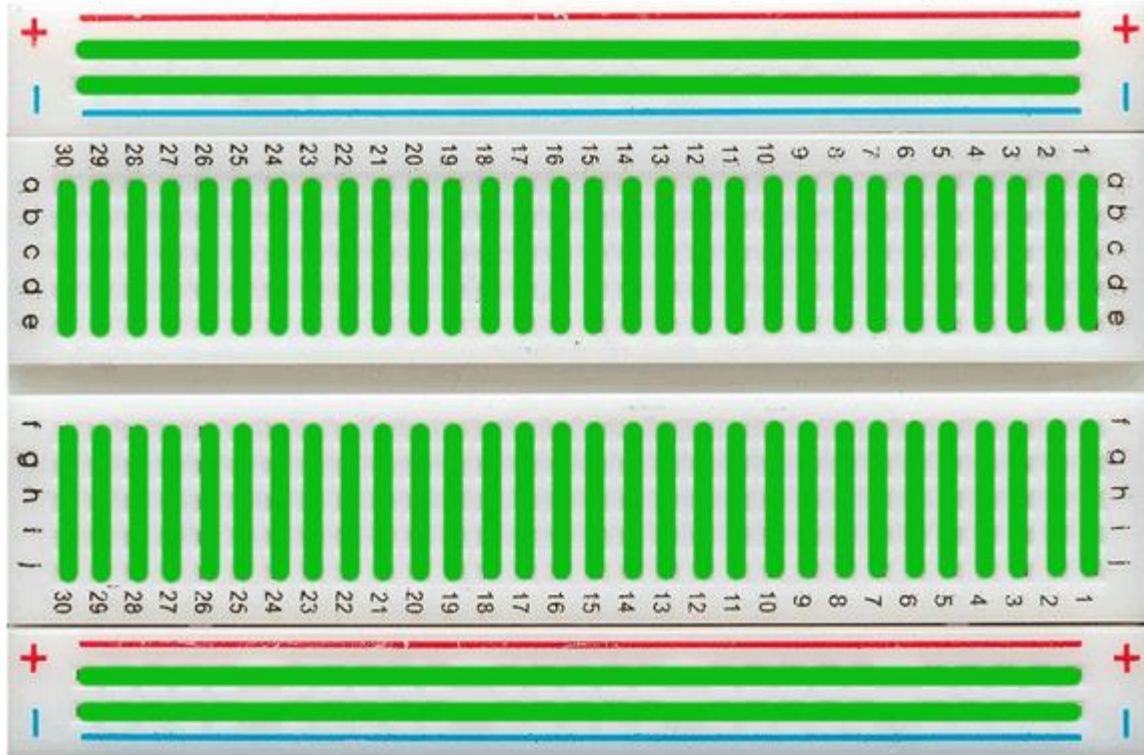
- Use breadboard to assemble a circuit
- Insert chips into breadboard
- Connect using internal busses of the breadboard
- Connect using jumpers

# Breadboard



- Insert pins of IC and other elements
- Connect elements via breadboard busses
- Connect elements via jumpers













































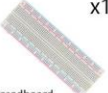

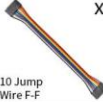










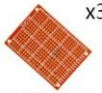




# Breadboard internal connections



# Basic elements of RP Pico kit

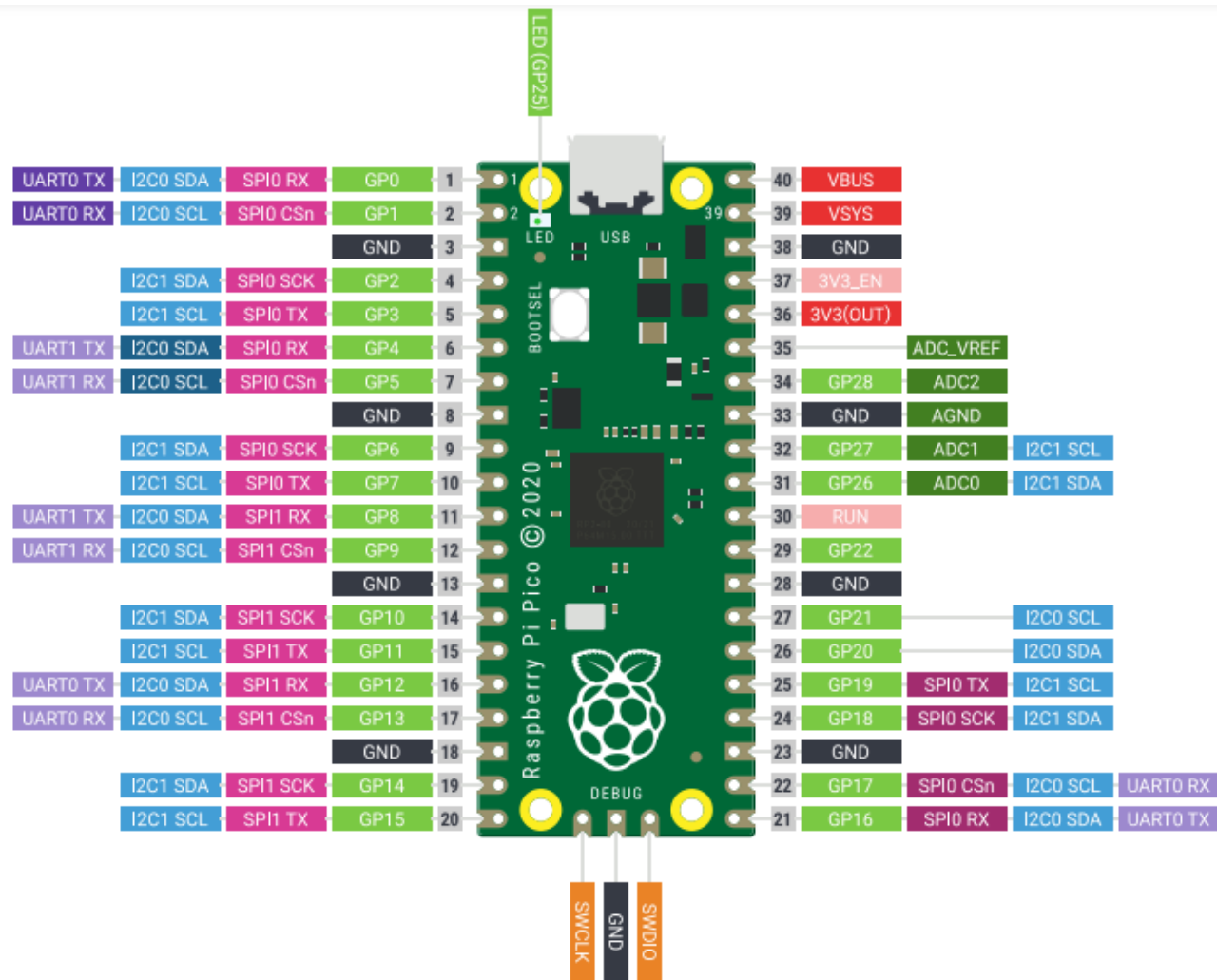
**Download Needed:**  
Tutorials and Code  
(No paper tutorial.)

The download link can be found on the box

<div>Download Needed: Tutorials and Code (No paper tutorial.)</div>		<div>Official Raspberry Pi Pico x1</div>		<div> micro:bit GPIO Extension Board x1</div>		<div> I2C LCD 1602 x1</div>		<div> Resistor-220 x20</div>							
<div>The download link can be found on the box!</div>						<div> Infrared Remote x1</div>		<div> 2X AA Battery Holder x1</div>		<div> Resistor-1K x10</div>					
<div> Potentiometer x3</div>		<div> Capacitor 0.1uf x2</div>		<div> Capacitor 10uf x2</div>		<div> Push Button x4</div>		<div> Big Push Button x4</div>		<div> Red Push Button Cap x1</div>		<div> Green Push Button Cap x1</div>		<div> Resistor-10K x10</div>	
<div> Switch x2</div>		<div> Vibration Switch x1</div>		<div> Keypad x1</div>		<div> 1N4001 Diode x2</div>		<div> 1N4148 Diode x2</div>		<div> Blue Push Button Cap x1</div>		<div> Yellow Push Button Cap x1</div>		<div> Red Led x10</div>	
<div> Photoresistor x1</div>		<div> 4-Digit 7-Segment Display x1</div>		<div> Passive Buzzer x1</div>		<div> Active Buzzer x1</div>		<div> L293D x1</div>		<div> 74HC595 x2</div>		<div> 8 RGB LED Module x1</div>		<div> Green Led x4</div>	
<div> DHT-11 x1</div>		<div> Thermistor x1</div>		<div> 7-Segment Display x1</div>		<div> Motor x1</div>		<div> Servo x1</div>		<div> Stepping Motor x1</div>		<div> RGB Led x1</div>		<div> Blue Led x4</div>	
<div> Joystick x1</div>		<div> Infrared Motion Sensor x1</div>		<div> Ultrasonic Ranging Module x1</div>		<div> Stepping Motor Driver x1</div>		<div> 9V Battery Cable x1</div>		<div> 65 Jump Wire M-M x1</div>		<div> Breadboard x1</div>		<div> Yellow Led x4</div>	
<div> 10 Jump Wire F-F x1</div>		<div> 10 Jump Wire F-M x1</div>		<div> NPN Transistor 8050 x2</div>		<div> PNP Transistor 8550 x2</div>		<div> Infrared Remote x1</div>		<div> 8*8 LEDMatrix x1</div>		<div> RFID Module x1</div>		<div> Led Bar Graph x1</div>	
<div> Relay x1</div>		<div> BreadBoardPower x1</div>		<div> MPU6050 x1</div>		<div> General Board x3</div>		<div> Pinout Card x1</div>		<div> Pinout Sticker x1</div>		<div> Resistor Color Code Card x1</div>		<div> Plastic Box x1</div>	

# RP Pico pins

■	Power
■	Ground
■	UART / UART (default)
■	GPIO, PIO, and PWM
■	ADC
■	SPI / SPI (default)
■	I2C / I2C (default)
■	System Control
■	Debugging





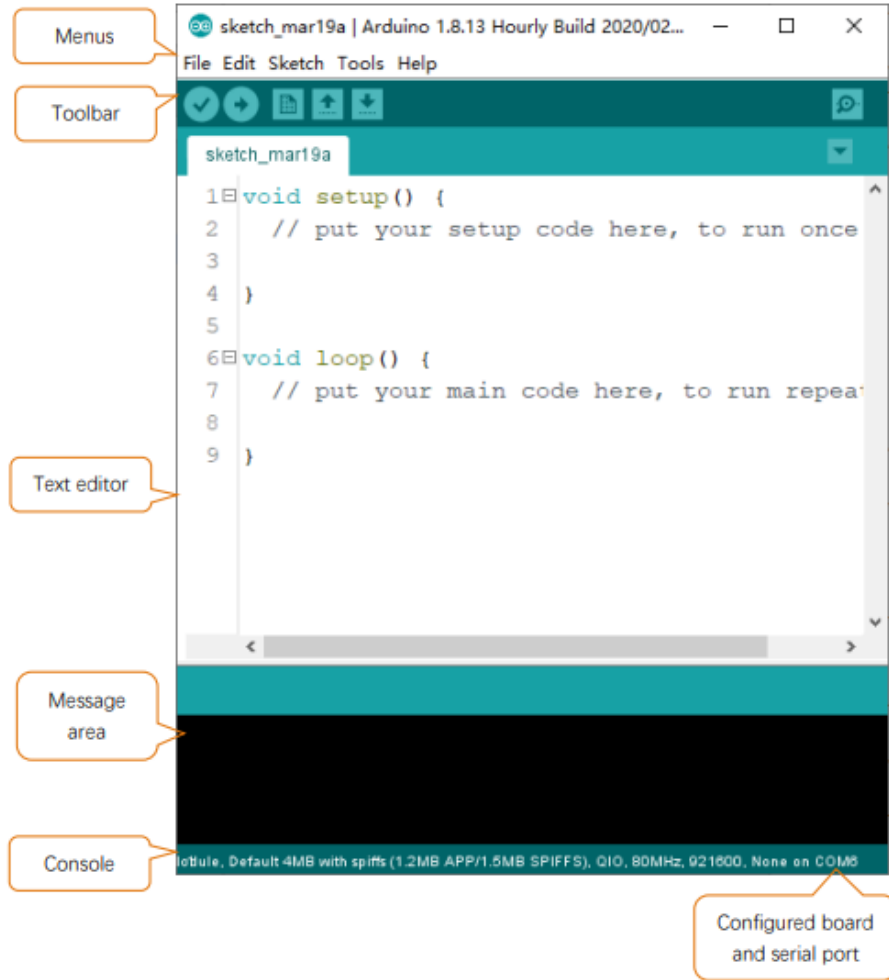
# Prototyping ES

- Electrical circuit layout
- Arduino IDE Sketch – program in C
- Electrical circuit board draft on a breadboard
- Connect RP Pico with computer
- Compile and upload sketch to RP Pico
- Check ES autonomous work

# Arduino IDE Sketch

- A sketch is the name that Arduino uses for a program. It's the unit of code that is uploaded to and run on an Arduino board.
- Two special functions are a part of every Arduino sketch: **setup()** and **loop()**.
- The **setup()** is called once, when the sketch starts.
- The **loop()** function is called over and over implementing basic loop of control.

# Basic elements of Arduino IDE interface



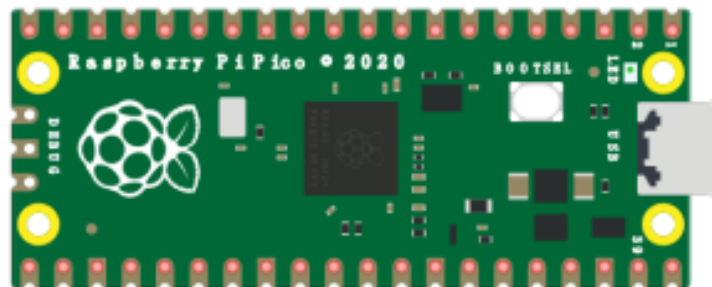
# Example: blink builtin LED (pin 25)

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

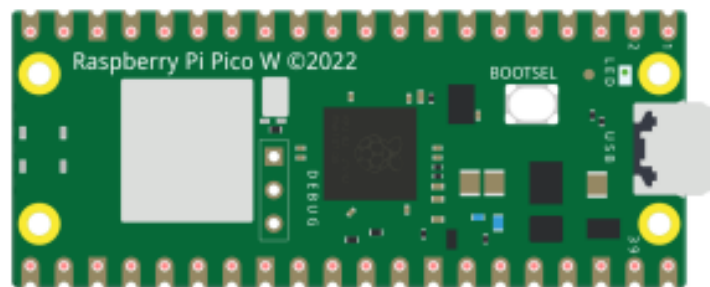
// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000);                     // wait for a second
  digitalWrite(LED_BUILTIN, LOW);  // turn the LED off by making the voltage LOW
  delay(5000);                     // wait for a second
}
```

## Component List

Raspberry Pi Pico (or Pico W) x1



or

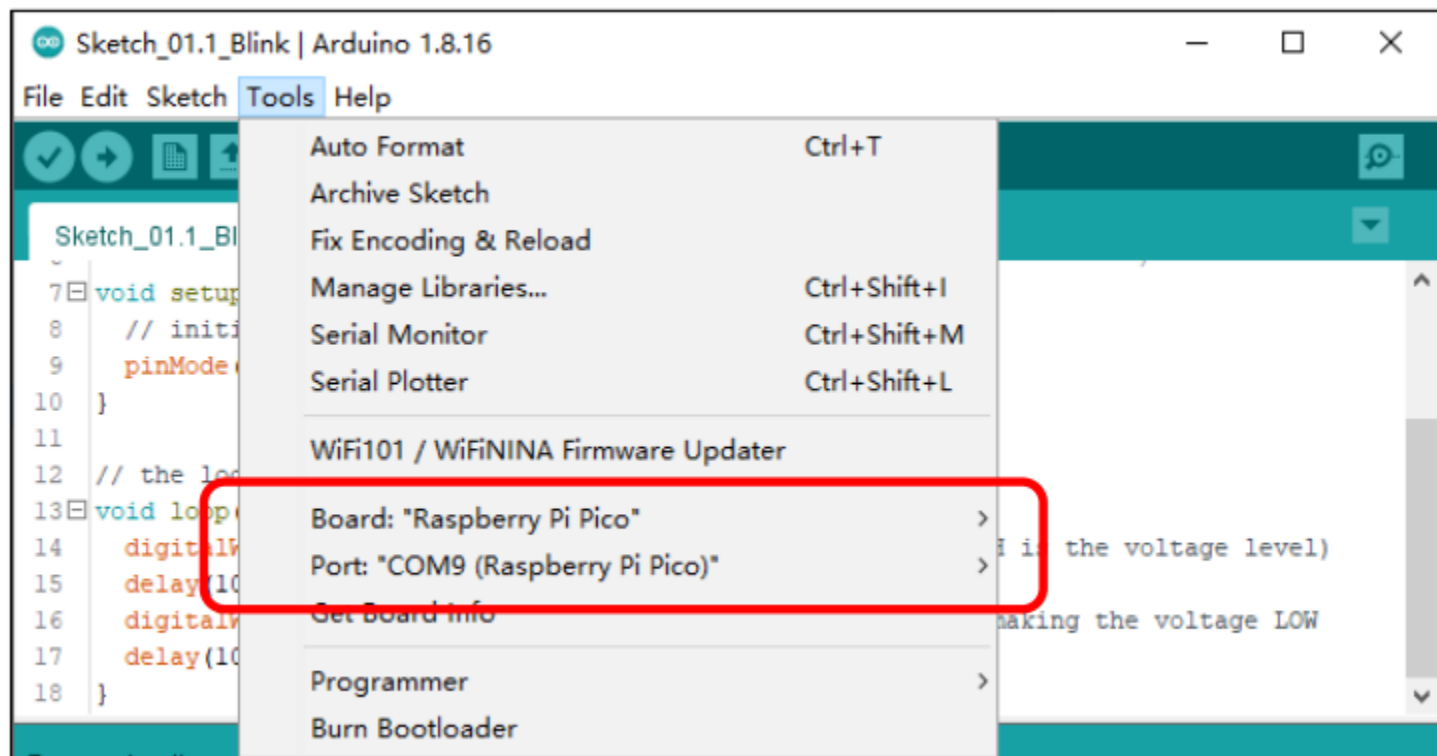


USB cable x1



Power

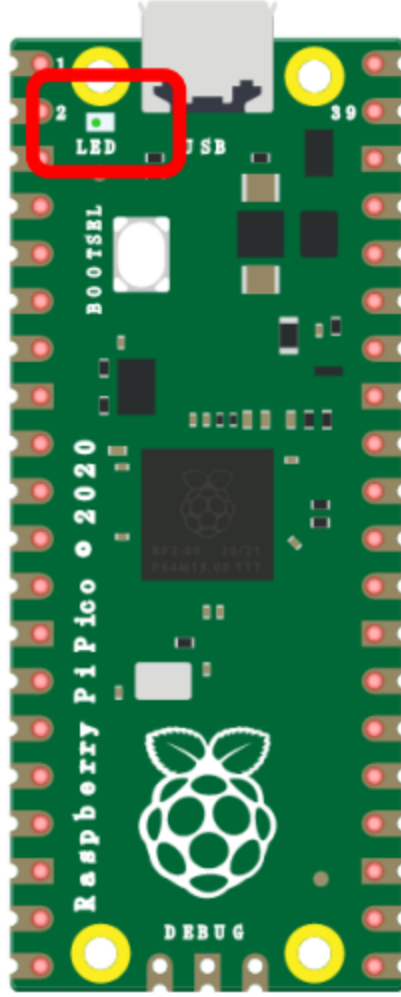
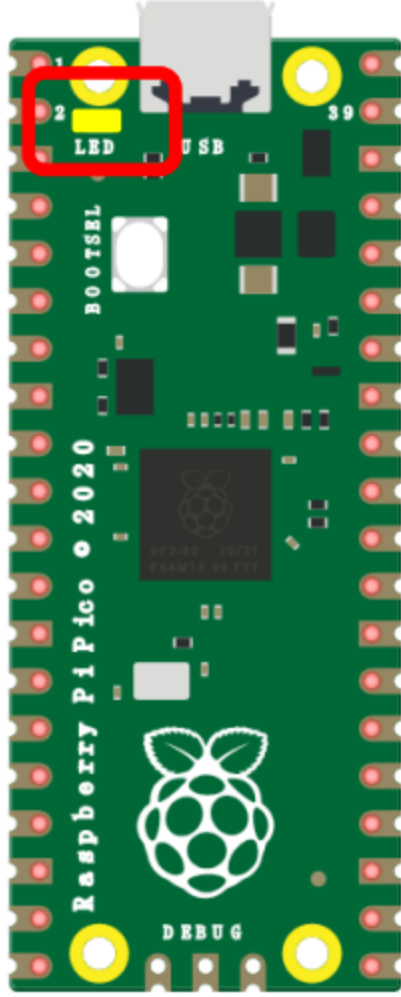
# Choose board and port



# Compile and upload sketch to Pico



# Internal LED flashing





# Example: blink external LED

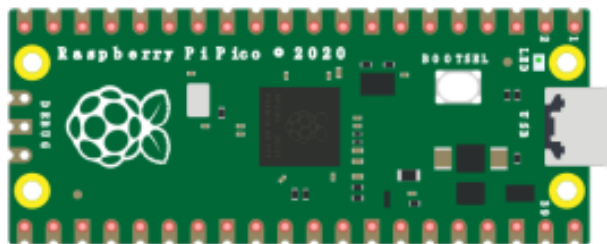
```
#define LED_PIN 15

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_PIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_PIN, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000);                  // wait for a second
  digitalWrite(LED_PIN, LOW);  // turn the LED off by making the voltage LOW
  delay(5000);                  // wait for a second
}
```

# Component list

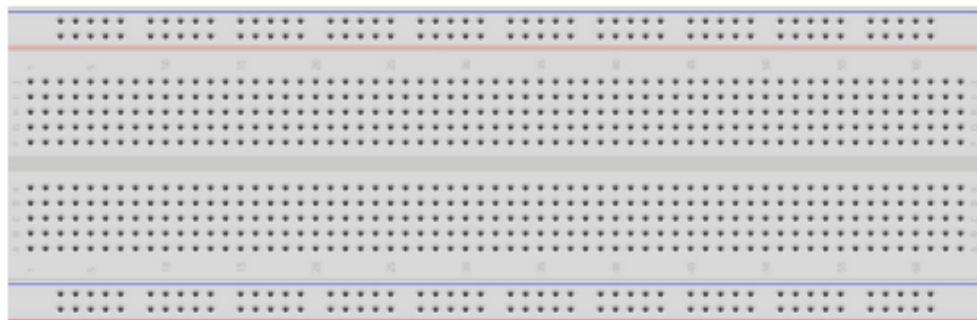
Raspberry Pi Pico x1



USB Cable x1



Breadboard x1



LED x1



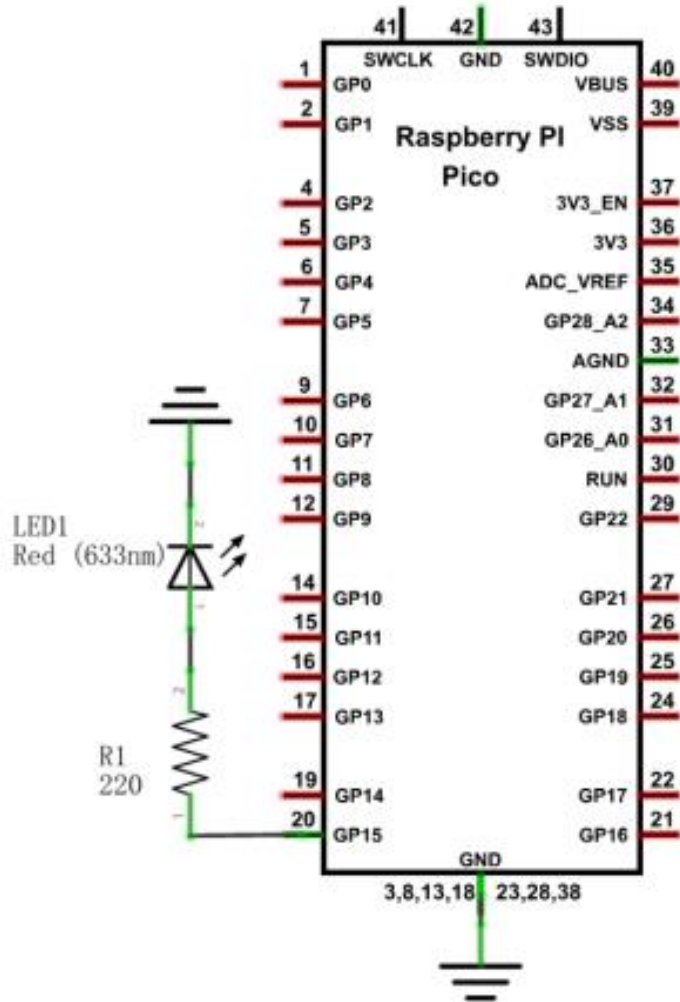
Resistor 220Ω x1



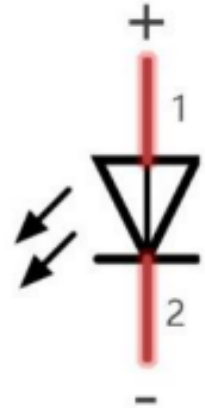
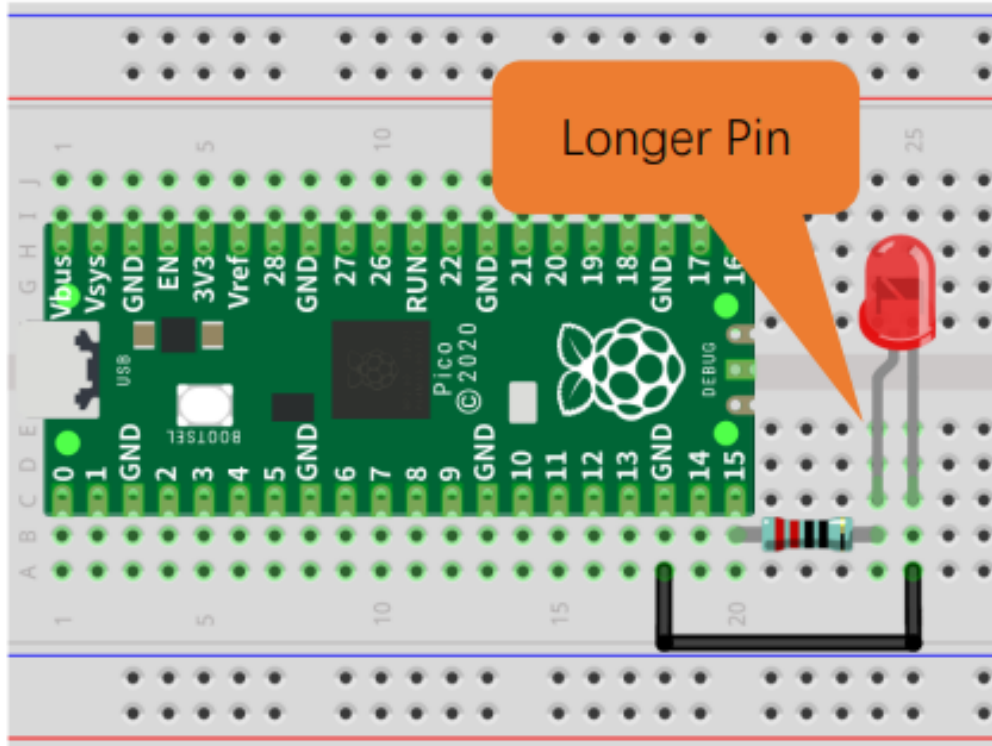
Jumper



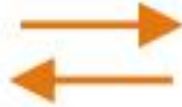
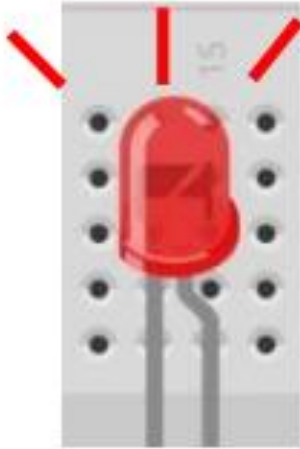
# Electric circuit



# Hardware connection



# Functioning



# Example: button (sensor) + LED (actuator)

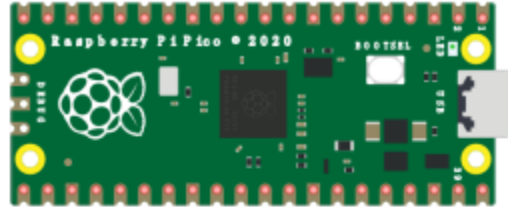
```
#define PIN_LED 15
#define PIN_BUTTON 13

void setup() {
  pinMode(PIN_LED, OUTPUT);
  pinMode(PIN_BUTTON, INPUT);
}

void loop() {
  if (digitalRead(PIN_BUTTON)==LOW)
  {
    digitalWrite(PIN_LED,HIGH);
  }else{
    digitalWrite(PIN_LED,LOW);
  }
}
```

# Component List

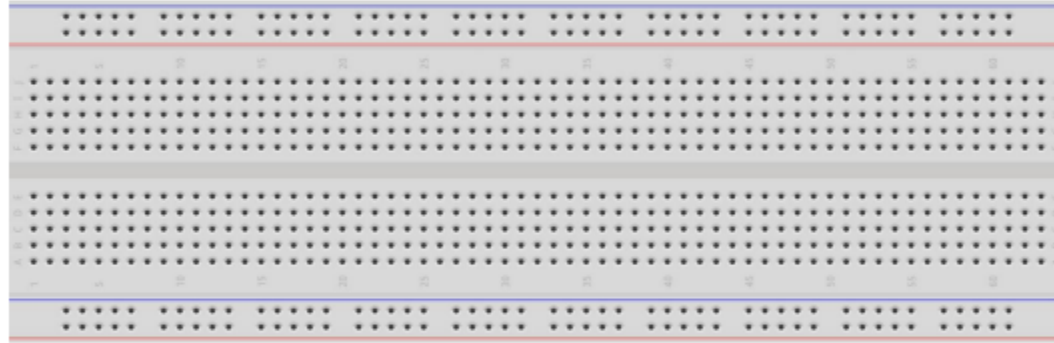
Raspberry Pi Pico x1



USB cable x1



Breadboard x1



Jumper



LED x1



Resistor  
220 $\Omega$  x1



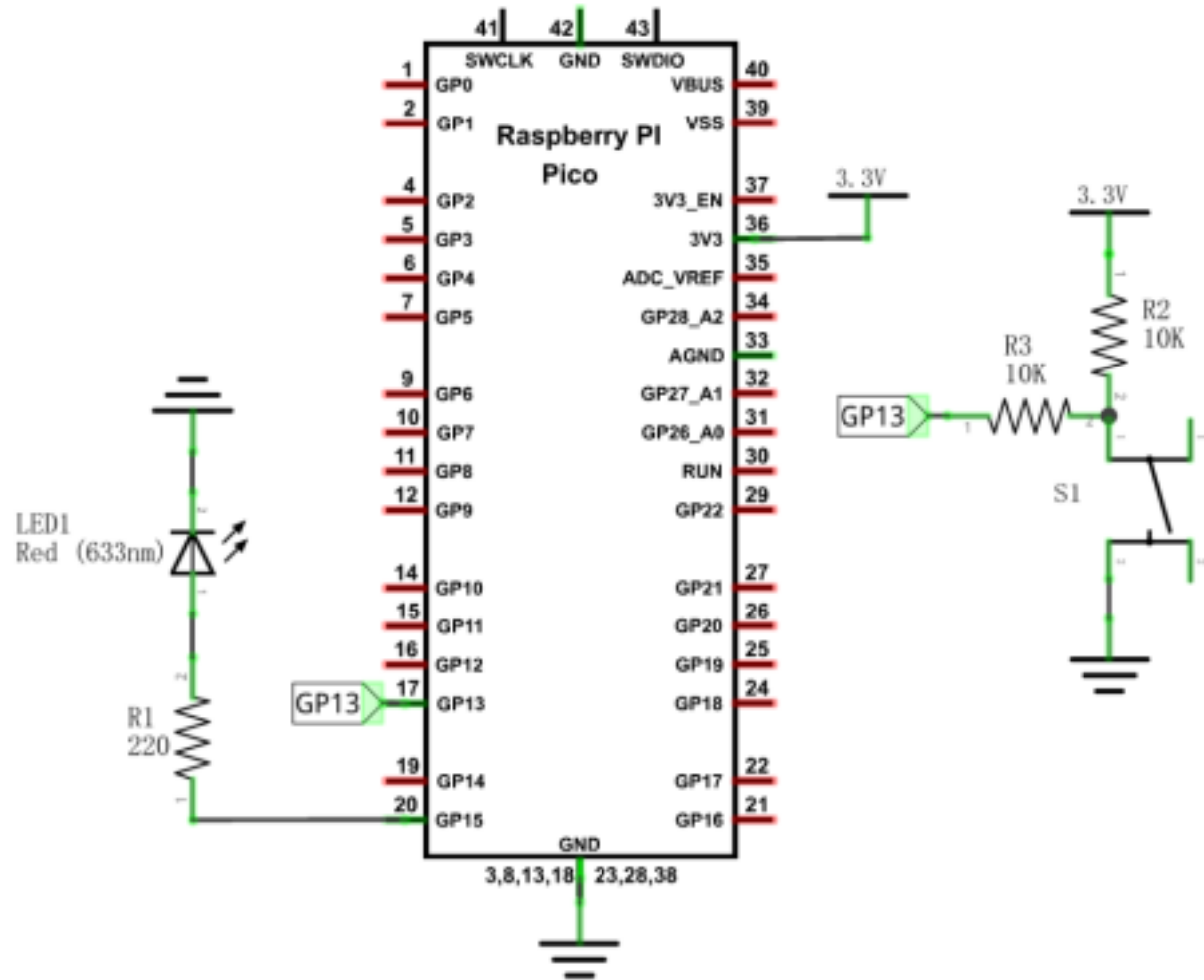
Resistor  
10k $\Omega$  x2



Push button  
x1

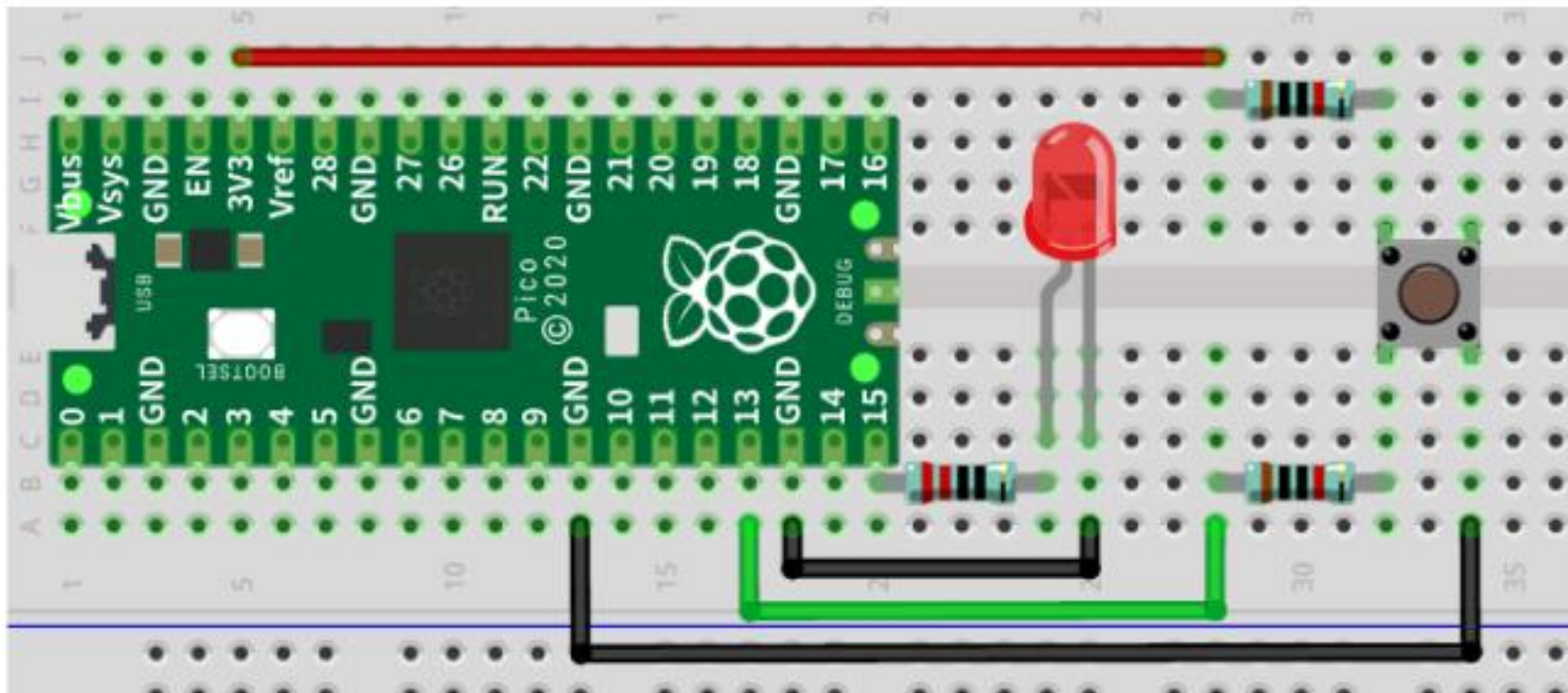


# Electric circuit

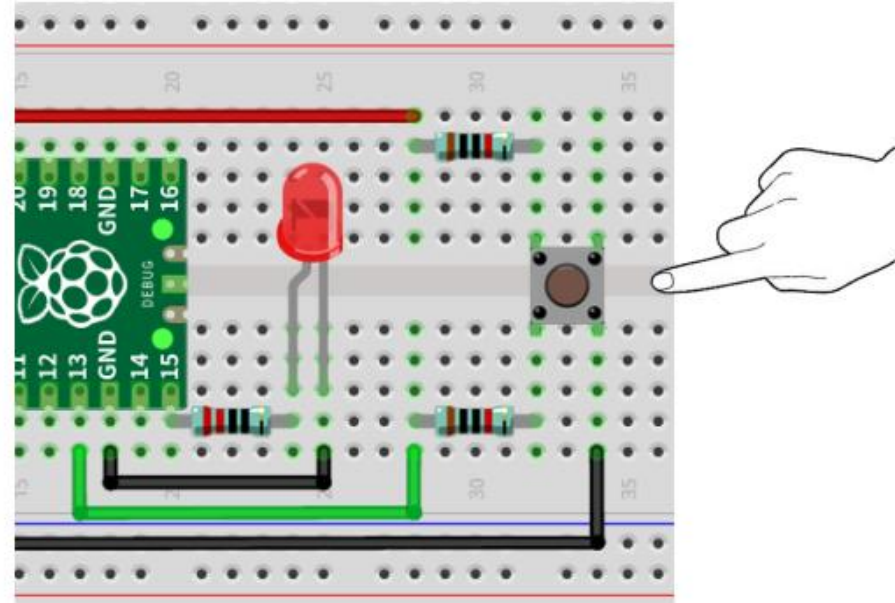
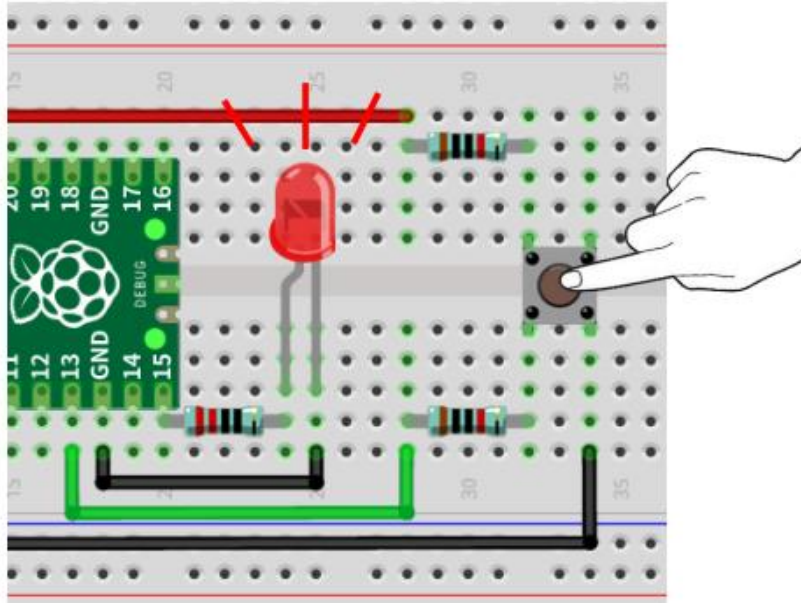




# Hardware connection



# Functioning



# Variations

- Vary blinking frequency and pattern
- Transmit Morse Code ([https://en.wikipedia.org/wiki/Morse\\_code](https://en.wikipedia.org/wiki/Morse_code)) of messages
- Use button to switch blinking frequency and pattern
- Use button to input data in Morse Code