


# NOVA

Proyecto NOVA — placa de desarrollo para control de colores basada en el diseño ColorEasyPICO2 (RP2350A). Esta documentación inicial explica qué es el proyecto, cómo montarlo, probarlo y opciones para venta/kit. Está basada en el proyecto referencia en OSHWLab: <https://oshwlab.com/lckfb-team/coloreasypicox>

 **Documentación PDF Mejorada:** Toda la documentación está disponible en PDFs profesionales con estilo visual inspirado en los handbooks de Raspberry Pi, usando los colores de marca NOVA (rojo y azul). Ver [docs/VISUAL\\_STYLE\\_GUIDE.md](#).

## Introducción

NOVA es una tarjeta de desarrollo ligera pensada para proyectos RGB/LED, prototipos y educación. Integra el microcontrolador RP2350A (familia Pico) y provee: fuente de alimentación en placa, botón BOOT/RESET, pines de E/S y un LED de usuario para pruebas rápidas.

## Características principales (explicado simple)

- Cerebro: un chip llamado RP2350A — es como el "cerebro" de la placa.
- Alimentación: se conecta con un cable USB tipo C (5V) y la placa baja la electricidad a 3.3V para las partes pequeñas.
- Memoria: tiene un chip pequeño donde se guarda el programa.
- Puntos para conectar cosas: tiene patitas (pines) para poner LEDs, sensores o cables.
- Botones: uno para entrar en modo de actualización ( **BOOT** ) y otro para reiniciar ( **RESET** ).
- Luz de prueba: hay un LED que se usa para comprobar que todo funciona.

## Especificaciones técnicas (resumen)

- Tensión de entrada: 5V (Type-C)
- Regulador 3.3V integrado (capaz para MCU y periféricos)
- Soporte de firmware: MicroPython (.uf2) y Arduino (core RP2040/RP2350)
- Conectividad: USB tipo C (datos + alimentación)
- Formato: placa compacta para montaje SMD

## ¿Qué puedo hacer con NOVA? (ideas y práctica rápida)

Después de meter el firmware, la placa NOVA sirve para aprender a programar y controlar hardware. Con ella puedes:

- Controlar LEDs y tiras RGB/NeoPixel para efectos de luz.
- Crear decoraciones o prototipos con patrones de color.
- Leer sensores (por ejemplo: temperatura, luz o distancia) usando I2C/SPI.
- Usar botones y potenciómetros como entradas para interactuar con programas.
- Controlar servos o módulos externos (con drivers adecuados).
- Practicar programación con MicroPython o con Arduino si prefieres.

### Práctica 1 — Encender un LED (solo con la placa)

Esta práctica usa solo la NOVA y un LED (o su LED interno) para verificar que la placa y el firmware funcionan.

Materiales mínimos:

- Placa NOVA y cable USB
- Un LED externo (opcional) y una resistencia de 220  $\Omega$  (si usas LED externo)

Conexiones:

- Para probar con el LED interno: no necesitas conexiones.
- Para un LED externo: conecta el ánodo (+) del LED al pin 15 de NOVA (por ejemplo), la resistencia en serie y el cátodo (-) al GND.

Código MicroPython (usar para onboard o externo):

```
from machine import Pin
import time

# Para LED interno usa 25, para externo usa 15
led = Pin(25, Pin.OUT) # cambia a Pin(15, Pin.OUT) si usas un LED externo

while True:
    led.value(1)
    time.sleep(0.5)
    led.value(0)
    time.sleep(0.5)
```

#### Pasos:

1. Flashea MicroPython si no está ya en la placa.
2. Conecta la placa por USB.
3. Copia este archivo como `main.py` en la placa.
4. Observa el LED parpadear; si usas LED externo, confirma la polaridad y la resistencia.

#### Práctica 2 — Control básico de una tira NeoPixel:

##### Materiales:

- Placa NOVA y cable USB
- Una tira o anillo NeoPixel (WS2812) de 5V (o un LED NeoPixel)
- Cables y (si la tira es grande) una fuente de 5V con suficiente corriente

##### Conexiones básicas:

- Conecta GND de la tira al GND de NOVA.
- Conecta el pin de datos de la tira al pin 16 de NOVA (puedes usar otro pin).
- Alimenta la tira con 5V (si la tira consume más de lo que puede dar la placa, usa una fuente externa con masa común).

Código MicroPython (guarda como `main.py`):

```

import machine
import neopixel
import time

PIN = 16
NUM = 8
np = neopixel.NeoPixel(machine.Pin(PIN), NUM)

def color_wipe(color, delay=0.05):
    for i in range(NUM):
        np[i] = color
        np.write()
        time.sleep(delay)

while True:
    color_wipe((255, 0, 0))    # rojo
    color_wipe((0, 255, 0))   # verde
    color_wipe((0, 0, 255))   # azul

```

Pasos:

1. Flashea MicroPython si no está ya en la placa.
2. Conecta NOVA por USB; aparecerá como unidad de almacenamiento.
3. Copia `main.py` a la placa; la ejecución debería comenzar sola.
4. Observa la tira cambiar de color. Modifica colores, velocidad y número de LEDs para experimentar.

Siguientes pruebas:

- Añadir un botón para cambiar efectos.
- Leer un sensor de luz y ajustar brillo automáticamente.
- Crear patrones más complejos o sincronizar varias placas.

Si quieres, adapto estas prácticas a la cantidad de LEDs que tengas o te doy el código para Arduino en vez de MicroPython.

## ¿Qué es "firmware" y cómo meterlo en la placa? (explicación para todos)

Firmware = el programa que vive dentro de la placa. Piensa en la placa como un robotito: el firmware es el conjunto de instrucciones que le dice qué hacer.

Meter el firmware (flasheo) = copiar ese programa dentro del robotito.

MicroPython (forma fácil, paso a paso)

1. Descargar el firmware (.uf2):

2. Puedes descargar la versión oficial de MicroPython para RP2 desde la web: <https://micropython.org/download/rp2/> (busca el archivo que termina en `.uf2`). Si tienes un firmware específico para NOVA, coloca el archivo `NOVA.uf2` dentro de tu repositorio `docs/firmware/` y enlázalo desde aquí.
3. Dónde guardar el archivo descargado:
4. Guarda el archivo `.uf2` primero en tu carpeta de descargas local (por ejemplo `Descargas` o `Downloads`). No lo abras; lo copiasremos a la placa cuando ésta aparezca como unidad USB.
5. Modo BOOT y copia del firmware (paso importante — resáltalo):
6. **PRESIONA EL BOTÓN `BOOT` Y, SIN SOLTARLO, CONECTA LA PLACA AL CABLE USB. LUEGO SUELTA `BOOT`.** Esto hace que la placa aparezca como una memoria USB en tu computadora.
7. Copiar el archivo `.uf2` a la placa:
8. Cuando la placa aparezca como unidad (por ejemplo `RPI-RP2`), copia el archivo `.uf2` desde tu carpeta de descargas a esa unidad. Al copiar el archivo la placa normalmente se reinicia automáticamente.
9. Cómo saber si el firmware se guardó correctamente:
10. Al terminar la copia la unidad USB suele desaparecer y la placa se reinicia; eso indica que el `.uf2` fue aceptado.
11. Para comprobarlo, abre un terminal serie o un IDE como Thonny (ver abajo): deberías ver el prompt de MicroPython ( `>>>` ) en el REPL. Otra señal es que si `main.py` está en la placa, su comportamiento (por ejemplo, un LED parpadeando) empezará.
12. Dónde guardar tu programa de ejemplo y cómo editarlo:
13. Archivo en la placa: el programa que ejecuta la placa se llama normalmente `main.py`. Para que se ejecute automáticamente, guarda `main.py` en la raíz de la unidad de la placa cuando ésta esté montada.
14. Editar con Thonny (recomendado): instala Thonny (<https://thonny.org/>).
  - Abre Thonny, conecta la placa por USB y en la esquina inferior derecha selecciona el intérprete MicroPython (Raspberry Pi Pico/RP2).
  - Usa la pestaña "Files" para ver la unidad de la placa y editar o subir `main.py` directamente.
15. Editar con el sistema de archivos: cuando la placa aparece como unidad USB puedes copiar `main.py` desde tu editor habitual (VS Code, Notepad, etc.) a la unidad montada; al expulsarla y reiniciarla, la placa ejecutará el `main.py`.

16. Programar desde WSL (Linux en Windows): usa mpremote para subir archivos y abrir REPL.

- Instala python3-venv: `sudo apt install -y python3.12-venv`
- Crea un venv: `python3 -m venv ~/nova_venv`
- Activa el venv: `source ~/nova_venv/bin/activate`
- Instala mpremote: `pip install mpremote`
- Subir main.py: `mpremote connect /dev/ttyS6 fs put main.py :/main.py`
- Abrir REPL: `mpremote connect /dev/ttyS6 repl`
- Nota: COM7 en Windows → /dev/ttyS6 en WSL (COMn → /dev/ttyS{n-1}).

17. Problemas comunes y comprobaciones rápidas:

18. Si la placa no aparece al conectar: cambia el cable USB por uno que soporte datos, prueba otro puerto o presiona **BOOT** de nuevo.

19. Si no ves el REPL en Thonny: verifica el puerto serie correcto y el intérprete seleccionado.

20. Si `main.py` no se ejecuta: revisa el nombre exacto ( `main.py` ) y que esté en la raíz de la unidad de la placa.

Si quieres, puedo añadir un enlace directo en este repositorio a un archivo `docs/firmware/NOVA.uf2` (si me das el archivo o el URL), y puedo añadir capturas paso a paso para Thonny.

Explicación sencilla del ejemplo de parpadeo (qué hace cada línea):

```
from machine import Pin    # Dice: "voy a usar una patita de la placa"
import time                # Dice: "voy a usar pausas (esperas)"

led = Pin(25, Pin.OUT)     # Conecta la patita 25 a una luz (LED)
while True:                # Repite esto para siempre
    led.value(1)            # Enciende la luz
    time.sleep(1)           # Espera 1 segundo
    led.value(0)           # Apaga la luz
    time.sleep(1)           # Espera 1 segundo
```

Arduino (otra forma de programar):

- Puedes usar Arduino IDE si sabes usarlo, pero para empezar MicroPython es más fácil. Si pruebas Arduino, instala el paquete de placas y selecciona "Raspberry Pi Pico/RP2040/RP2350".

## Pruebas sencillas (explicado con palabras muy simples)

- Primero: conecta el cable USB. ¿Se enciende alguna luz? Si sí, la placa tiene energía.
- Segundo: prueba el parpadeo. Carga el programa de ejemplo (el que enciende y apaga la luz). Si la luz parpadea, todo está bien.
- Tercero: prueba los botones. Pulsa **RESET** para reiniciar la placa; pulsa **BOOT** mientras conectas para poder copiar el programa.

Si algo no funciona, prueba esto en orden:

1. Cambia de cable USB (algunos cables solo cargan, no transfieren datos).
2. Prueba otro puerto USB en la computadora.
3. Si aún no aparece nada, revisa con una lupa la placa: busca soldaduras raras o piezas que estén torcidas.

Si quieres, te lo explico paso a paso en una videollamada o escribo una guía con fotos hechas por ti para que quede perfecto para los clientes.

## Venta, kits y contacto

Decide si ofrecerás la placa como kit (PCBs + componentes) o ensamblada. Para las fichas de producto y el instructivo para clientes incluye:

- Fotografías claras de la placa y del producto montado
- Listado de componentes (BOM) con referencias y proveedores
- Instrucciones paso a paso (montaje, flasheo, prueba)
- Opciones de compra: precio kit / ensamblado, tiempos de entrega

Proporciona aquí el email y/o enlace de compra que quieras mostrar a clientes.

## Licencia

La referencia original indica licencia LGPL 3.0; verifica y establece la licencia adecuada para NOVA antes de publicar.

## Referencias

- Diseño base: <https://oshwlab.com/lckfb-team/coloreasypicox>

- Editor y esquemático: enlace de Design Drawing en la página de referencia

## Documentación y PDFs

La documentación de NOVA está disponible en formato PDF con un diseño profesional

inspirado en los handbooks de Raspberry Pi:

### PDFs Disponibles

- [docs/manual\\_for\\_kids.pdf](#) - Manual de inicio paso a paso para principiantes
- [docs/Documento\\_NOVA.pdf](#) - Instructivo completo del producto
- [docs/NOVA\\_Guide.pdf](#) - Guía completa generada del README

### Estilo Visual NOVA

Los PDFs utilizan:

- **Colores de marca:** Rojo (#DC2626) y Azul (#2563EB)
- **Tipografía profesional:** DejaVu Sans
- **Elementos visuales:** Logo, iconos, bloques de código estilizados
- **Diseño atractivo:** Para niños, técnicos e ingenieros

### Generar PDFs

Para regenerar los PDFs con el estilo mejorado:

```
python3 scripts/build_all_pdfs.py
```

Ver [docs/VISUAL\\_STYLE\\_GUIDE.md](#) para más detalles sobre el estilo visual y cómo personalizar los PDFs.