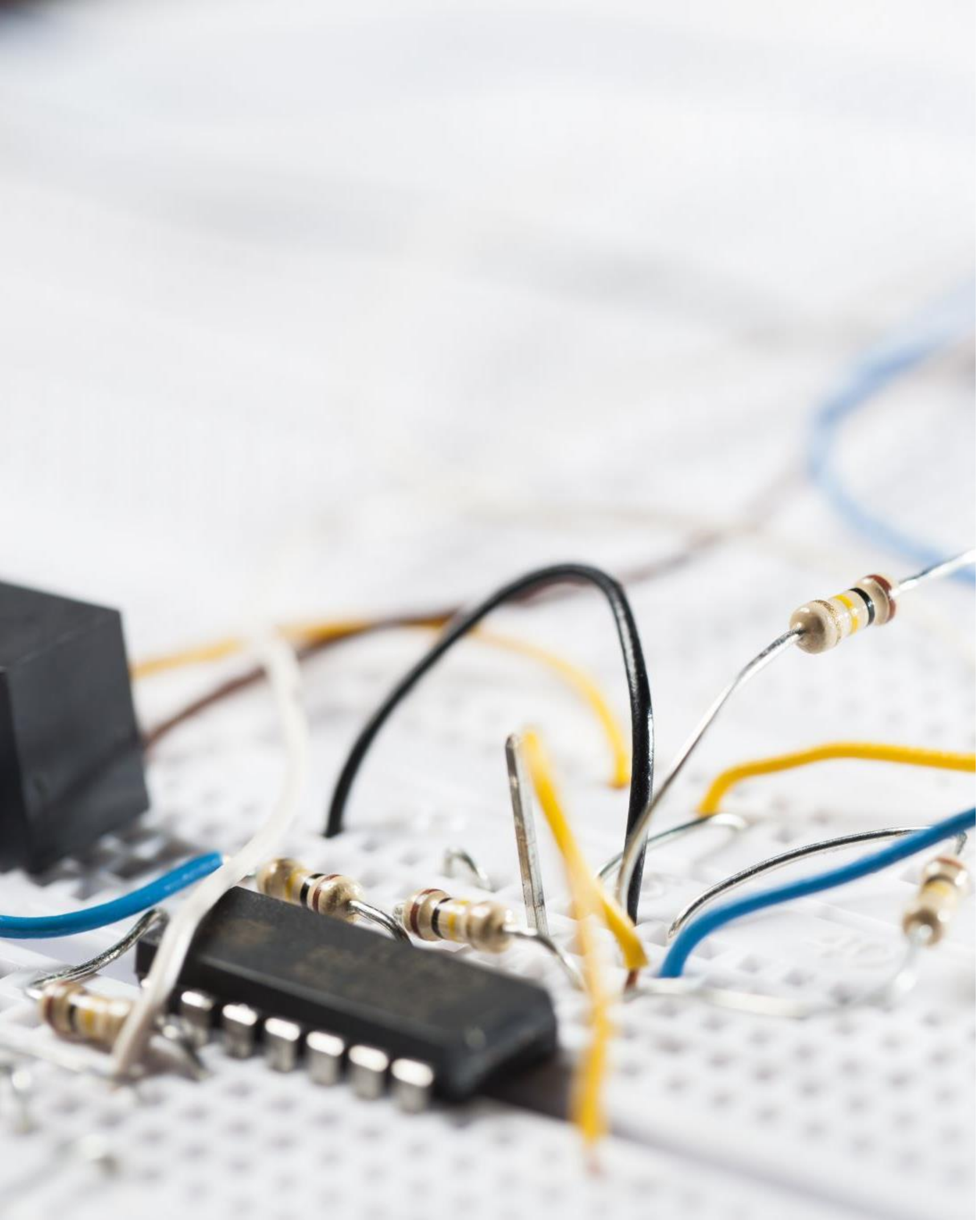


Pins and LEDs

Hardware 2 Course

The Project

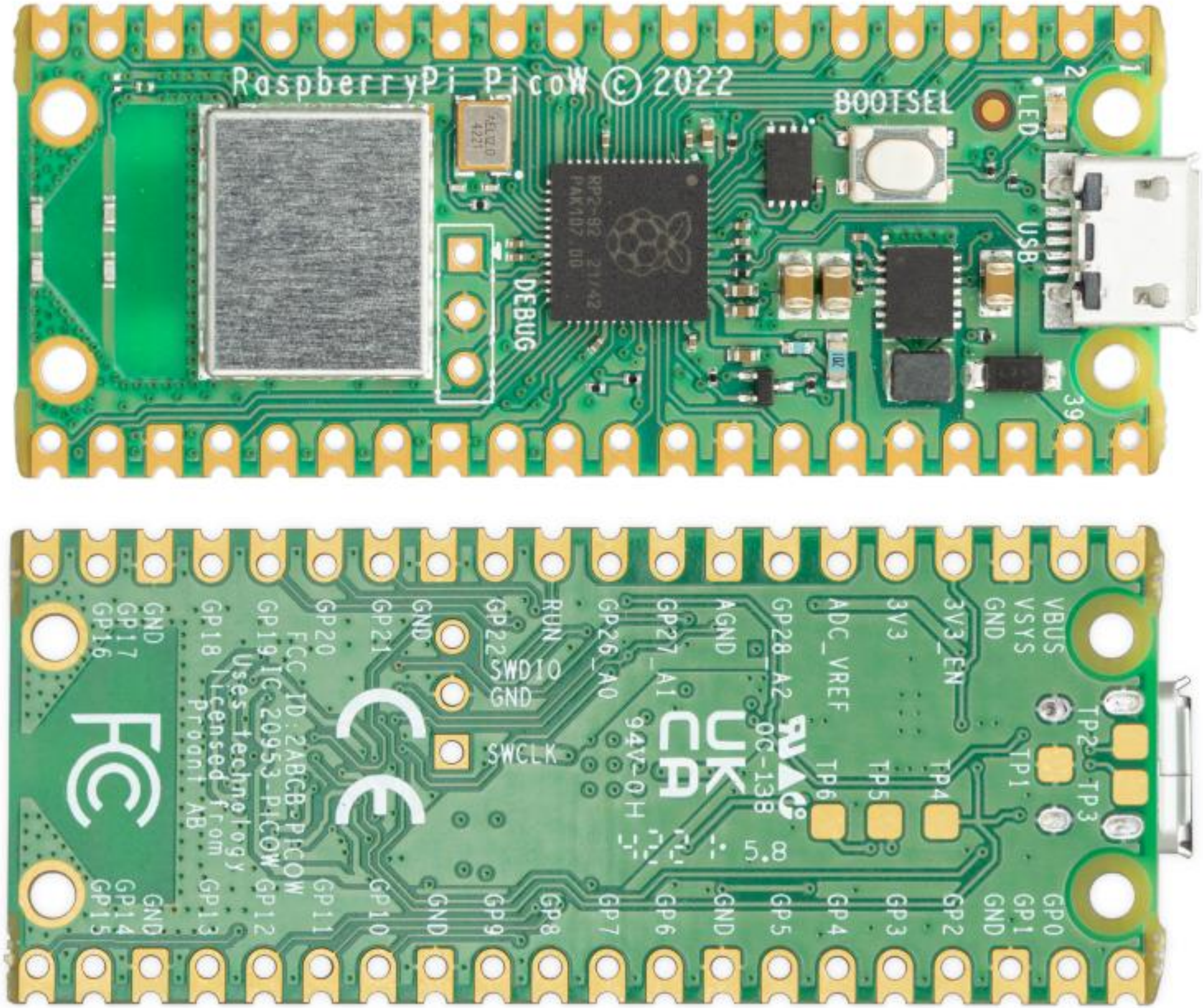
School of ICT

Metropolia

Contents

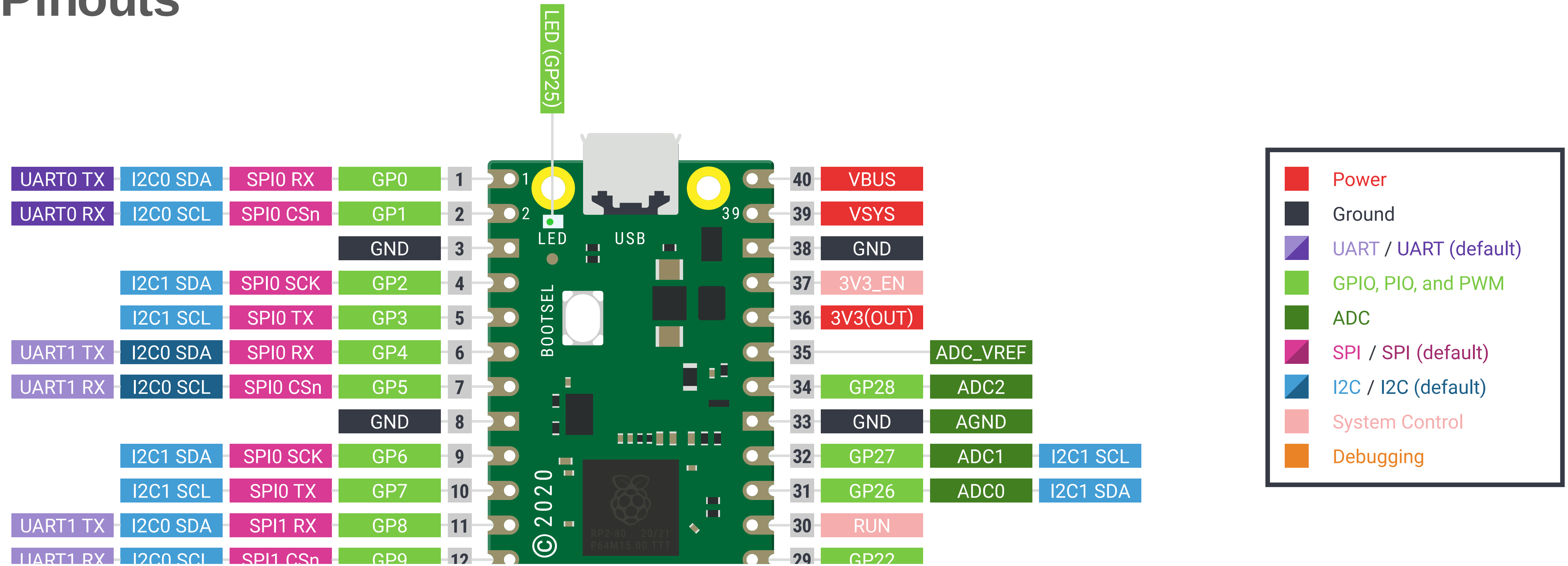
- Raspberry Pi Pico W board
- Development board
- MicroPython and Pins
- Pico's onboard LED
- LEDs on protoboard
- Exercise

The Raspberry Pi Pico W Rev3 board



[Raspberry Pi Pico W Datasheet]

Pinouts



UART0 TX	I2C0 SDA	SPI0 RX	GP0	1	40	VBUS		
UART0 RX	I2C0 SCL	SPI0 CSn	GP1	2	39	VSYS		
			GND	3	38	GND		
	I2C1 SDA	SPI0 SCK	GP2	4	37	3V3_EN		
	I2C1 SCL	SPI0 TX	GP3	5	36	3V3(OUT)		
UART1 TX	I2C0 SDA	SPI0 RX	GP4	6	35		ADC_VREF	
UART1 RX	I2C0 SCL	SPI0 CSn	GP5	7	34	GP28	ADC2	
			GND	8	33	GND	AGND	
	I2C1 SDA	SPI0 SCK	GP6	9	32	GP27	ADC1	I2C1 SCL
	I2C1 SCL	SPI0 TX	GP7	10	31	GP26	ADC0	I2C1 SDA
UART1 TX	I2C0 SDA	SPI1 RX	GP8	11	30	RUN		
UART1 RX	I2C0 SCL	SPI1 CSn	GP9	12	29	GP22		

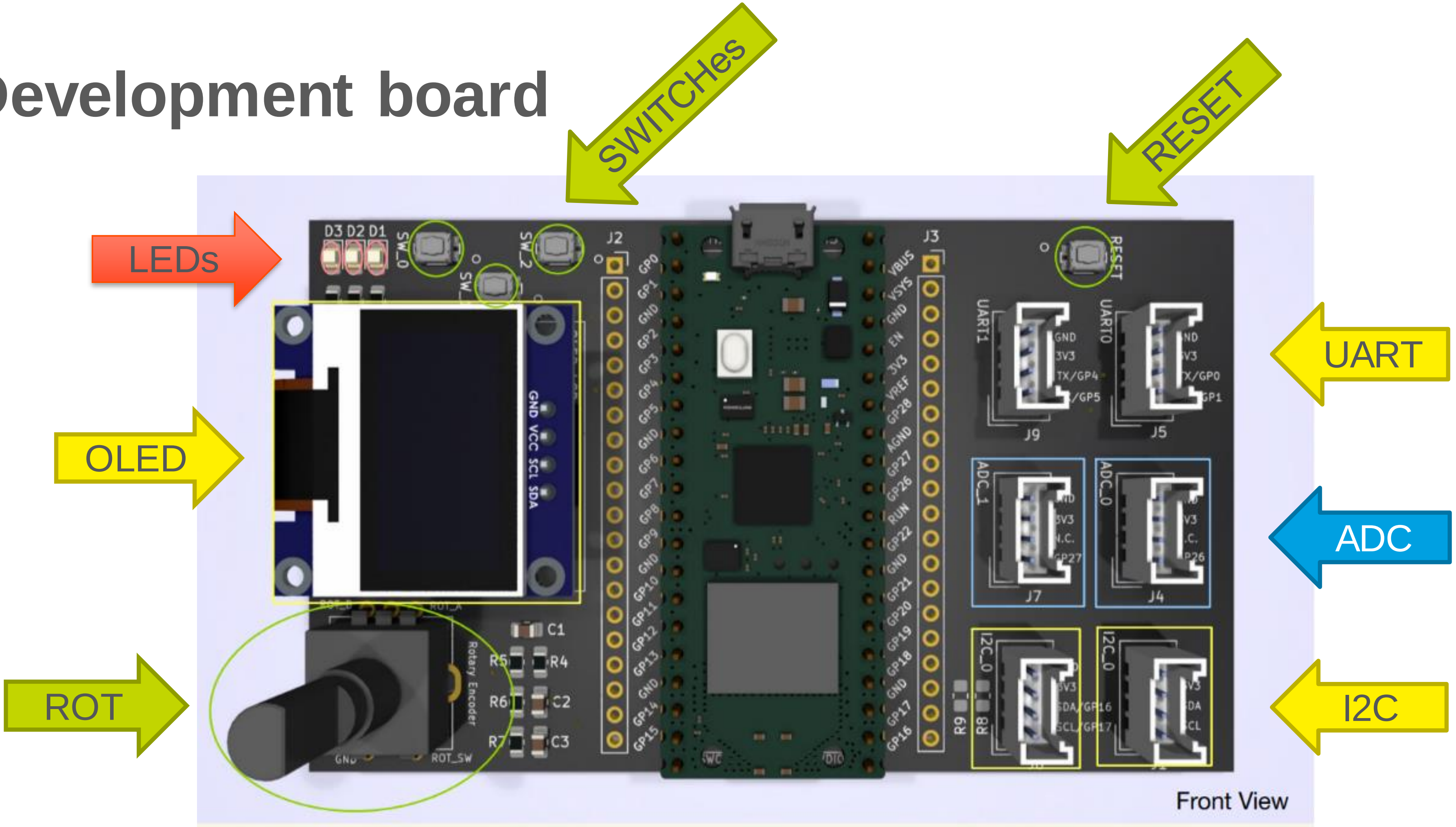
- 🟥 Power
- ⬛ Ground
- ▨ UART / UART (default)
- 🟩 GPIO, PIO, and PWM
- 🟩 ADC
- ▨ SPI / SPI (default)
- ▨ I2C / I2C (default)
- ■ System Control
- 🟧 Debugging

GP = General Purpose, SPI = Serial Peripheral Interface, I2C = Inter-Intergrated Circuit, UART = UniversalAsynchronous Receiver-Transmitter, SCK, SCL = Serial Clock, SDA = Serial Data, TX = Transmitter, RX = Receiver, ADC = Analog Digital Converter, PWM = Pulse Width Modulation

Pinouts and design files - Raspberry Pi Pico and Pico W

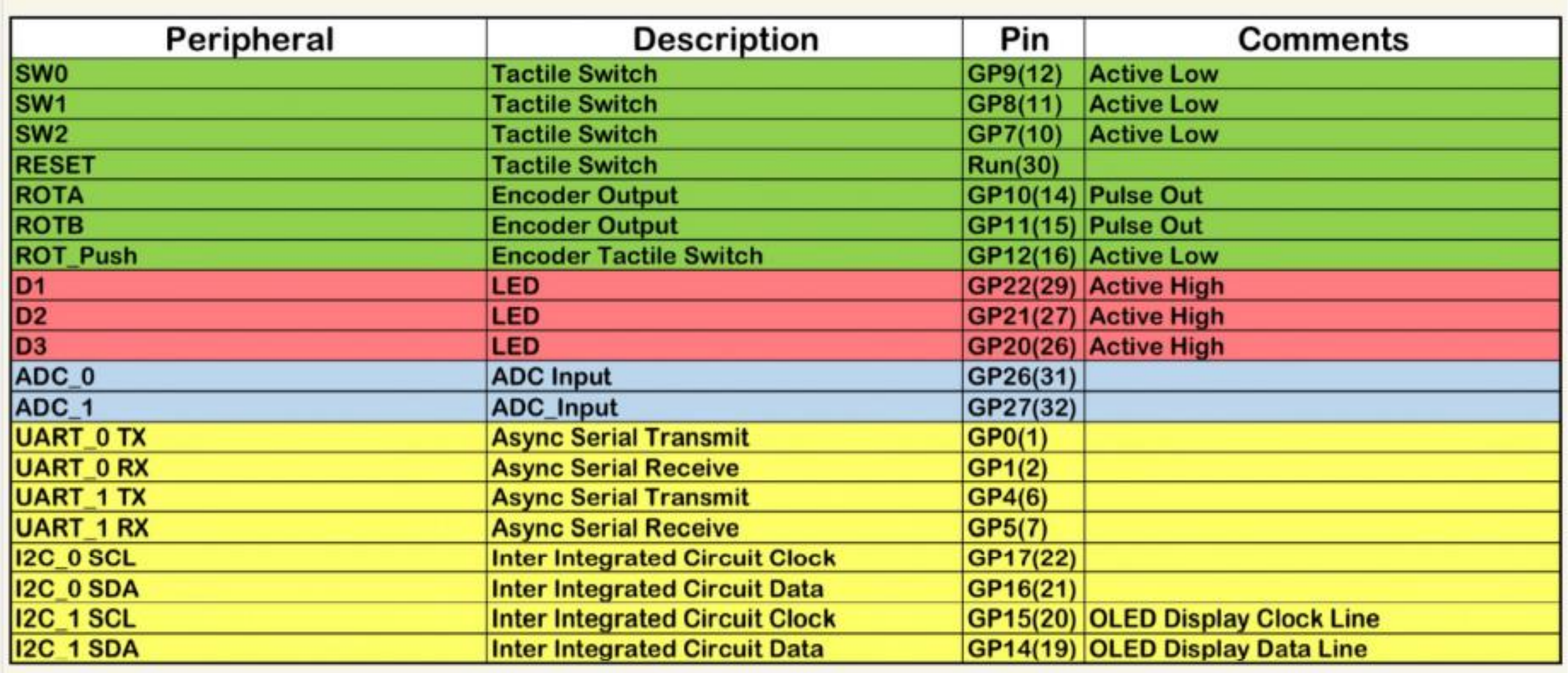
Metropolia

Development board



Pico W Dev Board Pinout.pdf

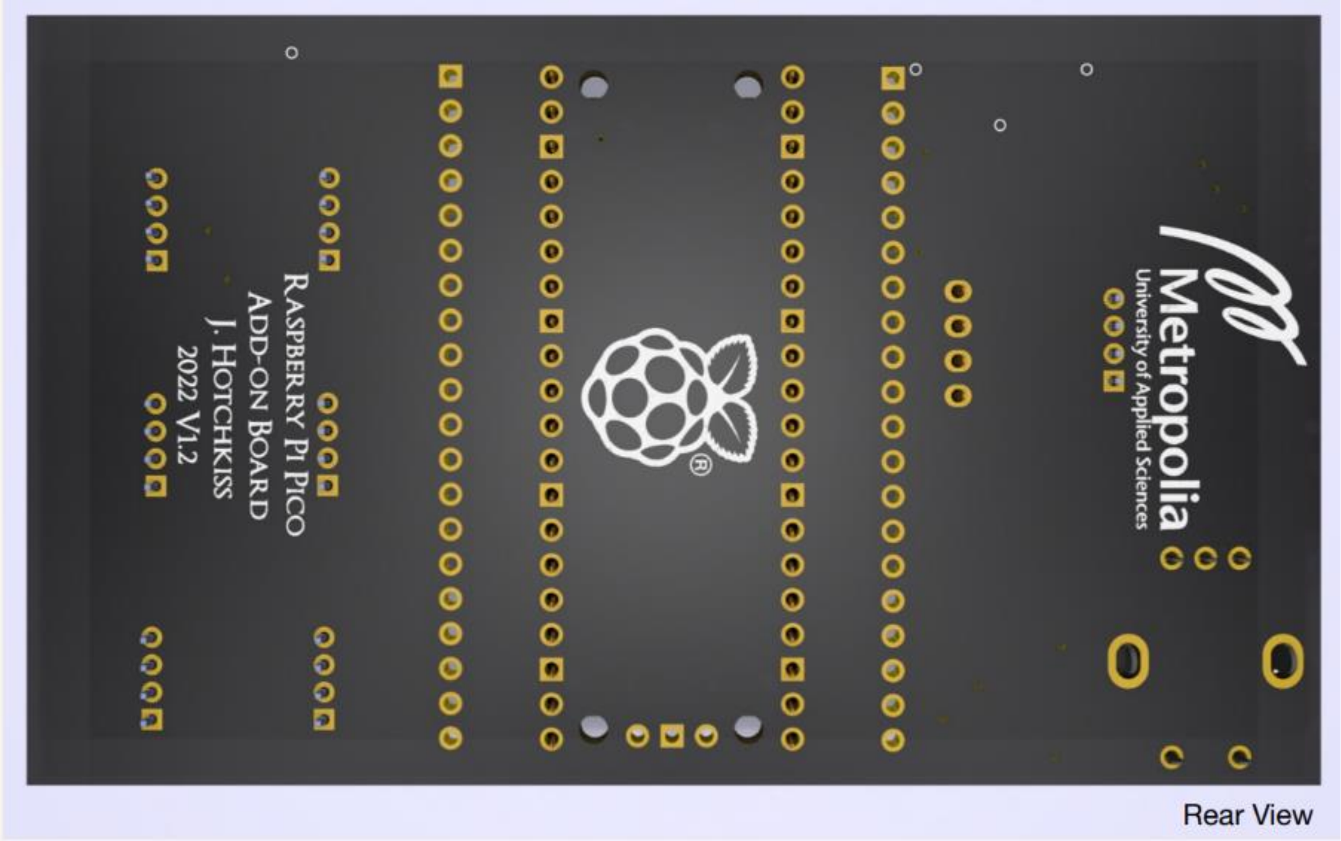
Development board pinout



Peripheral	Description	Pin	Comments
SW0	Tactile Switch	GP9(12)	Active Low
SW1	Tactile Switch	GP8(11)	Active Low
SW2	Tactile Switch	GP7(10)	Active Low
RESET	Tactile Switch	Run(30)	
ROTA	Encoder Output	GP10(14)	Pulse Out
ROTB	Encoder Output	GP11(15)	Pulse Out
ROT_Push	Encoder Tactile Switch	GP12(16)	Active Low
D1	LED	GP22(29)	Active High
D2	LED	GP21(27)	Active High
D3	LED	GP20(26)	Active High
ADC_0	ADC Input	GP26(31)	
ADC_1	ADC_Input	GP27(32)	
UART_0 TX	Async Serial Transmit	GP0(1)	
UART_0 RX	Async Serial Receive	GP1(2)	
UART_1 TX	Async Serial Transmit	GP4(6)	
UART_1 RX	Async Serial Receive	GP5(7)	
I2C_0 SCL	Inter Integrated Circuit Clock	GP17(22)	
I2C_0 SDA	Inter Integrated Circuit Data	GP16(21)	
I2C_1 SCL	Inter Integrated Circuit Clock	GP15(20)	OLED Display Clock Line
I2C_1 SDA	Inter Integrated Circuit Data	GP14(19)	OLED Display Data Line

Pico W Dev Board Pinout.pdf

Development board

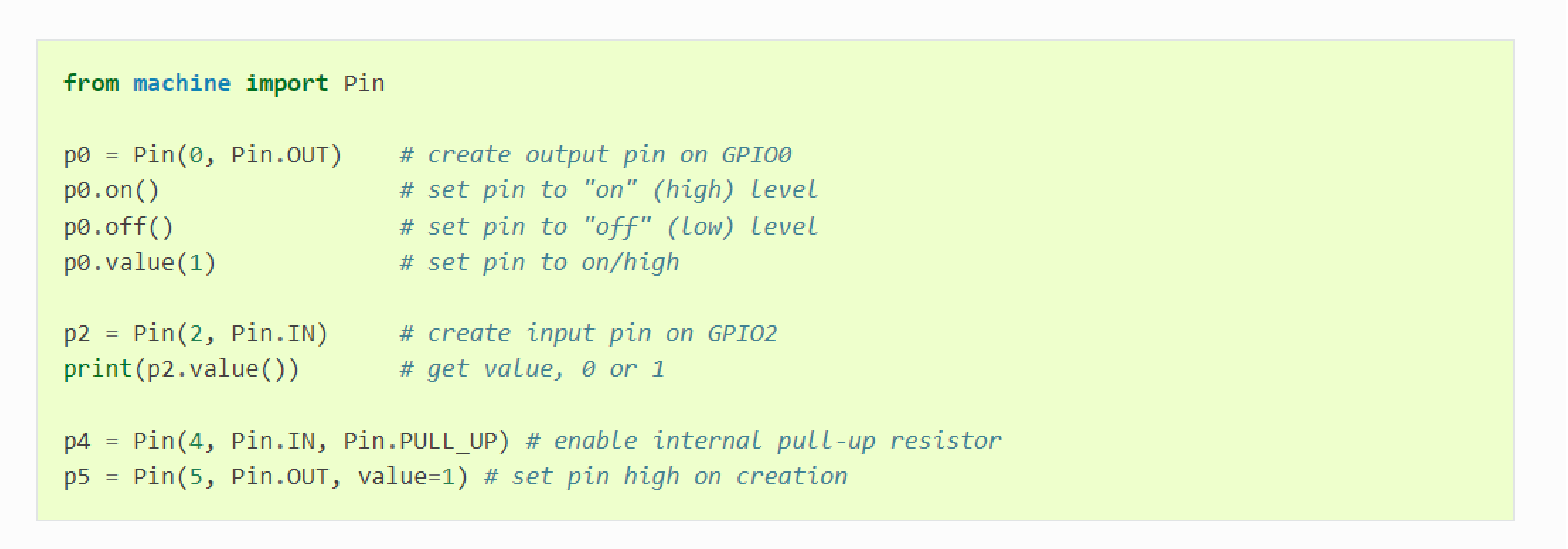


Pico W Dev Board Pinout.pdf

MicroPython and Pins

- A pin object is used to control I/O pins (also known as GPIO - general-purpose input/output).
- Pin objects are commonly associated with a physical pin that can drive an output voltage and read input voltages.
- The pin class has methods to set the mode of the pin (IN, OUT, etc) and methods to get and set the digital logic level.
- For the ability to input an analog signal to a pin, see the ADC class.

class Pin – control I/O pins — MicroPython documentation



```
from machine import Pin

p0 = Pin(0, Pin.OUT)    # create output pin on GPIO0
p0.on()                 # set pin to "on" (high) level
p0.off()                # set pin to "off" (low) level
p0.value(1)             # set pin to on/high

p2 = Pin(2, Pin.IN)     # create input pin on GPIO2
print(p2.value())       # get value, 0 or 1

p4 = Pin(4, Pin.IN, Pin.PULL_UP) # enable internal pull-up resistor
p5 = Pin(5, Pin.OUT, value=1) # set pin high on creation
```

Quick reference for the RP2 — MicroPython documentation

Structure and Function of GPIO pins

To understand GPIO pins some basic concepts need to be understood

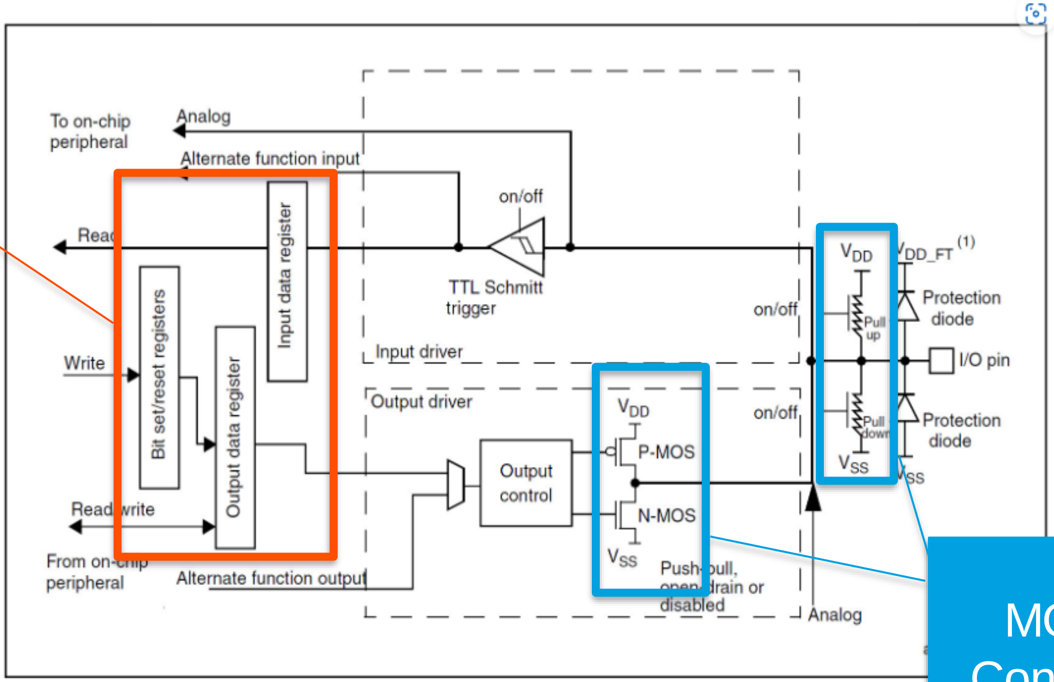
- GPIO Data Register
  - Special memory connected to CPU where data can be written to or read from (Binary 1s and 0s)

Logic 1 = VDD = 3.3V
Logic 0 = VSS = GND

- Transistor switch (MOSFET)
  - A device that can be operated as a switch by writing a 0 or 1 to it

GPIO System Internals

GPIO Registers



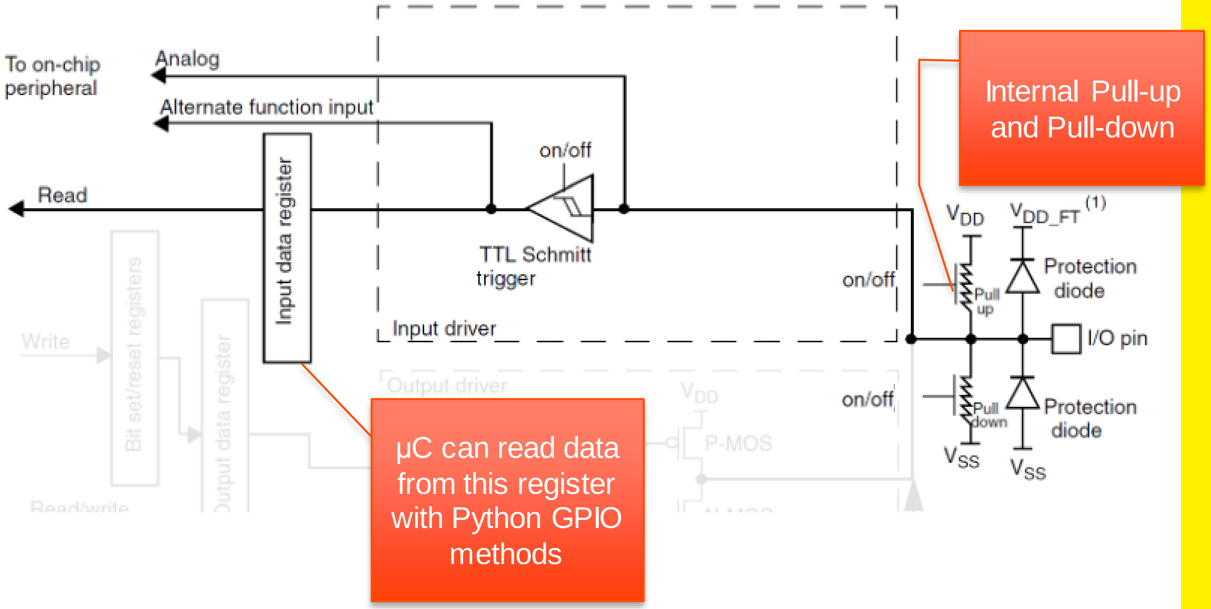
Typical structure of a GPIO pin. This is from an STM32F401 microcontroller.

MOSFETs Controlled by GPIO registers

All about GPIOs | Artekit Labs Tutorials

GPIO as Input

- Configuring GPIO as input allows for reading of incoming signals
- If 3.3V is applied to the pin then the register will contain a 1, and if the pin has 0V applied to it, the register will contain a zero



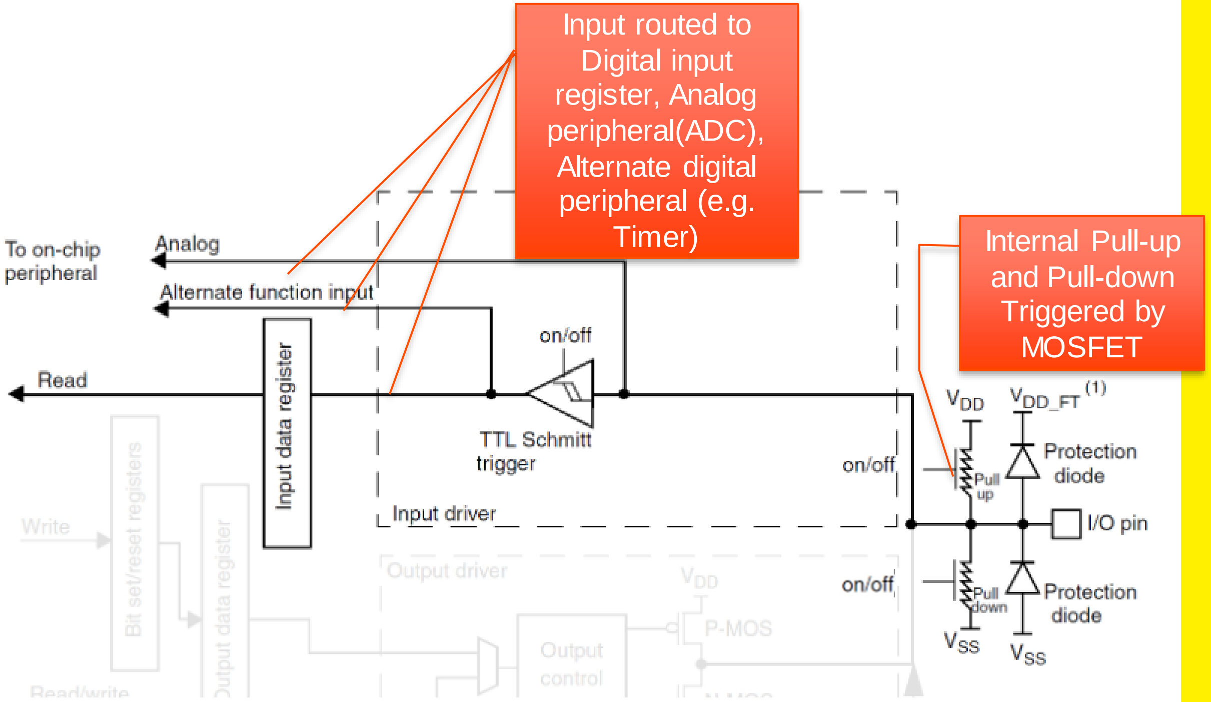
Internal Pull-up and Pull-down

µC can read data from this register with Python GPIO methods

All about GPIOs | Artekit Labs Tutorials

GPIO as Input

- Notice that the input signal is routed to multiple peripherals, this is why these are called general purpose
- The ability to enable and disable pull-up/down resistors by switching a transistor on and off is shown



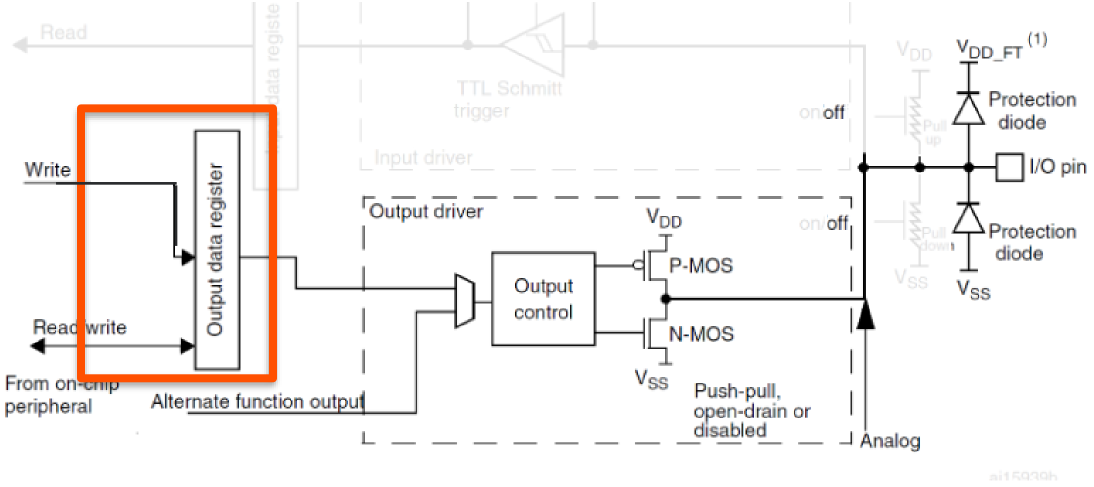
Input routed to Digital input register, Analog peripheral(ADC), Alternate digital peripheral (e.g. Timer)

Internal Pull-up and Pull-down Triggered by MOSFET

All about GPIOs | Artekit Labs Tutorials

GPIO as Output

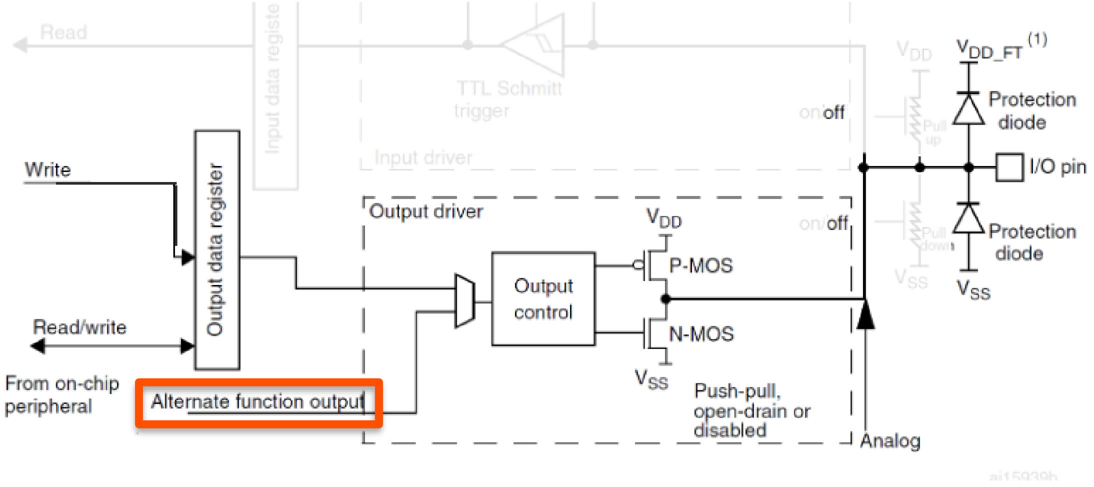
- Configuring GPIO as an output allows the µC to write a value to the output data register and have the pin connected to 3.3V or GND
- Careful care must be taken to not connect two output pins together, as this can easily cause a short between VDD and VSS



All about GPIOs | Artekit Labs Tutorials

GPIO as Output

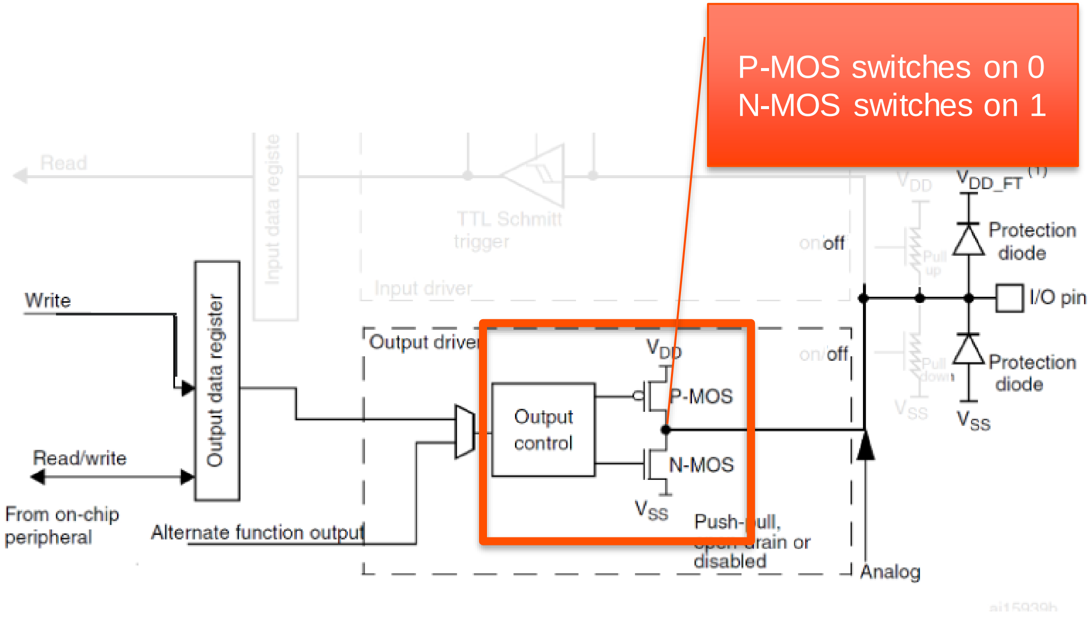
- As the name implies, general purpose output pins serve more than a single purpose
- Alternate GPIO functions can be routed to the GPIO output pins
- Pulse Width Modulation (PWM) is an example of a such an alternate function



All about GPIOs | Artekit Labs Tutorials

GPIO as Output

- Push-pull configuration
- When a value is written to the data register the Output control manages the P-MOS and N-MOS to either push the output pin high or pull it low

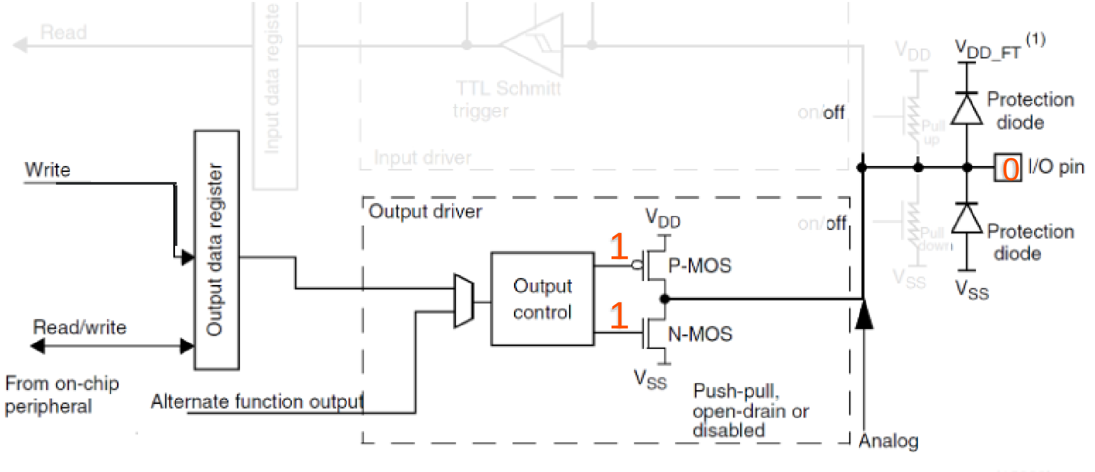


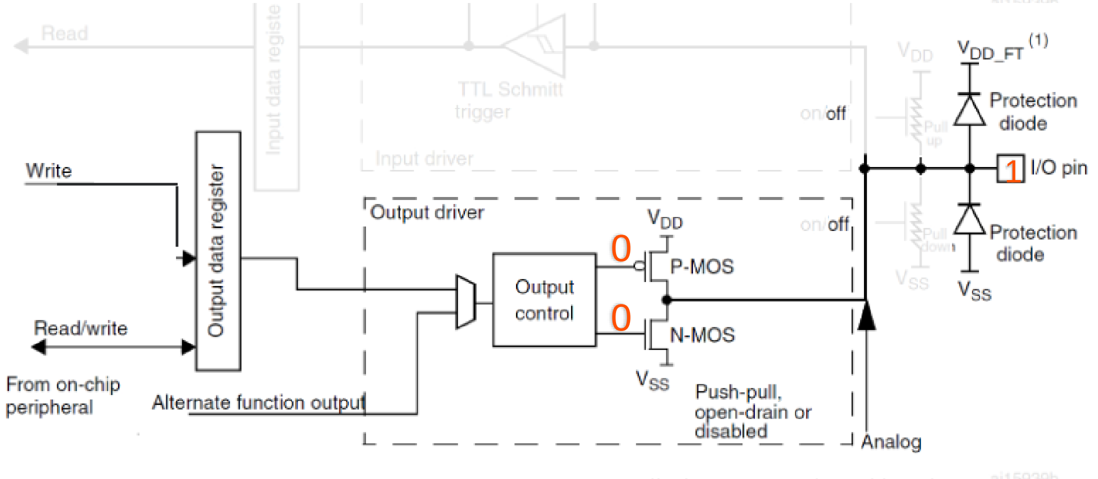
P-MOS switches on 0
N-MOS switches on 1

All about GPIOs | Artekit Labs Tutorials

GPIO as Output

- Push-pull configuration
- When a 1 is written to both MOSFETS the P-MOS will be off and the N-MOS will be on driving the pin to VSS (GND)
- When a one is written the opposite case is true and output is driven to VDD





All about GPIOs | Artekit Labs Tutorials

LEDs on Protoboard

A light-emitting diode (LED) is a semiconductor device that emits light when current flows through it.

- The hardware project protoboard has three (3) LEDs that are wired to GPIO pins.
- You can control the LEDs by configuring the pins as outputs and setting the pins high or low. Setting a pin high turns the LED on and setting the pin low turns the led off.
- You need to check the Pin numbers from the protoboard's design files.

Blink the onboard LED | Getting started with Raspberry Pi Pico

Pico W onboard LED

Raspberry Pi Pico W has one onboard LED. The onboard led is connected to a GPIO pin of the WiFi module instead of Pico's GPIO pin. However, you can still control it using MicroPython's Pin class. This highlights how object orientation simplifies programming: Pin class defines an interface that is common to all pins. The interface hides the details how the pin is actually accessed.

```
from machine import Pin
led = Pin("LED", Pin.Out)
led.toggle()
```

Light-emitting diode - Wikipedia

Exercise

BASIC

- Blink all LEDs, turn them on and off in sequence every second

EXTRA

- Turn the LEDs on and off smoothly using PWM.
- For more details, see Control LED brightness with PWM | Getting started with Raspberry Pi Pico.
- Note: You can't use PWM with the onboard LED

THANKS

QUESTIONS?