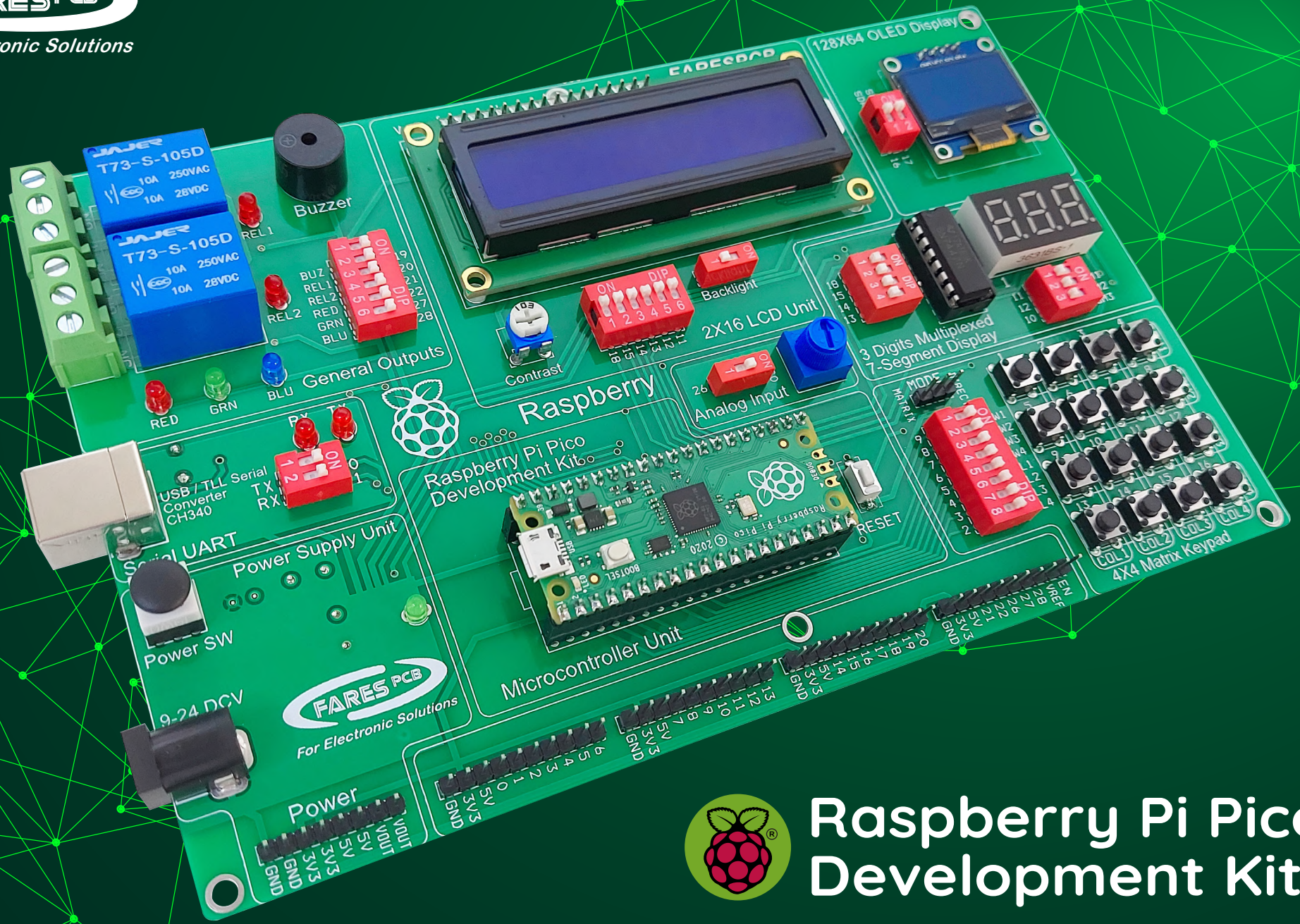
FARES PCB
For Electronic Solutions



Raspberry Pi Pico Development Kit

General Description

Raspberry Pi Pico development kit is a simple and powerful kit that contains many essential peripherals required to discover RPI Pico board capabilities and enable beginners to easily develop Pico-based projects. It comes with a wide variety of Input and output devices such as a keypad, analog input, LEDs, LCD, and more. In addition to USB virtual COM serial communication, It can be powered from a wide range of supply voltages (9-24V). All GPIOs and power rails are provided to pin header connectors to easily connect more advanced peripherals.

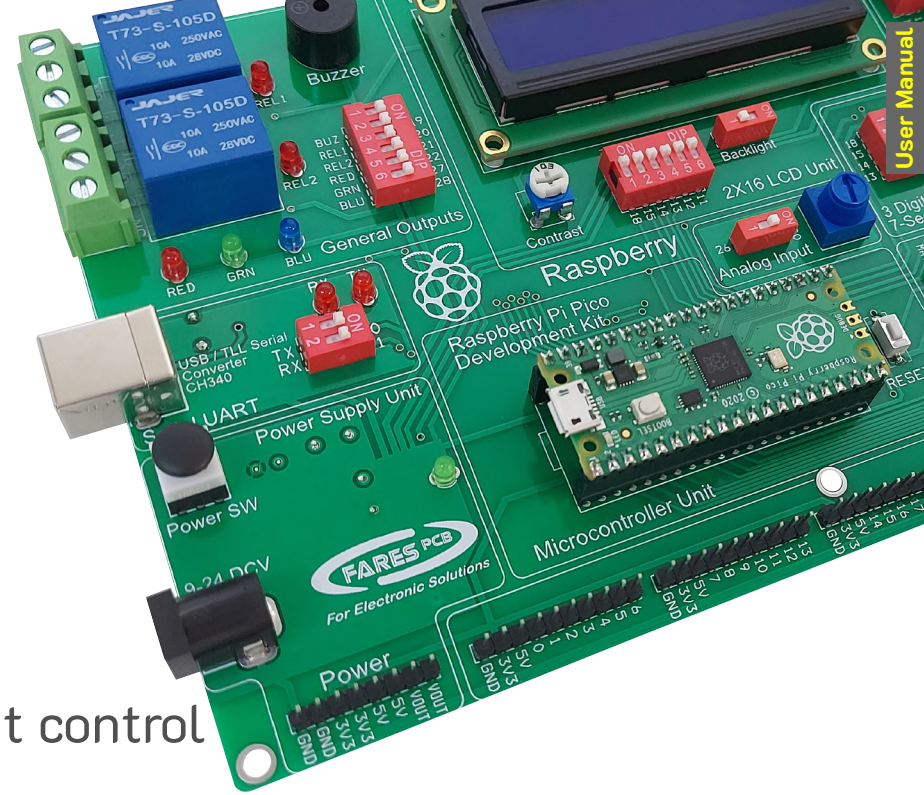
Raspberry Pi Pico Features

Core : Dual-core ARM Cortex M0+ processor
CPU frequency : Up to 133MHz
Memory : 264kB SRAM and 2MB on-board Flash memory
GPIOs : 26 multi-function GPIO pins
Analog : 3 X 16 Bit ADC
Peripherals : 2XUSART , 2XSPI , 2XI^2C , 16 PWM and USB 1.1 host and device support

This manual explains kit sections and shows how it can be programmed in brief.

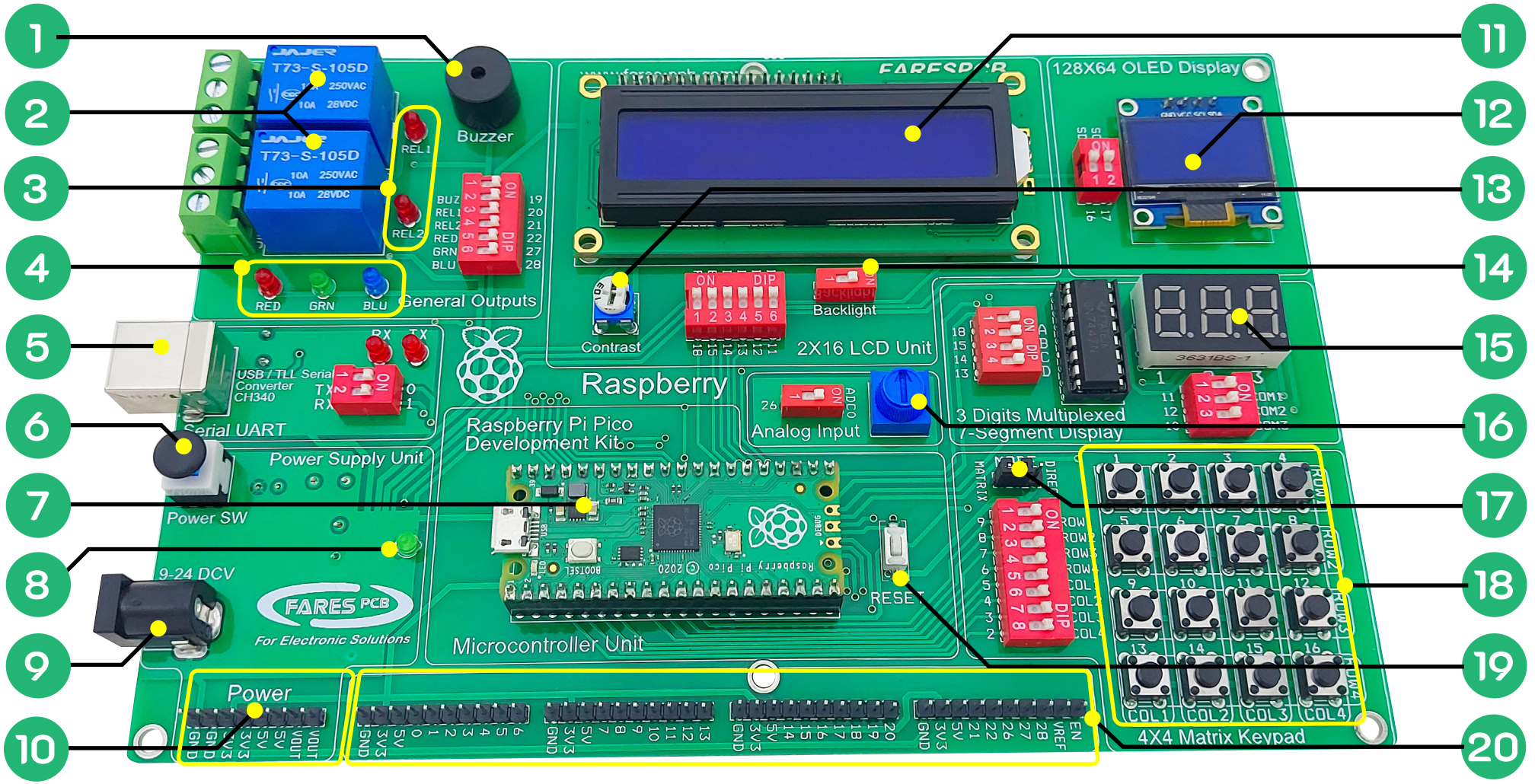
Pico Kit Features

- Supports both Pico and Pico W board
- USB or adaptor powered
- 16 Input switches
- Output unit involves:
    - Three output LEDs
    - Two output relays with LED indicators
    - One output buzzer
- 2X16 Character LCD with contrast and backlight control
- 128X64 I²C OLED display
- Three digits 7-segment display with BDC decoder
- Analog input
- USB/TTL serial converter CH340
- All GPIOs are brought out to pin header



FARES PCB

Kit Sections



FARES PCB

1 - 5V Buzzer

2 - 2X 10 Ampere relay

3 - LED indicator for relays

4 - Different color output LEDs

5 - USB type B serial interface

6 - Power switch

7 - Raspberry Pi Pico board

8 - Green LED for power indicator

9 - 2.1mm DC power socket

10 - Power over header pins

11 - 2X16 LCD character display

12 - 128X64 OLED display

13 - LCD contrast control

14 - Backlight on/off control

15 - Three digits 7-segment display

16 - 10 KΩ ceramic pot

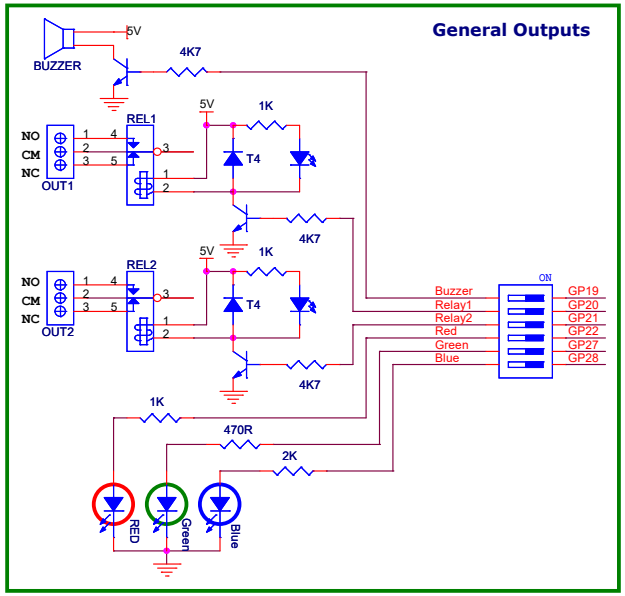
17 - Keypad mode select jumper

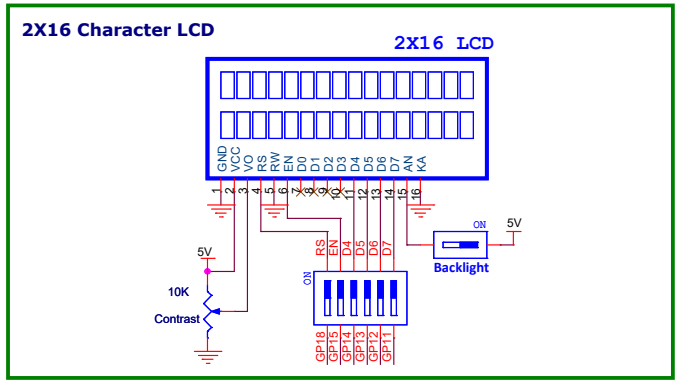
18 - 16 Switch keypad

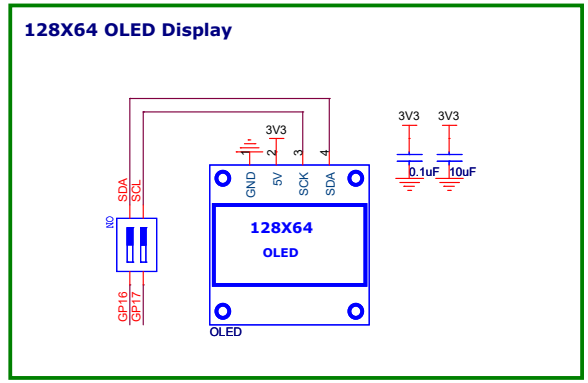
19 - RESET switch

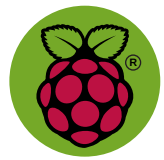
20 - GPIOs to pin header

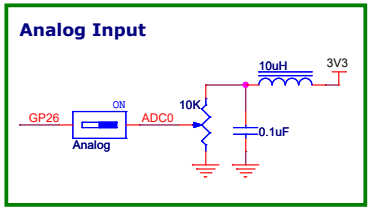
Schematic Diagram

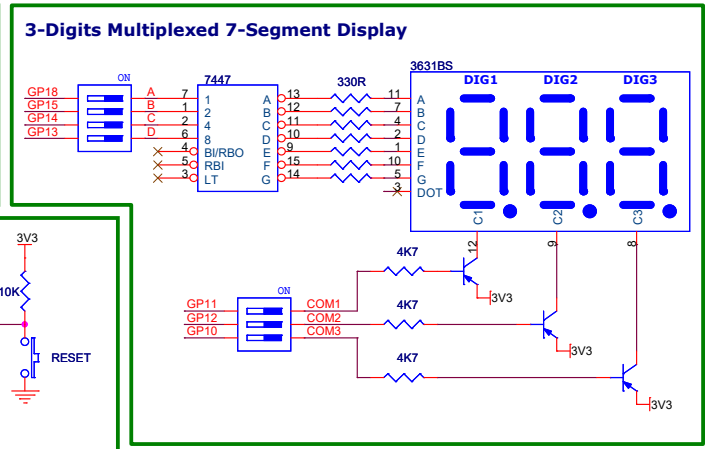


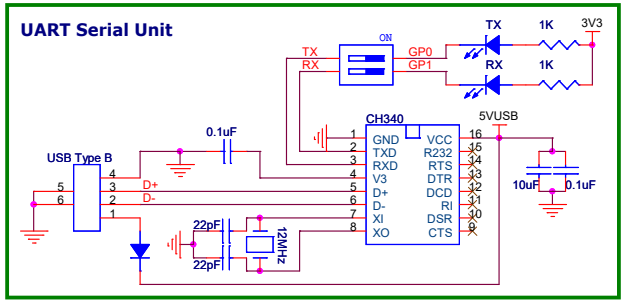


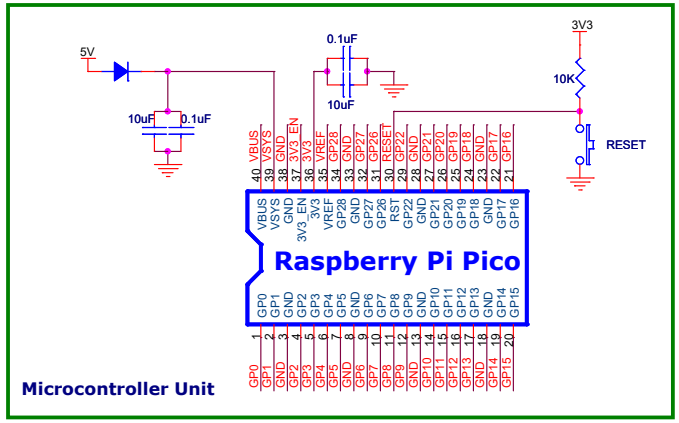


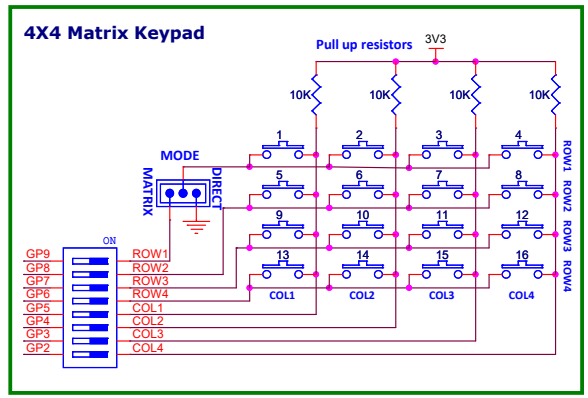


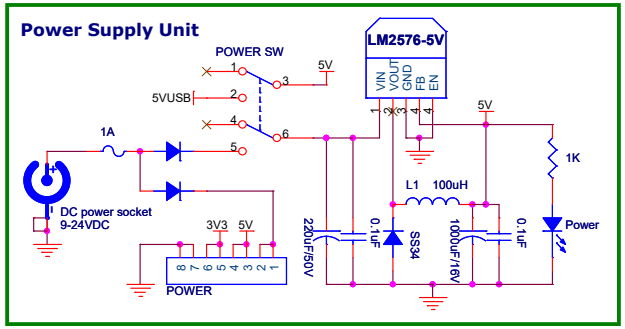


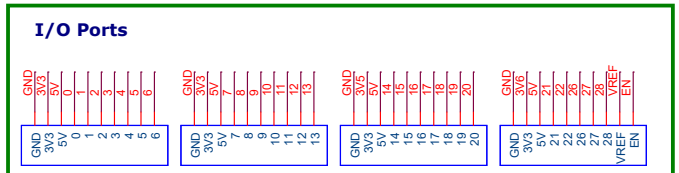






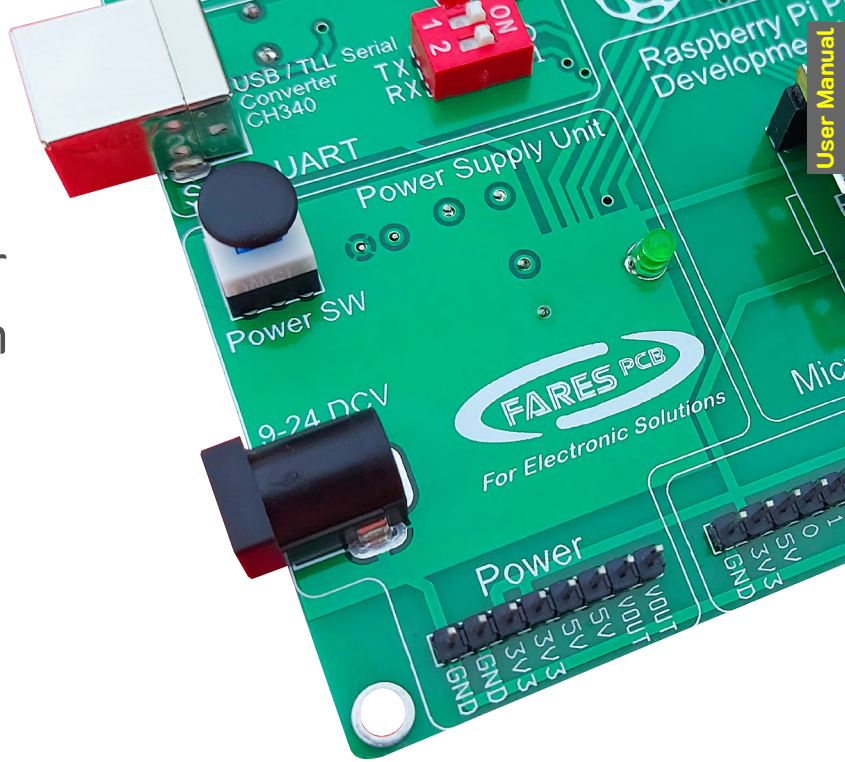




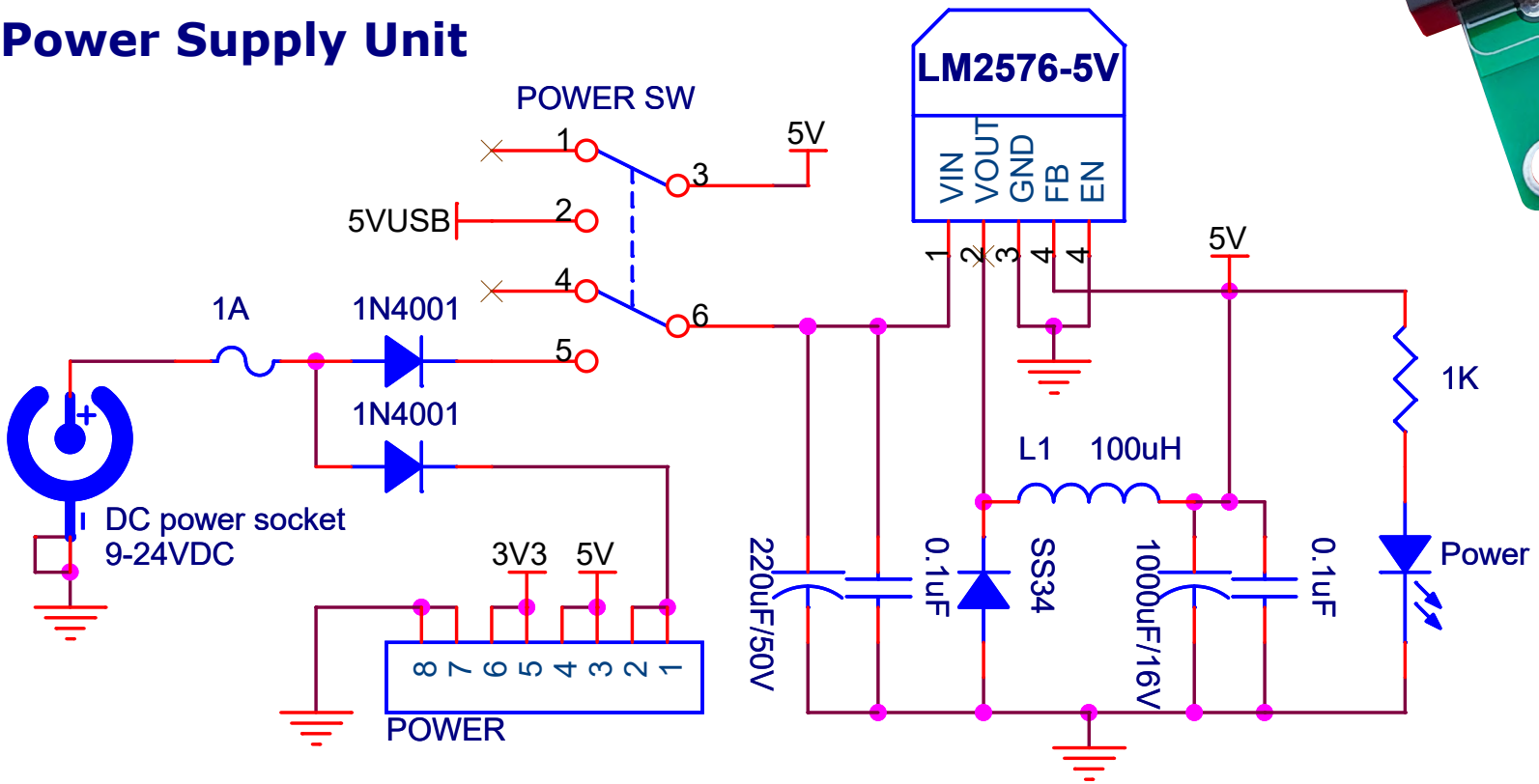


Power Supply Unit

Raspberry Pi Pico Kit can be powered from DC power supply adaptor (9-24V) via DC power socket or from 5V power source over USB type B connector.



Power Supply Unit



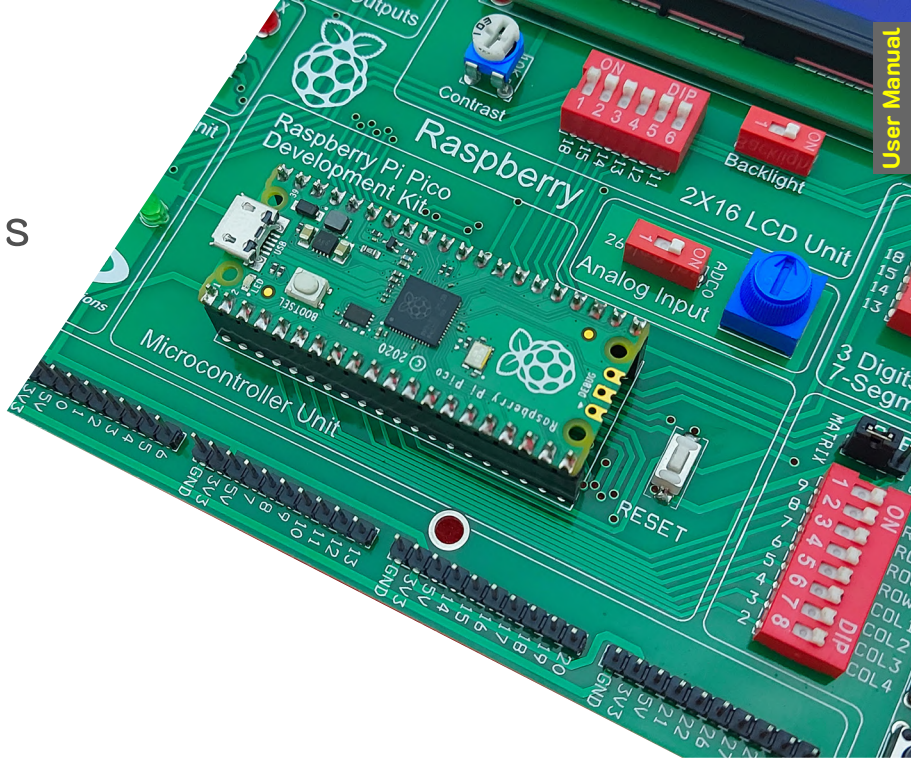
FARES PCB

PSU Specifications

- On/Off power switch
- Green LED for power indication
- 1A resettable fuse for over-current protection
- On-board 5V switch mode regulator LM2576-5V
- Input voltage and on-board generated voltages are available for external use via pin header
- On-board reverse current protection diode for safe operation

Microcontroller Unit

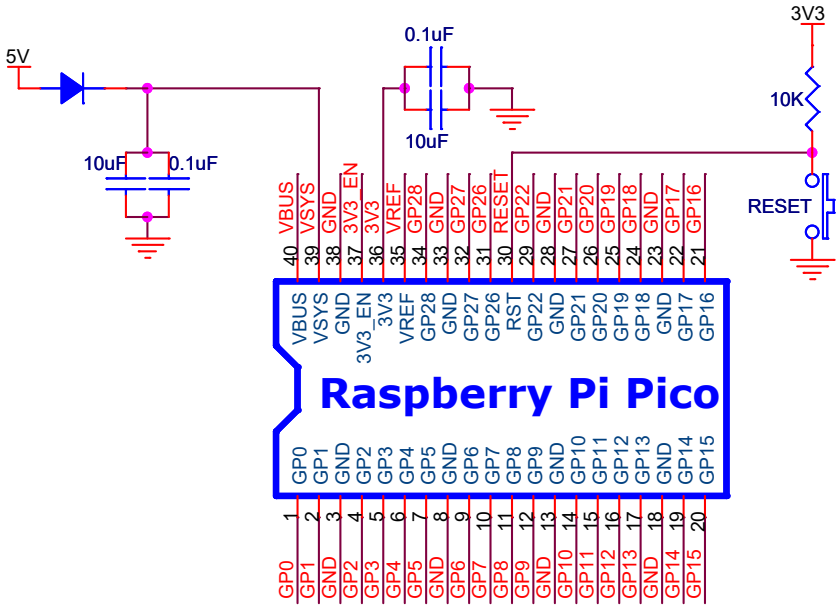
Pico Kit supports Raspberry Pi Pico board which is based on RP2040 microcontroller. It offers a dual-core ARM Cortex-M0+ microprocessor.



Pico Board Features

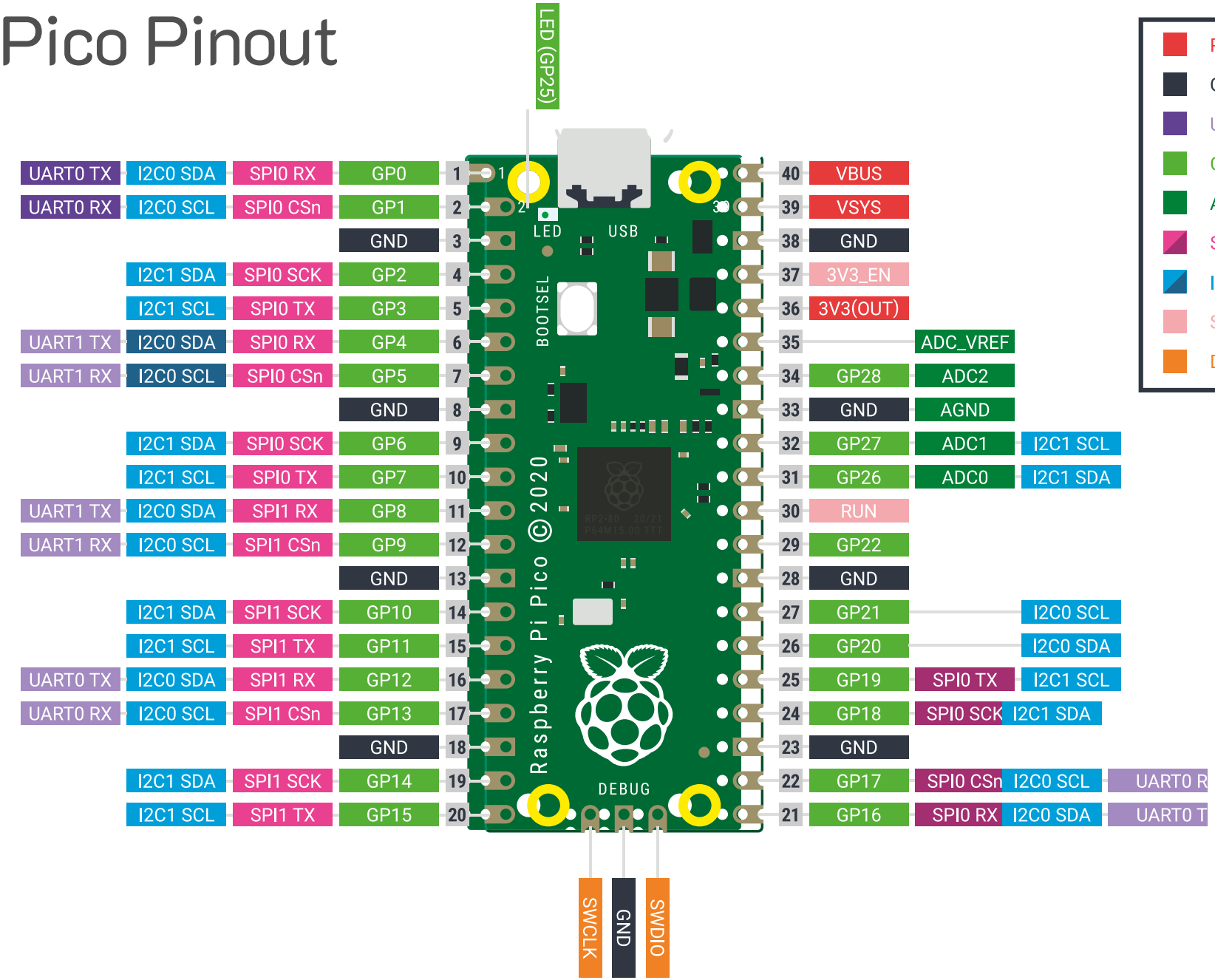
- RP2040 Dual-core ARM Cortex-M0+ (133MHz)
- 264KB SRAM and 2MB on-board flash memory
- USB type B micro connector
- On-board buck-boost generates 3.3V for RP2040
- On-board Boot Select switch
- Red LED connected to GPIO25

A RESET switch is included



FARES PCB

Pico Pinout



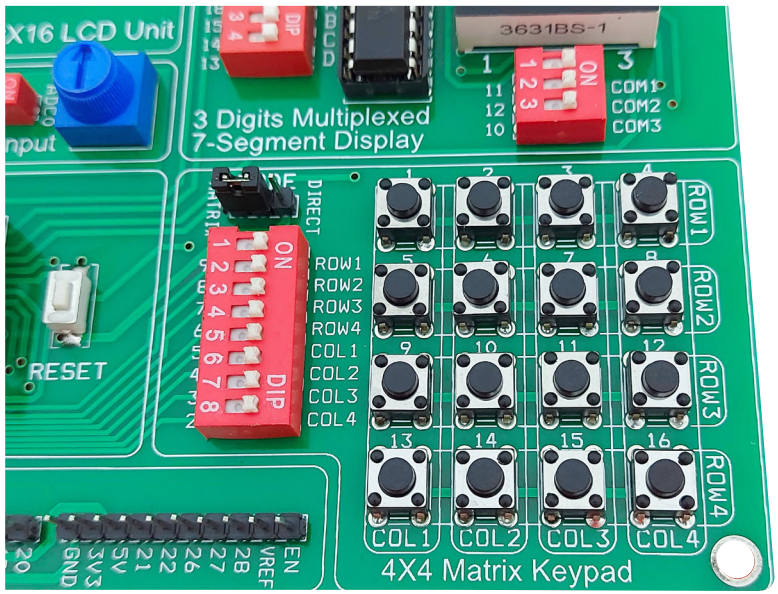
GPIOs mapping

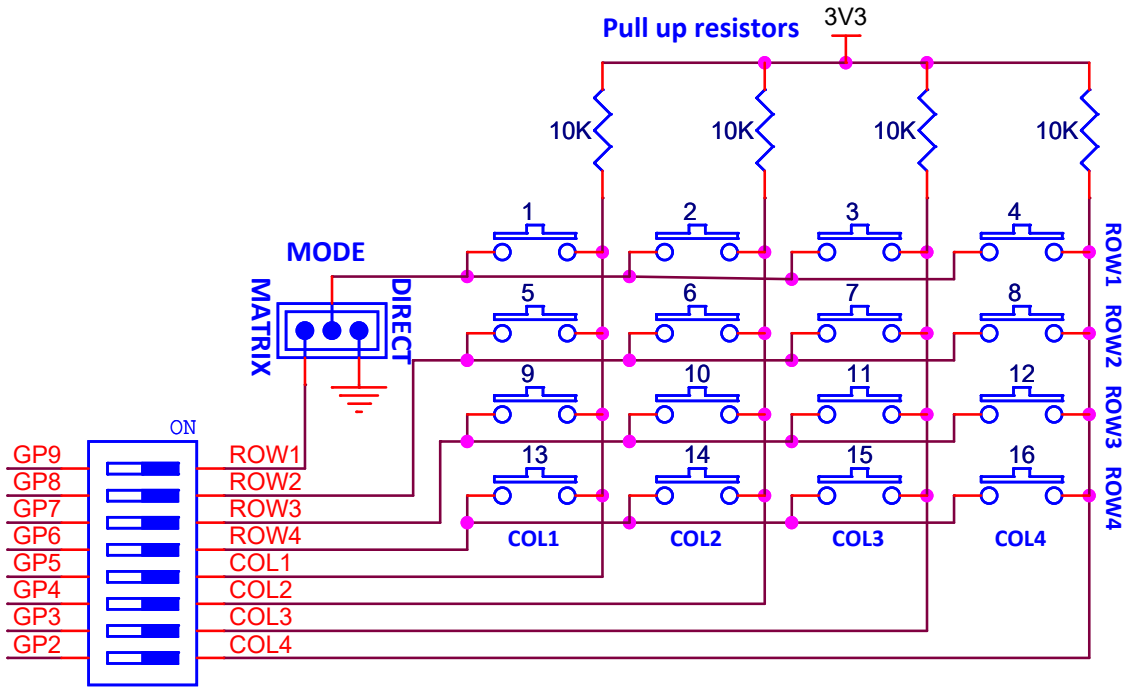
GPIO	Function
GP0	TX
GP1	RX
GP2	KB (COL4)
GP3	KB (COL3)
GP4	KB (COL2)
GP5	KB (COL1)
GP6	KB (ROW4)
GP7	KB (ROW3)
GP8	KB (ROW2)
GP9	KB (ROW1)
GP10	7-Seg (DIG3)
GP11	7-Seg (DIG1) / LCD (D7)
GP12	7-Seg (DIG2) / LCD (D6)

GPIO	Function
GP13	7-Seg (D) / LCD (D5)
GP14	7-Seg (C) / LCD (D4)
GP15	7-Seg (B) / LCD (EN)
GP16	I^2C (SDA)
GP17	I^2C (SCL)
GP18	7-Seg (A) / LCD (RS)
GP19	BUZZER
GP20	RELAY1
GP21	RELAY2
GP22	LED (RED)
GP26	ANALOG (ACD0)
GP27	LED (GREEN)
GP28	LED (BLUE)

Keypad Unit

Pico Kit contains 16 push-button switches configured as four rows intersected by four columns. Each intersection creates a switch position. Microcontroller can scan switches using one of two modes (direct mode or matrix mode).





Direct Mode

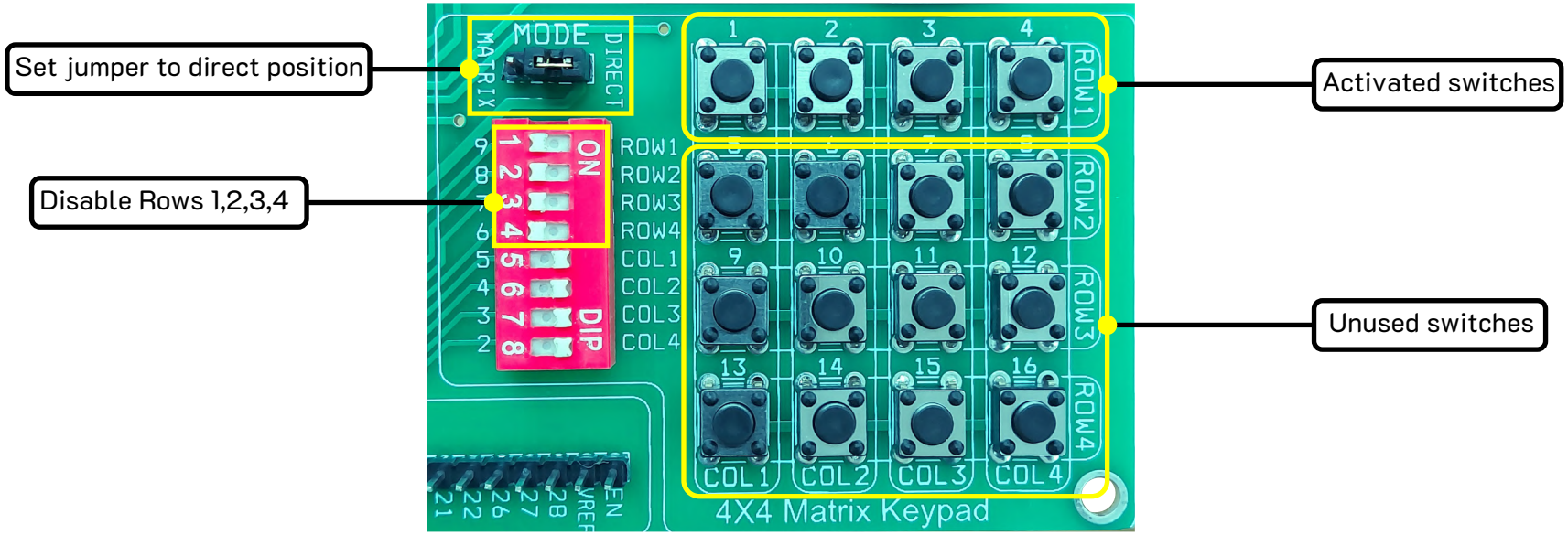
To configure keypad operation for direct mode, just set the MODE jumper to Direct position to assert a 0V level on the first row thus enabling switches 1,2,3 and 4. So that, these switches can be read directly as inputs through column lines that are pulled up by 10KΩ resistors. If a column line goes low, it means a switch is pressed otherwise, it is considered released.

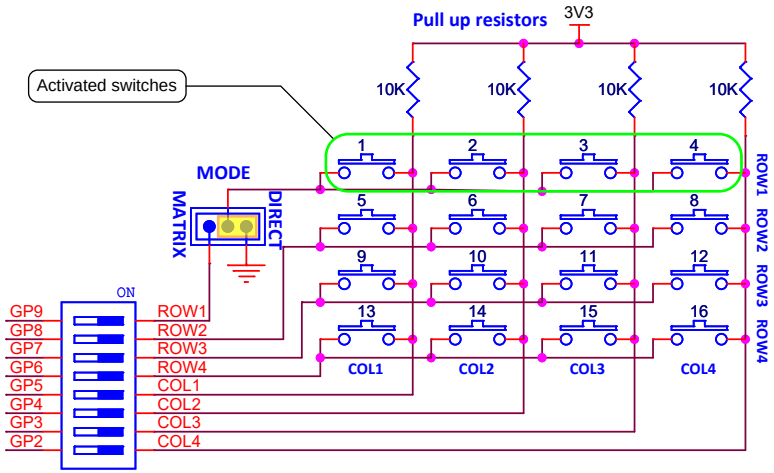
Direct mode is prefered to saves microcontroller I/O ports and reduces code size wherever no need for more than four switches.

Activated switches are connected to microcontroller as shown in table below:

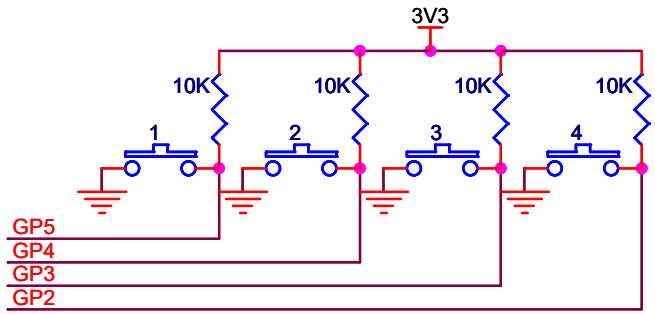
Switch	MCU pins
SW1	GP5
SW2	GP4
SW3	GP3
SW4	GP2

Direct mode configuration





Simplified keypad circuit diagram (direct mode)



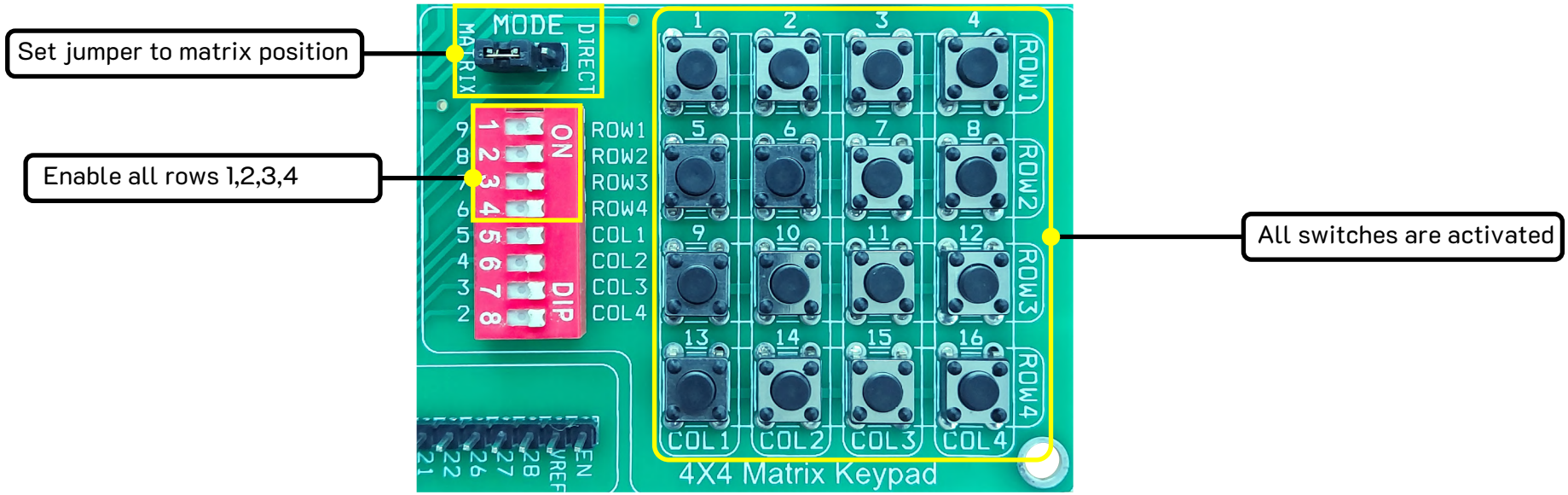
Matrix Mode

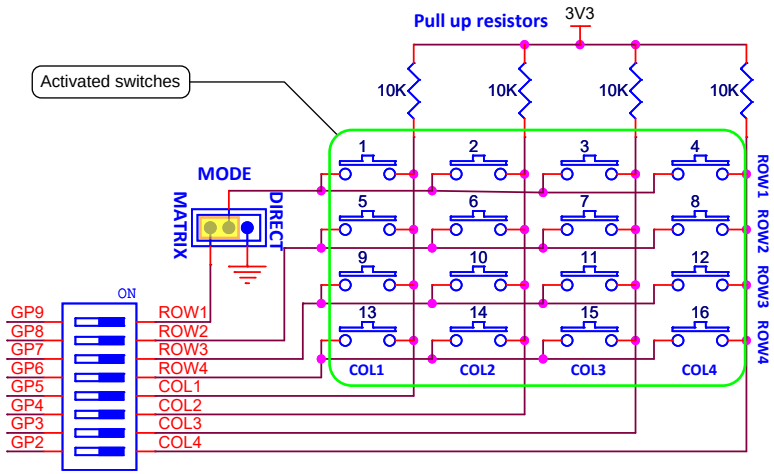
All 16 switches are activated and configured as four row lines and four column lines. Microcontroller scans these lines to detect a pressed button. Column lines are pulled up by 10KΩ resistors. (i.e. microcontroller port pin reads high if no switch is pressed). Scan operation starts by setting all rows and columns as inputs. To scan switches in a row, microcontroller configures it as output and initiates it to low, then checks columns one at a time. If a column line goes low, microcontroller detects a pressed switch otherwise, no pressed switch is detected in this row hence, it goes to scan next row and, so on.

Rows and columns are connected to microcontroller as shown in table

Switch	MCU pins
ROW1	GP9
ROW2	GP8
ROW3	GP7
ROW4	GP6
COL1	GP5
COL2	GP4
COL3	GP3
COL4	GP2

Matrix mode configuration

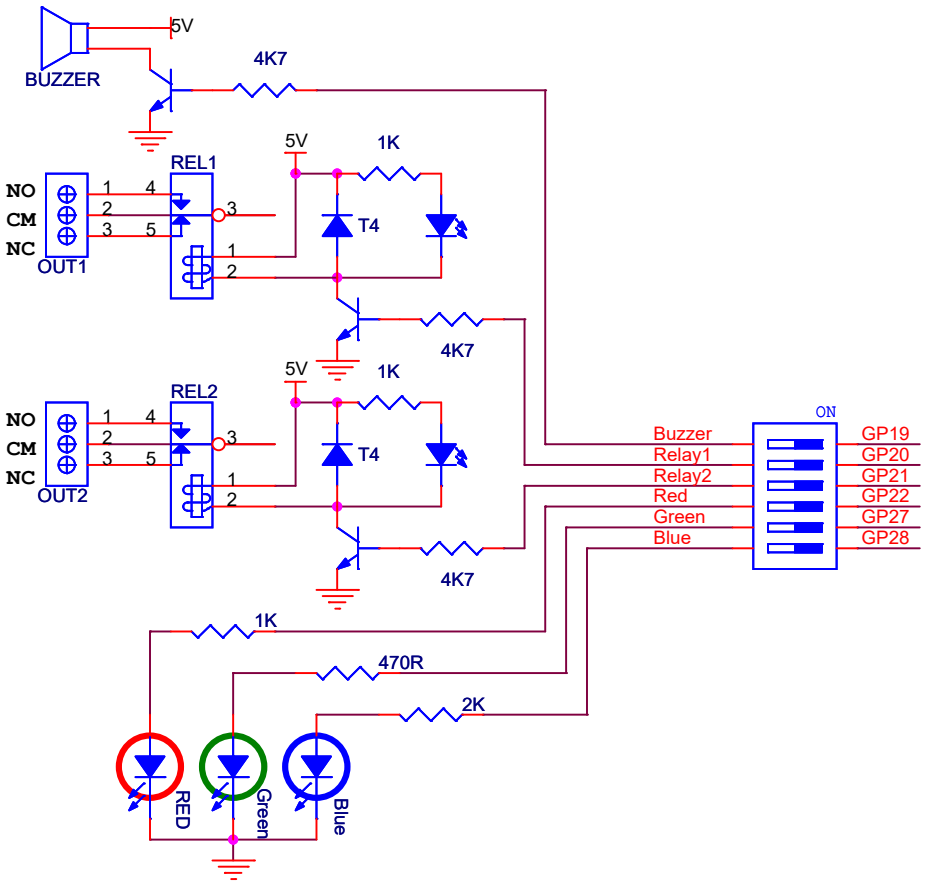


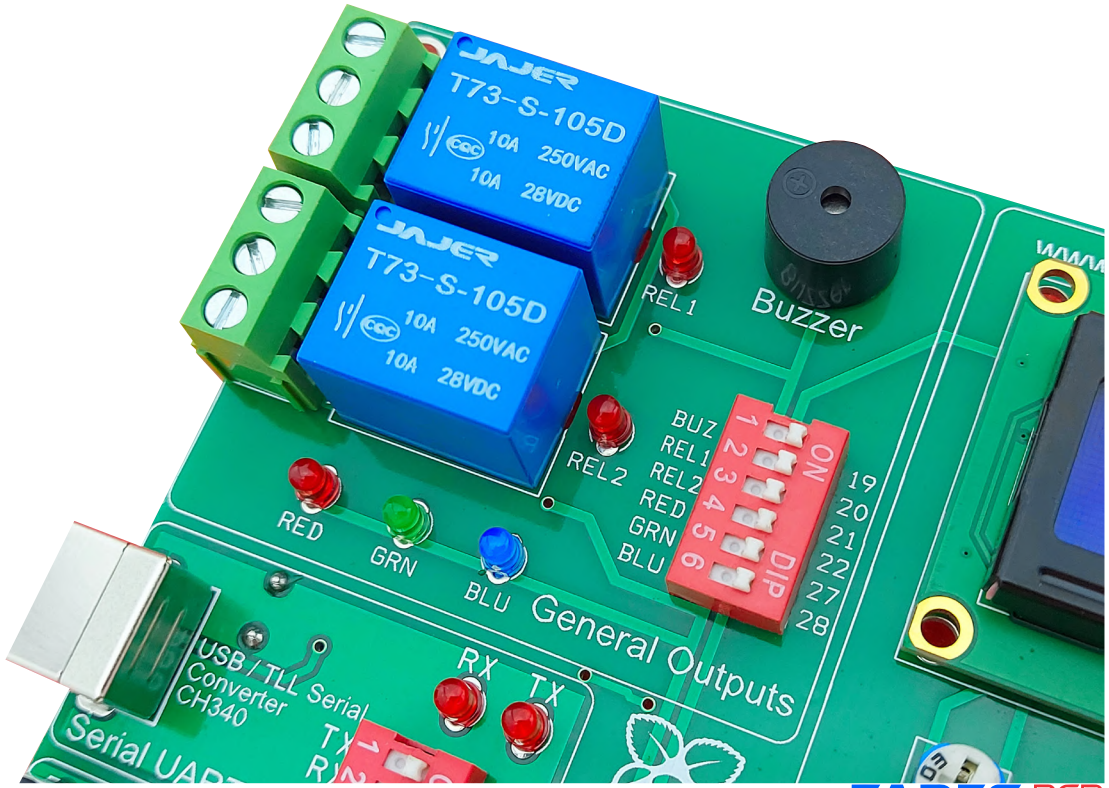


General Outputs

Pico kit provides a set of simple general outputs useful for beginners

- Three output LEDs; Red, Green, and Blue.
- Two output relays REL1 and REL2 up to 10A max current.
- One output Buzzer (5V).





Output LEDs

Three LEDs (Red, Green, and Blue) with current limiting resistors are connected to µC pins GP22, GP27 and GP28 respectively. LEDs are active high. i.e. output high turns LED on. Each LED can be individually enabled via a DIP switch.

Output Relays

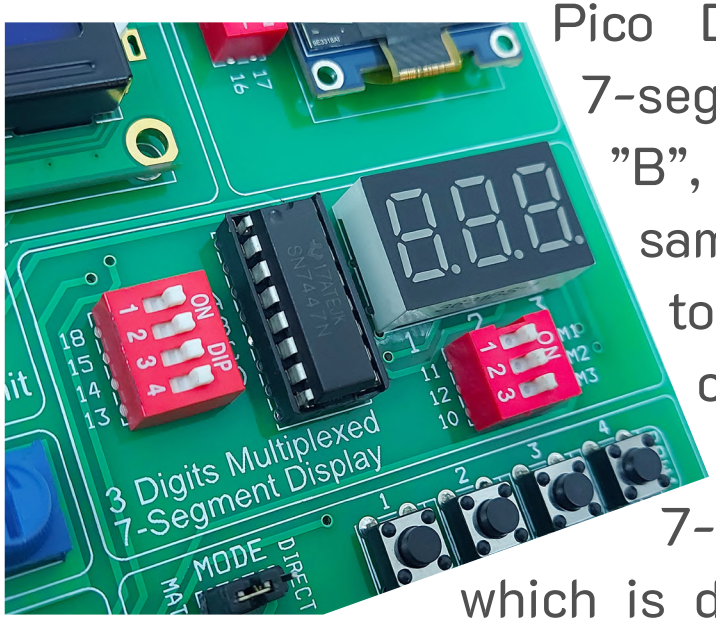
Pico kit is equipped with two output relays, which are suitable for AC or DC switching applications. Each relay has its own LED status indicator and can be individually enabled via a DIP switch. Relays are driven by NPN transistors. Freewheeling diodes are included to protect transistors from back EMF voltage that arises on the relay coil during switching off. Relay1 and Relay2 are driven by µC pins GP20 and GP21 respectively. Relays are 5V coil and rated up to 10A contacts (resistive load). Both normally open and normally closed contacts are brought out via screw terminals.

Output Buzzer

One output buzzer (5V) is driven by an NPN transistor and connected to µC pin GP19. Also, It can be enabled via a DIP switch.

7-Segment Display

7-segment diplay is very popular and has many applications. It is used to indicate numerical data. It can display digits from 0 to 9.



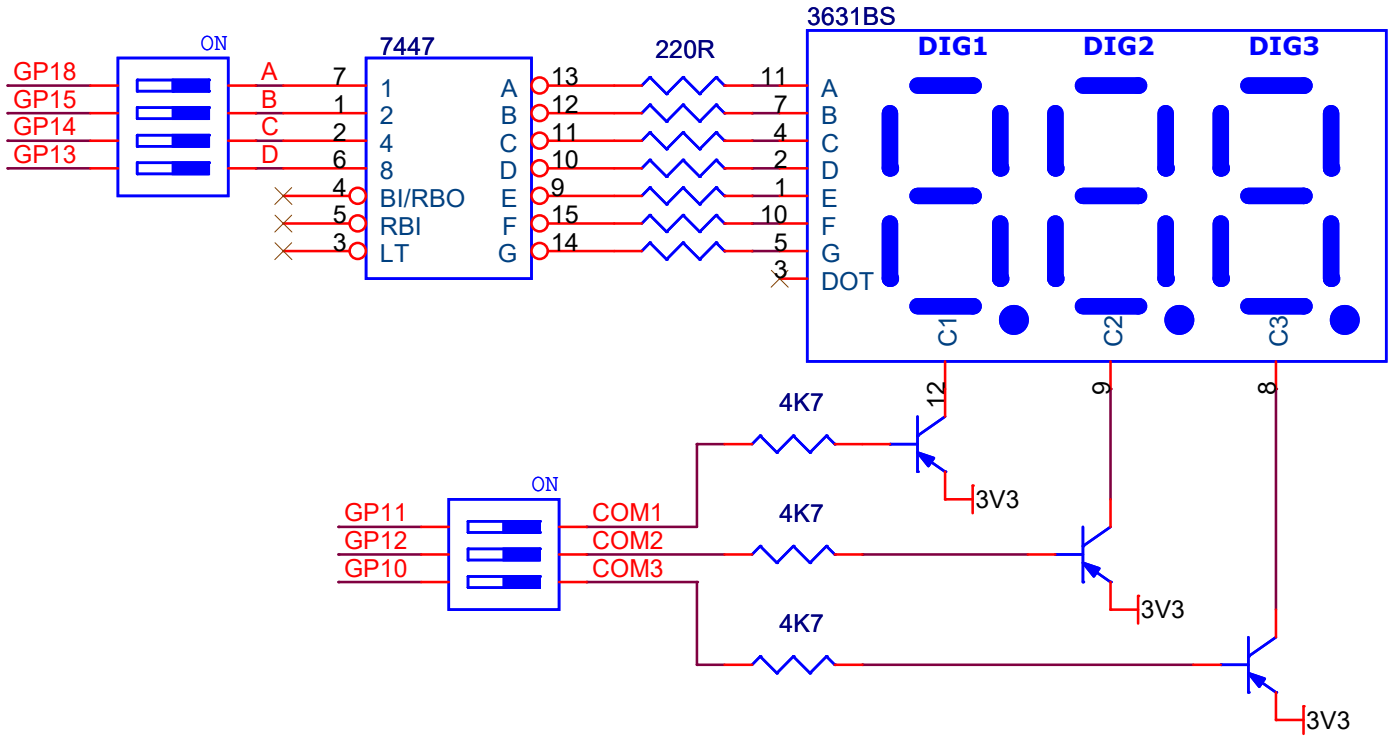
Pico Development Kit provides three multiplexed digits 7-segment display. Segments are referred to by letters “A”, ”B”, ”C”, ”D”, ”E”, ”F”, ”G”, and ”DOT”. All digits share the same segment. i.e. segment “A” of all digits are wired together and driven by the same output pin. 7-segment code is generated from a 7447 BCD decoder IC which converts the binary data received from µC into 7-segment code. Each digit has its own common anode which is driven by a PNP transistor. i.e. a low logic turns the transistor on, which in turn enables the digit.

Each 7seg digit can be enabled or disabled individually using three way DIP switch.

Data and commons are mapped to microcontroller as shown in tables below:

Data	MCU pin
A	GP18
B	GP15
C	GP14
D	GP13

Common	MCU pin
COM1	GP11
COM2	GP12
COM3	GP10



7-segment and LCD modules share the same µC port.
So, disable LCD before using 7-Segment display.

Binary data vs -7segment code generated by 7447 BCD decoder

Digit	Decoder Input				Decoder Output						
	D	C	B	A							
	Bit3	Bit2	Bit1	Bit0	A	B	C	D	E	F	G
0	0	0	0	0	0	0	0	0	0	0	1
1	0	0	0	1	1	0	0	1	1	1	1
2	0	0	1	0	0	0	1	0	0	1	0
3	0	0	1	1	0	0	0	0	1	1	0
4	0	1	0	0	1	0	0	1	1	0	0
5	0	1	1	1	0	1	0	0	1	0	0
6	0	1	1	0	0	1	0	0	0	0	0
7	0	1	0	1	0	0	0	1	1	1	1
8	1	0	0	0	0	0	0	0	0	0	0
9	1	0	1	1	0	0	0	0	1	0	0

2X16 LCD Display

LCD is a very popular electronic display module used in a wide range of applications such as calculators and printers.

It is inexpensive, simply programmable, and can be used for displaying some custom characters.

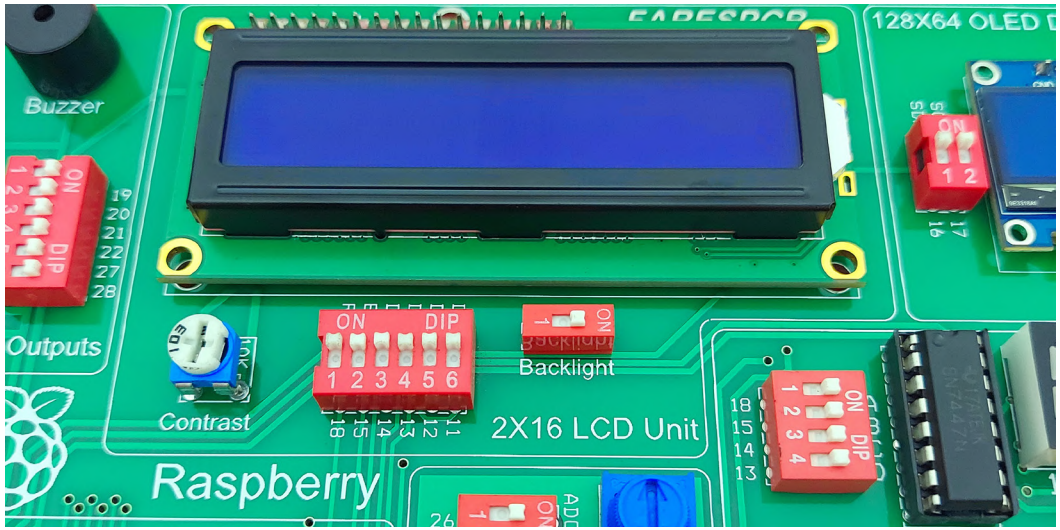
LCD Connector pinout is shown in table

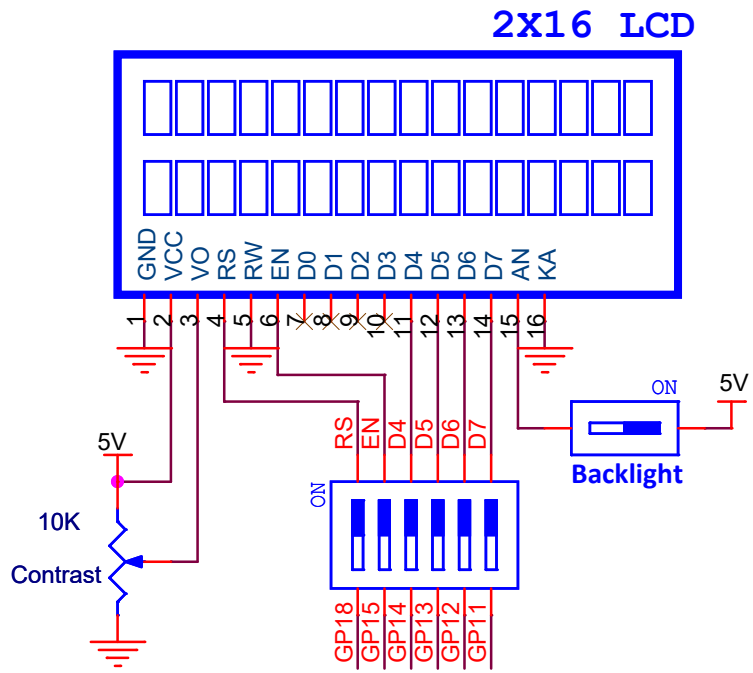
LCD Pin No	LCD Pin Name	Function
1	VSS	Ground
2	VCC	+5V
3	VO	Contrast
4	RS	Register Select
5	R/W	Read/Write
6	EN	Enable
7	D0	Data bit 0
8	D1	Data bit 1
9	D2	Data bit 2
10	D3	Data bit 3
11	D4	Data bit 4
12	D5	Data bit 5
13	D6	Data bit 6
14	D7	Data bit 7
15	A	Back light anode (+)
16	B	Back light cathode (-)

- LCD R/W control is tied to ground.
- 10KΩ potentiometer denoted by "Contrast" is used to adjust the LCD contrast.

LCD is connected to microcontroller as shown in table below:

LCD pin	MCU pin
RS	GP18
EN	GP15
D4	GP14
D5	GP13
D6	GP12
D7	GP11





LCD and 7segment modules share the same µC port. So, disable 7segment display (turn off COM1, COM2, and COM3) before using LCD.

OLED Display

OLED display is more efficient than other displays. It doesn't require backlight, which results in a high contrast in the dark. Additionally, its pixels consume energy only when they are on, so the OLED display consumes less power.

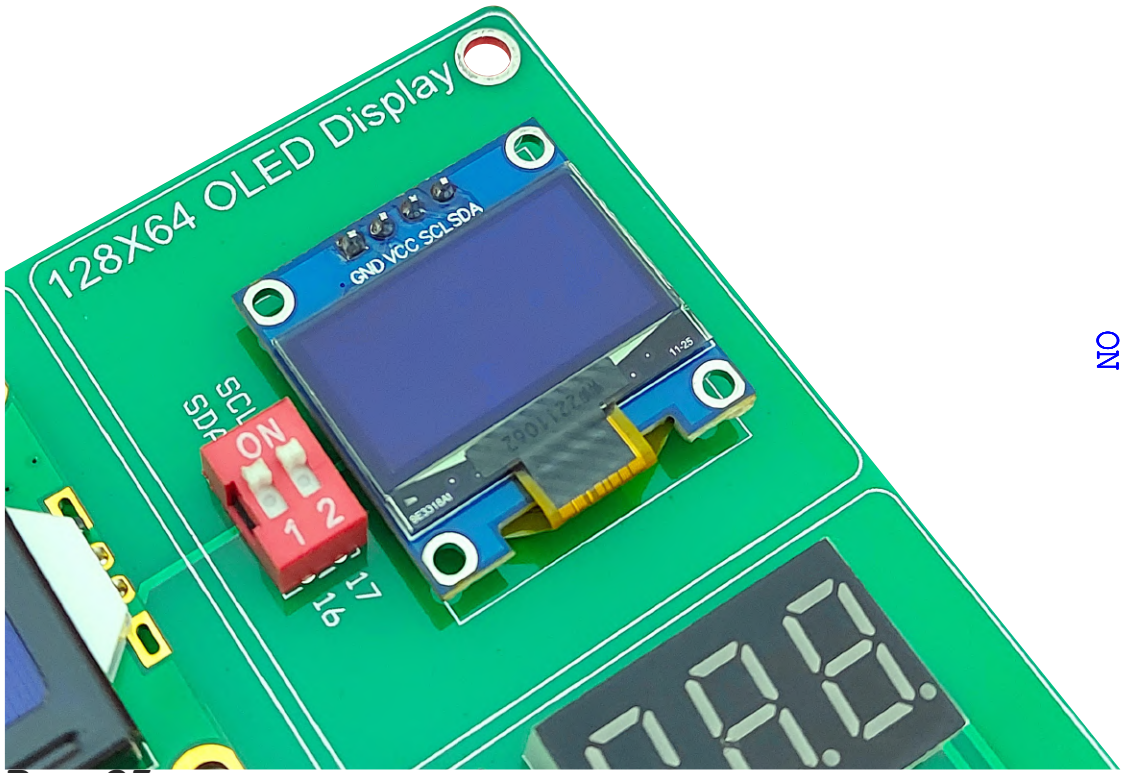
PICO Kit provides 128X64 monochrome 0.96 inch OLED display (SSD1306). It has only four pins and communicates with microcontroller using I^2C protocol. Operating voltage ranges from 3.3V to 5V.

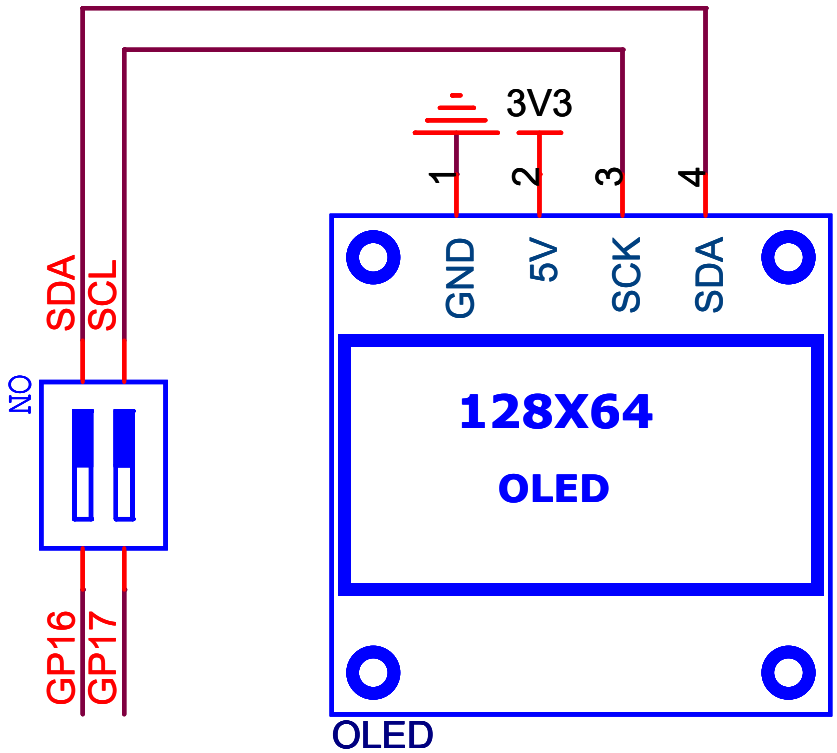
I^2C bus needs pull-up resistors both on SDA and SCL lines. OLED display has its own 10K ohm internal pull-up resistors (Lines pulled up to 3.3V). So, There is no need for using external pull-up resistors.

OLED display is connected to microcontroller as shown in table below

OLED pin	MCU pin
SCL	GP17
SDA	GP16

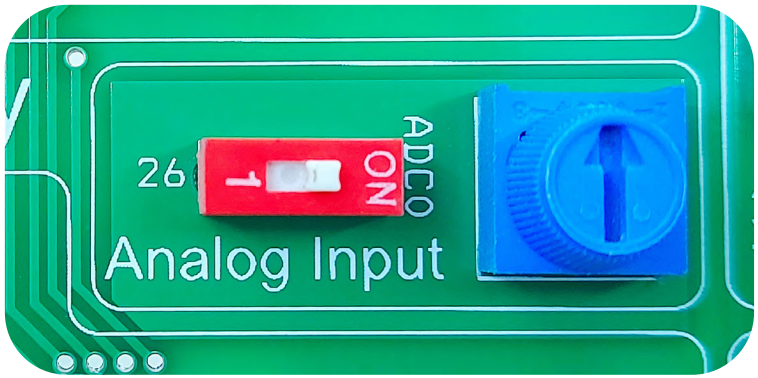
OLED Diplay module pull up the I²C bus lines to 3.3V through internal 10K ohm resistors. If OLED is disconnected (DIP switch is turned off) then external pull-up resistors are required to use I²C interface.

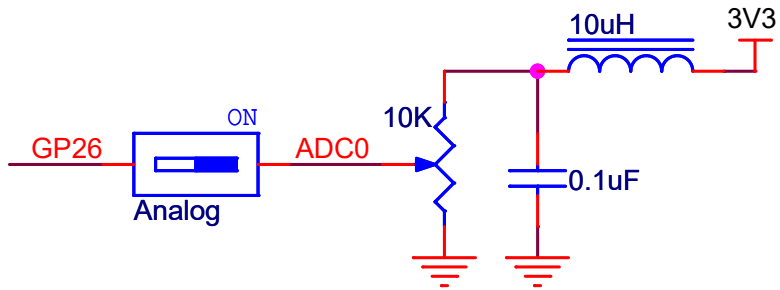




Analog Input

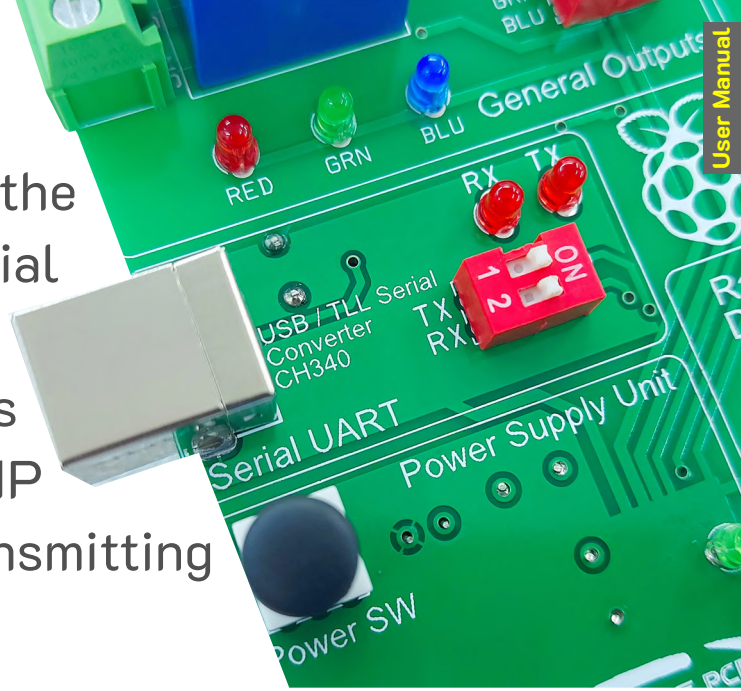
Pico kit provides a variable analog input 0-3.3V through a 10Kohm high precision potentiometer. Fixed terminals are connected to the power rails (GND and 3.3VDC) through an LC filter to eliminate power supply noise, and the variable terminal is routed to µC pin GP26 via a DIP switch. The variable resistor can be precisely adjusted to a potential value in the range from 0.00V to 3.30V.

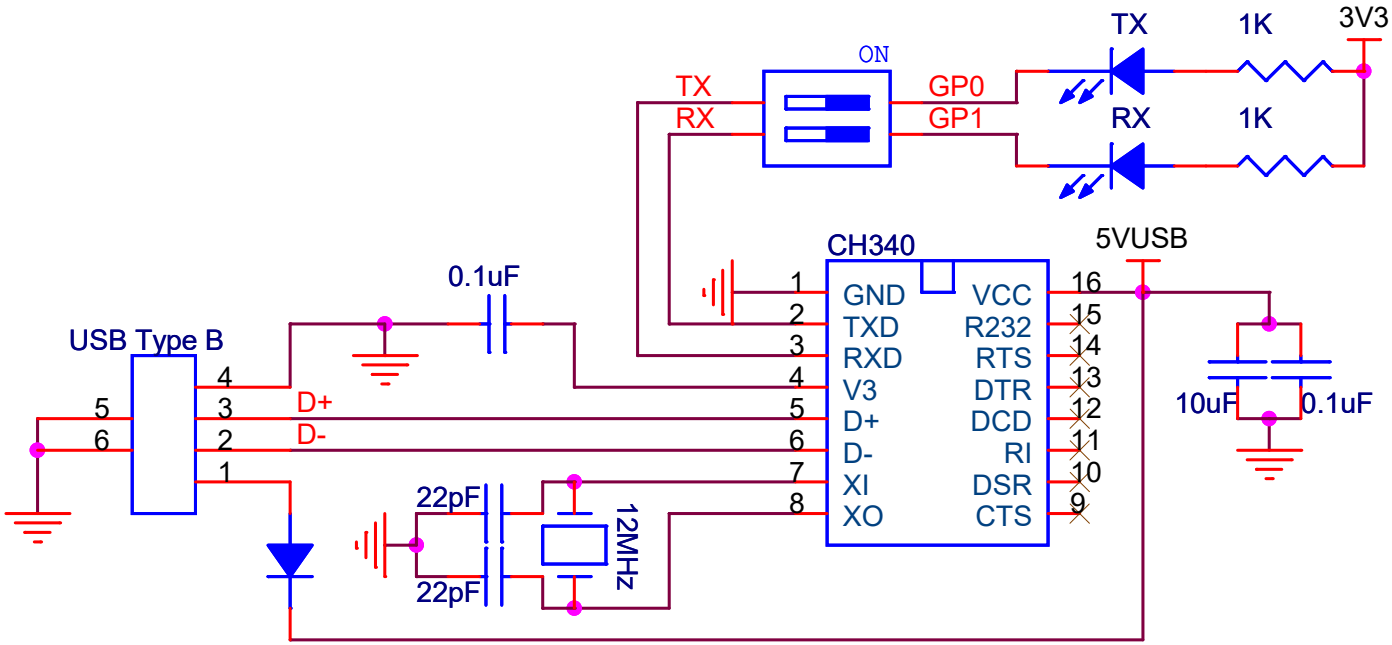




USB Virtual Serial COM

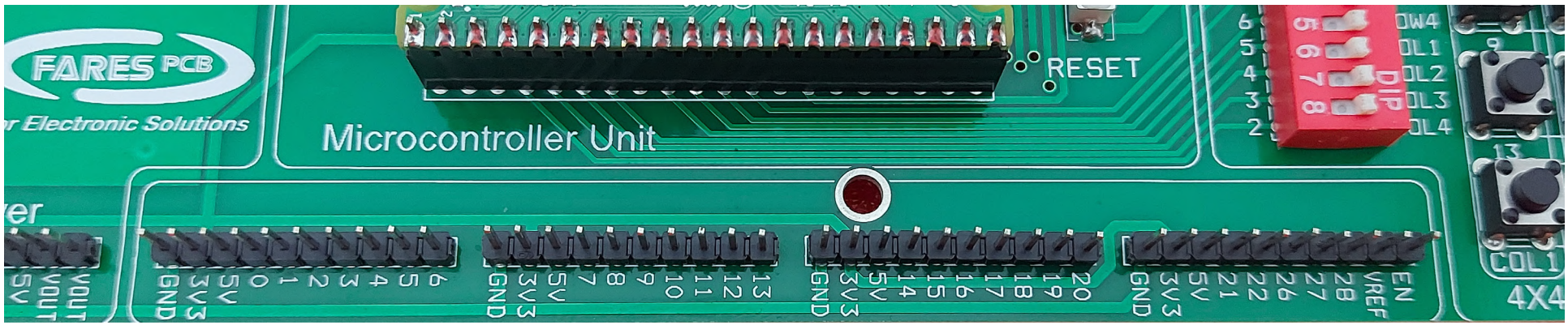
Pico kit can be interfaced with the host computer over the USB-B connector. This unit contains a USB to serial converter (CH340). CH340 is connected to RX, TX of the microcontroller internal UART peripheral through pins GP1 and GP0, which can be enabled individually via a DIP switch. Two red LEDs are included to indicate serial transmitting and receiving activities.





I/O Ports

Pico kit offers interfacing with external devices and peripherals. All GPIOs available in the Raspberry Pi Pico board are brought out for external use via headers. They are positioned at the lower side of the kit, so they can be easily accessed. The pins on these headers are labeled according to the GPIO number of the µC to which they are routed. Pin header connectors are grouped into three ports. Each represents one microcontroller port, in addition to power rails 5V, 3.3V, and GND to supply external circuits. Pin headers can be used to expand the connectivity of your **Pico** kit.



How to Program Pico Development Kit

Pico kit can be programmed using Python or C\C++ language.

Programming Pico using Python

To program Raspberry Pi Pico using Python you need to load MicroPython on Pico. MicroPython is a tiny Python interpreter that runs on a small embedded development board such as Pico board.
MicroPython is full-featured and supports most of Python's syntax. The simplicity of the Python programming language makes MicroPython a good choice for beginners who are new to programming and hardware.

How to load MicroPython on Pico

1 - Open the link below

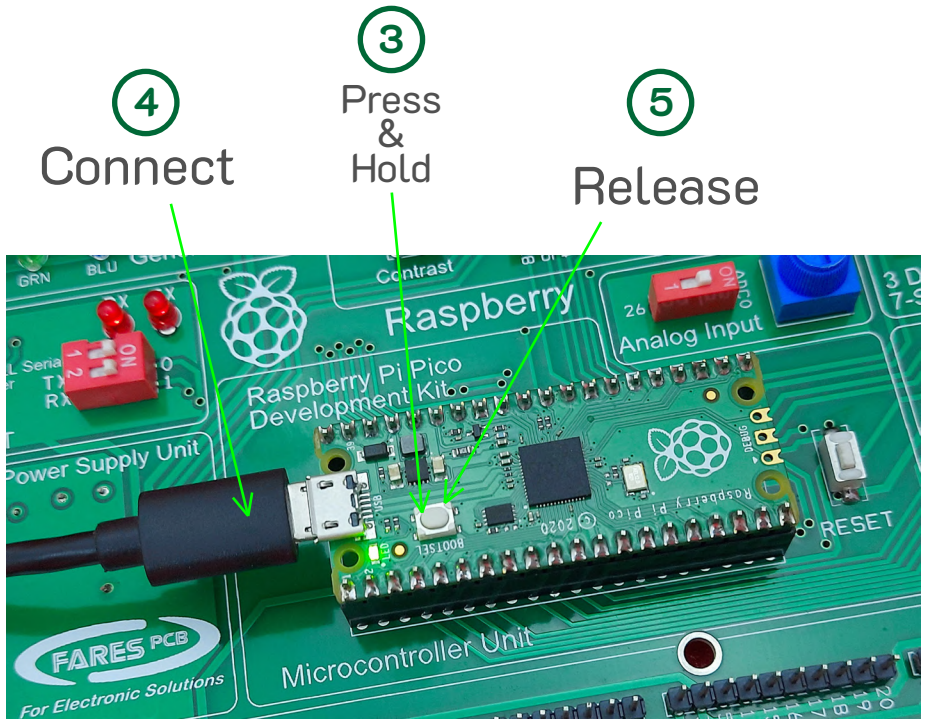
https://micropython.org/download/rp-2pico/

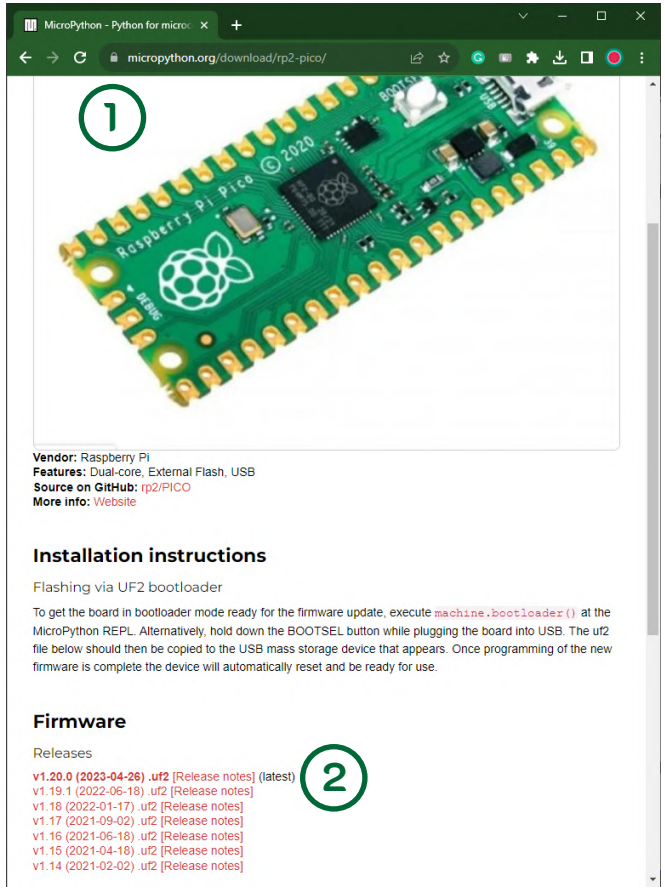
2 - Download the latest MicroPython UF2 file

3 - Press and hold BOOTSEL button.

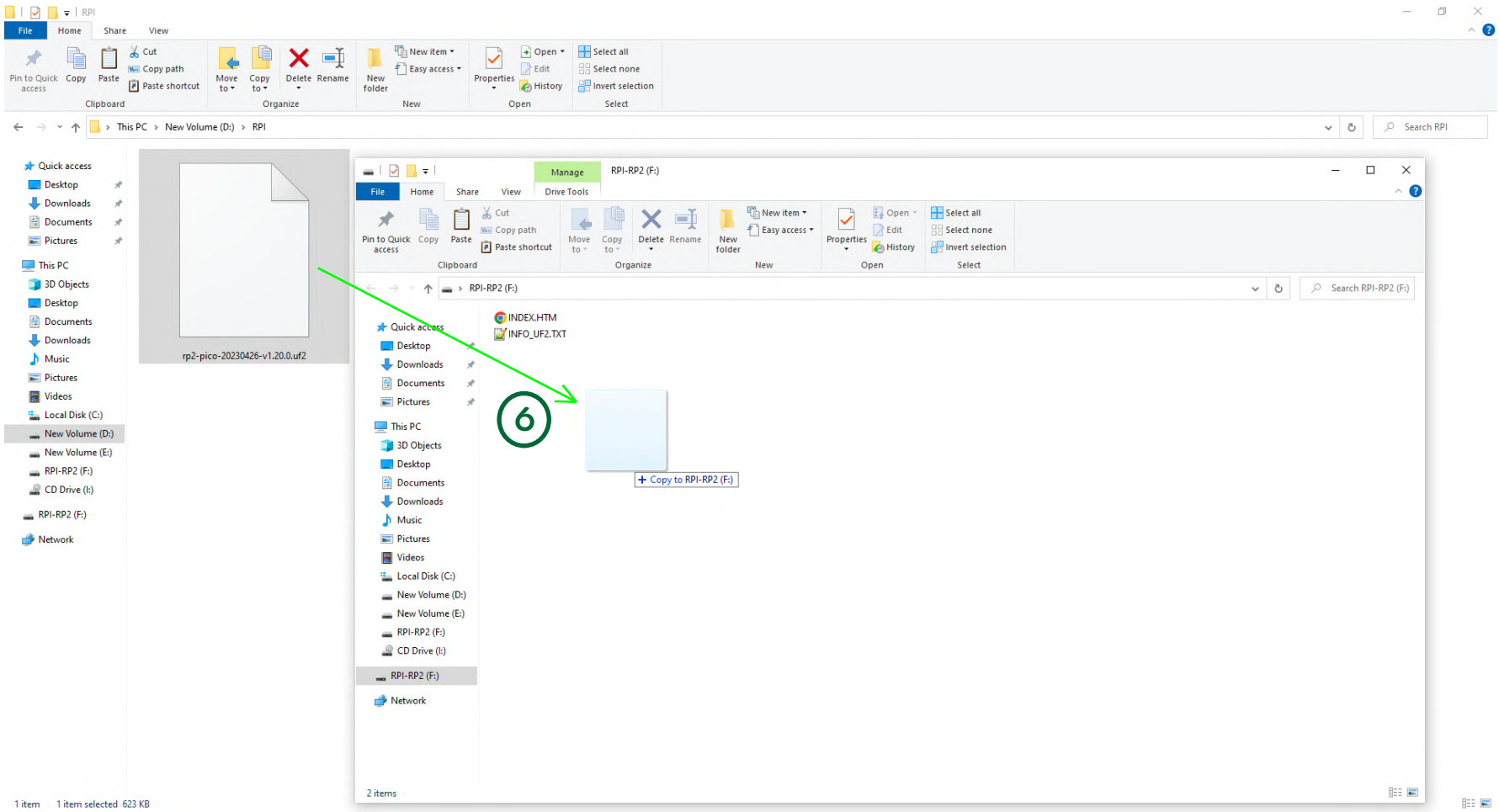
4 - Connect Pico to PC via a micro USB cable.

5 - Release BOOTSEL button once the drive RPI-RP2 appears.





6 - Copy UF2 file to the RPI-RP2 volume.



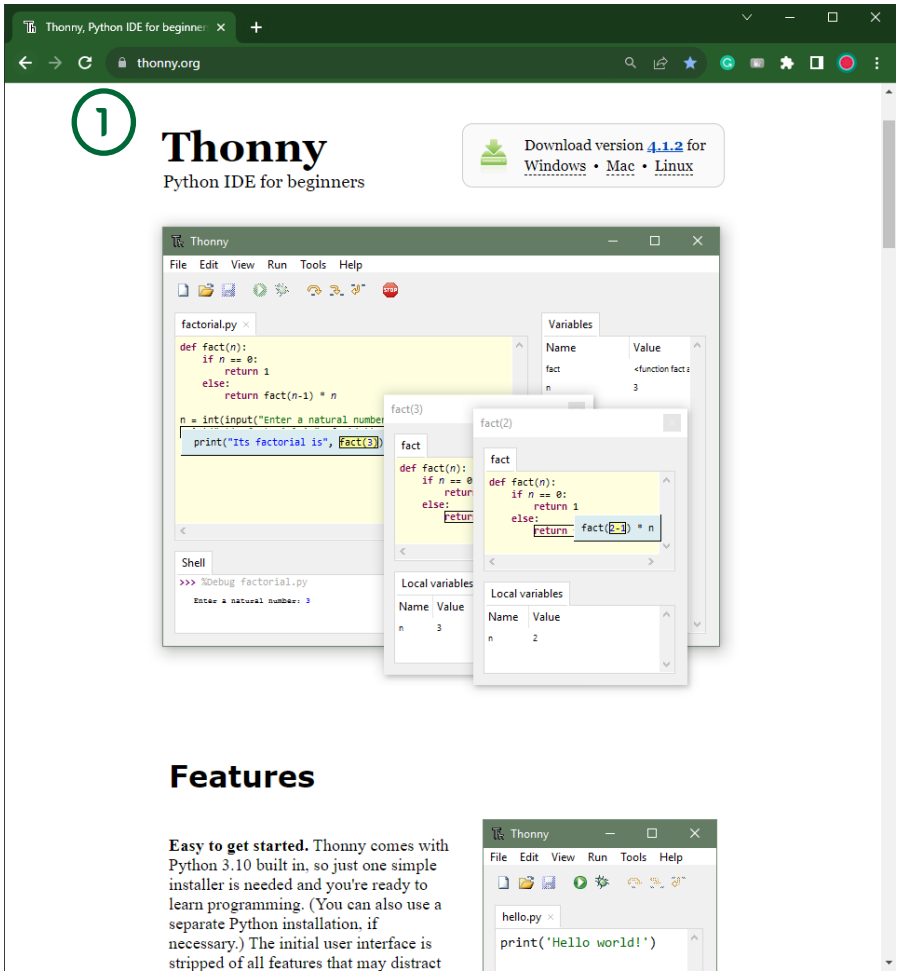
7 - Pico board will reboot and now MicroPython is running on your Pico

Once getting MicroPython loaded on your Pico, It is now ready to receive and execute your files.

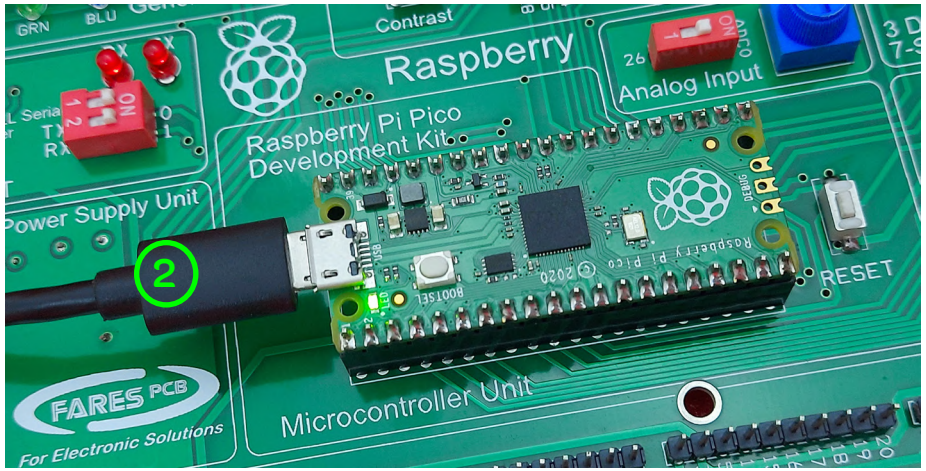
Installing Thonny Python IDE

1 - Download and install Thonny free from this link

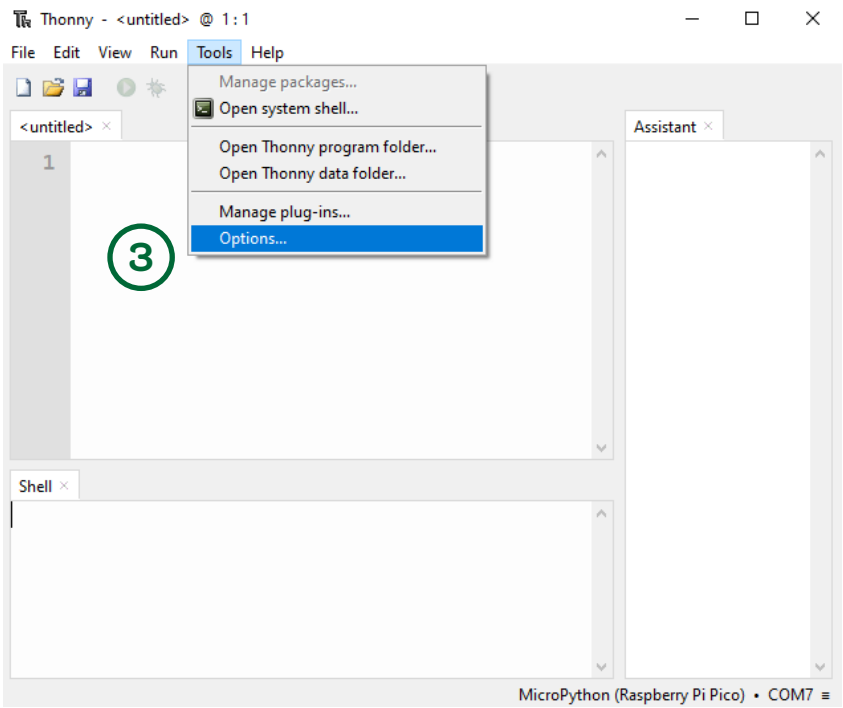
https://thonny.org/



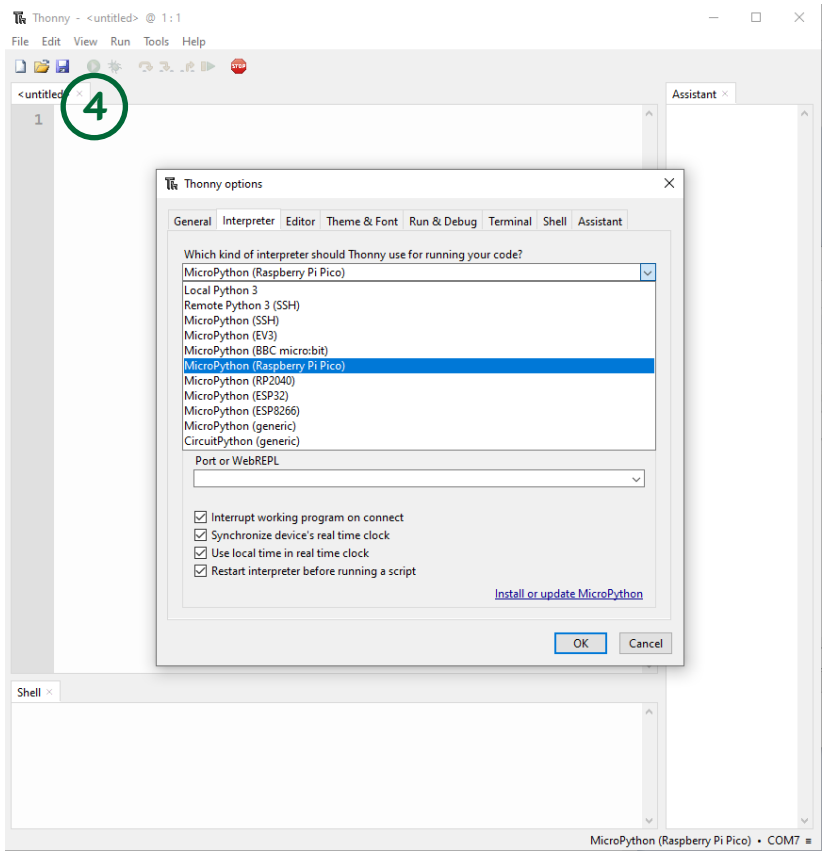
2 - Connect the Pico board to PC.



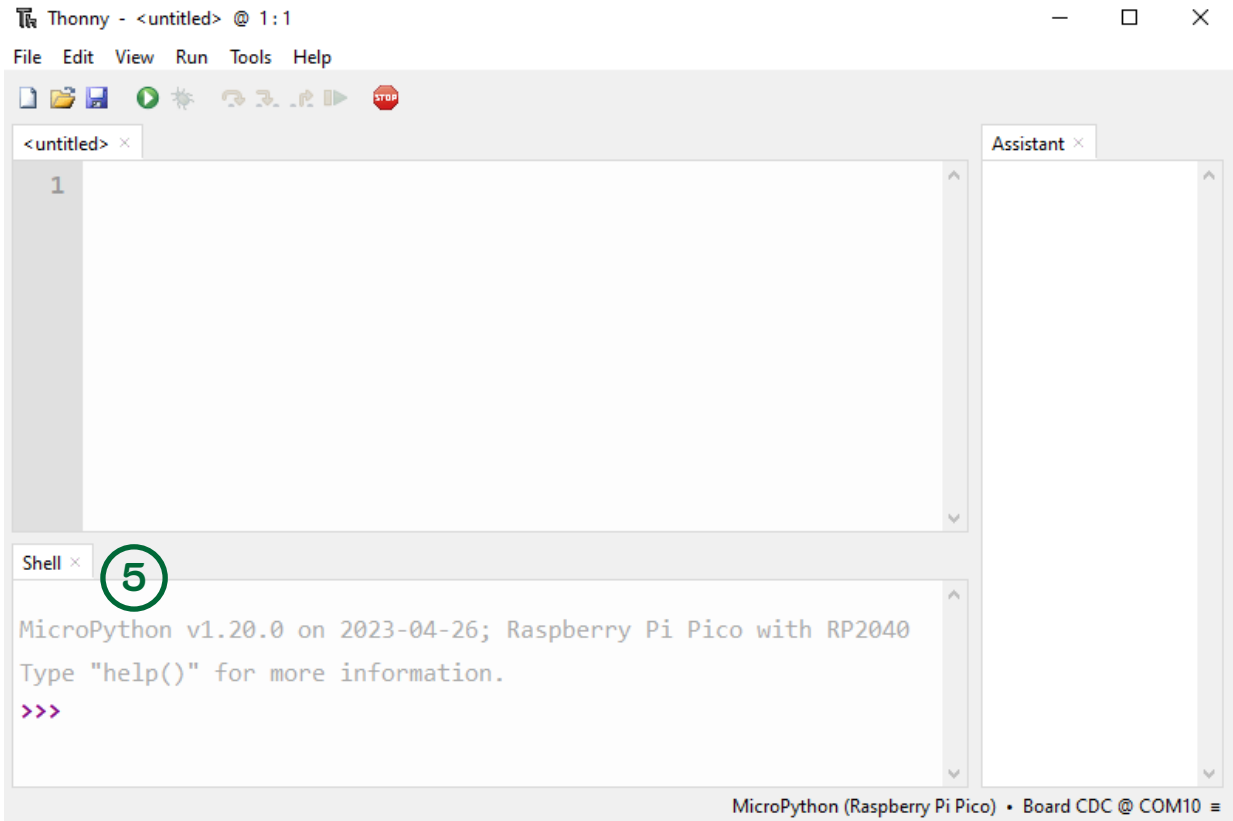
3 - Go to Tools>Options and click Interpreter tab.



4 - Select MicroPython (Raspberry Pi Pico) from the dropdown list.



5 - A Pyhton shell called REPL will popup to show that the Pico is connected.



Now you are ready to write your code and save files to Pico.

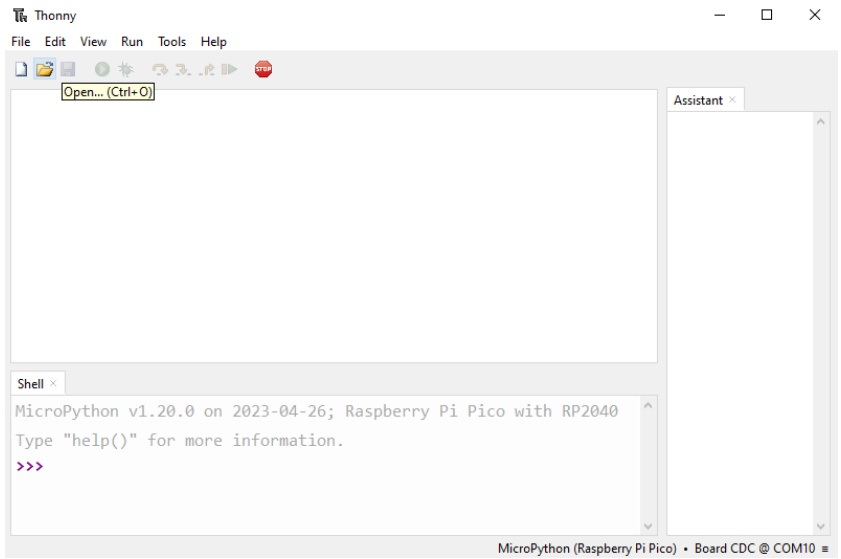
We have developed some example codes to test Raspberry Pi Pico kit sections separately. It is a good choice to download them from this link
https://fares-pcb.com/product/raspberry-pi-pico-development-kit/

How to load an example code and run it on Pico kit ?

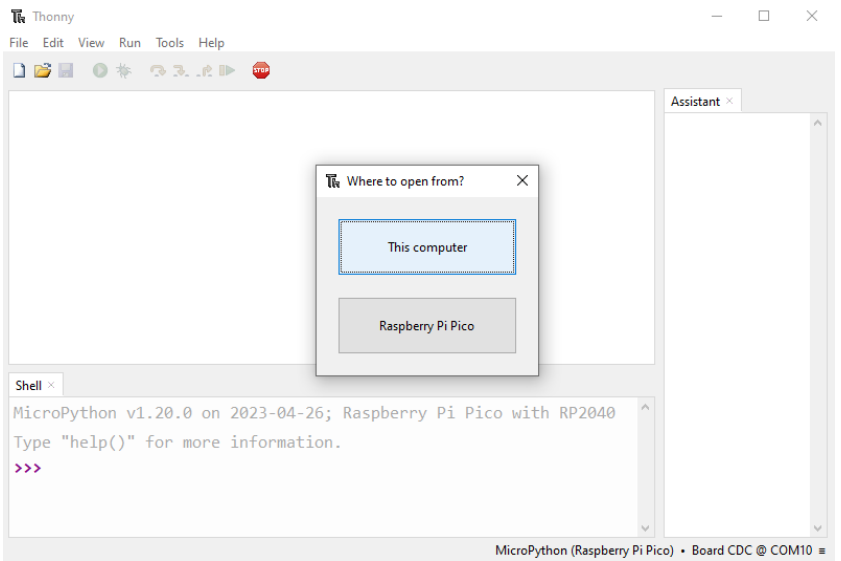
To load and run files on Pico development kit connect Pico board to PC and power kit either from USB-B connector or from DC adaptor.

Follow the next steps to load an example code which tests the built-in LED on Pico board by blinking it.

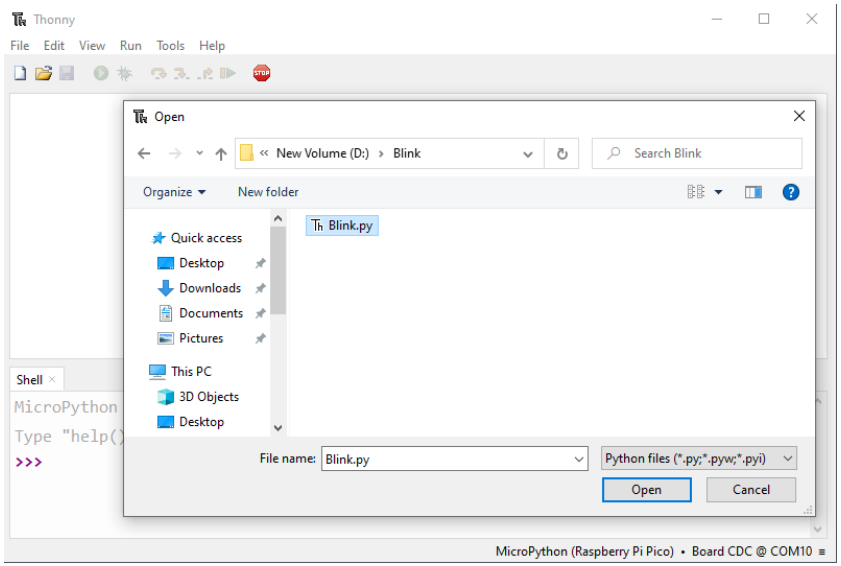
① Click “Open” icon



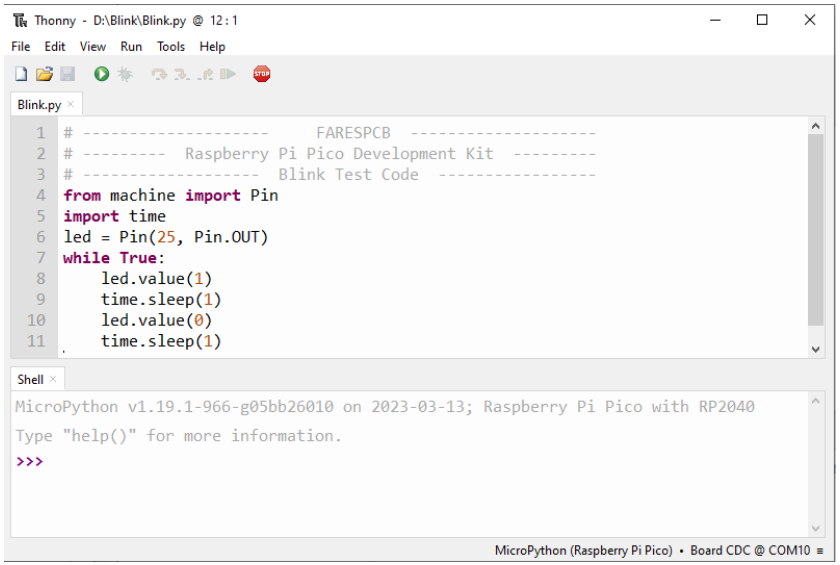
② Click “This computer” button to open files from your PC



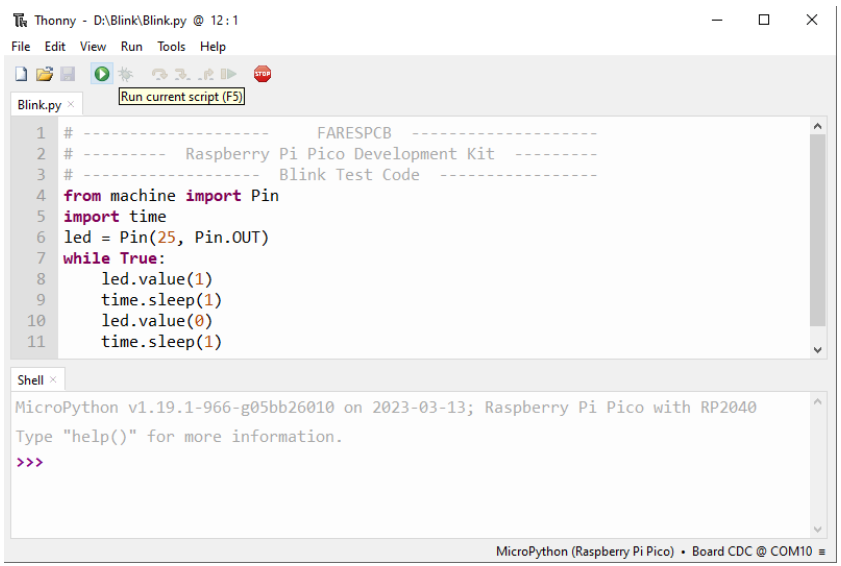
3 Browse and select Python file and click “Open”



4 Code will be shown in Thonny IDE



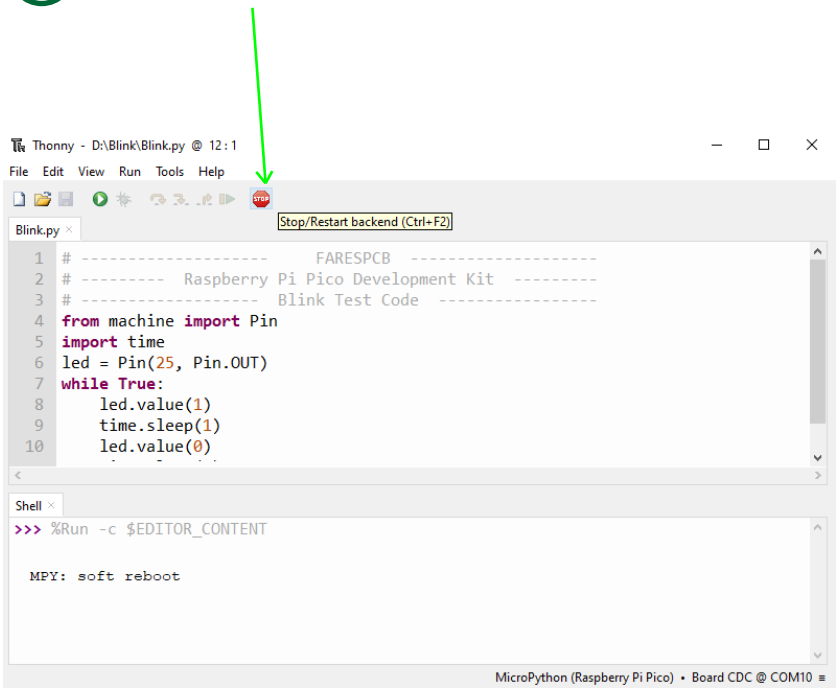
5 Click “Run current script” icon



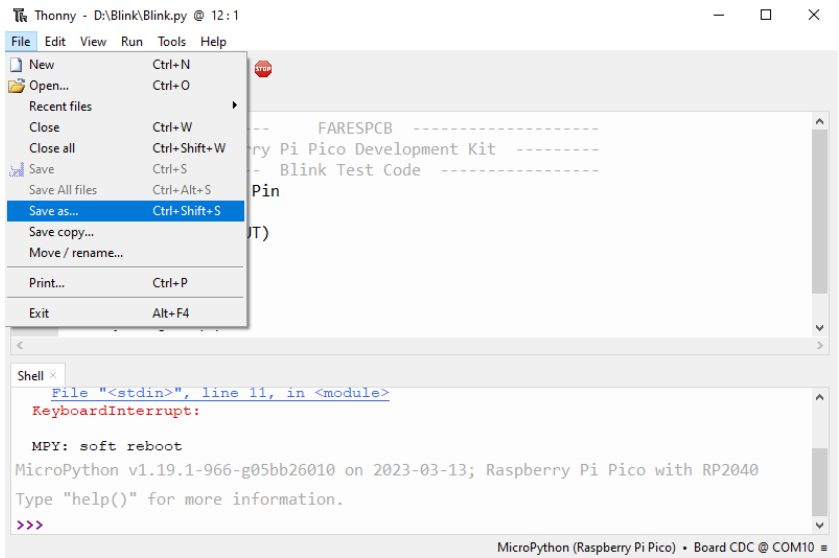
6 Shell shows that The file is uploaded and a soft reboot is initiated

It's worth noting that Blink file that has been run so far is not saved on Pico flash memory. It's just loaded and executed. Once you turn off the power and reconnect it the file is gone. To rerun Blink file again you need to load it as previously shown. However, to save Blink file permanently in Pico board flash memory follow the next steps

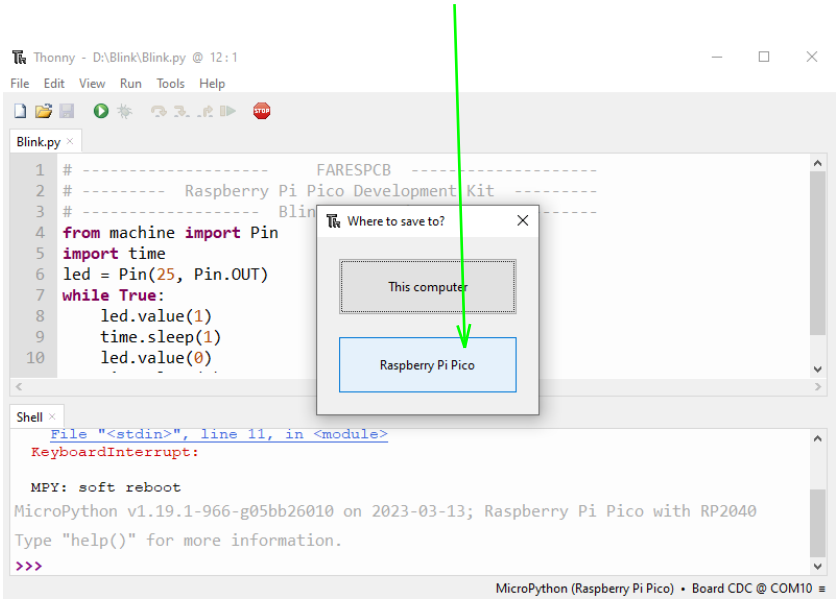
① Stop Running code



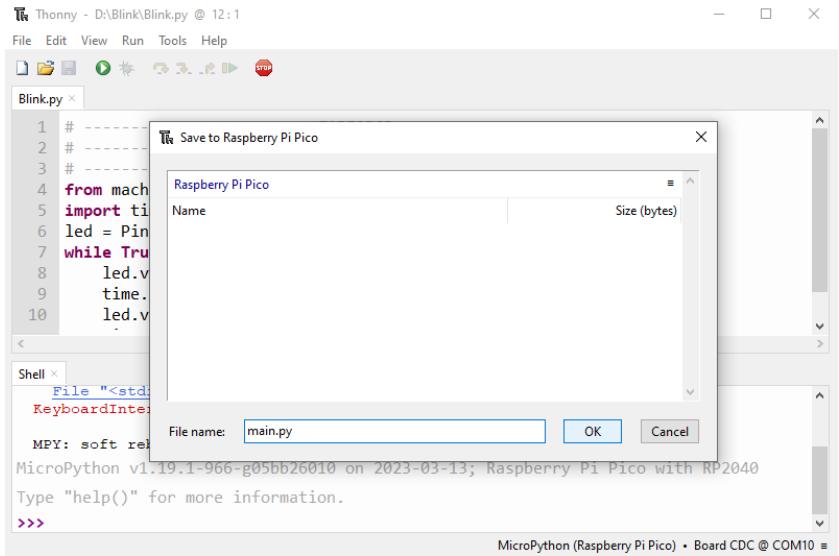
② Go to File>Save as



3 Click “Raspberry Pi Pico” button to save file to your Pico flash memory



4 Save file name as “main.py” not any other name



For our full range of products, see our website at **http://www.fares-pcb.com**
If you have any technical questions about our products,
e-mail us at **www.support@fares-pcb.com**

FARESPCB co. (Head office)
32 El-Falaky st, Bab El-Louq, Tahrir, Cairo, Egypt.
Tel: +202-23904484
Mob: +201000652977
Mob: +201022457902

FARESPCB Co reserves the right to make changes in circuit design, software and/or specifications at any time without prior notice. For the latest updated information, please visit our website at http://www.fares-pcb.com.

Information furnished by is believed to be accurate and reliable. However, **FARESPCB** assumes no responsibility arising from the use of the specifications described.

FARES PCB