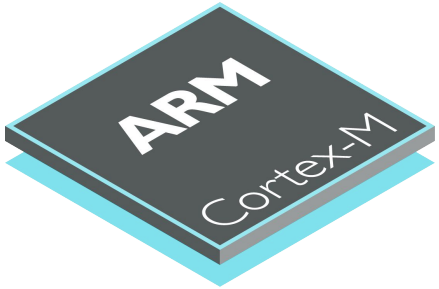
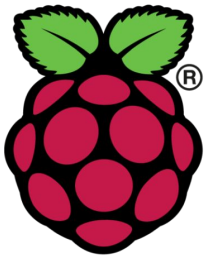


ARM Cortex M0+

José Rodrigo Solis Quiñónez 21002223

Ricardo José Farfán Miranda 21003210

Mariano José Tahay Alvarado 21001700



Raspberry Pi

Índice

Raspberry Pi Pico

Raspberry Pi Pico

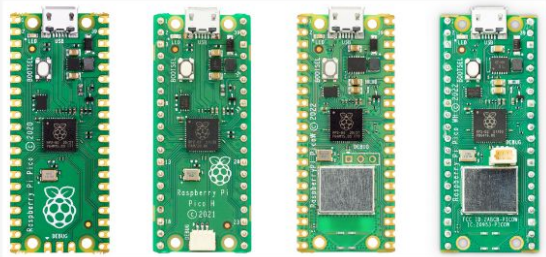
- RP2040
- Dual-core Arm Cortex M0+
- 133MHz
- 26 x multi-functional GPIO pins
- 2 x UART
- Temperature sensor

Raspberry Pi Documentation

Raspberry Pi Pico and Pico W

The family

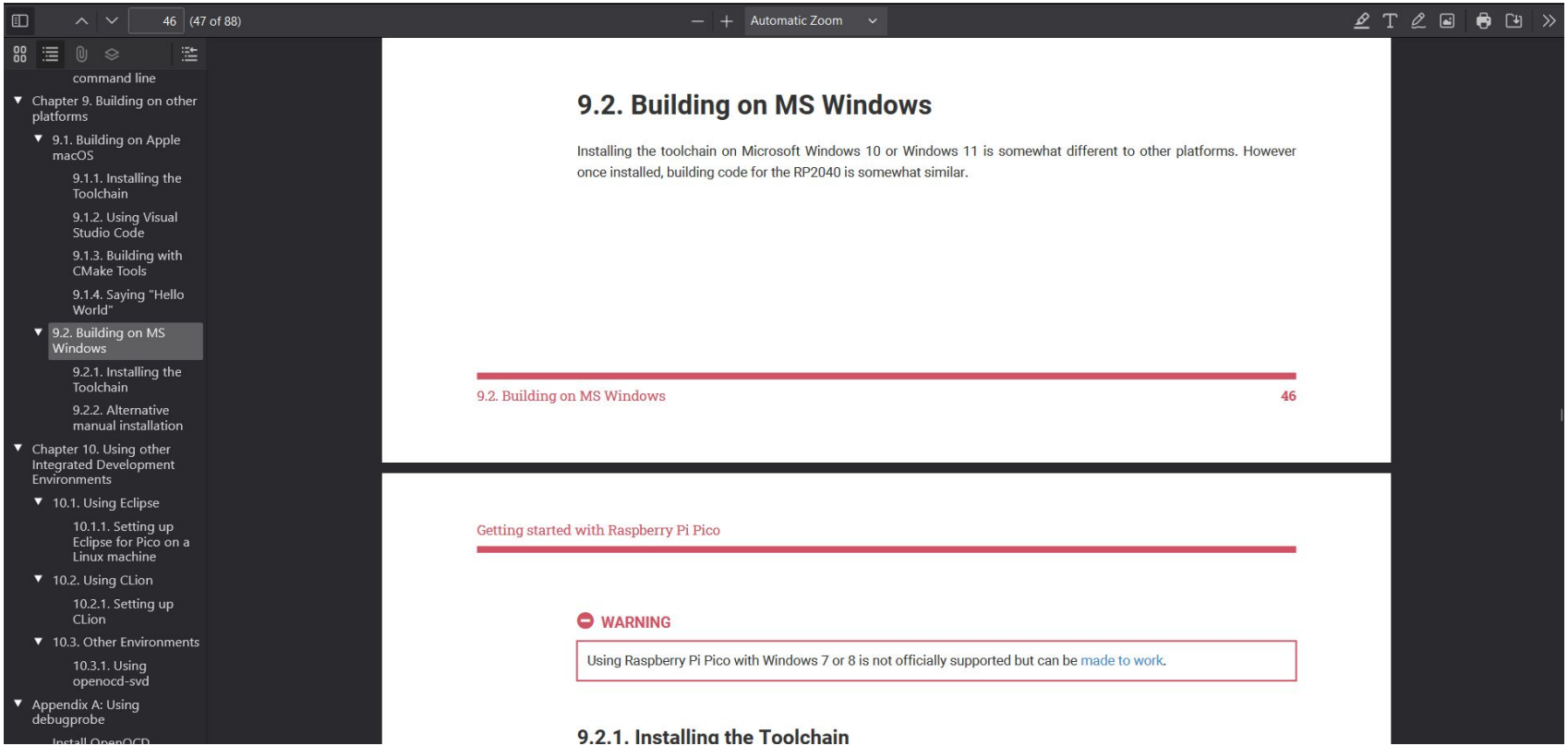
Edit this on GitHub



The Raspberry Pi Pico family currently consists of four boards; Raspberry Pi Pico (far left), Pico H (middle left), Pico W (middle right), and Pico WH (far right).

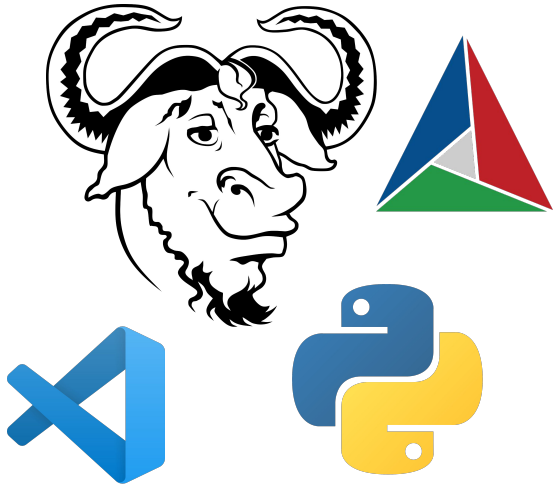
Instalación

Recursos


46 (47 of 88)
Automatic Zoom
command line
Chapter 9. Building on other platforms
9.1. Building on Apple macOS
9.1.1. Installing the Toolchain
9.1.2. Using Visual Studio Code
9.1.3. Building with CMake Tools
9.1.4. Saying "Hello World"
9.2. Building on MS Windows
9.2.1. Installing the Toolchain
9.2.2. Alternative manual installation
Chapter 10. Using other Integrated Development Environments
10.1. Using Eclipse
10.1.1. Setting up Eclipse for Pico on a Linux machine
10.2. Using CLion
10.2.1. Setting up CLion
10.3. Other Environments
10.3.1. Using openocd-svd
Appendix A: Using debugprobe
9.2. Building on MS Windows
Installing the toolchain on Microsoft Windows 10 or Windows 11 is somewhat different to other platforms. However once installed, building code for the RP2040 is somewhat similar.
9.2. Building on MS Windows 46
Getting started with Raspberry Pi Pico
WARNING
Using Raspberry Pi Pico with Windows 7 or 8 is not officially supported but can be made to work.
9.2.1. Installing the Toolchain

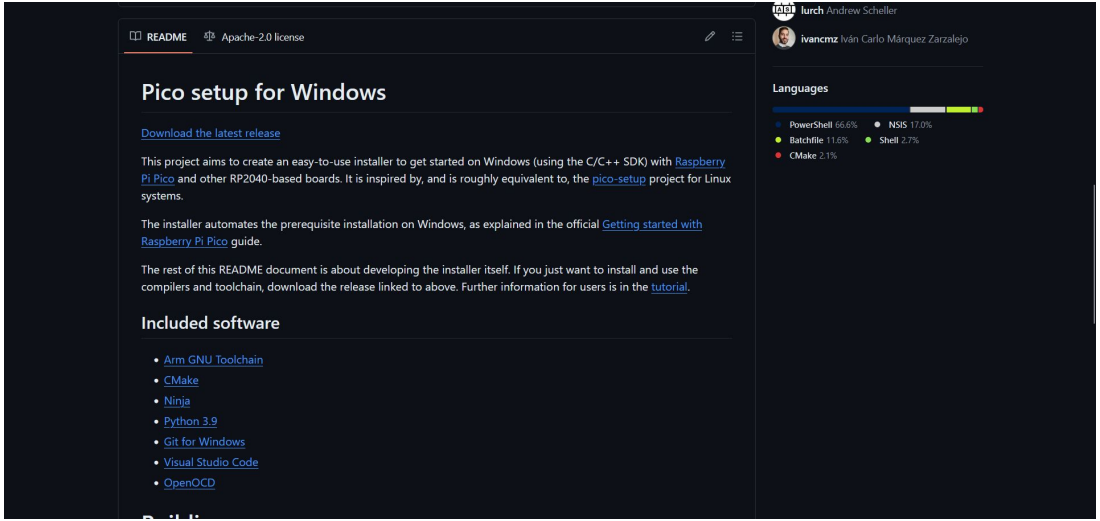
Windows

- Arm GNU Toolchain
- CMake
- Python 3.9
- Visual Studio Code
- OpenOCD



Recomendaciones

1. Usar chocolately
2. Instalar MinGW
3. Descargar el instalador del SDK
4. Siempre revisar que los PATH estén en orden



Debugger

Recursos

Debug with a second Pico

One Pico can reprogram and debug another using the `debugprobe` firmware, which transforms a Pico into a USB → SWD and UART bridge.

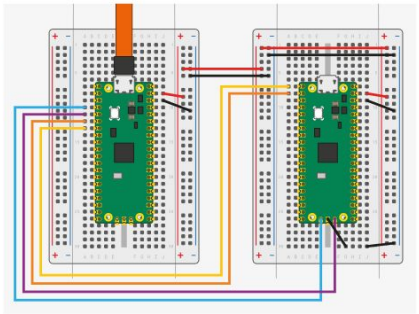


Figure 43. Wiring between Pico A (left) and Pico B (right) with Pico A acting as a debugger and Pico B as a system under test. You must connect at least the ground and the two SWD wires. Connect the UART serial port to provide access to the UART serial output of Pico B. You can also bridge the power supply to power both boards with one USB cable. For more information, see `debugprobe` wiring.

Install `debugprobe`

You can download a UF2 binary of `debugprobe` from the Pico documentation.

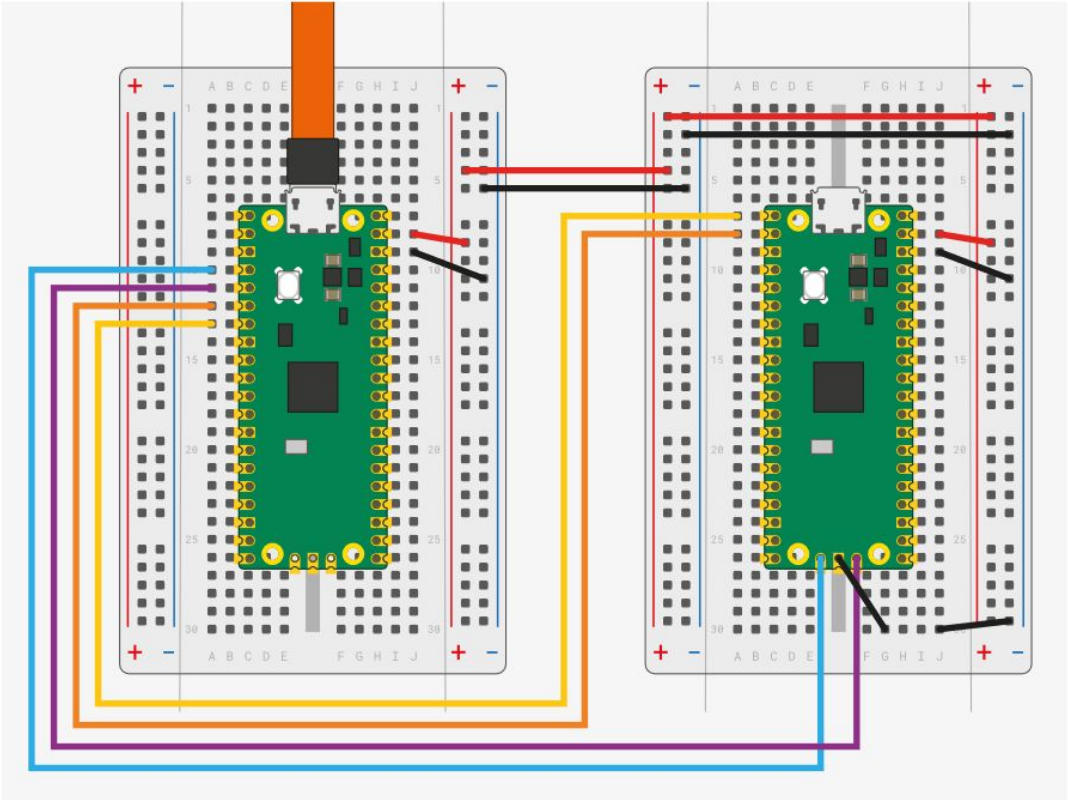
Boot the debugger Pico with the `BOOTSEL` button pressed. Copy `debugprobe_on_pico.uf2` onto the device to begin debugging.

NOTE

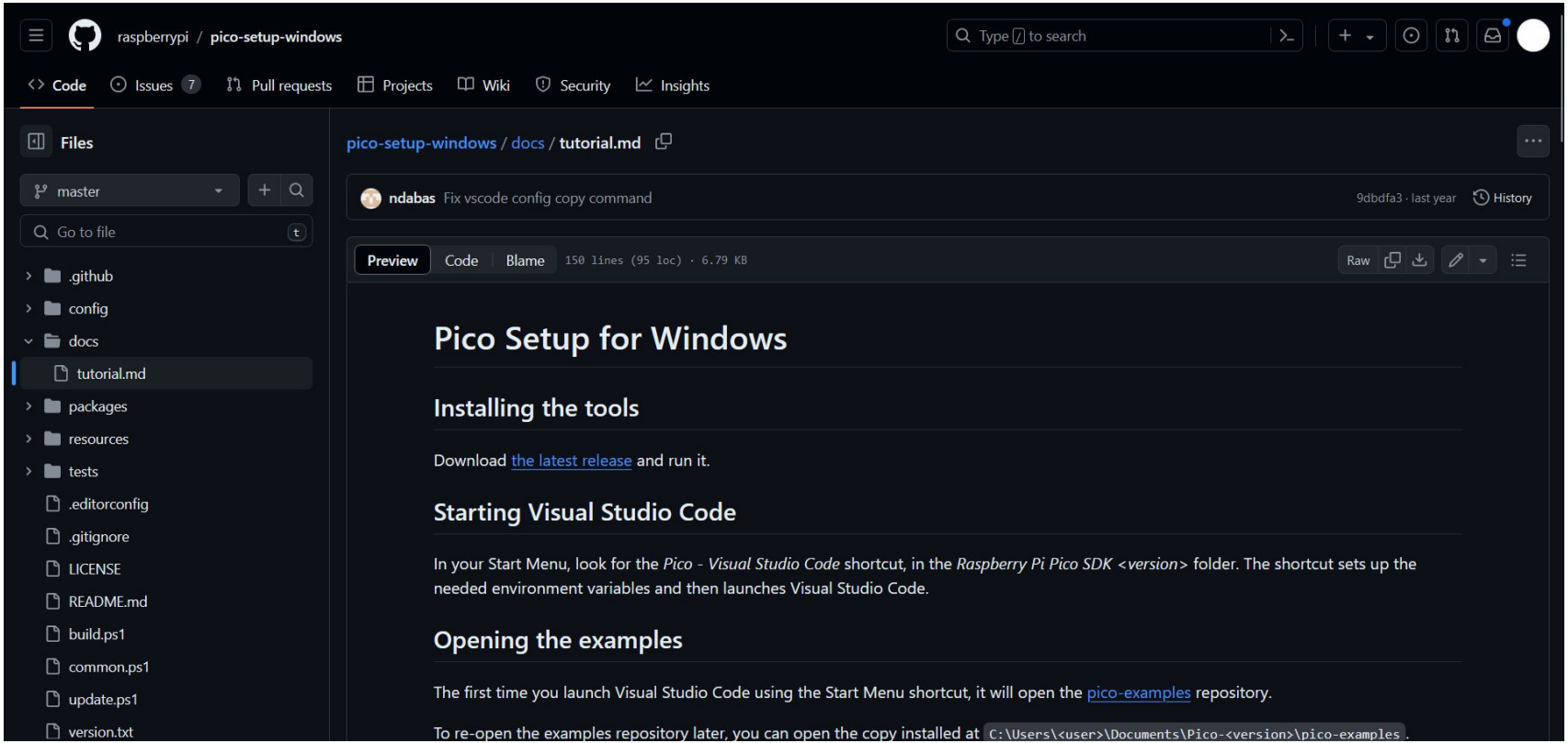
Use `debugprobe_on_pico.uf2` to use a Pico for debugging. Use `debugprobe.uf2` for the Debug Probe accessory hardware.

Build `debugprobe`

Circuito



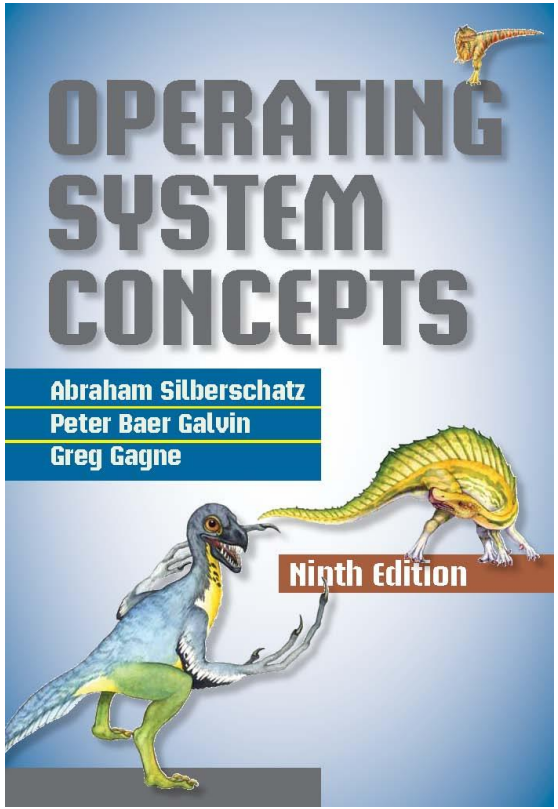
Crear un nuevo proyecto



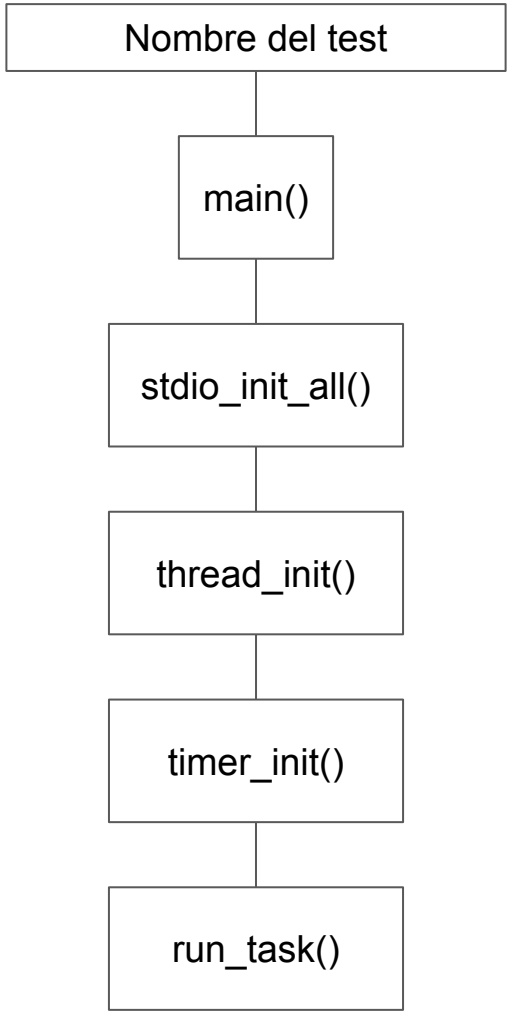
Código

Conceptos clave

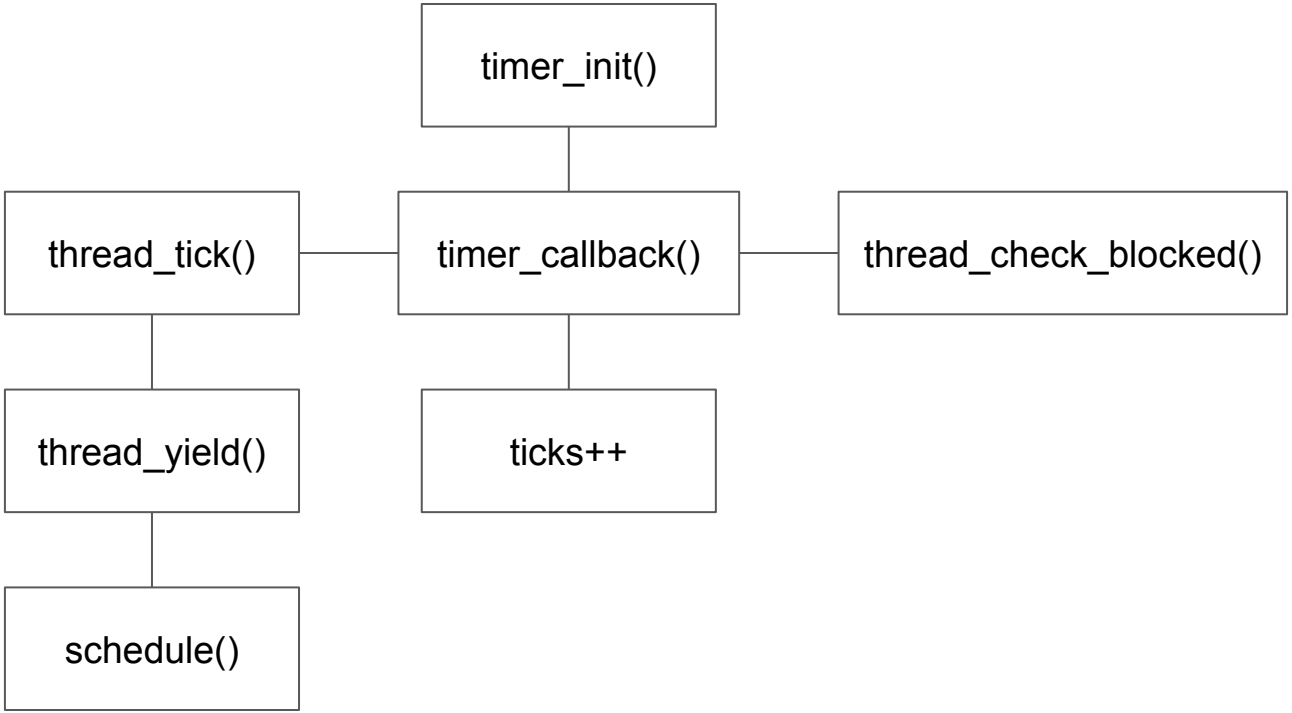
- Concepto de un thread
- Calendarizar un thread
- Algoritmo de “Round Robin”
- Context Switch
- Timers



Diagramas de flujo


Nombre del test
main()
stdio_init_all()
thread_init()
timer_init()
run_task()

main.c


timer_init()
thread_tick()
timer_callback()
thread_check_blocked()
thread_yield()
ticks++
schedule()

timer.c

Muchas Gracias

https://github.com/MarianoTahay/raspberry