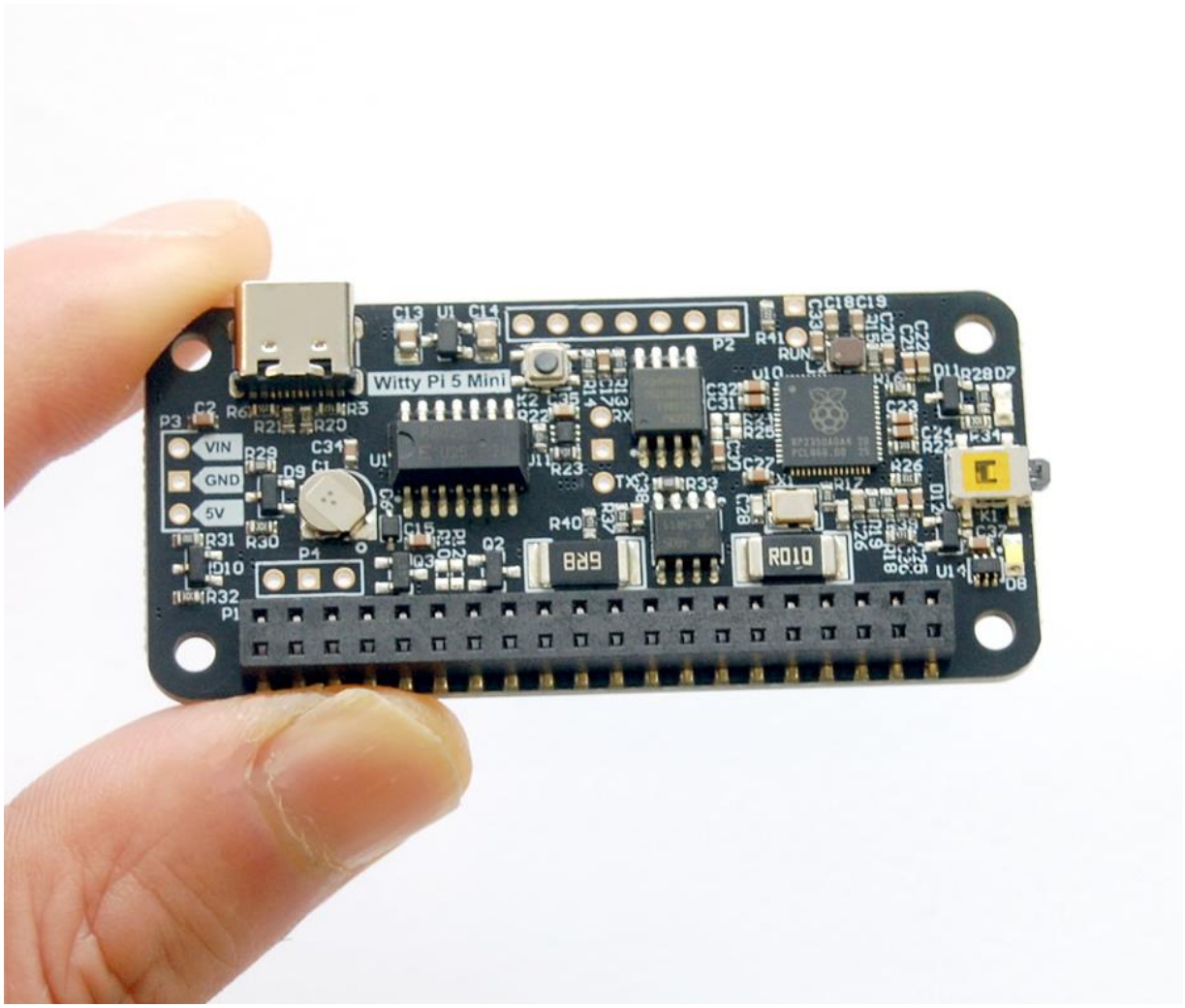
UUGear

Witty Pi 5 Mini

Realtime Clock and Power Management for Raspberry Pi

User Manual (revision 1.01)



Copyright © 2026 Dun Cat B.V., All rights reserved.

UUGear is a trade name of Dun Cat B.V.

UUGear

Table of Contents

1. Overview 4
   - 1.1 Introduction 4
   - 1.2 Witty Pi 5 Mini in the Witty Pi 5 Family 5
   - 1.3 Application Scenarios 6
   - 1.4 Board Layout 9
2. Specifications 10
3. Package Content 11
4. Working Principles 12
   - 4.1 How Witty Pi 5 Mini Powers Raspberry Pi 12
   - 4.2 The Three Key Factors: Time, Voltage, and Temperature 13
   - 4.3 The Schedule Scripts (.wpi, .act, and .skd Files) 13
   - 4.4 Everything via I^2C 14
   - 4.5 Low-power Hibernation 14
   - 4.6 Automatic Daylight Saving Time (DST) 15
5. Hardware Installation and Software Setup 16
   - 5.1 Mounting Witty Pi 5 Mini on Raspberry Pi Zero 16
   - 5.2 Mounting Witty Pi 5 Mini on Other Raspberry Pi Models 17
   - 5.3 Software Setup 20
   - 5.4 Software Usage 21
6. Update Firmware 27
   - 6.1 Get the New Firmware 27
   - 6.2 Upload the New Firmware 27
7. The Emulated USB Flash Drive 28
8. The Emulated USB Serial Device 30
9. About Schedule Script 31
   - 9.1 .wpi File 31
   - 9.2 .act File 32
   - 9.3 .skd File 32
   - 9.4 Run a Schedule Script 32
   - 9.5 Write Your Own Schedule Script 32
   - 9.6 Store Schedule Scripts in Witty Pi 5 Mini 35
   - 9.7 Using Schedule Script Generator 36
10. Additional Interfaces 37
    - 10.1 Serial Wire Debug (SWD) Port for RP2350 37
    - 10.2 Unpopulated RP2350 Reset Header 37

Copyright © 2026 Dun Cat B.V., All rights reserved.

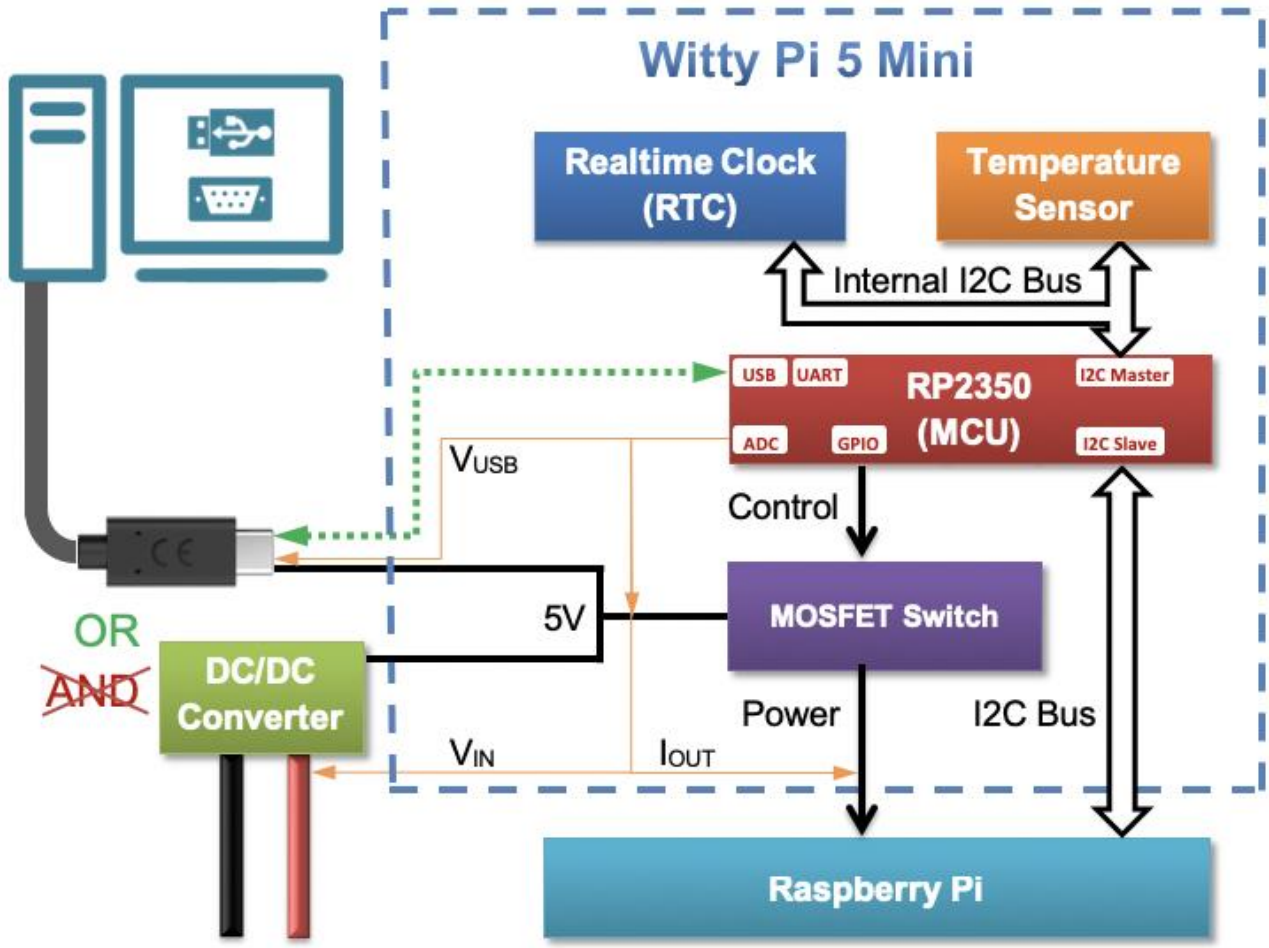
UUGear is a trade name of Dun Cat B.V.

10.3 Unpopulated Serial Port Connector 37
10.4 Unpopulated 7-Pin Header 37
10.5 Solder Pads for GP12, GP19, GP20 and GP21 38
10.6 Footprint for ID EEPROM 38
10.7 Power header / footprint for external DC/DC converter 38
11. Migrating from Witty Pi 4 Mini to Witty Pi 5 Mini 39
12. Log Files in the Witty Pi 5 Family 40
12.1 Software Log (on Raspberry Pi) 40
12.2 Firmware Log (on Witty Pi 5 Mini) 40
13. Frequently Asked Questions (FAQ) 41
13.1 What I2C address is used by Witty Pi 5 Mini? Can I change it? 41
13.2 What I2C Registers are provided by Witty Pi 5 Mini? 42
13.3 What GPIO Pins Are Used by Witty Pi 5 Mini? 48
14. Revision History 49

Copyright © 2026 Dun Cat B.V., All rights reserved.

UUGear is a trade name of Dun Cat B.V.

1. Overview



1.1 Introduction

Witty Pi is an add-on board for Raspberry Pi that brings real-time clock (RTC) and power management functionalities to your Raspberry Pi. It can define the on/off sequence of your Raspberry Pi, perform power on/off actions based on temperature or voltage thresholds, and significantly reduce overall energy consumption.

Witty Pi 5 Mini is the simplified member of the Witty Pi 5 family and the successor to Witty Pi 4 Mini. It inherits the firmware and software architecture of the Witty Pi 5 platform, while using a simpler and more flexible power-input design.

Like other members of the Witty Pi 5 family, Witty Pi 5 Mini is powered by the RP2350 microcontroller and can emulate both a USB flash drive and a USB serial device through its USB Type-C port. This allows the user to manage configuration files and scripts directly from a computer, and to monitor real-time logs via the serial interface.

Communication between Witty Pi 5 Mini and the Raspberry Pi is handled exclusively via the I²C interface, following the common Witty Pi 5 platform architecture. No additional Raspberry Pi GPIO pins are used.

Witty Pi 5 Mini includes the following hardware features:

- A high-precision real-time clock, with an accuracy of ±3.8 to 5 ppm.
- A dedicated temperature sensor with 0.0625°C resolution.
- A VIN sensing input and a GP12_EN control signal for use with an optional external DC/DC converter.
- An RP2350 microcontroller (MCU) with an external 16MB flash memory.
- The VUSB measurement monitors the common 5V supply rail used by either USB-C power or an external DC/DC converter.

Witty Pi 5 Mini does not have onboard DC/DC converter: an optional external DC/DC converter can be used when a regulated 5V supply must be generated from another input voltage.

1.2 Witty Pi 5 Mini in the Witty Pi 5 Family

Witty Pi 5 HAT+ was the first product in the Witty Pi 5 family and serves as its standard, full-featured model. It established the common RP2350-based firmware and software architecture that is also used by Witty Pi 5 Mini.

Witty Pi 5 Mini therefore does not introduce a separate software ecosystem. It uses the same wp5 software package, the same schedule-script architecture, the same USB-based firmware update mechanism, and the same general configuration and logging model as Witty Pi 5 HAT+.

Witty Pi 5 Mini does not meet the HAT+ specification, due to the lack of ID EEPROM. However the footprint for ID EEPROM has been reserved on the back of the board, and you may solder one by yourself when it is needed.

Many capabilities that were new when Witty Pi 5 HAT+ was introduced are consequently standard features of the Witty Pi 5 family. Schedule scripts are stored and processed on Witty Pi itself rather than by software running on the Raspberry Pi, allowing scheduling to remain functional even if the Raspberry Pi crashes or encounters problems during power transitions.

The shared Witty Pi 5 schedule architecture supports the traditional .wpi format as well as the .act and .skd formats. Witty Pi 5 Mini uses these formats in the same way as the other Witty Pi 5 models:

- The .wpi format is retained for compatibility with earlier Witty Pi generations and remains useful for compact definitions of repeating schedules.
- The .act format provides a linear list of timestamped power actions, while .skd is the compact format actually executed by the firmware.

Compared with Witty Pi 4 Mini, these shared platform features represent a major generational change. Compared with Witty Pi 5 HAT+, however, they should be understood as common Witty Pi 5 capabilities rather than Mini-specific additions.

Witty Pi 5 Mini also inherits the developer-oriented features of the platform. The firmware (written in C using the pico-sdk) and the wp5 software (written in C) are open-source and can be modified or adapted as needed.

Firmware updates use the standard RP2350 USB boot process: hold the BOOTSEL button while connecting USB, then copy a compiled .uf2 file to the virtual RP2350 drive. No dedicated programmer is required.

What distinguishes Witty Pi 5 Mini from Witty Pi 5 HAT+ is mainly its power-input design. The Mini omits the onboard high-voltage DC/DC converter and isolated dual-input power paths, and instead allows an optional external DC/DC converter to be selected according to the application. The following sections describe these Mini-specific power characteristics in detail.

With compatible firmware (v1.8 or later) and wp5 software v5.4.0 or later, Witty Pi 5 Mini also supports autonomous daylight saving time (DST) adjustments and additional voltage-confirmation readings. DST works even while the Raspberry Pi is off; voltage confirmations apply to the Mini's monitored upstream VIN in external-DC/DC mode. See Sections 4.6 and 5.4 for configuration details.



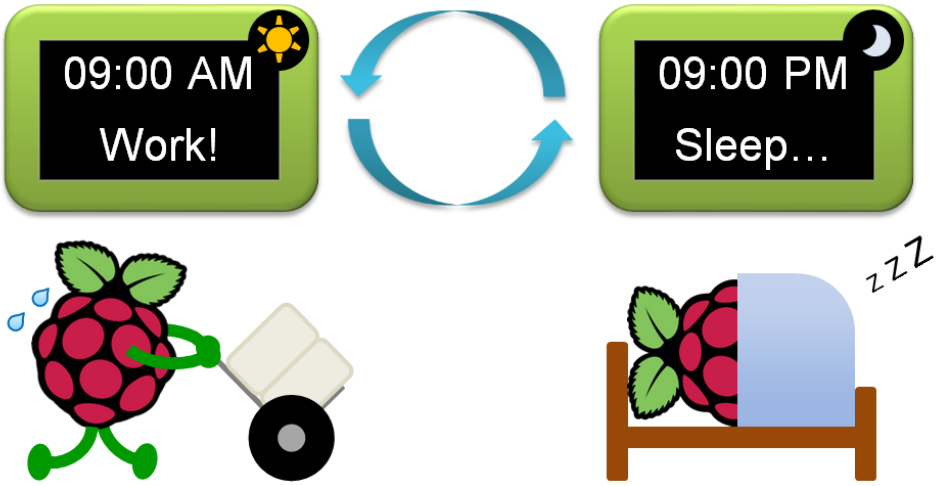


1.3 Application Scenarios

1.3.1 Time Controlled Device

The high-precision RTC (real-time clock) on Witty Pi 5 Mini has built-in temperature compensation and provides accurate timekeeping even without a network connection.

With Witty Pi 5 Mini, you can define the power-on and power-off times for your Raspberry Pi, or even create a complex schedule using scripting to fit your specific needs.

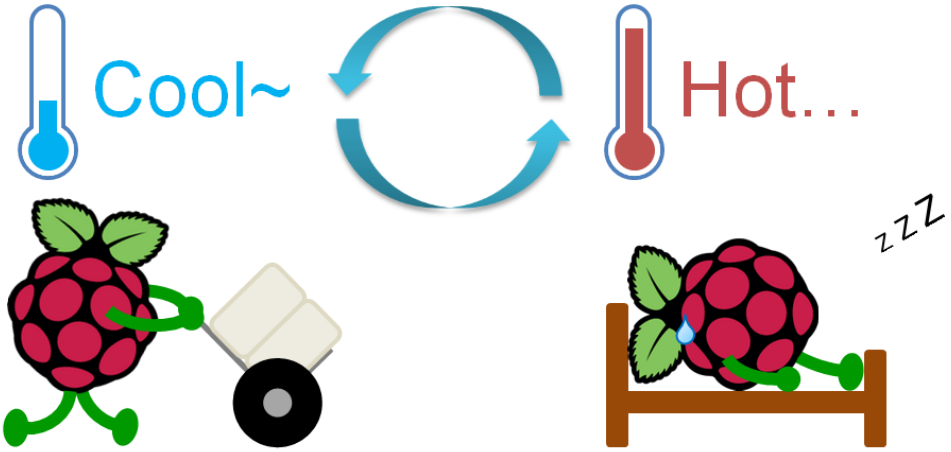


By powering down the Raspberry Pi when it's not needed and powering it up only at the right time, you can significantly reduce overall power consumption. This feature is especially valuable for battery-powered devices.

1.3.2 Temperature Controlled Device

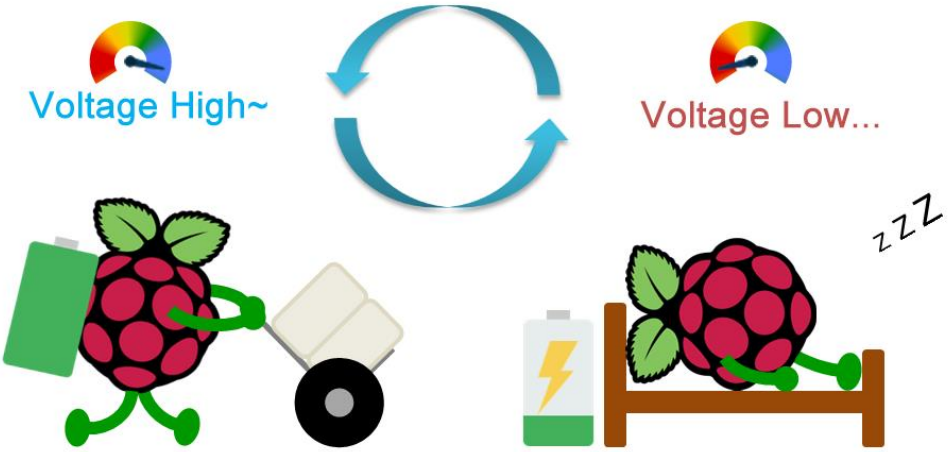
The temperature sensor on Witty Pi 5 Mini offers a resolution of 0.0625°C and supports both high and low temperature threshold interrupts.

You can configure Witty Pi 5 Mini to power on or off the Raspberry Pi based on temperature thresholds, effectively turning your Raspberry Pi into a temperature-controlled device.



1.3.3 Voltage Controlled Device

When Witty Pi 5 Mini is powered through an external DC/DC converter, it can monitor the converter's upstream input voltage through the VIN sensing input and make power-on/off decisions accordingly.



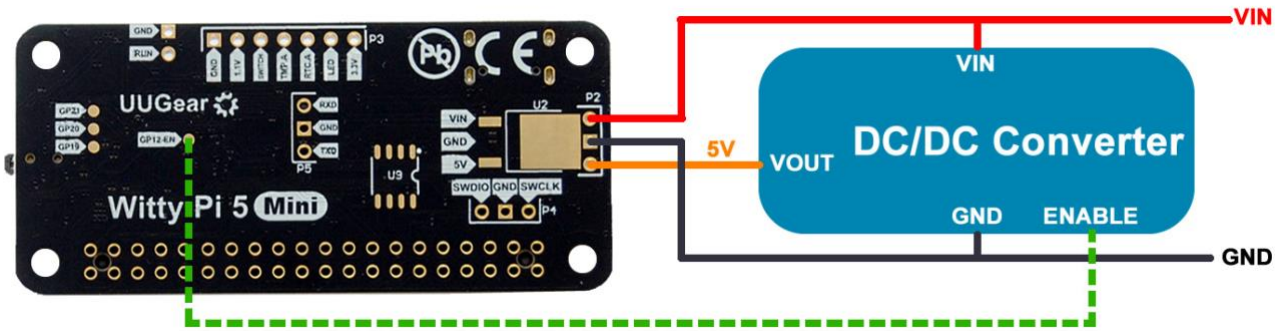
When the configured power source is V-IN via external DC/DC, Witty Pi 5 Mini can safely shut down the Raspberry Pi when VIN drops below the configured low-voltage threshold, and can power

it up again after VIN rises above the configured recovery-voltage threshold. These VIN thresholds are not used when V-USB is configured as the actual power source.

With firmware v1.8 or later, additional immediate voltage readings can be required before a low-VIN shutdown or recovery. This helps reject brief transients but is not a substitute for stable 5V output from the external power converter.

1.3.4 External DC/DC Power Supply

Witty Pi 5 Mini does not include an onboard DC/DC converter. If your power source is not a regulated 5V supply, you may use an external DC/DC converter and feed its regulated 5V output to the same 5V rail used by VUSB.



The external converter may be a buck, boost, or buck-boost design. Witty Pi 5 Mini does not assume a fixed minimum VIN; instead, the firmware can enable the converter through GP12_EN pad and verify the resulting common 5V rail through the VUSB measurement.

If the converter's enable input is connected to GP12_EN pad, the firmware can turn the converter on when V-IN via external DC/DC is selected and turn it off when V-USB is selected. If GP12_EN is not connected, changing the power-source setting cannot physically switch the external converter, and as a result the DC/DC converter will be always on.

The USB-C V_{BUS} input and the 5V output of an external DC/DC converter are not isolated from each other on Witty Pi 5 Mini. **Do not power both from independent sources at the same time unless your external circuit provides suitable power-path isolation, OR-ing, or source selection.**

For this reason, Witty Pi 5 Mini does not provide the built-in dual-source UPS behavior of Witty Pi 5 HAT+. Any redundant-power or automatic-source-switching function must be implemented by the external power system.

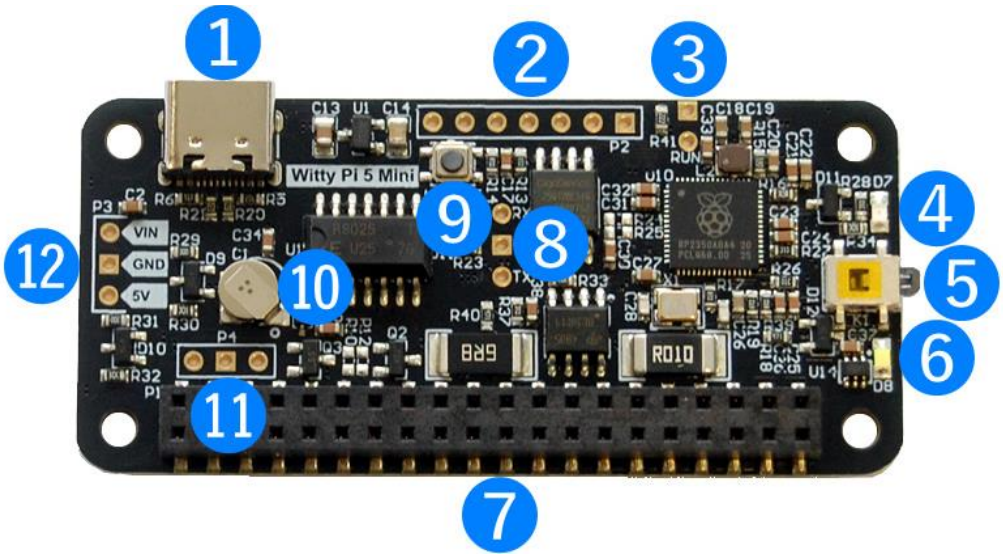
1.4 Board Layout

Witty Pi 5 Mini supports all Raspberry Pi models with a 40-pin GPIO header, including A+, B+, 2B, Zero, Zero W, Zero 2 W, 3B, 3B+, 3A+, 4B, and 5B.

For models like Zero, Zero W, and Zero 2 W, you will need to solder a 40-pin header onto the board beforehand to ensure a reliable connection with Witty Pi.

The PCB shape of Witty Pi 5 Mini is the same as that of Witty Pi 4 Mini, and the positions of the switch and the USB Type-C connector are also identical. However, some other connectors are placed differently on Witty Pi 5 Mini, which makes it physically incompatible for a direct replacement of Witty Pi 4 Mini.

The diagram below shows all available interfaces on the Witty Pi 5 Mini board.



1. USB Type-C connector for 5V power input (V_{USB})
2. Unpopulated 7-pin header
3. Unpopulated RP2350 reset header (RUN and GND)
4. Red LED as power indicator
5. ON/OFF button
6. White LED as status indicator
7. 2x20 pin stacking header for connecting to Raspberry Pi
8. Unpopulated serial port connector (RXD, GND and TXD)
9. BOOTSEL button
10. Super capacitor for off-power time keeping
11. Serial Wire Debug (SWD) port for RP2350
12. Unpopulated power header (VIN, GND and 5V)

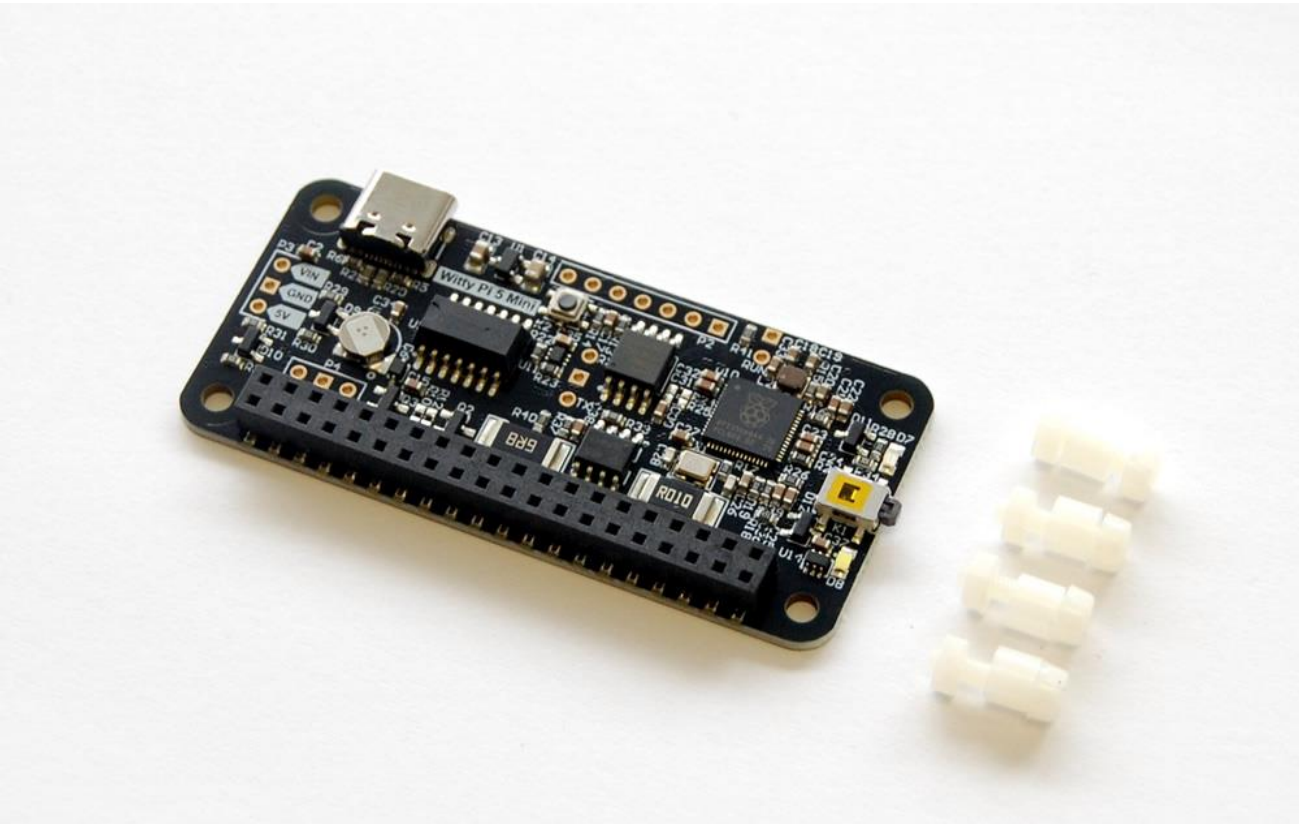
2. Specifications

Dimension	65mm x 30mm x 6mm
Weight	12g (net weight without accessories)
Microcontroller	RP2350 (datasheet)
Realtime Clock	RX8025T-UB (datasheet)
Temperature Sensor	TMP112 (datasheet)
DC/DC Converter	Not onboard. An optional external DC/DC converter may be used and controlled through GP12_EN pad.
MOSFET Switch	AO4805 (datasheet)
Supercapacitor	0.07F/3.3V (for time keeping only, when no power supply is connected)
Power In	Regulated DC 5V via USB Type-C Or Regulated 5V from an optional external DC/DC converter
Output Current	Up to 3A for Raspberry Pi and its peripherals
Standby Current	~0.4mA during hibernation (powering via USB 5V, after supercapacitor has been fully charged)
Operating Environment	Temperature -30°C~80°C (-22°F~176°F) Humidity 0~80%RH, no condensing, no corrosive gas

3. Package Content

Each package of Witty Pi 5 Mini contains:

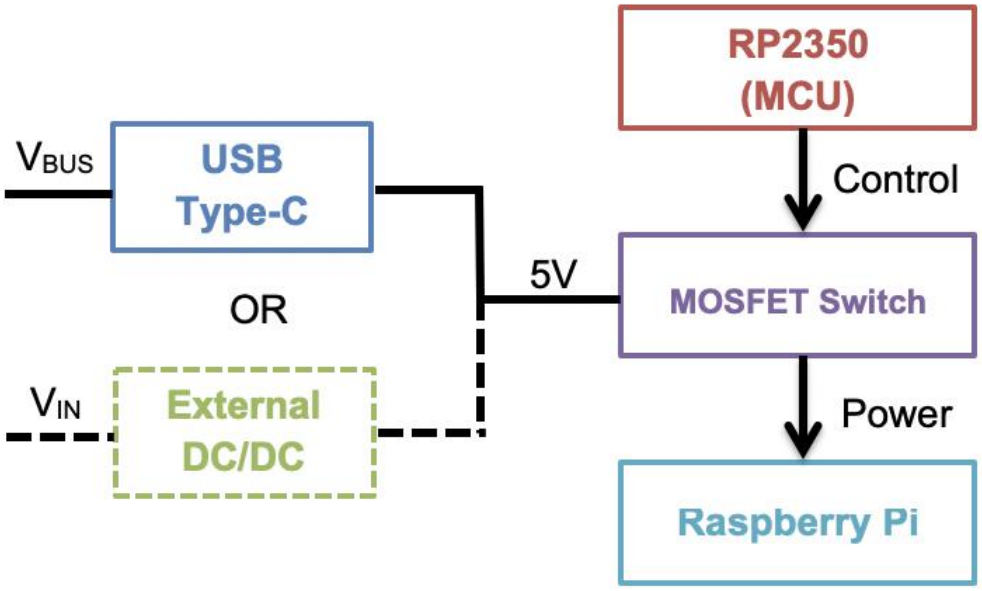
- Witty Pi 5 Mini board x 1
- M2.5 x 10mm plastic screws x 4
- 3.5mm plastic spacer x 4
- M2.5 plastic nut x 4



4. Working Principles

4.1 How Witty Pi 5 Mini Powers Raspberry Pi

Witty Pi 5 Mini uses a single controlled 5V supply rail for the Raspberry Pi. A regulated 5V supply can be provided through the USB Type-C connector. Alternatively, an optional external DC/DC converter can provide the regulated 5V rail while its upstream input voltage is monitored as V_{IN}.



Unlike Witty Pi 5 HAT+, Witty Pi 5 Mini has no onboard DC/DC converter and no ideal-diode isolation between USB-C V_{BUS} and the 5V output of an external converter. These sources share the same 5V node on the input side of the Raspberry Pi power switch. Therefore, two independent 5V sources must not be connected simultaneously unless external isolation or power-path management is provided.

On Witty Pi 5 Mini, the power-source setting specifies the actual power architecture rather than a priority between two simultaneously available sources. Select V-USB when the board is powered from the USB-C 5V input. Select V-IN via external DC/DC when an external converter provides the 5V rail. **The configured source must match the actual wiring.**

When V-USB is selected, the firmware keeps GP12_EN low. When V-IN via external DC/DC is selected, the firmware enables GP12_EN (active-high) and treats the VIN measurement as the upstream source voltage. The external converter may be buck, boost, or buck-boost, so the firmware does not impose a fixed minimum VIN for starting the system.

Before powering the Raspberry Pi in external-DC/DC mode, the firmware verifies that the common 5V rail is high enough using the VUSB measurement. Low-voltage shutdown and recovery thresholds, however, are applied to the raw VIN measurement. A dedicated MOSFET switch then controls whether the Raspberry Pi receives power.

4.2 The Three Key Factors: Time, Voltage, and Temperature

Time, voltage, and temperature jointly determine the ON/OFF state of the Raspberry Pi.

For shutdown, these three factors work in an OR logic: if any one of them triggers a shutdown condition, the Raspberry Pi will be shut down. This logic is intuitive—whether it's a scheduled shutdown, under-voltage, or over-temperature condition, each represents an urgent reason to shut the system down.

For startup, the logic is AND: all applicable conditions must be satisfied before the Raspberry Pi is powered on. On Witty Pi 5 Mini, VIN low-voltage and recovery thresholds apply only when V-IN via external DC/DC is configured as the actual power source. When V-USB is selected, VIN is not used to decide low-voltage shutdown or recovery.

4.3 The Schedule Scripts (.wpi, .act, and .skd Files)

The .wpi script format originates from earlier Witty Pi generations and is retained by the Witty Pi 5 family for compatibility. It defines a start time, an end time, and a recurring ON/OFF sequence between them. Its compact loop-based structure is convenient for repeating schedules, although it can require extra calculation when an exact one-time action is needed.

The Witty Pi 5 platform also supports the .act format. It lists power-on and power-off actions with precise timestamps in a linear, human-readable form, making one-time or irregular schedules easier to express. Witty Pi 5 Mini uses this same format as the other members of the Witty Pi 5 family.

The firmware actually executes a .skd script, which is a compact representation of scheduled actions. This behavior is common across the Witty Pi 5 family: if no .skd file is available, the firmware can generate one from an .act file, or convert a .wpi file through .act into .skd.

The .skd timestamps use the Witty Pi schedule time base (seconds since 2000-01-01), not Unix epoch seconds. When generating .skd files directly, account for the board's local time and any configured DST rule.

On the Witty Pi 5 platform, schedule scripts are stored and processed in onboard flash memory. Witty Pi 5 Mini inherits this architecture, so scheduling remains independent of the Raspberry Pi and can continue to function even if the Pi encounters problems during startup or shutdown.

If automatic DST is enabled, the firmware interprets scheduled local times using its configured rule. After changing the DST rule, reactivate any active schedule script to refresh pending actions, particularly around spring-forward and autumn fall-back transitions.

4.4 Everything via I²C

Like other Witty Pi 5 family members, Witty Pi 5 Mini uses the Raspberry Pi I²C interface (SDA and SCL) for normal communication, avoiding the legacy GPIO-based control signals used by earlier Witty Pi generations.

The shared wp5 software performs all normal interactions with Witty Pi 5 Mini over I²C, including status polling, reading variables, issuing commands, and saving configuration.

Even the **watchdog** functionality is implemented via I²C. The software periodically polls a register over I²C, which will trigger the firmware to update the heartbeat value. If the firmware detects the heartbeat value hasn't been updated in time, it records a Heartbeat Missed event. After a predefined number of consecutive missed heartbeats, the firmware performs a forced power cycle to recover the system.

The realtime clock and temperature sensor are connected to the RP2350 through an internal I2C bus. Their registers are mapped into the external I2C register space so the Raspberry Pi can access them through the Witty Pi 5 Mini I²C slave interface.

4.5 Low-power Hibernation

Low-power hibernation is part of the Witty Pi 5 platform. When the Raspberry Pi is OFF, Witty Pi 5 Mini can put the RP2350 into hibernation mode to significantly reduce standby current.

During hibernation, the RTC alarm, temperature alert, button input, and pulse timer are configured as wake-up sources. When the RP2350 is woken up by the pulse timer, it can briefly blink the white LED, drive the dummy load if configured, check the configured power conditions, and decide whether to power the Raspberry Pi ON or enter hibernation again.

If the emulated USB flash drive is mounted by a host computer, Witty Pi 5 Mini will not enter hibernation immediately, even when the Raspberry Pi is OFF. This is usually the case shortly after you connect the board to your PC via USB. After you manually eject the emulated USB flash drive, Witty Pi 5 Mini will be able to enter hibernation.

If the Raspberry Pi is turned OFF due to a schedule or a preconfigured condition, such as low voltage or a temperature alert, the Witty Pi 5 Mini firmware will eject the emulated USB flash drive and then enter hibernation mode.

With firmware v1.8 or later, Witty Pi 5 Mini can also process an automatic DST change while the Raspberry Pi remains powered off, using the RTC and the RP2350 wake-up mechanism.

4.6 Automatic Daylight Saving Time (DST)

Firmware v1.8 or later supports autonomous daylight saving time (DST). When enabled, the Mini firmware adjusts its RTC at the configured spring and autumn transition times even if the Raspberry Pi is off. DST is disabled by default.

Use the shared wp5 software v5.4.0 or later: Other settings → Automatic DST. The available presets are WET/WEST (UTC+0/+1), CET/CEST (UTC+1/+2), EET/EEST (UTC+2/+3), and common US DST. Alternatively, choose a custom recurring weekday rule or a fixed-date rule that repeats annually. A typical DST shift is +60 minutes.

Select a rule matching the Linux timezone, not merely the country's geographic region. For example, the Netherlands uses CET/CEST, while the United Kingdom uses WET/WEST. Check the Raspberry Pi timezone using timedatectl. A mismatched rule may cause automatic time synchronization to be refused or cause incorrect local-time interpretation.

Before enabling DST, wp5 verifies that the Mini RTC and the Linux local clock agree within approximately three seconds. Enabling DST does not overwrite the RTC. If they disagree, synchronize the clocks separately first. Disabling DST does not validate the Linux timezone or modify RTC time.

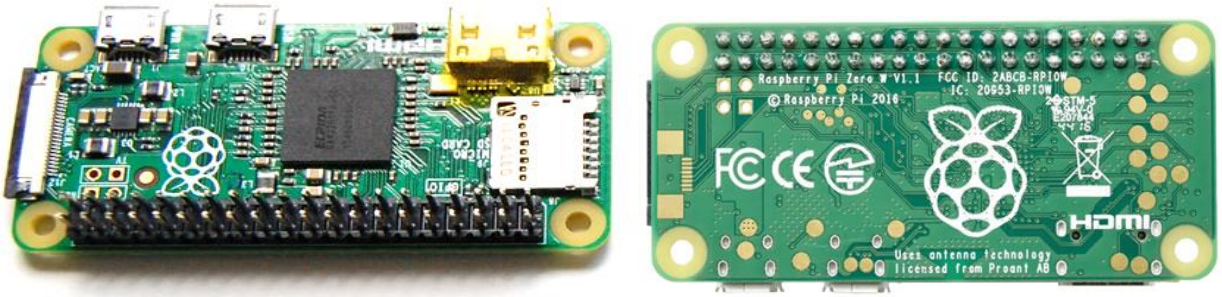
Changing a DST rule can leave previously calculated schedule actions based on the old rule. Reactivate the active schedule script after updating DST. Local times may be skipped during the spring-forward transition or repeated at autumn fall-back; review critical one-time actions near those boundaries.

5. Hardware Installation and Software Setup

5.1 Mounting Witty Pi 5 Mini on Raspberry Pi Zero

If you want to use Witty Pi 5 Mini on a Raspberry Pi Zero without pin header (model V1.2, V1.3, W and 2 W), you will need to solder the 20x2 pin male header on it first, or you can buy one with the header soldered already (model WH and 2 WH).

Remarks: putting the header in place will not be enough, and you have to solder it.

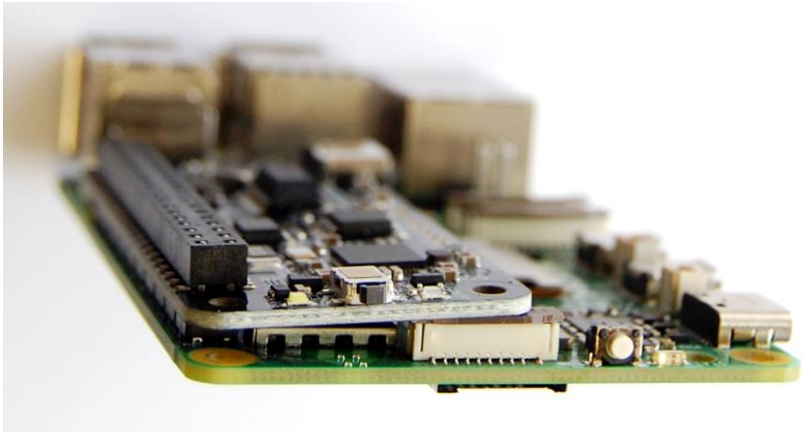


You can simply mount Witty Pi 5 Mini on your Raspberry P Zero's 40-pin header, and it can work just like that. However, if you wish, you can use the plastic screws, spacers and nuts in the package to tightly mount Witty Pi 5 Mini on your Raspberry Pi Zero. Please make sure NOT to put Witty Pi 5 Mini up-side-down, and the 40 pins on Raspberry Pi should connect to the female header via the holes underneath Witty Pi 5 Mini, as shown in figure below.

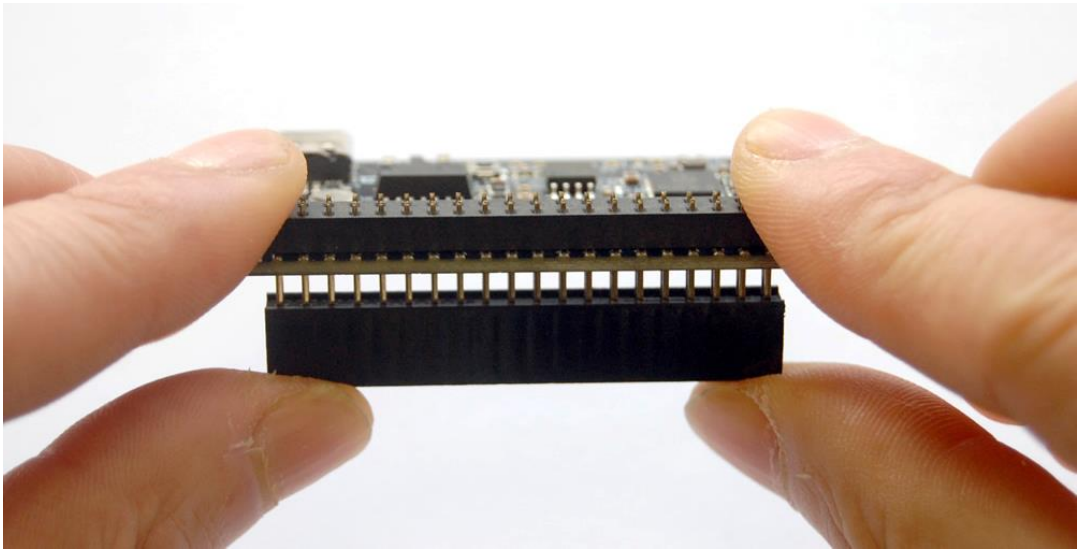


5.2 Mounting Witty Pi 5 Mini on Other Raspberry Pi Models

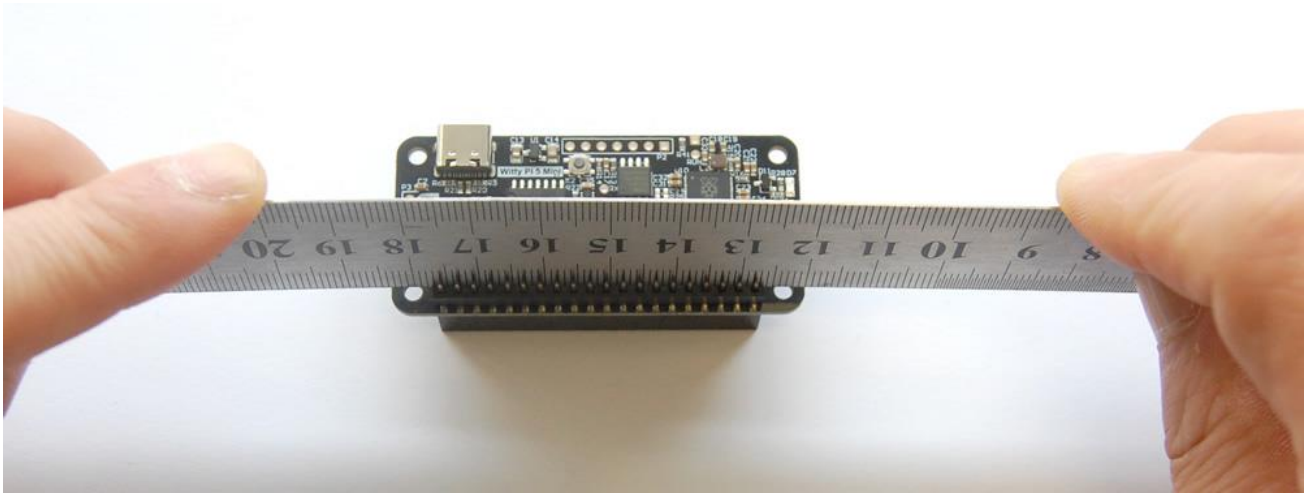
Other Raspberry Pi models (A+, B+, 2B, 3B, 3B+, 4B, 5B) have the 40 pin header soldered already, so it is possible to directly mount Witty Pi 5 Mini over them. However, because of the connector on the edge (PCIE connector on 5B, or display connector on earilier models), you may not be able to firmly connect them.



In order to make reliable connection, we suggest to use a stacking pin header (not included in the package).



Those 40 extra-long pins on stacking header should reach the female header via the holes on the back of Witty Pi 5 Mini (as shown in figure above). You can then place a steel ruler (or something similar) between the two rows of pins and push Witty Pi 5 Mini down until it reaches the plastic of the stacking header, as shown in figure below:



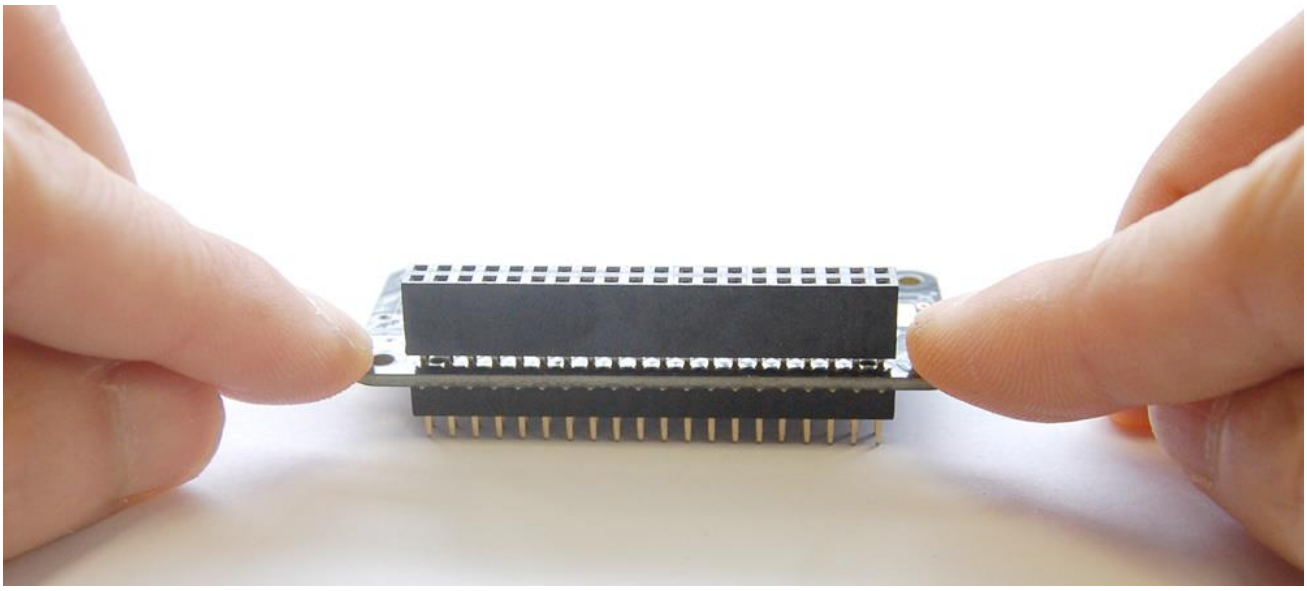
After mounting Witty Pi 5 Mini on your Raspberry Pi, and connect the power supply to the USB type C connector on Witty Pi 5 Mini, you can see **the white LED blinking, which means it is standing by**. **Please only connect power supply to Witty Pi and do not connect power supply to your Raspberry Pi directly**, otherwise Witty Pi will not be able to control the power to Raspberry Pi.



Now your Witty Pi 5 Mini is ready to go.

If you want to take off the stacking header from Witty Pi 5 Mini, the proper way is to put your Witty Pi 5 Mini up-side-down, and let the 40 pins contact a flat surface (e.g. desktop), hold the two edges

of the board and press down. That way the stacking header will be taken off safely. See the figure below:



5.3 Software Setup

Remarks: The wp5 software package is shared by the Witty Pi 5 family and is developed and tested under Raspberry Pi OS.

Please make sure to enable the I^2C interface on your Raspberry Pi, because the wp5 software communicates with Witty Pi 5 Mini via I^2C.

5.3.1 Installation with Internet Connection

If your Raspberry Pi has Internet connection, you may run the following command to download the latest .deb package:

```
pi@raspberrypi:~ $ wget https://www.uugear.com/repo/WittyPi5/wp5_latest.deb
```

Then install it with:

```
pi@raspberrypi:~ $ sudo apt install ./wp5_latest.deb
```

If a newer version of the software becomes available, you can repeat the above steps to update it.

Automatic DST configuration and voltage-confirmation settings in wp5 v5.4.0 require Mini firmware reporting v1.8 or later. The software and firmware have separate version numbers; updating the software alone does not add firmware features.

5.3.2 Installation without Internet Connection

If your Raspberry Pi does not have Internet connection. You may still download the latest .deb package on another machine, and then copy the file to your Raspberry Pi to install it.

5.3.3 Uninstall the Software

To uninstall the software in the future, run:

```
pi@raspberrypi:~ $ sudo apt remove wp5
```

5.4 Software Usage

```
pi@raspberrypi:~ $ wp5
================================================================================
|                                                                              |
|   Witty Pi - Realtime Clock + Power Management for Raspberry Pi              |
|                                                                              |
|            < Version 5.4.0 >    by Dun Cat B.V. (UUGear)                    |
|                                                                              |
================================================================================
--------------------------------------------------------------------------------
  Model: Witty Pi 5 Mini (firmware v1.8)
  Temperature: 30.625°C / 87.125°F
  V-USB: 5.084V   V-OUT: 5.021V   I-OUT: 0.485A
  SYS Time: 2026-11-17 14:40:18
  RTC Time: 2026-11-17 14:40:18
--------------------------------------------------------------------------------
  1. Write system time to RTC
  2. Write RTC time to system
  3. Synchronize with network time
  4. Schedule next shutdown
  5. Schedule next startup
  6. Manage schedule scripts...
  7. Set low voltage threshold
  8. Set recovery voltage threshold
  9. Set over temperature action
 10. Set below temperature action
 11. Other settings...
 12. Reset data...
 13. Administrate...
 14. Exit
 Please input 1~14:
```

To run the software, type `wp5` in any directory and press ENTER.

The program displays the current device status, followed by a list of available options. To select an option, type the corresponding number and press ENTER.

The menu example uses the shared wp5 software v5.4.0 and Witty Pi 5 Mini hardware. A DST status line appears beneath RTC Time only when automatic DST is enabled. Displayed times, voltages and firmware versions are illustrative.

5.4.1 Time Configuration

Option 1. Write system time to RTC copies the current system time of your Raspberry Pi to the RTC on Witty Pi 5 Mini.

Option 2. Write RTC time to system copies the time stored in the RTC on Witty Pi 5 Mini to your Raspberry Pi's system time.

Option 3. Synchronize with network time gets the current time from the Internet, sets it as the Raspberry Pi's system time, and then copies it to the RTC on Witty Pi 5 Mini.

In wp5 v5.4.0, the network HTTP Date header is interpreted as UTC before updating system time. When firmware DST is enabled, RTC-to-system synchronization also uses the DST-applied flag to distinguish the duplicated local hour during autumn fall-back. If Linux timezone settings do not match the configured firmware rule, synchronization may be rejected rather than silently setting an incorrect time.

5.4.2 Scheduled Shutdown and Startup

Option `4. Schedule next shutdown` allows you to specify the next shutdown time in the format DD HH:MM:SS.

Option `5. Schedule next startup` allows you to specify the next startup time in the format DD HH:MM:SS.

Option `6. Manage schedule scripts...` opens the following sub-menu:

```
--------------------------------------------------------------------------------
  Manage schedule scripts...
--------------------------------------------------------------------------------
  [1] Upload schedule script...
  [2] Activate schedule script...
  [3] Download schedule script...
  [4] Delete schedule script...
  [5] Return to main menu
  Please input 1~5:
```

Item [1] Upload schedule script... uploads a schedule script from the current directory on your Raspberry Pi to the schedule directory on Witty Pi 5 Mini.

Item [2] Activate schedule script... activates a schedule script that already exists in the schedule directory on Witty Pi 5 Mini. Once activated, the script will be used to calculate future startup and shutdown times.

Item [3] Download schedule script... downloads a schedule script from Witty Pi 5 Mini to the current directory on your Raspberry Pi.

Item [4] Delete schedule script... deletes a schedule script from the schedule directory on Witty Pi 5 Mini.

Item `[5] Return to main menu` returns to the main menu without making further changes.

5.4.3 Voltage-Related Behavior

Option 7. Set low voltage threshold allows you to specify the low-voltage threshold for VIN. On Witty Pi 5 Mini this threshold is used when the configured power source is V-IN via external DC/DC. If the measured upstream VIN drops below the threshold, Witty Pi 5 Mini will trigger a safe shutdown.

Option 8. Set recovery voltage threshold allows you to specify the recovery-voltage threshold for VIN. If the previous shutdown was caused by low VIN while using an external DC/DC converter, Witty Pi 5 Mini can turn on the Raspberry Pi again when VIN rises above this threshold.

When V-USB is configured as the actual power source, the VIN low-voltage and recovery thresholds do not control shutdown or startup behavior.

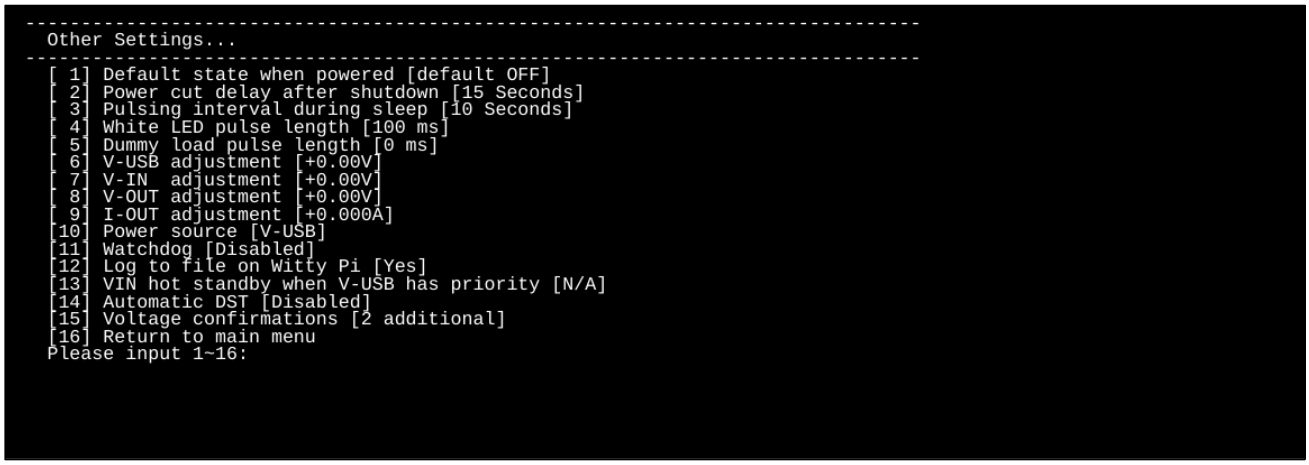
5.4.4 Temperature-Related Behavior

Option `9. Set over temperature action` allows you to specify an action to be taken when the measured temperature rises above a configured threshold. The action can be either startup or shutdown.

Option `10. Set below temperature action` allows you to specify an action to be taken when the measured temperature drops below a configured threshold. The action can be either startup or shutdown.

5.4.5 Other Settings

Option `11. Other settings...` opens the following sub-menu:



Item [1] Default state when powered sets the default state of the Raspberry Pi when Witty Pi 5 Mini is newly powered. By default, Witty Pi 5 Mini will not automatically boot the Raspberry Pi. If you want the Raspberry Pi to boot automatically when power is connected to Witty Pi 5 Mini, you can enable this option and specify a startup delay in seconds.

Item `[ 2] Power cut delay after shutdown` sets the delay, in seconds, between shutting down the Raspberry Pi and cutting its power. This delay should be long enough for the operating system to shut down safely.

Item [3] Pulsing interval during sleep sets the interval between sleep pulses. These pulses are used while Witty Pi 5 Mini is in low-power sleep mode.

Item `[ 4] White LED pulse length` sets how long the white LED will pulse during each sleep-pulse cycle. A longer pulse is easier to see, but it also consumes slightly more power.

Item [5] Dummy load pulse length sets how long the dummy load will be enabled during each sleep-pulse cycle. This can be useful when Witty Pi 5 Mini is powered by a USB power bank that requires a minimum load to stay on. Setting this value to 0 ms disables the dummy load pulse.

Item `[ 6] V-USB adjustment` sets a calibration offset for the measured V-USB voltage.

Item `[ 7] V-IN adjustment` sets a calibration offset for the measured VIN voltage.

Item `[ 8] V-OUT adjustment` sets a calibration offset for the measured V-OUT voltage.

Item `[ 9] I-OUT adjustment` sets a calibration offset for the measured output current.

Item `[10] Power source` specifies the actual power source used by Witty Pi 5 Mini. This is not a priority setting. Select V-USB when the common 5V rail is supplied from the USB-C input, or select V-IN via external DC/DC when an external converter supplies the 5V rail.

When V-USB is selected, the firmware drives GP12_EN low. If an external DC/DC converter has an active-high enable input compatible with 3.3V logic and it is wired to GP12_EN, the converter should remain disabled while the Mini is operating from USB-C power.

When V-IN via external DC/DC is selected, the firmware enables GP12_EN and interprets VIN as the upstream source voltage for low-voltage shutdown and recovery. The setting must match the actual wiring. Changing it may immediately enable or disable an external DC/DC converter, if GP12_EN is actually in use for enabling the DC/DC converter.

Item [11] Watchdog enables or disables the watchdog function. When enabled, Witty Pi 5 Mini can monitor the Raspberry Pi and take recovery action if the watchdog is not serviced as expected.

Item [12] Log to file on Witty Pi enables or disables logging to a file stored on Witty Pi 5 Mini.

Item `[13] VIN hot standby when V-USB has priority` is not applicable to Witty Pi 5 Mini and is shown as N/A. The corresponding configuration register is retained for protocol compatibility but is ignored by the Mini firmware.

Item [14] Automatic DST (firmware v1.8+) opens the DST settings submenu. It displays the Mini's current firmware DST rule, the Linux timezone and local time, and a recommended preset when available. Choose one of the following:

[1] Disable automatic DST
[2] WET/WEST (UTC+0/+1): Mar last Sun 01:00 → Oct last Sun 02:00
[3] CET/CEST (UTC+1/+2): Mar last Sun 02:00 → Oct last Sun 03:00
[4] EET/EEST (UTC+2/+3): Mar last Sun 03:00 → Oct last Sun 04:00
[5] US DST: Mar 2nd Sun 02:00 → Nov 1st Sun 02:00
[6] Custom recurring rule (month, week, weekday)
[7] Custom fixed-date rule (month, date; repeats yearly)
[8] Cancel

The preset rules shift the RTC by 60 minutes. Custom rules support a shift of 1–127 minutes. Recurring rules use a week-of-month and weekday; fixed-date rules use the same month and calendar date every year. These are annual rules, not one-time scheduling events.
When enabling DST, wp5 requires the selected rule to match the Linux timezone and the RTC and Linux local times to agree within three seconds. It does not set the RTC while configuring DST. Disabling DST does not perform these checks. After changing the DST rule, reactivate any active schedule script.
Item [15] Voltage confirmations (firmware v1.8+) sets the additional immediate readings taken after the initial threshold crossing. The valid range is 0–5 (default 2). Setting 0 preserves the original immediate action; setting 2 requires the initial crossing and two further readings. On Mini this applies to monitored VIN in the external DC/DC configuration; it does not create a second isolated power path.

Item [16] Return to main menu returns to the main menu without making further changes.

5.4.6 Reset Data

Option `12. Reset data...` opens the following sub-menu:

```
--------------------------------------------------------------------------------
  Reset data...
--------------------------------------------------------------------------------
  [ 1] Clear scheduled startup time
  [ 2] Clear scheduled shutdown time
  [ 3] Stop using schedule script
  [ 4] Clear low-voltage threshold
  [ 5] Clear recovery-voltage threshold
  [ 6] Clear over-temperature action
  [ 7] Clear below-temperature action
  [ 8] Reset all configuration values
  [ 9] Perform all actions above
  [10] Return to main menu
  Please input 1~10:
```

Item `[ 1] Clear scheduled startup time` clears the currently scheduled one-time startup time.

Item `[ 2] Clear scheduled shutdown time` clears the currently scheduled one-time shutdown time.

Item `[ 3] Stop using schedule script` deactivates the currently active schedule script. This does not necessarily delete the script file itself.

Item `[ 4] Clear low-voltage threshold` clears the configured low-voltage threshold. After this value is cleared, low-voltage shutdown will no longer be triggered by this threshold.

Item [5] Clear recovery-voltage threshold clears the configured recovery-voltage threshold. After this value is cleared, Witty Pi 5 Mini will no longer use this threshold to automatically turn on the Raspberry Pi after a low-voltage shutdown.

Item `[ 6] Clear over-temperature action` clears the configured action for over-temperature events.

Item `[ 7] Clear below-temperature action` clears the configured action for below-temperature events.

Item `[ 8] Reset all configuration values` restores the configuration values to their default settings.

Item `[ 9] Perform all actions above` performs all reset actions listed above.

Item `[10] Return to main menu` returns to the main menu without making further changes.

5.4.7 Administrate

Option `13. Administrate...` opens the following sub-menu:

```
--------------------------------------------------------------------------------
  Administrate...
--------------------------------------------------------------------------------
  [1] Print product information in log
  [2] Format Witty Pi disk
  [3] Reset RTC
  [4] Turn on/off ID EEPROM write protection
  [5] Synchronize configuration to file
  [6] Save log to file
  [7] Download log file...
  [8] Load and generate schedule scripts
  [9] Return to main menu
  Please input 1~9:
```

Item `[1] Print product information in log` prints product-related information to the log. This can be useful when you need to check or report the hardware, firmware, or software information of your Witty Pi 5 Mini.

Item `[2] Format Witty Pi disk` formats the storage area used by Witty Pi 5 Mini and recreates the default file structure. This operation will remove existing files stored on Witty Pi 5 Mini, such as logs, configuration files, or schedule scripts. Use it only when you need to reinitialize the Witty Pi disk.

Item `[3] Reset RTC` resets the RTC on Witty Pi 5 Mini. After resetting the RTC, you may need to write the correct time to it again.

Item `[4] Turn on/off ID EEPROM write protection` is part of the shared administrative menu. Witty Pi 5 Mini has no ID EEPROM populated by default, so this function is normally not applicable. It is relevant only if a compatible and correctly configured ID EEPROM has been fitted.

Item `[5] Synchronize configuration to file` writes the current configuration values in RAM to the configuration file on Witty Pi 5 Mini. This is useful when you want to make sure the configuration file reflects the current firmware settings.

Item `[6] Save log to file` write pending log content in RAM into the log file on Witty Pi 5 Mini.

Item `[7] Download log file...` downloads the log file from Witty Pi 5 Mini to the current directory on your Raspberry Pi.

Item `[8] Load and generate schedule scripts` loads schedule script definitions and generates schedule script files that can be used by Witty Pi 5 Mini.

Item `[9] Return to main menu` returns to the main menu without making further changes.

6. Update Firmware

Firmware updates on Witty Pi 5 Mini use the same RP2350 USB boot mechanism as the rest of the Witty Pi 5 family. Compared with Witty Pi 4 Mini, no dedicated programmer (such as USBasp) is required, and you no longer need to simulate one using the Raspberry Pi.

6.1 Get the New Firmware

You can download the latest firmware from the Releases page on GitHub.

Just download the file with "mini" in file name and with a .uf2 extension.

Make sure the UF2 image is built for Witty Pi 5 Mini rather than Witty Pi 5 HAT+. The devices share a source repository but use different hardware targets.

Of course, you may also compile the firmware yourself from the source code if you prefer.

6.2 Upload the New Firmware

To upload the firmware:

1. Make sure Witty Pi 5 Mini is not powered.
2. Press and hold the BOOTSEL button.
3. While holding the button, connect the USB Type-C port on Witty Pi 5 Mini to your computer.

Your computer will detect a virtual USB flash drive named "RP2350".

Drag and drop the .uf2 firmware file into this virtual drive.



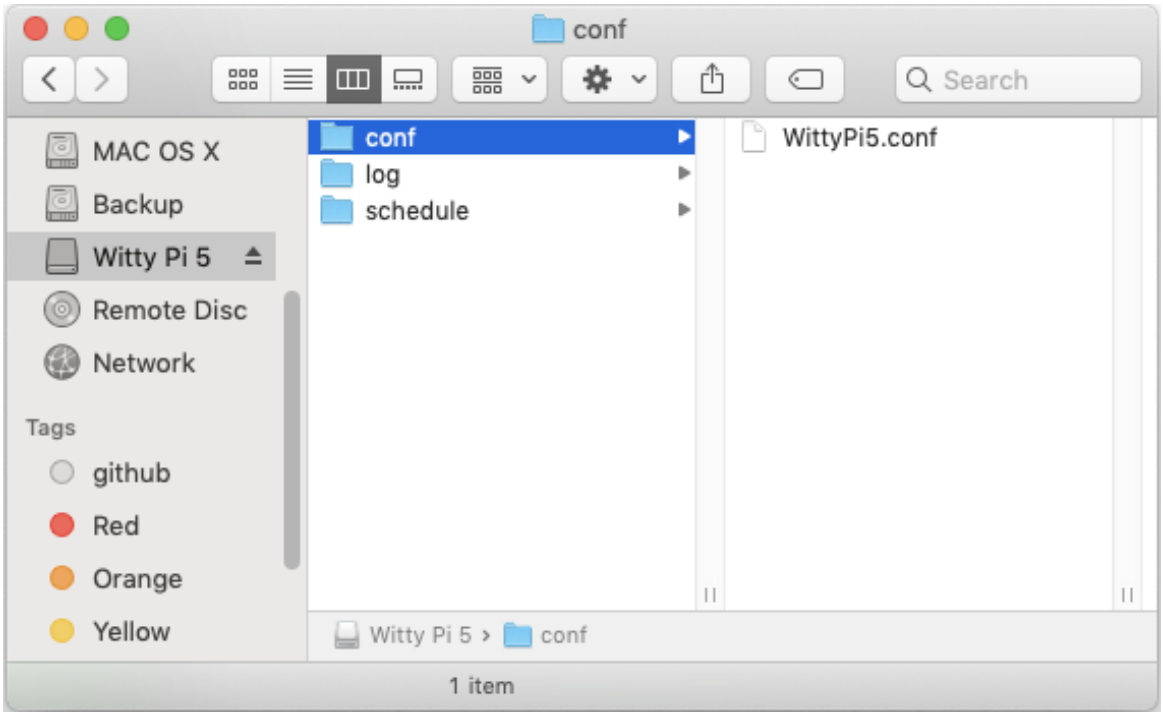
Alternatively, you may also copy it using the command line – the result is the same.

Once the upload is complete, the virtual RP2350 drive will automatically eject, and your new firmware is now installed.

After updating, launch wp5 to verify the firmware version reported by the board. Automatic DST and voltage confirmations require a Mini firmware image reporting v1.8 or later.

7. The Emulated USB Flash Drive

When you connect Witty Pi 5 Mini to a computer using a USB cable (without holding the BOOTSEL button), your system will detect the emulated USB flash drive provided by the Witty Pi 5 platform. This virtual drive has a 14MB capacity and uses the FAT12 file system.



The drive contains the following three directories:

- conf – stores Witty Pi 5 Mini's configuration files.
- log – contains Witty Pi 5 Mini's log files.
- schedule – holds the available schedule script files.

Firmware may temporarily make the FAT12 USB drive unavailable while storing configuration, schedules or logs to flash.

Important Notes

The Witty Pi 5 Mini firmware may also access the FAT12 disk, for example to write logs or save configuration files. To avoid conflicts between the host computer and the firmware, it will mark the USB drive as “not ready” before performing write operations.

In addition, after the Raspberry Pi is turned OFF, the RP2350 may enter low-power hibernation mode. Before entering hibernation, the firmware will also make the emulated USB drive unavailable to the host computer. This is why you may sometimes notice that the drive unexpectedly becomes inaccessible from your computer.

Safe Removal Procedure

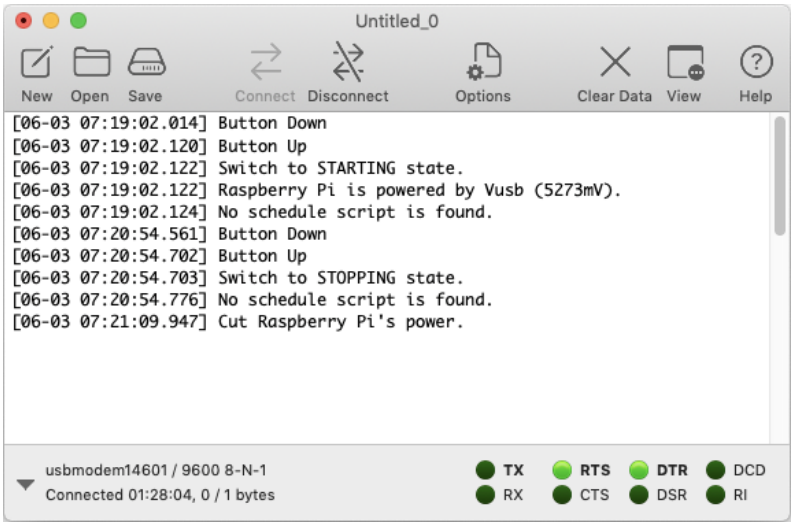
After you finish working with the files on the USB drive, it is strongly recommended to eject the drive from your operating system before physically disconnecting the cable.

If you notice the white LED on Witty Pi 5 Mini blinks rapidly after ejecting the drive, please wait a few moments. The LED will turn off completely once all internal operations are done. Only after the LED is off, you can safely disconnect the USB cable.

8. The Emulated USB Serial Device

When you connect Witty Pi 5 Mini to your computer via a USB cable, in addition to the USB flash drive, your system will also detect a USB serial device.

You can use your preferred serial terminal application (e.g., PuTTY, minicom, screen, CoolTerm etc.) to connect to this serial port and monitor the real-time logs from Witty Pi 5 Mini.



This is especially useful for debugging purposes.

The USB serial device will remain connected even when the emulated USB flash drive becomes inaccessible because the firmware is accessing the FAT12 file system. This allows you to continue monitoring real-time logs while the firmware is writing logs, saving configuration files, or performing other file-system operations.

After the Raspberry Pi is turned OFF, the RP2350 may enter low-power hibernation mode. In this case, the USB serial device will also be disconnected and become inaccessible until the RP2350 wakes up again.

Power Supply Consideration

Please note that many computer USB ports may not be able to supply enough current for a Raspberry Pi and its peripherals to boot reliably.

If you plan to use the USB serial interface with Witty Pi 5 Mini:

- Use the USB Type-C connection as the 5V source only when the computer or USB supply can safely provide the current required by the complete system.

or

- Use an external DC/DC converter to power the common 5V rail, and do not also connect USB-C V_{BUS} to an independently powered USB host unless suitable external isolation or power-path management is provided.

Powering the Raspberry Pi directly via its USB-C connector may be useful for debugging, but it bypasses Witty Pi 5 Mini power control, and can not be used as a normal power configuration.

9. About Schedule Script

In addition to manually setting the next startup or shutdown time, you can define a more complex power schedule for your Raspberry Pi using Schedule Scripts.

These scripts are plain text files stored in the onboard flash memory of Witty Pi 5 Mini and are executed directly by the firmware, following the common Witty Pi 5 schedule architecture.

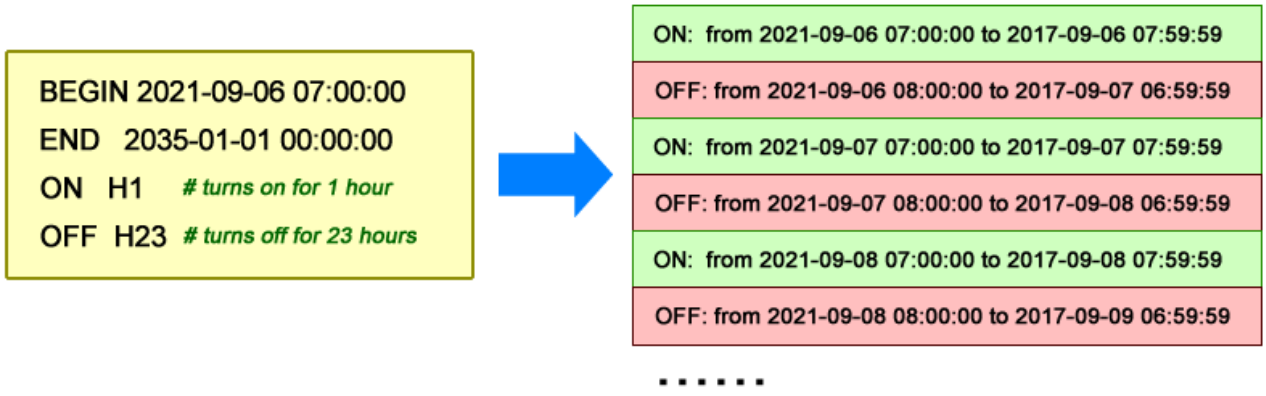
The Witty Pi 5 family supports three schedule-script formats: .wpi, .act, and .skd.

9.1 .wpi File

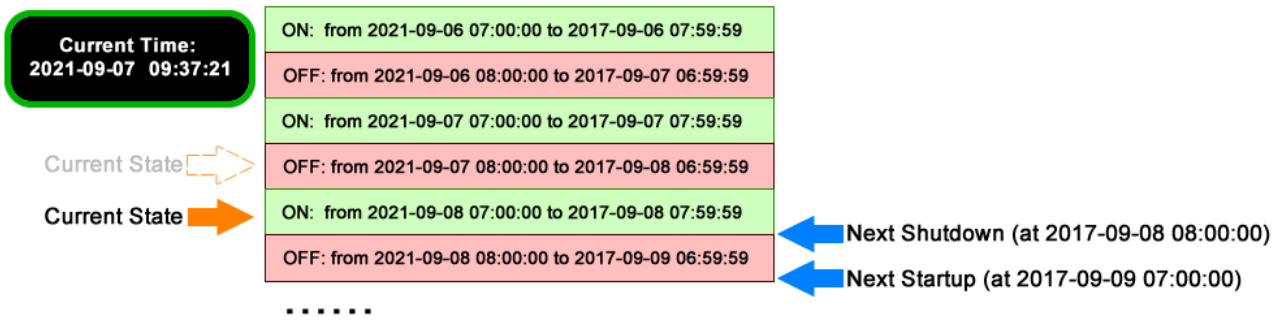
The .wpi format originates from earlier Witty Pi generations.

Witty Pi 5 Mini retains this format for compatibility. As with other Witty Pi 5 family members, the script is interpreted by the firmware instead of software running on the Raspberry Pi.

A .wpi file defines a looped schedule with a begin time, end time, and repeating ON/OFF states within that range. This effectively divides the time axis into alternating ON and OFF states.



With .wpi file you may define a long schedule with just a few lines. However, such simplicity sometimes sacrifices a bit flexibility. For example, if you request the next shutdown time (as the end of ON state) at a moment when the Pi is in an OFF state, you will get the end time of the **next ON state**.



This behavior, known as “shifting ON state”, can be confusing to some users—it may require extra effort to schedule a shutdown at your exact desired moment.

If you create unusually long .wpi schedules, the firmware may truncate the generated actions at its internal limit rather than reject the whole script. Check the resulting schedule and firmware log for truncation notices.

9.2 .act File

The .act format is part of the Witty Pi 5 family schedule architecture.

Compared to .wpi, the .act file has a simpler structure: it is a linear list, where each line specifies a startup or shutdown time.

Its advantage is flexibility: you can schedule each operation at any specific time without needing to calculate durations or loop timings.

The downside is that for a long schedule, you may need to write many lines to cover all actions.

9.3 .skd File

The .skd file can be seen as a compressed version of the .act file.

It trades off human readability for compactness, reducing the overall size of the script.

In fact, .skd is the format actually executed by Witty Pi 5 firmware. Both .wpi and .act scripts are converted into .skd format by the firmware before execution.

When a .wpi file is converted to .skd, unnecessary loop iterations are skipped, and only future ON/OFF transitions are translated into actual operations with timestamps.

9.4 Run a Schedule Script

Once a schedule script is selected to run, the firmware will parse it and determine the next startup and shutdown times (if any) from the current time.

These times will be saved into the configuration, and based on the current state of the Raspberry Pi (ON or OFF), the firmware will configure the RTC with the corresponding alarm.

When the RTC alarm is triggered, the firmware will perform the scheduled operation to power ON or OFF the Raspberry Pi.

With automatic DST enabled, the Mini uses its configured rule to interpret local schedule times. Some local times do not occur in spring, and others occur twice in autumn. For critical one-time schedules near a transition, check the outcome and reactivate the script after changing DST rules.

9.5 Write Your Own Schedule Script

A schedule script is actually a plain text file, and you can use any text editor to create and edit it. You can then place your own .wpi file into the “schedule” directory in the USB drive emulated by Witty Pi 5 Mini, so it can be listed and chosen via the shared wp5 software. It is also possible to upload the schedule script file into the emulated USB drive from the software.

9.5.1 .wpi Format

Below is a very simple .wpi schedule script and it will keep your Raspberry Pi on for 5 minutes in every 20 minutes.

```
# Turn on Raspberry Pi for 5 minutes, in every 20 minutes
BEGIN  2025-06-01 00:00:00
END    2025-07-31 23:59:59
ON     M5      # keep ON state for 5 minutes
OFF    M15     # keep OFF state for 15 minutes
```

Like many other scripting languages, .wpi schedule script also uses "#" to make single line comment.

The first two lines define the time range for executing the script. Please make sure to use the correct time format (YYYY-mm-DD HH:MM:SS). You can use one or more white characters (space or tab) between BEGIN/END and the time string.

The rest of the script defines the states in the loop. It could be "ON" or "OFF", and you should define at least one "ON" and one "OFF" states in the loop. The ON and OFF states are used in pair.

You should also specify the duration of each state. You can do so by putting one or more parameters after ON/OFF text, separated by space or tab. Each parameter starts with a capital letter and follows by a number, where the capital letter is the unit of time:

- D = Days (D2 means 2 days)
- H = Hours (H3 means 3 hours)
- M = Minutes (M15 means 15 minutes)
- S = Seconds (S30 means 30 seconds)

For example, if you wish to define an ON state for one and a half hours, you can write:

```
ON    H1    M30
```

When the script engine executes this line, it will schedule a shutdown at the end of the ON state.

If you wish to define an OFF state for two days, you can write:

```
OFF    D2
```

When this line gets executed, a startup will be scheduled at the end of the OFF state.

Sometimes you may want your own program to shut down Raspberry Pi as soon as the job is done. The Witty Pi 4 family supported the "WAIT" syntax to skip the next shutdown. The Witty Pi 5 family, including Witty Pi 5 Mini, no longer supports this syntax.

In fact, there is no longer a need to skip any schedules in the schedule script: the shutdown and startup are always scheduled, but you (or your own program) can shutdown Raspberry Pi before the scheduled shutdown is due.

9.5.2 .act Format

Below is an example of .act file. It is a linear list, and each line is an action (UP or DN) with its associated time.

```
# Example of .act schedule script
UP 2025-06-01 00:00:00
DN 2025-06-01 00:05:00
UP 2025-06-01 00:20:00
DN 2025-06-01 00:25:00
UP 2025-06-01 00:40:00
DN 2025-06-01 00:45:00
```

You can also use # to start a comment in .act file. This format is so simple, that the only thing to take care is the correct time format (YYYY-mm-DD HH:MM:SS).

9.5.3 .skd Format

Although .skd is the script format actually executed by Witty Pi 5 firmware, you usually do not need to write it yourself because the firmware automatically converts .wpi or .act files into .skd files.

However, knowing the .skd format may still be useful. For example, you may want to develop software that directly generates a .skd file for Witty Pi 5 Mini to run.

Below is an example. The .skd file is also a linear list, and each line is an action with its associated time. Different than .act file, the action type uses only one letter: “U” for “UP” and “D” for “Down”, also the time is converted to a timestamp value (total seconds since the beginning of year 2000).

```
# Example of .skd schedule script
U707356800
D707357100
U707358000
D707358300
U707359200
D707359500
```

9.6 Store Schedule Scripts in Witty Pi 5 Mini

Schedule script files should be stored in the schedule directory on the emulated USB flash drive of Witty Pi 5 Mini.

You can copy your schedule script files into this directory after connecting Witty Pi 5 Mini to your PC with a USB-C cable.

Alternatively, you can upload a schedule script file from the current directory on your Raspberry Pi to the schedule directory on Witty Pi 5 Mini by using the software menu:

Manage schedule scripts... → Upload schedule script....

Once a schedule script file is stored in the schedule directory on Witty Pi 5 Mini, it can be activated via the wp5 software. After activation, the firmware will parse the script, find the next applicable startup or shutdown event, and configure the corresponding alarm.

Activating the script on Witty Pi 5 may take a while, you will see the white LED quickly blinking when it is on work.

9.7 Using Schedule Script Generator

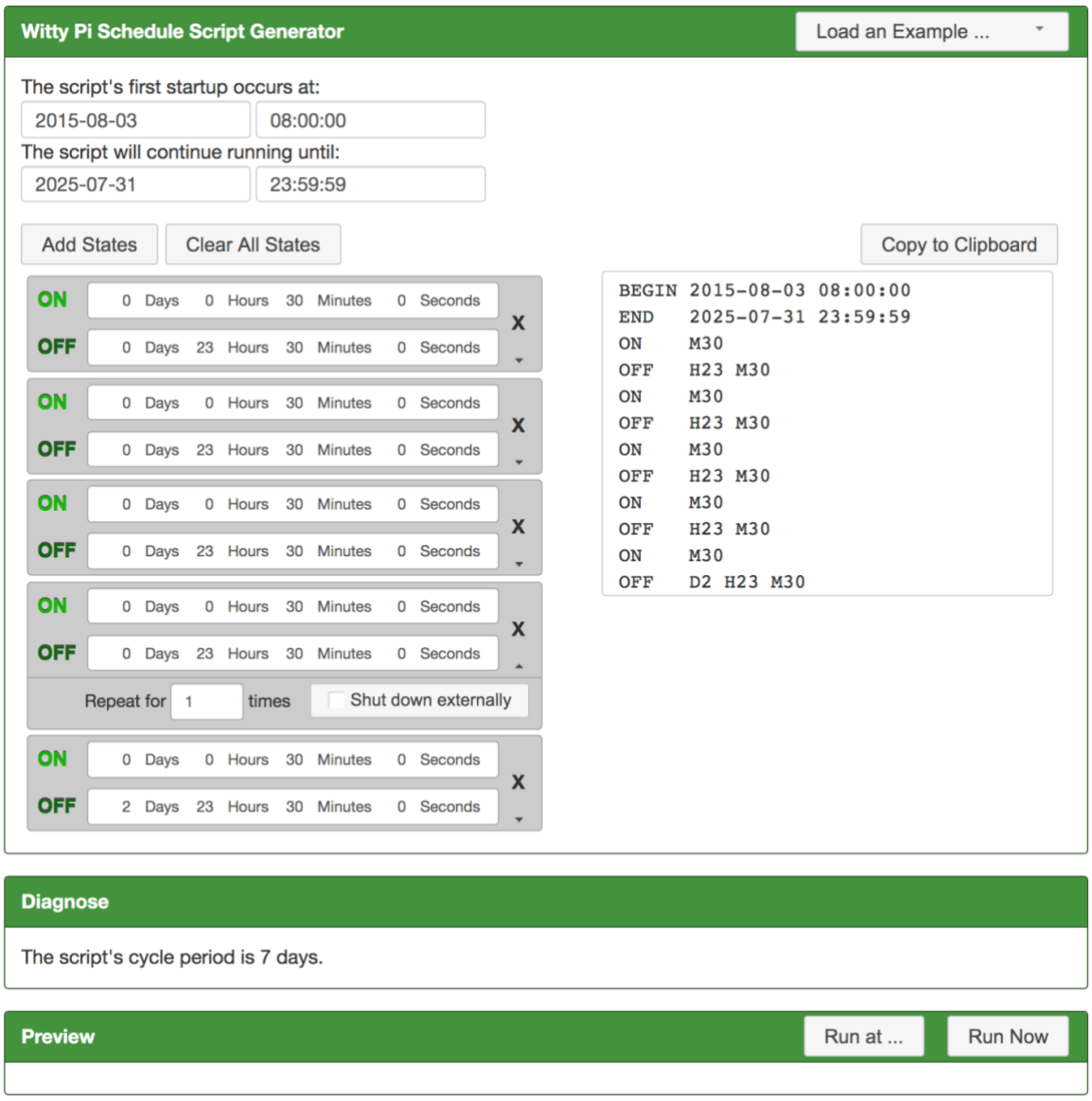
You may also use our web application to create your schedule script. At this moment only .wpi format is supported.

Just simply open this URL in your web browser:

https://www.uugear.com/app/wittypi-scriptgen/

This web application allows you to visually create the schedule script, and it immediately generate the final code on the right.

You can also click the "Run Now" button to preview how the schedule script will work. Alternatively, you can click the "Run at…" button and specify the moment to run the script.



10. Additional Interfaces

Witty Pi 5 Mini provides a number of unpopulated interfaces in the form of solder pads. These are usually labelled on the back side of the board.

10.1 Serial Wire Debug (SWD) Port for RP2350



This 3-pin header is the Serial Wire Debug (SWD) interface for RP2350. You can solder a connector here to connect a Raspberry Pi Debug Probe for firmware debugging purposes.

10.2 Unpopulated RP2350 Reset Header



This 2-pin header (RUN and GND) allows you to connect an external reset button to the RP2350 microcontroller.

10.3 Unpopulated Serial Port Connector



This 3-pin header (RXD, GND, TXD) allows you to connect a USB-to-Serial adapter, or link it to a Raspberry Pi Debug Probe for serial communication with RP2350.

10.4 Unpopulated 7-Pin Header



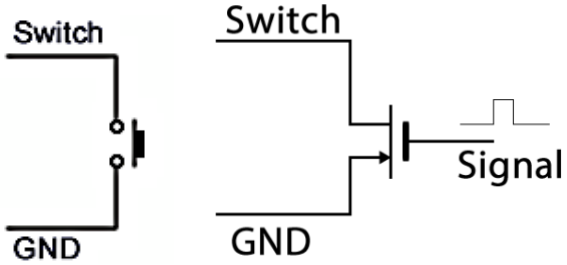
This header exposes several internal signals:

GND – Ground (connected to other GND points).

1.1V – The 1.1V power rail used by RP2350.

SWITCH – Signal from the onboard button, connected to RP2350 GPIO3. Pulled HIGH, goes LOW when pressed.

If you want to connect your own switch, you may wire the two leads to SWITCH and GND pins.



Alternatively, if you wish to trigger Witty Pi 5 with external signal, you can use an N-MOSFET to achieve this. The signal should be a positive pulse. Processing a pulse will be equal to tapping the switch once. This will turn on your Raspberry Pi when it is off, or turn off your Raspberry Pi when it is on.

TMP.A – Temperature sensor alert output, connected to RP2350 GPIO9. Default is HIGH, goes LOW on alert.

RTC.A – RTC alarm output pin, connected to RP2350 GPIO8. Default is HIGH (3.3V), goes LOW when an alarm occurs.

LED – Connected to RP2350 GPIO2 through a 2.2kΩ resistor. Intended for driving an external LED (not yet used in firmware).

3.3V – Power supply used by RP2350, RTC, and temperature sensor (not connected to Raspberry Pi's 3.3V rail).

10.5 Solder Pads for GP12, GP19, GP20 and GP21



These pads break out some GPIO pins from RP2350.

GP12_EN is an active-high 3.3V-logic output controlled by the Mini firmware. Connect it only to an external converter enable input with a compatible logic-level specification; it is not a power output.

GP19, GP20 and GP21 are currently not used by firmware.

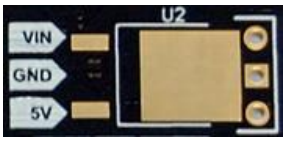
10.6 Footprint for ID EEPROM



Witty Pi 5 Mini does not include a populated HAT ID EEPROM. An optional footprint is provided for an appropriate I²C EEPROM (for example, CAT24C32); adding the component alone does not make the board HAT+ compliant. Correct contents and electrical compatibility are also required.

If installing an ID EEPROM, also populate the reserved I²C pull-up resistors R10 and R12 (4.7 kΩ each) and program valid identification data.

10.7 Power header / footprint for external DC/DC converter



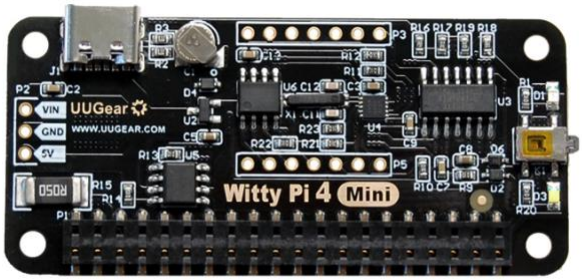
There are pads reserved for connecting external DC/DC converter.

Switching DC/DC converter is preferred for better performance,

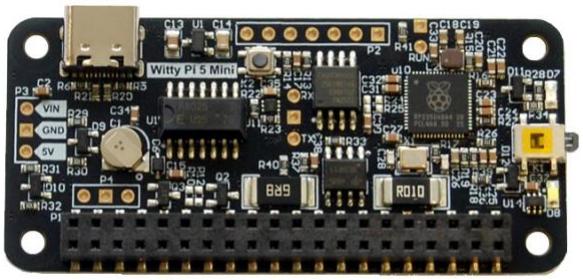
For low-dropout, low-current applications, an LDO (such as LM29150) can also be used directly.

11. Migrating from Witty Pi 4 Mini to Witty Pi 5 Mini

Witty Pi 5 Mini is the successor to Witty Pi 4 Mini, but it is not a drop-in electrical replacement. It moves the Mini model onto the common Witty Pi 5 firmware and software platform, while also changing the power architecture and auxiliary interfaces. Verify the final board layout and connector assignments before reusing mechanical parts or external wiring from an existing Witty Pi 4 Mini installation.



Witty Pi 4 Mini



Witty Pi 5 Mini

The software used for Witty Pi 4 Mini is not compatible with Witty Pi 5 Mini. If you are upgrading, install the wp5 software package shared by the Witty Pi 5 family.

The .wpi schedule scripts used on Witty Pi 4 Mini are still compatible with Witty Pi 5 Mini. To use them, copy them into the schedule directory inside the emulated USB drive. Witty Pi 5 Mini will look for a file named schedule.wpi and execute it after converting it to a .skd file. You may also use the software interface to select and execute a different script.

If your project communicates with Witty Pi via I²C, please be aware the I²C address has changed:

- Witty Pi 4 Mini used address 0x08
- Witty Pi 5 Mini uses I²C address 0x51 (product/firmware ID is 0x52)

The Mini product ID (0x52 in register 0) must not be confused with the I²C slave bus address (0x51 by default).

In addition, the set of I2C registers provided by Witty Pi 5 Mini is different from that of Witty Pi 4 Mini. Please refer to the What I2C Registers are Provided by Witty Pi 5 Mini section for details.

Witty Pi 5 Mini uses the I2C interface for normal communication with the Raspberry Pi and does not rely on the legacy GPIO-based control signals used by Witty Pi 4 Mini.

If your project uses external buttons or other auxiliary signals, do not assume that wiring from Witty Pi 4 Mini can be reused directly. Refer to the final Witty Pi 5 Mini board layout and interface definitions before connecting external signals.

12. Log Files in the Witty Pi 5 Family

On Witty Pi 4 Mini and earlier generations, logging relied mainly on software running on the Raspberry Pi. If the Pi failed to start, the most useful diagnostic information might therefore be unavailable.

Dual-layer logging is a standard feature of the Witty Pi 5 platform. Witty Pi 5 Mini inherits this system, with logs generated both by the shared wp5 software on the Raspberry Pi and by the firmware running on the Witty Pi board.

12.1 Software Log (on Raspberry Pi)

The shared Witty Pi 5 software daemon, wp5d, automatically starts after the Raspberry Pi boots. It maintains its own log file at:

/var/log/wp5d.log

This file records events such as system startup, shutdown requests, and power management decisions made by the daemon etc.

Software v5.3.x and later improve retry handling when flash writes temporarily suspend external I²C access. If automatic time synchronization is rejected, inspect wp5d.log and check the Linux timezone and firmware DST rule.

12.2 Firmware Log (on Witty Pi 5 Mini)

The Witty Pi 5 Mini firmware also maintains a dedicated log file:

WittyPi5.log

This file is stored inside the **log** directory on the emulated USB flash drive. It contains the most complete and low-level debug information, which is identical to the messages you would observe via the USB serial interface.

With this family-wide dual-layer logging architecture, Witty Pi 5 Mini can provide diagnostic information even when the Raspberry Pi fails to start, making hardware and timing problems much easier to trace.

If logging to the on-board file is disabled, serial log output remains available while USB serial is active, but earlier messages are not retroactively written into the file after logging is re-enabled.

13. Frequently Asked Questions (FAQ)

13.1 What I2C address is used by Witty Pi 5 Mini? Can I change it?

Witty Pi 5 Mini communicates with the realtime clock and temperature sensor over its internal I2C bus, and exposes one I2C slave device to the Raspberry Pi at address 0x51. This address is configurable and can be changed if needed.

Register 0 reports Mini firmware ID 0x52, which is not the bus address. Changing the I²C slave address should be reserved for actual address conflicts: software compiled for 0x51 will not discover a board moved to another address automatically. Firmware validates address changes and applies accepted changes following an RP2350 restart.

If you have Witty Pi 5 Mini connected to Raspberry Pi and run the "i2cdetect -y 1" command in the console, you will see this:

```
pi@raspberrypi:~ $ i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- 51 -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

If you want to change the I2C address used by Witty Pi 5 Mini, you can change the value of I2C register #16. For example, to change the address to 0x35, you can run:

```
pi@raspberrypi:~ $ i2cset -y 1 0x51 16 0x35
```

Before restarting the RP2350 at the new I²C address, modify and recompile both wp5 and wp5d (which share wp5lib.h) so that both use the new address. Changing the address without updating the software can make the Mini inaccessible to the standard tools.

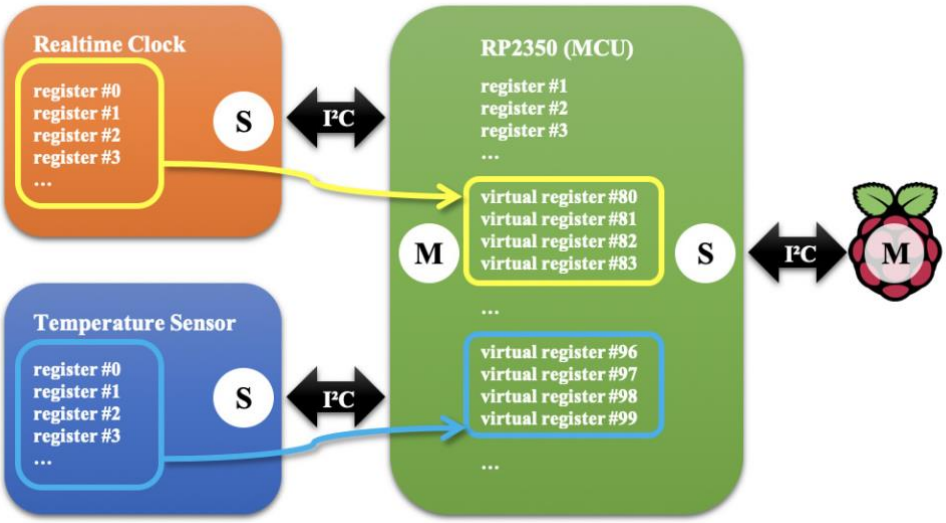
In the file "**wp5lib.h**", modify the value of **I2C_SLAVE_ADDR** from 0x51 to the value you want.

The final step is to shut down your Raspberry Pi, fully disconnect power from Witty Pi 5 Mini, and then reconnect it. The MCU will then start using the new address.

13.2 What I2C Registers are provided by Witty Pi 5 Mini?

The RP2350 microcontroller on Witty Pi 5 Mini works as an I2C slave, and the Raspberry Pi can read or write its registers through the I2C interface. The shared wp5 software configures Witty Pi 5 Mini by setting these registers.

The microcontroller also implements an I2C master to access the realtime clock and temperature sensor through an internal I2C bus. Their registers are mapped as virtual registers in the Witty Pi 5 Mini I2C slave device, so the Raspberry Pi can access them through the same external interface.



The realtime clock's registers are mapped to virtual registers #80~#95.

The temperature sensor's registers are mapped to virtual registers #96~#103.

From the Raspberry Pi's point of view, Witty Pi 5 Mini provides four kinds of I2C registers:

- Read-only I^2C registers
- Configuration I^2C registers
- Administrative I^2C registers
- Virtual I^2C registers

The table below shows the registers provided by Witty Pi 5 Mini. Some are read-only and can only be updated by the firmware itself:

Firmware v1.8 and later: R44–R53 configure automatic DST and track the DST-applied flag (R53); R56 controls 0–5 additional voltage confirmations (default 2). For ordinary use, configure DST with wp5 instead of writing registers individually, as the DST rule, RTC time and R53 must remain consistent.

Index	Description	Range	Default	Accessible
0	Firmware ID	--	0x52	Read-only
1	Firmware major version	1~99	--	Read-only
2	Firmware minor version	0~99	--	Read-only
3	The MSB of VUSB / common 5V rail (in mV)	0~255	0	Read-only
4	The LSB of VUSB / common 5V rail (in mV)	0~255	0	Read-only
5	The MSB of V_{IN} (in mV)	0~255	0	Read-only
6	The LSB of V_{IN} (in mV)	0~255	0	Read-only
7	The MSB of V_{OUT} (in mV)	0~255	0	Read-only
8	The LSB of V_{OUT} (in mV)	0~255	0	Read-only
9	The MSB of I_{OUT} (in mA)	0~255	0	Read-only
10	The LSB of I_{OUT} (in mA)	0~255	0	Read-only
11	Power mode: 0=via VUSB, 1=via VIN through external DC/DC, other=unknown	0 or 1	0	Read-only
12	Number of missed heartbeats	0~255	0	Read-only
13	Raspberry Pi state: 0=OFF, 1=STARTING, 2=ON, 3=STOPPING	0~3	0	Read-only
14	The latest action reason (higher 4 bits for startup and lower 4 bits for shutdown): 0=unknown, 1=startup alarm, 2=shutdown alarm 3=button click, 4=V_{IN} drops, 5=V_{IN} recovers, 6=over temperature, 7=below temperature, 8=newly powered, 9=reboot, 10=missed heartbeat, 11=external shutdown, 12=external reboot	--	0	Read-only
15	Miscellaneous state: bit0=schedule script in use	--	0	Read-only
16	I^2C slave address	--	0x51	Read & Write
17	The delay (in second) between power connection and turning on Pi: 0=immediately turn on, 255=stay off	0~255	255	Read & Write
18	The delay (in second) between Pi shutdown and power cut	0~255	15	Read & Write
19	Pulse interval in seconds, for LED and dummy load	0~255	10	Read & Write

20	How long the white LED should stay on (in ms), 0=no blink	0~255	100	Read & Write
21	How long the dummy load should be applied (in ms), 0=off.	0~255	0	Read & Write
22	Low voltage threshold (x10), 0=disabled	0~255	0	Read & Write
23	Recovery voltage threshold (x10), 0=disabled	0~255	0	Read & Write
24	Actual power source: 0=VUSB, 1=VIN via external DC/DC	0 or 1	0	Read &Write
25	Adjustment for measured V_{USB} (x100)	-128~127	0	Read & Write
26	Adjustment for measured V_{IN} (x100)	-128~127	0	Read & Write
27	Adjustment for measured V_{OUT} (x100)	-128~127	0	Read & Write
28	Adjustment for measured I_{OUT} (x1000)	-128~127	0	Read & Write
29	Allowed missed heartbeats before power cycle by watchdog, 0=disable	0~255	0	Read & Write
30	Whether to write log into file: 1=allowed, 0=not allowed	0 or 1	1	Read & Write
31	Whether to allow long press BOOTSEL and then click button for factory reset: 1=allowed, 0=not allowed	0 or 1	1	Read & Write
32	Second register for startup alarm (BCD format)	--	0	Read & Write
33	Minute register for startup alarm (BCD format)	--	0	Read & Write
34	Hour register for startup alarm (BCD format)	--	0	Read & Write
35	Day register for startup alarm (BCD format)	--	0	Read & Write
36	Second register for shutdown alarm (BCD format)	--	0	Read & Write
37	Minute register for shutdown alarm (BCD format)	--	0	Read & Write
38	Hour register for shutdown alarm (BCD format)	--	0	Read & Write
39	Day register for shutdown alarm (BCD format)	--	0	Read & Write
40	Action for below temperature: 0-do nothing; 1-startup; 2-shutdown	0~2	0	Read & Write
41	Below temperature threshold (signed degrees of Celsius)	-128~127	0	Read & Write
42	Action for over temperature: 0-do nothing; 1-startup; 2-shutdown	0~2	0	Read & Write

43	Over temperature threshold (signed degrees of Celsius)	-128~127	0	Read & Write
44	DST offset (bits 6–0, in minutes), 0=disabled; bit 7 selects mode: 0=recurring weekday, 1=fixed date (v1.8+)	0–127	0	Read & Write
45	DST begin month in BCD format	--	0	Read & Write
46	DST begin day: recurring mode upper nibble=week (0=last, 1–5=nth), lower nibble=weekday (0=Sun); fixed mode=calendar day (BCD)	--	0	Read & Write
47	DST begin hour in BCD format	--	0	Read & Write
48	DST begin minute in BCD format	--	0	Read & Write
49	DST end month in BCD format	--	0	Read & Write
50	DST end day: recurring mode upper nibble=week (0=last, 1–5=nth), lower nibble=weekday (0=Sun); fixed mode=calendar day (BCD)	--	0	Read & Write
51	DST end hour in BCD format	--	0	Read & Write
52	DST end minute in BCD format	--	0	Read & Write
53	DST applied: 0=standard time; 1=daylight saving time (v1.8+)	0 or 1	0	Read & Write
54	System clock (in MHz) for RP2350	--	48	Read & Write
55	VIN hot standby (not applicable to Mini; register retained for protocol compatibility)	0 or 1	0	Read & Write
56	Additional immediate VIN voltage confirmations before low-voltage shutdown or recovery (v1.8+)	0–5	2	Read & Write
57–63	Reserved for future usage	--	--	Read & Write
64	Register to specify directory	0~255	0	Admin
65	Register to provide extra context	0~255	0	Admin
66	Register to provide download stream	0~255	0	Admin
67	Register to provide upload stream	0~255	0	Admin
68	Password for administrative command	0~255	0	Admin
69	Administrative command to run	0~255	0	Admin
70	Heartbeat register for watchdog	0~255	0	Admin

71	Shutdown request: 0=none, 1=turn RPi off, 2=RPi is shutting down, 3=RPi is rebooting	0~3	0	Admin
72~79	Reserved for future usage	--	--	Admin
80	RX8025T: Second in RTC time	--	--	Virtual
81	RX8025T: Minute in RTC time	--	--	Virtual
82	RX8025T: Hour in RTC time	--	--	Virtual
83	RX8025T: Weekday in RTC time	--	--	Virtual
84	RX8025T: Date in RTC time	--	--	Virtual
85	RX8025T: Month in RTC time	--	--	Virtual
86	RX8025T: Year in RTC time	--	--	Virtual
87	RX8025T: RTC RAM register	--	--	Virtual
88	RX8025T: RTC minute alarm register	--	--	Virtual
89	RX8025T: RTC hour alarm register	--	--	Virtual
90	RX8025T: RTC day alarm register	--	--	Virtual
91	RX8025T: RTC timer Counter 0	--	--	Virtual
92	RX8025T: RTC timer Counter 1	--	--	Virtual
93	RX8025T: RTC extension register	--	--	Virtual
94	RX8025T: RTC flag register	--	--	Virtual
95	RX8025T: RTC control register	--	--	Virtual
96	TMP112: MSB of temperature	--	--	Virtual
97	TMP112: LSB of temperature	--	--	Virtual
98	TMP112: MSB of configuration	--	--	Virtual
99	TMP112: LSB of configuration	--	--	Virtual
100	TMP112: MSB of low-temperature threshold	--	--	Virtual

101	TMP112: LSB of low-temperature threshold	--	--	Virtual
102	TMP112: MSB of high-temperature threshold	--	--	Virtual
103	TMP112: LSB of high-temperature threshold	--	--	Virtual

Firmware-specific registers:

R44–R53 define the automatic DST rule and applied-state flag (firmware v1.8+);

R56 configures additional voltage confirmations (0–5, default 2, firmware v1.8+).

Values for months, days and transition times are in BCD where indicated.

Do not write individual DST registers while automatic DST is running unless you fully understand the relationship between the rule, R53, the RTC clock and scheduled actions. Use the wp5 menu for ordinary configuration.

Examples to access register

Below is an example to read the register with index 13, to know the current state of Raspberry Pi (0x02 means Raspberry Pi is ON):

```
pi@raspberrypi:~ $ i2cget -y 1 0x51 13
pi@raspberrypi:~ $ 0x02
```

And below is an example to write the register with index 19, to set the pulsing interval to 5 second:

```
pi@raspberrypi:~ $ i2cset -y 1 0x51 19 5
```

13.3 What GPIO Pins Are Used by Witty Pi 5 Mini?

The Raspberry Pi GPIO pins used by Witty Pi 5 Mini are marked in green in the table below.

GPIO (BCM)	Name	Physical		Name	GPIO (BCM)
	3.3V	1	2	5V	
GPIO 2	SDA 1	3	4	5V	
GPIO 3	SCL 1	5	6	GND	
GPIO 4		7	8	TXD	GPIO 14
	GND	9	10	RXD	GPIO 15
GPIO 17		11	12	PCM CLK	GPIO 18
GPIO 27		13	14	GND	
GPIO 22		15	16		GPIO 23
	3.3V	17	18		GPIO 24
GPIO 10	MOSI	19	20	GND	
GPIO 9	MISO	21	22		GPIO 25
GPIO 11	SCLK	23	24	CE0	GPIO 8
	GND	25	26	CE1	GPIO 7
GPIO 0	ID_SD	27	28	ID_SC	GPIO 1
GPIO 5		29	30	GND	
GPIO 6		31	32	PWM0	GPIO 12
GPIO 13	PWM1	33	34	GND	
GPIO 19	PCM FS	35	36		GPIO 16
GPIO 26		37	38	PCM DIN	GPIO 20
	GND	39	40	PCM DOUT	GPIO 21

Witty Pi 5 Mini uses the I2C buses and no other Raspberry Pi GPIO signals. This maximizes compatibility with other hardware.

GPIO 0/ID_SD and GPIO 1/ID_SC are not used by default. They can be used by the ID EEPROM, if you solder one on the back of your Witty Pi 5 Mini.

14. Revision History

Revision	Date	Description
1.00	2026.10.21	Initial revision for Witty Pi 5 Mini
1.01	2026.11.17	Document firmware v1.8 automatic DST and voltage confirmations, including Mini-specific VIN behavior. Update shared wp5 v5.4.0 menus, time synchronization, RTC checks and I²C register descriptions. Correct Mini I²C address versus product ID, clarify optional ID EEPROM and external DC/DC control, and revise the table of contents.