

# USB Serial/JTAG Controller Console

On chips with an integrated USB Serial/JTAG Controller, it is possible to use the part of this controller that implements a serial port (CDC) to implement the serial console, instead of using UART with an external USB-UART bridge chip. ESP32-S3 contains this controller, providing the following functions:

- Bidirectional serial console, which can be used with [IDF Monitor](#) or another serial monitor.
- Flashing using `esptool.py` and `idf.py flash`.
- JTAG debugging using e.g. OpenOCD, simultaneous with serial operations.

Note that, in contrast with the USB OTG peripheral in some Espressif chips, the USB Serial/JTAG Controller is a fixed function device, implemented entirely in hardware. This means it cannot be reconfigured to perform any function other than to provide a serial channel and JTAG debugging functionality.

## Hardware Requirements

Connect ESP32-S3 to the USB port as follows:

GPIO	USB
20	D+ (green)
19	D- (white)
GND	GND (black)
	+5V (red)


Some development boards may offer a USB connector for the USB Serial/JTAG Controller — in that case, no extra connections are required.

## Software Configuration

USB console feature can be enabled using `CONFIG_ESP_CONSOLE_USB_SERIAL_JTAG` option in menuconfig tool (see [CONFIG\\_ESP\\_CONSOLE\\_UART](#)).

Once the option is enabled, build the project as usual.

Alternatively, you can access the output through `usb_serial_jtag` port but make sure the option `CONFIG_ESP_CONSOLE_SECONDARY_USB_SERIAL_JTAG` in choice `ESP_CONSOLE_SECONDARY` is selected.

 Warning

Besides output, if you also want to input or use REPL with console, please select `CONFIG_ESP_CONSOLE_USB_SERIAL_JTAG`.

## Uploading the Application

The USB Serial/JTAG Controller is able to put the ESP32-S3 into download mode automatically. Simply flash as usual, but specify the USB Serial/JTAG Controller port on your system: `idf.py flash -p PORT` where `PORT` is the name of the proper port.

## Limitations

There are several limitations to the USB console feature. These may or may not be significant, depending on the type of application being developed, and the development workflow.

1. If the application accidentally reconfigures the USB peripheral pins, or disables the USB Serial/JTAG Controller, the device will disappear from the system. After fixing the issue in the application, you will need to manually put the ESP32-S3 into download mode by pulling low GPIO0 and resetting the chip.
2. If the application enters deep sleep mode, USB CDC device will disappear from the system.
3. The behavior between an actual USB-to-serial bridge chip and the USB Serial/JTAG Controller is slightly different if the ESP-IDF application does not listen for incoming bytes. An USB-to-serial bridge chip will just send the bytes to a (not listening) chip, while the USB Serial/JTAG Controller will block until the application reads the bytes. This can lead to a non-responsive looking terminal program.
4. If the application enters light-sleep (including automatic light-sleep) or software reset, etc. The USB CDC device will still work on the system. But be aware that this might increase the power consumption, if you don't need USB CDC in sleep and want to keep low power consumption, please disable the menuconfig `CONFIG_RTC_CLOCK_BBPLL_POWER_ON_WITH_USB`. Moreover, the power consumption will only increase when your USB CDC port is really in use (like data transaction), therefore, if your USB CDC just connects with power bank or battery, rather than something like computer, you don't need to care about the increasing power consumption mentioned above.