Grüezi YouTubers. Here is the guy with the Swiss accent. With a new episode around sensors and microcontrollers.

In video #147 I said, that the Arduino IDE can only use one Serial connection of the three available on the ESP32. A viewer told me, that this is wrong and pointed me to the right direction. And then, I even found an third, “secret” Serial connection.

Let’s start!

According the data sheet, the ESP32 chip has 3 “Universal Asynchronous Receiver/transmitters”, abbreviated UARTs, U0, U1 and U2. Each of them exposes 4 pins. RX, TX, RTS, and CTS. The Arduino IDE only uses RX and TX pins and we will later see, what CTS and RTS means. If we use Serial.begin(), U0 with pins 1 and 3 is used. These pins usually are also named RX/TX on our boards.

If we want to use one of the other two UARTs, we have to define them first, not as with the Arduino Mega, where they were already pre-defined. The command is: HardwareSerial Serial1(2) where Serial1 is the name and (2) is the UART number.

Here, we have a sketch which defines Serial1 on U1 and Serial2 on U2. Next, we issue a Serial.begin for all three UARTS and print test messages to all channels. Unfortunately, the ESP32 crashes after uploading the sketch. Not what I expected! What could be the reason?

Let’s quickly check the pins: UART1 pins are GPIO9 and 10 and UART2 pins are GPIO16 and 17. But on my Excel sheet from video #149, pins GPIO9 and 10 are marked red. Why? Because they are internally connected to the flash memory. You even do not find them on normal boards. So, we cannot use UART1, only UART2.

So, at least, we have a “official” second Serial connection.

But where is the secret one? The ESP32 has the capability to use nearly all GPIOs as pins for Serial connections. We just have to define it. But where? The definition is done by the IDE itself in the file “HardwareSerial.cpp”. Here, we see the definitions. If we change the definition of UART1 to GPIO4 and 2, and save the file, the ESP32 does not crash anymore. And if we connect the FTDI adapter to GPIO2, we can see the output. How cool is that? Of course, you could choose other pins. But, if you update the ESP32 files, this patch will disappear and your ESP will crash again.

And because we deal with Serial: Do you remember what CTS and RTS stands for? In the early days of computing, processors were not fast and also had not big buffers for serial communication. This is, why additional signals were used to signal the sender, that the receiver was busy and no more able to read any data. So, RTS means “request to send” and CTS “clear to send”. In our days, these signals are no more used, because the processors are very fast, and the serial drivers have buffers built-in to store incoming traffic, even if the processor is busy.

So, summarized,

* With the ESP32, we can use two Serial connections with the standard Arduino IDE
* And an additional with a small “patch”
* And we also know now, why we only need two wires for Serial communications, not four as in the old days…

I hope, this video was useful or at least interesting for you. If true, then like. Bye