Python Radio 7: The ESP32

Simon Quellen Field

Simon Quellen Field

Follow

3 min read

·

Aug 24, 2024

Listen

Share

More

More power!

Press enter or click to view image in full size

Image: Lilygo

The ESP32 is the big brother to the ESP8266.

It has two 32-bit processor cores on the chip, and a third less powerful processor for handling simple tasks while the more power-hungry cores sleep. It has over six times the RAM (520 KB) as the ESP8266 and runs at 240 MHz instead of 160 MHz. You can get one for about $12 at AliExpress.com.

Because it has several accessible hardware timers and UARTs (serial ports), it makes several radio projects possible that would be difficult to do with the smaller chip.

ASCII text by radio

The ESP32 has several UARTs (Universal Asynchronous Receiver-Transmitter) for sending serial data. This means we can use one for the USB connection to a laptop or desktop computer and still have fully functional serial data capabilities for sending bits to the radio. The ESP8266 only had the transmitting half of the serial port free for us to use.

We will use two ESP32 boards. The board we will call Ichabod will transmit messages to the board we will call Rumpelstiltskin.

We set up the UART to use pin 21 as output and pin 37 as input. If the machines are close together, a baud rate as high as 9600 will work, but for longer distances, a baud rate of 2400 works quite well, communicating without errors using our 433.92 MHz FS1000A at a distance of over 1,000 feet (probably much farther, but I ran out of room).

The program is very simple, since the hardware is doing all the work. We build a message and call uart.write() to send it, or we call uart.read() to read a byte stream from the UART, after which we convert it to a string and print it. Note that while uart.write() will accept a string, uart.read() returns a stream of bytes that have to be converted to utf-8.

From machine import UART

From name import name

TX = 21

RX = 37

Def main():

Uart = UART(1, baudrate=2400, tx=TX, rx=RX)

If name == “Ichabod”:

Count = 0

While True:

Msg = “UUU “ + str(count) + “: Hello, world!\r\n”

Print(”Sending”, msg, end=”)

Uart.write(msg)

Count += 1

While True:

Try:

M = uart.read()

If m:

Print(m.decode(’utf-8’), end=”)

Except UnicodeError:

Pass

Main()

It is convenient to have one simple main.py program for both boards, so I made a file called name.py and put the following line in it:

Name = “Rumpelstiltskin”

Before loading it onto one of the boards. Then I changed the line to

Name = “Ichabod”

And loaded it onto the other board. Now, as I make changes to main.py (such as changing the baud rate), I can just load the same file onto both boards.

One of my favourite ESP32 modules is the TTGO TDisplay, which has a built-in OLED display. The code above and the diagrams below assume you are using that board, although it should be trivial to change the code for a different module.

The circuit is simple:

Press enter or click to view image in full size

Image by author

The receiver circuit looks like this:

Press enter or click to view image in full size

Image by author

In later projects, we will be using the power of the ESP32 to do a lot more work. Some radio modes require a good deal of computing power.

It is also nice to have a display to print things to.

Esp32

Micropython

Python Programming

Radio Communications

Texting