You
Can you help me correct this code, I'm working with a SEEED XIAO rp2040 and a TFT ili9341 display. I have an error that says "bad sck pin" but I have the sck connected to the gpio 2 in micropython. Here is the code, help me fix it!

ChatGPT
Of course! To help you effectively, I'll need the specific part of your code where you're setting up or configuring the connection to the TFT ili9341 display using MicroPython. The "bad sck pin" error typically indicates an issue with how the serial clock (SCK) pin is defined or used in your code, potentially due to an incorrect pin number or misconfiguration.
In the meantime, here's a general approach to configuring an SPI connection with an ili9341 display on a SEEED XIAO RP2040 using MicroPython, which might help you identify the issue:

- Identify the correct pins: Ensure you're using the correct pins for SPI communication. For the SEEED XIAO RP2040, the default SPI pins are usually defined in the documentation. If GPIO 2 is indeed the SCK pin according to the documentation or your custom setup, make sure it's correctly defined in your code.
- Initialize the SPI bus: You typically need to initialize the SPI bus with the correct pins. Here's a basic example:

```
from machine import Pin, SPI
import ili9341

# Initialize SPI
# Replace 'sck=Pin(2)', 'mosi=Pin(3)', and 'miso=Pin(4)' with the actual pins if
they're different
spi = SPI(0, baudrate=40000000, sck=Pin(2), mosi=Pin(3), miso=Pin(4))

# Initialize display
# Replace 'cs', 'dc', and 'rst' with the actual pins
display = ili9341.ILI9341(spi, cs=Pin(5), dc=Pin(6), rst=Pin(7))
```

This example assumes you're connecting the display's SCK, MOSI, and MISO to GPIO 2, 3, and 4, respectively. Adjust the SPI(0,...) if you're using a different SPI bus.

- Check for pin conflicts: Ensure no other part of your code or another device is using the same pin, leading to conflicts.
- Review documentation: Double-check the SEEED XIAO RP2040 and the ili9341 display documentation to ensure you're using the pins correctly.
- Hardware check: Ensure the physical connections are secure and there are no faulty wires or solder joints.

If you can share the specific part of your code related to the SPI setup or the full code if it's not too long, I can provide more targeted advice.

You

```
from time import sleep
from ili9341 import Display, color565
from machine import Pin, SPI


def test():
    """Test code."""
    # Baud rate of 40000000 seems about the max
    spi = SPI(1, baudrate=10000000, sck=Pin(2), mosi=Pin(3))
    display = Display(spi, dc=Pin(0), cs=Pin(1), rst=Pin(7))

    # Set display color to red
    display.clear(color565(255, 0, 0))
    sleep(1)

    # Set display color to orange
    display.clear(color565(255, 128, 0))
    sleep(1)

    # Set display color to yellow
    display.clear(color565(255, 255, 0))
    sleep(1)

    # Set display color to green
    display.clear(color565(0, 255, 0))
    sleep(1)

    # Set display color to blue
    display.clear(color565(0, 0, 255))
    sleep(1)

    # Set display color to purple
    display.clear(color565(128, 0, 128))
    sleep(1)

    # Set display color to pink
    display.clear(color565(255, 192, 203))
    sleep(1)

    # Set display color to brown
    display.clear(color565(139, 69, 19))
    sleep(1)

    # Set display color to gray
    display.clear(color565(128, 128, 128))
    sleep(1)

    # Set display color to white
    display.clear(color565(255, 255, 255))
    sleep(1)

test()
```

ChatGPT

Looking at your code, the primary issue seems related to the selection of SPI and GPIO pins for interfacing with the ILI9341 display on a SEEED XIAO RP2040. The error "bad sck pin" typically means there's a problem with the pin assignment for the SPI clock.

For the SEEED XIAO RP2040, it's crucial to ensure you're using the correct pins that are designated for SPI communication. Here are some points to consider:

- SPI Bus Selection: You're using `SPI(1)` in your code. Ensure that this SPI bus is correctly mapped to the GPIO pins you intend to use for `SCK` and `MOSI`.
- SCK Pin: You've designated `Pin(2)` as the `SCK` pin. It's essential to verify that this pin can function as an SPI clock for the SPI bus you've chosen (`SPI(1)`). The RP2040 has flexible pin mapping for SPI, but not every pin can serve as an `SCK` pin for any SPI peripheral.
- Correct Pin Mapping: Ensure the pin mapping is correct for your SPI configuration and matches the RP2040's capabilities. Sometimes, documentation or online forums specific to the SEEED XIAO RP2040 can offer insights into which pins are best suited for SPI communication.

Suggested Steps

- Verify SPI Bus and Pin Capabilities: Refer to the SEEED XIAO RP2040 documentation to ensure that `SPI(1)` is the correct SPI peripheral to use and that `Pin(2)` supports serving as an `SCK` for this SPI bus.
- Hardware Connections: Double-check your physical connections to ensure they match your code's configuration.
- Example Adjustment: If `Pin(2)` cannot serve as `SCK` for `SPI(1)`, you will need to choose a pin that is compatible. The choice of pins might be constrained by the hardware capabilities of the SEEED XIAO RP2040.

```
from time import sleep
from ili9341 import Display, color565
from machine import Pin, SPI
```

```
def test():
 # Adjusted baud rate and pin assignments as necessary
 spi = SPI(1, baudrate=10000000, sck=Pin("correct_sck_pin"),
mosi=Pin("correct_mosi_pin"))
 display = Display(spi, dc=Pin("correct_dc_pin"), cs=Pin("correct_cs_pin"),
rst=Pin("correct_rst_pin"))

 # Your display test code follows...
```

Replace "correct_sck_pin", "correct_mosi_pin", "correct_dc_pin", "correct_cs_pin", and "correct_rst_pin" with the correct pin numbers according to your hardware setup and the SEEED XIAO RP2040 documentation.

Conclusion

If you're unsure about the correct pins to use, consult the SEEED XIAO RP2040's documentation or datasheet. It's also beneficial to look at examples or libraries specifically designed for the SEEED XIAO RP2040 with the ILI9341 display, as they might provide a proven configuration.