

Pupper Motors use Pins 3-16

Need to activate alternate uart: enter in /boot/firmware/usercfg.txt

enable\_uart=1

dtoverlay=uart5

Alternate UART on Pi 4B:

|  |  |  |  |
| --- | --- | --- | --- |
| Dtoverlay | Tx pin (SDA) | Rx pin (SCL) |  |
| UART1 |  |  |  |
| UART2 |  |  |  |
| UART3 |  |  |  |
| UART4 |  |  |  |
| UART5 | 32/GPIO12 | 33/GPIO13 |  |

To test, shorted pin 32 and 33

To monitor: minicom -D /dev/ttyAMA1 -b 9600

To send: minicom -D /dev/serial0 -b9 600

/dev/serial0 – primary UART port

/dev/serial1 – seconday UART port

/dev/ttyS0 – miniUART

/dev/ttyAMA0 – first PL011 (UART0)

Base configuration have /dev/serial1 -> /dev/ttyAMA0 (so Bluetooth is going to first PL011)

I want /dev/serial0 -> /dev/ttyS0 (I want uart5 to be the primary UART and set on the miniUART)

enable\_uart=1 : activates the miniUART and sets core\_freq to 250MHz

deleted `console=serial0, 115200` from /boot/firmware/cmdline.txt

trying adafruit simpletest with device /dev/ttyS0 and rst=19 (connected rst to GPIO19)

having to sudo pip install Adafruit-BNO055

Adafruit\_GPIO does not support Rpi4, had to add elif to Adafruit\_GPIO/Platform.py in detect\_version() for BCM2711

Trying with I2c

https://github.com/adafruit/Adafruit\_CircuitPython\_BNO055

In usercfg.txt set dt\_param=i2c\_arm\_baudrate=10000

Source activate imu

Pip install adafruit-circuitpython-bno055

Permission denied trying root install sudo pip3

Ok SO, have to use UART, set dtoverlay=uart5 to make Pin 32 TX and pin 33

To check if configuration worked use

raspi-gpio get 32

raspi-gpio get 33

These should say the functions are TXD0 and RXD0 (which means connected to /dev/ttyAMA0)

Install circuitpython packages, sudo pip3 install adafruit-circuitpython-bno055

\*\* Oh this might not be accurate, still says TXD0 and RXD0 and uart5 is not set

In ipython:

Import serial

Import adafruit\_bno055

uart = serial.Serial(‘/dev/ttyAMA0’, baudrate=9600)

sensor = adafruit\_bno055.BNO055\_UART(uart)

UART pin layouts

TXD RXD CTS RTS Board Pins

uart0 14 15 8 10

uart1 14 15 8 10

uart2 0 1 2 3 27 28 (I2C)

uart3 4 5 6 7 7 29

uart4 8 9 10 11 24 23 (SPI0)

uart5 12 13 14 15 32 33 (gpio-fan)

with uart5 enabled it should be set to AMA1

setting dtoverlay=uart5 does create /dev/ttyAMA1, but why can’t I access it

default

lrwxrwxrwx 1 root root 7 Oct 14 21:06 /dev/serial1 -> ttyAMA0

enable\_uart=1

lrwxrwxrwx 1 root root 5 Oct 14 20:51 /dev/serial0 -> ttyS0

lrwxrwxrwx 1 root root 7 Oct 14 20:51 /dev/serial1 -> ttyAMA0

enable\_uart=1

dtoverlay=miniuart-bt

lrwxrwxrwx 1 root root 7 Oct 14 21:07 /dev/serial0 -> ttyAMA0

lrwxrwxrwx 1 root root 5 Oct 14 21:07 /dev/serial1 -> ttyS0

So Bluetooth uses serial1

Bluetooth: serial1

miniUART: ttyS0

So with: enable\_uart=1 and dtoverlay=uart5, Pins32/33 are setup as ttyAMA1 and I get a OSError: UART access error

Best tutorial: <https://learn.adafruit.com/adafruit-bno055-absolute-orientation-sensor/python-circuitpython>

\*\*\* I got it working using I2C, I was using the wrong I2C pins earlier. Default I2C is pins 3 and 5

Apparently, only this I2C has clock stretching issues. On RPi4 I can activate another I2C

May be able to setup other GPIO as I2C by connecting pull-up resistors between the SDA/SCL and 3.3Vin and then using i2c-gpio dtoverlay

Following intructions board.SCL = 3 and board.SDA = 2

So I should be able to setup I2C on other pins

Also RPi4 should have internal pull-ups I can set on those pins

Setting in config:

dtparam=i2c1=off

dtparam=i2c0=on

setting i2c\_arm\_baudrate to 10000

these settings seem to also have activated i2c 10 and 11

checking i2c connectivity with i2cdetect -y 0

This worked

Using i2c = busion.I2C(1,0)

Creating test data

30 sec still

30 sec holding stillish in air

-90 to +90 yaw

+45 to – 45 pitch

-45 to +45 roll

REPEAT yaw, pitch , roll

SLOW

5” forward

5” back

3.5” left

3.5” right

FAST

Same moves

Lift in air, move slow then fast

Set down

Sensor started reporting None values at some point

Fixed by reinitializing sensor

120 seconds (ish)

5795 readings (expected rate is 60Hz, implemented with time.sleep(1/60))

270 dropped euler angles (4.7%)

203 dropped acceleration values (3.5%)

13 both were missing