

Mraa Usage

MRAA Introduction

Eclipse Mraa (Libmraa) is a C/C++ library with Java, Python, and JavaScript bindings for connecting I/O pins and buses on a variety of IoT and edge platforms, with a well-structured API and port names/numbers that match the boards on which they are located.

Install MRAA

```
sudo apt-get update -y
sudo apt-get install libmraa2 libmraa-dev libmraa-java python3-mraa mraa-tools -y
```

MRAA Command Line Tools

GPIO

- mraa-gpio list: List all available pins
- mraa-gpio get pin: Get Pin Status
- mraa-gpio set pin level: Set Pin Status
- mraa-gpio version: Get MRAA Version

I2C

- mraa-i2c list: List available I2C buses
- mraa-i2c version: Get mraa version and board name
- mraa-i2c detect bus: List detected devices on specified bus
- mraa-i2c get bus device reg: Get value from specified device register
- mraa-i2c set bus device reg value: Set specified device register to value

UART

- mraa-uart list: Lists UARTs on the system
- mraa-uart dev dev_num baud customized_baud send str: send str to port dev_num
- mraa-uart dev dev_num baud customized_baud recv 1000: receive 1000 bytes from port dev_num

```
~
> ssh radxa@192.168.2.234
radxa@192.168.2.234's password:
Linux rock-4c-plus 5.10.110-20-rockchip #e0ac49d1b SMP Tue Sep 26 08:20:36 UTC 2023 aarch64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Aug 9 08:38:30 2024 from 192.168.2.20
radxa@rock-4c-plus:~$ sudo mraa-uart list
[sudo] password for radxa:
0. --- error reading uart information ---
1. /dev/ttyS4   undefined    9600 8N1 (none) XONXOFF
radxa@rock-4c-plus:~$ sudo mraa-uart send "this"
Usage: mraa-uart { list | dev device } [ baud bps ] [ databits d ] [ parity p ] [ stopbits s ] [
ctsrts mode ] [ send string ] [ rcv timeout ] [ show ]

Simple tool to test UART functionality. Needs either list or dev arguments, the others are option
al
list      : lists uarts on the system (non intrusive)
dev       : select uart device, can be by name, by device name or by index (as listed in list)
baud      : set the baudrate to given parameter, in bits per second
parity    : set parity mode to given parameter - can be E, O, or N
stopbits  : set the number of stopbits - can be 1 or 2
ctsrts    : set CTS/RTS flow control to either on or off
send      : transmits a string
recv      : reads data on uart for timeout seconds, and displays the result on stdout
show      : show settings of selected uart
radxa@rock-4c-plus:~$ sudo mraa-uart send "this"

~
> picocom
picocom v3.1

port is       : /dev/ttyUSB0
flowcontrol   : none
baudrate is   : 1500000
parity is     : none
databits are  : 8
stopbits are  : 1
escape is     : C-a
local echo is : no
noinit is    : no
noreset is   : no
hangup is    : no
nolog is     : no
send_cmd is  : sz -vv
receive_cmd is : rz -vv -E
imap is      :
op is       :
up is       : crcrlf,delbs,
logfile is   : none
initstring   : none
exit_after is : not set
exit is      : no

Type [C-a] [C-h] to see available commands
Terminal ready
```

mraa-uart usage example

Sample Code

GPIO

C Python

▶ blink.c

Usage:

- Connect the led signal pin to PIN3, VCC to the board VCC, GND to the board GND
- Test

```
gcc blink.c -lmraa -o blink && sudo . /blink
```

I2C

C Python

▶ led_i2c_blink.c

Usage:

- Turn on Overlay of I2C2 and reboot.
- Wiring: SDA <--> SDA, SCL <--> SCL, GDN <--> GND, VCC <--> VCC
- Open the terminal and enter the following command to test

```
gcc led_i2c_blink.c -lmraa -o led_i2c_blink && sudo ./led_i2c_blink
```

If the test is successful, the I2C Led will have a running light effect.

PWM

C Python

▶ led_pwm_fade_out.c

Usage:

- Turn on the PWM Overlay of a certain PIN and restart, take the PWM of PIN28 as an example.
- Connect the signal pin of the led to the PIN28 pin, VCC to the VCC of the board, and GDN to the GND of the board.
- Test

```
gcc led_pwm_fade_out.c -lmraa -o led_pwm_fade_out && sudo ./led_pwm_fade_out 28
```

SPI

C Python

▶ spi_test.c

Usage:

- Turn on spi0 i.e. spidev0.0's Overlay and reboot!
- Test

Open a terminal and enter the following commands to test

```
gcc spi_test.c -o spi_test && sudo ./spi_test
```

If the test is successful, the terminal will output "0xaa".

UART

C Python

▶ snd.c

▶ recv.c

Usage:

- Open the overlay of two UARTs, take uart4 and uart6 as an example, i.e. /dev/ttyS4 and /dev/ttyS6.

- Cross-wire the TX and RX pins of the two UARTs.
- Test

Open two terminals and enter the following commands to test respectively.

```
gcc recv.c -lmraa -o recv && sudo ./recv 6 # /dev/ttyS6
```


```
gcc snd.c -lmraa -o snd && sudo ./snd 4 "hello, this is test" # /dev/ttyS4
```

If the test is successful, uart6 receives the cyclic message "hello, this is test" from uart4.

More

- [Mraa API reference](#)
- [GitHub](#)

 [Report Issue](#)

 [Edit this page](#)