Proposition OJ

- Exposé des avancées hardware (mb_2024_0_2 & mux_2024_0_X), Q&A

- Exposé des avancées software (dernières modifs Rémi, code_refactor), Q&A

- Définition des objectifs pour le hackathon

- Définition des objectifs pour le release hardware et software v2024
  → documentation carte; merge code_refactor + doc API; métrologie carte

- Revue des spécifications des composants (TX, RX, …) des versions 2023 et 2024

- Perspectives de publication sur la version 2024

- Brainstorming sur les objectifs pour le release v2025
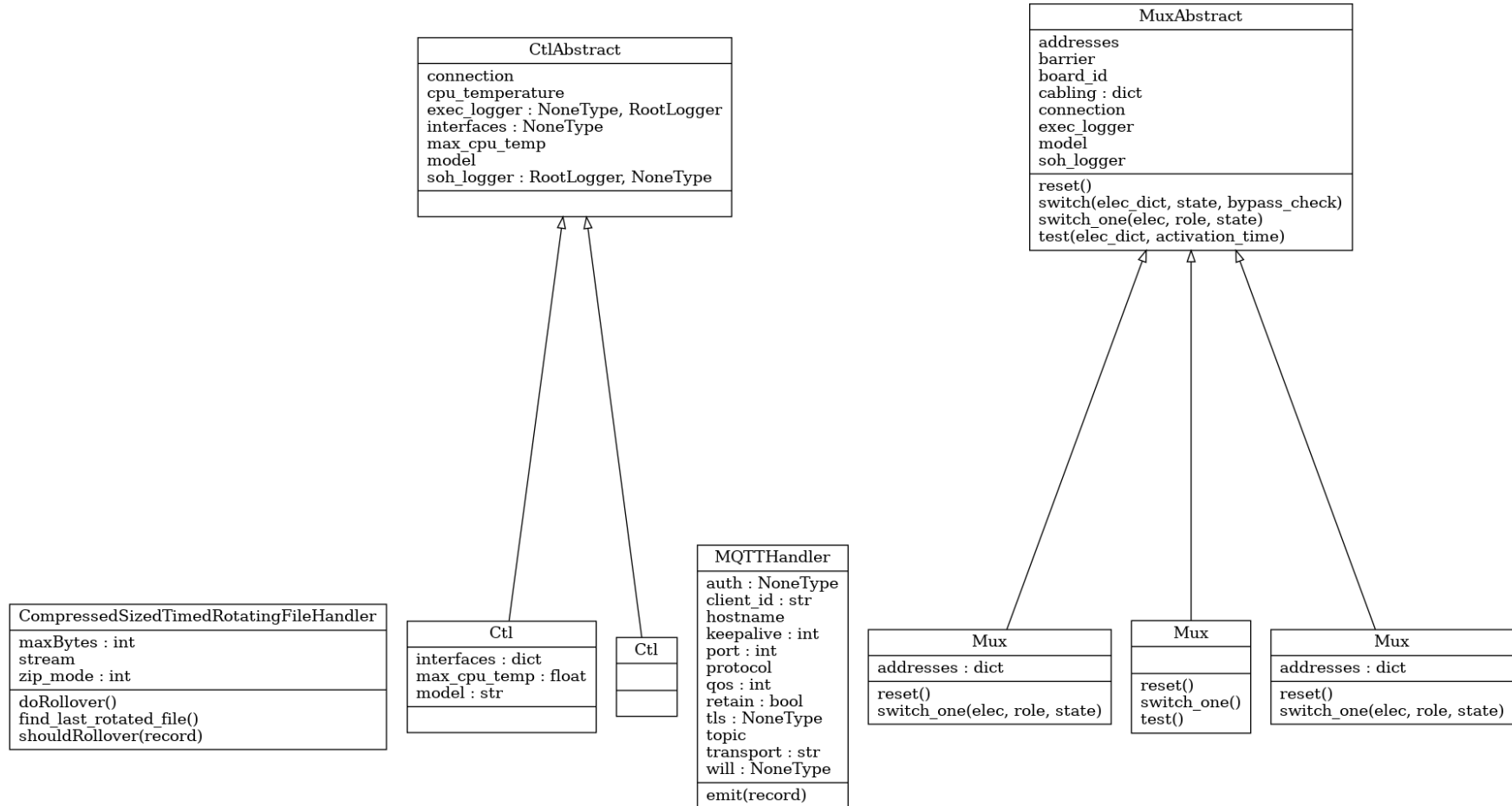 (IP?; multi-channel?...)

Code Refactor

Code Refactor

OhmPi

cmd_id : NoneType
controller : NoneType
data_logger : RootLogger, NoneType
exec_logger : RootLogger, NoneType
id : str
mqtt : bool
nb_samples : int
on_pi : bool, NoneType
sequence
sequence : ndarray, NoneType
settings : dict
soh_logger : RootLogger, NoneType
status : str
thread : Thread, NoneType

---

append_and_save(filename, last_measurement, cmd_id)
get_data(survey_names, cmd_id)
get_deprecated_methods(cls)
interrupt(cmd_id)
load_sequence(filename, cmd_id)
quit(cmd_id)
remove_data(cmd_id)
reset_mux(cmd_id)
restart(cmd_id)
rs_check(tx_volt, cmd_id)
run_measurement(quad, nb_stack, injection_duration, duty_cycle, autogain, strategy, tx_volt, best_tx_injtime, cmd_id)
run_multiple_sequences(cmd_id, sequence_delay, nb_meas)
run_sequence(cmd_id)
run_sequence_async(cmd_id)
set_sequence(sequence, cmd_id)
switch_mux_off(quadrupole, cmd_id)
switch_mux_on(quadrupole, bypass_check, cmd_id)
test_mux(activation_time, mux_id, cmd_id)
update_settings(settings, cmd_id)

hw

OhmPiHardware
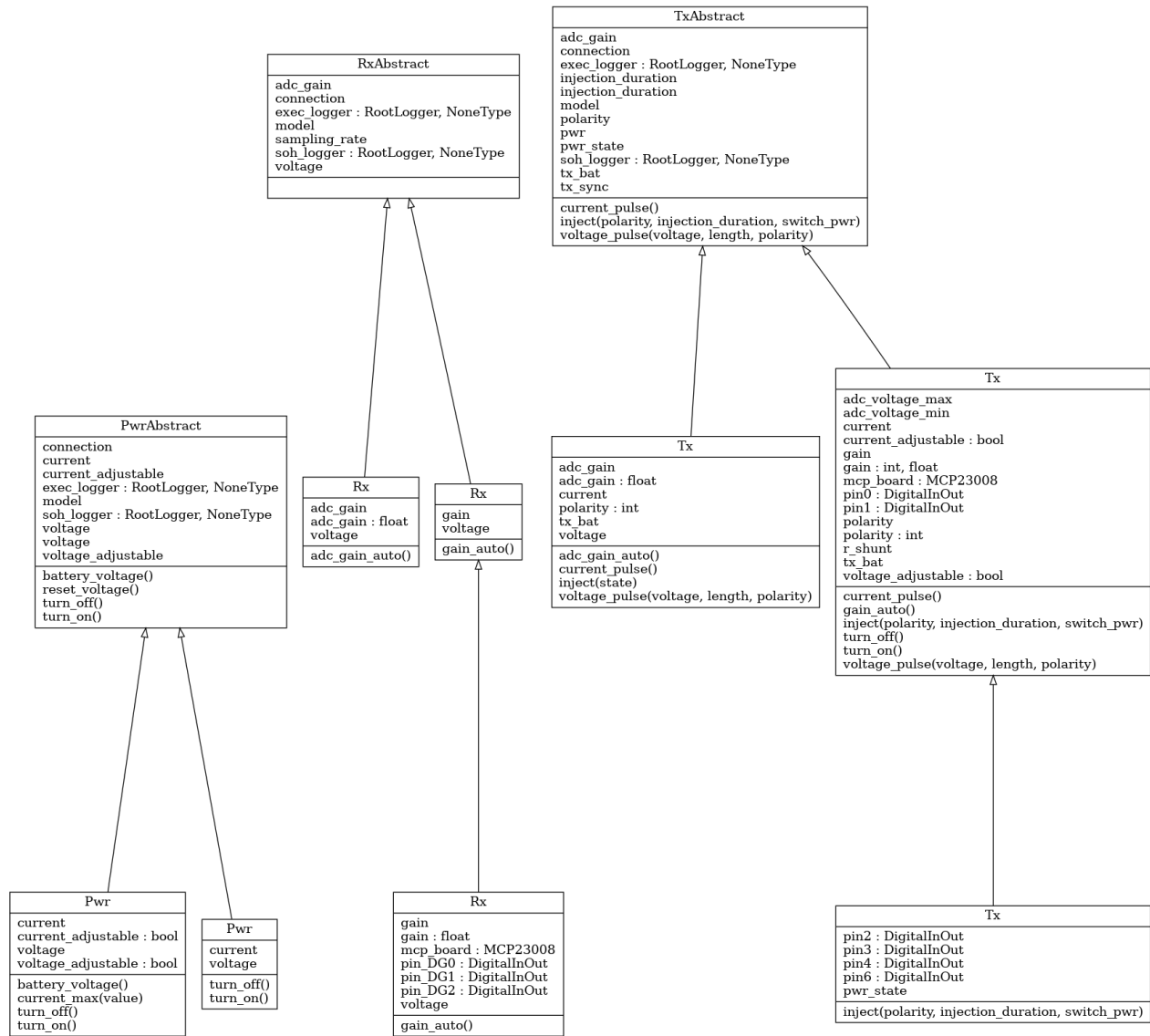
ctl
data_logger
exec_logger
mux_barrier : Barrier
mux_boards : dict
pulses
pwr
readings : ndarray
rx
soh_logger
sp
tx
tx_sync : Event

---

calibrate_rx_bias()
last_dev(delay)
last_resistance(delay)
reset_mux()
switch_mux(electrodes, roles, state)
test_mux(channel, activation_time)
vab_square_wave(vab, cycle_duration, sampling_rate, cycles, polarity, duty_cycle, append)

MyServer

do_POST()

Code Refactor

CtlAbstract

connection
cpu_temperature
exec_logger : NoneType, RootLogger
interfaces : NoneType
max_cpu_temp
model
soh_logger : RootLogger, NoneType

MuxAbstract

addresses
barrier
board_id
cabling : dict
connection
exec_logger
model
soh_logger

reset()
switch(elec_dict, state, bypass_check)
switch_one(elec, role, state)
test(elec_dict, activation_time)

CompressedSizedTimedRotatingFileHandler

maxBytes : int
stream
zip_mode : int

doRollover()
find_last_rotated_file()
shouldRollover(record)

Ctl

interfaces : dict
max_cpu_temp : float
model : str

Ctl

MQTTHandler

auth : NoneType
client_id : str
hostname
keepalive : int
port : int
protocol
qos : int
retain : bool
tls : NoneType
topic
transport : str
will : NoneType

emit(record)

Mux

addresses : dict

reset()
switch_one(elec, role, state)

Mux

reset()
switch_one()
test()

Mux

addresses : dict

reset()
switch_one(elec, role, state)

RxAbstract

adc_gain
connection
exec_logger : RootLogger, NoneType
model
sampling_rate
soh_logger : RootLogger, NoneType
voltage

TxAbstract

adc_gain
connection
exec_logger : RootLogger, NoneType
injection_duration
injection_duration
model
polarity
pwr
pwr_state
soh_logger : RootLogger, NoneType
tx_bat
tx_sync

current_pulse()
inject(polarity, injection_duration, switch_pwr)
voltage_pulse(voltage, length, polarity)

PwrAbstract

connection
current
current_adjustable
exec_logger : RootLogger, NoneType
model
soh_logger : RootLogger, NoneType
voltage
voltage
voltage_adjustable

battery_voltage()
reset_voltage()
turn_off()
turn_on()

Rx

adc_gain
adc_gain : float
voltage

adc_gain_auto()

Rx

gain
voltage

gain_auto()

Tx

adc_gain
adc_gain : float
current
polarity : int
tx_bat
voltage

adc_gain_auto()
current_pulse()
inject(state)
voltage_pulse(voltage, length, polarity)

Tx

adc_voltage_max
adc_voltage_min
current
current_adjustable : bool
gain
gain : int, float
mcp_board : MCP23008
pin0 : DigitalInOut
pin1 : DigitalInOut
polarity
polarity : int
r_shunt
tx_bat
voltage_adjustable : bool

current_pulse()
gain_auto()
inject(polarity, injection_duration, switch_pwr)
turn_off()
turn_on()
voltage_pulse(voltage, length, polarity)

Pwr

current
current_adjustable : bool
voltage
voltage_adjustable : bool

battery_voltage()
current_max(value)
turn_off()
turn_on()

Pwr

current
voltage

turn_off()
turn_on()

Rx

gain
gain : float
mcp_board : MCP23008
pin_DG0 : DigitalInOut
pin_DG1 : DigitalInOut
pin_DG2 : DigitalInOut
voltage

gain_auto()

Tx

pin2 : DigitalInOut
pin3 : DigitalInOut
pin4 : DigitalInOut
pin6 : DigitalInOut
pwr_state

inject(polarity, injection_duration, switch_pwr)
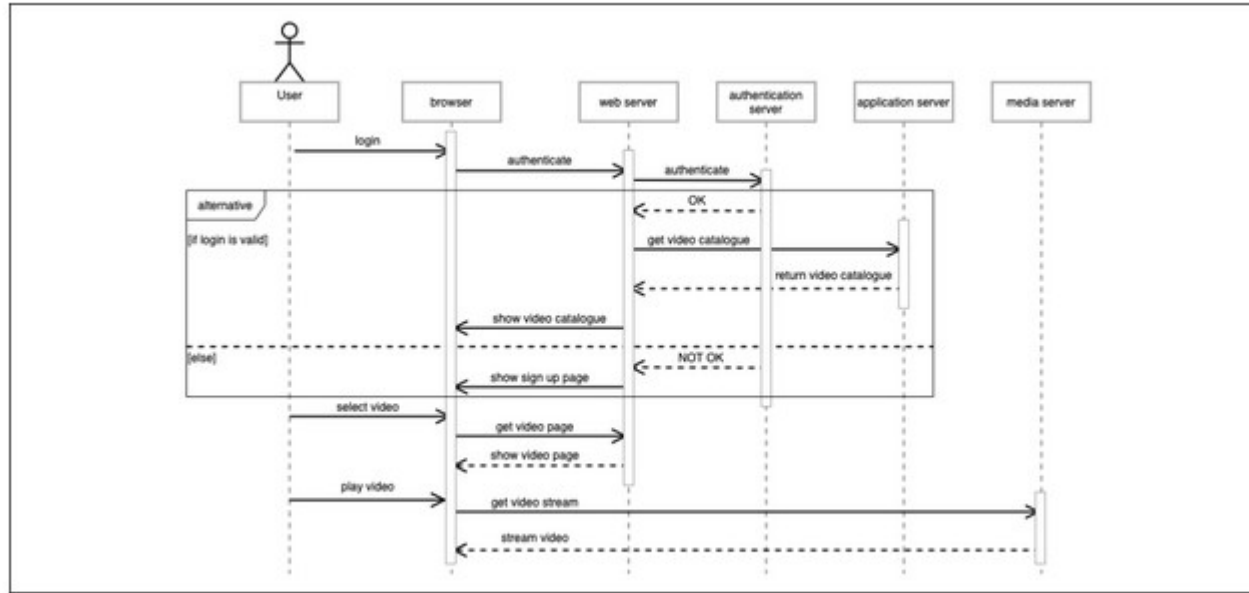
self._hw: Sequence Diagram?

Figure 7: UML sequence diagram; click to enlarge (Bob Reselman, CC BY-SA 4.0)

TODO code_refactor

- Tester les ajouts pwr switch on/off
- Tester mb_2023 avec code_refactor
- Tester MUX 2023 avec code_refactor
- Implémenter les stratégiques VMIN, VMAX
- Documenter
- Evaluer/implémenter les nouvelles approches de Rémi pour équilibrer les pulses positifs et négatifs

Documentation

- Docstrings
- Docs utilisateurs
- Docs développeurs software
- Docs développeurs hardware?

Review pwr_switching

- test switch_on / switch_off
- add 'hot_switchable' property?
- set tx.pwr_voltage in tx (can use a circuit either on pwr or on tx board)

Using more than one controller?

- To have a more robust handling of time in some operations

- To operate a part of the system remotely

component
MUX

component
Rx

controller
ctl_RPi

controller
ctl_mc
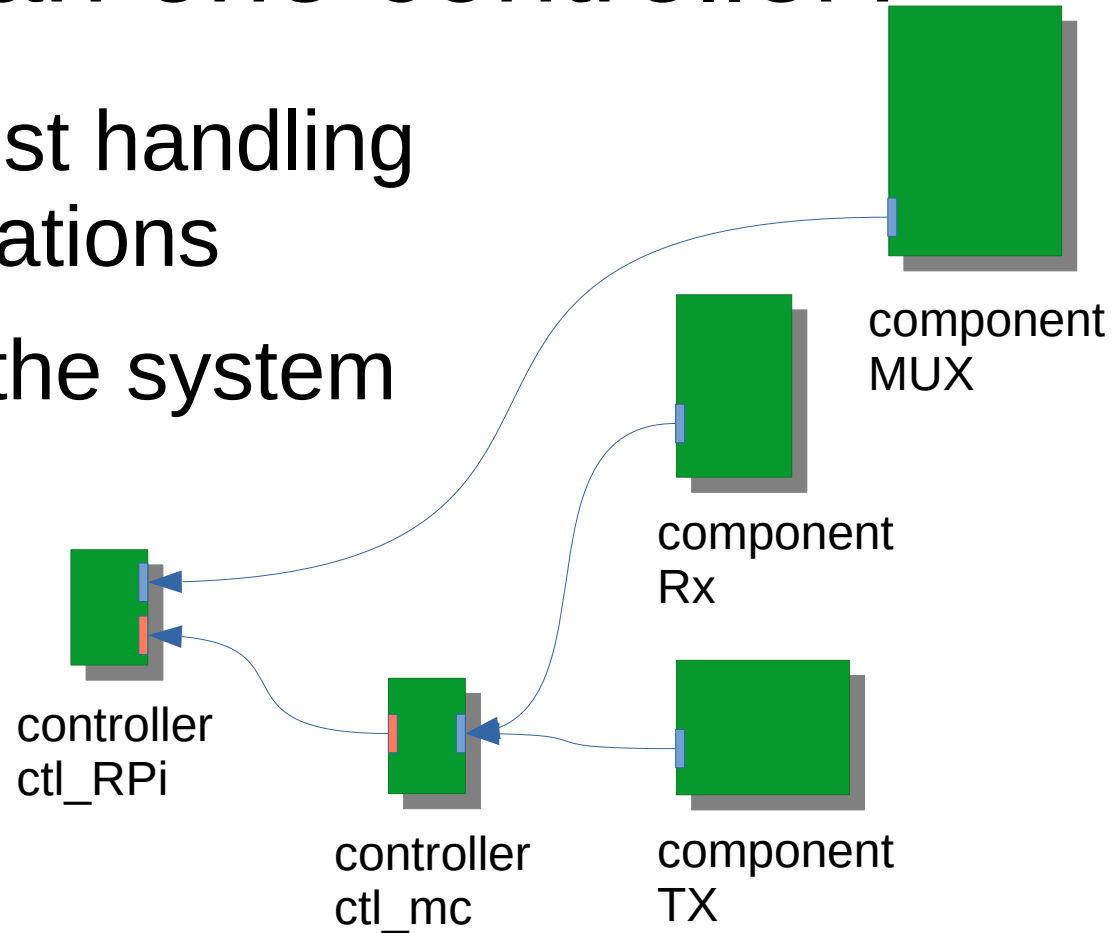
component
TX

Controllers

- ## Have several interfaces
  (I2C, modbus…)

- ## May have one connection
  (pointing towards a
  controller such as RPi)

Tree : each child points to its parent
Component.connection → ctl_mc.interfaces['i2c']
ctl_mc.connection → ctl_RPi.interfaces['wifi']

For ctl_RPi set the connection
to MQTT broker as connection?

component
MUX

component
Rx

component
TX

controller
ctl_RPi

controller
ctl_mc
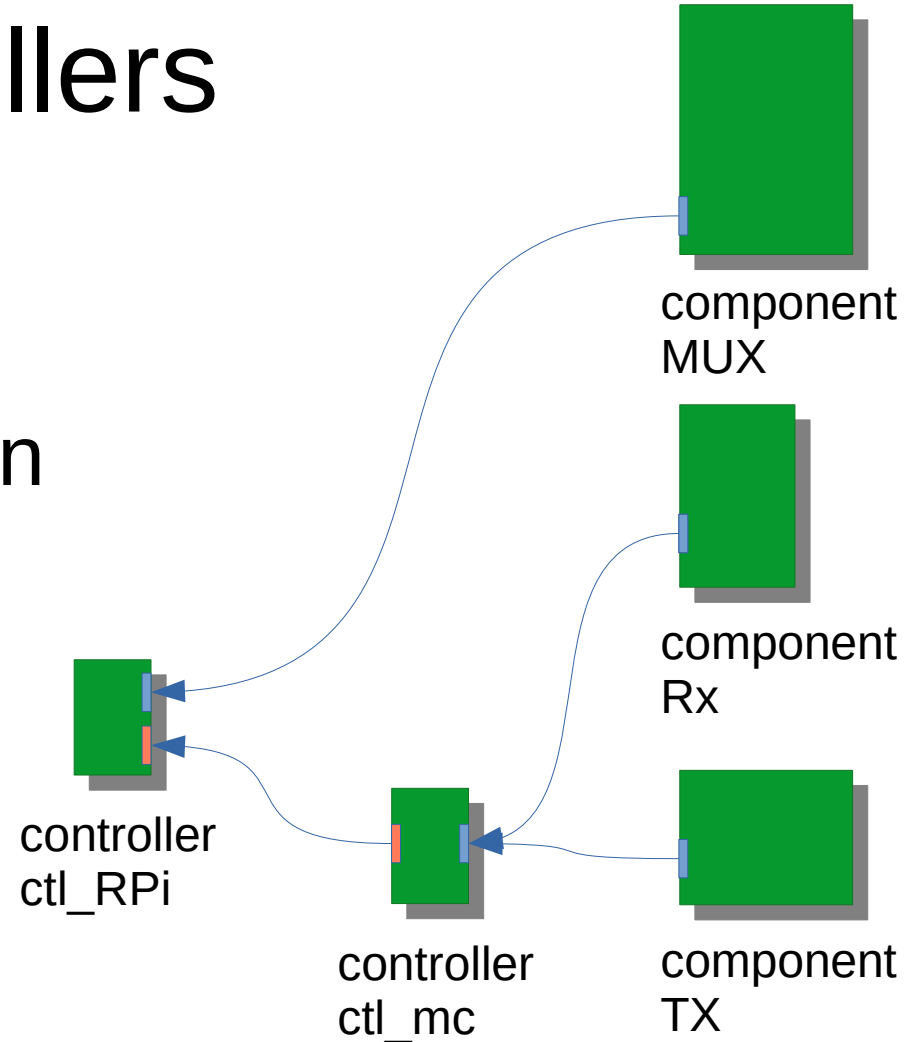
Config

- Used to define a hardware system
  (not characteristics of an isolated component)

- Should allow for multiple boards for each kind
  (MUX done, RX, TX, CTL… TODO)

- Specs (and default values) are given in components classes

- Should allow for defaults by kind of component

- Should allow for distinct values for each component

rx._bias

?

Sp - basic strategy

VMN

sp

t

m_p : Mean of positive pulses
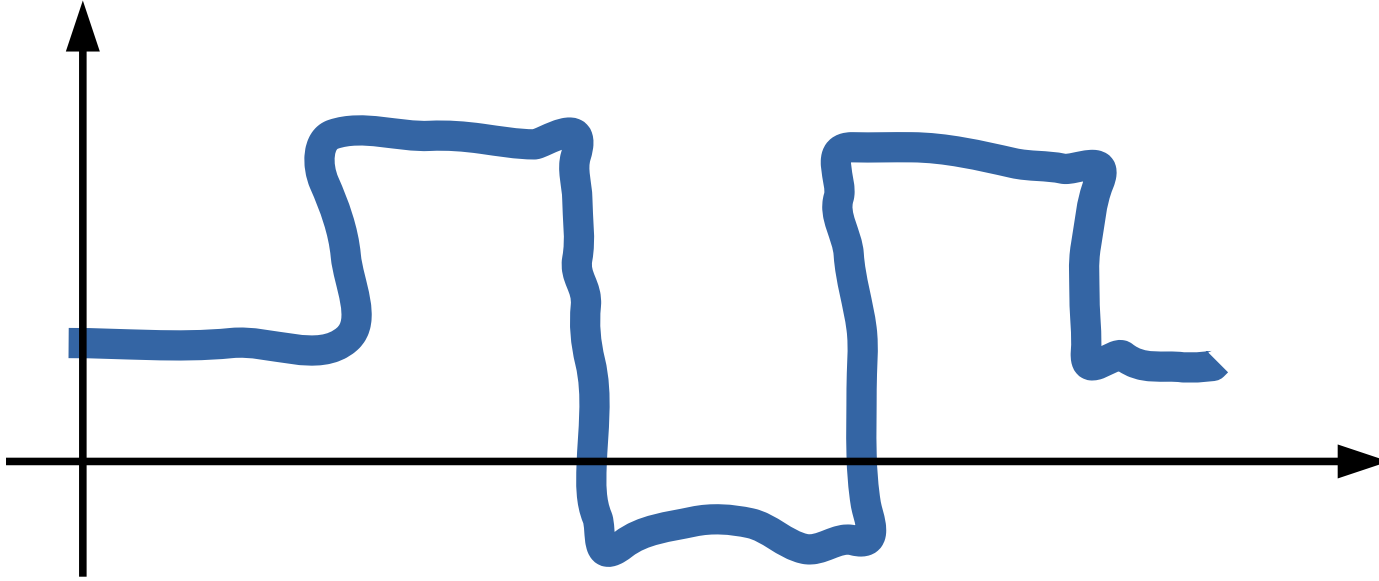
m_n : Mean of negative pulses
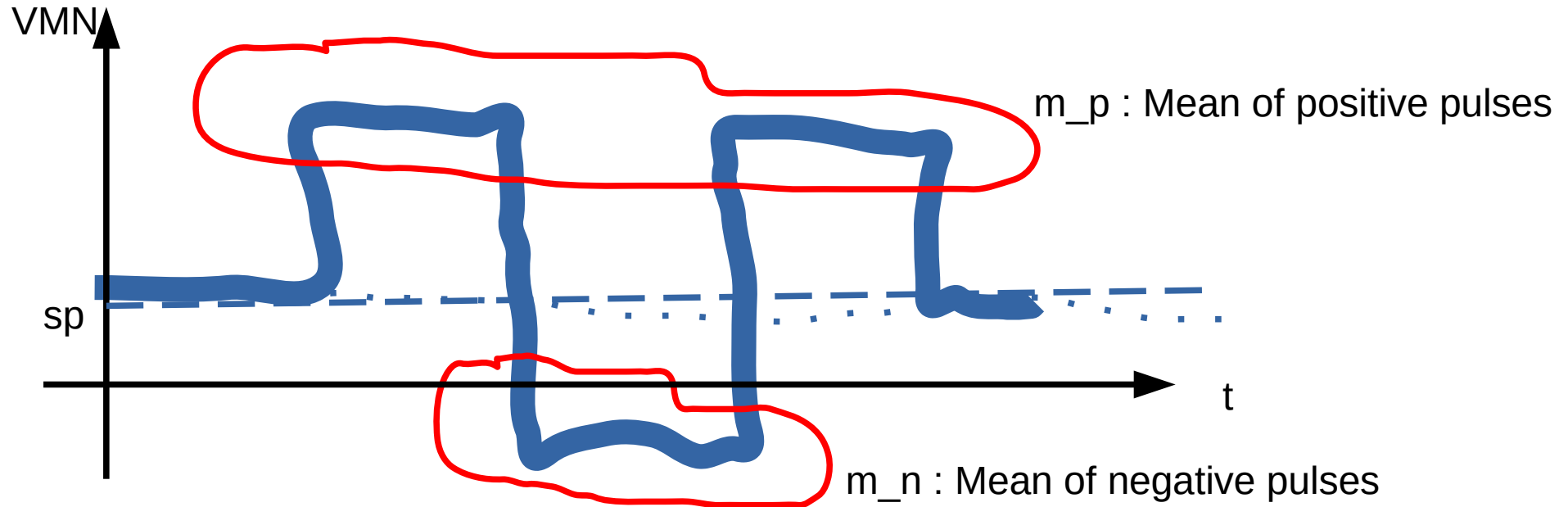
sp = mean(m_p, m_n)   basic strategy if sp is constant (in code_refactor)
(will also work for a linear drift if same number of positive and negative pulses)
Other strategies could be developed

Source de tension TEMNA 5V

Branch MUX_2024 cuve
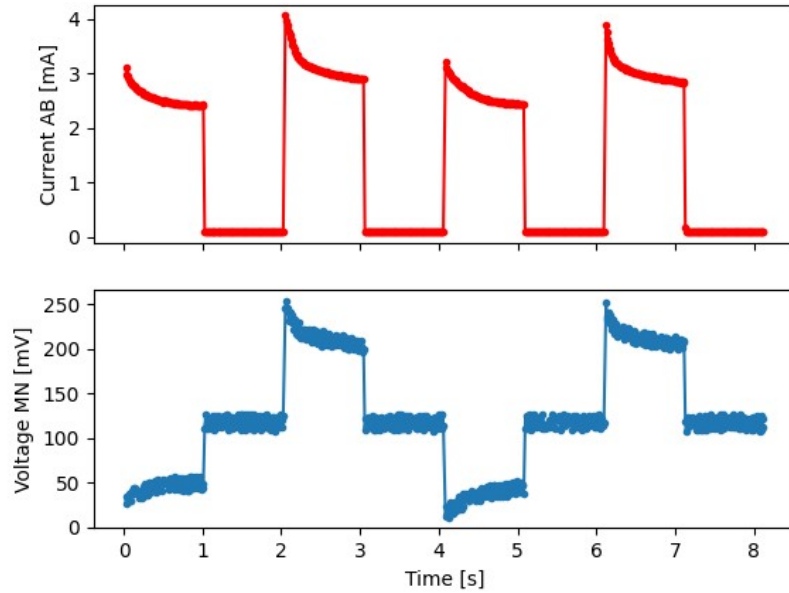
Batterie 12V

R: 45.78 [ohm]
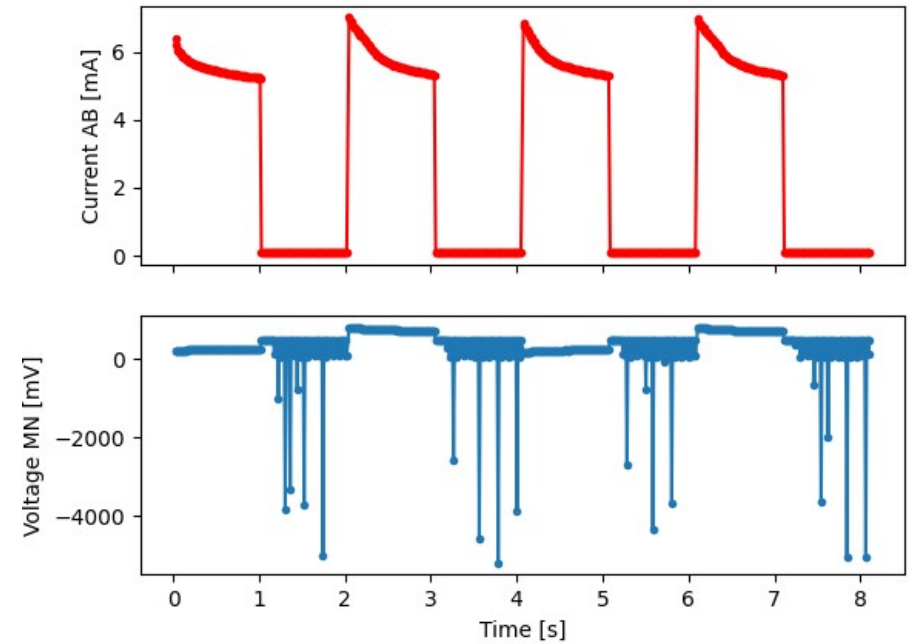
Branch MUX_2024 circuit réf.

Source de tension TEMNA 5V   Batterie 12V

Resistance: 214.82 ohm
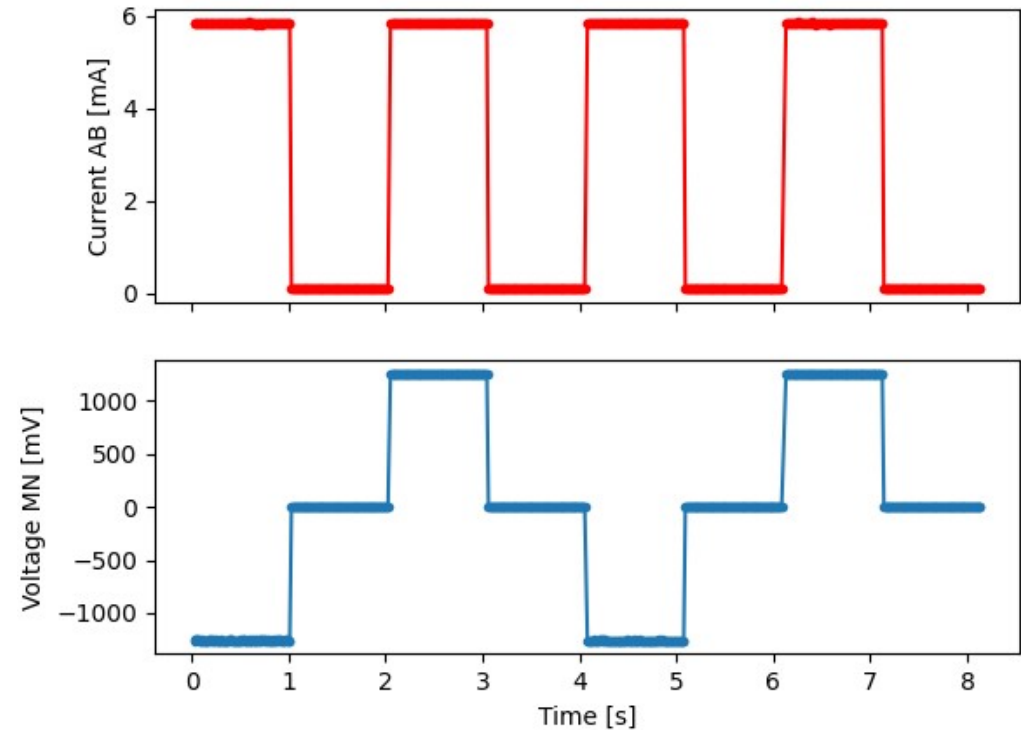
Branch MUX_2024 circuit réf. avec pile ~2.66V entre M et N

Source de tension TEMNA 5V
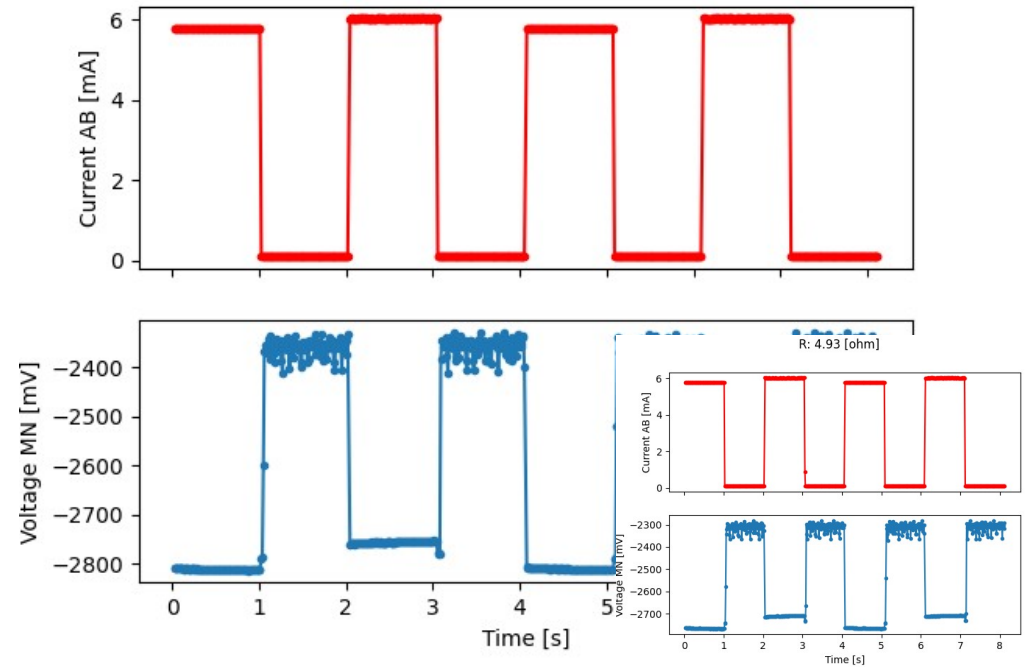
Batterie 12V

Code refactor circuit réf.

Source de tension TEMNA 5V
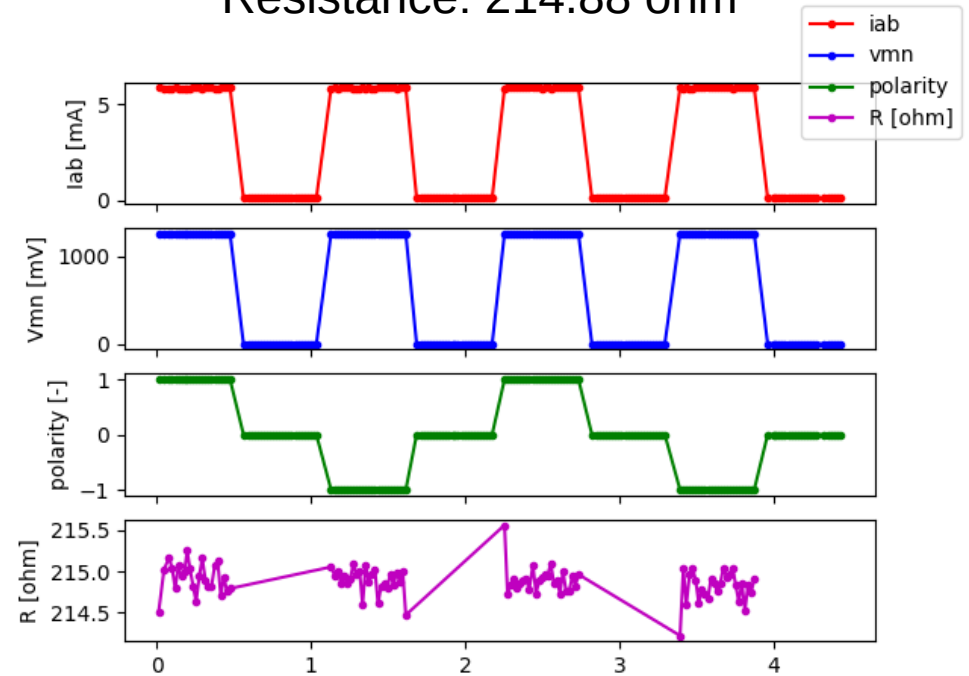
Batterie 12V

Resistance: 214.76 ohm
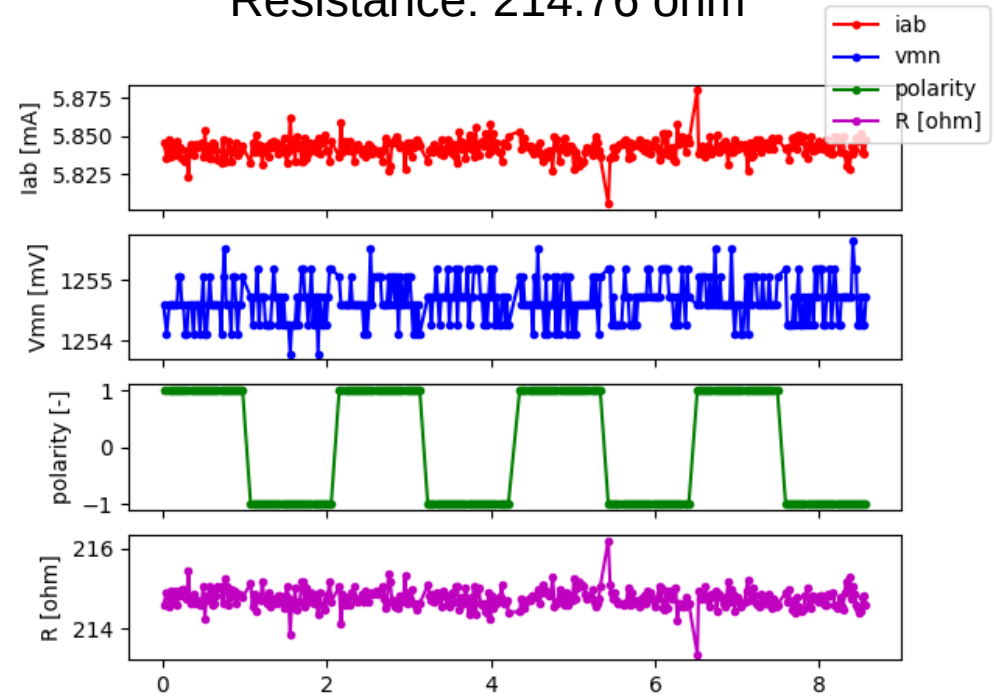
Code refactor circuit réf. duty_cycle = 0.5

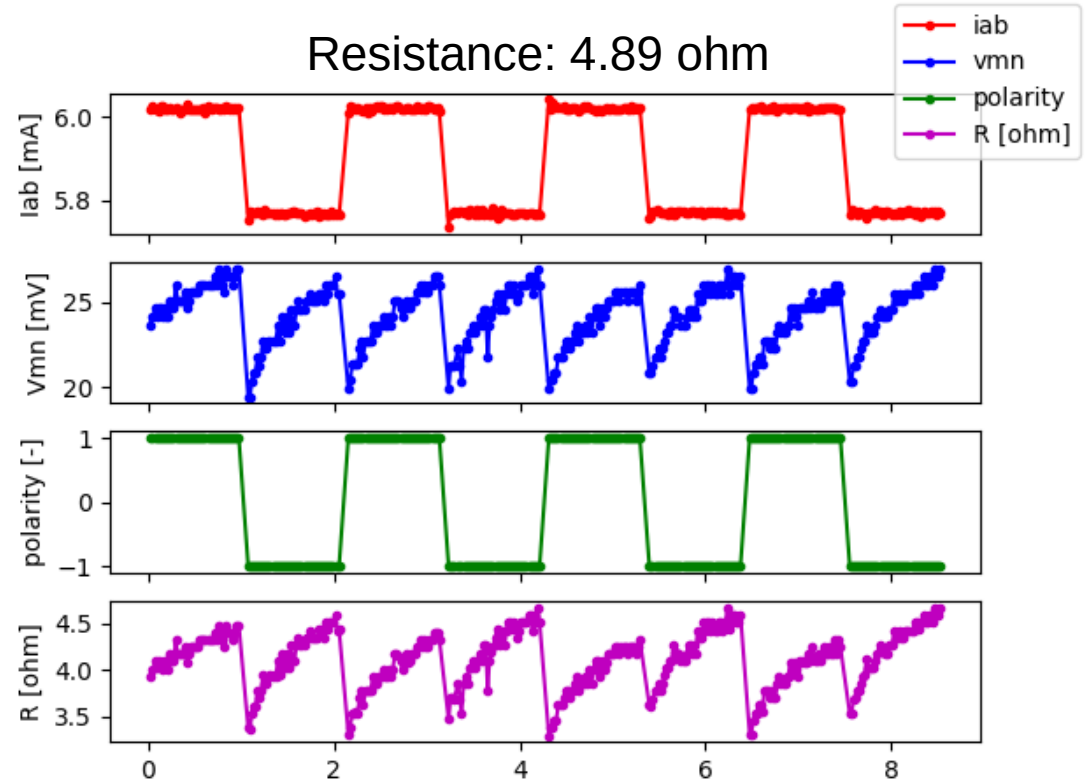Source de tension TEMNA 5V

Batterie 12V

Resistance: 214.88 ohm

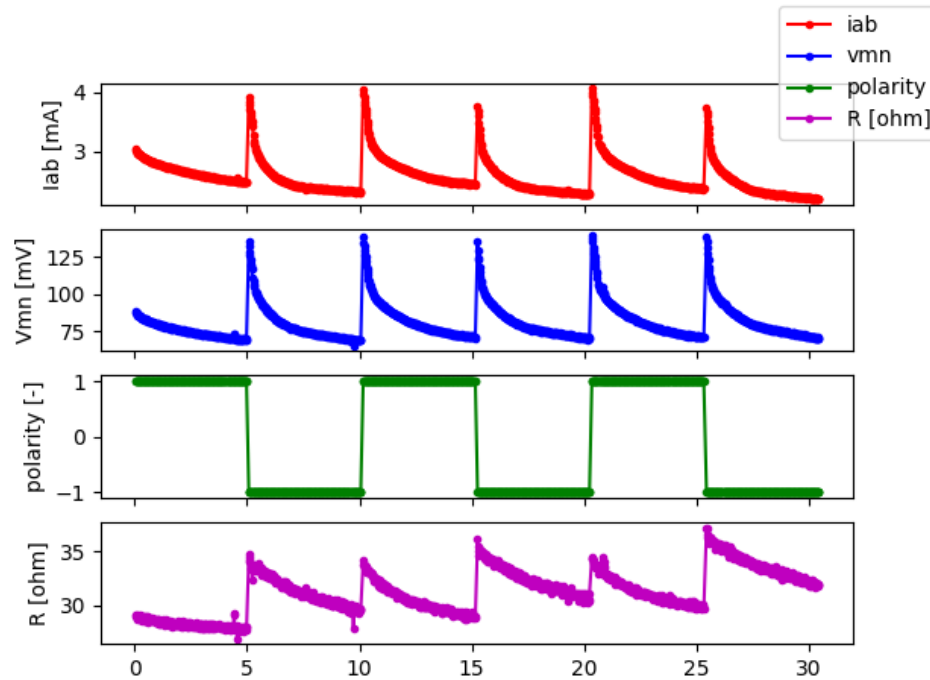Code refactor circuit ref. avec pile ~2.66V entre M et N

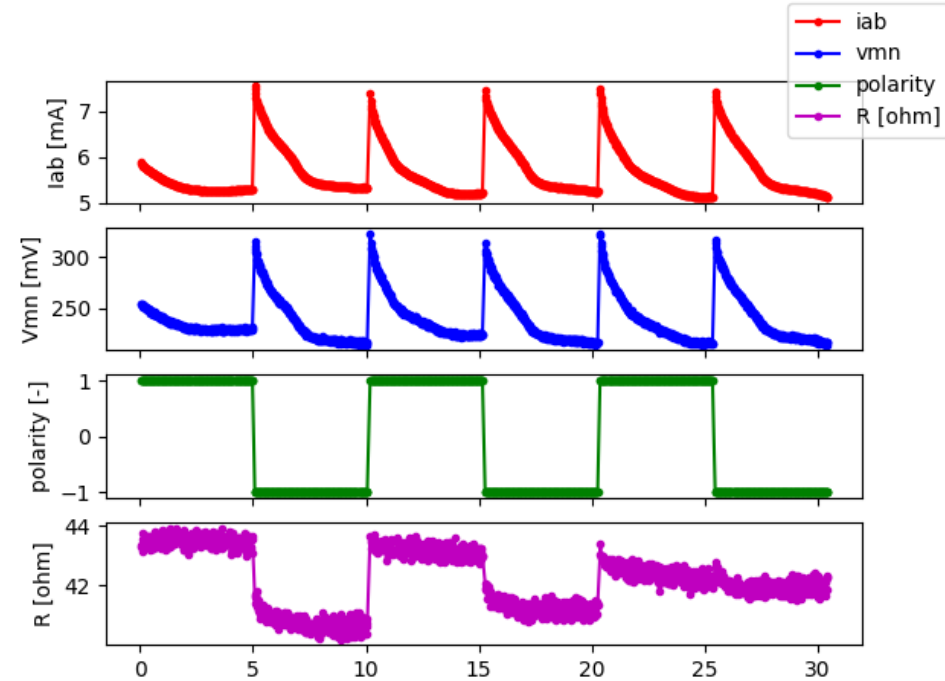Source de tension TEMNA 5V

Batterie 12V

Resistance: 4.89 ohm

Code refactor cuve

Code refactor cuve duty cycle = 0.5

Source de tension TEMNA 5V

Batterie 12V

Resistance: 45.76 ohm