Analog package provides drivers for analog converter and PWM drivers.

ADC (Analog-to-Digital Converter)

Summary

ADC interface provide analogRead functions to read analog inputs.

Supported devices

- ADS1000 (ADS1000.html) series (I2C): ADS1014, ADS1015, ADS1114, ADS1115
- MCP3000 (MCP3000.html) series (SPI): MCP3004, MCP3008, MCP3204, MCP3208

Methods list

analogCount()

Returns the analog channel count.

REST API : GET /devices/**name**/analog/count

- name : device name from configuration file

analogResolution()

Returns the analog resolution (bit count).

REST API : GET /devices/**name**/analog/count

- name : device name from configuration file

analogMaximum()

Returns the analog maximum integer value.

REST API : GET /devices/**name**/analog/maximum

- name : device name from configuration file

analogReference()

Returns the Voltage reference for a full scale.

REST API : GET /devices/**name**/analog/vref

- name : device name from configuration file

analogRead(channel)

Returns the integer value of the given analog channel.

REST API : GET /devices/**name**/analog/**channel**/integer

- name (str) : device name from configuration file
- channel (int) : analog channel number

analogReadFloat(channel)

Returns the float value of the given analog channel, from 0.0 to 1.0.

REST API : GET /devices/**name**/analog/**channel**/float

- name (str) : device name from configuration file
- channel (int) : analog channel number

analogReadVolt(channel)

Returns the voltage value of the given analog channel.

REST API : GET /devices/**name**/analog/**channel**/volt

- name (str) : device name from configuration file
- channel (int) : analog channel number

analogReadAll()

Returns a list containing all analog channels integer value.

REST API : GET /devices/**name**/analog/ * /integer

- name (str) : device name from configuration file

analogReadAllFloat()

Returns a list containing all analog channels float value.

REST API : GET /devices/**name**/analog/ * /float

- name (str) : device name from configuration file

analogReadAllVolt()

Returns a list containing all analog channels voltage value.

REST API : GET /devices/**name**/analog/ * /float

- name (str) : device name from configuration file

Python example

```
from webiopi import deviceInstance
from webiopi.devices.analog import ADS1015, MCP3208
ads = ADS1015(...)          # setup a ADS1015 I2C ADC
# or
ads = deviceInstance("ads") # retrieve device named "ads" in configuration file
ads.analogCount()           # returns ADS1015 analog channel count
ads.analogMaximum()         # returns ADS1015 maximum integer value
ads.analogRead(0)           # returns ADS1015 analog channel 0 as integer
ads.analogReadFloat(0)      # returns ADS1015 analog channel 0 as float
ads.analogReadVolt(0)       # returns ADS1015 analog channel 0 as volt

mcp = MCP3208(...)          # setup a MCP3208 SPI ADC
mcp.analogCount()           # returns MCP3208 analog channel count
mcp.analogMaximum()         # returns MCP3208 maximum integer value
mcp.analogRead(0)           # returns MCP3208 analog channel 0 as integer
mcp.analogReadFloat(0)      # returns MCP3208 analog channel 0 as float
mcp.analogReadVolt(0)       # returns MCP3208 analog channel 0 as volt
```

REST example

```
HTTP GET /devices/ads/analog/0/integer  # returns "ads" analog channel 0 as integer
HTTP GET /devices/ads/analog/0/float    # returns "ads" analog channel 0 as float
HTTP GET /devices/ads/analog/0/volt     # returns "ads" analog channel 0 as volt
```

DAC (Digital-to-Analog Converter)

Summary

DAC interfaces extends ADC to add analogWrite functions.

Supported devices

- MCP4725 (MCP4725.html) (I2C)
- MCP492x (MCP492x.html) (SPI) : MCP4921, MCP4922

Methods list

analogWrite(channel, value)

Write an integer value to the given analog channel.

REST API : POST /devices/**name**/analog/**channel**/integer/**value**

- name (str) : device name from configuration file
- channel (int) : analog channel number
- value (int) : integer value to output from 0 to analogMaximum

analogWriteFloat(channel, ratio)

Write a float ratio value to the given analog channel.

REST API : POST /devices/**name**/analog/**channel**/float/**value**

- name (str) : device name from configuration file
- channel (int) : analog channel number
- value (float) : float ratio to output from 0.0 to 1.0

analogWriteVolt(channel, volts)

Write a voltage value to the given analog channel.

REST API : POST /devices/**name**/analog/**channel**/volt/**value**

- name (str) : device name from configuration file
- channel (int) : analog channel number
- value (float) : voltage value to output from 0.0 to analogReference

Python example

```
from webiopi.devices.analog import MCP4725
mcp = MCP4725(...)           # setup a MCP4725 I2C DAC
mcp.analogCount()            # returns MCP4725 analog channel count
max = mcp.analogMaximum()    # returns MCP4725 maximum integer value
mcp.analogWrite(0, max)      # set 100% on MCP4725 analog channel 0
mcp.analogWriteFloat(0, 0.5) # set  50% on MCP4725 analog channel 0
mcp.analogWriteVolt(0, 0.0)  # set   0V on MCP4725 analog channel 0
mcp.analogRead(0)            # returns MCP4725 analog channel 0 as integer
mcp.analogReadFloat(0)       # returns MCP4725 analog channel 0 as float
mcp.analogReadVolt(0)        # returns MCP4725 analog channel 0 as volt
```

¶ REST example

```
HTTP POST /devices/mcp/analog/0/integer/0    # send 0 to "mcp" analog channel 0
HTTP POST /devices/mcp/analog/0/float/0.0    # send 0 to "mcp" analog channel 0
HTTP POST /devices/mcp/analog/0/volt/0.0     # send 0 to "mcp" analog channel 0
HTTP GET  /devices/mcp/analog/0/integer      # returns "mcp" analog channel 0 as integer
HTTP GET  /devices/mcp/analog/0/float        # returns "mcp" analog channel 0 as float
HTTP GET  /devices/mcp/analog/0/volt         # returns "mcp" analog channel 0 as volt
```

PWM (Pulse Width Modulation)

Summary

PWM interface provides pwmWrite functions to output duty-cycle ratio PWM.

Supported devices

- PCA9685 (PCA9685.html) (I2C)

Methods lists

pwmCount()

Returns the PWM channel count.

REST API : GET /devices/**name**/pwm/count

- name (str) : device name from configuration file

pwmResolution()

Returns the PWM resolution (bit count).

REST API : GET /devices/**name**/pwm/resolution

- name (str) : device name from configuration file

pwmMaximum()

Returns the PWM maximum integer value.

REST API : GET /devices/**name**/pwm/maximum

- name (str) : device name from configuration file

pwmWrite(channel, value)

Write an integer value to the given PWM channel.

REST API : POST /devices/**name**/pwm/**channel**/integer/**value**

- name (str) : device name from configuration file
- channel (int) : analog channel number
- value (int) : integer value to output from 0 to pwmMaximum

pwmWriteFloat(channel, ratio)

Write a float duty-cycle ratio value to the given PWM channel.

REST API : POST /devices/**name**/pwm/**channel**/float/**value**

- name (str) : device name from configuration file
- channel (int) : analog channel number
- value (float) : float duty-cycle ratio value to output from 0 to 1.0

pwmWriteAngle(channel, angle)

Write a servo angle value to the given PWM channel.

REST API : POST /devices/**name**/pwm/**channel**/angle/**value**

- name (str) : device name from configuration file
- channel (int) : analog channel number
- value (float) : angle value to output

pwmRead(channel)

Returns the integer value of the given PWM channel.

REST API : GET /devices/**name**/pwm/**channel**/integer

- name (str) : device name from configuration file
- channel (int) : analog channel number

pwmReadFloat(channel)

Returns the float duty-cycle ratio value of the given PWM channel.

REST API : GET /devices/**name**/pwm/**channel**/float

- name (str) : device name from configuration file
- channel (int) : analog channel number

pwmReadAngle(channel)

Returns the servo angle value of the given PWM channel.

REST API : GET /devices/**name**/pwm/**channel**/angle

- name (str) : device name from configuration file
- channel (int) : analog channel number

pwmReadAll()

Returns a list containing all PWM channels integer value.

REST API : GET /devices/**name**/pwm/ * /integer

- name (str) : device name from configuration file
- channel (int) : analog channel number

pwmReadAllFloat()

Returns a list containing all PWM channels float duty-cycle ratio value.

REST API : GET /devices/**name**/pwm/ * /float

- name (str) : device name from configuration file
- channel (int) : analog channel number

pwmReadAllAngle()

Returns a list containing all PWM channels servo angle value.

REST API : GET /devices/**name**/pwm/ * /angle

- name (str) : device name from configuration file
- channel (int) : analog channel number

pwmWildcard()

Returns a list containing all PWM channels value.

REST API : GET /devices/**name**/pwm/ *

- name (str) : device name from configuration file

Python example

```
from webiopi import deviceInstance
from webiopi.devices.analog import PCA9685
pca = PCA9685(...)          # setup a PCA9685 I2C PWM
# or
pca = deviceInstance("pca") # retrieve device named "pca" in configuration file
pca.pwmCount()              # returns PCA9685 pwm channel count
max = pca.pwmMaximum()      # returns PCA9685 maximum integer value
pca.pwmWrite(0, max)        # set 100% on PCA9685 pwm channel 0 (integer value)
pca.pwmWriteFloat(0, 0.5)   # set  50% on PCA9685 pwm channel 0 (float value)
pca.pwmWriteAngle(0, 0.0)   # set   0° on PCA9685 pwm channel 0 (servo angle value)
pca.pwmRead(0)              # returns PCA9685 pwm channel 0 as integer
pca.pwmReadFloat(0)         # returns PCA9685 pwm channel 0 as float
pca.pwmReadAngle(0)         # returns PCA9685 pwm channel 0 as angle
```

REST example

```
HTTP POST /devices/pca/pwm/0/integer/0  # set 0% on "pca" pwm channel 0 (integer value)
HTTP POST /devices/pca/pwm/0/float/0.0  # set 0% on "pca" pwm channel 0 (float value)
HTTP POST /devices/pca/pwm/0/angle/0.0  # set 0° on "pca" pwm channel 0 (servo angle value)
HTTP GET  /devices/pca/pwm/0/integer    # returns "pca" pwm channel 0 as integer
HTTP GET  /devices/pca/pwm/0/float      # returns "pca" pwm channel 0 as float
HTTP GET  /devices/pca/pwm/0/angle      # returns "pca" pwm channel 0 as servo angle
```