NodeMCU API Instruction

version 0.9.5 build 2015-01-24

目录

Summary

- Easy to access wireless router
- Based on Lua 5.1.4, Developers are supposed to have experience with Lua Program language.
- Event-Drive programming modal.
- Build-in file, timer, pwm, i2c, net, gpio, wifi, uart, adc module.
- Serial Port BaudRate:9600
- Re-mapped GPIO pin, use the index to program gpio, i2c, pwm.
- GPIO Map Table:

GPIO NEW TABLE (Build 20141219 and later)

IO index	ESP8266 pin	IO index	ESP8266 pin
0 [*]	GPIO16	8	GPIO15
1	GPIO5	9	GPIO3
2	GPIO4	10	GPIO1
3	GPIO0	11	GPIO9
4	GPIO2	12	GPIO10
5	GPIO14		
6	GPIO12		
7	GPIO13		

[*] D0(GPIO16) can only be used as gpio read/write. no interrupt supported. no pwm/i2c/ow supported.

-Back to Index

GPIO OLD TABLE (Before build 20141212)

IO index	ESP8266 pin	IO index	ESP8266 pin
0	GPIO12	8	GPIO0
1	GPIO13	9	GPIO2
2	GPIO14	10	GPIO4
3	GPIO15	11	GPIO5
4	GPIO3		

5	GPIO1		
6	GPIO9		
7	GPIO10		

-Back to Index

Burn/Flash Firmware

Address

nodemcu_512k.bin: 0x00000
See NodeMCU flash tool:
nodemcu-flasher

node module

node.restart()

Description

restart the chip.

Syntax

node.restart()

Parameters

nil

Returns

nil

Example

```
    node.restart();
```

See also

-
- Back to Index

node.dsleep()

Description

Enter deep sleep mode, wake up when timed out

Syntax

node.dsleep(us)
-Note: This function can only be used in the condition that esp8266 PIN32(RST) and PIN8(XPD_DCDC) are connected together.

Parameters

us: sleep time in micro second

Returns

nil

Example

```
    node.dsleep(us);
```

See also

-
- Back to Index

node.info()

Description

return NodeMCU version, chipid, flashid, flash size, flash mode, flash speed.

Syntax

node.info()

Parameters

nil

Returns

number: majorVer, minorVer, devVer, chipid, flashid, flashsize, flashmode, flashspeed.

Example

```
    majorVer, minorVer, devVer, chipid, flashid, flashsize, flashmode, flashspeed =
node.info();
    print("NodeMCU "..majorVer.."."..minorVer.."."..devVer)
```

See also

-
- Back to Index

node.chipid()

Description

return chip ID

Syntax

node.chipid()

Parameters

nil

Returns

number:chip ID

Example

```
    id = node.chipid();
```

See also

-

- Back to Index

node.flashid()

Description

return flashid ID

Syntax

node.flashid()

Parameters

nil

Returns

number:flash ID

Example

```
    flashid = node.flashid();
```

See also

-

- Back to Index

node.heap()

Description

return the remain HEAP size in bytes

Syntax

node.heap()

Parameters

nil

Returns

number: system heap size left in bytes

Example

```
    heap_size = node.heap();
```

See also

-
- Back to Index

node.key()

Description

define button function, button is connected to GPIO16.

Syntax

node.key(type, function())

Parameters

type: type is either string "long" or "short". long: press the key for 3 seconds, short: press shortly(less than 3 seconds)
function(): user defined function which is called when key is pressed. If nil, cancling the user defined function.
Default function: long: change LED blinking rate, short: reset chip

Returns

nil

Example

```
    node.key("long", function() print('hello world') end)
```

See also

-
- Back to Index

node.led()

Description

setup the on/off time for led, which connected to GPIO16, multiplexing with node.key()

Syntax

node.led(low, high)

Parameters

Low: LED off time, LED keeps on when low=0. Unit: milliseconds, time resolution: 80~100ms
High: LED on time. Unit: milliseconds, time resolution: 80~100ms

Returns

nil

Example

```
-- turn led on forever.
node.led(0);
```

See also

-
- Back to Index

node.input()

Description

accept a string and put the string into Lua interpretor.
same as pcall(loadstring(str)) but support multi seperated line.

Syntax

node.input(str)

Parameters

str: Lua chunk

Returns

nil

Example

```
-- never use node.input() in console. no effect.
sk:on("receive", function(conn, payload) node.input(payload) end)
```

See also

-
- Back to Index

node.output()

Description

direct output from lua interpretor to a call back function.

Syntax

node.output(function(str), serial_debug)

Parameters

function(str): a function accept every output as str, and can send the output to a socket.
serial_debug: 1 output also show in serial. 0: no serial output.

Returns

nil

Example

```
    function tonet(str)
      sk:send(str)
      -- print(str) WRONG!!! never ever print something in this function
      -- because this will cause a recursive function call!!!
    end
    node.ouput(tonet, 1)  -- serial also get the lua output.
    -- a simple telnet server
    s=net.createServer(net.TCP)
    s:listen(2323,function(c)
       con_std = c
       function s_output(str)
          if(con_std~=nil)
             then con_std:send(str)
          end
       end
       node.output(s_output, 0)   -- re-direct output to function s_ouput.
       c:on("receive",function(c,l)
          node.input(l)           -- works like pcall(loadstring(l)) but support
multiple separate line
       end)
       c:on("disconnection",function(c)
          con_std = nil
          node.output(nil)        -- un-regist the redirect output function, output goes
to serial
       end)
    end)
```

See also

-

- Back to Index

file module

file.remove()

Description

remove file from file system.

Syntax

file.remove(filename)

Parameters

filename: file to remove

Returns

nil

Example

```
-- remove "foo.lua" from file system.
file.remove("foo.lua")
```

See also

- file.open()
- file.close()
- Back to Index

file.open()

Description

open file.

Syntax

file.open(filename, mode)

Parameters

filename: file to be opened, directories are not supported
mode:
"r": read mode (the default)
"w": write mode
"a": append mode
"r+": update mode, all previous data is preserved
"w+": update mode, all previous data is erased

"a+": append update mode, previous data is preserved, writing is only allowed at the end of file

Returns

nil: file not opened, or not exists. true: file opened ok.

Example

```
-- open 'init.lua', print the first line.
file.open("init.lua", "r")
print(file.readline())
file.close()
```

See also

- file.close()
- file.readline()
- Back to Index

file.close()

Description

close the file.

Syntax

file.close()

Parameters

nil

Returns

nil

Example

```
-- open 'init.lua', print the first line.
file.open("init.lua", "r")
print(file.readline())
file.close()
```

See also

- file.open()
- file.readline()
- Back to Index

file.readline()

Description

read one line of file which is opened before.

Syntax

file.readline()

Parameters

nil

Returns

file content in string, line by line, include EOL('\n')
return nil when EOF.

Example

```
-- print the first line of 'init.lua'
file.open("init.lua", "r")
print(file.readline())
file.close()
```

See also

- file.open()
- file.close()
- Back to Index

file.writeline()

Description

write string to file and add a '\n' at the end.

Syntax

file.writeline(string)

Parameters

string: content to be write to file

Returns

true: write ok. nil: there is error

Example

```
-- open 'init.lua' in 'a+' mode
file.open("init.lua", "a+")
-- write 'foo bar' to the end of the file
file.writeline('foo bar')
```

```
file.close()
```

See also

- file.open()
- file.write()
- Back to Index

file.read()

Description

read content of file which is opened before.

Syntax

file.read()

Parameters

if nothing passed in, read all byte in file. if pass a number n, then read n byte from file, or EOF is reached. if pass a string "q", then read until 'q' or EOF is reached.

Returns

file content in string
return nil when EOF.

Example

```
-- print the first line of 'init.lua'
file.open("init.lua", "r")
print(file.read('\r'))
file.close()

-- print the first 5 byte of 'init.lua'
file.open("init.lua", "r")
print(file.read(5))
file.close()
```

See also

- file.open()
- file.close()
- Back to Index

file.write()

Description

write string to file.

Syntax

file.write(string)

Parameters

string: content to be write to file.

Returns

true: write ok. nil: there is error

Example

```
-- open 'init.lua' in 'a+' mode
file.open("init.lua", "a+")
-- write 'foo bar' to the end of the file
file.write('foo bar')
file.close()
```

See also

- file.open()
- file.writeline()
- Back to Index

file.flush()

Description

flush to file.

Syntax

file.flush()

Parameters

nil

Returns

nil

Example

```
-- open 'init.lua' in 'a+' mode
file.open("init.lua", "a+")
-- write 'foo bar' to the end of the file
file.write('foo bar')
file.flush()
file.close()
```

See also

- file.open()
- file.writeline()
- Back to Index

file.seek()

Description

Sets and gets the file position, measured from the beginning of the file, to the position given by offset plus a base specified by the string whence.

Syntax

file.seek(whence, offset)

Parameters

whence:
"set": base is position 0 (beginning of the file);
"cur": base is current position;(default value)
"end": base is end of file;
offset: default 0

Returns

success: returns the final file position
fail: returns nil

Example

```
    -- open 'init.lua' in 'a+' mode
    file.open("init.lua", "a+")
    -- write 'foo bar' to the end of the file
    file.write('foo bar')
    file.flush()
    file.seek("set")
    print(file.readline())
    file.close()
```

See also

- file.open()
- file.writeline()
- Back to Index

file.list()

Description

list all files.

Syntax

file.list()

Parameters

nil

Returns

a lua table which contains the {file name: file size} pairs

Example

```
l = file.list();
for k,v in pairs(l) do
  print("name:"..k..", size:"..v)
end
```

See also

- file.remove()
- Back to Index

file.format()

Description

format file system.

Syntax

file.format()

Parameters

nil

Returns

nil

Example

```
file.format()
```

See also

- file.remove()
- Back to Index

wifi module

CONSTANT

wifi.STATION, wifi.SOFTAP, wifi.STATIONAP

wifi.setmode(mode)

Description

setup wifi operation mode.

Syntax

wifi.setmode(mode)

Parameters

mode: value should be: wifi.STATION, wifi.SOFTAP or wifi.STATIONAP

Returns

current mode after setup

Example

```
wifi.setmode(wifi.STATION)
```

See also

- wifi.getmode()
- Back to Index

wifi.getmode(mode)

Description

get wifi operation mode.

Syntax

wifi.getmode()

Parameters

nil

Returns

wifi operation mode

Example

```
print(wifi.getmode())
```

See also

- wifi.setmode()
- Back to Index

wifi.startsmart()

Description

starts to auto configuration, if success set up ssid and pwd automatically .

Syntax

wifi.startsmart(channel, function succeed_callback())

Parameters

channel: 1~13, startup channel for searching, if nil, default to 6. 20 seconds for each channel.
succeed_callback: callback function called after configuration, which is called when got password and connected to AP.

Returns

nil

Example

```
    wifi.startsmart(6, function() end)
```

See also

- wifi.stopsmart()
- Back to Index

wifi.stopsmart()

Description

stop the configuring process.

Syntax

wifi.stopsmart()

Parameters

nil

Returns

nil

Example

```
    wifi.stopsmart()
```

See also

- wifi.startsmart()
- Back to Index

wifi.sleeptype()

Description

config the sleep type for wifi modem.

Syntax

type_actual = wifi.sleeptype(type_need)

Parameters

type_need:
wifi.NONE_SLEEP, wifi.LIGHT_SLEEP, wifi.MODEM_SLEEP

Returns

type_actual:
wifi.NONE_SLEEP, wifi.LIGHT_SLEEP, wifi.MODEM_SLEEP

Example

```
realtype = wifi.sleeptype(wifi.MODEM_SLEEP)
```

See also

- node.dsleep()
- Back to Index

wifi.sta module

wifi.sta.config()

Description

set ssid and password in station mode.

Syntax

wifi.sta.config(ssid, password)

Parameters

ssid: string which is less than 32 bytes.
password: string which is less than 64 bytes.

Returns

nil

Example

```
    wifi.sta.config("myssid","mypassword")
```

See also

- wifi.sta.connect()
- wifi.sta.disconnect()
- Back to Index

wifi.sta.connect()

Description

connect to AP in station mode.

Syntax

wifi.sta.connect()

Parameters

nil

Returns

nil

Example

```
    wifi.sta.connect()
```

See also

- wifi.sta.disconnect()
- wifi.sta.config()
- Back to Index

wifi.sta.disconnect()

Description

disconnect from AP in station mode.

Syntax

wifi.sta.disconnect()

Parameters

nil

Returns

nil

Example

```
wifi.sta.disconnect()
```

See also

- wifi.sta.config()
- wifi.sta.connect()
- Back to Index

wifi.sta.autoconnect()

Description

auto connect to AP in station mode.

Syntax

wifi.sta.autoconnect(auto)

Parameters

auto: 0 to disable auto connecting. 1 to enable auto connecting

Returns

nil

Example

```
wifi.sta.autoconnect()
```

See also

- wifi.sta.config()
- wifi.sta.connect()
- wifi.sta.disconnect()
- Back to Index

wifi.sta.getip()

Description

get ip, netmask, gateway address in station mode.

Syntax

wifi.sta.getip()

Parameters

nil

Returns

ip, netmask, gateway address in string, for example:"192.168.0.111"
return nil if ip = "0.0.0.0".

Example

```
-- print current ip, netmask, gateway
print(wifi.sta.getip())
-- 192.168.0.111  255.255.255.0  192.168.0.1
ip = wifi.sta.getip()
print(ip)
-- 192.168.0.111
ip, nm = wifi.sta.getip()
print(nm)
-- 255.255.255.0
```

See also

- wifi.sta.getmac()
- Back to Index

wifi.sta.setip()

Description

set ip, netmask, gateway address in station mode.

Syntax

wifi.sta.setip(cfg)

Parameters

cfg: table contain ip, netmask, and gateway

```
{
  "ip":"192.168.0.111",
  "netmask":"255.255.255.0",
  "gateway":"192.168.0.1"
}
```

Returns

true if success, false if fail.

Example

```
cfg =
{
  "ip":"192.168.0.111",
  "netmask":"255.255.255.0",
  "gateway":"192.168.0.1"
}
```

```
wifi.sta.setip(cfg)
```

See also

- wifi.sta.setmac()
- Back to Index

wifi.sta.getmac()

Description

get mac address in station mode.

Syntax

wifi.sta.getmac()

Parameters

nil

Returns

mac address in string, for example:"18-33-44-FE-55-BB"

Example

```
-- print current mac address
print(wifi.sta.getmac())
```

See also

- wifi.sta.getip()
- Back to Index

wifi.sta.setmac()

Description

set mac address in station mode.

Syntax

wifi.sta.setmac(mac)

Parameters

mac address in byte string, for example:"\024\024\024\024\024\024"

Returns

true if success, false if fail.

Example

```
print(wifi.sta.setmac("\024\024\024\024\024\024"))
```

See also

- wifi.sta.setip()
- Back to Index

wifi.sta.getap()

Description

scan and get ap list as a lua table into callback function.

Syntax

wifi.sta.getap(function(table))

Parameters

function(table): a callback function to receive ap table when scan is done
this function receive a table, the key is the ssid, value is other info in format:
authmode,rssi,bssid,channel

Returns

nil

Example

```
-- print ap list
function listap(t)
  for k,v in pairs(t) do
    print(k.." : "..v)
  end
end
wifi.sta.getap(listap)
```

See also

- wifi.sta.getip()
- Back to Index

wifi.sta.status()

Description

get current status in station mode.

Syntax

wifi.sta.status()

Parameters

nil

Returns

number： 0~5 0: STATION_IDLE, 1: STATION_CONNECTING, 2: STATION_WRONG_PASSWORD, 3: STATION_NO_AP_FOUND, 4: STATION_CONNECT_FAIL, 5: STATION_GOT_IP.

See also

-
- Back to Index

wifi.sta.getbroadcast()

Description

get getbroadcast address in station mode.

Syntax

wifi.sta.getbroadcast()

Parameters

nil

Returns

getbroadcast address in string, for example:"192.168.0.255"
return nil if ip = "0.0.0.0".

Example

```
bc = wifi.sta.getbroadcast()
print(bc)
-- 192.168.0.255
```

See also

- wifi.sta.getip()
- Back to Index

wifi.ap module

wifi.ap.config()

Description

set ssid and password in ap mode.

Syntax

wifi.ap.config(cfg)

Parameters

cfg: lua table to setup ap.

Example:

```
    cfg={}
    cfg.ssid="myssid"
    cfg.pwd="mypwd"
    wifi.ap.config(cfg)
```

Returns

nil

See also

-
- Back to Index

wifi.ap.getip()

Description

get ip, netmask, gateway in ap mode.

Syntax

wifi.ap.getip()

Parameters

nil

Returns

ip, netmask, gateway address in string, for example:"192.168.0.111"
return nil if ip = "0.0.0.0".

Example

```
    -- print current ip, netmask, gateway
    print(wifi.ap.getip())
    -- 192.168.4.1  255.255.255.0  192.168.4.1
    ip = wifi.ap.getip()
    print(ip)
    -- 192.168.4.1
    ip, nm = wifi.ap.getip()
    print(nm)
    -- 255.255.255.0
```

See also

- wifi.ap.getmac()
- Back to Index

wifi.ap.setip()

Description

set ip, netmask, gateway address in ap mode.

Syntax

wifi.ap.setip(cfg)

Parameters

cfg: table contain ip, netmask, and gateway

```
{
  "ip":"192.168.1.1",
  "netmask":"255.255.255.0",
  "gateway":"192.168.1.1"
}
```

Returns

true if success, false if fail.

Example

```
cfg =
{
  "ip":"192.168.1.1",
  "netmask":"255.255.255.0",
  "gateway":"192.168.1.1"
}
wifi.ap.setip(cfg)
```

See also

- wifi.ap.setmac()
- Back to Index

wifi.ap.getmac()

Description

get mac address in ap mode.

Syntax

wifi.ap.getmac()

Parameters

nil

Returns

mac address in string, for example:"1A-33-44-FE-55-BB"

Example

```
wifi.ap.getmac()
```

See also

- wifi.ap.getip()
- Back to Index

wifi.ap.setmac()

Description

set mac address in ap mode.

Syntax

wifi.ap.setmac(mac)

Parameters

mac address in byte string, for example:"\024\024\024\024\024\024"

Returns

true if success, false if fail.

Example

```
print(wifi.ap.setmac("\024\024\024\024\024\024"))
```

See also

- wifi.ap.setip()
- Back to Index

wifi.ap.getbroadcast()

Description

get getbroadcast address in ap mode.

Syntax

wifi.ap.getbroadcast()

Parameters

nil

Returns

getbroadcast address in string, for example:"192.168.0.255"
return nil if ip = "0.0.0.0".

Example

```
bc = wifi.ap.getbroadcast()
print(bc)
-- 192.168.0.255
```

See also

- wifi.ap.getip()
- Back to Index

timer module

tmr.delay()

Description

delay us micro seconds.

Syntax

tmr.delay(us)

Parameters

us: delay time in micro second

Returns

nil

Example

```
-- delay 100us
tmr.delay(100)
```

See also

- tmr.now()
- Back to Index

tmr.now()

Description

return the current value of system counter: uint31, us.

Syntax

tmr.now()

Parameters

nil

Returns

uint31: value of counter

Example

```
    -- print current value of counter
    print(tmr.now())
```

See also

- tmr.delay()
- Back to Index

tmr.alarm()

Description

alarm time.

Syntax

tmr.alarm(id, interval, repeat, function do())

Parameters

id: 0~6, alarmer id. Interval: alarm time, unit: millisecond
repeat: 0 - one time alarm, 1 - repeat
function do(): callback function for alarm timed out

Returns

nil

Example

```
    -- print "hello world" every 1000ms
    tmr.alarm(0, 1000, 1, function() print("hello world") end )
```

See also

- tmr.now()
- Back to Index

tmr.stop()

Description

stop alarm.

Syntax

tmr.stop(id)

Parameters

id: 0~6, alarmer id.

Returns

nil

Example

```
    -- print "hello world" every 1000ms
    tmr.alarm(1, 1000, 1, function() print("hello world") end )

    -- something else

    -- stop alarm
    tmr.stop(1)
```

See also

- tmr.now()
- Back to Index

tmr.wdclr()

Description

clear system watchdog counter.

Syntax

tmr.wdclr()

Parameters

nil.

Returns

nil

Example

```
    for i=1,10000 do
      print(i)
      tmr.wdclr()   -- should call tmr.wdclr() in a long loop to avoid hardware reset
caused by watchdog.
    end
```

See also

- tmr.delay()
- Back to Index

tmr.time()

Description

return rtc time since start up in second, uint31 form.

Syntax

tmr.time()

Parameters

nil.

Returns

number

Example

See also

- tmr.now()
- Back to Index

GPIO module

CONSTANT

gpio.OUTPUT, gpio.INPUT, gpio.INT, gpio.HIGH, gpio.LOW

gpio.mode()

Description

initialize pin to GPIO mode, set the pin in/out mode, internal pullup.

Syntax

gpio.mode(pin, mode, pullup)

Parameters

pin: 0~12, IO index
mode: gpio.OUTPUT or gpio.INPUT, or gpio.INT(interrupt mode) pullup: gpio.PULLUP or gpio.FLOAT, default: gpio.FLOAT.

Returns

nil

Example

```
    -- set gpio 0 as output.
    gpio.mode(0, gpio.OUTPUT)
```

See also

- gpio.read()
- Back to Index

gpio.read()

Description

read pin value.

Syntax

gpio.read(pin)

Parameters

pin: 0~12, IO index

Returns

number:0 - low, 1 - high

Example

```
    -- read value of gpio 0.
    gpio.read(0)
```

See also

- gpio.mode()
- Back to Index

gpio.write()

Description

set pin value.

Syntax

gpio.write(pin)

Parameters

pin: 0~12, IO index
level: gpio.HIGH or gpio.LOW

Returns

nil

Example

```
-- set pin index 1 to GPIO mode, and set the pin to high.
pin=1
gpio.mode(pin, gpio.OUTPUT)
gpio.write(pin, gpio.HIGH)
```

See also

- gpio.mode()
- gpio.read()
- Back to Index

gpio.trig()

Description

set the interrupt callback function for pin.

Syntax

gpio.trig(pin, type, function(level))

Parameters

pin: **1~12**, IO index, pin D0 does not support Interrupt.
type: "up", "down", "both", "low", "high", which represent rising edge, falling edge, both edge, low level, high level trig mode separately.
function(level): callback function when triggered. The gpio level is the param. Use previous callback function if undefined here.

Returns

nil

Example

```
-- use pin 0 as the input pulse width counter
pulse1 = 0
du = 0
gpio.mode(1,gpio.INT)
function pin1cb(level)
 du = tmr.now() - pulse1
 print(du)
 pulse1 = tmr.now()
 if level == 1 then gpio.trig(1, "down ") else gpio.trig(1, "up ") end
end
```

```
gpio.trig(1, "down ",pin1cb)
```

See also

- gpio.mode()
- gpio.write()
- Back to Index

PWM module

pwm.setup()

Description

set pin to PWM mode. Only 3 pins can be set to PWM mode at the most.

Syntax

pwm.setup(pin, clock, duty)

Parameters

pin: 1~12, IO index
clock: 1~1000, pwm frequency
duty: 0~1023, pwm duty cycle, max 1023(10bit)

Returns

nil

Example

```
-- set pin index 1 as pwm output, frequency is 100Hz, duty cycle is half.
pwm.setup(1, 100, 512)
```

See also

- pwm.start()
- Back to Index

pwm.close()

Description

quit PWM mode for specified pin.

Syntax

pwm.close(pin)

Parameters

pin: 1~12, IO index

Returns

nil

Example

```
    pwm.close(1)
```

See also

- pwm.start()
- Back to Index

pwm.start()

Description

pwm starts, you can detect the waveform on the gpio.

Syntax

pwm.start(pin)

Parameters

pin: 1~12, IO index

Returns

nil

Example

```
    pwm.start(1)
```

See also

- pwm.stop()
- Back to Index

pwm.stop()

Description

pause the output of PWM waveform.

Syntax

pwm.stop(pin)

Parameters

pin: 1~12, IO index

Returns

nil

Example

```
    pwm.stop(1)
```

See also

- pwm.start()
- Back to Index

pwm.setclock()

Description

set pwm frequency for pin.
-Note: setup pwm frequency will synchronously change others if there are any. Only one PWM frequency can be allowed for the system.

Syntax

pwm.setclock(pin, clock)

Parameters

pin: 1~12, IO index.
clock: 1~1000, pwm frequency.

Returns

nil

Example

```
    pwm.setclock(1, 100)
```

See also

- pwm.getclock()
- Back to Index

pwm.getclock()

Description

get pwm frequency of pin.

Syntax

pwm.getclock(pin)

Parameters

pin: 1~12, IO index.

Returns

number:pwm frequency of pin

Example

```
    print(pwm.getclock(1))
```

See also

- pwm.setclock()
- Back to Index

pwm.setduty()

Description

set duty clycle for pin.

Syntax

pwm.setduty(pin, duty)

Parameters

pin: 1~12, IO index
duty: 0~1023, pwm duty cycle, max 1023(10bit).

Returns

nil

Example

```
    pwm.setduty(1, 512)
```

See also

- pwm.getduty()
- Back to Index

pwm.getduty()

Description

get duty clycle for pin.

Syntax

pwm.getduty(pin)

Parameters

pin: 1~12, IO index

Returns

number: duty cycle, max 1023.

Example

```
-- D1 is connected to green led
-- D2 is connected to blue led
-- D3 is connected to red led
pwm.setup(1,500,512)
pwm.setup(2,500,512)
pwm.setup(3,500,512)
pwm.start(1)
pwm.start(2)
pwm.start(3)
function led(r,g,b)
  pwm.setduty(1,g)
  pwm.setduty(2,b)
  pwm.setduty(3,r)
end
led(512,0,0) --  set led to red
led(0,0,512) -- set led to blue.
```

See also

- pwm.setduty()
- Back to Index

net module

CONSTANT

net.TCP, net.UDP

net.createServer()

Description

create a server.

Syntax

net.createServer(type, timeout)

Parameters

type: net.TCP or net.UDP
timeout: for a TCP server, timeout is 1~28800 seconds, for a inactive client to disconnected.

Returns

net.server sub module

Example

```
    net.createServer(net.TCP, 30) -- 30s timeout
```

See also

- net.createConnection()
- Back to Index

net.createConnection()

Description

create a client.

Syntax

net.createConnection(type, secure)

Parameters

type: net.TCP or net.UDP
secure: 1 or 0, 1 for ssl link, 0 for normal link

Returns

net.server sub module

Example

```
    net.createConnection(net.UDP, 0)
```

See also

- net.createServer()
- Back to Index

net.server module

listen()

Description

listen on port from [ip] address.

Syntax

net.server.listen(port,[ip],function(net.socket))

Parameters

port: port number
ip:ip address string, can be omitted
function(net.socket): callback function, pass to Caller function as param if a connection is created successfully

Returns

nil

Example

```
    -- create a server
    sv=net.createServer(net.TCP, 30)    -- 30s time out for a inactive client
    -- server listen on 80, if data received, print data to console, and send "hello world" to remote.
    sv:listen(80,function(c)
      c:on("receive", function(c, pl) print(pl) end)
      c:send("hello world")
      end)
```

See also

- net.createServer()
- Back to Index

close()

Description

close server.

Syntax

net.server.close()

Parameters

nil

Returns

nil

Example

```
    -- create a server
    sv=net.createServer(net.TCP, 30)
    -- close server
    sv:close()
```

See also

- net.createServer()

- Back to Index

net.socket module

connect()

Description

connect to remote.

Syntax

connect(port, ip/domain)

Parameters

port: port number
ip: ip address or domain name in string

Returns

nil

See also

- net.socket:on()
- Back to Index

send()

Description

send data to remote via connection.

Syntax

send(string, function(sent))

Parameters

string: data in string which will be sent to remote
function(sent): callback function for sending string

Returns

nil

See also

- net.socket:on()
- Back to Index

on()

Description

register callback function for event.

Syntax

on(event, function cb())

Parameters

event: string, which can be: "connection", "reconnection", "disconnection", "receive", "sent"
function cb(net.socket, [string]): callback function. The first param is the socket.
If event is"receive", the second param is received data in string.

Returns

nil

Example

```
    sk=net.createConnection(net.TCP, 0)
    sk:on("receive", function(sck, c) print(c) end )
    sk:connect(80,"192.168.0.66")
    sk:send("GET / HTTP/1.1\r\nHost: 192.168.0.66\r\nConnection: keep-alive\r\nAccept: */*\r\n\r\n")
```

See also

- net.createServer()
- Back to Index

close()

Description

close socket.

Syntax

close()

Parameters

nil

Returns

nil

See also

- net.createServer()
- Back to Index

dns()

Description

get domain ip

Syntax

dns(domain, function(net.socket, ip))

Parameters

domain: domain name.
function (net.socket, ip): callback function. The first param is the socket, the second param is the ip address in string.

Returns

nil

Example

```
sk=net.createConnection(net.TCP, 0)
sk:dns("www.nodemcu.com",function(conn,ip) print(ip) end)
sk = nil
```

See also

- net.createServer()
- Back to Index

i2c module

CONSTANT

i2c.SLOW, i2c.TRANSMITTER, i2c. RECEIVER. FAST（400k）is not supported for now.

i2c.setup()

Description

initialize i2c.

Syntax

i2c.setup(id, pinSDA, pinSCL, speed)

Parameters

id = 0
pinSDA: 1~12, IO index

pinSCL: 1~12, IO index
speed: i2c.SLOW

Returns

speed: the seted speed.

See also

- i2c.read()
- Back to Index

i2c.start()

Description

start i2c transporting.

Syntax

i2c.start(id)

Parameters

id = 0

Returns

nil

See also

- i2c.read()
- Back to Index

i2c.stop()

Description

stop i2c transporting.

Syntax

i2c.stop(id)

Parameters

id = 0

Returns

nil

See also

- i2c.read()

- Back to Index

i2c.address()

Description

setup i2c address and read/write mode.

Syntax

i2c.address(id, device_addr, direction)

Parameters

id=0
device_addr: device address.
direction: i2c.TRANSMITTER for writing mode , i2c. RECEIVER for reading mode

Returns

true: get ack false: no ack get

See also

- i2c.read()
- Back to Index

i2c.write()

Description

write data to i2c, data can be multi numbers, string or lua table.

Syntax

i2c.write(id, data1, data2,...)

Parameters

id=0
data: data can be numbers, string or lua table.

Returns

number: number of bytes wrote.

Example

```
    i2c.write(0, "hello", "world")
```

See also

- i2c.read()
- Back to Index

i2c.read()

Description

read data for len bytes.

Syntax

i2c.read(id, len)

Parameters

id=0
len: data length

Returns

string:data received.

Example

```
id=0
sda=1
scl=2

-- initialize i2c, set pin1 as sda, set pin2 as scl
i2c.setup(id,sda,scl,i2c.SLOW)

-- user defined function: read from reg_addr content of dev_addr
function read_reg(dev_addr, reg_addr)
  i2c.start(id)
  i2c.address(id, dev_addr ,i2c.TRANSMITTER)
  i2c.write(id,reg_addr)
  i2c.stop(id)
  i2c.start(id)
  i2c.address(id, dev_addr,i2c.RECEIVER)
  c=i2c.read(id,1)
  i2c.stop(id)
  return c
end

-- get content of register 0xAA of device 0x77
reg = read_reg(0x77, 0xAA)
print(string.byte(reg))
```

See also

- i2c.write()
- Back to Index

adc module

CONSTANT

none

adc.read()

Description

read adc value of id, esp8266 has only one 10bit adc, id=0, pin TOUT

Syntax

adc.read(id)

Parameters

id = 0

Returns

adc value

See also

-

- Back to Index

uart module

CONSTANT

none

uart.setup()

Description

setup uart's baud, databits, parity, stopbits, echo.

Syntax

uart.setup(id, baud, databits, parity, stopbits, echo)

Parameters

id = 0, only 1 uart supported.
baud = 9600, 19200, 38400, 57600, 74880, 115200, 230400, 460800, 921600. not tested more than 115200
databits = 5, 6, 7, 8.
parity = 0(none).

stopbits = 1(1 stopbit), 2(2 stopbit).
echo = 0(close echo back).

Returns

baud.

See also

-
- Back to Index

uart.on()

Description

set the callback function to the uart event,
"data" event supported, means there is data input from uart.

Syntax

uart.on(method, [number/end_char], [function], [run_input])

Parameters

method = "data", there is data input from uart.
number/end_char: if pass in a number n if n=0, will receive every char in buffer.
if pass in a one char string "c", the callback will called when "c" is encounterd, or max n=255 received.
function: callback function, event "data" has a callback like this: function(data) end
run_input: 0 or 1, 0: input from uart will not go into lua interpreter, can accept binary data.
1: input from uart will go into lua interpreter, and run.

Returns

nil

Example

```
-- when 4 chars is received.
uart.on("data", 4,
  function(data)
    print("receive from uart:", data)
    if data=="quit" then
      uart.on("data")
    end
end, 0)
-- when '\r' is received.
uart.on("data", "\r",
  function(data)
    print("receive from uart:", data)
```

```
        if data=="quit\r" then
          uart.on("data")
        end
    end, 0)
```

See also

-

- Back to Index

uart.write()

Description

write string to uart.

Syntax

uart.write(id, string1, string2...)

Parameters

id = 0, only 1 uart supported.
string1..n: string write to uart.

Returns

nil

See also

-

- Back to Index

onewire module

CONSTANT

none

ow.setup()

Description

set a pin in onewire mode.

Syntax

ow.setup(pin)

Parameters

pin: 1~12, IO index

Returns

nil

See also

-

- Back to Index

ow.reset()

Description

Perform a 1-Wire reset cycle.

Syntax

ow.reset(pin)

Parameters

pin: 1~12, IO index

Returns

number: Returns 1 if a device responds with a presence pulse. Returns 0 if there is no device or the bus is shorted or otherwise held low for more than 250uS

See also

-

- Back to Index

ow.skip()

Description

Issue a 1-Wire rom skip command, to address all on bus.

Syntax

ow.skip(pin)

Parameters

pin: 1~12, IO index

Returns

nil

See also

-

- Back to Index

ow.select()

Description

Issue a 1-Wire rom select command, make sure you do the ow.reset(pin) first.

Syntax

ow.select(pin, rom)

Parameters

pin: 1~12, IO index
rom: string value, len 8, rom code of the salve device

Returns

nil

Example

```
-- 18b20 Example
pin = 9
ow.setup(pin)
count = 0
repeat
  count = count + 1
  addr = ow.reset_search(pin)
  addr = ow.search(pin)
  tmr.wdclr()
until((addr ~= nil) or (count > 100))
if (addr == nil) then
  print("No more addresses.")
else
  print(addr:byte(1,8))
  crc = ow.crc8(string.sub(addr,1,7))
  if (crc == addr:byte(8)) then
    if ((addr:byte(1) == 0x10) or (addr:byte(1) == 0x28)) then
      print("Device is a DS18S20 family device.")
        repeat
          ow.reset(pin)
          ow.select(pin, addr)
          ow.write(pin, 0x44, 1)
          tmr.delay(1000000)
          present = ow.reset(pin)
          ow.select(pin, addr)
          ow.write(pin,0xBE,1)
          print("P="..present)
```

```
        data = nil
        data = string.char(ow.read(pin))
        for i = 1, 8 do
          data = data .. string.char(ow.read(pin))
        end
        print(data:byte(1,9))
        crc = ow.crc8(string.sub(data,1,8))
        print("CRC="..crc)
        if (crc == data:byte(9)) then
           t = (data:byte(1) + data:byte(2) * 256) * 625
           t1 = t / 10000
           t2 = t % 10000
           print("Temperature="..t1.."."..t2.."Centigrade")
        end
        tmr.wdclr()
      until false
    else
      print("Device family is not recognized.")
    end
  else
    print("CRC is not valid!")
  end
end
```

See also

-
- Back to Index

ow.write()

Description

Write a byte. If 'power' is 1 then the wire is held high at the end for parasitically powered devices. You are responsible for eventually depowering it by calling depower() or doing another read or write.

Syntax

ow.write(pin, v, power)

Parameters

pin: 1~12, IO index
v: byte to be written to salve device
power: 1 for wire being held high for parasitically powered devices.

Returns

nil

Example

See also

-
- Back to Index

ow.write_bytes()

Description

Write multi bytes. If 'power' is 1 then the wire is held high at the end for parasitically powered devices. You are responsible for eventually depowering it by calling depower() or doing another read or write.

Syntax

ow.write_bytes(pin, buf, power)

Parameters

pin: 1~12, IO index
buf: string to be written to salve device
power: 1 for wire being held high for parasitically powered devices.

Returns

nil

Example

See also

-
- Back to Index

ow.read()

Description

read a byte.

Syntax

ow.read(pin)

Parameters

pin: 1~12, IO index

Returns

byte read from slave device.

Example

See also

-
- Back to Index

ow.read_bytes()

Description

read multi bytes.

Syntax

ow.read_bytes(pin, size)

Parameters

pin: 1~12, IO index
size: number of bytes to be read from slave device.

Returns

string: bytes read from slave device.

Example

See also

-
- Back to Index

ow.depower()

Description

Stop forcing power onto the bus. You only need to do this if you used the 'power' flag to ow.write() or used a ow.write_bytes() and aren't about to do another read or write.

Syntax

ow.depower(pin)

Parameters

pin: 1~12, IO index

Example

Returns

nil

See also

-
- Back to Index

ow.reset_search()

Description

Clear the search state so that it will start from the beginning again.

Syntax

ow.reset_search(pin)

Parameters

pin: 1~12, IO index

Returns

nil

Example

See also

-
- Back to Index

ow.target_search()

Description

Setup the search to find the device type 'family_code' on the next call to ow.search() if it is present.

Syntax

ow.target_search(pin, family_code)

Parameters

pin: 1~12, IO index
family_code: byte for family code.

Returns

nil

Example

See also

-
- Back to Index

ow.search()

Description

Look for the next device.

Syntax

ow.search(pin)

Parameters

pin: 1~12, IO index

Returns

if succeed return a string length of 8, which contain the rom code of slave device.
if failed in searching next device return nil.

Example

See also

-
- Back to Index

ow.crc8()

Description

Compute a Dallas Semiconductor 8 bit CRC, these are used in the ROM and scratchpad registers.

Syntax

ow.crc8(buf)

Parameters

buf: string value, data to be calculated check sum in string.

Returns

crc result in byte.

Example

See also

-
- Back to Index

ow.check_crc16()

Description

Compute the 1-Wire CRC16 and compare it against the received CRC.

Syntax

ow.check_crc16(buf, inverted_crc0, inverted_crc1, crc)

Parameters

buf: string value, data to be calculated check sum in string.
inverted_crc0: LSB of received CRC.
inverted_crc1: MSB of received CRC.
crc: crc starting value (optional)

Returns

bool: true, if the CRC matches; false for dismatches.

Example

See also

-
- Back to Index

ow.crc16()

Description

Compute a Dallas Semiconductor 16 bit CRC. This is required to check the integrity of data received from many 1-Wire devices. Note that the CRC computed here is **not** what you'll get from the 1-Wire network, for two reasons:
1) The CRC is transmitted bitwise inverted.
2) Depending on the endian-ness of your processor, the binary representation of the two-byte return value may have a different byte order than the two bytes you get from 1-Wire.

Syntax

ow.crc16(buf, crc)

Parameters

buf: string value, data to be calculated check sum in string.
crc: crc starting value (optional)

Returns

return The CRC16, as defined by Dallas Semiconductor.

Example

See also

-
- Back to Index

bit module

CONSTANT

none

bit.bnot()

Description

Bitwise negation, equivalent to ~value in C.

Syntax

bit.bnot(value)

Parameters

value: the number to negate.

Returns

number: the bitwise negated value of the number.

Example

See also

-
- Back to Index

bit.band()

Description

Bitwise AND, equivalent to val1 & val2 & ... & valn in C.

Syntax

bit.band(val1, val2, ... valn)

Parameters

val1: first AND argument.
val2: second AND argument.
valn: nth AND argument.

Returns

number: the bitwise AND of all the arguments.

Example

See also

-
- Back to Index

bit.bor()

Description

Bitwise OR, equivalent to val1 | val2 | ... | valn in C.

Syntax

bit.bor(val1, val2, ... valn)

Parameters

val1: first OR argument.
val2: second OR argument.
valn: nth OR argument.

Returns

number: the bitwise OR of all the arguments.

Example

See also

-
- Back to Index

bit.bxor()

Description

Bitwise XOR, equivalent to val1 ^ val2 ^ ... ^ valn in C.

Syntax

bit.bxor(val1, val2, ... valn)

Parameters

val1: first XOR argument.
val2: second XOR argument.
valn: nth XOR argument.

Returns

number: the bitwise XOR of all the arguments.

Example

See also

-
- Back to Index

bit.lshift()

Description

Left-shift a number, equivalent to value << shift in C.

Syntax

bit.lshift(value, shift)

Parameters

value: the value to shift.
shift: positions to shift.

Returns

number: the number shifted left

Example

See also

-
- Back to Index

bit.rshift()

Description

Logical right shift a number, equivalent to (unsigned)value >> shift in C.

Syntax

bit.rshift(value, shift)

Parameters

value: the value to shift.
shift: positions to shift.

Returns

number: the number shifted right (logically).

Example

See also

-
- Back to Index

bit.arshift()

Description

Arithmetic right shift a number equivalent to value >> shift in C.

Syntax

bit.arshift(value, shift)

Parameters

value: the value to shift.
shift: positions to shift.

Returns

number: the number shifted right (arithmetically).

Example

See also

-
- Back to Index

bit.bit()

Description

Generate a number with a 1 bit (used for mask generation). Equivalent to 1 << position in C.

Syntax

bit.bit(position)

Parameters

position: position of the bit that will be set to 1.

Returns

number: a number with only one 1 bit at position (the rest are set to 0).

Example

See also

-
- Back to Index

bit.set()

Description

Set bits in a number.

Syntax

bit.set(value, pos1, pos2, ..., posn)

Parameters

value: the base number.
pos1: position of the first bit to set.
pos2: position of the second bit to set.
posn: position of the nth bit to set.

Returns

number: the number with the bit(s) set in the given position(s).

Example

See also

-
- Back to Index

bit.clear()

Description

Clear bits in a number.

Syntax

bit.clear(value, pos1, pos2, ..., posn)

Parameters

value: the base number.
pos1: position of the first bit to clear.

pos2: position of the second bit to clear.
posn: position of thet nth bit to clear.

Returns

number: the number with the bit(s) cleared in the given position(s).

Example

See also

-
- Back to Index

bit.isset()

Description

Test if a given bit is set.

Syntax

bit.isset(value, position)

Parameters

value: the value to test.
position: bit position to test.

Returns

boolean: true if the bit at the given position is 1, false otherwise.

Example

See also

-
- Back to Index

bit.isclear()

Description

Test if a given bit is cleared.

Syntax

bit.isclear(value, position)

Parameters

value: the value to test.
position: bit position to test.

Returns

boolean: true if the bit at the given position is 0, false othewise.

Example

See also

-
- Back to Index

spi module

CONSTANT

MASTER, SLAVE, CPHA_LOW, CPHA_HIGH, CPOL_LOW, CPOL_HIGH, DATABITS_8, DATABITS_16

spi.setup()

Description

setup spi configuration.

Syntax

spi.setup(id, mode, cpol, cpha, databits, clock)

Parameters

id: spi id number.
mode: MASTER or SLAVE(not supported yet).
cpol: CPOL_LOW or CPOL_HIGH, clock polarity.
cpha: CPHA_HIGH or CPHA_LOW, clock phase.
databits: DATABITS_8 or DATABITS_16.
clock: spi clock (not supported yet).

Returns

number: 1.

Example

See also

-
- Back to Index

spi.send()

Description

send data to spi.

Syntax

wrote = spi.send(id, data1, [data2], ..., [datan])

Parameters

id: spi id number.
data: data can be either a string, a table or an 8-bit number

Returns

number: bytes writen count.

Example

See also

-
- Back to Index

spi.recv()

Description

recv data from spi.

Syntax

read = spi.recv(id, size)

Parameters

id: spi id number.
size: data size want to read.

Returns

data: string bytes read from spi.

Example

See also

-
- Back to Index

mqtt module

CONSTANT

mqtt.Client()

Description

create a mqtt client.

Syntax

mqtt.Client(clientid, keepalive, user, pass)

Parameters

clientid: the client id.
keepalive: keepalive second, a number.
user: user name, a string.
pass: user password, a string.

Returns

mqtt client.

Example

```
-- init mqtt client with keepalive timer 120sec
m = mqtt.Client("clientid", 120, "user", "password")

-- setup Last Will and Testament (optional)
-- Broker will publish a message with qos = 0, retain = 0, data = "offline"
-- to topic "/lwt" if client don't send keepalive packet
m:lwt("/lwt", "offline", 0, 0)

m:on("connect", function(con) print ("connected") end)
m:on("offline", function(con) print ("offline") end)

-- on publish message receive event
m:on("message", function(conn, topic, data)
  print(topic .. ":" )
  if data ~= nil then
    print(data)
  end
end)

-- for secure: m:connect("192.168.11.118", 1880, 1)
m:connect("192.168.11.118", 1880, 0, function(conn) print("connected") end)

-- subscribe topic with qos = 0
m:subscribe("/topic",0, function(conn) print("subscribe success") end)

-- publish a message with data = hello, QoS = 0, retain = 0
m:publish("/topic","hello",0,0, function(conn) print("sent") end)
```

```
m:close();
-- you can call m:connect again
```

See also

-
- Back to Index

mqtt client module

mqtt:lwt()

Description

setup Last Will and Testament (optional)
Broker will publish a message with qos = 0, retain = 0, data = "offline"
to topic "/lwt" if client don't send keepalive packet.

Syntax

mqtt:lwt(topic, message, qos, retain)

Parameters

topic: the topic to publish to, String.
message: the message to publish, Buffer or String.
qos: qos level, default 0.
retain: retain flag, default 0.

Returns

nil.

Example

See also

-
- Back to Index

mqtt:connect()

Description

Connects to the broker specified by the given host, port, and secure options

Syntax

mqtt:connect(host, port, secure, function(client))

Parameters

host: host domain or ip, string.
port: number, broker port.
secure: 0 or 1, default 0.
function(client): when connected, call this function.

Returns

nil.

Example

See also

-
- Back to Index

mqtt:close()

Description

Connects to the broker specified by the given host, port, and secure options

Syntax

mqtt:close()

Parameters

nil

Returns

nil.

Example

See also

-
- Back to Index

mqtt:publish()

Description

Publish a message

Syntax

mqtt:publish(topic, payload, qos, retain, function(client))

Parameters

topic: the topic to publish to, string
message: the message to publish, string
qos: qos level, default 0
retain: retain flag, default 0
function(client): callback fired when PUBACK received.

Returns

nil.

Example

See also

-
- Back to Index

mqtt:subscribe()

Description

Subscribe to a topic or topics

Syntax

mqtt:subscribe(topic, qos, function(client, topic, message))

Parameters

topic: a string topic to subscribe to
qos: qos subscription level, default 0
function(client, topic, message): callback fired when message received.

Returns

nil.

Example

See also

-
- Back to Index

mqtt:on()

Description

register callback function to event.

Syntax

mqtt:on(event, function(client, [topic], [message]))

Parameters

event: string, which can be: "connect", "message", "offline"
function cb(client, [topic], [message]): callback function. The first param is the client.
If event is "message", the 2nd and 3rd param are received topic and message in string.

Returns

nil.

Example

See also

-
- Back to Index