Pin-out ESP32 WROOM



Pinout of ERC32 (not the module pinout)

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **No.** | **Type** | **Function** |
| GND | 1 | P | Ground |
| 3V3 | 2 | P | Power supply |
| EN | 3 | I | Module-enable signal. Active high. |
| SENSOR\_VP | 4 | I | GPIO36, ADC1\_CH0, RTC\_GPIO0 |
| SENSOR\_VN | 5 | I | GPIO39, ADC1\_CH3, RTC\_GPIO3 |
| IO34 | 6 | I | GPIO34, ADC1\_CH6, RTC\_GPIO4 |
| IO35 | 7 | I | GPIO35, ADC1\_CH7, RTC\_GPIO5 |
| IO32 | 8 | I/O | GPIO32, XTAL\_32K\_P (32.768 kHz crystal oscillator input), ADC1\_CH4, TOUCH9, RTC\_GPIO9 |
| IO33 | 9 | I/O | GPIO33, XTAL\_32K\_N (32.768 kHz crystal oscillator output), ADC1\_CH5, TOUCH8, RTC\_GPIO8 |
| IO25 | 10 | I/O | GPIO25, DAC\_1, ADC2\_CH8, RTC\_GPIO6, EMAC\_RXD0 |
| IO26 | 11 | I/O | GPIO26, DAC\_2, ADC2\_CH9, RTC\_GPIO7, EMAC\_RXD1 |
| IO27 | 12 | I/O | GPIO27, ADC2\_CH7, TOUCH7, RTC\_GPIO17, EMAC\_RX\_DV |
| IO14 | 13 | I/O | GPIO14, ADC2\_CH6, TOUCH6, RTC\_GPIO16, MTMS, HSPICLK, HS2\_CLK, SD\_CLK, EMAC\_TXD2 |
| IO12 | 14 | I/O | GPIO12, ADC2\_CH5, TOUCH5, RTC\_GPIO15, MTDI, HSPIQ, HS2\_DATA2, SD\_DATA2, EMAC\_TXD3 |
| GND | 15 | P | Ground |
| IO13 | 16 | I/O | GPIO13, ADC2\_CH4, TOUCH4, RTC\_GPIO14, MTCK, HSPID, HS2\_DATA3, SD\_DATA3, EMAC\_RX\_ER |
| SHD/SD2\* | 17 | I/O | GPIO9, SD\_DATA2, SPIHD, HS1\_DATA2, U1RXD |
| SWP/SD3\* | 18 | I/O | GPIO10, SD\_DATA3, SPIWP, HS1\_DATA3, U1TXD |
| SCS/CMD\* | 19 | I/O | GPIO11, SD\_CMD, SPICS0, HS1\_CMD, U1RTS |
| SCK/CLK\* | 20 | I/O | GPIO6, SD\_CLK, SPICLK, HS1\_CLK, U1CTS |
| SDO/SD0\* | 21 | I/O | GPIO7, SD\_DATA0, SPIQ, HS1\_DATA0, U2RTS |
| SDI/SD1\* | 22 | I/O | GPIO8, SD\_DATA1, SPID, HS1\_DATA1, U2CTS |
| IO15 | 23 | I/O | GPIO15, ADC2\_CH3, TOUCH3, MTDO, HSPICS0, RTC\_GPIO13, HS2\_CMD, SD\_CMD, EMAC\_RXD3 |
| IO2 | 24 | I/O | GPIO2, ADC2\_CH2, TOUCH2, RTC\_GPIO12, HSPIWP, HS2\_DATA0, SD\_DATA0 |
| IO0 | 25 | I/O | GPIO0, ADC2\_CH1, TOUCH1, RTC\_GPIO11, CLK\_OUT1, EMAC\_TX\_CLK |
| IO4 | 26 | I/O | GPIO4, ADC2\_CH0, TOUCH0, RTC\_GPIO10, HSPIHD, HS2\_DATA1, SD\_DATA1, EMAC\_TX\_ER |
| IO16 | 27 | I/O | GPIO16, HS1\_DATA4, U2RXD, EMAC\_CLK\_OUT |
| IO17 | 28 | I/O | GPIO17, HS1\_DATA5, U2TXD, EMAC\_CLK\_OUT\_180 |
| IO5 | 29 | I/O | GPIO5, VSPICS0, HS1\_DATA6, EMAC\_RX\_CLK |
| IO18 | 30 | I/O | GPIO18, VSPICLK, HS1\_DATA7 |
| IO19 | 31 | I/O | GPIO19, VSPIQ, U0CTS, EMAC\_TXD0 |
| NC | 32 | - | - |
| IO21 | 33 | I/O | GPIO21, VSPIHD, EMAC\_TX\_EN |
| RXD0 | 34 | I/O | GPIO3, U0RXD, CLK\_OUT2 |
| TXD0 | 35 | I/O | GPIO1, U0TXD, CLK\_OUT3, EMAC\_RXD2 |
| IO22 | 36 | I/O | GPIO22, VSPIWP, U0RTS, EMAC\_TXD1 |
| IO23 | 37 | I/O | GPIO23, VSPID, HS1\_STROBE |
| GND | 38 | P | Ground |

Choix pin I/O

|  |  |  |
| --- | --- | --- |
| **Fonction HW** | **PIN** | **Rational** |
| Pump relay command | GPIO12 | High level at boot |
| Temp sensor DS18B20 communication | GPIO14 | Fully available |

Note: pullup resistor 4,7 kohm required on temp. sensor data line.

Test HW interfaces

* Temperature sensor DS18B20 : OK on 30/10/2021
* Relay board : OK on 31/10/2021

State variable list

* Current pump management mode
  + Summer 🡺 filtration time = mean water temp. / 2 / day
  + Mid-season 🡺 filtration time = mean water temp. / 3 / day
  + Winter 🡺 filtration time = 2h / day
* Last DS save time
* Last pump 28 ON/OFF pump event time (covers one week)
* Cumulated pump ON time between oldest and nearest event
* TTS: Temperature threshold between Summer and mid-season automatic switch: default 27°C
* TTW: Temperature threshold between Winter and mid-season automatic switch: default 12°C
* TTHP: Temperature threshold for heat pump activation: default 27.5 °C

Web server information

* Current date and time
* Current water temperature
* Water temperature variation during last hour and during last 24h
* Current pump management mode
* Current pump state: ON or OFF
* Time and type of next pump event: ON or OFF
* Mean pump activation time per day

Web data management

* Change temperature thresholds
* Reset state variables
* Force pump OFF or ON (nominal state will be restored at next scheduled pump event)
* Force pump management mode (nominal state will be restored at next scheduled mode change)

Automatic pump mode management

* Spring mode above TTS temperature
* Winter mode below TTW temperature
* Change mode only when the last 8 temperature measurements satisfy the required criteria (i.e. 2 hours)

Pump management

* Activation time as defined above
* 2 activation windows per day with 2 start times: 5h and 18h
* The morning activation is required for the pool heat pump
* The pump activation is shared within the 2 windows
* At each start time, the pump activation requirement is computed, based on water temperature and the pump is activated
* Each 15 mn, the pump activity time, if ON, is incremented and compared to the current requirement for the on-going window
* The pump is halted if functioning time is reached
* At 18h00, the same process is run excepted the computation
* If mode = "Spring"  
  and  
   if temperature < PAC threshold,  
  and if pump is not activated  
  then pump is activated (for heat pump activation)

PCB attachment

Cadre support en PLA  
Avec pieds de surélévation  
Carte ESP fixée par en dessous et composant dessous  
Carte relais fixée au dessus avec composants dessus.



**Bloc alim 5.2V**

**Connecteur micro USB**

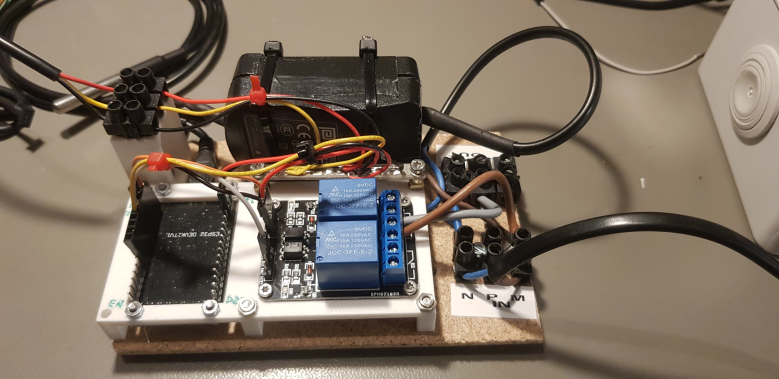
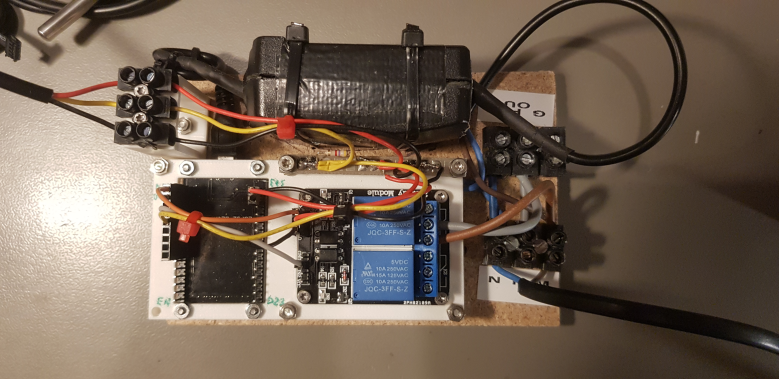
**Antenne WiFi**

**Bornier sonde de température**

**Carte interconnexion**

**Schematic**

Final hardware implementation

Power consumption

Measurement with an external power supply @5.0v 🡺 ≈ 60 mA I.e. 300 mW

**Algorithms**

Mode switch

if (current\_water\_temperature > TTS)

if (trigger\_TS)

Increment TS\_counter and clip to counter\_threshold

trigger\_TS = true

trigger\_TW = false

trigger\_TMS = false

TW\_counter = 0

TMS\_counter = 0

else if (current\_water\_temperature < TTW)

if (trigger\_TW)

increment TW\_counter and clip to counter\_threshold

trigger\_TW = true

trigger\_TS = false

trigger\_TMS = false

TS\_counter = 0

TMS\_counter = 0

else

if (trigger\_TMS)

increment TMS\_counter and clip to counter\_threshold

trigger\_TMS = true

trigger\_TW = false

trigger\_TS = false

TW\_counter = 0

TS\_counter = 0

if (TS\_counter >= counter\_threshold)

activate summer mode

if (TW\_counter >= counter\_threshold)

activate winter mode

if (TMS\_counter >= counter\_threshold)

activate mid season mode

Instruction set for commands through MQTT

TOPIC: SW/CONTROLER/SET

PAYLOAD:

* {"TEMP\_TEST":<float value>} 🡺 sets test temperature in °C, used for test of season modes
* {"TEMP\_PAC":<float value>} 🡺 sets heat pump target temperature in °C, used for pump automation
* {"WINDOW":[h1, m1, h2, m2]} 🡺 sets activation windows beginning time (hour, minutes)

TOPIC: SW/CONTROLER/CMD

PAYLOAD:

* {" TEST": "ON" or "OFF"} 🡺 enable / disable test mode (for season modes tests)
* {" PAC\_AUTO": "ON" or "OFF"} 🡺 enable / disable heat pump automation
* {" SEND": "MICRO"} 🡺 send microprocessor information on "SW/CONTROLER/TECH\_INFO" topic
  + other keywords might be implemented in the future
* {"STATE", ---} 🡺 not yet implemented

Note:

With heat pump automation mode, the filtering pump is activated when the water temperature is below the heat pump target temperature minus 0.5°C. The heat pump will activate automatically because the current temperature is below its target and because water flow is detected. When the water temperature will reach the target plus 1°C, the filtering pump will be stopped (excepted if an activation window is running).

Running time for the heat pump is taking into account for the global day running time.