Trough Automation PCB - Documentation

Here you can find all the necessary information to work with the Trough Automation PCB in your own project.

General Concept and internal workings

The board was developed as an easy tool to switch higher voltage loads (higher than 3.3V) remotely and automatically based on sensor readings done by the same board or even other sensors in the surroundings. It can connect to Wifi and can also be reconfigured without programming the whole board, thanks to Tasmota.

The board's main chip is an ESP8266, more specifically the ESP-12F. It can control a relay and has a current meter connected to the ADC which meters the current attached to the high voltage output. Furthermore, all GPIOs are available for attaching sensors. GPIO 14 is routed out specifically for OneWire Devices.

Power Management

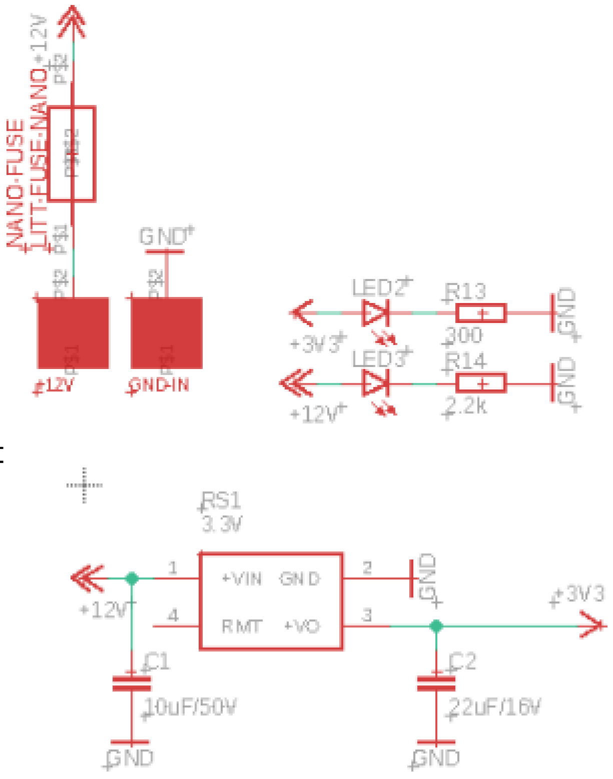


Fig. 1 The power management side is held pretty simple. The main work is done by a daughter board by RS Pro. It has an input voltage of 4.75V to 36V and can deliver a maximum of 500mA. It puts out a constant voltage of 3.3V, used by all the controlling electronics. The remote pin of the module is not used or routed out at the moment. This could be used as an on and off switch. In front of the regulator sits a nano fuse. This can differ from board to board. (So have a close look at that!) There are status LEDs for both 12V and 3.3V. When using another voltage than 12V, R14 may have to be adjusted, so the LED does not blow out. Both should light up clearly when using the board with 12V. When only 3.3V is connected to the programming header, the 3.3V LED will light up normally but the 12V LED will not be fully lit up. Never connect an FTDI-Breakout to the 3.3V-Pin when a high voltage is connected to the board!

MCU

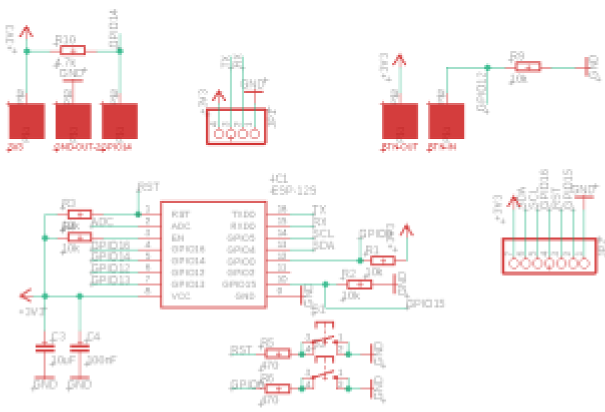
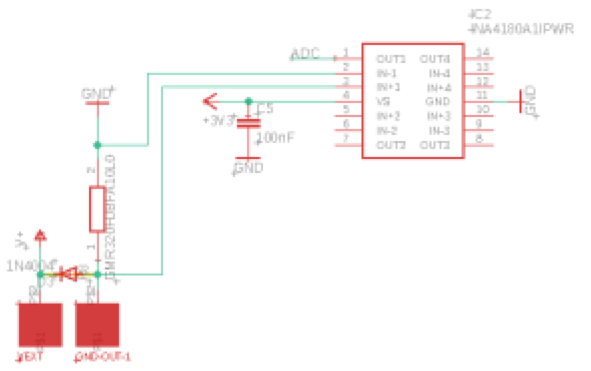


Fig. 2 The ESP8266/ESP-12F is implemented as described in the datasheet: Datasheet ESP12-F (5. Application circuit). Furthermore, some extra parts got added. To program the board it has to be set into the programming mode. Therefore two extra buttons are needed. These are hooked up to the reset pin and GPIO0. Two GPIOs are routed to specific solder pads needed in the application as a trough-automation. GPIO14 is pulled high so OneWire sensors like the DS18B20 can be attached directly to the solder pads. There are also solder pads for a button or a switch of some sort connected to GPIO12. It is normally pulled low. So a button press could be recognized as a rising edge. All other unused GPIOs (except GPIO2) are routed out to a little "development area". In this area, other sensors or daughter boards could be attached. Some examples can be found below in "Other possibilities".

High Voltage Switching Side



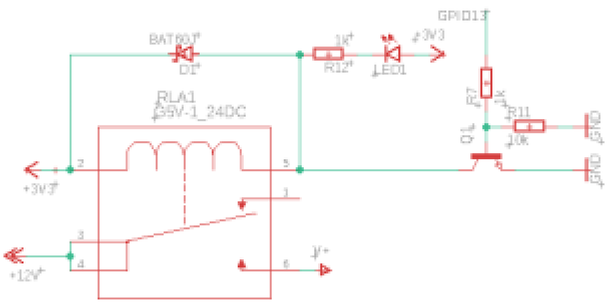


Fig. 3 The high voltage side is directly fed by whatever voltage is applied to the PCB Input. So it is only limited by the voltage accepted by the voltage regulator. The relay can be toggled by toggling GPIO13 on the microcontroller. The relay populated is a normally open one. If the relay is activated and the coil inside is energized, the LED by the relay will light up. This is not an indication that there is definitely a voltage at the high voltage output pad but you at least know the state of the relay. On the low side of the high voltage output, a current sensor by Texas Instruments is attached. The datasheet can be found here: Datasheet. Depending on the load attached, the shunt-resistor has to be selected. More about this topic can be found here: Shunt Resistor for Current Measuring. The output of the current sense amplifier is attached to the analog input pin of the ESP microcontroller. This information can be used to identify the state of the load attached or just to measure the current used over a period of time.

How to use the board

Programming

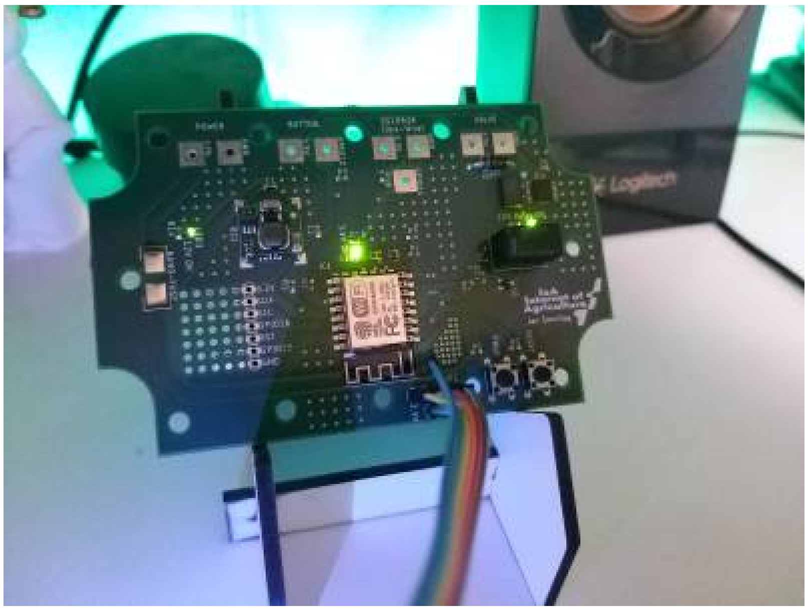


Fig. 4

On the right side of the ESP-Chip are 4 pins located. These are labeled with GND TX RX and VCC. To program the board, you have to use a so-called FTDI or UART programmer. Connect the pins like so:

UART/FTDI	ESP Board
VCC / 3.3V	VCC
GND	GND
TX	RX
RX	TX

You are now able to program the board with the Arduino IDE. The board is intended to work with Tasmota. If you want to learn how to flash Tasmota onto an ESP8266 you can read more here: Flashing Tasmota

Configuration and Pin Assignment

Other possibilities

Future improvements

- C1 should have a bigger footprint, there is no 50V rated cap with the footprint 0603
- Add a voltage divider in front of the ADC, the ADC is limited to a range of 0V to 1V
- There should™ be no copper under the PCB Antenna of the ESP8266
- The 4-channel current meter is suboptimal, to say the least (thanks chip-shortage), a 1-channel would also work
- The dot for the RS PRO Regulator is placed wrong