RPi Hat – GPIO expansion boards (24VDC, 8 inputs + 8 outputs)

Overview

Main motivation behind this design was to design an input/output system for home automation able to control 230VAC equipments without exposing hazardous voltages (as it is usually the case with popular relay expansion boards) as well as support high current requirements. DIN-rail mount relays meet both requirements and can be controlled using a 24V DC voltage and reasonable current levels. The same 24V voltage can also be used for input devices (buttons, switches …) while offering high noise immunity.

This Raspberry HAT design allows to control up to 8x DIN-rail mount relays (24VDC, ~25mA coil current) and sense up to 8x 24VDC inputs. It also integrates a 24V->5V DCDC converter that may be used to power the Raspberry motherboard and avoid the need for an additional USB adapter and associated wall plug in the distribution panel.

In order to offer robust connections and easy system plug-in/out, Phoenix connectors have been used for signal and power interfaces.

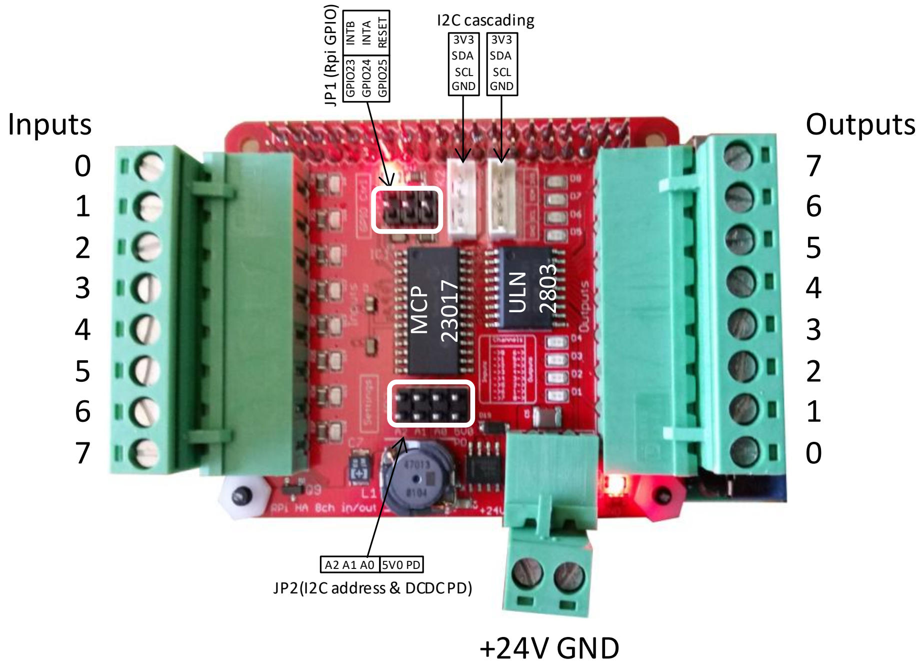


Figure 1: RPi HA 8ch in/out HAT

Description

The design is based on a MCP23017 I2C-based GPIO expander and integrates:

- One ULN2803 high-voltage Darlington driving up to 8 relay coils for the 8 output ports
- Eight 24V -> 3V3 level shifters for the 8 input ports

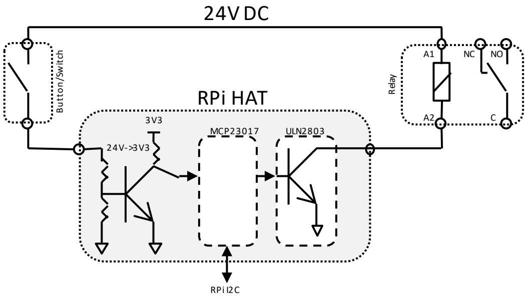


Figure 2: Input and output schematic

The I2C address of the MCP23017 can be configured between 0x20 and 0x27 using 3 jumpers (JP2, no jumpers = 0x27).

Reset pin as well as interrupt A and B of the MCP23017 are also available and can optionally be mapped to Raspberry‘s GPIO 25, 24 and 23 respectively using JP1.

Power supplies

The design is powered directly by the Raspberry’s 3V3 power supply (available from the 2x20 pin header).

The 24V->5V DCDC converter may alternatively be used to power the Raspberry motherboard thru the same pin header avoiding the need for an additional USB adapter. To resolve supply conflicts when both sources are connected, a voltage monitoring circuitry disconnects the 5V DCDC autonomously when power from the motherboard is detected (without supply transients).

This DCDC converter is optional (may be removed from the PCB BOM eventually) and can therefore be disabled by a jumper on JP1 (5V0 PD).

Note that 24VDC and GND signals should always be connected to the external power supply as the freewheeling protection diodes are connected to the 24VDC power supply and relay return currents flow thru GND.

Cascading option (up to 8x8=64 inputs/outputs)

Multiple expansion boards can be cascaded from a single Raspberry by using the two I2C connectors (carrying 3V3/GND power supply on top of I2C SCL/SDA signals). Each slave device has to be configured with a different I2C address obviously and connected to the common 24VDC/GND for the reasons explained above.

Power consumption extracted from the Raspberry 3V3 is minimal (<1mA/device), high currents being all derived from the 24V supply.

Connectors

- Inputs and outputs: Phoenix MSTB2.5/8-ST-5.08
- Power supply (24V/GND): Phoenix MSTB2.5/2ST5.08
- I2C: Molex 51065-0400 (header) + 50212-8000 (wire)
- Raspberry: 2x20 female pin header with long pin (min. 10mm height required for the output Phoenix connector sitting above RJ45/USB, e.g. https://www.adafruit.com/product/1979)

- Spacer: min. 12.7mm (10mm + 2.7mm RPi male pin header)

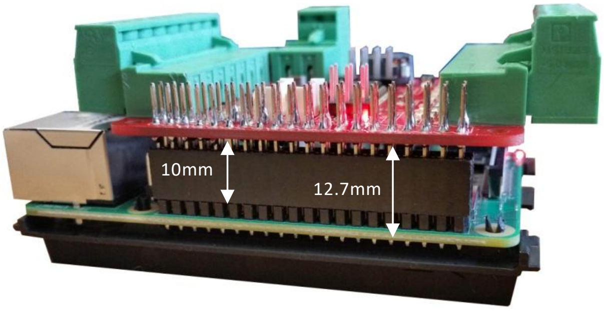
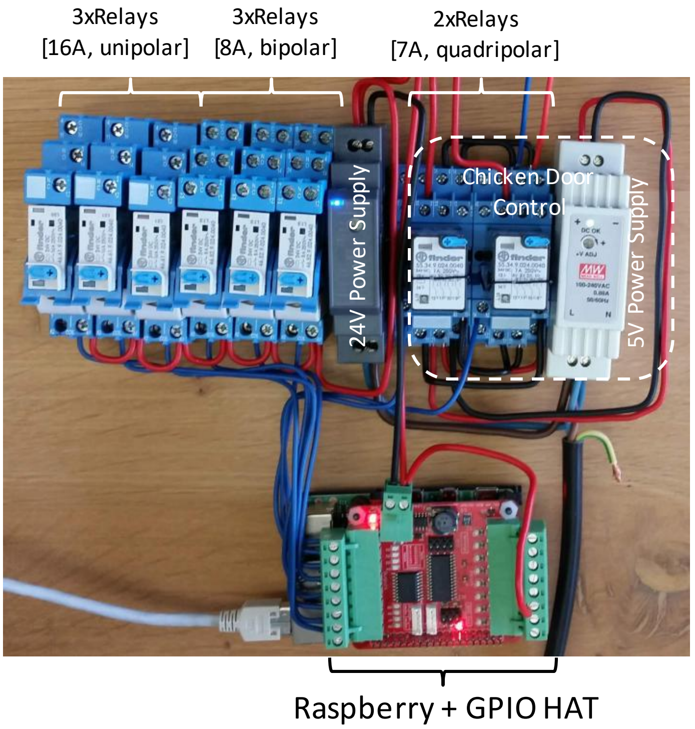


Figure 3: 2x20 female pin header height requirements

Implementation example



DIN modules references:

- Power supply (24V, 15W) :
  - Meanwell HDR-15-24
- Power relays
  - Unipolar 230VAC, 16A:
    - Finder 4C.01.9.024.0050SPA (relay 46.61.9.024.0040 + base 97.01)
  - Bipolar 230VAC, 8A:
    - Finder 4C.02.9.024.0050SPA (relay 46.52.9.024.0040 + base 97.02)

I2C access from Raspberry

Console

Cfr. MPC23017 datasheet (https://www.microchip.com/wwwproducts/en/MCP23017), table 3-5

Check I2C address (0x27 here):

```
$ i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- 27 -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

Configure first 8 pins as output (register IODIRA @ 0x0):

```
$ i2cset -y 1 0x27 0x00 0x00
```

Set GPIOA[0] to 1 (bit 0 of register OLATA @ 0x14)

```
$ i2cset -y 1 0x27 0x14 0x01
```

Read GPIOB inputs (register GPIOB @ 0x13)

```
$ i2cget -y 1 0x27 0x13
0xff
```

Python

Install python-smbus (i2c) package if not done already

```
$ sudo apt-get install python-smbus
```

Same operations as above but using python:

```
$ python
import smbus

bus = smbus.SMBus(1)

bus.write_byte_data(0x27,0x00,0x00)
bus.write_byte_data(0x27,0x14,0x01)
bus.read_byte_data(0x27,0x13)
```

HomeAssistant (https://www.home-assistant.io/) support

configuration.yaml:

```
binary_sensor:
  - platform: mcp23017
    i2c_address: 0x27
    scan_interval: 1
    invert_logic: true
    pins:
      8 : Button_0
      9 : Button_1
      10: Button_2
      11: Button_3
      12: Button_4
      13: Button_5
      14: Button_6
      15: Button_7

switch:
  - platform: mcp23017
    i2c_address: 0x27
    pins:
      0 : Output_0
      1 : Output_1
      2 : Output_2
      3 : Output_3
      4 : Output_4
      5 : Output_5
      6 : Output_6
      7 : Output_7
```