Monarco HAT Hardware Reference Manual

Monarco HAT – Lightweight I/O for the Raspberry Pi minicomputer

https://www.monarco.io

Revision 20180403-1

Contents

Revision History

- **2016/06/19**
  - Initial PRELIMINARY revision.
- **2016/06/23**
  - Ch. 3.2: Fixed P2 connector pinout table.
- **2016/08/12**
  - Ch. 2.2.: Added board layout drawings.
  - Ch. 3.1.: ID EEPROM contents description. Notes for Raspberry Pi 3.
  - Ch. 3.2.: Extended description of digital inputs and outputs technology functions.
- **2016/10/04**
  - Overall minor revisions.
- **2018/04/03**
  - Added “Other Resources” section. Added Ch. 2.3 “Software Integration”. Updated device-tree-overlay info in Ch. 3.1.2. Added I2C addresses in Ch. 4. Formatting fixes.

Other Resources

- **Monarco Homepage**
  - https://www.monarco.io/
- **GitHub Repositories**
  - https://github.com/monarco
  - **Documentation for the Monarco HAT**
    - https://github.com/monarco/monarco-hat-documentation
    - Latest version of Reference manual, SPI protocol specification, Information for Linux integration, Board schematics.
  - **Monarco HAT Firmware Images and Tools**
    - https://github.com/monarco/monarco-hat-firmware-bin
  - **C language driver library for the Monarco HAT**
    - https://github.com/monarco/monarco-hat-driver-c
  - **Node.js driver library for Monarco HAT**
    - https://github.com/monarco/monarco-hat-driver-nodejs
  - **Node-RED driver library for Monarco HAT**
    - https://github.com/monarco/node-red-contrib-monarco-hat

1. Introduction

Monarco HAT is an add-on board which provides input-output interfaces following industrial automation standards for the Raspberry Pi (B+ and newer) minicomputer. It is designed according to the HAT (Hardware Attached on Top) – https://github.com/raspberrypi/hats – specification.

Here is a brief overview which functions the Monarco HAT provides:

- **Power supply:** 10-30 VDC, powers also the Raspberry Pi
- **4 × digital IN**: 3.5-30 VDC, optical isolation, common GND
  - 2 × counter (pulse/DIR) or 2 × encoder (A/B), up to 500 kHz
  - counter values retention during power off
- **4 × digital OUT**: open-drain, max 40 VDC, 1 A per channel continuous
  - all with up to 100 kHz PWM
  - short-circuit protection (continuous)
- **2 × analog IN:** 0-10 V / 0-20 mA, 12-bit
  - electronic switching of voltage/current mode
  - protected against overvoltage and reverse polarity
  - 500 Hz bandwidth, configurable filter
- **2 × analog OUT:** 0-10 V, 0.5 ms settling time, 12-bit
- **1 × RS-485** with ESD protection
- **1 × 1-Wire** bus with ESD protection
- **9 × LED indicator**, by default indicate status of digital inputs and outputs and system status, can be switched to user controlled mode
- **High quality push-in terminals**, detachable connector
- **Battery-backed RTC chip** for keeping the track of time
- **Hardware watchdog** for power-cycling the Raspberry Pi in case of failure

Image 1.1: Monarco HAT board mounted on Raspberry Pi, with detached terminals connector.

2. Monarco HAT Overview

Note: Features denoted by "*" are currently being under development.

Additional documentation and resources can be found on: http://monarco.io

2.1. System Design

Monarco HAT is based around ARM Cortex-M3 microcontroller (MCU) which provides a wide set of embedded peripherals missing on the Raspberry Pi itself. It is PWMs for all digital outputs, versatile counters including quadrature encoder signal decoders, digital-to-analog and analog-to-digital converters and more. ARM MCU can also provide very deterministic IO timing in compare to Raspberry Pi with Linux. We have chosen the EFM32 series MCU from Silicon Labs with 16 kB of RAM, 64 kB of FLASH memory and 32 MHz crystal clock.

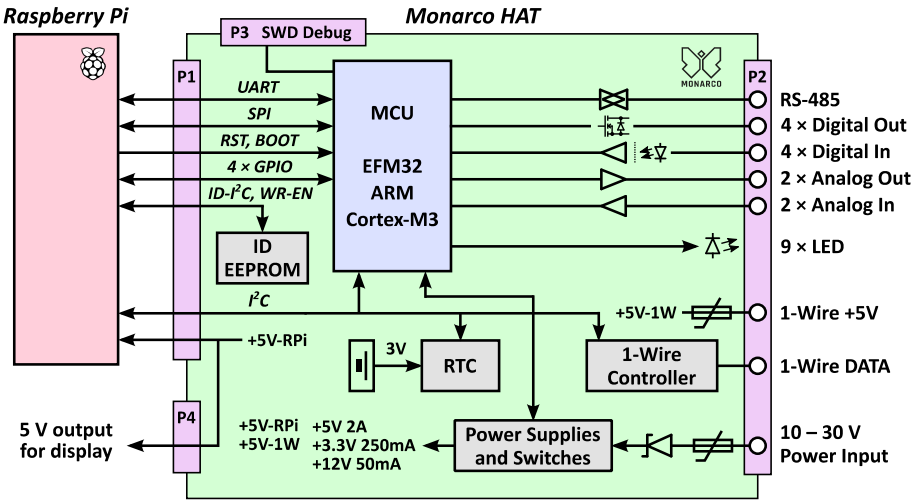

Image 2.1: Monarco HAT system design schematics.

MCU also handles the RS-485 interface. By default, data are forwarded between RS-485 and Raspberry's UART in both directions with correct buffering for half-duplex operation.

The primary communication channel between the MCU and the Raspberry is SPI. The MCU UART can be also used for in-system MCU firmware upgrade by built-in bootloader. For MCU reset and bootloader activation, 2 GPIO signals from the RPi are used. Also 4 GPIO signals from the RPi are routed to the MCU for optional use.

Monarco HAT also contains a Real Time Clock (RTC) backed by a battery (CR1225 type) and 1-Wire bus controller. These are connected by I2C bus directly to the RPi. The MCU has also access to I2C bus for case the Monarco HAT would be used as independent board without Raspberry Pi.

As industrial power supply standard is 24 volts, Monarco HAT has a switching power supply integrated which accepts 10 to 30 volts input. The power supply block generates +12V, +5V and +3.3V rails. RPi (and also an official 7" touchscreen if you have one) is powered from +5V rail which can be switched off by the MCU to reset the RPi (watchdog function is implemented in the MCU). Also the 5V output for 1-Wire bus is switchable.

2.2. Board Layout

Following image shows board layout with connectors on the top side highlighted – P1 (Raspberry Pi header), P2 (input and output signals), P3 (MCU debugger), P4 (display 5 V power output).

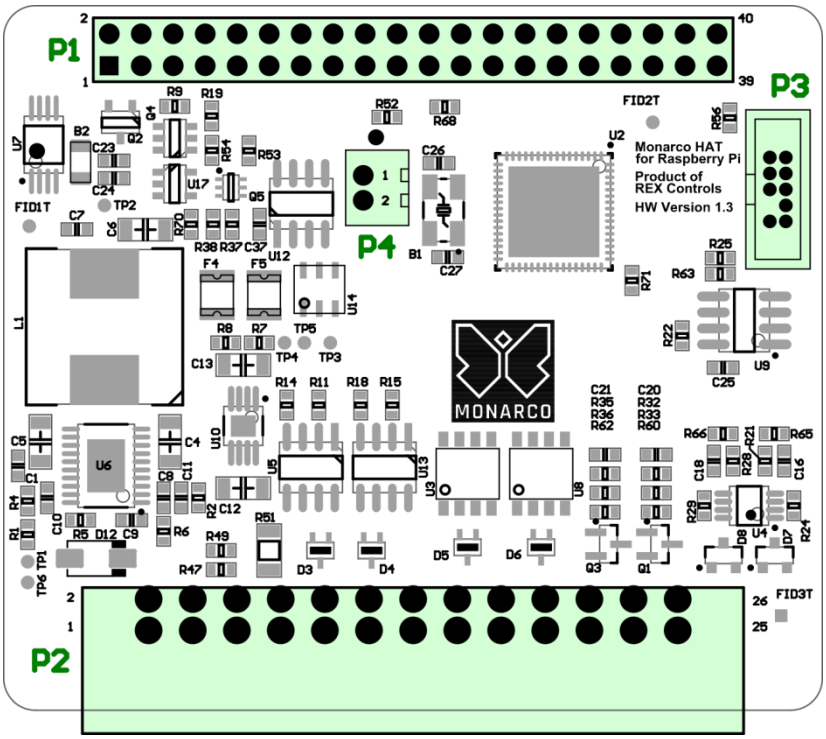

Image 2.2: Monarco HAT board layout – Top side.

On the bottom side, location of battery holder BAT1 and LED0 to LED8 indicators are highlighted.

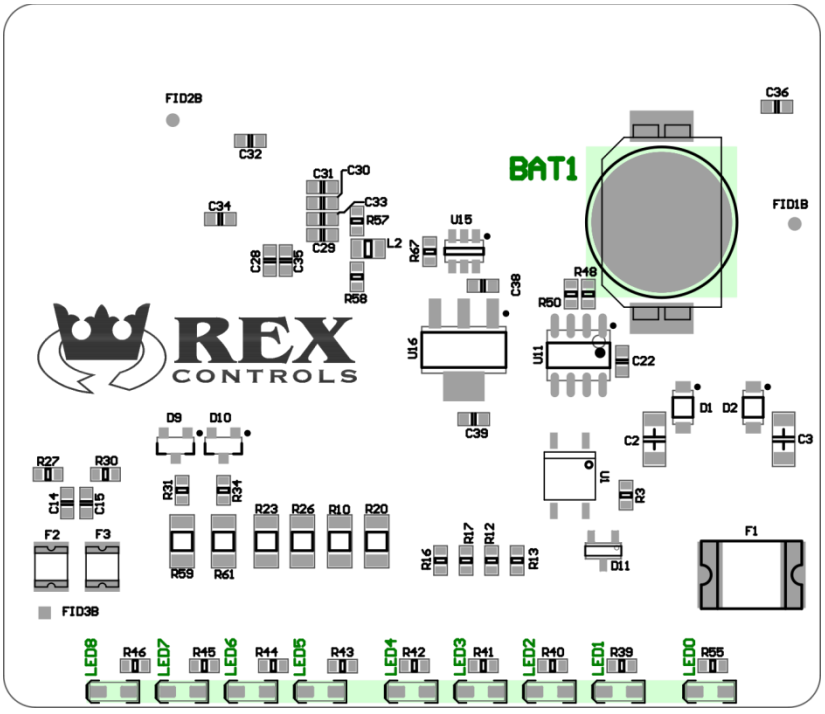

Image 2.3: Monarco HAT board layout – Bottom side.

2.3. Software Integration

The latest Monarco HAT MCU firmware, software tools and instructions for upgrading are available from the GitHub repository:

https://github.com/monarco/monarco-hat-firmware-bin

Monarco HAT software integration is officially supported on the Raspberry Pi[1] (B+ and newer) and the UpBoard[2] (original version mechanically compatible with Raspberry Pi B+) with Raspbian/Debian GNU/Linux operating system. Using Monarco HAT on these platforms should be as hassle-free as possible and we regularly test compatibility with current releases. Using other operating systems or over different boards with compatible pinout should be possible with some effort, but it would need a good understanding of operating system and hardware internals.

More information about embedded ID EEPROM which contains device-tree-overlay for the Raspberry Pi can be found in chapter 3.1.2.

Software libraries for using Monarco HAT with some of the most popular programming languages and platforms are available from the following GitHub repositories:

- **C language driver library for the Monarco HAT**
  - https://github.com/monarco/monarco-hat-driver-c
- **Node.js driver library for Monarco HAT**
  - https://github.com/monarco/monarco-hat-driver-nodejs
- **Node-RED driver library for Monarco HAT**
  - https://github.com/monarco/node-red-contrib-monarco-hat

Communication protocol over SPI bus, which is the primary interface for inputs/outputs connected through Monarco HAT, is fully open and documented. The latest documentation of communication protocol and other resources like schematics and more detailed information about integration with Linux operating system are available from the GitHub repository:

https://github.com/monarco/monarco-hat-documentation

[1] https://www.raspberrypi.org/
[2] http://www.up-board.org/up/

3. Hardware Description and I/O Connection Examples

3.1. Connecting Raspberry Pi (P1)

3.1.1. Interface Specification

Connector type: 2×20 pin 2.54 mm pitch socket on bottom side, long pins are available on the top board side for connection of additional devices to IOs not used by Monarco HAT

P1 – Raspberry Pi 40pin Header (pins not listed below are not used by Monarco HAT)		
Pin no. on header	**RPi Signal**	**Connection / Note**
2, 4	+5V	Power supply to the RPi
3	SDA / GPIO2	I2C-1 data – RTC, 1-Wire
5	SCL / GPIO3	I2C-1 clock – RTC, 1-Wire
8	TXD0 / GPIO14	UART Tx – MCU PE11 / USART0
10	RXD0 / GPIO15	UART Rx – MCU PE10 / USART0
16	GPIO23	Optional use, MCU PB7
18	GPIO24	Optional use, MCU PB8
19	MOSI / GPIO10	SPI0 – MCU PC2 / USART2
21	MISO / GPIO9	SPI0 – MCU PC3 / USART2
23	SCLK / GPIO11	SPI0 – MCU PC4 / USART2
24	CE0# / GPIO8	SPI0 – MCU PC4 / USART2
27	ID_SD / GPIO0	I2C-0 data – ID EEPROM
28	ID_SC / GPIO1	I2C-0 clock – ID EEPROM
29	GPIO5	Optional use, MCU PA4
31	GPIO6	Optional use, MCU PA3
37	GPIO26	ID EEPROM Write Enable (low active)
38	GPIO20	MCU Bootloader Enable (high active)
40	GPIO21	MCU RESETn (low active)
6, 9, 14, 20, 25, 30, 34, 39	GND	Ground

Raspberry Pi IOs in use by Monarco HAT – **do not use these as GPIOs for your applications**:

- *SPI-0 with CE0:* Primary data communication channel with Monarco MCU
- *UART-0:* RS-485 forwarded through MCU (can be disabled) / MCU firmware update
- *I2C-1 (SCL, SDA):* Real Time Clock, 1-Wire controller
- *I2C-0 (ID_SC, ID_SD):* ID EEPROM according to the HAT standard
- *GPIO26:* ID EEPROM write enable
- *GPIO20, GPIO21:* MCU bootloader enable, MCU RESETn – Do not touch!
- *GPIO05, GPIO06, GPIO23, GPIO24:* Optional use with MCU's GPIO (normally these signals can be freely used, MCU keeps them floating)

3.1.2. ID EEPROM

Monarco HAT contains EEPROM with device-tree-overlay which is automatically loaded by Raspberry Pi bootloader. This overlay enables SPI interface and I2C with real time clock (MCP79410) and 1-Wire controller (DS2482-100) device nodes.

Note for Raspberry Pi 3: In default configuration, UART0 (ttyAMA0, PL011 UART) is used for embedded Bluetooth interface on Raspberry Pi 3. Monarco HAT EEPROM overlay automatically remaps UART0 (ttyAMA0) to a standard location (header pins 8, 9) and UART1 (ttyS0, mini-UART) to Bluetooth controller – just like the overlay "pi3-miniuart-bt" provided by Raspbian. Additional configuration may be needed to make Bluetooth working because UART1 has limited functionality[3].

Use Raspbian version released at 2016-05-27 or later with Raspberry Pi 3, as this version introduced `hciuart.service` configured to use `serial1` device, which is aliased to `ttyS0` with Monarco.

The EEPROM contains HAT identification which can be read in `/proc/device-tree/hat/`:

- `vendor: REX Controls`
- `product: Monarco HAT`
- `product_id: 0x0001`
- `product_ver: 0x<dtv>106` (may be changed with future hardware revisions)
- `uuid: fe0f39bf-7c03-4eb6-9a91-df861ae5abcd`

Where **`<dtv>`** is a version of device-tree-overlay structure. In very rare cases internal device-tree format can be incompatibly modified between Linux kernel versions. Version history:

- 0: initial version for Raspbian Linux kernel 4.4, since 06/2016
- 1: updated version for Raspbian Linux kernel 4.9+ which introduced incompatible rename of serial ports related device-tree nodes, since 06/2017

Monarco HAT boards are shipped with version compatible with the latest released Raspbian at that time. After Raspbian upgrade between incompatible versions, additional device-tree-overlay file can be manually applied, or ID EEPROM can be upgraded to the new format using firmware flashing tool. More information can be found in the Monarco HAT Documentation GitHub repository[4].

Note for `owfs/owserver` users: Device tree overlay provided in EEPROM leads to automatic load of Linux kernel `w1` family drivers. If you want to use `owfs/owserver` userspace daemon, we recommend disabling kernel drivers and use `i2c-dev` directly from `owserver` for performance reasons. The most appropriate way to disable Linux kernel drivers and enable `i2c-dev` is to execute:

```
echo "blacklist ds2482" > /etc/modprobe.d/blacklist-ds2482.conf
echo "i2c-dev" > /etc/modules-load.d/i2c-dev.conf
```

[3] Option `core_freq=250` should be set in /boot/config.txt because baudrate of mini-UART is tied to core frequency. Baudrate in `hciuart.service` should be changed to 115200.

[4] https://github.com/monarco/monarco-hat-documentation

3.2. Connecting Input and Output Signals (P2), LED Indicators

3.2.1. P2 Terminal Connector Pinout

Connector type: 2 × 13 wire detachable terminals, push-in system by Phoenix Contact

- *Board side type:* DMC 1,5/13-G1F-3,5-LR P20THR (1787124)
- *Plug side type:* DFMC 1,5/13-ST-3,5-LR (1790593)

P2 – Input and Output Signals			
1	GND IN – Ground / Power in 0 V	**2**	+24V IN – Power in +10 to +30 V
3	RS485− / RS485 B Wire	**4**	RS485+ / RS485 A Wire
5	GND – Ground	**6**	GND – Ground
7	1W DATA – 1-Wire bus Data	**8**	1W 5V OUT – 1-Wire bus Power
9	DOUT1 – Digital Out 1	**10**	DOUT2 – Digital Out 2
11	DOUT3 – Digital Out 3	**12**	DOUT4 – Digital Out 4
13	GND – Ground	**14**	+24V OUT – Power output
15	DIN COM− – Digital In. Common	**16**	+24V OUT – Power output
17	DIN1 – Digital Input 1	**18**	DIN2 – Digital Input 2
19	DIN3 – Digital Input 3	**20**	DIN4 – Digital Input 4
21	AIN1 – Analog Input 1	**22**	AIN2 – Analog Input 2
23	AGND – Analog Ground	**24**	+24V OUT – Power output
25	AOUT1 – Analog Output 1	**26**	AOUT2 – Analog Output 2

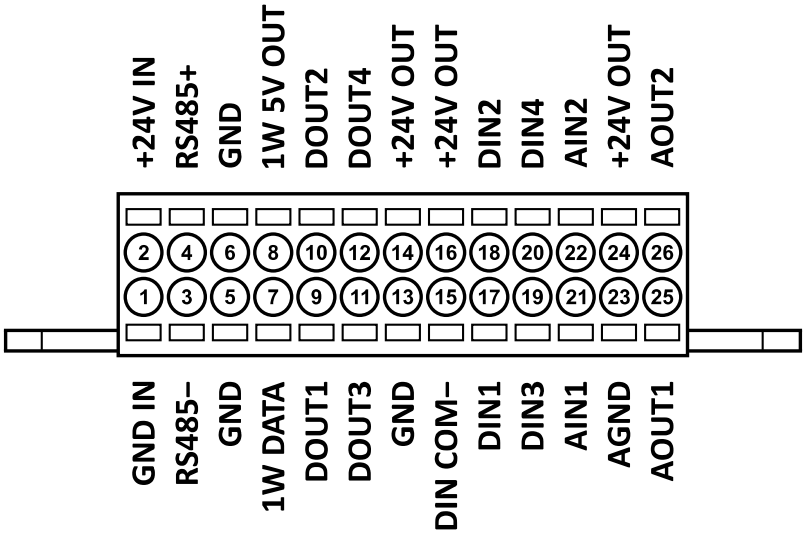

Image 3.1: P2 input-output terminal connector pinout (view from connector terminals side).

3.2.2. LED Indicators

There are 9 LED indicators on board bottom side under the P2 terminal connector. These are divided into three groups with predefined functions:

- LED0 (orange) – Monarco HAT firmware status:
  - Fast blink (5 Hz, 1:1): firmware running, outputs in safe state (no valid process data received during recent watchdog time period).
  - Slow blink (0.5 Hz, 1:1): firmware running, outputs under control by process data.
- LED1 to LED4 (green) – Digital outputs DOUT1 to DOUT4 state indication.
- LED5 to LED8 (green) – Digital inputs DIN1 to DIN4 state indication.

All LEDs are controlled by dedicated microcontroller pins, so they can be controlled by user provided process data instead of the predefined function.

3.2.3. Digital Inputs (DIN)

Description and Electrical Design

Monarco HAT has 4 digital inputs isolated by a high speed optocouplers. Digital inputs have a common ground (minus pole) so they are sink-type inputs.

Voltage levels are designed to be compatible with both 24 V industrial sensors and 5 V standard logic, with tolerance to negative voltage up to 30 volts.

- *Logic low voltage:* −30 V to 1.8 V
- *Logic high voltage:* 3.5 V to 30 V
- *Input current:* 0.8 mA (at 3.5 V) to 9.0 mA (at 30 V)

Here we show a few examples of devices which can be connected to Monarco's digital inputs:

- dry contacts (switches, push buttons, relays),
- proximity sensors with 10 V to 30 V power supply and PNP (source, switching to plus) or NPN (sink, switching to zero) output,
- standard PLC source-type 24 V outputs,
- standard 5 V logic outputs – e.g. 5 V encoders, logic gates,
- open drain outputs with pull-up which is able to supply at least 3.5 V / 0.8 mA.

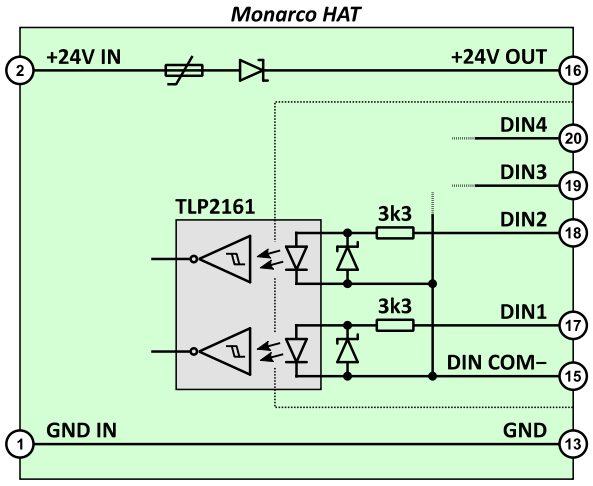

Image 3.2: Digital inputs internal implementation.

Technology Functions Overview

Thanks to an ARM microcontroller embedded in Monarco HAT, digital inputs (and also outputs) can be used also with advanced technology functions, which are not possible with Raspberry Pi integrated GPIOs, such as:

- pulse counter, with optional direction signal,
- quadrature (A/B) encoder counter,
- * measurement of frequency / period / pulse length..

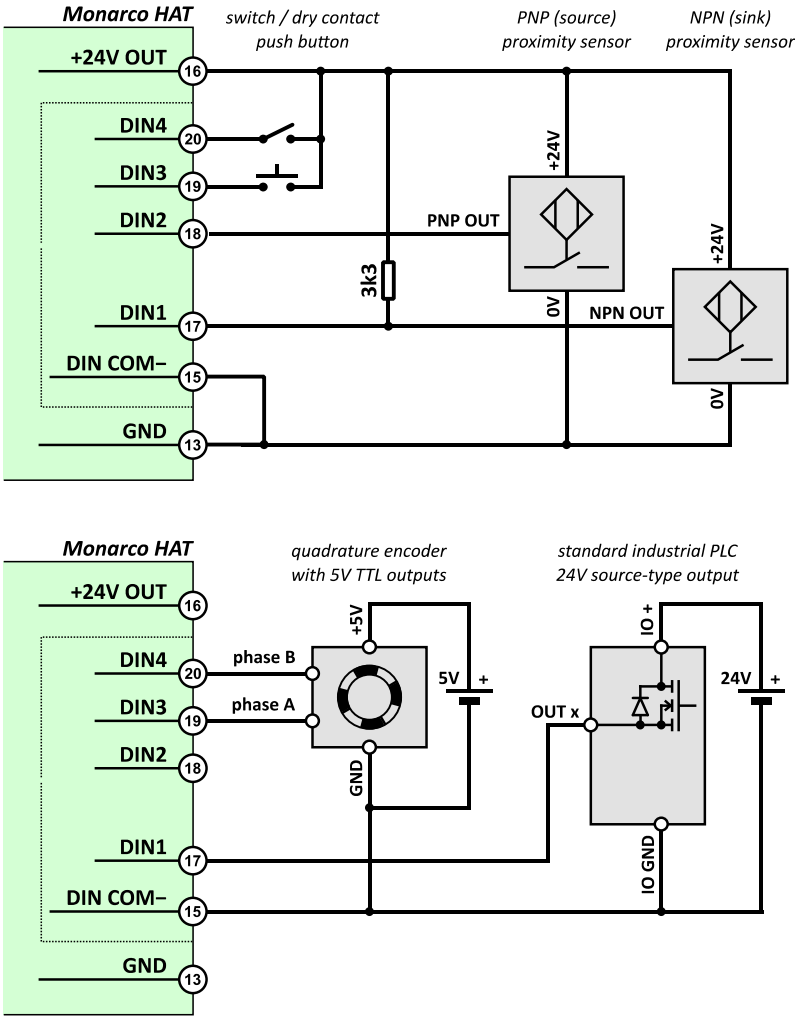

Image 3.3: Digital inputs connection examples with various sensor types.

For these functions, digital inputs are able to handle signals with frequency up to 500 kHz. Moreover, the counters values can be automatically stored to nonvolatile (flash) memory in case of power failure and then loaded back (under development).

Here are a few examples of devices which outputs can be handled by Monarco HAT DIN:

- utility meters (electricity, gas, water) with pulse output,
- standard quadrature encoders for position/velocity measurement,
- gear tooth sensors for position/velocity measurement,
- motor controllers with pulse/direction or quadrature position output,
- various industrial sensors (temperature, pressure, distance) with frequency output.

There is also a configurable input filter to remove short glitches on digital inputs implemented in Monarco's firmware.

COUNTER Technology Function

There are two COUNTER hardware function modules available:

- **COUNTER1 – Input channels: A = DIN1, B = DIN2,**
- **COUNTER2 – Input channels: A = DIN3, B = DIN4, C = DIN2** (shared with COUNTER1).

Each COUNTER module contains 16bit hardware counter which can count up or down. Counter value wrap-around on overflow.

Several function modes are available for each COUNTER module:

- MODE = 0: disabled,
- MODE = 1: pulse counting on input channel A,
- MODE = 2: quadrature encoder on input channels A + B.

In mode 1, additional configuration is available – count direction:

- DIR = 0: up-counting,
- * DIR = 1: direction given by channel B input, low = up-counting, high = down-counting,

and input channel A active edge:

- EDGE = 0: count on rising edge,
- EDGE = 1: count on falling edge,
- EDGE = 2: count on both (rising and falling) edges.

COUNTER2 provides additional *Counter Capture* function which allows you to store instantaneous counter value in auxiliary data register when active edge on input channel C occurred:

- * CAPTURE = 0: capture function disabled,
- * CAPTURE = 1: capture on rising edge,
- * CAPTURE = 2: capture on falling edge,
- * CAPTURE = 3: capture on both (rising and falling) edges.

Note: COUNTER2 module cannot be activated simultaneously with PWM2 module.

3.2.4. Digital Outputs (DOUT)

Description and Electrical Design

Monarco HAT has 4 digital non-isolated open-drain outputs. The open-drain topology was chosen for best flexibility in connecting various devices.

Outputs are able to sink up to 1 A current, which is enough for direct connection of a robust contactor coil or a small DC motor. Maximal switching voltage is 40 V. All outputs are protected against short-circuit, peak overvoltage and driver thermal overload.

To allow direct connection to a standard 5 V CMOS/TTL logic inputs, pull-up resistors (1.2 kOhm) with a diode to 5 V rail are integrated.

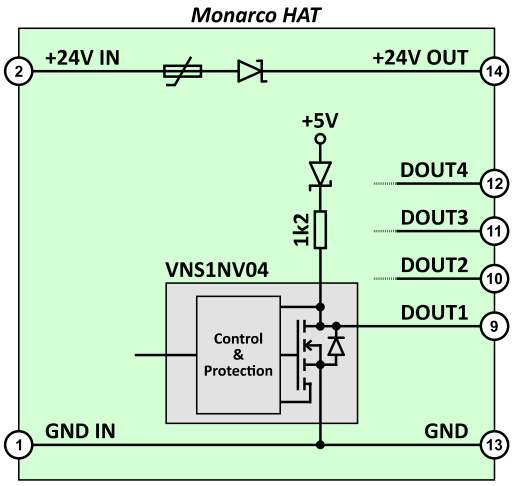

Image 3.4: Digital outputs internal implementation.

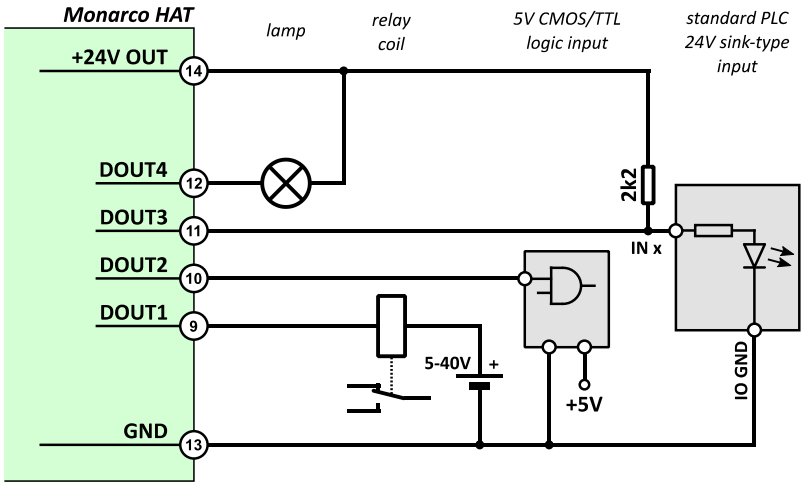

Image 3.5: Digital inputs connection examples with various actuator types.

Technology Functions Overview

Digital outputs can be also driven by advanced technology functions, which are not possible with Raspberry Pi integrated GPIOs:

- PWM (Pulse Width Modulator) with frequency widely adjustable in continuous range 1 Hz to 100 kHz, which can be also considered as a versatile square-wave generator which period and duty cycle is controlled by cyclic process data coming from Raspberry Pi,
- PULSE-DIR generator for motion control based on servo drives with "pulse/dir" control (Note: will be available in future firmware release).

Here are a few examples of devices which inputs can be driven by Monarco HAT DOUT with PWM:

- solid-state-relays for power modulation of various lighting or electric heating elements,
- DC motors with a proper H-bridge power stage,
- small servo motors with "RC servo" type input (1 ms to 2 ms pulse width control).

PWM Technology Function

There are two PWM hardware function modules available:

- **PWM1 – Output channels: A = DOUT1, B = DOUT2, C = DOUT3,**
- **PWM2 – Output channels: A = DOUT4.**

Each PWM module contains 16bit internal counter clocked by 32 MHz clock with selectable prescaler (1, 8, 64, 512, other values possible with firmware modification) and TOP value register. Counter value is incremented from zero up to TOP by prescaled clock. For each output channel there is a CMPx register which represents PWM modulated value for given channel. Thanks to a synchronous buffering on both TOP and CMPx registers, output signal is phase correct when changing both frequency and modulated value. Prescaler selection is not buffered. It's switched just when process data are received so switching it during operation can lead to unexpected behavior.

Note: From the description above it's obvious that all output channels on PWM1 module have common output frequency.

Note: PWM2 module cannot be activated simultaneously with COUNTER2 module.

Output frequency is given by formula: $f_{PWM} = 32\ \text{MHz} / (prescaler \cdot TOP)$

Where *prescaler* can be { 1, 8, 64, 512 }, and *TOP* can be 0 to 65532 - it is represented by 14bit value in process data and then multiplied by 4. Modulated value is represented by 16bits in process data and internally scaled to 0 to *TOP* range. Lowest possible *prescaler* should be preferred for given frequency, because it leads to bigger value of *TOP* and so better resolution of modulated value.

prescaler	recommended range low boundary	recommended range high boundary
1	≥ 1 kHz TOP = 32000	≤ 100 kHz TOP = 320
8	≥ 100 Hz TOP = 40000	< 1 kHz TOP = 4000
64	≥ 10 Hz TOP = 50000	< 100 Hz TOP = 5000
512	≥ 1 Hz TOP = 62500	< 10 Hz TOP = 6250

3.3. Connecting ARM Debugger (P3)

- There is a space for standard 9 (10) pin ARM Cortex debug connector with SWD (single wire debug) interface on board, marked as P3. Normally you do not need this, so it's not populated. If you want to hack Monarco MCU firmware, proper header can be soldered in (20021511-00010T4LF type by FCI).
- Compatible with "J-LINK 9-PIN CORTEX-M ADAPTER" provided by Segger.

P3 – ARM MCU Debug Interface			
1	VCC 3.3V	**2**	SWDIO
3	GND	**4**	SWDCLK
5	GND	**6**	SWO
7	–	**8**	Not Connected
9	GND	**10**	RESETn

3.4. Connecting Power for Touchscreen Display (P4)

- Monarco provides dedicated connector, marked as P4, with 5 VDC output to power a touchscreen display connected to the Raspberry Pi. We tested this with the official Raspberry Pi 7" Touchscreen Display.
- Output is switched off together with Raspberry Pi 5 V output by watchdog.
- *Available output current:* see *Technical Specifications / Power Supply* chapter.
- *Compatible connectors plug type:* DF3-2S-2C by Hirose.

P4 – 5 VDC output for touchscreen display			
1	GND	**2**	5 VDC

4. Technical Specifications

4.1. Mechanical

- *Board shape:* Compatible with the HAT (Hardware Attached on Top) standard[5]
- *Board dimensions:* 65 mm x 56 mm, 3 mm radius corners.
- *Height:*
  - *Spacers between Raspberry Pi and Monarco HAT:* 12 mm
  - *Board thickness:* 1.6 mm
  - *Highest component on top (P2 connector):* 11 mm
- *Note:* Can be used with the display cable attached to the Raspberry Pi.

4.2. Environment

- *Ambient operating temperature:* 0 °C to 55 °C
- *Ambient operating humidity:* 10 % to 90 % (with no condensation)
- *Atmosphere:* free from corrosive gases
- *Ambient storage temperature:* -20 to 70 °C (excluding battery)
- *RoHS compliant*

4.3. Wiring Terminals (P2)

- *Terminal system:* 26 pin detachable connector with push-in technology by Phoenix Contact. Spring-cage connection – just plug the wire in without any tool, easy wire release.
- *Wire gauge:* from 0.2 mm^2 up to 1.5 mm^2 (AWG 24 - AWG 16)
- *Plug type:* DFMC 1,5/13-ST-3,5-LR (1790593) by Phoenix Contact

4.4. Power Supply

- *Input (P2):* 10 - 30 V (use at least 12 V for analog output 10 V full range)
  - Reverse polarity protection
  - Overcurrent protection by resettable fuse, 2.2 A trip current
- *Internal Outputs Rating:* 5 V, 2.3 A; 3.3 V, 200 mA
- *Touchscreen Display Output (P4):* 5 V, 700 mA
- *Protection:* Peak Current Limit, Thermal Shutdown and Restart
- *Note:* Supply output to the Raspberry Pi and touchscreen display is software switchable from MCU, can be used for HW watchdog.

4.5. Microcontroller (MCU)

- 32-bit ARM Cortex-M3 core (Silicon Labs EFM32 series)
- EFM32G230F128
- 16 kB RAM
- 128 kB Program Flash
- 32 MHz crystal clock

[5] https://github.com/raspberrypi/hats/blob/master/hat-board-mechanical.pdf

4.6. Real Time Clock (RTC)

- *RTC chip type:* MCP79410 with 32768 Hz crystal
- *Interface:* I2C, address 0x6f (RTC/SRAM) + 0x57 (EEPROM)
- *Wiring:* connected to Raspberry Pi I2C-1 and to MCU I2C
- *Battery type:* CR1225 or CR1216

4.7. RS-485 Interface

- *Transceiver chip type:* SN65HVD75
- *Data rate:* up to 20 Mbps (as per transceiver specification, limited by MCU USART)
- *ESD protection:*
    - Transceiver internal - IEC 61000-4-2 of up to ±12 kV, IEC 61000-4-4 of up to ±4 kV
    - Additional - TVS Diode -7 V / +12 V around GND
- *Line load:* 3/20 unit load (213 transceivers on line)
- *Line termination:* on-board 120R termination resistor, software switchable
- *Wiring:* Connected to MCU UART. Standard function is bidirectional forwarding to/from Raspberry Pi UART with buffering and RS-485 Driver-Enable signal generation with proper timing. Alternative functions can be implemented in MCU firmware.

4.8. 1-Wire Interface

- *Controller chip type:* DS2482-100
- *Controller interface:* I2C, address 0x18
- *Wiring:* connected to Raspberry Pi I2C-1 and to MCU I2C
- *ESD protection:* DS9503 chip
- *Note:* 5 V output with software controlled power switch for controller and network reset in case of stuck

4.9. Digital Inputs

- *Optically isolated, common negative potential*
- *Logic low voltage:* -30 to 1.8 V
- *Logic high voltage:* 3.5 to 30 V
- *Note:* input low/high levels compatible with 5V CMOS logic
- *Input Resistance:* 3 kOhm + typ. 1.5 V drop
- *Input current:* typ. 7.5 mA at 24 V, 3.5 mA at 12 V, 1.2 mA at 5 V
- *Maximal input frequency:* 500 kHz
- *Input propagation delay (typical / maximal): 70 ns / 100 ns*
- *Dielectric strength:* 300 V
- *Optocoupler type:* TLP2161
- *Features:*
    a) Standard digital input, selectable input glitch filter (under development)
    b) HW counter, 2 channels (DI1-DI2, DI3-DI4) with selectable function:
        - Pulse counter, optional direction signal
        - Quadrature encoder counter
        - Frequency / Period / Pulse length measurement (under development)

4.10. Digital Outputs

- *Output type:* Open drain, not isolated, internal 1.2 kOhm pull-up and diode to 5 V
- *Note:* Pull-ups design with diode enables direct driving TTL inputs, but it's not intrusive when driving high voltage loads
- *Continuous current max:* 1 A
- *Switched voltage max:* 40 V
- *Maximal switching frequency:* 100 kHz
- Rise time to 5/24 V, 1 kOhm load: – to be specified –
- Fall time to 5/24 V, 1 kOhm load: – to be specified –
- *Driver fault tolerance:* current limit 1.7 A, thermal shutdown, short circuit protection, peak overvoltage protection
- *Driver type:* VNS1NV04DP-E
- *Features:*
    a) Standard digital output
    b) PWM with configurable frequency up to 100 kHz, up to 16 bit resolution - on all outputs (Note: PWM function at DO4 exclusive with DI3-DI4 counter functions)
    c) Pulse-dir for motion control (Note: will be available in future firmware release)

4.11. Analog Inputs

- *HW Resolution:* 12 bit (0 – 4095)
- *Accuracy:* 0.3 % of 10V range (30 mV)
- *Bandwidth:* 500 Hz
- *Input type:* Voltage / Current loop – software switchable
- *Configurable software low pass filter* (under development)
- *Voltage mode:*
    - *Range 1:* 0 - 10 V
    - *Range 2:* 0 - 5 V
    - *Input load resistance:* 5 kOhm
    - *Absolute maximum voltage:* +30 / -5 V
- *Current loop mode:*
    - *Range:* 0 - 25 mA
    - *Shunt resistance:* 200 Ohm ±0.1 %
    - *Absolute maximum current:* ±35 mA

4.12. Analog Outputs

- *HW Resolution:* 12 bit (0 - 4095)
- *Range:* 0 - 10 V
- *Accuracy:* 0.5 % of 10 V range, with 4 kOhm load to GND
- *Settling time:* 0.5 ms (0 to 100% output)
- *Max load:* 20 mA sink

4.13. LEDs

- 1 × orange, 8 × green LEDs
- software controlled from MCU
- standard functions: system status, digital inputs status, digital outputs status

4.14. ID EEPROM

- *Type:* 24LC32
- *Interface:* I2C (connected to Raspberry Pi according to the HAT standard), address 0x50
- *Capacity:* 4096 byte
- EEPROM is factory pre-programmed according to the HAT (Hardware Attached on Top) specification. It contains device identification and a device tree overlay – see chapter 3.1.2.