# **PHAT-RV TECHNICAL DESCRIPTION**

**pHAT Module Series**

Document: pHAT-RV Technical Description

Issue: 1.00.00

Date: 2nd May 2019

www.designersystems.co.uk

**Our aim is to provide customers with timely and comprehensive service. For any assistance, please contact our company headquarters:**

**Designer Systems Ltd.**
11 Castle Street, Truro, Cornwall TR1 3AF, United Kingdom.
Tel: +44 (0) 1872 262000
Email: sales@designersystems.co.uk

**For more information, please visit:**
http://www.designersystems.co.uk

**For technical support, or to report documentation errors, please visit:**
http://www.designersystems.co.uk/robotics
Or email to: support@designersystems.co.uk

## GENERAL NOTES

DESIGNER SYSTEMS OFFERS THE INFORMATION AS A SERVICE TO ITS CUSTOMERS. THE INFORMATION PROVIDED IS BASED UPON CUSTOMERS' REQUIREMENTS. DESIGNER SYSTEMS MAKES EVERY EFFORT TO ENSURE THE QUALITY OF THE INFORMATION IT MAKES AVAILABLE. DESIGNER SYSTEMS DOES NOT MAKE ANY WARRANTY AS TO THE INFORMATION CONTAINED HEREIN, AND DOES NOT ACCEPT ANY LIABILITY FOR ANY INJURY, LOSS OR DAMAGE OF ANY KIND INCURRED BY USE OF OR RELIANCE UPON THE INFORMATION. ALL INFORMATION SUPPLIED HEREIN IS SUBJECT TO CHANGE WITHOUT PRIOR NOTICE.

## COPYRIGHT

THE INFORMATION CONTAINED HERE IS PROPRIETARY TECHNICAL INFORMATION OF DESIGNER SYSTEMS LTD. TRANSMITTING, REPRODUCTION, DISSEMINATION AND EDITING OF THIS DOCUMENT AS WELL AS UTILIZATION OF THE CONTENT ARE FORBIDDEN WITHOUT PERMISSION. OFFENDERS WILL BE HELD LIABLE FOR PAYMENT OF DAMAGES. ALL RIGHTS ARE RESERVED IN THE EVENT OF A PATENT GRANT OR REGISTRATION OF A UTILITY MODEL OR DESIGN.

***Copyright © Designer Systems Ltd. 2019. All rights reserved.***

# About the Document

## History

| Revision | Description | Date | Author |
|---|---|---|---|
| 0.01.00 | Create | 3/2/19 | DIO |
| 1.00.00 | First release | 2/5/19 | - |

# Contents

# Tables



# Figures

# 1 Introduction

This document defines the pHAT-RV Remote Vehicle module and describes the hardware interface that is connected to the customers Raspberry-Pi application.
This document can help customers quickly understand module interface specifications, electrical and mechanical details, as well as other related information of the module. Associated with the quick start guide and demo software, customers can use this document to easily set up the module.

# 2 Product Concept

## 2.1 General Description

The Designer Systems pHAT-RV is a fully featured remote vehicle driver solution with four [4] unipolar stepper motor drivers, four [4] servo motor drivers and Lithium battery power control features. Specifically designed for the Raspberry-Pi Zero (other Raspberry-Pi boards fully supported) the pHAT-RV features Lithium battery power regulation, a battery charging system and high speed I2C communication for easy project integration.

pHAT-RV supports the majority of analogue servos and unipolar stepper motors (selectable 1 or 2 phase) and also provides global activation of new motor positions & movement complete registers for superior control. Motors can be fully speed controlled, reversed, started/stopped and stepper motors stepped with up to 65535 steps.

pHAT-RV provides a 5V @ 1A power supply (link selectable) to power the connected R-Pi Zero and features an on-board Lithium Polymer charge controller that can charge the battery at 1A (2000mAh min capacity) from an external 9-12VDC power adaptor (not supplied).

Due to compact form factor, ultra-low power consumption and extended temperature range, pHAT-RV is a best choice for a wide range of robotic applications.

The module fully complies with the RoHS directive of the European Union

## 2.2 Key Features

The following table describes the key features of the pHAT-RV.

**Table 1: Key Features**

| Features | Details |
|---|---|
| Servo motor | 4 x Analogue (standard) |
| Servo Voltage | 6.0 ~ 7.4V |
| Servo Power | 2 Amps (stalled) |
| Servo Control | • Enable/Disable/Reverse<br>• Soft Start<br>• Position with speed control<br>• Position complete flag |
| Servo Connection | 4 x 3pin header (standard) |
| Stepper motor | 4 x Unipolar stepper |
| Stepper Voltage | 5.0 ~ 7.4V |
| Stepper Power | 600mA per phase max. |
| Stepper Control | • Enable/Disable/Reverse/Stop with/without holding torque<br>• Full/half step selection<br>• Continuous step with speed control<br>• Step 1-65535 with speed control |
| Stepper Connection | 4 x 5pin JST ZHR 1.5mm pitch |
| Lithium Battery Voltage | 7.4VDC nom. |
| Lithium Battery Capacity | 2000mA/h (2A/h) min. |
| Lithium Battery Charge | 1A @ 8.4V max. |
| R-Pi Output Voltage | 5.0VDC ± 0.1V |
| R-Pi Output Power | 1A max. |
| $I^2C$ Speed | 400kHz max. |
| Environmental | Operating Temperature -10°C to 50°C |
| Dimensions | 65 x 30 x 8.5mm |
| Weight | 14g approx. |

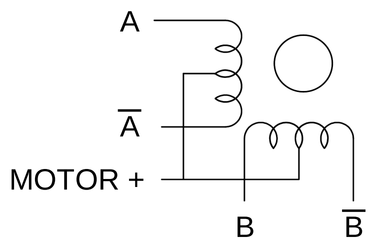
# 3 Application

## 3.1 Stepper Motor Connection

The stepper motor connections are marked 'MOTOR 1' to 'MOTOR 4' and consist of a five (5) pin JST ZHR 1.5mm pitch connector, see section 3.4.3 for pinning.

This connection is standard for most 24mmD stepper motors, such as the DS-STMOT1, but there is also a pre-made 200mm JST ZHR cable available, DS-STCAB1, and the JST ZHR mating receptacle and pins are also available from Farnell electronics as items 2078162 and 1830898 respectively.

Most unipolar stepper motor phases are wired as follows:




It is recommended that you double check your motor phase connections before power application. Six wire stepper motors are also supported by connecting the two centre taps both to MOTOR +.

## 3.2 Servo Motor Connection

The servo motor connections are marked 'SERVOS' from 1 to 4 and consist of three pins marked 'GND', 'V+' and 'SV'. Analogue servos should be connected to these pins, noting polarity, BLACK is normally GND, see section 3.4.4 for pinning.

## 3.3 Raspberry-Pi Board Support

The RV supports all of the current Raspberry-Pi board versions. However, when powering a Raspberry-Pi from the RV there is a 1A power limitation on the 5V supply from the RV. This means that some Raspberry-Pi boards such as the newer Model 3 B & B+ may struggle to operate correctly especially when other peripherals such as USB keyboards/mice/drives are connected. The RV is not designed for these types of applications and the 5V supply will shut-down if overloaded.

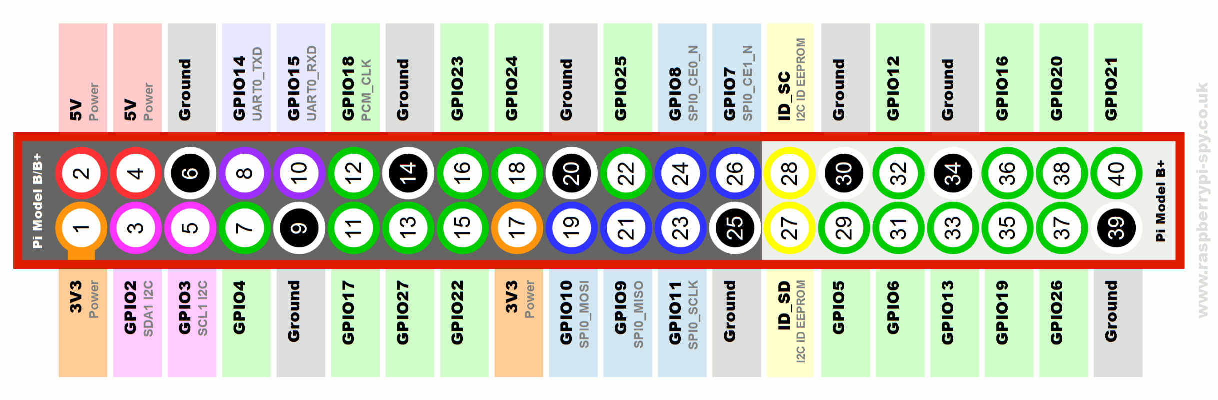
## 3.4 Operation

When the Lithium Battery is connected to the XT60 connector 5V power will be delivered to the connected Raspberry-Pi board and it will boot normally.

# 3.5 Pin Assignment

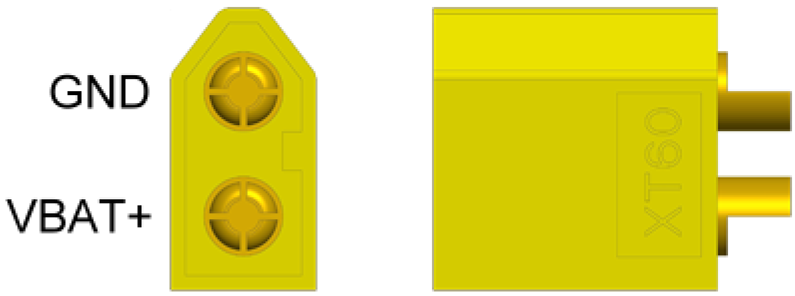
## 3.5.1 Raspberry-Pi Header

The RV module connects to the Raspberry-Pi 40pin header, pinout as follows:



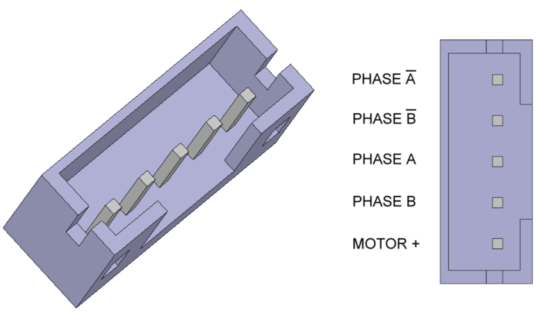

## 3.5.2 Lithium Battery Dean XT60 Socket

The Lithium Battery input connector is a Deans XT60 socket, pinout as follows:



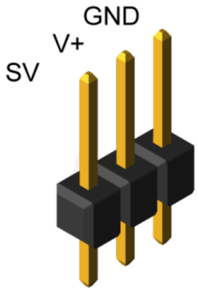

## 3.5.3 Stepper Motor Header

The 4 x Stepper motor headers are 5way JST ZHR 1.5mm pitch, pinout as follows:



### 3.5.4 Servo Header

The 4 x Servo motor headers are 3way 2.54mm pitch, pinout as follows:




### 3.5.5 Charger Input Socket

The Charger input is a 2.5mm centre pin coaxial power socket, pinout as follows:

## 3.6 Power Supply

### 3.6.1 Lithium Battery

The RV provides a Deans XT60 socket to connect to a 7.4V Lithium-Ion battery pack with a minimum capacity of 2000mA/h (2A/h).

**WARNING:** A Lithium-Ion or Lithium-Polymer battery is capable of delivering very high levels of power to the servo and stepper motors. This can cause a significant amount of heat to be generated in the motor which may cause the motor to fail if the motor is stalled.

To help to prevent motor damage fuse resistors R1, R4 and R5 have been included that will go open circuit if significant levels of current are drawn by the motors. These can be replaced with zero ohm (0R0) 0805 size resistors if damaged

### 3.6.2 Charger Input

The Charger Input requires a 9.0 ~ 12VDC @ 2Amp min. supply from an external power adaptor to charger the Lithium-Ion battery. NOTE: Charging should be undertaken during no motor activity for best results.

### 3.6.3 Raspberry-Pi Power

The RV provides a 5.0V supply output to provide the Raspberry-Pi board with power. The table below describes the module output supply and ground pins.

**Table 2: Power Supply Pins**

| Pin Name | Pin No | Description | Min | Typ. | Max | Unit |
|---|---|---|---|---|---|---|
| 5V | 2, 4 | Power Supply Output @ 1A | | 5.0 | | V |
| Ground | 6,9,14,20, 25,30,34, 39 | Power Ground | | | | |

If Raspberry-Pi power is not required jumper link JP1 may be removed with a soldering iron. Note however that this will invalidate the warranty.

## 3.7 I²C Interface

### 3.7.1 I²C Interface Pins

The RV provides I²C data (SDA) and clock (SCL) connections on the 20+20 header that connect to the SDA and SCL on the Raspberry-Pi board. The table below describes the module I²C pins.

**Table 3: I²C Interface Pins**

| Pin Name | Pin No | I/O | Description | Comment |
|---|---|---|---|---|
| SDA | 3 | DIO | I2C Data | 3.3V level |
| SCL | 5 | CO | I2C Clock | 3.3V level |

The RV does NOT have I²C pullups but relies on the pullups present on the Raspberry-Pi board. When not connecting to a Raspberry-Pi board external pullups of 4.7Kohms should be connected on SDA and SCL to a 3.3V supply.

### 3.7.2 I²C Communication

Up to four RV modules may be connected to the same Raspberry-Pi board or I²C bus and accessed individually using their own individual address.
The following table shows how the pads are soldered for the different binary addresses.

**Table 4: I²C Address Settings**

| Address (xx) | A0 | A1 |
|---|---|---|
| 00 | OPEN | OPEN |
| 01 | SHORT | OPEN |
| 10 | OPEN | SHORT |
| 11 | SHORT | SHORT |

The binary address (xx) above is used in conjunction with the device ID 11101xxD ($0xE8_{hex}$) to form the complete device address i.e. if both jumpers are left unconnected (default) then the device address would be $1110100D_{binary}$.
The ‘D’ bit determines if a read or a write to the RV is to be performed. If the ‘D’ bit is set ‘1’ then a register read is performed or if clear ‘0’ a register write.

### 3.7.3 I²C Registers

To write individual registers a device write must be undertaken by the Raspberry-Pi which consists of a Start condition, device ID ('D' bit cleared), register to start write, one or more bytes of data to be written and a stop condition (see Figure 2 for I²C write protocol).

To read individual data and status registers a device write then read must be undertaken by the Raspberry-Pi which consists of a Start condition, device ID ('D' bit clear), register to start read and a Stop condition.
This is followed by a read, which consists of a Start condition, device ID ('D' bit set), followed by data from the register specified and terminated with a Stop condition. The RV also auto increments the register specified for every additional read requested by the Master I²C device, which allows more than one register to be read in one transaction. This allows for example Register 0 to Register 5, current UTC time, to be read in one transaction (see Figure 3 for I²C read protocol).
There are 25 individual registers that can be written and 8 registers that can be read within the RV as follows:

**Table 5: I²C Registers**

| Register name | Type | Register address | | Description |
|---|---|---|---|---|
| | | Hex | Binary | |
| Register address | w | 00 | 00000000 | Start register to write to |
| Stepper 1 control | w | 01 | 00000001 | Stepper motor 1 control<br>U \| U \| U \| E \| D \| C \| B \| A<br>A = Operate (0 - Stepper disabled 1 - Stepper enabled)<br>B = Reverse (0 - Stepper normal 1 - Stepper reversed)<br>C = Rotate (0 - Single step 1 - Constant rotation)<br>D = Step (0 - Normal step 1 - Half step)<br>E = Hold (0 - No hold on stop 1 - Hold on stop)<br>U = Unused |
| Stepper 1 steps MSB | w | 02 | 00000010 | Number of steps MSB |
| Stepper 1 steps LSB | w | 03 | 00000011 | Number of steps LSB |
| Stepper 1 speed | w | 04 | 00000100 | Step speed in quarter mS (0.25mS) |

| Register name | Type | Register address | | Description |
|---|---|---|---|---|
| | | Hex | Binary | |
| Stepper 2 control | w | 05 | 00000101 | Stepper motor 2 control<br>U U U E D C B A<br>A = Operate (0 - Stepper disabled 1 - Stepper enabled)<br>B = Reverse (0 - Stepper normal 1 - Stepper reversed)<br>C = Rotate (0 - Single step 1 - Constant rotation)<br>D = Step (0 - Normal step 1 - Half step)<br>E = Hold (0 - No hold on stop 1 - Hold on stop)<br>U = Unused |
| Stepper 2 steps MSB | w | 06 | 00000110 | Number of steps MSB |
| Stepper 2 steps LSB | w | 07 | 00000111 | Number of steps LSB |
| Stepper 2 speed | w | 08 | 00001000 | Step speed in quarter mS (0.25mS) |
| Stepper 3 control | w | 09 | 00001001 | Stepper motor 3 control<br>U U U E D C B A<br>A = Operate (0 - Stepper disabled 1 - Stepper enabled)<br>B = Reverse (0 - Stepper normal 1 - Stepper reversed)<br>C = Rotate (0 - Single step 1 - Constant rotation)<br>D = Step (0 - Normal step 1 - Half step)<br>E = Hold (0 - No hold on stop 1 - Hold on stop)<br>U = Unused |
| Stepper 3 steps MSB | w | 0A | 00001010 | Number of steps MSB |
| Stepper 3 steps LSB | w | 0B | 00001011 | Number of steps LSB |
| Stepper 3 speed | w | 0C | 00001100 | Step speed in quarter mS (0.25mS) |

| Register name | Type | Register address | | Description |
|---|---|---|---|---|
| | | Hex | Binary | |
| Stepper 4 control | w | 0D | 00001101 | Stepper motor 3 control<br>U \| U \| U \| E \| D \| C \| B \| A<br>A = Operate (0 - Stepper disabled 1 - Stepper enabled)<br>B = Reverse (0 - Stepper normal 1 - Stepper reversed)<br>C = Rotate (0 - Single step 1 - Constant rotation)<br>D = Step (0 - Normal step 1 - Half step)<br>E = Hold (0 - No hold on stop 1 - Hold on stop)<br>U = Unused |
| Stepper 4 steps MSB | w | 0E | 00001110 | Number of steps MSB |
| Stepper 4 steps LSB | w | 0F | 00001111 | Number of steps LSB |
| Stepper 4 speed | w | 10 | 00010000 | Step speed in quarter mS (0.25mS) |
| Servo 1 position | w | 11 | 00010001 | Servo position (8uS resolution) |
| Servo 1 control | w | 12 | 00010010 | Servo 1 control<br>A \| B \| C \| D \| S \| S \| S \| S<br>A = Operate (0 – Servo disabled 1 – Servo enabled)<br>B = Reverse (0 – Servo normal 1 – Servo reversed)<br>C = Soft-start control (0 – Disabled 1 – Enabled)<br>D = Speed control (0 – Disabled 1 – Enabled)<br>S..S = Servo speed value 0 to 15 (0 = slowest) |
| Servo 2 position | w | 13 | 00010011 | Servo position (8uS resolution) |
| Servo 2 control | w | 14 | 00010100 | Servo 2 control<br>A \| B \| C \| D \| S \| S \| S \| S<br>A = Operate (0 – Servo disabled 1 – Servo enabled)<br>B = Reverse (0 – Servo normal 1 – Servo reversed)<br>C = Soft-start control (0 – Disabled 1 – Enabled)<br>D = Speed control (0 – Disabled 1 – Enabled)<br>S..S = Servo speed value 0 to 15 (0 = slowest) |

| Register name | Type | Register address | | Description |
|---|---|---|---|---|
| | | Hex | Binary | |
| Servo 3 position | w | 15 | 00010101 | Servo position (8uS resolution) |
| Servo 3 control | w | 16 | 00010110 | Servo 3 control<br>A \| B \| C \| D \| S \| S \| S \| S<br>A = Operate (0 – Servo disabled 1 – Servo enabled)<br>B = Reverse (0 – Servo normal 1 – Servo reversed)<br>C = Soft-start control (0 – Disabled 1 – Enabled)<br>D = Speed control (0 – Disabled 1 – Enabled)<br>S..S = Servo speed value 0 to 15 (0 = slowest) |
| Servo 4 position | w | 17 | 00010111 | Servo position (8uS resolution) |
| Servo 4 control | w | 18 | 00011000 | Servo 4 control<br>A \| B \| C \| D \| S \| S \| S \| S<br>A = Operate (0 – Servo disabled 1 – Servo enabled)<br>B = Reverse (0 – Servo normal 1 – Servo reversed)<br>C = Soft-start control (0 – Disabled 1 – Enabled)<br>D = Speed control (0 – Disabled 1 – Enabled)<br>S..S = Servo speed value 0 to 15 (0 = slowest) |
| Global update | w | 19 | 00011001 | Global update control |
| Stepper 1 status | r | 00 | 00000000 | Stepper motor 1 status<br>A \| 0 \| 0 \| 0 \| 0 \| 0 \| 0 \| 0<br>A = Movement (0 – Complete 1 – In-progress) |
| Stepper 2 status | r | 01 | 00000001 | Stepper motor 2 status<br>A \| 0 \| 0 \| 0 \| 0 \| 0 \| 0 \| 0<br>A = Movement (0 – Complete 1 – In-progress) |
| Stepper 3 status | r | 02 | 00000010 | Stepper motor 3 status<br>A \| 0 \| 0 \| 0 \| 0 \| 0 \| 0 \| 0<br>A = Movement (0 – Complete 1 – In-progress) |

| Register name | Type | Register address | | Description |
|---|---|---|---|---|
| | | Hex | Binary | |
| Stepper 4 status | r | 03 | 00000011 | Stepper motor 4 status<br>A \| 0 \| 0 \| 0 \| 0 \| 0 \| 0 \| 0<br>A = Movement (0 – Complete 1 – In-progress) |
| Servo 1 status | r | 04 | 00000100 | Servo 1 status<br>A \| B \| C \| D \| 0 \| 0 \| 0 \| 0<br>A = Operation (0 – Servo disabled 1 – Servo enabled)<br>B = Reverse (0 – Servo normal 1 – Servo reversed)<br>C = Soft-start (0 – Complete 1 – In-progress)<br>D = Movement (0 – Complete 1 – In-progress) |
| Servo 2 status | r | 05 | 00000101 | Servo 2 status<br>A \| B \| C \| D \| 0 \| 0 \| 0 \| 0<br>A = Operation (0 – Servo disabled 1 – Servo enabled)<br>B = Reverse (0 – Servo normal 1 – Servo reversed)<br>C = Soft-start (0 – Complete 1 – In-progress)<br>D = Movement (0 – Complete 1 – In-progress) |
| Servo 3 status | r | 06 | 00000110 | Servo 3 status<br>A \| B \| C \| D \| 0 \| 0 \| 0 \| 0<br>A = Operation (0 – Servo disabled 1 – Servo enabled)<br>B = Reverse (0 – Servo normal 1 – Servo reversed)<br>C = Soft-start (0 – Complete 1 – In-progress)<br>D = Movement (0 – Complete 1 – In-progress) |
| Servo 4 status | r | 07 | 00000111 | Servo 4 status<br>A \| B \| C \| D \| 0 \| 0 \| 0 \| 0<br>A = Operation (0 – Servo disabled 1 – Servo enabled)<br>B = Reverse (0 – Servo normal 1 – Servo reversed)<br>C = Soft-start (0 – Complete 1 – In-progress)<br>D = Movement (0 – Complete 1 – In-progress) |

| Register name | Type | Register address | | Description |
|---|---|---|---|---|
| | | Hex | Binary | |
| Firmware version | r | 08 | 01001000 | Firmware (Bit 0-3 = minor version, Bit 4-5 = major version) |

### 3.7.4 I$^2$C Stepper Control Register Format

Each stepper control register consists of five control bits defined as follows:

**Table 6: I$^2$C Stepper Control Register Format**

| Bit | Bit value | Function | Description |
|---|---|---|---|
| A | 1 (0x01) | Operate bit | When set activates the stepper motor being controlled |
| B | 2 (0x02) | Reverse bit | When set reverses the direction of the stepper motor being controlled |
| C | 4 (0x04) | Rotate bit | When set makes the stepper motor constantly rotate rather than step the number of steps defined in the step number register |
| D | 8 (0x08) | Step bit | When set allows the stepper motor to be configured for half-step rather than full step operation |
| E | 16 (0x10) | Hold bit | When set the stepper motor will hold its position (holding torque) once the number of steps have been made rather than go free (no holding torque). |

### 3.7.5 I$^2$C Stepper Status Register Format

Each stepper status register consists of one status bit defined as follows:

**Table 7: I$^2$C Stepper Status Register Format**

| Bit | Bit value | Function | Description |
|---|---|---|---|
| A | 128 (0x80) | Movement bit | Set when stepper movement is in-progress |

### 3.7.6 I²C Servo Control Register Format

Each servo control register consists of four control bits and a four bit speed control value defined as follows:

**Table 8: I²C Servo Control Register Format**

| Bit | Bit value | Function | Description |
|---|---|---|---|
| A | 128 (0x80) | Operate bit | When set activates the servo being controlled |
| B | 64 (0x40) | Reverse bit | When set reverses the position value for the servo being controlled |
| C | 32 (0x20) | SoftStart bit | When set on servo first activation, see operate bit above, feeds position pulses to the servo in a ramping manner until position is attained |
| D | 16 (0x10) | Speed bit | When set applies the speed value 0 to 15 |
| SSSS | 0 – 15 (0x00 – 0x0F) | Speed value | Speed value 0 to 15 |

### 3.7.7 I²C Servo Status Register Format

Each servo status register consists of four status bits defined as follows:

**Table 9: I²C Servo Status Register Format**

| Bit | Bit value | Function | Description |
|---|---|---|---|
| A | 128 (0x80) | Operation bit | Set when servo is operational |
| B | 64 (0x40) | Reverse bit | Set when servo is in reverse operation |
| C | 32 (0x20) | SoftStart bit | Set when SoftStart is in operation |
| D | 16 (0x10) | Movement bit | Set when servo movement is in-progress |

### 3.7.8 Servo Movement Complete Determination

Servo status register Bit (D) $16_{decimal}$ are cleared to indicate if the current servo movement has completed.
This indication is not derived from mechanical or electrical feedback from the servo being controlled but is a function of the current servo speed selected and position.
When the slowest servo speed (0) is selected the determination of movement completion is at its best. This is because the positional change of the servo between its current and final position has been split into many sub-positions which must be attained before the final

position is reached. These many sub-positions ensure that the mechanical position closely relates to the position requested by the pulse width and therefore the determination of final position (movement complete) will closely relate to mechanical position. As servo speed is increased the error between mechanical position and pulse width position increases and movement completion accuracy is degraded.

### 3.7.9 $I^2C$ Global Update Register

Once all the required position & control registers have been set a write to the $0x19_{hex}$ (Global update) must be made to activate all the new positions.

### 3.7.10 $I^2C$ Write Example

To step stepper motor 1 500 half-steps clockwise at 3mS per step and then hold in that position and servo 1 to position 30 running at speed 0 write:

| | |
|---|---|
| Byte 1 (RV Adr) | $11101000_{binary}$ |
| Byte 2 (Register 0) | $0_{decimal}$ |
| Byte 3 (Register 1) | $25_{decimal}$, $19_{hex}$ |
| Byte 4 (Register 2) | $1_{decimal}$, $01_{hex}$ |
| Byte 5 (Register 3) | $232_{decimal}$, $E8_{hex}$ |
| Byte 6 (Register 4) | $12_{decimal}$, $0C_{hex}$ |

Then write:

| | |
|---|---|
| Byte 1 (RV Adr) | $11101000_{binary}$ |
| Byte 2 (Register 0) | $17_{decimal}$ |
| Byte 3 (Register 17) | 30decimal, $1E_{hex}$ |
| Byte 4 (Register 18) | $144_{decimal}$, $90_{hex}$ |

Then to activate write:

| | |
|---|---|
| Byte 1 (RV Adr) | $11101000_{binary}$ |
| Byte 2 (Register 0) | $25_{decimal}$ |
| Byte 3 (Register 25) | $0_{decimal}$ |

## 3.8 Application Software

### 3.8.1 Demonstration Software

Raspberry-Pi demonstration software written in Python is available to download from the website www.designersystems.co.uk/robotics

# 4 Electrical Characteristics

## 4.1 Absolute Maximum Ratings

Absolute maximum ratings for power supply and voltage on digital pins of the module are listed in the following table.

**Table 10: Absolute Maximum Ratings**

| Parameter | Min. | Max. | Unit |
|---|---|---|---|
| Lithium Battery Voltage (V+) | -0.3 | 12.0 | V |
| Input Voltage on SDA and SCL | -0.3 | 3.6 | V |
| 5V Output Current | | 1.2 | A |
| Storage temperature | -20 | 85 | $^{o}$C |

## 4.2 Operating Conditions

Normal operational conditions are listed in the following table.

**Table 11: Normal Operating Conditions**

| Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|
| Lithium Battery Voltage (V+) | 6.0 | 7.4 | 9.0 | V |
| Input voltage on SDA and SCL | | 3.3 | | V |
| 5V Output Current | | 200 | 1000 | mA |
| Lithium Battery Charge Current | | 1.0 | | A |
| Peak Lithium Battery Current | | | 3 | A |
| Stepper Motor Phase Current | | | 600 | mA |
| Operating Temperature | -10 | 25 | 50 | $^{o}$C |

## 4.3 Current Consumption

Normal values for current consumption are listed in the following table.

**Table 12: Current Consumption**

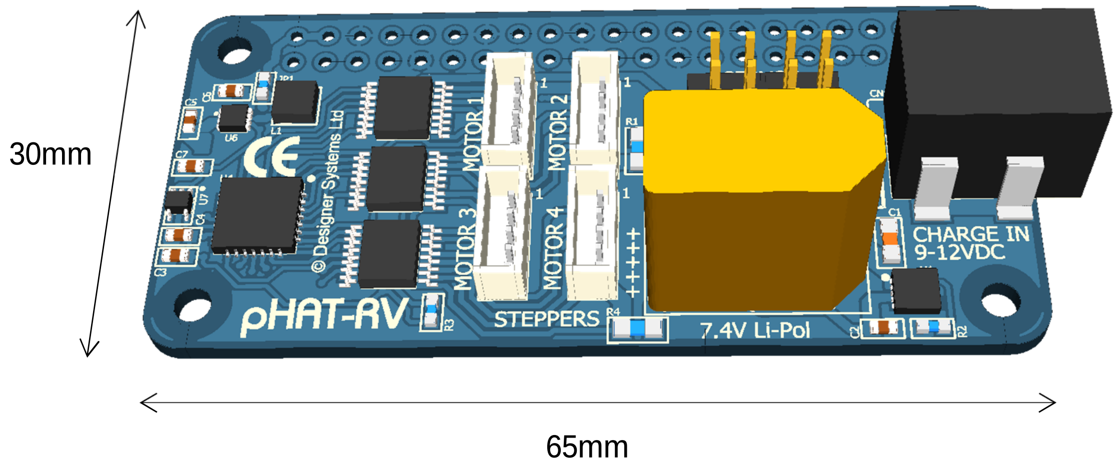
| Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|
| Battery Current – No R-Pi connected | | 4 | | mA |
| Battery Current – Powering R-Pi Zero (no peripherals) | | 100 | | mA |
| Battery Current – Powering R-Pi Zero (HDMI + USB keyboard) | | 250 | | mA |
| Battery Current – All steppers and servos running | | | 3 | A |

# 5 Mechanical

## 5.1 Dimensions

Mechanical drawing – all dimensions in millimetres.

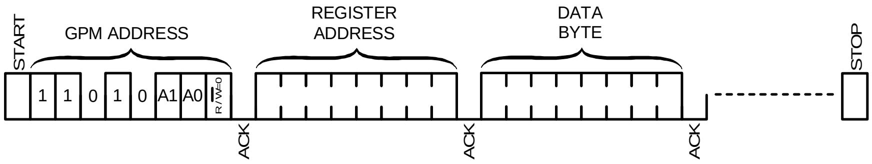
**Figure 1: Dimensions**



# 6 References

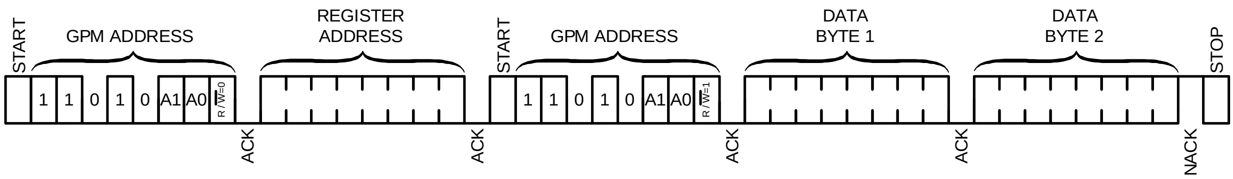
## 6.1 I²C protocols

### Figure 2: I²C Write protocol




Multiple bytes may be written before the ‘STOP’ condition. Data is written into registers starting at ‘REGISTER ADDRESS’, then ‘REGISTER ADDRESS’ +1, then ‘REGISTER ADDRESS’ +2 etc.
Each byte transfer is acknowledged ‘ACK’ by the RV until the ‘STOP’ condition.

### Figure 3: I²C Read protocol




‘DATA BYTE 1 & 2’ are register values returned from the RV. Each byte written is acknowledged ‘ACK’ by the RV , every byte read is acknowledged ‘ACK’ by the I2C Master. A Not-acknowledge ‘NACK’ condition is generated by the I2C Master when it has finished reading.

# 7 Appendix

**Table 13: Related Documents**

| Document Name | Remark |
|---|---|
| DS-STMOT1_Stepper motor.pdf | 24mmD Stepper motor for use with the RV |

**Table 14: Terms and Abbreviations**

| Abbreviation | Description |
|---|---|
| RV | Remote Vehicle |
| ESD | Electrostatic Discharge |
| $I^2C$ | Inter-Integrated Circuit |

# 8 Compliance

## WEEE Consumer Notice

This product is subject to Directive 2012/19/EC of the European Parliament and the Council of the European Union on Waste of Electrical and Electronic Equipment (WEEE) and, in jurisdictions adopting that Directive, is marked as being put on the market after August 13, 2005, and should not be disposed of as unsorted municipal/public waste. Please utilise your local WEEE collection facilities in the disposition and otherwise observe all applicable requirements. For further information on the requirements regarding the disposition of this product in other languages please visit www.designersystems.co.uk

## RoHS Compliance

This product complies with Directive 2011/65/EC (RoHS 2) and 2015/863/WU (RoHS 3) of the European Parliament and the Council of the European Union on the Restriction of Hazardous Substances (RoHS) which prohibits the use of various heavy metals (lead, mercury, cadmium, and hexavalent chromium), polybrominated biphenyls (PBB) and polybrominated diphenyl ethers (PBDE), Bis(2-Ethylhexyl) phthalate (DEHP), Benzyl butyl phthalate (BBP), Dibutyl phthalate (DBP) and Diisobutyl phthalate (DIBP).

## REACH Compliance

This product complies with Regulation 1907/2006 convering the Registration, Evaluation, Authorisation and restriction of Chemicals (REACH). Designer Systems Ltd confirms that none of its products or packaging contain any of the 174 Substances of Very High Concern (SVHC) on the REACH Candidate List in a concentration above the 0.1% by weight allowable limit.

## Battery Recycling

This product features an internal lithium coin cell that must be recycled at end of life. To remove slide the coin cell from its holder and to preserve natural resources please recycle the battery properly.