Athena Documentation

Release 0.0.1

The Athena contributors.

Jun 19, 2018

Contents

1 What is Athena? 1

1.1 HC-05 Bluetooth 1

1.2 IE2-1024 Encoder Family 4

1.3 Tetra locomotion 4

1.4 Indices and tables 5

CHAPTER 1

What is Athena?

Athena is a documentation project for robot builders.

1.1 HC-05 Bluetooth

1.1.1 Module

The HC-05 is a cheap and relatively small serial bluetooth device.

In both HC-05 and HC-06 modules, the hardware is exactly the same, the only difference is the firmware. There are notable differences between both firmwares as the HC-05 is more efficient, more configurable and has a much more extended AT command set.

The serial baudrate (hardware dependent) can be configured from 4800 to 3686400, allowing a pretty decend communication speed (very convenient for real-time communication with the robot).

Among all the available pins, the HC-05 firmware uses the following pins in a minimum system:

- UART_TXD and UART_RXD for serial transmission (communication with the microcontroller).
- PIO8 usually connects with a LED. When the module is power on, LED will flicker. And the flicker style will indicate which work mode is in since different modes have different flicker patrons. It will blink fast in pairing mode, slow in AT mode and fill double-blink each couple of seconds after paired with another device.
- PIO9 also is usually connected with a LED. It indicates whether the connection is built or not. When the Bluetooth serial is paired, the LED will be turned on. It means the connection is built successfully.
- PIO11 is the work mode switch. When this PIN port is input high level, the work mode will become order-response work mode (AT mode). While this PIN port is input low level or suspended in air (high impedance), the work mode will become automatic connection work mode.
- PIO0 and PIO1 to enable RX and TX lines.
- RESET, VCC and GND for obvious purposes.

Note: Although not mentioned in the datasheet, PIO8 already has a different blinking patron when the module is paired, so PIO9 can be a bit redundant.

Warning: It is important to mention that the UART_RXD line has no pull-up resistor, and it should be added if the microcontroller TXD does not have pull-up function.

With the HC-05 not only you are able to properly configure the module as master or slave, set the passkey or list nearby devices or recently connected devices, but also, this firmware allows you to configure output pins: PIO2-PIO7 and PIO10 (i.e.: to drive some LEDs or to set signals that do not need to switch state fast).

The PCM pins could be used for echo cancellation if we were using this module for audio transmission.

The SPI and USB pins can be used to change the firmware of the device. The firmware is not free software (neither the application to upload new firmware), and are both property of CSR. Buying CSR tools would allow us to use other pins and even allow us to modify the firmware to make use of the ADCs already available in the bluetooth device. But this would be interesting only if we did not mind using propietary software for applications in which the bluetooth device would work as a standalone device (i.e.: reading sensors, controlling signals, communicating with other devices…).

Module current consumption (real tested parameters, as the datasheet for this device is quite inaccurate):

- In AT mode, the consumption is under 3 mA.
- While pairing or searching for devices, the consumption is under 40 mA.
- After pairing, the current consumption is under 8 mA if there is no communication. While communicating, the consumption of the bluetooth device is about 20 mA.

1.1.2 AT mode

To enter AT mode simply:

1. Reset the device (input low level to pin 11 for at least 5 ms). Or power off the device.
2. Input high level to pin 34.
3. Input high level to pin 11 again (or high impedance). Or power on the device.

This way the module enters AT-mode with a well-known serial configuration, which is very convenient:

- 38400 baud rate.
- 8-bits data.
- 1 stop bit.
- No parity.

On the other hand, this way we need to reset the module (and therefore lose all bluetooth connections with other devices).

Alternatively, we can enter AT-mode as well without resetting the device by simply inputting high level to pin 34. This is quite convenient, but we will need to communicate with the device with the configured settings, rather than fixed, well-known settings. As long as we know the current serial communication, this is not a problem. Note that we can do this even if the device is paired with another device.

1.1.3 PCB-mounted devices

Many times the HC-05 comes already mounted on a PCB in a very reduced working system, just enough to use the serial communication. Note that:

- Although many of these mounted devices are compatible with 5 V logic levels, they do work just fine with 3.3 V.
- Some may have a small switch connected to pin 34. This is very convenient, as we can very easily enter AT-mode this way (just press the switch while you power on the device).

1.1.4 Configuration

Once you are in AT-mode, you should see the LED blinking less frequently, indicating you have successfully entered this mode.

Now, supposing you are connected to the bluetooth with a serial interface at `/dev/ttyUSB0` in your computer, first you need to set the appropriate serial configuration:

```
stty -F /dev/ttyUSB0 38400 cs8 -cstopb -parenb -echo
```

Then you can open a terminal to display the received data:

```
cat /dev/ttyUSB0
```

And send a simple command to test the connection:

```
echo -en "at+version?\r\n" > /dev/ttyUSB0
```

Which should result in the version being displayed in the first terminal.

A typical, very basic configuration could look like this:

```
echo -en "at+name=Theseus\r\n" > /dev/ttyUSB0
echo -en "at+uart=921600,0,0\r\n" > /dev/ttyUSB0
echo -en "at+role=0\r\n" > /dev/ttyUSB0
```

1.1.5 Connection

Checking the connection is easy. If you have a smartphone, you can install Bluetooth terminal[1] and connect to the device. Then, supposing you are still using a serial interface at `/dev/ttyUSB0` with a baud rate of 921600, no parity, 1 stop bit and 8 bits data:

```
stty -F /dev/ttyUSB0 921600 cs8 -cstopb -parenb -echo
echo -e "Hello world!" > /dev/ttyUSB0
```

If everything went well you should see a `Hello world!` message displayed on your phone screen! ^^

Note: Be sure to configure Bluetooth terminal application with ASCII input mode, no checksum and `\n` ending.

[1] https://github.com/Sash0k/bluetooth-spp-terminal

1.1.6 References

1.2 IE2-1024 Encoder Family

1.2.1 Introduction

The IE2-1024[1] is an incremental, two-channel magnetic encoder family with 64 to 1024 lines per revolution. Requires a 5 V supply voltage and outputs two digital square waves that can be used for the indication and control of both shaft velocity and direction of rotation as well as for positioning.

1.2.2 Wiring

These encoders have 6 wires with the following functions:

Pin	Function
1	Motor -
2	Motor +
3	GND
4	V_{dd}
5	Channel B
6	Channel A

Fig. 1: IE2 encoder wire functions.

Note: The connector may vary so, when in doubt, simply note that the wires are numbered from 1 to 6 starting from right to left, or better, starting from the wire that is partially colored in red.

1.2.3 References

1.3 Tetra locomotion

1.3.1 Introduction

A tetra design is composed with 2 motors, each driving two wheels. The motors have a pinion that drives two gears attached to the wheels.

Both the pinion and the gear must have the same module. There is a simple equation that relates the module, the number of teeth and the reference diameter of the gear:

$$M = \frac{D_r}{Z}$$

When working with gears there are 3 different diameter that we could refer to: the internal, the reference and the external diameters.

[1] https://fmcc.faulhaber.com/resources/img/EN_IE2-1024_DFF.PDF

Fig. 2: External, reference and internal diameters in a gear.

The external diameter can be calculated with:

$$D_e = M(Z + 2) = D_r + 2M$$

The resulting locomotion in a tetra design looks like this:

Fig. 3: Tetra locomotion design.

1.3.2 Restrictions

Note we have two important restrictions:

- The wheel must have a diameter greater than the external gear diameter D_e^{gear}. Otherwise the gear would be in contact with the floor.
- The wheel must have a diameter smaller than the reference diameter of the gear plus the reference diameter of the pinion $D_r^{gear} + D_r^{pinion}$. Otherwise the two wheels would be in contact with eachother.
- We will usually want to keep some space between the floor and the base of the robot (i.e.: the PCB).

In some cases, keeping that space between the floor and the PCB is not a problem:

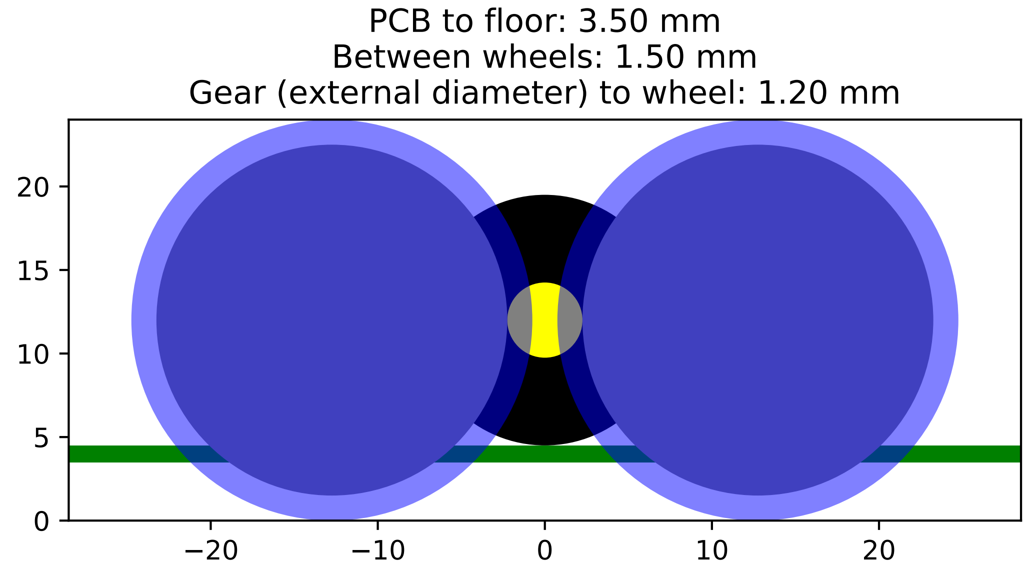
If the configuration leaves too little space between the PCB and the floor then we can always shift the motor a bit up:

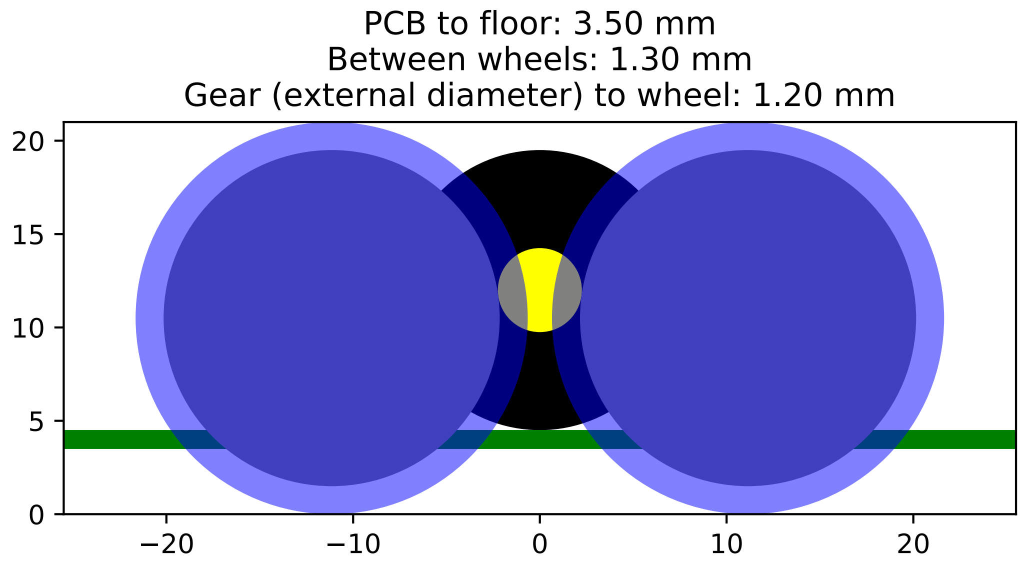
Note: Shifting the motor up will shift the center of mass of the robot a bit up as well. It will also compesate worse the forces generated because of the contact between the gears, although this might have a very little impact overall.

Of course, another alternative is to cut the base/PCB to allow the motor to pass through it. As a downside, cutting the PCB might not be easy and might not left much area connecting the front and the back parts of the robot.

1.4 Indices and tables

- genindex
- search


PCB to floor: 3.50 mm
Between wheels: 1.50 mm
Gear (external diameter) to wheel: 1.20 mm
20
15
10
5
0
−20
−10
0
10
20


PCB to floor: 3.50 mm
Between wheels: 1.30 mm
Gear (external diameter) to wheel: 1.20 mm
0
5
10
15
20
−20
−10
0
10
20

Index

A

abstract, 1
AT-mode, 2

B

bluetooth, 1

C

configuration, 3
connection, 3

E

encoder, 4

H

hc-05, 1

I

ie2-1024, 4
introduction, 4

L

locomotion, 4

M

module, 1
mounted, 2

R

restrictions, 5

T

tetra, 4

W

wiring, 4