PPM PWM 009

PPM Decoder and Servo Driver for Raspberry Pi and Nvidia Jetson

Summary

The Holbox PPM_PWM009 board provides RC signal decoding functionalities and servo control to single-board computers such as the Raspberry Pi or NVIDIA Jetson and Xavier boards.

Through an I2C connection, Pulse Position Modulation (PPM) values from an RC receiver can be decoded. Additionally, up to 9 Pulse Width Modulation (PWM) signals can be generated free running.

Detailed Description

Due to their general-purpose processor, single-board computers such as the Raspberry Pi are great for robotics, machine vision and real-time applications. However, these computers lack the microsecond-level timing capabilities required for decoding PPM signals from RC receivers and to generate PWM signals that drive multiple servos, brushless motors and LEDs.

PPM_PWM009 extends the capabilities of single-board computers by providing free-running Pulse Position Modulation decoding of RC signals and PWM/Servo signal drive. Such functionalities are added to your single-board through an I2C connection and an easy-to-use Python library.

All PPM_PWM009 boards are tested and ready to use. No need to solder, just plug and play!

Applications

- Semi-autonomous robotics
- Avionics, autonomous drones with manual override
- Industrial/intelligent systems with PID control
- RC receiver decoding (PPM)
- Motion control: servo and brushless motors
- Lightning control, LEDs

FAQS:

Why use the PPM_PWM009 for my system/project?

- This is the only plug and play option to read channels from PPM RC receivers.
- Easy integration to single-board computers such as the Raspberry Pi, Nvidia Jetson Nano and Nvidia Xavier boards.
- Leverage the efficiency of high-level programming. Single-board computers are Python compatible which is easier and faster to program than low-level languages such as Assembly or C, commonly used in microcontrollers.
- Powerful processing: Single-board computers have GHz clock signals and are directly compatible with machine learning routines commonly implemented in Python.

What are the advantages of single-board computers such as the Raspberry pi compared to microcontrollers?

- Easy to program (Python)
- Larger memory available
- Faster clock cycles (useful for complex computations)
- Direct integration with different kinds of sensors (cameras, accelerometer, gyroscopes, barometers, GPS)

How can I power the PPM_PWM009 board?

You can power this board by connecting a 5v power pin on your Raspberry Pi to the Vin pin on the board, connect their grounds too.

Power is provided to the RC receiver by connecting the + pin of the PPM_IN connection to your RC receiver. Don't forget to connect their grounds too.

If servos are used, it is recommended to use a different power supply than the one shared by the Raspberry Pi and PPM_PWM009 board. Connect this power supply to V_servo, sharing ground with the Raspberry Pi, PWM_PPM009 board and the RC receiver.

Pins and board diagram

Power Pins

- **Vin** – This is the power pin of the board. Connect it to a power supply in the range of 3.7V to 5.5V. Alternatively you can connect it to the Raspberry Pi 5V power pin (Same pin for Nvidia Jetson Nano.
- **GND (-)** – This is the ground pin of the board. Connect it to the same ground as your single-board computer. If servo motors are used connect them to the same ground too. All pins in the PPM_PWM_009 board shown as "**-**" are ground pins.

- **V_servo (V+)** – This is an optional power pin that will supply distributed power to servos and brushless motors. Leave disconnected if servos and brushless motors are not used. For usage stability and since motors drain large amounts of currents, use for **V_servo** a different power supply than the one used for **Vin**.

Control Pins

- **SCL** - I2C clock pin, connect to your single-board (i.e. Raspberry pi) I2C clock line. Can use 3V or 5V logic, and has a weak pullup to VCC

- **SDA** - I2C data pin, connect to single-board (i.e. Raspberry pi) I2C data line. Can use 3V or 5V logic, and has a weak pullup to VCC

- **ADDR** – Connect jumper cable if need to change I2C address: Default Address = 41. Alternative address = 51. (Jumper connected)

- **R1 and R2** – ***Optional, not required in most cases***. Connect two 4.7KOhm resistors to these pins if very long I2C connections are used.

Input Pins

- **PPM_IN** – There is one PPM input pin that can be connected to the PPM signal pin of an RC receiver to decode up to 14 channels simultaneously.

Output Pins

- **PWM Pins** – There are 9 PWM output ports. Each port has 3 pins: V (+), GND (-) and the PWM signal output (s). Each PWM runs completely independently but they must all have the same PWM mode. Either Servo or LED - You cannot use half for LEDs and half for servos (50Hz).

Output pins can also be used to drive Leds. Max current per pin is 10mA.

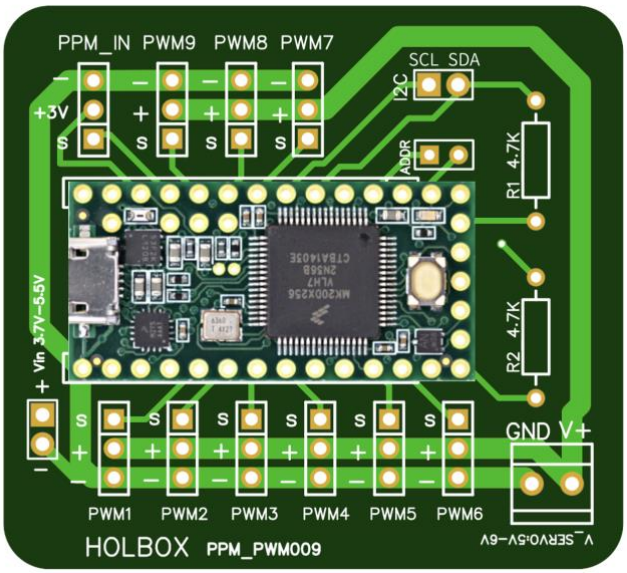


Figure 1. Board Pins

Connection Instructions

Below are the instructions that explain how to connect the Raspberry Pi to the PPM_PWM009 board. Raspberry Pi and Nvidia Jetson Nano share the same GPIOs, so connections are the same.

Read from RC receiver (PPM Decoding)

- *Raspberry Pi connection to PPM_PWM009:*
  - 5V power pin to V+. This provides power to the PPM_PWM009 board
  - GND to (–)
  - SDA (pin 3) to SDA
  - SCL (pin5) to SCL

- *RC receiver connection to PPM_PMW_009:*
  - PPM signal pin to PPM_IN (s) pin
  - RC (+) pin to PPM_IN (+). This provides power to the RC receiver. Do not connect another power supply to the RC receiver.
  - RC ground (-) to PPM_IN (-)

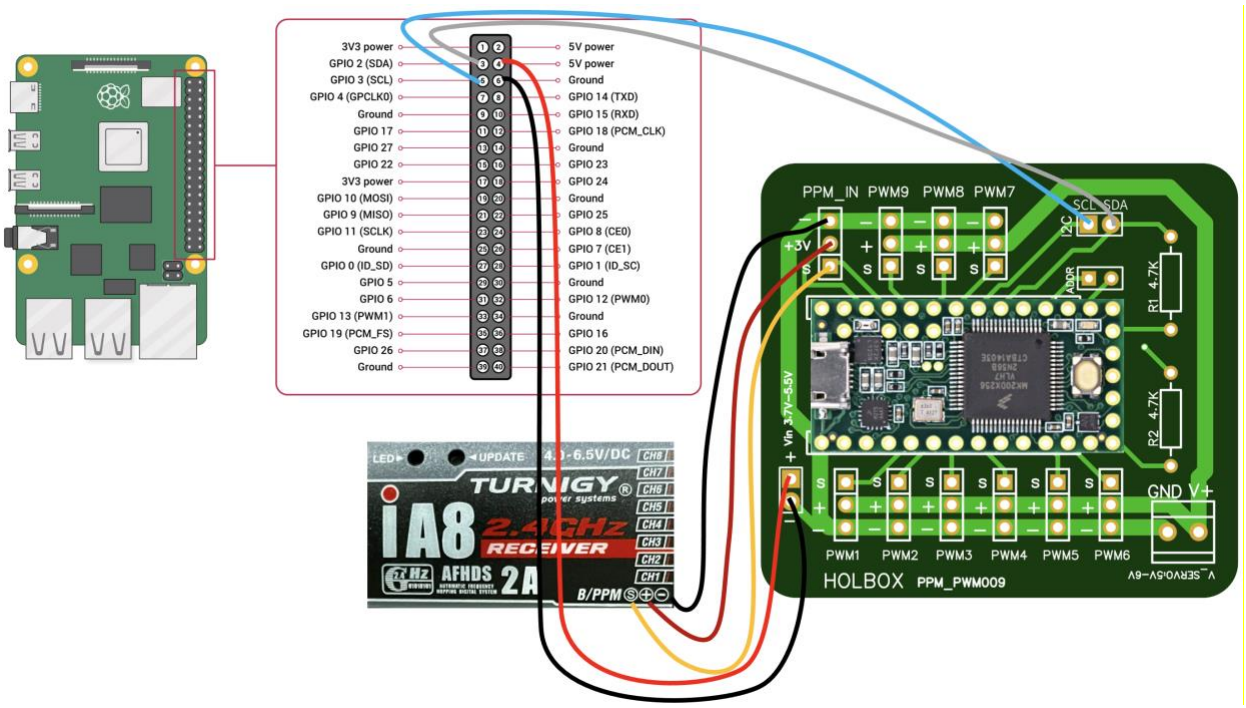


Figure 2. Connection diagram for RC read

Servo/Brushless Control

- *Raspberry Pi connection to PPM_PWM009:*
  - 5V power pin to V+
  - GND to (–)
  - SDA (pin 3) to SDA
  - SCL (pin5) to SCL

- *PPM_PWM_009 connection to Servo/brushless motors:*
  - Connect PWM (+), (s) and (–) pins to the servo. Signal pin is always in the middle on servos and brushless motors. Up to 9 PWM outputs
  - Connect V_servo (+) and GND to a 5V-6V power supply. It is recommended to use a power supply different to the one powering the Raspberry pi since motors draw high currents

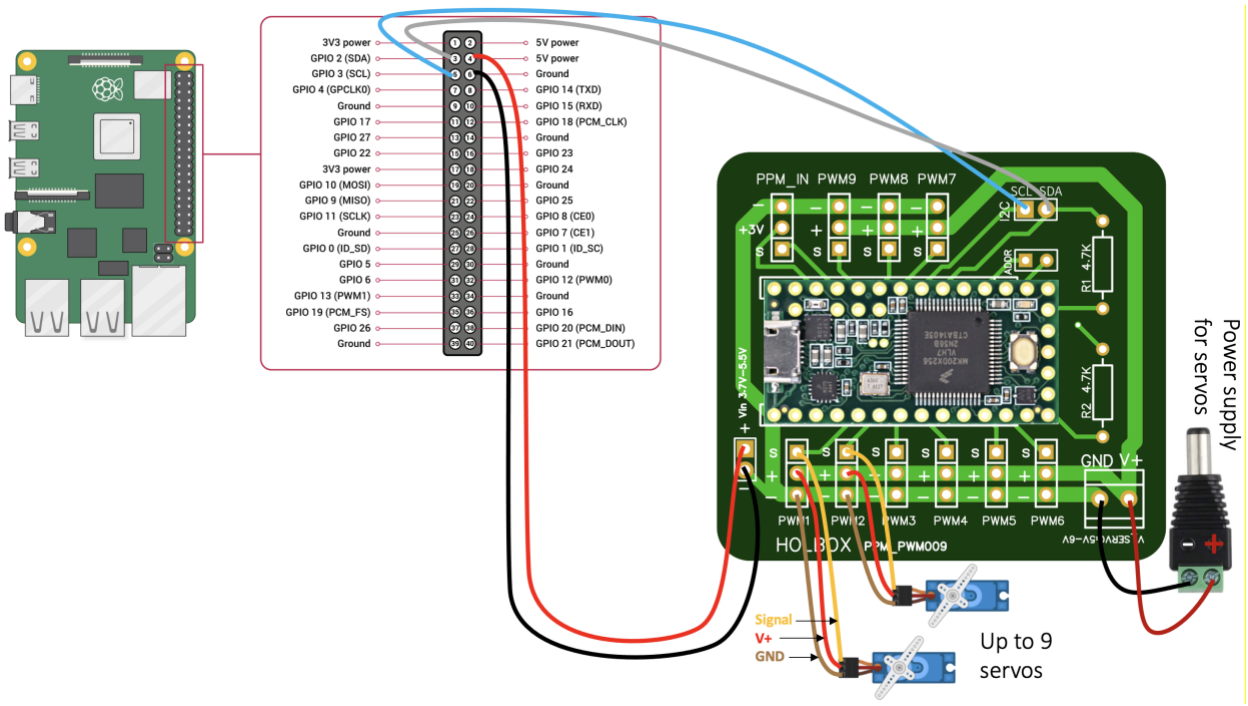


Figure 3. Connection diagram for servo/brushless control

PPM Decoding and Servo/Brushless Control

- *Raspberry Pi connection to PPM_PWM009:*
  - 5V power pin to V+
  - GND to (–)
  - SDA (pin 3) to SDA
  - SCL (pin5) to SCL

- *RC receiver connection to PPM_PMW009:*
  - PPM signal pin to PPM_IN (s) pin
  - RC (+) pin to PPM_IN (+). This provides power to the RC receiver
  - RC ground (-) to PPM_IN (-)

- *PPM_PWM_009 connection to Servo/brushless motors:*
  - Connect PWM (+), (s) and (–) pins to the servo. Signal pin is always in the middle on servos and brushless motors. Up to 9 PWM outputs
  - Connect V_servo (+) and GND to a 5V-6V power supply. It is recommended to use a power supply different to the one powering the Raspberry pi since motors draw high currents

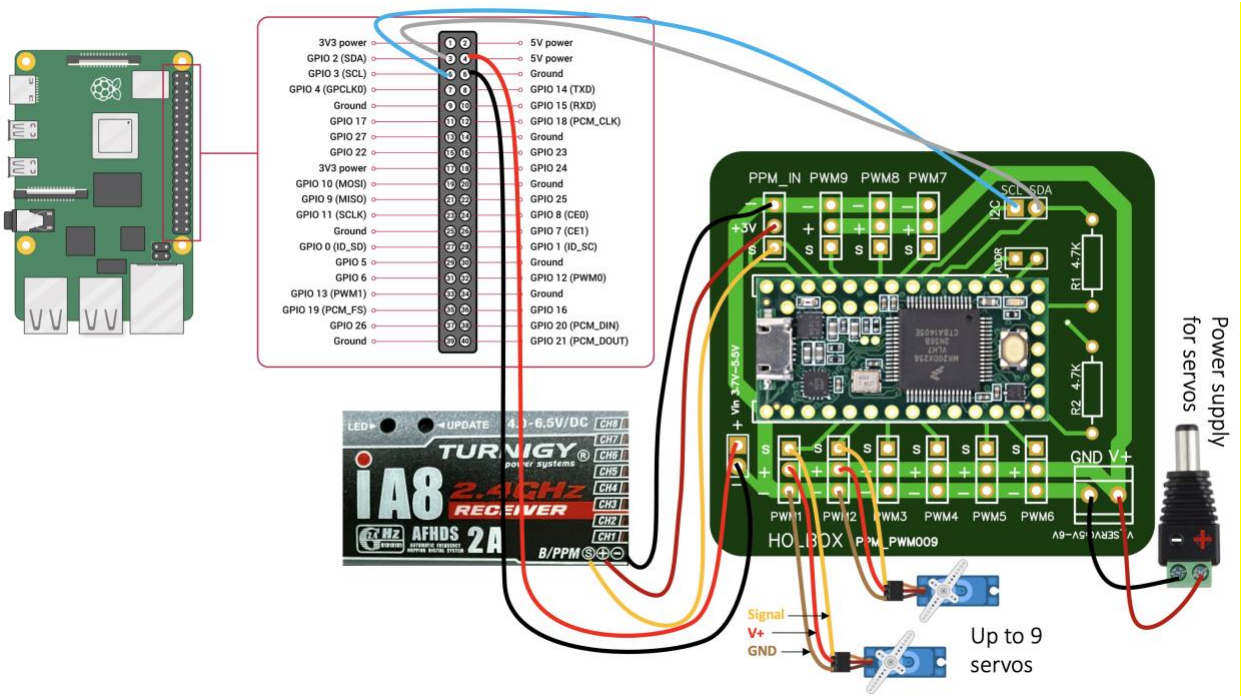


Figure 4. Connection diagram for RC read and servo/brushless control.

Change I2C address (Optional)

If needed, to change the I2C address of your PPM_PWM009 board (default is Address 40), connect a jumper cable between the two ADDR pins as show in Figure 5. This will change the I2C address to 50.

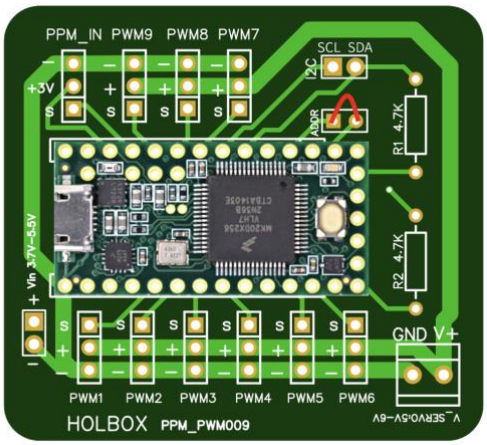


Figure 5. Jumper cable to change I2C address.

Code:

PPM_PWM009

I2C is a very commonly used standard designed to allow one chip to talk to another. Since the Raspberry Pi supports I2C we can connect it to a variety of I2C capable chips and modules.
The I2C bus allows multiple devices to be connected to your Raspberry Pi, each with a unique address, that can often be set by changing jumper settings on the module. It is very useful to be able to see which devices are connected to your Pi as a way of making sure everything is working.

Enable the I2C connection

On the Rasberry Pi open the terminal and type:

```
sudo apt-get install -y python-smbus
sudo apt-get install -y i2c-tools
```

The, run
`sudo raspi-config`

and follow the prompts to install I2C support for the ARM core and linux kernel

- Go to Interfacing Options (On older versions, look under Advanced)
- Select I2C
- Choose Enable
- Reboot the board with: sudo reboot

-
Finally test the I2C connection typing the following in the terminal:
`sudo i2cdetect -y 1`

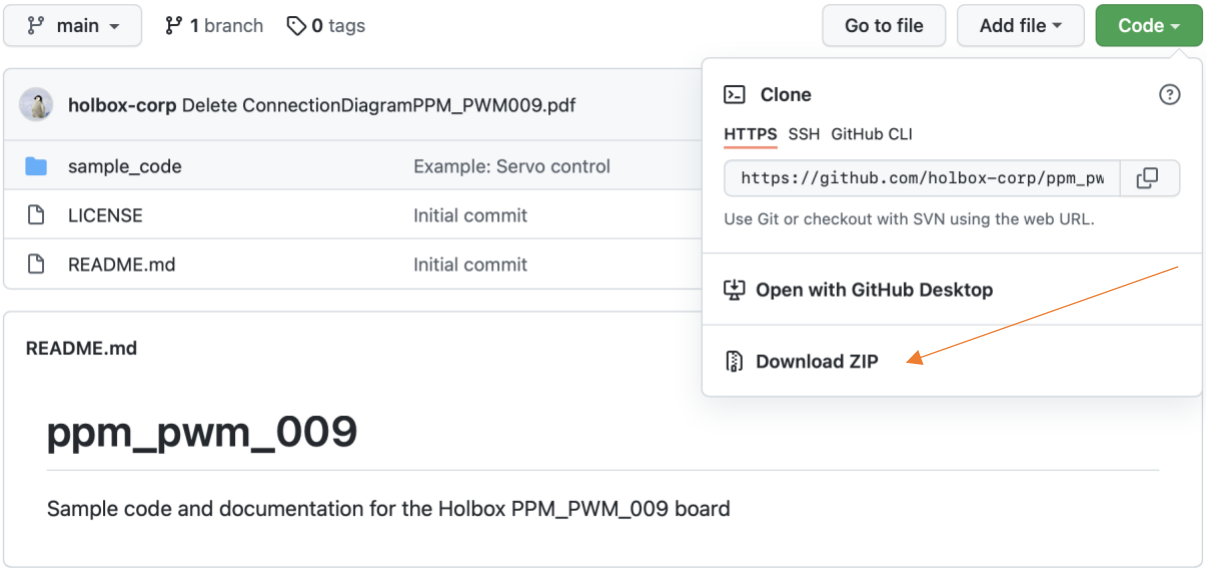
Note: If you are using the Nvidia Jetson Nano, I2C comes enabled by default.

Download the PPM_PWM library

Go to:

- https://github.com/holbox-corp/ppm_pwm_009

Click Code, download zip



Unzip and go to the sample_code folder

Copy the holbox_ppm_pwm_lib.py to your project folder on the raspberry pi. This library must be imported when using the PPM_PWM009 board. To import at the beginning of your script, include it as:

```
import holbox_ppm_pwm_lib
```

Code to read from RC receiver

After downloading the PPM_PWM library, follow the connection diagram of figure 2.

From the sample_code folder open and run: readRC_example.py on your Raspberry pi. Adjust sample code as needed.

Code to drive servos or leds

After downloading the PPM_PWM library, follow the connection diagram of figure 3.

From the sample_code folder open and run: driveServo_example.py
on your Raspberry pi. Adjust sample code as needed.

Code to read from RC receiver while driving servos and leds

After downloading the PPM_PWM library, follow the connection diagram of figure 4.
From the sample_code folder open and run: readRC_and_driveServo_example.py
on your Raspberry pi. Adjust sample code as needed.

Specifications:

1x PPM Input Pin (Up to 14 Channels, 10bits per channel, 50Hz update rate*)
9x configurable Servo outputs (Configurable either as Servo* or PWM driver, 10bit per output pin)
I2C Data Transfer Optimized for high speed (<3ms)
2x I2C addresses. Default Address 40, Alternative Address is 50.

*PPM and PWM signals are standardized to a 50Hz update rate. Reading at a higher rate will return the same values as the previous sample. Therefore, for optimal real-time control, the recommended cycle time in your Python script is 20ms**. This is fast enough to update the signal of any dynamical system such as quadcopters, cars, planes and robots.

**Actual calculations and data-reading from PWM_PPM009 and sensors occur much faster in the single-board computer (GHz clock signal).

Compatible PPM transmitters with receivers:

- Your RC receiver MUST support PPM format.

- Example of PPM receivers:
    - Flysky FW-i6
    - Turnigy 9x 9ch

Make sure to enable PPM (S-bus) in your transmitter settings.