Manual for 8 Channel PPM Encoder (v2), Firmware: v2.3.16

Written by Julian Oes, 03/01/2013

Table of Contents

Contents 1
Overview 1
Acknowledgment 1
Soldering 2
Wiring 2
Modes 3
  Normal mode 3
  Radio Passthrough mode (mux) 3
  Fail-safe 4
JP1 Solder Jumper 4
Updating firmware / Reprogramming 4

Overview

The PPM encoder allows to encode up to 8 PWM (pulse width modulated) signals into one PPM (pulse pause modulated) signal. This allows you to use any R/C receiver and a microcontroller that supports PPM (e.g. the PX4FMU).

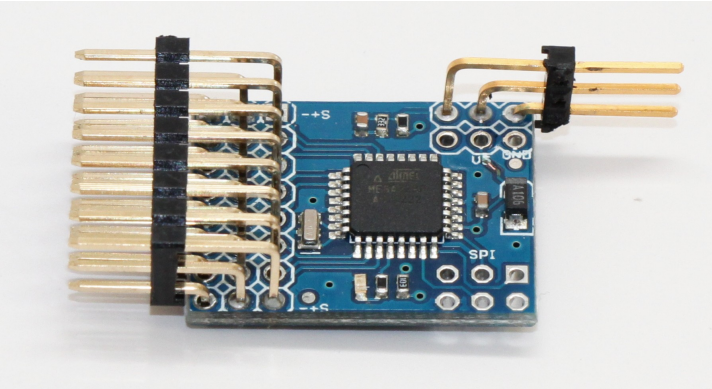
The design is based on the Atmega328P using and an external 16Mhz resonator.
By default negative pulse PPM is encoded. In order to change to positive pulse PPM, the firmware needs to be changed, compiled and reprogrammed as explained in the wiki.

Acknowledgment

The hardware design was originally based on the Atmega PPM Encoder Board by Paparazzi.

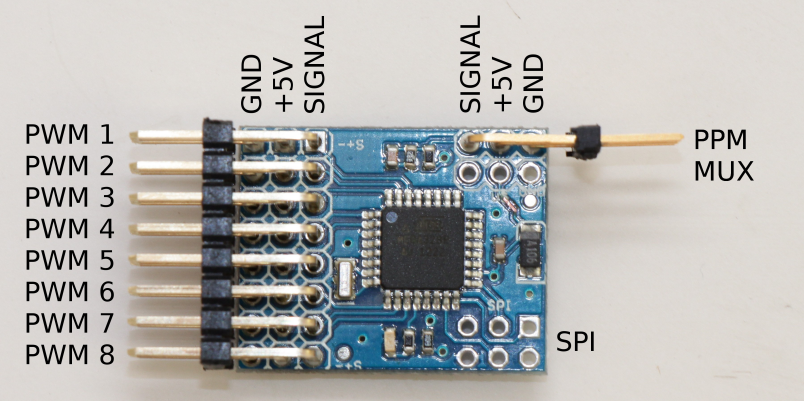
The PPM Encoder now uses the ArduPPM firmware, replacing the previously used Paparazzi PPM Encoder firmware. The new ArduPPM firmware has been designed from scratch to enhance performance and robustness, and to better accommodate our product needs now and in the future.
It has been written by John Arne and Olivier Adler and is available in the ardupilot repository and licensed under the GNU GPL v3.

Soldering



Use a 3x8 right angle pin header for the PWM connections and a 3x1 right angle pin header for the PPM connection (or a 3x2 right angle pin if mux connection is needed).

Wiring

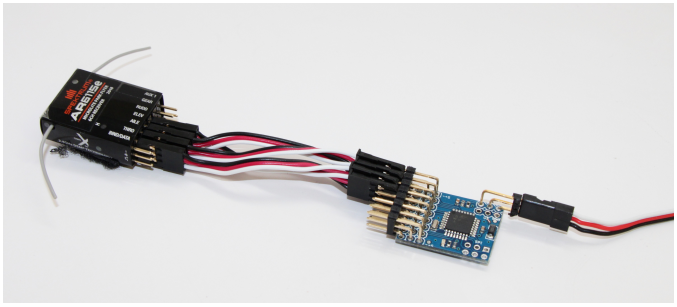


Connect the PWM signals coming from your receiver to the inputs. The “S+-” next to the pins indicates the polarity.
Make sure you use the channel assignments as shown below (details here)

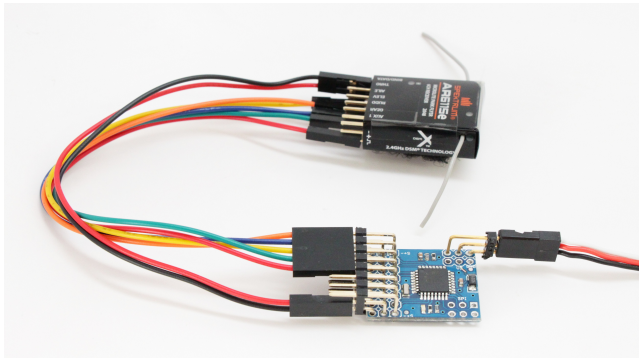
Channel 1	Roll
Channel 2	Pitch
Channel 3	Throttle
Channel 4	Yaw
Channel 5	… (Default ArduCopter: Flight Mode)
Channel 6	...
Channel 7	...
Channel 8	… (Default ArduPlane: Flight mode)

Please note that throttle must be on channel 3 in order for the failsafe to work properly!

Either connect every channel separately using servo jumper cables:



Or use a 2 pin jumper cable for +5V and ground, and a 5 or 6 pin jumper cable for the signals:



Modes

Carefully check that your radio system is working flawlessly before flying.
If you see the blue status LED blinking very fast almost continuously this is an indication that something is wrong in the decoding.

Normal mode

The blue LED is used for status reports :

- slow to fast blinking according to throttle channel position
- very fast blinking if servo all channels have been lost, or if the throttle channel (ch3) has been lost

Radio Passthrough mode (mux)

This feature is only available, when the PPM Encoder for ArduPlane firmware is used:
→ e. g. ArduPPM_v2.3.16_ATMega328p_for_ArduPlane.hex

This mode is described as hardware failsafe in ArduPlane terminology.
Radio Passthrough mode is trigged when servo channel 8 > 1800 µs.

When the passthrough mode is active, the Mux output on JP5 is enabled, so that an external bypass circuit can be switched on.

The blue LED has different behaviors in passthrough mode:

- If throttle position < 1200 µs, status LED is off
- If throttle position > 1200 µs, status LED is on

Fail-safe

- If a receiver servo channel is lost, the behaviour depends on the channel:

Channel 1	Roll	Set to center (1500 µs)
Channel 2	Pitch	Set to center (1500 µs)
Channel 3	Throttle	Set to low (900 µs)
Channel 4	Yaw	Set to center (1500 µs)
Channel 5	...	Remain at last value
Channel 6	...	Remain at last value
Channel 7	...	Remain at last value
Channel 8	...	Remain at last value

- If all channels are lost or the throttle channel is lost, the throttle signal will be set to low (900 µs) and this is an indication for the ArduCopter/ArduPlane/ArduRover software that something went wrong. Depending on the settings and parameters, this might initiate a fail-safe scenario like RTL.
  Before flying make sure to setup and test the fail-safe (see wiki for ArduCopter and ArduPlane).

JP1 Solder Jumper

The jumper JP1 has been included to provide some flexibility in the way you power the receiver, servos and PPM Encoder. In standard it is connected, solder is applied.

JP1 in place: The +5 Volts from the receiver and the PPM output are connected. The receiver will draw the power from whatever autopilot board is attached to the PPM output.
(standard setup)
JP1 removed: The +5 Volts from the Servos and the PPM output are separated.
The receiver needs an separate power source.
Note that failure to remove JP1 can damage the autopilot's +5v switching regulator when too much current is drawn.

Updating firmware / Reprogramming

The PPM Encoder comes with the PPM encoder firmware pre-programmed, and most users will never need or want to modify it.
However, some users may want to get into the code to change the way the PPM Encoder interprets RC signals or may want to update to the latest version.
Some rare users did report receiver compatility problems with the old version (before ArduPPM). For most cases, ArduPPM did solve them.

Instructions on how to flash a new firmware can be found in the wiki.