Malenki-Nano Programmable Six-Channel Receiver (Rx) & Triple Electronic Speed Controller (TESC)

Introduction

The Malenki-Nano is a 6-channel and integrated radio receiver (Rx) and triple electronic speed controller (TESC) in a single tiny PCB intended for builders of small robots, where space and weight are important. It was designed for “UK Antweight” and US Fairyweight robots, which have a maximum weight of 150g.

Safety First

Chemicals / Lead / fumes / dust

- It is NOT 100% Lead-free due to some components coming from another manufacturer.

Precautions:

- Wearing surgical gloves while handling any electrical components helps prevent excess contact
- After handling any electronics, wash hands before eating, etc
- Do not lick any part of any electronics
- Do not allow your child / pet to handle or lick any electronics

Fumes

- Even lead-free soldering produces fumes
- Always do soldering in a well-ventilated area, away from food preparation
- Use a fume extractor if possible

Dust

- PCBs contain small fibres (usually glass fibre) which are bad to breathe.
- If cutting, sanding or filing PCBs, do it in a well ventilated area away from food preparation areas (outside is best)

Fire hazard

Lithium Polymer or other types of batteries can start fires in the event of an electrical fault.

- Do not leave any robot or test-setup powered on unattended (even e.g. when the transmitter is off), electrical faults can happen at any time
- Take care to observe correct battery polarity (Malenkis do NOT have reverse polarity protection)
- Check (visually or electrically) for electrical shorts before connecting the battery.
- Use the right kind of charger
- Do not attempt to recharge damaged or swollen batteries
- Dispose of damaged batteries safely (e.g. not in a bin with waste paper)

Specifications

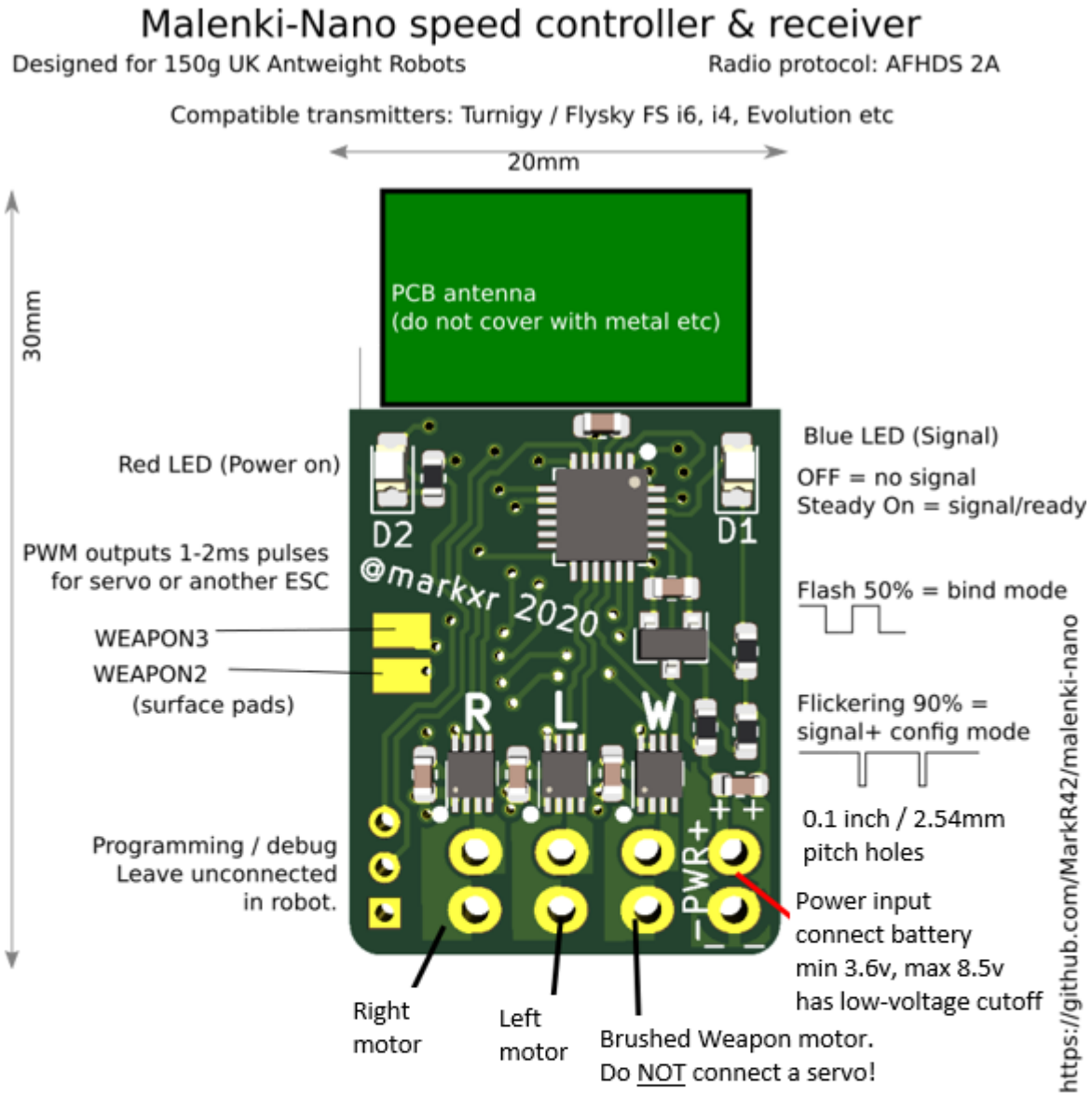
- Radio system: AFHDS 2A
- Input power voltage: 3.6 - 8.5 volts
- Brushed ESC channels: 3 (all reversible)
- Maximum current per channel: 1.8A
- PWM channels: 2 (independent of brushed channels)
- Telemetry: battery voltage, where supported by transmitter.

Change history

Firmware 2.0 - August 2021

- Enabled “servo stretcher” feature
- Removed temperature telemetry

Overview diagram



Channel mapping

1 = Steering
2 = Throttle
3 = WEAPON or WEAPON2
4 = WEAPON2 or WEAPON
5 = Invert / Config pulses
6 = WEAPON3 (PWM)

Configuration commands (enter config by 5 sec power w/o Tx)
- Assign a switch to Ch5 that is Low by default
- Send below number of High-Low pulses while in config.

2 = Toggle invert left-hand channel
3 = Toggle invert right-hand channel
4 = Toggle invert weapon channel
5 = Reset configuration to defaults (keep bind)
6 = Toggle mixing disable. (default = enabled = steer/throttle)
7 = Factory reset, including transmitter binding.
8 = Swap channels 3 & 4
9 = Disable braking (toggle)
10 = Servo stretching (toggle)

Getting started

Connecting power and motors

Connect the power and motors as shown in the overview diagram.
It is recommended to connect the left & right channels correctly, otherwise you will become confused with the mixing!
Power:

- Minimum 3.6 volts.
- One of the following is recommended:
    - 1S lipo pack (3.7v nominal)
    - 2S lipo pack (7.4v nominal)
- Maximum 8.5 volts. This means you cannot use a 3S Lipo!
- During testing, a bench supply or USB power bank is ok too (5v is within the acceptable range)
- Observe correct polarity, + and - are marked on both sides of the PCB!

Motors:

- Use motors with a stall current of less than 1.8A at the supply voltage.
- “N20” motors are usually suitable (N10 or N30 might be too)
- Connect two (or more) motors in parallel if their combined stall current is <= 1.8A - two N20s is usually ok.
- The weapon channel is electrically the same as the left & right.

Transmitter binding

Transmitters and receivers need to “bind” so that they know each others’ unique ID, so that multiple transmitters of the same type can be used within radio range and don’t get mixed up.

First, put the receiver into bind mode using one of these methods:
Either:

- If the unit is unbound (new, factory setting), it will enter bind-mode automatically on power on.
- If the unit is already bound to another transmitter, it will enter bind mode about 90 seconds after power on, **only if no signal is received from the bound transmitter.**

The receiver bind-mode is indicated by a flashing blue LED (D1).

Different transmitters have different procedures for entering bind mode, follow the instructions on your transmitter.

Once the receiver and transmitter are both in bind mode at the same time, binding should happen within 10 seconds. At this point the flashing blue LED will stop.

- On telemetry-transmitters, bind mode is usually automatically exited after telemetry is received, with a bleeping noise.
- For non-telemetry transmitters, follow the instructions to exit bind mode (usually power off & on)

Entering driving mode

To enter driving mode:

- Centre the stick on the brushed weapon channel (Channel 4 or channel 3) if it does not automatically centre
- Move the throttle stick a little forwards or backwards. The robot should drive (or motors turn)

If the motors do not run, try centering the weapon channel better. Tip: If not using the weapon channel, map a different stick or button, so that it is automatically centred.

Transmitter setup

The default configuration on most “Mode 2” stick-style transmitters will be to have driving on the right stick and weapons on the left.
This can usually be changed by modifying menu options, or in some cases, connecting to a computer to modify firmware settings.

“Channel 5” is not always mapped by default, it may be useful to map channel 5 to a switch for the configuration and invert features.

Improving robustness

It is recommended that the module should be wrapped in a non-conducting film layer, in order of preference:

- Heat shrink tube, preferably transparent (so LEDs can be seen)
- “Kapton” tape
- Some other type of electrical tape

Do this after all electrical connections are complete and it is tested and working.

This will reduce the chance of a short circuit during operation.

More features

Channel mapping

Channel 1 = steering
Channel 2 = throttle (forward / back)
Channel 3 = weapon2 (PWM output) or brushed weapon **
Channel 4 = brushed weapon or weapon2 (PWM output) **
Channel 5 = Invert throttle (when in driving mode) or configuration (when in configuration mode)
Channel 6 = weapon3 (PWM output)

** = Maybe be modified in configuration.

PWM Outputs

Two solder pads shown on the diagram will provide PWM outputs of pulses 1000-2000 microseconds (nominally, unstretched), as usually used to drive servos.

- Pulses are only sent when a signal is received and in driving mode.
- Pulse width is always the value sent in the digital packet (AFHDS2A packets contain the pulse width in microseconds) - to the best precision available on the PWM hardware.
- Pulses are sent at 3.3 volts only. In testing, this has been enough to work servos.

Firmware configuration mode

A special mode exists where some firmware settings can be changed. These settings are stored in non-volatile memory, so they don't need to be set every time after power-on.

To enter configuration mode:

- Turn the transmitter off
- Turn the receiver off
- Turn the receiver on
- Wait at least 5 seconds
- Turn the transmitter on

The receiver will then be in configuration mode until the weapon channel is centred and the throttle moved. In configuration mode, the blue LED (D1) flickers slightly with approximately 90% duty-cycle.

In configuration mode, commands are received on channel 5. **Channel 5 must be set high and then low to send a pulse**.

To send a command, press the switch N times, and then wait for 2 seconds.

Commands are:

Number of pulses	Function	Default value
1	No effect	
2	Toggle invert left-hand channel	Not inverted
3	Toggle invert right-hand channel	Not inverted
4	Toggle invert weapon channel	Not inverted
5	Reset configuration to defaults - but keep bind settings.	
6	Toggle mixing disable. After mixing is disabled, channels 1&2 become left and right instead of steering/ throttle.	Enabled
7	Factory reset, including transmitter binding.	
8	Swap channels 3 & 4	varies
9	Disable braking (toggle)	Enabled
10	Servo stretching (toggle)	Disabled

Invert or toggle commands can be repeated to undo the effect.

Driving invert

While in driving mode, channel 5 behaves as an “invert switch” and will reverse the throttle channel. This is to allow easier driving of inverted robots, e.g. a pushing robot which has become flipped over.

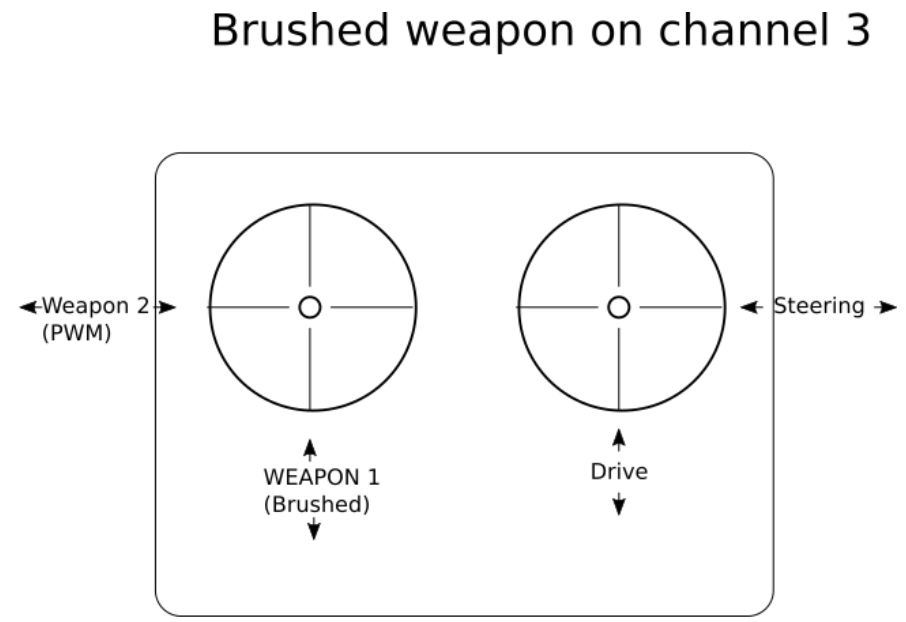
Swapping weapon channels mapping

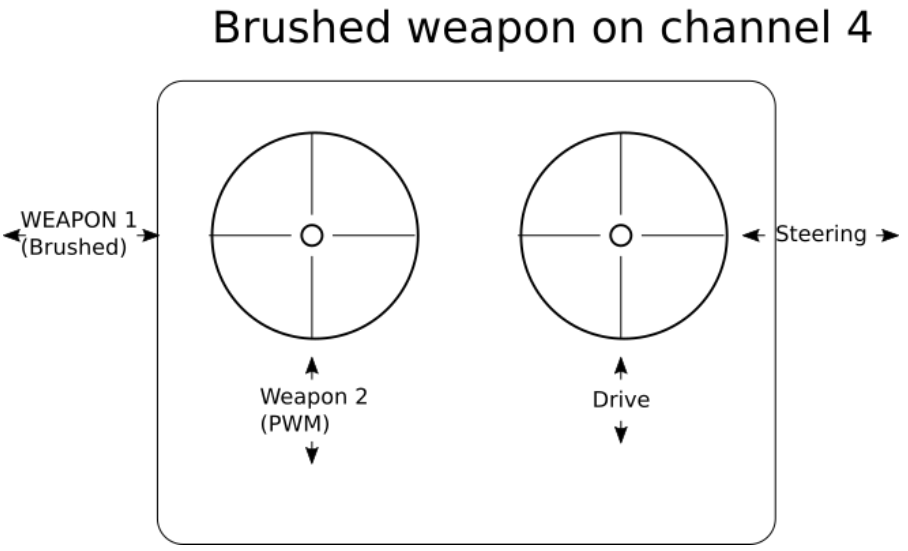
The configuration command (8 pulses, see above) to swap channels 3 & 4 is intended to make it easier to control robots using some types of weapon (e.g. spinners) on transmitters which don’t allow swapping.

For example, a “Mode 2” controller has the left-hand stick controlling channels 3 & 4, with channel 3 on the vertical axis which does not self-centre (typically used for throttle on drones or fixed-wing).

This is inconvenient for some setups, for example, a spinner where the non-centring channel would be used for the spinner ESC.

The picture below shows the effect on a “mode 2” transmitter.





Braking disable

If you really want, you can disable braking with this command. When braking is disabled, the drive motors will go “open circuit” when not running, so the robot can coast for a short time.

Servo Stretcher / extender / doubler

This option will enable a firmware feature to increase the difference in minimum / maximum pulse widths on the PWM servo outputs.

To better understand servos, read Sparkfun’s Tutorial.

WARNING: Enabling this option may reduce the life / damage your servos! Avoid continuously stalling servos as they may overheat and fail. Many servos have mechanical endstops inside the servo which cannot be reached with “normal” pulse widths but can be reached with extended pulse widths.

The servo stretcher option works by taking the input pulse width - which is sent digitally from the transmitter in “microseconds” - subtracting 1500, doubling the value, then adding 1500.

Pulse widths in microseconds

	Minimum pulse	Middle position	Maximum pulse
Without servo stretcher	~1000	1500	~2000
With servo stretcher	~500	1500	~2500

This feature enables servos to travel further than their nominal maximum - often up to hardware end-stops.

Notes:

- The servo stretcher affects BOTH PWM outputs or neither.
- It is probably not useful to use the servo stretcher with other ESC (e.g. brushless) - as it may not operate correctly with pulse widths outside the normal range
- It is advisable to reduce the transmitter endpoints if using the servo stretcher to reach the desired positions without too much stalling.
- If servos are operated continuously stalled against mechanical endstops, overheating / damage might occur

Automatic shutdown on low battery

The receiver detects the battery voltage on startup. If it looks “correct” for a 1S or 2S lipo pack, it will enable automatic shutdown. This is to protect the batteries from over-discharge.

Automatic shutdown happens when the pack voltage goes below 3.15 volts per cell for 4 seconds continuously.

After automatic shut down, the radio & all motors are turned off. Telemetry will be lost & the LEDs turn off.

The board will still use some power after automatic shutdown, so it should not be left in this state, it will eventually over-discharge the battery anyway.

If the voltage is outside the normal range at power-on (e.g. operating off USB power), automatic shutdown is disabled.

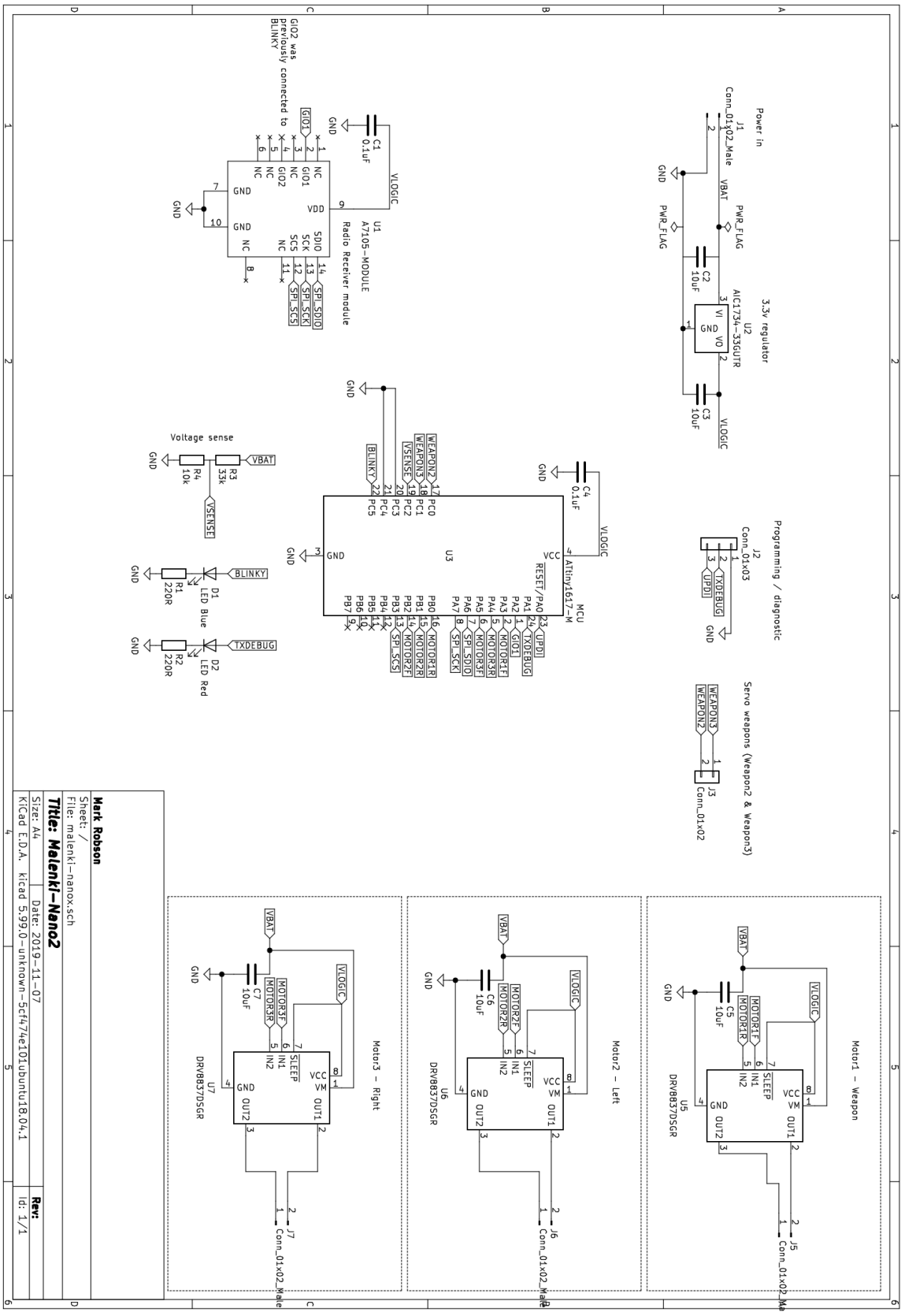
Telemetry

The Malenki-Nano receiver also sends telemetry packets back to the controller. At the time of writing, this sends the following information:

- Battery voltage

The temperature sensor was removed in firmware 2.0

Electrical schematic



Troubleshooting

Blue light turns on, but motors won't run / cannot drive

- The receiver is in configuration mode. Try to centre the brushed weapon channel (channel 3 or 4), then operate the throttle (channel 2)
- Channels incorrectly assigned on controller? Check the setup.

Blue light turns on, but immediately shuts down / resets when trying to drive

- Insufficient battery power
- Motors are drawing far too much current
- Bad connections / high resistance power connection
- Motor short circuit

Check the resistance of a motor channel (powered off) - anything less than ~ 4 ohms is probably bad.

Red light flashes regularly, no signal

This usually indicates a dead radio module. The receiver will need replacing (you could just replace the MD7105 radio module, but that is difficult)

FAQ

What transmitters is it compatible with?

- Turnigy Evolution *
- Turnigy i6 *
- Turnigy i6x
- Turnigy i4x (very cheap; no telemetry)
- I9x ?
- Multiprotocol transmitters?

*= tested

It requires 6 channels for full functionality.

How do I stop it from entering bind mode?

Bind mode is entered automatically after a delay if no signal is received.

To prevent automatic bind mode, just turn on the bound transmitter during the first minute after power on - if you don't want to drive, just turn it off again and the Malenki will not re-enter bind mode.

I expect this is sufficient to stop a situation where another robot builder is frantically trying to rebind their robot in the "pits" area e.g. after a fault with their main transmitter, and accidentally binds to a robot just entering the arena.

Why does it not enter driving mode immediately?

Because on the FS-i6 and Evolution transmitters, the weapon channel will be in “full reverse”- which means e.g. a flipper or axe weapon motor will be stalling immediately on power-on. This could result in overheating.

Also it means we can use the same channel for invert and configuration. An earlier version of the speed controller (w/o rx) used channel 7 or 8 for this. Some transmitters only have 6 channels, which is enough.

How much does it weigh?

Approximately 2.6 grams without wires.

Does it have braking?

Yes, the left/right drive channels apply “centre braking” automatically when receiving a signal and not driving. At the time of writing, the weapon channel does not use braking but it could be added in a future firmware update.

What is that huge oval-shaped metal can on the radio module? Can you make a version without it?

It is a crystal oscillator which runs at 16mhz to drive the radio gubbins. It is part of the radio module which I did not design.

I have tried replacing this crystal with a smaller one - it is tricky, but possible, and it actually works.

I have not found a compatible crystal which is flatter and also fits the footprint on the PCB, so some dead-bug wires are required.

Why is there no 5v BEC (Battery elimination circuit) ?

The internal voltage regulator runs on 3.3 volts because that is the correct voltage for the radio module. So there is no 5v regulator present on the PCB.

But also, it’s not usually required.

Is it really open source? Am I allowed to make my own?

Yes, and yes.

The source code for the firmware, and the design files for the hardware are here:
https://github.com/MarkR42/malenki-nano

Can you make a version for DSMX or some other protocol?

Probably not very easily. As I understand it, the Cypress radio chips that those radios used have been discontinued, but there might be newer replacements. It’s a “future work” :)

What does Malenki mean?

It’s the Russian word for “small”, маленький, it’s my transliteration.

Acknowledgements (by Mark R. - designer)

I would like to thank the following people in no particular order:

- User “essele” on EEVBlog forum - for https://github.com/essele/kicad_jlcpcba -which generates the BOM and location data for JLCPCB
- Flemming Frandsen - for https://gitlab.com/dren.dk/kicad-util generates mouse-bites for panelising PCBs
- “MrAardvark” for pyupdi.py https://www.avrfreaks.net/users/mraardvark programs the microcontrollers.
- Deviation TX firmware authors, and especially “goebish” on deviationtx.com forum who reverse engineered the AFHDS2A protocol and provided me with assistance in getting telemetry working
- Reading Hackspace directors and members for making an excellent place to do development work and giving technical help.
- Alasdair “Suvv” Sutherland for general encouragement
- Enzo from Reading Hackspace for lending me a transmitter during lockdown
- Joe Brown of the Bristol Bot Builders for being a beta tester and agreeing to stock them on the shop
- Ben (from Team Panic) for video reviews / demos
- Scott Siegel for amazing levels of enthusiasm and ideas
- All the UK Antweight builders for being creative & generally cool, and being formidable opponents!