

# D-O Control & Power Board

---

## System Documentation v2.2

D-O Control & Power Board

Complete user guide.

Version 2.2 — 2026-04-23

## Disclaimer and Liability Notice

D-O Control & Power Board is a do-it-yourself project. Building, wiring, configuring and operating the system involves work that, if done incorrectly, can cause damage to property or persons.

By using this documentation, hardware, and software you acknowledge:

- You are solely responsible for all work on your system.
- You have the required basic knowledge or seek qualified help.
- You verify voltage, polarity, fuses, connectors before every power-on.
- You only use components matching the application.
- You comply with local regulations.

**The author and all contributors accept no liability** for damage caused by incorrect wiring, wrong components, improper operation, or insufficient safety measures.

The documentation and software are provided **as is** without warranty of any kind. If in doubt, seek qualified help before working on the system.

# Table of Contents

<b>Disclaimer and Liability Notice</b>	<b>2</b>
<b>Table of Contents</b>	<b>3</b>
<b>Introduction — What is D-O Control &amp; Power Board?</b>	<b>7</b>
Key Features . . . . .	7
<b>Quick Start</b>	<b>8</b>
<b>Choosing a Board &amp; Sketch</b>	<b>10</b>
The two boards at a glance . . . . .	10
Available sketches . . . . .	10
Recommended combinations . . . . .	11
Upgrade note for v3.4.1 . . . . .	11
<b>Uploading the Sketch</b>	<b>12</b>
Required libraries . . . . .	12
Arduino IDE walkthrough . . . . .	12
Command-line alternative . . . . .	12
<b>IMU Calibration</b>	<b>13</b>
How to run calibration . . . . .	13
When to re-calibrate . . . . .	13
<b>Transmitter Setup (FlySky)</b>	<b>15</b>
1. Choosing the receiver (iBus output saves cabling) . . . . .	15
2. Binding the receiver . . . . .	15
3. Channel assignment on D-O . . . . .	16
4. Fixing the CH3 gap . . . . .	16
5. Failsafe behaviour . . . . .	16
<b>SBUS Support (v3.5.0+, Futaba / FrSky)</b>	<b>17</b>
1. Building the inverter . . . . .	17
2. Wiring: SBUS receiver to Mega . . . . .	17

3. Enabling SBUS in the menu . . . . .	17
4. Troubleshooting . . . . .	18
<b>Sound Files on the SD Card</b>	<b>19</b>
SD card requirements . . . . .	19
Filename format + file types . . . . .	19
Track number map (v3.4.1) . . . . .	20
How the DFPlayer addresses files (v3.4.1+) . . . . .	20
Speaker wiring . . . . .	20
<b>Configuration Menu</b>	<b>22</b>
Top-level commands . . . . .	22
Main menu structure . . . . .	23
Compile-time feature flags (v3.4.1+) . . . . .	23
Editing a value . . . . .	24
<b>PID Tuning Guide</b>	<b>25</b>
What each gain does . . . . .	25
Tuning procedure . . . . .	25
Adaptive PID (optional) . . . . .	25
<b>Personality — Idle Animations &amp; State Reactions</b>	<b>27</b>
Idle animations . . . . .	27
State reactions . . . . .	27
Tuning the behaviour . . . . .	27
<b>Battery Monitoring</b>	<b>28</b>
Wiring the voltage divider . . . . .	28
Default thresholds (v3.4) . . . . .	28
Calibrating the divider factor . . . . .	28
Enabling the feature . . . . .	28
<b>Troubleshooting</b>	<b>31</b>
<b>Migrating from v1.4 / v1.5 Boards</b>	<b>33</b>
Board mapping . . . . .	33
What changes on the software side . . . . .	33

Solder-jumper check . . . . . 33

**Safety** . . . . . **34**

## Part I

# Getting Started

# Introduction — What is D-O Control & Power Board?

The D-O Control & Power Board is an Arduino-based control and power system for the self-balancing D-O droid from Star Wars: The Rise of Skywalker. Two active board revisions are supported: **V1.6** (flexible, supports PWM receivers and an optional external Arduino Nano for sound) and **V1.7** (compact, iBus-only, with all sound handled by the Mega directly).

## Key Features

- **Self-balancing** via MPU6050 IMU with PID, compatible with common clones (MPU6500/9250/6886).
- **FlySky iBus** reception with 10 channels, or classic PWM receivers as fallback.
- **Four servos** for mainbar and three head axes.
- **DFPlayer Mini** sound system with personality: greetings, moods, idle animations, tilt warnings.
- **Cytron MDD10A** dual-channel motor driver (10 A per channel).
- **Interactive serial CLI** with EEPROM configuration (PID tuning, feature toggles).
- **Battery monitoring** with low-voltage protection for 2x 2S LiPo in series (4S, 14.8 V nominal).

**Tip:** read the safety chapter before powering on for the first time. LiPo packs and servo stall currents deserve respect.

# Quick Start

This chapter gets a freshly built D-O from wired-up to rolling in about 20 minutes, assuming the hardware is assembled and wired. If something breaks along the way, jump to the Troubleshooting chapter.

**Wheels off the ground for first power-on.** The balance PID can swing the droid violently during calibration and tuning.

## 1. Install libraries

Open Arduino IDE, go to Library Manager, install: **IBusBM** (Bolder Flight Systems) and **DFRobotDFPlayerMini** (DFRobot). Wire, Servo, EEPROM, SoftwareSerial ship with the AVR core.

## 2. Select board

Tools → Board → **Arduino Mega or Mega 2560**. Select the correct COM port.

## 3. Open sketch

Open `D-O_ibus_v3.4/D_O_printed_droid_rc_ibus_v3.4.ino`.

## 4. Upload

Click Upload. Wait for "Done uploading." First upload takes a bit longer because AVR-GCC is cold.

## 5. Open Serial Monitor at 9600 baud

You should see `=== D-O Universal Controller v3.4.1 ===` followed by diagnostic output.

## 6. Calibrate IMU immediately

Within 3 seconds of boot, place D-O upright and perfectly still, then send `c`. Hold still for 5 seconds. Calibration offsets are saved to EEPROM.

## 7. Power-cycle

Unplug USB / battery for a second, then reconnect. The droid should start balancing on its own when placed upright.

## 8. Check the receiver

Send `m` to open the menu, select "Show Current Status", verify iBus shows valid channel values. If channels read 0 or unstable, the receiver is not bound or wiring is wrong.

## 9. Trim defaults if needed

If the droid lists forward or backward even though it is physically upright, adjust **Target Angle** in the PID menu (option 2). Range usually -1.5 to +1.5.

## 10. Take it for a test roll

Wheels on ground, stand back, arm the transmitter. Arcade mixing mode is default (right stick forward/back + left/right).

**Tip:** if the boot messages scroll past too fast, send `h` to see the CLI help reprinted any time.

## Choosing a Board & Sketch

There are two actively supported board revisions and three sketch choices. Before wiring anything, pick the combination that matches your parts bin. This chapter lists which sketch is the best choice for each board and why.

### The two boards at a glance

Aspect	V1.6 (Standard Control PCB)	V1.7 (Mini iBus Board)
Form factor	Larger, room for a 10-channel PWM receiver	Compact, iBus receiver only
Receiver types	iBus or classic PWM	iBus only
Nano slot for sound	Yes (optional)	No
DFPlayer routing	Mega D7/D8 (default) or Nano D0/D1 via solder jumper	Fixed to Mega D7/D8
Best for	Flexible builds, upgrade paths	Clean new builds with iBus

### Available sketches

- **D-O\_ibus\_v3.4** (current, recommended) - Universal controller with DFPlayer running on the Mega in all modes. Supports both iBus (Mode 1) and classic PWM receivers (Mode 0). Watchdog timer, IMU clone support, adaptive PID, dynamic lean angle. No external Nano needed.
- **D-O\_ibus\_v2.1** (legacy) - The predecessor to v3. Uses three setup modes (PWM only / hybrid iBus+PWM / pure iBus) and still supports an external Arduino Nano for sound via setup\_type 0. Kept for users with existing Nano-based installs.
- **D\_O\_Nano\_Sketch\_v2** (companion) - Runs on an external Arduino Nano and handles sound only. Used together with D-O\_ibus\_v2.1 when the DFPlayer is wired to the Nano instead of the Mega. Not needed with v3.4.

## Recommended combinations

Board	Receiver	Best sketch	Why
V1.7	iBus (FS-RX2A Pro, 10 CH)	D-O_ibus_v3.4 (Mode 1)	Only iBus supported on V1.7, no Nano slot. v3.4 is current.
V1.6	iBus (FS-RX2A Pro, 10 CH)	D-O_ibus_v3.4 (Mode 1)	Cleanest wiring, DFPlayer on Mega, all features.
V1.6	Classic PWM	D-O_ibus_v3.4 (Mode 0)	Keeps PWM receiver, no Nano needed because v3.1 drives DFPlayer from Mega.
V1.6	Existing Nano-based setup	D-O_ibus_v2.1 D_O_Nano_Sketch_v2	+ For builds already wired with Nano on sound. v2.1 setup_type 0 expects this.
v1.4 / v1.5 (older)	same as V1.6 / V1.7	same as V1.6 / V1.7	Pin-compatible. Pick the same sketch as the equivalent new revision.

## Upgrade note for v3.4.1

**v3.4.1** (point release) adds three cross-ported features from the AIO32 sister firmware: DFPlayer filename-based addressing (no more sequential SD copy), optional Madgwick AHRS, and optional class-style PID with anti-windup. On top of that, SBUS support was added via a new menu option `r`, a runtime Madgwick beta tuning option `b` was added, and the dead "Max Acceleration" field was removed from the EEPROM layout. The sketch folder name stays `D-O_ibus_v3.4` for continuity with existing user downloads.

**v3.4.0** (previous) had brought back the deadband + expo RC shaping that was missing in earlier v3 builds, fixed the dynamic lean angle to use shaped RC, clamped RC shaping output to the configured stick range, and blocked idle animations when the receiver was off.

**EEPROM layout change:** the magic number was bumped from `0xD043` (v3.4.0) to **`0xD044`** (v3.4.1). On **first boot after upgrading from any earlier version** the sketch detects the old magic and resets all settings to defaults. Re-run IMU calibration with `c` and re-apply your preferences via the menu (`m`).

## Uploading the Sketch

The Arduino sketch is a standard `.ino` file — no build system, no external dependencies beyond the Arduino libraries listed below. Upload is done from the Arduino IDE or from the command line with `arduino-cli`.

### Required libraries

Library	Quelle / Source	Zweck / Purpose
IBusBM	Bolder Flight Systems (Library Manager)	iBus receiver protocol on Serial1
DFRobotDFPlayerMini	DFRobot (Library Manager)	MP3 playback via SoftwareSerial
Wire, Servo, EEPROM, SoftwareSerial	Arduino AVR core (included)	I2C, servo PWM, persistent config, DFPlayer link
avr/wdt.h	AVR libc (included)	Watchdog timer

### Arduino IDE walkthrough

- Install **IBusBM** and **DFRobotDFPlayerMini** from Library Manager.
- Open `D-O_ibus_v3.4/D-O_printed_droid_rc_ibus_v3.4.ino`.
- Tools → Board → **Arduino Mega or Mega 2560**.
- Tools → Port → the one your Mega shows up on.
- Click Upload (arrow button). Wait for "Done uploading." in the bottom status bar.
- Open Tools → Serial Monitor. Set baud rate to **9600**.

### Command-line alternative

With `arduino-cli` installed and configured with the Arduino AVR core plus the two libraries:

```
arduino-cli compile --fqbn arduino:avr:mega D-O_ibus_v3.4
arduino-cli upload --fqbn arduino:avr:mega --port COM5 D-O_ibus_v3.4
```

**Note:** pins D0 and D1 carry the servo signals for Mainbar and Head 1. These are also the hardware UART (Serial) pins. The sketch uses `Serial` for the CLI anyway, which is fine because the Mega has Serial1-3 as well. If you ever see weird servo jitter during CLI usage, unplug USB and use a dedicated BEC-powered setup.

# IMU Calibration

The MPU6050 accelerometer and gyroscope have small manufacturing offsets (typically a few tenths of a degree). The balance PID has no way to tell "true vertical" from "IMU chip tilted on the board", so we measure the offsets once and subtract them from every reading. Without calibration the droid will lean forward or backward by a constant amount forever.

## How to run calibration

- Power on the Mega with USB or battery. Serial Monitor at 9600 baud.
- Within the **first 3 seconds** of boot, place the droid upright and perfectly still.
- Send the character `c`.
- The Mega will print "Calibrating..." and hold for about 5 seconds averaging samples.
- When "Calibration complete." appears, the offsets are written to EEPROM and survive power-cycles.
- Alternatively: send `m` for menu, select option **8** (IMU Calibration) at any time.

## When to re-calibrate

- After remounting or replacing the MPU6050 module
- If the droid consistently lists forward or backward even with a sane Target Angle
- After flashing a new sketch version that bumps the EEPROM magic (fresh install resets offsets)
- Rarely: if you suspect the IMU got a hard shock and the internal bias shifted

**Tip:** calibration samples are averaged over 5 seconds. Even a slight hand tremor makes the average drift. Put the droid on a solid surface, then **take your hands off** before sending `c`.

**IMU clone compatibility:** v2.1 and v3.4 talk to the MPU6050 via raw I2C register writes (no third-party library). They scan both 0x68 and 0x69 addresses and identify the chip via WHO\_AM\_I (MPU6050, MPU6500, MPU9250, MPU6886). Typical AliExpress GY-521 modules just work. v1.1 and the Mega2560 ur-sketch only talk to 0x68 and cannot use modules with AD0 tied high.

## Part II

# Core Features

# Transmitter Setup (FlySky)

D-O is designed around the FlySky ecosystem: a 10-channel transmitter and a 10-channel iBus-capable receiver. The compact **FS-RX2A Pro** is the recommended choice (10 CH, ~6 USD). Note that the older **FS-iA6B** is a 6-channel receiver — it does not cover the full D-O channel map (we use CH7-CH10 for sound), so it is not sufficient on its own.

**Wichtig / Important:** FlySky transmitters do **not** assign CH3 to any stick or switch by default. You must manually map CH3 in the transmitter menu, otherwise the mainbar servo will not respond. See step 3 below.

## 1. Choosing the receiver (iBus output saves cabling)

Almost every modern FlySky 10-channel PWM receiver (FS-iA10B, FS-iA10, FS-R9B, FS-X6B and so on) also exposes an **iBus output pin** in addition to the classic per-channel headers. Use that single iBus line to the Mega (D19 / RX1) and you save running 10 separate signal wires. It also means you can choose setup Mode 1 (iBus) in the sketch, which is the cleanest configuration. The dedicated FS-RX2A Pro is smaller and iBus-only but functionally equivalent.

## 2. Binding the receiver

- Press and hold the **bind button** on the receiver while connecting its power.
- Release the button. The LED goes into bind mode (typically fast blink or solid blue).
- On the transmitter, enter **BIND mode** (usually System → RX Setup → Bind).
- When the transmitter confirms bind success, exit BIND mode on the transmitter.
- The receiver LED should now be solid when receiving signal, slow blink on signal loss.

### 3. Channel assignment on D-O

CH	Funktion / Function	FlySky Default Control
CH1	Drive 1 (Steering in Arcade, Motor 1 in Tank)	Right Stick L/R
CH2	Drive 2 (Throttle in Arcade, Motor 2 in Tank)	Right Stick U/D
CH3	Mainbar Servo	<b>not assigned</b> — map to VrA or a switch
CH4	Head Servo 1 (Pitch)	Left Stick U/D
CH5	Head Servo 2 (Yaw)	Left Stick L/R
CH6	Head Servo 3 (Roll)	VrB dial
CH7	Sound Mute (2-pos)	SwA
CH8	Sound Mode / Trigger (2-pos)	SwB
CH9	Sound Mood (3-pos, neg/mid/pos)	SwC
CH10	Sound Squeak (2-pos)	free — map to any unused control

### 4. Fixing the CH3 gap

On FlySky transmitters (FS-i6, FS-i6X, FS-i6S) CH3 is unassigned out of the box. To make the mainbar servo respond, enter **Functions** → **Aux. Channels** and map CH3 to **VrA**, or alternatively to any 2- or 3-position switch. After saving, check in the system monitor (Functions → Display) that CH3 now reacts to your control.

### 5. Failsafe behaviour

v3.4 holds the drive channels at center when the iBus signal drops for more than 500 ms and keeps balancing. Idle animations are suppressed during signal loss. v2.1 instead triggers an emergency stop on signal loss. Either way, set your transmitter failsafe to neutral sticks and neutral switches, not to random values.

**Tip:** if you prefer tank-style driving (one stick per motor), open the configuration menu, go to Driving Dynamics, set **Mixing Mode** to 0. Arcade Mode (1) is the default and matches standard FlySky stick assignment out-of-the-box.

# SBUS Support (v3.5.0+, Futaba / FrSky)

Starting with v3.5.0 the Mega sketch supports **SBUS** as an alternative to FlySky iBus. SBUS is the serial RC protocol used by Futaba and FrSky receivers. Channel mapping on D-O is identical to iBus (CH1 = Drive 1, CH2 = Drive 2, etc.), so once the hardware is wired correctly the droid behaves the same regardless of which protocol you picked.

**Hardware prerequisite:** the ATmega2560 UART cannot invert its RX line in hardware. SBUS uses an inverted signal, so an **external inverter** is mandatory between the receiver SBUS output and the Mega Serial1-RX pin (D19). Without it the Mega simply will not see any frames (packets = 0 in menu option 9).

## 1. Building the inverter

Two proven variants work on a piece of perfboard between receiver and Mega:

- **NPN-Transistor-Variante:** 1 × BC547 (or 2N3904), 1 × 10 kΩ (collector pull-up to +5V), 1 × 10 kΩ (base resistor from SBUS IN). Collector = inverted output to Mega D19. Emitter to GND. Small, ~0.20 EUR in parts.
- **74HC14-Variante:** single-gate Schmitt-trigger inverter IC. Pin 1 = SBUS IN, Pin 2 = inverted output to Mega D19, Pin 7 = GND, Pin 14 = +5V. Faster edges, but 6 unused gates sit idle — only worth it if you already have a 74HC14 around.

## 2. Wiring: SBUS receiver to Mega

- Receiver power (red): to +5V BEC.
- Receiver ground (black/brown): to common GND.
- Receiver SBUS out (white/yellow): into the inverter input.
- Inverter output: to Mega pin D19 (Serial1-RX).
- Inverter power (+5V, GND): from the Mega 5V/GND rail (low current, no BEC needed).

## 3. Enabling SBUS in the menu

- Open Arduino IDE Serial Monitor at 9600 baud and send `m` to enter the config menu.
- Press `r` (RC Protocol).
- Enter `1` for SBUS. The sketch re-initialises Serial1 immediately with 100000 baud / 8E2 parity.
- Press `s` (Save and Exit) to persist the choice in EEPROM.
- Verify: press `m` again, then option `9` (Show Status). Top line should say "RC Protocol: SBUS (100000 8E2, needs HW inverter)". The RC Drive values should respond to your sticks within a second.
- To roll back: menu → `r` → `0` → `s`.

## 4. Troubleshooting

Symptom	Likely cause
No RC response, packets = 0	Inverter not wired correctly — check SBUS IN → inverter → D19. Test with a multimeter in continuity mode.
Random channel values (garbage)	Missing inverter (droid sees inverted signal) or wrong baudrate — menu option 9 should show 100000 8E2.
Works for a second then stops	SBUS failsafe bit is firing. Check receiver bind, transmitter powered, antenna orientation.
Works but all channels reversed	Inverter and receiver both inverting (double-invert). Use one inverter only between RX and Mega.

**Tip:** You can keep both receiver types in your bin and switch by unplugging one and plugging the other in (adjust the inverter pass-through for SBUS). The Mega does not care whether the current transmitter is FlySky or FrSky as long as the menu protocol setting matches the wired receiver.

# Sound Files on the SD Card

The DFPlayer Mini plays MP3 files from a MicroSD card. The track numbering, folder name, and filename format are strict — the DFPlayer does not read filesystems flexibly. Follow the rules below or sounds will not play.

## SD card requirements

- MicroSD card, up to 32 GB.
- Formatted **FAT16 or FAT32**. Not exFAT, not NTFS.
- Create a folder named `/mp3/` at the root.
- All sound files go into that folder, no subfolders.

## Filename format + file types

The DFPlayer identifies a track by the first **four digits** of the filename. Everything after the prefix is free — you can name files in a way that actually tells you what is inside. Both **MP3 and WAV** are supported.

- All of these work and resolve to **track 1**:
  - `0001.mp3`
  - `0001_startup.mp3`
  - `0001 battery charged.wav`

Leading zeros are required: `1.mp3` does **not** work; it must be `0001.mp3` (or any filename starting with `0001`).

## Track number map (v3.4.1)

Track	Funktion / Function
0001	Startup sound — "battery charged"
0002	Default — "I am D-O"
0003–0005	Greetings (played on CH8 high, random pick)
0006–0009	Negative (played on CH9 low, random pick)
0010–0014	Positive (played on CH9 high, random pick)
0015–0020	Squeaky wheel (played on CH10 trigger, random pick)
0021	Tilt warning (auto, when angle > 15°)
0022	Recovery / relief (auto, after a tilt event)
0023	Low-battery warning (auto, below warning threshold)
0024–0030	Idle sounds (auto, when droid sits still)
0031	System ready (after boot)
0032	Signal lost (when receiver times out)

## How the DFPlayer addresses files (v3.4.1+)

Starting with sketch v3.4.1 (flag `USE_PLAYMP3FOLDER = 1`, default ON), the firmware asks the DFPlayer to look up files by the **filename prefix**, not the filesystem index. That means you can copy files in any order, replace a single file, or add new tracks without having to reformat and re-copy the whole card. Earlier sketches (v3.4.0 and before) relied on FAT-index order, which required strict 0001-first sequential copying. If you run an older sketch or set the flag to 0, the old rule applies.

## Speaker wiring

The board has **two speaker outputs**, each with its own + and - terminal. Both outputs are **driven directly by the DFPlayer Mini** — no external amplifier on the board, and none needed. The two outputs are wired in parallel to the DFPlayer mono audio stage, so they carry the same signal. You can use:

- A **single speaker** on either of the two outputs — the other output stays unused.
- **Two speakers in parallel** (one per output), for example one in the head and one in the body. Both play the same mono signal.

Recommended per-speaker spec: **8 Ω, 2-3 W, 28-40 mm diameter**. If you run two 8 Ω speakers in parallel the combined load is 4 Ω — still within the DFPlayer spec, but keep volume below max. Always observe the + / - marking when wiring two speakers in parallel so they move in phase (otherwise they cancel acoustically).

**Tip:** MP3 bitrate 128-320 kbps. Lower bitrates work too but sound is already compressed through a tiny speaker, so 192 kbps is a good sweet spot. WAV files are larger but avoid any decoder quirks for very short effects.

# Configuration Menu

Every tuning parameter in v3.4.1 is reachable from an interactive serial menu. Open the Arduino IDE Serial Monitor at 9600 baud and send `m` at any time. Changes are held in RAM until you save with option `s` — then they survive power-cycles via EEPROM.

## Top-level commands

Key / Taste	Function / Funktion
m / M	Open main configuration menu (anytime)
c / C	Run IMU calibration — only in the 3-second boot window after reset
h / H / ?	Show the top-level help text

*Status, reset and debug functions live inside the menu (option 9 for status, option 7 for feature toggles, etc.). There are no separate hotkeys for them.*

## Main menu structure

Option	Inhalt / What it configures
1	Setup Mode (0=PWM, 1=Serial RC — iBus or SBUS chosen via option r)
2	PID Configuration — KP, KI, KD, Target Angle, Max Integral
3	Adaptive PID — KP/KD values for slow / medium / fast speeds
4	Driving Dynamics — Ramp Rate, Max Lean Angle, Deadband, Expo Factor, Mixing Mode
5	Battery Settings — Warning V, Critical V, Voltage Divider Factor, Recovery Mode
6	Sound Settings — Volume (0-30), Sound Intervals, Idle Interval min/max
7	Feature Toggles — Mainbar Correction, Ramping, Adaptive PID, Dynamic Angle, Idle Actions, State Reactions, Battery Monitor, Servos, Watchdog
8	Run IMU calibration
9	Show current status (incl. active RC protocol + FAILSAFE flag on SBUS)
r	RC Protocol switch (iBus / SBUS) — runtime re-init, see SBUS chapter
b	Madgwick Beta tuning (only active when USE_MADGWICK_AHRS compile-flag=1)
m	Motor test & motor-swap / invert configuration
i	IMU axis test (live angle readout)
s	Save to EEPROM and exit
0	Exit without saving

## Compile-time feature flags (v3.4.1+)

v3.4.1 adds three compile-time flags at the top of the .ino file, ported from the AIO32 sister firmware. Default values keep v3.4.0 behaviour byte-identical so an upgrade is safe; flipping them to 1 enables optional improvements that should be bench-tested before free driving.

Flag	Default	What it does
USE_PLAYMP3FOLDER	1 (ON)	DFPlayer addresses by filename (/mp3/NNNN.mp3) instead of FAT index — SD copy order becomes irrelevant
USE_PID_CONTROLLER_CLASS	0 (OFF)	Replaces inline PID with class-style version: D-term first-order LPF + back-calculation anti-windup
USE_MADGWICK_AHRS	0 (OFF)	Replaces complementary filter with Madgwick 6-DoF quaternion AHRS. Tune beta via menu option b

## Editing a value

When you pick a sub-menu, each parameter is shown with its current value and a min/max range. Hit Enter to keep the current value, or type a new number and press Enter. Floats use a dot as decimal separator (0.8, not 0,8).

**Tip:** if you accidentally mess up a setting and the droid misbehaves, the simplest recovery is to flip the EEPROM magic number in the sketch (line with `uint16_t magic`) and re-flash once — that forces a clean defaults reset. Then flip it back and re-flash. Alternatively, exit the menu via option **0** (Exit without Saving) if the mistake is unsaved.

# PID Tuning Guide

Balance behaviour is controlled by a PID loop that reads the tilt angle from the MPU6050 and drives the motors to counteract any deviation from **Target Angle**. Three gains — KP, KI, KD — decide how aggressively the loop reacts. The default values (KP=25, KI=0, KD=0.8) are a reasonable starting point. If your droid oscillates, feels sluggish, or drifts, tune the values below.

## What each gain does

Gain	Wirkung / Effect	Symptom wenn zu niedrig / Symptom if too low	Symptom wenn zu hoch / Symptom if too high
KP (P)	How hard to push when tilted / Wie stark gegengesteuert wird	Droid falls over slowly / Droid kippt langsam	Rapid oscillation, buzzing motors / schnelles Schwingen
KI (I)	Corrects long-term drift / Korrigiert Langzeit-Drift	Droid slowly drifts forward or back / Droid driftet langsam	Wobble builds up over seconds / aufbauendes Schaukeln
KD (D)	Damping — resists sudden changes / Daempfung gegen ploetzliche Aenderungen	Overshoot after disturbance / Ueberschiessen nach Stoss	Jittery, nervous motion / zappeliges Verhalten

## Tuning procedure

- **Start with defaults.** KP=25, KI=0, KD=0.8. Target Angle = -0.3 (calibration-dependent).
- **Fix Target Angle first.** If the droid consistently leans forward/back while trying to balance upright, adjust Target Angle by 0.1-0.3 at a time until the droid sits roughly level.
- **Tune KP for authority.** If the droid feels sluggish and falls without much resistance, raise KP in steps of 2 (25, 27, 29, ...) until oscillation appears, then back off 20 %.
- **Tune KD for damping.** If the droid oscillates at any KP, raise KD in steps of 0.1 (0.8, 0.9, 1.0, ...) until oscillation dies out. Too much KD makes it nervous.
- **Add a small KI if drifting.** If the droid slowly creeps in one direction even though KP+KD feel right, set KI to 0.1 or 0.2. Rarely needed — most builds work with KI=0.
- **Check Max Integral.** Keep it at 400. Lower only if KI causes wind-up behaviour.
- **Save.** In the configuration menu, send `s`.

## Adaptive PID (optional)

v3.4 can switch between three PID gain sets depending on how fast the droid is moving: slow, medium, fast. This helps when a single PID set cannot handle both "standing still" and "full throttle" well. Tune the default gains first, only touch adaptive values if the droid behaves well at rest but unstable at speed (or vice versa).

**Tip:** test tuning on wheels off the ground first. A violent oscillation with wheels on the ground can fling the droid across the room and damage things.

# Personality — Idle Animations & State Reactions

D-O in the films is a nervous, curious little droid — he makes noises, looks around, reacts to being tipped. v3.4 recreates this with two systems: **idle animations** when the droid sits still, and **state reactions** triggered by events like tilting or low battery. Both are on by default and configurable in Feature Toggles.

## Idle animations

When there is no RC stick activity for 3 seconds, the droid enters idle mode. Every 5–15 seconds (configurable) a random action fires:

- **70 % chance:** random idle sound (tracks 0024–0030)
- **30 % chance:** random head movement (nod, look around, head shake)

In v3.4 idle animations are suppressed when the receiver signal is lost or has never been received since boot — no more D-O nodding in a room with the transmitter off. Emergency-stop also blocks idle servo moves.

## State reactions

Trigger / Ausloeser	Sound	Details
Tilt warning (angle > 15°)	0021	Plays once per tilt event, 5 s cooldown
Recovery (tilt returned to normal)	0022	Plays after a tilt event when droid is balanced again
Low battery (below warning V)	0023	Plays once every 30 s while below threshold
Signal lost (receiver timeout)	0032	Plays once when iBus drops for > 500 ms
System ready (end of boot)	0031	Plays once after successful init

## Tuning the behaviour

Open the configuration menu with `m`, option 6 (Sound Settings) and option 7 (Feature Toggles):

- **idle\_interval\_min / idle\_interval\_max** (Option 6) — how often idle actions fire
- **min\_sound\_interval** (Option 6) — minimum gap between any two sounds to prevent spam
- **Idle Actions** toggle (Option 7) — switch off entirely if you want a silent droid
- **State Reactions** toggle (Option 7) — switch off tilt/recovery/low-batt sound triggers

# Battery Monitoring

D-O runs on **2 × 2S LiPo packs wired in series = 4S total**: 16.8 V fully charged, 14.8 V nominal, 12.0 V at the deep-discharge limit of 3.0 V per cell. The Mega has an optional voltage-monitoring feature that warns before the cells drop too low — but the feature requires a simple voltage divider that is **not fitted on the PCB ex-works**. You have to solder it on yourself if you want this safety net.

**That is why Battery Monitor is OFF by default in v3.4.** Without the divider, pin A15 floats and reads random values — the feature would fire false criticals immediately. Only enable it **after** wiring the divider.

## Wiring the voltage divider

Two resistors (10 kΩ and 3.3 kΩ) scale the battery voltage down to under 5 V so the Mega ADC can measure it safely on pin A15:

```
Battery(+) -- [10k] -- A15 -- [3.3k] -- GND

Ratio: (10 + 3.3) / 3.3 = 4.03
At A15 = 5.0 V the battery is ~20.15 V (headroom above 4S max of 16.8 V)
At A15 = 3.67 V the battery is 14.8 V (4S nominal)
At A15 = 3.17 V the battery is 12.8 V (4S critical)
```

## Default thresholds (v3.4)

Config Field	Default	Meaning
battery_warning	13.6 V	~3.4 V per cell, long-beep warning, every 30 s
battery_critical	12.8 V	~3.2 V per cell, stops motors and alarms
voltage_divider	4.03	Calibration factor for 10k / 3.3k divider
battery_monitor	false	Feature toggle — enable only after wiring the divider
battery_recovery	false	If true, motors resume after battery voltage climbs back above critical + 0.2 V

## Calibrating the divider factor

Real resistors have 1-5 % tolerance. For accurate readings, measure the actual battery voltage with a multimeter while the droid is on, then adjust **voltage\_divider** in the Battery menu until the Mega-reported voltage matches. Example: if multimeter says 15.1 V and the Mega says 14.7 V, multiply the current divider factor by  $15.1 / 14.7 \approx 1.027$ .

## Enabling the feature

- Solder the 10 kΩ + 3.3 kΩ divider from Battery+ to GND with the midpoint on A15.

- Open the configuration menu (m) → Feature Toggles (7).
- Set **Battery Monitor** to true.
- Optionally set **Battery Recovery** to true if you want motors to resume after a brief voltage dip.
- Save with s.
- Verify by going to Show Status — battery voltage should now show a plausible reading.

**Tip:** if you never add the divider, leave Battery Monitor off. The droid works fine without it — you just lose the low-voltage safety net. A cheap LiPo alarm clipped to the balance connector achieves the same thing externally.

## Part III

# Reference & Appendices

# Troubleshooting

Common problems and the usual causes. Try the steps in order.

## Droid does nothing / no serial output

- Wrong board selected in Arduino IDE (must be Mega 2560, not Uno)
- Wrong COM port
- USB cable is charge-only without data lines
- Serial Monitor baud rate wrong (must be 9600)

## Compile errors about I2C or DFRobotDFPlayerMini

- Library not installed — use Library Manager (Sketch → Include Library → Manage Libraries)
- Wrong library version — DFRobotDFPlayerMini must be the DFRobot one, not a fork
- Stray files in the sketch folder — keep only the .ino for the relevant version

## Droid does not balance / tips over

- IMU not calibrated — send `c` within 3 seconds of boot, or option 8 in the menu
- Target Angle wrong — adjust in PID menu by 0.1 at a time until droid sits upright
- Motor wired backwards — use Motor Swap / Motor Invert in Feature Toggles
- MPU6050 wired wrong — SDA on D20, SCL on D21, not the analog SDA/SCL pins
- IMU mounted rotated — enable IMU Invert if forward/back feels reversed

## Droid oscillates rapidly

- KP too high — lower by 2-3 in PID menu
- KD too low — raise by 0.1 in PID menu
- Dirty USB power — run on battery only during tuning

## No RC response

- Receiver not bound — follow binding procedure in Transmitter Setup chapter
- iBus signal on wrong pin — must be D19 (Serial1 RX)
- Setup Mode wrong — menu option 1, switch to 1 (iBus) if you have an iBus receiver
- iBus baudrate mismatch — try toggling between 9600 and 115200

## Mainbar servo does not respond

- CH3 not assigned on transmitter — FlySky default has no CH3 mapping, see Transmitter Setup

## No sound

- SD card not FAT16/FAT32
- Filenames not in 0001.mp3 format (must be exactly four digits)
- Files not in `/mp3/` folder

- DFPlayer wiring reversed — Mega D7 reads, Mega D8 writes via 1 k $\Omega$  resistor
- Missing 1 k $\Omega$  resistor on TX line damages the DFPlayer RX signal
- Speaker loose or wrong polarity (8  $\Omega$  2 W recommended)
- Volume set to 0 in Sound Settings

### **Forward/backward much slower than turning**

- Mixing Mode is Tank but you are driving with arcade sticks — change to Arcade Mode (Driving Dynamics, option 1)

### **Battery voltage reads nonsense**

- Voltage divider not wired up — A15 floats without the 10k / 3.3k resistors
- Battery Monitor still on — turn it off (Feature Toggles) until the divider is in place
- Divider factor not calibrated — measure battery with multimeter and tune voltage\_divider

### **Droid restarts randomly**

- Watchdog fires because something hangs in the loop — try disabling watchdog temporarily (Feature Toggles) to see if the hang happens visibly
- Power supply brown-out — motors and servos must have their own BEC, not share the Mega 5 V rail
- I2C error loop on IMU — check SDA/SCL wiring, try a different MPU6050 module

### **Idle animations play with transmitter off**

- Only in v3.3.x or older — upgrade to v3.4, which gates idle on receiver signal

## Migrating from v1.4 / v1.5 Boards

The older v1.4 (Control PCB) and v1.5 (Mini iBus) boards are **pin-compatible** with the current V1.6 and V1.7 — nothing needs to be rewired on the hardware side. You only need to update the sketch and understand a few behavioural differences.

### Board mapping

Old board	Current equivalent	Recommended sketch
v1.4 Control PCB (large)	V1.6 Standard PCB	D-O_ibus_v3.4 Mode 0 or 1
v1.5 Mini iBus	V1.7 Mini iBus	D-O_ibus_v3.4 Mode 1

### What changes on the software side

- **Sketch version.** Older installs ran v1.1 or early v2.x sketches. v3.4 is a significant rewrite — read the Board & Sketch Choice chapter.
- **EEPROM layout.** First boot on v3.4 resets all settings to defaults because the EEPROM magic was bumped to 0xD043. Run IMU calibration (c) and re-enter your menu settings.
- **Battery thresholds.** Were labelled "2 × 2S" but actually sized for a single 2S pack. v3.4 corrects this for the real 4S setup (13.6 V / 12.8 V).
- **External Nano eliminated.** v1.4 boards typically ran an Arduino Nano for sound. v3.4 drives the DFPlayer directly from the Mega, regardless of Mode. Remove the Nano or leave it unpowered — it is no longer part of the signal path when v3.4 is flashed.
- **Hybrid mode gone.** v2.1 had three setup types including Hybrid (iBus CH1-6 + PWM CH7-10). v3.4 has only Mode 0 (all PWM) and Mode 1 (all iBus). Most users want Mode 1.

### Solder-jumper check

If you are going from v1.4 with external Nano to v3.4 with Mega-native sound, verify on your PCB that solder pads **P7** and **P8** are bridged — these route Mega D7/D8 to the DFPlayer. For Mode 0 (PWM receiver), also bridge **CH7**, **CH8**, **CH9**, **CH10** so the receiver sound switches reach the Mega. For Mode 1 (iBus) you only need P7 and P8; the 10 channels arrive over the single iBus wire.

**Tip:** if you are unsure about a solder-jumper being bridged, look at the PCB under bright light — bridged pads show a clear solder blob connecting the two halves. Unbridged pads have visible gap and copper separator.

# Safety

The full liability disclaimer is on page 2. This chapter lists the practical safety rules for operating D-O.

Rule	Details
LiPo handling	Use a LiPo-safe charger. Never charge unattended. Replace packs that swell or take damage.
Polarity	Verify every power connection before power-on. Reversed polarity kills the Mega instantly.
Voltage	2x 2S LiPo in series = 4S total. 16.8 V full, 14.8 V nominal, 12.0 V empty. Do not discharge below 3.0 V per cell.
Servo power	Power servos from a BEC (5-6 V, 3-5 A), NOT from the Mega 5 V rail.
Motor currents	Cytron MDD10A tolerates 10 A continuous per channel. Keep an eye on heat under stall conditions.
First power-on	Wheels off the ground. PID can swing the droid violently during calibration and tuning.
Calibration posture	Hold D-O upright and perfectly still for 5 seconds during IMU calibration.
Environment	Clear the area. A tipping droid can fall onto hands, pets, or your coffee.