# Runaway Takeoff Prevention

## Description

Runaway Takeoff Prevention is a safety feature designed to detect and auto-disarm if an unsafe response is detected by the PID controller on the first takeoff after power-up. Sometimes referred to as a "Tasmanian Devil", some configuration or hardware issues can cause a violent and dangerous out-of-control spin immediately after arming or as the throttle is raised for takeoff. Some common causes are:

- Props on incorrectly
- Motor order wrong
- Motors spinning the wrong direction
- Flight controller (gyro) orientation incorrect

This feature is designed to only intervene and auto-disarm during a runaway event on first takeoff after powering up, and should have no impact on normal flight.

## WARNING

**While this feature is designed to prevent most types of runaway takeoff events, it cannot be guaranteed that all such occurrences will be prevented. The wide range of hardware and variable circumstances mean that there may be cases where this feature may be unable to provide additional protection. As always it is the pilot's responsibility to ensure their aircraft is configured properly and safe to fly.**

## Operation

The Runaway Takeoff Prevention feature is enabled by default beginning with Betaflight 3.3. There are two phases of operation:

**Takeoff:**

After the first arm after powering up, the feature looks for excessive PID responses that indicate the craft is not responding properly and escalating towards an out-of-

control runaway event. If detected the flight controller will automatically disarm. A beep pattern - one long, one short - will indicate that a runaway takeoff event occurred and "RUNAWAY" will be displayed as the arming disabled reason in both the OSD and CLI.

**Successful Flight:**

Once the craft successfully takes off and is in controlled flight the feature will deactivate for the remainder of the battery. After that finial successful take-off, the pilot can arm and disarm at any time without re-activating the feature.

The detection of successful flight involves looking at throttle level and control stick activity along with the craft responding correctly to the PID controller over a period of time. By deactivating the feature after successful flight is detected we prevent subsequent inadvertent disarming caused by crashes or gate/branch clips.

This short video is an example of the feature activating:
https://youtu.be/FEczCjAWv_0

**Unsuccessful Flight:**

If triggered, the pilot should disarm immediately. The feature will remain active until a successful flight clears the problem. Re-arming without solving the problem will likely re-trigger the protection. False triggering can, rarely, occur. It can be minimised by modifying the configuration parameters.

## Configuration

In general there shouldn't be any need to adjust the configuration parameters. Default values were chosen based on testing and should give good results. However as is always the case, additional testing and feedback from the users may identify some further optimization to these settings. It's anticipated that once the Runaway Takeoff Prevention feature matures that some of the tunable values might be eliminated and replaced with optimized hard-coded values to simplify the configuration.

The first group of parameters configure how the feature detects and activates during a runaway takeoff event:

```
runaway_takeoff_prevention = ON
```

Set this to `OFF` to completely disable the feature. Note that there will be no protection against runaway takeoff events and the firmware will behave as it did before the feature was implemented. **WE DO NOT RECOMMEND DISABLING THE**

**FEATURE.** Runaway events can be quite violent and personal injury or damage to property can occur for even seemingly minor setup issues like putting the props on incorrectly.

The remaining parameters affect the logic used to detect normal controlled flight and deactivate the feature for the remainder of the battery:

`runaway_takeoff_deactivate_throttle_percent = 25` (reduced to 20 in BF4.0)

Determines the minimum throttle percentage threshold where successful flight can be considered. Valid values range from 0 to 100. Along with throttle level the logic also requires activity on the roll, pitch or yaw sticks (minimum 15% deflection) along with the PID controller successfully controlling the craft with the PID_sum staying under control. When these conditions are met the logic accumulates successful flight time. Generally you won't need to adjust this value as most quads require around 25% or more throttle to takeoff/hover. The exception may be if you have and extremely powerful or light craft that is capable of flying well below 25% throttle. In this case you may want to lower this value closer to your actual hover throttle percent. If this value is set too low it's possible that the logic will deactivate too quickly and may not trigger in a real runaway event. Setting it too high will result in the logic taking more flight time to deactivate as it only accumulates flight time when the throttle is above the setting.

`runaway_takeoff_deactivate_delay = 500`

This is the amount of successful flight time in milliseconds that must be accumulated to deactivate the feature. Valid values range from 100 (0.1 seconds) to 1000 (1 second). The default value of 500 (0.5 seconds) seems to be very reliable and shouldn't need to be adjusted. The goal is to deactivate the logic after a "reasonable" but short period of time once we've determined the craft is flying normally. However we want it to deactivate before we might reach the first point where a crash or other event may occur (like at the first gate during a race). Raising this value will delay the deactivation and it's possible that a crash or gate/branch clip could cause an unintended disarm. Lowering this value too much could result in the logic deactivating too soon and not providing protection in a runaway event. It's important to note that the delay is the **accumulated** amount of flight time where the other criteria like throttle level, stick activity (minimum 15% deflection), etc. are met. Thus the "real" elapsed time before deactivation may be longer than 0.5 seconds if the throttle was dropping below the limit or if the R/P/Y sticks were centered. The actual behavior can be viewed by using blackbox logging - see the debugging section below.

## Debugging

Additional data may be captured in a blackbox log that will give insight into the operation of the Runaway Takeoff Prevention feature. Enable capturing this additional data by:

```
set debug_mode = RUNAWAY_TAKEOFF
```

This will enable additional debug values in the log that can be used to observer the operation of the feature.

**debug[0]** - boolean value (0/1)

This value indicates whether the feature is active (1) and in detection mode. Once successful flight deactivates the feature the value will be (0).

**debug[1]** - boolean value (0/1)

Indicates whether the PID_sum value for some axis has exceeded the `runaway_takeoff_threshold` and we are in a countdown for `runaway_takeoff_activate_delay` milliseconds. The logic is preparing to auto-disarm assuming the PID_sum value continues to exceed the threshold.

**debug[2]** - boolean value (0/1)

Indicates that the criteria for deactivation (throttle level, stick activity and PID response) has been met and we are accumulating in-flight time.

**debug[3]** - number

Indicates the in-flight milliseconds accumulated so far. Once this reaches `runaway_takeoff_deactivate_delay` the feature will be disabled. Any time debug[2] = 1 then debug[3] will be accumulating time.

As the Runaway Takeoff Prevention feature matures based on feedback from the user community the debugging features may become unnecessary and removed to save space some time in the future.

## Known Issues

- Crashing severely immediately on takeoff may cause an auto-disarm. Not really an issue and probably a good thing. Simply don't crash on takeoff. :)
- Severe bouncing/bumping the ground on takeoff may cause a disarm. This is not actually very likely to occur as it takes a really big hit to trigger the disarm with the default configuration.

- Arming with no props and moving the R/P/Y sticks significantly may trigger a disarm. This is normal and is caused because the PID controller is trying to control the craft and it's not responding properly (because there are no props!). As a result the PID_sum values will grow as the PID controller begins to push harder and harder to make something happen. This can end up looking like a runaway PID_sum event. The feature has provisions in place to allow limited bench testing without disarming. Heavy yaw movements could still trigger a disarm but generally "wiggling" the sticks to see the motor responses won't be a problem. Testing **while connected to the configurator** will temporarily disable runaway takeoff prevention and allow unlimited bench testing without props.
- "Hand flying" with the props off could trigger a disarm. **NEVER TRY TO HOLD AN ARMED CRAFT IN YOUR HANDS WHILE THE PROPS ARE ON!!** If the props are off and you move the craft around while the motors are spinning the PID controller will oppose the motion but because there are no props the PID_sum value will grow possibly triggering a disarm. On the other hand, holding the craft still while moving the sticks to feel the motor response will not trigger a disarm.

## Version History

### Betaflight 4.0

- Changed `runaway_takeoff_deactivate_throttle_percent` default to 20.

### Betaflight 3.3

- Simplified configuration by removing unnecessary tuning parameters `runaway_takeoff_threshold` and `runaway_takeoff_activate_delay`. The previous default values were found to be appropriate and unnecessary to tune.
- Enhance logic to accelerate deactivation under higher throttle conditions.

### Betaflight 3.3RC3

- Added gyro rate checking to activation detection to improve bench testing

### Betaflight 3.3RC2

- Reactivate the feature after the pilot uses crash-flip to recover from a crash. Provides protection for the next arming and takeoff in case of broken props or other problems that might cause a runaway.
- Disable the feature temporarily while crash-flip is active to prevent triggering while the pilot is attempting to flip over.

- In coordination with the Betaflight App 10.2.0 or later, temporarily disable the feature when arming while connected to the configurator. This allows unrestricted bench testing without risk of triggering a disarm. Prevents any need to manually disable the feature which may result in forgetting to re-enable before flying.

### Betaflight 3.3RC1

- Initial release