Servos & Servo Tilt

Introduction

Starting with v3.1, default servo output assignments are deleted from the firmware. Instead, servo outputs must be explicitly assigned by `resource` CLI command. (For details of the `resource` command, please refer to Betaflight resource remapping.

Background Internal on STM32 processors are Timers that are used for timing the output pulse to motor, servos, etc. Each FC board can have different STM32 pins connected to input & outputs on the FC. This is one of the main reasons for different Target hex files. The pins and internal resource are Defined in the "target'c' source files found here.

For example: The timer and channel assignment for outputs SPRacingF3/target.c lists:

```
Output 1: TIM16 CH1
Output 2: TIM17 CH1
Output 3: TIM4 CH1
Output 4: TIM4 CH2
Output 5: TIM4 CH3
Output 6: TIM4 CH4
Output 7: TIM15 CH1
Output 8: TIM15 CH2
```

Grand rule is that channels of a timer can not be assigned to different functions. If you have motors connected to Outputs 3 and 4, then TIM4 is locked to motor and can not be used for something else. This means that you can not use Output 5 and 6 for servos. (The exceptional case is when motors are controlled by legacy PWM protocol; in this case, motors and servos may be able to be mixed on channels of the same timer. If this does not work for your board, see fixes below.)

Options are (1) Output 1 & 2: Servos and Output 3~6: Motors

or

(2) Output 1~4: Motors and Output 7 and 8: Servos

Then there is another point of consideration if you are using DShot; DMA channel conflict.

General Information

Use `resource list` to view current running assignments.

```
resource list
```

Servos are assigned to board pads/pins by resource CLI command.

```
resource servo n xYY
```

Here, n is a servo number starting from 1, xYY is a MCU pin identifier, for example, A8, c8 and B10.

Notice that the servo numbers are 1 origin. This is different from servo numbering in the configurator's Servos tab. Servos you assign here as 1 and 2 corresponds to servos 0 and 1 in the Servos tab.

Examples

Here are some examples of assignments. If you can't find a one that fit your needs, then you have to go read the hard boiled story: Working on your own in the later half of this page.

Target/board maintainers, please add example entries that reflect mappings based on the v3.0.1 `pwm_mapping.c`; it can cover all the mappings, not only for servo tilts

Example 1: NAZE32 "Shift by 2" style assignment

If you have a NAZE32 already setup based on "**Shift Motor Outputs by 2**" rule, and it was working prior to v3.1, here is your assignment.

Note: The default assignment uses motors 1-4 which will cause conflicts if you attempt to reassign servos to motors 5-6. On F1 boards, assign the Servos output

to motors #1 & 2 to avoid timer conflicts and reassign Motors 1-4 to previous 3-6 pin assignments to prevent timer issues.

```
resource motor 1 none
resource motor 2 none
resource servo 1 a8
resource servo 2 a11
resource motor 1 b6
resource motor 2 b7
resource motor 3 b8
resource motor 4 b9
save
```

Example 2: SPRacing F3 Controlling External PWM triggered Buzzer

This example is for controlling PWM triggered buzzer on Matek 5-in-1 PDB, but can be applied to other cases.

Objective: To control an external PWM controlled device (Matek 5-in-1 PDB buzzer), with SPRacing F3, using `IO_1[4]` (RC CH2) with AUX2 switch.

This can be accomplished by one of two ways:

> A. (Available for all v3.1) Use SERVO_TILT and assign AUX channel of your choice to servo 0 on the Servo tab.
> B. (Limited to v3.1.5 and later) Use CHANNEL_FORWARDING and set CLI variable channel_forwarding_start to your switch channel number.

Option A is recommended if gimbal servo is not used and number of devices are less than or equal to two (SERVO_TILT can only control two channels).

Here's step by step for option A, controlling `IO_1[4] (RC CH2)` with AUX2:

(1) BF3.1 does not configure any servo for you; you have to assign it explicitly.

(1-1) Make sure you are not using PPM input (CH2 shares a timer with PPM).

(1-2) Use following CLI command

```
resource pwm 2 none
resource servo 1 a1
save
```

(2) Turn on `SERVO_TILT` (And turn off `CHANNEL_FORWARDING` if you have it on.)

(3) On Servo tab, check `A2` on `servo 0`. Resource command above specified `servo 1`, but this number is 1-origin, servo tab servos are 0-origin, so these are the same.

(4) At this point, you should be able to confirm `servo 0` on Motor tab responding to AUX2 switch inputs.

(5) And if you connect your external device to `IO_1[4]`, you should be able to confirm it is working.

Example 3: SPRacing F3 EVO Controlling External Still Camera Shutter

Motor output 8 was used

```
resource servo 1 b1
```

Example4: Servo for Tri-Copter on an F3 board (thanks Bking1340)

Just to give some feedback how I got my tricopter servo to work 100% on a F3 board(Xracer F303) and betaflight 3.1.7

Motor 1-3 are connected to motor pin 1-3. I don`t want to draw power from the board with my servo, so the + - of the servo goes to a 5v ubec and the signal goes to motor pin 8 (On F3 boards Servos output to motors #7 & 8 to avoid Timer conflicts). Now you must assign your servo in cli: If you type 'resource' in the CLI, you will see that motor 8 are on A03, but Serial_Rx 2 are also on A03 What I did is:

```
Resource motor 8 None
Resource Serial_Rx 2 None
Resource servo 1 A03
```

And my servo are working, I did not have to change directions on servo or epa etc. I'm no pro and still don't know what Serial_Rx 2 are doing, but everything is working correctly in the last 2 flights. I have a taranis with a x4r on Sbus and smartport - pids are showing on my taranis as well as battery voltage - seems like I don't use Serial_Rx 2. I halved the P and I 50% from stock on yaw, but the tail were loose, went back to stock and it`s rock solid with no oscillation.

Example 5: Setup for Matek F411 mini for Tricopter in Betaflight 3.2.2 by Flashted

If Firmware update is needed on F411, it is best done in the CLI.

I have found these instructions work with the Betaflight 3.2.2, Something has changed in the higher versions that have made changes to how the tail servo

outputs, and this method does not work with Betaflight 3.4.1, or higher.

I have not tried between version 3.2.2 - 3.4.0 either. I am at the moment trying to figure out what has changed, and why, but, if you want to try with higher versions, try it, and try to figure it out. Betaflight 3.2.2 works with a minimum of effort to get your craft flying.

In the future, I want to experiment with GPS rescue for tricopter, which Betaflight 3.2.2 does not support, so I am going to go forward to version 3.4.1 which supports GPS rescue, and start there.

If you power down the board, press the bootloader button on the Matek F411 flight controller, and plug back in like a lot of other boards, it does not enter bootloader mode, and there is NO blinking LED confirming bootloader mode, at least with my windows 10 laptop. It does not work.

If computer can't detect it, reinstall DFU drivers with zadig or use this https://impulserc.blob.core.windows.net/utilities/ImpulseRC_Driver_Fixer.exe

To enter bootloader mode in CLI:

In CLI

Type

bl

enter

Then bootloader mode is enabled, and newer firmware can be flashed.

Do a full CLI dump, to have a reference to the original file before making ANY changes. You may need a fixed starting point in case you have to start over. You must assign it to be a tricopter "first", and then "save and reboot" it as a tricopter so that all resources, and their designations will show up in the CLI.

Setup:

The first servo output channel is used for tail servo The first motor output channel is used for tail motor The second motor output channel is used for right motor The third motor output channel is used for left motor.

The 2 front motors face forward and (tail) rear motor / servo facing towards you.

On the board:

Motor 1 - (rear) to motor output pin S1

Motor 2 (right) to motor output pin S2

Motor 3 (left) to motor output pin S3

Motor 4 motor output pin S4 is free, as this is a Tricopter.

Motor pin 4 will not work for tail servo due to timing issue.

(Tail servo setup is explained below, keep reading)

If you need more outputs for servos, or as I needed strobes, motor 4 (S4) can't be remapped to servo if motor 3 (S3) is assigned to a motor. If S5 is used for a servo, S6 can be remapped to a servo also. I used S5 for telemetry in Taranis.

You can also use RSSI pad or LED pads for servo outputs.

I chose to try to disable motor 4, and noticed that I lost motor 2 when I tried to calibrate the esc's due to some Betaflight strangeness. I re-asigned motor 4 back to the original assignment, and motor 2 worked again, no problems. Go figure...

In the BetaFlight GUI in the Receiver Tab is a selection for how the connection to the model is setup.

Betaflight Defaults to AETR1234 Aileron, Elevator, Throttle, Rudder(yaw) Aux 1 2 3 4.

This is an accepted standard setup on Taranis.

In the Configuration tab:

feature SERVO_TILT

feature CHANNEL_FOWARDING

Need to be OFF.

Selecting TRICOPTER sets yaw output to servo automatically.

How to set the tail servo:

I chose to use pin 6 in my setup for tail servo, as pin 5 in my setup is used for telemetry, as I stated earlier.

In the CLI:

Type

resource motor 6 none

save

Then the pin will be free.

Pin on board is S6, and B10 is the location of the pin.

Type:

resource servo 1 B10.

save

Nothing needs to be changed in the Servos tab for movement.

Do not draw power from the board with a servo especially on a tricopter.

Positive, black negative are going to a pdb 5v output.

The yellow or white wire, signal wire can be connected to motor pin 5, motor pin 6, 7 or 8, will work.

IF you want to disable the tail servo when it's not armed, go to the CLI.

Type:

set tri_unarmed_servo = OFF

save

Or if you want it on.

Type:

set tri_unarmed_servo = ON

save

Now check if your servo/motor are tilting in the correct direction.

If you move your yaw stick to the left, the motor must tilt to the right. If not, there are 2 ways to fix this.

In CLI:

Type

Set yaw_control_direction = -1

save

If you quickly move your tail to the right the motor must tilt quickly to the left, and vice versa.

You can just reverse the yaw direction on your Tx, but it is better to do it in the Betaflight App, so the flight controller does not have to process the data.

Set the endpoints in the Betaflight App so you have 40 degrees deflection in both directions, and neutral at as level as you can get it.

An analog servo works as will a digital one in most cases my directions were not reversed with an analog servo, but may be with a digital.

Hopes this helps.

Cheers!!!

New Example Place Holder

Working on your own

There are several rules for successful servo MCU pin assignment.

1. **The MCU pin must be connected to an accessible board pad or through-hole** There may be a free unused MCU pin on your board, but without the pin broken out as an accessible pad or through-hole, you will end up with horrendous work of breaking out MCU pins by micro soldering.
2. **The MCU pin must have a timer** MCU pins are associated with different functions. Some are capable of working as UART signal pins, some are capable of working as I2C or SPI bus signal pins, and yet some are only capable of driving the pin logic high or low. Servo outputs need the timer function. Pins associated with timer can be found out by executing `resource list` command and looking for "MOTOR" or "PWM" entries.
3. **The timer should not collide with other uses** This rule requires a bit of explanation. An MCU pin associated with *timer* is connected to a *timer channel* of the *timer*. Each *timer* has it's own *time base* (or clock), and all *timer channels* on the same *timer* behaves according to the *time base*. Think, for example, a

group of people looking at a (physical) clock with only one hand. Each is given a task of yelling out loud when the second hand advanced certain number of times. If you give them a clock with a hand advancing every second, they will yell based on seconds. Now think of another group of people and give them a clock with a hand that advances once every hour. They will be yelling based on hours. You can NOT mix someone told to yell at every X seconds with someone told to yell at every Y hours in a same group.

(To continue...)