# Fetch_PWM.c Report

## Usage

The Pulse Width Modulation (PWM) fetch function the marionette board mirrors the functions of the other fetch commands and has the following options:

- Config
- Start
- Step
- Reset
- Help

These functions and their parameters are shown in figure 1 below. Currently only two pins are set to be used as PWM. These are GPIOE_PIN13 and GPIOE_PIN5. These pins use PWM1 and PWM9 respectively.

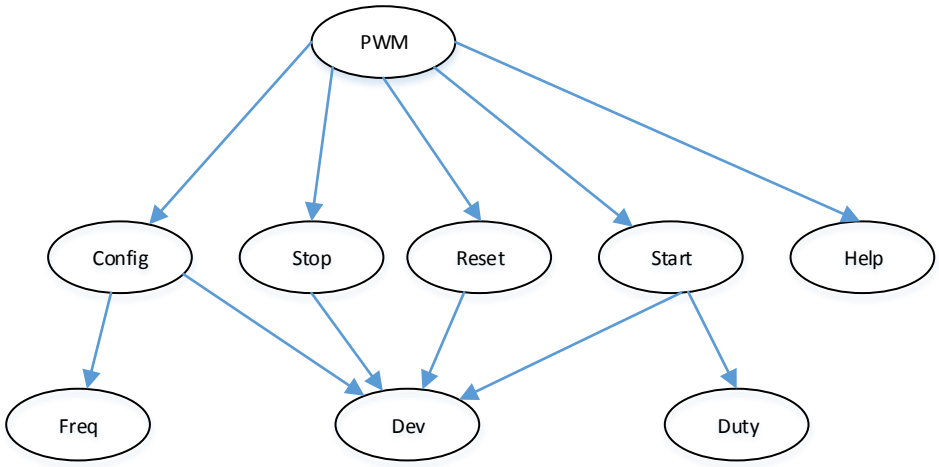


**Figure 1: Fetch_CAN Flow Chart**

The end user can configure PWM by selecting a device and a frequency. The frequencies available are currently limited to 1,2,3, 4, 5 Hz then range from 10 Hz to 1MHz in decade steps.

The start functionality allows the end user to select the device (Either 1 or 2 for GPIOE_PIN13 and GPIOE_PIN5 respectively) and the desired duty cycle of the pulse. This duty cycle is percentage based and can be any integer value between 1 and 99. ChibiOS allows for values to be chosen in the milli-percentage range but this has not been implemented in this revision.  Changing the duty cycle of the pulse does not require the PWM to be stopped first so it can be changed as needed.

The stop function of the PWM requires only the device number that the end user wishes to stop (Currently 1 or 2). Once stopped the user can reconfigure the device with a new frequency using the config function or start the pwm with a new duty cycle via the write function.

The reset function for PWM only requires the device (1 or 2) to be reset. When this function is called the driver for the device entered will be stopped and its pins will be set to their default states (input floating). Once reset the device will need to be configured again in order to use the write function.

The help function prints all of the PWM functions listed above with their required and optional parameters

## Registers and Configuration Structs

Fetch_PWM uses the PWMConfig struct for configuration. This struct has the following data members:

- uint32_t frequency – the timer clock Hz
- pwmcnt_t period – the PWM period in ticks
- pwmcallback_t callback – the periodic callback pointer (Set to NULL)
- PWMChannelConfig channels[PWM_CHANNELS] – The channel description
- uint16_t cr2 – the CR2 initialization register (Set to 0)
- uint16_t bdtr – the break & dead time register initialization data (Set to 0)

Of these six data members only frequency, period, and channels is used. Channel is a two dimensional array with an output (or complimentary output) and a callback if needed (currently set to NULL). To output on the desired channel of the timer PWM_OUTPUT_ACTIVE_HIGH (PWM_COMPLIMENTARY_ACTIVE_HIGH for the complimentary outputs). The frequency of the PWM output is determined by the frequency data member divided by the period data member. Frequency cannot be higher than then the twice the clock speed of the APBx bus that is on. If on APB1, the frequency cannot be greater than 84 MHz. If the PWM timer is on APB2, then the maximum frequency cannot be greater than 168 MHz. The period data member must be a multiple of two in order for a PWM to output (Need to check on this, but let's assume this for the time being).

## Updates for Future Revisions

Only two pins have PWM functionality and they have been taken from the GPIO banks. We can allocate more by making corrections to io_manage_defs.h. Currently the maximum output is set to 1 MHz.