****

Library 2 Components

19 April, 2024

Olivia Beattie, Isaac Garon, Pedro Pedro-Cristobal, and Chris Hernandez

# PWM SET

## E4235\_PWM\_SET

| int E4235\_PWM\_SET ( int GPIO, int FREQ, int DUTY ) |
| --- |
| **Overview**  This function controls the parameters for a hardware PWM on a GPIO at the specified frequency and duty cycle. |
| **Parameters**  **GPIO** - the GPIO to run the PWM on  **FREQ** -the frequency of the signal  **DUTY** -the duty cycle of the signal |
| **Returns**  **error\_code** - an integer that describes if the function executed successfully or not  0 - successful set of PWM signal  1 - GPIO is an invalid pin  2 - FREQ is out of range  3 - DUTY is out of range |

## 

**Description**

This function sets the parameters of a hardware PWM signal on one of the Pi’s four PWM pins: GPIOs 12, 13, 18, and 19. The user can choose a frequency and duty cycle to run on any of the four PWM pins. The frequency input can be anything greater than 5 Hz; however, the square wave will typically break down past 10MHz. The duty cycle can be any integer between 0 and 100 and represents what percent of the cycle is high. A duty cycle of 0 represents a continuously low output while a duty cycle of 100 represents a continuously high output. The user would use this function to generate a square wave or drive a load at a lower voltage. For example, the user might wish to drive a motor at a variable speed; choosing different duty cycles would change the speed of the motor.

An error code 1 is set when the GPIO input is not 12, 13, 18, or 19. An error code 2 means an integer less than 5 was input. Finally, error code 3 means that the duty cycle was less than 0 or greater than 100.

**Examples**

E4235\_PWM\_SET(12, 1000, 50)

Sets GPIO 12 to output a PWM signal with 1000 Hz frequency and a duty cycle of 50% on and 50% off.

E4235\_PWM\_SET(13, 500, 25)

Sets GPIO 13 to output a PWM signal with 500 Hz frequency and a duty cycle of 25% on and 75% off.

# PWM Enable

**E4235\_PWM\_Enable**

| void E4235\_PWM\_Enable (int GPIO, int enable) |
| --- |
| **Overview**  This function enables or disables the PWM output for a given GPIO. |
| **Parameters**  **GPIO** - the pin that is being enabled or disabled  **enable** - enables or disables the PWM output |
| **Returns**  **error\_code** - an integer that describes if the function executed successfully or not  0 - successful enable/disable of a GPIO  1 - GPIO is an invalid pin  2 - enable is out of range |

**Description**

This function enables/disables a PWM output on a given GPIO pin. The GPIO can be one of the four PWM pins: 12, 13, 18, or 19. Enable is an integer that determines the output state of the GPIO pin. 0 disables the PWM output while any positive value enables the PWM output (typically 1 is used to enable the output). The user would use this function to turn a PWM signal on or off for a given GPIO. All PWMs are initialized off on Pi bootup.

A note on the BCM2711 PWM implementation: The Pi PWM pins are run by two channels: channel 0 runs pins 12 and 18 while channel 1 runs pins 13 and 19. Therefore, if the user enables/disables pin 12, it will also enable/disable pin 18 since the code is actually enabling/disabling PWM channel 0. In all, there can only be two separate PWM outputs at any given time: one on pins 12 and 18 with channel 0 and one on pins 13 and 19 with channel 1.

An error code 1 is set when the GPIO input is not 12, 13, 18, or 19. Error code 2 is set when a negative value is entered for the enable parameter.

**Examples**

E4235\_PWM\_EN (12, 0)

Disables pin 12

E4235\_PWM\_EN (13, 1)

Enables pin 13

E4235\_PWM\_EN (12, 1); E4235\_PWM\_EN (18, 0)

Disables the output on pins 12 and 18 (the outputs of 12 and 18 are tied together).

# Test Plan

## E4235\_PWM\_Set

The following test cases will be executed to verify functionality of the E4235\_PWM\_Set function. The tests will check the function under nominal conditions and edge cases.

Case 1: Valid Input

This case checks that the function allows valid parameters and implements the desired PWM given the user input.

Implementation:

The PWM set functionality was tested by writing a program that checks for user input and then prints returns the error code 0 that will notify the user that the function successfully outputs the PWM. The following shows how to call the function within an assembly program and a C program.

Assembly Function Call:

mov r0, #13

mov r1, #1500

mov r2, #40

bl E4235\_PWM\_Set

C Function Call:

E4235\_PWM\_Set(13, 1500, 40);

Verification:

1. Run the program
2. Enter 13 to indicate pin channel, 1500 for desired frequency, and 40 for desired duty cycle
3. Ensure that the proper error code is returned and oscilloscope shows user desired output

Results: PASSED

When running the program and inputting the proper parameters, the correct error code 0 was returned and the oscilloscope showed a 1500 Hz, 40% duty cycle square wave on pin 13.

Case 2: Invalid Input

This case checks that the function returns the proper error code given invalid inputs.

Implementation:

The PWM set error code checks were tested by utilizing the same program in Case 1, but passing in invalid inputs that should return an error code. The following test cases will be run:

Error Code 1: Invalid GPIO Input

Assembly Function Call:

mov r0, #10

mov r1, #1500

mov r2, #40

bl E4235\_PWM\_Set

C Function Call:

E4235\_PWM\_Set(10, 1500, 40);

Error Code 2: Invalid Frequency Input

Assembly Function Call:

mov r0, #12

mov r1, #0

mov r2, #40

bl E4235\_PWM\_Set

C Function Call:

E4235\_PWM\_Set(12, 0, 40);

Error Code 3: Invalid Duty Cycle Input

Assembly Function Call:

mov r0, #12

mov r1, #100

mov r2, #150

bl E4235\_PWM\_Set

C Function Call:

E4235\_PWM\_Set(12, 100, 150);

Verification:

1. Run the program
2. Input the given invalid inputs to the main program
3. Ensure that the proper error corrections are returned

Results: PASSED

After the execution of the assembly program and C programs, the proper error codes were output that showed the user that an invalid input was inputted.

## E4235\_PWM\_Enable

The following test cases will be executed to verify the functionality of the E4235\_PWM\_EN function. The tests will check the function under both nominal conditions and edge cases.

Case 1: Valid Input

This case checks that the function successfully enables the given pin number that will be utilized to set the PWM.

Implementation:

The PWM enable function was tested by writing a program that takes in valid user input and then successfully runs the PWM set function. The following are example/test cases that are valid ways to test the function:

Assembly Function Call:

mov r0, #12

mov r1, #1

bl E4235\_PWM\_EN

mov r0, #12

mov r1, #1500

mov r2, #40

bl E4235\_PWM\_Set

C Function Call:

E4235\_PWM\_Set(12, 1500, 50);

E4235\_PWM\_En(12,1);

Verification:

1. Run the program
2. Input the given valid inputs to the functions called
3. Ensure that a PWM signal is sent to GPIO 12, with the given user parameters

Results: PASSED

When the program was run with the valid inputs, the program successfully returned the correct error codes and outputted the desired PWM signal.