

# Firmware Fan Control Example Project

## 1.20

## Features

- Full featured fan control application.
- Control completely in hardware.
- Fan Stall and Speed Regulation Failure alerts

## General Description

This example project demonstrates usage of the Fan Controller component in firmware control mode. Firmware fan control means that firmware is responsible for reaching and maintaining the desired speed.

## Development Kit Configuration

The following configuration instructions provide a guideline to test this design. For simplicity, the instructions describe the stepwise process to be followed when testing this design with the PSoC Development Kit (CY8CKIT-001) board, but can be generalized for the PSoC 3 Development Kit (CY8CKIT-030) and PSoC 5 LP Development Kit (CY8CKIT-050) as well.

1. Set LCD power jumper J12 to ON position and leave the rest of the board at default configuration.
2. Connect two 4-Wire Fans to the appropriate pins as shown in Figure 1. Note that if using the PSoC Thermal Management EBK (CY8CKIT-036), this can be connected to Port A on the CY8CKIT-001 or Port E on CY8CKIT-030/050. For the CY8CKIT-036, modify the project design-wide resources to reassign the fan PWM and tachometer pins from P0 to P3.
3. Connect P1[6] and P1[7] to SW1 and SW2 on the board. For the CY8CKIT-030/050, reassign these pins to P6[0] and P15[5].
4. Ensure that the Character LCD is connected to LCD header on the development board.

## Project Configuration

The TopDesign schematic looks as shown in Figure 1 below. The Character LCD is configured in its default mode. The FanController is set to Firmware Control. In the 'Fans' tab, 2 10-bit PWM outputs are defined. It is crucial to enter two data-points from the duty-cycle-to-RPM curve corresponding to each fan being controlled. These values are typically provided by the fan manufacturer and documented in the fan datasheet. SW1 and SW2 are chosen as digital input pins, and control the RPM of the fans. The tach\_1 and tach\_2 digital

input pins serve as indicators of the Fan RPM; while digital output pins, pwm\_1 and pwm\_2, drive the two fans. Finally, the 1-bit Status Register is configured to be sticky so that the EOC pulse is not missed.

## Firmware Fan Control

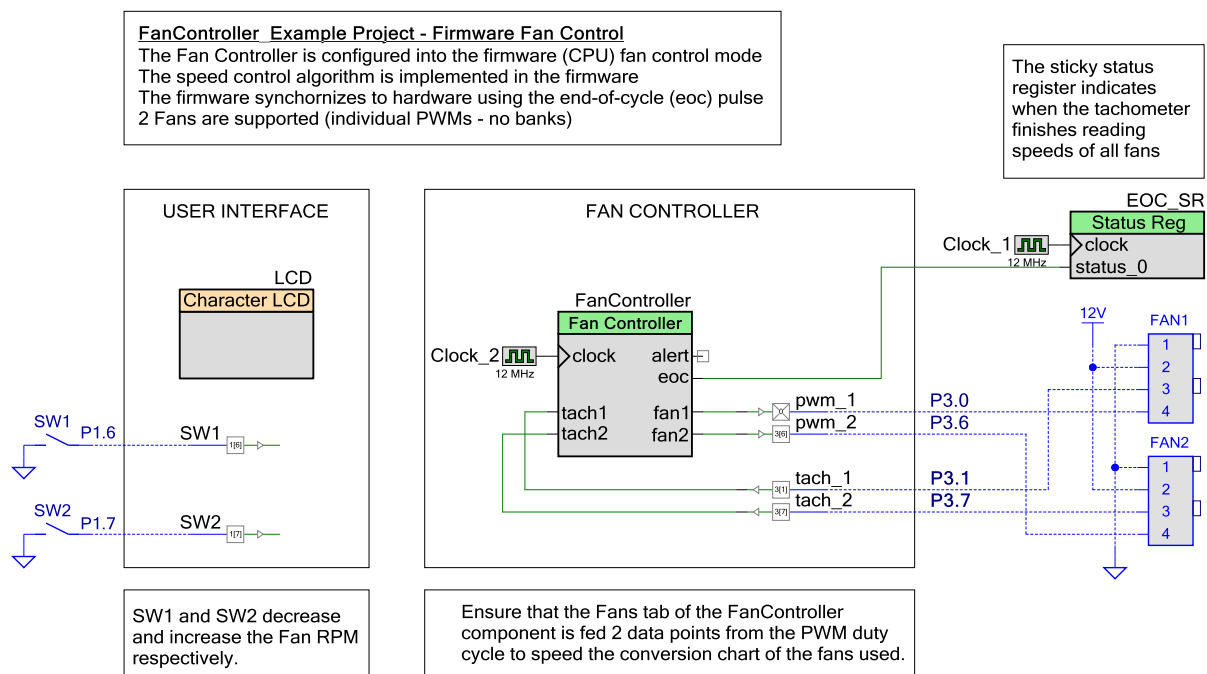


Figure 1. TopDesign schematic

Configure 'FanController'

Name:

**Basic** Fans Built-in

Fan control method

☒ Manual

☐ Automatic

☒ Firmware (CPU)

☐ Hardware (UDB)

Control loop period (sec):

Tolerance:

☐ Acoustic noise reduction

Alerts

☐ Fan stall / Rotor lock

☐ Speed regulation failure

Connections

☐ Display as bus

☒ External clock

Datasheet OK Apply Cancel

Configure 'FanController'

Name:

**Basic** **Fans** Built-in

Motor support

☒ 4-pole motors

☐ 6-pole motors

PWM output configuration

Number of fans:  Number of banks:

PWM resolution:  PWM frequency:

Fan number	Enter 2 datapoints (A, B) from duty cycle to RPM curve for each fan/bank.				Initial RPM
	Duty cycle A (%)	RPM A	Duty cycle B (%)	RPM B	
1	<input type="text" value="35"/>	<input type="text" value="4500"/>	<input type="text" value="70"/>	<input type="text" value="10500"/>	<input type="text" value="4500"/>
2	<input type="text" value="35"/>	<input type="text" value="4500"/>	<input type="text" value="70"/>	<input type="text" value="10500"/>	<input type="text" value="4500"/>

Datasheet OK Apply Cancel

Figure 2. FanController component configuration

## Project Description

In the main.c file, the FanController and LCD components are started and the desired initial RPM is set for both fans. The speed control algorithm then begins to monitor fan speed for each fan via the `FanController_GetActualSpeed()` API. If the measured fan speed is different from the desired speed, the algorithm will adjust the fan speed using the `FanController_SetDutyCycle()` API. The amount of adjustment depends on the difference between the current speed and the desired speed. This process is repeated until the measured fan speed equals the desired fan speed.

## Expected Results

The algorithm implemented in `main()` will maintain the desired speed selected via SW1 and SW2.

SW1 decreases fan speed, SW2 increases fan speed. The LCD Display shows the current desired speed (upper row, far left value) and the current measured speed and duty-cycle setting for each fan.

5	5	0	0		5	5	7	3		4	2	.	8	6	%
F	/	W			5	5	5	9		4	3	.	1	6	%

Figure 3. Expected LCD display

## Related Material

### Application Note

- [AN66627 - PSoC® 3 and PSoC 5 Intelligent Fan Controller](#)



Cypress Semiconductor  
198 Champion Court  
San Jose, CA 95134-1709

Phone : 408-943-2600  
Fax : 408-943-4730  
Website : [www.cypress.com](http://www.cypress.com)

© Cypress Semiconductor Corporation, 2012. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges. PSoC® is a registered trademark, and PSoC Creator™ and Programmable System-on-Chip™ are trademarks of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

This Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and/or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.

