# 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.

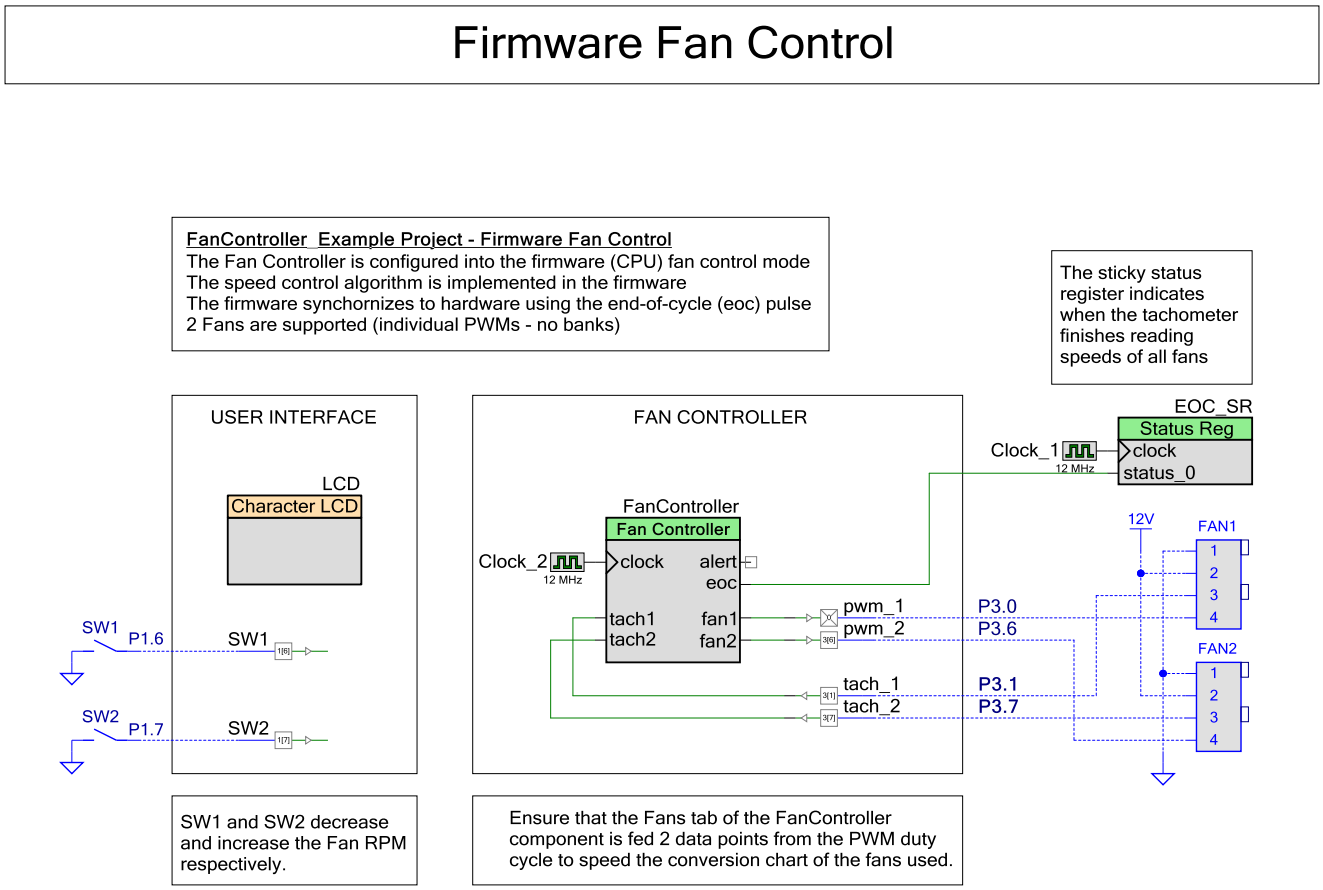
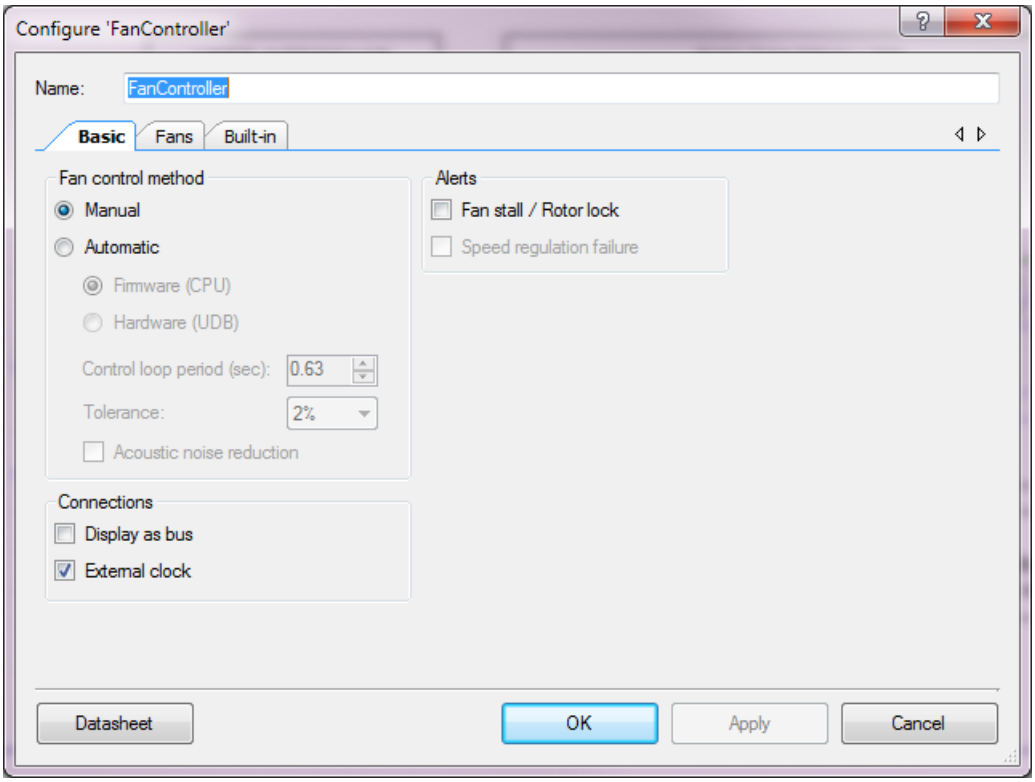



Figure 1. TopDesign schematic



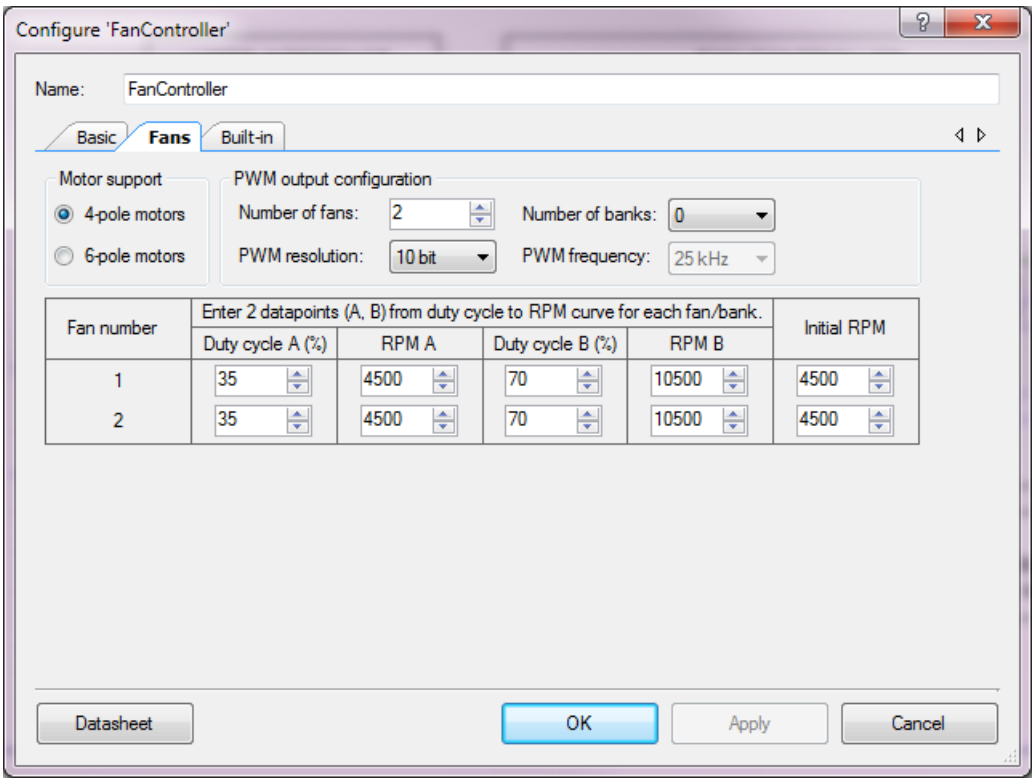




Figure 2. FanController component configuration

# 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.

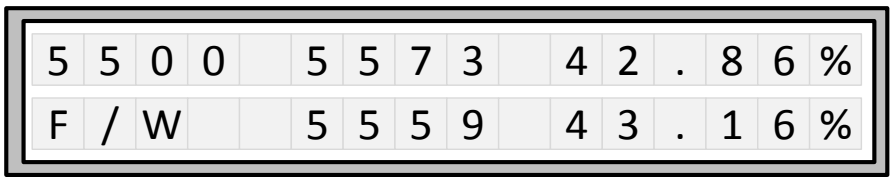



Figure 3. Expected LCD display

# 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