Application Note

Cooling FAN Controller with MSPM0

Gary Gao

ABSTRACT

There many cooling fans used in servers, networking equipment, or desktop computers. A cooling fan controller is normally used to control the fans under different operating conditions. There are two kinds of designs for cooling fan controllers: dedicated ICs and MCUs. The dedicated IC is simple to use but is slightly more expensive than some MCUs, and the IC cannot be modified to meet customer requirements. The MCU design is more flexible. This application note discusses MCU designs based on the MSPM0 device and provides a demo code for customers to quick start with such applications.

Key features supported in this demo:

- Supports up to four (MSPM0C1104) and six (MSPM0H3215/MSPM0C1106) 4-wire FANs control
- PWM duty cycle and frequency control
- TACH detection
- I2C interface for configuration
- ADC to sample temperature sensors
- FAN failure detection

Click here to download the software.

Table of Contents

Trademarks

1 Four Wire Cooling Fans

Four wire cooling fans have better performance and higher reliability than three wire or two wire cooling fans. Four wire cooling fans are popular in the cooling system in the server or PC. The four wire cooling fan signals are listed in Table 1-1.

Table 1-1. 4-Wire Cooling Fan Signals

Wire Color	Signal	Details
Black	GND	
Red	VCC	Power supply to the FAN, normally 12V
Yellow	TACH	Speed signal, normally open drain output
Blue	PWM	Speed control; pulled up to 5V in the fan module or open drain input

Change the PWM duty cycle to control the speed of the fan. The TACH signal is used to determine the actual speed of the fan. Use a Hall sensor in the fan module to generate one or more pulses to turn a circle. The number of pulses depends on the pole-pare numbers of the motor.

2 Cooling FAN Controller with MSPM0 Designs

This application note provides two different designs based on different MSPM0 devices: MCUs (MSPM0C1104 and MSPM0C1106) and 5V powered MCUs (MSPM0H3215 and MSPM0H3216).

2.1 Design with MSPM0C1104 and MSPM0C1106

For the fan module where the PWM signal is an open drain IO, TI recommends using a MSPM0C1104 device or a MSPM0C1106 device. Figure 2-1 shows the overview of the design.

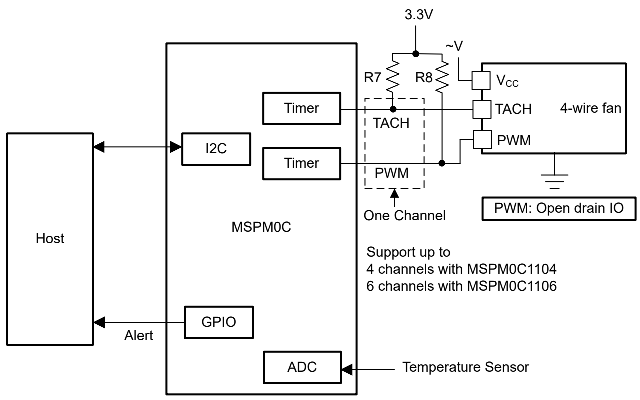

Figure 2-1. Cooling Fan Controller Design With MSPM0C1104 or MSPM0C1106

These devices are powered from 1.62V to 3.6V and 8-16kB flash. There is a I2C interface that can support up to a 1MHz clock frequency. The devices feature a 12bit ADC and three timers that can support up to 10 PWM outputs or captures. The devices can support to control four 4-wire fans.

MSPM0C1105 and MSPM0C1106 are also device options. These devices are powered from 1.62V to 3.6V and 32-64kB flash. There is a I2C interface that can support up to a 1MHz clock frequency. The interface features a 12 bit ADC and five timers that can support up to 14 PWM outputs or captures. The interface can support to control six 4-wire fans.

The TACH and PWM signals are both open drain IO. Pull up to 3.3V and connect to the capture and channel of the MSPM0C directly.

The temperature sensor can be a NTC or linear thermistor such as TMP61.

The I2C interface can be used to perform all configurations such as changing PWM duty and PWM frequency, reading the capture value of the TACH, reading ADC results and so on. For the communication protocol details, see *Function Description*.

There is a GPIO output to alter host that detects fan fault conditions.

2.2 Design with MSPM0H3215 and MSPM0H3216

For the fan module that the PWM signal is 5V pull up, TI recommends using MSPM0H3215 and MSPM0H3216. The overview of the design is shown in Figure 2-2.

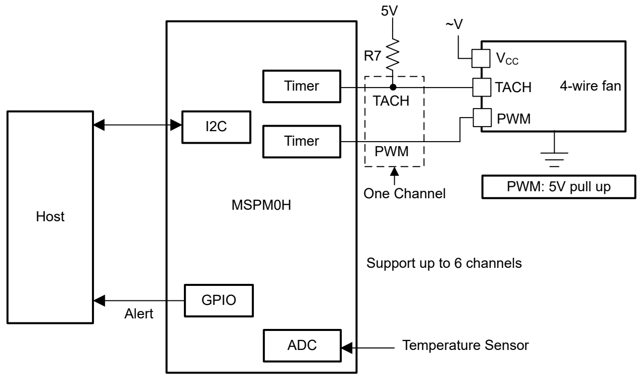

Figure 2-2. Cooling Fan Controller Design with MSPM0H3215 and MSPM0H3216

MSPM0H3215 and MSPM0H3216 is TI's first 5V MCU. The devices can be powered from 4.5V to 5.5V and 32-64kB flash. There are two I2C interfaces, a 12 bit ADC and five timers that can support up to 14 channels PWM output or captures. The interface can support to control six 4-wire fans.

The interface can save the transistor and switch on the PWM control channels when the PWM signal of the fan module is 5V and pulled up in the fan module.

3 Software Introduction

3.1 Software Working Flow

The software working flow is shown in Figure 3-1.

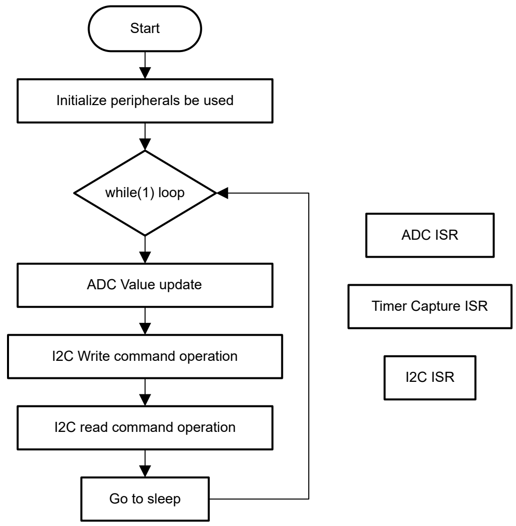

Figure 3-1. Software Working Flow Overview

In the software, the I2C command data received or send in the I2C ISR and is active in the while loop. Use the TACH signal and fan fault detection for the timer ISR.

3.2 Function Description

3.2.1 PWM Output

To start a PWM in a specific channel, the output can write a non-zero duty cycle into the specific duty cycle register (start from 0x20). To stop the PWM output, write zero duty cycle to the specific channel duty cycle register. The duty cycle valid bits for this demo is 9 bits and using two registers to identify one duty cycle in the channel. In this demo all the registers are one byte length.

This demo can also support to change the frequency of the PWM in the PWM frequency register (start from 0x10). Currently, the demo can support four different frequencies in the demo: 22.9Hz, 45.8Hz(default), 91.7Hz and 23.4kHz. The customer can change different frequencies as required in the code. Four bits can be used to configured different frequencies in one register, so one register can be used to configure two different timers. In the demo code, some PWM channels are generated in one timer. These PWMs can use the same frequency.

3.2.2 TACH Capture

The TACH signal is generated from the fan module and captured by the timers in the MSPM0. The timer capture the period of the pules and can be read from the register starting at 0x30. Two registers here show one capture value. For example 0x30 is the MSB of FAN1 and 0x31 is the LSB of FAN1. The valid bits for capture values are 16 bits. In this demo code, the capture timer frequency is configured as 8192Hz. Calculate the RPM with the capture value with Equation 1.

$$RPM = \left(\frac{8192}{NCAP} \times \frac{60}{Np}\right) \tag{1}$$

Ncap is the capture value.

Np: generated pulses turn a cycle.

Calcuate Tp as shown in Equation 2

$$Tp = Ncap/8192 \tag{2}$$

3.2.3 ADC Sample

In this demo code, provide an ADC channel to sample a signal from a temperature thermistor. The ADC value uses two registers to show and start from 0x40. In the test, the register using TMP61 on the LP-MSPM0L1306 to obtain the temperature of the environment. The ADC sample rate is approximately 1.6Hz that uses the zero event trigger of the capture timer of the ADC to start the sample. For this demo, the period time setting of the capture timer is 600ms.

3.2.4 FAN Fault Detection and Overflow Maximum Value

This demo code can support fan fault detection. If the fan PWM duty cycle is not 0, but there is no TACH signal capture for configured overflow maximum value of the overflow event or zero event of the TACH capture timer, the fan is in fault status and is recorded in the 0x60 register. For this register, one bit equates to the status of one fan. The overflow maximum value also can be configured at register 0x50. The default value is 3. That means if fan 1 started but there is no TACH signal is captured in 3 × 600ms = 1.8s, fan 1 is recorded as fault. Every time the fault detection register fault is read, the event clears. TI does not recommend reading the value two times within 1.8s, or the second time read value is always 0.

3.3 I2C Communication Protocol

For the I2C communication protocol, use Figure 3-2 using 0x40 as the address.

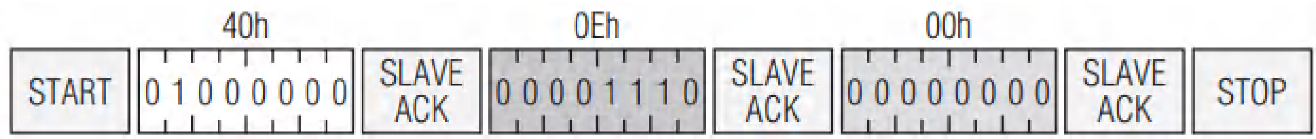

Figure 3-2. Write One Byte to Register 0x0E

Figure 3-3. Write Two Bytes to Register 0x0E

Figure 3-4. Write Two Bytes to Register 0x0E

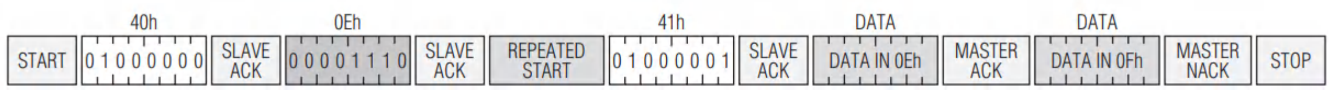

Figure 3-5. Read Two Bytes from Register 0x0E

The demo can only support one or two bytes in read or write mode.

3.4 Registers Definition in the Demo

Table 3-1. Registers Definition

R/W	Register	Reset Value	Function	Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7
R/W	0x10	0x11	PWM Frequency	0x0 = 22.9Hz 0xB = 20.8KHz				0x1 = 40.7Hz 0x3 = 81.5Hz 0xB = 20.8KHz			
R/W	0x20	0	FAN1 Duty Cycle	MSB							
R/W	0x21	0	FAN1 Duty Cycle	\	\	\	\	\	\	\	LSB
R/W	0x22	0	FAN2 Duty Cycle	MSB							
R/W	0x23	0	FAN2 Duty Cycle	\	\	\	\	\	\	\	LSB
R/W	0x24	0	FAN3 Duty Cycle	MSB							
R/W	0x25	0	FAN3 Duty Cycle	\	\	\	\	\	\	\	LSB
R/W	0x26	0	FAN4 Duty Cycle	MSB							
R/W	0x27	0	FAN4 Duty Cycle	\	\	\	\	\	\	\	LSB
R/W	0x28	0	FAN5 Duty Cycle	MSB							
R/W	0x29	0	FAN5 Duty Cycle	\	\	\	\	\	\	\	LSB
R/W	0x2A	0	FAN6 Duty Cycle								
R/W	0x2B	0	FAN6 Duty Cycle	\	\	\	\	\	\	\	LSB
R	0x30	0	FAN1 TACH	MSB							
R	0x31	0	FAN1 TACH	LSB							
R	0x32	0	FAN2 TACH	MSB							
R	0x33	0	FAN2 TACH	LSB							
R	0x34	0	FAN3 TACH	MSB							
R	0x35	0	FAN3 TACH	LSB							
R	0x36	0	FAN4 TACH	MSB							
R	0x37	0	FAN4 TACH	LSB							
R	0x38	0	FAN5 TACH	MSB							
R	0x39	0	FAN5 TACH	LSB							
R	0x3A	0	FAN6 TACH	MSB							
R	0x3B	0	FAN6 TACH	LSB							
R	0x40	0	ADC Value	MSB				\	\	\	\
R	0x40	0	ADC Value	LSB							
R/W	0x50	0x3	Over flow max	Must ≥ 1							
R/W	0x60	0	FAN Fault	FAN 1	FAN2	FAN3	FAN4	FAN5	FAN6	\	\
				0: No fault							

Note

Stand for reserved bits. MSPM0C1104 can support four fans. The registers of FAN5 and FAN 6 can be ignored.

4 Demo Test with Hardware

This application note provides the demo code based on MSPM0C1104 and MSPM0H2316. Because the MSPM0C1106 is pin-to-pin with MSPM0H2316, porting the code from MSPM0H2316 to MSPM0C1106 is simple.

4.1 Hardware Setup

Hardware requirements:

- 1-6 4-wire cooling fans
- LP-MSPM0L1306 as host device
- LP- MSPM0C1104 or LP-MSPM0H3216 as fan controller board
- Pull up circuits to pull up TACH or PWM signal
- Logic analyzer such as Seleae to capture signals
- Transistors or switches (Optional)

Table 4-1. Hardware Connection

Signal		Host Device	FAN Controller Device	
		LP-MSPM0L1306	LP-MSPM0C1104	LP-MSPM0H3216
I2C	SCL	PA1/I2C0_SCL	PA11/I2C0_SCL	PA11/I2C0_SCL
	SDA	PA0/I2C0_SDA	PA0/I2C0_SDA	PA0/I2C0_SDA
Temperature sensor	Power	J15-2	3V3	5V
	Vsensor	J1-2	PA27	PA27
Fan Fault Alert	GPIO	\	PA25	PA24
PWM	FAN1	\	PA26	PA23
	FAN2	\	PA2	PA18
	FAN3	\	PA16	PA10
	FAN4	\	PA23	PA6
	FAN5	\	\	PB14
	FAN6	\	\	PB13
TACH	FAN1	\	PA28	PA8
	FAN2	\	PA6	PA9
	FAN3	\	PA17	PB17
	FAN4	\	PA18	PA12
	FAN5	\	\	PB8
	FAN6	\	\	PA22

As mentioned above the temperature sensor is used the TMP61 on the LP-MSPM0L1306. Due to the calculation in the host demo code, TMP61 must be powered with the same power source with the ADC reference of the FAN controller device. (VDD). As a result, power the TMP61 from the FAN controller LaunchPad ™ listed in Table 4-1.

The TACH signal must be pulled up externally. In this test case, the signal pulls up with a 3k resistor to VDD.

For PWM signal, when the PWM channel of the fan module is open drain, the signal must be pulled up external that as same as TACH signal. If the PWM channel of the fan module is pull up to 5V internally, the channel can connect to the PWM control pin of the MSPM0H3216 directly. Resistors or switches must be added when using MSPM0C1104 and MSPM0C1106.

4.2 Software Setup

- Install CCS20.1.1 or up
- Install Sysconfig 1.23.0 or up
- Install SDK 2_04_00_06 or up
- Click the link to download the software

4.3 Running Demo Code

1. Connect the hardware as mentioned in previous sections.
2. Import the demo codes into CCS.
3. Open project FAN_Control_host_LP_MSPM0L1306 and comment out *#define MSPH3216_6FANs* if using MSPM0C1104 as the fan controller in the file fan_control_host.c. Comment the FANs operations code if the user is not connected to a FAN in hardware.
4. Build project FAN_Control_host_LP_MSPM0L1306 and download the project into LP-MSPM0L1306.
5. Build project FAN_Controller_MSPM0C1104 or FAN_Controller_MSPM0H3216 depending on which board is used and download the project into the board.
6. Keep both boards powered up, run the logic analyzer, and press the reset button on LP-MSPM0L1306 to resend the commands, or make the project of FAN_Control_host_LP_MSPM0L1306 start debug mode and start running debug mode. When all the fans stop after running, pause the code and add the *temp_c4* to watch the window to obtain the temperature result.
7. The fans starts running from FAN1 to FAN4/6 in turn and stopped at the same time.
8. Figure 4-1 shows signals captured with Saleae (The Saleae only has eight channels. Only a few signals were captured). Analyze the signals with I2C commands to verify if the Saleae matches expectations.

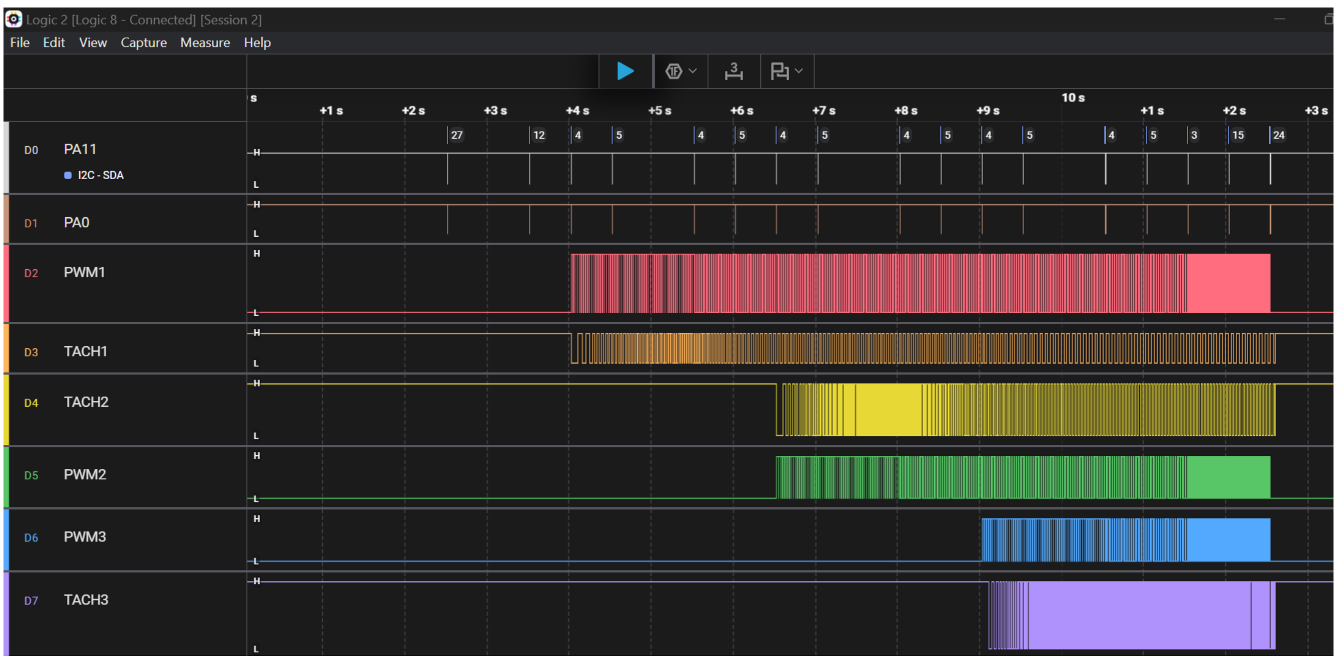

Figure 4-1. Few Test Results Captured by Saleae

5 Summary

This application note provides two different kinds of fan controller designs that are based on MSPM0C1104, MSPM0C1106, MSPM0H3215, and MSPM0H3216. Both of them are controlled by an I2C bus and can perform temperature sensing and fan fault detection. The functions verified functioned well during a final test.

6 References

- Texas Instruments, MSPM0 C-Series 24-MHz Microcontrollers, technical reference manual.