



CYPRESS

Мікропроцесорна техніка

(лекція 7)
Благітко Б.Я.
2019 р.

PSoC Creator 4.2
Designing with PSoC 3/5



CYPRESS

PERFORM



CYPRESS

PSoC@3/5 Fan Controller

PSoC Creator 4.2
Designing with PSoC 3/5



CYPRESS

PERFORM



CYPRESS

PSoC@3/5

Управління
швидкістю обертання ротора
двигунів постійного струму

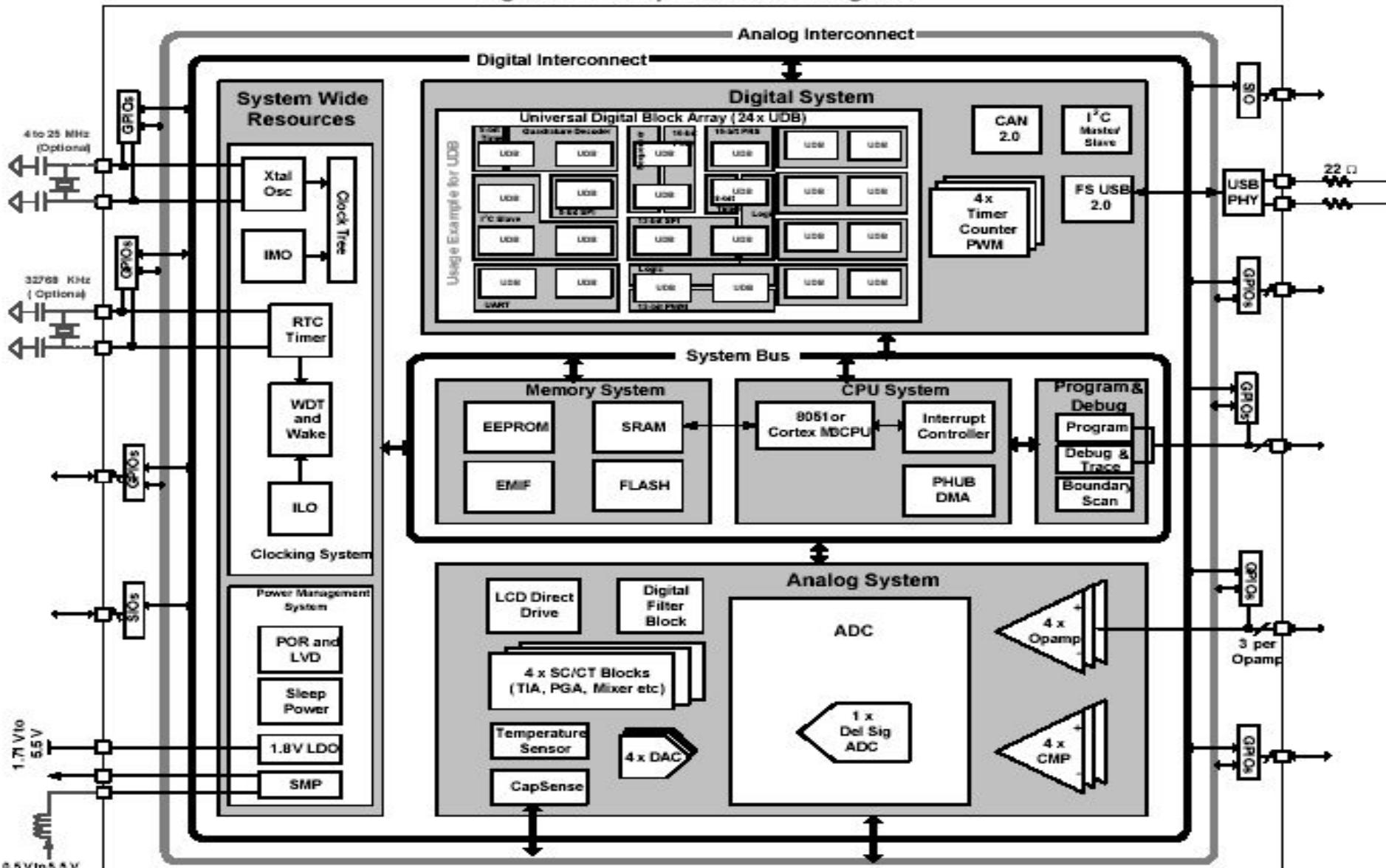
PSoC Creator 4.2
Designing with PSoC 3/5



CYPRESS

Цифрові модулі

Figure 1-1. Simplified Block Diagram





CYPRESS

Цифрові модулі

Figure 7-3. Component Catalog

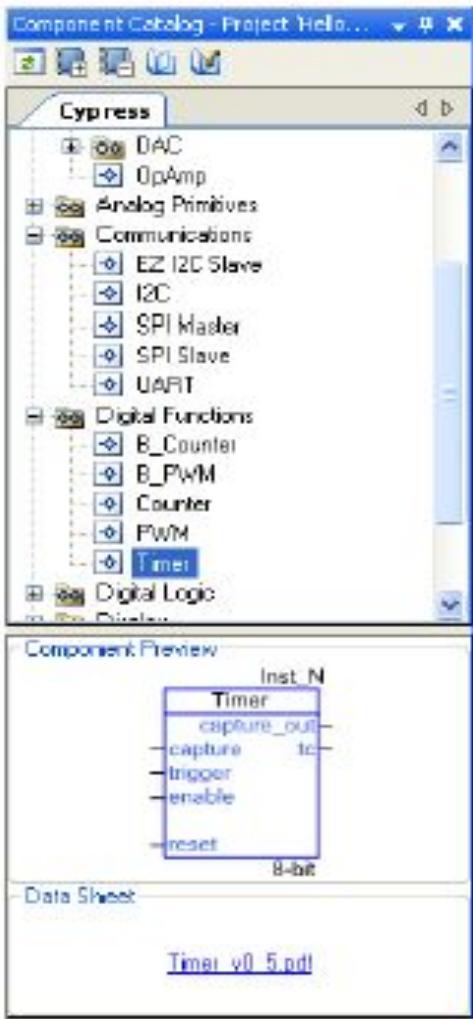
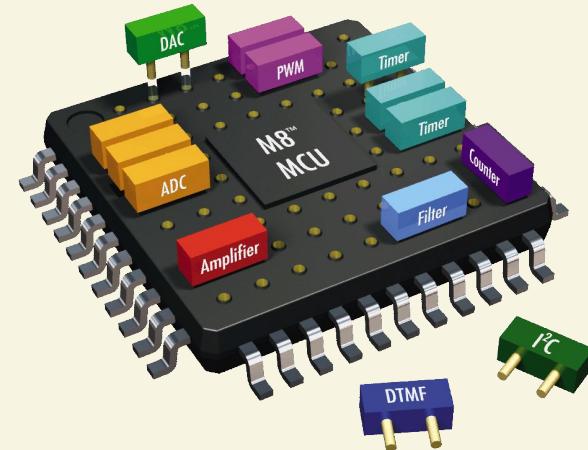
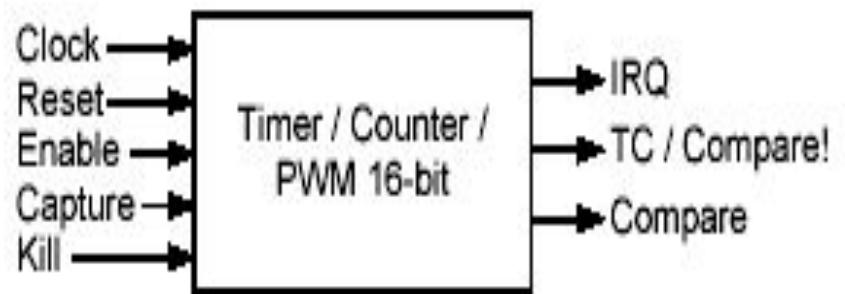


Figure 7-21. Timer/Counter/PWM





PWMs and Counters

PWMs and **Counters** share many capabilities but each provides specific capabilities.

When to Use a PWM

The most common use of the **PWM** is to generate periodic waveforms with adjustable duty cycles. The PWM also provides optimized features for power control, motor control, switching regulators and lighting control. The PWM can also be used as a clock divider by driving a clock into the clock input and using the terminal count or a PWM output as the divided clock output.

When to Use a Counter

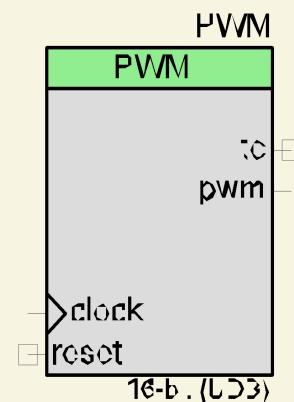
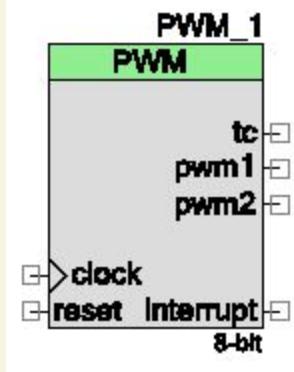
A **Counter** component is better used in situations that require the counting of a number of events but also provides rising edge capture input as well as a compare output.



CYPRESS

PWM

Output	May Be Hidden	Description
tc	N	The terminal count output is '1' when the period counter is equal to zero. In normal operation this output will be '1' for a single cycle where the counter is reloaded with period. If the PWM is stopped with the period counter equal to zero then this signal will remain high until the period counter is no longer zero. This output is synchronized to the block clock input of the component.
interrupt	Y	The interrupt output is the logical OR of the group of possible interrupt sources. This signal will go high while any of the enabled interrupt sources are true. The interrupt output shall remain asserted until the Status Register is read out by the software. In order to receive subsequent interrupts, the interrupt shall be cleared by reading the Status Register using the PWM_ReadStatusRegister() API. The interrupt output is not visible if the Use Interrupt parameter is not set. This allows the status register to be removed for resource optimization as necessary.
pwm/pwm1	Y	The pwm or pwm1 output is the first or only pulse width modulated output. This signal is defined by PWM Mode, compare modes(s), and compare value(s) as indicated in waveforms in the Configure dialog. When the instance is configured in one output, Dual Edged, Hardware Select, Center Aligned, or Dither PWM Modes, then the output "pwm" is visible. Otherwise the output "pwm1" is visible with "pwm2" the other pulse width signal. This output is synchronized to the block clock input of the component.

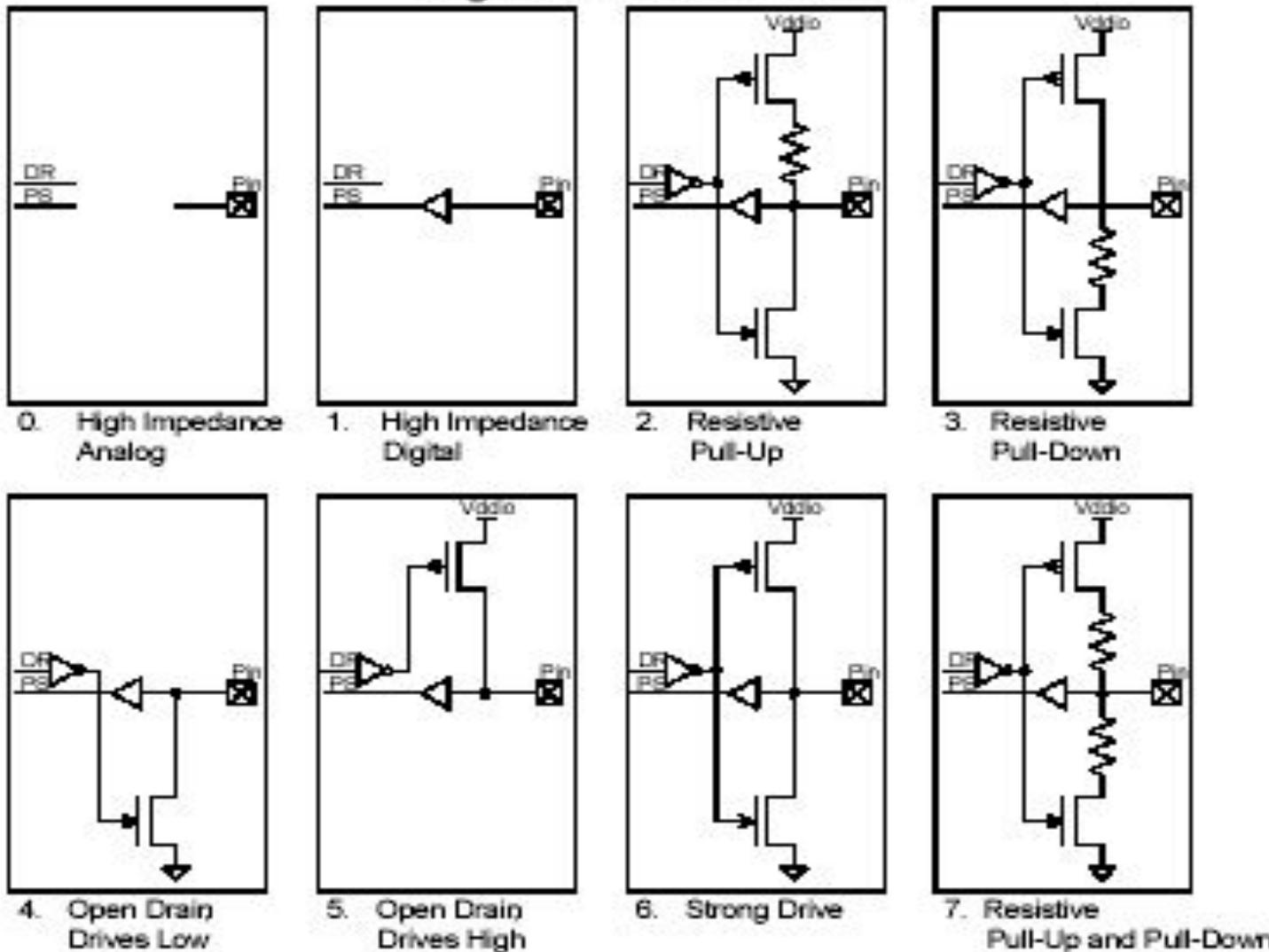




CYPRESS

Port's Pins

Figure 6-11. Drive Mode





How They Work

- A DC motor uses a commutator to supply continuous direct current to a coil placed between two or more magnets.
- A stepper motor receives pulses of electricity from a pulse generator and sends them through a similar coil-magnet setup.
- In both cases, the Oersted effect causes the coils to turn the motor's shaft.

Read more: [DC Motor vs. Stepper Motor | eHow.com](http://www.ehow.com/facts_6777501_dc-motor-vs-stepper-motor.html#ixzz2QKmmwrgy) http://www.ehow.com/facts_6777501_dc-motor-vs-stepper-motor.html#ixzz2QKmmwrgy



CYPRESS

Fan Controller PSoC@3/5

When to Use a Fan Controller

**Use the Fan Controller component
when you want to provide your
application with device.**



CYPRESS

Lab_2 PWM

Lab_2 - PSoC Creator 2.1 [D:\PSoC_3\Lab_2\Lab_2.cydsn\TopDesign\TopDesign.cysch]

File Edit View Debug Project Build Tools Window Help

Microsoft Sans Serif 10 B I U Debug

Workspace Explorer

Project 'Lab_2' [CY8C3866]

- TopDesign.cysch
- Lab_2.cydwr
- Header Files
- device.h
- Source Files
- main.c

Source Components Datasheets Results

Configure 'PWM'

Name: PWM

Configure Advanced Built-in

period 255 0 255 0

pwm1

pwm2

Implementation: Fixed Function UDB

Resolution: 8-Bit 16-Bit

PWM Mode: Two Outputs

Period: 255 Max Period = UNKNOWN SOURCE FREQ

CMP Value 1: 127 CMP Value 2: 63

Datasheet OK Apply Cancel

Notice List

2 Errors 0 Warnings

De... File

Page 1

Output

Show output from: All

Log file for this session is located at: C:\Documents and Settings\Admin

Component Catalog (174 co...)

Concept Cypress

- Analog MUX
- Comparator [v1.90]
- DAC
- Manual Routing
- Mixer [v1.91]
- Sample/Track and Hold
- VRef [v1.60]
- CapSense
- Communications
- Digital
 - Functions
 - Counter [v2.20]
 - CRC [v2.20]
 - Debouncer
 - Glitch Filter [v2.0]
 - PrISM [v2.10]
 - PRS [v2.10]
 - PWM [v2.20]
 - Quadrature Decode
 - Shift Register [v2.10]
 - Timer [v2.30]
 - Logic

Component Preview

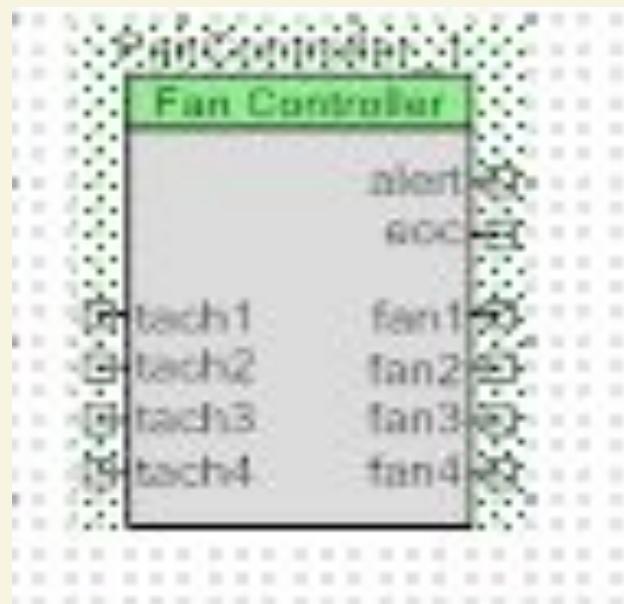
8 or 16-bit Pulse Width Modulator

Ready

Do_Present_2.doc - ... Lab_2 - PSoC Creator... EN 6:53

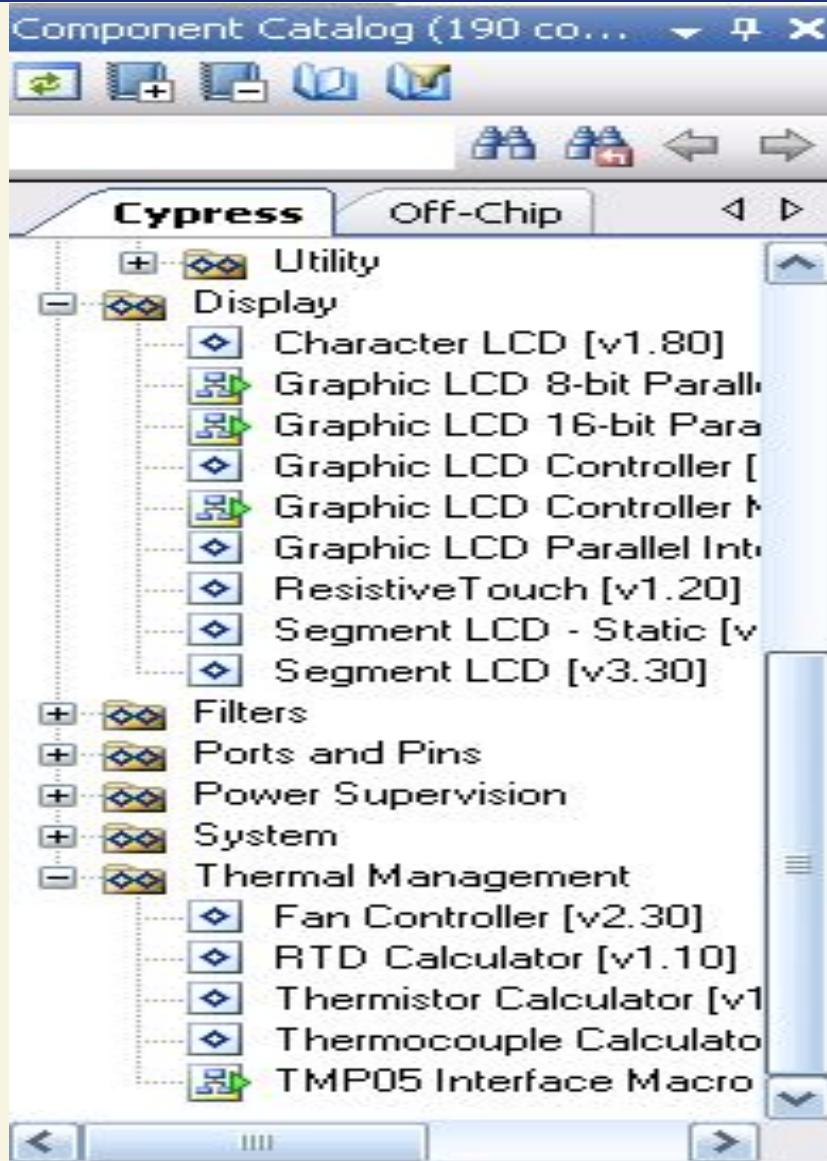
FanController

Компонента надає низькорівневий драйвер для управління швидкістю обертання ротора двигунів постійного струму . Крім того, ця компонента забезпечує для користувача настроювач, який полегшує налаштування дескрипторів



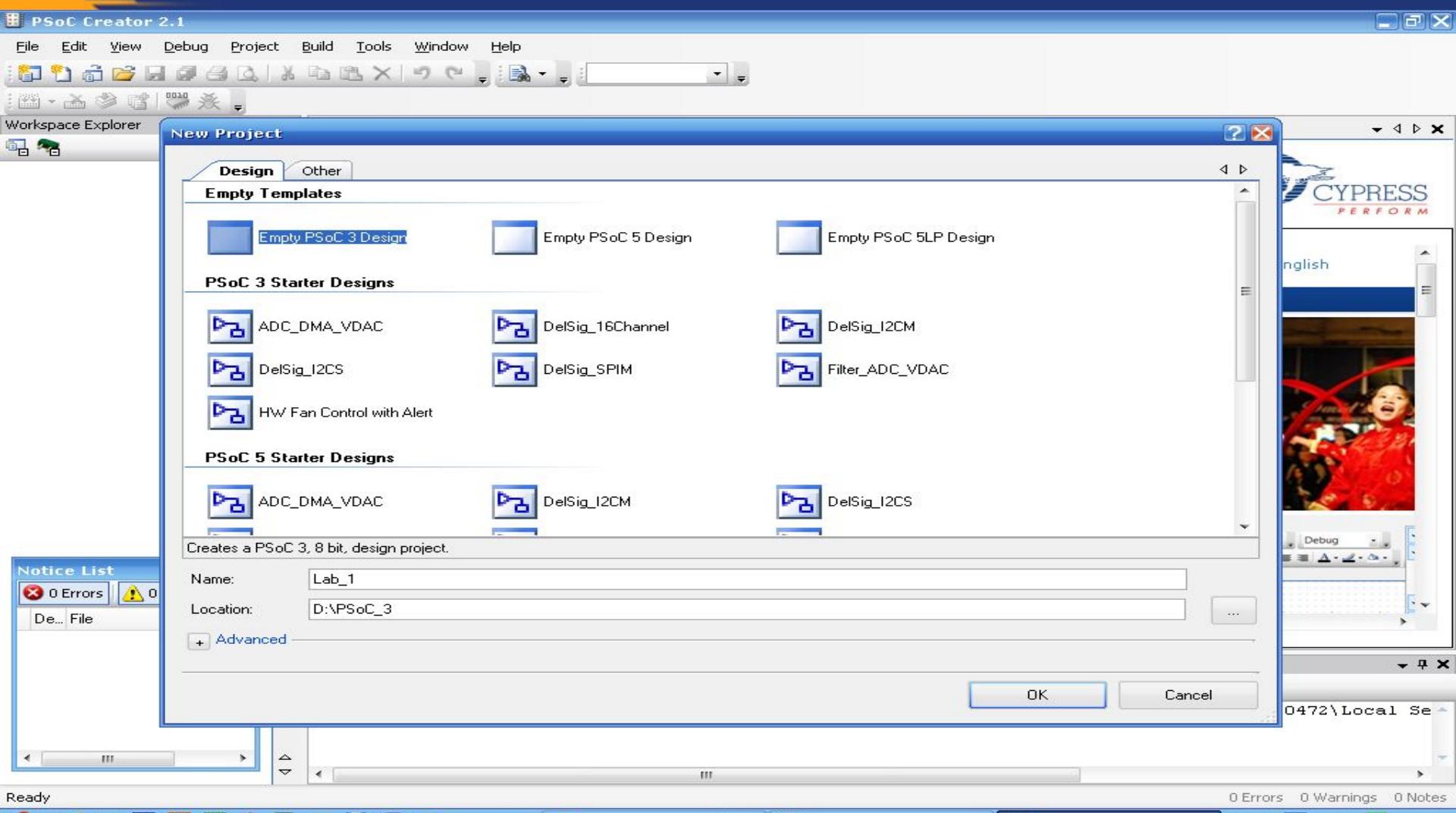
**Компонента
в PSoC Creator 4.2,
яка працює з Fan**
•Fan Controller

Ця компонента знаходитьться
в компонент каталозі,
в категорії
“Thermal Management”.





Empty PSoC 3/5 Design





CYPRESS

Lab_7 Fan

CHAPTER #LOW



Lab_7 Fan

CharLCD_CustomFont01 - PSoC Creator 2.1 [F:\...\CharLCD_CustomFont01.cydsn\TopDesign\TopDesign.cysch]

File Edit View Debug Project Build Tools Window Help

Microsoft Sans Serif 10 B I U

Workspace Explorer (1 project)

Start Page TopDesign.cysch CharLCD_Cust...Font01.cydwr main.c

Component Catalog (174 co...)

Project 'CharLCD_CustomFi' **Source** Components Datasheets Results

Notice List **0 Errors** **0 Warnings**

Output Show output from: All Log file for this session is located at: C:\Documents and Settings\Admin

Component Catalog Concept Cypress

- Cypress Component Catalog
 - Analog
 - ADC
 - Amplifiers
 - Analog MUX
 - Comparator [v1.90]
 - DAC
 - Manual Routing
 - Mixer [v1.91]
 - Sample/Track and Hold
 - VRef [v1.60]
 - CapSense
 - Communications
 - Digital
 - Display
 - Character LCD [v1.70]
 - Graphic LCD 8-bit Parallel
 - Graphic LCD 16-bit Parallel
 - Graphic LCD Controller I
 - Graphic LCD Controller II
 - Graphic LCD Parallel Interface
 - ResistiveTouch [v1.10]

Component Preview

Datasheet

0 Errors 0 Warnings 0 Notes



Configure LCD

Screenshot of PSoC Creator 2.1 software interface showing the configuration of a Character LCD.

The main window displays the "Configure 'CharLCD'" dialog box:

- Name:** LCD_Char
- General Tab:** Selected. Parameters include:
 - LCD Custom Character Set:
 - None
 - Vertical Bargraph
 - Horizontal Bargraph
 - User Defined
 - Include ASCII to Number Conversion Routines
- Buttons:** Datasheet, OK, Apply, Cancel

The "Component Catalog" sidebar shows the "Cypress Component Catalog" expanded, with the "Display" category selected, listing various LCD components.

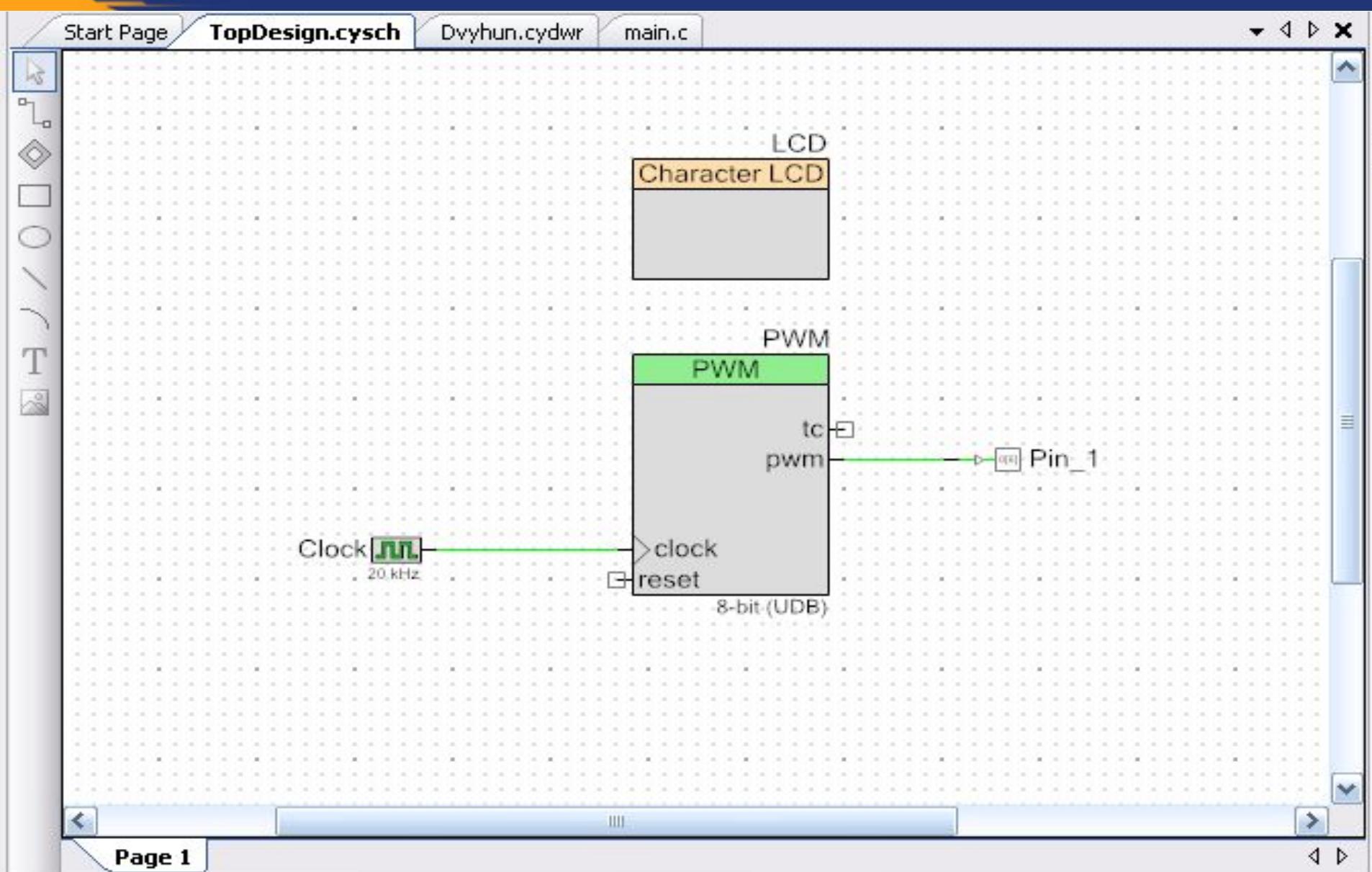
The "Notice List" panel indicates 0 Errors and 0 Warnings.

The status bar at the bottom right shows coordinates (X=295, Y=131) and system status (0 Errors, 0 Warnings, 0 Notes).



CYPRESS

Lab_7 Fan





Lab_7 Fan

Configure 'PWM'

Name: PWM

Configure Advanced Built-in

Implementation: Fixed Function UDB

Resolution: 8-Bit 16-Bit

PWM Mode: One Output

Period: 255 Max **Period = 12.8ms**

CMP Value 1: 75

Datasheet OK Apply Cancel

20



CYPRESS

Lab_7 Fan

Configure 'PWM'



Name: PWM

[Configure](#)**Advanced**[Built-in](#)

Enable Mode: Software Only

Run Mode: Continuous

Trigger Mode: None

Kill Mode: Disabled 1

Capture Mode: None

Interrupts:

 None

- Interrupt On Terminal Count Event
- Interrupt On Compare 1 Event
- Interrupt On Compare 2 Event
- Interrupt On Kill Event

[Datasheet](#)**OK**[Apply](#)[Cancel](#)

Lab_7 Fan

Configure 'cy_clock'



Name:

Basic Advanced Built-in



Clock type: New Existing

Source: <Auto>

Specify: Frequency: kHz
 Tolerance: - +

Summary

API Generated: Yes

Uses Clock Tree Resource: Yes

By default, all clocks are marked as 'start on reset'. The setting can be changed in the Design Wide Resources editor.

Datasheet

OK

Apply

Cancel



CYPRESS

Lab_7 Fan

Configure 'cy_pins'

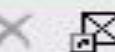


Name:

Pin_1

 Pins Mapping Reset Built-in

Number of Pins: 1



[All Pins]

 Pin_1_0

Type

General

Input

Output

 Analog Digital Input HW Connection Digital Output HW Connection Output Enable Bidirectional Show External Terminal

Preview:

 Datasheet OK Apply Cancel



CYPRESS

Lab_7 Fan

Configure 'cy_pins'

Name: Pin_1

Pins Mapping Reset Built-in

Number of Pins: 1

[All Pins] Pin_1_0

Type General Input Output

Drive Mode: Strong Drive

Initial State: Low (0)

Minimum Supply Voltage:

Datasheet OK Apply Cancel

The diagram illustrates the internal logic of a pin configuration. It shows a strong drive output stage connected to a pin labeled 'Pin'. The output path includes a pull-up resistor and a driver. A feedback path from the output is controlled by a driver with inputs 'DR' and 'PS'. The 'DR' input has a buffer symbol, indicating it is an inverted control signal.



CYPRESS

Lab_7 Fan

Configure 'cy_pins'



Name: Pin_1

Pins

Mapping

Reset

Built-in



Number of Pins: 1



[All Pins]

 Pin_1_0**Type****General**

Input

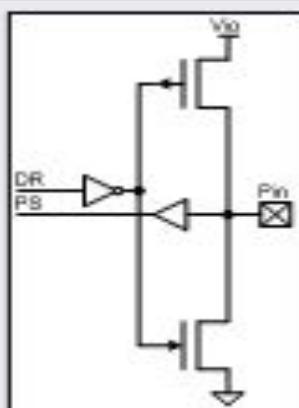
Output

Drive Mode

Strong Drive

Initial State:

High (1)



Minimum Supply Voltage:



Datasheet

OK

Apply

Cancel

Lab_7 Fan

Configure 'cy_pins'



Name: Pin_1

Pins Mapping Reset Built-in

Number of Pins: 1



[All Pins]

Pin_1_0

Type	General	Input	Output
			Slew Rate:
			Fast
			Drive Level:
			Vddio
			Current:
			4mA source, 8mA sink
			Output Mode:
			Transparent

Datasheet

OK

Apply

Cancel

Lab_7 Fan

Configure 'cy_pins'

Name: Pin_1

Pins Mapping Reset Built-in

Number of Pins: 1

[All Pins] Pin_1_0

Type	General	Input	Output
Slew Rate:	Fast		
Drive Level:	Vddio		
Current:	4mA source, 25mA sink		
Output Mode:	Transparent		

Datasheet OK Apply Cancel

Lab_7 Fan

Start Page *TopDesign.cysch Dvyhun.cydwr main.c

Alias	Name	Port	Pin	Lock
	\LCD:LCDPort[6:0]\	P2[6:0]	95..99,1..2	<input checked="" type="checkbox"/>
	Pin_1	P0[6] IDAC:HI	78	<input checked="" type="checkbox"/>

CY8C3866AXI-040 100-TQFP

Pins **Analog** **Clocks** **Interrupts** **DMA** **System** **Directives** **Flash Security**



Lab_7 Fan

Start Page *TopDesign.cysch Dvyhun.cydwr main.c

Add Design-Wide Clock... Delete Design-Wide Clock Edit Clock...

Type	Name	Domain	Desired Frequency	Nominal Frequency	Accuracy (%)	Tolerance (%)	Divider	Start on Reset	Source Clock
System	USB_CLK	DIGITAL	48.000 MHz	? MHz	±0	-	1	<input type="checkbox"/>	IMOx2
System	Digital Signal	DIGITAL	? MHz	? MHz	±0	-	0	<input type="checkbox"/>	
System	XTAL 32kHz	DIGITAL	32.768 kHz	? MHz	±0	-	0	<input type="checkbox"/>	
System	XTAL	DIGITAL	25.000 MHz	? MHz	±0	-	0	<input type="checkbox"/>	
System	ILO	DIGITAL	? MHz	1.000 kHz	-50, +100	-	0	<input checked="" type="checkbox"/>	
System	IMO	DIGITAL	3.000 MHz	3.000 MHz	±1	-	0	<input checked="" type="checkbox"/>	
System	BUS_CLK (CPU)	DIGITAL	? MHz	24.000 MHz	±1	-	1	<input checked="" type="checkbox"/>	MASTER_CLK
System	MASTER_CLK	DIGITAL	? MHz	24.000 MHz	±1	-	1	<input checked="" type="checkbox"/>	PLL_OUT
System	PLL_OUT	DIGITAL	24.000 MHz	24.000 MHz	±1	-	0	<input checked="" type="checkbox"/>	IMO
Local	Clock	DIGITAL	20.000 kHz	20.000 kHz	±1	±5	150	<input checked="" type="checkbox"/>	Auto: IMO

Pins Analog Clocks Interrupts DMA System Directives Flash Security



Lab_7 Fan

Start Page *TopDesign.cysch Dvyhun.cydwr main.c

Reset Expand Collapse

Option	Value
Configuration	<ul style="list-style-type: none">Device Configuration Mode: Compressed <input checked="" type="checkbox"/>Enable Error Correcting Code (ECC): <input type="checkbox"/>Store Configuration Data in ECC Memory: <input checked="" type="checkbox"/>Instruction Cache Enabled: <input checked="" type="checkbox"/>Enable Fast IMO During Startup: <input checked="" type="checkbox"/>Clear SRAM During Startup: <input checked="" type="checkbox"/>Unused Bonded IO: Allow but warn <input type="checkbox"/>
Programming\Debugging	<ul style="list-style-type: none">Debug Select: SWD+SWV (serial wire debug and viewer) <input type="checkbox"/>Enable Device Protection: <input checked="" type="checkbox"/>Require XRES Pin: <input checked="" type="checkbox"/>Use Optional XRES: <input type="checkbox"/>
Operating Conditions	

Pins Analog Clocks Interrupts DMA System Directives Flash Security

Notice List

0 Errors 0 Warnings 0 Notes Go To Error View Details

Start Page *TopDesign.cysch Dvyhun.cydwr *main.c

```
1 /* =====
2  * Copyright YOUR COMPANY, THE YEAR
3  * All Rights Reserved
4  * UNPUBLISHED, LICENSED SOFTWARE.
5  * CONFIDENTIAL AND PROPRIETARY INFORMATION
6  * WHICH IS THE PROPERTY OF your company.
7  * =====
8 */
9 #include <device.h>
10
11 void main()
12 {
13     /* Place your initialization/startup code here (e.g. MyInst_Start()) */
14     /* Enable the global interrupts */
15     CYGlobalIntEnable;
16     LCD_Start();
17     LCD_Position(0,9);
18     LCD_PrintString("Dvyhun");
19     /* CyGlobalIntEnable; */ /* Uncomment this line to enable global interrupts. */
20     PWM_Start();
21     for(;;)
22     {
23         /* Place your application code here. */
24     }
25 }
26
27 /* [] END OF FILE */
```

Notice List

0 Errors

0 Warnings

0 Notes

Go To Error

View Details



CYPRESS

Lab_7 Fan

Start Page

*TopDesign.cysch

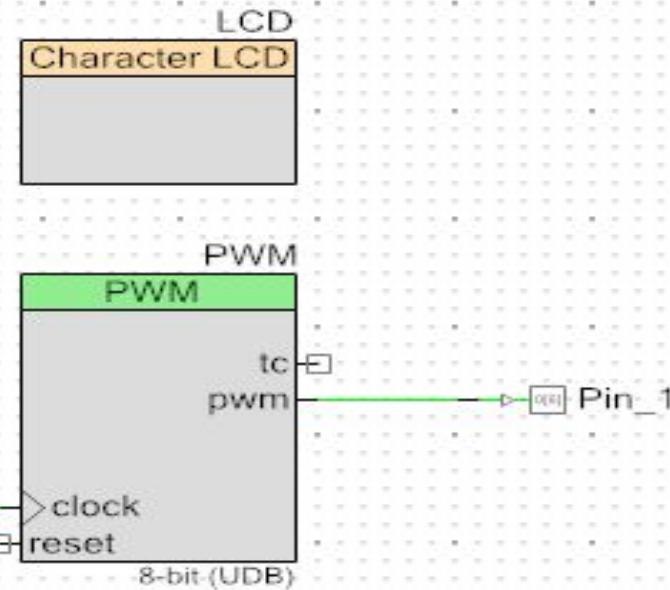
Dvyhun.cydwr

*main.c

▼ ▲ ×

**Вставити інтерфейс
користувача для
управління
швидкістю обертання
двигуна
постійного струму**

Clock [Hz]
20.kHz



Page 1

◀ ▶

Notice List

0 Errors

0 Warnings

0 Notes

Go To Error

View Details

▼ ▲ ×



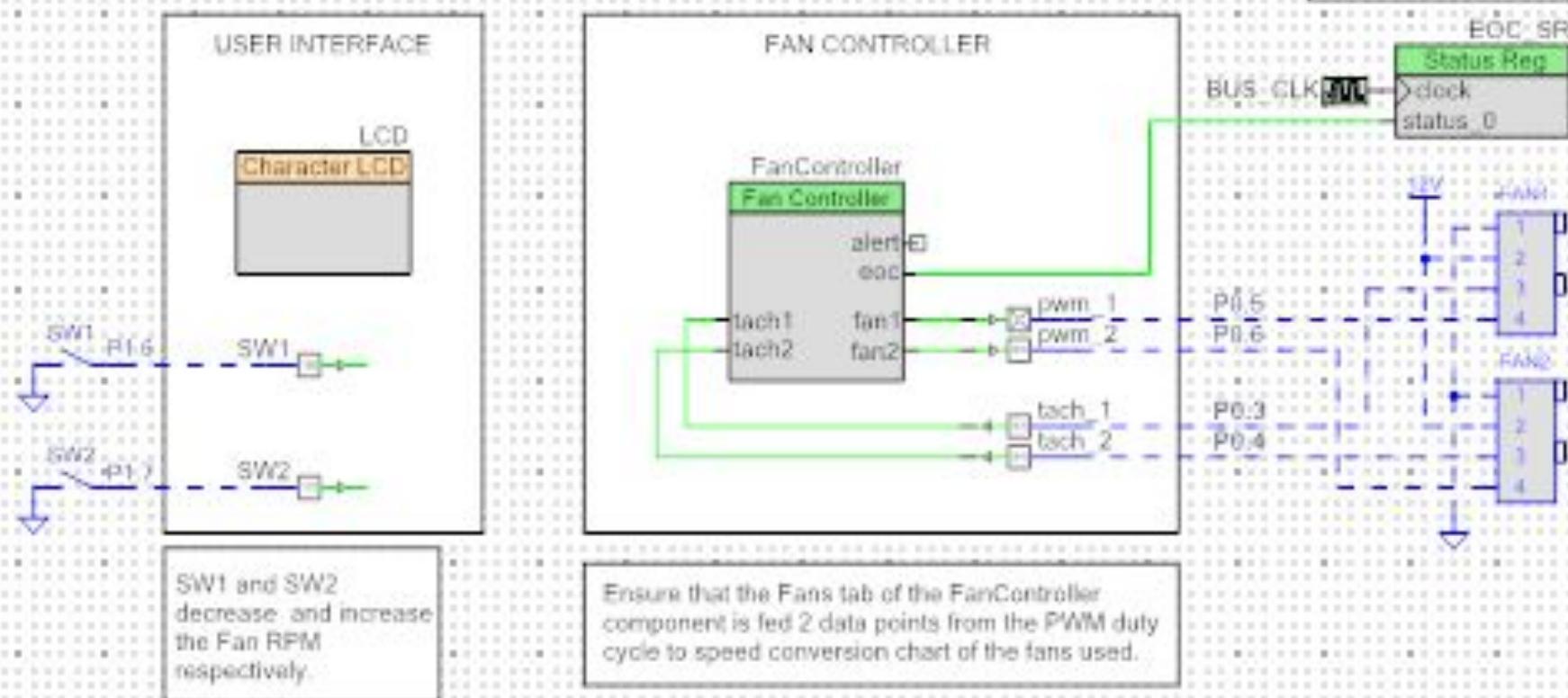
Lab_7 Fan

CHAPTER #HI

FanController Example Project - Firmware Fan Control

Fan Controller is configured into firmware (CPU) fan control mode
Speed control algorithm is implemented in firmware
Firmware synchronizes to hardware using end-of-cycle (eoc) pulse
2 Fans are supported (individual PWMs - no banks)

Sticky status register indicates when tachometer finishes reading speeds of all fans



Lab_7



Cypress **Off-Chip**

- |- Cypress Component Catalog
- |- Analog
 - + ADC
 - + Amplifiers
 - + Analog MUX
 - + Comparator [v2.0]
 - + DAC
 - + Manual Routing
 - + Mixer [v2.0]
 - + Sample/Track and Hold
 - + VRef [v1.60]
- |- CapSense
 - + CapSense_CSD [v3.30]
- |- Communications
- |- Digital
 - + Functions
 - + Logic
 - + Registers
 - + Control Register [v1.0]
 - + Status Register [v1.0]
 - + Utility

Component Preview



CYPRESS

Configuration Fan Controller

Configure 'FanController'

Name: **Basic** 

Fan Control Method

 Firmware (CPU) Hardware (UDB)Damping factor (sec): Spinners for adjusting the damping factor.Tolerance: Spinners for adjusting the tolerance. Acoustic noise reduction

Alerts

 Fan stall / Rotor lock UDB speed regulation failure

Connections

 Display as bus External clock



CYPRESS

Configuration Fan Controller

Configure 'FanController'



Name: FanController

[Basic](#) **Fans** [Built-in](#)

Motor support

 4-pole motors 6-pole motors

Pwm output configuration

Number of fans: 2

Number of banks: 0

Pwm resolution: 10 bit

Pwm frequency: 25 kHz

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	35	4500	70	10500	4500
2	35	4500	70	10500	4500

[Datasheet](#)[OK](#)[Apply](#)[Cancel](#)

Lab_7 .cywrr

Start Page TopDesign.cysch FW Fan Control01.cydwr

▼ ◀ ▶ ×

Alias	Name	Port	Pin
\LCD:LCDPort[6:0]\ P2[6:0]			95..99,1
pwm_1	P0[5] OpAmp-		77
pwm_2	P0[6] IDAC:HI		78
SW1	P1[6]		27
SW2	P1[7]		28
tach_1	P0[3] OpAmp-, DSM: ExtVref		74
tach_2	P0[4] OpAmp+		76



CY8C3866AXI-040
100-TQFP

The diagram shows the pin layout of the CY8C3866AXI-040 microcontroller. It has 100 pins arranged in a square package. Pins are color-coded: blue for VDD, red for GND, and various colors for digital and analog I/O. Key pins labeled include VDD, GND, SCK, CS, MISO, MOSI, SDA, SCL, INT, and several PWM and DAC pins.

 Pins

 Analog

 Clocks

 Interrupts

 DMA

 System

 Directives

 Flash Security

◀ ▶

Lab_7

Start Page TopDesign.cysch FW Fan Control01.cydwr main.c

Add Design-Wide Clock... Delete Design-Wide Clock Edit Clock...

Type	Name	Domain	Desired Frequency	Nominal Frequency	Accuracy (%)	Tolerance (%)	Divider	Start on Reset	Source Clock
System	USB_CLK	DIGITAL	48.000 MHz	? MHz	±0	-	1	<input type="checkbox"/>	IMOx2
System	Digital Signal	DIGITAL	? MHz	? MHz	±0	-	0	<input type="checkbox"/>	
System	XTAL 32kHz	DIGITAL	32.768 kHz	? MHz	±0	-	0	<input type="checkbox"/>	
System	XTAL	DIGITAL	25.000 MHz	? MHz	±0	-	0	<input type="checkbox"/>	
System	ILO	DIGITAL	? MHz	1.000 kHz	-50, +100	-	0	<input checked="" type="checkbox"/>	
System	IMO	DIGITAL	3.000 MHz	3.000 MHz	±1	-	0	<input checked="" type="checkbox"/>	
System	BUS_CLK (CPU)	DIGITAL	? MHz	24.000 MHz	±1	-	1	<input checked="" type="checkbox"/>	MASTER_CLK
System	MASTER_CLK	DIGITAL	? MHz	24.000 MHz	±1	-	1	<input checked="" type="checkbox"/>	PLL_OUT
System	PLL_OUT	DIGITAL	24.000 MHz	24.000 MHz	±1	-	0	<input checked="" type="checkbox"/>	IMO
Local	FanController_TACH_CLK_500K	DIGITAL	500.000 kHz	500.000 kHz	±1	±5	6	<input checked="" type="checkbox"/>	Auto: IMO
Local	FanController_PWM_CLOCK_25k_10b	DIGITAL	24.000 MHz	24.000 MHz	±1	-	1	<input checked="" type="checkbox"/>	MASTER_CLK
Local	FanController_BUS_CLK	DIGITAL	? MHz	24.000 MHz	±1	-	0	<input checked="" type="checkbox"/>	BUS_CLK
Local	Clock_1	DIGITAL	? MHz	24.000 MHz	±1	-	0	<input checked="" type="checkbox"/>	BUS_CLK

Pins Analog Clocks Interrupts DMA System Directives Flash Security



CYPRESS

Lab_7

Start Page TopDesign.cysch FW Fan Control01.cydwr main.c

Reset Expand Collapse

Option	Value
Configuration	
Device Configuration Mode	DMA
Enable Error Correcting Code (ECC)	<input type="checkbox"/>
Store Configuration Data in ECC Memory	<input checked="" type="checkbox"/>
Instruction Cache Enabled	<input checked="" type="checkbox"/>
Enable Fast IMO During Startup	<input checked="" type="checkbox"/>
Clear SRAM During Startup	<input checked="" type="checkbox"/>
Unused Bonded IO	Allow but warn
Programming\Debugging	
Debug Select	SWD+SWV (serial wire debug and viewer)
Enable Device Protection	<input type="checkbox"/>
Require XRES Pin	<input checked="" type="checkbox"/>
Use Optional XRES	<input type="checkbox"/>
Operating Conditions	
Vddd (V)	5.0
Vdda (V)	5.0
Variable Vdda	<input type="checkbox"/>
Vddio0 (V)	5.0
Vddio1 (V)	5.0

Pins Analog Clocks Interrupts DMA System Directives Flash Security



CYPRESS

Main.c

Lab_1 - PSoC Creator 2.1 [D:\PSoC_3\Lab_1\Lab_1.cydsn\main.c]

File Edit View Debug Project Build Tools Window Help

Debug

Workspace Explorer (1 project) X

Project 'Lab_1' [CY8C3866]

- TopDesign.cysch
- Lab_1.cydwr
- Header Files
 - device.h
- Source Files
 - main.c

Source Components Datasheets Results

Start Page *TopDesign.cysch main.c

```
1 /* ===== */
2 *
3 * Copyright YOUR COMPANY, THE YEAR
4 * All Rights Reserved
5 * UNPUBLISHED, LICENSED SOFTWARE.
6 *
7 * CONFIDENTIAL AND PROPRIETARY INFORMATION
8 * WHICH IS THE PROPERTY OF your company.
9 *
10 * =====
11 */
12 #include <device.h>
13
14 void main()
15 {
16     /* Place your initialization/startup code here (e.g. MyInst_Start()) */
17
18     /* CyGlobalIntEnable; */ /* Uncomment this line to enable global interrupts. */
19     for(;;)
20     {
21         /* Place your application code here. */
22     }
23 }
24
25 /* [] END OF FILE */
```

Notice List X

0 Errors 0 Warnings

De... File Error L

Output

Show output from: All X

Log file for this session is located at: C:\Documents and Settings\Admin.MICROSOFT\Local Se

Ready

Ln 1 Col 1 INS 0 Errors 0 Warnings 0 Notes

Lab_7

```
1  ****
2  * Project Name:      Firmware Fan Control
3  * Version:          1.10
4  *
5  * Device Used:      PSoC 3 CY8C3866AXI-040 or PSoC 5 CY8C5588AXI-060ES1
6  * Software Used:    PSoC Creator v2.1
7  * Compiler Used:   Keil(C51), ARM GNU CC
8  * Related Hardware: CY8CKIT-001 PSoC DVK
9  ****
10 * Theory of Operation:
11 *
12 *   Fan Controller is configured into firmware (CPU) fan control mode
13 *   Speed control algorithm is implemented in firmware
14 *   Firmware synchronizes to hardware using end-of-cycle (eoc) pulse
15 *   2 Fans are supported (individual PWMs - no banks)
16 *
17 *   1st line of LCD displays Desired Speed and Actual Speed and PWM Duty Cycle of Fan 1
18 *   2nd line of LCD displays Actual Speed and PWM Duty Cycle of Fan 2
19 *   SW1 decreases desired speed in RPM
20 *   SW2 increases desired speed in RPM
21 *
22 ****
23 * Copyright 2012, Cypress Semiconductor Corporation. All rights reserved.
24 * This software is owned by Cypress Semiconductor Corporation and is protected
25 * by and subject to worldwide patent and copyright laws and treaties.
26 * Therefore, you may use this software only as provided in the license agreement
27 * accompanying the software package from which you obtained this software.
28 * CYPRESS AND ITS SUPPLIERS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED,
29 * WITH REGARD TO THIS SOFTWARE, INCLUDING, BUT NOT LIMITED TO, NONINFRINGEMENT,
30 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
31 *****/
```

Lab_7

```
31  ****
32
33 #include <device.h>
34 #include <stdio.h>
35
36 /* PWM duty cycle controls - units are hundredths of a percent */
37 #define MIN_DUTY          (50u)
38 #define MAX_DUTY          (9950u)
39 #define DUTY_STEP_COARSE   (100u)
40 #define DUTY_STEP_FINE     (2u)
41
42 /* Speed controls - units are RPM */
43 #define MIN_RPM           (2500u)
44 #define MAX_RPM           (9500u)
45 #define INIT_RPM          (4500u)
46 #define RPM_STEP          (500u)
47 #define RPM_DELTA_LARGE   (500u)
48 #define RPM_TOLERANCE     (100u)
49
50 void main()
51 {
52     uint16  desiredSpeed = INIT_RPM;
53     uint16  dutyCycle[2];
54     uint8   fanNumber;
55     char    displayString[6];
56
57     FanController_Start();
58     FanController_SetDesiredSpeed(1u, desiredSpeed);
59     FanController_SetDesiredSpeed(2u, desiredSpeed);
60     dutyCycle[0] = FanController_GetDutyCycle(1u);
61     dutyCycle[1] = FanController_GetDutyCycle(2u);
62 }
```



```
61    dutyCycle[1] = FanController_GetDutyCycle(2u);
62    LCD_Start();
63    LCD_Position(0u, 0u);
64    LCD_PrintDecUInt16(desiredSpeed);
65    LCD_Position(1u, 0u);
66    LCD_PrintString("F/W");
67    while(1u)
68    {
69        /* Synchronize firmware to end-of-cycle pulse from FanController */
70        if(EOC_SR_Read())
71        {
72            for(fanNumber = 1u; fanNumber <= 2u; fanNumber++)
73            {
74                /* Display Fan Actual Speeds */
75                LCD_Position(fanNumber-1,5u);
76                LCD_PrintDecUInt16(FanController_GetActualSpeed(fanNumber));
77                LCD_PrintString("   ");
78
79                /* Firmware Speed Regulation */
80                LCD_Position(fanNumber-1,9u);
81
82                /* Fan Below Desired Speed */
83                if(FanController_GetActualSpeed(fanNumber) < desiredSpeed)
84                {
85                    if((desiredSpeed - FanController_GetActualSpeed(fanNumber)) > RPM_DELTA_LI
86                    {
87                        dutyCycle[fanNumber-1] += DUTY_STEP_COARSE;
88                    }
89                    else
90                    {
91                        dutyCycle[fanNumber-1] += DUTY_STEP_FINE;
92                    }
93                }
94            }
95        }
96    }
97 }
```



```
89         else
90     {
91         dutyCycle[fanNumber-1] += DUTY_STEP_FINE;
92     }
93     if(dutyCycle[fanNumber-1] > MAX_DUTY)
94     {
95         dutyCycle[fanNumber-1] = MAX_DUTY;
96     }
97 }
98 /* Fan Above Desired Speed */
99 else if(FanController_GetActualSpeed(fanNumber) > desiredSpeed)
100 {
101     if((FanController_GetActualSpeed(fanNumber) - desiredSpeed) > RPM_DELTA_LI
102     {
103         if(dutyCycle[fanNumber-1] > (MIN_DUTY+DUTY_STEP_COARSE))
104         {
105             dutyCycle[fanNumber-1] -= DUTY_STEP_COARSE;
106         }
107     }
108     else if((FanController_GetActualSpeed(fanNumber) - desiredSpeed) > RPM_TO
109     {
110         if(dutyCycle[fanNumber-1] > MIN_DUTY)
111         {
112             dutyCycle[fanNumber-1] -= DUTY_STEP_FINE;
113         }
114     }
115 }
116 FanController_SetDutyCycle(fanNumber, dutyCycle[fanNumber-1]);
117
118 /* Display Current Duty Cycle Settings (in 100ths of a percent) */
119 LCD_Position(fanNumber-1,10u);
```



Lab_7

```
120         sprintf(displayString, "%5.2f", (((float)dutyCycle[fanNumber-1])/100));
121         LCD_PrintString(displayString);
122         LCD_PrintString("%      ");
123     }
124     CyDelay(250u);

125
126     /* Check for Button Press to Change Speed */
127     if((!SW1_Read()) || (!SW2_Read()))
128     {
129         /* Decrease Speed */
130         if(!SW1_Read())
131         {
132             if(desiredSpeed > MIN_RPM)
133             {
134                 desiredSpeed -= RPM_STEP;
135             }
136         }
137
138         /* Increase Speed */
139     else
140     {
141         desiredSpeed += RPM_STEP;
142         if(desiredSpeed > MAX_RPM)
143         {
144             desiredSpeed = MAX_RPM;
145         }
146     }
147
148     /* Display Updated Desired Speed */
149     LCD_Position(0u, 0u);
150     LCD_PrintDecUint16(desiredSpeed);
```



CYPRESS

Lab_7

```
135             )
136         )
137
138         /* Increase Speed */
139     else
140     {
141         desiredSpeed += RPM_STEP;
142         if(desiredSpeed > MAX_RPM)
143         {
144             desiredSpeed = MAX_RPM;
145         }
146     }
147
148     /* Display Updated Desired Speed */
149     LCD_Position(0u, 0u);
150     LCD_PrintDecUint16(desiredSpeed);
151     FanController_SetDesiredSpeed(1u, desiredSpeed);
152     FanController_SetDesiredSpeed(2u, desiredSpeed);
153     dutyCycle[0] = FanController_GetDutyCycle(1u);
154     dutyCycle[1] = FanController_GetDutyCycle(2u);
155
156     /* Switch Debounce */
157     CyDelay(250u);
158
159     }
160 }
161 }
162
163
164 /* [] END OF FILE */
165
```



CYPRESS

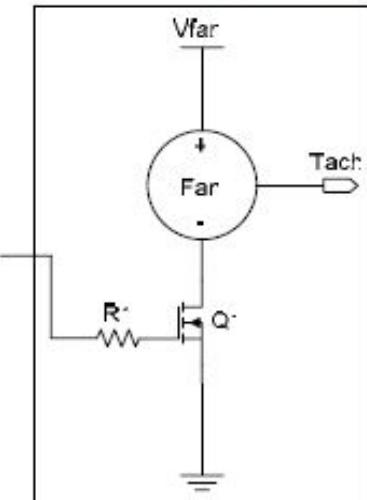
Lab_7

Fan Module

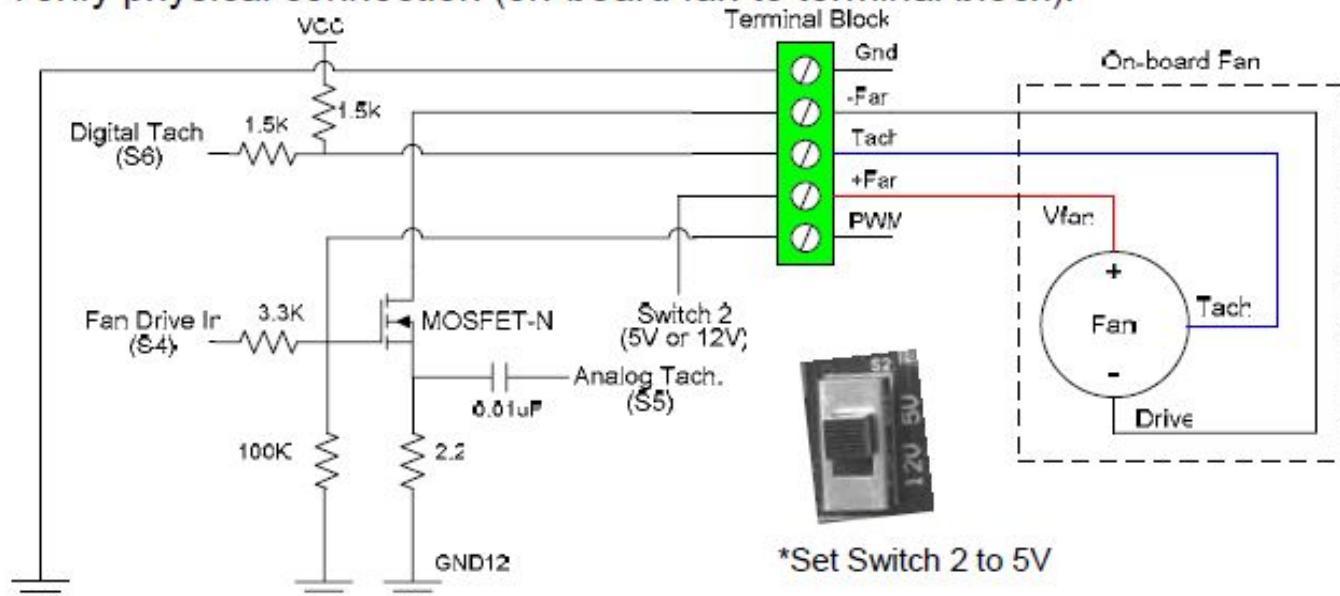
CY3210-ExpressDK and CY3210-FanMod

Using the on-board 3-wire, speed controllable, 5V external FET driven fan

- 1) Select appropriate PSoC Express output device:
Fan >> Brushless DC >> Speed Control >> External Drive FET >> 5V 2-/3-Wire.



- 2) Verify physical connection (on-board fan to terminal block).





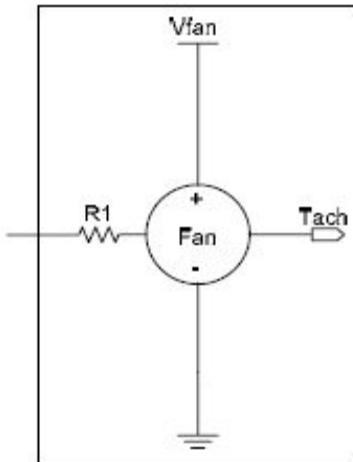
CYPRESS

Lab_7

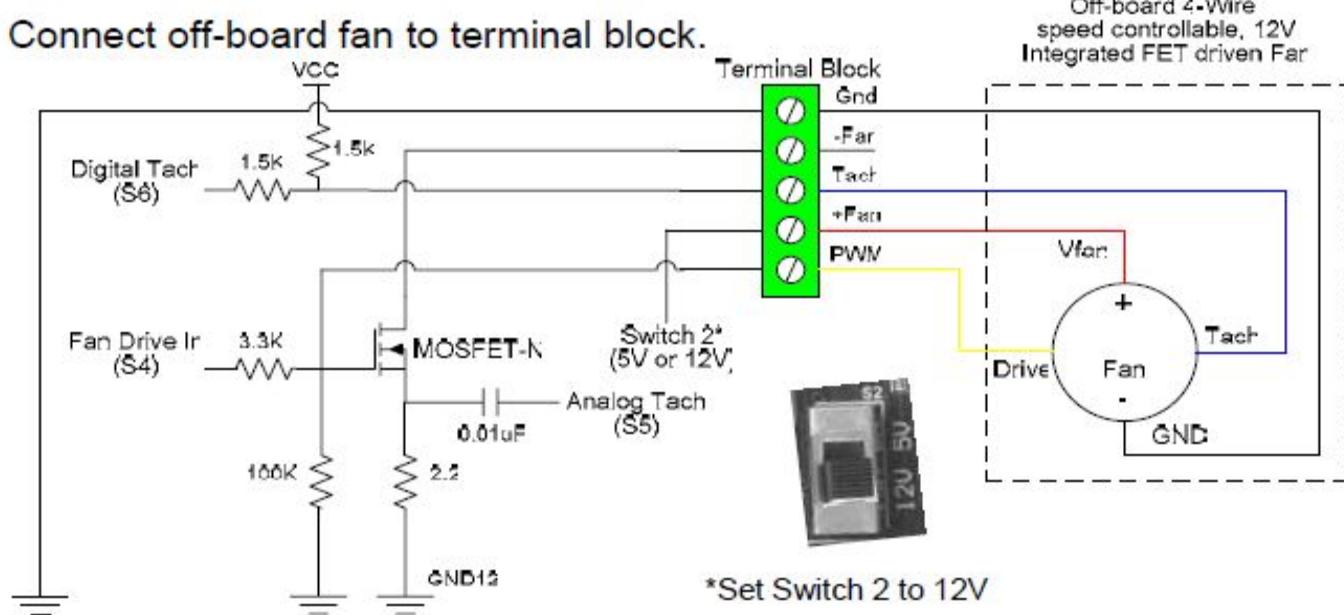
Attaching an off-board 4-wire, speed controllable, 12V integrated FET driven fan

- 1) Select appropriate PSoC Express output device::

Fan >> Brushless DC >> Speed Control >> Integrated Drive >> 12V 4-Wire.



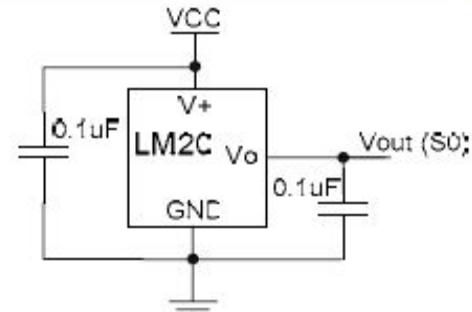
- 2) Connect off-board fan to terminal block.



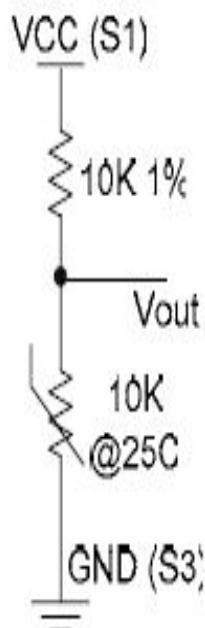
Using the LM20



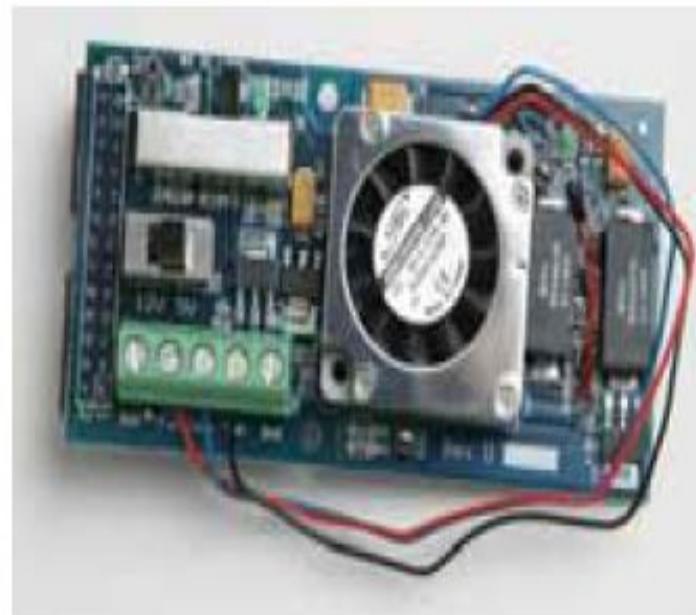
- 1) Select appropriate PSoC Express input device:
Temperature >> Integrated Circuits >> LM20 -55 to +130C.
- 2) Set Switch 1 to LM20. (LM20 Vout available on S0).



Using the Thermistor



- 1) Select appropriate PSoC Express input device:
Temperature >> Thermistor Temp Sensor.
- 2) Set Switch 1 to LM20
 - Vcc available on S1
 - VTherm available on S2
 - GND available on S3



Diode Sensor Topologies

The Diode Sensor will be supported in future releases of PSoC Express.



CYPRESS

Lab_7



Рекомендована література :

Основна:

1. AN66627 PSoC® 3 and PSoC 5LP Intelligent Fan Controller . 2011 . - 22 с.
2. PSoC® 3 Architecture TRM (Technical Reference Manual) – 379 с.

Додаткова:

1. PSoC® Creator™ Component Datasheet. Fan Controller – 6 с.

**На сайті фірми
Cypress знаходиться
більше 200
Application Notes і
Reference Designs,
які ілюструють
області
застосування
мікроконтролерів
PSoC.**

Design Support - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address: http://www.cypress.com/portal/server.pt?space=Community/Page&control=SetCommunity&Community

Cypress Home Products Applications Design Support Design Resources Qual. & Reliability Discussion Boards Technical Training CYPRESS Consultants Customer Design University Program Technical Support About Cypress Investor Relations Press Careers

Design Resources Select Product Group: All Product Groups Application Specific Clocks Async SRAMs Analog Products Backplane Interface & Clock Mgmt Bluetooth Solutions Apply Filter

Select Product Family: All Product Families Application Specific Clocks Async SRAMs Analog Products Backplane Interface & Clock Mgmt Bluetooth Solutions

Application Notes Datasheets Developer Kits Errata Update Evaluation Boards Models More Resources Reference Designs Software and Drivers Technical Articles

Product Family	Descriptive Name	Date	Downloads
PSoC Mixed-Signal Array	AN2267a - Standard - Single Cell Li-Ion Battery Charger using CY8C21xxx	Apr 19, 2005	AN2267a.PDF AN2267a.ZIP
PSoC Mixed-Signal Array	AN2260 - Standard - Rapid NiCd/NiMH Battery Charger and DC Brushed Motor Controller for Autonomous Appliances	Apr 15, 2005	AN2260.PDF AN2260.ZIP
PSoC Mixed-Signal Array	AN2026b - Support - In-System Serial Programming Protocol CY8C24794 and CY8C29xxx	Apr 8, 2005	AN2026b.PDF
PSoC Mixed-Signal Array	AN2266 - Support - 16-bit PWM/PWM-DACs using One Digital PSoC(TM) Block	Apr 8, 2005	AN2266.PDF AN2266.ZIP
PSoC Mixed-Signal Array	AN2279 - Support - Dynamic I2C Addressing Implemented with I2C Hardware User Modules	Apr 8, 2005	AN2279.PDF AN2279.ZIP
PSoC Mixed-Signal Array	AN2267 - Standard - Single Cell Li-Ion Battery Charger	Apr 1, 2005	AN2267.PDF AN2267.ZIP
PSoC Mixed-Signal Array	AN2222a - Support - Flex-Pod Soldering Guide	Mar 31, 2005	AN2222a.PDF
PSoC Mixed-Signal Array	AN2233a - Support - Capacitive Switch Scan	Mar 31, 2005	AN2233a.PDF
PSoC Mixed-Signal Array	AN2276 - Support - Binary Weighted Single-Pole IIR Low-Pass Filters	Mar 29, 2005	AN2276.PDF AN2276.ZIP
PSoC Mixed-Signal Array	AN2277 - Support - Capacitive Front Panel Display Demonstration	Mar 29, 2005	AN2277.PDF AN2277.ZIP

Found 201 items 1 - 10 | 11 - 20 | 21 - 30 | 31 - 40 | 41 - 50 | 51 - 60 | 61 - 70 | 71 - 80 | 81 - 90 | 91 - 100 | see 1 - 100 | next 100

Contact Us | Privacy | Terms & Conditions | How to Buy | 日本語 | 中文
© Copyright 1995-2005, Cypress Semiconductor Corporation. All rights reserved.

PART Part SUPP Cu CY Tra Un File



CYPRESS

Мікропроцесорн а техніка 2019 р. (лекція 7, кінець) Благітко Б.Я.



CYPRESS

PERFORM