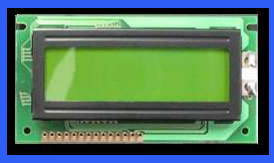
การทดลองที่ 2
การเชื่อมต่อไมโครคอนโทรลเลอร์ PSoC กับโมดูลแอลซีดีแบบดอทแมทริก (Interfacing the PSoC microcontroller with Dot Matrix LCD Module.)



หัวข้อ

1. วัตถุประสงค์
2. ความรู้เบื้องต้นเกี่ยวกับ LCD
   - โครงสร้างและลักษณะของ LCD
   - การทำงานของ LCD
3. การใช้งาน LCD กับ PSoC
   - ขั้นตอนการใช้งาน PSoC Designer เพื่อใช้ LCD
4. PSoC Designer LCD API

วัตถุประสงค์

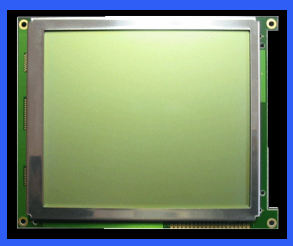
เพื่อให้สามารถใช้งานโมดูลต่างๆภายใน PSoC ได้

บทนำ

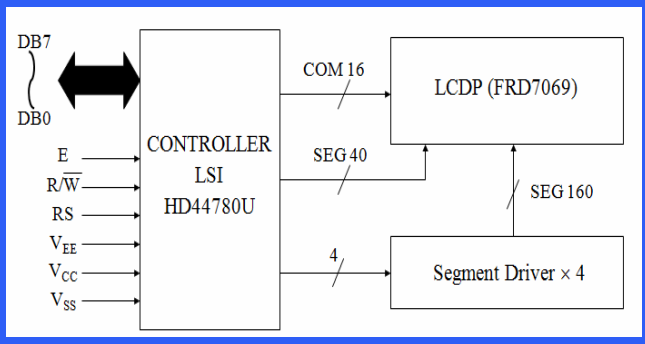
การทดลองนี้ ได้แสดงวิธีการใช้งานโมดูลแอลซีดีแบบดอทแมทริก เพื่อให้แสดงข้อความแบบต่างๆ

Dot Matrix LCD Module

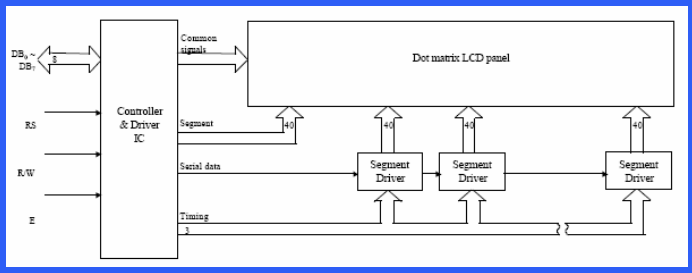
- Character LCD Module
- Graphic LCD Module
- Segment Display LCD Module



Block diagram ของ LCD Module

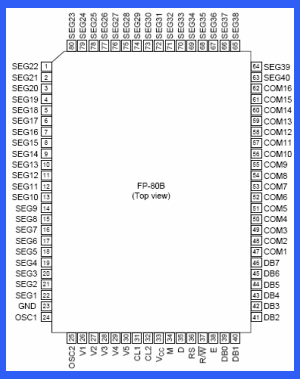


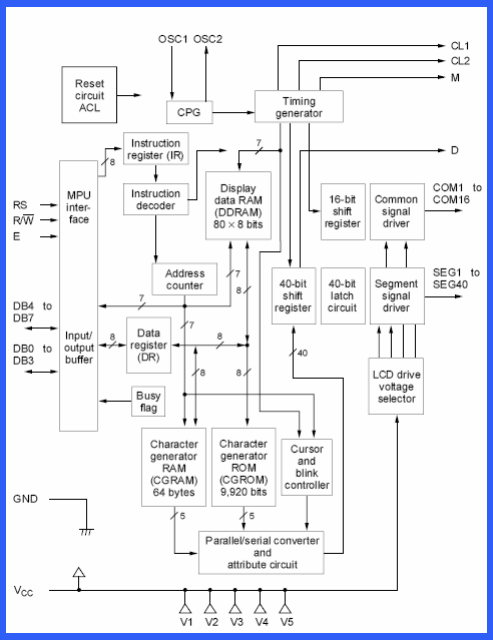
ส่วนประกอบของ LCD Module



1. Dot Matrix LCD เป็นตัวแสดงผล ทำงานในลักษณะของการปิดหรือเปิด ตัวเองกับแสง
2. Driver เป็นตัวขับ LCD รับสัญญาณมาจากส่วนควบคุม เบอร์ที่นิยมใช้ได้แก่ HD44100H และ MSM5259
3. Controller เป็นตัวรับข้อมูลจากอุปกรณ์ภายนอก แล้วควบคุมการทำงานของ LCD เบอร์ที่นิยมใช้ สำหรับแบบ Character ได้แก่ HD44780U ส่วนแบบ Graphic ได้แก่ HD61830

Block diagram of HD44780





คุณสมบัติพอสังเขป

- 5 x 8 and 5 x 10 dot matrix possible
- Low power operation support:
  - 2.7 to 5.5V
- Wide range of liquid crystal display driver power
  - 3.0 to 11V
- Liquid crystal drive waveform
  - A (One line frequency AC waveform)
- Correspond to high speed MPU bus interface
  - 2 MHz (when VCC = 5V)
- 4-bit or 8-bit MPU interface enabled
- 80 ´ 8-bit display RAM (80 characters max.)
- 9,920-bit character generator ROM for a total of 240 character fonts
  - 208 character fonts (5 ´ 8 dot)
  - 32 character fonts (5 ´ 10 dot)
- 64 x 8-bit character generator RAM
  - 8 character fonts (5 ´ 8 dot)
  - 4 character fonts (5 ´ 10 dot)
- 16-common ´ 40-segment liquid crystal display driver
- Programmable duty cycles
  - 1/8 for one line of 5 ´ 8 dots with cursor
  - 1/11 for one line of 5 ´ 10 dots with cursor
  - 1/16 for two lines of 5 ´ 8 dots with cursor
- Wide range of instruction functions:
  - Display clear, cursor home, display on/off, cursor on/off, display character blink, cursor shift, display shift
- Pin function compatibility with HD44780S
- Automatic reset circuit that initializes the controller/driver after power on
- Internal oscillator with external resistors
- Low power consumption

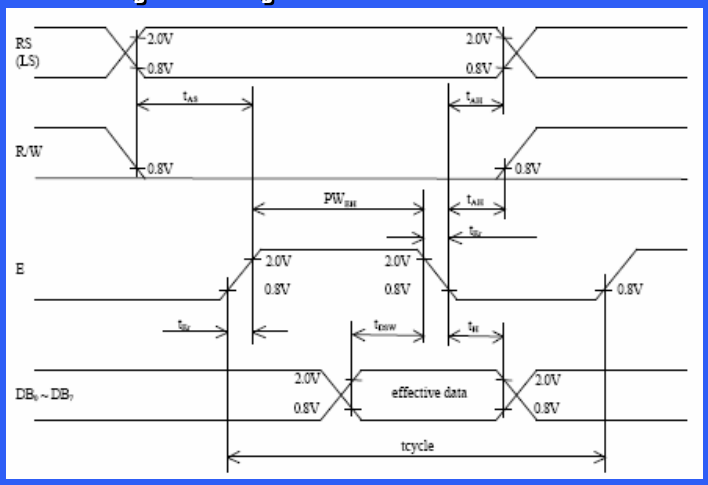
ขาของ LCD Module

ขาที่	สัญญาณ	รายละเอียด
1	Vss	0 V Gnd
2	Vcc	+ 5V
3	Vee	ใช้ปรับความสว่างของ LCD ถ้าต่อลงดินจะสว่างที่สุด
4	RS	สัญญาณ Register Select ใช้เลือกรีจิสเตอร์ควบคุม หรือหน่วยความจำแสดงผล - ถ้าเป็น "0" แสดงว่าต้องการติดต่อกับรีจิสเตอร์ควบคุม - ถ้าเป็น "1" แสดงว่าต้องการติดต่อกับริจิสเตอร์แสดงผล
5	R/W	สัญญาณควบคุมการอ่าน/เขียน ถ้าเป็น "0" " แสดงว่าต้องการเขียนหรือส่งข้อมูลให้แก่โมดูล ถ้าเป็น "1" แสดงว่าต้องการอ่านข้อมูลจากโมดูล

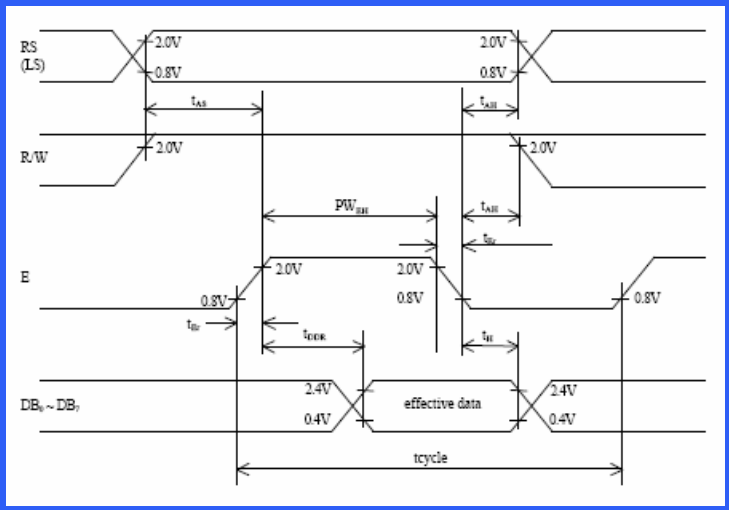
ขาของ LCD Module

ขาที่	สัญญาณ	รายละเอียด
6	E	Enable - สัญญานสั่งให้เริ่มต้นการทำงาน สำหรับการอ่าน/เขียนข้อมูล การรับส่งข้อมูลจะเกิดเมื่อเป็น '1' และขอบขาลง
7 ~ 0	DB0 ~ DB3	เป็นบัสแบบสองทิศทางใช้สำหรับส่งถ่ายข้อมูลระหว่างซีพียูกับโมดูลในกรณีทีการ ทำงานเป็นแบบ 4 บิต บัสนี้ไม่ได้ใช้และควรต่อลงดินด้วย แต่ถ้าเป็น การทำงานแบบ 8 บิต บัสนี้จะเป็น 4 บิตต่ำ ใช้เพื่อการส่งถ่ายข้อมูล
11~ 14	DB4 ~ DB7	เป็นบัสแบบสองทิศทางใช้สำหรับส่งถ่ายข้อมูลระหว่างซีพียูกับโมดูลในกรณีทีการทำงานเป็นแบบ 4 บิต จะใช้บัสนี้ส่งถ่ายข้อมูล แต่ถ้าเป็นการทำงานแบบ 8 บิต บัสนี้จะเป็น 4 บิตสูง นอกจากนี้ DB7 ยังใช้เป็นบิตแสดงสถานะBusy ด้วย

การเขียนข้อมูลให้โมดูล



การอ่านข้อมูลจากโมดูล



คำสั่งควบคุม

- DDRAM (Display Data Ram) คือหน่วยความจำภายในตัวโมดูลแอลซีดีที่เป็นบัฟเฟอร์ของข้อมูล ถ้าเขียนรหัส ASCII ใดๆ ลงในหน่วยความจำนี้ก็จะปรากฏเป็นตัวอักษรที่จอแสดงผลทันที
- CGRAM (Character Generator Ram) เป็นหน่วยความจำภายในตัวโมดูล ใช้สำหรับเก็บภาพตัวอักษรที่ผู้ใช้สร้างขึ้นเอง (สร้างได้ 8 ตัว) โดยอ้างตำหน่งได้ 64 ไบต์ (8 ตัวอักษรคูณด้วย 8 แถว)
- การเขียนข้อมูลให้โมดูลแต่ละครั้งต้องตรวจสอบความพร้อมของโมดูล โดยตรวจสอบได้จาก Busy Flag หรือระยะเวลาการทำงานของโมดูล ซึ่งดูได้จากตารางคำสั่ง ดังนั้นเมื่อเขียนข้อมูลหนึ่งๆไปเราต้องหน่วงเวลารอให้โมดูลพร้อมจึงจะเขียนชุดใหม่ต่อไป
- การเขียนข้อมูลให้โมดูลสามารถทำได้ทั้งแบบ 4 บิตและแบบ 8 บิตขึ้นอยู่กับการเชื่อมต่อกับโมดูล ถ้าเป็นการเชื่อมแบบ 4 บิต การเขียนข้อมูลหรืออ่านข้อมูลต้องทำ 2 ครั้งโดยครั้งแรกต้องเป็นบิต 4 ถึงบิต 7 และครั้งที่ 2 จะเป็นบิตที่ 0 ถึง 3

ตารางคำสั่ง

INSTRUCTION	RS	R/W	DATA BIT 7	6	5	4	3	2	1	0	EXE. TIME (MicroS)
CLEAR DISPLAY	0	0	0	0	0	0	0	0	0	1	1640
CURSOR AT HOME	0	0	0	0	0	0	0	0	1	*	1640
ENTRY MODE SET	0	0	0	0	0	0	0	1	I/D	S	40
DISPLAY ON/OFF	0	0	0	0	0	0	1	D	C	B	40
DISPLAY SHIFT	0	0	0	0	0	1	S/C	R/L	*	*	40
FUNCTION SET	0	0	0	0	1	DL	N	F	*	*	40
SET CGRAM ADD.	0	0	0	1	CGRAM ADDRESS						40
SET DDRAM ADD.	0	0	1	DDRAM ADDRESS							40
BUSY,ADD. READ	0	1	BF	ADDRESS							0
CGRAM,DDRAM WR	1	0	WRITE DATA								40
CGRAM,DDRAM RD	1	1	READ DATA								40

คำสั่ง Entry Mode Set

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	1	I/D	S

I/D=0 กำหนดทิศทางของ Cursor และ DDRAM ให้เป็นแบบ Decrement
I/D=1 กำหนดทิศทางของ Cursor และ DDRAM ให้เป็นแบบ Increment
S=0 เมื่อเขียนข้อมูลแล้ว ตัวCursorจะถูกเลื่อนไปตามทิศทางของค่า I/D
S=1 เมื่อเขียนข้อมูลแล้วตัว Cursor จะอยู่กับที่และตัวอักษรจะถูกดันไปตามทิศทางของค่า I/D

4.DISPLAY ON/OFF

D=0 กำหนดให้ Off Display
D=1 กำหนดให้ On Display
C=0 กำหนดให้ Off Cursor
C=1 กำหนดให้ On Cursor โดย Cursor จะเป็นเส้นขีดใต้ตัวอักษร
B=0 กำหนดให้ไม่มีการกระพริบที่ตำแหน่ง Cursor
B=1 กำหนดให้มีการกระพริบที่ตำแหน่ง Cursor (กระพริบเป็นรูป □)

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	1	D	C	B

5.DISPLAY SHIFT

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	1	S/C	R/L	*	*

S/C=0 กำหนดให้เลื่อน Cursor ตามทิศทาง R/L ไป 1 ตำแหน่ง
S/C=1 กำหนดให้เลื่อนข้อความบนแผงแสดงตามทิศทาง R/L ไป 1 Column (เลื่อนทุกบรรทัด)
R/L=0 กำหนดให้มีทิศทางไปทางซ้าย R/L=1 กำหนดให้มีทิศทางไปทางขวา

6.FUNCTION SET

DL=0 กำหนดให้การติดต่อกับ LCD Module เป็นแบบ 4 bit
DL=1 กำหนดให้การติดต่อกับ LCD Module เป็นแบบ 8 bit จะสังเกตว่าการกำหนดค่าD/L มีสามารถกระทำได้ที่ DB4-DB7 ซึ่งถ้ามีการกำหนดให้เป็นแบบ 4 bit ตั้งแต่ครั้งแรกหลังจากจ่ายไฟเลี้ยงก็จะทำให้ LCD Module มีการรับข้อมูลแบบ 4 bit ทันที
N=0 กำหนดจำนวนบรรทัดแบบ 1/8 Duty และ 1/11 Duty
N=1 กำหนดจำนวนบรรทัดแบบ 1/16 Duty
F=0 กำหนดให้ตัวอักษรเป็นแบบ 5*7 Dots
F=1 กำหนดให้ตัวอักษรเป็นแบบ 5*10 Dots
(กรณีที่ LCD Module เป็นแบบ 5*7 อยู่แล้วก็จะไม่มีผลอะไร)

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	DL	N	F	*	*

7. SET CGRAM ADDRESS

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	1	CGRAM ADDRESS					

สำหรับการกำหนด Address ของ CGRAM เมื่อได้ทำการกำหนดไว้แล้วการอ่านและเขียน Data ที่ต่อจากนี้จะเป็นไปตามAddressที่กำหนดทันที

8. SET DDRAM ADDRESS

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	DDRAM ADDRESS						

สำหรับการกำหนด Address ของ DDRAM เมื่อได้ทำการกำหนดไว้แล้วการอ่านและเขียน Data ที่ต่อจากนี้จะเป็นไปตามAddressที่กำหนดทันที ตำแหน่งของ Address ในแต่ละรุ่นจะมีความแตกต่างกันบ้าง เพราะจำนวนตัวอักษรต่อบรรทัดไม่เท่ากัน ซึ่งแสดงดังตารางต่อไปนี้(ตารางนี้จะกำหนดให้บิทที่ 7 เท่ากับ 1 เสมอเพื่อความสะดวกในการเรียกใช้)

รุ่น DMC 082

80	81	82	83	84	85	86	87
C0	C1	C2	C3	C4	C5	C6	C7

รุ่น DMC 161

80	81	82	83	84	85	86	87	C0	C1	C2	C3	C4	C5	C6	C7

รุ่น DMC 162

80	81	82	83	84	85	86	87	88	89	8A	8B	8C	8D	8E	8F
C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	CA	CB	CC	CD	CE	CF

รุ่น DMC 202

80	81	82	83	84	85	86	87	88	89	8A	8B	8C	8D	8E	8F	90	91	92	93
C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	CA	CB	CC	CD	CE	CF	D0	D1	D2	D3

รุ่น DMC 204

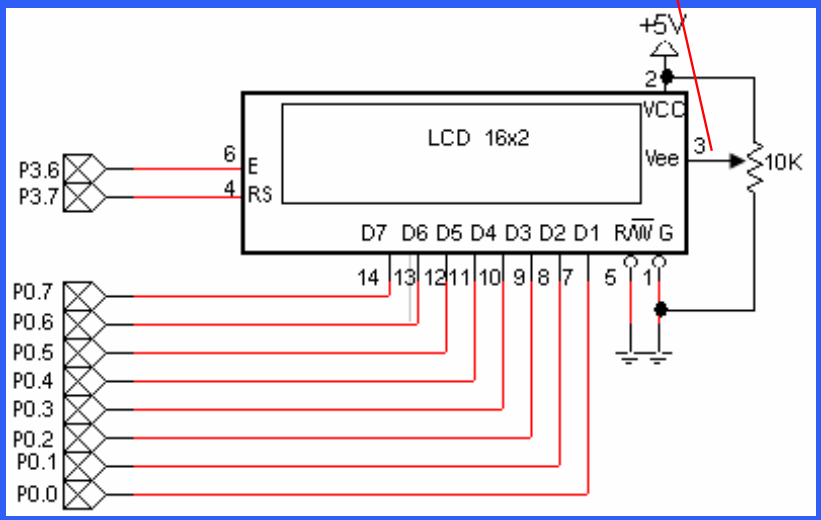
80	81	82	83	84	85	86	87	88	89	8A	8B	8C	8D	8E	8F	90	91	92	93
C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	CA	CB	CC	CD	CE	CF	D0	D1	D2	D3
94	95	96	97	98	99	9A	9B	9C	9D	9E	9F	A0	A1	A2	A3	A4	A5	A6	A7
D4	D5	D6	D7	D8	D9	DA	DB	DC	DD	DE	DF	E0	E1	E2	E3	E4	E5	E6	E7

รุ่น DMC 164

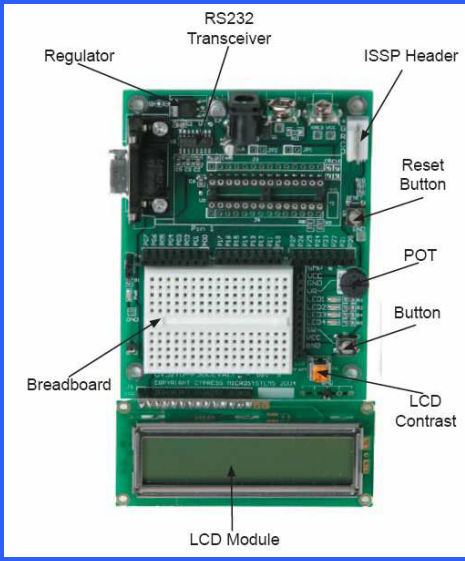
80	81	82	83	84	85	86	87	88	89	8A	8B	8C	8D	8E	8F
C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	CA	CB	CC	CD	CE	CF
90	91	92	93	94	95	96	97	98	99	9A	9B	9C	9D	9E	9F
D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	DA	DB	DC	DD	DE	DF

ตัวอย่างการต่อวงจร

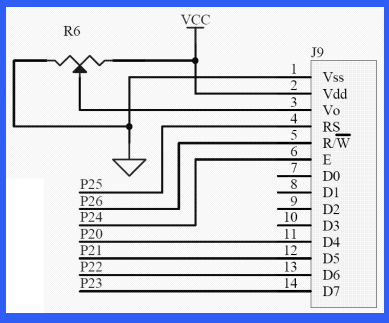
ประมาณ 0.75 โวลท์



PSoCEval



การต่อ LCD กับ PSoC



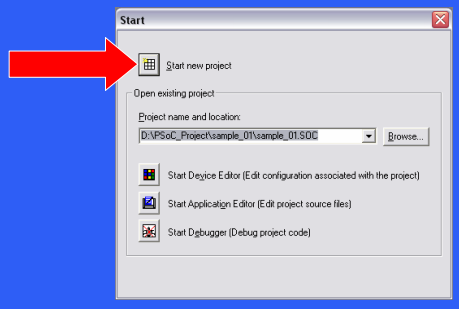
- ใช้พอร์ท 2 เป็นทั้งสัญญาณควบคุม และสัญญาณข้อมูล
- ต่อแบบ 4 บิต

สรุปขั้นตอนการใช้งาน PSoC Designer

ขั้นตอนที่	รายการ	ปุ่มคำสั่ง
1	สร้างโปรเจค	Start new project
2	กำหนดเบอร์ IC และภาษาที่ต้องการเขียน	
3	เลือกอุปกรณ์ภายในชิพ ทำการวางอุปกรณ์	Device Editor
4	กำหนดพารามิเตอร์	Interconnect View
5	สร้างระบบ	Generate Application
6	เขียนโปรแกรม Application	Application Editor
7	แปลโปรแกรม	Build
8	Download .Hex → chip	Program Part
9	Test program	

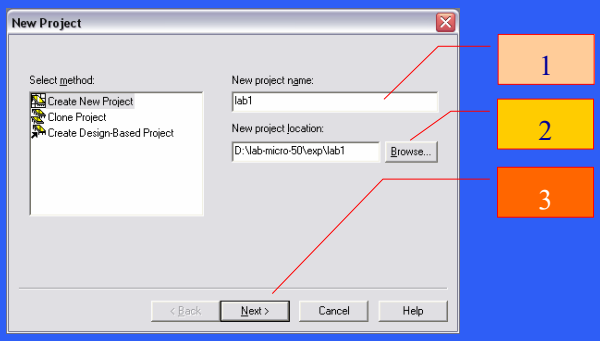
สร้างโปรเจค Start new Projec

1. รันโปรแกรม PSoC Designer โดยใช้ Start > Program > Cypress MicroSystem > PSoC Designer
2. เมื่อปรากฏหน้าต่าง Start ให้เลือก Start new project



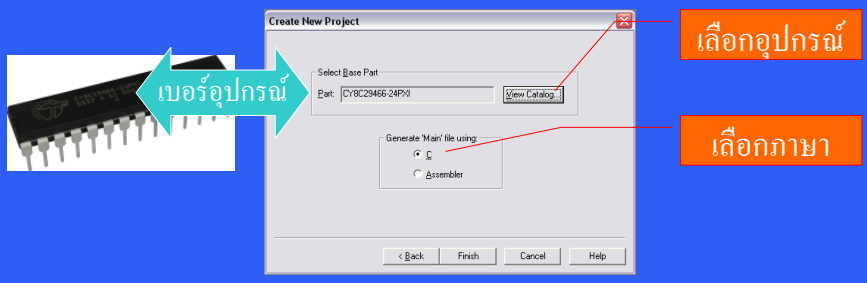
กำหนดชื่อและโฟลเดอร์ที่ต้องการเก็บโปรเจค

3. ในหน้าต่าง New Project ให้กำหนดโฟลเดอร์ที่จะเก็บโปรเจคและตั้งชื่อโปรเจค (จะมีการสร้างโฟลเดอร์ย่อยในโฟลเดอร์ที่ระบุ โดยมีชื่อตามชื่อโปรเจค) แล้วคลิกที่ปุ่ม Next



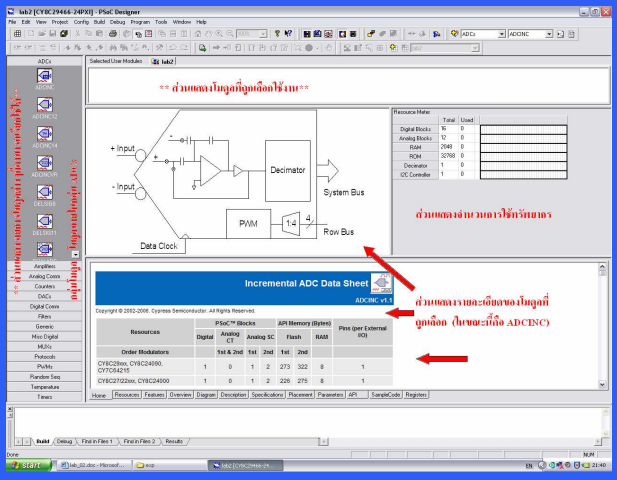
เลือกอุปกรณ์และภาษา

4. ในหน้าต่าง Create New Project ให้เลือก เบอร์ไมโครคอนโทรลเลอร์ตามที่อยู่ในบอร์ดทดลอง โดยดูจากด้านบนของตัวถัง IC แล้วคลิกปุ่ม View Catalog เพื่อเลือกเบอร์ ในที่นี้ใช้เบอร์ CY8C29466-24PXI ส่วน Main file ให้เลือกเป็นภาษา C ตามรูป แล้วคลิกที่ปุ่ม Finish

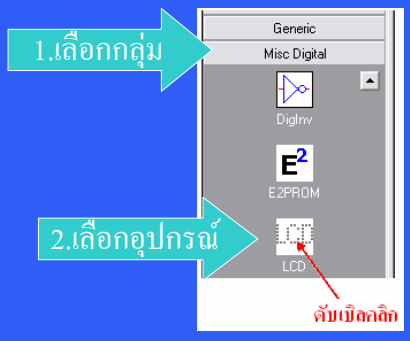


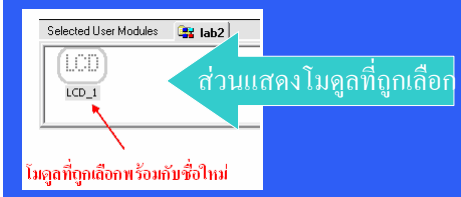
เลือกอุปกรณ์ภายในชิพ

- เลือก User Module Selection View Device Editor



เลือกอุปกรณ์ภายในชิพ
การเลือกใช้โมดูล LCD

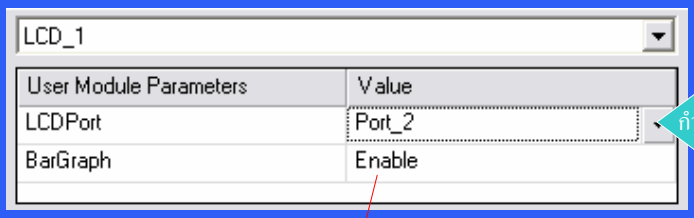




กำหนดค่าพารามิเตอร์

กำหนดค่าพารามิเตอร์ของ LCD

- คลิก Interconnect View 



ให้สามารถแสดงกราฟแท่งได้หรือไม่
- Enable ให้แสดงได้
- Disable ไม่แสดง

*** สำหรับ User port กำหนดโดยอัตโนมัติด้วยการกำหนดพอร์ทนี้ ถ้าต้องการใช้อย่างอื่นก็กำหนดเพิ่มเติมได้

สร้างระบบ

- คลิก Generate Application 

ได้สร้างซอร์ตไฟล์เบื้องต้น

```
//-------------------------------------------------------
// C main line
//-------------------------------------------------------
#include <m8c.h>          // part specific ...
#include "PSoCAPI.h"      // PSoC API .....
void main()
{
    // Insert your main …
}
```

```
/*********************************
* Generated by PSoC
Designer ver 4.4  b1884 : 14
Jan, 2007
*********************************/
#include "LCD_1.h"
```

เขียนโปรแกรม

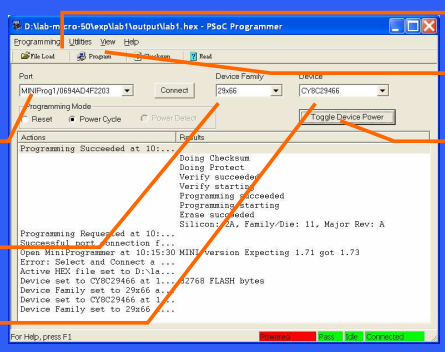
- **คลิกที่ปุ่ม Application Editor** 

  เพื่อเปิดหน้าต่าง Application เพื่อเขียน Source file ให้แก้ไขโปรแกรม main.c แล้วบันทึก

```
#include <m8c.h>        // part specific constants and macros
#include "PSoCAPI.h"    // PSoC API definitions for all User Modules
void main()
{
   char theStr1[] = "The 5th December";  // Define RAM string 1
   char theStr2[] = "Father's Day";      // Define RAM string 1
   LCD_1_Start();                        // Initialize LCD
   LCD_1_Position(0,0);                  // Place LCD cursor at row 0, col 0.
   LCD_1_PrString(theStr1);              // Print "PSoC LCD" on the LCD
   LCD_1_Position(1,2);                  // Place LCD cursor at row 1, col 2.
   LCD_1_PrString(theStr2);              // Print "PSoC LCD" on the LCD
}
```


แปลโปรแกรมและดาว์นโหลดโปรแกรม

1. แปลโปรแกรม คลิก Build 
2. ดาว์นโหลดโปรแกรม คลิก Program Part 



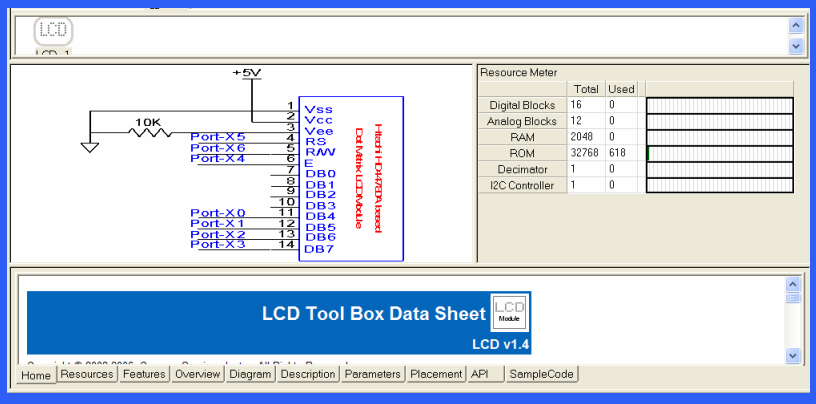
การเลือกทำงานที่หน้าต่างใดๆ

หน้าต่าง ต่างๆ เช่น

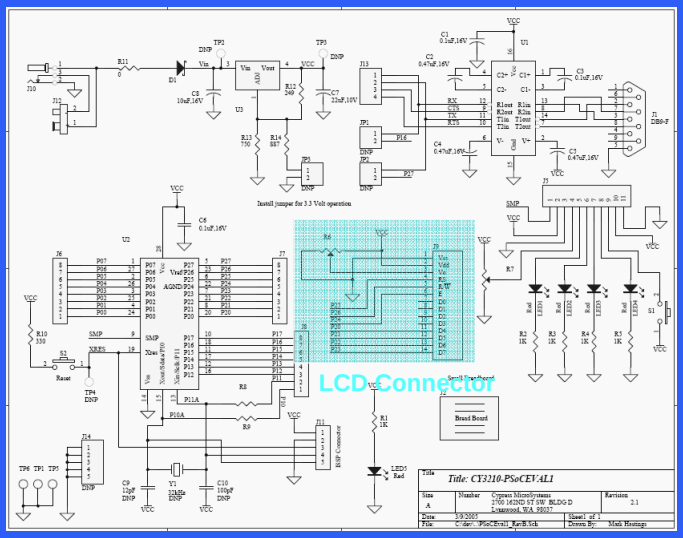
- *หน้าต่าง User Module Selection View*
- *หน้าต่าง Interconnect View*
- *หน้าต่าง Application Editor*

สามารถเปิดขึ้นมาดูเมื่อไรก็ได้ โดยคลิกที่ปุ่มคำสั่งเปิดหน้าต่างนั้นๆ

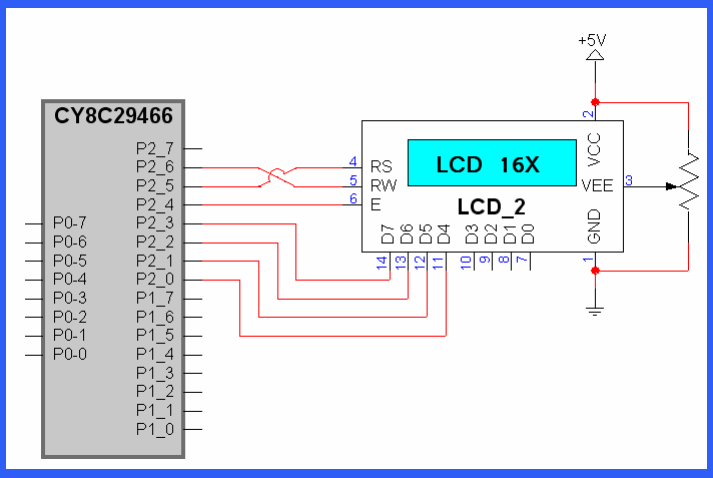
User Module Selection View



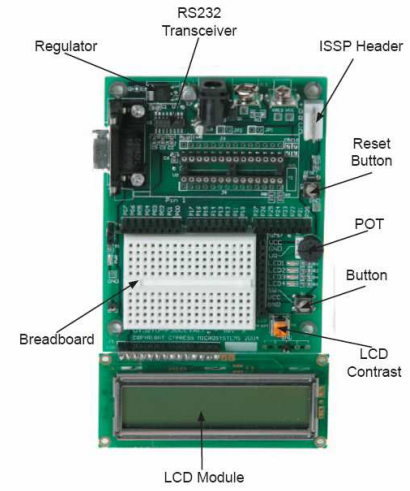
PSoC EVAL1 Schematic diagram



การต่อ LCD กับ PSoC



การต่อ LCD กับ PSoC



รายละเอียด โปรแกรมการทดลองที่ 1

```
#include <m8c.h>        // part specific constants and macros
#include "PSoCAPI.h"    // PSoC API definitions for all User Modules
void main()
{
  char theStr1[] = "The 5th December";      // Define RAM string 1
  char theStr2[] = "Father's Day";          // Define RAM string 1
  LCD_1_Start();                            // Initialize LCD
  LCD_1_Position(0,0);                      // Place LCD cursor at row 0, col 0.
  LCD_1_PrString(theStr1);                  // Print "PSoC LCD" on the LCD
  LCD_1_Position(1,2);                      // Place LCD cursor at row 1, col 2.
  LCD_1_PrString(theStr2);                  // Print "PSoC LCD" on the LCD
}
```

PSoCAPI.h

```
/*****************************************************************************
*  Generated by PSoC Designer ver 4.4  b1884 : 14 Jan, 2007
*****************************************************************************/
#include "LCD_1.h"
```


LCD_1.h

```
#pragma fastcall16  LCD_1_Start
#pragma fastcall16  LCD_1_PrString
#pragma fastcall16  LCD_1_PrHexInt
……..
//-------------------------------------------------
// Prototypes of the LCD_1 API.
//-------------------------------------------------
……..
extern void  LCD_1_Start(void);
extern void  LCD_1_WriteData(BYTE bData);
extern void  LCD_1_PrString(char * sRamString);
extern void  LCD_1_PrCString(const char * sRomString);
extern void  LCD_1_Position(BYTE bRow, BYTE bCol);
extern void  LCD_1_PrHexByte(BYTE bValue);
extern void  LCD_1_PrHexInt(INT iValue);
……..
……..
```

LCD API

ดู
http://narong.ece.engr.tu.ac.th/microlab/doc/index.html

#include <stdlib.h>

itoa(TextBuff,Number,10);

เป็น Library Functions สามารถเปิดดูได้จาก

Help > Document > C Language Compiler User Guide.pdf

Table 5-1. String Functions

Function	Prototype and Description	Header
abs	int abs(int); Returns the absolute value of number.	stdlib.h
atof	double atof(CONST char *); Converts a string to double. Returns the double value produced by interpreting the input characters as a number. The return value is 0.0 if the input cannot be converted to a value of that type. The return value is undefined in case of over-flow.	stdlib.h
atoi	int atoi(CONST char *); Converts a string to integer. Returns the int value produced by interpreting the input characters as a number. The return value is 0 if the input cannot be con-verted to a value of that type. The return value is undefined in case of overflow.	stdlib.h

อื่นๆที่มีอยู่ในคู่มือ

Compiler Basics

The following type definitions are included in the *m8c.h* file:

```
typedef unsigned char BOOL;
typedef unsigned char BYTE;
typedef signed char CHAR;
typedef unsigned int WORD;
typedef signed int INT;
typedef unsigned long DWORD;
typedef signed long LONG;
```


คำถาม

1. ให้สรุปผลที่ได้จากการทดลองของแต่ละโปรแกรม
2. ให้เขียนโปรแกรมเกมโดยใช้โมดูลแอลซีดี