การทดลองที่ 5

การใช้งาน PWM และ การควบคุมเซอร์โวมอเตอร์

(Implementation of PWM and Servo motor control by PWM module.)

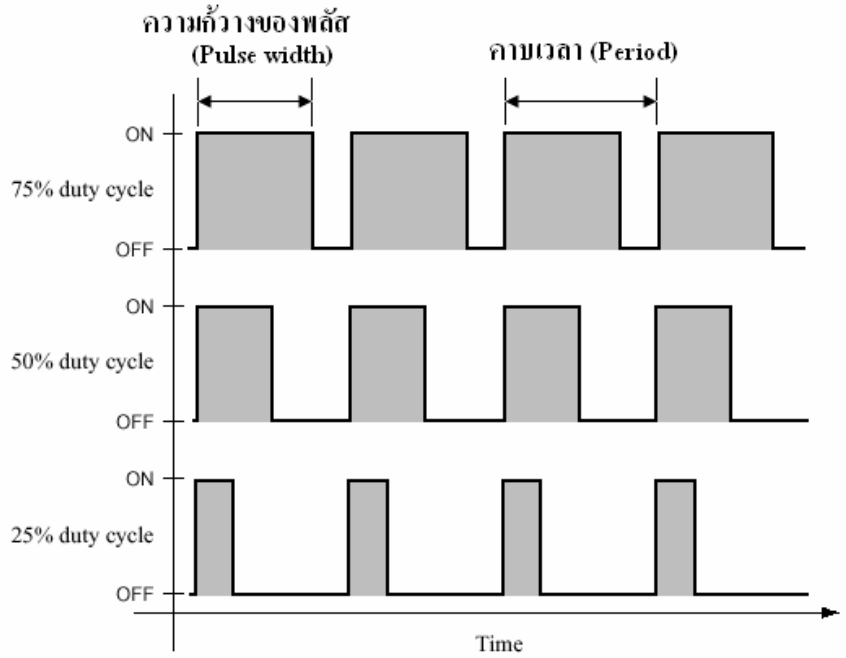
วัตถุประสงค์

1. เพื่อให้เข้าใจการทำงานของ PWM และสามารถเขียนโปรแกรมควบคุมการทำงานของ PWMได้
2. เพื่อให้เข้าใจการทำงานของเซอร์โวมอเตอร์ และสามารถเขียนโปรแกรมควบคุมการทำงานของเซอร์โวมอเตอร์ได้ ด้วยการใช้โมดูล PWM

บทนำ

การทดลองนี้ ได้แสดงวิธีการใช้งานโมดูล PWM และการนำโมดูล PWM ไปควบคุมการหมุนของเซอร์โวมอเตอร์

PWM (Pulse Width Modulation) เป็นวิธีหนึ่งที่นิยมใช้กันมากในงานควบคุม เช่นการควบคุมความเร็วมอเตอร์ หลักการของวิธีนี้คือกระแสที่ป้อนเข้ามอเตอร์จะเป็นพลัส ที่มีความถี่คงที่ แต่ความกว้างของพลัสเปลี่ยนแปลงได้ ถ้าพลัสแคบความเร็วจะต่ำ แต่ถ้าพลัสกว้างความเร็วจะสูง ในรูปแสดงลักษณะชองสัญญาณพลัซ์ที่มีความกว้างของพลัซเป็น 75% 50% และ 25% ของค่าคาบเวลา



การทดลองที่ 5.1 แสดงการใช้งานโมดูล PWM

1. ให้รันโปรแกรม PSoC Designer แล้วทำการสร้างโปรเจคใหม่ โดยใช้ไมโครคอนโทรลเลอร์เบอร์ CY8C29466-24PXI ส่วน Main file ให้เลือกเป็นภาษา C

2. เมื่อเข้าสู่ PSoC designer ในหน้าแรกจะเป็น User Module Selection View ในหน้าต่างนี้ให้เลือกใช้โมดูล แอลซีดี (LCD) และโมดูล PWM16 ในกลุ่ม PWMs
3. หลังจากเลือกโมดูลต่างๆแล้ว ให้คลิก  (Interconnect View) เพื่อทำการวางอุปกรณ์และกำหนดค่าพารามิเตอร์ ของ LCD และ Port_0_4 สำหรับต่อกับสวิตซ์ Push button (SW) สำหรับ PWM16_1 นั้น ต้องการใช้สัญญาณนาฬิกาจาก VC2 ที่ความถี่ 1 MHz เป็นสัญญาณนาฬิกาของโมดูล และเตรียม VC3 เป็นสัญญาณนาฬิกาสำหรับการกำหนดค่า Baudrate ของโมดูล UART ดังนั้นจึงกำหนดให้พารามิเตอร์ของ Grobal Resource และของ PWM16_1 เป็นดังนี้

   VC2 = VC1/3 จากกำหนดพารามิเตอร์ของ Global Resource

   VC1 = SysClk/8 จากกำหนดพารามิเตอร์ของ Global Resource

   VC2 = SysClk/(8x3) = 24MHz/24 = 1 MHz

   VC3 = 8 x Baudrate = 8 x 9600 = 76.8 K

   VC3 = VC1/n

   n = 3000/76.8 = 39.0625 ดังนั้นใช้ n หรือ VC3 Divider = 39

Global Resources	Value
Power Setting [Vcc / SysClk freq	5.0V / 24MHz
CPU_Clock	SysClk/8
32K_Select	Internal
PLL_Mode	Disable
Sleep_Timer	512_Hz
VC1= SysClk/N	8
VC2= VC1/N	3
VC3 Source	VC1
VC3 Divider	39
SysClk Source	Internal
SysClk*2 Disable	No
Analog Power	SC On/Ref Low
Ref Mux	(Vdd/2)+/-BandGap
AGndBypass	Disable
Op-Amp Bias	Low
A_Buff_Power	Low
SwitchModePump	OFF
Trip Voltage [LVD (SMP)]	4.81V (5.00V)
LVDThrottleBack	Disable
Watchdog Enable	Disable

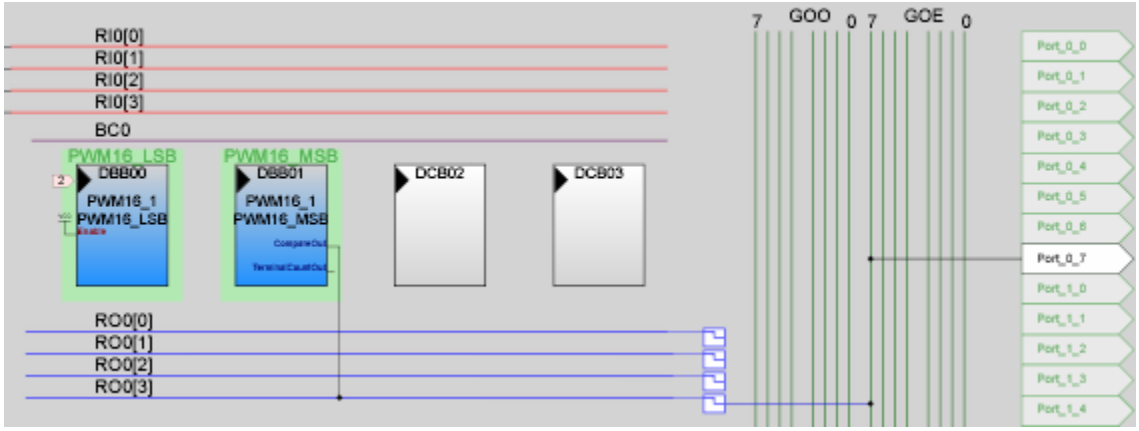
PWM16_1

User Module Parameters	Value
Clock	VC2
Enable	High
CompareOut	Row_0_Output_3
TerminalCountOut	None
Period	0
PulseWidth	0
CompareType	Less Than Or Equal
InterruptType	Terminal Count
ClockSync	Sync to SysClk
InvertEnable	Normal

Name	Port	Select	Drive	Interrupt
Port_0_0	P0[0]	StdCPU	High Z Analog	DisableInt
Port_0_1	P0[1]	StdCPU	High Z Analog	DisableInt
Port_0_2	P0[2]	StdCPU	High Z Analog	DisableInt
Port_0_3	P0[3]	StdCPU	High Z Analog	DisableInt
Port_0_4	P0[4]	StdCPU	Pull Down	DisableInt
Port_0_5	P0[5]	StdCPU	High Z Analog	DisableInt
Port_0_6	P0[6]	StdCPU	High Z Analog	DisableInt
Port_0_7	P0[7]	GlobalOutEv	Strong	DisableInt

ส่วนตำแหน่งขาเอาท์พุทของ PWM กำหนดให้ออกที่ Port_0_7 (ดูวิธีการกำหนดจากการทดลองที่ 4)

4. เมื่อเสร็จแล้วให้กด  (Generate Application) เพื่อสร้างซอร์ไฟล์ต่างๆ
5. ให้คลิกที่ปุ่ม  (Application Editor) เพื่อเปิดหน้าต่าง Application เพื่อเขียน Source file ให้แก้ไขโปรแกรม main.c เป็น ดังนี้

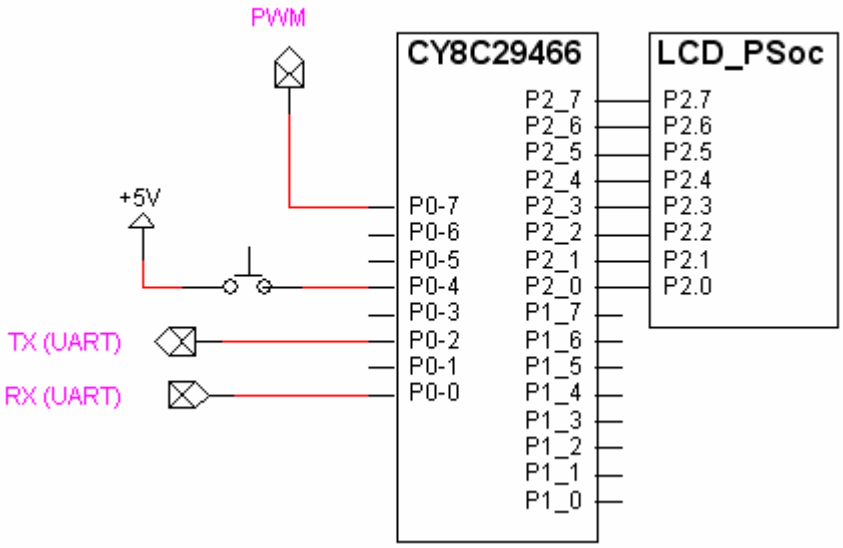


โปรแกรมที่ 5.1

```
//-------------------------------------------------------------------------
// Experiment 5.1
//-------------------------------------------------------------------------

#include <m8c.h>        // part specific constants and macros
#include "PSoCAPI.h"    // PSoC API definitions for all User Modules
#include <stdlib.h>

void main()
{
        DWORD i;
        char TextBuff[10];

        LCD_1_Start();                  // Initialize LCD
        PWM16_1_Start();                                        //
        PWM16_1_WritePeriod(2000);

        LCD_1_Position(0,0);
        LCD_1_PrCString("PWM Test");
        i = 1000;
        PWM16_1_WritePulseWidth(i);                     // Set Pulse width
        while(1)
        {
                if (i > 1999) {i = 20;}

        if(PRT0DR & 0X10)                               // Test Port_0_4
        {
                i++;
                LCD_1_Position(1,0);
                LCD_1_PrCString("Duty cycle ");
                itoa(TextBuff,(i*100)/2000,10);   //convert integer to Duty cycle string base 10
                        LCD_1_PrString(TextBuff);
                        LCD_1_PrCString(" %      ");
                        PWM16_1_WritePulseWidth(i);
                }
        }
}
```

6. ให้ทำการบันทึกและแปลโปรแกรม

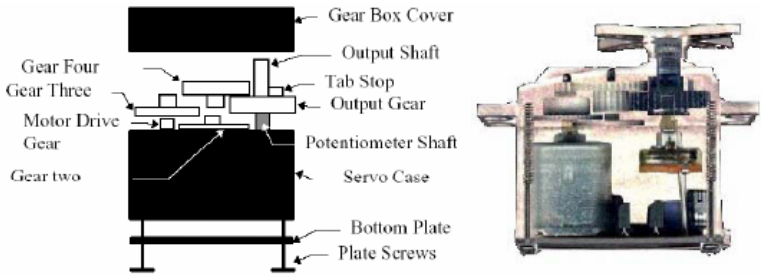
7. ติดตั้ง LCD ต่อสายสวิตซ์ และ สโคป เพื่อวัดสัญญาณ PWM ที่ Port_0_7 ตามรูป แล้วดาว์นโหลดโปรแกรมลงชิพ



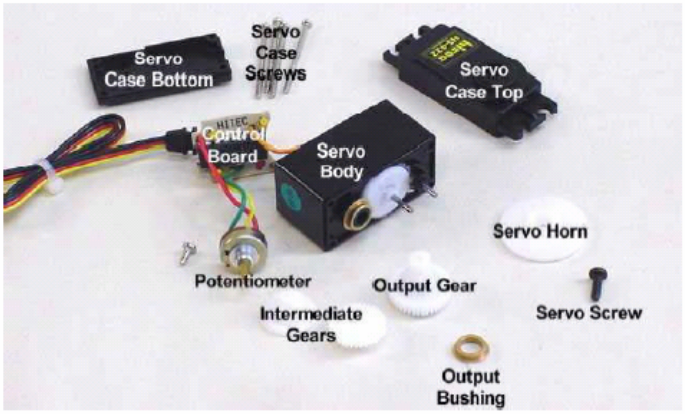
8. เมื่อดาว์นโหลดเสร็จ ให้ทำการทดสอบการทำงาน โดยการกดสวิตช์ แล้วใช้สโคปจับสัญญาณพลัซและทำการบันทึกค่าสัญญาณพลัซ์ที่ค่า Duty cycle ต่างๆมา 3 ค่า
9. ให้อธิบายถึงค่าความถี่และ Duty cycle ของสัญญาณว่าสอดคล้องกับค่าความถี่ VC2 กับค่า “i” ในโปรแกรมอย่างไร

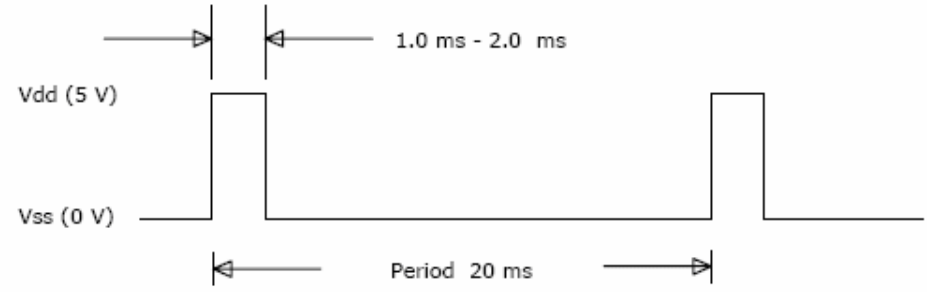
เซอร์โวมอเตอร์

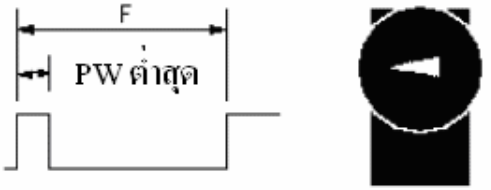
Servo motor คือ มอเตอร์ไฟฟ้ากระแสตรง (DC motor) ที่ถูกประกอบรวมกับ ชุดเกียร์ และ ส่วนควบคุม ต่างๆ ไว้ ในโมดูลเดียวกัน หรือ ภายในกล่องพาสติกเดียวกัน โดยมอเตอร์ชนิดนี้จะมีสายต่อใช้งานเพียง 3 เส้นเท่านั้น คือ VCC,GND และ สายสัญญาณควบคุม(Control Line) ซึ่งสามารถควบคุมให้มอเตอร์หมุนซ้าย หรือ ขวาได้จากสายสัญญาณเพียงเส้นเดียว โดยสัญญาณที่ใช้ควบคุมนี้จะเป็นสัญญาณพัลส์วิดมอด (PWM) แบบ TTL Level ระดับแรงดันที่จ่ายให้มอเตอร์นี้จะอยู่ใน ช่วงประมาณ 4 ถึง 6 โวลท์ ดังแสดงในรูป



ส่วนประกอบต่างๆของ Servo Motor

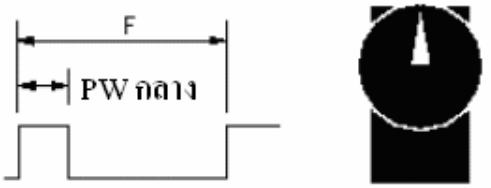






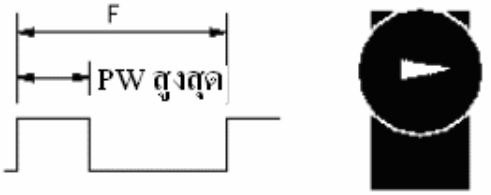
ตำแหน่งเซอร์โว 0 องศา

ความกว้างพลัซ 0.7 ms - 1 ms



ตำแหน่งเซอร์โว 90 องศา

ความกว้างพลัซ 1.5 ms



ตำแหน่งเซอร์โว 180 องศา

ความกว้างพลัซ 2 ms - 2.3 ms

หมายเหตุ ตำแหน่งมุมที่เรียกแบบนี้ เพื่อให้มุมเป็นเลขจำนวนเต็มบวก

ความสัมพันธ์ระหว่างค่า Pulse ของโมดูล PWM กับมุมของเซอร์โว

จากความถี่ 1 MHz : T = 1/F = 1/1MHz = 1 uS

หมายความว่าถ้านับสัญญาณนี้ 1 ไซเคิลจะได้เวลา 1 uS

ดังนั้นถ้าต้องการเวลา 20 mS ต้องใช้ = 20 mS/1uS = 20000 ไซเคิล

นั้นหมายถึงว่าค่าที่จะใช้ในการสร้างความก้วางของพลัซของโมดูล PWM เท่ากับ 20000

ในทำนองเดียวกัน

ที่มุม 0 องศา ต้องการเวลา 0.7 mS ต้องใช้ = 0.7 mS/1uS = 700

ที่มุม 90 องศา ต้องการเวลา 1.5 mS ต้องใช้ = 1.5 mS/1uS = 1500

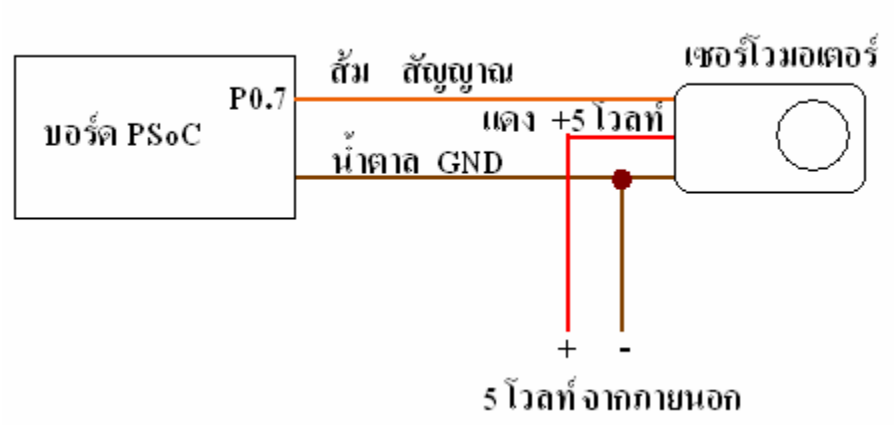
ที่มุม 180 องศา ต้องการเวลา 2.3 mS ต้องใช้ = 2.3 mS/1uS = 2300

การทดลองที่ 5.2 แสดงการควบคุมเซอร์โวมอเตอร์ด้วย PWM

1. แก้ไขโปรแกรม main.c เป็น ดังนี้

```
//-------------------------------------------------------------------------
// Experiment 5.2
//-------------------------------------------------------------------------

#include <m8c.h>        // part specific constants and macros
#include "PSoCAPI.h"    // PSoC API definitions for all User Modules
#include <stdlib.h>

void main()
{
        DWORD i;
        char TextBuff[10];
        LCD_1_Start();                          // Initialize LCD
        PWM16_1_Start();                        // Start PWM
        PWM16_1_WritePeriod(20000);
        LCD_1_Position(0,0);
        LCD_1_PrCString("Servo motor Test");
        i = 1500;                               // Set to 90 deg.
        PWM16_1_WritePulseWidth(i);             // Set Pulse width
        while(1)
        {
                i++;
                if (i > 2300) {i = 700;}

                if(PRT0DR & 0X10)                       // Test Port_0_4
                {
                        PWM16_1_WritePulseWidth(i);   // Set Pulse width
                        LCD_1_Position(1,0);
                        LCD_1_PrCString("Angle ");
                        itoa(TextBuff,((i-700)*9)/80,10);   //convert integer to Angle for display
                        LCD_1_PrString(TextBuff);
                        LCD_1_PrCString(" Deg ");
                }
        }
}
```

2. แล้วทำการบันทึกไฟล์ แล้วให้ทำการแปลโปรแกรม
3. ทำการติดตั้ง LCD เข้ากับบอร์ด และต่อสวิทช์ กดติดปล่อยดับเข้าที่ P0.4 เหมือนการทดลองที่ 5.1
4. ติดตั้งเซอร์โวมอเตอร์เข้ากับบอร์ดทดลองโดยใช้ P0.7 เป็นสายสัญญาณควบคุม ตามรูป แต่เนื่องจากไฟดีซีที่มาจ่ายให้กับบอร์ดทดลองได้มาจากพอร์ท USB ซึ่งมีปริมาณกระแสจำกัด ไม่พอที่จะจ่ายให้กับมอเตอร์ ดังนั้นต้องใช้แหล่งจ่าย +5 โวลท์จากภายนอก **ขั้นตอนนี้ขอให้ทำด้วยความระมัดระวังมิฉะนั้นอาจเกิดความเสียหายคอมพิวเตอร์และบอร์ดทดลองและหรือเครื่องพีซีได้**



5. ให้ต่อสาย USB แล้วดาว์นโหลดโปรแกรมลงชิพ
6. เมื่อดาว์นโหลดเสร็จ ให้ทำการทดสอบการทำงาน โดยกดสวิทช์เพื่อดูการหมุนของมอเตอร์และค่าที่แสดงออกทาง LCD

คำถาม

1. ให้อธิบายการทำงานของโปรแกรม ทั้ง 2 โปรแกรม
2. ให้เขียนโปรแกรมควบคุมการหมุนของเซอร์โวมอเตอร์ โดยป้อนองศาที่ต้องการให้หมุนผ่านทาง Hyper terminal แล้วแสดงค่าองศาออกทาง LCD และมอเตอร์หมุนไปยังมุมที่กำหนด (ดูตัวอย่างการป้อนผ่าน Hyper terminal จาก การทดลองที่ 4)

หมายเหตุสำหรับข้อสองการคิดคะแนนจะคิดจาก 100% หารด้วยจำนวนกลุ่มที่ใช้โมดูลแบบเดียวกัน