การใช้งาน PWM และ การควบคุมเซอร์โวมอเตอร์

รศ.ณรงค์ บวบทอง
ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์
คณะวิศวกรรมศาสตร์
มหาวิทยาลัยธรรมศาสตร์

หัวข้อ

- วัตถุประสงค์
- PWM คืออะไร
- การใช้งานโมดูล PWM ของ Arduino
- Application Programming Interface (API) เกี่ยวกับ PWM
- เซอร์โวมอเตอร์ (Servo motor) และการใช้งาน
- การใช้ไลบรารี่สำหรับสั่งงาน RC Servo Motor

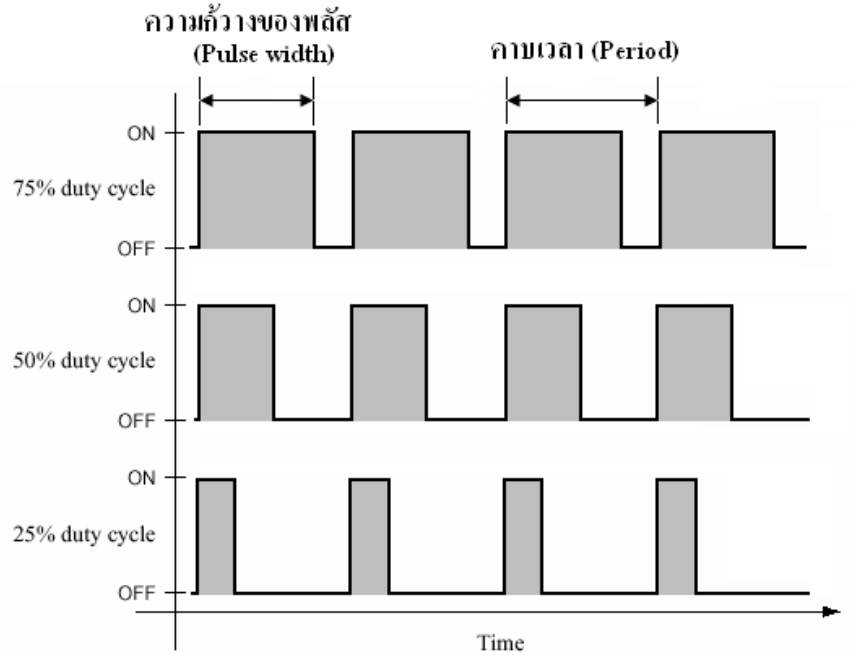
วัตถุประสงค์

- เพื่อให้เข้าใจการทำงานของ PWM และสามารถเขียนโปรแกรมควบคุมการทำงานของ PWMได้
- เพื่อให้เข้าใจการควบคุมความเร็วมอเตอร์ดีซีด้วย PWM
- เพื่อให้เข้าใจการทำงานของเซอร์โวมอเตอร์ และสามารถเขียนโปรแกรมควบคุมการทำงานของ เซอร์โวมอเตอร์ได้ ด้วยการใช้ Servo library

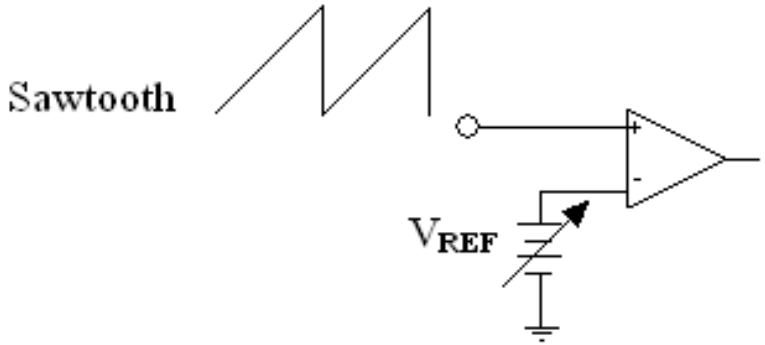
PWM คืออะไร

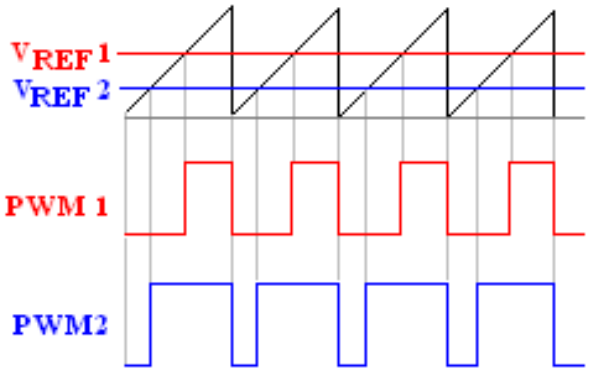
PWM หรือ Pulse Width Modulation คือสัญญาณพลัสที่มีค่าความถี่คงที่ แต่ความกว้างของพลัสเปลี่ยนแปลงได้

PWM เป็นวิธีหนึ่งที่นิยมใช้กันมากในงานควบคุม เช่นการควบคุมความเร็วมอเตอร์

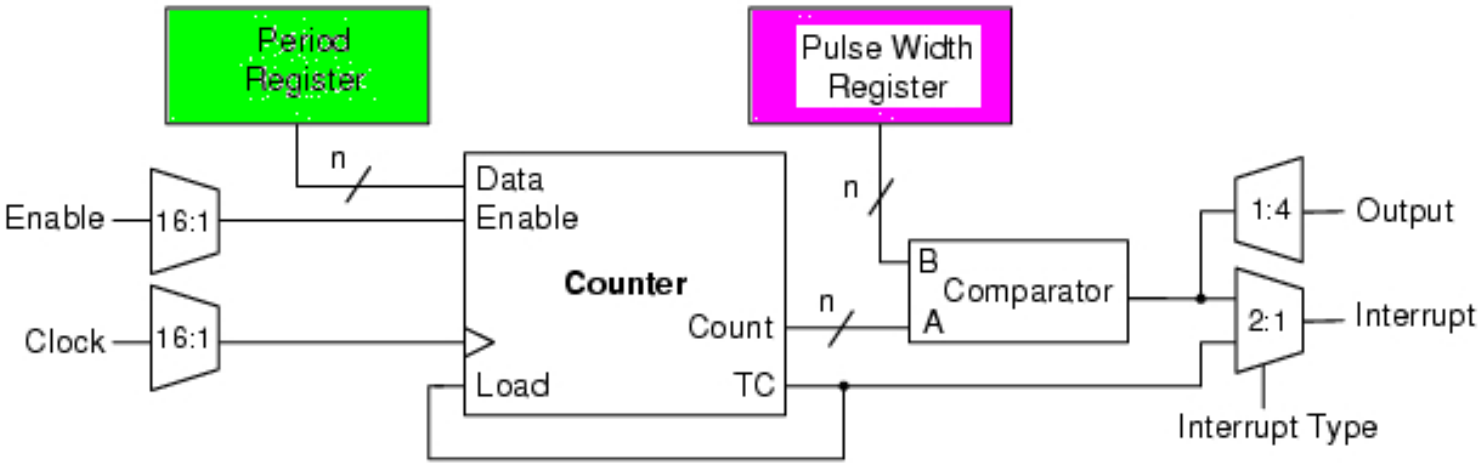


การสร้างสัญญาณ PWM ด้วยวิธีทาง Analog



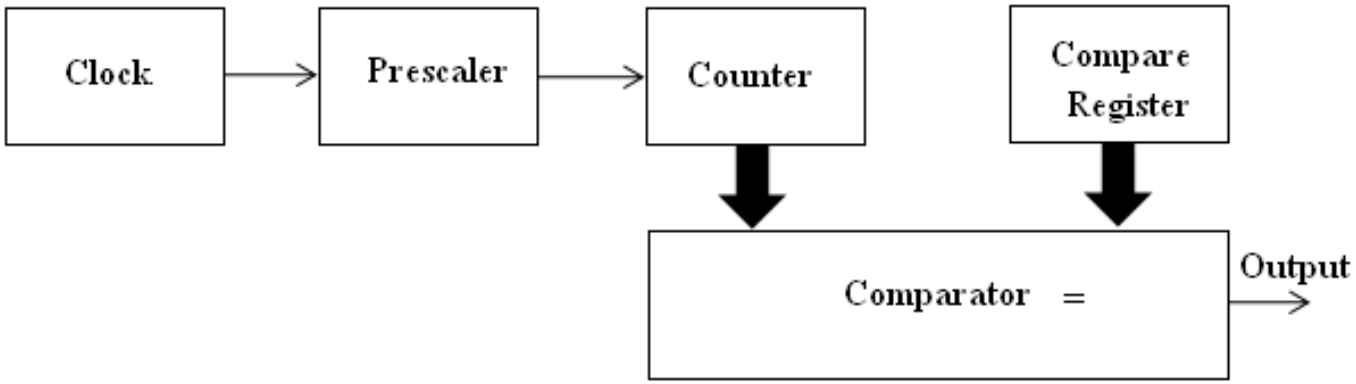


การสร้างสัญญาณ PWM ด้วยวิธีทาง Digital

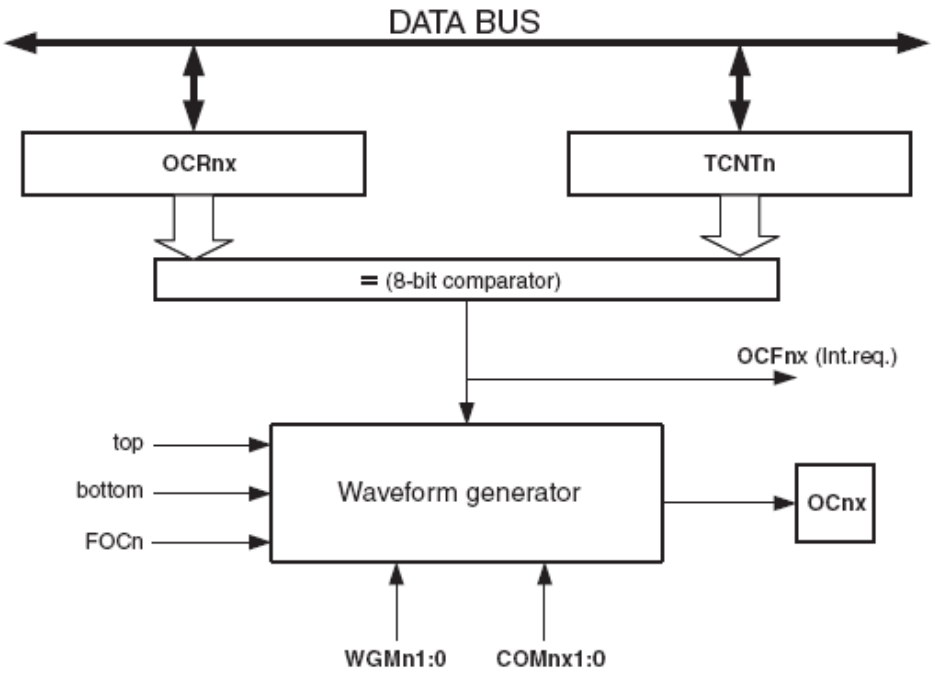


PWM Block Diagram, Data Path width n = 8 or 16

การสร้างสัญญาณ PWM ของ AVR

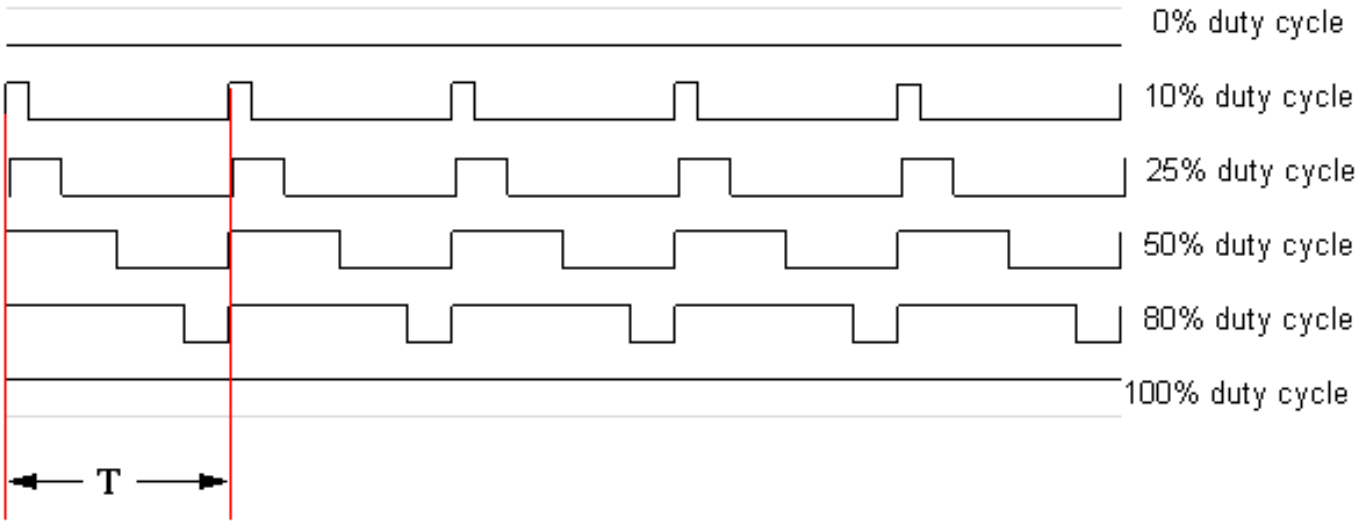


การสร้างสัญญาณ PWM ของ AVR (ต่อ)



OCRnx (OCR0A and OCR0B) = Output Compare Registers
TCNTn (TCNT0) = Timer/Counter Register
OCFnx (OCF0x) = Output Compare Flag
Ocnx = Timer/Counter1 output compare match output

สัญญาณ PWM



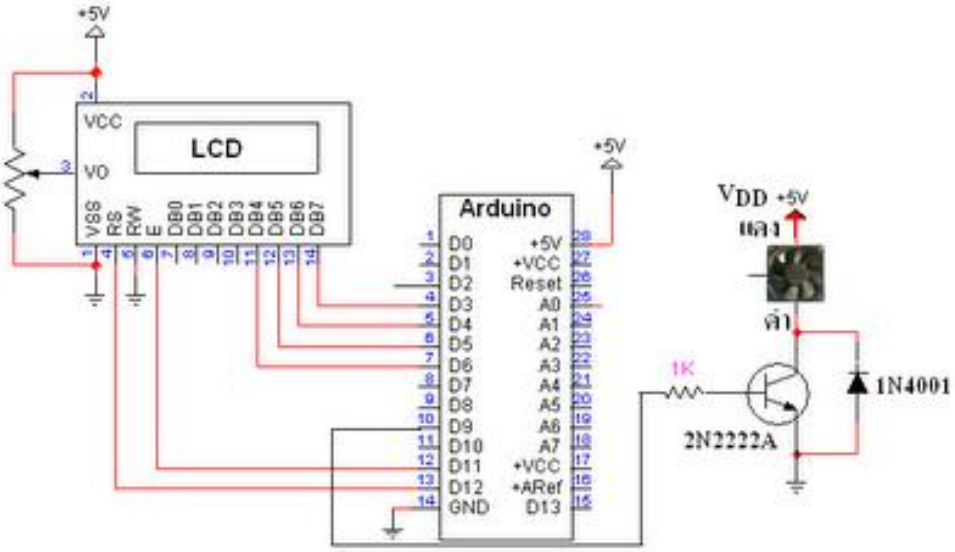
T = 2.04 mS
F = 1/T = 1000/2.04 = 490 Hz

การสร้างสัญญาณ PWM อย่างง่ายๆด้วย analogWrite

- เป็นฟังก์ชั่นสร้างสัญญาณ PWM ออกทางขา Digital Out โดยมีความถี่ประมาณ 490 Hz T= 2.04 mS
- สำหรับ ATmega168 หรือ ATmega328 ใช้ได้กับขา 3, 5, 6, 9, 10, และ 11
- วิธีการใช้งาน analogWrite(Pin, Value)
- Pin : ขาเอาท์พุท
- Val : ค่า duty cycle มีค่าอยู่ระหว่าง 0 (0%) ถ้า 127 ห้ค่า duty cycle ประมาณ 50% และ 255 (100%)

โปรแกรมที่ 1

```
// set output pin for the PWM
int pwm_out = 9;

void setup() {
    // declare the PWM as an OUTPUT:
    pinMode(pwm_out, OUTPUT);
}

void loop() {
    analogWrite(pwm_out, 127);
}
```



การสร้างสัญญาณ PWM ด้วยค่าไลบรารี TimerOne.h

การตั้งค่า

- Timer1.initialize(microseconds); กำหนดค่าคาบเวลา (Period) มีหน่วยเป็นวินาที
- Timer1.setPeriod(microseconds); กำหนดค่าคาบเวลาใหม่ หลังจากใช้ Timer1.initialize แล้ว

การสร้างสัญญาณ PWM ด้วยค่าไลบรารี TimerOne.h (ต่อ)

การค่วบคุม

- Timer1.start(); สั่งให้ Timer เริ่มทำงาน
- Timer1.stop(); สั่งให้ Timer หยุดทำงาน
- Timer1.restart(); สั่งให้ Timer เริ่มทำงานใหม่หลังจากกำหนดค่าคาบใหม่
- Timer1.resume(); สั่งให้ Timer เริ่มทำงาน หลังจากที่สั่งหยุด

การสร้างสัญญาณ PWM ด้วยค่าไลบรารี TimerOne.h (ต่อ)

ฟังก์ชั่นเกี่ยวกับสัญญาณ PWM

- Timer1.pwm(pin, duty); กำหนดค่า duty cycle มีค่าตั้งแต่ 0 (0%) ถึง 1023 (100%) โดยสัญญาณจะออกที่ขา pin Ocxn (กรณี 168 คือ D9)
- Timer1.setPwmDuty(pin, duty); กำหนดค่า duty cycle ใหม่
- Timer1.disablePwm(pin); หยุดสัญญาณ PWM

การสร้างสัญญาณ PWM ด้วยค่าไลบรารี TimerOne.h (ต่อ)

ฟังก์ชั่นเกี่ยวกับการอินเตอร์รัพท์

- Timer1.attachInterrupt(function); กำหนดฟังก์ชั่นเพื่อรองรับการอินเตอร์รัพท์ ที่จะเกิดขึ้นทุกครั้งที่ Timer นับเวลาครบ
- Timer1.detachInterrupt(); กำหนดสถานการณ์อินเตอร์รัพท์ให้เป็นดิสเอเบิล (Disable)

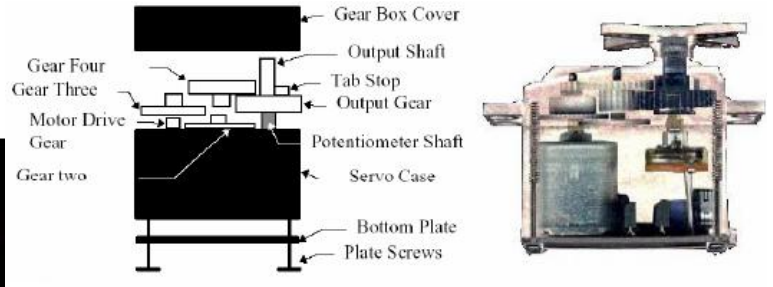
โปรแกรมที่ 2

```
#include "TimerOne.h"
int pwm_out = 9;
void setup()
{
  Timer1.initialize(1000);          // initialize timer1, and set 1000 mS
  Timer1.pwm(pwm_out, 512);  // setup pwm on pin 9, 50% duty
     cycle
}

void loop()
{
}
```

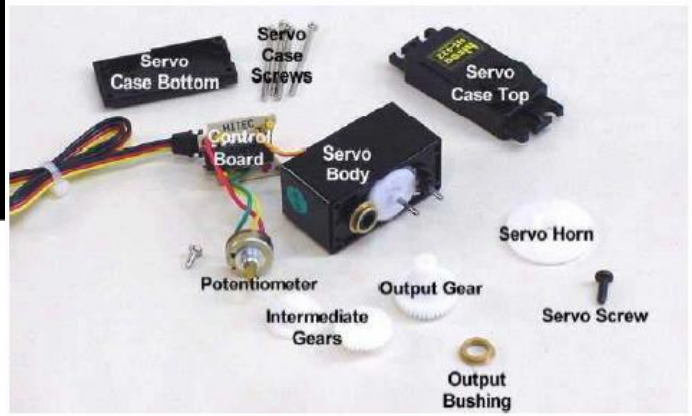
เซอร์โวมอเตอร์ (Servo motor) และการใช้งาน

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

ลักษณะของเซอร์โวมอเตอร์



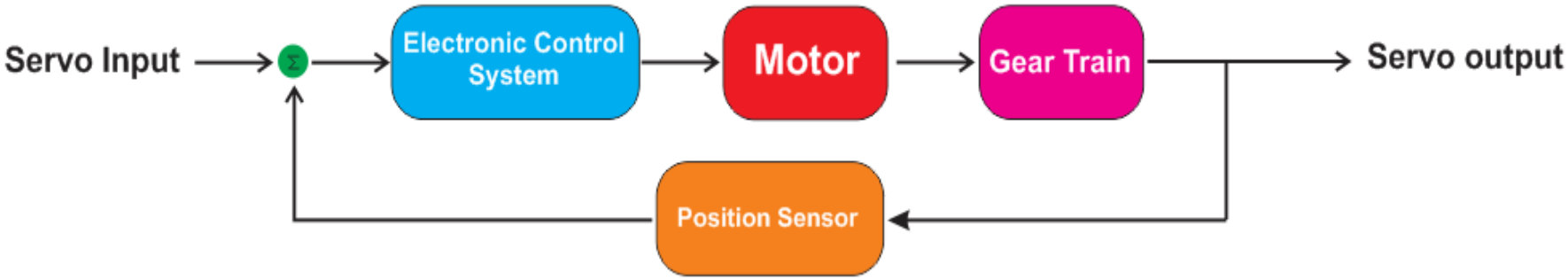
ส่วนประกอบต่างๆของ Servo Motor



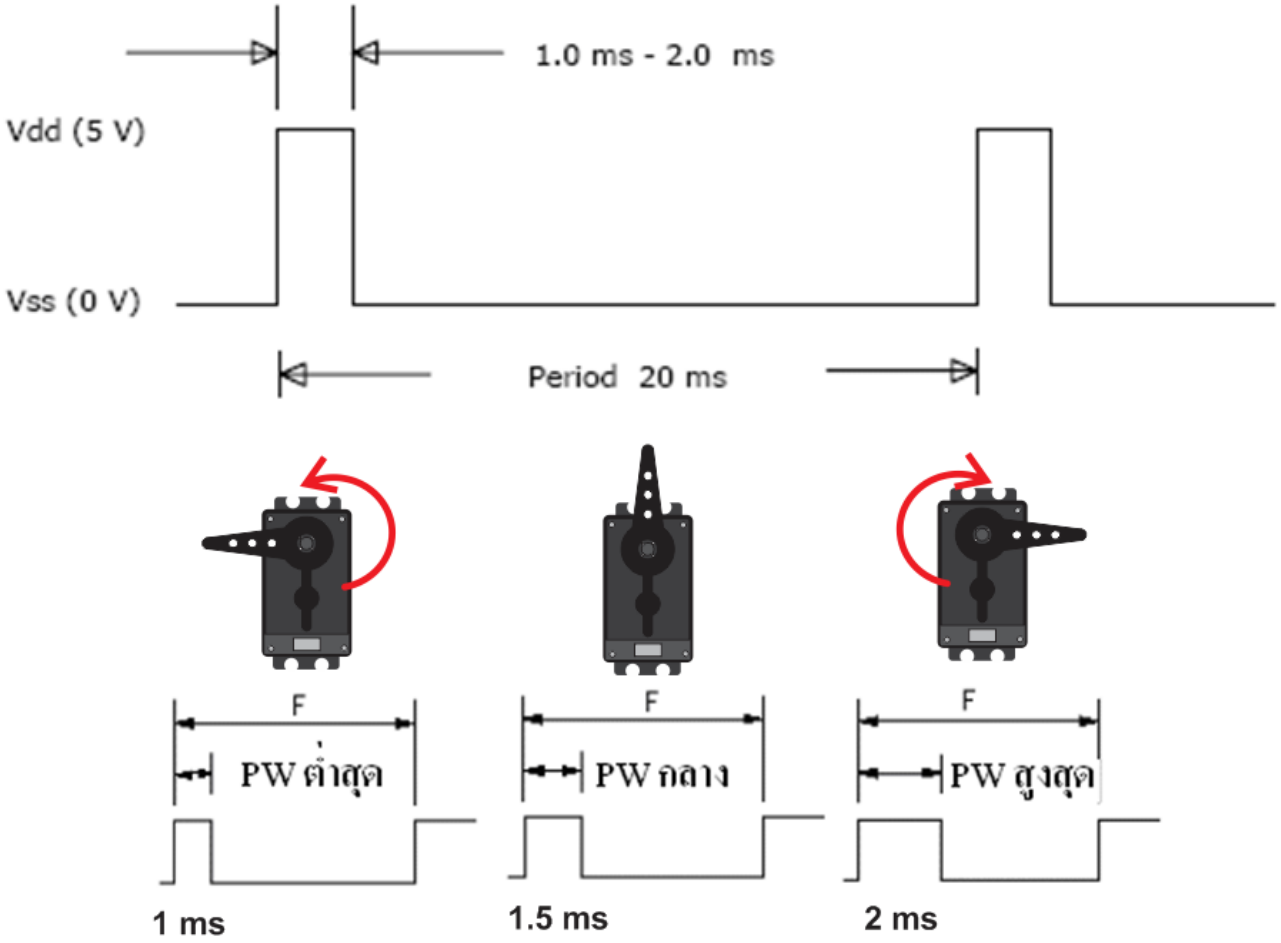
ลักษณะของเซอร์โวมอเตอร์ (ต่อ)

1. Motor เป็นส่วนของตัวมอเตอร์
2. Gear Train หรือ Gearbox เป็นชุดเกียร์ทดแรง
3. Position Sensor เป็นเซ็นเซอร์ตรวจจับตำแหน่งเพื่อหาค่าองศาในการหมุน
4. Electronic Control System เป็นส่วนที่ควบคุมและประมวลผล

Servo Motor Block Diagram



การควบคุมเซอร์โวมอเตอร์

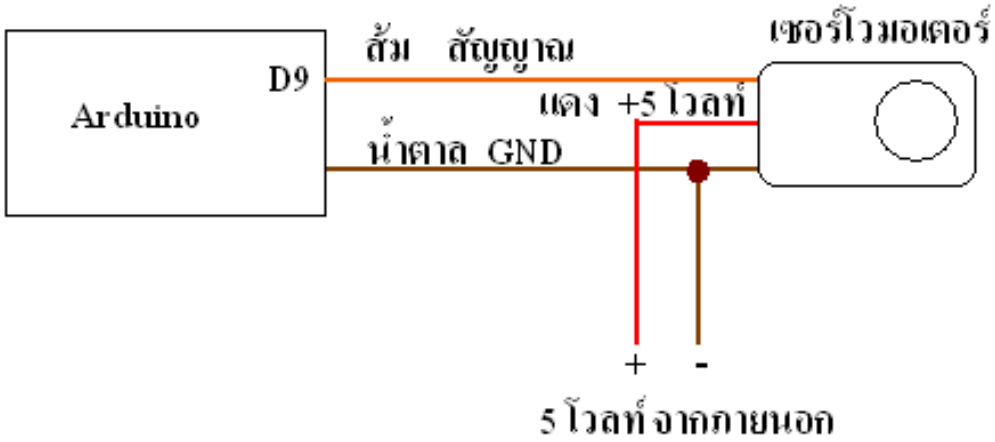


ตัวอย่างการกำหนดค่าความก้วางของพัลส์ สำหรับการควบคุมเซอร์โวมอเตอร์ ด้วยคำสั่ง ในไลบรารี TimerOne.h

มุม 0 – 180 องศา ต้องใช้พัลส์ 0.7 – 2.3 ms
ดังนั้น 180 ได้เป็นช่วงเวลา 1.6
n องศา คิดเป็นเวลา (1.6*n/180) + 0.7
แปลงจากองศา 0-180 เป็นค่า 0 - 1023
n องศา คิดเป็นค่า ((1.6*n/180) + 0.7)*1024/20

ที่มุม 0 องศา ต้องใช้ = 0.7 *1024/20
= 35
ที่มุม 90 องศา ต้องใช้ = ((1.6*90/180)+0.7) *1024/20
= 76
ที่มุม 180 องศา ต้องใช้ = ((1.6*180/180)+0.7) *1024/20
= 102

วงจรทดลอง


Arduino
D9
ส้ม สัญญาณ
แดง +5 โวลท์
น้ำตาล GND
เซอร์โวมอเตอร์
+
-
5 โวลท์ จากภายนอก


MAX
ES08D

โปรแกรมที่ 3 ควบคุมเซอร์โวมอเตอร์ โดยใช้ ไลบรารี TimerOne.h

```
#include "TimerOne.h"
int pwm_out = 9;
void setup()
{
  Timer1.initialize(20000);         // initialize timer1, and set 1000 uS
  Timer1.pwm(pwm_out, 76);  // setup pwm on pin 9, for 90 degree
  delay(200)                              // delay time
  Timer1.stop();
}

void loop()
{
 }
```

การใช้ไลบรารี่สำหรับสั่งงาน RC Servo Motor

- ไลบรารี่มีมาพร้อมกับ Arduino IDE แล้ว
- ฟังก์ชั่น
  - attach()
  - write()
  - writeMicroseconds()
  - read()
  - attached()
  - detach()

คำอธิบายดูจาก https://www.arduino.cc/en/Reference/Servo

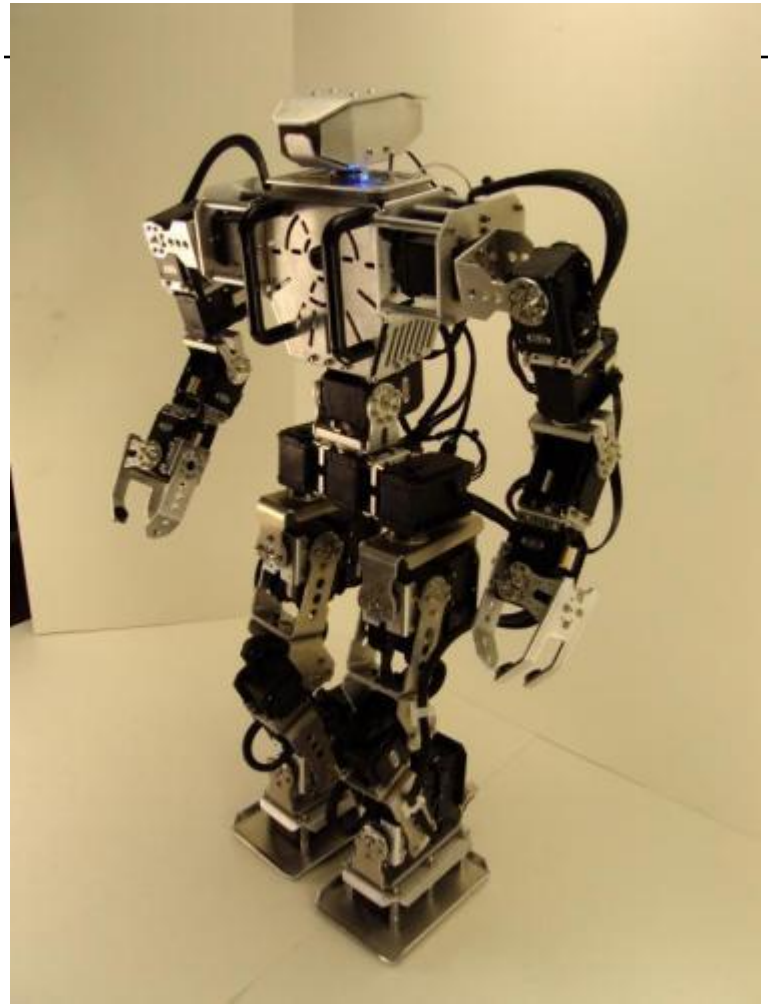
ตัวอย่าง

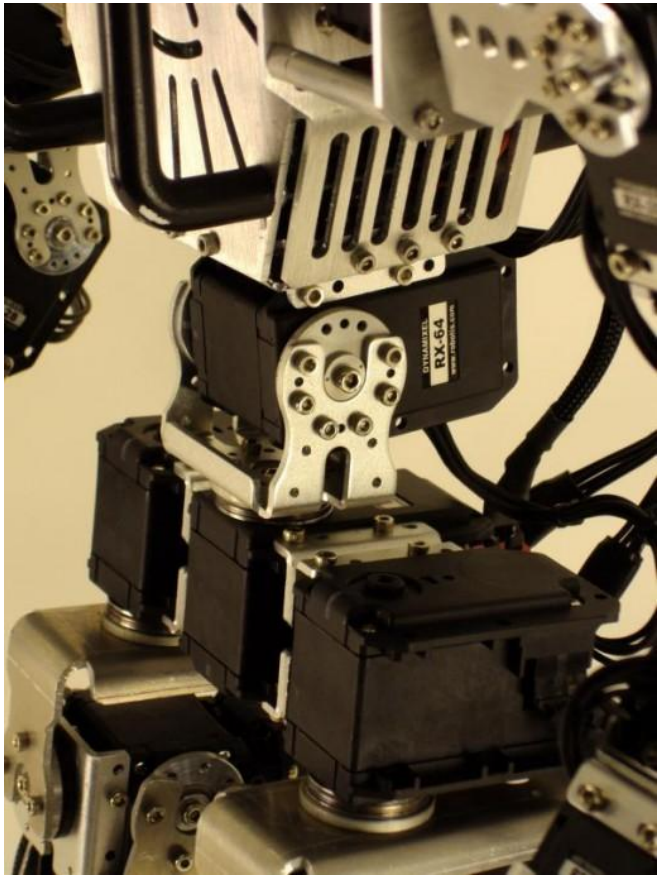
```
#include <Servo.h>
Servo myservo;
void setup()
{
  myservo.attach(9);
  myservo.write(90);  // set servo to mid-point
}

void loop() {}
```

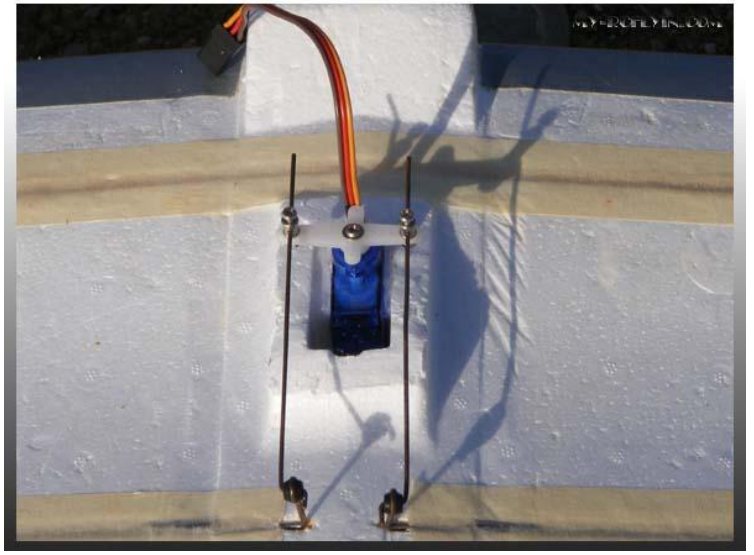
การทดลอง และงานมอบหมาย

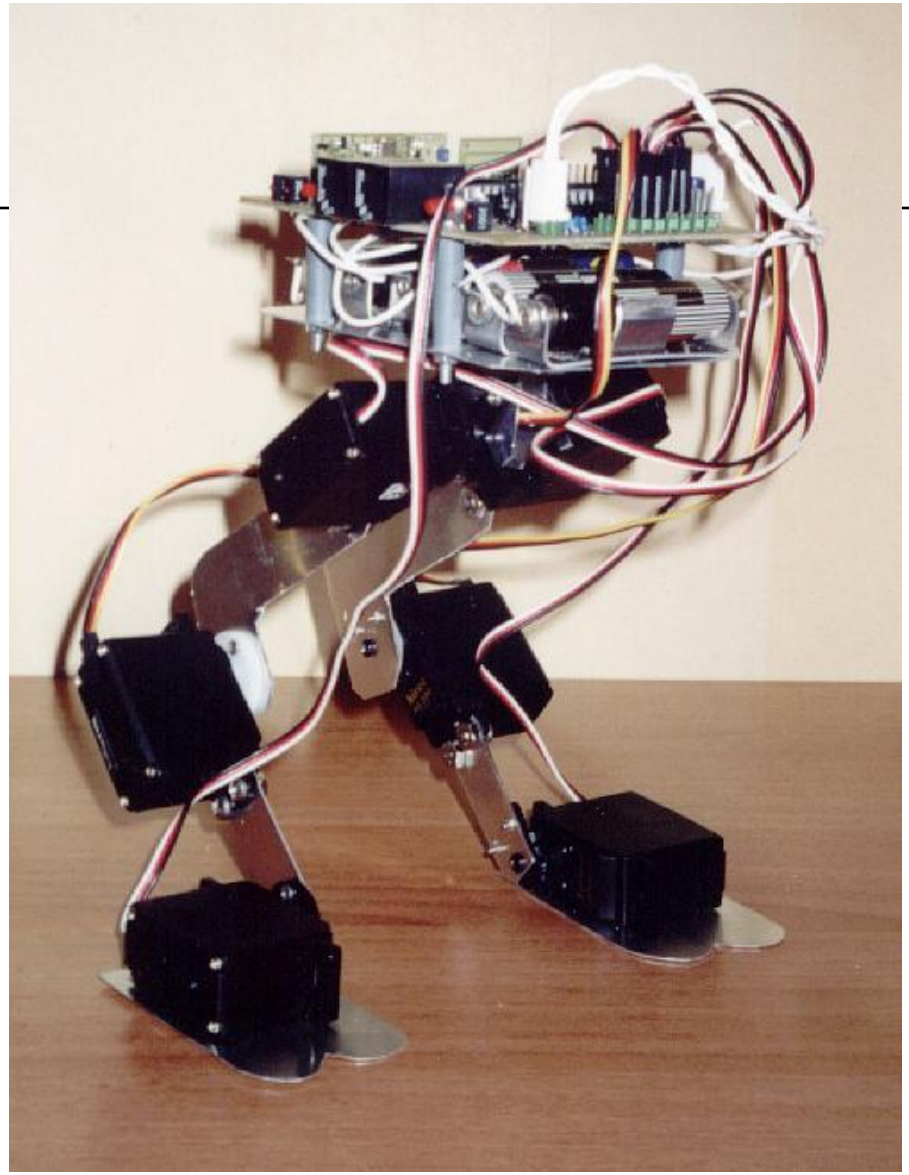
- ให้เขียนโปรแกรมสั่งให้มอเตอร์หมุนจาก 0 องศา ไปถึง 180 องศา โดยหมุนให้หมุนไปที่ละ 5 องศา แล้วหน่วงเวลาสเต็ปละ 200 ms

- ให้ให้เขียนโปรแกรมควบคุมการทำงานเซอร์โวมอเตอร์โดยใช้โปรแกรมบน Arduino และสั่งงานผ่านทาง Processing




RX-64





การทดลองและงานมอบหมาย

การทดลอง

1. ให้เขียนโปรแกรมควบคุมควบคุมความเร็วของ DC motor โดยใช้โปรแกรมที่ 1 หรือ โปรแกรมที่ 2
2. ให้เขียนโปรแกรมควบคุม ตำแหน่งของ Servo motor โดยใช้โปรแกรมที่ 2
3. ให้เขียนโปรแกรมควบคุมตำแหน่งของServo motor โดยค่าตำแหน่งส่งจากเครื่องพีซี

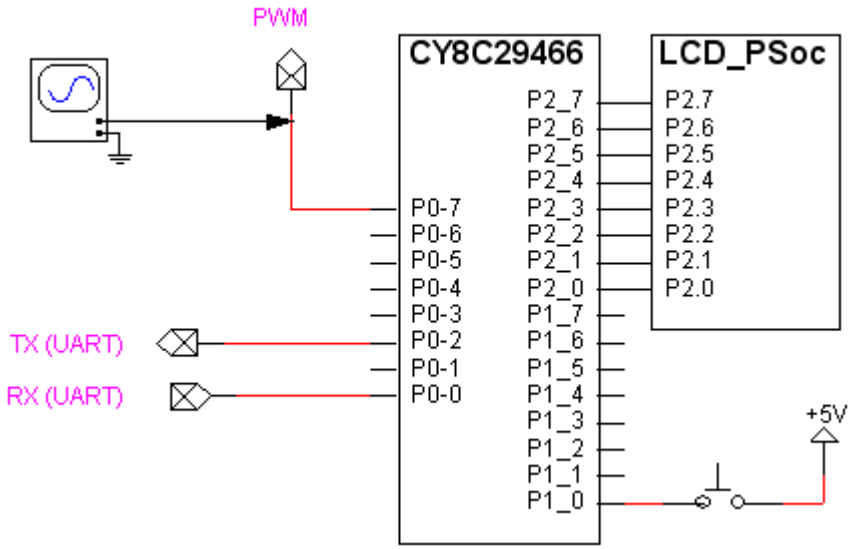
การกำหนดพารามิเตอร์ Global Resource

VC1 = SysClk/8
VC2 = VC1/3
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
ดังนั้น 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

การกำหนดพารามิเตอร์ PWM และพอร์ท

PWM16_1_WritePeriod(2000);
ดังนั้นได้ค่าคาบเวลาเท่ากับ ?



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

LCD_1

User Module Parameters	Value
LCDPort	Port_2
BarGraph	Enable

Name	Port	Select	Drive	Interrupt
Port_0_7	P0[7]	StdCPU	High Z Analog	DisableInt
Port_1_0	P1[0]	StdCPU	Pull Down	DisableInt
Port_1_1	P1[1]	StdCPU	High Z Analog	DisableInt

API ของ PWM16

- PWM16_Start
- PWM16_Stop
- PWM16_WritePeriod
- PWM16_WritePulseWidth

PWM16_Start

ฟังก์ชั่น PWM16_Start

รายละเอียด

Starts the PWM16 User Module. If the enable input is high, the counter will begin to down count.

การใช้งานด้วยภาษา C

```
void PWM16_Start(void);
```

พารามิเตอร์

ไม่มี

ค่าส่งกลับ

ไม่มี

PWM16_Stop

ฟังก์ชั่น PWM16_Stop

รายละเอียด

Stops the counter operation.

การใช้งานด้วยภาษา C

```
void PWM16_Stop(void);
```

พารามิเตอร์

ไม่มี

ค่าส่งกลับ

ไม่มี

PWM16_WritePulseWidth

ฟังก์ชั่น PWM16_WritePulseWidth

รายละเอียด

Writes the PulseWidth register with the pulse width value.

การใช้งานด้วยภาษา C

```
void PWM16_WritePulseWidth(WORD wPeriod);
```

พารามิเตอร์

wPulseWidth: wPulseWidth value is the value from 0 to the period value. MSB is passed in the X register and LSB is passed in the Accumulator.

ค่าส่งกลับ

ไม่มี

PWM16_WritePeriod

ฟังก์ชั่น PWM16_WritePeriod

รายละเอียด

Writes the Period register with the period value. The period value will be transferred from the Period register to the Counter register immediately, if the PWM16 is stopped or when the counter reaches the zero count.

การใช้งานด้วยภาษา C

```
void PWM16_WritePeriod(WORD wPeriod);
```

พารามิเตอร์

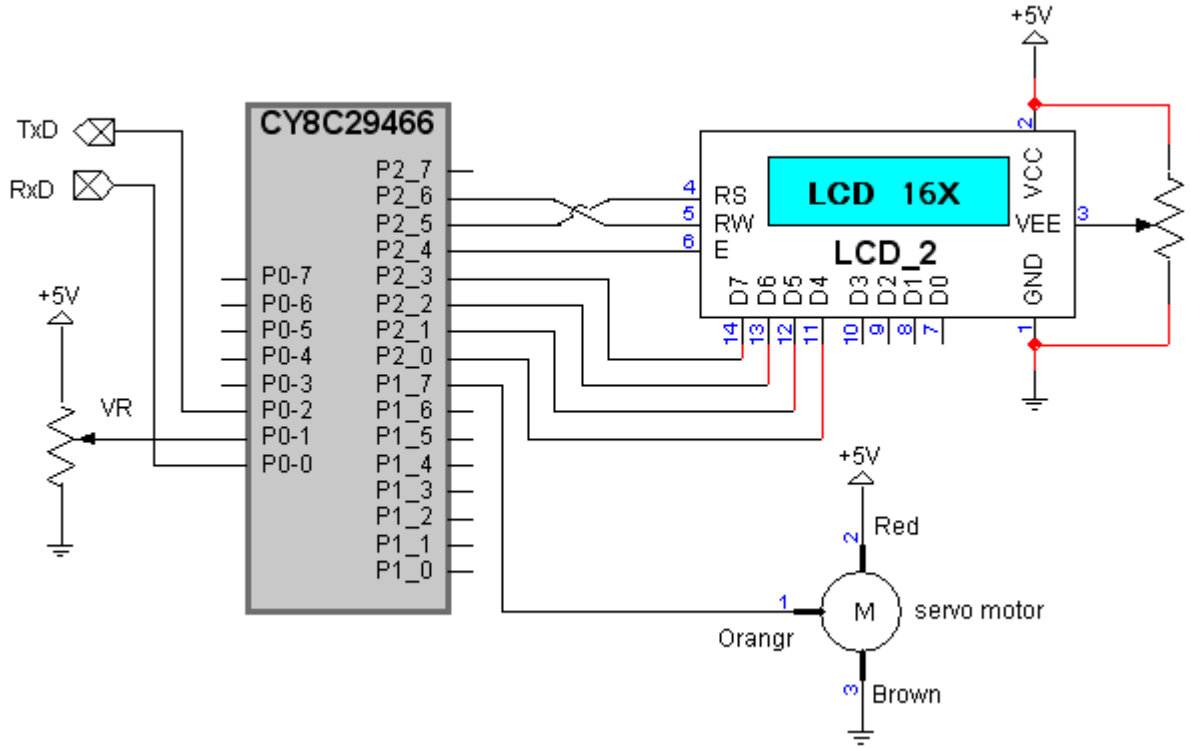
wPeriod: wPeriod value is a value from 0 to 216-1. MSB is passed in the X register and LSB is passed in the Accumulator.

ค่าส่งกลับ

ไม่มี

จบแล้วครับ

วงจรทดลอง


+5V
TxD
RxD
CY8C29466
P2_7
P2_6
P2_5
P2_4
P2_3
P2_2
P2_1
P2_0
P1_7
P1_6
P1_5
P1_4
P1_3
P1_2
P1_1
P1_0
P0-7
P0-6
P0-5
P0-4
P0-3
P0-2
P0-1
P0-0
RS
RW
E
LCD 16X
LCD_2
VCC
VEE
GND
D7
D6
D5
D4
D3
D2
D1
D0
+5V
VR
+5V
Red
M
servo motor
Orangr
Brown