BMS 302 MİKROKONTROLCÜLER

MOTOR UYGULAMALARI 2

Prof.Dr. Mutlu AVCI

Çukurova Üniversitesi Biyomedikal Mühendisliği Bölümü

Bahar 2020

Dört çeşit motor ile hayatımızın pek çok aşamasında karşılaşıyoruz, bunlar:

- DC motor (İşlendi)
- Adım (stepper) motor (İşlendi)
- Servo motor
- Fırçasız (brushless) DC motor

CCP PWM Modu Hatırlatma

- CCP birimi PWM modu, istenen CCPX ucundan istenen görev çevirimine (duty cycle – doluluk oranı) sahip PWM sinyali elde etmek için kullanılır.
- PWM birimi Timer2 zamanlayıcısını kullanır. PIC16F877A'da iki adet CCP modülü olduğundan iki adet de PWM çıkış ucu vardır.
- Bu uçlar RC1/CCP2 ve RC2/CCP1'dir.
- PWM sinyali; denetleyici osilatör frekansı, Timer2 zamanlayıcı biriminin PR2 değeri ve bölme oranı değeri ile belirlenir.

- Timer2 fonksiyonunda bulunan *postscaler* değerinin PWM sinyaline bir etkisi yoktur. PWM sinyalinin periyodu ve frekansı ;

$$T_{PWM} = T_{komut} \times (Timer2\_Bölme\_Oranı) \times (PR2\ değeri + 1)$$

$$T_{komut}(sn) = 1/f_{komut}$$

$$f_{komut} = Kontrolcü\ Osilatör\ Frekansı\ /\ 4 = f_{osc}/4$$

$$f_{PWM} = 1/\ T_{PWM}$$

şeklinde hesaplanır

- Örneğin ;

#use delay (clock=4000000)

setup_timer_2(T2_DIV_BY_1, PR2+1, 1);

komutları kullanılarak bir PWM sinyalinin frekansı 10 kHz e ayarlanmalıdır.

$f_{komut} = f_{osc}/4 = 1\ Mhz$

$TPWM = 1/f_{komut}$ ***x (PR2+1) x (Timer2 bölme oranı) = 1/10k***

(1/(1 MHz)) x (PR2+1) x 1 = 0.1 ms

PR2=99

şeklinde hesaplanır.

SET_PWMX_DUTY( ) Fonksiyonu:

- PWM modunda oluşturulan sinyalin görev çevrim süresini (duty cycle) belirlemeye yarayan fonksiyondur. Kullanımı ;

set_pwmx_duty(değer);

SETUP_CCPX ( ) Fonksiyonu

- Bu fonksiyon ile mikro kontrolcü içinde bulunan CCP modülü ile ilgili ayarlamalar yapılır. Kullanımı ;

setup_ccp1 (mod) ;

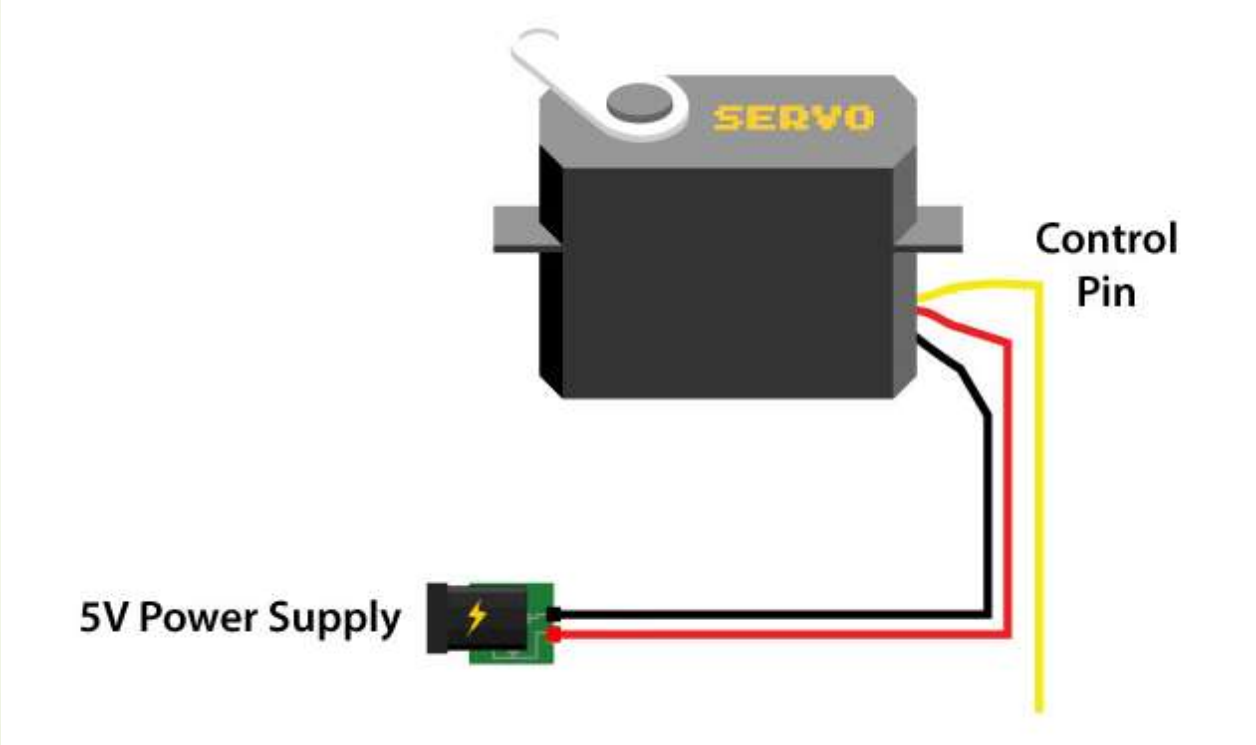
setup_ccp2(mod);

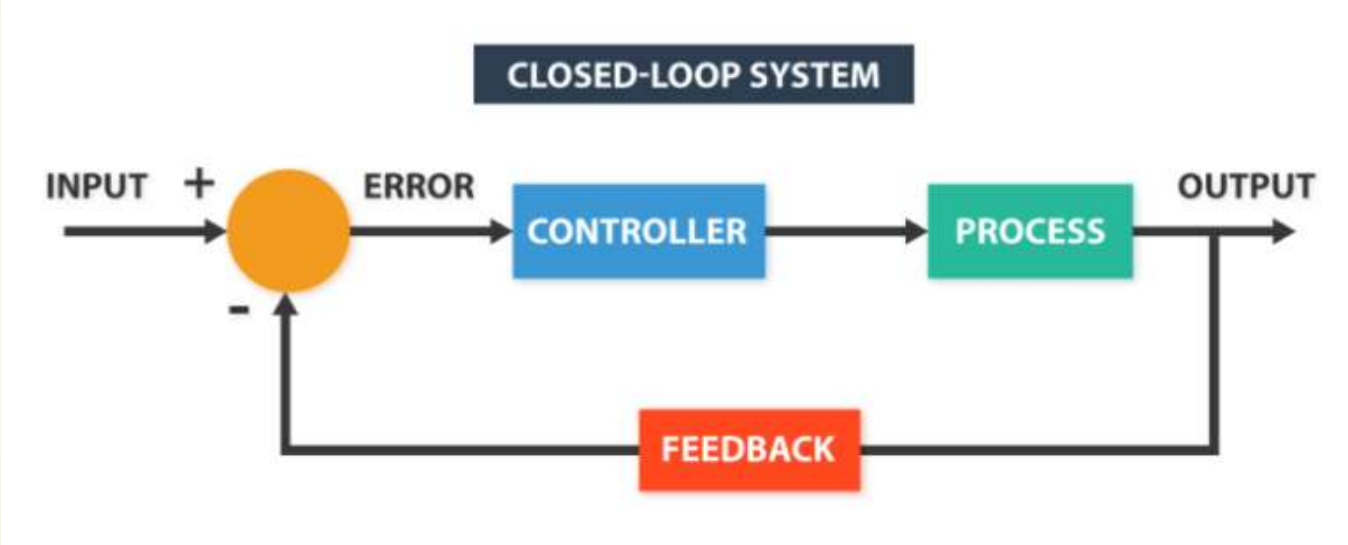
şeklindedir.

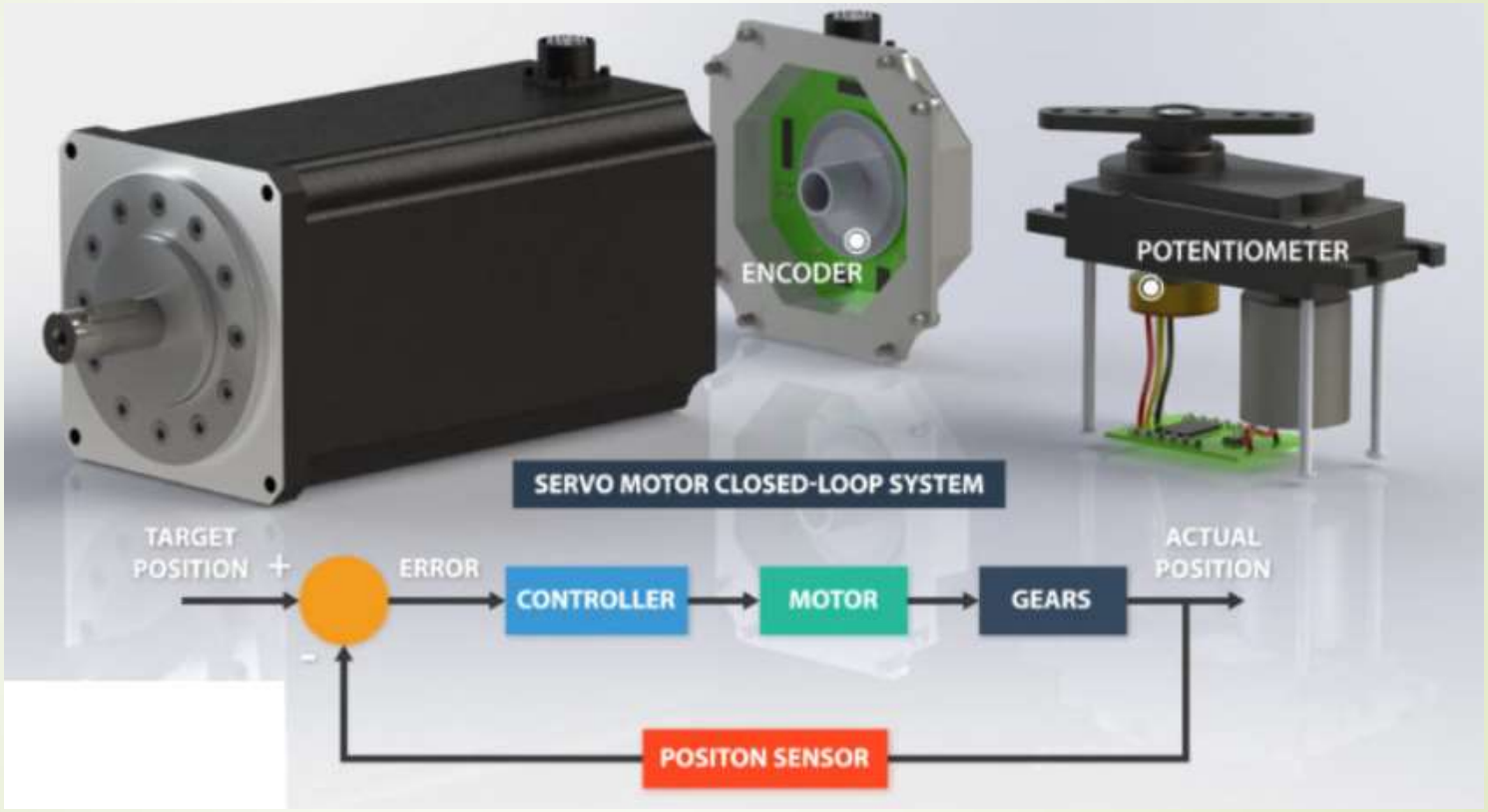
"mod" kısmına kullanılmak istenen moda göre sabit tanımlamalar yazılır.

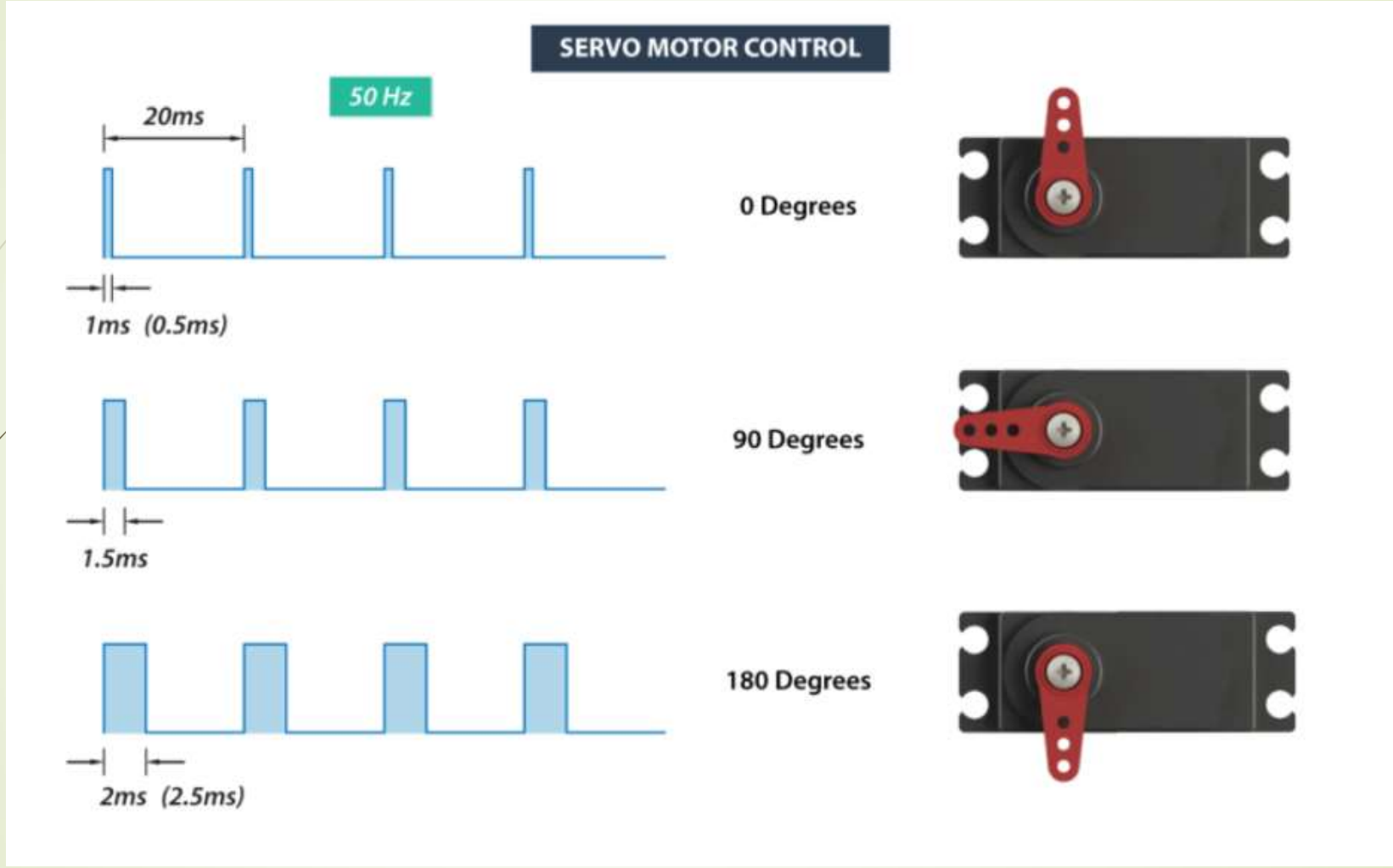
- PWM modu için mod kısmına, ***CCP_PWM*** yazılır.
- CCP birimini devre dışı yapmak için ***CCP_OFF*** yazılır.

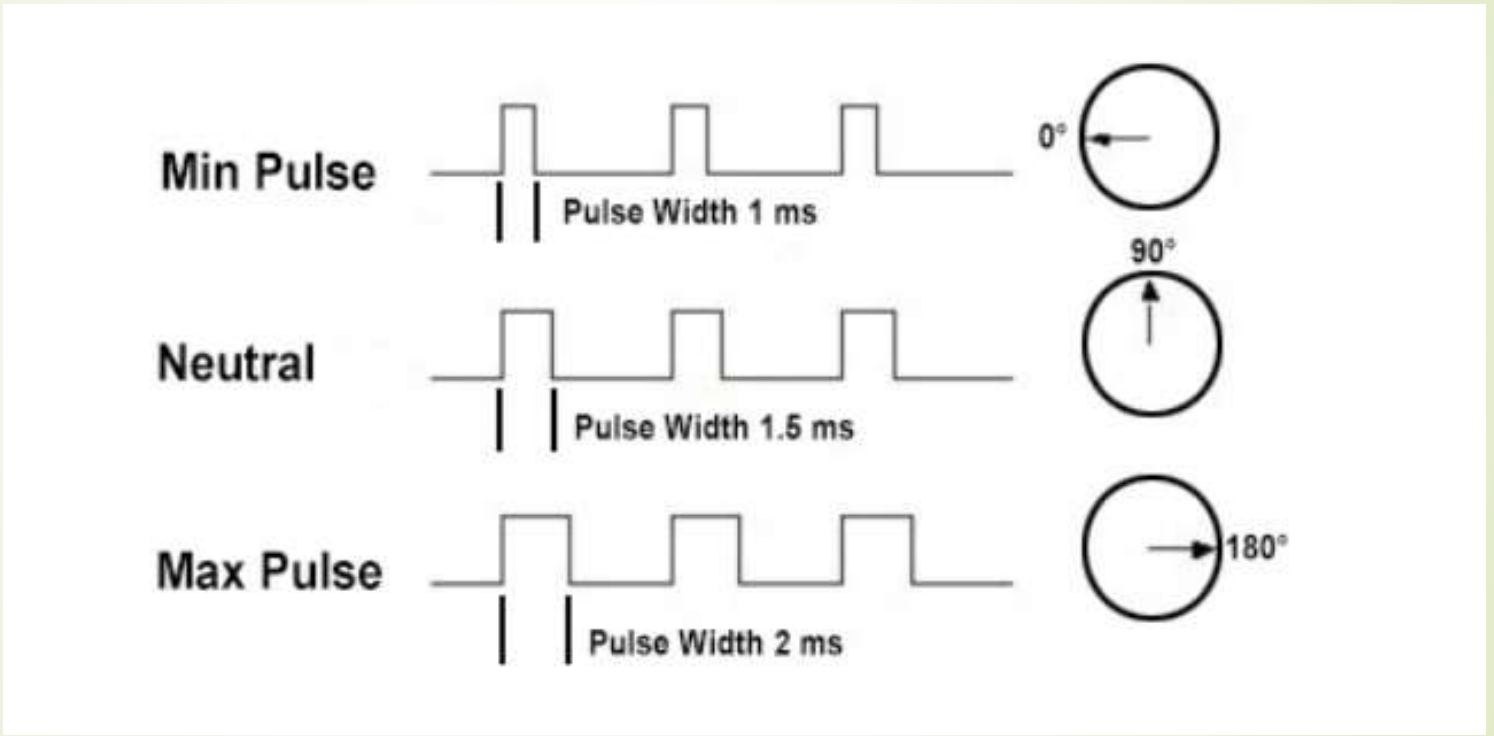
Servo Motorlar


SERVO
Control
Pin
5V Power Supply


CLOSED-LOOP SYSTEM
INPUT
+
ERROR
CONTROLLER
PROCESS
OUTPUT
-
FEEDBACK


ENCODER
POTENTIOMETER
SERVO MOTOR CLOSED-LOOP SYSTEM
TARGET POSITION
+
-
ERROR
CONTROLLER
MOTOR
GEARS
ACTUAL POSITION
POSITON SENSOR


SERVO MOTOR CONTROL
50 Hz
20ms
1ms (0.5ms)
0 Degrees
1.5ms
90 Degrees
2ms (2.5ms)
180 Degrees


Min Pulse
Pulse Width 1 ms
0°
90°
Neutral
Pulse Width 1.5 ms
Max Pulse
Pulse Width 2 ms
180°

Örnek: PIC 16F877A mikrokontrolcü ile bir servo motorun sola 60 derece ve sağa 60 derece devamlı dönmesi istenmektedir. Bu işlemi geliştiren donanımı tasarlayıp, CCS C programını yazın.

- Servo motorlar 50 Hz frekansla çalışırlar, bu frekansla PWM üretebilmek içim PIC daha düşük kristal osilatör ile çalıştırılır.

T_{PWM} = 1/50 = 20 ms

T_{PWM} = T_{komut} x (Timer2_Bölme_Oranı)x(PR2 değeri +1)

Diyelimki PR2 değeri 199, bölme oranı da 1 olsun.

20m = T_{komut} * 200 ise T_{komut} = 1/10 ms olur.

f_{komut} = 1/T_{komut} = 10 kHz

f_{komut} = Kontrolcü Osilatör Frekansı / 4 = f_{osc}/4

f_{osc} = 40 kHz

O halde

1ms darbe genişliği için duty cycle =10

1.5ms darbe genişliği için duty cycle = 15

2ms darbe genişliği için duty cycle = 20 olmalıdır.

1.5ms tam orta nokta olsun;

1ms ile 1.5ms arası 90 a bölünürse sola 60 dereceye karşılık değer:

1.5m-(0.5*60/90) den = 1,16666ms bulunur.

Bu da duty cycle = 12 demektir.

60 derece sağ için 1.5m + 0.5*(60/90)m = 1.83333 ms

Duty cycle = 18 yapar.

```
#include <prog.h>
#use delay (clock=40000)
#use fast_io(C)

int8 i=0;
void main(){
set_tris_c(0x00);
setup_ccp1(CCP_PWM);
setup_timer_2(T2_DIV_BY_1, 199, 1);
set_pwm1_duty(15);
}
```

```
While(TRUE){
set_pwm1_duty(12);
Set_pwm1_duty(18);
}
}
```

- Bu durum düşük frekanslı kristal bulmayı zorunlu tuttu,
- PIC'i tüm işlemlerde yavaşlattı,
- 1'er derecelik açılarla taramak istesek PWM ile bunu yapamayız.
- Her halukarda sonsuz döngü içinde pwm ayarlayarak motor dönmesini kontrol etmemiz gerekli.

O halde servo motoru PWM çıkışı ile sürmek yerine diğer uçlarla daha hassas sürebilirim. Mesela Timer0 ı kullanalım ve 10 ar derecelik adımlarla sola sağa 60'ar derece döndürelim. Kristalimizde 4 Mhz lik olsun. Servonun kontrol ucunu, B0 ucuna bağlayalım.

- İlk önce servo kontrolü için zamanlama ayarı yapalım:

- 1ms ile 2ms arası yani 1ms'lik bir aralık 180 dereceye bölünmüştür. O halde herbir derece için:

1m/180 = 5.5555us darbe genişliği zamanı değişimi olmalıdır.

10 derece için 55.5555 us olmalıdır.

1.166666ms ile 1.833333ms arasında 55.5555us adımla taranmalıdır.

Timer0 her 55.5555us de kesme oluşturmalıdır, buna en yakın 56us alırsak

- $T_{kesme} = T_{komut} * (Bölme\_oranı) * (256-TMR0)$

Tkomut = 1us Bölme_oranı = 4 ise

256-TMR0= 14 olur. Buradan TMR0 = 242 bulunur.

1.5m / 56 u = 26,79 yani = 27

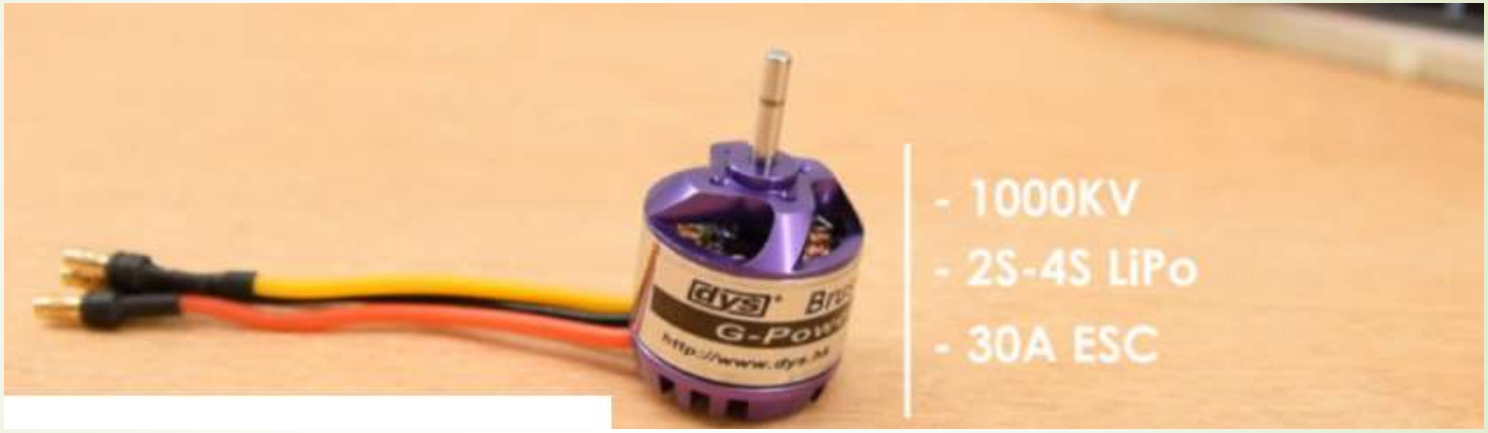
1.16666ms/56u = 20,83 yani = 21 değerini alır.

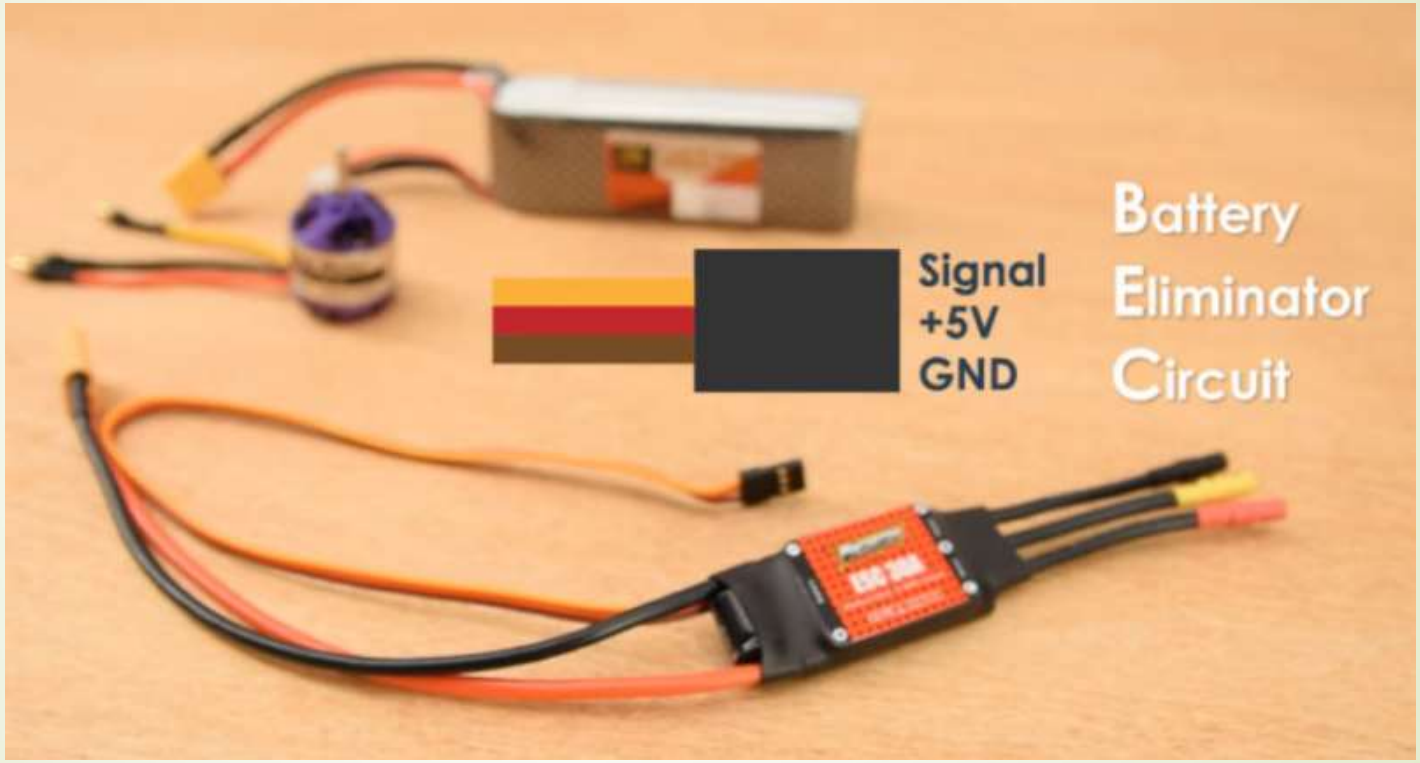
1.833333m/56u = 32.74 yani = 33 değerini alır.

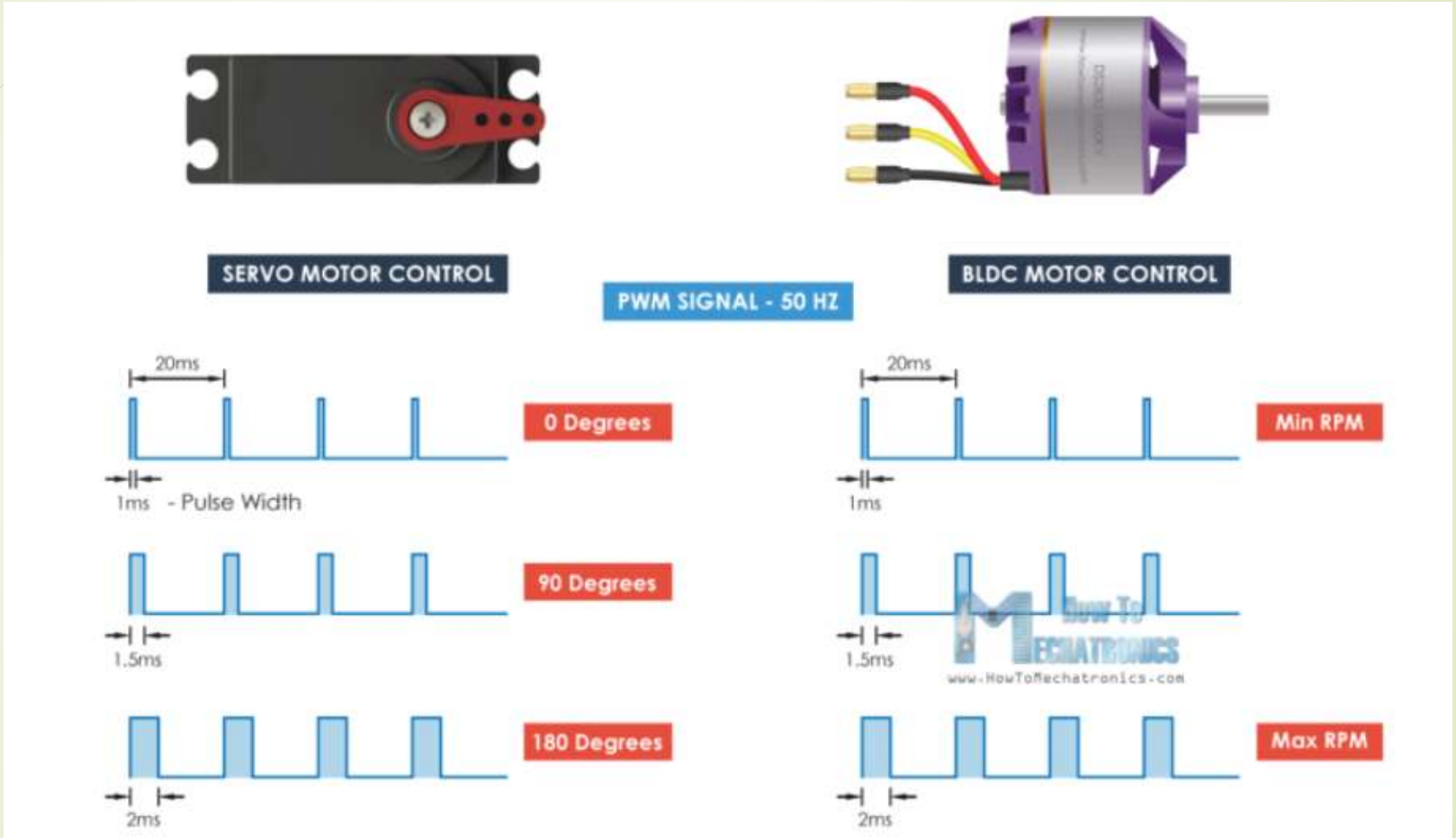
20m / 56u = 357 yapar.

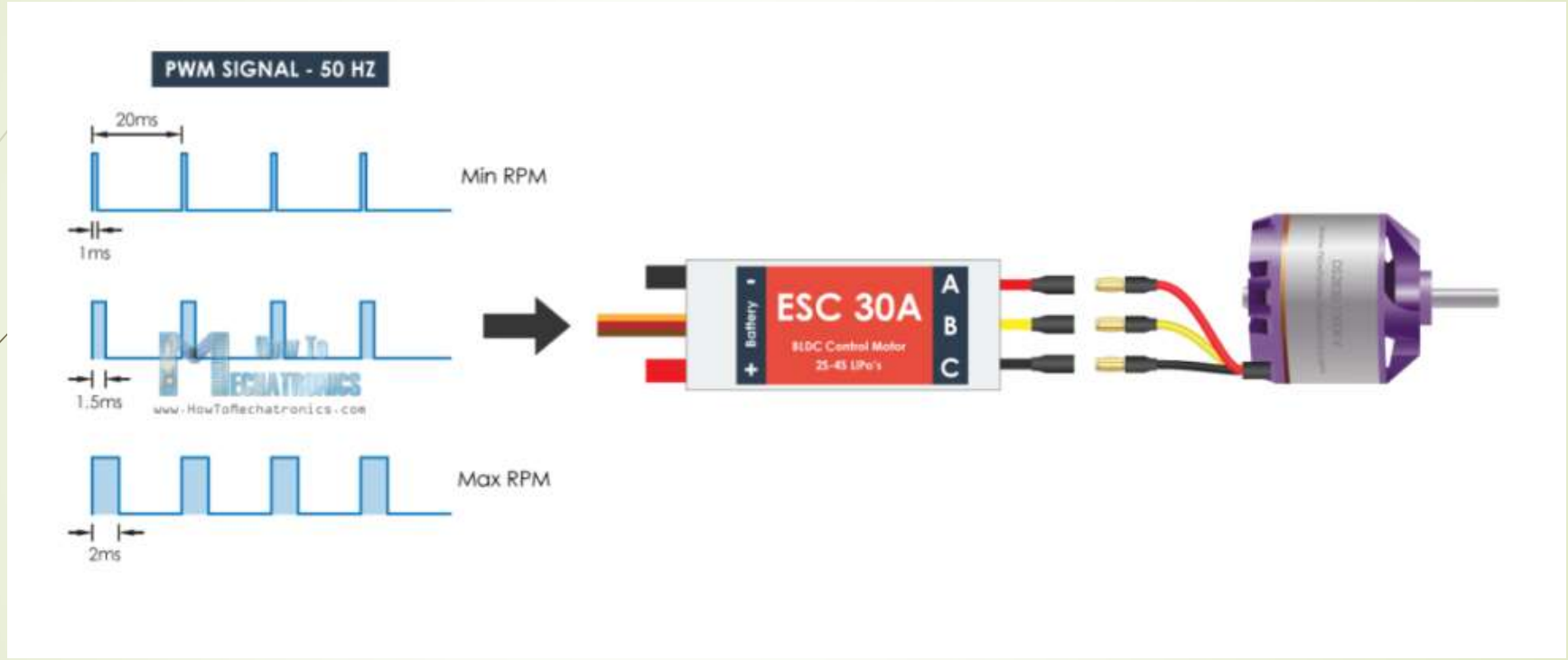
```
#include <prog.h>
#use delay (clock=4000000)
int8 duty=27,pwm=0, k=1;
İnt1 say=1;
int8 period=0;
#int_timer0
void kesme0(){
Set_timer0(242);
if (pwm<=0) output_high(pin_B0);
if(pwm>=duty) output_low(pin_B0);
pwm++;
İf(pwm>=100) pwm=100;
İf (say==1) period++;
say=~say;
İf (period>=129) {
pwm=0;
Period=0;
if (duty<=21) k=1;
if(duty>=33) k=0;
if (k>0) duty=duty+1;
else duty=duty-1;
}
}
void main(){
setup_timer_0(RTCC_INTERNAL|RTCC_DIV_4);
set_timer0(242);
enable_interrupts(INT_TIMER0);
Enable_interrupts(GLOBAL);
While(1);
}
```

Fırçasız Motorlar

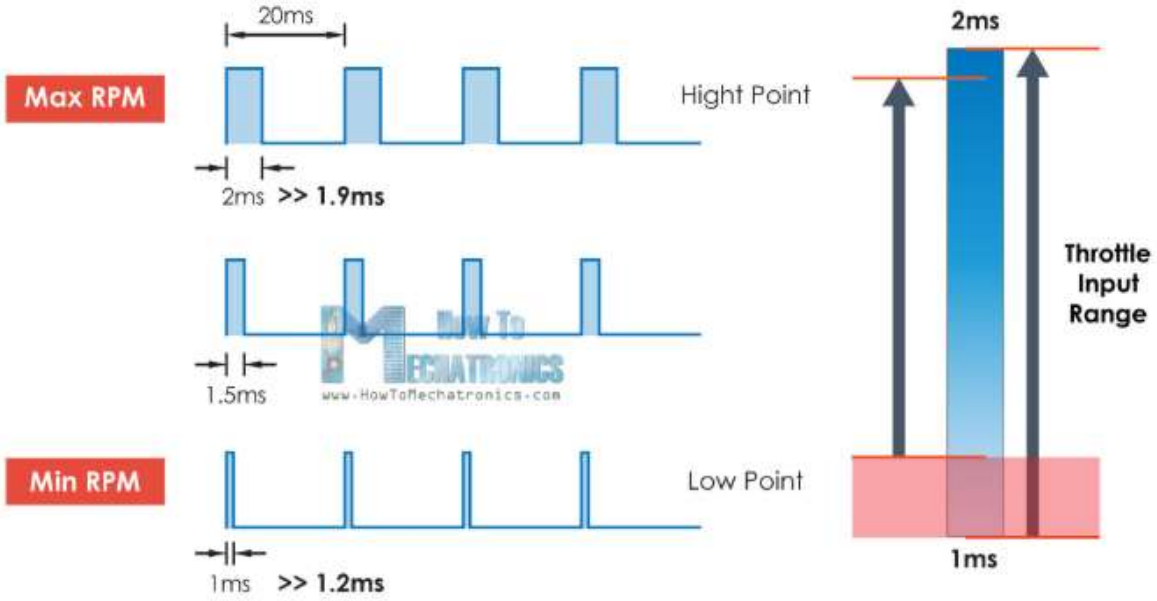



Signal
+5V
GND
Battery
Eliminator
Circuit


SERVO MOTOR CONTROL
BLDC MOTOR CONTROL
PWM SIGNAL - 50 HZ
20ms
0 Degrees
1ms - Pulse Width
90 Degrees
1.5ms
180 Degrees
2ms
20ms
Min RPM
1ms
1.5ms
Max RPM
2ms
www.HowToMechatronics.com


PWM SIGNAL - 50 HZ
20ms
Min RPM
1ms
1.5ms
Max RPM
2ms
ESC 30A
Battery
A
B
C

ESC Calibration



- **Örnek:** PIC16F877A mikrokontrolcüsünün B0 ucuna bağlı bir fırçasız motorun hız kontrolü A0 ve A1 ucuna bağlı iki push button kullanılarak yapılmak istenmektedir. Hız artış ve azalışı toplam 10 kademe olarak istenmektedir.

  Bu işlemi yapan CCS C program kodunu yazın.

Öncelikle analize başlayalım:

1ms aralık (1ms ile 2ms arası) 10 parçaya bölünürse;

1adım = 1ms/10 = 0.1ms olur.

Değerleri belirleyelim:

Timer0 her 0.1ms de yani 100 us de kesme oluşturmalıdır.

4 MHZ lik Xtall, Tkomut=1us, bölen= 4

256-TMR0 = 25 TMR0= 231

1ms /100u = 10 1.5m/100u = 15 2m/100u = 20

20ms/0.1m = 200

```
#include <prog.h>
#use delay (clock=4000000)
int8 duty=15, pwm=0;
int8 period=0;
#int_timer0
void kesme0(){
set_timer0(231);
if (pwm<=0) output_high(pin_B0);
if(pwm>=duty) output_low(pin_B0);
pwm++;
İf(pwm>=100) pwm=100;
İf (period>=200) {
pwm=0; Period=0;}
}
```

```
void main(){
setup_timer_0(RTCC_INTERNAL | RTCC_DIV_4);
set_timer0(231);
enable_interrupts(INT_TIMER0);
Enable_interrupts(GLOBAL);
While(1){
if(input(pin_A0){
while(pin_A0); duty++; }
if (duty>=20) duty=20;
}
if(input(pin_A1){
while(pin_A1); duty--; }
if (duty<=10) duty=10;
} }
```