BİLGİSAYAR DESTEKLİ SAYISAL KONTROL

1. Deneyin Amacı

Mikrodenetleyici ile gerçek zamanlı açık-kapalı ve oransal integral türev (OİT) kontrol yapmak ve ölçüm sonuçlarını anlık olarak bilgisayar ekranında görselleştirmek.

2. Deneyin Hedefleri

Bu deneyin başlıca hedefleri şunlardır:

1. Mikrodenetleyici ile gerçek zamanlı açık-kapalı sayısal kontrolün öğrenilmesi.
2. Mikrodenetleyici ile gerçek zamanlı OİT sayısal kontrolün öğrenilmesi.
3. Mikrodenetleyici ile bilgisayar arasında evrensel seri veriyolu (ESV) üzerinde haberleşmenin öğrenilmesi.
4. Sistemden alınan ölçüm sonuçlarının anlık olarak bilgisayar ekranında görselleştirilmesinin öğrenilmesi.
5. Sayısal kontrol yöntemleri kullanarak kapalı çevrim motor hız kontrolünün öğrenilmesi.

3. Hazırlık Soruları

Deney öncesinde aşağıdaki soruları cevaplayıp, hazırlık raporunuzda sununuz.

1. Açık çevrim kontrol nedir? Kısaca açıklayınız.
2. Kapalı çevrim kontrol nedir? Kısaca açıklayınız.
3. Açık kapalı kontrol nedir? Kısaca açıklayınız.
4. Açık kapalı kontrolün artıları ve eksileri nelerdir? Kısaca açıklayınız.
5. OİT kontrol nedir? Kısaca açıklayınız.
6. OİT kontrolün artıları ve eksileri nelerdir? Kısaca açıklayınız.

4. Kullanılacak Malzemeler

Bu deneyde aşağıdaki araç-gereç ve malzemeler kullanılacaktır.

1. Bilgisayar
2. Motor hız kontrol deney seti
3. ESV bağlantı kablosu

5. Genel Bilgileri

5.1. Motor Hız Kontrol Deney Seti

Motor hız kontrol deney seti aşağıdaki birimlerden oluşmaktadır:

1. Mikrodenetleyici kartı
2. Motor sürücü kartı
3. Kızılötesi devir algılayıcı modülü
4. Fırçasız doğru akım motoru
5. Delikli disk

Bu modülleri kısaca tanıyalım.

Mikrodenetleyici kartı: 16 MHz'de çalışan 8 bitlik bir işlemciye, 14 sayısal giriş-çıkış, 6 analog giriş ve 6 darbe genişlik modülasyonu çıkışına sahip bir denetleyici kartıdır. Bilgisayar üzerinde ESV üzerinden programlanmakta ve bu veriyolu üzerinden bilgisayara veri gönderip, alabilmektedir.

Motor sürücü kartı: 4,8 V – 46 V besleme gerilimi ve en fazla 2 A çalışma akımına sahip iki doğru akım motorunun hız ve yön kontrolünü sağlayabilen bir sürücü kartıdır.

Kızılötesi devir algılayıcı modülü: Motor hızını ölçmek için kullanılan sayısal çıkışlı bir algılayıcı modülüdür. Dahili kızılötesi vericisi ile alıcı arasında görüş varken lojik 1 ve görüş yokken lojik 0 çıkışı üretmektedir.

Fırçasız doğru akım motoru: 12 V çalışma gerilimi 0,4 A çalışma akımına sahip bir fandır.

Delikli disk: Motor devrini ölçmek için kullanılmaktadır. Boyutları Şekil 1'de verilmiştir. Bu disk ile motor hızı şu şekilde ölçülmektedir:

1. Kızılötesi devir algılayıcı modülü algılayıcıları arasında görüş olmadığı zaman lojik 0 üretmektedir. Mikrodenetleyici bu süreyi µs cinsinde ölçmekte ve motor hızını hesaplamaktadır. Lojik 0 süresini bulmak için öncelikle diskin delik olmayan yay

uzunluğunu bulmak gerekmektedir. Bu uzunluk (1) ile hesaplanmaktadır.

$$l = \frac{\theta}{360} 2\pi r \tag{1}$$

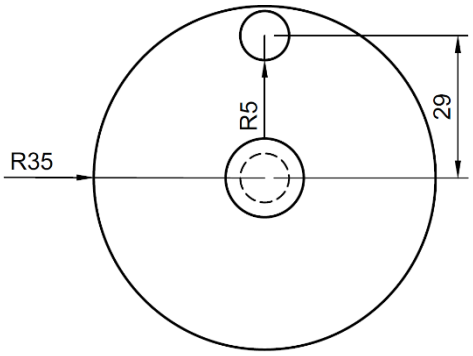
Burada l delik olmayan yay uzunluğu (mm), θ delik olmayan yay açısı (°) ve r yay yarı çapıdır (mm). Bu durumda yay uzunluğu 172,161 mm bulunur.

$$l = \frac{\theta}{360} 2\pi r = \frac{340,2}{360} 2\pi 29 = 172,161\ mm$$

2. Motor hızı devir/dakika (rpm) cinsinden (2) eşitliği ile hesaplanır.

$$v = \frac{60 \times 172.161}{t_l} \tag{2}$$

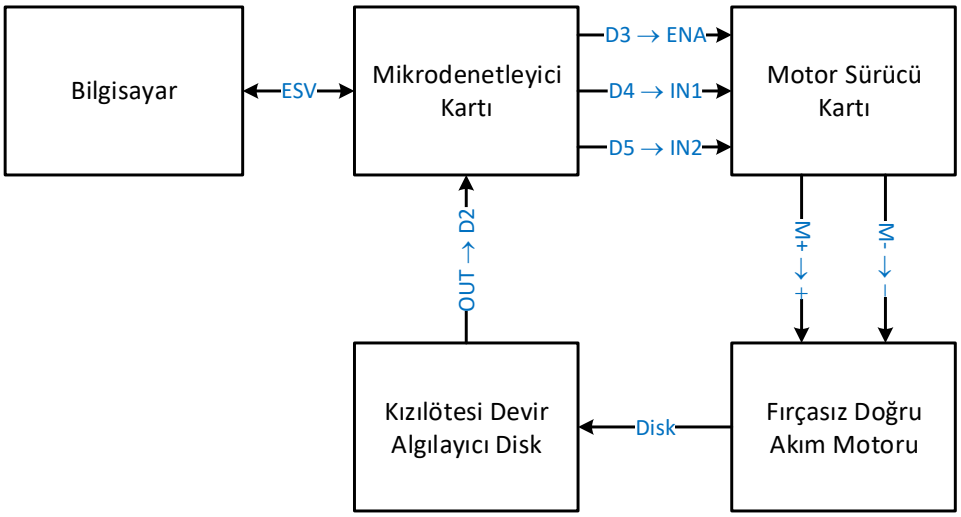
Burada v motor hızı (rpm) ve t_l devir algılayıcı çıkışının lojik 0 olma süresidir (µs).



Şekil 1. Delikli disk

5.2 Motor Hız Kontrol Deney Seti Bağlantı Şeması

Kontrol yazılımı Şekil 2'de verilen bağlantı şemasına göre yazılmalıdır.



Şekil 2. Deney seti bağlantı şeması

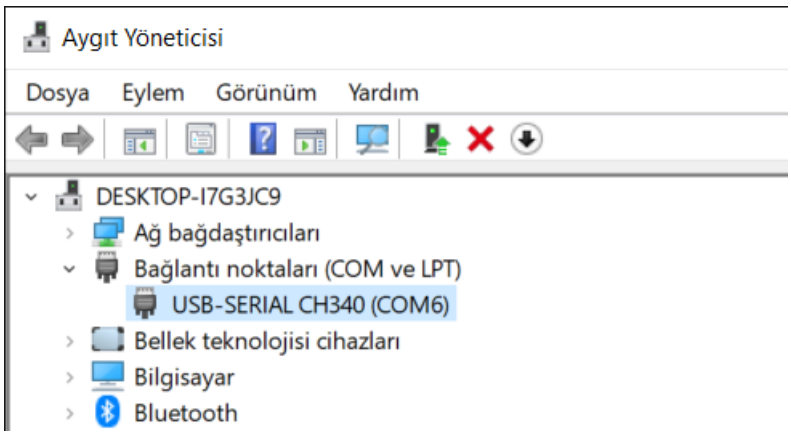
6. Uygulamalar

Bu deneyde mikrodenetleyici ile açık-kapalı ve OİT kontrol yapılacak ve ölçüm sonuçlarını ESV üzerinden bilgisayara aktarılacaktır.

6.1. Uygulama Bağlantıları

Uygulama bağlantılarını şu şekilde gerçekleştiriniz.

1. Bilgisayarı açınız.
2. Deney setinin topraklı fişini, deney masasındaki topraklı prize takınız.
3. ESV kablosunu deney setine ve bilgisayara takınız.
4. Masaüstündeki *Bilgisayarım* simgesine sağ tıklayıp *Özellikler* sekmesini tıklayınız. Ardında açılan pencerede *Aygıt Yöneticisine* tıklayınız. Son olarak açılan yeni pencerede Şekil 3'de görüldüğü gibi *Bağlantı noktaları (COM ve LPT)* seçeneğine tıklayarak, mikrodenetleyicinin bağlı olduğu *COM* portunu bulunuz.



Şekil 3. Bağlantı noktası

6.2. Uygulama Yazılımları

Bu deneyde bir fırçasız doğru akım motorunu hız denetimi hem açık-kapalı ve hem de OİT kontrol yöntemi kullanılarak yapılacak ve ölçüm sonuçları bilgisayarda görselleştirilecektir. Mikrodenetleyicide koşacak olan uygulama yazılımları *Arduino* editöründe ve bilgisayardaki görsel program ise *MATLAB* editöründe hazırlanacaktır.

Arduino editöründe yazılan *açık-kapalı hız kontrolü programı* Şekil 4'de verilmiştir.

```
1  // Tanımlamalar -------------------------------------------------------------
2  #define DEVIR 2                 // Devir algılayıcı girişi
3  #define HIZ   3                 // Motor PWM çıkışı
4  #define YON1  4                 // Motor yön çıkışı
5  #define YON2  5                 // Motor yön çıkışı
6
7  // Değişkenler --------------------------------------------------------------
8  int i     = 0;                  // Sayaç
9  int t0    = 0;                  // t0 zamanı (ms)
10 int t1    = 0;                  // t1 zamanı (ms)
11 int To    = 25;                 // Örnekleme priyodu (ms)
12 int Hiz   = 0;                  // Motor hızı (rpm)
13 int Ref   = 0;                  // Referans hız (rpm)
14 int Esik  = 25;                 // Eşik değeri (rpm)
15
16 // Ayarlar -----------------------------------------------------------------
17 void setup()
18 {
19   pinMode(DEVIR, INPUT);        // Giriş olarak ayarla
20   pinMode(HIZ,  OUTPUT);        // Çıkış olarak ayarla
21   pinMode(YON1, OUTPUT);        // Çıkış olarak ayarla
22   pinMode(YON2, OUTPUT);        // Çıkış olarak ayarla
23
24   digitalWrite(HIZ,  LOW);      // Motoru durdur
25   digitalWrite(YON1, HIGH);     // Dönüş yönünü saat yönü olarak belirle
26   digitalWrite(YON2, LOW);      // Dönüş yönünü saat yönü olarak belirle
27
28   Serial.begin(9600);           // 9600 bps veri hızında seri portu aç
29 }
30
31 // Ana döngü ----------------------------------------------------------------
32 void loop()
33 {
34   for (i = 0; i <= 200; i++)               // Ana kontrol döngüsü
35   {
36     t0 = millis();                         // t0 zamanını oku
37     Serial.println(Hiz);                   // Hız bilgisini yolla
38
39     if (Hiz < Ref - Esik) digitalWrite(HIZ, HIGH);      // Açık
40     if (Hiz > Ref + Esik) digitalWrite(HIZ, LOW);       // Kapalı
41
42     Hiz = HizOlc();                        // Motor hızını ölç
43
44     while ((t1 - t0) < To) t1 = millis();  // Örnekleme zamanı doldu mu?
45   }
46
47   digitalWrite(HIZ, LOW);                  // Motoru durudur
48
49   while(1) {};
50 }
51
52 // Motor hızını ölç ---------------------------------------------------------
53 int HizOlc(void)
54 {
55   int Hiz = 0;                         // Hız (rpm)
56   unsigned long Sure = 0;              // Algılayıcının 0 olma süresi (µs)
57
58   Sure = pulseInLong(DEVIR, LOW);      // 1 devir süresi (µs)
59   Hiz  = (60*172191) / Sure;           // Hız (rpm)
```

```
60
61    return(Hiz);                          // Hız bilgisini geri döndür
62  }
```

Şekil 4. Açık-kapalı hız kontrolü mikrodenetleyici yazılımı

Arduino editöründe yazılan *OİT hız kontrolü programı* Şekil 5'de verilmiştir.

```
 1  // Tanımlamalar ---------------------------------------------------------
 2  #define DEVIR 2                 // Devir algılayıcı girişi
 3  #define HIZ   3                 // Motor PWM çıkışı
 4  #define YON1  4                 // Motor yön çıkışı
 5  #define YON2  5                 // Motor yön çıkışı
 6
 7  // Değişkenler ----------------------------------------------------------
 8  int i     = 0;                  // Sayaç
 9  int t0    = 0;                  // t0 zamanı (ms)
10  int t1    = 0;                  // t1 zamanı (ms)
11  int To    = 25;                 // Örnekleme priyodu (ms)
12  int Hiz   = 0;                  // Motor hızı (rpm)
13  int Ref   = 4000;               // Referans hız (rpm)
14  int Hata  = 0;                  // Hata değeri (rpm)
15  int Hata0 = 0;                  // Önceki hata değeri (rpm)
16  int PID   = 0;                  // OİT eylemi (0 .. 255)
17  float P   = 0;                  // Oransal eylem (0 .. 255)
18  float I   = 0;                  // İntegral eylemi (0 .. 255)
19  float D   = 0;                  // Türev eylemi (0 .. 255)
20  float Kp  = 0;                  // Oransal kazanç
21  float Ki  = 0;                  // İntegral kazancı
22  float Kd  = 0;                  // Türev kazancı
23
24  // Ayarlar --------------------------------------------------------------
25  void setup()
26  {
27    pinMode(DEVIR, INPUT);          // Giriş olarak ayarla
28    pinMode(HIZ,  OUTPUT);          // Çıkış olarak ayarla
29    pinMode(YON1, OUTPUT);          // Çıkış olarak ayarla
30    pinMode(YON2, OUTPUT);          // Çıkış olarak ayarla
31
32    digitalWrite(HIZ,  LOW);        // Motoru durdur
33    digitalWrite(YON1, HIGH);       // Dönüş yönünü saat yönü olarak belirle
34    digitalWrite(YON2, LOW);        // Dönüş yönünü saat yönü olarak belirle
35
36    Serial.begin(9600);             // 9600 bps veri hızında seri portu aç
37  }
38
39  // Ana döngü ------------------------------------------------------------
40  void loop()
41  {
42    for (i = 0; i <= 200; i++)  // Ana kontrol döngüsü
43    {
44      t0 = millis();                // t0 zamanını oku
45      Serial.println(Hiz);          // Hız bilgisini yolla
46      Hata = Ref - Hiz;             // Negatif geri besleme
47
```

```
48      P = Kp * Hata;                  // Oransal eylem
49      if (P > 255) P = 255;           // Oransal eylem için en büyük değer
50      if (P <   0) P = 0;             // Oransal eylem için en küçük değer
51
52      I += Ki * Hata*(To/1000.0);     // İntegral eylemi
53      if (I > 255) I = 255;           // İntegral eylemi için en büyük değer
54      if (I <   0) I = 0;             // İntegral eylemi için en küçük değer
55
56      D = Kd * (Hata-Hata0)*(1000/To);    // Türev eylemi
57      if (D > 255) D = 255;           // Türev eylemi için en büyük değer
58      if (D <   0) D = 0;             // Türev eylemi için en küçük değer
59
60      PID = P + I + D;                // OİT eylemi
61      if (PID > 255) PID = 255;       // OİT eylemi için en büyük değer
62      if (PID <   0) PID = 0;         // OİT eylemi için en küçük değer
63
64      analogWrite(HIZ, PID);          // Motora OİT oranında PWM sinyali uygula
65
66      Hiz = HizOlc();                 // Motor hızını ölç
67
68      Hata0 = Hata;                   // Bir önceki adımdaki hatayı sakla
69
70      while ((t1 - t0) < To) t1 = millis();   // Örnekleme zamanı doldu mu?
71    }
72
73    digitalWrite(HIZ, LOW);           // Motoru durudur
74
75    while(1) {};
76  }
77
78  // Motor hızını ölç ------------------------------------------------------
79  int HizOlc(void)
80  {
81    int Hiz = 0;                            // Hız (rpm)
82    unsigned long Sure = 0;                 // Algılayıcının 0 olma süresi (µs)
83
84    Sure = pulseInLong(DEVIR, LOW);         // 1 devir süresi (µs)
85    Hiz  = (60*172191) / Sure;              // Hız (rpm)
86
87    return(Hiz);                            // Hız bilgisini geri döndür
88  }
```

Şekil 5. OİT hız kontrolü mikrodenetleyici yazılımı

MATLAB editöründe yazılan *görsel program* Şekil 6'da verilmiştir.

```
1  clear all;                               % Tüm değişkenleri sil
2  clc;                                     % Ekranı temizle
3
4  S1 = serial('COM6','BaudRate',9600);     % Seri port tanımla
5  fclose(S1);                              % S1 seri portunu kapat
6  fopen(S1);                               % S1 seri portunu aç
7
8  figure(1);                               % 1 numaralı şekli oluştur
9  hold on;                                 % Çizime devam et
10 grid on;                                 % Izgarayı aç
```

```
11  axis([0 5 0 5000]);                    % Eksen sınırlarını belirle
12  xlabel('zaman (ms)');                  % x ekseninin etiketini yaz
13  ylabel('hız (rpm)');                   % y ekseninin etiketini yaz
14
15  t = 0:0.025:5;                         % Zaman dizisini oluştur
16
17  Hiz(1) = fscanf(S1, '%d');             % S1 seri portu üzerinden hızı oku
18
19  for i = 2:201                          % Hız okuma döngüsü
20      Hiz(i) = fscanf(S1, '%d');         % S1 seri portu üzerinden hızı oku
21
22      a = [t(i-1) t(i)];                 % Bir önceki ve şimdiki zaman
23      b = [Hiz(i-1) Hiz(i)];             % Bir önceki ve şimdiki hız
24
25      figure(1);                         % 1 numaralı şekli aç
26      line(a, b, 'LineWidth', 3);        % Çizgi ekle
27      hold on;                           % Çizime devam et
28  end
29
30  fclose(S1);                            % S1 seri portunu kapat
```

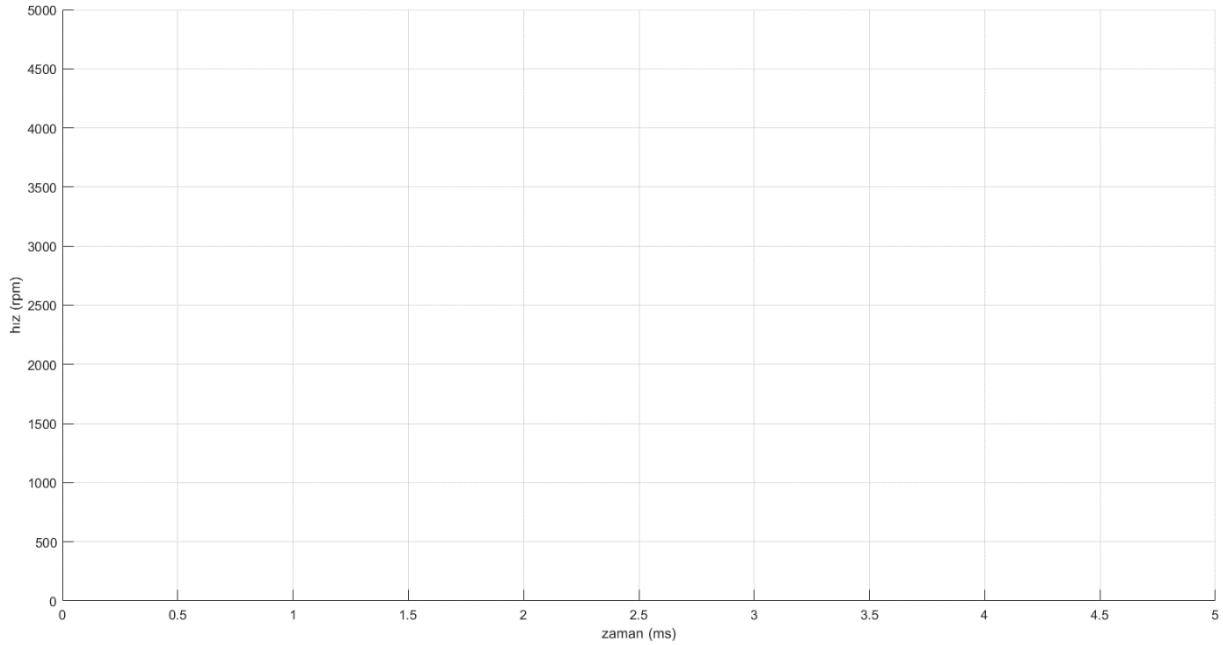
Şekil 6. Görsel bilgisayar yazılımı

6.3. Deneyin Yapılışı

6.3.1. Açık-Kapalı Kontrol

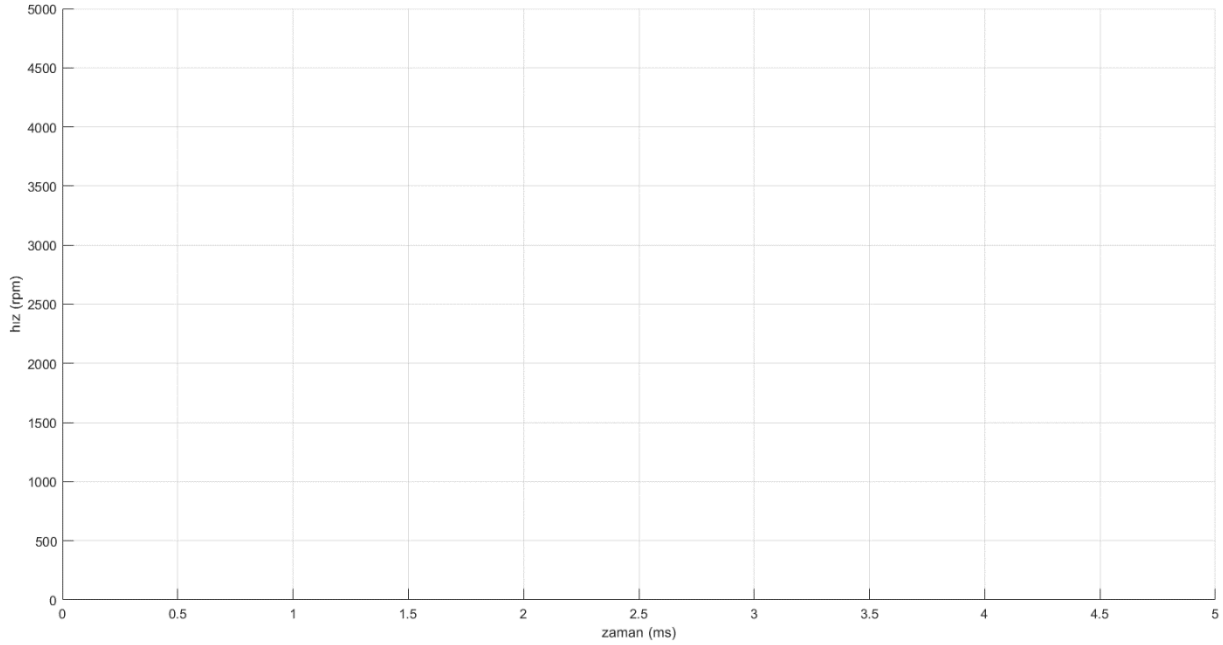
Aşağıdaki adımları takip ederek deneyi gerçekleştiriniz. Deney sonuçlarını ve gerekli notları belirtilen kısımlara yazınız.

1. *Arduino* editörünü açınız ve *Şekil 4*'deki kodu yazınız.
2. *Şekil 4*'deki kodun `13.` satırını `int Ref = 2000;` olarak değiştiriniz
3. *Yükle* butonuna basarak kodu işlemciye yükleyiniz.
4. *MATLAB* editörünü açınız ve *Şekil 6*'daki kodu yazınız.
5. *Şekil 6*'daki kodun `4.` satırındaki `S1 = serial('COM6','BaudRate',9600);` port numarasını mikrodenetleyicinin bağlı olduğu port numarası ile değiştirmeyi unutmayınız.
6. *Run* butonuna basarak kodu çalıştırınız.
7. *Hız-zaman grafiğini* çiziniz ve yorumlayınız.

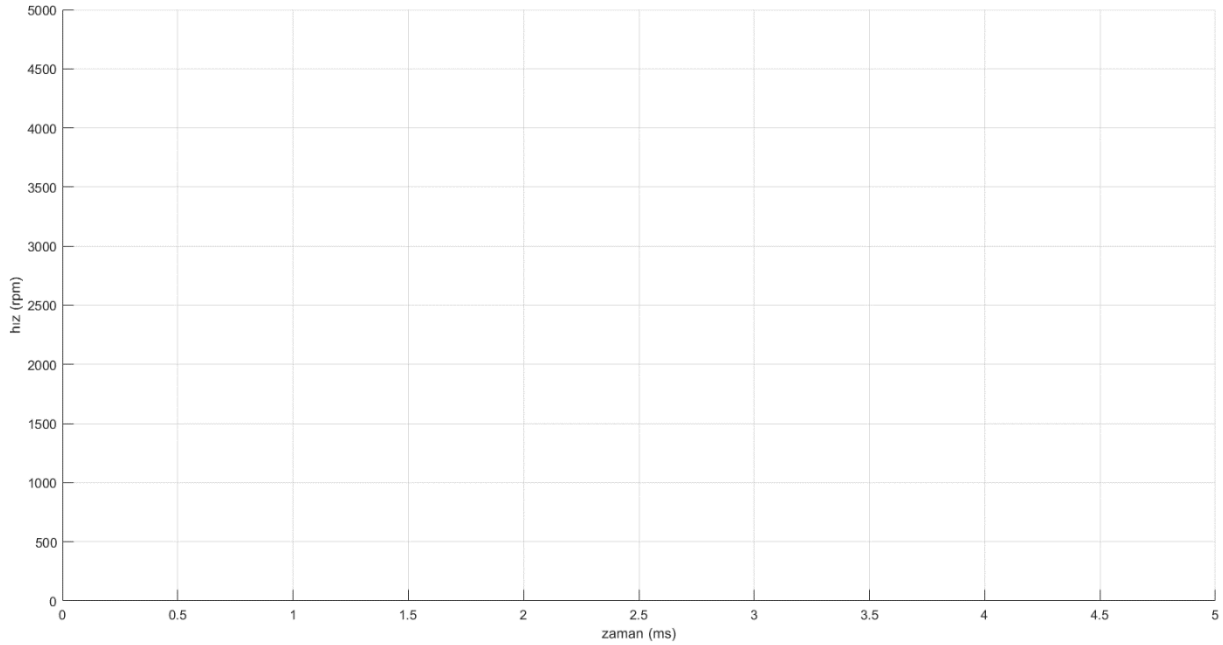


8. *Şekil 4*'deki kodun `13.` satırını `int Ref = 3000;` olarak değiştiriniz ve kodu işlemciye yükleyiniz.

9. *MATLAB* editörünü açınız ve *Run* butonuna basarak kodu çalıştırınız. Elde ettiğiniz *hız-zaman grafiğini* çiziniz ve yorumlayınız.



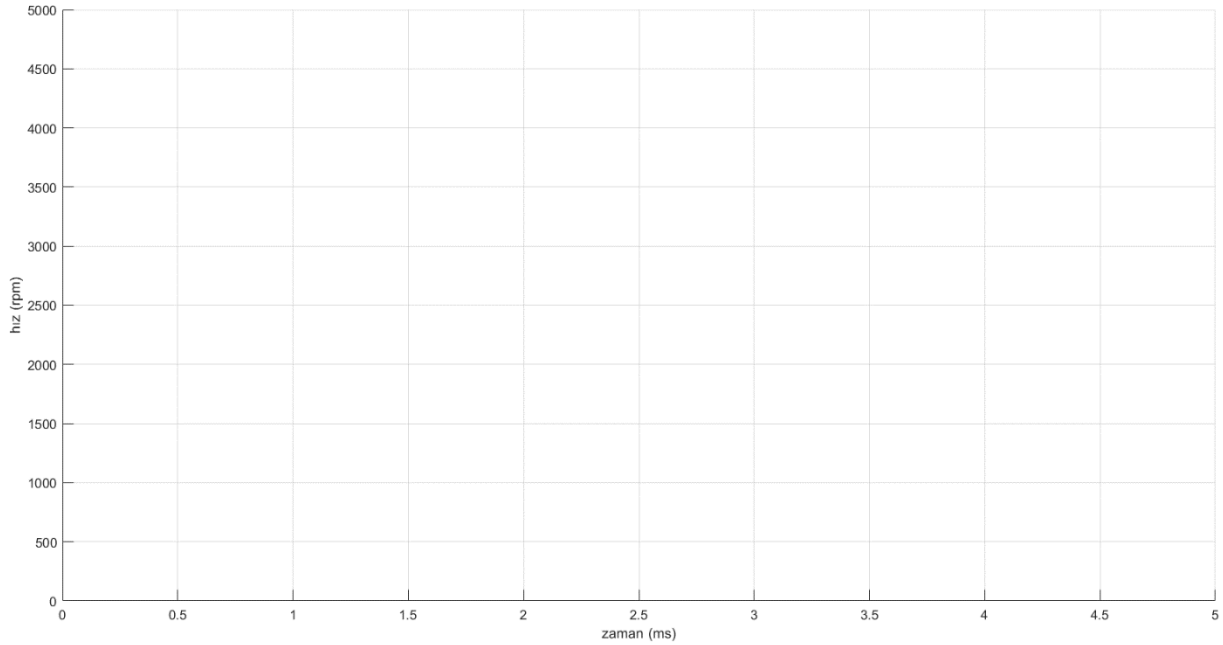
10. *Şekil 4*'deki kodun `13.` satırını `int Ref = 4000;` olarak değiştiriniz ve kodu işlemciye yükleyiniz.
11. *MATLAB* editörünü açınız ve *Run* butonuna basarak kodu çalıştırınız. Elde ettiğiniz *hız-zaman grafiğini* çiziniz ve yorumlayınız.



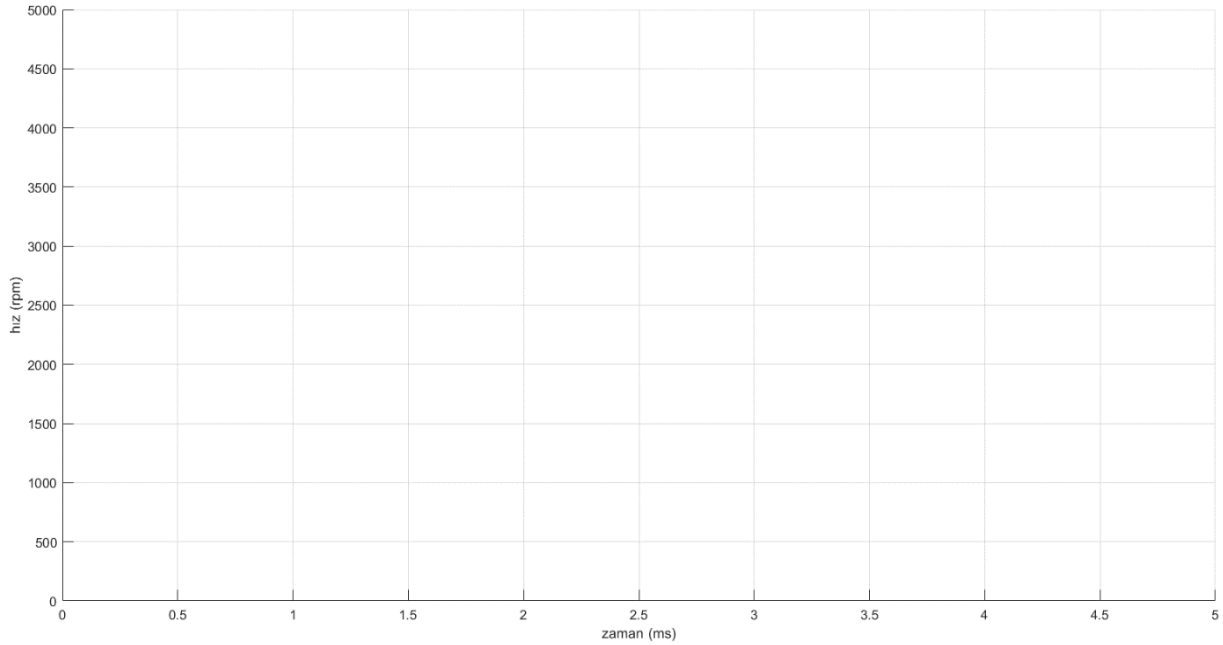
6.3.2. Oransal Kontrol

Aşağıdaki adımları takip ederek deneyi gerçekleştiriniz. Deney sonuçlarını ve gerekli notları belirtilen kısımlara yazınız.

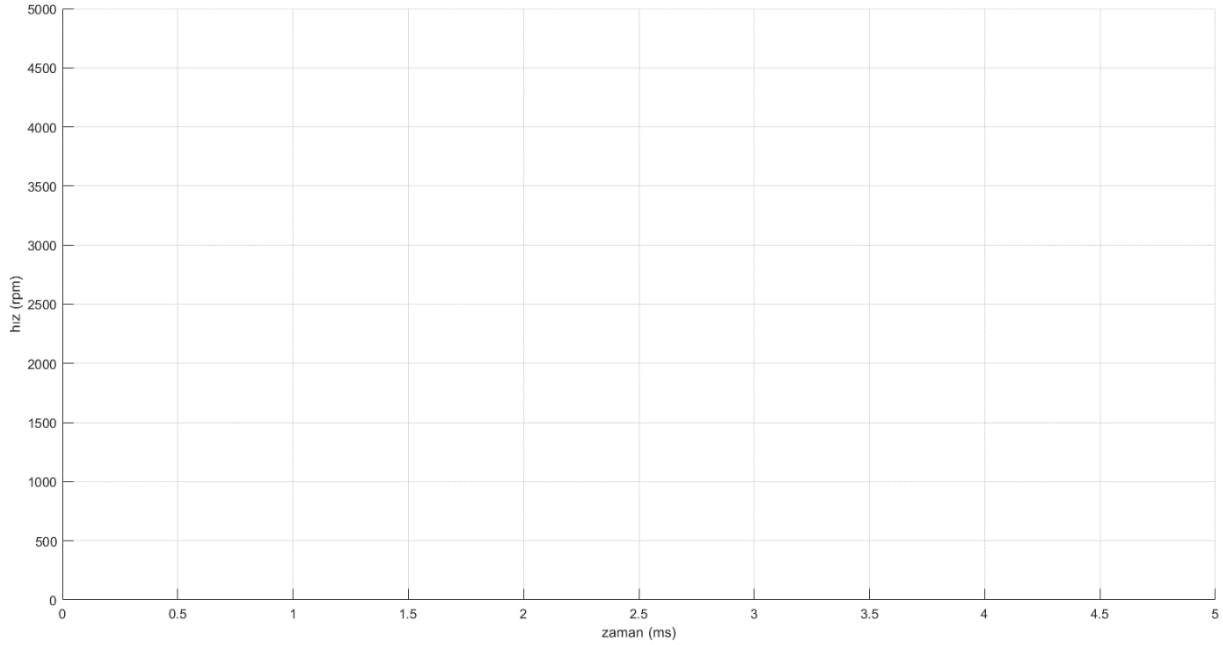
1. *Arduino* editörünü açınız ve *Şekil 5*'deki kodu yazınız.
2. *Şekil 5*'deki kodun `20.` satırını `float Kp = 1;` olarak değiştiriniz ve kodu işlemciye yükleyiniz.
3. *MATLAB* editörünü açınız ve *Run* butonuna basarak kodu çalıştırınız. Elde ettiğiniz *hız-zaman grafiğini* çiziniz ve yorumlayınız.



4. *Şekil 5*'deki kodun `20.` satırını `float Kp = 10;` olarak değiştiriniz ve kodu işlemciye yükleyiniz.
5. *MATLAB* editörünü açınız ve *Run* butonuna basarak kodu çalıştırınız. Elde ettiğiniz *hız-zaman grafiğini* çiziniz ve yorumlayınız.



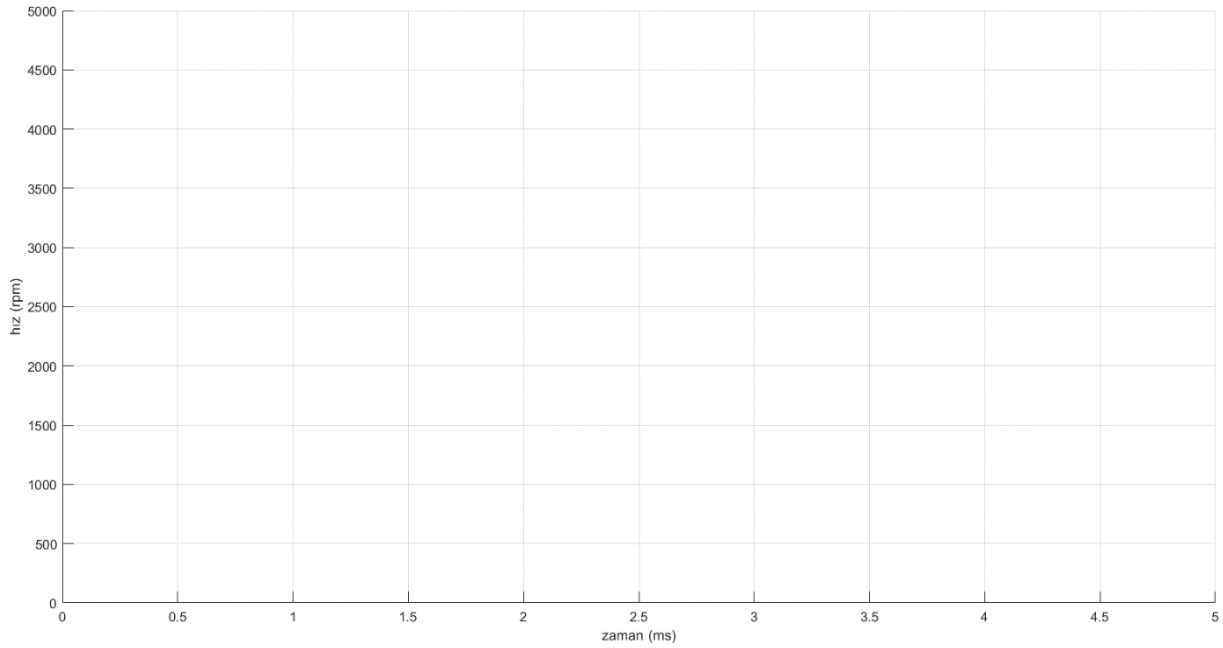
6. *Şekil 4*'deki kodun `20.` satırını `float Kp = 0.1;` olarak değiştiriniz ve kodu işlemciye yükleyiniz.
7. *MATLAB* editörünü açınız ve *Run* butonuna basarak kodu çalıştırınız. Elde ettiğiniz *hız-zaman grafiğini* çiziniz ve yorumlayınız.



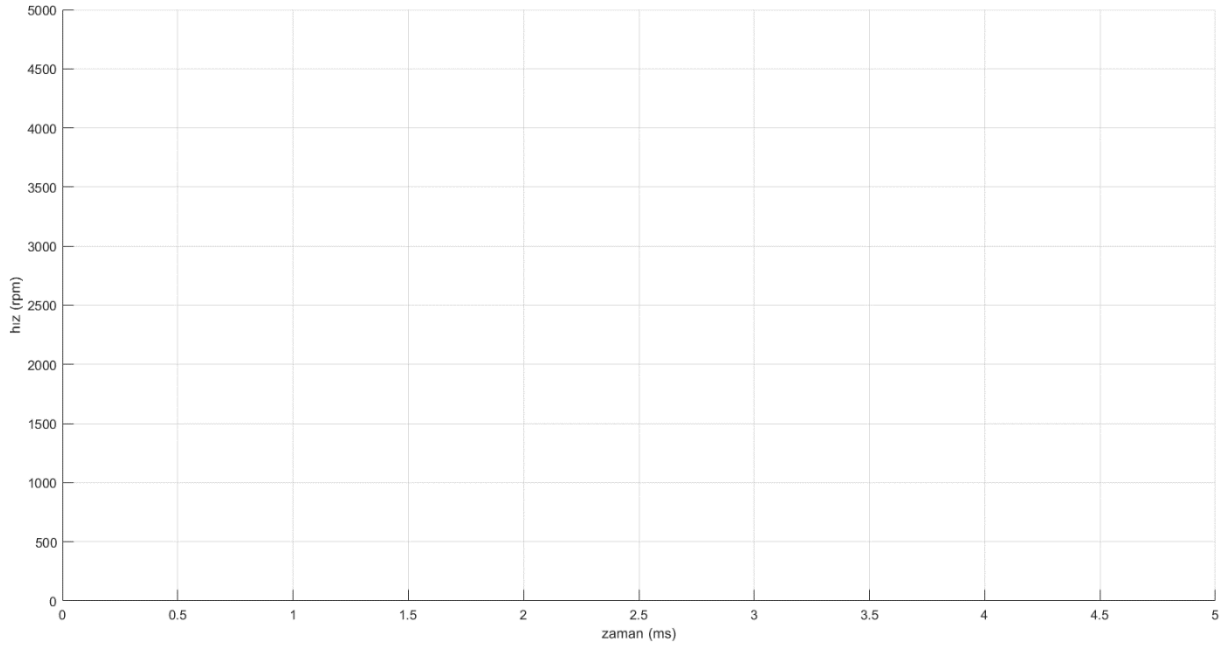
6.3.3. Oransal İntegral Kontrol

Aşağıdaki adımları takip ederek deneyi gerçekleştiriniz. Deney sonuçlarını ve gerekli notları belirtilen kısımlara yazınız.

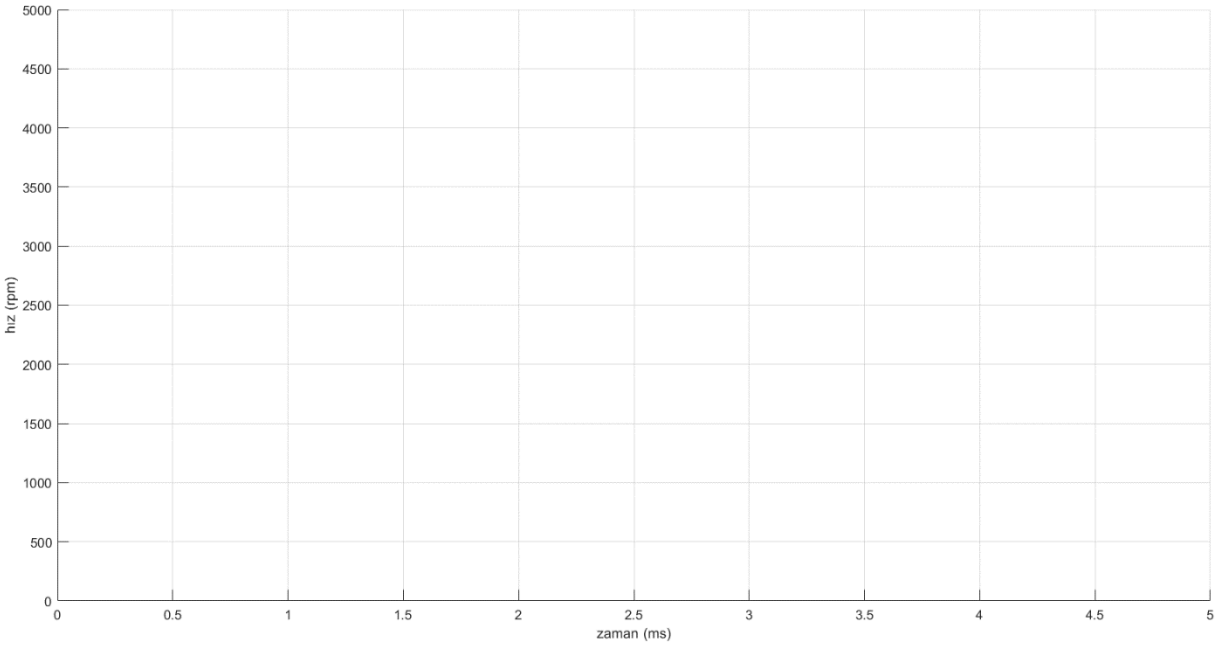
1. *Şekil 5*'deki kodun `20.` satırını `float Kp = 1;` olarak değiştiriniz.
2. *Şekil 5*'deki kodun `21.` satırını `float Ki = 1;` olarak değiştiriniz ve kodu işlemciye yükleyiniz.
3. *MATLAB* editörünü açınız ve *Run* butonuna basarak kodu çalıştırınız. Elde ettiğiniz *hız-zaman grafiğini* çiziniz ve yorumlayınız.



4. *Şekil 5*'deki kodun `20.` satırını `float Kp = 0.1;` olarak değiştiriniz.
5. *Şekil 5*'deki kodun `21.` satırını `float Ki = 1;` olarak değiştiriniz ve kodu işlemciye yükleyiniz.
6. *MATLAB* editörünü açınız ve *Run* butonuna basarak kodu çalıştırınız. Elde ettiğiniz *hız-zaman grafiğini* çiziniz ve yorumlayınız.



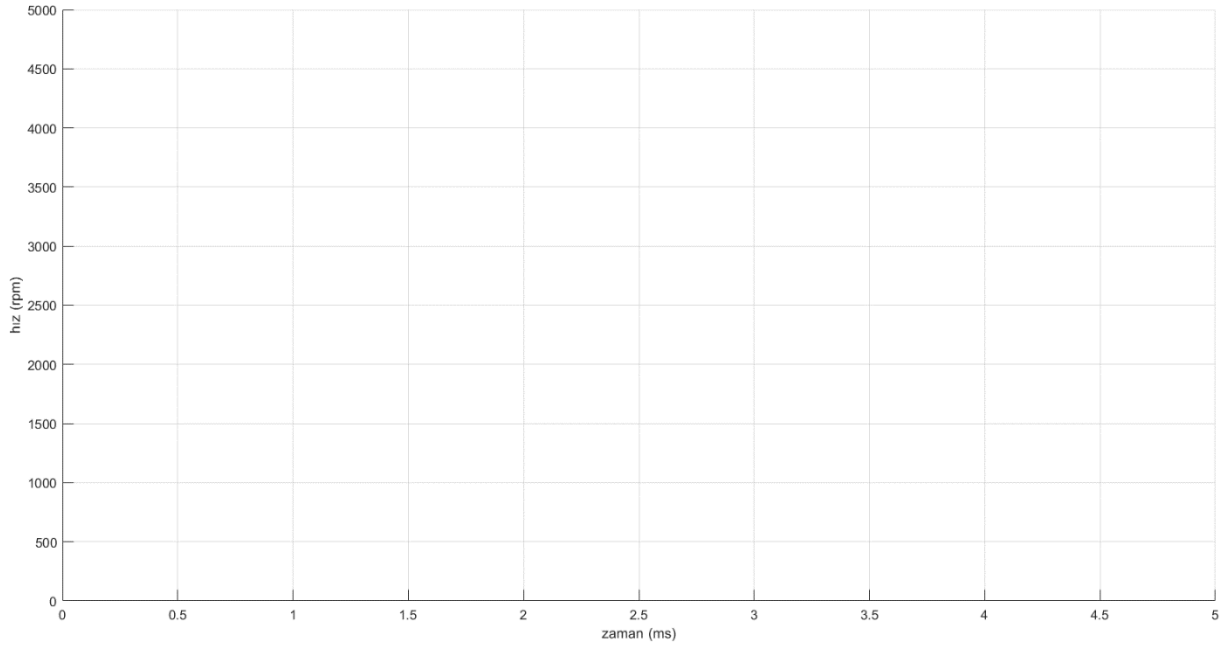
7. *Şekil 5*'deki kodun `20.` satırını `float Kp = 1;` olarak değiştiriniz.
8. *Şekil 5*'deki kodun `21.` satırını `float Ki = 0.15;` olarak değiştiriniz ve kodu işlemciye yükleyiniz.
9. *MATLAB* editörünü açınız ve *Run* butonuna basarak kodu çalıştırınız. Elde ettiğiniz *hız-zaman grafiğini* çiziniz ve yorumlayınız.



6.3.4. Oransal İntegral Türev Kontrol

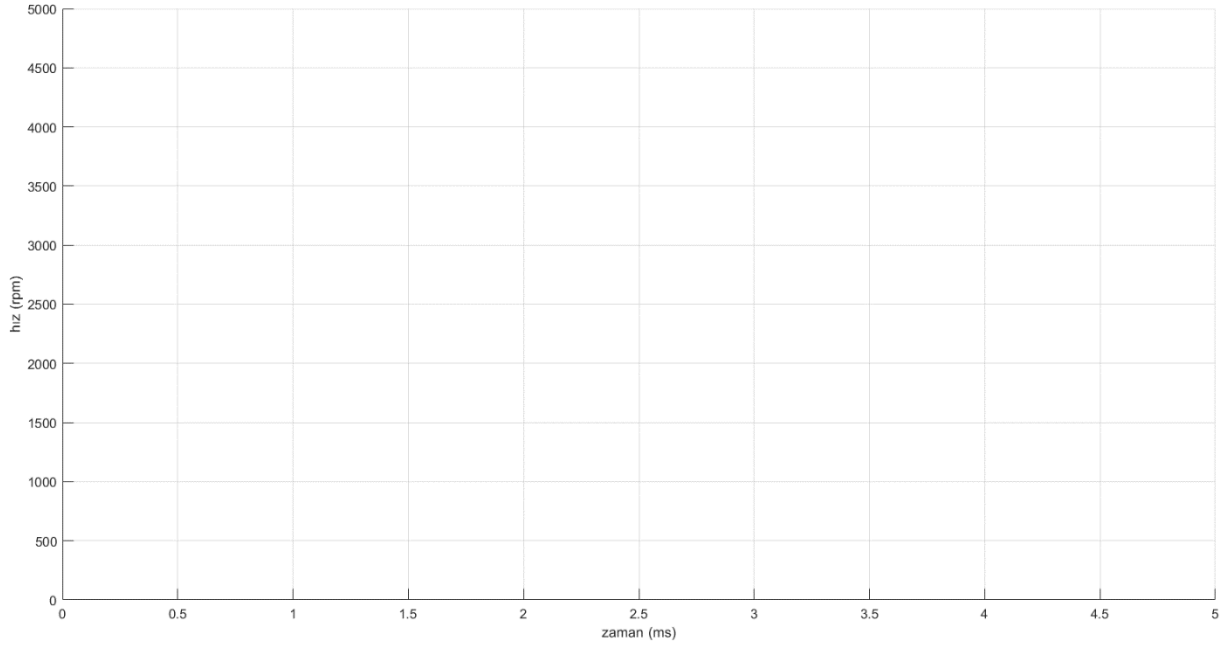
Aşağıdaki adımları takip ederek deneyi gerçekleştiriniz. Deney sonuçlarını ve gerekli notları belirtilen kısımlara yazınız.

1. *Şekil 5*'deki kodun `20.` satırını `float Kp = 1;` olarak değiştiriniz.
2. *Şekil 5*'deki kodun `21.` satırını `float Ki = 1;` olarak değiştiriniz.
3. *Şekil 5*'deki kodun `22.` satırını `float Kd = 1;` olarak değiştiriniz ve kodu işlemciye yükleyiniz
4. *MATLAB* editörünü açınız ve *Run* butonuna basarak kodu çalıştırınız. Elde ettiğiniz *hız-zaman grafiğini* çiziniz ve yorumlayınız.



5. *Şekil 5*'deki kodun `22.` satırını `float Kd = 0.1;` olarak değiştiriniz ve kodu işlemciye yükleyiniz (K_p = 1, K_i = 1 ve $\boldsymbol{K_d}$ **= 0,1**).

6. *MATLAB* editörünü açınız ve *Run* butonuna basarak kodu çalıştırınız. Elde ettiğiniz *hız-zaman grafiğini* çiziniz ve yorumlayınız.



7. *Şekil 5*'deki kodun `21.` satırını `float Ki = 0.1;` olarak değiştiriniz ve kodu işlemciye yükleyiniz (K_p = 1, $\boldsymbol{K_i}$ **= 0,1** ve K_d = 0,1).
8. *MATLAB* editörünü açı nız ve *Run* butonuna basarak kodu çalıştırınız. Elde ettiğiniz *hız-zaman grafiğini* çiziniz ve yorumlayınız.