ÇALIŞMA HAFTASI 1				Başlama Tarihi	11.06.2018
				Bitme Tarihi	16.06.2018
GÜN	**YAPILAN İŞLER**				**SAYFA NO**
Pazartesi	C# Programlama	GMap.NET Kütüphanesi ve Uygulamaları			1
Salı					2
Çarşamba		ZedGraph Kütüphanesi ve Uygulamaları			3
Perşembe	TATİL	Ramazan Bayramı Arifesi			–
Cuma		Ramazan Bayramı			
Cumartesi					
STAJ YETKİLİSİ	Ünvan Ad ve Soyad	**Doktor Öğretim Üyesi Türker Türker**		Firma İmza Kaşe	

ÇALIŞMA HAFTASI 2				Başlama Tarihi	18.06.2018
				Bitme Tarihi	23.06.2018
GÜN	**YAPILAN İŞLER**				**SAYFA NO**
Pazartesi	C# Programlama	ZedGraph Kütüphanesi ve Uygulamaları			4
Salı					5
Çarşamba	PLC Programlama	Giriş			6
Perşembe		Hafıza Birimi, Kontaklar ve Atama Operatörü			7
Cuma					8
Cumartesi	TATİL	Hafta Sonu Tatili			–
STAJ YETKİLİSİ	Ünvan Ad ve Soyad	**Doktor Öğretim Üyesi Türker Türker**		Firma İmza Kaşe	

ÇALIŞMA HAFTASI 3				Başlama Tarihi	25.06.2018
				Bitme Tarihi	30.06.2018
GÜN	**YAPILAN İŞLER**				**SAYFA NO**
Pazartesi	PLC Programlama	Zamanlayıcı Blokları			
Salı					
Çarşamba		Sayıcı Blokları			
Perşembe					
Cuma		Analog Sinyal			
Cumartesi	TATİL	Hafta Sonu Tatili			–
STAJ YETKİLİSİ	Ünvan Ad ve Soyad	**Doktor Öğretim Üyesi Türker Türker**		Firma İmza Kaşe	

ÇALIŞMA HAFTASI 4				Başlama Tarihi	02.07.2018
				Bitme Tarihi	07.07.2018
GÜN	**YAPILAN İŞLER**				**SAYFA NO**
Pazartesi	PLC Programlama	Analog Sinyal			
Salı		Artımsal Enkoder			
Çarşamba					
Perşembe		HMI Programlama			
Cuma					
Cumartesi	TATİL	Hafta Sonu Tatili			–
STAJ YETKİLİSİ	Ünvan Ad ve Soyad	**Doktor Öğretim Üyesi Türker Türker**		Firma İmza Kaşe	

ÇALIŞMA HAFTASI 5				Başlama Tarihi	09.07.2018
				Bitme Tarihi	14.07.2018
GÜN	**YAPILAN İŞLER**				**SAYFA NO**
Pazartesi	Qt C++ Programlama	Giriş			
Salı		Sık Kullanılan Sınıflar			
Çarşamba					
Perşembe		QList Sınıfı, QListIterator ve QMutableListIterator Sınıfları			
Cuma					
Cumartesi	TATİL	Hafta Sonu Tatili			–
STAJ YETKİLİSİ	Ünvan Ad ve Soyad	**Doktor Öğretim Üyesi Türker Türker**		Firma İmza Kaşe	

ÇALIŞMA HAFTASI 6				Başlama Tarihi	16.07.2018
				Bitme Tarihi	21.07.2018
GÜN	**YAPILAN İŞLER**				**SAYFA NO**
Pazartesi	Qt C++ Programlama	QLinkedList Sınıfı			
Salı		QMap Sınıfı			
Çarşamba		QHash Sınıfı			
Perşembe		QThread Sınıfı			
Cuma					
Cumartesi	TATİL	Hafta Sonu Tatili			–
STAJ YETKİLİSİ	Ünvan Ad ve Soyad	**Doktor Öğretim Üyesi Türker Türker**		Firma İmza Kaşe	

<table>
<tr><td colspan="3" rowspan="2">ÇALIŞMA HAFTASI 7</td><td>Başlama Tarihi</td><td>23.07.2018</td></tr>
<tr><td>Bitme Tarihi</td><td>28.07.2018</td></tr>
<tr><td>GÜN</td><td colspan="3">YAPILAN İŞLER</td><td>SAYFA NO</td></tr>
<tr><td>Pazartesi</td><td rowspan="2">Qt C++ Programlama</td><td colspan="2" rowspan="2">QtAlgorithms Kütüphanesi</td><td></td></tr>
<tr><td>Salı</td><td></td></tr>
<tr><td>Çarşamba</td><td colspan="3" rowspan="4">STAJ SONU</td><td rowspan="4">–</td></tr>
<tr><td>Perşembe</td></tr>
<tr><td>Cuma</td></tr>
<tr><td>Cumartesi</td></tr>
<tr><td>STAJ YETKİLİSİ</td><td>Ünvan
Ad ve Soyad</td><td colspan="2">Doktor Öğretim Üyesi Türker Türker</td><td>Firma
İmza
Kaşe</td><td></td></tr>
</table>

1. GMap.NET Kütüphanesi

1.1. GMap.NET Nedir

GMap.NET, .NET için hazırlanmış ücretsiz, çok platformlu ve açık kaynak kodlu harita kontrolüdür. Bünyesinde Google, Yahoo!, Bing, OpenStreetMap, ArcGIS, Pergo, SigPac, Yendux, Mapy.cz, Maps.lt, iKarte.lv, NearMap, HereMap, CloudMade, WikiMapia ve MapQuest gibi birçok harita sağlayıcısı bulunur. Bu haritalar üzerinde katmanlar (overlay) kullanarak işaret (marker) koyulabilir, alan (polygon) çizilebilir ve rota oluşturulabilir.

Önbellek (cache) destekleyen GMap.NET, internet yokken bile önbellekteki veriyi kullanarak çalışmaya devam eder. Kontrolü güçlü kılan en önemli özelliklerinden biri de harita sağlayıcısının katmanlardan bağımsız olmasıdır. Böylece harita sağlayıcısı kaynaklı problemlerde harita sağlayıcısı değiştirildiğinde katmanlar yeni harita sağlayıcısında da çalışmaya devam eder.

Güncel sürümü v1.8.5 olup NuGet üzerinden başvuru dosyaları indirilebilir.

1.2. GMap.NET Kütüphanesinin Uygulamaya Eklenmesi

Windows Forms projesine `GMap.NET.Core.dll` ve `GMap.NET.WindowsForms.dll` başvuru dosyaları eklendikten sonra araç kutusuna da eklenir ve `GMapControl` aracı forma sürüklenerek projeye dahil edilmiş olur.

`GMapControl` sınıfının bazı özellikleri şunlardır:

- **CanDragMap <bool>:** Fare tuşuyla harita kaydırmayı aktif eder.
- **MarkersEnabled <bool>:** İşaretleri aktif eder.
- **PolygonsEnabled <bool>:** Alanları aktif eder.
- **RoutesEnabled <bool>:** Rotaları aktif eder.
- **MouseWheelZoomType <enum> :** Harita zoom tipini belirler.

- **MinZoom <int> :** Yapılabilecek en az zoom derecesini belirler.
- **MaxZoom <int> :** Yapılabilecek en fazla zoom derecesini belirler.
- **MouseWheelZoomEnabled <bool>:** Fare tekerleğiyle zoomu aktifleştirir.
- **Zoom <int> :** Mevcut zoom derecesini gösterir.
- **Position <PointLatLng> :** Harita merkezinin konumunu belirler.

`PointLatLng` sınıfı harita üzerindeki noktaların koordinatlarını tutmayı sağlar. Bu sınıfa ait iki özellik şunlardır:

- **Lat <double>:** Noktanın enlemini tutar.
- **Lng <double>:** Noktanın boylamını tutar.

GMap.NET Uygulaması – Haritayı Oluşturmak

Aracın formda çalışması için aşağıdaki özellikler ayarlanmalıdır :

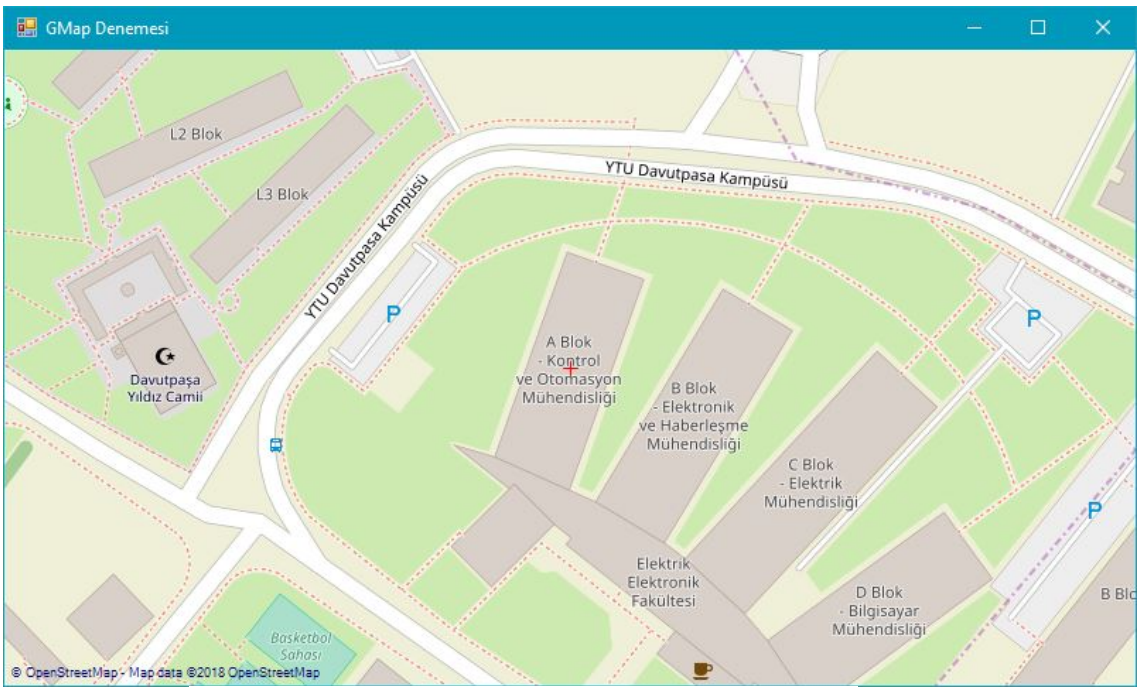
- **MapProvider :** Harita sağlayıcısı belirlenir (örn: `OpenStreetMap`).
- **Mode :** Harita çalışma modu belirlenir (örn: `ServerAndCache`).

Bu ayarlamalardan sonra `SetPositionByKeywords` methodu kullanarak kelimeyle veya doğrudan koordinat girerek pozisyon ayarlanabilir (Kod Parçası 1.1).

```
GMapControl gmap = new GMapControl();
// Harita kontrolü oluşturuldu.
gmap.MapProvider = GMapProviders.OpenStreetMap;
// Haritanın sağlayıcısı belirlenir.
GMaps.Instance.Mode = AccessMode.ServerAndCache;
// Haritanın çalışma modu belirlenir.
gmap.SetPositionByKeywords("YTÜ KOM");
// Kelimeyle koordinat bulunur.
```

Kod Parçası 1.1. gmap nesnesi kullanarak harita oluşturma

(Kod Parçası 1.1) derlendikten sonra program çalışacak ve program penceresi çizilecektir (Program Çıktısı 1.1).



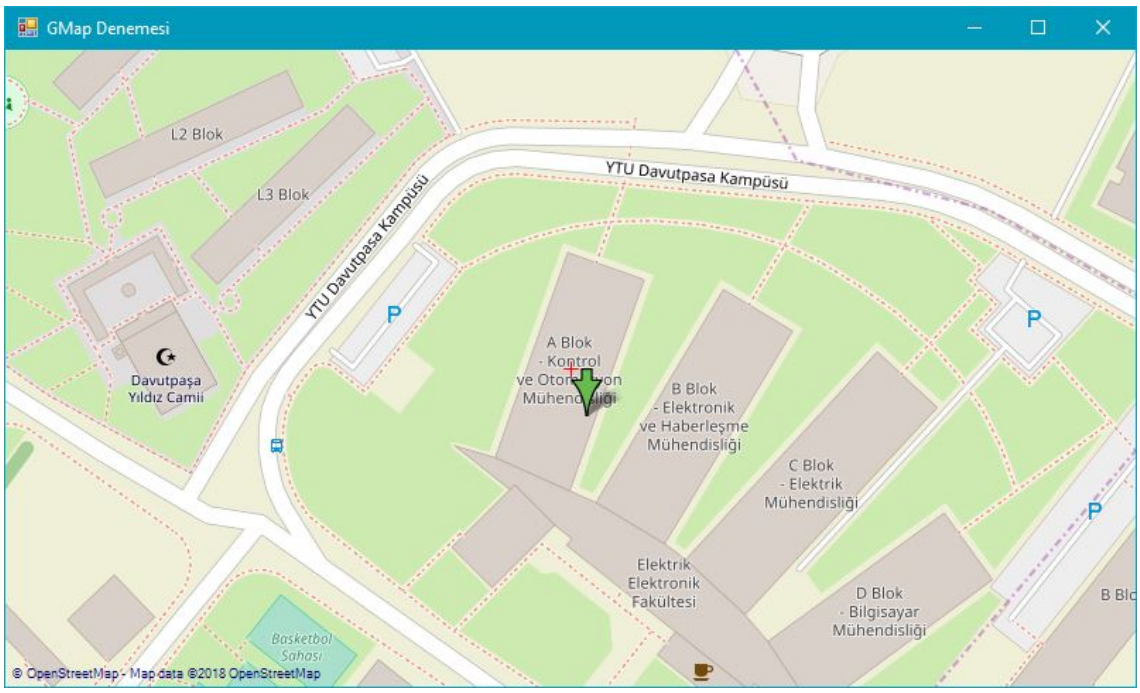
Program Çıktısı 1.1. YTÜ EEF'nin kuşbakışı görünüşü

1.3. GMap.NET Uygulaması – Katmanlar, İşaretler ve Araç Kutuları

Haritaya işaret, rota ya da alan eklenecekse bunların katmana eklenmesi gereklidir. `GMapOverlay` sınıfından katman oluşturulabilir. Projeye göre farklı katmanlar oluşturup her katmana farklı işaretlemeler yapmak mümkündür. `GMapMarker` sınıfından işaret oluşturulabilir. Oluşturucu methoduna `PointLatLng` ve `GMarkerGoogleType` parametreler verilerek istenen koordinatta ve işaret görselinde işaret oluşturulur. `GMarkerGoogleType` enum'u işaret görselini belirler.

```
GMapOverlay katman = new GMapOverlay(); // Katman oluşturuldu.
GMapMarker işaret = new GMarkerGoogle(
       new PointLatLng(48.8617774, 2.349272), // Koordinatlar
       GMarkerGoogleType.arrow); // İşaret tipi
// İşaret oluşturuldu.
katman.Markers.Add(işaret); // İşaret katmana eklendi.
gmap.Overlays.Add(katman);  // Katman haritaya eklendi.
```

Kod Parçası 1.2. Katman nesnesinin türetilip düzenlenmesi



Program Çıktısı 1.2. İşaret KOM bloğu üzerinde

2. ZedGraph

2.1. ZedGraph Nedir

ZedGraph, .NET için hazırlanmış ücretsiz ve açık kaynak kodlu grafik kontrolüdür. 2 boyutlu grafik (çizgi, sütun, pasta grafikleri) çizmek için kullanılır. Grafiği tamamen ve ayrıntılı şekilde düzenlemeyi sağlasa da çoğu ayarlarında varsayılanı olduğu için kullanımı kolaydır.

2.2. ZedGraph Kütüphanesinin Uygulamaya Eklenmesi

Windows Forms projesine `ZedGraph.dll` başvuru dosyaları eklendikten sonra araç kutusuna da eklenir ve `ZedGraphControl` aracı forma sürüklenerek bu kütüphane projeye dahil edilmiş olur.

Grafik alanıyla ilgili işlemler için `ZedGraphControl.GraphPane` sınıfı kullanılır.

Bu alana ait bazı üye sınıflar şunlardır:

- **Title:** Grafik başlığı ve ilgili parametrelerini tutar.
- **XAxis:** Grafiğin X ekseniyle ilgili bilgilerini tutar.
- **YAxis:** Grafiğin Y ekseniyle ilgili bilgilerini tutar.
- **IsEnableHZoom <bool> :** X ekseninde zoom yapmayı aktif eder.
- **IsEnableHPan <bool> :** X ekseninde alan seçerek zoom yapmayı aktif eder.
- **IsShowHScroolBar <bool> :** X ekseninde hareketi sağlayan barı aktif eder.

Eksen bilgilerini tutan `XAxis` ve `YAxis` sınıflarına ait bazı üyeler şunlardır:

- **Color :** Eksen rengini tutan sınıftır.
- **IsVisible <bool> :** Eksen görünürlüğünü aktif eder.
- **Title:** Eksen başlığı ve ilgili parametrelerini tutar.
- **Type <AxisType> :** Eksen tipini belirler.

2.3. ZedGraph Uygulaması – Grafiği Oluşturmak

Aşağıdaki kod parçasıyla projeye sürüklenerek daha önce eklenmiş harita kontrolünü düzenlemek mümkündür:

```
ZedGraphControl zgc_grafik = new ZedGraphControl();
zgc_grafik.GraphPane.Title.Text = “AGNO Değişim Grafiği”;
zgc_grafik.GraphPane.XAxis.Title.Text = “Dönem”;
zgc_grafik.GraphPane.YAxis.Title.Text = “AGNO”;
double[] x = { 1, 2, 3, 4 };
double[] y = { 2.5, 2.78, 3.17, 3.19 };
```

Kod Parçası 2.1. Grafik kontrolünün oluşturulması

(Kod Parçası 2.1) derlendikten sonra grafiğin düzenlenmiş başlıklarıyla beraber forma eklendiği görünecektir fakat bu grafiğe herhangi bir veri eklenmediği için grafik ekranı boş görünmektedir.

Grafiğe veri eklemek için `GraphPane` sınıfından `AddCurve` methodu kullanılır. Bu method bu veriyi temsil eden `LineItem` nesnesi döndürür. İsteğe bağlı olarak bu

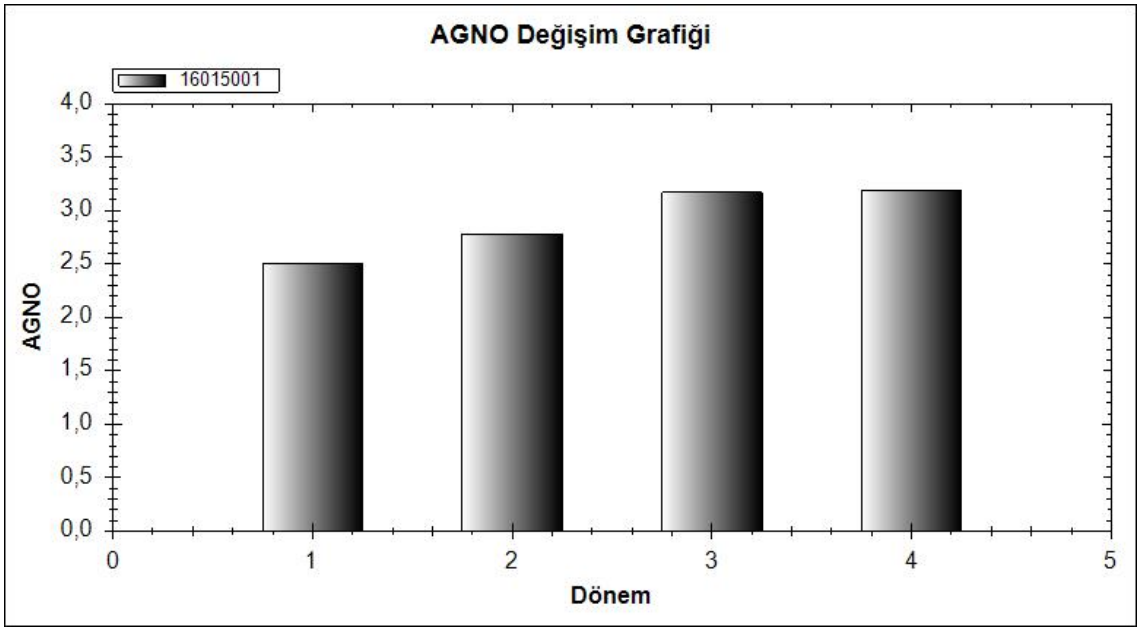
nesnenin referansı saklanabilir. Bu method için aşağıdaki paremetreler kullanılır:

- **String:** Verinin adı.
- **Double[]:** Verilerin X değerlerinin tutulduğu dizi.
- **Double[]:** Verilerin Y değerlerinin tutulduğu dizi.
- **Color:** Verilerin grafikte görüneceği renk.

Aşağıdaki kod parçası da yukardakiyle birlikte derlenerek veri gösterilebilir:

```
BarItem v1 = zgc_grafik.AddBar("16015001", x, y, Color.Black);
```

Kod Parçası 2.2. Grafik kontrolüne veri eklenmesi.



Tüm kodlar beraber derlendiğinde grafik aşağıdaki gibi oluşacaktır:

Program Çıktısı 2.1. Grafiğin formda çizilmiş hali

AddBar methodunun bu aşırı yüklemesi kullanıldığında dizilerin yapısından dolayı daha fazla nokta eklenemeyecektir. Noktalar üzerinde daha fazla kontrole sahip olmak için `PointPairList` veya `RollingPointPairList` sınıfı kullanılır.

`RollingPointPairList` sınıfının `PointPairList` sınıfından farkı `RollingPointPairList` sınıfın oluşturucu methoduna kapasite parametresi

verilerek listenin kaldırabileceği maksimum nokta sayısı belirlenir. Kapasiteye ulaşıldığındaysa FIFO mantığı işler.

Yukarıda verilen kod parçasını kullanmak yerine aşağıda verilen kod parçası kullanıldığında burada bahsettiğimiz gibi noktalar üzerinde daha fazla kontrolümüz olacaktır:

```
PointPairList xy = new PointPairList(x, y);
BarItem v1 = zgc_grafik.AddBar("16015001", xy, Color.Black);
noktalar.Add(5, 3.3);
```

Kod Parçası 2.3. Grafik kontrolüne çalışma zamanında düzenlenebilir veri eklenmesi

Kodda da görüldüğü üzere nesne türetildikten sonra yeni bahsedilen aşırı yükleme kullanılarak veri eklenmiş ve daha sonra da verilere yeni bir örnek eklenmiştir.

Program çalışmadan önce bu ekleme yapıldığı için yeni örnek sorunsuz bir şekilde görülebilmektedir, fakat çalışma zamanında örnek eklendiği takdirde örnek başarıyla eklenebilse de programda görünmeyecektir, bunun sebebi grafik kontrolünün görsel olarak güncellenmemesidir, diğer bir değişle yeniden çizilmemesidir.

Bir program ilgili form penceresinin konumu değiştiğinde yeniden çizilir. Örneğin simge durumuna küçültülünce ya da ekranda yeri değişince yeniden çizilir. Bunları yapmadan yeniden çizmeyi zorlamak için harita kontrolüne ait parametresiz *Refresh* methodu kullanılır.

Eğer eklenen yeni örnek grafiğin gösterdiği mevcut alanın dışındaysa yeni noktayı da içine alacak yeni bir alanın çizilmesi için yine *GraphPane* sınıfından parametresiz *AxisChange* methodu kullanılır.

```
zgc_grafik.Refresh();
zgc_grafik.GraphPane.AxisChange();
```

Kod Parçası 2.4. Grafiğin yeniden çizilmesi

3. PLC

3.1. PLC Nedir?

Programlanabilir lojik kontrolcü, makine kontrolü ve fabrika üretim hatları gibi alanlarda kullanılabilen otomasyon cihazıdır. Bilgisayarlara göre daha fazla giriş ve çıkışı olan bu cihazların en büyük artıları gürültülere, mekanik darbelere ve sıcaklık farklılıklarına dayanıklı olmalarıdır.

PLC'ler, cihazları üreten şirketlere göre farklı işletim sistemlerine sahip olur. Bu işletim sistemlerine kontrol edilecek sisteme uygun programlarlar yüklenebilir. Bu programlar da ms düzeyinde girişleri alarak gerçek zamanlıya yakın çıkışlar üretecek şekilde çalışır.

PLC'ler aşağıdaki 4 ana bölümden oluşur.

- Merkezi işlem birimi :

CPU, yazılan programdaki işlemleri gerçekleştiren birimdir.

- Giriş birimi :

Kontrol edilen sistemle ilgili analog girişleri PLC'nin anlayacağı sayısal seviyeye dönüştüren birimdir. Giriş birimi voltaj değerleri DC veya AC olabilir.

- Hafıza birimi :

PLC içinde farklı görevler yapan hafızalar bulunur. Bir kısmı yazılan programı saklarken bir kısmı veri saklar.

- Çıkış birimi :

PLC'de hesaplanan lojik gerilimi ilgili kumanda elemanlarını sürmek amacıyla dönüştüren birimdir. Çıkış röleli veya transistörlü devreden oluşabilir. Transistörlü çıkış us düzeyinde çalışabilirken röleli çıkış ms düzeyinde çalışabilir.

3.2. Hafıza Birimi

PLC programında kullanılabilecek 3 hafıza alanı bulunuyor. Bunlar aşağıdaki gibidir:

- Giriş Hafızası : Adres atamasında I operatörü kullanılır.
- Çıkış Hafızası : Adres atamasında Q operatörü kullanılır.
- Genel Hafıza : Adres atamasında M operatörü kullanılır.
- Hafızanın en küçük birimi bit olarak ifade edilir.
- 8 bitlik bir alan bayt olarak ifade edilir.
- 2 baytlık bir alan word olarak ifade edilir.
- 2 wordlük bir alan dword olarak ifade edilir.

Bazı adres atama örnekleri aşağıdaki gibidir:

- 0. bayt 7. bit çıkış -> Q0.7
- 0. bayt çıkış -> QB0
- 0. word çıkış (QB0 – QB1) -> QW0
- 0. dword çıkış (QW0 – QW1 veya QB0 – QB1 – QB2 – QB3) -> QD0

3.3. Kontaklar ve Atamalar

PLC programı yazılırken hafızadaki birimleri kullanabilmek için kontaklar kullanılır. Kontakların birbirine bağlanmasıyla kontrol edilecek sisteme göre sayısal devre kurulur. Bu kontaklar bool tipindedir. Bir bloğun enerjili olmasına yüksek, olmamasına düşük denir.

Kontrol edilecek sisteme göre kullanılabilecek kontak türleri aşağıdaki gibidir:

- **Normalde açık kontak :** Yüksekken açık, düşükken kapalı kontaktır. (F)
- **Normalde kapalı kontak :** Düşükken kapalı, yüksekken açık kontaktır. (F’)
- **Yükselen kenar algılayan kontak :** Yüksek olduğu an kapalı kontaktır.

- **Alçalan kenar algılayan kontak :** Düşük olduğu an kapalı kontaktır.

Bool bir atama yapılacağı zaman atama bloğu kullanılır. Kullanılabilecek atama blokları:

- **Pozitif atama :** Yüksek olduğunda yüksek, düşük olduğunda düşük atar.
- **Negatif atama :** Düşük olduğunda yüksek, yüksek olduğunda düşük atar.
- **Set atama :** Yüksek olduğunda durumu değişse bile hep yüksek atar.
- **Reset atama :** Yüksek olduğunda durumu değişse bile hep düşük atar.

3.4. SIEMENS Simatic S7-1200 CPU1214C DC/DC/DC PLC ve TIA PORTAL

Bu PLC, Siemens tarafından üretilen bir PLC modeli olup piyasada 3 farklı modeli bulunur. Bunlar CPU 1211C, CPU 1212C ve CPU1214C olarak adlandırılır. Staj dönemi boyunca CPU1214C DC/DC/DC modeli kullanıldı. DC/DC/DC olduğundan DC beslenir, DC giriş alır ve DC çıkış verir. Kullanılan bu PLC'de ek olarak CSM 1277 haberleşme modülü bulunuyor. Bu modülle Ethernet kablosu kullanarak Profinet üzerinden PC ve HMI ile haberleşme mümkündür.

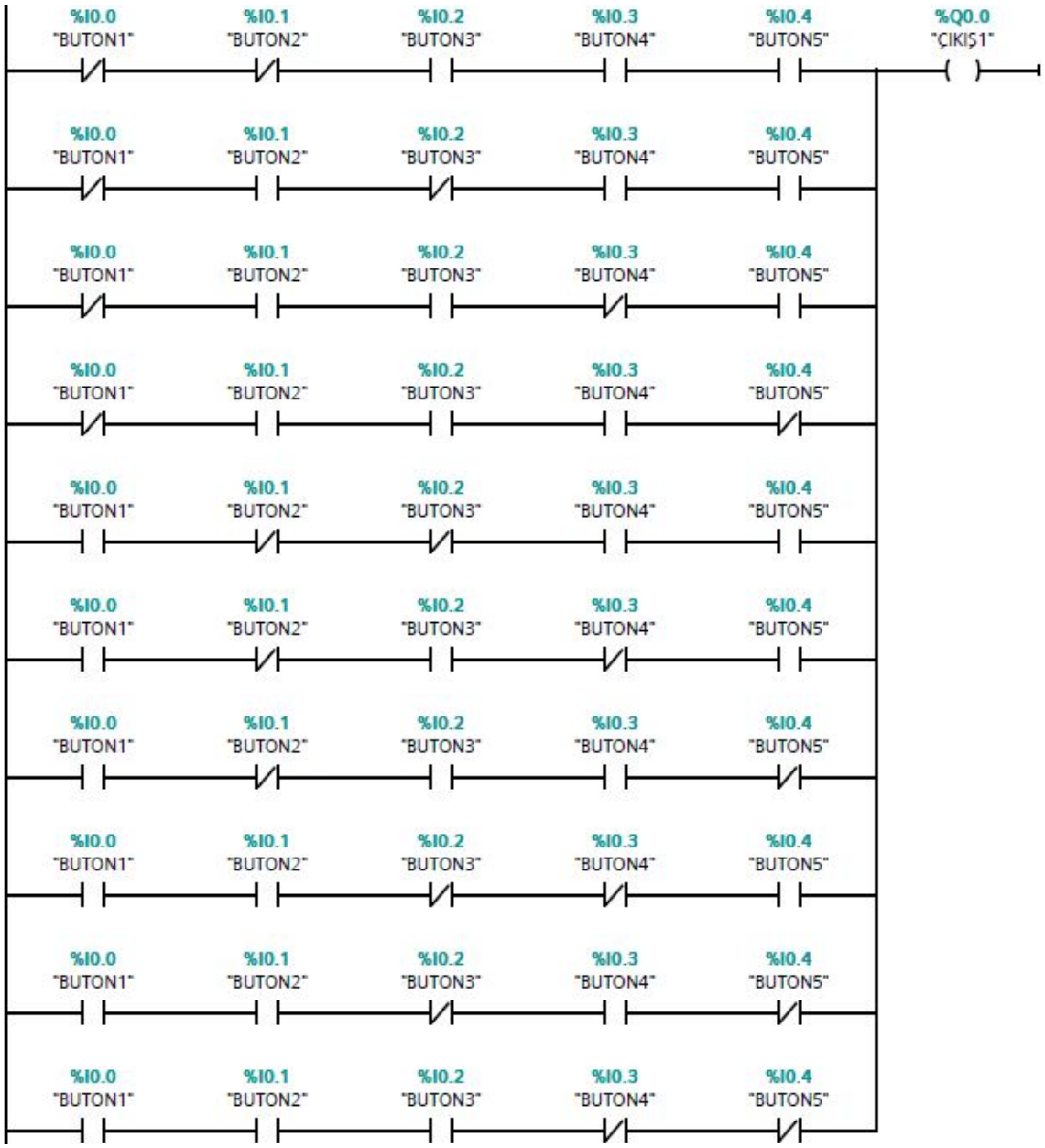
Siemens PLC'lerini programlamak için yine Siemens'in yazdığı Totally Integrated Automation Portal programı kullanılır. Staj dönemi boyunca V13 SP2 sürümü kullanıldı.

TIA PORTAL'da program yazmak için 3 dil bulunur. Bunlar Merdiven Diagramı (LAD), Fonksiyon Blok Diagramı (FBD) ve Yapısal Kontrol Dili (SCL) olarak adlandırılır. Staj döneminde yazılan tüm programlar LAD dilinde yazıldı.

3.5. LAD'e Giriş

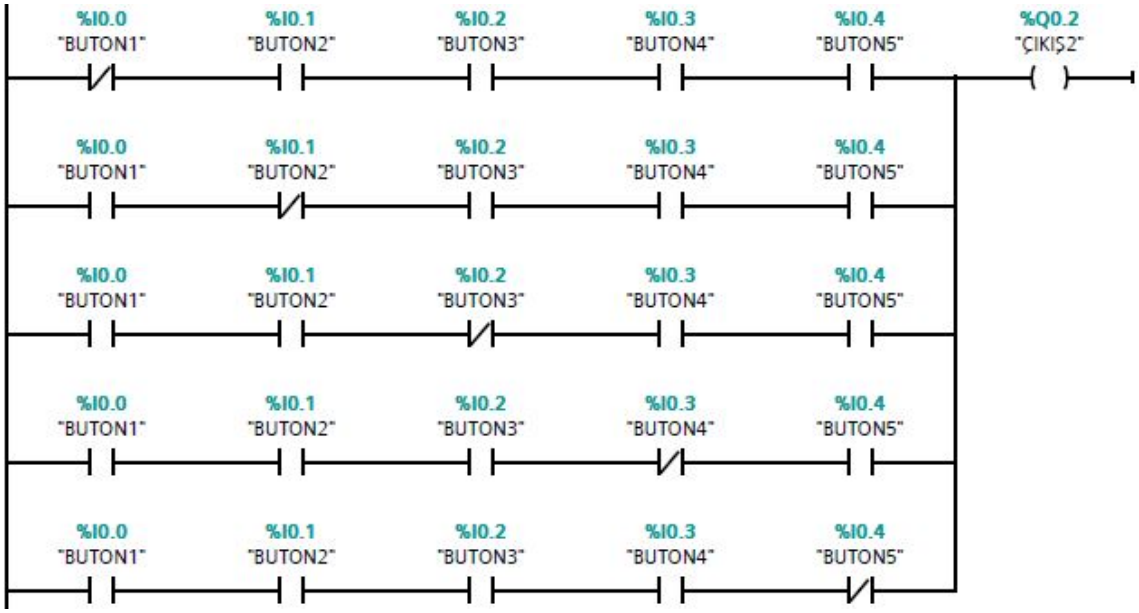
LAD'de soldan sağa ve yukarıdan aşağıya doğru bir akış bulunur. Her bir ağa ilgili bloklar koyularak algoritma oluşturulabilir. PLC, her ağdaki algoritmayı sırasıyla çalıştırır ve tüm ağlar bittikten sonra tekrar başa döner. Buna döngü denir ve bu döngü uzunluğu ağlardaki algoritmalara da bağlı olmak üzere ms seviyesinde olur.

LAD'i daha iyi anlamak için 5 girişten oluşan bir sistem düşünelim. 5 girişten sadece 3 tanesi yüksekse çıkışın yüksek olduğu bir algoritma tasarlayalım. 5 girişten 3 giriş seçilip yüksek, seçilmeyen 2 giriş düşük olacak. Bu cümlenin matematiksel karşılığı C(5,2)'dir. `C(5,2) = 10` olduğundan bu 10 ihtimali LAD ile yazıp ortak çıkışa bağlarsak bu algoritmayı tasarlamış oluruz.



Kod Parçası 3.1. 3 giriş yükseksen çıkışı yüksek olan sistem

Aynı sistemde bu sefer sadece 4 giriş yüksekken çıkışın yüksek olduğu algoritmayı düşünelim. Aynı mantıkla C(5,4) = 5 olduğundan bu 5 ihtimali LAD ile yazıp ortak çıkışa bağlarsak bu algoritmayı da tasarlamış oluruz.



Kod Parçası 3.2. 4 giriş yüksekken çıkışı yüksek olan sistem

3.6. Zamanlayıcı Blokları

Zamana göre tasarlayacağımız algoritmalar olduğunda kullanabileceğimiz bloklardır. İlk zamanlar sadece TON bloğundan oluşan zamanlayıcı blokları şuanda 4 tanedir, bunlar aşağıdaki gibidir:

- TON (Timer On-Delay)
- TOF (Timer Off-Delay)
- TP (Timer-Pulse)
- TONR (Timer On-Delay with Reset)

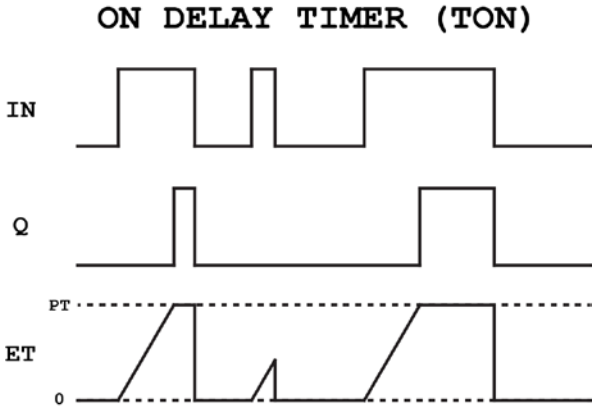
Bu zamanlayıcı bloklarının girişleri şunlardır:

- **IN (Input) (Bool)**: Zamanlayıcının girişi olup zamanlayıcıyı çalıştırır.
- **PT (Preset Time) (Time):** ET'nin üst sınırıdır.

Bu zamanlayıcı bloklarının çıkışları şunlardır:

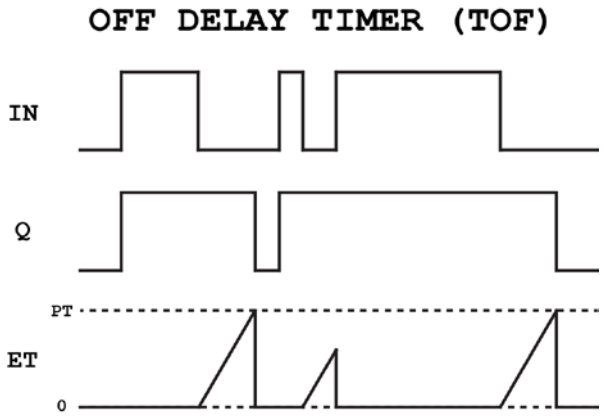
- **Q (Output) (Bool):** Zamanlayıcının çıkışıdır.
- **ET (Elapsed Time) (Time):** Zamanlayıcı çalışırken geçen süreyi verir.

3.6.1. TON (Timer On-Delay)



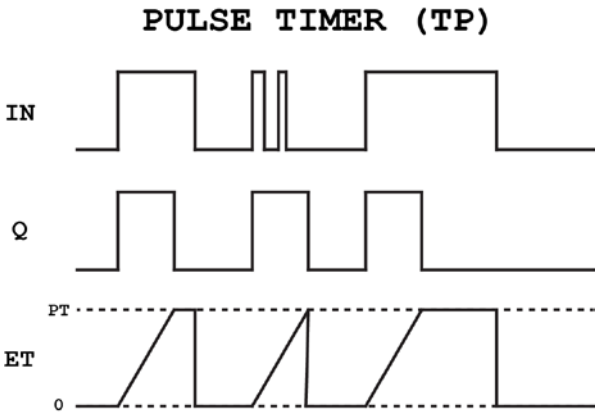
Bu zamanlayıcı bloğunda IN yüksekken ET artmaya başlar. Artarken PT değerine ulaşırsa Q yüksek olur. IN düşük olursa ET sıfırlanır ve Q da düşük olur. IN yüksekken ET, PT değerine ulaşmadan düşük olursa ET sıfırlanır.

3.6.2. TOF (Timer Off-Delay)



Bu zamanlayıcı bloğunda IN yüksekken Q yüksek olur. IN düşük olduğunda ET artmaya başlar. Artarken PT değerine ulaşırsa Q düşük olur ve ET sıfırlanır. IN düşükken ET, PT değerine ulaşmadan IN yüksek olursa ET sıfırlanır.

3.6.3. TP (Timer-Pulse)



Bu zamanlayıcı bloğunda IN yüksekken Q yüksek olur, ET artmaya başlar ve PT değerine ulaşınca IN değerinden bağımsız olarak Q düşük olur. ET artarken PT değerine ulaşana kadar IN değerinin değişmesi ET ve Q değerlerini etkilemez.

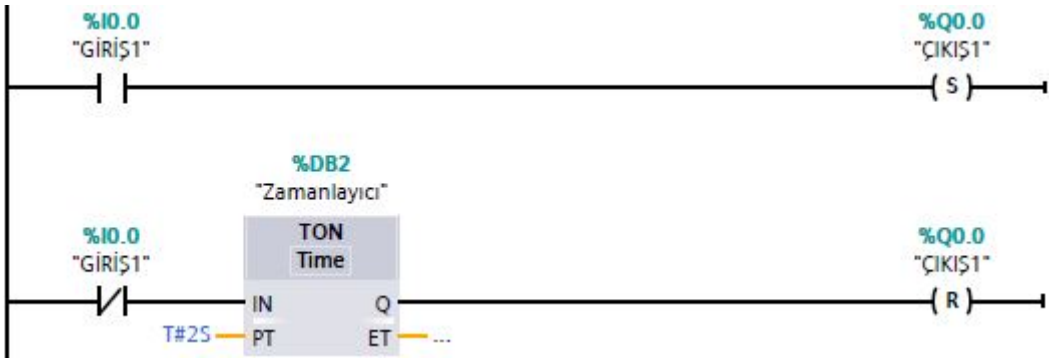
3.6.4. TONR (Timer On-Delay with Reset)

Bu zamanlayıcı bloğunda ek olarak aşağıdaki giriş bulunur:

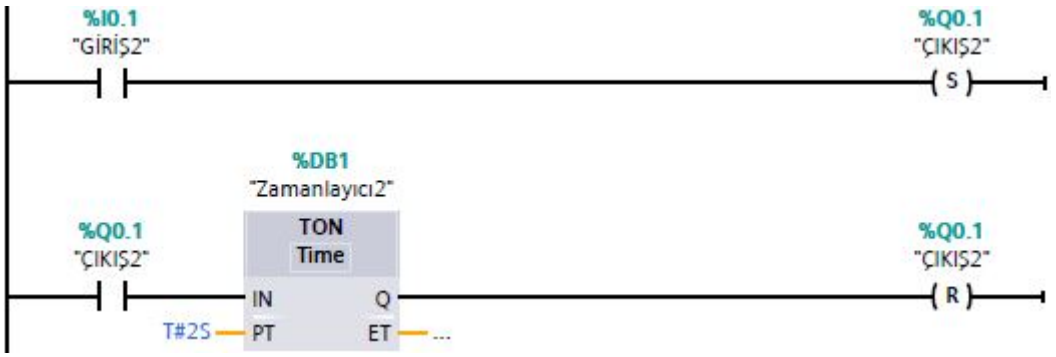
- **R (Reset) (Bool):** ET değerini sıfırlar.

Bu zamanlayıcı bloğu çalışma prensibi olarak TON bloğuna benzer. İki farkı IN düşükken ET değerinin sıfırlanmaması ve ET değerini sadece R değerinin sıfırlamasıdır. Böylece IN tekrar yüksek olduğunda ET değeri kaldığı yerden artmaya devam edebilecektir.

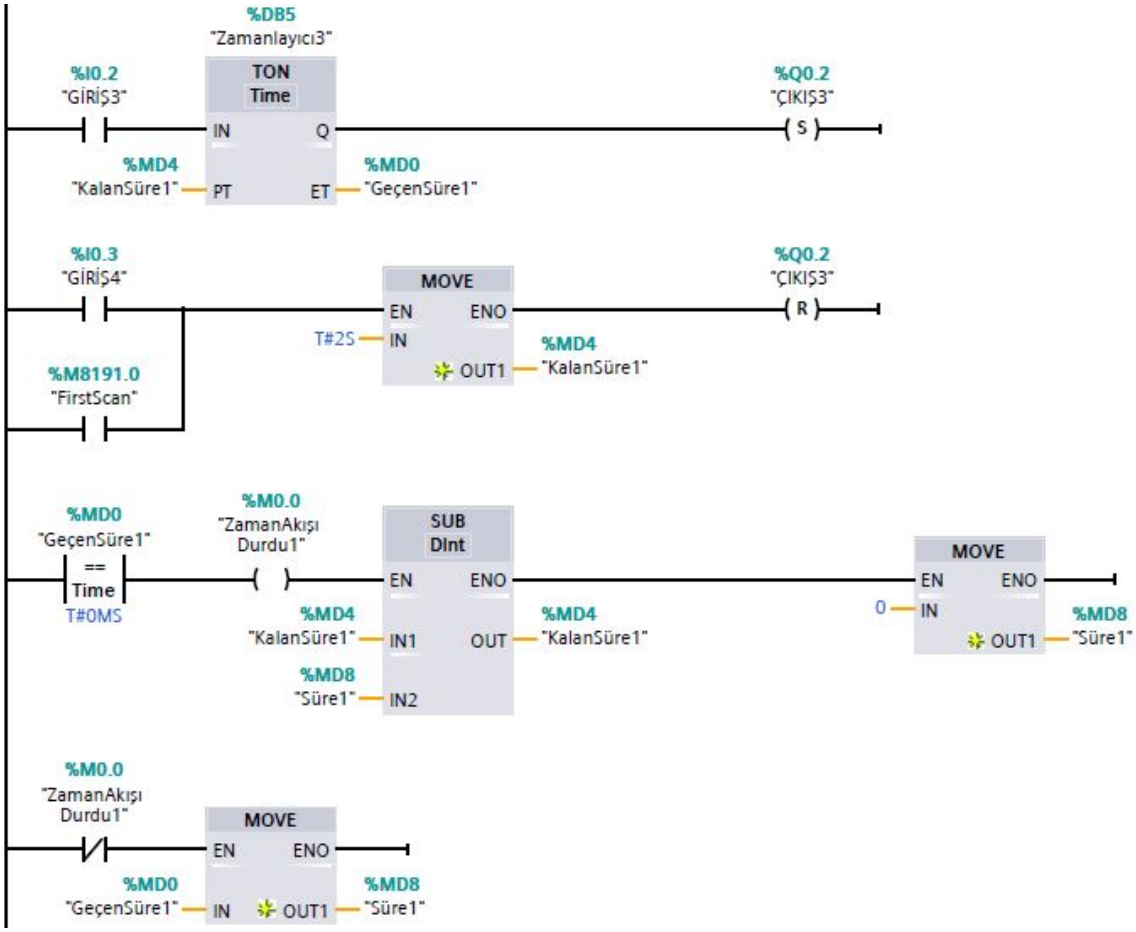
3.6.5. TON Bloğuyla Diğer Zamanlayıcı Bloklarının Tasarlanması



Kod Parçası 3.3. TON ile TOF tasarımı



Kod Parçası 3.4. TON ile TP tasarımı



Kod Parçası 3.5. TON ile TONR tasarımı

`FirstScan` sadece ilk taramada yüksek olup başlangıç değeri atamada kullanılır.

STAJDAN SORUMLU MÜHENDİS	Ünvan Ad ve Soyad	Doktor Öğretim Üyesi Türker Türker	Firma İmza Kaşe	

3.7. Sayıcı Blokları

Sayıcı blokları, tam sayı değerlerini artıran ve/veya azaltan ve istenilen değere ulaştığında çıkış veren bloklardır. Sayısı 3 tane olup bunlar aşağıdaki gibidir.

- CTU (Up Counter)
- CDU (Down Counter)
- CTUD (Up Down Counters)

Bu sayıcılarda ortak olan 2'şer giriş ve çıkış bulunur. Bunlar aşağıdaki gibidir:

- **PV (Preset Value) (Int):** Sayıcının çıkış vereceği değer.
- **Q (Output) (Bool):** Sayıcının çıkışıdır.
- **CV (Current Value) (Int):** Sayıcının mevcut değeridir.

3.7.1. CTU (Up Counter)

Yukarı yönde sayan sayıcıdır. CV >= PV olduğunda çıkış verir. Bu sayıcının girişi aşağıdaki gibidir:

- **CU (Count Up) (Bool):** Yükselen kenar algılayınca CV değerini artar.
- **R (Reset) (Bool):** Sayıcının CV değerini 0'lar.

```
IF R THEN
    CV := 0;
ELSIF CU THEN
    CV := CV + 1;
END_IF;
    Q := (CV >= PV);
```

Kod Parçası 3.6. CTU bloğunun SCL ile yazılmış hali

3.7.2. CTD (Down Counter)

Aşağı yönde sayan sayıcıdır. CV == 0 olduğunda çıkış verir. Bu sayıcının girişi aşağıdaki gibidir:

- **CD (Count Down) (Bool):** Yükselen kenar algılayınca CV değeri azalır.
- **LD (Load) (Bool):** Sayıcının CV değerini PV değeri yapar.

```
IF LD THEN
    CV := PV;
ELSIF CD THEN
    CV := CV - 1;
END_IF;
    Q := (CV <= 0);
```

Kod Parçası 3.7. CTD bloğunun SCL ile yazılmış hali

3.7.3. CTUD (Up Down Counter)

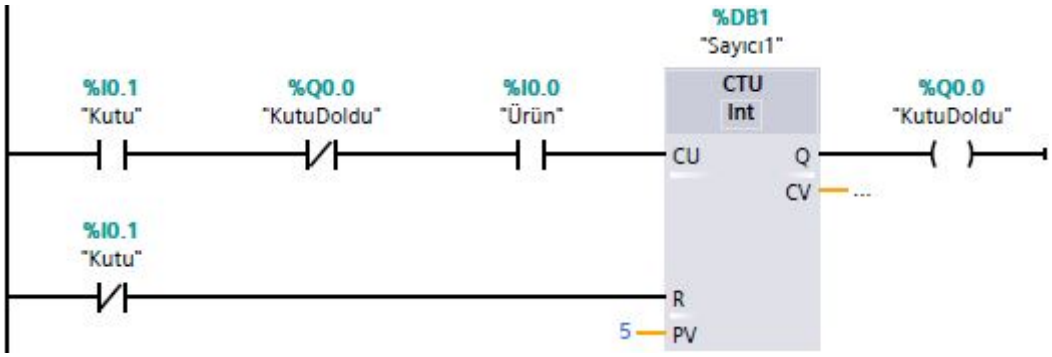
Diğer 2 sayacın birleştiği sayaçtır. Ortak giriş ve çıkışlara ek diğer sayaçlara özel girişler de sayaçta bulunur. Hem yukarı hem de aşağı sayılmak istendiğinde tek blok olarak kullanılabilir. 2 yön için ayrı 2 çıkış bulunur.

```
IF R THEN
    CV := 0;
ELSIF LD THEN
    CV := PV;
ELSE
    IF NOT (CU AND CD) THEN
        IF CU THEN
            CV := CV + 1;
        ELSIF CD THEN
            CV := CV - 1;
        END_IF;
    END_IF;
END_IF;
QU := (CV >= PV);
QD := (CV <= 0);
```

Kod Parçası 3.8. CTUD bloğunun SCL ile yazılmış hali

3.7.4. CTU Bloğuyla Algoritma Örneği

Banttan gelen ürünlerin kutuya düştüğü bir sistem düşünelim. Ürünlerin düştüğü kutunun ürün kapasitesi 5 olsun. Kutu dolduğunda bant duracak, yeni kutu koyulduğunda bant harekete başlayacak.



Kod Parçası 3.9. Algoritmanın LAD ile yazılmış hali

Eğer kutu varsa ve dolu değilse ürün geldikçe sayıcı sayacaktır. Kutu kapasitesine ulaşıldığında kutu dolmuş olacak ve daha fazla sayım olmayacaktır. Kutu alınıp yenisi koyulduğunda ürün geldikçe sayma işlemi yine kutu kapasitesine kadar devam edecektir.

3.8. Analog Girişler

PLC lojik bir kontrolcü olsa da takılabilecek modüllerle analog sinyallerle de işlem yapabilir hale gelir.

Dijital bir sinyal ya vardır ya yoktur, bu dijital sinyalin varlığı `1 (DOĞRU/YÜKSEK)` ile ifade edilirken yokluğu `0 (YANLIŞ/DÜŞÜK)` ile ifade edilir. Diğer bir ifadeyle alabileceği değer kesiklidir. Örneğin bir mesafe sensörü önündeki cismi ya görür ya da görmez.

Analog bir sinyalse belirli bir aralık arasında değişen değer alır, süreklidir. Örneğin bir sıcaklık sensörü, ölçtüğü sıcaklıkla orantılı bir gerilim çıkışı verir.

3.8.1. ADC ve DAC İşlemleri

PLC ise 0-10V arasındaki gerilimi ADC (Analog to Digital Convertion) işlemiyle 0-27648 aralığındaki tam sayıya dönüştürür. Burada 0V, 0 ile; 10V, 27468 ile temsil edilir. Hesaplanan tam sayıdan voltaj değerine geçiş için aşağıdaki algoritma kullanılır:

```
(Voltaj_ÜstSınır – Voltaj_AltSınır) * Hesaplanan_TamSayı
---------------------------------------------------------
          TamSayı_ÜstSınır – TamSayı_AltSınır
```

Kod Parçası 3.9. DAC (Digital to Analog Convertion) algoritması

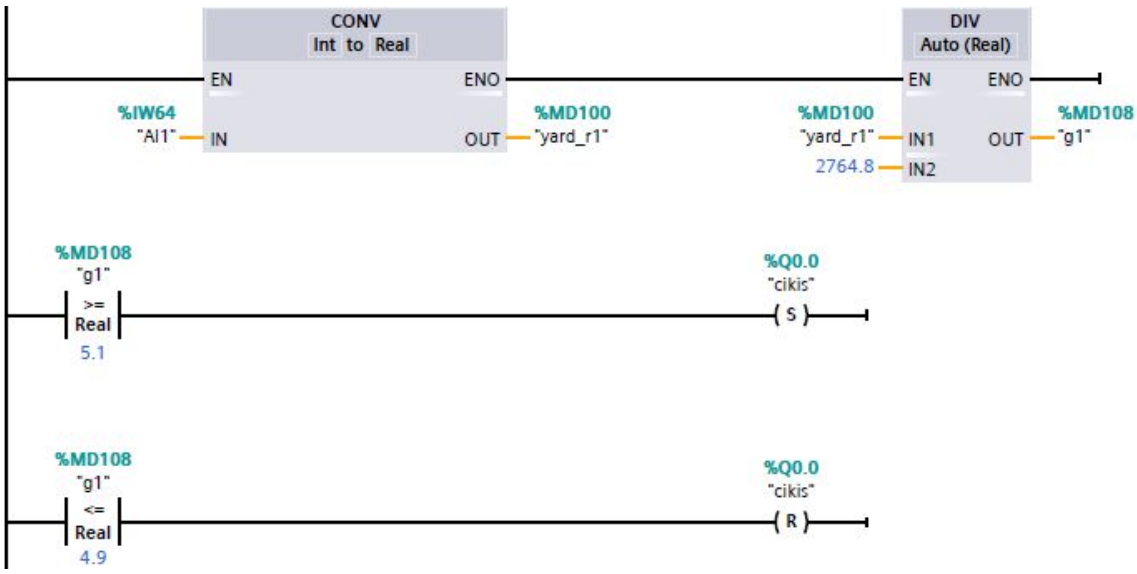
3.8.2. Çözünürlük

PLC'nin yaptığı ADC işlemi belirli bir hataya sahiptir ve buna çözünürlük denir. Çözünürlük, gerilim aralığı ne kadar dar ve tam sayı aralığı ne kadar geniş olursa o kadar yüksektir. Örneğin aynı gerilim aralığında 12 bitlik bir ADC işleminde çözünürlük `10/(2^12 – 1) = ~2,44mV` olurken 16 bitlik bir ADC işleminde çözünürlük `10/(2^16 – 1) = ~0,15mV` olur. Bu 4 bitlik artış `~2,28mV` daha hassasiyet kazandırır. Çözünürlükten düşük gerilim değişikliği ADC işleminde ayrıklaşamayacaktır.

3.8.3. Algoritma Örneği

0-10V aralığındaki kararsız gerilim PLC ile ölçülsün. Ölçümü filtrelemek ve çıkışı kararlaştırmak amacıyla ölçüm >5,1 ise yüksek, <4,9 ise düşük çıkış verilsin. [4,9 5,1] aralığındaysa çıkış değişmesin.

PLC bu sinyali tam sayıya dönüştürerek (ADC) okuyacaktır. Öncelikle yapılması gereken bu tam sayının temsil ettiği gerilim karşılığını hesaplamaktır. Bu hesaplama (DAC) yapıldıktan sonra hesaplanan değer >5,1 ise çıkış yüksek, <4,9 ise çıkış düşük olacak; bu aralıkta ise eski halini koruyacaktır.



Kod Parçası 3.10. Analog sinyali filtreleme algoritması

3.9. Artımsal Enkoder

Artımsal enkoder doğrusal veya dairesel hareketi elektrik sinyaline dönüştürür. 3 kanallı olan artımsal enkoderin bu kanalları A, B ve Z kanalları olarak adlandırılır. Kabaca yarık sayılarına göre ayrılırlar. Örneğin 500 yarıklı bir artımsal enkoder, tur başına bu sayıda pulse üretir. Buna göre pulse başına 0,72 derece döner. 0,72 dereceden az dönüşleriyse algılayamaz.

A ve B kanallarından 90 derece faz farkıyla pulse üretilirken Z kanalından tur sayısı elde edilir. Eğer sadece devirle ilgileniliyorsa tek kanaldan ölçüm yeterliyken yönle de ilgileniliyorsa çift kanaldan ölçüm yapılmalıdır.

Pulse sayarken normal sayıcıdan ziyade HSC (Yüksek Hızlı Sayıcı) kullanılmalıdır.