Bu kitaba sığmayan daha neler var!

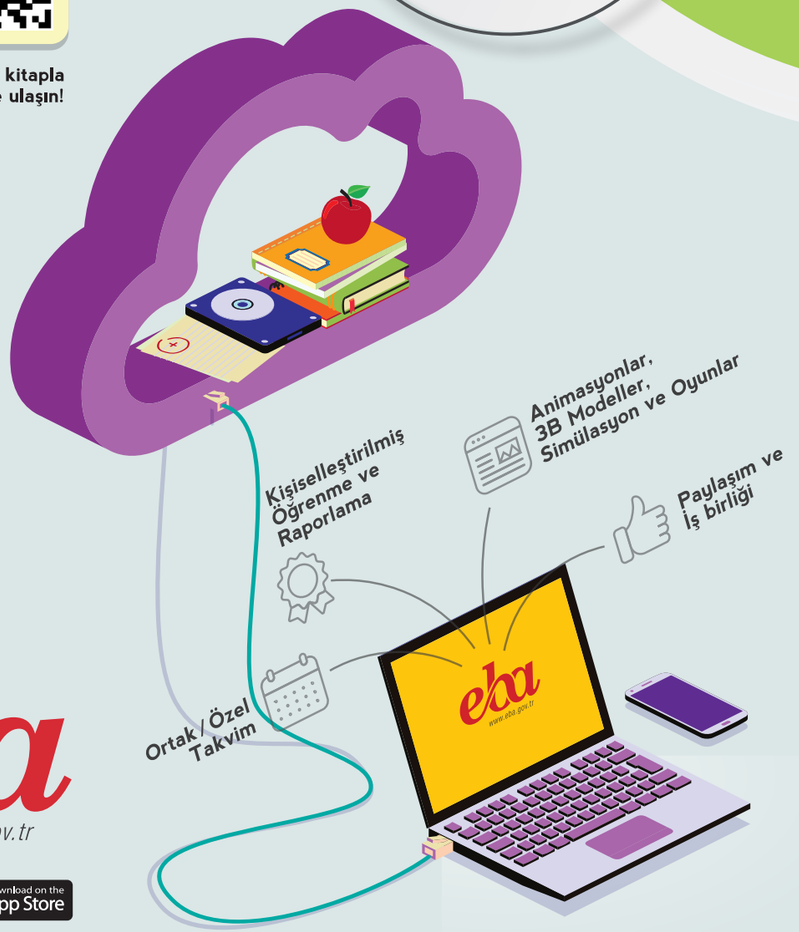
Karekodu okutun, bu kitapla ilgili EBA içeriklerine ulaşın!

ÖDS

ÖĞRENCİ/ÖĞRETMEN DESTEK SİSTEMİ

https://ods.eba.gov.tr

- Konu Anlatımlı Ders Videoları
- Soru Çözüm Videoları
- Ders Anlatım Videoları
- Çoktan Seçmeli Sorular



Google play | App Store

40181 700982

BU DERS KİTABI MİLLÎ EĞİTİM BAKANLIĞINCA ÜCRETSİZ OLARAK VERİLMİŞTİR.
PARA İLE SATILAMAZ.

ISBN: 978-975-11-6416-2

Bandrol Uygulamasına İlişkin Usul ve Esaslar Hakkında Yönetmelik'in 5'inci Maddesinin İkinci Fıkrası Çerçevesinde Bandrol Taşıması Zorunlu Değildir.

İLERİ PLC UYGULAMALARI DERS MATERYALİ

MESLEKİ VE TEKNİK ANADOLU LİSESİ

İLERİ PLC UYGULAMALARI

DERS MATERYALİ

MESLEKİ VE TEKNİK ANADOLU LİSESİ

İLERİ PLC UYGULAMALARI

DERS MATERYALİ

YAZARLAR
Mehmet GÖVERDİK
Mehmet AŞIK

MİLLÎ EĞİTİM BAKANLIĞI YAYINLARI 8358
YARDIMCI VE KAYNAK KİTAPLAR DİZİSİ 2250

HAZIRLAYANLAR

Dil Uzmanı | Tayfun KARTAL
Görsel Tasarım Uzmanı | Birgül AKBABA

ISBN: 978-975-11-6416-2

Millî Eğitim Bakanlığının 24.12.2020 gün ve 18433886 sayılı oluru ile Mesleki ve Teknik Eğitim Genel Müdürlüğünce ders materyali olarak hazırlanmıştır.

İSTİKLÂL MARŞI

Korkma, sönmez bu şafaklarda yüzen al sancak;
Sönmeden yurdumun üstünde tüten en son ocak.
O benim milletimin yıldızıdır, parlayacak;
O benimdir, o benim milletimindir ancak.

Çatma, kurban olayım, çehreni ey nazlı hilâl!
Kahraman ırkıma bir gül! Ne bu şiddet, bu celâl?
Sana olmaz dökülen kanlarımız sonra helâl.
Hakkıdır Hakk'a tapan milletimin istiklâl.

Ben ezelden beridir hür yaşadım, hür yaşarım.
Hangi çılgın bana zincir vuracakmış? Şaşarım!
Kükremiş sel gibiyim, bendimi çiğner, aşarım.
Yırtarım dağları, enginlere sığmam, taşarım.

Garbın âfâkını sarmışsa çelik zırhlı duvar,
Benim iman dolu göğsüm gibi serhaddim var.
Ulusun, korkma! Nasıl böyle bir imanı boğar,
Medeniyyet dediğin tek dişi kalmış canavar?

Arkadaş, yurduma alçakları uğratma sakın;
Siper et gövdeni, dursun bu hayâsızca akın.
Doğacaktır sana va'dettiği günler Hakk'ın;
Kim bilir, belki yarın, belki yarından da yakın.

Bastığın yerleri toprak diyerek geçme, tanı:
Düşün altındaki binlerce kefensiz yatanı.
Sen şehit oğlusun, incitme, yazıktır, atanı:
Verme, dünyaları alsan da bu cennet vatanı.

Kim bu cennet vatanın uğruna olmaz ki feda?
Şüheda fışkıracak toprağı sıksan, şüheda!
Cânı, cânânı, bütün varımı alsın da Huda,
Etmesin tek vatanımdan beni dünyada cüda.

Ruhumun senden İlâhî, şudur ancak emeli:
Değmesin mabedimin göğsüne nâmahrem eli.
Bu ezanlar -ki şehadetleri dinin temeli-
Ebedî yurdumun üstünde benim inlemeli.

O zaman vecd ile bin secde eder -varsa- taşım,
Her cerîhamdan İlâhî, boşanıp kanlı yaşım,
Fışkırır ruh-ı mücerret gibi yerden na'şım;
O zaman yükselerek arşa değer belki başım.

Dalgalan sen de şafaklar gibi ey şanlı hilâl!
Olsun artık dökülen kanlarımın hepsi helâl.
Ebediyyen sana yok, ırkıma yok izmihlâl;
Hakkıdır hür yaşamış bayrağımın hürriyyet;
Hakkıdır Hakk'a tapan milletimin istiklâl!

Mehmet Âkif Ersoy

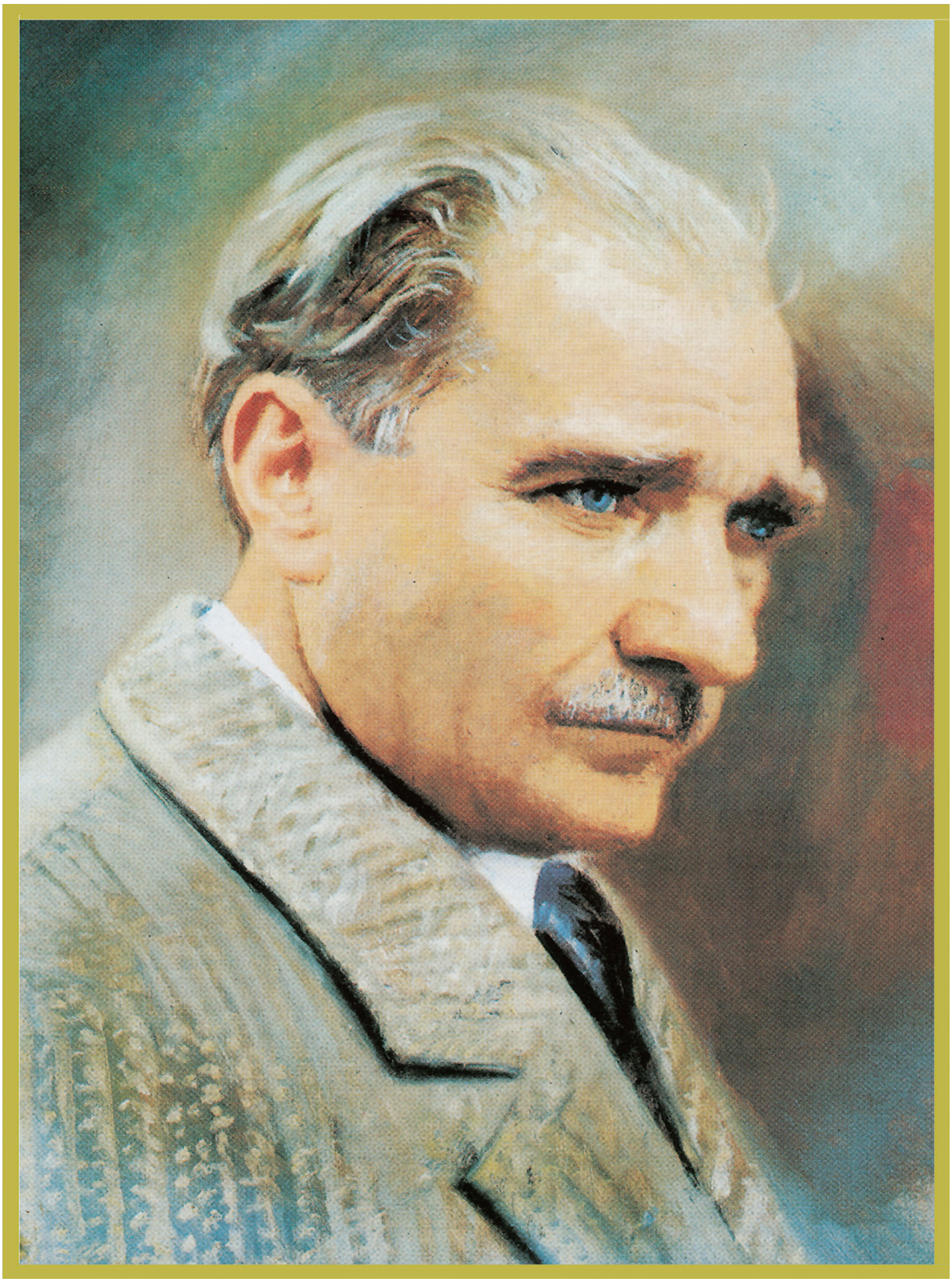
GENÇLİĞE HİTABE

Ey Türk gençliği! Birinci vazifen, Türk istiklâlini, Türk Cumhuriyetini, ilelebet muhafaza ve müdafaa etmektir.

Mevcudiyetinin ve istikbalinin yegâne temeli budur. Bu temel, senin en kıymetli hazinendir. İstikbalde dahi, seni bu hazineden mahrum etmek isteyecek dâhilî ve hâricî bedhahların olacaktır. Bir gün, istiklâl ve cumhuriyeti müdafaa mecburiyetine düşersen, vazifeye atılmak için, içinde bulunacağın vaziyetin imkân ve şeraitini düşünmeyeceksin! Bu imkân ve şerait, çok namüsait bir mahiyette tezahür edebilir. İstiklâl ve cumhuriyetine kastedecek düşmanlar, bütün dünyada emsali görülmemiş bir galibiyetin mümessili olabilirler. Cebren ve hile ile aziz vatanın bütün kaleleri zapt edilmiş, bütün tersanelerine girilmiş, bütün orduları dağıtılmış ve memleketin her köşesi bilfiil işgal edilmiş olabilir. Bütün bu şeraitten daha elîm ve daha vahim olmak üzere, memleketin dâhilinde iktidara sahip olanlar gaflet ve dalâlet ve hattâ hıyanet içinde bulunabilirler. Hattâ bu iktidar sahipleri şahsî menfaatlerini, müstevlîlerin siyasî emelleriyle tevhit edebilirler. Millet, fakr u zaruret içinde harap ve bîtap düşmüş olabilir.

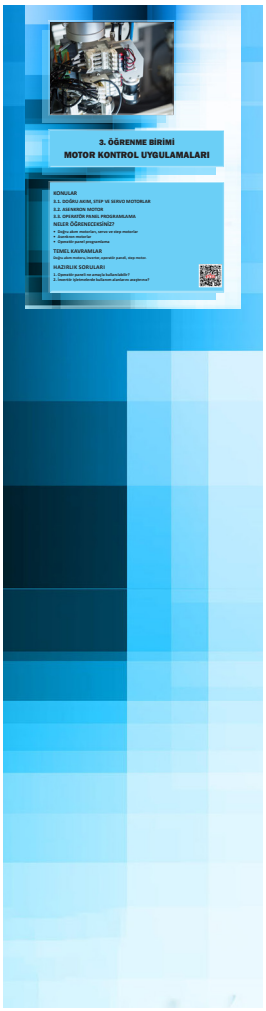
Ey Türk istikbalinin evlâdı! İşte, bu ahval ve şerait içinde dahi vazifen, Türk istiklâl ve cumhuriyetini kurtarmaktır. Muhtaç olduğun kudret, damarlarındaki asil kanda mevcuttur.

Mustafa Kemal Atatürk



MUSTAFA KEMAL ATATÜRK

İÇİNDEKİLER



DERS MATERYALİNİN TANITIMI

Öğrenme biriminin kapağını gösterir.

Öğrenme biriminin numarasını ve adını gösterir.

Öğrenme biriminin konularını gösterir.

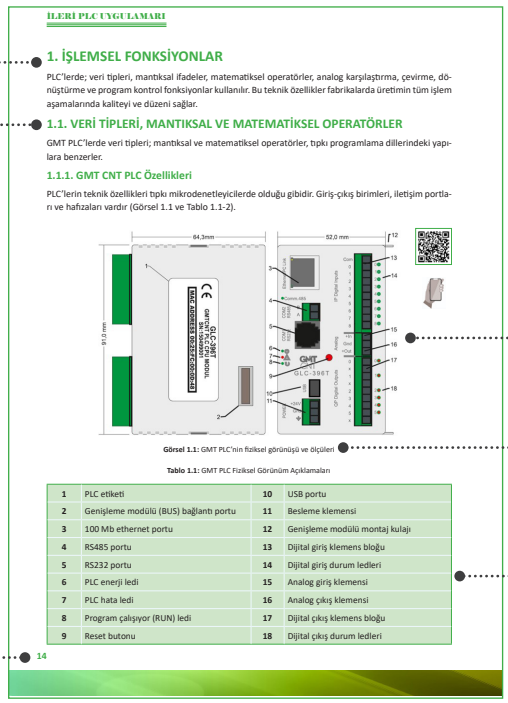
Öğrenme biriminin hazırlık sorularını gösterir.

Öğrenme biriminde neler öğrenileceğini gösterir.

Etkileşimli kitap, video, ses, animasyon, uygulama, oyun, soru vb. ilave kaynaklara ulaşılabilecek karekodu gösterir.

Öğrenme birimiyle ilgili anahtar kavramları gösterir.

Konu başlığını gösterir.



Konuyla ilgili alt başlığı gösterir.

Konuyla ilgili görseli gösterir.

Konuyla ilgili görselin adını ve numarasını gösterir.

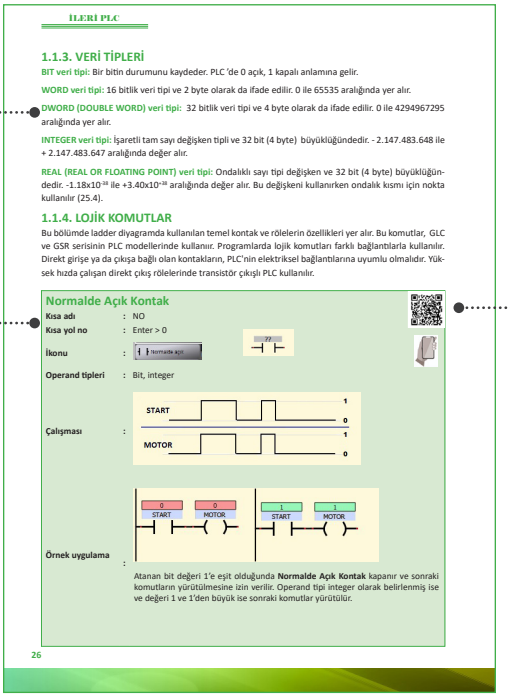
Konuyla ilgili tablo içeriği ve numarasını gösterir.

Sayfa numaralarını gösterir.

Kazanımlara uygun olarak hazırlanan içeriklerin bulunduğu alandır.

Konuyla ilgili komutların anlatıldığı şemadır.

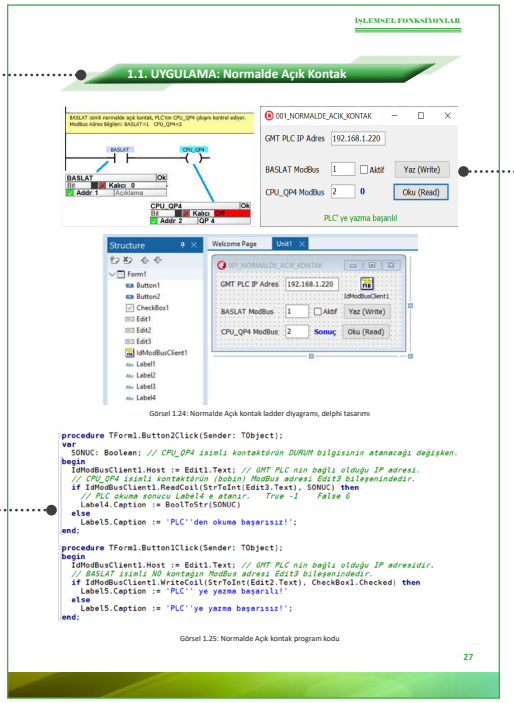
Etkileşimli kitap, video, ses, animasyon, uygulama, oyun, soru vb. ilave kaynaklara ulaşılabilecek karekodu gösterir.



Konuyla ilgili uygulama numarasını ve adını gösterir.

Uygulamayla ilgili program kodunu gösterir.

Uygulamayla ilgili program arayüzlerini gösterir.



* Bu ders materyalinde ölçü birimlerinin uluslararası kısaltmaları kullanılmıştır.

1. ÖĞRENME BİRİMİ
İŞLEMSEL FONKSİYONLAR

KONULAR

1.1. VERİ TİPLERİ, MANTIKSAL VE MATEMATİKSEL OPERATÖRLER

1.2. ANALOG VE KARŞILAŞTIRMA İŞLEMLERİ, ÇEVİRME İŞLEMLERİ

1.3. KAYDIRMA, DÖNDÜRME, TAŞIMA, PROGRAM KONTROL FONKSİYONLARI

NELER ÖĞRENECEKSİNİZ?

- GMT PLC teknik özellikleri
- GMT Suite programının kullanımı
- GMT PLC Ladder komutlarının kullanımı

TEMEL KAVRAMLAR

Analog işlem, döndürme, kaydırma, mantıksal operatör, matematiksel operatör, taşıma, veri tipi.

HAZIRLIK SORULARI

1. PLC'lerde bulunan analog çıkışların kullanılabileceği cihaz ya da makinalar neler olabilir?
2. Bir adet PLC ile en fazla kaç cihaz kontrol edilebilir?

1. İŞLEMSEL FONKSİYONLAR

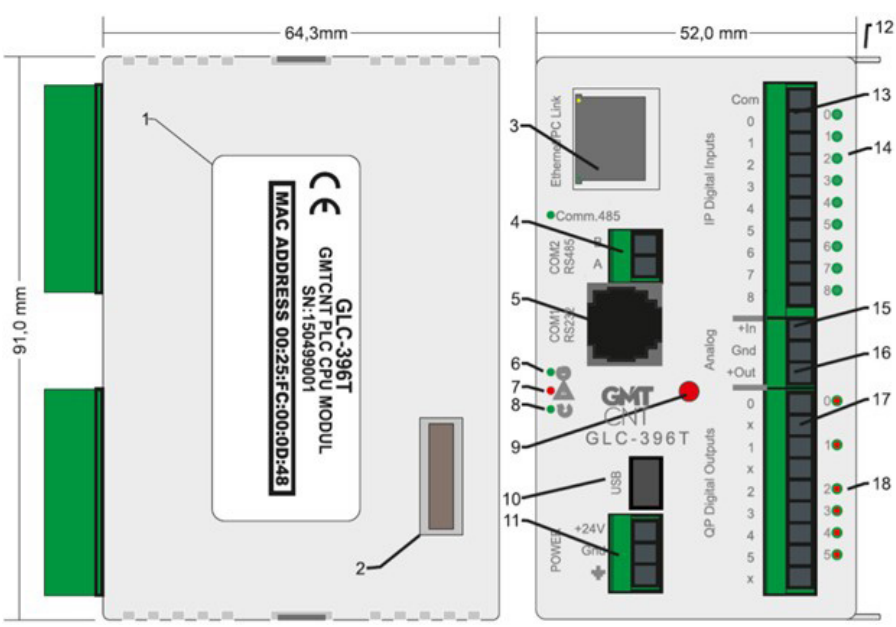
PLC'lerde veri tipleri, mantıksal ifadeler, matematiksel operatörler, analog karşılaştırma, çevirme, dönüştürme ve program kontrol fonksiyonlar kullanılır. Bu teknik özellikler fabrikalarda üretimin tüm işlem aşamalarında kaliteyi ve düzeni sağlar.

1.1. VERİ TİPLERİ, MANTIKSAL VE MATEMATİKSEL OPERATÖRLER

GMT PLC'lerde veri tipleri; mantıksal ve matematiksel operatörler, tıpkı programlama dillerindeki yapılara benzerler.

1.1.1. GMT CNT PLC Özellikleri

PLC'lerin teknik özellikleri tıpkı mikrodenetleyicilerde olduğu gibidir. Giriş-çıkış birimleri, iletişim portları ve hafızaları vardır (Görsel 1.1 ve Tablo 1.1-2).



Görsel 1.1: GMT PLC'nin fiziksel görünüşü ve ölçüleri

Tablo 1.1: GMT PLC Fiziksel Görünüm Açıklamaları

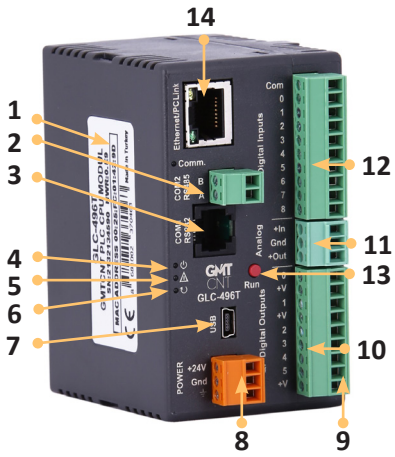
1	PLC etiketi	**10**	USB portu
2	Genişleme modülü (BUS) bağlantı portu	**11**	Besleme klemensi
3	100 Mb ethernet portu	**12**	Genişleme modülü montaj kulajı
4	RS485 portu	**13**	Dijital giriş klemens bloğu
5	RS232 portu	**14**	Dijital giriş durum ledleri
6	PLC enerji ledi	**15**	Analog giriş klemensi
7	PLC hata ledi	**16**	Analog çıkış klemensi
8	Program çalışıyor (RUN) ledi	**17**	Dijital çıkış klemens bloğu
9	Reset butonu	**18**	Dijital çıkış durum ledleri

Tablo 1.2: GMT CNT PLC Teknik Özellikleri

TEKNİK ÖZELLİKLER		
Özellikler	**Bölüm**	**Açıklama**
Genel	Besleme	24VDC, ±%15 tolerans
	Güç	Max 3W @ 24VDC
	CPU İşlem hızı	12ns/komut, 220MIPS Arm CortexM4F 168MHz
	Program dili	GMTSuite editör yazılımı ile merdiven (Ladder) metodu
	İşlemci hafizası	512kB
	RTC (gerçek zaman saati)	Saat ve tarih bilgisine göre program yazabilme imkânı
	İşlemler	Lojik, matematiksel, haberleşme, hızlı sayıcı, hızlı puls çıkışları, zamanlayıcılar ve özel fonksiyon blokları ile işlemler yapabilme
İletişim Portları	Ethernet portu	100MB ethernet portu üzerinden programlama, link kurma, MODBUS TCP Slave desteği, WMI teknolojisi ile internet üzerinden cihaza erişim imkânı
	USB portu	Bu girişten dönüştürücü yardımı ile USB flash belleğe kayıt imkânı, program yükleme imkânı
	RS232	ASCII veya MODBUS RTU protokolü ile 4800...115200 bps hız desteği
	RS485	ASCII veya MODBUS RTU protokolü ile 4800...115200 bps hız desteği
Giriş/Çıkış	Dijital çıkışlar	6 Adet Transistör Çıkışı 24VDC@300mA (Hızlı pulse çıkışları için çıkış frekansı max. 500kHz) / 6 Adet Röle Çıkışı 230VAC@5A, max. (İzolasyon voltajı 10kV)
	Dijital girişler	9 adet 24VDC pnp/npn girişi, 3 adet çift faz enkoder bağlanabilir, okuma frekansı max. 200kHz, max. izolasyon voltajı 1dk. için 3750VAC RMS
	Analog giriş	1 Adet 0...10VDC / 0...20mA / 4..20mA analog giriş. 12 bit çözünürlük
	Analog çıkış	1 adet 0...20mA analog çıkış. 14 Bit çözünürlük
Hafiza Kullanımı	Program boyutu	196 Kbyte
	Operant kullanımı	10 Kbyte
Çalışma Ortamı	Sıcaklık	0..+50°C çalışma aralığı (buzlanma olmadan)
	Nem	5..95%rH Nem çalışma aralığı kullanılmalıdır.
	Ortam	Yanıcı ve aşındırıcı gaz bulunmayan ortamlarda kullanılmalıdır.
PLC CPU modülleri 2 ana grupta 4 seriye ayrılır. Bunların genel özellikleri aşağıda belirtilmiştir.		
GLC – 196R	9 Dijital giriş 20kHz sayıcı, 6 röle çıkışı	
GLC – 196T	9 Dijital giriş 20kHz sayıcı, 6 transistör çıkış (3 kanal 20kHz)	
GLC – 296R	9 Dijital giriş 50kHz sayıcı, 6 röle çıkışı, 1 analog giriş, 1 analog çıkış	
GLC – 296T	9 Dijital giriş 50kHz sayıcı, 6 transistör çıkış (3 kanal 100kHz), 1 analog giriş, 1 analog çıkış	
GLC – 396R	9 Dijital giriş 50kHz sayıcı, 6 röle çıkışı, 1 analog giriş, 1 analog çıkış, WMI, RTC, ModBus TCP	
GLC – 396T	9 Dijital giriş 50kHz sayıcı, 6 transistör çıkış (3 kanal 100kHz), 1 analog giriş, 1 analog çıkış, WMI, RTC, ModBus TCP	
GLC – 496R	9 Dijital giriş 200kHz sayıcı, 6 röle çıkışı, 1 analog giriş, 1 analog çıkış, WMI, RTC, ModBus TCP	
GLC – 496T	9 Dijital giriş 200kHz sayıcı, 6 transistör çıkış (3 kanal 400kHz), 1 analog giriş, 1 analog çıkış, WMI, RTC, ModBus TCP	

Mekanik Özellikler

Cihaz, DIN ray montajlıdır. Konfigürasyonda genişleme modülü varsa modüller birbirine, montajı bittikten sonra raya monte edilir. Modül arkasında bulunan geçmeli klipsi ile raya sabitlenir (Görsel 1.2).



1	PLC etiketi (MAC adresi buradan okunur.)	8	+24 V besleme klemensi
2	RS485 portu	9	Genişleme modülü BUS bağlantı portu
3	RS232 portu	10	Dijital çıkış klemens bloğu
4	PLC enerji ledi	11	Analog giriş-çıkış klemensi
5	PLC hata ledi	12	Dijital giriş klemens bloğu
6	PLC RUN ledi	13	Reset butonu
7	USB portu	14	100Mb ethernet portu

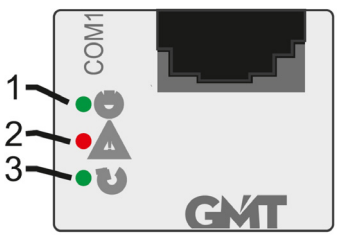
Görsel 1.2: PLC'ler için mekanik özellikler

LED Durumları

1. Enerji Ledi: PLC24V DC kaynak bağlantısı varsa yanar.

2. Hata Ledi: Genellikle yanmaz, ilk enerji verildiğinde 0,5 sn. yanıp söner. Yazılım ya da donanım hatası varsa periyodik olarak yanıp söner. Genişleme modül konfigürasyonu yanlış ise (eşleşme hatası varsa) 3 defa art arda periyodik olarak yanıp söner. Transistör çıkışlarından en az biri kısa devre ise devamlı yanar.

3. PLC RUN Ledi: Program çalışmıyorsa 1 sn. yanar ve 1 sn. söner. Program çalışıyorsa 0,25 sn. yanar ve 0,25 sn. söner.

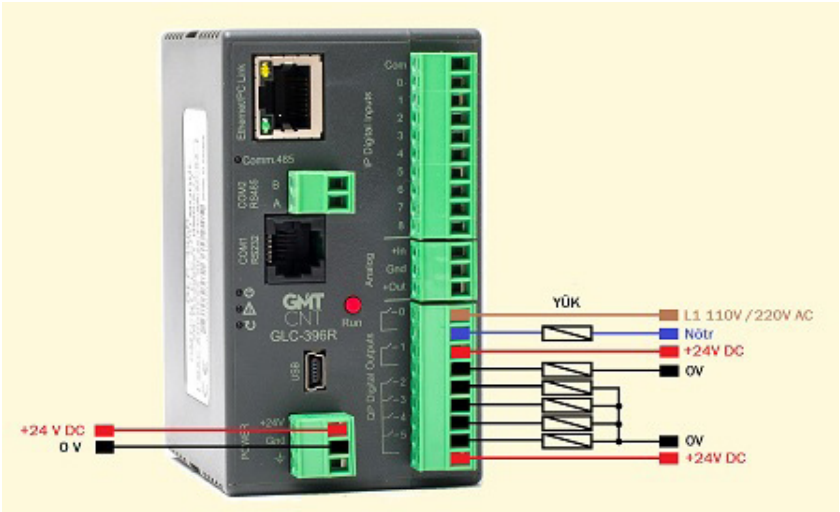


Görsel 1.3: PLC'nin LED durumları

Dijital Çıkış Bağlantıları

PLC CPU'su üzerindeki On/Off çıkışlarıdır. CPU modeline göre röle tipi, transistör röle veya transistör çeşitleri vardır. Program içerisinde CPU_QP[X] operandı ile kumanda edilir (Görsel 1.4).

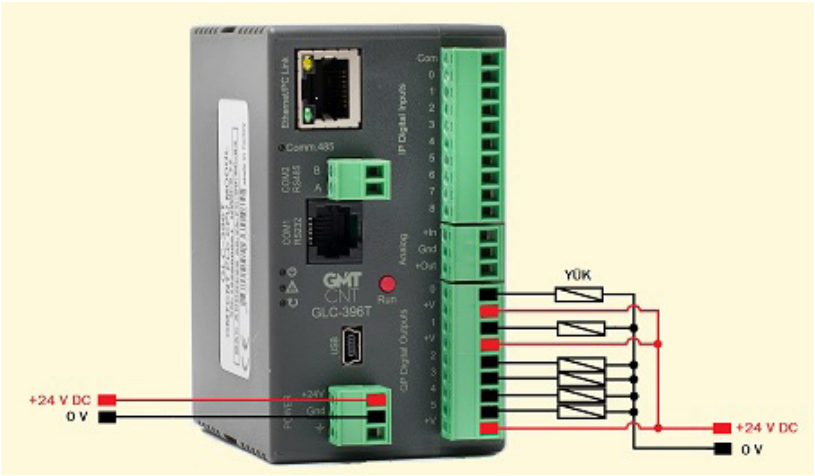
GLC-X96R (röle) Serisi İçin Çıkışlar: Çıkışlar 230V AC@5A dahili rölelerden oluşur. Röle çıkışları üç gurupta toplanır.



Görsel 1.4: Röleli PLC'lerin çıkış bağlantısı

CPU_QP[0] ve CPU_QP[1] ayrı ayrı tek başlarına; CPU_QP[2], CPU_QP[3], CPU_QP[4], CPU_QP[5] ise dörtlü guruba sahiptir. Röleler, kuru kontak olarak çalıştığı için akım sınırını geçmediği sürece AC veya DC bağlantılarına uygundur. Anahtarlama frekansı en fazla 100Hz'de tutulur. Maksimum voltaj değeri 270V AC / 125V DC'dir. Her çıkış için bir durum ledi vardır.

GLC-X96T (transistör) Serisi İçin Çıkışlar: Çıkışlar pnp tipi transistördür (Görsel 1.5). Bu çıkışlar direkt olarak röle, kontaktör, selonoid valf bobinlerine bağlanabilir (Görsel 1.5). Modele göre max. anahtarlama frekansı 20/100/400kHz 24V DC (max.) 450mA çıkışlıdır. Kısa devre koruması olduğundan aşırı akım anında PLC'ye uyarı verir. Transistör beslemesi dışarıdan yapılır. Bu beslenme 12 veya 24V DC aralığında olabilir. Hızlı çıkışlardan min. 50mA akım çekilmesi gerekir.



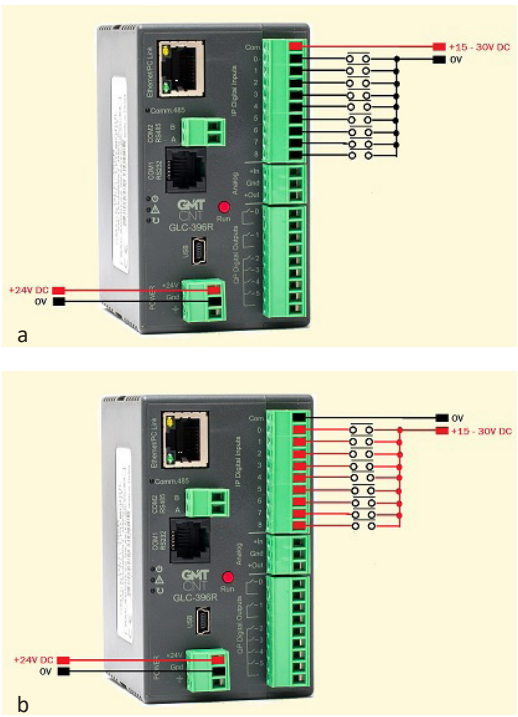
Görsel 1.5: Transistörlü PLC'lerin çıkış bağlantısı

Digital Giriş Bağlantıları

CPU üzerindeki On/Off girişleridir. Giriş tipi pnp veya npn olabilir (Görsel 1.6). Sayma hızı max. 200kHz'dır. Kısa puls girişlerine (0,15ms. genlik) duyarlıdır. GMTSuite editöründe filtreler kısmından duyarlık derecesi ayarlanabilir. Ortak uç (Gnd), CPU şasesinden ayrı olup optokuplör ile izole edilir.

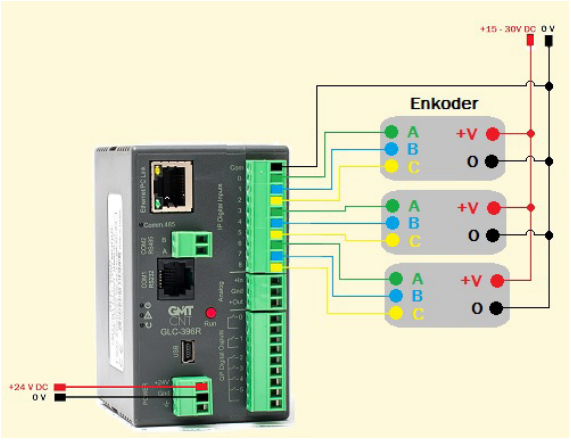
NPN Tipi Bağlantı: CPU'nun **IP DigitalInputs** terminallerini **NPN** tipi bağlamak için; COM ucuna, 24V DC güç kaynağının (+) hattını, **IP DigitalInputs** terminallerine ise kontak üzerinden (-) hattına bağlanır (Görsel 1. 6a).

PNP Tipi Bağlantı: CPU'nun **IP DigitalInputs** terminallerini **PNP** tipi bağlamak için; COM ucuna, 24V DC güç kaynağının (-) hattını, **IP DigitalInputs** terminallerine ise kontak üzerinden (+) hattına bağlanır (Görsel 1. 6b).

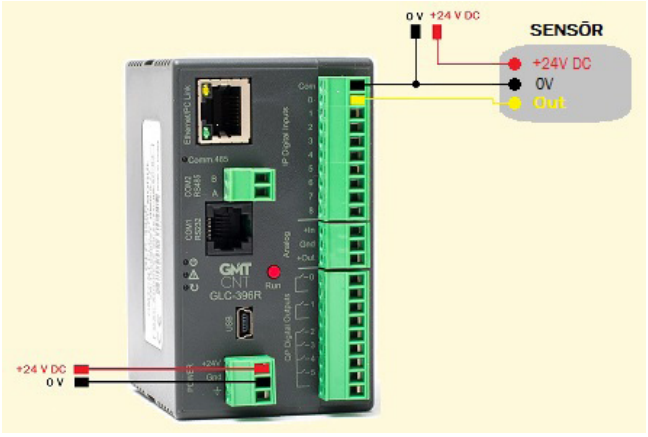


Görsel 1.6: NPN ve PNP tipli girişlerin bağlantıları

On/Off Giriş Tipi Bağlantı: Giriş kanalı ile COM arasında 24V DC potansiyel farkı var ise operand 1'dir. 18V DC altına düşerse operand değeri 0 olur (Görsel 1.7-8).



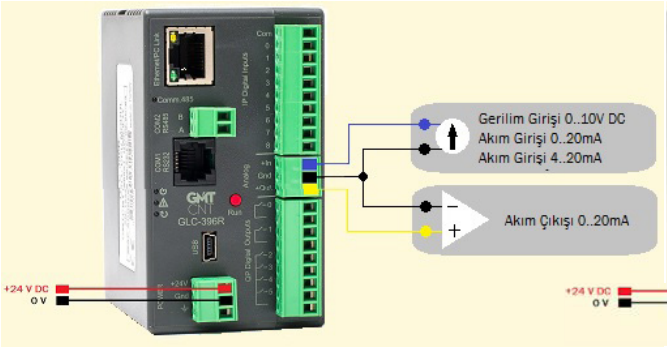
Görsel 1.7: PNP veya NPN bağlantılı çift fazlı üç encoder



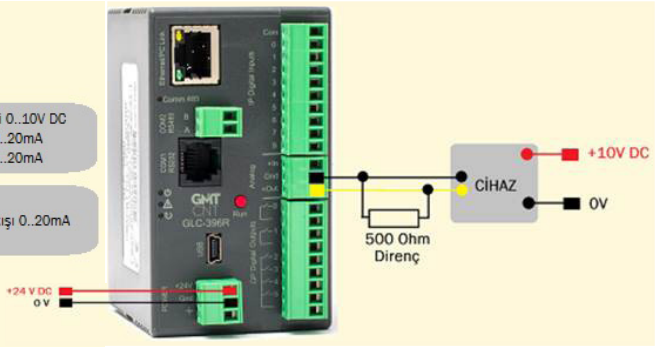
Görsel 1.8: Fotosel sensörlerin bağlantısı

Analog Giriş-Çıkış Bağlantıları

PLC, CPU'nun üzerindeki oransal giriş ve çıkışlardır. Giriş tipleri 0-10V, 0-20mA ve 4-20mADC'dir (Görsel 1.9). Çıkış tipi ise 0-20mADC'dir. Ayrıca istenildiğinde Görsel 1.10'daki gibi 500 ohm direnç kullanılarak 0-10V DC'ye dönüştürülür. Lineer olarak 0-16383 değere kadar sinyal üretir. Yineleme hızı PLC ladder tarama hızı ile aynıdır.



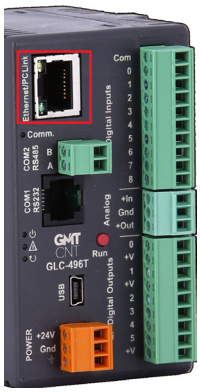
Görsel 1.9: 0-10 V analog giriş ve çıkış bağlantısı



Görsel 1.10: 0-10 V analog çıkışı veya 4-20 mA analog çıkış bağlantıları

Ethernet Bağlantısı

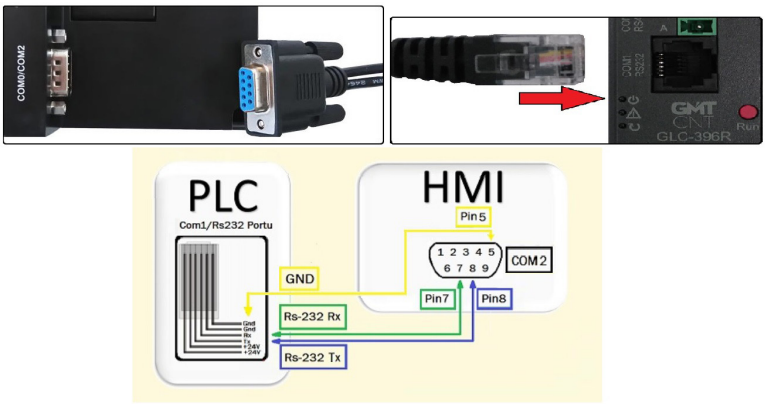
Ethernet portu, standart CAT5 kablo ile PLC'yi direkt PC'ye veya network ağına bağlar (Görse 1. 11). CPU üzerinde bulunan ethernet portunun kullanım amaçları, GMTSuite editöründen hazırlanan programın yüklendiği değerlerin izlenmesi ve değiştirilmesidir. MODBUS TCP slavemodunda, SCADA veya akıllı cihazlara bağlanır. WMI teknolojisi ile Internet üzerinden PLC'ye program yüklenebilir ve izlenebilir. PLC'nin kullanıldığı sistemde; otomasyonun yanı sıra üretim adedi, alarm gibi çeşitli durumları istenilen saatlerde elektronik posta olarak gönderilebilir (GLC 396X serilerinde).



Görsel 1.11: GMT PLC'nin ethernet port yuvası

COM1 - RS232 Portu

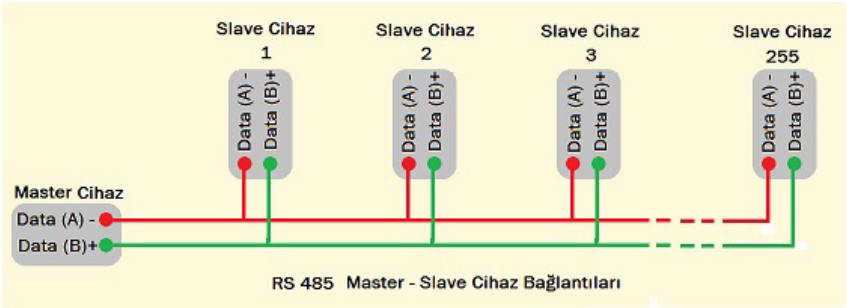
Master veya slave çalışabilen RS232 seri haberleşme portu, PLC CPU serisi ile beraber standart; host cihazlarına da kolaylıkla bağlantı imkânı sunar (Görsel 1. 12).



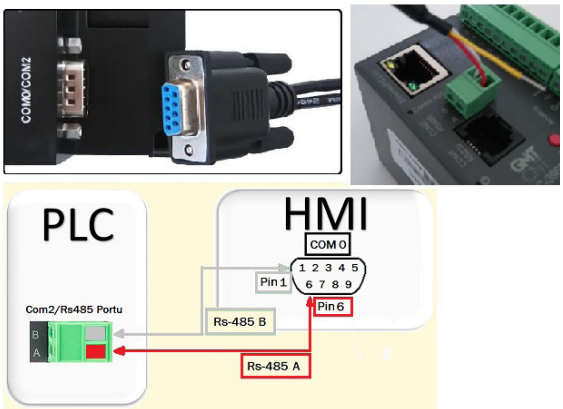
Görsel 1.12: GMT PLC'nin COM1 RS232 seri port yuvası

COM2 - RS485 Portu

Master veya slave çalışabilen RS485 seri haberleşme portu, PLC CPU serisi ile standart host cihazlarına da kolaylıkla bağlantı imkânı sunar (Görsel 1. 13). Bu tip konnektörlü bağlantı ile RS485 network bağlantısı kurmak mümkündür. RS485 bağlantı pini bulunan diğer PLC veya 3. parti cihazlardan 255 adede kadar MODBUS RTU protokolü üzerinden haberleşebilir (Görsel 1. 14). Opsiyonel olarak galvanik izolasyon kullanılır.



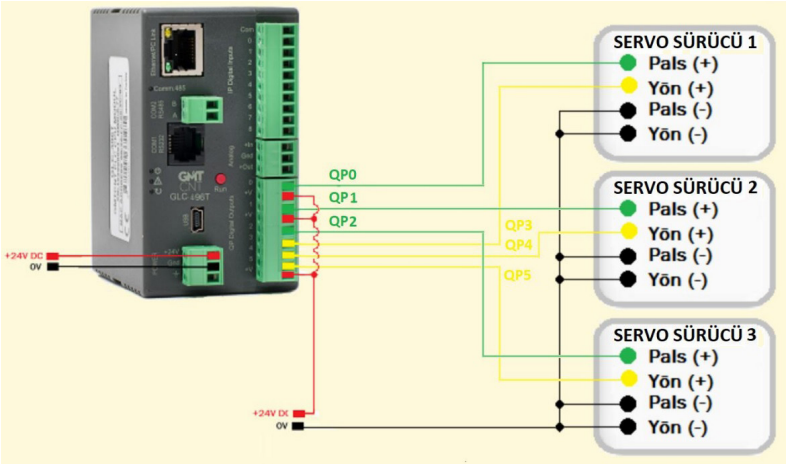
Görsel 1.13: RS485 ptotokolü ile MASTER ve SLAVE bağlantıları



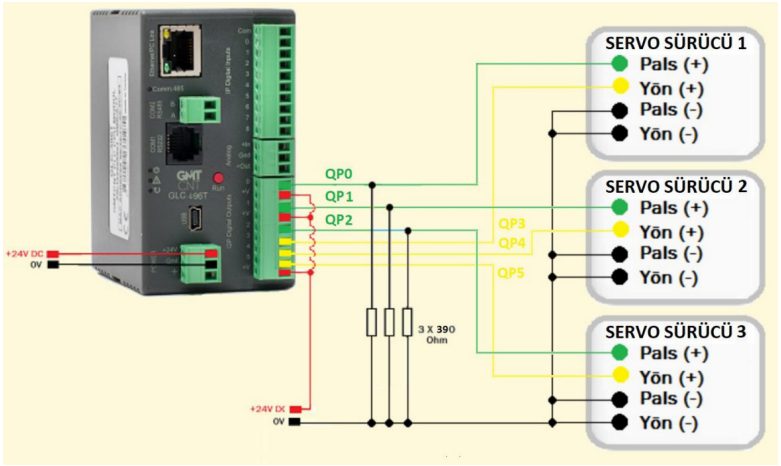
Görsel 1.14: GMT PLC'nin COM2 RS485 (A, B) port yuvası

Hızlı Pulse Çıkış Bağlantıları

Hızlı pals çıkışlarına bağlanan dirençler 390 ohm ve min. 2W olmalıdır. Yön çıkışları için direnç bağlantısına gerek yoktur. Farklı üretici firmaların servo sürücülerinin kullanılması durumunda, ilgili firmanın talimatları doğrultusunda direnç değerleri kullanılır (Görsel 1. 15-16).



Görsel 1.15: GMT PLC'nin (196T, 296T ve 396T modelleri için) hızlı çıkış bağlantısı



Görsel 1.16: GMT PLC'nin (model 496T) hızlı çıkış bağlantısı

Reset Butonu

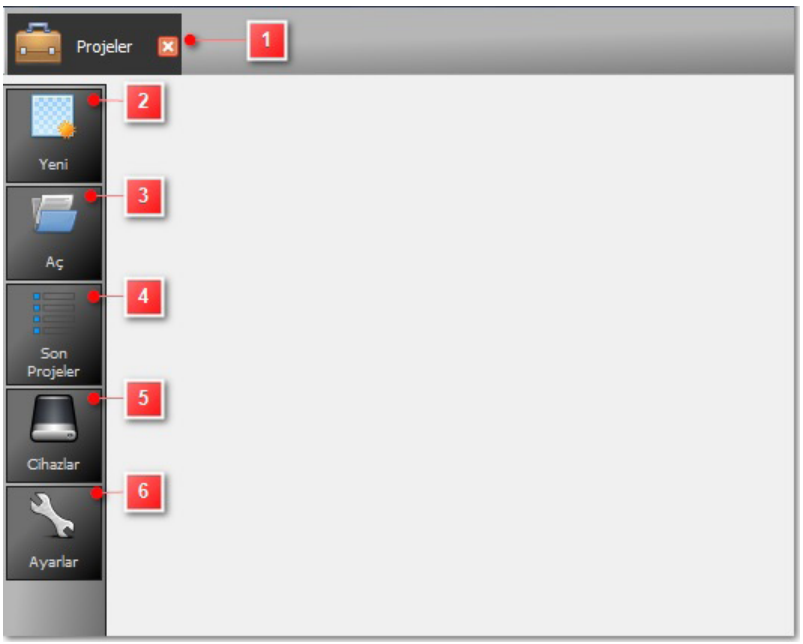
PLC'yi RUN konumundan STOP konumuna ya da STOP konumundan RUN konumuna geçirmek için kullanılır. Ayrıca RESET butonu basılı tutularak PLC'ye enerji verildiğinde ve 10 sn. basılı tutulduğunda, PLC kendini resetler ve ethernet bağlantı şifresini **1-2-3-4** olarak belirler (Görsel 1.17).

Görsel 1.17: GMT PLC'nin RESET buton yeri

1.1.2. PLC Editor (GMTSuite) Programı

Otomasyon sistem programlamasında ladder metodu yaygın olarak tercih edilir. GMTSuite editör programı bu ladder yöntemini kullanır. Bu metod ile istenilen tüm mantıksal senaryo, matematiksel işlemler, haberleşme işlemleri ve özel fonksiyonlar yürütülür. Programın en büyük avantajı dünya çapında farklı kullanıcı seviyelerine hitap edecek kadar basit olmasıdır. Bir diğer avantajı ise kullanıcı talepleri doğrultusunda her geçen gün farklı komutların eklenmesidir.

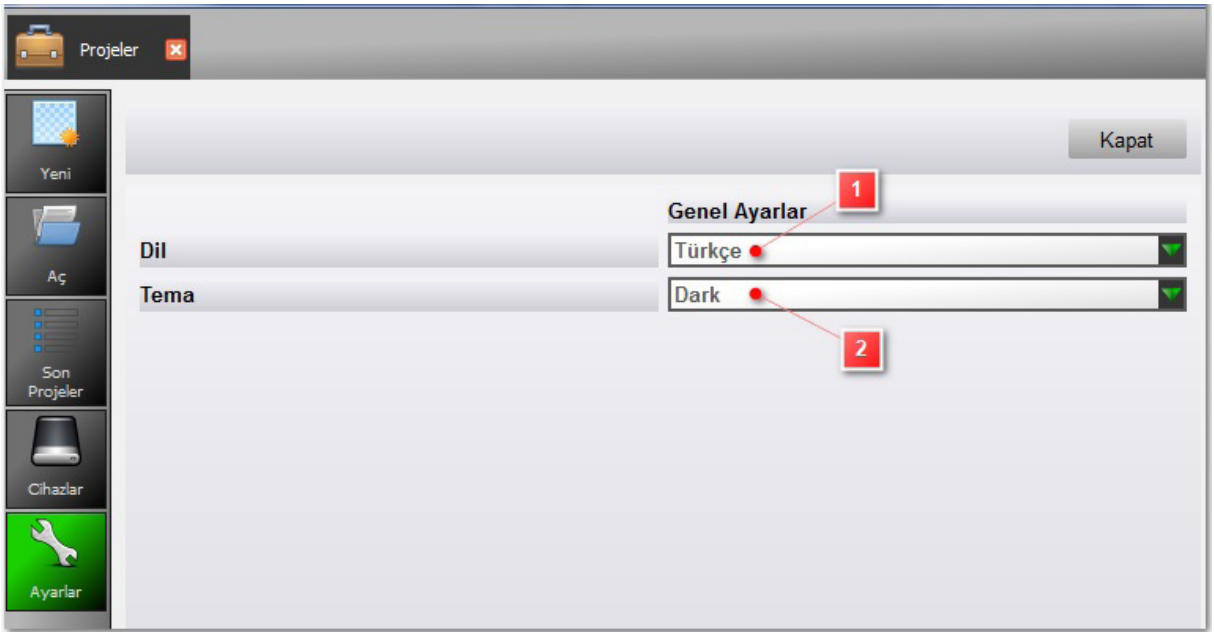
"http://www.gmtcontrol.com/yazilimlar/" adresinden komutlar indirilir ve kurulum aşamalarındaki seçeneklere göre kurulur.



Görsel 1.18: GMTSuite programının ekran görüntüsü

Tablo 1.3: GMTSuite Programının Ekran Görüntü Açıklamaları

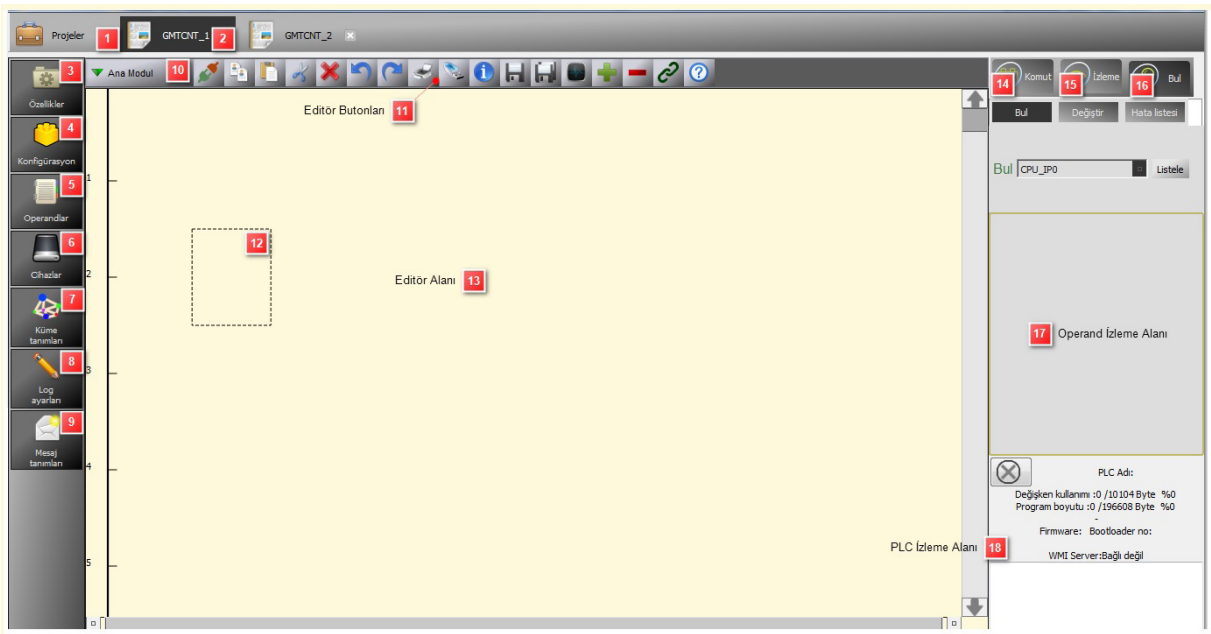
1	**Projeler:** Açık olan projelerin sıralandığı **BAR**'dır (Görsel 1. 18).
2	**Yeni:** Yeni bir proje oluşturmak için kullanılır. GMTSuite kullanıcıya yeni proje dosyası oluşturacağı dizini **farklı kaydet** penceresiyle sorar. Burada seçilen klasör tüm projelerin bulunacağı ana bir klasör olarak seçilebilir. Bundan sonra her kaydetme ve yüklemede GMTSuite bu klasörün adresini hatırlar ve burayı açar. Yeni proje seçilmesiyle aynı proje isminde otomatik bir klasör oluşturulur ve dosyalar buraya kaydedilir. GMTSuite proje dosyasıyla beraber farklı amaçlı yardımcı dosyaları da oluşturur. Bu yöntemle tüm dosyalar bir arada düzenli tutulur.
3	**Aç:** Daha önce oluşturulan projeyi açmak için kullanılır. Butona tıklandığında işletim sisteminin **Aç** penceresi açılır. Kayıtlı olan projenin bulunduğu konuma gelerek açılır.
4	**Son Projeler:** Kullanıcı tarafından son açılan beş projeyi açmak için kısayol olarak kullanılır. Bunu açmak için de projenin adı tıklanır.
5	**Cihazlar:** Bilgisayara bağlı PLC'leri görmek ya da yeni PLC eklemek için kullanılır.
6	**Ayarlar:** Editör dilini ve temasını seçmek için kullanılır.



Görsel 1.19: AYARLAR komutu ile gelen pencere

Editör Ekranı

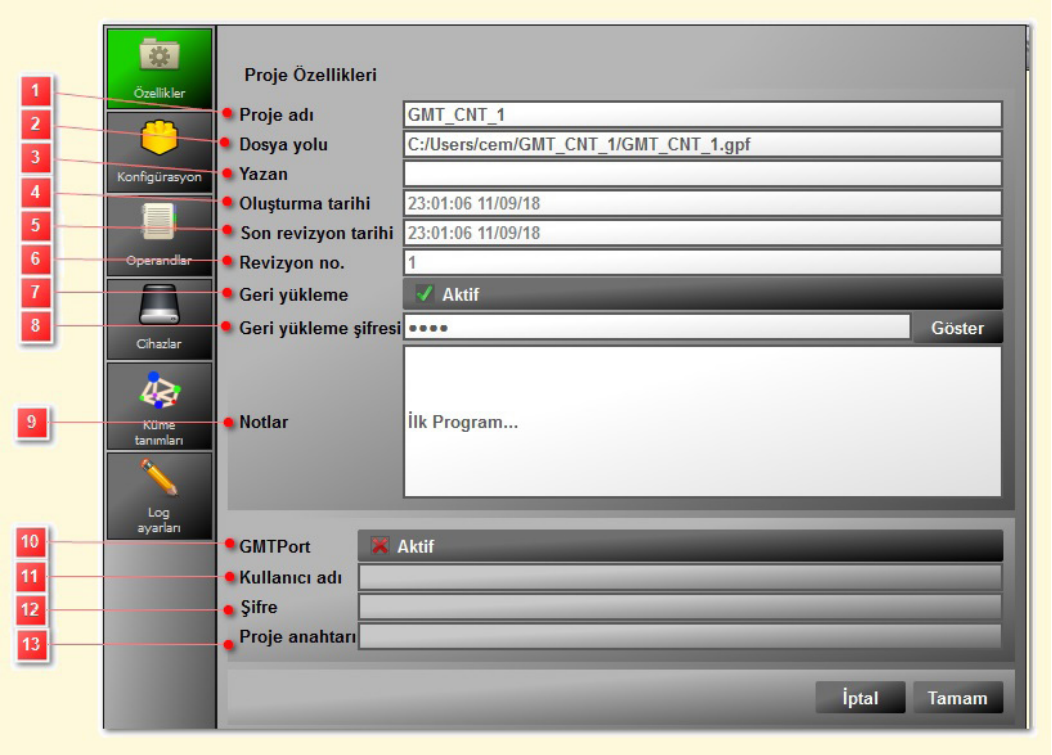
Boş ve yeni proje oluşturmak için YENİ butonu tıklanır (Görsel 1. 19). Proje, klasörü belli olan bir yere uygun bir isim verilerek kaydedilir. Ekrana GMTSuite editörü gelir (Görsel 1. 20, Tablo 1.4).



Görsel 1.20: GMTSuite programının editör ekran görüntüsü

Tablo 1.4: GMTSuite Programının Editör Ekran Görüntü Açıklamaları

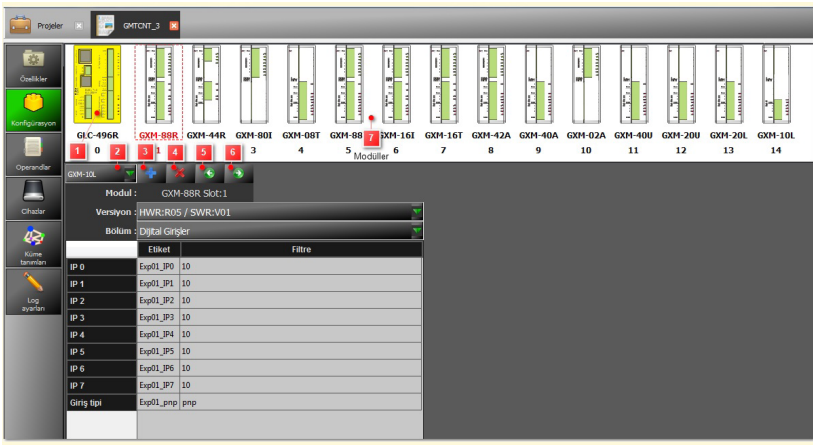
1	**Projeler:** Bir önceki bölümde anlatılan açılış ekranına geçişi sağlar.
2	**Geçerli projenin adı:** YENİ butonuyla oluşturulan proje, bu bar menüde görülür. Açılan diğer projeler de bu menüde bulunur. Projeler arası geçiş ise ilgili proje adı tıklanarak yapılır.
3	**Özellikler:** Proje özelliklerinin belirlendiği pencerenin açılmasını sağlar.
4	**Konfigürasyon:** Projede kullanılacak PLC ve modül konfigürasyon ayarlarının bulunduğu pencerenin açılmasını sağlar. Konfigürasyon tanımlaması yapılmadan LADDER komutları görülmez.
5	**Operandlar:** Projede kullanılacak operand ayarların bulunduğu listeyi açar.
6	**Cihazlar:** PLC bağlantısının ayarlandığı pencereyi açar.
7	**Küme tanımları:** Küme tanmlarının yapıldığı pencereyi açar.
8	**Log ayarları:** Proje, 496X tipi CPU ile hazırlanırsa bu botun görünür. Log ayarlarının yapıldığı pencereyi açar.
9	**Mesaj tanımları:** Proje, 496X tipi CPU ile hazırlanırsa bu botun görünür. Mesaj ayarlarının yapıldığı pencereyi açar.
10	**Alt programlar:** Alt program/programlar kullanılırsa bu botun tıklanarak tanımlanır.
11	**Editör butonları:** Program hazırlanmasında kullanılan butonlardır.
12	**Komut ekleme alanı:** Seçilen komut bu alana yerleşir.
13	**Editör alanı:** Komutları ekleyerek ve bu komutların bağlantılarını yaparak ladder diyagramlarının oluşturduğu program çalışma alanlarıdır. Bir satırda en fazla 10 komut kullanılır. Programda alt programlar hariç en fazla 1136 satır kullanılır.
14	**Komut:** GMTSuite programında kullanılacak komutaların bulunduğu menüyü açar.
15	**İzleme:** Simülasyonda operand içeriklerinin izleneceği alanı açar. **Tablo Ekle** butonuyla operand içerikleri grup hâlinde izlenir.
16	**Bul:** Büyük programların hazırlanmasında operndları bularak hazırlayan ve değiştiren pencereleri açar.
17	**Operand izleme alanı: İzleme > Tablo Ekle** butonuyla tanımlanan operandların izlendiği alandır.
18	**PLC izleme alanı:** PLC ile bağlantı kurulduktan sonra kullanıcı bu alanda PLC'ye ait temel bilgileri izler.



Görsel 1.21: GMTSuite programının ÖZELLİKLER komutuyla gelen ekran görüntüsü

Tablo 1.5: GMTSuite Programının **ÖZELLİKLER** Ekran Açıklamaları

1	**Proje adı:** Proje adı bu metin alanında görülür. Kullanıcı bu metin alanındaki proje ismini değiştirebilir ve **kaydet** butonu ile kaydedebilir (Görsel 1.21).
2	**Dosya yolu:** Projenin kaydedildiği dosya yolu bu alanda görülür.
3	**Yazan:** Proje yazarı bu alanda belirtilebilir.
4	**Oluşturma tarihi:** Projenin ilk oluşturulduğu saat ve tarih bilgisi bu alanda görülür.
5	**Son revizyon tarihi:** Projenin son kaydedildiği saat ve tarih bilgisi bu alanda görülür.
6	**Revizyon no:** Projenin revizyon numarası isteğe bağlı olarak bu alana yazılabilir.
7	**Geri yükleme:** Bu özelliği aktif ederek, projeyi dosya olarak PLC hafızasından bilgisayara geri yüklenebilir. Devre dışı bırakılırsa geri yüklemeye izin verilmez.
8	**Geri yükleme şifresi:** Projenin, PLC hafızasından bilgisayara yüklenebilmesi için kayıtlı şifre kullanıcı tarafından bilinmelidir.
9	**Notlar:** Proje hakkında bilgi vermek için bu alana not yazılabilir.
10	**GMT port: Aktif** kutucuğu işaretlendiğinde aşağıdaki **GMT Port** bilgi giriş alanları aktif olur.
11	**Kullanıcı adı: GMT Port**'a giriş için kullanılan kullanıcı adı bu alana yazılır.
12	**Şifre: GMT Port**'a giriş için kullanılan şifre yazılır.
13	**Proje anahtarı: GMT Port** tarafından üretilen proje anahtarı bu alana yazılır.



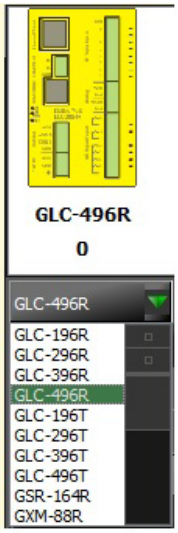
Görsel 1.22: GMTSuite programının **KONFİGÜRASYON** komutu ile gelen ekran görüntüsü

Tablo 1.6: GMTSuite Programının KONFİGÜRASYON Ekran Açıklamaları

1	**CPU:** Projede kullanılacak CPU 0. slota yerleşir. Her bir projede bir CPU kullanılır. CPU seçilmeden ladder komutları görülmez.
2	**CPU / modül seçimi:** Projede kullanılacak olan CPU ve modül seçimi bu açılır menüden yapılır.
3	**Ekle:** Seçilen CPU ya da modül, **Ekle** butonuna tıklamadan slot sıralamasına alınmaz.
4	**Sil:** Slota yerleştirilen CPU ya da modüller, mouse ile işaretlendikten sonra bu buton ile silinebilir.
5	**Sola kaydır:** Modüllerin slot sıralaması değiştirilebilir. Modülü sola kaydırmak için bu buton kullanılır. Fiziki yerleşim burada belirlendiği gibi olmalıdır.
6	**Sağa kaydır:** Modülü sağa kaydırmak için bu buton kullanılır.
7	**Modüller:** Projenin ihtiyacına göre farklı modüller bir ya da birden fazla kullanılabilir. Bu durumda modüller 1. slotdan itibaren yerleştirilir. 14 adet modül kullanılabilir.

PLC CPU Konfigürasyonu

Tüm CPU modellerinin giriş/çıkış özelliklerinde Görsel 1.23'te belitildiği gibi CPU seçimi yapılır, sonra girilecek parametre ayarları verilir.



Model	Röle Çıkışı	Transistör Çıkışı	Analog Giriş	Analog Çıkış	RTC	HSO Max. Frekans	WMI Desteği
GLC-196R	✓						
GLC-296R	✓		✓	✓			
GLC-396R	✓		✓	✓	✓		✓
GLC-496R	✓		✓	✓	✓		✓
GLC-196T		✓				20kHz	
GLC-296T		✓	✓	✓		100kHz	
GLC-396T		✓	✓	✓	✓	100kHz	✓
GLC-496T		✓	✓	✓	✓	400kHz	✓

Görsel 1.23: PLC CPU özellikleri

1.1.3. Veri Tipleri

BIT Veri Tipi: Bir bitin durumunu kaydeder. PLC'de 0 açık, 1 kapalı anlamına gelir.

WORD Veri Tipi: 16 bitlik veri tipi veya 2 byte olarak da ifade edilir. 0 ile 65535 aralığında yer alır.

DWORD (DOUBLE WORD) Veri Tipi: 32 bitlik veri tipi veya 4 byte olarak da ifade edilir. 0 ile 4.294.967.295 aralığında yer alır.

INTEGER Veri Tipi: İşaretli tam sayı değişken tipli ve 32 bit (4 byte) büyüklüğündedir. - 2.147.483.648 ile + 2.147.483.647 aralığında değer alır.

REAL (REAL OR FLOATING POINT) Veri Tipi: Ondalıklı sayı tipi değişken ve 32 bit (4 byte) büyüklüğündedir. $-1.18x10^{-38}$ ile $+3.40x10^{+38}$ aralığında değer alır. Bu değişkeni kullanırken ondalık kısmı için nokta kullanılır (25.4).

1.1.4. Lojik Komutlar

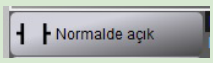
Bu bölümde ladder diyagramda kullanılan temel kontak ve rölelerin özellikleri yer alır. Bu komutlar, GLC ve GSR serisinin PLC modellerinde kullanır. Programlarda lojik komutları farklı bağlantılarla kullanılır. Direkt girişe ya da çıkışa bağlı olan kontakların, PLC'nin elektriksel bağlantılarına uyumlu olmalıdır. Yüksek hızda çalışan direkt çıkış rölelerinde transistör çıkışlı PLC kullanılır.

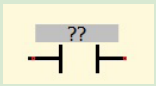
Normalde Açık Kontak

Kısa adı : NO

Kısa yol no. : Enter > 0

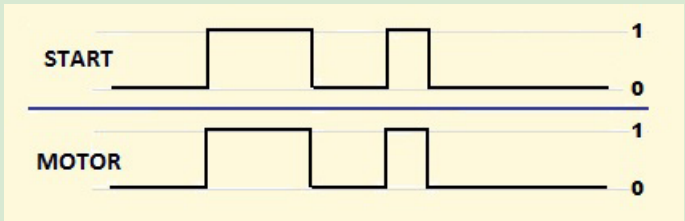
İkonu :

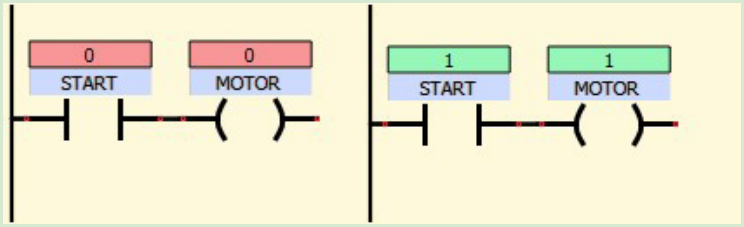




Operand tipleri : Bit, integer

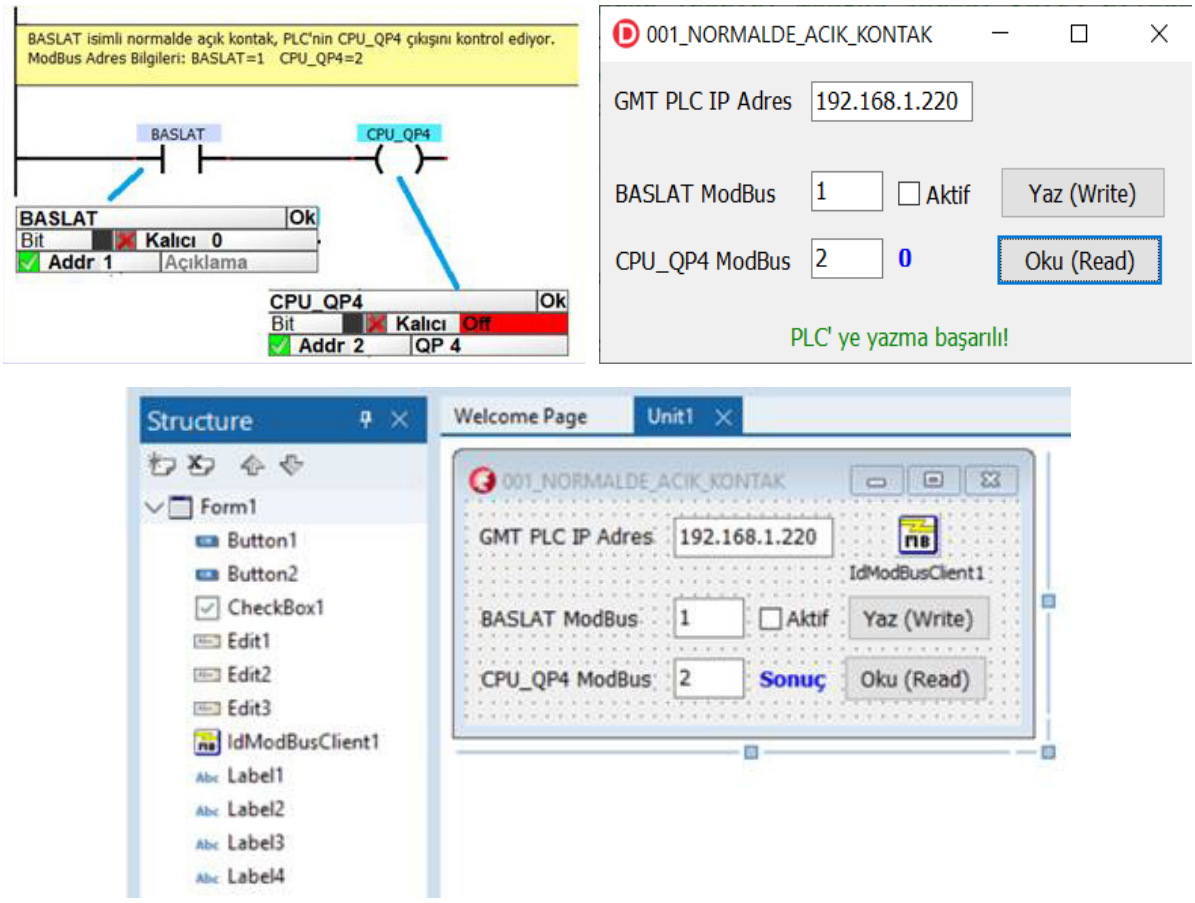
Çalışması :





Örnek uygulama : Atanan bit değeri 1'e eşit olduğunda **Normalde Açık Kontak** kapanır ve sonraki komutların yürütülmesine izin verilir. Operand tipi integer olarak belirlenmiş ve değeri 1 ve 1'den büyük ise sonraki komutlar yürütülür.

1.1. UYGULAMA: Normalde Açık Kontak



Görsel 1.24: Normalde açık kontak ladder diyagramı ve Delphi tasarım arayüzü

```
procedure TForm1.Button2Click(Sender: TObject);
var
  SONUC: Boolean; // CPU_QP4 isimli kontaktörün DURUM bilgisinin atanacağı değişken.
begin
  IdModBusClient1.Host := Edit1.Text; // GMT PLC nin bağlı olduğu IP adresi.
  // CPU_QP4 isimli kontaktörün (bobin) ModBus adresi Edit3 bileşenindedir.
  if IdModBusClient1.ReadCoil(StrToInt(Edit3.Text), SONUC) then
    // PLC okuma sonucu Label4 e atanır.   True -1    False 0
    Label4.Caption := BoolToStr(SONUC)
  else
    Label5.Caption := 'PLC''den okuma başarısız!';
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text; // GMT PLC nin bağlı olduğu IP adresidir.
  // BASLAT isimli NO kontağın ModBus adresi Edit3 bileşenindedir.
  if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

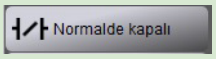
Görsel 1.25: Normalde açık kontak program kodu

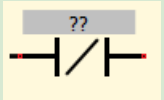
Normalde Kapalı Kontak

Kısa adı : NC

Kısa yol no. : Enter > 1

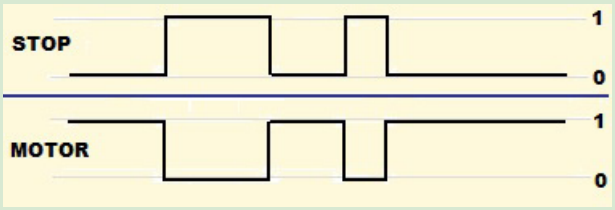
İkonu :

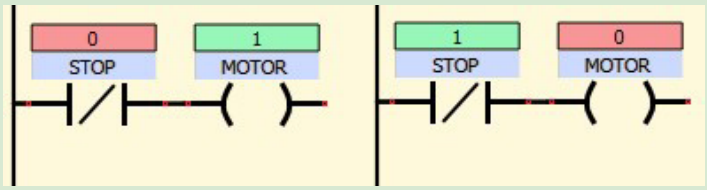




Operand tipleri : Bit, integer.

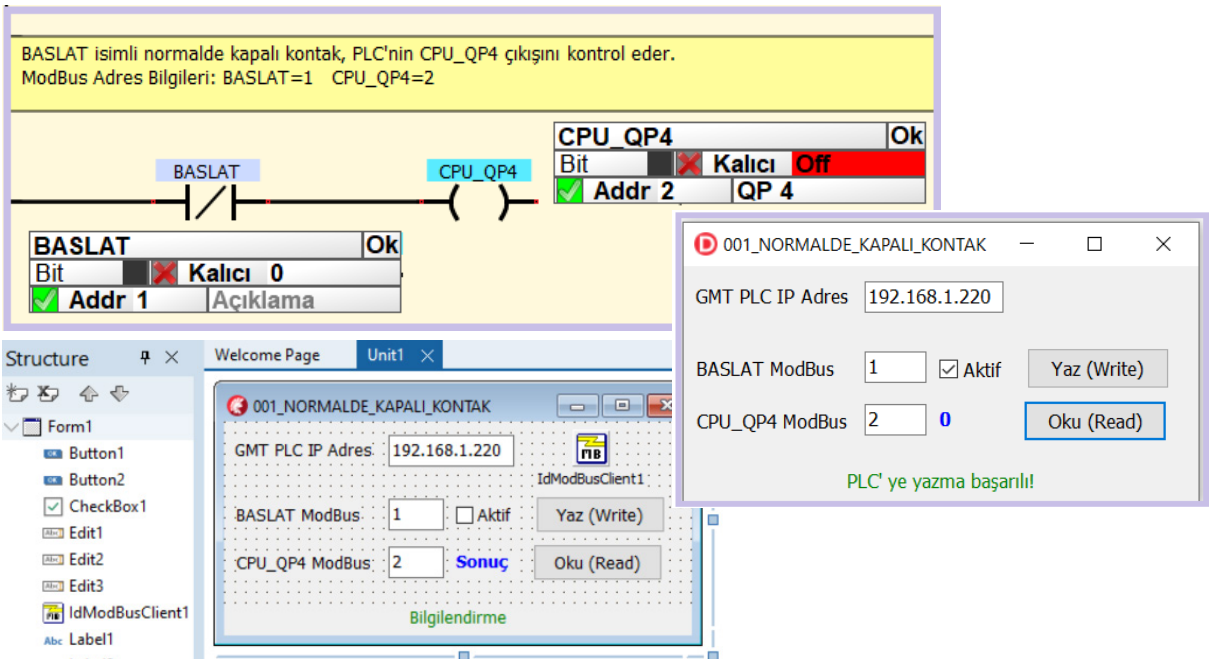
Çalışması :





Örnek uygulama : Atanan bit değeri 1'e eşit olduğunda **Normalde Kapalı Kontak** açılır ve sonraki komutların çalıştırılmasına izin verilmez. Operand tipi integer olarak belirlenmiş ve değeri 1 ve 1'den büyük ise sonraki komutlar çalıştırılmaz.

1.2. UYGULAMA: Normalde Kapalı Kontak



Görsel 1.26: Normalde kapalı kontak ladder diyagramı ve Delphi tasarım arayüzü

```
procedure TForm1.Button2Click(Sender: TObject);
var
  SONUC: Boolean; // CPU_QP4 isimli kontaktörün DURUM bilgisinin atanacağı değişken.
begin
  IdModBusClient1.Host := Edit1.Text; // GMT PLC nin bağlı olduğu IP adresi.
  // CPU_QP4 isimli kontaktörün (bobin) ModBus adresi Edit3 bileşenindedir.
  if IdModBusClient1.ReadCoil(StrToInt(Edit3.Text), SONUC) then
    // PLC okuma sonucu Label4 e atanır.   True -1    False 0
    Label4.Caption := BoolToStr(SONUC)
  else
    Label5.Caption := 'PLC''den okuma başarısız!';
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
  // BASLAT isimli NO kontağın ModBus adresi Edit3 bileşenindedir.
  if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

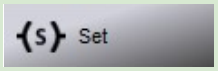
Görsel 1.27: Normalde kapalı kontak program kodu

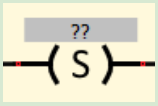
Set Komutu

Kısa adı : ST

Kısa yol no. : Enter > 4

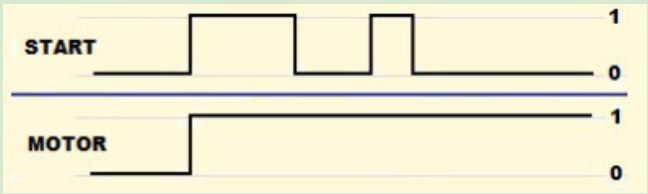
İkonu :

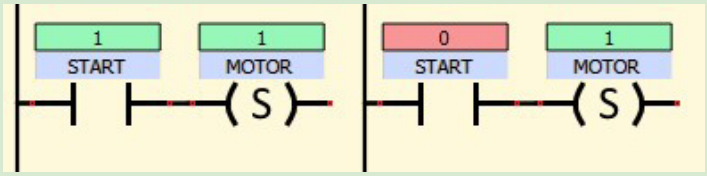




Operand tipleri : Bit

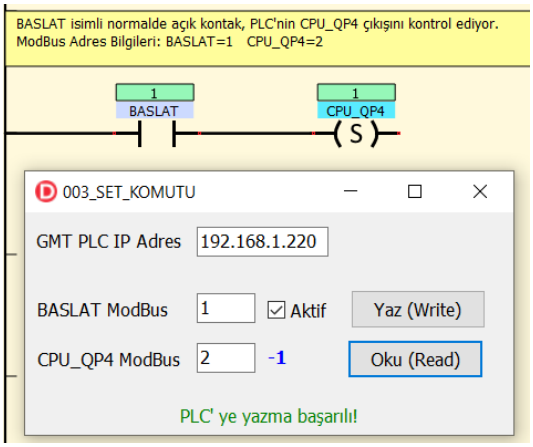
Çalışması :

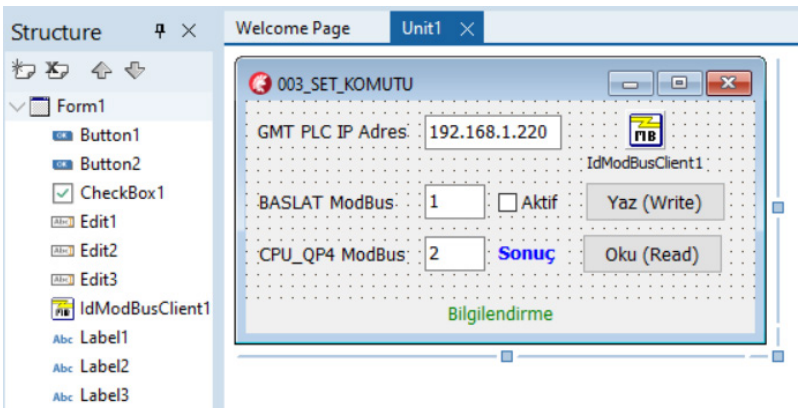




Örnek uygulama : Atanan **START** aktif edildiğinde **MOTOR** adlı çıkış rölesi enerjilenir. **START** kontağı tekrar pasif yapıldığında çıkış rölesi enerjili olarak kalmaya devam eder. **SET** edilen röle sadece **RESET** komutuyla pasif yapılabilir.

1.3. UYGULAMA: SET Komutu





Görsel 1.28: Set komutu ladder diyagramı ve Delphi tasarım arayüzü

```
procedure TForm1.Button2Click(Sender: TObject);
 var
   SONUC: Boolean; // CPU_QP4 isimli kontaktörün DURUM bilgisinin atanacağı değişken.
 begin
   IdModBusClient1.Host := Edit1.Text; // GMT PLC nin bağlı olduğu IP adresi.
   // CPU_QP4 isimli kontaktörün (bobin) ModBus adresi Edit3 bileşenindedir.
   if IdModBusClient1.ReadCoil(StrToInt(Edit3.Text), SONUC) then
     // PLC okuma sonucu Label4 e atanır.    True -1    False 0
     Label4.Caption := BoolToStr(SONUC)
   else
     Label5.Caption := 'PLC''den okuma başarısız!';
 end;

 procedure TForm1.Button1Click(Sender: TObject);
 begin
   IdModBusClient1.Host := Edit1.Text;
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
     Label5.Caption := 'PLC'' ye yazma başarılı!'
   else
     Label5.Caption := 'PLC''ye yazma başarısız!';
 end;
```

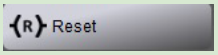
Görsel 1.29: Set komutu program kodu

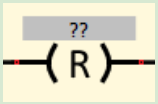
Reset Komutu

Kısa adı : RT

Kısa yol no. : Enter > 5

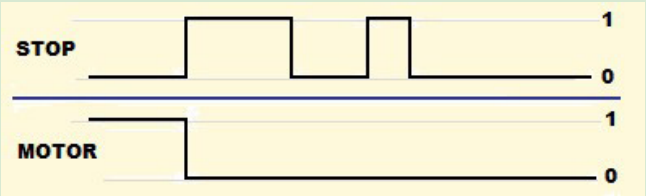
İkonu :

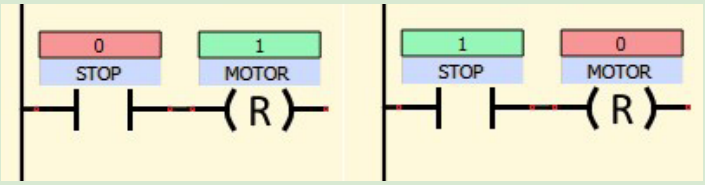




Operand tipleri : Bit

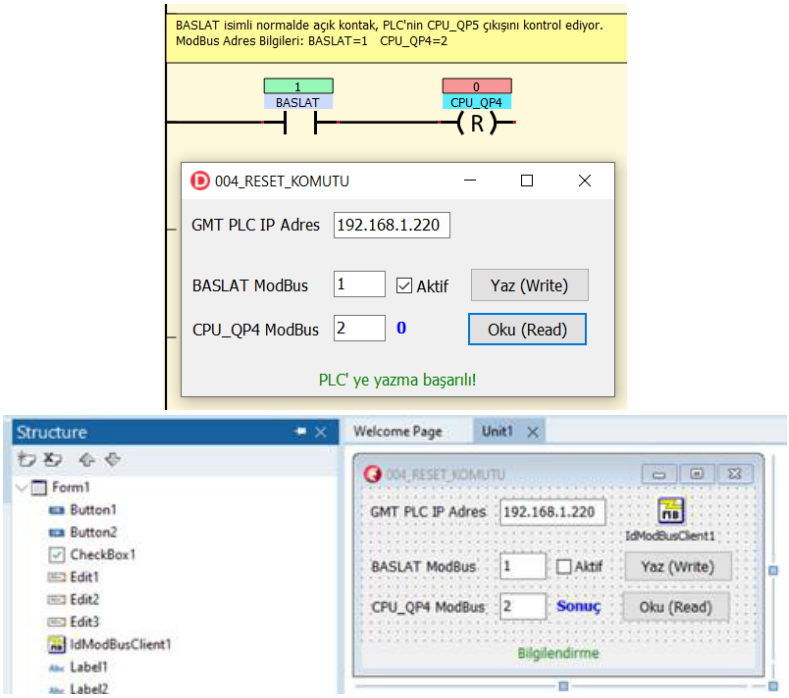
Çalışması :





Örnek uygulama : Daha önceden **SET** komutu ile aktif edilen **MOTOR** adlı çıkış rölesi, **STOP** komutunun aktif edilmesiyle pasif olur.

1.4. UYGULAMA: RESET Komutu



Görsel 1.30: Reset komutu ladder diyagramı ve Delphi tasarım arayüzü

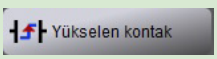
```
procedure TForm1.Button2Click(Sender: TObject);
var
  SONUC: Boolean;
begin
  IdModBusClient1.Host := Edit1.Text; // GMT PLC nin bağlı olduğu IP adresi.
  // CPU_QP4 isimli kontaktörün (bobin) ModBus adresi Edit3 bileşenindedir.
  if IdModBusClient1.ReadCoil(StrToInt(Edit3.Text), SONUC) then
    // PLC okuma sonucu Label4 e atanır.   True -1    False 0
    Label4.Caption := BoolToStr(SONUC)
  else
    Label5.Caption := 'PLC''den okuma başarısız!';
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
  // BASLAT isimli NO kontağın ModBus adresi Edit3 bileşenindedir.
  if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

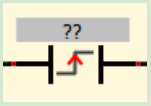
Görsel 1.31: Reset komutu program kodu

Yükselen Kontak

Kısa adı : YK

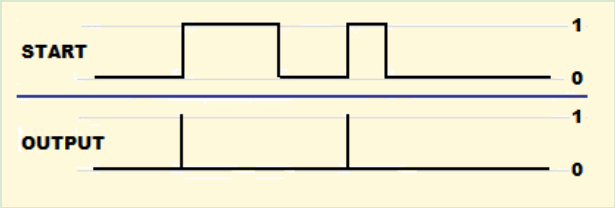
Kısa yol no. : Enter > 2

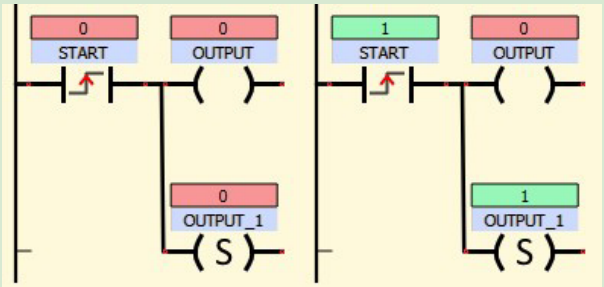
İkonu : 



Operand tipleri : Bit

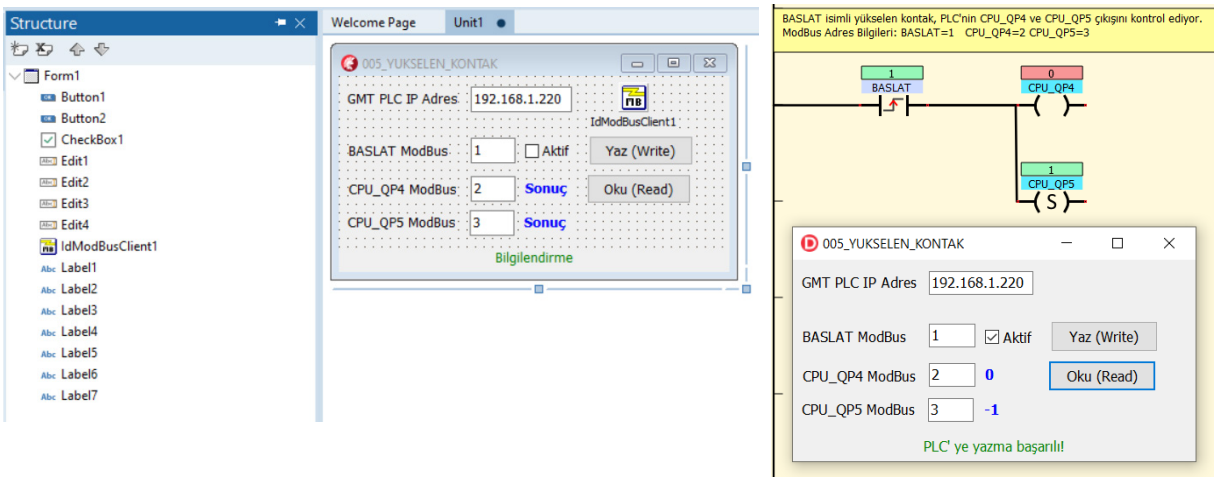
Çalışması :





Örnek uygulama : **START** aktif edildiğinde; **START**'ın yükselen kenarında **OUTPUT** aktif olur (programın tek döngü süresinde) sonra tekrar pasif olur. Bu işlem çok hızlı olduğu için **OUTPUT** pasif ancak SET komutu ile kullanılan **OUTPUT_1** ise aktif kalır.

1.5. UYGULAMA: Yükselen Kontak



Görsel 1.32: Yükselen kontak ladder diyagramı ve Delphi tasarım arayüzü

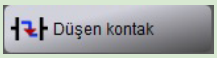
```
procedure TForm1.Button2Click(Sender: TObject);
var
  SONUC: Boolean; // CPU_QP4 isimli kontaktörün DURUM bilgisinin atanacağı değişken.
begin
  IdModBusClient1.Host := Edit1.Text; // GMT PLC nin bağlı olduğu IP adresi.
  // CPU_QP4 isimli kontaktörün (bobin) ModBus adresi Edit3 bileşenindedir.
  if IdModBusClient1.ReadCoil(StrToInt(Edit3.Text), SONUC) then
    // PLC okuma sonucu Label4 e atanıyor.   True -1    False 0
    Label4.Caption := BoolToStr(SONUC)
  else
    Label5.Caption := 'PLC''den okuma başarısız!';
    if IdModBusClient1.ReadCoil(StrToInt(Edit4.Text), SONUC) then
    Label7.Caption := BoolToStr(SONUC)
  else
    Label5.Caption := 'PLC''den okuma başarısız!';
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
   if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

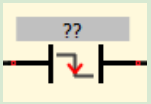
Görsel 1.33: Yükselen kontak program kodu

Düşen Kontak

Kısa adı : DK

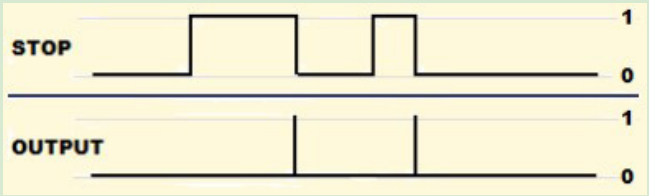
Kısa yol no. : Enter > 3

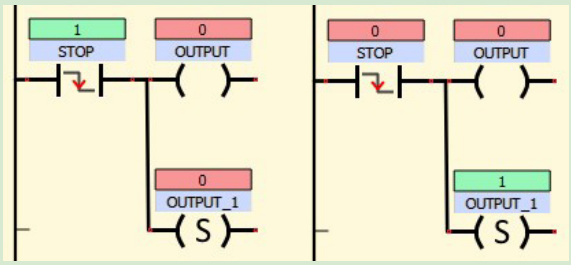
İkonu : 



Operand tipleri : Bit

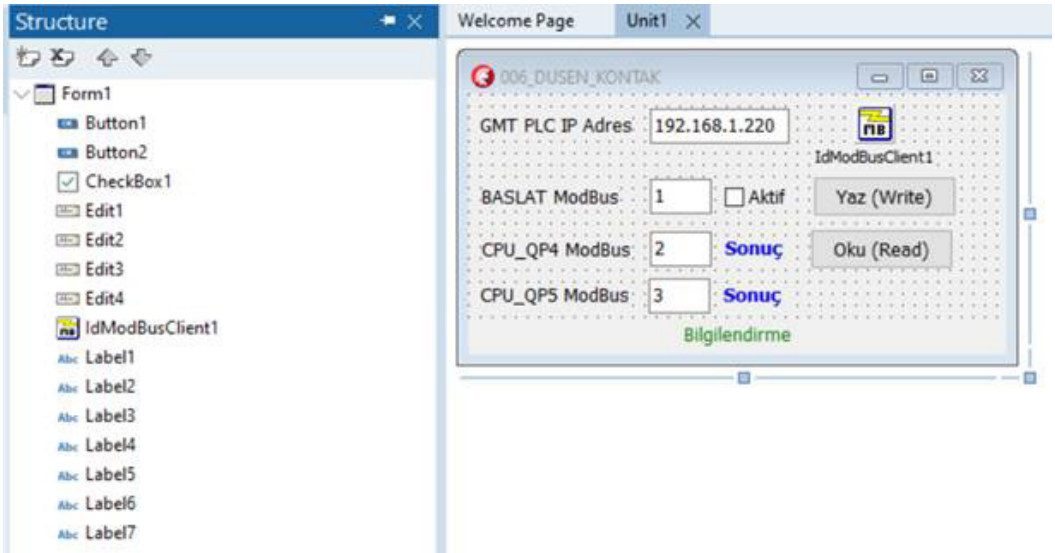
Çalışması :



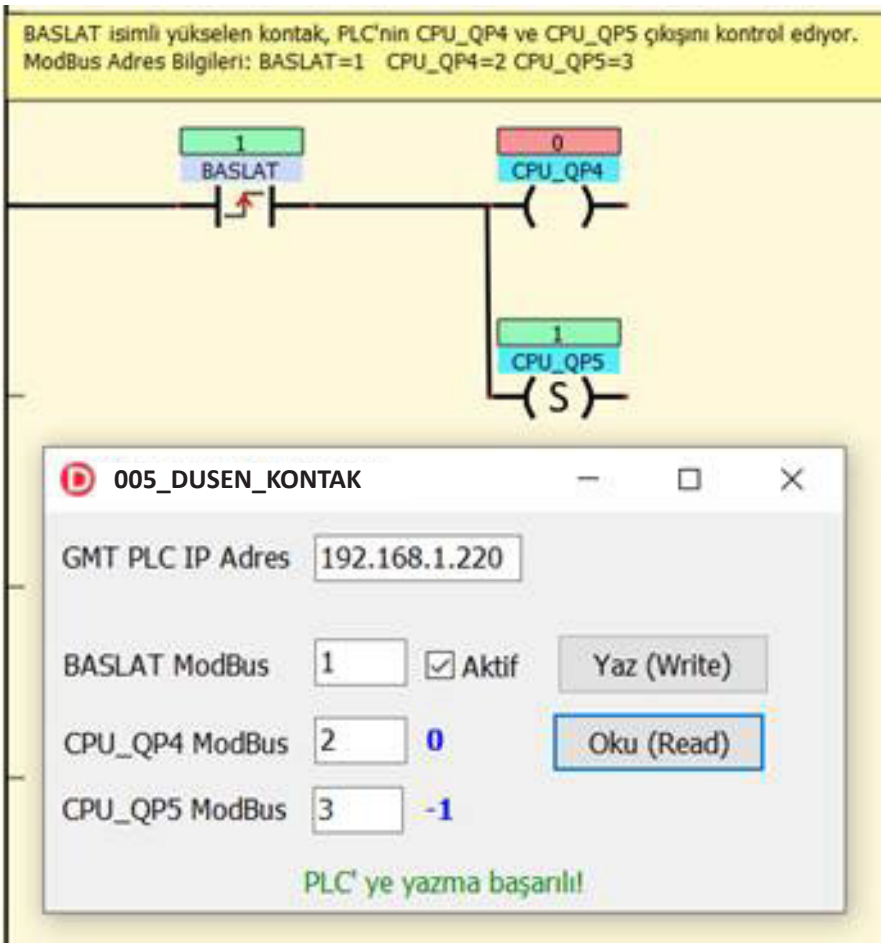


Örnek uygulama : **STOP** pasif edildiğinde; **STOP**'un düşen kenarında **OUTPUT** aktif olur (programın tek döngü süresinde), sonra tekrar pasif olur. Bu işlem çok hızlı olduğu için **OUTPUT** pasif ancak **SET** komutu ile kullanılan **OUTPUT_1** aktif kalır.

1.6. UYGULAMA: Düşen Kontak



Görsel 1.34: Düşen kontak Delphi tasarım arayüzü



Görsel 1.35: Düşen kontak ladder diyagramı

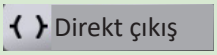
```
procedure TForm1.Button2Click(Sender: TObject);
var
  SONUC: Boolean; // CPU_QP4 isimli kontaktörün DURUM bilgisinin atanacağı değişken.
begin
  IdModBusClient1.Host := Edit1.Text; // GMT PLC nin bağlı olduğu IP adresi.
   if IdModBusClient1.ReadCoil(StrToInt(Edit3.Text), SONUC) then
   Label4.Caption := BoolToStr(SONUC)
  else
    Label5.Caption := 'PLC''den okuma başarısız!';
    if IdModBusClient1.ReadCoil(StrToInt(Edit4.Text), SONUC) then
    Label7.Caption := BoolToStr(SONUC)
  else
    Label5.Caption := 'PLC''den okuma başarısız!';
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text; // GMT PLC nin bağlı olduğu IP adresi.
   if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

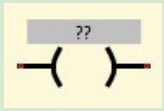
Görsel 1.36: Düşen kontak program kodu

Direkt Çıkış Rölesi

Kısa adı : DÇ

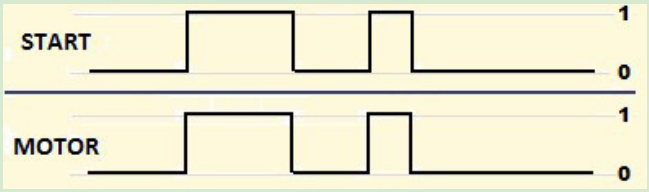
Kısa yol no. : Enter > 6

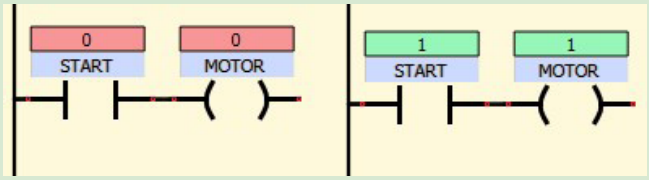
İkonu : 



Operand tipleri : Bit

Çalışması :



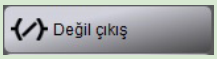


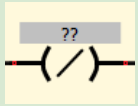
Örnek uygulama : **START** pasif edildiğinde **MOTOR** çıkış rölesi pasif olur. **START** aktif edildiğinde **MOTOR** çıkış rölesi aktif olur.

Değil Çıkış Rölesi

Kısa adı : IÇ

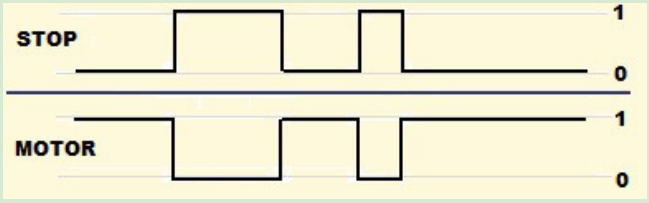
Kısa yol no. : Enter > 7

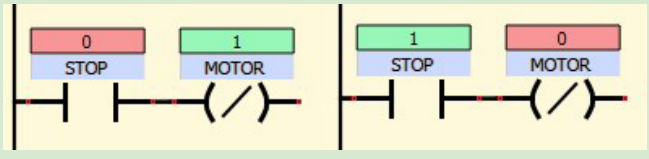
İkonu : 



Operand tipleri : Bit

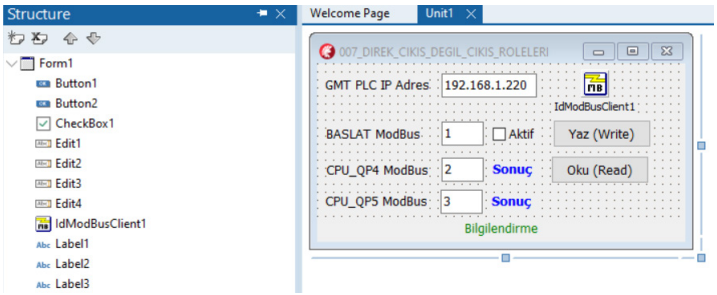
Çalışması :



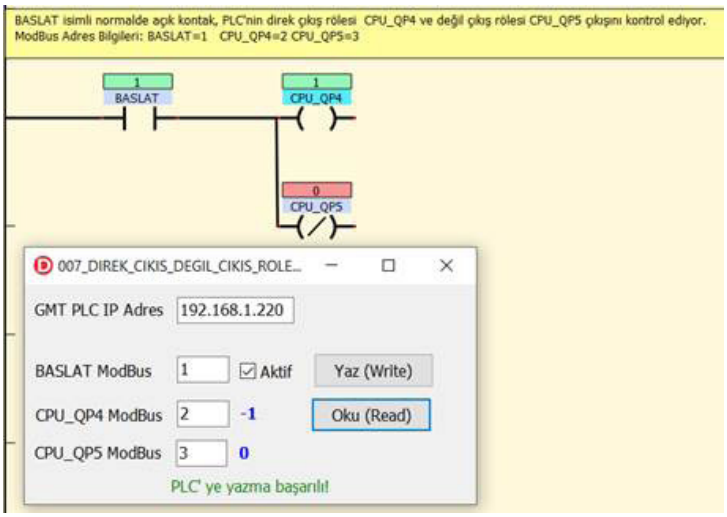


Örnek uygulama : **STOP** pasif edildiğinde **MOTOR** çıkış rölesi aktif olur. **STOP** aktif edildiğinde **MOTOR** çıkış rölesi pasif olur.

1.7. UYGULAMA: Direkt Çıkış ve Değil Çıkış Röleleri



Görsel 1.37: Direkt çıkış ve değil çıkış röleleri Delphi tasarım arayüzü



Görsel 1.38: Direkt çıkış ve değil çıkış röleleri ladder diyagramı

```
procedure TForm1.Button2Click(Sender: TObject);
var
  SONUC: Boolean; // CPU_QP4 isimli kontaktörün DURUM bilgisinin atanacağı değişken.
begin
  IdModBusClient1.Host := Edit1.Text; // GMT PLC nin bağlı olduğu IP adresi.
    if IdModBusClient1.ReadCoil(StrToInt(Edit3.Text), SONUC) then
    Label4.Caption := BoolToStr(SONUC)
  else
    Label5.Caption := 'PLC''den okuma başarısız!';
    if IdModBusClient1.ReadCoil(StrToInt(Edit4.Text), SONUC) then
    Label7.Caption := BoolToStr(SONUC)
  else
    Label5.Caption := 'PLC''den okuma başarısız!';
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text; // GMT PLC nin bağlı olduğu IP adresi.
  if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

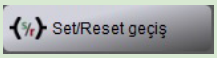
Görsel 1.39: Direkt çıkış ve değil çıkış röleleri program kodu

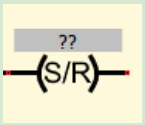
Set/Reset Geçiş Rölesi

Kısa adı : SR

Kısa yol no : Enter > 8

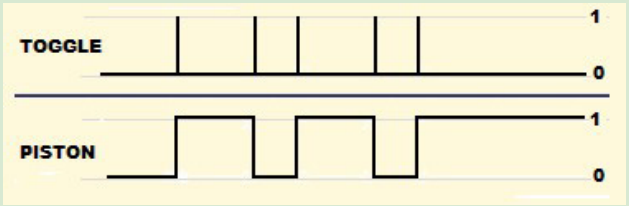
İkonu :

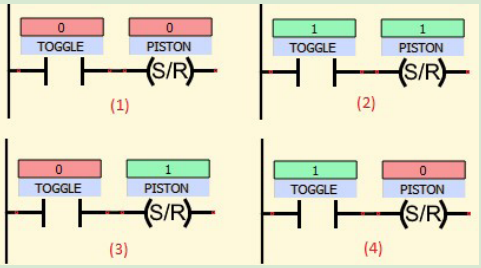




Operand tipleri : Bit

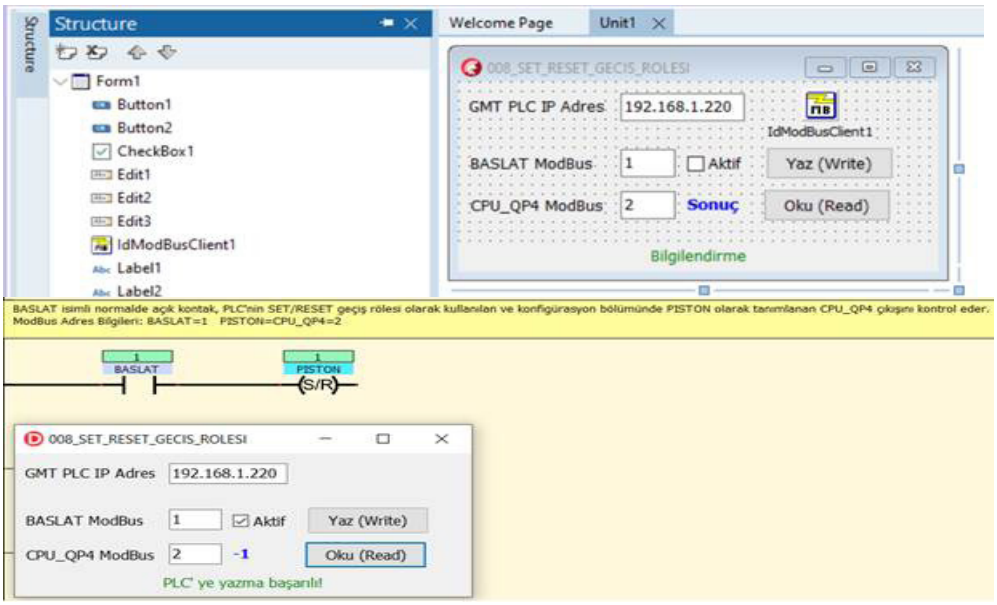
Çalışması :





Örnek uygulama : **TOGGLE** aktif edildiğinde **PISTON** adlı **S/R** çıkış rölesi enerjilenir. **TOGGLE** pasif yapıldığında **PISTON** adlı **S/R** çıkış rölesi konumunu korur. Bu şekilde rölenin her aktif edilmesinde bit değeri değişir. Bu EKRAN görüntüsü PLC YAZILIM'ındaki HELP ten alıntıdır.

1.8. UYGULAMA: SET/RESET Geçiş Rölesi



Görsel 1.40: SET/RESET geçiş rölesi Delphi tasarım arayüzü ve ladder diyagramı

```
procedure TForm1.Button2Click(Sender: TObject);
var
  SONUC: Boolean;
begin
  IdModBusClient1.Host := Edit1.Text; // GMT PLC nin bağlı olduğu IP adresi.
   if IdModBusClient1.ReadCoil(StrToInt(Edit3.Text), SONUC) then
     Label4.Caption := BoolToStr(SONUC)
  else
    Label5.Caption := 'PLC''den okuma başarısız!';
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
  if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

Görsel 1.41: SET/RESET geçiş rölesi program kodu

1.1.5. Matematik Komutları

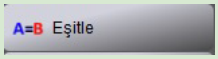
Bu bölümde açıklanan komutlar, GLC ve GSR serisinin PLC modellerinde kullanılır. Bu komutlar veri üzerinde temel matematik işlemlerini yapmaya yarar.

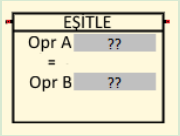
Eşitle Komutu

Kısa adı : ET

Kısa yol no. : Enter > 9

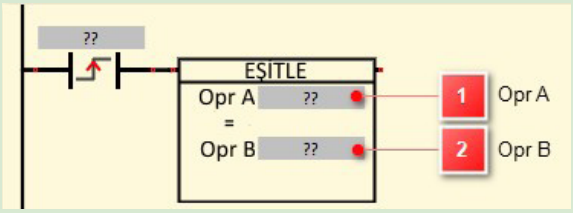
İkonu :





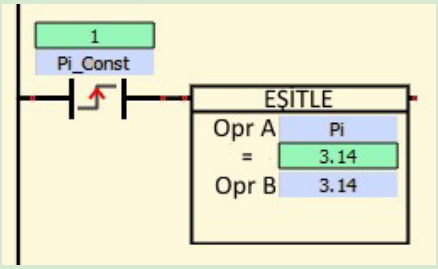
Operand tipleri : Word, double word, integer, real.

Çalışması :



1	**Opr A:** Eşitleme işlemi için hedef olan operand adı yazılır.
2	**Opr B**: Eşitleme işlemi için kaynak sabit değer ya da operand adı yazılır.

Örnek uygulama :



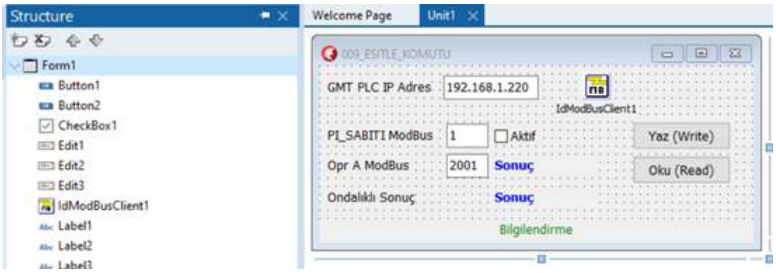
Pi_Const aktif edildiğinde; **Opr A** içeriğine (Pi), **Opr B** içeriği (3.14) taşınır. **Opr B** içeriği taşıma işlemiyle değişmez. 3.14 real bir değerdir. Bu sebeple hedef operandın tipi real olarak seçilir. Aksi durumda kaynak operand değerinin tam kısmı taşınacaktır.

BİLGİ

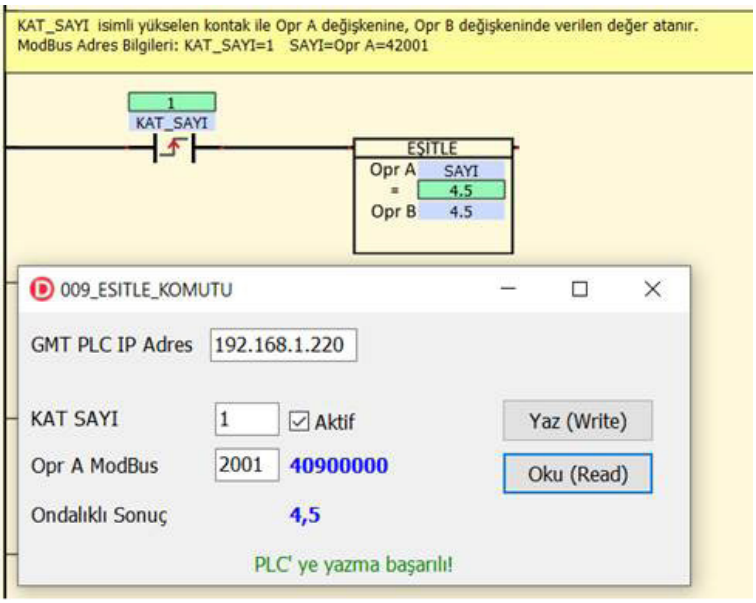
PLC'de MODBUS haberleşme kullanıldığında değişken adresleri 40.000 eksiltilerek yazılır. 42.001 olan adresten bilgi okurken 2001 yazılır. 40.001 olan adresten bilgi okurken 1 olarak yazılır.

40.000 gibi büyük sayıları üçlü ifade etmek için ya aralarına nokta ya da boşluk bırakılması önerilir.

1.9. UYGULAMA: Eşitle Komutu



Görsel 1.42: Eşitle komutu Delphi arayüzü



Görsel 1.43: Eşitle komutu ladder diyagramı

BİLGİ

PLC'den gelen bilgi, ondalıklı bir sayıya karşılık gelen heksadesimal sayı olarak gönderilir. Gönderilen bu değer, IEEE754 standardına göre ondalıklı sayıya dönüştürülebilir.

```
function TForm1.HexToIEEE754(hexVal: String): Real;
var
  sign,exponent,mantissa,position,currentBit:Integer;
  value:Real;
begin
  if (Length(hexVal) < 8) or (not TryStrToInt('$' + hexVal, mantissa)) then
  begin
    ShowMessage('8 haneli hex sayı giriniz.');
    HexToIEEE754 := 0.0;
    Exit;
  end;
  //Ondalıklı sayıya dönüştürülmek istenen onaltılı sayı label4'ten alınır.
  if (StrToInt('$'+ hexVal) and $7FFFFFFF) = 0 then
  begin
    HexToIEEE754 := 0.0;
    Exit;
  end;
  if StrToInt(LeftStr('$' + hexVal, 2)) >= 8 then
    sign := -1  // sayı negatif
  else
    sign := 1;  // sayı pozitif
  // Sayının tam sayı kısmı hesaplanır.
  exponent := StrToInt(LeftStr('$' + hexVal, 4));
  exponent := exponent and $7FF;
  exponent := exponent shr 3;
  exponent := exponent - 127;
  // Sayının ondaklık kısmı hesaplanır
  mantissa := StrToInt('$' + RightStr(hexVal, 6));
  mantissa := mantissa or $800000;
  value := 0.0;
  for position := 0 to 23 do
  begin
    currentBit := (mantissa shr position) and 1;
    value := value + (currentBit * Power(2, exponent - 23 + position));
    end;
  HexToIEEE754 := sign * value;
end;
```

```
procedure TForm1.Button2Click(Sender: TObject);
var
  Data: array[0..10] of Word;
  sLine: String;
  okuma_sayisi,i: Integer;
begin
  okuma_sayisi :=2;// Birden fazla data okunacak olursa buradaki değer değiştirilir.
  if (okuma_sayisi > 0) then
  begin
    IdModBusClient1.Host := edit1.Text;
    if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit3.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label4.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
      end;
      label6.Caption := FloatToStr(HexToIEEE754(label4.Caption));
    end
    else
      ShowMessage('PLC''den okuma başarısız!');
  end;
end;
```

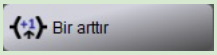
```
procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text; // GMT PLC nin bağlı olduğu IP adresi.
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

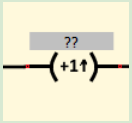
Görsel 1.44: Eşitle komutu program kodu

Bir Artır Komutu

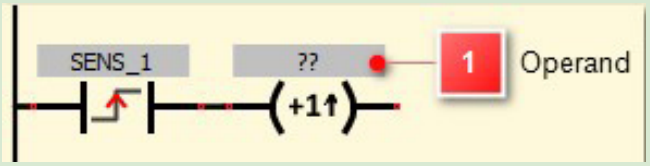
Kısa adı : BA

Kısa yol no. : Enter > 10

İkonu : 

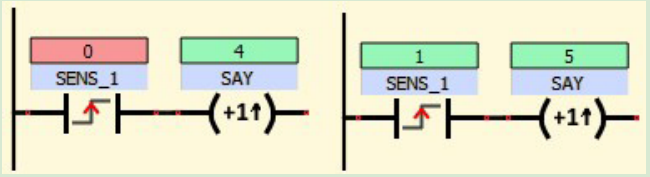


Operand tipleri : Word, double word, integer, real.



1	**Operand:** Bir artırılacak olan operandın adı yazılır.

Çalışması : Word için değer aralığı = 0 - 65.535
Double Word için değer aralığı = 0 - 4.294.967.295
Integer için değer aralığı = -2.147.483.648 - 2.147.483.647
Real için değer aralığı = -1.7976931348623158e+308 - 1.7976931348623158e+308

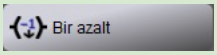


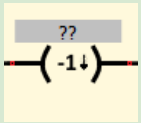
Örnek uygulama : Arttırma komutunun aktif olmasıyla **SAY** adlı operandın içeriğine **1** eklenir. Komutun yükselen kenar kontak ile kullanılması önemlidir.

Bir Azalt Komutu

Kısa adı : BZ

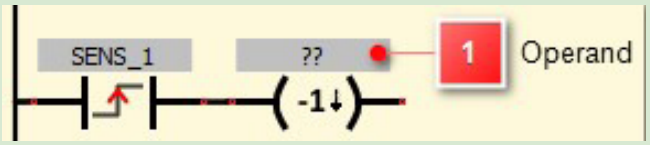
Kısa yol no. : Enter > 11

İkonu : 

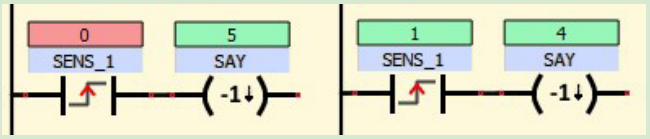


Operand tipleri : Word, double word, integer, real.

Çalışması :

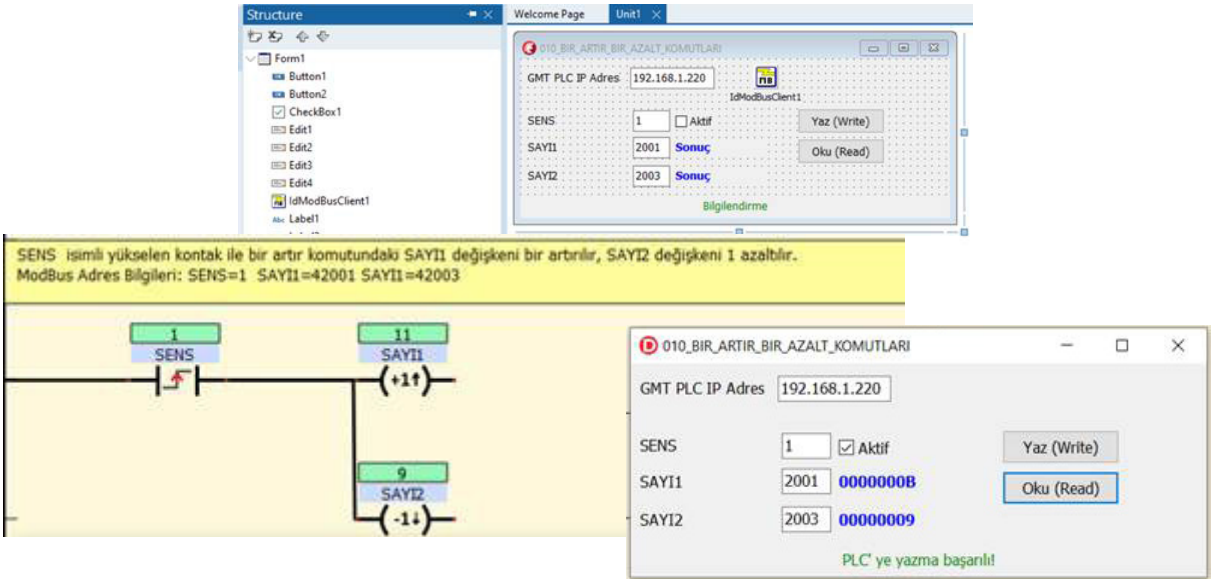


1	**Operand:** Bir azaltılacak olan operandın adı yazılır.



Örnek uygulama : **Azaltma** komutunun aktif olmasıyla **SAY** adlı operandın içeriğinden **1** çıkarılır. Komutun yükselen kenar kontak ile kullanılması önemlidir.

1.10. UYGULAMA: Bir Artır ve Bir Azalt Komutları



Görsel 1.45: Bir artır ve bir azalt komutları Delphi arayüzü ve ladder diyagramı

```
procedure TForm1.Button2Click(Sender: TObject);
 var
   Data: array[0..10] of Word;
   sLine: String;
   okuma_sayisi,i: Integer;
 begin
   okuma_sayisi :=2;// Birden fazla data okunacak olursa buradaki değer değiştirilir.
   if (okuma_sayisi > 0) then
   begin
     IdModBusClient1.Host := edit1.Text;
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit3.Text), okuma_sayisi, Data) then
     begin
       for i := 0 to (okuma_sayisi - 1) do
       begin
       label4.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
       end;
     end ;
      if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit4.Text), okuma_sayisi, Data) then
     begin
       for i := 0 to (okuma_sayisi - 1) do
       begin
       label7.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
       end;
     end
     else
       ShowMessage('PLC''den okuma başarısız!');
   end;
 end;
procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text; // GMT PLC nin bağlı olduğu IP adresi.
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

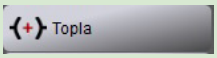
Görsel 1.46: Bir artır ve bir azalt komutları program kodu

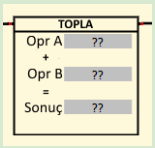
Topla Komutu

Kısa adı : TP

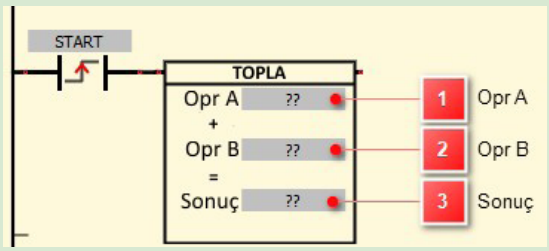
Kısa yol no. : Enter > 12

İkonu :





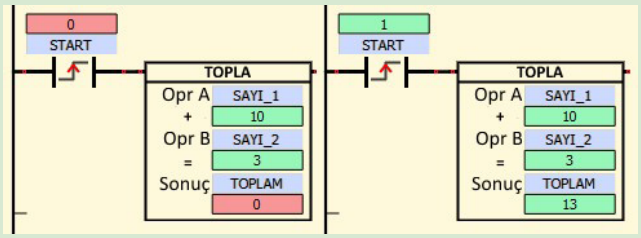
Operand tipleri : Word, double word, integer, real.



Çalışması :

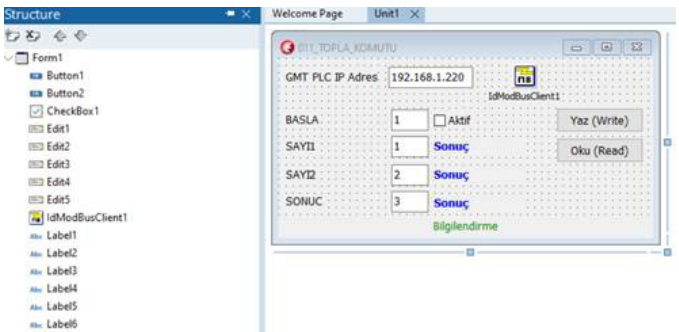
1	**Opr A:** Toplama işleminin yapılacağı sabit değer ya da operand adı yazılır.
2	**Opr B**: Toplama işleminin yapılacağı sabit değer ya da operand adı yazılır.
3	**Sonuç:** Sonucun kaydedileceği operandın adı yazılır.

Not: Sonuç operandının tipi, işlem sonucu kaydedilen operand tipinden olmalıdır (word, double word, integer, real).

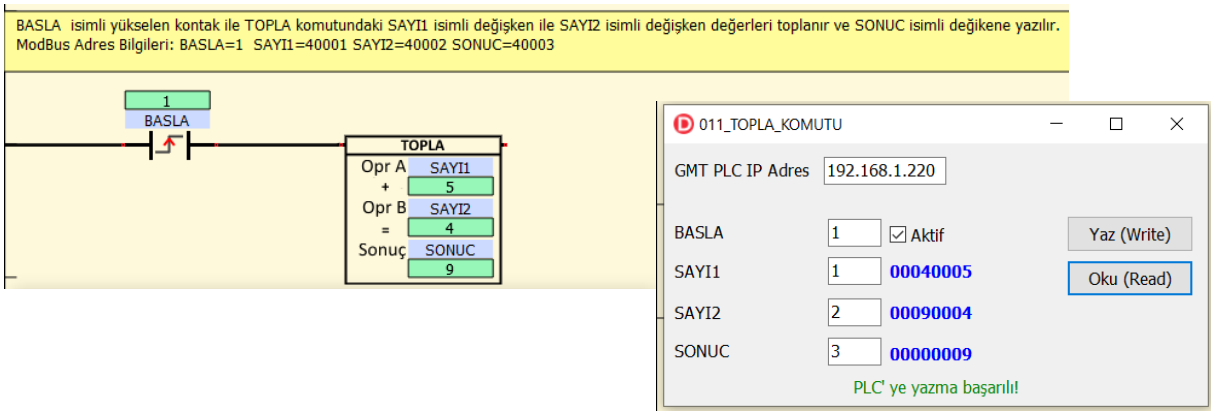


Örnek uygulama : Topla komutunun aktif olmasıyla **Opr A** değeri (SAYI_1) ile **Opr B** değeri (SAYI_2) matematiksel olarak toplanır. Toplama sonucu, **Sonuç** operandının içeriğine (TOPLAM) yazılır.

1.11. UYGULAMA: Topla Komutu



Görsel 1.47: Topla komutu Delphi arayüzü



Görsel 1.48: Topla komutu ladder diyagramı

```
procedure TForm1.Button2Click(Sender: TObject);
var
  Data: array[0..10] of Word;
  sLine: String;
  okuma_sayisi,i: Integer;
begin
  okuma_sayisi :=2;
  if (okuma_sayisi > 0) then
  begin
    IdModBusClient1.Host := edit1.Text;
    if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit3.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label4.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
      end;
    end ;
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit4.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label7.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
      end;
    end;
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit5.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label9.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
      end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');
  end;
end;
```

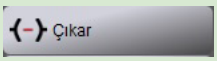
```
procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

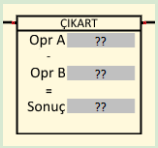
Görsel 1.49: Topla komutu program kodu

Çıkart Komutu

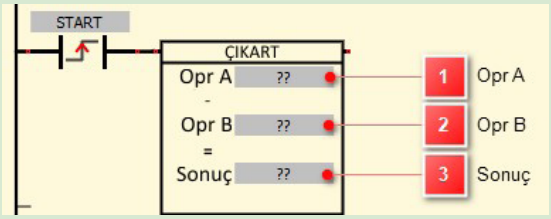
Kısa adı : ÇK

Kısa yol no. : Enter > 13

İkonu : 



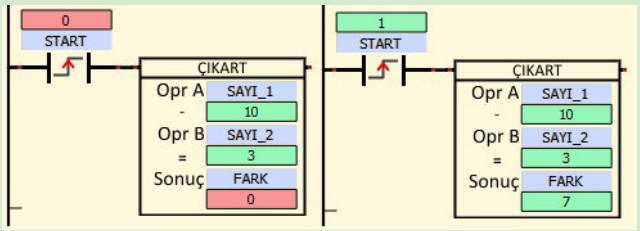
Operand tipleri : Word, double word, integer, real.



Çalışması :

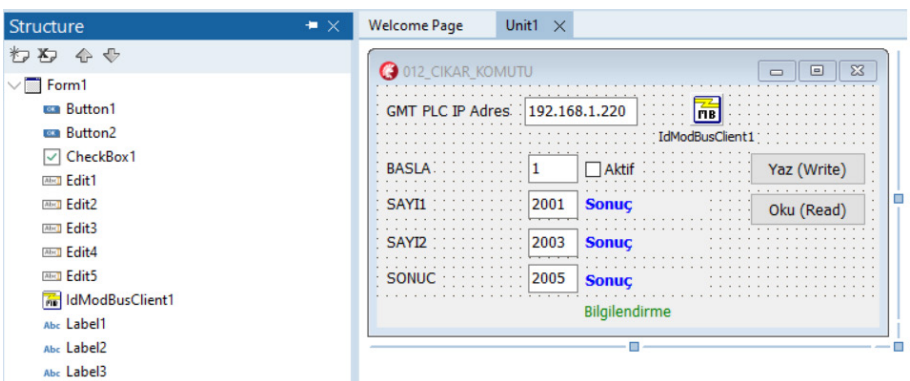
1	**Opr A:** Çıkartma işleminde çıkarılan sabit değer ya da operand adı yazılır.
2	**Opr B**: Çıkartma işleminde çıkaran sabit değer ya da operand adı yazılır.
3	**Sonuç:** Sonucun kaydedileceği operandın adı yazılır.

Not: Sonuç operandının tipi, işlem sonucu kaydedilen operand tipinden olmalıdır (word, double word, integer, real).



Örnek uygulama : Çıkart komutunun aktif olmasıyla **Opr A** değerinden (SAYI_1), **Opr B** değeri (SAYI_2) matematiksel olarak çıkartılır. Çıkartma işleminin sonucu, **Sonuç** operandının içeriğine (FARK) yazılır.

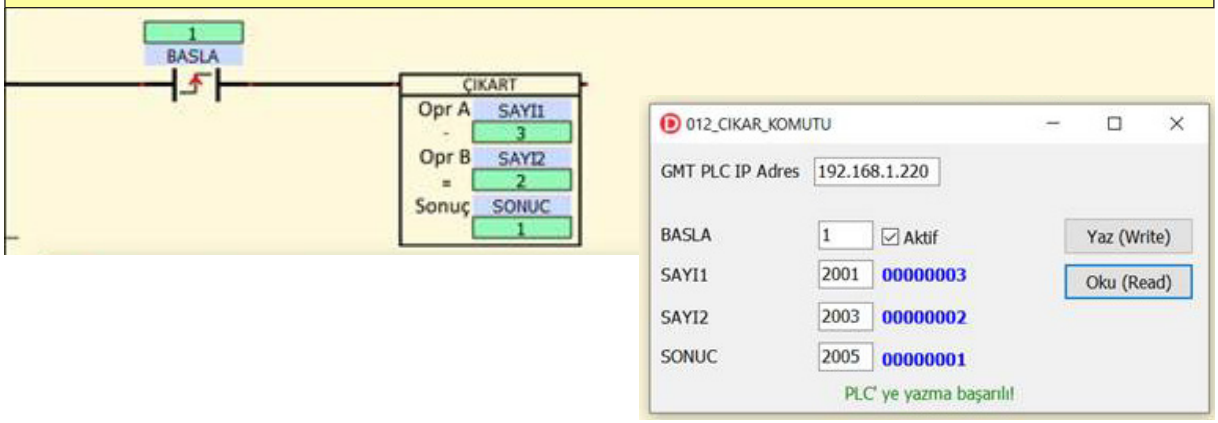
1.12. UYGULAMA: Çıkar Komutu



Görsel 1.50: Çıkar komutu Delphi arayüzü

BASLA isimli yükselen kontak ile ÇIKART komutundaki SAYI1 isimli değişkenden SAYI2 isimli değişken değeri çıkarılır ve SONUC isimli değişkene yazılır. İşlem yapılacak değerler sonucu alınabilecek değere göre değişken tipi seçilmelidir.

ModBus Adres Bilgileri: BASLA=1 SAYI1=42001 SAYI2=42003 SONUC=42005



Görsel 1.51: Çıkar komutu ladder diyagramı

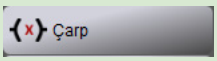
```
procedure TForm1.Button2Click(Sender: TObject);
var
  Data: array[0..10] of Word;
  sLine: String;
  okuma_sayisi,i: Integer;
begin
  okuma_sayisi :=2;
  if (okuma_sayisi > 0) then
  begin
    IdModBusClient1.Host := edit1.Text;
    if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit3.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label4.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
      end;
    end ;
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit4.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label7.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
      end;
    end;
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit5.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label9.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
      end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');
  end;
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

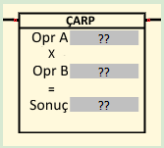
Görsel 1.52: Çıkar komutu program kodu

Çarp Komutu

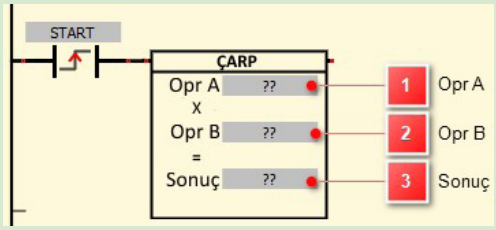
Kısa adı : ÇR

Kısa yol no. : Enter > 14

İkonu : 



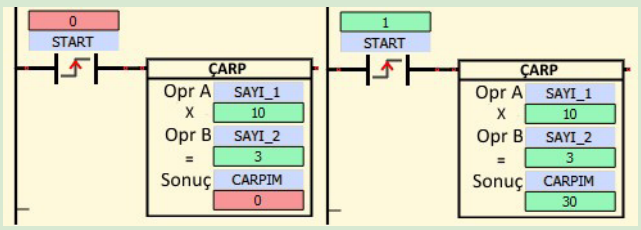
Operand tipleri : Word, double word, integer, real.



Çalışması :

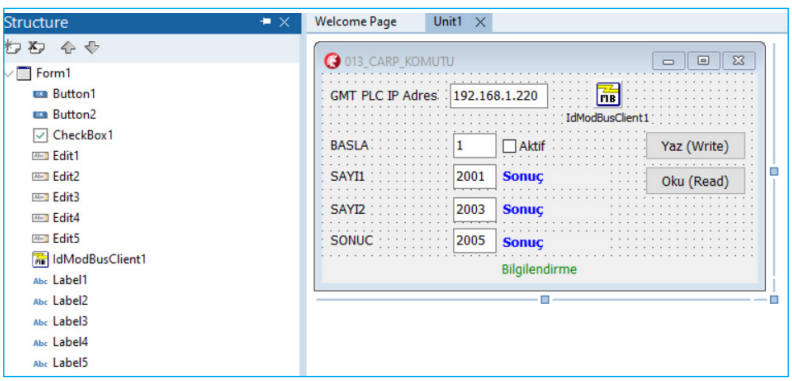
1	**Opr A:** Çarpım işleminin yapılacağı sabit değer ya da operand adı yazılır.
2	**Opr B**: Çarpım işleminin yapılacağı sabit değer ya da operand adı yazılır.
3	**Sonuç:** Sonucun kaydedileceği operandın adı yazılır.

Not: Sonuç operandının tipi, işlem sonucu kaydedilen operand tipinden olmalıdır (word, double word, integer, real).

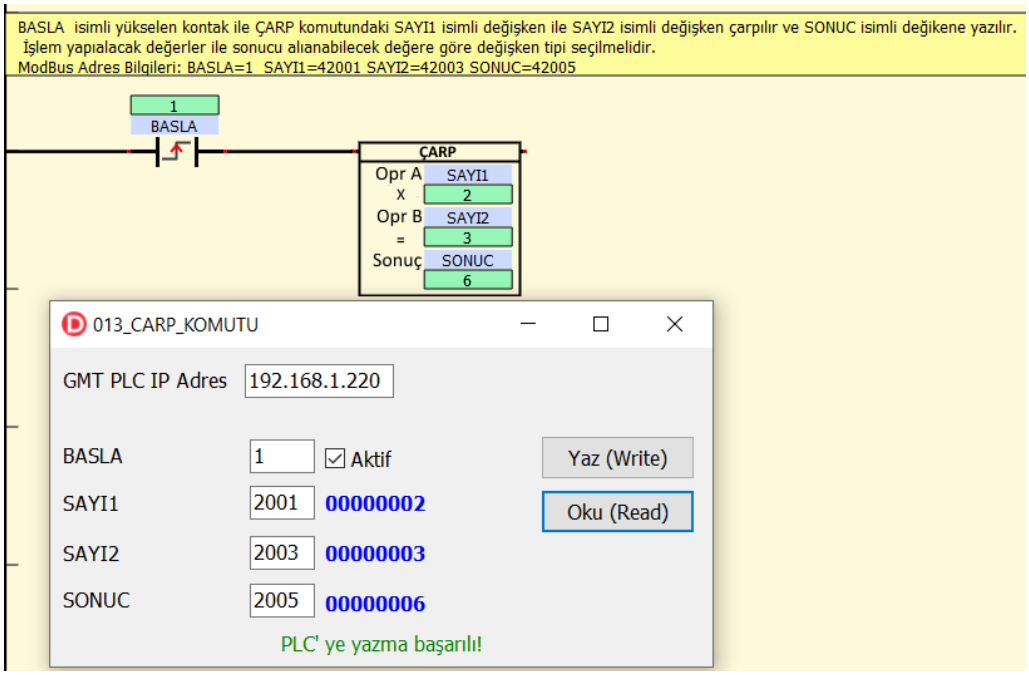


Örnek uygulama : Çarpma komutunun aktif olmasıyla **Opr A** değeri (SAYI_1) ile **Opr B** değeri (SAYI_2) matematiksel olarak çarpılır. Çarpım sonucu, **Sonuç** operandının içeriğine (CARPIM) yazılır.

1.13. UYGULAMA: Çarp Komutu



Görsel 1.53: Çarp komutu Delphi arayüzü



Görsel 1.54: Çarp komutu ladder diyagramı

```
procedure TForm1.Button2Click(Sender: TObject);
var
  Data: array[0..10] of Word;
  sLine: String;
  okuma_sayisi,i: Integer;
begin
  okuma_sayisi :=2;
  if (okuma_sayisi > 0) then
  begin
    IdModBusClient1.Host := edit1.Text;
    if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit3.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label4.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
      end;
    end ;
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit4.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label7.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
      end;
    end;
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit5.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label9.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
      end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');
  end;
end;
```

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

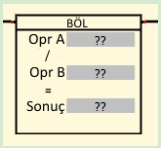
Görsel 1.55: Çarp komutu program kodu

Böl Komutu

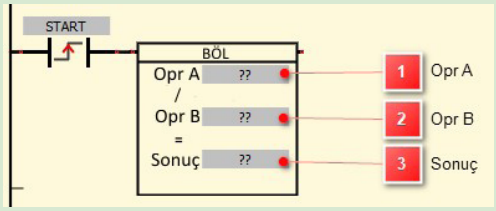
Kısa adı : BÖ

Kısa yol no. : Enter > 15

İkonu :



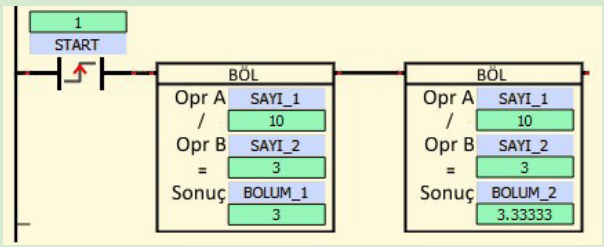
Operand tipleri : Word, double word, integer, real.



Çalışması :

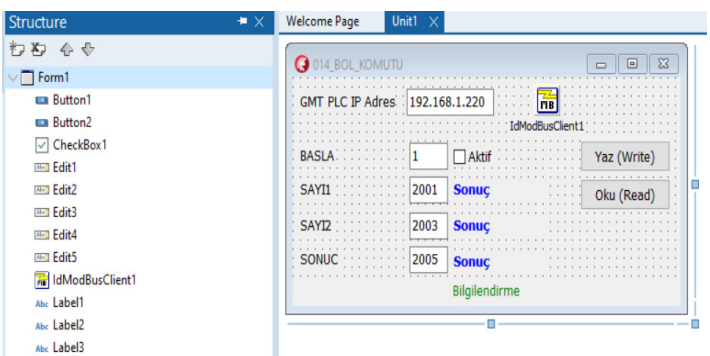
1	**Opr A:** Bölüm işleminde bölünen sabit değer ya da operand adı yazılır.
2	**Opr B**: Bölüm işleminde bölen sabit değer ya da operand adı yazılır.
3	**Sonuç:** Sonucun (bölüm) kaydedileceği operandın adı yazılır.

Not: Sonuç operandının tipi, işlem sonucu kaydedilen operand tipinden olmalıdır (word, double word, integer, real).

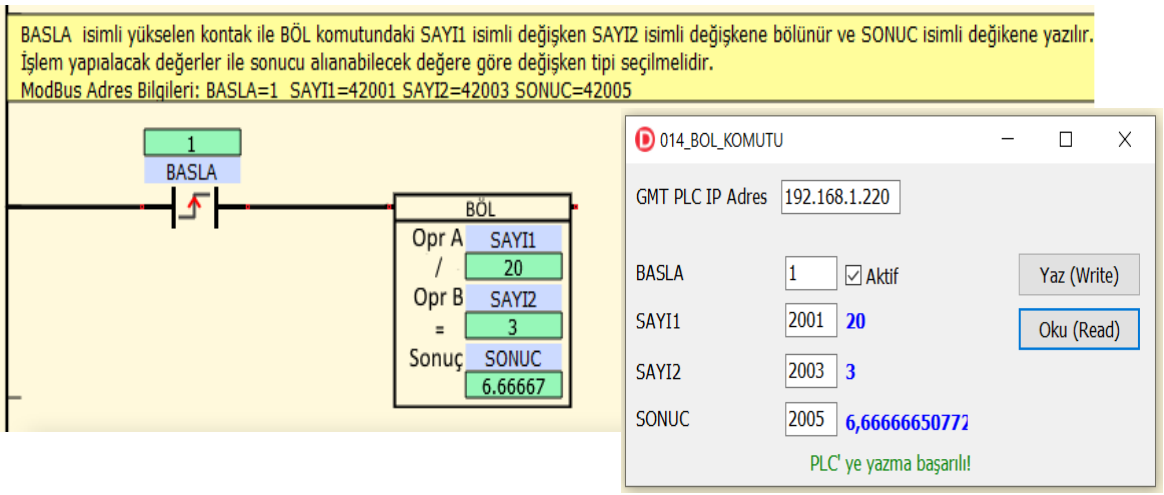


Örnek uygulama : Bölme komutunun aktif olmasıyla **Opr A** değeri (SAYI_1) ile **Opr B** değeri (SAYI_2) matematiksel olarak bölünür. Bölme işleminin sonucu, **Sonuç** operandının içeriğine (BOLUM_X) yazılır. **BOLUM_1** operandının tipi word olarak, **BOLUM_2** operandının tipi ise real olarak tanımlanır.

1.14. UYGULAMA: Böl Komutu



Görsel 1.56: Böl komutu Delphi arayüzü



Görsel 1.57: Böl komutu ladder diyagramı

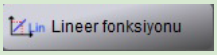
```
procedure TForm1.Button2Click(Sender: TObject);
var
  Data: array[0..10] of Word;
  sLine: String;
  okuma_sayisi,i: Integer;
begin
  okuma_sayisi :=2;
  if (okuma_sayisi > 0) then
  begin
    IdModBusClient1.Host := edit1.Text;
    if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit3.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label4.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
      label4.Caption:=FloatToStr(HexToIEEE754(label4.Caption));
      end;
    end ;
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit4.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label7.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
      label7.Caption:=FloatToStr(HexToIEEE754(label7.Caption));
      end;
    end;
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit5.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label9.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
      label9.Caption:=FloatToStr(HexToIEEE754(label9.Caption));
      end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');
  end;
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

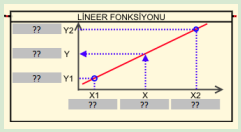
Görsel 1.58: Böl komutu program kodu

Lineer Fonksiyon Komutu

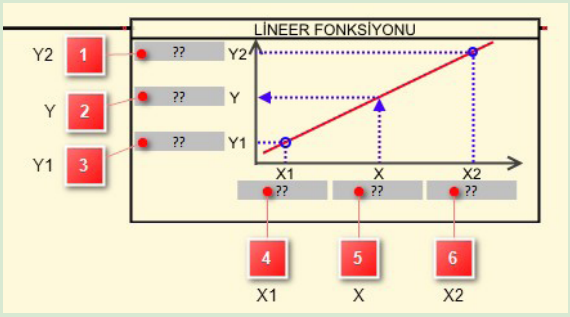
Kısa adı : LN

Kısa yol no. : Enter > 15

İkonu : 

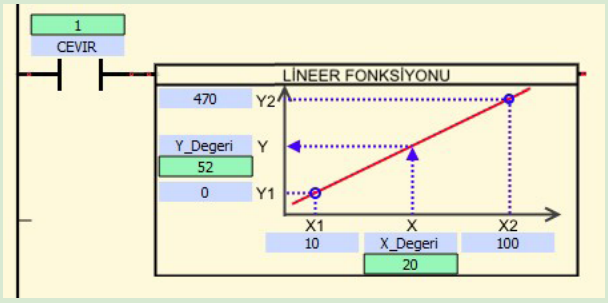


Operand tipleri : Word, double word, integer, real.



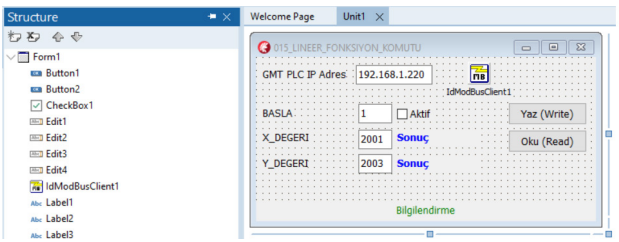
1	**Y2:** Y2 Operand adı ya da sabit değeri yazılır.
2	**Y:** İşlem sonucunun kaydedileceği operandın adı yazılır.
3	**Y1:** Y1 Operand adı ya da sabit değeri yazılır.
4	**X1:** X1 Operand adı ya da sabit değeri yazılır.
5	**X:** Hesaplanacak olan operand adı yazılır.
6	**X2:** X2 Operand adı ya da sabit değeri yazılır.

Çalışması : Lineer fonksiyon **X** operandının değeri ile **k** faktörünün [k = (Y2-Y1)/(X2-X1)] değeri çarpılır ve **Y** operandının içeriğine yazılır (Y = k X). Hesaplanan değer **X** operandının içeriğine yazılır. **Y_Değeri** operandının tipi, işlem sonucunu kaydedilen operand tipinde olmalıdır (word, double word, integer, real).

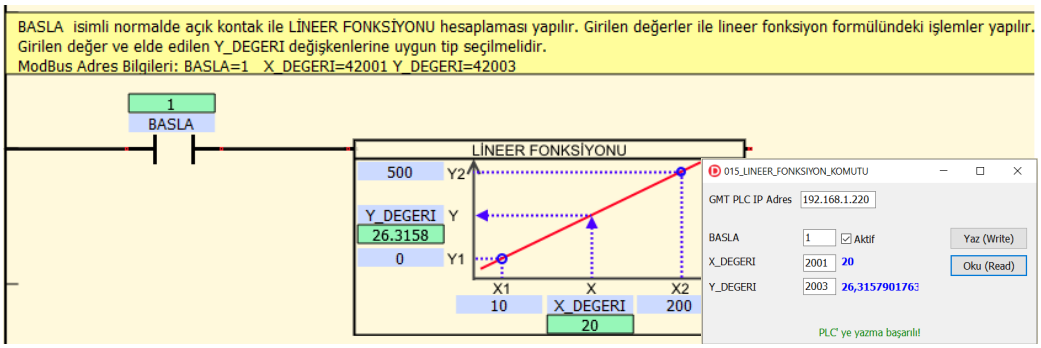


Örnek uygulama : **Y1 = 0, Y2 = 470, X1 = 10, X2 = 100** ve **X_Degeri = 20**'dir. Lineer fonksiyonu komutunun aktif edilmesiyle komut tarafından **Y** değeri 52 olarak hesaplanır. **X_Degeri**, **X2** değerinden büyük veya **X1** değerinden küçük değerde yazılabilir. Bu durumda **Y_Degeri** hesaplanır. Hesaplama sonucunun 0'dan küçük bir sayı çıkmasına karşın **Y_Degeri** operandının tipi real olarak tanımlanır.

1.15. UYGULAMA: LINEER FONKSIYON Komutu



Görsel 1.59: Lineer komutu Delphi arayüzü



Görsel 1.60: Lineer komutu ladder diyagramı

```
procedure TForm1.Button2Click(Sender: TObject);
 var
   Data: array[0..10] of Word;
   sLine: String;
   okuma_sayisi,i: Integer;
 begin
   okuma_sayisi :=2;
   if (okuma_sayisi > 0) then
   begin
     IdModBusClient1.Host := edit1.Text;
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit3.Text), okuma_sayisi, Data) then
     begin
       for i := 0 to (okuma_sayisi - 1) do
       begin
       label4.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
       label4.Caption:=FloatToStr(HexToIEEE754(label4.Caption));
       end;
     end ;
      if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit4.Text), okuma_sayisi, Data) then
     begin
       for i := 0 to (okuma_sayisi - 1) do
       begin
       label7.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
       label7.Caption:=FloatToStr(HexToIEEE754(label7.Caption));
       end;
     end
     else
       ShowMessage('PLC''den okuma başarısız!');
   end;
 end;
procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

Görsel 1.61: Lineer komutu program kodu

1.1.6. Matematik Komutları 2

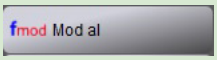
Bu bölümde açıklanan komutlar, GLC serisinin PLC modellerinde kullanılır. Bu komutlar veri üzerinde ileri matematik işlemlerini yapan fonksiyonlardır. Fonksiyon sonuçları için real tip operand tanımlanır.

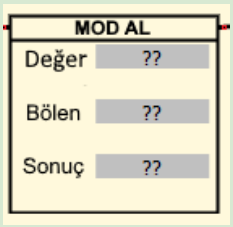
Mod Alma Fonksiyonu

Kısa adı : MO

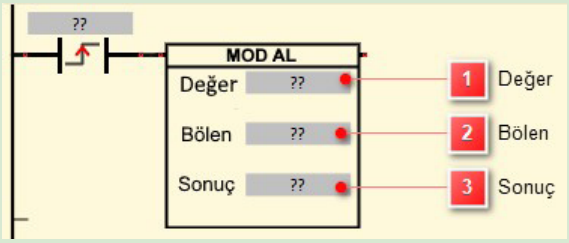
Kısa yol no. : Enter > 105

İkonu :





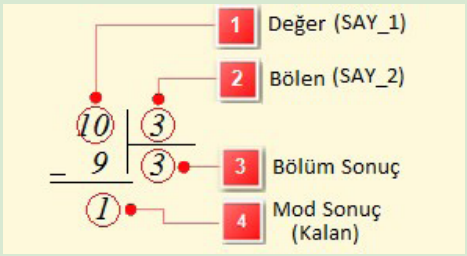
Operand tipleri : Word, double word, integer, real.

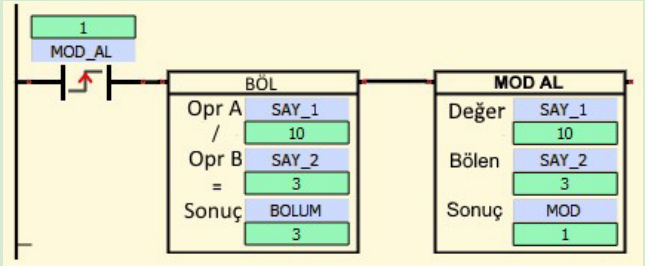


Çalışması :

1	**Değer:** Mod Alma işleminde bölünen operandın adı yazılır.
2	**Bölen:** Mod Alma işleminde bölen operandın adı yazılır.
3	**Sonuç:** Mod Alma işleminde kalan sonucu kaydedilen operandın adı yazılır.

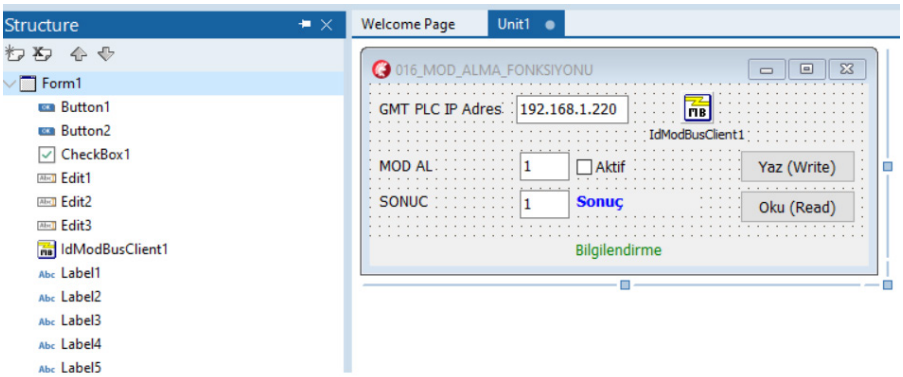
Not: Sonuç operandının tipi, işlem sonucu kaydedilen operand tipinden olmalıdır (word, double word, integer, real).



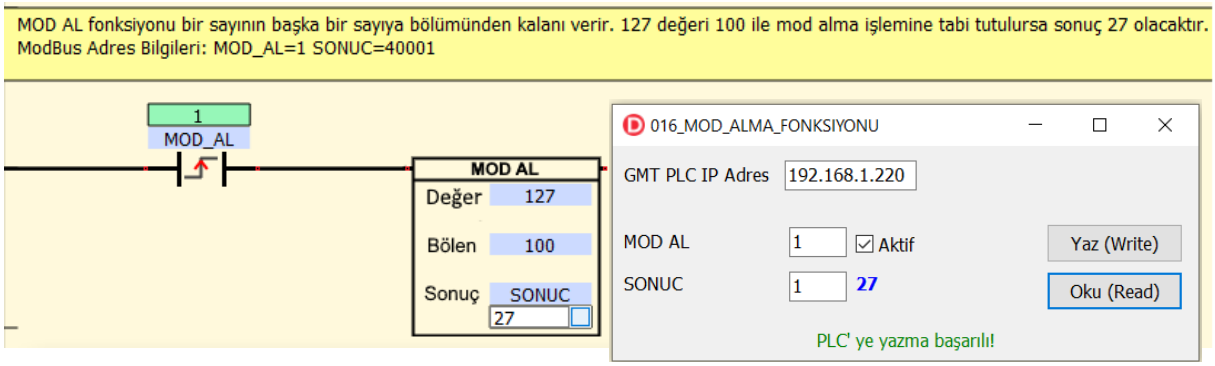


Örnek uygulama : Mod al komutunun aktif olmasıyla, **Değer** değeri (SAYI_1), **Bölen** (SAYI_2) değeriyle bölünür. Bölme işlemi sonucundan kalan değer, **Sonuç** operandına (MOD) yazılır. Örnek uygulamada BÖL komutu ile bölüm işleminin tam kısmı alınır. **MOD AL** komutu ile sonucun kalan kısmı alınır.

1.16. UYGULAMA: Mod Alma



Görsel 1.62: Mod alma komutu Delphi arayüzü



Görsel 1.63: Mod alma komutu ladder diyagramı

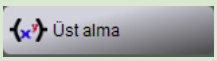
```
procedure TForm1.Button2Click(Sender: TObject);
var
  Data: array[0..10] of Word;
  sLine: String;  okuma_sayisi,i: Integer;
begin
  okuma_sayisi :=2;
  if (okuma_sayisi > 0) then
  begin
    IdModBusClient1.Host := edit1.Text;
    if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit3.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label4.Caption:=Inttostr(Data[0]);
      end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');
  end;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

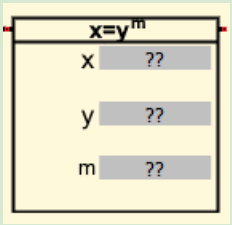
Görsel 1.64: Mod alma komutu program kodu

Üst Alma Fonksiyonu

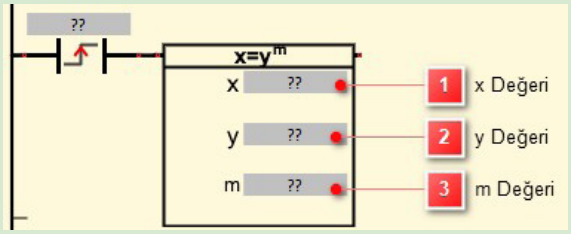
Kısa adı : ÜA

Kısa yol no. : Enter > 139

İkonu : 



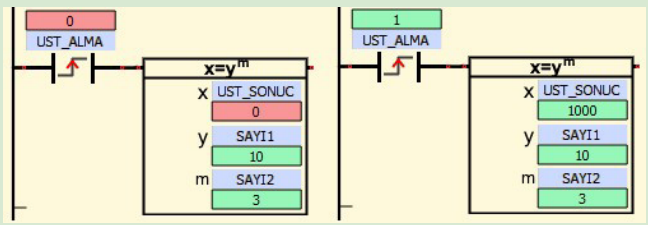
Operand tipleri : Word, double word, integer, real.



Çalışması :

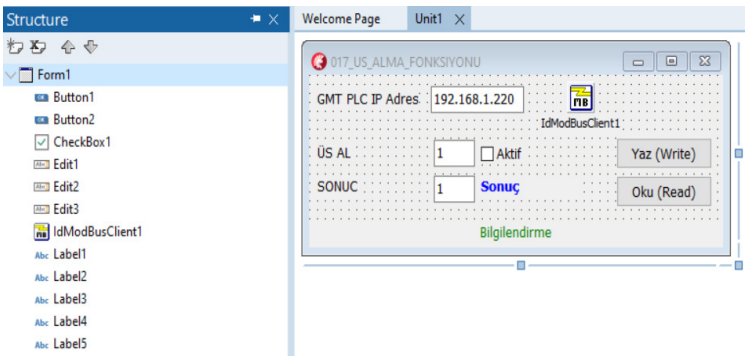
1	**x:** Üst alma işleminde sonuç kaydedilen operand adı
2	**y:** Üst alma fonksiyonunun taban sayısı
3	**m:** Üst alma fonksiyonunun kuvvet sayısı

Not: X operandının tipi, işlem sonucu kaydedilen operand tipinden olmalıdır (word, double word, integer, real).

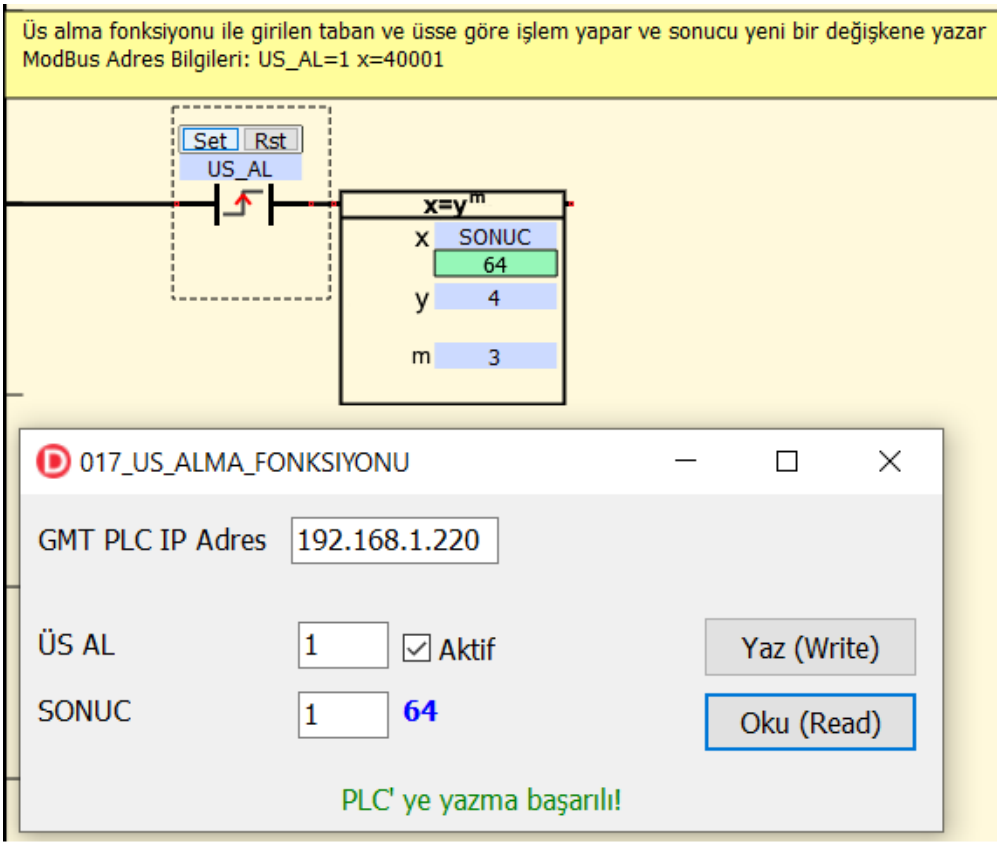


Örnek uygulama : Üst Alma Komutunun aktif olmasıyla 10'un 3. kuvvetinin sonucu **UST_SONUC** operandına yazılır.

1.17. UYGULAMA: Üst Alma Komutu



Görsel 1.65: Üst alma komutu Delphi arayüzü



Görsel 1.66: Üst alma komutu ladder diyagramı

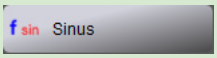
```
procedure TForm1.Button2Click(Sender: TObject);
var
  Data: array[0..10] of Word;
  sLine: String;  okuma_sayisi,i: Integer;
begin
  okuma_sayisi :=2;
  if (okuma_sayisi > 0) then
  begin
    IdModBusClient1.Host := edit1.Text;
    if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit3.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label4.Caption:=Inttostr(Data[0]);
      end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');
  end;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

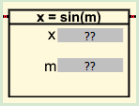
Görsel 1.67: Üst alma komutu program kodu

Sinüs Fonksiyonu

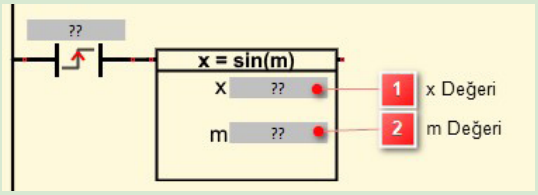
Kısa adı : SI

Kısa yol no. : Enter > 129

İkonu : 

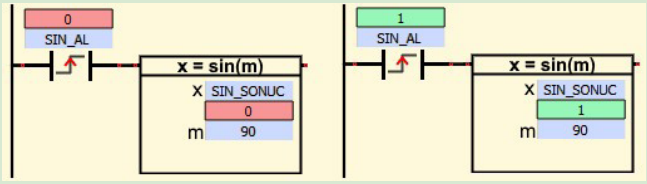


Operand tipleri : Real



Çalışması :

1	**x:** Sinüs alma işleminde sonuç kaydedilen operandın adı yazılır.
2	**m:** Değeri, derece biçimindedir.

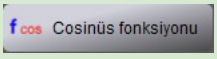


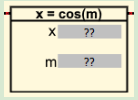
Örnek uygulama : Sinüs fonksiyon komutunun aktif olmasıyla 90°nin sinüsü **SIN_SONUC** operandına yazılır.

Cosinüs Fonksiyonu

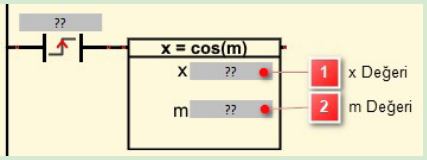
Kısa adı : CO

Kısa yol no. : Enter > 130

İkonu : 

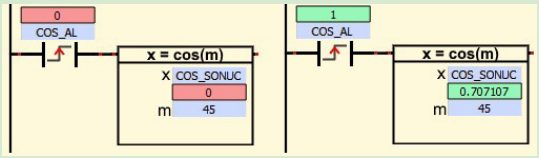


Operand tipleri : Real



Çalışması :

1	**x:** Cosinüs alma işleminde sonuç kaydedilen operandın adı yazılır.
2	**m:** Değeri, derece biçimindedir.

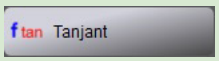


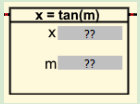
Örnek uygulama : Cosinüs fonksiyon komutunun aktif olmasıyla, 45°nin cosinüsü **COS_SONUC** operandına yazılır.

Tanjant Fonksiyonu

Kısa adı : TA

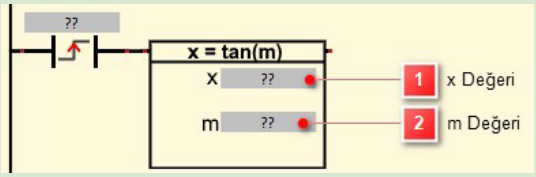
Kısa yol no. : Enter > 131

İkonu : 



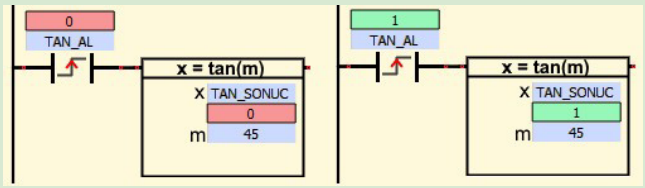
Operand tipleri : Real

Çalışması :



1	**x:** Tanjant alma işleminde sonuç kaydedilen operandın adı yazılır.
2	**m:** Değeri, derece biçimindedir.

Örnek uygulama :

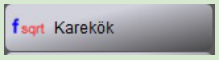


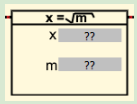
Tanjant fonksiyon komutunun olmasıyla 45°nin tanjantı **TAN_SONUC** operandına yazılır.

Karakök Fonksiyonu

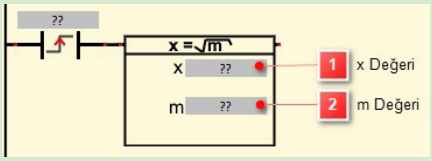
Kısa adı : KK

Kısa yol no. : Enter > 135

İkonu : 

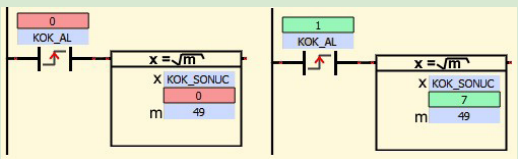


Operand tipleri : Real



Çalışması :

1	**x:** Karekök alma işleminde sonuç kaydedilen operandın adı yazılır.
2	**m:** Karekök fonksiyonu için operand adı ya da sabit değer yazılır.



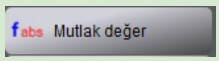
Örnek uygulama : Karekök fonksiyon komutunun aktif olmasıyla **49** sayısının karekökü **KOK_SONUC** operandına yazılır.

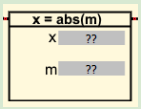
Mutlak Değer Fonksiyonu

Kısa adı : MD

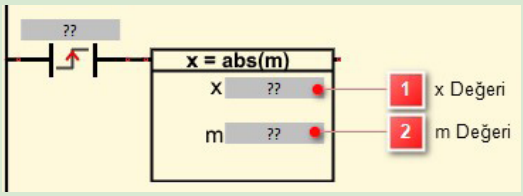
Kısa yol no : Enter > 136

İkonu :



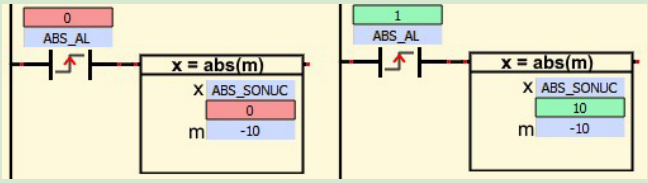


Operand tipleri : Real



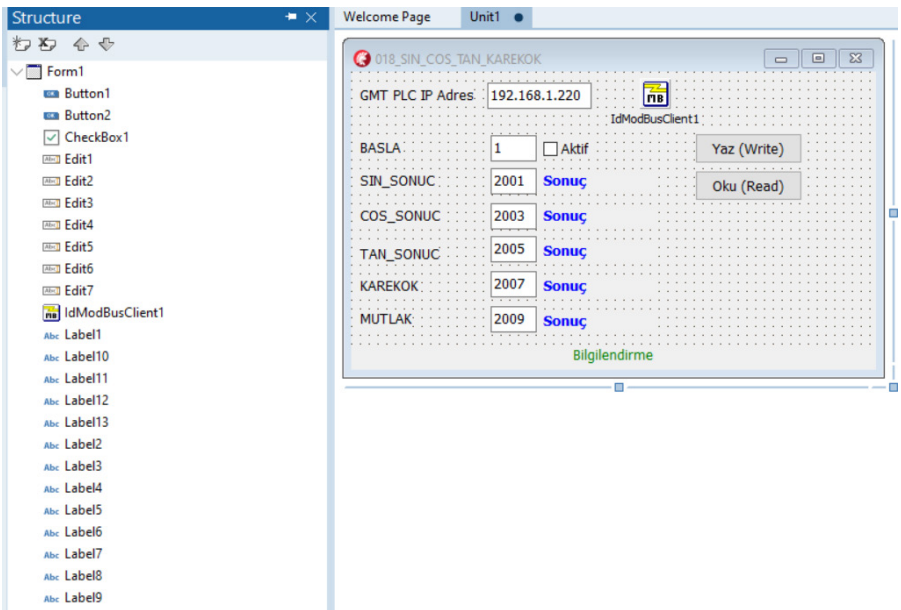
Çalışması :

1	**x:** Mutlak değer alma işleminde sonuç kaydedilen operandın adı yazılır.
2	**m:** Mutlak değer fonksiyonu için operand adı ya da sabit değer yazılır.



Örnek uygulama : Mutlak değer alma (abs) komutunun aktif olmasıyla, **-10**'un mutlak değeri **ABS_SONUC** operandına yazılır.

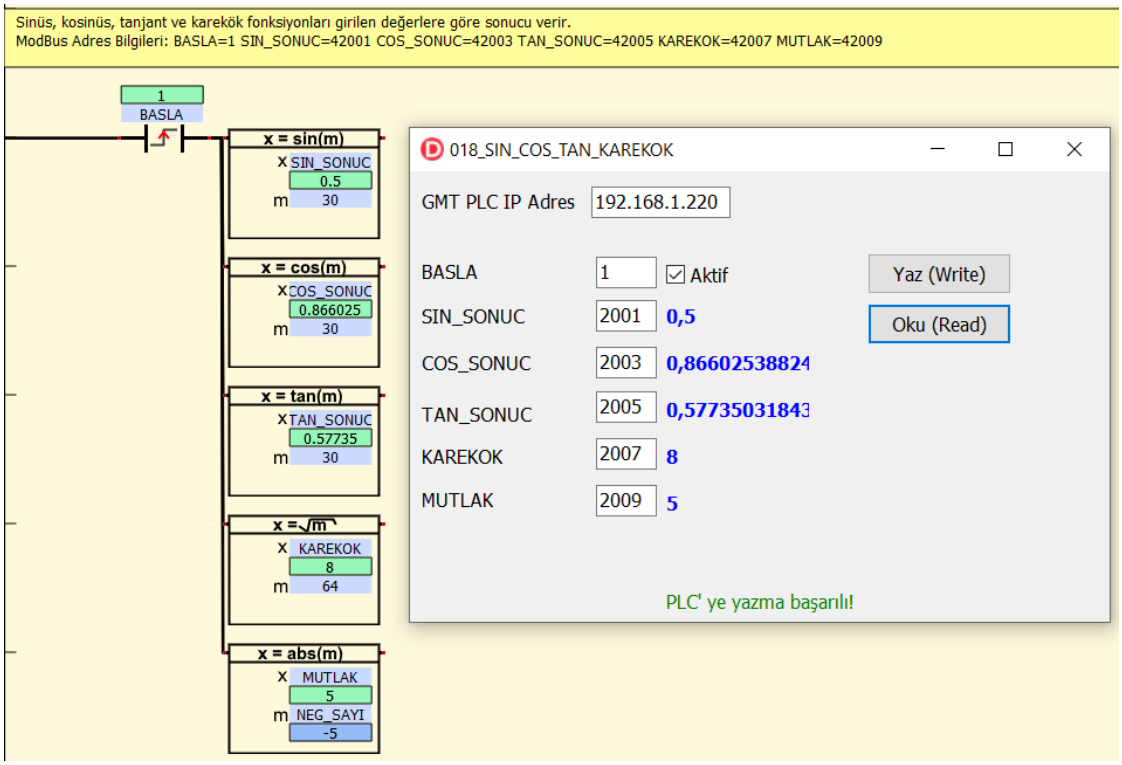
1.18. UYGULAMA: Sinüs, Kosinüs, Tanjant, Karekök ve Mutlak Değer Komutları



Görsel 1.68: Sinüs, kosinüs, tanjant, karekök ve mutlak değer komutları Delphi arayüzü

```
procedure TForm1.Button2Click(Sender: TObject);
 var
   Data: array[0..10] of Word;
   sLine: String;
   okuma_sayisi,i: Integer;
 begin
   okuma_sayisi :=2;
   if (okuma_sayisi > 0) then
   begin
     IdModBusClient1.Host := edit1.Text;
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit3.Text), okuma_sayisi, Data) then
     begin
       for i := 0 to (okuma_sayisi - 1) do
       begin
       label4.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
       label4.Caption:=FloatToStr(HexToIEEE754(label4.Caption));
       end;
     end ;
      if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit4.Text), okuma_sayisi, Data) then
     begin
       for i := 0 to (okuma_sayisi - 1) do
       begin
       label7.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
       label7.Caption:=FloatToStr(HexToIEEE754(label7.Caption));
       end;
   end;
    if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit5.Text), okuma_sayisi, Data) then
   begin
     for i := 0 to (okuma_sayisi - 1) do
     begin
     label9.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
     label9.Caption:=FloatToStr(HexToIEEE754(label9.Caption));
     end;
   end ;
   if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit6.Text), okuma_sayisi, Data) then
   begin
     for i := 0 to (okuma_sayisi - 1) do
     begin
     label11.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
     label11.Caption:=FloatToStr(HexToIEEE754(label11.Caption));
     end;
   end ;
   if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit7.Text), okuma_sayisi, Data) then
   begin
     for i := 0 to (okuma_sayisi - 1) do
     begin
     label13.Caption:=inttostr(Data[0]);
     end;
   end
    else
       ShowMessage('PLC''den okuma başarısız!');
   end;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin
   IdModBusClient1.Host := Edit1.Text;
     if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
     Label5.Caption := 'PLC'' ye yazma başarılı!'
   else
     Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

Görsel 1.69: Sinüs, kosinüs, tanjant, karekök ve mutlak değer komutları program kodu



Görsel 1.70: Sinüs, kosinüs, tanjant, karekök ve mutlak değer komutları ladder diyagramı

1.2. ANALOG VE KARŞILAŞTIRMA İŞLEMLERİ, ÇEVİRME İŞLEMLERİ

Analog değerlerin okunması ve karşılaştırılması işlemleri bu bölümdeki komutlarla yapılır.

1.2.1. Analog Komutlar

Bu bölümde açıklanan komutlar, 296X ve üstü GLC serisinin PLC modellerinde kullanılır. Bu model CPU'larda bir analog giriş ve bir de çıkış bulunur. Giriş tipleri 0-10V DC, 0-20mA ve 4-20mA'dir. Çıkış tipleri 0-10V DC ve 0-20mA'dir. Analog çıkış lineer olarak 0 ... 16.383 değere karşılık sinyal üretir. Analog girişin skala değeri ise 0 ... 4.095 arasında ayarlanır.

Yineleme hızı PLC ladder tarama hızı ile aynıdır. Birden fazla giriş/çıkışa ihtiyaç duyulması hâlinde ek analog modül kullanılır ve "konfigürasyon" kısmında bu modül/modüller tanımlanır. Band çıkış komutu, üniversal ve loadcell modüllerle uyumludur. Analog verinin (sıcaklık, ağırlık, basınç, hız, mesafe vb.) algılanmasında ve analog çıkışla kontrol edilen otomasyon uygulamalarında bu komutlar kullanılır.

Örnek

Set = 100

Band = 10

Band/2 = 10/2 = 5

Üst Band = Set + 5=100+5 = 105

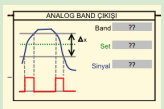
Alt Band = Set – 5 = 100-5 = 95

Analog Band Komutu

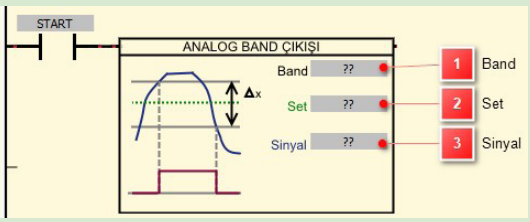
Kısa adı : AB

Kısa yol no. : Enter > 37

İkonu : 

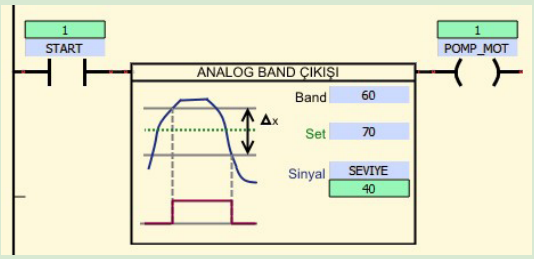


Operand tipleri : Word, double word, integer, real.



Çalışması :

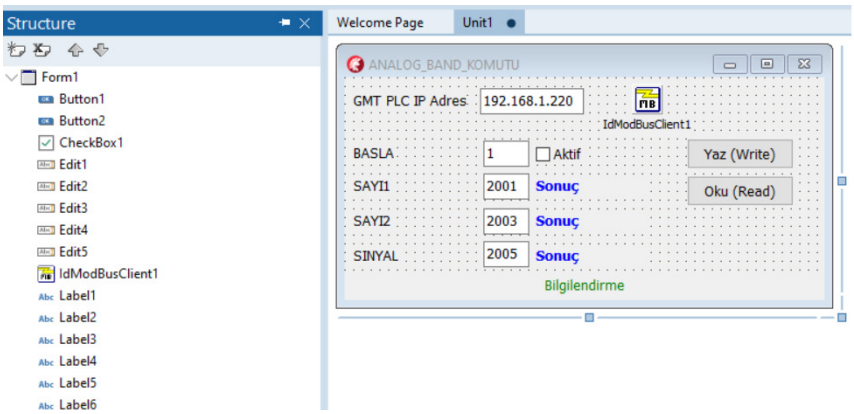
1	**Band:** Komutun çıkış vereceği band aralığı yazılır.
2	**Set: Sinyal** operandı, **Set - Band/2** değeri ile **Set + Band/2** değeri arasındaysa sonraki komut çalıştırılır.
3	**Sinyal:** İşlem sonucunun kaydedileceği operandın adı yazılır.



(70 - 60 / 2 = **40**) (70 + 60 / 2 = **100**)

Örnek uygulama : **POMP_MOT** adlı direkt çıkış rölesi, **SEVIYE** operandının 40 ile 100 değerleri arasında çalışır.

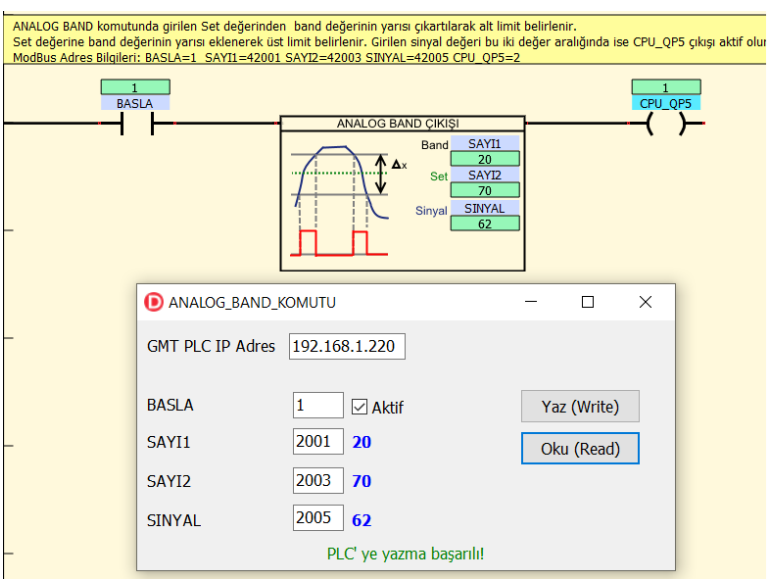
1.19. UYGULAMA: Analog Band Komutu



Görsel 1.71: Analog band komutu Delphi arayüzü

```
procedure TForm1.Button2Click(Sender: TObject);
 var
   Data: array[0..10] of Word;
   sLine: String;
   okuma_sayisi,i: Integer;
 begin
   okuma_sayisi :=2;
   if (okuma_sayisi > 0) then
   begin
     IdModBusClient1.Host := edit1.Text;
      if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit3.Text), okuma_sayisi, Data) then
     begin
       for i := 0 to (okuma_sayisi - 1) do
      begin
      label4.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
      label4.Caption:=FloatToStr(HexToIEEE754(label4.Caption));
      end;
     end ;
      if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit4.Text), okuma_sayisi, Data) then
     begin
       for i := 0 to (okuma_sayisi - 1) do
      begin
      label7.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
      label7.Caption:=FloatToStr(HexToIEEE754(label7.Caption));
      end;
     end;
      if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit5.Text), okuma_sayisi, Data) then
     begin
       for i := 0 to (okuma_sayisi - 1) do
      begin
      label9.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
      label9.Caption:=FloatToStr(HexToIEEE754(label9.Caption));
      end;
     end
     else
       ShowMessage('PLC''den okuma başarısız!');
   end;
 end;
 procedure TForm1.Button1Click(Sender: TObject);
 begin
   IdModBusClient1.Host := Edit1.Text;
     if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
     Label5.Caption := 'PLC'' ye yazma başarılı!'
   else
     Label5.Caption := 'PLC''ye yazma başarısız!';
 end;
```

Görsel 1.72: Analog band komutu program kodu

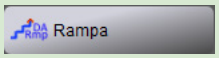


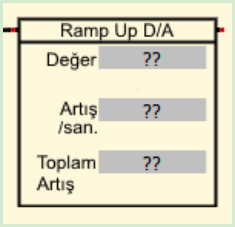
Görsel 1.73: Analog band komutu ladder diyagramı

Rampa Komutu

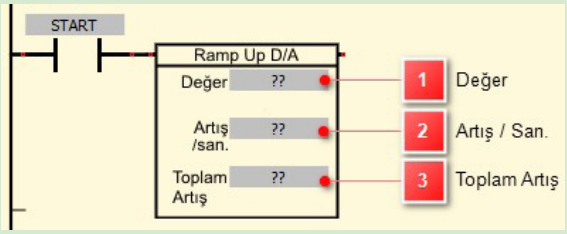
Kısa adı : RM

Kısa yol no. : Enter > 32

İkonu : 

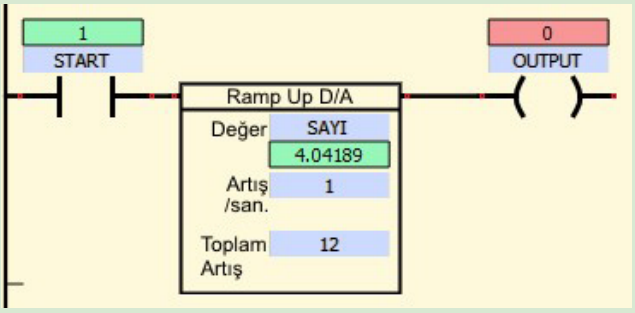


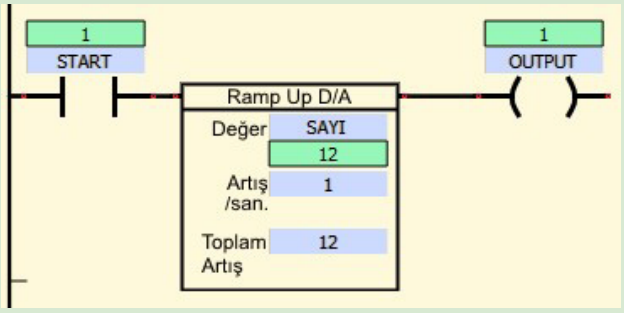
Operand tipleri : Word, double word, integer, real.



Çalışması :

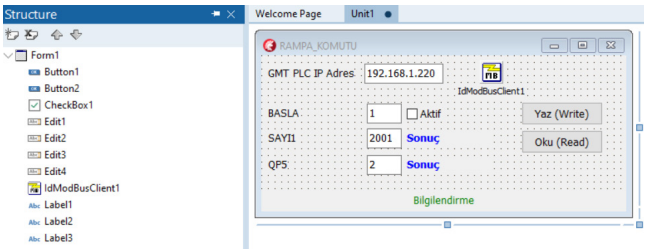
1	**Değer:** Rampa komutunun uygulandığı operandın adı yazılır.
2	**Artış/sn. :** Bir saniyedeki artış miktarı yazılır. Örneğin, **Toplam Artış** (hedef) değeri 10 ve bu değere 20 sn.'de ulaşılırsa **Artış/sn.** değeri (10/20=0,5) girilir.
3	**Toplam Artış:** Ulaşılacak hedef değer, operand adı veya sabit değer yazılır.





Örnek uygulama : Rampa komutunun aktif olmasıyla **Değer** içinde belirtilen operand, her bir saniyede **Artış/sn.** de belirtilen değer kadar artarak **Toplam Artış**ta belirtilen hedefe ulaşır. Ve hedef değere ulaştığında sonraki komutlar yürütülür. **Artış/sn.** için 0 ile 1 arasında bir değer seçilerek çok hassas bir artış elde edilir (örneğin 0.0001).

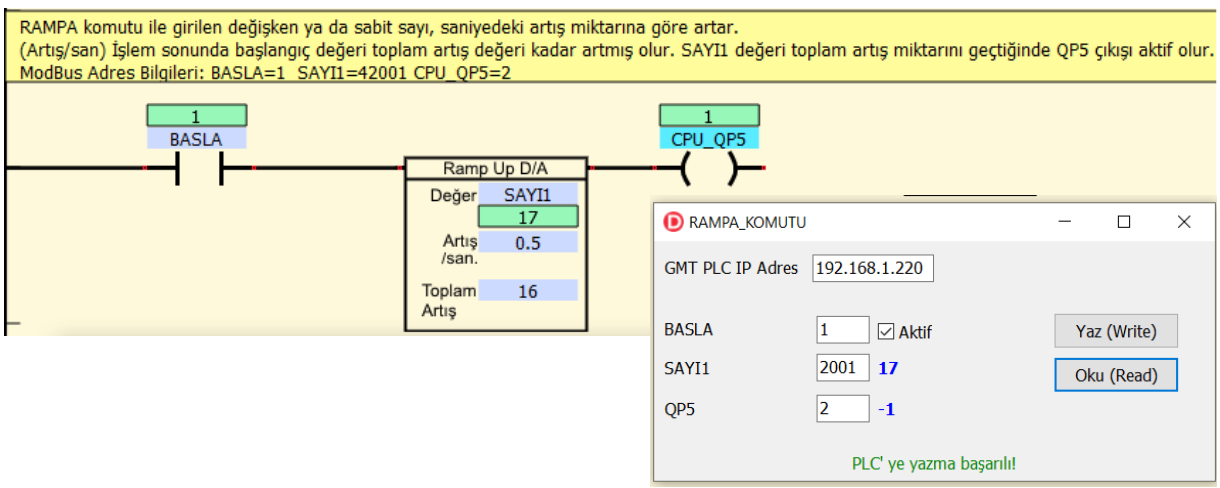
1.20. UYGULAMA: Rampa Komutu



Görsel 1.74: Rampa komutu Delphi arayüzü

```
procedure TForm1.Button2Click(Sender: TObject);
var
  Data: array[0..10] of Word;
  sLine: String;
  okuma_sayisi,i: Integer;
  SONUC: Boolean;
begin
  okuma_sayisi :=2;
  if (okuma_sayisi > 0) then
  begin
    IdModBusClient1.Host := edit1.Text;
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit3.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label4.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
      label4.Caption:=FloatToStr(HexToIEEE754(label4.Caption));
      end;
    end ;
    if IdModBusClient1.ReadCoil(StrToInt(Edit4.Text), SONUC) then
    Label7.Caption := BoolToStr(SONUC)
    else
      ShowMessage('PLC''den okuma başarısız!');
  end;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

Görsel 1.75: Rampa komutu program kodu



Görsel 1.76: Rampa komutu ladder diyagramı

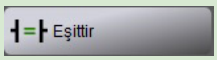
1.2.2. Karşılaştırma Komutları

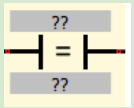
Bu bölümde açıklanan komutlar, GLC ve GSR serisinin PLC modellerinde kullanılır. GMTSuite programında yedi çeşit karşılaştırma komutu bulunur. Temel karşılaştırma komutları iki bellek konumunda saklanan değerleri karşılaştırır. Bu iki değer, iki farklı operand olabilir veya biri operand diğeri sabit bir değer olabilir. Şartın gerçekleşmesi hâlinde sonraki komutları çalıştırır.

Eşittir Komutu

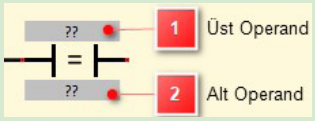
Kısa adı : ET

Kısa yol no. : Enter > 16

İkonu : 

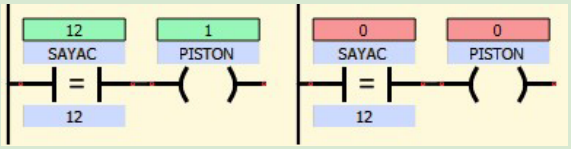


Operand tipleri : Word, double word, integer, real.



Çalışması :

1	**Üst Operand:** Sorgulanan operandın adı yazılır.
2	**Alt Operand:** Sorgulamada baz alınacak sabit değer veya operandın adı yazılır.

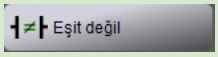


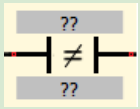
Örnek uygulama : Üst operandın değeri, alt operandın değerine eşitse bir sonraki komutu çalıştıran işlemdir. Aksi takdirde sonraki komut çalışmaz.

Eşit Değil Komutu

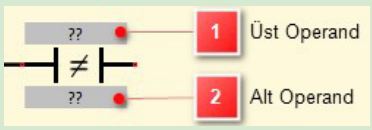
Kısa adı : ED

Kısa yol no. : Enter > 17

İkonu : 

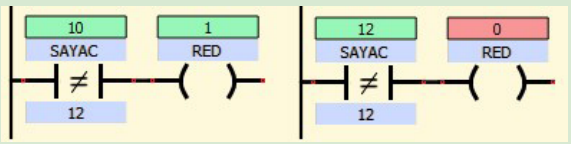


Operand tipleri : Word, double word, integer, real.



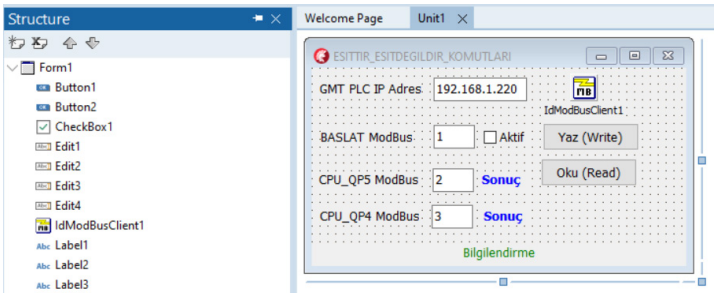
Çalışması :

1	**Üst Operand:** Sorgulanacak olan operand adı yazılır.
2	**Alt Operand:** Sorgulamada baz alınacak sabit değer veya operandın adı yazılır.



Örnek uygulama : Üst operandın değeri, alt operandın değerine eşit değilse bir sonraki komutu çalıştıran işlemdir. Aksi takdirde sonraki komut çalışmaz.

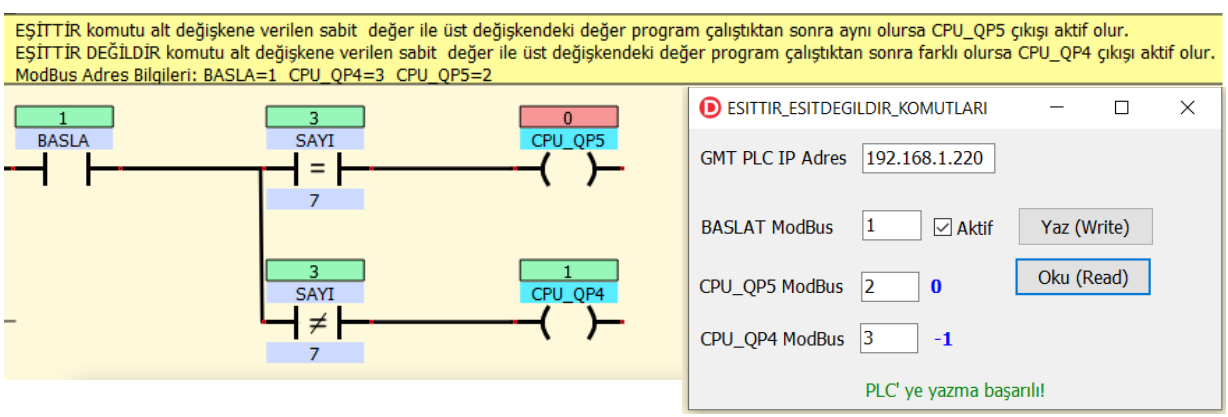
1.21. UYGULAMA: Eşittir ve Eşit Değildir Komutları



Görsel 1.77: Eşittir ve eşit değildir komutları Delphi arayüzü

```
procedure TForm1.Button2Click(Sender: TObject);
var
  SONUC: Boolean;
begin
  IdModBusClient1.Host := Edit1.Text;

  if IdModBusClient1.ReadCoil(StrToInt(Edit3.Text), SONUC) then
     Label4.Caption := BoolToStr(SONUC)
  else
    Label5.Caption := 'PLC''den okuma başarısız!';
  if IdModBusClient1.ReadCoil(StrToInt(Edit4.Text), SONUC) then
     Label7.Caption := BoolToStr(SONUC)
  else
    Label5.Caption := 'PLC''den okuma başarısız!';
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
  if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

Görsel 1.78: Eşittir ve eşit değildir komutları program kodu

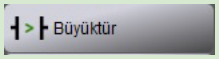


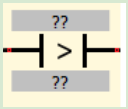
Görsel 1.79: Eşittir ve eşit değildir komutları ladder diyagramı

Büyüktür Komutu

Kısa adı : BÜ

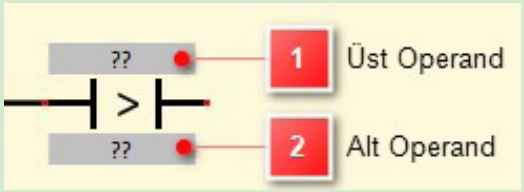
Kısa yol no. : Enter > 18

İkonu : 



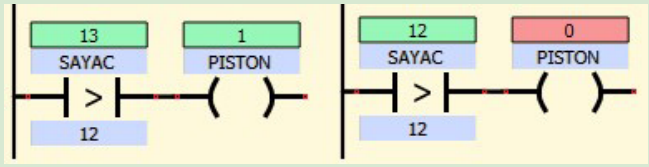
Operand tipleri : Word, double word, integer, real.

Çalışması :



1	**Üst Operand:** Sorgulanacak olan operand adı yazılır.
2	**Alt Operand:** Sorgulamada baz alınacak sabit değer veya operandın adı yazılır.

Örnek uygulama :

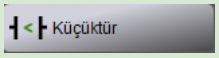


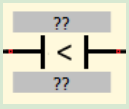
Üst operandın değeri, alt operandın değerinden büyük ise bir sonraki komutu çalıştırır. Aksi takdirde sonraki komut çalışmaz.

Küçüktür Komutu

Kısa adı : KÇ

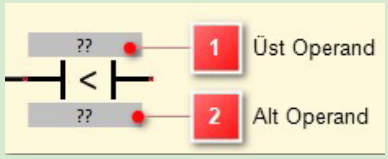
Kısa yol no. : Enter > 19

İkonu : 

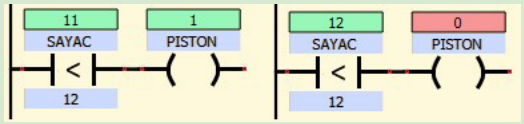


Operand tipleri : Word, double word, integer, real.

Çalışması :



1	**Üst Operand:** Sorgulanacak olan operand adı yazılır.
2	**Alt Operand:** Sorgulamada baz alınacak sabit değer veya operandın adı yazılır.

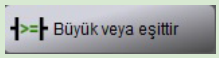


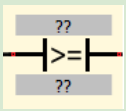
Örnek uygulama : Üst operandın değeri, alt operandın değerinden küçük ise bir sonraki komutu çalıştırır. Aksi takdirde sonraki komut çalışmaz.

Büyük veye Eşit ise Komutu

Kısa adı : BE

Kısa yol no. : Enter > 20

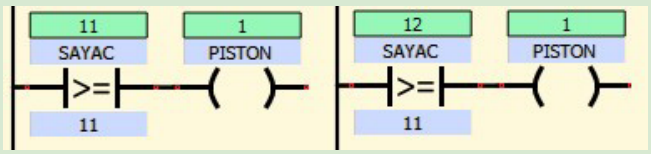
İkonu : 



Operand tipleri : Word, double word, integer, real.

Çalışması :

1	**Üst Operand:** Sorgulanacak olan operand adı yazılır.
2	**Alt Operand:** Sorgulamada baz alınacak sabit değer veya operandın adı yazılır.

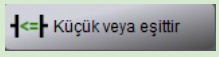


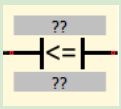
Örnek uygulama : Üst operandın değeri, alt operandın değerinden büyük veya eşit ise bir sonraki komutu çalıştırır. Aksi takdirde sonraki komut çalışmaz.

Küçük veye Eşit ise Komutu

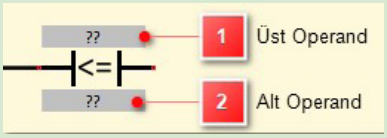
Kısa adı : KE

Kısa yol no. : Enter > 21

İkonu : 

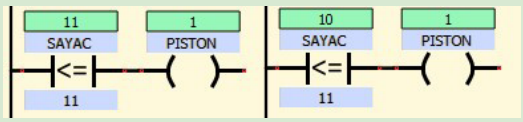


Operand tipleri : Word, double word, integer, real.



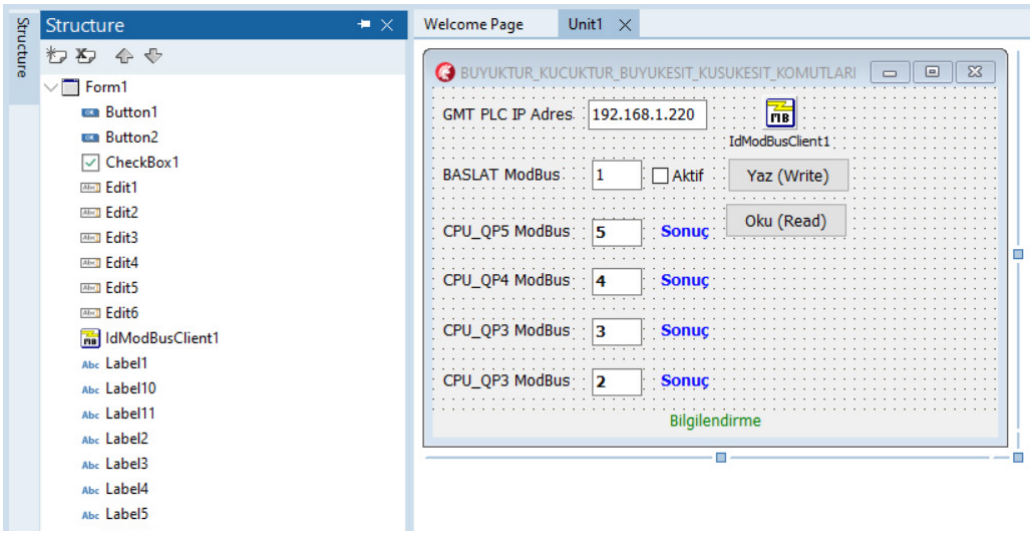
Çalışması :

1	**Üst Operand:** Sorgulanacak olan operand adı yazılır.
2	**Alt Operand:** Sorgulamada baz alınacak sabit değer veya operandın adı yazılır.



Örnek uygulama : Üst operandın değeri, alt operandın değerinden küçük ya da eşitse bir sonraki komutu çalıştırır. Aksi takdirde sonraki komut çalışmaz.

1.22. UYGULAMA: Büyüktür, Küçüktür, Büyük Eşit ve Küçük Eşit Komutları



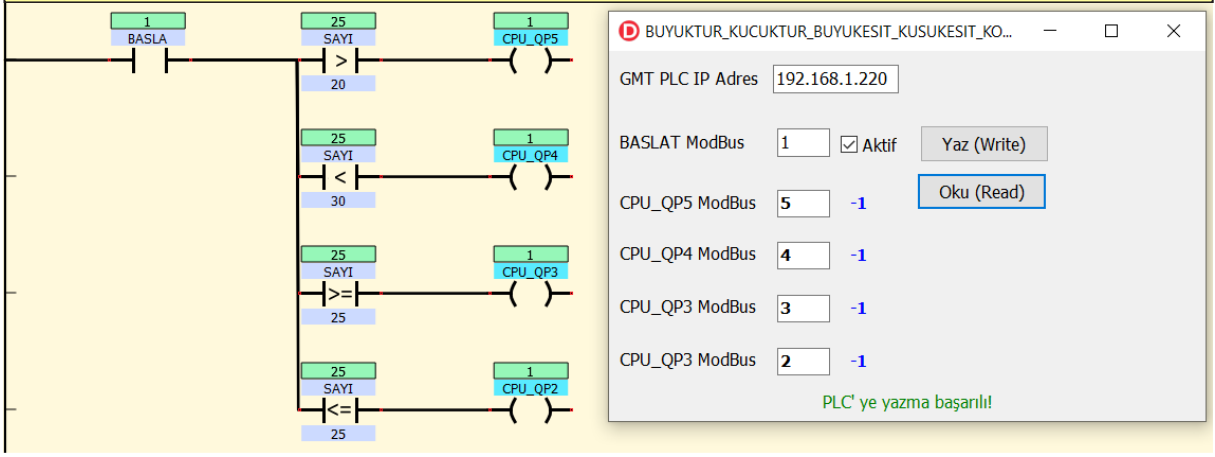
Görsel 1.80: Büyüktür, küçüktür, büyük eşit ve küçük eşit komutları Delphi arayüzü

```
procedure TForm1.Button2Click(Sender: TObject);
var
  SONUC: Boolean;
begin
  IdModBusClient1.Host := Edit1.Text;

  if IdModBusClient1.ReadCoil(StrToInt(Edit3.Text), SONUC) then
     Label8.Caption := BoolToStr(SONUC)
  else
    Label5.Caption := 'PLC''den okuma başarısız!';
  if IdModBusClient1.ReadCoil(StrToInt(Edit4.Text), SONUC) then
     Label9.Caption := BoolToStr(SONUC)
  else
    Label5.Caption := 'PLC''den okuma başarısız!';

  if IdModBusClient1.ReadCoil(StrToInt(Edit5.Text), SONUC) then
     Label10.Caption := BoolToStr(SONUC)
  else
    Label5.Caption := 'PLC''den okuma başarısız!';
  if IdModBusClient1.ReadCoil(StrToInt(Edit6.Text), SONUC) then
     Label11.Caption := BoolToStr(SONUC)
  else
    Label5.Caption := 'PLC''den okuma başarısız!';
end;
procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
  if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

Görsel 1.81: Büyüktür, küçüktür, büyük eşit ve küçük eşit komutları program kodu

Büyüktür, küçüktür, büyük veya eşittir, küçük veya eşittir işlemleri yapılır. Şart sağlandığında çıkışta aktif olur.

ModBus adres bilgileri: BASLA=1 CPU_QP3=5 CPU_QQP4=4 CPU_QQP3=3 CPU_QQP2=2



Görsel 1.82: Büyüktür, küçüktür, büyük eşit ve küçük eşit komutları ladder diyagramı

Aralık İçi Komutu

Kısa adı : AI

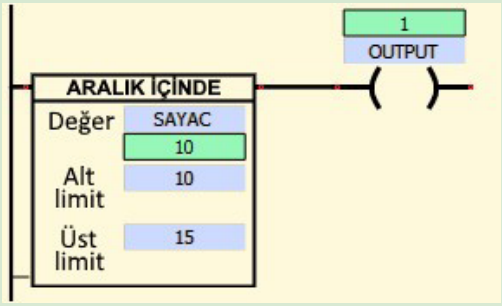
Kısa yol no. : Enter > 104

İkonu : 

Operand tipleri : Word, double word, integer, real.

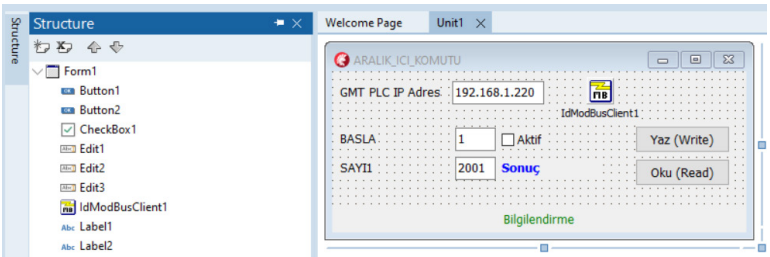
Çalışması :

1	**Değer:** Komutun sorguladığı operandın adı yazılır.
2	**Alt limit:** Karşılaştırma işlemi için alt limit değeri yazılır.
3	**Üst limit:** Karşılaştırma işlemi için üst limit değeri yazılır.



Örnek uygulama : **Değer,** işlenen değerin **Alt limit** ve **Üst limit** arasında ise veya bunlara eşitse, sonraki komut yürütülür. Aksi takdirde sonraki komut çalışmaz.

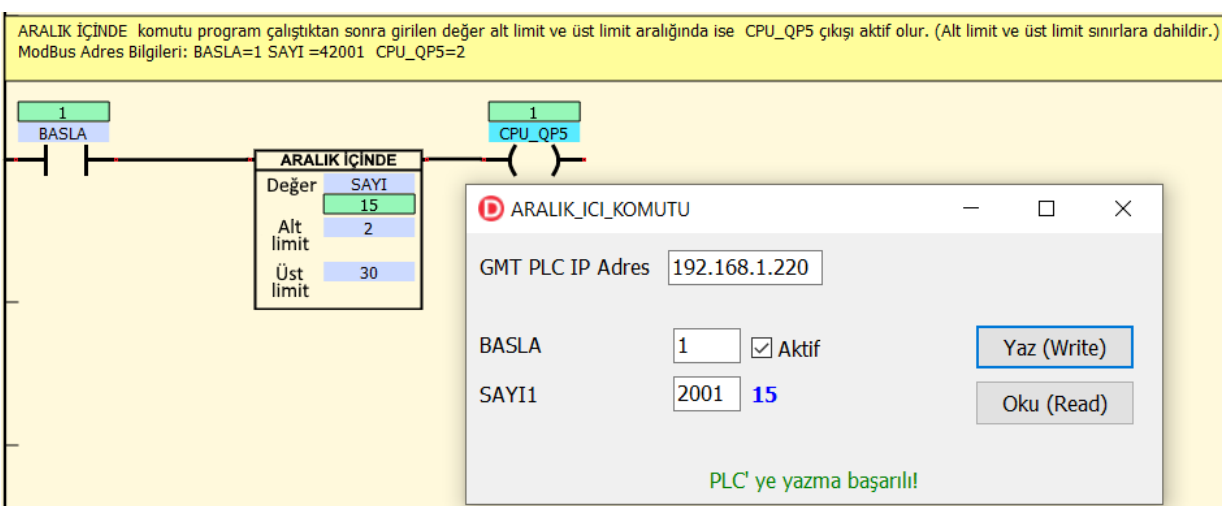
1.23. UYGULAMA: Aralık İçi Komutu



Görsel 1.83: Aralık içi komutu Delphi arayüzü

```
procedure TForm1.Button2Click(Sender: TObject);
 var
   Data: array[0..10] of Word;
   okuma_sayisi,i: Integer;
 begin
   okuma_sayisi :=2;
   if (okuma_sayisi > 0) then
   begin
     IdModBusClient1.Host := edit1.Text;
      if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit3.Text), okuma_sayisi, Data) then
     begin
       for i := 0 to (okuma_sayisi - 1) do
       begin
       label4.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
       label4.Caption:=FloatToStr(HexToIEEE754(label4.Caption));
       end;
     end
     else
       ShowMessage('PLC''den okuma başarısız!');
   end;
 end;
procedure TForm1.Button1Click(Sender: TObject);
 begin
   IdModBusClient1.Host := Edit1.Text;
     if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
     Label5.Caption := 'PLC'' ye yazma başarılı!'
   else
     Label5.Caption := 'PLC''ye yazma başarısız!';
 end;
```

Görsel 1.84: Aralık içi komutu program kodu



Görsel 1.85: Aralık içi komutu ladder diyagramı

1.2.3. Zaman Röleleri

Bu bölümde açıklanan komutlar, GLC ve GSR serisinin PLC modellerinde kullanılır. Zaman röleleri otomasyon sistemlerinde yaygın olarak kullanılır. GMTSuite programında dört farklı zaman rölesi kullanılır.

- Çekmede Gecikme (Düz) Zaman Rölesi
- Bırakmada Gecikme (Ters) Zaman Rölesi
- Puls Zaman Rölesi
- PWM Zaman Rölesi (Darbe Genişlik Modülasyonu)

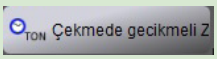
Bu zaman röleleri beş farklı zaman biriminde (milisaniye, saniye, dakika, saat ve gün olarak) çalışabilir.

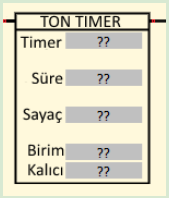
Çekmede Gecikme

Kısa adı : ÇG

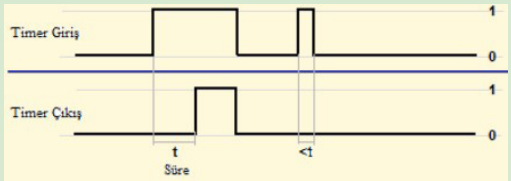
Kısa yol no. : Enter > 22

İkonu :

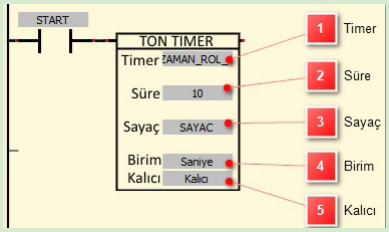




Operand tipleri : Word, double word, integer, real.



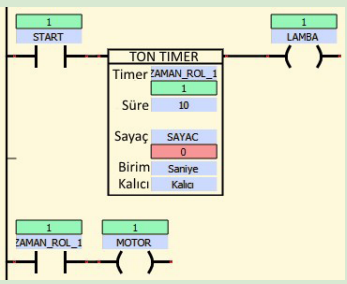
Çalışması : TON Timer, aktif olmasından itibaren ayarlanan **Süre** değerinden sonra çıkışı **1** yapar. Bu süre tamamlanmadan timer girişi pasif olursa çıkışı aktif olmaz.



Açıklama :

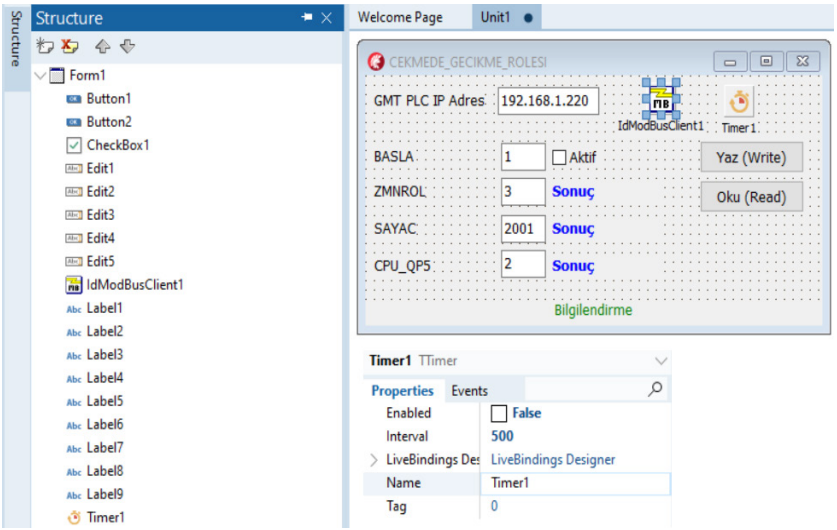
1	**Timer:** TON timer için tanımlanan isim yazılır. Timerin kontrol edildiği kontaklar bu isimle tanımlanır.
2	**Süre:** Timer süresi, sabit değer ya da operand adı olarak yazılır.
3	**Sayaç:** Zaman sayımı için belirlenen operandın adı yazılır. Bu değer sayım periyodu içinde kullanılır.
4	**Birim:** Milisaniye, saniye, dakika, saat veya gün seçeneklerinden biri seçilir.
5	**Kalıcı:** Kalıcı ve Resetli olmak üzere iki seçenek vardır. **Kalıcı** olarak belirlendiğinde, timer pasif olduğunda **Sayaç** içindeki değer silinmez. **Resetli** olarak seçilmiş ise timer pasif olduğunda **Sayaç** değeri silinir. TON timer tekrar aktif olduğunda **Sayaç** 0'dan saymaya başlar.

Örnek uygulama :



MOTOR ve **LAMBA**, TON timerın enerjilenmesinden 10 sn. sonra aktif olur. TON timer pasif ise bu çıkışlar da pasif olur.

1.24. UYGULAMA: Çekmede Gecikme Rölesi



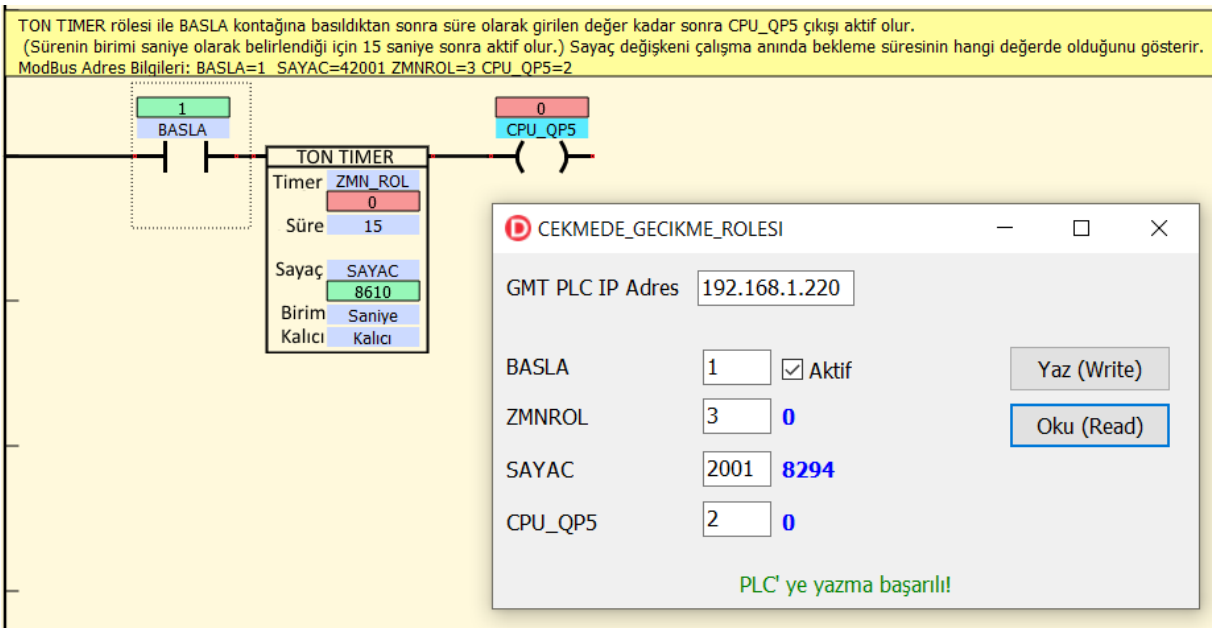
Görsel 1.86: Çekmede gecikme rölesi Delphi arayüzü

Uygulamada **timer1** interval değeri 500 mS olarak ayarlanır. 500 mS'de bir modbus değeri okunur. GMT süite uygulama ekranı ile program ekranı arasında sayısal değer farkı olur. Bunun sebebi PLC'de işlemlerin 500 mS'den daha hızlı olmasıdır.

```
procedure TForm1.Button2Click(Sender: TObject);
begin
  Timer1.Enabled:=TRUE;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
procedure TForm1.Timer1Timer(Sender: TObject);
 var
   Data: array[0..10] of Word;
   SONUC: Boolean; okuma_sayisi,i: Integer;
 begin
    okuma_sayisi :=2;
    IdModBusClient1.Host := edit1.Text;
     //SAYAC değeri alınır
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit4.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label7.Caption:=inttostr(Data[0]);
      end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');
    // ZAMAN ROLESI bilgisi alınır.
```

```
    if IdModBusClient1.ReadCoil(StrToInt(Edit3.Text), SONUC) then
     Label4.Caption := BoolToStr(SONUC)
    else
     Label5.Caption := 'PLC''den okuma başarısız!';
     //CPU_QP5 bilgisi alınır.
     if IdModBusClient1.ReadCoil(StrToInt(Edit5.Text), SONUC) then
     Label9.Caption := BoolToStr(SONUC)
    else
     Label5.Caption := 'PLC''den okuma başarısız!';
end;
```

Görsel 1.87: Çekmede gecikme rölesi program kodu



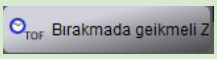
Görsel 1.88: Çekmede gecikme rölesi ladder diyagramı

Bırakmada Gecikme

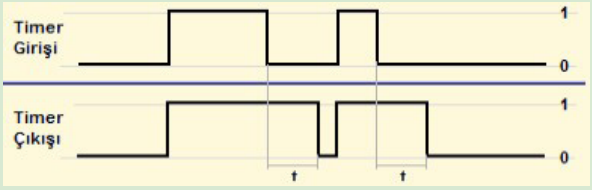
Kısa adı : BG

Kısa yol no. : Enter > 23

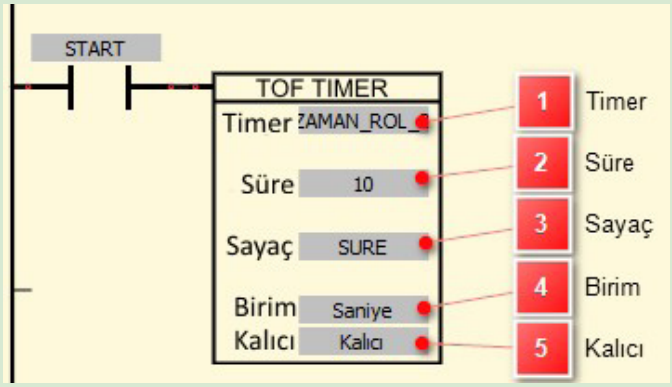
İkonu :



Operand tipleri : Word, double word, integer, real.

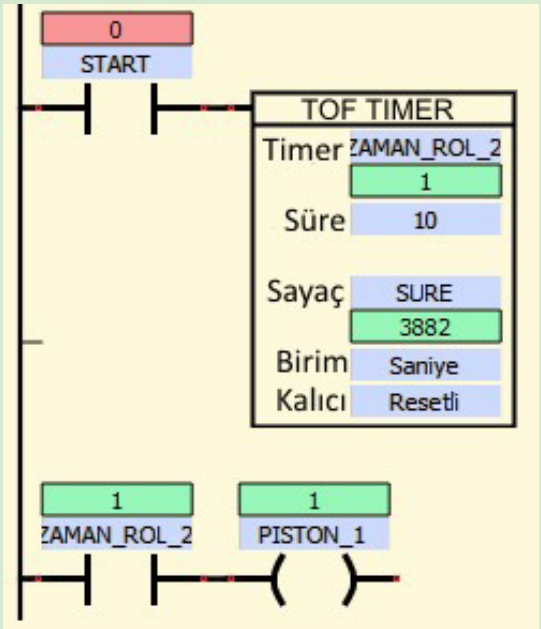


Çalışması : TOF timer, pasif olmasından itibaren ayarlanan **Süre** değerinden sonra çıkışı **0** yapar.



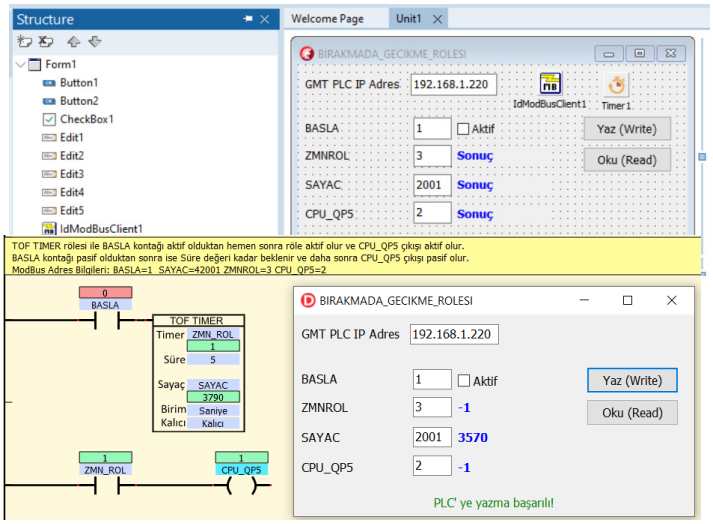
Açıklama :

1	**Timer:** TOF timer için tanımlanan isim yazılır. Timer ile kontrol edilen kontaklar bu adla tanımlanır.
2	**Süre:** Timer süresi, sabit ya da operand adı olarak yazılır.
3	**Sayaç**: Zaman sayımı için belirlenen operandın adı yazılır. Bu değer sayım periyodu için de kullanılır.
4	**Birim:** Milisaniye, saniye, dakika, saat veya gün seçeneklerinden biri seçilir.
5	**Kalıcı: Kalıcı** ve **Resetli** olmak üzere iki seçenek vardır. **Kalıcı** olarak belirlenmişse timer pasif olduğunda **Sayaç** içindeki değer silinmez. **Resetli** olarak seçilmiş ise timer pasif olduğunda **Sayaç** değeri silinir. TOF timer tekrar aktif olduğunda saymaya **0**'dan başlar.



Örnek uygulama : TOF timer rölesinin aktif edilmesiyle aynı isimle tanımlanmış olan kontak pozisyon değişir ve **PISTON_1** aktif olur. TOF timerın pasif edilmesinden itibaren 10 sn. daha **PISTON_1** aktif ve sonrasında pasif olur.

1.25. UYGULAMA: Bırakmada Gecikme Rölesi



Görsel 1.89: Bırakmada gecikme rölesi Delphi arayüzü ve ladder diyagramı

```
procedure TForm1.Button2Click(Sender: TObject);
begin
  Timer1.Enabled:=TRUE;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
procedure TForm1.Timer1Timer(Sender: TObject);
var
  Data: array[0..10] of Word;
  SONUC: Boolean; okuma_sayisi,i: Integer;
begin
    okuma_sayisi :=2;
    IdModBusClient1.Host := edit1.Text;
    if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit4.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label7.Caption:=inttostr(Data[0]);
      end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');
    // ZAMAN ROLESI bilgisi alınır.
   if IdModBusClient1.ReadCoil(StrToInt(Edit3.Text), SONUC) then
    Label4.Caption := BoolToStr(SONUC)
   else
    Label5.Caption := 'PLC''den okuma başarısız!';
    //CPU_QP5 bilgisi alınır.
    if IdModBusClient1.ReadCoil(StrToInt(Edit5.Text), SONUC) then
    Label9.Caption := BoolToStr(SONUC)
   else
    Label5.Caption := 'PLC''den okuma başarısız!';
end;
```

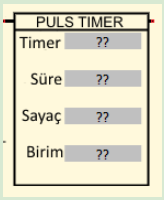
Görsel 1.90: Bırakmada gecikme rölesi program kodu

Puls Zaman Rölesi

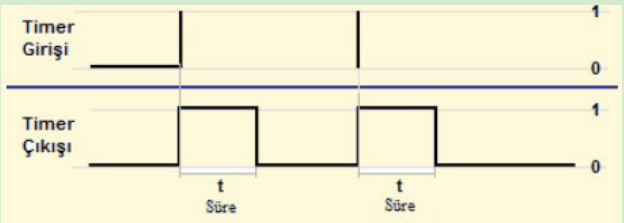
Kısa adı : PZ

Kısa yol no. : Enter > 24

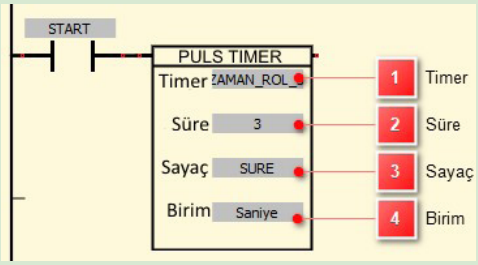
İkonu : Puls ZR



Operand tipleri : Word, double word, integer, real.

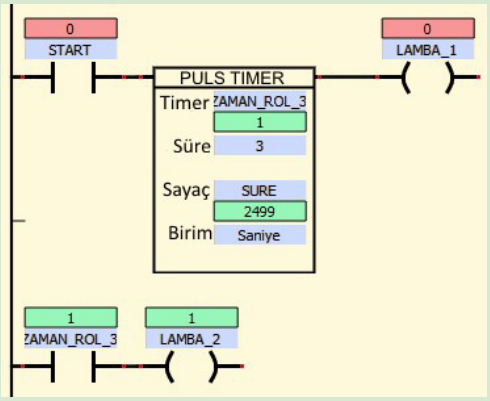


Çalışması : Pals zaman rölesi, aktif olduktan sonra önceden ayarlanan bir genişlik süresiyle darbe üretir.



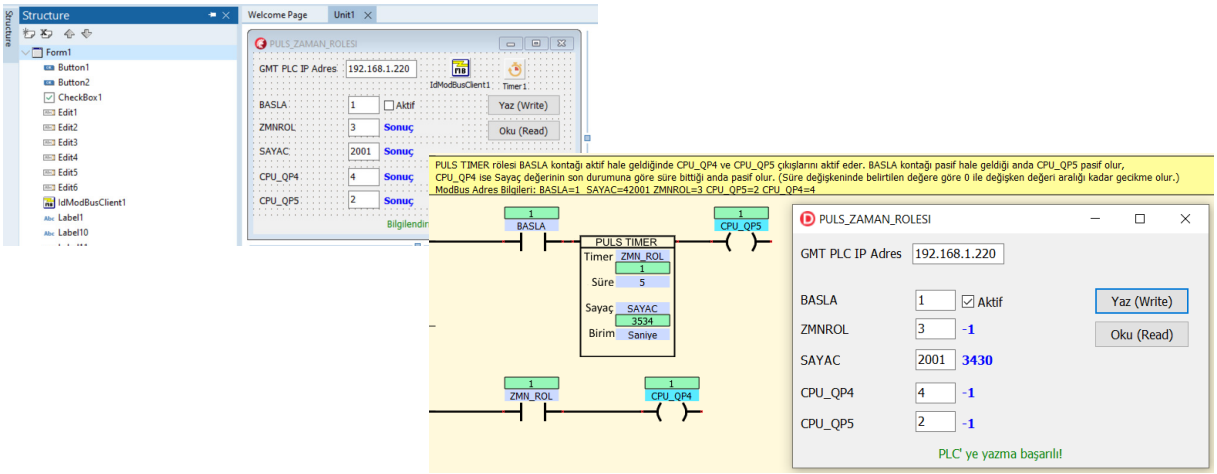
Açıklama :

1	**Timer:** TPL timer için tanımlanan isim yazılır. Timerin kontrol ettiği kontaklar bu isimle tanımlanır.
2	**Süre:** Timer süresi, sabit değer ya da operand adı olarak yazılır.
3	**Sayaç:** Zaman sayımı için belirlenen operandın adı yazılır. Bu değer sayım periyodu içinde kullanılır.
4	**Birim:** Milisaniye, saniye, dakika, saat veya gün seçeneklerinden biri seçilir.



Örnek uygulama : TPL timer rölesinin aktif edilmesiyle **LAMBA_1** ile **ZAMAN_ROL_3** pozisyon değiştirir ve **LAMBA_2** aktif olur. TPL timerın pasif edilmesinden itibaren 3 sn. daha **LAMBA_2** aktif ve sonra pasif olur. Ancak **LAMBA_1** ile TPL timerın durumu aynıdır. Aynı anda aktif/pasif olur.

1.26. UYGULAMA: PULS Zaman Rölesi



Görsel 1.91: Puls zaman rölesi Delphi arayüzü ve ladder diyagramı

```
procedure TForm1.Button2Click(Sender: TObject);
begin
  Timer1.Enabled:=TRUE;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;

procedure TForm1.Timer1Timer(Sender: TObject);
var
  Data: array[0..10] of Word;
  SONUC: Boolean; okuma_sayisi,i: Integer;
begin
    okuma_sayisi :=2;
    IdModBusClient1.Host := edit1.Text;
     //SAYAC değeri alınır
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit4.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label7.Caption:=inttostr(Data[0]);
      end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');
    // ZAMAN ROLESI bilgisi alınır.
   if IdModBusClient1.ReadCoil(StrToInt(Edit3.Text), SONUC) then
    Label4.Caption := BoolToStr(SONUC)
   else
    Label5.Caption := 'PLC''den okuma başarısız!';
    //CPU_QP4 bilgisi alınır.
    if IdModBusClient1.ReadCoil(StrToInt(Edit5.Text), SONUC) then
    Label9.Caption := BoolToStr(SONUC)
   else
    Label5.Caption := 'PLC''den okuma başarısız!';
        //CPU_QP5 bilgisi alınır.
    if IdModBusClient1.ReadCoil(StrToInt(Edit6.Text), SONUC) then
    Label11.Caption := BoolToStr(SONUC)
   else
    Label5.Caption := 'PLC''den okuma başarısız!';
end;
```

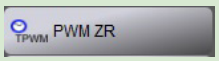
Görsel 1.92: Puls zaman rölesi program kodu

PWM Zaman Rölesi

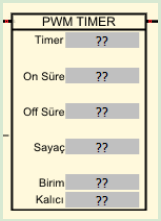
Kısa adı : PW

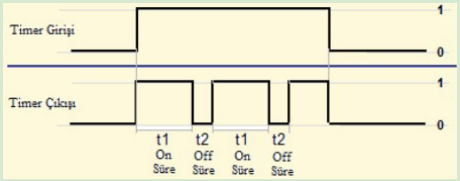
Kısa yol no. : Enter > 36

İkonu :

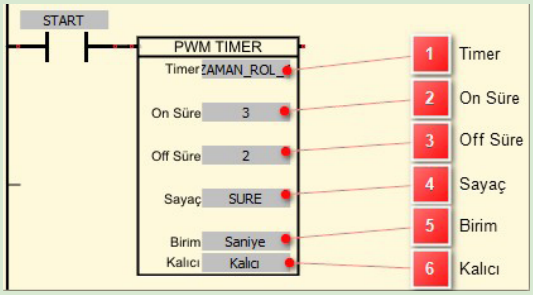


Operand tipleri : Word, double word, integer, real.





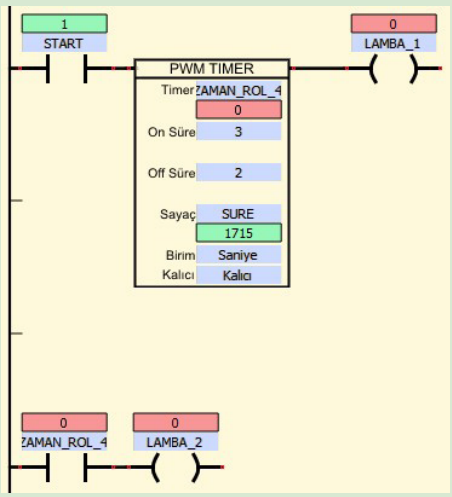
Çalışması : PWM zaman rölesi, aktif olduğu sürece $t_1 + t_2$ periyodunda darbeler üretir.



Açıklama :

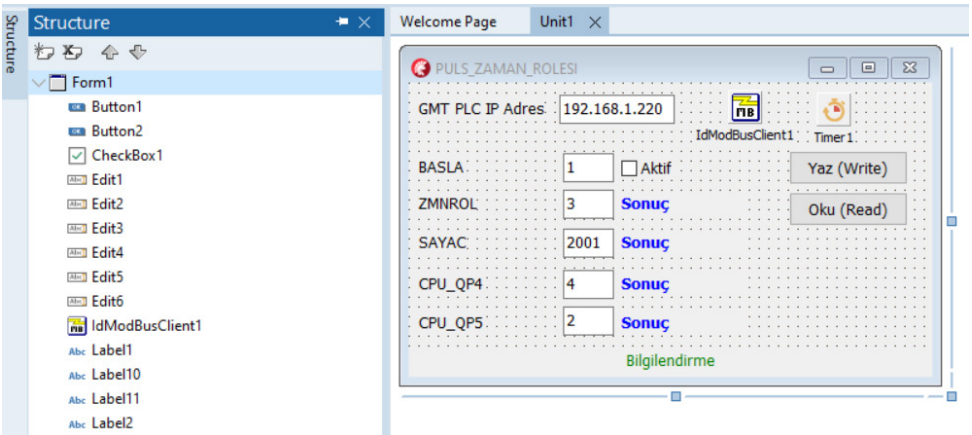
1	**Timer:** PWM timer için tanımlanan isim yazılır. Timerın kontrol ettiği kontaklar bu isimle tanımlanır.
2	**On Süre:** On süresi sabit ya da operand adı olarak yazılır.
3	**Off Süre:** Off süresi sabit değer ya da operand adı olarak yazılır.
4	**Sayaç:** Zaman sayımı için belirlenen operandın adı yazılır. Bu değer, sayım periyodu içinde kullanılır.
5	**Birim:** Milisaniye, saniye, dakika, saat veya gün seçeneklerinden biri seçilir.
6	**Kalıcı:** Kalıcı ve Resetli olmak üzere iki seçenek vardır. **Kalıcı** olarak belirlenen timer pasif olduğunda **Sayaç** içindeki değer silinmez. **Resetli** olarak seçilen timer pasif olduğunda **Sayaç** değeri silinmez, PWM timer tekrar aktif olduğunda **Sayaç** 0'dan saymaya başlar.

Örnek uygulama :



Timer ile aynı isimle tanımlanan kontak **ZAMAN_ROL_4**; timerın On süresince aktif, Off süresince pasif olur. Dolayısıyla bu kontağa bağlı olan **LAMBA_2**, PWM timerın kontağı ile aynı anda aktif ve pasif olur. PWM TIMER On süresi içinde iken timer pasif edilirse **LAMB_1** pasif, **LAMB_2** ise aktif kalır. Sayaçdaki süre durur. PWM TIMER tekrar aktif edildiğinde **Kalıcı** seçilmiş ise sayaç kaldığı süreden devam eder ve **Resetli** seçilen de sıfırlanarak devam eder.

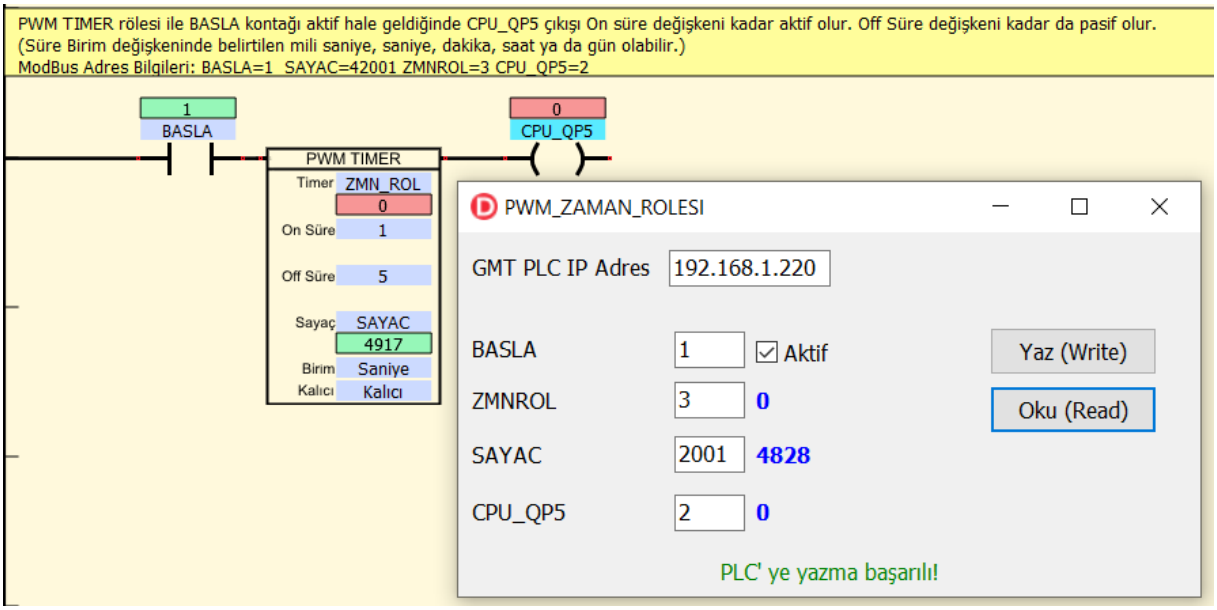
1.27. UYGULAMA: PWM Zaman Rölesi



Görsel 1.93: PWM zaman rölesi Delphi arayüzü

```
procedure TForm1.Button2Click(Sender: TObject);
begin
  Timer1.Enabled:=TRUE;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;

procedure TForm1.Timer1Timer(Sender: TObject);
 var
   Data: array[0..10] of Word;
   SONUC: Boolean; okuma_sayisi,i: Integer;
 begin
    okuma_sayisi :=2;
    IdModBusClient1.Host := edit1.Text;
     //SAYAC değeri alınır
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit4.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label7.Caption:=inttostr(Data[0]);
      end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');
    // ZAMAN ROLESI bilgisi alınır.
   if IdModBusClient1.ReadCoil(StrToInt(Edit3.Text), SONUC) then
    Label4.Caption := BoolToStr(SONUC)
   else
    Label5.Caption := 'PLC''den okuma başarısız!';
    //CPU_QP5 bilgisi alınır.
    if IdModBusClient1.ReadCoil(StrToInt(Edit5.Text), SONUC) then
    Label9.Caption := BoolToStr(SONUC)
   else
    Label5.Caption := 'PLC''den okuma başarısız!';
 end;
```

Görsel 1.94: PWM zaman rölesi program kodu



Görsel 1.95: PWM zaman rölesi ladder diyagramı

1.2.4. Sayaçlar

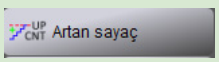
Bu bölümde açıklanan komutlar, GLC serisinin PLC modellerinde kullanılır. GSR serisi PLC modellerinde sadece Artan Sayaç ve Azalan Sayaç komutları kullanılır. Sayaçlar otomasyon sitemlerinde yaygın olarak kullanılır. GMTSuite programında artan veya azalan sayaçlar kullanılır. Bir sayaca ait 32 bit veri alanı ayrılır. Dolayısıyla sayaçlar -2.147.483.648 ile 2.147.483.647 sayılar arasında sayma işlemi yapabilir. Sayaçlar art arda bağlanabilir. Bu şekilde birbiriyle ilişkili sayıcılar elde edilir. Ayrıca iki adet HSC [High Speed Counter (hay spid kantır) (Yüksek Hızlı Sayıcı)] komutu bu menü altında bulunur. HSC komutları kullanılmadan önce **konfigürasyon** menüsü altından ilgili kanala ait HSC aktif edilir. Hızlı sayıcılar, hızlı olan giriş sinyallerinin (enkoder çıkışı vb.) sayılması için kullanılır. GLC serisi PLC'lerde **Kanal 0**, **Kanal 1** ve **Kanal 2** olmak üzere üç adet hızlı sayıcı aynı anda kullanılır.

Artan Sayaç

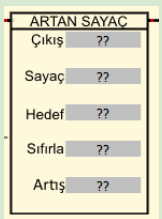
Kısa adı : AS

Kısa yol no. : Enter > 25

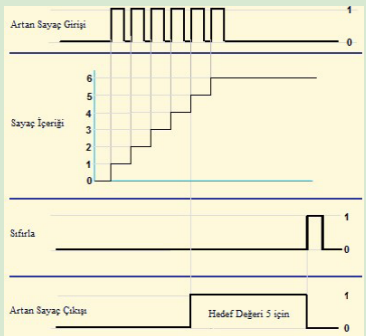
İkonu :



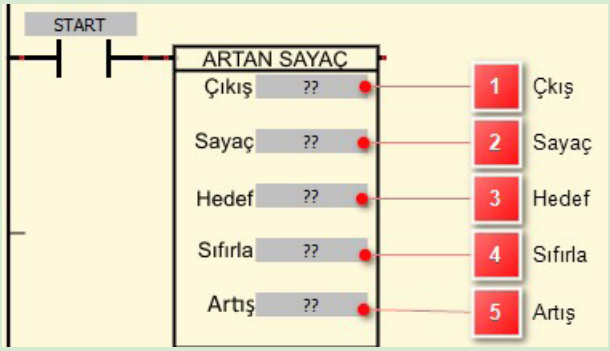
Operand tipleri : Word, double word, integer, real.



Çalışması :

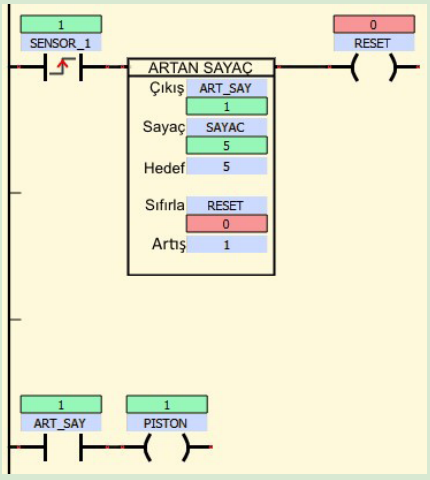


Artan Sayaç girişi 0'dan 1'e değiştiğinde **Sayaç** içeriği **Artış** miktarı kadar artar. Sayaç, hedef değere ulaştığında çıkış aktif olur.



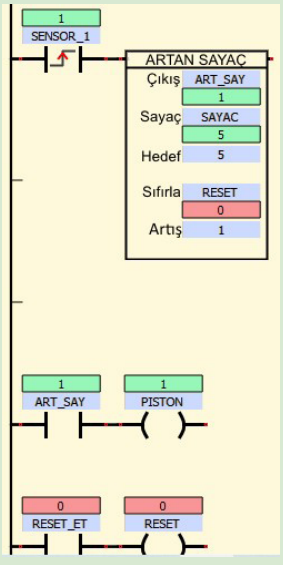
Açıklama :

1	**Çıkış:** Artan Sayaç için tanımlanan ad yazılır. Sayacın kontrol ettiği kontaklar bu isimle tanımlanır.
2	**Sayaç:** Sayım için belirlenen operandın adı yazılır. Bu değer, sayım periyodu için de kullanılabilir.
3	**Hedef:** Sayacın hedef değeri sabit değer ya da operand adı olarak yazılır. Sayacın çıkışı, sayaç istenilen değere ulaştığında aktif olur.
4	**Sıfırla:** Sayacı sıfırlayan operandın adı yazılır.
5	**Artış:** Artış miktarı sabit ya da operand adı olarak yazılır.



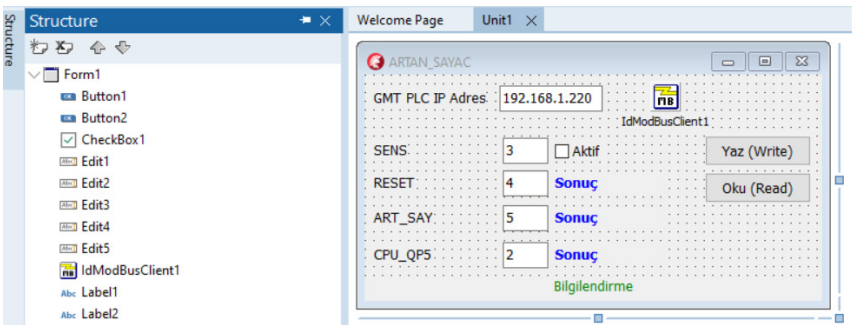
Sayacın hedefe ulaştığında kendini sıfırlaması istenirse sayacın çıkışına sıfırla alanında belirtilen operand ile aynı ada sahip **RESET** adlı bir **Direkt Çıkış** rölesi bağlanır.

Örnek uygulama :



Artan Sayaç'ın her 0'dan 1'e geçişinde **Sayac** birer artar. **Sayaç** 5'e eşit olduğunda Artan Sayaç çıkışı aktif olur. Sayaç **RESET** isimli Direkt Çıkış rölesini aktif hâle getirerek sıfırlanabilir. Güç kesildiğinde sayaç değerinin sıfırlanmaması istenirse **SAYAC** operandı kalıcı olarak tanımlanır.

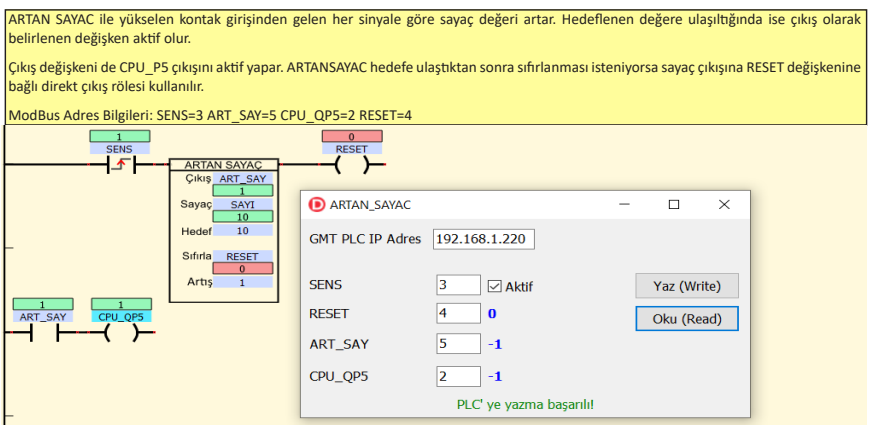
1.28. UYGULAMA: Artan Sayaç



Görsel 1.96: Artan sayaç Delphi arayüzü

```
procedure TForm1.Button2Click(Sender: TObject);
 var
   Data: array[0..10] of Word;
   SONUC: Boolean;
 begin
     IdModBusClient1.Host := edit1.Text;
     // RESET bilgisi alınır.
    if IdModBusClient1.ReadCoil(StrToInt(Edit3.Text), SONUC) then
     Label4.Caption := BoolToStr(SONUC)
    else
     Label5.Caption := 'PLC''den okuma başarısız!';
     // ART_SAY bilgisi alınır.
    if IdModBusClient1.ReadCoil(StrToInt(Edit4.Text), SONUC) then
     Label7.Caption := BoolToStr(SONUC)
    else
     Label5.Caption := 'PLC''den okuma başarısız!';
     //CPU_QP5 bilgisi alınır.
     if IdModBusClient1.ReadCoil(StrToInt(Edit5.Text), SONUC) then
     Label9.Caption := BoolToStr(SONUC)
    else
     Label5.Caption := 'PLC''den okuma başarısız!';
 end;
 procedure TForm1.Button1Click(Sender: TObject);
 begin
   IdModBusClient1.Host := Edit1.Text;
     if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
     Label5.Caption := 'PLC'' ye yazma başarılı!'
   else
     Label5.Caption := 'PLC''ye yazma başarısız!';
 end;
```

Görsel 1.97: Artan sayaç program kodu

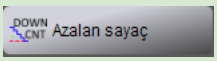


Görsel 1.98: Artan sayaç ladder diyagramı

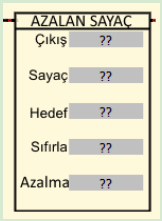
Azalan Sayaç

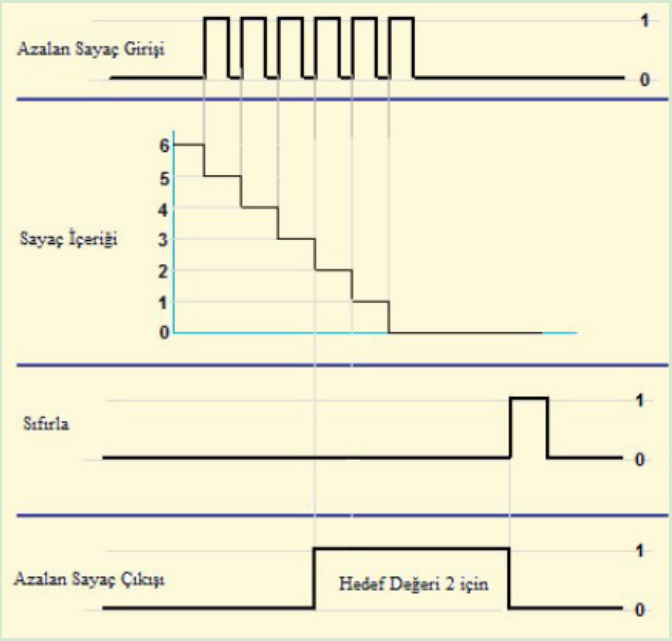
Kısa adı : ZS

Kısa yol no. : Enter > 26

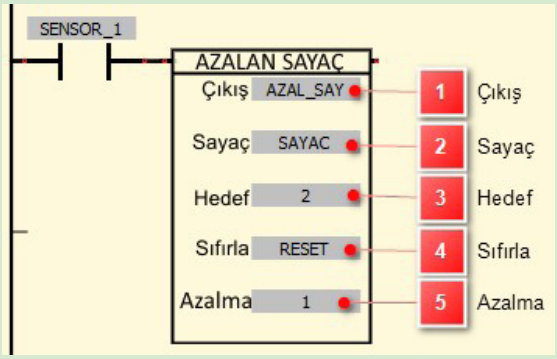
İkonu : 

Operand tipleri : Word, double word, integer, real.





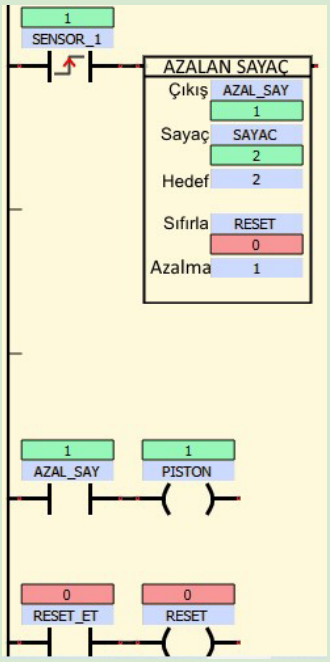
Açıklama : Azalan Sayaç girişi 0'dan 1'e değiştiğinde **Sayaç** içeriği **Azalma** miktarı kadar azalır. Sayaç, hedef değere ulaştığında çıkış aktif olur.



Çalışması :

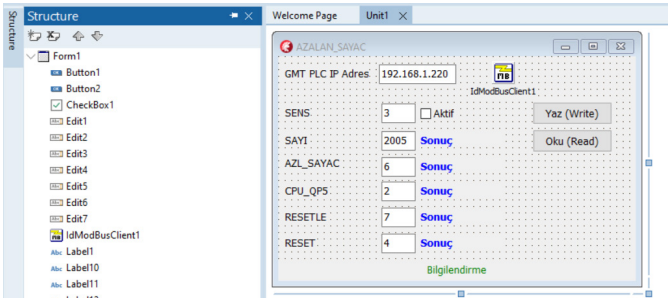
1	**Çıkış:** Azalan Sayaç için tanımlanan ad yazılır. Sayacın kontrol ettiği kontaklar bu adla tanımlanır.
2	**Sayaç:** Sayım için belirlenen operandın adı yazılır. Bu değer, sayım periyodu için de kullanılır.
3	**Hedef:** Sayacın hedef değeri, sabit değer ya da operand adı olarak yazılır. Sayaç bu değere ulaştığında sayacın çıkışı aktif olur.
4	**Sıfırla:** Sayacı sıfırlayacak olan operandın adı yazılır.
5	**Azalma:** Azalma miktarı sabit ya da operand adı olarak yazılır.

Örnek uygulama :



Azalan Sayaç'ın her 0'dan 1'e geçişinde **Sayac** birer azalır. Sayaç 2'ye eşit olduğunda Azalan Sayaç çıkışı etkinleşir. Sayaç **RESET** adlı **Direkt Çıkış** rölesini aktif hâle getirerek sıfırlanır. Güç kesildiğinde sayaç değerinin sıfırlanmaması istenirse **SAYAC** operandı kalıcı olarak tanımlanır.

1.29. UYGULAMA: Azalan Sayaç

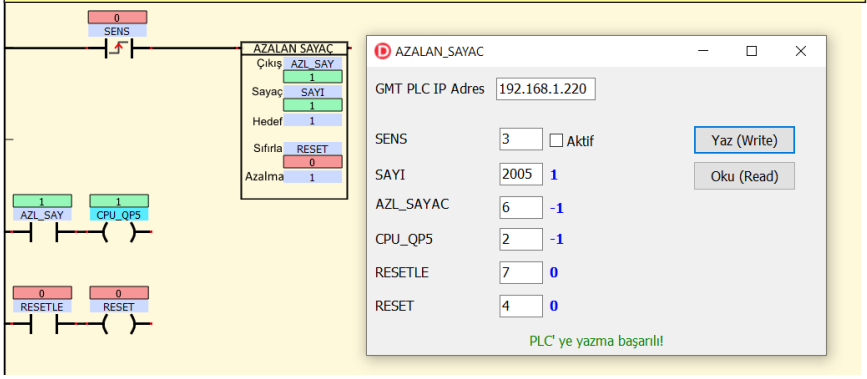


Görsel 1.99: Azalan sayaç Delphi arayüzü

AZALAN_SAYAC ile yükselen kontak girişinden gelen her sinyale göre sayaç değeri azalır. Hedeflenen değere ulaşıltığında ise çıkış olarak belirlenen değişken aktif olur.

Çıkış değişkeni de CPU_QP5 çıkışını aktif yapar. AZALAN SAYAÇ hedefe ulaştıktan sonra sıfırlanması isteniyorsa sayaç çıkışına RESETLE isimli normalde açık kontak ile direkt çıkış rölesi resetlenir.

ModBus Adres Bilgileri: SENS=3 AZL_SAY=6 CPU_QP5=2 RESET=4 SAYI=42005 RESETLE=7



Görsel 1.100: Azalan sayaç ladder diyagramı

```
procedure TForm1.Button2Click(Sender: TObject);
var
  Data: array[0..10] of Word;
  SONUC: Boolean; okuma_sayisi,i: Integer;
begin
    okuma_sayisi :=2;
    IdModBusClient1.Host := edit1.Text;
     //SAYI değeri alınır
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit3.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label4.Caption:=inttohex(Data[1])+inttohex(Data[0]);
      label4.Caption:=FloatToStr(HexToIEEE754(label4.Caption));
      end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');
    // AZL_SAYAC bilgisi alınır.
   if IdModBusClient1.ReadCoil(StrToInt(Edit4.Text), SONUC) then
    Label7.Caption := BoolToStr(SONUC)
   else
    Label5.Caption := 'PLC''den okuma başarısız!';
    //CPU_QP5 bilgisi alınır.
    if IdModBusClient1.ReadCoil(StrToInt(Edit5.Text), SONUC) then
    Label9.Caption := BoolToStr(SONUC)
   else
    Label5.Caption := 'PLC''den okuma başarısız!';
    //RESETLE bilgisi alınır.
    if IdModBusClient1.ReadCoil(StrToInt(Edit6.Text), SONUC) then
    Label11.Caption := BoolToStr(SONUC)
   else
    Label5.Caption := 'PLC''den okuma başarısız!';
    //RESET bilgisi alınır.
    if IdModBusClient1.ReadCoil(StrToInt(Edit7.Text), SONUC) then
    Label13.Caption := BoolToStr(SONUC)
   else
    Label5.Caption := 'PLC''den okuma başarısız!';
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text; // GMT PLC nin bağlı olduğu IP adresi.
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

Görsel 1.101: Azalan sayaç program kodu

1.3. KAYDIRMA, DÖNDÜRME, TAŞIMA VE PROGRAM KONTROL FONKSİYONLARI

PLC'ler ile mikrodenetleyicilerde bit işlemleri yapılır. Bitlerin sola ve sağa kaydırma işlemleri ile motor ve lambaların kontrolleri önemlidir.

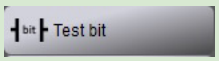
1.3.1. BIT Operasyon Komutları

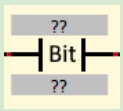
Bu bölümde açıklanan komutlar, GLC ve GSR serisinin PLC modellerinde kullanılır. Programlarda verilerin bitleri kontrol eden yedi adet bit operasyon komutu kullanılır. Tüm bit operasyonlarında operand (veri) tipi ya word ya da double worddur. Word tipi için veri değeri maksimum 65.535'tir. Double word için veri değeri maksimum 4.294.967.295'tir. Bu değerlerin dışına taşma olduğunda, program hata mesajı verir ve işlemler gerçekleştirilmez.

Test Bit Komutu

Kısa adı : TB

Kısa yol no. : Enter > 57

İkonu : 

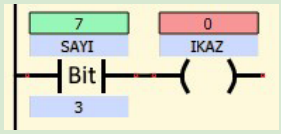


Operand tipleri : Word, double word.

Çalışması :

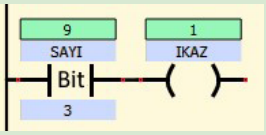
1	**Operand Adı:** Test edilecek operand adı yazılır.
2	**Test Ediecek Bit:** Test edilmesi istenen bit numarası yazılır (Sağdan 0 ile başlar. Word tipi operand için maksimum 15, double word için maksimum 31 değeri girilir.).

Örnek uygulama :



(Word tipi için; 7=0000 0000 0000 **0**111)

I. uygulamada **SAYI** adlı operandın 3. bitine bakılır. **SAYI** operandının değeri **7** ve bu sayının 3. biti **0**'dır. Bu sebeple **IKAZ** adlı direkt çıkış rölesi pasiftir.



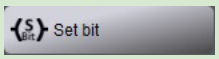
(Word tipi için; 9=0000 0000 0000 **1**001)

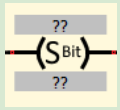
II. uygulamada **SAYI** adlı operandın değeri 9 olarak değiştirilir. **SAYI** operandının 3. bitinin **1** olması sebebiyle **IKAZ** adlı direkt çıkış rölesi aktiftir.

Set Bit Komutu

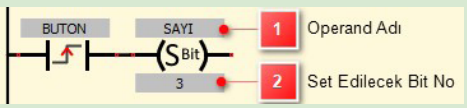
Kısa adı : SB

Kısa yol no. : Enter > 58

İkonu : 



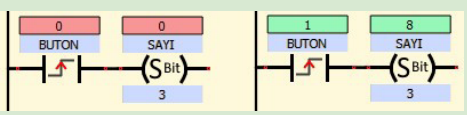
Operand tipleri : Word, double word.



Çalışması :

1	**Operand Adı:** Set edilecek operand adı yazılır.
2	**Set Ediecek Bit:** Set edilecek bit numarası yazılır (Sağdan 0 ile başlar. Word tipi operand için maksimum 15, double word için de maksimum 31 değeri girilir.).

Örnek uygulama :



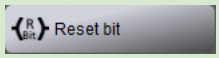
SAYI adlı operandın 3. biti **Set** edilir. Komut pasif iken **SAYI** operandının içeriği **0,** aktif olduğunda operand içeriği **8** olur.

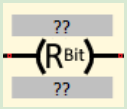
(0 = 0000 0000 0000 **0**000) (8 = 0000 0000 0000 **1**000)

Reset Bit Komutu

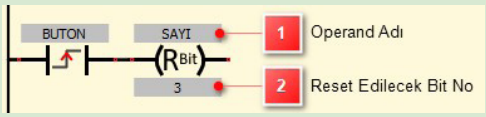
Kısa adı : RB

Kısa yol no. : Enter > 59

İkonu : 

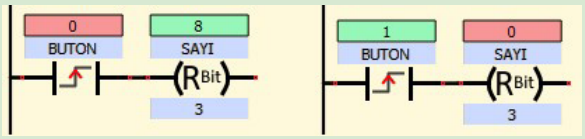


Operand tipleri : Word, double word.



Çalışması :

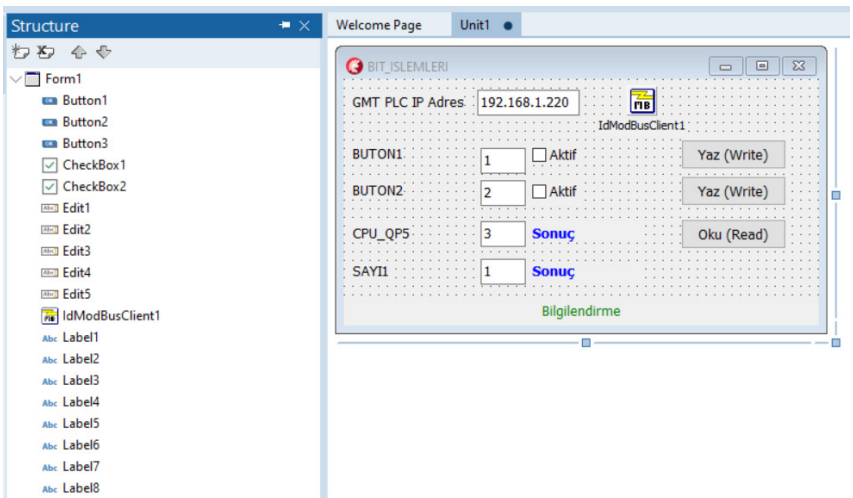
1	**Operand Adı:** Reset edilecek operand adı yazılır.
2	**Reset Ediecek Bit:** Sağdan itibaren kaçıncı bitin **0** yapılması istenirse bu bit numarası yazılır (Sağdan 0 ile başlar. Word tipi operand için maksimum 15, double word için de maksimum 31 değeri girilir.).



Örnek uygulama : (8 = 0000 0000 0000 **1**000) (0 = 0000 0000 0000 **0**000)

SAYI adlı operandın 3. biti **Reset** edilir. Komut pasif iken **SAYI** operandının içeriği **8**, aktif olduğunda operand içeriği **0** olur.

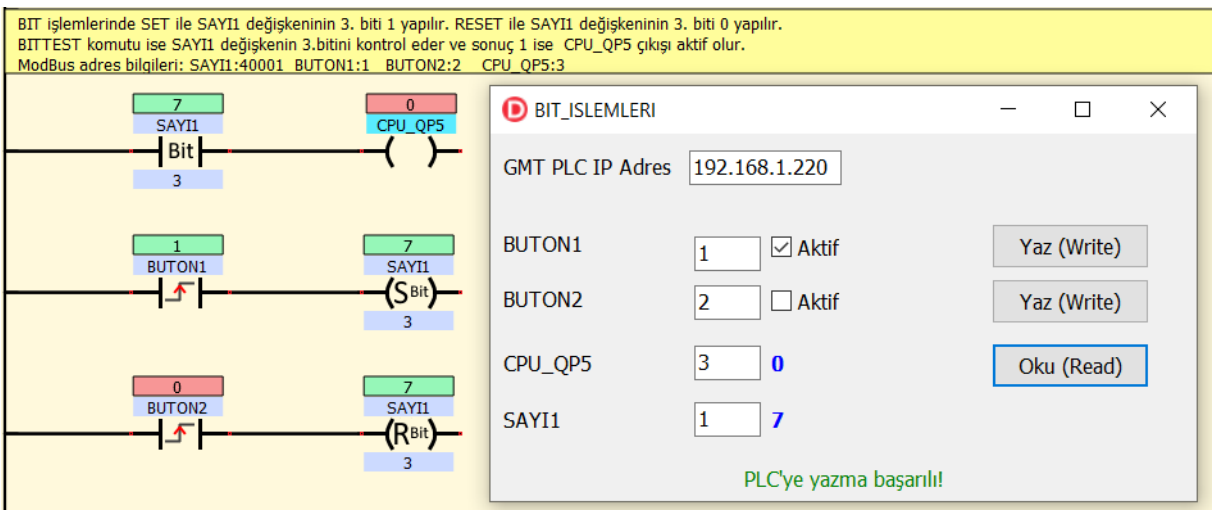
1.30. UYGULAMA: Bit İşlemleri



Görsel 1.102: Bit işlemleri Delphi arayüzü

```
procedure TForm1.Button2Click(Sender: TObject);
var
  Data: array[0..10] of Word;
  SONUC: Boolean; okuma_sayisi,i: Integer;
begin
    okuma_sayisi :=2;
    IdModBusClient1.Host := edit1.Text;
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit5.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label7.Caption:=inttostr(Data[0]);
      end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');
   if IdModBusClient1.ReadCoil(StrToInt(Edit4.Text), SONUC) then
    Label4.Caption := BoolToStr(SONUC)
   else
    Label5.Caption := 'PLC''den okuma başarısız!';
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC''ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;

procedure TForm1.Button3Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
  if IdModBusClient1.WriteCoil(StrToInt(Edit3.Text), CheckBox2.Checked) then
    Label5.Caption := 'PLC''ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

Görsel 1.103: Bit işlemleri program kodu

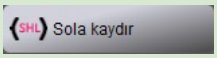


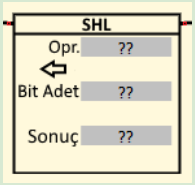
Görsel 1.104: Bit işlemleri ladder diyagramı

Sola Kaydır Komutu

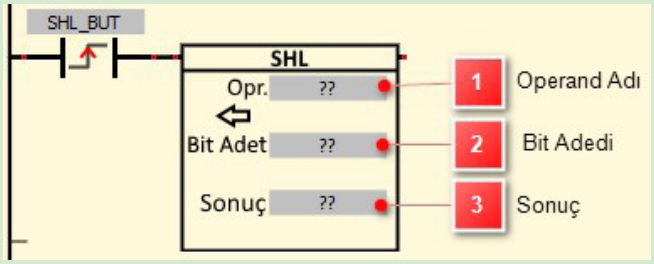
Kısa adı : OK

Kısa yol no. : Enter > 60

İkonu : 

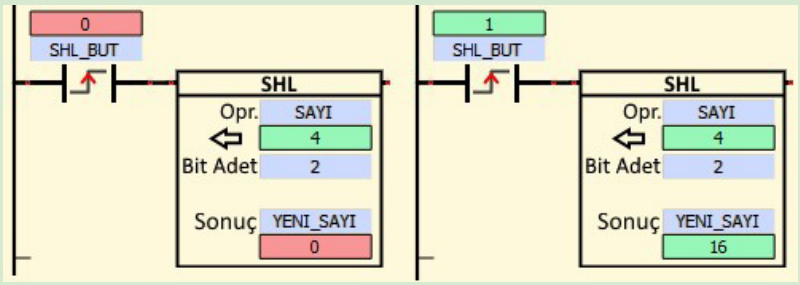


Operand tipleri : Word, double word.



Açıklama :

1	**Operand Adı:** Sola kaydırma işleminin uygulanacağı operandın adı yazılır. Word tipi için operand değeri maksimum 65.535, double word için ise operand değeri maksimum 4.294.967.295 olur.
2	**Bit Adedi:** Sola kaydırma işleminin kaç bit (kaç defa) yapılacağı yazılır. Word tipi operandlarda bit adedi 16 ve üstü değer yazılırsa sonuç **0** çıkar. Aynı şekilde double word tipi operandlarda bit adedi 32 ve üstü değer yazılırsa sonuç **0** çıkar.
3	**Sonuç:** Kaydırma işleminin sonunda yeni değerin kaydedileceği operandın adı yazılır. Operand tipi word ya da double word olabilir.



Örnek uygulama : Sola kaydır komutunun 0'dan 1'e değişmesiyle **SAYI** adlı operandın içeriği **Bit Adeti** kadar sola kayar.

4 = 0000 0000 0000 0100 → 16 = 0000 0000 0001 0000)

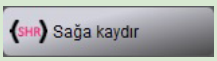
Sonuç kısmında **YENI_SAYI** adlı operandın değeri **8** olur. **Opr.** kısmında belirtilen **SAYI** adlı operandın değeri de 4 olarak kalır.

Bitler MSB'ye doğru kaydıkça LSB'ye **0** eklenir.

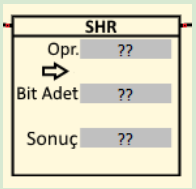
Sağa Kaydır Komutu

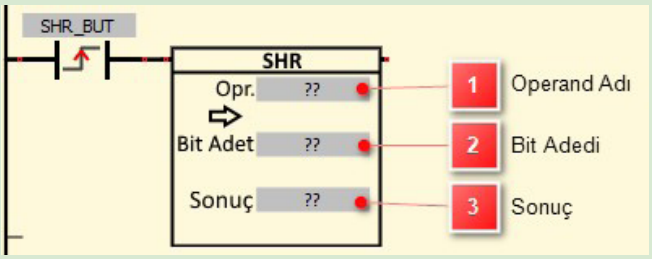
Kısa adı : AK

Kısa yol no. : Enter > 61

İkonu : 

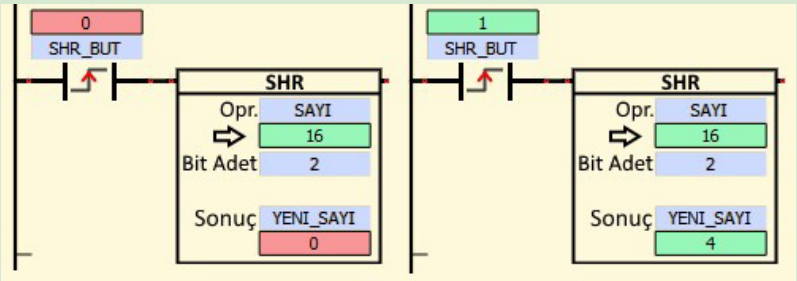
Operand tipleri : Word, double word.





Açıklama :

1	**Operand Adı:** Sağa kaydırma işleminin uygulanacağı operandın adı yazılır. Word tipi için operand değeri maksimum 65.535, double word için ise operand değeri maksimum 4.294.967.295 olur.
2	**Bit Adedi:** Sağa kaydırma işleminin kaç bit yapılacağı yazılır. Word tipi operandlarda bit adedi 16 ve üstü değer yazılırsa sonuç **0** çıkar. Aynı şekilde double word tipi operandlarda bit adedi 32 ve üstü değer yazılırsa sonuç **0** çıkar.
3	**Sonuç:** Kaydırma işleminin sonunda yeni değerin kaydedileceği operandın adı yazılır. Operand tipi word ya da double word olur.



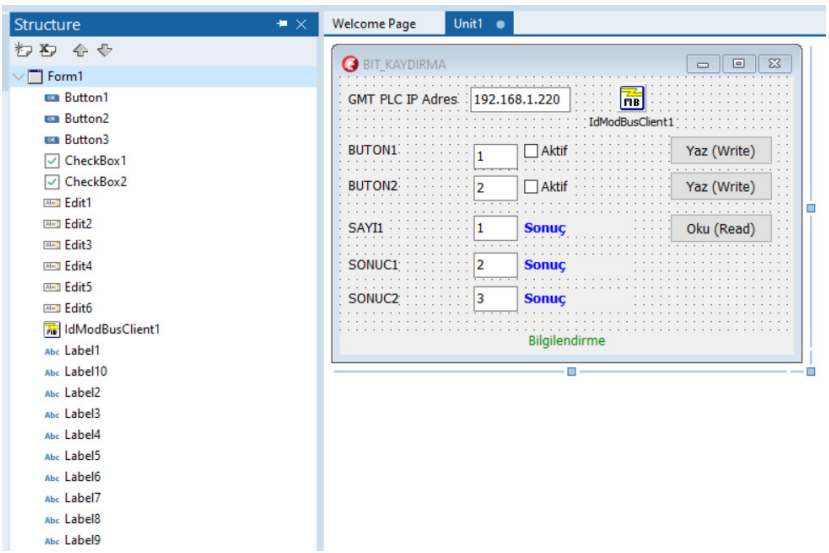
Örnek uygulama : Sağa kaydır komutunun 0'dan 1'e değişmesi ile **SAYI** adlı operandın içeriği **Bit Adeti** kadar sağa kayar.

(16 = 0000 0000 000**1** 0000 → 4 = 0000 0000 0000 0**1**00)

Sonuç kısmında **YENI_SAYI** adlı operandın değeri **4** olur. **Opr.** kısmında belirtilen **SAYI** adlı operandın değeri de 16 olarak kalır.

Bitler LSB'ye doğru kaydıkça MSB'ye **0** eklenir.

1.31. UYGULAMA: Bit Kaydırma

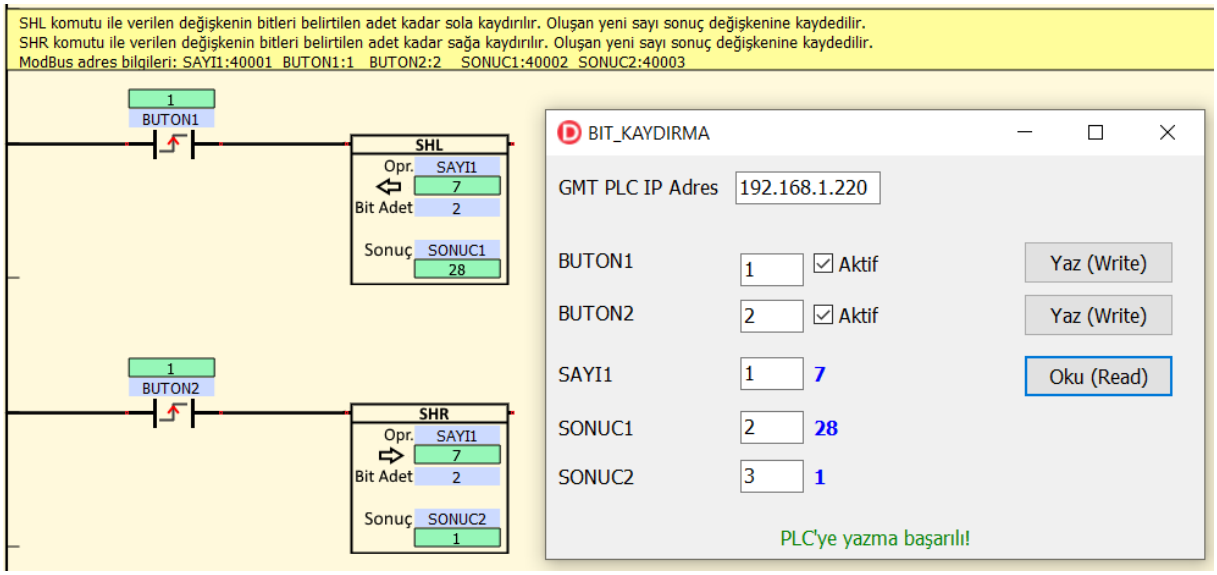


Görsel 1.105: Bit kaydırma Delphi arayüzü

```
procedure TForm1.Button2Click(Sender: TObject);
var
  Data: array[0..10] of Word;
  SONUC: Boolean; okuma_sayisi,i: Integer;
begin
    okuma_sayisi :=2;
    IdModBusClient1.Host := edit1.Text;
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit4.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label4.Caption:=inttostr(Data[0]);
      end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit5.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label7.Caption:=inttostr(Data[0]);
      end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');
   if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit6.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label9.Caption:=inttostr(Data[0]);
      end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');
end;
```

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC''ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;

procedure TForm1.Button3Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
  if IdModBusClient1.WriteCoil(StrToInt(Edit3.Text), CheckBox2.Checked) then
    Label5.Caption := 'PLC''ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

Görsel 1.106: Bit kaydırma program kodu



Görsel 1.107: Bit kaydırma ladder diyagramı

NOTLAR

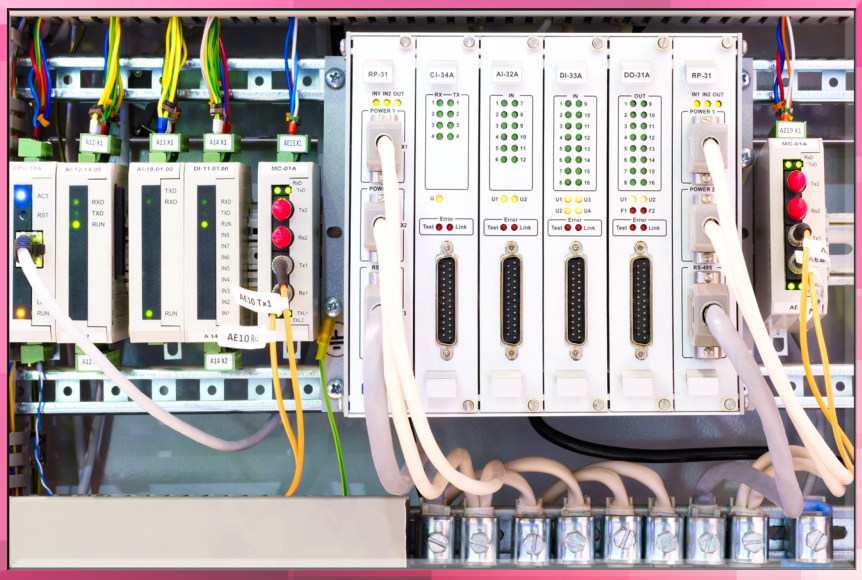
BARET Gezer vincin olduğu ve yüksekte çalışılan alanlarda, zemin, çatı arası gibi başı çarpma durumu olan yerlerde ve inşaatta çalışanlar başlarını korumak için baret kullanmalıdır.

İŞ AYAKKABISI Ayaklara yönelik tehlikelerin bulunduğu yerlerde, duruma göre, burnu çelik ayakkabı, tozluk veya çizme kullanılır.

MASKE Zararlı gazlardan, buhardan ve tozlardan korunmak için filtreli veya temiz hava verici maskeler kullanılır. Filtreli maskelerde doğru filtre seçilmesine dikkat edilmelidir.

SİPERLİK Fazla talaş, kıvılcım ve toz çıkaran işlerde (döner yüksek devirli testereler, taşlama işleri...) kullanılır. Ayrıca, çalışma esnasında derinizi etkileyen sıvılar veya maddelerle temas etmeniz gerektiğinde cilt koruyucu ilaçların tedavi edici olmadığını ve deri tahriş olmadan kullanılması gerektiğini bilmek gerekir.

GÖZLÜK Işınlardan, asitten, sıçramalardan, buhardan, taşlama veya polisajdan, talaşlı imalatta kullanılan makinelerden gözleri korumak için çeşitli nitelikte gözlükler vardır. İşin durumuna göre en uygunu seçilmeli ve kullanımı ihmal edilmemelidir.



2. ÖĞRENME BİRİMİ

FONKSİYONLAR, FONKSİYON BLOKLARI VE HABERLEŞME

KONULAR

2.1. Fonksiyon Blokları (Alt Programlar)

2.2. Fonksiyon ve Data Blok

2.3. TCP/IP Haberleşme

2.4. PID

NELER ÖĞRENECEKSİNİZ?

- Alt program
- Gerçek zaman saati
- PID

TEMEL KAVRAMLAR

Organizasyon blokları, fonksiyon ve data blok, PID, gerçek zaman.

HAZIRLIK SORULARI

1. Gerçek zaman saatinde süre artırma işlemi nasıl gerçekleşir?
2. Oransal kontrol nedir?

2. FONKSİYONLAR, FONKSİYON BLOKLARI, HABERLEŞME

Fonksiyonlar, kullanıcı tarafından yazılan küçük program parçalarıdır. Bu küçük program parçalarıyla oluşturulan bloklarsa **fonksiyon blokları** olarak adlandırılır. PLC ile mesajlaşmak için kullanılan bloksa **haberleşme bloğudur**.

2.1. FONKSİYON BLOKLARI (ALT PROGRAMLAR)

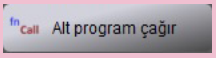
Bu bölümde açıklanan komutlar, **Ana Program** dışında çalıştırılması istenen programlardır. **Alt Program Komutu**, GLC ve GSR serisinin PLC modellerinde kullanılır. **Ekran Set Komutu**, GSR serisinin PLC'leri için kullanılır. **Mesaj Gönder Komutu** ise 496X ve 396X GLC serisinin PLC'lerinde kullanılır.

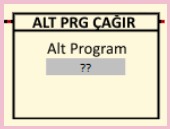
Alt Program

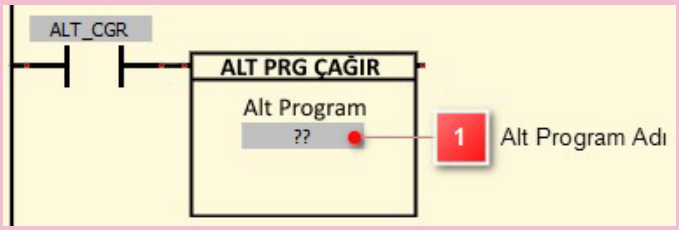
Kısa adı : AP

Kısa yol no. : Enter > 35

İkonu :

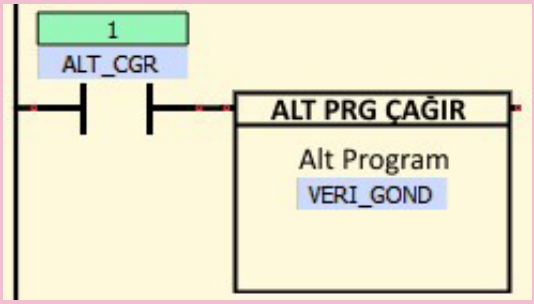






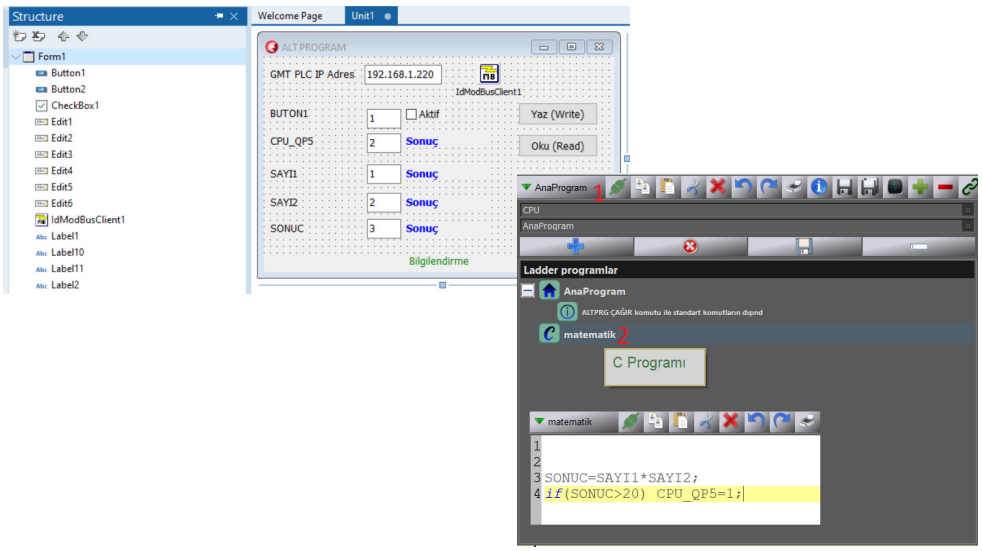
1	**Alt Program Adı:** Yürütülecek Alt Program seçimi yapılır.

Çalışması : Alt program kullanılmadan önce oluşturulur. Alt programlar, ladder diyagramı veya C programlama diliyle hazırlanır. Alt program tamamlanınca program akışı bir sonraki komuttan devam eder. Bu komut sadece ana programda kullanılır. Operandlar, ana program ve altprogramlar için ortaktır.

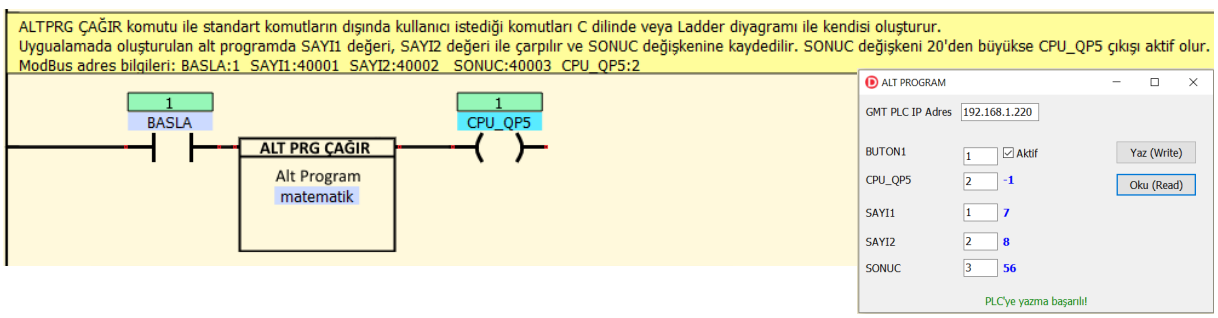


Örnek uygulama : **ALT PRG ÇAĞIR** komutu aktif olduğunda daha önceden hazırlanan **VERI_GOND** adlı alt program çalıştırılır.

2.1. UYGULAMA: Alt Program



Görsel 2.1: Alt program Delphi tasarım arayüzü



Görsel 2.2: Alt program ladder diyagramı

```
procedure TForm1.Button2Click(Sender: TObject);
var
  Data: array[0..10] of Word;
  SONUC: Boolean; okuma_sayisi,i: Integer;
begin
    okuma_sayisi :=2;
    IdModBusClient1.Host := edit1.Text;

     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit4.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label4.Caption:=inttostr(Data[0]);
      end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit5.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label7.Caption:=inttostr(Data[0]);
      end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');
```

```
    if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit6.Text), okuma_sayisi, Data) then
     begin
       for i := 0 to (okuma_sayisi - 1) do
       begin
       label9.Caption:=inttostr(Data[0]);
       end;
     end
     else
       ShowMessage('PLC''den okuma başarısız!');

     if IdModBusClient1.ReadCoil(StrToInt(Edit3.Text), SONUC) then
       Label11.Caption := BoolToStr(SONUC)
     else
       Label5.Caption := 'PLC''den okuma başarısız!'
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC''ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

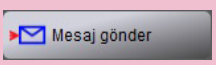
Görsel 2.3: Alt program program kodu

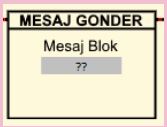
Mesaj Gönder

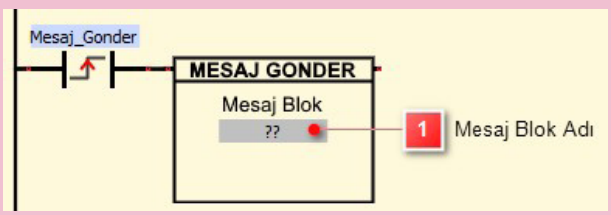
Kısa adı : MG

Kısa yol no. : Enter > 159

İkonu :





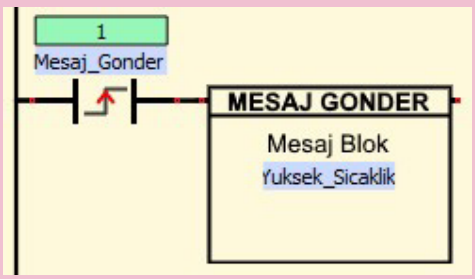


Çalışması :

1	**Mesaj Blok Adı:** Gönderilen mesajın adı seçilir.

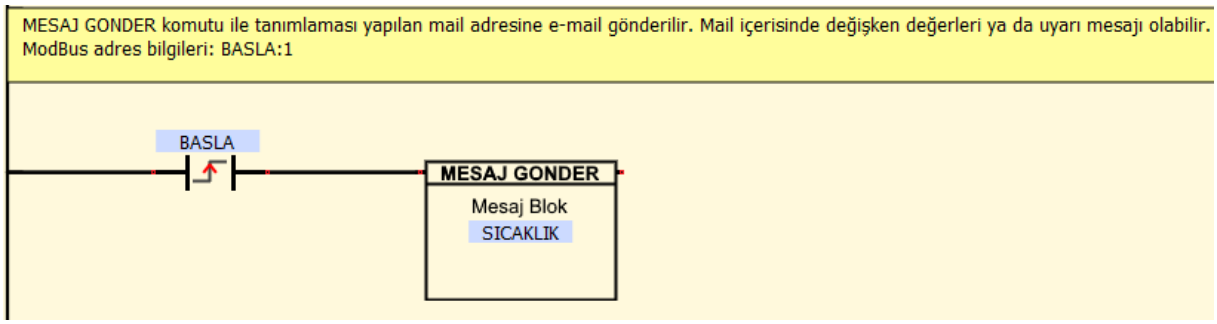
Mesaj Gönder komutu kullanılmadan önce **Mesaj Tanımları** yapılır.

Not: Gönderilen her mailden sonra en az 60 sn. beklenir.

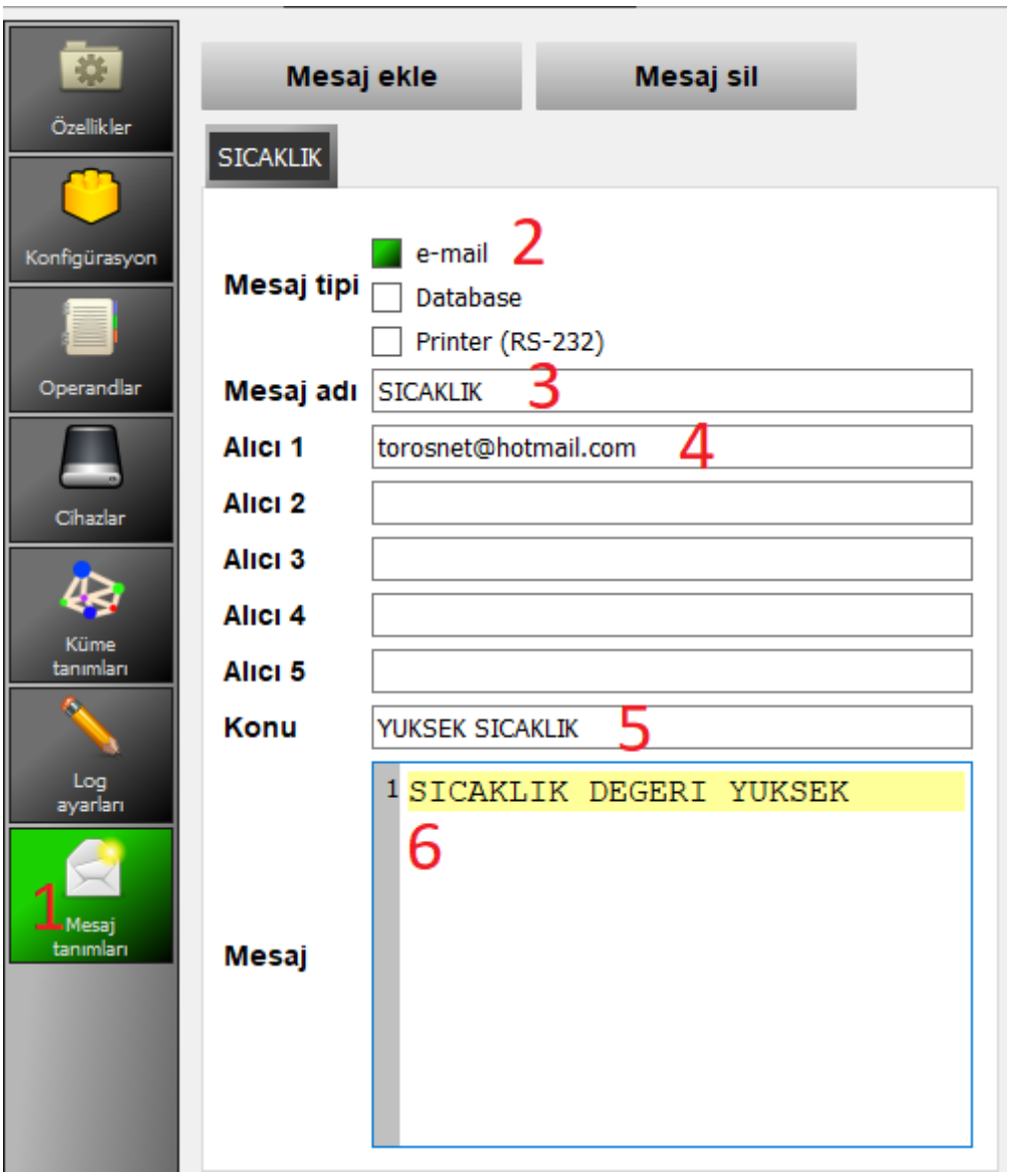


Örnek uygulama : **MESAJ GONDER** komutu aktif olduğunda daha önceden hazırlanan **Yuksek_Sicaklik** adlı mesaj, tanımlanan alıcı/alıcılara gönderilir.

2.2. UYGULAMA: Mesaj Gönder



Görsel 2.4: Mesaj gönder ladder diyagramı



Görsel 2.5: Mesaj gönder tasarım ekranı

2.1.1. RTC Komutları

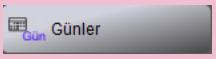
Bu komutlar, RTC özelliğine sahip 396X üst modelin GLC ile GSR serilerinin PLC'lerinde kullanılır. Komutların doğru çalışabilmesi için öncelikle Cihazlar/Ayarlar/RTC kısmından PLC saati güncellenir. Kullanıcılar, yıl boyunca herhangi bir otomatik sistemi kontrol etmek için RTC [Real Time Clock (Ril Taym Klak) (gerçek zaman saati)] komutlarından yararlanır. Dâhili pilleri sayesinde, güç kesildiğinde bile CPU'lar mevcut saat ve tarih bilgilerini silmez. RTC komutları kullanılarak yüksek düzeyde enerji tasarrufu sağlanır.

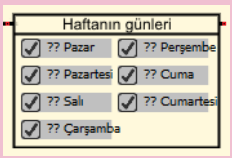
Günler Komutu

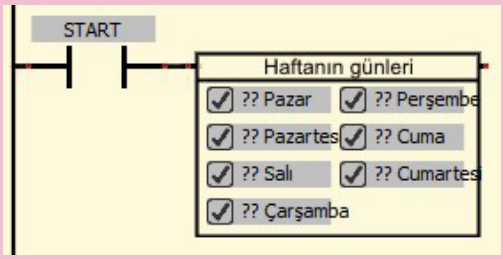
Kısa adı : GÜ

Kısa yol no. : Enter > 38

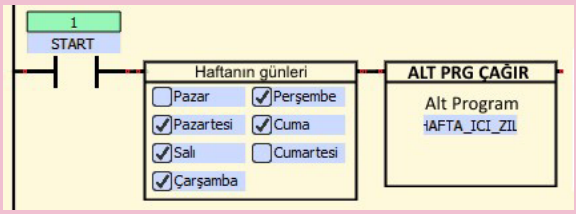
İkonu :







Çalışması : Komut çıkışının aktif olması gereken günler işaretlenir. Komut çıktısının pasif olması istenen günlerin işaretleri ise iptal edilir.



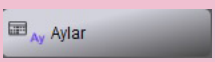
Örnek uygulama : Alt program olan **HAFTA_ICI_ZIL**, sadece gün komutunda belirtilen beş gün için yürütülür.

Aylar Komutu

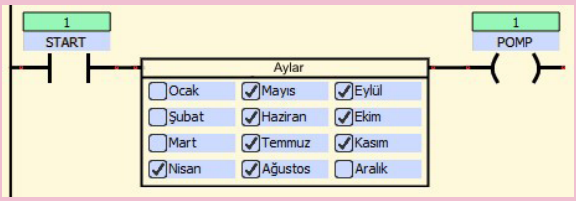
Kısa adı : AR

Kısa yol no. : Enter > 39

İkonu :



Çalışması : Komut çıkışının aktif olması gereken aylar işaretlenir. Komut çıktısının pasif olması istenen ayların işaretleri ise iptal edilir.

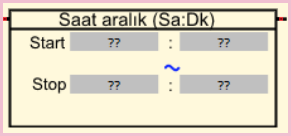


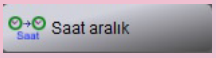
Örnek uygulama : Aylar komutunda belirtilen **POMP** çıkış rölesi 8 ay süresince çalışır.

Saat Aralık Komutu

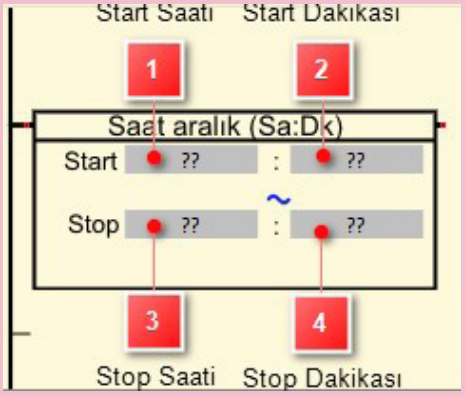
Kısa adı : SA

Kısa yol no. : Enter > 40



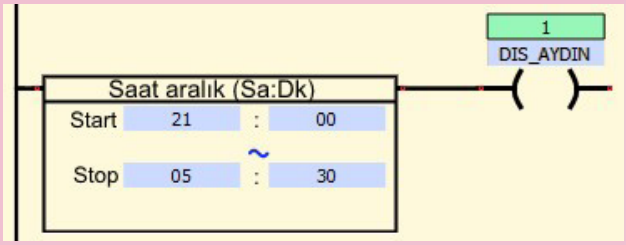
İkonu : 

Operand tipi : Word, double word, integer, real.



Çalışması :

1	**Start Saati:** Komut çıkışının aktif olma başlangıç saati.
2	**Start Dakikası:** Komut çıkışının aktif olma başlangıç dakikası.
3	**Stop Saati:** Komut çıkışının aktif olma bitiş saati.
4	**Stop Dakikası:** Komut çıkışının aktif olma bitiş dakikası.



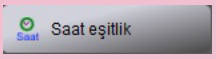
Örnek uygulama : **DIS_AYDIN** adlı çıkış rölesi saat 21.00 ile 05.30 arasında aktif olur.

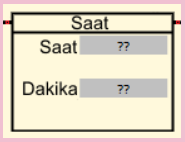
Saat Eşitlik Komutu

Kısa adı : SE

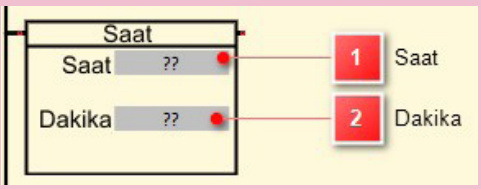
Kısa yol no. : Enter > 41

İkonu :



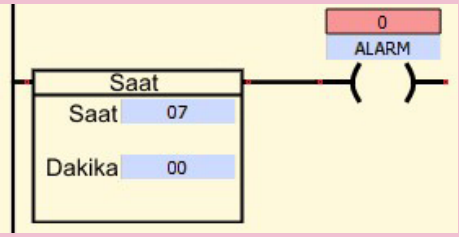


Operand tipi : Word, double word, integer, real.



Çalışması :

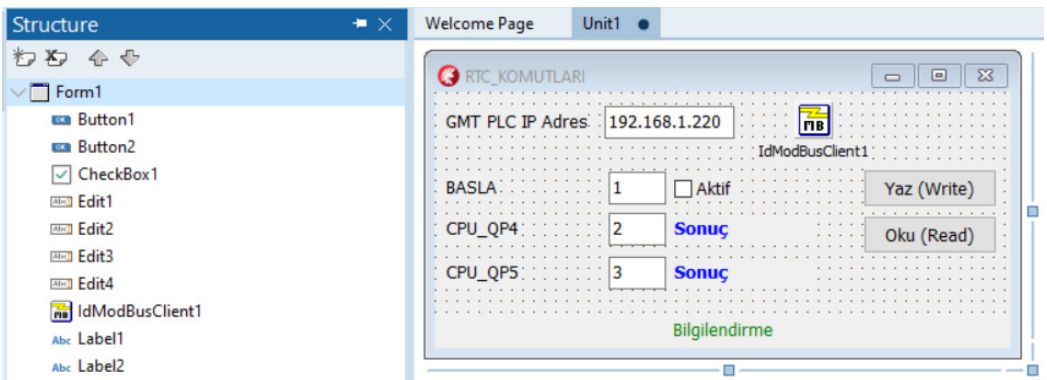
1	**Saat:** Komut çıkışının aktif olma başlangıç saati.
2	**Dakika:** Komut çıkışının aktif olma başlangıç dakikası.



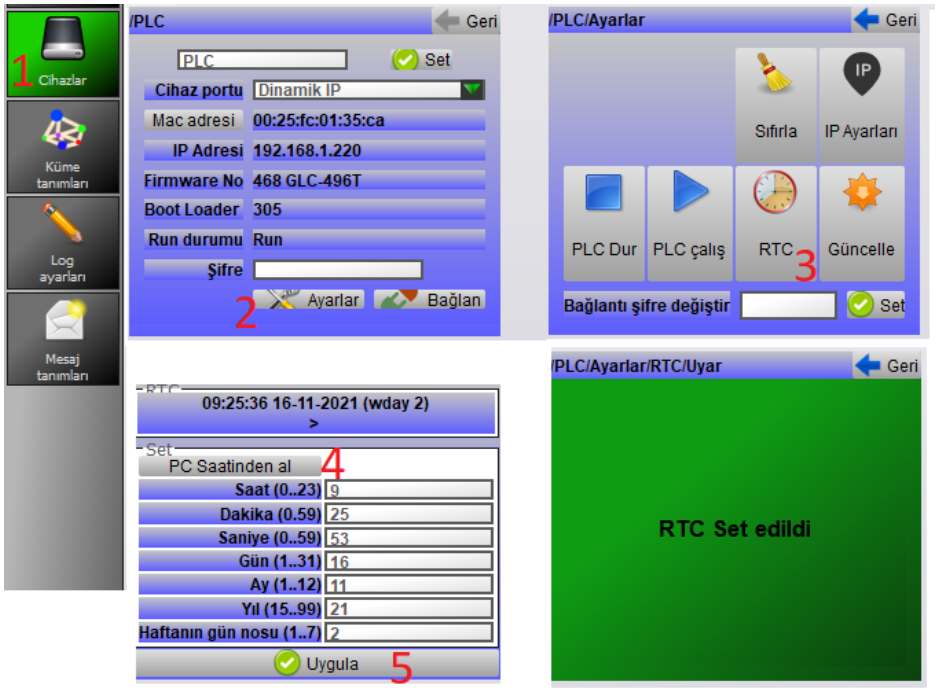
Örnek uygulama : **ALARM** adlı çıkış rölesi, saat 07.00'de çalışır ve 07.01'de ise durur.

2.3. UYGULAMA: RTC Komutları

RTC komutları kullanılarak eylül, ekim, kasım aylarında; salı ve cuma günleri saat 11.50'de başlayarak onar dakika boyunca sera sulama sistemi kurulur. Sulama işlemi 30 dakika sonra bitirilir. Aynı zamanda sulama işleminin başladığını gösteren bir çıkışa da uyarı lambası konulur. Bu lamba sulama başladıktan 1 dakika sonra söner.



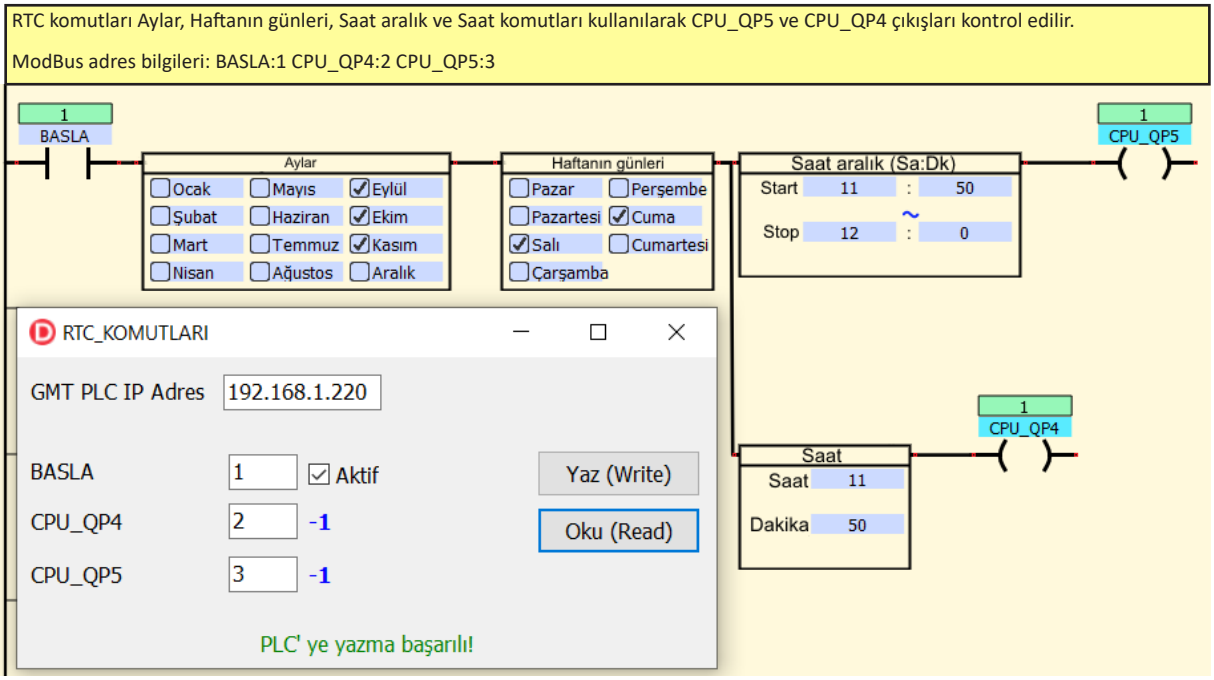
Görsel 2.6: RTC komutları Delphi tasarım arayüzü



Görsel 2.7: RTC komutları tasarım ekranı

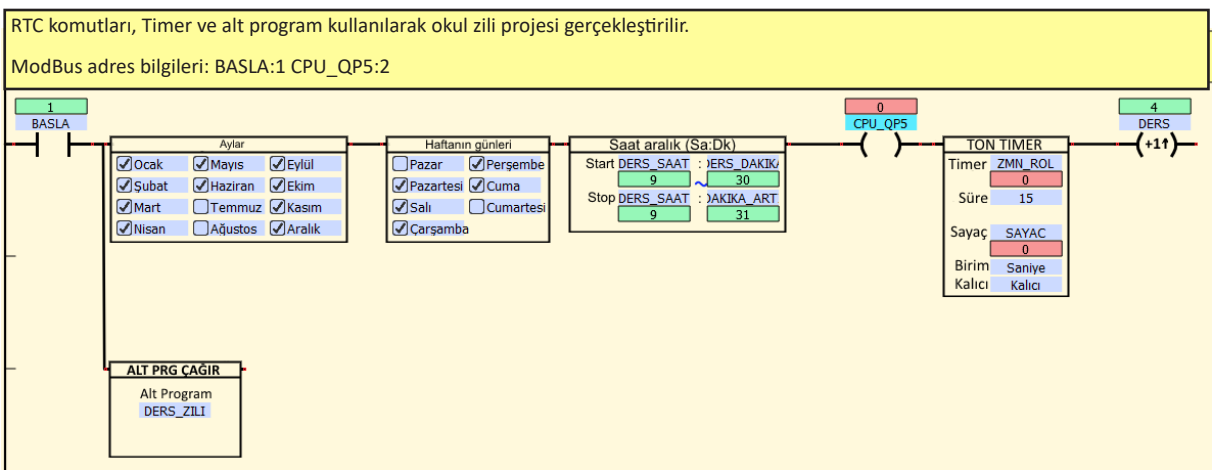
```
procedure TForm1.Button2Click(Sender: TObject);
var
  Data: array[0..10] of Word;
  SONUC: Boolean; okuma_sayisi,i: Integer;
begin
    okuma_sayisi :=2;
    IdModBusClient1.Host := edit1.Text;
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit4.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
      begin
      label7.Caption:=inttostr(Data[0]);
      end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');

   if IdModBusClient1.ReadCoil(StrToInt(Edit3.Text), SONUC) then
    Label4.Caption := BoolToStr(SONUC)
   else
    Label5.Caption := 'PLC''den okuma başarısız!';

    if IdModBusClient1.ReadCoil(StrToInt(Edit4.Text), SONUC) then
    Label7.Caption := BoolToStr(SONUC)
   else
    Label5.Caption := 'PLC''den okuma başarısız!';
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

Görsel 2.8: RTC komutları program kodu

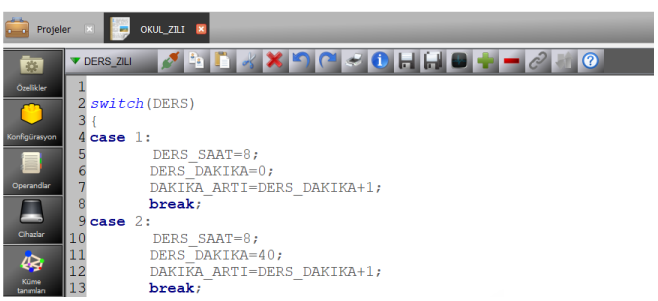


Görsel 2.9: RTC komutları ladder diyagramı

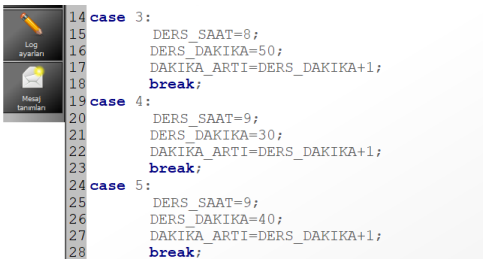
2.4. UYGULAMA: RTC ve Timer ile Okul Zili Uygulaması



Görsel 2.10: RTC ve Timer ile okul zili uygulaması ladder diyagramı



```
switch(DERS)
{
case 1:
       DERS_SAAT=8;
       DERS_DAKIKA=0;
       DAKIKA_ARTI=DERS_DAKIKA+1;
       break;
case 2:
       DERS_SAAT=8;
       DERS_DAKIKA=40;
       DAKIKA_ARTI=DERS_DAKIKA+1;
       break;
```



```
case 3:
       DERS_SAAT=8;
       DERS_DAKIKA=50;
       DAKIKA_ARTI=DERS_DAKIKA+1;
       break;
case 4:
       DERS_SAAT=9;
       DERS_DAKIKA=30;
       DAKIKA_ARTI=DERS_DAKIKA+1;
       break;
case 5:
       DERS_SAAT=9;
       DERS_DAKIKA=40;
       DAKIKA_ARTI=DERS_DAKIKA+1;
       break;
```

```
case 6:
      DERS_SAAT=10;
      DERS_DAKIKA=20;
      DAKIKA_ARTI=DERS_DAKIKA+1;
      break;
case 7:
      DERS_SAAT=10;
      DERS_DAKIKA=30;
      DAKIKA_ARTI=DERS_DAKIKA+1;
      break;
case 8:
      DERS_SAAT=11;
      DERS_DAKIKA=10;
      DAKIKA_ARTI=DERS_DAKIKA+1;
      break;
case 9:
      DERS_SAAT=11;
      DERS_DAKIKA=20;
      DAKIKA_ARTI=DERS_DAKIKA+1;
      break;
```

```
case 10:
      DERS_SAAT=12;
      DERS_DAKIKA=0;
      DAKIKA_ARTI=DERS_DAKIKA+1;
      break;
case 11:
      DERS_SAAT=12;
      DERS_DAKIKA=10;
      DAKIKA_ARTI=DERS_DAKIKA+1;
      break;
case 12:
      DERS_SAAT=12;
      DERS_DAKIKA=50;
      DAKIKA_ARTI=DERS_DAKIKA+1;
      break;
}

if(DERS>12) DERS=1;
```

Görsel 2.11: RTC ve Timer ile okul zili uygulaması ladder diyagramı

2.2. ORGANİZASYON BLOKLARI (KÜMELER-ARRAYS)

GMT PLC'lerde birden fazla komutları bir arada ve bir anda kullanma teknikleri de vardır. Bu yapı kümeler diye isimlenidirilirler.

2.2.1. Kümeler (Arrays)

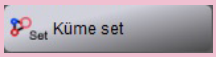
Bu komutların tamamı, GLC ve GSR serisinin PLC modellerinde kullanılır. Program içinde ortak özelliğe sahip operandlar üzerinde, tek bir komutla çeşitli operasyonların yapılması gerekli olabilir. Örneğin tek komutla tüm çıkışlar aktif ya da pasif yapılır. Böyle durumlarda operandlar, küme altına alınır ve operasyonlar kümelere uygulanır. Komutları kullanabilmek için öncelikle **Küme Tanımlaması** yapılır.

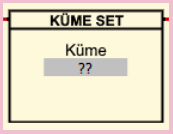
Küme Set Komutu

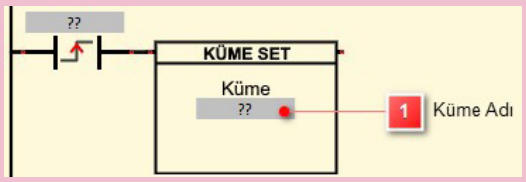
Kısa adı : KS

Kısa yol no. : Enter > 74

İkonu :





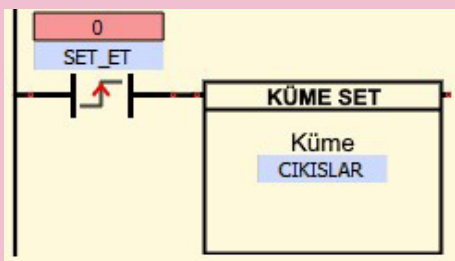


1	**Küme Adı:** Komutun uygulandığı küme adı seçilir.

Çalışması : Bu komutla kümenin tüm elemanları (operandları) set (1) olur. Elemanlar, **Küme Reset Komutu** ya da **Reset Komutu** ile reset (0) yapılıncaya kadar konumlarını korur.

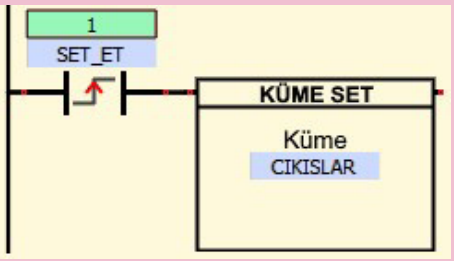
Not: Komut kullanılmadan önce **Küme Tanımlaması** yapılır.

Örnek uygulama :



Operand	Değer	Açıklama
CPU_QP0	0	QP 0
CPU_QP1	0	QP 1
CPU_QP2	0	QP 2
CPU_QP3	0	QP 3
CPU_QP4	0	QP 4
CPU_QP5	0	QP 5

SET_ET kontağının aktif olmasıyla **CIKISLAR** adlı kümenin elemanları **1** olur.



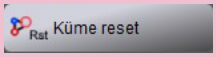
Operand	Değer	Açıklama
CPU_QP0	1	QP 0
CPU_QP1	1	QP 1
CPU_QP2	1	QP 2
CPU_QP3	1	QP 3
CPU_QP4	1	QP 4
CPU_QP5	1	QP 5

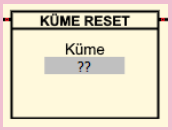
Küme Reset Komutu

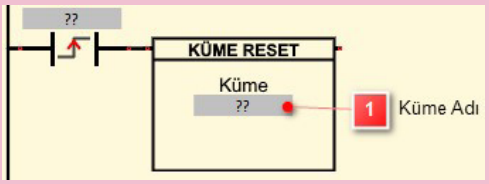
Kısa adı : KR

Kısa yol no. : Enter > 75

İkonu :



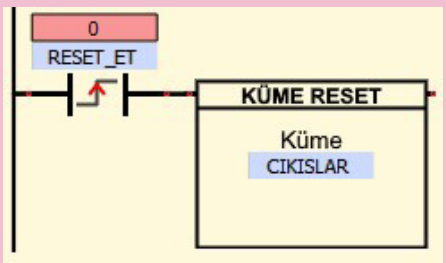




1	**Küme Adı:** Komutun uygulandığı küme adı seçilir.

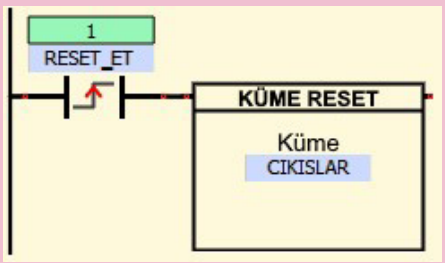
Çalışması : Kümenin tüm elemanlarını reset (0) yapar. Elemanlar, **Küme Set Komutu** ya da **Set Komutu** ile set (1) yapılıncaya kadar konumlarını korur.

Not: Komut kullanılmadan önce **Küme Tanımlaması** yapılır.



Operand	Değer	Açıklama
CPU_QP0	1	QP 0
CPU_QP1	1	QP 1
CPU_QP2	1	QP 2
CPU_QP3	1	QP 3
CPU_QP4	1	QP 4
CPU_QP5	1	QP 5

Örnek uygulama : **RESET_ET** kontağının aktif olmasıyla **CIKISLAR** adlı kümenin elemanları **0** olur.



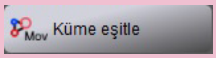
Operand	Değer	Açıklama
CPU_QP0	0	QP 0
CPU_QP1	0	QP 1
CPU_QP2	0	QP 2
CPU_QP3	0	QP 3
CPU_QP4	0	QP 4
CPU_QP5	0	QP 5

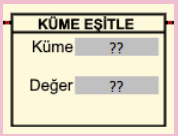
Küme Eşitle Komutu

Kısa adı : KŞ

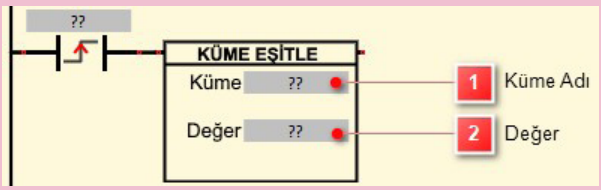
Kısa yol no. : Enter > 76

İkonu :





Operand tipleri : Word, double word, integer, real.

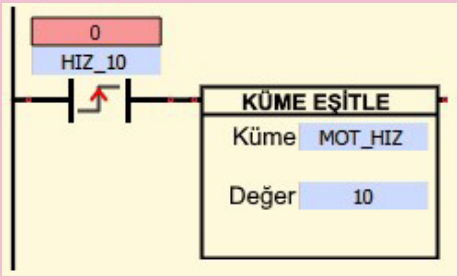


Çalışması :

1	**Küme Adı:** Komutun uygulandığı küme adı seçilir.
2	**Değer:** Küme elemanlarının değerleri yazılır.

Seçilen kümenin tüm elemanlarına aynı değer verilir.

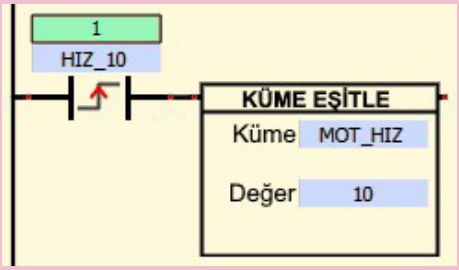
Not: Komut kullanılmadan önce **Küme Tanımlaması** yapılır.



Operand	Değer	Açıklama
MOT_HIZ_1	250	
MOT_HIZ_2	50	
MOT_HIZ_3	25	
MOT_HIZ_4	100	
MOT_HIZ_5	125	
MOT_HIZ_6	200	

HIZ_10 kontağının aktif olmasıyla **MOT_HIZ** adlı kümenin elemanlarının içeriği **10** olur.

Örnek uygulama :



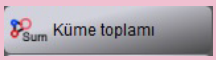
Operand	Değer	Açıklama
MOT_HIZ_1	10	
MOT_HIZ_2	10	
MOT_HIZ_3	10	
MOT_HIZ_4	10	
MOT_HIZ_5	10	
MOT_HIZ_6	10	

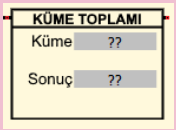
Küme Toplamı

Kısa adı : KT

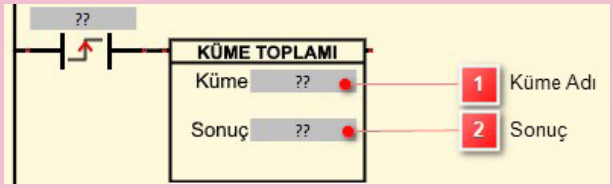
Kısa yol no. : Enter > 77

İkonu :





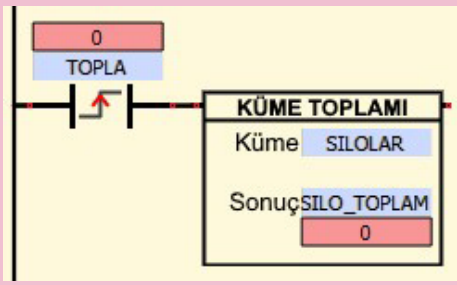
Operand tipleri : Integer, real.



1	**Küme Adı:** Komutun uygulandığı küme adı seçilir.
2	**Değer:** Küme elemanlarının değerleri yazılır.

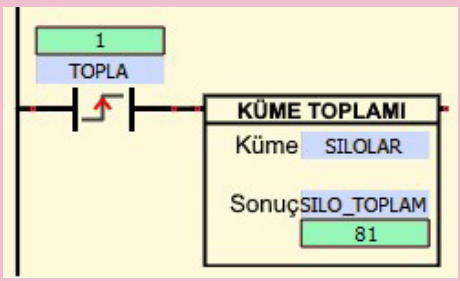
Çalışması : Seçilen kümenin tüm elemanlarına aynı değer verilir.

Not: Komut kullanılmadan önce **Küme Tanımlaması** yapılır.



Operand	Değer	Açıklama
SILO_1	10	
SILO_2	3	
SILO_3	5	
SILO_4	8	
SILO_5	20	
SILO_6	35	
SILO_TOPL...	0	

Örnek uygulama : **TOPLA** kontağının aktif olmasıyla **SILOLAR** adlı kümenin elemanlarının tamamı toplanır ve sonucu **Sonuç** kısmında belirtilen **SILO_TOPLAM** adlı operanda yazılır.



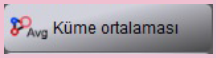
Operand	Değer	Açıklama
SILO_1	10	
SILO_2	3	
SILO_3	5	
SILO_4	8	
SILO_5	20	
SILO_6	35	
SILO_TOPL...	81	

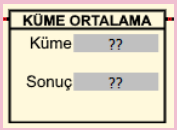
Küme Ortalaması

Kısa adı : KL

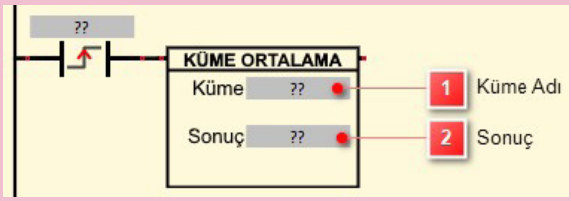
Kısa yol no. : Enter > 78

İkonu :





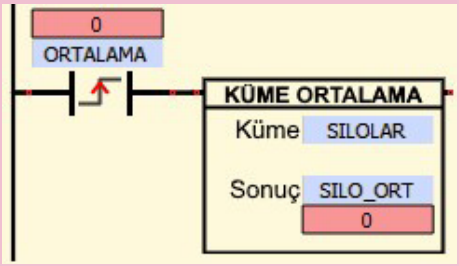
Operand tipleri : Integer, real.



1	**Küme Adı:** Komutun uygulandığı küme adı seçilir.
2	**Sonuç:** Küme elemanlarının ortalamasının kaydedildiği operand adı yazılır.

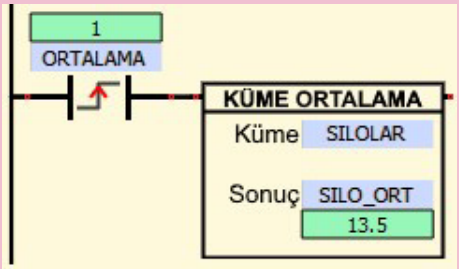
Çalışması : Kümenin tüm elemanlarının içerik ortalaması alınır.

Not: Komut kullanılmadan önce **Küme Tanımlaması** yapılır.



Operand	Değer	Açıklama
SILO_1	10	
SILO_2	3	
SILO_3	5	
SILO_4	8	
SILO_5	20	
SILO_6	35	
SILO_ORT	0	

Örnek uygulama : **ORTALAMA** kontağının aktif olmasıyla **SILOLAR** adlı kümenin elemanlarının tamamının ortalaması, **Sonuç** kısmında belirtilen **SILO_ORT** adlı operanda yazılır. **SILO_ORT** real tip olarak seçilir. Aksi durumda sonuç **13** görülür.

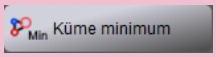


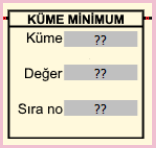
Operand	Değer	Açıklama
SILO_1	10	
SILO_2	3	
SILO_3	5	
SILO_4	8	
SILO_5	20	
SILO_6	35	
SILO_ORT	13.5	

Küme Minimum

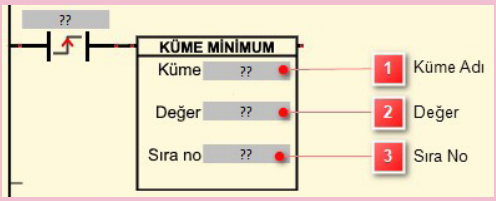
Kısa adı : KI

Kısa yol no. : Enter > 79

İkonu : 



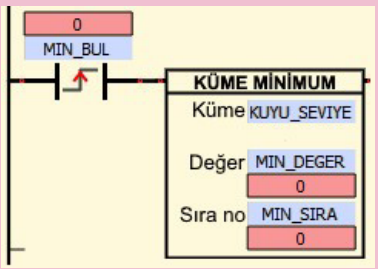
Operand tipleri : Integer, real.



1	**Küme Adı:** Komutun uygulandığı küme adı seçilir.
2	**Değer:** Minimum değerin kaydedildiği operand adı yazılır.
3	**Sıra No.:** Minimum değere sahip operandın sırasının kaydedildiği operand adı yazılır.

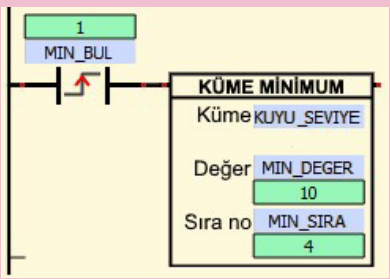
Çalışması : Bu komut, kümenin tüm elemanlarının içinden minimum değere sahip operandın içeriğini ve sırasını bulur.

Not: Komut kullanılmadan önce **Küme Tanımlaması** yapılır.



Operand	Değer	Açıklama
KUYU_1	50	
KUYU_2	35	
KUYU_3	25	
KUYU_4	45	
KUYU_5	10	
MIN_DEGER	0	
MIN_SIRA	0	

Örnek uygulama : **MIN_BUL** kontağının aktif olmasıyla minimum değere sahip operand grup içinden bulunur. Minimum değere sahip operandın içeriği **MIN_DEGER** içeriğine yazılır. Sıra numarası ise **MIN_SIRA** operandının içeriğine yazılır (Sıra 0'dan başlar.).

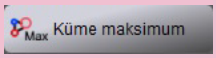


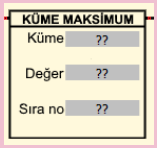
Operand	Değer	Açıklama
KUYU_1	50	
KUYU_2	35	
KUYU_3	25	
KUYU_4	45	
KUYU_5	10	
MIN_DEGER	10	
MIN_SIRA	4	

Küme Maksimum

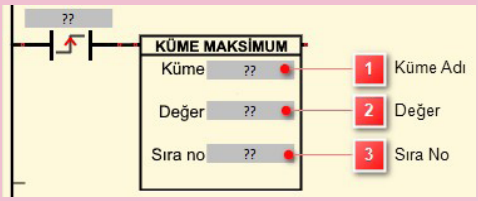
Kısa adı : KA

Kısa yol no. : Enter > 80

İkonu : 



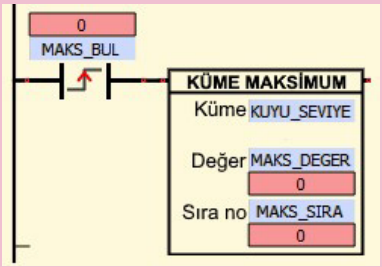
Operand tipleri : Double word, integer, real.



1	**Küme Adı:** Komutun uygulandığı küme adı seçilir.
2	**Değer:** Maksimum değerin kaydedildiği operand adı yazılır.
3	**Sıra No.:** Maksimum değere sahip operandın sırasının kaydedildiği operand adı yazılır.

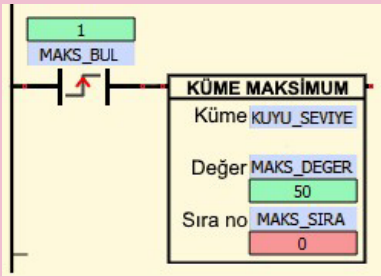
Çalışması : Bu komut, kümenin tüm elemanlarının içinden maksimum değere sahip operandın içeriğini ve sırasını bulur.

Not: Komut kullanılmadan önce **Küme Tanımlaması** yapılır.



Operand	Değer	Açıklama
KUYU_1	50	
KUYU_2	35	
KUYU_3	25	
KUYU_4	45	
KUYU_5	10	
MAKS_DEG...	0	
MAKS_SIRA	0	

Örnek uygulama : **MAKS_BUL** kontağının aktif olmasıyla maksimum değere sahip operand grup içinden bulunur. Maksimum değere sahip operandın içeriği **MAKS_DEGER** içeriğine yazılır. Sıra numarası da **MAKS_SIRA** operandının içeriğine yazılır (Sıra 0'dan başlar.).



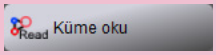
Operand	Değer	Açıklama
KUYU_1	50	
KUYU_2	35	
KUYU_3	25	
KUYU_4	45	
KUYU_5	10	
MAKS_DEG...	50	
MAKS_SIRA	0	

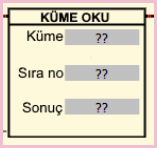
Küme Oku Komutu

Kısa adı : KO

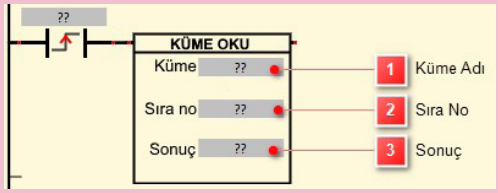
Kısa yol no. : Enter > 81

İkonu :





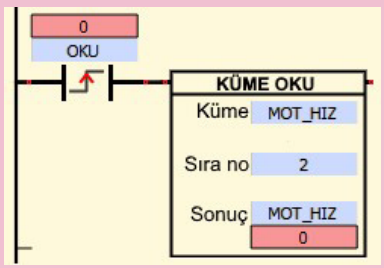
Operand tipleri : Integer, real.



1	**Küme Adı:** Komutun uygulandığı küme adı seçilir.
2	**Sıra No.:** Okunan operandın sıra numarası yazılır.
3	**Sonuç:** Kaydedilen oprandın sonucu yazılır.

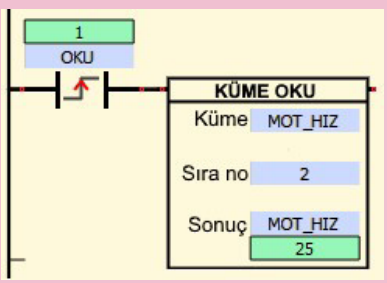
Çalışması : Bu komut, tanımlanan satır numarasının içeriğini okur ve **Sonuç** bölümünde belirtilen operanda yazar.

Not: Komut kullanılmadan önce **Küme Tanımlaması** yapılır.



Operand	Değer	Açıklama
MOT_HIZ_1	250	
MOT_HIZ_2	50	
MOT_HIZ_3	25	
MOT_HIZ_4	100	
MOT_HIZ_5	125	
MOT_HIZ_6	200	

Örnek uygulama : **OKU** kontağının aktif olmasıyla **MOT_HIZ** grubunun 2. elemanının içeriği okunur ve **MOT_HIZ** operandına yazılır (Sıra 0'dan başlar.).



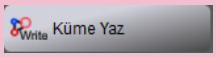
Operand	Değer	Açıklama
MOT_HIZ_1	250	
MOT_HIZ_2	50	
MOT_HIZ_3	25	
MOT_HIZ_4	100	
MOT_HIZ_5	125	
MOT_HIZ_6	200	

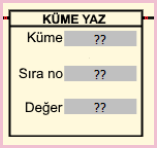
Küme Değer Atama Komutu

Kısa adı : KD

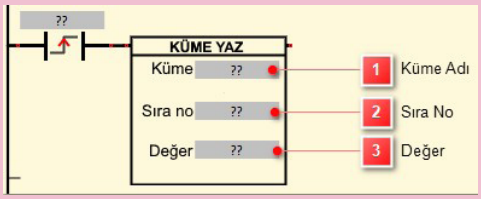
Kısa yol no. : Enter > 82

İkonu :





Operand tipleri : Integer, real.



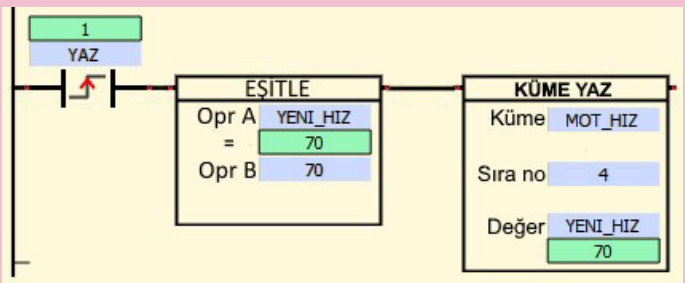
1	**Küme Adı:** Komutun uygulandığı küme adı seçilir.
2	**Sıra No.:** Değer yazılan operandın sıra numarasıdır.
3	**Değer:** Sıra numarası verilen operanda yazılan değerdir.

Çalışması : Bu komut, **Değer** alanında girilen değeri seçilen grup operandlarının içeriğine yazar.

Not: Komut kullanılmadan önce **Küme Tanımlaması** yapılır.

Operand	Değer	Açıklama
MOT_HIZ_1	250	
MOT_HIZ_2	50	
MOT_HIZ_3	25	
MOT_HIZ_4	100	
MOT_HIZ_5	125	
MOT_HIZ_6	200	

Operand	Değer	Açıklama
MOT_HIZ_1	250	
MOT_HIZ_2	50	
MOT_HIZ_3	25	
MOT_HIZ_4	100	
MOT_HIZ_5	70	
MOT_HIZ_6	200	



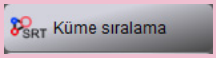
Örnek uygulama : **YAZ** kontağının aktif olmasıyla **YENI_HIZ** operandının içeriğine **70** değeri yazılır. Sonra bu operandın değeri **4** sıra numaralı **MOT_HIZ_5** operandında yazılır (Sıra 0'dan başlar.).

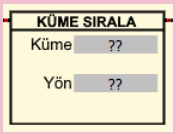
Küme Sıralama Komutu

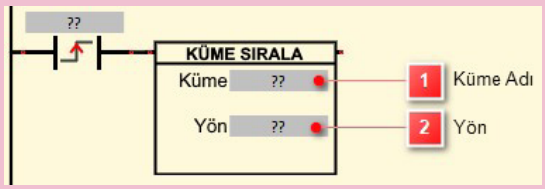
Kısa adı : KI

Kısa yol no. : Enter > 124

İkonu :







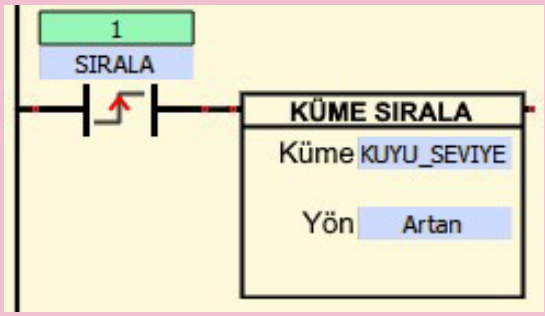
1	**Küme Adı:** Komutun uygulandığı küme adı seçilir.
2	**Yön:** Artan/Azalan sıralama yönü seçilir.

Çalışması : Bu komut, küme elemanlarının belirtilen yönde sıralamasını yapar.

Not: Komut kullanılmadan önce **Küme Tanımlaması** yapılır.

Operand	Değer	Açıklama
KUYU_1	50	
KUYU_2	35	
KUYU_3	25	
KUYU_4	45	
KUYU_5	10	

Operand	Değer	Açıklama
KUYU_1	10	
KUYU_2	25	
KUYU_3	35	
KUYU_4	45	
KUYU_5	50	

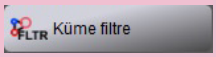


Örnek uygulama : **SIRALA** kontağının aktif olmasıyla **KUYU_SEVIYE** kümesinin operandları, küçük değerden büyük değere doğru sıralanır. **Azalan** seçeneğin seçilmesi durumunda ise büyükten küçüğe doğru sıralanır.

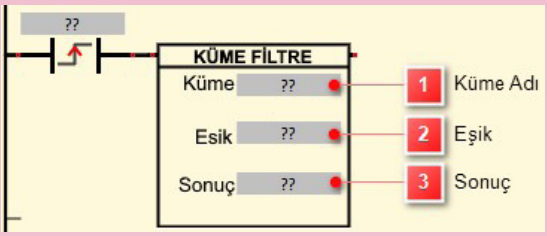
Küme Filtre Komutu

Kısa adı : KF

Kısa yol no. : Enter > 125

İkonu : 

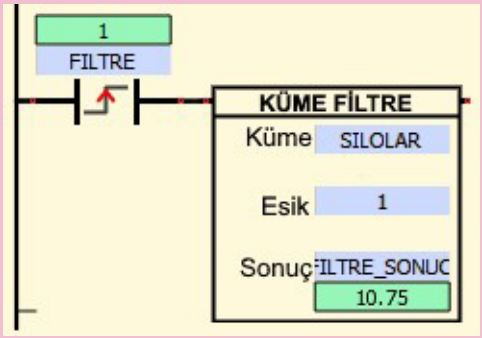
Operand tipleri : Integer, real.



1	**Küme Adı:** Komutun uygulandığı küme adı seçilir.
2	**Eşik:** Filtreleme eşik değeri yazılır.
3	**Sonuç:** Filtreleme sonucunun kaydedildiği operandın adı yazılır.

Çalışması : Bu komut, gruptaki operand değerlerini sıralar. Maksimum ve minimum değerlerden eşik değeri hariç tutulduktan sonra kalan değerlerin ortalamasını **Sonuç** operandına yazar.

Not: Komut kullanılmadan önce **Küme Tanımlaması** yapılır.



Operand	Değer	Açıklama
SILO_1	10	
SILO_2	3	
SILO_3	5	
SILO_4	8	
SILO_5	20	
SILO_6	35	

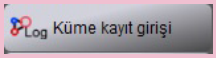
Örnek uygulama : **FILTRE** kontağının aktif olmasıyla **SILOLAR** adlı kümenin operandları sıralanır. Min. 3 ve max. 35 değerleri dışında kalan 5, 8, 10, 20 değerlerinin ortalaması hesaplanır ve sonuç değeri (10,75) **FILTRE_SONUC** operandına yazılır. Operand tipi **real** olarak seçilir.

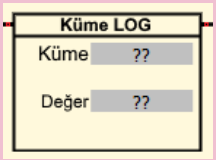
Küme Kayıt Girişi

Kısa adı : KG

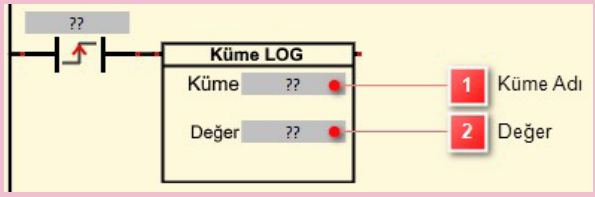
Kısa yol no. : Enter > 126

İkonu :





Operand tipleri : Integer, real.



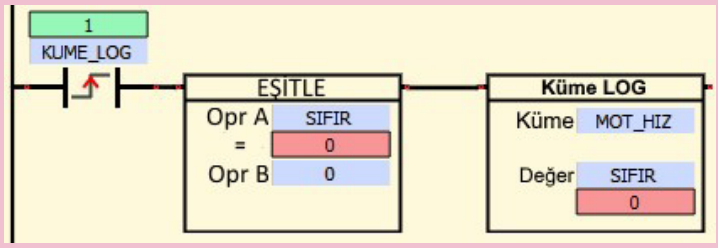
1	**Küme Adı:** Komutun uygulandığı küme adı seçilir.
2	**Değer:** Küme elemanlarına kaydedilen değer yazılır.

Çalışması : Bu komut, **Değer** alanında belirtilen değeri ilk satırdaki operanda yazar. Diğer verileri bir adım aşağı kaydırır. Komutun her aktif oluşunda bu kayma devam eder.

Not: Komut kullanılmadan önce **Küme Tanımlaması** yapılır.

Operand	Değer	Açıklama
MOT_HIZ_1	250	
MOT_HIZ_2	50	
MOT_HIZ_3	25	
MOT_HIZ_4	100	
MOT_HIZ_5	125	
MOT_HIZ_6	200	

Operand	Değer	Açıklama
MOT_HIZ_1	0	
MOT_HIZ_2	250	
MOT_HIZ_3	50	
MOT_HIZ_4	25	
MOT_HIZ_5	100	
MOT_HIZ_6	125	



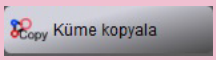
Örnek uygulama : **KUME_LOG** kontağının aktif olmasıyla **MOT_HIZ** adlı kümenin ilk operandına **SIFIR** içindeki veri yazılır. Diğer veriler bir aşağı kayar. Komutun her aktif oluşunda bu kayma devam eder.

Küme Kopyala

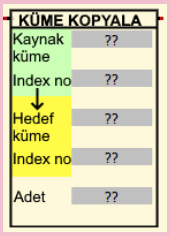
Kısa adı : KP

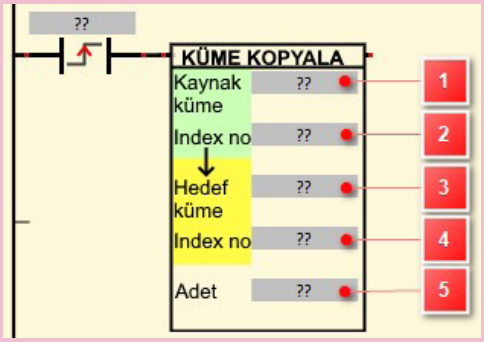
Kısa yol no. : Enter > 174

İkonu :



Operand tipleri : Integer, real.



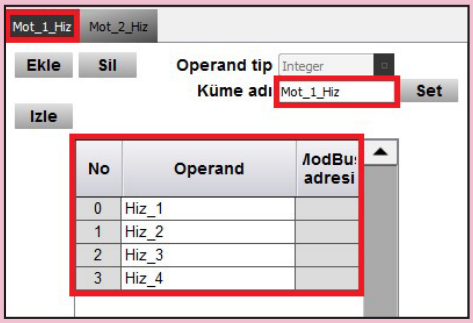


1	**Kaynak Küme:** Kopyalama için kaynak küme adı seçilir.
2	**Index No.:** Kopyalanan kümenin elemanlarının başlangıç numarası tanımlanır.
3	**Hedef Küme:** Kopyalama için hedef küme adı seçilir.
4	**Index No.:** Hedef kümede operandların yapıştırıldığı başlangıç numarası tanımlanır.
5	**Adet:** Kaç adet operandın kopyalanıp yapıştıralacağı tanımlanır.

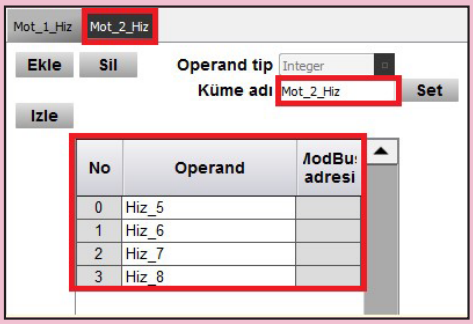
Çalışması : Bu komut, **Kaynak Küme** alanındaki kümenin **Index No.**'dan itibaren **Adet** alanında belirtilen sayıdaki operandını, **Hedef Küme** alanındaki kümenin **Index No.**'dan itibaren operandlarına yapıştırır.

Not: Komut kullanılmadan önce **Küme Tanımlaması** yapılır.

Örnek uygulama :



Öncelikle 4 adet integer türünde motor hızlarına ait operandları barındıran **Mot_1_Hiz** adlı ilk küme tanımlanır.



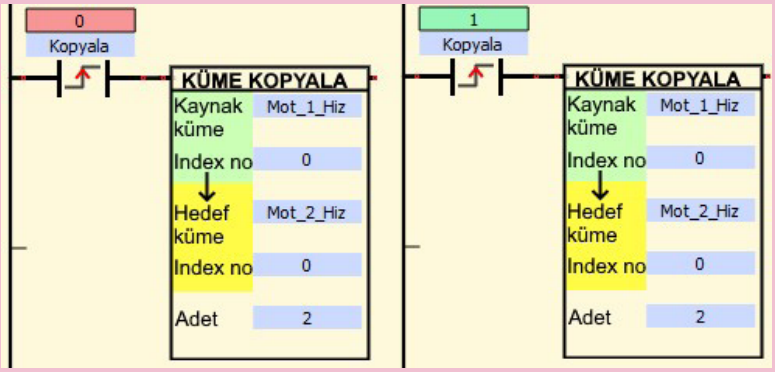
Sonra 4 adet integer türünde motor hızlarına ait operandları barındıran **Mot_2_Hiz** adlı ikinci küme tanımlanır.

Operandların açılış değerleri aşağıdaki gibi tanımlanır.

No	Operand	Açıklama	Veri tipi	Açılış değeri	Kalıcı hafıza
82	Hiz_1		Integer	25	
83	Hiz_2		Integer	50	
84	Hiz_3		Integer	75	
85	Hiz_4		Integer	100	
86	Hiz_5		Integer	100	
87	Hiz_6		Integer	200	
88	Hiz_7		Integer	300	
89	Hiz_8		Integer	400	
90	Kopyala		Bit	0	

Operand	Değer	Açıklama
Hiz_1	25	
Hiz_2	50	
Hiz_3	75	
Hiz_4	100	
Hiz_5	100	
Hiz_6	200	
Hiz_7	300	

Operand	Değer	Açıklama
Hiz_2	50	
Hiz_3	75	
Hiz_4	100	
Hiz_5	100	
Hiz_6	200	
Hiz_7	300	
Hiz_8	400	



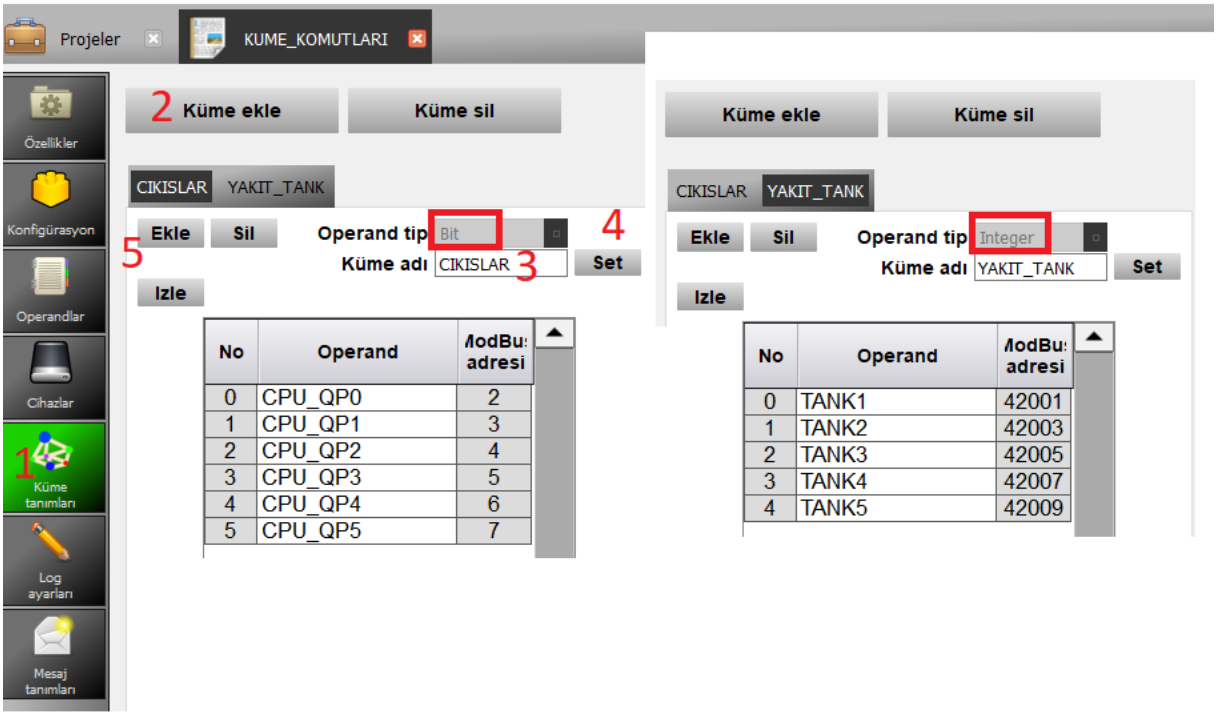
Operand	Değer	Açıklama
Hiz_1	25	
Hiz_2	50	
Hiz_3	75	
Hiz_4	100	
Hiz_5	100	
Hiz_6	200	
Hiz_7	300	

Operand	Değer	Açıklama
Hiz_2	50	
Hiz_3	75	
Hiz_4	100	
Hiz_5	25	
Hiz_6	50	
Hiz_7	300	
Hiz_8	400	

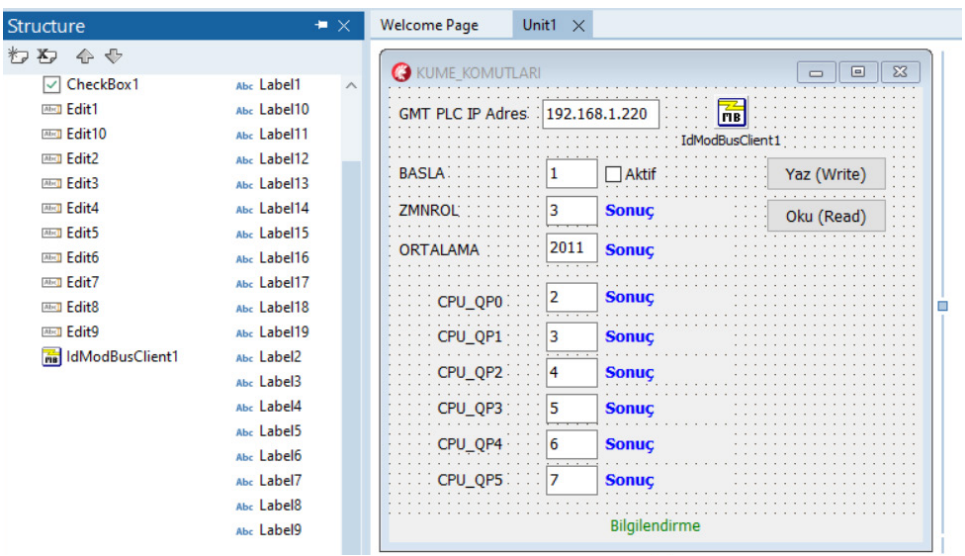
Kopyala kontağının aktif olmasıyla **Mot_1_Hiz** adlı kümenin 0. indexten sonraki 2 operand değerleri, **Mot_2_Hiz** adlı kümenin 0. indexinden itibaren operandlarına kopyalanır.

2.5. UYGULAMA: Küme Komutları

Uygulamada ilk olarak **CIKISLAR** ve **YAKIT_TANKI** isimli iki tane küme oluşturulur. **CIKISLAR** kümesine **CPU_QP0 – CPU_QP5** çıkışları eklenir. **YAKIT_TANKI** kümesine **TANK1-TANK5** isimli değişkenler eklenir. Ladder diyagramda **BASLA** komutunun ardından tank değerlerinin ortalaması **KÜME ORTALAMA** komutuyla bulunur. **ORTALAMA** değeri 30'dan büyükse **CIKISLAR** kümesine ait birimler, **KÜME SET** komutuyla aktif edilir. 5 saniye olarak belirlenen aktif kalma süresi sonunda **KÜME RESET** komutuyla çıkışlar reset edilir.



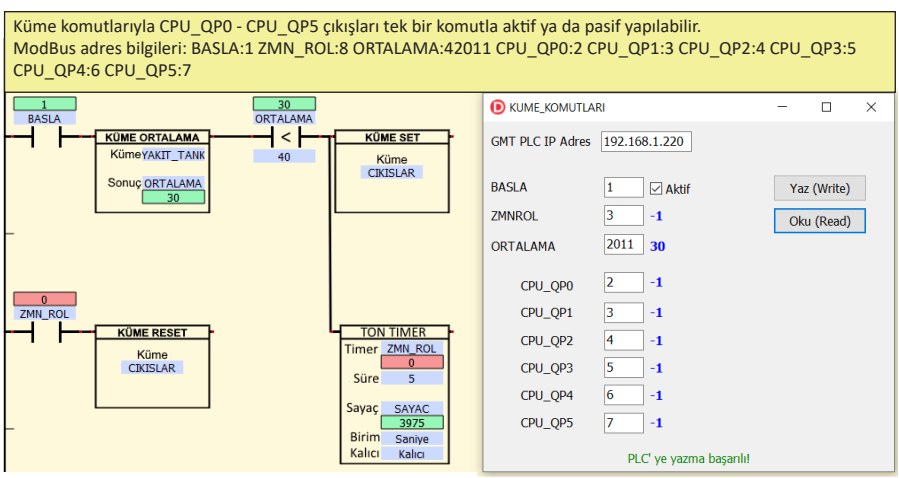
Görsel 2.12: Küme komutları tasarım ekranı



Görsel 2.13: Küme komutları Delphi tasarım arayüzü

```
procedure TForm1.Button2Click(Sender: TObject);
 var
   Data: array[0..10] of Word;
   SONUC: Boolean; okuma_sayisi,i: Integer;
 begin
     okuma_sayisi :=2;
     IdModBusClient1.Host := edit1.Text;
      if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit4.Text), okuma_sayisi, Data) then
     begin
       for i := 0 to (okuma_sayisi - 1) do
       begin
       label7.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
       label7.Caption:=FloatToStr(HexToIEEE754(label7.Caption));
       end;
     end
     else
       ShowMessage('PLC''den okuma başarısız!');
    if IdModBusClient1.ReadCoil(StrToInt(Edit3.Text), SONUC) then
     Label4.Caption := BoolToStr(SONUC)
    else
     Label5.Caption := 'PLC''den okuma başarısız!';
      if IdModBusClient1.ReadCoil(StrToInt(Edit5.Text), SONUC) then
     Label9.Caption := BoolToStr(SONUC)
    else
     Label5.Caption := 'PLC''den okuma başarısız!';
     if IdModBusClient1.ReadCoil(StrToInt(Edit6.Text), SONUC) then
     Label11.Caption := BoolToStr(SONUC)
    else
     Label5.Caption := 'PLC''den okuma başarısız!';
     if IdModBusClient1.ReadCoil(StrToInt(Edit7.Text), SONUC) then
     Label13.Caption := BoolToStr(SONUC)
    else
     Label5.Caption := 'PLC''den okuma başarısız!';
     if IdModBusClient1.ReadCoil(StrToInt(Edit8.Text), SONUC) then
     Label15.Caption := BoolToStr(SONUC)
    else
     Label5.Caption := 'PLC''den okuma başarısız!';
     if IdModBusClient1.ReadCoil(StrToInt(Edit9.Text), SONUC) then
     Label17.Caption := BoolToStr(SONUC)
    else
     Label5.Caption := 'PLC''den okuma başarısız!';
    if IdModBusClient1.ReadCoil(StrToInt(Edit10.Text), SONUC) then
     Label19.Caption := BoolToStr(SONUC)
    else
     Label5.Caption := 'PLC''den okuma başarısız!';
 end;
procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

Görsel 2.14: Küme komutları program kodu



Görsel 2.15: Küme komutları ladder diyagramı

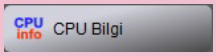
2.3. SİSTEM KOMUTLARI

Bu komutların tamamı GLC serisinin PLC modellerinde kullanılır.

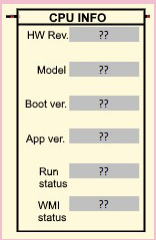
CPU Bilgi

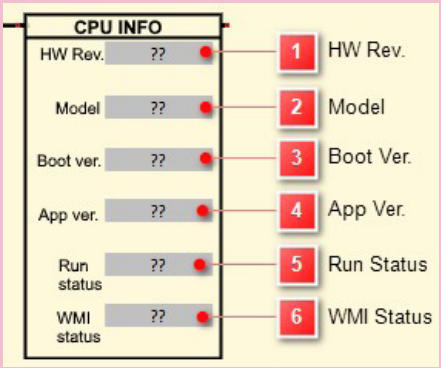
Kısa adı : CB

Kısa yol no. : Enter > 122

İkonu : 

Operand tipleri : Integer

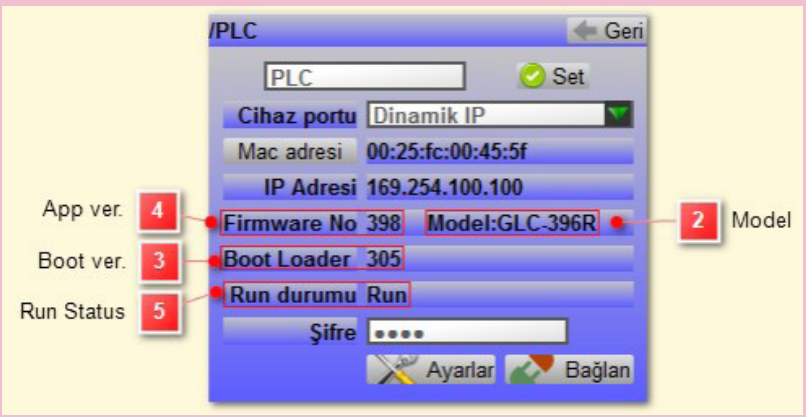


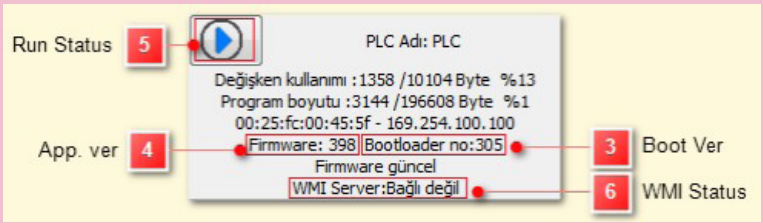


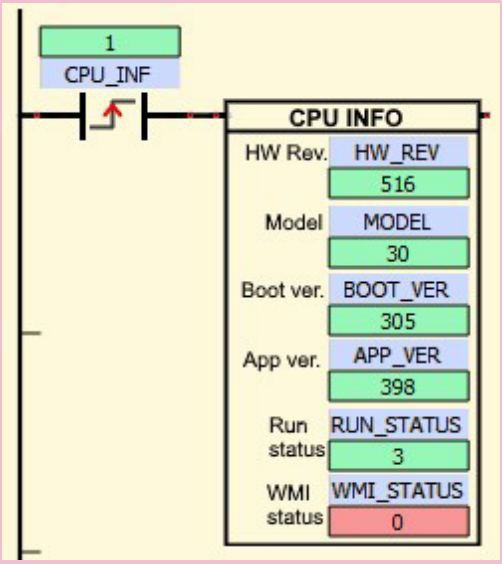
Çalışması :

1	**HW Rev: Hardware Revizyon** numarasının kaydedildiği operand adı yazılır.
2	**Model: PLC Model** numarasının kaydedildiği operand adı yazılır.
3	**Boot Ver. : PLC Boot Versiyon** numarasının kaydedildiği operand adı yazılır.
4	**App Ver. : PLC App Versiyon (Firmware No.)** numarasının kaydedildiği operand adı yazılır.
5	**Run Status: PLC Run** durumunun kaydedildiği operand adı yazılır.
6	**WMI Status:** PLC'nin **WMI (Windows Management Instrumentation) Server** bağlantı durumunu gösteren operand adı yazılır.

Örnek uygulama :







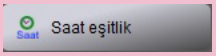
PLC'nin haberleşebildiği diğer PLC CPU bilgilerini alan komuttur. Bu uygulamada komut aktif edildiğinde, PLC CPU'ya ait bilgiler tanımlanan operandlara kaydedilir.

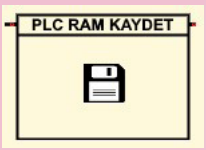
PLC RAM Kaydet

Kısa adı : RK

Kısa yol no. : Enter > 127

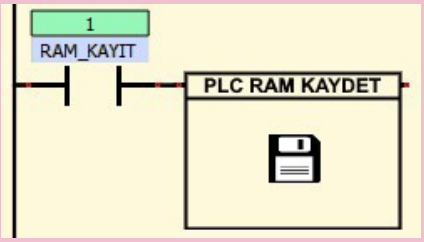
İkonu :





Açıklama : Komutun aktif olmasıyla **Operandlar** listesinde kalıcı olarak belirlenen operandlar, PLC'nin RAM belleğine kaydedilir.

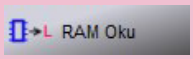
Örnek uygulama :

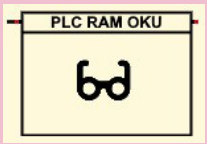


PLC RAM Oku

Kısa adı : RO

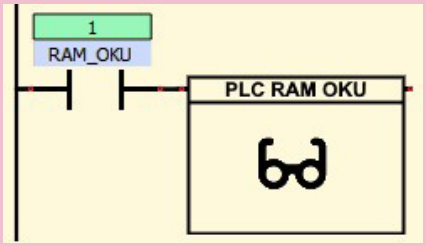
Kısa yol no. : Enter > 128

İkonu : 

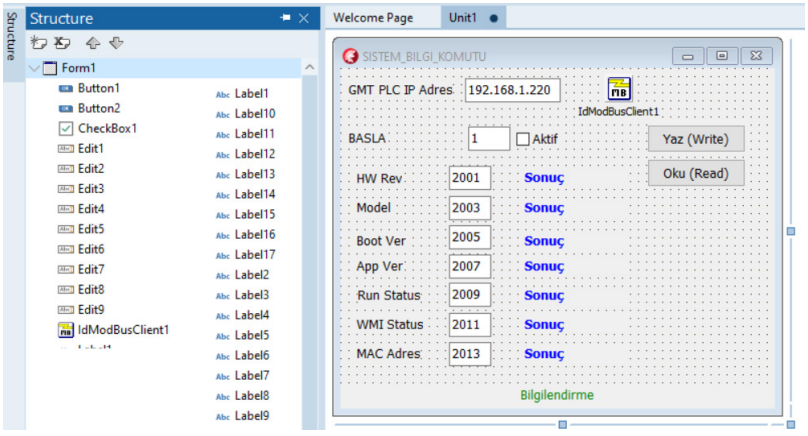


Açıklama : Komutun aktif olmasıyla daha önce PLC'nin RAM hafızasına kayıt edilen operand değerleri, tekrar operandlar listesine geri yüklenir.

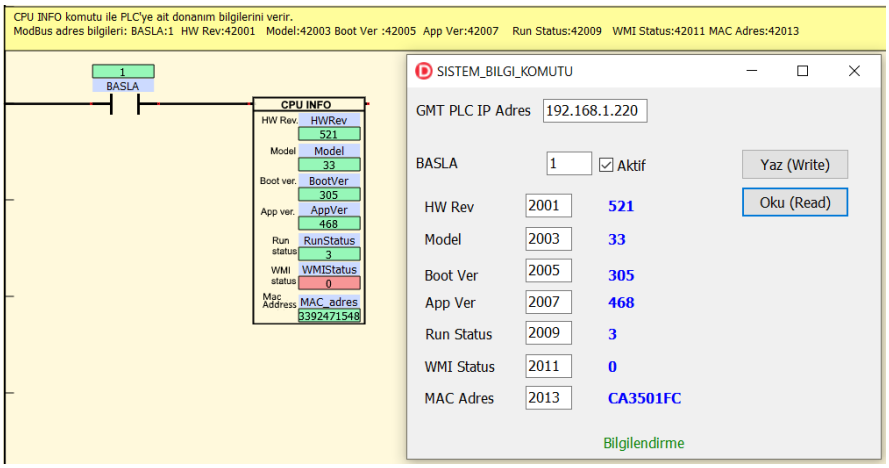
Örnek uygulama :



2.6. UYGULAMA: CPU Bilgi Komutu



Görsel 2.16: CPU bilgi komutu Delphi tasarım arayüzü



Görsel 2.17: CPU bilgi komutu ladder diyagramı

```
procedure TForm1.Button2Click(Sender: TObject);
var
  Data: array[0..10] of Word;
  SONUC: Boolean; okuma_sayisi,i: Integer;
begin
    okuma_sayisi :=2;
    IdModBusClient1.Host := edit1.Text;
     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit3.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
       begin
       label3.Caption:=IntToStr(Data[0]);
       end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');

     if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit4.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
       begin
       label4.Caption:=IntTostr(Data[0]);
       end;
    end
    else

  ShowMessage('PLC''den okuma başarısız!');
 if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit5.Text), okuma_sayisi, Data) then
begin
  for i := 0 to (okuma_sayisi - 1) do
   begin
   label8.Caption:=IntTostr(Data[0]);
   end;
end
else
  ShowMessage('PLC''den okuma başarısız!');
   if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit6.Text), okuma_sayisi, Data) then
begin
  for i := 0 to (okuma_sayisi - 1) do
   begin
   label10.Caption:=IntToStr(Data[0]);
   end;
end
else
  ShowMessage('PLC''den okuma başarısız!');
    if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit7.Text), okuma_sayisi, Data) then
begin
  for i := 0 to (okuma_sayisi - 1) do
   begin
   label12.Caption:=IntToStr(Data[0]);
   end;

    end
    else
      ShowMessage('PLC''den okuma başarısız!');
    if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit8.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
       begin
       label14.Caption:=IntTostr(Data[0]);

       end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');
    if IdModBusClient1.ReadHoldingRegisters(StrToInt(edit9.Text), okuma_sayisi, Data) then
    begin
      for i := 0 to (okuma_sayisi - 1) do
       begin
       label16.Caption:=IntToHex(Data[1])+IntToHex(Data[0]);
       end;
    end
    else
      ShowMessage('PLC''den okuma başarısız!');
end;
procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
    if IdModBusClient1.WriteCoil(StrToInt(Edit2.Text), CheckBox1.Checked) then
    Label5.Caption := 'PLC'' ye yazma başarılı!'
  else
    Label5.Caption := 'PLC''ye yazma başarısız!';
end;
```

Görsel 2.18: CPU bilgi komutu program kodu

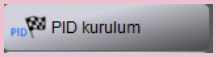
2.4. PID

Bu bölümde açıklanan komutların tamamı GLC ve GSR serisinin PLC modellerinde kullanılır. **GMTSuite Editör** programında **PID Blokları** ile kapalı çevrim kumandalı otomasyon projesi hazırlanır.

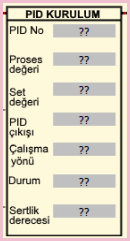
PID Kurulum

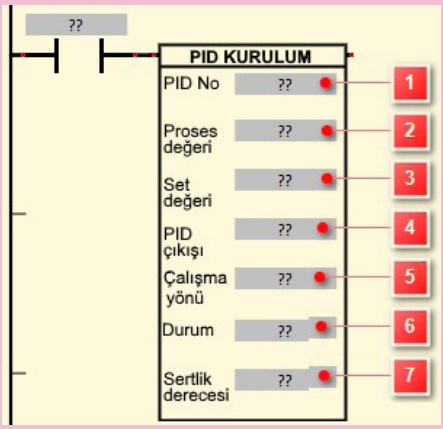
Kısa adı : PU

Kısa yol no. : Enter > 152

İkonu : 

Operand tipleri : Word, double word, integer, real.





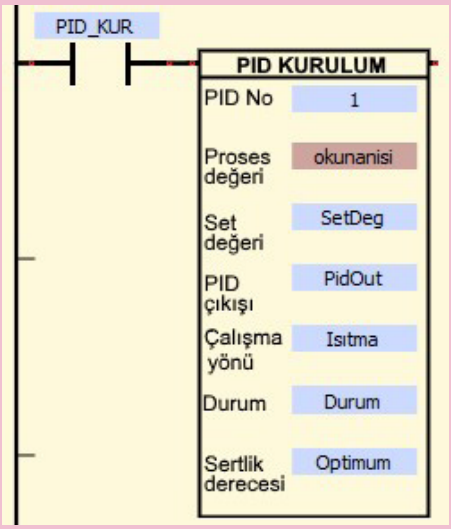
Çalışması :

1	**PID No.:** Kurulumu yapılan **PID No** yazılır.
2	**Proses Değeri:** Sıcaklık sensöründen elde edilen değerdir.
3	**Set Değeri:** PID için istenilen sıcaklık set değeridir. Set değeri sıcaklık değeri gibi x10 alınır.
4	**PID Çıkışı:** Seçilen PID'e ait çıkış gösterge operand adı yazılır.
5	**Çalışma Yönü:** Proses yönü Isıtma/Soğutma olarak seçilir.
6	**Durum:** Auto-tune safhası buradan izlenir. **1.** Tune başlar. **2.** PID değeri hesaplanır. **3.** Hesaplama başarıyla tamamlanır. **4.** Proses değeri (okunan değer) set değerinin %40 altına düşünceye kadar çıkış vermeden beklemeye geçer. Şartlar sağlandığında tune otomatik olarak başlar. **5. 6. 7. 8. 9. 10.** Hesaplama hatası, tekrar denenmelidir. **11.** Flash yazma hatası, tekrar denenmelidir. **12.** Log alma esnasında hafıza aşımı; auto-tune sonucunda PID katsayıları hesaplanmış ise daha iyi bir sonuç için PID auto-tune tablosundan, periyot (ms) değeri yükseltilir. Eğer PID katsayılarını hesaplayamadıysa yükseltilmeli ve auto-tune tekrar edilmelidir.

7	**Sertlik Derecesi:** PID katsayıları belirlenirken fonksiyonun ne kadar agresif çalışacağı belirlenir. • Optimum (Set değerine daha hızlı yaklaşır ve çok az limit aşımı yapabilir.) • Agresif (Set değerine çok hızlı yaklaşır, limit aşımı yapar ve toparlar.) • Yumuşak (Set değerine yavaş yaklaşır ve limit aşımı yapmaz.) • Kullanıcının, eğer sertlik dereceleri arasında geçiş yapmak isterse tekrar auto-tune yapmasına gerek yoktur. Yalnızca sertlik derecesini değiştirmesi yeterlidir.

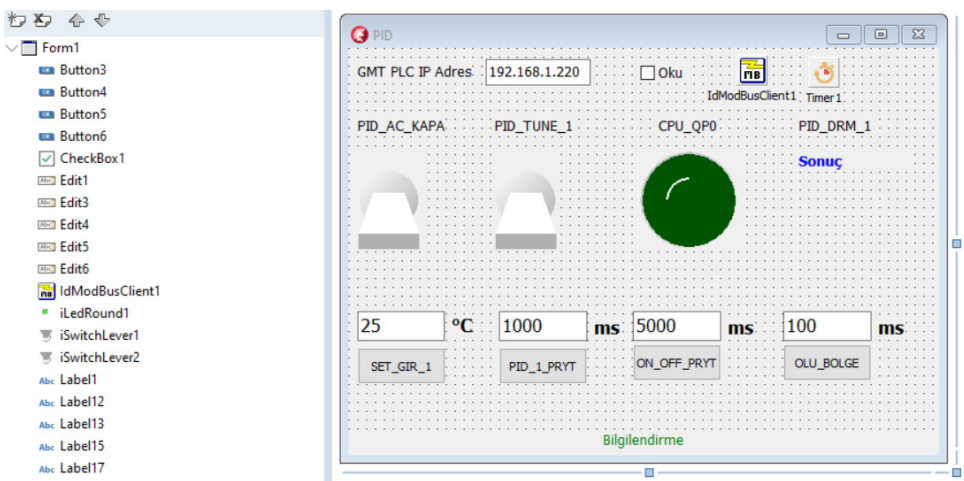
PID Kurulum komutu diğer PID komutlarından önce kullanılır.

Örnek uygulama :

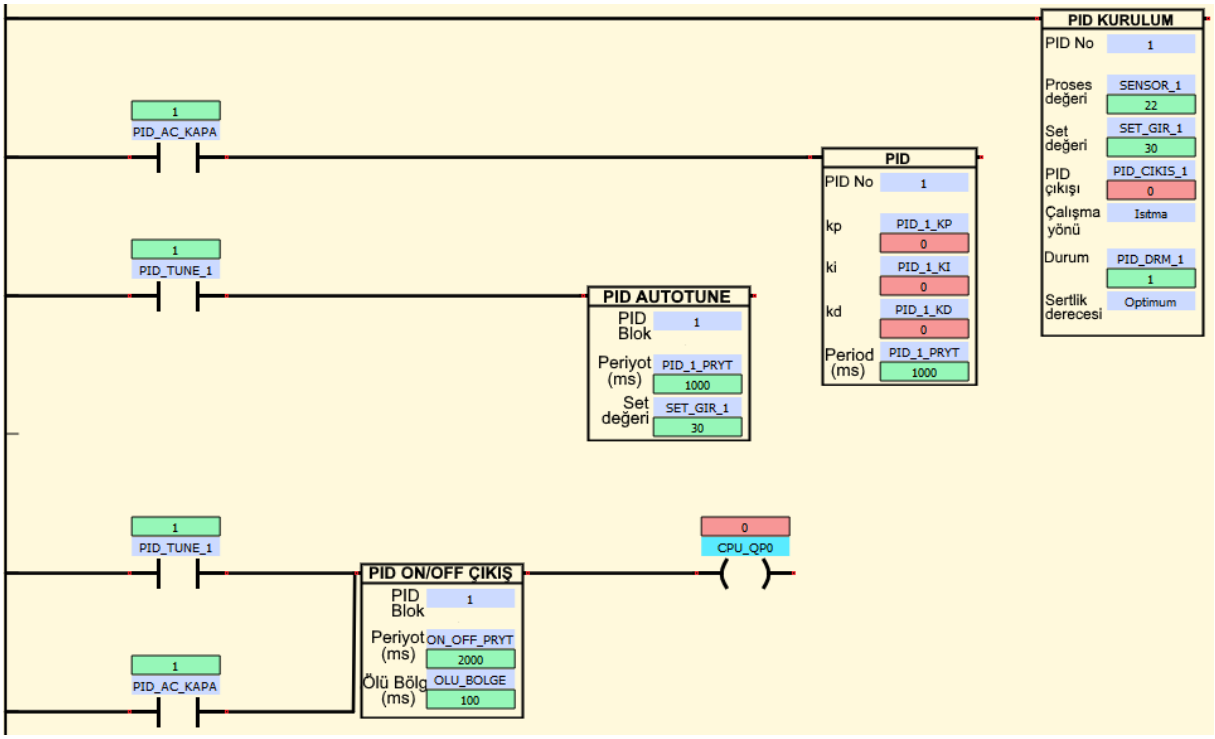


PID_KUR kontağının aktif olmasıyla PID kurulum fonksiyonu çalışır. Bu uygulamada **PID No** olarak **1** belirlenir. Diğer operand isimleri ve değerleri yazılır.

2.7. UYGULAMA: PID

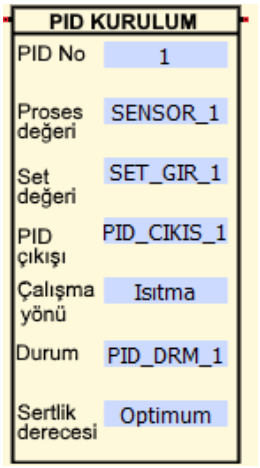


Görsel 2.19: PID Delphitasarım arayüzü



Görsel 2.20: PID ladder diyagramı

MODBUS Adres Bilgileri
SET_GIR_1: 42001
PID_1_PRYT: 42003
PID_DRM_1: 42005
ON_OFF_PRYT:40001
OLU_BOLGE: 40002
PID_AC_KAPA: 1
PID_TUNE_1: 2
CPU_QP0: 3



PID no: Kurulumu yapılan PID numarası yazılır. Birden çok PID olduğundan PID ile ilgili komutların birbirlerini tanımalarını sağlar.

Proses değeri: PID işlemi sırasında kullanılan ölçümün değeridir. Sensörden alınan bilgidir.

Set değeri: PID işlemi sırasında kullanılan set değeridir.

PID çıkışı: PID bloğunda 0-1000 aralığında hesaplanan PID çıkışını burada tanımlı etiket içinde verir.

Çalışma yönü: PID işleminin yönü ısıtma ya da soğutma olarak ayarlanabilir.

Durum: PID işlemi çalışma adımlarının gösterildiği bölümdür.

1 ↦ TUNE başlat.

2 ↦ PID değeri hesaplanıyor.

3 ↦ Hesaplama başarıyla tamamlandı.

4 ↦ Proses değeri (okunan değer) set değerinin %40 altına düşünceye kadar çıkış vermeden beklemeye geçer.

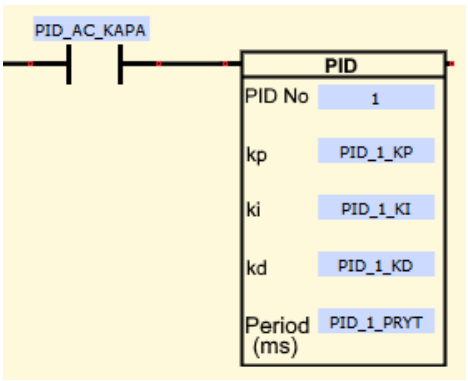
Şartlar sağlandığında TUNE otomatik olarak başlar.

5, 6, 7, 8, 9 ↦ Hesaplama hatası, tekrar denenir.

Sertlik derecesi: PID katsayıları belirlenirken fonksiyonun ne kadar agresif çalıştığı belirlenir.

- **Optimum:** Set değerine hızlı yaklaşır ve çok az limit aşımı yapabilir.
- **Agresif:** Set değerine çok hızlı yaklaşır, limit aşımı yapar ve toparlar.
- **Yumuşak:** Set değerine yavaş yaklaşır ve limit aşımı yapmaz.

PID bloğu **Normalde Açık** kontakla bağlanır.



PID no: PID numarası yazılır. PID KURULUM komutundaki numarayla aynı olmalıdır.

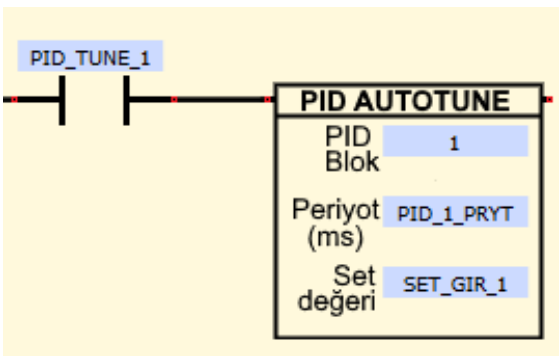
kp: Oransal kazanç katsayısı

ki: İntegral kazanç katsayısı

kd: Türev kazanç katsayısı

Çevrim süresi (ms): PID fonksiyon çevrim süresi milisaniye (ms) cinsinden yazılır. PID auto-tune bloğunda yer alan periyot ile aynı kullanıma sahiptir. PID auto-tune bloğu içerisine yazılan değer PID bloğuna da aktarılır.

PID AUTOTUNE bloğu **Normalde Açık** kontakla bağlanır.

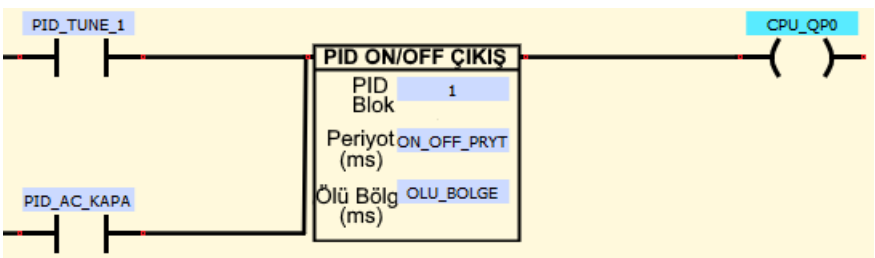


PID Blok: PID numarası yazılır.

Periyot (ms): Milisaniye cinsinden periyot bilgisi yazılır ya da PID komut bloğundaki PID_1_PRYT etiketiyle alınır.

Set değeri: PID KURULUM bloğu içerisindeki SET_GIR_1 değeri alınır.

PID ON/OFF ÇIKIŞ bloğu PID AUTOTUNE bloğuna ait Normalde Açık kontak ve PID hesap bloğunda kullanılan **Normalde Açık** kontağıyla bağlanır.



Bu kontaklardan **PD_TUNE_1**, **PID AUTOTUNE** bloğunun çalıştırılmasıyla aktif olması istenildiğinde kullanılır. **PID_AC_KAPA** kontağı ise manuel PID işleminde kullanılır. PID ON/OFF çıkışı da PID hesaplama bloğunun aktif edilmesiyle çalıştırılır.

Çıkış tipi analog olarak kullanılırsa **PID KURULUM** bloğunda yer alan PID çıkışı, analog çıkışa yönlendirilir.

Periyot (ms): PID ON/OFF bloğu içindeki periyot(ms), hesaplanan on ve off sürelerinin toplamıdır.

Ölü bölge (ms): PID ON/OFF bloğu çıkışının aktif olması için hesaplanan minimum ON süresidir.

Örnek hesaplama

Periyot **5000 ms**, ölü bölge 100ms ayarlandığı bir durumda;

Hesaplanan PID çıkışının **250 ms** olması için (PID çıkışı skala değeri 0…1000)

PID çıkışı ON süresi **= 5000 * 250 / 1000 = 1250 ms**

PID çıkışı OFF kalma süresi **= 5000 - 1250 : 3750 ms**

1250 ms değeri ölü bölge değeri olan **100 ms**'den büyük olduğu için PID ON/OFF çıkış bloğu çıkış verir.

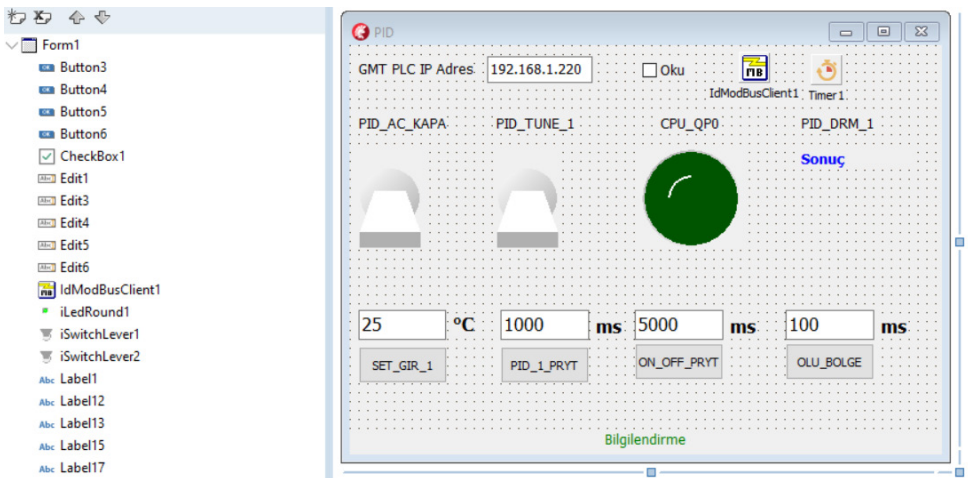
PID çıkışı ON süresi 100ms'den küçük olduğu durumlarda PID ON/OFF çıkış bloğu çıkış vermez.

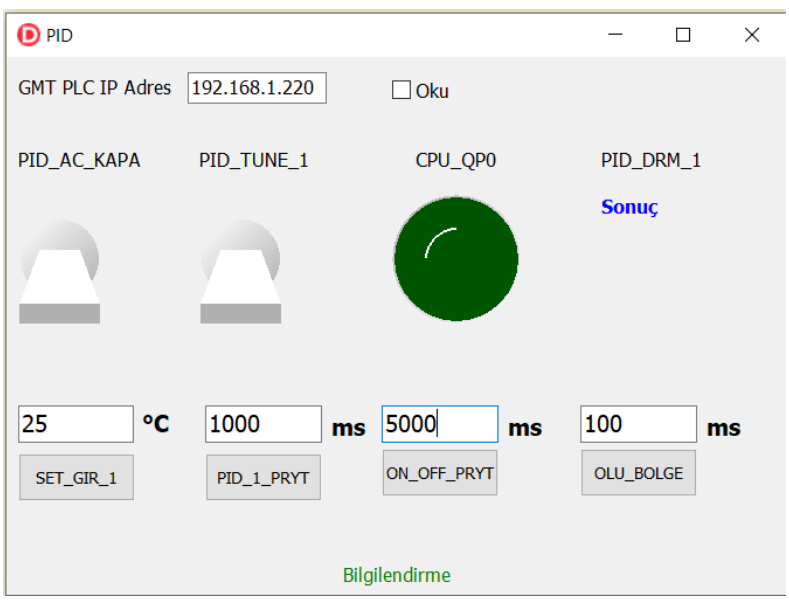
PID AUTOTUNE KULLANIM ADIMLARI

1. PID komutları sırasıyla ladder diyagramına eklenir ve yukarıda belirtilen kontaklar bloklarla ilişkilendirilir (PID KURULUM, PID, PID AUTOTUNE, PID ON/OFF ÇIKIŞ).
2. Kullanılan PID numarası ilgili kutucuğa girilir.
3. Komut blokları içerisine ilgili alanlar için registerların isimlendirmeleri yapılır.
4. PID işlemi için kullanılan çalışma belirlenir (ısıtma, soğutma).
5. Sertlik derecesi belirlenir.
6. PID hesaplama bloğu için periyot değeri girişi yapılır. Bu değer PID AUTOTUNE bloğunda bulunan periyot değeriyle aynı olup istenildiğinde PID AUTOTUNE bloğuna yazılan periyot değeri bu bloğa gönderilir.
7. PID AUTOTUNE bloğu içerisine belirlenen değerlerin girişi yapılır. Set değeri girişi, bu bloktan yapıldığında bloğun aktif edilmesiyle PID KURULUM bloğuna gönderilir.
8. PID ON/OFF çıkış bloğu, periyot ve ölü bölge girişleri yapılır.
9. Yukarıda girişi yapılması istenilen değerler sonrasında, set değeri ve periyot değeri PID AUTOTUNE bloğuna girilir.
10. PID hesaplama bloğu çalıştırılır.
11. PID AUTOTUNE bloğu çalıştırılır.
12. PID AUTOTUNE bloğunun çalışması ile PID ON/OFF bloğu çıkış vermeye başlar.
13. PID kurulum bloğunda PID çıkışı değişimi izlenebilir.
14. PID başlatılmasıyla PID durumu registerı içerisinde sırası ile **1** sonrasında hesaplama işleminin başlaması ile **2** hesaplamanın sonuçlanması ile **3** değeri gözlemlenir.
15. Bu durumla PID hesap bloğu içerisinde kp ,ki ve kd değerleri bulunur. PID AUTOTUNE işlemi başarıyla sonuçlanır.

PID MANUEL KULLANIM ADIMLARI

1. PID komutları sırasıyla ladder diyagramına eklenir ve yukarıda belirtilen kontaklar bloklar ile ilişkilendirilir (PID KURULUM, PID, PID ON/OFF ÇIKIŞ).
2. Kullanılan PID numarası ilgili kutucuğa girilir.
3. Komut blokları içerisine ilgili alanlar için registerların isimlendirmeleri yapılır.
4. PID işlemi için kullanılan çalışma belirlenir (ısıtma, soğutma).
5. Sertlik derecesi belirlenir.
6. PID hesaplama bloğu için periyot değeri girişi yapılır.
7. PID ON/OFF çıkış bloğu, periyot ve ölü bölge girişleri yapılır.
8. PID hesaplama bloğu önünde yer alan normalde açık kontak açık konumdayken kp, ki ve kd değerleri için manuel olarak giriş yapılır.
9. PID hesaplama bloğu çalıştırılır. Bloğun önünde yer alan normalde açık kontak kapatılır.
10. PID hesaplama bloğunun çalıştırılmasıyla PID ON/OFF bloğu önünde yer alan aynı isimli kontak ve PID ON/OFF bloğu çıkışı aktif hâle gelir.
11. PID KURULUM bloğunda PID çıkışı değişimi izlenebilir.





Görsel 2.21: PID Delphi tasarım arayüzü

```
procedure TForm1.iSwitchLever1Change(Sender: TObject);
begin
IdModBusClient1.Host := Edit1.Text;
//PID_AC_KAPA Modbus adresi:1
if iSwitchLever1.Active then
IdModBusClient1.WriteCoil(1,True) else
IdModBusClient1.WriteCoil(1,False)
end;

procedure TForm1.iSwitchLever2Change(Sender: TObject);
begin
IdModBusClient1.Host := Edit1.Text;
//PIN_TUNE_1 Modbus adresi:2
if iSwitchLever2.Active then
IdModBusClient1.WriteCoil(2,True) else
IdModBusClient1.WriteCoil(2,False)
end;

procedure TForm1.Button3Click(Sender: TObject);
begin
IdModBusClient1.Host := Edit1.Text;
//SET değeri girişi için Modbus adresi: 42001-40000 =2001
IdModBusClient1.WriteRegister(2001,strtoint(edit3.Text))
end;

procedure TForm1.Button4Click(Sender: TObject);
begin
 IdModBusClient1.Host := Edit1.Text;
//PID hesaplama periyodu değer girişi için Modbus adresi: 42003-40000 =2003
IdModBusClient1.WriteRegister(2003,strtoint(edit4.Text))
end;

procedure TForm1.Button5Click(Sender: TObject);
begin
 IdModBusClient1.Host := Edit1.Text;
//PID ON/OFF periyodu değer girişi için Modbus adresi: 40001-40000 =1
IdModBusClient1.WriteRegister(1,strtoint(edit5.Text))
end;

procedure TForm1.Button6Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
//Ölü bölge değer girişi için Modbus adresi: 40002-40000 =2
IdModBusClient1.WriteRegister(2,strtoint(edit6.Text))
end;

procedure TForm1.CheckBox1Click(Sender: TObject);
begin
timer1.Enabled:=CheckBox1.Checked;
end;

procedure TForm1.Timer1Timer(Sender: TObject);
 var
   Data: array[0..10] of Word;
   CIKIS_DURUM: Boolean; okuma_sayisi,i: Integer;
 begin
 IdModBusClient1.Host := Edit1.Text;
 //CPU_QP0: 3
 IdModBusClient1.ReadCoil(3,CIKIS_DURUM);
 if BoolToStr(CIKIS_DURUM)='-1' then
 iLedRound1.Active:=True else iLedRound1.Active:=False;
 //PID_DRM_1 Modbus adresi:42005-40000:2005
      if IdModBusClient1.ReadHoldingRegisters(2005, okuma_sayisi, Data) then
     begin
       for i := 0 to (okuma_sayisi - 1) do
       begin
       label12.Caption:=IntToStr(Data[0]);
       end;
     end
end;
```

Görsel 2.21: PID program kodu

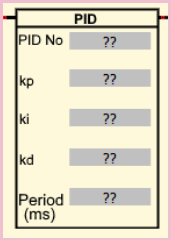
PID Fonksiyonu

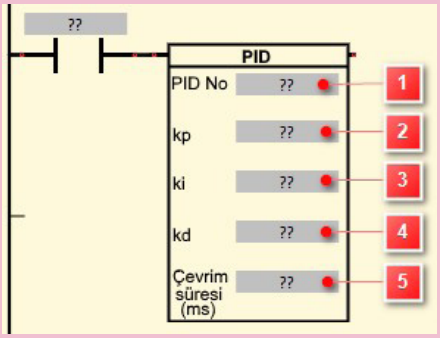
Kısa adı : PF

Kısa yol no. : Enter > 153

İkonu : fn(PID) PID fonksiyonu

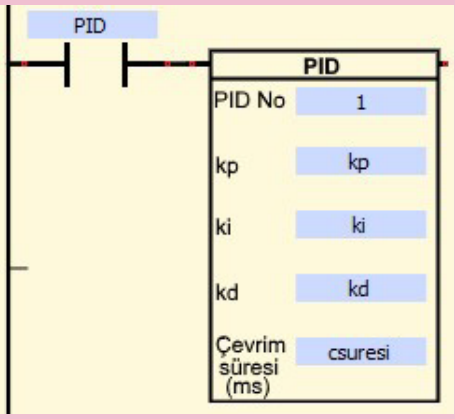
Operand tipleri : Word, double word, integer, real.





Çalışması :

1	**PID No.:** Çalıştırılacak olan **PID No.** yazılır.
2	**kp:** Fonksiyon kazanç katsayısı (oransal) yazılır.
3	**ki:** Fonksiyon katsayısı (integral) yazılır.
4	**kd:** Fonksiyon katsayısı (türev) yazılır.
5	**Çevrim Süresi (ms): PID** fonksiyon ms cinsinden fonksiyon çevrim süresi yazılır.
6	**WMI Status:** PLC'nin **WMI (Windows Management Instrumentation) Server** bağlantı durumunu gösteren operand adı yazılır.



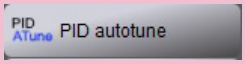
Örnek uygulama : **PID** komutunun aktif olmasıyla çevrim süresi içinde belirlenen katsayılarla hesaplanan **PID** fonksiyonu çalıştırılır.

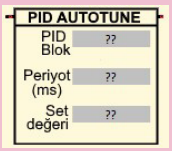
PID Auto - Tune

Kısa adı : PA

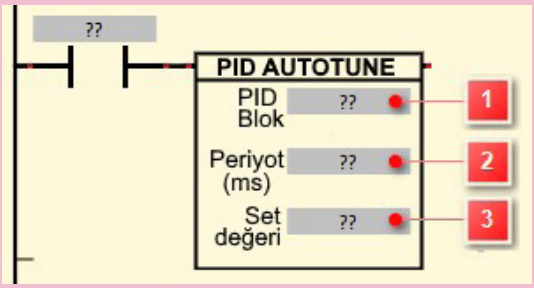
Kısa yol no. : Enter > 154

İkonu :



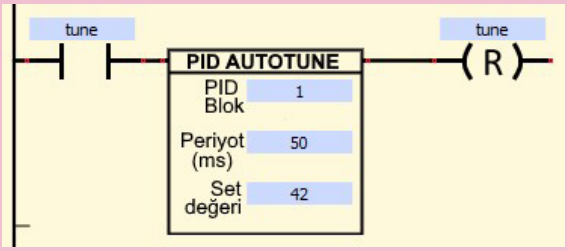


Operand tipleri : Word, double word, integer, real.



Çalışması :

1	**PID Blok:** Auto-tune işlemi yapılan **PID Block No.** yazılır.
2	**Periyot (ms):** Auto-tune esnasındaki örnekleme periyodudur. Bu değerler baz alınarak auto-tune yapılır.
3	**Set Değeri:** Auto-tune esnasında hedeflenen sıcaklık değeridir. Auto-tune hesaplaması bitim sıcaklığının set değeri ve o anki sıcaklık ortasında tamamlanır. Bu değerler baz alınarak auto-tune yapılır.



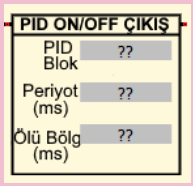
Örnek uygulama : **PID AUTOTUNE** fonksyionu çalıştırıldığında **Tune** işlemi başlar. Tune başladığında 50 ms periyotla örnekleme yapılır. 42 dereceye ulaşıncaya kadar **PID KURULUM** fonksiyonundaki **Durum** registerı 1 olur, sonra 2 olarak hesap işlemi yapar. Daha sonra 3 değerini tamamlayarak **PID** fonksiyonundaki katsayaları yükler.

PID On/Off Çıkış

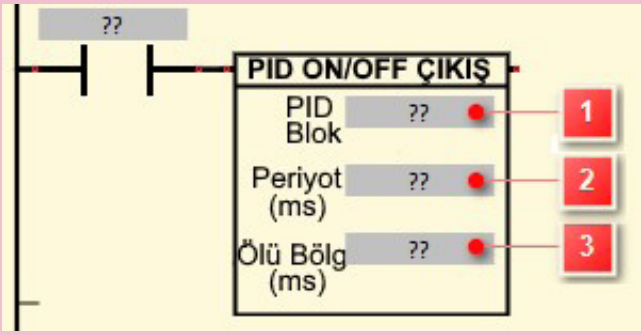
Kısa adı : PÇ

Kısa yol no. : Enter > 156

İkonu : PID On/Off çıkış

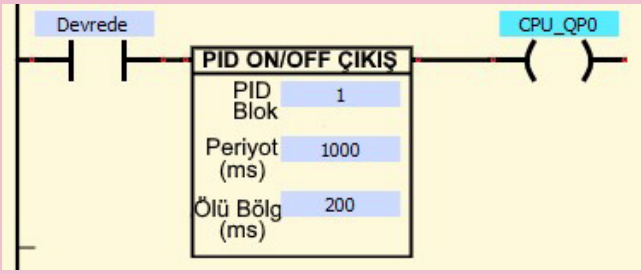


Operand tipleri : Word, double word, integer, real.

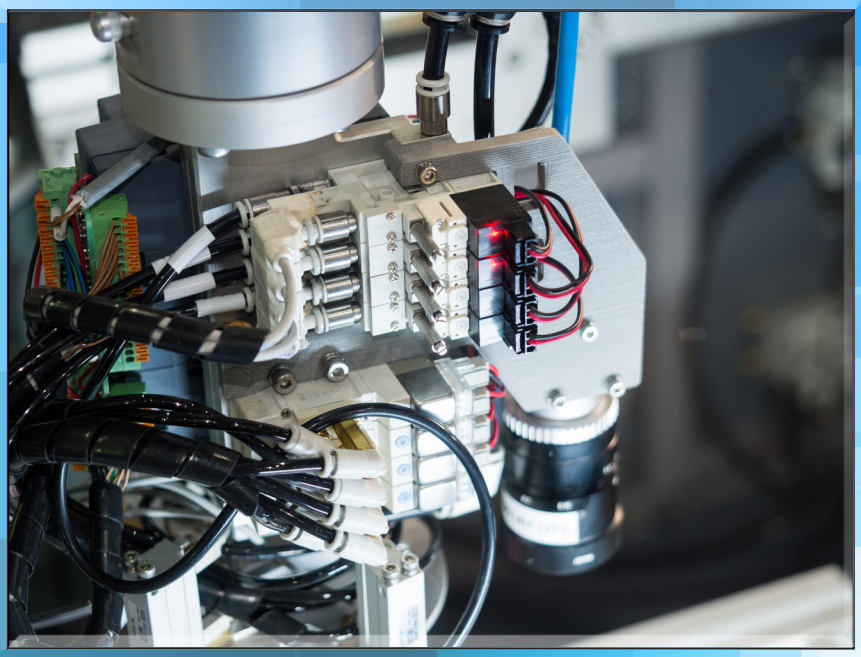


Çalışması :

1	**PID Blok:** Auto-tune işlemi yapılan **PID Block No.** yazılır.
2	**Periyot (ms.): PID Kontrol Çıkış** periyodudur. **PID** tablosundaki **PID** kontrol çevrim süresinden farklı olarak çıkışın çevrim süresidir. Örneğin PID kontrol çevrim süresi 250 ms iken bu periyot 1000 ms seçilirse PID kontrol hesaplaması her 250 ms'de bir değişirken, çıkış her 1000 ms'de hesaplanan PID kontrol çıkışına göre değişim gösterir.
3	**Ölü Bölge (ms): PID** çıkışının çıkış verebildiği eşik süresidir. Örneğin 200 ms ölü bölge seçilen bir kontrolde, kontaktlar 200 ms altındaki kontrol sürelerde çıkış vermez.



Örnek uygulama : **PID ON/OFF ÇIKIŞ** fonksiyonu çıkış tipi aç kapa şeklinde ise bu komut kullanılır. Analog çıkış kullanılırsa **PID KURULUM** fonksiyonundaki **PID Çıkışı** için belirlenen değer, analog çıkışa yönlendirilir.



3. ÖĞRENME BİRİMİ

MOTOR KONTROL UYGULAMALARI

KONULAR

3.1. DOĞRU AKIM, STEP VE SERVO MOTORLAR

3.2. ASENKRON MOTOR

3.3. OPERATÖR PANEL PROGRAMLAMA

NELER ÖĞRENECEKSİNİZ?

- Doğru akım motorları, servo ve step motorlar
- Asenkron motorlar
- Operatör panel programlama

TEMEL KAVRAMLAR

Doğru akım motoru, inverter, operatör paneli, step motor.

HAZIRLIK SORULARI

1. Operatör paneli ne amaçla kullanılabilir?
2. İnvertör işletmelerde kullanım alanlarını araştırınız?

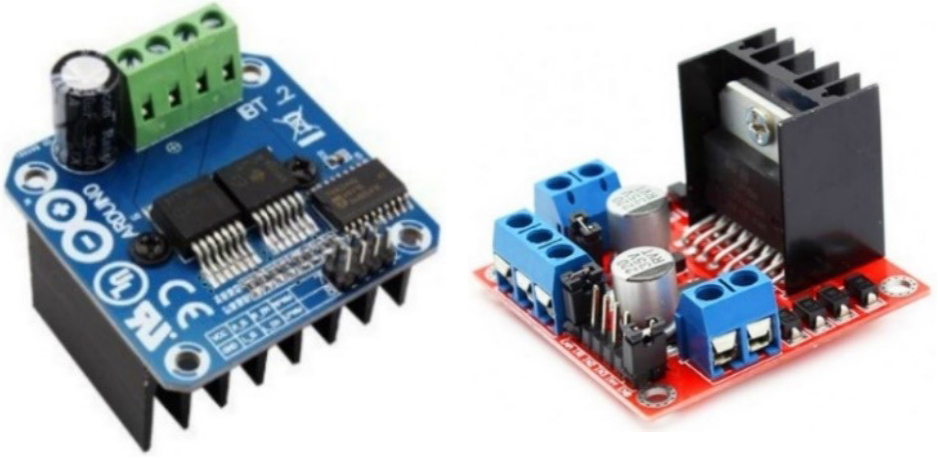
3. MOTOR KONTROL UYGULAMALARI

Doğru akım, step, servo ve asenkron en çok kullanılan motorlardır (Görsel 3.1). Fabrikalardaki sistem ve mekanizmalarda bu motorlar kullanılır.

3.1. DOĞRU AKIM, STEP VE SERVO MOTORLAR

PLC ile doğru akım, step ve servo motorlarının yön ve hız kontrolleri yapılır (Görsel 3.2). Doğru akım ile step motor sürücüleri, doğru akım ve step motorlarının hızını ve yönünü kontrol etmek için kullanılır. Uygulamada kullanılan PLC'nin konnektör uçlarındaki sinyaller; dc, step ve servo motorlarını kontrol eden sürücülerin her birinde kullanılır.

Sürücülerde pals, yön ve enable uçları vardır. PLC ve görsel programlama kodlamaları aynıdır. Motor uygulamalarında motorun kaç mm veya kaç cm gideceği bellidir. GMT, PLC'yi kullanarak motorları artımsal ve mutlak olmak üzere iki şekilde hareket ettirir. Artımsal, pals üretilerek motor hareket ettirilir. Örneğin motor 100 pals üretir ve durur. 100 palslik çalışma anında motor ne kadar ilerlerse ölçülür ve net ölçüm ortaya çıkar. Gerekirse pals rakamları ile test edilir ve istenilen motor hareket mesafesi ortaya çıkar. 100 palstan sonra 20 pals daha üretilirse 120 pals üretilir. Önceki rakamın üzerine ilave edilerek arka planda hesaplama yapılır. Bu hareket tekniği artımsal olarak tanımlanır. Hareket kontrol sekmesinden hareket oluşturma fonksiyonu çalışma alanına getirilir. Motor sürücünün enable ve yön pinleri için openadlar eklenir.

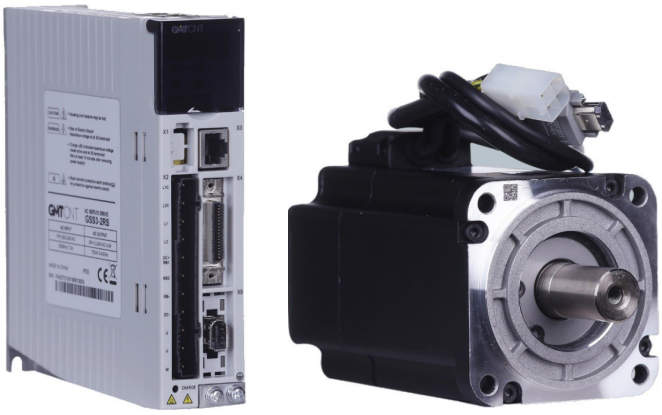


Görsel 3.1: Yüksek ve düşük güçlü doğru akım motor sürücüleri

Görsel 3.2: Step motor sürücü ve step motor

Step Motor 8 Kablo Bağlantısı

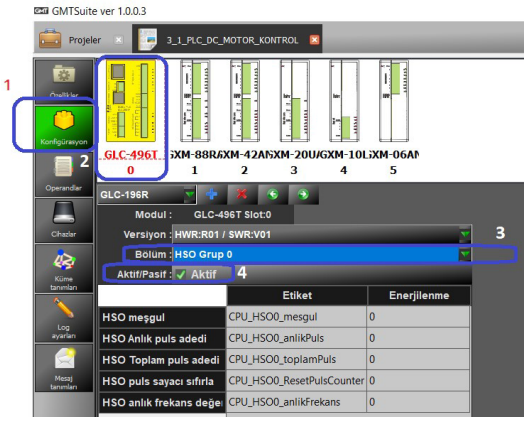
A+ = MAVİ-SARI. A- = KIRMIZI – YEŞİL. B+ = TURUNCU – KAHVE. B- = SİYAH - BEYAZ



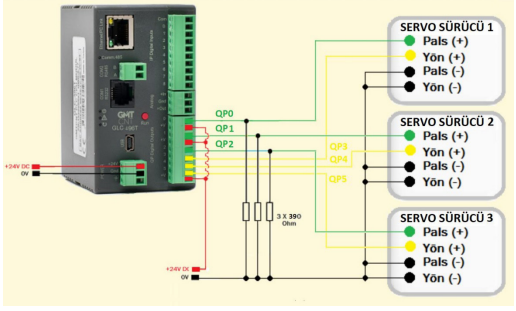
Görsel 3.3: Servo motor sürücü ve servo motor

GMTSuite editör programında konfigürasyon menüsünden ilgili kanal/lar aktif edilir. Üç adet hızlı çıkış kanalı vardır. 496T PLC modelinin 0,1 ve 2 digital çıkışları yüksek hızlı çıkış olarak kullanılır. 3,4 ve 5 numaralı digital çıkışlar ise yönlendirme için kullanılır. Yüksek hızlı uygulamalarda mutlaka 390 ohmluk direnç kullanılır (Görsel 3.3).

3.1. UYGULAMA: PLC ile Mesafeli Artımsal Motor Kontrol



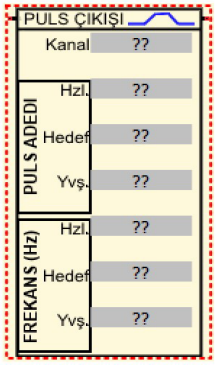
Görsel 3.4: PLC'nin hızlı çıkışını aktif etme işlemi



Görsel 3.5: GMT PLC'nin (model 496T) hızlı çıkış bağlantısı

İşlem Basamakları

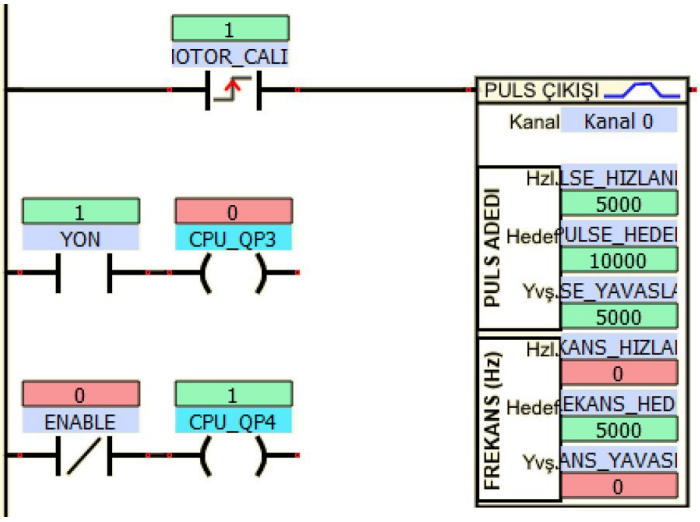
HSO Kanal seçimi (0, 1, 2)
Hedef frekansa kaç adımda ulaşılır?
Hedef frekansta iken kaç adım hareket edilir?
Hedef frekanstan kaç adımda inilir?
Hızlanma hangi frekansta başlar?
Hedef frekans kaç olur? (496T için max. : 400000)
Yavaşlama hangi frekansta biter?



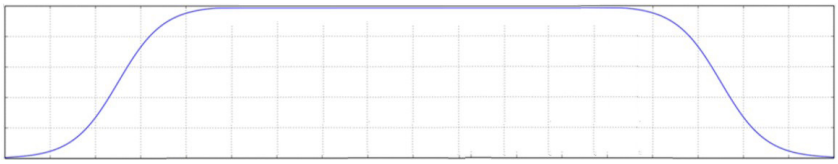
Görsel 3.6: Puls çıkışı komutu

PULSE_HIZLANMA	01	Word		☑	☑	40007
PULSE_HEDEF	02	Word		☑	☑	40006
PULSE_YAVASLAMA	03	Word		☑	☑	40005
FREKANS_HIZLANMA	04	Word		☑	☑	40004
FREKANS_HEDEF	05	Word		☑	☑	40003
FREKANS_YAVASLAMA	06	Word		☑	☑	40002
ENABLE	07	Bit		☑	☑	1
MOTOR_CALIS	08	Bit		☑	☑	5
YON	09	Bit		☑	☑	6
CPU_HSO0_anlikPuls	HSO Anlık puls adedi	Integer	0	☐	☑	42001
CPU_HSO0_mesgul	HSO meşgul	Bit	0	☐	☑	7
CPU_HSO0_toplamPuls	HSO Toplam puls adedi	Integer	0	☐	☑	42003

Görsel 3.7: Uygulamanın MODBUS adres bilgileri

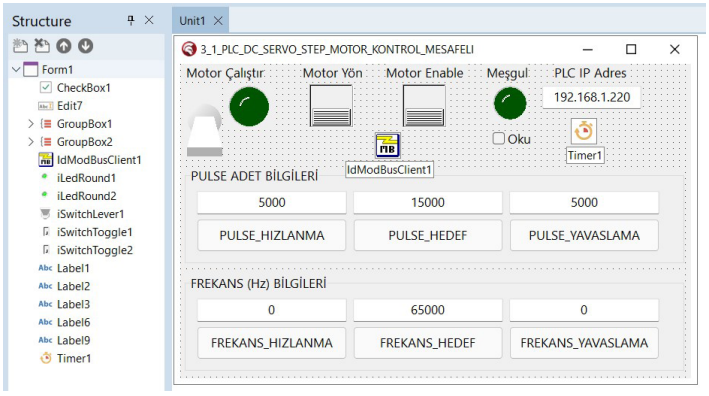


Görsel 3.8: Uygulamanın ladder program ekran görüntüsü



Görsel 3.9: Uygulamanın hızlanma, ilerleme ve yavaşlamasını gösteren grafik

Boş Delphi projesi oluşturulur ve boş hâliyle ilgili klasöre kaydedilir.

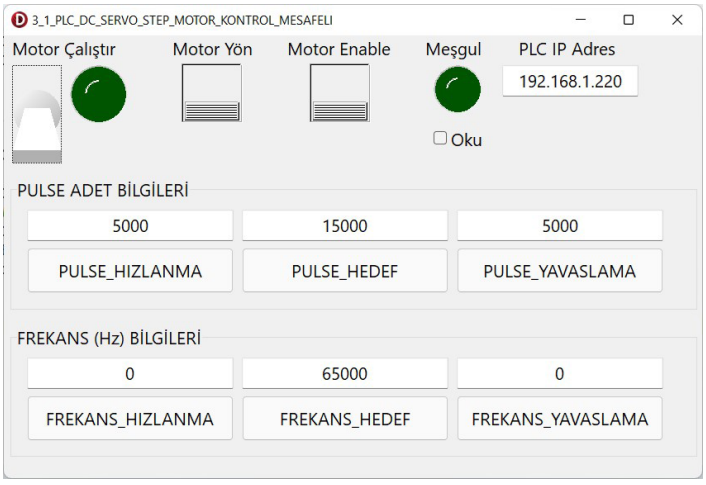


Görsel 3.10: Uygulamanın tasarım anı ekran görüntüsü

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit7.Text;
  // PULSE_HIZLANMA için ModBus : 40007 - 40000 = 7
  IdModBusClient1.WriteRegister(7, StrToInt(Edit1.Text));
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit7.Text;
  // PULSE_HEDEF için ModBus : 40006 - 40000 = 6
  IdModBusClient1.WriteRegister(6, StrToInt(Edit2.Text));
end;

procedure TForm1.Button3Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit7.Text;
  // PULSE_YAVASLAMA için ModBus : 40005 - 40000 = 5
  IdModBusClient1.WriteRegister(5, StrToInt(Edit3.Text));
end;
procedure TForm1.Button4Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit7.Text;
  // FREKANS_HIZLANMA için ModBus : 40004 - 40000 = 4
  IdModBusClient1.WriteRegister(4, StrToInt(Edit4.Text));
end;

procedure TForm1.Button5Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit7.Text;
  // FREKANS_HEDEF için ModBus : 40003 - 40000 = 3
  IdModBusClient1.WriteRegister(3, StrToInt(Edit5.Text));
end;

procedure TForm1.Button6Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit7.Text;
  // FREKANS_YAVASLAMA için ModBus : 40002 - 40000 = 2
  IdModBusClient1.WriteRegister(2, StrToInt(Edit6.Text));
end;
```

```
procedure TForm1.CheckBox1Click(Sender: TObject);
begin
  Timer1.Enabled := CheckBox1.Checked;
end;

procedure TForm1.iSwitchLever1Change(Sender: TObject);
begin
  iLedRound1.Active := iSwitchLever1.Active;
  IdModBusClient1.Host := Edit7.Text;
  if iSwitchLever1.Active then
  // Motor çalıştırma için ModBus = 5
  IdModBusClient1.WriteCoil(5, True) else
  IdModBusClient1.WriteCoil(5, False);
end;
procedure TForm1.iSwitchToggle1Change(Sender: TObject);
begin
  IdModBusClient1.Host := Edit7.Text;
  if iSwitchToggle1.Active then
  // Motor yönü için ModBus = 6
  IdModBusClient1.WriteCoil(6, True) else
  IdModBusClient1.WriteCoil(6, False);
end;
procedure TForm1.iSwitchToggle2Change(Sender: TObject);
begin
  IdModBusClient1.Host := Edit7.Text;
  if iSwitchToggle2.Active then
  // Motor enable için ModBus = 1
  IdModBusClient1.WriteCoil(1, True) else
  IdModBusClient1.WriteCoil(1, False);
end;

procedure TForm1.Timer1Timer(Sender: TObject);
var MESGUL_DURUM: Boolean;
  ANLIK_PULSE, TOPLAM_PULSE: word;
begin
  IdModBusClient1.Host := Edit7.Text;
  // Motor meşgul için ModBus = 7
  IdModBusClient1.ReadCoil(7, MESGUL_DURUM);
  if BoolToStr(MESGUL_DURUM) = '-1' then
  iLedRound2.Active := True else iLedRound2.Active := False;
end;
```

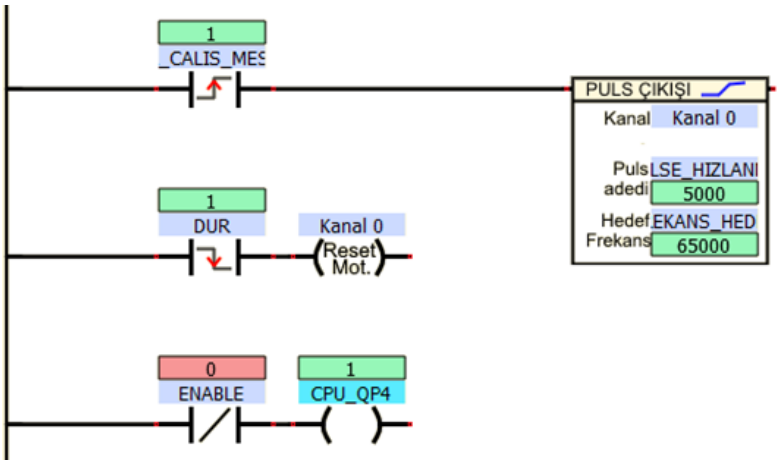
Görsel 3.11: Uygulamanın program kodları



Görsel 3.12: Uygulamanın çalışma anı ekran görüntüsü

3.2. UYGULAMA: PLC ile Mesafesiz Artımsal Motor Kontrol

Motorların gideceği mesafenin belli olmadığı ama motorların sürekli çalıştığı bir uygulamadır. İlk anda yavaşça çalışmaya başlar ve hedef frekansa 5.000 adımda ulaşır. Bu uygulamada pals çıkış ve kanal reset motor fonksiyonları kullanılır.

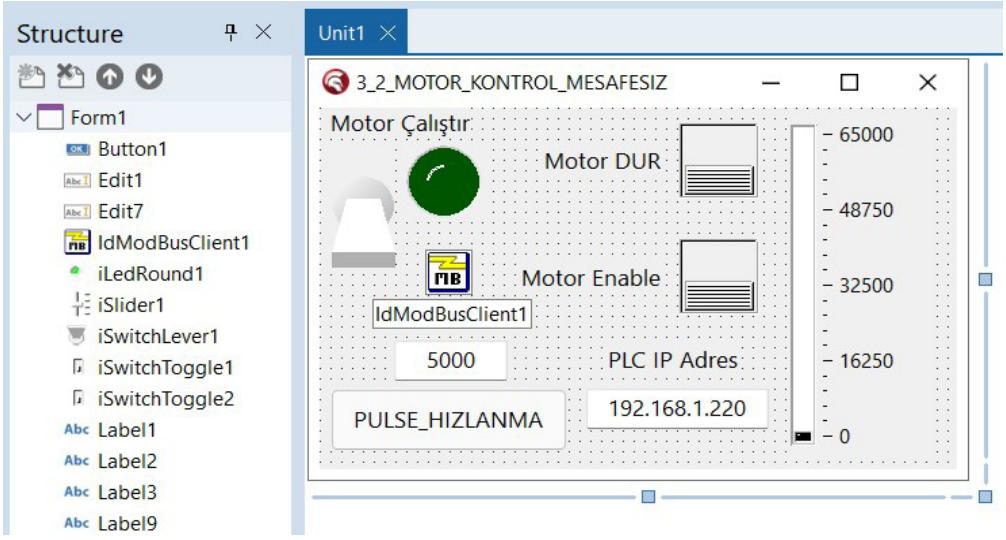


Görsel 3.13: Uygulamanın ladder program ekran görüntüsü

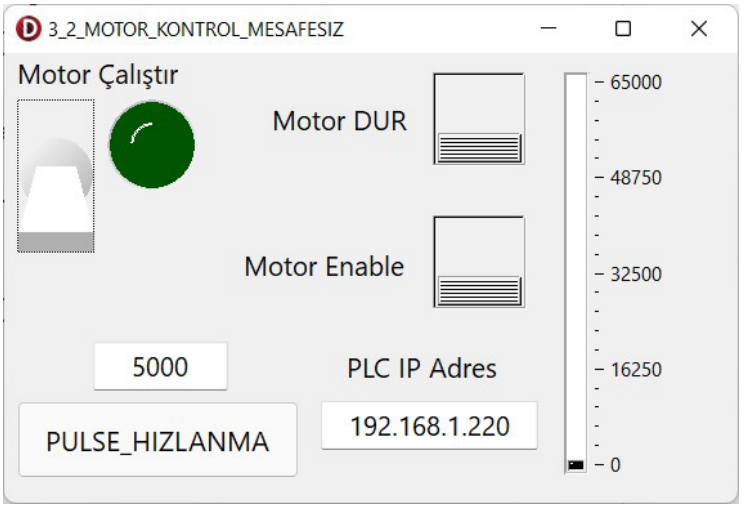
MOTOR_CALIS_MESA...	01	Bit	0	☐	☑	8
DUR	02	Bit		☑	☑	2
ENABLE	03	Bit		☑	☑	1
FREKANS_HIZLANMA	04	Word		☑	☑	40004
FREKANS_HEDEF	05	Word		☑	☑	40003

Görsel 3.14: Uygulamanın MODBUS adres bilgileri

Bu uygulamada motorların ne kadar ilerleyeceği belli olmadığından motorlar boş döner. Boş delphi projesi oluşturulur ve boş hâliyle ilgili klasöre kaydedilir.



Görsel 3.15: Uygulamanın tasarım anı ekran görüntüsü

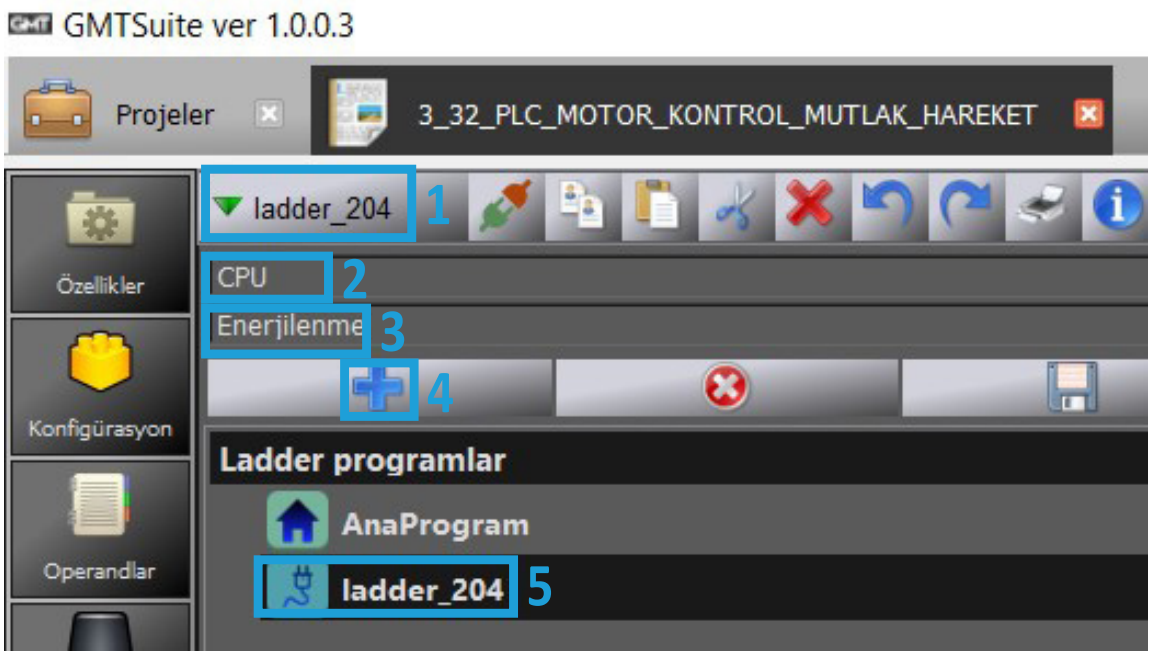


Görsel 3.16: Uygulamanın çalışma anı ekran görüntüsü

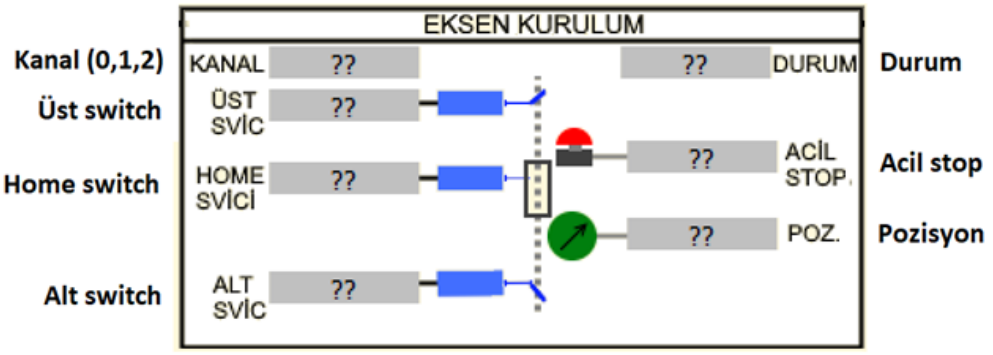
3.3. UYGULAMA: PLC ile Motorların Belli Bir Pozisyona Gitmesi

Motorların gideceği yere motorları göndermek için yapılan bir uygulamadır. **Mutlak (absolute) hareket** olarak da bilinir. Birden fazla ladder komutu (fonksiyon bloklar) kullanılır.

Öncelikle ana modülden CPU işaretlenir. Daha sonra enerjilenme sekmesinde artı işaretiyle eklenerek enerjilenme kısmı üretilir. Eksen kurulumu fonksiyonu sadece bir kere çalıştırılmak üzere enerjilenme kısmına eklenir. Enerjilenme sekmesindeki fonksiyonlar, PLC enerjilenince bir döngü boyunca bir kere çalışır ve bir daha çalışmaz. Eksen kurulum fonksiyonun enerjilenme kısmına konulması zorunludur.

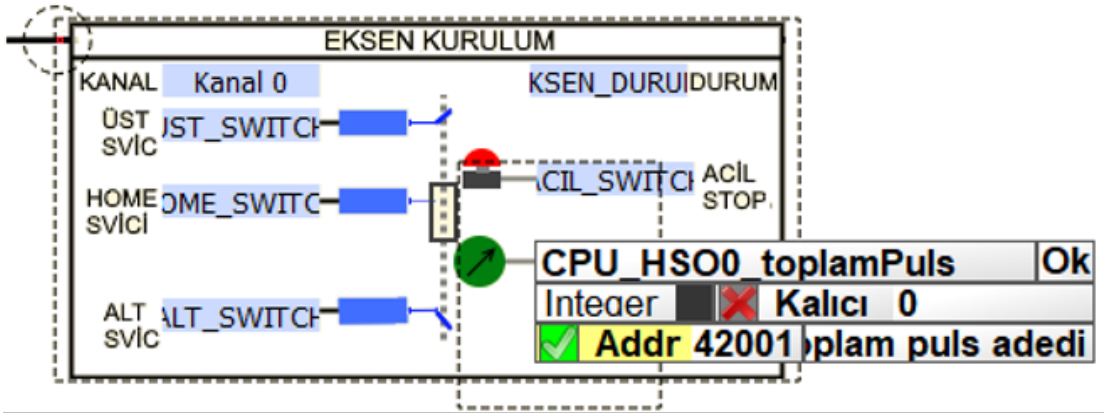


Görsel 3.17: Eksen kurulumunun çalışacağı konfigürasyon ekranı



Görsel 3.18: Eksen kurulum ladder komutu

Eksen kurulum fonksiyon bloğunda üst ve alt anahtar (switch) parametreleri, motor eksende hareket ederken eksenin ilk ve son noktalarındaki anahtarlara geldiğinde motor durur (Görsel 3.19). Koruma anahtarları pals kesilir. Home anahtar da eksenin hangi noktadan itibaren iş yaptırdığı bilgisini verir. Alt ve üst anahtarlara digital input bağlanır. Eğer digital input bağlanmazsa sıfır yazılır. Home anahtara ise digital input bağlanır. Durum parametresinde eksen hakkında bilgi vardır. Tanımlamayla görsel programlama dilinden o bilgiye erişilir. Acil stop anahtarı ise eksende hareket eden motoru anında durdurur. En önemli parametre olan pozisyon parametresi, home anahtara göre motorun bulunduğu mesafe bilgisini verir. Tanımlamayla görsel programlama diline bu bilgi gönderilir. PLC'nin yüksek hızlı çıkış bilgilerinden olan CPU_HSO0_toplamPuls bilgisi, eksen kurulum fonksiyonundaki pozisyon parametresine atanır (Görsel 3.20).

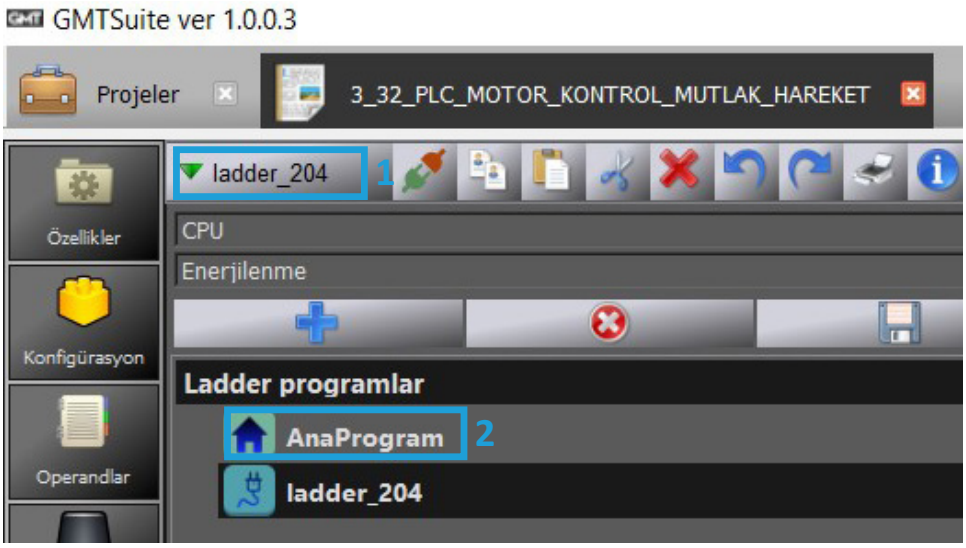


Görsel 3.19: Eksen kurulumunun MODBUS ayarları

UST_SWITCH		Bit	0	☐	☑	1
ALT_SWITCH		Bit	0	☐	☑	2
HOME_SWITCH		Bit	0	☐	☑	3
ACIL_SWITCH		Bit	0	☐	☑	4
EKSEN_DURUM		Word	0	☐	☑	40001
CPU_HSO0_toplamPuls	HSO Toplam puls adedi	Integer	0	☐	☑	42001

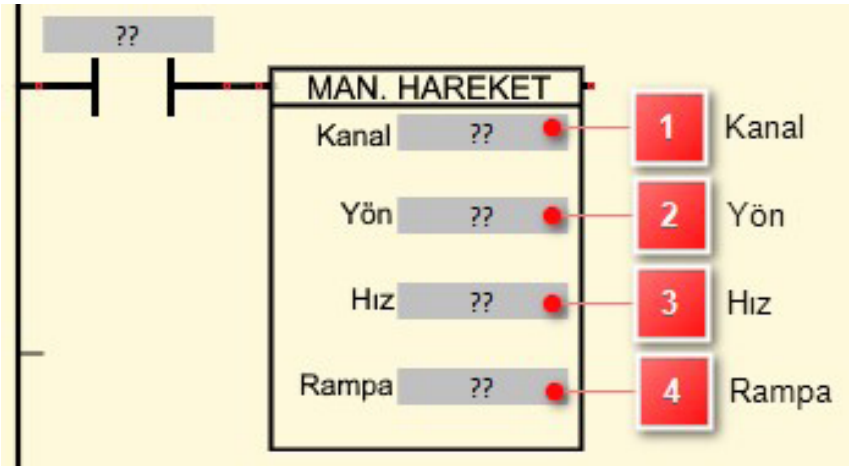
Görsel 3.20: Uygulamanın MODBUS adres bilgileri

Enerjilenme sekmesinden ana program sekmesine geçiş yapılır. Mutlak pozisyonlu motor hareketi manuel olarak yapılır.



Görsel 3.21: Eksen kurulumun ayar yerinden ana ekrana geçme arayüzü

Hareket kontrolü sekmesinden manuel hareket ladder komutu çalışma alanına getirilir. Manuel hareket komutu sinyal verdikçe motor belli bir yöne gider ve sinyal kesilince de motor durur. Hız ve rampa parametreleri frekans cinsindendir. Home gitme süresine farklı bir hız verilir.



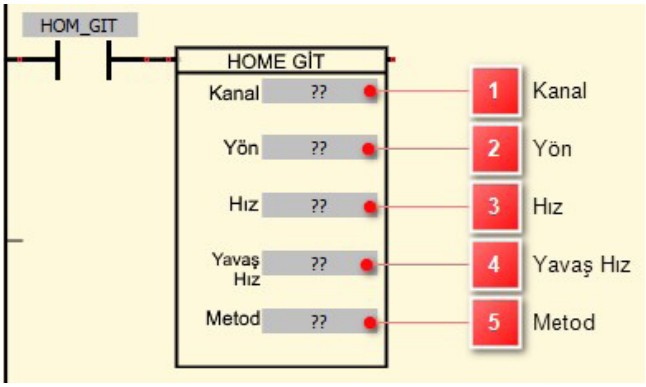
Görsel 3.22: Manuel hareket ladder komutu

Tablo 3.1: Manuel Hareket Komutu Açıklamaları

1	**Kanal:** Puls alınacak kanal seçimi yapılır (0, 1, 2).
2	**Yön:** Motorun home konumuna gidiş yönü Yukarı/Aşağı olarak seçilir.
3	**Hız:** Gidiş hızı yazılır.
4	**Rampa: Home**ye gidiş için gerekli hızın rampa süresi yazılır.

FREKANS		Integer		☑	☑	42003
HOME_GIDIS_SURE		Integer		☑	☑	42005

Görsel 3.23: Uygulamanın MODBUS adres bilgileri



Görsel 3.24: Home git ladder komutu

Tablo 3.2: Home Git Ladder Komutu Açıklaması

1	**Kanal:** Puls alınacak kanal seçimi yapılır (0, 1, 2).
2	**Yön:** Motorun home anahtar tarafı yazılır.
3	**Hız:** Gidiş hızı yazılır.
4	**Yavaş Hız:** Home yavaşlama hızı yazılır.
5	**Metod:** Home bulma metodu (0 Geçişli, 1 Geçişli ve 2 Geçişli) seçimi yapılır. **0 Geçişli:** Motor, home anahtarı görünce hemen durur. **1 Geçişli:** Motor, home anahtarı geçince yavaşlama hızıyla geri döner ve home anahtarını görünce durur. **2 Geçişli:** Motor, home anahtarı geçince yavaşlama hızıyla geri döner. Motor, home anahtarının geçme işi bitince tekrar geri döner ve yavaşlama hızının yarısı bir hızla home anahtarını görünce durur.

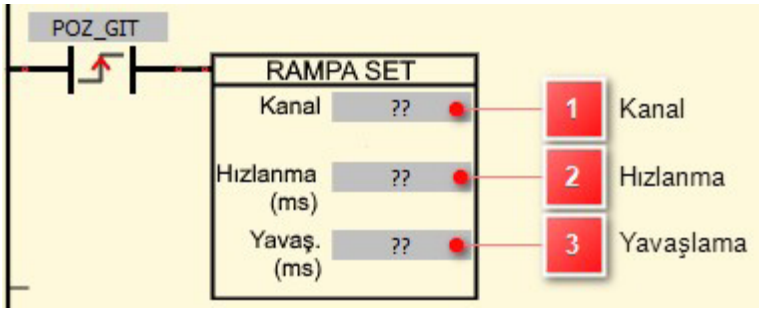
Görsel 3.25: Eksen pozisyon kalibrasyon ladder komutu

Tablo 3.3: Eksen Pozisyon Kalibrasyon Ladder Komutu Açıklaması

1	**Kanal:** Puls alınacak kanal seçimi yapılır (0, 1 ve 2).
2	**Ölçülen Pozis.:** Kalibrasyon öncesi pozisyon değerinin kayıtlı olduğu operand adı yazılır.
3	**Ofset:** Kalibrasyon için ofset değeri sabit ya da real tip operand değeri yazılır.
4	**Çıkış Pozis.:** Kalibre edilmiş yeni çıkış pozisyon değeri bu real tip operanda yazılır.

Home git komutu önce çalıştırılır. Motor home anahtarı görünce durur. Manuel hareket komutları çalıştırılır. Home anahtarına göre ters yönde motor hareket ettirilir. Motor herhangi bir yerde manuel hareket komutuyla durdurulur. Kumpas alarak home anahtarıyla motorun durduğu nokta arasındaki mesafe ölçülür.

Örneğin 150 mm'lik bir değer ölçülen pozisyon parametresine bilgi girişi olarak yazılır. Sonra eksen pozisyon kalibrasyon komutu, yükselen kenar komutuyla çalıştırılır. Eksen pozisyon kalibrasyon komutu çalıştırılınca 150 mm'nin kaç palseye denk geldiğini bilir. PLC, 1 mm'nin kaç palseye denk geldiğini veya 1 palsin kaç mm olduğunu öğrenir.

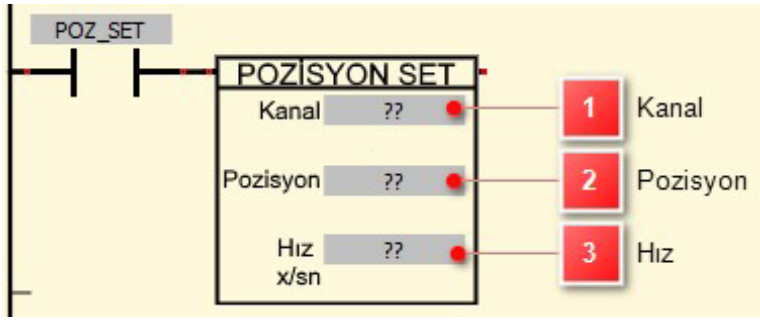


Görsel 3.26: Rampa set ladder komutu

Tablo 3.4: Rampa Set Komutu Açıklaması

1	**Kanal:** Puls alınacak kanal seçimi yapılır (0, 1 ve 2).
2	**Hızlanma:** Motor hızlanma süresi, ms cinsinden sabit ya da real tip operand adı olarak yazılır.
3	**Yavaşlama:** Motor yavaşlama süresi, ms cinsinden sabit ya da real tip operand adı olarak yazılır.

Motorun rampa set fonksiyon bloğunda, motorun ilk hareket anındaki sıfırdan maksimum hıza ulaşma süresi hızlanma parametresine milisaniye cinsinden yazılır. Motorun durma anı ise maksimum hızdan sıfır hıza kadarki geçen süre yavaşlama parametresine milisaniye cinsinden yazılır. Rampa set fonksiyonunu çalıştırmak için bir kere yükselen kenar kontak kullanılarak yapılır.



Görsel 3.27: Pozisyon set ladder komutu

Tablo 3.4: Pozisyon Set Komutu Açıklaması

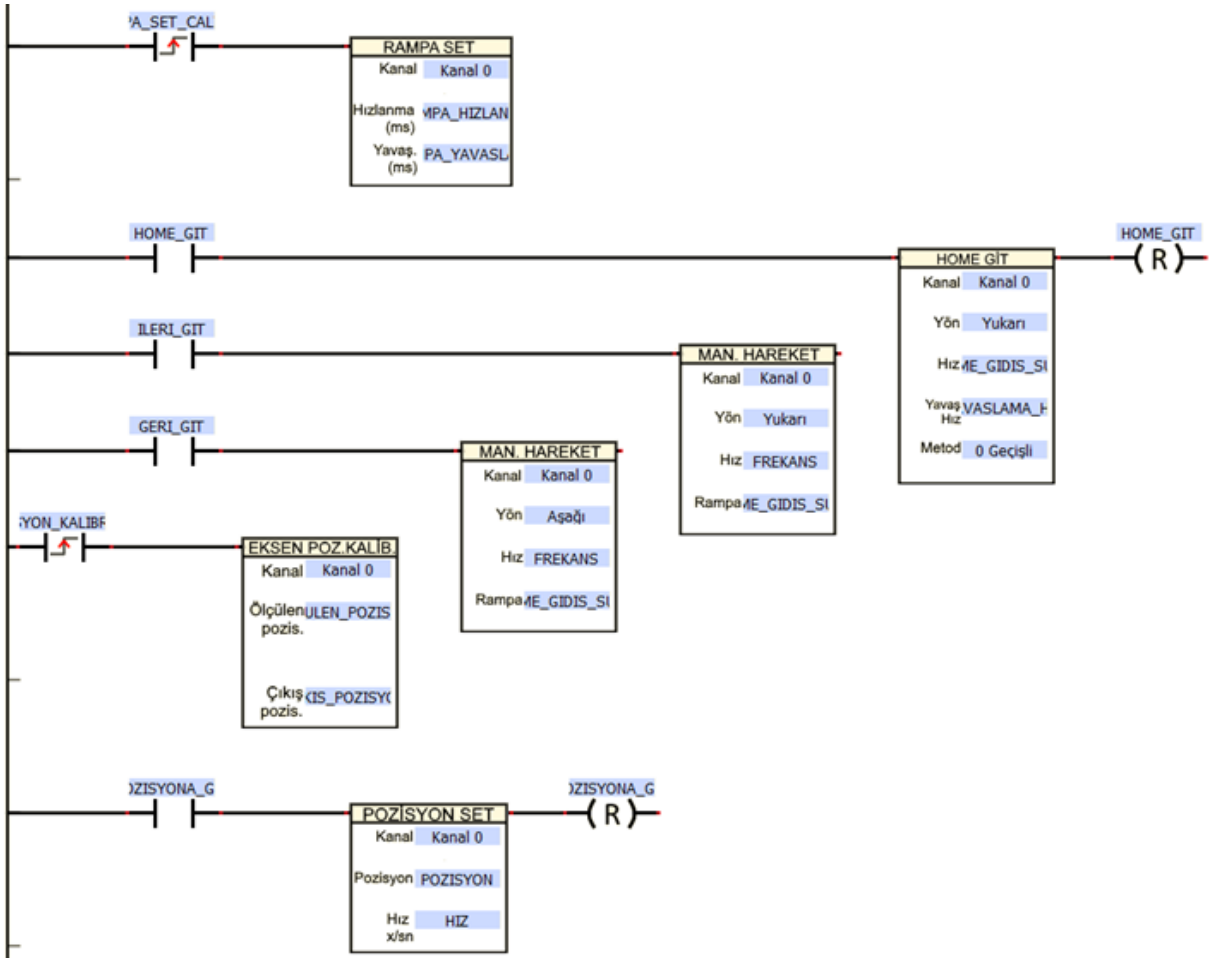
1	**Kanal:** Puls alınacak kanal seçimi yapılır (0, 1 ve 2).
2	**Pozisyon:** Hedef pozisyon sabit ya da operand adı olarak yazılır.
3	**Hız x/sn. :** Hedefe ulaşma hızı sabit ya da operand adı olarak yazılır.

Pozisyon set fonksiyon bloğunda pozisyon parametresine, eksen pozisyon kalibresindeki değerlere göre mm cinsinden bir değer yazılır. Normalde açık kontakla çalıştırılır ve pozisyon set komutu işini bitirince bir çıkış verir. Çıkışa göre reset bobini tekrar normalde açık kontak olan pozisyona git kontağını çalıştırır. Örneğin pozisyon parametresine 30 mm git diye yazılırsa PLC arka planda kaç pals üreteceğini hesaplar ve ona göre motoru hareketlendirir.

Hız parametresinde kalibrasyondaki ölçülen değer mm ise motorun hızı da mm/sn. olur. Bu parametre sanitmetre ise motorun hızı da cm/sn. olur. Kalibrasyon mm cinsinden hız parametresine 10 yazılırsa bu, 1 sn.'de 10 mm motor hareket eder demektir. Dikkat edilmesi gereken husus hız parametresine çok büyük değerler girilmez, PLC'nin çalışma hızının üzerine çıkılmaz. Örneğin, pozisyon parametresine 30 mm, hız parametresine de 5 mm/sn. yazılır. Bu durumda 1 sn.'de 5 mm'lik bir hareket gerçekleşir. 30 mm'de ise 6 sn.'lik bir zaman geçer.

POZISYONA_GIT		Bit	0	☐	☑	10
POZISYON		Integer	0	☐	☑	42015
HIZ		Integer	0	☐	☑	42017

Görsel 3.28: Uygulamanın MODBUS adres bilgileri



Görsel 3.29: Uygulamanın ladder program ekran görüntüsü

UST_SWITCH		Bit	0	☐	☑	1
HOME_SWITCH		Bit	0	☐	☑	3
ALT_SWITCH		Bit	0	☐	☑	2
ACIL_SWITCH		Bit	0	☐	☑	4
EKSEN_DURUM		Word	0	☐	☑	40001
ILERI_GIT		Bit	0	☐	☑	5
FREKANS		Integer	0	☐	☑	42003
HOME_GIDIS_SURE		Integer	0	☐	☑	42005

GERI_GIT		Bit	0	☐	☑	6
HOME_GIT		Bit	0	☐	☑	7
YAVASLAMA_HIZ		Integer	0	☐	☑	42007
POZISYON_KALIBRE_ET		Bit	0	☐	☑	8
OLCULEN_POZISYON		Real	0	☐	☑	42009
CIKIS_POZISYON		Real	0	☐	☑	42011
RAMPA_SET_CALISTIR		Bit	0	☐	☑	9
RAMPA_YAVASLAMA		Real	0	☐	☑	42013
POZISYONA_GIT		Bit	0	☐	☑	10
RAMPA_HIZLANMA		Real	0	☐	☑	42019
POZISYON		Integer	0	☐	☑	42015
HIZ		Integer	0	☐	☑	42017
TOPLAM_PALS		Word	0	☐	☑	40002

Görsel 3.30: Uygulamanın MODBUS adres bilgileri

Motor mutlak hareket işlemi için öncelikle home pozisyon işlemi aşağıdaki sırayla çalıştırılır:

- Rampa set komutu set reset şeklinde çalıştırılır.
- Home git komutu set reset şeklinde çalıştırılır. Motor home anahtara doğru hareket eder.
- Home anahtar hangi yönde ise o yöne doğru manuel hareket komutlarından biri çalıştırılır. Motor o yöne doğru hareket eder.
- Enerjilenme kısmında home anahtarı set reset şeklinde çalıştırılır. Bu durumda motor, home pozisyona gelir. Aynı işlem digital input sensörleriyle de yapılabilir.

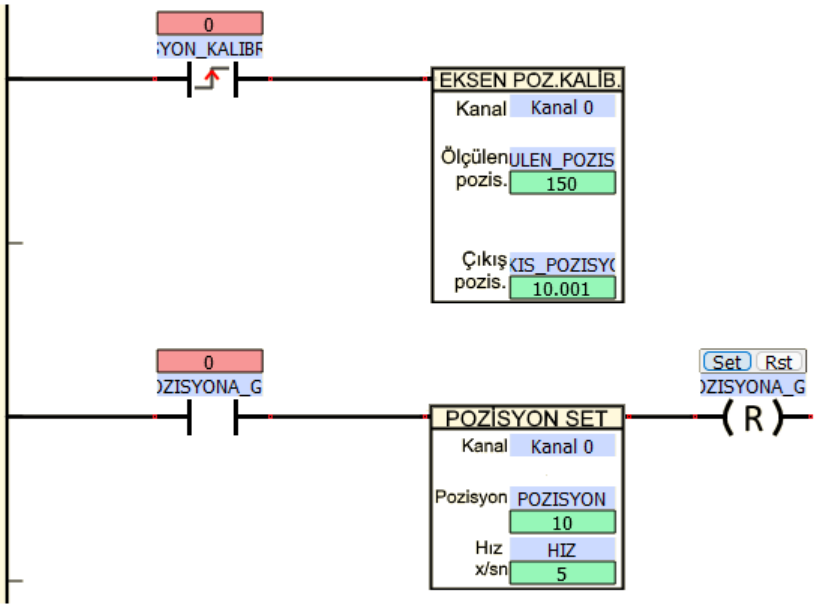
Eksen pozisyon kalibrasyon işlemi aşağıdaki sırayla çalıştırılır:

- Önce home pozisyon işlemi yapılır.
- Home pozisyondan uzaklaşacak şekilde olan manuel hareket komutuyla motor hareket ettirilir. Bir süre sonra motor durdurulur. Home noktasından motorun durduğu yere kadar olan mesafe kumpas ile ölçülür. Ölçüm mm, cm veya başka bir ölçü birimiyle yapılır (örneğin ölçülen değer 150 mm).
- Ölçülen rakam pozisyon parametresine 150 olarak yazılır. Eksen pozisyon kalibrasyon komutuna ait yükselen kontak set reset şeklinde çalıştırılır. PLC, 150 rakamını çıkış pozisyon parametresine atar ve kalibrasyon işlemini tamamlar.

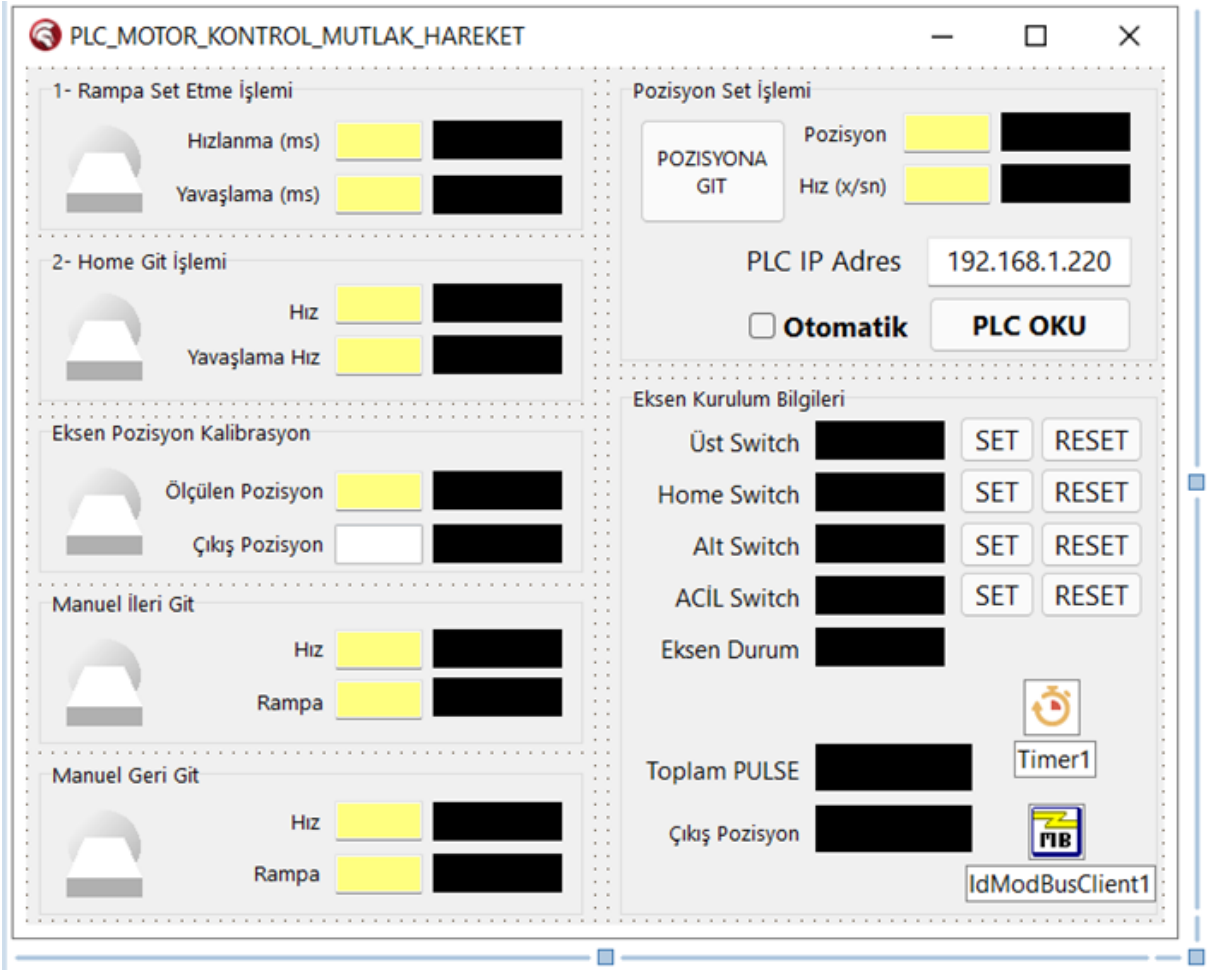
Pozisyon set fonksiyon bloğunun çalıştırılması için aşağıdaki sıra uygulanır:

- Öncelikle home pozisyon ve kalibrasyon işlemleri yapılır.
- Hız parametresine 1 mm'lik mesafeye kaç saniyede motor hareket edilir bilgisi yazılır.
- Pozisyon parametresine motor kaç mm gitsin bilgisi yazılır.
- Pozisyon set komutunu çalıştırmak için normalde açık kontağı set reset şeklinde çalıştırılır.
- Örneğin 10 rakamı pozisyon parametresine yazılarak motor hareket ettirilir. Toplam pals bilgisine denk gelen mm değeri, eksen pozisyon kalibrasyon komutunun çıkış pozisyon parametresinden ayrıca izlenebilir. Her bir rakam benzer şekilde çalıştırılır

PLC'nin enerjisi kesilip tekrar enerjilendiğinde HOME POZİSYON ve EKSEN KALİBRASYON işlemleri tekrar yapılır.



Görsel 3.31: Uygulamanın ladder program ekran görüntüsü



Görsel 3.32: Uygulamanın tasarım anı ekran görüntüsü

```
procedure TForm1.Button1Click(Sender: TObject);
var
  DATA: array [0 .. 2] of Word; // 2 Byte'lık bilgi için
begin

  if (Edit12.Text = '') or (Edit13.Text = '') then
  begin
    ShowMessage('Değer zorunlu');
    Exit;
  end;

  IdModBusClient1.Host := Edit7.Text;
  // Pozisyon Set POZISYON "Integer Type"
  DATA[0] := StrToInt(Edit12.Text);
  DATA[1] := 0;
  IdModBusClient1.WriteRegisters(2015, DATA);
  DATA[0] := StrToInt(Edit13.Text);
  DATA[1] := 0;
  IdModBusClient1.WriteRegisters(2017, DATA);
  IdModBusClient1.WriteCoil(10, True);
end;
procedure TForm1.Button5Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit7.Text;
  IdModBusClient1.WriteCoil(1, True);
end;

procedure TForm1.Button6Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit7.Text;
  IdModBusClient1.WriteCoil(1, False);
end;

procedure TForm1.Button7Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit7.Text;
  IdModBusClient1.WriteCoil(3, True);
end;

procedure TForm1.Button8Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit7.Text;
  IdModBusClient1.WriteCoil(3, False);
end;

procedure TForm1.Button9Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit7.Text;
  IdModBusClient1.WriteCoil(2, True);
end;

procedure TForm1.Button10Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit7.Text;
  IdModBusClient1.WriteCoil(2, False);
end;
```

```
procedure TForm1.Button11Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit7.Text;
  IdModBusClient1.WriteCoil(4, True);
end;

procedure TForm1.Button12Click(Sender: TObject);
begin
  IdModBusClient1.Host := Edit7.Text;
  IdModBusClient1.WriteCoil(4, False);
end;

procedure TForm1.CheckBox1Click(Sender: TObject);
begin
  Timer1.Enabled := CheckBox1.Checked;
end;

procedure TForm1.iSwitchLever1Change(Sender: TObject);
var
  DATA: array [0 .. 2] of Word; // 2 Byte'lık bilgi için
begin

  if (Edit3.Text = '') or (Edit4.Text = '') then
  begin
    ShowMessage('Değer zorunlu');
    iSwitchLever1.Active := False;
    Exit;
  end;

  IdModBusClient1.Host := Edit7.Text;
  if iSwitchLever1.Active then
  begin
    // Home Git HOME_GIDIS_SURE "Integer Type"
    DATA[0] := StrToInt(Edit3.Text);
    DATA[1] := 0;
    IdModBusClient1.WriteRegisters(2005, DATA);
    // Home Git YAVASLAMA_HIZ "Integer Type"
    DATA[0] := StrToInt(Edit4.Text);
    DATA[1] := 0;
    IdModBusClient1.WriteRegisters(2007, DATA);
    IdModBusClient1.WriteCoil(7, True);
  end
  else
    IdModBusClient1.WriteCoil(7, False);
  Button13Click(Self); // PLC'den oku
end;

procedure TForm1.Button13Click(Sender: TObject);
var
  DEGER: Single;
  DEGER1: array [0 .. 4096] of Word;
  LEDLER: array [0 .. 4096] of Boolean;
  EKSEN_DURUM: Word;
  I: Integer;
begin
  IdModBusClient1.Host := Edit7.Text;
  // Rampa Set HIZLANMA, YAVASLAMA "Real Type"
  IdModBusClient1.ReadSingle(2019, DEGER);
  Label26.Caption := DEGER.ToString;
  IdModBusClient1.ReadSingle(2013, DEGER);
  Label27.Caption := DEGER.ToString;
```

```
  // Eksen Pozisyon Kalibrasyon OLCULEN_POZISYON, CIKIS_POZISYON "Real Type"
  IdModBusClient1.ReadSingle(2009, DEGER);
  Label31.Caption := DEGER.ToString;
  IdModBusClient1.ReadSingle(2011, DEGER);
  Label30.Caption := FormatFloat('###,###,##0.000', DEGER);
  Label25.Caption := Label30.Caption;
  IdModBusClient1.ReadHoldingRegisters(2001, 20, DEGER1);
  // Toplam Pals  "Word Type"
  Label19.Caption := (DEGER1[0] + DEGER1[1]).ToString; // 2 Byte
  // Manuel İleri-Geri Git FREKANS, HOME_GIDIS_SURE "Integer Type"
  Label32.Caption := (DEGER1[2] + DEGER1[3]).ToString; // 2 Byte
  Label33.Caption := (DEGER1[4] + DEGER1[5]).ToString; // 2 Byte
  Label34.Caption := (DEGER1[2] + DEGER1[3]).ToString; // 2 Byte
  Label35.Caption := (DEGER1[4] + DEGER1[5]).ToString; // 2 Byte
  // Home Git HOME_GIDIS_SURE, YAVASLAMA_HIZ "Integer Type"
  Label29.Caption := (DEGER1[4] + DEGER1[5]).ToString; // 2 Byte
  Label28.Caption := (DEGER1[6] + DEGER1[7]).ToString; // 2 Byte
  // Pozisyon Set POZISYON, HIZ "Integer Type"
  Label37.Caption := (DEGER1[14] + DEGER1[15]).ToString; // 2 Byte
  Label36.Caption := (DEGER1[16] + DEGER1[17]).ToString; // 2 Byte
  IdModBusClient1.ReadCoils(1, 10, LEDLER);
  iLedRound7.Active := LEDLER[0]; // ok
  iLedRound9.Active := LEDLER[1]; // ok
  iLedRound8.Active := LEDLER[2]; // ok
  iLedRound10.Active := LEDLER[3]; // ok
  iLedRound4.Active := LEDLER[4];
  iLedRound5.Active := LEDLER[5];
  iLedRound2.Active := LEDLER[6];
  iLedRound3.Active := LEDLER[7];
  iLedRound1.Active := LEDLER[8];
  iLedRound6.Active := LEDLER[9];
  IdModBusClient1.ReadHoldingRegister(1, EKSEN_DURUM);
  Label15.Caption := EKSEN_DURUM.ToString;
end;
```

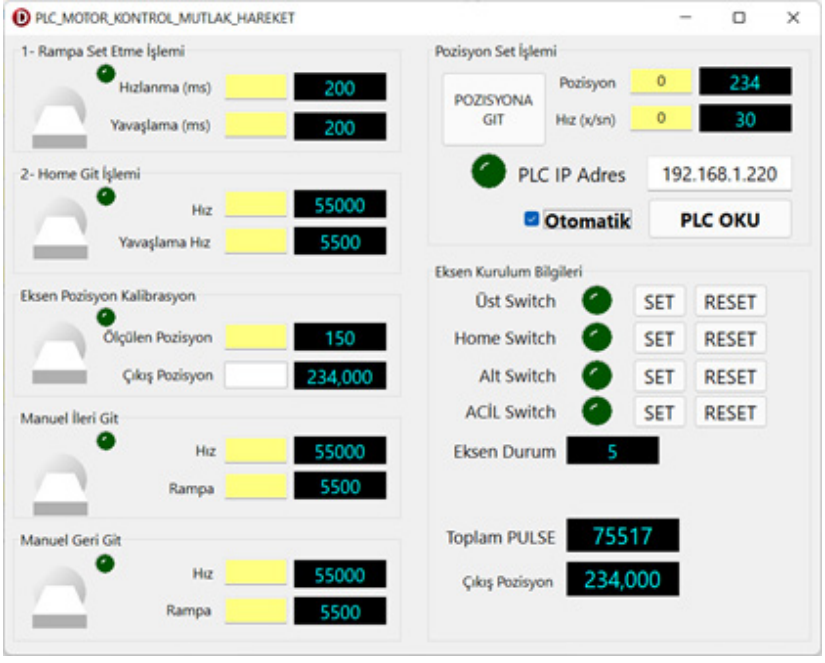
```
procedure TForm1.iSwitchLever2Change(Sender: TObject);
begin
  if (Edit1.Text = '') or (Edit2.Text = '') then
  begin
    ShowMessage('Değer zorunlu');
    iSwitchLever2.Active := False;
    Exit;
  end;

  IdModBusClient1.Host := Edit7.Text;
  if iSwitchLever2.Active then
  begin
    IdModBusClient1.WriteSingle(2019, StrToInt(Edit1.Text));
    IdModBusClient1.WriteSingle(2013, StrToInt(Edit2.Text));
    IdModBusClient1.WriteCoil(9, True);
  end
  else
  begin
    IdModBusClient1.WriteCoil(9, False);
  end;
  Button13Click(Self); // PLC'den oku
end;
```

```
procedure TForm1.iSwitchLever3Change(Sender: TObject);
begin
  if (Edit10.Text = '') then
  begin
    ShowMessage('Değer zorunlu');
    iSwitchLever3.Active := False;
    Exit;
  end;

  IdModBusClient1.Host := Edit7.Text;

  if iSwitchLever3.Active then
  begin
    IdModBusClient1.WriteSingle(2009, StrToInt(Edit10.Text));
    IdModBusClient1.WriteCoil(8, True);
  end
  else
  begin
    IdModBusClient1.WriteCoil(8, False);
  end;

  Button13Click(Self); // PLC'den oku
end;
```

```
procedure TForm1.iSwitchLever4Change(Sender: TObject);
var
  DATA: array [0 .. 2] of Word; // 2 Byte'lık bilgi için
begin

  if (Edit5.Text = '') or (Edit6.Text = '') then
  begin
    ShowMessage('Değer zorunlu');
    iSwitchLever4.Active := False;
    Exit;
  end;

  IdModBusClient1.Host := Edit7.Text;
  if iSwitchLever4.Active then
  begin
    // Manuel İleri Git HIZ "Integer Type"
    DATA[0] := StrToInt(Edit5.Text);
    DATA[1] := 0;
    IdModBusClient1.WriteRegisters(2005, DATA);
    // Manuel İleri Git RAMPA "Integer Type"
    DATA[0] := StrToInt(Edit6.Text);
    DATA[1] := 0;
    IdModBusClient1.WriteRegisters(2003, DATA);
    IdModBusClient1.WriteCoil(5, True);
  end
  else
    IdModBusClient1.WriteCoil(5, False);
  Button13Click(Self); // PLC'den oku
end;
```

```
procedure TForm1.iSwitchLever5Change(Sender: TObject);
var
  DATA: array [0 .. 2] of Word; // 2 Byte'lık bilgi için
begin
  if (Edit8.Text = '') or (Edit9.Text = '') then
  begin
    ShowMessage('Değer zorunlu');
    iSwitchLever5.Active := False;
    Exit;
  end;
  IdModBusClient1.Host := Edit7.Text;
  if iSwitchLever5.Active then
  begin
    // Manuel Geri Git HIZ "Integer Type"
    DATA[0] := StrToInt(Edit8.Text);
    DATA[1] := 0;                          // 42005 - 40000 = 2005
    IdModBusClient1.WriteRegisters(2005, DATA);
    // Manuel Geri Git RAMPA "Integer Type"
    DATA[0] := StrToInt(Edit9.Text);
    DATA[1] := 0;
    IdModBusClient1.WriteRegisters(2003, DATA);
    IdModBusClient1.WriteCoil(6, True);
  end
  else
    IdModBusClient1.WriteCoil(6, False);
  Button13Click(Self); // PLC'den oku
end;
```

Görsel 3.33: Uygulamanın program kodları



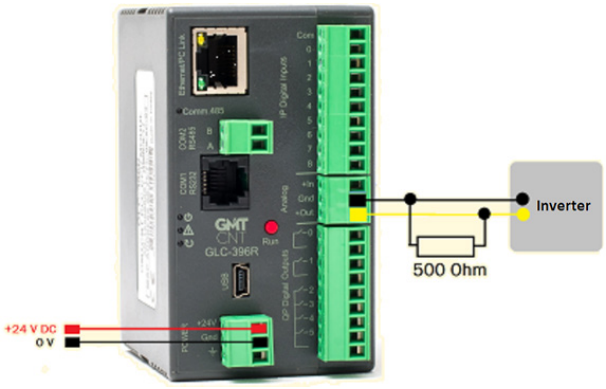
Görsel 3.34: Uygulamanın çalışma anı ekran görüntüsü

3.2. ASENKRON MOTOR

Sanayide, sıkça kullanılan asenkron motorların sürücülerine **inverter** ismi verilir. Inverter; asenkron motorların frekans değişimine göre hızlarını, devir yön değişimlerini, durdurma ve çalıştırma işlemlerini yapar.

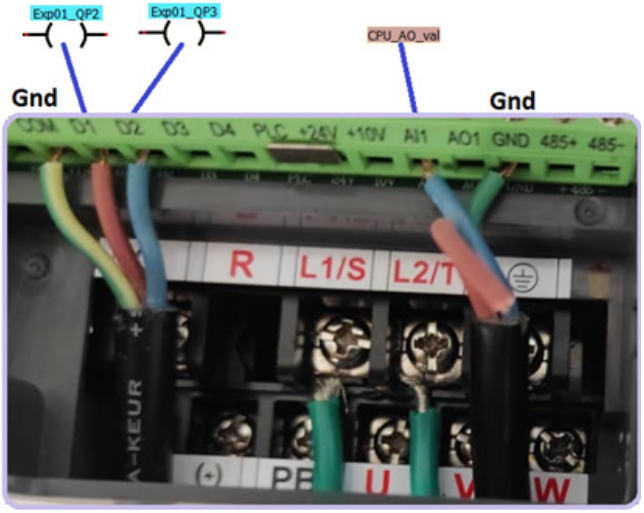
3.4. UYGULAMA: PLC ile Inverter ve Asenkron Motor Kontrol

PLC'nin analog outputu, inverterin analog inputuna (AI1) bağlanır. PLC'nin iki adet digital çıkışı, inverterin D1 ve D2 digitalinput girişlerine bağlanır. Inverterin D1 ucu sola dönme, D2 ucu ise sağa dönme işlemlerini gerçekleştirir. Inverterin U, V ve W uçları; asenkton motorun U, V ve W uçlarına bağlanır. Invertere 220 volt gerilim iş güvenliği tedbirleri doğrultusunda uygulanır.



Görsel 3.35: PLC'nin invertere bağlantı şeması

Görsel 3.36: İnverter

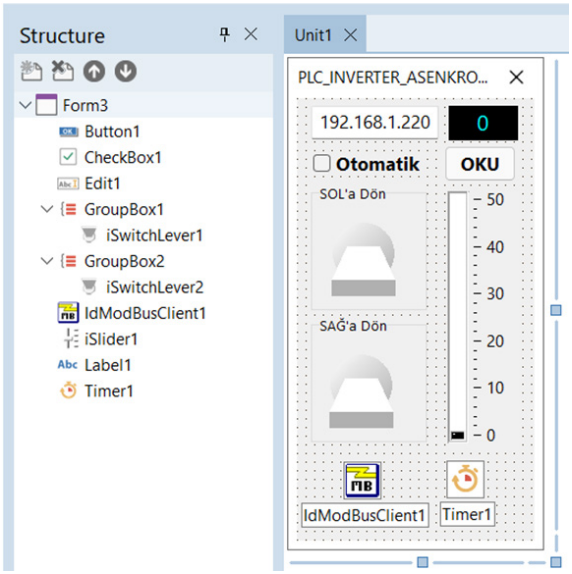


Görsel 3.37: İnverterin bağlantı klemens ekran görüntüsü

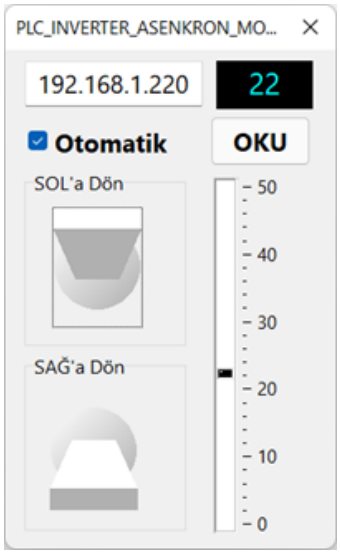
Görsel 3.38: Asenkron motor

Tablo 3.5: Bağlantı Uçları

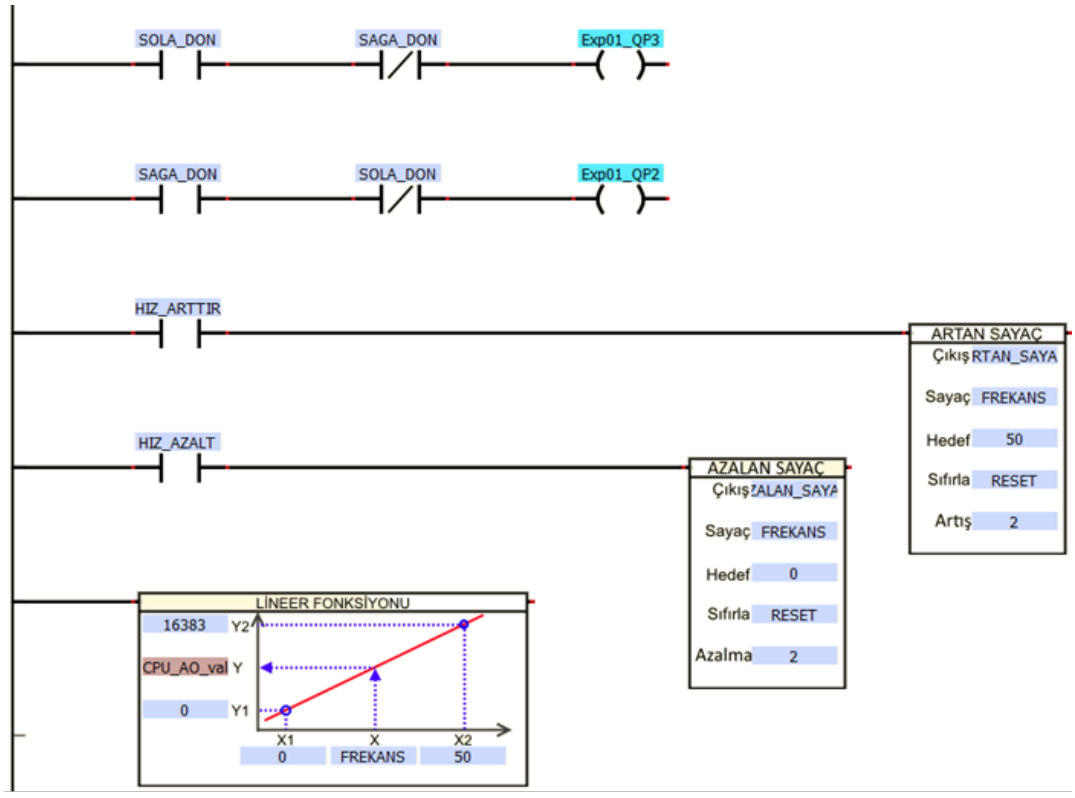
U, V, W	Üç fazlı asenkron motor (yıldız bağlı) bağlantı uçları
L, N	Faz ve nötr bağlantı uçları
COM, D1	Digital giriş (ON/OFF anahtarı) uçları SAĞ veya İLERİ FORWARD tuşları
COM, D2	Digital giriş (ON/OFF anahtarı) uçları SOL veya GERİ REVERSE tuşları
AI1, GND, +10V	Analog input (potansiyometre) uçları



Görsel 3.39: Uygulamanın tasarım anı ekran görüntüsü



Görsel 3.40: Uygulamanın çalışma anı ekran görüntüsü



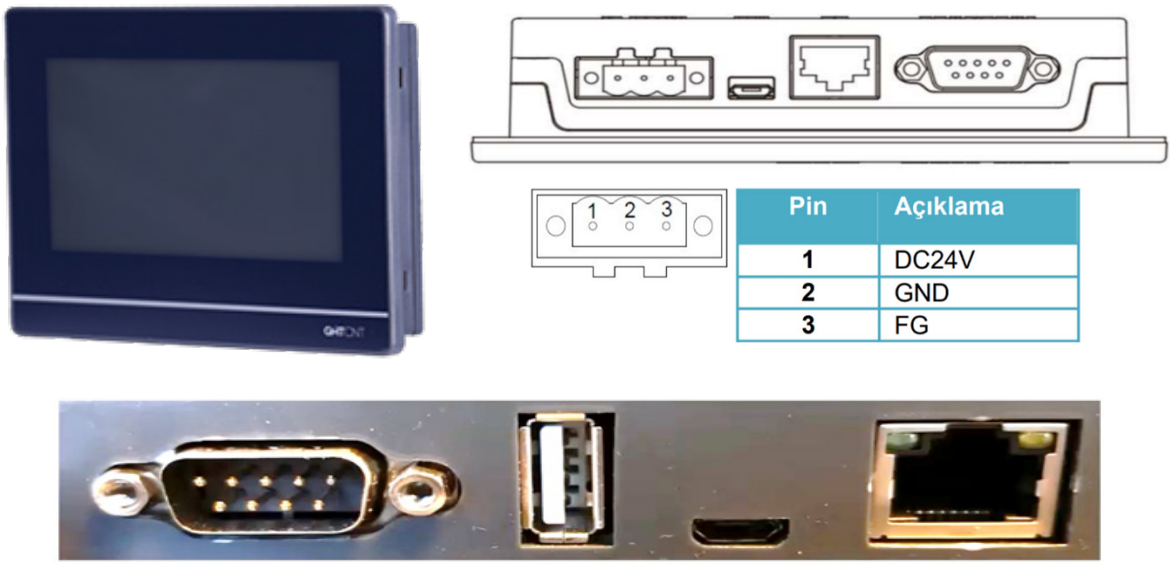
Görsel 3.41: Uygulamanın ladder program ekran görüntüsü

```
procedure TForm3.Button1Click(Sender: TObject);
var
  FREKANS: Word;
begin
  IdModBusClient1.Host := Edit1.Text;
  IdModBusClient1.ReadHoldingRegister(1, FREKANS);
  Label1.Caption := FREKANS.ToString;
end;

procedure TForm3.CheckBox1Click(Sender: TObject);
begin
  Timer1.Enabled := CheckBox1.Checked;
end;

procedure TForm3.iSlider1PositionChangeFinished(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
  IdModBusClient1.WriteRegister(1, Round(iSlider1.Position));
end;

procedure TForm3.iSwitchLever1Change(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
  if iSwitchLever1.Active then
    IdModBusClient1.WriteCoil(1, True)
  else
    IdModBusClient1.WriteCoil(1, False);
end;
procedure TForm3.iSwitchLever2Change(Sender: TObject);
begin
  IdModBusClient1.Host := Edit1.Text;
  if iSwitchLever2.Active then
    IdModBusClient1.WriteCoil(2, True)
  else
    IdModBusClient1.WriteCoil(2, False);
end;

procedure TForm3.Timer1Timer(Sender: TObject);
begin
  Button1Click(Self);
end;
```

Görsel 3.42: Uygulamanın program kodları

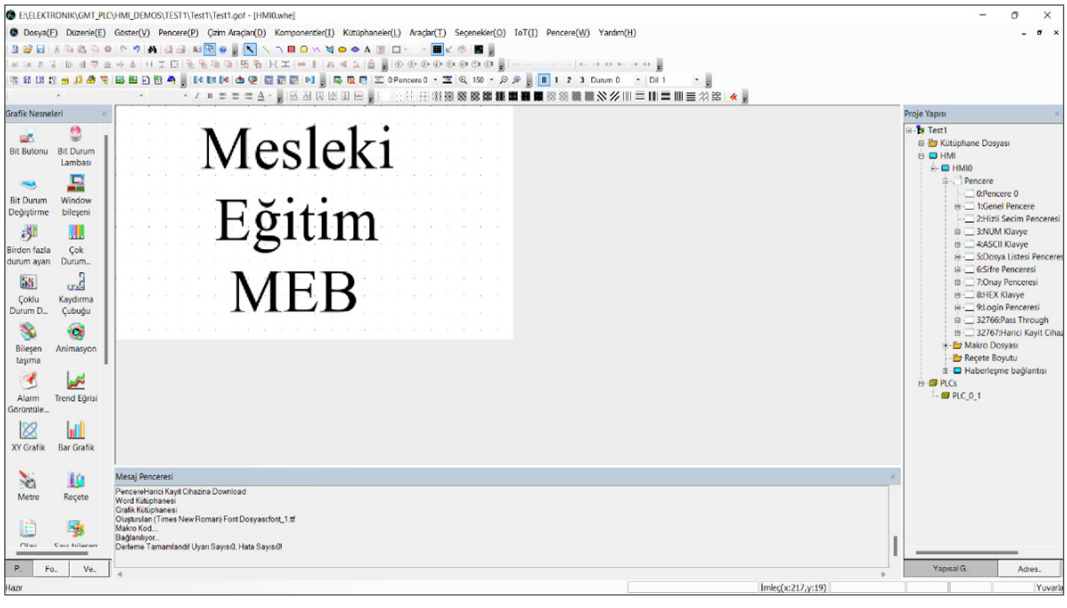
3.3. OPERATÖR PANEL PROGRAMLAMA

Bir işin insan ile makine arasında paylaşılmasına **otomasyon** denir. HMI (Human Machine Interface) insan makine arayüzü ismi verilen bu dokunmatik paneller otomasyonun görülen yüzüdür. Bu operatörün dokunmatik özelliği vardır. Panellerde haberleşme TCP, COM ve USB portlarıyla yapılırken modbus protokolü ile programlanır. Bu panellerin 700 ve üst MHz'larda, ARM 32 bitlik mikrodenetleyicileri yapılarında bulundurur (Görsel 3.43).



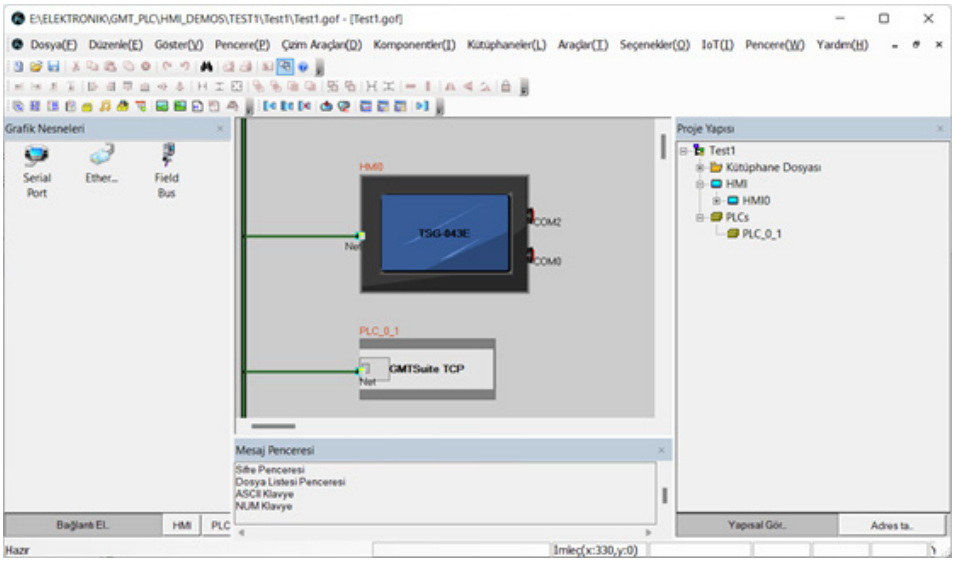
Görsel 3.43: HMI operatör panel, port yuvaları ve enerji bilgileri

Operatör panelin dokunmatik yüzeyine parmak basılıyken, enerji verilince sistem menüsü gelir. Buradan parlaklık, tarih saat, sabit IP vb. bilgiler ayarlanır ve dahili pili sayesinde bilgiler muhafaza edilir. Yerli üretim olan GMT fimasının web sitesindeki https://gmtcontrol.com/wp-content/uploads/2021/08/PDesigner_V3.5.2_Build_210526.zip linkinden operatör panel editör programı indirilir ve kurulum işlemleri gerçekleştirilir (Görsel 3.44).



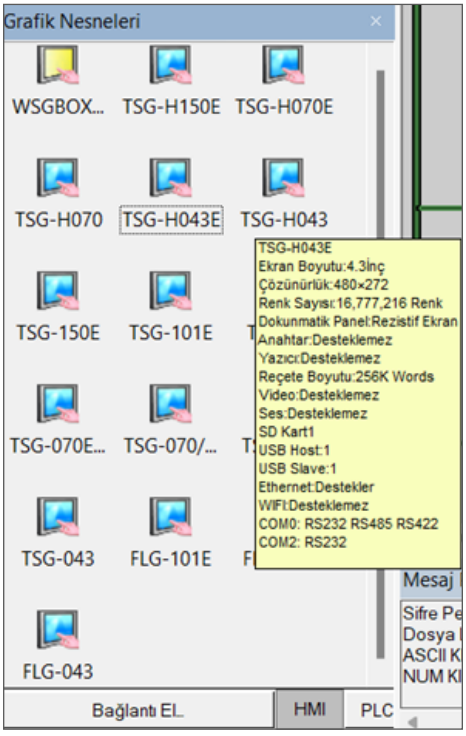
Görsel 3.44: HMI operatör panel editör programı ve PDesigner ekran görüntüsü

3.5. UYGULAMA: PDesigner Operatör Panel Editörünü Tanıma



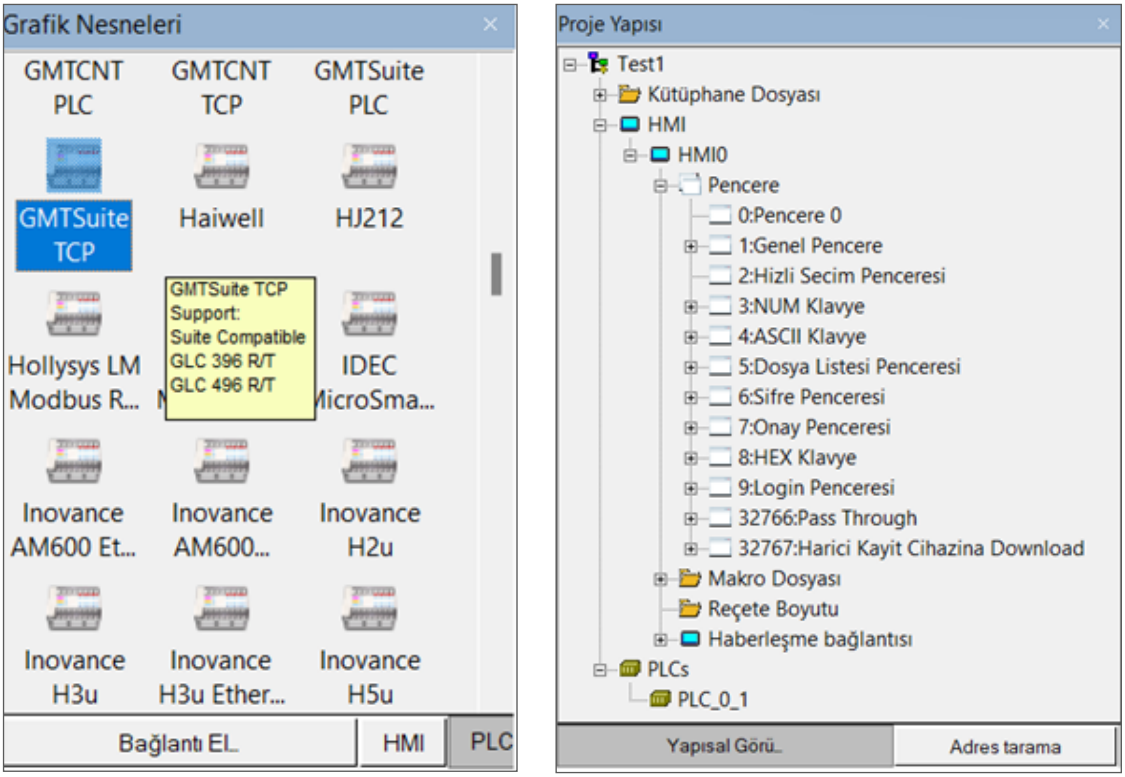
Görsel 3.45: PDesigner operatör panel editör ekranı

Yukarıda **Bağlantı Elemanları** sekmesinde hangi portla haberleşme gerçekleştirilecekse onun seçimi yapılır. HMI sekmesinden model operatörün panel seçimi yapılır (Görsel 3.45).

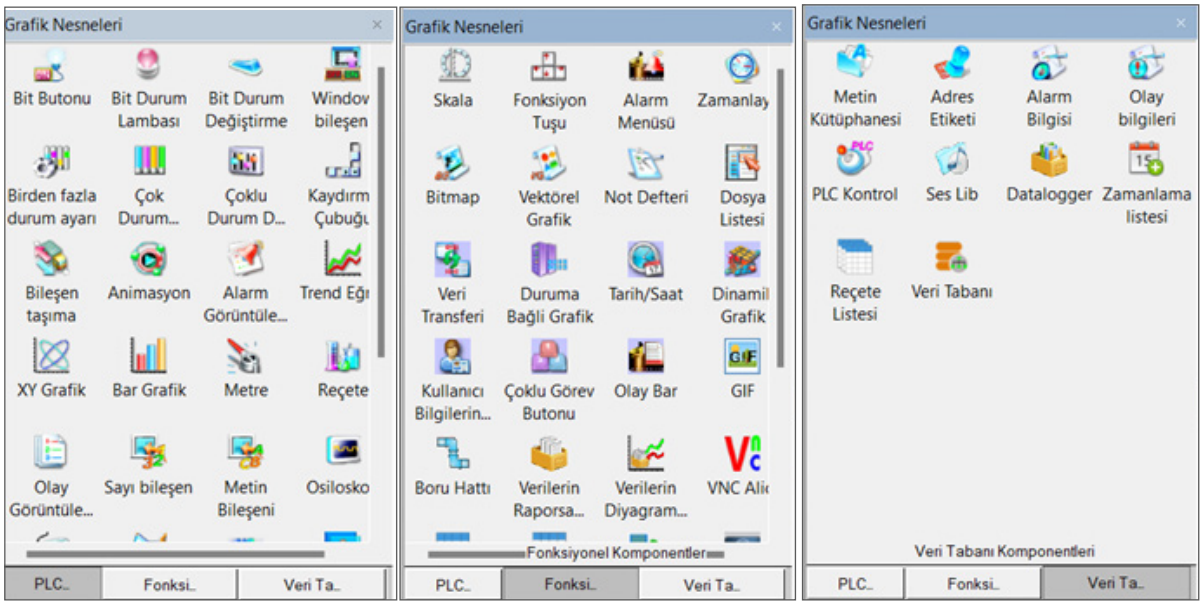


Görsel 3.46: HMI seçim ekranı

PLC sekmesinde HMI panel ile haberleşmesi yapılan cihazların seçimi yapılır. Birden fazla cihaz eklenebilir. Cihaz seçimi, arka planda o cihaza ait sürücü yüklenmesi demektir (Görsel 3.46).



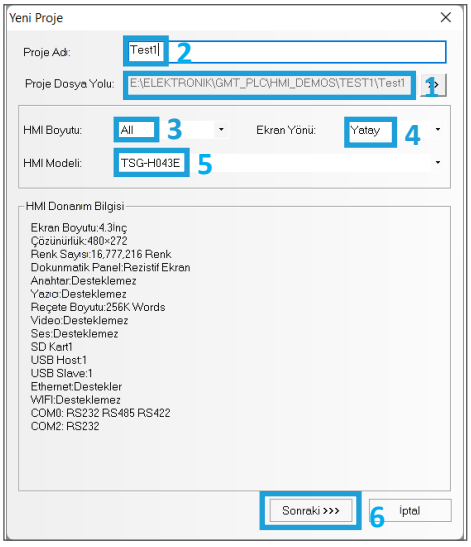
Görsel 3.47: Uygulamaya PLC eklenmesi ve proje yapısındaki görünümü



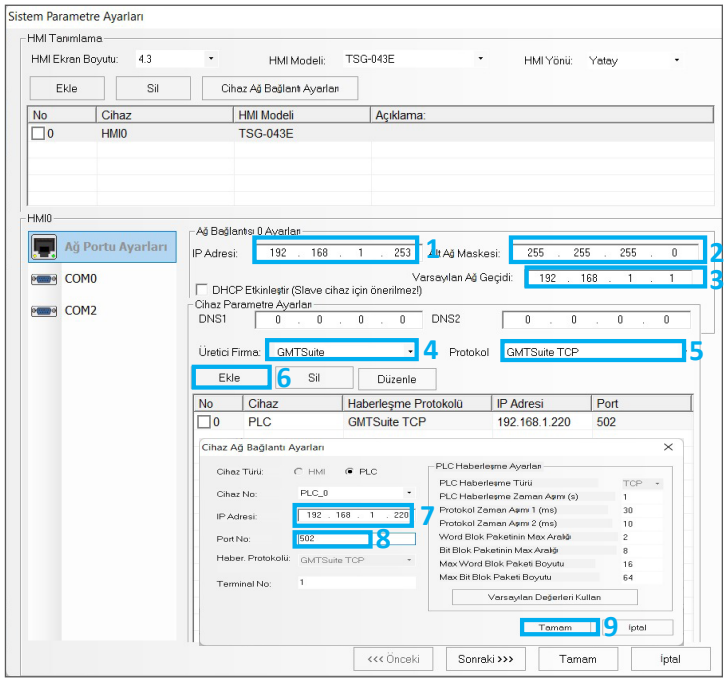
Görsel 3.48: PDesigner panel editör programında kullanılan bileşenler

3.6. UYGULAMA: PDesigner Operatör Panel Editörü ile Yeni Proje Oluşturma

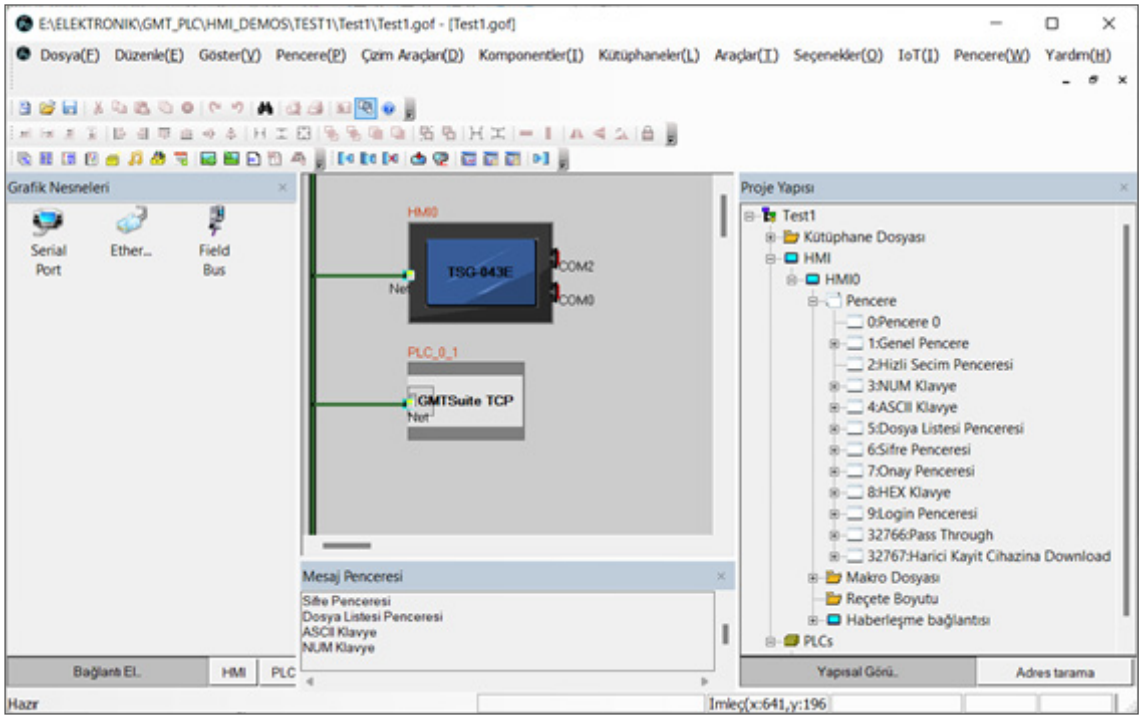
PDesginer programı çalıştırılır, **Dosya** menüsünden **Yeni Proje Aç** komutu verilir.



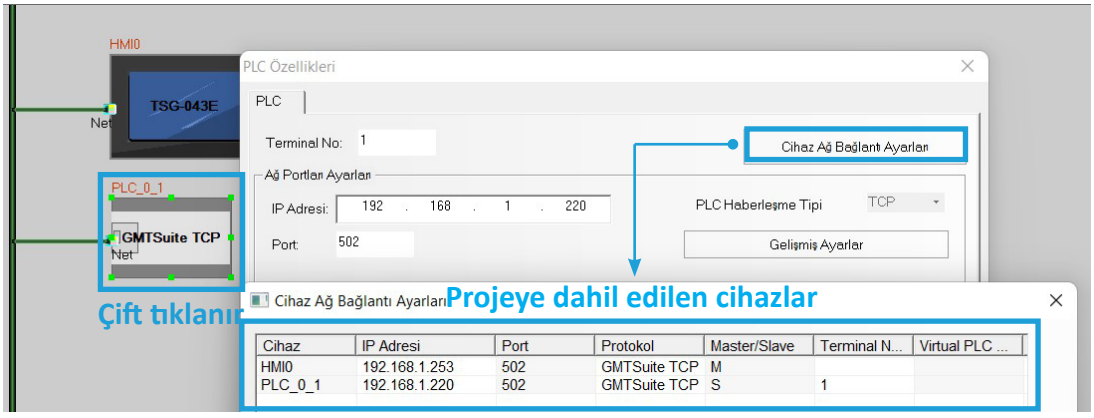
Görsel 3.49: Yeni proje oluşturma



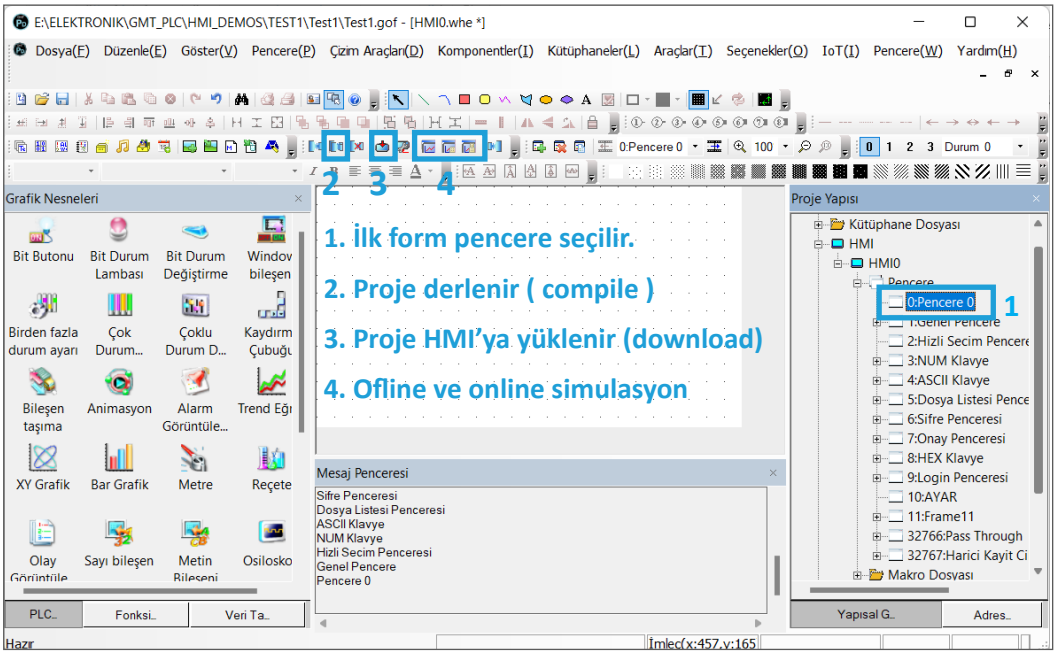
Görsel 3.50: HMI ve PLC'nin IP ayarları



Görsel 3.51: PDesigner operatör panel editör ekranı

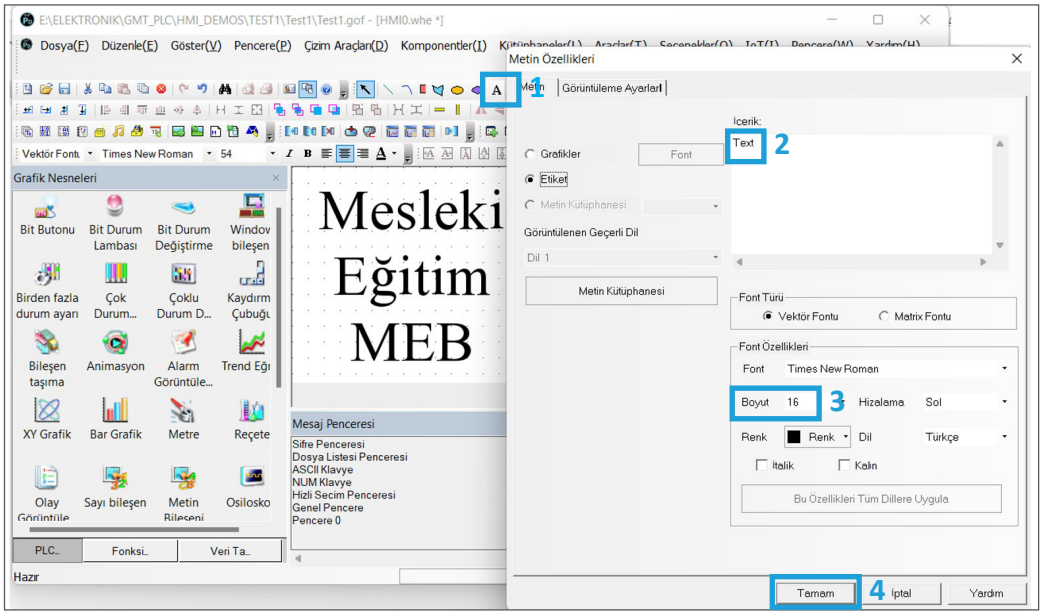


Görsel 3.52: Projeye dâhil edilen cihazlar



Görsel 3.53: Uygulamanın HMI'ya yüklenmesi

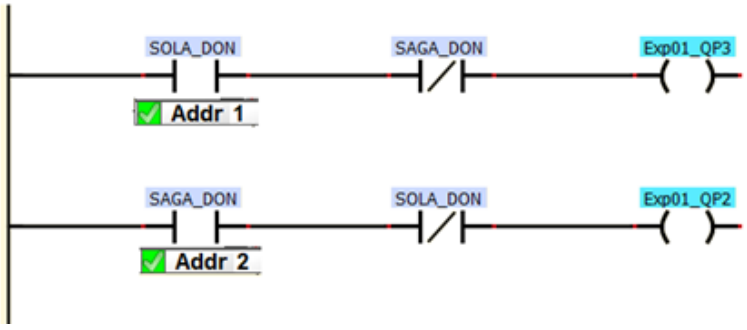
Görsel 3.54: HMI simülasyon ekranı

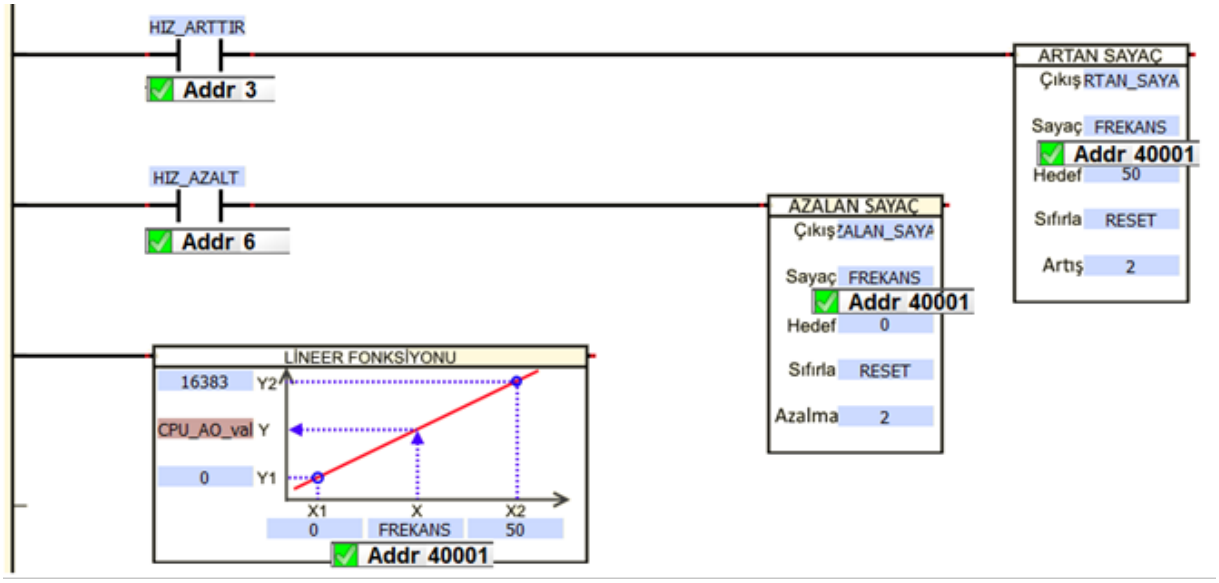


Görsel 3.55: Metin bileşenine bilgi girilmesi

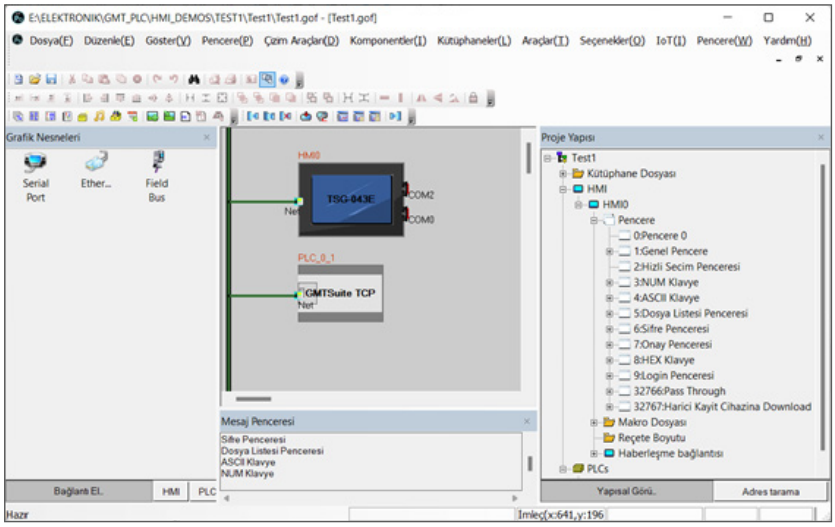
Görsel 3.56: Metin bileşenine girilen bilginin simülasyon görüntüsü

3.7. UYGULAMA: HMI ve PLC ile Asenkron Motor Kontrol

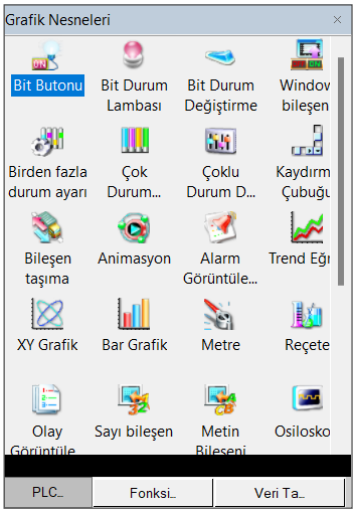




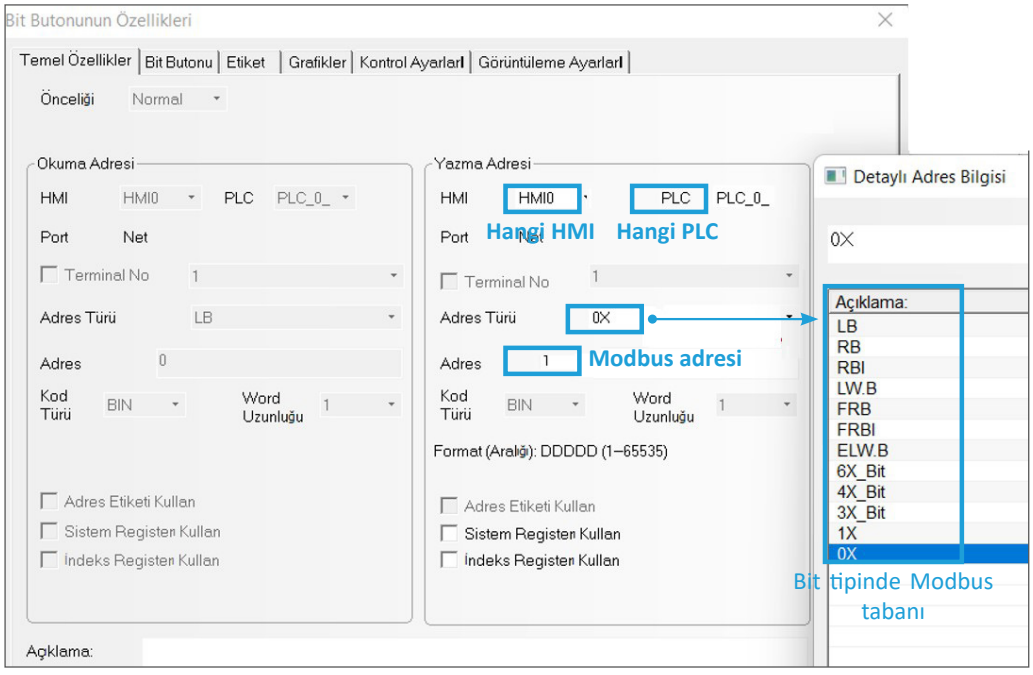
Görsel 3.57: Uygulamanın ladder program ekran görüntüsü



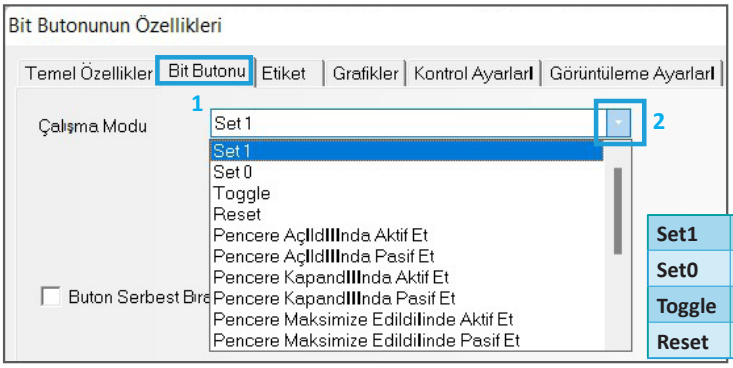
Görsel 3.58: PDesigner operatör panel editör ekranı



Görsel 3.59: Bileşenlerin ekran görüntüsü

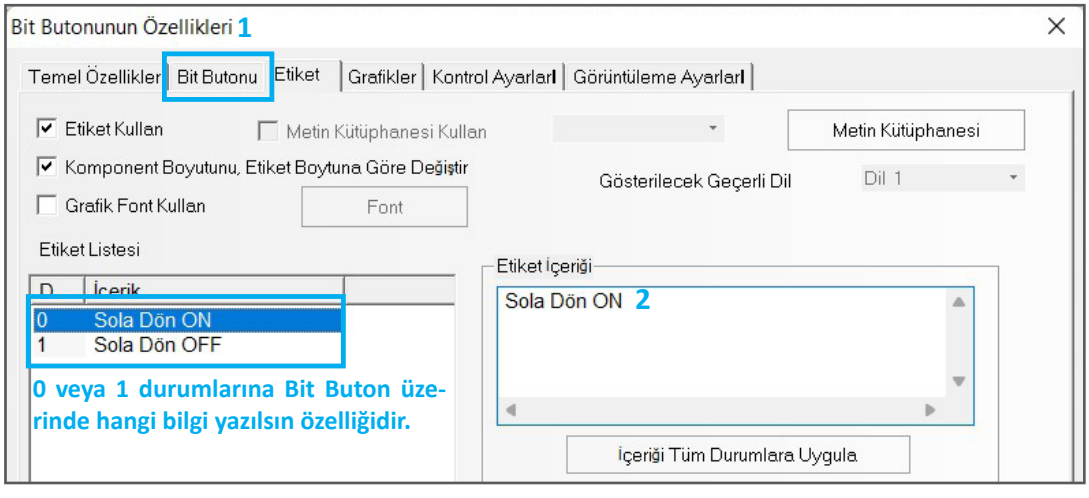


Görsel 3.60: Bit butonunun ayarları

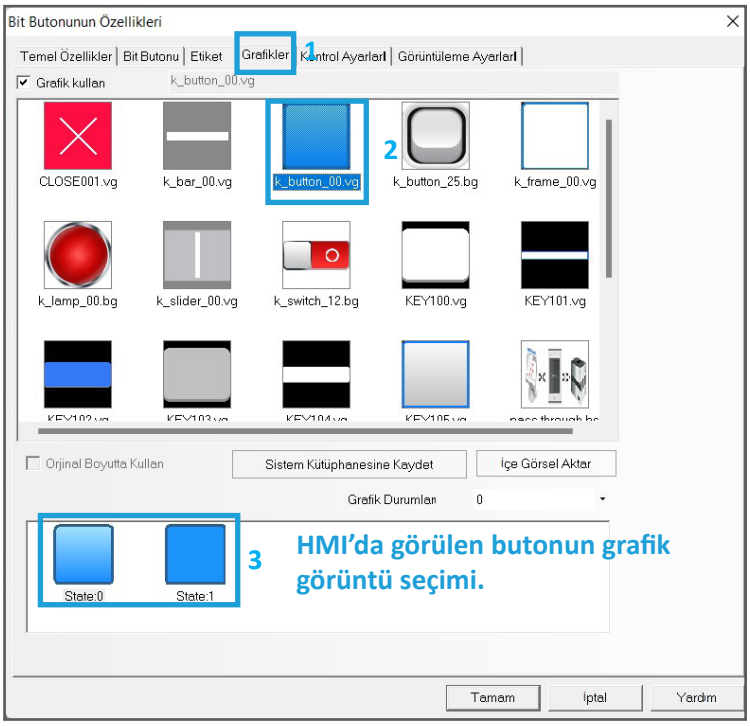


Set1	:	PLC'deki bilgiye 1 yaz.
Set0	:	PLC'deki bilgiye 0 yaz.
Toggle	:	PLC'deki bilgi 0 ise 1, 0 ise 1 yaz.
Reset	:	Butona basılı ise PLC'ye 1, basılmamış ise 0 yaz.

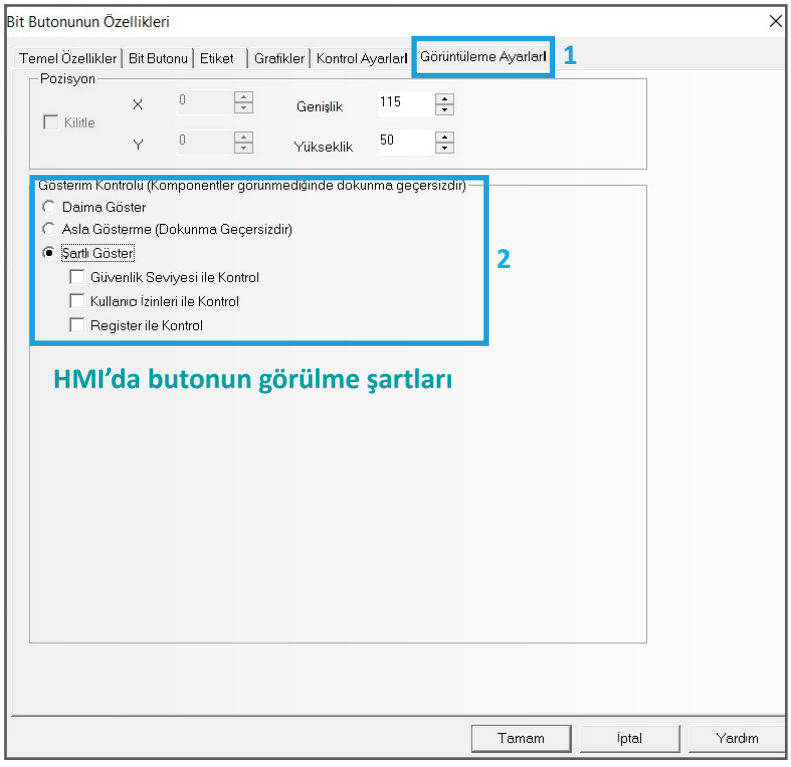
Görsel 3.61: Bit butonunun çalışma modları



Görsel 3.62: Bit butonunun etiket ayarları

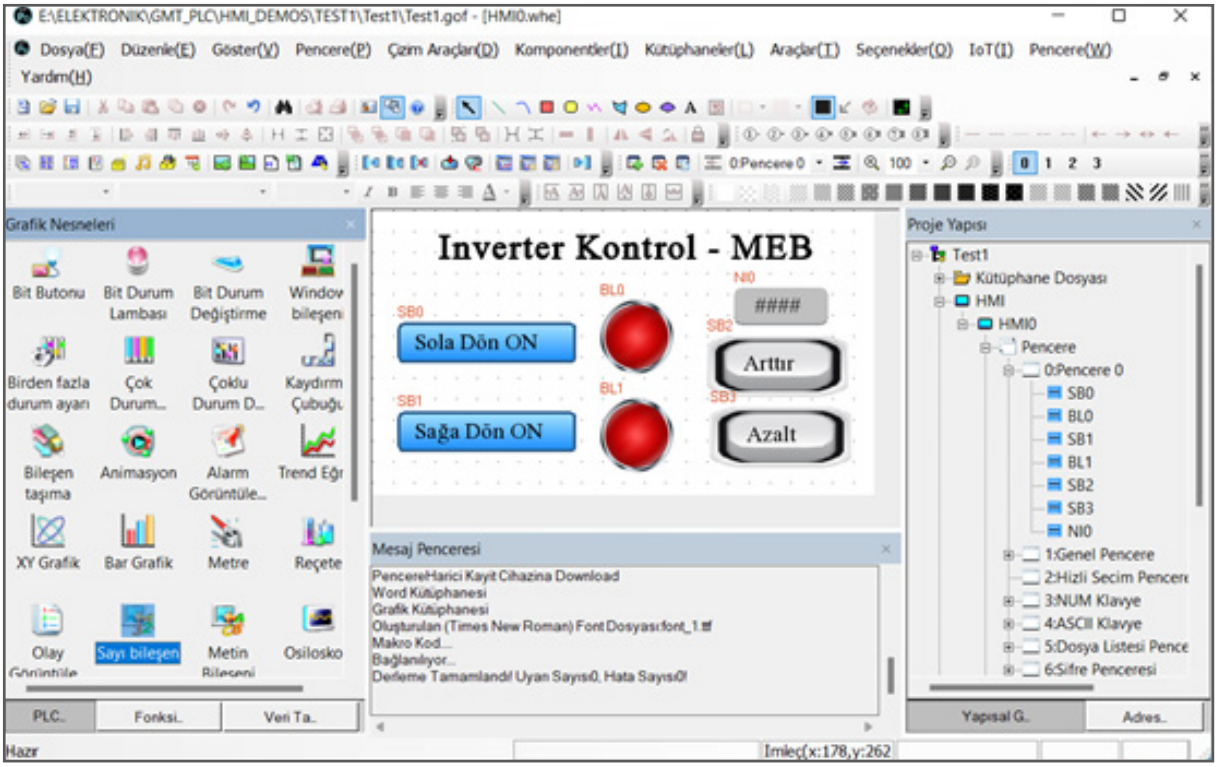


Görsel 3.63: Bit butonunun grafik görüntü ayarları

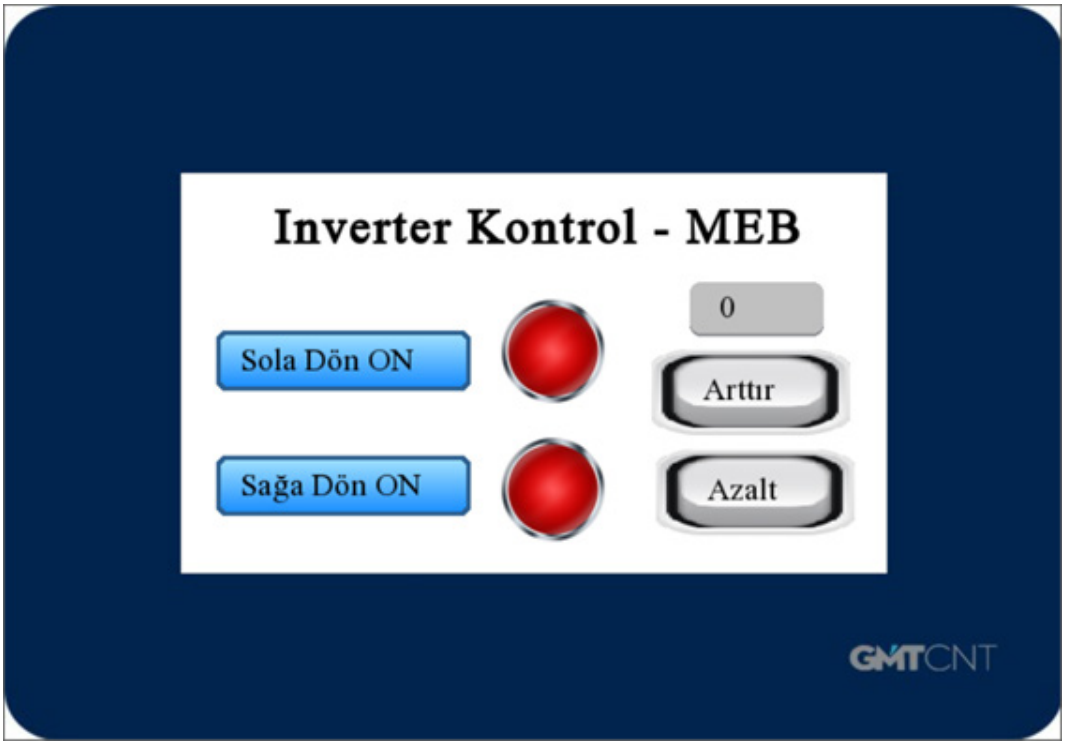


Görsel 3.64: Bit butonunun görüntülenme ayarları

PDesingerde kodlanan uygulama HMI'e download edildiğinde uygulama HMI üzerinden çalışır.



Görsel 3.65: Bileşen ayarlarından sonraki ekran görüntüsü



Görsel 3.66: Simülasyon ekran görüntüsü

ÖLÇME VE DEĞERLENDİRME

1. GLC PLC'lerde maksimum besleme gerilimi kaç volt olmalıdır?

A) 3.3
B) 5
C) 12
D) 24
E) 36

2. Aşağıdakilerden hangisi ethernet portu üzerinden yapılamaz?

A) Programlama
B) Link kurma
C) MODBUS TCP slave olarak kullanma
D) PLC'nin çektiği akımı ölçme
E) Sensör değerini okuma

3. Tüm GLC PLC modellerinde dijital giriş ve dijital çıkış sayıları kaçtır?

A) 8 giriş - 4 çıkış
B) 8 giriş - 8 çıkış
C) 9 giriş - 6 çıkış
D) 9 giriş - 16 çıkış
E) 9 giriş - 16 çıkış

4. Aşağıdakilerden hangisi GLC PLC modellerinde analog giriş tiplerinden biri değildir?

A) 0-10V
B) 0-2V
C) 0-20V
D) 0-20V
E) 4-20V

5. Aşağıdakilerden hangisi GLC PLC modellerinde analog çıkış tipidir?

A) 0-2mA
B) 0-10mA
C) 0-20mA
D) 0-20mA DC
E) 4-20mA DC

6. GLC PLC modellerinde analog çıkış değeri aralığı kaçtır?

A) 0-128
B) 0-255
C) 0-1024
D) 0-4096
E) 0-16383

7. MODBUS RTU protokolü ile en fazla kaç cihaz bağlanabilir?

A) 16
B) 32
C) 64
D) 255
E) 355

8. Ondalıklı bir işlem neticesi için tanımlanması gereken değişken tipi hangisidir?

A) Bit
B) Bool
C) Integer
D) Real
E) Word

9. PLC'de MODBUS haberleşme kullanıldığında değişken adreslerinden eksiltilecek sayı kaçtır?

A) 4000
B) 6400
C) 32000
D) 40000
E) 42000

10. Matematik 2 komutları bölümünde ileri matematik işlemlerini yapan fonksiyonlar kullanılır. **Bu nedenle bu bölümde tanımlanan operand tipleri hangisi olmalıdır?**

A) Bit
B) Bool
C) Integer
D) Real
E) Word

11. GMT PLC'de kullanılan zaman röleleri hangi zaman birimlerinde çalışır?

A) gün-saat-dakika-saniye-milisaniye
B) yıl-ay-gün-saat-dakika-saniye
C) ay-gün-saat-dakika-saniye
D) gün-saat-dakika-saniye
E) yıl-ay-dakika-saniye-salise

12. Bit operasyon komutlarında oparand (veri) tipleri ne olmalıdır?

A) Bit ya da bool
B) Word ya da real
C) Word ya da integer
D) Word ya da double word
E) Float ya da word

13. GLC-X96T (transistör) serisi PLC için maksimum anahtarlama frekansı değeri kaçtır?

A) 1KHz
B) 4KHz
C) 40KHz
D) 100KHz
E) 400KHz

14. 0-10V çıkışı elde etmek için PLC çıkışına bağlanması gereken direnç değeri kaç ohmdur?

A) 100
B) 220
C) 250
D) 330
E) 500

15. Hızlı pulse çıkışlarına bağlanan direnç değeri kaç ohmdur?

A) 220
B) 300
C) 330
D) 390
E) 500

16. PLC'deki ondalıklı bir sayı gönderilirken onaltılık olarak gönderilir. Onaltılık bu bilgi bir standarda göre tekrar ondalık hale getirilir. **Kullanılan bu standardın adı nedir?**

A) EIA-568
B) CAT5
C) CAT6
D) IEEE754
E) TSE

17. Aşağıdakilerden hangisi GMT PLC'de kullanılan zaman rölelerinden biri değildir?

A) Bırakmada Gecikme (Ters) Zaman Rölesi
B) Çekmede Gecikme (Düz) Zaman Rölesi
C) PWM Zaman Rölesi (Darbe Genişlik Modülasyonu)
D) Pulse zaman rölesi
E) Ters zaman rölesi

18. PLC'de kullanılan artan ve azalan sayaçlar hangi aralıklarda sayma işlemi yapabilirler?

A) -2.147.483.648 - 2.147.483.647
B) -32768 - 32767
C) 0 - 4096
D) 0 - 32767
E) 0 - 65535

19. NPN tipi bağlantıda COM ucuna bağlanması gereken kablo aşağıdakilerden hangisidir?

A) 5V kaynağının (+) hattı
B) 12V kaynağın (-) hattı
C) 12V kaynağın (+) hattı
D) 24V kaynağın (-) hattı
E) 24V kaynağın (+) hattı

20. PNP tipi bağlantıda COM ucuna bağlanması gereken kablo aşağıdakilerden hangisidir?

A) 5V kaynağının (+) hattı
B) 12V kaynağın (-) hattı
C) 12V kaynağın (+) hattı
D) 24V kaynağın (-) hattı
E) 24V kaynağın (+) hattı

21. Aşağıdakilerden hangisi reset butonun görevlerinden biri değildir?

A) PLC'yi "Run" konumundan "Stop" konumuna geçirmek.
B) "Stop" konumundan "Run" konumuna geçirmek.
C) PLC içerisindeki programı siler.
D) PLC kendini resetler ve ethernet bağlantı şifresini 1234 yapar.
E) Çalışmayan (kilitlenmiş) PLC'yi çalıştırmak

22. Motorun gideceği konum belli değil ise hangi hareket tipi kullanılır?

A) Mesafesiz artımsal (incremental)
B) Mutlak (absolute) hareket
C) Rastgele hareket
D) Nokta hareketi
E) Koordinatlı hareket

23. Eksen kurulum fonksiyon bloğunda üst ve alt anahtarların (switch) görevi nedir?

A) Motoru hareket ettirmek.
B) Motoru durdurmak.
C) PLC'yi durdurmak.
D) Eksenin ilk ve son noktalarındaki anahtarlara geldiğinde motor durur.
E) PLC'yi çalıştırmak.

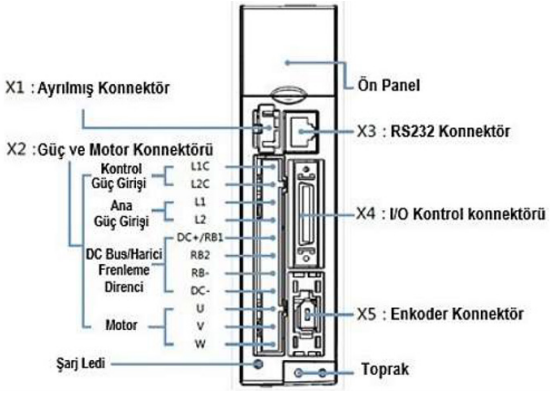
24. "Asenkron motorların frekans değişimine göre hızlarını, devir yön değişimlerini, durdurma ve çalıştırma işlemlerini yapar." **Tanımı aşağıdaki ifadelerden hangisine aittir?**

A) Ethernet
B) Güç kaynağı
C) Inverter
D) Motor
E) PLC

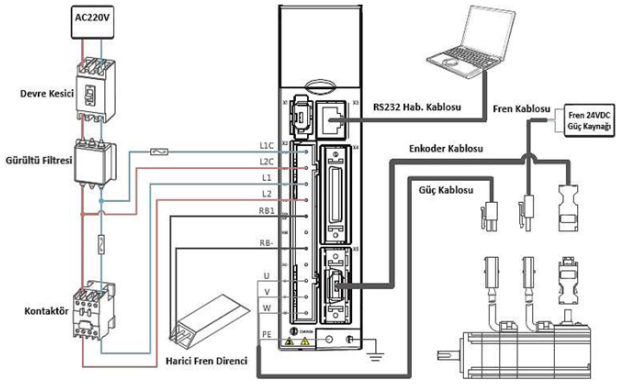
EKLER

SERVO SÜRÜCÜ ve MOTOR BAĞLANTILARI

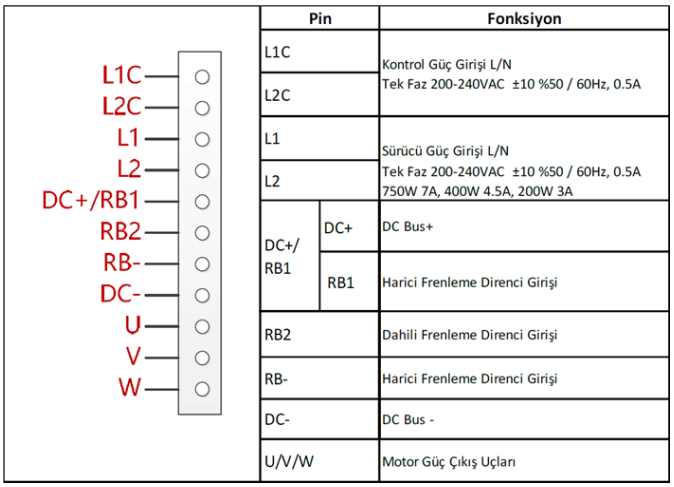
GSM60F-0200W-20B-30D-KLY MOTOR ID: 2.0 Servo sürücü 65536 pulse/tur şeklindedir. Motor milindeki hassasiyet 360/65536 derecedir.



Görsel 3.67: GSS3-2RS servo sürücünün görünümü

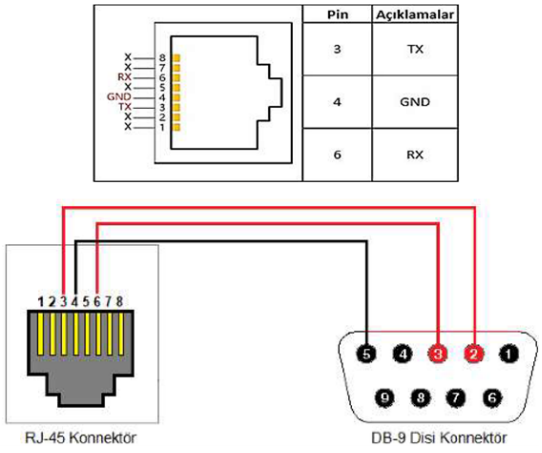


Görsel 3.68: Servo sürücü kablo bağlantısı

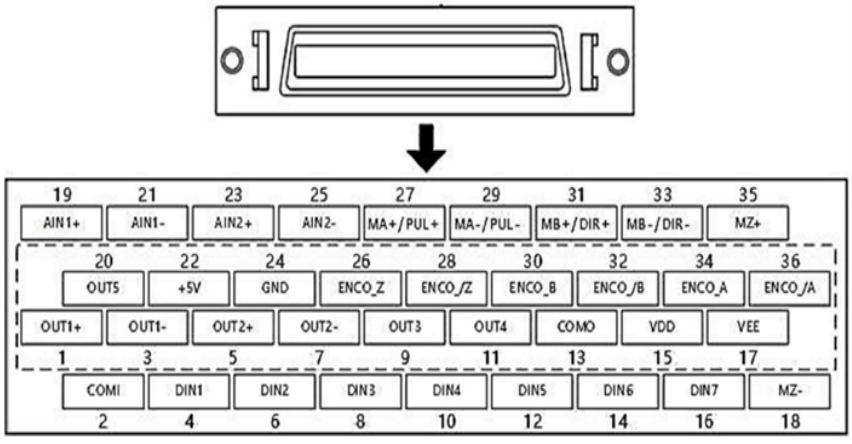


Pin		Fonksiyon
L1C		Kontrol Güç Girişi L/N Tek Faz 200-240VAC ±10 %50 / 60Hz, 0.5A
L2C		
L1		Sürücü Güç Girişi L/N Tek Faz 200-240VAC ±10 %50 / 60Hz, 0.5A 750W 7A, 400W 4.5A, 200W 3A
L2		
DC+/RB1	DC+	DC Bus+
	RB1	Harici Frenleme Direnci Girişi
RB2		Dahili Frenleme Direnci Girişi
RB-		Harici Frenleme Direnci Girişi
DC-		DC Bus -
U/V/W		Motor Güç Çıkış Uçları

Görsel 3.69: Güç ve motor X2 konnektör bağlantısı



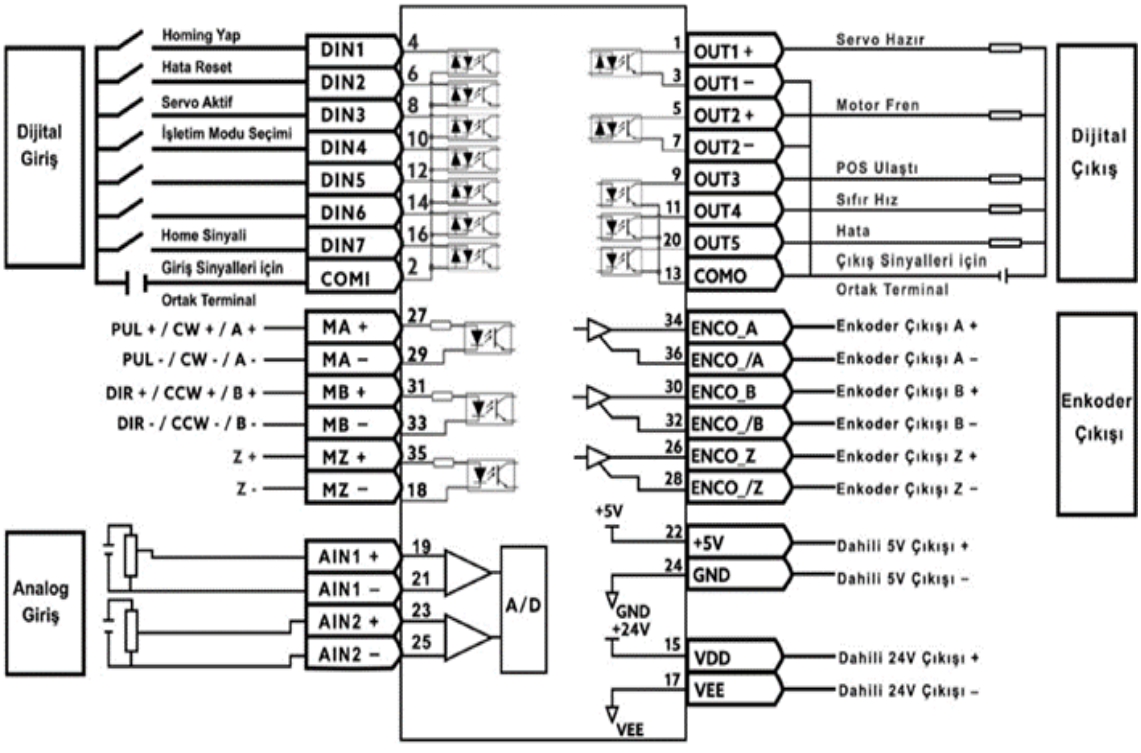
Görsel 3.70: RS232 seri port X3 konnektör bağlantısı



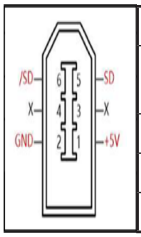
Görsel 3.71: Servo sürücü I/O X4 konnektör bağlantısı

Pin	Fonksiyon
DIN1-DIN7	Dijital Sinyal Girişleri VinH(aktif) : 12.5VDC - 30VDC VinL(pasif) : 0VDC – 5VDC Giriş Frekansı: <1kHz
COMI	Dijital girişler için ortak uç
OUT1+ / OUT1-	Dijital sinyal çıkışları Maksimum çıkış akımları: 100mA
OUT2+ / OUT2-	
OUT3/OUT4/OUT5	Dijital sinyal çıkışları Maksimum çıkış akımı: 20mA
COMO	OUT3, 4, 5 çıkışlar için ortak uç
MA+ (PUL+) / MA- (PUL-)	Puls Girişi Giriş gerilimi: 3.3V-24V Maksimum Frekans: 500KHz
MB+ (DIR+) / MB- (DIR-)	
MZ+ / MZ-	
ENCO_A+ / ENCO_A-	Enkoder Çıkışı Gerilim: Voh=3.4V, Vol=0.2V Maksimum akım: ±20mA, Maksimum frekans: 5MHz
ENCO_B+ / ENCO_B-	
ENCO_Z+ / ENCO_Z-	
AIN1+ / AIN1- & AIN2+ / AIN2-	Analog giriş Çözünürlük: 12 bit, Giriş direnci: 350 KΩ Giriş gerilim Aralığı: -10V ~ +10V
+5V / GND	5VDC güç kaynağı çıkışı Maksimum akım: 100mA
VDD/VEE	24VDC güç kaynağı çıkışı Gerilim Aralığı: 24VDC ± 20%, maksimum akım: 300 mA

Görsel 3.72: Servo sürücü X4 konnektörü I/O arayüz PIN açıklamaları



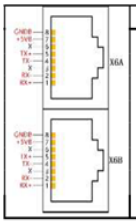
Görsel 3.73: Servo sürücü I/O arayüz bağlantıları



Pin	Açıklama	Fonksiyon
1	+5V	Enkoder için 5VDC beslemesi
2	GND	Signal ground (-5V)
5	SD	Seri veri sinyali
6	/SD	Seri veri sinyali

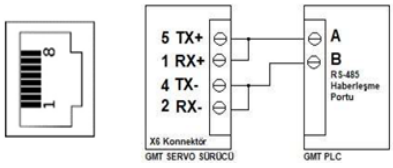
6 pin konnektör	Kablo rengi		Açıklama
Pin 1	Kırmızı		+5V
Pin 2	Siyah		GND
Pin 3	Kahverengi		BAT+
Pin 4	Mavi		BAT-
Pin 5	Sarı		SD
Pin 6	Yeşil		K/SD
Dış Kaplama	Kablo koruma		Kablo Koruma

Görsel 3.74: X5 enkoder giriş konnektörü

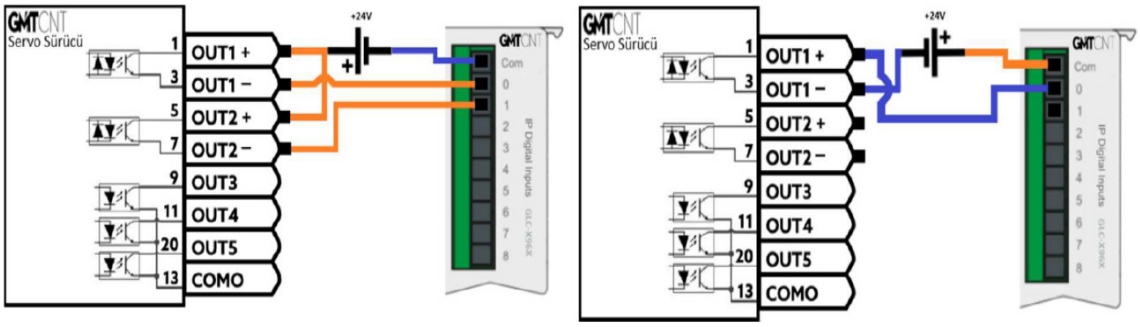


Port İsmi	Port Tipi	Pin	Pin Adı	Açıklama
RS485 Haberleşme Arayüzü	RJ45 Netport Dişi Soket	1	RX+	Gelen Veri +
		2	RX-	Gelen Veri -
		3	NC	-
		4	TX-	Gönderilen Veri -
		5	TX+	Gönderilen Veri +
		6	NC	-
		7	+5VB	+5V Çıkış
		8	GNDB	Ground

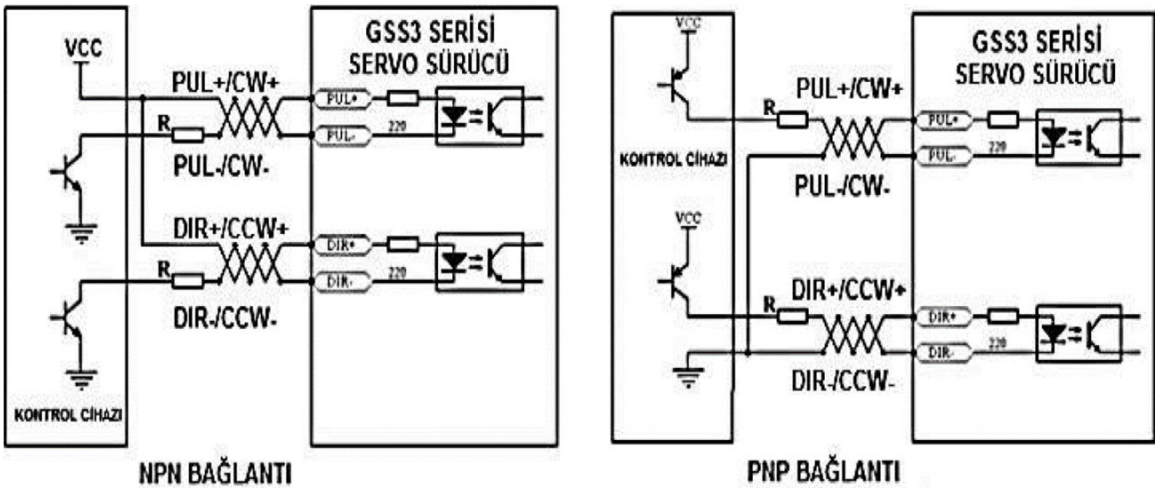
***GSS3-XRS, RS485 Haberleşme desteği sunan Servo Sürücü için kablo bağlantısı; RX+ ve TX+ uçları birleştirilerek A(+) bağlantı noktasına, RX- ve TX- uçları birleştirilerek B(-) bağlantı noktasına, şeklinde yapılmalıdır. Fabrika Değeri: 19200-8-n-1**



Görsel 3.75: X6 haberleşme konnektörü



Görsel 3.76: Servo sürücü dijital çıkış bağlantı şekilleri



Görsel 3.77: Servo sürücü PULSE ve DIR bağlantı şekilleri

CEVAP ANAHTARI

1.	D	13.	E
2.	D	14.	E
3.	C	15.	D
4.	B	16.	D
5.	C	17.	E
6.	E	18.	A
7.	D	19.	E
8.	D	20.	D
9.	D	21.	C
10.	D	22.	A
11.	A	23.	D
12.	D	24.	C

KAYNAKÇA

- Ders Bilgi Formu, Millî Eğitim Bakanlığı Mesleki ve Teknik Eğitim Endüstriyel OtomasyonTeknolojileri Alanı, Seçmeli İleri PLC Uygulamaları, Ankara, 2020

GENEL AĞ KAYNAKÇASI

- http://docwiki.embarcadero.com/PlatformStatus/en/Main_Page (12.11.2021)
- http://www.gmtcontrol.com/tr/
- http://meslek.eba.gov.tr/dokumanlar/DELPHI_GMT_PLC_UYGULAMALAR.rar
- https://sozluk.gov.tr/

GÖRSEL KAYNAKÇA

Kitapta kullanılacak karekod linki :

http://kitap.eba.gov.tr/karekod/Kaynak.php?KOD=2955

NOTLARIM

Neler Öğrendim?

NOTLARIM

Neler Öğrendim?