SIEMENS

SIMATIC

STEP 7 V5.3 ile çalışma

Başlarken

Bu elkitabı **6ES7810-4CA07-8BW0** sipariş numaralı belge paketinin bir parçasıdır.

Basım 01/2004
A5E00261403-01

Güvenlik kuralları

Bu elkitabında kişisel güvenliği temin etmek ve aynı zamanda ürünleri ve bağlı ekipmanı hasarlardan korumak amacıyla hazırlanmış hatırlatmalar vardır. Bu hatırlatmalar aşağıda gösterilen sembollerle vurgulanmış ve aşağıdaki metinlerin ciddiyetine göre tasnif edilmiştir:

Tehlike
Uygun önlemler alınmaması halinde ölüme, ciddi bedensel zarar veya önemli mali zarara yol açabileceğini gösterir.

İhtar
Uygun önlemler alınmaması halinde ölüme, ciddi bedensel zarar veya önemli mali zarara yol açabileceğini gösterir.

İkaz
Uygun önlemler alınmaması halinde küçük bedensel zarara yol açabileceğini gösterir.

Uyarı
Uygun önlemler alınmaması halinde mali zarara yol açabileceğini gösterir.

Dikkat
Ürün, ürünün kullanılması veya belgelerin önemli bir bölümü hakkında özellikle önemli bilgilere dikkatinizi çeker.

Vasıflı Personel

Bu ekipmanı kurmak ve üzerinde çalışmak üzere sadece **vasıflı personele** izin verilmelidir. Kalifiye personel yerleşik güvenlik uygulamaları ve standartlarına uygun olarak ekipman ve sistemleri işletmek, topraklamak, devreleri etiketlemek için yetkili kişiler olarak tanımlanır.

Doğru kullanma

Aşağıdakileri not edin:

Uyarı
Bu cihaz ve bileşenleri katalog veya teknik açıklamada belirtilen uygulamalar için ve sadece başka imalatçıların Siemens tarafından onaylanan veya tavsiye edilen cihazları ve bileşenleri ile bağlantı halinde kullanılabilir.

Bu ürün doğru olarak taşınır, depolanır, ayarlanır ve kurulursa ve tavsiye edilen şekilde çalıştırılır ve bakımı yapılırsa doğru ve güvenli şekilde çalışabilir.

Ticari Markalar

SIMATIC®, SIMATIC HMI® ve SIMATIC NET® markaları SIEMENS AG'nin tescilli markalarıdır.

Ticari markaların anıldığı bu belgedeki diğer isimleri kendi amaçları için kullanan üçüncü kişiler ticari marka sahiplerinin haklarına tecavüz edebilirler.

Sorumluluğun Reddi

Bu elkitabının içeriğinin açıklanan donanım ve yazılıma uygunluğu tarafımızdan kontrol edilmiştir. Sapmaların tamamıyla giderilmesi olanaksız olduğundan tam uyumu garanti edemeyiz. Ancak bu elkitabındaki veriler düzenli olarak gözden geçirilir ve gerekli değişiklikler sonraki yayımlara dahil edilir. Düzeltme önerileri memnuniyetle karşılanacaktır.

Siemens AG
Bereich Automation and Drives
Geschaeftsgebiet Industrial Automation Systems
Postfach 4848, D- 90327 Nuernberg

Siemens Aktiengesellschaft

A5E00261403-01

STEP 7’ye Hoş Geldiniz...

...SIMATIC S7-300/400 istasyonları için Sıralama Mantığında, Fonksiyon Blok Şeması veya İfade Listesinde programlanabilir kontrol programları oluşturmak için SIMATIC standart yazılımı.

Bu Başlarken Elkitabı Hakkında

Bu elkitabında SIMATIC STEP 7’nin temellerini anlamaya başlayacaksınız. Size en önemli ekran diyalog kutularını ve hemen her bölüme başlayabileceğiniz şekilde oluşturulan uygulamalı alıştırmaları kullanarak izlenecek yöntemleri size göstereceğiz.

Her kısım iki bölüme ayrılmıştır: gri olarak işaretli tanımlayıcı bölüm ve yeşil işaretli işleme yönelik bölüm. Talimatlar yeşil marjlı bir okla başlar ve birkaç sayfaya kadar uzayarak bir nokta ile ve ilgili mevzuların bulunduğu bir kutu ile son bulur.

Fare ile çalışma alışkanlığı, pencere kullanma, çekme menüler v.s. yararlı olacak ve programlanabilir mantık kontrolü temel prensiplerini tercihen bilmeniz gerekli olacaktır.

STEP 7 eğitim kursları size bu Başlarken Elkitabının içeriği hakkında derinliğine bilgi sağlar ve STEP 7 ile ne kadar bütün otomasyon çözümleri yaratılabileceğini öğretir.

“Başlarken Elkitabı” ile çalışırken ihtiyaç duyacağınız Gereksinimler

Bu Başlarken elkitabında STEP 7’nin uygulamalı alıştırmalarını yapabilmek için aşağıdakilere ihtiyacınız vardır:

- Siemens programlama cihazı veya PC
- STEP 7 yazılım paketi ve ilgili lisans anahtarı
- SIMATIC S7-300 veya S7-400 programlanabilir kontrol edici
  (Bölüm 7 "Programın İndirilmesi ve Hatalarının Giderilmesi ").

STEP 7 hakkında Ek Belgeler

- STEP 7 Temel Bilgiler
- STEP 7 Referans Bilgileri

STEP 7’yı kurduktan sonra elektronik elkitaplarını **Simatic > Documentation** (*Simatic* > *Belgeler*) altında bulacaksınız ya da başka bir seçenek olarak herhangi Siemens satış merkezinden sipariş edebilirsiniz. Elkitaplarındaki tüm bilgiler

STEP 7’de çevrimiçi yardım sırasında aranabilir.

İyi eğlenceler ve iyi şanslar!

SIEMENS AG

İçindekiler

1. STEP 7'ye Giriş

1.1 Ne Öğreneceksiniz

Step 7 ile Sıralama Mantığında, İfade Listesinde veya Fonksiyon Blok Şemasında program yapmanın ne kadar kolay olduğunu pratik alıştırmaları kullanarak size göstereceğiz.

Her bölümdeki ayrıntılı talimat STEP 7'yi kullanabileceğiniz birçok yolu size gösterecektir.

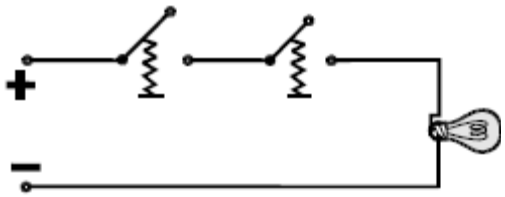
İkili (binary) Mantığı ile Programlama

2 - 7nci bölümlerde binary mantığında bir program yapacaksınız. Programlanmış mantık işlemlerini kullanarak, CPU'nuzun girdileri ve çıktılarına (varsa) başvurabilirsiniz.

Başlarken Elkitabındaki programlama örnekleri, başka şeylerle birlikte, üç temel binary mantığı işlemine dayanır.

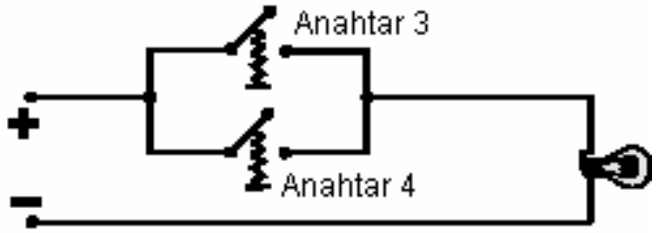
Daha sonra programlayacağınız ilk binary mantığı işlemi AND (VE) fonksiyonudur. AND (*VE*) fonksiyonu iki anahtar kullanan bir devre şemasında açıklanabilir.

Anahtar 1 Anahtar 2



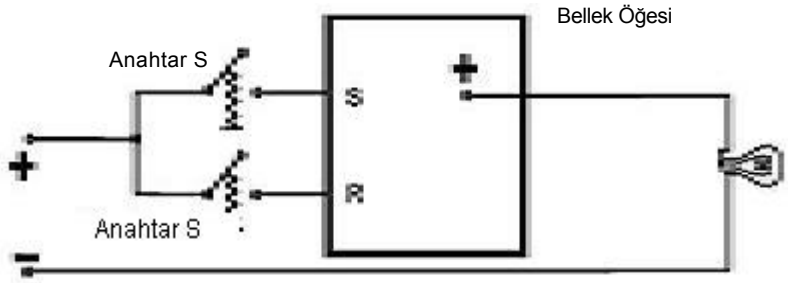
Eğer Anahtar 1 **ve (*and*)** basılırsa lamba yanar.

İkinci binary mantığı işlemi OR (*VEYA*) fonksiyonudur. Veya (*OR*) fonksiyonu da bir devre şemasında gösterilir.



Anahtar 3 veya Anahtar 4'ten **birine** basılırsa lamba yanar

Üçüncü binary mantığı işlemi bellek öğesidir. SR fonksiyonu belli voltaj durumunda bir devre şeması içinde tepki verir ve bunları aynı şekilde geçer.

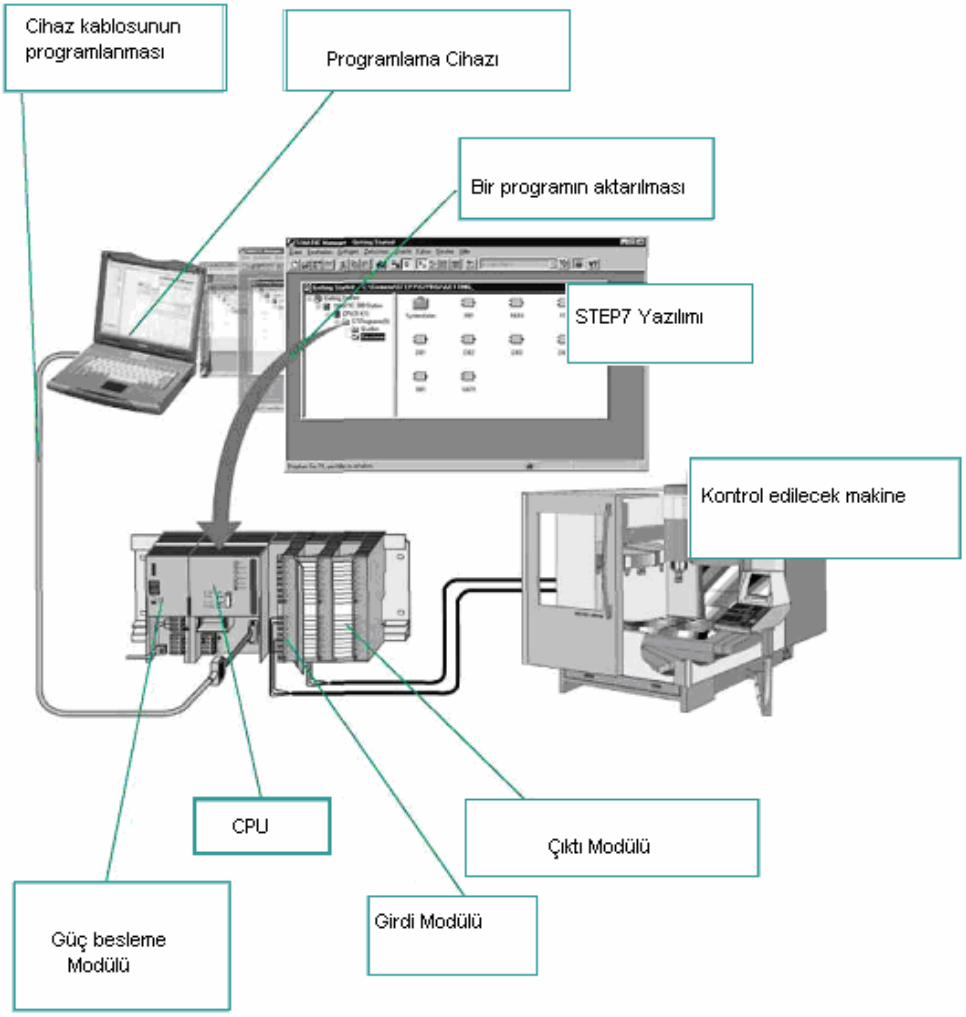


S anahtarına basılırsa lamba yanar ve
R anahtarına basılıncaya kadar öyle kalır.

1.2 Donanım ve Yazılımın Birleştirilmesi

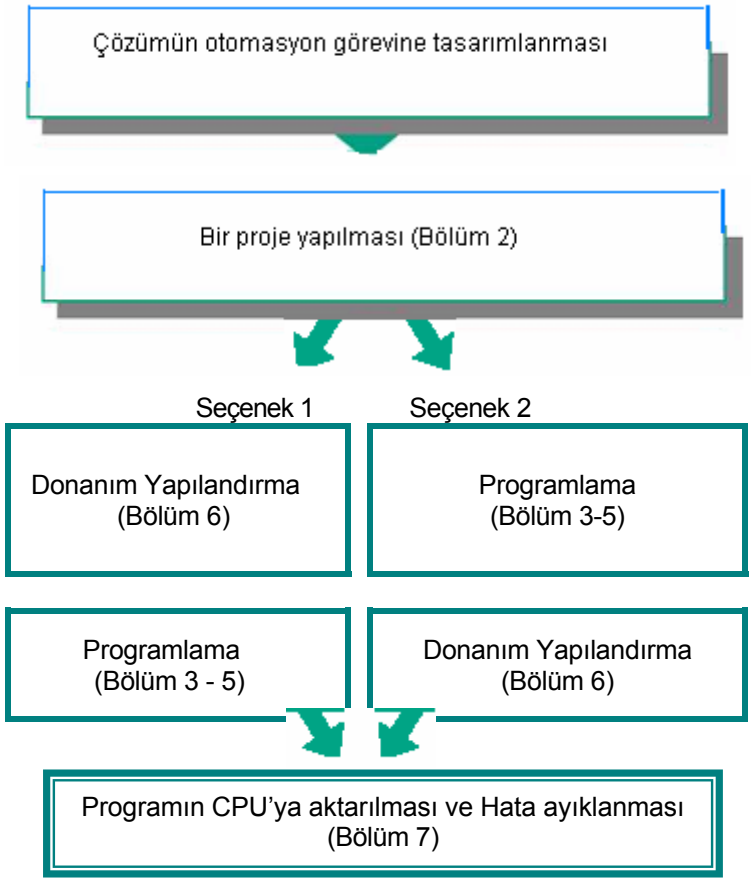
Bir proje içinde STEP 7yazılımını kullanarak kendi S7 programınızı yapabilirsiniz S7 programlanabilir kontrol edici güç besleme ünitesi, CPU ile girdi ve çıktı modüllerinden (I/O modülleri).

Programlanabilir mantıksal kontrol edici (PLC) makinenizi S/ programı ile izler ve kontrol eder. IS7 programında /O modüllerine adresleri üzerinden başvuru yapılır.



1.3 STEP 7'yi Kullanmanın Temel Yöntemi

Bir proje oluşturmadan önce STEP 7projelerinin farklı sıralarla yapılabileceğini bilmelisiniz.



Birçok girdisi ve çıktısı olan kapsamlı programlar yapıyorsanız önce donanımı yapılandırmanızı tavsiye ederiz. Bunun yararı Donanım Yapılandırma Düzenleyicisi STEP 7'nin olası adresleri göstermesidir.

İkinci şıkkı seçerseniz, her adresi seçtiğiniz bileşenlere göre kendiniz belirlemek zorundasınız.ve bu adresleri STEP 7 yardımıyla çağıramazsınız.

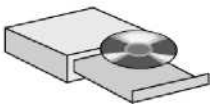
Donanım yapılandırmasında siz sadece adresleri tanımlamasınız fakat modüllerin parametrelerini ve özelliklerini de değiştirebilirsiniz. Örneğin birkaç CPU'yu çalıştırmak isterseniz CPU'ların MPI adreslerini birbirlerine uydurmak zorundasınız.

Başlarken Elkitabında sadece az sayıda girdi ve çıktı kullandığımız için, şimdilik donanım yapılandırmasını geçiyoruz ve programlama ile başlıyoruz.

1.4 STEP 7'nin Kurulması

Donanımı programlamak mı ya da yapılandırmakla mı başlamak istediğinize aldırmadan, önce STEP 7'yi kurmak zorundasınız. SIMATIC programlama cihazı kullanıyorsanız STEP 7 zaten kuruludur.

STEP 7 yazılımını, daha önce STEP 7'nin bir sürümü kurulu olmayan bir programlama cihazına veya PC'ye kurduğunuz zaman, yazılım ve donanım gereksinimlerine dikkat edin. Bunları STEP 7'CD'sinde **<Drive>:\STEP7\Disk1** altında Readme.wri dosyasında bulabilirsiniz.



Önce STEP 7'yi kurmanız gerekirse STEP 7 CD'sini CD sürücüye takın. Kurulum programı otomatik olarak başlar. Ekrandaki talimatlara uyun.

Kurulum otomatik olarak başlamazsa, kurulum programını CD-ROM'da <Drive>:\STEP 7 \Disk1\setup.exe altında da bulabilirsiniz.

SIMATIC Yöneticisi

Kurulum tamamlandıktan ve bilgisayarınızı yeniden başlattıktan sonra, "SIMATIC Yöneticisi" simgesi Windows masaüstünde gözükür.

Kurulumu takiben "SIMATIC Yöneticisi" simgesine çift tıklarsanız, STEP 7 Sihirbazı otomatik olarak başlayacaktır.

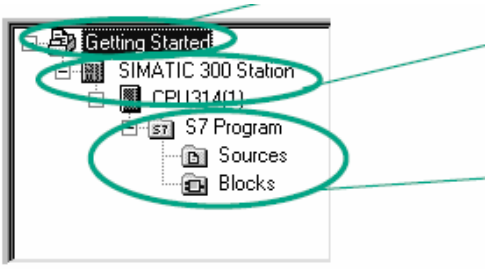
Kuruluş hakkında daha fazla bilgiyi STEP 7 CD'sinde **<Drive>:\STEP 7 \Disk1\Readme.wri** altındaki Readme.wri dosyasında bulabilirsiniz.

2 SIMATIC Yöneticisi

2.1 SIMATIC Yöneticisinin Başlatılması ve Proje Oluşturulması

SIMATIC Yöneticisi STEP 7 başlatıldığı zaman etkinleşen orta penceredir. Varsayılan ayar bir STEP 7 projesi yaparken sizi destekleyen STEP 7 Sihirbazı ile başlar. Proje yapısı tüm verileri ve programları düzenli tutmak için kullanılır.

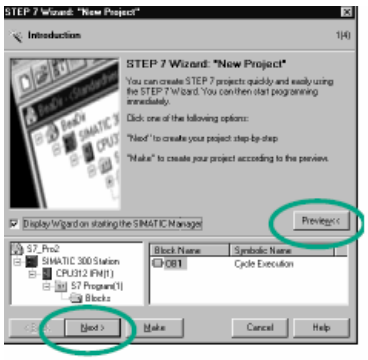
Proje içinde veriler hiyerarşik yapı içindeki cisimler biçiminde saklanır



SIMATIC istasyonu ve CPU donanımın yapılanmasını ve parametre verilerini içerir

S7 programı makineyi kontrol etmek için gerekli olan tüm program bloklarını kapsar

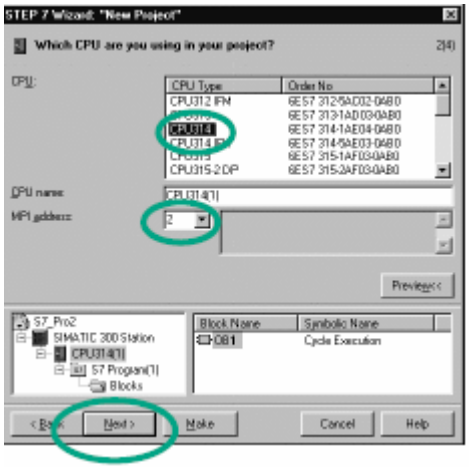
SIMATIC Yöneticisi



Windows masa üstündeki **SIMATIC Yöneticisi** simgesine çift tıklayın, sonra sihirbaz otomatik olarak başlamazsa **File > Wizard "New Project"** (*Dosya > Sihirbaz "Yeni Proje"*) menü komutunu seçin.

Yapılmakta olan proje yapısına **Preview** (*Önizleme*)'de proje geçiş yapabilir ve geri dönebilirsiniz.

Sonraki diyalog kutusuna geçmek için **Next** (*Sonraki*) üzerine tıklayın.

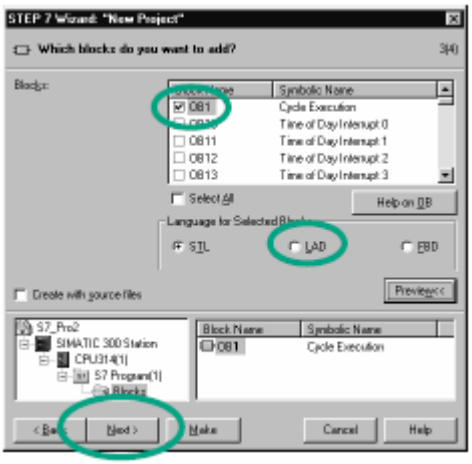


"Başlarken" örnek projesi için CPU 314'ü seçin. Örnek o şekilde yapılmıştır ki siz her zaman size verilebilen CPU'yu seçebilirsiniz.

MPI adresi için varsayılan ayar 2'dir.

Ayarı onaylamak için **Next** (*Sonraki*)'ni tıklayın ve sonraki diyalog kutusuna geçin.

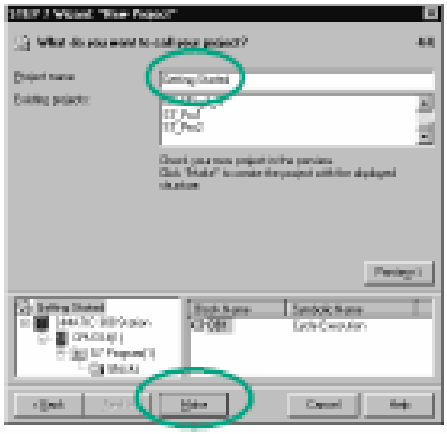
Her CPU'nun belli özellikleri vardır; örneğin, bellek yapılandırması veya adress bölgeleri bakımından. Bu nedenle programlamaya başlamadan önce CPU'yu seçmek zorundasınız.CPU'nuzun programlama cihazınız veya PC'nizle iletişim kurabilmeniz için MPI adresi (çok noktalı arabirim) gereklidir.



Organizasyon bloğu **OB1**'I seçin (önceden seçilmemişse).

Programlama dillerinden birini seçin: Sıralama Mantığı **(LAD),** İfade Listesi **(STL),** veya Fonksiyon Blok Şeması **(FBD).**
Next (*Sonraki*) ile seçiminizi onaylayın.

OB1 en yüksek programlama düzeyidir ve S7 programındaki başka blokları organize eder.
Programlama dilini daha sonraki bir tarihte değiştirebilirsiniz.



"Proje adı" alanında önerilen ismi seçmek için çift tıklayın ve "Başlarken" ile üzerine kaydedin.

Yeni projenizi ön izlemeye göre üretmek için **Make** (*Yap*) tuşuna tıklayın.

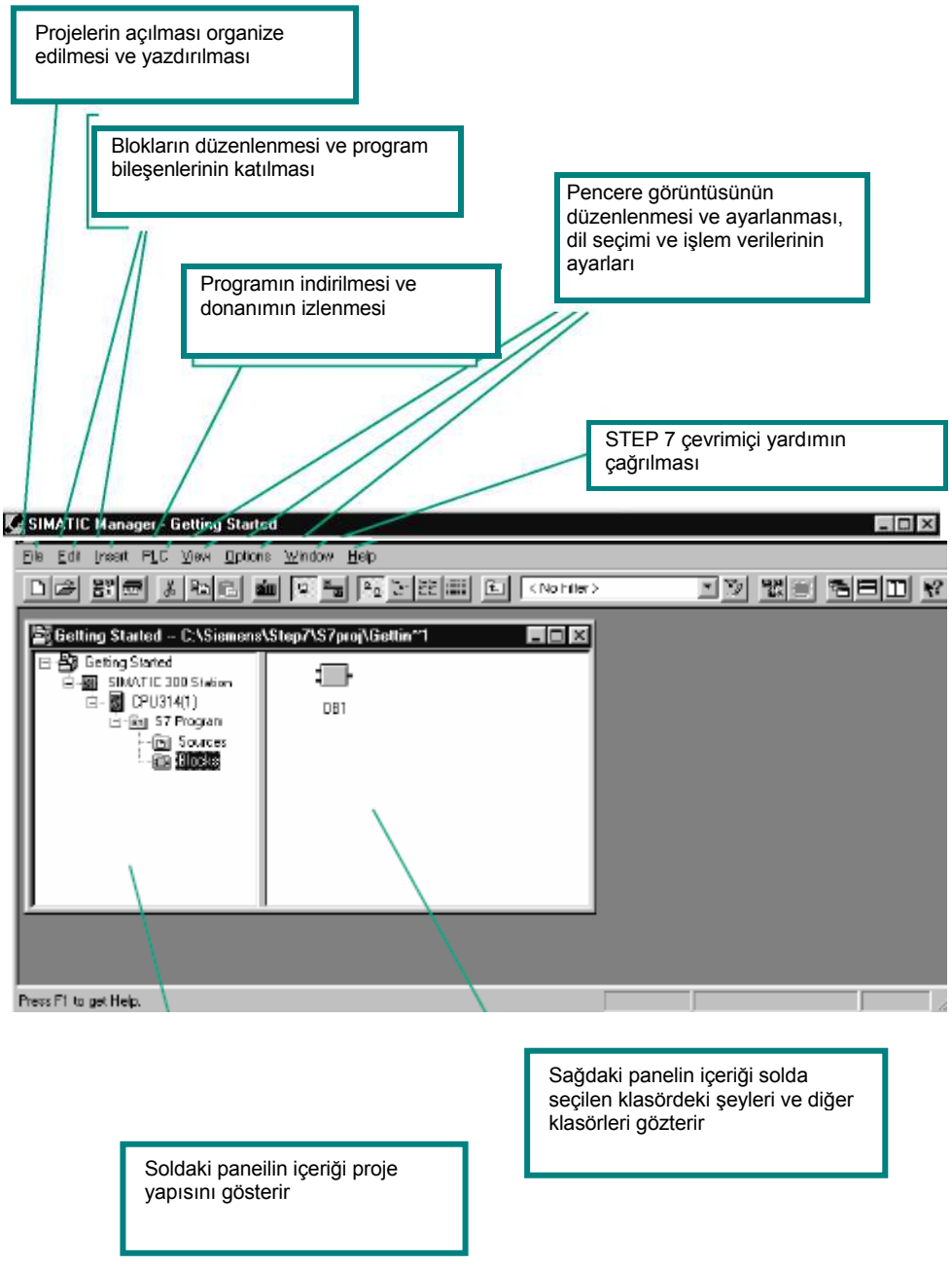
Make (*Yap*) tuşuna tıkladığınız zaman SIMATIC Yöneticisi yaptığınız projenin "Başlarken" penceresi ile birlikte açılır. Önümüzdeki sayfalarda yaptığınız dosyalar ve klasörlerin neye yaradığını ve onlarla etkin olarak nasıl çalışacağınızı göstereceğiz.

Programın her başlatılışında STEP 7 Sihirbazı etkinleşir. Bu varsayılan ayarın etkinliğini Sihirbazın ilk diyalog kutusunda kaldırabilirsiniz. Ancak, STEP 7 Sihirbazı olmadan projeler yaparsanız proje içinde her dizini kendiniz oluşturmalısınız.

"Projenin Kurulması ve Düzenlenmesi" başlığındaki **Help > Contents** (*Yardım* > *İçindekiler*) altında daha fazla bilgi bulabilirsiniz

2.2 SIMATIC Yöneticisinde Proje Yapısı ve Çevrimiçi Yardımın Aranması

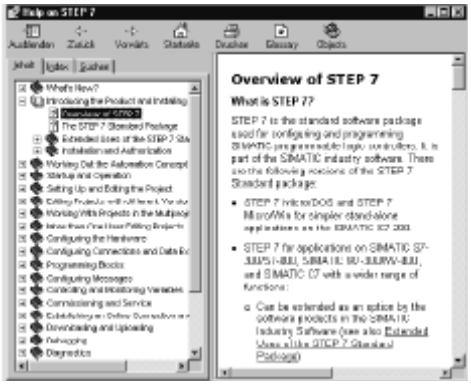
STEP 7 Sihirbazı kapanır kapanmaz, açılan “Başlarken” penceresi ile birlikte SIMATIC Yöneticisi görünür. Buradan tüm STEP 7 fonksiyonlarını ve pencerelerini başlatabilirsiniz.



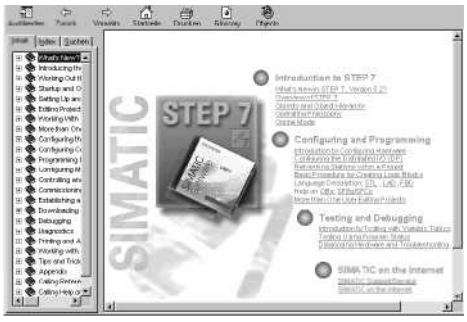
STEP 7 Yardımının aranması

F1

Seçenek 1:
İmleci herhangi bir komut üzerine getirin ve **F1** tuşuna basın. Seçilen komut menüsünün içeriğine duyarlı yardımı gelecektir.



Seçenek 2:
STEP 7 çevrimiçi yardım açmak için menüyü kullanın.
Çeşitli yardım başlıkları bulunan içindekiler sayfası sol alt panelde, seçilen başlıklar sağ panelde gösterilir.
İçindekiler listesindeki + işaretini tıklayarak istediğiniz konuda dolaşın.
Aynı zamanda seçilen başlıkların içeriği sağ panelde gösterilir.
Index (*Dizin*) ve **Find** (*Bul*)'u kullanarak, arama serilerine girebilir ve istediğiniz özel konuya bakabilirsiniz.



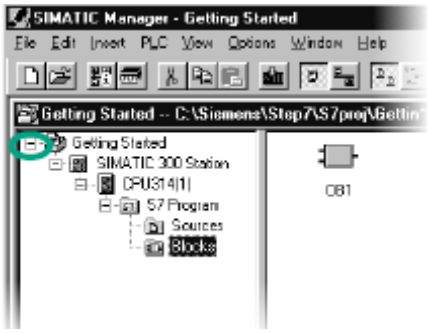
Seçenek 3:
STEP 7 Çevrimiçi Yardım'da "Başlat sayfası" simgesini tıklayıp bilgi kapısından girebilirisiniz. Bu kapı Çevrimiçi Yardımın:

- STEP 7'ye başlarken
- Yapılandırma ve programlama
- Test etme ve hata ayıklama
- İnternette SIMATIC

Gibi ana konularına kolay geçiş sağlar.

Seçenek 4:
Farenizi bir yardım imleci haline getirmek için araç çubuğundaki soru işaretine tıklayın.
Daha sonra belli bir konuya tıkladığınız zaman çevrimiçi yardım etkinleşir.

Proje Yapısı içinde Gezinme



Biraz önce hazırladığınız proje seçilen S7 istasyonunda ve CPU'da gösterilir.
Bir klasörü açmak veya kapamak için + veya – işaretine tıklayın.

Daha sonra sağ panelde gösterilen sembolleri tıklayarak başka fonksiyonları başlatabilirsiniz.

S7 Program (1) klasörünü tıklayın. Tüm gerekli program bileşenlerini içerir.

Adreslere sembolik isimler vermek için bölüm 3'teki sembol bileşenini kullanacaksınız.

Kaynak dosyalar bileşeni kaynak programları saklamak için kullanılır. Başlarken Elkitabında bunlardan bahsedilmez.

Blocks (*Bloklar*) klasörüne tıklayın. Bu klasörde daha önce yaptığınız **OB1** ve daha sonraki tüm bloklar bulunur.

Buradan, bölüm 4 ve 5'te, Sıralama Mantığı, İfade Listesi, ya da Fonksiyon Blok Şemasında programlamaya başlayacaksınız.

SIMATIC 300 İstasyon klasörünü tıklayın. Tüm donanım bağlantılı proje verileri burada saklanır.

Programlanabilir kontrol edicinizin parametrelerini belirlemek için Bölüm 6'daki donanım bileşenini kullanacaksınız.

Otomasyon göreviniz için daha fazla SIMATIC yazılımına, örneğin seçmeli paketlere, PLCSM (donanım simülasyon programı) veya S7 Graph (grafik programlama diline) ihtiyacınız olursa bunlar da STEP 7 ile bütünleşmiştir. SIMATIC Yöneticisini kullanarak, örneğin, doğrudan S/ Grafik fonksiyon bloğu gibi ilgili konuları açabilirsiniz.

Help > Contents (*Yardım > İçindekiler*) altında "Otomasyon Kavramının Hazırlanması" ve "Program Yapısını Tasarlamanın Esasları" konularında bulabilirsiniz.

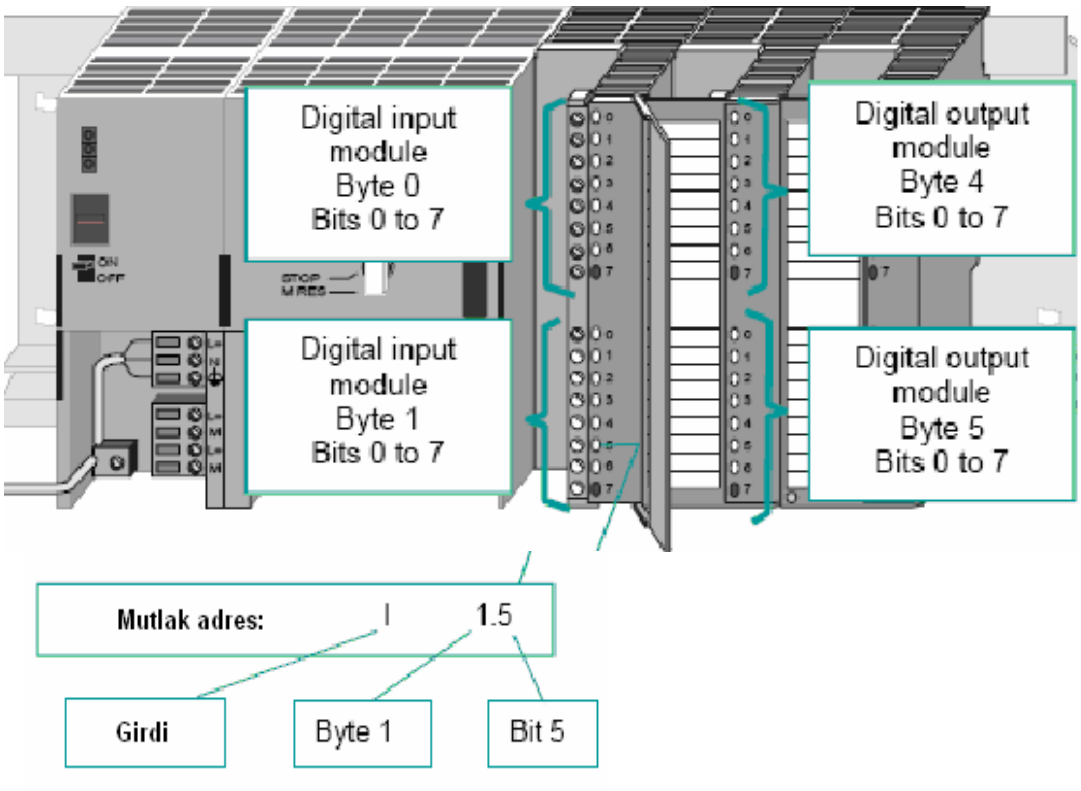
Daha fazla bilgiyi SIMATIC seçmeli paketler ST 70 katalogunda "Eksiksiz Entegre Otomasyon Bileşenlerinde bulabilirsiniz.

3 Sembollerle Programlama

3.1 Kesin Adresler

Her girdinin ve her çıktının donanım yapılandırmasında önceden belirlenen bir kesin adresi vardır. Bu adres doğrudan yani kesin olarak belirlenir.

Kesin adres yerine seçeceğiniz herhangi bir sembolik isim konabilir.



S7 programınızda birçok girdi ve çıktıyı yönetmek zorunda değilseniz sadece mutlak programlama kullanmalısınız.

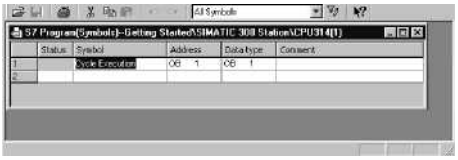
3.2 Sembolik Programlama

Sembol tablosunda, programınızda daha sonra başvuracağınız tüm kesin adreslere bir isim ve veri tipi, örneğin I girdisi için 0.1 sembolik isim anahtarı 1 belirlersiniz. Bu isimler programın tüm parçalarına uygulanır ve global değişkenler olarak bilinirler.

Sembolik programlama kullanarak, yaptığınız S7 programının okunabilirliğini önemli derecede geliştirebilirsiniz.

Sembol Düzenleyici ile Çalışma

"Başlarken" proje penceresinde, **S7 Program (1)**'a kadar gezinin ve **Symobols** (*Semboller*) bileşenini açmak için çift tıklayın.



Sembol tablonuz şu anda sadece önceden belirlenen organizasyon bloğu OB1'den oluşmaktadır.

	Status	Symbol	Address	Data type
1		Cycle Execution	OB 1	OB 1
2				

Örneğimiz için **Cycle Execution** (*Çevrim Uygulama*) üzerine tıklayın ve "Ana Program" ile üzerine kaydedin.

	Status	Symbol	Address	Data type
1		Main Program	OB 1	OB 1
2		Green Light	Q 4.0	BOOL

Sıra 2'de "Green Light (*Yeşil Işık*)" ve Q 4.0"' girin. Veri tipi otomatik olarak eklenir.

Comment

Sembol hakkında bir yorum eklemek için sıra 1 ve 2'nin yorum sütununa tıklayın. Bir sıradaki girdilerinizi **Enter**'a basarak tamamlayın.

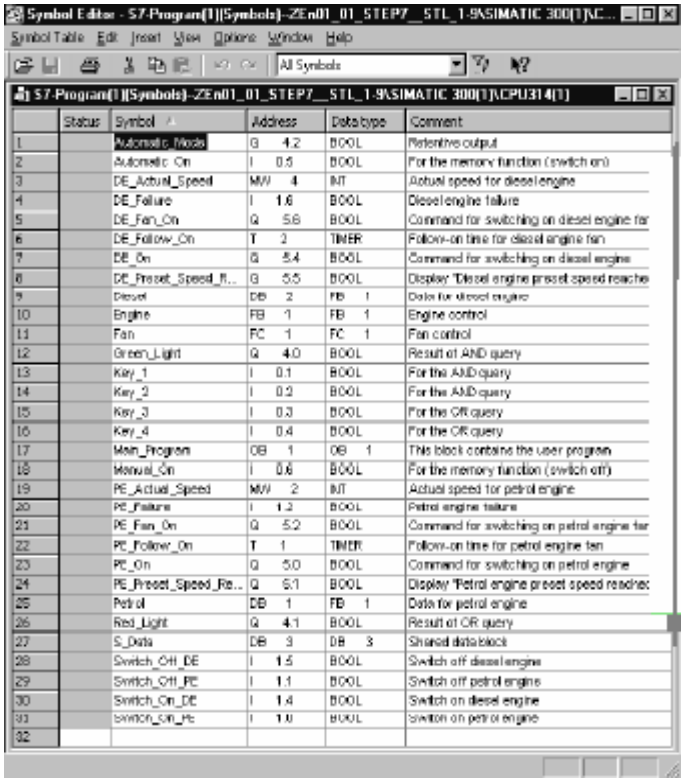
	Status	Symbol	Address	Data type
1		Main Program	OB 1	OB 1
2		Green Light	Q 4.0	BOOL
3		Red Light	Q 4.1	BOOL

Sıra 3'te "Enter "Red Light (*Kırmızı Işık*)" ve "Q 4.1" ve tamamlamak için Enter'a basın.

Bu yolla programınızın ihtiyacı olan girdilerin ve çıktıların tüm kesin adreslerine sembolik isimler verebilirsiniz.

Yapmış olduğunuz girişleri sembol tablosunda kaydedin ve pencereyi kapatın.

Tüm “Başlarken” projesi için birçok isim olduğundan sembol tablosunu Bölüm 4.1’deki “Başlarken” projenize kopya edebilirsiniz.



Burada İfade Listesi için "Başlarken" örneğindeki sembol tablosunu görebilirsiniz. Genel olarak söylemek gerekirse her S7 programı için, hangi programlama dilini seçtiğinize bakmadan, bir sembol tablosu oluşturulur. Bütün sembol tablosunda tümü yazdırılabilir karakterler, (örneğin, özel karakterler, b0şluklar v.s.) kabul edilir. "Blokların Programlanması" ve "Sembollerin Tanımlanması" başlıkları altındaki Help > Contents (Yardım > İçindekiler)de daha fazla bilgi bulabilirsiniz.

Daha önce sembol tablosuna eklenen veri türleri CPU’da işlem görecek sinyal türünü otomatik olarak belirler..STEP 7 başkalarıyla birlikte aşağıdaki veri türlerini kullanır:

BOOL BYTE WORD DWORD	Bu türden veriler bit kombinasyonlarıdır. 1 bit (BOOL türü) ila 32 bit arası (DWORD).
CHAR	Bu türden veriler ASCII karakter takımından tam olarak bir karakterlik yer işgal eder.
INT DINT REAL	Sayısal değerleri işlemek için kullanılırlar (örneğin aritmetiksel ifadeleri hesaplamak için.
S5TIME TIME DATE TIME_OF_DAY	Bu türden veriler STEP 7 içinde farklı saat ve tarih değerlerini temsil eder (örneğin, bir zamanlayıcı için tarih belirlemek veya saat girmek için).

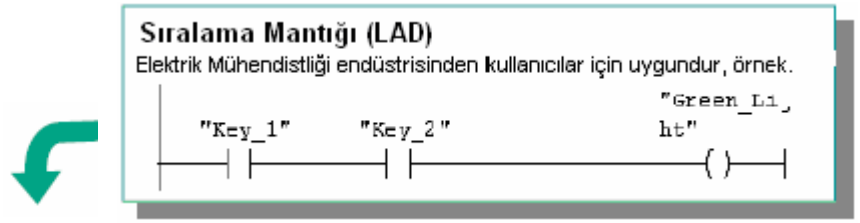
“Blokların Programlanması“ ve "Sembollerin Tanımlanması" konusunda. **Help > Contents** (*Yardım > İçindekiler*) altında daha fazla bilgi bulabilirsiniz.

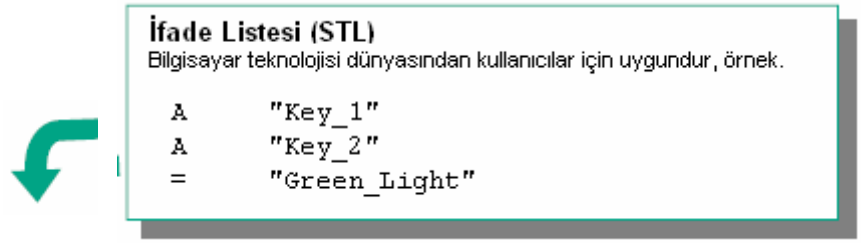
4 OB1'de Programlama

4.1 LAD/STL/FBD Program Penceresinin Açılması

Sıralama Mantığı, İfade Listesi veya Fonksiyon Blok Şeması Seçilmesi

STEP 7 ile Sıralama Mantığı (LAD), İfade Listesi (STL), veya Fonksiyon Blok Şeması (FBD) standart dillerinde S7 programları yaparsınız. Pratikte ve bu bölüm için de hangi dili kullanacağınıza karar vermelisiniz.





Şimdi OB1 bloğu Proje Sihirbazında oluşturduğunuz zaman seçtiğiniz dile göre açılacaktır. Yine de varsayılan program dilini her zaman yeniden değiştirebilirsiniz.

Sembol Tablosunun Kopyalanması ve OB1'in Açılması

Gerekli ise, "Başlarken" projenizi açın. Bunu yapmak için araç çubuğundaki **Open** (*Aç*) düğmesine tıklayın, yaptığınız "Başlarken" projesini seçin ve **OK** ile onaylayın.

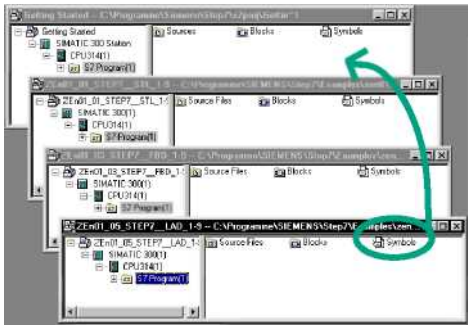
Kullanmaya karar verdiğiniz programlama diline bağlı olarak "Örnek projeler" sekmesinde aşağıdaki projelerden birini de beraber açın:

- ZEn01_05_STEP7__LAD_1-9
- ZEn01_01_STEP7__STL_1-9

veya

- ZEn01_03_STEP7__FDB_1-9

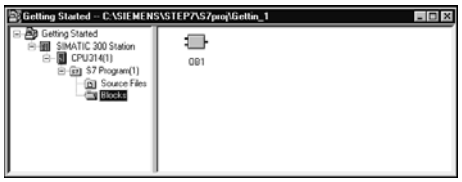
Yanda üç örnek projesinde gösterilişini görebilirsiniz.



Semboller bileşenine ulaşıncaya kadar "ZEn01_XXX"de gezinin ve onu sürükleyip bırakarak "Başlarken" pencerenizdeki **S7 Program** klasörüne kopyalayın.

Sonra "ZEn01_XXX" penceresini kapatın.

Sürükle ve bırak demek bir cismi fare ile tutarsınız ve fare tuşunu basılı tutarak hareket yerini değiştirirsiniz. Fare tuşunu bıraktığınız zaman cisim seçilen konuma yapıştırılmış olur.



"Başlarken" projesindeki **OB1**'e çift tıklayın. LAD/STL/FBD program penceresi açılır.

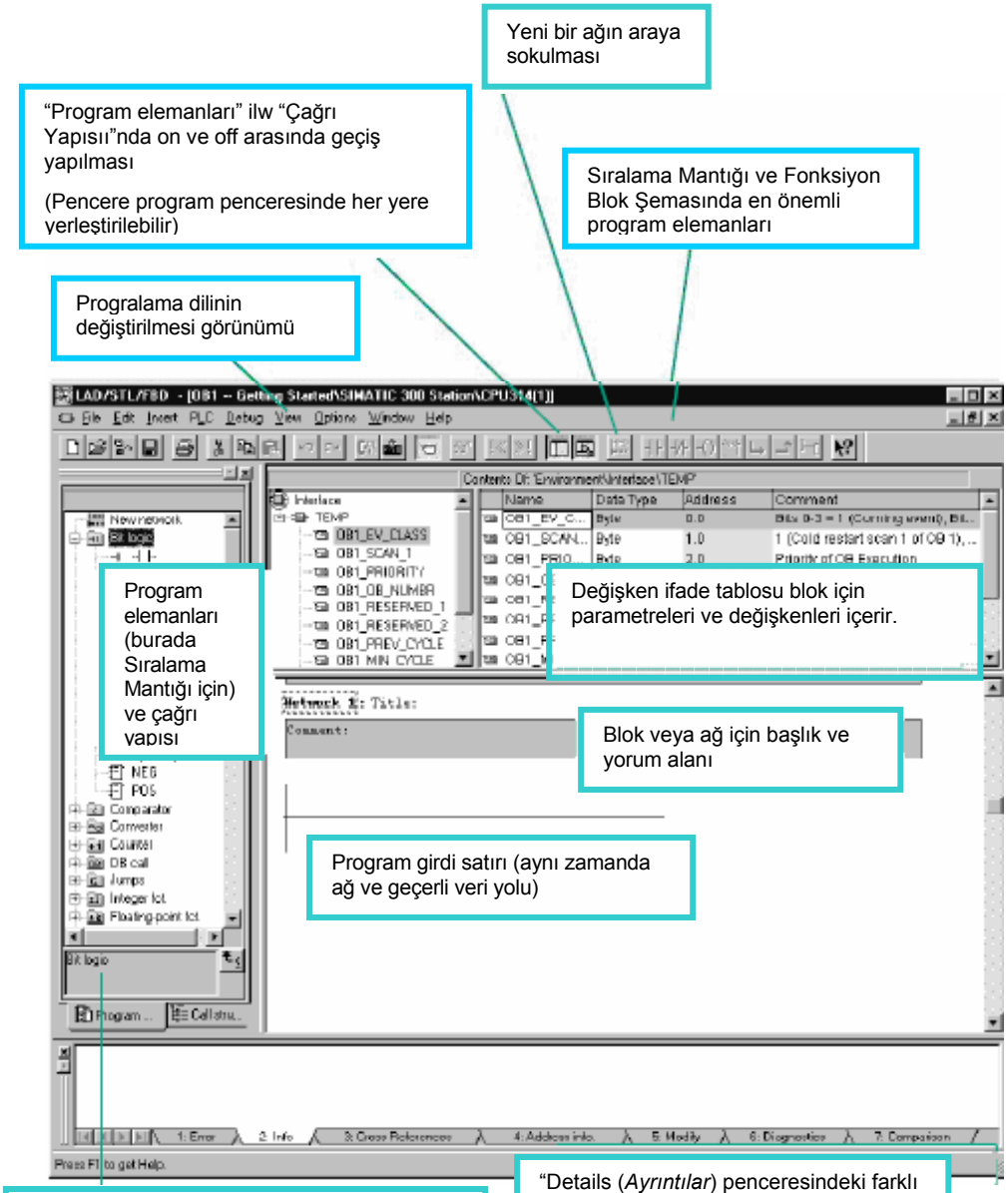
STEP 7'de, OB1 CPU tarafından çevrimsel olarak işlenir. CPU program komutlarını satır satır okur ve uygular. CPU ilk program satırına döndüğü zaman tam olarak bir çevrimi tamamlamıştır. Bunun için gerekli zaman tarama çevrim süresi olarak bilinir.

Seçtiğiniz programlama diline bağlı olarak Sıralama Mantığında Programlama için Bölüm 4,2'yi, İfade Listesi için Bölüm 4,3'ü, ya da Fonksiyon Blok Şeması için Bölüm 4,4'ü okumaya devam edin.

Help > Contents (*Yardım > İçindekiler*) altında "Blokların ve Kitaplıkların Programlanması" konularında daha fazla bigi bulabilirsiniz.

LAD/STL/FBD Program Penceresi

Tüm bloklar LAD/STL/FBD program penceresinde programlanır. Aşağıda Sıralama Mantığının görüntüsünü görebilirsiniz.

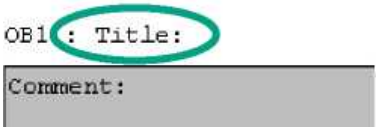


4.2 Sıralama Mantığında Seri Devre Programlaması

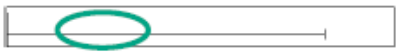
Aşağıdaki bölümde, Sıralama Mantığında (LAD)'de seri devre, paralel devre ve set /reset memory (*belleği ayarla / sıfırla*) programı yapacaksınız.

Sıralama Mantığında bir Seri Devre Programlaması

Gerekli ise LAD'i **View** (*Görünüm*) menüsünde programlama dili olarak düzenleyin.



OB1'in **Title** (*Başlık*) bölgesine tıklayın ve "Çevrimsel halinde işlenen ana programa girin.



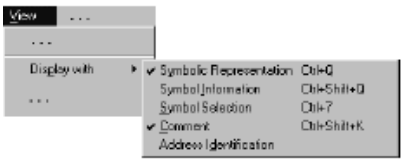
İlk öğeniz olarak mevcut veri yolunu seçin.

Araç çubuğundaki düğmeye tıklayın ve normal olarak açık olan bir kontak katın.

Aynı şekilde, ikinci normal olarak açık kontak katın.

Geçerli yolun sağ ucundaki bobini araya sokun.

Normal olarak açık kontakların adresleri ve bobin halen seri devrede yoktur.

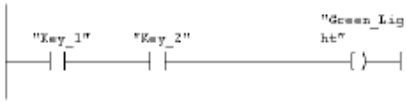


Sembolik gösterimin etkin olup olmadığına bakın.

??.? İşaretini tıklayın ve “Key_1” (tırnak içinde) sembolik adını girin. Başka bir seçenek olarak ismi aşağı açılan listeden de seçebilirsiniz.
Enter ile onaylayın.

İkinci normal olarak açık kontak için "Key_2" sembolik adını girin.

Bobin için "Green_Light" adını girin.

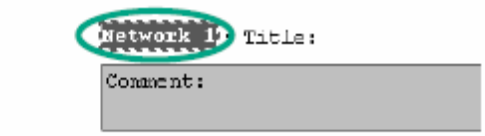


Şimdi tam bir seri devre programladınız.

Kırmızı gösterilen başka bir sembol yoksa bloğu kaydedin.

Bir sembol sembol tablosunda yoksa veya dizim hatası varsa kırmızı renkle gösterilir.

Sıralama Mantığında bir Paralel Devrenin Programlanması

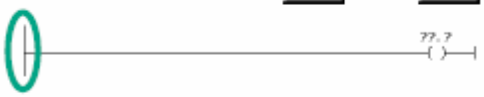


Network 1 seçin.

Yeni bir ağ katın.

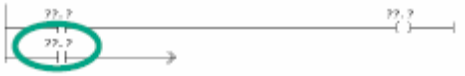
Geçerli yolu tekrar seçin.

Normal olarak açık bir kontak ve bir bobin katın.



Geçerli yolun dikey hattını seçin.

Paralel bir ayrım ekleyin.



Paralel ayrımda başka bir normal olarak açık kontak ekleyin.

Ayrımı kapatın (gerekli ise alt oku seçin).

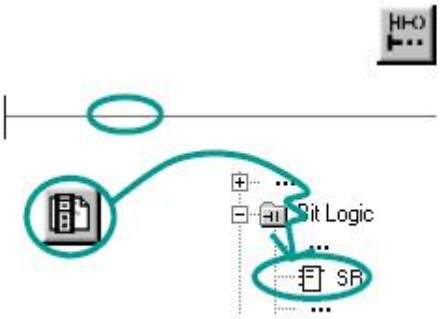
Paralel devrede halen adresler yoktur.

Sembolik adresler atamak için seri devredeki yolu izleyin.

Üst normal olarak açık kontağın üzerine "Key_3", alt kontağın üzerine "Key_4" ve bobin üzerine "Red_Light" yazın.

Bloğu kaydedin.

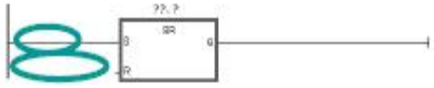
Sıralama Mantığında Belleğin Programlanması



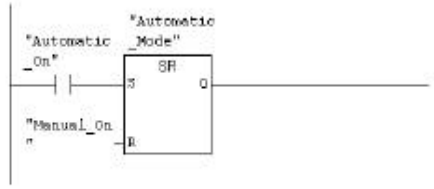
Network 2 seçin ve başka bir ağ katın.

Geçerli yolu tekrar seçin.

Bit Logic altında Program elemanları katalogunda SR elemanına ulaşıncaya kadar gezinin. Elemanı katmak için çift tıklayın.

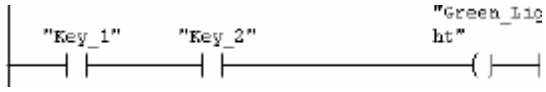


S ve R girdilerinin her birinin önüne normal olarak açılan bir kontak ekleyin.
SR elemanı için aşağıdaki sembolik isimleri girin:
Üst kontak "Automatic_On"
Alt kontak "Manual_On"
SR elemanı "Automatic Mode"

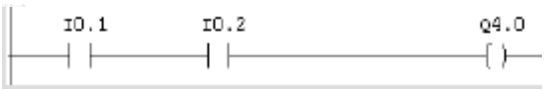


Bloğu kaydedin ve pencereyi kapatın.

Mutlak ve sembolik adreslemenin farkını görmek isterseniz **View > Display > Symbolic Representation** menü komutunun etkinliğini kaldırın.



Örnek:
LAD' de sembolik adresleme



Örnek:
LAD' de Mutlak adresleme

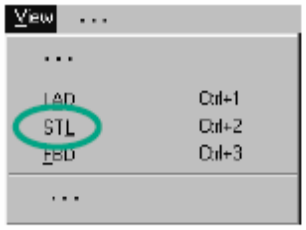
LAD/STL/FBD program penceresinde **Options > Customize** (*Seçenekler* > *Özelleştir*) menü komutunu kullanarak ve sonra "LAD/FBD" sekmesinde "Adres alanının genişliği"ni seçerek sembolik adresleme için satır sonunu değiştirebilirsiniz. Satır sonunu 10 ilâ 26 karakter arasında belirleyebilirsiniz.

"Blokların Programlanması", "Mantık Blokları Oluşturma" ve "Sıralama Talimatı Düzenleme" konularında **Help > Contents** (*Yardım* > *İçindekiler*) altında daha fazla bilgi bulabilirsiniz.

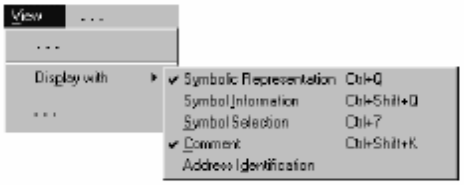
4.3 İfade Listesinde OB1 Programlaması

Aşağıdaki bölümde bir AND talimatı programlayacaksınız, bir OR talimatı ve hafıza talimatı set/reset İfade Listesinde (STL).

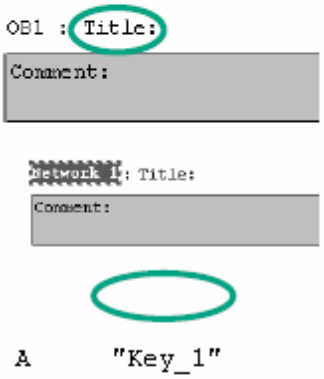
İfade Listesinde bir AND (*VE*) Talimatının Programlanması



Gerekli ise, **View** (*Görünüm*) menüsünde program dili olarak **STL**yi belirleyin.



Sembolik gösterimin etkin olup olmadığını kontrol edin.



OB1'in **title** (*başlık*) bölgesine tıklayın ve örneğin "çevrimsel olarak işlenen ana programı" girin.

İlk ifadenizin alanını seçin.

İlk program satırında bir A (AND) (*VE*) yazın, bir boşluk bırakın ve sonra sembolik isim "Key_1" (Tırnak içinde) girin.

Enter ile satırı tamamlayın. İmleç sonraki satıra atlar.

```
A"Key_l"
A"Key_2"
="GreerL_L±ght"
```

Aynı şekilde, AND (*VE*) talimatını gösterildiği gibi tamamlayın.

Artık tam bir (AND (*VE*)) talimatını programladınız. Kırmızı gösterilen başka bir sembol yoksa bloğu kaydedin.

Bir sembol sembol tablosunda yoksa veya söz dizim hatası varsa kırmızı renkle gösterilir.
Sembolik bir adı doğrudan sembol tablosundan koyabilirsiniz. ??.? işareti ve sonra Insert > Symbol (Ekle > Sembol) menü komutu üzerine tıklayın. Aşağıda görünen listeyi ilgili isme ulaşıncaya kadar kaydırın ve onu seçin. Sembolik isim otomatik olarak eklenir.

İfade Listesinde bir OR (*Veya*) Talimatının Programlanması

Network 1'l seçin..

Yeni bir ağ ekleyin ve girdi alanını tekrar seçin..

```
O     "Key_3"
```

Bir O (OR (*VEYA*)) ve sembolik isim "Key_3" (AND talimatı için uyguladığınız şekilde) girin.

```
O     "Key_3"
O     "Key_4"
=     "Red_Light"
```

OR (*VEYA*) talimatını girin ve kaydedin.

İfade Listesinde bir Bellek Talimatının Programlanması

Network 2 seçin ve başka bir ağ katın.

```
A    "Automatic_On"
```

Birinci satırda A talimatını "Automatic On" sembolik adı ile girin.

Bellek talimatını tamamlayın ve kaydedin. Bloğu kapatın.

```
A    "Automatic_On"
5    "Automatic_Mode"
A    "Manual_On"
R    "Automatic Mode"
```

Mutlak ve sembolik adresleme arasındaki farkı görmek isterseniz, **View** > **Display** > **Symbolic Representation** menü komutunun etkinliğini kaldırın.

```
A    "Key  1"
A    "Key_2"
=    "Green Light"
```

Örnek:
STL'de sembolik adresleme.

```
A    I    0.1
A    I    0.2
=    Q    4.0
```

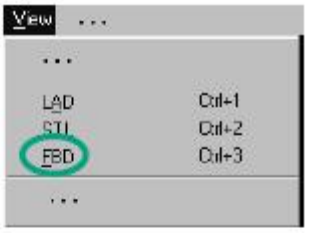
Örnek:
STL'de mutlak adresleme

"Blokların Programlanması", "Mantık Blokları Düzenlemesi" ve "STL ifadelerini Düzenleme" konularında **Help** > **Contents** (*Yardım* > *İçindekiler*) altında daha fazla bilgi bulabilirsiniz.

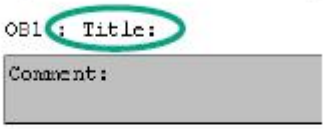
4.4 Fonksiyon Blok Şemasında OB1 Programlaması

Aşağıdaki bölümde, Fonksiyon Blok Şemasında (FBD) bir AND (*VE*) fonksiyonu, bir OR (*VEYA*) fonksiyonu ve bir bellek fonksiyonu programlayacaksınız.

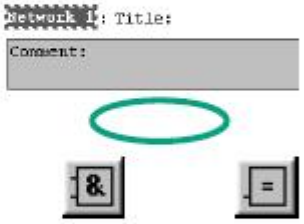
Fonksiyon Blok Şemasında bir AND (*VE*) Fonksiyonunun Programlanması



Gerekli ise, **View** (*Görünüm*) menüsünde programlama dili olarak **FBD** belirleyin.

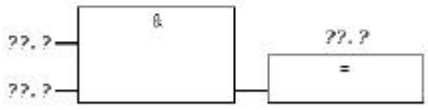


OB1 alanında **title** (*başlık*) üzerine tıklayın ve örneğin "Çevrimsel olarak işlenen ana programı" girin.

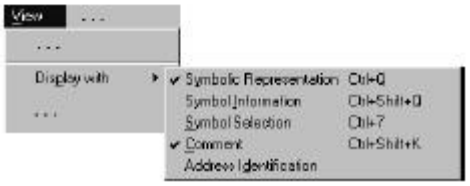


AND (*VE*) fonksiyonu için girdi bölgesini seçin (yorum alanının altında).

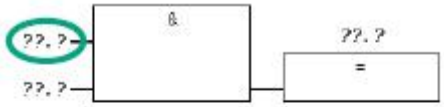
Bir AND (*VE*) kutusu (&) ve bir atama (=) ekleyin..



AND (*VE*) fonksiyonunda elemanların adresleri halen yoktur.

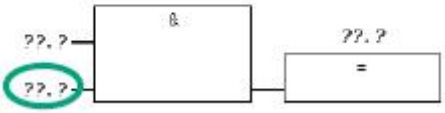


Sembolik gösterimin etkin olup olmadığını kontrol edin.

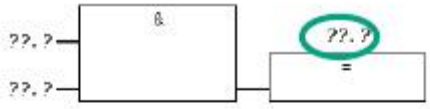


??.? İşareti üzerine tıklayın ve "Key_1" (tırnak içinde) sembolik adını girin. Başka bir seçenek olarak aşağı açılan listeden de seçebilirsiniz.

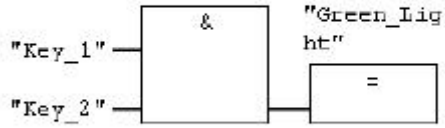
Enter ile onaylayın.



İkinci girdi için "Key_2" sembolik adını girin.



Atama için "Green_Light" adını girin.



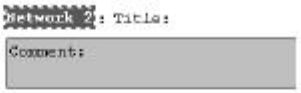
Şimdi siz tam bir AND (*VE*) fonksiyonu programladınız.

Kırmızı gösterilen başka sembol yoksa bloğu kaydedebilirsiniz.

Bir sembol sembol tablosunda yoksa veya dizim hatası varsa kırmızı renkle gösterilir.

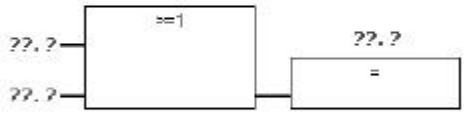
Fonksiyon Blok Şemasında bir OR (*VEYA*) Fonksiyonunun Programlanması

Yeni bir ağ girin.

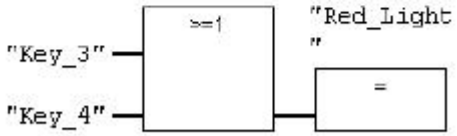


OR (*VEYA*) fonksiyonu için tekrar girdi bölgesi belirleyin.

Bir OR (*VEYA*) kutusu ve bir atama (=) ekleyin.



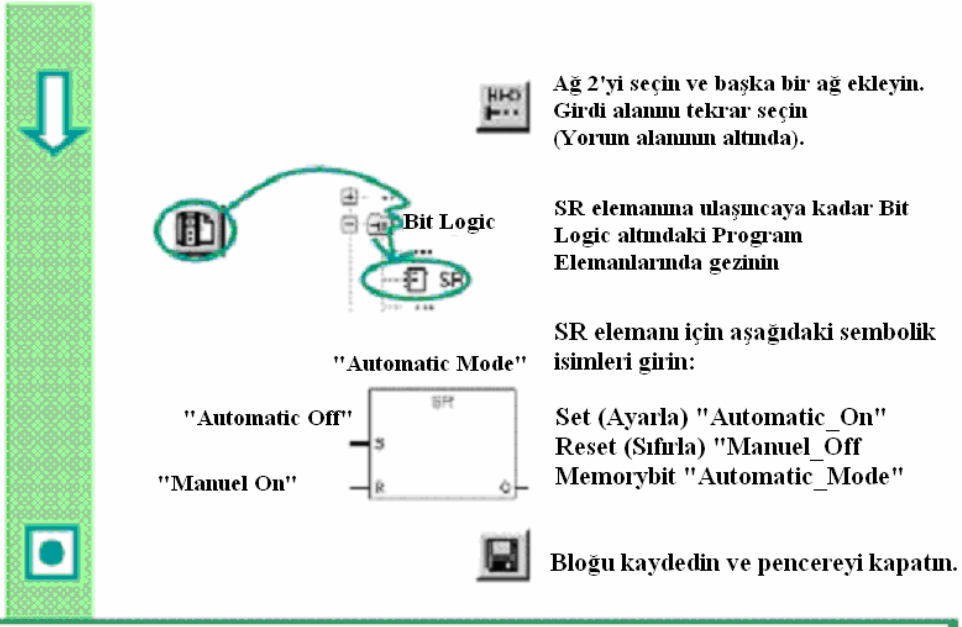
OR (*VEYA*) fonksiyonunda elemanların adresleri halen yoktur. AND (*VE*) fonksiyonundaki şekilde devam edin.



Üst girdi için "Key_3", alt girdi için "Key_4" ve atama için "Red_Light" girin.

Bloğu kaydedin.

Blok Şemasında bir Bellek Fonksiyonunun Programlaması



Ağ 2'yi seçin ve başka bir ağ ekleyin.
Girdi alanını tekrar seçin
(Yorum alanının altında).

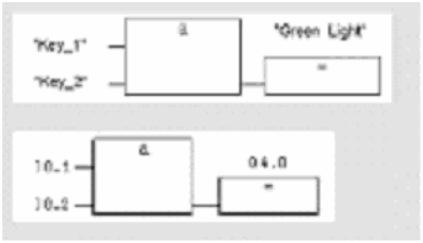
SR elemanına ulaşıncaya kadar Bit Logic altındaki Program Elemanlarında gezinin

SR elemanı için aşağıdaki sembolik isimleri girin:

Set (Ayarla) "Automatic_On"
Reset (Sıfırla) "Manuel_Off
Memorybit "Automatic_Mode"

Bloğu kaydedin ve pencereyi kapatın.

Mutlak ve sembolik adresleme arasındaki farkı görmek isterseniz, View>Display>Symbolic Representation menü komutunun etkinliğini kaldırın.



Örnek:
STL'de sembolik adresleme

Örnek:
STL'de mutlak adresleme

LAD/STL/FBD program penceresinde Options > Customize (Seçenekler>Özelleştir) menü komutunu kullanarak ve sonra "LAD/FBD" sekmesinde "Adres alanının genişliği"ni seçerek sembolik adresleme için satır sonunu değiştirebilirsiniz. Satır sonunu 10 ila 26 karakter sırasında belirleye bilirsiniz.

"Blokların Programlanması, "Mantık Blokları Düzenlenmesi" ve "STL ifadelerini Düzenleme" konularında Help>Contents (Yardım>İçindekiler) altında daha fazla bilgi bulabilirsiniz.

5 Fonksiyon Blokları ve Veri Blokları ile Programlama

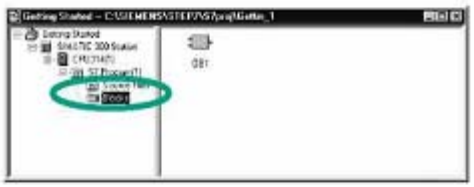
5.1 Fonksiyon Bloklarının (FB) Oluşturulması ve Açılması

Fonksiyon bloğu (FB) program hiyerarşisinde organizasyon bloğunun altında yer alır. Programın OB1 içinde birçok kere çağrılabilen bir bölümünü içerir. Fonksiyon bloğunun tüm biçimsel parametreleri ve statik verileri fonksiyon bloğuna atanan ayrı bir Veri Bloğunda (DB) saklanır.

Artık iyi bildiğiniz LAD/STL/FBD program penceresinde fonksiyon bloğunu (FB1, “Engine” (*Motor*) sembolik adını, sayfa 3,3’teki sembol tablosuna bakın) programlayacaksınız. Bunu yapmak için Bölüm 4’teki ile aynı programlama dilini (OB1 programlaması) kullanacaksınız.

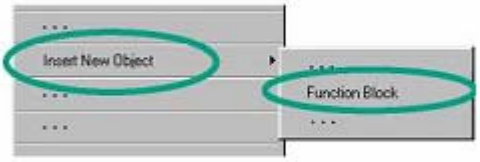
Sembol tablosunu “Başlarken” projenize önceden kopyalamış olmalısınız. Kopyalamadıysanız, bunu yapmak için, sayfa 4-2’den sembol tablolarının nasıl kopyalanacağını okuyun ve sonra bu bölüme dönün.

Gerekli ise “Başlarken” projesini açın.

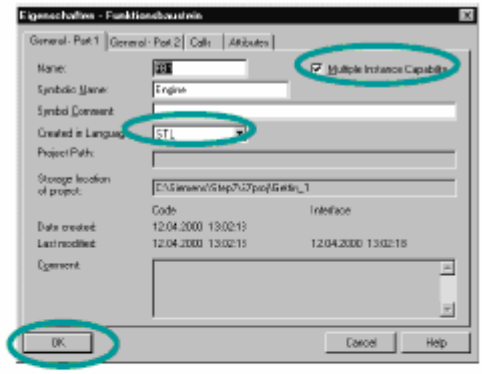


Blocks (*Bloklar*) klasörüne gezinin ve onu açın.

Pencerenin sağ yarısını farenin sağ tuşu ile açın.



Farenin sağ tuşunun açılır menüsünde menü çubuğunun en önemli komutları bulunur. Yeni bir konu olarak **function block** (*fonksiyon bloğu*)’nu ekleyin.



"Properties - Function Block" (*Özellikler – Fonksiyon Bloğu*) diyalog kutusunda blok oluşturmak istediğiniz dili seçin, **"Multiple instance FB,"** (*Çoklu Olay*) kontrol kutusunu etkinleştirin ve OK ile ayarlarınızı doğrulayın.

FB1 fonksiyon bloğu Blocks (*Bloklar*) klasörüne eklemiştir.

LAD/STL/FBD program penceresini açmak için **FB1** üzerine çift tıklayın..

Seçtiğiniz programlama diline bağlı olarak, ya Sıralama Mantığı için Bölüm **5.2**'yi, İfade Listesi için **5.3**'ü ya da Fonksiyon Blok Şeması için **5.4**'ü okumaya devam edin.

Help > Contents (*Yardım > İçindekiler*) altında "Blokların Programlanması" ve "Blokların ve Kitaplıkların Oluşturulması" başlıkları altında daha fazla bilgi bulabilirsiniz.

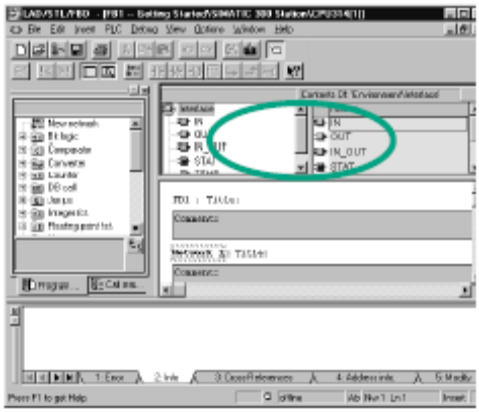
5.2 Sıralama Mantığında FB1 Programlaması

İki farklı Veri Bloğunu kullanarak örneğin bir benzinli veya dizel motoru kontrol edip izleyen bir fonksiyon bloğunu nasıl programlayacağınızı göstereceğiz.

Tüm "motora özel" sinyaller organizasyon bloğundan fonksiyon bloğuna blok parametreleri olarak aktarılır ve bu nedenle değişken açıklama tablosunda girdi ve çıktı parametreleri olarak ("giriş" ve "çıkış" açıklaması) listelenmelidir.

Bir seri devrenin, bir paralel devrenin ve bir bellek fonksiyonunun STEP 7 ile nasıl girileceğini önceden bilmeniz gerekir.

Önce Değişkenleri Açıkla / Tanımla



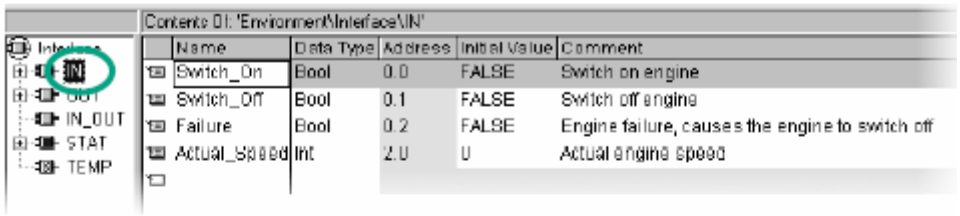
LAD/STL/FBD program pencereniz açılır ve **View** > **LAD** (*Görüntüle* > *LAD*) (programlama dili) etkindir.

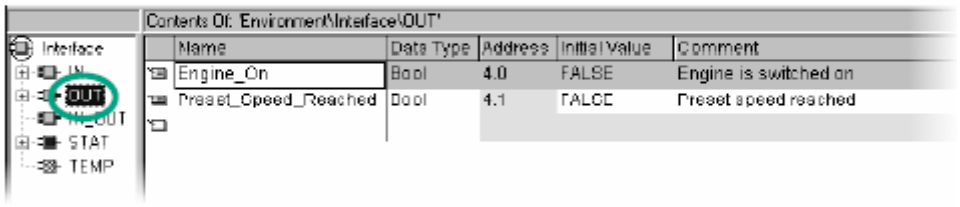
FB1'in artık başlıkta olduğunu not edin, çünkü program penceresini açmak için FB1'e çift tıkladınız.

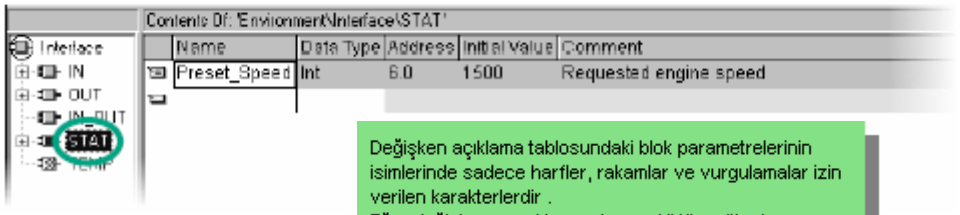
Değişken açıklama bölgesi bir değişken gözden geçirmesi (sol panel) ve değişken ayrıntı görünümü (sağ panel)'den oluşur..

Değişken gözden geçirmesinde "IN", "OUT" ve "STAT" açıklama tiplerini birbiri arkasına seçin ve sonraki açıklamaları ilgili değişken detaylarına girin.

Değişken gözden geçirmesinde ilgili hücrelere tıklayın ve sonraki rakamların girdilerini uygulayın. Verileri aşağıya açılan listeden seçebilirsiniz.

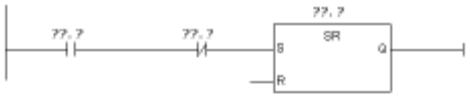




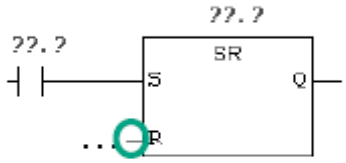


Değişken açıklama tablosundaki blok parametrelerinin isimlerinde sadece harfler, rakamlar ve vurgulamalar izin verilen karakterlerdir .
Eğer değişken ayrıntılarınızda gerekli tüm sütunlar gösterilmiyorsa , kısayol menü komutu (fare sağ tıklaması)yardımıyla bunları gösterebilirsiniz.

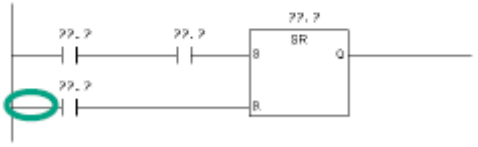
Bir Motorun Anahtarının On ve Off (*Açık ve Kapalı*) Programlanması



Ara çubuğundaki ilgili tuşları veya Program Elemanları katalogunu kullanarak, normal olarak açık bir kontağı, normal olarak kapalı bir kontağı ve Network 1 serisinde bir SR elemanını ekleyin.

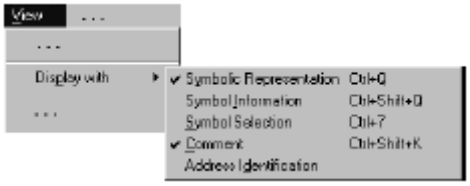


Sonra R girdisinden önce hemen geçerli yolu girin.



Başka bir normal olarak açık kontak ekleyin. Bu görüşmeden önce hemen geçerli yolu girin.

Normal olarak açık kontağa paralel olarak normal olarak kapalı kontak ekleyin.

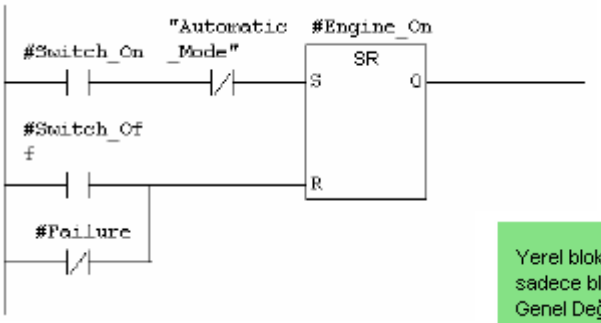


Sembolik gösterimin etkin olup olmadığını kontrol edin.

Soru işaretlerini seçin ve değişken açıklama tablosundan karşılık isimleri girin (# işareti otomatik olarak atanır).

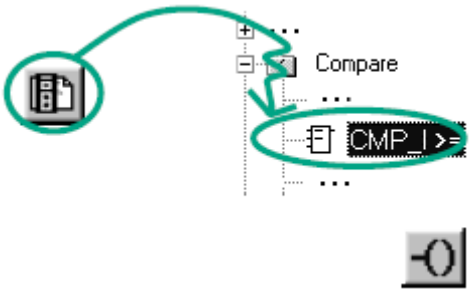
Seri devrede normal olarak kapalı kontak için “Automatic_mode” sembolik adını girin.

Sonra programınız kaydedin.



Yerel blok değerleri bir# işareti ile gösterilir ve sadece blok içinde geçerlidir.
Genel Değişkenler tırnak içinde gözükürler. Bunlar sembol tablosunda tanımlanır ve programın tamamı için geçerlidirler.
"Otomatik Mod" sinyal durumu OB1'de (Network 3, bak sayfa 4.7) başka bir SR elemanı tarafından tanımlanır ve FB1'de sorgulanırlar.

Hız İzleme Programlanması



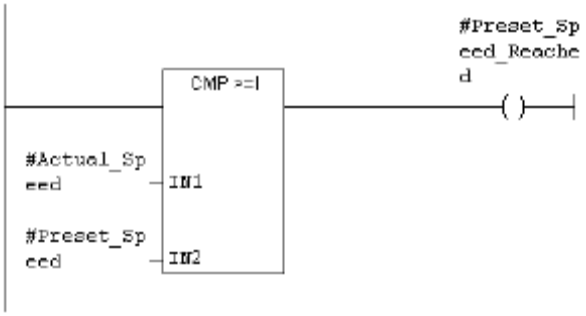
Yeni bir ağ ekleyin ve geçerli yolu seçin.

Compare (*Karşılaştır*) fonksiyonuna ulaşıncaya kadar Program elemanları katalogunda gezinin ve **CMP>=I** ekleyin.

Geçerli yola bir de bobin ekleyin.

Tekrar soru işaretlerini seçin ve bobini ve karşılaştırıcıyı değişken açıklama tablosundan isimlerle adlandırın.

Sonra programınızı kaydedin.



Motor anahtarı ne zaman on ve off (*Açık ve kapalı*) **olur?**

#Switch_On değişkeni “1” sinyal durumunda ve “automatic_Mode değişkeni “0” sinyal durumunda olduğu zaman, motor anahtarı on durumuna gelir (*motor çalışır*). Bu fonksiyon, “Automatic_Mode” negatif hale getirilinceye kadar, etkinleşmez (normal olarak kapalı kontak)..

#Switch_Off değişkeni “1” sinyal durumunda ve #Fault (*Hata*) değişkeni “0” sinyal durumunda olduğu zaman, motor anahtarı off durumuna gelir (*motor durur*). Bu fonksiyon #Fault sinyalinin negatif hale gelmesiyle tekrar elde edilir (#Fault bir “sıfır-aktif” sinyaldir ve normal durumda “1” sinyali verir ve eğer bir hata oluşursa “0”sinyali olur).

Karşılaştırıcı motor hızını nasıl izler?
Karşılaştırıcı #Actual_Speed ve #Setpoint_Speed hızlarını karşılaştırır ve sonucu #Setpoint_Speed_Reached sonuçlarına atar (sinyal durumu "1").

Help > Contents (*Yardım > İçindekiler*) altında “Blokların Programlanması”, “Mantık Blokları Oluşturulması” ve “Değişken Açıklaması Düzenlenmesi" veya “LAD Talimatı Düzenlenmesi” başlıkları altında daha fazla bilgi bulabilirsiniz.

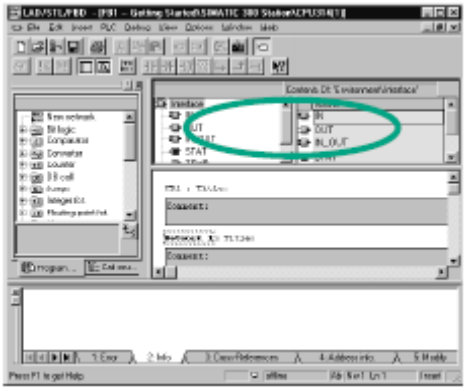
5.3 İfade Listesinde FB1 Programlaması

İki farklı Veri Bloğunu kullanarak örneğin bir benzinli veya dizel motoru kontrol edip izleyen bir fonksiyon bloğunu nasıl programlayacağınızı göstereceğiz.

Tüm "motora özel" sinyaller organizasyon bloğundan fonksiyon bloğuna blok parametreleri olarak aktarılır ve bu nedenle değişken açıklama tablosunda girdi ve çıktı parametreleri olarak ("giriş" ve "çıkış" açıklaması) listelenmelidir.

Bir seri devrenin, bir paralel devrenin ve bir bellek fonksiyonunun STEP 7 ile nasıl girileceğini önceden bilmeniz gerekir.

Önce Değişkenleri Açıkla / Tanımla



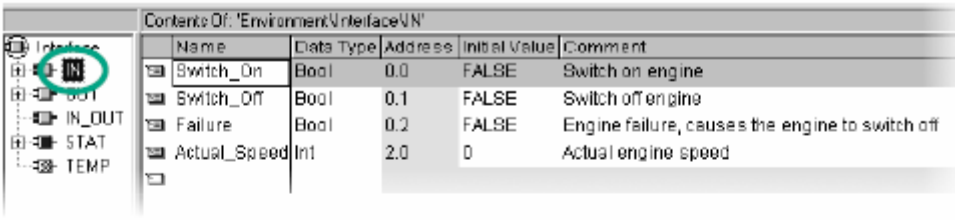
LAD/STL/FBD program pencereniz açılır ve **View** > **LAD** (*Görüntüle* > *LAD*) (programlama dili) etkindir.

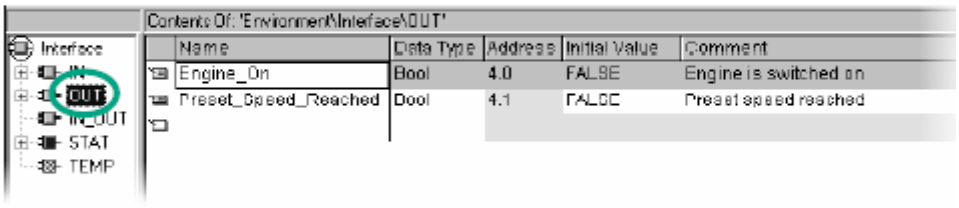
FB1'in artık başlıkta olduğunu not edin, çünkü program penceresini açmak için FB1'e çift tıkladınız.

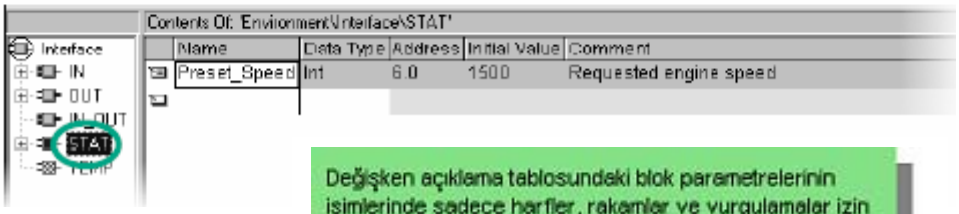
Değişken açıklama bölgesi bir değişken gözden geçirmesi (sol panel) ve değişken ayrıntı görünümü (sağ panel)'den oluşur..

Değişken gözden geçirmesinde "IN", "OUT" ve "STAT" açıklama tiplerini birbiri arkasına seçin ve sonraki açıklamaları ilgili değişken detaylarına girin.

Değişken gözden geçirmesinde ilgili hücrelere tıklayın ve sonraki rakamların girdilerini uygulayın. Verileri aşağıya açılan listeden seçebilirsiniz.

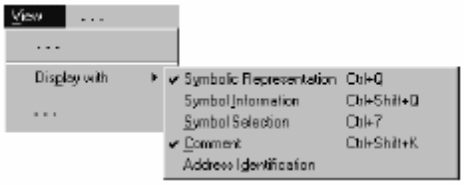






Değişken açıklama tablosundaki blok parametrelerinin isimlerinde sadece harfler, rakamlar ve vurgulamalar izin verilen karakterlerdir .

Bir Motorun Anahtarının On ve Off (*Açık ve Kapalı*) **Programlanması**



Sembolik gösterimin etkin olup olmadığını kontrol edin.

```
A     #Switch_On
AN    "Automatic_Mode"
S     #Engine_On
O     #Switch_Off
ON    #Failure
R     #Engine_On
```

Network 1'de karşılık talimatı girin.

Yerel blok değerleri bir# işareti ile gösterilir ve sadece blok içinde geçerlidir.
Genel Değişkenler tırnak içinde gözükürler. Bunlar sembol tablosunda tanımlanır ve programın tamamı için geçerlidirler.
"Otomatik Mod" sinyal durumu OB1'de (Network 3, bak sayfa 4.10) başka bir SR elemanı tarafından tanımlanır ve FB1'de sorgulanırlar.

Hız İzlemenin Programlanması

```
L     #Actual_Speed
L     #Preset_Speed
>=I
=     #Preset_Speed_Reached
```

Yeni bir ağ ekleyin ve ilgili talimatları girin. Sonra programınızı kaydedin.

Motor anahtarı ne zaman on ve off (*Açık ve kapalı*) **olur?**

#Switch_On değişkeni “1” sinyal durumunda ve “automatic_Mode değişkeni “0” sinyal durumunda olduğu zaman, motor anahtarı on durumuna gelir (*motor çalışır*). Bu fonksiyon, “Auromatic_Mode” negatif hale getirilinceye kadar, etkinleşmez (normal olarak kapalı kontak)..

#Switch_Off değişkeni “1” sinyal durumunda ve #Fault (*Hata*) değişkeni “0” sinyal durumunda olduğu zaman, motor anahtarı off durumuna gelir (*motor durur*). Bu fonksiyon #Fault sinyalinin negatif hale gelmesiyle tekrar elde edilir (#Fault bir “sıfır-aktif” sinyaldir ve normal durumda “1” sinyali verir ve eğer bir hata oluşursa “0”sinyali olur).

Karşılaştırıcı motor hızını nasıl izler?
Karşılaştırıcı #Actual_Speed ve #Setpoint_Speed hızlarını karşılaştırır ve sonucu #Setpoint_Speed_Reached sonuçlarına atar (sinyal durumu "1").

Help > Contents (*Yardım > İçindekiler*) altında “Blokların Programlanması”, “Mantık Blokları Oluşturulması” ve “Değişken Açıklaması Düzenlemesi” veya “LAD Talimatı Düzenlenmesi” başlıkları altında daha fazla bilgi bulabilirsiniz.

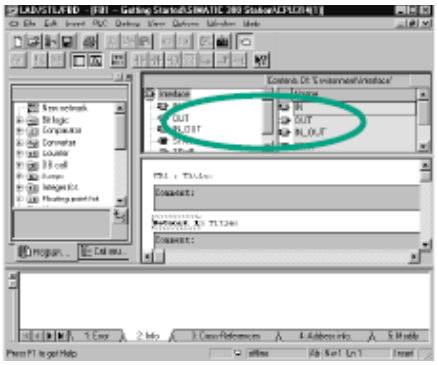
5.4 Fonksiyon Blok Şemasında FB1 Programlaması

İki farklı Veri Bloğunu kullanarak örneğin bir benzinli veya dizel motoru kontrol edip izleyen bir fonksiyon bloğunu nasıl programlayacağınızı göstereceğiz.

Tüm "motora özel" sinyaller organizasyon bloğundan fonksiyon bloğuna blok parametreleri olarak aktarılır ve bu nedenle değişken açıklama tablosunda girdi ve çıktı parametreleri olarak ("giriş" ve "çıkış" açıklaması) listelenmelidir.

Bir seri devrenin, bir paralel devrenin ve bir bellek fonksiyonunun STEP 7 ile nasıl girileceğini önceden bilmeniz gerekir.

Önce Değişkenleri Açıkla / Tanımla



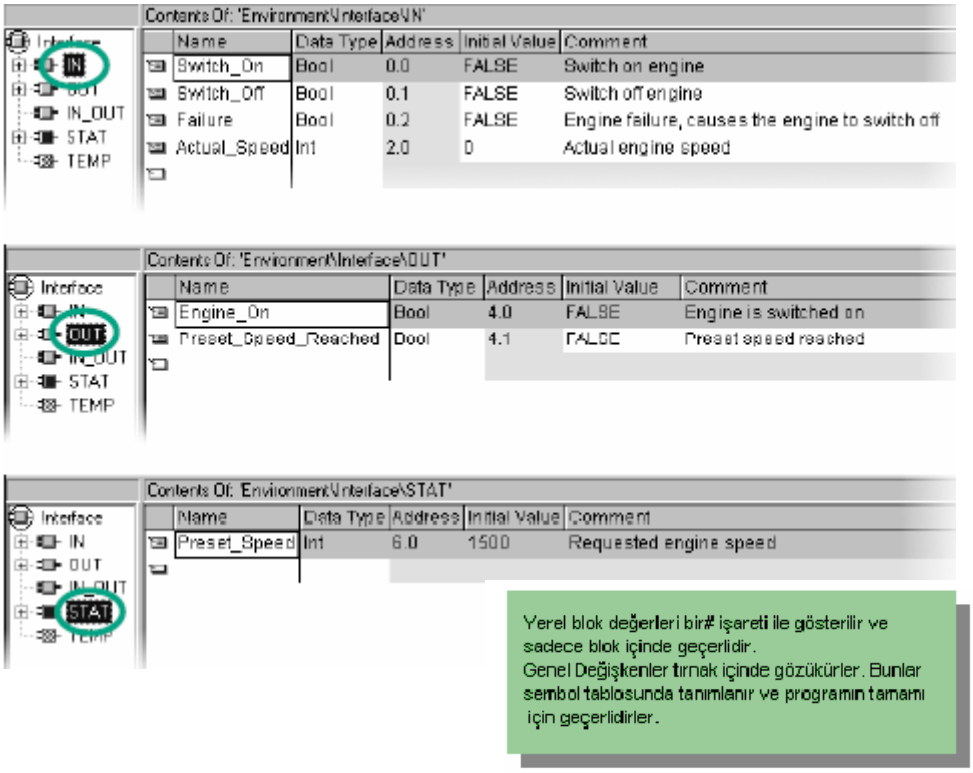
LAD/STL/FBD program pencereniz açılır ve **View** > **FBD** (*Görüntüle* > *FBD*) (programlama dili) etkindir.

FB1'in artık başlıkta olduğunu not edin, çünkü program penceresini açmak için FB1'e çift tıkladınız.

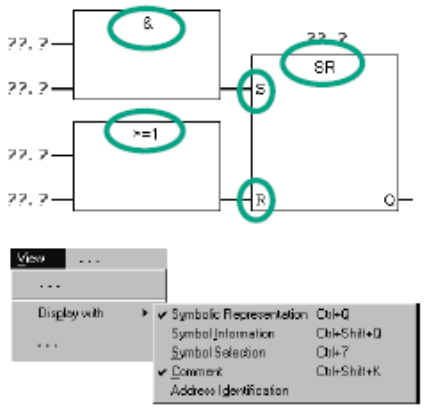
Değişken açıklama bölgesi bir değişken gözden geçirmesi (sol panel) ve değişken ayrıntı görünümü (sağ panel)'den oluşur..

Değişken gözden geçirmesinde "IN", "OUT" ve "STAT" açıklama tiplerini birbiri arkasına seçin ve sonraki açıklamaları ilgili değişken detaylarına girin.

Değişken gözden geçirmesinde ilgili hücrelere tıklayın ve sonraki rakamların girdilerini uygulayın. Verileri aşağıya açılan listeden seçebilirsiniz.



Bir Motoru Açma ve Kapama Programlaması



Program elemanları katalogunu (Bit Mantık klasörü) kullanarak Network 1'e bir SR fonksiyonu ekleyin.

S (Set (*Ayar*)) girdisine bir AND (*VE*), ve R (Reset (*Sıfırla*) girdisine bir OR (*VEYA*) ekleyin.

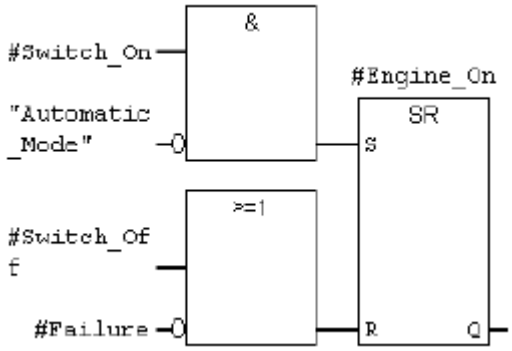
Sembolik gösterimin etkin olup olmadığını kontrol edin.

??.? İşaretini tıklayın ve açıklama tablosundan uygun isimleri girin. (# işareti otomatik olarak atanır).

Bir AND (VE) fonksiyonu girdisinin "Automatic_Mode" sembolik adıyla yönlendirildiğinden emin olun.

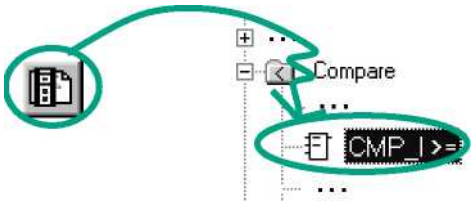
"Automatic_Mode" ve #Fault (*Hata*) girdilerini araç çubuğundan uygun tuşla olumsuz hale getirin.

Sonra programınızı kaydedin.



Yerel blok değerleri bir# işareti ile gösterilir ve sadece blok içinde geçerlidir.
Genel Değişkenler tırnak içinde gözükürler. Bunlar sembol tablosunda tanımlanır ve programın tamamı için geçerlidirler.
"Otomatik Mod" sinyal durumu OB1'de (Network 3, bak sayfa 4.17) başka bir SR elemanı tarafından tanımlanır ve FB1'de sorgulanırlar.

Hız İzlemenin Programlanması

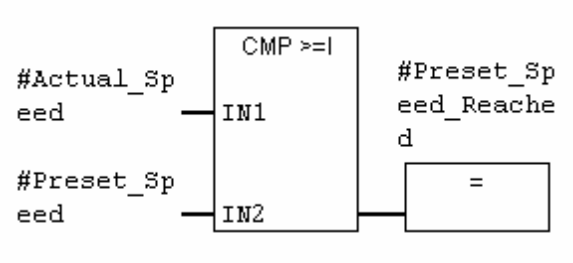


Yeni bir ağ ekleyin ve girdi alanını secin.

Compare (*Karşılaştır*) fonksiyonuna ulaşıncaya kadar Program elemanları katalogunda gezinin ve **CMP>=I** ekleyin.

Karşılaştırıcıya bir çıktı ataması iliştirin ve değişken ve yan tablosundan adları olan girdiler yönlendirin.

Sonra programınızı kaydedin.



Motor anahtarı ne zaman on ve off (*Açık ve kapalı*) **olur?**

#Switch_On değişkeni “1” sinyal durumunda ve “automatic_Mode değişkeni “0” sinyal durumunda olduğu zaman, motor anahtarı on durumuna gelir (*motor çalışır*). Bu fonksiyon, “Auromatic_Mode” negatif hale getirilinceye kadar, etkinleşmez (normal olarak kapalı kontak).

#Switch_Off değişkeni “1” sinyal durumunda ve #Fault (*Hata*) değişkeni “0” sinyal durumunda olduğu zaman, motor anahtarı off durumuna gelir (*motor durur*). Bu fonksiyon #Fault sinyalinin negatif hale gelmesiyle tekrar elde edilir (#Fault bir “sıfır-aktif” sinyaldir ve normal durumda “1” sinyali verir ve eğer bir hata oluşursa “0”sinyali olur).

Karşılaştırıcı motor hızını nasıl izler?
Karşılaştırıcı #Actual_Speed ve #Setpoint_Speed hızlarını karşılaştırır ve sonucu #Setpoint_Speed_Reached sonuçlarına atar (sinyal durumu "1").

Help > Contents (*Yardım* > *İçindekiler*) altında “Blokların Programlanması”, “Mantık Blokları Oluşturulması” ve “Değişken Açıklaması Düzenlemesi” veya “LAD Talimatı Düzenlenmesi” başlıkları altında daha fazla bilgi bulabilirsiniz.

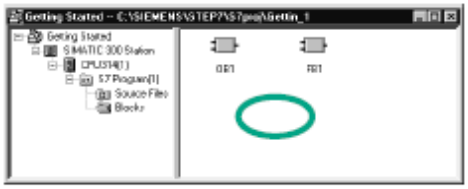
5.5 Kademeli Veri Blokları Üretilmesi ve Gerçek Değerlerin Değiştirilmesi

FB1 (“Motor”) fonksiyon bloğunu az önce programladınız ve başka şeylerle birlikte, değişkenler açıklama tablosunu motora özgü parametreleri tanımladınız.

OB1’de daha sonra fonksiyon bloğunu çağırmayı programlayabilmeniz için mukabil Veri Bloğunu üretmelisiniz. Bir fonksiyon bloğuna her zaman örnek bir veri bloğu atanır.

Fonksiyon bloğu benzinli veya dizel bir motoru kontrol etmek ve izlemek içindir. Farklı motor hızları ayar noktaları, gerçek değerin (#Setpoint_Speed) değiştirildiği iki farklı veri bloğunda kaydedilir.

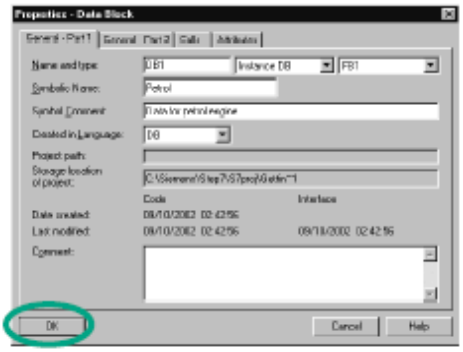
Fonksiyon bloğunu sadece bir kere merkezden programlayarak ilgili programlamanın miktarını azaltabilirsiniz.



SIMATIC Yöneticisinde “Başlarken” projesi açılır.

Blocks (*Bloklar*)’de gezinin ve pencerenin sağ yarısına fare ile tıklayın.

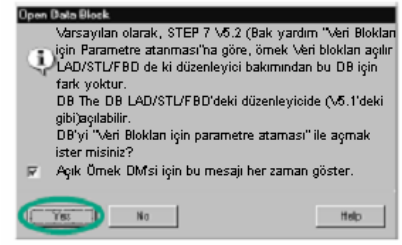
Açılır menüyü kullanarak sağ fare tuşu ile bir **Veri bloğu** ekleyin.

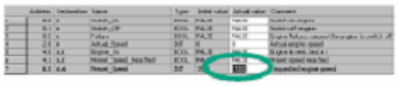


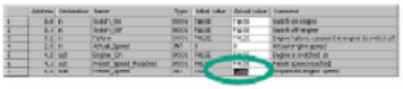
"Özellikler Veri Bloğu" diyalog kutusunda DB1 adını uygulayın, sonra bitişik aşağı açılan listede “Örnek DB” uygulamasını seçin ve “FB1” atanan fonksiyon bloğunun adını uygulayın. “Özellikler” diyalog kutusunda gösterilen tüm ayarları **OK** ile uygulayın..

DB1 Veri bloğu “Başlarken” projesine eklenir.

DB1’i açmak için çift tıklayın.







Örnek Veri Bloklarına parametreler atamak için sonraki diyalogu **Yes** ile onaylayın.

Sonra Benzinli motor için Gerçek Değer sütununa (“Setpoint_Speed) sırasına “1500” girin. Şimdi bu motor için maksimum hızı tanımlamış oldunuz.

DB1’i kaydedin ve program penceresini kapatın.

DB1 ile aynı şekilde FB1 için ayrı bir Veri Bloğu, DB2’yi üretin.

Şimdi dizel motor için gerçek değer olarak “1200” girin.

DB2’yi kaydedin ve program penceresini kapatın.

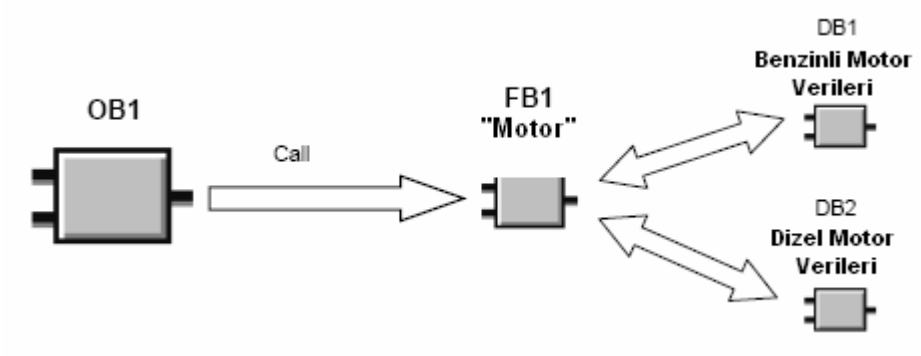
Gerçek değerleri değiştirerek, iki motoru sadece bir fonksiyon bloğundan kontrol etmek için hazırlıkları bitirdiniz. Daha çok sayıda motoru kontrol etmek için tüm yapacağınız ek Veri Blokları üretmektir..

Yapmanız gereken sonraki şey, OB1’deki fonksiyon bloğu için çağrının programlanmasıdır. Bunu yapmak için, kullanmakta olduğunuz programlama diline bağlı olarak Sıralama Mantığı için Bölüm 5.6’yı, İfade Listesi için Bölüm 5.7’yi, Fonksiyon Blok Şeması niçin Bölüm 5.8’i okumaya devam edin.

Help > Contents (*Yardım* > *İçindekiler*) altında “Blokların Programlanması”, “Veri Blokları Oluşturulması” ve “Değişken Açıklaması Düzenlemesi” başlıkları altında daha fazla bilgi

5.6 Sıralama Mantığında Blok Çağrı Programlaması

Bir fonksiyon bloğunu programlamakla yaptığınız tüm çalışma, siz bu bloğu bir OB1 bloğuna çağırmadıkça yararsızdır. Her fonksiyon bloğu çağrısı için bir Veri bloğu kullanılır ve bu yolla her iki motoru kontrol edebilirsiniz.

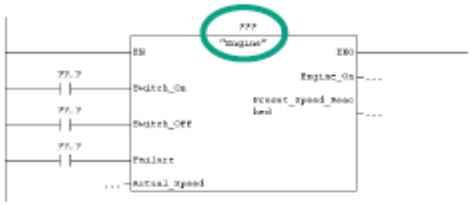


SIMATIC Yöneticisi “Başlarken” projenizle birlikte açılır.

Blocks (*Bloklar*) klasöründe gezinin ve **OB1**’i açın.

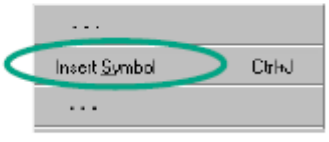
Network 3’ü seçin ve sonra LAD/STL/FBD program penceresinde network 4’ü ekleyin.

Program elemanları katalogunda **FB1**’e gezinin ve çift tıklama yardımıyla onu ekleyin.

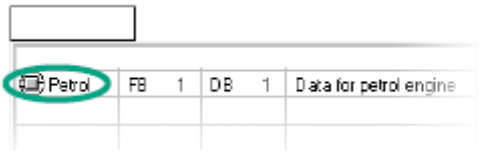


Aşağıdakilerden her birinin önüne normal olarak açık bir kontak koyun: Switch_On, Switch_Off ve Fault.

“Motor”un üzerindeki ??? İşaretine tıklayın ve sonra imleci aynı durumda tutarak sağ fare tuşu ile girdi çerçevesine tıklayın.

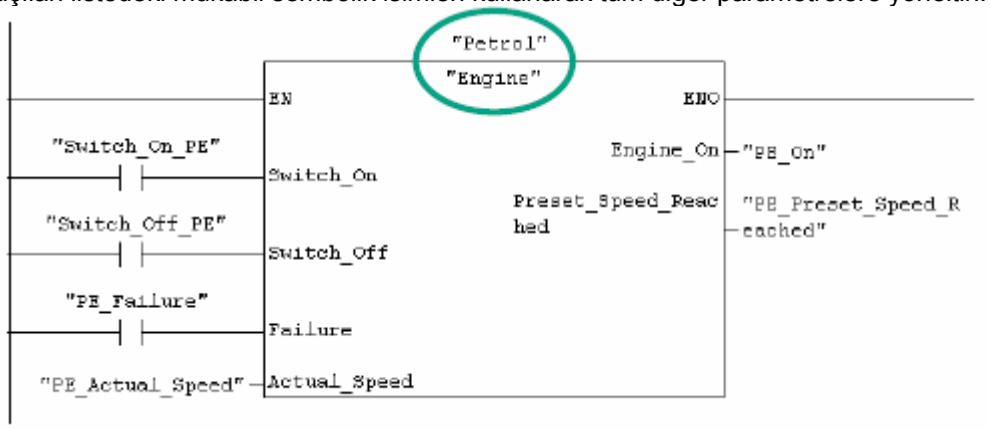


Kısayol menüsünde **Insert Symbol** (*Sembol Ekle*) üzerine sağ fare tuşu ile tıklayın. Aşağıya doğru bir liste gözükür.



Petrol (*Benzin*) Veri bloğu üzerine çift tıklayın. Sonra bu blok girdi çerçevesine otomatik olarak tırnak içinde girilir.

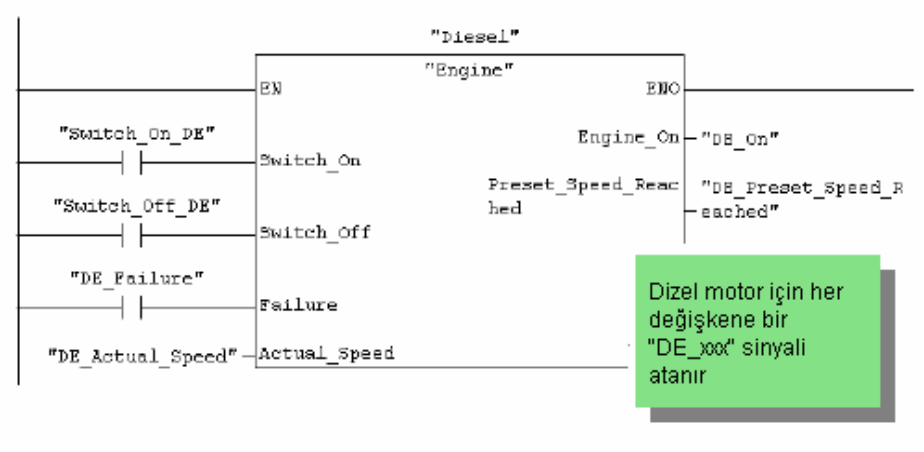
Soru işaretine tıklayın ve tırnak işaretilerini girdikten sonra fonksiyon bloğunun aşağı açılan listedeki mukabil sembolik isimleri kullanarak tüm diğer parametrelere yöneltin.



FB "Motor"unda motora özgü girdi ve çıktı değişkenleri ("in" ve "out" beyanı) gösterilir.

Benzinli motorun her değişkenine bir "PE_xxx" sinyali atanır.

Yeni bir ağda “Diesel” Veri Bloğu (DB2) ile fonksiyon bloğu için çağrı programlayın ve aşağı açılan listeden mukabil adresleri kullanın.



Programınızı kaydedin ve bloğu kapatın.

Organizasyon blokları, fonksiyon blokları ve veri blokları ile program yapıları oluşturduğunuz zaman, hiyerarşide onların üzerindeki blokta bağlı bloklar için (örnek **OB1** için) çağrı programlamalısınız. Yöntem her zaman aynıdır.

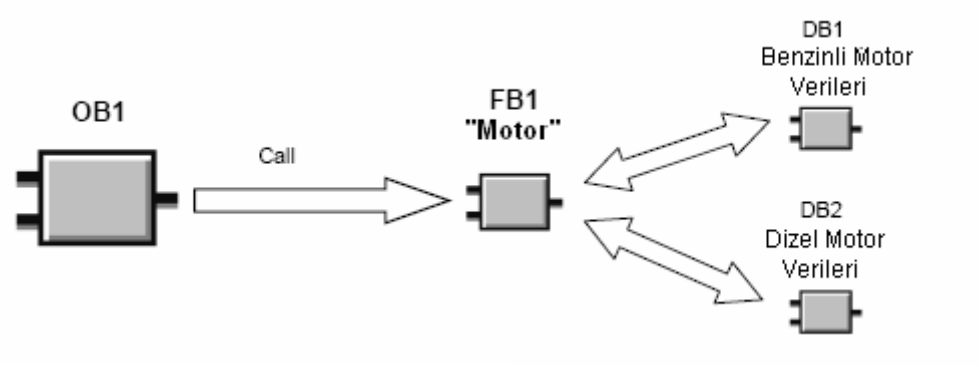
Çeşitli bloklarada sembol tablosundaki sembolik isimleri verebilirsiniz (örneğin, FB1’in adı “Motor”dur ve DB1’rinki ise “Benzin” olur).

Programlanmış blokları her zaman arşivleyebilir veya yazdırabllirsiniz. Mukabil fonksiyonları **File > Archive** (*Dosya* > *Arşiv*) veya **File > Print** (*Dosya* > *Yazdır*) altındaki menü komutlarında SIMATIC Yöneticisinde bulabilirsiniz.

"Language Description (*Dil Açıklaması*): LAD," ve "Program Control Instructions. (*Program Kontrol Talimatı*)" başlıklarındaki **Help > Contents** (*İçindekiler*) altında daha fazla bilgi bulabilirsiniz.

5.7 İfade Listesinde Blok Çağrı Programlaması

Bir fonksiyon bloğunu programlamakla yaptığınız tüm çalışma, siz bu bloğu bir OB1 bloğuna çağırmadıkça yararsızdır. Her fonksiyon bloğu çağrısı için bir Veri bloğu kullanılır ve bu yolla her iki motoru kontrol edebilirsiniz.



SIMATIC Yöneticisi "Başlarken" projenizle birlikte açılır.

Blocks (*Bloklar*) klasöründe gezinin ve **OB1**'i açın.

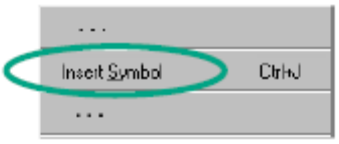
Network 3'ü seçin ve sonra LAD/STL/FBD program penceresinde network 4'ü ekleyin.

```
CALL  "Engine" , "Petrol"
Switch_On              :=
Switch_Off             :=
Failure                :=
Actual_Speed           :=
Engine_On              :=
Preset_Speed_Reached:=
```

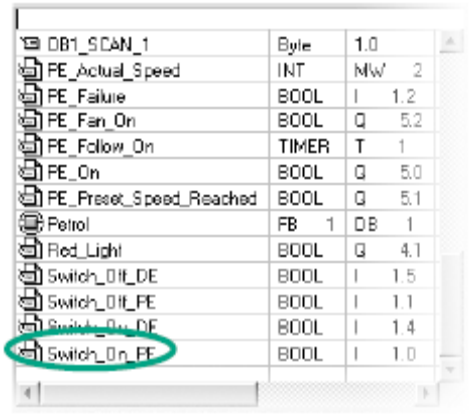
Kod seçiminde **CALL "Engine** (*Motor*)**" "Petrol** (*Benzinli*)**"** yazın ve sonra **Enter**'a basın.

"**Petrol**" Fonksiyon bloğunun tüm parametreleri gösterilir.

İmleci **Switch On**'un eşit işaretinden sonraya getirin ve sağ fare tuşuna tıklayın.



Kısayol menüsünde **Insert Symbol** (*Sembol Ekle*) üzerine sağ fare tuşu ile tıklayın. Aşağı açılan bir menü görüntülenir.



OB1_SCAN_1	Byte	1.0
PE_Actual_Speed	INT	MW 2
PE_Failure	BOOL	I 1.2
PE_Fan_On	BOOL	Q 5.2
PE_Follow_On	TIMER	T 1
PE_On	BOOL	Q 5.0
PE_Preset_Speed_Reached	BOOL	Q 5.1
Petrol	FB 1	DB 1
Red_Light	BOOL	Q 4.1
Switch_Off_DE	BOOL	I 1.5
Switch_Off_PE	BOOL	I 1.1
Switch_On_DE	BOOL	I 1.4
Switch_On_PE	BOOL	I 1.0

Switch_On_PE adını tıklayın. Bu aşağı açılan listeden alınır ve otomatik olarak tırnak içinde eklenir.

```
CALL  "Engine" , "Petrol"
 Switch_On            :="Switch_On_PE"
 Switch_Off           :="Switch_Off_PE"
 Failure              :="PE_Failure"
 Actual_Speed         :="PE_Actual_Speed"
 Engine_On            :="PE_On"
 Preset_Speed_Reached:="PE_Preset_Speed_Reached"
```

Aşağı açılan listeyi kullanarak blok fonksiyonunun değişkenleri için gereken tüm adresleri atayın.

Benzinli motor için her değişkene bir "PE_xxx" sinyali atanır

```
CALL  "Engine" , "Diesel"
 Switch_On            :="Switch_On_DE"
 Switch_Off           :="Switch_Off_DE"
 Failure              :="DE_Failure"
 Actual_Speed         :="DE_Actual_Speed"
 Engine_On            :="DE_On"
 Preset_Speed_Reached:="DE_Preset_Speed_Reached"
```

Yeni bir ağda "Engine (*Motor*)" (FB1) fonksiyon bloğunu “Diesel” (DB2) Veri Bloğu ile birlikte çağırmayı programlayın. Diğer çağrı ile aynı yolda ilerleyin.

Programınızı kaydedin ve bloğu kapatın.

Organizasyon blokları, fonksiyon blokları ve veri blokları ile program yapıları oluşturduğunuz zaman, hiyerarşide onların üzerindeki blokta bağlı bloklar için (örnek **OB1** için) çağrı programlamalısınız. Yöntem her zaman aynıdır.

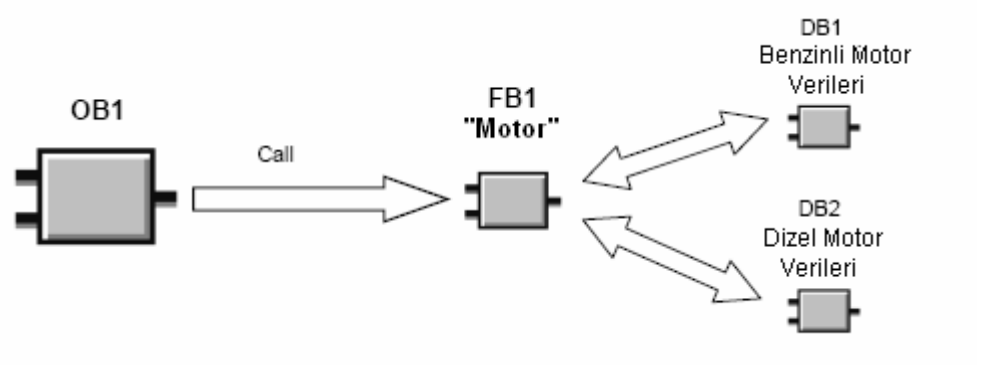
Çeşitli bloklara da sembol tablosundaki sembolik isimleri verebilirsiniz (örneğin, FB1’in adı “Motor”dur ve DB1’ınki ise “Benzin” olur).

Programlanmış blokları her zaman arşivleyebilir veya yazdırabllirsiniz. Mukabil fonksiyonları **File > Archive** (*Dosya > Arşiv*) veya **File > Print** (*Dosya > Yazdır*) altındaki menü komutlarında SIMATIC Yöneticisinde bulabilirsiniz.

"Language Description (*Dil Açıklaması*): STL," ve "Program Control Instructions. (*Program Kontrol Talimatı*)" başlıklarındaki **Help > Contents** (*İçindekiler*) altında daha fazla bilgi bulabilirsiniz.

5.8 Fonksiyon Blok Şemasında Blok Çağrı Programlaması

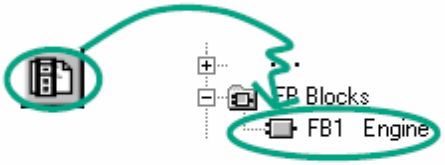
Bir fonksiyon bloğunu programlamakla yaptığınız tüm çalışma, siz bu bloğu bir OB1 bloğuna çağırmadıkça yararsızdır. Her fonksiyon bloğu çağrısı için bir Veri bloğu kullanılır ve bu yolla her iki motoru kontrol edebilirsiniz.



SIMATIC Yöneticisi "Başlarken" projenizle birlikte açılır.

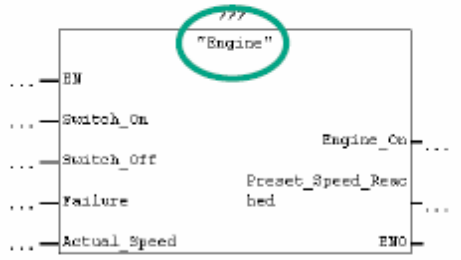
Blocks (*Bloklar*) klasöründe gezinin ve **OB1**'i açın.

Network 3'ü seçin ve sonra LAD/STL/FBD program penceresinde network 4'ü ekleyin.

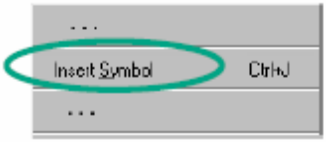


FB1'a ulaşıncaya kadar Program Elemanları katalogunda gezinin ve bu bloğu ekleyin.

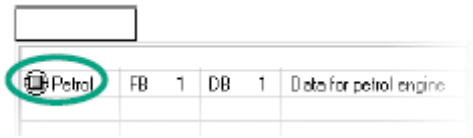
Tüm motora özgü girdi ve çıktı değişkenleri gösterilir.



"Engine (*Motor*)"un üzerindeki ??? İşaretine tıklayın ve sonra, imleci aynı konumda tutarak sağ fare tuşu ile girdi çerçevesine tıklayın.

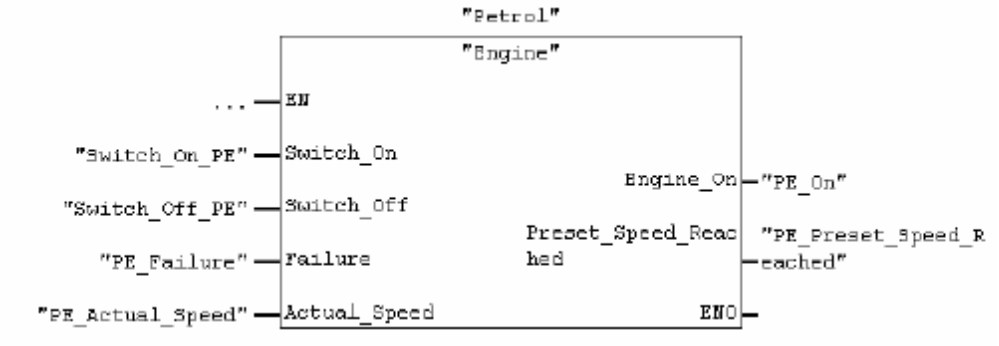


Kısayol menüsünde **Insert Symbol** (*Sembol Ekle*) üzerine sağ fare tuşu ile tıklayın.



Petrol (*Benzinli*) Veri Bloğuna çift tıklayın. Aşağı açılan listeden alınır ve girdi çerçevesine otomatik olarak tırnak içinde girilir.

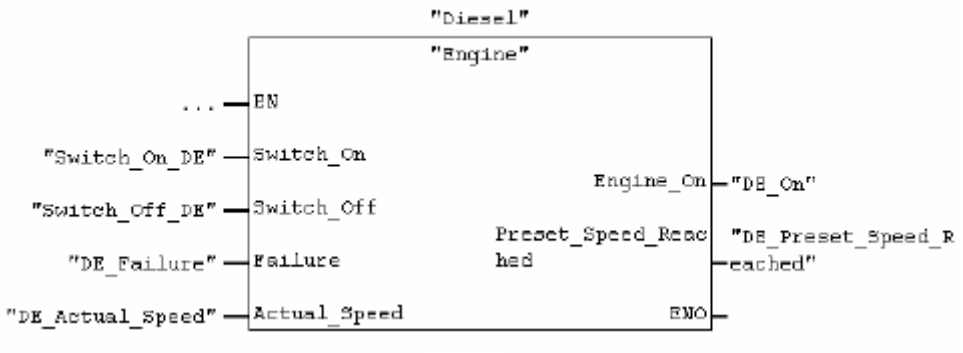
Aşağı açılan listedeki mukabil isimleri kullanarak tüm diğer fonksiyon bloğu parametrelerine yönelin.



Benzinli motor için her değişkene bir "PE_xxx" sinyali atanır

Yeni bir ağda “Diesel” Veri Bloğu (DB2) ile fonksiyon bloğu için çağrı programlayın ve aşağı açılan listeden mukabil adresleri kullanın.

Benzinli motor için her değişkene bir "PE_xxx" sinyali atanır



Programınızı kaydedin ve bloğu kapatın.

Organizasyon blokları, fonksiyon blokları ve veri blokları ile program yapıları oluşturduğunuz zaman, hiyerarşide onların üzerindeki blokta bağlı bloklar için (örnek **FB1** için) çağrı programlamalısınız. Yöntem her zaman aynıdır.

Çeşitli bloklara da sembol tablosundaki sembolik isimleri verebilirsiniz (örneğin, FB1’in adı “Motor”dur ve DB1’ınki ise “Benzin” olur).

Programlanmış blokları her zaman arşivleyebilir veya yazdırabllirsiniz. Mukabil fonksiyonları **File > Archive** (*Dosya* > *Arşiv*) veya **File > Print** (*Dosya* > *Yazdır*) altındaki menü komutlarında SIMATIC Yöneticisinde bulabilirsiniz.

"Language Description (*Dil Açıklaması*): LAD," ve "Program Control Instructions. (*Program Kontrol Talimatı*)" başlıklarındaki **Help > Contents** (*İçindekiler*) altında daha fazla bilgi bulabilirsiniz.

6 PLC Donanım Ayarları

6.1 Donanımın Yapılandırılması

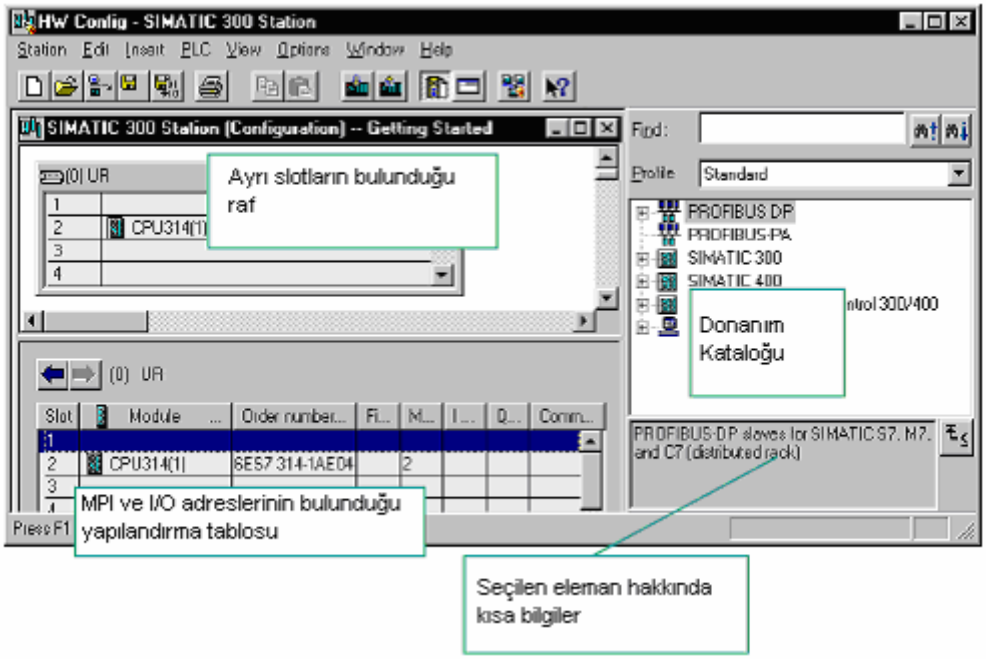
Bir SIMATIC istasyonu ile proje yaptığınız zaman donanımı yapılandırabilirsiniz.Bölüm 2.1'de STEP 7 Sihirbazı ile yapılan proje yapısı bunun bütün şartlarını karşılar.

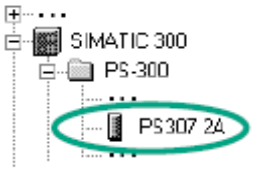
Donanım STEP 7 ile yapılandırılır. Bu yapılandırma verileri daha sonra "indirme" ile programlanabilir kontrol ediciye aktarılır (bak Bölüm 7).

Başlangıç noktası "Başlarken" projesi ile birlikte açık olan SIMATIC Yöneticisidir.

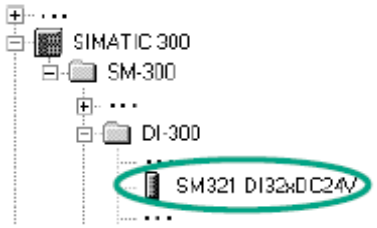
SIMATIC 300 Station klasörünü açın ve **Hardware** (*Donanım*) sembolüne çift tıklayın.

"HW Config" penceresi açılır. Proje hazırlarken seçtiğiniz CPU gösterilir. "Başlarken" projesi için bu CPU 314'tür.

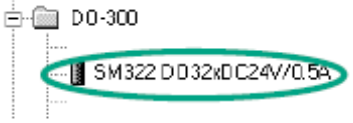




İlk önce bir güç besleme modülüne ihtiyacınız var. **PS307 2A**'ya ulaşıncaya kadar katalogda gezinin ve onu sürükleyip slot 1'deki yerine bırakın.



Girdi modülü (d1, Sayısal girdi) **SM321 DI32xDC24V**'yi buluncaya kadar gezinin ve onu da slot 4'teki yerine koyun. Slot 3 boş kalır.



Aynı şekilde çıktı modülü **SM322 DO32xDC24V/0.5A**'yı slot 5'e koyun.

Bir proje içinde bir modülün parametrelerini (örneğin, adresini) değiştirmek için modüle çift tıklayın. Sadece değişikliklerin programlanabilir kontrol edici üzerinde ne gibi etkileri olacağını biliyorsanız parametreleri değiştirmelisiniz.

"Başlarken" projesinde hiçbir değişiklik gerekli değildir.

Slot	Module	Order Number	MPI Address	I Add...	O...	Comment
1	PS307 2A	6ES7 307-1BA00-0AA0				
2	CPU314(1)	6ES7 314-1AE04-0AB0	2			
3						
4	DI32xDC24V	6ES7 321-1BL00-0AA0		0...3		
5	DO32xDC24V/0.5A	6ES7 322-1BL00-0AA0			4...7	
6						
7						
8						
9						
10						
11						

Veriler **Save and Compile** (*Sakla ve derle*) menü komutu kullanarak aktarılmaya hazırdır**.**

"HW Config" uygulamasını kapatır kapatmaz, Bloklar klasöründe Sistem Verileri sembolü gözükecektir.

Yapılandırmanızın hatası olup olmadığını **Station > Consistency Check** (*İstasyon > Tutarlılık Kontrolü*) menü komutunu kullanarak da kontrol edebilirsiniz. STEP 7 olabilecek hatalarla ilgili olası çareleri size sağlayacaktır.

"Configuring theHardware (*Donanımın Yapılandırılması*)" ve "Configuring Central Racks (*Merkezi Rafların Yapılandırılması*)" başlıklarındaki **Help > Contents** (*İçindekiler*) altında daha fazla bilgi bulabilirsiniz.

Programın İndirilmesi ve Hata ayıklanması

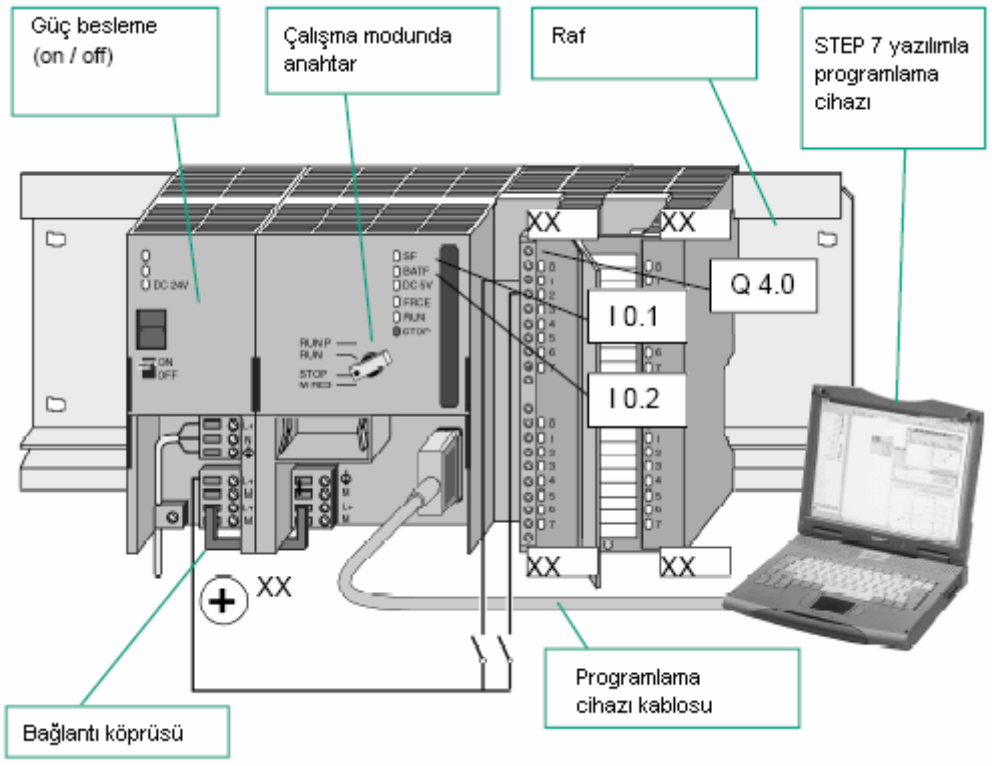
7.1 Çevrimiçi Bağlantı Kurulması

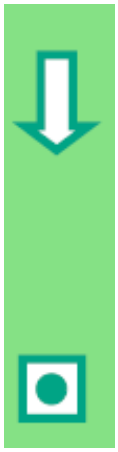
Verilen "GS-LAD_Örnek" veya yaptığınız "Başlarken" projesini ve basit bir test yapılandırmasını kullanarak, size programlanabilir kontrol ediciye (PLC) programın nasıl indirilebileceğini ve hatalarının ayıklanabileceğini göstereceğiz.

Gerekenler:

- “Başlarken” projesi için yapılandırılmış donanım (bak Bölüm 6)
- Donanımın kullanım elkitabına göre kurulması

Bir seri devre örneği (AND (*VE*) fonksiyonu):
Anahtar I 0.1 **and** (*ve*) Anahtar I 0.2 aynı anda basılı olmadıkça Q 4.0 yanmaz (Q 4.0 diyotu sayısal çıktı modülünde yanar). Kablolar ve CPU’nuzu kullanarak test yapılandırmasını kurun.





Donanımın yapılandırılması

Bir modülü ray üzerine takmak için aşağıda verilen sıra ile hareket edin:

- Modülü bar bağlayıcısına takın
- Modülü ray üzerine asın ve aşağı doğru sallayın
- Modülü yerine vidalayın
- Kalan modülleri takın
- Tüm modülleri takmayı bitirdikten sonra anahtarı CPU'ya sokun.

Şemada gösterilenden farklı bir donanım kullanıyor olsanız bile test yapabilirsiniz. Girdilerin ve çıktıların adreslemesine bağlı kalmanız yeterli olacaktır. STEP 7 programınızın hatalarını ayıklamanın çeşitli yollarını sunmaktadır; örneğin, program statüsünü kullanabilir veya değişken tablasundan yararlanabilirsiniz.

PLC Donanım ayarları hakkında "S7-300, Donanım ve Kurulum / Modül Teknik Özellikleri" ve "S7-400 / M7-400 - Donanım" elkitaplarında daha fazla bilgi bulabilirsiniz.

7.2 Programın Programlanabilir Kontrol Ediciye İndirilmesi

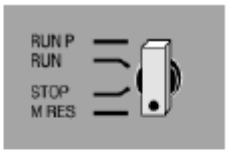
Programı indirmek için önceden bir çevrimiçi bağlantı kurmuş olmalısınız.

Akım Verilmesi

ON/OFF (*AÇ/KAPA*) şalterini kullanarak cereyanı açın. CPU üzerindeki “DC 5V”’lük diyot yanacaktır.

İşletme modu anahtarını STOP konumuna çevirin (STOP’ TA değilse). Kırmızı "STOP" LED yanacaktır

CPU’nun sıfırlanması ve CPU’nun RUN (*ÇALIŞTIR*)’a Çevrilmesi



İşletme modu anahtarını **MRES** konumuna getirin ve kırmızı “STOP” LED yavaş yavaş yanıp sönünceye kadar en az 3 saniye orada tutun.

Bellek sıfırlaması CPU üzerindeki tüm verileri siler. O zamanCPU başlangıç durumuna döner.

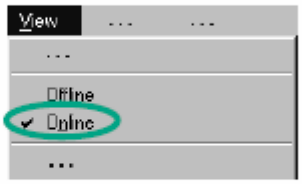
Anahtarı bırakın ve en fazla 3 saniye sonra, tekrar **MRES** konumuna çevirin. "STOP" LED hızla yanıp söndüğü zaman, CPU sıfırlanmış olur.

"STOP" LED hızla yanıp sönmeye başlamazsa, işlemi tekrarlayın.

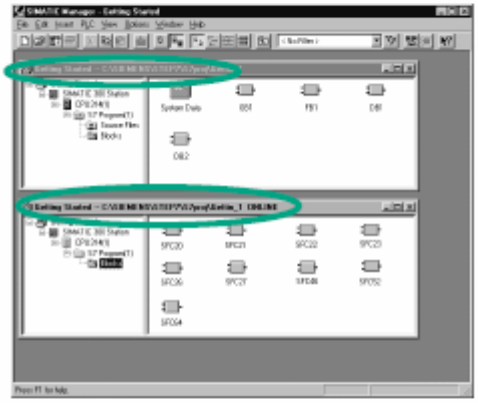
Programın CPU’ya İndirilmesi

Şimdi programı indirmek için işletme modu düğmesini tekrar “STOP”’a çevirin.

SIMATIC Yöneticisini başlatın ve “Open (*Aç*)” diyalog kutusunda “Başlarken” projesini açın (önceden açık değilse).



“Çevrimdışı Başlarken” penceresine ek olarak, “ÇEVRİMİÇİ Başlarken” penceresini de açın. Çevrimiçi veya çevrimdışı durumu farklı renkli başlıklarla gösterilir.

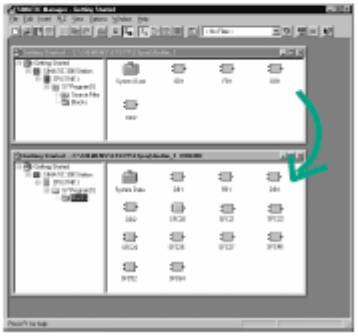


Her iki pencerede de **Blocks** (*Bloklar*) penceresine gezinin.

Çevrimdışı pencere programlama cihazındaki durumu gösterir; çevrimiçi pencere ise CPU’daki durumu gösterir.

Bellek sıfırlaması yapsanız bile sistem fonksiyonları (SFCs) CPU'de kalır. CPU işletim sisteminin bu fonksiyonlarını sağlar. Onların indirilmesi gerekmez, ancak onlar silinemez.

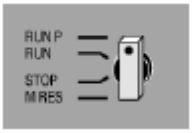
Çevrimdışı pencerede **Blocks** klasörünü seçin ve sonra **PLC** > **Download** menü komutunu kullanarak programı CPU’ya indirin. İstem talebini **OK** ile doğrulayın.



Program bloklarını indirdiğiniz zaman çevrimiçi pencerede gösterilir.

Araç çubuğunda veya açılır menüdeki mukabil tuşu kullanarak sağ fare tuşuyla PLC > Download menü komutunu da çağırabilirsiniz.

CPU Anahtarının Açılması ve İşletim Modunun Kontrol Edilmesi



İşletim Modu anahtarını **RUN-P**'ye getirin. Yeşil "RUN (*ÇALIŞIYOR*) LED yanar ve kırmızı "STOP" LED söner. CPU çalışmaya hazırdır.

Yeşil LED yandığı zaman programı test etmeye başlayabilirsiniz.

LED yanık kalmaya devam ederse, bir hata oluşmuştur. Hatayı bulmak için hata bulma tamponunu değerlendirirsiniz.

Ayrı blokların indirilmesi

Hatalara uygulamada hızla cevap vermek için, bloklar CPU'ya birer birer sürükle bırak fonksiyonu ile aktarılabilir.
Blokları indirdiğiniz zaman, CPU üzerindeki işletim modu anahtarı ya "RUN-P" ya da "STOP" modda olmalıdır. "RUN-P" modunda indirilen bloklar hemen etkinleşir. Bu nedenle aşağıdakileri unutmamalısınız:

- Eğer hatasız bloklar hatalı blokların üzerine yazılmışsa bu bir tesis bozulmasına yol açacaktır. Blokları indirmeden önce test ederek bunu önleyebilirsiniz.
- Blokların indirileceği sırayı– önce alt düzeyde blokları ve sonra daha yüksek düzeyde blokları – dikkate almazsanız CPU "STOP" moduna girecektir. Tüm programı CPU'ya indirerek bunun önüne geçebilirsiniz.

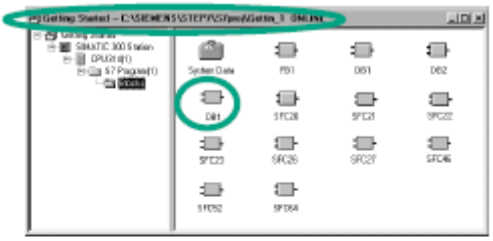
Çevrimiçi programlama

Pratikte test amaçları için blokları önceden CPU'ya indirilen blokları değiştirmeniz gerekmiş olabilir. Bunu yapmak için, LAD/STL/FBD program penceresini açmak için çevrimiçi pencerede gerekli bloğa çift tıklayın. Sonra bloğu olağan şekilde programlayın. Programlanan bloğun CPU'da hemen etkinleşeceğini unutmayın.

Help > Contents (*Yardım* > *İçindekiler*) altında ve sonra "İndirme ve Yükleme" ile "Çevrimiçi Bağlantı Kurulması ve CPU Ayarlarının Yapılması" konularında daha fazla bilgi bulabilirsiniz.

7.3 Program Statüsü ile Programın Test Edilmesi

Program statüsü fonksiyonunu kullanarak programı blok halinde test edebilirsiniz. Bunun için gereken CPU ile çevrimiçi bağlantı kurulmuş olması, CPU'nun RUN veya RUN-P modunda olması ve programın indirilmiş olmasıdır.

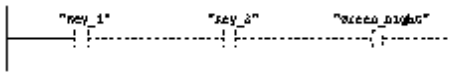


"ÇEVRİMİÇİ Başlarken" penceresinde **OB1**'i açın.

LAD/STL/FBD program penceresi açılır.

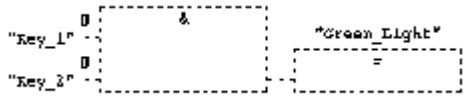
Debug > Monitor fonksiyonunu etkinleştirin.

Sıralama Mantığı ile Hata Ayıklama



Network 1'de seri devre Sıralama Mantığında gösterilir. Geçerli yol Key 1 (I 0.1)'e kadar bütün bir hat olarak temsil edilir; bunun anlamı devrede akım bulunduğudur.

Fonksiyon Blok Şeması ile Hata Ayıklama



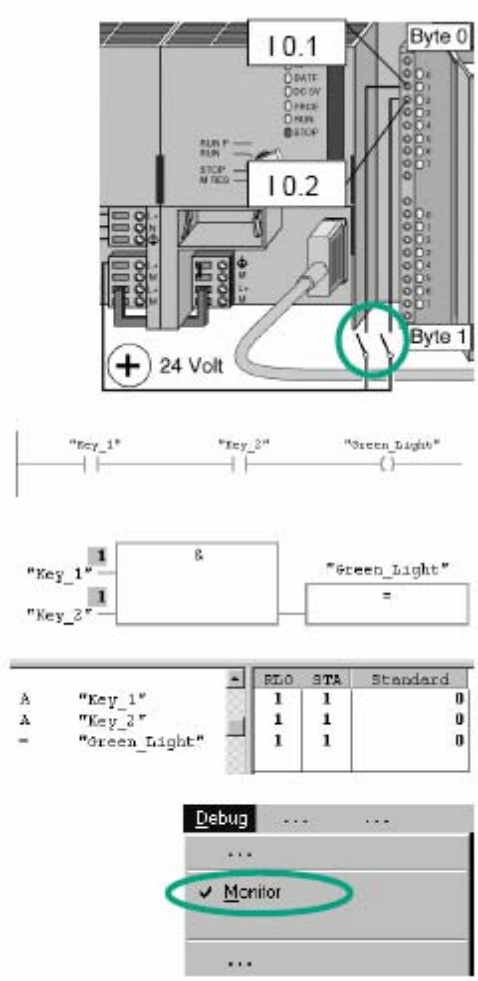
Sinyal durumu "0" ve "1" olarak gösterilir. Noktalı hat mantıksal işlemin sonucunun olmadığı anlamındadır.

İfade Listesi ile Hata Ayıklama

İfade Listesi için tablo halinde aşağıdakiler gösterilir:

- Mantıksal işlemin sonucu (RLO)
- Durum biti (STA)
- Standart durum (STANDARD)

Options > Customize (Seçenekler > Özelleştir)'i kullanarak programlama dilinin kullanılış tarzını test sırasında değiştirebilirsiniz.



Şimdi test yapılandırmasındaki iki anahtara basın.

Girdi modülünde I 0.1 ve I 0.2 girdilerinin diyotları yanar. Çıktı modülünde de Q 4.0 çıktısının diyotu yanar.

Sıralama Mantığı ve Fonksiyon Blok Şeması grafiksel programlama dillerinde, programlanmış ağdaki renk değişimini izleyerek test sonucunu izleyebilirsiniz. Bu renk değişimi mantıksal işlemin sonucunun bu noktaya kadar yerine getirildiğini gösterir.

Mantıksal işlemin yerine getirilmesi ile İfade Listesi Programlama dili ile STA ve RLO sütunlarındaki gösterim değişir.

Debug > Monitor fonksiyonunun etkinliğini kaldırın ve pencereyi kapatın.

Sonra SIMATIC Yöneticisindeki çevrimiçi pencereyi kapatın.

Geniş kapsamlı programları çalıştırmak için CPU'nuza tam olarak yüklememenizi tavsiye ederiz, çünkü Olası hata kaynaklarının çok fazla olması nedeniyle hataların bulunması daha zor olur. Bunun yerine blokları birer birer indirmeli ve daha iyi inceleyebilmek için sonra test etmelisiniz.

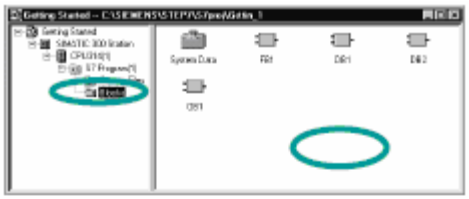
"Temizleme" ve "Program Statüsü ile Test Etme" konularındaki **Help > Contents** (*Yardım* > *İçindekiler*) altında daha fazla bilgi bulabilirsiniz.

7.4 Değişken Tablo ile Programın Test Edilmesi

Ayrı program değişkenlerini izleyerek ve değiştirerek test edebilirsiniz. Bunun için gereken CPU ile çevrimiçi bağlantı kurulmuş olması, CPU'nun RUN veya RUN-P modunda olması ve programın indirilmiş olmasıdır.

Program statüsünü test ederken değişken tablosunda Network 1 (seri devre veya AND (VE) fonksiyonu)'da girdileri ve çıktıları izleyebilirsiniz. FB1'de motor hızının karşılaştırıcısını da gerçek hızı sıfırlayarak test edebilirsiniz.

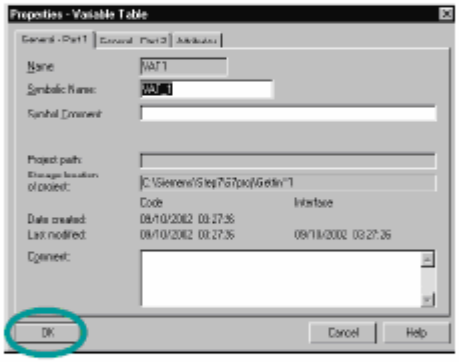
Değişken Tablo Oluşturulması



Başlama noktası yine "Çevrimdışı Başlarken" proje penceresi açık olan SIMATIC Yöneticisidir.

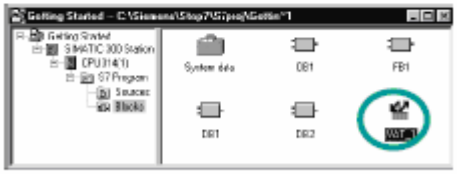
Blocks klasörüne gezinin ve pencerenin sağ yarısına sağ fare tuşu ile tıklayın.

Açılır menüden bir **Değişken Tablo** eklemek için sağ fare tuşunu kullanın.



"Properties (*Özellikler*)" diyalog kutusunu KO ile kapatarak varsayılan ayarları uygulayın.

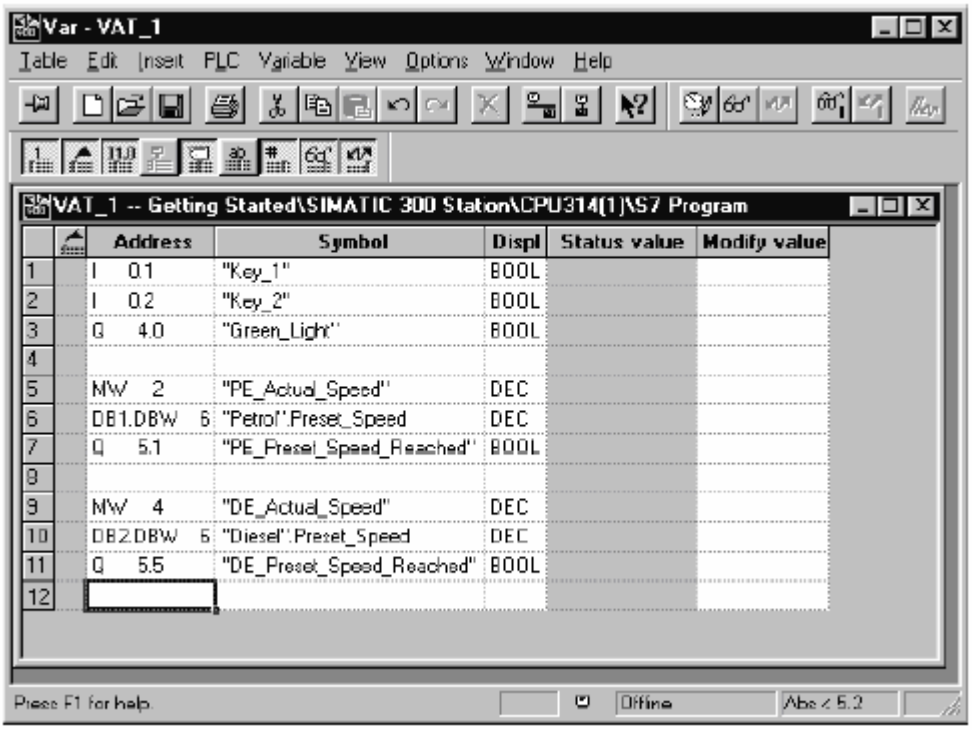
Başka bir seçenek olarak da değişken tablosuna bir sembol adı verebilir ve bir sembol yorumu girebilirsiniz.



Blocks klasöründe bir VAT1 (değişken tablo) oluşturulur.

VAT1'i açmak için çift tıklayın, "Değişkenlerin İzlenmesi ve Değiştirilmesi" penceresi yeniden açılır.

Başlangıçta değişkenler tablosu boştur. “Başlarken” örneği için aşağıdaki örneğe göre sembolik isimleri veya adresleri girin. Girdilerinizi Enter ile tamamladığınız zaman kalan ayrıntılar eklenmiş olacaktır.

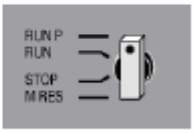


Değişken tablonuzu kaydedin.

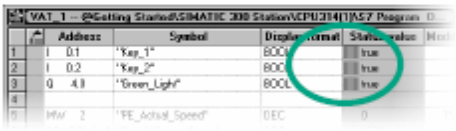
Değişken Tabloya Çevrimiçi Anahtarlama

Yapılandırılmış CPU ile bir bağlantı kurun. CPU’nun işletim modu durum çubuğunda görüntülenir.

CPU’nun anahtarını **RUN-P** olarak ayarlayın (daha önce yapmadıysanız)..



Değişkenlerin İzlenmesi



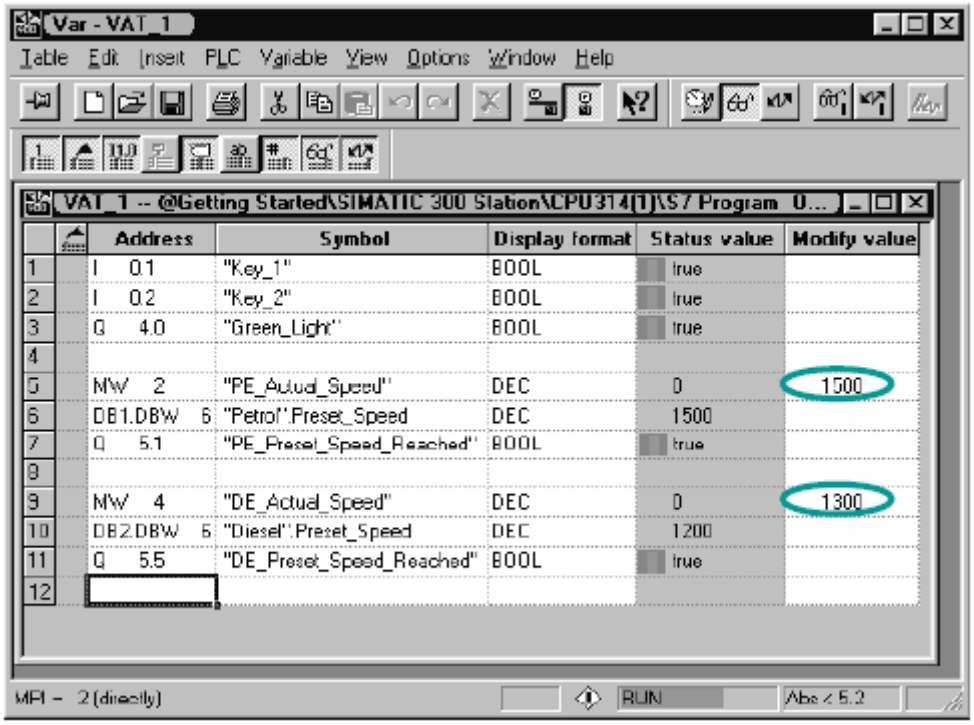
Ataç çubuğundaki **Monitor Variables** (*Değişiklikleri İzle*) düğmesine tıklayın.

Test yapılandırmanızda Key 1 ve Key 2'ye basın ve değişken tablosunda sonucu izleyin.

Değişken tablosundaki durum değerleri yanlıştan doğruya değişecektir.

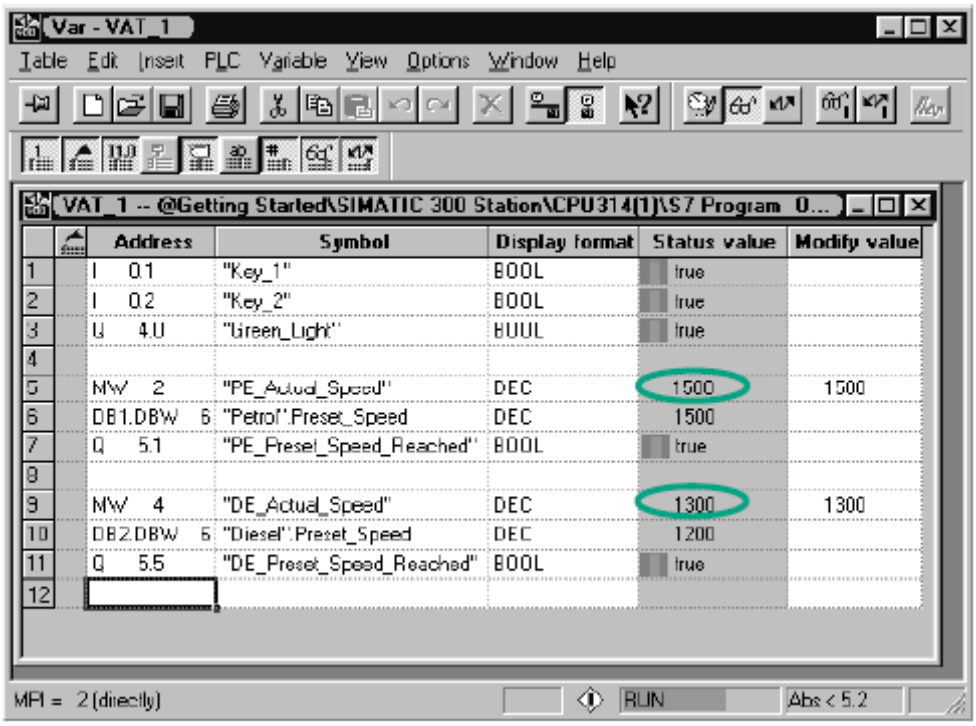
Değişkenlerin değiştirilmesi

Değeri Değiştir sütununda MW2 adresi için "1500" ve MW4 adresi için "1300" girin.



Değişik değerleri CPU'nuza aktarın.

Aktarmadan sonra, bu değerler CPU'nuzda işlenecektir. Karşılaştırmanın sonucu görünür hale gelir.



Sınırlı ekran alanı nedeniyle çok büyük değişken tabloları genelde tam olarak gösterilemez. Eğer büyük değişken tablolarınız varsa STEP 7 kullanan bir S7 programı için birkaç tablo yapmanızı tavsiye ederiz. Değişken tablolarını kendi test şartlarınıza kesin olarak uyarlayabilirsiniz.

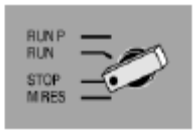
Ayrı değişken tablolarına bloklar için olduğu gibi ayrı isimler (örneğin VAT1 yerine OB1_Network 1) verebilirsiniz. Yeni isimler vermek için sembol tablosunu kullanın.

"Değişken Tablo ile Test Etme"deki "Debugging" konularındaki **Help > Contents** (*Yardım > İçindekiler*) altında daha fazla bilgi bulabilirsiniz.

7.5 Diagnostic Buffer (*Arıza Bulma Tamponu*)nun Değerlendirilmesi

Eğer, aşırı bir durumda, S7 programında CPU STOP'a giderse veya programı indirdikten sonra CPU'yu RUN durumuna çeviremezseniz, arıza bulma tamponunda listelenen olaylardan hatanın sebebini belirleyebilirsiniz.

Bunun için gereken CPU ile çevrimiçi bağlantı kurulmuş olması, CPU'nun RUN modunda olmasıdır.



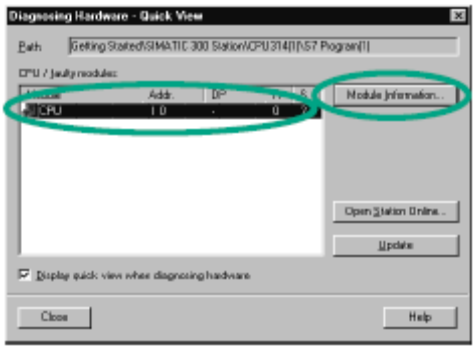
Önce CPU'daki işletim modu anahtarını STOP'a çevirin.

Başlama noktası yine "Çevrimdışı Başlarken" proje penceresi açık olan SIMATIC Yöneticisidir.

Blocks klasörünü seçin.

Projenizde birkaç CPU varsa, önce hangi CPU'nun STOP'a girdiğini belirleyin.

Tüm erişilebilir CPU'lar "Diagnosing Hardware (*Donanımın Arızasının Bulunması*)" diyalog kutusunda listelenir. STOP işletim modundaki CPU'nun altı çizilir.

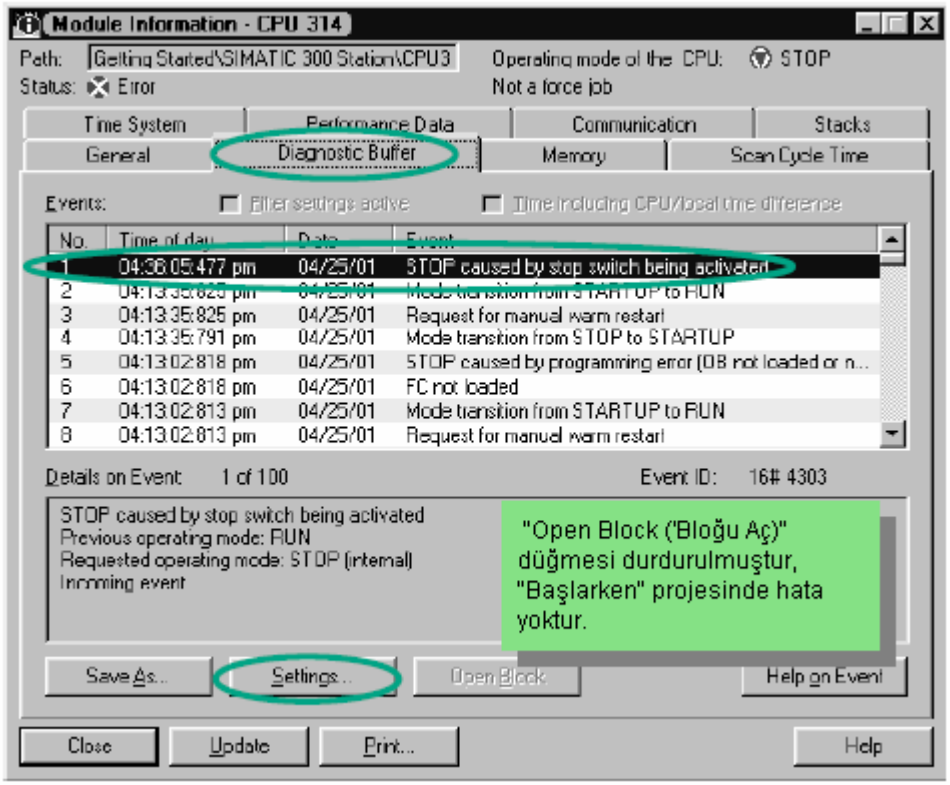


"Başlarken" projesinin gösterilen bir CPU'su vardır.

Bu CPU'nun arıza bulma tamponunu değerlendirmek için **Module Information** (*Modül Bilgileri*)ne tıklayın.

Sadece tek bir CPU bağlı ise bu CPU'nun modül bilgilerini doğrudan, PLC > Module Information menü komutunu kullanarak araştırabilirsiniz.

"Module Information" penceresi size CPU'nuzun özellikleri ve parametreleri hakkında bilgiler verir. Şimdi STOP halinin nedenini belirlemek için "Daiagnostic Buffer (*Arıza Bulma Tamponu*)" sekmesini seçin.



En son olay (1 numara) listenin başındadır. STOP halinin nedeni gösterilir. SIMATIC Yöneticisi hariç tüm pencereleri kapatın.

Bir programlama hatası CPU'nun STOP moda girmesine neden olursa, olayo seçin ve "Open Block" düğmesine basın.

Blok, o zaman, bilinen LAD/STL/FBD program penceresinde açılır ve hatalı ağın altı çizilir.

Bu bölümle birlikte proje yapmadan bitmiş projenin hatalarının ayıklanmasına kadar, "Başlarken" örnek projesini başarıyla tamamladınız. Sonraki bölümde bilgilerinizi seçilen alıştırmalarla genişletebilirsiniz.

"Modül Bilgilerinin Çağrılması konusundaki "Diagnostics (*Arıza Bulma*)" konularındaki **Help > Contents** (*Yardım > İçindekiler*) altında daha fazla bilgi bulabilirsiniz.

8 Bir Fonksiyonun Programlanması

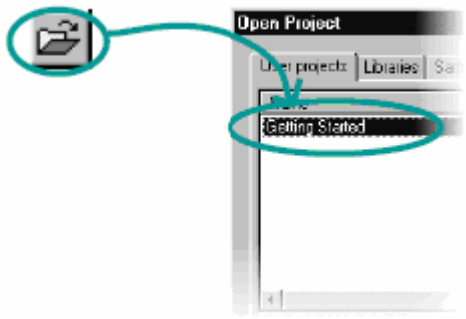
8.1 Fonksiyonların Oluşturulması ve Açılması (FC)

Fonksiyonlar, fonksiyon blokları gibi, program hiyerarşisinde organizasyon bloğunun altında yer alır. Bir fonksiyonun CPU tarafından işlenmesi için hiyerarşide onun da üstünde çağrılmalıdır. Ancak, fonksiyon bloğunun aksine, hiç veri bloğu gerekli değildir.

Fonksiyonlarla birlikte parametreler de değişken açıklama tablosunda listelenir, fakat statik yerel verilere izin verilmez.

Bir programı LAD/STL/FBD program penceresini kullanarak fonksiyon bloğu ile aynı yolla programlayabilirsiniz.

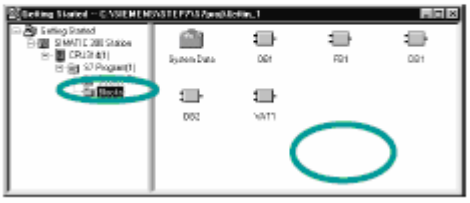
Sıralama Mantığında, Fonksiyon Blok Şeması veya İfade Listesinde programlamayı (bak Bölüm 4 ve 5) ve sembolik programlamayı da (bak Bölüm 3) iyi biliyor olmalısınız.



"Başlarken" projesini Bölüm 1'den 7'ye kadar iyice incelediyseniz şimdi bu pencereyi açın.

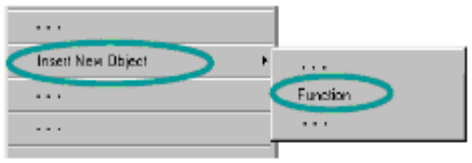
Değilse, **File > "New Project" Wizard** (*Dosya > "Yeni Proje" Sihirbazı*) menü komutunu kullanarak yeni bir proje yapın. Bunu yapmak için Bölüm 2.1'deki talimatı izleyin ve "Başlarken Fonksiyonu" adını verin.

"Başlarken" projesi ile devam edeceğiz. Yine de yeni bir proje kullanarak her adımı uygulamaya devam edebilirsiniz.

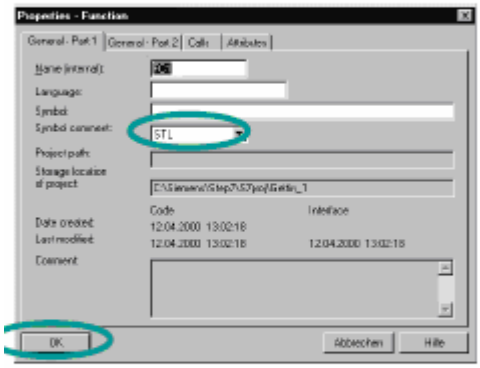


Blocks klasörüne gezinin ve onu açın.

Pencerenin sağ yarısını sağ fare tuşu ile tıklayın.

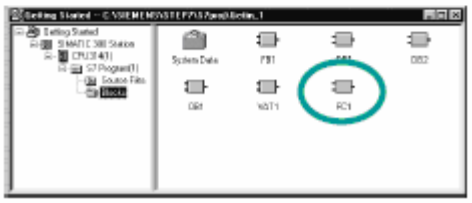


Açılır menüden bir **Fonksiyon** (FC) ekleyin.



"Properties – Function (*Özellikler – Fonksiyon*)" diyalog kutusunda FC1 adını kabul edin ve istenen programlama dilini seçin.

Kalan varsayılan ayarları OK ile onaylayın.



FC1 fonksiyonu Blocks klasörüne eklenir.

FC!'i açmak için çift tıklayın.

Fonksiyon bloğunun aksine, açıklama tablosunda bir fonksiyon için hiç statik veri tanımlanamaz.

Bir fonksiyon bloğunda tanımlanan statik veriler blok kapandığı zaman tutulur. Statik veriler, örmeğin, "Hız" sınır değerleri (bak Bölüm 5) için kullanılan bellek bitleri olabilir.

Bir fonksiyonu programlamak için sembol tablosundan sembolik isimleri kullanabilirisiniz.

"Otomasyon Kavramının İncelenmesi", "Bir Program Yapısını tasarlamanın Temelleri" ve "Kullanıcı Programı Altındaki Bloklar" konularındaki **Help > Contents** (*Yardım > İçindekiler*) altında daha fazla bilgi bulabilirsiniz.

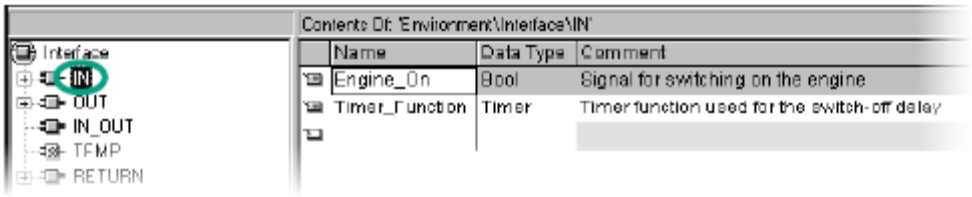
8.2 Fonksiyonların Programlanması

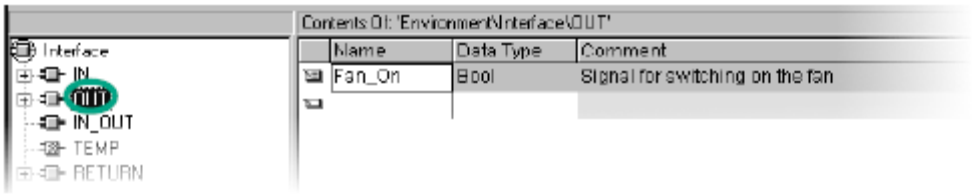
Bu bölümdeki örneğimizde bir zamanlayıcı fonksiyonu programlayacaksınız. Zamanlayıcı fonksiyonu bir motor çalıştırıldığı sürece bir fanın da çalıştırılmasını sağlar (bak Bölüm 5), ve motor durdurulduktan sonra fan dört saniye süre ile çalışmaya devam eder (durdurma ertelemesi).

Önceden bahsedildiği gibi, değişken ayrıntı görünüm tablosunda fonksiyonun girdi ve çıktı parametrelerini ("giriş" ve "çıkış" açıklaması) belirtmelisiniz.

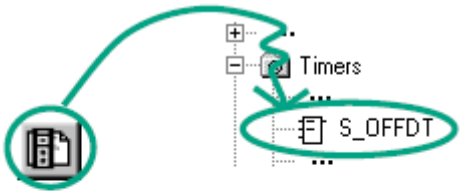
LAD/STL/FBD program penceresi açılır. Bu değişken ayrıntı görünümü ile fonksiyon bloğu ile aynı şekilde çalışacaksınız (bak Bölüm 5).

Aşağıdaki açıklamaları girin:



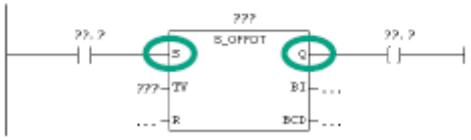


Sıralama Mantığında Zamanlayıcı Fonksiyonunun Programlanması



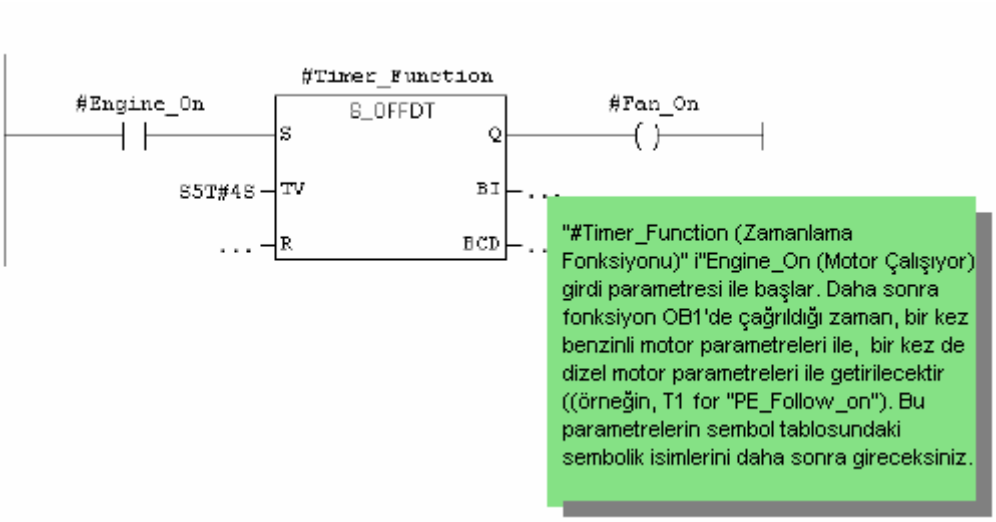
Sıralama talimatını girmek için geçerli yolu girin.

Program elemanları katalogunda **S_OFFDT** (gecikme ertelemesi zamanlayıcıyı başlat) elemanına ulaşıncaya kadar gezinin ve elemanı seçin.



S girdisinin önüne normal olarak açık bir kontak koyun.
Q çıktısından sonra bir bobin koyun.

Soru işaretlerini seçin, "#" girin ve mukabil isimleri seçin.



İfade Listesinde Zamanlayıcı Fonksiyonunun Programlanması

```
A     #Engine_On
L     S5T#4S
SF    #Timer_Function
A     #Timer_Function
=     #Fan_On
```

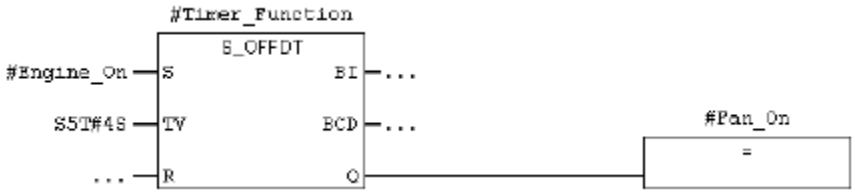
İfade Listesinde programlama yapıyorsanız ağın aşağısında bir girdi bölgesi seçin ve ifadeyi burada gösterilen şekilde girin.

Sonra fonksiyonu kaydedin ve pencereyi kapatın.

Fonksiyon Blok Şemasında Zamanlayıcı Fonksiyonunun Programlanması

Fonksiyon Blok Şemasında programlama yapıyorsanız, ağın aşağısında bir girdi bölgesi seçin ve FBD programını burada gösterilen şekilde girin.

Sonra fonksiyonu kaydedin ve pencereyi kapatın.



Bir zamanlayıcı fonksiyonun işlenebilmesi için blok hiyerarşisinde daha yüksekteki bir bloktaki bir fonksiyonu (örneğimizde OB1) çağırmanız gerekir.

"Referans Yardımlarının Çağrılması", " STL, FBD, veya LAD Dil Açıklaması" ve "Zamanlayıcı Fonksiyonları"" konularındaki **Help > Contents** (*Yardım* > *İçindekiler*) altında daha fazla bilgi bulabilirsiniz.

8.3 Fonksiyonun OB1'de Çağrılması

FC1 fonksiyonu için çağrı OB1'deki fonksiyon bloğu için çağrı ile aynı yolla yapılır. Fonksiyonun tüm parametreleri benzinli veya dizel motorların mukabil adresleri ile birlikte OB1'de sağlanır.

Bu adresler sembol tablosunda henüz tanımlanmadığından adreslerin sembolik isimleri şimdi eklenecektir.

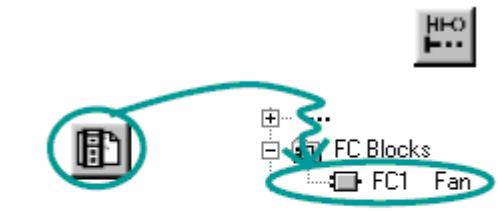
Adres STEP 7 ifadesinin bir parçasıdır ve işlemcinin talimatı nerede yerine getireceğini gösterir. Adresler mutlak da olabilir sembolik de.

SIMATIC Yöneticisi "Başlarken" projesi ile ya da yeni projenizle birlikte açılır.

Blocks klasöründe gezinin ve **OB1**'i açın.

LAD/STL/FBD program penceresi açılır.

Sıralama Mantığında Çağrı Programlaması



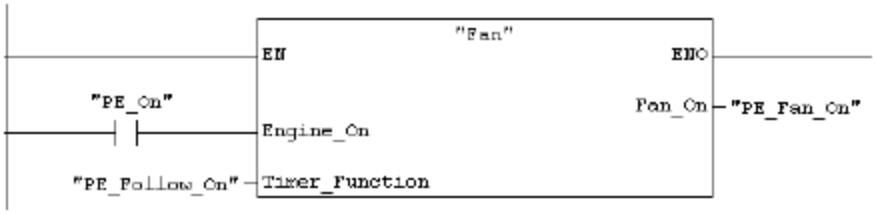
LAD görünümündesiniz. Network No. 5'i seçin ve yeni ağ No. 6'yı ekleyin.

FC1'e ulaşıncaya kadar program Elemanları katalogunda gezinin ve fonksiyonu ekleyin.

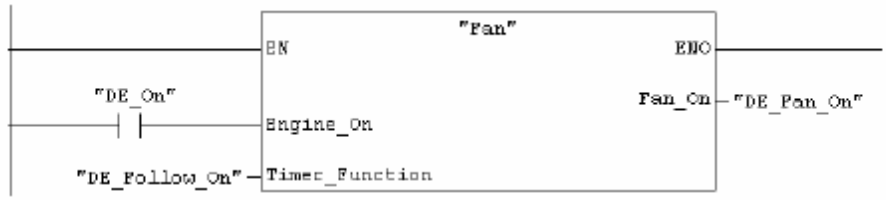
"Engine_On"un önüne normal olarak açık bir kontak ekleyin.

View > Display > Symbolic Representation (Görünüm > Göster > Sembolik Gösterim) menü komutunu kullanarak sembolik ve mutlak adresler arasında geçiş yapabilirsiniz.

FC1 çağrısı için soru işaretlerine tıklayın ve sembolik isimleri ekleyin.



Network 7'de, Dizel motorun adreslerini kullanarak, FC1 fonksiyonu için bir çağrı programlayın. Bunu önceki ağla aynı şekilde yapabilirsiniz (sembol tablosuna dizel motor için adresleri önceden eklemiştiniz).



Bloğu kaydedin ve sonra pencereyi kapatın.

Her ağdaki adresler hakkındaki bilgileri görüntülemek için View > Display > Symbol Information menü komutunu etkinleştirin.
Çeşitli ağları ekranda görüntülemek için, View > Display > Comment ve, gerekli ise, View > Display > Symbol Information menü komutunun etkinliğini kaldırın.
View > Zoom Factor menü komutunu kullanarak gösterilen ağın büyüklüğünü değiştirebilirsiniz.

İfade Listesinde Çağrı Programlaması

```
Network 6 : Controlling the Fan for the Petrol Engine
    CALL  "Fan"
     Engine_On      :="PE_On"
     Timer_Function:="PE_Follow_On"
     Fan_On         :="PE_Fan_On"

Network 7 : Controlling the Fan for the Diesel Engine
    CALL  "Fan"
     Engine_On      :="DE_On"
     Timer_Function:="DE_Follow_On"
     Fan_On         :="DE_Fan_On"
```

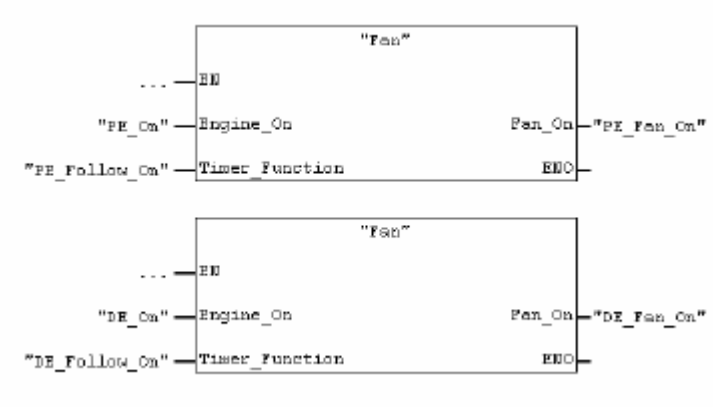
İfade Listesinde programlama yapıyorsanız, yeni bir ağın aşağısında bir girdi bölgesi seçin ve STL ifadelerini burada gösterilen şekilde girin.

Sonra çağrıyı kaydedin ve pencereyi kapatın.

Fonksiyon Blok Şemasında Çağrı Programlanması

Fonksiyon Blok Şemasında programlama yapıyorsanız, ağın aşağısında bir girdi bölgesi seçin ve FBD programını burada gösterilen şekilde girin.

Sonra fonksiyonu kaydedin ve pencereyi kapatın.



Fonksiyonlar için çağrı, örneğimizde, şartsız bir çağrı olarak programlanmıştır, yani, fonksiyon her zaman işlenecektir.

Otomasyon görevinizin gereksinimlerine bağlı olarak bir fonksiyon veya fonksiyon bloğu için belli şartlara, örneğin, bir girdiye veya önceki bir mantıksal işleme bağlı bir çağrı yapabilirsiniz. EN girdis veya ENO çıktısı programlama kutusunda verilir.

Help > Contents (*Yardım* > *İçindekiler*) altında ve sonra "LAD, FBD, veya STL Dili Açıklaması" konularındaki "Referans Yardımları Çağırma" altında daha fazla bilgi bulabilirsiniz.

.

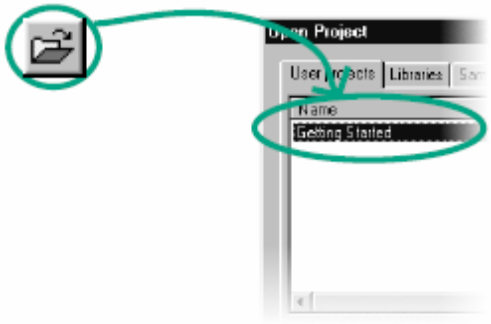
9 Paylaşılan Veri Bloğunun Programlanması

9.1 Paylaşılan Veri Bloklarının Oluşturulması ve Açılması

Bir CPU'da tüm verilerin saklanması için yeterli dahili bellek yoksa belli verileri paylaşılan bir veri bloğunda saklayabilirsiniz.

Paylaşılan bir veri bloğundaki veriler diğer her bloğun kullanımına açıktır. Örnek bir veri bloğu, diğer taraftan, belli bir veri bloğuna atanır ve verileri yerel olarak bu fonksiyon bloğunun kullanımına açıktır (bak Belim 5.5).

Sıralama Mantığında, Fonksiyon Blok Şeması veya İfade Listesinde (bak Bölüm 4 ve 5) programlamayı ve sembolik programlamayı da (bak Bölüm 3) iyi bilmeniz gereklidir.

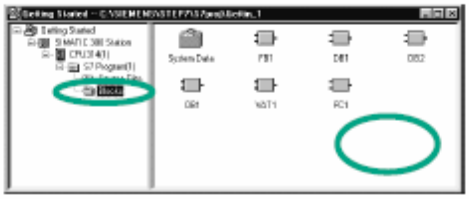


"Başlarken" projesini Bölüm 1'den 7'ye kadar iyice incelediyseniz şimdi bu pencereyi açın.

Değilse, **File > "New Project" Wizard** (*Dosya > "Yeni Proje" Sihirbazı*) menü komutunu kullanarak yeni bir proje yapın. Bunu yapmak için Bölüm 2.1'deki talimatı izleyin ve "Başlarken Fonksiyonu" adını verin.

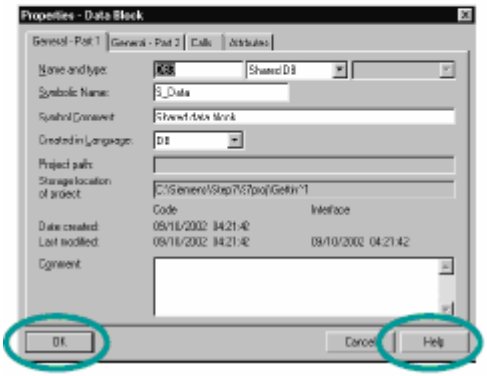
"Başlarken" projesi ile devam edeceğiz. Yine de yeni bir proje kullanarak her adımı uygulamaya devam edebilirsiniz.

Blocks klasörüne gezinin ve onu açın.



Pencerenin sağ yarısını sağ fare tuşu ile tıklayın.

Açılır menüden bir **Fonksiyon** (FC) ekleyin.



"Properties – Function (*Özellikler – Fonksiyon*)" diyalog kutusunda tüm varsayılan ayarları **OK** ile kabul edin.

Daha fazla bilgi için “Help” Tuşunu kullanın.

Veri bloğu DB3 **Blocks** klasörüne eklenmiştir.

DB3’ü açmak için çift tıklayın.

Hatırlayın, Bölüm 5.5'te, option "Data block referencing a function block.(Bir fonsiyon bloğuna başvuran veri bloğu)" seçeneğini etkinleştirerek bir örnek veri bloğu üretmiştiniz. Tersine, "Veri bloğu"nu kullanarak paylaşılan bir veri bloğu oluşturuyorsunuz.

Veri Bloğundaki Değişkenlerin Programlanması

İsim sütununa "PE_Actual_Speed" girin.

Tür seçmek için açılır menüden **Elementary Types** > **INT** menü komutunu kullanarak sağ fare tuşunu tıklayın.

Aşağıdaki örneğimizde, DB3’te üç paylaşılmış veri tanımlanır. Bu verileri değişken açıklama tablosuna buna göre girin.

Address	Name	Type	Initial val	Comment
0.0		STRUCT		
+0.0	DB_VAR	INT	0	Temporary placeholder variable
=2.0		END_STRUCT		

Veri bloğundaki gerçek hızların değişkenleri "PE_Actual_Speed" ve "DE_Actual_Speed" MW2 (PE_Actual_Speed) ve MW4 (DE_Actual_Speed) kelimeleri ile aynı işleme tabi tutulur. Gelecek bölümde bunu görebilirsiniz

Paylaşılan veri bloğunu kaydedin.

Sembollerin Atanması

Veri bloklarına sembolik isimler de verebilirsiniz.

Symbol Table (*Sembol Tablosu*)nu açın ve DN3 veri bloğu için "S_Data" sembolik adını girin.

Bölüm 4'te örnek bir projeden bir sembol tablosunu (zEn01_02_STEP7__STL_1-10, zEn01_06_STEP7__LAD_1-10 veya zEn01_04_STEP7__FBD_1-10) "Başlarken" projesine kopyaladıysanız, şimdi sembol eklemeniz gerekmez.

Symbol *(Sembol)*	Address	Data Type	Comment *(Yorum)*
...	...	...	...
S_Data	DB 3	DB 3	Shared data block *(Paylaşılan veri bloğu)*

Sembol tablosunu kaydedin ve “Sembol Düzenleyici” penceresini kapatın.

Paylaşılan veri bloğunu da kapatın.

Değişken açıklama tablosunda paylaşılan veri blokları:

View > **Data View** (*Görünüm* > *Verileri Görüntüle*) menü komutunu kullanarak paylaşılan veri bloğu tablosunda INT tipi verilerin gerçek değerlerini değiştirebilirsiniz (bak Bölüm 5.5).

Sembolik tabloda paylaşılan veri blokları:

Örnek veri bloğunun aksine, sembol tablosundaki paylaşılan veri bloğunun veri türü daima mutlak adrestir. Örneğimizde veri türü “DB3”tür. Örnek veri bloğu ile mukabil fonksiyon bloğu daima veri türü olarak belirtilir.

"Blokların Programlanması" ve "Veri Bloklarının Oluşturulması" konularındaki **Help > Contents** (*Yardım* > *İçindekiler*) altında daha fazla bilgi bulabilirsiniz.

10 Çoklu Bir Kademenin Programlanması

10.1 Yüksek Düzeyde bir Fonksiyon Bloğunun Oluşturulması ve Açılması

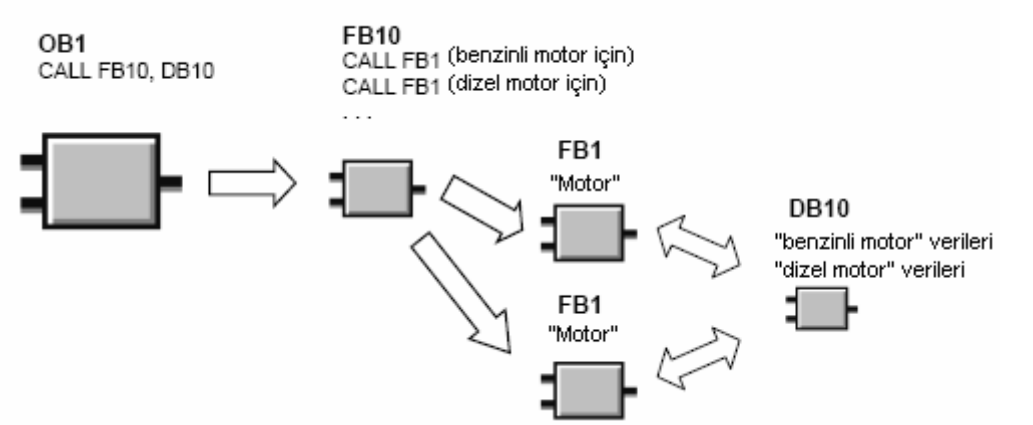
Bölüm 5'te "Motor" fonksiyon bloğu (FB1) ile bir motorun kontrolü için bir program yaptınız. Organizasyon bloğu OB1'de FB1 bloğu çağrıldığı zaman, "Petrol (*Benzin*)" (DB1) ve "Diesel (*Dizel*)" (DB2) veri bloklarını kullanmıştı. Her veri bloğunda motorların farklı verileri (örneğin, #Setpoint_Speed) vardı.

Şimdi otomasyon göreviniz için motoru kontrol etmek üzere başka programlara ihtiyacınız olduğunu hayal edin; örneğin, bir kolza yağı ya da hidrojen kullanan bir motoru v.s. kontrol etmek için bir program.

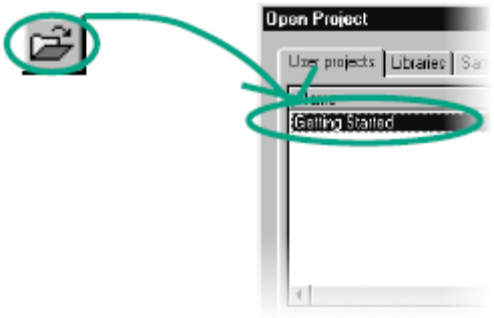
Buraya kadar öğrendiğiniz yöntemi izleyerek her ilave motorun kontrol programını kullanacak ve her defasında o motora yeni bir veri bloğu, örneğin kolza yağı kullanan motoru kontrol etmek için FB1 ile DB3, hidrojen kullanan motor için FB1 ile DB4 v.s. ayıracaksınız. Siz yeni motor kontrol programları yaptıkça blokların sayısı dikkati çekecek şekilde artacaktır.

Çoğul örneklerle çalışarak, diğer taraftan, blokların sayılarını azaltabilirsiniz. Bunu yapmak için yeni, daha yüksek düzeyde bir fonksiyon bloğu (örneğimizde FB10) oluşturursunuz ve bundaki değişmeyen FB1'i "yerel örnek" olarak adlandırırsınız. Her çağrıda alt FB1 kendi verilerini daha yüksek düzeydeki FB10'un DB10 veri bloğunda saklar. Yani, FB1'e herhangi veri bloğu atamak zorunda değilsiniz. Tüm fonksiyon blokları geriye doğru tek bir veri bloğuna (burada DB10'a) döner.

DB1 ve DB2 veri blokları DB10' entegre edilmiştir. Bunu yapmak için FN1'in FB10'un statik yerel verilerinde olduğunu açıklamalısınız.



Sıralama Mantığında, Fonksiyon Blok Şeması veya İfade Listesinde programlamayı (bak Bölüm 4 ve 5) ve sembolik programlamayı da (bak Bölüm 3) iyi biliyor olmalısınız).



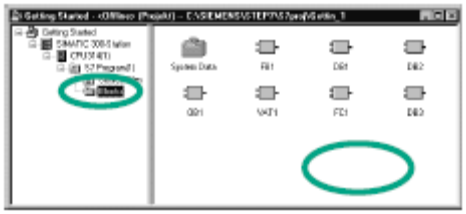
"Başlarken" projesini Bölüm 1'den 7'ye kadar iyice incelediyseniz şimdi bu pencereyi açın.

Değilse, SIMATIC Yöneticisinde aşağıdakilerden birini açın:

Sıralama Mantığı için ZEn01_05_STEP7__LAD_1-9,

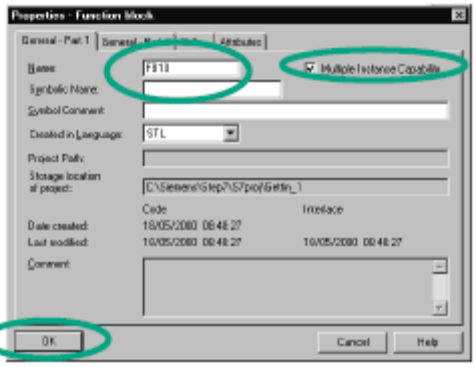
İfade Listesi için ZEn01_01_STEP7__STL_1-9

Fonksiyon Blok Şeması için ZEn01_03_STEP7__FBD_1-9.



Blocks klasörüne gezinin ve onu açın.

Pencerenin sağ yarısını sağ fare tuşu ile tıklayın ve açılır menüyü kullanarak bir fonksiyon bloğu ekleyin.



Bloğun adını FB10 olarak değiştirin ve gerekli program dilini seçin.

Multiple instance FB' yi **(**gerekli ise) etkinleştirin ve kalan varsayılan ayarları **OK** ile kabul edin.

FB10 Blocks klasörüne eklenmiştir. **FB10**'u açmak için çift tıklayın**.**

Her fonksiyon bloğu için, hatta vana kontrol programları için bile, çoğul örnekler oluşturabilirsiniz, örneğin çoğul örneklerle çalışmak istiyorsanız hem çağıran hem de çağrılan fonksiyon bloklarının çoğul örnek yeteneği olması gerektiğini unutmayın.

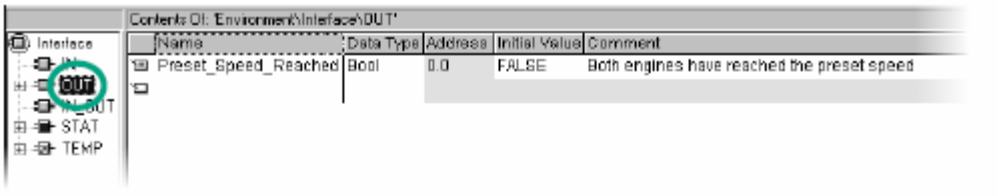
"Blokların Programlanması" ve "Blokların ve Kitaplıkların Oluşturulması" konularındaki **Help > Contents** (*Yardım* > *İçindekiler*) altında daha fazla bilgi bulabilirsiniz.

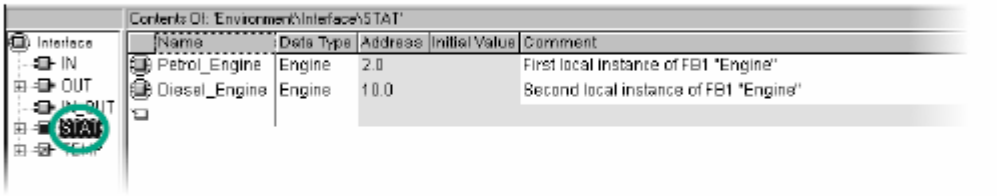
10.2 FB10'un Programlanması

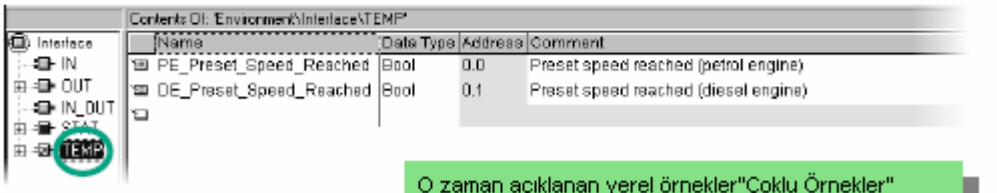
FB10'un bir "yerel örneği" olarak FB1'i çağırmak için, değişken ayrıntı görünümünde her planlı FB1 çağrısı için statik bir değişken farklı bir isimle açıklanmalıdır. Burada veri tür FB1 ("Motor")dur.

Değişkenleri Açıkla / Tanımla

LAD/STL/FBD program penceresinde FB10 açılır. Önceki imajın açıklamalarını değişken ayrıntı görünümünüze aktarın. Bunu yapmak için, "OUT", "STAT" ve "tTEMP" açıklama türlerini birbiri arkasından seçin ve girişlerinizi değişken ayrıntı görünümünde yapın. Aşağı açılan listeden "STAT" açıklama türü için veri türünü "FB<nr>" olarak seçin ve karakter dizi sayısı "<nr>FB <nr>"yi 1 olarak değiştirin.





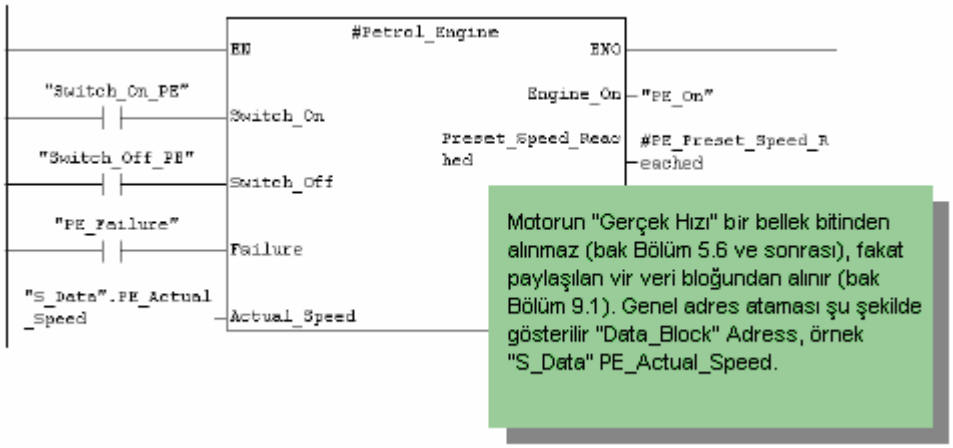


O zaman açıklanan yerel örnekler"Çoklu Örnekler" altında "Program elemanları" sekmesinde gözükecektir.

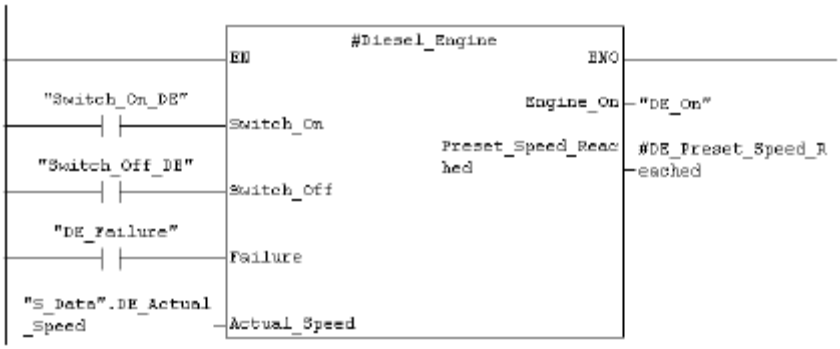
Sıralama Mantığında FB10'un Programlanması

"Petrol_Engine"i Network 1'e "Petrol_Engine" çoğul örnek bloğu olarak ekleyin.

Sonra gerekli normal olarak açık kontakları ekleyin ve çağrıyı sembolik isimlerle tamamlayın.



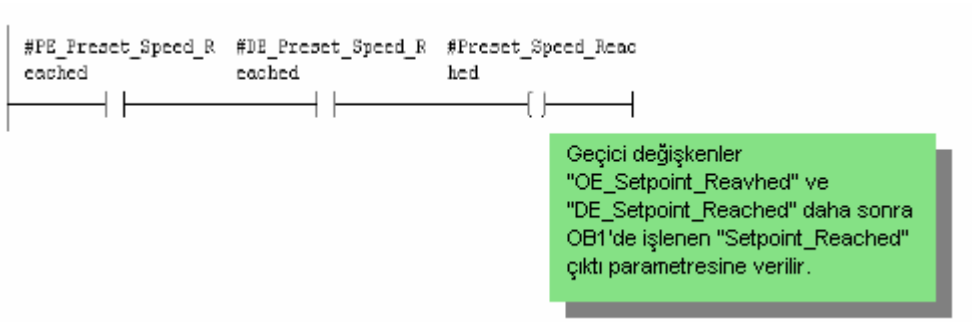
Yeni bir ağ ekleyin ve dizel motor için çağrıyı programlayın. Network 1'deki yolun aynısıyla devam edin.



Yeni bir ağ ekleyin ve mukabil adresleri olan bir seri devre programlayın. Sonra programınızı kaydedin ve bloğu kapatın.

Kendi geçici değişkenlerini kullanın. Geçici değişkenleri aşağı açılan menüde solda gösterilen simgeler yardımıyla tanıyacaksınız.

Sonra programınızı kaydedin ve bloğu kapatın.



İfade Listesinde FB10'un Programlanması

```
CALL  #Petrol_Engine
 Switch_On            :="Switch_On_PE"
 Switch_Off           :="Switch_Off_PE"
 Failure              :="PE_Failure"
 Actual_Speed         :="S_Data".PE_Actual_Speed
 Engine_On            :="PE_On"
 Preset_Speed_Reached:=#PE_Preset_Speed_Reached

CALL  #Diesel_Engine
 Switch_On            :="Switch_On_DE"
 Switch_Off           :="Switch_Off_DE"
 Failure              :="DE_Failure"
 Actual_Speed         :="S_Data".DE_Actual_Speed
 Engine_On            :="DE_On"
 Preset_Speed_Reached:=#DE_Preset_Speed_Reached

A     #PE_Preset_Speed_Reached
A     #DE_Preset_Speed_Reached
=     #Preset_Speed_Reached
```

İfade Listesinde programlama yapıyorsanız yeni bir ağ altında bir girdi bölgesi seçin ve burada gösterilen STL talimatını girin.

Sonra programınızı kaydedin ve bloğu kapatın.

FB10'un Programlanması Fonksiyon Blok Şemasında

Fonksiyon Blok Şemasında programlama yapıyorsanız, yeni bir ağ altında bir girdi bölgesi seçin ve burada gösterilen FBD talimatını girin.

Sonra programınızı kaydedin ve bloğu kapatın

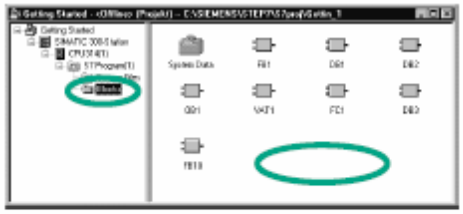
FB1'in iki çağrısını FB10'da düzenlemek için FB10'un kendisi çağrılmalıdır.

Çoğul örnekler sadece fonksiyon blokları için programlanabilir.Fonksiyonlar (FCs) için çoğul örnek oluşturulması mümkün değildir.

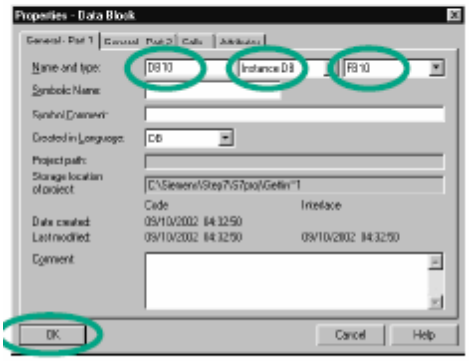
"Blokların Programlanması", "Mantıksal Bloklar Oluşturulması" ve "Değişken Açıklamasında Çoğul Örnekler" konularındaki **Help > Contents** (*Yardım* > *İçindekiler*) altında daha fazla bilgi bulabilirsiniz.

10.3 DB10 Üretilmesi ve Gerçek Değerin Uyarlanması

Yeni veri bloğu DB10, DB1 ve DB2 veri bloklarının yerini alacaktır. Benzinli motor ve dizel motor verileri DB10'da saklanacak ve daha sonra OB1'de FB10'un çağrılmasında gerekli olacaktır (bak "OB1'de FB1'in çağrılması" Bölüm 5.6 ve sonrası).



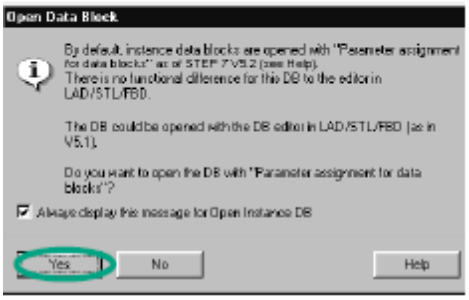
SIMATIC Yöneticisindeki "Başlarken" projesinde açılır menüyü kullanarak **Blocks** klasöründe DB10 veri bloğunu oluşturun.



Bunu yapmak için, "Özellikler – Veri Bloğu" diyalog kutusundaki veri bloğunun adını DB10 olarak değiştirin, sonra bitişiğindeki aşağı acılan listede "Instance DB (*Örnek DB*)" uygulamasını seçin. Sağ aşağı açılan listede "FB10" fonksiyon bloğunu atamak üzere seçin ve kalan ayarları **OK** ile doğrulayın.

DB10 veri bloğu "Başlarken" projesine eklenmiştir.

DB10 üzerine çift tıklayın.



Sonraki diyalog kutusunda, Örnek DB'yi açmak için **Yes** (*Evet*) cevabı verin. **View** > **Data View** (*Görünüm* > *Veri Görüntüle*) menü komutunu seçin.

Veri görünümü DB10'daki, FB1 ("yerel örnekler") için iki çağrının "iç" değişkenleri dahil her değişkeni gösterir.
Beyan görünümü değişkenleri FB10'da açıklanan şekilde gösteri

Dizel motorun gerçek değerini “1300” olarak değiştirin, bloğu kaydedin ve sonra kapatın.

	Address	Declaration	Name	Type	Initial value	Actual value	Comment
1	0.0	out	Preset_Speed_Reached	BOOL	FALSE	FALSE	Both engines have reached the preset speed
2	2.0	stat:in	Petrol_Engine.Switch_On	BOOL	FALSE	FALSE	Switch on engine
3	2.1	stat:in	Petrol_Engine.Switch_Off	BOOL	FALSE	FALSE	Switch off engine
4	2.2	stat:in	Petrol_Engine.Failure	BOOL	FALSE	FALSE	Engine failure, causes the engine to switch off
5	4.0	stat:in	Petrol_Engine.Actual_Speed	INT	0	0	Actual engine speed
6	6.0	stat:out	Petrol_Engine.Engine_On	BOOL	FALSE	FALSE	Engine is switched on
7	6.1	stat:out	Petrol_Engine.Preset_Speed_Reached	BOOL	FALSE	FALSE	Preset speed reached
8	8.0	stat	Petrol_Engine.Preset_Speed	INT	1500	1500	Requested engine speed
9	10.0	stat:in	Diesel_Engine.Switch_On	BOOL	FALSE	FALSE	Switch on engine
10	10.1	stat:in	Diesel_Engine.Switch_Off	BOOL	FALSE	FALSE	Switch off engine
11	10.2	stat:in	Diesel_Engine.Failure	BOOL	FALSE	FALSE	Engine failure, causes the engine to switch off
12	12.0	stat:in	Diesel_Engine.Actual_Speed	INT	0	0	Actual engine speed
13	14.0	stat:out	Diesel_Engine.Engine_On	BOOL	FALSE	FALSE	Engine is switched on
14	14.1	stat:out	Diesel_Engine.Preset_Speed_Reached	BOOL	FALSE	FALSE	Preset speed reached
15	16.0	stat	Diesel_Engine.Preset_Speed	INT	1500	1300	Requested engine speed

Tüm değişkenler artık DB10’un değişken açıklama tablosunda kayıtlıdır. İlk yarıda “Petrol Engine (*Benzinli Motor*)” fonksiyon bloğunun çağrısı için değişkenleri ve ikinci yarıda “Diesel Engine (*Dizel Motor*) için değişkenleri görebilirsiniz (bak Bölüm 5.5).

FB1’in “iç” değişkenleri kendi isimlerini korurlar; örneğin “, "Switch_On". Yerel örneğin adı şimdi bu adların önünde yer alır; örneğin, "Petrol_Engine.Switch_On."

"Blokların Programlanması" ve " Veri Blokları Oluşturulması" konularındaki **Help > Contents** (*Yardım* > *İçindekiler*) altında daha fazla bilgi bulabilirsiniz.

10.4 OB1'de FB10'un Çağrılması

Örneğimizde OB1'de FB10 için çağrı yapılır. Bu çağrı OB1'de FB 1'i çağrıyı programlarken öğrendiğiniz fonksiyonun aynısını yapar (bak Bölüm 5.6 ve sonrası.). Çoğul örnekleri kullanarak, Bölüm 5.6 ve sonrasında programladığınız Network 4 ve 5'i değiştirebilirsiniz.

FN10'u henüz programladığınız projedeki **OB1**'i açın.

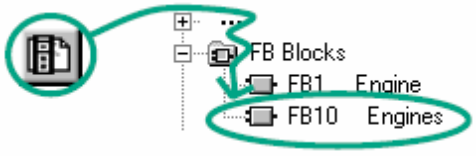
Sembolik İsimlerin Tanımlanması

LAD/STL/FBD program penceresi açılır. **Options > Symbol Table** (*Seçenekler > sembol Tablosu*) menü komutunu kullanarak sembol tablosunu açın ve sembol tablosundaki FB10 fonksiyon blokları ile DB10 veri bloğu için isimleri girin.

Sonra sembol tabloyu kaydedin ve pencereyi kapatın.

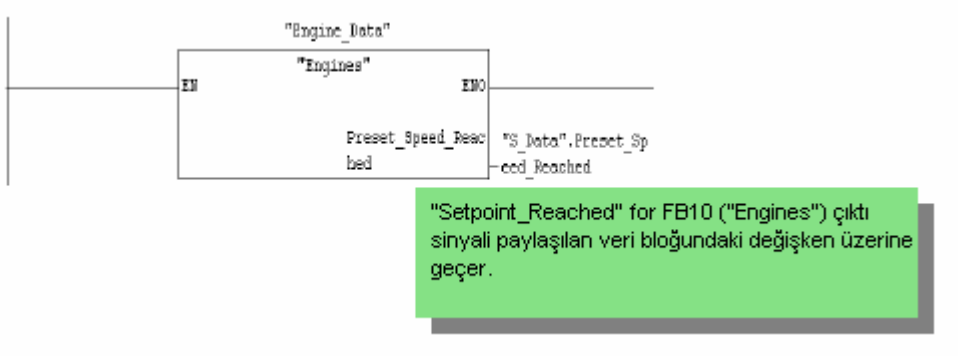
Symbol	Address	Data Type	Comment
...	...	...	...
Engines	FB 10	FB 10	Example of multiple instances
Engine_Data	DB 10	FB 10	Instance data block for FB10 10
...	...	...	...

Sıralama Mantığında Çağrı Programlaması



OB1'in sonuna yeni bir ağ ekleyin ve **FB10** ("Engines") çağrısını girin.

Aşağıdaki çağrıyı mukabil sembolik isimlerle tamamlayın.



İfade Listesinde Çağrı Programlaması

Yeni bir ağ altında bir girdi bölgesi seçin ve burada gösterilen STL talimatını girin. Bunu yapmak için, Program Elemanları kataloğunad **FB Blocks > FB10 Engines**'i kullanın.

Şimdi biz FB1'i FB10 üzerinden merkezi olarak çağıracağımızdan FB1 için OB1'deki çağrıyı silin (Network 4 ve 5, Bölüm 5.6 ve sonrası).

Sonra programınızı kaydedin ve bloğu kapatın.

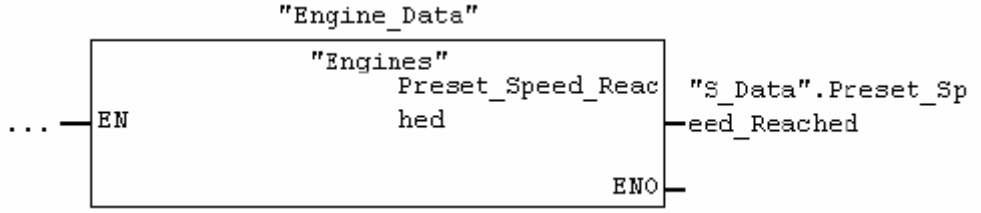
```
CALL  "Engines" , "Engine_Data"
   Preset_Speed_Reached:="S_Data".Preset_Speed_Reached
```

Fonksiyon Blok Şemasında Çağrı Programlanması

Fonksiyon Blok Şemasında Programlama yapıyorsanız, yeni ağ altında girdi bölgesini seçin ve aşağıdaki FBD talimatlarını girin. Bunu yapmak için, **FB Blocks > FB10 Engines'** i Program Elemanları Katalogundan kullanın.

Şimdi biz FB1'i FB10 üzerinden merkezi olarak çağıracağımızdan FB1 için OB1'deki çağrıyı silin (Network 4 ve 5, Bölüm 5.6 ve sonrası).

Sonra programınızı kaydedin ve bloğu kapatın.



Otomasyon görevleriniz için ek motor kontrol programları, örneğin gaz motorları, hidrojen motorları v.s., yapmanız gerekirse bunları çoğul örnekler olarak programlayabilir ve bunları aynı yolla ve FB10'dan çağırabilirsiniz.

Bunu yapmak için, ek motorları FB10 ("Engines (*Motorlar*)"ın değişken açıklama tablosunda gösterildiği gibi açıklayın ve FB1 için çağrıyı FB10 (program Elemanları kataloğunda çoğul örnek)'da programlayın. Sonra yeni sembolik isimleri, örneğin, sembol tablosundaki anahtar açma anahtar kapama yöntemleri için tanımlayabilirsiniz.

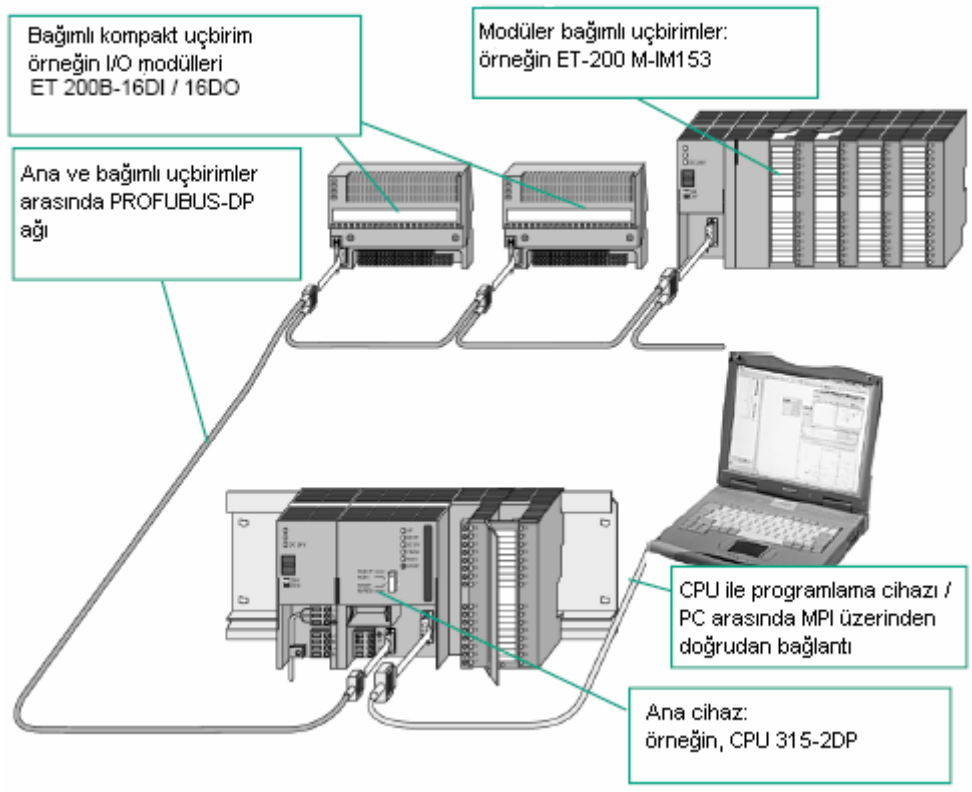
Help > Contents (*Yardım* > *İçindekiler*) altında ve sonra "The STL, FBD, veya LAD Dil Açıklaması" konularındaki "Calling References Helps (Referans Yardımların Çağrılması)" altında daha fazla bilgi bulabilirsiniz.

11 Dağıtılmış I/O'ların Yapılandırılması

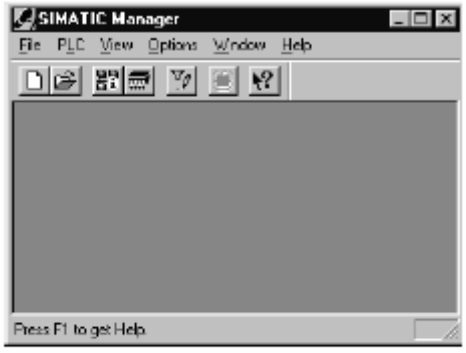
11.1 Dağıtılmış I/O'ların PROFIBUS DP ile Yapılandırılması

Konvansiyonel yapılandırmalı otomasyon sistemleri, merkezi programlanabilir kontrol edicilerin I'O modüllerine doğrudan takılan alıcılar ve erişim düzenekleri ile kablo bağlantıları vardır. Bu genellikle önemli miktarda kablo kullanılması gerekir demektir.

Dağıtılmış bir yapılandırma kullanarak, alıcılar ve erişim düzeneklerinin yakınına girdi ve çıktı modülleri koyup gerekli kablo miktarını önemli derecede azaltabilirsiniz. Programlanabilir mantıksal kontrol ediciler, I'O modülleri ve PROFIBUS DP'yi kullanan alan cihazı arasında bağlantı kurabilirsiniz. Bölüm 6'da konvansiyonel bir yapılandırmayı nasıl programlayacağınızı bulabilirsiniz. Merkezi bir yapılandırma veya dağıtılmış bir yapılandırma oluşturmanızın farkı yoktur.. Kullanılacak modülleri donanım katalogundan seçersiniz, onları rafta düzenlersiniz ve özelliklerini ihtiyaçlarınıza göre uyarlarsınız.

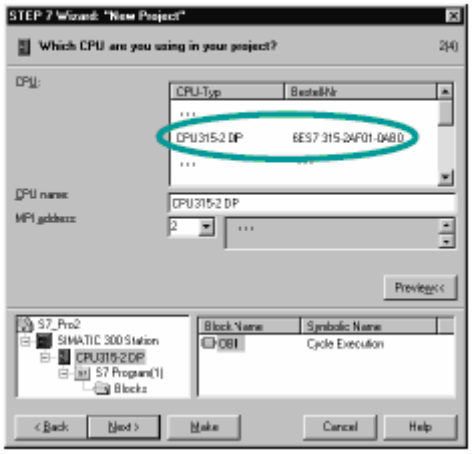


Yeni bir Proje Yapma



Başlangıç noktası SIMATIC Yöneticisidir. İşlerin izlenmesini kolaylaştırmak için açık projeleri kapatın.

Yeni bir proje yapın.

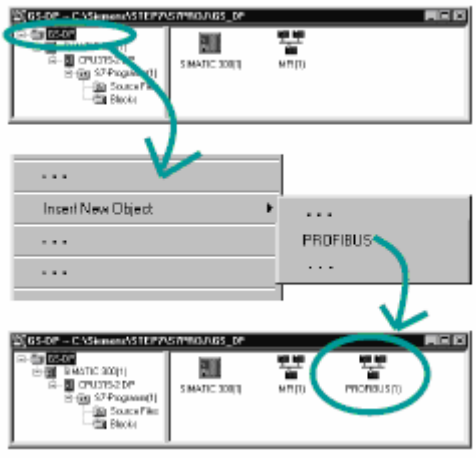


Mukabil diyalog kutusunda **CPU 315-2DP'**yi seçin (PROFIBUS-DP'li ağ).

Şimdi Bölüm 2.1 ile aynı yolda ilerleyin ve projeye "GS-DP" (Başlarken – Dağıtılmış I/O) adını verin.

Bu noktada kendi yapılandırmanızı oluşturmak isterseniz, şimdi CPU'nuzu belirtin. CPU'nuzun dağıtılmış I/O'ları desteklemesi gerektiğini unutmayın.

Inserting the PROFIBUS Network

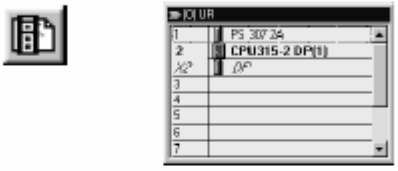


GS-DP klasörünü seçin.

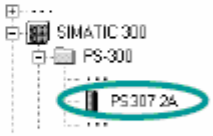
Sağ fare tuşunu kullanarak pencerenin sağ yarısına **PROFIBUS** ağını koyun.

İstasyonun Yapılandırılması

SIMATIC 300 Station klasörünü seçin ve Donanıma çift tıklayın. "HW Config" penceresi açılır (bak Bölüm 6.1).



CPU 315-2 DP zaten rafta gözükür. Gerekli ise **View** > **Hardware Catalog** menü komutunu veya araç çubuğundaki mukabil tuşu kullanarak Donanım katalogunu açın.

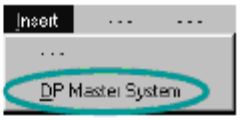


Güç besleme modülü **PS307 2A**'yı sürükleyip bırakarak slot 1'e koyun.

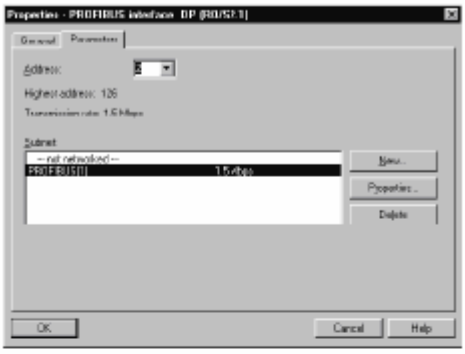
Aynı yolla **DI32xDC24V** and **DO32xDC24V/0.5A** I/O modüllerini slot 4 ve 5'e koyun.

Aynı rafa dağıtılan I/O'ları destekleyen CPU'ya ek olarak başka CPU'lar da yerleştirebilirsiniz.

DP-Master Sistemin Yapılandırılması



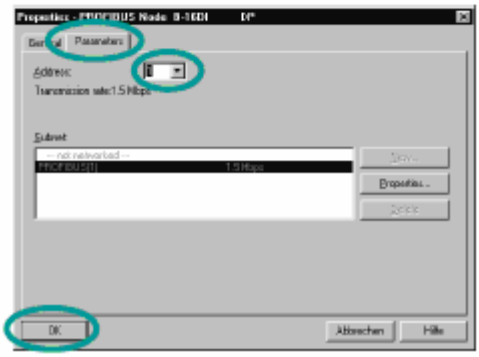
Slot 2.1'deki DP master'ı seçin ve bir **DP-master system** koyun.



Görünen diyalog kutusunda önerilen adresleri uygulayın. "Subnet (*Alt ağ*) alanında "PROFIBUS(1)" seçin ve sonra ayarlarınızı OK ile uygulayın.

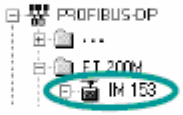
Ana sisteme koyduğunuz şeyleri artık sol fare tuşunu basılı tutup çekerek hareket ettirebilirsiniz.

B-16DI modülüne ulaşıncaya kadar Donanım katalogunda gezinin ve bu modülü master sisteme koyun (imleç "+" işaretine dönünceye kadar cismi sürükleyin, sonra bırakın).



Koyduğunuz modülün düğüm adresini "Özellikler" diyalog kutusunun "Parametreler" sekmesinde değiştirebilirsiniz.
Önerilen adresi OK ile doğrulayın.

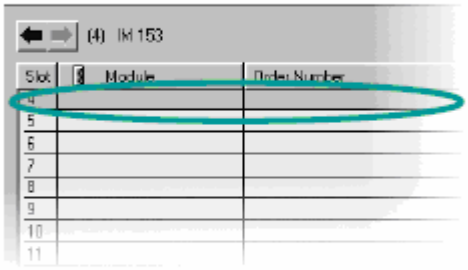
Aynı şekilde **B-16DO** modülünü master sistemin üzerine sürükleyip bırakın.



IM153 arabirim modülünü master sistem üzerine sürükleyip bırakın ve düğüm adresini **OK** ile doğrulayın.

Örneğimizde varsayılan mod adresleri kullanıyoruz. Ancak ihtiyaçlarınızı karşılamak üzere , adresleri her zaman değiştirebilirsiniz.

Ağda **ET200M**'yi seçin. ET200M için boş slotlar alt yapılandırma tablosunda gösterilir.

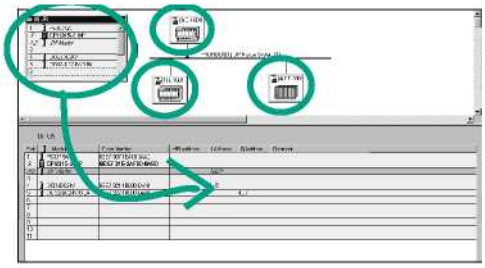


Burada slot 4'ü seçin.

ET200M'nin kendisinin ek I'O modülleri olabilir. Örneğin, slot 4 için **DI32xDC24V** modülünü seçin ve onu koymak için bu modüle çift tıklayın.

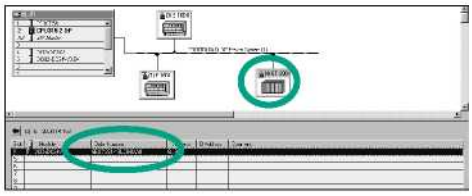
Donatım kataloğunu kullanırken doğru klasörde olduğunuzdan emin olmalısınız. Örneğin ET200M'in modüllerini seçmek için ET200M klasöründe olmalısınız.

Düğüm Adresinin Değiştirilmesi



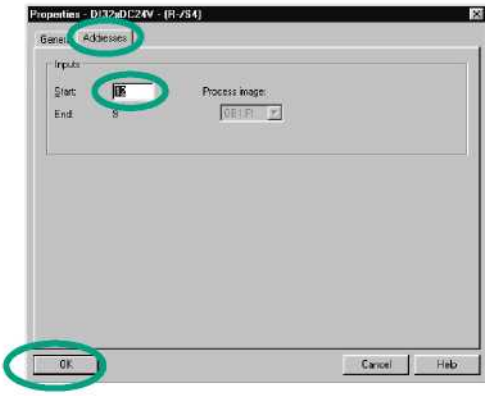
Örneğimizde düğüm adresini değiştirmemiz gerekmez. Pratikte ekseriye gerekir.

Diğer düğümleri birbiri arkasına seçin ve girdi ve çıktı adreslerini kontrol edin. "Donanımın Yapılandırılması" uygulaması tüm adresleri uyarlamıştır, bu nedenle çift atama olmaz.



ET100M'nin adresini değiştirmek istediğinizi düşünelim:

ET200M'i seçin ve **DI32xDC24V** (slot 4) üzerine çift tıklayın.

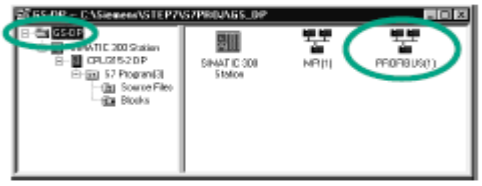


Şimdi "Özellikler" diyalog kutusunun "Adresler" sekmesindeki girdi adreslerini 6'dan 1w2'ye değiştirin. Diyalog kutusunu **OK** ile kapatın

Son olarak dağıtılmış I'O yapılandırmasını **kaydedin ve derleyin**.

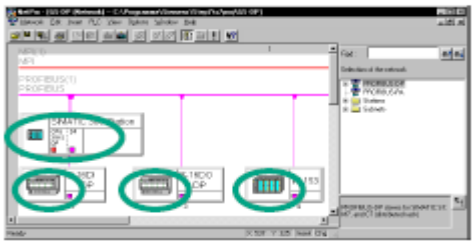
Save and Compile (Kaydet ve Derle) menü komutu yapılandırmanın tutarlılığının otomatik olarak kontrol edildiği anlamına gelir. Hata yoksa sistem verileri üretilir ve programlanabilir kontrol ediciye indirilebilir.
Save (Kaydet) ile hatalar olsa bile yapılandırmayı kaydedebilirsiniz. Ancak bu durumda onu programlanabilir kontrol ediciye indiremezsiniz.

Seçenek: Ağların Yapılandırılması



Dağıtılmış I'O'ları seçimlik "Ağların Yapılandırılması" paketini kullanarak da yapılandırabilirsiniz.

SIMATIC Yöneticisinde **PROFIBUS (1)** ağına çift tıklayın.



"NetPro" penceresi açılır.

EK DP bağımlı uçbirimleri ağ cisimleri katalogundan PROFIBUS DP üzerine sürükleyip bırakabilirsiniz.

Herhangi elemanı yapılandırmak için çift tıklayın. "Donanımın Yapılandırılması" penceresi açılır.

Station > Consistency Check ("Donanımın Yapılandırılması" penceresi) ve **Network > Consistency Check** ("Ağın Yapılandırılması " penceresi) menü komutlarını kullanarak, kaydetmeden önce yapılandırma hatalarını kontrol edebilirsiniz. Hatalar gösterilir ve STEP 7 olası çözümleri önerecektir.

"Donanımın yapılandırılması" and "Dağıtılmış I/O'ların Yapılandırılması" konularındaki **Help > Contents** (*Yardım* > *İçindekiler*) altında daha fazla bilgi bulabilirsiniz

Tebrikler! Başlarken Elkitabını incelediniz ve en önemli şartları, yöntemleri ve STEP 7 fonksiyonlarını öğrendiniz. Artık kendi ilk projenizi başlatabilirsiniz.

Eğer gelecekteki projelerde çalışırken belirli fonksiyonlar ararsanız veya STEP 7'deki işletim talimatlarından herhangi birini unutmuş olursanız STEP 7'deki kapsamlı yardımımızı kullanabilirsiniz.

STEP 7 bilginizi genişletmek isterseniz, birçok uzmanlaşmış eğitim kursları vardır. Yerel Siemens temsilciniz size yardım etmekten mutlu olacaktır.

Projelerinizde başarılar dileriz!

Siemens AG

Ek A

Başlarken Elkitabı için Örnek Projelerin İncelenmesi

- **ZEn01_02_STEP7__STL_1-10:**
  STL programlama dilinde sembol tablosu dahil programlanmış bölümler 1 ila 10.
- **ZEn01_01_STEP7__STL_1-9:**
  STL programlama dilinde sembol tablosu dahil programlanmış bölümler 1 ila 9.
- **ZEn01_06_STEP7__LAD_1-10:**
  LAD programlama dilinde sembol tablosu dahil programlanmış bölümler 1 ila 10.
- **ZEn01_05_STEP7__LAD_1-9:**
  LAD programlama dilinde sembol tablosu dahil programlanmış bölümler 1 ila 9.
- **ZEn01_04_STEP7__FBD_1-10:**
  FBD programlama dilinde sembol tablosu dahil programlanmış bölümler 1 ila 10.
- **ZEn01_03_STEP7__FBD_1-9:**
  FBD programlama dilinde sembol tablosu dahil programlanmış bölümler 1 ila 9.
- **ZEn01_07_STEP7__Dist_IO:**
  Dağıtılmış I/O'lu programlanmış bölüm 11.

Dizin

M

O

P

S

V

Y

Z