**İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ**

# DEMİRYOLU SİNYALİZASYON SİSTEMİ SİMÜLATÖRÜ VE SCADA SİSTEMİ

**YÜKSEK LİSANS TEZİ**

**Fatih MECİTOĞLU**

**Kontrol ve Otomasyon Mühendisliği Anabilim Dalı**

**Kontrol ve Otomasyon Mühendisliği Programı**

**Tez Danışmanı: Doç. Dr. Mehmet Turan SÖYLEMEZ**

**OCAK 2013**

**İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ**

# DEMİRYOLU SİNYALİZASYON SİSTEMİ SİMULATÖRÜ VE SCADA SİSTEMİ

**YÜKSEK LİSANS TEZİ**

**Fatih MECİTOĞLU**
**504091107**

**Kontrol ve Otomasyon Mühendisliği Anabilim Dalı**

**Kontrol ve Otomasyon Mühendisliği Programı**

**Tez Danışmanı: Doç. Dr. Mehmet Turan SÖYLEMEZ**

**OCAK 2013**

İTÜ, Fen Bilimleri Enstitüsü'nün 504091107 numaralı Yüksek Lisans Öğrencisi Fatih MECİTOĞLU, ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı "Demiryolu Sinyalizasyon Sistemi Simülatörü ve SCADA Sistemi" başlıklı tezini aşağıda imzaları olan jüri önünde başarı ile sunmuştur.

| | | |
|---|---|---|
| **Tez Danışmanı :** | **Doç. Dr. Mehmet Turan SÖYLEMEZ**<br>İstanbul Teknik Üniversitesi | ..... |
| **Jüri Üyeleri :** | **Doç. Dr. Salman KURTULAN**<br>İstanbul Teknik Üniversitesi | ..... |
| | **Yrd. Doç. Dr. Turgay ALTILAR**<br>İstanbul Teknik Üniversitesi | ..... |

**Teslim Tarihi : 17 Aralık 2012**
**Savunma Tarihi : 24 Ocak 2013**

## ÖNSÖZ

Bu çalışmada, yardım ve desteği bulunan, bilgi ve tecrübesiyle yol gösteren değerli danışman hocam Doç. Dr. Mehmet Turan SÖYLEMEZ'e teşekkürü bir borç bilirim.

Aralık 2012

Fatih MECİTOĞLU
Elektrik Mühendisi

**İÇİNDEKİLER**

**Sayfa**

**ÖNSÖZ** ..... **v**
**İÇİNDEKİLER** ..... **vii**
**KISALTMALAR** ..... **ix**
**ÇİZELGE LİSTESİ** ..... **xi**
**ŞEKİL LİSTESİ** ..... **xiii**
**ÖZET** ..... **xv**
**SUMMARY** ..... **xvii**
**1. GİRİŞ** ..... **1**
1.1 Tezin Amacı ..... 1
1.2 Literatür Araştırması ..... 2
**2. MERKEZİ KONTROLLÜ DEMİRYOLU SİNYALİZASYON SİSTEMİ** ..... **5**
2.1 Trafik Kontrol Merkezi ..... 5
2.2 Saha Donanımları ..... 6
2.2.1 Ray devreleri ..... 6
2.2.2 Ray blokları ..... 6
2.2.3 Makaslar ..... 6
2.2.4 Sinyaller ..... 6
2.3 Anklaşman Sistemi ..... 7
**3. DEMİRYOLU SİNYALİZASYON SİSTEMİ TASARIMI** ..... **9**
3.1 İşlevler ..... 9
3.1.1 Rota tanziminin yapılması ve sonlandırılması ..... 9
3.1.2 Rota tanziminin iptal edilmesi ..... 10
3.1.3 Makas konumunun değiştirilmesi ..... 11
3.1.4 Sinyaller ..... 11
3.2 UML Diyagramları ..... 12
3.2.1 UML bileşen diyagramı ..... 12
3.2.2 UML sınıf diyagramları ..... 13
3.2.3 UML durum diyagramı ..... 18
3.2.4 UML sıra diyagramı ..... 18
**4. YAZILIMLARIN GELİŞTİRİLMESİ** ..... **21**
4.1 SCADA Yazılımı ..... 21
4.2 PLC Simülatörü ..... 24
**5. YAZILIMLARIN TEST EDİLMESİ** ..... **31**
**6. SONUÇ VE ÖNERİLER** ..... **45**
**KAYNAKLAR** ..... **47**
**EKLER** ..... **49**
**ÖZGEÇMİŞ** ..... **52**

**KISALTMALAR**

| | |
|---|---|
| **DI** | **:** Digital Input |
| **DO** | **:** Digital Output |
| **I/O** | **:** Input/Output |
| **PLC** | **:** Programmable Logic Controller |
| **SCADA** | **:** Supervisory Control and Data Acquisition |
| **TCP** | **:** Transfer Control Protocol |
| **UML** | **:** Unified Modelling Language |

## ÇİZELGE LİSTESİ

**Sayfa**

**Çizelge 3.1 :** Sinyal lambaları ve renkler. ..... 11
**Çizelge 4.1 :** Anklaşman tablosu. ..... 25
**Çizelge 4.2 :** Rota çakışma tablosu. ..... 25
**Çizelge 4.3 :** Ray bloklarına ait sensörler için PLC dijital girişleri. ..... 26
**Çizelge 4.4 :** Makaslar için PLC dijital girişleri. ..... 26
**Çizelge 4.5 :** Sinyaller için PLC dijital çıkışları. ..... 28
**Çizelge 4.6 :** Rota tanzim isteği ve cevaplarına ait PLC bellek alanları ..... 29
**Çizelge A.1 :** Modbus TCP veri paketi yapısı. ..... 50
**Çizelge A.2 :** Modbus TCP'de sıklıkla kullanılan fonksiyon kodları. ..... 51

## ŞEKİL LİSTESİ

**Sayfa**

**Şekil 2.1 :** Tren kontrol merkezi. ..... 5
**Şekil 3.1 :** UML bileşen diyagramı. ..... 13
**Şekil 3.2 :** SCADA yazılımı UML sınıf diyagramı. ..... 14
**Şekil 3.3 :** PLC Simülatörü UML sınıf diyagramı. ..... 16
**Şekil 3.4 :** Rota tanzimini denetleyen durum makinesi için durum diyagramı. ..... 18
**Şekil 3.5 :** UML sıra diyagramı. ..... 19
**Şekil 4.1 :** Qt derleyicisi arayüzü. ..... 22
**Şekil 4.2 :** SCADA yazılımı arayüzü. ..... 22
**Şekil 4.3 :** Modbus TCP/IP PLC Simulatör yazılımı. ..... 23
**Şekil 4.4 :** Rota tanzimi sonlu durum diyagramı arayüzü. ..... 23
**Şekil 4.5 :** Demiryolu bölgesi. ..... 24
**Şekil 4.6 :** PLC Simülatör yazılımı dijital giriş ve çıkışlar. ..... 27
**Şekil 4.7 :** PLC Simülatörü saha donanımları dijital giriş ve çıkışlar ..... 27
**Şekil 5.1 :** Simülatörde makas dijital çıkışları. ..... 31
**Şekil 5.2 :** SCADA durum diyagramı (durum 2). ..... 32
**Şekil 5.3 :** SCADA yazılımında makasların konumlandırılması. ..... 32
**Şekil 5.4 :** Simülatörde makas konumlarının değiştirilmesi. ..... 33
**Şekil 5.5 :** SCADA durum diyagramı (durum 3). ..... 33
**Şekil 5.6 :** SCADA onayının beklenmesi. ..... 34
**Şekil 5.7 :** SCADA durum diyagramı (durum 4). ..... 34
**Şekil 5.8 :** Simülatörde makasların kilitlenmesi. ..... 35
**Şekil 5.9 :** SCADA arayüzünde makasların kilitlenmesi. ..... 35
**Şekil 5.10 :** Simülatörde makasların kilitli bilgisinin değiştirilmesi. ..... 36
**Şekil 5.11 :** SCADA durum diyagramı (durum 5). ..... 36
**Şekil 5.12 :** SCADA arayüzünde tanzimin yapılması. ..... 37
**Şekil 5.13 :** Simülatörde 1BT ray bloğuna tren yerleştirilmesi. ..... 38
**Şekil 5.14 :** 1BT ray bloğunda tren varken SCADA arayüzü. ..... 38
**Şekil 5.15 :** Simülatörde 1BT ve 3T ray bloğuna tren yerleştirilmesi. ..... 39
**Şekil 5.16 :** Tren 1BT ve 3T ray bloğundayken SCADA arayüzü. ..... 39
**Şekil 5.17 :** Tren 3T ray bloğundayken SCADA arayüzü. ..... 40
**Şekil 5.18 :** Simülatörde trenin 3ST ray bloğuna yerleştirilmesi. ..... 40
**Şekil 5.19 :** Tren 3ST ray bloğundayken SCADA arayüzü. ..... 41
**Şekil 5.20 :** SCADA durum diyagramı (durum 6). ..... 41
**Şekil 5.21 :** Simülatörde makas kilitlerinin çözülmesi. ..... 42
**Şekil 5.22 :** SCADA durum diyagramı (durum 0). ..... 42
**Şekil 5.23 :** SCADA arayüzü (durum 0). ..... 43
**Şekil 5.24 :** 1BT-3ST rotasıyla çakışan rotalar. ..... 43
**Şekil A.1 :** Modbus örnek ağ mimarisi ..... 50

**DEMİRYOLU SİNYALİZASYON SİSTEMİ SİMULATÖRÜ VE SCADA SİSTEMİ**

## ÖZET

Demiryolu ulaşımının giderek hızlanması ve sürekli olarak büyümesi, demiryolu sinyalizasyon ve anklaşman sistemlerinin merkezden izlenmesini ve kontrolünü gerekli hale getirmiştir. Demiryolu sinyalizasyon sistemlerinin güvenli olarak çalışması için, demiryolu sinyalizasyon sistemi donanım ve yazılımının güvenilir yapıda olması önem arz etmektedir. Bu nedenle, yazılım tasarımında biçimsel yöntemler kullanılmasının önemi büyüktür. Demiryolu sinyalizasyon sisteminde oluşabilecek en ufak hata, büyük can, mal ve zaman kayıplarına yol açabilmektedir. Demiryolu sinyalizasyon sistemi, SCADA operatörü ve makinistin yapabileceği hataları da önleyebilmelidir. Aynı zamanda, donanım ve yazılımlarda oluşabilecek hatalarda da güvenliği sağlayabilmelidir. Bu yüzden demiryolu sinyalizasyon sistemlerinin simulasyonu büyük öneme sahiptir.

Bu çalışmada, demiryolu sinyalizasyon sistemini bilgisayar ortamında simüle eden yazılımların tasarımı ve geliştirilmesi amaçlanmıştır. Geliştirilen yazılımlar, SCADA yazılımı ve PLC Simülatörüdür. Öncelikle bu yazılımların işlevleri belirlenmiştir. SCADA ve simülatör arasındaki haberleşme protokolü, Modbus TCP olarak belirlenmiştir. Daha sonra, UML diyagramları ile tasarıma geçilmiştir. Yazılımlar Nesne Yönelimli Programlama yöntemi ile tasarlanmıştır. UML bileşen diyagramı, sınıf diyagramları, durum diyagramları ve sıra diyagramı çizilmiştir. Anklaşman sistemi ve saha donanımları yazılımsal olarak modellenmiştir. Demiryolu sinyalizasyon sistemlerinin simülasyonu, izleme ve kontrolü için bir kütüphane oluşturulmuştur. Elde edilen modeller, anklaşman tablosu ve rota çakışma tablosu yardımıyla bir demiryolu bölgesi simülatörde oluşturulmuştur. Rota tanzimlerini yöneten rota nesnesi oluşturulmuştur. Bu rota nesnesi, durum makinesi olarak çalışmaktadır. Sadece giriş/çıkış ve bellek adresi girişleri değiştirilerek tüm rotalar için kullanılabilmektedir. Tasarım aşamasından sonra programların yazılmasına başlanmıştır. Programlar Qt derleyicisinde C++ programlama dili kullanılarak geliştirilmiştir. Geliştirilen programlar test edilmiştir. Sonuç olarak, gerçek anklaşman sistemine benzer şekilde çalışan bir simülasyon ve test ortamı elde edilmiştir. Geliştirilen sistem, PLC veya başka harici bir donanıma ihtiyaç duymamaktadır. Programlar aynı bilgisayarda çalıştırılabildiği gibi aynı ağda bulunan farklı bilgisayarlarda da çalıştırılabilmektedir. Kullanıcı istediği giriş değerlerini değiştirerek simülatörün ürettiği çıkışları ve bellek alanlarının değerlerini izleyebilmektedir. Geliştirilen SCADA yazılımı, Modbus TCP protokolü ile PLC Simülatörü dışında gerçek PLC ile de haberleşebilecek yapıdadır.

Geliştirilen Simülatör ve SCADA yazılımları farklı senaryolar göz önüne alınarak test edilmiştir. Yapılan testler sonucu, yazılımların belirlenen işlevleri yerine getirdiği görülmüştür. Geliştirilen yazılımlar eğitim ve test amacıyla kullanılabilecektir.

**RAILWAY SIGNALIZATION SYSTEM SIMULATOR AND SCADA SYSTEM**

## SUMMARY

Railway transportation has been growing rapidly for a century. High-speed trains are reducing travel time. Safe and high-speed travel is an advantage of railway transportation. For a safe and efficient signaling and interlocking, centrally controlled railway interlocking and signaling systems have become necessary. For the fail-safe operation, railway-signaling system needs to have a reliable structure. Therefore, the usage of formal software design methods is important because a minor failure may lead to loss of life, property and time. Railway signaling system is able to prevent the mistakes of SCADA operator and train driver. At the same time, it is able to run safely in the case of hardware or software failure.

In this study, software design and development of a railway signaling system simulator and SCADA system was aimed. SCADA software and a railway PLC simulator have been developed. Software functions were defined to achieve this. Modbus TCP was used as communication protocol between the Simulator and SCADA. Then, designing with UML diagrams was followed. In this step, UML component diagrams, class diagrams, state diagrams and sequence diagram were drawn. Software model of interlocking systems and field equipment were created. Object-oriented programming method was used here. Simulation environment of an example railway yard was created with the help of these software models. The route object was obtained by using these models. This object is responsible for managing route reservations and works as a state machine. All routes for a railway yard can be defined by using this route object by only giving its members such as signals, switches, rail blocks and their input/output and memory addresses. Programming was the next stage after the design stage. The program was developed using Qt compiler and C++ programming language. A simulation and testing environment were developed that work similar to a real interlocking system. PLC or other external equipment was not required for the developed system. Simulator and SCADA software can be run on the same computer or multiple computers. The user can monitor outputs of simulator and values of memory areas (registers) by changing required input values. SCADA software will be also able to communicate with a real PLC by Modbus TCP protocol.

Railway interlocking and signaling systems have a vital role in controlling and assuring the safety of railway transport. EN 50126, EN 50128 and EN 50129 and EN 61508 standards relating to the functional safety requirements for railway systems have been published by CENELEC (European Electrotechnical Committee). The standard EN 50126 includes the topics of reliability, availability, maintenance and safety analysis. EN 50128 includes software development requirements and EN 50129 includes hardware requirements. In addition to these standards, EN 61508 is a standard that includes requirements for functional safety of all electronic and programmable devices.

There have been many studies in the literature related to the implementation of functional security in railway applications. CENELEC recommended some of the railway signaling system design methods. These methods include Finite State Machines (state transition diagrams) and Petri Nets, and other formal and semi-formal methods. Petri Nets and Automation Petri Nets were used in many studies in the literature for the design of railway signaling systems and automation. In these studies, the designed pattern is implemented by a PLC.

Simulation is very important for railway signaling systems. Two trains have crashed at Ladbroke Grove in London on 5 October 1999. After this train accident has happened, the importance in training of railway personnel has increased. Therefore, many railway system manufacturers have provided their train control centers with simulators in order to prevent train accidents. There are some studies in the literature about the railway interlocking design with the help of UML diagrams. The analysis of the studies in the literature indicates that simulators mostly have been developed using PLC.

This study includes a software simulator that can run on PC and does not require PLC. Purpose of this study is to develop a railway interlocking and signaling system simulator and SCADA software. Simulation environment, which does not require any hardware other than a PC, was aimed to be developed. The developed simulator will be able to be used for development, testing and training purposes. The implementation of the designed algorithms in PLC, installation of a testing environment and carrying out software tests can take lots of time. For this reason, many of PLC and automation system manufacturer have added software simulation module to their programming software applications. Testing and development time are extremely shortened with these simulators. In this study, a PLC simulator specific to the railway signaling system has been developed. SCADA software, which the PLC simulator communicates with, has also been developed in this study.

The components of the system and the expected system functions have been firstly described for the design of railway signaling system. System design has been established with Object Oriented Programming method by drawing UML diagrams. In Object Oriented Programming, classes are used to define the common features and operations of objects. The variables or features of objects can be defined as attributes, whereas the commands or operations of these objects can be defined as methods. Objects are instances of classes, in another word objects are generated from classes. The generated object shows the similar or same characteristics and behaviors as the objects generated from the same class. As a result, by using classes, design and coding becomes easier and simpler because there is no need to write all the code for each object.

The actual system components will be included as inputs and outputs of the software. The simulator will be able to run similar to the real interlocking system. The developed simulator simulates PLC program cycle and the inputs and outputs of the PLC. SCADA software visualizes the variables of the simulation. Status of railway signaling system components can be monitored and the route reservation requests can be made through SCADA software. The communication between the simulator

and SCADA is carried out through TCP / IP. Modbus TCP is used as communication protocol.

Simulation software checks the suitability of the route reservation requests of SCADA according to train existence on railway blocks, switches' positions and route conflicts. It also sends information to SCADA about the suitability of route request.

In this study, the railway signaling system, PLC and SCADA software simulator were designed and developed. A software library was created for simulation, control and monitoring of railway signaling systems by using Object-Oriented Programming method. Software models of the railway signaling system components were created. By this way, a simulation environment will be able to be created for different railway yards. Simulation environment for a railway yard can be generated with the help of railway yard topology, interlocking table, and route conflict table. The developed simulators and SCADA software were tested by considering different scenarios. The results of these tests indicate that the functions of the software have been fulfilled. Information such as the presence of train in railway blocks can be monitored using the developed software. Route reservation state can also be monitored with the help of these software applications. In addition, switches' position, lock status, signal status can be monitored. Operator can make route reservation requests. The information regarding the appropriateness of route is received from interlocking system PLC simulator and the operator is informed. If the reservation is appropriate, switches are taken to the appropriate position and then approval of operator is requested. When the approval of the operator is received, switches are locked. Route reservation takes place by making the signal declaration after approval of operator.

The route reservation completely terminates when the train gets into last railway block. If the switches do not take their appropriate position in the specified period of time or the approval of route reservation is not received, the operator is informed about the timeout.

For the future study, features such as enabling user to define the railway area, interlocking table and route conflict configurations can be added to the developed software applications. The feature of saving and loading these configurations can also be implemented. By this way, the user can create the required simulation environment of a railway yard by using a configuration interface. Many features may be added to the software such as recording the actions of operator. In addition, the feature of simulating train movements can be added. By this way, the movements' of trains can be simulated automatically by defining trains, their directions and speeds. There are so many other features available, which can be added to these software applications.

The aim of this thesis and literature studies have been mentioned in the introduction of this study. Centrally controlled railway signaling systems and their components were introduced in the second section of the study. In the third section, the design stages were discussed. The development stage of the software was described in Section 4. The test results are reported in Section 5. In the conclusion, the obtained results and recommendations took place.

# 1. GİRİŞ

Demiryolu anklaşman ve sinyalizasyon sistemleri, demiryolu ulaşımının kontrol ve güvenliğinin sağlanmasında hayati öneme sahiptir. Demiryolu sinyalizasyon sistemlerinde kullanılan yazılım ve donanımların güvenilir olması gerekmektedir. Bu bölümde, tezin amacından ve konu hakkında yapılan literatür araştırmalarından ve demiryolu sinyalizasyon sistemlerine ait güvenlik standartlarından bahsedilmiştir.

Bu çalışmanın giriş bölümünde tezin amacı ve literatür çalışmalarından bahsedilmiştir. 2. Bölümde merkezi kontrollü demiryolu sinyalizasyon sistemi ve bileşenleri tanıtılmıştır. 3. Bölümde tasarım aşamasından bahsedilmiştir. 4. Bölümde yazılımların geliştirilmesi, 5. bölümde ise test aşaması anlatılmıştır. Sonuç bölümünde ise elde edilen sonuçlara ve önerilere yer verilmiştir.

## 1.1 Tezin Amacı

Bu çalışmanın amacı, demiryolu anklaşman ve sinyalizasyon sistemi simülatörü ve SCADA yazılımını geliştirmektir. Bir adet PC dışında herhangi bir donanım gerektirmeyen bir simülasyon ortamının geliştirilmesi amaçlanmıştır. Geliştirilen simülatör, test ve eğitim amaçlı olarak kullanılabilecektir. Tasarlanan algoritmaların PLC'de gerçeklenmesi, test ortamının kurulup, testlerinin yapılması uzun zaman alabilmektedir. Bu yüzden birçok PLC ve otomasyon sistemi üreticisi, programlama yazılımlarına simülasyon modülü eklemişlerdir. Bu simülatörler ile yazılımların geliştirme ve test süresi oldukça kısalmaktadır. Bu tezde, demiryolu sinyalizasyon sistemine özel PLC Simülatörü ve bu simülatörle haberleşen bir SCADA yazılımının geliştirilmesi amaçlanmıştır.

Simülatör yazılımında, gerçek sistemdeki bileşenler, yazılımın giriş ve çıkışları olarak yer alacaktır. Simülatör, gerçek anklaşman sistemine benzer şekilde çalışacaktır. Simülatör, PLC ve PLC'ye ait giriş çıkışları simüle edecektir. SCADA sistemi arayüzü, simülasyon yazılımına ait değişkenlerin izlenebildiği, rota tanzim isteklerinin yapılabileceği bir arayüz olarak tasarlanmıştır.

SCADA ve simülatör arasındaki haberleşme TCP/IP ile gerçekleştirilecektir. Haberleşme protokolü olarak Modbus TCP kullanılacaktır.

Simülasyon yazılımı, SCADA'dan gelen rota tanzim isteklerini, mevcut rota tanzim durumları, ray bloklarındaki tren varlık yokluk bilgisi, makas konumları ve rota çakışması durumlarına göre değerlendirecek ve SCADA arayüzüne tanzimin uygun olup olmadığı bilgisini gönderecektir.

### 1.2 Literatür Araştırması

Tezin ilk aşamasında literatürde yapılan çalışmalar incelendi. Mevcut çalışmalara yenilik ve farklılık getirebilmek amaçlandı. Literatür araştırmasında IEEE, ScienceDirect v.b. elektronik kaynaklardan faydalanılmıştır. Demiryolu sistemleriyle ilgili fonksiyonel güvenlik gereksinimleri hakkında EN 50126, EN 50128 ve EN 50129 ve EN 61508 standartları CENELEC (Avrupa Elektroteknik Komitesi) yayınlanmıştır[1]. Bu standartlardan EN 50126, güvenilirlik, kullanılabilirlik, bakım-onarım ve güvenlik analizi konularını içermektedir. EN 50128 yazılım geliştirme gereksinimlerini, EN 50129 donanım gereksinimlerini içermektedir. EN 61508 ise tüm elektronik ve programlanabilir cihazlara ait fonksiyonel güvenlik gereksinimlerini içeren bir standarttır [1,4]. Literatürde, demiryolu uygulamalarında fonksiyonel güvenliğin uygulanmasına yönelik birçok çalışma bulunmaktadır [1-4].

CENELEC tarafından demiryolu sinyalizasyon sistemlerinin tasarımında tavsiye edilen yöntemler, Sonlu Durum Makineleri (Durum Geçiş Diyagramları), Petri Netler ve diğer biçimsel ve yarı-biçimsel yöntemler olarak verilmiştir [4]. Literatürde, demiryolu sinyalizasyon sistemlerinin tasarımlarına yönelik birçok çalışmada Petri Ağları ve Otomasyon Petri Ağları kullanılmıştır. Bu yapılan çalışmalarda tasarım sonucu elde edilen model PLC ile gerçeklenmiştir [5-10].

Demiryolu sinyalizasyon sisteminde simülasyonun büyük önemi bulunmaktadır. Ladbroke Grove'da meydana gelen tren kazasından sonra, simülasyonun demiryolu personelinin eğitimindeki önemi artmıştır. GE ve Westinghouse, kurdukları tren kontrol merkezlerine simülatör sağlamışlardır [13]. Ayrıca, PLC tabanlı anklaşman sistemleri için, PLC ve PC'den oluşan bir test ortamı geliştirilmiştir [14]. Buradaki test ortamında, anklaşman sistemi ile PC arasındaki haberleşme PLC'si ile

gerçekleşmektedir [14]. Benzer şekilde PLC'nin tren kontrol simülasyonunda kullanıldığı, başka çalışmalar da bulunmaktadır [15-16]. UML diyagramları ile demiryolu anklaşman tasarımı yapılan çalışmalar da mevcuttur [17-19]. Pataricza ve diğerleri [17] tarafından kritik güvenli bir demiryolu kontrolü yazılım modülünün UML ile analizi yapılmıştır. UML diyagramları ve Petri Ağlarının birlikte kullanıldığı dağılmış yapıdaki kontrol sistemlerinin analizi ve tasarımına yönelik bir çalışma da bulunmaktadır [20].

Literatürde yapılan çalışmalar incelendiğinde simülatörlerin genel olarak PLC kullanılarak geliştirildiği görülmüştür. Mevcut çalışmada ise, farklı olarak PC'de çalışabilen, donanım gerektirmeyen bir yazılım simülatörü geliştirilmiştir.

## 2. MERKEZİ KONTROLLÜ DEMİRYOLU SİNYALİZASYON SİSTEMİ

Demiryolu ulaşımının giderek hızlanması ve sürekli olarak büyümesi, demiryolu sinyalizasyon ve anklaşman sistemlerinin merkezden izlenmesini ve kontrolünü gerekli hale getirmiştir. Bu bölümde, tasarıma geçmeden önce demiryolu sinyalizasyon sistemine ait bileşenler tanıtılacaktır. Merkezi kontrollü demiryolu sinyalizasyon sistemi üç ana bileşenden oluşmaktadır. Bu bileşenler; trafik kontrol merkezi, saha donanımları ve anklaşman sistemidir.

### 2.1 Trafik Kontrol Merkezi

Trafik kontrol merkezi, tren trafiğinin operatörler tarafından SCADA yazılımı ile yönetildiği merkezdir. Tüm saha donanımlarından alınan bilgiler trafik kontrol merkezinden izlenebilmektedir. SCADA yazılımı, sahadaki anklaşman sistemi ile haberleşmektedir. Kontrol merkezindeki operatör, rota tanzim isteklerini SCADA yazılımı ile gerçekleştirir. Rota tanzim isteği, SCADA tarafından ağ üzerinden TCP/IP v.b. protokol ile anklaşman sistemine iletilir. Tanzim isteği anklaşman sistemi tarafından değerlendirilir ve SCADA'ya tanzimin uygun olup olmadığı bilgisi gönderilir. Operatör, saha donanımlarının durumlarını kontrol merkezinden SCADA ile izleyebilmektedir. Şekil 2.1'de bir tren kontrol merkezi görülmektedir.

**Şekil 2.1 :** Tren kontrol merkezi [25].

## 2.2 Saha Donanımları

Demiryolu sinyalizasyon sistemine ait saha donanımları ray devreleri, makaslar ve sinyallerdir. Bu donanımlar, anklaşman sistemi tarafından kontrol edilmektedir.

### 2.2.1 Ray devreleri

Ray devreleri, raylar üzerindeki tren varlığını algılayan elektrik devreleridir [6]. Ray devreleri tren varken lojik 0 işareti, tren yokken lojik 1 işareti üretir. Bu sayede, lojik işareti taşıyan kabloda herhangi bir kopma olduğu zaman da tren varmış gibi algılanır. Bu durum, demiryolu sinyalizasyon sisteminin hatada güvenli (fail-safe) bir sistem olarak çalışmasını sağlar.

### 2.2.2 Ray blokları

Aynı anda bir tek trenin bulunmasına izin verilen bir veya daha fazla ray devresinden oluşan bölüme ray bloğu adı verilir.

### 2.2.3 Makaslar

Trenlerin, diğer tekerlekli araçlardan farkı yönlendirme sistemlerinin bulunmamasıdır [10]. Trenlerin raylar arasındaki geçişleri makaslar ile sağlanmaktadır. Makasların konum bilgileri, kilit durumları algılanabilmektedir. Makaslara konum değiştirme ve kilitleme kumandası verilebilmektedir. Bu çalışmada, makasın treni yoluna sapma yaparak devam ettiren konumu "konum 1" olarak, makasın treni düz istikamette sapma yapmadan yoluna devam ettiren konumu ise "konum 0" olarak kabul edilmiştir. "Konum 0" ve "konum 1" girişleri ayrı dijital girişler olarak algılanmaktadır.

### 2.2.4 Sinyaller

Sinyaller, demiryolu ulaşımının güvenliğini sağlayan, makinisti ilerideki ray bloklarının meşguliyeti konusunda bilgilendiren renkli ve ışıklı işaretlerdir. Bu işaretlerin anlamları kısaca şu şekilde özetlenebilir:

- Kırmızı sinyal; rotanın tanzim edilmediğini veya ilerideki ray bloğunun dolu olduğunu makiniste bildirir. Tehlikeli bir durum var anlamına gelir.

- Sarı sinyal; makiniste, iki sonraki ray bloğunun dolu olduğunu veya bir sonraki sinyalin kırmızı olduğunu ve hızını düşürmesi gerektiğini bildirir. Uyarı anlamına gelir.
- Yeşil sinyal ise rotanın açık olduğunu ve bir sonraki sinyalin de harekete engel teşkil etmediğini makiniste bildirir [6].

### 2.3 Anklaşman Sistemi

Dispeçer olarak adlandırılan Demiryolu SCADA operatörünün rota tanzim isteklerini değerlendiren, trenlerin tanzim edilen rota üzerinde güvenli olarak hareketini sağlayacak ayarlamaları yapan sistemdir [10]. Rota tanzimi, makasların konumlarının ayarlanması, uygun sinyal bildirimi, anklaşman sistemi tarafından gerçekleştirilir.

# 3. DEMİRYOLU SİNYALİZASYON SİSTEMİ TASARIMI

Demiryolu sinyalizasyon sistemi tasarımında ilk olarak sisteme ait bileşenler ve sistemden beklenen işlevler tanımlanmıştır. Sistem tasarımı, Nesne Yönelimli Programlama (Object Oriented Programming) yöntemiyle, UML diyagramları çizilerek gerçekleştirilmiştir.

Nesne yönelimli programlamada, ortak özelliklere ve yeteneklere sahip nesneler "Sınıf" adı verilen soyut bir kavramla tanımlanabilmektedir. Nesnelerin değişkenleri nitelikler olarak, bu nesnelere ait komutlar da metotlar olarak tanımlanmaktadır [20]. Nesne, sınıfın somut bir örneğidir. Aynı tür sınıftan üretilen nesneler benzer davranış ve özellikleri göstermektedir. Bu sayede her bir nesne için ayrı ayrı tasarıma ve kodlamaya ihtiyaç duyulmamaktadır. Tasarım ve kodlama oldukça basitleşmektedir.

## 3.1 İşlevler

Sistemin tasarımına başlamadan önce sistemin işlevleri belirlenmiştir. Bu bölümde rota tanziminin yapılması ve sonlandırılması, makasların konumlandırılması ve sinyallerden bahsedilmiştir.

### 3.1.1 Rota tanziminin yapılması ve sonlandırılması

i. Kontrol Merkezi, rota tanzim talebinde bulunur.

ii. Anklaşman sistemi; ilgili ray devreleri, kilit ve rota çakışmasını değerlendirir.

   a. Koşullar uygun değilse tanzim isteği geri çevrilir ve bu durum kontrol merkezine bildirilir.

   b. Koşullar uygun ise ilgili makasların doğru biçimde konumlandırılması için gerekli kumanda işaretleri üretilir.

iii. Anklaşman Sistemi, makaslar uygun konumda ve ilgili ray devreleri uygun ise rota tanzimine hazır olduğunu kontrol merkezine bildirir.

   a. Kontrol merkezinden onay gelmez ise tanzim talebi iptal edilir.

b. Kontrol merkezinden onay gelirse ilgili makaslara kilitleme kumandası verilir.

iv. İlgili makaslardan en az birinden kilitlenmedi bilgisi alınırsa tanzim talebi reddedilir ve kontrol merkezine bildirilir.

v. İlgili makasların tümünden kilitlendi bilgisi alınır ve diğer koşullar da uygunsa (rota çakışması yok ve ray devreleri boş) rotanın tanzimi ve gerekli sinyal bildirimi yapılır.

vi. Tren, tanzimi yapılan rotayı kullanıp son ray devresine ulaşınca tanzim sonlandırılır ve ilgili kilitler çözülür [11].

### 3.1.2 Rota tanziminin iptal edilmesi

i. Kontrol Merkezinden tanzim iptal isteği gelir.

   a. Rotanın ilk bloğu boş değilse istek reddedilir.

   b. Rotanın ilk bloğu boş ise istek kabul edilir.

ii. İlgili sinyale Kırmızı bildirimi verilmesi sağlanır.

iii. Giriş sinyalinden önceki ray devresi boş ise

   a. 30 saniye beklenir.

   b. Giriş sinyalinden önceki ray devresi boş ise rotadaki ilgili kilitlerin çözülmesi sağlanır.

iv. Giriş sinyalinden önceki ray devresi boş değil ise 180 saniye beklenir.

v. Giriş sinyalinden önceki ray devresi boş ise rotadaki kilitlerin çözülmesi sağlanır.

vi. Giriş sinyalinden önceki ray devresi boş değil ise mevcut durum korunur ve tanzim sonlandırma isteği reddedilir [11].

Bu çalışmada, simülasyonu yapılan demiryolu bölgesinde giriş sinyalinden önceki ray devresi bulunmamaktadır. Ayrıca, rota tanziminin iptali için simulatör ve SCADA yazılımlarında birçok ilave değişiklik yapılması gerekmektedir. Örneğin tanzim iptal butonları ve ilgili komutların ilave edilmesi gerekmektedir. Bu yüzden yazılımlarda rota tanzim iptali gerçekleştirilememiştir. Gelecekteki çalışmalarda rota tanziminin yazılımlara eklenmesi planlanmaktadır.

### 3.1.3 Makas konumunun değiştirilmesi

a. Kontrol merkezinden makas konumunu değiştirme isteği alınır.

   i. Makas elektronik kilitli değildir ve makas bölgesi ray devresi boştur.

   ii. Makas konumunun değiştirme süreci başlatılır ve sonlandırılır.

b. Makas konumu değiştirme süreci

   i. İlgili makasın konumu okunur

   ii. Aynı güç kaynağına bağlı bir tek makas motorunun çalışmasına izin verilir.

   iii. İlgili makasın 7 s içinde istenen konuma ulaşması beklenir.

   iv. Konum değiştirme işlemi tamamlanır.

c. Makas kilitlidir veya makas bölgesi ray devresi boş değilse makas konum değiştirme isteği reddedilir [11].

### 3.1.4 Sinyaller

Sinyaller ray devrelerinde tren bulunmasına, makas konumları ve rota tanzim durumuna göre farklı renk lambaları yakmaktadır. Sinyaller ve anlamları Tablo 3.1'de verilmiştir.

**Çizelge 3.1 :** Sinyal lambaları ve renkler [11].

| Sinyal | Sinyalin Anlamı | Açıklama |
|---|---|---|
| Kırmızı | Dur | Bir öndeki ray devresi dolu veya hat tanzimi yapılmamış |
| Sarı | Kontrollü ilerle | Bir sonraki ray devresi boş, ondan sonraki dolu |
| Yeşil | İlerle | Önündeki en az iki ray devresi boş |
| Sarı üzeri Kırmızı | Her an durabilecek şekilde ilerle | Bir sonraki makas bölgesinden sapma yapılacak, ancak öndeki ray devresi dolu |
| Sarı üzeri Sarı | Kontrollü ilerle | Bir sonraki makas bölgesinden sapma yapılacak ve önündeki bir ray devresi boş |
| Sarı üzeri Yeşil | İlerle | Bir sonraki ray devresinden sapma yapılacak ve önündeki en az iki ray devresi boş |

### 3.2 UML Diyagramları

İşlevleri belirlenen demiryolu sinyalizasyon sisteminin UML diyagramları oluşturulmuştur. UML diyagramları, açık kaynak kodlu WhiteStarUML yazılımı ile çizilmiştir. Tasarım aşamasında, bileşen diyagramı, sınıf diyagramları, durum diyagramı ve sıra diyagramı çizilmiştir.

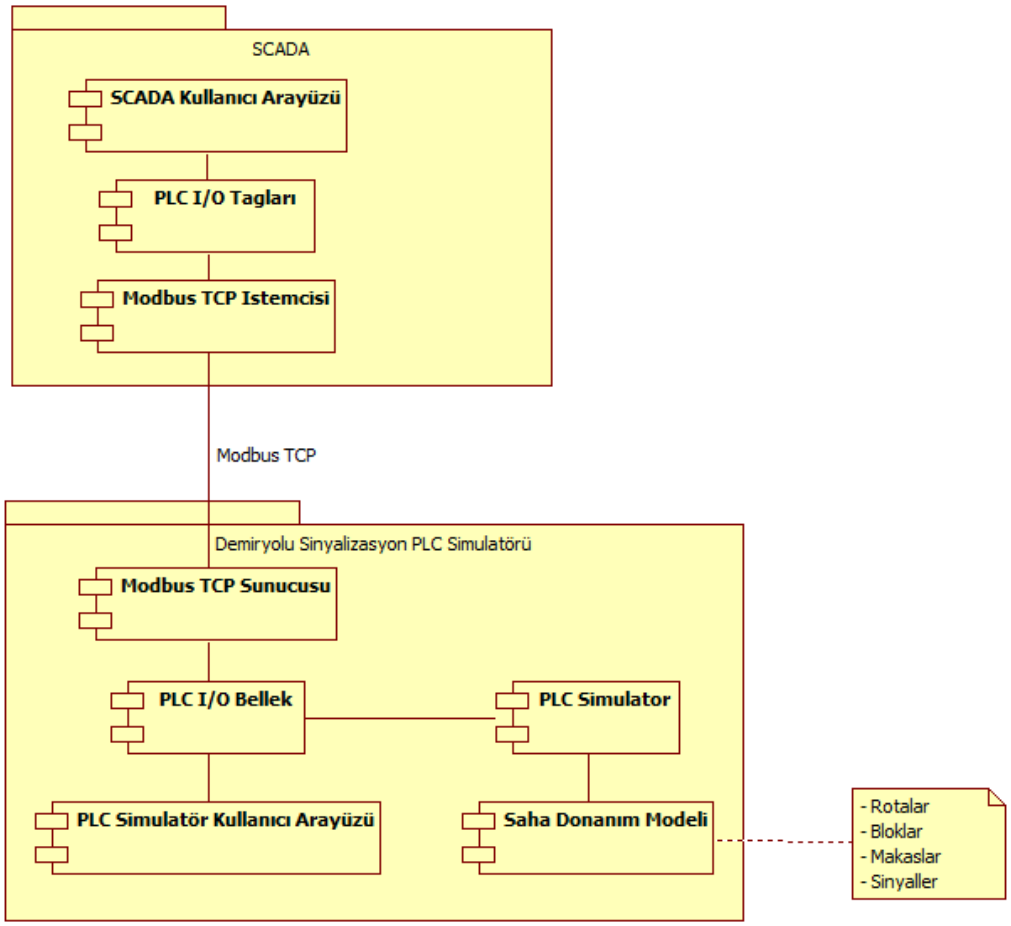
#### 3.2.1 UML bileşen diyagramı

Öncelikle sistemdeki bileşenleri ve bu bileşenler arasındaki iletişimi gösteren bileşen diyagramı Şekil 3.1'deki gibi çizilmiştir. Geliştirilen sistem, SCADA ve PLC Simülatörü olmak üzere iki ana uygulamadan oluşmuştur. SCADA ve PLC Simülatörü Modbus TCP protokolü ile haberleşmektedir. PLC Simülatörü; kullanıcı arayüzü, PLC belleği, PLC simülasyon motoru, saha donanım modeli ve Modbus TCP sunucusundan oluşmaktadır. Saha donanım modeli ise, rotalar, ray blokları, makaslar ve sinyallere ait modellerden oluşmaktadır. PLC simülasyon motoru, PLC belleğindeki giriş görüntü belleğindeki değerlere göre çalışarak, çıkış değerleri üretmektedir. Kullanıcı arayüzü ile dijital giriş değerleri kullanıcı tarafından değiştirilebilmekte, simülatörün ürettiği çıkış değerleri izlenebilmektedir. Üretilen değerler PLC çıkış görüntü belleğine ve bellek alanlarına yazılmaktadır.

Modbus TCP sunucusu, SCADA'dan gelen okuma, yazma işlemlerini değerlendirerek PLC belleği üzerinde okuma ve yazma işlemlerini gerçekleştirmektedir.

SCADA yazılımı; kullanıcı arayüzü, PLC I/O Tagları ve Modbus TCP istemcisinden oluşmaktadır. Modbus TCP istemcisi, PLC Simülatörü uygulamasındaki Modbus TCP sunucusuna 502 numaralı porttan bağlanarak, PLC Simülatörünün belleğindeki giriş/çıkış görüntü belleği ve bellek alanlarındaki değerleri Modbus TCP protokolü ile okumaktadır. Benzer şekilde, istenilen bellek alanlarına SCADA tarafından Modbus TCP protokolü ile yazma işlemi yapılabilmektedir.

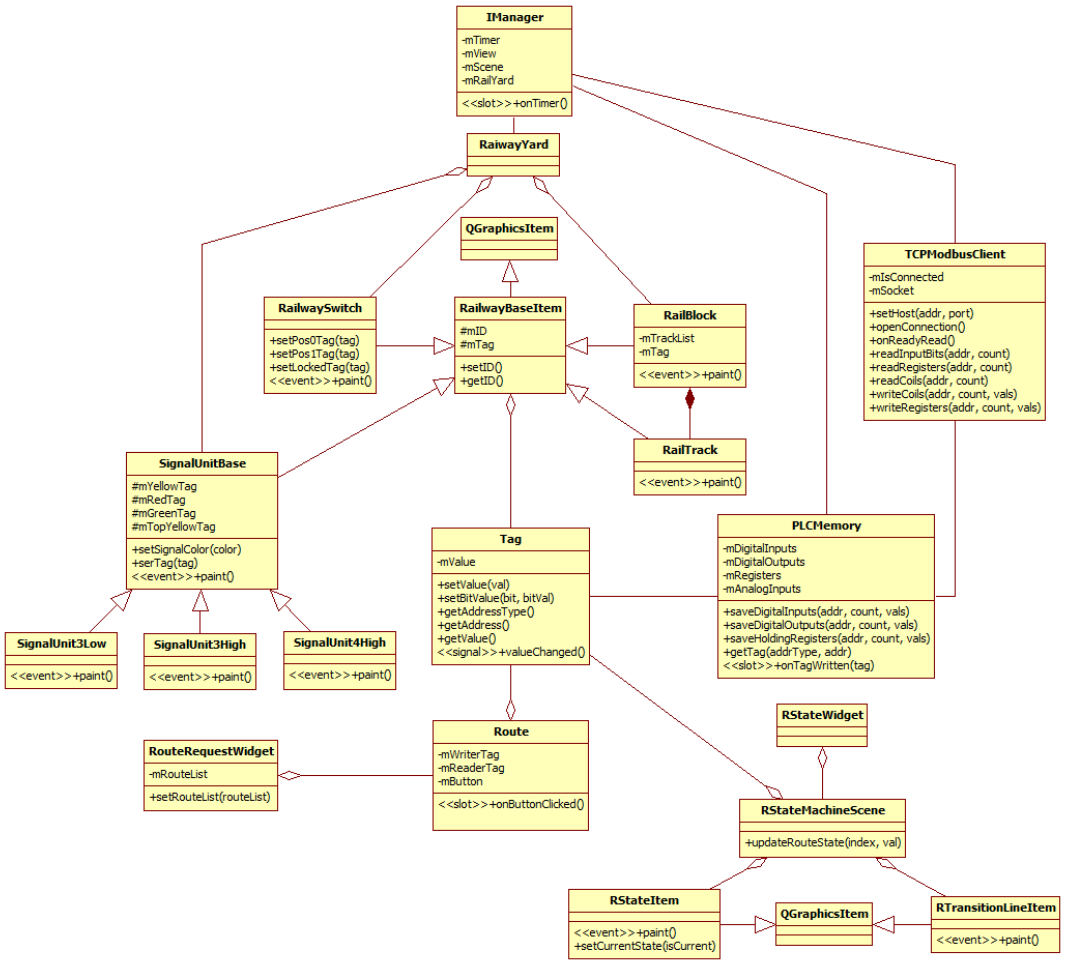
PLC I/O Tagları (etiketleri), bellek alanlarına ait değerlerin kullanıcı arayüzü tarafından görsel hale getirilmesini sağlar. Görsel nesneler ile bellek değişkenleri arasındaki bağlantı taglar ile kurulur. Örneğin ray bloğu, ilgili dijital giriş lojik 0'ken (tren varken) kırmızı, lojik 1'ken (tren yokken) yeşil renkte gösterilir. Benzer şekilde sinyaller ve makasların durumları da taglar ile SCADA'da gösterilir.




**Şekil 3.1 :** UML bileşen diyagramı.

### 3.2.2 UML sınıf diyagramları

UML sınıf diyagramı, sınıflara ait temel özellikleri ve işlemleri (metotları) gösteren ve sınıfların birbiriyle olan etkileşimlerini gösteren diyagramdır. Sınıf diyagramı, sınıflar arasındaki statik ilişkileri göstermektedir [20]. Sınıflar arasında kalıtım (inheritance), toplama (aggregation), oluşum (composition), bağıntı (association) ve bağımlılık (dependency) ilişkileri bulunabilmektedir. UML sınıf diyagramları, SCADA ve simülatör için ayrı olarak özet halinde çizilmiştir. SCADA yazılımı için çizilen sınıf diyagramı Şekil 3.2'deki gibidir. Burada RailwaySwitch, RailBlock, RailTrack, SignalUnitBase sınıfları RailwayBaseItem sınıfından türetilmiştir (kalıtım ilişkisi). RailwayBaseItem Tag nesnesini içerdiği için aralarında toplama ilişkisi bulunmaktadır. PLCMemory (PLC belleği) ile TCPModbusClient (TCP Modbus istemcisi) ve Tag sınıfı arasında bağıntı ilişkisi vardır. TCPModbusClient ve Tag sınıfı PLCMemory nesnesiyle haberleşmektedir. PLC belleği üzerinde okuma ve yazma işlemlerini gerçekleştirmektedirler.




**Şekil 3.2 : SCADA** yazılımı UML sınıf diyagramı**.**

PLC belleği dijital girişler, dijital çıkışlar, bellek alanları (register) ve analog girişlerin değerlerini nitelik olarak içermektedir. Ayrıca bu değerlerin belleğe yazılmasını sağlayan metotlara sahiptir. PLCMemory nesnesi istenilen tür değişkene (dijital giriş, dijital çıkış veya bellek alanı) ve adrese ait Tag nesnesi oluşturmakta ve bu Tag nesnesinin adresini geri döndürmektedir. Tag nesnesi ile görsel elemanlar arasında Qt'deki sinyal slot bağlantısı kurulur. Böylece Tag nesnesinin bağlı bulunduğu bellek adresi değiştiğinde sinyal üretilir, bağlantı kurulan görsel nesnedeki bu sinyale bağlı olan slot metodu çağrılarak değişimden haberdar edilir.

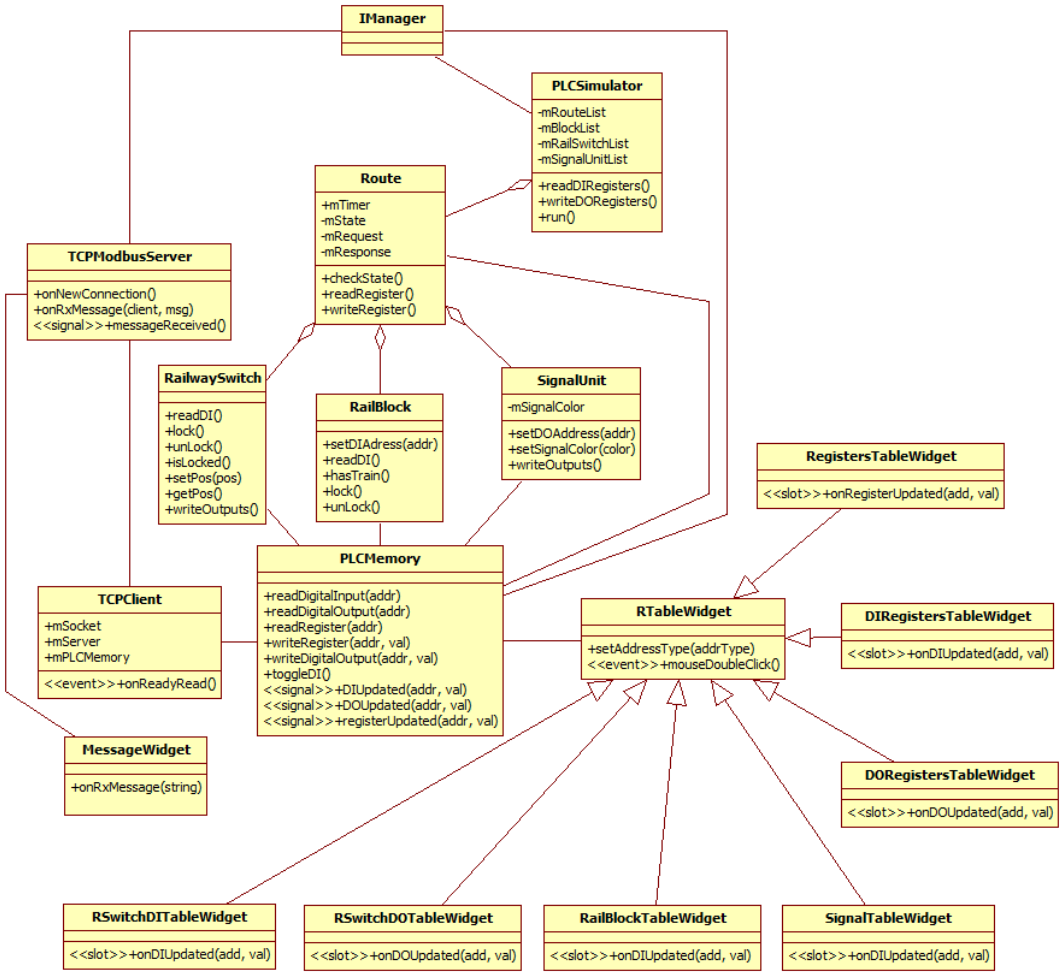
RailTrack (ray devresi) sınıfı ile RailBlock (ray bloğu) sınıfı arasında parça bütün (oluşum) ilişkisi bulunmaktadır. Çünkü ray bloğu ray devrelerinden meydana gelmektedir. RailwayYard (demiryolu bölgesi) sınıfı ise SignalUnitBase (sinyal ünitesi), RailwaySwitch (makas) ve RailBlock nesnelerinin bir araya gelmesiyle oluşur. RailwayYard ile bu nesneler arasında toplama ilişkisi bulunmaktadır. Route

(rota) nesnesi tanzim isteğinde bulunmayı ve tanzim yanıtını almayı sağlayan Tag nesnelerini içermektedir. Şekil 3.2'deki SCADA UML diyagramındaki sınıfların işlevleri Çizelge 3.1'de özetlenmiştir.

**Çizelge 3.1 :** SCADA yazılımındaki sınıfların işlevleri.

| **Sınıf** | **Anlam** | **İşlev** |
|---|---|---|
| IManager | Yönetici sınıf | Sınıflar arasındaki ilişkileri kuran ve yöneten sınıf |
| RailwayYard | Demiryolu bölgesi | Tüm demiryolu nesnelerini içeren ve bunlar aralarındaki bağlantıları kuran sınıf |
| QGraphicsItem | Qt grafik bileşeni sınıfı | RailwayBaseItem sınıfı ve diğer görsel nesnelere ait sınıflar bu sınıftan türetilir. |
| RailwayBaseItem | Demiryolu temel sınıfı | Makas, ray devresi vb. demiryolu nesneleri bu grafik sınıfından türetilir. |
| RailwaySwitch | Makas | Makasların konumunu gösteren görsel nesne |
| RailTrack | Ray devresi | Ray devresi görsel bileşeni |
| RailBlock | Ray bloğu | Ray bloğu görsel bileşeni |
| Route | Rota | Rota isteklerini ve anklaşman sisteminden gelen cevapları değerlendiren sınıf |
| RouteRequestWidget | Rota tanzimi görsel bileşeni | Rota tanziminin isteklerinin yapıldığı ve sonuçları gösteren görsel bileşen |
| SignalUnitBase | Temel sinyal ünitesi | Temel sinyal ünitesi |
| SignalUnit3Low | Üçlü cüce sinyal ünitesi | Üçlü cüce sinyal ünitesi sınıfı |
| SignalUnit3High | Üçlü yüksek sinyal ünitesi | Üçlü yüksek sinyal görsel sınıfı |
| SignalUnit4High | Dörtlü yüksek sinyal ünitesi | Dörtlü yüksek sinyal görsel sınıfı |
| PLCMemory | PLC belleği | PLC'den okunan değerleri tutan sanal bellek |
| Tag | Etiket | PLC belleğiyle görsel nesneler arasındaki bağlantıyı sağlayan veri elemanı |
| TCPModbusClient | TCP Modbus istemcisi | PLC Simülatörüyle Modbus/TCP protokolü ile haberleşmeyi sağlayan istemci sınıfı |
| RStateWidget | Durum diyagramı görsel bileşeni | Rotalara ait durum diyagramını ve aktif olan durumu gösteren sınıf |
| RStateMachineScene | Durum diyagramı çizim alanı | RStateItem nesnelerinden oluşan ve bu nesnelerin çizimini sağlayan sınıf. |
| RStateItem | Durum görsel nesnesi | Durum diyagramındaki tek bir duruma karşılık gelen sınıf |
| RTransitionLineItem | Durumlar arası geçiş çizgisi | Durumlar arası geçişleri gösteren görsel sınıf |

PLC Simülatörü için çizilen UML sınıf diyagramı Şekil 3.3'teki gibidir. Route (rota) nesnesi; Railblock (ray bloğu), RailwaySwitch (makas), ve SignalUnit (sinyal ünitesi) nesnelerinin bir araya gelmesiyle oluşur. RailBlock, RailSwitch ve SignalUnit nesnelerine ait dijital giriş ve dijital çıkışlar PLCMemory (PLC belleğinde) tutulmaktadır. Demiryolu bölgesindeki rotalar Route nesneleri PLCSimulator nesnesi içerisinde oluşturularak tanımlanır. Bu rotaya dahil olan ray blokları, makaslar ve sinyaller tanımlanır. PLCSimulator nesnesi tüm rotaların simülasyonunu gerçekleştirir. RSwitchDITableWidget, RSwitchDOTableWidget, RailBlockTableWidget sınıfları RTableWidget sınıfından türetilmiştir. RTableWidget sınıfı bellek değişkenlerinin gösterimini sağlayan temel sınıftır. TCPModbusServer SCADA'dan gelen bağlantıları dinler. Gelen bağlantılar için TCPClient nesnesi oluşturur. SCADA ve PLC Simülatörü arasındaki haberleşme TCPClient nesnesindeki soket tarafından sürdürülür.




**Şekil 3.3 :** PLC Simülatörü UML sınıf diyagramı.

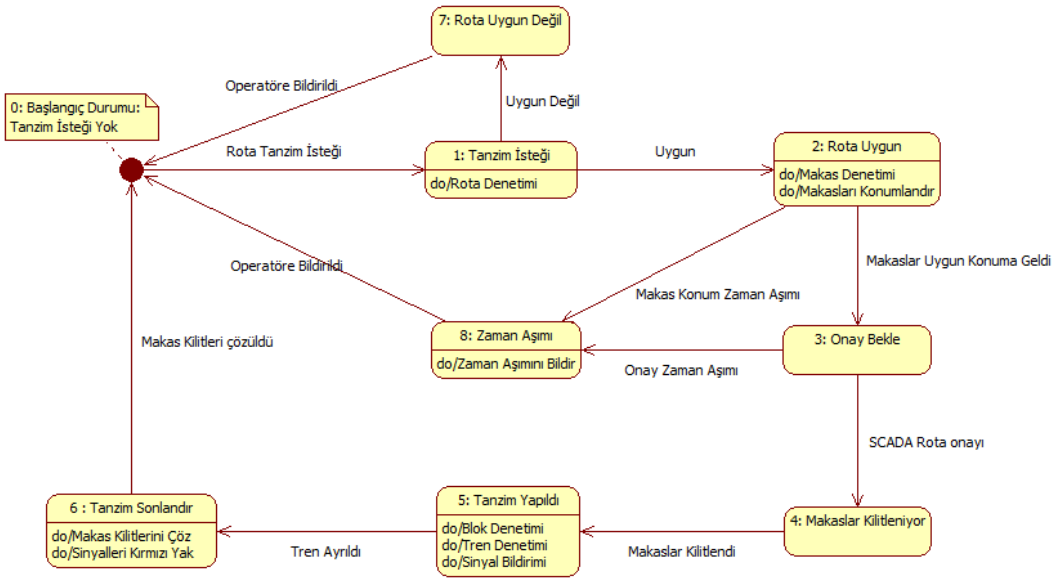
PLCMemory nesnesi üzerinde okuma ve yazma işlemleri TCPClient tarafından gerçekleştirilir. MessageWidget ise SCADA'dan gelen mesajların gösterilmesini sağlayan sınıftır. Şekil 3.3'teki PLC Simülatörü UML diyagramındaki sınıfların işlevleri Çizelge 3.2'de özetlenmiştir.

**Çizelge 3.2 :** PLC Simülatör yazılımındaki sınıfların işlevleri

| **Sınıf** | **Anlam** | **İşlev** |
|---|---|---|
| IManager | Yönetici sınıf | Sınıflar arası sinyal/slot bağlantısını kuran yönetici sınıf |
| PLCSimulator | PLC Simülatörü | Demiryolu sinyalizasyon ve anlaşman sistemi PLC simülasyon motoru |
| Route | Rota | Rota tanzimini denetleyen durum makinesi |
| RailwaySwitch | Makas | Makasa ait giriş/çıkış değerlerini içeren model. |
| RailBlock | Ray bloğu | Ray bloğuna ait model |
| SignalUnit | Sinyal ünitesi | Sinyal ünitesine ait model |
| PLCMemory | PLC belleği | PLC sanal belleği (giriş/çıkış ve bellek alanlarına ait değerleri içeren model) |
| TCPModbusServer | TCP Modbus sunucusu | SCADA istemcisinden gelen yeni bağlantı isteklerini alarak, TCPClient nesnesi oluşturur. |
| TCPClient | TCP istemcisi | SCADA istemcisinden gelen okuma ve yazma isteklerini değerlendirerek cevaplar. |
| MessageWidget | Mesaj görsel bileşeni | İstemci ve sunucu mesajlarını liste halinde gösterir. |
| RTableWidget | Bellek alanı temel görsel tablo nesnesi | Dijital giriş/çıkış ve bellek alanı değerlerinin gösterilmesini sağlayan temel sınıf |
| DITableWidget | Dijital girişler görsel tablo nesnesi | Dijital girişlerin değerlerini gösteren ve değerlerinin değiştirebildiği tablo |
| DOTableWidget | Dijital çıkışlar görsel tablo nesnesi | Dijital çıkışların değerlerini gösteren tablo |
| RegistersTableWidget | Bellek alanları görsel tablo nesnesi | Bellek alanlarının değerlerini gösteren tablo |
| RailBlockTableWidget | Ray bloğu görsel tablo nesnesi | Ray bloklarında tren varlığını gösteren tablo |
| RSwitchDITableWidget | Makas dijital girişleri görsel tablo nesnesi | Makaslara ait dijital girişlerin durumunu gösteren tablo |
| RSwitchDOTableWidget | Makas dijital çıkışları görsel tablo nesnesi | Makaslara ait dijital çıkışların durumunu gösteren tablo |
| SignalTableWidget | Sinyal tablosu görsel nesnesi | Sinyal ünitelerine ait dijital çıkışların değerini gösteren tablo |

### 3.2.3 UML durum diyagramı

UML durum diyagramı, PLC Simülatöründeki rota tanzimini denetleyen durum makinesi için Şekil 3.4'teki gibi çizilmiştir. Durum makinesinin başlangıç durumu, tanzim isteğinin olmadığı ve rota tanziminin yapılmadığı koşul için tanımlanmıştır. Rota tanzim isteği geldiğinde durum makinesi Durum 1'e geçer. Rota tanziminin uygunluğu Durum 1'de denetlenir. Eğer tanzim uygun değilse Durum 7'ye geçilir. Rotanın uygun olmadığı operatöre bildirilince Durum 0'a dönülür. Eğer tanzim uygunsa Durum 2'ye geçilir. Durum 2'de makaslar konumlandırılır ve denetlenir. Makaslar 7 s içinde uygun konuma gelmezse Durum 8'e geçilir. Makaslar uygun konuma geldiyse Durum 3'e geçilir ve SCADA'dan onay beklenir. SCADA'dan onay gelmezse Durum 8'e geçilir. Durum 8'de zaman aşımı operatöre bildirilince Durum 0'a dönülür. SCADA'dan onay gelince Durum 4'e geçilir ve makaslar kilitlenir. Eğer koşullar uygunsa ve tüm makaslar kilitlendiyse Durum 5'e geçilerek sinyal bildiriminde bulunulur ve rota tanzimi yapılır. Tren tanzim edilen rotanın başından sonuna kadar gelince Durum 6'ya geçilir. Durum 6'da makas kilitleri çözülür ve kırmızı sinyal bildiriminde bulunulur. Makas kilitleri çözülünce başlangıç durumuna (Durum 0'a) dönülür.

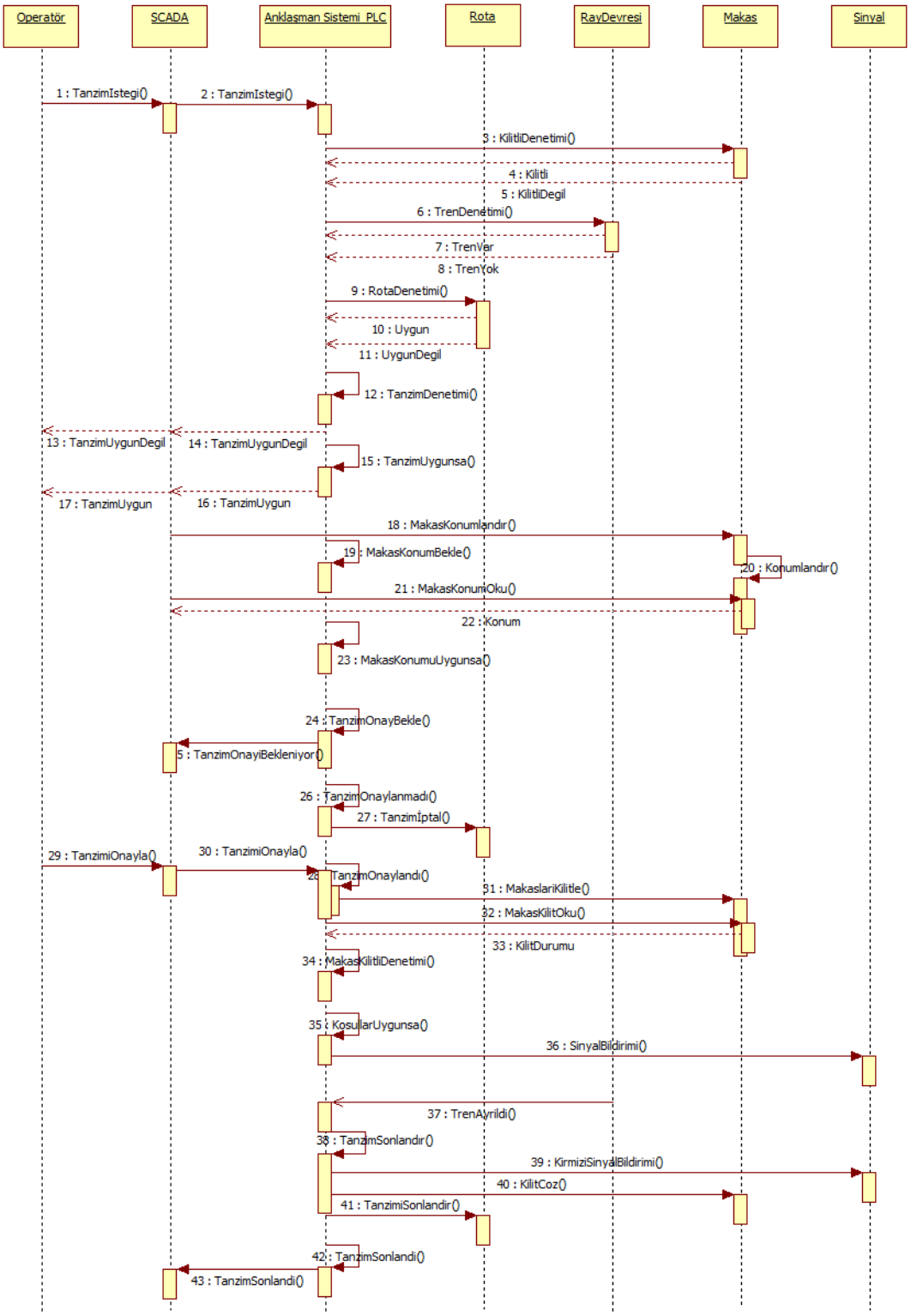



**Şekil 3.4 :** Rota tanzimini denetleyen durum makinesi için durum diyagramı.

### 3.2.4 UML sıra diyagramı

UML sıra diyagramı, nesneler arası işlemlerin zaman içerisinde hangi sırada ve nasıl gerçekleştiğini gösteren diyagramdır. Şekil 3.5'te merkezi kontrollü demiryolu

sinyalizasyon sistemi için çizilen sıra diyagramı verilmiştir. Buradaki nesneler; operatör, SCADA, anklaşman sistemi PLC'si, rota, ray devresi, makas ve sinyal olarak tanımlanmıştır.




**Şekil 3.5 :** UML sıra diyagramı.

# 4. YAZILIMLARIN GELİŞTİRİLMESİ

UML diyagramları ile tasarlanan SCADA ve PLC Simülatörü yazılımları açık kaynak kodlu Qt derleyicisinde C++ programlama dili kullanılarak geliştirilmiştir. Windows, Linux ve Embedded Linux gibi birçok işletim sistemini desteklemesi nedeniyle Qt derleyicisi tercih edilmiştir. Qt derleyicisi sinyal slot mekanizması sağlamaktadır. İki nesne arasında sinyal slot bağlantısı kurulabilmektedir. Bu sayede bir nesne sinyal ürettiğinde, diğer nesnenin bu sinyale bağlanmış olan slot metodu çağrılabilmektedir. Bu sayede nesneler arası haberleşme oldukça basitleşmektedir.

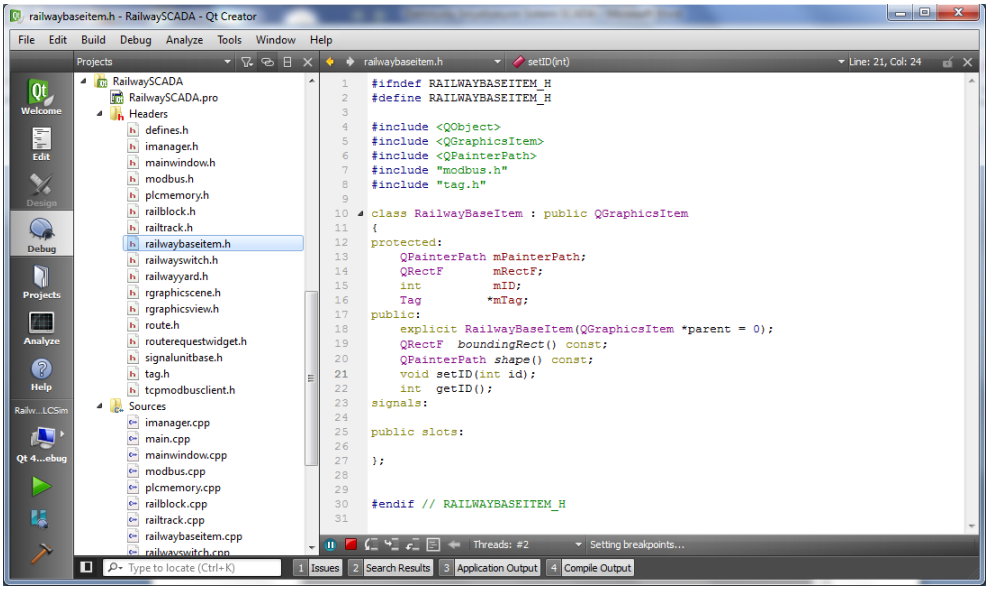
UML diyagramlarıyla tasarlanan yazılımlar C++ programlama dilinde Nesne Yönelimli Programlama yöntemiyle geliştirilmiştir. İlk olarak SCADA yazılımının geliştirilmesine başlanmıştır. Bu sayede simülatör geliştirilirken SCADA yazılımı test arayüzü olarak kullanılabilmiştir.

## 4.1 SCADA Yazılımı

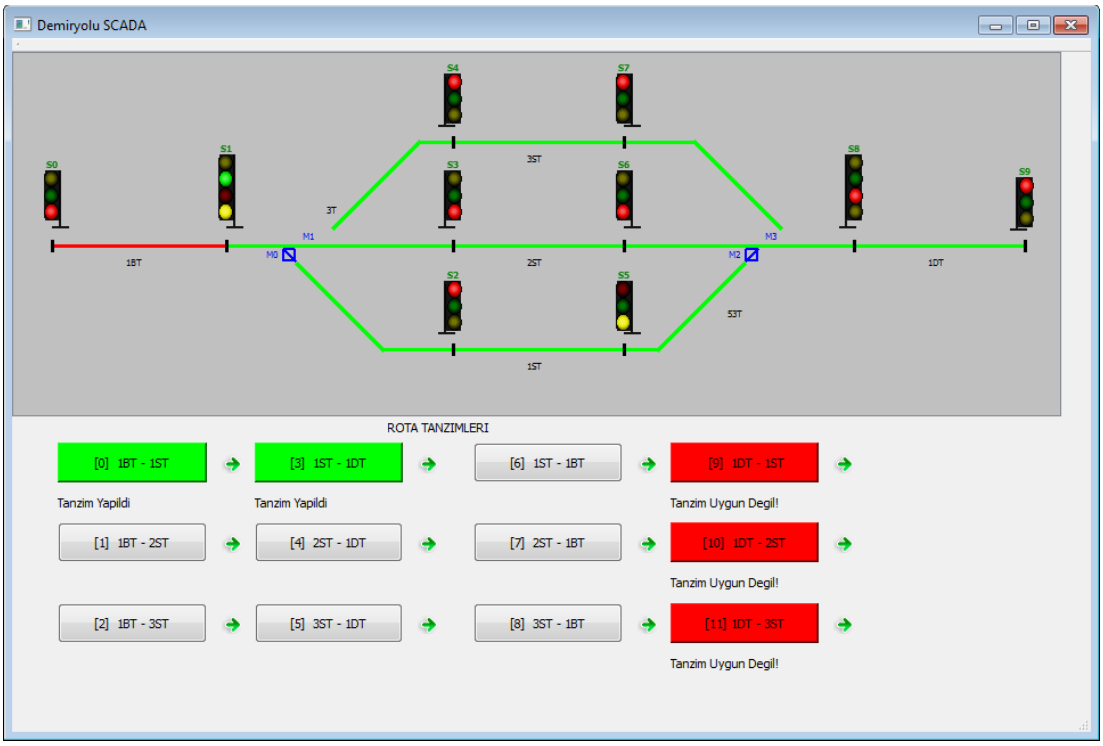
SCADA yazılımın geliştirilmesi adım adım gerçekleştirilmiştir. Öncelikle SCADA yazılımı için QT derleyicisinde proje oluşturulmuştur. Daha sonra kullanıcı arayüzünün tasarımı yapılmıştır. Arayüzdeki nesneler UML sınıf diyagramında tasarlandığı şekliyle, C++ programlama dilinde yazılmıştır. Şekil 4.1'de QT derleyicisi görülmektedir. Geliştirilen SCADA yazılımın arayüzü Şekil 4.2'deki gibidir.

SCADA yazılımında görsel nesneler ile bellek değişkenleri arasındaki bağlantı taglar (etiket) aracılığıyla kurulmuştur. Taglar; giriş, çıkış ve bellek alanı tagları şeklinde tanımlanmıştır. Görsel nesneler, QT C++ kütüphanesi kullanılarak, programlama yoluyla çizilmiştir. Görsel nesnelerde ikon veya resim dosyası kullanılmamıştır. Bu sayede nesnelere program çalışma zamanında (run-time) istenilen şekil, renk ve boyut programlama yoluyla verilebilmektedir. Bu sayede herhangi bir resim editörü kullanmaya ihtiyaç duyulmamaktadır.

Özellikle SCADA uygulamalarında görsel nesnelerin renk, şekil ve boyutları PLC'den okunan değişkenlerin değerine göre dinamik olarak değişmektedir.



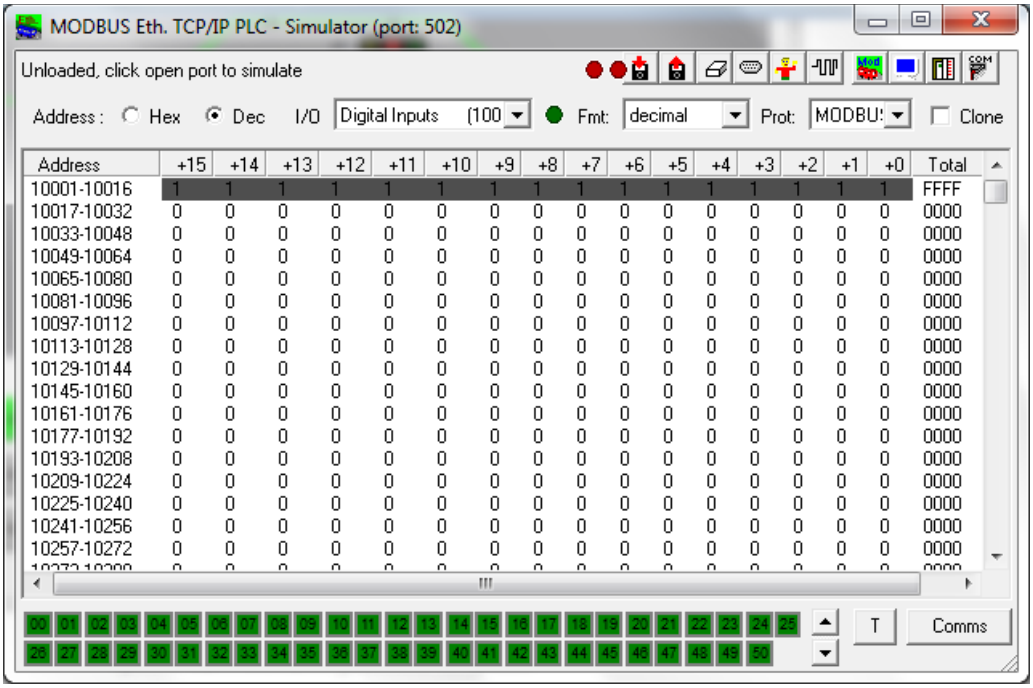

**Şekil 4.1 :** Qt derleyicisi arayüzü.



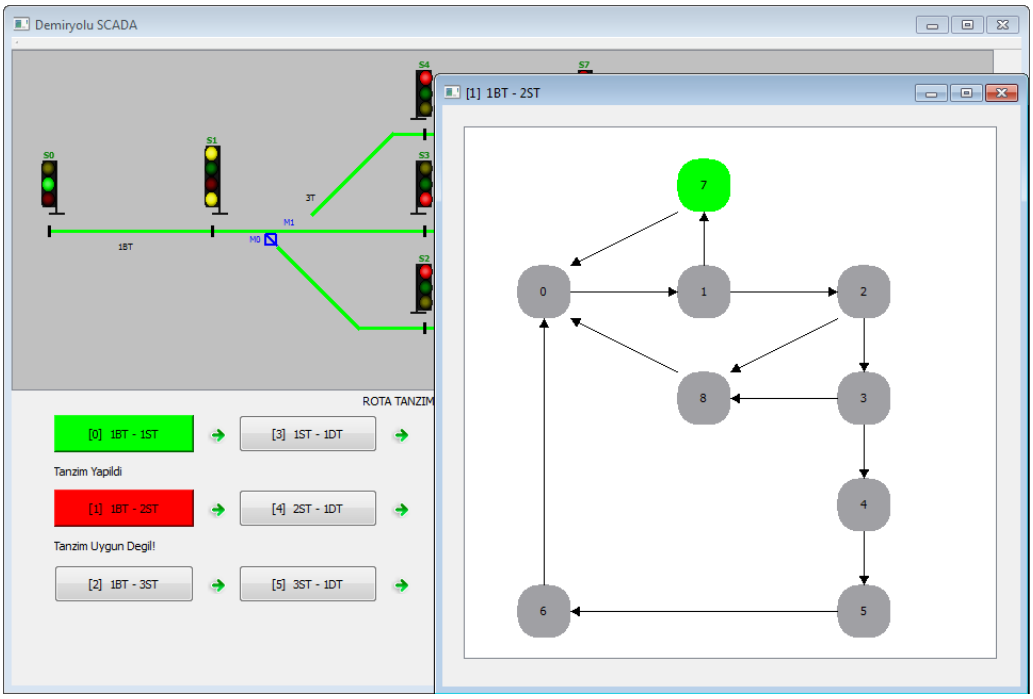

**Şekil 4.2 :** SCADA yazılımı arayüzü.

SCADA yazılımının PLC Simülatörü ile haberleşmesi için TCPModbusClient (TCP Modbus İstemcisi) sınıfı yazılmıştır. Haberleşmeyi test etmek için, internetten indirilen Şekil 4.3'te ekran görüntüsü verilen, Modbus TCP/IP PLC Simülatör

yazılımı kullanılmıştır. Yapılan testler sonucunda simülatördeki giriş çıkış değerleri değiştirildiğinde, SCADA arayüzdeki nesnelerin de buna uygun olarak görünümlerinin değiştiği görülmüştür. SCADA arayüzüne Şekil 4.4'teki durum diyagramı gösterimi ilave edilerek, rota tanziminde simülatöre ait program akışının izlenmesi sağlanmıştır.



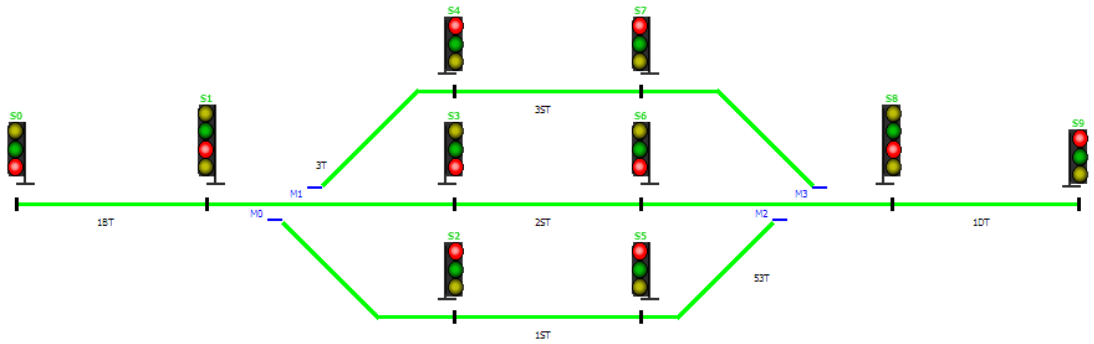

**Şekil 4.3 :** Modbus TCP/IP PLC Simulatör yazılımı.




**Şekil 4.4 :** Rota tanzimi sonlu durum diyagramı arayüzü.

## 4.2 PLC Simülatörü

PLC Simülatörü, PLC'ye ve saha donanımlarına ihtiyaç duymadan gerçek demiryolu sinyalizasyon sistemine benzer şekilde çalışan bir yazılım olarak tasarlanmıştır. Simülatör yazılımında tanımlanan rota sınıfı ile farklı topolojiye sahip rota nesneleri ve demiryolu bölgeleri oluşturulabilmektedir. Simülasyonda Şekil 4.5'teki demiryolu bölgesi kullanılmıştır.




**Şekil 4.5 :** Demiryolu bölgesi.

Öncelikle, her bir rota nesnesine ait ray blokları, sinyal üniteleri, makaslar, çakışan rotalar, tanzimi isteği ve cevaplarına ait bellek adresleri tanımlanmıştır. Bu tanımlama anklaşman tablosu kullanılarak basit bir şekilde gerçekleştirilebilir. Çizelge 4.1'de, rotalara ait kontrol edilen sinyalin, sonrasındaki sinyalin durumuna göre yaktığı renk ve makas konumları verilmiştir. Bu çizelgedeki (K), (Y), (S); sırasıyla kırmızı, sarı ve yeşil renkleri ifade etmektedir. Örneğin 2 nolu 1BT-3ST rotası tanzim edildiğinde, S1 sinyalinin sarı üzeri yeşil (S-Y) yanması için, S7 sinyalinin sarı (S) veya yeşil (Y) yanması gerekmektedir. Bu rotanın tanzim edilebilmesi için M0 makası (0) konumunda, M1 makası (1) konumunda ve kilitli olmalıdır. Çizelge 4.1'deki 12 adet rotaya ait ray blokları, sinyaller ve makaslar PLC Simülatöründe oluşturulan "Rota" türünden nesnelere tanımlanmıştır. Bu sayede her bir rota için, ayrı ayrı kod yazmaya gerek kalmadan, tek bir "Rota" sınıfı ile tüm anklaşman sistemi gerçekleştirilebilmiştir.

Çizelge 4.1'deki anklaşman tablosu oluşturulduktan sonra çakışan rotalar tespit edilerek Çizelge 4.2'deki rota çakışma tablosuna işaretlenmiştir. Örneğin, Rota 0 ile 1, 2, 6, 7, 8 ve 9 nolu rotaların çakıştığı Çizelge 4.2'de görülebilmektedir. Bu çizelge kare matris şeklinde olup simetrik yapıdadır. Çakışma tablosunun simetrik olduğu bilindiği halde, yalnızca oluşturulan çakışma tablosunu doğrulamak amacıyla kare

şeklinde bir çizelge oluşturulmuştur. Oluşturulan bu çizelgeye göre, her bir rota nesnesi için çakışan rotalar tanımlanmıştır.

**Çizelge 4.1 :** Anklaşman tablosu.

| ROTA | | KONTROL EDİLEN SİNYAL | | SONRAKİ SİNYAL LAMBASI | | MAKAS KONUM |
|---|---|---|---|---|---|---|
| 0 | 1BT-1ST | S1 | (S-Y)<br>(S-S) | S5 | (Y) veya (S)<br>(K) | M0 (1) |
| 1 | 1BT-2ST | S1 | (Y)<br>(S) | S6 | (Y) veya (S)<br>(K) | M0 (0)<br>M1 (0) |
| 2 | 1BT-3ST | S1 | (S-Y)<br>(S-S) | S7 | (Y) veya (S)<br>(K) | M0 (0)<br>M1 (1) |
| 3 | 1ST-1DT | S5 | (S) | | | M2 (1) |
| 4 | 2ST-1DT | S6 | (S) | | | |
| 5 | 3ST-1DT | S7 | (S) | | | M3 (1) |
| 6 | 1ST-1BT | S2 | (S) | | | M0 (1) |
| 7 | 2ST-1BT | S3 | (S) | | | |
| 8 | 3ST-1BT | S4 | (S) | | | M1 (1) |
| 9 | 1DT-1ST | S8 | (S-Y)<br>(S-S) | S2 | (Y) veya (S)<br>(K) | M3 (0)<br>M2 (1) |
| 10 | 1DT-2ST | S8 | (Y)<br>(S) | S3 | (Y) veya (S)<br>(K) | M3 (0)<br>M2 (0) |
| 11 | 1DT-3ST | S8 | (S-Y)<br>(S-S) | S4 | (Y) veya (S)<br>(K) | M3(1) |

**Çizelge 4.2 :** Rota çakışma tablosu.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | | X | X | | | | X | X | X | X | | |
| 1 | X | | X | | | | X | X | X | | X | |
| 2 | X | X | | | | | X | X | X | | | X |
| 3 | | | | | | | X | | | X | X | X |
| 4 | | | | | | | | X | | X | X | X |
| 5 | | | | | | | | | X | X | X | X |
| 6 | X | X | X | X | | | | | | | | |
| 7 | X | X | X | | X | | | | | | | |
| 8 | X | X | X | | | X | | | | | | |
| 9 | X | | | X | X | X | | | | | | |
| 10 | | X | | X | X | X | | | | | | |
| 11 | | | X | X | X | X | | | | | | |

Simülatör, sanal giriş çıkışlara ve bellek alanına sahiptir. Kullanıcı istediği girişleri değiştirebilmektedir. Simülatör kullanıcı tarafından verilen girişlere göre çıkış değerleri üretmekte, bellek alanlarındaki değerleri değiştirmektedir. Simülatör, Modbus sunucusu olarak çalışmaktadır. SCADA istemcisi, simülatöre bağlantı kurarak Modbus TCP protokolüyle, simülatörün giriş, çıkış ve bellek alanları değerlerini okuyabilmekte, çıkışlara ve bellek alanlarına değer yazabilmektedir. PLC Simülatörü arayüzünde, dijital giriş ve çıkışlar Şekil 4.6'daki gibidir.

PLC Simülatör yazılımı ve SCADA için PLC giriş/çıkış ve bellek alanlarının işlevleri tanımlanmıştır. Çizelge 4.3'te ray blokları için tanımlanan dijital girişlerin adresleri verilmiştir. Bu girişlerdeki sensörler, ray bloğunda tren varken lojik 0, tren yokken lojik 1 değerini üretmektedir.

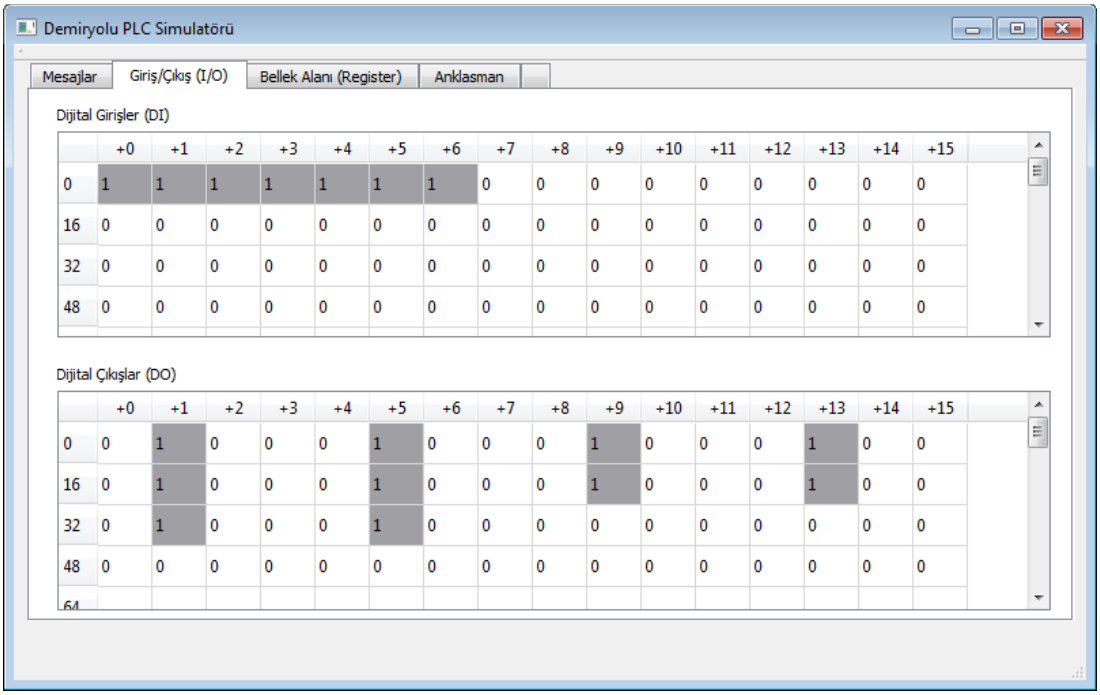
**Çizelge 4.3 :** Ray bloklarına ait sensörler için PLC dijital girişleri.

| DI Adres | Giriş | Tür | Modbus Adresi |
|---|---|---|---|
| 0 | 1BT blok sensörü | Bit | 10001 |
| 1 | 3T blok sensörü | Bit | 10002 |
| 2 | 1ST blok sensörü | Bit | 10003 |
| 3 | 2ST blok sensörü | Bit | 10004 |
| 4 | 3ST blok sensörü | Bit | 10005 |
| 5 | 53T blok sensörü | Bit | 10006 |
| 6 | 1DT blok sensörü | Bit | 10007 |

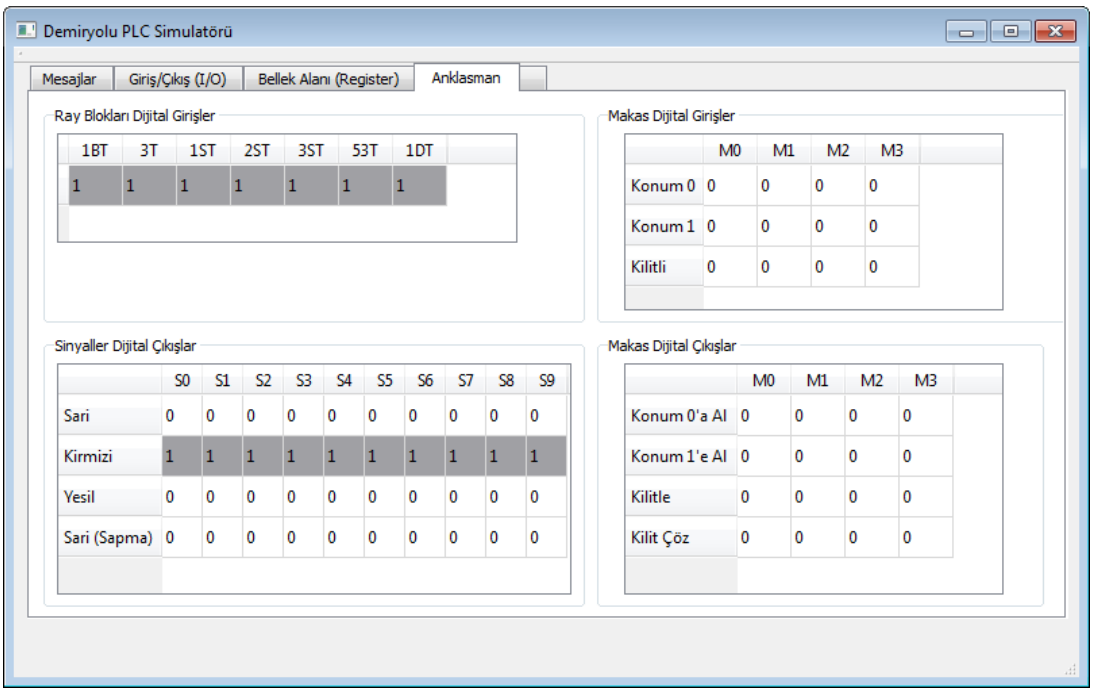
Makaslar için tanımlanan dijital girişler Çizelge 4.4'te verilmiştir. Makaslar için Konum 0, Konum 1 ve Kilitli durumları algılanmaktadır. Konum 0, trenin rota üzerinde sapma yapmadan ilerlemesini sağlayan makas konumudur. Konum 1 ise, trenin rota üzerinde sapma yaparak ilerlemesini sağlayan makas konumudur.

**Çizelge 4.4 :** Makaslar için PLC dijital girişleri.

| DI Adres | Makas | Giriş | Tür | Modbus Adresi |
|---|---|---|---|---|
| 7 | Makas 0 | Konum 0 | Bit | 10008 |
| 8 | | Konum 1 | Bit | 10009 |
| 9 | | Kilitli | Bit | 10010 |
| 10 | Makas 1 | Konum 0 | Bit | 10011 |
| 11 | | Konum 1 | Bit | 10012 |
| 12 | | Kilitli | Bit | 10013 |
| 13 | Makas 2 | Konum 0 | Bit | 10014 |
| 14 | | Konum 1 | Bit | 10015 |
| 15 | | Kilitli | Bit | 10016 |
| 16 | Makas 3 | Konum 0 | Bit | 10017 |
| 17 | | Konum 1 | Bit | 10018 |
| 18 | | Kilitli | Bit | 10019 |




**Şekil 4.6 :** PLC Simülatör yazılımı dijital giriş ve çıkışlar.




**Şekil 4.7 :** PLC Simülatörü saha donanımları dijital giriş ve çıkışlar.

Sinyaller için tanımlanan dijital çıkışlar Çizelge 4.5'te verilmiştir. Burada her bir sinyal için 4 dijital çıkış ayrılmıştır. Bu çıkışlar dörtlü sinyal üniteleri için sırasıyla sarı, kırmızı ve yeşil ve sarı sapma sinyalidir.

Rota tanzim istekleri ve cevapları için PLC bellek alanları Çizelge 4.6'daki gibi tanımlanmıştır. Bu bellek alanlarındaki değerler tanzim sürecinin SCADA tarafından

izlenmesini ve kontrolünü sağlamaktadır. SCADA'dan rota tanzimiyle ilgili gelen istekler, PLC Simülatörünün bu isteklere verdiği cevaplar, rotalara ait zaman aşımı sayaçlarının değerleri ve rotanın tanzim durumuyla ilgili bilgiler bu çizelgede verilmiştir

**Çizelge 4.5 :** Sinyaller için PLC dijital çıkışları.

| DO Adres | Sinyal | Çıkış | Tür | Modbus Adresi |
|---|---|---|---|---|
| 0 | S0 | Sarı | Bit | 1 |
| 1 | | Kırmızı | Bit | 2 |
| 2 | | Yeşil | Bit | 3 |
| 3 | | - | Bit | 4 |
| 4 | S1 | Sarı | Bit | 5 |
| 5 | | Kırmızı | Bit | 6 |
| 6 | | Yeşil | Bit | 7 |
| 7 | | Sarı üst (sapma) | Bit | 8 |
| 8 | S2 | Sarı | Bit | 9 |
| 9 | | Kırmızı | Bit | 10 |
| 10 | | Yeşil | Bit | 11 |
| 11 | | - | Bit | 12 |
| 12 | S3 | Sarı | Bit | 13 |
| 13 | | Kırmızı | Bit | 14 |
| 14 | | Yeşil | Bit | 15 |
| 15 | | - | Bit | 16 |
| 16 | S4 | Sarı | Bit | 17 |
| 17 | | Kırmızı | Bit | 18 |
| 18 | | Yeşil | Bit | 19 |
| 19 | | - | Bit | 20 |
| 20 | S5 | Sarı | Bit | 21 |
| 21 | | Kırmızı | Bit | 22 |
| 22 | | Yeşil | Bit | 23 |
| 23 | | - | Bit | 24 |
| 24 | S6 | Sarı | Bit | 25 |
| 25 | | Kırmızı | Bit | 26 |
| 26 | | Yeşil | Bit | 27 |
| 27 | | - | Bit | 28 |
| 28 | S7 | Sarı | Bit | 29 |
| 29 | | Kırmızı | Bit | 30 |
| 30 | | Yeşil | Bit | 31 |
| 31 | | - | Bit | 32 |
| 32 | S8 | Sarı | Bit | 33 |
| 33 | | Kırmızı | Bit | 34 |
| 34 | | Yeşil | Bit | 35 |
| 35 | | - | Bit | 36 |
| 36 | S9 | Sarı | Bit | 37 |
| 37 | | Kırmızı | Bit | 38 |
| 38 | | Yeşil | Bit | 39 |
| 39 | | - | Bit | 40 |

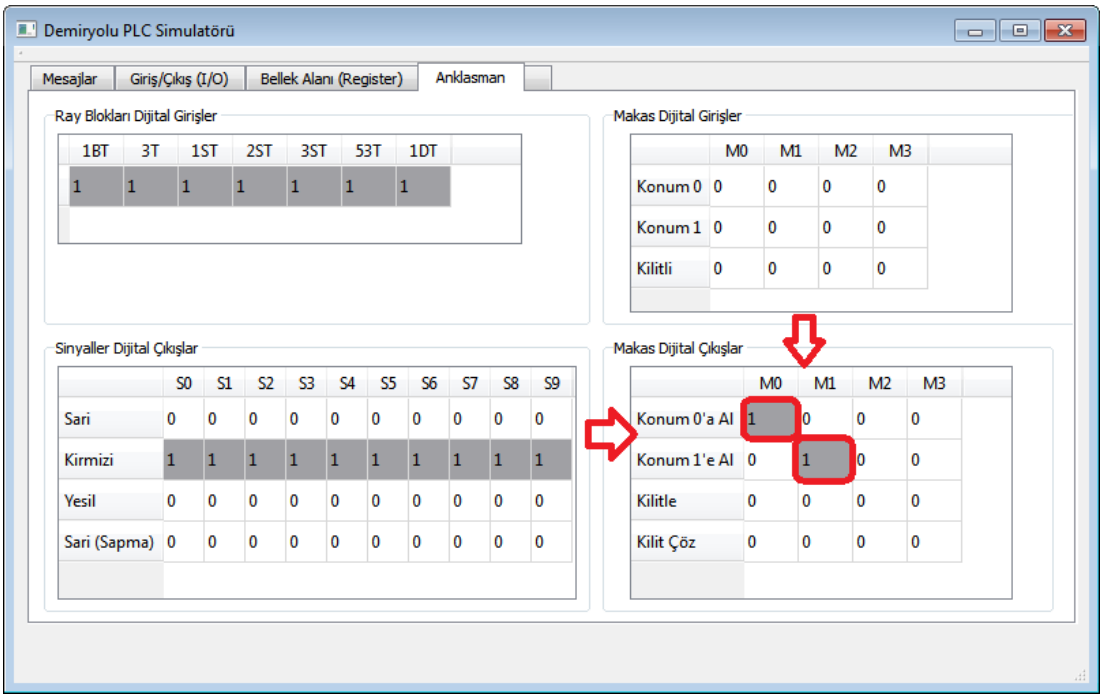
**Çizelge 4.6 :** Rota tanzim isteği ve cevaplarına ait PLC bellek alanları

| Bellek Adresi | Rota | İşlev | Tür | Modbus Adresi |
|---|---|---|---|---|
| 0 | 1BT - 1ST | Tanzim İstekleri (SCADA) | Word | 40001 |
| 1 | 1BT - 2ST | | Word | 40002 |
| 2 | 1BT - 3ST | | Word | 40003 |
| 3 | 1ST - 1DT | | Word | 40004 |
| 4 | 2ST - 1DT | | Word | 40005 |
| 5 | 3ST - 1DT | | Word | 40006 |
| 6 | 1ST - 1BT | | Word | 40007 |
| 7 | 2ST - 1BT | | Word | 40008 |
| 8 | 3ST - 1BT | | Word | 40009 |
| 9 | 1DT - 1ST | | Word | 40010 |
| 10 | 1DT - 2ST | | Word | 40011 |
| 11 | 1DT - 3ST | | Word | 40012 |
| 12 | 1BT - 1ST | Tanzim Cevabı (PLC) | Word | 40013 |
| 13 | 1BT - 2ST | | Word | 40014 |
| 14 | 1BT - 3ST | | Word | 40015 |
| 15 | 1ST - 1DT | | Word | 40016 |
| 16 | 2ST - 1DT | | Word | 40017 |
| 17 | 3ST - 1DT | | Word | 40018 |
| 18 | 1ST - 1BT | | Word | 40019 |
| 19 | 2ST - 1BT | | Word | 40020 |
| 20 | 3ST - 1BT | | Word | 40021 |
| 21 | 1DT - 1ST | | Word | 40022 |
| 22 | 1DT - 2ST | | Word | 40023 |
| 23 | 1DT - 3ST | | Word | 40024 |
| 24 | 1BT - 1ST | Zaman Aşımı Sayacı | Word | 40025 |
| 25 | 1BT - 2ST | | Word | 40026 |
| 26 | 1BT - 3ST | | Word | 40027 |
| 27 | 1ST - 1DT | | Word | 40028 |
| 28 | 2ST - 1DT | | Word | 40029 |
| 29 | 3ST - 1DT | | Word | 40030 |
| 30 | 1ST - 1BT | | Word | 40031 |
| 31 | 2ST - 1BT | | Word | 40032 |
| 32 | 3ST - 1BT | | Word | 40033 |
| 33 | 1DT - 1ST | | Word | 40034 |
| 34 | 1DT - 2ST | | Word | 40035 |
| 35 | 1DT - 3ST | | Word | 40036 |
| 36 | 1BT - 1ST | Rota Durum | Word | 40037 |
| 37 | 1BT - 2ST | | Word | 40038 |
| 38 | 1BT - 3ST | | Word | 40039 |
| 39 | 1ST - 1DT | | Word | 40040 |
| 40 | 2ST - 1DT | | Word | 40041 |
| 41 | 3ST - 1DT | | Word | 40042 |
| 42 | 1ST - 1BT | | Word | 40043 |
| 43 | 2ST - 1BT | | Word | 40044 |
| 44 | 3ST - 1BT | | Word | 40045 |
| 46 | 1DT - 1ST | | Word | 40046 |
| 46 | 1DT - 2ST | | Word | 40047 |
| 47 | 1DT - 3ST | | Word | 40048 |

## 5. YAZILIMLARIN TEST EDİLMESİ

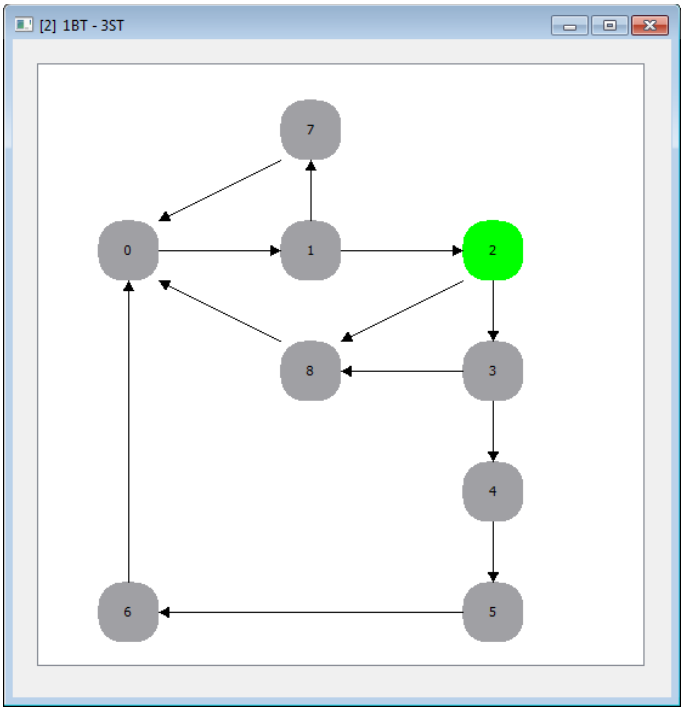
Simülatör ve SCADA yazılımları TCP/IP protokolü ile haberleştiği için aynı bilgisayarda veya aynı ağdaki farklı bilgisayarlarda çalışabilmektedir. Yazılımların her iki durumda da sorunsuz olarak çalıştığı görülmüştür.

Örnek olarak, 1BT-3ST rotasının tanzim edilmesi test edilecektir. Öncelikle, 1BT-3ST rotasındaki 1BT, 3T ve 3ST ray bloklarına ait dijital girişler lojik 1 yapılarak ray bloklarındaki trenler kaldırılır. Daha sonra, 1BT-3ST butonuna tıklanarak rota tanzim isteğinde bulunulur. Rota tanzim isteği PLC Simülatörü tarafından değerlendirilir. Rota tanzim isteği uygunsa, Şekil 5.1'de gösterildiği gibi, simülatör M0 makasını 0 konumuna ve M1 makasını 1 konumuna alacak şekilde dijital çıkışlarını aktif eder. Tanzimin uygun olduğu SCADA'ya bildirilir.



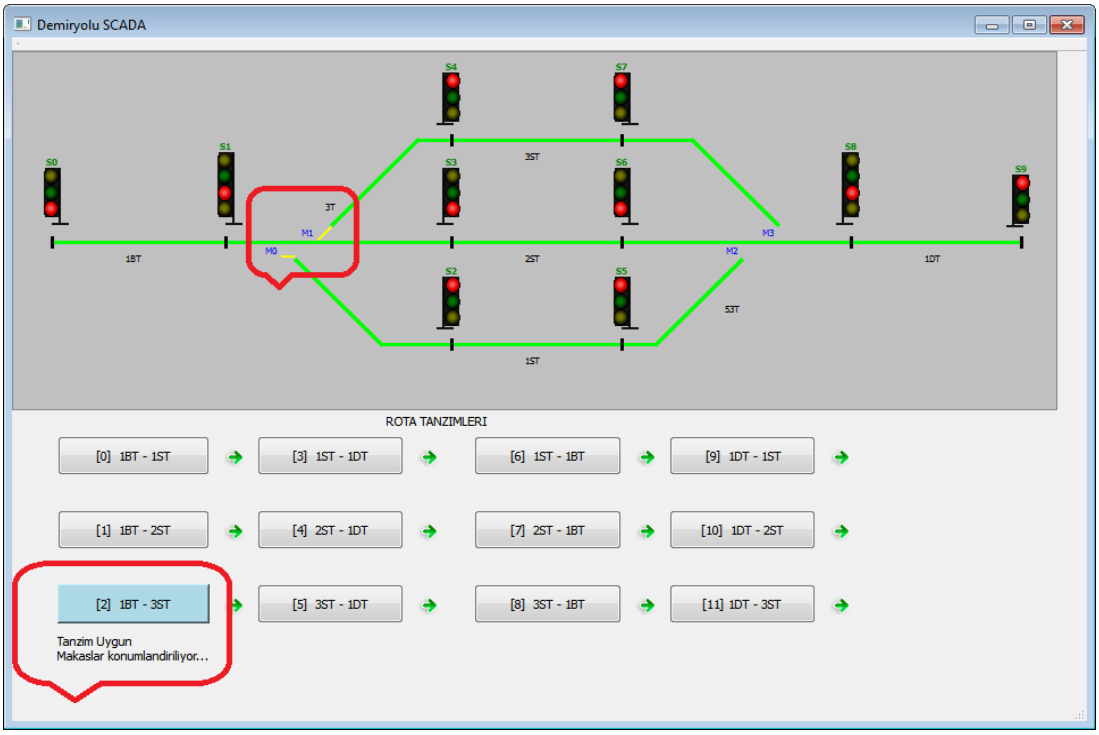

**Şekil 5.1 :** Simülatörde makas dijital çıkışları.

Şekil 5.2'de SCADA yazılımından izlenebilen durum diyagramına ait arayüzde Durum 2'nin aktif olduğu görülmektedir. Durum diyagramı arayüzü Bölüm 3'te tasarlanan durum diyagramı (Şekil 3.4) ile aynıdır. Durum 2'de tanzim uygundur ve makaslar konumlandırılır.



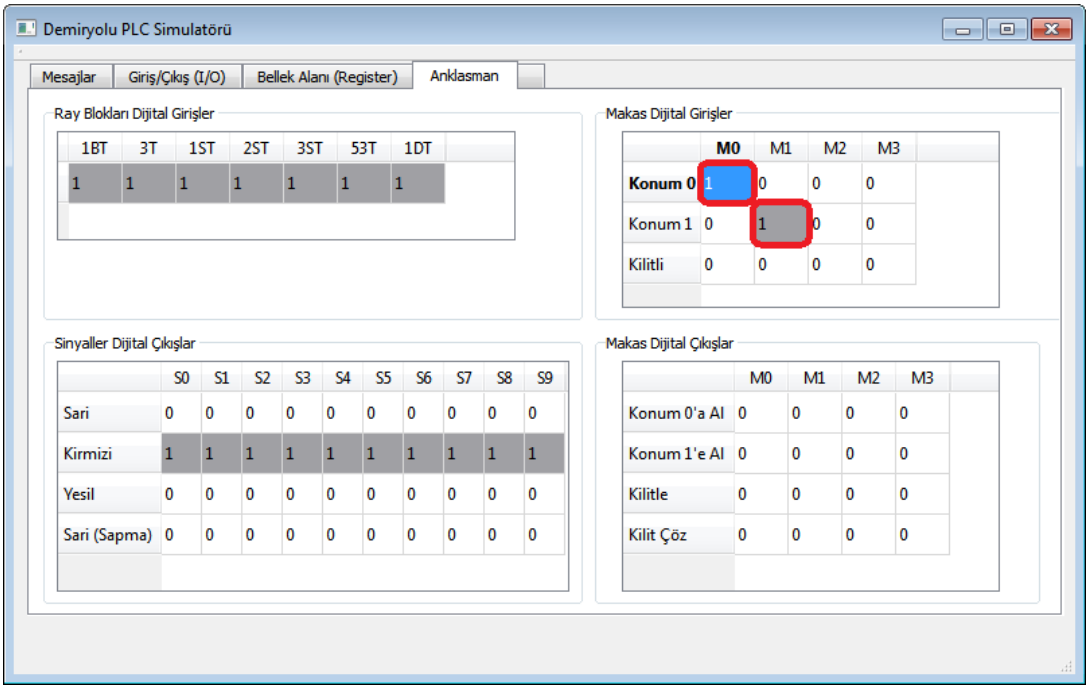

**Şekil 5.2 :** SCADA durum diyagramı (durum 2).

SCADA yazılımından makasların konumlandırılması Şekil 5.3'teki gibi izlenebilmektedir. M0 makası ve M1 makasının alması gereken konum yanıp sönen sarı çizgiyle operatöre bildirilmektedir. Aynı zamanda operatör, tanzimin uygun olduğu ve makasların uygun konuma getirildiği konusunda bilgilendirilir.



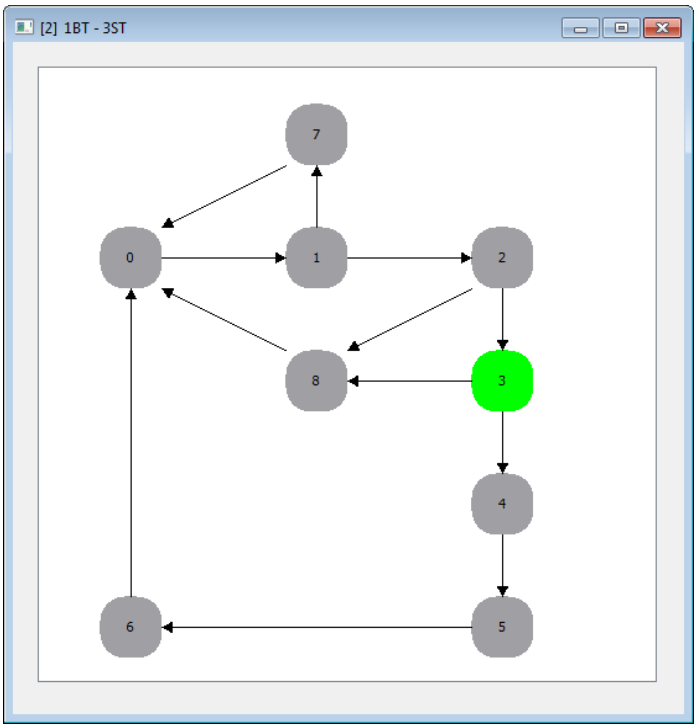

**Şekil 5.3 :** SCADA yazılımında makasların konumlandırılması.

Simülatör yazılımında, ilgili dijital girişlere çift tıklanarak veya Enter tuşuna basılarak M0 makası 0 konumuna, M1 makası 1 konumuna getirilir. Şekil 5.4'te makas konumlarının simülatörden değiştirilmesi görülmektedir.



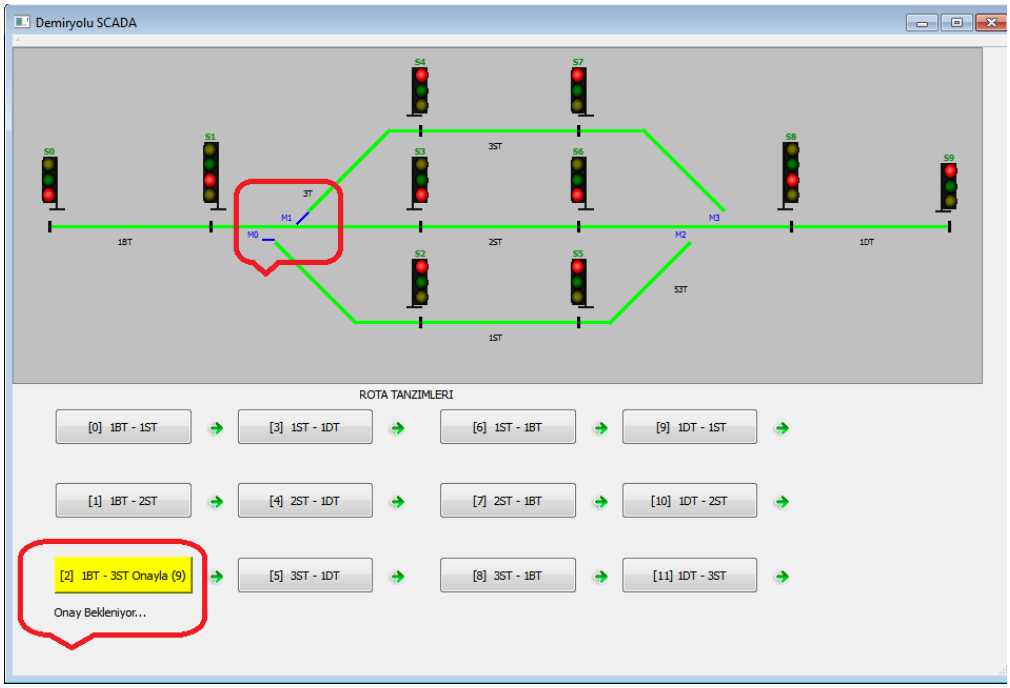

**Şekil 5.4 :** Simülatörde makas konumlarının değiştirilmesi.

Şekil 5.5'teki durum diyagramına ait arayüzde, Durum 3'ün aktif olduğu görülmektedir. Durum 3'te makaslar uygun konumdadır, SCADA'dan onay beklenir.



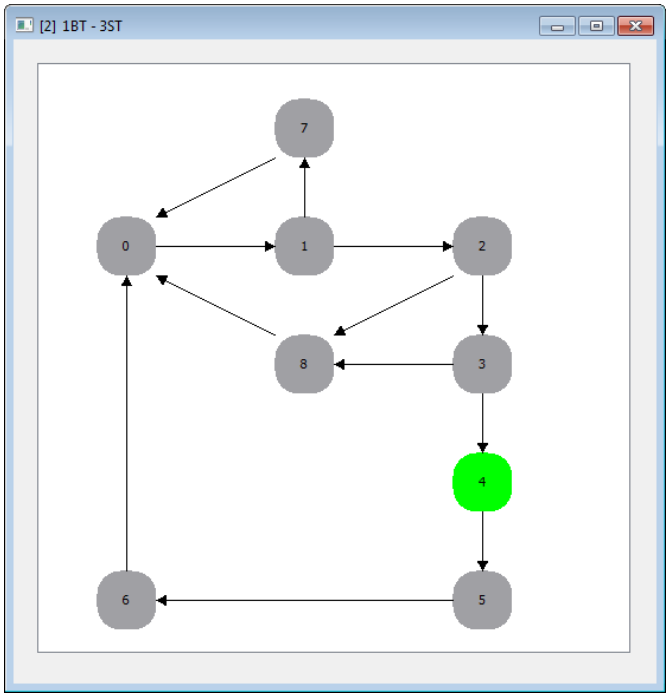

**Şekil 5.5 :** SCADA durum diyagramı (durum 3).

Şekil 5.6'daki SCADA ekranında, makasların uygun konumda olduğu ve kullanıcıdan onay beklendiği görülmektedir.



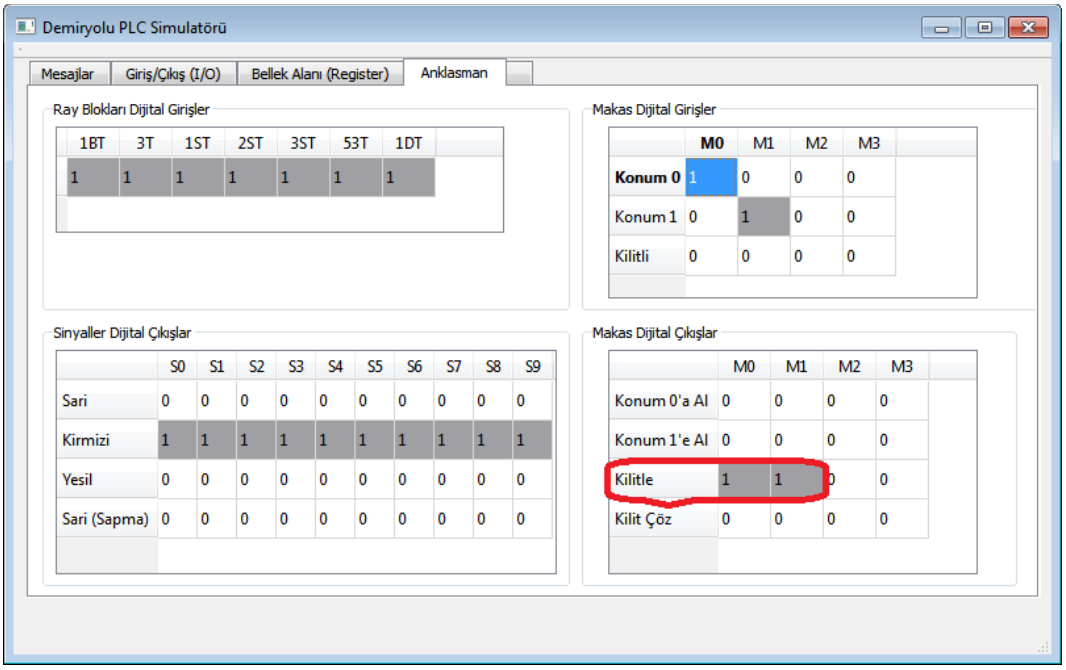

**Şekil 5.6 :** SCADA onayının beklenmesi.

1BT-3ST onayla butonuna basılarak tanzim onaylanır. SCADA tarafından tanzim onayı simülatöre iletilir. Tanzim onaylanınca Şekil 5.7'deki gibi Durum 4'e geçilir. Durum 4'te makaslara kilitleme kumandası gönderilir.



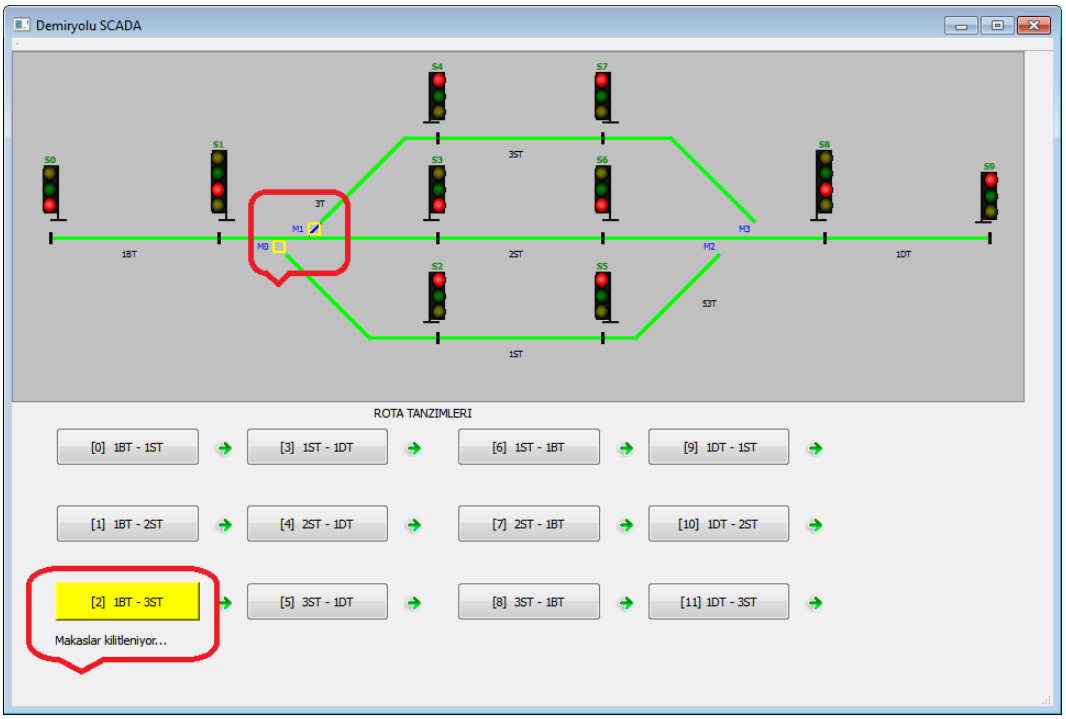

**Şekil 5.7 :** SCADA durum diyagramı (durum 4).

Şekil 5.8'deki simülatör ekranında M0 ve M1 makaslarına kilitleme kumandası verildiği görülmektedir.



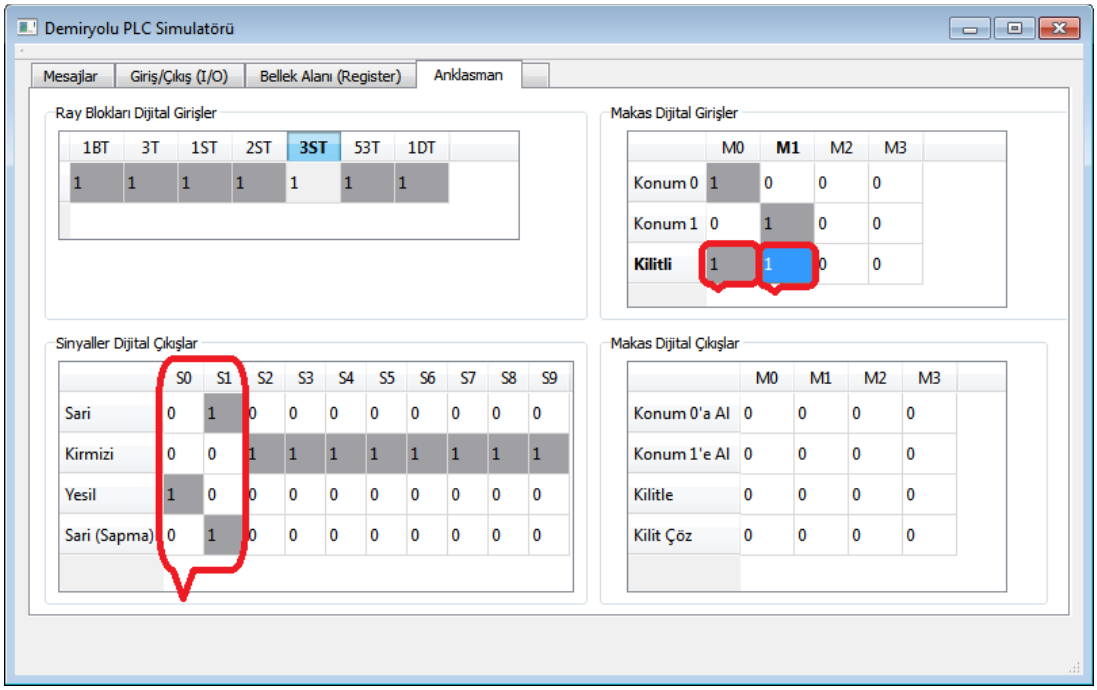

**Şekil 5.8 :** Simülatörde makasların kilitlenmesi.

Şekil 5.9'daki SCADA ekranında M0 ve M1 makaslarının kilitlenme işlemi görülmektedir. Makasların kilitlenme işleminin devam ettiği yanıp sönen kare şeklindeki sarı kutucukla gösterilmektedir.

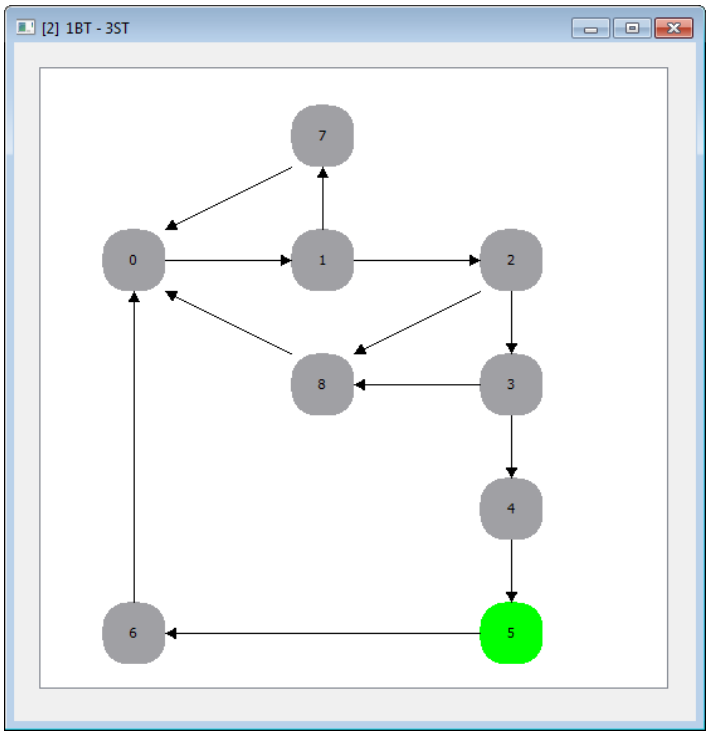



**Şekil 5.9 :** SCADA arayüzünde makasların kilitlenmesi.

Şekil 5.10'da, simülatörde M0 ve M1 makaslarının kilitli dijital girişleri lojik 1 olarak değiştirilmiştir. Bu durumda rota tanzimi yapılır. S0 sinyali yeşil, S1 sinyali sarı üzeri sarı olarak yanar. Durum diyagramında Durum 5, Şekil 5.11'deki gibi aktif olur.

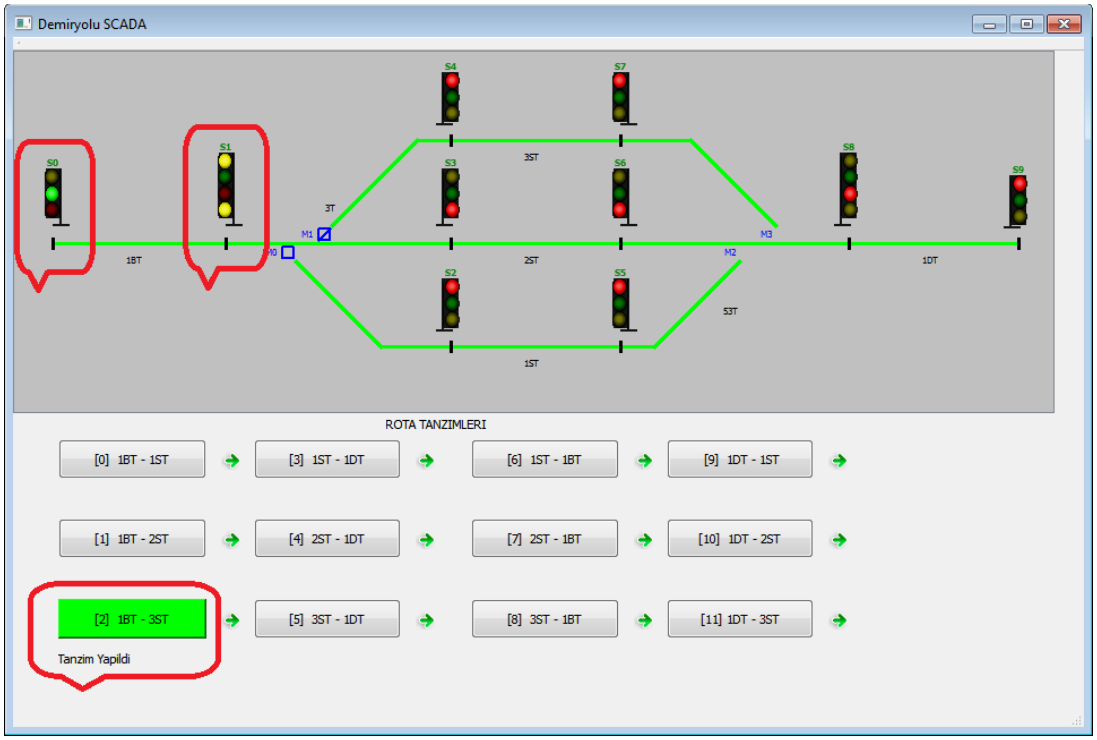



**Şekil 5.10 :** Simülatörde makasların kilitli bilgisinin değiştirilmesi.




**Şekil 5.11 :** SCADA durum diyagramı (durum 5).

Şekil 5.12'deki SCADA arayüzünde tanzimin yapıldığı, M0 makasının 0 konumunda ve kilitli, M1 makasının 1 konumunda ve kilitli olduğu; S0 sinyalinin yeşil, S1 sinyalinin sarı üzeri sarı yandığı görülmektedir. S1 sinyali, bir sonraki sinyal S7 kırmızı olduğu için ve 3T makas ray bloğunda sapma olduğu için sarı üzeri sarı (S-S) yanar. Bir sonraki rota olan 3ST-1DT rotası tanzim edildiğinde, 53T ray bloğu da boşsa S7 sinyali sarı yanar. Bu durumda S3 sinyali sarı üzeri yeşil yanacaktır. Bölüm 4'te, Çizelge 4.1'de verilen anklaşman tablosundan da aynı sonuç elde edilebilmektedir. Buradan, PLC Simülatörünün, hazırlanan anklaşman tablosuna uygun olarak çalıştığı sonucuna varılmıştır.

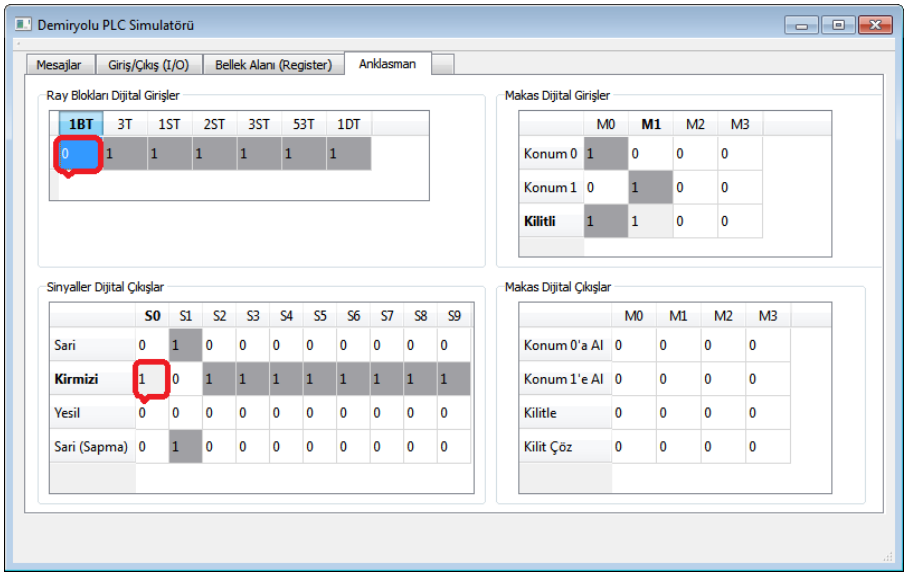



**Şekil 5.12 :** SCADA arayüzünde tanzimin yapılması.

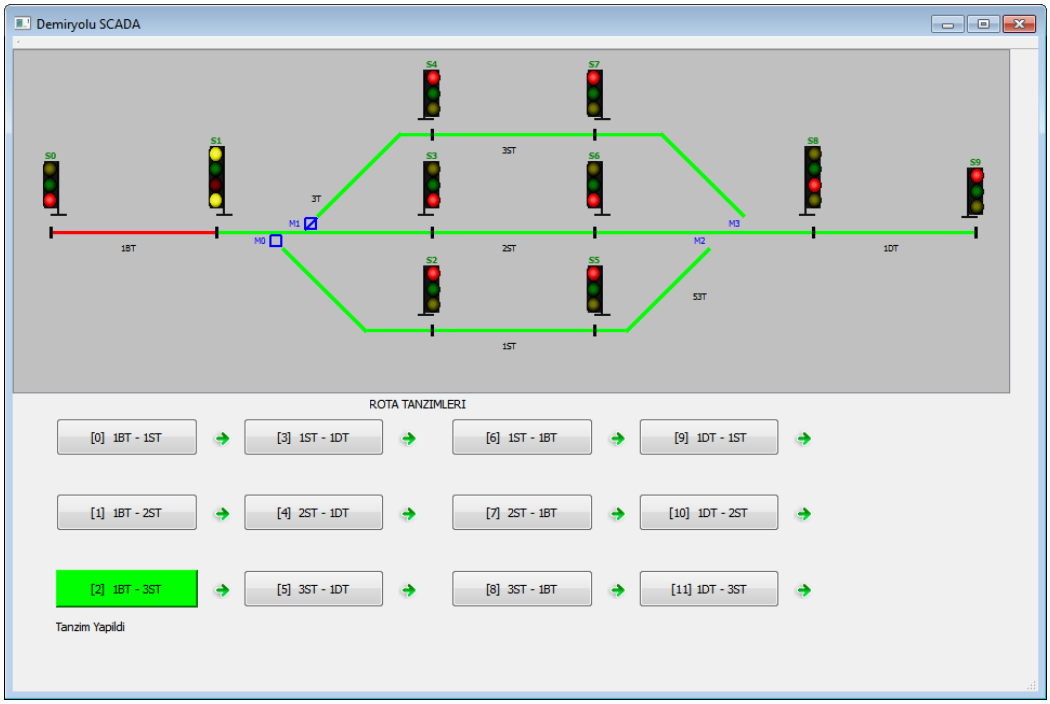
1BT-3ST rotasının tanzimi yapıldıktan sonra tren sırasıyla 1BT, 3T ve 3ST ray bloklarından geçecektir. Tren 1BT'den 3T'ye geçtiği anda hem 1BT hem de 3T ray bloklarında tren algılanacaktır. Trenin son vagonu 1BT'den ayrıldığı zaman yalnızca 3T ray bloğunda tren algılanacaktır. Bunun için PLC Simülatöründe bu ray bloklarına sırayla tren yerleştirilecektir. Öncelikle 1BT ray bloğuna Şekil 5.13'teki gibi ilgili dijital giriş lojik 0 yapılarak tren yerleştirilirse, S0 sinyali kendisinden bir sonraki ray bloğu olan 1BT'de tren bulunduğu için kırmızı yanar.

Şekil 5.14'teki SCADA arayüzünde 1BT ray bloğunun dolu olduğu, buna bağlı olarak S0 sinyalinin kırmızı yandığı görülmektedir. S1 sinyali sarı üzeri sarı yanmaya devam etmektedir.

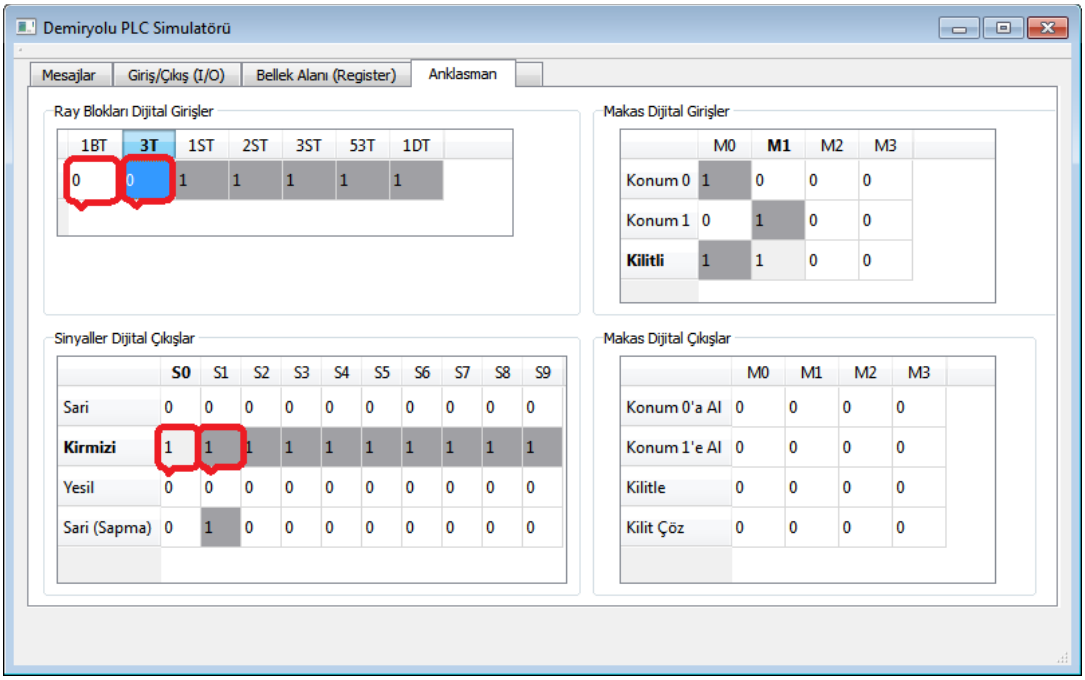
Daha sonra, Şekil 5.15'teki gibi 3T ray bloğuna da tren yerleştirilir. Bu durumda trenin bir kısmı 3T, diğer kısmı 1BT ray bloğunda bulunmaktadır. Bu yüzden S0 sinyali kırmızı, S1 sinyali sarı üzeri kırmızı yanar.




**Şekil 5.13 :** Simülatörde 1BT ray bloğuna tren yerleştirilmesi.

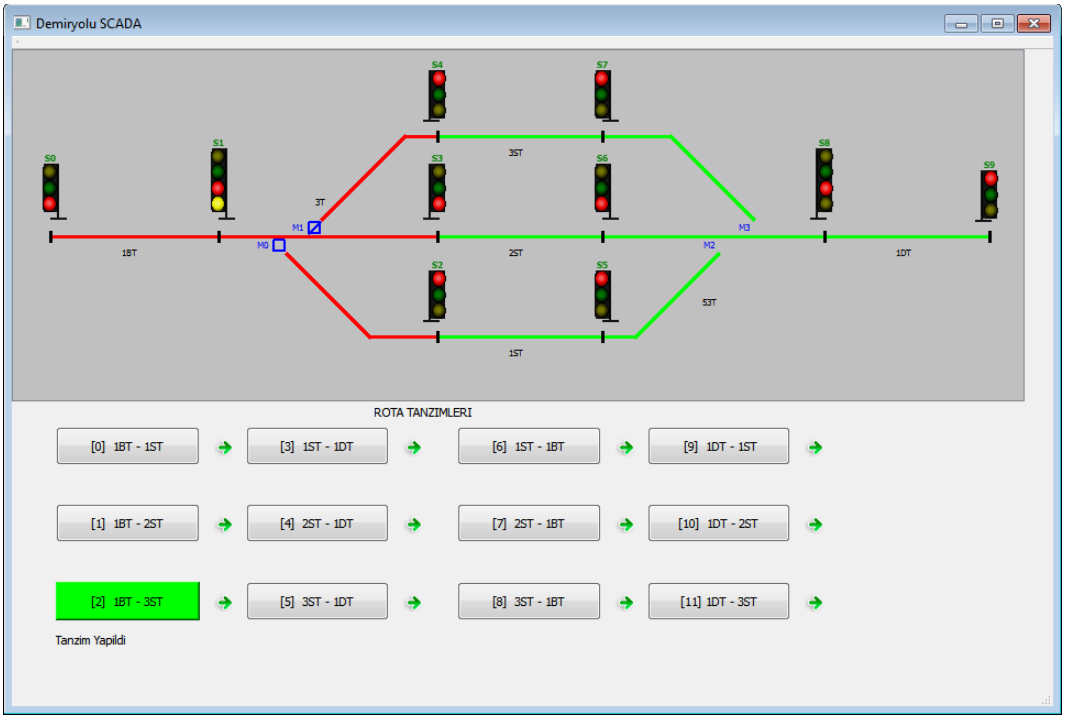



**Şekil 5.14 :** 1BT ray bloğunda tren varken SCADA arayüzü.



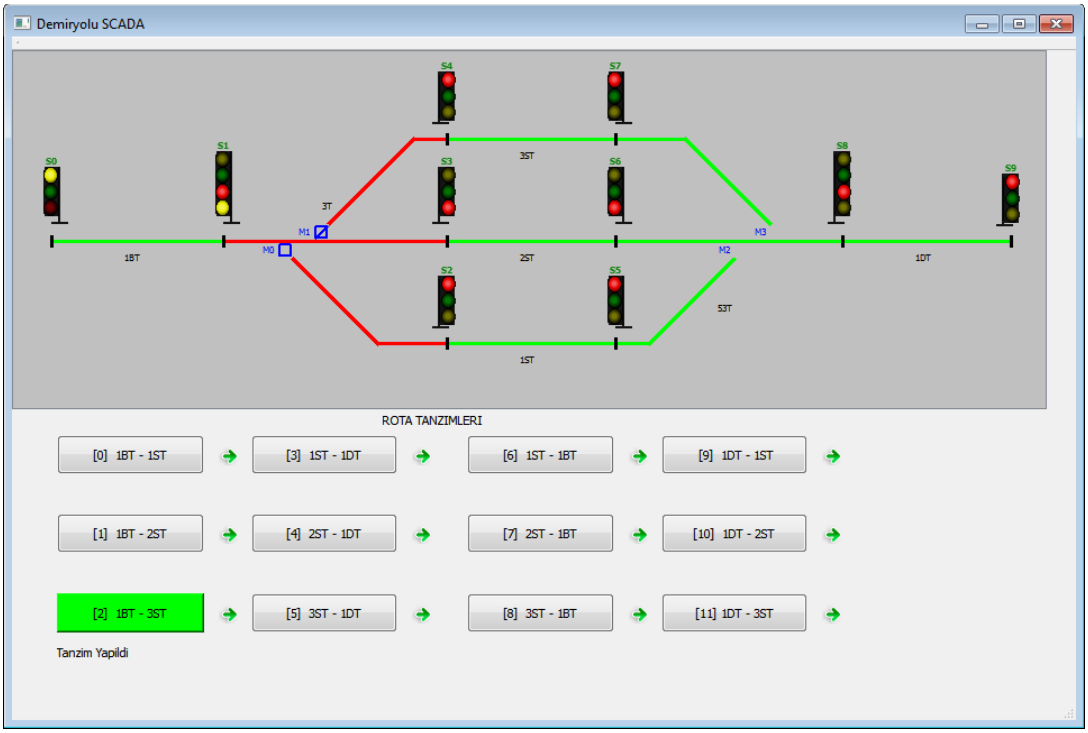

**Şekil 5.15 :** Simülatörde 1BT ve 3T ray bloğuna tren yerleştirilmesi.

Şekil 5.16'daki SCADA arayüzünde 1BT ve 3T dolu olduğu, S0 sinyalinin kırmızı S1 sinyalinin sarı üzeri kırmızı yandığı görülür.



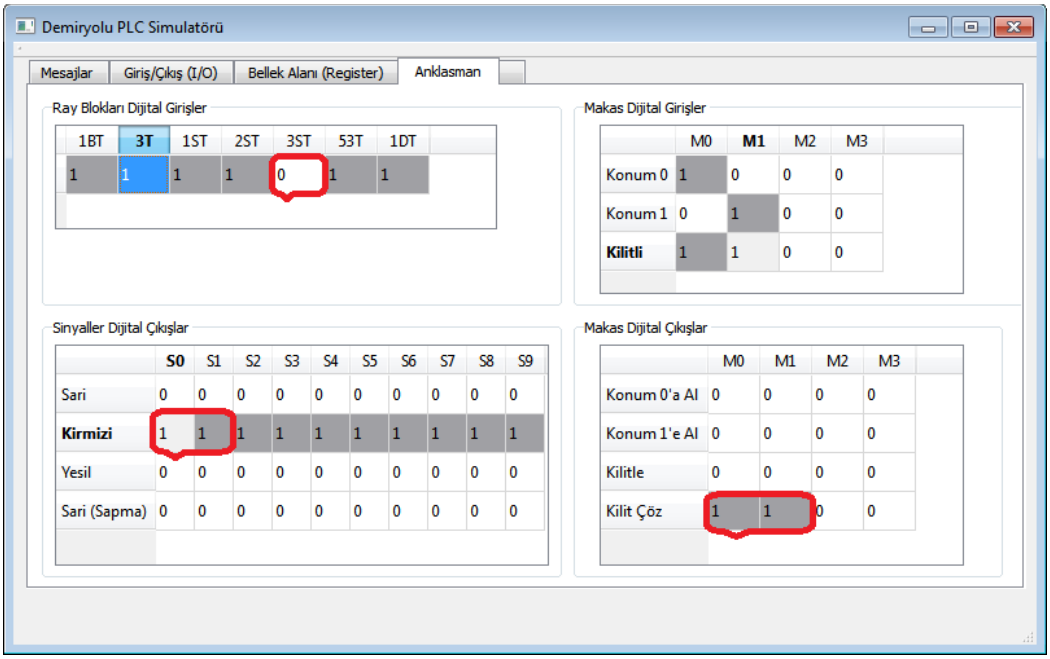

**Şekil 5.16 :** Tren 1BT ve 3T ray bloğundayken SCADA arayüzü.

Şekil 5.17'de trenin tamamı 3T ray bloğunda bulunmaktadır. S1 sinyali sarı üzeri kırmızı, S0 sinyali sarı yanmaktadır.




**Şekil 5.17 :** Tren 3T ray bloğundayken SCADA arayüzü.

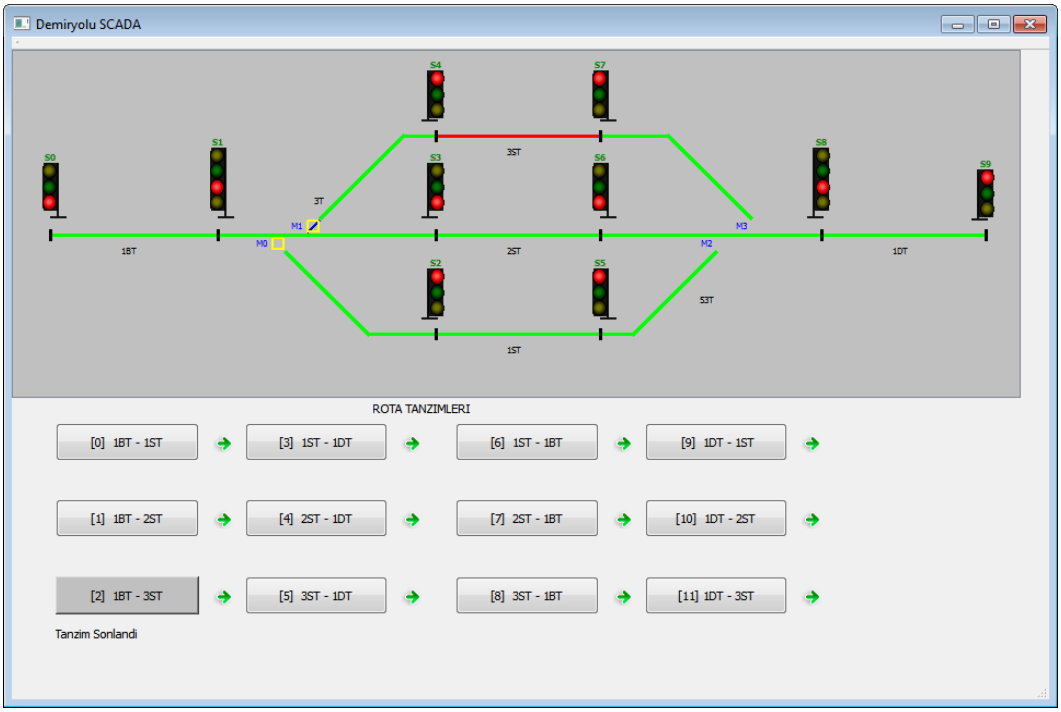
Şekil 5.18'deki gibi tren 3ST ray bloğuna yerleştirilir. Tren son ray bölgesine geldiği için tanzim sonlandırılır. Bu durumda M0 ve M1 makaslarının kilitlerinin çözülmesi için kumanda verilmektedir.



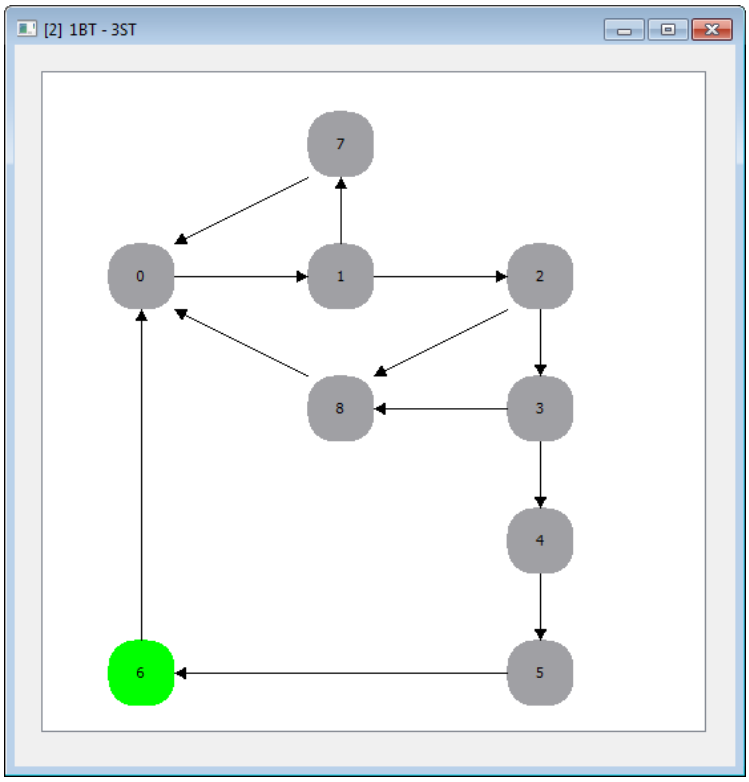

**Şekil 5.18 :** Simülatörde trenin 3ST ray bloğuna yerleştirilmesi.

Tren 3ST ray bloğundayken SCADA arayüzünden (Şekil 5.19) tanzimin sonlandığı ve M0 ve M1 makaslarına ait kilitlerin çözülmekte olduğu görülür. Şekil 5.20'deki

durum diyagramında Durum 6'nın aktif olduğu görülür. Durum 6'da makas kilitlerinin çözülmesi için beklenir. Makas kilitlerinin çözüldüğü bilgisi alınınca başlangıç durumuna (Durum 0) geçilir.

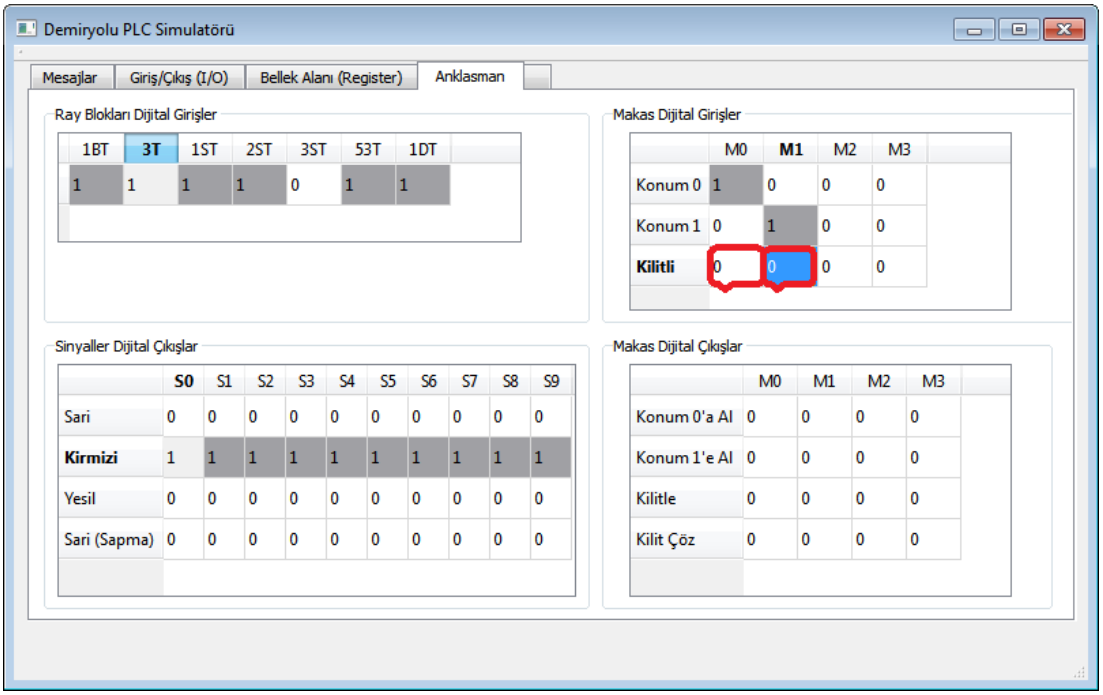



**Şekil 5.19 :** Tren 3ST ray bloğundayken SCADA arayüzü.

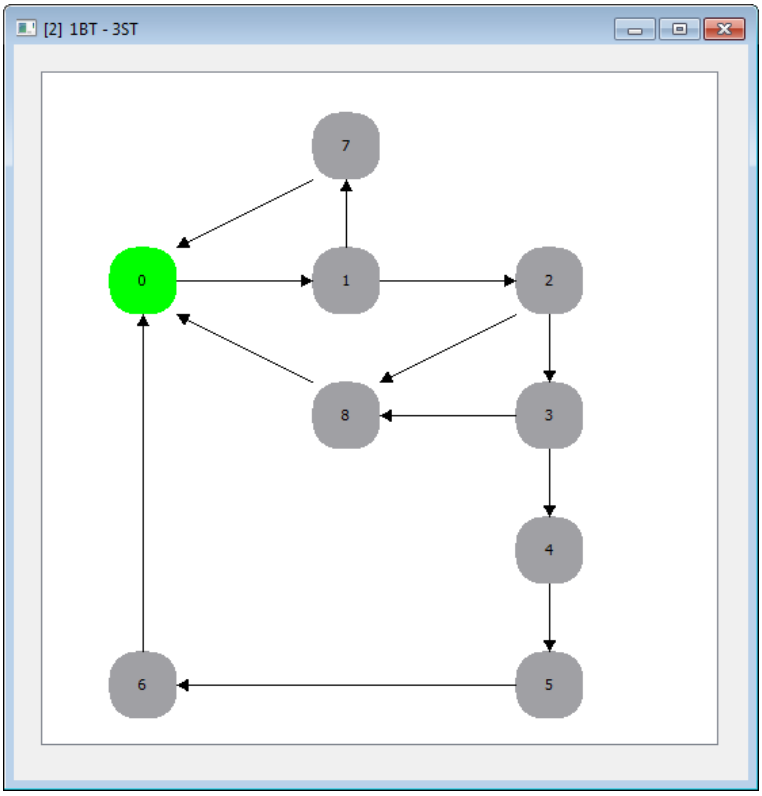



**Şekil 5.20 :** SCADA durum diyagramı (durum 6).

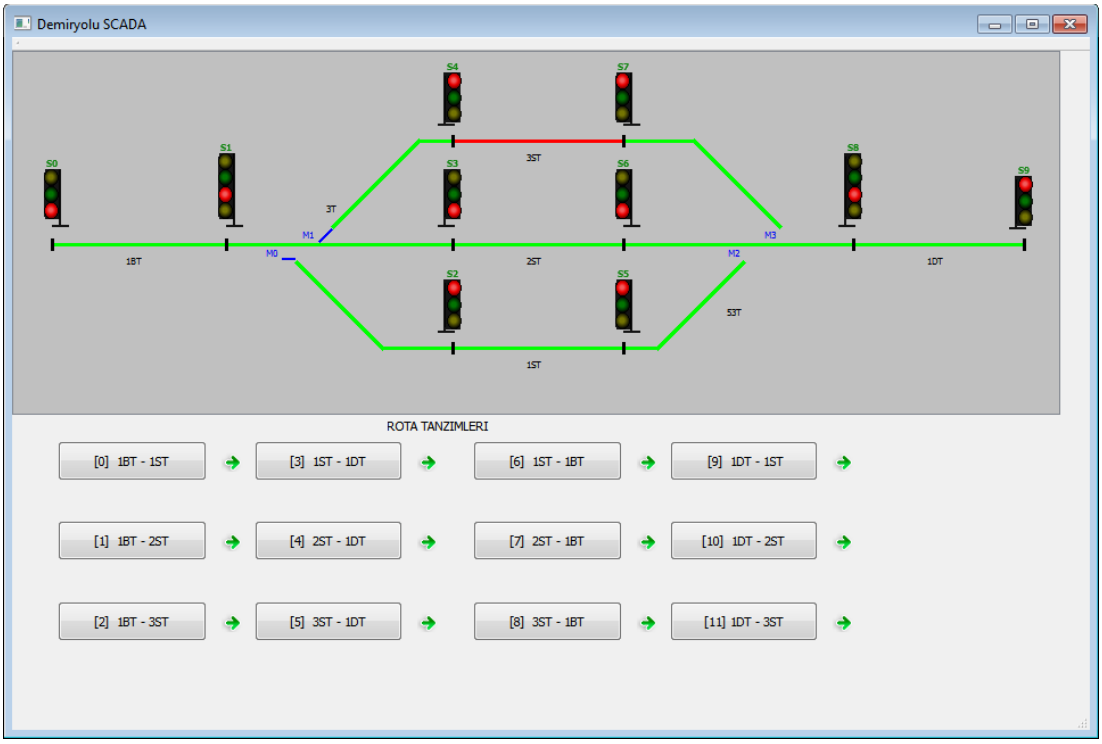
Şekil 5.21'deki gibi M0 ve M1 makas kilitlerine ait dijital girişler lojik 0 yapılır. Şekil 5.22'deki durum diyagramında Durum 0 (başlangıç durumu) aktif olur. Şekil 5.23'te ise Durum 0 için SCADA arayüzü gösterilmiştir.




**Şekil 5.21 :** Simülatörde makas kilitlerinin çözülmesi.

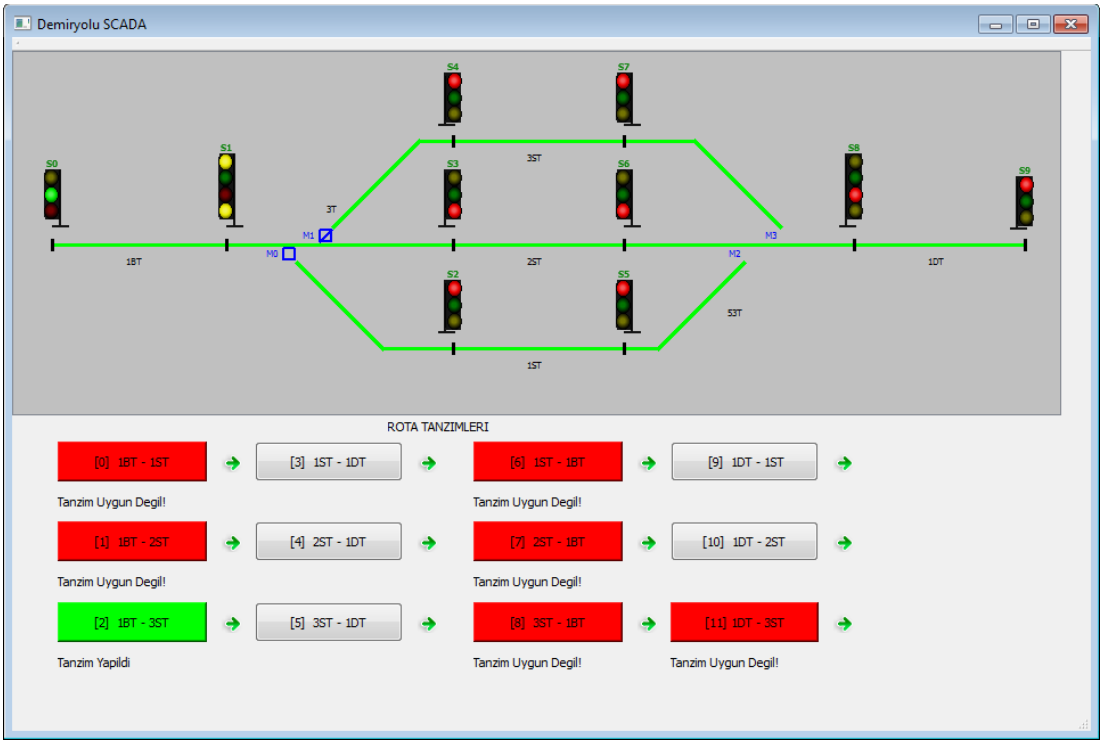



**Şekil 5.22 :** SCADA durum diyagramı (durum 0).




**Şekil 5.23 :** SCADA arayüzü (durum 0).

1BT-3ST rotası tanzim edildikten sonra bu rota ile çakışan rotalara tanzim isteğinde bulunulduğunda Şekil 5.24'teki gibi "tanzim uygun değil" şeklinde kullanıcıya bilgi verilmektedir. İlgili rota tanzim butonları kırmızı renk almaktadır.




**Şekil 5.24 :** 1BT-3ST rotasıyla çakışan rotalar.

# 6. SONUÇ VE ÖNERİLER

Bu çalışmada, demiryolu sinyalizasyon sistemi PLC Simülatörü ve SCADA yazılımı tasarlanmış ve geliştirilmiştir. Nesne Yönelimli Programlama yöntemi kullanılarak demiryolu sinyalizasyon sistemlerinin simülasyonu, izleme ve kontrolü için bir kütüphane oluşturulmuştur. Demiryolu sinyalizasyon sisteminin bileşenlerine ait modeller oluşturulmuştur. Bu sayede, farklı demiryolu bölgeleri için de simülasyon ortamı oluşturulabilecektir. Demiryolu bölgesine ait çizim, anklaşman tablosu ve rota çakışma tablosu yardımıyla bu simülasyon ortamı oluşturulabilecektir.

Geliştirilen Simülatör ve SCADA yazılımları farklı senaryolar göz önüne alınarak test edilmiştir. Yapılan testler sonucu, yazılımların belirlenen işlevleri yerine getirdiği görülmüştür. Geliştirilen yazılımlar, test ve eğitim amacıyla kullanılabilecektir.

Geliştirilen yazılımlar ile:

- Ray bloklarındaki tren varlık yokluk bilgisi izlenebilmektedir.
- Rotaların tanzim durumları izlenebilmektedir.
- Makasların konumları, kilit durumları, sinyallerin durumları izlenebilmektedir.
- Operatör, rota tanzim isteğinde bulunabilmektedir.
- Rota tanzim isteği anklaşman sistemine iletilmektedir.
- Rotanın uygun olup olmadığı bilgisi anklaşman sistemi PLC Simülatöründen alınarak operatör bilgilendirilmektedir.
- Tanzim uygunsa makaslar uygun konuma getirilmekte ve operatörden onay beklenmektedir.
- Operatörden tanzim onayı gelince makaslar kilitlenmekte, sinyal bildirimi yapılarak rota tanzimi yapılmaktadır.
- Tren son ray bloğuna tamamen girdiğinde ise tanzim sonlandırılmaktadır.

- Makaslar belirlenen sürede uygun konuma geçmezse veya tanzim onayı alınmazsa zaman aşımı konusunda operatör bilgilendirilir.

Çalışmanın bundan sonraki kısmında, demiryolu bölgesi, anklaşman tablosu ve rota çakışmalarının kullanıcı tarafından konfigüre edilebilir olması özelliği ve bu konfigürasyonların dosyaya yazılması ve dosyadan okunması özelliği ilave edilebilir. Bu sayede kullanıcı istediği demiryolu bölgesine ait simülasyon ve SCADA ortamını, konfigürasyon arayüzüyle oluşturabilecektir. Trenlerin hareketlerinin otomatik simülasyonu da ilave edilebilecek özellikler arasında sayılabilir. Operatörün yaptığı işlemlerin kayıt altına alınması vb. diğer birçok özellik yazılımlara ilave edilebilecektir.

## KAYNAKLAR


[1] **Durmuş, M. S., Yıldırım U., Söylemez M. T.** (2011). Application of functional safety on railways Part I: modelling & design. *Proceedings of 2011 8th Asian Control Conference (ASCC),* Kaohsiung, Taiwan.

[2] **Yıldırım, U., Durmuş, M. S., Söylemez M. T.** (2011). Application of functional safety on railways part II: software development. *Proceedings of 2011 8th Asian Control Conference (ASCC),* Kaohsiung, Taiwan.

[3] **Eriş, O., Kurtulan, S.** (2011). Application of functional safety on railways part III: development of a main controller. *Proceedings of 2011 8th Asian Control Conference (ASCC),* Kaohsiung, Taiwan.

[4] **Durmuş, M. S., Yıldırım, U., Söylemez M. T.** (2011). Demiryolu Sinyalizasyon Tasarımında Fonksiyonel Güvenlik ve Ayrık Olay Sistem Yaklaşımı. *Elektrikli Ulaşım Sistemleri Sempozyumu ve Sergisi (EUSIS)* Bursa-Eskişehir, Turkey.

[5] **Eriş O., Mutlu İ.** (2010). Design of signal control structures using formal methods for railway interlocking systems. *11th International Conference on Control* Automation *Robotics & Vision (ICARCV),* Singapore.

[6] **Saygın, S., Yakın, İ., Durmuş, M. S., Söylemez, M. T.** (2009). Petri ağları ile demiryolu makas bölgelerinin anklaşman ve sinyalizasyonu tasarımı. *Otomatik Kontrol Ulusal Toplantısı, TOK'09.* İstanbul.

[7] **Durmuş, M. S., Söylemez M. T.** (2009). Railway Signalization and Interlocking Design via Automation Petri Nets. *Proceedings of 7th Asian Control Conference,* Hong Kong, China, 27-29 Ağustos.

[8] **Yıldırım, U., Durmuş, M. S., Söylemez M. T.** (2010). Fail-Safe Signalization and Interlocking Design for a Railway Yard: an Automation Petri Net Approach, *Proceedings of 7th International Symposium on Intelligent and Manufacturing Systems (IMS2010),* Sarajevo, Bosnia.

[9] **Durmuş, M. S., Söylemez M. T.** (2009). Automation Petri Net Based Railway Interlocking and Signalization Design. *International Symposium on Innovations in Intelligent Systems and Applications, INISTA'09,* Karadeniz Technical University, Trabzon, Turkey.

[10] **Akın, K., Durmuş, M. S., Söylemez, M. T.** (2010). Demiryolu sinyalizasyon sistemi bileşenlerinin otomasyon petri ağları ile modellenmesi ve PLC ile gerçeklenmesi. *Otomatik Kontrol Ulusal Toplantısı, TOK'10,* Gebze.

[11] **Kurtulan, S.** (2010). Endüstriyel Otomasyon Sistemleri dersi proje notları. İTÜ, Kontrol ve Otomasyon Müh.

[12] **Kurtulan, S.** (2010). PLC ile Endüstriyel Otomasyon, Birsen Yay., İstanbul.

[13] **Ryland H.** (2005). Operations – the value of training simulators [railway signaller training. *IEE Seminar on Safety Assurance*.

[14] **Mutlu, I., Ovatman, T., Soylemez, M. T., Sumer, L. G.** (2011). A new test environment for PLC based interlocking systems. *International Conference on Transportation, Mechanical, and Electrical Engineering (TMEE)*, 686-690. Changchun, China

[15] **KunMing L., DaMing J.** (2012). PLC used in the train control simülation system. *IEEE International Conference on Computer Science and Automation Engineering (CSAE),* Zhangjiajie, China.

[16] **Hulea, M., Avram, C., Letia T., Muresan D., Radu S.** (2008). Distributed Real-Time Railway Simülator. *Proceedings of the High Performance Computing & Simülation Conference, ECMS.* Nicosia, Cyprus.

[17] **Pataricza A., Majzik I., Huszerl G., Várnai G**. (2003). UML-based design and formal analysis of a safety-critical railway control software module. *Formal Methods for Railway Operation and Control Systems*, Budapest.

[18] **Abbaneo C., Flammini F., Lazzaro A., Marmo P., Mazzocca N., Sanseviero A.** (2006). UML Based Reverse Engineering for the Railway Control Logics. *Proceedings of the International Conference on Dependability of Computer Systems (DEPCOS-RELCOMEX'06),* 3-10. Szklarska Poreba.

[19] **Hon Y. M., Maik K.** (2006). Simülation and Verification of UML-based Railway Interlocking Designs. *Automatic Verification of Critical Systems, AVOCS'06.*

[20] **BordBar B., Giacomini L., Holding D. J.** (2000). UML and Petri Nets for design and analysis of distributed systems. *Proceedings of the IEEE International Conference on Control Applications.* Anchorage, Alaska, USA.

[21] **Baresi L., Pezze M.** (2001). Improving UML with Petri Nets. *Electronic Notes in Theoretical Computer Science,* 1-13.

[22] **Modbus Application Protocol Specification v1.1b**. (2006). *Modbus.org.* Alındığı tarih: 05.11.2012, adres: http://www.modbus.org/docs/ Modbus_Application_Protocol_V1_1b.pdf

[23] **Durmuş, M. S., Yıldırım, U., Söylemez M. T.** (2012). Automatic generation of Petri Net supervisors for interlocking design. *Australian Control Conference,* University of New South Wales, Sydney, Australia.

[24] **Durmuş, M. S., Yıldırım, U., Söylemez M. T.** (2012). Safety-Critical Interlocking Software Development Process for Fixed-Block Signalization Systems. *13th IFAC Symposium on Control in Transportation Systems (CTS),* Sofia, Bulgaria.

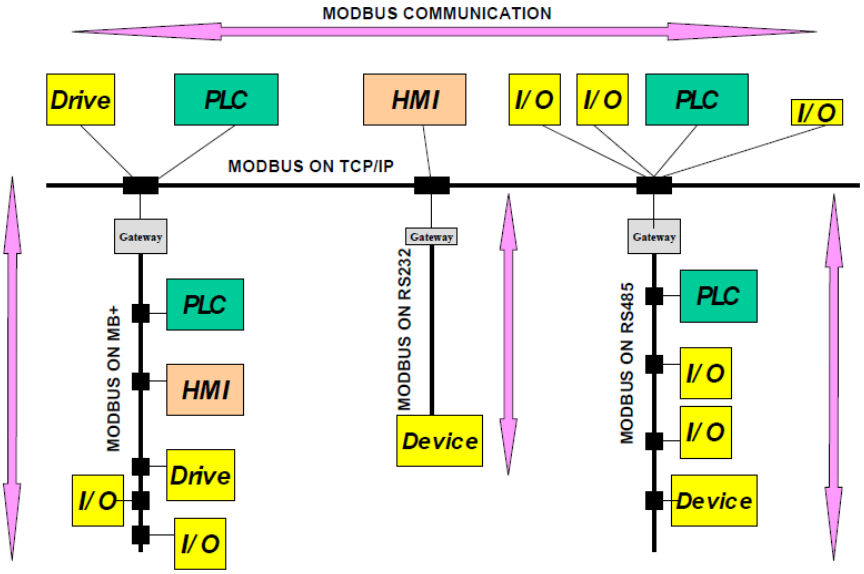
[25] **Url-1** *<http://www.automation.siemens.com/mcms/human-machine-interface/ en/visualization-software/simatic-wincc-open-architecture/news-overview/pages/wincc-oa-newsletter-2012-4-news01.aspx>,* alındığı tarih: 10.12.2012.

**EKLER**

**EK A:** Modbus Protokolü

**EK A**

Modbus, farklı tür veri yolu ve ağlarda sunucu/istemci haberleşmesini sağlayan bir uygulama katmanı haberleşme protokolüdür. 1979'dan itibaren kullanılmaya başlanan Modbus, bugün milyonlarca otomasyon cihazının haberleşmesini sağlamaktadır. Veri iletim ortamından bağımsızdır. RS-232, RS-485 ve Ethernet ortamında çalışabilmektedir. Modbus TCP, TCP arayüzüyle Ethernet'te çalışan bir RTU (Remote Terminal Unit) protokolüdür [22]. Şekil A.1'de RS-232, RS-485 ve TCP/IP'nin birlikte kullanıldığı örnek Modbus ağ mimarisi verilmiştir.




**Şekil A.1 :** Modbus örnek ağ mimarisi [22].

Çizelge A.1'de Modbus TCP veri paketi yapısı görülmektedir. Çizelge A.2'de ise sıklıkla kullanılan Modbus fonksiyon kodları verilmiştir.

**Çizelge A.1 :** Modbus TCP veri paketi yapısı [22].

| Modbus Uygulama Protokolü Başlığı (7 Byte) | | | | Protokol Verisi | |
|---|---|---|---|---|---|
| Işlem Kimlik No | Protokol Kimlik No | Paket Boyutu | Birim Kimlik No | Fonksiyon Kodu | Veri |
| 2 Byte | 2 Byte | 2 Byte | 1 Byte | 1Byte | Değişken |

**Çizelge A.2 :** Modbus TCP'de sıklıkla kullanılan fonksiyon kodları [22].

| Veri Erişimi | Modbus Fonksiyonu | Fonksiyon Kodu | Fonksiyon Kodu (Hex) |
|---|---|---|---|
| Bit Erişimi | Dijital Çıkışları Oku | 01 | 01 |
| | Dijital Girişleri Oku | 02 | 02 |
| | Dijital Çıkışa Yaz | 05 | 05 |
| | Çok Sayıda Dijital Çıkışa Yaz | 15 | 0F |
| 16 Bit (Word) Erişimi | Giriş Bellek Alanını Oku | 04 | 04 |
| | Bellek Alanlarını Oku | 03 | 03 |
| | Bellek Alanlarına Yaz | 16 | 10 |

**ÖZGEÇMİŞ**

**Ad Soyad:** Fatih MECİTOĞLU

**Doğum Yeri ve Tarihi:** İstanbul, 22 Eylül 1985

**Adres:** Canan Sk. No: 12 Yeşilpınar/İstanbul

**E-Posta:** fatihm85@gmail.com

**Lisans:** İstanbul Teknik Üniversitesi, Elektrik-Elektronik Fakültesi, Elektrik Mühendisliği

**Mesleki Deneyim ve Ödüller:**

2008-2009 İSBAK A.Ş. Proje Uygulama Mühendisi

2009-2011 İSBAK A.Ş. Ar-Ge Yazılım Mühendisi

2012- İSBAK A.Ş. Ar-Ge Yazılım Uzmanı