

**İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ**

**RAYLI ULAŞIM SİNYALİZASYON SİSTEMLERİ İÇİN  
OTOMATİK ANKLAŞMAN ALGORİTMASI VE KODU  
ÜRETME YÖNTEMİ**

**YÜKSEK LİSANS TEZİ  
Serhat TÜRK**

**Anabilim Dalı : Kontrol ve Otomasyon Mühendisliği**

**Programı : Kontrol ve Otomasyon Mühendisliği**

**HAZİRAN 2010**



**İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ**

**RAYLI ULAŞIM SİNYALİZASYON SİSTEMLERİ İÇİN  
OTOMATİK ANKLAŞMAN ALGORİTMASI VE KODU  
ÜRETME YÖNTEMİ**

**YÜKSEK LİSANS TEZİ  
Serhat TÜRK  
(504081125)**

**Tezin Enstitüye Verildiği Tarih : 07 Mayıs 2010  
Tezin Savunulduğu Tarih : 10 Haziran 2010**

**Tez Danışmanı : Doç. Dr. Mehmet Turan SÖYLEMEZ (İTÜ)  
Diğer Jüri Üyeleri : Doç. Dr. Salman KURTULAN (İTÜ)  
Yrd. Doç. Dr. D. Turgay ALTILAR (İTÜ)**

**HAZİRAN 2010**



## ÖNSÖZ

Bu tez çalışması sırasında, yoğun programında bana zaman ayırıp yardımcı olan, yol gösteren ve bu konuda çalışma fırsatı veren tez danışmanım Sayın Doç. Dr. Mehmet Turan SÖYLEMEZ'e, tez boyunca birlikte çalıştığım arkadaşım Arcan SONAT'a, mesai arkadaşım Ahmet KUZU'ya ve yüksek lisans öğrenimim boyunca vermiş olduğu destek ve katkılarından dolayı TÜBİTAK BİDEB'e sonsuz teşekkürü bir borç bilirim.

Yaklaşık 8 aydır Ulusal Demiryolu Sinyalizasyon Projesi (UDSP) kapsamında çalışmakta olduğum TÜBİTAK UEKAE de anlayışları ve destekleri için başta UDSP yürütücüsü Ayşen Daloğlu PETAN olmak üzere tüm mesai arkadaşlarıma teşekkürlerimi sunarım.

Son olarak bütün hayatım boyunca hem maddi hem manevi tüm desteklerinden dolayı aileme sonsuz teşekkürlerimi sunarım.

Haziran 2010

Serhat Türk

Elektrik Mühendisi



# İÇİNDEKİLER

## Sayfa

ÖNSÖZ.....	iii
İÇİNDEKİLER .....	v
KISALTMALAR .....	iii
ÇİZELGE LİSTESİ.....	v
ŞEKİL LİSTESİ.....	vii
ÖZET.....	ix
SUMMARY .....	xi
<b>1. GİRİŞ .....</b>	<b>1</b>
1.1 Tezin Amacı .....	1
<b>2. RAYLI ULAŞIM SİSTEMLERİNDE SİNYALİZASYON .....</b>	<b>3</b>
2.1 Raylı Ulaşım Sistemlerinde Sinyalizasyonun Tarihçesi .....	3
2.1.1 Elle çalışır blok sistemi .....	4
2.1.2 Kontrollü elle çalışır blok sistemi .....	5
2.1.3 Yarı otomatik blok sistemi .....	5
2.1.4 Otomatik blok sistemi .....	5
2.1.5 Mekanik blok sistemi .....	5
2.2 Sinyalizasyon Sistemi ve Temel Öğeleri .....	6
2.2.1 Kontrol merkezi .....	7
2.2.2 Anlaşman sistemi .....	8
2.2.2.1 Güzergah tanzimi .....	9
2.2.2.2 Güzergah tanziminin sonlandırılması .....	10
2.2.3 Saha ekipmanları .....	10
2.2.3.1 Ray devreleri.....	10
2.2.3.2 Makaslar .....	12
2.2.3.3 Sinyaller.....	13
2.2.3.4 Hemzemin geçit.....	14
<b>3. SİNYALİZASYON SİSTEMLERİNDE GÜVENLİK .....</b>	<b>15</b>
3.1 Hatada Güvenli Kavramı.....	15
3.2 Fonksiyonel Emniyet.....	16
3.3 Emniyet Bütünlüğü Seviyesi .....	17
3.4 Anlaşman Tablosu .....	19
<b>4. AYRIK OLAY SİSTEMLERİ .....</b>	<b>23</b>
4.1 Ayrık Olay Sistemlerinin Karakteristik Özellikleri.....	24
4.2 Dil Kavramı ve Modelleme Biçimi .....	25
4.2.1 Ayrık olay sistemlerinde dil kavramı .....	25
4.2.2 Otomat modelleme biçimi.....	25
4.2.2.1 Otomatlarla modellenen diller.....	26
4.2.2.2 Kilitlenme.....	26
4.2.2.3 Deterministik otomat.....	27
4.2.2.4 Deterministik olmayan otomat.....	28
4.3 Otomat Gösterimini PLC İle Gerçekleme .....	28
4.3.1 Kurma ve silme komutları ile gerçekleştirme yöntemi .....	29

4.3.1.1 Çığ etkisi sorunu.....	29
4.3.2 Mantık fonksiyonları ile gerçekleştirme yöntemi .....	30
<b>5. OTOMATİK ANKLAŞMAN ALGORİTMASI VE KODU ÜRETME</b>	
<b>YÖNTEMİ.....</b>	<b>43</b>
5.1 Otomatik Anlaşman Algoritması ve Kodu Üretme Yazılımının Tanıtılması. 43	
5.1.1 Tanzim otomati .....	45
5.1.2 Makas otomati .....	48
5.1.3 Sinyal otomati .....	51
<b>6. OTOMATİK ANKLAŞMAN ALGORİTMASI VE KODU ÜRETME</b>	
<b>YÖNTEMİ İLE ÖRNEK BİR HAT İÇİN ANKLAŞMAN ALGORİTMASI</b>	
<b>TASARLAMA .....</b>	<b>55</b>
6.1 Giriş .....	55
6.2 Örnek Hattın Tanıtılması ve Anlaşman Tablosu .....	55
6.3 Anlaşman Algoritması Tasarımı.....	58
6.3.1 Tanzim bloğu.....	58
6.3.2 Makas bloğu .....	61
6.3.3 Sinyal bloğu.....	63
<b>7. SİMÜLASYON.....</b>	<b>69</b>
7.1 Giriş .....	69
7.2 Modbus .....	69
7.3 Kullanıcı Veri Bloğu İletişim Kuralları.....	70
7.4 Simülasyon Arayüzü.....	71
<b>8. SONUÇ VE ÖNERİLER.....</b>	<b>75</b>
<b>KAYNAKLAR.....</b>	<b>77</b>
<b>ÖZGEÇMİŞ.....</b>	<b>79</b>



## **KISALTMALAR**

<b>AOS</b>	: Ayrık Olay Sistemi
<b>AS</b>	: Anlaşman Sistemi
<b>CENELEC</b>	: European Committee for Electrotechnical Standardization
<b>DNS</b>	: Domain Name System
<b>FBD</b>	: Function Block Diagram
<b>KM</b>	: Kumanda Merkezi
<b>LAD</b>	: Ladder Diagram
<b>LRT</b>	: Light Rail Transit
<b>PLC</b>	: Programmable Logic Controller
<b>RAMS</b>	: Reliability, availability, maintainability, safety
<b>SFC</b>	: Squential Function Chart
<b>SIL</b>	: Safety Integrity Level
<b>SNMP</b>	: Simple Network Management Protocol
<b>ST</b>	: Structured Text
<b>STL</b>	: Statement List
<b>TCDD</b>	: Türkiye Cumhuriyeti Devlet Demiryolları
<b>TCP</b>	: Transmission Control Protocol
<b>TFTP</b>	: Trivial File Transfer Protocol
<b>THR</b>	: Tolerable Hazard Rate
<b>UDP</b>	: User Datagram Protocol
<b>UDSP</b>	: Ulusal Demiryolu Sinyalizasyon Projesi



## ÇİZELGE LİSTESİ

### Sayfa

Çizelge 3.1 : SIL seviyelerine göre THR (Gündoğdu, 2005).....	18
Çizelge 3.2 : Risk grafiği sonuçları (Gündoğdu, 2005).....	18
Çizelge 3.3 : Risk parametreleri (Gündoğdu 2005).....	19
Çizelge 3.4 : 2DT-BT güzergahı için hazırlanmış anlaşıman tablosu satırı.....	20
Çizelge 6.1 : Örnek hat anlaşıman tablosu (Rota1-Rota3).....	56
Çizelge 6.2 : Örnek hat anlaşıman tablosu (Rota4-Rota18).....	57
Çizelge 6.3 : Örnek hat anlaşıman tablosunun 2D sinyali ile ilgili satırları.....	64



## ŞEKİL LİSTESİ

### Sayfa

Şekil 2.1 : Sinyalizasyon sistemi temel öğeleri .....	6
Şekil 2.2 : Kontrol merkezi.....	8
Şekil 2.3 : Kodlu ray devresi örneği, Söyler (2005)'ten uyarlanmıştır .....	11
Şekil 2.4 : Aks sayıcı örnekleri.....	12
Şekil 2.5 : Makas sinyalizasyonu örneği, Söyler (2005)'ten uyarlanmıştır.....	13
Şekil 3.1 : Hatada güvenli kavramı örneği, Goddard (2008)'dan uyarlanmıştır .....	15
Şekil 3.2 : IEC 61508 risk grafiği .....	18
Şekil 3.3 : Örnek bir sinyalizasyon planı.....	19
Şekil 4.1 : Sürekli sistem .....	24
Şekil 4.2 : Ayrık olay sistemi .....	24
Şekil 4.3 : Örnek bir otomat modelleme biçimi .....	26
Şekil 4.4 : Açmaz ve kısır döngü örneği .....	27
Şekil 4.5 : Deterministik otomat.....	27
Şekil 4.6 : Deterministik olmayan otomat. ....	28
Şekil 4.7 : Çiğ etkisi sorunu yaşanan otomat ve PLC kodu. ....	30
Şekil 4.8 : Eşitlik 4.6'ya karşılık gelen FBD dili programı. ....	32
Şekil 4.9 : Örnek 4.1'e ilişkin FBD dilinde PLC programı. ....	33
Şekil 4.10 : Altı durumlu otomat. ....	35
Şekil 4.11 : Otomatın ilk duruma kurulması. ....	35
Şekil 4.12 : Otomatın hem 0 numaralı durumda hem de 1 numaralı durumda kalması. .....	36
Şekil 4.14 : Otomat bir çevrim boyunca hiçbir durumda değil. ....	37
Şekil 4.15 : İlk duruma kurma problemine önerilen yeni çözüm yöntemi ile yazılmış program.....	37
Şekil 4.16 : Yeni ilk duruma kurma yöntemi ile otomatın 1 numaralı duruma geçişi. .....	38
Şekil 4.17 : Yeni ilk duruma kurma yöntemi ile otomatın 3 numaralı duruma geçişi. .....	38
Şekil 4.19 : Eşitlik 4.14 ve Eşitlik 4.15'e göre düzenlenmiş PLC programı.....	40
Şekil 5.1 : Geliştirilen yazılımın blok şeması.....	44
Şekil 5.2 : Yazılım menüsü.....	44
Şekil 5.3 : Tanzim otomatı .....	45
Şekil 5.4 : Tanzim otomatı tasarımı arayüzü. ....	48
Şekil 5.5 : Makas otomatı. ....	49
Şekil 5.6 : Makas otomatı arayüzü .....	50
Şekil 5.7 : Sinyal otomatı .....	51
Şekil 5.8 : Sinyal otomatı arayüzü.....	53
Şekil 6.1 : Örnek hat sinyalizasyon planı .....	56
Şekil 6.2 : 1BT-AT güzergahına ilişkin verilerin arayüze girilmesi .....	58
Şekil 6.3 : 1BT-AT güzergahı için tanzim otomatı ve geçişler. ....	59
Şekil 6.4 : 1BT-AT güzergahı için tanzim otomatı mantık fonksiyonları.....	60
Şekil 6.5 : 1 numaralı makas için verilerin arayüze girilmesi .....	61

<b>Şekil 6.6</b> : 1 numaralı makas otomatı ve geçişleri. ....	62
<b>Şekil 6.7</b> : 1 numaralı makas otomatı mantık fonksiyonları .....	63
<b>Şekil 6.8</b> : 2D sinyaline ilişkin verilerin arayüze girilmesi.....	63
<b>Şekil 6.9</b> : 2D sinyaline ilişkin geçişlerin tanımlanması.....	65
<b>Şekil 6.10</b> : 2D sinyaline ilişkin tüm geçişler. ....	66
<b>Şekil 6.11</b> : 2D sinyali otomatı. ....	66
<b>Şekil 6.12</b> : 2D sinyali mantık fonksiyonları. ....	67
<b>Şekil 7.1</b> : Simülatör arayüzü.....	71
<b>Şekil 7.2</b> : 1BT – AT güzergahının tanzim edilmesi. ....	72
<b>Şekil 7.3</b> : 1DT-AT güzergahının reddedilmesi.....	72
<b>Şekil 7.4</b> : İki güzergahın aynı anda işletilmesi .....	73

# RAYLI ULAŞIM SİNYALİZASYON SİSTEMLERİ İÇİN OTOMATİK ANKLAŞMAN ALGORİTMASI VE KODU ÜRETME YÖNTEMİ

## ÖZET

Günümüzde nüfusun artmasıyla birlikte ulaşım önemli bir sorun haline gelmiştir. Ulaşım ihtiyacının etkin ve hızlı bir şekilde karşılanabilmesi için toplu taşımaya yönelmek gereklidir. Raylı ulaşım sistemleri, hızlı ve oldukça fazla yolcu taşıma kapasiteleri ile toplu taşımada önemli bir yere sahiptirler. Raylı ulaşım sistemlerinde dakik, hızlı, güvenli ve ekonomik bir yolculuk için, sinyalizasyon büyük önem taşımaktadır. Her şeyden daha önemlisi güvenli bir taşıma yapabilmek ve kazaların önüne geçmek başarılı bir şekilde sinyalizasyon sistemleri tasarlamak gereklidir.

Raylı ulaşım sinyalizasyon sistemlerinde güvenliği sağlayan alt sistem anlaşıman sistemidir. Anlaşıman sistemi elektronik veya mekanik olarak röleler ile tasarlanabilmektedir. Anlaşıman sistemi tasarımı, Ayrık Olay Sistemi (AOS) olarak modellenerek gerçekleştirilebilir. AOS'lerde tasarım Otomatlar, Petri ağları gibi formal metotlarla yapılabilmektedir.

Bu tez çalışmasında, bir anlaşıman algoritma üretici yazılımının geliştirilmesi ve Microsoft Visual Basic ile uygulaması ele alınmıştır. Geliştirilen yazılım, girilen anlaşıman tablosunu gerçekleştirecek olan anlaşıman algoritmasının otomat modelini ve PLC gerçekleştirmesini sağlayan sözde (pseudo) kodu çıktı olarak vermektedir. Sözde kodun geliştirilen yazılım tarafından oluşturulabilmesi için çığ etkisi ve ilk duruma kurma sorununu engelleyen bir yöntem tanıtılmıştır. Ancak bu yöntemde problem çıkarabilecek durumlar saptanmış ve bu problemin giderilmesi için yeni bir yöntem önerilmiştir. Ayrıca önerilen otomatik anlaşıman yazılımı ve kodu üretme yazılımı, küçük bir istasyon bölgesinin anlaşıman tablosu girdi alınarak uygulanmış ve elde edilen çıktılar gösterilmiştir. Çıkış olarak üretilen kodlar, FBD diline dönüştürülmüş ve PLC üzerinde programlama gerçekleştirilmiştir. Elde edilen anlaşıman yazılımı simülatör ortamında ile test edilmiştir.

Bu çalışma ile literatüre yapılan katkı otomatları temel alan bir anlaşıman algoritması ve kodu üretici geliştirmiş olmaktır. Yapılan literatür çalışması sonucu elde edilen kanı, demiryolu sistemleri için Petri Ağları temelli otomatik algoritma üreticileri olmasına karşın otomatları temel alan böyle bir sistemin olmamasıdır.





# **AUTOMATIC INTERLOCKING ALGORITHM AND CODE GENERATION METHOD FOR RAILWAY TRANSPORTATION SIGNALIZATION SYSTEMS**

## **SUMMARY**

Nowadays transportation has become a significant issue with the increasing population. Heading towards public transportation is necessary in order to satisfy the needs of transportation fast and effectively. Being fast and with great passenger capacity, railway transportation systems have a considerable place in public transportation. Signalization has a great importance for a punctual, fast, safe and economic journey with railway transportation systems. The most important fact is to be able to design successful signalization systems so as to prevent accidents and provide safe transportation.

Interlocking system is the subsystem which provides security of a railway transportation signalization system. Interlocking systems can be designed electronically or mechanically via relays. Modelling of such systems can be achieved using Discrete Event Systems (DES) approach. In DES, design can be done using formal methods such as Automata Theory and Petri Nets.

In this thesis, development of an interlocking generator algorithm and its implementation on Microsoft Visual Basic has been considered. The output of the developed software is the automata model of interlocking and relevant pseudo codes for PLC implementation to realize the interlocking table. A method which prevents avalanche effect and the problem of setting first state is introduced for generating pseudo codes via the proposed software. However, in this method some problematic situations are detected and a new method has been suggested in order to eliminate the problematic situations. Besides an interlocking table of a sample railway yard has been taken into the proposed software as input and the output of the software has been exhibited. Pseudo codes generated as output has been converted to FBD and run on PLC. The generated interlocking has been tested via simulator.

In the course of this work a novel automata-based interlocking algorithm generator was designed and implemented. As a result of literature study, the opinion has been acquired that while there are many examples of works applying Petri Nets to automatic interlocking generation, very few examples of work utilizing the automata-based approach may be found in research literature.



## 1. GİRİŞ

Günümüzde ulaşım problemi önemli bir sorun teşkil etmektedir. Bu sorunun çözümünde en etkili yöntemlerden birinin raylı ulaşım olduğu aşikârdır. Raylı ulaşımı ön plana çıkaran en önemli unsur ise çok sayıda bireye hızlı ve güvenli bir ulaşım imkânı sağlamasıdır. Raylı ulaşım sistemlerinin hızlı ve güvenli bir ulaşım imkânı sunması başarılı bir şekilde tasarlanmış sinyalizasyon sistemlerine bağlıdır. Sinyalizasyon sistemlerinin de güvenlik ise anlaşılan sistemi ile birlikte sağlanır.

Demiryolu sinyalizasyon ve anlaşılan sistemlerinin davranış ifadesi zaman içerisinde anlık olarak gerçekleşen olaylara bağlı olarak verilebilmektedir. Bunlar trenin bir ray devresinden diğerine geçmesi, sinyalin bildirim vermesi veya hataya düşmesi, Kontrol Merkezinden anlaşılan sistemine bir talepte bulunulması, makasın arıza bildirim vermesi gibi anlık olaylar olabilir. Anlık olayların meydana gelmesi sistemi bir durumdan başka bir duruma taşımaktadır. Bu şekilde, bu tür sistemlerin davranışları, zamanda asenkron ve anlık olarak oluşan olaylara bağlı karakterize edilmektedir. Bu nedenle, böyle bir dinamik sınıfı oluşturan bu sistemler Ayrık Olay Sistemleri-AOS (discrete event systems-DES) olarak adlandırılmaktadırlar.

Bu çalışmada da sistem, AOS olarak modellenmiş ve modelleme için formal metotlardan otomat gösterim biçimi kullanılmıştır.

### 1.1 Tezin Amacı

Bu tezin amacı, tasarımcıya otomat gösterim biçimini kullanarak otomatik olarak anlaşılan algoritması üretme imkânı ve bu algoritmanın yine otomatik olarak Programlanabilir Lojik Kontrolör (PLC) kodlarını oluşturma imkânı sağlayan bir arayüz tasarlayarak, tasarımcının hem hızlı bir şekilde hem de minimum hata ile anlaşılan algoritması üretmesini sağlamaktır.

Bu amaç doğrultusunda geliştirilen yazılım kullanılarak örnek bir demiryolu hattı için anlaşılan algoritması ve kodu üretilmiş, üretilen bu algoritma Microsoft Visual C++ ortamında geliştirilmiş olan simülatör ile test edilmiştir.



## **2. RAYLI ULAŞIM SİSTEMLERİNDE SİNYALİZASYON**

### **2.1 Raylı Ulaşım Sistemlerinde Sinyalizasyonun Tarihçesi**

Raylı ulaşım sistemlerinin temeli 1841 yılında George Stephenson'ın ilk buharlı lokomotif demiryolu işletmesine almasına dayanmaktadır.

Raylı ulaşımın başladığı bu yıllarda hat, kavşak sayısı az olduğu ve tren katarları da az sayıda araçtan oluştuğundan güvenlik ile ilgili problemler için herhangi bir önlem almak gerekli görülüyordu; ancak yaşanan kazalar sonucunda güvenlik ihtiyacı hissedilmeye başlanmıştır. Çözüm yolu olarak ise el kol işaretleri ve bayraklarla demiryolu trafiği kontrol edilmeye çalışılmıştır.

Günden güne artan hat ve kavşak sayılarının yanı sıra tren katarlarının artan uzunluğu karşısında bu ilkel yöntemin yetersizliği ortaya çıkmış, daha etkili sistemler tasarlayabilmek için yoğun çalışmalar başlatılmıştır.

Bu çalışmaların öncülüğünü yapan ülke ise İngiltere olmuştur. Bu nedenle ilk anlaşılan sistemini ve blok sinyallerini tasarlayan ve tatbik edenler İngilizlerdir. Bu konuda faaliyet gösteren ikinci ülke ise İngiltere'nin ardından Amerika olmuştur.

El, kol ve bayrak işaretleriyle başlayan sinyal alanındaki gelişme ancak blok sinyalleri ve mekanik cihazların yavaş yavaş geliştirilmesinden sonra hızlanmıştır.

Bir trenin hareketini diğerine bildirmek için siyah beyaz flamanlar kullanılmaya başlanmıştır, ancak bu flamanların uzaktan seçilmeleri zorluk çıkarmıştır. Bunun üzerine 3'er mil (4944 m) aralıklarla dikilmiş 10 m yüksekliğinde direkler üzerine asılmış siyah ve beyaz renkte bezle kaplanmış top şeklinde sepet asma usulüne başlanmıştır.

Tren istasyondan ayrıldığı zaman beyaz sepet direk üzerine asılmaktaydı. Yolcu veya eşya indirip bindirme sırasında ise beyaz sepet yarı yüksekliğe indirilmekteydi. Beyaz sepetin direğin aşağısına indirilmesi ise dur ve bekle anlamına gelmekteydi. Siyah sepetin direğe asılması ise trenin geciktiği veya arıza nedeniyle yolda kaldığı anlamına gelmekteydi. Günümüzdeki sinyallerin yerine kullanılan bu sepetlerin durumları ise dürbünlerle izlenmekteydi.

Daha sonraları top şeklindeki sepetlerin yerine 1.25 m çapında üzerinde tehlike yazan kırmızı diskler kullanılmaya başlanmıştır. Direk üzerinde dönebilen bu diskin demiryoluna paralel ve üzerinde beyaz ışık asılı olması (gece için) geç anlamına gelmekteydi. Eğer diskin konumu demiryoluna dik ve kırmızı ışık asılı ise dur ve bekle bildirimini vermekteydi.

Tren hareketlerinin hız, emniyet ve ekonomi bakımından kontrolü için ilk olarak zaman aralığı metodu kullanılmıştır. Bu metoda göre trenler için belli aralıklar tespit edilmiş ve bu müddetlerde belirlenen karşılaşma noktalarına varma talimatları verilmişti, ancak bu yöntemde hareket halindeki bir trenin önünde aynı istikamette ilerleyen veya karşı istikametten gelen başka bir trenden haberi olmamaktaydı. Bunu önlemek için hattın belli yerlerine flamacılar konulmaktaydı. Hat kapasitesinin devamlı artması nedeniyle bu yöntemden de vazgeçilerek mesafe aralık metoduna geçilmiştir. Mesafe aralık metodu ile demiryolu hattı birçok kısma bölünerek bloklar oluşturulmuş ve her blok başına da bir işaret konulmuştur. Bu işaretler aracılığıyla tren makinistlerine girecek oldukları bloğun meşgul olup olmadığı hakkında bilgi veriliyordu. Bu metodun uygulanması birçok sabit hat sinyalinin icadına da vesile olmuştur.

1839' da İngiltere' de blok sisteminin etkili bir şekilde uygulanması, telgrafın kullanılmaya başlanmasıyla mümkün olmuştur. O yıllarda telgrafla yalnızca hat serbest veya meşgul bilgisi gönderilebiliyordu. 1851 yılında İngiltere'de sinyalleri zil sesleri ile verme yöntemi uygulanmıştır. 1854 yılında ise İngiltere'de zil ve telgraf birlikte kullanılmaya başlanmıştır. 1875 yılında Mr. W.R. Skyes elektrikli makas kilitleme tertibini keşfetti. Bu keşfin ardından trenler istasyonlar arasında daha güvenli bir şekilde seyretmeye başlamıştır. Bu tertip ile sinyal operatörleri blok sinyallerini elektrikselsel olarak kumanda edebiliyordu. Bir önceki istasyondaki operatör bir sonraki operatörden izin istiyor ve bir sonraki operatör kendi sinyal devresini çalıştırdıktan sonra izni isteyen operatör kendi devresini çalıştırabiliyordu.

Blok sinyalciliği 1897 yılına kadar gösterdiği gelişmeler sonucunda beş ana sınıfa ayrılmıştır. (Bölüm 2.1, Özdemir (2000)'den uyarlanmıştır.)

### **2.1.1 Elle çalışır blok sistemi**

Bu sistemde blok sinyalleri istasyonlarda bulunan sinyal operatörleri tarafından el ile çalıştırılır (Özdemir, 2000).

### **2.1.2 Kontrollü elle çalışır blok sistemi**

Bir ilerdeki istasyon sinyal operatörü tarafından bir gerideki sinyalin kontrol edildiği fakat blok giriş sinyallerinin giriş noktalarındaki sinyal operatörleri tarafından el ile çalıştırıldığı sistemlerdir (Özdemir, 2000).

### **2.1.3 Yarı otomatik blok sistemi**

Sinyallerin çalışması Sykes sistemindeki şeklinde olup ilave olarak tehlike işaretinin otomatik olarak trenler tarafından çalıştırılması şeklindedir (Özdemir, 2000).

### **2.1.4 Otomatik blok sistemi**

Blok sinyallerinin elektriksel olarak veya tazyikli hava ile tamamen otomatik olarak çalıştığı ve sinyal operatörlerinin bulunmadığı sistemlerdir (Özdemir, 2000).

### **2.1.5 Mekanik blok sistemi**

Tek hat kumandalı blok sistemi olup sinyallere ilave olarak bazı parçaların mekanik olarak kilitletiği ve elektriksel olarak ayrıldığı mekanizmaları içerir (Özdemir, 2000).

Demiryollarındaki gelişmeler ile birlikte tren sayısının ve hızının artması hat üzerinde bulunan elemanların bir merkezden kumandasını gerekli kılmıştır (Özdemir, 2000).

Bu tür gelişmelerin sonucunda makasların ve sinyallerin bir kişi tarafından kumandasına 1843 yılında İngiltere’ de başlanmıştır. Böylelikle anlaşılan sisteminin temelleri atılmıştır (Özdemir, 2000).

İngiltere’deki bu gelişmeler Osmanlı İmparatorluğuna 1856 yılında İzmir Aydın demiryolu hattının İngilizler tarafından işletmeye alınmasıyla yansımıştır. Cumhuriyetin kuruluşundan sonra milli demiryollarımız geliştirilmeye çalışılmış ve ilk defa 1955 yılında Sirkeci Halkalı banliyö hattına SIEMENS UND HALSKE firması tarafından otomatik blok sistemi kurularak devreye alınmıştır. Bunu 1957 yılında Haydarpaşa, Ankara, Zonguldak hattında kurulan merkezi kontrol sinyalizasyonu takip etmiştir, ancak bu hat 1965 yılına kadar kesintiye uğramıştır. 1965 yılında tekrar Amerikan UNION WABCO WESTINGHOUSE firması tarafından Haydarpaşa Ankara arasına merkezi kontrol sinyalizasyonunun montajı Amerikalı uzmanlar ve Türk teknik elemanlarca başlanmış, 1978 yılında

tamamlanmış ve hala çalışmaktadır (Özdemir, 2000). Günümüzde, ülkemizde 9000 km civarında demiryolu ana hattı vardır ve bu ana hatların %20 – 25'inde sinyalizasyon sistemi kuruludur (Söyler, 2005).

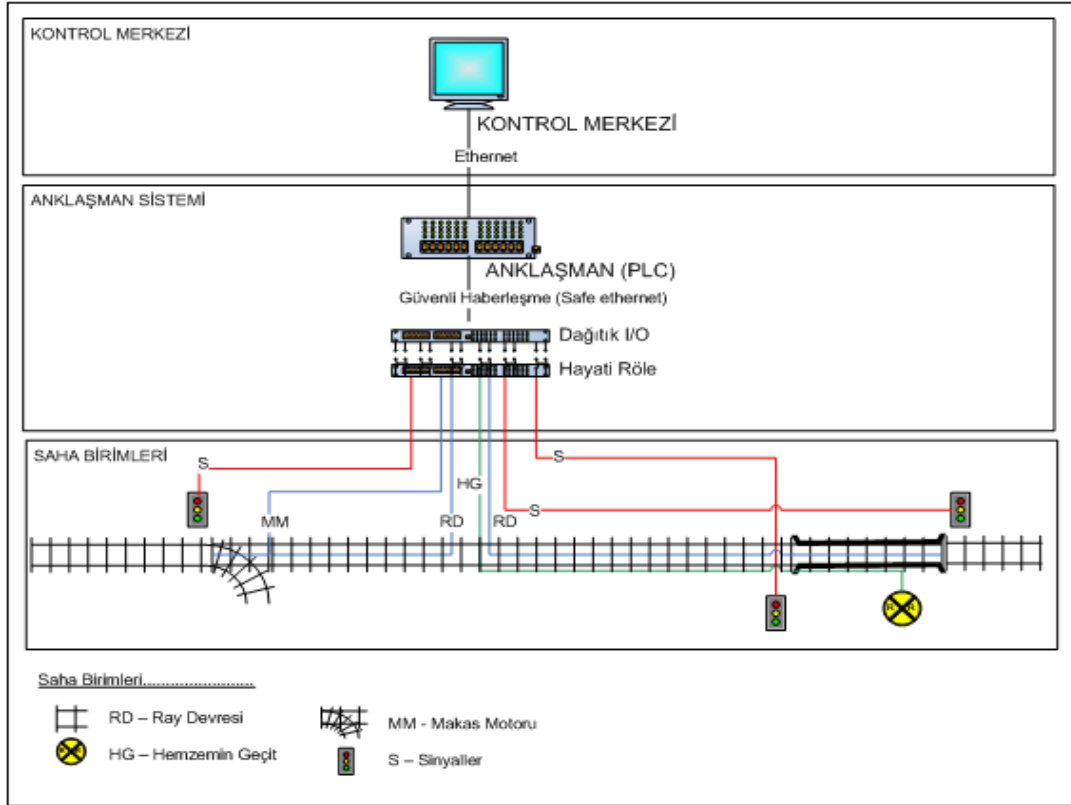
Son yıllarda teknolojinin gelişmesiyle birlikte gelişen bilgisayar ve PLC sistemleri ile çok başarılı sinyalizasyon sistemleri tasarlanabilmektedir. Bu çalışmada da PLC kullanılarak sinyalizasyon sisteminin temel öğelerinden biri olan elektronik anlaşıman tasarımı gerçekleştirilecektir.

## 2.2 Sinyalizasyon Sistemi ve Temel Öğeleri

Demiryolu sistemlerinde tren hareketinin kontrolü sinyalizasyon sistemi tarafından yapılmaktadır.

Sinyalizasyon sistemi üç ana öğeden oluşur.

- 1- Kumanda merkezi (KM)
- 2- Anlaşıman sistemi (AS)
- 3- Saha ekipmanları



Şekil 2.1 : Sinyalizasyon sistemi temel öğeleri.



Sinyalizasyon sisteminin temel fonksiyonlarını ve amacını;

- 1- Trenler arasındaki kazaların önlenmesi
- 2- Makasların yanlış tanzimi ve kilitlenmesi sonucu oluşabilecek raydan çıkma ve kazaların önlenmesi
- 3- Güzergah tanzimiyle çakışmayacak şekilde trenlere hareket etme yetkisi verilmesi
- 4- Hemzemin geçitlerin korunmasının sağlanması

olarak sıralayabiliriz (Gündoğdu ,2008).

Sinyalizasyon sisteminden elde edilecek faydaları ise aşağıdaki şekilde özetleyebiliriz.

- 1- İşletmecilik emniyeti artar.
- 2- Mevcut demiryolu kapasitesi artar.
- 3- Hattı işletmek kolaylaşır.
- 4- Zaman kazancı sağlanmış olur.
- 5- Personel istifadesi sağlanır.
- 6- Tren / saat başına groston / km yükselir.

Amerika'nın Ohio eyaletinde Stanley – Berwick arasındaki 64.7 km'lik demiryolu hattına 1927 yılında sinyalizasyon sistemi kurulmasıyla trenlerin hızları %36 artmış, tren / saat başına groston-km %39, hattın kapasitesi %40 ise oranında artmıştır. Sistemin sağladığı tasarruf tesis masraflarının %65'i olmuştur. Verilerden de açıkça görüldüğü üzere demiryolu işletmeleri için sinyalizasyon sistemleri başta güvenlik ve verim olmak üzere pek çok konuda son derece önemlidir (Özdemir, 2000).

### **2.2.1 Kontrol merkezi**

KM, dispeçer ile anlaşılan sistemi arasındaki arayüzdür. Dispeçer, demiryolu trafiğini idare etmekle yükümlü operatördür. Dispeçer, güzergah tanzimi komutunu ve diğer komutlarını bu arayüzü kullanarak vermektedir. Sahadan alınan bilgiler ise yine bu arayüz ekranında gösterilerek dispeçerin sahanın o anki durumu hakkında bilgi sahibi olması sağlanmaktadır.



**Şekil 2.2 : Kontrol merkezi.**

Sahadan alınan bilgiler KM tasarımına göre bir, iki veya üç grafik ekranında gösterilebilir.

### **2.2.2 Anlaşman sistemi**

AS'nin en önemli görevi sinyalizasyon sisteminin emniyet bütünlüğünü sağlamak ve sistemi hatada güvenli olarak çalıştırmaktır. Sinyalizasyon sistemlerinde, hatada güvenli ve anlaşman kavramları vazgeçilmez bir unsur kabul edilerek, tasarlanacak sistemlerin hatada güvenli tasarım kriterlerine uygun olarak yapılması gerekmektedir (Söyler, 2008). Hatada güvenli kavramı Bölüm 3.1'de ayrıntılı olarak ele alınacaktır.

AS, sinyalizasyon sisteminin mantık ve güvenlik işlevlerini yerine getirir. AS, her hareketi oluşturmak ve denetlemek suretiyle, trenlerin istasyon ve kontrol sahası boyunca güvenli hareketini sağlamak için kullanılır (Gündoğdu, 2008).

Ayrıntılı olarak AS'nin işlevleri aşağıdaki gibi sıralanabilir:

- 1- KM'den gelen komutların yönetimi
- 2- Güzergahların kontrol ve denetimi
- 3- Saha elemanlarının kontrol ve denetimi
- 4- Sinyalizasyon mantığının işletilmesi
- 5- Periyodik olarak sahanın durumu hakkında KM'ye bilgi aktarımı

AS, bu işlevleri yerine getirirken saha ekipmanlarından aldığı bilgilere göre trenin bir ray bölgesine girmesine izin verip vermeyeceğine karar verir. Bir makas ya da ray bölgesine herhangi bir tren girdiğinde o tren bu makas veya ray bölgesini terk edene

kadar bölge kilitlenir ve bölgede herhangi bir işlem yapılmasına izin verilmez. Temel olarak bu şekilde trenlerin karşılaşması ve çarpışması engellenmiş olur (Söyler, 2005).

AS, içinde merkezi işlem birimi barındıran PLC ve benzeri cihazlar geliştirilmeden önce röleler ile tasarlanmaktaydı. Röleli tasarımda meşgul olan bölgenin rölesi çeker ve söz konusu bölge ile ilgili başka komutlar işlenmezdi (Söyler, 2005). Yeni sinyalizasyon sistemlerinde ise AS yazılımsal (elektronik) tasarlanmaktadır.

AS donanımını ise hatada güvenli ve emniyet bütünlüğü seviyesi (Safety Integrity Level - SIL) 3 – 4 olan PLC ve giriş çıkış birimleri oluşturmaktadır.

### **2.2.2.1 Güzergah tanzimi**

Bir sinyalizasyon sisteminde güzergah tanzimi sadece AS tarafından yapılabilir. Güzergahın tanzim için uygun olup olmamasının kararı ise AS'nin aşağıdaki kontrolleri yapması sonucunda verilir.

- 1- Ray devreleri kontrolü
- 2- Güzergah kilitleri kontrolü
- 3- Çakışan güzergah kontrolü
- 4- Makas pozisyonu kontrolü
- 5- Eğer varsa hemzemin geçit kontrolü

Bu kontroller sonucunda

- 1- Ray devreleri müsaitse
- 2- Güzergah kilitleri açıksa
- 3- Çakışan güzergah tanzim talebi veya tanzim yoksa

Makaslar uygun konumda ise herhangi bir işlem yapılmaz. Eğer istenilen konumda değil ise istenilen konuma getirilir. Söz konusu güzergahın makasları ve kilitleri elektronik olarak kilitlenir. Son olarak yol boyu sinyalleri yakılır. Böylece güzergah tanzimi tamamlanmış olur. Eğer söz konusu şartlardan biri sağlanmamış ise güzergah tanzimi reddedilir.

### **2.2.2.2 Güzergah tanziminin sonlandırılması**

Kilitlenen bölgeden trenin çıkışı ile birlikte diğer trenlerin geçişine müsaade edilebilmesi için kilitler çözülür ve güzergah tekrar tanzim edilebilir duruma gelir. Güzergah tanzimi başka bir şekilde dispeçerin iptal komutuyla da sonlanabilir.

### **2.2.3 Saha ekipmanları**

Sinyalizasyon sistemi saha ekipmanları; ray devreleri, motorlu veya motorsuz makaslar, sinyaller ve hemzemin geçitlerden oluşmaktadır.

#### **2.2.3.1 Ray devreleri**

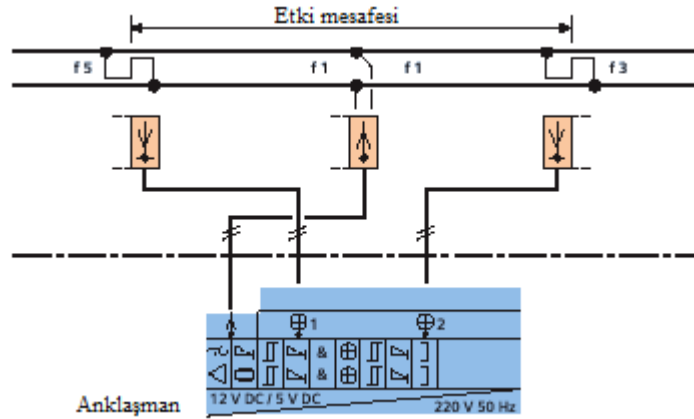
Ray devreleri hat üzerinde trenin yerini belirlemek için kullanılmaktadırlar. AS, bu sayede treni algılayabilmektedir. Dört tipte ray devresi bulunmaktadır.

#### **İzole cebireli ray devreleri :**

İzole cebireler ile birbirinden elektriksel olarak ayrılmış ray bölgelerine uygulanan gerilimin kontrol edilmesi ile trenin varlığı algılanır. Demiryolu hattı izole cebire ile belli bölgelere ayrıldıktan sonra bu bölgelerin herhangi bir tarafından bir besleme gerilimi verilir ve ray bölgesinin diğer tarafından da gerilim kontrol edilir. Eğer izole edilmiş bölgeden uygulanan gerilime göre bir geri dönüş gerilimi alınıyorsa ilgili ray bölgesinde tren yoktur. Tren bir ray bölgesine girince iki ray arasını kısa devre eder. Bu durumda raya uygulanan gerilimden geriye dönüş olmayacağından bölgede trenin varlığı algılanmış olur. Burada tren algılama sistemi ters mantıkla çalışır. Yani gerilim varsa tren yok, gerilim yoksa tren var anlamına gelir. Bunun sebebi ise hatada güvenli kavramının sinyalizasyon sistemi için vazgeçilmez bir unsur olmasıdır. Herhangi bir sebepten (kablo kopması, kısa devre, donanım arızası vs) uygulanan gerilim geri alınamazsa o bölgede trenin olduğu kabul edilir ve sistemde arıza olsa dahi en emniyetli duruma geçeceği için kazalar önlenmiş olur. Bu tip ray devrelerinde izole cebireler kullanıldığından bu ray devreleri üzerinde seyreden trenlerde yolculuk konforu düşüktür. Konforun düşük olması ise izole cebirelerden dolayı rayların arasında boşluk kalması ve yolculuk esnasında sese ve sarsıntıya sebep olmasıdır. İstanbul LRT hattı, İzmir Metrosu ve TCDD banliyö ve şehirlerarası hatlarında izole cebireli ray devreleri kullanılmaktadır (Söyler, 2005).

### Kodlu ray devreleri :

Kodlu ray devrelerinde rayları izole cebire ile ayırmaya gerek yoktur. Onun yerine ray bölgeleri arasında kapasitif ayırıcılar kullanılır. Ray bölgesinin bir ucundan verici vasıtası ile raya verilen ses frekansı ray bölgesinin diğer ucundan bir alıcı vasıtası ile alınır ve ölçülür. Eğer frekansta bir sapma varsa hatada güvenli kavramına göre tren varmış gibi düşünülür ve bölge kilitlenir. Son yıllarda inşa edilen olan sabit bloklu sistemlerde, ses frekanslı ray devreleri kullanılmaktadır. Özellikle kısa mesafeler de trenin algılanmasını gerektiren düşük zaman aralıklı tren işletmesi yapılan sistemlerde kullanılması avantajlıdır. Ayrıca ray kesintisiz olduğu içinde yolculuk konforu artar ve bakım maliyeti düşer. Son zamanlarda işletmeye açılan Ankaray raylı sistemler ve Taksim – 4 Levent İstanbul Metrosu kodlu ray devresi kullanmaktadır (Söyler, 2005).



Şekil 2.3 : Kodlu ray devresi örneği. (Söyler (2005)'ten uyarlanmıştır)

### Aks sayıcılı ray devreleri :

Aks sayıcılı ray devreleri, ray bölgesine giren çıkan tren akslarını sayarak trenin bölgede olup olmadığını algılayan ray devreleridir. Eğer bölgeye giren aks sayısı bölgeden çıkan aks sayısına eşit değilse hatada güvenli kavramına göre bölgede tren var kabul edilir. Özellikle şehirlerarası raylı sistemlerde bu tip ray devreleri tercih edilmektedir. Aks sayıcılı ray devrelerinin kullanıldığı sistemlerde izole cebire kullanmadığından bakımı kolaydır ve ray kesintisiz olduğu için yolculuk daha konforludur. Ülkemizde ise aks sayıcılı ray devresi Bursaray hattında kullanılmıştır. Dünya da ise özellikle şehirlerarası hatlarda hızla yaygınlaşmaktadır (Söyler, 2005).



**Şekil 2.4 :** Aks sayıcı örnekleri.

### **Hareketli blok ray devreleri :**

Hareketli blok ray devreleri aslında fiziki olarak yoktur. Bu tip ray devreleri sanaldır. Uzunlukları trenin hızına, durma mesafesine, fren gücüne, bölgenin karp ve eğim parametrelerine göre değişmektedir. KM'deki program her trenin önündeki mesafeyi otomatik olarak ayarlar ve trenin hızını buna göre düşürür veya artırır. Bu şekilde ray devresi olarak kullanılan mesafe kısa olacağı veya gereksiz yere uzun tutulmayacağı için hattın kullanılabilirliği artar. Hareketli blok ray devrelerinin genellikle 90 saniye ve altındaki hat kapasitelerinde kullanılması daha ekonomik olmaktadır. Ülkemizde Ankara metrosunda hareketli blok ray devresi kullanılmıştır (Söyler, 2005).

### **2.2.3.2 Makaslar**

Trenlerin demiryolu hattı üzerinde yön değişimleri makaslar sayesinde olmaktadır. Makaslar genel olarak uzaktan kumandalı ve el ile kumandalı olarak ikiye ayrılır.

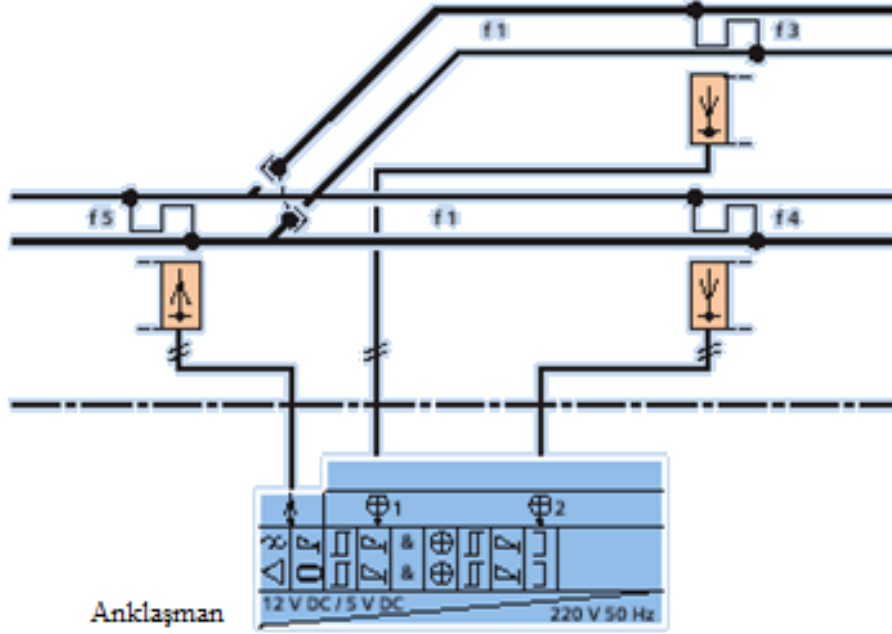
#### **Uzaktan kumandalı makaslar :**

Bu makaslar elektrik motoru ile çalışırlar. Bu motora, makas motoru ismi verilir. Makaslar, AS tarafından kontrol edilmektedir ve KM'den gelen makas ile ilgili komutlar AS üzerinden sahaya iletilmektedir. Makasların konumları ile ilgili bilgiler AS tarafından KM'ye periyodik olarak gönderilir.

#### **El ile kumandalı makaslar :**

Bu makaslar toplu makaslardır. Bu makas tanzim edileceği zaman mekanizma üzerinde bulunan kol kilitsiz duruma getirilir. Sonra makas topu el ile kumanda edilerek istenen komuna çevrilir. Bu tip makasların da yine konum bilgileri AS tarafından KM'ye iletilir.

Makaslar da ray devreleri gibi hatada güvenli kavramına göre çalışmaktadırlar. Motorlu makasların bulunduğu ray devresinde meşgul veya şüpheli bir durum söz konusu ise AS bu makasların hareket ettirilmesine izin vermez (Söyler, 2005).



Şekil 2.5 : Makas sinyalizasyonu örneği. (Söyler (2005)'ten uyarlanmıştır)

### 2.2.3.3 Sinyaller

Her ray bölgesinin veya yol girişlerinin başlangıcında trenlerin ilerlemesini veya durmasını kumanda eden trafik ışıkları bulunur. Bu trafik ışıkları yol boyu sinyalleri olarak adlandırılmaktadır. Sinyaller üçe ayrılmaktadırlar.

#### 4'lü yüksek sinyaller :

Bu sinyaller 3 – 3.5 m yükseklikte direk üzerine yerleştirilmiş dört birimli sinyallerdir. Çift hat uygulaması yapılan bölgelerdeki istasyonlarda çıkış sinyali olarak kullanılmaktadırlar. Sinyallerdeki renk dizilişi aşağıdan yukarıya doğru sarı, kırmızı, yeşil, sarı şeklindedir. Yeşil düz yola girileceğini ve çıkılacağını, sarı üzeri yeşil istasyona sapmalı girileceğini, çıkılacağını, durmadan geçileceğini bildirir. Sarı düz yola girilip ilerideki ilk sinyalde durulacağını, sarı üzeri sarı istasyona sapmalı girilip veya çıkılıp ilerideki ilk sinyal önünde durulacağını bildirir. Sarı üzeri kırmızı, ray devresiz yani korumasız yollara gidileceğini veya meşgul istasyon yoluna gidileceğini bildirir. Böylece korumasız yollarda veya istasyon yolunda bulunan vagon dizileri üzerine gidip ekleme yapılabilir.

### **3'lü yüksek sinyaller :**

Bu tip sinyaller trenin sapma imkânının olmadığı ana yol üzerine yerleştirilirler. Üç birimli sinyallerdir. 3'lü sinyallerdeki renk dizilişleri ise aşağıdan yukarıya doğru kırmızı, yeşil, sarı şeklindedir. Bu sinyallerinde verdikleri bildirimler sarı, yeşil ve kırmızı olup anlamları 4'lü yüksek sinyaller ile aynıdır.

### **Cüce sinyaller :**

Bu sinyaller sapmalı yollardan çıkışta kullanılmaktadırlar. Genelde gabari kurtarmayan hat aralarında kullanılmak üzere düşünülmüşlerdir. Üç birimli sinyallerdir. Renk dizilişleri aşağıdan yukarıya doğru sarı, yeşil, kırmızı şeklindedir. Yüksek sinyallerden farklı olarak yanar söner kırmızı, önünde sinyal bulunmayan yollardan gelip ray devresiz yollardan geçip ray devreli yollara gidileceğini bildirir. Yanar söner yeşil ve sarı, bloğun durumuna göre önünde sinyal bulunmayan yollardan trenin çıkışı için kullanılır.

#### **2.2.3.4 Hemzemin geçit**

Demiryolu ile karayollunun kesiştiği yerlere hemzemin geçit denilmektedir. Hemzemin geçitlerin kontrolü ile trenin bu bölgelerden güvenli bir şekilde geçmesi sağlanmaktadır.

AS, hemzemin geçidi kapsayan bir tanzim olmadığı zaman hemzemin geçidin açık tutulması sağlamak için hatada güvenli kavramına göre hemzemin geçide sürekli deaktif bilgisi göndermektedir. İlgili hemzemin geçidi kapsayan herhangi bir güzergah tanzim edildiğinde ise deaktif bilgisi kesilecek ve hemzemin geçit karayolu trafiğine kapatılacaktır. Kablo kopması gibi durumlarda hemzemin geçide AS tarafından deaktif bilgisi gönderilemeyeceğinden hatada güvenli olarak çalışacak ve kapanacaktır.



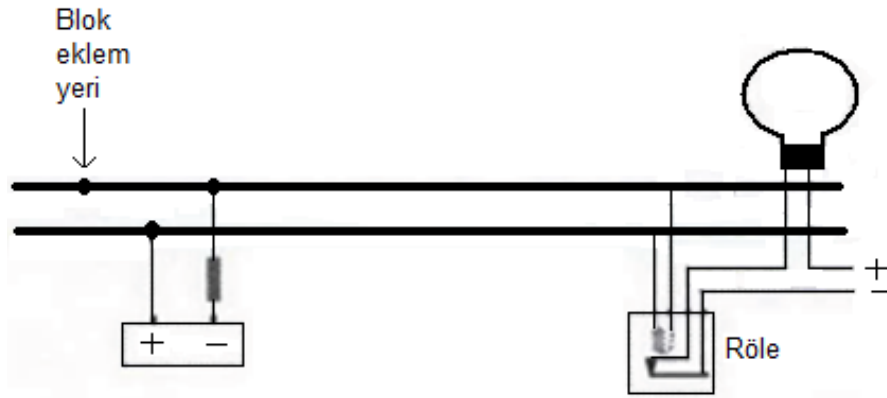
### 3. SİNYALİZASYON SİSTEMLERİNDE GÜVENLİK

#### 3.1 Hatada Güvenli Kavramı

Hatada güvenli kavramı demiryolu sinyalizasyon sistemlerinin temel tasarım prensibi olarak sistemin tümünde ya da tek bir bileşeninde, tehlike ve hata oluşturabilecek unsurları ortadan kaldırmak amacıyla tanımlanmıştır (Gündoğdu, 2008).

Hatada güvenli kavramını şöyle açıklayabiliriz; sistem herhangi bir işlem yaparken başarısız olup hataya düştüğünde, sistemin hatada güvenli duruma geçmesidir. Başarısız/hatalı olan işlem neticesinde sistemin güvenlik bütünlüğü zedelenmeden, sistem o durum için en güvenli olan pozisyona geçer (Gündoğdu, 2008).

Hatada güvenli kavramına örnek olarak aşağıdaki şekil kullanılarak ray devrelerinin çalışma mantığı verilebilir.



**Şekil 3.1 :** Hatada güvenli kavramı örneği. (Goddard (2008)'dan uyarlanmıştır)

İki izole cebire arasında bulunan ray devresinde tren yok iken röle kontağını çekmiş konumdadır ve lamba yeşil ışık verir. Trenin iki izole cebire arasına yani ray devresine girmesiyle röle kısa devre olur ve röle kontağı düşer. Bunun sonucunda da lamba söner (Goddard, 2008).

Kablo kopması, bataryanın bitmesi gibi beklenmedik bir durumda lamba yine sönecektir. Gerçek sistemde bu olay hataya yol açabilecek beklenmedik bir durumda AS'nin ray devresinden lojik olarak sıfır okuması ve ilgili ray devresinde tren

olduğunu kabul ederek işlem yapması anlamına gelir. Böylece de güvenli duruma geçilmiş ve olası bir kaza engellenmiş olur.

Sistem tasarımında, sistemi oluşturan tüm alt sistemlerin gerçekleştireceği fonksiyonlar için tehlikeli durum ve risk analizleri yapılarak fonksiyonel emniyet sağlanmalı ve oluşabilecek riskler tasarım aşamasında ortadan kaldırılmalıdır. Fonksiyonel emniyet, sistem emniyetinin bir parçasıdır ve sistemin girdilerine göre doğru işlemi yapmasına bağlıdır. Sistemlerin tasarımında, sistem emniyetine etki edecek tüm fonksiyonlar çıkarılarak her fonksiyon için hatada güvenli durumlar oluşturulmalıdır. Hatada güvenli durumlar tespit edildikten sonra sistemde, sistem emniyetine etki edecek herhangi bir hata oluşması durumunda sistemin o fonksiyon için belirlenmiş olan hatada güvenli duruma geçmesi garanti edilmelidir. Sistemlerin hatada güvenli olarak tasarlanması, sistemin emniyet bütünlüğü seviyesinin (Safety Integrity Level-SIL) yükseltilmesinde önemli rol oynamaktadır. Sinyalizasyon sistemleri tasarlanırken, tasarlanacak olan sistemin emniyet seviyesi dikkate alınmalıdır. Emniyet seviyesi belirlenirken işletmelerin ihtiyacı ve talepleri de göz önünde bulundurularak, o sistem için gerekli olan seviye belirlenir. Sinyalizasyon sistemleri için en yüksek emniyet seviyesi, tolere edilebilir riskin minimum düzeyde, yolcu veya personel için maksimum risk miktarının normal yaşamdaki riske eşit seviyede olmasıdır (Gündoğdu, 2008).

### **3.2 Fonksiyonel Emniyet**

Sistemlerin emniyetli olabilmeleri için tüm alt fonksiyonlarını emniyetli bir şekilde gerçekleştirmeleri gerekmektedir. Fonksiyonel emniyetin sağlanması, tehlikeli durumların giderilmesini gerektirir. Tehlikeli durumların ortadan kaldırılması, azaltılması ve kalıcı emniyetin sağlanması tasarım aşamasında gerçekleştirilir (Gündoğdu, 2005).

Fonksiyonel emniyet kavramını daha da netleştirmek için şu örnek verilebilir. Dönen demir bıçaklı, koruma kapağı olan bir makineyi ele aldığımız düşünelim. Makede temizlik işlemleri kapak kaldırılarak yapılabilir olsun. Koruma kapağı iç kilitleme sistemine sahip, kapak açıldığı zaman elektrik kesici devre vasıtası ile motorun enerjisi kesildiğinden bıçaklar duruyor ve operatör güvenli bir şekilde temizlik işlemini gerçekleştirebiliyor (Gündoğdu, 2005).

Güvenliğin sağlandığından emin olmak için çeşitli analizler yapmak gerekmektedir. Bu analizler tehlikeli durum ve risk değerlendirme analizleridir (Gündoğdu, 2005).

Tehlikeli durum analizi, döner bıçakların temizlenmesiyle oluşacak tehlikeleri tanımlamaktadır. Operatörün zarar görmemesi için koruma kapağının 5 mm'den fazla açıldığında sistemin acil frenlemeye girerek makineyi durdurması gerektiği tespit edilmiştir. Daha ileri analizler için kapak açıldığında makinenin 1 sn içinde durdurulması gerektiği düşünülebilir (Gündoğdu, 2005).

Risk değerlendirme analizi, emniyet işlemlerinin performansını ve başarısını belirlemektedir. Risk değerlendirmesinin amacı, emniyet bütünlüğünü sağlamak için gerçekleştirilen emniyet işlemlerinin, tehlikeli durumla ilgili kabul edilemez risk değerini aşmamasını sağlamaktır (Gündoğdu, 2005).

Emniyet işleminin hatası operatörün zarar görmesi ile sonuçlanabilir. Buradaki risk, koruma kapağının açılma sıklığıyla ilgilidir. Gerekli olan emniyet bütünlüğü seviyesi yaralanmanın şiddeti ve tehlikeli durumun oluşma sıklığıyla artmaktadır (Gündoğdu, 2005).

### **3.3 Emniyet Bütünlüğü Seviyesi**

Demiryolu standartlarında, tehlikeli durum tanımlamasının ve risk analizlerinin yapılması çok önemli bir yer tutar. Bu analizler sonucunda teknik sistemlerin oluşturabilecekleri risk değerleri bulunur. Bu değerler kullanılarak sistemin sahip olması gereken emniyet bütünlüğü seviyesi belirlenir (Gündoğdu, 2005).

Demiryolu standartları, Avrupa Elektroteknik Standartlar Enstitüsü (CENELEC) tarafından geliştirilen EN 50126, EN 50128 ve EN 50129 standartları ile belirlenmiştir. EN 50126 tüm raylı sistemleri kapsar ve RAMS hesapları ile ilgilidir. EN 50129 emniyet ilişkili elektrik elektronik, kontrol ve koruma sistemlerinde uyulması gerekli standartları belirler. EN 50128 emniyet ilişkili kontrol ve koruma sistemleri yazılımlarını kapsamaktadır. EN 50128 ve EN 50129 standartları, Uluslararası Elektrik-Elektronik, Programlanabilir Elektronik standardı IEC 61508'in raylı sistemlerle ilgili kısımların geniş yorumu ve raylı sistemlerdeki uygulamasıdır (Gündoğdu, 2005).

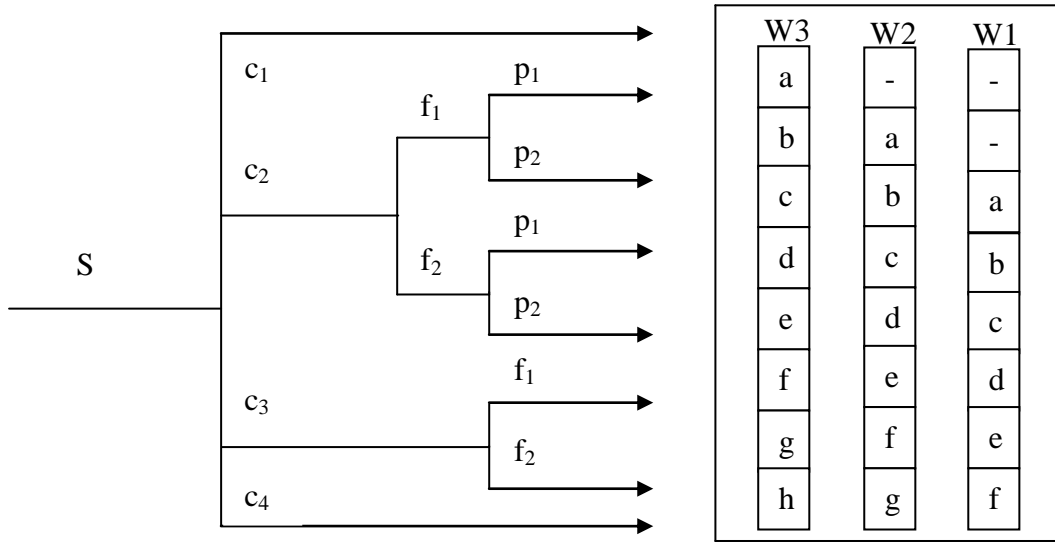
CENELEC raylı sistemler standartları kabul edilebilir risk değerlerini belirlemiştir. Risk analizi ve risk değerlendirme metotlarının uygulanmasıyla emniyet bütünlüğü

seviyeleri elde edilir. Çizelge 3.1’de SIL seviyelerine göre tolere edilebilir tehlike oranları (Tolerable Hazard Rate - THR) verilmiştir (Gündoğdu, 2005).

**Çizelge 3.1** : SIL seviyelerine göre THR (Gündoğdu, 2005).

TOLERE EDİLEBİLİR TEHLİKE ORANI, THR (Fonksiyon/saat)	EMNİYET BÜTÜNLÜĞÜ SEVİYESİ, SIL
$10^{-9} \leq \text{THR} < 10^{-8}$	4
$10^{-8} \leq \text{THR} < 10^{-7}$	3
$10^{-7} \leq \text{THR} < 10^{-6}$	2
$10^{-6} \leq \text{THR} < 10^{-5}$	1

Emniyet bütünlüğü seviyesi, IEC 61508 standardında belirtilen ve Şekil 3.2’de verilen risk grafiği metoduyla da hesaplanabilir.



**Şekil 3.2** : IEC 61508 risk grafiği.

Çizelge 3.3’de verilen risk parametrelerinin Şekil 3.2’deki risk grafiğine girilmesiyle elde edilen harflerin Çizelge 3.2’de verilen risk grafiği sonuçlarından kontrol edilerek sistemin emniyet bütünlüğü seviyesi belirlenebilmektedir.

**Çizelge 3.2** : Risk grafiği sonuçları (Gündoğdu, 2005).

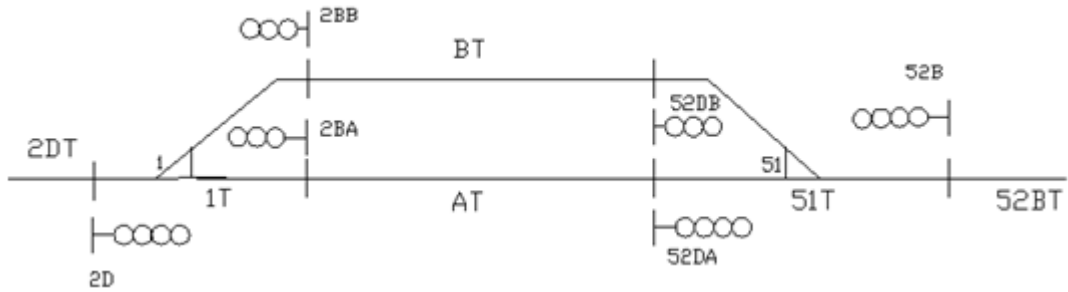
GEREKLİ MİNİMUM RİSK AZALTMA	GÜVENLİK SEVİYESİ
-	Güvenlik gerekli değil
a	Özel güvenlik ekipmanı gerekli değil
b, c	1
d	2
e, f	3
g	4
h	E / EE / PE SRS yeterli değil

**Çizelge 3.3 :** Risk parametreleri (Gündoğdu 2005).

RİSK PARAMETRE	TANIMLAMA
C1	Küçük yaralanma
C2	Birden fazla kişinin ciddi yaralanması ya da bir kişinin ölümü
C3	Birkaç kişinin ölümü
C4	Çok kişinin ölümü
F1	Nadiren-Sıklıkla oluşma ihtimali
F2	Devamlı sürekli oluşma ihtimali
P1	Bazı şartlar altında mümkün
P2	Mümkün değil
W2	Çok az ihtimalle oluşabilir ve tekrarlanabilir
W2	Az ihtimalle oluşabilir ve tekrarlanabilir
W3	Daha çok ihtimalle oluşabilir veya tekrarlanabilir

### 3.4 Anlaşman Tablosu

Anlaşman tablosu, sinyalizasyon sisteminin fonksiyonel emniyeti için çok önemlidir (Tombs, 2002). Anlaşman tablosunda saha ekipmanlarının birbirine göre olmaları gereken durumlar, tanzimli güzergahlar için kilitli olması gereken diğer güzergah ve makaslar, sinyallerin hangi durumlarda hangi bildirimleri vermesi gerektiği gibi güvenlik açısından önemli bilgiler bulunmaktadır. Bu bilgiler anlaşman yazılımı için de temel oluşturmaktadır. Anlaşman yazılımı tasarımı da anlaşman tablosunun hazırlanmasıyla başlamış olur. Anlaşman tablosunun tasarımcıya sağladığı bilgiler ışığında anlaşman yazılımı ve sisteminin hangi girişlere karşılık hangi çıkışlar üretmesi gerektiği ortaya çıkar.



**Şekil 3.3 :** Örnek bir sinyalizasyon planı.

Şekil 3.3’de verilen hattın anlaşman tablosunun 2DT-BT güzergahı için hazırlanmış satırı Çizelge 3.4’de verilmiştir.

**Çizelge 3.4 : 2DT-BT güzergahı için hazırlanmış anlaşıman tablosu satırı.**

ROTA	SİNYAL NO		SİNYAL DURUMU		KİLİT	SİNYAL KONTROL	ROTA KİLİT	
2DT - BT	2D	B	YS	52DB S	①	2BB 52BB	1T BT	(1T)
			SS	52DB K				
			SK					
						1T (BT)		

Rota bölümüne güzergahın (rotanın) adı yazılmaktadır. Örnekte ele alınan güzergah 2D sinyalinin başlayıp 52DB sinyalinde bittiğinden güzergah 2DT-BT olarak isimlendirilmiştir. Yukarıda verilen anlaşıman tablosu örnek hattın anlaşıman tablosunun bir kısmını oluşturmaktadır. Anlaşıman tablosunun tamamı hatta bulunan tüm güzergahlar için oluşturulacak satırlardan sonra tamamlanacaktır.

2D ise güzergah sinyalinin numarasını belirtmektedir. Sinyal durumu sütununda 2D sinyalinin, 2DT-BT güzergahı tanzim edildiği takdirde 2D sinyali ile aynı istikametteki ilk sinyal olan 52DB sinyaline göre hangi bildirimleri vermesi gerektiği gösterilmektedir.

Kilit sütununda yer alan daire içindeki 1 rakamı aslında ilgili güzergah için 1 numaralı makasın sapan konumunda kilitlenmesi gerektiğini belirtmektedir. Eğer 1 rakamı üzerinde daire olmasaydı bu sefer de düz konumda kilitlenmesi gerektiğini belirtilmiş olacaktı. Aynı sütunda yer alan 2BB sinyali ise güzergaha ters yönde bir sinyal olduğundan kilitlenmesi gerekmektedir. Sinyalin kilitlenmesi ise kırmızı bildirim vermesi anlamına gelmektedir. 52BB, 52B sinyalinin geçerek BT ray devresinde sonlanan güzergahın kilitli olması gerektiğini göstermektedir. Böylece iki trenin kafa kafaya gelmesi önenebilecektir.

Sinyal kontrol ve algılayıcı kilit kısmında yer alan 1T ve BT ise 2DT – BT güzergahı için bir güzergah tanzimi talebi geldiğinde kontrol edilmesi gereken ray devrelerini göstermektedir. Tanzim talebinin uygun bulunup 2D sinyalinin sarı üstü yeşil veya sarı üzeri sarı bildirim vermesi 1T ve BT ray devrelerinin müsait olmasına bağlıdır. Aynı kısmın alt satırında bulunan 1T ve daire içine alınmış BT ise 1T ray devresinin müsait olması BT ray devresinin ise meşgul olabileceği anlamına gelmektedir. Bu durumda ise 2D sinyali sarı üzeri kırmızı bildirim verecektir. Bu durum sadece istasyon yolundaki ray devreleri için geçerlidir. İstasyon yolunda bulunan ray devreleri dışındaki ray devrelerinin güzergah tanzimi için mutlaka müsait olması gerekir.

Rota kildi veya diđer bir deyişle güzergah kilidi ise güzergah tanzim talebinin gelmesiyle kilitlenir ve ancak tanzim talebinin uygun bulunmayıp reddedilmesi veya tanzimin sonlanmasıyla çözülür. Böylece AS, 2DT-BT güzergahı tanzimliken veya tanzim işlemi devam ederken aynı makasları kullanacak olan başka bir güzergahın örneğin AT-2DT güzergahının tanzim talebinin gelmesi durumunda bu talep reddeder. Tren güzergah kilidini veya kilitlerini terk edip güzergahın son ray devresine girdikten hemen sonra güzergah tanzimi düşürülür ve kilitler açılır. Örnekte ele alınan güzergah için tren 1T ray devresinden çıkıp BT ray devresine girince tanzim sonlanacak ve kilitler açılacaktır.

Anklaşman tabloları için tek bir gösterim biçimi veya format yoktur. Sütunların isimlendirilmeleri, sıraları ve gösterimleri farklılık gösterebilmektedir. Ancak tüm anklaşman tabloları farklı gösterimlerle de olsa aynı amaç doğrultusunda fonksiyonel emniyetin sağlanabilmesi için oluşturulmaktadır (Tombs, 2002).





#### 4. AYRIK OLAY SİSTEMLERİ

Hızlı teknolojik gelişmeler ile birlikte sistem dinamiği zamana bağlı sürekli olarak ifade edilemeyen sistemleri beraberinde getirmiştir. Bu sistemlere örnek olarak çeşitli ulaşım sistemleri, bilgisayar ağı sistemleri, haberleşme sistemleri, üretim sistemleri, trafik sistemleri, veritabanı sistemleri, robotlar verilebilir. Bu sistemlerde zaman sürekli olarak ilerlemez ve dolayısıyla değişkenler sürekli değildir (Hasdemir, 2008).

Sürekli sistemlerde,  $X$  durum uzayı,  $\mathbf{R}$  reel sayılar kümesi olmak üzere  $x(t)$  durumu bu kümeden

$$\dot{x}(t) = f(x(t), u(t), t) \quad (4.1)$$

ifadesi ile belirlenen bir değer alabilirken, ayrık zamanda bu diferansiyel denklemin karşılığı

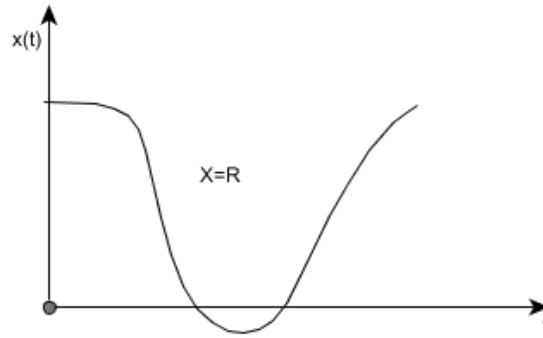
$$x(k + 1) = f(x(k), u(k), t) \quad (4.2)$$

şeklinde bir fark denklemdir. Bu denklemlerdeki  $u$  ise sisteme uygulanan girişi göstermektedir.

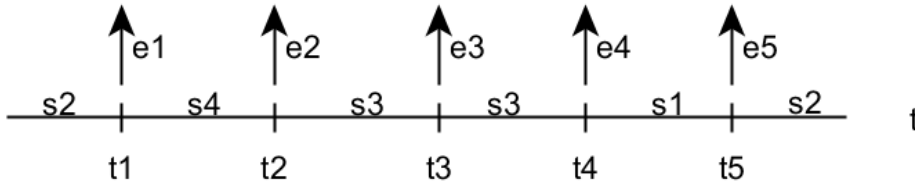
Ayrık olay sistemlerin durum uzayları  $X = \{s_1, s_2, s_3, \dots, s_n\}$  gibi ayrık kümelerden oluşan lineer olmayan sistemlerdir. Durumlar arası geçişler sürümlüdür ve bu geçişler tetikleme koşulunun sağlanmasıyla gerçekleşir. Sistemin durum geçişi zamana bağlı olarak belirlenmez, durum geçişini belirleyen unsur ise herhangi bir anda oluşabilen olaylardır. Ayrık olay sistemlerinde geçişler asenkron olarak tetiklenebilmektedir (Hasdemir, 2008).

Şekil 4.1 ve Şekil 4.2 karşılaştırılarak sürekli bir sistem ile ayrık olaylı sistem arasındaki fark anlaşılabilir. Ele alınan ayrık olaylı sistemin durum uzayı  $X = \{s_1, s_2, s_3, s_4\}$  şeklindedir. Şekilden de anlaşıldığı üzere ayrık olaylı sistemin durumu bir olay meydana geldiğinde değişebilmektedir, ancak her olay meydana geldiğinde sistem durum değiştirecek diye bir ön yargı oluşmamalıdır. Sistem bir durumdayken ancak sistemi bulunduğu durumdan çıkaracak ilgili geçişin gelmesi ile sistem durum

değiştirir. Durumla ilgisiz bir geçişin oluşması durum geçişini sağlamaz (Hasdemir, 2008).



Şekil 4.1 : Sürekli sistem.



Şekil 4.2 : Ayrık olay sistemi.

Sonuç olarak, ayrık olaylı sistemler ile ilgili olarak sistemin durum uzayındaki yerini sadece olayların belirlediğini söyleyebiliriz.

#### 4.1 Ayrık Olay Sistemlerinin Karakteristik Özellikleri

Eş zamanlılık: Birçok işlem ayrık olay sistemlerinde aynı anda meydana gelebilir (Hasdemir, 2008).

Asenkron işlemler: Ayrık olay sistemlerinde her değişim senkronize edilmez. Olaylar çoğu zaman asenkron olarak meydana gelir (Hasdemir, 2008).

Olay sürümlülük: Bir olay diğer olayların oluşumuna bağlı olabilir (Hasdemir, 2008).

Belirsizlik: Belirsizlik kesin olmayan olay oluşumları sonucunda meydana gelir. Örneğin, verilen bir durumdan farklı gelişmeler oluşması mümkün olabilir (Hasdemir, 2008).

## 4.2 Dil Kavramı ve Modelleme Biçimi

### 4.2.1 Ayrık olay sistemlerinde dil kavramı

Ayrık olay sistem modelleri dil olarak adlandırılmaktadır. Ayrık olay sistem modellerinin dil olarak adlandırılmasının nedeni sistemin ilgili olaylar kümesinin alfabe olarak değerlendirilmesidir (Cassandras, 1999). Örneğin, Şekil 4.3'deki sistemin olay kümesi  $E = \{e_1, e_2, e_3, e_4\}$  bir alfabe olarak değerlendirilebilir. Bu kümenin elemanları harf, bu harflerden oluşan olay dizileri de kelime olarak değerlendirilir. Olay dizisi örnekleri ise  $e_1e_2e_3$ ,  $e_1e_2$ ,  $e_2e_2e_3$ ,  $e_1$  şeklinde verilebilir. Sistem için geçerli olan sonlu uzunluklu olay dizilerinin oluşturduğu küme ise  $E$  alfabeti ile oluşturulmuş dili meydana getirir. Dil örneği ise  $L_1 = \{e_1, e_1e_2, e_2e_2e_3, e_1e_2e_3\}$  şeklinde verilebilir. Bu dil dört kelimeli olarak tanımlanmıştır.

### 4.2.2 Otomat modelleme biçimi

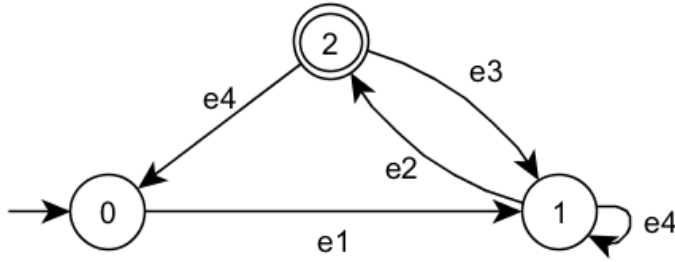
Ayrık olay sistemlerinin çoğunda olası tüm olay dizilerini listeleyerek dil tanımlanamaz çünkü çoğu ayrık olaylı sistemde dil sonlu olmamaktadır. Bu yüzden ayrık olaylı sistemlerin modellenmesinde sonlu yapılara ihtiyaç vardır. Sistem davranışlarını ifade eden bu yapılar modelleme biçimleri veya gösterim biçimleri olarak adlandırılmaktadır. Literatürde ayrık olaylı sistemlerin davranışlarının ifade edilmesinde sıkça otomat ve Petri ağları modelleme biçimleri kullanılmaktadır (Hasdemir, 2008).

Daha açıklayıcı olmak için bir önceki paragrafı özetleyecek olursak ayrık olay sistemlerine ilişkin tanımlanan diller çoğunlukla sonsuz bir kümeye denk düşer. Bu nedenle otomat veya Petri ağları gibi sonlu yapıda olan gösterim biçimlerine ihtiyaç duymaktayız. Sonlu otomatlara sonlu durum makinesi de denilmektedir (Hasdemir, 2008).

Bu çalışmada otomat modelleme biçimi üzerinde durulacak ve yapılacak tasarımda bu modelleme biçimi kullanılacaktır.

En basit ifadeyle otomat gösterimi durum geçiş diyagramı olarak tanımlanabilir. Durum geçiş diyagramında oklar durum geçiş fonksiyonlarına, okların etiketleri olaylara, daireler ise durumlara denk gelmektedir. Küçük bir ok ile işaret edilen

durum ise ilk durumu temsil eder (Hasdemir, 2008). Şekil 4.3’de örnek bir otomat modelleme biçimi gösterilmiştir.



Şekil 4.3 : Örnek bir otomat modelleme biçimi.

Bu otomatın durum kümesi  $Q = \{q_0, q_1, q_2\}$ , olay kümesi ise  $E = \{e_1, e_2, e_3, e_4\}$  biçimindedir. Otomatın ikinci durumunda çift daire olmasının nedeni bu durumun işaretli olmasıdır. İşaretli durum, bir duruma özel bir anlam ve işlev yüklenmek istendiğinde kullanılır. İşaretli durum otomatın işlevini tamamladığı final durumları için de kullanılır. Yukarıdaki örnek için durum geçiş fonksiyonu  $f(q_0, e_1) = q_1$ ,  $f(q_1, e_2) = q_2$ ,  $f(q_1, e_4) = q_1$ ,  $f(q_2, e_3) = q_1$ ,  $f(q_2, e_4) = q_0$  şeklindedir. Durum geçiş fonksiyonunun anlamını açıklamak gerekirse,  $f(q_0, e_1) = q_1$  durum geçiş fonksiyonu, otomat  $q_0$  durumundayken  $e_1$  olayı meydana gelirse otomatın  $q_1$  durumuna geçeceğini ifade eder.  $q_1$  durumunda  $e_4$  etiketli okun kendi üzerine dönmesi özçevrim olarak adlandırılır ve anlamı otomat  $q_1$  durumundayken  $e_4$  olayı meydana gelirse otomatın durum değiştirmeyeceğidir.

#### 4.2.2.1 Otomatlarla modellenen diller

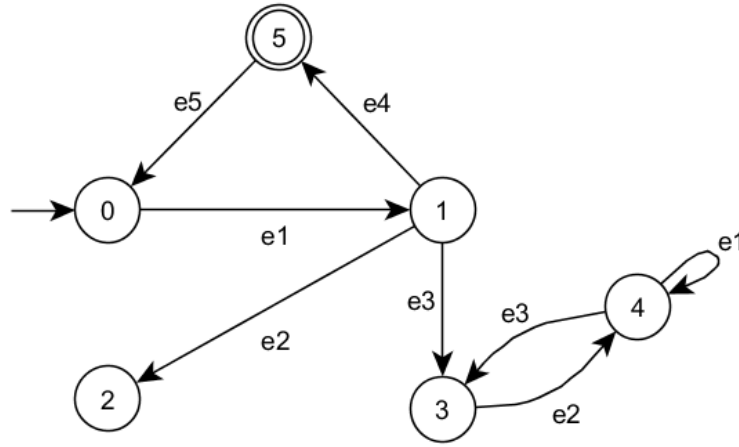
G isimli bir otomat tarafından modellenen dil  $L(G)$  biçiminde gösterilmektedir. İşaretlenen dil ise  $L_m(G)$  şeklinde ifade edilir.  $L_m(G)$ ,  $L(G)$ 'nin alt kümesidir ve otomat durum geçiş diyagramında işaretli durumlara giden tüm kelimeleri kapsar (Hasdemir, 2008).

İşaretlenen dile örnek olarak  $E = \{e_1, e_2\}$  olay kümesi üzerinde tanımlanmış  $L_m(G) = \{e_1, e_1e_1, e_2e_1, e_2e_1e_1, \dots\}$  verilebilir.

#### 4.2.2.2 Kilitlenme

Bir G otomatı  $\overline{L_m(G)} \subset L(G)$  özelliğini sağlıyor ise kilitlenmeli,  $\overline{L_m(G)} = L(G)$  özelliğini sağlıyor ise kilitlenmesizdir. Bir otomatın kilitlenebilir olması açmaza veya

kısır döngüye girebileceği anlamına gelir (Cassandras, 1999). Şekil 4.4' de açmaz ve kısır döngü örneği aynı otomatta verilmiştir.

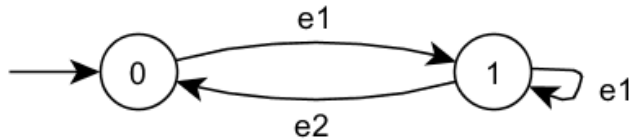


Şekil 4.4 : Açmaz ve kısır döngü örneği.

Başlangıç durumunda sırasıyla  $e_1$  ve  $e_2$  olaylarının meydana gelmesiyle otomat işaretli olmayan ikinci duruma girecek, otomat görevini yerine getirememiş olacak ve kilitlenecektir.  $e_1 e_2 \in L(G)$ ,  $e_1 e_2 \notin \overline{L_m(G)}$ 'dir. Dolayısıyla  $\overline{L_m(G)} \subset L(G)$  özelliği sağlanır ve otomatın kilitlenmeli olduğu gösterilmiş olur. Benzer durum başlangıç durumunda  $e_1, e_3$  olaylarının sırasıyla meydana gelmesiyle oluşmaktadır. Bu sefer otomat işaretli olmayan üçüncü ve dördüncü durum arasında kısır döngüye girecek ve canlı kilitlenme olarak adlandırılan durum ortaya çıkacaktır. Bu tür kilitlenmede de  $\overline{L_m(G)} \subset L(G)$  özelliği sağlanır.

#### 4.2.2.3 Deterministik otomat

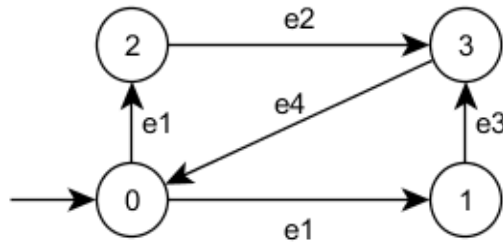
Deterministik otomatta herhangi bir durumda meydana gelen olay, sistemi birden fazla yeni duruma götürmez (Cassandras, 1999). Şekil 4.5'de deterministik otomat örneği verilmiştir.



Şekil 4.5 : Deterministik otomat.

#### 4.2.2.4 Deterministik olmayan otomat

Otomat tanımında, durum geçişleri herhangi bir durumda meydana gelen bir olayın sistemi hangi yeni duruma götüreceğini belirler. Ancak herhangi bir  $x$  durumunda oluşan  $e$  olayı sistemi birden fazla yeni duruma da götürebilir (Cassandras, 1999). Buna otomatlarda izin verilmemesinin sebebi göz ardı edilmek istenmesinden kaynaklanmaktadır (Tiryaki, 2007). Deterministik olmayan otomata örnek Şekil 4.6'da verilmiştir.



Şekil 4.6 : Deterministik olmayan otomat.

Otomat başlangıç durumundayken  $e_1$  olayının meydana gelmesiyle hem iki hem de bir durumuna geçilebilir.

### 4.3 Otomat Gösterimini PLC İle Gerçekleme

Bu bölümde sonlu durumu makinesiyle veya diğer adıyla otomat ile ifade edilen bir tasarımın teknolojik araçlar kullanılarak gerçekleştirilmesi üzerinde durulacaktır. Bu gerçekleştirme için PLC seçilmiştir. PLC'ler 1969 yılında röleli elektriksel kumanda devrelerinin yerine kullanılmak üzere geliştirilmiş mikrokontrolör tabanlı özel sayısal işlemcilerdir. PLC'ler günümüzde endüstriyel kumanda, kontrol uygulamalarında ve seri üretimin yapıldığı hemen her tesiste kullanılmaktadır (Hasdemir, 2008).

PLC'lerin programlanmasında metin ya da grafik tabanlı farklı programlama dilleri kullanılmaktadır. IEC 61131 – 3 standardı tarafından, komut (STL: Statement List), yapısal metin (ST: Structured Text), merdiven mantığı (LAD: Ladder Diagram), fonksiyon blok (FBD: Functional Block Diagram) ve ardışıl fonksiyon gösterimi (SFC: Sequential Function Chart) olmak üzere PLC'ler için beş farklı programlama dili tanımlanmıştır.

Bu çalışmada, gerçekleştirme aşamasında HIMA HIMATRIX hatada güvenli PLC ve bu PLC'ye ilişkin FBD programlama dili kullanılmıştır. 5. Bölümde tanıtılacak olan otomatik olarak anlaşılan sisteminin otomat gösterimini ve sözde kodunu oluşturan program, bu bölümde tanıtılacak olan mantıksal fonksiyonlarla gerçekleştirme yönteminden yararlanarak otomatik olarak sözde kod üretimini sağlamaktadır.

Otomatların PLC'lerde gerçekleştirilebilmesi için iki yöntem tanıtılacaktır. Bu yöntemlerden ilki kurma ve silme komutları ile gerçekleştirme, ikincisi ise bu çalışmada kullanılmış olan mantık fonksiyonları ile gerçekleştirmedir. Ayrıca otomatların PLC'de gerçekleştirilmesindeki sorunlardan söz edilecektir. Bu sorunlardan çıkış etkisinin kesin çözümü ile ilgili bilgiler verilecektir. İlk duruma kurma sorunu ile ilgili önceden önerilmiş olan bir yöntem irdelenecek, bu yöntemin sebep olabileceği problemlerden bahsedilecek ve ilk duruma kurma probleminin çözümü için yeni bir öneri sunulacaktır.

#### **4.3.1 Kurma ve silme komutları ile gerçekleştirme yöntemi**

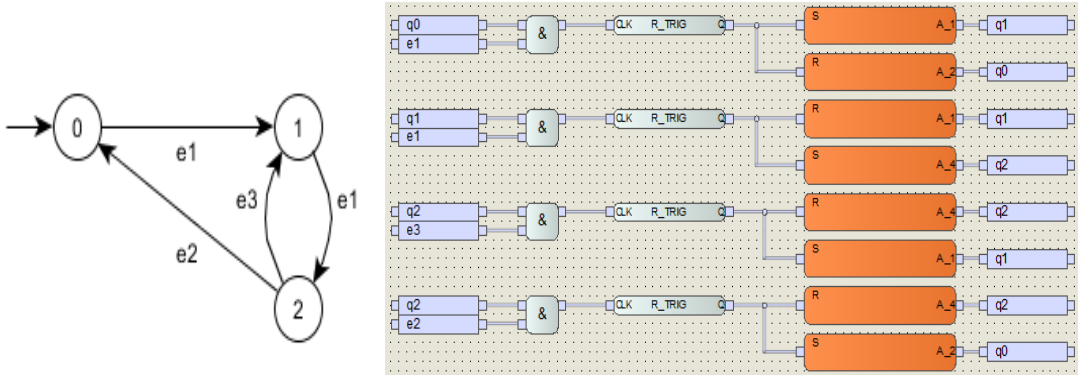
PLC'lerin teorik olarak tanımlanan otomat davranışına en yakın davranışı göstermesi için bazı kurallara ihtiyaç duyulur. Bu bölümde anlatılacak olan gerçekleştirme yöntemi de bu kurallara dayanmaktadır.

Otomat gösterimindeki durumlar ve olaylar bitler ile gösterilebilir. Belli bir duruma denk gelen bir bitin mantıksal bir seviyesinde olması otomatın o durumda olduğu anlamına gelir. Olaylar ise ilgili olayı temsil eden bitin mantıksal olarak bir olmasıyla veya sıfır olmasıyla meydana gelir veya gelmez. Durum geçiş fonksiyonları için ise mantıksal VE ile kurma (SET) veya silme (RESET) komutlarının kullanılması gerekmektedir. Bu yöntemde belirli bir duruma karşılık gelen bit ile o durumdan başka bir duruma geçişe neden olan olayı temsil eden bite mantıksal VE işlemi uygulanır ve sonuca göre yeni durum biti kurulur. Çıkkılan eski durum biti ise silinir (Hasdemir, 2008).

##### **4.3.1.1 Çıkış etkisi sorunu**

Kurma ve silme komutları ile gerçekleştirme yöntemi kolay uygulanabilir olmasına rağmen otomatın yapısına bağlı olarak beklenenden farklı davranışlar sergileyip kodun hatalı şekilde çalışmasına neden olabilir. Bu hatalı davranışın nedeni gerçekleştirilen otomattaki bir takım istenmeyen durum geçişleriyle ilgilidir ve çıkış etkisi

olarak adlandırılır (Hasdemir, 2008). Şekil 4.7’de PLC gerçekleştirilmesi bu yöntem ile yapıldığında çığ etkisi sorununun yaşanacağı bir otomat ve PLC kodu verilmiştir.



**Şekil 4.7 :** Çığ etkisi sorunu yaşanan otomat ve PLC kodu.

Yukarıda verilen otomatın, ilk durumdayken  $e_1$  olayının meydana gelmesiyle kuramsal olarak birinci duruma geçmesi, ikinci kez  $e_1$  olayının meydana gelmesiyle de ikinci duruma geçmesi beklenir. Ancak kurma ve silme komutlarıyla Şekil 4.7’de görüldüğü gibi gerçekleştirildiği takdirde otomat ilk durumdayken  $e_1$  olayının meydana gelmesiyle otomat birinci duruma geçecek ve  $q_1$  biti kurulacaktır, aynı çevrim içinde hem birinci durumu temsil eden  $q_1$  biti hem de  $e_1$  olayını temsil eden bit kurulu olacağından otomat ikinci kez  $e_1$  olayının gerçekleşmesini beklemeden ikinci duruma geçecektir. Bu sebeple hiçbir zaman otomat birinci durumda kalamayacaktır. Bu sorunun bu yöntemde çözümü ile bazı çalışmalar yapılmış ve çeşitli çözüm yolları geliştirilmiştir. Bu çalışmada bu yöntem tercih edilmediği için bu konu ile ilgili detaylı bilgilere burada değinilmeyecektir.

#### 4.3.2 Mantık fonksiyonları ile gerçekleştirme yöntemi

Önceki bölümün sonunda bu çalışmada kurma ve silme yöntemi ile gerçekleştirilmenin tercih edilmediği belirtilmişti. Bunun sebebi ise bu bölümde açıklanacaktır.

Kurma ve silme komutları ile gerçekleştirme yönteminde kullanılan kurma ve silme (Set ve Reset) komutları normal bit işlem komutlarına göre PLC belleğinde daha fazla yer kapladığından, kullanılan program belleği oldukça fazla olmaktadır. Ayrıca kurma ve silme komutlarının kullanılması herhangi bir duruma ilişkin geçiş fonksiyonlarının PLC programında takip edilmesini oldukça zorlaştırmaktadır. Örneğin, Şekil 4.7’de  $q_1$  durumuna ilişkin 1 adet silme ve 2 adet kurma komutu kullanılmıştır. Bu da  $q_1$  durumuna ilişkin geçiş fonksiyonlarını belirleyebilmek için bu 3 adet komuta denk



gelen koşulların incelenip elde edilen bilginin birleştirilmesini gerekli kılar. Otomatın 1 numaralı durumuna giren ve çıkan olaylar arttıkça bu komutların sayısı daha da artacak ve programın takibi gittikçe zorlaşacaktır. Özellikle hata ayıklamada veya programda yapılacak değişikliklerde bu yapı programcıya zorluk çıkaracaktır. Tüm bu sebeplerden ötürü bu çalışmada kurma ve silme komutlarıyla gerçekleştirme tercih edilmemiştir.

Bunun yerine aşağıda tanıtılacak olan mantık fonksiyonları ile gerçekleştirme yöntemi, kurma ve silme komutlarının kullanılmadığı için bellek avantajı sağlayan, çıg etkisi problemi için kesin bir çözüm oluşturan bir yapı sunmaktadır. Ayrıca bu yöntemin, takip edilmesi ve değişikliklerin uygulanması açısından kolay olması diğer avantajlarıdır (Hasdemir, 2008).

Gerçekleme, PLC'nin çalışma prensibine uygun olarak zamanda ayırık anlarda işletilen mantıksal fonksiyonlarla ifade edilecektir. Bir mantıksal  $q$  değişkeninin kurma koşulunu  $S$ , silme koşulu ise  $R$  ile gösterilsin.  $z \in \mathbf{N}$  olmak üzere  $q$  değişkeninin mevcut değeri  $q(z)$ , bir sonraki değeri  $q(z + 1)$  ile gösterilirse

$$q(z + 1) = S \vee q(z) \wedge \bar{R} \quad (4.3)$$

Mantıksal fonksiyonunun  $q$ 'yu  $S$  ile kurduğu  $R$  ile sildiği kolaylıkla görülebilir. Burada ' $\vee$ ' mantıksal VEYA işlemini, ' $\wedge$ ' mantıksal VE işlemini,  $\bar{R}$  ise  $R$ 'nin mantıksal DEĞİL'ini göstermektedir. Önerilen yöntemde Eşitlik 4.3'deki mantık fonksiyonu verilen bir otomatın durum geçiş fonksiyonlarının gerçekleşmesinde kullanılacaktır. Bu nedenle durum geçiş koşullarının Eşitlik 4.3'de verilen kurma ve silme koşullarına karşılık gelecek tanımlara ihtiyaç duyulur. Bu tanımlar aşağıda mantıksal fonksiyonlar kullanılarak verilebilir. Bu mantıksal fonksiyonlarda kullanılacak değişkenlerin bir  $z$  adımıdaki değerleri, durumlar için  $q_i(z)$  ve geçişler için  $e_i(z)$  ifadeleri ile gösterilmektedir. ' $\vee$ ' sembolü ( ya da toplam operatörü  $\sum$  ) mantıksal VEYA, ' $\wedge$ ' sembolü (ya da çarpım operatörü  $\prod$  ) mantıksal VE işlemine karşılık gelmektedir (Hasdemir, 2008).

**Tanım 4.1.**  $q_i$  durumuna ilişkin kurma koşulunun  $z$  adımıdaki mantıksal ifadesi

$$S_i(z) = \sum_{j \in I_{SQ}(i)} q_j(z) \cdot e_{j,i} \quad (4.4)$$

Eşitlik 4.4'deki  $I_{SQ}(i)$  kümesinin elemanları,  $q_i$  durumuna geçişin mümkün olduğu durum indislerine karşılık gelmektedir.  $e_{j,i}$  ise otomatın  $q_j$  durumundan  $q_i$  durumuna geçişine neden olacak tüm olayların mantıksal koşulunu ifade eder.

**Tanım 4.2.**  $q_i$  durumuna ilişkin silme koşulunun  $z$  adımındaki mantıksal ifadesi

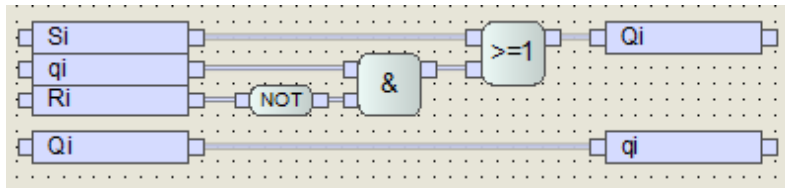
$$\overline{R_i(z)} = \prod_{k \in I_{R\Sigma}(i)} \overline{e_{i,k}} \quad (4.5)$$

Eşitlik 4.5'deki  $I_{R\Sigma}(i)$ ,  $q_i$  durumundan çıkışa neden olan olay indis kümesini,  $e_{i,k}$  ise otomatın  $q_i$  durumundan  $q_k$  durumuna geçişine neden olacak olayın mantıksal koşulunu ifade eder.

Tanım 4.1 ve 4.2 ile  $i$  indisli duruma ilişkin kurma ve silme koşulları tanımlandığından bu duruma ilişkin mantıksal ifade Eşitlik 4.3'den yararlanılarak aşağıdaki şekilde verilebilir:

$$q_i(z+1) = S_i(z) \vee q_i(z) \wedge \overline{R_i(z)} \quad (4.6)$$

Eşitlik 4.6'e ait PLC programı kolaylıkla FBD programlama dili ile elde edilebilir.  $q_i(z+1)$  mantıksal değişkeni  $Q_i$ ,  $q_i(z)$  mantıksal değişkeni  $q_i$  ile gösterilirse Eşitlik 4.6'ya denk gelen PLC programı Şekil 4.8'de verilmiştir. Programın son bölümünde  $Q_i$  değişkeni  $q_i$  değişkenine atanmaktadır. Bu atamaya göre PLC programındaki  $q_i$  mantıksal değişkeni,  $q_i$  durumunun mevcut PLC tarama çevrimindeki;  $Q_i$  mantıksal değişkeni ise bu durumun bir sonraki PLC çevrimindeki gösterimine karşı geldiği şekilde düşünülebilir.



**Şekil 4.8 :** Eşitlik 4.6'ya karşılık gelen FBD dili programı.

**Örnek 4.1.** Şekil 4.7'de verilen otomat ele alınırsa, Tanım 4.1 ve Tanım 4.2'ye göre otomatın  $q_1$  durumu için

$$S_1(z) = \sum_{j \in I_{SQ}(1)} q_j(z) \cdot e_{j,1} = q_0(z) \cdot e_1 + q_2(z) \cdot e_3 \quad (4.7)$$

$$\overline{R_1(z)} = \prod_{k \in I_{R_1}(1)} \overline{e_{1,k}} = \overline{e_1} \quad (4.8)$$

Eşitlikleri elde edilir. Bu ifadeler Eşitlik 4.6'da yerine koyulursa  $q_1$  durumuna ait mantıksal ifade

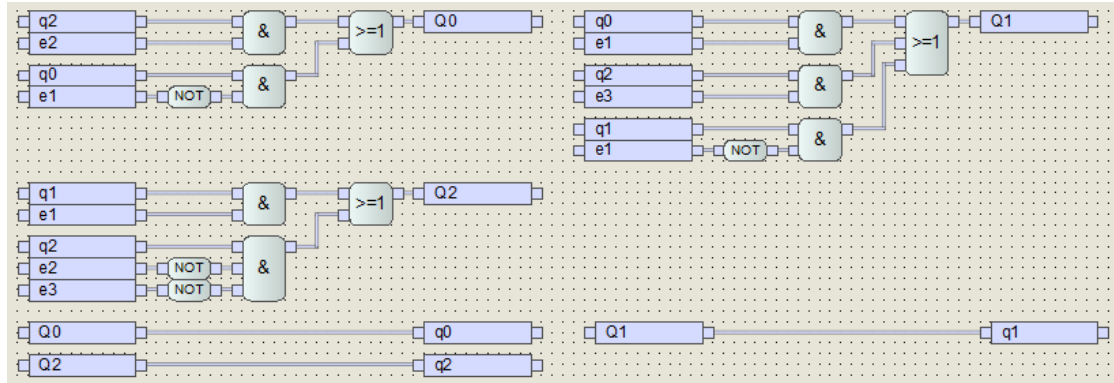
$$q_1(z+1) = q_0(z) \wedge e_1 \vee q_2(z) \wedge e_3 \vee q_1(z) \wedge \overline{e_1} \quad (4.9)$$

Aynı yöntemle  $q_0$  ve  $q_2$  durumlarına ait mantıksal fonksiyonlar

$$q_0(z+1) = q_2(z) \wedge e_2 \vee q_0(z) \wedge \overline{e_1} \quad (4.10)$$

$$q_2(z+1) = q_1(z) \wedge e_1 \vee q_2(z) \wedge \overline{e_2} \wedge \overline{e_3} \quad (4.11)$$

olarak verilir. Eşitlik 4.9, 4.10 ve 4.11 ele alınarak oluşturulan PLC programı Şekil 4.9'da verilmiştir. Bu programda olaylara karşılık gelen bitlerin işaretlerin yükselen ya da düşen kenarları ile elde edilmiş olduğu kabul edilmektedir. Eğer bir duruma getiren ve çıkaran olaylar farklı ise ( $e_{ji} \neq e_{ik}$ ) düşen ve çıkan kenarlar kullanılmayabilir.



Şekil 4.9 : Örnek 4.1'e ilişkin FBD dilinde PLC programı.

4.3.1.1 bölümünde ele alınan ve incelenen Şekil 4.7'deki otomatta kurma ve silme yöntemi ile gerçekleştirildiğinde çığ etkisi sorunu yaşanmaktaydı. Bu bölümde tanıtılan mantık fonksiyonları ile gerçekleştirilme de ise çığ etkisi sorunu aşılmaktadır. Şekil 4.9'daki PLC programı incelendiğinde,  $q_0$  durumundan  $q_1$  durumuna geçişin gerçekleştiği fonksiyon gösteriminde  $e_1$  olayının gerçekleşmesiyle  $Q_1$  bellek bitinin mantıksal 1 seviyesine kurulduğu görülmektedir.  $Q_2$ 'nin kurulmasını sağlayan fonksiyon gösteriminde  $Q_2$  bellek bitinin kurulması  $q_1$  ve  $e_1$  değişkenlerinin mantıksal 1 seviyesinde olmasına bağlıdır. Otomatın 1 numaralı durumu için  $q_1$  bellek biti,  $Q_1$  bellek biti olmak üzere iki farklı değişken kullanılması nedeniyle  $Q_2$

bellek bitinin değeri, programın bir önceki satırında değeri belirlenen  $Q_1$  değişkeninden etkilenmez. Böylece otomat 0 numaralı durumda iken  $e_1$  olayının meydana gelmesi tek çevrim içinde  $Q_2$  bellek bitinin mantıksal 1 değerine kurulmasının, ya da başka bir değişle  $e_1$  etkisi probleminin önüne geçilmiş olur.

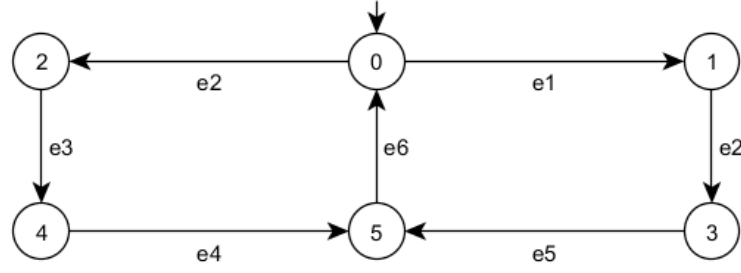
Şekil 4.9’da da görüldüğü üzere bu yöntemde hangi durumunun ne zaman mantıksal 1 ve ne zaman mantıksal 0 olacağı kolaylıkla takip edilebilmektedir. Bu da daha önceden de bahsedildiği üzere yöntemin uygulanabilirliğini arttırmaktadır. Örneğin,  $Q_1$  bellek bitinin, otomat  $q_0$  durumundayken  $e_1$  olayının meydana gelmesiyle veya  $q_2$  durumundayken  $e_3$  olayının meydana gelmesiyle mantıksal 1 olacağı,  $q_0$ ,  $e_1$  ve  $q_2$ ,  $e_3$  bitlerinin ayrı ayrı VE bloğuna sokulup VEYA’lanmasıyla gösterilmiş ve kolaylıkla anlaşılabilir. Aynı şekilde  $Q_1$  bellek bitinin ne zaman silineceği ise  $q_1$  ve  $\bar{e}_1$  bitlerinin mantıksal VE bloğuna sokulmasıyla belirtilmiştir. Bu üç VE bloğunu da VEYA bloğuna giriş olarak alıp Eşitlik 4.9’un PLC gerçekleştirilmesi yapılmış olur. Kurma ve silme komutları ile aynı otomatın PLC gerçekleştirilmesi hatırlanacağı üzere Şekil 4.7’de verilmişti ve her durum için programın farklı satırlarının takip edilmesinin gerekli olduğu ve hangi durumda kurulup hangi durumda silindiğinin anlaşılmasının oldukça zor olduğunu vurgulanmıştı. Bu da programı okumayı, hata ayıklamayı ve değişiklik yapmayı mantık fonksiyonları ile gerçekleştirme yöntemine göre oldukça güçleştirmektedir.

*İlk durumun programlanması:*

Gerçekleme işleminin tamamlanabilmesi için durum geçişlerinin yanı sıra ilk durumun da programlanması gerekmektedir. Bunun için “otomat hiçbir durumda değil ise ilk durumdadır” mantığı ile ilk duruma kurma problemi aşılabılır (Hasdemir, 2008). Bu sözel ifadenin gerçekleştirilmesinde kullanılacak mantıksal ifadelerden biri aşağıdaki gibi verilebilir.

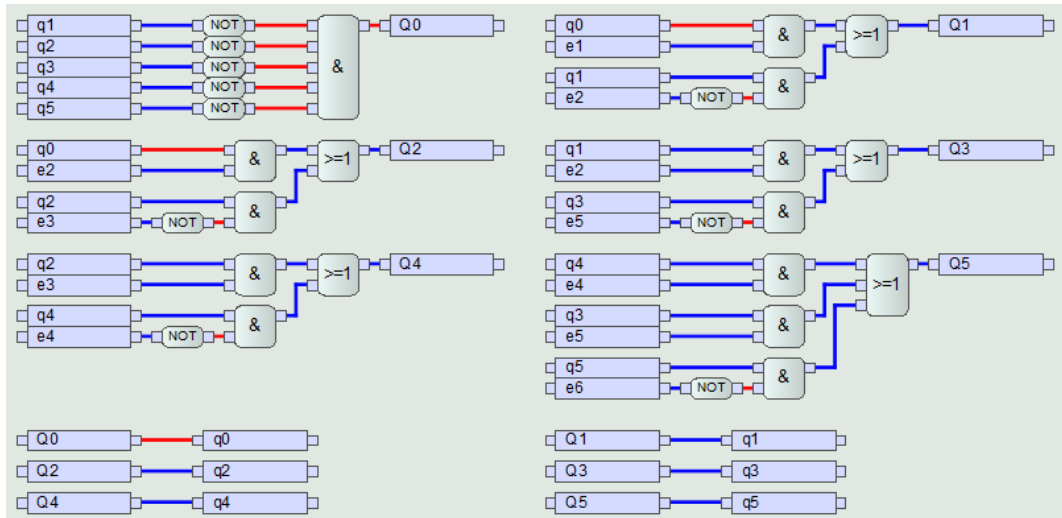
$$q_0(z + 1) = \prod_{i \in I_Q \setminus i_0} \overline{q_i(z)} \quad (4.12)$$

Eşitlik 4.12’de  $I_Q$ , durum indis kümesini,  $i_0$  ise ilk durumun indis numarasını göstermektedir. Buna göre, Şekil 4.10’da verilen altı durumlu otomatın ilk duruma ilişkin mantıksal fonksiyonu  $Q_0 = \bar{q}_1 \wedge \bar{q}_2 \wedge \bar{q}_3 \wedge \bar{q}_4 \wedge \bar{q}_5$  şeklinde olacaktır.



Şekil 4.10 : Altı durumlu otomat.

Bu otomatın gerçekleştirilmesine ait simülasyon görüntüsü ve ilk duruma kurulması Şekil 4.11’de verilmiştir.

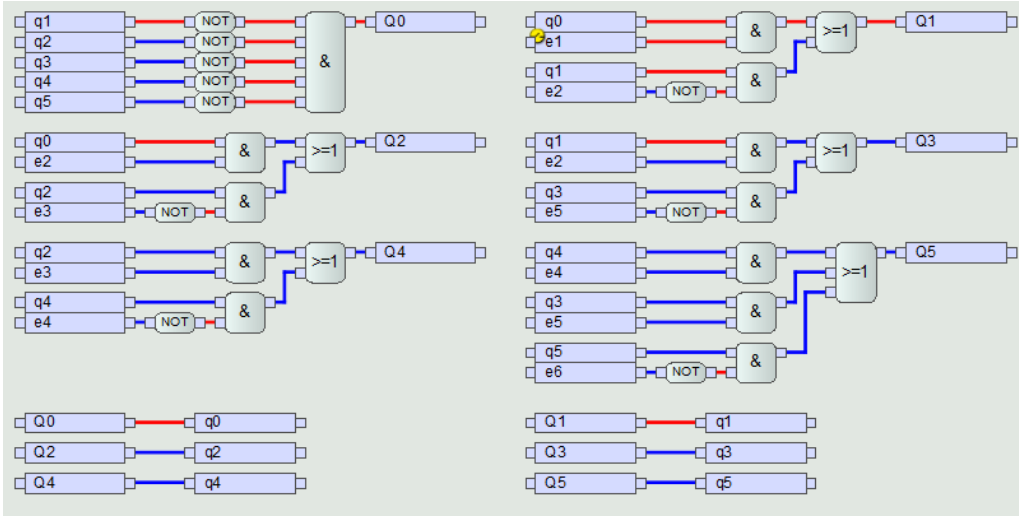


Şekil 4.11 : Otomatın ilk duruma kurulması.

İlk duruma kurma probleminin aşılabilmesi için izlenen bu yöntem genel olarak başarılı olsa da otomatı ilk durumdan çıkaran olay meydana geldiğinde, otomat bir çevrim süresi boyunca hem ilk durumda hem de ilk durumdan geçiş yaptığı durumda kalacaktır. Bu da otomatın bir çevrim boyunca aynı anda iki durumda birden kalması anlamına gelmektedir. Benzer durum, otomatı son durumdan başlangıç durumuna geçiren olayın meydana gelmesinde de oluşacaktır. Bu sefer de otomat bir çevrim süresi boyunca hiçbir durumda olmayacak ve bu çevrimin ardından ilk duruma kurulacaktır. Bu sorunun irdelenmesi ve yol açabileceği sorunları tartışmak adına Şekil 4.10’daki otomat örnek olarak verilebilir.

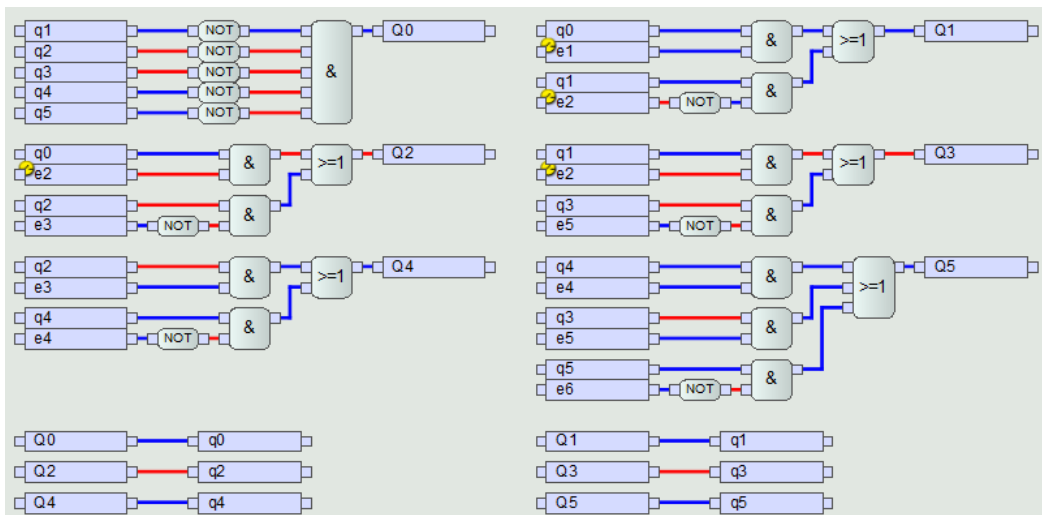
Şekil 4.11’de görüldüğü gibi otomat ilk durumda kurulu iken  $e_1$  olayı meydana geldiğinde  $Q_1$  bellek biti mantıksal 1 seviyesine çekilecek, bu bellek biti programın sonunda  $q_1$  bellek bitine atanacak ve bir program çevrimi tamamlanacaktır. Ancak PLC bir sonraki program çevrimine kadar ilk duruma kurma kodunu işletmeyeceği

için otomat 0 numaralı durumdan çıkamayacak ve bu çevrimde  $Q_0$ ,  $q_0$  bellek bitleri ile  $Q_1$ ,  $q_1$  bellek bitleri mantıksal 1 seviyesinde olacaklardır. Açıklanmaya çalışılan bu durum Şekil 4.12’ de verilmiştir.



**Şekil 4.12 :** Otomatın hem 0 numaralı durumda hem de 1 numaralı durumda kalması.

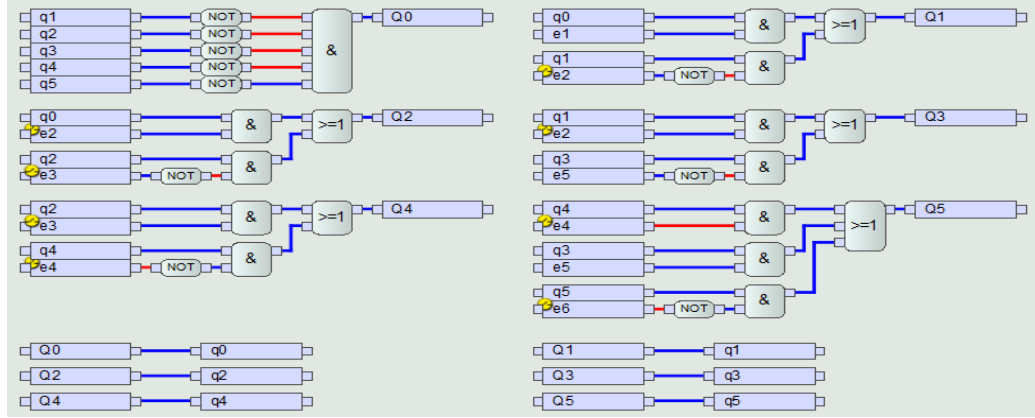
Şekil 4.12’de de görüldüğü üzere ilk durumda kurulu iken  $e_1$  bellek biti mantıksal 1 yapıp program bir çevrim çalıştırıldığında, otomat hem 0 numaralı durumda hem de 1 numaralı durumda kalmaktadır. Bu çevrimde  $e_2$  olayının meydana geldiği varsayılırsa Şekil 4.13’de verilen durum ortaya çıkacak ve otomat hem 2 numaralı duruma hem de 3 numaralı duruma geçecek ve ilgili olaylar meydana gelene kadar bu durumlarda kalmaya devam edecektir.



**Şekil 4.13 :** Otomatın hem 2 numaralı durumda hem de 3 numaralı durumda kalması.

Karşılaştığımız bu durum ise bu düşünceye ters düşmektedir ve tasarımcının önceden düşünmediği bir sorunla karşılaşmasına yol açabilir.

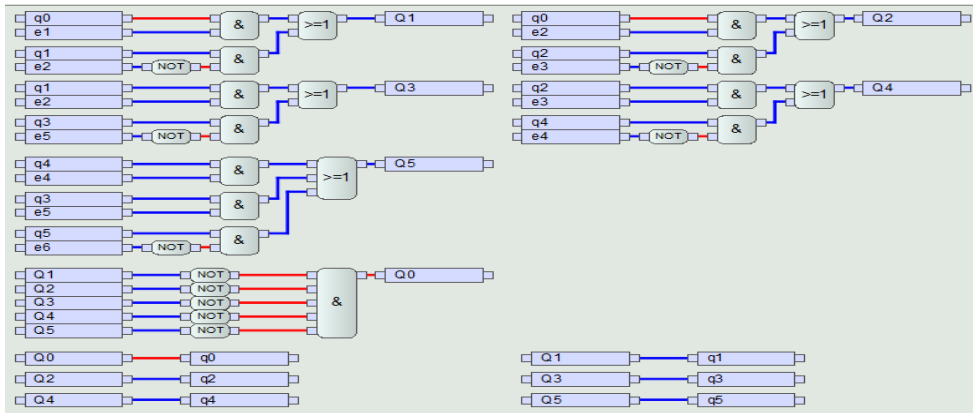
Daha önceden de bahsedilen diğer problem ise 5 numaralı durumdan başlangıç durumuna geçiren olayın meydana gelmesinde otomatın bir çevrim boyunca hiçbir durumda kalmaması idi. Bu durumda Şekil 4.14’ de gösterilmiştir.



**Şekil 4.14 :** Otomat bir çevrim boyunca hiçbir durumda değil.

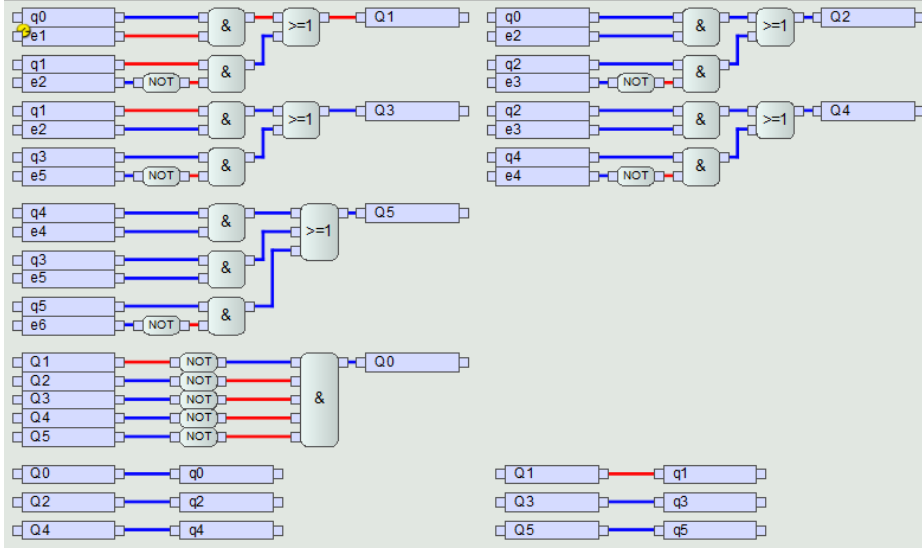
Belirtilen bu sorunların giderilebilmesi için önerilen çözüm şu şekildedir: İlk duruma kurma işlemi  $Q_0 = \overline{Q_1} \wedge \overline{Q_2} \wedge \overline{Q_3} \wedge \overline{Q_4} \wedge \overline{Q_5}$  mantıksal ifadesi ile verilir ve Şekil 4.15’de gösterildiği gibi geçiş fonksiyonlarının altına yazılır ise sorun çözülmüş olur ve otomat her program çevriminde sadece tek durumda kalacak şekilde gerçekleşmiş olur. En genel haliyle önerilen ilk duruma kurma ifadesi aşağıdaki gibi verilebilir.

$$q_0(z + 1) = \prod_{i \in I_Q \setminus i_0} \overline{q_i(z + 1)} \quad (4.13)$$



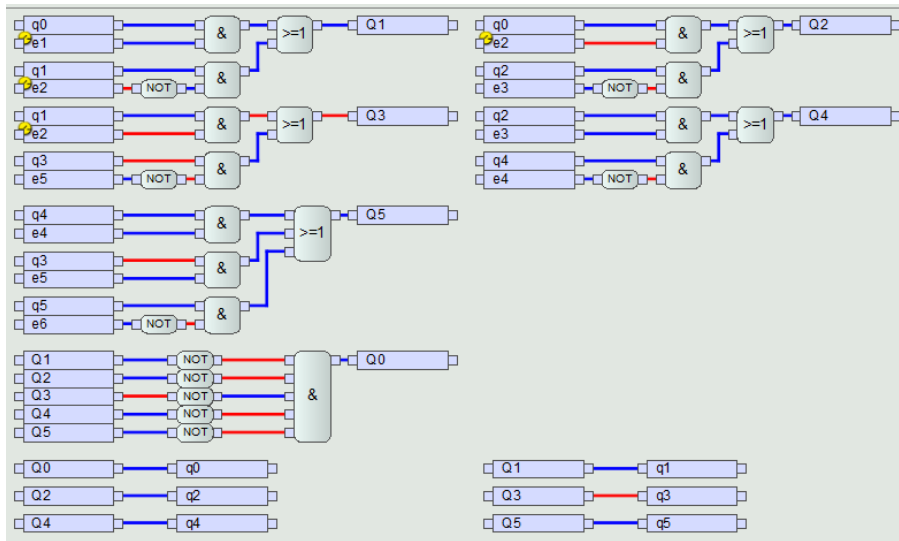
**Şekil 4.15 :** İlk duruma kurma problemine önerilen yeni çözüm yöntemi ile yazılmış program.

Gerçekleme önceki şekliyle yapıldığında soruna yol açan olayları aynı sırada meydana getirerek, önerilen çözümü içeren Şekil 4.15'deki gerçekleştirme yöntemi için deneyelim ve sorunun çözülüp çözülmediğini irdeleyelim.  $e_1$  bellek biti mantıksal 1 yapıp, program bir çevrim çalıştırıldığında otomatın durumu Şekil 4.16'da verilmiştir.



Şekil 4.16 : Yeni ilk duruma kurma yöntemi ile otomatın 1 numaralı duruma geçişi.

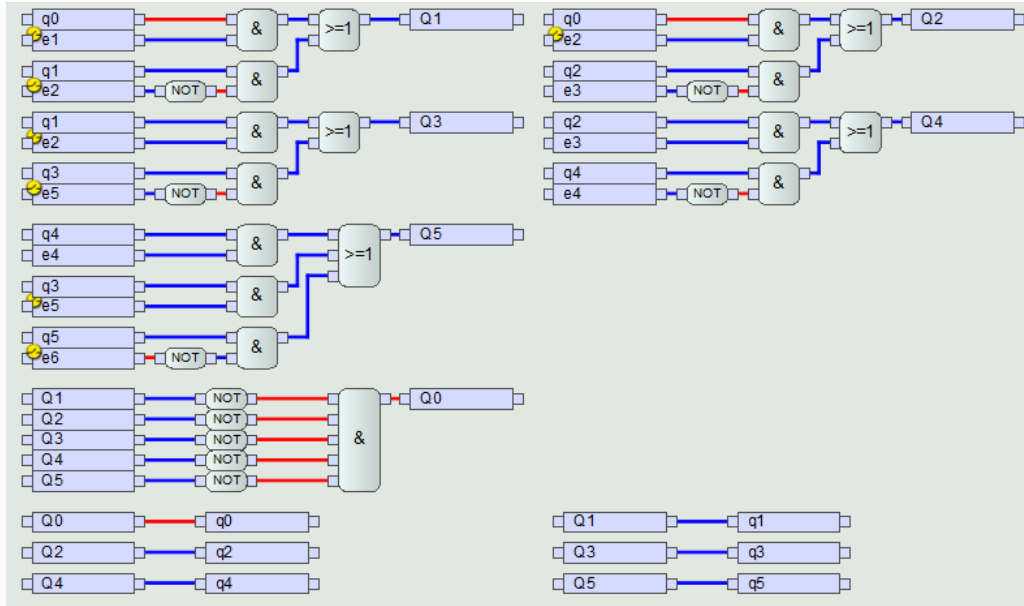
Şekil 4.16'da da görüldüğü üzere otomat tek çevrim içinde ilk durumdan çıkıp, 1 numaralı duruma geçmiş ve aynı çevrim içinde iki durumda kalmamıştır. Aynı anda iki durumda birden kalmadığından bu esnada  $e_2$  olayı meydana gelse dahi Şekil 4.17'de gösterildiği üzere sorun yaşanmayacak ve otomat beklenildiği üzere 3 numaralı duruma geçecektir.



Şekil 4.17 : Yeni ilk duruma kurma yöntemi ile otomatın 3 numaralı duruma geçişi.



5 numaralı durumdan başlangıç durumuna dönerken bir çevrim boyunca hiçbir durumda olmama sorunu da Şekil 4.18’de gösterildiği üzere aşılmıştır.



Şekil 4.18 : Yeni ilk duruma kurma yöntemi ile 5. durumdan ilk duruma geçiş.

Şekil 4.10’da aynı durumdan çıkışı sağlayan geçişlerin aynı anda mantıksal 1 olamayacağı kabul edilerek otomat tasarlanmıştır. Örneğin otomatı 0 numaralı durumdan çıkaran  $e_1$  ve  $e_2$  geçişlerinin aynı anda gerçekleşmediği varsayılmıştır. Aksi halde böyle bir tasarımda otomat ilk durumdayken  $e_1$  ve  $e_2$  geçişlerinin aynı anda tetiklenmesiyle otomat 1 ve 2 numaralı durumlara geçecektir. Otomatın tasarımı aşamasında bu durumlara dikkat edilmelidir. Bu tarz durumların önüne geçmek için iki yöntem önerilebilir.

Bu yöntemlerden ilki otomatın tasarımı aşamasında aynı anda tetiklenerek otomatı birden fazla duruma geçirebilecek geçişler belirlenerek bu geçişlerden birine öncelik verilebilir. Örneğin, Şekil 4.10’da verilen otomatın ilk durumdan çıkmasını sağlayan  $e_1$  ve  $e_2$  geçişleri aynı anda tetiklenebilecek geçişler olsun.  $e_1$  geçişine öncelik verilerek 0 numaralı durumdan 2 numaralı duruma geçiş  $e_2 \wedge \bar{e}_1$  şeklinde yazılabilir. Böylelikle otomat 0 numaralı durumdayken  $e_1$  ve  $e_2$  geçişleri aynı anda tetiklense dahi  $e_1$  geçişine öncelik verilmiş olduğundan otomat 1 numaralı duruma geçecektir.

İkinci yöntem ise böyle bir durumu tasarım aşamasında göz önünde bulundurup gerekli geçişlere öncelik vererek değil, kodlama aşamasında engellemektedir. Ayrıca bu yöntemde dış etkilerden kaynaklanabilecek beklenmedik bit değişimlerinin,

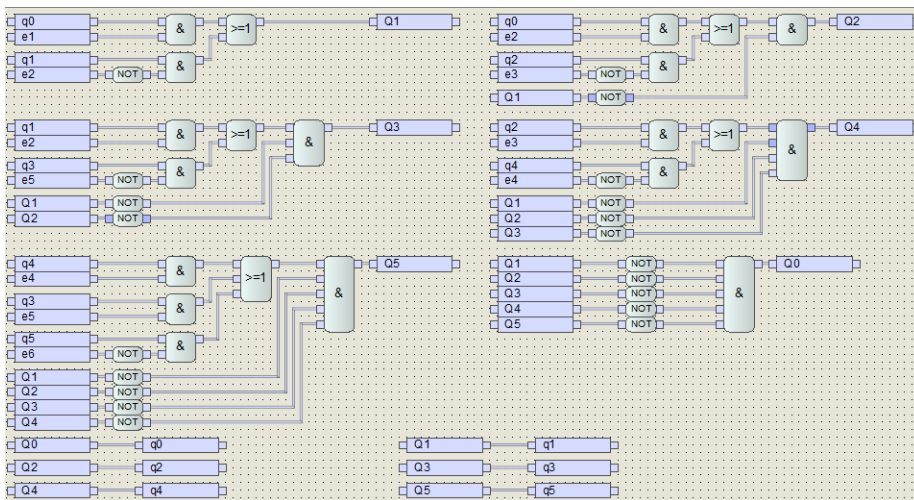
otomatı birden fazla duruma dallandırması da engellenmektedir. Otomatın her zaman için sadece bir durumda kalmasını garanti edeceği düşünülen yeni bir yaklaşım genel haliyle  $n$  durumlu bir otomat için 0 ilk durumun indis numarası olmak üzere aşağıdaki gibi önerilebilir.

$$q_1(z + 1) = S_1(z) \vee q_1(z) \wedge \overline{R_1(z)} \quad (4.14)$$

$$q_m(z + 1) = \prod_{p=1}^{m-1} \overline{q_p(z + 1)} \cdot (S_m(z) + q_m(z) \cdot \overline{R_m(z)}) \quad (4.15)$$

Eşitlik 4.15’de  $m$ , 0 ve 1 numaralı durumlar dışındaki durumların indis numaralarını göstermektedir. PLC gerçekleştirilmesinde Eşitlik 4.14 ve 4.15 ile ifade edilen geçiş fonksiyonlarının altına önceden de ifade edildiği gibi Eşitlik 4.13’de verilen ifade ilk duruma kurmak için eklenmelidir.

Eşitlik 4.15’de verilen ifadede kodladığımız her yeni durumda, önceki durumların mantıksal DEĞİL’i alınmaktadır. Bu şekilde, otomatın bir önceki durumdan çıkmadan yeni bir duruma geçmemesi önerilen bu mantıksal ifadeyle sağlanmaktadır. Böylece otomatın aynı anda birden fazla durumda olmaması garanti altına alınmış olur. Ancak mantıksal ifadenin uzunluğu durum sayısına bağlı olarak artacağından dolayı, durum sayısının fazla olduğu otomatlarda kod boyutunun artması da kaçınılmaz olmaktadır. Güvenliğin ön planda olduğu ve bellek problemi yaşanmayacak programlarda bu gerçekleştirme yönteminin kullanılması önerilebilir. Şekil 4.10’daki otomatın Eşitlik 4.14 ve Eşitlik 4.15’de verilen mantıksal ifade kullanılarak PLC gerçekleştirilmesi yapılırsa Şekil 4.19’da verilen program elde edilmektedir.



Şekil 4.19 : Eşitlik 4.14 ve Eşitlik 4.15’e göre düzenlenmiş PLC programı.

Bu şekilde otomat 0 numaralı durumda iken  $e_1$  ve  $e_2$  geişleri aynı anda tetiklendiğinde önce  $Q_1$  kurulacak ve  $Q_2$ 'nin kurulması için gerekli şartlar incelendiğinde  $q_0$  ve  $e_2$  bitleri mantıksal 1 olmasına rağmen  $Q_1$ 'in DEĞİL'i de sorgulandığı için  $Q_2$  bellek biti kurulamayacaktır. Böylece otomat yalnızca 1 numaralı duruma geçecektir.



## **5. OTOMATİK ANKLAŞMAN ALGORİTMASI VE KODU ÜRETME YÖNTEMİ**

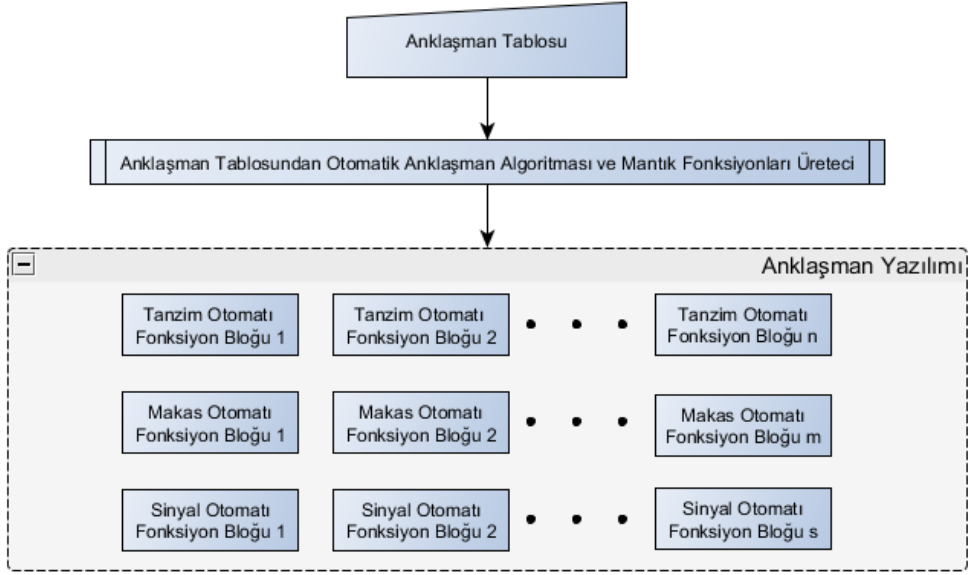
Bu çalışmada Microsoft Visual Basic ortamında tasarlanan bir program kullanılarak otomatik olarak anlaşıman algoritması ve bu algoritmanın sözde kodlarının üretilmesi amaçlanmıştır. Program tasarlanırken, tasarlanan programın otomatik anlaşıman algoritması üretebilmesi için bir gösterim biçimine ihtiyaç duyulmuştur. Bu gösterim biçimi olarak otomatlar kullanılmıştır. Otomatın PLC gerçekleştirmesinin otomatik olarak yapılabilmesi için de programın kullanabileceği bir gerçekleştirme yöntemi olarak önceki bölümde açıklanan mantık fonksiyonları ile gerçekleştirme yöntemi tercih edilmiştir.

### **5.1 Otomatik Anlaşıman Algoritması ve Kodu Üretim Yazılımının Tanıtılması**

Bu bölümde tasarlanan otomatik anlaşıman algoritması ve kodu üretim programı hakkında bilgi verilecektir. Bölüm 3.4'de de bahsedilmiş olduğu üzere anlaşıman algoritması tasarımının ilk adımı ilgili hat için anlaşıman tablosunun oluşturulmasıdır. Kullanıcının anlaşıman tablosunu oluşturmasıyla program için gerekli olan giriş bilgileri de oluşturulmuş olur. Kullanıcının oluşturduğu anlaşıman tablosundaki bilgileri program giriş olarak alıp sistemin otomat gösterimini ve PLC için gerçekleştirilen kodu kullanıcıya çıktı olarak verecektir.

Otomat gösterimindeki en önemli sorun karmaşık sistemler için durum patlaması olayıdır. Bu sorunu gidermek için çözüm, sistemi modüllere ayırarak tasarım yapmaktır. Bu nedenle tasarlanan programda da kullanıcıya modüler bir programlama imkanı sunulmaktadır. Anlaşıman algoritması üç bölüme ayrılarak tanzim, makas ve sinyal modülleri için programda bir menü sunularak her biri için ayrı bir tasarım imkanı sunulmuştur. Böylece durum patlaması gibi bir problemle karşılaşılması ve daha basit bir çözüm yolu sunulması hedeflenmiştir. Ayrıca bu şekilde bir yaklaşım kullanıcıya PLC kodlarını oluştururken de fayda sağlayacaktır. Menüden seçilecek tanzim, makas veya sinyal otomatı tasarımı opsiyonlarından her biri için ayrı bir otomat oluşturulacaktır. Bu otomatları gerçekleyen PLC kodları da

diğer otomatlardan ayrı olacak ve tüm bu farklı otomatları fonksiyon blokları şeklinde yazabilme imkanı doğacaktır. Açıklanan bu yapı Şekil 5.1’de gösterilmiştir.

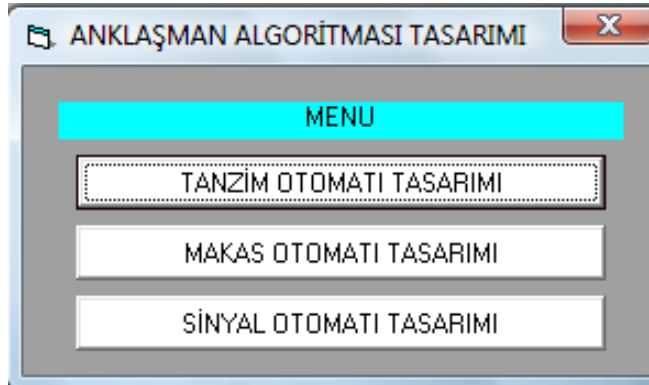


Şekil 5.1 : Geliştirilen yazılımın blok şeması.

Şekil 5.1’de;  $n$ , anlaşman tablosundaki güzergah sayısını,  $m$  motorlu makas sayısını,  $s$  ise sinyal sayısını vermektedir.

Bu şekilde bir tasarım daha önceden de bahsedildiği üzere kullanıcıya modüler bir yapıda etkin ve daha az hata ile programlama imkanı verecektir. Programlama esnasında oluşabilecek hatalarda böylelikle daha kolay ayıklanabilecektir. Böyle bir program oluşturma ihtiyacı da bu sebeple doğmuştur.

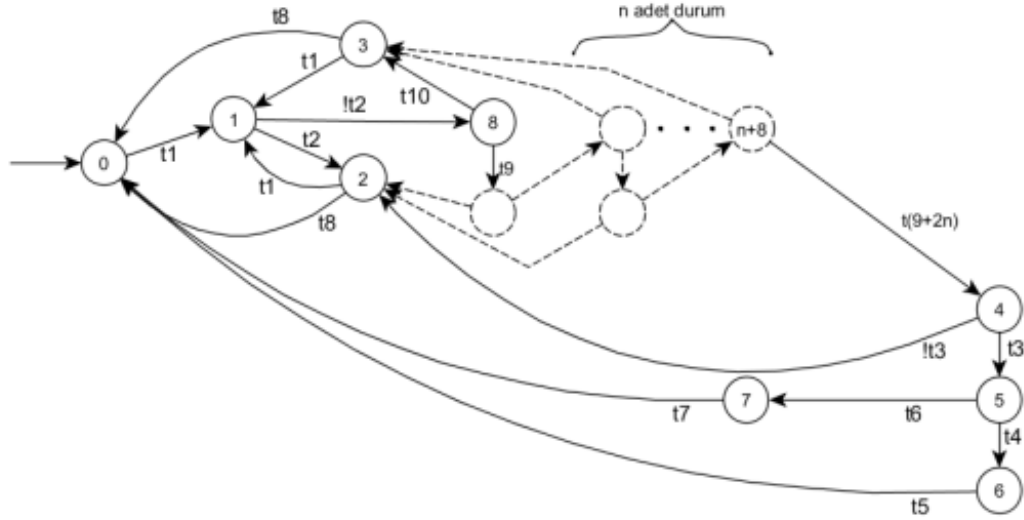
Önceki paragrafta bahsedilen tanzim, makas ve sinyal otomatı tasarımı seçeneklerini sunan yazılım menüsü aşağıdaki şekilde verilmiştir.



Şekil 5.2 : Yazılım menüsü.

### 5.1.1 Tanzim otomatı

Tanzim otomatı, ilgili güzergahtaki ray devrelerini, rota kilitlerini, çakışan güzergahların tanzimli olup olmadıklarını ve tanzim taleplerini kontrol edip, tanzime aykırı bir durum tespit etmemesi durumunda makasların güzergahın gerektirdiği konumları alması için makas otomatlarına sırayla komut gönderir. Makaslar çevrilirken olası makas hatalarını da kontrol edilip makasların istenen konumları aldıklarından emin olunur. Son aşamada ray devreleri tekrar kontrol edilip sinyallerin yakılması için sinyal otomatına bildirimde bulunulur. Sinyalden beklenen indikasyon belli bir süre içinde alınmaz ise güzergah iptal edilir. Tanzim işlemi sırasında yapılan kontrollerin tanzime engel bir durum oluşturması durumunda tanzim talebi reddedilir. Tanzim otomatına ilişkin gösterim Şekil 5.3’de verilmiştir.



Şekil 5.3 : Tanzim otomatı.

Tanzim otomatı, güzergah tanzim talebiyle birlikte başlangıç durumundan (Durum 0) çıkar. İlgili güzergahın tanzim edilebilmesi için gerekli olan şartların sağlanıp sağlanmadığının kontrolünü yapar. Bu şartların sağlanması durumunda tanzim otomatı ilgili güzergahı kilitler (Durum 8) ve güzergahta bulunan ilgili makaslara güzergahın gerektirdiği konumlara çevrilmeleri için, makasların bölgedeki konumlarına uygun bir sırada gerekli komutları (Durum 8, 9, ....., n+8) yollar. Eğer tanzim şartları sağlanmaz veya makaslardan en az biri hataya düşer ise tanzim talebi reddedilir (Durum 2, 3). İlgili güzergahtaki bütün makaslar uygun konuma gelip kilitlendikten sonra güzergahtaki son kontrolleri (Durum 4) yapar ve bu kontrollerin

uygun bulması durumunda bu güzergahı tanzim eder (Durum 5). Tren tanzim edilen güzergaha girince durum atlanıp 6. duruma geçilir. Güzergah tanzim edildikten sonra eğer tren bu güzergaha girmemişse, güzergah iptal talebi gelmesi durumunda güzergah sinyali kırmızıya çekilir (Durum 7). Güzergah ve makas kilitleri önlem amaçlı olarak 3 dakika sonra açılır (Durum 0). Aynı prosedür güzergah tanzim edildikten sonra ilgili güzergah sinyalinin hataya düşmesiyle de gerçekleşir. Güzergah tanziminin sonlanması için trenin güzergahtan çıkması, iptal talebinin gelmesi veya sinyalin hataya düşmesi gerekmektedir. Tanzimin sonlanmasıyla birlikte tanzim otomatı başlangıç durumuna döner ve yeni bir güzergah tanzim talebini değerlendirmek için hazır duruma gelir (Türk, 2010).

Her bir güzergah için yapılan kontroller standarttır, dolayısıyla kontroller için kullanılan geçiş ve durum sayıları her güzergah için aynıdır. Güzergahlar arasındaki farkı belirleyen unsur ise güzergahta bulunan makas sayısıdır. Makaslar motorlarının güç kaynağından fazla akım çektikleri için aynı anda çalışmaları istenmez. Bu nedenle her makas için bir durum eklenmelidir, yani makas sayısı arttıkça güzergahın tanzim otomatındaki durum sayısı da artmaktadır. Şekil 5.3’de noktalı olarak gösterilen durumlarda güzergahtaki makas sayısına göre artacak durumları göstermektedir (Türk, 2010).

Tanzim otomatı geçişleri aşağıdaki gibi tanımlanmıştır.

$$t1 = rtx \uparrow \quad (5.1)$$

$x$  güzergah numarası,  $rtx$  ise  $x$  numaralı güzergah tanzim talebi göstermektedir.

$$t2 = \left( \bigvee_{i=1}^m rd(i) \right) \vee \left( \bigvee_{j=1}^p rk(j) \right) \vee \left( \bigvee_{k=1}^r rt(k)_{istek} \right) \quad (5.2)$$

5.2 numaralı denklemde;  $m$ , güzergahta bulunan, istasyon yolu dışındaki ray devreleri sayısını,  $p$ , ilgili güzergahtaki rota (güzergah) kilidi sayısını,  $r$  ise ilgili güzergahla çakışan güzergah sayısını gösterir.  $rd(i)$  güzergahta bulunan ray devrelerinin durumunu,  $rk(j)$  güzergah kilitlerini ve  $rt(k)_{istek}$  ise çakışan güzergahların tanzimli olduklarını veya tanzim taleplerini gösterir. Çakışan güzergahın anlamı, iki zıt yöndeki güzergahın son ray devrelerinin ortak olmasıdır. Birden fazla ortak ray devreleri bulunan ve zıt yönlerdeki güzergahlar çakışan güzergah olarak isimlendirilmemiştir; çünkü tanzim talebinde bulunulan güzergah ile birden fazla ortak ray devreleri bulunan ve zıt yöndeki güzergahların tanzimi veya



tanzim talepleri zaten  $rk(j)$  olarak isimlendirilmiş rota kilitleri ile tespit edilebilmektedir. Ancak son ray devreleri ortak olan ve zıt yönlerdeki güzergahlar aynı rota kilitlerine sahip olmadığından bu güzergahların tanzimli olmalarını veya tanzim taleplerinin değerlendirilmekte olduğunu belirlemek için  $rt(k)_{istek}$  biti tanımlanmıştır.

$$t3 = \bigwedge_{i=1}^m rd(i) \quad (5.3)$$

$$t4 = \overline{rd1} \quad (5.4)$$

$rd1$  güzergahın ilk ray devresi iken

$$t5 = \overline{rd(m)} \wedge rd(m-1) \quad (5.5)$$

$rd(m)$  güzergahta bulunan son ray devresini,  $rd(m-1)$  güzergahtaki sondan bir önceki ray devresini temsil eder.

Güzergahta varılacak yolda ray devresi yok ise  $t_5$  geçişini aşağıdaki gibi düzenlemek gerekir.

$$t5 = rd(m) \wedge rd(m-1) \quad (5.6)$$

$$t6 = (rd1 \wedge rtx_{iptal}) \vee s_x_{hata} \quad (5.7)$$

$rtx_{iptal}$ ,  $x$  güzergahı tanzim iptal talebini,  $s_x_{hata}$  ise güzergah sinyal hatasını belirtir.

$$t7 = 3 \text{ dakika bekleme (7. durumda)} \quad (5.8)$$

$$t8 = 10 \text{ saniye bekleme (2. ve 3. durumda)} \quad (5.9)$$

$$t9 = m_1_{kilit} \quad (5.10)$$

$m_1_{kilit}$  güzergahtaki ilk makasın kilitlendiğini gösterirken

$$t10 = m_1_{hata} \quad (5.11)$$

$m_1_{hata}$  güzergahtaki ilk makasın hataya düştüğünü gösterir.

$$t(9 + 2n) = m_{(n)}_{kilit} \quad (5.12)$$

$m_{(n)}_{kilit}$  güzergahtaki  $n$  numaralı makasın kilitlendiğini gösterirken

$$t(10 + 2n) = m_{(n)}_{hata} \quad (5.13)$$

$m_{(n)}_{hata}$  güzergahtaki  $n$  numaralı makasın hataya düştüğünü gösterir.

Tanzim otomatu tasarımı için geliştirilen yazılımın kullanıcıya sunduğu arayüz Şekil 5.4'de verilmiştir.

The image shows a software window titled "Tanzim Otomatu Tasarımı". It has a light gray background and a standard Windows-style title bar. The interface is organized into several sections, each with a label and a set of controls:

- Rota Adı**: A text input field.
- Sinyal Adı**: A text input field.
- Ray Devresi**: Contains two cyan buttons: "Ray Devresi Ekle" and "Ray Devresi Çıkar". To the right, there are two checkboxes: "Varılacak yolda ray devresi yok" and "Varılacak yol istasyon yolu", followed by a small text input field.
- Makas**: Contains two cyan buttons: "Makas Ekle" and "Makas Çıkar", and a small text input field.
- Çakışan Rota**: Contains two cyan buttons: "Çakışan Rota Ekle" and "Çakışan Rota Çıkar", and a small text input field.
- Rota Kilidi**: Contains two cyan buttons: "Rota Kilidi Ekle" and "Rota Kilidi Çıkar", and a small text input field.
- Tanzim Otomatu**: A large cyan button centered at the bottom of the window.

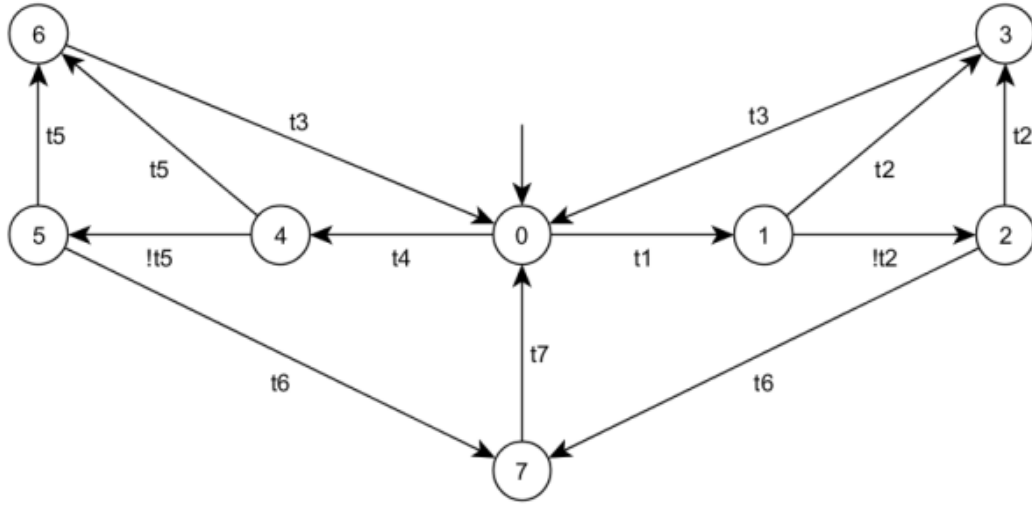
Şekil 5.4 : Tanzim otomatu tasarımı arayüzü.

Rota adı bölümüne tanzim otomatu tasarımı yapılacak güzergahın adı yazılmalıdır. Sinyal adı ise ilgili güzergah için anlaşılan tablosunun ilgili satırında yer alan sinyal adı girilmelidir. İlgili güzergahta bulunan ray devreleri, makaslar, çakışan güzergahlar ve rota kilitleri yine anlaşılan tablosundaki ilgili güzergah satırından yararlanılarak arayüzdeki yerlerine yazılır. Her güzergahta bulunan ray devresi, makas, çakışan güzergah ve rota kilidi sayısı farklı olabileceğinden ekle ve çıkar butonlarıyla ilgili güzergahın gerektirdiği kadar ekleme veya çıkarma yapılabilir. Varılacak yolun ray devresiz olması veya istasyon yolu olması geçişleri değiştireceği için arayüzde güzergahta varılacak yolun ray devresiz veya istasyon yolu olması durumunda gerekli işaretlemeler yapılmalıdır.

### 5.1.2 Makas otomatu

Makas otomatu, tanzim otomatından normal veya sapan konuma geçmesi için ilgili sinyalin gelmesiyle gereken duruma geçip makasın hangi konuma gitmesi istenmiş ise o yöne çevrilmesini sağlar ve makası elektronik olarak kilitleyip bu bilgiyi tanzim otomatına gönderir. Herhangi bir sebep ile makasın önceden belirlenen bir sürede

talep edilen konuma geçememesi durumunda hata bildirimini vererek tanzim otomatına hata sinyali gönderir. Makas otomatına ilişkin gösterim Şekil 5.5’de verilmiştir.



**Şekil 5.5 :** Makas otomatı.

Makas otomatı, tanzim otomatından gelen normale çevir veya sapana çevir talepleriyle birlikte başlangıç durumundan çıkar. Makasın normal konuma gönderilmesi talep edilmiş ise otomat 1. duruma, sapan konuma gönderilmesi istenmiş ise 4. duruma geçecektir. Makas algılayıcılarından alınan konum bilgilerine göre güzergahın gerektirdiği konumda olan makas elektronik olarak kilitlenir (Durum 3, 6). Makasın güzergahın gerektirdiği konumda olmaması durumunda, güzergahın gerektirdiği konuma almak için makasa komut yollar (Durum 2, 5). Makas talep edilen konuma geldikten sonra elektronik olarak kilitlenir ve makas otomatı bu bilgiyi tanzim otomatına gönderir (Durum 3, 6). Makas herhangi bir sebepten dolayı talep edilen konuma, makas tipine göre belirlenen belli bir sürede geçemezse, makas otomatı hata durumuna geçer (Durum 7). Bu bilgi tanzim otomatına yollar ve tanzim otomatında gerçekleşmekte olan tanzim işlemi makasın hataya düşmesiyle birlikte sonlanır ve tanzim talebi reddedilir. Makas otomatları tasarımı her makas için aynıdır, çünkü her makas aynı işlemleri gerçekleştirir. (Türk, 2010).

Makas otomatı geçişleri aşağıdaki gibi tanımlanmıştır.

$$t1 = m_{x_n\_istek} \quad (5.14)$$

$x$ , makasın ismini,  $m_{x_n\_istek}$  ise  $x$  makasını normale çevir talebini gösterir.

$$t2 = i\_m\_x\_n \wedge \overline{i\_m\_x\_i} \quad (5.15)$$

$i\_m\_x\_n$ ,  $x$  makası normal indikasyonunu,  $i\_m\_x\_i$  ise  $x$  makası sapan indikasyonunu temsil eder. Üstü çizili olması mantıksal DEĞİL anlamına gelmektedir.

$$t3 = \bigwedge_{i=1}^k \overline{rk(i)} \quad (5.16)$$

$k$ , makas ile ilgili güzergah kilidi sayısını,  $rk(i)$  güzergah kilitlerini temsil eder.

$$t4 = m\_x\_i\_istek \quad (5.17)$$

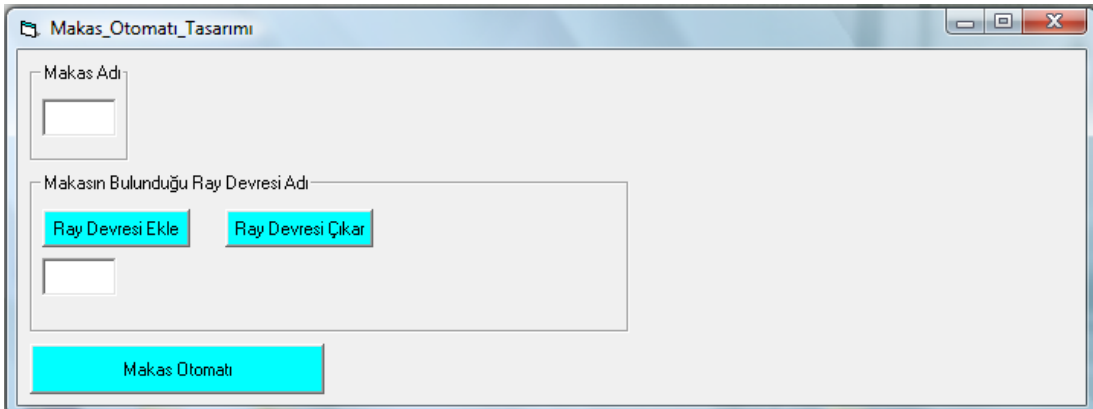
$m\_x\_i\_istek$ ,  $x$  makasını sapana çevir talebini belirtir.

$$t5 = \overline{i\_m\_x\_n} \wedge i\_m\_x\_i \quad (5.18)$$

$$t6 = 7 \text{ saniye bekleme (2. durumda)} \quad (5.19)$$

$$t7 = \bigwedge_{i=1}^k rk(i) \quad (5.20)$$

Makas otomati tasarımı için geliştirilen yazılımın kullanıcıya sunduğu arayüz Şekil 5.6'de verilmiştir.

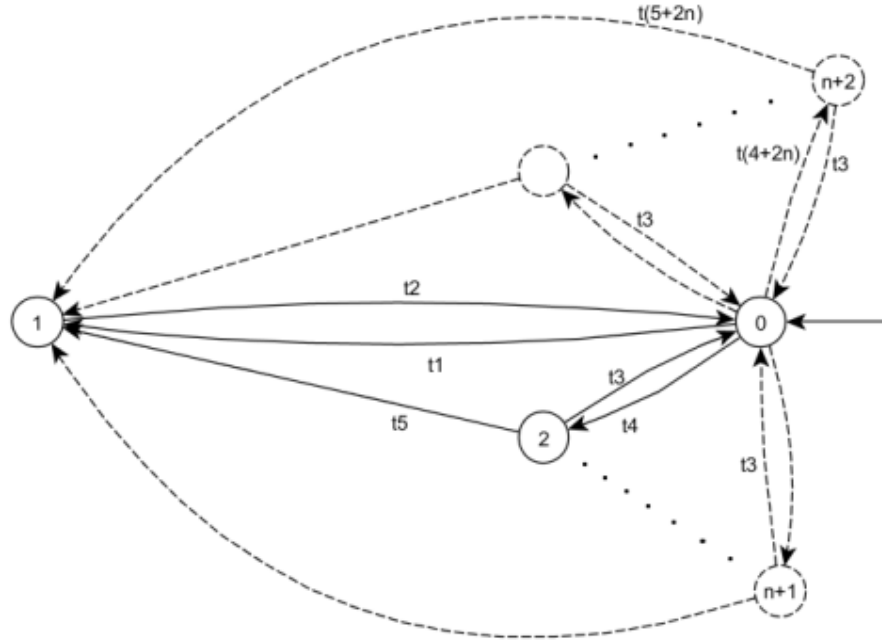


**Şekil 5.6 :** Makas otomati arayüzü.

Şekildeki arayüze otomat tasarımı yapılacak makasın adı ve makasın bulunduğu ray devresi veya devrelerinin adı girilmelidir. “S makas” olarak adlandırılan makaslar iki farklı ray devresinde bulunmaktadır.

### 5.1.3 Sinyal otomatı

Sinyal otomatı, tanzim otomatının güzergah tanzimi için gerekli kontrolleri yapması ve makasların gerekli konumlara gidip kilitlendiği bilgisini almasıyla, ilgili sinyalin gerekli bildirim vermesini sağlar. Sinyal otomatı sahadan gelen verilere göre sinyalin ilgili bildirim vermesi için çıkış üretir. Sinyal bildirim verdikten sonra ilgili indikasyon alınmaz ise sinyal hataya düşer ve bu bilgi tanzim otomatına gönderilir. (Türk, 2010). Sinyal otomatına ilişkin gösterim Şekil 5.7'de verilmiştir.



Şekil 5.7 : Sinyal otomatı.

Sinyal otomatında, başlangıç durumu sinyalin kırmızı bildirim verdiği durumdur. 1 numaralı durum sinyalin hata durumudur. 2 numaralı durumdan n+2 numaralı duruma kadar olan durumlar ise sinyale komut yollanacağı durumları temsil etmektedir. Bu durumların sayısı ilgili sinyalin kırmızı bildirim dışında vereceği bildirim sayısı kadar olmalıdır (Türk, 2010). Başlangıç durumunda sinyale hiçbir komut göndermeyerek donanımsal olarak sinyalin kırmızı bildirim vermesinin sağlanacağı kabul edilmiştir.

İlgili güzergahta bulunan tüm makasların kilitletmesinin ardından, tanzim otomatı son kontrolleri yapar ve bu kontroller sonucunda güzergahı tanzime uygun bulursa, sinyal otomatı başlangıç durumundan çıkar. Sinyal otomatı, komut göndereceği sinyalden sonra gelen aynı istikametteki ilk sinyali ve güzergahta bulunan ray devrelerini kontrol ederek güzergah sinyaline ilgili renk bildirim komutunu

göndereceği duruma (Durum 2, ..., n+2) geçer. Otomatın, sinyalin bildirim vermesi için geçtiği durumlarda ilgili sinyal indikasyonu belli bir süre içinde alınmaz ise sinyal otomatı 1 numaralı hata durumuna geçer. Güzergahın ilk ray devresinin dolmasıyla birlikte otomat başlangıç durumuna geri döner (Türk, 2010). Sinyal otomatı geçişleri aşağıdaki gibi tanımlanmıştır.

$$t1 = \overline{i\_s\_x\_k} \wedge t_d \quad (5.21)$$

$x$ , ilgili güzergah sinyalinin adı,  $i\_s\_x\_k$  ise  $x$  sinyalinin kırmızı indikasyonunu temsil eder.  $t_d$ , indikasyonun gelmesi için beklenen sürenin dolmasıyla mantıksal 1 olan değişkeni göstermektedir.

$$t2 = \bigvee_{j=1}^l rt(j) \quad (5.22)$$

$l$ , ilgili sinyali kapsayan güzergahların sayısını,  $rt(j)$  ise sinyalin bulunduğu güzergah taleplerini belirtir.

$$t3 = \left( \bigvee_{j=2}^{2+n} t(2j) \uparrow \right) \vee \left( \bigvee_{j=2}^{2+n} t(2j) \downarrow \right) \quad (5.23)$$

Sinyalin otomatının durum değiştirmesini gerektiren olaylardan biri meydana geldiğinde, bir çevrim süresi boyunca  $t_3$  geçişi mantıksal 1 değerinde olacaktır. 5.23 numaralı denklemdeki  $\uparrow$  işareti yükselen kenar,  $\downarrow$  işareti ise düşen kenar anlamına gelmektedir.

$$t4 = \bigvee_{k=1}^t \left( rtx\_hxr \wedge \left( \bigwedge_{i=1}^r rd(i) \right) \wedge \left( \bigvee_{j=1}^s i\_s\_xk\_c(j) \right) \right) \quad (5.24)$$

5.24 numaralı ifadede  $t$ , 2 numaralı durumda sinyale gönderilecek komut ile ilgili geçiş sayısını;  $x$ , ilgili güzergah numarasını,  $r$  ise güzergahta bulunan ancak istasyon bölgesinde olmayan ray devreleri sayısını;  $s$ , ilgili sinyalle aynı istikametteki bir sonraki sinyalin ilgili durum için indikasyon sayısını belirtir.  $rtx\_hxr$ ,  $x$  güzergahına tanzim izni verildiğini;  $rd(i)$ ,  $x$  güzergahında bulunan ancak istasyon bölgesinde olmayan ray devrelerinin durumunu,  $i\_s\_xk\_c(j)$  ise ilgili sinyalle aynı istikametteki bir sonraki sinyalin indikasyonunu gösterir. 5.24 numaralı denklem sinyalin sarı üzeri kırmızı bildirim verdiği durumu kapsamaz. 2 numaralı durumun sarı üzeri

kırmızı bildirim verildiği durum olduğu varsayılırsa 5.24 numaralı ifadeyi aşağıdaki gibi düzenlemek gerekir. Bunun nedeni ilgili güzergahtaki istasyon ray devresi meşgul olsa dahi tanzim izninin verilebilmesidir.

$$t4 = \bigvee_{k=1}^t \left( rtx\_hZR \wedge \left( \bigwedge_{i=1}^r rd(i) \right) \wedge \overline{ird} \right) \quad (5.25)$$

$ird$ , istasyon ray devresinin durumunu göstermektedir.

$$t5 = \overline{i\_s\_x\_c1} \wedge t_d \quad (5.26)$$

$c1$ , sinyalin kırmızıdan başka bir rengini,  $i\_s\_x\_c1$  ise x sinyalinin  $c1$  indikasyonunu gösterir.

Sinyal otomati tasarımı için geliştirilen yazılımın kullanıcıya sunduğu arayüz Şekil 5.8'de verilmiştir.

**Şekil 5.8 :** Sinyal otomati arayüzü.

Otomat tasarımı yapılacak sinyalin adının, sinyali kapsayan güzergahların, bu güzergahların içinde bulunan ray devrelerinin ve ilgili sinyal ile aynı istikametteki ilk sinyallerin adları gerekli bölümlere girilir. Anlaşman tablosundan ilgili sinyalin verdiği bildirimlerden biri işaretlenir ve bu bildirim verilebilmesi için gerekli şartlar yine anlaşman tablosundan okunarak girilir ve ilgili bölümler seçilerek sinyalin vereceği bildirim şartları tanımlanmış olur. Daha sonra aynı işlemler sinyalin vereceği diğer bildirimler için de tekrarlanır. Böylelikle aslında otomat geçişleri tanımlanmış olur.

Bu bölümde tanıtılan yazılım arayüzü kullanımına ilişkin örnek Bölüm 6.3'de verilecektir.





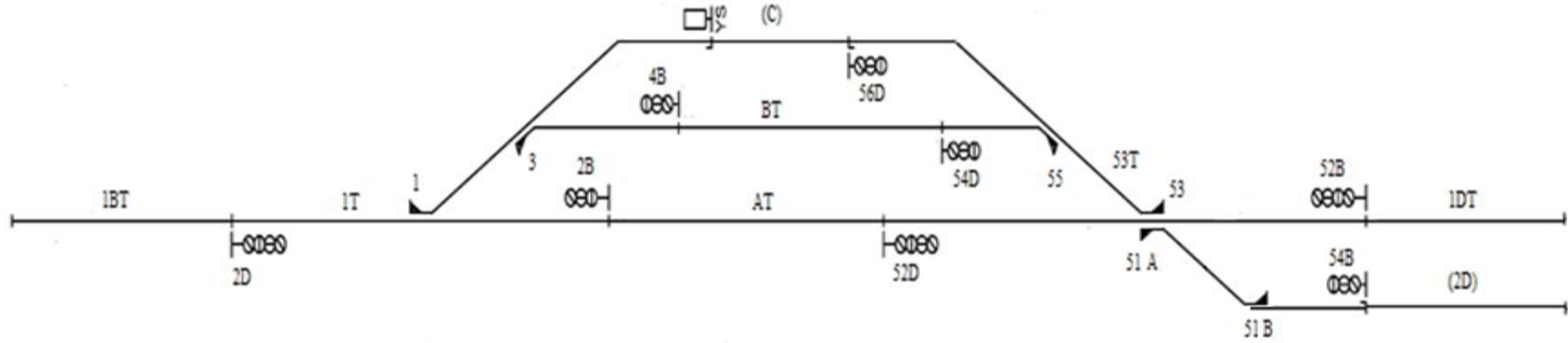
## **6. OTOMATİK ANKLAŞMAN ALGORİTMASI VE KODU ÜRETME YÖNTEMİ İLE ÖRNEK BİR HAT İÇİN ANKLAŞMAN ALGORİTMASI TASARLAMA**

### **6.1 Giriş**

Sinyalizasyon sistemi ve otomatlar hakkında bilgi sunduktan sonra, bu bölümde örnek bir hattın, otomatik anklâşman algoritması ve kodu üretme yöntemi kullanılarak anklâşman algoritmasının gerçekleştirilmesi anlatılacaktır.

### **6.2 Örnek Hattın Tanıtılması ve Anklâşman Tablosu**

Bu tez çalışmasında ele alınan hatta Şekil 6.1’de görüldüğü gibi 6 ray devresi, 5 adet makas, 3 adet 4’lü yüksek sinyal, 3 adet 3’lü yüksek sinyal ve 4 adet cüce sinyal bulunmaktadır. (C) ve (2D) yollarında ray devreleri bulunmamaktadır. Sahanın batısında bulunan ve batı yönünde seyreden trenler için bildirimde bulunan 3’lü yüksek sinyal 2B ve cüce sinyal 4B, doğu yönünde bildirim yapan 4’lü yüksek sinyal ise 2D olarak isimlendirilmiştir. Sahanın doğusunda bulunan ve doğu yönünde hareket eden trenler için bildirimde bulunan 3’lü cüce sinyaller 56D ve 54D, 4’lü yüksek sinyal ise 52D olarak adlandırılmışken batı yönünde bildirimde bulunan cüce sinyale 54B, 4’lü yüksek sinyale de 52B adı verilmiştir. Ray devreleri numara ve harfler ile isimlendirilirken ray devreleri isimlerinin sonunda yer alan “T” harfi İngilizce “track” kelimesinden gelmektedir. Parantez içinde belirtilmiş olan (C) ve (2D) yollarında ray devresi bulunmadığından isimlerinin sonuna “T” harfi eklenmemiştir. Sahanın batısında bulunan makaslar 1’den başlayarak tek sayılar ile numaralandırılmış, doğuda bulunanlar ise numaralandırılmaya 51’den başlayarak yine tek sayılar şeklinde devam numaralandırılmıştır. Bu hat için, KM tarafından tanzim talebinde bulunulabilecek 18 adet güzergah bulunmaktadır. Bu güzergahlar Çizelge 6.1 ve Çizelge 6.2 verilen örnek hat anklâşman tablosunda belirtilmiştir.



Şekil 6.1 : Örnek hat sinyalizasyon planı.

Çizelge 6.1 : Örnek hat anlaşıman tablosu (Rota1-Rota3).

TANIM		ROTA	SİNYAL NO	SİNYAL DURUMU		KİLİT	SİNYAL KONTROL	ROTA KİLİT	
İSİM	ROTA NO			Y	S				
GİRİŞ SİNYALİ	rt1	1BT - AT	2D	A	Y 52D S	1	2B 52BA 54BA	1T AT	(1T)
				S	52D K, SK			1T (AT)	
				SK					
	rt2	1BT - BT	2D	B	SY 54D S	(1) 3	4B 52BB 54BB	1T BT	(1T)
				SS	54D K, SK			1T (BT)	
				SK					
rt3	1BT - (C)	(C)	SK		(1) 3	52BC 54BC	1T	(1T)	

**Çizelge 6.2 : Örnek hat anlaşıman tablosu (Rota4-Rota18).**

TANIM			SİNYAL NO		SİNYAL DURUMU		KİLİT		SİNYAL KONTROL		ROTA KİLİT
İSİM	ROTA NO	ROTA									
ÇIKIŞ	rt4	AT-1BT	2B		S		1	2D	1T 1BT	(1T)	
DO	rt5	BT-1BT	4B		S		③①	2D	1T 1BT	(1T)	
DO	rt6	(C)-1BT	4B		FS		3 ①	2D	1T 1BT	(1T)	
DO	rt7	AT-1DT	52D	1D	S		51 53	52B	53T 1DT	(53T)	
	rt8	AT-(2D)		(2D)	SK		⑤① 53	54B	53T	(53T)	
DO	rt9	BT-1DT	54D	1D	S		⑤⑤ ⑤③ 51	52B	53T 1DT	(53T)	
	rt10	BT-(2D)		(2D)	SK		⑤⑤ ⑤③ ⑤①	54B	53T	(53T)	
DO	rt11	(C)-1DT	56D	1D	S		55 ⑤③ 51	52B	53T 1DT	(53T)	
	rt12	(C)-(2D)		(2D)	SK		55 ⑤③ ⑤①	54B	53T	(53T)	
GİRİŞ SİNYALİ	rt13	1DT-AT	52B	A	Y	2B S	53 51	52D 2DA	53T AT	(53T)	
					S	2B K					
					SK						
	rt14	1DT-BT		B	SY	4B S	51 ⑤③ ⑤⑤	54D 2DB	53T BT	(53T)	
					SS	4B K					
					SK						
	rt15	1DT-(C)	(C)	SK		51 ⑤③ 55	56D 2DC	53T	(53T)		
	rt16	(2D)-AT	54B	A	Y	2B S	⑤① 53	52D 2DA	53T AT	(53T)	
					S	2B K					
					SK						
rt17	(2D)-BT	B		Y	4B S	⑤① ⑤③ ⑤⑤	54D 2DB	53T BT	(53T)		
				S	4B K						
				SK							
rt18	(2D)-(C)	(C)	SK		⑤① ⑤③ 55	56D 2DC	53T	(53T)			

### 6.3 Anlaşman Algoritması Tasarımı

Önceki bölümde örnek hat için anlaşman tablosunun oluşturulmasıyla algoritma tasarımı için gerekli olan girişler oluşturulmuş oldu. Bu bölümde yapılacak tasarımda anlaşman yazılımı Bölüm 5.1’de değinildiği üzere 3 ayrı fonksiyon bloğu olarak ele alınacak, böylelikle hem durum patlaması probleminin üstesinden gelmek için hem de tasarım ve uygulama kolaylığı açısından fayda sağlayacağı düşünülmektedir (Türk, 2010).

#### 6.3.1 Tanzim bloğu

Bölüm 5.1.1’de açıklanan tanzim otomatının, otomatik anlaşman algoritması ve kodu üretme yöntemi ile gerçekleştirilmesi bu bölümde örnek üzerinde ele alınacaktır. Tanzim bloğunun, anlaşman tablosunda yer alan tüm güzergahlar için ayrı ayrı tasarlanması düşünülebilir. Bu şekilde her tanzim otomati bir fonksiyon bloğuna karşılık gelecektir. Anlaşman tablosunda 18 adet güzergah olduğundan tasarlanması gereken tanzim otomati ve buna bağlı olarak tanzim fonksiyon bloğu sayısı 18 olmalıdır.

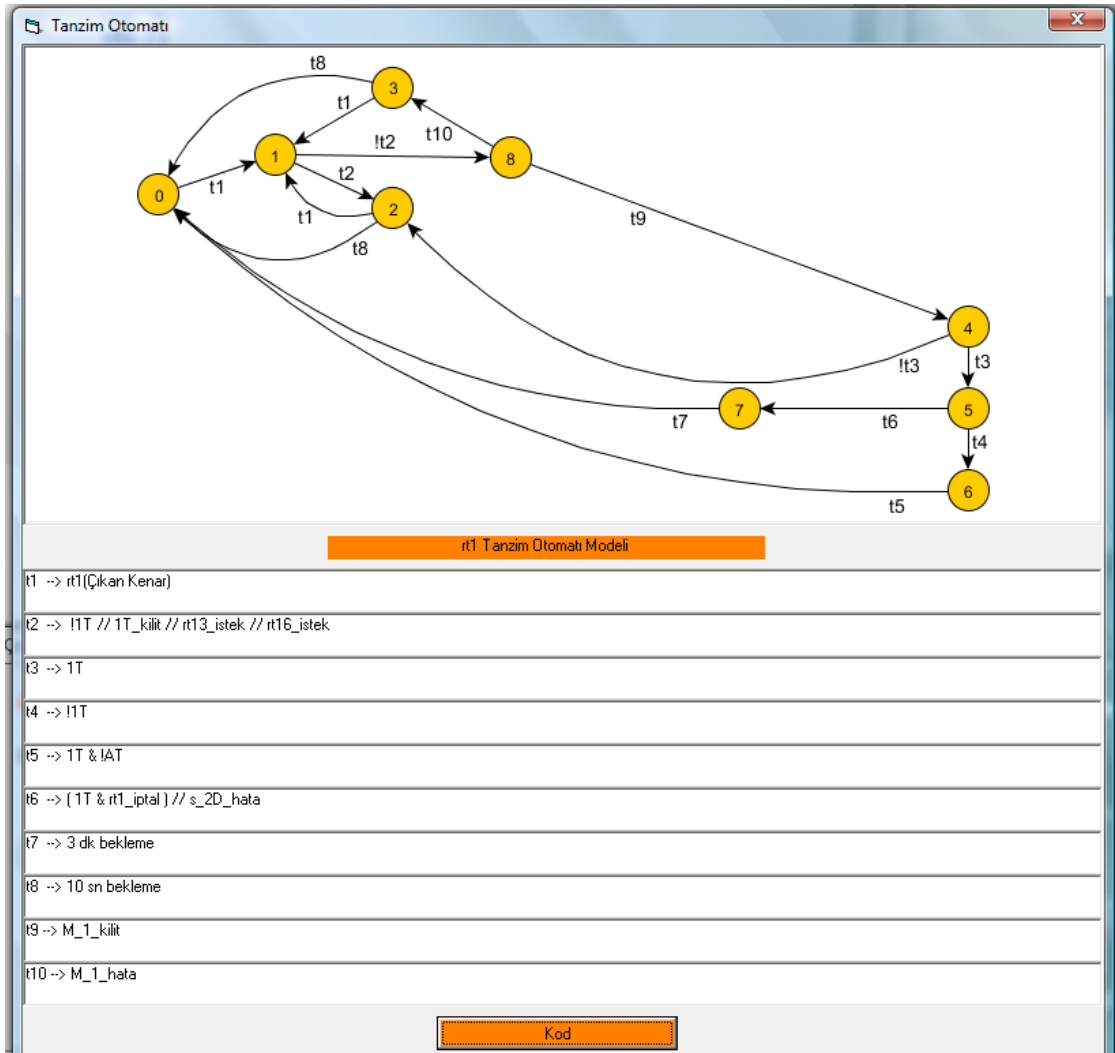
Bu bölümde, Çizelge 6.1’de verilen anlaşman tablosunun ilk güzergahı olan 1BT-AT için geliştirilen yazılım kullanılarak yapılan tanzim bloğu tasarımı adım adım verilecektir.

The screenshot shows a software window titled "Tanzim Otomati Tasarımı". It contains several sections for configuring a route:

- Rota Adı:** Input field with "rt1".
- Sinyal Adı:** Input field with "2D".
- Ray Devresi:** Buttons "Ray Devresi Ekle" and "Ray Devresi Çıkar". Checkboxes: "Varılacak yolda ray devresi yok" (unchecked), "Varılacak yol istasyon yolu" (checked). Input field "AT".
- Makas:** Buttons "Makas Ekle" and "Makas Çıkar". Input field "1".
- Çıkış Rota:** Buttons "Çıkış Rota Ekle" and "Çıkış Rota Çıkar". Input fields "rt13" and "rt16".
- Rota Kiliti:** Buttons "Rota Kiliti Ekle" and "Rota Kiliti Çıkar". Input field "1T".
- Bottom:** Large cyan button "Tanzim Otomati".

Şekil 6.2 : 1BT-AT güzergahına ilişkin verilerin arayüze girilmesi.

Şekil 6.2’de verilen arayüze güzergahın, güzergahta bulunan ray devrelerinin, makasların, çakışan güzergahların ve güzergah kilitlerinin isimleri anlaşılan tablosunun ilgili satırından okunarak belirtilen alanlara girilmesiyle çıkış olarak üretilecek otomat geçişleri ve mantıksal fonksiyonları için giriş bilgileri alınmış olur. Bölüm 5.1.1’de bahsedildiği üzere tanzim otomatu durum sayısını belirleyen unsur Şekil 5.3’de de görüldüğü üzere güzergahtaki makas sayısıdır. Ele alınan güzergahtaki makas sayısı bir olduğu için dokuz durumlu bir otomat çıktı olarak üretilmiştir. Üretilen bu otomat ve geçişleri Şekil 6.3’de verilmiştir.



Şekil 6.3 : 1BT-AT güzergahı için tanzim otomatu ve geçişler.

Üretilen otomat ve geçişler kullanılarak gerçekleştirilen Bölüm 4.3.2’de açıklandığı üzere, PLC’nin çalışma prensibine uygun olarak zamanda ayırık anlarda işletilen mantıksal fonksiyonlarla ifade edilecektir. Şekil 6.4’de yukarıda verilen otomatın geliştirilen yazılım tarafından üretilen mantıksal fonksiyonları verilmiştir.

DURUMLAR	ÇIKIŞLAR
$Q1 = (q0 + q2 + q3) \cdot t1$	$rt1\_istek = q1 + q4 + q5 + q6 + q7 + q8$
$Q2 = q1 \cdot t2 + q4 \cdot t3 + q2 \cdot t1 \cdot t8$	$rt1\_ret = q2 + q3$
$Q3 = q8 \cdot t10 + q3 \cdot t1 \cdot t8$	$rt1\_hzt = q5 + q6$
$Q4 = q8 \cdot t9$	$M\_1\_N\_istek = q8$
$Q5 = q4 \cdot t3 + q5 \cdot t4 \cdot t6$	$1T\_kilit = q4 + q5 + q6 + q7 + q8$
$Q6 = q5 \cdot t4 + q6 \cdot t5$	
$Q7 = q5 \cdot t6 + q7 \cdot t7$	
$Q8 = q1 \cdot t2 + q8 \cdot t9 \cdot t10$	
$Q0 = IQ1 \cdot IQ2 \cdot IQ3 \cdot IQ4 \cdot IQ5 \cdot IQ6 \cdot IQ7 \cdot IQ8$	
$Q0 = q0$	
$Q1 = q1$	
$Q2 = q2$	
$Q3 = q3$	
$Q4 = q4$	
$Q5 = q5$	
$Q6 = q6$	
$Q7 = q7$	
$Q8 = q8$	

**Şekil 6.4 :** 1BT-AT güzergahı için tanzim otomatı mantık fonksiyonları.

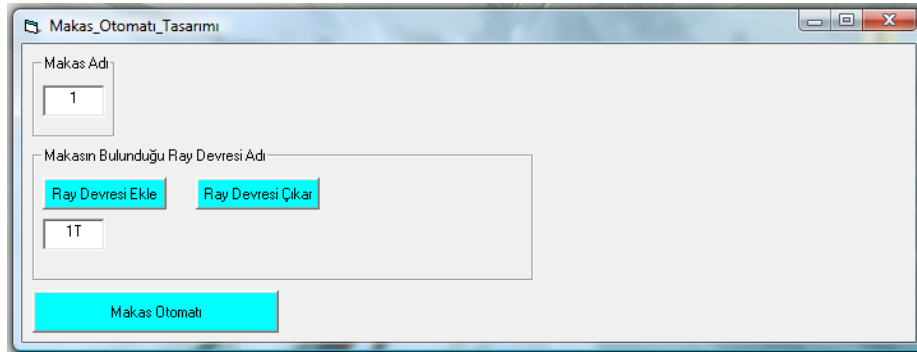
Şekil 6.4’de verilen mantıksal fonksiyonları STL, ST, FBD, LAD veya SFC dillerinden biri tercih edilerek PLC yazılımına dönüştürülmesiyle ilk güzergah için tanzim otomatının PLC gerçekleştirilmesi yapılmış olur. Geriye kalan 17 güzergah için de aynı şekilde anlaşılan tablosundan gerekli satırlar okunup, yazılıma girilerek 18 güzergah için tanzim otomatının PLC gerçekleştirilmesi tamamlanmış olur.

Tasarımı biraz daha kolaylaştırmak adına her güzergah için ayrı ayrı yazılıma giriş yaparak tasarım yapmak yerine şu şekilde bir yöntem izlenebilir. Bu yöntemde anlaşılan tablosundan en fazla ray devresi, makas ve çakışan güzergah seçilerek ilgili girişler yazılıma girilir, bu güzergah için bir defa otomat ve mantıksal fonksiyonları çıktı olarak alınır. Mantık fonksiyonları PLC yazılımına dönüştürülürken yazılımcı, en geniş güzergah tanzim otomatı için tanımladığı fonksiyon bloğu içinde kullanacağı tüm değişkenleri lokal olarak tanımlayabilir. Böylelikle fonksiyon bloğunda kullanılan ilgili lokal değişkenlere güzergahta kullanılan gerçek global değişkenler atanarak tasarım tamamlanabilir. Bu yöntemde en geniş güzergah için tasarım yapıldığından daha az ray devresi, makas, çakışan güzergah bulunduran güzergahlar için gerekli olmayan bellek bitleri kullanılmış olacaktır. Örneğin örnek hat anlaşılan tablosunda en fazla makas bulunduran güzergahta üç makas bulunmaktadır. Bunun yanından tek makas bulunduran güzergahlar da mevcuttur. Üç makaslı güzergah için içinde lokal değişkenler bulunan fonksiyon bloğu tasarlandığında ve bu blok tek makaslı güzergah için kullanıldığında fazladan olan iki makas ile ilgili bitlere duruma göre mantıksal 1 veya 0 atanarak kullanılabilir. Gerçekte tek makaslı güzergah için zaten olmayan makasın hataya

düşme olasılığı olmadığı için hata bitine mantıksal 0, makasın kilitlendiği gösteren bitine de kilitlenmiş olarak kabul edilip mantıksal 1 atanır. Böylece, tasarımın başında gerçekleşen tek bir fonksiyon bloğu ile bu bloğun sadece girişlerine her seferinde güzergahın gerektirdiği global değişkenler atanarak ve 18 adet kopyalanarak hızlı bir şekilde ve hata yapma olasılığını azaltarak tasarım tamamlanabilir. Önerilen bu yöntem hızdan kazandırırken tasarım en geniş güzergah için yapıldığından daha az saha ekipmanını kapsayan güzergahlar için gereksiz bellek kullanımına yol açacaktır. Her bir güzergah için ayrı ayrı tasarım yapmanın faydası ise gereksiz bellek kullanımına yol açmamasıdır, ancak hızdan belli ölçüde kaybedilecektir. Tasarımın geliştirilen yazılım kullanılarak yapıldığı düşünülürse, kaybedilecek zaman ancak anlaşılan tablosundan her güzergah için ilgili satırı okuyup yazılıma girme süresi kadar olacaktır.

### 6.3.2 Makas bloğu

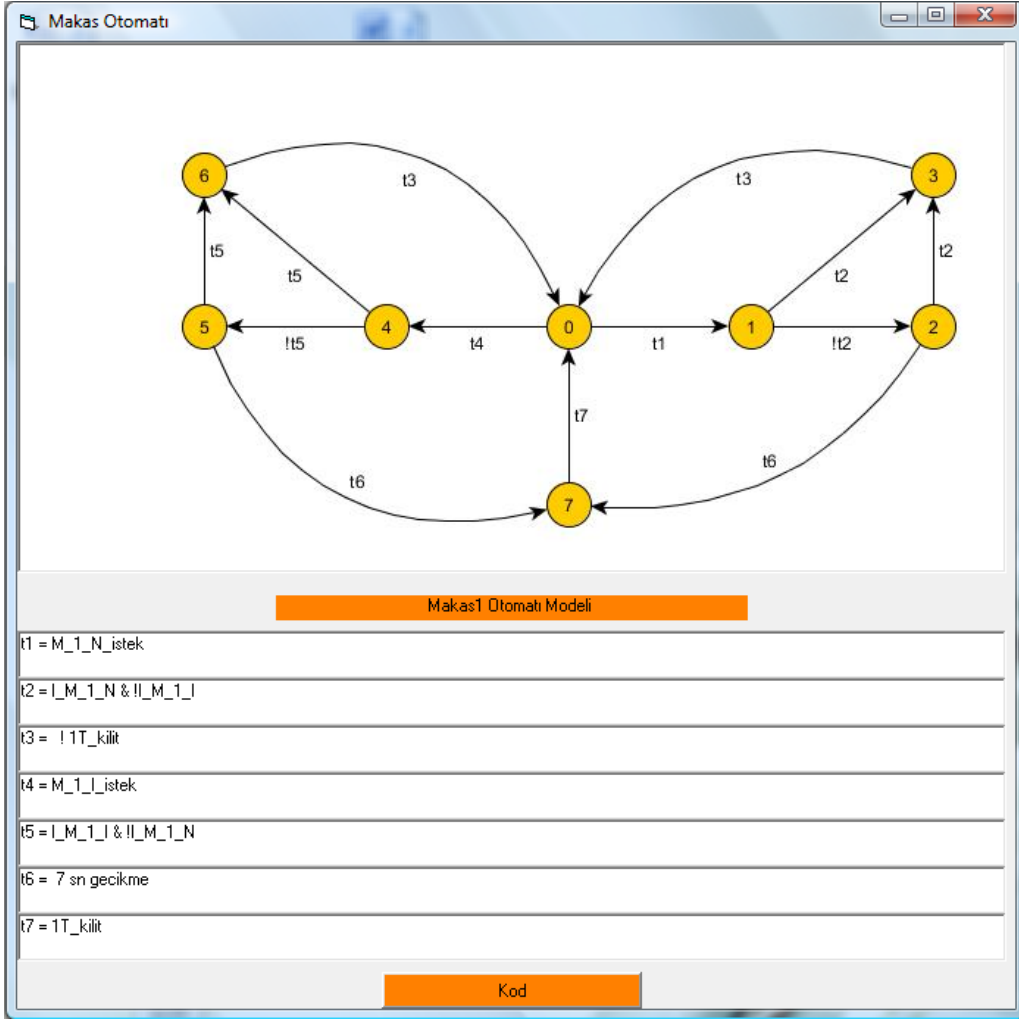
Bölüm 5.1.2’de açıklanan makas otomatının, otomatik anlaşılan algoritması ve mantıksal fonksiyonlarını üretme yöntemi ile gerçekleşmesi bu bölümde bir örnek üzerinde ele alınacaktır. Makas bloğunun, sahada yer alan makaslar için ayrı ayrı tasarlanabilir. Tanzim otomatıyla benzer mantıkla, her makas otomatı bir fonksiyon bloğuna karşılık gelecektir. Örnek sahada 5 adet makas bulunduğundan tasarlanması gereken tanzim otomatı ve buna bağlı olarak tanzim fonksiyon bloğu sayısı 5’dir. Makas bloğu tasarımı için örnek olarak 1 numaralı makas seçilerek aşağıdaki gibi yazılıma girişler yapılmıştır.



**Şekil 6.5 :** 1 numaralı makas için verilerin arayüze girilmesi.

Şekil 6.5’de verilen arayüze makasın adı ve makasın bulunduğu ray devresi adının girilmesiyle çıkış olarak üretilecek otomat geçişleri ve mantıksal fonksiyonları için giriş bilgileri alınmış olur. Bölüm 5.1.2’de belirtildiği üzere makas otomatı durum

sayısı sabittir ve yazılıma alınan girişler geçiş, çıkış bitlerinin isimlendirilmesinde etkilidir. Üretilen bu otomat ve geçişleri Şekil 6.6'da verilmiştir.

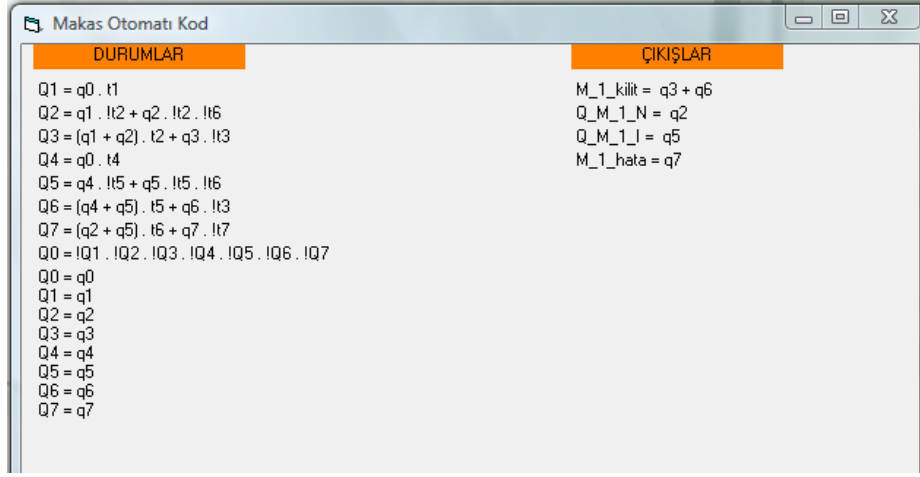


Şekil 6.6 : 1 numaralı makas otomati ve geçişleri.

Üretilen otomat ve geçişler kullanılarak gerçekleştirilen Bölüm 4.3.2'de açıklandığı üzere, aynı tanzim otomatının gerçekleştirilmesinde olduğu gibi PLC'nin çalışma prensibine uygun olarak zamanda ayrıklıklarda işletilen mantıksal fonksiyonlarla gösterilmektedir. Şekil 6.7'de yukarıda verilen otomatın geliştirilen yazılım tarafından üretilen mantıksal fonksiyonları verilmiştir.

Tanzim bloğu bölümünde açıklandığı gibi aynı mantıkla tek bir makas bloğu için tasarım yapılarak bu bloğun değişkenleri lokal tanımlanırsa, lokal değişkenlere ilgili makasın global değişkenleri atanarak tasarım tamamlanır.



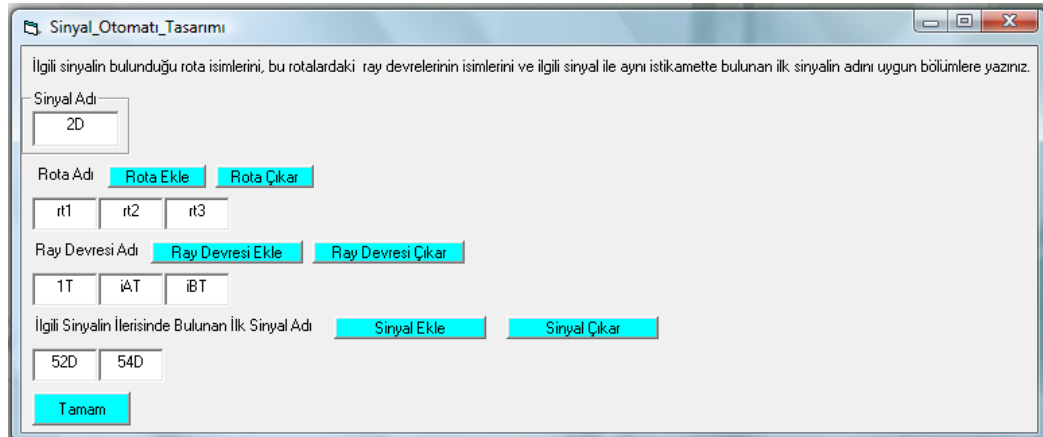


Şekil 6.7 : 1 numaralı makas otomati mantık fonksiyonları.

Makas otomatının durum sayısı makastan makasa değişmeyip sabit olduğundan fonksiyon bloğu değişkenlerini lokal tanımlayıp global değişkenlerin lokal değişkenlere atanması çok daha mantıklı olacaktır.

### 6.3.3 Sinyal bloğu

Sinyal bloğu tasarımı da diğer bloklarda olduğu gibi her sinyal için ayrı bir otomat ve bu otomatın PLC gerçeklemesinin yapıldığı fonksiyon bloğu şeklinde olacaktır. Sinyal fonksiyon bloğu tasarımına örnek olarak 2D sinyali verilecektir. Şekil 6.8'de 2D sinyali için yazılıma girişler yapılmıştır.



Şekil 6.8 : 2D sinyaline ilişkin verilerin arayüze girilmesi.

2D sinyali için tasarım yapılırken gerekli olan giriş bilgileri anlaşılan tablosunun 2D sinyali ile ilgili satırlarının okunmasıyla elde edilmektedir. Anlaşılan tablosunun 2D sinyali ile ilgili satırları Çizelge 6.3'de verilmiştir.

**Çizelge 6.3 :** Örnek hat anlaşıman tablosunun 2D sinyali ile ilgili satırları.

TANIM		ROTA	SİNYAL NO	SİNYAL DURUMU		KİLİT	SİNYAL KONTROL	ROTA KİLİT	
İSİM	ROTA NO			Y	S				
GİRİŞ SİNYALİ	rt1	1BT - AT	2D	A	52D S	1	2B 52BA 54BA	1T AT	(1T)
				S	52D K, SK				
				SK					
	rt2	1BT - BT		B	54D S	1 3	4B 52BB 54BB	1T BT	
				SS	54D K, SK				
				SK					
	rt3	1BT - (C)		(C)	SK	1 3	52BC 54BC	1T	

2D sinyali tabloda görüldüğü üzere *rt1*, *rt2* ve *rt3* güzergahları için bildirimde bulunabilmektedir. Bu güzergahlarda bulunan ray devreleri de tablonun sağ tarafında yer almaktadır. Bu ray devreleri görüldüğü üzere *1T*, *AT* ve *BT*'dir. *AT* ve *BT* ray devreleri istasyon bölgesinde buldukları için isimlerinin başlarına yazılıma girilirken "i" harfi konulması gerekmektedir. Önceden de değinildiği üzere istasyon bölgesinde meşguliyet olsa dahi sarı üzeri kırmızı bildirimini ile güzergah tanzimi yapılabilirdi. Bu nedenle istasyon bölgelerinde bulunan ray devrelerinin yazılıma giriş esnasında belirtilmesi gerekmektedir. İstasyon bölgesindeki ray devrelerini diğerlerinden ayırt edebilmek için sinyal otomati arayüzünde böyle bir yol tercih edilmiş ve istasyon bölgesi ray devrelerinin başına "i" harfi konulması gerekli kılınmıştır.

Tasarımın ikinci adımında 2D sinyalinin, *rt1* numaralı güzergah tanzim edildiğinde yeşil bildirimde bulunması için gerekli şartlar anlaşıman tablosundan bakılarak yazılıma girilir ve kaydedilir. Şekil 6.9'da 2D sinyalinin *rt1* numaralı güzergah tanzim edildiğinde yeşil bildirim verebilmesi için gerekli şartların girilmesi gösterilmiştir. Yazılımın bu girişlere karşılık ürettiği mantıksal ifade ise Eşitlik 6.1'de verilmiştir.

$$t_{y1} = rt1\_hzt \wedge 1T \wedge AT \wedge i\_S\_52D\_S \quad (6.1)$$

**Şekil 6.9 :** 2D sinyaline ilişkin geçişlerin tanımlanması.

Bu eşitlikte  $t_{y1}$ , sinyalin yeşil bildirim verebilmesi için gerekli olan geçişlerden birincisini ifade etmektedir;  $rt1\_hzr$ ,  $rt1$  numaralı güzergah için tanzimin gerçekleştiğini,  $1T$  ve  $AT$  ise ray devrelerinin durumlarını göstermektedir.  $i\_S\_52D\_S$ , ise 52D sinyalinin sarı indikasyonu anlamına gelmektedir. Sonuç olarak, sinyalin yeşil bildirim verebilmesi gerekli şartlardan biri  $t_{y1}$  geçişinin mantıksal 1 olmasıdır. Bunun için de  $rt1$  nolu güzergah için tanzim bitinin mantıksal 1 olması,  $1T$  ve  $AT$  ray devrelerinin müsait olması, 52D sinyalinin de sarı bildirim veriyor olması gerekmektedir.

2D sinyalinin ilgili hat için verebileceği bildirimlerin hepsi Çizelge 6.4'de verilmiştir. Bu bildirimlerin verilebilmesi için Çizelge 6.4'deki şartların yazılıma girilmesi gerekmektedir. Anlaşman tablosunun 2D sinyali için düzenlenmiş sinyal durumu sütununda 7 adet satır bulunduğundan 7 defa gerekli şartların girilmesi ve ilgili geçişlerin kaydedilmesi gerekmektedir. Şekil 6.10'da gösterilen şekilde gerekli girişlerin hepsi yapılmıştır.

Şekil 6.10'daki girişlere karşılık düşen yazılım çıktısı Şekil 6.11'de verilmiştir. Çıktı olarak 7 durumlu bir otomat üretilmiştir. Bölüm 5.1.3'de de açıklandığı üzere ilk durum başlangıç durumu, birinci durum sinyalin hata durumu göstermektedir. Bu

durumlarda sinyal kırmızı bildirim vermektedir. Diğer durumlar ise sinyalin kırmızı dışında bildirim verdiği durumları kapsamaktadır.

Sinyal\_Otomati\_Tasarımı

İlgili sinyalin bulunduğu rota isimlerini, bu rotalardaki ray devrelerinin isimlerini ve ilgili sinyal ile aynı istikamette bulunan ilk sinyalin adını uygun bölümlere yazınız.

Sinyal Adı:

Rota Adı:

Ray Devresi Adı:

İlgili Sinyal ile Aynı İstikamette Bulunan İlk Sinyal Adı:

İlgili Sinyal Bildirimleri:

Y  S  SY  SS  SK  FY  FS  FK  FSK

2D sinyalinin sarı üstü kırmızı bildirim vermesi gereken rotayı işaretleyiniz.

rt1  rt2  rt3

2D sinyalinin sarı üstü kırmızı bildirim vermesi için ilgili rotanın içerdiği ray devrelerini işaretleyiniz.

1T  AT  BT

İlgili rota için 2D sinyali ile aynı istikamette bulunan ilk sinyali seçiniz.

52D  54D

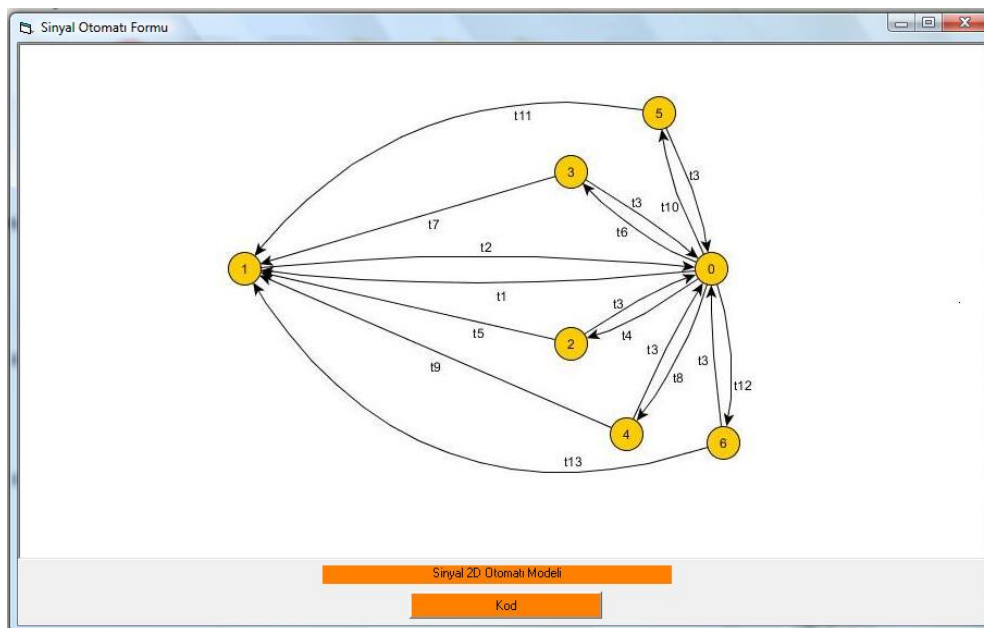
2D sinyalinin sarı üstü kırmızı bildirim vermesi için gerekli şart kaydedildi.

```

t_y 1 = rt1_hzr & 1T & AT & (i_s_52D_S)
t_s 1 = rt1_hzr & 1T & AT & (i_s_52D_SK // i_s_52D_K)
t_sk 1 = rt1_hzr & 1T & iAT
t_sy 1 = rt2_hzr & 1T & BT & (i_s_54D_S)
t_ss 1 = rt2_hzr & 1T & BT & (i_s_54D_SK // i_s_54D_K)
t_sk 2 = rt2_hzr & 1T & iBT
t_sk 3 = rt3_hzr & 1T

```

Şekil 6.10 : 2D sinyaline ilişkin tüm geçişler.



Şekil 6.11: 2D sinyali otomati.

2D sinyal otomatı geçişleri ve mantıksal fonksiyonları Şekil 6.12’de verilmiştir.  $t_1$ ,  $t_5$ ,  $t_7$ ,  $t_9$ ,  $t_{11}$  ve  $t_{13}$  ifadeleri, sinyale komut gönderilip belli bir süre sonra indikasyon bilgisi alınamaması durumunda mantıksal olarak 1 olacak geçişlerdir.  $t_2$  ise sinyali kapsayan bir güzergah tanzim talebiyle sinyalin hata durumundan çıkmasını sağlayacak geçiştir.  $t_3$  geçişine ise Bölüm 5.1.3’den de hatırlanacağı üzere sinyalin durum değiştirmesinin gerektiren bir olay meydana gelmesiyle 1 çevrim süresi boyunca mantıksal 1 olacak update olarak isimlendirilmiş bellek biti atanmıştır. Diğer geçişler de sinyal otomatının kırımızı dışındaki bildirimleri vermesini sağlayacak durumlara geçebilmesini sağlayacak geçişlerdir. Bu geçişler, Şekil 6.10’da sinyalin gerekli bildirimleri verebilmesi için anlaşılan tablosundan yararlanılarak oluşturulan şartların sinyalin aynı bildirimini vermesini sağlayacak şekilde gruplanıp mantıksal VEYA’lanarak ifade edilmesidir. Örneğin  $t_{12}$  geçişi sinyali sarı üzeri kırmızı bildirim vermesini sağlayacak geçiştir. Bu geçiş Şekil 6.10’da gösterilen  $t_{sk1}$ ,  $t_{sk2}$ ,  $t_{sk3}$  şartlarının mantıksal VEYA’lanmasıyla elde edilmiştir.

```

Sinyal Otomatı Kod

GEÇİŞLER
t1 = K_hata
t2 = r1T // r2 // r3
t3 = update
t4 = ty 1
t5 = Y_hata
t6 = ts 1
t7 = S_hata
t8 = tsy 1
t9 = SY_hata
t10 = tss 1
t11 = SS_hata
t12 = tsk 1 // tsk 2 // tsk 3
t13 = SK_hata

DURUMLAR
Q1 = q0 . t1 + q2 . t5 + q3 . t7 + q4 . t9 + q5 . t11 + q6 . t13 + q1 . t2
Q2 = q0 . t4 + q2 . t3 . t5
Q3 = q0 . t6 + q3 . t3 . t7
Q4 = q0 . t8 + q4 . t3 . t9
Q5 = q0 . t10 + q5 . t3 . t11
Q6 = q0 . t12 + q6 . t3 . t13
Q0 = !Q1 . !Q2 . !Q3 . !Q4 . !Q5 . !Q6
q0 = Q0
q1 = Q1
q2 = Q2
q3 = Q3
q4 = Q4

ÇIKIŞLAR
2D_hata = q1
Q_S2D_Y = q2
Q_S2D_S = q3
Q_S2D_SY = q4
Q_S2D_SS = q5
Q_S2D_SK = q6

```

Şekil 6.12: 2D sinyali mantık fonksiyonları.

Sinyal bloğu tasarımında da aynı makas ve tanzim bloğundaki şekliyle Şekil 6.12’de sinyal otomatının PLC gerçekleştirilmesini sağlayacak olan mantık fonksiyonlarına denk

düŖecek kodların yazılıp bir fonksiyon bloęu içine gömülmesiyle sinyal bloęu tasarımı tamamlanacaktır.

Makas ve tanzim bloęunda açıklanan aynı sebeplerden ötürü en fazla bildirim veren sinyal seçilerek ve bu sinyal bloęunda yer alacak deęişkenlerin lokal olarak tanımlanması şeklinde bir tasarım düşünölebilir. Bu lokal deęişkenlere ilgili sinyale ait global deęişkenler ve ilgili sinyal için gereksiz olan lokal bitlere de gereken duruma göre mantıksal 1 veya 0 atanarak tasarım tamamlanmış olur.

## 7. SİMÜLASYON

### 7.1 Giriş

Bu bölümde tasarlanan ankaşman algoritmasının hata ayıklamasının yapılabilmesi için kullanılan Ahmet KUZU (Mart 2010) tarafından UDSP kapsamında Microsoft Visual C++ ortamında geliştirilen simülatör hakkında bilgi verilecektir. Simülatör bu çalışmada ele alınan örnek hatta uyarlanmıştır. Bu simülatörün tercih edilmesinin sebebi PLC programlama arayüzünün sunmuş olduđu simülatörde fazla giriş çıkış sayısı bulunan programların akışının takip edilmesinin oldukça zor olmasıdır. Görsel ve kullanışlı bir arayüz sunan bu simülatör sayesinde algoritmanın test aşaması hızlı bir şekilde tamamlanabilmektedir.

Simülatör ile PLC arasındaki haberleşme sonraki bölümde hakkında genel olarak bilgi verilecek olan Modbus UDP üzerinden sağlanmaktadır. Adreslenen değışkenler yazılıp okunarak bilgisayarda koşan simülatör ile PLC’de koşan ankaşman algoritması arasındaki haberleşme sağlanmaktadır. Algoritmanın test edilebilmesi için sahadan ve KM’den alınması gereken bilgiler, simülatörün sahayı simüle ederek ankaşman tarafından gönderilen komutlara karşılık ürettiğı indikasyonlar ve KM’yi simüle ederek gönderdiği talepler ile elde edilmektedir.

### 7.2 Modbus

Modbus haberleşme sistemi de Profibus gibi endüstriyel ortamlarda çalışan sistemler arası iletişimi sağlamak için geliştirilmiştir. Modbus sistemi daha basit yapıdaki donanımlarla gerçekleştirilebilir. Çok hızlı olmayan bir haberleşme kullanıldığı için de pek çok cihaz sisteme uyumlu olarak geliştirilebilir. RS232 olarak bilinen standart PC seri portu üzerinden ya da RS485 arabirimi üzerinden ya da Ethernet ağı üzerinden gerçekleştirilebilir. (URL1)

Modbus temelde veri alışverişi yapısını tanımlar. Bu alış verişi gerçekleştirebilecek herhangi fiziksel hat üzerinden de Modbus haberleşme sağlanabilir. Veri alış verişi yapısı halka açık olarak sunulmuştur ve İnternet vb. kaynaklardan kolaylıkla

bulunabilir. Bu yapıya uygun olarak programlanmış haberleşme yeteneği olan her türlü cihaza uygulanabilir. (URL1)

Bu tez çalışmasında HIMA HIMATRIX serisinden bir PLC kullanılmıştır. PLC ile Modbus üzerinden haberleşebilmek için gerekli konfigürasyonlar yapıp değişken adresleri belirlenmelidir. Bunun için Silworx programında “Protocols” menüsünden “Modbus Slave Set” modülü eklemek gerekmektedir. Bu modülün eklenmesinin ardından, PLC’ye giriş ve çıkış olarak tanımlanan değişkenlerin adreslemesinin yapılması gerekmektedir. Bu adresler simülatör ile PLC haberleşmesini sağlayacak bellek alanlarıdır. PLC ve simülatör belli zamanlarda bu alanları okuyup veya bu alanlara yazarak haberleşmektedir.

### **7.3 Kullanıcı Veri Bloğu İletişim Kuralları**

UDP (User Datagram Protocol - Kullanıcı Veri Bloğu İletişim Kuralları), TCP/IP protokol takımının iki aktarım katmanı protokolünden birisidir. Verileri bağlantı kurmadan yollar.

Gelişmiş bilgisayar ağlarında paket anahtarlama bilgisayar iletişiminde bir veri bloğu modu oluşturabilmek için UDP protokolü yazılmıştır. Bu protokol minimum protokol mekanizmasıyla bir uygulama programından diğerine mesaj göndermek için bir prosedür içerir. UDP için genel olarak şu bilgiler verilebilir:

- Geniş alan ağlarında ses ve görüntü aktarımı gibi gerçek zamanlı veri aktarımlarında UDP kullanılır.
- UDP bağlantı kurulum işlemlerini, akış kontrolü ve tekrar iletim işlemlerini yapmayarak veri iletim süresini en aza indirir.
- UDP ve TCP aynı iletişim yolunu kullandıklarında UDP ile yapılan gerçek zamanlı veri transferinin servis kalitesi TCP’nin oluşturduğu yüksek veri trafiği nedeniyle azalır.
- UDP’yi kullanan protokollerden bazıları DNS, TFTP, ve SNMP protokolleridir. Uygulama programcıları birçok zaman UDP’yi TCP’ye tercih eder, çünkü UDP ağ üzerinde fazla bant genişliği kaplamaz.

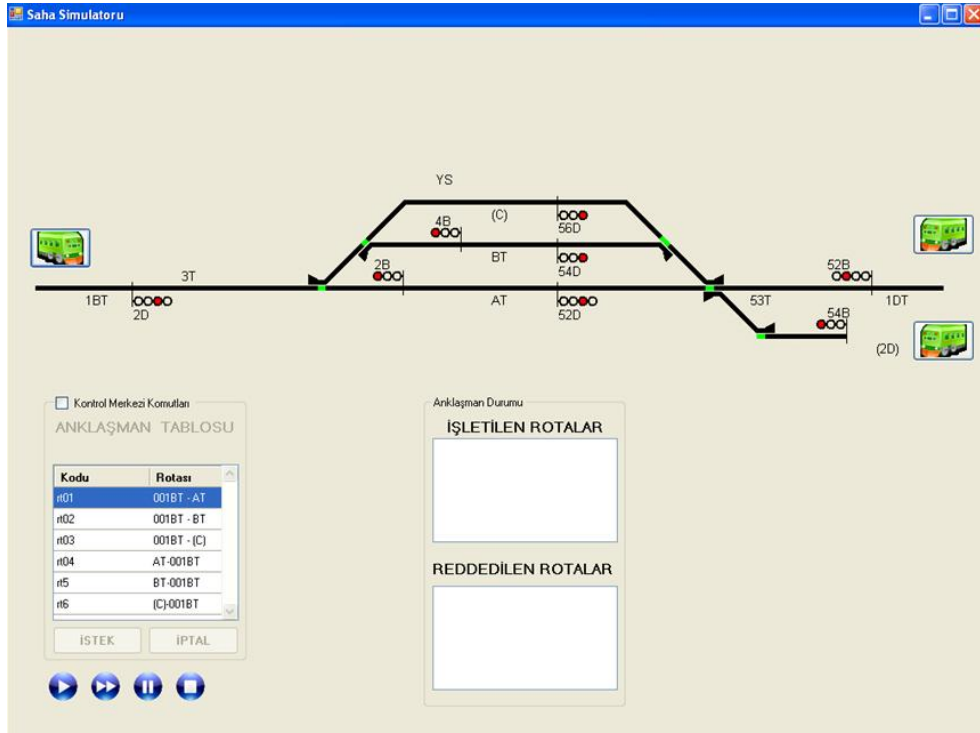
UDP güvenilir olmayan bir aktarım protokolüdür. Ağ üzerinden paketi gönderir ama gidip gitmediğini takip etmez ve paketin yerine ulaşip ulaşmayacağına onay verme



yetkisi yoktur. UDP üzerinden güvenilir şekilde veri göndermek isteyen bir uygulama bunu kendi yöntemleriyle yapmak zorundadır veya TCP protokolü kullanılmalıdır. ( Bölüm 7.3 URL2’den uyarlanmıştır.)

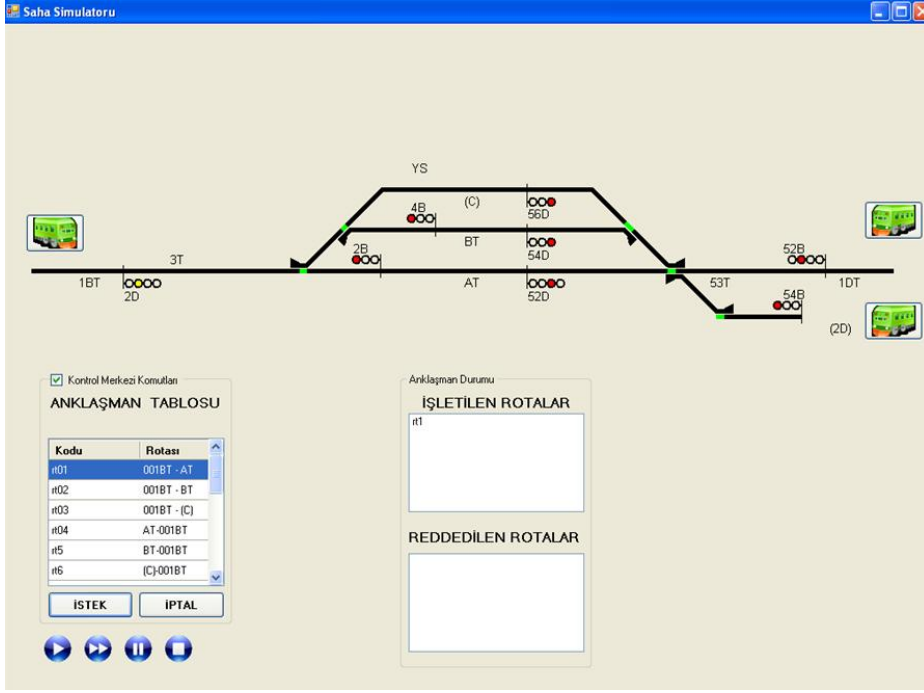
#### 7.4 Simülasyon Arayüzü

Simülasyon arayüzünde KM tarafından gönderilecek olan tanzim taleplerini karşılamak üzere rt1’den başlayıp rt18’ye kadar giden güzergah numaralarından biri işaretlenip “İSTEK” butonuna basılır. Böylece ilgili güzergah talebine denk gelen adrese mantıksal 1 yazılır ve bu adresin PLC tarafından okunmasıyla ilgili güzergah için tanzim talebi alınmış olur. AS, tanzim talebini uygun bulursa, yol tanzim edildikten sonra tren simgesine tıklanarak ilgili güzergaha tren gönderilebilir. Bölüm 6’da ele alınan sahaya ait simülasyon arayüzü Şekil 7.1’de gösterilmektedir.



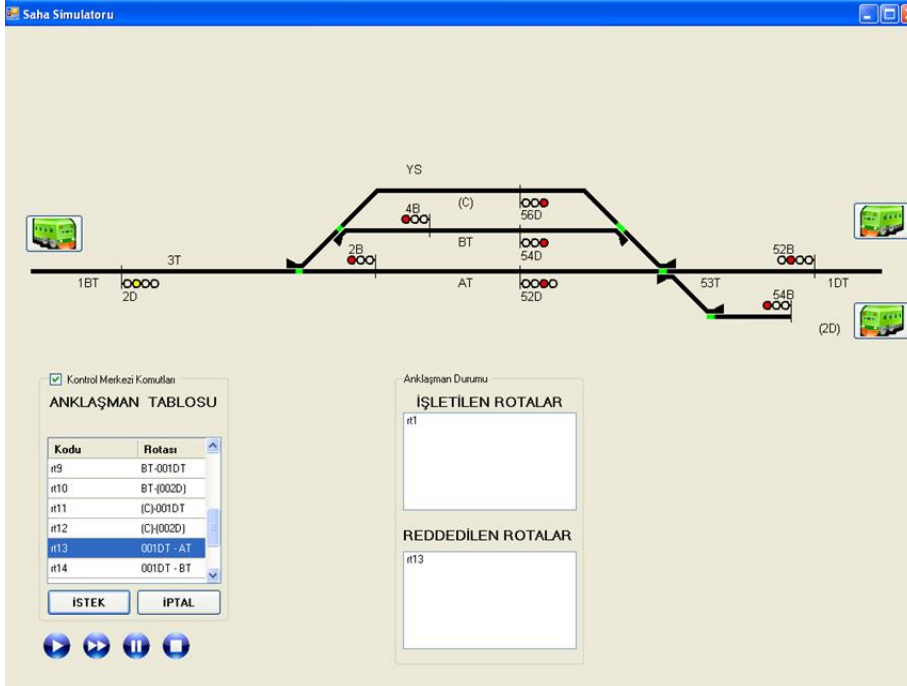
Şekil 7.1 : Simülasyon arayüzü.

Şekil 7.2’de 1 numaralı 1BT-AT güzergahı için tanzim talebi yapılmış ve bu bilgiyi simülasyonara göndererek “İŞLETİLEN ROTALAR” kısmında 1 numaralı güzergahın işletildiği bilgisinin görünmesi sağlanmıştır. Bu talebin AS tarafından uygun bulunmasıyla makaslar uygun konumu aldıktan sonra 2D sinyaline ilgili kontroller sonucu sarı bildirim ver komutu gönderilerek sarı yanması sağlanmış ve 1 numaralı güzergah tanzim edilmiştir.



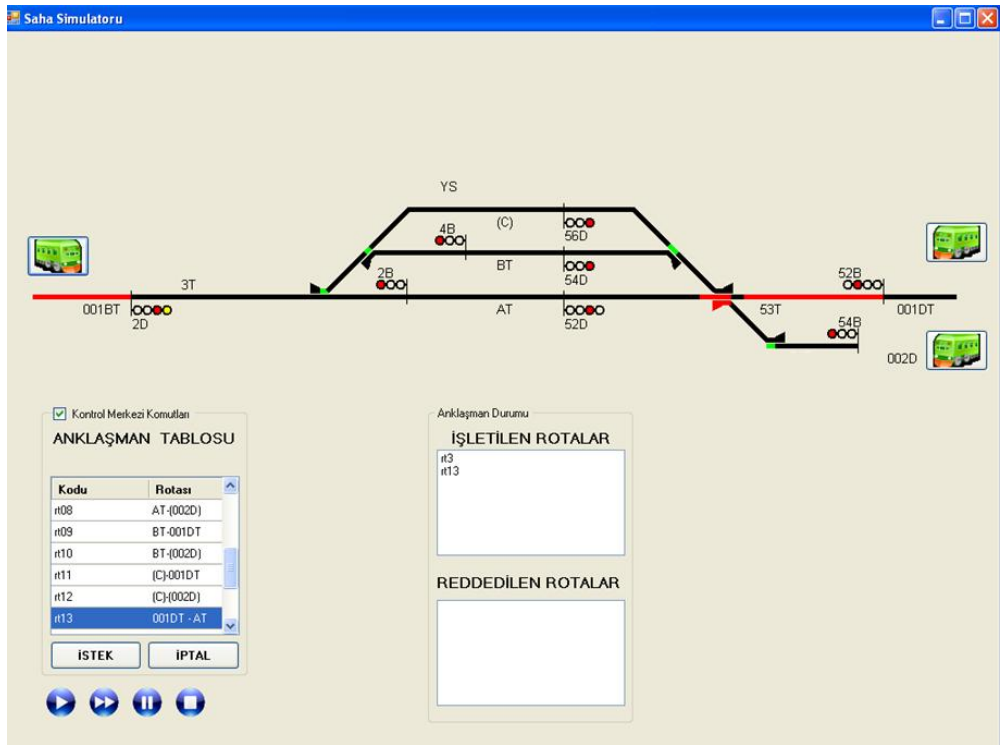
Şekil 7.2 : 1BT – AT güzergahının tanzim edilmesi.

Şekil 7.3’de, 1BT-AT güzergahı Şekil 7.2’de görüldüğü gibi tanzimli iken 1DT-AT güzergahı için tanzim talebinde bulunulmuş ve bu talep, 1DT-AT güzergahının 1BT-AT güzergahı ile çakışan bir güzergah olması sebebiyle AS tarafından uygun bulunmamış ve reddedilmiştir.



Şekil 7.3 : 1DT-AT güzergahının reddedilmesi.

Simülâtörde aynı anda iki güzergahı işletme ve birden fazla tren gönderebilmek mümkündür. Şekil 7.4’de simülâtörde aynı anda iki güzergahın işletilmesi gösterilmiştir. Ray devresi bulunmayan yol ile biten 1BT-(C) güzergahı ile 1DT-AT güzergahı aynı anda işletilmekte ve aynı anda iki tren gönderilmektedir. 1BT- (C) güzergahı sinyali, ray devresi bulunmayan yol için bildirim verdiği için sarı üzeri kırmızı yakılmıştır. Aynı anda 1DT-AT güzergahı da tanzim edilerek güzergah sinyali 52B sinyali sarı bildirimde bulduktan sonra trenin 53T ray devresine girmesiyle simülâtörde görüldüğü üzere 52B sinyali kırmızı bildirim vermeye başlamıştır.



Şekil 7.4 : İki güzergahın aynı anda işletilmesi.



## 8. SONUÇ VE ÖNERİLER

Anklaşman sisteminin en önemli görevi sinyalizasyon sistemlerinin emniyet bütünlüğünü sağlamak ve sistemi hatada güvenli olarak çalıştırmaktır. Bu nedenle anklaşman, sinyalizasyon sistemlerinin güvenliğini garanti altına alan bir sistemdir. Anklaşman sisteminin tasarımında göz önünde bulundurulması gereken kurallar anklaşman tablosunda yer almaktadır. Anklaşman tablosu bu özelliği ile anklaşman sisteminin başarıyla tasarlanabilmesi ve sinyalizasyon sisteminin fonksiyonel güvenliğinin sağlanabilmesi için çok önemlidir.

Bu tez çalışmasında anklaşman tablosundan anklaşman algoritmasının otomat modelini ve bu otomat modelinin PLC'ye uygulanmasını sağlayacak olan sözde kodunu oluşturan bir algoritma tasarımı ve Microsoft Visual Basic uygulaması ele alınmıştır. Örnek bir istasyon için oluşturulan anklaşman tablosu, bu yazılıma giriş olarak alınarak bu istasyon için üretilen anklaşman algoritması ve sözde kodu incelenmiştir. Geliştirilen yazılımdan çıktı olarak elde edilen sözde kod FBD dilinde PLC koduna çevrilip HIMA HIMATRIX serisinden bir PLC'ye yüklenerek, Microsoft Visual C++ ortamında geliştirilmiş simülatör ile test edilmiştir.

Otomat modelinden otomatik olarak sözde PLC kodu elde etmek için matematiksel denkleme dayanan bir gerçekleştirme yöntemi tanıtılmıştır. Bu yöntem sayesinde uygulamada sıklıkla karşılaşılan literatürde çığ etkisi olarak adlandırılan sorun çözülmektedir. Ancak tanıtılan bu yöntemde gerçekleştirimin problemlili olduğu örnekler verilmiş ve problemin kaynağı olan başlangıç durumuna kurma irdelemesi yapılarak bu problemin çözümü için yeni bir yaklaşım önerilmiştir.

Bu çalışma ile literatüre yapılan katkı otomatları baz alan bir anklaşman algoritma ve sözde kodu üretici geliştirmiş olmaktadır. Yapılan literatür çalışması sonucu elde edilen kanı, raylı sistemler için Petri temelli otomatik algoritma üreteçleri olmasına karşın otomatları baz alan böyle bir sistemin olmamasıdır.

Bu çalışmanın devamında yapılacak çalışmalarda çıktı olarak üretilen sözde kodun uygun bir PLC programlama diline otomatik olarak çevrilerek PLC içine direk alınmasıyla hatayı daha da aza indirmek amaçlanabilir.



## KAYNAKLAR

- Cassandras, Christos G. and Lafortune, Stephane**, 1999: Introduction To Discrete Event Systems, Kluwer Academic Publishers, Boston/Dordrecht/London.
- Goddard, E.**, 2008: Overview of signalling and train control systems, Electric Traction Systems, *IET Professional Development Course on*, s: 314 – 322.
- Gündoğdu, F. ve Söyler H.**, 2005: Raylı sistemlerde emniyet standartları ve makas otomasyon sistemine uygulanması, *Elektrik – Elektronik – Bilgisayar Mühendisliği 11. Ulusal Kongresi ve Fuarı*, İstanbul, 22– 25 Eylül.
- Gündoğdu, F. ve Söyler H.**, 2008: Demiryolu sinyalizasyon sistemlerinde tasarım kriterleri ve fail-safe kavramı , *Kentiçi Raylı Sistemler Bülteni*, sayı 8.
- Hasdemir T.**, 2008: Doktora Tezi: Ayırık olay sistemlerinin tasarımı için yeni bir gerçekleştirme ve otomatik kod üretme yöntemi, İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü, Eylül.
- Özdemir S.**, 2000: Yüksek Lisans Tezi: Demiryollarında sinyalizasyon, Kocaeli Üniversitesi Fen Bilimleri Enstitüsü, Ocak.
- Söyler H., Açıkbaş S.**, 2005: Raylı toplu taşımada sinyalizasyon sistemleri, *Elektrik – Elektronik – Bilgisayar Mühendisliği 11. Ulusal Kongresi ve Fuarı*, İstanbul, 22– 25 Eylül.
- Tiryaki Ö.**, 2007: Yüksek Lisans Tezi: Biçimsel dillerden endüstriyel işlemcilere otomatik kod üretme : Otomat yaklaşımı, İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü, Ocak.
- Tombs, D., Robinson N. and Nikandros G.**, 2002: Signalling control table generation and verification, *Conference on Railway Engineering Wollongong*, s: 10-13.
- Türk S., Sonat A., Kuzu A., Söylemez M.T., Songüler Ö.**, 2010: Raylı ulaşım sinyalizasyon sistemleri için anlaşılan tablosundan otomatik anlaşılan algoritması üretici, *Otomatik Kontrol Ulusal Toplantısı 2010(gönderildi)*, 21-23 Eylül.
- Url-1** <<http://www.mrs.com.tr/teknikbilgi/modbus.htm>>, alındığı tarih 29.04.2010.
- Url-2** <<http://cekirdek.uludag.org.tr/~meren/belgeler/udp/udp.htm>>, alındığı tarih 29.04.2010.





## ÖZGEÇMİŞ

**Ad Soyad:** Serhat Türk

**Doğum Yeri ve Tarihi:** Eskişehir 22.05.1985

**Lisans Üniversitesi:** Elektrik Mühendisliği, Yıldız Teknik Üniversitesi

### **Yayın Listesi:**

- **Türk S.**, Sonat A., Kuzu A., Söylemez M.T., Songüler Ö., 2010: Raylı ulaşım sinyalizasyon sistemleri için anlaşıman tablosundan otomatik anlaşıman algoritması üretici, *Otomatik Kontrol Ulusal Toplantısı 2010 (gönderildi)*, 21-23 Eylül, Kocaeli, Türkiye.